IBM® DB2® Universal Database

XML Extender - Gestione e programmazione

Versione 7

IBM® DB2® Universal Database

XML Extender - Gestione e programmazione

Versione 7

Prima di utilizzare questo prodotto e le relative informazioni, consultare la sezione “Informazioni particolari” a pagina 243.

Indice

Tabelle

Informazioni relative alla pubblicazione

Questa sezione descrive le seguenti informazioni:

A chi è rivolta questa pubblicazione

Questa pubblicazione è rivolta ai seguenti utenti:

- Gli utenti che utilizzano dati XML nelle applicazioni DB2 e che sono esperti dei concetti XML. E' necessario che i lettori di questa documentazione abbiano una generale conoscenza di XML e DB2. Per ulteriori informazioni su XML e gli argomenti correlati, fare riferimento al seguente sito Web:

  http://www.w3c.org/XML

  Per ulteriori informazioni su DB2, fare riferimento al seguente sito Web:

  http://www.ibm.com/software/data/db2/library
- I responsabili di database DB2 che hanno conoscenza dei concetti, degli strumenti e delle tecniche di gestione DB2.
- I programmatori di applicazioni DB2 che sono esperti dell'SQL e di uno o più linguaggi di programmazione che possono essere utilizzati per le applicazioni DB2.

Come ottenere la versione corrente di questa pubblicazione

E' possibile richiedere la versione aggiornata di questa pubblicazione al sito Web di XML Extender:

http://www.ibm.com/software/data/db2/extenders/xmlext/library.html

Come utilizzare questa pubblicazione

La struttura di questa pubblicazione è la seguente:

Sezione 1. Introduzione
L'introduzione fornisce una panoramica di XML Extender e delle relative modalità di utilizzo nelle applicazioni aziendali. Inoltre contiene un scenario introduttivo che supporta l'utente nell'impostazione ed esecuzione del prodotto.

Sezione 2. Gestione
Questa sezione illustra come preparare e gestire un database DB2 per i dati XML. Leggere questa sezione se si deve gestire un database DB2 che contiene dati XML.

Sezione 3. Programmazione
Questa sezione illustra le modalità di gestione dei dati XML. Leggere questa sezione se si deve accedere e gestire dati XML in un programma DB2.

Sezione 4. Riferimento

Questa sezione illustra l'utilizzo dei comandi di gestione XML Extender, i tipi definiti dall'utente, le funzioni definite dall'utente e le procedure memorizzate. Inoltre elenca i messaggi e i codici relativi a XML Extender. Consultare questa sezione se si è esperti dei concetti e delle attività XML Extender ma si intendere accedere alle informazioni relative a un determinato UDT, UDF, comando, messaggio, tabella metadati o codice.

Sezione 5. Appendici

Le appendici descrivono il DTD relativo alla definizione di accesso documento, esempi vari e quelli relativi allo scenario di introduzione e altri prodotti XML IBM.

Novità presenti in questo manuale per DB2 Versione UDB Fixpak 2

Il presente documento contiene informazioni nuove o aggiornate relative ai seguenti argomenti:

- Impostazione e avvio del wizard di gestione
- Modifica dei documenti XML mediante UDF Update
- Restituzione dei risultati da parte di UDF Clob XML
- Come incrementare i limiti CLOB per procedure memorizzate
- Modifiche dei requisiti DAD:
  - Inclusione delle istruzioni di elaborazione dei fogli di stile nella composizione di nuovi documenti XML
  - Utilizzo di una condizione mancante o vuota per il primo nodo RDB quando è specificata una sola tabella.
- Come emettere indicatori di parametro con JDBC
- Gestione di code page:
  - Dichiarazioni di codifica legale
  - Presupposti di conversione
  - Scenari di conversione
- Appendice sui limiti dei parametri XML Extender

Le informazioni modificate sono evidenziate da una barra verticale, "|".

Fare riferimento al file readme per tutte le informazioni relative alle correzioni per questo fixpak.

Consultare la pagina Support del sito web XML Extender per le domande più frequenti (FAQ) e per l'elenco dei problemi, nonché per ulteriori informazioni sulle correzioni e sulle soluzioni ai problemi comuni:
http://www.ibm.com/software/data/db2/extenders/xmlext/support.html

Inclusione di questo manuale nel Centro informazioni di DB2 UDB Versione 7

La pubblicazione XML Extender può essere inclusa nel Centro informazioni DB2 effettuando le seguenti operazioni:

- Copiare i file HTML di questo manuale dal sottoindirizzario `DOC` della propria lingua del prodotto XML nel sottoindirizzario della pubblicazione DB2 UDB della propria lingua:

  Per Windows, l'indirizzario del Centro informazioni è `sqllib\doc\html\db2sx`

Per UNIX, l'indirizzario del Centro informazioni è *indirizzario di installazione*/doc/html/db2sx

- Riavviare il Centro informazioni e questo manuale sarà incluso nel separatore **Manuali**.

Convenzioni di evidenziazione

Questa pubblicazione utilizza le seguenti convenzioni:

Grassetto	Il testo in grassetto indica: • Comandi • Nomi di campi • Nomi di menu • Pulsanti
Corsivo	Il testo in corsivo indica: • I parametri delle variabili che devono essere sostituiti da un valore • Parole rilevanti • Primo utilizzo di un termine di glossario
MAIUSCOLO	Le lettere in maiuscolo indicano: • Tipi di dati • Nomi di colonna • Nomi di tabella
`Esempio`	Il testo di esempio indica: • Messaggi di sistema • Valori immessi • Codifica esempi • Nomi di indirizzari • Nomi d file • Nomi percorso

Come leggere i diagrammi di sintassi

In questa pubblicazione, la sintassi dei comandi e delle istruzioni SQL è descritta utilizzando i diagrammi di sintassi.

Leggere i diagrammi di sintassi come segue:

- Leggere i diagrammi di sintassi da sinistra a destra, dall'alto verso il basso seguendo la direzione della riga.

  Il simbolo ▸▸—— indica l'inizio di un'istruzione.

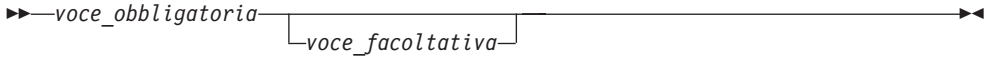
  Il simbolo ——▸ indica che la sintassi dell'istruzione continua sulla riga successiva.

  Il simbolo ▸—— indica che un'istruzione continua dalla riga precedente.

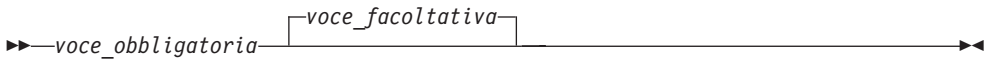
  Il simbolo ——▸◂ indica la fine dell'istruzione.

  I diagrammi delle unità sintattiche diverse dalle istruzioni complete iniziano con il simbolo ▸—— e terminano con il simbolo ——▸.
- Le voci obbligatorie vengono visualizzate sulla riga orizzontale (il percorso principale).

- Le voci facoltative vengono visualizzate sotto il percorso principale.

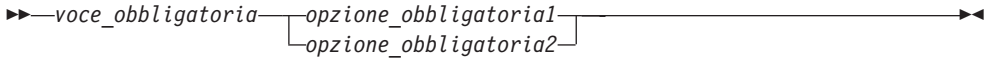


Se nel percorso principale viene visualizzata una voce facoltativa, tale voce non ha alcun effetto sull'esecuzione dell'istruzione ed è utilizzata solo per consentirne la lettura.

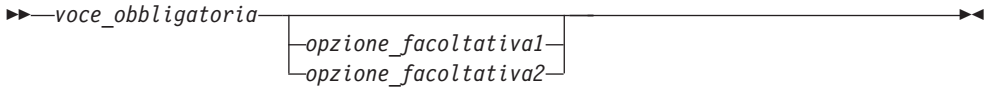


- Se si presenta l'alternativa di due o più voci, tali voci vengono visualizzate in gruppo verticalmente.

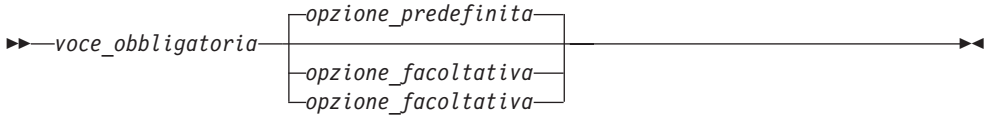
  Se è *necessario* scegliere una determinata voce tra quelle comprese nel gruppo, tale voce viene visualizzata nel percorso principale.



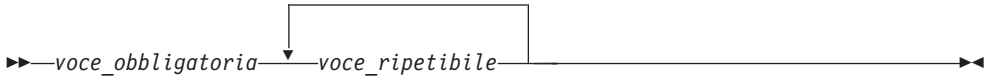
Se tutte le voci sono facoltative, l'intero gruppo viene visualizzato al di sotto del percorso principale.



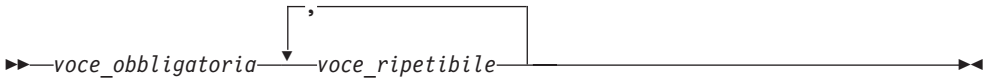
La voce che rappresenta il valore predefinito viene visualizzata al di sopra del percorso principale e le altre opzioni al di sotto di esso.



- Un freccia rivolta verso sinistra, al di sopra della riga principale, indica una voce che può essere ripetuta.



Se la freccia contiene segni di punteggiatura, è necessario separare le voci ripetute con il segno di punteggiatura specificato.



Una freccia di ripetizione posta al di sopra di un gruppo di voci indica che è possibile ripetere le voci del gruppo.

- Le parole chiave appaiono in maiuscolo (ad esempio, `FROM`). In XML Extender, le parole chiave possono essere in maiuscolo o minuscolo. I termini che non sono parole chiave appaiono in minuscolo (ad esempio *nome-colonna*). Nella sintassi, esse rappresentano i nomi o i valori forniti dall'utente.
- Se i segni di punteggiatura, parentesi, operatori matematici o altri simboli simili vengono mostrati, sarà necessario immetterli come parte della sintassi.

Informazioni correlate

Consultare le seguenti documentazioni quando si utilizzano XML Extender e i prodotti correlati:

Documento	Numero ordine	Descrizione
Call Level Interface Guide and Reference	SC09–2950	Questa pubblicazione descrive come scrivere le applicazioni utilizzando CLI per accedere ai server DB2.
DB2 Application Development Guide	SC09–2949	Questa pubblicazione illustra il processo di sviluppo applicazioni e le modalità di codifica, compilazione ed esecuzione dei programmi applicativi che utilizzano API ed SQL integrate per accedere al database.
DB2 Extender page	N/D	Questa pagina contiene informazioni sugli extender DB2 e sulle relative tecnologie. L'indirizzo Web della pagina DB2 Extenders: http://www.software.ibm.com/data/db2/extenders
DB2 SQL Reference for Universal Database Parts 1 and 2	• Parte 1: SC09–2974 • Parte 2: SC09–2975	Queste pubblicazioni descrivono la sintassi, la semantica e le regole del linguaggio SQL. Inoltre, include le informazioni sulle incompatibilità tra release, i limiti del prodotto e le viste di catalogo.
• *DB2 Universal Database Administration Guide: Implementation* • *DB2 Universal Database Administration Guide: Performance* • *DB2 Universal Database Administration Guide: Planning*	• Implementazione: SC09–2944 • Prestazioni: SC09–2945 • Pianificazione: SC09–2946	Queste pubblicazioni illustrano come progettare, implementare e gestire un database DB2.

Documento	Numero ordine	Descrizione
DB2 Universal Database Image, Audio, and Video Extenders Administration and Programming	SC26–9929	Questa pubblicazione descrive come gestire un database DB2 per i dati video, immagine o audio. Inoltre descrive l'utilizzo delle API (application programming interface) fornite dagli extender per accedere e gestire questi tipi di dati.
DB2 Universal Database Text Extender Administration and Programming	SC26–9930	Questa pubblicazione illustra come gestire un database DB2 per i dati di testo. Inoltre descrive l'utilizzo delle API (application programming interface) fornite dagli extender per accedere e gestire questi tipi di dati.

Parte 1. Introduzione

L'introduzione fornisce una panoramica di XML Extender e delle relative modalità di utilizzo nelle applicazioni aziendali.

Capitolo 1. Introduzione a XML Extender

La famiglia IBM® DB2® Extenders™ fornisce soluzioni di gestione dati metadati per la gestione di dati tradizionali e non tradizionali. XML Extender consente di integrare le potenzialità di IBM DB2 Universal Database (DB2 UDB)™ con la flessibilità di XML.

XML Extender DB2 consente di memorizzare e accedere ai documenti XML, di generare documenti XML dai documenti XML scomposti e di dati relazionali esistenti (scomposizione, memorizzazione del contenuto di elementi o attributi privi di tag) in dati relazionali. XML Extender fornisce nuovi tipi di dati, funzioni e procedure memorizzate per la gestione di dati XML in DB2.

XML Extender è disponibile per i seguenti sistemi operativi:

- Windows NT
- AIX
- Sun Solaris
- Linux
- NUMA-Q

Documenti XML

Il mercato informatico offre numerose applicazioni, ciascuna con le proprie caratteristiche positive e meno positive. L'utente può scegliere l'applicazione che esegue specifiche funzioni in base alle proprie esigenze. Tuttavia poiché gli utenti tendono a utilizzare dati condivisi tra le diverse applicazioni, spesso necessitano di eseguire funzioni di replica, trasformazione, esportazione, o salvataggio dei dati in un formato diverso per poterli importare in altre applicazioni. Questo può rappresentare un problema rilevante nelle applicazioni aziendali, poiché quando si utilizzano i processi di trasformazione, spesso si verifica una perdita di dati oppure si richiedono all'utente procedure elaborate per verificare la congruenza dei dati. Queste operazioni comportano un notevole dispendio di tempo e denaro.

Uno dei metodi con cui gli sviluppatori di applicazioni stanno cercando di ovviare a tali inconvenienti è scrivere applicazioni *ODBC (Open Database Connectivity)* per salvare i dati in un DMS. In tal modo i dati possono essere gestiti e presentati nel formato richiesto dalle altre applicazioni. E' necessario che le applicazioni database siano scritte per supportare la conversione dei dati nel formato richiesto da un'applicazione. Tuttavia le applicazioni cambiano continuamente e diventano spesso obsolete. Le applicazioni che convertono i dati in formato HTML forniscono soluzioni di presentazione, ma in pratica i dati presentati non possono essere utilizzati per altri scopi. Sarebbe in tal caso necessario utilizzare un altro metodo che separa i dati dalla presentazione e che pertanto rappresenta una forma pratica di interscambio tra le applicazioni.

XML dispone di funzioni per la risoluzione di questo problema. XML è un acronimo di *eXtensible Markup Language*. E' estensibile poiché utilizza un metalinguaggio che consente di creare il proprio linguaggio specifico per le esigenze dell'azienda. E' possibile utilizzare XML non solo per la cattura dei dati di una specifica applicazione, ma anche per la struttura dei dati. XML non rappresenta l'unico formato di interscambio. Tuttavia XML è stato riconosciuto

come standard accettato di interscambio dei dati. La conformità con tale standard assicura la condivisione dei dati tra applicazioni senza necessità di trasformarli utilizzando i formati specifici.

Applicazioni XML

Poiché XML rappresenta attualmente lo standard accettato per l'interscambio dei dati, molte applicazioni potranno utilizzarlo in modo conveniente.

Si supponga, ad esempio di utilizzare una specifica applicazione di gestione progetti e di dover condividere alcuni dei dati con l'applicazione programmata. XML consente di effettuare facilmente tale operazione. Nell'attuale società universale, è necessario per i produttori di applicazioni fornire programmi di utilità di interscambio XML incorporati nelle applicazioni. Pertanto, nell'esempi, sarà possibile esportare attività dall'applicazione di gestione progetti in XML, quindi reimportarle nell'applicazione programmata (se le informazioni risultano conformi a un DTD accettato).

XML e DB2

Anche se XML risolve numerosi problemi fornendo un formato standard per l'interscambio dei dati, alcuni problemi persistono. Quando si crea un'applicazione di dati aziendali, è necessario considerare le seguenti condizioni:

- La frequenza con cui si esegue la replica dei dati.
- Il tipo di informazioni che si desidera condividere tra le applicazioni.
- La velocità di ricerca delle informazioni necessarie.
- Le modalità con cui è possibile effettuare un'operazione specifica, quale l'aggiunta di una nuova voce o il trigger di un interscambio di dati automatico.

Questo tipo di problemi può essere gestito solo da un DBMS (database management system). Incorporando le informazioni e le metainformazioni XML direttamente nel database, è possibile ottenere rapidamente i risultati XML necessari a tutte le altre applicazioni per esigenze specifiche. In tal caso risulta molto efficace l'utilizzo di XML Extender. XML Extender consente di sfruttare le elevate potenzialità di DB2 in molte applicazioni XML.

Con il contenuto dei documenti strutturati XML in un database DB2, è possibile combinare informazioni strutturate XML con i dati relazionali tradizionali. In base all'applicazione utilizzata, è possibile scegliere se memorizzare tutti i documenti XML in DB2 come tipo di UDF non tradizionale oppure è possibile associare il contenuto XML come dati tradizionali nelle tabelle relazionali. Per i tipi di dati non tradizionali XML, XML Extender aggiunge le potenzialità di ricerca dei tipi di dati complessi dell'elemento o dei valori di attributi XML, in aggiunta alla ricerca di testo strutturale fornita da DB2 UDB Text Extender.

XML Extender consente inoltre di effettuare le seguenti operazioni:

- Memorizzare documenti interi XML come *dati di colonna* nella tabella di un'applicazione o esternamente come file locali, durante l'estrazione degli elementi o degli attributi di colonna XML desiderati in *tabelle laterali* per la ricerca. Utilizzando il metodo di colonna XML, è possibile effettuare quanto segue:
  - Eseguire ricerche rapide sugli elementi o gli attributi XML di *tipi di dati* generici SQL estratti nelle tabelle laterali e indicizzati
  - Aggiornare il contenuto di un *elemento XML* o il valore di un *attributo XML*

  - Estrarre elementi o attributi XML in modo dinamico utilizzando interrogazioni SQL
  - Convalidare documenti XML durante l'inserimento e l'aggiornamento
  - Eseguire ricerche di testo strutturale con Text Extender
- Comporre o scomporre il contenuto di documenti XML con una o più tabelle relazionali, utilizzando un metodo di accesso e memoria di raccolta XML

Integrazione di XML in DB2

XML Extender fornisce le seguenti funzioni per consentire una gestione e una consultazione facilitata dei dati XML con DB2:

- Strumenti di gestione per consentire un'integrazione facilitata dei dati XML nelle tabelle relazionali
- Metodi di utilizzo e memorizzazione per i dati XML.
- Un magazzino DTD per la memorizzazione dei DTD utilizzati per convalidare i dati XML: DTD_REF
- Uno schema di associazione denominato file DAD (Document Access Definition) per associare documenti XML a dati relazionali

Strumenti di gestione

Gli strumenti di gestione XML Extender consentono di abilitare il database e le colonne di tabelle per XML e di associare i dati XML alle strutture relazionali DB2. XML Extender fornisce diversi strumenti di gestione che variano a seconda delle attività che si desidera eseguire, quali lo sviluppo delle applicazioni, l'esecuzione delle attività di gestione o l'utilizzo di un wizard. E' possibile utilizzare i seguenti strumenti per completare le attività di gestione per XML Extender:

- I wizard di gestione XML Extender forniscono un'interfaccia utente grafica per le attività di gestione.
- Il comando **dxxadm** fornisce un'opzione di riga comandi per le attività di gestione.
- Le procedure memorizzate di gestione XML Extender forniscono opzioni di sviluppo applicazioni per le attività di gestione.

Metodi di accesso e memorizzazione

XML Extender fornisce due metodi di accesso e memorizzazione per l'integrazione di documenti XML in DB2: colonna XML e raccolta XML. Questi metodi dispongono di varie funzionalità, ma possono essere utilizzati nella stessa applicazione.

colonna XML
: Questo metodo consente la memorizzazione di documenti XML integri in DB2. La colonna XML consente di archiviare i dati in modo ottimale. I documenti vengono inseriti nelle colonne abilitate per XML e potranno essere aggiornati, richiamati e ricercati. I dati di elementi e attributi possono essere associati alle tabelle DB2 (tabelle laterali), che a loro volta potranno essere indicizzate per rapide ricerche strutturali.

Raccolta XML
: Questo metodo consente di associare strutture di documenti XML a tabelle DB2 in modo da comporre documenti XML per dati DB2 esistenti, o scomporre (memorizzare il contenuto di elementi o attributi privi di tag) documenti XML in dati DB2. Questo metodo risulta efficace per le

applicazioni di interscambio di dati, in particolare quando il contenuto dei documenti XML viene aggiornato frequentemente.

Magazzino DTD

XML Extender fornisce un *magazzino DTD (Document Type Definition)* XML che corrisponde a una serie di dichiarazioni relative agli elementi e agli attributi XML. Quando un database è *abilitato* per XML, viene creata una *tabella di riferimento DTD (DTD_REF)*. Ciascuna riga di questa tabella rappresenta una DTD con informazioni sui metadati aggiuntive. Gli utenti possono accedere a questa tabella per inserire le relative DTD. Le *DTD* della tabella DTD_REF vengono utilizzate per convalidare documenti XML.

DAD (Document Access Definition)

Specificare la struttura dei documenti XML per definire la relativa gestione *DAD (document access definition)*. Il DAD è un documento con formato XML. Consente di associare la struttura dei documenti XML a un database DB2 quando si utilizzano le colonne XML o le raccolte XML. La struttura di un DAD è diversa quando si definisce una colonna XML rispetto a una raccolta XML.

I file DAD vengono gestiti utilizzando la tabella XML_USAGE, che viene creata quando si abilita un database per XML.

Colonna XML: memorizzazione e richiamo dei documenti strutturati

Poiché XML contiene tutte le informazioni necessarie per la creazione di una serie di documenti, potrebbe essere necessario memorizzare e mantenere la struttura attuale del documento.

Per una società di pubblicazione di notizie, ad esempio, che pubblica articoli sul Web, potrebbe essere necessario archiviare gli articoli pubblicati. In questo tipo di scenario, XML Extender consente di memorizzare tutti o parte degli articoli XML in una colonna di una tabella DB2. Questo tipo di memoria dei documenti XML viene denominato *colonna XML*, come mostrato in Figura 1.

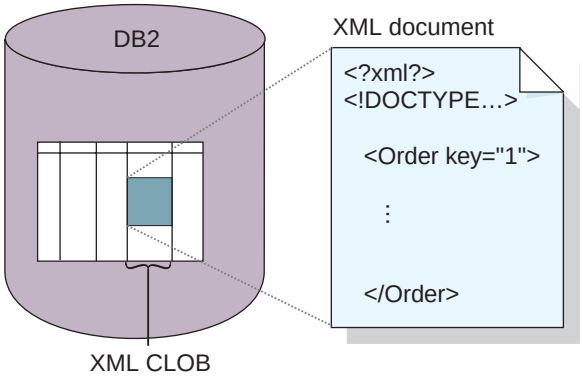


Figura 1. Memorizzazione dei documenti XML strutturati in una colonna di tabella DB2

XML Extender fornisce i seguenti UDT (user-defined type) per l'utilizzo con le colonne XML:

- XMLVarchar
- XMLCLOB

- XMLFILE

Tutti gli UDT XML Extender presentano il prefisso db2xml, che rappresenta il *nome schema* degli UDT XML Extender DB2. Questi tipi di dati vengono utilizzati per identificare il tipo di memorizzazione dei documenti XML nella tabella dell'applicazione. XML Extender supporta i file flat; non è richiesta la memorizzazione dei documenti XML in DB2. E' inoltre possibile memorizzare documenti XML come file sul *file system locale*, come specificato nel nome file locale.

DB2 XML Extender fornisce *UDF (user-defined function)* di alte prestazioni per la memorizzazione e il richiamo di documenti XML nelle colonne XML e per l'estrazione di valori di attributi o elementi XML. L'UDF è una funzione definita nel sistema di gestione del database a cui è possibile far riferimento nelle interrogazioni SQL. XML Extender fornisce i seguenti tipi di UDF:

- UDF di memorizzazione: Memorizza i documenti XML integri in colonne abilitate XML per i tipi di dati XML
- UDF di estrazione: Estrae i documenti XML o i valori degli elementi e degli attributi specificati come tipi di dati di base
- UDF di aggiornamento: Aggiorna interi documenti XML o i valori degli elementi e degli attributi specificati

Le funzioni di estrazione consentono di eseguire potenti ricerche su tipi di dati SQL generici. E' inoltre possibile utilizzare DB2 UDB Text Extender con XML Extender per eseguire *ricerche complete* o strutturali nel testo dei documenti XML. Questa potente funzione di ricerca può essere utilizzata, ad esempio, per migliorare l'utilizzo di un sito Web su cui viene pubblicata un'elevata quantità di testo, come articoli di giornale o applicazioni *EDI (Electronic Data Interchange)*, contenenti elementi o attributi in cui vengono effettuate frequentemente le ricerche.

