IBM DB2 Universal Database

Extensions Image, Audio et Vidéo Administration et programmation

Version 7

SC11-1682-00

IBM DB2 Universal Database

Extensions Image, Audio et Vidéo Administration et programmation

Version 7

SC11-1682-00

Important

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations générales figurant à l'«Annexe C. Remarques» à la page 605.

Réf. US : SC26-9929-00

LE PRESENT DOCUMENT EST LIVRE "EN L'ETAT". IBM DECLINE TOUTE RESPONSABILITE, EXPRESSE OU IMPLICITE, RELATIVE AUX INFORMATIONS QUI Y SONT CONTENUES, Y COMPRIS EN CE QUI CONCERNE LES GARANTIES DE QUALITE MARCHANDE OU D'ADAPTATION A VOS BESOINS. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. Les informations qui y sont fournies sont susceptibles d'être modifiées avant que les produits décrits ne deviennent eux-mêmes disponibles. En outre, il peut contenir des informations ou des références concernant certains produits, logiciels ou services non annoncés dans ce pays. Cela ne signifie cependant pas qu'ils y seront annoncés.

Pour plus de détails, pour toute demande d'ordre technique, ou pour obtenir des exemplaires de documents IBM, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial.

Vous pouvez également consulter les serveurs Internet suivants :

- http://www.fr.ibm.com (serveur IBM en France)
- http://www.can.ibm.com (serveur IBM au Canada)
- http://www.ibm.com (serveur IBM aux Etats-Unis)

Compagnie IBM France
Direction Qualité
Tour Descartes
92066 Paris-La Défense Cedex 50

Table des matières

Figures

Tableaux

Avis aux lecteurs canadiens

Le présent document a été traduit en France. Voici les principales différences et particularités dont vous devez tenir compte.

Illustrations

Les illustrations sont fournies à titre d'exemple. Certaines peuvent contenir des données propres à la France.

Terminologie

La terminologie des titres IBM peut différer d'un pays à l'autre. Reportez-vous au tableau ci-dessous, au besoin.

IBM France	IBM Canada
ingénieur commercial	représentant
agence commerciale	succursale
ingénieur technico-commercial	informaticien
inspecteur	technicien du matériel

Claviers

Les lettres sont disposées différemment : le clavier français est de type AZERTY, et le clavier français-canadien de type QWERTY.

OS/2 et Windows - Paramètres canadiens

Au Canada, on utilise :

- les pages de codes 850 (multilingue) et 863 (français-canadien),
- le code pays 002,
- le code clavier CF.

Nomenclature

Les touches présentées dans le tableau d'équivalence suivant sont libellées différemment selon qu'il s'agit du clavier de la France, du clavier du Canada ou du clavier des États-Unis. Reportez-vous à ce tableau pour faire

correspondre les touches françaises figurant dans le présent document aux touches de votre clavier.

France	Canada	Etats-Unis
(Pos1)		Home
Fin	Fin	End
(PgAr)		PgUp
(PgAv)		PgDn
Inser	Inser	Ins
Suppr	Suppr	Del
Echap	Echap	Esc
Attn	Intrp	Break
Impr écran	ImpEc	PrtSc
Verr num	Num	Num Lock
Arrêt défil	Défil	Scroll Lock
(Verr maj)	FixMaj	Caps Lock
AltGr	AltCar	Alt (à droite)

Brevets

Il est possible qu'IBM détienne des brevets ou qu'elle ait déposé des demandes de brevets portant sur certains sujets abordés dans ce document. Le fait qu'IBM vous fournisse le présent document ne signifie pas qu'elle vous accorde un permis d'utilisation de ces brevets. Vous pouvez envoyer, par écrit, vos demandes de renseignements relatives aux permis d'utilisation au directeur général des relations commerciales d'IBM, 3600 Steeles Avenue East, Markham, Ontario, L3R 9Z7.

Assistance téléphonique

Si vous avez besoin d'assistance ou si vous voulez commander du matériel, des logiciels et des publications IBM, contactez IBM direct au 1 800 465-1234.

Avant-propos

Le présent manuel décrit l'utilisation de DB2 Extensions pour la préparation et la gestion d'une base de données DB2® pour des données de type image, audio ou vidéo. Il décrit également comment utiliser les fonctions définies par l'utilisateur (UDF) et les interfaces de programmation d'applications (API) fournies par DB2 Extensions pour l'accès et la manipulation de ces types de données. L'incorporation des UDF dans les instructions SQL de votre programme et des API vous permet d'accéder à des types de données non traditionnelles telles que les images et les clips vidéo ainsi qu'à des données numériques et de type caractère classiques.

Dans le présent manuel, les références à ″DB2″ se rapportent à DB2 UDB.

Lecteurs concernés

Le présent manuel est destiné aux administrateurs de bases de données DB2 familiarisés avec les concepts, les outils et les techniques d'administration de DB2.

Il s'adresse également aux programmeurs d'applications DB2 familiarisés avec le langage SQL et un ou plusieurs langages de programmation pouvant être utilisés pour les programmes d'application DB2.

Il s'adresse aux utilisateurs d'extensions DB2 Image, Audio et Vidéo. Les utilisateurs de l'extension Texte doivent se reporter au manuel *Extension Texte DB2 - Administration et programmation*.

Structure du manuel

Le présent manuel contient les parties suivantes :

«Partie 1. Introduction»

Cette partie constitue une vue d'ensemble des Extensions DB2. Consultez-la si vous n'êtes pas familiarisé avec l'administration et la programmation à l'aide des Extensions DB2.

«Partie 2. Administration des données image, audio et vidéo»

Cette partie décrit la préparation et la gestion pour les données de type image, audio, et vidéo. Consultez-la si vous souhaitez administrer une base de données DB2 contenant des données de type image, audio ou vidéo.

«Partie 3. Programmation pour des données image, audio ou vidéo»

Cette partie indique comment utiliser les API et les UDF de DB2 Extensions pour demander des opérations sur des données de type image, audio ou vidéo. Consultez-la si vous souhaitez accéder à des données image, audio ou vidéo et les manipuler dans un programme d'application DB2.

«Partie 4. Références»

Cette partie présente les informations de référence relatives aux UDF et API de DB2 Extensions, aux commandes de gestion et aux informations de diagnostic telles que les messages et les codes. Consultez-la si vous êtes familiarisé avec les concepts et les tâches des extensions DB2 et que vous souhaitez obtenir des informations sur une UDF ou une API DB2 Extensions, une commande, un message ou un code.

«Annexes»

Les annexes décrivent :

- Comment définir les variables d'environnement utilisées par les extensions DB2 afin de trouver les fichiers et identifier les programmes d'affichage ou de lecture d'objets image, audio et vidéo.
- Comment installer et utiliser les modèles de programmes et les fichiers de support fournis avec les extensions.

Informations relatives aux plateformes

DB2 Extensions peut être utilisé conjointement avec l'environnement de bases de données monopartition de DB2 Universal Database, ou avec l'environnement de bases de données multipartition de DB2 Universal Database Enterprise Extended Edition.

Le présent manuel contient les informations relatives à l'utilisation des extensions DB2 dans les deux environnements. Les informations se rapportant à la seule utilisation des extensions dans l'environnement multipartition de DB2 Universal Database Enterprise Extended Edition sont signalées par la mention "**Produit EEE uniquement**". Les informations se rapportant à la seule utilisation des extensions dans l'environnement monopartition de DB2 Universal Database sont signalées par la mention "**Produit non-EEE uniquement**". Les informations ne se rapportant à aucun environnement particulier s'appliquent aux deux environnements.

Conventions de mise en évidence

Les conventions ci-après sont utilisées :

Gras Indique la définition d'un nouveau terme.

Italique
Indique les paramètres de variable que vous devez remplacer par une valeur ou met en relief certains mots utilisés dans le texte.

MAJUSCULES
Signalent les éléments suivants :
- Types de données
- Noms de répertoire
- Nom de zones
- Appels API
- Commandes
- Mots clés
- Noms de variable

Exemple
Le texte des exemples présente un message système, une valeur que vous entrez ou des exemples de codage.

Lecture des diagrammes de syntaxe

Tout au long de ce manuel, les commandes et la syntaxe SQL sont présentées à l'aide de diagrammes de syntaxe. Ils se lisent de la manière suivante :

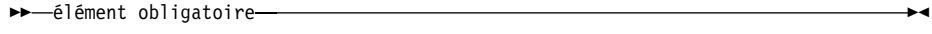
- Les diagrammes de syntaxe se lisent de la gauche vers la droite et du haut vers le bas. Ils figurent après le chemin d'accès.

  Le symbole ▸▸— indique le début d'une instruction.

  Le symbole —▸ indique que la fin de la syntaxe figure sur la ligne suivante.

  Le symbole ▸— indique que le début d'une instruction figure sur la ligne précédente.

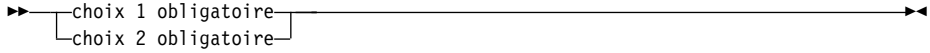
  Le symbole —▸◂ indique la fin d'une instruction.
- Les éléments obligatoires apparaissent sur la ligne horizontale (chemin principal).



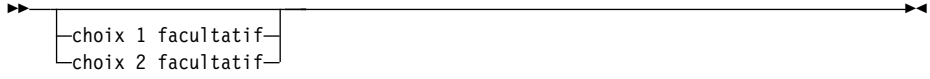
- Les éléments facultatifs apparaissent au-dessous du chemin principal.

- Si vous pouvez choisir entre plusieurs éléments, ceux-ci apparaissent les uns au-dessus des autres.

  Si vous *devez* choisir l'un de ces éléments, un élément de la pile apparaît dans le chemin principal.



  Si l'absence d'élément est admise, l'ensemble de la pile apparaît en-dessous du chemin principal.



  Une flèche de répétition au-dessus d'une pile indique que vous pouvez choisir plusieurs éléments de la pile.

- Les mots clés apparaissent en majuscules, (par exemple, `/DB2IMAGE:`). L'orthographe de ces mots clés doit être respectée. Les variables apparaissent en minuscules (par exemple, `srcpath`). Elles représentent les noms ou les valeurs fournis dans la syntaxe.
- Les marques de ponctuation, parenthèses, opérateurs arithmétiques et tout autre symbole présentés font également partie de la syntaxe.

Informations connexes

DB2 Universal Database

Les manuels *Mise en route* (GC11-1648 pour OS/2®, GC11-1651 pour Windows® et GC11-1650 pour UNIX) décrivent la planification et l'installation de DB2 sur la plateforme appropriée.

Les manuels *DB2 Universal Database Enterprise-Extended Edition - Mise en route* (GC11-1644 pour AIX® et GC11-1643 pour Windows) décrivent la planification, l'installation et la configuration de DB2 Universal Database Enterprise Extended Edition sur la plateforme appropriée.

Le manuel *Administration Guide: Planning*, volume 1 (SC09-2946) présente les concepts de bases de données, contient des informations

relatives aux opérations de conception (par exemple, la conception de bases de données logiques et physiques) et aborde la notion de haute disponibilité.

Le manuel *Administration Guide: Implementation*, volume 2 (SC09-2944) contient des informations sur l'implémentation des opérations de conception, l'accès aux bases de données, le contrôle, la sauvegarde et la récupération de données.

Le manuel *Administration Guide: Performance*, volume 3 (SC09-2945) contient des informations sur l'environnement de bases de données ainsi que sur l'évaluation et l'adaptation des performances des applications.

Le manuel *Application Development Guide* (SC09-2949) décrit comment développer des applications accédant à des bases de données DB2 à l'aide d'instructions SQL ou JDBC imbriquées. Il explique également comment créer des procédures mémorisées et des fonctions définies par l'utilisateur, comment spécifier des types définis par l'utilisateur et comment utiliser les déclencheurs.

Le manuel *CLI Guide and Reference* (SC09-2950) décrit comment développer des applications accédant à des bases de données DB2 à l'aide de l'interface DB2 Call Level, qui est une interface SQL compatible avec la norme ODBC de Microsoft.

Le manuel *Command Reference* (SC09-2951) décrit comment utiliser l'Interpréteur de commandes DB2 et fournit des informations de référence sur les commandes DB2.

Le manuel *Guide des messages* (GC11-1653 et GC11-1654) répertorie les messages et les codes utilisés par DB2, et en fournit les explications. Il indique également les actions à entreprendre pour résoudre les incidents correspondants.

DB2 Universal Database Extension Texte

Le manuel *DB2 UDB Extension Texte - Administration et programmation, version 7* (SC11-1683) indique comment administrer une base de données DB2 pour des données textuelles. Il décrit également l'utilisation des interfaces de programmation d'applications (API) fournies par l'extension Texte de DB2 pour l'accès et la manipulation de ce type de données.

DB2 Universal Database Extension XML

Le manuel *DB2 UDB Extension XML - Administration et programmation* indique comment administrer une base de données DB2 pour des document XML. Il décrit également l'utilisation des interfaces de programmation d'applications (API) fournies par l'extension XML de DB2 pour l'accès et la manipulation de ce type de données.

DB2 UDB Extension Spatiale

Le manuel *Extension Spatiale - Guide d'utilisation et de référence* (SC11-1684) contient des informations sur l'installation, la configuration, l'administration, la programmation et l'identification des incidents de l'extension Spatiale. Il décrit également les concepts des données spatiales et offre des informations de référence (messages et SQL) propres à cette extension.

DB2 Universal Database pour OS/390 Extensions Image, Audio et Vidéo

Le manuel *DB2 Universal Database for OS/390 Version 6 Image, Audio, and Video Extenders Administration and Programming* (SC26-9650) décrit comment administrer un serveur de bases de données DB2 pour OS/390 pour les données image, audio et vidéo. Il décrit également comment utiliser des fonctions définies par l'utilisateur et les API (interfaces de programmation d'applications) fournies par l'extension image, audio et vidéo de DB2 pour OS/390 pour l'accès et la manipulation des données de type image, audio et vidéo.

DB2 Universal Database Extension Texte pour OS/390

Le manuel *DB2 Universal Database for OS/390 Version 6 Text Extender Administration and Programming* (SC26-9651) contient les informations relatives à l'administration d'un serveur de bases de données DB2 pour OS/390 pour des données de type texte. Il décrit également comment utiliser les interfaces de programmation d'applications fournies par l'extension Texte de DB2 pour OS/390 pour l'accès et la manipulation des données de type texte.

World Wide Web

Le site Web de DB2 Extensions contient des informations sur DB2 Extensions et sur les technologies associées. L'adresse URL de la page d'accueil DB2 Extensions est la suivante :

http://www.ibm.com/software/data/db2/extenders

Remarques du lecteur

Vos commentaires nous permettent d'améliorer la qualité de nos documents. Nous vous serions reconnaissants de nous faire part de vos observations au sujet du présent manuel ou de toute autre documentation relative aux Extensions DB2. Vous disposez des méthodes suivantes :

- Envoi de vos commentaires par le Web. Consultez le site Web à l'adresse :

  http://www.ibm.com/software/data/db2/extenders

  Le site Web propose une page destinée à recueillir vos commentaires et à les envoyer.
- Envoyez vos commentaires par e-mail à comments@vnet.ibm.com. N'oubliez pas d'indiquer le nom du produit, son numéro de version, et les nom et numéro de référence du manuel s'il y a lieu. Si vos commentaires

concernent une section en particulier, indiquez précisément où se trouve le texte en question (par exemple, un chapitre, un titre de section, un numéro de tableau ou de page, ou le titre d'une rubrique d'aide).

- Envoyez vos remarques par courrier à l'adresse suivante :

  IBM Corporation,
  Department HHX/H3
  P.O. Box 49023
  San Jose, CA 95161-9023
  U.S.A.

- Envoyez vos commentaires par télécopie au 800-426-7773 (aux Etats-Unis et au Canada).
- Transmettez vos commentaires à votre partenaire commercial IBM.

IBM pourra disposer comme elle l'entendra des informations contenues dans vos commentaires, sans aucune obligation de sa part. Il va de soi que ces informations pourront continuer à être utilisées par leur auteur.

Pour plus d'informations sur la façon de prendre contact avec IBM, reportez-vous à la section «Comment prendre contact avec IBM» à la page 625.

Partie 1. Introduction

Chapitre 1. Présentation

DB2 (DB2) Universal Database (UDB) est un système de gestion de bases de données relationnelles très puissant. Il permet le stockage et la protection des données alphanumériques classiques, aussi bien que des objets LOB (objets complexes). Les extensions DB2 vous aident à prendre en charge les caractéristiques orientées objet de DB2. Ces extensions définissent des types de données et des fonctions spécifiques pour la gestion des objets de type image, audio, vidéo et texte, vous épargnant ainsi le temps et les efforts nécessaires à leur définition. Ces types de données prédéfinis étant utilisables au moyen d'instructions SQL, vos applications disposent d'un point d'accès unique à l'ensemble de ces types de données, ainsi qu'aux types classiques binaires ou alphanumériques. En outre, les Extensions vous permettent d'utiliser de nouvelles techniques pour la recherche des données. Par exemple, les applications peuvent rechercher des images sur la base de leurs caractéristiques visuelles en mettant à votre disposition tout un échantillonnage de couleurs et de textures.

Exploitation des fonctions DB2

Les extensions DB2 tirent parti des fonctions orientées objet de DB2. DB2 vous permet, plus particulièrement, d'effectuer les opérations suivantes :

- Stockage d'objets LOB d'une taille pouvant atteindre 2 Go dans une base de données DB2.
- Définition de types de données distincts pour ces objets LOB. Ces types de données utilisateur (types UDT) permettent l'identification des données représentées par un objet, par exemple, une image ou un enregistrement audio.
- Définition de fonctions spécifiques susceptibles d'être appelées pour un type UDT. Par exemple, vous pouvez définir une fonction destinée au calcul du nombre de couleurs d'une image ou à celui de la fréquence d'échantillonnage d'un enregistrement audio. Ces fonctions utilisateur (UDF) sont appelées à l'aide d'instructions SQL, de la même manière que les fonctions SQL classiques.

Les extensions DB2 créent un ensemble de types UDT et de fonctions UDF spécifiques aux objets de type image, audio, vidéo et texte, qui vous seront d'une aide précieuse pour les éléments suivants :

- Développement d'applications. Les types de données et les fonctions étant prédéfinis par les Extensions, vous n'aurez pas à les redéfinir dans vos nouvelles applications.

- Cohérence. Le même ensemble de types UDT et de fonctions UDF étant disponible pour toutes vos applications, les extensions vous permettent de bénéficier d'un niveau de cohérence immédiat.
- Création de requêtes efficaces. L'appel des fonctions UDF s'effectuant de la même façon que pour les autres fonctions SQL, vos applications peuvent comporter des requêtes portant sur plusieurs types de données. La même instruction SQL permet ainsi d'accéder aussi bien à des objets de type image, audio, vidéo et texte qu'à des données alphanumériques classiques. Les types UDT et les fonctions UDF peuvent être spécifiés dans des instructions SQL imbriquées, ainsi que dans les appels de l'interface DB2 CLI (DB2 Call Level Interface).

Par ailleurs, étant donné que les objets LOB peuvent être stockés dans une base de données DB2, vous disposez des mêmes garanties en termes de sécurité, d'intégrité et de récupération que dans le cas de données classiques.

De plus, les extensions DB2 tirent parti de l'environnement de bases de données partitionnées de DB2 Universal Database Enterprise - Extended Edition. Le partitionnement permet aux applications d'utiliser une base de données trop volumineuse pour un seul ordinateur. Il permet également l'exécution simultanée d'opérations SQL, ce qui accélère l'exécution des requêtes et utilitaires SQL.

Nouvelles techniques de recherche

DB2 Extensions offre à vos applications une souplesse exceptionnelle pour la recherche des informations. Les objets LOB peuvent être recherchés à l'aide des données alphanumériques qui leur sont associées dans les bases de données. Une séquence audio peut ainsi être retrouvée au moyen de sa description ou de sa date d'enregistrement. Les applications peuvent également rechercher les objets à l'aide de critères propres à leurs caractéristiques intrinsèques, tels que la durée d'une séquence vidéo. Les Extensions déterminent et stockent automatiquement ces caractéristiques, pour en permettre l'utilisation lors des recherches.

Les applications peuvent également rechercher des objets graphiques en fonction de leur contenu. Elles disposent en effet d'un ensemble d'échantillons visuels, permettant à l'utilisateur de retrouver des images présentant les mêmes caractéristiques que l'exemple sélectionné, en termes de couleur ou de texture. La fonction QBIC (Query by Image Content) de DB2 Extensions permet d'effectuer ce type de recherche.

DB2 Extensions

La famille de produits DB2 Extensions comporte une extension Image, une extension Audio, une extension Vidéo et une extension Texte.

Le présent manuel est consacré aux extensions Image, Audio et Vidéo. Les termes «extensions» ou «extensions DB2» utilisés dans le présent manuel désignent les extensions Image, Audio et Vidéo, sauf indication contraire. Pour plus de détails sur l'extension Texte, reportez-vous au manuel *Extension Texte - Administration et programmation*. Pour plus de détails sur l'extension XML, reportez-vous au manuel *DB2 UDB Extension XML - Administration et programmation*.

SDK et environnement d'exécution

Le module d'installation de DB2 Extensions fournit le SDK (Software Developers Kit) ainsi que les environnements d'exécution client et serveur. Vous pouvez développer des applications DB2 Extensions sur tout poste client ou serveur sur lequel vous avez installé le SDK de DB2 Extensions.

Vous pouvez exécuter les applications DB2 Extensions sur un poste serveur doté du code d'exécution client et serveur de DB2 Extensions. (Le code d'exécution client est automatiquement installé lorsque le code d'exécution serveur est installé.) Vous pouvez également exécuter des applications DB2 Extensions sur un poste client doté du code d'exécution client de DB2 Extensions. Dans ce cas, vous devez vous assurer que la connexion entre ce client et le serveur peut être établie.

Utilisation des extensions

Les fonctions UDF peuvent être appelées à partir d'un programme d'applications DB2 ou de façon interactive à l'aide de l'interpréteur de commandes DB2.

Les extensions fournissent également les interfaces de programmation d'applications (API) suivantes :

- des API d'administration pour la préparation et la maintenance d'une base de données pour les données de type image, audio, et vidéo,
- des API graphiques et de lecture pour l'affichage des images et la lecture des séquences vidéo et audio,
- des API QBIC pour la préparation des images en vue de recherches portant sur leur contenu et pour le lancement de ces recherches (une recherche sur le contenu des images peut également être lancée au moyen des fonctions UDF),
- des API de détection de prises vidéo, pour l'identification de séquences d'images en fonction des changements de plan dans une vidéo.

Les extensions DB2 comportent également un interpréteur de commandes pour l'exécution de commandes d'administration. Afin d'éviter la confusion entre l'interpréteur de commandes fourni par les Extensions et celui fourni

par DB2, nous utiliserons le terme «interpréteur de commandes db2ext» pour le premier, et le terme «interpréteur de commandes DB2» pour le second.

Exemples

Une agence de publicité archive l'ensemble de ses réalisations publicitaires dans une base de données DB2. Jusqu'à présent, seules les données alphanumériques relatives à chaque campagne publicitaire pouvaient être stockées (nom du client, date de lancement de la campagne, etc.). L'installation de DB2 UDB et des Extensions DB2 permet à présent à l'agence d'archiver également le contenu des publicités dans la base de données. Les objets ainsi stockés comprennent notamment les affiches publicitaires et les encarts publiés dans la presse, les films diffusés à la télévision et les flashs radiophoniques. Comme l'illustre la figure 1 à la page 7, toutes les données publicitaires connexes sont regroupées dans une table appelée ADS.

Figure 1. Table de base de données multimédia. Cette table contient des données de type image, audio et vidéo, ainsi que des données alphanumériques classiques. Une séquence vidéo, un flash audio et une image sont représentés sur la figure.

Exemple 1 : Recherche d'une séquence vidéo en fonction de ses caractéristiques

L'un des concepteurs de l'agence souhaite visionner les films publicitaires réalisés pour le compte d'IBM en 1997, mais limiter sa recherche aux films d'une durée inférieure ou égale à 30 secondes.

La figure 2 à la page 8, illustre une requête permettant d'accéder aux séquences vidéo. Cette requête fait appel aux fonctions UDF `Filename` et `Duration` de l'extension Vidéo.

Exemples

```
SELECT FILENAME(ADS_VIDEO)
       FROM ADS
WHERE CLIENT='IBM' AND
SHIP_DATE>='01/01/1997' AND
DURATION(ADS_VIDEO) <=30
```

Figure 2. Requête accédant à des séquences vidéos.

La requête renvoie le nom des fichiers répondant aux critères indiqués. Il ne reste plus au publicitaire qu'à visionner le contenu de chaque film à l'aide du lecteur de son choix.

La figure 2 présente un exemple de requête susceptible d'être émise de façon interactive par le gestionnaire du compte. Toutefois, l'utilisateur fera plus généralement appel à un programme d'application pour la recherche et la lecture des séquences vidéos. Par exemple, la figure 3 illustre certains éléments essentiels d'une telle application codée en C. Cette application permet d'extraire le nom des fichiers à l'aide d'une variable DB2 appelée `hvVid_fname`. Elle fait également appel à une API graphique, appelée `DBvPlay`, pour la lecture des séquences vidéo.

```
#include <dmbvideo.h>

int count = 0;


EXEC SQL BEGIN DECLARE SECTION;
char hvClient[30];                      /*nom du client*/
char hvCampaign[30];                    /*nom de la campagne*/
char hvSdate[8];                        /*date de livraison*/
char hvVid_fname [251]                  /*nom du fichier vidéo*/
  EXEC SQL END DECLARE SECTION;


EXEC SQL DECLARE c1 CURSOR FOR
     SELECT CLIENT, CAMPAIGN, SHIP_DATE, FILENAME(ADS_VIDEO)
       FROM ADS
       WHERE CLIENT='IBM' AND
         SHIP_DATE≥'01/01/1997' AND
         DURATION(ADS_VIDEO)≤30
FOR FETCH ONLY;
```

Figure 3. Application accédant à des séquences vidéo et permettant leur lecture (Numéro 1 de 2)

```
EXEC SQL OPEN c1;
for (;;){
        EXEC SQL FETCH c1 INTO :hvClient, :hvCampaign,
                         :hvSdate, :hvVid_fname;

         if (SQLCODE != 0)
              break;

    printf("\nRecord %d:\n", ++count);
    printf("Client = '%s'\n", hvClient);
    printf("Campaign = '%s'\n", hvCampaign);
    printf("Sdate = '%s'\n", hvSdate);

    rc=DBvPlay(NULL,MMDB_PLAY_FILE,hvVid_fname,MMDB_PLAY_WAIT);
}
EXEC SQL CLOSE c1;
```

Figure 3. Application accédant à des séquences vidéo et permettant leur lecture (Numéro 2 de 2)

Exemple 2 : Recherche d'images en fonction de leur contenu

Un graphiste participe à la création d'une nouvelle affiche publicitaire pour un client. Il souhaite utiliser en arrière-plan une tonalité de bleu spécifique, et veut savoir si cette couleur a déjà été utilisée pour des réalisations précédentes. Pour cela, il exécute une application permettant d'effectuer des recherches portant sur le contenu d'images. Les images sont stockées dans une table de base de données (voir figure 1 à la page 7). Le graphiste est invité à indiquer un échantillon visuel, c'est-à-dire une image représentant la couleur recherchée. L'application analyse alors la couleur choisie et recherche les images dont la couleur s'en rapproche le plus.

La figure 4 à la page 10 illustre la fenêtre d'interrogation dans laquelle apparaît l'échantillon visuel et les résultats de la recherche, qui regroupent les images répondant le mieux au critère indiqué.

Exemples

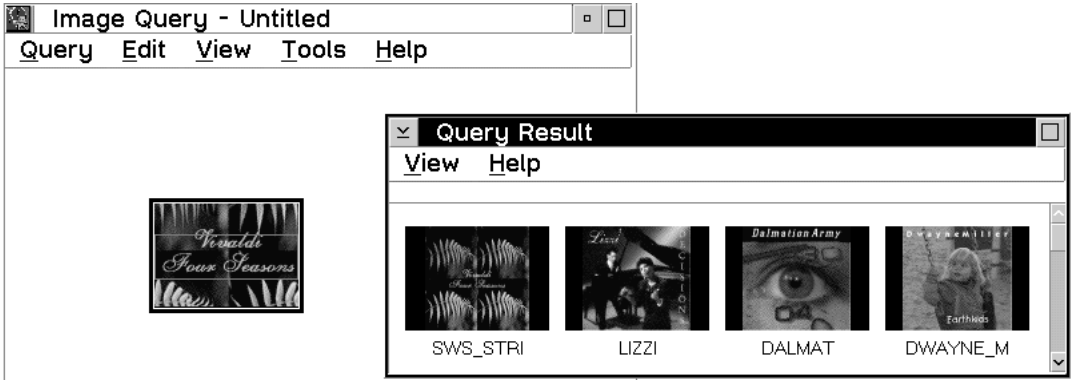


Figure 4. Recherche d'images en fonction de leur contenu. Les images dont la couleur est proche de l'échantillon proposé sont extraites de la base de données.

La figure 5 à la page 11, illustre certains éléments essentiels de l'application utilisée. Cette dernière fait appel à une API QBIC appelée `QbQueryCreate` pour la création d'une requête QBIC, aux API `QbQueryAddFeature` et `QbQuerySetFeatureData` pour l'ajout de la couleur sélectionnée à la requête, à l'API `QbQuerySearch` pour le lancement de la requête, et à l'API `QbQueryDelete` pour sa suppression. L'application utilise également une API graphique, appelée `DBiBrowse`, pour l'affichage des images extraites.

```
#include <dmbqbqpi.h>

#define MaxQueryReturns  10

static SQLHENV henv;
static SQLHDBC hdbc;
static SQLHSTMT hstmt;
static SQLRETURN rc;

void main(int argc, char* argv[])
{
 char           line[4000];
 char*          handles[MaxQueryReturns];
 QbQueryHandle  qHandle=0;
 QbResult       results[MaxQueryReturns];
 SQLINTEGER     count;
 SQLINTEGER     resultType=qbiArray;

 SQlAllocEnv(&henv);
 SQLAllocConnect(henv, &hdbc);
 rc = SQLConnect(hdbc, (SQLCHAR*)"qtest", SQL_NTS,
              (SQLCHAR*)"", SQL_NTS, (SQLCHAR*)"", SQL_NTS);

 if (argc !=2) {
 printf("usage: query colorname\n");
          exit(1);
 }

 QbImageSource is;
 is.type = qbiSource_AverageColor;

 /* run the get color subroutine */
 getColor(argv[1], is.average.Color);

 QbQueryCreate(&qhandle);
 QbQueryAddFeature(qhandle, "QbColorFeatureClass");
 QbQuerySetFeatureData(qhandle, "QbColorFeatureClass",&is);
 QbQuerySearch(qhandle, "ADS", "ADS_IMAGE", 10, 0, resultType
                              &count, results);
 for (int j = 0; j <count; j++) {
 printf(j,":\n");

 DBiBrowse("usr/local/bin/xv %s", MMDB_PLAY_HANDLE, handles[j],
                MMDB_PLAY_WAIT);
 }
```

Figure 5. Application permettant de rechercher des images en fonction de leur contenu (Numéro 1 de 2)

Exemples

```
QbQueryDelete(qhandle);

 SQLDisconnect(hdbc);
 SQLFreeConnect(hdbc);
 SQLFreeEnv(henv);
}
```

Figure 5. Application permettant de rechercher des images en fonction de leur contenu (Numéro 2 de 2)

Environnements d'exploitation

Les Extensions DB2 version 7 fonctionnent avec DB2 Universal Database version 7.1 (ou suivante) dans un environnement client-serveur. Les niveaux de version et d'édition minimaux requis pour ces plateformes sont les mêmes que pour DB2 Universal Database version 7.1.

Les plateformes prises en charge sont les suivantes : OS/2, AIX, Windows NT® et suivante, Windows 95, Windows 98, Solaris et HP-UX.

Les plateformes serveur prises en charge sont les suivantes : OS/2, AIX, Windows NT et version suivante, Solaris et HP-UX.

La figure 6 à la page 13, répertorie les plateformes prises en charge.

Un autre produit DB2 Extensions, DB2 Universal Database pour OS/390 Extensions, prend en charge les clients et serveurs OS/390. Pour plus d'informations sur ce produit, reportez-vous au manuel *DB2 Universal Database for OS/390 Image, Audio, and Video Extenders Administration and Programming* ou au manuel *DB2 Universal Database for OS/390 Text Extender Administration and Programming*.

Les Extensions DB2 fonctionnent dans un environnement de bases de données monopartition.

Produit EEE uniquement : Les extensions peuvent également fonctionner dans un environnement de bases de données multipartition sur les plateformes suivantes : AIX, Solaris et Windows NT et version suivante.

Les extensions DB2 doivent être utilisées avec DB2 Universal Database Enterprise-Extended Edition pour fonctionner dans un environnement de bases de données multipartition.

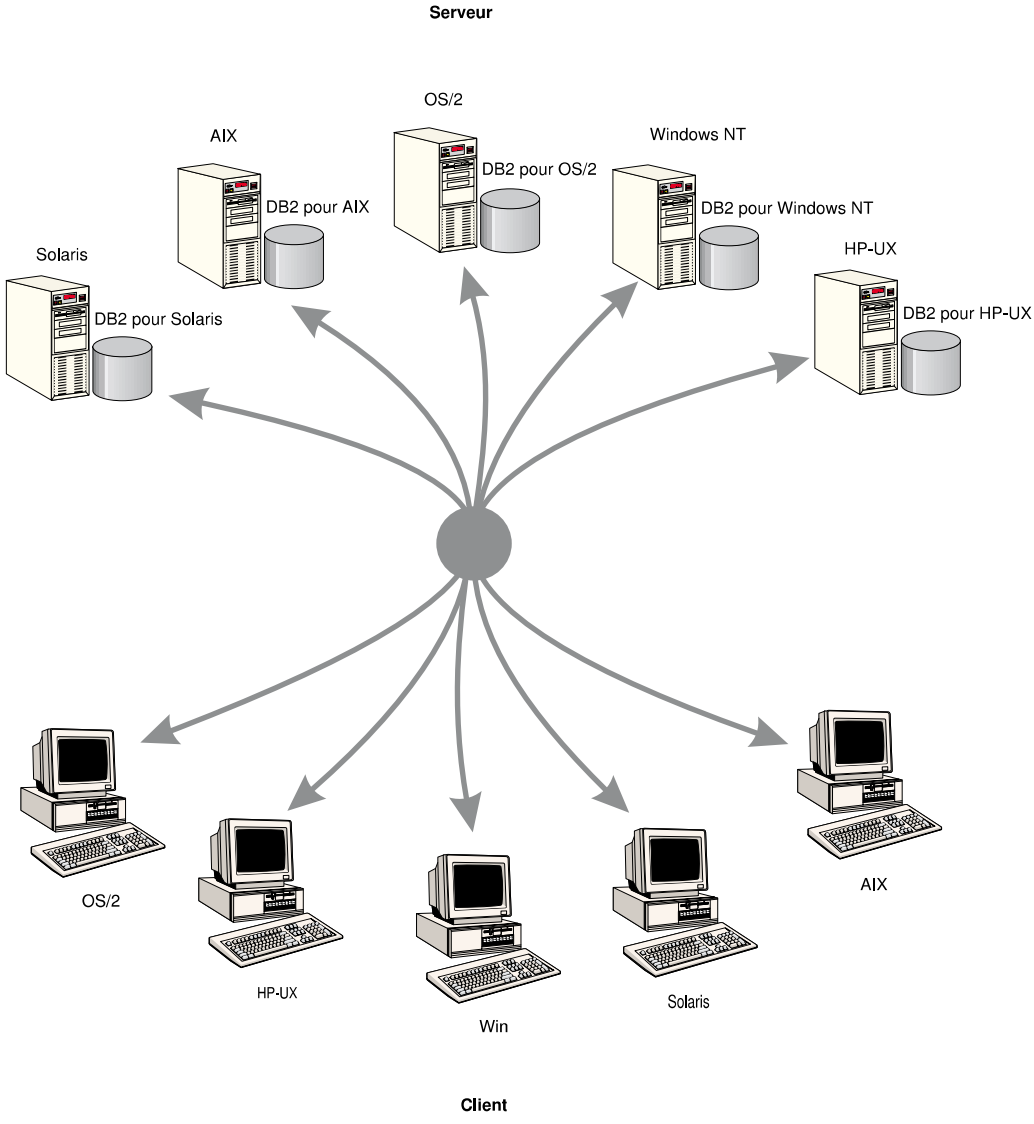


Figure 6. Plateformes prises en charge pour l'utilisation des extensions DB2

Exemples

Chapitre 2. Concepts

Le présent chapitre décrit les concepts dont la compréhension est nécessaire à l'utilisation des Extensions DB2.

Pour plus de détails sur l'orientation objet, reportez-vous au manuel *DB2 Application Development Guide.*

Orientation objet

DB2 prend en charge l'**orientation objet**. Cette technologie repose sur le principe suivant lequel toute chose, concrète ou abstraite peut être représentée dans une application sous forme d'objet se composant d'une série d'opérations et de données. Par exemple, un document peut être représenté par un objet document, formé des données qui le caractérisent et des procédures permettant de les manipuler (archivage, envoi, impression, etc.). Une séquence vidéo peut être représentée par un objet vidéo, regroupant les données vidéo ainsi que les opérations susceptibles d'être exécutées sur ces données (lecture, extraction d'une image vidéo spécifique, etc.). Comme dans le cas des objets que nous manipulons au quotidien, les objets informatiques possèdent des attributs. Par exemple, les attributs d'un objet vidéo peuvent être son type de compression et sa fréquence d'échantillonnage.

Le deuxième principe de l'approche objet consiste à regrouper les objets par type. Les objets de même type possèdent tous les mêmes attributs et ont en commun le même comportement : ils sont tous associés aux mêmes procédures, et ne diffèrent que par leur partie "données". Par exemple, si un type vidéo est créé avec un attribut définissant le type de compression, tous les objets de type vidéo posséderont cet attribut. Si un objet de type vidéo peut être lu, tous les objets du même type pourront également l'être.

La prise en charge par DB2 de l'orientation objet vous permet de stocker les instances des types d'objets dans les colonnes de tables, et d'exécuter des procédures sur ces instances au moyen de fonctions appelées par des

instructions SQL. Ainsi, les objets vidéo peuvent être stockés dans une colonne de table et être manipulés à l'aide de fonctions SQL. En outre, vos applications peuvent toutes partager les attributs et le comportement des objets ainsi stockés : elles «perçoivent» toutes le même ensemble d'attributs et le même comportement pour un même type d'objet.

Par nature, les objets vidéo sont très volumineux et très complexes. Il en est de même pour les objets image et audio. La prise en charge de l'orientation objet par DB2 permet le stockage des objets LOB (objets de grande taille) dans une base de données. Elle vous offre également la possibilité de définir et de manipuler les objets LOB au moyen de types utilisateur (types UDT), de fonctions utilisateur (fonctions UDF) et de déclencheurs.

Objets LOB

DB2 permet le stockage des **objets LOB** (objets de grande taille) sous les formes suivantes :

- objets BLOB (objets binaires de grande taille),
- objets CLOB (objets caractère de grande taille),
- objets DBCLOB (objets caractère double-octet de grande taille).

Les objets BLOB sont des chaînes binaires. Les objets image, audio et vidéo sont stockés dans une base de données DB2 sous forme d'objets BLOB. Les objets CLOB sont des chaînes de caractères à un octet associées à une page de codes. Il s'agit d'objets texte complexes. Les objets DBCLOB sont des chaînes de caractères à deux octets associées à une page de codes. Il s'agit également d'objets texte complexes.

La taille d'un objet LOB est limitée à 2 Go. Il est toutefois possible de définir plusieurs colonnes LOB dans chaque table, ce qui permet le stockage d'objets LOB pouvant occuper jusqu'à 24 Go par ligne et jusqu'à 4 To par table.

En raison de leur taille, les objets LOB ne sont pas directement stockés dans la table utilisateur. Ils y sont identifiés par des descripteurs d'objet LOB, qui sont utilisés pour accéder aux objets, stockés ailleurs sur le disque.

Les extensions DB2 vous offrent davantage de souplesse encore, en vous permettant de stocker le contenu de chaque objet LOB dans un fichier et de pointer sur ce dernier à partir de la base de données. Cette désignation s'effectue lorsque vous utilisez une extension DB2 pour stocker un objet.

Types UDT

Les objets image, vidéo et audio sont représentés dans la base de données sous la forme d'objets BLOB. Les **types utilisateur** (UDT), également connus sous le nom de **type distinct**, permettent de différencier les BLOB. On peut imaginer, par exemple, qu'un type UDT soit créé pour les objets image et un

autre, pour les objets audio. Bien qu'ils soient stockés sous la forme de BLOB, les objets image et audio sont traités comme des types d'objets distincts des BLOB, et différents les uns des autres.

La création des types UDT s'effectue par l'instruction SQL CREATE DISTINCT TYPE. Exemple : vous développez une application destinée au traitement de données cartographiques. Vous pouvez créer un type distinct appelé `carte` pour les objets de type carte, en entrant la commande suivante :

```
CREATE DISTINCT TYPE map AS BLOB (1M)
```

L'objet de type carte est représenté en interne sous la forme d'un BLOB d'un Mo, mais il est traité comme un type d'objet distinct.

Vous pouvez utiliser les types UDT de la même façon que les types intégrés SQL, pour décrire les données stockées dans les colonnes des tables. Dans l'exemple qui suit, une table est créée avec une colonne destinée au stockage des données de type carte :

```
CREATE TABLE communes
  (idemplacmt  INTEGER NOT NULL,
   emplacement CHAR (50),
   grille      map)
```

Chaque extension DB2 crée un type UDT propre à ses caractéristiques (à savoir, image, audio ou vidéo).

Fonctions UDF

Les **fonctions utilisateur** (ou fonctions UDF) permettent de créer des fonctions SQL, et d'enrichir ainsi le jeu de fonctions intégrées fourni par DB2. Vous pouvez notamment créer des fonctions UDF pour l'exécution d'opérations propres aux objets de type image, audio et vidéo. Vous pouvez créer, par exemple, une fonction UDF pour la détermination du format de compression des séquences vidéo, ou pour le calcul de la fréquence d'échantillonnage des séquences audio. Vous avez ainsi la possibilité de définir le comportement des objets d'un type particulier. Les objets vidéo, par exemple, se comportent conformément aux fonctions créées pour le type vidéo, et les objets image suivant celles créées pour le type image.

La création des fonctions UDF s'effectue par l'instruction SQL CREATE FUNCTION. Cette dernière spécifie notamment le type de données auquel s'applique la fonction. Dans l'exemple fourni ci-après, l'instruction crée une fonction UDF appelée `échelle_carte` qui calcule l'échelle des cartes. La fonction identifie par (carte) le type de données auquel elle peut s'appliquer. Le code mettant cette fonction en oeuvre est écrit en C et est identifié dans la clause EXTERNAL NAME :

```
CREATE FUNCTION Echelle_carte (carte)
  RETURNS SMALLINT
  EXTERNAL NAME 'scale!map'
```

```
  LANGUAGE C
  PARAMETER STYLE DB2SQL
  NO SQL
  DETERMINISTIC
  NO EXTERNAL ACTION
```

Les fonctions UDF peuvent être utilisées dans une instruction SQL de la même façon que les fonctions intégrées. Dans l'exemple qui suit, la fonction échelle_carte est appelée à partir d'une instruction SQL SELECT pour obtenir l'échelle d'une carte stockée dans la colonne appelée grille.

```
SELECT échelle_carte (grille)
  FROM communes
  WHERE emplacement='SAN JOSE, CALIFORNIE'
```

Chaque extension DB2 crée un ensemble de fonctions UDF qui lui sont propres (à savoir, des fonctions image, audio ou vidéo). Vous utilisez ces fonctions dans des instructions SQL pour exécuter des opérations propres aux extensions DB2, telles que le stockage d'une image dans une table, le calcul de la fréquence d'échantillonnage d'une vidéo, ou l'ajout de commentaires à une séquence audio.

Dénomination des fonctions UDF et des types UDT

Le nom complet d'une fonction DB2 se présente sous la forme *nom-schéma*.*nom-fonction*, où *nom-schéma* est l'identificateur d'un regroupement logique d'objets SQL. Le nom du schéma utilisé pour les fonctions des extensions DB2 est MMDBSYS. Ce nom sert également de qualificatif pour les types UDT des extensions DB2.

Vous pouvez utiliser son nom complet chaque fois que vous faites référence à une fonction UDF ou à un type UDT. Par exemple, MMDBSYS.CONTENT identifie une fonction UDF dont le nom de schéma est MMDBSYS et le nom de fonction est CONTENT. MMDBSYS.DB2IMAGE identifie un type UDT dont le nom de schéma est MMDBSYS et le nom du type distinct est DB2IMAGE. Il est toutefois possible d'omettre le nom de schéma lorsqu'il est fait référence à UDF ou UDT ; dans ce cas, DB2 utilise le chemin des fonctions pour déterminer la fonction ou le type de données distinct requis.

Chemin des fonctions

Le **chemin des fonctions** est une liste ordonnée des noms de schémas. DB2 suit leur ordre d'apparition dans la liste pour résoudre les références aux fonctions et aux types de données distincts. Vous pouvez spécifier le chemin des fonctions en indiquant l'instruction SQL SET CURRENT FUNCTION PATH. Cette dernière définit le chemin des fonctions dans le registre spécial CURRENT FUNCTION PATH.

Avec DB2 Extensions, il est recommandé d'ajouter le schéma `mmdbsys` du chemin des fonctions. Cela vous permet d'entrer des noms UDF et UDT DB2

Extensions sans avoir à leur attribuer un préfixe à l'aide de mmdbsys. Voici un exemple d'ajout du schéma mmdbsys au chemin des fonctions :

```
SET CURRENT FUNCTION PATH = mmdbsys, CURRENT FUNCTION PATH
```

Ne définissez pas mmdbsys comme premier schéma dans le chemin des fonctions si vous vous connectez à l'aide de l'ID utilisateur mmdbsys : dans ce cas, mmdbsys est défini comme premier schéma dans le chemin des fonctions. Par conséquent, si vous définissez la valeur mmdbsys pour le premier schéma du chemin des fonctions à l'aide de l'instruction SET CURRENT FUNCTION PATH, deux occurrences de ce nom de schéma seront présentes dans le chemin des fonctions, ce qui provoquera une erreur.

Noms de fonctions multi-référencés

Les noms de fonctions peuvent être **multi-référencés**. Cela signifie que plusieurs fonctions UDF, y compris au sein d'un même schéma, peuvent porter le même nom. En revanche, deux fonctions ne peuvent pas porter la même **signature**. La signature d'une fonction résulte de la concaténation du nom qualifié complet de la fonction et des types de données de tous les paramètres de cette fonction.

Déclencheurs

Un **déclencheur** définit un ensemble d'opérations qui sont exécutées lorsqu'une modification est apportée à une table. Les déclencheurs peuvent être utilisés pour des opérations telles que la validation des données d'entrée, la génération automatique d'une valeur pour une nouvelle ligne insérée, la lecture d'autres tables pour la résolution de références croisées, ou l'écriture de données dans d'autres tables à des fins de contrôle. Le plus souvent, les déclencheurs sont utilisés pour le contrôle de l'intégrité des données et pour l'application de règles propres à l'activité de l'entreprise.

La création d'un déclencheur s'effectue à l'aide de l'instruction SQL CREATE TRIGGER. Dans l'exemple qui suit, un déclencheur est créé pour appliquer une règle inhérente à la gestion des stocks. Le déclencheur provoque le passage d'une nouvelle commande lorsque le nombre d'articles disponibles tombe sous la barre des 10 % du stock initial.

```
CREATE TRIGGER reorder
       AFTER UPDATE OF on_hand, max_stocked ON parts
       REFERENCING NEW AS n_row
       FOR EACH ROW MODE DB2SQL

       WHEN (n_row.on_hand < 0.10 * n_row.max_stocked)
       BEGIN ATOMIC
              VALUES(issue_ship_request(n_row.max_stocked -
                                        n_row.on_hand,
                                        n_row.partno));
       END
```

Orientation objet

Les Extensions DB2 créent et utilisent des tables de gestion destinées au stockage des données image, audio et vidéo dans une base de données. (Pour plus de détails, reportez-vous à la section «Tables de gestion».) Les Extensions utilisent des déclencheurs pour la réactualisation de ces tables en cas d'insertion, de mise à jour ou de suppression de données dans la base de données.

Structures des données gérées par les extensions

Les extensions Image, Audio et Vidéo créent et utilisent des tables et des descripteurs de gestion, afin de permettre le stockage des données images, audio et vidéo. L'extension Image crée également des catalogues QBIC afin de permettre la recherche de données par contenu. L'extension Vidéo utilise, quant à elle, des fichiers d'indexation et des catalogues des prises de vue pour accéder à des informations sur les changements de plan dans les séquences vidéo.

Tables de gestion

Les **tables de gestion**, également appelées tables de métadonnées, contiennent des informations nécessaires aux Extensions pour le traitement des requêtes utilisateur portant sur les objets image, audio et vidéo. Ces informations sont souvent désignées par le terme «métadonnées».

Comme l'illustre la figure 7 à la page 21, certaines des tables de gestion identifient les tables et colonnes utilisateur activées pour une utilisation par les extensions DB2. Ces tables pointent sur d'autres tables de gestion dans lesquelles sont consignés les attributs des objets contenus dans les colonnes activées. Les informations consignées par les Extensions dans ces secondes tables de gestion portent à la fois sur les attributs spécifiques d'un type de données et sur les attributs communs aux différents types de données gérés par les extensions. Par exemple, l'extension Image assure la gestion d'informations propres aux objets image (largeur, hauteur, nombre de couleurs, etc.), ainsi que d'autres attributs communs à tous les types de données (identificateur de l'utilisateur responsable de l'importation de l'objet dans la base de données, identificateur de l'utilisateur responsable de la dernière mise à jour de l'objet, etc.).

Outre ces informations, les tables de gestion peuvent contenir les objets eux-mêmes au format BLOB. Si tel n'est pas le cas, les objets sont stockés ailleurs, dans des fichiers, et sont uniquement référencés dans les tables de gestion. Par exemple, une séquence vidéo peut être stockée sous la forme d'un objet BLOB dans une table de gestion ou conservée dans un fichier référencé par cette table.

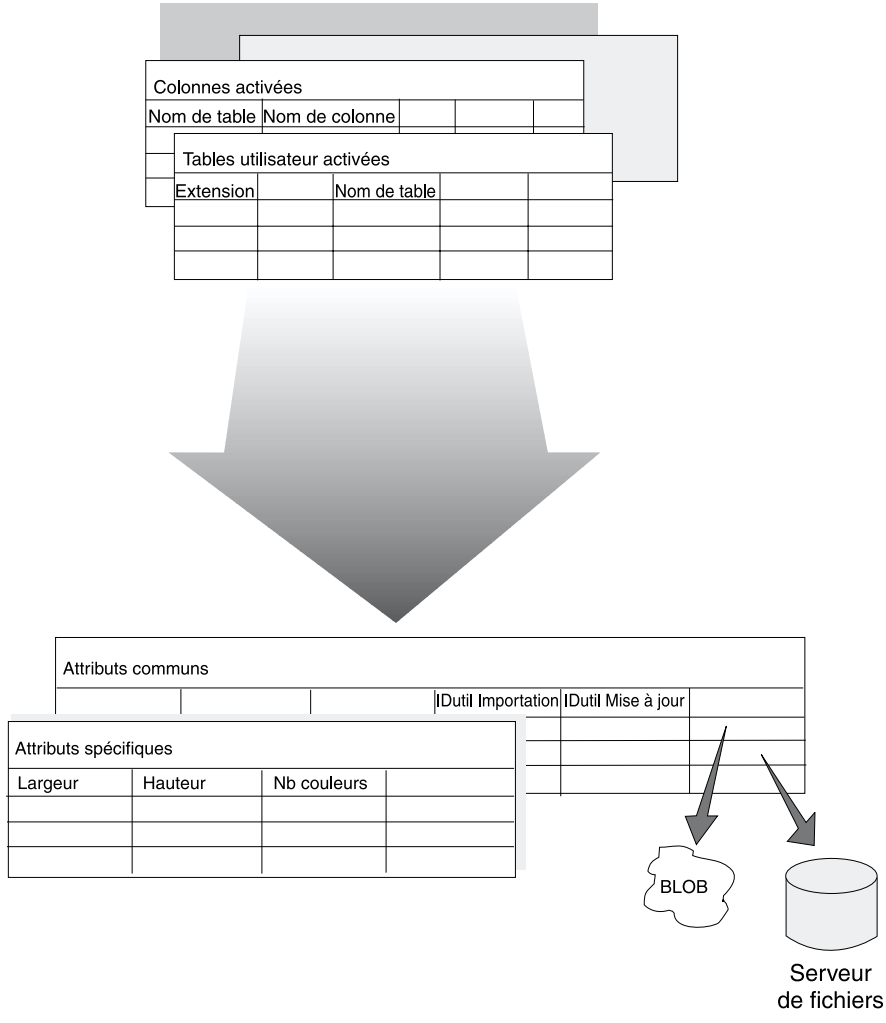


Figure 7. Tables de gestion

Descripteurs

Lorsque vous archivez un objet image, audio ou vidéo dans une table utilisateur, l'objet lui-même n'est pas véritablement stocké dans la table. En effet, l'extension DB2 correspondante crée une chaîne de caractères, appelée **descripteur**, représentant l'objet et la stocke dans la table utilisateur. L'objet lui-même est stocké soit dans une table de gestion, soit dans un fichier. Dans ce dernier cas, l'extension stocke l'identificateur du fichier dans une table de gestion. Elle consigne également les attributs et le descripteur de l'objet dans les tables de gestion. De cette façon, l'extension peut établir le lien entre le descripteur stocké dans la table utilisateur et les informations stockées dans

les tables de gestion. La figure 8, illustre le principe d'utilisation des descripteurs pour deux images archivées dans une table utilisateur.

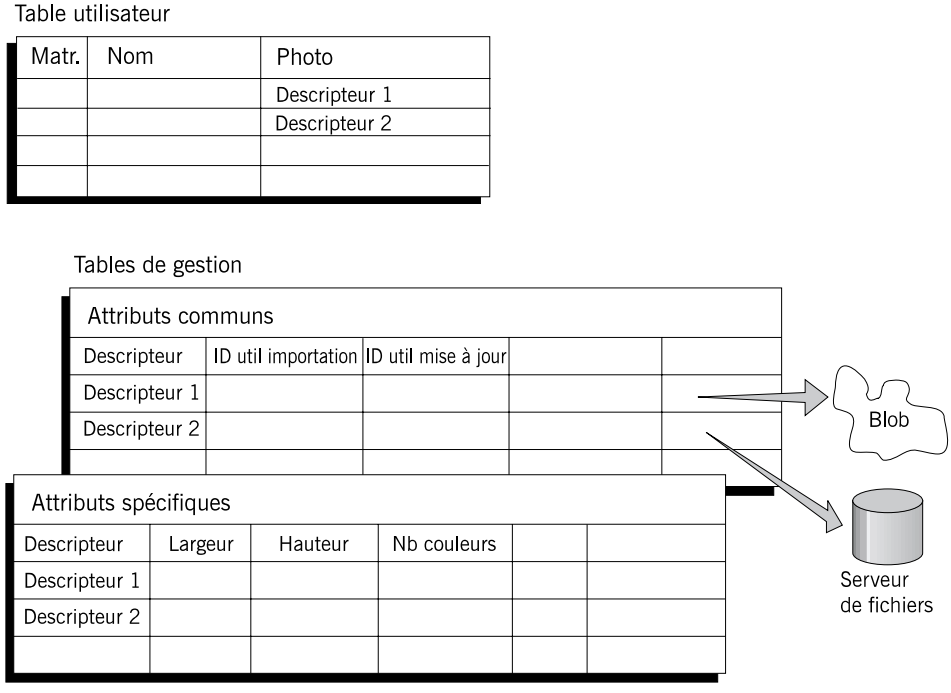


Figure 8. Descripteurs

Catalogues QBIC

Un **catalogue QBIC** est un jeu de fichiers contenant les données relatives aux caractéristiques visuelles des images. L'extension Image utilise ces données pour les recherches par contenu d'image (ou recherches QBIC).

Un catalogue QBIC doit être créé pour chaque colonne d'images d'une table utilisateur sur laquelle vous souhaitez pouvoir effectuer des recherches QBIC. Lors de la création du catalogue, vous identifiez les caractéristiques que l'extension Image doit analyser et stocker et sur lesquelles pourront porter les recherches. Une fois le catalogue créé, il est toujours possible d'ajouter de nouvelles caractéristiques au catalogue ou d'en supprimer.

Les caractéristiques suivantes peuvent être stockées dans un catalogue QBIC :

Couleur moyenne
Somme des couleurs de tous les pixels d'une image, rapportée au nombre de pixels de cette image. (On appelle *pixel* le plus petit composant d'une image auquel puissent être affectées une couleur et une intensité.) Par exemple, si la moitié d'une

image est constituée de pixels bleus et l'autre, de pixels rouges, le violet sera la couleur moyenne de l'image. La couleur moyenne est surtout utilisée pour rechercher des images ayant une couleur prédominante. En effet, dans le cas de telles images, la couleur moyenne est proche de la couleur prédominante.

Couleur d'histogramme
Mesure de la distribution des couleurs d'une image par rapport à un spectre de 64 couleurs. Pour chacune de ces couleurs, l'extension analyse l'image et détermine le pourcentage de pixels correspondant. On peut imaginer, par exemple, que l'histogramme d'une image soit représenté par 40% de pixels blancs, 50% de pixels bleus et 10% de pixels rouges, et que l'image ne comporte aucun pixel correspondant aux autres couleurs du spectre. Cette caractéristique permet de rechercher des images comportant plusieurs couleurs.

Couleur positionnelle
Couleur moyenne des pixels compris dans une zone déterminée de l'image. Par exemple, si l'on imagine qu'un soleil d'une couleur jaune vif apparaisse dans le coin supérieur gauche d'une image, la couleur positionnelle de cette zone sera le jaune vif. Cette caractéristique peut être utilisée pour la recherche d'images présentant une couleur prédominante dans une zone spécifique.

Texture Mesure du grain, du contraste et de la directionnalité de l'image. Le grain indique la taille des éléments répétitifs d'une image (par exemple, relief granité par opposition à surface lisse). Le contraste identifie les variations de luminosité (zones claires par opposition aux zones foncées). La directionnalité traduit la prédominance d'une direction (verticalité d'une palissade formée d'une rangée de pieux, par exemple) ou son absence (cas d'une étendue de sable, par exemple). La texture permet de rechercher les images présentant un aspect particulier.

Pour qu'une image puisse faire l'objet d'une recherche QBIC, elle doit être cataloguée. Lors de son catalogage, l'extension Image analyse l'image en calculant la valeur de chacune de ses caractéristiques et stocke ces valeurs dans un catalogue QBIC.

Lorsque vous effectuez une recherche QBIC, votre requête indique chaque caractéristique à utiliser comme critère (par exemple, la couleur moyenne), la source servant de référence à la recherche (par exemple, un échantillon graphique), ainsi qu'un jeu cible d'images cataloguées. L'extension Image

calcule alors la valeur de la caractéristique choisie pour l'échantillon fourni et la compare aux valeurs enregistrées pour les images cibles. Elle attribue ensuite un score aux images retrouvées en fonction de leur degré de correspondance par rapport à l'échantillon source.

Vous pouvez demander le renvoi des images les plus proches de l'échantillon fourni. Vous obtiendrez alors le descripteur de fichier et le score de chaque image. Il est également possible de demander uniquement le renvoi du score d'une image.

Index vidéo

Un **index vidéo** est un fichier utilisé par l'extension Vidéo pour retrouver une prise de vue ou une image spécifique dans une séquence vidéo.

L'extension Vidéo est capable de détecter les changements de plan dans les séquences vidéo. Un **changement de plan** est un point d'une séquence vidéo caractérisé par une modification notable entre deux images successives. On constate, par exemple, qu'un changement de plan se produit lorsque l'objectif de la caméra est déplacé au cours d'une séquence vidéo. Les images comprises entre deux changements de plan constituent une **prise de vue**.

Vous pouvez utiliser les fonctions de détection de changement de plan de l'extension Vidéo pour retrouver une prise de vue, ou une image individuelle, dans une séquence vidéo. Pour cela, il est nécessaire que l'extension dispose de données d'indexation pour la prise de vue ou l'image recherchée. Ces données sont stockées dans un **fichier d'indexation**.

Catalogues de prises de vue

Un **catalogue des prises de vue** est une table de base de données susceptible d'être utilisée pour le stockage de données sur les prises de vue d'une séquence vidéo. Le catalogue des prises de vue peut être stocké dans une base de données ou dans un fichier.

Un catalogue des prises de vue stocké dans un fichier contient les informations suivantes, relatives aux prises de vue :

- Nom du catalogue des prises de vue
- Valeurs gérant la façon dont l'extension Vidéo détecte une prise de vue, par exemple, le nombre minimal d'images vidéo dans une prise de vue
- Valeurs gérant le nombre d'images vidéo et celles qui vont être stockées en tant qu'images représentatives d'une prise de vue
- Numéro de la prise de vue
- Numéro de la première image
- Numéro de la dernière image
- Numéro de l'image représentative

- Nom d'un fichier contenant le détail de l'image représentative

Vous pouvez accéder aux données du fichier de catalogue des prises de vue ou à une vue du catalogue des prises de vue stocké dans une base de données. Cette vue contient des colonnes pour les données suivantes :

- Descripteur de prise de vue
- Nom de la table vidéo
- Nom des colonnes vidéo
- Descripteur de la séquence vidéo
- Nom de fichier de la séquence vidéo
- Numéro de la première image
- Numéro de la dernière image
- Numéro de l'image représentative
- Données de l'image représentative
- Commentaire

Base de données partitionnée (Produit EEE uniquement)

Les extensions DB2 peuvent fonctionner avec DB2 Extended Enterprise Edition et peuvent donc tirer parti du support de base de données partitionnée fourni par DB2 Extended Enterprise Edition.

Une **base de données partitionnée** est une base de données répartie entre plusieurs machines indépendantes. Pour l'utilisateur final et le développeur d'applications, elle apparaît comme une base de données unique sur une machine unique. Le partitionnement permet aux applications de mieux exploiter une base de données trop volumineuse pour être gérée par une seule machine.

Une base de données partitionnée est composée d'au moins deux partitions, chacune étant gérée par son propre **serveur de partitions de base de données**. Ce dernier comprend un gestionnaire de bases de données et l'ensemble des données et ressources système gérées par ce serveur. Généralement, un serveur de partitions de bases de données est affecté à chaque machine. Cependant, il est possible de disposer de plusieurs serveurs de partitions de base de données sur la même machine. Chaque serveur de partitions de base de données détient une partie de la base complète. Ce serveur est parfois désigné sous le terme de **noeud**.

Comme l'illustre la figure 9 à la page 26, les partitions de base de données peuvent faire l'objet d'un regroupement logique et se voir affecter un nom. Chaque groupe de partitions de base de données est désigné sous le nom de **groupe de noeuds**. La définition d'un groupe de noeuds vous permet, par

exemple, de limiter les requêtes de l'application aux partitions de base de données sélectionnées et donc d'accélérer les délais requis pour les transactions. Un groupe de noeuds peut contenir une seule ou plusieurs partitions de bases de données. S'il en contient plusieurs, il est appelé **groupe de noeuds multipartition**. Toutes les partitions nommées dans ce groupe doivent alors résider sur la même base de données.

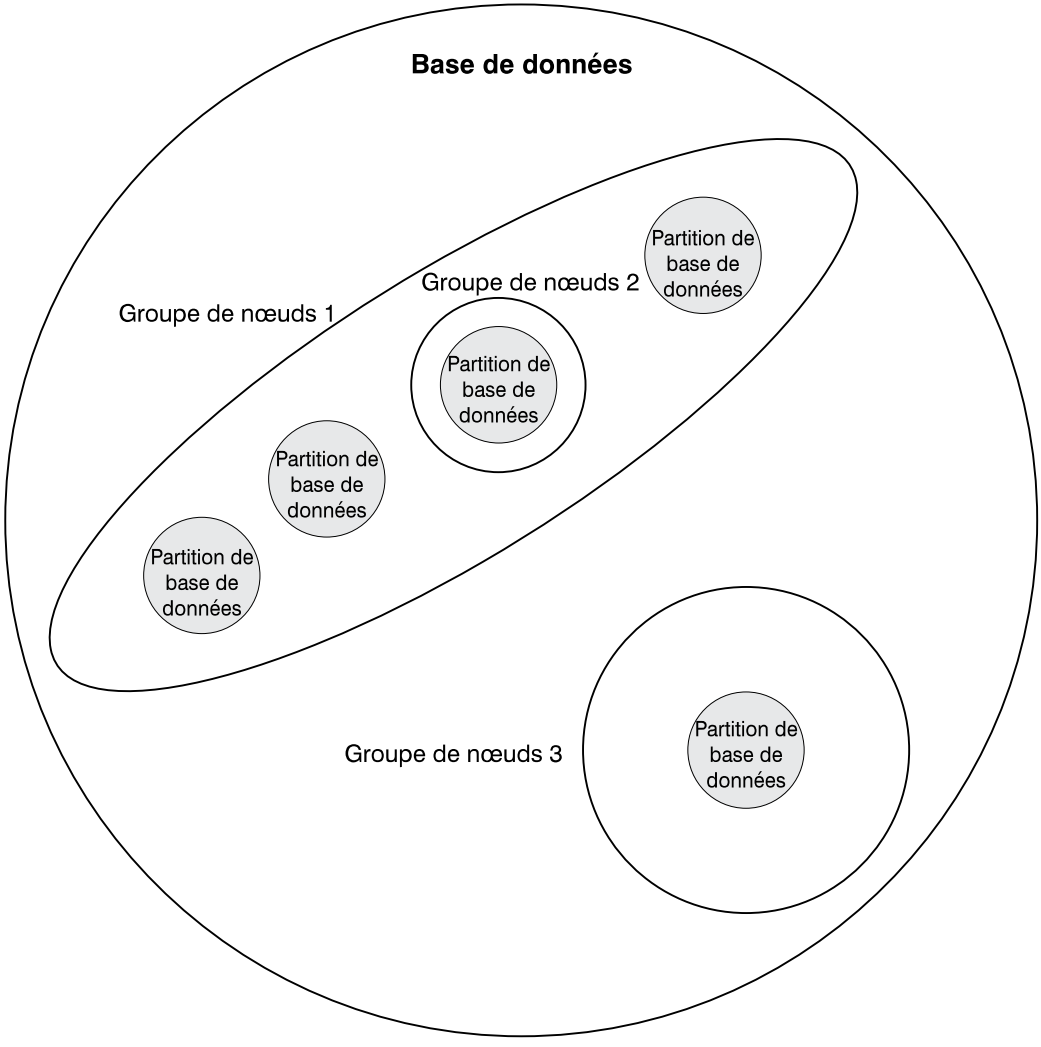


Figure 9. Groupes de noeuds dans une base de données

L'utilisation des extensions dans un système de bases de données partitionnées vous permet de :

- Réduire les goulots d'étranglement d'entrée-sortie et de traitement en répartissant les données entre plusieurs partitions.
- Accroître la taille de votre base de données en ajoutant des machines supplémentaires et en redistribuant les données.

Traitement parallèle

Une base de données partitionnée peut utiliser plusieurs unités centrales pour répondre aux demandes d'informations. La récupération et la mise à jour des requêtes sont automatiquement affectées à des sous-requêtes et exécutées en parallèle sur les serveurs de partitions de bases de données sur chaque machine.

Pour illustrer la puissance de traitement d'un système de bases de données partitionnées, supposons que vous vouliez scanner 100 000 000 enregistrements dans une base de données monopartition. Cette opération nécessite la recherche de 100 000 000 enregistrements par un seul gestionnaire de bases de données. Supposons à présent que ces enregistrements soient répartis de façon égale entre 20 serveurs de partitions de bases de données ; chaque gestionnaire de bases de données n'a que 5 000 000 enregistrements à scanner. Si chacun de ces gestionnaires scanne au même moment et à la même vitesse, le temps requis pour mener à bien l'opération sera approximativement égal à 5% du temps nécessaire à un système monopartition pour exécuter la même tâche.

Evolutivité

Puisque la taille de votre base de données s'est accrue, vous pouvez ajouter des serveurs de partitions de base de données au système de bases de données pour de meilleures performances. Ce processus est désigné sous le terme de **changement d'échelle** du système de bases de données.

Lorsque vous changez l'échelle d'une base de données, vous ajoutez un serveur de partitions de base de données, lequel ajoute à son tour une partition de base de données à chaque base du système. Vous pouvez alors affecter la nouvelle partition à un groupe de noeuds existant pour cette base de données. Enfin, vous pouvez redistribuer les données dans ce groupe de noeuds pour pouvoir utiliser la nouvelle partition de base de données.

Utilisation des extensions DB2 dans un environnement de bases de données partitionnées

L'utilisation des extensions DB2 dans un environnement de bases de données partitionnées vous permet d'exploiter les fonctions prenant particulièrement bien en charge la manipulation des objets LOB. D'importants référentiels d'objets LOB (la longueur de chacun de ces référentiels pouvant aller jusqu'à

Base de données partitionnée

2 gigaoctets) peuvent être stockés dans la même base de données, puisque celle-ci peut être répartie sur plusieurs machines.

De plus, les extensions DB2 prennent part au traitement parallèle des opérations SQL puisqu'elles sont gérées par DB2 Extended Enterprise Edition. Lorsque ce dernier exécute une requête en parallèle, toute fonction UDF de DB2 Extensions incluse dans la requête est également exécutée en parallèle sur les partitions de base de données individuelles.

Sécurité et récupération

Les objets image, audio et vidéo stockés sous la forme de BLOB dans les bases de données DB2 offrent les mêmes garanties en termes de sécurité et de récupération que les données alphanumériques classiques. Il en est de même pour les informations stockées sur ces objets dans les tables de gestion. Les utilisateurs doivent disposer des privilèges nécessaires pour la sélection, l'insertion et la mise à jour des objets.

Un utilisateur crée des fonctions UDF pour sélectionner, insérer, mettre à jour ou supprimer des objets dans une table utilisateur. Ces fonctions doivent alors pouvoir accéder aux tables de gestion contenant les informations d'attribution d'objets et, si nécessaire, mettre à jour ces tables. Les extensions permettent aux fonctions UDF d'effectuer ces opérations sur les tables de gestion si l'utilisateur dispose du privilège approprié sur la table utilisateur.

Le niveau d'habilitation DBADM est nécessaire à l'exécution de certaines opérations d'administration faisant appel aux extensions DB2. Reportez-vous au «Chapitre 16. Interfaces de programmation d'applications» à la page 279, pour plus de détails sur les droits requis pour les API d'administration de DB2 Extensions, et au «Chapitre 17. Commandes d'administration du poste client» à la page 487, pour plus de détails sur les droits requis pour les commandes d'administration de DB2 Extensions.

Lorsque le contenu d'une image, d'une séquence audio ou vidéo est stocké dans un fichier référencé dans la base de données, les métadonnées associées à l'objet sont protégées par DB2. Le fichier doit être stocké dans un répertoire accessible à tous les utilisateurs (répertoire dit PUBLIC).

Les objets BLOB et les métadonnées peuvent être sauvegardés et récupérés de la même façon que les autres types de données DB2. La sauvegarde et la récupération du contenu des objets stockés dans des fichiers peuvent être effectuées à l'aide d'outils non DB2. Les catalogues QBIC et les index vidéo peuvent être sauvegardés et récupérés à l'aide d'outils d'indexation non DB2. Pour plus de détails sur la procédure à suivre dans le cas d'un catalogue QBIC, reportez-vous à la page 140. Pour plus de détails sur la sauvegarde d'un index vidéo, reportez-vous à la page191.

Chapitre 3. Fonctionnement des extensions DB2

Les Extensions DB2 mettent en oeuvre plusieurs processus pour la gestion des requêtes portant sur les données images, audio ou vidéo. La meilleure façon de comprendre le principe de fonctionnement des Extensions DB2 est d'observer ce qui se produit lorsque vous les utilisez. Le présent chapitre décrit un scénario faisant appel aux extensions Image et Audio. Les opérations effectuées par l'utilisateur y sont décrites, ainsi que les réactions des deux extensions.

Exemple d'utilisation des extensions DB2

Le service des ressources humaines d'une entreprise souhaite créer une base de données (sous DB2 pour AIX) afin de formaliser le suivi de chacun des collaborateurs de la société.

Création d'une base de données permettant le stockage d'images : Comme l'illustre la figure 10, la table Employés sera créée dans la base de données pour l'enregistrement du nom, du matricule et de la photo de chaque employé.

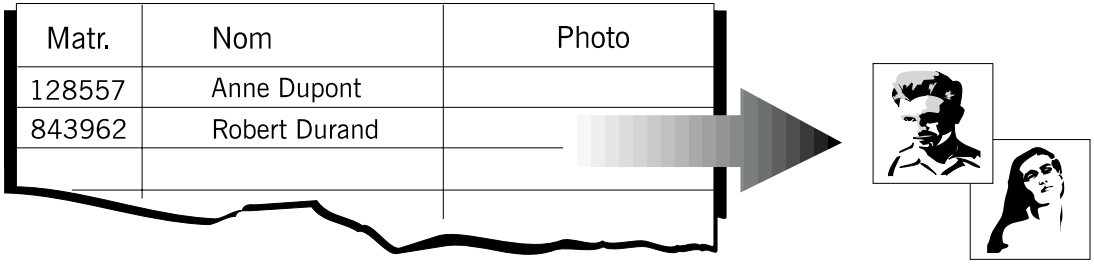


Figure 10. Table Employés

Pour préparer la base de données au traitement graphique, l'administrateur système (c'est-à-dire un utilisateur disposant des droits SYSADM) commence par démarrer les fonctions DB2 Extensions. Il crée ensuite la base de données et la configure pour l'extension Image.

Un administrateur de base de données (DBA), ou tout utilisateur disposant des privilèges correspondants, crée la table *Employés*. Il configure cette dernière ainsi que la colonne *Photo* pour que l'une et l'autre puissent être utilisées par l'extension Image.

Exemple d'utilisation

Création d'une base de données permettant le stockage de séquences audio : Une fois la base de données Personnl et la table Employés préparées au traitement graphique, le service des ressources humaines décide d'ajouter à la table un échantillon vocal pour chaque employé. Cette opération est illustrée par la figure 11.

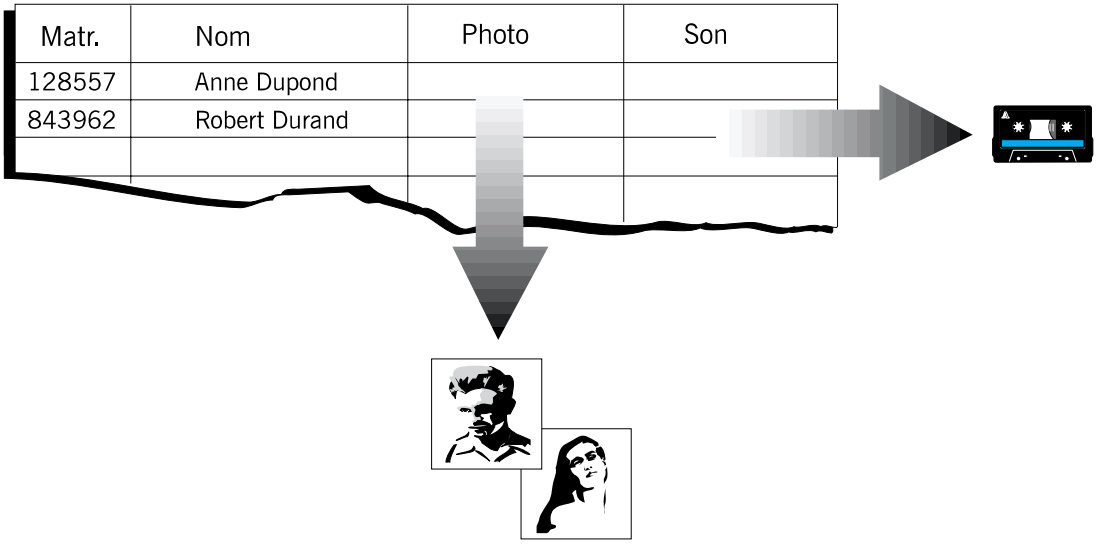


Figure 11. Ajout d'une colonne audio à la table Employés

L'administrateur système modifie la table en lui ajoutant une nouvelle colonne. Il configure ensuite la base de données, la table et cette colonne pour l'extension Audio.

Les utilisateurs du service des ressources humaines peuvent ensuite insérer des données dans la table, les sélectionner, les afficher, les mettre à jour et les supprimer, suivant leurs besoins.

Démarrage des fonctions DB2 Extensions

Les Extensions DB2 utilisent certaines fonctions du serveur pour les processus qu'elles mettent en oeuvre. Si ces fonctions ne sont pas automatiquement disponibles en tant que modules lancés au démarrage du serveur, l'administrateur système doit les démarrer.

Opérations effectuées par l'administrateur système : L'administrateur système se connecte au serveur AIX en tant que propriétaire de l'instance des extensions. Il entre ensuite la commande suivante à partir du serveur :

```
DMBSTART
```

Résultat obtenu : Les fonctions DB2 Extensions sont démarrées pour l'instance des extensions sur le serveur. Si aucune instance DB2 n'était encore active, la commande DMBSTART en démarre également une.

Préparation d'une base de données

L'administrateur système crée et configure la base de données Personnl pour que l'extension Image puisse l'utiliser.

Opérations effectuées par l'administrateur système : L'administrateur système crée la base de données Personnl sous DB2 pour AIX à l'aide de l'instruction SQL suivante :

```
CREATE DATABASE Personnl                /*nom de la base de données*/
       ON /persdb                       /nom du répertoire de stockage*/
       WITH "base de données Personnl"  /*commentaire*/
```

Il se connecte à la base de données et la configure pour l'extension Image, puis utilise l'interpréteur de commandes db2ext pour émettre les commandes suivantes :

```
CONNECT TO personnl
ENABLE DATABASE FOR DB2IMAGE
```

Résultat obtenu : En réponse à la commande ENABLE DATABASE, l'extension Image :

- Crée un type UDT appelé DB2IMAGE pour les objets image.
- Crée des tables de gestion pour les objets image.
- Crée des fonctions UDF pour les objets image. Les fonctions UDF ainsi créées sont répertoriées dans le tableau 1.

Tableau 1. Fonctions UDF créées par l'extension Image

Fonction UDF	Description
Commentaire	Extraction ou mise à jour des commentaires utilisateur.
Content	Extraction ou mise à jour du contenu d'une image.
DB2Image	Stockage du contenu d'une image.
Filename	Détermination du nom du fichier contenant une image.
Format	Détermination du format d'une image (par exemple, GIF).
Height	Détermination de la hauteur (en pixels) d'une image.
Importer	Détermination de l'ID utilisateur responsable de l'importation d'une image.
ImportTime	Détermination de l'horodatage d'importation d'une image.
NumColors	Détermination du nombre de couleurs utilisées dans une image.

Tableau 1. Fonctions UDF créées par l'extension Image (suite)

Fonction UDF	Description
QbScoreFromName	Détermination du score de similarité d'une image (à l'aide d'une requête nommée)
QbScoreFromStr	Détermination du score de similarité d'une image (à l'aide d'une chaîne de requête)
QbScoreTBFromName	Détermination des scores de similarité d'une colonne d'images (à l'aide d'une requête nommée)
QbScoreTBFromStr	Détermination d'une table de scores de similarité pour une colonne d'images (à l'aide d'une chaîne de requête)
Replace	Mise à jour du contenu d'une image et des commentaires utilisateur qui lui sont associés.
Size	Détermination de la taille (en octets) d'une image.
Thumbnail	Obtention d'une version miniature d'une image.
Updater	Détermination de l'ID utilisateur responsable de la mise à jour d'une image.
UpdateTime	Détermination de l'horodatage de mise à jour d'une image.
Width	Détermination de la largeur (en pixels) d'une image.

Préparation d'une table

L'administrateur de base de données crée une table *Employés* et la configure, ainsi que la colonne *Photo*, pour l'extension Image.

Opérations effectuées par l'administrateur de base de données : Pour plus de facilité, l'administrateur de base de données ajoute le schéma mmdbsys dans le chemin des fonctions en cours, à l'aide de l'instruction SQL suivante :

```
SET CURRENT FUNCTION PATH = mmdbsys, CURRENT FUNCTION PATH
```

Lorsque le nom de schéma mmdbsys figure dans le chemin des fonctions, il n'est pas nécessaire d'utiliser des noms qualifiés complets pour les fonctions UDF et les types UDT. (Il n'est pas obligatoire que mmdbsys soit le premier schéma dans le chemin des fonctions.) Pour plus de détails, reportez-vous à la section «Dénomination des fonctions UDF et des types UDT» à la page 18.

L'administrateur de base de données crée la table Employés. Il utilise l'interpréteur de commandes DB2 pour émettre l'instruction SQL suivantes :

```
CREATE TABLE   Employés          /*nom de la table*/
       (id     CHAR(6)           /*matricule du collaborateur*/
       nom     VARCHAR(40)       /*nom du collaborateur*/
       photo   DB2IMAGE)         /*photo du collaborateur*/
```

L'administrateur de base de données entre ensuite les commandes suivantes à partir de l'interpréteur de commandes db2ext :

```
ENABLE TABLE Employés FOR DB2IMAGE
ENABLE COLUMN Employés Photo FOR DB2IMAGE
```

Résultat obtenu : En réponse à la commande ENABLE TABLE, l'extension Image exécute les opérations suivantes :

- Identification de la table Employés en vue de son utilisation.
- Création de tables de gestion contenant des informations sur les attributs des objets image présents dans les colonnes activées.

En réponse à la commande ENABLE COLUMN, l'extension Image exécute les opérations suivantes :

- Identification de la colonne *Photo* en vue d'une utilisation future.
- Création de déclencheurs. Ces derniers sont utilisés pour la mise à jour de diverses tables de gestion suite aux opérations d'insertion, de mise à jour et de suppression portant sur la table *Employés*.

Modification d'une table

L'administrateur de base de données ajoute une colonne Audio à la table *Employés* et la configure pour l'extension Audio.

Opérations effectuées par l'administrateur de base de données :
L'administrateur de base de données utilise l'interpréteur de commandes db2ext pour activer la base de données personnel pour une utilisation par l'extension Audio :

```
ENABLE DATABASE FOR DB2AUDIO
```

Il émet ensuite l'instruction SQL suivante pour modifier la table Employés. Il utilise l'interpréteur de commandes DB2 pour émettre cette instruction.

```
ALTER TABLE Employés            /*nom de la table*/
      ADD Son DB2AUDIO          /*enregistrement audio collaborateur*/
```

L'administrateur de base de données configure ensuite la table Employés et la colonne Son pour l'extension Audio, à l'aide des commandes suivantes, entrées à partir de l'interpréteur de commandes db2ext :

```
ENABLE TABLE Employés FOR DB2AUDIO

ENABLE COLUMN Employés Son FOR DB2AUDIO
```

Résultat obtenu : En réponse à la commande ENABLE DATABASE, l'extension Audio exécute les opérations suivantes :

- Création d'un type UDT appelé DB2AUDIO pour les objets audio.

- Création de tables de gestion pour les objets audio.
- Création de fonctions UDF pour les objets audio. Les fonctions UDF ainsi créées sont répertoriées dans le tableau 2.

Tableau 2. Fonctions UDF créées par l'extension Audio

Fonction UDF	Description
AlignValue	Détermination du nombre d'octets par échantillon audionumérique dans la séquence audio.
BitsPerSample	Détermination du nombre de bits utilisés pour représenter la séquence audio.
BytesPerSec	Détermination du nombre moyen d'octets par seconde de la séquence audio.
Comment	Extraction ou mise à jour des commentaires utilisateur.
Content	Extraction ou mise à jour du contenu d'une séquence audio.
DB2Audio	Stockage du contenu d'une séquence audio.
Duration	Détermination de la durée d'une séquence audio.
Filename	Détermination du nom du fichier contenant une séquence audio.
FindInstrument	Détermination du numéro de la piste sur laquelle est enregistré un instrument spécifique d'une séquence audio.
FindTrackName	Détermination du numéro d'une piste nommée dans un enregistrement audio.
Format	Détermination du format audio.
GetInstruments	Détermination du nom des instruments enregistrés dans une séquence audio.
GetTrackNames	Détermination du nom des pistes d'une séquence audio.
Importer	Détermination de l'ID utilisateur responsable de l'importation de la séquence audio.
ImportTime	Détermination de l'horodatage d'importation de la séquence audio.
NumAudioTracks	Détermination du nombre de pistes enregistrées d'une séquence audio.
NumChannels	Détermination du nombre de canaux audio.
Replace	Mise à jour du contenu de l'enregistrement audio ou des commentaires utilisateur qui lui sont associés.
SamplingRate	Détermination de la fréquence d'échantillonnage de la séquence audio.

Tableau 2. Fonctions UDF créées par l'extension Audio (suite)

Fonction UDF	Description
Size	Détermination de la taille (en octets) d'une séquence audio.
TicksPerQNote	Détermination du nombre de battements de métronome par quart de mesure utilisé dans l'enregistrement d'une séquence audio.
TicksPerSec	Détermination du nombre de battements de métronome par seconde utilisé dans l'enregistrement d'une séquence audio.
Updater	Détermination de l'ID utilisateur responsable de la mise à jour d'un enregistrement audio.
UpdateTime	Détermination de l'horodatage de mise à jour d'un enregistrement audio.

En réponse à la commande ENABLE TABLE, l'extension Audio exécute les opérations suivantes :

- Identification de la table Employés en vue de son utilisation.
- Création des tables de gestion contenant des informations sur les attributs des objets audio situés dans les colonnes configurées.

En réponse à la commande ENABLE COLUMN, l'extension Audio exécute les opérations suivantes :

- Identification de la colonne *Son* en vue d'une utilisation future.
- Création de déclencheurs. Ces derniers sont utilisés pour la mise à jour de diverses tables de gestion suite aux opérations d'insertion, de mise à jour et de suppression portant sur la table *Employés*.

Insertion de données dans une table

Un utilisateur insère une entrée concernant Anne Dupont dans la table *Employés*. Cette entrée comporte le matricule de la collaboratrice (128557), son nom, sa photo et un échantillon de sa voix. L'image source et l'échantillon vocal sont des fichiers disponibles sur le serveur. L'image est stockée dans la table sous la forme d'un objet BLOB ; quant au contenu de la séquence audio, il est conservé dans le fichier du serveur (l'entrée de la table pointe sur ce fichier).

Opérations effectuées par l'utilisateur : L'utilisateur insère l'entrée dans la table Employés via un programme d'application comportant les instructions présentées à la figure 12 à la page 36.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvInt_Stor;
long hvExt_Stor;
  EXEC SQL END DECLARE SECTION;

hvInt_Stor = MMDB_STORAGE_TYPE_INTERNAL;
hvExt_Stor = MMDB_STORAGE_TYPE_EXTERNAL;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',                          /*id*/
        'Anne Dupont',                     /*nom*/
        DB2IMAGE(                           /*UDF de l'extension Image*/
           CURRENT SERVER,                  /*nom du serveur de bases de données*/
                                            /*dans le registre CURRENT SERVER*/
           '/Employés/images/adupont.bmp'   /*fichier source image*/
           'ASIS',                          /*conservation du format de l'image*/
           :hvInt_Stor,                     /*stockage de l'image sous forme de BLOB*/
           'Photo d''Anne Dupont'),         /*commentaire*/
        DB2AUDIO(                           /*UDF de l'extension Audio*/
           CURRENT SERVER,                  /*nom du serveur de bases de données*/
                                            /*dans le registre CURRENT SERVER*/
           '/Employés/audio/adupont.wav',   /*fichier source audio*/
           'WAVE',                          /* format audio */
           :hvExt_Stor,                     /*conservation dans un fichier du serveur*/
           'Voix d''Anne Dupont')           /*commentaire*/
      );
```

Figure 12. Insertion de données dans une table

Résultat obtenu : En réponse à l'appel de la fonction UDF DB2Image dans l'instruction INSERT, l'extension Image exécute les opérations suivantes :

- Lecture des attributs de l'image (tels que la hauteur, la largeur et le nombre de couleurs) à partir de l'en-tête du fichier image source.
- Création d'un descripteur unique désignant l'image, et enregistrement des données ci-dessous dans une table de gestion :
  - descripteur de l'image,
  - horodatage,
  - taille en octets de l'image,
  - commentaire «Photo d'Anne Dupont»,
  - contenu de l'image.

  L'image source est stockée sur le serveur, dans un fichier appelé adupont.bmp. Le contenu de ce fichier est inséré sous la forme d'un BLOB dans l'entrée créée dans la table de gestion. Le format de l'image ainsi stockée est le même que celui de l'image source. Aucune conversion de format n'est effectuée.
- Stockage d'une entrée dans une autre table de gestion. Cette entrée contient des attributs graphiques spécifiques de l'image (nombre de couleurs, par exemple), ainsi qu'une version miniature de l'image.

En réponse à l'appel de la fonction UDF DB2Audio dans l'instruction INSERT, l'extension Audio exécute les opérations suivantes :

- Lecture des attributs de l'échantillon vocal (tels que le nombre de pistes et de canaux audio) à partir de l'en-tête du fichier audio.

- Création d'un descripteur unique désignant l'échantillon vocal.
- Stockage d'une entrée dans une autre table de gestion. Cette entrée contient les éléments suivants :
  - descripteur de la séquence audio,
  - horodatage,
  - taille en octets de la séquence audio,
  - commentaire «Voix d'Anne Dupont».

  La séquence audio source est stockée sur le serveur, dans un fichier appelé adupont.wav. L'entrée créée dans la table de gestion pointe sur ce fichier.
- Stockage d'une entrée dans une autre table de gestion. Cette entrée contient des attributs audio spécifiques, tels que la fréquence d'échantillonnage de l'enregistrement audio.

Des déclencheurs insèrent la valeur des attributs de l'image et de la séquence audio dans diverses tables de gestion.

Sélection de données dans une table

Un utilisateur interroge la base de données afin de déterminer à quand remonte le stockage de la photo et de l'échantillon vocal associés à Robert Durand.

Opérations effectuées par l'utilisateur : L'utilisateur obtient les informations recherchées, à l'aide d'un programme d'application comportant les instructions SQL présentées à la figure 13.

```
EXEC SQL BEGIN DECLARE SECTION;
char[255]  hvImg_Time;
char[255]  hvAud_Time;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT IMPORTTIME(PICTURE),  /*horodatage de stockage de la photo*/
       IMPORTTIME(VOICE)              /*horodatage de stockage de l'enregistrement*/
       INTO :hvImg_Time, :hvAud_Time
       FROM EMPLOYEE
       WHERE NAME='Robert Durand';
```

Figure 13. Sélection de données dans une table

Résultat obtenu : En réponse à l'appel de la fonction UDF ImportTime portant sur la colonne Photo, l'extension Image renvoie un horodatage indiquant la date et l'heure de stockage de la photo. En réponse à l'appel de la fonction UDF ImportTime portant sur la colonne *Son*, l'extension Audio renvoie un horodatage indiquant la date et l'heure de stockage de l'échantillon vocal.

Affichage et lecture d'objets

Un utilisateur souhaite afficher la photo de Robert Durand et entendre l'échantillon vocal correspondant . La photo est stockée sous la forme d'un objet BLOB dans la table *Employés*, et l'échantillon vocal dans un fichier du serveur.

Opérations effectuées par l'utilisateur : L'utilisateur affiche l'image et écoute l'enregistrement audio à l'aide d'un programme d'application comportant les instructions SQL présentées à la figure 14.

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_hdl [251];
char hvAud_hdl [251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT PICTURE,                    /*Extraction du descripteur*/
                                            /*de l'image*/
                VOICE                       /*Extraction du descripteur de*/
                                            /*la séquence audio*/
       INTO :hvImg_hdl, :hvAud_hdl
       FROM EMPLOYEE
       WHERE NAME='Robert Durand';

rc=DBiBrowse(
        NULL,                               /*Utilisation de l'afficheur graphique*/
                                            /*par défaut*/
        MMDB_PLAY_HANDLE,                   /*Utilisation du descripteur*/
        hvImg_hdl,                          /*Descripteur de l'image*/
        MMDB_PLAY_NO_WAIT);                 /*Exécution indépendante de l'afficheur*/

rc=DBaPlay(
        NULL,                               /*Utilisation du programme de*/
                                            /*lecture audio par défaut*/
        MMDB_PLAY_HANDLE,                   /*Utilisation du descripteur*/
        hvAud_hdl,                          /*Descripteur de la séquence audio*/
        MMDB_PLAY_WAIT);                    /*Attente de l'arrêt du programme*/
                                            /*audio avant poursuite*/
```

Figure 14. Affichage et lecture d'objets

Résultat obtenu : DB2 commence par extraire le descripteur de la photo et de l'échantillon vocal de Robert Durand. Ensuite, en réponse à l'appel de l'API DBiBrowse, l'extension Image extrait le contenu de l'image associée au descripteur fourni par DB2. L'extraction s'effectue à partir de la base de données et l'image est placée dans un fichier client temporaire pour en permettre l'affichage. Le paramètre NULL indique que le programme d'affichage graphique défini par défaut sur le poste de l'utilisateur sera utilisé. L'exécution du programme d'affichage s'effectue indépendamment du programme appelant. L'exécution de ce dernier peut donc se poursuivre sans qu'il soit nécessaire d'attendre l'arrêt du programme d'affichage.

En réponse à l'appel de l'API DBaPlay, l'extension Audio extrait le nom du fichier audio associé au descripteur fourni par DB2 et le transmet au programme de lecture audio. Le paramètre NULL indique que le programme

de lecture audio défini par défaut sur le poste de l'utilisateur doit être utilisé. Le programme appelant attendra en revanche l'arrêt du programme audio avant de continuer.

Mise à jour des données d'une table

Anne Dupont dispose d'une photo plus récente et souhaite réactualiser la table *Employés*. La nouvelle photo est stockée dans un fichier du serveur.

Opérations effectuées par l'utilisateur : L'utilisateur remplace la photo stockée dans la table Employés à l'aide d'un programme d'application comportant les instructions SQL présentées à la figure 15.

```
EXEC SQL BEGIN DECLARE SECTION;
  char hvComment [16385];
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

strcpy(hvComment. "Photo d'Anne Dupont prise lors de sa promotion");
hvStorageType=MMDB_STORAGE_TYPE_INTERNAL;

EXEC SQL UPDATE EMPLOYEE
       SET PICTURE=REPLACE(
              PICTURE                       /*descripteur de l'image*/
              '/images/photo2.bmp',         /*fichier image source*/
                 'BMP',                     /*format source*/
              :hvStorageType,               /*stockage de l'image sous forme de BLOB*/
              :hvComment)                   /*remplacement du commentaire*/
       WHERE NAME='Anne Dupont';
```

Figure 15. Mise à jour des données d'une table

Résultat obtenu : En réponse à l'appel de la fonction UDF Replace dans l'instruction UPDATE, l'extension Image lit les attributs de la nouvelle photo. L'extension Image utilise les attributs de la nouvelle photo pour mettre à jour les valeurs stockées pour la photo précédente dans les tables de gestion. L'image source est stockée sur le serveur, dans un fichier appelé photo2.bmp. Le contenu de ce fichier est inséré sous la forme d'un objet BLOB dans l'entrée de la table de gestion, à la place de l'objet BLOB précédent.

Des déclencheurs remplacent la valeur des attributs de l'image dans diverses tables de gestion.

Suppression de données d'une table

Un utilisateur supprime l'entrée correspondant à Anne Dupont dans la table *Employés*.

Opérations effectuées par l'utilisateur : L'utilisateur supprime l'entrée de la table Employés à l'aide d'un programme d'application comportant l'instruction SQL suivante :

```
DELETE FROM EMPLOYEE
       WHERE NAME='Anne Dupont';
```

Résultat obtenu : Les déclencheurs suppriment toutes les entrées associées à Anne Dupont dans les diverses tables de gestion.

Partie 2. Données image, audio et vidéo - Administration

Chapitre 4. Administration - Présentation

Ce chapitre fournit une vue d'ensemble des tâches d'administration liées à la création d'applications qui utilisent DB2 Extensions.

La famille DB2 Extensions permet d'effectuer des tâches d'administration de deux manières :

- Interfaces de programmation d'applications (API). Vous pouvez inclure les API DB2 Extensions dans votre programme en langage C. Pour plus d'informations sur ces API, reportez-vous au «Chapitre 16. Interfaces de programmation d'applications» à la page 279.
- Commandes d'administration. Vous pouvez soumettre des commandes d'administration à l'interpréteur de commandes db2ext. Ces commandes ne peuvent pas être exécutées sur la ligne de commande DB2. Reportez-vous au «Chapitre 17. Commandes d'administration du poste client» à la page 487 pour savoir comment entrer des commandes d'administration et pour plus d'informations.

Tâches d'administration exécutables avec DB2 Extensions

Les tâches d'administration se divisent en cinq catégories :

- Gestion des fonctions d'extension. Les extensions DB2 fonctionnent sur leurs propres serveurs, à partir de DB2. Pour que les applications puissent utiliser les données d'extensions, l'administrateur système doit démarrer les fonctions d'extension, et l'utilisateur, se connecter à une base de données qui contient les données d'extensions.
- Préparation des objets de données pour les données d'extensions. Pour préparer les bases de données, les tables et les colonnes devant recevoir les données d'extensions, vous devez les activer. Lorsque vous activez un objet de données, les extensions créent et tiennent à jour les tables de gestion (appelées également tables de métadonnées) afin de gérer les données d'extensions.
- **Produit EEE uniquement.** Redistribution des données d'extensions dans un environnement partitionné. Lors de l'ajout ou de la suppression de partitions d'une base de données partitionnée, vous pouvez redistribuer les données pour tirer parti de la nouvelle configuration du noeud.
- Recherche d'objets de données et de fichiers de support. Lorsque vous déboguez les applications qui utilisent les extensions DB2, il est utile de connaître les objets de données qui sont activés pour les données d'extensions. Il est également utile de comprendre la corrélation existant entre les tables utilisateur et les fichiers de support externes.

- Nettoyage des tables de gestion. Lorsque vous utilisez les extensions DB2, des entrées obsolètes finissent par s'accumuler dans les tables de gestion. La suppression des métadonnées périmées peut permettre d'améliorer les performances et de récupérer de l'espace mémoire.

Le tableau 3 répertorie toutes les tâches qu'implique l'administration de données d'extensions. Il indique les outils qui sont fournis pour l'exécution de chaque tâche, ainsi que l'endroit où se trouvent les informations.

Dans la colonne **API d'extension**, `x` représente le troisième caractère de chaque instruction d'API. Ce caractère varie en fonction de l'extension que vous utilisez :

Caractère	Extension
a	Audio
i	Image
v	Vidéo

Par exemple, l'API permettant d'activer une table de données d'image est DB**i**EnableTable ; l'API permettant d'activer une table de données audio est DB**a**EnableTable et l'API permettant d'activer une table de données vidéo est DB**v**EnableTable. La valeur `Non` dans la colonne API d'extension signifie qu'il n'existe pas d'API d'extension pour la tâche correspondante. La valeur `Non` dans la colonne Commande d'extension signifie qu'il n'existe pas de commande d'extension pour la tâche correspondante.

La recherche QBIC exige plus de tâches d'administration : Si vous envisagez d'utiliser la fonction de recherche QBIC, vous devrez effectuer des tâches d'administration supplémentaires, telles que la création d'un catalogue QBIC. Pour plus d'informations sur ces tâches, reportez-vous au «Chapitre 13. Recherche d'images par contenu» à la page 137.