Tutte le UDF XML Extender presentano il prefisso db2xml, che rappresenta il nome schema delle UDF XML Extender DB2. Le UDF vengono applicate agli UDT XML e vengono utilizzate per le colonne XML.

Percorso di ubicazione

Un *percorso di ubicazione* è una sequenza di *tag XML* che identifica un elemento o attributo XML. XML Extender utilizza il percorso di ubicazione per identificare la struttura del documento XML, indicando il contesto dell'elemento e dell'attributo. Il percorso con singola barra (/) indica che il contesto è un documento intero. Il percorso di ubicazione viene utilizzato per i seguenti scopi:

- Per identificare gli elementi e gli attributi da estrarre, quando si esegue l'estrazione di UDF
- Per specificare il file di associazione tra un elemento o attributo XML e una colonna DB2 durante la definizione dello schema di indicizzazione nelle colonne DAD per XML
- Per identificare un elemento o un attributo XML quando si utilizza Text Extender per le ricerche strutturali e di testo

La Figura 2 a pagina 8 mostra un esempio di percorso di ubicazione e delle relative relazioni alla struttura del documento XML.

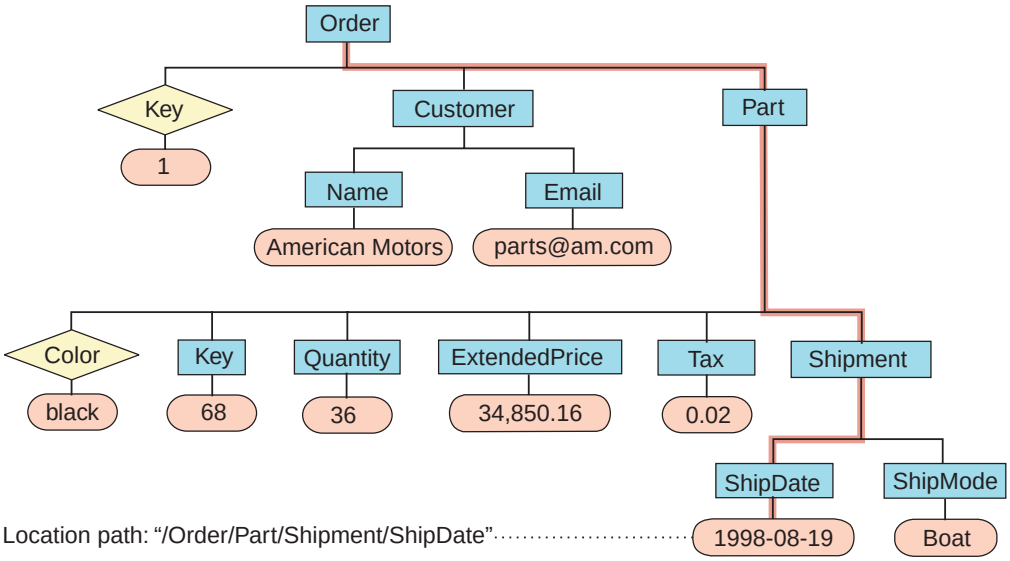


Figura 2. Memorizzazione dei documenti come documenti XML strutturati in una colonna di tabella DB2

Per specificare il percorso di ubicazione, XML Extender utilizza una serie secondaria di *XSLT (XML Stylesheet Language Transformation)* e *XPath (XML Path Language)*. Questo manuale utilizza il termine *percorso di ubicazione*, che è definito nella specifica di XPath. Il percorso di ubicazione è una sequenza di tag XML che identificano un elemento o un attributo XML. In questo manuale si utilizza inoltre la sintassi abbreviata per XSLT o XPath del *percorso di ubicazione assoluto*, specificato nelle specifiche XPath. Il percorso di ubicazione assoluto corrisponde al nome percorso completo di un oggetto.

XSLT è un linguaggio per la trasformazione di documenti XML in altri documenti XML. E' progettato per l'utilizzo come parte di *XSL (XML Stylesheet Language)* che è un linguaggio di fogli di stile per XML. Oltre a XSLT, XSL include un dizionario XML per la specifica del formato. XSL specifica lo stile di un documento XML utilizzando XSLT per descrivere le modalità in cui il documento viene trasformato in un altro documento XML che utilizza il dizionario di formattazione.

XPath è il linguaggio per l'indirizzamento di parti di un documento XML, progettato per l'utilizzo da parte di XSLT. Ciascun percorso di ubicazione può essere espresso utilizzando la sintassi definita per XPath.

Per ulteriori informazioni su XSLT e XPath, consultare le seguenti pagine Web:

- Per XSLT, visitare il sito: http://www.w3.org/TR/WD-xslt
- Per XPath, visitare il sito: http://www.w3.org/TR/xpath

Per la sintassi e le restrizioni, consultare la sezione "Percorso di ubicazione" a pagina 44.

Terminologia della colonna XML

Questa sezione descrive i concetti e la terminologia XML cui si fa riferimento in questo manuale.

DAD (document access definition)
: Per colonne XML, un'associazione della struttura di documenti XML alle tabelle laterali DB2 indicizzate per le interrogazioni strutturali.

DXX_INSTALL
L'indirizzario di installazione XML Extender.

tabella laterale
Tabelle aggiuntive create da XML Extender per migliorare le prestazioni durante le ricerca di elementi o attributi presenti in una colonna XML.

colonna XML
Un metodo di memorizzazione e accesso ai documenti XML, abilitando una colonna DB2 per i tipi di dati XML e memorizzando un documento XML integro nella colonna abilitata. E' incluso il riferimento a una colonna abilitata per XML, utilizzando uno degli strumenti di gestione.

Tabella XML
Una tabella applicativa che include una o più colonne abilitate per XML, utilizzando uno degli strumenti di gestione.

XML UDF
Una UDF DB2 fornita da XML Extender.

UDT XML
Un UDT DB2 fornita da XML Extender.

Raccolta XML: raccolta dati integrati

I dati SQL tradizionali vengono *scomposti* dai documenti XML in entrata oppure vengono utilizzati per *comporre* documenti XML in uscita. Se si desidera condividere i dati con altre applicazioni, potrebbe essere necessario comporre e scomporre documenti XML in entrata e in uscita e gestire i dati per sfruttare al meglio le funzionalità relazionali di DB2. Questo tipo di memoria dei documenti XML viene denominato *raccolta XML*.

Un esempio di raccolta XML è presente in Figura 3.

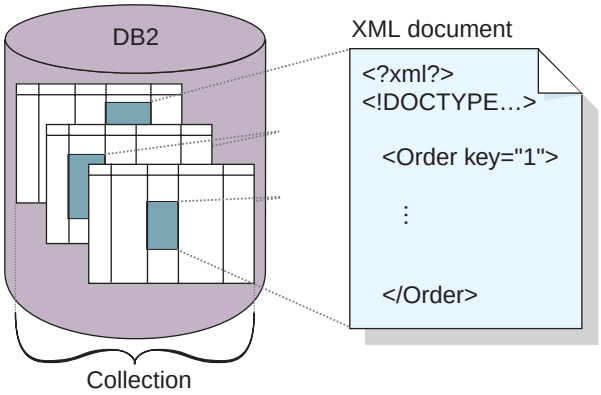


Figura 3. Memorizzazione dei documenti come dati privi di tag nelle tabelle DB2

La raccolta XML è definita in un file DAD, che specifica le modalità di associazione degli elementi e degli attributi a una o più tabelle relazionali. E' possibile definire il nome di una raccolta eseguendone l'abilitazione, quindi utilizzandola con le procedure memorizzate per comporre o scomporre documenti XML.

Quando si definisce una raccolta nel file DAD, è possibile utilizzare uno dei due tipi di schemi di associazione, associazione SQL e associazione RDB_node. L'associazione SQL utilizza istruzioni SQL SELECT per definire le tabelle e le condizioni utilizzate da DB2 per la raccolta. L'associazione RDB_node utilizza un RDB_node basato su XPath per definire le tabelle, le colonne e le condizioni.

Vengono fornite procedure memorizzate per la composizione o la scomposizione di documenti XML. Le procedure memorizzate utilizzano il prefisso db2xml, che corrisponde al *nome schema* di XML Extender. Utilizzare le seguenti procedure memorizzate con le raccolte XML:

- Composizione:
  - dxxGenXML(): utilizza un file DAD per una raccolta XML per la composizione di documenti XML
  - dxxRetrieveXML(): utilizza una raccolta XML abilitata per la composizione di documenti XML
- Scomposizione:
  - dxxShredXML(): utilizza un file DAD per una raccolta XML per la scomposizione di documenti XML
  - dxxInsertXML(): utilizza una raccolta XML abilitata per la scomposizione di documenti XML

Terminologia della raccolta XML

I seguenti termini sono univoci in XML Extender e vengono utilizzati frequentemente in questo manuale.

composizione
: Generazione di documenti XML da dati relazionali esistenti, in base alla definizione del file DAD.

scomposizione
: Memorizzazione di documenti come dati privi di tag, relazionali, in base alla definizione del file DAD.

DAD (document access definition)
: Per raccolte XML, un'associazione di strutture di documenti XML alle strutture di dati DB2 per la composizione e la scomposizione di documenti XML.

DXX_INSTALL
: L'indirizzario di installazione XML Extender.

Raccolta XML
: Un metodo di memorizzazione e accesso ai dati XML utilizzando una serie di tabelle relazionali. I dati privi di tag possono essere composti in documenti XML oppure possono essere scomposti dai documenti XML. Comprendono inoltre la serie di tabelle in cui vengono composti o da cui vengono scomposti i documenti XML.

procedure memorizzate XML
: Procedure memorizzate per la composizione o la scomposizione di documenti XML.

Capitolo 2. Introduzione a XML Extender

Questo capitolo illustra l'utilizzo di XML Extender per l'accesso e la modifica dei dati XML. Seguendo le sezioni del supporto didattico, è possibile impostare un database utilizzando i dati di esempio, associare i dati SQL a un documento XML, memorizzare documenti XML nel database e quindi ricercare ed estrarre dati dai documenti XML.

Nelle lezioni di gestione, si utilizza la finestra Comandi DB2 con i comandi di gestione. E' possibile eseguire tali attività anche con il wizard di gestione di XML Extender descritto in questa pubblicazione. Nelle lezioni per la gestione dati XML, si utilizzano le procedure memorizzate e le funzioni UDF fornite da XML Extender. La maggior parte degli esempi contenuti nella pubblicazione si basano sui dati di esempio utilizzati in questo capitolo.

Obbligatorio: per completare le lezioni di questo capitolo, è necessario disporre di DB2 UDB Versione 6.1 o successive. Se si utilizza la Versione 6.1, è necessario installare Fixpak 2. Inoltre, i passi riportati in queste lezioni presumono che si utilizzi Windows NT®.

Le lezioni sono le seguenti:

- Memorizzare un documento XML integro in una colonna della tabella DB2
  - Pianificare gli UDT XML in cui memorizzare il documento, gli attributi e gli elementi XML da ricercare frequentemente.
  - Impostare il database e le tabelle
  - Abilitare il database per XML
  - Inserire la DTD nella tabella del magazzino DTD
  - Preparare una DAD per una colonna XML
  - Aggiungere una colonna in una tabella esistente di tipo XML
  - Abilitare la nuova colonna per XML
  - Creare indici nelle tabelle laterali
  - Memorizzare un documento XML nella colonna XML
  - Ricercare la colonna XML utilizzando le funzioni UDF di XML Extender
- Creare un documento XML dai dati esistenti
  - Pianificare la struttura del documento XML
  - Impostare il database e le tabelle
  - Abilitare il database per XML
  - Preparare un file DAD (document access definition) per una raccolta XML
  - Creare il documento XML dai dati esistenti
  - Richiamare il documento XML dal database
- Cancellazione dati dal database

Scenario delle lezioni

Le seguenti lezioni sono dirette a un utente impiegato della ACME Auto Direct, una società che distribuisce macchine e autocarri ai rivenditori auto. All'utente viene affidato il compito di eseguire due attività. Un'attività di impostazione di un sistema in cui è possibile archiviare gli ordini nel database SALES_DB per le

interrogazioni da parte del settore vendite. La seconda attività consiste nella raccolta delle informazioni in un database per ordini di acquisto esistente, SALES_DB, e nell'estrazione delle informazioni chiave da memorizzare nei documenti XML.

Lezione: Memorizzare un documento XML in una colonna XML

Scenario

L'attività da eseguire è l'archiviazione dei dati vendita per il settore assistenza clienti. I dati vengono memorizzati nei documenti XML che utilizzano la stessa DTD. Il settore assistenza clienti utilizzerà questi documenti XML per rispondere alle richieste e ai commenti dei clienti.

Il settore assistenza clienti fornisce una struttura standard per i documenti XML e specifica i dati degli elementi per i quali verranno effettuate interrogazioni più frequenti. E' necessario che i documenti XML memorizzati nella tabella SALES_TAB siano memorizzati nel database SALES_DB in modo da ricercarli rapidamente. La tabella SALES_DB contiene due colonne con i dati relativi a ciascuna vendita e una terza colonna in cui contenere il documento XML. Questa colonna è denominata ORDER.

I compiti dell'utente saranno quelli di determinare i tipi di dati XML, in cui memorizzare il documento XML, gli attributi e gli elementi XML per i quali verranno eseguite interrogazioni più frequenti. Successivamente, occorre impostare il database SALES_DB per XML, creare la tabella SALES_TAB e abilitare la colonna ORDER in modo da memorizzare il documento integro in DB2 e infine inserire una DTD per la convalida del documento XML che verrà memorizzato come tipo di dati XMLVARCHAR. Una volta abilitata la colonna, occorre definire le tabelle laterali per le quali creare un indice per la ricerca strutturale del documento in un file DAD (document access definition) e un documento XML che specifica la struttura delle tabelle laterali. Per visualizzare gli esempi del file DAD e il documento XML consultare l'"Appendice B. Esempi" a pagina 223.

La tabella SALES_TAB è descritta nella Tabella 1.

Tabella 1. Tabella SALES_TAB

Nome colonna	Tipo di dati
INVOICE_NUM	CHAR(6) NOT NULL PRIMARY KEY
SALES_PERSON	VARCHAR(20)
ORDER	XMLVARCHAR

Pianificazione

Prima di utilizzare XML Extender per la memorizzazione dei documenti, è necessario conoscere la struttura del documento XML in modo da stabilire le modalità di ricerca del documento. Una volta pianificate le modalità di ricerca del documento, è necessario determinare:

- L'UDT XML in cui verrà memorizzato il documento XML
- Gli attributi e gli elementi XML che il settore assistenza clienti ricercherà frequentemente, in modo da creare indici per migliorare le prestazioni.

Le seguenti sezioni descrivono come individuare tali elementi.

La struttura del documento XML

La struttura del documento XML di questa lezione raccoglie le informazioni relative a un ordine specifico strutturato con la chiave ordine come primo livello, quindi il cliente, la parte e le informazioni sulla spedizione come livelli successivi. Il documento XML è descritto nella Figura 4.

Questa lezione inoltre fornisce una DTD di esempio che consente l'interpretazione e la convalida della struttura del documento XML. E' possibile consultare il file DTD nell'"Appendice B. Esempi" a pagina 223. Tale file corrisponde alla struttura nella Figura 4.

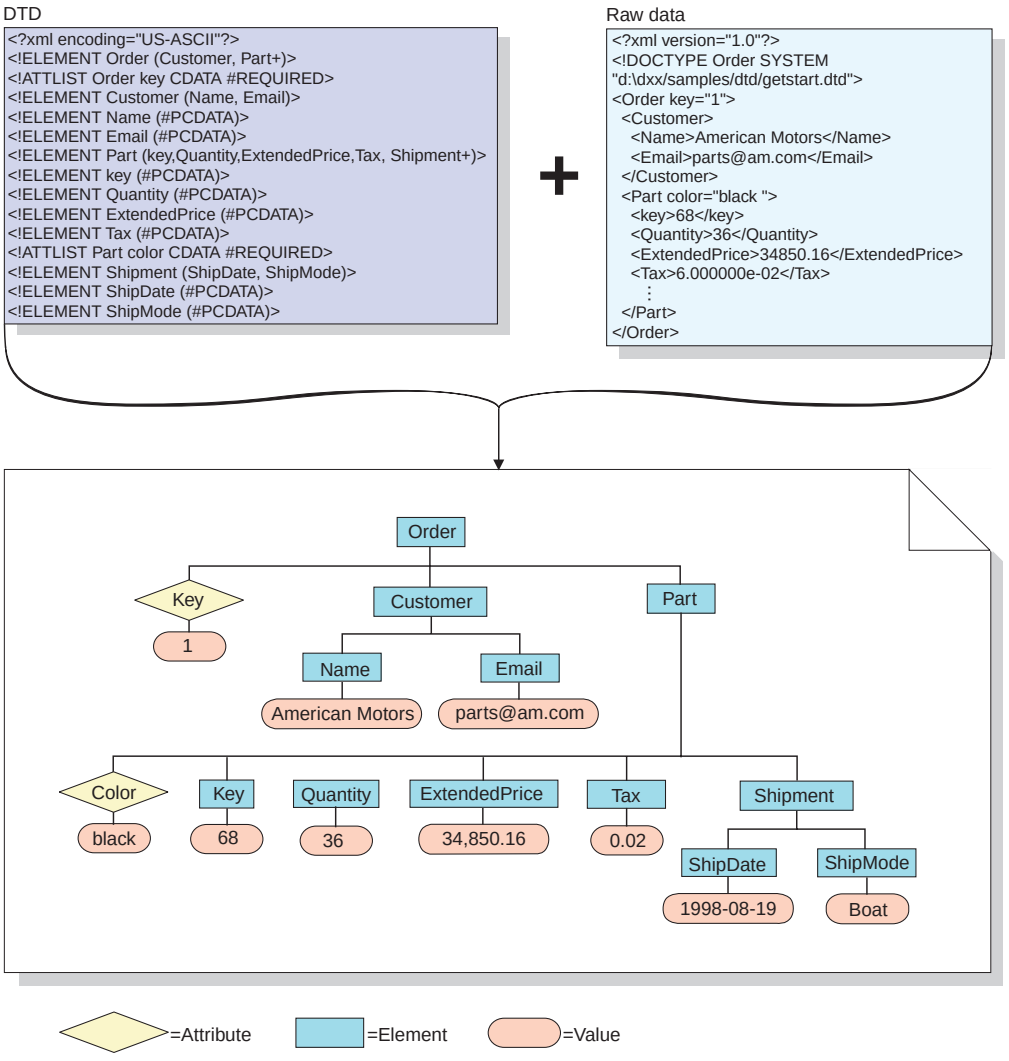


Figura 4. La struttura gerarchica della DTD e del documento XML

Determinazione del tipo di dati XML per la colonna XML

XML Extender fornisce gli UDT XML con cui definire una colonna per conservare i documenti XML. Questi tipi di dati sono:

- XMLVarchar: per i documenti di piccole dimensioni memorizzati in DB2
- XMLCLOB: per i documenti di grandi dimensioni memorizzati in DB2
- XMLFILE: per i documenti memorizzati al di fuori di DB2

In questa lezione, si memorizza un documento di piccole dimensioni in DB2 e si utilizzerà quindi, il tipo di dati XMLVarchar.

Determinazione degli elementi e degli attributi da ricercare

Una volta compresa la struttura del documento XML e i requisiti dell'applicazione, è possibile determinare gli elementi e gli attributi da ricercare, gli elementi e gli attributi che verranno ricercati o estratti frequentemente oppure quelli per i quali l'interrogazione è più complessa. Il settore assistenza clienti ha specificato come interrogazioni più frequenti quelle relative alla chiave ordine, al nome cliente, al prezzo e alla data di spedizione di un ordine e richiede un'esecuzione rapida per tali ricerca. Queste informazioni sono contenute negli elementi e negli attributi della struttura del documento XML. Tabella 2 descrive i percorsi dell'ubicazione di ciascun elemento e attributo.

Tabella 2. Gli elementi e agli attributi da ricercare

Dati	Percorso di ubicazione
chiave ordine	/Order/@key
cliente	/Order/Customer/Name
prezzo	/Order/Part/ExtendedPrice
data spedizione	/Order/Part/Shipment/ShipDate

Associazione del documento XML alle tabelle laterali

In questo supporto didattico, verrà creato un file DAD per la colonna XML utilizzata per memorizzare il documento XML in DB2. Inoltre il contenuto degli attributi e degli elementi XML viene associato alle tabelle laterali DB2 utilizzate per l'indice che migliora l'esecuzione delle ricerche. Nell'ultima sezione, vengono riportati gli attributi e gli elementi da ricercare. In questa sezione, vengono fornite le informazioni complete relative all'associazione dei valori dell'attributo e dell'elemento alle tabelle DB2 che è possibile indicizzare.

Una volta identificati gli elementi e gli attributi da ricercare, stabilire le modalità di organizzazione di tali elementi nelle tabelle laterali, il numero di tabelle e il numero di colonne in ciascuna tabella. Generalmente, le tabelle laterali vengono organizzate immettendo informazioni simili nella stessa tabella. La struttura viene inoltre determinata dalla possibilità di ripetere più volte il percorso di ubicazione di ciascun elemento nel documento. Ad esempio nel documento di esempio, l'elemento della parte può essere ripetuto più volte e, di conseguenza, anche gli elementi data e prezzo possono ricorrere più volte. Gli elementi che ricorrono più volte devono essere presenti nelle relative tabelle.

Inoltre, è necessario determinare i tipi di base DB2 dei valori attributo o elemento da utilizzare. Generalmente il tipo di base DB2 è facilmente rilevabile dal formato dei dati. Se i dati sono rappresentati da testo, scegliere VARCHAR; se i dati sono un valore intero, scegliere INTEGER; se i dati sono costituiti da una data e si desidera effettuare ricerche delimitate, scegliere DATE.

In questo supporto didattico, gli elementi e gli attributi vengono associati alle seguenti tabelle laterali:

ORDER_SIDE_TAB

Nome colonna	Tipo di dati	Percorso di ubicazione	Ricorrenze multiple?
ORDER_KEY	INTEGER	/Order/@key	No
CUSTOMER	VARCHAR(16)	/Order/Customer/Name	No

PART_SIDE_TAB

Nome colonna	Tipo di dati	Percorso di ubicazione	Ricorrenze multiple?
PRICE	DECIMAL(10,2)	/Order/Part/ExtendedPrice	Sì

SHIP_SIDE_TAB

Nome colonna	Tipo di dati	Percorso di ubicazione	Ricorrenze multiple?
DATE	DATE	/Order/Part/Shipment/ShipDate	Sì

In questo supporto didattico, viene fornita una serie di script per consentire l'impostazione dell'ambiente. Questi script si trovano nella directory *DXX_INSTALL*\samples\cmd (dove *DXX_INSTALL* è l'unità e la directory di installazione di XML Extender, ad esempio c:\dxx\samples\cmd) ed essi sono i seguenti:

getstart_db.cmd
Crea il database e compila quattro tabelle.

getstart_prep.cmd
Esegue il bind del database con le procedure memorizzate XML Extender e le CLI DB2.

getstart_insertDTD.cmd
Inserisce la DTD utilizzata per convalidare il documento XML nella colonna XML.

getstart_createTabCol.cmd
Crea una tabella applicativa che avrà una colonna abilitata per XML.

getstart_alterTabCol.cmd
Modifica la tabella applicativa aggiungendo la colonna che sarà abilitata per XML.

getstart_enableCol.cmd
Abilita la colonna XML.

getstart_createIndex.cmd
Crea indici nelle tabelle laterali per la colonna XML.

getstart_insertXML.cmd
Inserisce il documento XML nella colonna XML.

getstart_queryCol.cmd
Esegue un'istruzione select nella tabella applicativa e restituisce il documento XML.

getstart_clean.cmd
Ripulisce l'ambiente del supporto didattico.

Impostazione

In questa sezione viene descritta la preparazione del database per l'utilizzo con XML Extender. Operazione da effettuare:

1. Creare il database.
2. Abilitare il database.

Creazione del database

In questa sezione si utilizza un comando per l'impostazione del database. Questo comando crea un database di esempio, ne stabilisce il collegamento, crea le tabelle in cui memorizzare i dati e quindi inserisce i dati.

Per creare il database:

1. Entrare nella directory *DXX_INSTALL*\samples\cmd, dove *DXX_INSTALL* è l'unità e la directory di installazione di XML Extender. In queste lezioni si utilizza la directory c:\dxx. Modificare tali valori se l'unità e la directory sono diverse.
2. Aprire la finestra Comandi DB2 dal menu Avvio di Windows NT oppure immettere il seguente comando dal prompt dei comandi di Windows NT:

   ```
   DB2CMD
   ```
3. Dalla finestra Comandi DB2, immettere il seguente comando:

   ```
   getstart_db.cmd
   ```

Abilitazione del database

Per memorizzare le informazioni XML nel database, è necessario abilitarlo per XML Extender. Quando si abilita un database per XML, XML Extender effettua le seguenti operazioni:

- Crea tutti i tipi definiti dall'utente (UDT) e le funzioni definite dall'utente (UDF).
- Crea e compila le tabelle di controllo utilizzando i metadati necessari richiesti da XML Extender.
- Crea lo schema db2xml e assegna i privilegi richiesti.