Tableau 3. Tâches et fonctions d'administration de DB2 Extensions

Tâche	API d'extension	Commande d'extension	Voir
Gestion des fonctions d'extension			
Démarrage des fonctions d'extension	Non	DMBSTART	p. 49
Etat des fonctions d'extension	Non	DMBSTAT	p. 52
Arrêt des fonctions d'extension	Non	DMBSTOP	p. 51

Tableau 3. Tâches et fonctions d'administration de DB2 Extensions (suite)

Tâche	API d'extension	Commande d'extension	Voir
Connexion à une base de données	Non	CONNECT	p. 49
Démarrage d'une fonction d'extension pour votre base de données	Non	START SERVER	p. 51
Etat de la fonction d'extension pour votre base de données	Non	GET SERVER STATUS	p. 52
Arrêt d'une fonction d'extension pour votre base de données	Non	STOP SERVER	p. 51
Création et gestion d'instances d'extension	Non	DMBICRT, DMBILIST, DMBIDROP, DMBIMIGR	p. 52
Préparation des objets de données pour des données multimédia			
Activation d'une base de données	DBxEnableDatabase	ENABLE DATABASE	p. 57
Désactivation d'une base de données	DBxDisableDatabase	DISABLE DATABASE	p. 64
Activation d'une table	DBxEnableTable	ENABLE TABLE	p. 60
Désactivation d'une table	DBxDisableTable	DISABLE TABLE	p. 64
Activation d'une colonne	DBxEnableColumn	ENABLE COLUMN	p. 63
Désactivation d'une colonne	DBxDisableColumn	DISABLE COLUMN	p. 64
Redistribution des données d'extensions dans un environnement partitionné (Produit EEE uniquement)			
Redistribution des données d'extensions sur la base d'une nouvelle configuration de groupe de noeuds	DMBRedistribute	REDISTRIBUTE NODEGROUP	p. 65
Recherche d'objets de données et de fichiers de support			
Vérification de l'activation des bases de données	DBxIsDatabaseEnabled	GET EXTENDER STATUS	p. 67
Vérification de l'activation des tables	DBxIsTableEnabled	GET EXTENDER STATUS	p. 67
Vérification de l'activation des colonnes	DBxIsColumnEnabled	GET EXTENDER STATUS	p. 67

Tableau 3. Tâches et fonctions d'administration de DB2 Extensions (suite)

Tâche	API d'extension	Commande d'extension	Voir
Recherche des entrées de table faisant référence à des fichiers se trouvant dans des tables dont le qualificatif est l'ID utilisateur en cours	DBxIsFileReferenced	Non	p. 68
Recherche des entrées de table faisant référence à des fichiers se trouvant dans toutes les tables ayant un qualificatif spécifique ou dans toutes les tables de la base de données	DBxAdminIsFileReferenced	Non	p. 68
Recherche des fichiers identifiés par des entrées de tables dont le qualificatif est l'ID utilisateur en cours	DBxGetReferencedFiles	GET REFERENCED FILES	p. 69
Recherche des fichiers identifiés par des entrées de table se trouvant dans toutes les tables ayant un qualificatif spécifique ou dans toutes les tables de la base de données	DBxAdminGetReferencedFiles	GET REFERENCED FILES	p. 69
Recherche des fichiers inaccessibles identifiés par les entrées de toutes les tables dont le qualificatif est l'ID utilisateur en cours	DBxGetInaccessibleFiles	GET INACCESSIBLE FILES	p. 70
Recherche des fichiers inaccessibles identifiés par les entrées de tables se trouvant dans toutes les tables ayant un qualificatif spécifique ou dans toutes les tables de la base de données	DBxAdminGetInaccessibleFiles	GET INACCESSIBLE FILES	p. 70
Nettoyage de la table de gestion (métadonnées)			
Nettoyage des tables de métadonnées pour une table utilisateur spécifique ou toutes les tables utilisateur dont le qualificatif est l'ID utilisateur en cours.	DBxReorgMetadata	REORG	p. 73

Tableau 3. Tâches et fonctions d'administration de DB2 Extensions (suite)

Tâche	API d'extension	Commande d'extension	Voir
Nettoyage des tables de métadonnées pour toutes les tables utilisateur ayant un qualificatif spécifique ou toutes les tables d'une base de données	DBxAdminReorgMetadata	REORG	p. 73

Séquence de tâches d'administration : La liste ci-après récapitule les tâches d'administration que vous effectuez lorsque vous utilisez les extensions pour la première fois. Vous utilisez des commandes ou des instructions DB2 pour effectuer certaines tâches. D'autres sont exécutées à l'aide des extensions DB2. Pour exécuter les tâches ci-après, votre système DB2 doit être en cours d'exécution.

Tâches obligatoires :

1. Démarrage des fonctions d'extension.
2. Création d'une base de données (à l'aide de DB2).
3. Connexion à la base de donnéesau serveur de bases de données.
4. Activation de la base de données.
5. Création d'une table et d'une colonne (à l'aide de DB2).
6. Activation d'une table dans la base de données.
7. Activation d'une colonne dans la table.

Tâches facultatives :

1. Recherche d'objets de données et de fichiers de support.
2. Définition du chemin des fonctions (à l'aide de DB2).
3. Nettoyage des tables de gestion.

Exemples : La plupart des exemples figurant dans les cinq chapitres suivants traitent des tâches effectuées par l'administrateur système (SYSADM) ou l'administrateur de base de données (DBA). Vous pouvez néanmoins effectuer certaines tâches sans disposer des droits DBA ou SYSADM.

Les exemples supposent que l'administrateur de base de données (DBA) a ajouté le schéma MMDBSYS dans le chemin en cours des fonctions. Il peut ainsi spécifier des noms UDT sans les faire précéder d'un nom de schéma MMDBSYS. Pour plus d'informations sur les noms UDT, reportez-vous à la section «Dénomination des fonctions UDF et des types UDT» à la page 18.

Plusieurs exemples d'API figurant dans cette section sont basés sur le modèle de code d'application fourni avec les extensions. Le modèle de code se trouve dans le sous-répertoire EXEMPLES figurant sur le poste client.

Chapitre 5. Gestion des serveurs d'extensions

DB2 Extensions s'exécute dans un environnement client-serveur DB2. Cet environnement est composé d'un serveur de bases de données et d'un ou de plusieurs clients de base de données éloignés. Les fonctions de DB2 Extensions s'exécutent sur le serveur. Pour y accéder, vous devez les démarrer.

Une fois l'environnement configuré, vous pouvez arrêter et relancer les fonctions de DB2 Extensions à partir du client. Vous pouvez connaître l'état des extensions à partir du client ou du serveur.

Produit EEE uniquement : Dans un environnement multipartition, vous pouvez également ajouter et supprimer des partitions de bases de données.

Configuration des environnements de DB2 Extensions

Pour lancer les fonctions de DB2 Extensions, entrez la commande DMBSTART à partir de la ligne de commande du système d'exploitation, sur le serveur :

```
dmbstart
```

Cette commande permet le démarrage des extensions pour toutes les bases de données pouvant prendre en charge des données d'extensions. Cette commande lance également le système DB2, s'il n'est pas déjà en cours d'exécution. Pour exécuter la commande, vous devez disposer de droits SYSADM, SYSCTRL ou SYSMAINT. Sous AIX, vous devez être connecté en tant que propriétaire de l'instance de DB2 Extensions.

A ce stade, l'application en langage C peut accéder aux extensions via les interfaces API lorsque l'application se connecte à une base de données. De même, si vous souhaitez utiliser la ligne de commande db2ext, vous devez vous connecter à la base de données que vous souhaitez utiliser. La ligne de commande db2ext exige qu'une connexion indépendante de celle utilisée par la ligne de commande DB2 soit établie.

Ouvrez l'interpréteur de commandes db2ext sur le poste client et exécutez la commande CONNECT pour DB2 Extensions. Dans l'exemple qui suit, la commande permet d'établir une connexion à la base de données PERSONNL. Elle permet d'accéder à des tables ayant le qualificatif anne, au moyen du mot de passe ANPASS :

```
connect to personnl user anne using anpass
```

Produit EEE uniquement : Si vous utilisez les extensions DB2 dans un environnement de bases de données partitionnées, la commande DMBSTART lance les services d'extension sur tous les serveurs de partitions de bases de données définis pour l'instance. Si vous voulez démarrer les services d'extension sur un seul serveur de partitions de bases de données, il vous faut spécifier dans la commande le noeud que vous voulez démarrer. L'exemple ci-dessous illustre la commande à entrer pour démarrer les services d'extension au noeud numéro 2.

```
dmbstart nodenum 2
```

Produit EEE uniquement : Pour démarrer un serveur de partitions de base de données unique, vous devez lancer DB2 sur ce noeud.

A ce stade, vous pouvez exécuter les autres commandes de DB2 Extensions, répertoriées dans le «Chapitre 17. Commandes d'administration du poste client» à la page 487.

Ajout et suppression de partitions de bases de données (Produit EEE uniquement)

Pour toute utilisation des extensions dans un environnement de bases de données partitionnées, il faut que les partitions définies pour les extensions correspondent à celles définies pour DB2. La commande DMBSTART permet de lancer les serveurs d'extensions sur chacun des noeuds définis pour l'instance en cours. Le serveur détecte automatiquement si la création du noeud en cours d'exécution est récente et procède à toute initialisation éventuellement nécessaire. En cas de suppression d'un noeud de DB2, les fichiers d'extensions associés à ce noeud doivent faire l'objet d'une suppression manuelle.

Pour plus d'informations sur les commandes DB2 destinées à l'ajout et à la suppression de partitions, reportez-vous au manuel *IBM DB2 Universal Database Enterprise Extended Edition - Mise en route*.

Pour ajouter une partition de bases de données, procédez comme suit :

1. Créez une partition pour DB2 à l'aide de la commande DB2NCRT ou DB2START ADDNODE.
2. Créez une partition pour les extensions à l'aide de la commande d'extension DMBSTART NODENUM.
3. Redistribuez les données DB2 pour tirer parti de la configuration du nouveau noeud à l'aide de la commande DB2 REDISTRIBUTE NODEGROUP.
4. Redistribuez les données d'extensions pour tirer parti de la configuration du nouveau noeud à l'aide de la commande REDISTRIBUTE NODEGROUP.

Pour supprimer une partition de bases de données, procédez comme suit :

1. Redistribuez les données DB2 pour les supprimer de la partition à supprimer à l'aide de la commande DB2 REDISTRIBUTE NODEGROUP.
2. Redistribuez les données d'extensions pour les supprimer de la partition à supprimer à l'aide de la commande d'extensions REDISTRIBUTE NODEGROUP.
3. Supprimez une partition pour DB2 à l'aide de la commande DB2 DB2NDROP ou DB2STOP DROP.
4. Supprimez une partition pour les extensions à l'aide de la commande d'extension DMBSTART NODENUM.
5. Supprimez manuellement les fichiers d'extensions associés à la partition supprimée.

Les données d'extensions relatives à une partition de bases de données se trouvent dans un sous-répertoire appelé num*noeud*, où *num* est le numéro de noeud correspondant à la partition de bases de données. Ce sous-répertoire se trouve dans un répertoire spécifié par la valeur de la variable d'environnement DB2MMDATAPATH. Pour supprimer les données d'extensions relatives à une partition de bases de données supprimée, supprimez le sous-répertoire nom*noeud* approprié et tous les sous-répertoires inférieurs. (Pour plus d'informations sur DB2MMDATAPATH, reportez-vous à la section «Utilisation de la variable d'environnement DB2MMDATAPATH (Produit EEE uniquement)» à la page 587.)

Arrêt et démarrage des serveurs de DB2 Extensions

Lorsque vous démarrez des applications faisant appel aux fonctions de DB2 Extensions, le serveur reste actif jusqu'à ce que vous l'arrêtiez de manière explicite ou jusqu'à sa régénération. Vous pouvez arrêter tous les serveurs de DB2 Extensions en entrant la commande DMBSTOP à partir du serveur sur la ligne de commande du système d'exploitation.

Pour arrêter et redémarrer les extensions à partir du client, exécutez les commandes STOP SERVER et START SERVER sur la ligne de commande db2ext. Ces commandes entraînent l'arrêt, puis le redémarrage des extensions de la base de données en cours.

Produit EEE uniquement : Dans un environnement de bases de données partitionnées, DMBSTART permet de démarrer l'ensemble des serveurs de partitions de bases de données définis pour l'instance ou un seul serveur de partitions de bases de données. Si aucun paramètre n'est spécifié, DMBSTART lance tous les serveurs de partitions de bases de données. Si vous voulez en démarrer un seul, il vous faut spécifier dans la commande le noeud à démarrer, comme suit :

```
dmbstart nodenum 2
```

Arrêt et démarrage des serveurs

Une fois le serveur démarré sur un noeud particulier, vous devez reconnecter le serveur en question à la base de données. Utilisez la commande d'extension RECONNECT SERVER, comme suit :

```
reconnect server at nodenum 2
```

Produit EEE uniquement : Si vous utilisez les extensions DB2 dans un environnement de bases de données partitionnées et si aucun paramètre n'est spécifié, DMBSTOP arrête l'ensemble des serveurs de partitions de bases de données définis pour l'instance. Si vous voulez en arrêter un seul, vous devez commencer par le déconnecter de la base de données. Utilisez la commande d'extension DISCONNECT SERVER, comme suit :

```
disconnect server at nodenum 2
```

Vous pouvez alors exécuter DMBSTOP en spécifiant dans la commande le noeud à arrêter. L'exemple ci-dessous illustre la commande à entrer pour arrêter les services d'extension au noeud numéro 2.

```
dmbstop nodenum 2
```

Produit EEE uniquement : N'exécutez pas DMBSTOP sur un noeud particulier tant que votre base de données ne fonctionne pas en mode maintenance. De plus, vous devez vous assurer qu'aucune activité d'extensions n'est déclenchée sur ce noeud tant qu'il est désactivé. Dans le cas contraire, des erreurs imprévues risquent de se produire.

Affichage de l'état du serveur

A partir du serveur, vous pouvez afficher l'état du serveur de DB2 Extensions au moyen de la commande DMBSTAT. Par exemple, la commande ci-dessous permet de répertorier les bases de données actives et d'indiquer si les extensions sont en cours d'exécution. Connectez-vous au serveur avant d'exécuter cette commande.

```
dmbstat
```

A partir du client, vous pouvez connaître l'état du serveur de DB2 Extensions pour une base de données en utilisant la commande GET SERVER STATUS. Par exemple, la commande ci-dessous permet de connaître l'état de la base de données PERSONNL :

```
get server status personnl
```

Création et gestion d'instances multiples

Vous pouvez créer et utiliser plusieurs instances du serveur DB2 Extensions. Vous devez en créer plusieurs si vous avez créé plusieurs instances du serveur DB2. Chaque instance du serveur DB2 Extensions est associée à une instance du serveur DB2 et porte le même nom. Vous pouvez également répertorier les

instances du serveur DB2 Extensions disponibles sur un système, exécuter plusieurs instances simultanément ou en supprimer.

Création de plusieurs instances du serveur DB2 Extensions

Une instance DB2 Extensions initiale ou par défaut est créée lorsque vous installez DB2 Extensions. Le nom de l'instance DB2 par défaut lui est attribué. Sous Windows et OS/2, l'instance DB2 Extensions par défaut s'appelle DB2. Sous UNIX, elle porte le nom qui était attribué à l'instance DB2 par défaut initiale. Pour créer des instances supplémentaires du serveur DB2 Extensions, vous devez disposer des droits SYSADMIN. Sous Unix, vous devez disposer des droits root (superutilisateur).

Utilisez la commande DMBICRT pour créer une instance supplémentaire du serveur DB2 Extensions Image, Audio et Vidéo. Pour créer une instance du serveur DB2 Extensions pour l'instance DB2 DEVINST, entrez la commande suivante sur la ligne de commande :

```
dmbicrt  devinst
```

Lorsque vous exécutez la commande DMBICRT, un sous-répertoire est créé pour l'instance et celle-ci est ajoutée à la liste des instances gérée par les extensions DB2.

Produit EEE uniquement :

- Sous Windows, l'instance du serveur DB2 Extensions par défaut s'appelle DB2MPP.
- Lorsque vous utilisez la commande DMBICRT pour créer des instances supplémentaires du serveur DB2 Extensions Image, Audio et Vidéo, vous devez préciser le répertoire utilisé par DB2 Extensions pour des opérations diverses dans un environnement de bases de données partitionnées. Il s'agit du répertoire spécifié dans la variable d'environnement DB2MMDATAPATH sous UNIX et dans le registre sous Windows. Ce répertoire doit être partagé et exister sur tous les noeuds de cette instance.
- Vous devez également préciser un intervalle de ports TCP/IP à utiliser sous Windows. Sous UNIX, cet intervalle doit être ajouté au fichier `/etc/services` (voir la section «DMBICRT» à la page 530).

Listage des instances

Exécutez la commande DMBILIST pour répertorier toutes les instances du serveur DB2 Extensions disponibles sur un système. Pour rechercher l'instance active, entrez la commande suivante :

```
echo  %DB2INSTANCE%    (sous Windows ou OS/2)

echo $DB2INSTANCE      (sous UNIX)
```

Création et gestion d'instances multiples

Exécution simultanée d'instances multiples

Pour exécuter simultanément plusieurs instances du serveur DB2 Extensions, procédez comme suit pour chaque instance :

Sous Windows ou OS/2

A partir de la ligne de commande :

1. Indiquez dans la variable DB2INSTANCE le nom de l'instance à démarrer :

   ```
   set db2instance=nom-instance
   ```

2. Démarrez les fonctions d'extension.

Sous UNIX

1. Connectez-vous en tant que propriétaire de l'instance ou utilisateur disposant des droits d'administration pour cette instance.
2. Configurez l'environnement.
3. Démarrez le gestionnaire de bases de données.

Définition de l'instance en cours

Lorsque vous exécutez les commandes de démarrage ou d'arrêt des fonctions d'une instance, elles s'appliquent à l'instance en cours. Spécifiez l'instance du serveur DB2 Extensions à utiliser en indiquant son nom dans la variable d'environnement DB2INSTANCE.

Suppression d'instances

Pour supprimer une instance DB2 Extensions, procédez comme suit :

1. Mettez fin à toutes les applications utilisant cette instance.
2. Arrêtez les fonctions d'extension et les sessions d'interpréteur de commandes db2ext à l'aide des commandes DMBSTOP et db2ext TERMINATE.
3. Sauvegardez les fichiers du répertoire d'instance DB2 Extensions que vous souhaitez conserver (par exemple, les fichiers du catalogue QBIC). Les fichiers contenus dans ce répertoire sont effacés lors de la suppression de l'instance.
4. Entrez la commande DMBIDROP pour l'instance à supprimer. Par exemple, pour supprimer l'instance DEVINST, entrez :

   ```
   dmbidrop devinst
   ```

La suppression d'une instance DB2 Extensions à l'aide de la commande DMBIDROP ne supprime pas l'instance DB2 associée. Cette dernière doit être supprimée séparément. Si vous supprimez l'instance DB2 associée à une instance DB2 Extensions, celle-ci n'est pas supprimée. Cependant, vous ne pouvez plus l'utiliser.

Migration d'instances

Sous UNIX, une fois la nouvelle version de DB2 UDB et DB2 Extensions installée, vous devez effectuer la migration des instances DB2 Extensions.

Pour faire migrer des instances DB2 Extensions existantes créées à l'aide d'une version antérieure :

1. Faites migrer l'instance DB2 UDB associée à l'instance DB2 Extensions.
2. Entrez la commande DMBIMIGR pour faire migrer l'instance. Par exemple, pour faire migrer l'instance OLDINST, entrez :

```
dmbimigr  oldinst
```

Création et gestion d'instances multiples

Chapitre 6. Préparation des objets de données pour les données d'extensions

Pour préparer les bases de données, les tables et les colonnes devant recevoir les données d'extensions, vous devez les activer. Commencez par activer la base de données, puis activez une table dans celle-ci. Pour finir, activez une colonne dans la table.

Lorsque vous n'avez plus besoin de données d'extensions dans vos objets de données, vous pouvez désactiver les objets.

Vous pouvez activer et désactiver des objets au moyen des interfaces API du programme en langage C ou à partir de la ligne de commande db2ext. Dans le présent chapitre, des exemples sont fournis pour chaque méthode.

Activation des bases de données

Utilisez l'API DBxEnableDatabase (où `x` correspond à `a` pour Audio, `i` pour Image ou `v` pour Vidéo) ou la commande ENABLE DATABASE pour activer une base de données pour une extension DB2.

Lorsque vous activez une base de données, l'extension effectue les opérations suivantes :

- Création d'un type utilisateur (UDT) intitulé DB2`xxxxx` pour les objets de données, où `xxxxx` correspond à `Image`, `Audio` ou `Vidéo`. Cet UDT est utilisé pour définir une colonne dans la table utilisateur contenant des descripteurs pour les objets de ce type.
- Création de tables de gestion (également appelées tables de métadonnées) pour la base de données. Il ne s'agit pas de tables utilisateur (dans lesquelles les utilisateurs stockent des données commerciales). Elles sont utilisées par les extensions pour gérer des données d'extensions. Ne les modifiez pas manuellement.
- Création des fonctions utilisateur (UDF) associées à l'extension. Les fonctions UDF ainsi créées sont répertoriées dans la section «Fonctions UDF» à la page 210.

Lors de l'activation d'une base de données, vous devez également spécifier les espaces table pour maintenir les tables de prise en charge de la gestion (et leurs index) pour la base de données. Un ou plusieurs espaces table spécifiés peuvent avoir une valeur nulle, auquel cas un espace table par défaut est utilisé.

Activation des bases de données

Pour activer une base de données, vous devez disposer du droit DBA.

Produit EEE uniquement : Lors de l'activation d'une base de données pour une extension dans un environnement partitionné, l'espace table spécifié doit être défini dans un groupe de noeuds comprenant tous les noeuds dans le système de bases de données partitionnées. Par ailleurs, l'espace table mentionné doit être situé dans le même groupe de noeuds que la table utilisateur.

Exemples

Dans les exemples qui suivent, une base de données est activée pour la prise en charge des données de type image, via l'espace table par défaut.

Utilisation de l'interface API : Le code représenté à la figure 16 permet d'établir une connexion à une base de données avant de l'activer. Cet exemple met en oeuvre l'interface DB2 CLI. Il comprend du code de configuration et de vérification des erreurs. Le modèle de programme complet se trouve dans le fichier ENABLE.C du sous-répertoire SAMPLES.

```
/*---- Configuration ------------------------------------------*/
#include <stdio.h>
#include <stlib.h>
#include <string.h>
#include "dmbimage.h"  /* prototypes de fonctions d'extension image (DBi) */
#include "utility.h"     /* utilitaires */

#define MMDB_ERROR_MSG_TEXT_LEN         1000
#define SERVER_IS_DB2390  (strcmp(dbms, "DB2")==0 || strcmp(dmbs, "DSN06010")==0)

int
main(int argc, char *argv[])
{
        SQLHENV henv = SQL_NULL_HENV;
        SQLHDBC hdbc = SQL_NULL_HDBC;
        SQLHSTMT hstmt = SQL_NULL_HSTMT;
SQLCHAR uid[18+1];
SQLCHAR pwd[30+1];
        SQLCHAR dbname[SQL_MAX_DSN_LENGTH+1];
        SQLCHAR buffer[500];
        SQL SMALLINT dbms_sz = 0;
        char    dbms[20];

        SQLRETURN rc = SQL_SUCCESS;
        SQLINTEGER sqlcode = 0;
        char errorMsgText[MMDB_ERROR_MSG_TEXT_LEN+1];
        char *program = "enable;
        char *step;
```

Figure 16. Modèle de code permettant d'activer une base de données (Numéro 1 de 3)

```
/*---- Message d'invite nom base de données, ID util. et mot de passe ----*/
        if (argc > 5) ||  (argc >=2 && strcmp(argv[1],"?")== 0))
          {
            printf("Syntax for enable - enabling a DB2 UDB database: \n"
                   "   enable database_name userid password\n");
      exit(0);
          }

        if (argc == 4) {
          strcpy((char *)dbname, argv[1]);
          strcpy((char *)uid   , argv[2]);
          strcpy((char *)pwd   , argv[3]);
          }
     else {
          printf("Enter database name:\n");
  gets((char *) dbName);
          printf("Enter userid:\n");
  gets((char *) uid);
          printf("Enter password:\n");
  gets((char *) pwd);
          }
/*---- Connexion à la base de données --------------------------------*/
         rc = cliInitialize(&henv, &hdbc, dbname, uid, pwd);
cliCheckError(henv, hdbc, SQL_NULL_HSTMT, rc);
         if (rc < 0) goto SERROR;

/*----- Application connectée à DB2/UDB ou à DB2/390 ? ----------------*/
         rc = SQLGetInfo(hdbc, SQL_DBMS_NAME, (SQLPOINTER) &dbms,
                         sizeof(dbms), &dbms_sz);
cliCheckError(henv, hdbc, SQL_NULL_HSTMT, rc);
         if (rc < 0) goto SERROR;
```

Figure 16. Modèle de code permettant d'activer une base de données (Numéro 2 de 3)

Activation des bases de données

```
/****** Activation du serveur pour l'extension image ******/
     if (!SERVER_IS_DB2390)
           {
            printf("%s: Enabling database......\n", program);
           }
         printf("%s: This may take a few minutes, please wait......\n", program);

     if (!SERVER_IS_DB2390)
           {
            step="DBiEnableDatabase with NULL tablespace"
            rc=DBiEnableDatabase(NULL);
           }
if (rc < 0) {
                printf("%s: %s failed!\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
           if (sqlcode)
             printf("sqlcode=%i, ",sqlcode);
} else if (rc > 0) {
           printf("%s: %s, warning detected.\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
            printf("warning MsgText=%s\n", errorMsgText);
     } else
           printf("%s: %s OK\n", program, step);
/****** Fin de l'activation du serveur ************************/
```

Figure 16. Modèle de code permettant d'activer une base de données (Numéro 3 de 3)

Utilisation de la ligne de commande db2ext : Dans cet exemple, la base de données est déjà connectée.

```
enable database for db2image
```

Activation de tables

Pour activer une table pour une extension DB2, utilisez l'API DBxEnableTable (où x correspond à a pour Audio, i pour Image ou v pour Vidéo) ou la commande ENABLE TABLE.

Lors de l'activation d'une table utilisateur, vous devez également spécifier les espaces table devant contenir les tables de gestion associées (et leurs index). Un ou plusieurs espaces table spécifiés peuvent avoir une valeur nulle, auquel cas un espace table par défaut est utilisé.

Produit EEE uniquement : Lors de l'activation d'une table pour une extension dans un environnement partitionné, l'espace table spécifié doit être défini dans un groupe de noeuds comprenant tous les noeuds du système de bases de données partitionnées. Par ailleurs, l'espace table mentionné doit être situé dans le même groupe de noeuds que la table utilisateur.

Produit EEE uniquement : Vous ne pouvez pas utiliser une colonne d'extension DB2 comme colonne de clés de partitionnement dans un environnement de bases de données partitionnées.

Vous devez disposer du droit Control ou Alter sur la table utilisateur. La base de données doit être active pour que vous puissiez activer l'une de ses tables.

Dans les exemples ci-après, une table est activée pour la gestion des données de type image, via l'espace table par défaut. La base de données est déjà active.

Utilisation de l'interface API : Dans la figure 17 à la page 62 le code permet de créer la table et de valider les modifications. L'exemple comprend le code de vérification des erreurs. Le modèle de programme complet se trouve dans le fichier ENABLE.C du sous-répertoire SAMPLES.

```
        SQLCHAR szCreate_DB2UDB[]="CREATE TABLE %s(%s mmdbsys.DB2Image,
        %s mmdbsys.DB2Video, %s mmdbsys.DB2Audio, artist varchar(25),
        title varchar(25), stock_no char(11), tw char(10), price char(10))";

        SQLRETURN rc = SQL_SUCCESS;
        SQLINTEGER sqlcode = 0;
        char errorMsgText[MMDB_ERROR_MSG_TEXT_LEN+1];
        char tableName[8+18+1] = "sobay_catalog";
        char audioColumn[18+1] = "music";
char imageColumn[18+1] = "covers";
        char videoColumn[18+1] = "video";
        char *program = "enable";
        char *step;

/*-----création d'une table
------------------------------------------------*/
       printf("%s: Creating table ......\n", program);
     if (!SERVER_IS_DB2390)
         sprintf((char*) buffer, (char*) szCreate_DB2UDB,
                   tableName, imageColumn, videoColumn, audioColumn):

rc = SQLAllocStmt(hdbc, &hstmt);
cliCheckError(SQL_NULL_HENV, hdbc, SQL_NULL_HSTMT, rc);
       rc = SQLExecDirect(hstmt, buffer, SQL_NTS);
         cliCheckError(SQL_NULL_HENV, SQL_NULL_HDBC, hstmt, rc);

/*---- Activation d'une table pour l'extension Image ----------*/
        printf("%s: Enabling table......\n", program);
step="DBiEnableTable";
     if (!SERVER_IS_DB2390)
         rc = DBiEnableTable(NULL,tableName);
           }
if (rc < 0) {
               printf("%s: %s failed!\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
           if (sqlcode)
                 printf("sqlcode=%i, "sqlcode");
               printf("errorMsgText=%s\n", errorMsgText);
} else if (rc > 0) {
           printf("%s: %s, warning detected.\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
           printf("warningMsgText=%s\n", errorMsgText);
     } else
               printf("%s: %s OK\n", program, step)
/*---- Fin de l'activation de la table --------------------------*/
```

Figure 17. Modèle de code permettant d'activer une table

Utilisation de la ligne de commande DB2 : Dans cet exemple, la table existe déjà et la base de données est active.

```
enable table Employés for db2image
```

Activation de colonnes

Pour activer une colonne pour une extension DB2, utilisez l'API DBxEnableColumn (où `x` correspond à `a` pour Audio, `i` pour Image ou `v` pour Vidéo) ou la commande ENABLE COLUMN. Lorsque vous lancez l'API ou la commande, vous indiquez la table et la colonnes pertinentes.

Lorsque vous activez une colonne, l'extension ajoute des informations aux tables de gestion appartenant à la table utilisateur. Vous devez disposer du droit Control ou Alter sur la table utilisateur dans laquelle se trouve la colonne. La base de données et la table doivent être actives pour que la colonne puisse être activée.

Dans les exemples ci-après, la colonne Photo de la table Employés est activée pour la gestion des données de type image. La base de données et la table sont déjà actives.

Utilisation de l'interface API : Cet exemple comprend le code de vérification des erreurs. Le modèle de programme complet se trouve dans le fichier ENABLE.C du sous-répertoire SAMPLES.

Activation de colonnes

```
char imageColumn[18+1] = "covers";

/*---- Activation d'une colonne pour l'extension image  ----*/
     printf("%s: Enabling columns......\n", program);
step="DBiEnableColumn";
     rc = DBiEnableColumn(tableName, imageColumn);
if (rc < 0) {
                printf("%s: %s failed!\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
           if (sqlcode)
              printf("sqlcode=%i, ", sqlcode);
           printf("errorMsgText=%s\n", errorMsgText)

} else if (rc > 0) {
           printf("%s: %s, warning detected.\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
           printf("warningMsgText=%s\n", errorMsgText);
     } else
           printf("%s: %s OK\n", program, step);
/*---- Activation d'une colonne pour l'extension image  ----*/
```

Figure 18. Modèle de code permettant d'activer une colonne

Utilisation de la ligne de commande db2ext : Dans cet exemple, la colonne existe déjà, la base de données et la table sont actives.

```
enable column Employés photo for db2image
```

Désactivation des objets de bases de données

Si vous supprimez des données d'extensions d'une base de données, une table ou une colonne, il n'est plus nécessaire que ces objets soient actifs. Vous pouvez les désactiver à l'aide de la commande DISABLE ou des interfaces API. Pour plus d'informations sur les commandes des extensions, reportez-vous au «Chapitre 17. Commandes d'administration du poste client» à la page 487. Pour plus d'informations sur les API des extensions, reportez-vous au «Chapitre 16. Interfaces de programmation d'applications» à la page 279.

Avant de supprimer une table ou une base de données contenant des données d'extensions, désactivez-la et arrêtez le serveur de cette base de données.

Chapitre 7. Redistribution des données d'extensions dans un environnement de bases de données partitionnées (Produit EEE uniquement)

DB2 Extended Enterprise Edition vous permet d'ajouter et de supprimer des serveurs de partitions de bases de données (également appelés noeuds) dans un environnement de bases de données partitionnées. Une fois les noeuds ajoutés (ou avant leur suppression), vous pouvez redistribuer les données existantes pour tirer parti de la nouvelle configuration.

Deux étapes sont nécessaires à la redistribution des données d'extensions. En premier lieu, vous devez redistribuer les données DB2. Vous pouvez ensuite redistribuer les données d'extensions DB2.

Redistribution de données DB2

Avant de redistribuer les données, vous devez redistribuer les données DB2 à l'aide de la commande DB2 REDISTRIBUTE NODEGROUP.

Pour plus d'informations sur la redistribution des données DB2, reportez-vous au manuel *DB2 Administration Guide*.

Redistribution de données d'extensions

Une fois les données DB2 redistribuées, vous pouvez passer à la redistribution des données d'extensions. Entrez la commande d'extension REDISTRIBUTE NODEGROUP pour démarrer la redistribution des données d'extensions.

```
redistribute nodegroup
```

La commande `REDISTRIBUTE NODEGROUP` redistribue les données d'extension audio, image et vidéo, ainsi que les données présentant les caractéristiques QBIC, en les plaçant sur le même noeud que celui des données utilisateur correspondantes.

Si le processus de redistribution renvoie une erreur, vous pouvez relancer la commande, avec ou sans le paramètre CONTINUE, selon les instructions fournies par la réponse de la commande. Cette option permet au système de reprendre le traitement là où il s'était arrêté plutôt que de tout recommencer dès le début. Le paramètre CONTINUE ne peut pas être utilisé lors du premier lancement de la commande REDISTRIBUTE NODEGROUP après l'exécution de la commande DB2 REDISTRIBUTE NODEGROUP.

Redistribution de données

Pour conserver l'intégrité des données, procédez à la redistribution d'un groupe de noeuds à la fois. Attendez qu'un groupe de noeuds ait terminé la redistribution avant d'en démarrer un autre.

Connectez-vous à la base de données avant d'utiliser cette commande.

Vous devez disposer du droit SYSADM ou DBADM pour lancer cette commande.

Chapitre 8. Recherche d'objets de données et de fichiers de support

Lorsque vous créez et déboguez des applications utilisant les extensions DB2, il est judicieux de connaître les objets de données activés pour les données d'extensions. Par exemple, si vous pouvez déterminer qu'une table particulière est active pour des données de type image, l'application peut alors stocker les fichiers d'image dans cette table.

Il est également utile de comprendre la corrélation existant entre les tables utilisateur et les fichiers de support externes et de savoir, par exemple, quelles tables font référence à un fichier particulier ou quels fichiers sont référencés par une table donnée. Par ailleurs, il est utile de vérifier si des tables font référence à des fichiers qui n'existent plus dans le système.

Privilèges appropriés nécessaires : Pour pouvoir rechercher des données dans une table, vous devez disposer du droit d'accès sur cette dernière. Pour effectuer des opérations de recherche globale (par exemple retrouver quelles entrées, dans toutes les tables utilisateur de la base de données font référence à un fichier), vous devez disposer du droit SYSADM, du droit DBADM ou du privilège SELECT sur les colonnes actives dans toutes les tables utilisateur explorées et dans les tables de gestion associées. Si vous n'avez pas accès à toutes les tables, les extensions renvoient des informations uniquement sur les tables sur lesquelles vous disposez d'un droit d'accès. Elles renvoient également un code indiquant que vous n'êtes pas autorisé à accéder à certaines des tables requises.

Vérification de l'état des objets de données

Vous pouvez vérifier si les bases de données, les tables et les colonnes sont actives pour la prise en charge des données d'extensions.L'exemple ci-après permet de déterminer si la base de données en cours est active pour l'extension Image. La base de données est déjà connectée. Le modèle de programme complet se trouve dans le fichier API.C du sous-répertoire SAMPLES.

Utilisation de l'interface API : Le code de la figure 19 à la page 68 contient le code de vérification des erreurs.

```
/*-- Interrogation de la base de données via l'API DBiIsDatabaseEnabled. --*/
step="DBiIsDatabaseEnabled API";
rc = DBiIsDatabaseEnabled(&status);
if (rc < 0) {
        printf("%s: %s FAILED!\n", argv[0], step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
        printf("sqlcode=%i, errorMsgText=%s\n", sqlcode, errorMsgText);
        fail = TRUE;
} else if (rc > 0) {
        printf("%s: %s, warning detected.\n", argv[0], step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
        printf("sqlcode=%i, errorMsgText=%s\n", sqlcode, errorMsgText);
} else {
     if (status == 1) {
         printf("%s: \"%s\" database is enabled for Image Extender\n",
                argv[0], dbName);
   printf("%s: %s PASSED\n\n", argv[0], step);
     } else if (status == 0) {
         printf("%s: \"%s\" database is not enabled for Image Extender\n",
                argv[0], dbName);
   printf("%s: %s PASSED\n\n", argv[0], step);
     } else
         printf("%s: %s FAILED, invalid status!\n", argv[0], step);
}
```

Figure 19. Modèle de code permettant de vérifier si une base de données est active

Utilisation de la ligne de commande db2ext :

```
get extender status
```

La procédure de vérification de l'état des tables et des colonnes utilisateur est similaire à celle de l'état d'une base de données. Utilisez les interfaces API DBxIsTableEnabled et DBxIsColumnEnabled ou la commande GET EXTENDER STATUS.

Recherche d'entrées de tables faisant référence à des fichiers

Vous pouvez vérifier les entrées de table utilisateur faisant référence à un fichier de supports externes. Utilisez l'API DBxAdminIsFileReferenced API pour vérifier quelles entrées des tables ou de certaines tables de la base de données en cours font référence à un fichier de supports externes. Utilisez l'API DBxIsFileReferenced pour vérifier quelles entrées d'une table utilisateur particulière font référence à un fichier de supports externes.

Utilisation de l'interface API : Le modèle de code dans la figure 20 à la page 69 indique le nombre de fois et l'emplacement où le fichier est référencé.

Il inclut le code de vérification des erreurs. Le modèle de programme complet se trouve dans le fichier API.C du sous-répertoire SAMPLES.

```
/*-- Interrogation de la base de données via l'API DBiAdminIsFileReferenced --*/
step="DBiAdminIsFileReferenced API";
rc = DBiAdminIsFileReferenced((char*) uid, filename, &count, &filelist);
if (rc < 0) {
   printf("%s: %s FAILED!\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
        printf("sqlcode=%i, errorMsgText=%s\n", sqlcode, errorMsgText);
} else if (rc > 0) {
   printf("%s: %s, warning detected.\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
     printf("sqlcode=%i, errorMsgText=%s\n", sqlcode, errorMsgText);
} else {
     if (count == 0)
          printf("%s: \"%s\" file is not referenced\n",
                 program, filename);
     else {
          printf("%s: \"%s\" file is referenced %d times\n",
                 program, filename);
          for (i=0; i < count; i++)
             {
                 /* le nom de fichier est NULL pour les API IsFileReferenced */

   printf ("filename = %s\n", filelist[i].filename);
   printf ("\tqualifier = %s\n", filelist[i].tqualifier);
   printf ("\ttable = %s\n", filelist[i].tname);
   printf ("\thandle = %s\n", filelist[i].handle);
   printf ("\tcolumn = %s\n", filelist[i].column);
if (filelist[i].filename)
               free (filelist[i].filename);
            }
          }
   if (filelist)
       free (filelist);
        printf("%s: %s PASSED\n\n", argv[0], step);
}
```

Figure 20. Modèle de code permettant de vérifier si un fichier est référencé par des tables utilisateur

Recherche de fichiers référencés par des entrées de tables

Utilisez l'API DBxAdminGetReferencedFiles ou la commande GET REFERENCED FILES pour répertorier les fichiers de supports externes qui sont référencés par les tables utilisateur ou par certaines de ces tables dans la base de données en cours. Utilisez l'API DBxGetReferencedFiles ou la commande GET REFERENCED FILES pour répertorier les fichiers de supports externes qui sont référencés dans une table particulière.

Recherche de fichiers référencés

Utilisation de l'interface API : Le modèle de code de la figure 21 indique le nombre de fichiers trouvés et les répertorie. Le modèle de programme complet se trouve dans le fichier API.C du sous-répertoire SAMPLES.

```
/*-- Interrogation de la base de données via l'API DBiAdminGetReferencedFiles. --*/
step="DBiAdminGetReferencedFilesAPI"
rc = DBiAdminGetReferencedFiles((char*) uid, &count, &filelist);
if (rc < 0) {
   printf("%s: %s FAILED!\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
   printf{"sqlcode=%i, errorMsgText=%s\n", sqlcode, errorMsgText);
} else if (rc > 0) {
   printf("%s: %s, warning detected.\n", program, step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
        printf("sqlcode=%i, errorMsgText=%s\n", sqlcode, errorMsgText);
} else {
     if (count == 0)
     printf("%s: no referenced files\n", program);
     else {
     printf("%s: %d referenced files\n", program, count);
          for (i=0; i < count; i++)
       {
   printf ("filename = %s\n", filelist[i].filename);
   printf ("\tqualifier = %s\n", filelist[i].tqualifier);
   printf ("\ttable = %s\n", filelist[i].tname);
   printf ("\thandle = %s\n", filelist[i].handle);
   printf ("\tcolumn = %s\n", filelist[i].column);
if (filelist[i].filename)
               free (filelist[i].filename);
     }
   }
   if (filelist)
       free (filelist);
   printf("%s: %s PASSED\n\n", argv[0], step);
}
```

Figure 21. Modèle de code permettant d'obtenir une liste de fichiers référencés

Utilisation de la ligne de commande db2ext :

```
get referenced files user anitas for db2image
```

Vérification de l'existence des fichiers de support

Supposons qu'une personne supprime un fichier de support du système sans mettre à jour la table utilisateur y faisant référence. Vous pouvez répertorier l'ensemble des fichiers de support inaccessibles auxquels les tables utilisateur font référence.

Vérification des supports inaccessibles

Utilisez l'API DBxAdminGetInaccessibleFiles ou la commande GET INACCESSIBLE FILES pour répertorier les fichiers de supports inaccessibles qui sont référencés par les tables utilisateur ou par certaines de ces tables dans la base de données en cours. Utilisez l'API DBxGetInaccessibleFiles ou la commande GET INACCESSIBLE FILES pour répertorier les fichiers de supports inaccessibles qui sont référencés par une table particulière.

Vérification des supports inaccessibles

Chapitre 9. Nettoyage des tables de gestion

Lorsque vous utilisez les extensions DB2, des entrées obsolètes finissent par s'accumuler dans les tables de gestion. Un utilisateur peut supprimer un fichier de support sans supprimer les références à ce fichier dans la base de données. La suppression des métadonnées périmées peut permettre d'améliorer les performances et de récupérer de l'espace mémoire.

Utilisation de l'interface API : Le modèle de code de la figure 22 permet de nettoyer les métadonnées d'image pour toutes les tables utilisateur appartenant à la table ANITAS. Il inclut le code de vérification des erreurs. Le modèle de programme complet se trouve dans le fichier API.C dans le sous-répertoire SAMPLES.

```
/*-- Interrogation de la base de données via l'API DBiAdminReorgMetadata --*/
step="DBiAdminReorgMetadata API";
rc = DBiAdminReorgMetadata("anitas");
if (rc < 0) {
        printf("%s: %s FAILED!\n", argv[0], step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
     printf("sqlcode=%i, errorMsgText=%s\n", sqlcode, errorMsgText);
        fail = TRUE;
} else if (rc > 0) {
        printf("%s: %s, warning detected.\n", argv[0], step);
   printMsg(rc);
     DBiGetError(&sqlcode, errorMsgText);
     printf("sqlcode=%i, errorMsgText=%s\n", sqlcode, errorMsgText);
     } else
        printf("%s: %s PASSED\n\n", argv[0], step);

/*---- Fin de l'interrogation via l'API DBiAdminReorgMetadata ----*/
```

Figure 22. Modèle de code permettant de nettoyer des tables de gestion

Utilisation de la ligne de commande db2ext :

```
reorg database user anitas for db2image
```

Si vous n'êtes pas administrateur de base de données mais que vous disposez du droit CONTROL, vous pouvez utiliser l'interface API DBxReorgMetadata ou la commande REORG pour nettoyer des métadonnées correspondant aux tables vous appartenant.

Partie 3. Données image, audio et vidéo - Programmation

Chapitre 10. Programmation - Présentation

Le présent chapitre donne une vue d'ensemble de la programmation pour les Extensions DB2. Il fournit des informations qui vous seront utiles avant de commencer la programmation des Extensions, et présente un modèle d'application qui illustre le codage d'une extension.

Utilisation des UDF et des API d'extension

Les Extensions DB2 proposent des fonctions définies par l'utilisateur (UDF) permettant le stockage, l'accès et la manipulation de données image, audio et vidéo dans une base de données. Vous devez coder des requêtes pour ces fonctions UDF dans vos programmes d'applications, à l'aide des instructions SQL, de la même manière que vous demandez des fonctions SQL intégrées. Les fonctions UDF sont lancées sur le serveur de bases de données, comme les fonctions intégrées.

Les fonctions SQL suivantes, figurant dans un programme d'application écrit en C, requièrent une fonction UDF de l'extension Image appelée `DB2Image` pour le stockage d'une image dans la table de la base de données. Le contenu de l'image source se trouve dans le fichier du serveur :

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_INTERNAL

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',                          /*id*/
        'Anne Dupont',                     /*nom*/
        DB2IMAGE(                          /*UDF de l'extension Image*/
           CURRENT SERVER,                 /*base de données */
          '/Employés/images/adupont.bmp', /*contenu de l'image*/
           'ASIS',                         /*conservation du format de l'image*/
          :hvStorageType,                  /*stockage image dans BD en tant que BLOB*/
          'Photo d''Anne')                 /*commentaire*/
       );
```

Les API d'extension permettent l'affichage d'images et la lecture d'objets audio ou vidéo. Ces API peuvent être codées via des appels de fonctions client en langage C. Ces fonctions sont lancées sur le poste de travail client de votre base de données.

Les instructions en langage C qui suivent incluent l'API DBiBrowse. Cette API extrait les données d'un descripteur d'image et lance un navigateur afin d'afficher l'image :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_hdl [251];
EXEC SQL END DECLARE SECTION

EXEC SQL SELECT PICTURE INTO :hvImg_hdl
      WHERE NAME='Robert Durand';

rc=DBiBrowse(
             "ib %s",               /*afficheur d'image*/
             MMDB_PLAY_HANDLE,      /*utilisation du descripteur d'image*/
             hvImg_hdl,             /*descripteur d'image*/
             MMDB_PLAY_NO_WAIT);    /*exécution autonome de l'afficheur*/
```

Les fonctions UDF doivent s'exécuter sous l'ID utilisateur de l'instance : Les UDF d'extension DB2 doivent s'exécuter sous le même ID utilisateur que celui de l'instance d'extension DB2. De plus, si vous créez une instance d'extension DB2, ou si vous en utilisez une existante, les fonctions UDF doivent s'exécuter sous le même ID utilisateur que l'instance DB2.

DB2 doit être configuré correctement : Vous devez configurer DB2 correctement pour assurer le bon fonctionnement des extensions DB2, en particulier celui des fonctions UDF d'extension DB2. Veillez tout particulièrement à ce que le paramètre de configuration de la base de données, APP_CTL_HEAP_SZ, soit correctement défini.

Tâches pouvant être effectuées via des fonctions UDF et des API d'extension

Le tableau 4 répertorie les tâches pouvant être exécutées avec les UDF et les API d'extension et indique où trouver les descriptions des tâches.

Tableau 4. Tâches pouvant être effectuées via des API

Tâche	Voir
Stockage d'un objet image, audio ou vidéo	Page 94
Extraction d'un objet image, audio ou vidéo	Page 107
Extraction et utilisation d'attributs d'image, audio et vidéo	Page 113
Extraction de commentaires associés à un objet image, audio ou vidéo	Page 115
Mise à jour d'un objet image, audio ou vidéo	Page 116
Affichage d'un objet image	Page 131
Affichage d'une image miniature ou d'une trame vidéo	Page 134
Lecture d'un objet audio ou vidéo	Page 136

Tableau 4. Tâches pouvant être effectuées via des API (suite)

Tâche	Voir
Recherche d'images par contenu	Page 137
Détection de changements de plan vidéo	Page 179

Modèle de table pour une extension

Tout au long du présent chapitre, vous trouverez des exemples de programmation utilisant les extensions DB2. Ces exemples supposent que vous avez préalablement créé une table de base de données appelée Employés contenant des informations sur le personnel. Cette table comprend des colonnes pour le numéro matricule et le nom des employés. En fonction de l'extension utilisée, la table peut également inclure une colonne contenant les photos des employés, un message vocal et des séquences vidéo.

La figure 23 à la page 80 illustre la structure de la table Employés et indique les instructions SQL utilisées pour sa création.

```
CREATE TABLE  employés(
    matr.       CHAR(6),
    nom         VARCHAR(40),
    photo       DB2Image,

    son         DB2Audio,

    vidéo       DB2Video
    );
```

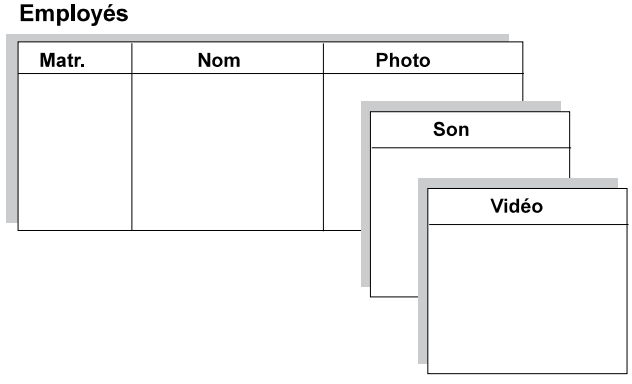


Figure 23. Table utilisée dans les exemples de programmation d'extensions DB2

Préalables à la programmation d'extensions DB2

Avant de développer un programme utilisant les extensions DB2, vous devez être familiarisé avec le processus de développement d'applications DB2 et les techniques de programmation décrites dans le manuel *DB2 Application Development Guide*. Le processus de développement de programmes utilisant les Extensions DB2 est pratiquement identique à celui utilisé pour les applications DB2 classiques.

Le code de votre programme d'application sera légèrement différent de celui utilisé pour une application DB2 classique, en raison de nouveaux types de données et des fonctions définies par les Extensions. Par exemple, la figure 24 à la page 82, illustre une application codée en langage C, utilisant l'extension Image pour l'identification d'images au format GIF stockées dans la table de base de données. Une fois les images identifiées, le programme fait appel à un afficheur d'images pour permettre leur visualisation.

Comme présenté dans l'exemple, une application utilisant une extension DB2 doit effectuer les fonctions suivantes :

1 Inclusion des définitions d'extension. Le fichier `dmbimage.h` utilisé dans l'exemple est le fichier d'inclusion (en-tête) pour l'extension Image. Il définit les constantes, les variables et les prototypes de fonction pour l'extension.

2 Définition des variables SQL en fonction des besoins pour qu'elles contiennent une entrée ou une sortie d'une fonction UDF ou une entrée pour un appel d'API. Dans l'exemple indiqué, `hvFormat`, `hvSize`, `hvWidth`, `hvHeight` et `hvComment` sont des variables SQL qui doivent contenir des données extraites par les fonctions UDF de l'extension Image. La variable SQL `hvImg_hdl` doit contenir un descripteur d'image défini en entrée d'appel API d'une extension Image.

3 Indication des requêtes de fonction UDF en fonction des besoins. Dans l'exemple indiqué, `SIZE`, `WIDTH`, `HEIGHT`, `COMMENT` et `FORMAT` sont des fonctions UDF de l'extension Image.

4 Indication des appels API en fonction des besoins. Dans l'exemple indiqué, `DBiBrowse` est un appel API vers une fonction locale en langage C, qui affiche des images dont les descripteurs sont extraits d'une table.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sqlenv.h>
#include <sqlcodes.h>
#include <dmbimage.h>  1

int count = 0;


long
main(int argc,char *argv[])
{
EXEC SQL BEGIN DECLARE SECTION;  2
         char hvImg_hdl[251];                  /* descripteur d'image */
         char hvDBName[19];                    /* nom de base de données */
         char hvName[40];                      /* nom de l'employé */
         char hvFormat[9];                     /* format de l'image */
         long hvSize;                          /* taille de l'image */
         long hvWidth;                         /* largeur de l'image */
         long hvHeight;                        /* hauteur de l'image */
struct {
        short len;
                  char data[32700]
         } hvComment;                          /* commentaire sur l'image */
 EXEC SQL END DECLARE SECTION;

/*  Connexion à une base de données  */

strcpy(hvDBName, argv[1]);                     /* copie du nom de la base de données */

EXEC SQL CONNECT TO :hvDBName IN SHARE MODE;
/*
 * Définition du chemin des fonctions en cours
*/
EXEC SQL SET CURRENT FUNCTION PATH = mmdbsys, CURRENT FUNCTION PATH;
```

Figure 24. Application utilisant une extension DB2 (Numéro 1 de 2)

```
/*
 * Sélection (requête) utilisant une fonction UDF d'extension Image
 *
 * L'instruction SQL ci-dessous recherche toutes les images au format GIF.
 */
EXEC SQL DECLARE c1 CURSOR FOR
         SELECT PICTURE, NAME,                3
                SIZE(PICTURE), WIDTH(PICTURE),
                HEIGHT(PICTURE), COMMENT(PICTURE)
       FROM EMPLOYEE
         WHERE PICTURE IS NOT NULL AND
               FORMAT(PICTURE) LIKE 'GIF%'
FOR FETCH ONLY;

EXEC SQL OPEN c1;
for (;;){
         EXEC SQL FETCH c1 INTO :hvImg_hdl, :hvName, :hvSize,
                                :hvWidth, :hvHeight, :hvComment;
         if (SQLCODE != 0)
              break;

         printf("\nRecord %d:\n", ++count);
         printf("nom employé = '%s'\n", hvName);
         printf("taille image = %d octets, width=%d, height=%d\n",
                                              hvSize, hvWidth, hvHeight);
         hvComment.data[Comment.len]='\0';
         printf("comment len = %d\n", hvComment.len);
 printf("comment = %s\n", hvComment.data);
/*
* L'appel d'API ci-après affiche les images
*/
 4  rc=DBiBrowse ("ib %s",MMDB_PLAY_HANDLE,hvImg_hdl,
                          MMDB_PLAY_WAIT);
}

EXEC SQL CLOSE c1;

/* fin du programme */
```

Figure 24. Application utilisant une extension DB2 (Numéro 2 de 2)

Inclusion des définitions d'extension

Vous devez disposer d'un fichier d'inclusion (d'en-tête) dans votre application pour chaque extension que vous utilisez. Chaque fichier d'inclusion définit des constantes, des variables et des prototypes de fonction utilisés pour l'extension. Les noms des fichiers d'inclusion sont les suivants :

Fichier d'inclusion	Extension
dmbimage.h	Image
dmbqbapi.h	Image (recherche par contenu d'image)

dmbaudio.h	Audio
dmbvideo.h	Vidéo
dmbshot.h	Vidéo (détection de changement de plan vidéo)

Vous introduisez un fichier d'inclusion dans un programme écrit en langage C à l'aide d'une directive #include. Par exemple, la directive suivante appelle le fichier d'inclusion pour l'extension Image :

```
#include <dmbimage.h>
```

Indication des noms UDF et UDT

Le nom complet d'une fonction UDF d'une extension DB2 se présente sous la forme mmdbsys.nom-fonction. Le nom complet d'un type UDT d'une extension DB2 se présente sous la forme mmdbsys.nom-type où mmdbsys est le nom de schéma de la fonction ou un type distinct. Par exemple, le nom complet de la fonction UDF Content est mmdbsys.Content ; le nom complet du type de données DB2Image créé par l'extension Image est mmdbsys.DB2Image. Vous n'avez pas à indiquer le nom de schéma mmdbsys si vous avez préalablement attribué la valeur mmdbsys au chemin des fonctions en cours, par exemple :

```
SET CURRENT FUNCTION PATH = mmdbsys, CURRENT FUNCTION PATH
SET CURRENT PATH = mmdbsys, CURRENT PATH
```

Transmission des objets LOB

Plusieurs méthodes permettent de transférer les objets LOB, tels que les images, les séquences audio et vidéo entre votre application et une base de données DB2. Le choix de la méthode de transfert dépend de l'emplacement source et cible de l'objet : vers un fichier ou à partir de celui-ci ; dans une mémoire tampon ou à partir de celle-ci sur un poste client ou serveur de bases de données.

Si l'objet est transféré entre une table et un fichier du serveur

Lorsque vous effectuez un transfert d'objet entre une table de base de données et un fichier du serveur, indiquez le chemin du fichier dans la requête de la fonction UDF de l'extension appropriée. Etant donné que la fonction UDF et le fichier se trouvent sur le serveur, l'extension peut localiser le fichier. Par exemple, dans l'instruction SQL ci-après, une image dont le contenu se trouve dans un fichier du serveur est stockée dans la table de la base de données.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_INTERNAL;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
```

```
      'Anne Dupont',
        DB2Image(
          CURRENT SERVER,
          '/Employés/images/adupont.bmp',
                   'ASIS',
          :hvStorageType,
          'Photo d''Anne')
       );
```

Si l'objet est transféré dans une mémoire tampon client ou à partir de celle-ci

DB2 Extensions ne peut pas accéder directement à une mémoire tampon. Si vous voulez effectuer un transfert d'objet dans une mémoire tampon de votre machine ou à partir de celle-ci, vous pouvez simplement indiquer l'adresse de la mémoire tampon. Vous pouvez faire transiter l'objet via une variable SQL. C'est le mode normal de transfert des objets entre une application et une base de données DB2.

Les variables SQL pour les objets LOB s'utilisent et se définissent de manière analogue à celle des objets alphanumériques et numériques classiques. Vous devez déclarer les variables SQL dans une section DECLARE, leur attribuer des valeurs pour la transmission ou accéder à des valeurs qui leur sont transmises.

Lorsque vous déclarez une variable SQL pour des données de type image, audio ou vidéo, indiquez BLOB comme type de données. Lorsque vous utilisez une fonction UDF pour stocker, extraire ou mettre à jour un objet, vous devez indiquer la variable SQL appropriée comme argument dans la requête UDF. Utilisez le même format que pour les autres variables spécifiées dans l'instruction SQL.

Par exemple, les instructions SQL suivantes déclarent et utilisent une variable SQL appelée `hvaudio` pour transférer une séquence audio à une base de données :

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB (2M) hvaudio;
  EXEC SQL END DECLARE SECTION;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
        DB2Audio(
          CURRENT SERVER,
          :hvaudio,
                 'WAVE',
CAST(NULL as LONG VARCHAR),
          'Voix d''Anne Dupont')
       );
```

Utilisation des releveurs de coordonnées LOB

Les objets LOB, tels que les séquences audio et vidéo, peuvent être de très grande taille, et l'utilisation de variables risque de ne pas constituer le meilleur moyen de les manipuler. L'utilisation du **releveur de coordonnées LOB** peut en revanche s'avérer plus efficace pour la manipulation d'objets LOB dans vos applications.

Le releveur de coordonnées LOB est une valeur faible (4 octets) stockée dans une variable SQL, qui peut être utilisée par votre programme pour identifier un objet LOB de grande taille situé dans la base de données DB2. Lorsque vous utilisez un releveur de coordonnées LOB, votre programme peut manipuler l'objet LOB comme si ce dernier était stocké dans une variable SQL normale. La différence réside dans le fait qu'il n'est plus nécessaire d'effectuer le transfert de l'objet LOB entre le serveur de bases de données et l'application se trouvant sur la machine client. Par exemple, lorsque vous sélectionnez un objet LOB dans une table de base de données, ce dernier reste sur le serveur et le releveur de coordonnées LOB est transféré sur le poste client.

Vous déclarez un objet LOB dans la section DECLARE et vous l'utilisez de la même manière que la variable SQL. Lorsque vous déclarez un releveur de coordonnées LOB pour des données image, audio ou vidéo indiquez BLOB_LOCATOR comme type de données. Par exemple, les instructions SQL suivantes déclarent et utilisent un releveur de coordonnées LOB appelé `video_loc` pour extraire une séquence vidéo provenant d'une table de bases de données :

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB_LOCATOR video_loc;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT CONTENT(VIDEO)
     INTO :video_loc
       FROM EMPLOYEE
       WHERE NAME='Anne Dupont';
```

Utilisation de releveurs de coordonnées LOB par les fonctions UDF : Les fonctions UDF d'extensions DB2 qui stockent, extraient et mettent à jour des objets image, audio et vidéo utilisent des releveurs de coordonnées LOB. Dans la version 1 de DB2 Extensions, ces fonctions UDF ne faisaient pas appel à des releveurs de coordonnées LOB. Elles ne pouvaient donc pas traiter des objets dont la taille était supérieure à 2 Mo. Cette restriction obligeait les utilisateurs à transférer de tels objets par segments. Or ce n'est plus le cas grâce aux releveurs de coordonnées LOB.

Si l'objet est transféré dans un fichier client ou à partir de celui-ci

Utilisez une variable de référence à un fichier pour transférer les objets dans un fichier sur un client ou à partir de ce fichier.

La variable de référence à un fichier est déclarée dans une section DECLARE et est utilisée de manière analogue à une variable SQL. Lorsque vous déclarez une variable de référence à un fichier pour des données de type image, audio ou vidéo, indiquez BLOB_FILE comme type de données. Contrairement à la variable SQL qui contient l'ensemble du contenu d'un objet, la variable de référence à un fichier ne contient que le nom du fichier. La taille du fichier ne peut pas excéder celle de l'objet BLOB défini pour la fonction UDF.

Plusieurs options s'offrent à vous pour l'utilisation d'une variable de référence à un fichier d'entrée et de sortie. Sélectionnez l'option de votre choix en définissant la zone FILE_OPTIONS dans la structure de la variable de référence à un fichier de votre programme. Choisissez l'une des options suivantes :

Options en entrée :

SQL_FILE_READ. Vous pouvez ouvrir ce fichier, y accéder en lecture et le refermer. La longueur des données du fichier (en octets) est déterminée à l'ouverture du fichier. La zone longueur_données de la structure de la variable de référence à un fichier contient la longueur du fichier (en octets).

Options en sortie :

SQL_FILE_CREATE. Cette option permet de créer un fichier s'il n'existe pas déjà. S'il existe, vous recevez un message d'erreur. La zone longueur_données de la structure de la variable de référence à un fichier contient la longueur du fichier (en octets).

SQL_FILE_OVERWRITE. Cette option permet de créer un fichier s'il n'existe pas déjà. Si le fichier existe, les données contenues dans le fichier sont remplacées par les nouvelles données. La zone longueur_données de la structure de la variable de référence à un fichier contient la longueur du fichier (en octets).

SQL_FILE_APPEND. Cette option permet d'ajouter les données de sortie au fichier si ce dernier existe déjà. Si ce n'est pas le cas, elle crée un nouveau fichier. La zone longueur_données de la structure de la variable de référence au fichier contient la longueur des données ajoutées au fichier (en octets), mais pas la longueur totale du fichier.

Par exemple, les instructions suivantes déclarent une variable de référence à un fichier appelée `Fichier_Img` qui servira à stocker une image dont le contenu se trouve dans un fichier client, dans la table de base de données. Notez l'affectation de SQL_FILE_READ dans la zone FILE_OPTIONS :

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB_FILE Img_file;
  EXEC SQL END DECLARE SECTION;

strcpy (Fichier_Img.name,"/Employés/images/adupont.bmp");
```

```
Img_file.name_length=strlen(Img_file.name);
Fichier_Img.file_options=SQL_FILE_READ;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
        DB2Image(
          CURRENT SERVER,
          :Fichier_Img,
                   'ASIS',
CAST(NULL as LONG VARCHAR),
          'Photo d''Anne')
       );
```

Noms des fichiers lors du transfert des objets

DB2 Extensions autorise une certaine souplesse quant à l'attribution des noms de fichier lors du stockage, de l'extraction et de la mise à jour d'objets.

Bien que vous puissiez indiquer un nom qualifié complet (c'est-à-dire le chemin d'accès complet suivi du nom du fichier), il est préférable d'indiquer un nom de fichier relatif lors des opérations de stockage, d'extraction et de mise à jour. Sous AIX, HP-UX et Solaris, un nom de fichier relatif est un nom de fichier ne commençant pas par une barre oblique. Sous OS/2 et Windows, il s'agit d'un nom de fichier ne commençant pas par une lettre d'unité suivie de deux points et d'une barre oblique inversée.

Si vous indiquez un nom de fichier relatif, les extensions utilisent les spécifications de répertoire dans plusieurs variables d'environnement client et serveur pour identifier le nom de fichier. Un nom de chemin absolu est constitué d'une première partie, qui est généralement associée aux points de montage, suivie d'un chemin d'accès définissant de manière unique le fichier requis. Le chemin d'accès indiqué à la fin du nom est spécifié dans des fonctions UDF. Les variables d'environnement contiennent une liste des chemins d'accès placés en tête du nom de chemin absolu et permettant d'identifier les noms de fichiers relatifs. Pour plus d'informations sur les variables d'environnement utilisées par DB2 Extensions pour l'identification des noms de fichier, reportez-vous à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

Les Extensions permettent également de convertir les noms de fichier dans le format approprié. Lorsqu'un nom de fichier est transféré sur le serveur, il est converti au format correspondant au système d'exploitation du serveur. Par exemple, le fichier OS/2 c:\dir1\abc.bmp devient /dir1/abc.bmp lorsqu'il est transféré sur un serveur AIX.

Gestion des codes retour

Toutes les instructions SQL imbriquées ou appels DB2 CLI de votre programme, y compris celles qui demandent les fonctions UDF de DB2

Extensions, génèrent des codes indiquant si l'instruction SQL imbriquée ou l'appel DB2 CLI a abouti ou non. D'autres API de DB2 Extensions, telles que les API d'administration, peuvent également renvoyer des codes indiquant si l'opération a abouti ou non. Votre programme doit vérifier les codes renvoyés par les instructions SQL imbriquées, les appels CLI et les API et y répondre.

Pour plus d'informations sur la gestion de ces codes, reportez-vous au «Chapitre 19. Informations de diagnostic» à la page 543.

Dans le cas où une API d'extension ne peut pas mener à bien son unité d'oeuvre, une opération d'annulation est réalisée. L'API renvoie également un code d'erreur. L'annulation est effectuée dans le but de restaurer le point de cohérence précédent de la base de données. Pour plus de détails, reportez-vous au «Chapitre 16. Interfaces de programmation d'applications» à la page 279.

Support Unicode

Les restrictions ci-dessous doivent être prises en considération pour l'utilisation du support Unicode avec les extensions Image, Audio et Vidéo :

- Les seuls paramètres pouvant être constitués d'une chaîne Unicode sont les zones de commentaire des fonctions UDF suivantes :
  - mmdbsys.db2image( ) import an image
  - mmdbsys.db2audio( ) import an audio
  - mmdbsys.db2video( ) import a video
  - mmdbsys.replace( ) replace an image, an audio, or a video
  - mmdbsys.comment( ) comment update
- Si vous envisagez d'accéder à une base de données Unicode, vous devez utiliser une instance DB2 Extensions prenant en charge le format Unicode. Une instance Unicode ne peut traiter qu'une base de données de type Unicode.

  Pour qu'une instance d'extension prenne en charge le format Unicode, vous devez affecter à la variable d'environnement DB2CODEPAGE la valeur 1208 avant d'exécuter la commande DMBSTART.

Chapitre 11. Stockage, extraction et mise à jour d'objets

Le présent chapitre indique comment utiliser les fonctions définies par les extensions DB2 pour stocker, extraire et mettre à jour un objet image, audio ou vidéo.

Formats image, audio et vidéo

Le tableau 5 répertorie les formats dans lesquels vous pouvez stocker, extraire ou mettre à jour des objets image, audio et vidéo. Pour les objets image uniquement, l'extension Image peut convertir le format de l'image au moment du stockage, de l'extraction ou de la mise à jour. (Les formats audio et vidéo ne peuvent pas être convertis lors de ces opérations.)

Les colonnes Lecture et Ecriture du tableau indiquent les formats qui peuvent être lus et ceux qui peuvent être convertis au moment de l'écriture. Une croix (x) dans la colonne Lecture signifie que le format correspondant peut être utilisé lors des opérations de stockage, d'extraction ou de mise à jour. Une croix (x) dans la colonne Ecriture indique que l'objet (image uniquement) peut être converti au format correspondant pendant ces mêmes opérations. Exemple : une image BMP peut être convertie au format GIF lorsqu'elle est stockée, extraite ou mise à jour. Une image JPG peut être convertie au format TIF, mais pas le contraire.

Bien que les formats apparaissent en majuscules dans le tableau, le système ne distingue pas les majuscules et les minuscules dans les requêtes de stockage, d'extraction ou de mise à jour. Exemple : GIF, gif et Gif sont équivalents.

Tableau 5. Formats pris en charge par DB2 Extensions

Format	Description	Lecture	Ecriture
	Formats image		
_IM	AVC (Audio Video Connection) PS/2	x	
BMP	Bitmap OS/2 - Microsoft** Windows	x	x
EPS	PostScript** encapsulé		x
EP2	PostScript niveau 2 encapsulé		x
GIF	Compuserve GIF89a et 87	x	x
IMG	Image IOCA	x	x
IPS	Fichiers de cartes Brooktrout FAX	x	x
JPG	JPEG[3](format JFIF)	x	

Formats

Tableau 5. Formats pris en charge par DB2 Extensions (suite)

Format	Description	Lecture	Ecriture
PCX	Fichier Paint PC (niveaux de gris uniquement)	x	x
PGM	Portable Gray Map (de PBMPLUS)	x	x
PS	PostScript		x
PSC	Image en PostScript compressé		x
PS2	PostScript niveau 2 (couleur)		x
TIF	Tous formats TIFF 5.0	x	x
YUV	Format vidéo numérique pour YUV	x	x
	Formats audio		
AIF ou AIFF	Audio Interchange File Format	x	
AIFFC	Audio Interchange File Format Compressed	x	
AU	Sun audio file format	x	
MIDI	Musical Instrument Digital Interface	x	
MPG1 ou MPEG1	Moving Pictures Expert Group 1	x	
WAV ou WAVE	Wave	x	
	Formats vidéo		
AVI	Audio/Video Interleaved	x	
MPG1 ou MPEG1	Motion Picture Coding Expert Group 1	x	
MPG2 ou MPEG2	Motion Picture Coding Expert Group 2	x	
QT	Quicktime (AVI)	x	

Options de conversion d'image

Le tableau 6 à la page 93 répertorie les options de conversion (en plus des conversions de format) pouvant être spécifiées pour une image lorsqu'elle est stockée, extraite ou mise à jour. L'extension image applique vos spécifications à l'image cible ; l'image source n'est pas modifiée.

Chaque option de conversion est définie sous forme d'un couple paramètre/valeur. Les valeurs autorisées pour chaque paramètre sont répertoriées dans le tableau.

1. La lecture est prise en charge par OS/2 version 1, OS/2 version 2, Windows version 2, Windows version 3 et Windows NT au format BMP. L'écriture est prise en charge par OS/2 version 1 au format BMP.
2. L'extension DB2 Image ne stocke les informations d'attributs que pour la première image dans le fichier animé GIF.
3. Le logiciel utilisé reprend en partie les travaux réalisés par Independent JPEG Group.

Tableau 6. Options de conversion d'image

Paramètre	Description	Valeur
-b	Nombre de bits utilisés pour représenter chaque modèle d'image	1 ou 8 bits
-s[4]	Facteur d'échelle	Toute valeur décimale supérieure à zéro. Le facteur d'échelle spécifie le ratio taille de l'image convertie par rapport à l'original. Par exemple, un facteur d'échelle de 0,5 ramène l'image à la moitié de sa taille originale. Un facteur d'échelle de 2 multiplie la taille de l'image par deux.
-p	Photométrique (inversion d'image). Cette option permet de modifier l'interprétation d'une image, en fonction de la valeur indiquée. Mais elle ne permet pas de modifier l'image elle-même. Elle ne concerne que les images en noir et blanc ou en échelle de gris, et non les images au format GIF.	0 = Les bits à 1 sont noirs 1 = Les bits à 1 sont blancs
-n	Photométrique (inversion d'image). Cette option permet de modifier l'image par inversion de la polarité (inversion du blanc en noir, et vice versa). Elle ne concerne que les images en noir et blanc ou en échelle de gris.	Aucune
-r[4]	Rotation	0 = 0 degrés (aucune rotation) 1 = 90 degrés (en sens inverse des aiguilles d'une montre) 2 = 90 degrés (dans le sens des aiguilles d'une montre) 3 = 180 degrés
-x[4]	Largeur en pixels	Nombre de pixels
-y[4]	Hauteur en pixels	Nombre de pixels

Tableau 6. Options de conversion d'image (suite)

Paramètre	Description	Valeur
-c	Type de compression	0 = IBM MMR 1 = CCITT Group 3 1-D 2 = CCITT Group 3 2-D (k=2) 3 = CCITT Group 3 2-D (k=4) 4 = CCITT Group 4 6 = TIFF Type 2 10 = Non compressé 14 = LZW 15 = TIFF Packbits 25 = JBIG

Stockage d'objets image, audio ou vidéo

Pour stocker un objet image, audio ou vidéo dans une base de données, utilisez la fonction UDF DB2Image, DB2Audio ou DB2Video dans une instruction SQL INSERT.

Vous pouvez stocker un objet dont la source se trouve dans la mémoire tampon ou dans un fichier du poste client ou dans un fichier du serveur. Quelle que soit la source, vous pouvez stocker l'objet dans une table de la base de données en tant qu'objet BLOB ou dans un fichier du serveur de bases de données.

Lorsque vous appelez la fonction UDF, vous devez spécifier :

- le nom du serveur de bases de données auquel vous êtes actuellement connecté (il se trouve dans le registre spécial CURRENT SERVER) ;
- la source de l'objet ; l'objet peut se trouver dans la mémoire tampon ou dans un fichier du client, ou encore dans un fichier du serveur ;
- si vous voulez stocker l'objet dans une table de base de données en tant qu'objet BLOB ou sur un serveur de fichiers ;
- le format de la source ;
- un commentaire facultatif à stocker avec l'objet (ou la valeur NULL ou une chaîne vide si vous ne voulez pas entrer de commentaire).

Les extensions Image, Audio et Vidéo permettent de stocker un objet dont elles ne reconnaissent pas obligatoirement le format. Dans ce cas, vous devez préciser les attributs de l'objet. Lorsque vous stockez un objet image ou vidéo

4. Si vous spécifiez cette option pour une image au format GIF, vous devez également indiquer un type de compression LZW.

avec des attributs définis par l'utilisateur, vous pouvez également en enregistrer une version miniature (image réduite représentant l'objet image ou vidéo).

Pour les images uniquement, vous avez la possibilité de convertir le format au moment du stockage. En cas de conversion de format, il convient d'indiquer le format source et le format cible de l'image. Lors d'une demande de conversion de format, vous pouvez également spécifier d'autres modifications concernant l'image, telles que le détourage ou la rotation. Vous pouvez indiquer ces modifications en spécifiant les options de conversion.

Validation de l'opération de stockage : Validez l'unité d'oeuvre après avoir stocké l'objet image, audio ou vidéo dans une base de données. Cela permet de déverrouiller l'objet (celui-ci ayant été verrouillé par les extensions pour sa mise à jour).

Formats des fonctions UDF DB2Image, DB2Audio et DB2Video

En fonction de son utilisation, chaque fonction UDF DB2Image, DB2Audio ou DB2Video peut avoir l'un des formats suivants (xxxxx correspond à Image, Audio ou Vidéo) :

Format 1 : Stockage d'un objet à partir de la mémoire tampon ou d'un fichier client :

```
DB2xxxxx(
     CURRENT SERVER,          /* nom de la base de données dans CURRENT SERVER REGISTER */
     content,                 /* contenu de l'objet */
     format,                  /* format source */
     target_file,             /* nom du fichier cible pour stockage sur le */
                              /* serveur de fichiers */
                              /* ou NULL pour stockage dans une table */
                              /* en tant qu'objet BLOB */
     comment                  /* commentaire */
  );
```

Format 2 : Stockage d'un objet à partir d'un fichier du serveur :

```
DB2xxxxx(
     CURRENT SERVER,           /* nom de la base de données dans CURRENT SERVER REGISTER */
     source_file,              /* nom du fichier source */
     format,                   /* format source */
     stortype,                 /* MMDB_STORAGE_TYPE_EXTERNAL=stockage */
                               /* sur un serveur de fichiers*/
                               /* MMDB_STORAGE_TYPE_INTERNAL=stockage */
                               /* en tant qu'objet BLOB */
     comment                   /* commentaire */
  );
```

Format 3 : Stockage d'un objet dont les attributs sont définis par l'utilisateur, à partir de la mémoire tampon ou d'un fichier du poste client :

```
DB2xxxxx(

    CURRENT SERVER,          /* nom de la base de données dans CURRENT SERVER */
                             /* REGISTER */
    content,                 /* contenu de l'objet */
    target_file,             /* nom du fichier cible pour stockage sur le */
                             /* serveur de fichiers */
                             /* ou NULL pour stockage dans une table */
                             /* en tant qu'objet BLOB */
    comment,                 /* commentaire */
    attrs,                   /* attributs définis par l'utilisateur */
    thumbnail                /* miniature (objet image et vidéo uniquement) */
 );
```

Format 4 : Stockage d'un objet dont les attributs sont définis par l'utilisateur, à partir d'un fichier du serveur :

```
DB2xxxxx(

    CURRENT SERVER,          /* nom de la base de données dans CURRENT SERVER */
                             /* REGISTER */
    source_file,             /* nom du fichier source */
    stortype,                /* MMDB_STORAGE_TYPE_EXTERNAL=stockage */
                             /* sur un serveur de fichiers*/
                             /* MMDB_STORAGE_TYPE_INTERNAL=stockage */
                             /* en tant qu'objet BLOB */
    comment,                 /* commentaire */
    attrs,                   /* attributs définis par l'utilisateur */
    thumbnail                /* miniature (objet image et vidéo uniquement) */
 );
```

La fonction UDF DB2Image comprend les formats supplémentaires suivants :

Format 5 : Stockage d'une image à partir de la mémoire tampon ou d'un fichier client avec conversion de format :

```
      DB2Image(
    CURRENT SERVER,          /* nom de la base de données dans CURRENT SERVER REGISTER */
    content,                 /* contenu de l'objet */
    source_format,           /* format source */
    target_format,           /* format cible */
    target_file,             /* nom du fichier cible pour stockage sur le */
                             /* serveur de fichiers */
                             /* ou NULL pour stockage dans une table */
                             /* en tant qu'objet BLOB */
    comment                  /* commentaire */
 );
```

Format 6 : Stockage d'une image à partir d'un fichier du serveur avec conversion de format :

```
      DB2Image(
    CURRENT SERVER,          /* nom de la base de données dans CURRENT SERVER REGISTER */
    source_file,             /* nom du fichier du serveur */
    source_format,           /* format source */
    target_format,           /* format cible */
    target_file,             /* nom du fichier cible pour stockage sur le */
                             /* serveur de fichiers */
                             /* ou NULL pour stockage dans une table */
                             /* en tant qu'objet BLOB */
    comment                  /* commentaire */
 );
```

Format 7 : Stockage d'une image à partir de la mémoire tampon ou d'un fichier client avec conversion de format et modifications supplémentaires :

```
       DB2Image(
    CURRENT SERVER,          /* nom de la base de données dans CURRENT SERVER REGISTER */
    content,                 /* contenu de l'objet */
    source_format,           /* format source */
    target_format,           /* format cible */
    conversion_options,      /* options de conversion  */
    target_file,             /* nom du fichier cible pour stockage sur le */
                             /* serveur de fichiers */
                             /* ou NULL pour stockage dans une table */
                             /* en tant qu'objet BLOB */
    comment                  /* commentaire */
  );
```

Format 8 : Stockage d'une image à partir d'un fichier du serveur avec conversion de format et modifications supplémentaires :

```
       DB2Image(
    CURRENT SERVER,          /* nom de la base de données dans CURRENT SERVER REGISTER */
    source_file,             /* nom du fichier du serveur */
    source_format,           /* format source */
    target_format,           /* format cible */
    conversion_options       /* options de conversion */
    target_file,             /* nom du fichier cible pour stockage sur le */
                             /* serveur de fichiers */
                             /* ou NULL pour stockage dans une table */
                             /* en tant qu'objet BLOB */
    comment                  /* commentaire */
  );
```

Par exemple, les instructions ci-après, incluses dans un programme d'application en langage C, insèrent une ligne contenant une image dans la table Employés. L'image source est stockée sur le serveur, dans un fichier appelé adupont.bmp. La photo est stockée sous la forme d'un objet BLOB dans la table *Employés*. (Cette opération correspond au format 2.)

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_INTERNAL;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',                          /*id*/
        'Anne Dupont',                     /*nom*/
        DB2IMAGE(                           /*UDF de l'extension Image*/
CURRENT SERVER,                  /*base de données*/
          '/Employés/images/adupont.bmp',  /*fichier source */
                'ASIS',                           /*conservation du format de l'image*/
          :hvStorageType,                  /*stockage image dans BD en tant que BLOB*/
          'Photo d''Anne')                 /*commentaire */
      );
```

Les instructions ci-après, incluses dans un programme d'application en langage C, permettent de stocker la même ligne que dans l'exemple précédent dans la table Employés, à cette différence près que l'image est convertie du format BMP au format GIF lors du stockage. (Cela correspond au format 6 ci-dessus.)

```
EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',                          /*id*/
        'Anne Dupont',                     /*nom*/
        DB2IMAGE(                           /*fonction UDF extension Image*/
CURRENT SERVER,                  /*base de données*/
          '/Employés/images/adupont.bmp',  /*fichier source */
```

```
        'ASIS',                          /*format image source*/
        'GIF',                           /*format image cible*/
        'Photo d''Anne')                /*commentaire*/
    );
```

Lorsque vous stockez un objet image, audio ou vidéo, l'extension calcule les attributs tels que le nombre de couleurs de l'image, la durée de lecture ou le format de compression vidéo. Si vous stockez un objet dans un format non reconnu, il vous revient de spécifier ces attributs pour la fonction UDF. L'extension les enregistre dans la base de données avec les autres attributs, tels que les commentaires sur l'objet et l'ID de l'utilisateur qui a stocké l'objet. Ces attributs peuvent ensuite être utilisés dans les requêtes.

Stockage d'un objet résidant sur le poste client

Utilisez une variable SQL ou une variable de référence à un fichier pour transférer le contenu d'un objet image, audio ou vidéo d'une mémoire tampon client ou d'un fichier client vers le serveur.

Si l'objet se trouve dans un fichier du poste client, utilisez une variable de référence à un fichier pour transférer le contenu en vue de son stockage sur le serveur. Par exemple, les instructions ci-après, présentes dans un programme d'application en langage C, définissent une variable de référence à un fichier appelée `fichier_audio` qui servira à transférer un enregistrement audio se trouvant dans un fichier client. Le clip audio est stocké dans une table de base de données du serveur. A noter que la zone `file_option` de la variable de référence à un fichier a pour valeur `SQL_FILE_READ` et que la variable de référence à un fichier sert d'argument à la fonction UDF DB2Audio.

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB_FILE fichier_audio;
  EXEC SQL END DECLARE SECTION;

strcpy (fichier_audio.name, "/Employés/sons/adupont.wav");
fichier_audio.name_length= strlen(fichier_audio.name);
fichier_audio.file_options= SQL_FILE_READ;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
               DB2AUDIO(
          CURRENT SERVER,
          :fichier_audio,
/* variable de référence à un fichier */
                'WAVE',
                CAST(NULL as LONG VARCHAR),

          'Voix d''Anne Dupont')
      );
```

Si l'objet se trouve dans la mémoire tampon du poste client, utilisez une variable SQL (définie en tant que BLOB ou BLOB_LOCATOR) pour en transférer le contenu en vue de son stockage sur le serveur. Les instructions

ci-après, incluses dans un programme d'application en langage C, font appel à une variable SQL appelée séq_vidéo pour transférer le contenu d'une séquence vidéo en vue de son stockage sur le serveur. La séquence vidéo est stockée dans une table de base de données en tant qu'objet BLOB. A noter que la variable SQL sert d'argument à la fonction UDF DB2Video.

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB_LOCATOR séq_vidéo;
  EXEC SQL END DECLARE SECTION;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2VIDEO(
          CURRENT SERVER,
          :séq_vidéo,         /* variable SQL */
                'MPEG1',
          '',
          'Séquence vidéo d''Anne')
       );
```

Vérification de la mémoire UDF disponible : Lorsque vous modifiez un objet dont les données de mise à jour se trouvent dans la mémoire tampon du client, assurez-vous que le paramètre UDF_MEM_SZ du fichier de configuration du gestionnaire de bases de données est au moins de 4 Mo. Vous pouvez modifier ce paramètre par la commande DB2 UPDATE DATABASE MANAGER CONFIGURATION. Pour plus d'informations sur la commande UPDATE DATABASE MANAGER, reportez-vous au manuel *DB2 Command Reference*.

Stockage d'un objet résidant sur le serveur

Lorsque l'objet image, audio ou vidéo à stocker se trouve sur un fichier du serveur, indiquez son chemin comme argument de la fonction UDF. Par exemple, l'instruction ci-après, incluse dans un programme d'application en langage C, stocke une ligne contenant une image dans la base de données. La source de l'image se trouve dans un fichier du serveur. L'image stockée est conservée dans le fichier du serveur et la base de données inclut un pointeur sur ce dernier.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_EXTERNAL;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2IMAGE(
          CURRENT SERVER,
          '/Employés/images/adupont.bmp', /*source dans le fichier du serveur */
```

```
        'BMP',
        :hvStorageType,
        'Photo d''Anne')
    );
```

Spécification du chemin correct : Lorsque vous stockez un objet dont la source se trouve dans un fichier du serveur, vous pouvez spécifier le nom complet du fichier ou son nom relatif. Si vous indiquez un nom de fichier relatif, assurez-vous que les variables d'environnement appropriées du serveur DB2 contiennent le chemin correct du fichier. Pour plus de détails sur la définition des variables d'environnement, reportez-vous à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

Stockage dans une base de données ou dans un fichier

Vous pouvez stocker un objet image, audio ou vidéo dans une table de base de données en tant qu'objet BLOB, ou dans un fichier du serveur. Dans le deuxième cas, la base de données contient un pointeur sur le fichier.

Lorsque vous stockez un objet provenant de la mémoire tampon ou d'un fichier client, vous indiquez le type de stockage (objet BLOB ou fichier du serveur) dans le paramètre fichier cible (target_file). Si vous spécifiez un nom de fichier, l'objet sera stocké dans un fichier du serveur. Si vous entrez la valeur NULL ou une chaîne vide, le fichier est stocké en tant qu'objet BLOB dans une table de base de données. Le type de données du paramètre fichier cible est LONG VARCHAR. Si vous entrez la valeur NULL, n'oubliez pas de convertir son type de données en LONG VARCHAR (par CAST).

Par exemple, l'instruction ci-après, incluse dans un programme d'application en langage C, stocke une ligne contenant une image dans une table de base de données. L'image source se trouve dans une mémoire tampon sur le poste client. L'image est stockée dans un fichier du serveur et la table de base de données contient un pointeur sur ce fichier :

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB_LOCATOR Img_buf
  EXEC SQL END DECLARE SECTION;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2IMAGE(
          CURRENT SERVER,
                   :Img_buf,
                   'ASIS',
          '/Employés/images/adupont.bmp',   /* stockage de l'image dans un */
                                            /* fichier du serveur */
          'Photo d''Anne')
       );
```

Si vous stockez un objet à partir d'un fichier du serveur, spécifiez la constante MMDB_STORAGE_TYPE_INTERNAL pour enregistrer l'objet dans une table de base de données en tant qu'objet BLOB. Si vous voulez stocker l'objet en le conservant dans le fichier du serveur, indiquez la constante MMDB_STORAGE_TYPE_EXTERNAL. La valeur de MMDB_STORAGE_TYPE_INTERNAL est 1. La valeur de MMDB_STORAGE_TYPE_EXTERNAL est 0.

Par exemple, le programme d'application en C ci-dessous stocke une séquence audio dans un fichier du serveur. Le son source se trouve déjà dans un fichier du serveur. L'opération de stockage place le fichier dans la base de données et le rend donc accessible à l'aide d'instructions SQL.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_EXTERNAL;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
               DB2AUDIO(
          CURRENT SERVER,
          '/Employés/sons/adupont.wav',
                 'WAVE',
          :hvStorageType,             /* stockage de la séquence audio dans */
                                      /* un fichier du serveur */
          'Voix d''Anne Dupont')
       );
```

Indication du format de stockage

Lorsque vous stockez un objet, il convient d'indiquer son format. Les formats que vous pouvez spécifier sont répertoriés dans le tableau 5 à la page 91. Les extensions stockent normalement les objets image, audio ou vidéo dans le même format que la source. Pour les objets image uniquement, l'extension Image peut convertir le format de l'image stockée. Si vous décidez de convertir le format de l'image, vous devez indiquer le format de l'image source et celui de l'image cible (c'est-à-dire l'image stockée).

Indication du format de stockage sans conversion

Lorsque vous stockez l'objet sans conversion, indiquez le format de l'objet image, audio ou vidéo source. Par exemple, l'instruction ci-après, incluse dans un programme d'application en langage C, stocke une image bitmap (BMP) dans une table de base de données. La source se trouve dans un fichier du serveur. L'image cible aura le même format que la source.

```
EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2IMAGE(
```

```
        CURRENT SERVER,
        '/Employés/images/adupont.bmp',
        'BMP',                          /* image au format BMP */
        '',
        'Photo d''Anne')
    );
```

Vous pouvez également indiquer comme format la valeur NULL, une chaîne vide ou, dans le cas de l'extension Image uniquement, la chaîne de caractères ASIS. L'extension détermine alors le format en examinant les données source.

Utilisation de NULL ou ASIS pour les formats reconnus : Spécifiez la valeur NULL, une chaîne vide ou ASIS uniquement si le format est reconnaissable par l'extension, c'est-à-dire s'il s'agit des formats répertoriés pour cette extension dans le tableau 5 à la page 91. Sinon, l'extension ne peut pas stocker l'objet.

Indication des formats et des options de conversion pour un stockage avec conversion de format

Lors d'un stockage avec conversion de format, indiquez le format des images source et cible. Le tableau 5 à la page 91 répertorie les conversions de format admises.

En outre, vous pouvez spécifier des options de conversion identifiant des modifications supplémentaires, telles que la rotation ou la compression, que vous souhaitez appliquer à l'image stockée. Vous spécifiez chaque option de conversion grâce à un paramètre et à une valeur associée. Les paramètres et les valeurs autorisées sont répertoriés dans le tableau 6 à la page 93. Vous pouvez effectuer plusieurs changements sur une image stockée en spécifiant plusieurs couples paramètres/valeurs.

Dans l'exemple ci-dessous, une image bitmap (BMP) dont la source se trouve dans un fichier du serveur est convertie au format GIF lors de son stockage dans une table de base de données.

```
EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2IMAGE(
          CURRENT SERVER,
          '/Employés/images/adupont.bmp',
          'BMP',                              /* format source */
          'GIF',                              /* format cible */
          '',
          'Photo d''Anne')
       );
```

Dans l'exemple ci-dessous, l'image de l'exemple précédent est convertie au format GIF lors de son stockage dans une table de base de données. En outre,

l'image est réduite à une largeur de 110 pixels et une hauteur de 150 pixels lors du stockage, puis elle est compressée par une compression de type LZW.

```
EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2IMAGE(
          CURRENT SERVER,
          '/Employés/images/adupont.bmp',
          'BMP',                              /* format source */
          'GIF',                              /* format cible */
          '-x 110 -y 150 -c 14',              /* options de conversion */
          '/Employés/images/adupont.gif',
          'Photo d''Anne')
       );
```

Stockage d'un objet dont les attributs sont définis par l'utilisateur

Lorsque vous stockez un objet image, audio ou vidéo, vous n'êtes pas limité aux formats reconnus par les extensions. Vous pouvez spécifier votre propre format. Dans la mesure où les extensions ne reconnaissent pas le format, vous devez définir les attributs de l'objet source. Définissez les valeurs des attributs sous forme de structure. La structure d'attributs doit être contenue dans la zone de données de la variable LONG VARCHAR FOR BIT DATA de la fonction UDF.

Le code UDF qui s'exécute sur le serveur attend toujours des données au format «big endian». Ce dernier est utilisé pour la plupart des plateformes UNIX. Dans le cas d'un objet au format «little endian», vous devez préparer les données dont les attributs sont définis par l'utilisateur de sorte que le code UDF sur le serveur puisse les traiter correctement. Le format little endian est, en règle générale, utilisé sur les plateformes Intel et d'autres types de microprocesseurs. (Même si vous n'avez pas recours au format little endian, il est recommandé de préparer les données dont les attributs sont définis par l'utilisateur.) Utilisez l'interface API DBiPrepareAttrs pour préparer les attributs des objets image, l'interface API DBaPrepareAttrs pour les objets audio, et l'interface API DBvPrepareAttrs pour les objets vidéo.

Par exemple, les instructions ci-après, incluses dans un programme d'application en langage C, stockent une ligne contenant une image dans une table de base de données. L'image source qui se trouve dans un fichier du serveur, a un format défini par l'utilisateur, une hauteur de 640 pixels et une largeur de 480 pixels. Notez que les attributs sont préparés avant le stockage de l'image.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
struct {
        short len;
       char data[400];
       }hvImgattrs;
  EXEC SQL END DECLARE SECTION;
```

```
DB2IMAGEATTRS    *pimgattr;

hvStorageType=MMDB_STORAGE_TYPE_INTERNAL;

pimgattr = (DB2IMAGEATTRS *) hvImgattrs.data;
strcpy(pimgattr→format,"FormatI");
pimgattr→width=640;
pimgattr→height=480;
hvImgattrs.len=sizeof(DB2IMAGEATTRS);

DBiPrepareAttrs(pimgattr);

DBEXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2IMAGE(
          CURRENT SERVER,
          '/Employés/images/adupont.bmp',
          :hvStorageType,
          'Photo d''Anne',
          :hvImgattrs,          /* attributs définis par l'utilisateur */
          CAST(NULL as LONG VARCHAR)
       );
```

L'instruction ci-après, incluse dans un programme d'application en langage C, stocke une ligne contenant une séquence audio dans une table de base de données. La séquence audio source, qui se trouve dans un fichier du serveur, a un format défini par l'utilisateur, une fréquence d'échantillonnage de 44,1 kHz, et comporte deux canaux enregistrés. Le clip audio n'étant pas de type MIDI, des chaînes vides sont indiquées pour les noms de pistes et les instruments.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
struct (
        short len;
       char data[600];
}hvAudattr;
  EXEC SQL END DECLARE SECTION;

MMDBAudioAttrs          *paudiattr;

hvStorageType=MMDB_STORAGE_TYPE_INTERNAL;

paudioattr=(MMDBAudioAttrs *) hvAudattr.data;
strcpy(paudioAttr→cFormat,"FormatA");
paudioAttr→ulSamplingRate=44100;
paudioAttr→usNumChannels=2;
hvAudattrs.len=sizeof(MMDBAudioAttrs);

DBaPrepareAttrs(paudioAttr);

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
               DB2AUDIO(
```

```
        CURRENT SERVER,
        '/Employés/sons/adupont.aud',
        :hvStorageType,
        'Voix d''Anne',
        :hvAudattr)        /* attributs définis par l'utilisateur */
    );
```

Stockage d'une miniature (objets image et vidéo uniquement)

Lorsque vous stockez un objet image ou vidéo dans un format personnalisé, vous avez la possibilité d'enregistrer une **miniature** de l'image. C'est vous qui définissez la taille et le format de la miniature. Lorsque vous stockez une image dans un format reconnu par l'extension image, celle-ci génère et stocke automatiquement une miniature pour l'objet (au format GIF et d'une taille de 112 x 84 pixels).

Lorsque vous stockez un objet vidéo dans un format personnalisé, vous pouvez également enregistrer une miniature symbolisant l'objet. Si le format de l'objet est reconnu par l'extension vidéo, celle-ci génère et stocke automatiquement une miniature pour l'objet (au format GIF et d'une taille de 108 x 78 pixels).

Si vous ne voulez pas stocker de miniature avec l'objet image ou vidéo dont les attributs sont définis par l'utilisateur, remplacez la référence à la miniature par la valeur NULL ou une chaîne vide.

Générez la miniature dans votre programme car les extensions ne fournissent pas d'API à cette fin. Créez également la structure de la miniature dans votre programme et spécifiez-la dans la fonction UDF.

Les instructions ci-après, incluses dans un programme d'application en langage C, stockent une ligne contenant une séquence vidéo dans une table de base de données. La séquence vidéo source, qui se trouve dans un fichier sur le serveur, est dans un format défini par l'utilisateur. Son contenu reste sur le serveur et la table comporte un pointeur sur ce fichier. Une miniature d'une image vidéo représentative est également enregistrée.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
         struct  {
        short len;
        char data [4000];
       }hvVidattrs;
struct {
        short len;
       char data[10000];
       }hvThumbnail;
  EXEC SQL END DECLARE SECTION;

MMDBVideoAttrs         *pvideoAttr;

hvStorageType=MMDB_STORAGE_TYPE_EXTERNAL;
```

```
pvideoAttr=(MMDBVideoAttrs *)hvVidattrs.data;
strcpy(pvideoAttr→cFormat,"Formatv");
hvVidattrs.len=sizeof(MMDBVideoAttrs);

/* Création d'une miniature et affectation des données à la structure de la vidéo */

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2VIDEO(
          CURRENT SERVER,
          '/Employés/vidéo/adupont.vid',
          :hvStorageType,
          'Séquence vidéo d''Anne',
          :hvVidattrs,
          :hvThumbnail)                    /* Miniature*/
       );
```

Stockage d'un commentaire

Pour stocker un commentaire avec un objet image, audio ou vidéo, entrez-le dans la requête UDF. Un commentaire est un texte quelconque de type LONG VARCHAR d'une longueur maximale de 32700 octets. Si vous ne voulez pas enregistrer de commentaire avec l'objet, entrez la valeur NULL ou une chaîne vide à la place du commentaire. Dans ce cas, n'oubliez pas de convertir le type de la valeur NULL en données LONG VARCHAR.

Par exemple, les instructions suivantes, incluses dans un programme d'application en langage C, stockent un commentaire avec la séquence vidéo :

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_EXTERNAL;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2VIDEO(
          CURRENT SERVER,
          '/Employés/vidéo/adupont.mpg',
                 'MPEG1',
          :hvStorageType,
          'Séquence vidéo d''Anne')        /* commentaire */
       );
```

Les instructions suivantes, incluses dans un programme d'application en langage C, stockent une image sans commentaire :

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;
```

```
hvStorageType=MMDB_STORAGE_TYPE_INTERNAL;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2IMAGE(
          CURRENT SERVER,
          '/Employés/images/adupont.bmp',
          'GIF',
          :hvStorageType,
          Cast(NULL as LONG
VARCHAR)            /* pas de commentaire */
       );
```

Extraction d'objets image, audio, ou vidéo

Pour extraire un objet image, audio ou vidéo d'une table de base de données, utilisez la fonction UDF Content dans une instruction SQL SELECT. Vous pouvez extraire un objet pour le ranger dans la mémoire tampon, un fichier client, ou un fichier du serveur.

Formats d'extraction de la fonction UDF Content

La fonction UDF Content peut avoir différents formats selon son utilisation. Ces formats sont les suivants :

Format 1 : Extraction d'un objet vers la mémoire tampon du poste client ou vers un fichier client :

```
Content(
     handle,                  /* descripteur objet */
  );
```

Format 2 : Extraction d'un segment d'objet vers la mémoire tampon du poste client ou vers un fichier client :

```
Content(
     handle,                  /* descripteur objet */
     offset,                     /* adresse relative de début d'extraction (décalage) *
     size                        /* nombre d'octets à extraire */
  );
```

Format 3 : Extraction d'un objet vers un fichier du serveur :

```
Content(
     handle,                   /* descripteur objet */
     target_file,             /* nom du fichier du serveur */
     overwrite                /* 0=pas de remplacement du fichier cible s'il existe */
                              /* 1=remplacement du fichier cible */

  );
```

En outre, la fonction UDF Content comprend les formats suivants réservés aux objets image :

Format 4 : Extraction d'un objet image vers la mémoire tampon ou un fichier avec conversion de format :

```
Content(
     handle,                  /* descripteur objet */
     target format            /* format cible */
       );
```

Format 5 : Extraction d'un objet vers un fichier du serveur avec conversion de format :

```
Content(
     handle,           /* descripteur objet */
     target_file,      /* nom du fichier du serveur */
     overwrite,        /* 0=pas de remplacement du fichier cible s'il existe */
                       /* 1=remplacement du fichier cible */
     target format     /* format cible */

  );
```

Format 6 : Extraction d'un objet vers la mémoire tampon ou un fichier avec conversion de format et modifications supplémentaires :

```
Content(
     handle,           /* descripteur objet */
     target format,    /* format cible */
     conversion_options      /* options de conversion */
  );
```

Format 7 : Extraction d'un objet vers un fichier du serveur avec conversion de format et modifications supplémentaires :

```
Content(
     handle,           /* descripteur objet */
     target_file,      /* nom du fichier du serveur */
     overwrite,        /* 0=pas de remplacement du fichier cible s'il existe */
                       /* 1=remplacement du fichier cible */
     target format,    /* format cible */
     conversion_options   /* options de conversion */
  );
```

Par exemple, les instructions suivantes permettent d'extraire une image de la table Employés vers un fichier du serveur. (Cette opération correspond au format 3.)

```
EXEC SQL SELECT CONTENT(                /* fonction UDF d'extraction */
                 PICTURE,               /* descripteur image */
       '/Employés/images/adupont.bmp',     /* fichier cible */
 1)                                     /* remplacement du fichier cible */
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';
```

Les instructions ci-après, incluses dans un programme d'application en langage C, permettent l'extraction d'une image de la table Employés vers un fichier du serveur. Le format de l'image est converti à l'extraction. (Cette opération correspond au format 5.)

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_fname[255];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT CONTENT(                    /* fonction UDF d'extraction */
                 PICTURE,                             /* descripteur image */
       '/Employés/images/adupont.bmp',      /* fichier cible */
       1,                                   /* remplacement du fichier cible */
        'GIF')                                /* format cible */
INTO :hvImg_fname
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';
```

Extraction d'un objet et copie sur le poste client

Vous pouvez utiliser la fonction UDF Content pour extraire un objet image, audio ou vidéo et le copier dans la mémoire tampon ou un fichier du poste client sans conversion de format. En outre, l'extension Image vous permet de convertir le format d'une image lors de son extraction.

Extraction d'un objet et copie sur le poste client sans conversion de format

Utilisez un releveur de coordonnées LOB pour extraire un objet image, audio ou vidéo et le copier dans la mémoire tampon ou un fichier du poste client, ou bien procédez à l'extraction de l'objet LOB. Utilisez une variable de référence à un fichier pour extraire un objet image, audio ou vidéo vers un fichier client.

L'extraction d'un objet image, audio ou vidéo dans la mémoire tampon d'un poste client à l'aide d'une variable SQL ou dans un fichier du poste client à l'aide d'une variable de référence à un fichier est recommandée lorsque le contenu de l'objet est stocké dans une table de base de données en tant qu'objet BLOB. Si le contenu se trouve dans un fichier du serveur, la copie de ce dernier dans un fichier du poste client peut être plus rapide.

Spécifiez le descripteur de l'objet. Vous pouvez également spécifier l'adresse relative par rapport à l'octet 1 (décalage), à laquelle va commencer l'extraction, ainsi que le nombre d'octets à extraire.

Les instructions suivantes, incluses dans un programme d'application en langage C, utilisent un releveur de coordonnées LOB appelé `loc_audio` pour extraire une séquence audio et la copier dans la mémoire tampon du poste client :

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB_LOCATOR loc_audio;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT CONTENT(
        SOUND)                                  /* descripteur audio */
     INTO :loc_audio
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';
```

Vérification de la mémoire UDF disponible : Lorsque vous réalisez l'extraction d'un objet et sa copie dans la mémoire tampon du client, il convient de vérifier que le paramètre UDF_MEM_SZ (indiqué dans la configuration du gestionnaire de bases de données est au moins de 4 Mo. Vous pouvez modifier ce paramètre par la commande DB2 UPDATE DATABASE MANAGER CONFIGURATION. Pour plus d'informations sur la commande UPDATE DATABASE MANAGER, reportez-vous au manuel *DB2 Command Reference*.

Extraction d'une image et copie sur le poste client avec conversion

Utilisez un releveur de coordonnées pour extraire une image stockée et la copier dans la mémoire tampon d'un poste client avec conversion de format, ou effectuez l'extraction du LOB. Utilisez une variable de référence à un fichier pour extraire une image stockée et la copier dans un fichier du poste client avec conversion de format.

L'extraction d'un objet image dans la mémoire tampon d'un poste client à l'aide d'une variable SQL ou dans un fichier du poste client à l'aide d'une variable de référence à un fichier est recommandée lorsque le contenu de l'objet image est stocké dans une table de base de données en tant qu'objet BLOB. Si le contenu se trouve dans un fichier du serveur, la copie de ce dernier dans un fichier du poste client peut être plus rapide.

Lors de l'extraction d'une image après conversion de format, spécifiez son format cible, c'est-à-dire le format converti. Le tableau 5 à la page 91 identifie les conversions de format autorisées. Vous pouvez également spécifier les options de conversion identifiant des modifications supplémentaires, telles que la rotation ou le changement d'échelle, que vous souhaitez appliquer à l'image extraite. Le tableau 6 à la page 93 répertorie les options de conversion que vous pouvez spécifier.

Par exemple, les instructions ci-après, incluses dans un programme d'application en langage C, permettent d'extraire une image et de la copier

dans un fichier du poste client. L'image source est au format bitmap et elle est stockée dans une table de base de données en tant qu'objet BLOB. L'image extraite est convertie au format GIF et agrandie 3 fois.

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB_FILE Img_file;
  EXEC SQL END DECLARE SECTION;

strcpy (Img_file.name, "/Employés/images/adupont.gif");
Img_file.name_length= strlen(Img_file.name);
Img_file.file_options= SQL_FILE_CREATE;

EXEC SQL SELECT CONTENT(
                 PICTURE,                          /* descripteur image */
        'GIF',                              /* format cible */
        '-s 3.0')                           /* options de conversion */
     INTO :Img_file,
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';
```

Extraction d'un objet et copie dans un fichier du serveur

La fonction UDF Content vous permet d'extraire un objet image, audio ou vidéo et de le copier dans un fichier du serveur sans conversion de format. Elle vous permet également d'extraire une image et de la copier dans un fichier du serveur avec conversion de format.

Lors de l'extraction d'un objet image, audio ou vidéo et de sa copie dans un fichier du serveur sans conversion, spécifiez le descripteur objet, le nom du fichier cible et l'indicateur de remplacement. Ce dernier indique à l'extension si les données du fichier cible doivent ou non être remplacées par les données extraites s'il existe déjà un fichier de ce nom sur le serveur. Si le fichier cible n'existe pas, l'extension le crée.

Si vous spécifiez un indicateur de remplacement 1, l'extension remplace le fichier cible par les données extraites. Si vous entrez un indicateur de remplacement 0, il n'y a pas de remplacement : l'extraction n'a donc pas lieu.

L'indicateur de remplacement est ignoré si l'objet à extraire est stocké dans une table de base de données en tant qu'objet BLOB. Le fichier cible sera créé ou remplacé quel que soit l'indicateur de remplacement spécifié.

L'extraction d'un objet et sa copie dans un fichier du serveur renvoie le nom de ce fichier. Par exemple, l'instruction ci-après, incluse dans un programme d'application en langage C, extrait une séquence vidéo et la copie dans un fichier du serveur. Le nom du fichier du serveur est enregistré dans la variable SQL hvVid_fname.

```
EXEC SQL BEGIN DECLARE SECTION;
struct{
        short len;
        char data [250];
```

```
     }hvVid_fname[;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT CONTENT(
                VIDEO,                          /* descripteur vidéo */
        '/Employés/vidéo/adupont.mpg',          /* fichier du serveur */
        1)                                      /* remplacement du fichier cible */
     INTO :hvVid_fname;
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';
```

L'utilisation de la fonction UDF Content pour extraire un objet et le copier dans un fichier du serveur sans conversion est recommandée lorsque l'objet est stocké dans une table de base de données en tant qu'objet BLOB. Si l'objet est stocké dans un fichier du serveur, la copie de ce dernier dans le fichier cible peut être plus rapide.

Lors de l'extraction d'une image vers un fichier du serveur avec conversion de format, spécifiez le descripteur image, le nom du fichier cible, l'indicateur de remplacement et le format cible. Le tableau 5 à la page 91 répertorie les conversions de format admises. Vous pouvez également choisir d'indiquer comme format cible la valeur NULL, une chaîne vide ou encore la chaîne ASIS. Dans ce cas, l'image extraite est au même format que la source.

Par exemple, les instructions ci-après, incluses dans un programme d'application en langage C, permettent l'extraction d'une image et sa copie dans un fichier du serveur. L'image source est au format bitmap et elle est stockée dans une table de base de données en tant qu'objet BLOB. L'image extraite est convertie au format GIF. Le nom du fichier du serveur est enregistré dans la variable SQL hvImg_fname.

```
EXEC SQL BEGIN DECLARE SECTION;
struct{
        short len;
         char [400];
        }hvImg_fname[;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT CONTENT(
                PICTURE,                                /* descripteur image */
        '/Employés/images/adupont.gif',         /* fichier cible */
        1,                                      /* remplacement du fichier cible */
        'GIF')                                  /* format cible */
     INTO :hvImg_fname
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';
```

Le fichier du serveur doit être accessible : Lors de l'extraction d'un objet dans un fichier du serveur, vous devez spécifier le nom complet du fichier cible. Vous pouvez également vous assurer que les variables d'environnement

DB2IMAGEEXPORT, DB2AUDIOEXPORT et DB2VIDEOEXPORT sont définies pour résoudre sans erreur un nom de fichier incomplet.

Extraction et utilisation d'attributs

Lorsque vous stockez un objet image, audio ou vidéo dans une base de données, l'extension enregistre également les attributs de celui-ci dans la base. Lorsque vous mettez à jour un objet, l'extension modifie les attributs de l'objet stocké dans la base de données. Vous pouvez utiliser ces attributs dans les requêtes.

Les extensions créent des fonctions UDF pour chaque attribut qu'elles gèrent. Il en résulte que vous pouvez spécifier ces fonctions UDF dans des instructions SQL pour accéder aux attributs d'un objet et les utiliser. Le tableau 7 répertorie les attributs gérés par les extensions, ainsi que les fonctions UDF correspondantes. Il indique également les types d'objet pour chaque attribut. Certains attributs, tels que le format d'un objet et le nom de fichier, sont communs à tous les types d'objet ; c'est-à-dire qu'ils sont associés aussi bien aux objets image, audio que vidéo. D'autres attributs, tels la fréquence d'échantillonnage ou le type de compression, sont propres aux objets audio et vidéo.

Tableau 7. Attributs gérés par DB2 Extensions. Vous pouvez accéder à chaque attribut via sa fonction UDF.

Attribut	Fonction UDF	Image	Audio	Vidéo
Nom du fichier du serveur dans lequel l'objet est stocké	Filename	x	x	x
ID de l'utilisateur qui a stocké l'objet	Importer	x	x	x
Date et heure de stockage de l'objet	ImportTime	x	x	x
Taille de l'objet en octets	Size	x	x	x
ID du dernier utilisateur ayant mis l'objet à jour	Updater	x	x	x
Date et heure de la dernière mise à jour de l'objet	UpdateTime	x	x	x
Format de l'objet (par exemple, GIF ou MPEG1)	Format	x	x	x
Commentaires sur l'objet	Comment	x	x	x
Hauteur de l'objet (en pixels)	Height	x		x
Largeur de l'objet (en pixels)	Width	x		x
Nombre de couleurs de l'objet	NumColors	x		
Image miniature de l'objet	Thumbnail	x		x

Utilisation d'attributs

Tableau 7. Attributs gérés par DB2 Extensions (suite). Vous pouvez accéder à chaque attribut via sa fonction UDF.

Attribut	Fonction UDF	Image	Audio	Vidéo
Nombre d'octets renvoyés par échantillon dans une séquence audio ou dans une piste audio d'une vidéo	AlignValue		x	x
Nombre de bits utilisés pour représenter chaque échantillon	BitsPerSample		x	x
Nombre de canaux enregistrés	NumChannels		x	x
Durée (en secondes)	Duration		x	x
Fréquence d'échantillonnage (en échantillons par seconde)	SamplingRate		x	x
Nombre moyen d'octets par seconde transférés	BytesPerSec		x	
Numéro de piste audio pour l'instrument	FindInstrument		x	
Numéro de la piste nommée	FindTrackName		x	
Nom des instruments enregistrés	GetInstruments		x	
Numéros et noms des pistes des instruments enregistrés	GetTrackNames		x	
Battements de métronome par seconde	TicksPerSec		x	
Battements de métronome par noire	TicksPerQNote		x	
Rapport hauteur/largeur	AspectRatio			x
Format de compression vidéo (MPEG1, par exemple)	CompressType			x
Débit en images par seconde	FrameRate			x
Débit maximal (en octets par seconde)	MaxBytesPerSec			x
Nombre de pistes audio	NumAudioTracks		x	x
Nombre d'images vidéo	NumFrames			x
Nombre de pistes vidéo	NumVideoTracks			x

Vous pouvez utiliser une fonction UDF d'attribut comme argument dans la clause SELECT ou comme critère de recherche dans la clause WHERE d'une

instruction SQL. Lorsque vous appelez la fonction UDF, entrez le nom de la colonne de la table de base de données qui contient le descripteur objet.

Par exemple, l'instruction suivante utilise la fonction UDF Updater dans la clause SELECT d'une instruction SQL SELECT pour extraire l'ID du dernier utilisateur ayant modifié une image dans la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvUpdatr[30];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT UPDATER(PICTURE)
     INTO :hvUpdatr
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';
```

L'instruction suivante utilise la fonction UDF Filename dans la clause SELECT d'une instruction SELECT et la fonction UDF NumAudioTracks dans la clause WHERE pour rechercher les séquences vidéo de la table Employés qui possèdent des bandes son :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvVid_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(VIDEO)
     INTO :hvVid_fname
       FROM EMPLOYEE
     WHERE NUMAUDIOTRACKS(VIDEO)>0;
```

Extraction de commentaires

La fonction UDF Comment permet d'extraire les commentaires stockés avec un objet image, audio ou vidéo. Lors de l'extraction du commentaire d'un objet, indiquez la colonne de la table de base de données qui contient le descripteur objet. Par exemple, l'instruction suivante permet l'extraction d'un commentaire enregistré avec une séquence audio dans la table Employés.

```
EXEC SQL BEGIN DECLARE SECTION;
         struct  {
        short len;
        char data [32700];
       }hvComment
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT COMMENT(SOUND)
     INTO :hvComment
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';
```

Vous pouvez également utiliser la fonction UDF Comment comme prédicat dans la clause WHERE d'une requête SQL. Par exemple, l'instruction suivante permet l'extraction de toutes les images de la table Employés qui ont subi une «retouche».

Extraction de commentaires

```
EXEC SQL BEGIN DECLARE SECTION;
         struct  {
        short len;
        char data [250];
       }hvImg_fname
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(PICTURE)
INTO :hvImg_fname
       FROM EMPLOYEE
     WHERE COMMENT(PICTURE)
           LIKE '%retouche';
```

Mise à jour d'un objet image, audio ou vidéo

Utilisez la fonction UDF Content dans une instruction SQL UPDATE pour mettre à jour un objet image, audio ou vidéo dans une table de base de données. Utilisez la fonction UDF Replace dans une instruction SQL UPDATE pour mettre à jour un objet image, audio ou vidéo dans une table de base de données, ainsi que le commentaire associé à l'objet. Dans les deux cas, l'extension met à jour les attributs associés à l'objet.

Vous pouvez mettre à jour un objet stocké dans une table de base de données en tant qu'objet BLOB ou dans un fichier du serveur (la base de données comprend un pointeur sur ce fichier). La source de la mise à jour peut se trouver dans une mémoire tampon, un fichier client, ou un fichier de serveur.

Le tableau 5 à la page 91 répertorie les formats de mise à jour autorisés pour les objets image, audio et vidéo. Cependant, vous pouvez également mettre à jour un objet dont le format n'est pas reconnu par l'extension. Dans ce cas, les attributs de l'objet ont été définis par l'utilisateur au moment du stockage. Lorsque vous mettez à jour un tel objet, vous devez indiquer ses attributs modifiés. Utilisez la fonction UDF ContentA dans une instruction SQL UPDATE pour mettre à jour un objet image, audio ou vidéo dont les attributs sont définis par l'utilisateur dans une base de données. Utilisez la fonction UDF ReplaceA dans une instruction SQL UPDATE pour mettre à jour un objet image, audio ou vidéo dont les attributs sont définis par l'utilisateur dans une table de base de données, ainsi que le commentaire associé à l'objet. Dans ce cas, vous devez préciser les attributs de l'objet, son format et, lorsqu'il s'agit d'un objet vidéo, son format de compression.

Vous pouvez également mettre à jour la miniature d'un objet image ou vidéo stocké.

Validation de l'opération de mise à jour : Validez l'unité d'oeuvre après avoir mis à jour l'objet image, audio ou vidéo dans une base de données. Cela permet de déverrouiller l'objet (celui-ci ayant été verrouillé par les extensions pour sa mise à jour).

Formats de mise à jour de la fonction UDF Content

La fonction UDF Content peut avoir différents formats selon son utilisation. Ces formats sont les suivants :

Format 1 : Mise à jour d'un objet à partir de la mémoire tampon ou d'un fichier client :

```
Content(
     handle,                 /* descripteur objet */
     content,                /* contenu de l'objet */
     source_format,          /* format source */
     target_file             /* nom du fichier cible pour stockage sur */
                             /* un serveur */
                             /* de fichiers ou valeur NULL pour stockage */
                             /* en tant qu'objet BLOB */
  );
```

Format 2 : Mise à jour d'un objet à partir d'un fichier du serveur.

```
Content(
     handle,                 /* descripteur objet */
     source_file,            /* nom du fichier du serveur */
     source_format,          /* format source */
     stortype                /* MMDB_STORAGE_TYPE_EXTERNAL=stockage */
                             /* sur un serveur de fichiers */
                             /* MMDB_STORAGE_TYPE_INTERNAL=stockage */
                             /* en tant qu'objet BLOB */
  );
```

Format 3 : Mise à jour d'un objet dont les attributs sont définis par l'utilisateur, à partir de la mémoire tampon ou d'un fichier du poste client :

```
Content(
     handle,                 /* descripteur objet */
     content,                /* contenu de l'objet */
     target_file,            /* nom du fichier cible pour stockage sur le */
                             /* serveur de fichiers */
                             /* ou NULL pour stockage dans une table */
                             /* en tant qu'objet BLOB */
     attrs,                  /* attributs définis par l'utilisateur */
     thumbnail               /* miniature (objet image et vidéo uniquement) */
  );
```

Format 4 : Mise à jour d'un objet dont les attributs sont définis par l'utilisateur à partir d'un fichier du serveur :

```
Content(
     handle,                  /* descripteur objet */
     source_file,             /* nom du fichier source */
     stortype,                /* MMDB_STORAGE_TYPE_EXTERNAL=stockage */
                              /* sur un serveur de fichiers*/
                              /* MMDB_STORAGE_TYPE_INTERNAL=stockage */
```

```
                                /* en tant qu'objet BLOB */
    attrs,                      /* attributs définis par l'utilisateur */
    thumbnail                   /* miniature (objet image et vidéo uniquement) */
  );
```

Pour les images uniquement, la fonction UDF Content présente des formats supplémentaires :

Format 5 : Mise à jour d'un objet à partir de la mémoire tampon ou d'un fichier client avec conversion de format :

```
Content(
     handle,                    /* descripteur objet */
     content,                   /* contenu de l'objet */
     source format,             /* format source */
     target format,             /* format cible */
     target_file                /* nom du fichier cible pour stockage sur */
                                /* un serveur de fichiers */
                                /* ou NULL pour stockage dans une table */
                                /* en tant qu'objet BLOB */
  );
```

Format 6 : Mise à jour d'un objet à partir d'un fichier du serveur avec conversion de format :

```
Content(
     handle,                    /* descripteur objet */
     source_file,               /* nom du fichier du serveur */
     source format,             /* format source */
     target format,             /* format cible */
     target_file                /* nom du fichier cible pour stockage sur */
                                /* un serveur de fichiers */
                                /* ou NULL pour stockage dans une table */
                                /* en tant qu'objet BLOB */
  );
```

Format 7 : Mise à jour d'un objet à partir de la mémoire tampon ou d'un fichier client avec conversion de format et modifications supplémentaires :

```
Content(
     handle,                    /* descripteur objet */
     content,                   /* contenu de l'objet */
     source format,             /* format source */
     target format,             /* format cible */
     conversion_options,        /* options de conversion  */
     target_file                /* nom du fichier cible pour stockage sur */
                                /* un serveur de fichiers */
                                /* ou NULL pour stockage dans une table */
                                /* en tant qu'objet BLOB */
  );
```

Format 8 : Mise à jour d'un objet à partir d'un fichier du serveur avec conversion de format et modifications supplémentaires :

```
Content(
     handle,                 /* descripteur objet */
     source_file,            /* nom du fichier du serveur */
     source format,          /* format source */
     target format,          /* format cible */
     conversion_options,     /* options de conversion  */
     target_file             /* nom du fichier cible pour stockage sur */
                             /* un serveur de fichiers */
                             /* ou NULL pour stockage dans une table */
                             /* en tant qu'objet BLOB */
  );
```

Par exemple, les instructions ci-après, incluses dans un programme d'application en langage C, mettent à jour une image dans la table Employés. Les données de mise à jour se trouvent dans un fichier du serveur appelé adupont.bmp. L'image modifiée est stockée dans la table Employés en tant qu'objet BLOB. (Cette opération correspond au format 2.)

```
EXEC SQL UPDATE EMPLOYEE
   SET PICTURE=CONTENT(
                PICTURE,                          /*descripteur image*/
                '/Employés/nouvimg/adupont.bmp', /*fichier source*/
           'ASIS',                          /*conservation du format de l'image*/
                '');                        /*stockage de l'image dans la */
                                          /*BD en tant qu'objet BLOB*/
       WHERE NAME='Anne Dupont';
```

Les instructions ci-après, incluses dans un programme d'application en langage C, mettent à jour la même image que dans l'exemple précédent, à cette différence près que l'image est convertie du BMP au format GIF lors de la mise à jour. (Cela correspond au format 6 ci-dessus.)

```
EXEC SQL UPDATE EMPLOYEE
   SET PICTURE=CONTENT(
                PICTURE,                          /*descripteur image*/
                '/Employés/nouvimg/adupont.bmp', /*fichier source*/
                'BMP',                           /*format source*/
                'GIF',                           /*format cible*/
                '');                             /*stockage de l'image dans */
                                          /* la BD en tant qu'objet BLOB*/
       WHERE NAME='Anne Dupont';
```

Formats de mise à jour de la fonction Replace

La fonction UDF Replace peut avoir différents formats selon son utilisation. Ces formats sont les suivants :

Format 1 : Mise à jour d'un objet et son commentaire à partir de la mémoire tampon ou d'un fichier client :

```
Replace(
     handle,                  /* descripteur objet */
     content,                 /* contenu de l'objet */
     source_format,           /* format source */
```

```
    target_file,              /* nom du fichier cible pour stockage */
                              /* dans un fichier */
    comment                   /* commentaire */
  );
```

Format 2 : Mise à jour d'un objet et de son commentaire à partir d'un fichier du serveur

```
Replace(
    handle,                   /* descripteur objet */
    source_file,              /* nom du fichier du serveur */
    source_format,            /* format source */
    stortype,                 /* MMDB_STORAGE_TYPE_EXTERNAL=stockage */
                              /* sur un serveur de fichiers*/
                              /* MMDB_STORAGE_TYPE_INTERNAL=stockage */
                              /* en tant qu'objet BLOB */
    comment                   /* commentaire */
  );
```

Format 3 : Remplacement d'un objet dont les attributs sont définis par l'utilisateur et de son commentaire à partir de la mémoire tampon ou d'un fichier du poste client

```
Replace(
    handle,                   /* descripteur objet */
    content,                  /* contenu de l'objet */
    target_file,              /* nom du fichier cible pour stockage */
                              /* dans un fichier */
                              /* ou NULL pour stockage dans une table */
                              /* en tant qu'objet BLOB */
    comment,                  /* commentaire */
    attrs,                    /* attributs définis par l'utilisateur */
    thumbnail                 /* miniature */
  );
```

Format 4 : Stockage d'un objet dont les attributs sont définis par l'utilisateur, à partir d'un fichier du serveur :

```
Replace(
    handle,                   /* descripteur objet */
    source_file,              /* nom du fichier du serveur */
    stortype,                 /* MMDB_STORAGE_TYPE_EXTERNAL=stockage */
                              /* sur un serveur de fichiers*/
                              /* MMDB_STORAGE_TYPE_INTERNAL=stockage */
                              /* en tant qu'objet BLOB */
    comment,                  /* commentaire */
    attrs,                    /* attributs définis par l'utilisateur */
    thumbnail                 /* miniature */
  );
```

Pour les objets image uniquement, la fonction UDF Replace présente des formats supplémentaires :

Format 5 : Mise à jour d'un objet et de son commentaire à partir de la mémoire tampon ou d'un fichier client avec conversion de format :

```
Replace(
     handle,                 /* descripteur objet */
     content,                /* contenu de l'objet */
     source_format,          /* format source */
     target_format,          /* format cible */
     target_file,            /* nom du fichier cible pour stockage sur le */
                             /* serveur de fichiers */
                             /* ou NULL pour stockage dans une table */
                             /* en tant qu'objet BLOB */
     comment                 /* commentaire */
  );
```

Format 6 : Mise à jour d'un objet et de son commentaire à partir d'un fichier du serveur avec conversion de format :

```
Replace(
     handle,                 /* descripteur objet */
     source_file,            /* nom du fichier du serveur */
     source_format,          /* format source */
     target_format,          /* format cible */
     target_file,            /* MMDB_STORAGE_TYPE_EXTERNAL=stockage */
                             /* sur un serveur de fichiers */
                             /* MMDB_STORAGE_TYPE_INTERNAL=stockage */
                             /* en tant qu'objet BLOB */
     comment                 /* commentaire */
  );
```

Format 7 : Mise à jour d'une image et de son commentaire à partir de la mémoire tampon ou d'un fichier client avec conversion de format et modifications supplémentaires :

```
Replace(
     handle,                 /* descripteur objet */
     content,                /* contenu de l'objet */
     source_format,          /* format source */
     target_format,          /* format cible */
     conversion_options,     /* options de conversion  */
     target_file,            /* nom du fichier cible pour stockage sur le */
                             /* serveur de fichiers */
                             /* ou NULL pour stockage dans une table */
                             /* en tant qu'objet BLOB */
     comment                 /* commentaire */
  );
```

Format 8 : Mise à jour d'un objet et de son commentaire à partir d'un fichier du serveur avec conversion de format et modifications supplémentaires :

```
Replace(
     handle,                 /* descripteur objet */
     source_file,            /* nom du fichier du serveur */
     source_format,          /* format source */
     target_format,          /* format cible */
     conversion_options,     /* options de conversion  */
     target_file,            /* MMDB_STORAGE_TYPE_EXTERNAL=stockage */
                             /* sur un serveur de fichiers */
                             /* MMDB_STORAGE_TYPE_INTERNAL=stockage */
                             /* en tant qu'objet BLOB */
     comment                 /* commentaire */
  );
```

Par exemple, les instructions ci-après, incluses dans un programme d'application en langage C, mettent à jour une séquence audio et son commentaire dans la table Employés. Les données de mise à jour se trouvent dans un fichier du serveur appelé adupont.wav. La séquence audio modifiée est stockée dans la table Employés en tant qu'objet BLOB sans conversion de format (l'extension Audio ne prend pas en charge la conversion de format). Cette opération correspond au format 2.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_INTERNAL;

EXEC SQL UPDATE EMPLOYEE
   SET SOUND=REPLACE(
                SOUND,                          /*descripteur audio*/
                '/Employés/nouvaud/adupont.wav', /*fichier source */
                'WAV',                          /*conserv.format audio*/
                :hvStorageType,                  /*stockage de la séquence*/
                                                 /*audio dans la BD*/
                                                 /*en tant que BLOB*/
                'Nouveau message de bienvenue d''Anne')   /*commentaire*/
       WHERE NAME='Anne Dupont';
```

L'exemple suivant illustre la mise à jour d'une image et de son commentaire. Les données de mise à jour se trouvent dans un fichier du serveur. L'image modifiée est stockée dans la table Employés en tant qu'objet BLOB et convertie du format BMP au format GIF au moment de la mise à jour. (Cela correspond au format 6 ci-dessus.)

```
EXEC SQL UPDATE EMPLOYEE
       SET PICTURE=REPLACE(
                PICTURE,                        /*descripteur image*/
                '/Employés/nouvimg/adupont.bmp', /*fichier source*/
                'BMP',                          /*format source*/
                'GIF',                          /*format cible*/
                ''                              /*stockage de l'image dans la*/
                                                /*BD en tant qu'objet BLOB*/
                  'Nouvelle photo d''Anne')
   WHERE NAME='Anne Dupont';                    /* commentaire utilisateur */
```

Mise à jour d'un objet à partir du poste client

Utilisez une variable SQL ou une variable de référence à un fichier pour mettre à jour le contenu d'un objet image, audio ou vidéo d'une mémoire tampon client ou d'un fichier client.

Si les données de mise à jour se trouvent dans un fichier sur le poste client, utilisez une variable de référence à un fichier pour les transférer. Par exemple, les instructions ci-après, incluses dans un programme d'application en langage C, définissent une variable de référence à un fichier appelé `fichier_audio` qui sera utilisée pour la mise à jour d'une séquence audio stockée en tant qu'objet BLOB dans la table de base de données. Les données de mise à jour se trouvent dans un fichier du poste client. A noter que la zone `file_options` de la variable de référence à un fichier a pour valeur `SQL_FILE_READ` et qu'elle sert d'argument à la fonction UDF Content.

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB_FILE fichier_audio;
  EXEC SQL END DECLARE SECTION;

strcpy (fichier_audio.name, "/Employés/nouvson/adupont.wav");
fichier_audio.name_length= strlen(fichier_audio.name);
fichier_audio.file_options= SQL_FILE_READ;

EXEC SQL UPDATE EMPLOYEE
   SET SOUND=CONTENT(
               SOUND,
               :fichier_audio            /*variable de référence à un fichier*/
               'WAVE',                   /*conservation du format de l'image*/
               CAST(NULL as LONG VARCHAR))

      WHERE NAME='Anne Dupont';
```

Si l'objet de la mise à jour se trouve dans la mémoire tampon du poste client, utilisez une variable SQL pour en transférer le contenu. L'exemple ci-après de programme d'application en C utilise une variable SQL appelée `seq_vidéo` pour transmettre une séquence vidéo à mettre à jour. Le commentaire associé à la séquence est également modifié. La séquence vidéo est stockée dans une table de base de données en tant qu'objet BLOB. A noter que la variable SQL sert d'argument à la fonction UDF Replace.

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB (2M) séq_vidéo
  EXEC SQL END DECLARE SECTION;

EXEC SQL UPDATE EMPLOYEE
   SET VIDEO=REPLACE(
               VIDEO,
                :séq_vidéo  /*variable SQL*/
         'MPEG1',
                CAST(NULL as LONG VARCHAR),

                'Nouvelle séquence vidéo d''Anne')
       WHERE NAME='Anne Dupont';
```

Vérification de la mémoire UDF disponible : Lorsque vous mettez à jour un objet dont les données de mise à jour se trouvent dans la mémoire tampon du

client, assurez-vous que le paramètre UDF_MEM_SZ du fichier de configuration du gestionnaire de bases de données est au moins de 4 Mo. Vous pouvez modifier ce paramètre par la commande DB2 UPDATE DATABASE MANAGER CONFIGURATION.

Mise à jour d'un objet à partir du serveur

Lorsque les données de mise à jour d'un objet image, audio ou vidéo se trouvent dans un fichier du serveur, indiquez le chemin du fichier comme argument pour la fonction UDF. Par exemple, l'instruction ci-après, incluse dans un programme d'application en langage C, met à jour une image dans une base de données. L'image à mettre à jour se trouve dans un fichier du serveur et la base de données contient un pointeur sur ce fichier. Les données de mise à jour se trouvent également dans un fichier du serveur.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_EXTERNAL;

EXEC SQL UPDATE EMPLOYEE
   SET PICTURE=CONTENT(
                PICTURE,                              /* descripteur image */
                '/Employés/nouvimg/adupont.bmp',      /* fichier source*/
                  'ASIS',
                :hvStorageType)
       WHERE NAME='Anne Dupont';
```

Spécification du chemin correct : Lorsque vous modifiez un objet dont la source se trouve dans un fichier du serveur, vous pouvez spécifier le nom complet du fichier ou son nom relatif. Si vous indiquez un nom de fichier relatif, assurez-vous que les variables d'environnement appropriées du serveur DB2 contiennent le chemin correct du fichier. Pour plus de détails sur la définition des variables d'environnement, reportez-vous à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

Mise à jour d'un objet stocké dans une base de données ou dans un fichier

Vous pouvez mettre à jour un objet image, audio ou vidéo stocké dans une table de base de données en tant qu'objet BLOB ou dans un fichier du serveur (la base de données contient un pointeur sur ce dernier).

Lorsque vous mettez à jour un objet provenant de la mémoire tampon ou d'un fichier client, vous indiquez le type de stockage (objet BLOB ou fichier du serveur) dans le paramètre nom de fichier (filename). La saisie d'un nom de fichier indique que l'objet à mettre à jour se trouve dans un fichier du serveur. Si vous indiquez comme nom de fichier NULL, cela signifie que l'objet à mettre à jour est stocké en tant qu'objet BLOB dans une table de base de données.

Par exemple, les instructions ci-après, incluses dans un programme d'application en langage C, mettent à jour une image se trouvant dans un fichier du serveur. Les données de mise à jour se trouvent en mémoire tampon sur le poste client. Le commentaire est également modifié.

```
EXEC SQL BEGIN DECLARE SECTION;
  SQL TYPE IS BLOB (2M) Img_buf
  EXEC SQL END DECLARE SECTION;

EXEC SQL UPDATE EMPLOYEE
       SET PICTURE=REPLACE(
                 PICTURE,
                   :Img_buf,
                   'ASIS',
                   '/Employés/nouvimg/adupont.bmp',  /*mise à jour d'une image*/
                                          /*dans un fichier du serveur*/
                   'Nouvelle photo d''Anne')
       WHERE NAME='Anne Dupont';
```

Lors de la mise à jour d'un objet à partir d'un fichier du serveur, indiquez MMDB_STORAGE_TYPE_INTERNAL pour modifier un objet stocké dans une table de base de données en tant qu'objet BLOB. Pour mettre à jour un objet stocké dans un fichier du serveur, spécifiez MMDB_STORAGE_TYPE_EXTERNAL.

Par exemple, le programme d'application suivant en C met à jour une séquence audio dont le contenu se trouve dans un fichier du serveur. Les données de mise à jour se trouvent également dans un fichier du serveur.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_EXTERNAL;

EXEC SQL UPDATE EMPLOYEE
   SET SOUND=CONTENT(
                 SOUND,
                 '/Employés/nouvimg/adupont.wav',
          'WAVE',
                 :hvStorageType)     /*mise à jour d'une séquence audio*/
                                     /*dans un fichier du serveur*/
       WHERE NAME='Anne Dupont';
```

Indication du format de mise à jour

Lors de la mise à jour d'un objet, vous devez indiquer son format. Les extensions stockent les données de mise à jour d'un objet image, audio ou vidéo dans leur format d'origine. Pour les objets image uniquement, l'extension Image peut convertir le format de l'image mise à jour. Si vous souhaitez convertir l'image, il convient de spécifier le format des données de mise à jour et le format de l'image cible (c'est-à-dire le format de stockage de l'image modifiée).

Indication du format de mise à jour sans conversion

Lors d'une mise à jour sans conversion de format, indiquez le format de l'objet image, audio ou vidéo source. Par exemple, l'instruction ci-après, incluse dans un programme d'application en langage C, met à jour une image bitmap (BMP) qui réside dans un fichier du serveur. Le format de l'image mise à jour est conservé.

```
EXEC SQL UPDATE EMPLOYEE
   SET PICTURE=CONTENT(
                PICTURE,
                '/Employés/nouvimg/adupont.bmp',
                'BMP',                          /*format image*/
                '')
       WHERE NAME='Anne Dupont';
```

Vous pouvez également indiquer comme format la valeur NULL, une chaîne vide ou, dans le cas de l'extension Image uniquement, la chaîne de caractères ASIS. L'extension détermine alors le format en examinant les données source.

Utilisation de NULL ou ASIS pour les formats reconnus : Spécifiez la valeur NULL, une chaîne vide ou ASIS uniquement si le format est reconnaissable par l'extension, c'est-à-dire s'il s'agit des formats répertoriés pour cette extension dans le tableau 5 à la page 91. Sinon, l'extension ne peut pas mettre à jour l'objet.

Indication du format et des options de conversion pour une mise à jour avec conversion de format

Lors d'une mise à jour avec conversion de format, indiquez le format des images source et cible. Le tableau 5 à la page 91, répertorie les conversions de format admises.

En outre, vous pouvez spécifier les options de conversion identifiant des modifications supplémentaires, telles que la rotation ou la compression, que vous souhaitez appliquer à l'image modifiée. Vous spécifiez chaque option de conversion grâce à un paramètre et à une valeur associée. Les paramètres et les valeurs autorisées sont répertoriés dans le tableau 6 à la page 93. Vous pouvez effectuer plusieurs changements sur une image mise à jour en spécifiant plusieurs couples paramètres/valeurs.

L'exemple suivant illustre la mise à jour d'une image se trouvant dans un fichier du serveur. Les données de mise à jour sont au format bitmap (BMP). Le format est converti de BMP en GIF à la mise à jour.

```
EXEC SQL UPDATE EMPLOYEE
   SET PICTURE=CONTENT(
                PICTURE,
                '/Employés/nouvimg/adupont.bmp',
```

```
                'BMP',                                  /*format source*/
                'GIF',                                  /*format cible*/
                '')
      WHERE NAME='Anne Dupont';
```

Dans l'exemple ci-après, la même image est convertie au format GIF lors de la mise à jour. En outre, l'image subit une rotation de 90 degrés dans le sens des aiguilles d'une montre, lors de la mise à jour.

```
EXEC SQL UPDATE EMPLOYEE
   SET PICTURE=CONTENT(
                PICTURE,
                '/Employés/nouvimg/adupont.bmp',
                'BMP',                          /*format source*/
                'GIF',                          /*format cible*/
                '-r 1',                         /* options de conversion */
                '')
      WHERE NAME='Anne Dupont';
```

Mise à jour d'un objet dont les attributs sont définis par l'utilisateur

Lors de la mise à jour d'un objet image, audio ou vidéo stocké avec des attributs définis par l'utilisateur, vous devez spécifier les attributs des données de mise à jour. Définissez les valeurs des attributs sous forme de structure. La structure d'attributs doit être contenue dans la zone de données de la variable LONG VARCHAR FOR BIT DATA de la fonction UDF.

Le code UDF qui s'exécute sur le serveur attend toujours des données au format «big endian». Ce dernier est utilisé pour la plupart des plateformes UNIX. Dans le cas d'un objet au format «little endian», vous devez préparer les données dont les attributs sont définis par l'utilisateur de sorte que le code UDF sur le serveur puisse les traiter correctement. Le format little endian est, en règle générale, utilisé sur les plateformes Intel et d'autres types de microprocesseurs. (Même si vous n'avez pas recours au format little endian, il est recommandé de préparer les données dont les attributs sont définis par l'utilisateur.) Utilisez l'interface API DBiPrepareAttrs pour préparer les attributs des objets image, l'interface API DBaPrepareAttrs pour les objets audio, et l'interface API DBvPrepareAttrs pour les objets vidéo.

Par exemple, les instructions ci-après, incluses dans un programme d'application en langage C, mettent à jour une image se trouvant dans un fichier du serveur. L'image a un format défini par l'utilisateur, une hauteur de 640 pixels et une largeur de 480 pixels. Notez que les attributs sont préparés avant la mise à jour de l'image.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
struct {
        short len;
        char data [400];
       }hvImgattrs;
  EXEC SQL END DECLARE SECTION;
```

```
DB2IMAGEATTRS    *pimgattr;

hvStorageType=MMDB_STORAGE_TYPE_INTERNAL;

pimgattr = (DB2IMAGEATTRS *) hvImgattrs.data;
strcpy(pimgattr→Format,"FormatI");
pimgattr→width=640;
pimgattr→height=480;
hvImgattrs.len=sizeof(DB2IMAGEATTRS);

DBiPrepareAttrs(pimgattr);

EXEC SQL UPDATE EMPLOYEE
       SET PICTURE=REPLACE(
                 PICTURE,
                 '/Employés/nouvimg/adupont.bmp',
            :hvStorageType,
                 'Nouvelle photo d''Anne',
                 :ImgAttrs,   /*attributs définis par l'utilisateur*/
            CAST(NULL as LONG VARCHAR))
       WHERE NAME='Anne Dupont';
```

Mise à jour d'une miniature (objets image et vidéo uniquement)

La fonction UDF Thumbnail permet de mettre à jour la miniature d'un objet image ou vidéo (ou d'ajouter une miniature si l'image ou la séquence vidéo n'en possède pas). Lorsque vous utilisez cette fonction UDF, spécifiez le descripteur de l'objet dont vous voulez modifier la miniature, ainsi que le contenu de la miniature nouvelle ou modifiée.

Générez la miniature dans votre programme car les extensions ne fournissent pas d'API à cette fin. C'est vous qui définissez la taille et le format de la miniature de mise à jour. Créez également la structure de la miniature dans votre programme et spécifiez-la dans la fonction UDF.

Par exemple, l'instruction suivante, incluse dans un programme d'application en langage C, met à jour la miniature associée à une séquence vidéo stockée :

```
EXEC SQL BEGIN DECLARE SECTION;
struct {
        short len;
       char data[10000];
     }hvThumbnail;
  EXEC SQL END DECLARE SECTION;

/*Création de la miniature et stockage dans la variable hvThumbnail*/


EXEC SQL UPDATE  EMPLOYEE
   SET picture=Thumbnail(
                 picture,
                 :hvThumbnail)
   WHERE name='Anne Dupont';
```

Vous pouvez également mettre à jour la miniature d'un objet image ou vidéo dont les attributs sont définis par l'utilisateur. En réalité, lorsque vous modifiez un tel objet image ou vidéo, vous devez spécifier une miniature en entrée. Si vous ne souhaitez pas mettre à jour la miniature en même temps que l'objet, spécifiez la valeur NULL ou une chaîne vide en remplacement de la référence à la miniature.

Les instructions ci-après, incluses dans un programme d'application en langage C, mettent à jour une séquence vidéo associée à des attributs définis par l'utilisateur, ainsi que sa miniature.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
struct {
        short len;
        char data [400];
       }hvVidattrs;
         struct  {
        short len;
       char data[10000];
     }hvThumbnail;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_EXTERNAL;

MMDBVideoAttrs         *pvideoAttr;
pvideoAttr=(MMDBVideoAttrs *)hvVidattrs.data;
strcpy(pvideoAttr→cformat,"Formatv");
hvVidattrs.len=sizeof(MMDBVideoAttrs);

/* Mise à jour d'une séquence vidéo et de sa miniature */

EXEC SQL UPDATE EMPLOYEE
   SET VIDEO=REPLACE(
                VIDEO,
                 '/Employés/nouvvid/adupont.mpg',
          :hvStorageType,
                 'Nouvelle séquence vidéo d''Anne',
                 :VidAttrs,
                 :hvThumbnail)                /*miniature*/
       WHERE NAME='Anne Dupont';
```

Mise à jour d'un commentaire

Vous pouvez mettre à jour un commentaire isolément ou en même temps que l'objet qui lui est associé.

Utilisez la fonction UDF Comment pour modifier uniquement le commentaire. Indiquez le nouveau commentaire ainsi que la colonne de la table contenant le descripteur objet. Servez-vous d'une variable SQL pour transférer le commentaire sur le serveur. Par exemple, les instructions ci-après déclarent la variable SQL `hvRemarks` qui sera utilisée pour la mise à jour du commentaire d'une séquence vidéo stockée.

```
EXEC SQL BEGIN DECLARE SECTION;
struct {
        short len;
        char data [40];
  }hvRemarks;
  EXEC SQL END DECLARE SECTION;

/* Extraction de l'ancien commentaire */

EXEC SQL SELECT COMMENT(VIDEO)
     INTO :hvRemarks
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';

/* Ajout à l'ancien commentaire */

hvRemarks.data[Remarks.len]='\0';
hvRemarks.len=strlen(hvRemarks.data);

strcat (hvRemarks.data, "Nouvelle séquence vidéo");
EXEC SQL UPDATE EMPLOYEE
     SET VIDEO=COMMENT(VIDEO, :hvRemarks)
     WHERE NAME = 'Anne Dupont';
```

Utilisez la fonction UDF Replace pour mettre à jour un commentaire en même temps que l'objet auquel il est associé. Par exemple, les instructions ci-après mettent à jour une séquence vidéo stockée dans un fichier du serveur ainsi que son commentaire.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType=MMDB_STORAGE_TYPE_EXTERNAL;

EXEC SQL UPDATE EMPLOYEE
   SET VIDEO=REPLACE(
                VIDEO,
                 '/Employés/nouvvid/adupont.mpg',
          'MPEG1',
          :hvStorageType,
                 'Nouvelle séquence vidéo d''Anne')   /*Nouveau commentaire*/
       WHERE NAME='Anne Dupont';
```

Chapitre 12. Affichage ou lecture d'un objet image, audio ou vidéo

Le présent chapitre explique comment utiliser les interfaces de programmation d'applications DB2 Extensions pour afficher ou lire un objet image, audio ou vidéo stocké dans une base de données.

Utilisation des API d'affichage ou de lecture

Les API DB2 Extensions permettent d'afficher un objet image ou vidéo stocké dans une base de données. Cet affichage peut s'effectuer en version miniaturisée ou en grandeur réelle. Il est également possible d'utiliser ces API pour la lecture des objets audio ou vidéo stockés dans une base de données.

Pour l'affichage ou la lecture d'objets, utilisez les API suivantes :

Utilisez cette API	Pour
DBiBrowse	L'affichage d'un objet image ou vidéo
DBaPlay	La lecture d'une séquence audio
DBvPlay	La lecture d'une séquence vidéo ou l'affichage d'une image vidéo

Pour utiliser l'une de ces API, vous devez préciser :

- le nom du programme de lecture ou d'affichage ;
- l'emplacement de l'objet à afficher ou lire (dans une table de base de données en tant qu'objet BLOB ou dans un fichier indiqué par la table) ;
- le nom du fichier source ou le descripteur stocké dans la table de base de données ;
- le moment auquel l'objet doit être affiché ou lu (avant ou après la fermeture du programme d'exécution).

Identification d'un programme d'affichage ou de lecture

Spécifiez le nom de l'afficheur d'images, du lecteur audio ou du lecteur vidéo à utiliser en le faisant suivre de %s. DB2 Extensions remplace ultérieurement le signe %s par le nom du fichier contenant l'objet. Par exemple, l'instruction suivante d'un programme d'application en C démarre l'afficheur d'images OS/2 (ib) :

```
rc = DBiBrowse(
     "ib %s", /* programme d'affichage d'image */
      MMDB_PLAY_FILE,
     "/Employés/images/adupont.bmp",
     MMDB_PLAY_NO_WAIT
     );
```

Il est possible d'indiquer une valeur nulle à la place du nom de programme. Dans ce cas, l'extension démarre l'afficheur d'images, le lecteur audio ou le lecteur vidéo par défaut indiqué dans la variable d'environnement DB2IMAGEBROWSER, DB2AUDIOPLAYER ou DB2VIDEOPLAYER. Pour plus d'informations sur l'utilisation des variables d'environnement par DB2 Extensions, reportez-vous à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

Par exemple, l'instruction suivante d'un programme d'application en C permet de démarrer le lecteur audio par défaut identifié par la variable d'environnement DB2AUDIOPLAYER :

```
rc = DBaPlay(
     NULL,                          /* utilisation du lecteur audio par défaut */
      MMDB_PLAY_FILE,
     "/Employés/sons/dupont.wav",
     MMDB_PLAY_NO_WAIT
     );
```

La variable d'environnement doit identifier un programme : Si vous demandez un programme d'affichage ou de lecture par défaut (en spécifiant une valeur nulle), vérifiez que la variable d'environnement correspondante mentionne bien un tel programme. Si aucun programme de ce type n'est mentionné, l'API renvoie un code d'erreur.

Spécification du contenu de l'objet BLOB ou du fichier

Vous pouvez afficher ou lire un objet stocké dans une table de base de données sous forme d'objet BLOB ou d'objet dont le contenu est stocké dans un fichier (et est désigné à partir de la table de base de données). Dans le cas d'un objet BLOB, indiquez MMDB_PLAY_HANDLE. Si le contenu de l'objet est stocké dans un fichier, indiquez MMDB_PLAY_FILE. MMDB_PLAY_HANDLE et MMDB_PLAY_FILE sont des constantes définies par les extensions.

Par exemple, l'instruction suivante, incluse dans un programme d'application en langage C, permet d'exécuter un objet vidéo dont le contenu se trouve dans un fichier :

```
rc = DBvPlay(
     "explore %s",
     MMDB_PLAY_FILE,                    /* contenu dans un fichier */
     "/Employés/vidéos/dupont.mpg",
     MMDB_PLAY_NO_WAIT
     );
```

Les programmes d'affichage et de lecture acceptent généralement des entrées provenant d'un fichier. Si vous indiquez la valeur MMDB_PLAY_FILE, l'extension l'utilise dans les variables d'environnement pour résoudre le nom et le chemin d'accès relatifs du fichier. Ensuite, l'extension démarre le programme de survol et lui transmet le nom du fichier. Si vous indiquez MMDB_PLAY_HANDLE, l'extension extrait le nom du fichier à partir du descripteur, à condition qu'il ne s'agisse pas de la valeur NULL. Si c'est le cas, l'objet est stocké en tant qu'objet BOLB. L'extension crée un fichier temporaire sur le poste client et copie le contenu de l'objet de la table dans le fichier client. L'extension démarre ensuite le programme et lui transmet le nom du fichier (ou du fichier temporaire) approprié.

Par exemple, les instructions suivantes d'un programme d'application en C permettent d'extraire le descripteur d'une image stockée sous forme d'objet BLOB et l'utilisent pour afficher l'image.

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_hdl [251];
  EXEC SQL END DECLARE SECTION;

rc = DBiBrowse(
     "ib %s",
     MMDB_PLAY_HANDLE,                  /* contenu sous forme d'objet BLOB */
     hvImg_hdl,
     MMDB_PLAY_NO_WAIT
     );
```

Le contenu doit être accessible : Vérifiez que le programme d'affichage ou de lecture peut accéder au contenu de l'objet. Si le contenu se trouve dans un fichier du serveur, mais que le programme nécessite la présence du contenu sur le poste client, copiez le fichier dans un fichier client ou utilisez la fonction UDF Content. Si le contenu est stocké sous forme d'objet BLOB, l'extension le récupère automatiquement et le place sur le poste client.

Spécification d'un indicateur d'attente

Le programme d'application peut éventuellement attendre que l'utilisateur mette fin au programme d'affichage ou de lecture pour se poursuivre (c'est-à-dire qu'il attend que l'API DBiBrowse, DBaPlay ou DBvPlay renvoie un code). Si vous souhaitez que le programme d'application attende, indiquez

MMDB_PLAY_WAIT. Dans le cas contraire, indiquez MMDB_PLAY_NO_WAIT. MMDB_PLAY_WAIT et MMDB_PLAY_NO_WAIT sont des constantes définies par les extensions.

Si vous indiquez MMDB_PLAY_WAIT, le programme d'affichage ou de lecture s'exécute sur la même unité d'exécution ou le même processus que le programme d'application. Si vous indiquez MMDB_PLAY_NO_WAIT, le programme d'affichage ou de lecture s'exécute sur sa propre unité d'exécution ou son propre processus, indépendamment du programme d'application.

Par exemple, lorsque vous précisez les instructions suivantes, le programme d'application attend que l'utilisateur ferme l'afficheur d'images pour poursuivre son exécution :

```
rc = DBiBrowse(
     "explore %s",
      MMDB_PLAY_FILE,
     "/Employés/images/adupont.bmp",
     MMDB_PLAY_WAIT                    /* attente fermeture afficheur */
     );
```

Attention si vous spécifiez DBxPlay et MMDB_PLAY_WAIT : Lorsque vous exécutez des instructions DBaPlay ou DBvPlay, l'extension crée un fichier temporaire pour l'objet à lire ou à afficher si :

- l'objet est stocké en tant qu'objet BLOB ;
- le nom relatif du fichier ne peut pas être résolu à l'aide de variables d'environnement ;
- le fichier n'est pas accessible sur le poste client.

Ce fichier est créé dans le répertoire désigné par la variable d'environnement TMP. Si vous indiquez MMDB_PLAY_WAIT, l'extension supprime le fichier temporaire après la lecture de l'objet. Cependant, si vous indiquez MMDB_PLAY_NO_WAIT, le fichier temporaire n'est pas supprimé. Vous devez alors effectuer cette suppression vous-même.

Affichage d'un objet image ou vidéo miniaturisé

Une miniature est une version réduite d'un objet image ou vidéo. Lorsque vous stockez une image dans la base de données, l'extension Image crée une miniature de l'image et la stocke dans une table d'attributs. Lorsque vous stockez un objet vidéo dans la base de données, l'extension Vidéo stocke, dans une table d'attributs, une miniature générique qui symbolise cet objet.

Par défaut, la taille d'une miniature créée automatiquement par l'extension Image est d'environ 112 x 84 pixels. La taille de la miniature vidéo générique créée par l'extension Vidéo est de 108 x 78 pixels. Les deux objets (miniature image et miniature vidéo générique) sont stockés au format GIF. Selon la

densité des données de l'objet image ou vidéo, cela correspond approximativement à 4,5 à 5 ko. Lorsque vous stockez ou mettez à jour un objet image ou vidéo ayant des attributs définis par l'utilisateur, vous pouvez spécifier une miniature de la taille et du format qui vous conviennent.

Utilisez la fonction UDF Thumbnail de l'instruction SQL SELECT pour extraire une miniature de la base de données, et une variable de référence à un fichier pour transmettre la miniature au fichier concerné. Lorsque vous indiquez la fonction UDF, vous devez préciser le nom de la colonne de la table de base de données contenant le descripteur de l'objet image ou vidéo. Utilisez ensuite l'API DBiBrowse pour afficher l'objet image ou vidéo miniaturisé.

Par exemple, les instructions suivantes permettent de récupérer une image miniaturisée et de l'afficher :

```
long rc, outCount;
char nomfich_miniature[254];
FILE *indic_fich;


EXEC SQL BEGIN DECLARE SECTION;
         struct  {
         short len
       char data[10000];
         }tampon_miniature;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT THUMBNAIL(PICTURE)
     INTO :tampon_miniature
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';

strcpy (nomfich_miniature,"/tmp/dupont.tmb");
indic_fich=fopen(nomfich_miniature,"wb+");
outCount=fwrite(tampon_miniature.data, 1, tampon_miniature.len,
indic_fich);
fclose(indic_fich);
rc = DBiBrowse(
               NULL,                          /* utiliser programme affichage par défaut */
               MMDB_PLAY_FILE,                /* image miniature dans fichier */
               nomfich_miniature              /* contenu image miniature */
               MMDB_PLAY_WAIT);               /* attente fin programme */
```

Affichage d'un objet image ou vidéo en grandeur réelle

Utilisez l'API DBiBrowse pour afficher une image stockée dans une table de base de données. Pour plus d'informations sur l'utilisation de cette API, reportez-vous à la section «Utilisation des API d'affichage ou de lecture» à la page 131.

Utilisez l'API DBvGetNextFrame ou DBvSeekFrame pour obtenir un objet vidéo en grandeur réelle. L'objet est stocké au format YUV dans une mémoire tampon et peut être converti au format RGB à l'aide de l'API DBvFrameDatato24BitRGB. Ajoutez un en-tête à l'objet converti (par exemple, un en-tête de type de fichier BMP) et spécifiez l'en-tête et les données de l'objet dans un fichier. Utilisez ensuite l'API DBiBrowse pour afficher le contenu du fichier. Pour plus d'informations sur l'utilisation des API DBvGetNextFrame, DBvSeekNextFrame et DBvFrameDatato24BitRGB et sur

l'affichage d'un objet vidéo, reportez-vous au «Chapitre 14. Détection de changements de plan vidéo» à la page 179.

Lecture d'un objet audio ou vidéo

Utilisez l'API DBaPlay pour lire un objet audio stocké dans une table de base de données, et l'API DBvPlay, pour afficher ce type d'objet. Pour plus d'informations sur l'utilisation de ces API, reportez-vous à la section «Utilisation des API d'affichage ou de lecture» à la page 131.

Chapitre 13. Recherche d'images par contenu

La figure 25 présente un programme d'application permettant de rechercher des images dans une base de données à l'aide d'un modèle visuel utilisé comme critère de recherche, c'est-à-dire une image d'une couleur ou d'une texture prédominante. Dans une telle application, la recherche s'effectue à partir d'une image choisie par l'utilisateur. L'application compare ensuite la couleur ou la texture de l'image source avec celle des images stockées, puis propose les images dont la couleur ou la texture est la plus proche.

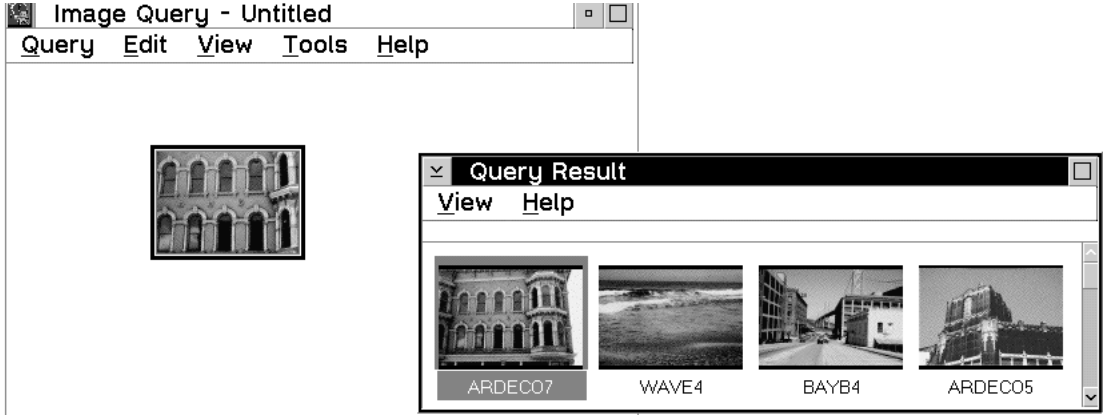


Figure 25. Recherche par contenu d'image. La couleur ou la texture d'un exemple visuel sert de base à la recherche d'images stockées dans une table de base de données.

Cette recherche d'images fondée sur leurs caractéristiques visuelles est appelée **Recherche par contenu d'image (QBIC)** [5]. Ce chapitre indique comment utiliser les API et les fonctions utilisateur (UDF) fournies par l'extension Image pour générer des applications de ce type. Il décrit également comment utiliser les commandes et les API fournies par l'extension Image pour effectuer des tâches administratives QBIC.

Recherche par contenu d'image

Pour effectuer une recherche par contenu d'image :

1. Créez un catalogue QBIC pour y stocker les images.

5. L'extension Image comporte des logiciels développés par l'Université de Californie, Berkeley, et ses collaborateurs.

2. Cataloguez les images. Cela signifie que vous devez ajouter des entrées pour les images dans le catalogue et stocker les valeurs relatives aux caractéristiques de l'image.

   Pour obtenir une description des catalogues QBIC et des caractéristiques d'images, reportez-vous à la section «Catalogues QBIC» à la page 22.
3. Constituez votre requête. Elle identifie les caractéristiques à utiliser en tant que critères de recherche, les valeurs et les poids (importance respective) qui leur sont associés. Vous pouvez spécifier ces attributs dans une chaîne de caractères appelée chaîne de requête. Vous pouvez également créer un objet de requête auquel seront associés ces attributs. Vous pouvez alors sauvegarder cette chaîne de requête et la réutiliser.
4. Exécutez la requête. Lors de cette opération, vous indiquez une chaîne de requête en entrée ou vous identifiez un objet de requête. Dans les deux cas, vous identifiez également les images à rechercher et vous pouvez lancer l'opération à partir de la ligne de commande DB2 ou d'un programme.

Suite à cela, l'extension Image calcule les valeurs des caractéristiques pour la requête. Elle compare ces valeurs à celles des caractéristiques stockées dans le catalogue QBIC pour les images cible. L'extension Image calcule ensuite un score qui définit le degré de correspondance des images retrouvées par rapport à l'échantillon source.

Vous pouvez demander à l'extension Image d'afficher les images dont les valeurs de caractéristique sont les plus proches de l'échantillon source, ou de renvoyer les scores d'une ou plusieurs images.

Gestion des catalogues QBIC

Pour effectuer une recherche d'images par contenu, ces dernières doivent être cataloguées dans un catalogue QBIC. Un catalogue QBIC contient les données relatives aux caractéristiques visuelles des images.

Un catalogue QBIC doit être créé pour chaque colonne d'images d'une table utilisateur sur laquelle vous souhaitez pouvoir effectuer des recherches par contenu. Il ne peut exister qu'un catalogue QBIC par colonne d'images d'une table utilisateur ; en outre, plusieurs colonnes ne peuvent pas partager le même catalogue QBIC.

Lors de la création du catalogue, vous identifiez les caractéristiques que l'extension Image doit stocker. Vous indiquez également si vous souhaitez qu'elle catalogue automatiquement une image. Le catalogage automatique signifie que l'extension Image crée automatiquement des entrées pour une image dans le catalogue lors du stockage de cette image dans une table utilisateur. Une image non cataloguée automatiquement doit l'être

manuellement. Pour cela, vous devez demander explicitement à l'extension Image de créer des entrées relatives à l'image.

Après avoir créé un catalogue QBIC, vous pouvez :

- ouvrir le catalogue pour y effectuer d'autres opérations ;
- passer du catalogage automatique au catalogage manuel, ou inversement ;
- identifier, en les ajoutant dans le catalogue, les caractéristiques pour lesquelles vous souhaitez que l'extension Image stocke les données ;
- supprimer des caractéristiques du catalogue ;
- extraire des informations relatives au catalogue, telles que le nom de la table utilisateur ou de la colonne associée au catalogue, ou les caractéristiques stockées dans ce dernier ;
- cataloguer manuellement des images dans le catalogue ;
- décataloguer une image (supprimer les entrées correspondantes du catalogue) ;
- recataloguer des images ;
- redistribuer le catalogue (lors de l'ajout ou de la suppression de noeuds dans un système de bases de données partitionnées) ;
- fermer le catalogue ;
- supprimer le catalogue.

Vous pouvez effectuer ces tâches, y compris la création d'un catalogue QBIC, à l'aide des API fournies par l'extension Image ou du processeur de commandes db2ext.

Création d'un catalogue QBIC

Utilisez l'API QbCreateCatalog ou la commande CREATE QBIC CATALOG pour créer un catalogue QBIC. Pour ce faire, vous devez être le propriétaire de la table utilisateur dont les images sont cataloguées. De plus, vous devez avoir le droit CREATE TABLE sur la base de données qui contient le catalogue. La table utilisateur et la colonne image doivent être activées pour l'extension Image afin qu'un catalogue QBIC portant sur les images de cette colonnes puisse être créé.

Lorsque vous créez un catalogue QBIC, vous devez :

- Nommer la table utilisateur et la colonne contenant les images à cataloguer.
- Indiquer si les images seront cataloguées automatiquement. Cela signifie que l'extension Image catalogue une image lors du stockage de cette dernière dans une table utilisateur. Elle vérifie régulièrement si une image est en attente de catalogage. Vous spécifiez la périodicité de cette vérification (en secondes), en définissant pour la variable d'environnement DB2CATALOGDELAY une valeur de 1 seconde ou davantage. La valeur par défaut est de 60 secondes.

Le catalogage manuel consiste à demander explicitement à l'extension Image de cataloguer une image. (Pour plus d'informations, reportez-vous à la section «Catalogage manuel d'une image» à la page 146.)

Activation de la table utilisateur et de la colonne : La table utilisateur et la colonne image doivent être accessibles pour l'extension Image pour qu'un catalogue QBIC puisse être créé pour les images de cette colonne. (Pour plus d'informations sur l'activation des tables utilisateur et des colonnes, reportez-vous au «Chapitre 6. Préparation des objets de données pour les données d'extensions» à la page 57.)

Utilisation de l'interface API : Lors de l'utilisation de l'API QbCreateCatalog, vous indiquez un catalogage automatique ou manuel par la valeur du paramètre auto-catalog. La valeur 1 correspond au catalogage automatique et la valeur 0 au catalogage manuel.

Par exemple, les instructions ci-après créent un catalogue QBIC pour les images de la colonne photo de la table Employés. Les images seront automatiquement cataloguées lors de leur stockage dans la table Employés :

```
SQLINTEGER autoCatalog=1;                          /* catalogage automatique */

rc=QbCreateCatalog(
                 "Employés",                      /* table utilisateur */
                 "photo",                         /* colonne image */
                  autoCatalog);                    /* paramètre auto-catalog */
```

Utilisation de la ligne de commande : Lorsque vous lancez la commande CREATE QBIC CATALOG, vous demandez le catalogage automatique à l'aide de la valeur ON, ou le catalogage manuel à l'aide de la valeur OFF. Celle-ci est la valeur par défaut.

Par exemple, la commande ci-dessous crée le même catalogue QBIC que dans l'exemple relatif à l'API :

```
CREATE QBIC CATALOG Employés Photo on
```

Sauvegarde du catalogue QBIC : L'extension Image stocke un catalogue QBIC dans des fichiers. Nous vous recommandons de sauvegarder régulièrement ces fichiers pour pouvoir éventuellement récupérer le catalogue. Sur un serveur AIX, HP-UX ou Sun Solaris, les fichiers se trouvent dans le répertoire /home/*propriétaire_instance*/dmb/qbic, où *propriétaire_instance* est l'ID utilisateur du propriétaire de l'instance. Sur un serveur OS/2 ou Windows, les fichiers se trouvent dans le répertoire *destination*\instance*nom_instance*\qbic, où *destination* est le répertoire dans lequel est installée l'extension Image et *nom_instance*, le nom de l'instance de l'extension.

Ouverture d'un catalogue QBIC

Vous devez ouvrir un catalogue QBIC pour pouvoir y effectuer des modifications, par exemple, ajouter une caractéristique.

Pour ouvrir un catalogue QBIC, utilisez l'appel d'API QbOpenCatalog ou la commande OPEN QBIC CATALOG. Lors de l'ouverture d'un catalogue QBIC, vous devez :

- Nommer la table utilisateur et la colonne image du catalogue.
- Indiquer le mode d'ouverture du catalogue (cette indication est implicite avec la commande OPEN QBIC CATALOG). Vous pouvez ouvrir un catalogue afin d'y effectuer des opérations de lecture, comme la recherche d'images par contenu, ou des opérations le mettant à jour comme l'ajout d'une caractéristique. Vous devez disposer du droit SELECT pour que la table utilisateur puisse ouvrir le catalogue en lecture. Vous devez disposer du droit UPDATE pour que la table utilisateur puisse ouvrir le catalogue en mise à jour.

Ouverture préalable d'un catalogue : Vous ne pouvez pas ouvrir un catalogue en mise à jour s'il est déjà ouvert dans le même but dans une autre session. Lorsque vous ouvrez un catalogue QBIC, l'extension Image ferme les catalogues QBIC ouverts dans la session en cours.

Utilisation de l'interface API : Lors de l'utilisation de l'API QbOpenCatalog, vous indiquez explicitement le mode d'ouverture du catalogue. Spécifiez :

- Le paramètre API qbiRead pour ouvrir le catalogue en lecture.
- Le paramètre API qbiUpdate pour ouvrir le catalogue en mise à jour.

QbiRead et QbiUpdate sont des constantes définies dans le fichier (en-tête) d'inclusion pour QBIC, dmbqbapi.h.

Vous devez également pointer sur le descripteur du catalogue. Ce descripteur comporte un type de données propre à QBIC, QbCatalogHandle. Ce type de données est également défini dans dmbqbapi.h. L'extension Image renvoie la valeur du descripteur du catalogue en tant que sortie de l'API.

Par exemple, l'appel d'API ci-dessous ouvre un catalogue QBIC en lecture :

```
mode SQLINTEGER;
QbCatalogHandle  *CatHdl;

mode=qbiRead;                                   /* ouverture du catalogue */
                                                /* opérations de lecture */

rc=QbOpenCatalog(
                 "Employés",                    /* table utilisateur */
                 "photo",                       /* colonne image */
                 mode,                          /* mode d'ouverture du catalogue */
                  &CatHdl);                     /* descripteur du catalogue */
```

Utilisation de la ligne de commande : Lorsque vous lancez la commande OPEN QBIC CATALOG, l'extension Image tente d'ouvrir le catalogue en mise à jour. Si celui-ci est en cours de mise à jour dans une autre session, l'extension Image l'ouvre en lecture.

Par exemple, la commande ci-dessous ouvre un catalogue QBIC ; l'extension Image tente de l'ouvrir en mise à jour :

```
OPEN QBIC CATALOG Employés photo
```

Fermeture du catalogue après les opérations QBIC : Lors de l'ouverture d'un catalogue QBIC, l'extension Image lui affecte des ressources telles que de la mémoire. Fermez le catalogue lorsque vous avez terminé les activités relatives à QBIC, ce qui permet de libérer les ressources allouées.

Modification du paramètre Auto Catalog

Utilisez l'API QbSetAutoCatalog ou la commande SET QBIC AUTOCATALOG pour passer du catalogage automatique au catalogage manuel, et inversement. Pour ce faire, vous devez ouvrir le catalogue QBIC.

Modification non rétroactive : Lorsque vous modifiez le paramètre auto-catalog, cela ne concerne que les images ajoutées à la colonne de la table utilisateur après la modification, et non les images déjà stockées dans cette colonne. Par exemple, si vous passez du catalogage manuel au catalogage automatique, seules les images ajoutées à la colonne de la table utilisateur après cette modification seront cataloguées automatiquement. Si vous souhaitez cataloguer des images déjà stockées dans la colonne de la table, vous devez le faire manuellement. (Pour plus d'informations, reportez-vous à la section «Catalogage manuel d'une image» à la page 146.)

Utilisation de l'interface API : Lors de l'utilisation de l'API QbSetAutoCatalog, spécifiez le descripteur du catalogue QBIC (ce descripteur est renvoyé lorsque vous ouvrez le catalogue à l'aide de l'API QbOpenCatalog). Indiquez également pour le paramètre auto-catalog la valeur 1 pour un catalogage automatique ou la valeur 0 pour un catalogage manuel.

Dans l'exemple suivant, le catalogage manuel est indiqué pour un catalogue QBIC associé aux images de la colonne photo de la table Employés. Notez que le catalogue QBIC est d'abord ouvert en mise à jour.

```
mode SQLINTEGER;
SQLINTEGER autoCatalog=0;                /* catalogage manuel */

QbCatalogHandle  *CatHdl;

mode=qbiUpdate;                          /* ouverture du catalogue */
                                         /* en mise à jour */
/* Ouverture d'un catalogue QBIC */
```

```
rc=QbOpenCatalog(
                 "Employés",             /* table utilisateur */
                 "photo",                /* colonne image */
                 mode,                   /* mode d'ouverture du catalogue */
                  &CatHdl);              /* descripteur du catalogue */

/* Modification du paramètre auto catalog */
rc=QbSetAutoCatalog(
                CatHdl,                      /* descripteur du catalogue */
                  autoCatalog);          /* indicateur auto catalog */
```

Utilisation de la ligne de commande : Lorsque vous lancez la commande SET QBIC AUTOCATALOG, vous demandez le catalogage automatique à l'aide de la valeur ON, ou le catalogage manuel à l'aide de la valeur OFF. Cette commande affecte le catalogue ouvert.

Par exemple, la commande ci-dessous désactive le catalogage automatique pour le catalogue QBIC ouvert :

```
SET QBIC AUTOCATALOG off
```

Ajout d'une caractéristique à un catalogue QBIC

Utilisez l'API QbAddFeature ou la commande ADD QBIC FEATURE pour ajouter une caractéristique à un catalogue QBIC. Vous devez ajouter au moins une caractéristique à un catalogue QBIC avant de pouvoir y cataloguer une image. Le catalogue QBIC doit être ouvert en mise à jour avant cette opération.

Lorsque vous ajoutez une caractéristique à un catalogue, indiquez-en le nom (les noms de caractéristiques sont répertoriés dans le tableau 8).

Tableau 8. Noms des caractéristiques QBIC

Nom de la caractéristique	Description
QbColorFeatureClass	Couleur moyenne
QbColorHistogramFeatureClass	Couleur d'histogramme
QbDrawFeatureClass	Couleur positionnelle
QbTextureFeatureClass	Texture

Recatalogage d'images : Si vous ajoutez une caractéristique à un catalogue QBIC, l'extension Image ne stocke pas automatiquement les données correspondantes pour les images déjà cataloguées, même si le catalogage automatique est activé. Afin d'inclure ces données pour les images déjà cataloguées, vous devez recataloguer ces dernières (reportez-vous à la section «Recatalogage d'images» à la page 148).

Utilisation de l'interface API : Lors de l'utilisation de l'API QbAddFeature, spécifiez le descripteur du catalogue QBIC en plus du nom de la caractéristique. Notez que la constante qbiMaxFeatureName est utilisée pour

définir la longueur du nom de la caractéristique. Cette constante est définie par 50 dans le fichier (en-tête) d'inclusion pour QBIC, dmbqbapi.h.

Dans l'exemple suivant, l'API QbAddFeature est utilisée pour ajouter la caractéristique couleur d'histogramme à un catalogue QBIC :

```
char  featureName[qbiMaxFeatureName];

QbCatalogHandle   CatHdl;

strcpy(featureName,"QbColorHistogramFeatureClass");

rc=QbAddFeature(
                CatHdl,                 /* descripteur du catalogue */
                featureName);             /* nom de la caractéristique */
```

Utilisation de la ligne de commande : La commande ADD QBIC FEATURE affecte le catalogue ouvert. Dans l'exemple suivant, elle sert à ajouter la caractéristique couleur positionnelle au catalogue ouvert :

```
ADD QBIC FEATURE QbDrawFeatureClass
```

Suppression d'une caractéristique d'un catalogue QBIC

Utilisez l'API QbRemoveFeature ou la commande REMOVE QBIC FEATURE pour supprimer une caractéristique d'un catalogue QBIC. L'extension Image supprime la table du catalogue correspondant à cette caractéristique. Par conséquent, lorsque vous cataloguez une image, les données relatives à cette caractéristique ne sont pas stockées. Le catalogue QBIC doit être ouvert en mise à jour avant la suppression d'une caractéristique.

Lorsque vous supprimez une caractéristique d'un catalogue, indiquez-en le nom.

Utilisation de l'interface API : Lors de l'utilisation de l'API QbRemoveFeature, spécifiez le descripteur du catalogue QBIC en plus du nom de la caractéristique.

Dans l'exemple suivant, l'API QbRemoveFeature est utilisée pour supprimer la caractéristique couleur d'histogramme d'un catalogue QBIC :

```
char  featureName[qbiMaxFeatureName];

QbCatalogHandle   CatHdl;

strcpy(featureName,"QbColorHistogramFeatureClass");

rc=QbRemoveFeature(
               CatHdl,                /* descripteur du catalogue */
               featureName);           /* nom de la caractéristique */
```

Utilisation de la ligne de commande : La commande REMOVE QBIC FEATURE affecte le catalogue ouvert. Dans l'exemple suivant, elle sert à supprimer la caractéristique couleur positionnelle du catalogue ouvert :

```
REMOVE QBIC FEATURE QbDrawFeatureClass
```

Extraction d'informations sur un catalogue QBIC

Vous pouvez extraire les informations suivantes sur un catalogue QBIC :

- Le nom de la table utilisateur et de la colonne image associées au catalogue.
- Le nombre de caractéristiques pour lesquelles des données sont stockées dans le catalogue et leurs noms.
- Si l'extension Image catalogue automatiquement les images lorsqu'elles sont stockées dans la table utilisateur.

Utilisez l'API QbGetCatalogInfo pour extraire le nom de la table utilisateur et de la colonne, le nombre de caractéristiques et le paramètre auto-catalog. Utilisez l'API QbListFeatures pour extraire les noms de caractéristiques. Vous pouvez également utiliser la commande GET QBIC CATALOG INFO pour extraire toutes ces informations.

Le catalogue QBIC doit être ouvert en mise à jour avant l'extraction d'informations.

Utilisation de l'interface API : Lors de l'utilisation de l'API QbGetCatalogInfo, spécifiez le descripteur du catalogue QBIC. Vous devez également pointer sur une structure dans laquelle l'extension Image renvoie les informations relatives au catalogue. La structure des informations relatives au catalogue est définie dans le fichier (en-tête) d'inclusion pour QBIC, dmbqapi.h, comme suit :

```
typedef struct{
          char       tableName[qbiMaxTableName+1]    /* table utilisateur */
          char       columnName[qbiMaxColumnName+1]  /* colonne image */
          SQLINTEGER featureCount;                   /* nombre de caractéristiques */
          SQLINTEGER autoCatalog;                    /* indicateur auto-catalog */
} QbCatalogInfo;
```

Lorsque vous lancez l'appel d'API QbListFeatures, vous devez affecter une mémoire tampon afin d'y placer les noms de caractéristiques renvoyés qui y sont séparés par un espace. Vous devez également spécifier le descripteur du catalogue et la taille de la mémoire tampon recevant les noms de caractéristiques renvoyés. Pour estimer la taille de la mémoire tampon nécessaire, multipliez le nombre de caractéristiques renvoyé par l'API QbGetCatalogInfo par le nom de caractéristique le plus long. Vous pouvez utiliser la constante qbiMaxFeatureName comme taille du nom de caractéristique le plus long.

Les appels d'API de l'exemple suivant extraient des informations relatives au catalogue QBIC. Notez l'utilisation du nombre de caractéristiques renvoyé par l'API QbGetCatalogInfo et la constante qbiMaxFeature pour le calcul de taille de la mémoire tampon pour l'API QbListFeatures :

```
long  bufSize;
long  count;
char  *featureNames;

QbCatalogHandle   CatHdl;
QbCatalogInfo    catInfo;

/* Extraction du nom de la table utilisateur, de la colonne image, du nombre */
/* et du paramètre auto_catalog */

rc=QbGetCatalogInfo(
                CatHdl,                              /* descripteur du catalogue */
                &catInfo);                           /* structure info. catalogue */

/* Liste des noms de caractéristique */

bufSize=catInfo.featureCount*qbiMaxFeatureName;
featureNames=malloc(bufSize);

rc=QbListFeatures(
                CatHdl,                      /* descripteur du catalogue */
                bufSize                     /* taille de la mémoire tampon */
                count,                       /* nombre de caractéristiques */
                featureNames);                  /* mémoire tampon pour les noms */
                                                /* de caractéristiques */
```

Utilisation de la ligne de commande : La commande GET QBIC CATALOG INFO affecte le catalogue ouvert. Dans l'exemple suivant, elle extrait des informations relatives au catalogue QBIC ouvert :

```
GET QBIC CATALOG INFO
```

Catalogage manuel d'une image

Lorsque vous créez un catalogue, vous indiquez si vous souhaitez que l'extension Image catalogue automatiquement une image lors de son stockage dans une table utilisateur. A défaut, vous devez la cataloguer manuellement après son stockage dans la table utilisateur. Vous pouvez cataloguer manuellement une image ou une colonne d'images complète.

Catalogage manuel d'une image

Utilisez l'API QbCatalogImage pour cataloguer manuellement une image. Vous ne pouvez pas cataloguer une image par une commande, car il est impossible d'identifier une image sur la ligne de commande. Lorsque vous utilisez l'API, indiquez le descripteur du catalogue et celui de l'image (vous pouvez extraire ce dernier de la table utilisateur). Le catalogue QBIC doit être ouvert en mise à jour avant le catalogage manuel d'une image.

Par exemple, les instructions ci-dessous extraient le descripteur d'une image dans une table utilisateur, puis cataloguent l'image :

```
/* Extraction du descripteur de l'image */


EXEC SQL BEGIN DECLARE SECTION;
char Img_hdl[251];
  EXEC SQL END DECLARE SECTION;

QbCatalogHandle   CatHdl;

EXEC SQL SELECT PICTURE INTO :Img_hdl
       FROM EMPLOYEE
```

```
       WHERE NAME='Anne Dupont';

/* Catalogage de l'image*/

rc=QbCatalogImage(
                CatHdl,                     /* descripteur du catalogue */
                Img_hdl);                  /* descripteur de l'image */
```

Catalogage manuel d'une colonne d'images

Utilisez l'API QbCatalogColumn ou la commande CATALOG QBIC COLUMN pour cataloguer manuellement une colonne d'images. L'extension Image catalogue uniquement les images ajoutées, mises à jour ou supprimées depuis le dernier catalogage de la colonne. Elle les catalogue pour toutes les caractéristiques du catalogue. Le catalogue QBIC doit être ouvert en mise à jour avant le catalogage manuel d'une colonne d'images.

Utilisation de l'interface API : Lors de l'utilisation de l'API QbCatalogColumn, précisez le descripteur du catalogue. L'extension Image utilise les images de la colonne de la table utilisateur associée au catalogue spécifié.

Par exemple, l'appel d'API suivant catalogue les images non cataloguées d'une colonne d'une table utilisateur associée au catalogue indiqué. Les images sont cataloguées pour toutes les caractéristiques du catalogue :

```
QbCatalogHandle   CatHdl;

rc=QbCatalogColumn(
                CatHdl);                    /* descripteur du catalogue */
```

Utilisation de la ligne de commande : Lancez la commande CATALOG QBIC COLUMN pour cataloguer manuellement une colonne d'images. Vous pouvez également l'utiliser pour recataloguer des images (reportez-vous à la section «Recatalogage d'images» à la page 148). Spécifiez les paramètres FOR et NEW (qui sont les paramètres par défaut).

Dans l'exemple ci-après, cette commande sert à cataloguer les images décataloguées de la colonne de la table associée au catalogue ouvert. Les images sont cataloguées pour toutes les caractéristiques du catalogue :

```
CATALOG QBIC COLUMN FOR NEW
```

Décatalogage d'une image

Décataloguer une image revient à supprimer les entrées correspondantes dans le catalogue QBIC. Utilisez l'API QbUncatalogImage pour décataloguer une image. Vous ne pouvez pas décataloguer une image par une commande, car il est impossible d'identifier une image sur la ligne de commande. Lorsque vous utilisez l'API, indiquez le descripteur du catalogue et celui de l'image (vous pouvez extraire ce dernier de la table utilisateur). Le catalogue QBIC doit être ouvert en mise à jour avant le décatalogage d'une image.

Par exemple, les instructions suivantes extraient le descripteur d'une image d'une table utilisateur, puis décataloguent cette dernière :

```
/* Extraction du descripteur de l'image */


EXEC SQL BEGIN DECLARE SECTION;
char Img_hdl[251];
  EXEC SQL END DECLARE SECTION;

QbCatalogHandle   CatHdl;

EXEC SQL SELECT PICTURE INTO :Img_hdl
       FROM EMPLOYEE
       WHERE NAME='Anne Dupont';

/* Décatalogage de l'image */

rc=QbUncatalogImage(
               CatHdl,                      /* descripteur du catalogue */
               Img_hdl);                        /* descripteur de l'image */
```

Recatalogage d'images

Lorsque vous cataloguez une image, l'extension Image analyse ses caractéristiques qui ont été identifiées à l'intention du catalogue QBIC et stocke dans ce dernier les valeurs qui leur ont été associées. Lorsque vous ajoutez une caractéristique à un catalogue QBIC, l'extension Image n'analyse pas automatiquement cette dernière pour les images déjà cataloguées. Pour ajouter au catalogue des valeurs pour la nouvelle caractéristique, vous devez recataloguer toutes les images.

Utilisez l'API QbReCatalogColumn ou la commande CATALOG QBIC COLUMN pour recataloguer les images d'un catalogue QBIC. L'extension Image supprime alors toutes les données relatives aux caractéristiques présentes dans le catalogue, puis analyse les images pour toutes les caractéristiques (y compris les nouvelles) et les catalogue. Le catalogue QBIC doit être ouvert pour que les images puissent être recataloguées.

Utilisation de l'interface API : Lors de l'utilisation de l'API QbReCatalogColumn, précisez le descripteur du catalogue.

Dans l'exemple suivant, les images d'un catalogue QBIC sont réanalysées :

```
QbCatalogHandle   CatHdl;

rc=QbReCatalogColumn(
               CatHdl);              /* descripteur du catalogue */
```

Utilisation de la ligne de commande : Lancez la commande CATALOG QBIC COLUMN pour recataloguer des images. Cette commande affecte le catalogue

ouvert. Vous pouvez également l'utiliser pour cataloguer des images manuellement (reportez-vous à la section «Catalogage manuel d'une image» à la page 146).

Lorsque vous lancez cette commande, indiquez les paramètres FOR et ALL. Cela indique à l'extension Image que vous souhaitez recataloguer la totalité des images.

Dans l'exemple ci-dessous, les images cataloguées dans le catalogue QBIC ouvert sont recataloguées :

```
CATALOG QBIC COLUMN FOR ALL
```

Redistribution d'un catalogue QBIC (Produit EEE uniquement)

Utilisez l'API DMBRedistribute ou la commande REDISTRIBUTE NODEGROUP pour redistribuer les données présentant des caractéristiques QBIC lors de l'ajout ou de la suppression d'un noeud dans un groupe de noeuds. La commande place les données présentant des caractéristiques QBIC sur le même noeud que celui des données utilisateur correspondantes.

Si le processus de redistribution renvoie une erreur, vous pouvez relancer la commande avec ou sans le paramètre CONTINUE, selon les instructions fournies par la réponse de la commande. Cette option permet au système de reprendre le traitement là où il s'était arrêté plutôt que de tout recommencer dès le début. Le paramètre CONTINUE ne peut pas être utilisé lors du premier lancement de la commande REDISTRIBUTE NODEGROUP après l'exécution de la commande DB2 REDISTRIBUTE NODEGROUP.

Pour conserver l'intégrité des données, procédez à la redistribution d'un groupe de noeuds à la fois. Attendez qu'un groupe de noeuds ait terminé la redistribution avant d'en démarrer un autre.

Utilisation des API : L'exemple suivant illustre comment redistribuer des données présentant des caractéristiques QBIC dans un groupe de noeuds appelé *groupe1* :

```
#include <dmbrdst.h>

rc = DMBRedistribute(groupone,"continue");
```

Utilisation de la ligne de commande : L'exemple suivant illustre comment utiliser la commande REDISTRIBUTE NODEGROUP pour redistribuer les données pour le noeud appelé *mon_groupe_noeuds* à l'aide du paramètre CONTINUE :

```
redistribute nodegroup mon_groupe_noeuds continue
```

Fermeture d'un catalogue QBIC

Utilisez l'API QbCloseCatalog ou la commande CLOSE QBIC CATALOG pour fermer un catalogue QBIC. Le catalogue doit être ouvert pour pouvoir être fermé.

Utilisation de l'interface API : Lors de l'utilisation de l'API QbCloseCatalog, précisez le descripteur du catalogue. Par exemple :

```
QbCatalogHandle   CatHdl;

rc=QbCloseCatalog(
                CatHdl);            /* descripteur du catalogue */
```

Utilisation de la ligne de commande : La commande CLOSE QBIC CATALOG affecte le catalogue ouvert. Dans l'exemple suivant, elle sert à fermer le catalogue QBIC ouvert :

```
CLOSE QBIC CATALOG
```

Suppression d'un catalogue QBIC

La suppression d'un catalogue QBIC entraîne la suppression de toutes les données correspondant aux caractéristiques de ses tables. Par conséquent, les images associées ne peuvent plus être utilisées dans le cadre d'une recherche par contenu. Pour supprimer un catalogue QBIC, vous devez disposer des droits ALTER ou CONTROL pour la table associée au catalogue. Le catalogue doit être ouvert pour pouvoir être supprimé.

Utilisez l'API QbDeleteCatalog ou la commande DELETE QBIC CATALOG pour supprimer un catalogue QBIC. Lorsque vous supprimez un catalogue QBIC, indiquez le nom de la table utilisateur et de la colonne qui lui sont associées.

Utilisation de l'interface API : Dans l'exemple ci-dessous, l'API QbDeleteCatalog est utilisée pour supprimer un catalogue QBIC :

```
rc=QbDeleteCatalog(
                 "Employés",                    /* table utilisateur */
                  "photo");                      /* colonne image */
```

Utilisation de la ligne de commande : La commande DELETE QBIC CATALOG affecte le catalogue ouvert. Dans l'exemple suivant, elle sert à supprimer le catalogue QBIC ouvert :

```
DELETE QBIC CATALOG Employés photo
```

Modèle de programme créant un catalogue QBIC

La figure 26 à la page 152 présente un extrait d'un programme codé en langage C qui crée un catalogue QBIC. Ce programme catalogue également une colonne d'images dans ce catalogue. Vous trouverez le programme complet dans le fichier QBCATDMO.C du sous-répertoire SAMPLES. Avant d'exécuter le programme complet, vous devez exécuter les modèles de programmes ENABLE et POPULATE (qui se trouvent également dans le sous-répertoire SAMPLES). Pour plus d'informations sur les modèles de programmes, reportez-vous à l'«Annexe B. Modèles de programmes et fichiers de support» à la page 595.

La structure de cet extrait est la suivante :

1 Inclusion du fichier en-tête dmbqbapi.

2 Connexion à la base de données.

3 Création du catalogue. Il est créé lorsque la fonction de catalogage automatique est désactivée.

4 Ouverture du catalogue en mise à jour.

5 Ajout de la caractéristique couleur moyenne au catalogue.

6 Catalogage d'une colonne d'images.

7 Fermeture du catalogue.

```
#include <sql.h>
#include <sqlcli.h>
#include <sqlcli1.h>
#include <dmbqbqpi.h>  1
#include <stdio.h>

/*****************************************************************************/
/*  Définition des prototypes de fonctions */
/*****************************************************************************/

void printError(SQLHSTMT hstmt);
void createCatalog();
void openCatalog();
void closeCatalog();
void addFeature();
void catalogImageColumn();

QbCatalogHandle cHdl = 0;

static SQLHENV henv;
static SQLHDBC hdbc;
static SQLHSTMT hstmt;
static SQLRETURN rc;
char tableName[] = "sobay_catalog";
char columnName[] = "covers";

SQLCHAR uid[18+1];
SQLCHAR pwd[30+1];
SQLCHAR dbnName[SQL_MAX_DSN_LENGTH+1];

void main ()
{
/*---- invite à entrer le nom de la base de données, ----*/
       /*---- l'ID utilisateur et le mot de passe           ----*/
  printf("Enter database name:\n");
  gets((char *) dbName);
  printf("Enter userid:\n");
  gets((char *) pwd);
/* définition de l'environnement CLI SQL */
 SQlAllocEnv(&henv);
 SQLAllocConnect(henv, &hdbc);
 rc = SQLConnect(hdbc,dbname,SQL_NTS,uid,SQL_NTS,pwd,SQL_NTS);  2
 if (rc != SQL_SUCCESS)
 {
  printError(SQL_NULL_HSTMT);
          exit(1);
 }
```

Figure 26. Modèle de programme créant un catalogue QBIC (Numéro 1 de 4)

```
 createCatalog();
openCatalog();
addFeature();
getCatalogInfo();
 listFeatures();
 catalogImageColumn();
 closeCatalog();

 SQLDisconnect(hdbc);
 SQLFreeConnect(hdbc);
 SQLFreeEnv(henv);
}
/*********************************************************************/
void createCatalog()
{
 SQLINTEGER autoCatalog = 0;
 SQLINTEGER retLen;
   SQLINTEGER errCode = 0;
   char errMsg[500];

   QbCreateCatalog(  3
         (char *) tableName,
         (char *) columnName,
         autoCatalog,
0
        );

     DBiGetError(&errCode, errMsg);
     if(errCode) printf("Code d'erreur %d Message d'erreur %s", errCode, errMsg);
}
/*********************************************************************/
void openCatalog()
{
   SQLINTEGER errCode = 0;
   char errMsg[500];
 SQLINTEGER mode = qbiUpdate;

   QbOpenCatalog(  4
         (char *) tableName,
         (char *) columnName,
         mode,
         &cHdl
        );

     DBiGetError(&errCode, errMsg);
     if(errCode) printf("Code d'erreur %d Message d'erreur %s", errCode, errMsg);
}
```

Figure 26. Modèle de programme créant un catalogue QBIC (Numéro 2 de 4)

```
/*********************************************************************/
void addFeature()
{
 SQLINTEGER errCode=0;
 char errMsg[5
 if(cHdl) /* si un catalogue est ouvert, "else" n'a aucun effet */
 {
   char featureName*lbrk.] = "QbColorFeatureClass"; 5
     QbaddFeature(
         cHdl,
         featureName
        );

     DBiGetError(&errCode, errMsg);
     if(errCode) printf("Code d'erreur %d Message d'erreur %s", errCode, errMsg);
 }
   else
 {
          exit(1);
 }
}
/*********************************************************************/
void catalogImageColumn()
{
   SQLINTEGER errCode = 0;
   char errMsg[500];

 if(cHdl) /* si un catalogue est ouvert, "else" n'a aucun effet */
 {
  SQLRETURN rc;
    QbCatalogColumn(  6
         cHdl,
        );

     DBiGetError(&errCode, errMsg);
     if(errCode) printf("Code d'erreur %d Message d'erreur %s", errCode, errMsg);

   else
 {
          exit(1);
 }
}
```

Figure 26. Modèle de programme créant un catalogue QBIC (Numéro 3 de 4)

```
/*********************************************************************/
void closeCatalog()
{
 if(cHdl) /* si un catalogue est ouvert, "else" n'a aucun effet */
 {
    QbCloseCatalog( 7
         cHdl,
        );
 }
}
/*********************************************************************/
```

Figure 26. Modèle de programme créant un catalogue QBIC (Numéro 4 de 4)

Constitution de requêtes

Lorsque vous effectuez une recherche par contenu d'images, vous précisez les données d'entrée de la requête, ainsi qu'un échantillon d'images cataloguées cible. Les données d'entrée de la requête sont constituées du nom des caractéristiques à utiliser dans la requête, des valeurs et des poids (importance respective) qui sont associés à celles-ci.

Deux méthodes permettent de fournir les données d'entrée :

- Définition d'une chaîne de requête dans la requête. Il s'agit d'une chaîne de caractères précisant les caractéristiques, leurs valeurs et leurs poids pour la requête.
- Création d'un objet de requête et y faire référence dans la requête. L'objet de requête précise les caractéristiques et leur poids. Il identifie également une source de données pour chacune d'entre elles, qui fournit la valeur correspondant à chaque caractéristique.

Définition d'une chaîne de requête

Vous pouvez utiliser une chaîne de requête pour identifier les caractéristiques, leurs valeurs et leur poids pour votre requête. Une **chaîne de requête** est une chaîne de caractères se présentant sous la forme *nom_caractéristique valeur*, où *nom_caractéristique* est un nom de caractéristique QBIC et *valeur* une valeur associée à celle-ci.

Vous pouvez indiquer plusieurs caractéristiques dans une requête. Vous précisez ensuite un couple nom/valeur pour chacune d'entre elles, comme décrit à la section «Valeur de caractéristique» à la page 156. Les couples successifs sont séparés par une clause AND. Lorsque vous spécifiez de nombreuses caractéristiques dans une requête, vous pouvez également affecter un poids à une ou plusieurs d'entre elles, comme décrit à la section «Poids de caractéristique» à la page 158. La chaîne de requête se présente alors sous la forme *nom_caractéristique valeur poids*, où *poids* correspond au poids affecté à la caractéristique.

L'extension Image comprend une interface API (QbQueryStringSearch) et deux fonctions utilisateur (QbScoreFromStr et QbScoreTBFromStr) utilisant une chaîne de requête. Lorsque vous lancez une requête, vous devez utiliser l'API ou la fonction utilisateur appropriée, et spécifier la chaîne de requête en tant que paramètre d'entrée. (Pour plus de détails, reportez-vous à la section «Lancement de requêtes par contenu d'images» à la page 167.)

Valeur de caractéristique

Dans la chaîne de requête, précisez une valeur pour chacune des caractéristiques contenues dans la requête.

Lors du passage d'une requête à l'intérieur d'une commande DB2, il vous faut respecter certaines conventions d'appellation de fichier pour le bon fonctionnement de la requête. Les noms de fichier contenant des espaces ou le signe (> doivent être indiqués entre guillemets. Les guillemets sont facultatifs pour les autres noms de fichier. Chaque guillemet encadrant un nom de fichier doit être précédé d'un caractère d'échappement (\). Si la requête n'est pas transmise à l'intérieur d'une commande DB2, il n'est pas nécessaire d'inclure des caractères d'échappement avec les guillemets.

Dans l'exemple suivant, une chaîne de requête est transmise à l'aide d'une commande DB2 :

```
db2 "sélectionner id_image dans la table
(mmdbsys.QbScoreTBFromStr
('texture file=<server,patterns/ptrn07.gif>',
'fabric',
'swatch_img',
10))
as T1"
```

Le tableau 9 à la page 157 répertorie les valeurs autorisées pour chaque caractéristique. Sous chaque nom de caractéristique, vous trouverez une version courte pouvant être utilisée.

Tableau 9. Valeurs de caractéristiques autorisées dans la chaîne de requête

Nom de la caractéristique	Valeur
averageColor, average ou QbColorFeatureClass	color=<*val_rouge*, *val_vert*, *val_bleu*> Chaque valeur de couleur est un nombre entier compris entre 0 et 255, identifiant la valeur rouge (*val_rouge*), la valeur vert (*val_vert*) et la valeur bleu (*val_bleu*) de l'image. file=<*empl_fichier*, *nom_fichier*> *Empl_fichier* correspond à l'emplacement du fichier du `serveur`. *nom_fichier*, correspond au chemin d'accès complet du fichier, dans le format convenant au système sur lequel il réside, ou au nom de fichier relatif. Ce dernier est résolu par DB2 Extensions à l'aide des variables d'environnement (reportez-vous à la section «Utilisation des variables d'environnement pour la résolution des noms de fichiers» à la page 585).
histogram, histogramcolor ou QbColorHistogramFeatureClass	histogram=<(*val_hist*, *val_rouge*, *val_vert*, *val_bleu*>), ... Chaque valeur de couleur d'histogramme est précisée dans une clause identifiant le pourcentage (de 1 à 100) de cette couleur dans l'histogramme (*valeur_hist*), ainsi que la valeur de rouge (*val_rouge*), de vert (*val_vert*) et de bleu (*val_bleu*) pour cette couleur. file=<*empl_fichier*, *nom_fichier*> *Empl_fichier* correspond à l'emplacement du fichier du `serveur`. *nom_fichier*, correspond au chemin d'accès complet du fichier, dans le format convenant au système sur lequel il réside, ou au nom de fichier relatif. Ce dernier est résolu par DB2 Extensions à l'aide des variables d'environnement.
draw, positional ou QbDrawFeatureClass	file=<*empl_fichier*, *nom_fichier*> handle=<*descripteur_image*> *Empl_fichier* correspond à l'emplacement du fichier du `serveur`. *nom_fichier*, correspond au chemin d'accès complet du fichier, dans le format convenant au système sur lequel il réside, ou au nom de fichier relatif. Ce dernier est résolu par DB2 Extensions à l'aide des variables d'environnement.

Tableau 9. Valeurs de caractéristiques autorisées dans la chaîne de requête (suite)

Nom de la caractéristique	Valeur
texture ou QbTextureFeatureClass	file=<*empl_fichier*, *nom_fichier*> handle=<*descripteur_image*> *Empl_fichier* correspond à l'emplacement du fichier du `serveur`. *nom_fichier*, correspond au chemin d'accès complet du fichier, dans le format convenant au système sur lequel il réside, ou au nom de fichier relatif. Ce dernier est résolu par DB2 Extensions à l'aide des variables d'environnement.