Per abilitare il database per XML:

Dalla finestra Comandi DB2, eseguire il seguente script per abilitare il database SALES_DB:

```
getstart_prep.cmd
```

Questo script esegue il bind del database con le procedure memorizzate XML Extender e le CLI DB2. Inoltre esegue l'opzione del comando **dxxadm** per abilitare il database:

```
dxxadm enable_db SALES_DB
```

Creazione della colonna XML

XML Extender fornisce un metodo per la memorizzazione e l'accesso a tutti i documenti XML contenuti nel database, tale metodo e denominato colonna XML. Utilizzando il metodo della colonna XML, è possibile memorizzare il documento mediante l'utilizzo del tipo XMLFILE, indicizzare la colonna in tabelle laterali e quindi interrogare o ricercare il documento XML. Questo metodo di memorizzazione è particolarmente utile per le applicazioni di archivio in cui i documenti non vengono frequentemente aggiornati. Per eseguire questo supporto didattico. il documento XML fornito verrà memorizzato nella colonna XML.

In questa lezione, il documento verrà memorizzato nella tabella SALES_TAB. Per memorizzare il documento:

1. Inserire la DTD relativa al documento XML nella tabella di riferimento DTD, DTD_REF.
2. Preparare un file DAD che specifichi l'ubicazione del documento XML e le tabelle laterali per la ricerca strutturale.
3. Aggiungere una colonna nella tabella SALES_TAB con UDT XML di tipo XMLVARCHAR.

4. Abilitare la colonna per XML.
5. Indicizzare le tabelle laterali per la ricerca strutturale.
6. Memorizzare il documento utilizzando una UDF fornita da XML Extender.

Memorizzazione della DTD nel magazzino DTD

E' possibile utilizzare una DTD per convalidare i dati XML in una colonna XML. XML Extender crea una tabella nel database abilitato per XML, denominata DTD_REF. La tabella è riconosciuta come riferimento DTD ed è disponibile per consentire la memorizzazione di DTD. Quando si decide di convalidare i documenti XML, è necessario memorizzare la DTD in questo magazzino. La DTD relativa a questo supporto didattico è `c:\dxx\samples\dtd\getstart.dtd`.

Per inserire la DTD:

Dalla finestra Comandi DB2, immettere il seguente comando SQL INSERT, sulla stessa riga:

```
DB2 CONNECT TO SALES_DB
DB2 INSERT into db2xml.dtd_ref values('c:\dxx\samples\dtd\getstart.dtd',
    db2xml.XMLClobFromFile('c:\dxx\samples\dtd\getstart.dtd'), 0, 'user1',
    'user1', 'user1')
```

E' inoltre possibile eseguire il seguente file dei comandi per inserire la DTD:

`getstart_insertDTD.cmd`

Preparazione del file DAD

Il file DAD della colonna XML presenta una struttura semplice. Specificare colonna XML come modalità di memorizzazione e definire le tabelle e le colonne per l'indicizzazione.

Nei seguenti passi, si fa riferimento agli elementi del file DAD come *tag* e agli elementi della struttura del documento XML come *elementi*. Un esempio di un file DAD simile a quello creato è nel percorso `c:\dxx\samples\dad\getstart_xcolumn.dad` il quale presenta alcune differenze dal file generato nei seguenti passi. Se si utilizza tale file per la lezione, è possibile che i percorsi file differiscano da quelli dell'ambiente dell'utente; il valore <validation> è impostato su NO anziché su YES.

Per preparare il file DAD:

1. Aprire un editor di testo e denominare il file `getstart_xcolumn.dad`

   Tenere presente che tutte le tag utilizzate nel file DAD sono sensibili al maiuscolo e minuscolo.
2. Creare l'intestazione DAD, con dichiarazioni XML e Doctype.

   ```
   <?xml version="1.0"?>
   <!DOCTYPE DAD SYSTEM "c:\dxx\dtd\dad.dtd">
   ```

   Il file DAD è un documento XML e richiede dichiarazioni XML.
3. Inserire tag <DAD></DAD> di apertura e di chiusura. Tutte le altre tag sono situate all'interno di queste tag.
4. Inserire tag di apertura e di chiusura <DTDID></DTDID> per specificare l'identificativo ID DTD che associa la DAD alla DTD del documento XML e specifica l'ubicazione DTD sul client.

   ```
   <dtdid>c:\dxx\samples\dtd\getstart.dtd</dtdid>
   ```

   Nella sezione "Memorizzazione della DTD nel magazzino DTD" verificare che questa stringa corrisponda al valore utilizzato come primo valore di

parametro quando si inserisce la DTD nella tabella di riferimento DTD. Ad esempio, il percorso utilizzato per l'ID DTD potrebbe differire dalla stringa precedente se si sta utilizzando un'unità macchina differente.

5. Specificare le tag <validation></validation> di apertura e di chiusura per indicare che XML Extender deve convalidare la struttura del documento XML utilizzando la DTD inserita nella tabella del magazzino DTD.

   ```
   <validation>YES</validation>
   ```

   Il valore di <validation> deve essere espresso in maiuscolo.

6. Inserire le tag <Xcolumn></Xcolumn> di apertura e di chiusura per definire colonna XML come metodo di memorizzazione. Il metodo definisce la memorizzazione dei dati XML in una colonna XML.

   ```
   <Xcolumn>
   </Xcolumn>
   ```

7. Inserire le tag <table></table> di apertura e di chiusura per ciascuna tabella laterale che deve essere creata.

   ```
   <Xcolumn>
        <table name="order_side_tab">
   </table>
        <table name="part_side_tab">
   </table>
        <table name="ship_side_tab">
   </table>
   </Xcolumn>
   ```

8. Inserire le tag <column></column> di apertura e di chiusura per ciascuna colonna da includere nelle tabelle laterali. Ciascuna tag <column> presenta quattro attributi:
   - **name:** il nome della colonna
   - **type:** il tipo di dati della colonna
   - **path:** il percorso di ubicazione dell'elemento corrispondente nel documento XML, con la sintassi XPathx. Per informazioni sulla sintassi del percorso di ubicazione, consultare la sezione "Percorso di ubicazione" a pagina 44.
   - **multi-occurrence:** indica se il percorso di ubicazione dell'elemento può ricorrere più volte nella struttura del documento XML

   ```
   <Xcolumn>
        <table name="order_side_tab">
      <column name="order_key"
              type="integer"
              path="/Order/@key"
              multi_occurrence="NO"/>
      <column name="customer"
              type="varchar(50)"
              path="/Order/Customer/Name"
             multi_occurrence="NO"/>
   </table>
        <table name="part_side_tab">
            <column name="price"
               type="decimal(10,2)"
               path="/Order/Part/ExtendedPrice"
             multi_occurrence="YES"/>
   </table>
        <table name="ship_side_tab">
            <column name="date"
               type="DATE"
               path="/Order/Part/Shipment/ShipDate"
             multi_occurrence="YES"/>
   </table>
   </Xcolumn>
   ```

9. Accertarsi che sia presente una tag </Xcolumn> di chiusura dopo l'ultima tag </table>.
10. Accertarsi che sia presente una tag </DAD> di chiusura dopo la tag </Xcolumn>.
11. Salvare il file come `getstart_xcolumn.dad`.

E' possibile confrontare il file appena creato con il file di esempio, `c:\dxx\samples\dad\getstart_xcolumn.dad`. Questo file è una copia di lavoro del file DAD richiesto per abilitare la colonna XML e per creare le tabelle laterali. Il file di esempio contiene istruzioni di percorso che è possibile modificare in modo da soddisfare i requisiti dell'ambiente utilizzato.

Creazione della tabella SALES_TAB

In questa sezione viene creata la tabella SALES_TAB. Inizialmente, tale tabella presenta due colonne con le informazioni sulla vendita relative all'ordine.

Per creare la tabella:

Dalla finestra Comandi DB2, immettere la seguente istruzione CREATE TABLE:

```
DB2 CONNECT TO SALES_DB

DB2 CREATE TABLE SALES_TAB(INVOICE_NUM CHAR(6) NOT NULL PRIMARY KEY,
         SALES_PERSON VARCHAR(20))
```

In alternativa, per creare la tabella è possibile eseguire il seguente file dei comandi:

```
getstart_createTabCol.cmd
```

Aggiunta della colonna di tipo XML

Aggiungere una nuova colonna nella tabella SALES_TAB. Questa colonna conterrà il documento XML integro creato precedentemente che deve essere di tipo UDT XML. XML Extender fornisce tipi di dati multipli, descritti nel "Capitolo 8. XML Extender - UDT" a pagina 139. In questo supporto didattico, il documento verrà memorizzato come XMLVARCHAR.

Per aggiungere la colonna di tipo XML:

Dalla finestra Comandi DB2, immettere la seguente istruzione SQL:

```
DB2 ALTER TABLE SALES_TAB ADD ORDER DB2XML.XMLVARCHAR
```

In alternativa, per modificare la tabella è possibile eseguire il seguente file dei comandi:

```
getstart_alterTabCol.cmd
```

Abilitazione della colonna XML

Una volta creata la colonna di tipo XML, abilitarla per XML Extender. Quando si abilita la colonna, XML Extender legge il file DAD e crea le tabelle laterali. Prima di abilitare la colonna, è necessario:

- Stabilire se si desidera creare una vista predefinita della colonna XML, che contiene il documento XML, e le colonne della tabella laterale. E' possibile specificare la vista predefinita quando si eseguono interrogazioni sul documento XML. In questa lezione, la vista verrà specificata con il parametro `-v`.
- Stabilire se si desidera specificare una chiave primaria come *ROOT ID*, il nome della colonna della chiave primaria nella tabella applicativa che associa tutte le tabelle laterali alla tabella applicativa. Se non si specifica la chiave primaria, XML Extender aggiunge la colonna DXXROOT_ID alla tabella applicativa e alle tabelle laterali. La colonna ROOT_ID collega le tabelle laterali e quelle

applicative, consentendo a XML Extender di aggiornare automaticamente le tabelle laterali nel caso in cui il documento XML venga aggiornato. In questa lezione, il nome della chiave primaria del comando (INVOICE_NUM) verrà specificato con il parametro `-r`. XML Extender quindi utilizzerà la colonna specificata come ROOT_ID e aggiungerà la colonna alle tabelle laterali.

- Stabilire se specificare un table space oppure utilizzare quello predefinito. In questa lezione, si utilizzerà il table space predefinito.

Per abilitare la colonna per XML:

Dalla finestra Comandi DB2, immettere il seguente comando:

```
dxxadm enable_column SALES_DB SALES_TAB ORDER GETSTART_XCOLUMN.DAD
        -v SALES_ORDER_VIEW -r INVOICE_NUM
```

In alternativa, per abilitare la colonna per XML è possibile eseguire il seguente file dei comandi:

```
getstart_enableCol.cmd
```

XML Extender crea le tabelle laterali con la colonna INVOICE_NUM e crea la vista predefinita.

Importante: non modificare assolutamente le tabelle laterali. E' possibile solo aggiornare il documento XML utilizzando le UDF fornite da XML Extender. XML Extender automaticamente aggiornerà le tabelle laterali quando si aggiorna il documento XML nella colonna XML.

Visualizzazione della colonna e delle tabelle laterali

Quando si abilita la colonna XML, viene creata una vista della colonna XML e delle tabelle laterali. E' possibile utilizzare questa vista quando si lavora con la colonna XML.

Per visualizzare la colonna XML e le colonne delle tabella laterali:

Dalla finestra Comandi DB2, immettere la seguente istruzione SQL SELECT:

```
DB2 SELECT * FROM SALES_ORDER_VIEW
```

La vista riporta le colonne delle tabelle laterali come specificato nel file `getstart_xcolumn.dad`.

Creazione degli indici nelle tabelle laterali

La creazione degli indici nelle tabelle laterali consente di eseguire rapide ricerche strutturali del documento XML. In questo passo, vengono creati gli indici sulle colonne chiave delle tabelle laterali che sono state create quando è stata abilitata la colonna XML, ORDER. Il settore assistenza clienti specifica le colonne nelle quali gli impiegati effettueranno un numero maggiore di interrogazioni. La Tabella 3 descrive queste colonne da indicizzare:

Tabella 3. Colonne delle tabelle laterali da indicizzare

Colonna	Tabella laterale
ORDER_KEY	ORDER_SIDE_TAB
CUSTOMER	ORDER_SIDE_TAB
PRICE	PART_SIDE_TAB
DATE	SHIP_SIDE_TAB

Per indicizzare le tabelle laterali:

Dalla finestra Comandi DB2, immettere ii seguenti comandi SQL:

```
DB2 CREATE INDEX KEY_IDX
      ON ORDER_SIDE_TAB(ORDER_KEY)

DB2 CREATE INDEX CUSTOMER_IDX
      ON ORDER_SIDE_TAB(CUSTOMER)

DB2 CREATE INDEX PRICE_IDX
      ON PART_SIDE_TAB(PRICE)

DB2 CREATE INDEX DATE_IDX
      ON SHIP_SIDE_TAB(DATE)
```

In alternativa, per creare gli indici è possibile eseguire il seguente file dei comandi:

```
getstart_createIndex.cmd
```

Memorizzazione del documento XML

Una volta abilitata una colonna per contenere il documento XML e indicizzato le tabelle laterali, è possibile memorizzare il documento utilizzando le funzioni fornite da XML Extender. Quando si memorizzano i dati in una colonna XML, utilizzare le funzioni cast predefinite o le UDF XML Extender. Poiché si deve memorizzare un oggetto di tipo di base VARCHAR in una colonna XML UDT XMLVARCHAR, è necessario utilizzare le funzioni cast predefinite. Per ulteriori informazioni sulle funzioni cast predefinite di memorizzazione e le UDF fornite da XML Extender consultare la sezione “Memorizzazione dati” a pagina 98.

Per memorizzare il documento XML:

Importante: Aprire il documento XML `c:\dxx\samples\xml\getstart.xml`. Accertarsi che il percorso file in DOCTYPE corrisponda all’ID DTD specificato nel file DAD e durante l’inserimento della DTD nel magazzino DTD. E’ possibile verificare tale corrispondenza eseguendo un’interrogazione della tabella db2xml.DTD_REF e controllando l’elemento DTDID nel file DAD. Se si utilizza un’unità e una struttura di indirizzario differenti da quelle predefinite, è necessario modificare il percorso nella dichiarazione DOCTYPE.

Dalla finestra Comandi DB2, immettere il seguente comando SQL INSERT:

```
DB2 INSERT INTO SALES_TAB (INVOICE_NUM, SALES_PERSON, ORDER) VALUES('123456',
   'Sriram Srinivasan', db2xml.XMLVarcharFromFile('c:\dxx\samples\cmd\getstart.xml'))
```

Quando si memorizza il documento XML, XML Extender automaticamente aggiorna le tabelle laterali.

In alternativa, per memorizzare il documento è possibile eseguire il seguente file dei comandi:

```
getstart_insertXML.cmd
```

Per verificare che le tabelle siano stato aggiornate, dalla finestra Comandi DB2 eseguire le seguenti istruzioni SELECT relative alle tabelle:

```
DB2 SELECT * FROM SALES_TAB

DB2 SELECT * FROM PART_SIDE_TAB

DB2 SELECT * FROM ORDER_SIDE_TAB

DB2 SELECT * FROM SHIP_SIDE_TAB
```

Ricerca del documento XML

E' possibile ricercare il documento XML con un'interrogazione diretta nelle tabelle laterali. In questo passo, si ricercheranno tutti gli ordini con un prezzo superiore a 2500.00.

Per interrogare le tabelle laterali:

Dalla finestra Comandi DB2, immettere la seguente istruzione SELECT:

```
DB2 "SELECT DISTINCT SALES_PERSON FROM SALES_TAB S, PART_SIDE_TAB P
   WHERE PRICE > 2500.00 AND
   S.INVOICE_NUM=P.INVOICE_NUM"
```

La serie dei risultati è rappresentata dai nomi dei venditori degli articoli con prezzo superiore a 2500.00.

In alternativa, per ricercare il documento è possibile eseguire il seguente file dei comandi:

```
getstart_queryCol.cmd
```

Il supporto didattico di introduzione per la memorizzazione dei documenti XML nelle tabelle DB2 è stato completato. Una buona parte degli esempi di questa pubblicazione si basa sulle seguenti lezioni.

Lezione: Composizione di un documento XML

Scenario del supporto didattico

L'utente ha il compito di raccogliere informazioni nel database degli ordini di acquisto esistente, SALES_DB, e di estrarre le informazioni chiave da memorizzare nei documenti XML. Il settore assistenza clienti utilizzerà questi documenti XML per rispondere alle richieste e ai commenti dei clienti. Il settore assistenza clienti richiede dati specifici da includere e fornisce una struttura standard per i documenti XML.

Utilizzando i dati esistenti, si comporrà un documento XML, `getstart.xml`, dai dati compresi in queste tabelle.

Inoltre si pianifica e si crea un file DAD che associa le colonne delle tabelle correlate a una struttura di documento XML che fornisce un record degli ordini di acquisto. Poiché questo documento si compone di più tabelle, si creerà una raccolta XML, associando le tabelle a una struttura XML e a una DTD. Utilizzare questa DTD per definire la struttura del documento XML. E' inoltre possibile utilizzarla per convalidare il documento XML composto nelle applicazioni.

I dati del database esistente relativi al documento XML è descritto nelle seguenti tabelle. I nomi di colonna specificati in *corsivo* sono le colonne richieste dal settore assistenza clienti nella struttura del documento XML.

ORDER_TAB

Nome colonna	Tipo di dati
ORDER_KEY	INTEGER
CUSTOMER	VARCHAR(16)
CUSTOMER_NAME	VARCHAR(16)
CUSTOMER_EMAIL	VARCHAR(16)

PART_TAB

Nome colonna	Tipo di dati
PART_KEY	INTEGER
COLOR	CHAR(6)
QUANTITY	INTEGER
PRICE	DECIMAL(10,2)
TAX	REAL
ORDER_KEY	INTEGER

SHIP_TAB

Nome colonna	Tipo di dati
DATE	DATE
MODE	CHAR(6)
COMMENT	VARCHAR(128)
PART_KEY	INTEGER

Pianificazione

Prima di utilizzare XML Extender per la composizione dei documenti, è necessario stabilire la struttura del documento XML e la relativa corrispondenza alla struttura dei dati del database. Questa sezione fornisce una panoramica della struttura del documento XML richiesta dal settore assistenza clienti, della DTD da utilizzare per definire la struttura del documento XML e delle associazioni di questo documento con le colonne contenenti i dati utilizzati per compilare i documenti.

Determinazione della struttura del documento

La struttura del documento XML raccoglie le informazioni per un ordine specifico da più tabelle e crea un documento XML a esso relativo. Ciascuna di queste tabelle contiene informazioni relative all'ordine è possono essere unite nelle relative colonne chiave. Il settore assistenza clienti desidera che il documento sia strutturato con il numero ordine come primo livello, il cliente, la parte e le informazioni sulla spedizione come livelli successivi. Inoltre desidera che la struttura sia flessibile, con gli elementi che siano una descrizione dei dati anziché della struttura del documento. Ad esempio, il nome del cliente deve essere riportato nell'elemento denominato "cliente (customer)" e non in un paragrafo. A seconda della richiesta effettuata, la struttura gerarchica della DTD e del documento XML deve essere simile a quella descritta in Figura 5 a pagina 24.

Una volta stabilita la struttura del documento, sarà necessario creare una DTD per descrivere la struttura del documento XML. Questo supporto didattico fornisce un documento XML e una DTD. E' possibile consultare il file DTD nell'"Appendice B. Esempi" a pagina 223. Tale file corrisponde alla struttura riportata nella Figura 5 a pagina 24.

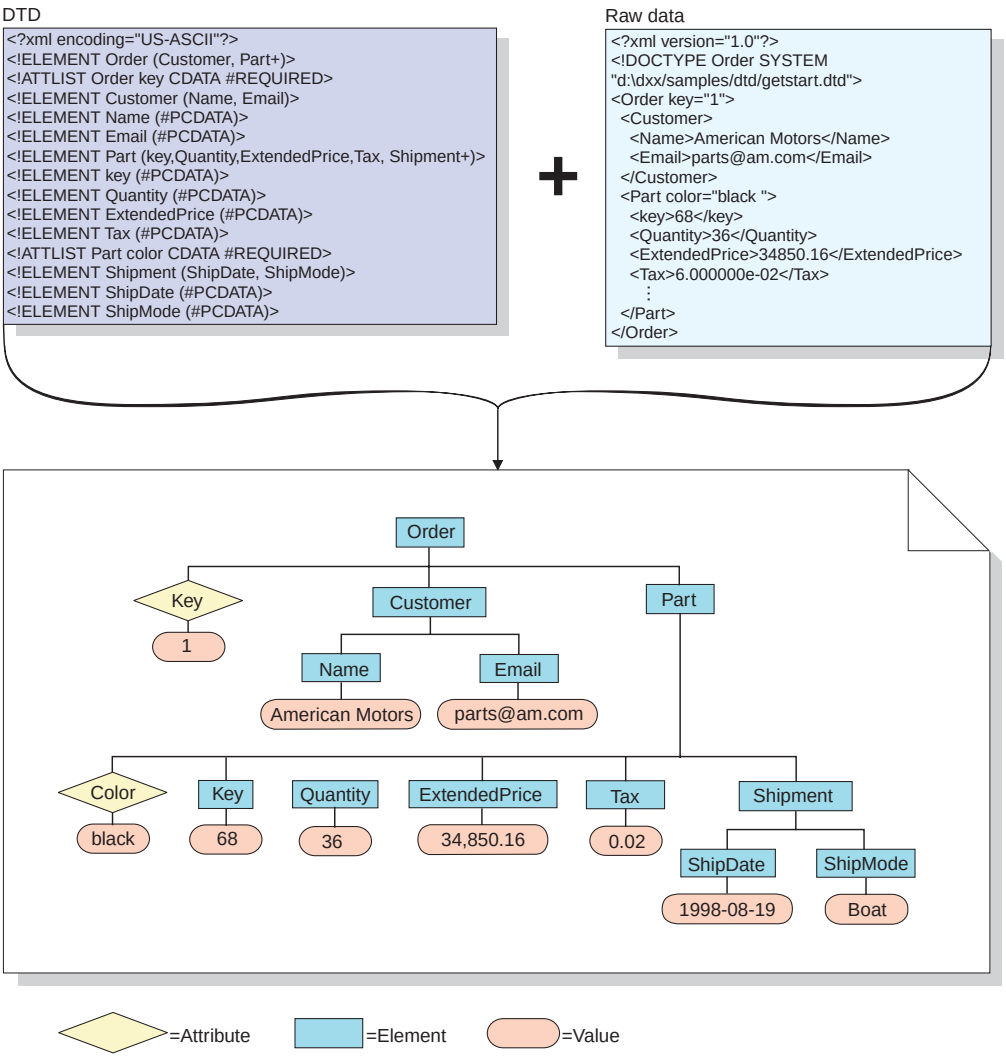


Figura 5. La struttura gerarchica della DTD e del documento XML

Associazione del documento XML alla relazione database

Una volta stabilita la struttura e creata la DTD, è necessario conoscere le relazioni tra la struttura del documento e le tabelle DB2 che verranno utilizzate per compilare gli elementi e gli attributi. E' possibile associare la struttura gerarchica alle colonne specifiche delle tabelle relazionali, come riportato nella Figura 6 a pagina 25.

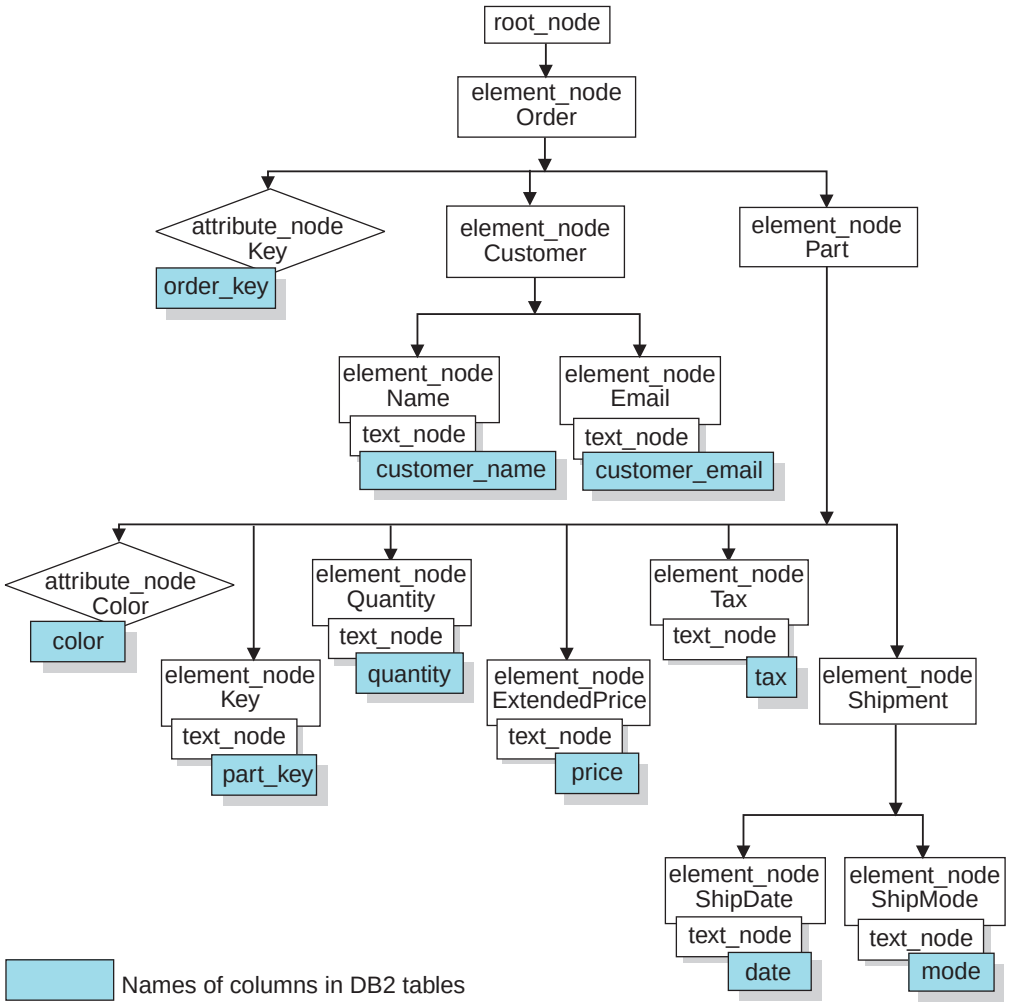


Figura 6. il documento XML associato alle colonne della tabella relazionale

Utilizzare questa descrizione per creare file DAD che definiscono le relazioni tra i dati relazionali e la struttura del documento XML.