Poids de caractéristique

Lorsque vous spécifiez de nombreuses caractéristiques dans une requête, vous pouvez également affecter un poids à une ou plusieurs d'entre elles. Celui-ci indique l'importance accordée par l'extension Image à la caractéristique concernée lors du calcul de scores de similarité et du renvoi des résultats d'une requête par contenu d'image. L'importance de la caractéristique au sein de la requête est proportionnelle à son poids. Celui-ci est exprimé sous la forme d'un nombre réel supérieur à 0.0, par exemple, 2.5 ou 10.0. Si vous n'affectez pas de poids à une caractéristique dans une chaîne de requête, l'extension Image attribue la valeur par défaut correspondante. Il est inutile d'affecter un poids à une caractéristique si la chaîne de requête n'en contient qu'une seule. (Dans ce cas, tout le poids de la requête porte sur cette caractéristique.)

Le poids d'une caractéristique est calculé par rapport aux autres caractéristiques spécifiées dans la requête. Par exemple, supposons que vous indiquiez les caractéristiques de couleur moyenne et de texture dans une chaîne de requête et que vous spécifiez également une valeur de poids de 2.0 pour la couleur moyenne. Cela indique à l'extension Image d'attribuer deux fois plus d'importance à la valeur de la couleur moyenne qu'à celle de la texture.

Exemples

La chaîne de requête suivante précise une couleur moyenne de rouge :

```
averageColor color=<255, 0, 0>
```

La chaîne de requête suivante spécifie un histogramme composé de 10% de rouge, 50% de vert et 40% de bleu :

```
histogram histogram=<(10, 255, 0, 0), (50, 0, 255, 0),
                                      (40, 0, 0, 255)>
```

La chaîne de requête suivante spécifie une valeur de couleur moyenne et une valeur de texture. La valeur de texture est fournie par une image contenue dans un fichier serveur. Le poids de la texture est le double de celui de la couleur moyenne :

```
averageColor color=<30, 200, 25> and
  texture file=<server, "\patterns\pattern7.gif"> weight=2.0
```

Utilisation d'un objet de requête

Vous pouvez utiliser un objet de requête pour identifier les caractéristiques, leurs valeurs et leur poids pour votre requête. Vous créez l'objet de requête et lui ajoutez des caractéristiques. Ensuite, vous spécifiez une source de données pour chacune de celles-ci. La source de données fournit la valeur correspondant à la caractéristique et peut être, par exemple, une image contenue dans un fichier. Si la couleur moyenne est la caractéristique importante, la couleur moyenne de l'image est associée à l'objet de requête. En cas de spécification de nombreuses caractéristiques dans une requête, vous pouvez également affecter un poids à une ou plusieurs d'entre elles.

L'extension Image fournit trois API (QbQuerySearch, QbQueryStringSearch et QbQueryNameSearch) et deux fonctions UDF (QbScoreFromName et QbScoreTBFromName) permettant d'utiliser un objet de requête. Lorsque vous lancez une requête, vous devez utiliser l'API ou la fonction utilisateur appropriée, et spécifier l'objet de requête en tant que paramètre d'entrée. (Pour plus de détails, reportez-vous à la section «Lancement de requêtes par contenu d'images» à la page 167.)

Création d'un objet de requête

Utilisez l'API QbQueryCreate pour créer un objet de requête. Suite à cela, l'extension Image renvoie un descripteur correspondant à cet objet. Ce descripteur est d'un type de données propres à QBIC,QbQueryHandle, qui est défini dans le fichier (en-tête) d'inclusion pour QBIC, dmbqbapi.h.

Lorsque vous utilisez cette API, vous devez spécifier le descripteur de l'objet de recherche. Vous devez également indiquer ce descripteur dans les API effectuant d'autres opérations sur l'objet de requête, telles que l'ajout d'une caractéristique.

Par exemple, l'appel d'API suivant crée un objet de requête :

```
QbQueryHandle  qHandle;

rc=QbQueryCreate(
             &qHandle);                          /* descripteur de l'objet */
                                                 /* de requête */
```

Ajout d'une caractéristique à un objet de requête

Vous identifiez la caractéristique de l'image que l'extension Image doit rechercher en l'ajoutant à un objet de requête.

Utilisez l'API QbQueryAddFeature pour ajouter une caractéristique à un objet de requête. Lorsque vous utilisez cette API, indiquez le descripteur de l'objet de requête ainsi que le nom de la caractéristique. Vous ne pouvez spécifier qu'une seule caractéristique dans l'API. Vous devez émettre un appel API distinct pour chaque caractéristique à ajouter à un objet de requête.

Dans l'exemple ci-dessous, l'API QbQueryAddFeature est utilisée pour ajouter la caractéristique couleur moyenne à un objet de requête :

```
char  featureName[qbiMaxFeatureName];
QbQueryHandle  qHandle;

rc=QbQueryAddFeature(
               qHandle,                  /* descripteur de l'objet de requête */
               "QbColorFeatureClass");          /* nom de la caractéristique */
```

Indication de la source de données pour une caractéristique contenue dans un objet de requête

Utilisez l'API QbQuerySetFeatureData afin d'indiquer la source de données pour une caractéristique contenue dans un objet de requête. La source de données peut être :

- une image cataloguée ou décataloguée d'une colonne de table utilisateur,
- un fichier image sur un poste de travail client,
- une image dans une mémoire tampon sur un poste de travail client.

En outre, vous pouvez indiquer explicitement des données pour la caractéristique couleur moyenne ou couleur d'histogramme. Par exemple, vous pouvez indiquer les valeurs de rouge, vert et bleu d'une couleur moyenne.

Lorsque vous utilisez cette API :

- Spécifiez le descripteur de l'objet de requête.
- Précisez le nom de la caractéristique.
- Pointez sur la structure QbImageSource (pour plus de détails, reportez-vous à la page 160).

Utilisation de structures de source de données: Trois structures sont utilisées pour fournir des informations de source de données pour un objet de requête. Il s'agit de :

- QbImageSource (structure)
- QbColor (structure)
- QbHistogramColor (structure)

QbImageSource : La structure QbImageSource identifie le type de source pour une caractéristique contenue dans un objet de requête. Cette structure est définie dans le fichier (en-tête) d'inclusion pour QBIC, `dmbqbapi.h`, comme suit :

```
typedef struct{
          SQLINTEGER    type;
          union {
          char       imageHandle[MMDB_BASE_HANDLE_LEN+1];
                QbImageFile       clientFile;
                QbImageBuffer     buffer;
                QbSampleSource    reserved;
                QbColor           averageColor;
                QbHistogramColor  histogramColor[qbiHistogramCount];
          };
} QbImageSource;
```

La zone type de la structure QbImageSource indique le type de la source. Vous pouvez définir la valeur de cette zone comme suit :

Valeur	Signification
qbiSource_ImageHandle	La source se trouve dans une colonne de table utilisateur.
qbiSource_ClientFile	La source se trouve dans un fichier du poste client.
qbiSource_Buffer	La source se trouve dans la mémoire tampon du poste client.
qbiSource_ServerFile	La source se trouve dans un fichier de serveur.
qbiSource_AverageColor	La source est une spécification de couleur moyenne.
qbiSource_HistogramColor	La source est une spécification de couleur d'histogramme.

Ces paramètres ne sont valables que pour la caractéristique correspondante. Par exemple, qbiSource_AverageColor n'est valable que pour la caractéristique couleur moyenne.

Si vous attribuez la valeur qbiSource_ServerFile à la zone de type, utilisez clientFile comme nom et type du fichier sur le serveur.

En fonction du type de source, l'extension Image analyse également d'autres informations, comme l'indique le tableau 10 à la page 162.

Tableau 10. Informations analysées par l'extension Image dans QbImageSource

Source	Informations analysées par l'extension Image	Zone
table utilisateur	descripteur d'image	zone du descripteur d'image de QbImageSource
fichier	nom du fichier format du fichier	zone clientFile de QbImageSource
mémoire tampon	nom du fichier	QbImageBuffer (pour plus de détails sur l'utilisation de cette structure, reportez-vous à la page 162)
spécification de la couleur moyenne	valeurs des couleurs rouge, vert et bleu	QbColor (pour plus de détails sur l'utilisation de cette structure, reportez-vous à la page 162)
spécification de la couleur d'histogramme	valeurs et pourcentages de couleur	QbHistogramColor (pour plus de détails sur l'utilisation de cette structure, reportez-vous à la page 162)

QbImageBuffer : Utilisez la structure QbImageBuffer pour indiquer le format, la longueur et le contenu d'une image lorsque la source de données se trouve dans une mémoire tampon. Cette structure est définie dans le fichier (en-tête) d'inclusion pour QBIC, dmbqbapi.h, comme suit :

```
typedef struct{
          char          format[qbiImageFormatLength+1];
          SQLINTEGER    length;
          char*         image;
} QbImageBuffer;
```

QbColor : Utilisez la structure QbColor pour indiquer les valeurs de rouge, de vert et de bleu d'une couleur moyenne lorsque la source de données est une spécification de couleur moyenne. Cette structure est définie dans le fichier (en-tête) d'inclusion pour QBIC, dmbqbapi.h, comme suit :

```
typedef struct{
          SQLUSMALLINT    red;          /*0 mini - 65535 (maxi) */
          SQLUSMALLINT    green;        /*0 mini - 65535 (maxi) */
          SQLUSMALLINT    blue;         /*0 mini - 65535 (maxi) */
} QbColor;
```

Définissez les valeurs de la structure QbColor afin d'indiquer le pourcentage de pixels de rouge, vert et bleu à intégrer dans le calcul de la valeur de la couleur moyenne. Vous pouvez indiquer une valeur comprise entre 0 et 65535. La valeur 0 indique que l'entrée doit être ignorée.

QbHistogramColor : Utilisez la structure QbHistogramColor pour indiquer chaque composant couleur d'une spécification de couleur d'histogramme. La spécification complète d'une couleur d'histogramme se trouve dans un tableau de structures QbHistogramColor. Chaque structure contient une valeur de couleur et un pourcentage. La valeur de couleur est constituée de valeurs de pixels de rouge, vert et bleu. Le pourcentage est celui de la couleur souhaité dans l'image cible.

Cette structure est définie dans le fichier (en-tête) d'inclusion pour QBIC, dmbqbapi.h, comme suit :

```
typedef struct{
          QbColor         color;
          SQLUSMALLINT    percentage;   /*0 - 100 */
} QbHistogramColor;
```

Définissez les valeurs de la structure QbColor afin d'indiquer le pourcentage de pixels de rouge, vert et bleu constituant la couleur. Vous pouvez indiquer une valeur comprise entre 0 et 65535. Indiquez le pourcentage souhaité de la couleur spécifiée dans l'image cible. Vous pouvez indiquer une valeur comprise entre 1 et 100. La somme des pourcentages pour les composants couleur d'une couleur d'histogramme ne peut dépasser 100.

Exemples : Dans l'exemple ci-après, l'API indique la source de données pour la caractéristique couleur d'histogramme dans un objet de requête. La source de données est un fichier résidant sur le poste client.

```
char  featureName[qbiMaxFeatureName];
QbQueryHandle  qHandle;
QbImageSource        imgSource;

imgSource.type=qbiSource_ClientFile;
strcpy(imgSource.clientFile.fileName,"/tmp/image.gif");
strcpy(imgSource.clientFile.format,"GIF");

rc=QbQuerySetFeatureData(
               qHandle,                 /* descripteur de l'objet de requête */
               "QbColorHistogramFeatureClass",  /* nom de la caractéristique */
               &imgSource);                /* source de données de la */
                                           /* caractéristique */
```

Dans l'exemple suivant, la source de données est une spécification de couleur moyenne pour le rouge :

```
char  featureName[qbiMaxFeatureName];
QbColor              avgColor;
QbImageSource        imgSource;

imgSource.type=qbSource_AverageColor;
avgColor.red=255;
avgColor.green=0;
avgColor.blue=0;
```

```
strcpy(featureName,"QbColorFeatureClass");

rc=QbQuerySetFeatureData(
                qHandle,                    /* descripteur de l'objet de requête */
                featureName,;                    /* nom de la caractéristique */
                &imgSource);                /* source de données de la */
                                            /* caractéristique */
```

Configuration du poids d'une caractéristique contenue dans un objet de requête

En cas d'ajout de plusieurs caractéristiques à un objet de requête, vous pouvez configurer le poids à affecter à une ou plusieurs d'entre elles dans la requête. Utilisez l'API QbQuerySetFeatureWeight pour spécifier le poids d'une caractéristique. Celui-ci indique l'importance accordée par l'extension Image à la caractéristique concernée lors du calcul de scores de similarité et du renvoi des résultats d'une requête par contenu d'image. L'importance de la caractéristique au sein de l'objet de requête est proportionnelle au poids qui lui est associé.

Vous pouvez affecter un poids à une ou plusieurs caractéristiques dans un objet de requête, bien que vous ne puissiez spécifier que le poids d'une seule caractéristique lorsque vous lancez l'API QbQuerySetFeatureWeight. Si vous n'affectez pas de poids à une caractéristique dans un objet de requête, l'extension Image attribue la valeur par défaut correspondante. Il est inutile d'affecter un poids à une caractéristique si l'objet de requête n'en contient qu'une seule. (Dans ce cas, tout le poids de la requête porte sur cette caractéristique.)

Lorsque vous utilisez cette API :

- Spécifiez le descripteur de l'objet de requête.
- Indiquez le nom de la caractéristique.
- Spécifiez le poids de la caractéristique. Celui-ci est défini par un nombre réel supérieur à 0, par exemple, 2.5 ou 10.0. L'importance de la caractéristique au sein de l'objet de requête est proportionnelle au poids qui lui est associé. La configuration modifie tout poids défini précédemment pour la caractéristique dans l'objet de requête.

Dans l'exemple ci-dessous, l'objet de requête contient au minimum une autre caractéristique, outre celle de couleur moyenne. L'API QbQuerySetFeatureWeight est utilisée pour préciser le poids de la caractéristique de couleur moyenne dans l'objet de requête :

```
char  featureName[qbiMaxFeatureName];
double              weight;
QbQueryObjectHandle  qoHandle;

strcpy(featureName,"QbColorFeatureClass");
weight=5.00;
```

```
rc=QbQuerySetFeatureWeight(
                qoHandle,               /* descripteur de l'objet */
                                         /* de requête */
                featureName,;           /* nom de la caractéristique */
                &weight);               /* poids de la caractéristique */
```

Sauvegarde et réutilisation d'une chaîne de requête

Les objets de requête sont transitoires sauf si vous les sauvegardez. Ils n'existent que pendant une seule connexion à une base de données. Vous pouvez sauvegarder la chaîne de requête pour la réutiliser ultérieurement dans le programme, ou après abandon de la connexion à la base de données via des appels de programmes.

L'extension Image fournit l'API QbQueryGetString qui renvoie la chaîne de requête d'un objet de requête. Vous pouvez ainsi utiliser cette chaîne comme données d'entrée dans l'API QbQueryStringSearch ou dans les fonctions UDF QbScoreFromStr et QbScoreTBFromStr dans d'autres requêtes d'images par contenu (reportez-vous à la section «Lancement de requêtes par contenu d'images» à la page 167).

La chaîne de requête est créée lorsque la requête est générée à l'aide de :

- QbQueryCreate
- QbQueryAddFeature
- QbQuerySetFeatureData
- QbQuerySetFeatureWeight
- QbQueryRemoveFeature

Une fois la requête générée, vous pouvez appeler l'API QbQueryGetString pour obtenir la chaîne. Vous pouvez utiliser cette chaîne de requête dans les appels émis au sein de ce programme ou la sauvegarder dans un fichier afin de l'utiliser dans des appels ultérieurs lancés vers l'application et dans d'autres connexions à la base de données. Après utilisation de la chaîne renvoyée par QbQueryGetString, vous devez libérer explicitement l'espace utilisé.

Dans l'exemple suivant, l'API QbQueryGetString est utilisée pour extraire la chaîne de requête à partir d'un objet de requête :

```
  SQLRETURN rc;
char* qryString;
QbQueryHandle  qHandle;

.....           /* Création et utilisation de la chaîne de requête */

rc = QbQueryGetString(qHandle, &qryString);
if ( rc == 0) {
```

```
   ...         /* Utilisation de la chaîne de requête en entrée */
   free((void *)qryString);
   qryString=(char *)0;
}
```

Restriction : Lorsque vous utilisez un fichier client pour indiquer la source de données d'une caractéristique, la chaîne de requête ne reflète pas les données de cette dernière.

Extraction d'informations relatives à un objet de requête

Vous pouvez déterminer les caractéristiques qui ont été ajoutées à un objet de requête (le cas échéant), ainsi que le poids actuellement affecté à une caractéristique.

Utilisez cette API	pour extraire
QbQueryGetFeatureCount	le nombre de caractéristiques figurant dans un objet de requête
QbQueryListFeatures	les noms des caractéristiques figurant dans un objet de requête

Lorsque vous utilisez l'API QbQueryGetFeatureCount, indiquez le descripteur de l'objet de requête. Vous devez également pointer sur un compteur. L'extension Image renvoie dans ce dernier le décompte des caractéristiques.

Dans l'exemple ci-dessous, l'API QbQueryGetFeatureCount est utilisée pour déterminer le nombre de caractéristiques contenues dans un objet de requête :

```
 SQLINTEGER     count;
QbQueryHandle  qHandle;

rc=QbQueryGetFeatureCount(
                qHandle,              /* descripteur de l'objet de requête */
                &count);              /* nombre de caractéristiques */
```

Lorsque vous émettez l'appel d'API QbQueryListFeatures, vous devez affecter une mémoire tampon dans laquelle sera placé le nom de caractéristique renvoyé. Vous devez également indiquer le descripteur du catalogue et la taille de la mémoire tampon pour les noms de caractéristiques renvoyés.

Dans l'exemple suivant, l'API QbQueryListFeatures est utilisée pour extraire le nom de chaque caractéristique contenue dans un objet de requête:

```
SQLINTEGER     retCount,bufSize;
char*          featureName;
QbQueryHandle  qHandle;

bufSize=qbiMaxFeatureName;
featureName=(char*)malloc(bufSize);

rc=QbQueryListFeatures(
```

```
                qHandle,                /* descripteur de l'objet de requête */
                bufSize                  /* taille de la mémoire tampon */
                &retCount,               /* nombre de caractéristiques */
                featureName);            /* mémoire tampon pour les noms */
                                         /* de caractéristiques */
```

Suppression d'une caractéristique dans un objet de requête

Pour supprimer une caractéristique dans un objet de requête, utilisez l'API QbQueryRemoveFeature. Lorsque vous utilisez cette API, indiquez le descripteur de l'objet de requête et le nom de la caractéristique.

Dans l'exemple suivant, l'API QbQueryRemoveFeature est utilisée pour supprimer la caractéristique couleur d'histogramme d'un objet de requête :

```
char  featureName[qbiMaxFeatureName];
QbQueryHandle  qHandle;

strcpy(featureName,"QbColorHistogramFeatureClass");

rc=QbQueryRemoveFeature(
                qHandle,              /* descripteur de l'objet de requête */
                featureName);        /* nom de la caractéristique */
```

Suppression d'un objet de requête

Supprimez un objet de requête ne portant pas de nom à l'aide d'une API QbQueryDelete. L'extension Image supprime la requête de la base de données à laquelle vous êtes connecté.

Lorsque vous utilisez l'API QbQueryDelete, indiquez le descripteur de l'objet de requête.

Dans l'exemple suivant, l'API QbQueryDelete est utilisée pour supprimer un objet de requête :

```
QbQueryHandle  qHandle;

rc=QbQueryDelete(
              qHandle);                          /* descripteur de l'objet */
                                                 /* de requête */
```

Si vous avez utilisé une requête nommée, supprimez l'objet de requête à l'aide de l'API QbQueryNameDelete.

Lancement de requêtes par contenu d'images

Après le catalogage des images, vous pouvez effectuer une recherche par contenu sur une ou plusieurs images. Dans une recherche de ce type, vous identifiez les données d'entrée de la requête, ainsi qu'un échantillon cible d'images cataloguées. Vous pouvez indiquer ces données d'entrée dans une chaîne de requête (reportez-vous à la section «Définition d'une chaîne de

requête» à la page 155) ou un objet de requête (reportez-vous à la section «Utilisation d'un objet de requête» à la page 159).

Lorsque vous utilisez une chaîne de requête, vous pouvez lancer l'opération à partir de la ligne de commande DB2 ou d'un programme. Avec un objet de requête, vous pouvez soumettre la requête à partir d'un programme en indiquant le descripteur correspondant.

L'extension Image compare la valeur de caractéristique indiquée dans la requête à celles des images cible et calcule un score pour chaque image. Ce dernier indique le degré de similarité entre la valeur de caractéristique de l'image cible et celle indiquée dans la requête.

Vous pouvez extraire les images dont les valeurs de caractéristique sont les plus proches de celle de la requête. Vous pouvez également ne rechercher qu'une image cataloguée et obtenir son score, ou extraire les scores de toutes les images cataloguées dans une colonne de table.

Recherche d'images

L'extension Image contient trois API permettant de rechercher les images cataloguées d'une colonne de table. Ces API ne diffèrent que par le format des données d'entrée, à savoir une chaîne ou un objet de requête :

API	Données d'entrée
QbQueryStringSearch	Chaîne de requête
QbQuerySearch	Descripteur de l'objet de requête
QbQueryNameSearch	Nom de l'objet de requête

Pour ces trois API, vous devez également :

- Nommer la table et la colonne contenant les images faisant l'objet de la recherche. Elles doivent être cataloguées dans un catalogue QBIC.
- Indiquer le nombre maximal de résultats à renvoyer.
- Pointer sur une structure indiquant la portée de la requête. Attribuez la valeur 0, NULL, ou une chaîne de caractères vide au pointeur. Ces valeurs indiquent que la recherche porte sur toutes les images cataloguées de la colonne de la table.
- Spécifiez la constante qbiArray pour indiquer que les résultats sont stockés dans un tableau. La constante qbiArray est définie dans le fichier (en-tête) d'inclusion pour QBIC, dmbqbapi.h.

Vous devez également pointer sur un tableau de structures de sortie afin que les résultats de la recherche y soient placés. Ensuite, l'extension Image renvoie dans ces structures les indicateurs des images cibles dont les valeurs de caractéristique sont les plus proches de celle de la requête. Elle renvoie également un score pour chaque image qui précise le degré de similarité de la

valeur de caractéristique de l'image par rapport à la requête. Cette structure est définie dans le fichier (en-tête) d'inclusion pour QBIC, dmbqbapi.h, comme suit :

```
typedef struct{
          char       imageHandle[MMDB_BASE_HANDLE_LEN+1];
          SQLDOUBLE  SCORE
} QbResult;
```

Vous devez affecter un tableau suffisamment grand pour contenir le nombre maximal de résultats et pointer sur ce tableau dans l'API. Vous devez également pointer sur un compteur : l'extension Image lui attribue une valeur en fonction du nombre des résultats qu'elle renvoie.

Dans l'exemple ci-dessous, l'API QbQueryStringSearch est utilisée pour effectuer une requête QBIC sur les images cataloguées d'une colonne de table. Notez que le pointeur sur la portée de la requête est défini par 0.

```
QbResult       returns[MaxQueryReturns];
SQLINTEGER     maxResults=qbiMaxQueryReturns;
 SQLINTEGER     count;
QbQueryHandle  qHandle;
QbResult       results[qbiMaxQueryReturns];

rc=QbQueryStringSearch(
          "QbColorFeatureClass color=<255, 0, 0>" /* chaîne de requête */
          "Employés",               /* table utilisateur */
          "photo",                  /* colonne image */
               maxResults,          /* nombre maximal de résultats */
               0,                   /* pointeur sur la portée de la requête * /
               qbiArray,            /* stockage de résultats dans un tableau */
               &count,              /* nombre d'images renvoyées */
               results);             /* tableau des résultats renvoyés */
```

La requête ci-dessous permet d'effectuer la même recherche à l'aide de l'API QbQuerySearch. Notez que le descripteur de l'objet de requête est défini comme données d'entrée.

```
QbResult       returns[MaxQueryReturns];
SQLINTEGER     maxResults=qbiMaxQueryReturns;
 SQLINTEGER     count;
QbQueryHandle  qHandle;
QbResult       results[qbiMaxQueryReturns];

rc=QbQuerySearch(
               qHandle,              /* descripteur de l'objet de requête */
               "Employés",          /* table utilisateur */
               "photo",             /* colonne image */
               maxResults,          /* nombre maximal de résultats */
               0,                   /* pointeur sur la portée de la requête * /
               qbiArray,            /* stockage de résultats dans un tableau */
               &count,                    /* nombre d'images renvoyées */
               results);                  /* tableau des résultats renvoyés */
```

Extraction du score d'une image

Utilisez dans une instruction SQL l'une des quatre fonctions utilisateur fournies par l'extension Image pour extraire le score d'une image cataloguée dans une colonne de table. Il s'agit d'une valeur en virgule flottante, double précision, allant de 0.0 à une valeur proche de l'infini. Plus cette valeur est faible, plus les valeurs des caractéristiques de l'image sont proches de celles spécifiées dans la requête. La valeur 0.0 indique que l'image correspond exactement.

Ces fonctions utilisateur (UDF) sont les suivantes :

- QbScorefromStr
- QbScoreTBfromStr
- QbScoreFromName
- QbScoreTBFromName

Recommandation : Utilisez la fonction UDF QbScoreFromStr pour extraire le score d'une seule image cataloguée. Utilisez la fonction UDF QbScoreTBFromStr pour extraire le score de plusieurs images cataloguées dans une colonne de table.

Extraction du score d'une seule image

Utilisez la fonction UDF QbScoreFromStr pour extraire le score d'une seule image cataloguée dans une colonne de table. Indiquez une chaîne de requête comme données d'entrée pour la fonction UDF QbScoreFromStr. Si vous utilisez la fonction UDF QbScoreFromName, indiquez le nom d'un objet de requête comme données d'entrée pour la fonction UDF QbScoreFromName. Pour toutes deux, vous devez également spécifier la colonne de table contenant l'image cible.

Dans la requête suivante, la fonction UDF QbScoreFromStr permet d'extraire les images cataloguées contenues dans une colonne de table et dont le score pour la couleur moyenne est très proche du rouge.

```
SELECT name, description
decimal (QbScoreFromStr(swatch_img,
            'QbColorFeatureClass color=<255, 0, 0>'),   /* chaîne de requête */
                 10, 5) AS score
FROM fabric                                             /* colonne de table */
ORDER BY score
```

Extraction du score de plusieurs images

Utilisez la fonction UDF QbScoreTBFromStr pour extraire le score de plusieurs images cataloguées dans une colonne de table. Vous pouvez utiliser la fonction UDF QbScoreTBFromName si vous disposez d'une requête nommée. Elles renvoient toutes deux une table comportant deux colonnes contenant respectivement les descripteurs d'image et les scores. Les lignes de la table

sont classées par ordre croissant de score. Dans la table de résultats, la colonne des descripteurs et la colonne des scores portent respectivement le nom de IMAGE_ID et SCORE.

Indiquez une chaîne de requête en tant que données d'entrée pour la fonction UDF QBScoreTBFromStr, et le nom d'un objet de requête pour la fonction QbScoreTBFromName. Pour toutes deux, vous devez également spécifier le nom de la colonne et de la table contenant les images cible. Vous pouvez aussi préciser le nombre maximal de lignes à renvoyer dans la table de résultats. Sinon, la fonction utilisateur renvoie une ligne pour chaque image cataloguée dans la colonne de table cible.

Dans la requête suivante, la fonction utilisateur QbScoreTBFromStr permet d'extraire les dix images cataloguées contenues dans une colonne de table dont la texture est la plus proche de celle d'une image d'un fichier serveur.

```
SELECT name, description
FROM fabric
WHERE CAST (swatch_img as varchar(250)) IN
 SELECT CAST (image_id as varchar(25)) FROM TABLE
 (QbScoreTBFromStr
 (QbTextureFeatureClass file=<server,"patterns/ptrn07.gif">' /*chaîne de requête*/

      'fabric',                        /* table */
      'swatch_img',              /* colonne de table */
      10))                                    /* nombre maximal de résultats */
  AS T1));
```

Modèle de programme de recherche QBIC

La figure 27 à la page 173 est un extrait d'un programme codé en C qui crée et exécute une requête QBIC. Ce programme effectue une recherche d'images par couleur moyenne. Il invite l'utilisateur à entrer le nom d'une couleur ou d'un fichier image. L'utilisateur peut également exploiter une image renvoyée par une requête comme échantillon graphique pour une recherche ultérieure. Le programme utilise alors la couleur nommée ou la couleur de l'image comme couleur moyenne pour effectuer une recherche dans une colonne d'images.

Vous trouverez le programme complet dans le fichier QBICDEMO.C du sous-répertoire SAMPLES. Il peut être utilisé pour rechercher des images par couleur d'histogramme, positionnelle, ou moyenne. Pour exécuter le programme complet, vous devez d'abord exécuter les modèles de programme ENABLE, POPULATE et QBCATDMO (qui se trouvent également dans le sous-répertoire SAMPLES). Pour plus d'informations sur les modèles de programmes, reportez-vous à l'«Annexe B. Modèles de programmes et fichiers de support» à la page 595.

Notez les points suivants sur la figure 27 à la page 173 :

1 Inclusion du fichier en-tête dmbqbapi.

2 Invite d'entrée des informations sur la base de données.

3 Connexion à la base de données.

4 Création d'un objet de requête.

5 Ajout d'une caractéristique à l'objet de requête.

6 Invite d'entrée relative au type d'entrée (nom de couleur, fichier image ou image précédemment extraite).

7 Indication de la source de données pour la caractéristique. La source de données est une spécification explicite de la couleur moyenne.

8 Lancement de la requête. L'extension Image effectue la recherche dans la totalité de la colonne d'images. Elle indique également le nombre maximal d'images à renvoyer (10).

9 Affichage de l'image suivante du jeu d'images renvoyées. Pour plus d'informations sur l'affichage des images, reportez-vous à la section «Affichage d'un objet image ou vidéo en grandeur réelle» à la page 135.

10 Suppression de l'objet de requête.

Le sous-répertoire SAMPLES comprend un autre programme montrant comment créer et utiliser une requête QBIC. Ce programme, `QbicQry.java`, permet à l'utilisateur de préciser graphiquement les critères de recherche pour une requête QBIC. Par exemple, il propose une palette de couleurs permettant de sélectionner la couleur moyenne et convertit l'option choisie en chaîne de requête.

```
#include <sql.h>
#include <sqlcli.h>
#include <sqlcli1.h>
#include <dmbqbqpi.h>  1
#include <stdio.h>
#include <string.h>
#include <malloc.h>
#include <color.h>
#include <ctype.h>

#define MaxQueryReturns  10

#define   MaxDatabaseNameLength  SQL_SH_IDENT
#define   MaxUserIdLength        SQL_SH_IDENT
#define   MaxPasswordLength      SQL_SH_IDENT
#define   MaxTableNameLength     SQL_LG_IDENT
#define   MaxColumnNameLength    SQL_LG_IDENT

static  char       databaseName[MaxDatabaseNameLength+1];
static  char       userid[MaxUserIdLength+1];
static  char       password[MaxPasswordLength+1];

static  char       tableName[MaxTableNameLength+1];
static  char       columnName[MaxColumnNameLength+1];

static  char       line[4000];

static  QbResult   results[MaxQueryReturns];
static  long       currentImage   = -1;
static  long       imageCount  = 0;

static  char*      tableAndColumn;

static  QbQueryHandle averageHandle = 0;
static  QbQueryHandle histogramHandle = 0;
static  QbQueryHandle drawHandle = 0;
static  QbQueryHandle lastHandle = 0;

static SQLHENV henv;
static SQLHDBC hdbc;
static SQLHSTMT hstmt;
static SQLRETURN rc;

static  char*    listQueries =
"SELECT NAME,DESCRIPTION FROM MMDBSYS.QBICQUERIES ORDER BY NAME";

static  char*    menu[] = {
```

Figure 27. Modèle de programme de recherche QBIC (Numéro 1 de 6)

```
/*
 12345678901234567890123456789012345678901234567890123456789012345678901234567890 */
"",
"+-------------------------------------------------------------------------------+",
"| AVERAGE COLOR colorname                                                       |",
"| AVERAGE FILE filename format                                                  |",
"| AVERAGE LAST                                                                  |",
"| Press Enter to display the next image in the series                           |",
"+-------------------------------------------------------------------------------+",
"",
0
};

static  char*       help[] = {
"",
"AVERAGE     Execute an average color query",
"  COLOR       Specifies the color to query for",
"  FILE        Specifies the file to compute the average color from",
"  LAST        Specifies the last displayed image be used to compute the color",
"  NEXT        Displays the next image from the current query or nothing if",
"                all of the image have been displayed."
"",
">>pause<<",
0
};
/*****************************************************************************/
/* doNext()                                                                  */
/*****************************************************************************/
static void doNext(void)
{
int ret;
   if (currentImage < imageCount)
      currentImage++;
   if (currentImage < imageCount)
      ret = DBiBrowse("/usr/local/bin/xv %s", MMDB_PLAY_HANDLE,
            results[currentImage].imageHandle, MMDB_PLAY_NO_WAIT); 9
}
```

Figure 27. Modèle de programme de recherche QBIC (Numéro 2 de 6)

```
/*****************************************************************************/
/* doAverage()                                                               */
/*****************************************************************************/
static void doAverage(void)
{
   QbQueryHandle qohandle = 0; QbImageSource is; char* type;
   char* arg1; char* arg2;

   type = nextWord(0);
   if (abbrev(type, "color", 1)) {
 is.type = qbiSource_AverageColor;
      arg1 = nextWord(0);
      if (arg1 == 0) {
         printf("AVERAGE COLOR command requires a colorname argument.\n");
      return;
      }
      if (getColor(arg1, &is.averageColor) == 0) {
         printf("The colorname entered was not recognized.\n");
      return;
      }
   }
   else if (abbrev(type, "file", 1)) {
      is.type = qbiSource_ClientFile;
      arg1 = nextWord(0);
      if (arg1 == 0) {
         printf("AVERAGE FILE command requires a filename argument.\n");
      return;
      }
      arg2 = nextWord(0);
      if (arg2 == 0) {
         printf("AVERAGE FILE command requires a file format argument.\n");
      return;
      }
      strcpy(is.clientFile.fileName, arg1);
      strcpy(is.clientFile.format, arg2);
   }
   else if (abbrev(type, "last", 1)) {
      is.type = qbiSource_ImageHandle;
      if (0 <= currentImage &&; currentImage < imageCount)
         strcpy(is.imageHandle, results[currentImage]imageHandle);
     else {
         printf("No last image for AVERAGE LAST command\n");
      return;
      }
   }
```

Figure 27. Modèle de programme de recherche QBIC (Numéro 3 de 6)

```
        else {
      printf("AVERAGE command only supports COLOR, FILE, and LAST types.\n");
      return;
   }

   _QbQuerySetFeatureData(averageHandle, "QbColorFeatureClass", &is);  7
   _QbQuerySearch(averageHandle, tableAndColumn, "IMAGE",
         MaxQueryReturns, 0, 0, &imageCount, results);  8
   lastHandle = averageHandle;

   currentImage = -1;
}
/*****************************************************************************/
/* commandLoop()                                                             */
/*****************************************************************************/
void commandLoop(void)
{
   int done = 0;

   while (!done) {  6
      displayText(menu);
      printf("%d", currentImage + 1);
      if (0 <= currentImage &&; currentImage < imageCount)
            printf("  %8.6f", results[currentImage].score);
      printf("> ");
   gets(line);
      done = processCommand(line);
   }
}
```

Figure 27. Modèle de programme de recherche QBIC (Numéro 4 de 6)

```
/*****************************************************************************/
/* main()                                                                    */
/*****************************************************************************/
void main(void)
{
   char* inst;
   int i;

   printf("\n\n");
   printf("Please enter: database_name [user_id] [password] "\n"); 2
   gets(line);
   if (copyWord(line, databaseName, sizeof(databaseName)) == 0)
      exit(0);
   copyWord(0, userid, sizeof(userid));
   copyWord(0, password, sizeof(password));
   printf("\n");

   if (SQLAllocEnv(&henv) != SQL_SUCCESS)
      sqlError(SQL_NULL_HSTMT);
   if (SQLAllocConnect(henv, &hdbc) != SQL_SUCCESS)
      sqlError(SQL_NULL_HSTMT);
   if (SQLConnect(hdbc, 3
                  (SQLCHAR*)databaseName,
                  SQL_NTS,
                  (SQLCHAR*)userid,
                  SQL_NTS,
                  (SQLCHAR*)password,
                  SQL_NTS) != SQL_SUCCESS)
      sqlError(SQL_NULL_HSTMT);

   printf("Initializing . . .\n");
```

Figure 27. Modèle de programme de recherche QBIC (Numéro 5 de 6)

```
   inst = getenv("DB2INSTANCE");
   if (inst != 0 &&; strcmp(inst, "keeseyt") == 0)
      tableAndColumn = "KEESEY.TEST";
   else
      tableAndColumn = "QBICDEMO.TEST";

   _QbQueryCreate(&averageHandle); 4
   _QbQueryAddFeature(averageHandle, "QbColorFeatureClass");
   _QbQueryCreate(&histogramHandle);
   _QbQueryAddFeature(histogramHandle, "QbColorHistogramFeatureClass");
   _QbQueryCreate(&drawHandle);
   _QbQueryAddFeature(drawHandle, "QbDrawFeatureClass"); 5

   commandLoop();

   _QbQueryDelete(drawHandle);
   _QbQueryDelete(histogramHandle); 10
   _QbQueryDelete(averageHandle);

   if (SQLDisconnect(hdbc) != SQL_SUCCESS)
      sqlError(SQL_NULL_HSTMT);
   if (SQLFreeConnect(hdbc) != SQL_SUCCESS)
      sqlError(SQL_NULL_HSTMT);
   if (SQLFreeEnv(henv) != SQL_SUCCESS)
      sqlError(SQL_NULL_HSTMT);
}
```

Figure 27. Modèle de programme de recherche QBIC (Numéro 6 de 6)

Chapitre 14. Détection de changements de plan vidéo

Ce chapitre explique comment détecter des changements de plan dans une séquence vidéo à l'aide des API de DB2 Extension Vidéo. Ces API sont disponibles sur toutes les plateformes de DB2 Extension Vidéo, sauf sous Windows 3.1. Cette détection n'est admise que pour les séquences vidéo au format MPEG-1.

Définition d'un changement de plan

Prenons l'exemple d'un studio de télévision qui enregistre des programmes sur des bandes vidéo en vue d'émissions diffusées ultérieurement. Depuis peu, il stocke ses séquences sur bandes vidéo dans une base de données DB2 utilisant l'extension Vidéo. Cette méthode permet à son personnel d'effectuer des recherches d'informations de type classique dans les programmes ou de visualiser des séquences.

Le personnel du studio souhaite pouvoir prévisualiser les séquences vidéo à l'aide d'une synopsis, appelée storyboard. La figure 28 à la page 180 présente un exemple de storyboard. La visualisation de storyboards permet au personnel de connaître l'essentiel d'une vidéo sans avoir à la visualiser entièrement. Elle permet également de déterminer si une vidéo correspond aux besoins, par exemple si elle vaut la peine d'être chargée et visualisée. Cela est très important pour le studio car la visualisation d'un storyboard, au lieu d'une vidéo complète, peut réduire considérablement les temps de chargement et de visualisation. Pour plus de détails sur l'utilisation des fonctions de détection de changements de plan vidéo dans cette optique, reportez-vous à la section «Stockage d'informations relatives à toutes les prises de vue d'une séquence vidéo» à la page 198.

Changement de plan

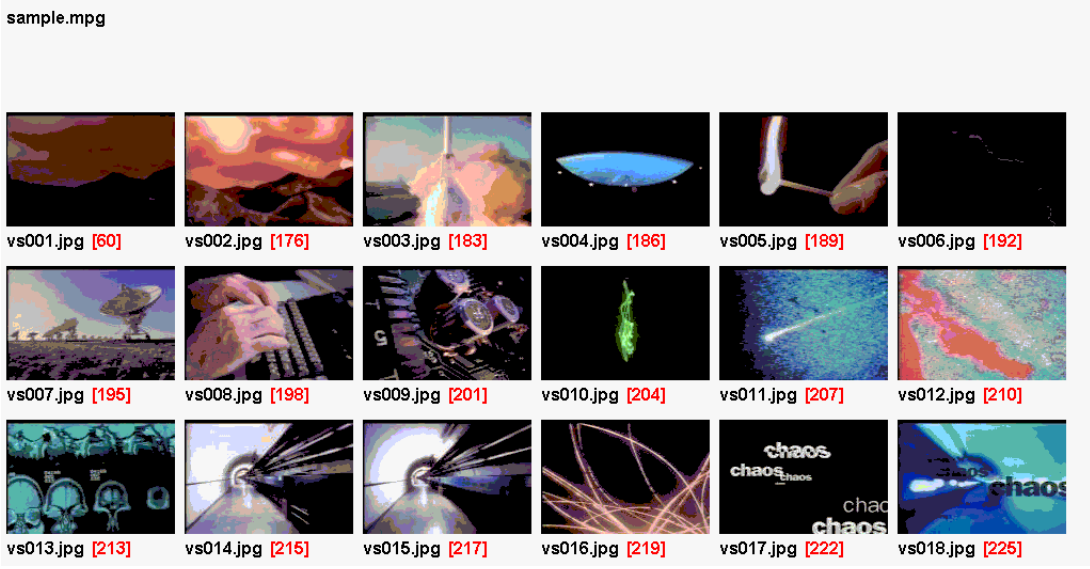


Figure 28. Storyboard vidéo. Les images représentatives résument le contenu et le déroulement d'une séquence vidéo.

Le studio envisage d'utiliser les fonctions de détection de changement de plan vidéo de l'extension Vidéo pour saisir des images représentatives pour ses storyboards.

Un **changement de plan vidéo** est un point d'une séquence vidéo où se produit une modification sensible entre 2 images successives. Cela se produit, par exemple, lorsque la caméra modifie son angle de prise de vue. Les images comprises entre deux changements de plan constituent une **prise de vue**.

Lorsque l'extension Vidéo détecte un changement de plan[6], elle enregistre les données de la prise de vue associée. Ces données comprennent le numéro de la première image de la prise de vue, celui de la dernière image et celui d'une image significative de la prise de vue. Elles comprennent également le contenu en pixels de l'image représentative.

Recherche et utilisation de changements de plan

L'extension Vidéo fournit un jeu d'interfaces de programmation d'applications permettant de rechercher des prises de vue ou des images dans une séquence vidéo. Une fois l'élément recherché retrouvé, il est possible d'accéder aux données le concernant (par exemple, le numéro de la première ou de la

6. Le code de détection de changement de plan vidéo englobe le décodeur MEPG de l'University of California (Berkeley), modifié par Boston University Multimedia Communication Laboratory.

dernière image de la prise de vue ou contenu en pixels d'une image). Vous pouvez ensuite transmettre ces informations à un programme pour traitement (par exemple, pour afficher l'image).

L'extension Vidéo fournit également des API permettant de stocker des données de prise de vue dans un **catalogue des prises de vue**. Celui-ci peut être stocké dans une base de données ou un fichier. Vous pouvez accéder à ce catalogue dans un fichier ou accéder à une vue du catalogue en lecture seule.

Un fichier catalogue des prises de vue est constitué de zones destinées à recevoir les données suivantes :

- Nom du catalogue des prises de vue
- Valeurs gérant la façon dont l'extension Vidéo détecte une prise de vue, par exemple, le nombre minimal d'images vidéo dans une prise de vue
- Valeurs gérant le nombre et la nature d'images vidéo qui seront stockées en tant qu'images représentatives d'une prise de vue
- Numéro de la prise de vue
- Numéro de la première image
- Numéro de la dernière image
- Numéro de l'image représentative
- Nom d'un fichier où est stocké le contenu de l'image représentative

La vue du catalogue des prises de vue dans une base de données contient des colonnes pour les données suivantes :

- Descripteur de prise de vue
- Nom de la table vidéo
- Nom des colonnes vidéo
- Descripteur de la séquence vidéo
- Nom de fichier de la séquence vidéo
- Numéro de la première image
- Numéro de la dernière image
- Numéro de l'image représentative
- Données de l'image représentative
- Commentaire

Il est possible d'accéder aux données du fichier catalogue des prises de vue ou de rechercher les données lorsque le catalogue se trouve dans une base de données. Les informations relatives aux images représentatives sont particulièrement utiles lors de la visualisation de storyboards. En outre, si le catalogue des prises de vue est dans une base de données, vous pouvez associer les données de prises de vue à des données connexes dans d'autres

tables. Par exemple, le personnel du studio vidéo peut créer un catalogue des prises de vue dans une base de données. Il peut ensuite associer les données du catalogue à une table contenant les séquences vidéo et les informations associées. Cela lui permet d'extraire, par une seule requête, une séquence vidéo et des informations connexes, ainsi que d'identifier les prises de vue de cette séquence.

Structure des données de détection de prises de vue

Les données associées à la détection d'une prise de vue sont stockées dans des structures figurant dans le fichier d'en-tête de détection de prises de vue, dmbshot.h. Pour la plupart des API associées, vous devez pointer sur une ou plusieurs de ces structures, dont certaines servent à stocker des données d'entrée pour l'extension Vidéo. Par exemple, la structure de contrôle de prise de vue contient des informations intervenant dans la détection de prises de vue. La plupart des structures sont utilisées par l'extension Vidéo pour stocker des données qu'elle a récupérées à partir d'un objet vidéo. Par exemple, la structure de données d'image vidéo contient le contenu en pixels d'une image.

Les structures utilisées pour la détection de prise de vue sont DBvIOType, DBvShotControl, DBvShotType, DBvFrameData et DBvStoryboardCtrl.

DBvIOType

La structure de données DBvIOType contient les données relatives à un objet vidéo (format, dimensions et nombre d'images). La structure de données est définie comme suit :

```
typedef struct {

   FILE *hFile;                    /* descripteur de fichier pour l'objet vidéo */
   char vhandle[255];              /* descripteur vidéo (à partir bd) */
   char vtable[255];               /* nom table vidéo (à partir bd) */
   char vcolumn[255];              /* nom colonne vidéo (à partir bd) */
   char vFile[255];                /* nom fichier vidéo */
   char idxFile[255];              /* nom fichier index */
   char isIdx;                     /* 1 si l'index existe, sinon 0 */
   char isInDb;                    /* 1 à partir bd, 0 à partir fichier */
   int format;                     /* Format de l'objet vidéo */
   unsigned long dx, dy;           /* Dimensions de l'objet vidéo */
   unsigned long totalFrames;      /* Nb total images dans l'objet vidéo */
   unsigned long markFrame;        /* Utilisé par la détection de prises de vue */
   unsigned long currentFrame;     /* Image vidéo en cours */
   DBvFrameData fd;                /* Données de l'image en cours */
   DBvDCFrameData fdDc;            /* Données des images DC */
   unsigned char BGRValid;         /* réservé */
   unsigned short usDeviceID;      /* réservé */
   unsigned long hwnd;             /* réservé */
   int videoReset;                 /* Signaler si vidéo ouverte ou analysée */
   int firstshot;                  /* Utilisé en interne pour indiquer le premier appel *
   void *reserved                  /* réservé */

} DBvIOType;
```

DBvShotControl

La structure de données DBvShotControl contient des informations utilisées pour le contrôle de la détection de prises de vue (par exemple, la méthode de détection). La structure de données est définie comme suit :

```
typedef struct {

  unsigned long reserved;
  unsigned long method;          /* méthode de détection */

    #define DETECT_CORRELATION  0x00000001
    #define DETECT_HISTOGRAM    0x00000002
    #define DETECT_CORRHIST     0x00000003
    #define DETECT_CORRHISTDISS 0x00000004


  int normalCorrValue;           /* Seuil de correspondance */
  int sceneCutSkipXY;            /* Réservé */
  int CorrHistThresh;            /* Seuil histogramme */
  int DissThresh;                /* Seuil de fondu */
  int DissCacheSize;             /* Nbre images dans courbe de référence */
  int DissNumCaches;             /* Nbre images dans vérification de cohérence */
  int minShotSize;               /* Nombre minimal images dans prise */

} DBvShotControl;
```

Le tableau 11 décrit les zones de DBvShotControl et indique les paramètres correspondants admis et par défaut. Pour initialiser ces zones à leur valeur par défaut, utilisez l'API DBvInitShotControl selon la procédure décrite à la section «Initialisation des valeurs dans les structures de données de détection de prises de vue» à la page 187.

Les paramètres DBvShotControl dépendent du type de séquence vidéo : Les changements de plan d'une séquence vidéo numérisée varient fortement en fonction du contenu et du format de la séquence. En outre, la précision des algorithmes de changement de plan varie d'une séquence vidéo à une autre. Les changements de plan clairement définis avec des différences évidentes dans l'apparence globale de l'image sont détectés plus facilement que les modifications plus subtiles ou les modifications pour lesquelles les couleurs restent globalement les mêmes. Bien que les paramètres par défaut pour la zone DBvShotControl soient adaptés à la plupart des applications, vous devrez peut-être les adapter pour réduire les risques de détection erronée ou impossible.

Tableau 11. Zones de DBvShotControl

Zone	Signification
method	Identifie la méthode utilisée par l'extension Vidéo pour détecter un changement de plan. Vous avez le choix entre les méthodes suivantes : DETECT_CORRELATION. Compare les pixels dans deux images successives. Si la différence dépasse le seuil de correspondance, il y a détection d'un changement de plan. DETECT_HISTOGRAM. Compare les valeurs d'histogramme de deux images successives. Ces valeurs mesurent la répartition des couleurs dans l'image. Si la différence dépasse le seuil d'histogramme, il y a détection d'un changement de plan. DETECT_CORRHIST. Utilisez la méthode de correspondance pour identifier d'éventuels changements de plan, puis utilisez la méthode d'histogramme pour les images repérées comme éventuels changements de plan. Si le seuil d'histogramme est dépassé, il y a détection d'un changement de plan. DETECT_CORRHISTDISS. Même chose que pour DETECT_CORRHIST, mais avec examen d'images supplémentaires pour les enchaînements. La méthode par défaut est DETECT_CORRHIST.
normalCorrValue	Nombre entier compris entre 0 et 100 qui indique le seuil de correspondance. Ce nombre correspond à la valeur minimale du coefficient de correspondance entre pixels dans deux images successives. La valeur 0 signifie qu'il y a toujours détection d'un changement de plan à l'image suivante. La valeur 100 signifie qu'il n'y a détection d'un changement de plan que si tous les pixels changent entre une image et la suivante. La valeur par défaut est 60.
sceneCutSkipXY	Réservé.
CorrHistThresh	Entier compris entre 0 et 100 qui indique le seuil d'histogramme. Cette valeur mesure la différence entre les valeurs d'histogramme d'images successives. La valeur 0 signifie qu'un changement de plan n'est détecté que si toutes les valeurs d'histogramme changent d'une image à la suivante. La valeur 100 signifie qu'il y a toujours détection d'un changement de plan par rapport à l'image suivante. La valeur par défaut est 10.
DissThresh	Entier compris entre 0 et 100 qui indique le seuil de test de fondu. Cette valeur mesure le pourcentage de pixels dans une image devant satisfaire à un test approprié pour qu'un fondu enchaîné soit détecté. La valeur 0 signifie qu'il y a toujours détection d'un fondu par rapport à l'image suivante. La valeur 100 signifie qu'il n'y a détection d'un fondu enchaîné que si tous les pixels de l'image réussissent le test de fondu. La valeur par défaut est 15.
DissCacheSize	Entier qui indique le nombre d'images utilisées dans la partie courbe de référence du test d'enchaînement. La valeur par défaut est 4.

Tableau 11. Zones de DBvShotControl (suite)

Zone	Signification
DissNumCaches	Entier qui indique le nombre d'images utilisées dans la partie vérification de cohérence du test d'enchaînement. La valeur par défaut est 7.
minShotSize	Entier qui indique le nombre minimum d'images par prise de vue. Pour qu'une prise de vue soit détectée, le nombre d'images qu'elle contient doit être au moins égal au minimum indiqué. La valeur par défaut est 5.

DBvShotType

La structure de données DBvShotType contient des informations relatives à une prise de vue (par exemple, le numéro de la première image, le numéro de la dernière image et le numéro d'image représentative) et un pointeur sur le contenu en pixels de l'image représentative. La structure de données est définie comme suit :

```
typedef struct {


unsigned long startFrame;         /* Numéro de la première image */
  unsigned long endFrame;         /* Numéro de la dernière image */
  unsigned long repFrame;         /* Numéro d'image représentative */
  DBvFrameData fd;                /* Données de la prise de vue représentative */

  unsigned long dx;               /* Largeur image en pixels */
  unsigned long dy;               /* Hauteur image en pixels */
  char *comment;                  /* Remarques sur la prise de vue */

} DBvShotType;
```

DBvFrameData

La structure de données DBvFrameData précise le contenu en pixels d'une image. La structure de données est définie comme suit :

```
typedef struct                /* Données image vidéo */
{
  /* MPEG 1 pixels */
  unsigned char *luminance;   /* Plan des pixels de luminance (noir et blanc) */
  unsigned char *Cr;          /* Plan des pixels Cr */
  unsigned char *Cb;          /* Plan des pixels Cb */
  unsigned char *reserved;

  } DBvFrameData;
```

DBvStoryboardCtrl

La structure de données DBvStoryboardCtrl contient des valeurs qui contrôlent les images représentatives d'une prise de vue stockées dans un catalogue vidéo, ainsi que leur nombre. Pour connaître l'utilisation de ces valeurs, reportez-vous à la section «Création d'un storyboard» à la page 199. La structure de données est définie comme suit :

```
typedef struct {

  int thresh1;              /* seuil pour plans petits à moyens */
  int thresh2;              /* seuils pour plans moyens à gros */
  int delta;                /* décalage utilisé pour les images représentatives */

} DBvStoryboardCtrl;
```

Le tableau 12 décrit les zones de DBvStoryboardCtrl et indique les paramètres correspondants admis et par défaut. Pour initialiser ces zones à leur valeur par défaut, utilisez l'API DBvInitStoryboardCtrl selon la procédure décrite à la section «Initialisation des valeurs dans les structures de données de détection de prises de vue» à la page 187.

Les paramètres DBvStoryboardCtrl dépendent du type de séquence vidéo : Le choix et le nombre des images représentatives les plus adaptés à un storyboard peuvent être différents d'un type de séquence vidéo à l'autre. Bien que les paramètres par défaut pour la zone DBvStoryboardCtrl conviennent à de nombreux types de séquences, il peut être nécessaire de les utiliser d'abord sur un sous-ensemble de vidéos d'essai. Vous pourrez ensuite affiner les paramètres avant de créer des storyboards pour une série de vidéos plus étendue.

Tableau 12. Zones de DBvStoryboardCtrl

Zone	Signification
thresh1	Identifie le seuil des prises de vue courtes. Les prises de vue comportant moins d'images que la valeur de thresh1 sont des prises courtes. Si elles sont cataloguées, les informations d'une prise de vue comprennent une seule image représentative (l'image médiane). La valeur par défaut est 90. Si la valeur définie pour thresh1 est -1, la prise de vue sera considérée comme courte, quelle que soit sa longueur réelle.

Tableau 12. Zones de DBvStoryboardCtrl (suite)

Zone	Signification
thresh2	Identifie le seuil des prises de vue moyennes ou longues. Les prises de vue comportant un nombre d'images égal ou inférieur à la valeur de thresh2, mais au moins égal à celle de thresh1, sont considérées comme des prises de vue moyennes. Si elles sont cataloguées, les informations d'une prise de vue moyenne comprennent deux images représentatives. La position des images représentatives est contrôlée par la valeur de la zone delta. Les prises de vue comportant un nombre d'images supérieur à la valeur de thresh2 sont considérées comme des prises longues. Si elles sont dans le catalogue, les informations d'une prise de vue longue comprennent trois images représentatives. La position de la première et de la dernière image représentative est contrôlée par la valeur de la zone delta. La deuxième image est l'image médiane. La valeur par défaut est 150. Si la valeur définie pour thresh1 est -1, la prise de vue sera considérée comme courte, quelle que soit sa longueur réelle.
delta	Identifie le décalage utilisé pour les images représentatives. Pour les prises de vue moyennes et longues, la première image représentative est décalée, par rapport au début de la prise, du nombre d'images défini dans delta. La dernière image représentative est décalée, par rapport à la fin de la prise, du nombre d'images défini dans delta. La valeur par défaut est 5.

Initialisation des valeurs dans les structures de données de détection de prises de vue

Les valeurs contenues dans la structure de données DBvShotControl contrôlent la détection de prises de vue. Celles qui se trouvent dans la structure de données DBvStoryboardCtrl contrôlent la création d'un storyboard. Vous pouvez indiquer explicitement des valeurs pour les zones de ces structures. De plus, vous pouvez initialiser les valeurs contenues dans ces structures à leur valeur par défaut. Pour connaître les valeurs par défaut utilisées dans la structure de données DBvShotControl, reportez-vous au tableau 11 à la page 184. Pour les valeurs par défaut utilisées dans la structure de données DBvStoryboardCtrl, reportez-vous au tableau 12 à la page 186.

Utilisez l'API DBvInitShotControl pour initialiser les valeurs de la structure de données DBvShotControl. Vous devez alors préciser la structure de contrôle de prise de vue. Par exemple, les instructions suivantes initialisent les zones de la structure DBvShotControl à leur valeur par défaut :

```
DBvShotControl    shotCtrl;

rc=DBvInitShotControl(
              shotCtrl);  /* pointeur sur la structure de contrôle de prise de vue */
```

Utilisez l'API DBvInitStoryboardCtrl pour initialiser les valeurs de la structure de données DBvStoryboardCtrl. Vous devez alors préciser la structure de contrôle de storyboard. Par exemple, les instructions suivantes initialisent les zones de la structure DBvStoryboardCtrl à leur valeur par défaut :

```
DBvStoryboardCtrl    sbCtrl;

rc=DBvInitStoryboardCtrl(
            sbCtrl);   /* pointeur sur la structure de contrôle de storyboard */
```

Extraction d'une prise de vue ou d'une image

L'extension Vidéo permet d'extraire une prise de vue ou une image à partir d'un objet vidéo. Pour cela, vous devez préalablement ouvrir l'objet vidéo pour la détection de prises de vue. L'extension Vidéo utilise un index pour accéder aux images et aux prises de vue. Vous devez donc préalablement créer un index pour la séquence vidéo pour pouvoir effectuer une extraction.

Une fois que la séquence vidéo est ouverte et l'index créé, vous pouvez extraire la prise de vue ou l'image suivante de la vidéo ou une image particulière en indiquant son numéro. L'extension Vidéo prend en charge les séquences vidéo au format MPEG-1. Si vous envisagez d'utiliser l'image extraite avec un programme exigeant le format RGB, il est possible d'effectuer une conversion de format à l'aide d'une API de l'extension Vidéo.

Ouverture d'une séquence vidéo pour la détection de prises de vue

Utilisez l'API DBvOpenFile pour ouvrir une séquence vidéo stockée dans un fichier. Le fichier doit être accessible à partir du poste client. Utilisez l'API DBvOpenHandle pour ouvrir une séquence vidéo stockée dans une table de base de données. L'application doit préalablement se connecter à la base de données. En outre, l'extension Vidéo copie la séquence vidéo dans un fichier temporaire. Ce fichier est créé dans le répertoire désigné par la variable d'environnement client DB2VIDEOTEMP. L'ouverture d'une séquence vidéo initialise la fonction de détection de prises de vue. L'extension Vidéo définit un pointeur orienté sur le début de la séquence (image 0).

Lorsque vous utilisez l'une de ces API, vous devez pointer sur une zone contenant le pointeur indiquant la structure de données vidéo (DBvIOType). L'extension Vidéo alloue cette structure en réponse à l'appel de l'API et l'utilise pour stocker des informations relatives à la séquence vidéo. Cette structure désigne également la structure de données d'image (DBvFrameData) contenant la composition en pixels de l'image en cours. Pour obtenir une description de ces structures, reportez-vous à la section «Structure des données de détection de prises de vue» à la page 182. Pour l'API

DBvOpenFile, vous devez aussi indiquer le nom du fichier vidéo. En ce qui concerne l'API DBvOpenHandle, vous devez préciser le descripteur vidéo.

Par exemple, les instructions suivantes permettent d'ouvrir un objet vidéo stocké dans un fichier en vue de la détection de prises de vue :

```
DBvIOType    *videoptr;

rc=DBvOpenFile (
       &videoptr,                          /* pointeur sur la structure vidéo */
       "/Employés/vidéo/rdurand.mpg");  /* fichier vidéo */
```

Les instructions suivantes permettent d'ouvrir un objet vidéo stocké dans une table de base de données en vue de la détection de prises de vue :

```
EXEC SQL BEGIN DECLARE SECTION;
char Vid_hdl[251];
  EXEC SQL END DECLARE SECTION;

DBvIOType    *videoptr;

EXEC SQL SELECT VIDEO INTO :Vid_hdl
       FROM EMPLOYEE
WHERE NAME="Anne Dupont";

rc=DBvOpenHandle(
       &videoptr,                          /* pointeur sur la structure vidéo */
       vid_hdl);                           /* descripteur vidéo */
```

Indexation d'un objet vidéo

L'extension Vidéo utilise un index pour accéder aux images et aux prises de vue d'une séquence vidéo. Cet index doit être créé préalablement et permet d'extraire les images ou les prises de vue (le format MPEG ne fournit pas d'index pour les images et les prises de vue). L'index met les numéros d'images en correspondance avec les flux de bits composant une séquence vidéo MPEG-1.

Vous pouvez créer un index pour une séquence vidéo à l'aide de l'API DBvCreateIndexFromVideo API ou DBvCreateIndex. Cependant, si vous avez ouvert la séquence vidéo pour détecter les prises de vue en utilisant l'API DBvOpenFile ou DBvOpenHandle, il n'est pas nécessaire de créer explicitement un index ; l'extension Vidéo en crée automatiquement un (pour savoir comment ouvrir une séquence vidéo, reportez-vous à la section «Ouverture d'une séquence vidéo pour la détection de prises de vue» à la page 188).

Lorsqu'un index est créé (explicitement ou automatiquement), l'extension Vidéo DB2 tente de stocker l'index dans le même chemin que le fichier vidéo. Il commence par tenter de stocker le fichier index en tant que fname.ext.idx, fname étant le nom du fichier vidéo et ext son extension. Si cette tentative

échoue, l'extension Vidéo tente de stocker le fichier en tant que fname.idx dans le même répertoire que celui du fichier vidéo. Si cette tentative échoue, il tente de stocker le fichier index dans le répertoire local, d'abord en tant que fname.ext.idx, puis en tant que fname.idx.

A l'ouverture du fichier, l'extension Vidéo cherche le fichier index dans l'ordre suivant :

1. Priorité de la version en écriture sur une version en lecture seulement.
2. Priorité d'un fichier index situé dans le même chemin que celui du fichier vidéo sur un fichier index situé dans le répertoire en cours.
3. Priorité d'un index dont le nom estfname.ext.idx sur un index dont le nom est fname.idx (fname correspond au nom du fichier vidéo et ext à son l'extension).

Par exemple, si un index est créé pour un fichier vidéo dont le nom est myvideo.mpg, l'extension Vidéo recherche en premier lieu un index en écriture dont le nom est myvideo.mpg.idx dans le même chemin que celui du fichier vidéo.

Lorsque vous utilisez l'API DBvCreateIndexFromVideo, précisez la structure de données DBvIOType. L'extension Vidéo stocke le nom du fichier d'index dans la structure. Pour une description de cette structure, reportez-vous à la section «Structure des données de détection de prises de vue» à la page 182. Par exemple, les instructions suivantes permettent de créer un index pour une séquence vidéo qui a été ouverte auparavant pour une détection de prises de vue :

```
DBvIOType    *video;

rc=DBvCreateIndexFromVideo(
       video);                  /* pointeur sur la structure vidéo */
```

Lorsque vous utilisez l'API DBvCreateIndex, précisez le nom du fichier vidéo. L'extension Vidéo stocke l'index dans un fichier (sous le même répertoire que la séquence vidéo). Par exemple, les instructions suivantes permettent de créer un index pour un fichier vidéo (ce dernier n'a pas été précédemment ouvert pour une détection de prises de vue) :

```
rc=DBvCreateIndex(
       "/Employés/vidéo/rdurand.mpg");  /* fichier vidéo */
```

Il est également possible de déterminer s'il existe un index pour une séquence vidéo. Utilisez pour cela l'API DBvIsIndex. L'API définit une variable d'état par 0 s'il n'existe pas d'index, et par 1 s'il en existe un. Par exemple, les instructions suivantes permettent de déterminer s'il existe un index pour une séquence vidéo :

```
      short *status

rc=DBvIsIndex(
       "/Employés/vidéo/rdurand.mpg"    /* fichier vidéo */
       &status);                         /* indicateur d'état */
```

Sauvegarde de l'index vidéo : Sauvegardez le fichier d'index vidéo au cas où vous auriez besoin de le récupérer. Ce fichier se trouve sous le même répertoire que l'extension Vidéo.

Extraction d'une image

Vous pouvez extraire l'image en cours d'une séquence vidéo ou définir l'image en cours par un numéro d'image spécifique. Utilisez l'API DBvGetFrame pour extraire l'image en cours d'une séquence vidéo. Pour définir l'image en cours par un numéro d'image spécifique, utilisez l'API DBvSetFrameNumber.

Lorsque vous utilisez l'API DBvGetFrame, précisez la structure vidéo. Par exemple, les instructions suivantes permettent d'extraire l'image en cours d'une séquence vidéo :

```
DBvIOType    *video;

rc=DBvGetFrame(
       video);                  /* pointeur sur la structure vidéo */
```

Lorsque vous utilisez l'API DBvSetFrameNumber, précisez la structure vidéo et le numéro de l'image que vous voulez définir comme image en cours. Par exemple, les instructions suivantes permettent de définir le numéro 85 pour l'image en cours, puis d'extraire cette dernière :

```
DBvIOType    *video;

rc=DBvSetFrameNumber(
       video,              /* pointeur sur la structure vidéo */
       85);                     /* numéro de l'image */

rc=DBvGetFrame(
       video);                  /* pointeur sur la structure vidéo */
```

En sortie, l'API DBvSetFrameNumber réinitialise la zone currentFrame de la structure DBvIOType. L'API DBvGetFrame place le contenu en pixels de l'image dans la structure DBvFrameData. Pour obtenir une description de ces structures, reportez-vous à la section «Structure des données de détection de prises de vue» à la page 182.

Extraction d'une prise de vue

Pour extraire la prise de vue suivante d'une séquence vidéo, utilisez l'API DBvDetectShot. Lorsque vous utilisez cette API, vous devez pointer sur les structures de données suivantes :

- Vidéo (DBvIOType)
- Contrôle de prise de vue (DBvShotControl)
- Type de prise de vue (DBvShotType)

Vous devez également pointer sur une image à laquelle doit commencer la recherche. L'extension Vidéo commence la recherche de la prise de vue suivante à partir de ce point de la séquence vidéo.

En sortie de l'API, l'extension Vidéo définit un indicateur shotDetected et pointe sur la première image de la prise de vue suivante, ainsi que sur ses données. Lorsque l'indicateur shotDetected est égal à 1, une prise de vue a été détectée. Dans ce cas, l'extension Vidéo :

- définit la zone currentFrame de DBvIOType par la première image de la prise de vue suivante ;
- place les données de la première image de la prise de vue suivante dans la zone fd de DBvIOType ;
- définit DBvShotType afin qu'il contienne le numéro de la première image, celui de la dernière image, le numéro et les données de l'image représentative et les commentaires relatifs à la prise de vue suivante.

Lorsque l'indicateur shotDetected est égal à 0, aucune prise de vue n'a été détectée. Dans ce cas, l'extension Vidéo renvoie un code indiquant que la fin de la séquence vidéo a été atteinte.

Pour obtenir une description de ces structures, reportez-vous à la section «Structure des données de détection de prises de vue» à la page 182.

Par exemple, les instructions suivantes permettent de rechercher la prise de vue suivante d'une séquence vidéo :

```
DBvIOType    *video;
long start_frame = 1;
char shotDetected = 0;
DBvShotControl    shotCtrl;
DBvShotType    shot;

shotCtrl→method=DETECT_CORRHIST
shotCtrl→normalCorrValue=60;
shotCtrl→sceneCutSkipXY=1;
shotCtrl→CorrHistThresh=10;
shotCtrl→DissThresh=10;
shotCtrl→DissCacheSize=4;
shotCtrl→DissNumCaches=7;
shotCtrl→minShotSize=0;

rc=DBvDetectShot(
       video,              /* pointeur sur la structure vidéo */
       start_frame,        /* première image pour la recherche */
       &shotDetected,      /* indicateur de détection de prises de vue */
                           /* 1 = détecté, 0 = non détecté */
       shotCtrl,           /* pointeur sur structure de contrôle de prise de vue */
       &shot);             /* pointeur sur la structure de type de prise de vue */
```

Conversion du format d'une image extraite

Le contenu d'une image MPEG-1 est au format YUV. Ce format comprend des informations sur le plan des pixels de luminance, des pixels Cr et des pixels Cb d'une image. Pour modifier une image vidéo, il peut être utile de convertir l'image du format YUV au format RGB. L'extension Vidéo fournit l'API DBvFrameDatato24BitRGB qui permet de convertir une image extraite MPEG-1 du format YUV au format RGB 24 bits. Pour utiliser l'API, vous devez tout d'abord allouer une mémoire tampon cible.

Lorsque vous émettez l'API, vous devez pointer sur la mémoire tampon cible et les données d'image à convertir. Vous devez aussi indiquer la hauteur et la largeur de l'image. Toutes ces données peuvent être extraites de la structure DBvIOType correspondante. Par exemple, les instructions suivantes permettent de convertir une image MPEG-1 au format RGB 24 bits :

```
char RGB[18000];
DBvIOType    *video;
DBvFrameData   fd;

rc=DBvGetNextFrame(
       video);                  /* pointeur sur la structure vidéo */

fd=video.fd
dx=video.dx
dy=video.dy

rc=DBvFrameDataTo24BitRGB (
       RGB,                      /* pointeur sur la mémoire tampon cible */
       &fd,                      /* pointeur sur les données d'image */
       dx,                       /* largeur de l'image */
       dy);                      /* hauteur de l'image */
```

Fermeture d'un fichier vidéo

L'API DBvClose permet de fermer un fichier vidéo qui a été ouvert pour une détection de prises de vue. Lorsque vous utilisez cette API, vous devez définir un pointeur sur la structure vidéo du fichier.

Par exemple, les instructions suivantes permettent de fermer un fichier vidéo ouvert pour une détection de prises de vue :

```
DBvIOType    *video;

rc=DBvClose (video);
```

Affichage d'une image extraite

Le contenu d'une image MPEG-1 extraite est au format YUV que la plupart des programmes d'affichage d'images ne peuvent pas afficher. Vous devez donc le convertir dans un format pouvant être affiché, tel que le format BMP. Par exemple, pour afficher une image MPEG-1, vous pouvez procéder de la manière suivante :

1. Utilisez l'API DBvFrameDatato24BitRGB pour convertir une image MPEG-1 extraite du format YUV au format RGB 24 bits. Pour plus d'informations sur l'utilisation de l'API DBvFrameDatato24BitRGB, reportez-vous à la section «Conversion du format d'une image extraite» à la page 193.
2. Ajoutez un en-tête approprié à l'image convertie. Par exemple, le format BMP nécessite un en-tête comprenant des informations telles que la largeur et la hauteur de l'image.
3. Copiez le contenu de l'image (avec son en-tête) dans un fichier.
4. Utilisez ensuite l'API DBiBrowse pour afficher le contenu du fichier. Pour plus d'informations, reportez-vous à la section «Utilisation des API d'affichage ou de lecture» à la page 131.

Catalogage des prises de vue

Vous pouvez stocker des informations relatives à une prise de vue dans un catalogue. L'extension Vidéo fournit des API qui permettent de :

- Créer et gérer un catalogue des prises de vue dans une base de données. Vous utilisez les API pour :
  - créer un catalogue des prises de vue dans une base de données,
  - stocker dans un catalogue des informations relatives à une prise de vue unique,
  - stocker dans un catalogue des informations relatives à toutes les prises de vue d'une séquence vidéo,
  - modifier les informations stockées relatives à une prise de vue dans un catalogue,
  - fusionner les informations de prises de vue dans un catalogue,
  - supprimer des informations de prises de vue d'un catalogue,
  - supprimer un catalogue des prises de vue d'une bases de données,
- Créer un catalogue des prises de vue et y stocker des informations relatives à toutes les prises de vue d'une séquence vidéo. Une API est fournie. Elle crée le fichier catalogue et y inclut les données de prises de vue. Vous pouvez accéder aux données du fichier catalogue des prises de vue et manipuler ces données. Toutefois, aucune API n'est fournie à cet effet.

Les prises de vue cataloguées fournissent les entrées des storyboards : Une fois que vous avez stocké des informations de prises de vue dans un catalogue (qu'il s'agisse d'une base de données ou d'un fichier), vous pouvez utiliser ces informations dans une application associée aux prises de vue. Par exemple, vous pouvez créer une application qui extrait des images représentatives de toutes les prises de vue d'une séquence vidéo et les affiche dans un storyboard.

Vous ne devez créer un catalogue des prises de vue que pour une base de données : Vous devez créer un catalogue des prises de vue uniquement si vous souhaitez que ce dernier réside dans une base de données. L'extension Vidéo crée automatiquement un fichier catalogue des prises de vue lorsque vous stockez les données des prises de vue d'une séquence vidéo et indiquez que vous souhaitez que la sortie soit placée dans un fichier.

Préalables à la création d'un catalogue (base de données uniquement)

Avant de créer et d'utiliser un catalogue dans une base de données, vous devez effectuer les opérations suivantes :

- Emettre un appel SQLConnect. Un catalogue stocké dans une base de données DB2 est constitué d'un ensemble de tables. Avant de créer un catalogue des prises de vue dans une base de données ou d'exécuter des opérations sur ce dernier, vous devez vous connecter à la base de données via un appel SQLConnect. (SQLConnect est un appel DB2 CLI). L'appel renvoie alors un descripteur de connexion que vous devez indiquer dans l'API qui gère le catalogue des prises de vue.
- Activer la base de données pour les données d'image. Vous devez activer une base de données pour le type de données DB2Image avant de créer un catalogue des prises de vue dans cette base de données. Outre les diverses informations qu'elle stocke dans le catalogue des prises de vue, l'extension Vidéo stocke des données d'image représentative pour chaque prise de vue cataloguée. Les données de l'image représentative sont de type DB2Image.

Création d'un catalogue des prises de vue (base de données uniquement)

Utilisez l'API DBvCreateShotCatalog pour créer un catalogue des prises de vue dans une base de données. L'extension Vidéo crée automatiquement un fichier catalogue des prises de vue lorsque vous stockez les données des prises de vue et indiquez que vous souhaitez que la sortie soit placée dans un fichier. Le catalogue se compose de tables contenant des informations relatives aux prises de vue. Il est possible de faire porter une requête sur une vue des tables à l'aide de SQL. Le tableau 13 présente un exemple de colonnes de ce type de vue.

Tableau 13. Colonnes d'une vue de catalogue des prises de vue

Nom de la colonne	Type de données	Description
SHOTHANDLE	CHAR(36)	Descripteur de prise de vue
VIDEOHANDLE	VARCHAR(254)	Descripteur vidéo. Cette colonne ne contient une valeur que si l'objet vidéo est ouvert par l'API DBvOpenHandle.

Tableau 13. Colonnes d'une vue de catalogue des prises de vue (suite)

Nom de la colonne	Type de données	Description
VIDEOTABLE	VARCHAR(254)	Table contenant l'objet vidéo. Cette colonne ne contient une valeur que si l'objet vidéo est ouvert par l'API DBvOpenHandle.
VIDEOCOLUMN	VARCHAR(254)	Colonne de la table contenant l'objet vidéo. Cette colonne ne contient une valeur que si l'objet vidéo est ouvert par l'API DBvOpenHandle.
VIDEOFILE	VARCHAR(254)	Nom du fichier vidéo. Cette colonne ne contient une valeur que si l'objet vidéo est ouvert par l'API DBvOpenFile.
STARTFRAME	INTEGER	Numéro de la première image
ENDFRAME	INTEGER	Numéro de la dernière image
REPFRAME	INTEGER	Numéro de l'image représentative
REPFRAMEDATA	DB2IMAGE	Données de l'image représentative
COMMENTS	LONG VARCHAR	Commentaire

Vous avez le choix du nombre de catalogues des prises de vue créés dans une base de données et des prises de vue pour lesquelles vous stockez des informations dans chaque catalogue. Vous pouvez créer un catalogue pour les informations concernant plusieurs séquences vidéos, stocker les informations relatives à chaque séquence vidéo dans des catalogues séparés ou stocker les informations concernant plusieurs prises de vue d'une même séquence vidéo dans plusieurs catalogues.

Lorsque vous utilisez l'API, vous devez indiquer le nom du catalogue. Les noms comportant plus de 16 caractères sont tronqués. Vous devez également préciser le descripteur de connexion à la base de données renvoyé par l'appel SQLConnect à la base de données. Par exemple, les instructions suivantes permettent de créer un catalogue appelé cinéma :

```
SQLHDBC      hdbc;

rc = SQLConnect(hdbc,"cinéma",SQL_NTS,id,SQL_NTS,password,SQL_NTS);
```

```
rc=DBvCreateShotCatalog(
                "cinéma",       /* nom du catalogue des prises de vue */
       hdbc);                   /* descripteur de connexion à la base de données */
```

Les vues du catalogue des prises de vue sont appelées MMDBSYS.SV*nomcat*, (*nomcat* est le nom du catalogue des prises de vue). Par exemple, une vue du catalogue ″cinéma″ s'appelle MMDBSYS.SVCINEMA.

Stockage d'informations relatives à une prise de vue unique (base de données uniquement)

L'API DBvInsertShot permet de stocker des informations relatives à une prise de vue unique dans un catalogue. Vous ne pouvez stocker des informations relatives à une prise de vue d'une séquence vidéo que si le catalogue des prises de vue est stocké dans une base de données. Ces informations comprennent les éléments suivants :

- Descripteur de prise de vue
- Nom de la table vidéo (pour les séquences vidéo stockées dans une table)
- Nom de la colonne vidéo (pour les séquences vidéo stockées dans une table)
- Descripteur vidéo (pour les séquences vidéo stockées dans une table)
- Nom du fichier vidéo (pour les séquences vidéo stockées dans un fichier)
- Numéro de la première image
- Numéro de la dernière image
- Numéro de l'image représentative
- Données de l'image représentative

Aucun commentaire concernant la prise de vue n'est stocké. Pour savoir comment ajouter un commentaire aux informations de prise de vue stockées, reportez-vous à la section «Spécification d'un commentaire relatif à une prise de vue (base de données uniquement)» à la page 202.

Lorsque vous utilisez l'API DBvInsertShot, vous devez définir le nom du catalogue des prises de vue et un pointeur sur la prise de vue. Pour positionner le pointeur sur la prise de vue, vous pouvez extraire la prise de vue suivante, comme indiqué à la section «Extraction d'une prise de vue» à la page 191. Vous devez également préciser le descripteur de connexion à la base de données renvoyé par l'appel SQLConnect à la base de données. Par exemple, les instructions suivantes permettent d'extraire la prise de vue qui suit l'image 1 et de stocker des informations sur la prise de vue dans un catalogue appelé cinéma :

```
SQLHDBC      hdbc;
SQLHENV  henv;
DBvIOType    *video;
long start_frame = 1;
char shotDetected = 0;
DBvShotControl    shotCtrl;
```

```
DBvShotType    shot;

shotCtrl→method=DETECT_CORRHIST
shotCtrl→normalCorrValue=60;
shotCtrl→sceneCutSkipXY=1;
shotCtrl→CorrHistThresh=10;
shotCtrl→DissThresh=10;
shotCtrl→DissCacheSize=4;
shotCtrl→DissNumCaches=7;
shotCtrl→minShotSize=0;

SQLAllocConnect(henv,&hdbc)

rc = SQLConnect(hdbc,"cinéma",SQL_NTS,id,SQL_NTS,password,SQL_NTS);

rc=DBvDetectShot(
                vidéo,
       start_frame,
       &shotDetected,
       &shotCtrl,
       &shot)

rc=DBvInsertShot (
        "cinéma",                      /*nom du catalogue des prises de vue*/
       shot,                        /*pointeur sur la prise de vue*/
        hdbc);                     /*descripteur de connexion à la base*/
                                   /*de données*/
```

Stockage d'informations relatives à toutes les prises de vue d'une séquence vidéo

Les API DBvBuildStoryboardTable ou DBvBuildStoryboardFile permettent de stocker dans un catalogue des informations relatives à toutes les prises de vue d'une séquence vidéo. L'API DBvBuildStoryboardTable stocke les informations dans un catalogue des prises de vue résidant dans une base de données. L'API DBvBuildStoryboardFile crée un fichier catalogue des prises de vue et y stocke les informations associées.

Pour ces deux API, la séquence vidéo source peut se trouver dans une table de base de données ou dans un fichier.

Lorsque vous utilisez l'une de ces API, vous devez :

- indiquer le nom du catalogue des prises de vue ;
- pointer sur la structure vidéo ;
- pointer sur la structure de données DBvShotControl ;
- pointer sur la structure de données DBvStoryboardCtrl. Les valeurs contenues dans cette structure contrôlent le choix et le nombre d'images vidéo qui seront stockées comme images représentatives dans le catalogue des prises de vue. Pour plus de détails sur la définition de ces valeurs, reportez-vous à la section «Création d'un storyboard» à la page 199.

Pour l'API DBvBuildStoryboardTable uniquement, vous devez aussi préciser le descripteur de connexion à la base de données renvoyé par l'appel SQLConnect à la base de données.

Par exemple, les instructions suivantes permettent de stocker les informations relatives à toutes les prises de vue d'une séquence vidéo dans un catalogue. Ce dernier se trouve dans une base de données.

```
SQLHDBC      hdbc;
SQLHENV  henv;
DBvIOType    *video;
DBvShotControl    shotCtrl;
DBvStoryBoardCtrl    sbCtrl;

sbCtrl→thresh1=50
sbCtrl→thresh2=500;
sbCtrl→delta=20;

SQLAllocConnect(henv,&hdbc)

rc = SQLConnect(hdbc,"cinéma",SQL_NTS,id,SQL_NTS,password,SQL_NTS);

rc=DBvBuildStoryboardTable (
        "cinéma",                       /*nom du catalogue des prises de vue*/
       video,                       /*pointeur sur la structure vidéo*/
       shotCtrl,                    /*pointeur sur la structure de contrôle */
                                    /*de prises de vue*/
       sbctrl,                      /*pointeur sur la structure de storyboard*/
        hdbc);                      /*descripteur de connexion à la base*/
                                    /*de données*/
```

Les instructions suivantes permettent de créer un fichier catalogue des prises de vue et d'y stocker les informations relatives à toutes les prises de vue d'une séquence vidéo :

```
DBvIOType    *video;
DBvShotControl    shotCtrl;
DBvStoryBoardCtrl    sbCtrl;

sbCtrl→thresh1=50
sbCtrl→thresh2=500;
sbCtrl→delta=20;

rc=DBvBuildStoryboardFile (
       "cinéma",            /*nom du catalogue des prises de vue*/
       video,               /*pointeur sur la structure vidéo*/
       shotCtrl,            /*pointeur sur la structure de contrôle */
                            /*de prises de vue*/
       sbctrl);             /*pointeur sur la structure de contrôle de storyboard*/
```

Création d'un storyboard

Comme leur nom l'indique, les API DBvBuildStoryboardTable et DBvBuildStoryboardFile sont particulièrement utiles pour le stockage d'informations qui seront utilisées dans un storyboard. Un **storyboard** est une synopsis visuelle d'une séquence vidéo. Vous créez un storyboard en affichant les images représentatives stockées pour une séquence vidéo dans un catalogue des prises de vue.

Utilisation de changements de plan

Les API DBvBuildStoryboardTable API et DBvBuildStoryboardFile permettent de stocker une ou plusieurs images représentatives d'une prise de vue. Les valeurs que vous indiquez dans la structure DBvStoryboardCtrl contrôlent le nombre d'images représentatives stockées pour une prise de vue, ainsi que le choix des images qui seront utilisées. Pour connaître la définition de la structure DBvStoryboardCtrl, reportez-vous à la section «Structure des données de détection de prises de vue» à la page 182. La figure 29 indique comment sont utilisées les valeurs des zones DBvStoryboardCtrl.

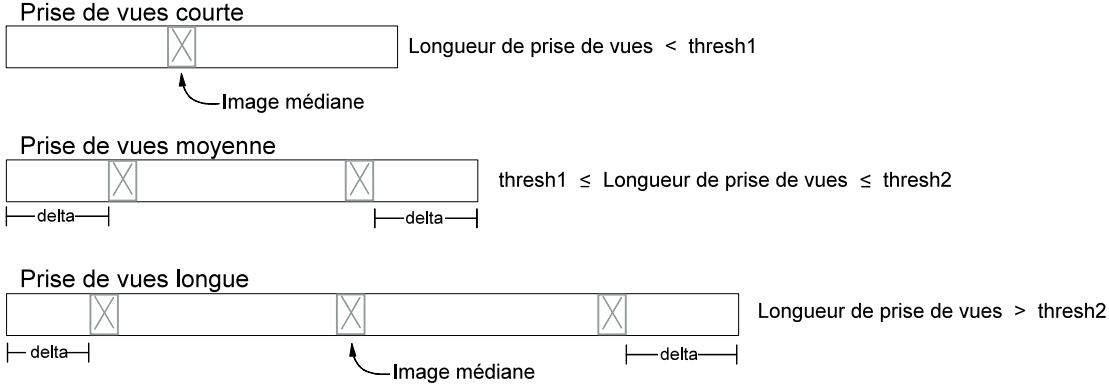


Figure 29. Utilisation des valeurs de la structure DBvStoryboardCtrl

Comme l'illustre la figure 29 :

- Une seule image représentative est stockée pour une prise de vue courte. Le nombre d'images d'une prise de vue courte est inférieur à la valeur de thresh1 dans la structure de données DBvStoryboardCtrl. L'image représentative est l'image médiane de la prise.
- Deux images représentatives sont stockées pour une prise de vue moyenne. Le nombre d'images d'une prise de vue moyenne est supérieur ou égal à la valeur de thresh1 et inférieur ou égal à la valeur de thresh2 dans la structure de données DBvStoryboardCtrl. La valeur de la zone delta dans la structure de données DBvStoryboardCtrl détermine le nombre d'images comprises entre le début de la séquence vidéo et la première image représentative. Cette valeur détermine également le nombre d'images comprises entre la deuxième image représentative et la fin de la séquence vidéo.
- Trois images représentatives sont stockées pour une prise de vue longue. Le nombre d'images d'une prise de vue longue est supérieur à la valeur de thresh2 dans la structure de données DBvStoryboardCtrl. La valeur de la zone delta dans la structure de données DBvStoryboardCtrl détermine le nombre d'images comprises entre le début de la séquence vidéo et la première image représentative. La deuxième image représentative est

l'image médiane de la séquence vidéo. La distance de la troisième image représentative par rapport à la fin de la vidéo est déterminée par la valeur de la zone delta.

Toute prise de vue peut être traitée comme une prise courte si la valeur de thresh1 ou de thresh2 est définie par -1. Dans ce cas, une seule image représentative, l'image médiane, est stockée pour la prise de vue dans le catalogue.

Outre les valeurs contenues dans la structure de données DBvStoryboardCtrl, plusieurs zones de la structure DBvShotControl ont une incidence sur le choix d'images représentatives stockées en vue de leur affichage ultérieur dans un storyboard. Les zones CorrHistThresh, normalcorrValue et minShotSize de la structure de données DBvShotControl, par exemple, indiquent des seuils pour la détection de prises de vue. Elles influent donc sur le choix d'images qui seront affichées dans le storyboard d'une séquence vidéo. Lorsque vous utilisez les API DBvBuildStoryBoardTable et DBvBuildStoryBoardFile pour stocker des informations de prises de vue qui seront utilisées dans un storyboard, vous pouvez lancer un essai en utilisant les valeurs initiales des structures de données DBvStoryBoardCtrl et DBvShotControl. Vous affinerez ensuite les résultats en modifiant les valeurs des diverses zones de ces structures.

Affichage d'un storyboard

Vous pouvez créer un programme permettant d'afficher un storyboard. Pour cela, vous accédez aux images représentatives stockées dans un catalogue des prises de vue pour une séquence vidéo. Si l'API DBvBuildStoryboardFile a été utilisée pour stocker les prises de vue de la séquence vidéo, le fichier catalogue des prises de vue pointe sur les fichiers GIF des images représentatives. Ces fichiers peuvent être visualisés via un afficheur ou un programme d'affichage, selon les cas.

Si l'API DBvBuildStorybBoardTable a été utilisée pour stocker les prises de vue de la séquence vidéo, le catalogue des prises de vue (stocké dans une base de données) contient les données des images représentatives. Vous pouvez accéder à ces dernières dans la vue du catalogue des prises de vue (pour une description de la vue, reportez-vous au tableau 13 à la page 195). Les données de l'image représentative sont au format YUV que la plupart des programmes d'affichage d'images ne peuvent pas exploiter. Pour afficher les images représentatives, vous pouvez convertir les données de l'image à l'aide de l'API DBvFrameDatato24BitRGB, comme indiqué à la section «Affichage d'une image extraite» à la page 193. Ces images peuvent ensuite être visualisées via un afficheur ou un programme d'affichage, selon les cas.

Modèles de programmes de storyboard

Le sous-répertoire SAMPLES contient deux modèles de programmes qui indiquent comment créer et afficher un storyboard pour une vidéo. L'un de ces modèles, qui se trouve dans le fichier `makesf.exe`, utilise l'API DBvBuildStoryBoardFile pour créer un fichier catalogue des prises de vue et y stocker des données de prises de vue. L'autre modèle de programme, `makehtml.exe`, accède au fichier catalogue et crée des pages HTML qui pourront être affichées par un afficheur Web.

Spécification d'un commentaire relatif à une prise de vue (base de données uniquement)

Vous pouvez préciser un commentaire relatif à une prise de vue, qui est stocké avec d'autres informations connexes dans un catalogue des prises de vue. Pour cela, utilisez l'API DBvSetShotComment.

Lorsque vous utilisez cette API, vous devez préciser le nom du catalogue des prises de vue dans lequel sera stocké le commentaire, le descripteur de la prise de vue à laquelle correspondent le commentaire et le commentaire lui-même. Vous devez également préciser le descripteur de connexion à la base de données renvoyé par l'appel SQLConnect à la base de données. Par exemple, les instructions suivantes permettent d'ajouter un commentaire relatif à une prise de vue (qui commence à l'image n°85) dans un catalogue appelé cinéma :

```
SQLHDBC      hdbc;
SQLHENV  henv;
char shothandle[37];

SQLAllocConnect(henv,&hdbc)

rc = SQLConnect(hdbc,"cinéma",SQL_NTS,id,SQL_NTS,password,SQL_NTS);

EXEC SQL SELECT SHOTHANDLE INTO :shothandle
  FROM MMDBSYS.SVHOTSHOTS
  WHERE STARTFRAME=85;

rc=DBvSetShotComment (
        "cinéma",               /*nom du catalogue des prises de vue*/
        shothandle,                     /*descripteur de prise de vue*/
        "plage au coucher du soleil",    /*commentaire*/
        hdbc);                  /*descripteur de connexion à la base*/
                                /*de données*/
```

Modification des informations stockées relatives à une prise de vue (base de données uniquement)

Il est possible de modifier les informations relatives à une prise de vue stockées dans un catalogue. Pour cela, utilisez l'API DBvUpdateShot. Placez les nouvelles informations dans une structure DBvShotType. Vous devez également compléter les zones restantes, même si elles ne sont pas modifiées. Lorsque vous utilisez l'API DBvUpdateShot, précisez le nom du catalogue et

un pointeur sur la structure DBvShotType. Vous devez également préciser le descripteur de connexion à la base de données renvoyé par l'appel SQLConnect à la base de données.

Il est possible de modifier le commentaire (le cas échéant) en même temps que les informations relatives à une prise de vue en l'indiquant dans la structure DBvShotType. Pour conserver l'ancien commentaire, indiquez une valeur nulle dans la structure DBvShotType.

Par exemple, les instructions suivantes permettent de modifier les informations relatives à une prise de vue stockées dans un catalogue appelé cinéma (la prise de vue commence à l'image n°85) :

```
SQLHDBC      hdbc;
SQLHENV  henv;
char shothandle[37];
DBvShotType    shot;
DBvFrameData  fd110;

/* extraction du descripteur de prises de vue */

EXEC SQL SELECT SHOTHANDLE INTO :shothandle
  FROM MMDBSYS.SVHOTSHOTS
  WHERE STARTFRAME=85;

/* modification des attributs de prise de vue */

shot.startFrame=110;
shot.endFrame=200;
shot.repframe=110;
shot.fd=fd110;
shot.comment=NULL;

/* mise à jour des informations de prise de vue */

SQLAllocConnect(henv,&hdbc)

rc = SQLConnect(hdbc,"cinéma",SQL_NTS,id,SQL_NTS,password,SQL_NTS);

rc=DBvUpdateShot (
        "cinéma",               /*nom du catalogue des prises de vue*/
        shot,                           /*informations de prise de vue*/
        hdbc);                  /*descripteur de connexion à la base*/
                                /*de données*/
```

Fusion d'informations de prises de vue dans un catalogue (base de données uniquement)

Il est possible de fusionner les informations relatives à deux prises de vue dans un catalogue. Lorsque vous effectuez cette opération, vous indiquez un ordre de fusion en identifiant la première et la seconde prises de vue. Le numéro de la première image de la première prise de vue est stocké en tant que numéro de la première image de la prise de vue fusionnée. Le plus grand

numéro d'image des deux prises de vue est stocké en tant que numéro de la dernière image de la prise de vue fusionnée. Lors d'une fusion, les informations de la première prise de vue sont remplacées par celles de la prise de vue fusionnée et les informations de la seconde prise de vue sont supprimées du catalogue.

Utilisez l'API DBvMergeShots pour fusionner les informations de deux prises de vue dans un catalogue. Vous devez indiquer le nom du catalogue suivi des descripteurs de la première et de la seconde prises de vue à fusionner. Vous devez également préciser le descripteur de connexion à la base de données renvoyé par l'appel SQLConnect à la base de données. Par exemple, les instructions suivantes permettent de fusionner les informations de deux prises de vue dans un catalogue appelé cinéma (la première prise commence à l'image 85 et la seconde à l'image 210) :

```
SQLHDBC      hdbc;
SQLHENV  henv;
char shothandle1[37];
char shothandle2[37];

EXEC SQL SELECT SHOTHANDLE INTO :shothandle1
  FROM MMDBSYS.SVHOTSHOTS1
  WHERE STARTFRAME=85;

EXEC SQL SELECT SHOTHANDLE INTO :shothandle2
  FROM MMDBSYS.SVHOTSHOTS2
  WHERE STARTFRAME=210;

SQLAllocConnect(henv,&hdbc)

rc = SQLConnect(hdbc,"cinéma",SQL_NTS,id,SQL_NTS,password,SQL_NTS);

rc=DBvMergeShots (
        "cinéma",               /*nom du catalogue des prises de vue*/
        shothandle1,                   /*descripteur de la première prise */
        shothandle2,                   /*descripteur de la seconde prise*/
        hdbc);                  /*descripteur de connexion à la base*/
                                /*de données*/
```

Suppression d'informations de prises de vue dans un catalogue (base de données uniquement)

Pour supprimer des informations relatives à une prise de vue dans un catalogue, utilisez l'API DBvDeleteShot. Vous devez indiquer le nom du catalogue suivi du descripteur de prise de vue. Vous devez également préciser le descripteur de connexion à la base de données renvoyé par l'appel SQLConnect à la base de données. Par exemple, les instructions suivantes permettent de supprimer des informations relatives à une prise de vue (qui commence à l'image n° 85) dans un catalogue appelé cinéma :

```
SQLHDBC      hdbc;
SQLHENV  henv;
char shothandle[37];
```

```
EXEC SQL SELECT shothandle INTO :shothandle
  FROM mmdbsys.svcinéma
  WHERE startframe=85;

rc=DBvDeleteShot (
        "cinéma",               /*nom du catalogue des prises de vue*/
        shothandle,                    /*descripteur de prise de vue*/
        hdbc);                  /*descripteur de connexion à la base*/
                                /*de données*/
```

Suppression d'un catalogue des prises de vue (base de données uniquement)

Pour supprimer un catalogue des prises de vue, utilisez l'API DBvDeleteShotCatalog. Vous devez préciser le nom du catalogue à supprimer et le descripteur de connexion à la base de données renvoyé par l'appel SQLConnect à la base de données. Par exemple, les instructions suivantes permettent de supprimer un catalogue appelé cinéma :

```
SQLHDBC      hdbc;
SQLHENV  henv;

rc=DBvDeleteShotCatalog (
        "cinéma",               /*nom du catalogue des prises de vue*/
        hdbc);                  /*descripteur de connexion à la base*/
                                /*de données*/
```

Utilisation de changements de plan

Partie 4. Référence

Chapitre 15. Types UDT et fonctions UDF

Ce chapitre fournit des informations de référence sur les types de données distincts et les fonctions UDF créés par la famille de produits DB2 Extensions.

Schéma

Les extensions utilisent le schéma MMDBSYS pour tous les objets de base de données relationnelle, notamment les types UDT et les fonctions UDF.

Types UDT

Le tableau 14 répertorie et décrit les types de données distincts créés par DB2 Extensions. Il indique également le type de données source DB2 de chaque type de données UDT.

Tableau 14. Types de données distincts créés par la famille de produits DB2 Extensions

UDT	Type de données source	Description
DB2IMAGE	VARCHAR(250)	Descripteur image. Chaîne de caractères de longueur variable contenant des informations permettant d'accéder à un objet image. Les descripteurs image sont stockés dans une colonne de table utilisateur, utilisable par l'extension Image.
DB2AUDIO	VARCHAR(250)	Descripteur audio. Chaîne de caractères de longueur variable contenant des informations permettant d'accéder à un objet audio. Les descripteurs audio sont stockés dans une colonne de table utilisateur, utilisable par l'extension Audio.
DB2VIDEO	VARCHAR(250)	Descripteur vidéo. Chaîne de caractères de longueur variable contenant des informations permettant d'accéder à un objet vidéo. Les descripteurs vidéo sont stockés dans une colonne de table utilisateur, utilisable par l'extension Vidéo.

Fonctions UDF

Cette section fournit des informations de référence sur la famille de produits DB2 Extensions. Les fonctions UDF sont répertoriées par ordre alphabétique.

Pour chaque fonction UDF, les informations suivantes sont fournies :

- Les extensions qui fournissent la fonction UDF ;
- Une brève description ;
- Le fichier (en-tête) d'inclusion de la fonction UDF ;
- La syntaxe SQL de la fonction UDF ;
- Une description des paramètres de la fonction UDF, y compris le type de données ;
- La valeur renvoyée par la fonction UDF, y compris son type de données ;
- Des exemples.

Le tableau 15 répertorie les fonctions UDF et identifie les extensions qui fournissent chaque fonction UDF. Il indique également la page à laquelle vous pouvez trouver plus d'informations sur chaque UDF. Les fonctions UDF de ce tableau peuvent être codées en instructions SQL imbriquées ou en appels DB2 CLI.