Per creare il file DAD della raccolta XML, è necessario comprendere le modalità di corrispondenza del documento XML alla struttura del database, come descritto nella Figura 6, in modo che sia possibile risalire alle tabelle e alle colonne di origine da cui derivano i dati della struttura del documento XML relativi agli elementi e agli attributi. Queste informazioni verranno utilizzate per creare il file DAD della raccolta XML.

In questo supporto didattico, viene fornita una serie di script per consentire l'impostazione dell'ambiente. Questi script si trovano nella directory *DXX_INSTALL*\samples\cmd (dove *DXX_INSTALL* è l'unità e la directory di installazione di XML Extender, ad esempio c:\dxx\samples\cmd) ed essi sono i seguenti:

getstart_db.cmd
Crea il database e compila quattro tabelle.

getstart_prep.cmd
Esegue il bind del database con le procedure memorizzate XML Extender e le CLI DB2.

getstart_stp.cmd
Esegue la procedura memorizzata per creare la raccolta XML.

getstart_exportXML.cmd
Esporta il documento XML dal database per l'utilizzo in un'applicazione.

getstart_clean.cmd
Ripulisce l'ambiente del supporto didattico.

Impostazione

Creazione del database

In questa sezione si utilizza un comando per l'impostazione del database. Questo comando crea un database di esempio, ne stabilisce il collegamento, crea le tabelle in cui memorizzare i dati e quindi inserisce i dati.

Importante: se la lezione sulla colonna XML è stata completata e non è stata eseguita la cancellazione dei dati dell'ambiente in uso, ignorare questo passo. Accertarsi di disporre di un database SALES_DB.

Per creare il database:

1. Entrare nella directory *DXX_INSTALL*\samples\cmd, dove *DXX_INSTALL* è l'unità e la directory di installazione di XML Extender. In queste lezioni si utilizza la directory c:\dxx. Modificare tali valori se l'unità e la directory sono diverse.
2. Aprire la finestra Comandi DB2 dal menu Avvio di Windows NT oppure immettere il seguente comando dal prompt dei comandi di Windows NT:
   ```
   DB2CMD
   ```
3. Dalla finestra Comandi DB2, immettere il seguente comando:
   ```
   getstart_db.cmd
   ```

Abilitazione del database

Per memorizzare le informazioni XML nel database, è necessario abilitarlo per XML Extender. Quando si abilita un database per XML, XML Extender effettua le seguenti operazioni:

- Crea tutti i tipi definiti dall'utente (UDT) e le funzioni definite dall'utente (UDF).
- Crea e compila le tabelle di controllo utilizzando i metadati necessari richiesti da XML Extender.
- Crea lo schema db2xml e assegna i privilegi richiesti.

Importante: se la lezione sulla colonna XML è stata completata e non è stata eseguita la cancellazione dei dati dell'ambiente in uso, ignorare questo passo.

Per abilitare il database per XML:

Dalla finestra Comandi DB2, eseguire il seguente script per abilitare il database SALES_DB:

```
getstart_prep.cmd
```

Questo script esegue il bind del database con le procedure memorizzate XML Extender e le CLI DB2. Inoltre esegue l'opzione del comando **dxxadm** per abilitare il database:

```
dxxadm enable_db SALES_DB
```

Creazione della raccolta XML: preparazione del file DAD

Poiché i dati sono già presenti in più tabelle, si creerà una raccolta XML che associa le tabelle al documento XML. Per creare una raccolta XML, definirla preparando un file DAD.

Nella sezione "Pianificazione" a pagina 23 sono state stabilite le colonne del database relazionale in cui sono presenti i dati e come i dati delle tabelle verranno strutturati in un documento XML. In questa sezione, si creerà lo schema di associazione nel file DAD che specifica la relazione tra le tabelle e la struttura del documento XML.

Nei seguenti passi, si fa riferimento agli elementi del file DAD come *tag* e agli elementi della struttura del documento XML come *elementi*. Un esempio di un file DAD simile a quello creato è nel percorso `c:\dxx\samples\dad\getstart_xcollection.dad` il quale presenta alcune differenze dal file generato nei seguenti passi. Se si utilizza tale file per la lezione, è possibile che i percorsi file differiscano dall'ambiente utilizzato.

Per creare il file DAD per la composizione di un documento XML:

1. Dalla directory `c:\dxx\samples\cmd`, aprire un editor di testo e creare un file con nome `getstart_xcollection.dad`.
2. Creare l'intestazione DAD, utilizzando il seguente testo:

   ```
   <?xml version="1.0"?>
   <!DOCTYPE DAD SYSTEM "c:\dxx\dtd\dad.dtd">
   ```

   XML Extender presume che il prodotto sia stato installato in `c:\dxx`. In caso contrario, modificare, nel seguente passo e in quelli successivi, questo valore con quello relativo all'unità e alla directory specificata durante l'installazione del prodotto.
3. Inserire le tag <DAD></DAD>. Tutte le altre tag sono situate all'interno di queste tag.
4. Specificare le tag <validation> </validation> per indicare se XML Extender esegue la convalida della struttura del documento XML utilizzando la DTD inserita nella tabella del magazzino DTD.

   ```
   <validation>NO</validation>
   ```
5. Utilizzare le tag <Xcollection></Xcollection> per definire il metodo di memorizzazione e di accesso come raccolta XML. I metodi di memorizzazione e accesso consentono di definire se i dati XML sono memorizzati in una raccolta delle tabelle DB2.

   ```
   <Xcollection>
   </Xcollection>
   ```
6. Specificare un'istruzione SQL per specificare le tabelle e le colonne utilizzate per la raccolta XML. Questo metodo è denominato Associazione SQL ed è uno dei due metodi di associazione dei dati relazionali alla struttura del documento XML. Per ulteriori informazioni sugli schemi di associazione, consultare la sezione "Tipi di schema di associazione" a pagina 50. Immettere la seguente istruzione:

   ```
   <SQL_stmt>
   SELECT o.order_key, customer_name, customer_email, p.part_key, color, quantity,
   price, tax, ship_id, date, mode from order_tab o, part_tab p,
   table (select substr(char(timestamp(generate_unique())),16)
     as ship_id, date, mode, part_key from ship_tab) s
         WHERE o.order_key = 1 and
                   p.price > 20000 and
   ```

```
            p.order_key = o.order_key and
            s.part_key = p.part_key
      ORDER BY order_key, part_key, ship_id
 </SQL_stmt>
```

Questa istruzione SQL utilizza le seguenti indicazioni quando utilizza l'associazione SQL. Per la struttura del documento fare riferimento alla Figura 6 a pagina 25.

- Le colonne vengono specificate dall'alto verso il basso, in base alla gerarchia della struttura dei documenti XML. Ad esempio, come prima colonna quella relativa agli elementi ordine e cliente, come seconda colonna l'elemento parte e terza colonna quella relativa alla spedizione.
- Le colonne relative a un'entità vengono raggruppate e ciascun gruppo presenta una colonna ID oggetto: ORDER_KEY, PART_KEY e SHIP_ID.
- La colonna ID oggetto è la prima di ciascun gruppo. Ad esempio, O.ORDER_KEY precede le colonne relative all'attributo chiave e p.PART_KEY precede le colonne relative all'elemento Parte.
- La tabella SHIP_TAB non presenta un'unica colonna condizionale chiave, quindi viene utilizzata la funzione incorporata DB2 generate_unique per creare la colonna SHIP_ID.
- Le colonne ID oggetto vengono, quindi, elencate in ordine dall'alto verso il basso in un'istruzione ORDER BY. Le colonne in ORDER BY non devono essere qualificate da alcun nome tabella o schema e devono corrispondere ai nomi colonna presenti nella clausola SELECT.

Per i requisiti relativi alla scrittura di un'istruzione SQL consultare la sezione "Requisiti dello schema di associazione" a pagina 52.

7. Aggiungere il seguente prologo da utilizzare nel documento XML composto.

   ```
   <prolog>?xml version="1.0"?</prolog>
   ```

   E' richiesto il testo esatto per tutti i file DAD.

8. Aggiungere le tag <doctype></doctype> da utilizzare nel documento XML che si compone. La tag <doctype> contiene il percorso della DTD memorizzata nel client.

   ```
   <doctype>! DOCTYPE Order SYSTEM "c:\dxx\samples\dtd\getstart.dtd"</doctype>
   ```

9. Definire l'elemento root del documento XML utilizzando le tag <root_node></root_node>. Nella tag root_node, specificare gli elementi e gli attributi che costituiscono il documento XML.
10. Associare la struttura del documento XML alla struttura della tabella relazionale utilizzando i seguenti tre tipi di nodi:

    element_node
    Specifica l'elemento nel documento XML. Element_nodes può presentare element_nodes secondari.

    attribute_node
    Specifica l'attributo di un elemento del documento XML.

    text_node
    Specifica il contenuto del testo di un elemento e i dati della colonna in una tabella relazionale per gli element_nodes di livello inferiore.

    Per ulteriori informazioni relative a questi nodi consultare la sezione "File DAD" a pagina 47. La Figura 6 a pagina 25 illustra la struttura gerarchica del documento XML e le colonne della tabella DB2 e indica i tipi di nodi

utilizzati. Le caselle a sfondo grigio indicano i nomi delle colonne delle tabelle DB2 da cui i dati verranno estratti per comporre il documento XML.

Eseguire i seguenti passi per aggiungere i vari tipi di nodi, uno per volta.

a. Definire una tag <element_node> per ciascun elemento del documento XML.

```
    <root_node>
<element_node name="Order">
          <element_node name="Customer">
    <element_node name="Name">
     </element_node>
    <element_node name="Email">
     </element_node>
</element_node>
          <element_node name="Part">
    <element_node name="key">
     </element_node>
    <element_node name="Quantity">
</element_node>
          <element_node name="ExtendedPrice">
</element_node>
<element_node name="Tax">
</element_node>
     <element_node name="Shipment" multi_occurrence="YES">
               <element_node name="ShipDate">
</element_node>
               <element_node name="ShipMode">
</element_node>
    </element_node> <!-- end Shipment -->
  </element_node> <!-- end Part -->
</element_node> <!-- end Order -->
    </root_node>
```

Tenere presente che l'elemento secondario <Shipment> presenta un attributo multi_occurrence="YES". Questo attributo è utilizzato per gli elementi privi di attributo che si ripetono nel documento. L'elemento <Part> non utilizza l'attributo multi-occurrence in quanto dispone di un attributo color che lo rende unico.

b. Definire una tag <attribute_node> per ciascun attributo del documento XML. Questi attributi sono nidificati nel relativo element_node. Gli ulteriori attribute_nodes sono evidenziati in grassetto:

```
    <root_node>
<element_node name="Order">
  <attribute_node name="key">
    </attribute_node>
          <element_node name="Customer">
    <element_node name="Name">
     </element_node>
     <element_node names"Email">
     </element_node>
</element_node>
          <element_node name="Part">
    <attribute_node name="color">
    </attribute_node>
    <element_node name="key">
    </element_node>
<element_node name="Quantity">
</element_node>

...
```

```
  </element_node> <!-- end Part -->
</element_node> <!-- end Order -->
    </root_node>
```

c. Per ciascun element_node di livello inferiore, definire le tag <text_node> che indicano che l'elemento XML contiene dati carattere da estrarre da DB2 durante la composizione del documento.

```
    <root_node>
<element_node name="Order">
<attribute_node name="key">
  </attribute_node>
          <element_node name="Customer">
    <element_node name="Name">
   <text_node>
        </text_node>
    </element_node>
    <element_node name="Email">
   <text_node>
        </text_node>
    </element_node>
</element_node>
          <element_node name="Part">
    <attribute_node name="color">
</attribute_node>
    <element_node name="key">
   <text_node>
        </text_node>
    </element_node>
<element_node name="Quantity">
   <text_node>
        </text_node>
</element_node>
          <element_node name="ExtendedPrice">
   <text_node>
        </text_node>
</element_node>
<element_node name="Tax">
   <text_node>
        </text_node>
</element_node>
    <element_node name="Shipment" multi-occurrence="YES">
             <element_node name="ShipDate">
   <text_node>
        </text_node>
</element_node>
             <element_node name="ShipMode">
   <text_node>
        </text_node>
</element_node>
    </element_node> <!-- end Shipment -->
  </element_node> <!-- end Part -->
</element_node> <!-- end Order -->
    </root_node>
```

d. Per ciascun element_node di livello inferiore, definire una tag <column>. Queste tag specificano le colonne da cui estrarre i dati durante la composizione del documento XML e generalmente sono interne alle tag <attribute_node> o <text_node>. Tenere presenta che le colonne definite devono essere comprese nella clausola <SQL_stmt> SELECT.

```
    <root_node>
<element_node name="Order">
<attribute_node name="key">
    <column name="order_key"/>
  </attribute_node>
          <element_node name="Customer">
```

```
    <element_node name="Name">
   <text_node>
        <column name="customer_name"/>
   </text_node>
</element_node>
    <element_node name="Email">
   <text_node>
        <column name="customer_email"/>
   </text_node>
</element_node>
</element_node>
          <element_node name="Part">
    <attribute_node name="color">
       <column name="color"/>
</attribute_node>
    <element_node name="key">
   <text_node>
        <column name="part_key"/>
   </text_node>
<element_node name="Quantity">
   <text_node>
        <column name="quantity"/>
   </text_node>
</element_node>
          <element_node name="ExtendedPrice">
   <text_node>
        <column name="price"/>
   </text_node>
</element_node>
<element_node name="Tax">
   <text_node>
        <column name="tax"/>
   </text_node>
</element_node>
    <element_node name="Shipment" multi-occurrence="YES">
             <element_node name="ShipDate">
   <text_node>
          <column name="date"/>
   </text_node>
</element_node>
             <element_node name="ShipMode">
   <text_node>
          <column name="mode"/>
   </text_node>
</element_node>
    </element_node> <!-- end Shipment -->
  </element_node> <!-- end Part -->
</element_node> <!-- end Order -->
    </root_node>
```

11. Verificare se viene posizionata una tag </root_node> dopo l'ultima tag </element_node>.
12. Verificare se viene posizionata una tag </Xcollection> dopo la tag </root_node>.
13. Verificare se viene posizionata una tag </DAD> dopo la tag </Xcollection>.
14. Salvare il file come `getstart_xcollection.dad`

E' possibile confrontare il file appena creato con il file di esempio `c:\dxx\samples\dad\getstart_xcollection.dad`. Questo file è una copia di lavoro del file DAD richiesto per la composizione del documento XML. Il file di esempio contiene istruzioni di percorso che è possibile modificare in modo da soddisfare i requisiti dell'ambiente utilizzato.

Nell'applicazione, l'utente utilizzerà frequentemente una raccolta XML per comporre documenti, è possibile definire un nome di raccolta abilitando la raccolta. L'abilitazione della raccolta esegue la relativa registrazione nella tabella XML_USAGE e ne migliora le prestazione quando si specifica il nome della raccolta (e non il nome del file DAD) durante l'esecuzione delle procedure memorizzate. Nelle seguenti lezioni, non si eseguirà l'abilitazione della raccolta. Per ulteriori informazioni su tale operazione consultare la sezione "Abilitazione delle raccolte XML" a pagina 91.

Composizione del documento XML

In questo passo si utilizza la procedura memorizzata dxxGenXML() per comporre il documento XML specificato dal file DAD. Questa procedura restituisce il documento come XMLVARCHAR UDT.

Per comporre il documento XML:

1. Dalla finestra Comandi DB2, immettere il seguente comando per eseguire la procedura memorizzata:

   ```
   getstart_stp.cmd
   ```

   Il documento XML viene composto e memorizzato nella tabella RESULT_TAB.

   Gli esempi delle procedure memorizzate che è possibile utilizzare in questo passo sono riportati nei seguenti file:

   - `c:\dxx\samples\c\tests2x.sqc` illustra come richiamare la procedura memorizzata utilizzando SQL integrata e genera il file eseguibile `texts2x` utilizzato dal comando `getstart_stp.cmd`.
   - `c:\dxx\samples\cli\sql2xml.c` illustra come richiamare la procedura memorizzata utilizzando CLI.
2. Esportare il documento XML dalla tabella in un file che utilizza la funzione di richiamo di XML Extender, Content():

   ```
   DB2 CONNECT TO SALES_DB

   DB2 SELECT db2xml.Content(db2xml.xmlvarchar(doc),
      'c:\dxx\samples\cmd\getstart.xml') FROM RESULT_TAB
   ```

   In alternativa, per esportare il file è possibile eseguire il seguente comando:

   ```
   getstart_exportXML.cmd
   ```

   Questa lezione illustra come utilizzare la funzione della serie dei risultati delle procedure memorizzate DB2 che consente la lettura sequenziale delle righe per richiamare il documento XML composto. E' possibile esportare in un file le righe richiamate dal documento, questo rappresenta il metodo più semplice per dimostrare tale funzione. Per dimostrazioni più efficienti della lettura sequenziale dei dati consultare gli esempi CLI riportati in `c:\dxx\samples\cli`.

Cancellazione dati dall'ambiente del supporto didattico

Se si desidera cancellare i dati dall'ambiente del supporto didattico, è possibile eseguire il file `getstart_clean.cmd`. Questo file:

- Disabilita la colonna XML, ORDER
- Cancella le tabelle create nel supporto didattico
- Cancella la DTD dalla tabella di riferimento DTD

Questo file dei comandi non disabilita o cancella il database SALES_DB; il database è ancora disponibile per l'utilizzo di XML Extender. Se non si completano entrambe le lezioni di questo capitolo, è possibile che vengano visualizzati alcuni messaggi di errore. E' possibile ignorarli.

Per cancellare i dati dall'ambiente del supporto didattico.

1. Dalla finestra Comandi DB2, immettere il seguente comando:

   ```
   getstart_clean.cmd
   ```

2. Se si desidera disabilitare il database, dalla finestra Comandi DB2 è possibile eseguire il seguente comando di XML Extender:

   ```
   dxxadm disable_db SALES_DB
   ```

   Questo comando cancella le tabelle di controllo gestione DTD_REF e XML_USAGE e elimina i tipi definiti dall'utente e le funzioni fornite da XML Extender.

3. Se si desidera cancellare il database, dalla finestra Comandi DB2 è possibile eseguire il seguente comando:

   ```
   db2 drop database SALES_DB
   ```

   Questo comando cancella SALES_DB.

Parte 2. Gestione

Questa sezione descrive le modalità di gestione delle attività relative a XML Extender.

Capitolo 3. Informazioni preliminari sulla gestione di XML Extender:

Questo capitolo descrive i requisiti per l'installazione e la pianificazione di XML Extender.

Requisiti per l'installazione

Le seguenti sezioni descrivono i requisiti per l'installazione di XML Extender.

Requisiti software

XML Extender è disponibile su AIX, Windows NT e Sun Solaris.

Software richiesto: XML Extender richiede DB2 Universal Database Versione 7.1 o successive.

Software opzionale:

- Per la ricerca di testo strutturale, DB2 Universal Database Text Extender Versione 7.1 o successive
- Per lo strumento di gestione XML Extender:
  - JDBC DB2 UDB (disponibile con DB2 UDB Versione 7.1 o successive)
  - JDK 1.1.7 o JRE 1.1.7 (disponibile con il Centro di controllo DB2 UDB)
  - JFC 1.1 con Swing 1.1 (disponibile con il Centro di controllo DB2 UDB)

Requisiti per l'installazione

Consultare il file README relativo al proprio sistema operativo per le seguenti attività:

- Esecuzione del bind di XML Extender al database DB2 UDB.

  Per motivi di sicurezza, è necessario eseguire il bind di XML Extender ad ogni database. Per ulteriori dettagli sull'esecuzione del bind, consultare la sezione "Informazioni preliminari" a pagina 174 o

  DXX_INSTALL\samples\cmd\getstart_prep.cmd
- Visualizzazione delle istruzioni di installazione su UNIX.
- Creazione di un database per l'accesso XML.

Requisiti per l'autorizzazione

Per eseguire le attività di gestione occorre l'autorizzazione DB2ADM.

Strumenti di gestione

XML Extender fornisce tre metodi di gestione: il wizard di gestione XML Extender, il comando di gestione XML Extender e le *procedure memorizzate* XML Extender.

- Il wizard di gestione assiste l'utente nelle varie attività di gestione e rappresenta il metodo di gestione consigliato. L'utilizzo di questo strumento è descritto nelle attività di gestione della sezione "Capitolo 4. Gestione dei dati XML" a pagina 57.
- Il comando di gestione, **dxxadm**, fornice opzioni per le varie attività di gestione. Le informazioni sull'utilizzo di questo comando sono descritte nelle sezioni

"Capitolo 4. Gestione dei dati XML" a pagina 57 e "Capitolo 7. Comando di gestione XML Extender: **dxxadm**" a pagina 131.

- Le procedure memorizzate di gestione forniscono anche opzioni per le varie attività di gestione. Queste procedure memorizzate vengono descritte nella sezione "Procedure memorizzate di gestione" a pagina 175.

Pianificazione della gestione

Quando si pianifica un'applicazione che utilizza i documenti XML, occorre innanzitutto stabilire:

- Se si desidera eseguire la composizione dei documenti XML dai dati del database
- Se si desidera memorizzare i documenti XML preesistenti come documenti intatti o scomposti in dati DB2

Successivamente sarà possibile pianificare le altre attività di gestione:

- Convalida dei documenti XML
- Indicizzazione dei dati delle colonne XML per ricerche veloci
- Associazione della struttura del documento XML alle tabelle relazionali DB2

Le modalità di utilizzo di XML Extender variano a seconda dell'applicazione richiesta. Come indicato nella sezione "Capitolo 1. Introduzione a XML Extender" a pagina 3, è possibile eseguire la composizione dei documenti XML dai dati DB2 esistenti e memorizzare questi documenti nel DB2 come documenti intatti o dati DB2. Ciascuno di questi metodi di accesso e memorizzazione presenta requisiti di pianificazione differenti. Le seguenti sezioni descrivono le considerazioni sulla pianificazione.

Metodi di accesso e memorizzazione

XML Extender fornisce due metodi di accesso e pianificazione per utilizzare DB2 come un magazzino XML: Colonna XML e Raccolta XML. Selezionare il metodo più adatto alle proprie esigenze.

Colonna XML
: Memorizza e richiama i documenti XML come dati della colonna DB2. I dati XML vengono rappresentati da una colonna XML.

Raccolta XML
: Esegue la composizione o la scomposizione dei documenti XML in una raccolta delle tabelle referenziali.

Il tipo di applicazione determina il metodo di accesso e memorizzazione da utilizzare e come strutturare i dati XML. I seguenti scenari descrivono le situazioni in cui viene utilizzato il metodo di accesso e memorizzazione appropriato.

Quando utilizzare le colonne XML

Utilizzare le colonne XML nei casi in cui:

- I documenti XML esistono già o sono stati richiamati da un'origine esterna e si desidera memorizzarli nel formato XML nativo. I documenti verranno memorizzati nel DB2 per l'integrità a scopo di controllo e archiviazione.
- I documenti XML vengono generalmente letti ma non aggiornati.
- Si desidera utilizzare i tipi di dati del nome file per memorizzare i documenti XML nel file system locale o remoto e utilizzare DB2 per le operazioni di gestione e ricerca.

- Si desidera instradare la ricerca in base ai valori degli elementi o degli attributi XML e si conoscono quelli ricercati di frequente.
- I documenti sono costituiti da elementi con blocchi di testo di grandi dimensioni e si desidera utilizzare DB2 Text Extender per la ricerca di testo strutturale lasciando i documenti intatti.

Quando utilizzare le raccolte XML

Utilizzare le raccolte XML nei casi in cui:

- Le tabelle relazionali esistenti contengono dati e si desidera eseguire la composizione dei documenti XML in base a una determinata DTD.
- Si desidera memorizzare i propri documenti XML con le raccolte di dati che si associano perfettamente alle tabelle relazionali.
- Si desidera creare delle nuove viste di dati relazionali utilizzando diversi schemi di associazione.
- I documenti XML sono stati richiamati da altre origini di dati. Si desidera memorizzare nel database solo i dati senza le tag. Si desidera disporre dell'alternativa di memorizzare i dati nelle tabelle esistenti o in quelle nuove.
- Una serie di documenti XML di dimensioni ridotte richiede l'aggiornamento frequente e le prestazioni di aggiornamento risultano critiche.
- Occorre memorizzare i dati dei documenti XML in entrata ma spesso si desidera richiamare soltanto la relativa serie secondaria.
- I documenti XML superano i 2 gigabyte ed occorre scomporli.

Viene utilizzato il file DAD (document access definition) per associare i dati XML alle tabelle DB2 tramite questi due metodi di accesso e memorizzazione.La Figura 7 a pagina 40 indica come DAD specifica i metodi di accesso e memorizzazione.

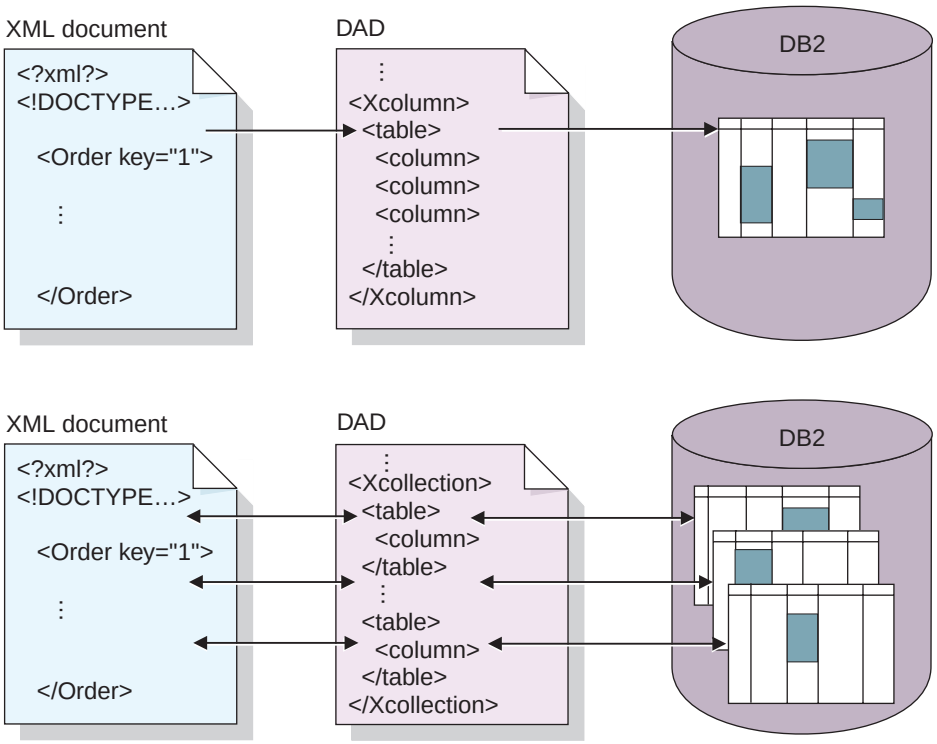


Figura 7. Il file DAD associa la struttura del documento XML al DB2 e specifica il metodo di accesso e memorizzazione.

Il file DAD svolge un ruolo importante nella gestione di XML Extender. Definisce l'ubicazione dei file chiave, come DTD, e specifica come la struttura dei documenti XML viene associata ai dati DB2. Inoltre, definisce i metodi di accesso e memorizzazione da utilizzare nelle applicazioni.

Pianificazione per le colonne XML

Le seguenti sezioni descrivono le attività di pianificazione per le colonne XML.

Convalida

Dopo aver scelto un metodo di accesso e memorizzazione, è possibile specificare se *convalidare* i propri dati. I dati XML vengono convalidati utilizzando una DTD per assicurarsi che il documento XML sia valido e per eseguire ricerche strutturate nei dati XML. La DTD è memorizzata nel magazzino DTD, oppure, può essere memorizzata nel file system a cui ha accesso il server DB2.

E' possibile convalidare i documenti nella stessa colonna XML utilizzando DTD differenti. In altri termini, è possibile avere dei documenti che presentano la stessa struttura, con elementi e attributi simili, che richiamano DTD differenti. Per fare riferimento a più DTD, utilizzare le seguenti indicazioni:

- L'ID di sistema del documento XML nella definizione DOCTYPE deve specificare il file DTD con un nome percorso completo.
- Specificare YES per la convalida nel file DAD.
- Almeno una delle DTD deve essere memorizzata nella tabella DTD_REF. Tutte le DTD possono essere memorizzate in questa tabella.
- Le DTD devono presentare la stessa struttura con elementi secondari differenti.
- Il file DAD deve specificare elementi e attributi comuni a tutte le DTD a cui fanno riferimento i documenti della colonna.

Importante: Specificare se si desidera eseguire la convalida prima di inserire i dati XML nel DB2. XML Extender non supporta la convalida dei dati già inseriti nel DB2.

Considerazioni:

- Si consiglia di convalidare i dati XML con una DTD, a meno che i documenti XML non vengano memorizzati a scopo di archiviazione. Per eseguire la convalida, è necessario che il magazzino XML Extender contenga una DTD. Per informazioni su come memorizzazione una DTD nel magazzino, consultare la sezione "Memorizzazione di DTD in un magazzino DTD" a pagina 61.
- Non occorre una DTD per memorizzare o archiviare i documenti XML.
- L'operazione di convalida dei dati XML può influire in maniera irrilevante sulle prestazioni.
- E' possibile utilizzare più DTD, ma solo gli elementi e gli attributi comuni possono essere indicizzati.
- Se non si sceglie di convalidare un documento, la DTD specificata dal documento XML non viene elaborata. E' importante che le DTD vengano elaborate per risolvere valori predefiniti di entità e attributi, anche quando si elaborano frammenti di documenti che non possono essere convalidati.

UDT XML

Memorizzare un documento XML in una colonna XML come DTD. Per informazioni sulle UDT disponibili, consultare la sezione Tabella 4.

Tabella 4. UDT XML Extender

Colonna UDT	Tipo di dati origine	Descrizione
XMLVARCHAR	VARCHAR(*varchar_len*)	Memorizza un intero documento XML come VARCHAR nel DB2.
XMLCLOB	CLOB(*clob_len*)	Memorizza un intero documento XML come *CLOB* nel DB2.
XMLFILE	VARCHAR(1024)	Memorizza il nome file di un documento XML nel DB2 e memorizza il documento XML in un file locale sul server DB2.

Tabelle laterali

Quando si esegue la pianificazione delle tabelle laterali, occorre stabilire le modalità in base alle quali si desidera organizzare le tabelle, il numero di tabelle da creare e se definire una vista predefinita per queste tabelle. Queste decisioni sono basate in parte su diverse condizioni: la ricorrenza di elementi e attributi e i requisiti relativi alle prestazioni delle interrogazioni.

Ricorrenze multiple: Quando per un documento ricorrono più percorsi di ubicazione, XML Extender aggiungerà una colonna DXX_SEQNO di tipo INTEGER in ogni tabella per memorizzare una traccia dell'ordine degli elementi che ricorrono più volte. Con DXX_SEQNO, è possibile richiamare un elenco di elementi che utilizza lo stesso ordine del documento XML originale specificando ORDER BY DXX_SEQNO in un'interrogazione SQL.

Viste predefinite e prestazioni di interrogazione: Quando si abilita una colonna XML, è possibile specificare un valore predefinito, una vista di sola lettura che

collega la tabella applicativa alle tabelle laterali utilizzando un ID univoco, denominato ROOT ID. Con la vista predefinita è possibile ricercare documenti XML interrogando le tabelle laterali. Ad esempio, se si specifica la tabella applicativa SALES_TAB e le tabelle laterali ORDER_TAB, PART_TAB e SHIP_TAB:

```
SELECT sales_person FROM sales_order_view
   WHERE price > 2500.00
```

L'istruzione SQL restituisce i nomi dei venditori in SALES_TAB i cui ordini sono memorizzati nella colonna ORDER con il valore PRICE superiore a 2500.00.

L'interrogazione della vista predefinita consente la visualizzazione di una singola vista virtuale della tabella applicativa e delle tabelle laterali. Tuttavia, tenere presente che la complessità di un'interrogazione è direttamente proporzionale al numero di tabelle laterali create. Quindi, la creazione della vista predefinita è consigliata solo se il numero totale di colonne delle tabelle laterali è ridotto. Le applicazioni possono creare delle proprie viste, unendo le principali colonne delle tabelle laterali.

Indici per i dati della colonna XML

Per la pianificazione è necessario stabilire se indicizzare i documenti delle colonne XML. Questa decisione deve essere valutata in base alla frequenza di accesso ai dati e allo stato delle prestazioni durante le ricerche strutturali.

Quando si utilizzano le colonne XML contenenti documenti XML interi, è possibile creare le tabelle laterali per includere le colonne dei valori degli attributi e degli elementi XML e gli indici in queste colonne. E' necessario specificare gli elementi e gli attributi per i quali si desidera creare l'indice.

L'indicizzazione delle colonne XML consente l'interrogazione frequente dei vari tipi di dati, come integer o decimal, e la creazione dell'indice utilizzando il supporto di indice DB2 nativo dall'engine del database. XML Extender richiama i valori degli elementi e degli attributi XML dai documenti XML e li memorizza nelle *tabelle laterali*, consentendo la creazione degli indici in queste tabelle.

E' possibile specificare ogni colonna di una tabella laterale con un percorso di ubicazione che identifica un elemento o attributo XML e un tipo di dati SQL. La Figura 8 a pagina 43 illustra una colonna XML con le tabelle laterali.

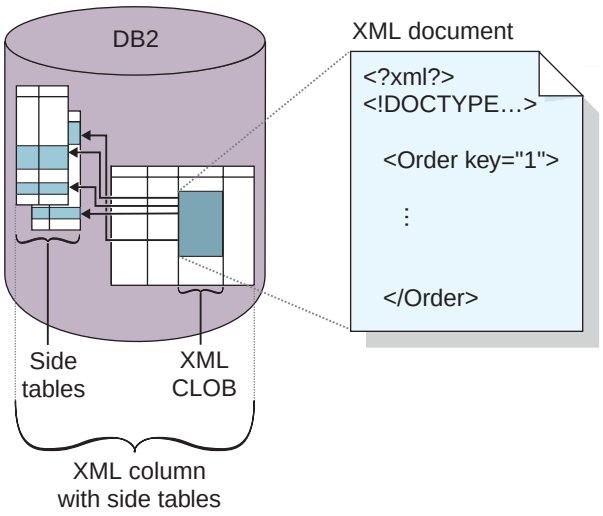


Figura 8. Una colonna XML con le tabelle laterali

XML Extender compila automaticamente la tabella laterale durante la memorizzazione dei documenti XML nella colonna XML.

Per una ricerca veloce, creare gli indici in queste colonne utilizzando la tecnologia DB2 *Indicizzazione strutturata B*. I metodi utilizzati per creare un indice variano a seconda del sistema operativo e XML Extender supporta questi metodi differenti.

Considerazioni:

- Per gli elementi o gli attributi a *ricorrenza multipla* contenuti in un documento XML, è necessario creare una tabella laterale separata per ogni elemento o attributo XML a ricorrenza multipla a causa della struttura complessa dei documenti XML.

  Ad esempio, è possibile creare un indice in `/Order/Part/ExtendedPrice` e specificare il tipo di dati REAL per `/Order/Part/ExtendedPrice`. In questo caso, XML Extender memorizza il valore di `/Order/Part/ExtendedPrice` della colonna PRICE in una tabella laterale.
- E' possibile creare più indici in una colonna XML. Utilizzando l'esempio precedente, è possibile creare due colonne in due tabelle laterali, una per `ExtendedPrice` e l'altra per `ShipDate`.
- E' possibile associare le tabelle laterali alla tabella applicativa utilizzando ROOT ID, il nome della colonna della chiave primaria nella tabella applicativa e un identificativo univoco che associa tutte le tabelle laterali alla tabella applicativa. E' possibile specificare la chiave primaria della tabella applicativa come ROOT ID, anche se non potrà essere definita come chiave composta. Si consiglia di utilizzare questo metodo.

  Se la chiave primaria singola non esiste nella tabella applicativa oppure non viene utilizzata, XML Extender modifica la tabella applicativa aggiungendovi una colonna DXXROOT_ID, che memorizza un ID univoco creato durante l'inserimento. Tutte le tabelle laterali avranno un colonna DXXROOT_ID con un ID univoco. Se la chiave primaria viene utilizzata come ROOT ID, tutte le tabelle laterali avranno una colonna con lo stesso nome della colonna della chiave primaria nella tabella applicativa e i dati delle chiavi primarie verranno memorizzati.
- Se si abilita una colonna XML per DB2 Text Extender, è anche possibile utilizzare la funzione di testo strutturale Text Extender. Text Extender fornisce il supporto

di *"ricerca sezione"*, che estende le capacità di ricerca testo convenzionali consentendo la ricerca delle parole corrispondenti all'interno di un determinato contesto del documento specificato nei percorsi di ubicazione. L'*indice di testo strutturale* può essere utilizzato con la funzione di indicizzazione XML Extender per tutti i tipi di dati SQL.

Percorso di ubicazione

Un *percorso di ubicazione* è una sequenza di *tag XML* che identifica un elemento o attributo XML. XML Extender utilizza il percorso di ubicazione nelle seguenti situazioni:

- Durante l'estrazione delle UDF per identificare gli elementi e gli attributi da estrarre.
- Per specificare il file di associazione tra un elemento o attributo XML e una colonna DB2 durante la definizione dello schema di indicizzazione nelle colonne DAD per XML
- Da Text Extender per la ricerca di testo strutturale

La Figura 9 mostra un esempio di percorso di ubicazione e delle relative relazioni alla struttura del documento XML.

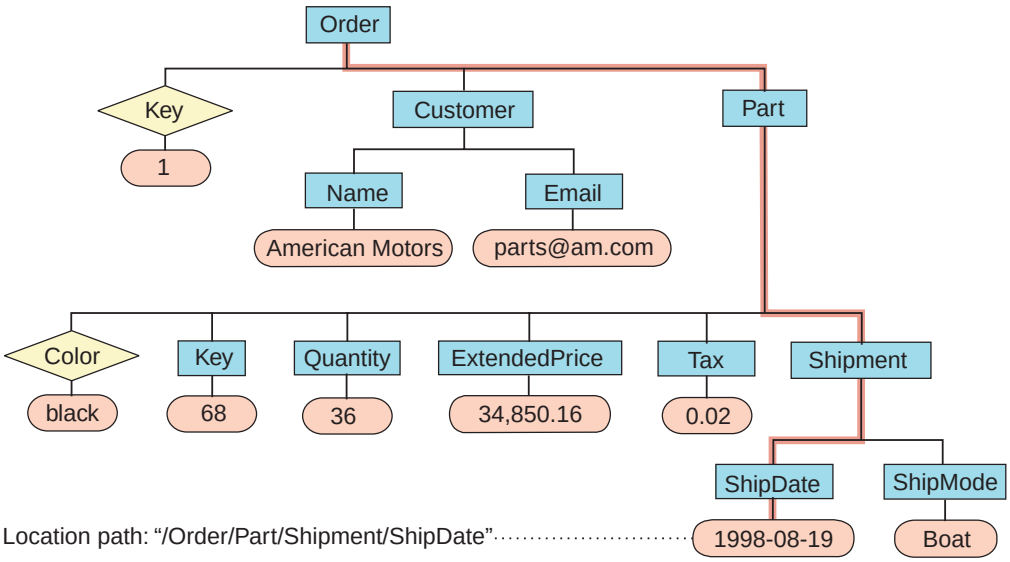


Figura 9. Memorizzazione dei documenti come documenti XML strutturati in una colonna di tabella DB2

Sintassi del percorso di ubicazione: Il seguente elenco descrive la sintassi del percorso di ubicazione supportata da XML Extender. Il percorso con singola barra (/) indica che il contesto è un documento intero.

1. / Rappresenta l'*elemento root* XML.

2. */tag1*
Rappresenta l'elemento *tag1* nella root.

3. */tag1/tag2/..../tagn*
Rappresenta un elemento denominato *tagn* come elemento secondario di una catena discendente da root, *tag1*, *tag2* fino a *tagn*-1.

4. *//tagn*
Rappresenta un qualsiasi elemento denominato *tagn* e le doppie barre (//) indicano zero o più tag arbitrarie.

5. */tag1//tagn*

Rappresenta un qualsiasi elemento denominato *tagn*, un elemento secondario denominato *tag1* nella root e le doppie barre (//) indicano zero o più tag arbitrarie.

6. */tag1/tag2/@attr1*

Rappresenta l'attributo *attr1* di un elemento denominato *tag2*, che è l'elemento secondario di *tag1* nella root.

7. */tag1/tag2[@attr1="5"]*

Rappresenta un elemento denominato *tag2* il cui attributo *attr1* ha come valore 5. *tag2* è l'elemento secondario di *tag1* nella root.

8. */tag1/tag2[@attr1="5"]/.../tagn*

Rappresenta un elemento denominato *tagn*, che è un elemento secondario della catena decrescente, *tag1*, *tag2*, fino a *tagn*-1, dove il valore dell'attributo *attr1* di *tag2* è 5.

Caratteri jolly: E' possibile sostituire un asterisco per un elemento in un percorso di ubicazione per la corrispondenza con altre stringhe.

Percorso di ubicazione semplice: *Percorso di ubicazione semplice* rappresenta la sintassi del percorso di ubicazione utilizzata per specificare elementi e attributi per tabelle laterali, definite nel file DAD della colonna XML. Il percorso di ubicazione semplice viene rappresentato come una sequenza di nomi di tipo elemento collegati mediante una singola barra (/). I valori degli attributi sono racchiusi tra parentesi quadre seguiti dal tipo di elemento. In Tabella 5 viene descritta la sintassi per il percorso di ubicazione semplice.

Tabella 5. Sintassi del percorso di ubicazione semplice

Argomento	Percorso di ubicazione	Descrizione
elemento XML	*/tag1/tag2/..../tagn-1/tagn*	Il contenuto di un elemento identificato dall'elemento denominato *tagn* e dai relativi elementi secondari
attributo XML	*/tag_1/tag_2/..../tag_n-1/tag_n/@attr1*	Un attributo denominato *attr1* dell'elemento identificato da *tagn* e dai relativi elementi secondari

XML Extender - Limitazioni: XML Extender presenta delle limitazioni nell'utilizzo del percorso di ubicazione quando viene definito un elemento o attributo nel file DAD. Poiché XML Extender utilizza l'associazione uno a uno tra un elemento o attributo e una colonna DB2, le regole di sintassi consentite nel file DAD e nelle funzioni sono limitate. La Tabella 6 descrive le limitazioni per il percorso di ubicazione. I numeri specificati nella colonna del percorso di ubicazione supportata fanno riferimenti agli esempi di sintassi riportati nella sezione "Sintassi del percorso di ubicazione" a pagina 44.

Tabella 6. Limitazioni di XML Extender nell'utilizzo del percorso di ubicazione

Utilizzo del percorso di ubicazione	Percorso di ubicazione supportato
Elemento nell'associazione DAD di colonna XML per tabelle laterali	3, 6 (percorso di ubicazione semplice descritto nella Tabella 5)
UDF di estrazione	1-8[1]
UDF Update	1-8[1]

Tabella 6. Limitazioni di XML Extender nell'utilizzo del percorso di ubicazione (Continua)

Utilizzo del percorso di ubicazione	Percorso di ubicazione supportato
UDF di ricerca Text Extender	1-8

[1] Le UDF di estrazione e aggiornamento supportano percorsi di ubicazione che presentano predicati con attributi, ma non elementi.

File DAD

Per le colonne XML, la DAD specifica innanzitutto il modo in cui indicizzare i documenti memorizzati in una colonna XML. La DAD è un documento formattato in XML che risiede sul client. Se i sceglie di convalidare i documenti XML con una DTD, è possibile associare il file DAD a questa DTD. Il tipo di dati del file DAD è CLOB.Questo file può raggiungere la dimensione di 100 KB.

Il file DAD per le colonne XML contiene un'intestazione XML, che specifica i percorsi di indirizzario sul client per il file DAD e la DTD, e fornisce associazioni per tutti i dati XML memorizzati nelle tabelle laterali per l'indicizzazione.

Per specificare il metodo di accesso e memorizzazione delle colonne XML, utilizzare la seguente tag nel file DAD.

<Xcolumn>

Specifica che i dati XML devono essere memorizzati e richiamati come documenti XML interi nelle colonne DB2 abilitate per i dati XML.

Una colonna abilitata per XML è di tipo UDF XML Extender. Le applicazioni possono includere la colonna in qualsiasi *tabella utente*. E' possibile accedere ai dati della colonna XML attraverso le istruzioni SQL e le UDF XML Extender.

E' possibile utilizzare il wizard di gestione XML Extender oppure un editor per creare e aggiornare la DAD.

Pianificazione per le raccolte XML

Durante la pianificazioni delle raccolte XML, è possibile scegliere di eseguire la composizione dei documenti dai dati DB2, la scomposizione di un documento XML nei dati DB2 o entrambe. Le seguenti sezioni forniscono informazioni sulla pianificazione delle raccolte XML e considerazioni sulla composizione e scomposizione.

Convalida

Dopo aver scelto il metodo di accesso e memorizzazione, è possibile determinare se convalidate i propri dati. Convalidare i dati XML utilizzando una DTD. L'utilizzo di una DTD verifica se il documento XML è valido e consente di eseguire le ricerche strutturali sui dati XML. La DTD viene memorizzata nel magazzino DTD.

Consiglio: Convalidare i dati XML utilizzando una DTD. Per eseguire la convalida, è necessario che il magazzino XML Extender contenga una DTD. Per informazioni su come memorizzazione una DTD nel magazzino, consultare la sezione "Memorizzazione di DTD in un magazzino DTD" a pagina 61. I requisiti DTD differiscono a seconda se si desidera eseguire la composizione o la scomposizione dei documenti XML.

- Per la composizione, è possibile convalidare solo i documenti XML per una DTD. La DTD da utilizzare è specificata nel file DAD.

- Per la scomposizione, è possibile convalidare i documenti per la composizione utilizzando diverse DTD. Quindi, è possibile scomporre i documenti utilizzando lo stesso file DAD, ma con DTD differenti. Per fare riferimento a più DTD, utilizzare le seguenti indicazioni:
  - Almeno una delle DTD deve essere memorizzata nella tabella DTD_REF. Tutte le DTD possono essere memorizzate in questa tabella.
  - Le DTD devono presentare la stessa struttura con elementi secondari differenti.
  - E' necessario specificare la convalida nel file DAD.
  - L'ID di sistema del documento XML deve specificare il file DTD con un nome percorso completo.
  - Il file DAD contiene la specifica su come eseguire la scomposizione del documento e pertanto è possibile specificare solo gli elementi e gli attributi comuni per la scomposizione. Gli elementi e gli attributi univoci in una DTD non possono essere scomposti.

Importante: Specificare se si desidera eseguire la convalida dei dati XML prima di inserire i dati XML nel DB2. XML Extender non supporta la convalida dei dati già inseriti nel DB2.

Considerazioni:

- E' necessario utilizzare una DTD quando si utilizza XML come formato di interscambio.
- L'operazione di convalida dei dati XML può influire in maniera irrilevante sulle prestazioni.
- E' possibile scomporre solo gli elementi e gli attributi comuni quando si utilizzano più DTD per la scomposizione.
- E' possibile scomporre tutti gli elementi e gli attributi quando si utilizza una DTD.
- E' possibile utilizzare una sola DTD per la composizione.

File DAD

Per le raccolte XML, il file DAD associa la struttura del documento XML alle tabelle DB2 da cui è possibile eseguire la composizione o la scomposizione del documento.

Ad esempio, se si specifica un elemento denominato <Tax> nel documento XML, sarà necessario associare <Tax> alla colonna TAX. Definire la relazione tra i dati XML e quelli relazionali nel file DAD.

Il file DAD è specificato durante l'abilitazione di una raccolta o quando viene utilizzato nelle *procedure memorizzate* della raccolta XML. La DAD è un documento formattato in XML che risiede sul client. Se i sceglie di convalidare i documenti XML con una DTD, è possibile associare il file DAD a questa DTD. Se viene utilizzato come parametro di input delle procedure memorizzate XML Extender, il tipo di dati del file DAD è CLOB.Questo file può raggiungere la dimensione di 100 KB.

Per specificare il metodo di accesso e memorizzazione delle raccolte XML, utilizzare la seguente tag nel file DAD:

<Xcollection>
Specifica che i dati XML devono essere composti o scomposti dai o nei documenti XML di una raccolta delle tabelle referenziali.

Una raccolta XML è un nome virtuale per una serie di tabelle relazionali che contengono i dati XML. Le applicazioni possono abilitare una raccolta XML delle tabelle di qualsiasi utente. Queste tabelle utente possono essere delle tabelle di dati aziendali esistenti o nuove tabelle create da XML Extender. E' possibile accedere ai dati delle raccolte XML tramite le procedure memorizzate fornite da XML Extender.

Il file DAD definisce la struttura ad albero dei documenti XML utilizzando i seguenti tipi di nodo:

root_node
Specifica l'elemento root del documento.

element_node
Identifica un elemento come elemento root o elemento secondario.

text_node
Indica il testo CDATA di un elemento.

attribute_node
Rappresenta un attributo di un elemento.