Tableau 15. Fonctions UDF de DB2 Extensions

Fonction UDF	Description	Image	Audio	Vidéo	Voir page
AlignValue	Renvoie le nombre d'octets par échantillon dans une séquence audio de type WAVE, ou dans une piste audio d'une vidéo.		x	x	214
AspectRatio	Renvoie le rapport hauteur/largeur de la première piste d'une vidéo MPEG1 et MPEG2.			x	216
BitsPerSample	Renvoie le nombre de bits de données utilisés pour constituer un échantillon d'une séquence audio de type WAVE ou AIFF ou d'une piste audio d'une vidéo.		x	x	217
BytesPerSec	Renvoie la vitesse de transfert de données, en nombre moyen d'octets par seconde, d'une séquence audio de type WAVE.		x		218
Comment	Renvoie ou met à jour un commentaire stocké avec un objet image, audio ou vidéo.	x	x	x	219

Tableau 15. Fonctions UDF de DB2 Extensions (suite)

Fonction UDF	Description	Image	Audio	Vidéo	Voir page
CompressType	Renvoie le format de compression d'une vidéo, tel que MPEG-1.			x	221
Content	Extrait ou met à jour le contenu d'un objet image, audio ou vidéo à partir d'une base de données.	x	x	x	222
DB2Audio	Stocke le contenu d'une séquence audio dans une table de base de données.		x		228
DB2Image	Stocke le contenu d'une image dans une table de base de données.	x			232
DB2Video	Stocke le contenu d'une séquence vidéo dans une table de base de données.			x	237
Duration	Renvoie la durée (temps de lecture en secondes) d'une séquence audio de type WAVE ou AIFF ou d'une vidéo.		x	x	241
Filename	Renvoie le nom du fichier du serveur dans lequel se trouve le contenu d'un objet image, audio ou vidéo.	x	x	x	242
FindInstrument	Renvoie le numéro de piste de la première occurrence d'un instrument spécifique d'une séquence audio de type MIDI.		x		243
FindTrackName	Renvoie le numéro d'une piste nommée spécifique dans une séquence audio de type MIDI.		x		244
Format	Renvoie le format d'un objet image, audio ou vidéo.	x	x	x	245
FrameRate	Renvoie la vitesse de défilement d'une vidéo en images/seconde.			x	246
GetInstruments	Renvoie le nom de tous les instruments d'une séquence audio de type MIDI.		x		247
GetTrackNames	Renvoie le nom de toutes les pistes d'une séquence audio de type MIDI.		x		248

Tableau 15. Fonctions UDF de DB2 Extensions (suite)

Fonction UDF	Description	Image	Audio	Vidéo	Voir page
Height	Renvoie la hauteur, en pixels, d'une image de base de données ou de vidéo.	x		x	249
Importer	Renvoie l'ID de l'utilisateur ayant stocké un objet image audio ou vidéo dans une table de base de données.	x	x	x	250
ImportTime	Renvoie un horodateur indiquant la date de stockage d'un objet image, audio ou vidéo dans une table de base de données.	x	x	x	251
MaxBytesPerSec	Renvoie le débit maximal d'une séquence vidéo en nombre d'octets par seconde.			x	252
NumAudioTracks	Renvoie le nombre de pistes audio d'une vidéo ou d'une séquence audio de type MIDI.		x	x	253
NumChannels	Renvoie le nombre de canaux audio enregistrés dans une séquence audio de type WAVE ou AIFF ou dans une vidéo.		x	x	254
NumColors	Renvoie le nombre de couleurs composant une image.	x			255
NumFrames	Renvoie le nombre d'images composant une séquence vidéo.			x	256
NumVideoTracks	Renvoie le nombre de pistes vidéo d'une séquence vidéo.			x	257
QbScoreFromName	Renvoie le score d'une image (à l'aide d'un objet de requête nommée). (Remplace QbScore.)	x			258
QbScoreFromStr	Renvoie le score d'une image (à l'aide d'une chaîne de requête).	x			260
QbScoreTBFromName	Renvoie une table de scores à partir d'une colonne Image (à l'aide d'un objet de requête nommée).	x			262
QbScoreTBFromStr	Renvoie une table de scores à partir d'une colonne Image (à l'aide d'une chaîne de requête).	x			264

Tableau 15. Fonctions UDF de DB2 Extensions (suite)

Fonction UDF	Description	Image	Audio	Vidéo	Voir page
Replace	Met à jour le contenu d'un objet image, audio ou vidéo stocké dans une base de données, ainsi que le commentaire correspondant.	x	x	x	266
SamplingRate	Renvoie la fréquence d'échantillonnage d'une séquence audio de type WAVE ou AIFF, ou d'une piste audio d'une vidéo, en nombre d'échantillons par seconde.		x	x	270
Size	Renvoie, en octets, la taille d'un objet image, audio ou vidéo.	x	x	x	271
Thumbnail	Renvoie ou met à jour une version miniature d'un objet image ou vidéo stocké dans une base de données.	x		x	272
TicksPerQNote	Renvoie les battements de métronome par noire d'une séquence audio de type MIDI enregistrée.		x		274
TicksPerSec	Renvoie les battements de métronome par seconde d'une séquence audio de type MIDI enregistrée.		x		275
Updater	Renvoie l'ID de l'utilisateur qui a effectué la dernière mise à jour d'un objet image audio ou vidéo dans une table de base de données.	x	x	x	276
UpdateTime	Renvoie un horodateur indiquant la date de la dernière mise à jour d'un objet image, audio ou vidéo dans une table de base de données.	x	x	x	277
Width	Renvoie la largeur en pixels d'un objet image ou vidéo.	x		x	278

AlignValue

Image	Audio	Vidéo
	X	X

Renvoie le nombre d'octets par échantillon dans une séquence audio de type WAVE, ou dans une piste audio d'une vidéo. Les données d'une séquence audio de type WAVE peuvent être enregistrées à l'aide d'un octet par échantillon (séquence mono 8 bits ; on parle «d'alignement sur l'octet»), de deux octets par échantillon (séquence stéréo 8 bits ou mono 16 bits ; on parle «d'alignement sur le mot») ou de quatre octets par échantillon (séquence stéréo 16 bits ; on parle «d'alignement sur le double mot»).

Fichier d'inclusion

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio.

Codes retour (type de données)

Valeur, en octets par échantillon, d'une séquence audio de type WAVE ou d'une piste audio d'une vidéo (SMALLINT). Ces valeurs sont les suivantes :

1 un octet par bloc (alignement sur l'octet)

2 deux octets par bloc (alignement sur le mot)

4 quatre octets par bloc (alignement sur le double mot)

Valeur NULL séquence audio de format différent

Exemples

Extraction du nom de fichier de toutes les séquences audio stockées dans la colonne Son de la table Employés, qui sont définies par deux octets :

```
EXEC SQL BEGIN DECLARE SECTION;
 char hvAud_fname[251];
  EXEC SQL END DECLARE SECTION;
```

```
EXEC SQL SELECT FILENAME (SOUND)
      INTO :hvAud_fname
       FROM EMPLOYEE
      WHERE ALIGNVALUE(SOUND) = 2;
```

Recherche du nombre d'octets par échantillon d'une piste audio d'une séquence vidéo stockée dans la colonne vidéo de la table Employés pour Anne Dupont :

```
EXEC SQL BEGIN DECLARE SECTION;
      short hvAlign_val; EXEC SQL END DECLARE SECTION;
 EXEC SQL SELECT ALIGNVALUE(VIDEO)
      INTO :hvAlign_val
      FROM EMPLOYEE
      WHERE NAME='Anne Dupont';
```

AspectRatio

Image	Audio	Vidéo
		X

Renvoie le rapport hauteur/largeur de la première piste d'une vidéo MPEG.

Fichier d'inclusion
dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence vidéo.

Codes retour (type de données)
Rapport hauteur/largeur de la première piste d'une vidéo de type MPEG, ou valeur nulle pour les vidéos de format différent (SMALLINT).

Exemples
Extraction du rapport hauteur/largeur de la vidéo stockée pour Robert Durand dans la colonne vidéo de la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
      short hvAsp_ratio;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT ASPECTRATIO(VIDEO)
      INTO :hvAsp_ratio
       FROM EMPLOYEE
     WHERE NAME='Robert Durand';
```

BitsPerSample

Image	Audio	Vidéo
	X	X

Renvoie le nombre de bits de données utilisés pour constituer un échantillon d'une séquence audio de type WAVE ou AIFF ou d'une piste audio d'une vidéo.

Fichier d'inclusion

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio ou vidéo.

Codes retour (type de données)

Nombre de bits de données utilisés pour constituer un échantillon de vidéo ou de séquence audio de type WAVE ou AIFF (SMALLINT). Renvoie une valeur nulle pour les séquences audio de format différent.

Exemples

Extraction du nom de fichier de toutes les séquences audio de type WAVE stockées dans la colonne Son de la table Employés dont le nombre de bits par échantillon est égal à 8 :

```
EXEC SQL BEGIN DECLARE SECTION;
 char hvAud_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME (SOUND)
      INTO :hvAud_fname
       FROM EMPLOYEE
      WHERE FORMAT(SOUND)='WAVE'
      AND BITSPERSAMPLE(SOUND) = 8;
```

BytesPerSec

Image	Audio	Vidéo
	X	

Renvoie la vitesse de transfert de données, en nombre moyen d'octets par seconde, d'une séquence audio de type WAVE.

Fichier d'inclusion
dmbaudio.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio.

Codes retour (type de données)
Vitesse de transfert de données (INTEGER). Renvoie une valeur nulle pour les séquences audio de format différent.

Exemples
Extraction du nom de fichier de toutes les séquences audio stockées dans la colonne Son de la table Employés dont la vitesse de transfert de données, en nombre moyen d'octets par seconde, est inférieure à 44100 :

```
EXEC SQL BEGIN DECLARE SECTION;
 char hvAud_fname[251];
EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(SOUND)
      INTO :hvAud_fname
      FROM EMPLOYEE
      WHERE BYTESPERSEC(SOUND) < 44100;
```

Comment

Image	Audio	Vidéo
X	X	X

Renvoie ou met à jour un commentaire stocké avec un objet image, audio ou vidéo.

Fichier d'inclusion

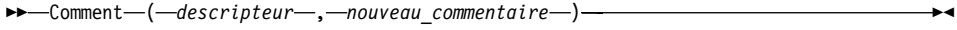
image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe
Extraction

Syntaxe
Mise à jour



Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

nouveau_commentaire (LONG VARCHAR)
Commentaire mis à jour. Une valeur nulle ou une chaîne vide entraîne la suppression du commentaire existant.

Codes retour (type de données)
Pour la mise à jour, le descripteur de l'objet image, audio ou vidéo (DB2IMAGE, DB2AUDIO ou DB2VIDEO). Pour l'extraction, le commentaire (LONG VARCHAR).

Exemples
Extraction du nom de fichier de toutes les images de la colonne Photo de la table Employés auxquelles est associé le commentaire «confidentiel» :

```
EXEC SQL BEGIN DECLARE SECTION;
 char hvImg_fname[255;
EXEC SQL END DECLARE SECTION;
```

```
EXEC SQL SELECT FILENAME(PICTURE)
     INTO :hvImg_fname
     FROM EMPLOYEE
     WHERE COMMENT(PICTURE)
        LIKE '%confidentiel%';
```

Mise à jour du commentaire associé à la séquence vidéo d'Anne Dupont dans la colonne vidéo de la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
struct{
        short len;
        char data [4000];
      }hvRemarks;
  EXEC SQL END DECLARE SECTION;

/* Extraction de l'ancien commentaire */

EXEC SQL SELECT COMMENT(VIDEO)
     INTO :hvRemarks
       FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';

/* Mise à jour du commentaire */

hvRemarks.data[hvRemarks.len]='\0';

strcat (hvRemarks.data, "Nouvelle séquence vidéo");
hvRemarks.len=strlen(hvRemarks.data);

EXEC SQL UPDATE EMPLOYEE
     SET VIDEO=COMMENT(VIDEO, :hvRemarks)
     WHERE NAME = 'Anne Dupont';
```

CompressType

Image	Audio	Vidéo
		X

Renvoie le format de compression d'une vidéo, tel que MPEG-1.

Fichier d'inclusion
dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la vidéo.

Codes retour (type de données)
Format de compression de la vidéo (VARCHAR(8))

Exemples
Extraction des noms de toutes les vidéos stockées dans la colonne vidéo de la table Employés dont le format de compression est MPEG-1 :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvVid_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME (VIDEO)
     INTO :hvVid_fname
       FROM EMPLOYEE
     WHERE COMPRESSTYPE(VIDEO) = 'MPEG1';
```

Content

Image	Audio	Vidéo
X	X	X

Extrait ou met à jour le contenu d'un objet image, audio ou vidéo à partir d'une base de données. Ce contenu peut être extrait vers une mémoire tampon client, un fichier client ou un fichier du serveur.

Fichier d'inclusion

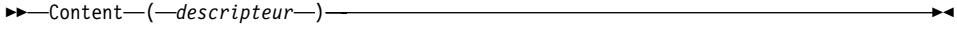
image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

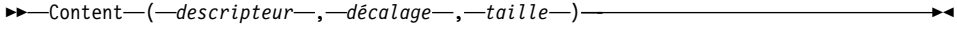
Syntaxe

Extraction du contenu vers une mémoire tampon ou un fichier client



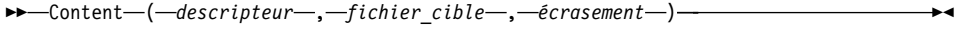
Syntaxe

Extraction d'un segment de contenu vers une mémoire tampon ou un fichier client



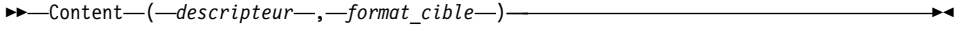
Syntaxe

Extraction du contenu vers un fichier du serveur



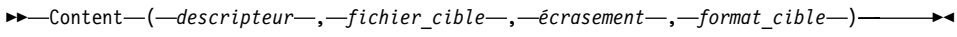
Syntaxe

Extraction du contenu vers une mémoire tampon ou un fichier client avec conversion de fichier (image uniquement)



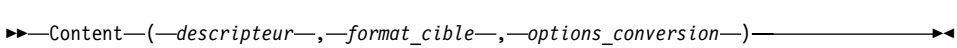
Syntaxe

Extraction du contenu vers un fichier serveur avec conversion de format (image uniquement)



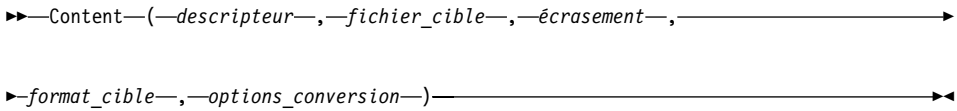
Syntaxe

Extraction du contenu vers une mémoire tampon ou un fichier client avec conversion et modifications supplémentaires (image uniquement)



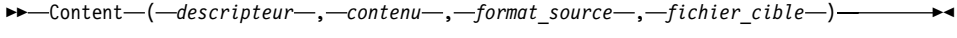
Syntaxe

Extraction du contenu à partir d'un fichier serveur avec conversion de format et modifications supplémentaires (image uniquement)



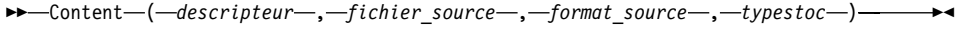
Syntaxe

Mise à jour du contenu à partir d'une mémoire tampon ou d'un fichier client



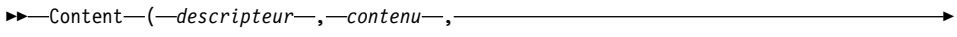
Syntaxe

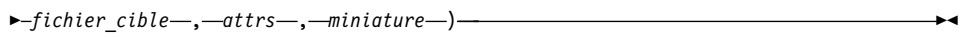
Mise à jour du contenu à partir du fichier serveur



Syntaxe

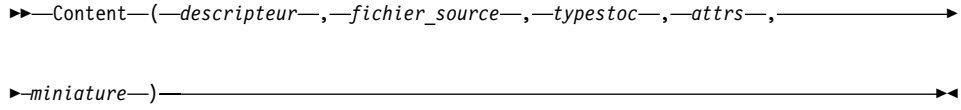
Mise à jour du contenu par des attributs définis par l'utilisateur à partir d'une mémoire tampon ou d'un fichier client





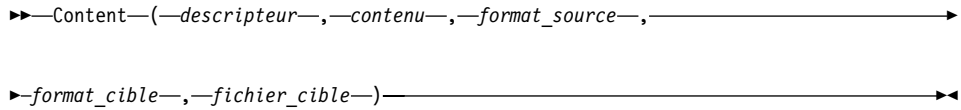
Syntaxe

Mise à jour du contenu par des attributs définis par l'utilisateur à partir d'un fichier serveur



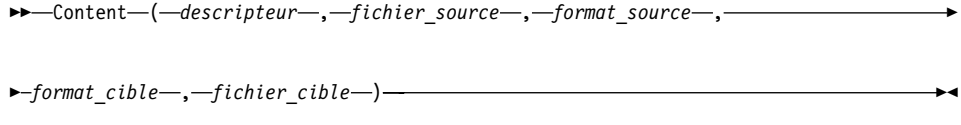
Syntaxe

Mise à jour du contenu à partir d'une mémoire tampon ou d'un fichier client avec conversion de fichier (image uniquement)



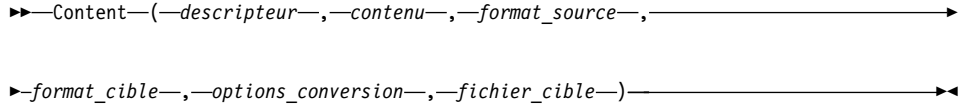
Syntaxe

Mise à jour du contenu à partir du fichier serveur avec conversion de format (image uniquement)



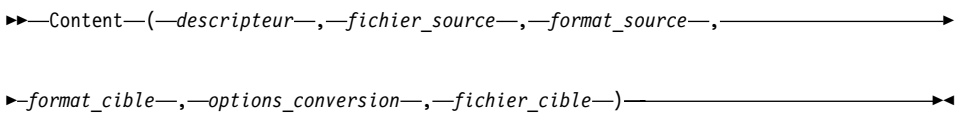
Syntaxe

Mise à jour du contenu à partir d'une mémoire tampon ou d'un fichier client avec conversion et modifications supplémentaires (image uniquement)



Syntaxe

Mise à jour du contenu à partir d'un fichier serveur avec conversion de format et modifications supplémentaires (image uniquement)



Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

décalage (INTEGER)
Décalage de début (en partant de 1) d'un objet image, audio, ou vidéo à extraire.

taille (INTEGER)
Nombre d'octets d'un objet image, audio ou vidéo à extraire.

fichier_source (LONG VARCHAR)
Nom du fichier dans lequel se trouve le contenu pour mise à jour de l'objet image, audio ou vidéo.

fichier_cible(LONG VARCHAR)
Pour l'extraction, il s'agit du nom du fichier à partir duquel est extrait l'objet image, audio ou vidéo. Pour la mise à jour, nom du fichier contenant l'objet image, audio ou vidéo à mettre à jour.

typestoc (INTEGER)
Valeur indiquant où sera stocké l'objet image, audio ou vidéo mis à jour. La constante MMDB_STORAGE_TYPE_INTERNAL (value=1) indique que l'objet mis à jour sera stocké dans la base de données en tant qu'objet BLOB. La constante MMDB_STORAGE_TYPE_EXTERNAL (value=0) indique que l'objet mis à jour sera stocké dans un fichier du serveur.

écrasement (INTEGER)
Valeur indiquant si le fichier cible doit être écrasé s'il existe déjà. Cette valeur peut être 0 ou 1. La valeur 0 signifie que le fichier cible n'est pas écrasé (en fait, l'extraction n'a pas lieu). La valeur 1 signifie que le fichier cible est écrasé s'il existe déjà.

format_cible (VARCHAR(8))
Format de l'image après extraction ou mise à jour. Le format de l'image source est converti en fonction des besoins. Dans le cas d'une extraction d'une image vers un fichier du serveur, si le paramètre fichier_cible est identique au fichier_source, le format cible doit également être identique au format source. Pour le format MPG1, vous pouvez indiquer MPG1, mpg1, MPEG1 ou mpeg1. Pour le format MPG2, vous pouvez indiquer MPG2, mpg2, MPEG2 ou mpeg2.

options_conversion (VARCHAR(100))
Indique les modifications (rotation ou compression, par exemple) à appliquer à l'image lors de son extraction ou de sa mise à jour. Pour connaître les options de conversion admises, reportez-vous au tableau 6 à la page 93.

contenu (BLOB(2G) AS LOCATOR)
Variable SQL dans laquelle est stocké le contenu de la mise à jour d'un objet image, audio ou vidéo. Cette variable peut être de type BLOB, BLOB_FILE ou BLOB_LOCATOR. DB2 transmet le type de données du contenu à BLOB_LOCATOR, et le releveur de coordonnées LOB à la fonction UDF Content.

format_source (VARCHAR(8))
Format de la source pour la mise à jour de l'objet image, audio ou vidéo. Il peut s'agir d'une valeur nulle ou d'une chaîne de caractères vide et, pour les images uniquement, d'une chaîne de caractères ASIS ; dans ces trois cas, l'extension tente de déterminer le format automatiquement. Pour le format MPG1, vous pouvez indiquer MPG1, mpg1, MPEG1 ou mpeg1. Pour le format MPG2, vous pouvez indiquer MPG2, mpg2, MPEG2 ou mpeg2.

attrs (LONG VARCHAR FOR BIT)
Attributs de l'objet image, audio ou vidéo

miniature (LONG VARCHAR FOR BIT DATA)
Version miniature de l'image de base de données ou de vidéo (uniquement pour ces deux types d'images)

Codes retour (type de données)

Contenu de l'objet image, audio ou vidéo s'il s'agit d'une extraction vers une mémoire tampon, (BLOB(2G) AS LOCATOR). S'il s'agit d'une extraction vers un fichier, VARCHAR(254).

Pour la mise à jour, le descripteur de l'objet image, audio ou vidéo à mettre à jour (DB2IMAGE, DB2AUDIO ou DB2VIDEO).

Exemples

Extraction dans un fichier du serveur de l'image stockée pour Anne Dupont dans la colonne Photo de la table Employés :

```
struct{
        short len;
        char data [250];
         }hvImg_fname;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT CONTENT (PICTURE,
```

```
       '/Employés/images/adupont.bmp',1)
INTO :hvImg_fname
       FROM EMPLOYEE
       WHERE NAME='Anne Dupont';
```

Extraction dans une mémoire tampon client de la séquence audio de 1 Mo stockée pour Robert Durand dans la colonne Son de la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
     SQL TYPE IS BLOB_LOCATOR loc_audio;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT CONTENT (SOUND, 1, 1000000)
      INTO :audio_loc
       FROM EMPLOYEE
     WHERE NAME='Robert Durand';
```

Mise à jour de l'image d'Anne Dupont dans la colonne Photo de la table Employés ; conversion de l'image du format BMP en format GIF et réduction de l'image à 50% de sa taille d'origine :

```
EXEC SQL UPDATE EMPLOYEE
  SET picture = CONTENT(PICTURE,
        '/Employés/nouvimg/adupont.bmp',
          'BMP',
          'GIF',
        '-s 0.5',
       '');
       WHERE NAME='Anne Dupont';
```

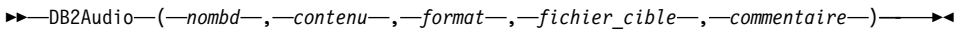
DB2Audio

Image	Audio	Vidéo
	X	

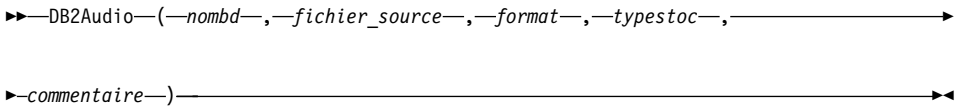
Stocke le contenu d'une séquence audio dans une table de base de données. La source audio peut se trouver dans une mémoire tampon client, un fichier client ou dans un fichier du serveur. L'objet audio peut être stocké dans la table de la base de données en tant qu'objet BLOB ou dans un fichier du serveur (référencé par la table de la base de données). Le format de la source audio peut être pris en charge ou non. Dans le premier cas, l'extension DB2Audio identifie ses attributs pour le stockage. Dans le second cas, vous devez indiquer ses attributs dans la fonction UDF.

Fichier d'inclusion
dmbaudio.h

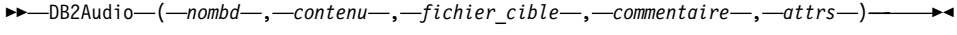
Syntaxe
Stockage du contenu à partir d'une mémoire tampon ou d'un fichier client



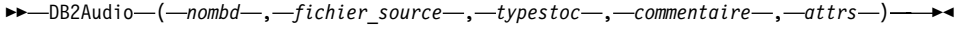
Syntaxe
Stockage du contenu à partir d'un fichier serveur



Syntaxe
Stockage du contenu avec des attributs définis par l'utilisateur à partir d'une mémoire tampon ou d'un fichier client



Syntaxe
Stockage du contenu avec des attributs définis par l'utilisateur à partir d'un fichier serveur



Paramètres (type de données)

nombd (VARCHAR(18))
Nom de la base de données à laquelle vous êtes connecté, indiqué par le registre spécial CURRENT SERVER.

contenu (BLOB(2G) AS LOCATOR)
Variable SQL représentant le contenu de l'objet audio. Cette variable peut être de type BLOB, BLOB_FILE ou BLOB_LOCATOR. DB2 considère le type de données du contenu comme le BLOB-LOCATOR et transfère le releveur de coordonnées LOB à la fonction UDF DB2Audio.

format (VARCHAR(8))
Format de l'objet audio source. Vous pouvez indiquer une valeur nulle ou une chaîne de caractères vide. Dans ce cas, l'extension audio tente de déterminer le format source automatiquement. L'objet audio est stocké dans le même format que l'objet source correspondant. Pour connaître les formats audio admis, reportez-vous au tableau 5 à la page 91.

fichier_cible(LONG VARCHAR)
Nom du fichier du serveur cible (dans le cas d'un stockage dans un fichier du serveur), valeur nulle ou chaîne de caractères vide (dans le cas d'un stockage dans une table de base de données en tant qu'objet BLOB). Le nom du fichier cible peut être qualifié complet. S'il n'est pas qualifié, le fichier est localisé à l'aide des variables d'environnement DB2AUDIOSTORE et DB2MMSTORE sur le serveur.

fichier_source (LONG VARCHAR)
Nom du fichier du serveur source. Ce nom peut être qualifié complet ou non qualifié, mais ne doit pas être une valeur nulle ou une chaîne de caractères vide. Si le nom du fichier n'est pas qualifié, ce dernier est localisé à l'aide des variables d'environnement DB2AUDIOPATH et DB2MMPATH sur le serveur.

typestoc (INTEGER)
Valeur indiquant où l'objet audio est stocké. La constante MMDB_STORAGE_TYPE_INTERNAL (value=1) indique que l'objet audio est stocké dans la base de données en tant qu'objet BLOB ; la constante MMDB_STORAGE_TYPE_EXTERNAL (value=0) indique que le contenu de l'objet audio est stocké dans un fichier du serveur (la base de données contient un pointeur sur ce fichier).

commentaire (LONG VARCHAR)

Commentaire à stocker avec l'objet audio.

attrs (LONG VARCHAR FOR BIT DATA)
Attributs de l'objet audio.

Codes retour (type de données)
Descripteur de l'objet audio (DB2AUDIO)

Exemples
Insertion d'un enregistrement comprenant une séquence audio pour Anne Dupont dans la table Employés. La source audio se trouve dans une mémoire tampon client. Stockage de la séquence audio dans la table en tant qu'objet BLOB :

```
EXEC SQL BEGIN DECLARE SECTION;
     SQL TYPE IS BLOB (5M) aud_seg;
  EXEC SQL END DECLARE SECTION;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
               DB2AUDIO(
          CURRENT SERVER,
         :aud_seg,
          'WAVE',
                 CAST(NULL as LONG VARCHAR),

         'Voix d''Anne'));
```

Insertion d'un enregistrement comprenant une séquence audio pour Robert Durand dans la table Employés. La source audio se trouve dans un fichier du serveur. L'enregistrement de la table Employés pointe sur ce fichier.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType = MMDB_STORAGE_TYPE_EXTERNAL;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
      '384779',
      'Robert Durand',
               DB2AUDIO(
          CURRENT SERVER,
         '/Employés/sons/rdurand.wav',
         'WAV',
          :hvStorageType,
         'Voix de Robert'));
```

Insertion d'un enregistrement comprenant une séquence audio pour Anne Dupont dans la table Employés. Stockage de la séquence audio en tant qu'objet BLOB. La séquence audio source, qui se trouve dans un fichier du serveur, a un format défini par l'utilisateur, une fréquence d'échantillonnage de 44,1 KHz, et comporte deux canaux enregistrés.

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
         struct  {
        short len;
```

```
       char data[600];
}hvAudattr;
  EXEC SQL END DECLARE SECTION;

MMDBAudioAttrs          *paudiattr;

hvStorageType = MMDB_STORAGE_TYPE_INTERNAL;

paudioattr=(MMDBAudioAttrs *) hvAudattr.data;
strcpy(paudioAttr→cFormat,"cFormatA");
paudioAttr→ulSamplingRate=44100;
paudioAttr→usNumChannels=2;
hvAudattrs.len=sizeof(MMDBAudioAttrs);

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
               DB2AUDIO(
          CURRENT SERVER,
          '/Employés/sons/adupont.aud',
          :hvStorageType,
               'Voix d"Anne',
               :hvAudattr)
       );
```

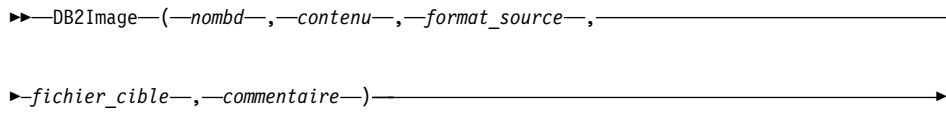
DB2Image

Image	Audio	Vidéo
X		

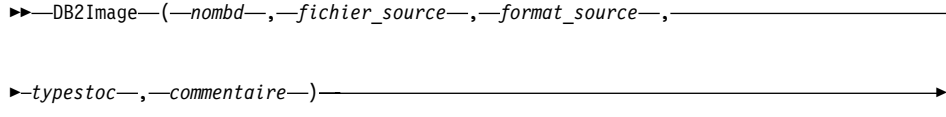
Stocke le contenu d'une image dans une table de base de données. L'objet image source peut se trouver dans une mémoire tampon client, un fichier client ou dans un fichier du serveur. L'objet image peut être stocké dans la table de la base de données en tant qu'objet BLOB ou dans un fichier du serveur (référencé par la table de la base de données). Le format de l'objet image source peut être pris en charge ou pas. Dans le premier cas, l'extension DB2Image identifie ses attributs pour le stockage. Dans le second cas, les attributs doivent être indiqués dans la fonction UDF.

Fichier d'inclusion
dmbimage.h

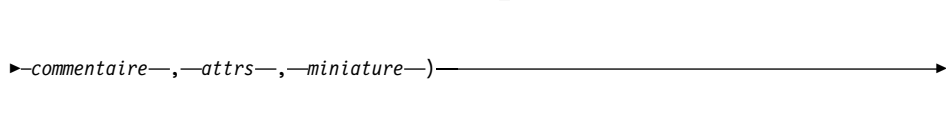
Syntaxe
Stockage du contenu à partir d'une mémoire tampon ou d'un fichier client



Syntaxe
Stockage du contenu à partir d'un fichier serveur

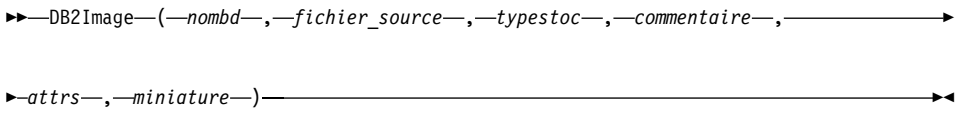


Syntaxe
Stockage du contenu avec des attributs définis par l'utilisateur à partir d'une mémoire tampon ou d'un fichier client



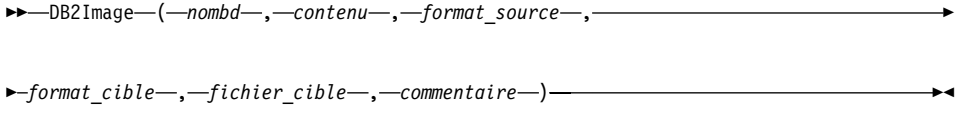
Syntaxe

Stockage du contenu avec des attributs définis par l'utilisateur à partir d'un fichier serveur



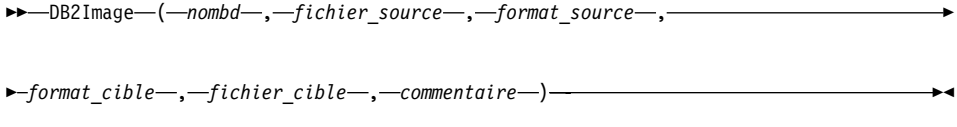
Syntaxe

Stockage du contenu à partir d'une mémoire tampon ou d'un fichier client avec conversion de fichier



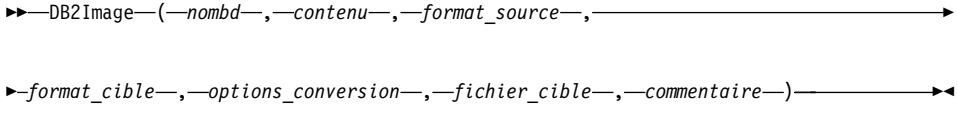
Syntaxe

Stockage du contenu à partir d'un fichier serveur avec conversion de format



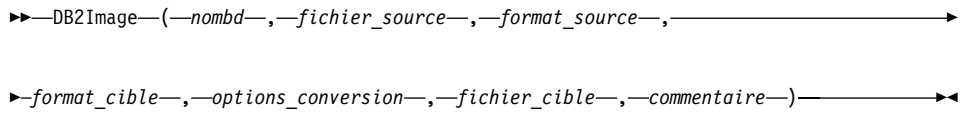
Syntaxe

Stockage du contenu à partir d'une mémoire tampon ou d'un fichier client avec conversion et modifications supplémentaires



Syntaxe

Stockage du contenu à partir d'un fichier serveur avec conversion de format et modifications supplémentaires



Paramètres (type de données)

nombd (VARCHAR(18))
Nom de la base de données à laquelle vous êtes connecté, indiqué par le registre spécial CURRENT SERVER.

contenu (BLOB(2G) AS LOCATOR)
Variable SQL dans laquelle se trouve le contenu de l'objet image. Cette variable peut être de type BLOB, BLOB_FILE ou BLOB_LOCATOR. DB2 transmet le type de données du contenu à BLOB_LOCATOR, et le releveur de coordonnées LOB à la fonction UDF DB2Audio.

format_source (VARCHAR(8))
Format de l'image source. Il peut s'agir d'une valeur nulle, d'une chaîne de caractères vide ou de la chaîne de caractères ASIS ; dans ces trois cas, l'extension Image tente de déterminer le format source automatiquement. L'image est stockée dans le même format que l'image source correspondante. Pour connaître les formats d'image admis, reportez-vous au tableau 5 à la page 91.

format_cible (VARCHAR(8))
Format de l'image après stockage. Le format de l'image source est converti en fonction des besoins.

fichier_cible(LONG VARCHAR)
Nom du fichier du serveur cible (dans le cas d'un stockage dans un fichier du serveur), valeur nulle ou chaîne de caractères vide (dans le cas d'un stockage dans une table de base de données en tant qu'objet BLOB). Le nom du fichier cible peut être qualifié complet. S'il n'est pas qualifié, le fichier est localisé à l'aide des variables d'environnement DB2IMAGESTORE et DB2MMSTORE sur le serveur. Si l'image est stockée avec conversion de format, le chemin d'accès au fichier cible doit être indiqué dans les variables d'environnement DB2IMAGEPATH et DB2MMPATH.

fichier_source (LONG VARCHAR)
Nom du fichier du serveur source. Ce nom peut être qualifié complet ou non qualifié, mais ne doit pas être une valeur nulle ou une chaîne

de caractères vide. Si le nom du fichier n'est pas qualifié, ce dernier est localisé à l'aide des variables d'environnement DB2IMAGEPATH et DB2MMPATH sur le serveur.

typestoc (INTEGER)
Valeur indiquant où l'image est stockée. La constante MMDB_STORAGE_TYPE_INTERNAL (value=1) indique que l'image est stockée dans la base de données en tant qu'objet BLOB ; la constante MMDB_STORAGE_TYPE_EXTERNAL (value=0) indique que le contenu de l'image est stocké dans un fichier du serveur (la base de données contient un pointeur sur ce fichier).

commentaire (LONG VARCHAR)
Commentaire à stocker avec l'image.

attrs (LONG VARCHAR FOR BIT DATA)
Attributs de l'image.

miniature (LONG VARCHAR FOR BIT DATA)
Version miniature de l'image.

options_conversion (VARCHAR(100))
Indique les modifications (rotation ou compression, par exemple) à appliquer à l'image lors de son stockage. Pour connaître les options de conversion admises, reportez-vous au tableau 6 à la page 93.

Codes retour (type de données)
Descripteur de l'image (DB2IMAGE)

Exemples
Insertion d'un enregistrement incluant une image d'Anne Dupont dans la table Employés. L'image source se trouve dans une mémoire tampon sur le poste client. Stockage de l'image dans la table en tant qu'objet BLOB :

```
EXEC SQL BEGIN DECLARE SECTION
     SQL TYPE IS BLOB (2M) hvImg
  EXEC SQL END DECLARE SECTION;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2IMAGE(
          CURRENT SERVER,
         :hvImg,
          'ASIS',
                 CAST(NULL as LONG VARCHAR),

          'Photo d''Anne'));
```

Insertion d'un enregistrement incluant une photo de Robert Durand, dans la table Employés. L'objet image source se trouve dans un fichier du serveur. L'enregistrement de la table Employés pointe sur ce fichier. Conversion de

l'image du format BMP au format GIF lors de son stockage. Réduction de l'image à une largeur de 110 pixels et une hauteur de 150 pixels et compression de l'image en utilisant le type de compression LZW :

```
EXEC SQL INSERT INTO EMPLOYEE VALUES(
      '384779',
      'Robert Durand',
      DB2IMAGE(
          CURRENT SERVER,
         '/employés/photo/rdurand.bmp',
          'BMP',
          'GIF',
         '-x 110 -y 150 -c 14',
         '',
         'photo de Robert'));
```

Insertion d'un enregistrement incluant une photo de Robert Durand, dans la table Employés. L'image source qui se trouve dans un fichier du serveur, a un format défini par l'utilisateur, une hauteur de 640 pixels et une largeur de 480 pixels. Stockage de l'image en tant qu'objet BLOB :

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
         struct  {
        short len;
        char data [400];
}hvImgattrs;
  EXEC SQL END DECLARE SECTION;

DB2IMAGEATTRS    *pimgattr;

hvStorageType = MMDB_STORAGE_TYPE_INTERNAL;

pimgattr = (DB2IMAGEATTRS *) hvImgattrs.data;
strcpy(pimgattrt→cFormat,"FormatI");
pimgattr→width=640;
pimgattr→height=480;
hvImgattrs.len=sizeof(DB2IMAGEATTRS);

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2IMAGE(
          CURRENT SERVER,
          '/Employés/images/adupont.bmp',
          :hvStorageType,
          'Photo d''Anne',
          :hvImgattrs,
          CAST(NULL as LONG VARCHAR))
       );
```

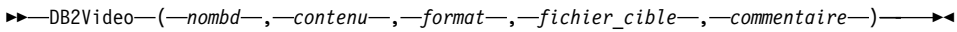
DB2Video

Image	Audio	Vidéo
		X

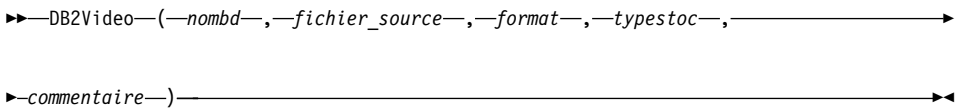
Stocke le contenu d'une séquence vidéo dans une table de base de données. L'objet vidéo source peut se trouver dans une mémoire tampon client, un fichier client ou un fichier du serveur. L'objet vidéo peut être stocké dans la table de la base de données en tant qu'objet BLOB ou dans un fichier du serveur (référencé par la table de la base de données). Le format de l'objet vidéo source peut être pris en charge ou pas. Dans le premier cas, l'extension DB2Video identifie ses attributs pour le stockage. Dans le second cas, les attributs doivent être indiqués dans la fonction UDF.

Fichier d'inclusion
dmbvideo.h

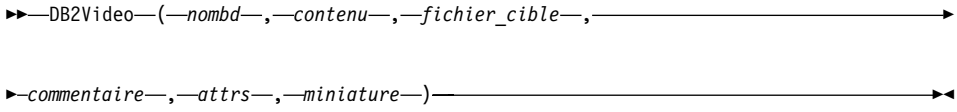
Syntaxe
Stockage du contenu à partir d'une mémoire tampon ou d'un fichier client



Syntaxe
Stockage du contenu à partir d'un fichier serveur

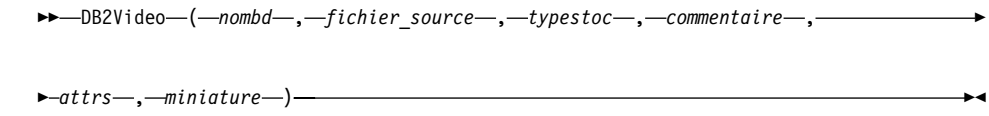


Syntaxe
Stockage du contenu avec des attributs définis par l'utilisateur à partir d'une mémoire tampon ou d'un fichier client



Syntaxe

Stockage du contenu avec des attributs définis par l'utilisateur à partir d'un fichier serveur



Paramètres (type de données)

nombd (VARCHAR(18))
Nom de la base de données à laquelle vous êtes connecté, indiqué par le registre spécial CURRENT SERVER.

contenu (BLOB(2G) AS LOCATOR)
Variable SQL représentant le contenu de la vidéo. Cette variable peut être de type BLOB, BLOB_FILE ou BLOB_LOCATOR. DB2 transmet le type de valeur du contenu à BLOB_LOCATOR, et le releveur de coordonnées LOB à la fonction UDF DB2Video. Si ce contenu se trouve dans une mémoire tampon client, cette dernière doit en comporter au moins les premiers 640 Ko afin de garantir la lecture complète de l'en-tête de la vidéo.

format (VARCHAR(8))
Format de la vidéo source. Si vous indiquez une valeur nulle ou une chaîne de caractères vide, l'extension Vidéo tente de déterminer le format source automatiquement. La vidéo est stockée dans le même format que sa source. Pour connaître les formats vidéo admis, reportez-vous au tableau 5 à la page 91. Pour le format MPG1, vous pouvez indiquer MPG1, mpg1, MPEG1 ou mpeg1. Pour le format MPG2, vous pouvez indiquer MPG2, mpg2, MPEG2 ou mpeg2.

fichier_cible(LONG VARCHAR)
Nom du fichier du serveur cible (dans le cas d'un stockage dans un fichier du serveur), valeur nulle ou chaîne de caractères vide (dans le cas d'un stockage dans une table de base de données en tant qu'objet BLOB). Le nom du fichier du serveur doit être qualifié complet. S'il n'est pas qualifié, le fichier est localisé à l'aide des variables d'environnement DB2VIDEOSTORE et DB2MMSTORE sur le serveur.

fichier_source (LONG VARCHAR)
Nom du fichier du serveur source. Ce nom peut être qualifié complet ou non qualifié, mais ne doit pas être une valeur nulle ou une chaîne de caractères vide. Si le nom du fichier n'est pas qualifié, ce dernier est localisé à l'aide des variables d'environnement DB2VIDEOPATH et DB2MMPATH sur le serveur.

typestoc (INTEGER)
Valeur indiquant où la vidéo est stockée. La constante MMDB_STORAGE_TYPE_INTERNAL (value=1) indique que la vidéo est stockée dans la base de données en tant qu'objet BLOB ; la constante MMDB_STORAGE_TYPE_EXTERNAL (value=0) indique que le contenu de la vidéo est stocké dans un fichier du serveur (la base de données contient un pointeur sur ce fichier).

commentaire (LONG VARCHAR)
Commentaire à stocker avec la vidéo.

attrs (LONG VARCHAR FOR BIT DATA)
Attributs de la vidéo.

miniature (LONG VARCHAR FOR BIT DATA)
Image miniature représentant la vidéo.

Codes retour (type de données)
Descripteur de la vidéo (DB2VIDEO)

Exemples
Insertion d'un enregistrement incluant une séquence vidéo pour Anne Dupont, dans la table Employés. La source de la vidéo se trouve dans une mémoire tampon client. Stockage de la séquence vidéo dans la table en tant qu'objet BLOB :

```
EXEC SQL BEGIN DECLARE SECTION
     SQL TYPE IS BLOB (8M) vid;
  EXEC SQL END DECLARE SECTION;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2VIDEO(
          CURRENT SERVER,
         :vid,
          'MPEG1',
CAST(NULL as LONG VARCHAR),
         'Séquence vidéo d''Anne'));
```

Insertion d'un enregistrement incluant une séquence vidéo pour Robert Durand, dans la table Employés. La source de la vidéo se trouve dans un fichier du serveur. L'enregistrement de la table Employés pointera sur ce fichier :

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
  EXEC SQL END DECLARE SECTION;

hvStorageType = MMDB_STORAGE_TYPE_EXTERNAL;

EXEC SQL INSERT INTO EMPLOYEE VALUES(
      '384779',
```

```
      'Robert Durand',
      DB2VIDEO(
          CURRENT SERVER,
         '/Employés/vidéos/rdurand.mpg',
          'MPEG1',
          :hvStorageType,
                 'Séquence vidéo de Robert'));
```

Insertion d'un enregistrement incluant une séquence vidéo, dans une table de base de données. La séquence vidéo source, qui se trouve dans un fichier du serveur, possède un format défini par l'utilisateur. Conservation du contenu de la vidéo dans le fichier du serveur (l'enregistrement de la table de base de données pointe sur ce fichier). Stockage d'une image miniature représentant la vidéo :

```
EXEC SQL BEGIN DECLARE SECTION;
long hvStorageType;
         struct  {
        short len;
        char data [400];
       }hvVidattrs;
         struct  {
        short len;
       char data[10000];
}hvThumbnail;
  EXEC SQL END DECLARE SECTION;

MMDBVideoAttrs         *pvideoAttr;

hvStorageType = MMDB_STORAGE_TYPE_EXTERNAL;

pvideoAttr=(MMDBVideoAttrs *)hvVidattrs.data;
strcpy(pvideoAttr→cFormat,"Formatv");
pvideoAttr.len=sizeof(MMDBVideoAttrs);
:
:

/* Création d'une miniature et affectation */
/* de données dans la structure de la vidéo */
:
:

EXEC SQL INSERT INTO EMPLOYEE VALUES(
        '128557',
      'Anne Dupont',
      DB2VIDEO(
          CURRENT SERVER,
          '/Employés/vidéo/adupont.vid',
          :hvStorageType,
          'Séquence vidéo d''Anne',
          :hvVidattrs,
          :hvThumbnail)
       );
```

Duration

Image	Audio	Vidéo
	X	X

Renvoie la durée (temps de lecture en secondes) d'une séquence audio de type WAVE ou AIFF ou d'une vidéo.

Fichier d'inclusion

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio ou vidéo.

Codes retour (type de données)

Durée, en secondes, d'une vidéo ou d'une séquence audio de type WAVE, AIFF ou définie par l'utilisateur (INTEGER). Renvoie une valeur nulle pour les séquences audio de format différent.

Exemples

Affichage de la durée de toutes les séquences vidéos stockées dans la colonne vidéo de la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
long hvDur_vid;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT DURATION(VIDEO)
     INTO :hvDur_vid
     FROM EMPLOYEE;
```

Filename

Image	Audio	Vidéo
X	X	X

Renvoie le nom du fichier du serveur dans lequel se trouve le contenu d'un objet image, audio ou vidéo, s'il est stocké dans un fichier (une table de base de données contient un pointeur sur ce fichier). Si l'objet image, audio ou vidéo est stocké dans une table de base de données en tant qu'objet BLOB, le système renvoie une valeur nulle.

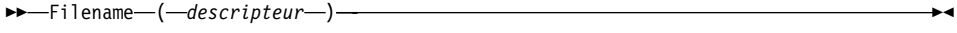
Fichier d'inclusion

image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe



Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

Codes retour (type de données)

Nom du fichier du serveur si le contenu de l'objet se trouve dans un fichier du serveur (VARCHAR(250)) ; valeur nulle si l'objet est stocké en tant qu'objet BLOB.

Exemples

Affichage du nom de fichier de la vidéo pour l'entrée Robert Durand dans la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvVid_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME (VIDEO)
     INTO :hvVid_fname
       FROM EMPLOYEE
     WHERE NAME='Robert Durand';
```

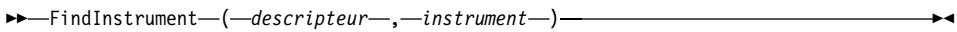
FindInstrument

Image	Audio	Vidéo
	X	

Renvoie le numéro de piste de la première occurrence d'un instrument spécifique d'une séquence audio de type MIDI.

Fichier d'inclusion
dmbaudio.h

Syntaxe



Paramètres (type de données)

descripteur (DB2AUDIO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio.

instrument (VARCHAR(255))
Nom de l'instrument à rechercher. L'extension Audio recherche un instrument dont le nom est identique au nom fourni.

Codes retour (type de données)
Numéro de la piste contenant la première occurrence d'un nom d'instrument spécifié (SMALLINT) ; la valeur -1 est renvoyée si l'extension ne trouve pas d'instrument portant le nom spécifié. La valeur NULL est renvoyée pour les séquences audio de format différent.

Exemples
Recherche de la première occurrence de l'instrument PIANO dans l'enregistrement audio de type MIDI de Robert Durand stocké dans la colonne Son de la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
      short hvInstr;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FINDINSTRUMENT(SOUND, 'PIANO')
      INTO :hvInstr
       FROM EMPLOYEE
      WHERE NAME = 'Robert Durand';
```

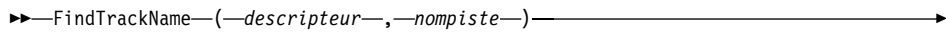
FindTrackName

Image	Audio	Vidéo
	X	

Renvoie le numéro d'une piste nommée spécifique dans une séquence audio de type MIDI.

Fichier d'inclusion
dmbaudio.h

Syntaxe



Paramètres (type de données)

descripteur (DB2AUDIO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio.

nompiste (VARCHAR(255))
Nom de la piste à rechercher. L'extension Audio recherche une piste dont le nom est identique au nom fourni.

Codes retour (type de données)
Numéro de la piste nommée de l'instrument indiqué (SMALLINT). La valeur -1 est renvoyée si l'extension ne trouve pas de piste portant le nom indiqué. Une valeur nulle est renvoyée pour les séquences audio de format différent.

Exemples
Recherche d'une piste nommée WELCOME dans l'enregistrement audio de type MIDI de Robert Durand. L'enregistrement audio est stocké dans la colonne Son de la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
      short hvTrack;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FINDTRACKNAME(SOUND,
          'WELCOME')
      INTO :hvTrack
       FROM EMPLOYEE
      WHERE NAME = 'Robert Durand';
```

Format

Image	Audio	Vidéo
X	X	X

Renvoie le format d'un objet image, audio ou vidéo.

Fichier d'inclusion

image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

Codes retour (type de données)

Format de l'objet image, audio ou vidéo (VARCHAR(8)). Pour plus d'informations sur les formats des objets image, audio ou vidéo pris en charge, reportez-vous au tableau 5 à la page 91.

Exemples

Extraction des noms de tous les employés dont les images stockées dans la colonne Photo de la table Employés ont le format GIF :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvName[30];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT NAME
     INTO :hvName
       FROM EMPLOYEE
     WHERE FORMAT(PICTURE)='GIF';
```

FrameRate

Image	Audio	Vidéo
		X

Renvoie la vitesse de défilement d'une vidéo en images/seconde.

Fichier d'inclusion
dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence vidéo.

Codes retour (type de données)
Vitesse de défilement d'une vidéo (SMALLINT). Renvoie une valeur nulle si le débit est variable.

Exemples
Extraction de la vitesse de défilement d'une vidéo stockée dans la colonne vidéo de la table Employés pour Anne Dupont :

```
EXEC SQL BEGIN DECLARE SECTION;
short hvFm_rate;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FRAMERATE (VIDEO)
       FROM EMPLOYEE
      INTO :hvFm_rate
       WHERE NAME='Anne Dupont';
```

GetInstruments

Image	Audio	Vidéo
	X	

Renvoie le nom de tous les instruments d'une séquence audio de type MIDI.

Fichier d'inclusion
dmbaudio.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio.

Codes retour (type de données)
Nom de tous les instruments d'une séquence audio de type MIDI (VARCHAR(1536)). Les valeurs sont renvoyées par ordre de numéro de piste (par exemple, PIANO, TRUMPET, BASS). Le résultat est divisé en *n* zones, *n* correspondant au nombre de pistes de la séquence audio MIDI. Si aucun instrument n'est associé à une piste, la zone correspondante est vide. Une valeur nulle est renvoyée pour les formats audio différents de MIDI.

Exemples
Recherche de tous les instruments (numéros de piste et noms des instruments) dans l'enregistrement audio de type MIDI de Robert Durand.
L'enregistrement audio est stocké dans la colonne Son de la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
 char hvAud_Instr[1536];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT GETINSTRUMENTS(SOUND)
      INTO :hvAud_Instr
       FROM EMPLOYEE
      WHERE NAME = 'Robert Durand';
```

GetTrackNames

Image	Audio	Vidéo
	X	

Renvoie le nom de toutes les pistes d'une séquence audio de type MIDI.

Fichier d'inclusion
dmbaudio.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio.

Codes retour (type de données)
Nom de toutes les pistes d'une séquence audio de type MIDI (VARCHAR(1536)). Les valeurs sont renvoyées par ordre de numéro de piste (par exemple, PIANO TUNE, TRUMPET FANFARE). Le résultat est divisé en *n* zones, *n* correspondant au nombre de pistes de la séquence audio MIDI. Si une piste n'a pas de nom, la zone correspondante est vide. Une valeur nulle est renvoyée pour les formats audio différents de MIDI.

Exemples
Extraction de tous les numéros et noms de pistes de l'enregistrement audio de type MIDI de Robert Durand stocké dans la colonne Son de la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
 char hvTracks[1536];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT GETTRACKNAMES(SOUND)
      INTO :hvTracks
       FROM EMPLOYEE
      WHERE NAME = 'Robert Durand';
```

Height

Image	Audio	Vidéo
X		X

Renvoie la hauteur, en pixels, d'une image de base de données ou de vidéo.

Fichier d'inclusion

image dmbimage.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2IMAGE ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de l'image ou de la vidéo.

Codes retour (type de données)

Hauteur en pixels (INTEGER)

Exemples

Extraction du nom de fichier de toutes les images de la colonne Photo de la table Employés d'une hauteur inférieure à 500 pixels :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(PICTURE)
INTO :hvImg_fname
       FROM EMPLOYEE
     WHERE HEIGHT(PICTURE)<500;
```

Importer

Image	Audio	Vidéo
X	X	X

Renvoie l'ID de l'utilisateur ayant stocké un objet image audio ou vidéo dans une table de base de données.

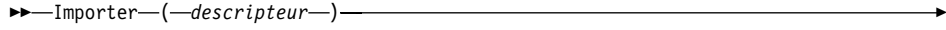
Fichier d'inclusion

image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe



Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

Codes retour (type de données)

ID de l'utilisateur (CHAR(8))

Exemples

Extraction du nom de tous les fichiers pour les séquences audio stockées dans la colonne Son de la table Employés par l'ID utilisateur rdurand :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvAud_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME (SOUND)
      INTO :hvAud_fname
       FROM EMPLOYEE
     WHERE IMPORTER(SOUND)='rdurand';
```

ImportTime

Image	Audio	Vidéo
X	X	X

Renvoie un horodateur indiquant la date de stockage d'un objet image, audio ou vidéo dans une table de base de données.

Fichier d'inclusion

image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

Codes retour (type de données)

Horodatage indiquant la date de stockage d'un objet image, audio ou vidéo (TIMESTAMP)

Exemples

Extraction des noms de tous les fichiers pour les images stockées dans la colonne Photo de la table Employés il y a plus d'un an :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(PICTURE)
INTO :hvImg_fname
       FROM EMPLOYEE
     WHERE(CURRENT TIMESTAMP -
        IMPORTTIME(PICTURE))>365;
```

MaxBytesPerSec

Image	Audio	Vidéo
		X

Renvoie le débit maximal d'une séquence vidéo en nombre d'octets par seconde.

Fichier d'inclusion
dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence vidéo.

Codes retour (type de données)
Débit d'une vidéo (INTEGER). Renvoie une valeur nulle si le débit est variable.

Exemples
Extraction de la vitesse de défilement d'une vidéo stockée dans la colonne vidéo de la table Employés pour Anne Dupont :

```
EXEC SQL BEGIN DECLARE SECTION;
long hvMax_BytesPS;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT MAXBYTESPERSEC(VIDEO)
     INTO :hvMax_BytesPS
       FROM EMPLOYEE
       WHERE NAME='Anne Dupont';
```

NumAudioTracks

Image	Audio	Vidéo
	X	X

Renvoie le nombre de pistes audio d'une vidéo ou d'une séquence audio de type MIDI.

Fichier d'inclusion

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio ou vidéo.

Codes retour (type de données)

Nombre de pistes audio d'une vidéo ou d'une séquence audio de type MIDI (SMALLINT). Renvoie une valeur nulle pour les séquences audio de format différent.

Exemples

Extraction des noms des fichiers vidéo de la colonne vidéo de la table Employés ne contenant pas de pistes audio :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvVid_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME (VIDEO)
     INTO :hvVid_fname
       FROM EMPLOYEE
     WHERE NUMAUDIOTRACKS(VIDEO) = 0;
```

NumChannels

Image	Audio	Vidéo
	X	X

Renvoie le nombre de canaux audio enregistrés dans une séquence audio de type WAVE ou AIFF ou dans une vidéo.

Fichier d'inclusion

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio ou vidéo.

Codes retour (type de données)

Nombre de canaux audio enregistrés dans la vidéo ou dans la séquence audio de type WAVE ou AIFF (SMALLINT). Renvoie une valeur nulle pour les séquences audio de format différent.

Exemples

Extraction des noms de tous les fichiers audio de la colonne Son de la table Employés enregistrés en stéréo (sur deux canaux) :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvAud_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME (SOUND)
      INTO :hvAud_fname
       FROM EMPLOYEE
     WHERE NUMCHANNELS(SOUND) = 2;
```

NumColors

Image	Audio	Vidéo
X		

Renvoie le nombre de couleurs composant une image.

Fichier d'inclusion
dmbimage.h

Syntaxe

Paramètres (type de données)

descripteur (DB2IMAGE)
Nom de colonne ou variable SQL contenant le descripteur de l'image.

Codes retour (type de données)
Nombre de couleurs d'une image (INTEGER)

Exemples
Extraction des noms de fichiers d'images de la colonne Photo de la table Employés pour les images comportant moins de 16 couleurs :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(PICTURE)
INTO :hvImg_fname
       FROM EMPLOYEE
     WHERE NUMCOLORS(PICTURE) < 16;
```

NumFrames

Image	Audio	Vidéo
		X

Renvoie le nombre d'images composant une séquence vidéo.

Fichier d'inclusion
dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence vidéo.

Codes retour (type de données)
Nombre d'images d'une vidéo (INTEGER). Renvoie une valeur nulle si le débit est variable.

Exemples
Extraction du nombre d'images de la vidéo stockée dans la colonne vidéo de la table Employés pour Robert Durand :

```
EXEC SQL BEGIN DECLARE SECTION;
long hvNum_Frames;
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT NUMFRAMES (VIDEO)
     INTO :hvNum_Frames
       FROM EMPLOYEE
     WHERE NAME='Robert Durand';
```

NumVideoTracks

Image	Audio	Vidéo
		X

Renvoie le nombre de pistes vidéo d'une séquence vidéo.

Fichier d'inclusion
dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence vidéo.

Codes retour (type de données)
Nombre de pistes vidéo (SMALLINT).

Exemples
Extraction du nom de fichier de toutes les vidéos de la colonne vidéo de la table Employés comportant plusieurs pistes vidéo :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvVid_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME (VIDEO)
     INTO :hvVid_fname
       FROM EMPLOYEE
     WHERE NUMVIDEOTRACKS(VIDEO) > 1;
```

QbScoreFromName

Image	Audio	Vidéo
X		

Renvoie, sous forme d'un chiffre, le score d'une image, c'est-à-dire le degré de correspondance de ses caractéristiques par rapport à celles de l'objet de la requête. Ce chiffre est calculé par le biais du catalogue QBIC associé à la colonne à laquelle appartient le descripteur de l'image. Moins ce chiffre est élevé, plus les caractéristiques de l'image correspondent à celles de l'objet de la requête. (QbScoreFromName remplace QbScore, mais QbScore est toujours admis.)

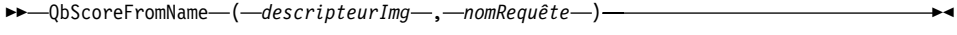
Remarques :

1. **Produit EEE uniquement :** QbScoreFromName n'est pas pris en charge dans un environnement de bases de données partitionnées. Utilisez la fonction UDF QbScoreFromStr après avoir obtenu la chaîne de requête à l'aide de l'API QbQueryGetString.
2. La fonction QbScoreFromName ne sera plus utilisée dans les prochaines versions des environnements de bases de données non partitionnées. Pour sauvegarder une requête en vue de l'utiliser ultérieurement dans votre application, il est donc conseillé d'extraire la chaîne de requête à l'aide de l'API QbQueryGetString.

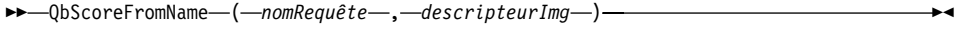
Fichier d'inclusion

Aucun.

Syntaxe



Syntaxe
Version déconseillée



Paramètres (type de données)

descripteurImg (DB2Image)
Descripteur de l'image.

nomRequête (varchar(18))
Nom de l'objet de la requête.

Codes retour (type de données)

Score de l'image (DOUBLE). La valeur du score peut aller de 0.0 à une valeur proche de l'infini. Plus cette valeur est faible, plus les valeurs des caractéristiques de l'image cible sont proches de celles spécifiées dans la requête. La valeur 0.0 indique que l'image correspond exactement. Une valeur nulle signifie que l'image n'a pas été cataloguée ; dans la version déconseillée, la valeur du score est -1 lorsque l'image n'a pas été cataloguée.

Exemples

Recherche des images cataloguées contenues dans une colonne de table et dont la couleur moyenne est très proche du rouge :

```
EXEC SQL BEGIN DECLARE SECTION;
char Img_fnd[100];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT NAME
  INTO :Img_fnd
  FROM FABRIC
  WHERE (QBSCOREFROMNAME(SWATCH_IMG,
         'fshavgcol'))<0.1;
```

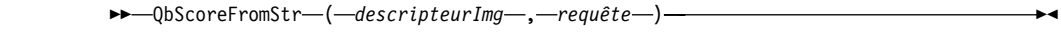
QbScoreFromStr

Image	Audio	Vidéo
X		

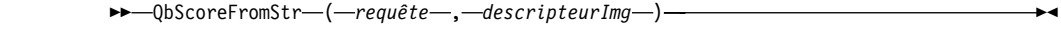
Renvoie, sous forme d'un chiffre, le score d'une image, autrement dit le degré de correspondance de ses caractéristiques par rapport à celles de la chaîne de requête. Ce chiffre est calculé par le biais du catalogue QBIC associé à la colonne à laquelle appartient le descripteur de l'image. Moins ce chiffre est élevé, plus les caractéristiques de l'image correspondent à celles de la chaîne de requête.

Fichier d'inclusion
Aucun.

Syntaxe



Syntaxe
Version déconseillée



Paramètres (type de données)

descripteurImg (DB2Image)
Descripteur de l'image.

requête (VARCHAR(1024))
Chaîne de requête.

Codes retour (type de données)
Score de l'image (DOUBLE). La valeur du score peut aller de 0.0 à une valeur proche de l'infini. Moins cette valeur est élevée, plus les valeurs des caractéristiques de l'image cible sont proches de celles spécifiées dans la requête. La valeur 0.0 indique que l'image correspond exactement. Une valeur nulle signifie que l'image n'a pas été cataloguée ; dans la version déconseillée, la valeur du score est -1 lorsque l'image n'a pas été cataloguée.

Exemples
Recherche des images cataloguées contenues dans une colonne de table et dont la couleur moyenne est très proche du rouge :

```
SELECT name
FROM fabric
WHERE (QbScoreFromStr(Swatch_Img,
       'QbColorFeatureClass color=<255, 0, 0>'))<0.1
```

QbScoreTBFromName

Image	Audio	Vidéo
X		

Renvoie une table de scores pour une colonne Image. Chaque score est un chiffre qui exprime le degré de correspondance des caractéristiques de l'image par rapport à celles de l'objet de la requête. Ce chiffre est calculé par le biais du catalogue QBIC associé à la table et à la colonne spécifiées auxquelles appartient le descripteur de l'image. Moins ce chiffre est élevé, plus les caractéristiques de l'image correspondent à celles de l'objet de la requête.

Remarques :

1. **Produit EEE uniquement :** QbScoreTBFromName n'est pas pris en charge dans un environnement de bases de données partitionnées. Utilisez la fonction UDF QbScoreFromStr après avoir obtenu la chaîne de requête à l'aide de l'API QbQueryGetString.
2. La fonction QbScoreTBFromName ne sera plus utilisée dans les prochaines versions des environnements de bases de données non partitionnées. Pour sauvegarder une requête en vue de l'utiliser ultérieurement dans votre application, il est donc conseillé d'extraire la chaîne de requête à l'aide de l'API QbQueryGetString.

Fichier d'inclusion

Aucun.

Syntaxe

Scores renvoyés pour toutes les images cataloguées d'une colonne

```
►►—QbScoreTBFromName—(—nomRequête—,—table—,—colonne—)————————————►◄
```

Syntaxe

Scores renvoyés pour un nombre spécifique d'images cataloguées d'une colonne

```
►►—QbScoreTBFromName—(—nomRequête—,—table—,—colonne—,—retoursMax—)————————►◄
```

Paramètres (type de données)

nomRequête (VARCHAR(18))
Nom de l'objet de la requête.

table (CHAR (18))
Nom qualifié de la table contenant la colonne Image. Vous pouvez

indiquer un nom de table non qualifié si le schéma de table correspond à l'ID utilisateur servant à lancer les services de DB2 Extensions.

colonne (CHAR(18))
Nom de la colonne Image.

retoursMax (INTEGER)
Nombre maximal de descripteurs qui seront renvoyés par la table de résultats. Si aucune valeur n'est indiquée, le nombre maximal de descripteurs renvoyés st 100.

Codes retour (type de données)

Table des descripteurs et des scores pour les images de la colonne. La table de résultats comporte deux colonnes : IMAGE_ID (DB2Image), qui contient les descripteurs d'image, et SCORE (DOUBLE), qui contient les scores. Les valeurs de cette table sont en ordre croissant de score. La valeur du score peut aller de 0.0 à une valeur proche de l'infini. Plus cette valeur est faible, plus les valeurs des caractéristiques de l'image cible sont proches de celles spécifiées dans la requête. La valeur 0.0 indique que l'image correspond exactement. La valeur -1 indique que l'image n'a pas été cataloguée.

Exemples

Compare la texture des images d'une colonne de table à celle spécifiée dans un objet de requête ; renvoie les descripteurs d'image ainsi que leurs scores :

```
SELECT name, description
INTO :hvName, :hvDesc
FROM fabric
WHERE CAST (swatch_img as varchar(250)) IN
 (SELECT CAST (image_id as varchar(250)) FROM TABLE
 (QbScoreTBFromName
     'fstxtr',
     'clothes.fabric',
     'swatch_img'))
 AS T1));
```

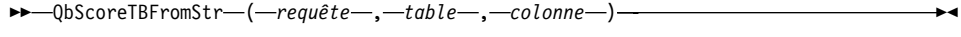
QbScoreTBFromStr

Image	Audio	Vidéo
X		

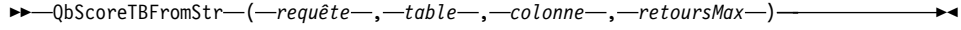
Renvoie une table de scores à partir d'une colonne Image. Chaque score est un chiffre qui exprime le degré de correspondance des caractéristiques de l'image rapport à celles indiquées dans une chaîne de requête. Ce chiffre est calculé par le biais du catalogue QBIC associé à la table et à la colonne auxquelles appartient le descripteur de l'image. Moins ce chiffre est élevé, plus les caractéristiques de l'image correspondent à celles de la chaîne de requête.

Fichier d'inclusion
Aucun.

Syntaxe
Scores renvoyés pour toutes les images cataloguées d'une colonne



Syntaxe
Scores renvoyés pour un nombre spécifique d'images cataloguées d'une colonne



Paramètres (type de données)

requête (VARCHAR(1024))
Chaîne de requête.

table (CHAR (18))
Nom qualifié de la table contenant la colonne Image. Vous pouvez indiquer un nom de table non qualifié si le schéma de table correspond à l'ID utilisateur servant à lancer les services de DB2 Extensions.

colonne (CHAR(18))
Colonne Image à laquelle s'applique la requête.

retoursMax (INTEGER)
Nombre maximal de descripteurs qui seront renvoyés par la table de résultats. Si aucune valeur n'est indiquée, le nombre maximal de descripteurs d'image renvoyés est 100.

Codes retour (type de données)

Table des descripteurs et des scores pour les images de la colonne. La table de résultats comporte deux colonnes : IMAGE_ID (DB2Image), qui contient les descripteurs d'image, et SCORE (DOUBLE), qui contient les scores. Les valeurs de cette table sont en ordre croissant de score. La valeur du score peut aller de 0.0 à une valeur proche de l'infini. Plus cette valeur est faible, plus les valeurs des caractéristiques de l'image cible sont proches de celles spécifiées dans la requête. La valeur 0.0 indique que l'image correspond exactement. La valeur -1 indique que l'image n'a pas été cataloguée.

Exemples

Recherche les 10 images cataloguées contenues dans une colonne de table dont la texture est la plus proche de celle d'un fichier serveur :

```
SELECT name, description
FROM fabric
WHERE CAST (swatch_img as varchar(250)) IN
 (SELECT CAST (image_id as varchar(250)) FROM TABLE
 (QbScoreTBFromStr
 (QbTextureFeatureClass file=<server,"patterns/ptrn07.gif">'
      'clothes.fabric',
      'swatch_img',
      10))
  AS T1));
```

Replace

Image	Audio	Vidéo
X	X	X

Met à jour le contenu d'un objet image, audio ou vidéo stocké dans une base de données, ainsi que le commentaire correspondant.

Fichier d'inclusion

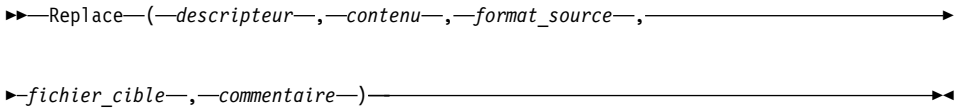
image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

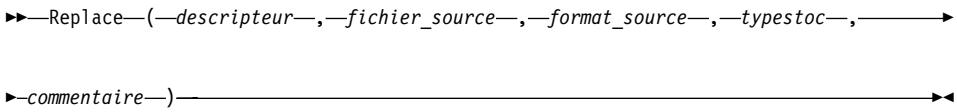
Syntaxe

Mise à jour du contenu à partir d'une mémoire tampon ou d'un fichier client



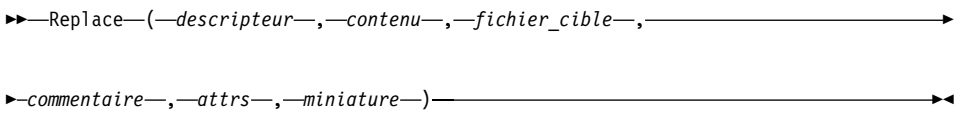
Syntaxe

Mise à jour du contenu à partir du fichier serveur et mise à jour du commentaire



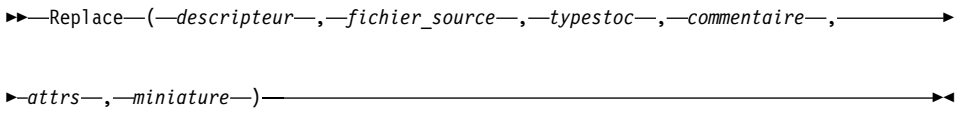
Fichier d'inclusion

Mise à jour du contenu par des attributs définis par l'utilisateur à partir d'une mémoire tampon ou d'un fichier client et mise à jour du commentaire



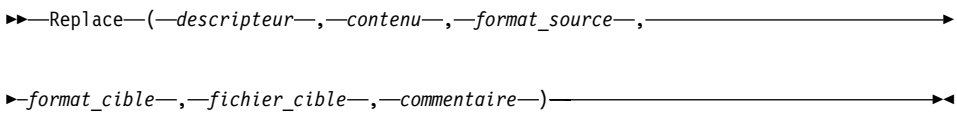
Fichier d'inclusion

Mise à jour du contenu par des attributs définis par l'utilisateur à partir d'un fichier serveur et mise à jour du commentaire



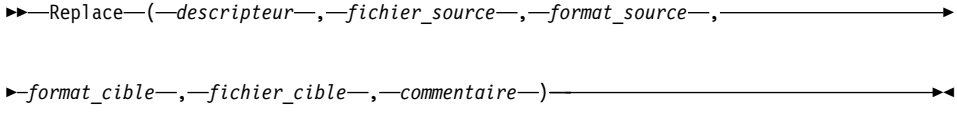
Syntaxe

Mise à jour du contenu à partir d'une mémoire tampon ou d'un fichier client avec conversion de format et mise à jour du commentaire (image uniquement)



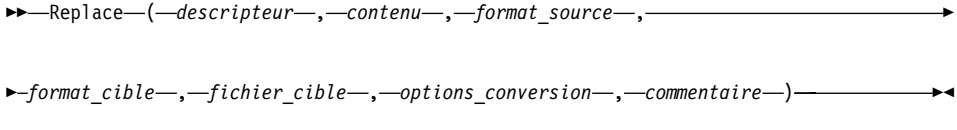
Syntaxe

Mise à jour du contenu à partir d'un fichier serveur avec conversion de format et mise à jour du commentaire (image uniquement)



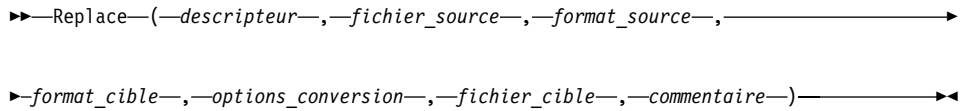
Syntaxe

Mise à jour du contenu à partir d'une mémoire tampon ou d'un fichier client avec conversion de format, modifications supplémentaires et mise à jour du commentaire (image uniquement)



Syntaxe

Mise à jour du contenu à partir d'un fichier serveur avec conversion de format et modifications supplémentaires et mise à jour du commentaire (image uniquement)



Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

fichier_source (LONG VARCHAR)
Nom du fichier dans lequel se trouve le contenu pour mise à jour de l'objet image, audio ou vidéo.

fichier_cible(LONG VARCHAR)
Nom du fichier dans lequel se trouve le contenu de l'objet image, audio ou vidéo à mettre à jour.

création_cible (INTEGER)
Valeur indiquant si un fichier cible doit être créé si le contenu de l'objet source se trouve dans un fichier du serveur. Cette valeur peut être 0 ou 1. La valeur 0 signifie que le fichier cible n'est pas créé (l'extraction n'a pas lieu). La valeur 1 indique que le fichier cible est créé (s'il existe déjà, cette valeur indique son écrasement). Si le contenu de l'objet source se trouve dans un objet BLOB, le fichier cible est toujours créé (s'il existe déjà, il est écrasé).

format_cible (VARCHAR(8))
Format de l'image après extraction. Le format de l'image source est converti en fonction des besoins. Si le contenu est mis à jour avec conversion de format, le chemin d'accès au fichier cible doit être indiqué dans les variables d'environnement DB2IMAGEPATH et DB2MMPATH. Pour le format MPG1, vous pouvez indiquer MPG1, mpg1, MPEG1 ou mpeg1. Pour le format MPG2, vous pouvez indiquer MPG2, mpg2, MPEG2 ou mpeg2.

contenu (BLOB(2G) AS LOCATOR)
Variable SQL dans laquelle est stocké le contenu de la mise à jour d'un objet image, audio ou vidéo. Cette variable peut être de type BLOB, BLOB_FILE ou BLOB_LOCATOR. DB2 transmet le type de données à BLOB_LOCATOR, et le releveur de coordonnées LOB à la fonction UDF Replace.

format_source (VARCHAR(8))
Format de la source pour la mise à jour de l'objet image, audio ou vidéo. Il peut s'agir d'une valeur nulle ou d'une chaîne de caractères vide, ou, pour les images uniquement, d'une chaîne de caractères ASIS ; dans ces trois cas, l'extension tente de déterminer le format automatiquement. Pour le format MPG1, vous pouvez indiquer MPG1, mpg1, MPEG1 ou mpeg1. Pour le format MPG2, vous pouvez indiquer MPG2, mpg2, MPEG2 ou mpeg2.

commentaire (LONG VARCHAR)
Commentaire.

attrs (LONG VARCHAR FOR BIT DATA)
Attributs de l'objet image, audio ou vidéo

miniature (LONG VARCHAR FOR BIT DATA)
Version miniature de l'image de base de données ou de vidéo (uniquement pour ces deux types d'images)

options_conversion (VARCHAR(100))
Indique les modifications (rotation ou compression, par exemple) à appliquer à l'image lors de sa mise à jour. Pour connaître les options de conversion admises, reportez-vous au tableau 6 à la page 93.

Codes retour (type de données)
Descripteur de l'objet image, audio ou vidéo à mettre à jour (DB2IMAGE, DB2AUDIO ou DB2VIDEO).

Exemples
Mise à jour de l'image d'Anne Dupont dans la colonne Photo de la table Employés, conversion du format BMP de l'image en format GIF :

```
EXEC SQL BEGIN DECLARE SECTION;
  long hvStorageType;
EXEC SQL END DECLARE SECTION;

hvStorageType = MMDB_STORAGE_TYPE_INTERNAL;

EXEC SQL UPDATE EMPLOYEE
  SET PICTURE = REPLACE(PICTURE,
        '/Employés/nouvimg/adupont.bmp',
        'BMP',
        'GIF',
        :hvStorageType,
        'Nouvelle photo d''Anne')
  WHERE NAME='Anne Dupont';
```

SamplingRate

Image	Audio	Vidéo
	X	X

Renvoie la fréquence d'échantillonnage d'une séquence audio de type WAVE ou AIFF, ou d'une piste audio d'une vidéo, en nombre d'échantillons par seconde.

Fichier d'inclusion

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio ou vidéo.

Codes retour (type de données)

Fréquence d'échantillonnage d'une vidéo ou d'une séquence audio de type WAVE ou AIFF (INTEGER). Renvoie une valeur nulle pour les séquences audio de format différent.

Exemples

Extraction du nom de fichier de toutes les séquences audio de la colonne Son de la table Employés dont la fréquence d'échantillonnage est 44.1 KHz :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvAud_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME (SOUND)
      INTO :hvAud_fname
       FROM EMPLOYEE
     WHERE SAMPLINGRATE(SOUND) = 44100;
```

Size

Image	Audio	Vidéo
X	X	X

Renvoie, en octets, la taille d'un objet image, audio ou vidéo.

Fichier d'inclusion

image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

Codes retour (type de données)

Taille, en octets, d'un objet image, audio ou vidéo (INTEGER).

Exemples

Extraction du nom de fichier de toutes les images de la colonne photo de la table Employés dont la taille est supérieure à 310 Ko :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(PICTURE)
INTO :hvImg_fname
       FROM EMPLOYEE
     WHERE SIZE(PICTURE) > 310000;
```

Thumbnail

Image	Audio	Vidéo
X		X

Renvoie ou met à jour une version miniature d'un objet image ou vidéo stocké dans une base de données.

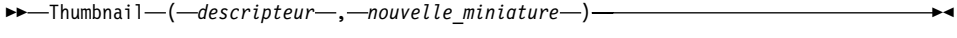
Fichier d'inclusion

image dmbimage.h

vidéo dmbvideo.h

Syntaxe
Extraction d'une miniature

Syntaxe
Mise à jour d'une miniature



Paramètres (type de données)

descripteur (DB2IMAGE ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de l'image ou de la vidéo.

nouvelle_miniature (LONG VARCHAR FOR BIT DATA)
Contenu source pour la mise à jour de la miniature.

Codes retour (type de données)
Pour l'extraction, contenu de la miniature extraite (LONG VARCHAR FOR BIT DATA) pour la mise à jour, descripteur de l'image ou de la vidéo (DB2IMAGE or DB2VIDEO).

Exemples
Extraction de la version miniature de l'image d'Anne Dupont stockée dans la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
 struct{
       short len;
       char data [32000];
         }hvThumbnail;
EXEC SQL END DECLARE SECTION;
```

```
EXEC SQL SELECT THUMBNAIL(PICTURE)
     INTO :hvThumbnail
     FROM EMPLOYEE
     WHERE NAME = 'Anne Dupont';
```

Mise à jour de la version miniature associée à la vidéo d'Anne Dupont dans la table Employés :

```
EXEC SQL BEGIN DECLARE SECTION;
         struct  {
        short len;
       char data[10000];
}hvThumbnail;
  EXEC SQL END DECLARE SECTION;

/*  Création d'une miniature et    */
/*  stockage dans hvThumbnail    */

EXEC SQL UPDATE EMPLOYEE
   SET VIDEO=THUMBNAIL(
               VIDEO,
               :hvThumbnail)
       WHERE NAME='Anne Dupont';
```

TicksPerQNote

Image	Audio	Vidéo
	X	

Renvoie les battements de métronome par noire d'une séquence audio de type MIDI enregistrée.

Fichier d'inclusion
dmbaudio.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio.

Codes retour (type de données)
Nombre de battements de métronome par noire d'une séquence audio de type MIDI (SMALLINT). Renvoie une valeur nulle pour les séquences audio de format différent.

Exemples
Extraction des noms de fichiers de toutes les séquences audio de type MIDI stockées dans la colonne Son de la table Employés qui ont été enregistrées à une vitesse supérieure à 200 battements de métronome par noire :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvAud_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME (SOUND)
      INTO :hvAud_fname
       FROM EMPLOYEE
      WHERE FORMAT(SOUND)='MIDI'
      AND TICKSPERQNOTE(SOUND)>200;
```

TicksPerSec

Image	Audio	Vidéo
	X	

Renvoie les battements de métronome par seconde d'une séquence audio de type MIDI enregistrée.

Fichier d'inclusion
dmbaudio.h

Syntaxe

Paramètres (type de données)

descripteur (DB2AUDIO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence audio.

Codes retour (type de données)
Nombre de battements de métronome par seconde d'une séquence audio de type MIDI (SMALLINT). Renvoie une valeur nulle pour les séquences audio de format différent.

Exemples
Extraction des noms de fichiers de toutes les séquences audio de type MIDI stockées dans la colonne Son de la table Employés qui ont été enregistrées à une vitesse inférieure à 50 battements de métronome par seconde :

```
EXEC SQL BEGIN DECLARE SECTION;
 char hvAud_fname[251];
EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(SOUND)
      INTO :hvAud_fname
      FROM EMPLOYEE
      WHERE FORMAT(SOUND)='MIDI'
      AND TICKSPERSEC(SOUND)<50;
```

Updater

Image	Audio	Vidéo
X	X	X

Renvoie l'ID de l'utilisateur qui a effectué la dernière mise à jour d'un objet image audio ou vidéo dans une table de base de données.

Fichier d'inclusion

image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

Codes retour (type de données)

ID de l'utilisateur qui a effectué la dernière mise à jour d'un objet image, audio ou vidéo.

Exemples

Extraction de l'ID de l'utilisateur ayant effectué la dernière mise à jour de la vidéo stockée dans la colonne vidéo de la table Employés pour Robert Durand :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvUpdater[30];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT UPDATER(VIDEO)
     INTO :hvUpdater
       FROM EMPLOYEE
     WHERE NAME='rdurand';
```

UpdateTime

Image	Audio	Vidéo
X	X	X

Renvoie un horodateur indiquant la date de la dernière mise à jour d'un objet image, audio ou vidéo dans une table de base de données.

Fichier d'inclusion

image dmbimage.h

audio dmbaudio.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2IMAGE, DB2AUDIO ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de la séquence image, audio ou vidéo.

Codes retour (type de données)

Horodatage indiquant la date de la dernière mise à jour d'un objet image, audio ou vidéo (TIMESTAMP).

Exemples

Extraction des noms de fichiers pour les images de la colonne Photo de la table Employés qui ont été mises à jour au cours des deux derniers jours :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(PICTURE)
INTO :hvImg_fname
       FROM EMPLOYEE
     WHERE(CURRENT TIMESTAMP -
       UPDATETIME(PICTURE))< 2;
```

Width

Image	Audio	Vidéo
X		X

Renvoie la largeur en pixels d'un objet image ou vidéo.

Fichier d'inclusion

image dmbimage.h

vidéo dmbvideo.h

Syntaxe

Paramètres (type de données)

descripteur (DB2IMAGE ou DB2VIDEO)
Nom de colonne ou variable SQL contenant le descripteur de l'image ou de la vidéo.

Codes retour (type de données)

Largeur, en pixels (INTEGER).

Exemples

Extraction du nom de fichier de toutes les images de la colonne Photo de la table Employés d'une largeur inférieure à 300 pixels :

```
EXEC SQL BEGIN DECLARE SECTION;
char hvImg_fname[251];
  EXEC SQL END DECLARE SECTION;

EXEC SQL SELECT FILENAME(PICTURE)
INTO :hvImg_fname
       FROM EMPLOYEE
     WHERE WIDTH(PICTURE)<300;
```

Chapitre 16. Interfaces de programmation d'applications

Le présent chapitre fournit des informations sur les API d'administration de DB2 Extensions. Les API sont répertoriées dans l'ordre alphabétique.

Pour chaque API, les informations suivantes sont précisées :

- l'extension qui fournit l'API ;
- une brève description ;
- les droits nécessaires pour utiliser l'API ;
- le fichier bibliothèque de l'API ;
- le fichier (en-tête) d'inclusion de la fonction API ;
- la syntaxe en langage C de l'appel d'API correspondant ;
- une description des paramètres d'API ;
- les valeurs renvoyées par l'API ;
- des exemples.

DBaAdminGetInaccessibleFiles

Image	Audio	Vidéo
	X	

Renvoie les noms des fichiers inaccessibles référencés dans les colonnes Audio des tables utilisateur. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaAdminGetInaccessibleFiles(
      char *qualifier,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

qualifier (entrée)
ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, la recherche porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables de la base de données en cours.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers inaccessibles référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

SQL_ERROR ou autres codes retour SQL
Erreur renvoyée par DB2.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

Exemples

Liste de tous les fichiers inaccessibles référencés dans les colonnes Audio des tables associées à l'ID utilisateur rdurand.

```
#include <dmbaudio.h>
long idx;

rc = DBaAdminGetInaccessibleFiles("rdurand",
       &count, &filelist);
```

DBaAdminGetReferencedFiles

Image	Audio	Vidéo
	X	

Renvoie les noms des fichiers référencés dans les colonnes Audio des tables utilisateur. Si un fichier est inaccessible (en raison, par exemple, de la non résolution de son nom à l'aide des spécifications de variable d'environnement), le nom de ce fichier est précédé du signe astérisque (*). Cette API n'utilise pas la zone FILENAME de la structure de données FILEREF. Par conséquent, elle attribue la valeur NULL à cette zone. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez annuler la structure de données de la liste de fichiers (filelist).

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaAdminGetReferencedFiles(
      char *qualifier,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

qualifier (entrée)

ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, la recherche porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables de la base de données en cours.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

Exemples

Liste de tous les fichiers référencés dans les colonnes audio des tables appartenant à l'ID utilisateur adupont.

```
#include <dmbaudio.h>
long idx;

rc = DBaAdminGetReferencedFiles("adupont",
       &count, &fileList);
```

DBaAdminIsFileReferenced

Image	Audio	Vidéo
	X	

Renvoie la liste des entrées de la colonne Audio dans les tables utilisateur faisant référence au fichier indiqué. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaAdminIsFileReferenced(
      char *qualifier,
      char *fileName,
      long *count,
      FILEREF *(*tableList)
      );
```

Paramètres

qualifier (entrée)
ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, la recherche porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables de la base de données en cours.

fileName (entrée)
Nom du fichier référencé.

count (sortie)
Nombre d'entrées dans la liste de sortie.

tableList (sortie)
Liste des entrées de table faisant référence au fichier indiqué.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

Exemples

Liste des entrées dans les colonnes audio de toutes les tables de la base de données qui font référence au fichier /audios/rdurand.wav :

```
#include <dmbaudio.h>
long idx; rc =
DBaAdminIsFileReferenced(NULL,
       "/audios/rdurand.wav",
       &count, &tableList);
```

DBaAdminReorgMetadata

Image	Audio	Vidéo
	X	

«Rafraîchit» les tables de métadonnées associées à l'extension audio en exécutant par exemple les opérations suivantes :

- récupération de l'espace qui n'est plus utilisé dans les tables de métadonnées audio ;
- suppression, dans les tables de métadonnées audio, de références à des fichiers audio qui n'existent plus.

Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaAdminReorgMetadata(
      char *qualifier
      );
```

Paramètres

qualifier (entrée)
ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, le nettoyage porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, le nettoyage porte sur toutes les tables de la base de données en cours.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

Exemples

Nettoyage des tables de métadonnées pour les colonnes Audio des tables associées à l'ID utilisateur rdurand.

```
#include <dmbaudio.h>

rc = DBaAdminReorgMetadata("rdurand");
```

DBaDisableColumn

Image	Audio	Vidéo
	X	

Désactive une colonne de sorte que les données DB2Audio ne soient plus prises en charge. Les entrées de la colonne sont définies par NULL et les métadonnées associées à cette colonne sont supprimées. Tous les déclencheurs définis par l'extension Audio pour cette colonne sont également supprimés. De nouvelles lignes peuvent être insérées dans la table contenant la colonne désactivée. Ces nouvelles lignes peuvent contenir des données de type DB2Audio mais aucune métadonnée (dans les tables de gestion) ne leur est associée. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données. Une fois l'API appelée, iI est recommandé d'émettre l'instruction SQL COMMIT.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaDisableColumn(
      char *tableName,
      char *colName,
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant la colonne Audio.

colName (entrée)
Nom de la colonne Audio.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Désactivation de la colonne Son dans la table Employés pour les données DB2Audio :

```
#include <dmbaudio.h>

rc = DBaDisableColumn("Employés", "son");
```

DBaDisableDatabase

Image	Audio	Vidéo
	X	

Désactive une base de données de sorte que les données DB2Audio ne soient plus prises en charge. Toutes les tables de la base de données définies pour DB2Audio sont également désactivées. Les métadonnées et les fonctions UDF définies par l'extension Audio pour la base de données sont supprimées. De nouvelles lignes peuvent être insérées dans des tables de type DB2Audio mais aucune métadonnée (dans les tables de gestion) ne leur est associée. Une fois l'API appelée, iI est recommandé d'émettre l'instruction SQL COMMIT.

Classe d'autorisation

DBADM, SYSADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaDisableDatabase(
     );
```

Paramètres

L'API DBaDisableDatabase ne comporte aucun paramètre.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Désactivation de la base de données en cours pour les données DB2Audio :

```
#include <dmbaudio.h>

rc = DBaDisableDatabase();
```

DBaDisableTable

Image	Audio	Vidéo
	X	

Désactive une table de sorte que les données DB2Audio ne soient plus prises en charge. Toutes les colonnes de la table définie pour DB2Audio sont également désactivées. Certaines métadonnées définies par l'extension Audio pour la table sont supprimées. De nouvelles lignes peuvent être insérées dans des tables de type DB2Audio, mais aucune métadonnée (dans les tables de gestion) ne leur est associée. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données. Une fois l'API appelée, iI est recommandé d'émettre l'instruction SQL COMMIT.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaDisableTable(
      char *tableName
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant une colonne Audio.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Désactivation de la table Employés pour les données DB2Audio :

```
#include <dmbaudio.h>

rc = DBaDisableTable("Employés");
```

DBaEnableColumn

Image	Audio	Vidéo
	X	

Active une colonne pour les données DB2Audio. L'API définit et gère les relations entre cette colonne et les tables de métadonnées. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données. Une fois l'API appelée, iI est recommandé d'émettre l'instruction SQL COMMIT.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Le privilège USE (utiliser) est également obligatoire pour les espaces table et les pools de mémoire tampon indiqués dans les paramètres de l'API.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaEnableColumn(
      char *tableName,
      char *colName,
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant la colonne Audio.

colName (entrée)
Nom de la colonne Audio.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La colonne est déjà activée.

MMDB_RC_WRONG_SIGNATURE
Le type de données de la colonne indiquée est incorrect. Le système attend le type de données défini par l'utilisateur (MMDBSYS.DB2AUDIO).

MMDB_RC_COLUMN_DOESNOT_EXIST
La colonne n'est pas définie dans la table indiquée.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NOT_ENABLED
La base de données ou la table n'est pas activée.

Exemples

Activation de la colonne Son dans la table Employés pour les données DB2Audio :

```
#include <dmbaudio.h>

rc = DBaEnableColumn("Employés", "son");
```

DBaEnableDatabase

Image	Audio	Vidéo
	X	

Active une base de données pour les données DB2Audio. Cette API est appelée une fois pour chaque base de données. Elle définit un type DB2 défini par l'utilisateur, DB2Audio, dans le gestionnaire de bases de données. Elle crée également toutes les fonctions UDF manipulant des données DB2Audio. Une fois l'API appelée, iI est recommandé d'émettre l'instruction SQL COMMIT.

Classe d'autorisation

DBADM, SYSADM, SYSCTRL

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaEnableDatabase(
      char *tableSpace
      );
```

Paramètres

tableSpace (entrée)

Nom de l'espace table (ensemble de conteneurs dans lesquels des tables de gestion sont stockées). La spécification de l'espace table se compose des trois parties suivantes : *datats, indexts, longts*, *datats* étant l'espace table dans lequel sont créées les tables de métadonnées ; *indexts* l'espace table dans lequel sont créés les index des tables de métadonnées et *longts* l'espace table dans lequel sont stockées les valeurs contenues dans les colonnes longues des tables de métadonnées (telles que celles contenant les types de données LONG VARCHAR et LOB). Si vous indiquez une valeur nulle pour n'importe quelle partie de la spécification de l'espace table, l'espace table par défaut pour cette partie est utilisé.

Produit EEE uniquement : Les espaces table spécifiés lors de l'activation d'une base de données pour une extension doivent être

définis sur un groupe de noeuds comprenant tous les noeuds dans le système de bases de données partitionnées.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La base de données est déjà active.

MMDB_RC_API_NOT_SUPPORTED_FOR_SERVER
Le serveur connecté ne prend pas en charge cette commande.

MMDB_WARN_NOT_ALL_NODES
L'espace table spécifié ne comprend pas tous les noeuds pour l'extension. **(Produit EEE uniquement)**

MMDB_RC_NOT_SAME_NODEGROUP
Les espaces table spécifiés ne font pas partie du même groupe de noeuds. **(Produit EEE uniquement)**

Exemples

Activation de la base de données en cours pour les données DB2Audio dans l'espace table MYTS. Utilisation des valeurs par défaut pour les espaces table longs et d'indexation :

```
#include <dmbaudio.h>

rc = DBaEnableDatabase("myts,,");
```

Activation de la base de données en cours pour les données DB2Audio. Utilisation des espaces tables par défaut :

```
#include <dmbaudio.h>

rc = DBaEnableDatabase(NULL);
```

DBaEnableTable

Image	Audio	Vidéo
	X	

Active une table pour les données DB2Audio. Cette API est appelée une fois pour chaque table. Elle crée des tables de métadonnées pour le stockage et la gestion des attributs des colonnes Audio dans une table. Pour éviter tout risque de verrouillage, l'application doit valider des transactions avant d'appeler cette API. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données. Une fois l'API appelée, iI est recommandé d'émettre l'instruction SQL COMMIT.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaEnableTable(
      char *tableSpace,
      char *tableName
      );
```

Paramètres

tableSpace (entrée)

Nom de l'espace table (ensemble de conteneurs dans lesquels des tables de gestion sont stockées). La spécification de l'espace table se compose des trois parties suivantes : *datats*, *indexts*, *longts*, *datats* étant l'espace table dans lequel sont créées les tables de métadonnées ; *indexts* l'espace table dans lequel sont créés les index des tables de métadonnées et *longts* l'espace table dans lequel sont stockées les valeurs contenues dans les colonnes longues des tables de métadonnées (telles que celles contenant les types de données LONG VARCHAR et LOB). Si vous indiquez une valeur nulle pour l'une des parties de la spécification de l'espace table, l'espace table par défaut est utilisé pour cette partie.

Si vous indiquez une valeur nulle pour n'importe quelle partie de la spécification de l'espace table, l'espace table par défaut pour cette partie est utilisé.

Produit EEE uniquement : L'espace table spécifié doit appartenir au même groupe de noeuds que la table utilisateur.

tableName (entrée)
Nom de la table devant contenir une colonne Audio.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La table est déjà activée.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_TABLE_DOESNOT_EXIST
La table n'existe pas.

MMDB_RC_TABLESPACE_NOT_SAME_NODEGROUP
L'espace table spécifié n'appartient pas au même groupe de noeuds que la table utilisateur. **(Produit EEE uniquement)**

Exemples

Activation de la table Employés pour les données DB2Audio dans l'espace table MYTS. Utilisation des valeurs par défaut pour les espaces table longs et d'indexation :

```
#include <dmbaudio.h>

rc = DBaEnableTable("myts,,",
       "Employés");
```

Activation de la table Employés pour les données DB2Audio. Utilisation des espaces table par défaut :

```
#include <dmbaudio.h>

rc = DBaEnableTable(NULL,
       "Employés");
```

DBaGetError

Image	Audio	Vidéo
	X	

Renvoie une description de la dernière erreur. Vous pouvez appeler cette API lorsqu'une autre API renvoie un code d'erreur.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaGetError(
      SQLINTEGER *sqlcode,
      char *errorMsgText
      );
```

Paramètres

sqlcode (sortie)
Code d'erreur SQL générique.

errorMsgText (sortie)
Texte du message d'erreur SQL.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

Exemples

Extraction de la dernière erreur avec insertion du code d'erreur SQL dans le descripteur `errCode` et du texte du message dans le descripteur `errMsg`.

```
#include <dmbaudio.h>

rc = DBaGetError(&errCode, &errMsg);
```

DBaGetInaccessibleFiles

Image	Audio	Vidéo
	X	

Renvoie les noms des fichiers inaccessibles référencés dans les colonnes Audio des tables utilisateur. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SELECT sur les colonnes audio activées dans toutes les tables utilisateur sur lesquelles porte la recherche, ainsi que les tables de gestion associées.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaGetInaccessibleFiles(
      char *tableName,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié, non qualifié ou nul. Si un nom de table est indiqué, la recherche des fichiers inaccessibles porte sur cette table. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables ayant le qualificatif indiqué.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers inaccessibles référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste de tous les fichiers inaccessibles référencés dans les colonnes audio de la table Employés.

```
long idx;
#include <dmbaudio.h>

rc = DBaGetInaccessibleFiles("Employés",
       &count, &filelist);
```

DBaGetReferencedFiles

Image	Audio	Vidéo
	X	

Renvoie les noms des fichiers référencés dans les colonnes Audio des tables utilisateur. Si un fichier est inaccessible (en raison, par exemple, de la non résolution de son nom à l'aide des spécifications de variable d'environnement), le nom de ce fichier est précédé du signe astérisque (*). Cette API n'utilise pas la zone FILENAME de la structure de données FILEREF. Par conséquent, elle attribue la valeur NULL à cette zone. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez annuler la structure de données de la liste de fichiers (filelist).