La Figura 10 mostra un esempio del tipo di associazione utilizzato in un file DAD. I nodi associano il contenuto del documento XML alle colonne di tabella in una tabella referenziale.

```
<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM "c:\dtd\dad.dtd">
<DAD>
  ...
<Xcollection>
<SQL_stmt>
       ...
 </SQL_stmt>
<prolog>?xml version="1.0"?</prolog>
<doctype>!DOCTYPE DAD SYSTEM "c:\dxx\sample\dtd\getstart.dtd"</doctype>
<root_node>
   <element_node name="Order">       --> Identifica l'elemento <Order>
     <attribute_node name="key">       --> Identifica l'attributo "key"
        <column name="order_key"/>     --> Definisce il nome della colonna, "order_key",
                                           a cui vengono associati l'elemento e l'attributo.
</attribute_node>
     <element_node name="Customer">  --> Identifica un elemento secondario di <Order> come
                                            <Customer>
        <text_node>                    --> Specifica il testo CDATA per l'elemento
                                            <Customer>
          <column name="customer">     --> Definisce il nome della colonna, "customer",
                                           a cui viene associato l'elemento secondario.
   </text_node>
</element_node>
        ...
</element_node>

      ...
<root_node>
</Xcollection>
</DAD>
```

Figura 10. Definizioni di nodo

In questo esempio, le prime due colonne nell'istruzione SQL presentano elementi e attributi ad esse associati.

XML Extender supporta anche istruzioni di elaborazione per fogli di stile, mediante l'elemento <stylesheet>. Questo deve trovarsi all'interno del nodo root del file DAD, con doctype e prolog definiti per il documento XML. Ad esempio:

```
<Xcollection>
...
<prolog>...</prolog>
<doctype>...</doctype>
<stylesheet>?xml-stylesheet type="text/css" href="order.css"?</stylesheet>
<root_node>...</root_node>
...

</Xcollection>
```

E' possibile utilizzare il wizard di gestione XML Extender oppure un editor per creare e aggiornare il file DAD.L'elemento <stylesheet> non è attualmente supportato dal wizard di gestione XML Extender.

Schemi di associazione per le raccolte XML

Se si sta utilizzando una raccolta XML, occorre selezionare uno *schema di associazione* che definisce la rappresentazione dei dati XML in un database relazionale. Poiché le raccolte XML associano la struttura gerarchica utilizzata nei documenti XML a una struttura relazionale, è necessario comprendere le caratteristiche delle due strutture. La Figura 11 a pagina 50 indica come la struttura gerarchica può essere associata alle colonne della tabella relazionale.

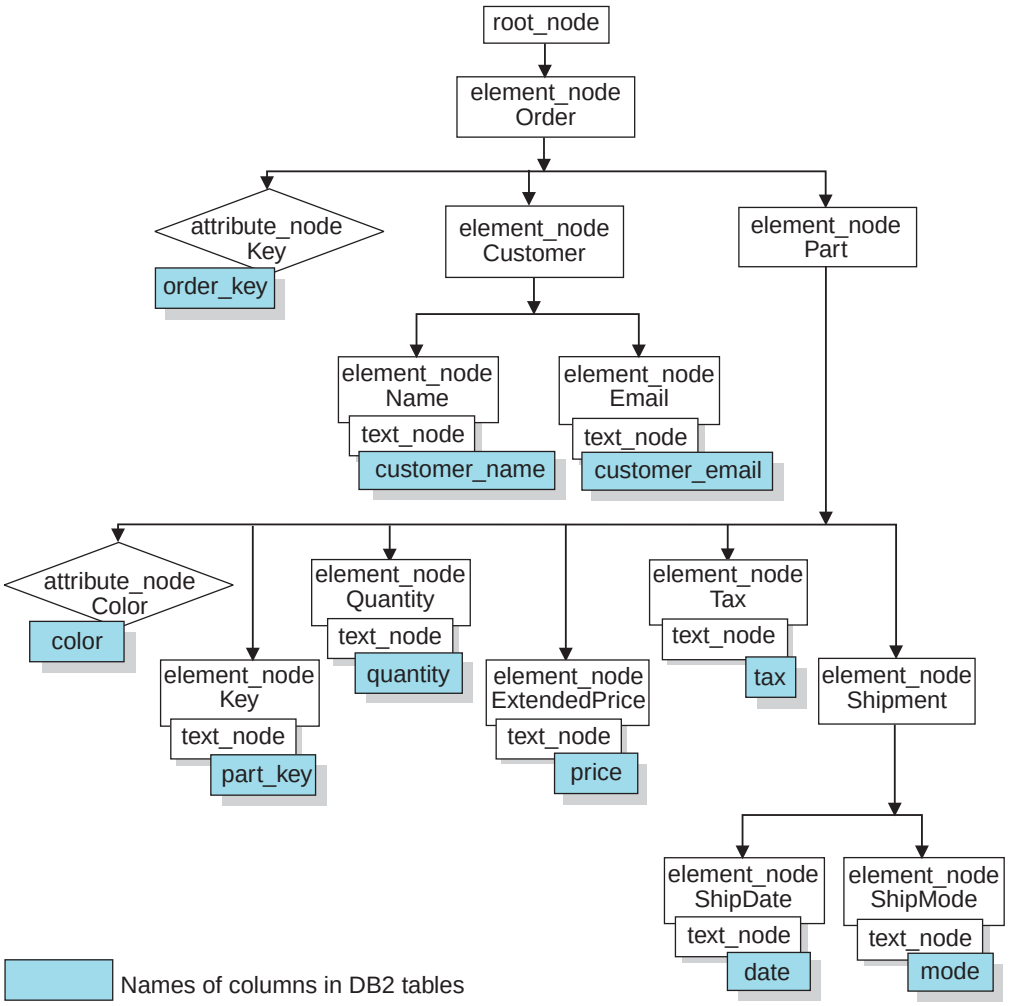


Figura 11. Documento XML strutturato associato alle colonne della tabella relazionale

XML Extender utilizza lo schema di associazione durante la composizione o scomposizione dei documenti XML ubicati in più tabelle relazionali. XML Extender fornisce un wizard di supporto nella creazione del file DAD. Comunque, prima di creare il file DAD, stabilire il modo in cui si desidera associare i dati XML alla raccolta XML.

Tipi di schema di associazione: Lo schema di associazione è specificato nell'elemento <Xcollection> del file DAD. XML Extender fornisce due tipi di schemi di associazione: *associazione SQL* e *associazione RDB_node*. Entrambi i metodi utilizzano il modello XSLT per definire la gerarchia del documento XML.

Associazione SQL

Consente l'associazione semplice e diretta tra i dati relazionali e i documenti XML tramite una singola istruzione SQL e il *modello di dati XSLT*. L'associazione SQL viene utilizzata solo per la composizione e non per la scomposizione. L'associazione SQL è definita dall'elemento SQL_stmt nel file DAD. Il contenuto di SQL_stmt è un'istruzione SQL valida. SQL_stmt associa le colonne della clausola SELECT agli elementi o attributi XML utilizzati nel documento XML. Se l'associazione SQL viene definita per la composizione dei documenti XML, i nomi colonna nella clausola SELECT dell'istruzione SQL vengono utilizzati per definire il

valore di *attribute_node* o il contenuto di *text_node*. La clausola FROM definisce le tabelle contenenti i dati; la clausola WHERE specifica le *condizioni* di ricerca e *collegamento*.

L'associazione SQL fornice agli utenti DB2 la possibilità di associare i dati utilizzando SQL. Quando si utilizza l'associazione SQL, occorre unire tutte le tabelle in un'istruzione SELECT per formare un'interrogazione. Se una sola istruzione SQL risulta insufficiente, utilizzare l'associazione RDB_node. Per raggruppare tutte le tabelle, si consiglia di eseguire l'associazione tra la *chiave primaria* e la *chiave esterna* di queste tabelle.

Associazione RDB_node

Definisce l'ubicazione del contenuto di un elemento XML o il valore di un attributo XML in modo che XML Extender possa determinare dove memorizzare o richiamare i dati XML.

RDB_node che contiene una o più definizione di nodo relative alle tabelle, alle colonne e alle condizioni facoltative. Le tabelle e le colonne vengono utilizzate per definire come i dati XML devono essere memorizzati nel database. La condizione specifica i criteri di selezione dei dati XML o le modalità di unione delle tabelle della raccolta XML.

Per definire uno schema di associazione, creare una DAD con un elemento <Xcollection>. La Figura 12 a pagina 52 mostra un file DAD di esempio con l'associazione SQL delle raccolte XML che compone una serie di documenti XML dai dati di tre tabelle relazionali.

```
<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM "c:\dtd\dad.dtd">
<DAD>
  <dtdid>c:\dxx\samples\dad\getstart.dtd</dtdid>
  <validation>YES</validation>
<Xcollection>
<SQL_stmt>
       SELECT o.order_key, customer, p.part_key, quantity, price, tax, date,
              mode, comment
         FROM order_tab o, part_tab p,
               table(select substr(char(timestamp(generate_unique())),
               as ship_id, date, mode, from ship_tab) as s
        WHERE p.price > 2500.00 and s.date > "1996-06-01" AND
              p.order_key = o.order_key and s.part_key = p.part_key
 </SQL_stmt>
<prolog>?xml version="1.0"?</prolog>
    <doctype>!DOCTYPE DAD SYSTEM "c:\dxx\samples\dtd\getstart.dtd"</doctype>
    <root_node>
   <element_node name="Order">
  <attribute_node name="key">
<column_name="order_key"/>
  </attribute_node>
  <element_node name="Customer">
  <text_node>
  <column name="customer"/>
  </text_node>
  <element_node>

...

  </element_node><!-end Part->
   </element_node><!-end Order->
    </root_node>
</Xcollection>
</DAD>
```

Figura 12. schema di associazione SQL

XML Extender fornice diverse procedure memorizzate che gestiscono i dati in una raccolta XML. Queste procedure memorizzate supportano entrambi i tipi di associazione, ma richiedono che il file DAD rispetti le regole descritte nella sezione “Requisiti dello schema di associazione”.

Requisiti dello schema di associazione: Le seguenti sezioni descrivono i requisiti per ogni tipo di schema di associazione delle raccolte XML.

Requisiti per l’associazione SQL

In questo schema di associazione, occorre specificare l’elemento SQL_stmt nell’elemento <Xcollection> DAD. SQL_stmt deve contenere una singola istruzione SQL che possa unire più tabelle relazionali al *predicato* dell’interrogazione. Inoltre, sono richieste le seguenti clausole:

- **Clausola SELECT**
  - Accertarsi che il nome della colonna sia univoco. Se due tabelle hanno lo stesso nome colonna, utilizzare la parola chiave AS per creare un nome alias per una di loro.
  - Raggruppare le colonne della stesa tabella e utilizzare il livello di gerarchia logico delle tabelle relazionali. Quindi, è necessario raggruppare le tabelle per livello di importanza quando vengono associate alla struttura gerarchica del documento XML. Nella clausola

SELECT, le colonne delle tabelle di alto livello devono precedere quelle di livello inferiore. Il seguente esempio mostra la relazioni gerarchica tra le tabelle:

```
SELECT o.order_key, customer, p.part_key, quantity, price, tax,
       ship_id, date, mode
```

In questo esempio, `order_key` e `customer` della tabella ORDER_TAB hanno il livello relazionale più alto poiché si trovano nella parte iniziale della struttura gerarchica del documento XML. Gli elementi `ship_id`, `date` e `mode` della tabella SHIP_TAB presentano un livello relazionale inferiore.

– Utilizzare una chiave candidata come singola colonna per ogni livello. Se questa chiave non è disponibile in una tabella, l'interrogazione ne deve generare una per la tabella che utilizza un'espressione e la funzione integrata generate_unique(). Nell'esempio precedente, `o.order_key` è la chiave primaria di ORDER_TAB e `part_key` è la chiave primaria di PART_TAB. Questi elementi vengono visualizzati all'inizio del proprio gruppo di colonne da selezionare. Poiché la tabella SHIP_TAB non ha una chiave primaria, occorre crearne una, ad esempio `ship_id`. Questa viene elencata come prima colonna per il gruppo di tabelle SHIP_TAB. Utilizzare la clausola FROM per generare la colonna della chiave primaria, come riportato nel seguente esempio.

- **Clausola FROM**
  – Utilizzare un'espressione di tabella e la funzione, generate_unique(), per creare una singola chiave per le tabelle che non hanno una chiave primaria singola. Ad esempio:

```
FROM order_tab as o, part_tab as p,
       table(select substr(char(timestamp(generate_unique())),16) as
       ship_id, date, mode from ship_tab) as s
```

In questo esempio, la funzione generate_unique() viene collegata a un tipo dati CHAR di TIMESTAMP e le viene assegnato un alias denominato `ship_id`.

  – Utilizzare un nome alias, se necessario, per distinguere una colonna. Ad esempio, è possibile utilizzare `o` per ORDER_TAB, `p` per PART_TAB e `s` per SHIP_TAB.

- **Clausola WHERE**
  – Specificare una relazione tra la chiave primaria e quella esterna come condizione di collegamento per raggruppare le tabelle nella raccolta. Ad esempio:

```
WHERE p.price > 2500.00 AND s.date > "1996-06-01" AND
      p.order_key = o.order_key AND s.part_key = p.part_key
```

  – Specificare un'altra condizione di ricerca nel predicato. E' possibile utilizzare un qualsiasi predicato valido.

- **Clausola ORDER BY**
  – Definire la clausola ORDER BY alla fine di SQL_stmt.
  – Accertarsi che i nomi colonna corrispondano a quelli specificati nella clausola SELECT.
  – Specificare i nomi colonna o gli identificativi che identificano le entità in modo univoco nella progettazione della relazione tra entità del

database. E' possibile generare un identificativo utilizzando un'espressione di tabella e la funzione integrata generate_unique o una funzione definita dall'utente.

– Rispettare l'ordine gerarchico decrescente delle entità. La colonna specificata nella clausola ORDER BY deve essere la prima colonna elencata per ogni entità. Se si rispetta l'ordine, i nuovi documenti XML non conterranno duplicati non validi.
– Non specificare le colonne in ORDER BY per nome tabella o schema.

Anche se SQL_stmt presenta tali caratteristiche, le relative funzionalità non risultano ridotte poiché è possibile specificare un qualsiasi predicato nella clausola WHERE. In tal caso, è però necessario che l'espressione del predicato utilizzi le colonne delle tabelle.

Requisiti per l'utilizzo dell'associazione RDB_node

Se si specifica questo metodo di associazione, non utilizzare SQL_stmt nell'elemento <Xcollection> del file DAD. Utilizzare invece l'elemento RDB_node in ciascun nodo iniziale per *element_node* e per ogni attribute_node e text_node.

- **RDB_node per element_node**

  Il primo *element_node* nel file DAD rappresenta l'elemento root del documento XML. Specificare un RDB_node per il primo element_node nel seguente modo:

  – Specificare tutte le tabelle associate ai documenti XML. Ad esempio, la seguente associazione mostra tre tabelle nell'elemento RDB_node di element_node <Order>, che rappresenta il primo element_node:

```
<element_node name="Order">
        <RDB_node>
        <table name="order_tab"/>
            <table name="part_tab"/>
        <table name="ship_tab"/>
            <condition>
order_tab.order_key = part_tab.order_key AND
part_tab.part_key = ship_tab.part_key
            </condition>
        </RDB_node>
```

  L'elemento condition può essere vuoto oppure può mancare se nella raccolta è presente una sola tabella.

  – Durante la scomposizione, o l'abilitazione della raccolta XML specificata dal file DAD, è necessario specificare una chiave primaria per ciascuna tabella. La chiave primaria è costituita da una o più colonne denominate chiavi composte. Per specificare la chiave primaria, aggiungere una chiave di attributo all'elemento tabella di RDB_node. Quando viene fornita una chiave composta, l'attributo chiave è specificato dai nomi delle colonne chiave separate da uno spazio. Ad esempio:

```
<table name="part_tab" key="part_key, price"/>
```

  Le informazioni specificate per la scomposizione vengono ignorate durante la composizione di un documento.

  – Utilizzare l'attributo orderBy per ricomporre i documenti XML contenenti gli elementi o gli attributi a ricorrenza multipla riportandoli alla struttura originale. Questo attributo consente di specificare il nome di una colonna che sarà la chiave utilizzata per

conservare l'ordine del documento. L'attributo orderBy fa parte dell'elemento di tabella nel file DAD ed è un attributo facoltativo.

Specificare le singole lettere del nome tabella e del nome colonna.

- **RDB_node per ogni attribute_node e text_node**

  In questo schema di associazione, i dati risiedono negli elementi attribute_node e text_node per ogni element_node. Quindi, è necessario specificare a XML Extender la posizione dei dati nel database per poter eseguire la ricerca. Specificare un RDB_node per ogni attribute_node e text_node, indicando alla procedura memorizzata la tabella, la colonna e la condizione in base alle quali è possibile richiamare i dati. Specificare i valori colonna e tabella; il valore della condizione è facoltativo.

  – Specificare il nome della tabella contenente i dati della colonna. Il nome tabella deve essere incluso nell'elemento RDB_node del primo element_node. In questo esempio, per l'elemento text_node di <Price>, la tabella è specificata come PART_TAB.

  ```
  <element_node name="Price">
     <text_node>
          <RDB_node>
              <table name="part_tab"/>
              <column name="price"/>
              <condition>
                   price > 2500.00
              </condition>
          </RDB_node>
     </text_node>
  </element_node>
  ```

  – Specificare il nome della colonna che contiene i dati del testo dell'elemento. Nell'esempio precedente, la colonna è specificata come PRICE.

  – Specificare una condizione se si desidera che i documenti XML vengano generati utilizzando la condizione di interrogazione. Nell'esempio precedente, la condizione è specificata come `price > 2500.00`. Verranno inclusi nei documenti XML generati solo i dati che soddisfano la condizione. La condizione deve essere una clausola WHERE valida.

  – Se si sta eseguendo la scomposizione di un documentoo l'abilitazione della raccolta XML specificata dal file DAD, è necessario specificare il tipo di colonna per ogni attribute_node e text_node. Sarà così garantito l'utilizzo del tipo di dati corretto per ogni colonna quando verranno create delle nuove tabelle durante l'abilitazione di una raccolta XML. I tipi di colonna vengono specificati aggiungendo il tipo di attributo all'elemento colonna. Ad esempio,

  ```
  <column name="order_key" type="integer"/>
  ```

  Le informazioni specificate per la scomposizione vengono ignorate durante la composizione di un documento.

Se si utilizza l'associazione RDB_node, non occorre specificare le istruzioni SQL. Tuttavia, l'inserimento di condizioni di collegamento complesse nell'elemento RDB_node può risultare molto difficile. Ad esempio, l'utilizzo di un'operazione o di un'espressione *union* risulta meno efficace di SQL per XML.

Requisiti per la scomposizione delle dimensioni della tabella

La scomposizione utilizza l'associazione RDB_node per specificare come un documento XML si scompone in tabelle DB2, estraendo i valori attributo ed

elemento nelle righe della tabella. I valori di ciascun documento XML vengono memorizzati in uno o più tabelle DB2. Una tabella può presentare massimo 1024 righe scomposte da ciascun documento.

Ad esempio, se un documento XML si scompone in cinque tabelle, ogni tabella può presentare massimo 1024 righe per quel determinato documento. Se la tabella presenta righe per più documenti, è possibile che contenga massimo 1024 righe per ciascun documento. Se la tabella presenta 20 documenti, è possibile che contenga 20.480 righe, 1024 per ciascun documento.

L'utilizzo di elementi con ricorrenza multipla (elementi con percorsi di ubicazione multipla nella struttura XML) incide sul numero di righe. Ad esempio, un documento che contiene un elemento <Part> che ricorre 20 volte, è necessario che sia scomposto in 20 righe in una tabella. Quando si utilizzano elementi con ricorrenza multipla, considerare questa restrizione delle dimensioni di tabella.

Capitolo 4. Gestione dei dati XML

Le attività di gestione XML Extender consentono di abilitare il database e le colonne di tabella per XML e di associare i dati XML alle strutture relazionali DB2. XML Extender fornisce diversi strumenti di gestione che variano a seconda delle attività che si desidera eseguire, quali lo sviluppo delle applicazioni, l'esecuzione delle attività di gestione o l'utilizzo di un wizard. E' possibile utilizzare i seguenti strumenti per completare le attività di gestione per XML Extender:

- Wizard di gestione XML Extender
- Comando **dxxadm**
- Procedure memorizzate di gestione XML Extender

Questo capitolo descrive le attività di gestione associate al wizard di gestione e al comando **dxxadm**. Le procedure memorizzate di gestione vengono descritte in "Procedure memorizzate di gestione" a pagina 175.

Per completare le attività riportate in questo capitolo, è necessaria la conoscenza dei concetti e delle attività di pianificazione descritti nella sezione "Pianificazione della gestione" a pagina 38.

Le seguenti sezioni descrivono le attività di gestione XML Extender:

1. "Avvio del wizard di gestione"
2. "Abilitazione di un database per XML" a pagina 60
3. "Memorizzazione di DTD in un magazzino DTD" a pagina 61
4. "Definizione delle colonne o delle raccolte XML" a pagina 62
5. "Operazioni con le colonne XML" a pagina 62
6. "Operazioni con le raccolte XML" a pagina 72

Avvio del wizard di gestione

Questa sezione fornisce informazioni sull'installazione e il richiamo del wizard di gestione XML Extender.

Installazione del wizard di gestione

Seguire attentamente le istruzione sull'installazione e sulla configurazione del wizard di gestione riportate nel file readme relativo al proprio sistema operativo. Eseguire l'istruzione bind e includere il software richiesto nelle istruzioni CLASSPATH.

- Le istruzioni bind vengono fornite nei file readme del wizard e nel file di esempio della guida introduttiva:

  `/dxx_install/samples/cmd/getstart_prep.cmd`

- L'istruzione CLASSPATH è simile a quanto riportato di seguito (le interruzioni di riga sono dovute solo a motivi di presentazione):

```
.;C:\java\db2java.zip;C:\java\runtime.zip;C:\java\sqlj.zip;
C:\dxx\dxxadmin\dxxadmin.jar;C:\dxx\dxxadmin\dxxadmin.cmd;
C:\dxx\dxxadmin\html\dxxahelp*.htm;C:\java\jdk\lib\classes.zip;
C:\java\swingall.jar
```

  Importante: Il wizard richiede un nome di percorso senza alcuno spazio. Se si dispone dell'installazione predefinita di IBM DB2 Universal Database V7.1,

SQLLIB\java è ubicato nell'indirizzario Program Files; *copiare* il codice Java in un percorso più semplice. Non spostare il codice Java e non modificare CLASSPATH; durante l'installazione il Control Center richiede che sia specificato CLASSPATH.

Il wizard di gestione XML Extender utilizza un file di classe. Il nome file completo del file di classe di gestione principale XML Extender è:

```
com.ibm.dxx.admin.Admin.
```

Modificare questo file per fare in modo che il proprio sistema richiami il wizard.

- Per il richiamo mediante JDK, immettere:

  ```
  java -classpath classpath com.ibm.dxx.admin.Admin
  ```

- Per il richiamo mediante JRE, immettere:

  ```
  jre -classpath classpath com.ibm.dxx.admin.Admin
  ```

  dove *classpath* può indicare:

  – La variabile di ambiente %CLASSPATH% per specificare dove sono sistemati i file di classe del wizard di gestione. Quando si utilizza questa opzione, CLASSPATH deve puntare all'indirizzario *dxx_install*/dxxadmin, che contiene i seguenti file: dxxadmin.jar, xml4j.jar e db2java.zip. Ad esempio:

  ```
  java -classpath %CLASSPATH% com.ibm.dxx.admin.Admin
  ```

  – Una sovrapposizione della variabile di ambiente %CLASSPATH% con puntatori verso i file contenuti nell'indirizzario *dxx_install*/dxxadmin, da cui si sta eseguendo il wizard di gestione di XML Extender. Ad esempio:

  ```
  java -classpath dxxadmin.jar;xml4j.jar;db2java.zip com.ibm.dxx.admin.Admin url=jdbc:db2:mydb
     userid=db2xml password=db2xml driver=COM.ibm.db2.jdbc.app.DB2Driver
  ```

  Facoltativamente, è possibile specificare i seguenti parametri al runtime:

  url Il percorso URL completo per l'origine dati IBM DB2 UDB a cui collegarsi. Ad esempio: jdbc:db2://dxx.stl.ibm.com:8080/guidb. Nel wizard viene etichettato come "Indirizzo".

  userid L'ID utente da utilizzare per accedere all'origine dati. Ad esempio: db2guest.

  password
  La parola d'ordine dell'utente precedente. Ad esempio: guest.

  driver Nome del driver JDBC per l'URL sopra riportato. Valore predefinito: COM.ibm.db2.jdbc.net.DB2Driver. Nel wizard viene etichettato come "Driver JDBC".

  Consultare "Richiamo del wizard di gestione" per ulteriori informazioni su questi valori.

Richiamo del wizard di gestione

Attenersi alla seguente procedura per richiamare il wizard di gestione XML Extender.

1. Richiamare il wizard.

   Per Windows NT:

   Fare doppio clic sull'icona relativa al wizard di gestione XML Extender dal desktop.