Classe d'autorisation

SELECT sur les colonnes audio activées dans toutes les tables utilisateur sur lesquelles porte la recherche, ainsi que les tables de gestion associées.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaGetReferencedFiles(
      char *tableName,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

tableName (entrée)

Nom de table qualifié, non qualifié ou nul. Si un nom de table est indiqué, la recherche des fichiers porte sur cette table. Si une valeur nulle est indiquée, la recherche porte sur toutes les tables associées à la base de données à laquelle l'ID utilisateur est connecté.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste de tous les fichiers référencés dans les colonnes audio de la table Employés.

```
#include <dmbaudio.h>
long idx;

rc = DBaGetReferencedFiles("Employés",
       &count, &filelist);
```

DBaIsColumnEnabled

Image	Audio	Vidéo
	X	

Détermine si une colonne a été activée pour les données DB2Audio. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Droits SYSADM, DBADM, être propriétaire de la table ou disposer sur celle-ci du privilège SELECT.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaIsColumnEnabled(
      char *tableName,
      char *colName,
      short *status
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié ou non qualifié.

colName (entrée)
Nom d'une colonne.

status (sortie)
Indique si la colonne est activée. Ce paramètre renvoie une valeur numérique. L'extension renvoie également une constante qui indique l'état de la base. Les valeurs et les constantes sont les suivantes :

1 MMDB_IS_ENABLED

0 MMDB_IS_NOT_ENABLED

-1 MMDB_INVALID_DATATYPE

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Déterminer si la colonne Son de la table Employés est activée pour DB2Audio :

```
#include <dmbaudio.h>

rc = DBaIsColumnEnabled("Employés",
       "son", &status);
```

DBaIsDatabaseEnabled

Image	Audio	Vidéo
	X	

Détermine si une base de données a été activée pour les données DB2Audio. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaIsDatabaseEnabled(
      short *status
      );
```

Paramètres

status (sortie)
Indique si la base de données est activée. Ce paramètre renvoie une valeur numérique. L'extension renvoie également une constante qui indique l'état de la base. Les valeurs et les constantes sont les suivantes :

1 MMDB_IS_ENABLED

0 MMDB_IS_NOT_ENABLED

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Déterminer si la base de données Personnl est activée pour les données DB2Audio :

```
#include <dmbaudio.h>

rc = DBaIsDatabaseEnabled(&status);
```

DBaIsFileReferenced

Image	Audio	Vidéo
	X	

Renvoie la liste des entrées de table faisant référence au fichier indiqué. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SELECT sur les colonnes audio activées dans toutes les tables utilisateur sur lesquelles porte la recherche, ainsi que les tables de gestion associées.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaIsFileReferenced(
      char *tableName,
      char *fileName,
      long *count,
      FILEREF *(*tableList)
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié, non qualifié ou nul. Si un nom de table est indiqué, la recherche du fichier indiqué porte sur cette table. Si une valeur nulle est indiquée, la recherche porte sur toutes les tables appartenant à l'ID utilisateur en cours.

fileName (entrée)
Nom du fichier référencé.

count (sortie)
Nombre d'entrées dans la liste de sortie.

tableList (sortie)
Liste des entrées de table faisant référence au fichier indiqué.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste des entrées dans les colonnes audio de la table Employés faisant référence au fichier /audios/adupont.wav :

```
#include <dmbaudio.h>
long idx;

rc = DBaIsFileReferenced(NULL,
       "/audios/adupont.wav",
       &count, &tableList);
```

DBaIsTableEnabled

Image	Audio	Vidéo
	X	

Détermine si une table a été activée pour les données DB2Audio. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaIsTableEnabled(
      char *tableName,
      short *status
      );
```

Paramètres

tableName (entrée)
Nom d'une table.

status (sortie)
Indique si la table est activée. Ce paramètre renvoie une valeur numérique. L'extension renvoie également une constante qui indique l'état de la base. Les valeurs et les constantes sont les suivantes :

1 MMDB_IS_ENABLED

0 MMDB_IS_NOT_ENABLED

-1 MMDB_INVALID_DATATYPE

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Déterminer si la table Employés est activée pour les données DB2Audio :

```
#include <dmbaudio.h>

rc = DBaIsTableEnabled("Employés", &status);
```

DBaPlay

Image	Audio	Vidéo
	X	

Ouvre le lecteur sur le poste client et lit une séquence audio. La séquence peut être stockée dans une colonne Audio ou un fichier externe.

- Si la séquence audio est stockée dans un fichier externe, vous pouvez communiquer le nom de ce fichier ou le descripteur audio à l'API qui utilise la variable d'environnement DB2AUDIOPATH pour déterminer l'emplacement du fichier. Le fichier doit être accessible à partir du poste client.
- Si la séquence audio est stockée dans une colonne, vous devez communiquer le descripteur audio à l'API. L'application doit être connectée à la base de données et avoir accès en lecture à la table dans laquelle les données vidéo sont stockées.

Si les données audio sont stockées dans une colonne, l'extension crée un fichier temporaire et copie dans celui-ci le contenu de l'objet à partir de la colonne. L'extension peut également créer un fichier temporaire si les données audio sont stockées dans un fichier externe si le nom relatif de celui-ci ne peut pas être résolu à l'aide des variables d'environnement ou si ce fichier n'est pas accessible sur le poste client. Ce fichier est créé dans le répertoire désigné par la variable d'environnement DB2AUDIOTEMP. L'extension lit ensuite la séquence audio à partir de ce fichier temporaire.

Classe d'autorisation

SELECT sur la table utilisateur si une séquence audio est lue à partir d'une colonne.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

DBaPlay

Syntaxe

Lecture d'un objet audio stocké dans une colonne

```
long DBaPlay(
      char *playerName,
      MMDB_PLAY_HANDLE,
      DB2Audio *audioHandle,
      waitFlag
      );
```

Syntaxe

Lecture d'un objet audio stocké dans un fichier

```
long DBaPlay(
      char *playerName,
      MMDB_PLAY_FILE,
      char *fileName,
      waitFlag
      );
```

Paramètres

playerName (entrée)
Nom du lecteur audio. Si ce paramètre a une valeur NULL, le lecteur audio par défaut défini par la variable d'environnement DB2AUDIOPLAYER est utilisé.

MMDB_PLAY_HANDLE (entrée)
Constante qui indique que les données audio sont stockées en tant qu'objet BLOB.

MMDB_PLAY_FILE (entrée)
Constante qui indique que les données audio sont stockées sous forme de fichier accessible à partir du poste client.

audioHandle (entrée)
Descripteur de la séquence audio. Ce paramètre doit être fourni lorsque vous lisez une séquence audio se trouvant dans une colonne. Si le descripteur audio représente un fichier externe, la variable d'environnement DB2VIDEOPATH est utilisée pour déterminer l'emplacement du fichier.

fileName (entrée)
Nom du fichier contenant les données audio.

waitFlag (entrée)
Constante qui indique si le programme attend que l'utilisateur arrête le lecteur avant de continuer. Le paramètre MMDB_PLAY_WAIT entraîne la mise en oeuvre du lecteur dans la même unité d'exécution que l'application. Le paramètre MMDB_PLAY_NO_WAIT entraîne la mise en oeuvre du lecteur dans une unité d'exécution distincte.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Lecture de la séquence audio identifiée par le descripteur audio. Mise en oeuvre du lecteur par défaut dans la même unité d'exécution que l'application.

```
#include <dmbaudio.h>

rc = DBaPlay(NULL, MMDB_PLAY_HANDLE,
       audioHandle, MMDB_PLAY_WAIT);
```

DBaPrepareAttrs

Image	Audio	Vidéo
	X	

Prépare les attributs audio fournis par l'utilisateur. Cette API est utilisée lors du stockage ou de la mise à jour d'un objet audio muni d'attributs fournis par l'utilisateur. Le code UDF qui s'exécute sur le serveur attend toujours des données au format «big endian», utilisé sur la plupart des plateformes UNIX. Si un objet audio est stocké ou mis à jour au format «little endian», c'est-à-dire à partir d'un client non UNIX, l'API DBaPrepare doit être appelée pour que la demande de stockage ou de mise à jour puisse aboutir.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudio.lib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
void DBaPrepareAttrs(
      MMDBAudioAttrs *audAttr
      );
```

Paramètres

audAttr (entrée)
Attributs des données audio fournis par l'utilisateur.

Exemples

Préparation des attributs des données audio fournis par l'utilisateur :

```
#include <dmbaudio.h>

DBaPrepareAttrs(&imgattr);
```

DBaReorgMetadata

Image	Audio	Vidéo
	X	

«Rafraîchit» les tables de métadonnées associées à l'extension audio en exécutant par exemple les opérations suivantes :

- récupération de l'espace qui n'est plus utilisé dans les tables de métadonnées audio ;
- suppression, dans les tables de métadonnées audio, de références à des fichiers audio qui n'existent plus.

Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Alter, Control, SYSADM, SYSCTRL, SYSMAINT, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbaudiolib	libdmbaudio.a (AIX) libdmbaudio.sl (HP-UX) libdmbaudio.so (Solaris)

Fichier d'inclusion

dmbaudio.h

Syntaxe

```
long DBaReorgMetadata(
      char *tableName
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié, non qualifié ou nul. Si un nom de table est spécifié, le nettoyage porte sur les tables de métadonnées audio associées à la table utilisateur indiquée. Lorsqu'une valeur nulle est indiquée, le nettoyage porte sur les tables de métadonnées des colonnes Audio de toutes les tables associées à l'ID utilisateur en cours.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Nettoyage des tables de métadonnées pour les colonnes Audio de la table Employés.

```
#include <dmbaudio.h>

rc = DBaReorgMetadata("Employés");
```

DBiAdminGetInaccessibleFiles

Image	Audio	Vidéo
X		

Renvoie les noms des fichiers inaccessibles référencés dans les colonnes Image des tables utilisateur. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiAdminGetInaccessibleFiles(
      char *qualifier,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

qualifier (entrée)
ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, la recherche porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables de la base de données en cours.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers inaccessibles référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste de tous les fichiers inaccessibles référencés dans les colonnes Image des tables associées à l'ID utilisateur adupont.

```
#include <dmbimage.h>
long idx;

rc = DBiAdminGetInaccessibleFiles
       ("adupont", &count, &filelist);
```

DBiAdminGetReferencedFiles

Image	Audio	Vidéo
X		

Renvoie les noms des fichiers référencés dans les colonnes Image des tables utilisateur. Si un fichier est inaccessible (en raison, par exemple, de la non résolution de son nom à l'aide des spécifications de variable d'environnement), le nom de ce fichier est précédé du signe astérisque (*). Cette API n'utilise pas la zone FILENAME de la structure de données FILEREF. Par conséquent, elle attribue la valeur NULL à cette zone. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez annuler la structure de données de la liste de fichiers (filelist).

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiAdminGetReferencedFiles(
      char *qualifier,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

qualifier (entrée)

ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, la recherche porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables de la base de données en cours.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste de tous les fichiers référencés dans les colonnes Image des tables appartenant à l'ID utilisateur adupont.

```
#include <dmbimage.h>
long idx;

rc = DBiAdminGetReferencedFiles("adupont",
       &count, &filelist);
```

DBiAdminIsFileReferenced

Image	Audio	Vidéo
X		

Renvoie la liste des entrées de la colonne Image dans les tables utilisateur faisant référence au fichier indiqué. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiAdminIsFileReferenced(
      char *qualifier,
      char *fileName,
      long *count,
      FILEREF *(*tableList)
      );
```

Paramètres

qualifier (entrée)
ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, la recherche porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables de la base de données en cours.

fileName (entrée)
Nom du fichier référencé.

count (sortie)
Nombre d'entrées dans la liste de sortie.

tableList (sortie)
Liste des entrées de table faisant référence au fichier indiqué.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste des entrées dans les colonnes image de toutes les tables de la base de données en cours faisant référence au fichier /images/rdurand.bmp :

```
#include <dmbimage.h>
long idx;

rc = DBiAdminIsFileReferenced(NULL,
       "/images/rdurand.bmp",
       &count, &tableList);
```

DBiAdminReorgMetadata

Image	Audio	Vidéo
X		

«Nettoie» les tables de métadonnées associées à l'extension Image en exécutant, par exemple, les opérations suivantes :

- récupération de l'espace qui n'est plus utilisé dans les tables de métadonnées d'image ;
- suppression des références à des fichiers image qui n'existent plus dans les tables de métadonnées d'image.

Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiAdminReorgMetadata(
      char *qualifier
      );
```

Paramètres

qualifier (entrée)
ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, le nettoyage porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, le nettoyage porte sur toutes les tables de la base de données en cours.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

Exemples

Nettoyage des tables de métadonnées pour les colonnes Image dans les tables associées à l'ID utilisateur rdurand.

```
#include <dmbimage.h>

rc = DBiAdminReorgMetadata("rdurand");
```

DBiBrowse

Image	Audio	Vidéo
X		

Ouvre l'afficheur d'images sur le poste client et affiche une image. L'image peut être stockée dans une colonne Image ou dans un fichier externe.

- Si l'image est stockée dans un fichier externe, vous devez communiquer le nom du fichier ou le descripteur d'image à l'API qui utilise la variable d'environnement DB2IMAGEPATH pour déterminer l'emplacement du fichier. Le fichier doit être accessible à partir du poste client.
- Si l'image est stockée dans une colonne, vous devez communiquer le descripteur d'image à l'API. L'application doit être connectée à la base de données et avoir accès en lecture à la table dans laquelle l'image est stockée.

Si l'afficheur ne peut pas accéder directement à l'image, l'extension crée un fichier temporaire dans le répertoire défini dans la variable d'environnement DB2IMAGETEMP. L'extension affiche ensuite l'image à partir du fichier temporaire.

Classe d'autorisation

SELECT sur la table utilisateur si une image est visualisée à partir d'une colonne.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

Lecture d'un objet image stocké dans une colonne

```
long DBiBrowse(
      char *browserName,
      MMDB_PLAY_HANDLE,
      DB2Image *imageHandle,
      waitFlag
      );
```

DBiBrowse

Syntaxe

Lecture d'un objet image stocké dans un fichier

```
long DBiBrowse(
      char *browserName,
      MMDB_PLAY_FILE,
      char *fileName,
      waitFlag
      );
```

Paramètres

browserName (entrée)
Nom de l'afficheur d'images. Si ce paramètre est défini par NULL, l'afficheur d'images par défaut défini par la variable d'environnement DB2IMAGEBROWSER est utilisé.

MMDB_PLAY_HANDLE (entrée)
Constante qui indique que l'image est stockée en tant qu'objet BLOB.

MMDB_PLAY_FILE (entrée)
Constante qui indique que l'image est stockée sous forme de fichier accessible à partir du poste client.

imageHandle (entrée)
Descripteur de l'image. Ce paramètre doit être fourni lorsque vous visualisez une image dans une colonne. Si le descripteur d'image représente un fichier externe, la variable d'environnement client DB2IMAGEPATH est utilisée pour déterminer l'emplacement du fichier.

fileName (entrée)
Nom du fichier contenant l'image.

waitFlag (entrée)
Constante qui indique si le programme attend que l'utilisateur arrête l'afficheur avant de continuer. Le paramètre MMDB_PLAY_WAIT entraîne la mise en oeuvre de l'afficheur dans la même unité d'exécution que l'application. Le paramètre MMDB_PLAY_NO_WAIT entraîne la mise en oeuvre de l'afficheur dans une unité d'exécution distincte.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Affichage de l'image identifiée par le descripteur d'image. Mise en oeuvre de l'afficheur par défaut dans la même unité d'exécution que l'application.

```
#include <dmbimage.h>

rc = DBiBrowse(NULL, MMDB_PLAY_HANDLE,
       imageHandle, MMDB_PLAY_WAIT);
```

DBiDisableColumn

Image	Audio	Vidéo
X		

Désactive une colonne de sorte que les données DB2Image ne soient plus prises en charge. Les entrées de la colonne sont définies par NULL et les métadonnées associées à cette colonne sont supprimées. Le catalogue QBIC associé à cette dernière l'est également. Tous les déclencheurs définis par l'extension Image pour cette colonne sont également supprimés. De nouvelles lignes peuvent être insérées dans la table contenant la colonne désactivée. Ces nouvelles lignes peuvent contenir des données de type DB2Image mais aucune métadonnée (dans les tables de gestion) ne leur est associée. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiDisableColumn(
      char *tableName,
      char *colName,
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant la colonne Image.

colName (entrée)
Nom de la colonne Image.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Désactivation de la colonne Photo dans la table Employés de sorte que les données DB2Image ne soient plus prises en charge :

```
#include <dmbimage.h>

rc = DBiDisableColumn("Employés", "photo");
```

DBiDisableDatabase

Image	Audio	Vidéo
X		

Désactive une base de données de sorte que les données DB2Image ne soient plus prises en charge. Toutes les tables de la base de données définies pour DB2Image sont également désactivées. Les métadonnées et les fonctions UDF définies par l'extension Image pour le serveur de bases de données sont supprimées. De nouvelles lignes peuvent être insérées dans des tables de type DB2Image mais aucune métadonnée (dans les tables de gestion) ne leur est associée.

Classe d'autorisation

DBADM, SYSADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiDisableDatabase(
     );
```

Paramètres

L'API DBiDisableDatabase ne comporte aucun paramètre.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Désactivation de la base de données en cours pour les données DB2Image :

```
#include <dmbimage.h>

rc = DBiDisableDatabase();
```

DBiDisableTable

Image	Audio	Vidéo
X		

Désactive une table de sorte que les données DB2Image ne soient plus prises en charge. Toutes les colonnes de la table définie pour DB2Image sont également désactivées. Certaines métadonnées définies par l'extension Image pour la table sont supprimées. Tous les catalogues QBIC associés aux colonnes d'image dans la table sont également supprimés. De nouvelles lignes peuvent être insérées dans des tables de type DB2Image mais aucune métadonnée (dans les tables de gestion) ne leur est associée. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiDisableTable(
      char *tableName
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant une colonne Image.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Désactivation de la table Employés pour les données DB2Image :

```
#include <dmbimage.h>

rc = DBiDisableTable("Employés");
```

DBiEnableColumn

Image	Audio	Vidéo
X		

Active une colonne pour les données DB2Image. L'API définit et gère les relations entre cette colonne et les tables de métadonnées. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données et la table utilisateur doit être validée.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiEnableColumn(
      char *tableName,
      char *colName,
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant la colonne Image.

colName (entrée)
Nom de la colonne Image.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La colonne est déjà activée.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_WRONG_SIGNATURE
Le type de données de la colonne indiquée est incorrect. Le système attend le type de données défini par l'utilisateur (MMDBSYS.DB2IMAGE).

MMDB_RC_COLUMN_DOESNOT_EXIST
La colonne n'est pas définie dans la table indiquée.

MMDB_RC_NOT_ENABLED
La base de données ou la table n'est pas activée.

Exemples

Activation de la colonne Photo dans la table Employés pour les données DB2Image :

```
#include <dmbimage.h>

rc = DBiEnableColumn("Employés",
       "photo");
```

DBiEnableDatabase

Image	Audio	Vidéo
X		

Active une base de données pour les données DB2Image. Cette API est appelée une fois pour chaque base de données. Elle définit un type DB2 défini par l'utilisateur, DB2Image, dans le gestionnaire de bases de données. Elle crée également toutes les fonctions utilisateur manipulant des données DB2Image.

Classe d'autorisation

DBADM, SYSADM, SYSCTRL

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiEnableDatabase(
      char *tableSpace
      );
```

Paramètres

tableSpace (entrée)

Nom de l'espace table (ensemble de conteneurs dans lesquels des tables de gestion sont stockées). La spécification de l'espace table se compose des trois parties suivantes : *datats*, *indexts*, *longts*, *datats* étant l'espace table dans lequel sont créées les tables de métadonnées ; *indexts* l'espace table dans lequel sont créés les index des tables de métadonnées et *longts* l'espace table dans lequel sont stockées les valeurs contenues dans les colonnes longues des tables de métadonnées (telles que celles contenant les types de données LONG VARCHAR et LOB). Si vous indiquez une valeur nulle pour n'importe quelle partie de la spécification de l'espace table, l'espace table par défaut pour cette partie est utilisé.

Produit EEE uniquement : Les espaces table spécifiés lors de l'activation d'une base de données pour une extension doivent être

définis sur un groupe de noeuds comprenant tous les noeuds dans le système de bases de données partitionnées.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La base de données est déjà active.

MMDB_RC_API_NOT_SUPPORTED_FOR_SERVER
Le serveur connecté ne prend pas en charge cette commande.

MMDB_WARN_NOT_ALL_NODES
L'espace table spécifié ne comprend pas tous les noeuds pour l'extension. **(Produit EEE uniquement)**

MMDB_RC_NOT_SAME_NODEGROUP
Les espaces table spécifiés ne font pas partie du même groupe de noeuds. **(Produit EEE uniquement)**

Exemples

Désactivation de la base de données en cours pour les données DB2Image dans l'espace table MYTS. Utilisation des valeurs par défaut pour les espaces table longs et d'indexation :

```
#include <dmbimage.h>

rc = DBiEnableDatabase("myts,,");
```

Activation de la base de données en cours pour les données DB2Image. Utilisation des espaces table par défaut :

```
#include <dmbimage.h>

rc = DBiEnableDatabase(NULL);
```

DBiEnableTable

Image	Audio	Vidéo
X		

Active une table pour les données DB2Image. Cette API est appelée une fois pour chaque table. Elle crée des tables de métadonnées pour le stockage et la gestion des attributs des colonnes Image dans une table. Pour éviter tout risque de verrouillage, l'application doit valider des transactions avant d'appeler cette API. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiEnableTable(
      char *tableSpace,
      char *tableName
      );
```

Paramètres

tableSpace (entrée)

Nom de l'espace table (ensemble de conteneurs dans lesquels des tables de gestion sont stockées). La spécification de l'espace table se compose des trois parties suivantes : *datats*, *indexts*, *longts*, *datats* étant l'espace table dans lequel sont créées les tables de métadonnées ; *indexts* l'espace table dans lequel sont créés les index des tables de métadonnées et *longts* l'espace table dans lequel sont stockées les valeurs contenues dans les colonnes longues des tables de métadonnées (telles que celles contenant les types de données LONG VARCHAR et LOB). Si vous indiquez une valeur nulle pour l'une des parties de la spécification de l'espace table, l'espace table par défaut est utilisé pour cette partie.

Si vous indiquez une valeur nulle pour n'importe quelle partie de la spécification de l'espace table, l'espace table par défaut pour cette partie est utilisé.

Produit EEE uniquement : L'espace table spécifié doit appartenir au même groupe de noeuds que la table utilisateur.

tableName (entrée)
Nom de la table devant contenir une colonne Image.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La table est déjà activée.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_TABLE_DOESNOT_EXIST
La table n'existe pas.

MMDB_RC_TABLESPACE_NOT_SAME_NODEGROUP
L'espace table spécifié n'appartient pas au même groupe de noeuds que la table utilisateur. **(Produit EEE uniquement)**

Exemples

Activation du serveur de bases de données pour les données DB2Image dans l'espace table MYTS. Utilisation des valeurs par défaut pour les espaces table longs et d'indexation :

```
#include <dmbimage.h>

rc = DBiEnableTable("myts,,",
       "Employés");
```

Activation de la table Employés pour les images (données DB2Image). Utilisation des espaces table par défaut :

```
#include <dmbimage.h>

rc = DBiEnableTable(NULL,
       "Employés");
```

DBiGetError

Image	Audio	Vidéo
X		

Renvoie une description de la dernière erreur. Vous pouvez appeler cette API lorsqu'une autre API renvoie un code d'erreur.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiGetError(
      SQLINTEGER *sqlcode,
      char *errorMsgText
      );
```

Paramètres

sqlcode (sortie)
Code d'erreur SQL générique.

errorMsgText (sortie)
Texte du message d'erreur SQL.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

Exemples

Extraction de la dernière erreur avec insertion du code d'erreur SQL dans le descripteur `errCode` et du texte du message dans le descripteur `errMsg`.

```
#include <dmbimage.h>

rc = DBiGetError(&errCode, &errMsg);
```

DBiGetInaccessibleFiles

Image	Audio	Vidéo
X		

Renvoie les noms des fichiers inaccessibles référencés dans les colonnes Image des tables utilisateur. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SELECT sur les colonnes Image activées dans toutes les tables utilisateur sur lesquelles porte la recherche, ainsi que les tables de gestion associées.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiGetInaccessibleFiles(
      char *tableName,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié, non qualifié ou nul. Si un nom de table est indiqué, la recherche des fichiers inaccessibles porte sur cette table. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables ayant le qualificatif indiqué.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers inaccessibles référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste de tous les fichiers inaccessibles référencés dans les colonnes Image de la table Employés.

```
#include <dmbimage.h>
long idx;

rc = DBiGetInaccessibleFiles("Employés",
       &count, &filelist);
```

DBiGetReferencedFiles

Image	Audio	Vidéo
X		

Renvoie les noms des fichiers référencés dans les colonnes Image des tables utilisateur. Si un fichier est inaccessible (en raison, par exemple, de la non résolution de son nom à l'aide des spécifications de variable d'environnement), le nom de ce fichier est précédé du signe astérisque (*). Cette API n'utilise pas la zone FILENAME de la structure de données FILEREF. Par conséquent, elle attribue la valeur NULL à cette zone. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez annuler la structure de données de la liste de fichiers (filelist).

Classe d'autorisation

SELECT sur les colonnes Image activées dans toutes les tables utilisateur sur lesquelles porte la recherche, ainsi que les tables de gestion associées.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiGetReferencedFiles(
      char *tableName,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

tableName (entrée)

Nom de table qualifié, non qualifié ou nul. Si un nom de table est indiqué, la recherche des fichiers porte sur cette table. Si une valeur nulle est indiquée, la recherche porte sur toutes les tables appartenant à l'ID utilisateur en cours.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste de tous les fichiers référencés dans les colonnes Image de la table Employés :

```
#include <dmbimage.h>
long idx;

rc = DBiGetReferencedFiles("Employés",
       &count, &filelist);
```

DBiIsColumnEnabled

Image	Audio	Vidéo
X		

Détermine si une colonne a été activée pour les données DB2Image. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Droits SYSADM, DBADM, être propriétaire de la table ou disposer sur celle-ci du privilège SELECT.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiIsColumnEnabled(
      char *tableName,
      char *colName,
      short *status
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié ou non qualifié.

colName (entrée)
Nom d'une colonne.

status (sortie)
Indique si la colonne est activée. Ce paramètre renvoie une valeur numérique. L'extension renvoie également une constante qui indique l'état de la base. Les valeurs et les constantes sont les suivantes :

1 MMDB_IS_ENABLED

0 MMDB_IS_NOT_ENABLED

-1 MMDB_INVALID_DATATYPE

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La colonne est déjà activée.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Détermine si la colonne Photo de la table Employés est activée pour les données DB2Image.

```
#include <dmbimage.h>

rc = DBiIsColumnEnabled("Employés",
       "photo", &status);
```

DBiIsDatabaseEnabled

Image	Audio	Vidéo
X		

Détermine si une base de données a été activée pour les données DB2Image. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiIsDatabaseEnabled(
      short *status
      );
```

Paramètres

status (sortie)
Indique si la base de données est activée. Ce paramètre renvoie une valeur numérique. L'extension renvoie également une constante qui indique l'état de la base. Les valeurs et les constantes sont les suivantes :

1 MMDB_IS_ENABLED

0 MMDB_IS_NOT_ENABLED

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Déterminer si la base de données Employés est activée pour les données DB2Image :

```
#include <dmbimage.h>

rc = DBiIsDatabaseEnabled(&status);
```

DBiIsFileReferenced

Image	Audio	Vidéo
X		

Renvoie une liste des entrées de table de colonnes Image, qui font référence au fichier indiqué. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SELECT sur les colonnes Image activées dans toutes les tables utilisateur sur lesquelles porte la recherche, ainsi que les tables de gestion associées.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiIsFileReferenced(
      char *tableName,
      char *fileName,
      long *count,
      FILEREF *(*tableList)
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié, non qualifié ou nul. Si un nom de table est indiqué, la recherche du fichier indiqué porte sur cette table. Si une valeur nulle est indiquée, la recherche porte sur toutes les tables appartenant à l'ID utilisateur en cours.

fileName (entrée)
Nom du fichier référencé.

count (sortie)
Nombre d'entrées dans la liste de sortie.

tableList (sortie)
Liste des entrées de table faisant référence au fichier indiqué.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste des entrées dans les colonnes Image de la table Employés faisant référence au fichier /images/adupont.bmp :

```
#include <dmbimage.h>
long idx;

rc = DBiIsFileReferenced(NULL,
       "/images/adupont.bmp",
       &count, &tableList);
```

DBiIsTableEnabled

Image	Audio	Vidéo
X		

Détermine si une table a été activée pour les données DB2Image. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiIsTableEnabled(
      char *tableName,
      short *status
      );
```

Paramètres

tableName (entrée)
Nom d'une table.

status (sortie)
Indique si la table est activée. Ce paramètre renvoie une valeur numérique. L'extension renvoie également une constante qui indique l'état de la base. Les valeurs et les constantes sont les suivantes :

1 MMDB_IS_ENABLED

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Déterminer si la table Employés est activée pour les données DB2Image :

```
#include <dmbimage.h>

rc = DBiIsTableEnabled("Employés",
       &status);
```

DBiPrepareAttrs

Image	Audio	Vidéo
X		

Prépare les attributs des images fournis par l'utilisateur. Cette API est utilisée lors du stockage ou de la mise à jour d'un objet image muni d'attributs fournis par l'utilisateur. Le code UDF qui s'exécute sur le serveur attend toujours des données au format «big endian», utilisé sur la plupart des plateformes UNIX. Si un objet image est stocké ou mis à jour au format «little endian», c'est-à-dire à partir d'un client non UNIX, l'API DBiPrepare doit être appelée pour que la demande de stockage ou de mise à jour puisse aboutir.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
void DBiPrepareAttrs(
      MMDBImageAttrs *imgAttr
      );
```

Paramètres

imgAttr (in)
Attributs de l'image fournis par l'utilisateur.

Exemples

Préparation des attributs de l'image fournis par l'utilisateur :

```
#include <dmbimage.h>

DBiPrepareAttrs(&imgattr);
```

DBiReorgMetadata

Image	Audio	Vidéo
X		

«Nettoie» les tables de métadonnées associées à l'extension Image en exécutant, par exemple, les opérations suivantes :

- récupération de l'espace qui n'est plus utilisé dans les tables de métadonnées d'image ;
- suppression des références à des fichiers image qui n'existent plus dans les tables de métadonnées d'image.

Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Alter, Control, SYSADM, SYSCTRL, SYSMAINT, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbimage.lib	libdmbimage.a (AIX) libdmbimage.sl (HP-UX) libdmbimage.so (Solaris)

Fichier d'inclusion

dmbimage.h

Syntaxe

```
long DBiReorgMetadata(
      char *tableName
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié, non qualifié ou nul. Si un nom de table est spécifié, le nettoyage porte sur les tables de métadonnées d'image associées à la table utilisateur indiquée. Lorsqu'une valeur nulle est indiquée, le nettoyage porte sur les tables de métadonnées des colonnes Audio de toutes les tables associées à l'ID utilisateur en cours.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Nettoyage des tables de métadonnées pour les colonnes Image de la table Employés.

```
#include <dmbimage.h>

rc = DBiReorgMetadata("Employés");
```

DBvAdminGetInaccessibleFiles

Image	Audio	Vidéo
		X

Renvoie les noms des fichiers inaccessibles référencés dans les colonnes Vidéo des tables utilisateur. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvAdminGetInaccessibleFiles(
      char *qualifier,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

qualifier (entrée)
ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, la recherche porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables de la base de données en cours.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers inaccessibles référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste de tous les fichiers inaccessibles référencés dans les colonnes Vidéo des tables associées à l'ID utilisateur rdurand.

```
#include <dmbvideo.h>
long idx;

rc = DBvAdminGetInaccessibleFiles
       ("rdurand", &count,
       &filelist);
```

DBvAdminGetReferencedFiles

Image	Audio	Vidéo
		X

Renvoie les noms des fichiers référencés dans les colonnes Vidéo des tables utilisateur. Si un fichier est inaccessible (en raison, par exemple, de la non résolution de son nom à l'aide des spécifications de variable d'environnement), le nom de ce fichier est précédé du signe astérisque (*). Cette API n'utilise pas la zone FILENAME de la structure de données FILEREF. Par conséquent, elle attribue la valeur NULL à cette zone. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez annuler la structure de données de la liste de fichiers (filelist).

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvAdminGetReferencedFiles(
      char *qualifier,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

qualifier (entrée)

ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, la recherche porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables de la base de données en cours.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste de tous les fichiers référencés dans les colonnes Vidéo des tables appartenant à l'ID utilisateur adupont.

```
#include <dmbvideo.h>
long idx;

rc = DBvAdminGetReferencedFiles
       ("adupont", &count,
       &filelist);
```

DBvAdminIsFileReferenced

Image	Audio	Vidéo
		X

Renvoie la liste des entrées de la colonne Vidéo dans les tables utilisateur faisant référence au fichier indiqué. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvAdminIsFileReferenced(
      char *qualifier,
      char *fileName,
      long *count,
      FILEREF *(*tableList)
      );
```

Paramètres

qualifier (entrée)
ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, la recherche porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables de la base de données en cours.

fileName (entrée)
Nom du fichier référencé.

count (sortie)
Nombre d'entrées dans la liste de sortie.

tableList (sortie)
Liste des entrées de table faisant référence au fichier indiqué.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste des entrées dans les colonnes vidéo de toutes les tables de la base de données en cours faisant référence au fichier /videos/rdurand.mpg :

```
#include <dmbvideo.h>
long idx;

rc = DBvAdminIsFileReferenced(NULL,
       "/videos/rdurand.mpg",
       &count, &tableList);
```

DBvAdminReorgMetadata

Image	Audio	Vidéo
		X

«Nettoie» les tables de métadonnées associées à l'extension Vidéo en exécutant, par exemple, les opérations suivantes :

- récupération de l'espace qui n'est plus utilisé dans les tables de métadonnées vidéo :
- suppression des références à des fichiers vidéo qui n'existent plus dans les tables de métadonnées vidéo.

Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvAdminReorgMetadata(
      char *qualifier
      );
```

Paramètres

qualifier (entrée)
ID utilisateur correct ou valeur nulle. Si un ID utilisateur est spécifié, le nettoyage porte sur toutes les tables ayant le qualificatif indiqué. Si une valeur nulle est spécifiée, le nettoyage porte sur toutes les tables de la base de données en cours.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_NO_AUTH
L'utilisateur ne dispose pas des droits nécessaires pour appeler cette API.

Exemples

Nettoyage des tables de métadonnées pour les colonnes Vidéo dans les tables associées à l'ID utilisateur rdurand.

```
#include <dmbvideo.h>

rc = DBvAdminReorgMetadata("rdurand");
```

DBvBuildStoryboardFile

Image	Audio	Vidéo
		X

Crée des entrées dans un catalogue des prises de vue pour toutes les prises de vue d'une séquence vidéo. La vidéo source peut se trouver dans une base de données ou dans un fichier. Pour chaque prise de vue, l'API stocke son numéro, le numéro de la première image, celui de la dernière image ainsi que les informations relatives à une image représentative au moins. Les valeurs contenues dans la structure de données DBvStoryboardCtrl déterminent le mode d'identification de plusieurs images représentatives pour une prise de vue. Pour les prises de vue dont la longueur est inférieure à une valeur de seuil dans DBvStoryboardCtrl, l'API identifie une image représentative. Pour les prises de vue dont la longueur est comprise entre une valeur de seuil minimale et une valeur de seuil maximale dans DBvStoryboardCtrl, l'API identifie deux images représentatives. Pour les prises de vue dont la longueur est supérieure à la valeur de seuil maximale dans DBvStoryboardCtrl, l'API identifie trois images représentatives. Les informations relatives à l'image représentative contiennent le numéro de l'image et le nom du fichier où se trouve le contenu de l'image. Ces informations peuvent être utilisées pour afficher un storyboard, à savoir un résumé visuel d'une séquence vidéo.

Classe d'autorisation

Insert, Control

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvBuildStoryboardFile(
      char *fileName,
      DBvIOType *video,
      DBvShotControl *shotCtrl,
      DBvStoryBoardCtrl *sbCtrl
            );
```

Paramètres

catalogName (entrée)
Pointeur sur le nom du catalogue des prises de vue.

video (entrée)
Pointeur sur la structure vidéo.

shotCtrl (entrée)
Pointeur sur la structure de contrôle des prises de vue.

sbCtrl (entrée)
Pointeur sur la structure de contrôle du storyboard.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_INVALID_CATALOG
Le catalogue est incorrect ou n'existe pas.

Exemples

Création d'un catalogue des prises de vue appelé hotshots et stockage des données relatives à toutes les prises de vue d'une séquence vidéo :

```
#include <dmbshot.h>

rc = DBvBuildStoryboardFile("hotshots",
       video, &shotCtrl, &sbCtrl);
```

DBvBuildStoryboardTable

Image	Audio	Vidéo
		X

Crée des entrées dans un catalogue pour toutes les prises de vue d'une séquence vidéo. La vidéo source peut se trouver dans une base de données ou dans un fichier. Ce dernier se trouve dans une base de données. Pour chaque prise de vue, l'API stocke les informations relatives au descripteur ou au fichier pour la vidéo source. L'API stocke également le numéro de la prise de vue, le numéro de la première image, celui de la dernière image ainsi que les informations relatives à une image représentative au moins. Les valeurs contenues dans la structure de données DBvStoryboardCtrl déterminent le mode d'identification de plusieurs images représentatives pour une prise de vue. Pour les prises de vue dont la longueur est inférieure à une valeur de seuil dans DBvStoryboardCtrl, l'API identifie une image représentative. Pour les prises de vue dont la longueur est comprise entre une valeur de seuil minimale et une valeur de seuil maximale dans DBvStoryboardCtrl, l'API identifie deux images représentatives. Pour les prises de vue dont la longueur est supérieure à la valeur de seuil maximale dans DBvStoryboardCtrl, l'API identifie trois images représentatives. Les informations relatives à l'image représentative comprennent le numéro et les données se rapportant à l'image. Les informations relatives à l'image représentative stockées dans le catalogue des prises de vue peuvent être utilisées pour l'affichage d'un storyboard, à savoir un résumé visuel d'une séquence vidéo.

Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Insert, Control

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvBuildStoryboardTable(
      char *catalogName,
      DBvIOType *video,
```

```
DBvShotControl *shotCtrl,
DBvStoryBoardCtrl *sbCtrl,
SQLHDBC hdbc
);
```

Paramètres

catalogName (entrée)
Pointeur sur le nom du catalogue des prises de vue.

video (entrée)
Pointeur sur la structure vidéo.

shotCtrl (entrée)
Pointeur sur la structure de contrôle des prises de vue.

sbCtrl (entrée)
Pointeur sur la structure de contrôle du storyboard.

hdbc (entrée)
Descripteur de base de données obtenu par SQLConnect.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_INVALID_CATALOG
Le catalogue est incorrect ou n'existe pas.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Création d'entrées dans un catalogue des prises de vue appelé hotshots pour une séquence video :

```
#include <dmbshot.h>

rc = DBvBuildStoryboardTable("hotshots",
       video, &shotCtrl, &sbCtrl, hdbc);
```

DBvClose

Image	Audio	Vidéo
		X

Ferme un fichier vidéo ayant été ouvert pour la détection de changement de plan.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbmpeg.lib	libdmbmpeg.a (AIX) libdmbmpeg.sl (HP-UX) libdmbmpeg.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvClose(
      DB2vIOType *video
      );
```

Paramètres

video (entrée)
Pointeur sur la structure vidéo.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_CANNOT_CLOSE
Le fichier vidéo n'a pas pu être ouvert.

Exemples

Fermeture d'un fichier vidéo qui a été ouvert pour la détection de changement de plan :

```
#include <dmbshot.h>

rc=DBvClose (video);
```

DBvCreateIndex

Image	Audio	Vidéo
		X

Crée un index pour une séquence vidéo stockée dans un fichier. Cet index est utilisé par l'extension Vidéo pour accéder à des prises de vue et à des images dans une séquence vidéo. L'index est stocké dans un fichier ordinaire dans le même répertoire que le fichier vidéo source.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbmpeg.lib	libdmbmpeg.a (AIX) libdmbmpeg.sl (HP-UX) libdmbmpeg.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvCreateIndex(
      char *fileName
      );
```

Paramètres

fileName (entrée)
Pointeur sur le nom du fichier vidéo.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_OPEN_VIDEO
Le fichier vidéo n'a pas pu être ouvert pour le traitement.

MMDB_RC_INDEX_FAIL
L'index n'a pas pu être créé.

Exemples

Création d'un index pour la séquence vidéo dans un fichier \videos\adupont.mpg :

```
#include <dmbshot.h>

rc = DBvCreateIndex("\videos\ajones.mpg");
```

DBvCreateIndexFromVideo

Image	Audio	Vidéo
		X

Crée un index pour une séquence vidéo. La séquence vidéo doit d'abord être ouverte pour permettre la détection des prises de vue. Cet index est utilisé par l'extension Vidéo pour accéder à des prises de vue et à des images dans une séquence vidéo. L'index est stocké dans un fichier ordinaire. Le nom du fichier est stocké dans la structure de données DBvIOType.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvCreateIndexFromVideo(
      DBvIOType *video
      );
```

Paramètres

video (mise à jour)
Pointeur sur la structure d'une séquence vidéo.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_OPEN_VIDEO
Le fichier vidéo n'a pas pu être ouvert pour le traitement.

MMDB_RC_INDEX_FAIL
L'index n'a pas pu être créé.

Exemples

Création d'un index pour une séquence vidéo :

```
#include <dmbshot.h>

rc = DBvCreateIndexFromVideo(video);
```

DBvCreateShotCatalog

Image	Audio	Vidéo
		X

Crée un catalogue des prises de vue composé d'un ensemble de tables qui contiennent des informations sur les prises de vue (par exemple, les numéros d'image).

L'application doit être connectée à une base de données activée pour db2Video et db2Image.

Classe d'autorisation

Create, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvCreateShotCatalog(
      char *catalogName,
      SQLHDBC hdbc
      );
```

Paramètres

catalogName (entrée)
Nom du catalogue des prises de vue à créer.

hdbc (entrée)
Descripteur de base de données obtenu par SQLConnect.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Création d'un catalogue des prises de vue intitulé hotshots.

```
#include <dmbshot.h>

rc = DBvCreateShotCatalog("hotshots", hdbc);
```

DBvDeleteShot

Image	Audio	Vidéo
		X

Supprime une prise de vue d'un catalogue.

Classe d'autorisation

Insert, Control

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvDeleteShot(
      char *catalogName,
      char *shotHandle,
      SQLHDBC hdbc
      );
```

Paramètres

catalogName (entrée)
Nom du catalogue.

shotHandle (entrée)
Descripteur de prise de vue.

hdbc (entrée)
Descripteur de base de données obtenu par SQLConnect.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_ACCESS
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_INVALID_CATALOG
Le catalogue est incorrect ou n'existe pas.

Exemples

Suppression d'une prise de vue du catalogue hotshots via le descripteur de prise de vue.

```
#include <dmbshot.h>

rc = DBvDeleteShot("hotshots", shot,
       hdbc);
```

DBvDeleteShotCatalog

Image	Audio	Vidéo
		X

Supprime un catalogue des prises de vue.

Classe d'autorisation

Control, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvDeleteShotCatalog(
      char *catalogName,
      SQLHDBC hdbc
      );
```

Paramètres

catalogName (entrée)
Nom du catalogue des prises de vue à supprimer.

hdbc (entrée)
Descripteur de base de données obtenu par SQLConnect.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_ACCESS
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_INVALID_CATALOG
Le catalogue est incorrect ou n'existe pas.

Suppression du catalogue des prises de vue intitulé hotshots.

Exemples

```
#include <dmbshot.h>

rc = DBvDeleteShotCatalog("hotshots",
       hdbc);
```

DBvDetectShot

Image	Audio	Vidéo
		X

Recherche la prise de vue suivante dans un fichier de séquence vidéo. En cas de détection d'une prise de vue, enregistre le numéro et les données de la première image de la prise de vue détectée. Vous devez examiner le descripteur shotDetected pour déterminer si une prise de vue a été détectée.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvDetectShot(
      DBvIOType *video,
      unsigned long *start_frame,
      char *shotDetected,
      DBvShotControl *shotCtrl,
      DBvShotType *shot,
            );
```

Paramètres

video (mise à jour)
Pointeur sur la structure vidéo.

start_frame (entrée/sortie)
Numéro d'image utilisé comme point de départ pour la recherche. En retour, le paramètre est mis à jour avec la position pour commencer la recherche de la prise de vue suivante.

shotDetected (sortie)
Descripteur de détection d'une prise de vue : 1 = image détectée, 0 = aucune image détectée.

shotCtrl (entrée)
Pointeur sur les données de contrôle de prise de vue.

shot (sortie)
Pointeur sur la prise de vue détectée et les données de prise de vue.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_EOF
Fin du fichier atteinte.

MMDB_NO_INDEX
L'index vidéo n'existe pas.

Exemples

Recherche la prise de vue suivante dans un fichier de séquences vidéo en partant de l'image 1 :

```
#include <dmbshot.h>

long start_frame=1;

rc = DBvDetectShot(video, start_frame&Detected,
       &shotCtrl, &shot);
```

DBvDisableColumn

Image	Audio	Vidéo
		X

Désactive une colonne de sorte que les données DB2Video ne soient plus prises en charge. Les entrées de la colonne sont définies par NULL et les métadonnées associées à cette colonne sont supprimées. Tous les déclencheurs définis par l'extension Vidéo pour cette colonne sont également supprimés. De nouvelles lignes peuvent être insérées dans la table contenant la colonne désactivée. Ces nouvelles lignes peuvent contenir des données de type DB2Video mais aucune métadonnée (dans les tables de gestion) ne leur est associée. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvDisableColumn(
      char *tableName,
      char *colName,
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant la colonne Vidéo.

colName (entrée)
Nom de la colonne Vidéo.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Désactivation de la colonne tv_ads dans la table Employés pour les données DB2Video :

```
#include <dmbvideo.h>
rc = DBvDisableColumn("Employés",
       "tv_ads");
```

DBvDisableDatabase

Image	Audio	Vidéo
		X

Désactive une base de données de sorte que les données DB2Video ne soient plus prises en charge. Toutes les tables contenues dans la base de données définie pour DB2Video sont également désactivées. Les métadonnées et les fonctions UDF définies par l'extension Vidéo pour la base de données sont supprimées. De nouvelles lignes peuvent être insérées dans des tables de type DB2Video dans la base de données mais aucune métadonnée (dans les tables de gestion) ne leur est associée.

Classe d'autorisation

DBADM, SYSADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvDisableDatabase(
     );
```

Paramètres

L'API DBvDisableDatabase ne comporte aucun paramètre.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Désactivation de la base de données en cours pour les données DB2Video :

```
#include <dmbvideo.h>

rc = DBvDisableDatabase();
```

DBvDisableTable

Image	Audio	Vidéo
		X

Désactive une table de sorte que les données DB2Video ne soient plus prises en charge. Toutes les colonnes de la table définies pour DB2Video sont également désactivées. Certaines métadonnées définies par l'extension Vidéo pour la table sont supprimées. De nouvelles lignes peuvent être insérées dans des tables de type DB2Video mais aucune métadonnée (dans les tables de gestion) ne leur est associée. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvDisableTable(
      char *tableName
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant une colonne Vidéo.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Désactivation de la table Employés pour les données DB2Video :

```
#include <dmbvideo.h>

rc = DBvDisableTable("Employés");
```

DBvEnableColumn

Image	Audio	Vidéo
		X

Active une colonne pour les données DB2Video. L'API définit et gère les relations entre cette colonne et les tables de métadonnées. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données et la table utilisateur doit être validée.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Le privilège USE (utiliser) est également obligatoire pour les espaces table et les pools de mémoire tampon indiqués dans les paramètres de l'API.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvEnableColumn(
      char *tableName,
      char *colName,
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant la colonne Vidéo.

colName (entrée)
Nom de la colonne Vidéo.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La colonne est déjà activée.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_WRONG_SIGNATURE
Le type de données de la colonne indiquée est incorrect. Le système attend le type de données défini par l'utilisateur (MMDBSYS.DB2VIDEO).

MMDB_RC_COLUMN_DOESNOT_EXIST
La colonne n'est pas définie dans la table indiquée.

MMDB_RC_NOT_ENABLED
La base de données ou la table n'est pas activée.

Exemples

Activation de la colonne Vidéo dans la table Employés pour les données DB2Video :

```
#include <dmbvideo.h>

rc = DBvEnableColumn("Employés",
       "vidéo");
```

DBvEnableDatabase

Image	Audio	Vidéo
		X

Active une base de données pour les données dDB2Video. Cette API est appelée une fois pour chaque base de données. Elle définit un type DB2 défini par l'utilisateur, DB2Video, dans le gestionnaire de bases de données. Elle crée également toutes les fonctions utilisateur manipulant des données DB2Video.

Classe d'autorisation

DBADM, SYSADM, SYSCTRL

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvEnableDatabase(
      char *tableSpace
      );
```

Paramètres

tableSpace (entrée)

Nom de l'espace table (ensemble de conteneurs dans lesquels des tables de gestion sont stockées). La spécification de l'espace table se compose des trois parties suivantes : *datats, indexts, longts*, *datats* étant l'espace table dans lequel sont créées les tables de métadonnées ; *indexts* l'espace table dans lequel sont créés les index des tables de métadonnées et *longts* l'espace table dans lequel sont stockées les valeurs contenues dans les colonnes longues des tables de métadonnées (telles que celles contenant les types de données LONG VARCHAR et LOB). Si vous indiquez une valeur nulle pour n'importe quelle partie de la spécification de l'espace table, l'espace table par défaut pour cette partie est utilisé.

Produit EEE uniquement : Les espaces table spécifiés lors de l'activation d'une base de données pour une extension doivent être définis sur un groupe de noeuds comprenant tous les noeuds dans le système de bases de données partitionnées.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La base de données est déjà active.

MMDB_RC_API_NOT_SUPPORTED_FOR_SERVER
Le serveur connecté ne prend pas en charge cette commande.

MMDB_WARN_NOT_ALL_NODES
L'espace table spécifié ne comprend pas tous les noeuds pour l'extension. **(Produit EEE uniquement)**

MMDB_RC_NOT_SAME_NODEGROUP
Les espaces table spécifiés ne font pas partie du même groupe de noeuds. **(Produit EEE uniquement)**

Exemples

Activation de la base de données en cours pour les données DB2Video dans l'espace table MYTS. Utilisation des valeurs par défaut pour les espaces table longs et d'indexation :

```
#include <dmbvideo.h>

rc = DBvEnableDatabase("myts,,");
```

Activation de la base de données en cours pour les données DB2Video. Utilisation des espaces table par défaut :

```
#include <dmbvideo.h>

rc = DBvEnableDatabase(NULL);
```

DBvEnableTable

Image	Audio	Vidéo
		X

Active une table pour les données DB2Video. Cette API est appelée une fois pour chaque table. Elle crée des tables de métadonnées pour le stockage et la gestion des attributs des colonnes Vidéo dans une table. Pour éviter tout risque de verrouillage, l'application doit valider des transactions avant d'appeler cette API. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Control, Alter, SYSADM, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvEnableTable(
      char *tableSpace,
      char *tableName
      );
```

Paramètres

tableSpace (entrée)

Nom de l'espace table (ensemble de conteneurs dans lesquels des tables de gestion sont stockées). La spécification de l'espace table se compose des trois parties suivantes : *datats, indexts, longts*, *datats* étant l'espace table dans lequel sont créées les tables de métadonnées ; *indexts* l'espace table dans lequel sont créés les index des tables de métadonnées et *longts* l'espace table dans lequel sont stockées les valeurs contenues dans les colonnes longues des tables de métadonnées (telles que celles contenant les types de données LONG VARCHAR et LOB). Si vous indiquez une valeur nulle pour l'une des parties de la spécification de l'espace table, l'espace table par défaut est utilisé pour cette partie.

Si vous indiquez une valeur nulle pour n'importe quelle partie de la spécification de l'espace table, l'espace table par défaut pour cette partie est utilisé.

Produit EEE uniquement : L'espace table spécifié doit appartenir au même groupe de noeuds que la table utilisateur.

tableName (entrée)
Nom de la table devant contenir une colonne Vidéo.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_WARN_ALREADY_ENABLED
La table est déjà activée.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_TABLE_DOESNOT_EXIST
La table n'existe pas.

MMDB_RC_TABLESPACE_NOT_SAME_NODEGROUP
L'espace table spécifié n'appartient pas au même groupe de noeuds que la table utilisateur. **(Produit EEE uniquement)**

Exemples

Activation de la table Employés pour les données DB2Video dans l'espace table MYTS. Utilisation des valeurs par défaut pour les espaces table longs et d'indexation :

```
#include <dmbvideo.h>

rc = DBvEnableTable("myts,,",
       "Employés");
```

Activation de la table Employés pour les données DB2Video. Utilisation des espaces table par défaut :

```
#include <dmbvideo.h>

rc = DBvEnableTable(NULL,
       "Employés");
```

DBvFrameDataTo24BitRGB

Image	Audio	Vidéo
		X

Convertit une image vidéo du format couleur YUV (par exemple, MPEG) au format RGB 24 bits. L'utilisateur doit affecter une mémoire tampon cible avant d'émettre l'appel d'API.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbmpeg.lib	libdmbmpeg.a (AIX) libdmbmpeg.sl (HP-UX) libdmbmpeg.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvFrameDataTo24BitRGB(
      unsigned char *RGB,
      DBvFrameData *fd,
      unsigned long dx,
      unsigned long dy
      );
```

Paramètres

RGB (sortie)
Pointeur sur la mémoire tampon RGB cible.

fd (entrée)
Pointeur sur les données d'image à convertir.

dx (entrée)
Largeur de l'image.

dy (entrée)
Hauteur de l'image.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

Exemples

Conversion d'une image vidéo du format MPEG au format RGB 24 bits.

```
#include <dmbshot.h>

rc = DBvFrameDataTo24BitRGB(RGB, &video->fd,
       video->dx, video->dy);
```

DBvGetError

Image	Audio	Vidéo
		X

Renvoie une description de la dernière erreur. Vous pouvez appeler cette API lorsqu'une autre API renvoie un code d'erreur.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvGetError(
      SQLINTEGER *sqlcode,
      char *errorMsgText
      );
```

Paramètres

sqlcode (sortie)
Code d'erreur SQL générique.

errorMsgText (sortie)
Texte du message d'erreur SQL.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

Exemples

Extraction de la dernière erreur avec insertion du code d'erreur SQL dans le descripteur `errCode` et du texte du message dans le descripteur `errMsg`.

```
#include <dmbvideo.h>

rc = DBvGetError(&errCode, &errMsg);
```

DBvGetFrame

Image	Audio	Vidéo
		X

Extrait l'image en cours dans un fichier de séquences vidéo. Les données d'image sont renvoyées dans la structure vidéo DBvFrameData.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbmpeg.lib	libdmbmpeg.a (AIX) libdmbmpeg.sl (HP-UX) libdmbmpeg.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvGetFrame(
      DBvIOType *video
      );
```

Paramètres

video (mise à jour)
Pointeur sur la structure vidéo.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_EOF
Fin du fichier atteinte.

Exemples

Extraction de l'image en cours dans un fichier de séquences vidéo :

```
#include <dmbshot.h>

rc = DBvGetFrame(video);
```

DBvGetInaccessibleFiles

Image	Audio	Vidéo
		X

Renvoie les noms des fichiers inaccessibles référencés dans les colonnes Vidéo des tables utilisateur. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

Le privilège SELECT sur les colonnes vidéo activées dans toutes les tables utilisateur sur lesquelles porte la recherche, ainsi que les tables de gestion associées.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvGetInaccessibleFiles(
      char *tableName,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié, non qualifié ou nul. Si un nom de table est indiqué, la recherche des fichiers inaccessibles porte sur cette table. Si une valeur nulle est spécifiée, la recherche porte sur toutes les tables ayant le qualificatif indiqué.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers inaccessibles référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste des fichiers inaccessibles référencés dans les colonnes Vidéo de la table Employés.

```
#include <dmbvideo.h>
long idx;

rc = DBvGetInaccessibleFiles("Employés",
       &count, &filelist);
```

DBvGetReferencedFiles

Image	Audio	Vidéo
		X

Renvoie les noms des fichiers référencés dans les colonnes Vidéo des tables utilisateur. Si un fichier est inaccessible (en raison, par exemple, de la non résolution de son nom à l'aide des spécifications de variable d'environnement), le nom de ce fichier est précédé du signe astérisque (*). Cette API n'utilise pas la zone FILENAME de la structure de données FILEREF. Par conséquent, elle attribue la valeur NULL à cette zone. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez annuler la structure de données de la liste de fichiers (filelist).

Classe d'autorisation

Le privilège SELECT sur les colonnes vidéo activées dans toutes les tables utilisateur sur lesquelles porte la recherche, ainsi que les tables de gestion associées.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvGetReferencedFiles(
      char *tableName,
      long *count,
      FILEREF *(*fileList)
      );
```

Paramètres

tableName (entrée)

Nom de table qualifié, non qualifié ou nul. Si un nom de table est indiqué, la recherche des fichiers porte sur cette table. Si une valeur nulle est indiquée, la recherche porte sur toutes les tables appartenant à l'ID utilisateur en cours.

count (sortie)
Nombre d'entrées dans la liste de sortie.

fileList (sortie)
Liste des fichiers référencés dans la table.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste de tous les fichiers référencés dans les colonnes Vidéo de la table Employés.

```
#include <dmbvideo.h>
long idx;

rc = DBvGetReferencedFiles("Employés",
       &count, &filelist);
```

DBvInitShotControl

Image	Audio	Vidéo
		X

Initialise les valeurs dans la structure des données de contrôle des prises de vue.

Classe d'autorisation

Insert, Control

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvInitShotControl(
      DBvShotControl *shotCtrl,
      );
```

Paramètres

shotCtrl (entrée)
Pointeur sur la structure des données de contrôle de prise de vue.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_ACCESS
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Initialisation des valeurs dans la structure des données de contrôle de prise de vue.

```
#include <dmbshot.h>

rc = DBvInitShotControl(shotCtrl);
```

DBvInitStoryboardCtrl

Image	Audio	Vidéo
		X

Initialise les valeurs dans la structure des données de contrôle du storyboard.

Classe d'autorisation

Insert, Control

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvInitStoryboardCtrl(
      DBvStoryboardCtrl *sbCtrl,
      );
```

Paramètres

shotCtrl (entrée)
Pointeur sur la structure des données de contrôle de prise de vue.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_ACCESS
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Initialisation des valeurs dans la structure des données de contrôle du storyboard :

```
#include <dmbshot.h>

rc = DBvInitStoryboardCtrl(shotCtrl);
```

DBvInsertShot

Image	Audio	Vidéo
		X

Insère une prise de vue dans un catalogue de prises de vue.

Classe d'autorisation

Insert, Control

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvInsertShot(
      char *catalogName,
      DBvShotType *shot,
      DBvIOType *video,
      char *shotHandle,
      SQLHDBC hdbc
      );
```

Paramètres

catalogName (entrée)
Nom du catalogue.

shot (entrée)
Pointeur sur la prise de vue étendue à insérer dans le catalogue.

shotHandle (entrée)
Descripteur de prise de vue.

hdbc (entrée)
Descripteur de base de données obtenu par SQLConnect.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_ACCESS
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_INVALID_CATALOG
Le catalogue est incorrect ou n'existe pas.

Exemples

Insertion d'une prise de vue dans un catalogue de prises de vue :

```
rc = DBvInsertShot(
"hotshots",      /* nom du catalogue des prises de vue */
shot,            /* pointeur sur la structure des prises de vue */
video,           /* pointeur sur la structure vidéo */
shotHandle,      /* pointeur sur le descripteur des prises de vue */
hdbc);           /* descripteur de la connexion à la base de données */
```

DBvIsColumnEnabled

Image	Audio	Vidéo
		X

Détermine si une colonne a été activée pour les données DB2Video. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Droits SYSADM, DBADM, être propriétaire de la table ou disposer sur celle-ci du privilège SELECT.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvIsColumnEnabled(
      char *tableName,
      char *colName,
      short *status
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié ou non qualifié.

colName (entrée)
Nom d'une colonne.

status (sortie)
Indique si la colonne est activée. Ce paramètre renvoie une valeur numérique. L'extension renvoie également une constante qui indique l'état de la base. Les valeurs et les constantes sont les suivantes :

1 MMDB_IS_ENABLED

0 MMDB_IS_NOT_ENABLED

-1 MMDB_INVALID_DATATYPE

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Déterminer si la colonne Vidéo de la table Employés a été activée pour DB2Video :

```
#include <dmbvideo.h>

rc = DBvIsColumnEnabled("Employés",
       "vidéo", &status);
```

DBvIsDatabaseEnabled

Image	Audio	Vidéo
		X

Détermine si une base de données a été activée pour les données DB2Video. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvIsDatabaseEnabled(
      short *status
      );
```

Paramètres

status (sortie)
Indique si la base de données est activée. Ce paramètre renvoie une valeur numérique. L'extension renvoie également une constante qui indique l'état de la base. Les valeurs et les constantes sont les suivantes :

1 MMDB_IS_ENABLED

0 MMDB_IS_NOT_ENABLED

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Déterminer si la base de données Employés est activée pour les données DB2Video :

```
#include <dmbvideo.h>

rc = DBvIsDatabaseEnabled(&status);
```

DBvIsFileReferenced

Image	Audio	Vidéo
		X

Renvoie la liste des entrées de table dans la colonne Vidéo, qui font référence au fichier indiqué. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Il est nécessaire de libérer les ressources affectées par l'API après son appel. Plus précisément, vous devez supprimer le contenu de la zone Nom de fichier ainsi que la structure de données correspondante dans chaque entrée de la liste de fichiers (filelist).

Classe d'autorisation

Le privilège SELECT sur les colonnes vidéo activées dans toutes les tables utilisateur sur lesquelles porte la recherche, ainsi que les tables de gestion associées.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvIsFileReferenced(
      char *tableName,
      char *fileName,
      long *count,
      FILEREF *(*tableList)
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié, non qualifié ou nul. Si un nom de table est indiqué, la recherche du fichier indiqué porte sur cette table. Si une valeur nulle est indiquée, la recherche porte sur toutes les tables appartenant à l'ID utilisateur en cours.

fileName (entrée)
Nom du fichier référencé.

count (sortie)
Nombre d'entrées dans la liste de sortie.

tableList (sortie)
Liste des entrées de table faisant référence au fichier indiqué.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_MALLOC
Le système ne peut pas affecter de mémoire pour renvoyer les résultats.

Exemples

Liste des entrées dans les colonnes vidéo de la table Employés faisant référence au fichier /videos/adupont.mpg :

```
#include <dmbvideo.h>
long idx;

rc = DBvIsFileReferenced(NULL,
       "/videos/adupont.mpg",
       &count, &tableList);
```

DBvIsIndex

Image	Audio	Vidéo
		X

Vérifie si l'index vidéo existe. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbmpeg.lib	libdmbmpeg.a (AIX) libdmbmpeg.sl (HP-UX) libdmbmpeg.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvIsIndex(
      char *fileName,
      short *status
      );
```

Paramètres

fileName (entrée)
Nom du fichier référencé.

status (sortie)
Indique si l'index existe. La valeur 1 signifie que l'index existe et la valeur 0 signifie qu'il n'existe pas.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_ERROR
L'état n'est pas correct.

Exemples

Vérification de l'existence d'un index se rapportant au fichier de séquences vidéo \videos\adupont.mpg :

```
#include <dmbshot.h>

rc = DBvIsIndex("\videos\adupont.mpg", &status);
```

DBvIsTableEnabled

Image	Audio	Vidéo
		X

Détermine si une table a été activée pour les données DB2Video. Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvIsTableEnabled(
      char *tableName,
      short *status
      );
```

Paramètres

tableName (entrée)
Nom d'une table.

status (sortie)
Indique si la table est activée. Ce paramètre renvoie une valeur numérique. L'extension renvoie également une constante qui indique l'état de la base. Les valeurs et les constantes sont les suivantes :

1 MMDB_IS_ENABLED

0 MMDB_IS_NOT_ENABLED

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Déterminer si la table Employés est activée pour les données DB2Video :

```
#include <dmbvideo.h>

rc = DBvIsTableEnabled("Employés",
       &status);
```

DBvMergeShots

Image	Audio	Vidéo
		X

Fusionne deux prises de vue en une seule. La prise de vue qui en résulte utilise le descripteur de prise de vue et la première image de la première prise de vue. La dernière image de la plus longue des deux séquences est utilisée dans la séquence résultante. La ligne sur laquelle pointe la seconde prise de vue est supprimée.

Classe d'autorisation

Control, Select, Delete, Update

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvMergeShots(
      char *catalogName,
      char *shotHandle1,
      char *shotHandle2,
      SQLHDBC hdbc
      );
```

Paramètres

catalogName (entrée)
Nom du catalogue des prises de vue.

shotHandle1 (entrée)
Descripteur de la première prise de vue.

shotHandle2 (entrée)
Descripteur de la seconde prise de vue.

hdbc (entrée)
Descripteur de base de données obtenu par SQLConnect.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_CANNOT_MERGE
Fusion des prises de vue impossible.

MMDB_RC_INVALID_CATALOG
Le catalogue est incorrect ou n'existe pas.

Exemples

Fusion des prises de vue comportant les descripteurs shotHandle1 et shotHandle2 dans le catalogue hotshots :

```
#include <dmbshot.h>

rc = DBvMergeShots("hotshots", shotHandle1,
       shotHandle2, hdbc);
```

DBvOpenFile

Image	Audio	Vidéo
		X

Affecte de l'espace à une structure DBvIOType et ouvre le fichier vidéo de sorte qu'il puisse être accédé au niveau du pixel. Une fois le fichier vidéo ouvert, l'API pointe sur le premier numéro d'image (image 0).

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbmpeg.lib	libdmbmpeg.a (AIX) libdmbmpeg.sl (HP-UX) libdmbmpeg.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvOpenFile(
      DBvIOType **video,
      char *fileName,
      );
```

Paramètres

video (sortie)
Pointeur sur le pointeur de la structure vidéo.

fileName (entrée)
Nom du fichier vidéo à ouvrir.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_CANNOT_OPEN
Ouverture du fichier vidéo impossible.

MMDB_RC_NO_MEMORY
Mémoire insuffisante.

MMDB_RC_NO_INDEX
Index RAM vidéo inexistant.

Exemples

Ouverture du fichier de séquences vidéo \videos\adupont.mpg :

```
#include <dmbshot.h>

rc = DBvOpenFile(&videoa,
       "\videos\adupont.mpg");
```

DBvOpenHandle

Image	Audio	Vidéo
		X

Affecte de l'espace à une structure DBvIOType et ouvre le descripteur vidéo de sorte qu'il puisse accéder au niveau du pixel. La structure pointe sur le premier numéro d'image (image 0). La séquence vidéo peut être un objet BLOB. Elle est copiée dans le fichier temporaire dans un répertoire défini par la variable d'environnement DB2VIDEOTEMP. Le descripteur isIdx est défini en fonction de l'existence d'un index RAM.

Classe d'autorisation

SELECT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvOpenHandle(
      DBvIOType **video,
      DB2Video *videoHandle
      SQLHDBC hdbc
      );
```

Paramètres

video (sortie)
Pointeur sur la structure vidéo.

videoHandle (entrée)
Descripteur vidéo.

hdbc (entrée)
Descripteur de base de données obtenu par SQLConnect.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_CANNOT_OPEN
Ouverture du fichier vidéo impossible.

MMDB_RC_NO_MEMORY
Mémoire insuffisante.

MMDB_RC_NO_INDEX
Index RAM vidéo inexistant.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_INVALID_HANDLE
Le descripteur vidéo n'est pas correct.

Exemples

Ouverture du descripteur vidéo via le pointeur videoa :

```
#include <dmbshot.h>

rc = DBvOpenHandle(&oa, videoHandle, hdbc);
```

DBvPlay

Image	Audio	Vidéo
		X

Ouvre le lecteur vidéo sur le poste client et lit une séquence vidéo. Celle-ci peut être stockée dans une colonne Vidéo ou dans un fichier externe.

- Si la séquence vidéo est stockée dans un fichier externe, vous pouvez communiquer le nom de ce fichier ou le descripteur vidéo à l'API. L'API utilise la variable d'environnement DB2VIDEOPATH pour chercher l'emplacement du fichier. Le fichier doit être accessible à partir du poste client.
- Si la séquence vidéo est stockée dans une colonne, vous devez communiquer le descripteur vidéo à l'API. L'application doit être connectée à la base de données et avoir accès en lecture à la table dans laquelle les données vidéo sont stockées.

Si les données vidéo sont stockées dans une colonne, l'extension crée un fichier temporaire et copie dans celui-ci le contenu de l'objet à partir de la colonne. L'extension peut également créer un fichier temporaire si les données vidéo sont stockées dans un fichier externe, si le nom relatif de celui-ci ne peut pas être résolu à l'aide des variables d'environnement ou si ce fichier n'est pas accessible sur le poste client. Ce fichier est créé dans le répertoire désigné par la variable d'environnement DB2VIDEOTEMP. L'extension lit ensuite la séquence vidéo à partir du fichier temporaire.

Classe d'autorisation

SELECT sur la table utilisateur si une séquence vidéo est lue à partir d'une colonne.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

Lecture d'un objet vidéo stocké dans une colonne

```
long DBvPlay(
      char *playerName,
      MMDB_PLAY_HANDLE,
      DB2Video *videoHandle,
      waitFlag
      );
```

Syntaxe

Lecture d'un objet vidéo stocké dans un fichier

```
long DBvPlay(
      char *playerName,
      MMDB_PLAY_FILE,
      char *fileName,
      waitFlag
      );
```

Paramètres

playerName (entrée)
Nom du lecteur vidéo. Si ce paramètre a une valeur NULL, le lecteur vidéo par défaut défini par la variable d'environnement DB2VIDEOPLAYER est utilisé.

MMDB_PLAY_HANDLE (entrée)
Constante qui indique que la séquence vidéo est stockée dans une colonne.

MMDB_PLAY_FILE (entrée)
Constante qui indique que la séquence vidéo est stockée sous forme de fichier accessible à partir du poste client.

videoHandle (entrée)
Descripteur de la séquence vidéo. Ce paramètre doit être communiqué lorsque vous faites défiler une séquence vidéo dans une colonne. Si le descripteur vidéo représente un fichier externe, la variable d'environnement client DB2VIDEOPATH est utilisée pour déterminer l'emplacement du fichier.

fileName (entrée)
Nom du fichier contenant la séquence vidéo. L'API utilise la variable d'environnement DB2VIDEOPATH pour chercher l'emplacement du fichier. Le fichier doit être accessible à partir du poste client.

waitFlag (entrée)
Constante qui indique si le programme attend que l'utilisateur arrête le lecteur avant de continuer. Le paramètre MMDB_PLAY_WAIT entraîne la mise en oeuvre du lecteur dans la même unité d'exécution que l'application. Le paramètre MMDB_PLAY_NO_WAIT entraîne la mise en oeuvre du lecteur dans une unité d'exécution distincte.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Lecture de la séquence vidéo identifiée par le descripteur vidéo. Mise en oeuvre du lecteur par défaut dans la même unité d'exécution que l'application.

```
#include <dmbvideo.h>

rc = DBvPlay(NULL, MMDB_PLAY_HANDLE,
       MMDB_PLAY_WAIT);
```

DBvPrepareAttrs

Image	Audio	Vidéo
		X

Prépare les attributs des séquences vidéo fournis par l'utilisateur. Cette API est utilisée lors du stockage ou de la mise à jour d'un objet vidéo muni d'attributs fournis par l'utilisateur. Le code UDF qui s'exécute sur le serveur attend toujours des données au format «big endian», utilisé sur la plupart des plateformes UNIX. Si un objet vidéo est stocké ou mis à jour au format «little endian», c'est-à-dire à partir d'un client non UNIX, l'API DBvPrepare doit être appelée pour que la demande de stockage ou de mise à jour puisse aboutir.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
void DBvPrepareAttrs(
      MMDBVideoAttrs *vidAttr
      );
```

Paramètres

vidAttr (entrée)
Attributs de la séquence vidéo fournis par l'utilisateur.

Exemples

Préparation d'attributs de séquence vidéo fournis par l'utilisateur :

```
#include <dmbvideo.h>

DBvPrepareAttrs(&vidattr);
```

DBvReorgMetadata

Image	Audio	Vidéo
		X

«Nettoie» les tables de métadonnées associées à l'extension Vidéo en exécutant, par exemple, les opérations suivantes :

- récupération de l'espace qui n'est plus utilisé dans les tables de métadonnées vidéo :
- suppression des références à des fichiers vidéo qui n'existent plus dans les tables de métadonnées vidéo.

Pour que cette API puisse être appelée, l'application doit être connectée à une base de données.

Classe d'autorisation

Alter, Control, SYSADM, SYSCTRL, SYSMAINT, DBADM

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbvideo.lib	libdmbvideo.a (AIX) libdmbvideo.sl (HP-UX) libdmbvideo.so (Solaris)

Fichier d'inclusion

dmbvideo.h

Syntaxe

```
long DBvReorgMetadata(
      char *tableName,
      );
```

Paramètres

tableName (entrée)
Nom de table qualifié, non qualifié ou nul. Si un nom de table est spécifié, le nettoyage porte sur les tables de métadonnées vidéo associées à la table utilisateur indiquée. Si une valeur nulle est indiquée, le nettoyage porte sur les tables de métadonnées des colonnes Vidéo de toutes les tables associées à l'ID utilisateur en cours.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NO_AUTH
Le demandeur ne dispose pas des droits d'accès corrects.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

Exemples

Nettoyage des tables de métadonnées pour les colonnes Vidéo de la table Employés.

```
#include <dmbvideo.h>

rc = DBvReorgMetadata("Employés");
```

DBvSetFrameNumber

Image	Audio	Vidéo
		X

Définit l'image courante à l'aide d'un numéro d'image spécifique.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbmpeg.lib	libdmbmpeg.a (AIX) libdmbmpeg.sl (HP-UX) libdmbmpeg.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvSetFrameNumber(
      DBvIOType *video
      unsigned long frameNumber
      );
```

Paramètres

video (entrée)
Pointeur sur la structure vidéo.

frameNumber (entrée)
Numéro de l'image demandée.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_FRAME_NOT_FOUND
L'image demandée est introuvable.

MMDB_NO_INDEX
L'index vidéo n'existe pas.

Exemples

Insertion de l'image portant le numéro 85 dans un fichier vidéo :

```
#include <dmbshot.h>

rc = DBvSetFrameNumber(video, 85);
```

DBvSetShotComment

Image	Audio	Vidéo
		X

Mise à jour du commentaire en lecture seulement dans la prise de vue.

Classe d'autorisation

Control, Update

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvSetShotComment(
      char *catalogName,
      char *shotHandle,
      char *comment,
      SQLHDBC hdbc
      );
```

Paramètres

catalogName (entrée)
Nom du catalogue.

shotHandle (entrée)
Descripteur de la prise de vue à mettre à jour.

comment (entrée)
Nouveau commentaire pour la prise de vue.

hdbc (entrée)
Descripteur de base de données obtenu par SQLConnect.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_CANNOT_UPDATE
L'API ne peut pas mettre à jour la prise de vue.

MMDB_RC_INVALID_CATALOG
Le catalogue est incorrect ou n'existe pas.

Exemples

Remplacement de la remarque décrivant la prise de vue par le descripteur de prise de vue dans le catalogue hotshots.

```
#include <dmbshot.h>

rc = DBvSetShotComment("hotshot",
shotHandle, "C'est une séquence", hdbc);
```

DBvUpdateShot

Image	Audio	Vidéo
		X

Remplace les attributs d'une prise de vue vidéo dans le catalogue. Tous les attributs, à l'exception du commentaire, sont remplacés par les attributs de la structure DBvShotType. Si le pointeur sur la remarque a une valeur NULL, la remarque existante n'est pas modifiée.

Classe d'autorisation

Control, Update

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbshot.lib	libdmbshot.a (AIX) libdmbshot.sl (HP-UX) libdmbshot.so (Solaris)

Fichier d'inclusion

dmbshot.h

Syntaxe

```
long DBvUpdateShot(
      char *catalogName,
      DBvShotType *shot,
      char *shotHandle,
      SQLHDBC hdbc
      );
```

Paramètres

catalogName (entrée)
Nom du catalogue.

shot (entrée)
Pointeur sur la structure d'informations de la prise de vue contenant les attributs de celle-ci.

shotHandle (entrée)
Descripteur de prise de vue.

hdbc (entrée)
Descripteur de base de données obtenu par SQLConnect.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RC_NOT_CONNECTED
L'application n'a pas établi une connexion correcte à une base de données.

MMDB_RC_CANNOT_UPDATE
L'API ne peut pas mettre à jour la prise de vue.

MMDB_RC_NO_SHOT
La prise de vue n'existe pas.

MMDB_RC_INVALID_CATALOG
Le catalogue est incorrect ou n'existe pas.

Exemples

Mise à jour des attributs d'une prise de vue dans le catalogue hotshots.

```
#include <dmbshot.h>

rc = DBvUpdateShot("hotshots", shot,
       shothandle, hdbc);
```

DMBRedistribute (Produit EEE uniquement)

Image	Audio	Vidéo
X		

Redistribue les données présentant des caractéristiques QBIC lors de l'ajout ou de la suppression d'un noeud dans un groupe de noeuds ou lors de l'établissement d'un plan de partition pour le groupe de noeuds.

Classe d'autorisation

L'API doit être exécutée à partir de l'ID propriétaire de l'instance.

Fichier bibliothèque

Windows	AIX et Solaris
dmbrd.lib	libdmbrd.a (AIX)
	libdmbrd.so (Solaris)

Fichier d'inclusion

dmbrdst.h

Syntaxe

```
long DMBRedistribute (
       char  *pNodeGroupName,
       char  DataRedistOption    /* ""continue"" utilisez le paramètre CONTINUE *
       );
                                 /* espace : démarrez la redistribution */
```

Paramètres

pNodeGroupName (entrée)
Nom du groupe de noeuds à redistribuer.

Codes d'erreur

MMDB_SUCCESS
Le traitement de l'appel d'API a abouti.

MMDB_RD_NO_CONTINUE
Nouvelle soumission sans le paramètre CONTINUE.

MMDB_RD_CONTINUE
Nouvelle soumission avec le paramètre CONTINUE.

Exemples

Redistribution des données d'extensions QBIC dans le groupe de noeuds groupone :

```
#include <dmbrdst.h>

rc = DMBRedistribute(groupone,"continue");
```

QbAddFeature

Image	Audio	Vidéo
X		

Ajoute une fonction au catalogue ouvert. QbAddFeature crée la table pour la caractéristique indiquée dans la base de données. Après avoir ajouté des images dans la colonne Image de la table utilisateur, faites appel à l'API QbReCatalogColumn qui permet d'ajouter une entrée pour chaque image dans la table de fonctions et d'analyser les images.

Classe d'autorisation

Alter

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbAddFeature(
      QbCatalogHandle cHdl,
      char *featureName
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

featureName (entrée)
Nom de la fonction. Les fonctions suivantes sont fournies avec l'extension Image :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

qbicECCatalogReadOnly
Le catalogue est ouvert en lecture seulement.

qbicECDupFeature
La fonction figure déjà dans le catalogue.

qbiECinvalidFeatureClass
Le format du nom de la fonction indiquée n'est pas correct.

Exemples

Ajout de la fonction QbColorFeatureClass au catalogue identifié par le descripteur CatHdl :

```
#include <dmbqbapi.h>

rc = QbAddFeature(CatHdl,
       QbColorFeatureClass);
```

QbCatalogColumn

Image	Audio	Vidéo
X		

Catalogue les images dans la colonne Image de la table utilisateur qui n'ont pas déjà été cataloguées. L'API ajoute une entrée pour chaque image dans la table de fonctions et analyse les images. Lorsque l'API analyse l'image, elle crée des données d'image et les stocke dans l'entrée image de la table de fonctions. Les paramètres par défaut de la fonction sont utilisés. Le catalogue doit être ouvert.

Classe d'autorisation

Insert

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbCatalogColumn(
      QbCatalogHandle cHdl,
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

qbicECInvalidCatalog
Le descripteur ou la colonne de table indiqué n'est pas valide pour le catalogue.

Erreurs qbicECCatalog
Erreurs survenues et journalisées lors du catalogage d'images individuelles. L'annulation n'est pas effectuée.

qbicECImageNotFound
L'image est introuvable ou n'est pas accessible.

qbicECCatalogRO
Le catalogue est accessible en lecture seulement.

qbicECSQLError
Une erreur SQL s'est produite.

Exemples

```
#include <dmbqbapi.h>

rc = QbCatalogColumn(CatHdl);
```

QbCatalogImage

Image	Audio	Vidéo
X		

Catalogue une image entière. L'API ajoute une entrée pour l'image dans la table de fonctions et analyse l'image. Lorsque l'API analyse l'image, elle crée des données d'image et les stocke dans l'entrée image de la table de fonctions. Le descripteur d'image doit être extrait de la colonne Image associée au catalogue QBIC en cours. L'image est cataloguée en fonction des classes de fonctions définies. Les paramètres par défaut des fonctions du catalogue sont utilisés.

Classe d'autorisation

Insert

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbCatalogImage(
      QbCatalogHandle cHdl,
      char *imgHandle
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

imgHandle (entrée)
Descripteur de l'image.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

qbicECImageNotFound
L'image est introuvable ou n'est pas accessible.

qbicECCatalogRO
Le catalogue est accessible en lecture seulement.

Exemples

Catalogage d'une image identifiée par le descripteur Img_hdl :

```
#include <dmbqbapi.h>

rc = QbCatalogColumn(CatHdl, Img_hdl);
```

QbCloseCatalog

Image	Audio	Vidéo
X		

Ferme le catalogue. L'API libère le descripteur de catalogue ouvert et les ressources affectées.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbCloseCatalog(
      QbCatalogHandle cHdl
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

Exemples

Fermeture du catalogue identifié par le descripteur CatHdl :

```
#include <dmbqbapi.h>

rc = QbCloseCatalog(CatHdl);
```

QbCreateCatalog

Image	Audio	Vidéo
X		

Crée un catalogue dans la base de données connectée pour la colonne Image indiquée. La colonne doit être activée pour les données image. L'API crée le catalogue sous un nom qui est utilisé en tant que qualificatif.

Classe d'autorisation

Alter

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbCreateCatalog(
      char *tableName,
      char *columnName,
     SQLINTEGER autoCatalog,
     char *reserved
     );
```

Paramètres

tableName (entrée)
Nom de la table contenant une colonne Image.

columnName (entrée)
Nom de la colonne Image pour laquelle vous créez un catalogue.

autoCatalog (entrée)
Indique si les images ajoutées dans la colonne Image doivent être automatiquement cataloguées, par conséquent ajoutées aux tables de fonctions et analysées. Indiquez 1 pour activer la fonction de catalogage automatique, et 0, pour la désactiver. Si vous n'activez pas la fonction de catalogage automatique, utilisez l'API QbCatalogColumn ou QbCatalogImage pour cataloguer les images que vous ajoutez dans la colonne Image.

reserved (entrée)
Non utilisé.

Codes d'erreur

qbicECSqlError
Une erreur SQL s'est produite.

qbicECNotEnabled
La base de données, la table ou la colonne n'est pas activée pour le type de données DB2Image.

qbicECDupCatalog
Le catalogue existe déjà.

qbicECunsupportedOption
Une option non prise en charge a été indiquée.

qbicECerrorParameterTooLong
Un paramètre indiqué est trop long pour être traité.

qbicECqerr
Une erreur QBIC s'est produite, et un message a été généré.

qbicECqerrUnknown
Une erreur QBIC interne s'est produite, et un message générique a été émis.

Exemples

Création d'un catalogue pour les images dans la colonne Photo de la table Employés. Activation du catalogage automatique :

```
#include <dmbqbapi.h>

rc = QbCreateCatalog("Employés",
        "photo", 1);
```

QbDeleteCatalog

Image	Audio	Vidéo
X		

Supprime le catalogue indiqué de la base de données en cours.

Classe d'autorisation

Alter

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbDeleteCatalog(
      char *tableName,
      char *columnName
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant la colonne Image.

columnName (entrée)
Nom de la colonne Image associée au catalogue.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

qbicECCatalogInUse
Le catalogue est en cours d'utilisation par un autre utilisateur.

qbicECCatalogRO
Le catalogue est accessible en lecture seulement.

qbicECSystem
Une erreur système s'est produite.

qbicECSqlError
Une erreur SQL s'est produite.

Exemples

Suppression du catalogue QBIC associé à la colonne Photo de la table Employés.

```
#include <dmbqbapi.h>

rc=QbDeleteCatalog("Employés", "photo");
```

QbGetCatalogInfo

Image	Audio	Vidéo
X		

Renvoie une structure QbCatalogInfo contenant les informations suivantes :

- nom de la table utilisateur et de la colonne image auxquelles le catalogue appartient ;
- nombre de fonctions contenues dans le catalogue ;
- indique si la fonction de catalogage automatique est activée.

Classe d'autorisation

SELECT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbGetCatalogInfo(
      QbCatalogHandle cHdl,
      QbCatalogInfo *infoCat
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

infoCat (sortie)
Informations relatives au catalogue.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

Exemples

Extraction d'informations relatives au catalogue identifié par le descripteur CatHdl et renvoi de ces informations dans une structure appelée catInfo.

```
#include <dmbqbapi.h>

rc = QbGetCatalogInfo(CatHdl, &catInfo);
```

QbListFeatures

Image	Audio	Vidéo
X		

Renvoie la liste des fonctions actuellement actives contenues dans un catalogue. La liste est renvoyée dans la mémoire tampon affectée.

Classe d'autorisation

SELECT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbListFeatures(
      QbCatalogHandle cHdl,
      SQLINTEGER bufSize,
      SQLINTEGER *count,
      char *featureNames
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

bufSize (entrée)
Capacité de la mémoire tampon. Pour estimer la taille de mémoire tampon nécessaire, vous pouvez utiliser le nombre de fonctions renvoyé par l'API QbGetCatalogInfo et le multiplier par la longueur du nom de fonction le plus long. Les noms de fonctions stockés en mémoire tampon sont séparés par un espace.

count (sortie)
Nombre de noms de fonctions renvoyés.

featureNames (sortie)
Tableau de noms de fonctions dans la mémoire tampon.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

qbicECTruncateData
Les données sont tronquées car la mémoire tampon de réception est insuffisante.

Exemples

Insertion d'une liste des fonctions actives dans le catalogue identifié par le descripteur CatHdl. Stockage des informations dans le tableau featureNames.

Dans un premier temps, calculez la taille de la mémoire tampon nécessaire. Utilisez l'API QbGetCatalogInfo pour renvoyer le nombre de fonctions dans la structure catInfo. Multipliez ensuite ce nombre par la valeur de la constante qbiMaxFeatureName qui correspond à la taille du nom de fonction le plus long.

```
#include <dmbqbapi.h>

rc = QbGetCatalogInfo(CatHdl, &catInfo);

bufSize =
   catInfo.featureCount*qbiMaxFeatureName;

rc = QbListFeatures(CatHdl, bufSize,
       count, featureNames);
```

QbOpenCatalog

Image	Audio	Vidéo
X		

Ouvre le catalogue QBIC pour une colonne Image spécifique. Vous pouvez ouvrir le catalogue en mode lecture ou mise à jour. L'API renvoie un descripteur pour le catalogue ouvert. Vous pouvez ensuite utiliser le descripteur dans d'autres API pour gérer et personnaliser le catalogue.

N'oubliez pas de fermer le catalogue lorsque vous ne l'utilisez plus.

Classe d'autorisation

Aucune

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbOpenCatalog(
      char *tableName,
      char *columnName,
      SQLINTEGER mode,
      QbCatalogHandle *cHdl
      );
```

Paramètres

tableName (entrée)
Nom de la table contenant la colonne Image.

columnName (entrée)
Nom de la colonne Image.

mode (entrée)
Mode dans lequel vous ouvrez le catalogue. Les valeurs correctes sont qbiRead et qbiUpdate.

cHdl (sortie)
Pointeur sur le descripteur du catalogue.

Codes d'erreur

qbicECCatalogNotFound
Le catalogue est introuvable.

qbicECCatalogInUse
Le catalogue est en cours d'utilisation par un autre utilisateur.

qbicECOpenFailed
Le catalogue n'a pas pu être ouvert.

qbicECNotEnabled
Le catalogue n'est pas activé.

qbicECNoCatalogFound
Aucun catalogue n'a été trouvé.

qbicECSqlError
Une erreur SQL s'est produite.

qbicECSystem
Une erreur système s'est produite.

Exemples

Ouverture en mode lecture du catalogue correspondant à la colonne Photo dans la table Employés.

```
#include <dmbqbapi.h>

rc=QbOpenCatalog("Employés",  "photo",
     qbiread, &CatHdl);
```

QbQueryAddFeature

Image	Audio	Vidéo
X		

Ajoute la caractéristique spécifiée au catalogue QBIC.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryAddFeature(
      QbQueryHandle qObj,
      char *featureName
      );
```

Paramètres

qObj (entrée)
Descripteur de l'objet de la requête.

featureName (entrée)
Nom de la requête à ajouter. Les fonctions suivantes sont fournies avec l'extension Image :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objetde la requête indiqué ne fait pas référence à un objet de requête valide.

qbiECunknownFeatureClass
Le nom de classe de la fonction indiquée n'est pas reconnu.

qbiECinvalidFeatureClass
Le format du nom de la fonction indiquée n'est pas correct.

qbiECfeaturePresent
La fonction indiquée est déjà un membre de l'objet de la requête.

qbiECallocation
Le système ne peut pas affecter suffisamment de mémoire.

Exemples

Ajout de la fonction QbColorFeatureClass à l'objet de la requête identifié par le descripteur qoHandle :

```
#include <dmbqbapi.h>

rc = QbQueryAddFeature(qoHandle,
        "QbColorFeatureClass");
```

QbQueryCreate

Image	Audio	Vidéo
X		

Crée un objet de requête et renvoie un descripteur. Vous pouvez utiliser le descripteur avec d'autres API pour manipuler l'objet de la requête.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryCreate(
      QbQueryHandle *qObj
      );
```

Paramètres

qObj (sortie)
Pointeur sur le descripteur de requête. Si la requête n'aboutit pas, le descripteur a la valeur 0.

Codes d'erreur

qbiECallocation
Le système ne peut pas affecter suffisamment de mémoire.

Exemples

Création d'un objet de requête et renvoi du descripteur dans qoHandle :

```
#include <dmbqbapi.h>

rc = QbQueryCreate(&qoHandle);
```

QbQueryDelete

Image	Audio	Vidéo
X		

Supprime l'objet d'une requête non nommée. L'API libère la mémoire utilisée par l'objet de la requête et les fonctions ajoutées.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryDelete(
      QbQueryHandle qObj
      );
```

Paramètres

qObj (entrée)
Descripteur de l'objet de la requête.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objet de la requête indiqué ne fait pas référence à une requête correcte.

Exemples

Suppression de l'objet identifié par le descripteur qoHandle :

```
#include <dmbqbapi.h>
rc = QbQueryDelete(qoHandle);
```

QbQueryGetFeatureCount

Image	Audio	Vidéo
X		

Renvoie le nombre de fonctions ajoutées à l'objet de la requête. Les caractéristiques suivantes sont fournies avec l'extension Image :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryGetFeatureCount(
      QbQueryHandle qObj,
      SQLINTEGER* count
      );
```

Paramètres

qObj (entrée)
Descripteur de l'objet de la requête.

count (sortie)
Pointeur sur la variable définissant le nombre de fonctions présentes.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objet de la requête indiqué ne fait pas référence à un objet de requête valide.

Exemples

Renvoi du nombre de fonctions pour l'objet de la requête identifié par le descripteur qoHandle :

```
#include <dmbqbapi.h>

rc = QbQueryGetFeatureCount(qoHandle,
       &count);
```

QbQueryGetString

Image	Audio	Vidéo
X		

Renvoie la chaîne de requête de la requête. Celle-ci peut être utilisée en entrée dans des fonctions UDF de votre application, par exemple avec QbScoreFromStr ou l'API QbQueryStringSearch.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryGetString(
      QbQueryHandle qObj,
      (char*)*  queryString
      );
```

Paramètres

qObj (entrée)
Descripteur de l'objet de requête.

queryString (sortie)
Pointeur sur la chaîne de requête de l'objet de la requête.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de requête indiqué ne fait pas référence à une requête correcte.

Exemples

Renvoi de la chaîne de l'objet de requête identifié par le descripteur qrHandle.

```
#include <dmbqbapi.h>

  SQLRETURN rc;
char *queryString;
QbQueryHandle qrHandle
```

```
rc = QbQueryGetString(qrHandle, &queryString);
if (rc == 0)  {
   ...                  /* use the returned queryString for input to UDFs */
   free((void*)queryString);  /* you must free queryString  */
   queryString=(char*)0;
}
```

QbQueryListFeatures

Image	Audio	Vidéo
X		

Renvoie une liste à jour de fonctions dans l'objet de la requête. L'API renvoie la liste dans la mémoire tampon que vous avez désignée. Les caractéristiques suivantes sont fournies avec l'extension Image :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryListFeatures(
      QbQueryHandle qObj,
      SQLINTEGER bufSize,
      SQLINTEGER* count,
      char *featureNames
      );
```

Paramètres

qObj (entrée)
Descripteur de l'objet de la requête.

bufSize (entrée)
Taille de la mémoire tampon du tableau featureNames. Utilisez la valeur de la constante qbiMaxFeatureName comme taille de mémoire tampon. Les fonctions de l'objet de la requête sont identifiées par une chaîne de caractères.

count (sortie)
Nombre de noms de fonctions renvoyés.

featureNames (sortie)
Pointeur sur le tableau des noms des fonctions contenues dans l'objet de la requête. Le tableau est stocké dans la mémoire tampon que vous avez affectée.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de requête indiqué ne fait pas référence à une requête correcte.

Exemples

Renvoi du nombre de fonctions dans l'objet de la requête identifié par le descripteur qoHandle. Utilisez la valeur de la constante qbiMaxFeatureName pour déterminer la taille de la mémoire tampon nécessaire. Renvoi du nom de fonction à la mémoire tampon FEATS et du nombre de fonctions à la variable retCount :

```
#include <dmbqbapi.h>

bufSize = qbiMaxFeatureName;

rc = QbQueryListFeatures(qoHandle, bufSize,
        &retCount, feats);
```

QbQueryNameCreate

Image	Audio	Vidéo
X		

Stocke un objet de requête et lui affecte un nom de sorte que vous puissiez l'utiliser dans une fonction utilisateur (UDF). Indiquez le nom et, facultativement, la description de l'objet de la requête.

Remarques :

1. **Produit EEE uniquement :** QbQueryNameCreate n'est pas pris en charge dans un environnement de bases de données partitionnées.
2. La fonction QbQueryNameCreate ne sera plus utilisée dans les prochaines versions des environnements de bases de données non partitionnées. Pour sauvegarder une requête en vue de l'utiliser ultérieurement dans votre application, il est donc conseillé d'extraire la chaîne de requête à l'aide de l'API QbQueryGetString.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryNameCreate(
      QbQueryHandle qObj,
      char *name,
      char *description
      );
```

Paramètres

qObj (entrée)
Descripteur de l'objet de la requête.

name (entrée)
Nom de l'objet de la requête. Ce dernier peut comporter jusqu'à 18 caractères.

description (entrée)
Brève description de l'objet de la requête (250 caractères au maximum).

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objet de requête indiqué ne fait pas référence à une requête correcte.

Exemples

Affectation d'un nom et d'une description à l'objet de requête créé au moyen de l'API QbQueryCreate :

```
#include <dmbqbapi.h>

rc = QbQueryNameCreate(qHandle,
       "fshavgcol",
       "requête couleur moyenne, 10/15/96");
```

QbQueryNameDelete

Image	Audio	Vidéo
X		

Supprime l'objet d'une requête. L'objet de la requête doit avoir été nommé et stocké au moyen de l'API QbQueryNameCreate.

Remarques :

1. **Produit EEE uniquement :** QbQueryNameDelete n'est pas pris en charge dans un environnement de bases de données partitionnées.
2. La fonction QbQueryNameDelete ne sera plus utilisée dans les prochaines versions des environnements de bases de données non partitionnées.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryNameDelete(
      char *name
      );
```

Paramètres

name (entrée)
Nom de l'objet de requête que vous supprimez.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objetde la requête indiqué ne fait pas référence à un objet de requête valide.

Exemples

Suppression de l'objet de requête fshavgcol :

```
#include <dmbqbapi.h>

rc = QbQueryNameDelete("fshavgcol",);
```

QbQueryNameSearch

Image	Audio	Vidéo
X		

Recherche dans le catalogue QBIC les images répondant au critère de recherche indiqué dans un objet de requête. L'objet de la requête est identifié par son nom. Les résultats (comprenant les descripteurs d'image et les scores de recherche QBIC) sont consignés dans un tableau dans la mémoire client. Les résultats sont triés sur la valeur de leurs scores.

Remarques :

1. **Produit EEE uniquement :** QbQueryNameSearch n'est pas pris en charge dans un environnement de bases de données partitionnées.
2. La fonction QbQueryNameSearch ne sera plus utilisée dans les prochaines versions des environnements de bases de données non partitionnées. Pour sauvegarder une requête en vue de l'utiliser ultérieurement dans votre application, il est donc conseillé d'extraire la chaîne de requête à l'aide de l'API QbQueryGetString.

Classe d'autorisation

SELECT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryNameSearch(
      char *qName,
      char *tableName,
      char *columnName,
      SQLINTEGER maxReturns,
      QbQueryScope* scope,
      SQLINTEGER resultType,
      SQLINTEGER* count,
      QbResult* returns
      );
```

QbQueryNameSearch

Paramètres

qName (entrée)
Nom de l'objet de la requête.

tableName (entrée)
Nom de la table contenant la colonne Image dans laquelle vous souhaitez faire la recherche.

columnName (entrée)
Nom de la colonne Image. La colonne doit être activée pour les données image.

maxReturns (entrée)
Nombre maximal d'images que vous souhaitez renvoyer.

scope (entrée) (Réservé)
Doit avoir la valeur 0 (NULL).

resultType (entrée) (Réservé)
Doit avoir la valeur de qbiArray.

count (sortie)
Pointeur sur le nombre d'images renvoyé. Si la valeur zéro est renvoyée, assurez-vous que la colonne Image est cataloguée pour toutes les fonctions dans l'objet de requête.

returns (sortie)
Pointeur sur le tableau des structures QbResult contenant les résultats renvoyés. Assurez-vous que vous avez affecté suffisamment de mémoire tampon pour prendre en charge les résultats attendus.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objetde la requête indiqué ne fait pas référence à un objet de requête correct.

Exemples

Exécution de la requête FSHAVGCOL sur les images cataloguées dans la colonne Photo de la table Employés. Assurez-vous que le nombre d'images renvoyé ne dépasse pas six.

```
#include <dmbqbapi.h>

rc = QbQueryNameSearch("fshavgcol",
       "Employés", "photo",
       6, 0, qbiArray, &count, &returns);
```

QbQueryRemoveFeature

Image	Audio	Vidéo
X		

Supprime une fonction dans l'objet de la requête ainsi que toute mémoire associée. Les caractéristiques suivantes sont fournies avec l'extension Image :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryRemoveFeature(
      QbQueryHandle qObj,
      char *featureName
      );
```

Paramètres

qObj (entrée)
Descripteur de l'objet de la requête.

featureName (entrée)
Nom de la fonction à supprimer.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objet de requête spécifié ne fait pas référence à un objet de requête correct.

qbiECinvalidFeatureClass
Le format du nom de la fonction indiquée n'est pas correct.

qbiECfeatureNotPresent
La fonction indiquée n'est pas membre de l'objet de la requête.

Exemples

Suppression de la fonction QbColorFeatureClass de l'objet de la requête identifié par le descripteur qoHandle :

```
#include <dmbqbapi.h>

rc = QbQueryRemoveFeature(qoHandle,
        "QbColorFeatureClass");
```

QbQuerySearch

Image	Audio	Vidéo
X		

Recherche dans le catalogue QBIC les images répondant au critère de recherche indiqué dans un objet de requête. L'objet de la requête est identifié par un descripteur d'objet de requête. Les résultats (comprenant les descripteurs d'image et les scores de recherche QBIC) sont consignés dans un tableau dans la mémoire client. Ils sont triés sur la valeur de leurs scores.