   Per AIX, Sun Solaris e Linux:

   Eseguire il file dxxadmin.

Viene aperta la finestra di collegamento al wizard di gestione.

Quando si richiama il wizard di gestione XML Extender, viene visualizzata la finestra di collegamento. Collegarsi al database che si desidera utilizzare per la gestione dei dati XML. XML Extender si collega all'istanza corrente.

2. Nel campo **Indirizzo**, immettere l'URL JDBC completo per l'origine dati IBM DB2 UDB a cui ci si collega. L'indirizzo presenta la seguente sintassi:

   Per configurazioni autonome (consigliate):

   ```
   jdbc:db2:database_name
   ```

   Dove:

   database_name
   Il database a cui ci si sta collegando e in cui si stanno memorizzando i documenti XML.

   Ad esempio,

   ```
   jdbc:db2:sales_db
   ```

   Per configurazioni di rete:

   ```
   jdbc:db2://server_name:port_number/database_name
   ```

   Dove:

   server_name
   E' il nome del server su cui è ubicato XML Extender.

   port_number
   Il numero di porta utilizzato per collegarsi al server. Per individuare il numero di porta, immettere il seguente comando da una riga comandi DB2 sulla macchina server:

   ```
   db2jstrt port#
   ```

   Gli utenti Windows NT possono verificare il numero di porta nel file \winnt\system32\driver\etc\services.

   database_name
   Il database a cui ci si sta collegando e in cui si stanno memorizzando i documenti XML.

   Ad esempio,

   ```
   jdbc:db2://host1.ibm.com:8080/sales_db
   ```

3. Nei campi **ID utente** e **Parola d'ordine**, immettere o verificare la parola d'ordine e l'ID utente DB2 del database a cui ci si sta collegando.
4. Nel campo **Driver JDBC**, verificare il nome del driver JDBC per l'indirizzo specificato utilizzando i seguenti valori:

   Per configurazioni autonome (predefinite e consigliate):

   ```
   COM.ibm.db2.jdbc.app.DB2DRIVER
   ```

   Per le configurazioni di rete:

   ```
   COM.ibm.db2.jdbc.net.DB2DRIVER
   ```

5. Fare clic su **Fine** per collegarsi al wizard e passare alla finestra LaunchPad.

La finestra LaunchPad consente l'accesso a cinque wizard di gestione. Questi wizard consentono di:

- Abilitare un database
- Aggiungere DTD al magazzino DTD
- Gestire i file DAD per:
  - colonne XML
  - raccolte XML
- Gestire le colonne XML
- Gestire le raccolte XML

Abilitazione di un database per XML

Per memorizzare e richiamare i documenti XML dal DB2 mediante XML Extender, abilitare il database per XML. XML Extender abilita il database a cui si è collegati utilizzando l'istanza corrente.

Quando si abilita un database per XML, XML Extender:

- Crea tutti i tipi definiti dall'utente (UDT) e le funzioni definite dall'utente (UDF)
- Crea e compila le tabelle di controllo utilizzando i metadati necessari richiesti da XML Extender
- Crea lo schema db2xml con i privilegi richiesti

  Il nome completo di una funzione XML è *schema-name.function-name*, dove *schema-name* è un identificativo che fornisce un raggruppamento logico per gli *oggetti* SQL. E' possibile utilizzare il nome completo dovunque si faccia riferimento a una UDF o un UDT. Inoltre, è possibile omettere il nome schema quando si fa riferimento a UDF o UDT. In questo caso, DB2 utilizza il percorso della funzione per stabilire la funzione o il tipo di dati desiderato.

Utilizzo del wizard di gestione

Eseguire i passi riportati di seguito per abilitare un database per i dati XML:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Abilitare database** dalla finestra LaunchPad per abilitare il database corrente.

   Se un database è già abilitato, l'opzione **Disabilitare un database** è selezionabile.

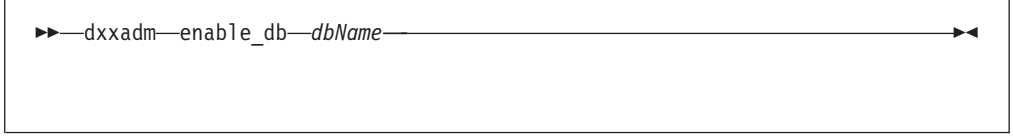
   Una volta abilitato il database, si ritorna alla finestra LaunchPad.

Shell dei comandi DB2

Immettere **dxxadm** dalla riga comandi, specificando il database che si desidera abilitare.

Sintassi:

dxxadm enable_db



Parametri:

dbName
Il nome del database che si desidera abilitare.

Esempio: Per abilitare un database esistente denominato SALES_DB, immettere:

```
dxxadm enable_db SALES_DB
```

Memorizzazione di DTD in un magazzino DTD

E' possibile utilizzare una DTD per convalidare i dati XML in una raccolta o colonna XML. DTD convalida la colonna XML e definisce i file DAD utilizzati per la ricerca strutturale XML e per la composizione delle raccolte.

Tutte le DTD vengono memorizzate nel magazzino DTD, una tabella DB2 denominata DTD_REF. Il nome schema della tabella è db2xml. Ogni DTD nella tabella DTD_REF ha un ID univoco. XML Extender crea la tabella DTD_REF quando si abilita un database per XML.

Per ulteriori informazioni sull'utilizzo delle DTD, consultare "Pianificazione per le colonne XML" a pagina 40 e "Pianificazione per le raccolte XML" a pagina 46.

E' possibile inserire la DTD dalla shell dei comandi DB2 oppure utilizzando il wizard di gestione.

Utilizzo del wizard di gestione

Effettuare i seguenti passi per inserire una DTD:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Importare una DTD** dalla finestra LaunchPad per importare un file DTD esistente nel magazzino DTD del database corrente. Viene visualizzata la finestra Importare una DTD.
3. Immettere il nome file DTD nel campo **Nome file DTD** oppure fare clic su **...** per sfogliare e selezionare un file esistente.
4. Immettere l'ID DTD nel campo **ID DTD**.

   L'ID DTD è un identificativo della DTD e può essere il percorso che specifica l'ubicazione della DTD sul sistema locale. L'ID DTD deve corrispondere al valore specificato nel file DAD per l'elemento <DTDID>.
5. Facoltativamente, immettere il nome dell'autore della DTD nel campo **Autore**.

   XML Extender visualizza automaticamente il nome dell'autore se è specificato nella DTD.
6. Fare clic su **Fine** per inserire la DTD nella tabella del magazzino DTD, DB2XML.DTD_REF e ritornare alla finestra LaunchPad.

Shell dei comandi DB2

Immettere un'istruzione SQL INSERT per la tabella DTD_REF utilizzando lo schema in Tabella 7 a pagina 62:

Tabella 7. Schema per la tabella DTD_REF DTD

Nome colonna	Tipo di dati	Descrizione
DTDID	VARCHAR(128)	Chiave primaria (univoca e non nulla). La chiave primaria viene utilizzata per identificare la DTD e deve essere uguale a quella definita per SYSTEM ID alla riga DOCTYPE di ogni documento XML, quando viene specificata la convalida. Quando viene specificata la chiave primaria nel file DAD, questo file deve rispettare lo schema definito dalla DTD.
CONTENT	XMLCLOB	Il contenuto della DTD.
USAGE_COUNT	INTEGER	Il numero di colonne e raccolte XML nel database che utilizzano questa DTD per definire una DAD.
AUTHOR	VARCHAR(128)	L'autore della DTD, informazioni facoltative per l'utente da immettere.
CREATOR	VARCHAR(128)	L'ID utente che esegue il primo inserimento.
UPDATOR	VARCHAR(128)	L'ID utente che esegue l'ultimo aggiornamento.

Ad esempio:

```
DB2 INSERT into db2xml.dtd_ref values('c:\dxx\samples\dtd\getstart.dtd',
    db2xml.XMLClobFromFile('c:\dxx\samples\dtd\getstart.dtd'), 0, 'user1',
    'user1', 'user1')
```

Per le raccolte XML: L'ID DTD è un percorso che specifica l'ubicazione della DTD sul sistema locale. L'ID DTD deve corrispondere al valore specificato nel file DAD per l'elemento <DTDID>.

Definizione delle colonne o delle raccolte XML

Le sezioni riportate di seguito descrivono come impostare e definire il database per le colonne o le raccolte XML e come preparare gli schemi di associazione dati necessari.

Queste sezioni sono:

- "Operazioni con le colonne XML"
- "Operazioni con le raccolte XML" a pagina 72

Operazioni con le colonne XML

Per impostare le colonne XML, è necessario definire il file DAD per accedere ai dati XML e per abilitare le colonne per questi dati in una tabella XML. Un importante concetto nella creazione della DAD è la comprensione della sintassi del percorso di ubicazione, in quanto viene utilizzata per associare i valori dell'elemento e dell'attributo alle tabelle DB2. Per ulteriori informazioni sulla sintassi del percorso di ubicazione, consultare la sezione "Percorso di ubicazione" a pagina 44.

Creazione o editazione del file DAD

Quando si specifica un file DAD, vengono definiti gli attributi e gli elementi chiave dei dati che si desidera ricercare. XML Extender utilizza queste informazioni per

creare le tabelle laterali in cui vengono ordinati i dati per poterli richiamare velocemente. Per ulteriori informazioni sulla pianificazione della creazione del file DAD, consultare la sezione “File DAD” a pagina 46.

Informazioni preliminari

- Esaminare la struttura gerarchica dei dati XML in modo da poter definire gli elementi chiave e gli attributi per la creazione dell'indice e l'ottimizzazione delle funzioni di ricerca.
- Preparare ed inserire la DTD del documento XML nella tabella DTD_REF. Questo passo è richiesto per la convalida.

Utilizzo del wizard di gestione

Eseguire i passi riportati di seguito per creare un file DAD:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare “Avvio del wizard di gestione” a pagina 57.
2. Fare clic su **Operazioni con i file DAD** dalla finestra LaunchPad per editare o creare un file DAD XML. Viene visualizzata la finestra Specificare una DAD.
3. Scegliere se editare un file DAD esistente oppure creare un nuovo file DAD.
   - **Per editare un file DAD esistente:**
     a. Fare clic su **...** per sfogliare e selezionare un file DAD esistente nel menu a discesa oppure immettere il nome del file DAD nel campo **Nome file**.
     b. Verificare se il wizard riconosce il file DAD specificato.
        – Se il wizard riconosce il file DAD specificato, **Avanti** è selezionabile e la colonna XML viene visualizzata nel campo **Tipo**.
        – Se il wizard non riconosce il file DAD specificato, non è possibile selezionare **Avanti**. Immettere nuovamente il nome del file DAD nel campo **Nome file** oppure fare clic su **Aprire** per selezionare nuovamente un file DAD esistente. Continuare fino a quando **Avanti** diventa selezionabile.
     c. Fare clic su **Avanti**.
   - **Per creare una nuova DAD:**
     a. Lasciare il campo **Nome file** vuoto.
     b. Dal menu **Tipo**, fare clic su **Colonna XML**.
     c. Fare clic su **Avanti**.
4. Nella finestra Selezionare convalida, scegliere se convalidare i documenti XML con una DTD.
   - Per convalidare:
     a. Fare clic su **Convalidare i documenti XML con la DTD**.
     b. Selezionare la DTD da utilizzare per la convalida dall'elenco **ID DTD**.

     Se non è stata importata alcuna DTD nel magazzino DTD del database, non è possibile convalidare i documenti XML.
   - Fare clic su **NON convalidare i documenti XML con la DTD** per continuare senza convalidare i documenti XML.
5. Fare clic su **Avanti**.
6. Nella finestra Tabelle laterali scegliere se aggiungere una nuova tabella laterale, editare oppure eliminare una tabella laterale esistente.
   - **Per aggiungere una nuova tabella laterale o una colonna di tabella laterale:**

Per aggiungere una nuova tabella laterale, definire le colonne nella tabella. Completare i passi riportati di seguito per ciascuna colonna in una tabella laterale.

a. Completare i campi della casella **Dettagli** presente nella finestra Tabelle laterali.
   1) **Nome tabella**: Immettere il nome della tabella contenente la colonna. Ad esempio:
      ```
      ORDER_SIDE_TAB
      ```
   2) **Nome colonna**: Immettere il nome della colonna. Ad esempio:
      ```
      CUSTOMER_NAME
      ```
   3) **Tipo**: Selezionare il tipo di colonna dal menu. Ad esempio:
      ```
      XMLVARCHAR
      ```
   4) **Lunghezza (solo per il tipo VARCHAR)**: Immettere il numero massimo di caratteri di VARCHAR. Ad esempio:
      ```
      30
      ```
   5) **Percorso**: Immettere il percorso di ubicazione dell'elemento o dell'attributo. Ad esempio:
      ```
      /ORDER/CUSTOMER/NAME
      ```

      Per informazioni sulla sintassi del percorso di ubicazione, consultare la sezione "Percorso di ubicazione" a pagina 44.
   6) **Più ricorrenze**: Selezionare **No** o **Sì** dal menu.

      Indica se il percorso di ubicazione di questo elemento o attributo può essere utilizzato più volte all'interno di un documento.

      Importante Se si specifica l'opzione Più ricorrenze per una colonna, è possibile specificare solo una colonna all'interno della tabella laterale contenente tale colonna.

b. Fare clic su **Aggiungere** per aggiungere una colonna.

c. Continuare ad aggiungere, editare o eliminare le colonne della tabella laterale oppure fare clic su **Avanti**.

- **Per editare una colonna di tabella laterale:**

  E' possibile aggiornare una tabella laterale modificando le definizioni delle colonne esistenti.

  a. Fare clic sulla tabella laterale e sul nome colonna che si desidera editare.
  b. Editare i campi della casella **Dettagli**.
  c. Fare clic su **Modificare** per salvare le modifiche.
  d. Continuare ad aggiungere, editare o eliminare le colonne di ciascuna tabella laterale oppure fare clic su **Avanti**.

- **Per eliminare una colonna di tabella laterale:**

  a. Fare clic sulla tabella laterale e sulla colonna che si desidera eliminare.
  b. Fare clic su **Eliminare**.
  c. Continuare ad aggiungere, editare o eliminare le colonne delle tabelle laterali oppure fare clic su **Avanti**.

- **Per eliminare una tabella laterale esistente:**

  Per eliminare un'intera tabella laterale, è necessario cancellare tutte le colonne della tabella.

  a. Fare clic su ciascuna colonna della tabella laterale della tabella che si desidera eliminare.
  b. Fare clic su **Eliminare**.

c. Continuare ad aggiungere, editare o eliminare colonne di tabella laterale oppure fare clic su **Avanti**.

7. Immettere un nome file di output per il file DAD modificato nel campo **Nome file** della finestra Specificare una DAD.
8. Fare clic su **Fine** per salvare il file DAD e ritornare alla finestra LaunchPad.

Shell dei comandi DB2

Il file DAD è un file XML che può essere creato in qualsiasi editor di testo.

Eseguire i passi riportati di seguito per creare un file DAD:

1. Aprire un editor di testo.
2. Creare un'intestazione del file DAD utilizzando la seguente sintassi:

```
<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM "path\dtd\dad.dtd"> --> il percorso
e il nome file della
DTD per il file DAD
```

3. Inserire le tag <DAD></DAD>.
4. Nella tag <DAD>, specificare facoltativamente l'identificativo DTD che associa il file DAD alla DTD del documento XML per la convalida.

```
<dtdid>path\dtd_name.dtd</dtdid> --> il percorso e il nome file della DTD
                                     per l'applicazione
```

   L'ID DTD è richiesto per la convalida e deve corrispondere al valore dell'ID DTD utilizzato per l'inserimento della DTD nella tabella di riferimento DTD (db2xml.DTD_REF).

5. Specificare se si desidera utilizzare una DTD per accertarsi che il documento XML è un documento valido. Ad esempio:

```
<validation>YES</validation>   --> specificare Sì o No
```

   Se si specifica Sì, è necessario che nel passo precedente sia stato specificato un ID DTD e che una DTD sia stata inserita nella tabella DTD_REF.

6. Utilizzare l'elemento <Xcolumn> per definire il metodo di memorizzazione e accesso come colonna XML.

```
<Xcolumn>
</Xcolumn>
```

7. Definire ogni tabella laterale e i principali elementi e attributi da indicizzare per la ricerca strutturale. Ripetere i seguenti passi per ciascuna tabella. Questi passi utilizzano gli esempi riportati nel file DAD campione illustrato nella sezione "File DAD: colonna XML" a pagina 224:
   a. Inserire le tag <TABLE></TABLE> e l'attributo nome.

```
<table name="order_tab">
</table>
```

   b. Dopo la tag <TABLE>, inserire una tag <COLUMN> e i relativi attributi per ciascuna colonna della tabella:
      - ***name***: il nome della colonna
      - ***type***: il tipo di colonna
      - ***path***: il percorso di ubicazione dell'elemento o attributo. Per informazioni sulla sintassi del percorso di ubicazione, consultare la sezione "Percorso di ubicazione" a pagina 44.
      - ***multi_occurrence***: indica se questo elemento o attributo può essere utilizzato più volte all'interno di un documento

```
<table ...>
   <column name="order_key"
           type="integer"
           path="/Order/@key"
           multi_occurrence="NO"/>
   <column name="customer"
           type="varchar(50)"
           path="/Order/Customer/Name"
           multi_occurrence="NO"/>
</table>
```

8. Verificare se viene posizionata una tag </TABLE> dopo l'ultima definizione di colonna.
9. Verificare se viene posizionata una tag </Xcolumn> dopo l'ultima tag </TABLE>.
10. Verificare se viene posizionata una tag </DAD> dopo la tag </Xcolumn>.

Creazione o modifica di una tabella XML

Per memorizzare documenti XML intatti in una tabella, è necessario creare o modificare una tabella in modo da contenere una colonna con un UDT XML. La tabella è nota come *tabella XML* contenente documenti XML. La tabella non può essere una tabella modificata o nuova. Quando una tabella contiene una colonna di tipo XML, è possibile abilitare la colonna per XML.

E' possibile modificare una tabella esistente con una colonna di tipo XML utilizzando il wizard di gestione o la shell dei comandi DB2.

Utilizzo del wizard di gestione

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Operazioni con le colonne XML** dalla finestra LaunchPad. Viene visualizzata la finestra Selezionare un'attività.
3. Fare clic su **Aggiungere una colonna XML**. Viene visualizzata la finestra Aggiungere una colonna XML.
4. Selezionare il nome della tabella dal menu a discesa **Nome tabella** oppure immettere il nome della tabella che si desidera modificare. Ad esempio:

   ```
   SALES_DB
   ```

5. Immettere il nome della colonna da aggiungere alla tabella nel campo **Nome colonna**. Ad esempio:

   ```
   ORDER
   ```

6. Selezionare l'UDT per la colonna dal menu a discesa **Tipo colonna**. Ad esempio:

   ```
   XMLVARCHAR
   ```

7. Fare clic su **Fine** per aggiungere la colonna di tipo XML.

Shell dei comandi DB2

Creare o modificare una tabella con una colonna di tipo XML nella clausola dell'istruzione CREATE TABLE o ALTER TABLE.

Esempio: Nell'applicazione sales, è possibile memorizzare un ordine di righe formattate da XML nella colonna ORDER di una tabella applicativa denominata SALES_TAB. Questa tabella contiene anche le colonne INVOICE_NUM e SALES_PERSON. Poiché si tratta di un ordine di piccole dimensioni, memorizzarlo utilizzando il tipo XMLVARCHAR. La chiave primaria è INVOICE_NUM. La seguente istruzione CREATE TABLE crea la tabella con una colonna di tipo XML:

```
CREATE TABLE sales_tab(
             invoice_num  char(6) NOT NULL PRIMARY KEY,
             sales_person     varchar(20),
             order            XMLVarchar);
```

Abilitazione delle colonne XML

Per memorizzare un documento XML in un database DB2, occorre abilitare una colonna per XML. L'abilitazione di una colonna comporta la creazione di un indice in modo da eseguire ricerche più veloci. E' possibile abilitare una colonna utilizzando il wizard di gestione XML Extender o la shell dei comandi DB2. La colonna deve essere di tipo XML.

Per abilitare una colonna XML, XML Extender:

- Legge il file DAD per:
  - Convalidare questo file DAD per la relativa DTD.
  - Richiamare l'ID DTD dalla tabella DTD_REF, se specificato.
  - Creare le tabelle laterali l'indicizzazione delle colonne XML.
  - Preparare la colonna che dovrà contenere i dati XML.
- Facoltativamente, crea una *vista predefinita* della tabella XML e delle tabelle laterali definite.
- Specifica un valore ID ROOT nel caso in cui non sia già stato indicato.

Dopo aver abilitato la colonna XML, sarà possibile:

- Creare indici nelle tabelle laterali
- Inserire documenti XML nella colonna XML
- Interrogare, aggiornare o ricercare i documenti XML nella colonna XML.

Informazioni preliminari

Creare una tabella XML generando o modificando una tabella DB2 con una colonna UDT XML.

Utilizzo del wizard di gestione

Eseguire i passi riportati di seguito per abilitare le colonne XML:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Operazioni con le colonne XML** dalla finestra LaunchPad per visualizzare le attività relative alle colonne XML Extender. Viene visualizzata la finestra Selezionare un'attività.
3. Fare clic su **Abilitare una colonna** e quindi su **Avanti** per abilitare una colonna di tabella esistente nel database.
4. Selezionare la tabella contenente la colonna XML dal campo **Nome tabella**. Ad esempio

   ```
   SALES_TAB
   ```
5. Selezionare la colonna da abilitare dal campo **Nome colonna**. Ad esempio:

   ```
   ORDER
   ```

   La colonna deve esistere ed essere di tipo XML.
6. Immettere il il nome file DAD ed il percorso nel campo **Nome file DAD** oppure fare clic su **...** per selezionare un file DAD esistente. Ad esempio:

   ```
   c:\dxx\samples\dad\getstart.dad
   ```
7. Facoltativamente, immettere il nome di un table space esistente nel campo **Table space**.

Il table space contiene tabelle laterali create da XML Extender. Se si specifica un table space, le tabelle laterali vengono create nel table space specificato. Se non si specifica un table space, le tabelle laterali vengono create nel table space predefinito.

8. Facoltativamente, immettere il nome della vista predefinita nel campo **Vista predefinita**.

   Quando specificato, la vista predefinita viene creata automaticamente quando la colonna viene abilitata e unisce la tabella XML e tutte le relative tabelle laterali.

9. Facoltativamente, immettere il nome colonna della chiave primaria nella tabella applicativa nel campo **ID root**. Questa operazione è consigliata.

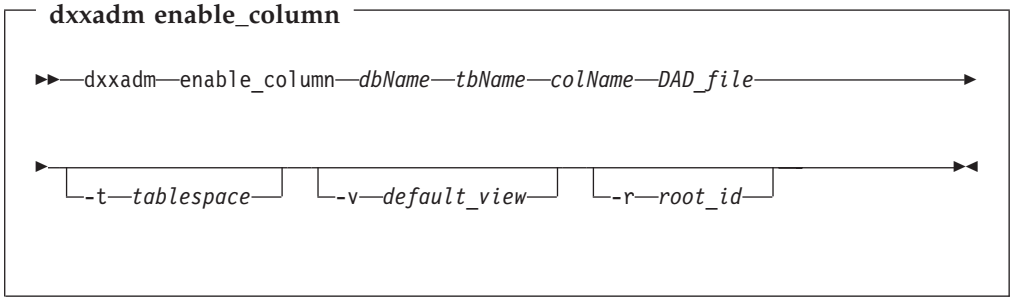
   XML Extender utilizza il valore ID ROOT come identificativo univoco per associare tutte le tabelle laterali alla tabella applicativa. Se non è specificato, XML Extender aggiunge la colonna DXXROOT_ID alla tabella applicativa e genera un identificativo.

10. Fare clic su **Fine** per abilitare la colonna XML, creare le tabelle laterali e ritornare alla finestra LaunchPad.
    - Se la colonna è stata abilitata con esito positivo, viene visualizzato un messaggio `Abilitazione colonna terminata con esito positivo`.
    - Se la colonna non è stata abilitata con esito positivo, viene visualizzata una casella di errore. Correggere i valori nel campo di immissione fino a quando la colonna viene abilitata con esito positivo.

Shell dei comandi DB2

Per abilitare una colonna XML, immettere il seguente comando:

Sintassi:



Parametri:

dbName
: Il nome del database.

tbName
: Il nome della tabella che contiene la colonna che deve essere abilitata.

colName
: Il nome della colonna XML abilitata.

DAD_file
: Il nome del file che contiene la DAD (document access definition).

tablespace
: Un table space creato in precedenza che contiene le tabelle laterali create da XML Extender. Se non viene specificato, viene utilizzato il table space predefinito.

default_view
Facoltativo. Il nome della vista predefinita creata da XML Extender per associare una tabella applicativa a tutte le tabelle laterali correlate.

root_id Facoltativo. Il nome colonna della chiave primaria nella tabella applicativa e un identificativo univoco utilizzati per associare tutte le tabelle laterali alla tabella applicativa. XML Extender utilizza il valore *root_id* come identificativo univoco per associare tutte le tabelle laterali alla tabella applicativa. Si consiglia di specificare l'ID ROOT. Se l'ID ROOT non è specificato, XML Extender aggiunge la colonna DXXROOT_ID alla tabella applicativa e genera un identificativo.