Classe d'autorisation

SELECT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQuerySearch(
      QbQueryHandle qObj,
      char *tableName,
      char *columnName,
      SQLINTEGER maxReturns,
      QbQueryScope* scope,
      SQLINTEGER resultType,
      SQLINTEGER* count,
      QbResult* returns
      );
```

Paramètres

qObj (entrée)
Descripteur de l'objet de la requête.

tableName (entrée)
Nom de la table contenant la colonne Image dans laquelle vous souhaitez faire la recherche.

columnName (entrée)
Nom de la colonne Image. La colonne doit être activée pour les données image.

maxReturns (entrée)
Nombre maximal d'images que vous souhaitez renvoyer.

scope (entrée) (Réservé)
Doit avoir la valeur 0 (NULL).

resultType (entrée) (Réservé)
Doit avoir la valeur de qbiArray.

count (sortie)
Pointeur sur le nombre d'images renvoyé. Si la valeur zéro est renvoyée, assurez-vous que la colonne Image est cataloguée pour toutes les fonctions dans l'objet de requête.

returns (sortie)
Pointeur sur le tableau des structures QbResult contenant les résultats renvoyés. Assurez-vous que vous avez affecté suffisamment de mémoire tampon pour prendre en charge les résultats attendus.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objet de requête spécifié ne fait pas référence à un objet de requête correct.

Exemples

Interrogation des images cataloguées dans la colonne Photo de la table Employés. Assurez-vous que le nombre d'images renvoyé ne dépasse pas six.

```
#include <dmbqbapi.h>

rc = QbQuerySearch(qHandle, "Employés",
       "picture", 6, 0, qbiArray,
       &count, &returns);
```

QbQuerySetFeatureData

Image	Audio	Vidéo
X		

Définit la source de données image pour une fonction dans un objet de requête. Cette opération ne peut être effectuée qu'une fois la fonction ajoutée à l'objet de la requête. La source de données peut être une image dans une table de base de données, un fichier ou la mémoire tampon d'un poste de travail. Un fichier client ou une mémoire tampon de poste de travail ne peut être utilisé comme source de données que dans un environnement de bases de données partitionnées. En outre, vous pouvez indiquer explicitement des données pour la caractéristique couleur moyenne ou couleur d'histogramme.

Utilisez la fonction QbQueryStringSearch après avoir défini la source de données image dans un fichier serveur à l'aide de QbQuerySetFeatureData. QbQuerySearch ne porte pas sur la source des données image d'un fichier serveur définie avec QbQuerySetFeatureData.

Les fonctions suivantes sont fournies avec l'extension Image :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQuerySetFeatureData(
      QbQueryHandle qObj,
      char *featureName,
      QbImageSource* imgSource
      );
```

QbQuerySetFeatureData

Paramètres

qObj (entrée)
Descripteur de l'objet de la requête.

featureName (entrée)
Nom de la fonction à définir.

imgSource (entrée)
Pointeur sur la structure de l'image. Si vous indiquez la valeur 0 (NULL) pour le paramètre sourceImg, cela signifie que les informations ne doivent pas être modifiées dans la fonction. Pour plus de détails, reportez-vous à la section «Utilisation de structures de source de données» à la page 160.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objet de requête spécifié ne fait pas référence à un objet de requête correct.

qbiECunknownFeatureClass
Le nom de classe de la fonction indiquée n'est pas reconnu.

qbiECinvalidFeatureClass
Le format du nom de la fonction indiquée n'est pas correct.

qbiECfeatureNotPresent
La fonction indiquée n'est pas membre de l'objet de la requête.

qbiECfileUnreadable
Le fichier source d'images est introuvable ou ne peut être lu.

Exemples

Définition de la source de données pour la caractéristique couleur d'histogramme dans un objet de requête. La source de données de la caractéristique est un fichier sur le poste de travail client :

```
#include <dmbqbapi.h>

QbQueryHandle qoHandle;
QbImageSource imgSource;

imgSource.sourceType = qbiSource_ClientFile;
strcpy(featureName, "QbColorHistogramFeatureClass");
strcpy(imgSource.clientFile, "/tmp/image.gif");

rc = QbQuerySetFeatureData(qoHandle, featureName, &imgSource);
```

QbQuerySetFeatureWeight

Image	Audio	Vidéo
X		

Renvoie le poids de la fonction indiquée dans un objet de requête.

Classe d'autorisation

Aucune.

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQuerySetFeatureWeight(
      QbQueryHandle qObj,
      sqldouble* weight
      );
```

Paramètres

qObj (entrée)
Descripteur de l'objet de la requête.

weight (sortie)
Pointeur sur la variable définissant le poids des fonctions.

Codes d'erreur

qbiECinvalidQueryHandle
Le descripteur de l'objet de requête spécifié ne fait pas référence à un objet de requête correct.

Exemples

Extraction du poids affecté à la caractéristique de couleur moyenne figurant dans un objet de requête identifié par le descripteur qoHandle :

```
#include <dmbqbapi.h>

weight=2.0
rc = QbQuerySetFeatureWeight(qoHandle, "QbColorFeatureClass", &weight);
```

QbQueryStringSearch

Image	Audio	Vidéo
X		

Recherche dans le catalogue QBIC les images répondant au critère de recherche indiqué dans une chaîne de requête. Les résultats (comprenant les descripteurs d'image et les scores de recherche QBIC) sont consignés dans un tableau dans la mémoire client. Ils sont triés sur la valeur de leurs scores.

Classe d'autorisation

SELECT

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqqry.lib	libdmbqqry.a (AIX) libdmbqqry.sl (HP-UX) libdmbqqry.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbQueryStringSearch(
      char *queryString,
      char *tableName,
      char *columnName,
      SQLINTEGER maxReturns,
      QbQueryScope* scope,
      SQLINTEGER resultType,
      SQLINTEGER* count,
      QbResult* returns
      );
```

Paramètres

queryString (entrée)
Chaîne de requête.

tableName (entrée)
Nom de la table contenant la colonne Image dans laquelle vous souhaitez faire la recherche.

columnName (entrée)
Nom de la colonne Image. La colonne doit être activée pour les données image.

maxReturns (entrée)
Nombre maximal d'images que vous souhaitez renvoyer.

scope (entrée) (Réservé)
Doit avoir la valeur 0 (NULL).

resultType (entrée) (Réservé)
Doit avoir la valeur de qbiArray.

count (sortie)
Pointeur sur le nombre d'images renvoyé. Si la valeur zéro est renvoyée, assurez-vous que la colonne Image est cataloguée pour toutes les fonctions dans la chaîne de requête.

returns (sortie)
Pointeur sur le tableau des structures QbResult contenant les résultats renvoyés. Assurez-vous que vous avez affecté suffisamment de mémoire tampon pour prendre en charge les résultats attendus.

Codes d'erreur

qbiECinvalidQueryString
La chaîne de requête indiquée est incorrecte.

Exemples

Interrogation des images cataloguées dans la colonne Photo de la table Employés. Assurez-vous que le nombre d'images renvoyé ne dépasse pas six.

```
#include <dmbqbapi.h>

rc = QbQueryStringSearch("QbColorFeatureClass color=<255, 0, 0>"
       "Employés",
       "picture", 6, 0, qbiArray,
       &count, &returns);
```

QbReCatalogColumn

Image	Audio	Vidéo
X		

Ré-analyse toutes les images existantes dans le catalogue QBIC ouvert pour une nouvelle fonction. Les paramètres par défaut des fonctions sont utilisés. Utilisez cette API lorsque vous ajoutez une nouvelle fonction au catalogue contenant déjà des images.

Classe d'autorisation

Update, Insert

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbReCatalogColumn (
      QbCatalogHandle cHdl
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

qbicECInvalidCatalog
Le descripteur ou la colonne de table indiqué n'est pas valide pour le catalogue.

Erreurs qbicECCatalog
Erreurs survenues et journalisées lors du catalogage d'images individuelles. L'annulation n'est pas effectuée.

qbicECImageNotFound
L'image est introuvable ou n'est pas accessible.

qbicECCatalogRO
Le catalogue est accessible en lecture seulement.

qbicECSQLError
Une erreur SQL s'est produite.

Exemples

Ré-analyse toutes les images existantes dans le catalogue QBIC ouvert pour une nouvelle fonction.

```
#include <dmbqbapi.h>

rc = QbReCatalogColumn(CatHdl);
```

QbRemoveFeature

Image	Audio	Vidéo
X		

Supprime la table de fonctions de la fonction indiquée du catalogue ouvert.

Classe d'autorisation

Alter

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbRemoveFeature(
      QbCatalogHandle cHdl,
      char *featureName
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

featureName (entrée)
Nom de la fonction.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

qbicECCatalogReadOnly
Le catalogue est ouvert en lecture seulement.

qbicECFeatureNotFound
La fonction ne figure pas dans le catalogue.

qbiECinvalidFeatureClass
Le format du nom de la fonction indiquée n'est pas correct.

Exemples

Suppression de la fonction QbColorHistogramFeatureClass dans le catalogue identifié par le descripteur CatHdl.

```
#include <dmbqbapi.h>

rc=QbRemoveFeature(CatHdl,
     "QbColorHistogramFeatureClass");
```

QbSetAutoCatalog

Image	Audio	Vidéo
X		

Catalogue automatiquement les images importées dans une colonne Image. L'API ajoute une entrée pour chaque image dans la table de fonctions et analyse les images. Lorsque l'API analyse l'image, elle crée des données d'image et les stocke dans l'entrée image de la table de fonctions.

Si vous n'activez pas la fonction de catalogage automatique, utilisez l'API QbCatalogColumn ou QbCatalogImage pour cataloguer des images une fois qu'elles ont été ajoutées dans la colonne Image.

Classe d'autorisation

Alter

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbSetAutoCatalog(
      QbCatalogHandle cHdl
      SQLINTEGER autoCatalog
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

autoCatalog (entrée)
Indique si les images ajoutées dans la colonne Image seront automatiquement ajoutées aux tables de fonctions et analysées. Indiquez 1 pour activer la fonction de catalogage automatique, et 0, pour la désactiver.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

Exemples

Activation de la fonction de catalogage automatique pour le catalogue identifié par le descripteur CatHdl :

```
#include <dmbqbapi.h>

rc=QbSetAutoCatalog(CatHdl, 1);
```

QbUncatalogImage

Image	Audio	Vidéo
X		

Supprime une image d'un catalogue. Le descripteur d'image doit être extrait de la colonne Image associée au catalogue QBIC ouvert. L'image est ensuite supprimée du catalogue ouvert. La ligne correspondante dans la table d'attributs d'image indique que l'image n'est pas cataloguée.

Classe d'autorisation

Delete

Fichier bibliothèque

OS/2 et Windows	AIX, HP-UX et Solaris
dmbqbapi.lib	libdmbqbapi.a (AIX) libdmbqbapi.sl (HP-UX) libdmbqbapi.so (Solaris)

Fichier d'inclusion

dmbqbapi.h

Syntaxe

```
SQLRETURN QbUncatalogImage(
      QbCatalogHandle cHdl,
      char *imgHandle
      );
```

Paramètres

cHdl (entrée)
Pointeur sur le descripteur du catalogue.

imgHandle (entrée)
Descripteur de l'image. Vous pouvez extraire ce descripteur à partir de la table utilisateur.

Codes d'erreur

qbicECIvalidHandle
Le descripteur de catalogue n'est pas correct.

qbicECImageNotFound
L'image est introuvable ou n'est pas accessible.

qbicECCatalogRO
Le catalogue est accessible en lecture seulement.

Exemples

Suppression de l'image identifiée par le descripteur Img_hdl à partir du catalogue identifié par le descripteur CatHdl.

```
#include <dmbqbapi.h>

rc=QbUncatalogImage(CatHdl, Img_hdl);
```

QbUncatalogImage

Chapitre 17. Commandes d'administration du poste client

Le présent chapitre décrit la procédure d'entrée des commandes d'administration de DB2 Extensions pour le poste client. Il comprend également des informations de référence sur chacune de ces commandes.

Entrée des commandes d'administration de DB2 Extensions

Vous pouvez entrer les commandes d'administration de DB2 Extensions à partir de l'interpréteur de commandes db2ext en mode interactif ou en mode commande. Le mode interactif se distingue par l'invite db2ext. Dans ce mode, vous ne pouvez saisir que des commandes d'administration de DB2 Extensions. Dans le mode commande, vous entrez les commandes à partir de l'invite du système d'exploitation ; il peut s'agir de commandes inhérentes à DB2 Extensions comme de commandes DB2 ou propres au système d'exploitation.

N'entrez pas de commandes DB2 Extensions à partir de l'invite DB2.

Pour démarrer l'interpréteur de commandes db2ext en mode interactif, procédez comme suit :

Client	Opération
OS/2	Cliquez deux fois sur l'icône correspondant à l'interpréteur de commandes DB2EXT dans le dossier DB2 Extensions, ou entrez la commande DB2EXT à partir de l'invite OS/2.
AIX, HP-UX, Solaris	Entrez la commande DB2EXT à partir de l'invite du système d'exploitation.
Windows	Cliquez deux fois sur l'icône correspondant à l'interpréteur de commandes DB2EXT dans le dossier DB2 Extensions, ou entrez la commande DB2EXT à partir de la fenêtre de commande DB2.

Pour désactiver le mode interactif, entrez la commande QUIT ou TERMINATE. La commande QUIT ferme le mode interactif mais maintient ouverte la connexion avec DB2 en cours. La commande TERMINATE ferme le mode interactif et interrompt la connexion avec DB2 en cours.

Pour lancer les commandes de DB2 Extensions en mode commande, entrez-les à partir de l'invite du système d'exploitation. Vous devez toutes les faire précéder de db2ext, par exemple :

```
db2ext
enable database for db2image using mydataspace, myindxspace, mylongspace
```

Accès à l'aide en ligne pour les commandes DB2 Extensions

Pour accéder à l'aide en ligne des commandes de DB2 Extensions, entrez :

```
db2ext ?
```

ADD QBIC FEATURE

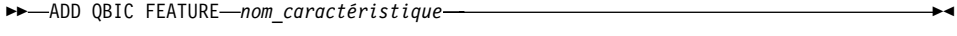
Image	Audio	Vidéo
X		

Crée une table de caractéristiques pour la caractéristique spécifiée dans le catalogue en cours. Les images existantes du catalogue ne sont pas automatiquement réanalysées par l'extension Image.

Classe d'autorisation

Alter, Control, SYSADM, DBADM

Syntaxe de la commande



Paramètres de la commande

nom_caractéristique
Nom de la caractéristique ajoutée au catalogue QBIC. Les fonctions suivantes sont fournies avec l'extension Image :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Exemples

Ajout de la caractéristique QbColorFeatureClass au catalogue ouvert :

```
add qbic feature qbcolorfeatureclass
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Le catalogue doit être ouvert.

CATALOG QBIC COLUMN

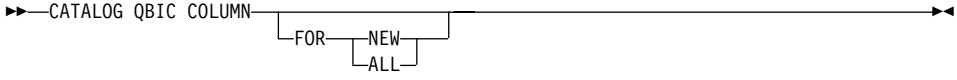
Image	Audio	Vidéo
X		

Catalogue les images dans la colonne Image et met à jour le catalogue QBIC ouvert en fonction des données de caractéristiques. Vous pouvez mettre à jour le catalogue en utilisant toutes les images de la colonne correspondante ou uniquement les images ajoutées à cette colonne depuis la dernière analyse du catalogue.

Classe d'autorisation

Insert, Control, SYSADM, DBADM

Syntaxe de la commande



Paramètres de la commande

Aucune.

Exemples

Catalogage des nouvelles images dans le catalogue en cours, c'est-à-dire des images non cataloguées :

```
catalog qbic column for new
```

Remarques

Lorsque NEW est spécifié, l'extension Image met à jour le catalogue en utilisant uniquement les images non cataloguées. Lorsque ALL est spécifié, l'extension Image analyse toutes les images de la colonne correspondante du catalogue en cours. NEW est la valeur par défaut.

Connectez-vous à la base de données avant d'utiliser cette commande.

Le catalogue doit être ouvert.

CLOSE QBIC CATALOG

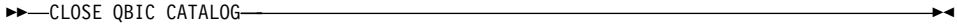
Image	Audio	Vidéo
X		

Ferme un catalogue QBIC.

Classe d'autorisation

Aucune.

Syntaxe de la commande



Paramètres de la commande

Aucune.

Exemples

Fermeture du catalogue en cours :

```
close qbic catalog
```

Remarques

Le catalogue QBIC doit être ouvert.

CONNECT

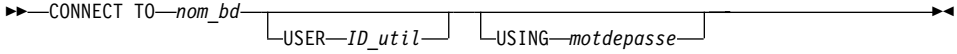
Image	Audio	Vidéo
X	X	X

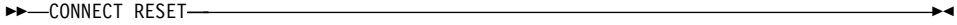
Effectue la connexion à une base de données. Les extensions exigent une connexion indépendante à la base de données, distincte de la connexion DB2.

Classe d'autorisation

Connect

Syntaxe de la commande





Paramètres de la commande

nom_bd
Nom de la base de données.

ID_util
ID utilisateur ayant le droit de se connecter à la base de données.

motdepasse
Mot de passe correspondant à l'ID utilisateur.

RESET
Déconnecte la base de données après validation des modifications en attente.

Exemples

Connexion à la base de données PERSONNL. L'ID utilisateur est anne et le mot de passe annepas :

```
connect to personnl user anne
   using annepas
```

Remarques

La base de données est connectée en mode SHARE (partagé).

Exécutez cette commande avant toute autre commande de l'extension.

CREATE QBIC CATALOG

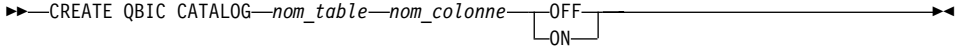
Image	Audio	Vidéo
X		

Crée un catalogue QBIC dans la base de données en cours pour la colonne DB2IMAGE indiquée. L'extension génère automatiquement le nom du catalogue.

Classe d'autorisation

Alter, Control, SYSADM, DBADM

Syntaxe de la commande



Paramètres de la commande

nom_table
Nom de la table DB2IMAGE activée.

nom_colonne
Nom de la colonne DB2IMAGE activée.

OFF Les images sont cataloguées manuellement.

ON Les images sont cataloguées automatiquement.

nom_espacetable
Spécification de l'espace table et options d'indexation pour le catalogue QBIC. La spécification comporte les quatre parties suivantes :

- Le nom de l'espace table pour les tables du catalogue qui contiennent des données relatives aux caractéristiques. L'espace table doit être indiqué. Il doit s'agir d'un espace table segmenté.
- Toute combinaison de bloc d'utilisation, bloc disponible et bloc gbpcache, ou d'options d'indexation pour les index non partitionnés de type 2. Cette combinaison concerne l'index créé pour les tables du catalogue. Cette spécification est facultative. Si vous ne l'indiquez pas, les valeurs par défaut s'appliquent.
- Le nom de l'espace table pour la table journal du catalogue. Cet espace table peut être de type simple ou segmenté. Cette spécification est facultative. Si vous n'indiquez pas d'espace table pour la table journal, le système utilise l'espace table défini pour les données relatives aux caractéristiques.

- Toute combinaison de bloc d'utilisation, bloc disponible et bloc gbpcache, ou d'options d'indexation pour les index non partitionnés de type 2. Cette combinaison concerne l'index créé pour la table d'enregistrements d'erreurs. Cette spécification est facultative. Si vous ne l'indiquez pas, les valeurs par défaut s'appliquent.

Exemples

Création d'un catalogue QBIC pour la colonne photo dans la table Employés, avec activation du catalogage automatique (ON) :

```
create qbic catalog Employés photo on
```

Remarques

Si vous indiquez ON, les images importées dans la colonne sont automatiquement cataloguées dans le catalogue QBIC associé. La valeur par défaut est OFF.

Connectez-vous à la base de données avant d'utiliser cette commande.

DELETE QBIC CATALOG

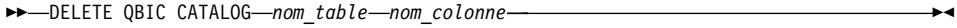
Image	Audio	Vidéo
X		

Supprime un catalogue QBIC, ainsi que toutes les données de recherche QBIC.

Classe d'autorisation

Alter, Control, SYSADM, DBADM

Syntaxe de la commande



Paramètres de la commande

nom_table
Nom de la table DB2IMAGE activée.

nom_colonne
Nom de la colonne DB2IMAGE activée.

Exemples

Suppression du catalogue QBIC associé à la colonne Photo de la table Employés :

```
delete qbic catalog Employés photo
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

DISABLE COLUMN

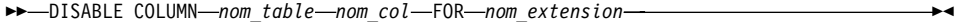
Image	Audio	Vidéo
X	X	X

Désactive la colonne spécifiée pour le stockage des données de support indiquées.

Classe d'autorisation

SYSADM, DBADM, Control, Alter

Syntaxe de la commande



Paramètres de la commande

nom_table
Nom de table dans la base de données en cours.

nom_col
Nom de la colonne à désactiver.

nom_extension
Nom de l'extension pour laquelle vous souhaitez désactiver la colonne. Les noms d'extension corrects sont db2image, db2audio et db2video.

Exemples

Désactivation de la colonne photo dans la table Employés, pour qu'elle ne puisse pas contenir des données Image :

```
disable column Employés photo for db2image
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Lorsque vous désactivez une colonne :

- Elle ne peut plus contenir de données pour l'extension indiquée. Cette restriction n'affecte pas l'activation ou la désactivation des autres colonnes de la table pour les types de données multimédia.
- Les entrées de la colonne ont une valeur NULL et les lignes correspondantes dans les tables d'administration sont supprimées.
- Les déclencheurs associés à la colonne sont annulés.

DISABLE DATABASE

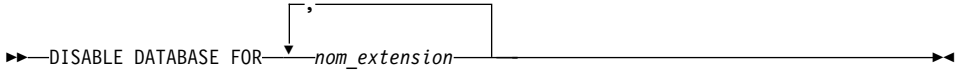
Image	Audio	Vidéo
X	X	X

Désactive la base de données en cours pour le stockage de données de support.

Classe d'autorisation

SYSADM, DBADM

Syntaxe de la commande



Paramètres de la commande

nom_extension
Nom de l'extension pour laquelle vous souhaitez désactiver la base de données en cours. Les noms d'extension corrects sont db2image, db2audio et db2video.

Exemples

Désactivation de la base de données en cours pour le stockage de données Image.

```
disable database for db2image
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Lorsque vous désactivez une base de données, le système :

- désactive toutes les tables activées pour l'extension spécifiée ;
- annule les tables de gestion UDF pour l'extension spécifiée.

DISABLE TABLE

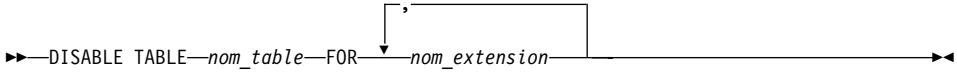
Image	Audio	Vidéo
X	X	X

Désactive la table spécifiée pour le stockage des données de support.

Classe d'autorisation

SYSADM, DBADM, Control, Alter

Syntaxe de la commande



Paramètres de la commande

nom_table
Nom de la table à désactiver dans la base de données en cours.

nom_extension
Nom de l'extension pour laquelle vous souhaitez désactiver la table. Les noms d'extension corrects sont db2image, db2audio et db2video.

Exemples

Désactivation de la table Employés pour le stockage de données Image :

```
disable table Employés for db2image
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Lorsque vous désactivez une table, le système :

- désactive toutes les colonnes de la table activées pour l'extension spécifiée ;
- annule les tables de gestion associées à cette table.

DISCONNECT SERVER AT NODENUM (Produit EEE uniquement)

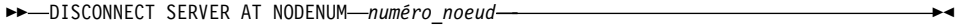
Image	Audio	Vidéo
X	X	X

Déconnecte le serveur du noeud spécifique sur toutes les bases de données.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT, DBADM

Syntaxe de la commande



Paramètres de la commande

numéro_noeud
Noeud que vous voulez déconnecter du serveur.

Exemples

Déconnexion du serveur de toutes les bases de données au noeud numéro 2 :

```
disconnect server at nodenum 2
```

Remarques

La commande DMBSTOP vous permet de déconnecter le serveur de toutes les bases de données sur tous les noeuds.

DISCONNECT SERVER FOR DATABASE (Produit EEE uniquement)

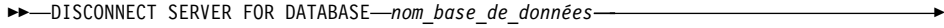
Image	Audio	Vidéo
X	X	X

Déconnecte le serveur de tous les noeuds de la base de données mentionnée.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT, DBADM

Syntaxe de la commande



Paramètres de la commande

nom_base_de_données
Base de données que vous voulez déconnecter du serveur.

Exemples

Déconnexion du serveur de la base de données MY_DATABASE :

```
disconnect server for database my_database
```

Remarques

La commande DMBSTOP vous permet de déconnecter le serveur de toutes les bases de données sur tous les noeuds.

DISCONNECT SERVER FOR DATABASE AT NODENUM (Produit EEE uniquement)

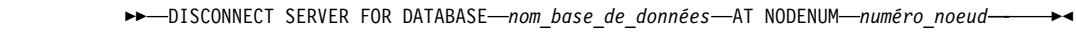
Image	Audio	Vidéo
X	X	X

Déconnecte le serveur de la base de données mentionnée au noeud mentionné.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT, DBADM

Syntaxe de la commande



Paramètres de la commande

nom_base_de_données
Base de données que vous voulez déconnecter du serveur.

numéro_noeud
Noeud que vous voulez déconnecter du serveur.

Exemples

Déconnexion du serveur de la base de données MY_DATABASE au noeud numéro 2 :

```
disconnect server for database my_database at nodenum 2
```

Remarques

La commande DMBSTOP vous permet de déconnecter le serveur de toutes les bases de données sur tous les noeuds.

ENABLE COLUMN

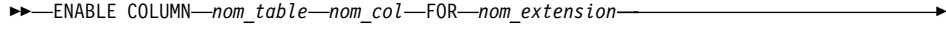
Image	Audio	Vidéo
X	X	X

Active la colonne indiquée pour le stockage de données de support.

Classe d'autorisation

SYSADM, DBADM, Control, Alter

Syntaxe de la commande



Paramètres de la commande

nom_table
Nom de table dans la base de données en cours.

nom_col
Nom de la colonne à activer.

nom_extension
Nom de l'extension pour laquelle vous souhaitez activer la table. Les noms d'extension corrects sont db2image, db2audio et db2video.

Exemples

Activation de la colonne photo dans la table Employés pour le stockage de données Image :

```
enable column Employés photo for db2image
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

ENABLE DATABASE

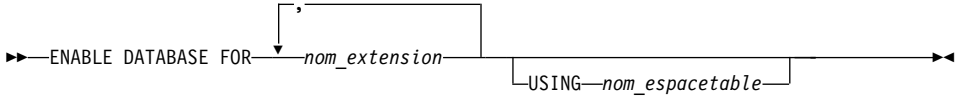
Image	Audio	Vidéo
X	X	X

Active la base de données en cours pour le stockage de données de support dans l'espace table spécifié.

Classe d'autorisation

SYSADM, SYSCTRL, DBADM

Syntaxe de la commande



Paramètres de la commande

nom_extension
Nom de l'extension pour laquelle vous souhaitez activer la base de données en cours. Les noms d'extension corrects sont db2image, db2audio et db2video.

nom_espacetable
Nom de l'espace table (ensemble de conteneurs dans lesquels des tables de gestion sont stockées). Le nom de l'espace table se compose des trois parties suivantes : *datats, indexts, longts*. *datats* est l'espace table dans lequel sont créées les tables de métadonnées, *indexts* l'espace table dans lequel sont créés les index des tables de métadonnées et *longts* l'espace table dans lequel sont stockées les valeurs contenues dans les colonnes longues des tables de métadonnées (telles que celles contenant les types de données LONG VARCHAR et LOB). Si vous indiquez une valeur NULL pour l'une des parties du nom de l'espace table, le nom de l'espace table par défaut pour cette partie est utilisé. L'espace table indiqué doit être défini sur un groupe de noeuds incluant tous les noeuds dans le système de bases de données partitionnées.

Exemples

Activation de la base de données en cours pour le stockage des données Image :

```
enable database for db2image using mydataspace, myindxspace, mylongspace
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Si aucun espace table n'est précisé, le système utilise l'espace table USERSPACE1 pour les tables de gestion.

ENABLE TABLE

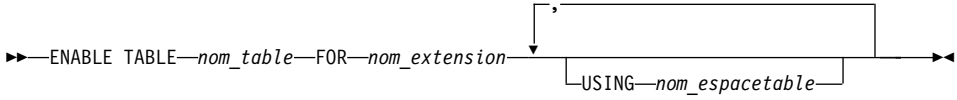
Image	Audio	Vidéo
X	X	X

Active la table indiquée pour le stockage de données de support dans l'espace table spécifié.

Classe d'autorisation

SYSADM, DBADM, Control, Alter

Syntaxe de la commande



Paramètres de la commande

nom_table
Nom de la table à activer dans la base de données en cours.

nom_extension
Nom de l'extension pour laquelle vous souhaitez activer la table. Les noms d'extension corrects sont db2image, db2audio et db2video.

nom_espacetable
Nom de l'espace table (ensemble de conteneurs dans lesquels des tables de gestion sont stockées). La spécification de l'espace table se compose des trois parties suivantes : *datats, indexts, longts*, *datats* étant l'espace table dans lequel sont créées les tables de métadonnées ; *indexts* l'espace table dans lequel sont créés les index des tables de métadonnées et *longts* l'espace table dans lequel sont stockées les valeurs contenues dans les colonnes longues des tables de métadonnées (telles que celles contenant les types de données LONG VARCHAR et LOB). Si vous indiquez une valeur NULL pour n'importe quelle partie de la spécification de l'espace table, l'espace table par défaut pour cette partie est utilisé.

Si vous indiquez une valeur nulle pour n'importe quelle partie de la spécification de l'espace table, l'espace table par défaut pour cette partie est utilisé.

Produit EEE uniquement : L'espace table spécifié doit appartenir au même groupe de noeuds que la table utilisateur.

Exemples

Activation de la table Employés pour le stockage de données Image :

```
enable table Employés for db2image
   using mydataspace, myindxspace, mylongspace
```

Activation de la table Employés pour le stockage de données Image. Utilisation des espaces table par défaut :

```
enable table Employés for db2image
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Si aucun espace table n'est précisé, le système utilise l'espace table défini lors de l'activation de la base de données en cours.

GET EXTENDER STATUS

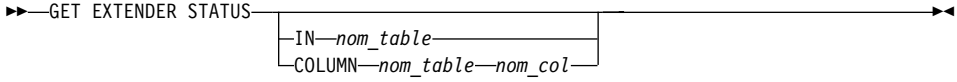
Image	Audio	Vidéo
X	X	X

Affiche le nom des extensions, s'il y en a, pour lesquelles une colonne, une table ou la base de données en cours est activée.

Classe d'autorisation

Aucune

Syntaxe de la commande



Paramètres de la commande

nom_table
Nom de la table dans la base de données en cours.

nom_col
Nom de la colonne.

Exemples

Affichage du nom des extensions activées dans la base de données :

```
get extender status
```

Affichage de l'état de la table Employés :

```
get extender status in Employés
```

Affichage de l'état de la colonne ADRESSE dans la table Employés :

```
get extender status column Employés adresse
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

GET INACCESSIBLE FILES

Image	Audio	Vidéo
X	X	X

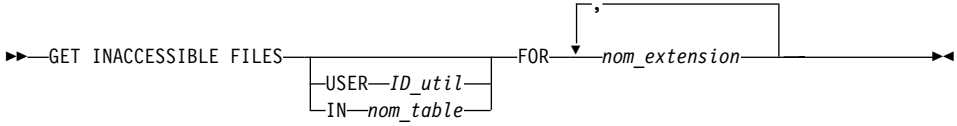
Dresse la liste de tous les fichiers de support inaccessibles auxquels il est fait référence dans une table, dans plusieurs tables possédant un qualificatif spécifique ou dans toutes les tables de la base de données en cours.

Classe d'autorisation

Pour toutes les tables de la base de données en cours, c'est-à-dire, si vous ne spécifiez pas USER ou IN : SYSADM, SYSCTRL, SYSMAINT, DBADM.

Pour une table particulière (si vous indiquez IN) ou des tables possédant un qualificatif (si vous indiquez USER) : Select.

Syntaxe de la commande



Paramètres de la commande

ID_util
Qualificatif des tables de la base de données en cours dont vous souhaitez établir la liste des fichiers inaccessibles.

nom_table
Nom de la table de la base de données en cours dont vous souhaitez établir la liste des fichiers inaccessibles.

nom_extension
Nom de l'extension. Les noms d'extension corrects sont db2image, db2audio et db2video.

Exemples

Liste de tous les fichiers d'images auxquels il est fait référence dans la base de données, mais qui sont inaccessibles :

```
get inaccessible files
   FOR db2image
```

Liste de tous les fichiers d'images auxquels il est fait référence dans les tables sous le qualificatif anne, mais qui sont inaccessibles :

```
get inaccessible files
   user anne FOR db2image
```

Liste de tous les fichiers d'images auxquels il est fait référence dans les entrées de la table Employés, mais qui sont inaccessibles :

```
get inaccessible files
   in Employés FOR db2image
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Si vous indiquez une table, la commande dresse la liste des fichiers inaccessibles de cette dernière. Si vous indiquez un qualificatif, la commande dresse la liste des fichiers inaccessibles des tables portant ce qualificatif. Si vous ne précisez rien, la commande dresse la liste des fichiers inaccessibles de toutes les tables de la base de données en cours.

GET QBIC CATALOG INFO

Image	Audio	Vidéo
X		

Renvoie les informations suivantes concernant le catalogue actuellement ouvert :

- le nom de la table utilisateur et de la colonne image associées au catalogue ;
- le nom des caractéristiques figurant dans le catalogue ;
- le nombre de caractéristiques contenues dans le catalogue ;
- l'état d'activation de la fonction d'analyse automatique.

Classe d'autorisation

Select, Control, SYSADM, DBADM

Syntaxe de la commande

Paramètres de la commande

Aucune.

Exemples

Extraction d'informations sur le catalogue QBIC ouvert :

```
get qbic catalog info
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Le catalogue doit être ouvert.

GET REFERENCED FILES

Image	Audio	Vidéo
X	X	X

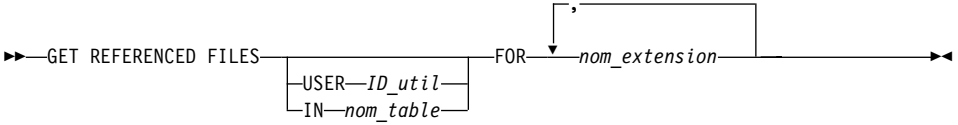
Dresse la liste de tous les fichiers de support et des noms de colonnes qui y font référence dans une table, dans plusieurs tables possédant un qualificatif particulier ou dans toutes les tables de la base de données en cours.

Classe d'autorisation

Pour toutes les tables de la base de données en cours, c'est-à-dire, si vous ne spécifiez pas USER ou IN : SYSADM, SYSCTRL, SYSMAINT, DBADM

Pour une table particulière (si vous indiquez IN) ou des tables possédant un qualificatif (si vous indiquez USER) : Select.

Syntaxe de la commande



Paramètres de la commande

ID_util
Qualificatif des tables de la base de données dont vous souhaitez établir la liste des fichiers référencés. La commande n'effectue de recherches que dans les tables possédant ce qualificatif.

nom_table
Nom de la table de la base de données en cours dont vous souhaitez établir la liste des fichiers référencés. La commande n'effectue de recherches que dans cette table.

nom_extension
Nom de l'extension. Les noms d'extension corrects sont db2image, db2audio et db2video.

Exemples

Liste de tous les fichiers d'images auxquels il est fait référence dans les entrées de toutes les tables de la base de données :

```
get referenced files
   FOR db2image
```

GET REFERENCED FILES

Liste de tous les fichiers d'images auxquels il est fait référence dans les tables sous le qualificatif anne :

```
get referenced files
   user anne FOR db2image
```

Liste de tous les fichiers d'images auxquels il est fait référence dans les entrées de la table Employés :

```
get referenced files
   in Employés for db2image
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Si vous ne précisez rien, la commande dresse la liste de toutes les tables de la base de données.

GET SERVER STATUS

Image	Audio	Vidéo
X	X	X

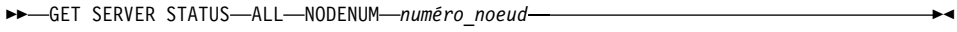
Affiche l'état du serveur d'extensions pour la base de données en cours ou pour l'ensemble des bases de données.

Produit EEE uniquement : Si un noeud est mentionné, la commande affiche l'état du serveur d'extensions—pour la base de données en cours ou pour toutes les bases de données—sur ce noeud uniquement.

Classe d'autorisation

Aucune

Syntaxe de la commande



Paramètres de la commande

ALL Affiche l'état de toutes les bases de données.

numéro_noeud
Numéro du noeud. La commande affiche l'état de ce noeud. **(Produit EEE uniquement)**

Exemples

Affichage de l'état du serveur d'extensions de la base de données en cours :

```
get server status
```

Affichage de l'état du serveur d'extensions de toutes les bases de données :

```
get server status all
```

Affichage de l'état du serveur d'extensions pour le noeud numéro 2 pour toutes les bases de données :

```
get server status all nodenum 2
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Si vous ne précisez aucun paramètre, la commande affiche l'état de tous les noeuds pour la base de données en cours répertoriés dans le fichier db2nodes.cfg.

OPEN QBIC CATALOG

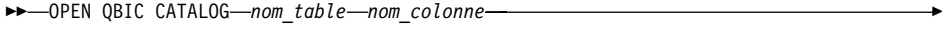
Image	Audio	Vidéo
X		

Ouvre le catalogue correspondant à la colonne DB2IMAGE indiquée. La base de données poursuit ses tentatives d'ouverture du catalogue en mode de mise à jour. Si le catalogue est déjà dans ce mode, il est alors ouvert en mode de lecture.

Classe d'autorisation

Connect

Syntaxe de la commande



Paramètres de la commande

nom_table
Nom de la table DB2IMAGE activée.

nom_colonne
Nom de la colonne DB2IMAGE activée.

Exemples

Ouverture du catalogue QBIC correspondant à la colonne photo dans la table Employés :

```
open qbic catalog Employés photo
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Cette commande provoque la fermeture de tout catalogue ouvert.

QUIT

Image	Audio	Vidéo
X	X	X

Ferme l'interpréteur de commandes db2ext pour l'entrée de commandes en mode interactif. La connexion avec DB2 persiste afin que vous puissiez envoyer des commandes à l'interpréteur de commandes db2ext en mode commande.

Classe d'autorisation

Aucune

Syntaxe de la commande

Paramètres de la commande

Aucune.

Exemples

Fermeture de l'interface de l'interpréteur de commandes pour le mode interactif :

```
quit
```

Remarques

La commande QUIT maintient la connexion à la base de données ouverte.

RECONNECT SERVER AT NODENUM (Produit EEE uniquement)

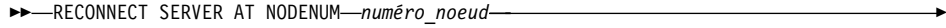
Image	Audio	Vidéo
X	X	X

Reconnecte le serveur au noeud indiqué sur toutes les bases de données.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT, DBADM

Syntaxe de la commande



Paramètres de la commande

numéro_noeud
Noeud que vous voulez reconnecter au serveur.

Exemples

Reconnexion du serveur à toutes les bases de données au noeud numéro 2 :

```
reconnect server at nodenum 2
```

Remarques

La commande DMBSTART vous permet de reconnecter le serveur de toutes les bases de données sur tous les noeuds.

RECONNECT SERVER FOR DATABASE (Produit EEE uniquement)

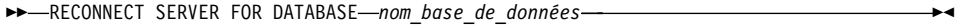
Image	Audio	Vidéo
X	X	X

Reconnecte le serveur à tous les noeuds de la base de données indiquée.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT, DBADM

Syntaxe de la commande



Paramètres de la commande

nom_base_de_données
Base de données que vous voulez reconnecter au serveur.

Exemples

Reconnexion du serveur à la base de données MY_DATABASE :

```
disconnect server for database my_database
```

Remarques

La commande DMBSTART vous permet de reconnecter le serveur à toutes les bases de données sur tous les noeuds.

RECONNECT SERVER FOR DATABASE AT NODENUM (Produit EEE uniquement)

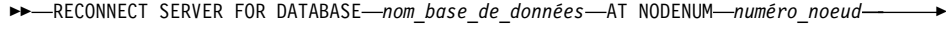
Image	Audio	Vidéo
X	X	X

Reconnecte le serveur à la base de données indiquée au noeud indiqué.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT, DBADM

Syntaxe de la commande



Paramètres de la commande

nom_base_de_données
Base de données que vous voulez reconnecter au serveur.

numéro_noeud
Noeud que vous voulez reconnecter au serveur.

Exemples

Reconnexion du serveur à la base de données MY_DATABASE au noeud numéro 2 :

```
reconnect server for database my_database at nodenum 2
```

Remarques

La commande DMBSTART vous permet de reconnecter le serveur à toutes les bases de données sur tous les noeuds.

REDISTRIBUTE NODEGROUP (Produit EEE uniquement)

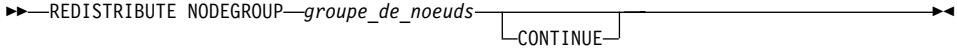
Image	Audio	Vidéo
X		

Redistribue les données d'extensions lors de l'ajout ou de la suppression d'un noeud dans un groupe de noeuds ou lors de l'établissement d'un nouveau plan de partition pour le groupe de noeuds.

Classe d'autorisation

SYSADM, DBADM

Syntaxe de la commande



Paramètres de la commande

groupe de noeuds
Nom du groupe de noeuds que vous voulez redistribuer.

CONTINUE
Si le processus de redistribution renvoie une erreur, vous pouvez relancer la commande avec ou sans le paramètre CONTINUE, selon les instructions fournies par la réponse de la commande. Cette option permet au système de reprendre le traitement là où il s'était arrêté plutôt que de tout recommencer dès le début.

Exemples

Redistribution du groupe de noeuds my_nodegroup :

```
redistribute nodegroup my_nodegroup
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Le paramètre CONTINUE ne doit pas être utilisé lors du premier lancement de la commande REDISTRIBUTE NODEGROUP après l'exécution de la commande DB2 REDISTRIBUTE. S'il est utilisé lors de ce premier lancement, une erreur est consignée et la redistribution reprend depuis le début.

Pour conserver l'intégrité des données, procédez à la redistribution d'un groupe de noeuds à la fois. Attendez qu'un groupe de noeuds ait terminé la redistribution avant d'en démarrer un autre.

Si la commande REDISTRIBUTE NODEGROUP échoue, reportez-vous au fichier ″redist.log″ pour une explication détaillée dans l'un des répertoires suivants :

- **Unix :** /<home-instance>/dmb/redist
- **Windows :** \\<instance_owning_machine>\DB2<instance_name> \<instance_name>\dmb\redist

REMOVE QBIC FEATURE

Image	Audio	Vidéo
X		

Supprime la table correspondant à la caractéristique spécifiée du catalogue ouvert.

Classe d'autorisation

Alter, Control, SYSADM, DBADM

Syntaxe de la commande

Paramètres de la commande

nom_caractéristique

Nom de la caractéristique supprimée du catalogue QBIC. Les fonctions suivantes sont fournies avec l'extension Image :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Exemples

Suppression de la caractéristique QbColorFeatureClass du catalogue ouvert :

```
remove qbic feature qbcolorfeatureclass
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

Le catalogue doit être ouvert.

REORG

Image	Audio	Vidéo
X	X	X

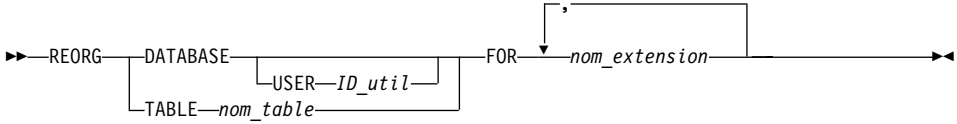
"Nettoie" les tables de gestion (table de gestion et table d'attributs) associées à une table spécifique, à plusieurs tables comportant un qualificatif spécifique ou à toutes les tables de la base de données en cours.

Classe d'autorisation

Pour une table spécifique (si vous exécutez REORG TABLE), ou pour des tables comportant un qualificatif spécifique (si vous exécutez REORG DATABASE) : SYSADM, SYSCTRL, SYSMAINT, DBADM, Control.

Pour toutes les tables de la base de données (si vous exécutez REORG DATABASE) : SYSADM, SYSCTRL, SYSMAINT, DBADM.

Syntaxe de la commande



Paramètres de la commande

ID_util
Qualificatif des tables.

nom_table
Nom de la table de la base de données en cours dont vous souhaitez nettoyer les tables de gestion.

nom_extension
Nom de l'extension. Les noms d'extension corrects sont db2image, db2audio et db2video.

Exemples

Réorganisation et nettoyage des tables de gestion des images de la base de données en cours :

```
reorg database for db2image
```

Réorganisation et nettoyage des tables de gestion des images de toutes les tables possédant le qualificatif anne :

```
reorg database user anne for db2image
```

Réorganisation et nettoyage des tables de gestion des images de la table Employés :

```
reorg table Employés for db2image
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

SET QBIC AUTOCATALOG

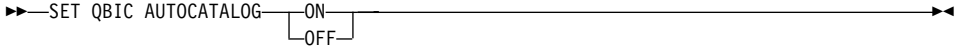
Image	Audio	Vidéo
X		

Catalogue automatiquement les images importées dans une colonne. Les images sont ajoutées au catalogue QBIC associé à la colonne.

Classe d'autorisation

Alter, Control, SYSADM, DBADM

Syntaxe de la commande



Paramètres de la commande

Aucune.

Exemples

Activation du catalogage automatique :

```
set qbic autocatalog on
```

Remarques

Le catalogue QBIC doit être ouvert.

START SERVER (Produit non EEE uniquement)

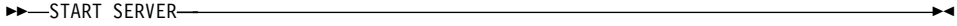
Image	Audio	Vidéo
X	X	X

Démarre le serveur d'extensions pour la base de données en cours.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT, DBADM

Syntaxe de la commande



Paramètres de la commande

Aucune.

Exemples

Démarrage du serveur d'extensions pour la base de données en cours :

```
start server
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

STOP SERVER (Produit non EEE uniquement)

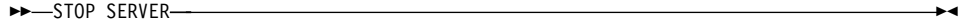
Image	Audio	Vidéo
X	X	X

Arrête le serveur d'extensions pour la base de données en cours.

Classe d'autorisation

SYSADM, SYSCTRL, SYSMAINT, DBADM

Syntaxe de la commande



Paramètres de la commande

Aucune.

Exemples

Arrêt du serveur d'extensions pour la base de données en cours :

```
stop server
```

Remarques

Connectez-vous à la base de données avant d'utiliser cette commande.

TERMINATE

Image	Audio	Vidéo
X	X	X

Ferme l'interpréteur de commandes db2ext et interrompt la connexion à DB2.

Classe d'autorisation

Aucune

Syntaxe de la commande

Paramètres de la commande

Aucune.

Exemples

Fermeture de l'interpréteur de commandes db2ext :

```
quit
```

Remarques

TERMINATE interrompt la connexion à la base de données.

Chapitre 18. Commandes d'administration du poste serveur

Les commandes présentées dans ce chapitre s'exécutent à partir de la ligne de commande du système d'exploitation du serveur. Ne les exécutez pas à partir de la ligne de commande DB2 ou de l'interpréteur de commandes db2ext. Exécutez la commande DMBSTART chaque fois que vous arrêtez puis redémarrez le système serveur.

Produit EEE uniquement : Vous pouvez également émettre les commandes de serveur DMBSTART et DMBSTOP dans un environnement de bases de données multipartition. Lors de l'émission d'une commande de serveur dans un environnement de bases de données multipartition, la commande s'applique à tous les noeuds, sauf si vous incluez un numéro de noeud, auquel cas la commande ne s'applique qu'au noeud indiqué.

Produit EEE uniquement : Il n'est pas possible d'exécuter la commande DMBSTAT dans un environnement multipartition. La commande GET STATUS ALL permet de vérifier l'état du serveur dans un environnement multipartition.

DMBICRT

Image	Audio	Vidéo
X	X	X

Crée une instance DB2 Extensions. Vous devez en créer plusieurs si vous avez créé plusieurs instances du serveur DB2. Sous UNIX, vous créez une instance client lorsque vous installez le client DB2 Extensions ; la création de l'instance client configure votre environnement pour permettre l'utilisation de DB2 Extensions.

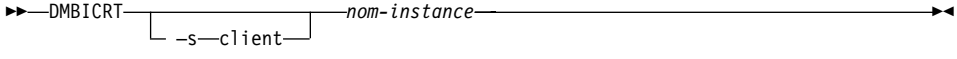
Classe d'autorisation

SYSADM

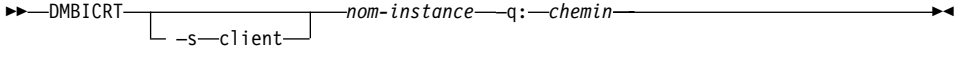
Sous Unix, vous devez disposer des droits root (superutilisateur).

Syntaxe de la commande

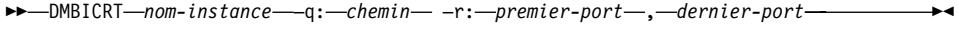
Dans un environnement de bases de données non partitionnées :



Dans un environnement de bases de données partitionnées sous UNIX :



Dans un environnement de bases de données partitionnées sous Windows :



Paramètres de la commande

nom-instance Nom d'une instance DB2 existante. S'il n'existe aucune instance DB2 de ce nom, un message vous propose de la créer.

–s client Crée une instance client uniquement. Lorsque vous utilisez ce paramètre, *nom-instance* correspond à l'ID utilisateur du client. La création d'une instance client configure l'environnement pour le client.**(UNIX uniquement)**

chemin Nom d'un répertoire ou d'un système de fichiers partagé ; le répertoire doit exister sur tous les noeuds. Il est défini dans la

variable d'environnement DB2MMDATAPATH sous UNIX et dans le registre sous Windows. **(Produit EEE uniquement)**

premier-port, dernier-port
Intervalle de ports TCP/IP utilisables. Il doit être supérieur ou égal au nombre de noeuds avec lesquels vous travaillez. Les numéros de port sont définis au niveau du registre Windows. **(Produit EEE pour Windows uniquement)**

Exemples

Création d'une instance de serveur DB2 Extensions pour l'instance DB2 DEVINST dans un environnement de bases de données non partitionnées :

```
dmbicrt  devinst
```

Remarques

La commande DMBICRT crée un répertoire d'instance DB2 Extensions pour les fichiers utilisés par l'instance. Ce répertoire s'intitule :

- *répertoire-installation*\INSTANCE*nom-instance*, où *répertoire-installation* désigne le répertoire dans lequel vous avez installé DB2 Extensions **(Windows, OS/2)**
- *INSTHOME*/dmb, où *INSTHOME* est le répertoire personnel du propriétaire de l'instance **(UNIX)**

S'il n'existe aucune instance DB2 portant le nom spécifié avec la commande DMBICRT, un message vous propose de la créer.

Produit EEE uniquement :

Bien que DMBICRT puisse s'exécuter à partir d'un ID utilisateur de n'importe quel noeud impliqué, il est recommandé de créer toutes les instances du serveur DB2 Extensions à l'aide du même noeud. Il doit s'agir du noeud utilisé pour créer l'instance DB2 et sur lequel se trouve le répertoire d'instance DB2. Si vous utilisez un autre noeud pour créer l'instance de serveur DB2 Extensions, la liste des instances stockées sur les noeuds risque de ne pas être complète.

Le répertoire partagé ou le système de fichiers spécifié en tant que *chemin* est sauvegardé comme valeur de la variable d'environnement DB2MMDATAPATH dans *$INSTHOME*/dmb/dmbprofile sous UNIX, et dans la clé de registre sous Windows :

```
\\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\DB2 Extenders\PROFILES
      \nom-instance\DB2MMDATAPATH
```

Sous UNIX, l'intervalle de ports autorisés doit être ajouté au fichier /etc/services avant que l'instance ne soit créée. Ajoutez les deux entrées concernées en respectant la syntaxe suivante :

– DMB_*nom-instance premier-port*

– DMB_*nom-instance*_END *dernier-port*

L'intervalle doit être assez large pour accueillir tous les noeuds contenus dans l'environnement de bases de données partitionnées.

Vous devez obligatoirement créer l'instance DB2 avant de créer une instance de serveur DB2 Extensions.

Pour créer une instance Extension Texte dans un environnement de bases de données partitionnées, utilisez la commande TXTICRT, comme décrit dans le manuel *DB2 UDB Extension Texte - Administration et programmation*.

DMBIDROP

Image	Audio	Vidéo
X	X	X

Supprime une instance DB2 Extensions.

Classe d'autorisation

SYSADM

Sous UNIX, vous devez disposer des droits root (superutilisateur).

Syntaxe de la commande

Paramètres de la commande

nom-instance Nom de l'instance DB2 Extensions à supprimer.

Exemples

Supprimez une instance de serveur DB2 Extensions intitulée DEVINST :

```
dmbidrop  devinst
```

Remarques

Avant d'exécuter cette commande :

- Arrêtez toutes les applications qui utilisent l'instance ainsi que tous les interpréteurs de commandes db2ext.
- Arrêtez les fonctions d'extension.

La commande DMBIDROP supprime le répertoire d'instance DB2 Extensions, l'entrée correspondante dans la liste des instances, ainsi que toutes les informations relatives à l'instance.

La commande DMBIDROP ne supprime que l'instance DB2 Extensions ; elle est sans effet sur l'instance DB2 qui lui est associée. Vous devez supprimer explicitement l'instance DB2.

Si vous supprimez l'instance DB2 associée à une instance de serveur DB2 Extensions, celle-ci n'est pas supprimée. Cependant, vous ne pouvez plus l'utiliser.

(Produit EEE uniquement) Si vous supprimez une instance DB2 Extensions, les entrées de port définies pour celle-ci doivent également être supprimées du fichier `/etc/services` sous UNIX, et du fichier

\WINNT\system32\drivers\etc\Services sous Windows. Ces entrées s'intitulent DMB_*nom-instance*mpp et DMB_*nom-instance*mpp_END.

DMBILIST

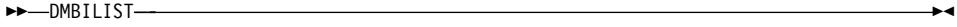
Image	Audio	Vidéo
X	X	X

Répertorie toutes les instances DB2 Extensions.

Classe d'autorisation

Aucune.

Syntaxe de la commande



Paramètres de la commande

Aucun.

Exemples

Liste des instances DB2 Extensions :

```
dmbilist
```

DMBIMIGR

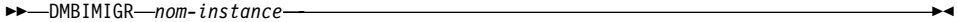
Image	Audio	Vidéo
X	X	X

(UNIX uniquement) Fait migrer une instance DB2 Extensions d'une ancienne version vers la version en cours.

Classe d'autorisation

Vous devez disposer des droits root (superutilisateur).

Syntaxe de la commande



Paramètres de la commande

nom-instance Nom de l'instance DB2 Extensions à faire migrer.

Exemples

Migration d'une instance DB2 Extensions intitulée OLDINST :

```
dmbimigr  oldinst
```

Remarques

Avant d'exécuter cette commande :

- Vous devez avoir installé la version actuelle de DB2 Extensions.
- Vous devez opérer la migration de l'instance DB2 associée.

Exécutez la commande DMBIMIGR pour chaque instance DB2 Extensions. Utilisez DMBILIST pour répertorier les instances.

DMBSTART

Image	Audio	Vidéo
X	X	X

Démarre toutes les fonctions Extensions de l'instance d'extension.

Produit EEE uniquement : Si un noeud est indiqué, la commande lance les fonctions Extension sur ce noeud uniquement. DMBSTART initie également la fonction de création/suppression de noeud sur chaque noeud. Cette fonction permet de vérifier que les noeuds définis pour DB2 correspondent aux noeuds définis pour les extensions. Si tel n'est pas le cas, la fonction de création/suppression de noeud ajoute ou supprime des noeuds, le cas échéant.

Classe d'autorisation

SYSADM

Syntaxe de la commande

Paramètres de la commande

numéro-noeud
Noeud sur lequel vous voulez démarrer les fonctions Extension. **(Produit EEE uniquement)**

Exemples

Démarrage des fonctions Extensions :

```
dmbstart
```

Démarrage des fonctions Extensions au noeud numéro 2 :

```
dmbstart nodenum 2
```

Remarques

Exécutez cette commande :

- Après vous être connecté en tant que propriétaire d'instance (sous AIX, HP-UX ou Solaris).
- A partir d'une fenêtre dans laquelle la variable d'environnement DB2INSTANCE est identique à l'instance à démarrer (sous OS/2 ou Windows).
- Chaque fois que vous arrêtez puis redémarrez le système serveur.

Dans un environnement monopartition, DMBSTART démarre également l'instance DB2, si celle-ci n'est pas en cours.

Produit EEE uniquement :

Dans un environnement multipartition, DMBSTART ne démarre pas l'instance DB2. Vous devez lancer DB2 avant d'exécuter DMBSTART dans un environnement partitionné.

Si DMBSTART échoue, procédez aux vérifications suivantes :

- Assurez-vous que la valeur de la variable DBD2MMDATAPATH est correcte.
- Vérifiez que le répertoire partagé ou le système de fichiers dans la variable existe et qu'il est accessible sur tous les noeuds.

DMBSTAT

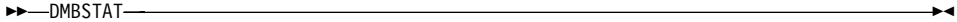
Image	Audio	Vidéo
X	X	X

Affiche les bases de données activées et indique si les fonctions Extensions pour ces bases sont en cours d'exécution.

Classe d'autorisation

Aucune

Syntaxe de la commande



Paramètres de la commande

Aucun.

Exemples

Affichage de l'état des fonctions Extensions :

```
dmbstat
```

DMBSTOP

Image	Audio	Vidéo
X	X	X

Arrête les fonctions Extensions pour l'instance d'extension.

Produit EEE uniquement : Si un noeud est indiqué, DMBSTOP arrête les fonctions Extension uniquement sur ce noeud.

Classe d'autorisation

SYSADM

Syntaxe de la commande

Paramètres de la commande

numéro-noeud
Noeud sur lequel vous voulez arrêter les fonctions Extension. **(Produit EEE uniquement)**

Exemples

Arrêt des fonctions Extensions :

```
dmbstop
```

Arrêt des fonctions Extension au noeud numéro 2 :

```
dmbstop nodenum 2
```

Remarques

Exécutez cette commande :

- Après vous être connecté en tant que propriétaire d'instance (sous AIX, HP-UX ou Solaris).
- A partir d'une fenêtre dans laquelle la variable d'environnement DB2INSTANCE est identique à l'instance à arrêter (sous OS/2 ou Windows).

DMBSTOP n'arrête pas l'instance DB2.

Produit EEE uniquement : N'exécutez pas la commande DMBSTOP sur un noeud particulier tant que votre base de données ne fonctionne pas en mode maintenance. De plus, vous devez vous assurer qu'aucune activité d'extensions n'est déclenchée sur ce noeud tant qu'il est désactivé. Dans le cas

contraire, des erreurs imprévues risquent de se produire.

Chapitre 19. Informations de diagnostic

Toutes les instructions SQL imbriquées ou appels DB2 CLI de votre programme, y compris celles qui appellent les fonctions UDF de DB2 Extensions, génèrent des codes indiquant si l'instruction SQL imbriquée ou l'appel DB2 CLI a abouti ou non. D'autres API de DB2 Extensions, telles que les API d'administration, peuvent également renvoyer des codes indiquant si l'opération a abouti ou non. Votre programme doit vérifier et réagir à ces codes retour.

Votre programme peut également obtenir des informations complémentaires sur ces codes, à savoir des codes SQLSTATE et des messages d'erreur. Vous pouvez utiliser ces informations de diagnostic pour identifier les incidents et y remédier.

Il peut arriver que la cause d'un incident soit difficile à identifier. Dans ce cas, vous devrez fournir certaines informations au personnel de maintenance pour faciliter le diagnostic de l'incident. DB2 Extensions comprend une fonction de trace qui enregistre les activités du logiciel. Ces données de trace sont très utiles au personnel de maintenance ; utilisez la fonction de trace uniquement à sa demande.

Le présent chapitre indique comment accéder aux informations de diagnostic, notamment :

- comment interpréter les codes retour des fonctions UDF et des API de DB2 Extensions ;
- comment gérer la fonction de trace.

Il répertorie et décrit également les codes SQLSTATE et les messages d'erreur susceptibles d'être renvoyés par les extensions.

Codes retour des fonctions UDF

Les instructions SQL imbriquées renvoient des codes dans les zones SQLCODE, SQLWARN et SQLSTATE de la structure SQLCA. Cette structure est définie dans un fichier d'inclusion SQLCA. (Pour plus de détails sur la structure SQLCA et le fichier d'inclusion du même nom, reportez-vous au manuel *DB2 Application Development Guide*.)

Les appels DB2 CLI renvoient les valeurs SQLCODE et SQLSTATE que vous pouvez extraire par la fonction SQLError. (Pour plus de détails sur cette opération, reportez-vous au manuel *CLI Guide and Reference*.)

Codes des fonctions UDF

Une valeur SQLCODE définie à 0 indique que l'instruction s'est exécutée avec succès (avec d'éventuelles conditions d'avertissement). Une valeur SQLCODE positive indique que l'instruction a été exécutée correctement avec un avertissement. (Les instructions SQL imbriquées renvoient l'avertissement associé à une valeur SQLCODE 0 ou positive dans la zone SQLWARN.) Une valeur SQLCODE négative signale un incident.

DB2 associe un message à chaque valeur SQLCODE. Si une fonction UDF de DB2 Extensions détecte un avertissement ou une erreur, elle transmet les informations correspondantes à DB2 pour qu'il les intègre dans le message SQLCODE.

Les valeurs SQLSTATE contiennent des codes complémentaires des messages SQLCODE. Pour plus de détails sur les codes SQLSTATE renvoyés par DB2 Extensions, reportez-vous à la section «Codes SQLSTATE» à la page 545.

Les instructions SQL imbriquées et les appels DB2 CLI qui appellent des fonctions UDF de DB2 Extensions peuvent renvoyer des messages SQLCODE et des valeurs SQLSTATE propres à ces fonctions UDF, mais DB2 les renvoie de la même manière que pour les autres instructions SQL imbriquées ou les autres appels DB2 CLI. Ainsi, la méthode d'accès à ces valeurs est la même que pour toutes les instructions SQL ou appels DB2 CLI.

Pour plus de détails sur les valeurs SQLSTATE et les numéros de message associés aux extensions, reportez-vous à la section «Codes SQLSTATE» à la page 545 . Pour obtenir une description des messages, reportez-vous à la section «Messages» à la page 549.

Codes retour des API

Chaque appel d'une API de DB2 Extensions renvoie un code. Un code retour 0 indique que l'appel de l'API a abouti. Un code retour différent de 0 indique que l'appel de l'API a été traité mais avec avertissement ou n'a pu être traité suite à un incident.

Le «Chapitre 16. Interfaces de programmation d'applications» à la page 279, répertorie les valeurs symboliques et décrit les codes renvoyés par les API de DB2 Extensions.

Il est possible d'obtenir des informations supplémentaires sur les erreurs renvoyées par une API. Utilisez pour cela l'API DB*x*GetError, où *x* correspond à a pour l'extension Audio, à i pour l'extension Image et à v pour l'extension Vidéo. L'API DB*x*GetError renvoie le code d'erreur SQL et le message de la dernière API de DB2 Extensions qui a rencontré une erreur. Pour plus de détails sur les codes d'erreur SQL, reportez-vous au manuel *DB2 - Guide des*

messages. Pour plus de détails sur les messages renvoyés par l'API DB*x*GetError, reportez-vous à la section «Messages» à la page 549.

Par exemple, les instructions suivantes d'un programme d'application en C activent une table pour l'extension Audio et recherchent d'éventuelles erreurs.

```
rc=DBaEnableTable((char *)NULL, "Employés");

rc = DBaGetError(&errCode, &errMsg);
```

Codes SQLSTATE

Le tableau 16 répertorie et décrit les valeurs SQLSTATE pouvant être renvoyées par DB2 Extensions. La description de ces valeurs inclut leur représentation symbolique. Le tableau indique également le numéro de message associé à chaque valeur SQLSTATE. Pour obtenir une description des messages, reportez-vous à la section «Messages» à la page 549.

Tableau 16. Codes SQLSTATE et numéros de message correspondants

SQLSTATE	N° du message	Description
00000		MMDB_SQLSTATE_OK L'opération a abouti
01H01	DMB0211W	MMDB_SQLSTATE_WARN_NO_OVERWRITE Le contenu du fichier n'a pas été écrasé
38A00	DMB0101E	MMDB_SQLSTATE_AUDIO_NULL_PARM Le paramètre d'entrée d'une fonction UDF ne peut pas être défini par NULL
38A02	DMB0209E	MMDB_SQLSTATE_AUDIO_OPEN_HDR_ERROR Incident à l'ouverture de l'en-tête d'un fichier audio
38A03	DMB0209E	MMDB_SQLSTATE_AUDIO_BAD_WAVE_HDR Fichier wave incorrect
38V00	DMB0101E	MMDB_SQLSTATE_VIDEO_NULL_PARM Le paramètre d'entrée d'une fonction UDF ne peut pas être défini par NULL
38V02	DMB0051E	MMDB_SQLSTATE_VIDEO_OPEN_HDR_ERROR Incident à l'ouverture de l'en-tête d'un fichier vidéo
38V03	DMB0105E	MMDB_SQLSTATE_VIDEO_BAD_MPEG1_HDR Fichier mpeg1 incorrect
38V04	DMB0104E	MMDB_SQLSTATE_VIDEO_BLOB_TOO_SHORT Mémoire tampon vidéo insuffisante
38V05	DMB0106E	MMDB_SQLSTATE_VIDEO_BAD_AVI_HDR Fichier AVI incorrect
38V07	DMB0106E	MMDB_SQLSTATE_VIDEO_BAD_QT_HDR Fichier Quicktime incorrect

Tableau 16. Codes SQLSTATE et numéros de message correspondants (suite)

SQLSTATE	N° du message	Description
38600	DMB0075E DMB0101E DMB0102E DMB0103E DMB0210E	MMDB_SQLSTATE_INVALID_INPUT Paramètre d'entrée de la fonction UDF incorrect
38601	DMB0009E	MMDB_SQLSTATE_MALLOC_FAIL Echec de l'allocation de mémoire
38602	DMB0386E	MMDB_SQLSTATE_CANNOT_COLLOCATE Contiguïté impossible avec les données utilisateur
38603	DMB0077E	MMDB_SQLSTATE_READ_FILE_FAIL Impossible de lire le fichier
38604	DMB0080E	MMDB_SQLSTATE_WRITE_FILE_FAIL Impossible d'écrire dans le fichier
38610	DMB0070E	MMDB_SQLSTATE_INVALID_HANDLE La colonne de support contient des données incorrectes
38611	DMB0073E	MMDB_SQLSTATE_INVALID_SESSION_HANDLE Descripteur de session UDF incorrect
38612	DMB0074E	MMDB_SQLSTATE_INVALID_STATEMENT_HANDLE Descripteur d'instruction UDF incorrect
38613	DMB0083E	MMDB_SQLSTATE_INVALID_IMPORT_REQUEST Demande d'importation incorrecte
38615	DMB0071E	MMDB_SQLSTATE_CONNECT_FAIL Incident lors de la connexion à la base de données
38617	DMB0071E	MMDB_SQLSTATE_ALLOC_STMT_FAIL Incident lors de l'allocation d'un nouveau descripteur d'instruction
38618	DMB0208E DMB0138E	MMDB_SQLSTATE_FREE_STMT_FAIL Incident à la libération de l'instruction
38619	DMB0208E DMB0132E	MMDB_SQLSTATE_BIND_FAIL Incident lors de la définition des accès
38620	DMB0208E	MMDB_SQLSTATE_BIND_COLUMN_FAIL Incident lors de la définition des accès à une colonne
38621	DMB0208E	MMDB_SQLSTATE_BIND_FILE_FAIL Incident lors de la définition des accès à un fichier
38622	DMB0208E DMB0132E	MMDB_SQLSTATE_SET_PARAM_FAIL Incident lors de la définition d'un paramètre
38623	DMB0208E DMB0131E	MMDB_SQLSTATE_PREPARE_FAIL Incident lors de la préparation d'une instruction SQL (par PREPARE)

Tableau 16. Codes SQLSTATE et numéros de message correspondants (suite)

SQLSTATE	N° du message	Description
38624	DMB0208E DMB0133E DMB0172E	MMDB_SQLSTATE_EXECUTE_FAIL Incident pendant l'exécution d'une instruction
38625	DMB0208E DMB0133E	MMDB_SQLSTATE_EXEC_DIRECT_FAIL Incident pendant l'exécution directe de l'instruction SQL d'une fonction UDF
38626	DMB0208E DMB0133E	MMDB_SQLSTATE_FETCH_FAIL Incident pendant l'extraction de la ligne de données suivante
38627	DMB0208E	MMDB_SQLSTATE_COMMIT_FAIL Incident lors de la validation de la transaction
38628	DMB0208E	MMDB_SQLSTATE_GET_LENGTH_FAIL Incident lors de l'extraction de la longueur d'une valeur de type chaîne
38629	DMB0208E	MMDB_SQLSTATE_GET_SUBSTRING_FAIL Incident lors de l'extraction d'une partie d'une valeur de type chaîne
38650	DMB0077E DMB0079E	MMDB_SQLSTATE_COPY_BLOB_2_FILE_FAIL Incident lors de la copie d'un objet BLOB dans un fichier
38651	DMB0086E	MMDB_SQLSTATE_BLOB_BUFFER_TOO_SMALL Mémoire tampon insuffisante
38652	DMB0082E	MMDB_SQLSTATE_BUILD_HANDLE Incident lors de la création des données de la colonne de support
38653	DMB0083E	MMDB_SQLSTATE_INVALID_INSERT_VIA_SELECT Demande d'insertion à l'aide de SELECT interdite
38654	DMB0081E	MMDB_SQLSTATE_INVALID_OFFSET_SIZE Valeur de décalage incorrecte
38655	DMB0068E	MMDB_SQLSTATE_METATABLE_DOESNOT_EXIST Table de métadonnées inexistante
38670	DMB0134E DMB0103E	MMDB_SQLSTATE_UNKNOWN_FORMAT Format du support stocké inconnu
38671	DMB0135E	MMDB_SQLSTATE_CREATE_THUMBNAIL_FAIL Incident lors de la création de la miniature
38672	DMB0114E	MMDB_SQLSTATE_FORMAT_CONVERSION_FAIL Incident lors de la conversion du format de fichier
38673	DMB0363E	MMDB_SQLSTATE_INVALID_UPDATE Incident lors de l'appel d'une fonction UDF de mise à jour sans désignation d'une table

Tableau 16. Codes SQLSTATE et numéros de message correspondants (suite)

SQLSTATE	N° du message	Description
38674	DMB0361E	MMDB_SQLSTATE_NOT_ENABLED Incident lors de l'exécution d'une fonction UDF d'importation sur une colonne qui n'est pas activée pour l'extension
38675	DMB0129E	MMDB_SQLSTATE_VIDEO_INTERNAL Erreur interne : fonctions UDF de l'extension Vidéo
38676	DMB0129E	MMDB_SQLSTATE_AUDIO_INTERNAL Erreur interne : fonctions UDF de l'extension Audio
38677	DMB0129E	MMDB_SQLSTATE_IMAGE_INTERNAL Erreur interne : fonctions UDF de l'extension Image
38678	DMB0089E DMB0208E	MMDB_SQLSTATE_BASE_INTERNAL_ERROR Erreur interne au niveau de la couche commune
38681	DMB0108E	MMDB_SQLSTATE_IMPORT_ENV_NOT_SETUP Définition incorrecte de la variable d'environnement pour l'importation
38682	DMB0111E	MMDB_SQLSTATE_STORE_ENV_NOT_SETUP Définition incorrecte de la variable d'environnement pour le stockage
38683	DMB0107E	MMDB_SQLSTATE_EXPORT_ENV_NOT_SETUP Définition incorrecte de la variable d'environnement pour l'exportation
38684	DMB0117E	MMDB_SQLSTATE_TEMP_ENV_NOT_SETUP Définition incorrecte de la variable d'environnement pour la création de fichiers
38686	DMB0109E	MMDB_SQLSTATE_CANT_RESOLVE_IMPORT_FILE Incident pendant la résolution du nom du fichier d'importation
38687	DMB0112E	MMDB_SQLSTATE_CANT_RESOLVE_STORE_FILE Incident pendant la résolution du nom du fichier de sauvegarde
38688	DMB0110E	MMDB_SQLSTATE_CANT_RESOLVE_EXPORT_FILE Incident pendant la résolution du nom du fichier d'exportation
38689	DMB0116E	MMDB_SQLSTATE_CANT_CREATE_TMP_FILE Incident lors de la création d'un fichier temporaire
38690	DMB0076E	MMDB_SQLSTATE_OPEN_IMPORT_FILE_FAIL Impossible d'ouvrir le fichier d'importation
38691	DMB0115E	MMDB_SQLSTATE_OPEN_STORE_FILE_FAIL Impossible d'ouvrir le fichier d'importation
38692	DMB0114E	MMDB_SQLSTATE_OPEN_EXPORT_FILE_FAIL Impossible d'ouvrir le fichier d'exportation
38693	DMB0118E	MMDB_SQLSTATE_OPEN_TEMP_FILE_FAIL Impossible d'ouvrir le fichier temporaire

Tableau 16. Codes SQLSTATE et numéros de message correspondants (suite)

SQLSTATE	N° du message	Description
38694	DMB0117E	MMDB_SQLSTATE_OPEN_CONTENT_FILE_FAIL Impossible d'ouvrir le fichier du contenu
38695	DMB0135E	MMDB_SQLSTATE_OPEN_THUMBNAIL_FILE_FAIL Impossible d'ouvrir le fichier de la miniature
38696	DMB0135E	MMDB_SQLSTATE_READ_THUMBNAIL_FILE_FAIL Impossible de lire le fichier de la miniature
38697	DMB0207E	MMDB_SQLSTATE_OVERWRITE_NOT_ALLOWED Impossible d'écraser le fichier
38699	DMB0171E	MMDB_SQLSTATE_QUERY_NAME_NOT_FOUND Aucune requête de ce nom n'a été trouvée
38700		MMDB_SQLSTATE_NO_MANAGEBLOB
38701		MMDB_SQLSTATE_UDFLOCATOR_FAIL
38702		MMDB_SQLSTATE_SQL_FAIL
38703		MMDB_SQLSTATE_INVALID_UPDATE
38704		MMDB_SQLSTATE_NOT_ENABLED
38705	DMB0366E DMB0382E	MMDB_SQLSTATE_QBIC_QUERY_FAIL_TO_BUILD Un incident s'est produit lors de la création de la requête
38706	DMB0205E	MMDB_SQLSTATE_QBIC_TABLE_COLUMN_PAIR_NOT_VALID Un incident s'est produit lors de la tentative d'accès à un catalogue QBIC. Soit le descripteur d'image ne se trouve pas dans le catalogue, soit la combinaison nom de table-nom de colonne n'est pas associée à un catalogue.
38707	DMB0383E	MMDB_SQLSTATE_QBIC_QUERY_EXECUTE_FAILED Un incident s'est produit lors de l'exécution de la requête
38708		MMDB_SQLSTATE_QBIC_UKNOWN_ERROR Un incident non identifié s'est produit dans QBIC
38709	DMB0208E	MMDB_COPY_FILE_TO_LOCATOR_FAILURE Un incident s'est produit lors de la copie d'un fichier vers un releveur de coordonnées LOB
38710	DMB0534E	MMDB_SQLSTATE_QBIC_UNSUPPORTED_UDF La fonction UDF n'est pas prise en charge.

Messages

DMB0001E Le serveur DB2 Extensions n'était pas connecté. Code anomalie : "<code>".

Cause: Pour effectuer cette opération, les services DB2 Extensions doivent être actifs.

Action: A partir du serveur, lancez la commande DMBSTART à l'invite système.

DMB0003W La fonction de trace de DB2 Extensions est en cours d'exécution dans cette session.

Cause: La fonction de trace fait appel aux ressources système.

Action: Si les performances de votre système sont affectées, désactivez la fonction de trace.

DMB0004I Ce programme ne peut être exécuté que par le propriétaire de l'instance : "<nom>".

Cause: Les serveurs DB2 Extensions doivent être démarrés avec l'ID utilisateur sous lequel l'instance a été créée.

Action: Lancez la commande DMBSTART sous l'ID de l'utilisateur qui a créé l'instance.

DMB0005E La base de données en cours n'est pas activée pour l'extension "<nom>".

Cause: Vous avez tenté une opération qui exige que la base de données soit activée pour une extension DB2 particulière. Par exemple, si vous voulez activer une table pour des données DB2IMAGE, vous devez d'abord activer la base de données dans laquelle se trouve la table pour l'extension en question.

Action: Activez la base de données pour le type de données de l'extension voulue et recommencez.

DMB0006E L'utilisateur "<nom>" n'est pas autorisé à appeler cette API.

Cause: Vous avez tenté d'appeler une API sous un ID utilisateur qui ne possède pas les droits d'accès requis pour cette API.

Action: Lancez l'application sous un ID utilisateur différent ou demandez à l'administrateur de base de données de modifier les droits de l'ID en cours.

DMB0007E La table utilisateur "<nom>" n'est pas activée pour l'extension "<nom>".

Cause: Vous avez tenté une opération sur une table qui n'est pas activée pour cette extension DB2. Par exemple, vous devez activer une table pour les données audio avant d'activer une colonne de cette table pour ce type de données.

Action: Assurez-vous d'abord que la table est activée pour cette extension. Puis activez la colonne.

DMB0008E Une erreur s'est produite au cours de l'exécution de la procédure mémorisée "<nom>".

Cause: Il s'est produit une erreur dans la procédure mémorisée indiquée dans le message ou un incident avec l'environnement.

Action: Vérifiez l'application et recommencez.

DMB0009E Erreur d'allocation de mémoire.

Cause: Le système n'a pas pu allouer la mémoire nécessaire pour l'opération demandée.

Action: Assurez-vous que le système dispose de suffisamment de mémoire pour effectuer l'opération.

DMB0010E L'extension "<nom>" a déjà été définie pour le type UDT "<nom>".

Cause: Ce nom de type UDT (défini par l'utilisateur) sert déjà pour un type UDT dans une autre extension DB2.

Action: Choisissez un autre nom pour votre type UDT.

DMB0011E La colonne utilisateur ″<nom>″ ne peut pas être activée pour l'extension ″<nom>″. La définition de la colonne utilisateur est incompatible avec le type distinct ″MMDBSYS.<nom>″ associé au programme d'extension.

Cause: La colonne indiquée n'étant pas définie pour le type de données mentionné dans le message, elle ne peut pas être activée pour cette extension.

Action: Assurez-vous que la colonne à activer a été définie pour le type de données correspondant à cette extension.

DMB0012E La table utilisateur indiquée ″<nom>″ n'existe pas.

Cause: Il n'existe pas de table de ce nom.

Action: Vérifiez le nom et l'existence de la table.

DMB0013E La colonne ″<nom>″ n'est pas définie dans la table ″<nom>″.

Cause: L'opération demandée cite un nom de colonne qui n'existe pas dans la table indiquée.

Action: Vérifiez les noms de la table et de la colonne.

DMB0014W La colonne ″<nom>″ de la table utilisateur ″<nom>″ est déjà activée pour l'extension ″<nom>″.

Cause: Vous avez tenté d'activer la colonne pour une extension pour laquelle elle était déjà activée.

Action: Aucune.

DMB0015W La base de données est déjà activée pour l'extension <nom>″.

Cause: Vous avez tenté d'activer une base de données pour une extension pour laquelle elle était déjà activée.

Action: Aucune.

DMB0016W La table utilisateur ″<nom>″ est déjà activée pour l'extension ″<nom>″.

Cause: Vous avez tenté d'activer une table pour une extension pour laquelle elle était déjà activée.

Action: Aucune.

DMB0017E La table utilisateur ″<nom>″ est déjà activée pour l'extension ″<nom>″. Cependant, au moins une des tables de métadonnées associées, ″<nom>″ ou ″<nom>″ n'existe pas.

Cause: Une ou plusieurs des tables de gestion (métadonnées) associées à cette table est endommagée ou détruite. Sans les tables de métadonnées, la table utilisateur ne peut servir pour les données de cette extension.

Action: Désactivez la table utilisateur, puis réactivez-la pour l'extension.

DMB0018E Le système ne peut pas créer un nom de déclencheur unique pour la colonne ″<nom>″ de la table ″<nom>″.

Cause: Lors de l'activation d'une colonne de la table utilisateur, il s'est produit une erreur pendant la création des déclencheurs utilisés par DB2 Extensions.

Action: Recommencez l'opération. Si l'incident persiste, contactez l'administrateur de base de données, puis le personnel de maintenance IBM.

DMB0019I ″<nombre>″ fichiers cités dans la table ″<nom>″ pour l'extension ″<nom>″.

Cause: Ce message indique le nombre de fichiers de support externes cités par une table utilisateur pour une extension donnée.

Action: Aucune.

DMB0020I **″<nombre>″ fichiers sont référencés dans les tables correspondant au schéma de table ″<nom>″ pour l'extension ″<nom>″.**

Cause: Ce message indique le nombre de fichiers de support externes cités par les tables utilisateur correspondant à un nom de schéma donné.

Action: Aucune.

DMB0021I **Il y a ″<nombre>″ fichiers inaccessibles référencés dans la table ″<nom>″ pour l'extension ″<nom>″.**

Cause: Ce message indique le nombre de fichiers de support externes cités par une table utilisateur pour une extension donnée et qui sont inaccessibles. Ces fichiers ont pu être supprimés.

Action: Aucune.

DMB0022I **Il y a ″<nombre>″ fichiers inaccessibles référencés par l'extension ″<nom>″.**

Cause: Ce message indique le nombre de fichiers de support externes :

- cités par les tables utilisateur de la base de données en cours,
- pour une extension déterminée (vidéo, par exemple),
- inaccessibles ; par exemple, les fichiers ont pu être supprimés.

Action: Aucune.

DMB0023I **Il y a ″<nombre>″ fichiers inaccessibles référencés dans les tables correspondant au schéma de table ″<nom>″ pour l'extension ″<nom>″.**

Cause: Ce message indique le nombre de fichiers de support externes cités par les tables utilisateur correspondant à un schéma donné et qui sont inaccessibles. Ces fichiers ont pu être supprimés. Le message indique également le nombre d'extensions pour lesquelles les tables sont activées.

Action: Aucune.

DMB0024I **La base de données en cours est activée pour ″<nombre>″ extensions.**

Cause: Ce message indique le nombre d'extensions DB2 pour lesquelles la base de données en cours est activée.

Action: Aucune.

DMB0025I **La table ″<nom>″ est activée pour ″<nombre>″ extensions.**

Cause: Ce message indique le nombre d'extensions DB2 pour lesquelles cette table est activée.

Action: Aucune.

DMB0026I **La colonne ″<nom>″ de la table ″<nom>″ est activée pour ″<nombre>″ extensions.**

Cause: Ce message indique le nombre d'extensions DB2 pour lesquelles cette colonne est activée.

Action: Aucune.

DMB0027I **La base de données en cours est activée pour l'extension ″<nom>″.**

Cause: Ce message indique l'extension DB2 pour laquelle la base de données en cours est activée.

Action: Aucune.

DMB0028I **La table ″<nom>″ est activée pour l'extension ″<nom>″.**

Cause: Ce message indique le type de données (support) que la table utilisateur peut contenir.

Action: Aucune.

DMB0029I La colonne ″<nom>″ de la table ″<nom>″ est activée pour l'extension ″<nom>″.

Cause: Ce message indique le type de données (support) que la colonne utilisateur peut contenir.

Action: Aucune.

DMB0030E La base de données en cours ne peut pas être activée pour l'extension ″<nom>″. RC = ″<code>″.

Cause: La base de données n'existe pas ou vous n'êtes pas autorisé à l'activer.

Action: Vérifiez que la base de données existe et que vous êtes autorisé à l'activer.

DMB0031E La table ne peut pas être activée pour l'extension ″<nom>″. RC = ″<code>″.

Cause: La base de données n'existe pas, la table n'est pas activée ou vous n'êtes pas autorisé à activer la table.

Action: Vérifiez que la base de données existe et que la base et la table sont activées pour l'extension. Assurez-vous que vous êtes autorisé à activer la table.

DMB0032E La colonne ne peut pas être activée pour l'extension ″<nom>″. RC = ″<code>″.

Cause: Cette colonne n'a pas été définie pour le type de données de cette extension, la colonne n'existe pas, la table n'est pas activée ou vous n'êtes pas autorisé à activer la colonne.

Action: Assurez-vous que la colonne est définie pour le bon type de données. Vérifiez que la table est activée et que vous êtes autorisé à activer la colonne.

DMB0033E Vous n'êtes pas autorisé à exécuter cette commande.

Cause: Votre ID utilisateur ne possède pas le niveau d'accès requis pour exécuter cette commande.

Action: Lancez la commande sous un autre ID ou demandez à l'administrateur de base de données de modifier les droits de l'ID en cours.

DMB0034I Le démarrage du serveur DB2 Extensions de la base de données ″<nom>″ a abouti.

Cause: Le serveur pour la base de données en cours a été démarré sans incident.

Action: Aucune.

DMB0035I Le serveur DB2 Extensions pour la base de données ″<nom>″ a été arrêté.

Cause: Le serveur pour la base de données en cours a été arrêté correctement.

Action: Aucune.

DMB0036E Le serveur DB2 Extensions n'a pas pu être démarré ou arrêté. Il est probable que le démon du serveur DB2 Extensions ne s'exécute pas. Prenez contact avec l'administrateur de base de données.

Cause: Il s'est produit un incident pendant le démarrage ou l'arrêt du serveur DB2 Extensions. Il est probable que le démon du serveur DB2 Extensions ne s'exécute pas.

Action: Prenez contact avec l'administrateur de base de données.

DMB0037E Le descripteur de session USE est incorrect.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se

reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0038E Le descripteur d'instruction USE est incorrect.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0039E Erreur USE : "<erreur>".

Cause: Il s'est produit une erreur interne.

Action: Suivez les instructions du message d'erreur correspondant et recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0040E Erreur SQL : "<erreur>"

Cause: Il s'est produit une erreur interne.

Action: Suivez les instructions du message d'erreur correspondant et recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0041W La base de données en cours est réactivée pour l'extension "<nom>" en utilisant le nouvel espace table indiqué.

Cause: Lors de l'activation précédente de la base de données en cours, elle utilisait un espace table différent. La base de données est à présent activée avec un nouvel espace table pour les tables de gestion.

Action: Aucune.

DMB0042E La colonne "<nom>" de la table "<nom>" n'est pas activée pour l'extension "<nom>".

Cause: Cette colonne n'est pas activée pour l'extension requise par l'opération demandée. Par exemple, vous avez tenté de désactiver une colonne qui n'était pas activée pour l'extension indiquée.

Action: Assurez-vous que la colonne est activée pour l'extension indiquée dans le message.

DMB0043I La base de données en cours est désactivée pour l'extension "<nom>".

Cause: La désactivation a abouti.

Action: Aucune.

DMB0044I La table "<nom>" est désactivée pour l'extension "<nom>".

Cause: La désactivation a abouti.

Action: Aucune.

DMB0045I La colonne "<nom>" de la table "<nom>" est désactivée pour l'extension "<nom>".

Cause: La désactivation a abouti.

Action: Aucune.

DMB0046E La base de données en cours ne peut pas être désactivée pour l'extension "<nom>". RC = "<code>".

Cause: La base de données n'existe pas, n'est pas activée pour l'extension ou vous n'êtes pas autorisé à la désactiver.

Action: Assurez-vous que la base de données existe et qu'elle est activée pour cette extension. Vérifiez que vous êtes autorisé à désactiver la base de données.

DMB0047E La table ne peut pas être désactivée pour l'extension "<nom>". RC = "<code>".

Cause: La table n'existe pas, n'est pas activée pour l'extension ou vous n'êtes pas autorisé à la désactiver.

Action: Assurez-vous que la table existe et qu'elle est activée pour cette extension. Vérifiez que vous êtes autorisé à désactiver la table.

DMB0048E La colonne ne peut pas être désactivée pour l'extension ″<nom>″. RC = ″<code>″

Cause: La colonne n'étant pas activée pour l'extension indiquée dans le message, elle ne peut pas être désactivée.

Action: Vérifiez le nom de l'extension et si la colonne est bien celle que vous voulez désactiver.

DMB0049E Vous n'êtes pas autorisé à exécuter cette commande.

Cause: Votre ID utilisateur ne possède pas le niveau d'accès requis pour exécuter cette commande.

Action: Lancez l'application sous un ID utilisateur différent ou demandez à l'administrateur de base de données de modifier les droits de l'ID en cours.

DMB0050E Vous ne disposez pas des droits ″<niveau>″ sur la table ″<table>″.

Cause: Votre ID utilisateur ne possède pas le niveau d'accès requis pour cette opération.

Action: Effectuez l'opération sous un ID possédant les droits différent ou demandez à l'administrateur de base de données de modifier les droits de l'ID en cours.

DMB0051E En-tête de fichier de support erroné.

Cause: Le système ne peut pas lire ou ouvrir l'en-tête du fichier de support. Le fichier est endommagé ou il ne s'agit pas d'un fichier de support.

Action: Vérifiez que le fichier est un fichier de support et qu'il n'est pas endommagé.

DMB0052I Le démarrage du serveur DB2 Extensions de ″<nom BD>″ a abouti.

Cause: Le serveur a été démarré correctement.

Action: Aucune.

DMB0053I L'arrêt du serveur DB2 Extensions de ″<nom BD>″ a abouti.

Cause: Le serveur a été arrêté sans incident.

Action: Aucune.

DMB0054E Le serveur DB2 Extensions ne peut pas se connecter à la base de données ou allouer un descripteur d'instruction DB2. Le serveur DB2 Extensions de la base de données ″<nom>″ est probablement arrêté.

Cause: Le serveur DB2 Extensions ne peut pas se connecter à la base de données ou allouer un descripteur d'instruction DB2. Il est probable que le serveur DB2 Extensions de la base de données n'est pas actif.

Action: Assurez-vous que le serveur DB2 Extensions de la base de données est actif. Si ce n'est pas le cas, lancez le serveur en question ou demandez à l'administrateur système de redémarrer les services DB2 Extensions.

DMB0055I L'exécution de la commande ″nom″ est terminée.

Cause: La commande a abouti.

Action: Aucune.

DMB0056E Une marque inattendue ″<marque>″ a été détectée après ″<motclé>″. Les marques attendues peuvent englober : <nomExtension>.

Cause: Le système attendait le nom d'une extension DB2 et non la marque indiquée dans le message.

Action: Vérifiez la syntaxe de la commande et recommencez.

DMB0057E L'espace table ″<nom>″ est incorrect.

Cause: L'espace table indiqué n'existe pas.

Action: Vérifiez le nom et l'existence de l'espace table.

DMB0058I ″<nombre>″ fichiers sont référencés par l'extension ″<nom>″.

Cause: Ce message indique le nombre de fichiers de support externes cités par une extension donnée.

Action: Aucune.

DMB0059E ″<nom>″ n'est pas un nom correct pour un programme DB2 Extensions. Les noms de programme Extension corrects sont ″<nom-extension,>″ DB2VIDEO, DB2AUDIO et DB2IMAGE.

Cause: Le nom de l'extension est mal orthographié.

Action: Vérifiez le nom de l'extension.

DMB0060E La syntaxe correcte pour ″<fonction>″ est : ″<syntaxe>.″

Cause: La syntaxe de la commande que vous avez saisie est erronée.

Action: Appliquez la syntaxe décrite dans le message.

DMB0061E Le nom de table ″<nom>″ qui suit ″<motclé>″ est incorrect.

Cause: La commande attend un nom de table.

Action: Vérifiez la syntaxe de la commande et recommencez.

DMB0062E Le nom de colonne ″<nom>″ qui suit ″<motclé>″ est incorrect.

Cause: La commande attend le nom d'une colonne.

Action: Vérifiez la syntaxe de la commande et recommencez.

DMB0064E Le système ne reconnaît pas la marque ″<marque>″ qui suit ″<motclé>″.

Cause: La commande n'attendait pas la marque indiquée dans le message.

Action: Vérifiez la syntaxe de la commande et recommencez.

DMB0065E L'ID utilisateur ″<id>″ qui suit ″<motclé>″ est incorrect.

Cause: La commande attend un ID utilisateur correct.

Action: Vérifiez l'ID que vous voulez utiliser et recommencez.

DMB0066E Le mot de passe ″<motdepasse>″ qui suit ″<motclé>″ est incorrect.

Cause: La commande attendait un ID utilisateur correct et non la marque indiquée dans le message.

Action: Vérifiez le mot de passe et recommencez.

DMB0067E La commande que vous avez entrée est incorrecte.

Cause: Le nom de la commande est mal orthographié ou la syntaxe est erronée.

Action: Vérifiez la syntaxe de la commande et recommencez.

DMB0068E Cette table de métadonnées n'existe pas.

Cause: La fonction a tenté d'utiliser une table de gestion (métadonnées) qui devrait exister

pour l'objet données. La table de métadonnées est peut-être endommagée ou supprimée.

Action: Vérifiez le nom et l'existence de la table de métadonnées. Si des tables de métadonnées ont été accidentellement supprimées ou endommagées, désactivez et réactivez l'objet données.

DMB0069E Le nom de base de données ″<nom>″ est incorrect.

Cause: Il n'existe pas de base de données de ce nom.

Action: Vérifiez le nom et l'existence de la base de données.

DMB0070E Descripteur incorrect.

Cause: La valeur de descripteur transmise à votre application est probablement endommagée.

Action: Vérifiez l'application pour vous assurer que les valeurs de descripteur de l'extension n'ont pas été changées.

DMB0071E La connexion à ″<nomBD>″ est impossible.

Cause: Il est possible que le serveur DB2 Extensions de la base de données ne soit pas démarré.

Action: Vérifiez l'état du serveur. Si le serveur n'est pas actif, relancez-le en entrant la commande START SERVER sur la ligne de commande DMB.

DMB0072E Le serveur UDF SQL ne peut pas déconnecter la base de données.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0073E Descripteur de session USE incorrect.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0074E Descripteur d'instruction USE incorrect.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0075E Indiquez un nom de fichier.

Cause: Cette opération exige un nom de fichier de support.

Action: Entrez le nom d'un fichier de support.

DMB0076E Impossible d'ouvrir le fichier d'importation.

Cause: Le fichier d'importation manque ou est endommagé.

Action: Vérifiez le nom et l'existence du fichier d'importation.

DMB0077E Impossible d'ouvrir ou de lire le fichier.

Cause: Le descripteur de l'extension pointe sur un fichier inexistant ou altéré. L'extension ne peut accéder au fichier.

Action: Recherchez le nom du fichier à l'aide de la fonction UDF FILENAME ou vérifiez l'existence du fichier.

DMB0078E Impossible de créer le fichier d'exportation.

Cause: Le fichier d'exportation manque ou est altéré.

Action: Vérifiez le nom et l'existence du fichier d'exportation.

DMB0079E Impossible de copier des données BLOB dans le fichier.

Cause: Le fichier ne peut pas accepter d'objet BLOB. Il se peut que l'espace soit insuffisant.

Action: Comparez la taille de l'objet BLOB à l'espace disponible et augmentez ce dernier, le cas échéant.

DMB0080E Impossible d'écrire dans le fichier.

Cause: Le fichier est endommagé, inexistant ou son nom est mal orthographié.

Action: Vérifiez le nom et l'existence du fichier.

DMB0081E Décalage ou taille incorrect.

Cause: L'opération n'a pas pu trouver les données attendues dans la structure. La taille de la zone ou le décalage est incorrect.

Action: Vérifiez le décalage et la taille de la zone.

DMB0082E Impossible de créer le descripteur.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0083E "<nom-extension>" et "<nom-extension>" sont incompatibles.

Cause: Les deux extensions mentionnées dans le message ne sont pas compatibles pour cette opération. L'insertion, par instruction FULL SELECT ou SELECT partielle, est interdite.

Action: Assurez-vous que l'objet cible est activé pour la même extension que l'objet source.

DMB0084E Demande d'importation incorrecte : nomfichier, contenu, type de mémoire.

Cause: L'importation a échoué. Le nom de fichier, le contenu ou le type de mémoire est incorrect.

Action: Vérifiez les données et recommencez.

DMB0085E Demande de mise à jour incorrecte : nomfichier, contenu, type de mémoire.

Cause: La mise à jour a échoué. Le nom de fichier, le contenu ou le type de mémoire est incorrect.

Action: Vérifiez les données et recommencez.

DMB0086E La taille demandée est trop élevée.

Cause: La taille demandée dépasse la taille maximale d'un objet BLOB pour la fonction UDF.

Action: Demandez une taille inférieure.

DMB0087E Nom de fichier incorrect.

Cause: Il n'existe pas de fichier de ce nom.

Action: Vérifiez le nom et l'existence du fichier.

DMB0088E Descripteur NULL.

Cause: La fonction UDF attend un descripteur différent de NULL.

Action: Assurez-vous que l'application reçoit un descripteur correct et le transmet à la fonction UDF.

DMB0089E Cette valeur de descripteur n'existe pas.

Cause: Le descripteur transmis à la fonction UDF est incorrect.

Action: Assurez-vous que l'application transmet un descripteur correct.

DMB0090E Données tronquées.

Cause: La taille du fichier ou de la mémoire tampon ne permet pas de recevoir les données.

Action: Augmentez la taille du fichier ou de la mémoire tampon.

DMB0091W Le fichier contient déjà ces données.

Cause: Le fichier contient déjà des données. Le contenu sera remplacé.

Action: Aucune.

DMB0092E L'opération d'insertion que vous avez tenté d'exécuter sur la colonne ″<nom>″ est incorrecte. Cette colonne est activée pour l'extension ″<nom>″.

Cause: Le type des données à insérer est différent de celui de l'extension pour laquelle la colonne est activée.

Action: Assurez-vous que l'objet cible est activé pour la même extension que l'objet source.

DMB0093E L'opération de mise à jour que vous avez tenté d'exécuter sur la colonne ″<nom>″ est incorrecte. Cette colonne est activée pour l'extension ″<nom>″.

Cause: Le type des données de mise à jour est différent de celui de l'extension pour laquelle la colonne est activée.

Action: Assurez-vous que l'objet cible est activé pour la même extension que l'objet source.

DMB0094I La table ″<nom>″ n'existe pas.

Cause: Le système n'a pu trouver de table de ce nom. Elle existe peut-être dans une autre base de données.

Action: Aucune.

DMB0095W La table ″<nom>″ n'est pas activée pour l'extension ″<nom>″.

Cause: Cette table n'est pas activée pour l'extension indiquée.

Action: Aucune.

DMB0096W La colonne ″<nom>″ de la table ″<nom>″ n'était pas activée pour l'extension ″<nom>″.

Cause: Le système attendait une colonne activée.

Action: Aucune.

DMB0097W La base de données en cours n'est pas activée pour l'extension ″<nom>″.

Cause: Le système attendait une base de données activée.

Action: Activez la base de données pour l'extension indiquée dans le message.

DMB0098E L'utilisateur ne dispose pas des droits ″<niveau>″ sur la table ″<nom>″.

Cause: Votre ID utilisateur ne possède pas le niveau d'accès requis pour cette opération.

Action: Effectuez l'opération sous l'ID utilisateur propriétaire de la table ou demandez à l'administrateur de base de données de modifier les droits de l'ID en cours.

DMB0099E Impossible de valider la transaction.

Cause: Il est possible que le serveur DB2 Extensions de la base de données en cours soit arrêté.

Action: Vérifiez l'état du serveur. Si le serveur n'est pas actif, relancez-le en entrant la commande START SERVER sur la ligne de commande db2ext.

DMB0100E ″<nom>″ n'est pas un nom de table correct.

Cause: Il n'existe pas de table de ce nom.

Action: Vérifiez le nom et l'existence de la table et recommencez.

DMB0101E Paramètre NULL incorrect.

Cause: La commande attendait un paramètre différent de NULL.

Action: Vérifiez la syntaxe et recommencez.

DMB0102E Type de stockage incorrect.

Cause: Dans DB2 Extensions, cette valeur indique l'emplacement de stockage des données du support.

Action: Indiquez 0 pour un emplacement externe (dans un fichier) et 1 pour un emplacement interne (dans la base de données).

DMB0103E Format non pris en charge.

Cause: DB2 Extensions ne prend pas en charge le format de cet objet.

Action: Convertissez l'objet en un format accepté.

DMB0104E La mémoire tampon vidéo est insuffisante.

Cause: La séquence vidéo est trop grande pour la mémoire tampon qui lui est allouée.

Action: Augmentez la taille de la mémoire tampon.

DMB0105E En-tête MPEG1 incorrect.

Cause: L'en-tête du fichier MPEG1 manque ou est altéré.

Action: Vérifiez qu'il s'agit bien d'un fichier MPEG1.

DMB0106E En-tête AVI incorrect.

Cause: L'en-tête du fichier AVI manque ou est altéré.

Action: Vérifiez qu'il s'agit bien d'un fichier AVI.

DMB0107E L'environnement d'exportation n'est pas défini.

Cause: Dans DB2 Extensions, les variables de l'environnement d'exportation ne sont pas définies correctement.

Action: Assurez-vous que les variables d'environnement sont définies correctement en vous reportant à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

DMB0108E L'environnement d'importation n'est pas défini.

Cause: Dans DB2 Extensions, les variables de l'environnement d'importation ne sont pas définies correctement.

Action: Assurez-vous que les variables d'environnement sont définies correctement en vous reportant à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

DMB0109E Impossible de résoudre le nom du fichier d'importation.

Cause: Il n'existe pas de fichier d'importation de ce nom.

Action: Vérifiez le nom et l'existence du fichier et assurez-vous que les variables d'environnement sont définies correctement en vous reportant à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

DMB0110E Impossible de résoudre le nom du fichier d'exportation.

Cause: Il n'existe pas de fichier d'exportation de ce nom.

Action: Vérifiez le nom et l'existence du fichier et assurez-vous que les variables d'environnement sont définies correctement en vous reportant à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

DMB0111E L'environnement de stockage n'est pas défini.

Cause: Les variables de l'environnement de stockage ne sont pas définies correctement.

Action: Assurez-vous que les variables d'environnement sont définies correctement en vous reportant à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

DMB0112E Impossible de résoudre le nom du fichier de sauvegarde.

Cause: Il n'existe pas de fichier de sauvegarde de ce nom.

Action: Vérifiez le nom et l'existence du fichier et assurez-vous que les variables d'environnement sont définies correctement en vous reportant à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

DMB0113E Impossible d'ouvrir le fichier d'importation.

Cause: Le fichier est peut-être verrouillé par un autre utilisateur, manque ou est altéré.

Action: Vérifiez le nom, l'existence et l'état du fichier, ainsi que vos droits d'accès.

DMB0114E Impossible d'ouvrir le fichier d'exportation.

Cause: Le fichier est peut-être verrouillé par un autre utilisateur, manque ou est altéré.

Action: Vérifiez le nom, l'existence et l'état du fichier, ainsi que vos droits d'accès.

DMB0115E Impossible d'ouvrir le fichier de sauvegarde.

Cause: Le système tente d'écrire dans un fichier qui existe déjà. Le serveur ne dispose pas des droits voulus pour remplacer le fichier existant.

Action: Vérifiez le nom, l'existence et l'état du fichier, ainsi que vos droits d'accès.

DMB0116E Impossible de créer le fichier temporaire.

Cause: Il se peut que l'espace soit insuffisant pour créer le fichier temporaire.

Action: Assurez-vous que les variables de l'environnement temporaire pour cette extension sont définies correctement. Augmentez l'espace, le cas échéant.

DMB0117E L'environnement temporaire n'est pas défini.

Cause: Les variables de l'environnement temporaire ne sont pas définies correctement.

Action: Assurez-vous que les variables d'environnement sont définies correctement en vous reportant à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

DMB0118E Impossible d'ouvrir le fichier temporaire

Cause: Le fichier temporaire a pu être écrasé ou endommagé.

Action: Assurez-vous que les variables d'environnement sont définies correctement en vous reportant à l'«Annexe A. Définition des variables d'environnement de DB2 Extensions» à la page 585.

DMB0119I Le serveur dmbsrv est en cours de lancement pour ″<nom>″ avec ″<nombre>″ connexions.

Cause: Ce message indique le nombre de connexions effectuées lors du démarrage du serveur.

Action: Aucune.

DMB0120E Le démarrage du serveur dmbsrv pour ″<nom>″ avec ″<nombre>″ connexions a échoué.

Cause: DB2 n'est peut-être pas encore démarré, la base de données n'existe pas ou le nombre de connexions autorisé par la licence de votre système est insuffisant.

Action: Assurez-vous que DB2 est lancé et que la base de données existe. Si l'incident persiste, contactez IBM pour obtenir davantage de licences.

DMB0121I Le serveur dmbsrv est démarré pour ″<nom>″ avec ″<nombre>″ connexions.

Cause: Ce message indique le nombre de connexions effectuées lors du démarrage du serveur.

Action: Aucune.

DMB0122I Le serveur dmbssd est prêt.

Cause: Le serveur est prêt à exécuter votre application.

Action: Aucune.

DMB0129E Opération incorrecte : ″<nom>″.

Cause: Il n'existe pas de commande ou d'API de ce nom.

Action: Vérifiez la commande ou l'API et recommencez.

DMB0130E La liaison entre la colonne ″<nom>″ et l'instruction SQL a échoué.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0131E L'instruction SQL PREPARE a échoué.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0132E L'instruction SQL SET a échoué.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0133E L'instruction SQL EXECUTE a échoué.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0134E La conversion de format de fichier a échoué.

Cause: La conversion du format des données multimédia stockées n'est pas prise en charge.

Action: Vous ne pouvez pas convertir le format de ce fichier.

DMB0135E Impossible d'ouvrir ou de lire la miniature.

Cause: Le fichier de la miniature manque ou est endommagé.

Action: Vérifiez le nom, l'existence et l'intégrité du fichier de la miniature.

DMB0136E Fichier de liens introuvable.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0137E Connexion à la base de données ″<nom>″ impossible.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0138E Impossible de libérer une instruction SQL.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0139E Le nom de fonction ″<nom>″ qui suit ″<motclé>″ est incorrect.

Cause: L'extension Image attend un nom de fonction correct pour cette commande.

Action: Relancez la commande avec un nom de fonction valable. Les noms de fonction autorisés sont :
- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

DMB0141E Le qualificatif ″<identificateur>″ qui suit ″<motclé>″ est incorrect.

Cause: Le système ne peut pas identifier le qualificatif dans la commande.

Action: Vérifiez le qualificatif et recommencez.

DMB0142E Aucun catalogue n'a été ouvert.

Cause: Dans DB2 Extensions, la commande ne cours nécessite l'ouverture d'un catalogue QBIC.

Action: Ouvrez le catalogue QBIC par de la commande OPEN QBIC CATALOG.

DMB0143I Dans le catalogue QBIC, pour la colonne ″<nom>″ de la table ″<nom>″, la valeur ″<état>″ a été affectée au catalogage automatique. Il y a ″<nombre>″ fonctions :

Cause: Ce message indique le nombre de fonctions défini dans le catalogue QBIC pour une colonne image donnée et si le catalogage automatique est activé.

Action: Aucune.

DMB0145E Le descripteur de requête est incorrect.

Cause: Le descripteur de requête utilisé dans l'appel d'API est incorrect.

Action: Vérifiez l'application pour déterminer si vous obtenez le bon descripteur de requête.

DMB0146E Le nom de classe de fonction ″<nomclasse>″ est incorrect.

Cause: Il n'existe pas de classe de fonction de ce nom. Les noms de fonction autorisés sont :
- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Action: Corrigez le nom de fonction et recommencez.

DMB0147E Le nom de classe de fonction ″<nomclasse>″ manque ou est incorrect.

Cause: Les noms de fonction autorisés sont :
- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Action: Corrigez le nom de fonction et recommencez.

DMB0148E La fonction ″<nom>″ est déjà un membre de la requête.

Cause: La requête contient déjà la fonction indiquée dans le message.

Action: Aucune.

DMB0149E La fonction ″<nom>″ n'est pas un membre de la requête.

Cause: La requête ne contient pas le nom de fonction indiqué.

Action: Pour ajouter la fonction à la requête avant d'appeler des API qui accèdent à la fonction, utilisez l'API QbQueryAddFeature.

DMB0150E Le système ne peut pas allouer de mémoire.

Cause: Le système n'a pas pu allouer la mémoire nécessaire pour l'opération demandée.

Action: Assurez-vous que le système dispose de suffisamment de mémoire pour effectuer l'opération.

DMB0151E Le pointeur sur la valeur de retour est NULL.

Cause: L'appel d'API n'a pas abouti car le pointeur sur une valeur de retour ne peut pas être défini par la valeur NULL.

Action: Assurez-vous que l'appel d'API reçoit des paramètres autorisés et que la syntaxe est correcte.

DMB0152E Le pointeur sur la valeur de retour de la liste est NULL.

Cause: L'appel d'API n'a pas abouti car le pointeur sur une valeur de retour ne peut pas être défini par la valeur NULL.

Action: Assurez-vous que l'appel d'API reçoit des paramètres autorisés et que la syntaxe est correcte.

DMB0153E Le paramètre de portée est réservé et doit avoir la valeur 0.

Cause: Ce paramètre est réservé à un usage ultérieur.

Action: Donnez à la portée la valeur 0.

DMB0154E Le pointeur sur le nom de classe de la fonction est incorrect.

Cause: L'appel d'API attendait un pointeur correct sur le nom de classe de la fonction d'entrée.

Action: Assurez-vous que l'appel d'API reçoit des paramètres autorisés et que la syntaxe est correcte.

DMB0155I Une taille de mémoire tampon égale à zéro a été transmise à la fonction ″<nom>″.

Cause: L'appel d'API a besoin de la mémoire tampon pour renvoyer des informations.

Action: Aucune.

DMB0156E Le pointeur QbImageSource est NULL.

Cause: La valeur NULL indique que la structure ne doit pas être modifiée.

Action: Aucune.

DMB0157E Le type QbImageSource ″<type>″ est incorrect.

Cause: Le type de données de la structure citée par cette API d'extension DB2 n'est pas valable.

Action: Le type de données de la structure doit être QbImageSource.

DMB0159E Le pointeur sur la mémoire tampon image QbImageSource est NULL.

Cause: L'appel d'API attendait un pointeur en retour.

Action: Vérifiez l'application pour déterminer si l'appel d'API et la mémoire tampon sont spécifiés correctement.

DMB0160I La longueur de la mémoire tampon image ou du fichier image est égale à zéro.

Cause: La longueur est nulle.

Action: Aucune.

DMB0161E Le pointeur sur le nom de table et/ou de colonne est NULL.

Cause: L'appel d'API attendait un pointeur.

Action: Vérifiez l'application pour déterminer si les données d'entrée de l'appel d'API sont spécifiées correctement.

DMB0162I Vous avez affecté la valeur zéro à requestedHits.

Cause: Avec requestedHits = 0, vous n'obtenez pas d'informations en retour.

Action: Aucune.

DMB0163I Cette fonction n'est pas encore prise en charge.

Cause: Cette fonction n'est pas encore prise en charge.

Action: Aucune.

DMB0164E Le système ne peut pas traiter la requête (<nom>).

Cause: Il s'est produit une erreur à la création de la requête.

Action: Vérifiez les données d'entrée de la commande ou de l'API et recommencez.

DMB0165E Le système ne peut pas exécuter la requête (<nom>).

Cause: Il s'est produit une erreur à la création de la requête.

Action: Vérifiez les données d'entrée de la commande ou de l'API et recommencez.

DMB0166E Une erreur d'instruction a été détectée dans ″<nom>″ au cours de l'exécution de ″<nom>″ : ″<erreur>″

Cause: Il s'est produit une erreur interne IBM.

Action: Prenez contact avec l'administrateur de base de données.

DMB0167E Une erreur s'est produite lors de la lecture de GenericImageDataClass (<erreur>).

Cause: Il s'est produit une erreur interne IBM.

Action: Prenez contact avec l'administrateur de base de données.

DMB0168E La fonction ″<nom>″ d'une requête n'a pas été définie avant la recherche.

Cause: La requête n'a pas été exécutée car elle ne comportait aucune fonction.

Action: Ajoutez une fonction à la requête à l'aide de l'API QbAddFeature ou de la commande ADD QBIC FEATURE.

DMB0169E L'erreur suivante s'est produite dans CLI : ″<erreur>″.

Cause: Erreur CLI.

Action: Suivez les instructions du message.

DMB0170E Le nom de requête ″<nom>″ est déjà en cours d'utilisation.

Cause: Il existe une autre requête de ce nom.

Action: Choisissez un nom différent.

DMB0171E Le nom de requête ″<nom>″ n'a pas été stocké.

Cause: Après avoir créé la requête, le système n'a pas pu la stocker.

Action: Assurez-vous que vous disposez de l'accès en écriture et de suffisamment d'espace pour stocker la requête.

DMB0172E Erreur SQL : ″<erreur>″

Cause: Il s'est produit une erreur interne.

Action: Suivez les instructions du message d'erreur correspondant et recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0173E Le catalogue est ouvert, mais en lecture seulement : ″<nomcatalogue>″.

Cause: Vous ne pouvez pas mettre à jour le catalogue car un autre utilisateur l'a ouvert en écriture ou vous ne disposez pas de l'accès en écriture.

Action: Attendez la fin de l'opération effectuée par l'autre utilisateur, lancez l'application sous un ID utilisateur différent ou demandez à l'administrateur de base de données de modifier les droits de votre ID en cours.

DMB0174E Une erreur système s'est produite : ″<erreur>″.

Cause: Il s'est produit une erreur interne IBM.

Action: Suivez les instructions du message d'erreur correspondant et recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0175I Images introuvables : ″<informations>″.

Cause: Aucune image correspondant à la requête n'a été trouvée. Il est possible que la base de données soit vide.

Action: Aucune.

DMB0176I La colonne est déjà associée à un catalogue QBIC : ″<nomtable nomcolonne>″.

Cause: Il existe déjà un catalogue de ce nom.

Action: Aucune.

DMB0177E Le système ne peut pas ouvrir le catalogue ; message d'erreur : ″<erreur>″.

Cause: Le catalogue est endommagé.

Action: Suivez les instructions du message.

DMB0178E Le système ne peut pas supprimer le catalogue ; message d'erreur : ″<erreur>″.

Cause: Le catalogue n'existe pas ou il est endommagé.

Action: Vérifiez le nom et l'existence du catalogue et recommencez.

DMB0179E Le descripteur de catalogue est incorrect : ″<erreur>″.

Cause: Le descripteur de catalogue utilisé dans l'appel d'API est incorrect.

Action: Vérifiez l'application pour déterminer si vous obtenez le bon descripteur de catalogue.

DMB0180I L'accès au catalogue est refusé : ″<erreur>″.

Cause: L'accès est refusé.

Action: Aucune.

DMB0181I Le catalogue est en cours d'utilisation ″<erreur>″.

Cause: Une autre opération utilise ce catalogue.

Action: Aucune.

DMB0184I La fonction de trace a déjà été lancée.

Cause: La fonction de trace est déjà active.

Action: Aucune.

DMB0185I La fonction de trace n'a pas encore été lancée.

Cause: La fonction de trace n'a pas encore été lancée.

Action: Aucune.

DMB0186I La fonction de trace a été activée à ″<heure>″ à partir du répertoire ″<nom>″. Le fichier de trace est ″<nomfich>″. ″<nombre> octets de données de trace ont été écrits.

Cause: La fonction de trace est active.

Action: Aucune.

DMB0187E La communication ne peut pas être établie car le système ne peut pas ouvrir le fichier ″<nomfich>″ en écriture.

Cause: Vous n'êtes pas le propriétaire de l'instance en cours décrite par la variable d'environnement DB2INSTANCE ou des variables d'environnement telles que DB2MMTOP ne sont pas définies correctement.

Action: Connectez-vous sous l'ID utilisateur propriétaire de l'instance. Vérifiez les variables d'environnement.

DMB0188I Une erreur s'est produite lors de la création du démon de trace : ″<erreur>″

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0189I Le lancement de la fonction de trace a abouti.

Cause: La fonction de trace est déjà active.

Action: Aucune.

DMB0190E La fonction de trace ne peut pas être lancée.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0191E La variable d'environnement ″<nom>″ doit être définie.

Cause: La configuration du système est incorrecte.

Action: Définissez la variable et recommencez.

DMB0192I La désactivation de la fonction de trace a abouti.

Cause: La fonction de trace a été désactivée.

Action: Aucune.

DMB0193E Le système ne peut pas écrire dans le fichier ″<nom>″.

Cause: Vous ne disposez pas de l'accès en écriture au répertoire contenant ce fichier.

Action: Prenez contact avec l'administrateur de base de données pour modifier vos droits d'accès.

DMB0194E Le système ne peut pas lire dans le fichier ″<nom>″.

Cause: Le fichier n'existe pas ou vous ne disposez pas de l'accès en lecture au fichier.

Action: Vérifiez que le fichier existe et que vous disposez de l'accès en lecture au fichier.

DMB0198E Le code de trace ″<code>″ du fichier d'entrée est inconnu. Le fichier d'entrée est peut-être endommagé.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0199E Vous ne disposez pas des droits ″<niveau>″ sur les tables référencées.

Cause: Votre ID utilisateur ne possède pas le niveau d'accès requis pour effectuer cette opération.

Action: Effectuez l'opération sous un autre ID utilisateur ou demandez à l'administrateur de base de données de modifier les droits de l'ID en cours.

DMB0200W Vous ne disposez pas des droits ″<niveau>″ sur une au moins des tables référencées.

Cause: Votre ID utilisateur ne possède pas les droits requis pour accéder à certaines tables.

Lorsque vous affichez des fichiers cités, ceux-ci correspondent à des tables pour lesquelles vous disposez des droits Select. S'il existe des tables pour lesquelles vous ne disposez pas de ces droits, les fichiers cités par ces tables ne s'affichent pas.

Si vous réorganisez des métadonnées, le système ne se préoccupe que des tables pour lesquelles vous disposez des droits Control.

Action: Pour accéder à tous les fichiers, effectuez l'opération sous un ID utilisateur différent ou demandez à l'administrateur de modifier les droits associés à votre ID en cours.

DMB0201I Il existe déjà une fonction portant ce nom : ″<nom>″.

Cause: Le catalogue QBIC contient déjà une fonction de ce nom.

Action: Aucune.

DMB0202E Le nom de fonction est incorrect : ″<nom>″.

Cause: Il n'existe pas de classe de fonction de ce nom. Les noms de fonction autorisés sont :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Action: Corrigez le nom de fonction et recommencez.

DMB0203E Fonction introuvable : ″<nom>″.

Cause: Il n'existe pas de classe de fonction de ce nom ou elle n'appartient pas au catalogue QBIC. Les noms de fonction autorisés sont :

- QbColorFeatureClass
- QbColorHistogramFeatureClass
- QbDrawFeatureClass
- QbTextureFeatureClass

Action: Corrigez le nom de fonction et recommencez.

DMB0204E La colonne n'est pas activée pour DB2IMAGE : ″<nom>″.

Cause: Cette colonne n'est pas activée pour l'extension Image.

Action: Assurez-vous que la colonne est activée pour l'extension Image de DB2.

DMB0205E Aucun catalogue n'a été trouvé pour ″<nomtable nomcolonne>″.

Cause: Il n'existe pas de catalogue QBIC associé à ce nom de colonne.

Action: Créez le catalogue QBIC pour la colonne avant de tenter d'autres opérations QBIC.

DMB0206W La colonne indiquée n'est activée pour aucune extension.

Cause: La colonne n'existe pas ou son type de données est incompatible avec les extensions.

Action: Assurez-vous que la colonne a été définie en utilisant un type de données correct.

DMB0207E Impossible de remplacer le fichier.

Cause: Ce fichier existe déjà mais la fonction UDF EXPORT ne peut pas le remplacer.

Action: Exportez le fichier sous un nom différent ou autorisez la fonction UDF EXPORT à le remplacer.

DMB0208E sqlcode=<code> clistate=<code>.

Cause: Il s'est produit une erreur interne.

Action: Recommencez l'opération. Si l'erreur se reproduit, prenez contact avec le personnel de maintenance IBM.

DMB0209E En-tête audio incorrect.

Cause: L'en-tête du fichier audio manque ou est altéré.

Action: Vérifiez si le format du fichier audio est pris en charge par DB2 Extensions.

DMB0211W Le fichier existe déjà sans remplacement.

Cause: Le fichier spécifié existe déjà et n'est pas écrasé.

Action: Aucune.

DMB0212E Le paramètre resultType est réservé et doit avoir pour valeur 0.

Cause: Ce paramètre est réservé à un usage ultérieur.

Action: Donnez à resultType la valeur 0.

DMB0214E Le pointeur sur le nom de requête est incorrect.

Cause: L'appel d'API attendait un pointeur correct sur le nom de requête indiqué.

Action: Assurez-vous que l'appel d'API reçoit des paramètres autorisés et que la syntaxe est correcte.

DMB0352E Environnement de ligne de commande non initialisé.

Cause: L'environnement de ligne de commande n'est pas initialisé pour exécuter l'interpréteur de commandes db2ext. (Ce message s'applique uniquement aux environnements Windows NT et Windows 95.)

Action: Lancez la commande db2cmd pour ouvrir une fenêtre DB2CLP, puis la commande db2ext pour lancer l'interpréteur de commandes db2 dans la fenêtre.

DMB0353E Impossible de communiquer avec le processus d'arrière-plan de l'interpréteur de commandes db2ext.

Cause: Le processus d'arrière-plan de l'interpréteur de commandes db2ext est actif, mais l'interpréteur deb2ext ne peut pas communiquer avec ce processus.

Action: Essayez de lancer la commande db2ext dans une autre fenêtre.

DMB0354E Impossible de démarrer le processus d'arrière-plan de l'interpréteur de commandes db2ext.

Cause: Le processus d'arrière-plan de l'interpréteur de commandes db2ext est actif, mais l'interpréteur deb2ext ne peut pas communiquer avec ce processus.

Action: Vérifiez que le module exécutable du processus d'arrière-plan (db2extb ou db2extb.exe) existe et que le répertoire correspondant est bien dans la variable d'environnement PATH.

DMB0355E Dépassement de délai au niveau du processus d'arrière-plan de l'interpréteur de commandes db2ext.

Cause: Le processus d'arrière-plan de l'interpréteur de commandes db2ext est démarré, mais l'interpréteur db2ext ne peut pas communiquer avec ce processus dans le délai imparti.

Action: Essayez de lancer la commande db2ext dans une autre fenêtre.

DMB0356E Impossible de communiquer avec le processus d'arrière-plan de l'interpréteur de commandes db2ext.

Cause: L'interpréteur de commandes db2ext a envoyé une demande au processus d'arrière-plan, mais celle-ci n'a pas été reçue.

Action: Assurez-vous que le processus d'arrière-plan de l'interpréteur de commandes db2ext est toujours en exécution.

DMB0357E Le processus d'arrière-plan de l'interpréteur de commandes db2ext ne répond pas.

Cause: L'interpréteur de commandes db2ext a envoyé une demande au processus d'arrière-plan, mais celui-ci n'a pas répondu dans le délai imparti.

Action: Assurez-vous que le processus d'arrière-plan de l'interpréteur de commandes db2ext est toujours en exécution.

DMB0359E La file d'attente de requêtes du processus d'arrière-plan de l'interpréteur de commandes db2ext ou la file d'attente d'entrée n'a pas été créée dans le délai imparti.

Cause: Le processus d'arrière-plan de l'interpréteur de commandes db2ext n'a pas créé les files d'attente de messages dans le délai imparti. (Ce message s'applique uniquement aux environnements UNIX.)

Action: Assurez-vous que le disque sur lequel réside le répertoire personnel de l'instance DB2 n'est pas saturé. En effet, le processus d'arrière-plan a besoin de ce répertoire pour créer un fichier destiné aux files d'attente de messages. Si le disque n'est pas saturé, vérifiez si vous n'avez pas lancé un trop grand nombre de processus db2extb. Ce risque existe si vous exécutez l'interpréteur de commandes db2ext dans plusieurs fenêtres. Un processus d'arrière-plan est démarré dans une fenêtre la première fois que vous émettez une requête pour l'interpréteur de commandes db2ext en mode commande. N'oubliez pas de lancer la commande `db2ext terminate` pour arrêter l'interpréteur de commandes db2ext lorsqu'il ne vous est plus utile. Les files d'attente de messages du processus expéditeur ne sont supprimées que si vous lancez la commande `terminate`.

DMB0361E Colonne ou table non activée.

Cause: Une fonction UDF d'importation a été spécifiée, mais la colonne de table indiquée n'est pas activée pour l'extension.

Action: Activez la colonne de table et recommencez.

DMB0363E Table et nom de colonne manquants

Cause: Une fonction UDF de mise à jour a été appelée, mais aucune table n'a été indiquée.

Action: Précisez une table et recommencez.

DMB0364E L'extension "<nom>" a déjà été définie pour l'espace table "<nom>".

Cause: La base de données, table ou colonne spécifiée a déjà été activée pour l'extension avec un espace table différent de celui qui est indiqué.

Action: Vérifiez que l'espace table spécifié est correct.

DMB0365E Privilège CONTROL non accordé sur "<table métadonnées>" et "<table métadonnées>" pour "<schéma>"."<table>".

Cause: Votre requête a été refusée car vous ne disposez pas du privilège CONTROL requis sur les tables de métadonnées de la table utilisateur indiquée.

Action: Demandez à l'administrateur de base de données de vous accorder le privilège CONTROL sur les tables de métadonnées.

DMB0366E Nom de fonction attendu.

Cause: Un nom de fonction est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0367E Couleur | histogramme | fichier attendu

Cause: «couleur», «histogramme» ou «fichier» est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0368E Signe ',' attendu.

Cause: Un signe ',' est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0369E Fichier incorrect.

Cause: Le fichier indiqué dans la chaîne de requête n'est pas correct.

Action: Corrigez la chaîne de requête et recommencez.

DMB0370E Nom de fichier attendu.

Cause: Un nom de fichier est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0371E Serveur | client attendu.

Cause: «serveur» ou «client» est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0372E Signe '(' attendu.

Cause: Un signe '(' est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0373E Signe ')' attendu.

Cause: Un signe ')' est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0374E Pourcentage attendu.

Cause: La valeur du pourcentage est attendue dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0375E Couleur attendue.

Cause: Les valeurs de couleur rouge, vert et bleu sont attendues dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0376E Signe '=' attendu.

Cause: Un signe '=' est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0377E Signe '<' attendu.

Cause: Un signe '<' est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0378E Signe '>' attendu.

Cause: Un signe '>' est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0379E 'AND' attendu.

Cause: Un opérateur 'AND' est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0380E Poids attendu.

Cause: Un poids est attendu dans la chaîne de requête.

Action: Corrigez la chaîne de requête et recommencez.

DMB0381E Fonction non définie.

Cause: La fonction n'a pas été ajoutée dans le catalogue QBIC.

Action: Ajoutez la fonction dans le catalogue QBIC, puis recataloguez les images.

DMB0382E Création de la requête impossible.

Cause: Il est possible que le serveur DB2 Extensions de la base de données en cours soit arrêté.

Action: Vérifiez l'état du serveur. Si le serveur n'est pas actif, relancez-le en entrant la commande START SERVER sur la ligne de commande db2ext.

DMB0383E Exécution de la requête impossible.

Cause: Il est possible que le serveur DB2 Extensions de la base de données en cours soit arrêté.

Action: Vérifiez l'état du serveur. Si le serveur n'est pas actif, relancez-le en entrant la commande START SERVER sur la ligne de commande db2ext.

DMB0384E Extraction de l'élément suivant impossible.

Cause: La fin de la liste est atteinte.

Action: Assurez-vous que votre application ne tente pas d'extraire un élément alors que la fin de la liste est atteinte.

DMB0386E Contiguïté impossible avec les données utilisateur

Cause: L'API SQL sqluihsh() a retourné un code différent de zéro.

Action: Recommencez l'opération. Si l'incident persiste, contactez IBM.

DMB0387E Le groupe de noeuds pour les espaces table indiqués est différent de celui de la table utilisateur.

Cause: Un ou plusieurs espaces table (pour la table de métadonnées, les index ou les objets BLOB) transmis en entrée pour l'activation d'une table sont définis sur un groupe de noeuds différent de celui sur lequel est définie la table utilisateur.

Action: Utilisez les espaces table définis sur le même groupe de noeuds que celui de la table utilisateur à activer.

DMB0388E Les espaces table de type REGULAR, LONG ou INDEX ne sont pas définis sur le même groupe de noeuds.

Cause: Un ou plusieurs espaces table (pour la table de métadonnées, les index ou les objets BLOB) transmis en entrée pour l'activation d'une base de données ne sont pas définis sur le même groupe de noeuds que celui des autres espaces table.

Action: Utilisez les espaces table définis sur le même groupe de noeuds.

DMB0389W Le groupe de noeuds pour les espaces table indiqués n'inclut pas tous les serveurs de partitions.

Cause: Les espaces table transmis en entrée sont définis sur un groupe de noeuds n'incluant pas l'ensemble des serveurs de partitions.

Action: Aucune. Toutefois, les fonctions UDF d'importation et de mise à jour seront plus performantes si les espaces table sont définis sur un groupe de noeuds couvrant l'ensemble des serveurs de partitions, surtout si les applications d'extension stockent le contenu des supports en format BLOB.

DMB0391I Cette commande ne peut s'exécuter que lorsqu'un client UDB DB2 accède à un serveur UDB DB2.

Cause: L'interpréteur de commandes db2ext n'est pas connecté à un serveur UDB DB2 ou n'a pas été lancé par un client UDB DB2. Par exemple, la commande START SERVER est correcte uniquement si l'interpréteur de commandes db2ext est connecté à un serveur DB2 autre que Extended Enterprise Edition.

Action: N'émettez pas cette commande dans la configuration client/serveur en cours.

DMB0392I Cette commande ne peut s'exécuter que lorsqu'un client UDB DB2 accède à un serveur UDB Extended Enterprise Edition DB2. Par exemple, la commande DISCONNECT SERVER est correcte uniquement si l'interpréteur de commandes db2ext est connecté à un serveur DB2 Extended Enterprise Edition.

Cause: L'interpréteur de commandes db2ext n'est pas connecté à un serveur UDB Extended Enterprise Edition DB2 ou n'a pas été lancé par un client UDB DB2.

Action: N'émettez pas cette commande dans la configuration client/serveur en cours.

DMB0402E L'option ″<nom-option>″ pour la commande ″<nom-commande>″ n'est correcte que si l'application est connectée à un serveur DB2 ″<type-serveur>″.

Cause: Le paramètre indiqué n'est pas correct car l'interpréteur de commandes db2ext n'est pas connecté au type de serveur prenant en charge cette option. Par exemple, la commande GET SERVER STATUS ne peut être indiquée avec le paramètre NODENUM <numnoeud> que si l'interpréteur de commandes db2ext est connecté à un serveur DB2 Extended Enterprise Edition.

Action: N'émettez pas cette combinaison commande-paramètre dans la configuration client/server en cours.

DMB0411E Port de base incorrect

Cause: Un numéro de port TCP/IP incorrect a été entré en tant que port de base lors de la création de l'instance.

Action: La syntaxe correcte est dmbicrt -r:port_base,port_fin -t:port_base,port_fin. Corrigez le paramètre et relancez la commande.

DMB0412E Port de fin incorrect

Cause: Un numéro de port TCP/IP incorrect a été entré en tant que port de fin lors de la création de l'instance.

Action: La syntaxe correcte est dmbicrt -r:port_base,port_fin -t:port_base,port_fin. Corrigez le paramètre et relancez la commande.

DMB0413E Impossible de résoudre le chemin d'installation des extensions DB2.

Cause: Le programme de création de l'instance n'a pas pu trouver de valeur pour la variable d'environnement ″DMBPATH.″

Action: Définissez la variable ″DMBPATH″ et relancez l'application.

DMB0414E Impossible de résoudre le nom hôte de l'ordinateur.

Cause: Un erreur interne s'est produite au cours de la tentative de résolution du nom de l'ordinateur.

Action: Prenez contact avec l'assistance IBM.

DMB0415E Impossible de résoudre le numéro de noeud pour cette machine.

Cause: La machine sur laquelle s'exécute la création de l'instance n'est pas répertoriée dans le fichier ″db2nodes.cfg.″

Action: Ajoutez la machine au fichier ″db2nodes.cfg″ et relancez l'application.

DMB0416E Ce programme doit être lancé par l'utilisateur root. Impossible de continuer.

Cause: L'ID utilisateur sous lequel s'exécute le programme ne dispose pas des droits root.

Action: Connectez-vous en tant qu'utilisateur root et relancez l'application.

DMB0417E Ce programme doit être exécuté par un utilisateur détenant les droits d'administrateur. Impossible de continuer.

Cause: L'ID utilisateur sous lequel s'exécute le programme ne dispose pas des droits d'administrateur.

Action: Connectez-vous avec un ID utilisateur disposant des droits d'administrateur et relancez l'application.

DMB0418E Impossible d'obtenir des informations sur l'utilisateur : ″<idutil>″.

Cause: Une erreur interne s'est produite lors de la tentative d'obtention des informations relatives à l'utilisateur associées à l'instance en cours de création.

Action: Vérifiez qu'il existe un ID utilisateur correct portant le même nom que l'instance en cours de création et relancez l'application.

DMB0419E Impossible de créer un répertoire d'extensions AIV ″<nom_répertoire>″. Code retour = <code>

Cause: Un erreur s'est produite lors de la tentative de création du répertoire indiqué. Le code retour représente l'erreur renvoyée par le système d'exploitation.

Action: Vérifiez que le système ou l'unité de fichiers indiqué dans le nom du répertoire existe et que les droits autorisent la création d'un répertoire.

DMB0420E Impossible de créer un lien pour le répertoire AIV Extenders ″<nom_répertoire>″. Code retour = <code>

Cause: Une erreur s'est produite lors de la tentative de création du lien symbolique indiqué. Le code retour représente l'erreur renvoyée par le système d'exploitation.

Action: Vérifiez que le système ou l'unité de

fichiers indiqué dans le nom du répertoire existe et que les droits autorisent la création d'un lien.

DMB0421E Impossible d'ouvrir le fichier : "<nom_fichier>". Code retour = <code>

Cause: Une erreur s'est produite lors de la tentative d'ouverture du fichier indiqué. Le code retour représente l'erreur renvoyée par le système d'exploitation.

Action: Vérifiez que le fichier existe et que les droits autorisent l'ouverture du fichier.

DMB0422E Impossible d'écrire dans le fichier : "<nom_fichier>". Code retour = <code>

Cause: Une erreur s'est produite lors de la tentative d'écriture dans le fichier indiqué. Le code retour représente l'erreur renvoyée par le système d'exploitation.

Action: Vérifiez que le fichier existe et que les droits autorisent l'écriture dans le fichier.

DMB0424E Impossible de trouver le fichier "db2nodes.cfg".

Cause: Le fichier DB2 "db2nodes.cfg" n'a pas pu être localisé.

Action: Vérifiez que la version correcte de DB2 UDB Extended Enterprise Edition a été installée et relancez l'application.

DMB0426E Erreur : "<code_erreur>" d'ouverture de la clé "<clé_registres>".

Cause: Une erreur s'est produite lors de la tentative d'ouverture de la clé de la base de registres indiquée.

Action: Notez le code retour et prenez contact avec l'assistance IBM.

DMB0427E La variable "<variable>" n'a pas été définie dans le registre de profils.

Cause: La valeur indiquée n'a pas été trouvée dans le registre Windows NT.

Action: Vérifiez que le nom d'une variable d'extension DB2 correcte a été indiqué.

DMB0430E Impossible de trouver les valeurs de registre DB2

Cause: Les valeurs de registre utilisées par DB2 sont introuvables.

Action: Vérifiez que la version correcte de DB2 UDB Extended Enterprise Edition a été installée et relancez l'application.

DMB0431E Impossible de créer la clé de la base de registres Extensions : "<clé_registre>".

Cause: Une erreur interne s'est produite lors de la tentative de création d'une clé de la base de registre Extensions.

Action: Prenez contact avec l'assistance IBM.

DMB0432E Impossible de définir une valeur pour la clé de la base de registre Extensions : "<clé_registres>".

Cause: Une erreur interne s'est produite lors de la tentative de définition de la valeur d'une clé de la base de registres Extensions.

Action: Prenez contact avec l'assistance IBM.

DMB0435E Impossible d'accéder au fichier de contrôle "<fichier_contrôle>".

Cause: Le fichier de contrôle indiqué n'a pas pu être localisé.

Action: Prenez contact avec l'assistance IBM.

DMB0443E Impossible d'ouvrir le répertoire "<nom_répertoire>". Retour = <code>

Cause: Une erreur s'est produite lors de la tentative d'ouverture du répertoire indiqué. Le code retour représente l'erreur renvoyée par le système d'exploitation.

Action: Vérifiez que le système ou l'unité de fichiers indiqué dans le nom du répertoire existe et que les droits autorisent l'ouverture d'un répertoire.

DMB0449W -q:datapath est requis pour la création de l'instance DB2 Extensions.

Cause: Le paramètre `-q` n'a pas été indiqué lors de la tentative de création d'une instance BD2 Extensions.

Action: Indiquez le paramètre et relancez la commande.

DMB0450W Un ou plusieurs des ports "<port>" indiqués est/sont déjà en cours d'utilisation.

Cause: Un port a été indiqué à l'usage de DB2 Extensions et est déjà répertorié dans le fichier de services comme étant en cours d'utilisation.

Action: Indiquez un ou plusieurs ports qui ne sont pas en cours d'utilisation et relancez l'application.

DMB0452E Le numéro du noeud "<num_noeud>" n'a pas été trouvé dans le fichier "db2nodes.cfg".

Cause: Le numéro du noeud de cette machine n'a pas été trouvé dans le fichier `db2nodes.cfg`.

Action: Ajoutez le numéro du noeud au fichier `db2nodes.cfg` et relancez l'application.

DMB0460W Impossible de déterminer si les ports TCP/IP sont disponibles.

Cause: Une erreur s'est produite lors de la tentative de vérification d'une éventuelle utilisation des ports TCP/IP indiqués.

Action: Vérifiez que les ports indiqués ne sont pas répertoriés dans le fichier de services comme étant en cours d'utilisation par une autre application.

DMB0462E Impossible d'initialiser ce noeud. Code retour = <code>

Cause: Une erreur s'est produite dans le démarrage des Extensions lors de la tentative d'initialisation du noeud en cours.

Action: Prenez contact avec l'assistance IBM.

DMB0495E Cette version d'Extensions AIV ne prend pas en charge les noms longs.

Cause: Vous avez spécifié un identificateur de type long pour appeler une API d'administration des extensions ou exécuter une commande de ligne de commande db2ext. La longueur maximale des identificateurs pris en charge par cette version des extensions AIV se présente comme suit :

- ID d'autorisation local (AUTHID) — 8 caractères
- Schéma de table (TABSCHEMA) — 8 caractères
- Noms de table (TABNAME) — 18 caractères
- Noms de colonne — 18 caractères

Veillez à ce que l'appel d'API ou la commande ne comporte que des identificateurs courts.

DMB0496E Un nom de table ou de colonne incorrect est spécifié.

Cause: Vous avez spécifié un identificateur incorrect pour appeler une API d'administration des extensions ou exécuter une commande de ligne de commande db2ext. Le nom de l'identificateur est probablement trop long. Pour plus d'informations sur les longueurs de noms autorisées dans DB2 UDB, consultez le manuel *Mise en route*.

Veillez à ce que l'appel d'API ou la commande ne comporte que des identificateurs courts.

DMB497E Accès refusé à DB2MMDATAPATH.

Cause: (Produit EEE uniquement) Vous avez spécifié un répertoire ou un nom partagé qui n'est pas accessible sur tous les noeuds. Or, le répertoire ou le nom partagé que vous avez indiqué lors de la création d'une instance DB2 Extensions doit exister et être accessible sur tous les noeuds. Vérifiez que le répertoire ou le nom partagé concerné remplit ces conditions.

DMB498E Au moins une partie du chemin DB2MMDATAPATH n'est pas un répertoire.

Cause: (Produit EEE uniquement) Vous avez spécifié un répertoire ou un nom partagé qui n'existe pas sur l'un des noeuds. Or, le répertoire ou le nom partagé que vous avez indiqué lors de la création d'une instance DB2 Extensions doit exister et être accessible sur tous les noeuds. Vérifiez que le répertoire ou le nom partagé concerné remplit ces conditions.

DMB499E La chaîne DB2MMDATAPATH est trop longue.

Cause: (Produit EEE uniquement) Vous avez spécifié un répertoire ou un nom partagé pour lequel la variable DB2MMDATAPATH est trop longue. Or, le répertoire ou le nom partagé que vous avez indiqué lors de la création d'une instance DB2 Extensions doit exister et être accessible sur tous les noeuds. Vérifiez que le répertoire ou le nom partagé concerné remplit ces conditions.

DMB500E Le répertoire DB2MMDATAPATH n'existe pas.

Cause: (Produit EEE uniquement) Vous avez spécifié un répertoire ou un nom partagé qui n'existe pas sur l'un des noeuds. Or, le répertoire ou le nom partagé que vous avez indiqué lors de la création d'une instance DB2 Extensions doit exister et être accessible sur tous les noeuds. Vérifiez que le répertoire ou le nom partagé concerné remplit ces conditions.

DMB501E Erreur de type stat() inconnue dans DB2MMDATAPATH

Cause: (Produit EEE uniquement) Un incident s'est produit lors d'une tentative d'accès à un répertoire ou à un nom partagé contenu dans cette variable d'environnement. Or, le répertoire ou le nom partagé que vous avez indiqué lors de la création d'une instance DB2 Extensions doit exister et être accessible sur tous les noeuds. Vérifiez que le répertoire ou le nom partagé concerné remplit ces conditions.

DMB502E DB2MMDATAPATH existe mais ce n'est pas un répertoire.

Cause: (Produit EEE uniquement) Vous avez spécifié un répertoire ou un nom partagé dont le nom ne correspond pas à celui d'un répertoire ou d'un nom partagé sur tous les noeuds. Or, le répertoire ou le nom partagé que vous avez indiqué lors de la création d'une instance DB2 Extensions doit exister et être accessible sur tous les noeuds. Vérifiez que le répertoire ou le nom partagé concerné remplit ces conditions.

DMB503E DB2MMDATAPATH existe mais n'est pas accessible en lecture.

Cause: (Produit EEE uniquement) Vous avez spécifié un répertoire ou un nom partagé dont la lecture n'est pas autorisée sur tous les noeuds. Or, le répertoire ou le nom partagé que vous avez indiqué lors de la création d'une instance DB2 Extensions doit exister et être accessible sur tous les noeuds. Vérifiez que le répertoire ou le nom partagé concerné remplit ces conditions.

DMB504E DB2MMDATAPATH existe mais n'est pas accessible en écriture.

Cause: (Produit EEE uniquement) Vous avez spécifié un répertoire ou un nom partagé dont l'écriture n'est pas autorisée sur tous les noeuds. Or, le répertoire ou le nom partagé que vous avez indiqué lors de la création d'une instance DB2 Extensions doit exister et être accessible en lecture/écriture sur tous les noeuds. Vérifiez que le répertoire ou le nom partagé concerné remplit ces conditions.

DMB504E DB2MMDATAPATH n'est pas défini.

Cause: (Produit EEE uniquement) La variable d'environnement DB2MMDATAPATH n'était pas définie lorsque vous avez créé l'instance DB2 Extensions. S'il s'agit d'une nouvelle instance, supprimez-la à l'aide de l'instruction DMBIDROP, puis recréez-la en spécifiant l'option `-q` correctement.

S'il ne s'agit pas d'une nouvelle instance DB2 Extensions, procédez comme suit :

- Dans l'environnement UNIX :
  1. Vérifiez que le répertoire existe, qu'il est correct et accessible sur tous les noeuds.
  2. Modifiez *$INSTHOME*`/dmb/dmbprofile` de façon à pouvoir exporter DB2MMDATAPATH en tant que répertoire.
- Dans l'environnement Windows :
  1. Vérifiez que le nom partagé est correct et que le répertoire existe et est accessible sur tous les noeuds.
  2. Ajoutez une entrée DB2MMDATAPATH dans le registre d'instance Extensions IAV en y indiquant ce nom partagé. Procédez comme suit : `\\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\DB2 Extenders\PROFILE\`*nom-instance*`\DB2MMDATAPATH`.

DMB506E Nom d'instance non défini.

Cause: La variable d'environnement DB2INSTANCE n'était pas définie lorsque vous avez exécuté DMBSTART. Assurez-vous que DB2START fonctionne correctement avant de lancer les fonctions DB2 Extensions à l'aide de cette commande.

DMB507E dmbssd *nom* **node** *arguments*

Cause: Erreur interne. Prenez contact avec votre partenaire commercial IBM.

DMB508E Le nombre de noeuds doit être égal ou supérieur à 0.

Cause: Erreur interne. Prenez contact avec votre partenaire commercial IBM.

DMB509E Ce programme ne doit pas être démarré manuellement.

Cause: Erreur interne. Prenez contact avec votre partenaire commercial IBM.

DMB512E Syntaxe : *arguments* **nomInstdmb.**

Cause: Erreur interne. Prenez contact avec votre partenaire commercial IBM.

DMB513E *Nom* **n'est pas une instance correcte.**

Cause: Le nom d'instance DB2 Extensions que vous tentez de supprimer n'a pas été reconnu. Vérifiez que le nom spécifié est correct et que le répertoire *$INSTHOME*`/dmb` portant ce nom existe.

DMB514I Ni le serveur, ni le client n'est installé pour cette instance.

Cause: Vous avez tenté de supprimer une instance DB2 Extensions alors que les Extensions n'ont pas été installées. Assurez-vous que l'installation est correcte et que les répertoires d'installation ont été renommés.

DMB515I L'instance DB2 N'a PAS été supprimée. Elle peut l'être à l'aide de la commande DB2IDROP.

Cause: Lorsque vous supprimez une instance DB2 Extensions, l'instance DB2 associée n'est pas automatiquement effacée. Vous devez alors utiliser DB2IDROP pour la supprimer.

DMB518E Erreur inattendue. Fonction = *nom fonction*, **Code retour =** *code_retour*.

Cause: Une erreur inattendue s'est produite lorsque vous avez tenté de créer ou de supprimer une instance DB2 Extensions. Vérifiez l'installation et la configuration du programme.

DMB520E Ce programme ne peut pas être exécuté avec les droits de niveau root.

Cause: Assurez-vous que vous disposez des droits appropriés pour effectuer cette opération.

DMB521E Une tentative de modification des droits d'accès de *nom* a échoué.

Cause: Assurez-vous que vous disposez des privilèges appropriés pour modifier vos droits d'accès.

DMB522E Une tentative de modification de la propriété de *nom* a échoué.

Cause: Assurez-vous que vous disposez des privilèges appropriés pour modifier la propriété.

DMB523E Une tentative de modification de la propriété du groupe de *nom* a échoué.

Cause: Assurez-vous que vous disposez des privilèges appropriés pour modifier la propriété du groupe.

DMB524E Le fichier ou le répertoire *nom* existe déjà.

Cause: Il existe déjà un fichier ou un répertoire portant le nom spécifié. Choisissez un autre nom et relancez la commande.

DMB525E Une tentative de création de *nom* a échoué.

Cause: Assurez-vous que vous disposez des droits appropriés pour effectuer cette opération.

DMB526E Le fichier ou le répertoire *nom* est manquant.

Cause: Le fichier ou le répertoire indiqué est introuvable. Vérifiez que le nom spécifié est correct.

DMB527E Une tentative de copie du fichier ou du répertoire *nom* vers *nom* a échoué.

Cause: Vérifiez que vous disposez des droits appropriés pour copier le fichier ou le répertoire, et que l'espace disponible est suffisant pour la copie.

DMB528E L'ID utilisateur *ID-utilisateur* est incorrect.

Cause: Vous avez indiqué un ID utilisateur qui n'est pas autorisé. Vérifiez l'ID utilisateur et relancez la commande.

DMB529E Le groupe principal *groupe* de l'ID utilisateur *ID-utilisateur* est incorrect.

Cause: Vous avez indiqué un groupe principal incorrect pour l'ID utilisateur. Recherchez le groupe principal approprié et relancez la commande.

DMB530E Le nom d'instance *nom* est incorrect.

Cause: Vous avez indiqué un nom incorrect lorsque vous avez tenté de créer ou d'utiliser une instance. Recherchez un nom d'instance correct et relancez la commande.

DMB531E Système d'exploitation non pris en charge : *nom*, version *numéro-version*.

Cause: Vous avez tenté d'exécuter cette commande sur une version non prise en charge du système d'exploitation. Vérifiez les conditions requises pour effectuer cette opération.

DMB535E Le fichier spécifié est inaccessible.

Cause: Vérifiez que vous pouvez accéder à ce fichier avant d'exécuter la commande.

DMB0533E L'API <*nom API*> n'est pas prise en charge en environnement serveur de bases de données partitionnées.

Cause: Vous ne pouvez pas utiliser l'API indiquée dans un environnement de bases de données partitionnées.

Action: Pour plus d'informations sur le mode de traitement de cette fonction dans votre application, reportez-vous à la section relative à l'API spécifiée.

DMB0534E Fonction UDF non prise en charge.

Cause: Vous ne pouvez pas utiliser la fonction utilisateur spécifiée dans un environnement de bases de données partitionnées.

Action: Consultez le message SQL0443N pour déterminer quelle fonction UDF a rencontré une erreur. Pour plus d'informations sur le mode de traitement de cette fonction dans votre application, reportez-vous à la section relative à l'UDF spécifiée.

Utilisation de la fonction de trace à des fins de diagnostic

DB2 Extensions comprend une fonction de trace qui enregistre les activités du serveur DB2 Extensions. Utilisez la fonction de trace uniquement à la demande du personnel de maintenance IBM.

La fonction de trace enregistre dans un fichier du serveur des informations sur divers événements tels que l'accès ou la sortie d'une extension DB2 ou le renvoi d'un code d'erreur par une extension DB2. En raison du nombre élevé d'événements consignés, il convient de n'utiliser la fonction de trace qu'en cas de nécessité, par exemple, lors du diagnostic d'erreurs. En outre, vous devez limiter le nombre d'applications actives lorsque vous utilisez la trace. Cela permet en effet d'isoler plus rapidement l'origine d'un incident.

La commande DMBTRC permet de contrôler la fonction de trace. Vous pouvez la lancer à partir de la ligne de commande d'un serveur OS/2, AIX ou Windows NT ou version suivante. Vous devez disposer du droit SYSADM, SYSCTRL ou SYSMINT.

La commande DMBTRC permet :

- de lancer la fonction de trace,
- d'arrêter la fonction de trace,
- de reformater les données de trace pour les rendre plus lisibles,
- d'afficher l'état de la fonction de trace.

Lancement de la fonction de trace

La fonction de trace est activée par la commande suivante :

dmbtrc on *chemin*

où *chemin* correspond au chemin du fichier du serveur qui contiendra les données de trace.

Exemple :

```
dmbtrc on /tmp/trace.txt
```

Arrêt de la fonction de trace

La fonction de trace est désactivée par la commande :

dmbtrc off

Reformatage des données de trace

Les données de trace sont enregistrées en binaire. Vous pouvez les reformater afin de les rendre plus lisibles à l'aide de la commande suivante :

`dmbtrc format` *fich_entrée fich_sortie*

où *fich_entrée* représente le fichier contenant les données de trace en binaire et *fich_sortie*, le fichier qui contiendra les données reformatées. Le paramètre *fich_sortie* est facultatif ; si vous l'omettez, les informations de reformatage sont affichées à l'écran.

Par exemple, la commande ci-après permet de reformater les informations de trace :

```
dmbtrc format /tmp/trace.txt /tmp/fmttrace.txt
```

Affichage de l'état de la fonction de trace

Entrez la commande :

```
dmbtrc info
```

pour afficher les informations suivantes sur la trace :

- état d'activation,
- chemin du fichier de trace.

Fonction de trace

Partie 5. Annexes

Annexe A. Définition des variables d'environnement de DB2 Extensions

DB2 Extensions autorise une certaine souplesse quant à l'attribution des noms de fichier lors du stockage, de l'extraction et de la mise à jour des objets image, audio ou vidéo. Ce produit est également très souple quant à la spécification des programmes d'affichage ou de lecture des objets image, audio ou vidéo qui sont extraits d'une table de base de données.

Utilisation des variables d'environnement pour la résolution des noms de fichiers

Bien que vous puissiez indiquer un nom qualifié complet (c'est-à-dire le chemin d'accès complet suivi du nom du fichier), il est préférable d'indiquer un nom de fichier relatif lors des opérations de stockage, d'extraction et de mise à jour. Sous AIX, HP-UX ou Solaris, il s'agit d'un nom de fichier ne commençant pas par une barre oblique ; sous OS/2 et Windows, il s'agit d'un nom de fichier ne commençant pas par une lettre d'unité suivie de deux points et d'une barre oblique inversée.

Si vous indiquez un nom de fichier relatif, les extensions utiliseront les spécifications de répertoire dans plusieurs variables d'environnement client et serveur pour identifier le nom de fichier. Les fichiers peuvent être ainsi déplacés dans un environnement client-serveur sans changer de nom. (Le nom qualifié complet doit être modifié chaque fois qu'un fichier est déplacé.)

Le tableau 17 répertorie et décrit les variables d'environnement que vous pouvez définir en vue d'une utilisation par les extensions Image, Audio et Vidéo pour la résolution des noms de fichiers.

Tableau 17. Variables d'environnement de DB2 Extensions

Extension Image	Extension Audio	Extension Vidéo	Description
Variables d'environnement serveur			
DB2IMAGEPATH	DB2AUDIOPATH	DB2VIDEOPATH	Utilisée pour résoudre le nom du fichier source pour les opérations de stockage, extraction et mise à jour à partir d'un fichier du serveur.
DB2IMAGESTORE	DB2AUDIOSTORE	DB2VIDEOSTORE	Utilisée pour résoudre le nom du fichier cible pour les opérations de stockage et mise à jour sur un fichier du serveur.

Tableau 17. Variables d'environnement de DB2 Extensions (suite)

Extension Image	Extension Audio	Extension Vidéo	Description
DB2IMAGEEXPORT	DB2AUDIOEXPORT	DB2VIDEOEXPORT	Utilisée pour résoudre le nom du fichier cible pour les opérations d'extraction vers un fichier du serveur.
DB2IMAGETEMP			Utilisée pour résoudre le nom du fichier cible pour les opérations de création de fichiers du serveur temporaires. Cependant, si la variable d'environnement TMP est spécifiée, c'est le répertoire TMP qui est utilisé pour résoudre les noms de fichiers.
Variables d'environnement client			
DB2IMAGEPATH	DB2AUDIOPATH	DB2VIDEOPATH	Utilisée pour résoudre le nom du fichier source pour les opérations d'affichage et de lecture sur un fichier client.
DB2IMAGETEMP	DB2AUDIOTEMP	DB2VIDEOTEMP	Utilisée pour résoudre le nom du fichier cible pour les opérations de création de fichiers client temporaires. Cependant, si la variable d'environnement TMP est spécifiée, c'est le répertoire TMP qui est utilisé pour résoudre les noms de fichiers.

Si vous ne définissez pas la variable d'environnement appropriée pour une extension spécifique, l'extension utilise les variables d'environnement suivantes pour la résolution des noms de fichiers :

Variable d'environnement	Description
DB2MMPATH	Utilisée pour résoudre le nom du fichier source pour les opérations de stockage, d'extraction et de mise à jour.
DB2MMSTORE	Utilisée pour résoudre le nom du fichier cible pour les opérations de stockage et de mise à jour.
DB2MMEXPORT	Utilisée pour résoudre le nom du fichier cible pour les opérations d'extraction.
DB2MMTEMP	Utilisée pour résoudre le nom du fichier pour les opérations de création de fichiers temporaires.

Utilisation des variables d'environnement pour identifier les programmes d'affichage ou de lecture

Outre la résolution des noms de fichiers, les variables d'environnement permettent l'identification des programmes d'affichage d'objets image extraits par l'extension Image et de lecture des objets audio ou vidéo extraits par l'extension Audio ou Vidéo. Pour afficher ou lire ces objets, vous devez utiliser respectivement les API DBiBrowse, DBaPlay et DBvPlay. Lorsque vous utilisez une API, vous pouvez spécifier un programme d'affichage ou de lecture, ou indiquer que vous souhaitez utiliser un programme par défaut.

DB2 Extensions utilise les variables d'environnement suivantes sur le poste client pour identifier les programmes d'affichage ou de lecture par défaut :

Variable d'environnement	Description
DB2IMAGEBROWSER	Utilisée pour identifier le programme d'affichage d'image par défaut.
DB2AUDIOPLAYER	Utilisée pour identifier le programme de lecture audio par défaut.
DB2VIDEOPLAYER	Utilisée pour identifier le programme de lecture vidéo par défaut.

Utilisation de la variable d'environnement DB2MMDATAPATH (Produit EEE uniquement)

Les extensions DB2 utilisent la variable d'environnement DB2MMDATAPATH pour la résolution des emplacements d'un certain nombre d'opérations dans un environnement de bases de données partitionnées. Par exemple, l'extension DB2 Image utilise la valeur de DB2MMDATAPATH pour le stockage des données QBIC dans un environnement de bases de données partitionnées.

La définition de DB2MMDATAPATH a lieu lors de la création d'une instance d'extension DB2, comme décrit à la section «DMBICRT» à la page 530 et dans chacun desfichiers «readme» d'installation suivants :

- install.txt dans le répertoire aixeee (Installation des extensions DB2 pour une utilisation avec DB2 Extended Enterprise Edition sous AIX)
- install.txt dans le répertoire soleee (Installation des extensions DB2 pour une utilisation avec DB2 Extended Enterprise Edition sous environnement d'exploitation Solaris)

Voici un exemple d'utilisation de DB2MMDATAPATH pour le stockage des caractéristiques QBIC et des données d'index. Sous UNIX, l'extension DB2 Image stocke ces données QBIC dans le répertoire suivant :

db2mmdatapath /NODE*num_noeud*/QBIC/*nom_bd*

où *db2mmdatapath* représente la valeur de la variable d'environnement DB2MMDATAPATH, *num_noeud* le numéro du noeud et *nom_bd* le nom de la base de données.

Considérons l'exemple AIX suivant. Supposons que la valeur de la variable DB2MMDATAPATH soit /localfs/dmbdata. Supposons également qu'une base de données appelée "exemple" soit partitionnée en noeuds 0, 2 et 5. Les données QBIC seront stockées pour la base de données "exemple" dans les répertoires suivants :

Noeud 0 : /localfs/dmbdata/NODE0000/QBIC/exemple

Noeud 2 : /localfs/dmbdata/NODE0002/QBIC/exemple

Noeud 5 : /localfs/dmbdata/NODE0005/QBIC/exemple

Définition des variables d'environnement

Il est possible de définir des variables d'environnement sous AIX, HP-UX, Solaris, OS/2 et Windows.

Définition de variables d'environnement sur les postes serveur et client AIX, HP-UX, Solaris

Sous AIX, HP-UX, et Solaris, les variables d'environnement sont spécifiées sans les scripts Korn, Bourne et C shell. Lorsque les Extensions DB2 sont installées, les variables d'environnement du serveur sont définies comme suit :

C shell

```
setenv
DB2MMPATH /usr/lpp/db2ext/exemples:/tmp
setenv DB2MMTEMP /tmp
setenv DB2MMSTORE /tmp
setenv DB2MMEXPORT /tmp
```

Korn et Bourne shells

```
DB2MMPATH=/usr/lpp/db2ext/exemples:/tmp
export DB2MMPATH

DB2MMSTORE=/tmp
export DB2MMSTORE

DB2MMEXPORT=/tmp
export DB2MMEXPORT

DB2MMTEMP=/tmp
export DB2MMTEMP
```

Les variables d'environnement du serveur sont initialement définies par des valeurs qui vous permettent d'accéder aux fichiers de support utilisés dans les

modèles de programmes fournis avec DB2 Extensions. (Pour plus d'informations, reportez-vous à l'«Annexe B. Modèles de programmes et fichiers de support» à la page 595.)

Les variables d'environnement client sont définies comme suit lors de l'installation de DB2 Extensions sur un poste client AIX, HP-UX ou Solaris :

C shell

```
setenv DB2MMPATH /tmp
setenv DB2MMTEMP /tmp
```

Korn et Bourne shells

```
DB2MMPATH=/tmp
export DB2MMPATH

DB2MMTEMP=/tmp
export DB2MMTEMP
```

Définissez les variables d'environnement client et serveur utilisées pour résoudre les noms de fichiers. Indiquez des valeurs adaptées à votre environnement. Vous pouvez indiquer plusieurs répertoires, séparés par un délimiteur, pour les variables se terminant par PATH. Celles qui se terminent par STORE, EXPORT et TEMP sont définies avec un seul répertoire.

Indiquez respectivement les noms des programmes appropriés d'affichage d'image et de lecture audio et vidéo dans les variables d'environnement client DB2IMAGEBROWSER, DB2AUDIOPLAYER et DB2VIDEOPLAYER.

Vous pouvez modifier les définitions initiales des variables d'environnement comme suit :

C shell

La commande SETENV permet de définir les variables d'environnement :

```
setenv var-env répertoire
```

Par exemple :

```
setenv DB2MMPATH
/usr/lpp/db2ext/exemples:/media
setenv DB2IMAGEPATH /Employés/photos:/images
setenv DB2AUDIOSTORE /Employés/sons
setenv DB2IMAGEBROWSER 'xv %s'
```

Bourne shell

La commande EXPORT permet de définir les variables d'environnement :

```
var-env=répertoire
export var-env
```

Par exemple :

```
DB2MMPATH=/usr/lpp/db2ext/exemples:/media
export DB2MMPATH

DB2IMAGEPATH=/Employés/photos:/images
export DB2IMAGEPATH

DB2AUDIOSTORE=/Employés/sons
export DB2AUDIOSTORE
```

Korn Shell

La commande EXPORT permet de définir les variables d'environnement :

```
export var-env=répertoire
```

Par exemple :

```
export
DB2MMPATH=/usr/lpp/db2ext/exemples:/media
export DB2IMAGEPATH=/Employés/photos:/images
export DB2AUDIOSTORE=/Employés/sons
```

Définition de variables sur les postes serveur et client OS/2

Sous OS/2, les variables d'environnement sont ajoutées à votre fichier CONFIG.SYS et automatiquement définies durant l'installation.

Si vous installez DB2 Extensions sur un serveur OS/2, les variables d'environnement du serveur sont définies comme suit :

```
SET DB2MMPATH=rép-install\EXEMPLES;rép-fich-temp
SET DB2MMSTORE=rép-fich-temp
SET DB2MMEXPORT=rép-fich-temp
SET DB2MMTEMP=rép-fich-temp
```

où *rép-install* est le répertoire d'installation et *rép-fich-temp* est le répertoire de fichiers temporaires. Le répertoire d'installation par défaut est `C:\DMB` et le répertoire des fichiers temporaires par défaut est `C:\DMB\TMP`. Vous pouvez modifier l'emplacement de ces répertoires lors de l'installation. Il est important d'indiquer correctement l'emplacement du répertoire de fichiers temporaires.

Les variables d'environnement du serveur sont initialement définies par des valeurs qui vous permettent d'accéder aux fichiers de support utilisés dans les

modèles de programmes fournis avec DB2 Extensions. (Pour plus d'informations, reportez-vous à l'«Annexe B. Modèles de programmes et fichiers de support» à la page 595).

Si vous installez DB2 Extensions sur un client OS/2, les variables d'environnement client sont définies comme suit :

```
SET DB2MMPATH=rép-fich-temp
SET DB2MMTEMP=rép-fich-temp
```

Utilisez la commande SET pour redéfinir les variables d'environnement. Vous pouvez indiquer plusieurs répertoires, séparés par un délimiteur, pour les variables se terminant par PATH. Celles qui se terminent par STORE, EXPORT et TEMP sont définies avec un seul répertoire.

Utilisez la commande SET pour spécifier respectivement les programmes appropriés d'affichage d'image, et de lecture audio et vidéo dans les variables d'environnement DB2IMAGEBROWSER, DB2AUDIOPLAYER et DB2VIDEOPLAYER.

Indiquez la commande SET comme suit :

```
SET var-env=répertoire
```

par exemple,

```
SET DB2MMPATH=C:\DMB\EXEMPLES;\D:\MEDIA
SET DB2IMAGEPATH=C:\Employés\PHOTOS;D:\IMAGES
SET DB2AUDIOSTORE=C:\Employés\SONS
SET DB2IMAGEBROWSER=ib.exe %s
```

Définition de variables d'environnement sur les postes serveur et client Windows

Sous Windows, le mode de définition des variables d'environnement dépend de l'utilisation ou non des extensions DB2 dans un environnement de bases de données partitionnées (à savoir, avec DB2 Extended Enterprise Edition pour Windows).

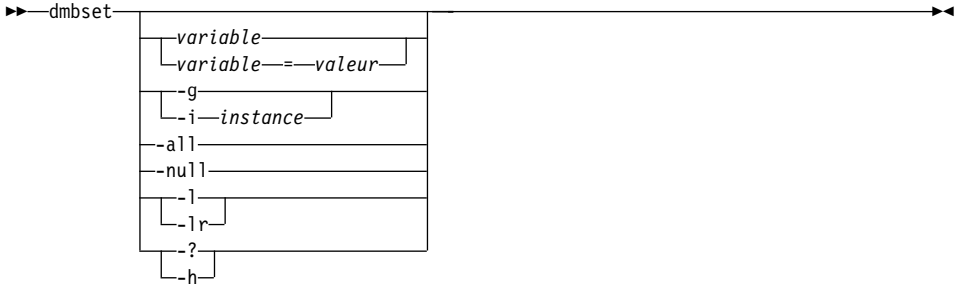
Définition de variables d'environnement dans des environnements de bases de données non partitionnées sous Windows (ne s'applique pas au produit EEE)

Sous Windows, les variables d'environnement sont stockées dans le registre système. Les variables peuvent être définies par l'ouverture du panneau de configuration et la sélection de l'icône système. Sélectionnez l'onglet Environnement de la boîte de dialogue Propriétés du système. Deux fenêtres contiennent les variables d'environnement et leurs valeurs. La fenêtre supérieure affiche les variables s'appliquant à tous les utilisateurs. La fenêtre inférieure affiche les variables s'appliquant uniquement à l'utilisateur courant.

Définition de variables d'environnement dans les environnements de bases de données partitionnées Windows (Produit EEE uniquement)

Dans un environnement partitionné Windows, toutes les variables utilisées par les extensions DB2 sont stockées dans une zone privée du registre système. Un programme appelé DMBSET permet la vérification et la modification des variables d'extension.

La syntaxe du programme est la suivante :



Pour interroger la valeur d'une variable, entrez dmbset *nom_variable*. Par exemple :

```
dmbset DB2MMPATH
```

Pour définir la valeur d'une variable, entrez : dmbset *nom_variable=valeur*. Par exemple :

```
dmbset DB2MMPATH=C:\DMB\EXEMPLES
```

Pour afficher la valeur de toutes les variables d'une instance définie, entrez dmbset -i *nom_instance*. Par exemple :

```
dmbset -i dmbinst1
```

Pour qu'une valeur soit nulle, entrez dmbset *nom_variable -null*. Par exemple :

```
dmbset DB2MMPATH -null
```

Pour afficher la valeur des variables utilisées par toutes les instances, entrez `dmbset -g`.

Pour dresser la liste de toutes les variables utilisées par DB2 Extensions, entrez `dmbset -lr`.

Pour répertorier les noms de tous les profils d'instance définis dans le registre, entrez `dmbset -l`.

Vous disposez d'une grande marge de manoeuvre pour la définition des variables d'environnement pour DB2 Extensions dans un environnement de bases de données partitionnées. Par exemple, vous pouvez indiquer des valeurs pour chaque variable d'environnement, à l'exception de DB2MMDATAPATH, dans l'un des formats suivants :

- Convention de dénomination universelle : *nom_machine**nom_partage*. Par exemple :

  ```
  \\harmony\JimsShr
  ```

- Chemin de l'unité. Par exemple :

  ```
  f:\media
  ```

- Autre : *nom_partage**nom_répertoire*. Par exemple :

  ```
  JimsShr\images
  ```

Annexe B. Modèles de programmes et fichiers de support

DB2 Extensions est fourni avec de nombreux modèles de programmes. Ces modèles utilisent des fichiers image, audio et vidéo également fournis. La plupart des modèles de programmes sont écrits en langage C. Tous les modèles de programmes C sont disponibles en format CLI (Call Level Interface). En outre, plusieurs modèles de programmes Java et un modèle de fichier de macros Net.Data sont fournis.

Les modèles de programmes sont installés dans le sous-répertoire `SAMPLES` du répertoire cible lors de l'installation des extensions DB2. Les fichiers image, audio et vidéo sont installés dans ce même sous-répertoire lors de l'installation des extensions DB2. Lors de l'installation, les variables d'environnement des extensions sont définies pour désigner ce sous-répertoire.

Modèles de programmes

Plusieurs fichiers constituent les modèles de programmes pour les extensions DB2. Ces fichiers sont les suivants :

Fichier	Description
enable.c	Active une base de données pour les extensions Audio, Image et Vidéo, crée une table et active la table et ses colonnes.
populate.c	Importe des données dans la table (le programme est en format C).
Populate.java	Importe des données dans la table (le programme est en format Java).
query.c	Recherche des données dans la table (le programme est en format C).
Query.java	Recherche des données dans la table (le programme est en format Java).
api.c	Utilise les API d'extension pour interroger la base de données.
handle.c	Effectue une démonstration de l'utilisation d'indicateurs dans les fonctions UDF et des comparaisons de clauses WHERE dans les instructions SELECT.
qbcatdmo.c	Crée un catalogue QBIC et y catalogue une colonne d'images.
qbicdemo.c	Effectue des recherches dans un catalogue QBIC.

color.c	Effectue des déclarations de table de couleurs pour `qbicdemo.c`
QbicQry.java	Présente des sélecteurs de couleur moyenne et d'histogramme pour une requête QBIC.
makesf.c	Crée un catalogue des prises de vue pour utilisation avec `makehtml·exe`.
makehtml.c	Accède au fichier catalogue et crée des pages HTML qui pourront être affichées par un afficheur Web.
storybrd.java	Applet permettant d'afficher les prises de vue appelées par les pages HTML, elles-mêmes générées par le modèle de programme `makehtml.c`
utility.c	Routines d'utilitaire.
utility.h	Fichier d'en-tête pour routines d'utilitaire.
makefile.aix	Fichier makefile permettant de concevoir des programmes sous AIX.
makefile.os2	Fichier makefile permettant de concevoir des programmes sous OS/2.
makefile.iva	Fichier makefile permettant de concevoir des programmes sous Windows NT (ou version suivante) à l'aide d'IBM VisualAge C++.
makefile.mvc	Fichier makefile permettant de concevoir des programmes sous Windows à l'aide de Microsoft Visual C++.
makefile.sun	Fichier makefile permettant de concevoir des programmes sous Solaris.
makefile.hp	Fichier makefile permettant de concevoir des programmes sous HP-UX.

Les modèles de programmes suivants sont fournis avec des fichiers exécutables. Ces programmes doivent être exécutés dans l'ordre indiqué.

1. Enable
2. Populate
3. Query
4. API
5. Handle
6. Qbcatdmo
7. Qbicdemo
8. QbicQry
9. Makesf

10. Makehtml

Des fichiers de classe exécutables (`Populate.class`, `Query.class`, `QbicQry.class` et `storybrd.class`) sont fournis avec les modèles de programmes Java.

Avant d'exécuter les modèles de programmes, vous devez créer une base de données sur votre serveur. Les fonctions d'extension doivent également avoir été démarrées sur le serveur. Pour exécuter un modèle de programme, entrez le nom du programme afin de lancer le fichier exécutable correspondant. Vous êtes ensuite invité à entrer le nom de la base de données, l'ID utilisateur et le mot de passe. Utilisez l'ID et le mot de passe de l'utilisateur ayant créé la base de données.

Vous pouvez aussi concevoir vos propres fichiers exécutables pour les modèles de programmes. Pour cela :

1. Copiez les fichiers du modèle de programme dans un répertoire modifiable.
2. Modifiez le fichier makefile afin de préciser l'emplacement de DB2, des extensions et du compilateur sur votre système.
3. Utilisez les commandes make ou nmake pour compiler les fichiers en programmes exécutables.

Pour plus d'informations sur l'installation et l'utilisation de modèles de programmes, reportez-vous au fichier `README.CNT` du répertoire de modèles de programmes.

Modèles de fichiers image, audio et vidéo

Modèles de fichiers image, audio et vidéo livrés avec les extensions DB2 :

- Fichiers image
  - `lizzi.bmp`
  - `sws_stri.bmp`
  - `nitecry.bmp`
  - `ranger_r.bmp`
  - `fuzzblue.bmp`
- Fichiers audio
  - `lizzi.wav`
  - `sws_stri.wav`
  - `nitecry.wav`
  - `ranger_r.wav`
  - `fuzzblue.wav`

Modèles de fichiers de support

- Fichiers vidéo
  - `nitecry.avi`
  - `sample.mpg`

Modèle de fichier de macros Net.Data

Un fichier de macros Net.Data, `extender.d2w`, est fourni avec les extensions DB2. Lorsqu'il est lancé sur un serveur Web, ce fichier exécute des instructions SQL qui appellent des fonctions UDF d'extension DB2. Le fichier de macros renvoie des résultats, ces derniers étant affichés par un navigateur Web. Comme illustré à la figure 30, chaque page de résultat indique également l'instruction SQL qui a servi à générer le résultat. La figure 31 à la page 599, présente le contenu du modèle de fichier de macros Net.Data.

Pour lancer ce modèle, entrez l'URL suivant sous un navigateur Web :
`http://`*`votre serveur`*`/cgi-bin/db2www/extender.d2w/startHere`

où *votre serveur* est le nom du serveur Web utilisé.

Figure 30. Application Web exécutant le modèle de fichier de macros Net.Data. Chaque page de résultat indique l'instruction SQL qui a servi à générer le résultat.

```
%{ ------------------------------------------------------------------------ %}
%{ Copyright International Business Machines Corporation, 1998.            %}
%{ All rights reserved.                                                     %}
%{                                                                          %}
%{ Sample Net.Data macro which shows how to call image, audio, and video   %}
%{ extender UDFs.                                                          %}
%{                                                                          %}
%{ To run, put this macro in your MACRO_PATH root, make sure the tmplobs   %}
%{ directory exists under your web server's document root, and create      %}
%{ the database to be used when running the extender sample programs       %}
%{ 'enable' and 'populate'.  Run 'enable' and 'populate'.  If you name your %}
%{ database something other than 'testdb2', you'll need to change the     %}
%{ definition of DATABASE below.  The extender environment variable       %}
%{ DB2MMEXPORT needs to be set for the instance used by Net.Data to point  %}
%{ to the webserver's <document root>/tmplobs directory.  Then restart DB2 %}
%{ and the extenders to have the variable take effect.                     %}
%{ If you are not running Net.Data's Connection Manager, you'll need to    %}
%{ provide the LOGIN and PASSWORD to the database.  If these instructions  %}
%{ seem unfamiliar to you, you should read the Net.Data documentation at   %}
%{ http://www.software.ibm.com/data/netdata/docs (or the extender documen- %}
%{ tation on the extender sample programs).                                %}
%{                                                                          %}
%{ To disable the showing of SQL statements, change the value of SHOWSQL   %}
%{ below to "no".                                                          %}
%{ ------------------------------------------------------------------------ %}

%{ ------------------------------------------------------------------------ %}
%{ Definitions section                                                     %}
%{ ------------------------------------------------------------------------ %}
%define{
  DATABASE="testdb2"
  SHOWSQL="yes"
%}
```

Figure 31. Modèle de fichier de macros Net.Data (Numéro 1 de 5)

```
%{ ------------------------------------------------------------------------ %}
%{ SQL functions                                                            %}
%{ ------------------------------------------------------------------------ %}
%function (DTW_SQL) startHereSQL(){
  select artist, title, stock_no, price from sobay_catalog

  %REPORT{
   <table border="2" bgcolor="#b1b1b1">
           <tr><th>Artist <th> Title <th> Stock<th> Number <th> Price </tr>
    %ROW{ <tr><td> $(V_artist) <td> $(V_title) <td>  $(V_stock_no) <td> $(V_price) <tr>
    %}
    </table>
   %}
%}

%function (DTW_SQL) addThumbsSQL(){
  select cast(mmdbsys.thumbnail(covers) as blob(10000)),
         cast(mmdbsys.thumbnail(video) as blob(3000)),
         mmdbsys.comment(music), artist, title, price, stock_no
    from sobay_catalog

  %REPORT{
   <table border="2" bgcolor="#b1b1b1">
                <tr><th>Cover <th>Video <th>Audio <th>Artist <th>Title <th>Price </tr>
    %ROW{  <tr><td>< a href="showCover?stock_no=$(V_stock_no)"><img src="$V1)"></a>
               <td><  a href="getVideo?stock_no=$(V_stock_no)"><img src="$V2)"></a>
               <td><  a href="getAudio?stock_no=$(V_stock_no)&filename=$V3")>[Listen]</a>
               <td> $(V_artist) <td> $(V_title) <td> $(V_price) </tr>
    %}
    </table>
  %}
%}

%function (DTW_SQL) showCoverSQL(){
  select cast(mmdbsys.content(covers, 'GIF') as blob(150000)), mmdbsys.format(covers)
    from sobay_catalog
   where stock_no = '$(stock_no)'

  %REPORT{
    %ROW{  <img src="$(V1) "> <br><br><b>Original image format: $(V2)</b>%}
  %}
%}
```

Figure 31. Modèle de fichier de macros Net.Data (Numéro 2 de 5)

```
%{ The following Content call depends on DB2MMEXPORT being set properly to
   point to the tmplobs directory under the web server's document root.  %}

%function (DTW_SQL) showVideoSQL(){
  select mmdbsys.comment(video), mmdbsys.content(video, mmdbsys.comment(video), 1),
         mmdbsys.format(video)
    from sobay_catalog
   where stock_no = '$(stock_no)'

  %REPORT{
    %ROW{ <a href="/tmplobs/$(V1)"><i><b> Play Video Clip></b></i></a>
       <br><br><b>Format: $(V3) <br>(Note: NT/Win95 may not come with
          a decompressor<br>for this video format.  OS/2 Warp does.)</br>
    %}
  %}
%}

%{ The following Content call depends on DB2MMEXPORT being set properly to
   point to the tmplobs directory under the web server's document root.  %}

%function (DTW_SQL) showAudioSQL(){
  select mmdbsys.comment(music), mmdbsys.content(music, mmdbsys.comment(music), 1),
         mmdbsys.format(music)
    from sobay_catalog
   where stock_no = '$(stock_no)'

  %REPORT{
      %ROW{ < a href="/tmplobs/$(V1) "<i><b>Play Audio Clip</b></i></a>
         <br><br><b>Format: $(V3)</b>
    %}
  %}
%}
```

Figure 31. Modèle de fichier de macros Net.Data (Numéro 3 de 5)

```
%{ ------------------------------------------------------------------------ %}
%{ HTML sections                                                            %}
%{   E.g., http://<your server>/cgi-bin/db2www/extender.d2w/startHere       %}
%{ ------------------------------------------------------------------------%}
%{   E.g., http://
%{   E.g., http://
%HTML(startHere){
<html>
   <head><title>UDB Extenders Macro Sample: Simple Row Listing</title></head>
   <body bgcolor="#ffffff">
   <font color="#3300ff" size="3"><b>If no data appears below, you might need
    to run the UDB Extender sample programs <i>enable</i> and <i>populate</i>.
    This first HTML section of the extender.d2w macro simply retrieves all the
    traditional data for all the rows in the UDB Extenders' sample database.
    %if ( "$(SHOWSQL)" == "yes" || "$(SHOWSQL)" == "YES" )
    <br><br> By default, every page generated by this macro shows the SQL used
      to generate that page.  Here is the SQL statement for this page:
    %else
      <br>
    %endif
    </b></font>
     <br>@startHereSQL()
    <br><b>Click < a href="addThumbs"><i>here</i></a> to display thumbnails
    and links to image/audio/video data.</b>
 </body>
</html>
%}

%HTML(addThumbs){
<html>
   <head><title>UDB Extenders Macro Sample: Add Thumbnails</title></head>
   <body bgcolor="#ffffff">
     <font color="#3300ff" size="3"><b>This page adds album cover thumbnails
    and links to display the multimedia content of the database.  To access
    the multimedia content:
    <ul>
      <li> Click on a thumbnail of a CD cover to view a full-size image
      <li> Click on a "video thumbnail" to view a video
      <li> Click on a "[Listen]" link to listen to an audio clip
    </ul>
    </b></font> @addThumbsSQL()
    <br><b>Click < a href="startHere"><i>here</i></a> to go back to the first page.</b>
  </body>
</html>
%}
```

Figure 31. Modèle de fichier de macros Net.Data (Numéro 4 de 5)

```
%HTML(showCover){
<html>
   <head><title>UDB Extenders Macro Sample: Cover for item $(stock_no)</title></head>
   <body bgcolor="#ffffff">
     <font color="#3300ff" size="3"><b>For this page, the macro gets a full-size cover
image, converting the image format to GIF so that a browser can show it:
    </b></font><br><br>
    <table width="400" border="2" bgcolor="#b1b1b1" cellpadding="5">
      <tr><td align=center> @showCoverSQL()
      <tr><td align=center> <b>Stock Number: $(stock_no)</b>
    </table>
    <br><b>Go <a href="addThumbs"><i>back</i></a>.</b>
  </body>
</html>
%}

%HTML(getVideo){
<html>
   <head><title>UDB Extenders Macro Sample: Video clip for item $(stock_no)</title></head>
   <body bgcolor="#ffffff">
     <font color="#3300ff" size="3"><b>From this page, you can view a video clip:
    </b></font><br><br>
    <table width="400" border="2" bgcolor="#b1b1b1" cellpadding="5">
      <tr><td align=center> @showVideoSQL()
      <tr><td align=center> <b>Stock Number: $(stock_no)</b>
    </table>
    <br><b>Go <a href="addThumbs"><i>back</i></a>.</b>
  </body>
</html>
%}

%HTML(getAudio){
<html>
   <head><title>UDB Extenders Macro Sample: Audio clip for item $(stock_no)</title></head>
   <body bgcolor="#ffffff">
     <font color="#3300ff" size="3"><b>From this page, you can listen to an audio clip:
    </b></font><br><br>
    <table width="400" border="2" bgcolor="#b1b1b1" cellpadding="5">
      <tr><td align=center> @showAudioSQL()
      <tr><td align=center> <b>Stock Number: $(stock_no)</b>
    </table>
    <br><b>Go <a href="addThumbs"><i>back</i></a>.</b>
 <body>
</html>
%}
```

Figure 31. Modèle de fichier de macros Net.Data (Numéro 5 de 5)

Modèles de fichiers de support

Annexe C. Remarques

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevets couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM EMEA Director of Licensing
IBM Europe Middle-East Africa
Tour Descartes
La Défense 5
2, avenue Gambetta
92066 Paris-La Défense Cedex
France

Pour le Canada, veuillez adresser votre courrier à :

IBM Director of Commercial Relations
IBM Canada Ltd
3600 Steeles Avenue East
Markham, Ontario
L3R 9Z7
Canada

Les informations sur les licences concernant les produits utilisant un jeu de caractères double octet peuvent être obtenues par écrit à l'adresse suivante :

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japon

Le paragraphe suivant ne s'applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales : LE PRESENT DOCUMENT EST LIVRE «EN L'ETAT». IBM DECLINE TOUTE RESPONSABILITE, EXPRESSE OU IMPLICITE, RELATIVE AUX INFORMATIONS QUI Y SONT CONTENUES, Y COMPRIS EN CE QUI CONCERNE LES GARANTIES DE QUALITE MARCHANDE OU D'ADAPTATION A VOS BESOINS. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Il est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut modifier sans préavis les produits et logiciels décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies.

Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

IBM Canada Limited
Office of the Lab Director
1150 Eglinton Ave. East
North York, Ontario
M3C 1H7
CANADA

Ces informations peuvent être soumises à des conditions particulières prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans ce document et tous les éléments sous licence disponibles s'y rapportant sont fournis par IBM conformément aux termes du Contrat sur les produits et services IBM, des Conditions internationales d'utilisation des logiciels IBM ou de tout autre accord équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles. IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis, et doit être considérée uniquement comme un objectif.

Ce document peut contenir des exemples de données et des rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

LICENCE DE COPYRIGHT :

Le présent logiciel peut contenir des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes plateformes d'exploitation. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquelles ils ont été écrits ou aux interfaces de programmation IBM. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes.

Toute copie totale ou partielle de ces programmes exemples et des oeuvres qui en sont dérivées doit comprendre une notice de copyright, libellée comme suit :

© (nom de votre société) (année). Des segments de code sont dérivés des Programmes exemples d'IBM Corp. © Copyright IBM Corp. _indiquez l'année ou les années_. All rights reserved.

Marques

Les termes qui suivent, accompagnés d'un astérisque (*) dans le document, sont des marques d'International Business Machines Corporation dans certains pays.

ACF/VTAM
AISPO
AIX
AIX/6000
AIXwindows
AnyNet
APPN
AS/400
BookManager
CICS
C Set++
C/370
DATABASE 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eNetwork
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
LAN DistanceMVS
MVS/ESA
MVS/XA
Net.Data
OS/2
OS/390
OS/400
PowerPC
QBIC
QMF
RACF
RISC System/6000
RS/6000
S/370
SP
SQL/DS
SQL/400
System/370
System/390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WIN-OS/2

Les termes qui suivent sont des marques d'autres sociétés :

Microsoft, Windows et Windows NT sont des marques de Microsoft Corporation dans certains pays.

Java, ou toutes les marques et logos incluant Java, et Solaris sont des marques de Sun Microsystems, Inc.

Tivoli et NetView sont des marques de Tivoli Systems Inc. dans certains pays.

UNIX est une marque enregistrée aux Etats-Unis et/ou dans d'autres pays et utilisée avec l'autorisation exclusive de la société X/Open Company Limited.

D'autres sociétés sont propriétaires des autres marques, noms de produits ou logos accompagnés de deux astérisques (**) qui pourraient apparaître dans ce document.

Glossaire

analyser : Calculer des valeurs numériques pour les caractéristiques d'une image et ajouter ces valeurs dans un catalogue QBIC.

API : Voir *Interface de programmation d'applications*.

base de données partitionnée : Base de données composée d'au-moins deux partitions de base de données. Les données figurant dans les tables utilisateur peuvent se trouver dans une ou plusieurs partitions de base de données. Lorsqu'une table est à cheval sur plusieurs partitions, certaines de ses lignes sont stockées dans une partition et d'autres ailleurs dans d'autres partitions.

caractéristique : Attribut visuel d'une image, tel que la *couleur moyenne*.

catalogue des prises de vue : Table ou fichier de base de données utilisé pour stocker des données relatives à des prises de vue, telles que le numéro de la première et de la dernière image d'une séquence vidéo. Un utilisateur peut accéder à une vue de la table via une requête SQL, ou accéder aux données dans le fichier.

catalogue QBIC : Référentiel qui contient des données relatives aux caractéristiques visuelles d'images.

chaîne de requête : Chaîne de caractères spécifiant les caractéristiques, leurs valeurs et leurs poids pour une requête QBIC. La chaîne de requête peut être entrée dans une requête à partir de la ligne de commande DB2. S'oppose à objet de requête

changement d'échelle : Ajout de *noeuds* à une base de données pour augmenter les performances et la mémoire de stockage.

changement de plan : Point d'une *séquence vidéo* au niveau duquel existe une différence significative entre deux images successives. On constate, par exemple, qu'un changement de plan se produit lorsque l'objectif de la caméra est déplacé au cours d'une séquence vidéo.

contraste : Attribut de *texture* relatif à l'accentuation des brillances entre parties claires et parties sombres de l'image et qui se mesure par un histogramme des gris.

couleur d'histogramme : Mesure des différentes couleurs d'une image. Les données concernant chaque couleur sont stockées séparément dans un *catalogue QBIC*.

couleur moyenne : Mesure de couleur permettant de calculer la moyenne des valeurs de couleur contenues dans les pixels d'une image.

couleur positionnelle : *Couleur moyenne* des pixels dans une zone spécifique d'une image.

déclencheur : Définition d'une série d'actions devant être exécutées lors de la modification d'une table. Les déclencheurs peuvent être utilisés pour des opérations telles que la validation des données d'entrée, la génération automatique d'une valeur pour une nouvelle ligne insérée, la lecture d'autres tables pour la résolution de références croisées, ou l'écriture de données dans d'autres tables à des fins de contrôle. Le plus souvent, les déclencheurs sont utilisés pour le contrôle de l'intégrité des données et pour l'application de règles propres à l'activité de l'entreprise.

descripteur : Chaîne de caractères créée par une extension et qui est utilisée pour représenter un objet image, audio ou vidéo dans une table. Le descripteur d'un objet est stocké dans une table utilisateur et dans des *tables de gestion*. L'extension peut ainsi établir un lien entre le descripteur stocké dans la table utilisateur et les informations stockées dans les tables de gestion.

directionnalité : Attribut de *texture* qui indique si l'image possède une orientation (comme des brins d'herbe) ou ressemble à un objet lisse (comme une plaque de verre).

enchaîner : Diminuer la puissance d'un signal d'une image vidéo au fur et à mesure de l'accroissement de celle de l'image vidéo suivante.

enregistrement audio : Portion d'informations enregistrées pouvant être ré-écoutées.

extension : Voir *extension DB2*.

extension DB2 : Programme faisant partie d'un groupe de programmes, permettant de stocker et d'extraire des types de données sortant du cadre traditionnel des données de type caractères et des données numériques, tels que image, son, vidéo et des documents complexes.

fichier d'indexation : Fichier contenant des informations d'indexation utilisées par l'extension Vidéo pour rechercher un *plan* ou une image individuelle dans une séquence vidéo.

fonction utilisateur (UDF) : Fonction définie par un utilisateur dans DB2. Une fois définie, la fonction peut être utilisée dans des requêtes SQL et dans des objets vidéo. Par exemple, des fonctions UDF peuvent être créées pour obtenir le format de compression d'un objet vidéo ou pour renvoyer la fréquence d'échantillonnage d'un objet audio. Vous avez ainsi la possibilité de définir le comportement des objets d'un type particulier.

gigaoctet (Go) : Un milliard (10^9) d'octets. Lorsqu'on se réfère à la capacité mémoire, 1 073 741 824 octets.

grain : Attribut de *texture* qui mesure l'échelle de texture (relief granité par rapport à surface lisse).

groupe de noeuds : Groupe nommé constitué d'une ou de plusieurs partitions de base de données.

groupe de noeuds multipartition : *groupe de noeuds* contenant plusieurs *serveurs de partitions de bases de données*.

image : Représentation électronique d'un dessin.

index vidéo : Fichier utilisé par l'extension Vidéo pour rechercher une *prise de vue* ou une image particulière dans une séquence vidéos.

instance : Environnement serveur logique de l'extension DB2. Vous pouvez disposer de plusieurs instances de serveur DB2 Extensions sur le même poste de travail, mais une seule instance pour chaque instance DB2. Vous pouvez utiliser ces instances pour :

Séparer l'environnement de développement de l'environnement de production

Limiter l'accès à des informations confidentielles à un groupe de personnes particulier.

Interface de programmation d'applications (API) :

(1) Interface fonctionnelle fournie par le système d'exploitation ou par un logiciel sous licence à commander. Elle permet à un programme d'application écrit en langage évolué d'utiliser des données ou des fonctions spécifiques au système d'exploitation ou aux logiciels sous licence.

(2) Dans DB2, fonction d'interface. Par exemple, l'API "Get error message".

(3) DB2 Extensions comporte des API permettant de demander des fonctions UDF, des opérations de gestion, des opérations d'affichage et la fonction de détection de changement de plan vidéo.

kilo-octet (ko) : Mille (10^3) octets. Lorsque l'on se réfère à la capacité de mémoire, 1024 octets.

mégaoctet (Mo) : Un million (10^6) d'octets. Lorsque l'on se réfère à la capacité mémoire, 1 048 576 octets.

miniature : Image de taille réduite.

noeud : Dans le partitionnement de base de données, synonyme de partition de base de données.

objet : En programmation orientée objet, abstraction se composant de données et d'opérations associées à ces données.

objet BLOB : Chaîne binaire dont la longueur peut aller jusqu'à 2 Go. Les objets image, audio et vidéo sont stockés dans une base de données DB2 sous forme d'objets BLOB.

objet CLOB : Chaîne de caractères mono-octet dont la taille peut aller jusqu'à 2 Go. Les objets CLOB sont associés à une page de codes. Les objets texte qui contiennent des caractères mono-octets sont stockés dans une base de données DB2 sous forme d'objets CLOB.

objet DBCLOB : Chaîne de caractères double octets, ou association de caractères mono et double octets, dont la taille peut aller jusqu'à 2 Go. Les objets DBCLOB sont associés à une page de codes. Les objets texte qui contiennent des caractères à double octets sont stockés dans une base de données DB2 sous forme d'objets DBCLOB.

objet de requête : Un objet spécifiant les caractéristiques, leurs valeurs et leurs poids pour une requête QBIC. L'objet peut être nommé et enregistré pour une utilisation ultérieure avec une requête QBIC. S'oppose à chaîne de requête

objet LOB : Séquence d'octets dont la longueur peut aller jusqu'à 2 Go. Un objet LOB peut être de trois types : *objet BLOB*, *objet CLOB* ou *objet DBCLOB*.

orientation objet : En programmation, approche par laquelle tout élément, réel ou abstrait, peut être représenté dans une application sous forme d'objet se composant d'une série d'opérations et de données. Par exemple, un document peut être représenté par un objet document, formé des données qui le caractérisent et des procédures permettant de les manipuler (archivage, envoi, impression, etc.). Une séquence vidéo peut être représentée par un objet vidéo, regroupant les données vidéo ainsi que les opérations susceptibles d'être exécutées sur ces données (lecture, extraction d'une image vidéo spécifique, etc.).

partition de base de données : Partie de la base de données regroupant les propres données utilisateur de la base, ses index, fichiers de configuration et journaux de transactions. Appelée parfois noeud ou noeud de base de données.

pixel : Dans une image, élément le plus petit pouvant être affiché.

prise de vue : Ensemble des images entre deux changements de plan.

recherche par contenu d'image (QBIC) : Fonction fournie par l'extension Image qui permet aux utilisateurs de rechercher des images en fonction de leurs caractéristiques visuelles (par exemple, *couleur moyenne* et *texture*).

releveur de coordonnées LOB : Valeur faible (4 octets), stockée dans une variable SQL, qui peut être utilisée dans un programme pour faire référence à un objet LOB plus grand dans une base de données DB2. L'utilisation d'un releveur de coordonnées LOB permet à un utilisateur de manipuler l'objet LOB comme s'il était stocké dans une variable SQL normale, et cela, sans être obligé de déplacer l'objet LOB entre l'application du poste client et le serveur de bases de données.

score : Valeur calculée reflétant le degré de similarité entre les valeurs de caractéristique et celles spécifiées dans une recherche par contenu d'image. Plus le chiffre est élevé, plus la correspondance est importante. Le score est utilisé pour trier les résultats d'une recherche par contenu d'image.

séquence audio : Section de données audio enregistrées.

séquence vidéo : Section de données filmées ou enregistrées sur bande vidéo.

serveur de partitions de base de données : Gère une *partition de bases de données*. Un serveur de partitions de bases de données comprend un gestionnaire de bases de données et l'ensemble

des données et ressources système gérées par ce serveur. Généralement, un serveur de partitions de bases de données est affecté à chaque machine.

storyboard : Synopsis visuelle d'une séquence vidéo. L'extension Vidéo permet d'identifier et de stocker des images représentatives des prises de vue. Ces images peuvent être utilisées pour constituer un storyboard.

table de gestion : Une des tables utilisées par un programme DB2 Extension pour traiter les demandes d'utilisateurs à propos d'objets constitués de vidéo, de son et d'image. Certaines tables de gestion identifient les tables et colonnes utilisateur activées pour une extension. D'autres contiennent des informations relatives aux attributs d'objets figurant dans des colonnes activées. Synonyme de *table de métadonnées*.

table de métadonnées : Voir *table de gestion*.

téraoctet : Un trillion (10^{12}) d'octets. (dix puissance douze). Lorsque l'on se réfère à la capacité mémoire, 1 099 511 627 776 octets.

texture : Une des caractéristiques pouvant être utilisées dans une recherche par contenu d'image. Il peut s'agir du grain, du contraste ou de la directionnalité d'une image.

type distinct : Voir *type défini par l'utilisateur*.

type utilisateur (UDT) : Type de données défini par un utilisateur dans DB2. Les types UDT sont utilisés pour différencier un objet LOB d'un autre. On peut imaginer, par exemple, qu'un type UDT soit créé pour les objets image et un autre, pour les objets audio. Bien qu'ils soient stockés sous la forme de BLOB, les objets image et audio sont traités comme des types d'objets distincts des BLOB, et différents les uns des autres.

UDF : Voir *fonction définie par l'utilisateur*.

UDT : Voir *type défini par l'utilisateur*.

variable d'environnement : Variable utilisée pour décrire l'environnement d'exploitation de DB2 Extensions et fournir les valeurs par défaut de cet environnement.

variable de référence à un fichier : Variable de programmation permettant de déplacer un objet LOB dans un fichier situé sur un poste de travail client et à partir de ce type de fichier.

variable SQL : Variable d'un programme d'application à laquelle il peut être fait référence dans des instructions SQL imbriquées. Les variables hôtes constituent le mécanisme primaire de transmission de données entre une base de données et des zones de travail d'un programme d'application.

vidéo : Relatif à la partie d'informations enregistrées pouvant être visualisées.

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W

Comment prendre contact avec IBM

Si votre question est d'ordre technique, étudiez tout d'abord les solutions présentées dans le manuel *Troubleshooting Guide* avant de prendre contact avec le Service clients DB2. Ce manuel indique les informations susceptibles d'aider le Service clients à mieux répondre à vos besoins.

Pour obtenir des informations ou commander des produits DB2 avant de prendre contact avec le Service clients DB2 Universal Database, prenez contact avec votre partenaire commercial IBM.

Aux États-Unis, composez l'un des numéros suivants :

- 1-800-237-5511 pour obtenir le Service clients,
- 1-888-426-4343 pour connaître les options de service disponibles.

Infos produit

Aux États-Unis, composez l'un des numéros ci-après.

- Pour commander des produits ou obtenir des informations générales, composez le 1-800-IBM-CALL (1-800-426-2255) ou 1-800-3IBM-OS2 (1-800-342-6672).
- Pour commander des manuels, composez le 1-800-879-2755.

http://www.ibm.com/software/data/
Les pages DB2 World Wide Web fournissent des informations sur DB2, des descriptions de produit, les programmes de formation et d'autres informations.

http://www.ibm.com/software/data/db2/library/
DB2 Product and Service Technical Library permet d'accéder à des forums Q&A (questions/réponses), d'obtenir des correctifs et les dernières informations techniques sur DB2.

Remarque : (Il est possible que ces informations ne soient disponibles qu'en anglais.)

http://www.elink.ibmlink.ibm.com/pbl/pbl/
Le site Web de commande internationale de manuels fournit les informations correspondantes.

http://www.ibm.com/education/certify/
Le programme Professional Certification Program du site Web IBM fournit des informations sur les tests de certification concernant différents produits IBM, dont DB2.

ftp.software.ibm.com
Établissez une connexion anonyme. Des démonstrations, des correctifs, des informations et des outils associés à DB2 ou à des produits connexes sont disponibles dans le répertoire /ps/products/db2.

comp.databases.ibm-db2, bit.listserv.db2-l
Ces newsgroups sont accessibles à tous ceux qui souhaitent partager leurs expériences sur les produits DB2.

Sur Compuserve : GO IBMDB2
Exécutez cette commande pour accéder aux forums IBM DB2. Tous les produits DB2 sont pris en charge sur ces forums.

En dehors des Etats-Unis, pour savoir comment prendre contact avec IBM, consultez l'annexe A du manuel *IBM Software Support Handbook*. Pour accéder à ce document, allez sur le site Web : http://www.ibm.com/support/, puis effectuez une recherche sur le mot clé «handbook».

Remarque : Dans certains pays, les distributeurs agréés peuvent contacter leur centre d'assistance au lieu de prendre contact avec le centre de support IBM.

SC11-1682-00