Limitazione: Se la tabella applicativa contiene il nome colonna DXXROOT_ID, ma questa colonna non presenta alcun valore per *root_id*, è necessario specificare il parametro *root_id*, altrimenti si verificherà un errore.

Esempio: Il seguente esempio consente di abilitare una colonna utilizzando la shell dei comandi DB2. Il file DAD e il documento XML possono essere trovati nell'"Appendice B. Esempi" a pagina 223.

```
dxxadm enable_column SALES_DB sales_tab order getstart.dad
           -v sales_order_view -r invoice_num
```

In questo esempio, la colonna ORDER è abilitata nella tabella SALES_DB.SALES_TAB. Il file DAD è `getstart.dad`, la vista predefinita è sales_order_view e l'ID ROOT è INVOICE_NUM.

In questo esempio la tabella SALES_TAB presenta il seguente schema:

Nome colonna	INVOICE_NUM	SALES_PERSON	ORDER
Tipo di dati	CHAR(6)	VARCHAR(20)	XMLVARCHAR

Le seguenti tabelle laterali vengono create in base alla specifica DAD:

ORDER_SIDE_TAB:

Nome colonna	ORDER_KEY	CUSTOMER	INVOICE_NUM
Tipo di dati	INTEGER	VARCHAR(50)	CHAR(6)
Espressione percorso	`/Order/@key`	`/Order/Customer/Name`	N/D

PART_SIDE_TAB:

Nome colonna	PART_KEY	PRICE	INVOICE_NUM
Tipo di dati	INTEGER	DOUBLE	CHAR(6)
Espressione percorso	`/Order/Part/@key`	`/Order/Part/ExtendedPrice`	N/D

SHIP_SIDE_TAB:

Nome colonna	DATE	INVOICE_NUM
Tipo di dati	DATE	CHAR(6)

Nome colonna	DATE	INVOICE_NUM
Espressione percorso	/Order/Part/Shipment/ShipDate	N/D

Tutte le tabelle laterali hanno lo stesso tipo di colonna INVOICE_NUM, perché l'ID ROOT è stato specificato dalla chiave primaria INVOICE_NUM nella tabella applicativa. Dopo aver abilitato la colonna, il valore INVOICE_NUM viene inserito nelle tabelle laterali. Se si specifica il parametro *default_view* durante l'abilitazione della colonna XML, ORDER, viene creata la vista predefinita sales_order_view. Questa vita associa le tabelle precedenti utilizzando la seguente istruzione:

```
CREATE VIEW sales_order_view(invoice_num, sales_person, order,
                                order_key, customer, part_key, price, date)
AS
     SELECT sales_tab.invoice_num, sales_tab.sales_person, sales_tab.order,
            order_tab.order_key, order_tab.customer,
            part_tab.part_key, part_tab.price,
            ship_tab.date
     FROM sales_tab, order_tab, part_tab, ship_tab
     WHERE sales_tab.invoice_num = order_tab.invoice_num
               AND sales_tab.invoice_num = part_tab.invoice_num
               AND sales_tab.invoice_num = ship_tab.invoice_num
```

Se il table space viene specificato nel comando **enable_column**, le tabelle laterali vengono create nel table space specificato. Se non viene specificato alcun table space, le tabelle laterali vengono create nel table space predefinito.

Indicizzazione delle tabelle laterali

Dopo aver abilitato una colonna XML e creato le tabelle laterali, è possibile creare un indice per le tabelle laterali. Le tabelle laterali contengono dati XML nelle colonne specificate durante la creazione del file DAD. L'indicizzazione di queste tabelle ottimizza le prestazioni delle interrogazioni per i documenti XML.

Informazioni preliminari

- Creare un file DAD che specifichi le tabelle laterali per la struttura dei documenti XML.
- Abilitare la colonna XML utilizzando il file DAD per creare le tabelle laterali.

Comando DB2 CREATE INDEX

Utilizzare il comando DB2 CREATE INDEX.

Esempio:

Il seguente esempio consente di creare gli indici in quattro tabelle laterali:

```
DB2 CREATE INDEX KEY_IDX
      ON ORDER_SIDE_TAB(ORDER_KEY)

DB2 CREATE INDEX CUSTOMER_IDX
      ON ORDER_SIDE_TAB(CUSTOMER)

DB2 CREATE INDEX PRICE_IDX
      ON PART_SIDE_TAB(PRICE)

DB2 CREATE INDEX DATE_IDX
      ON SHIP_SIDE_TAB(DATE)
```

Disabilitazione delle colonne XML

Disabilitare una colonna se occorre aggiornare un file DAD per la colonna XML o se si desidera cancellare la colonna XML o la tabella che contiene la colonna. Dopo aver disabilitato la colonna, è possibile riabilitarla con il file DAD aggiornato, cancellarla o eseguire altre attività. E' possibile disabilitare una colonna utilizzando il wizard di gestione XML Extender o la shell dei comandi DB2.

Per abilitare una colonna XML, XML Extender:

- Cancella la voce colonna dalla tabella XML_USAGE.
- Cancella le tabelle laterali associate alla colonna.

Importante: Se si cancella una tabella con una colonna XML, senza disabilitare la colonna, XML Extender non può cancellare le tabelle laterali associate alla colonna XML, altrimenti potrebbero verificarsi risultati non previsti.

Informazioni preliminari

Verificare se la colonna che si desidera disabilitare esiste nel database DB2 corrente.

Utilizzo del wizard di gestione

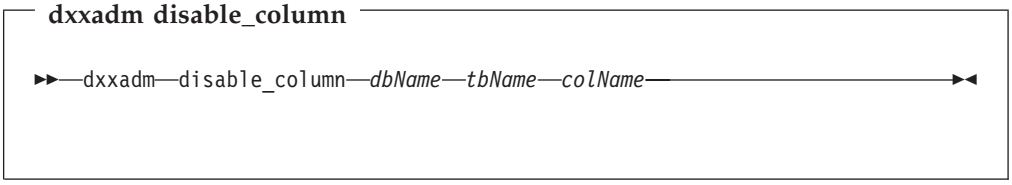
Eseguire i passi riportati di seguito per disabilitare le colonne XML:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Operazioni con le colonne XML** dalla finestra LaunchPad per visualizzare le attività relative alle colonne XML Extender. Viene visualizzata la finestra Selezionare un'attività.
3. Fare clic su **Disabilitare una colonna** e quindi su **Avanti** per disabilitare una colonna di tabella esistente nel database.
4. Selezionare la tabella contenente la colonna XML dal campo **Nome tabella**.
5. Selezionare la colonna da disabilitare nel campo **Nome colonna**.
6. Fare clic su **Fine**.
   - Se la colonna è stata disabilitata con esito positivo, viene visualizzato un messaggio `Colonna disabilitata con esito positivo`.
   - Se la colonna non è stata disabilitata con esito positivo, viene visualizzata una casella di errore. Correggere i valori nel campo di immissione fino a quando la colonna viene disabilitata con esito positivo.

Shell dei comandi DB2

Per disabilitare una colonna XML, immettere il seguente comando:

Sintassi:



Parametri:

dbName
Il nome del database.

tbName
Il nome della tabella che contiene la colonna che si desidera disabilitare.

colName
Il nome della colonna XML che viene disabilitata.

Esempio: Il seguente esempio consente di disabilitare una colonna utilizzando la shell dei comandi DB2. Il file DAD e il documento XML possono essere trovati nell'"Appendice B. Esempi" a pagina 223.

```
dxxadm disable_column SALES_DB sales_tab order
```

In questo esempio, la colonna ORDER viene disabilitata nella tabella SALES_DB.SALES_TAB.

Quando la colonna viene disabilitata, vengono cancellate le tabelle laterali.

Operazioni con le raccolte XML

L'impostazione delle raccolte XML richiede la creazione di uno schema di associazione, e facoltativamente l'abilitazione della raccolta con un nome virtuale, che associa le tabelle DB2 ad un file DAD.

Sebbene l'abilitazione della raccolta XML non sia obbligatoria, essa fornisce prestazioni ottimizzate.

Creazione o editazione del file DAD per lo schema di associazione

E' necessario creare un file DAD quando si utilizzano le raccolte XML. Un file DAD definisce la relazione tra i dati XML e le numerose tabelle relazionali. XML Extender utilizza il file DAD per:

- Eseguire la composizione di un documento XML dai dati relazionali
- Eseguire la scomposizione di un documento XML nei dati relazionali

E' possibile utilizzare i seguenti metodi per associare i dati delle tabelle XML e della tabella DB2: associazione SQL e associazione RDB_node:

Associazione SQL
Utilizza un elemento dell'istruzione SQL per specificare l'interrogazione SQL per le tabelle e le colonne che contengono i dati XML. L'associazione SQL può essere utilizzata solo per la composizione dei documenti XML.

Associazione RDB_node
Utilizza un elemento univoco XML Extender, un nodo del database relazionale o un RDB_node, che specifica le tabelle, le colonne, le condizioni e l'ordine per i dati XML. L'associazione RDB_node supporta associazioni più complesse rispetto a un'istruzione SQL. L'associazione RDB_node può essere utilizzata sia per la composizione, che per la scomposizione dei documenti XML.

Entrambi i metodi di associazione utilizzano il *modello di dati XPath* descritto nella sezione "File DAD" a pagina 47.

Informazioni preliminari

- Stabilire una relazione tra le tabelle DB2 e il documento XML. Questo passo deve includere il metodo di associazione della gerarchia del documento XML e la specifica del modo in cui i dati del documento vengono associati a una tabella DB2.
- Se si desidera convalidare i documenti XML, inserire la DTD per il documento XML che viene composto e scomposto nella tabella di riferimento DTD, db2xml.DTD_REF.

Composizione dei documenti XML con l'associazione SQL

Utilizzare l'associazione SQL per la composizione dei documenti XML con SQL.

Utilizzo del wizard di gestione: Eseguire i passi riportati di seguito per creare un file DAD utilizzando l'associazione SQL delle raccolte XML:

Per creare un file DAD per la composizione utilizzando l'associazione SQL:

Usare l'associazione SQL per la composizione dei documenti XML quando si desidera utilizzare un'istruzione SQL per definire la tabella e le colonne da cui derivano i dati nel documento XML.

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Operazioni con i file DAD** dalla finestra LaunchPad. Viene visualizzata la finestra Specificare una DAD.
3. Scegliere se editare un file DAD esistente oppure creare un nuovo file DAD.

   Per creare un nuovo file DAD:

   a. Lasciare il campo **Nome file** vuoto.
   b. Dal menu **Tipo**, selezionare **Associazione SQL di raccolta XML**.
   c. Fare clic su **Avanti** per aprire la finestra Selezionare convalida.

   Per editare un file DAD esistente:

   a. Immettere il nome del file DAD nel campo **Nome file** oppure fare clic su **...** per selezionare un file DAD esistente.
   b. Verificare se il wizard riconosce il file DAD specificato.
      - Se il wizard riconosce il file specificato, **Avanti** è selezionabile l'associazione SQL di raccolta XML viene visualizzata nel campo **Tipo**.
      - Se il wizard non riconosce il file DAD specificato, non è possibile selezionare **Avanti**. Immettere nuovamente il nome file DAD oppure fare clic su **...** per sfogliare e selezionare nuovamente un file DAD esistente. Correggere i valori del campo di immissione fino a quando il pulsante **Avanti** non risulti selezionabile.
   c. Fare clic su **Avanti** per aprire la finestra Selezionare convalida.
4. Nella finestra Selezionare convalida, scegliere se convalidare i documenti XML con una DTD.
   - Per convalidare:
     a. Fare clic su **Convalidare i documenti XML con la DTD**.
     b. Selezionare la DTD da utilizzare per la convalida dall'elenco **ID DTD**.

     Se non è stata importata alcuna DTD nel magazzino DTD del database, non è possibile convalidare i documenti XML.
   - Fare clic su **NON convalidare i documenti XML con la DTD** per continuare senza convalidare i documenti XML.

5. Fare clic su **Avanti** per aprire la finestra Specificare testo.
6. Immettere il nome prologo nel campo **Prologo**, per specificare il prologo del documento XML da creare.

   ```
   <?xml version="1.0"?>
   ```

   Se si sta editando una DAD esistente, il prologo viene visualizzato automaticamente nel campo **Prologo**.
7. Immettere il tipo di documento del documento XML nel campo **Tipo di documento** presente nella finestra Specificare testo, indicando la DTD del documento XML. Ad esempio:

   ```
   !DOCTYPE DAD SYSTEM "c:\dxx\samples\dtd\getstart.dtd"
   ```

   Se si sta editando una DAD esistente, il tipo di documento viene visualizzato automaticamente nel campo **Tipo di documento**.
8. Fare clic su **Avanti** per aprire la finestra Specificare istruzione SQL.
9. Immettere un'istruzione SQL SELECT valida nel campo **Istruzione SQL**. Ad esempio:

   ```
   SELECT o.order_key, customer_name, customer_email, p.part_key, color, quantity,
   price, tax, ship_id, date, mode from order_tab o, part_tab p,
   table (select substr(char(timestamp(generate_unique())),16)
     as ship_id, date, mode, part_key from ship_tab) s
         WHERE o.order_key = 1 and
                  p.price > 20000 and
               p.order_key = o.order_key and
               s.part_key = p.part_key
         ORDER BY order_key, part_key, ship_id
   ```

   Se si sta editando una DAD esistente, l'istruzione SQL viene visualizzata automaticamente nel campo **Istruzione SQL**.
10. Fare clic su **Verificare SQL** per verificare la validità dell'istruzione SQL.
    - Se l'istruzione SQL è valida, i risultati di esempio vengono visualizzati nel campo **Risultati di esempio**.
    - Se l'istruzione SQL non è valida, viene visualizzato un messaggio di errore nel campo **Risultati di esempio**. Il messaggio di errore richiede di correggere l'istruzione SQL SELECT e riprovare.
11. Fare clic su **Avanti** per aprire la finestra Associazione SQL.
12. Selezionare un nodo di elemento o attributo da cui creare un'associazione facendo clic su di esso nel campo visualizzato a sinistra della finestra Associazione SQL.

    Associare gli elementi ed attributi del documento XML ai nodi di elemento e attributo che corrispondono ai dati DB2. Questi nodi forniscono un percorso dai dati XML ai dati DB2.
    - **Per aggiungere il nodo root:**
      a. Selezionare l'icona **Root**.
      b. Fare clic su **Nuovo elemento** per definire un nuovo nodo.
      c. Nella casella **Dettagli** specificare **Tipo nodo** come **Elemento**.
      d. Immettere il nome del nodo di livello superiore nel campo **Nome nodo**.
      e. Fare clic su **Aggiungere** per creare il nuovo nodo.

         E' stato creato un elemento o nodo root, che è principale rispetto a tutti gli altri nodi di attributo ed elemento nella struttura dell'associazione. Adesso è possibile aggiungere attributi ed elementi secondari a questo nodo.

- **Per aggiungere un elemento secondario o un nodo di attributo:**
  a. Per aggiungere un attributo o elemento secondario, fare clic su un nodo principale nel campo visualizzato a sinistra.

     Se non è stato selezionato un nodo principale, **Nuovo elemento** non è selezionabile.
  b. Fare clic su **Nuovo elemento**.
  c. Selezionare il tipo di nodo dal menu **Tipo nodo** nella casella **Dettagli**.

     Nel menu **Tipo nodo** vengono visualizzati solo i tipi di nodo validi in questo punto del processo di associazione:

     Elemento
     Rappresenta un elemento XML definito nella DTD associata al documento XML. Viene utilizzato per associare l'elemento XML ad una colonna in una tabella DB2. Un nodo di elemento può avere nodi di attributo, nodi di elemento secondari o nodi di testo. Un nodo di livello inferiore ha un nodo di testo e un nome colonna associati ad esso nella vista della struttura.

     Attributo
     Rappresenta un attributo XML definito nella DTD associata al documento XML. Viene utilizzato per associare l'attributo XML ad una colonna in una tabella DB2. Un nodo di attributo può avere un nodo di testo ed ha un nome colonna associato ad esso nella vista della struttura.

     Testo Specifica il contenuto del testo per un nodo di elemento o attributo che deve essere associato a una tabella relazionale. Un nodo di testo ha un nome colonna associato ad esso nella vista della struttura.

     Tabella
     Specifica il nome tabella di un valore dell'elemento o dell'attributo che deve essere associato a una tabella relazionale.

     Column
     Specifica il nome colonna per un valore dell'elemento o dell'attributo che deve essere associato a una tabella relazionale.

     Condizione
     Specifica una condizione per la colonna.
  d. Immettere il nome nodo nel campo **Nome nodo** presente nella casella **Dettagli**. Ad esempio:

     ```
     Ordine
     ```
  e. Se è stato specificato **Attributo**, **Elemento** o **Testo** per un elemento di livello inferiore come Tipo nodo, selezionare una colonna nel campo **Colonna** della casella **Dettagli**. Ad esempio:

     ```
     Customer_Name
     ```

     Limitazione: Non è possibile creare nuove colonne utilizzando il wizard di gestione. Se si specifica **Colonna** come tipo di nodo, è possibile selezionare solo una colonna già esistente nel database DB2.
  f. Fare clic su **Aggiungere** per aggiungere il nuovo nodo.

     E' possibile modificare un nodo successivamente facendo clic su di esso nel campo visualizzato a sinistra e apportandovi le modifiche necessarie nella casella **Dettagli**. Fare clic su **Modificare** per aggiornare l'elemento.

Inoltre, è possibile elementi o attributi secondari al nodo evidenziandolo, ripetendo così il processo di aggiunta.

g. Continuare ad editare l'associazione SQL oppure fare clic su **Avanti** per aprire la finestra Specificare una DAD.

- **Per eliminare un nodo:**
  a. Fare clic su un nodo nel campo visualizzato a sinistra.
  b. Fare clic su **Eliminare**.
  c. Continuare ad editare l'associazione SQL oppure fare clic su **Avanti** per aprire la finestra Specificare una DAD.

  Notare che se si rimuove un nodo di livello inferiore, un altro elemento diventerà nodo di livello inferiore e potrebbe essere necessario definire un nome colonna per questo nodo.

13. Immettere un nome file di output per il file DAD modificato nel campo **Nome file** della finestra Specificare una DAD.
14. Fare clic su **Fine** per ritornare alla finestra LaunchPad.

Shell dei comandi DB2: Utilizzare l'associazione SQL per la composizione di un documento XML con SQL.

Il file DAD è un file XML che può essere creato utilizzando qualsiasi editor di testo. I seguenti passi illustrano alcuni esempi riportati nella sezione "File DAD" a pagina 224. Per ulteriori dettagli, fare riferimento a questi esempi.

1. Aprire un editor di testo.
2. Creare l'intestazione DAD:

```
<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM "path\dad.dtd"> --> il percorso e il
nome file della DTD
per
DAD
```

3. Inserire le tag <DAD></DAD>.
4. Dopo la tag <DAD>, specificare l'ID DTD che associa il file DAD alla DTD del documento XML.

```
<dtdid>path\dtd_name.dtd> --> il percorso e il nome file della DTD per l'applicazione
```

5. Specificare se si desidera utilizzare una DTD per accertarsi che il documento XML è un documento valido. Ad esempio:

```
<validation>NO</validation>   --> specificare Sì o No
```

6. Utilizzare l'elemento <Xcollection> per definire il metodo di memorizzazione e accesso come raccolta XML. I metodi di memorizzazione e accesso consentono di definire se il contenuto del documento XML deriva dai dati memorizzati nelle tabelle DB2.

```
<Xcollection>
</Xcollection>
```

7. Specificare una o più istruzioni SQL per interrogare o inserire i dati nelle tabelle DB2. Per ulteriori informazioni, consultare la sezione "Requisiti dello schema di associazione" a pagina 52. Ad esempio, specificare una singola interrogazione SQL nel seguente modo:

```
 <SQL_stmt>
SELECT o.order_key, customer_name, customer_email, p.part_key, color, quantity,
price, tax, ship_id, date, mode from order_tab o, part_tab p,
table (select substr(char(timestamp(generate_unique())),16)
  as ship_id, date, mode, part_key from ship_tab) s
      WHERE o.order_key = 1 and
               p.price > 20000 and
```

```
            p.order_key = o.order_key and
            s.part_key = p.part_key
      ORDER BY order_key, part_key, ship_id
 </SQL_stmt>
```

8. Aggiungere le seguenti informazioni relative al prologo:

   ```
   <prolog>?xml version="1.0"?</prolog>
   ```

   E' richiesto il testo esatto.

9. Aggiungere le tag <doctype></doctype>. Ad esempio:

   ```
   <doctype>! DOCTYPE Order SYSTEM "c:\dxx\samples\dtd\getstart.dtd"</doctype>
   ```

10. Definire il nodo root utilizzando le tag <root_node></root_node>. Nella tag root_node, specificare gli elementi e gli attributi che costituiscono il documento XML.

11. Associare gli elementi ed attributi del documento XML ai nodi di elemento e attributo che corrispondono ai dati DB2. Questi nodi forniscono un percorso dai dati XML ai dati DB2.

    a. Definire una tag <element_node> per ciascun elemento del documento XML che si associa ad una colonna della tabella DB2.

       ```
       <element_node name="name"></element_node>
       ```

       element_node può contenere i seguenti nodi:

       - attribute_node
       - element_node secondario
       - text_node

    b. Definire una tag <attribute_node> per ciascun attributo del documento XML che si associa ad una colonna della tabella DB2. Fare riferimento alle DTD di esempio per l'associazione SQL, riportate all'inizio di questa sezione, e alla DTD per il file DAD descritta nella sezione "Appendice A. DTD per il file DAD" a pagina 217, che fornisce la sintassi completa per il file DAD.

       Ad esempio, è richiesta una chiave attributo per l'elemento <Order>. Il valore della chiave viene memorizzato in una colonna PART_KEY.

       File DAD: Nel file DAD, creare un nodo di attributo per la chiave e indicare la tabella in cui verrà memorizzato il valore 1.

       ```
       <attribute_node name="key">
           <column name="part_key"/>
       </attribute_node>
       ```

       Documento XML composto: Il valore della chiave viene richiamato dalla colonna PART_KEY.

       ```
           <Order key="1">
       ```

12. Definire una tag <text_node> per ogni elemento o attributo il cui contenuto deriva da una tabella DB2. L'elemento <column> del nodo testo specifica la colonna che fornisce il contenuto.

    Ad esempio, è possibile specificare un elemento XML <Tax> con un valore che verrà richiamato da una colonna denominata TAX:

    Elemento DAD:

    ```
    <element_node name="Tax">
       <text_node>
             <column name="tax"/>
       </text_node>
    </element_node>
    ```

Il nome colonna deve essere incluso nell'istruzione SQL all'inizio del file DAD.

Documento XML composto:

```
<Tax>0.02</Tax>
```

Il valore `0.02` verrà richiamato dalla colonna TAX.

13. Verificare se viene posizionata una tag </root_node> dopo l'ultima tag </element_node>.
14. Verificare se viene posizionata una tag </Xcollection> dopo la tag </root_node>.
15. Verificare se viene posizionata una tag </DAD> dopo la tag </Xcollection>.

Composizione dei documenti XML con l'associazione RDB_node

Utilizzare l'associazione RDB_node per la composizione dei documenti XML.

Questo metodo utilizza la tag <RDB_node> per specificare le tabelle DB2, le colonne e le condizioni per un nodo di attributo o elemento. <RDB_node> utilizza i seguenti elementi:

- <table>: definisce la tabella corrispondente all'elemento
- <column>: definisce la colonna contenente l'elemento corrispondente
- <condition>: specifica facoltativamente una condizione per la colonna

Gli elementi secondari utilizzati nella tag <RDB_node> dipendono dal contesto del nodo e rispettano le seguenti regole:

Se il tipo di nodo è:	**Viene utilizzato l'elemento secondario RDB:**		
	Tabella	**Colonna**	**Condizione[1]**
Elemento root	Y	S	Y
Attributo	Y	Y	facoltativa
Testo	Y	Y	facoltativa

(1) Condizione obbligatoria con più tabelle

Utilizzo del wizard di gestione: Per creare una DAD per la composizione utilizzando l'associazione RDB_node:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Operazioni con i file DAD** dalla finestra LaunchPad. Viene visualizzata la finestra Specificare una DAD.
3. Scegliere se editare un file DAD esistente oppure creare un nuovo file DAD.

   Per editare un file DAD esistente:

   a. Immettere il nome del file DAD nel campo **Nome file** oppure fare clic su **...** per selezionare un file DAD esistente.
   b. Verificare se il wizard riconosce il file DAD specificato.
      - Se il wizard riconosce il file DAD specificato, **Avanti** è selezionabile e l'associazione RDB_node di raccolta XML è visualizzata nel campo **Tipo**.
      - Se il wizard non riconosce il file DAD specificato, non è possibile selezionare **Avanti**. Immettere nuovamente il nome del file DAD nel

campo **Nome file** oppure fare clic su **...** per selezionare un file DAD esistente. Continuare questi passi fino a quando **Avanti** diventa selezionabile.

c. Fare clic su **Avanti** per aprire la finestra Selezionare convalida.

Per creare una nuova DAD:

a. Lasciare il campo **Nome file** vuoto.
b. Selezionare Associazione RDB_node di raccolta XML dal menu **Tipo**.
c. Fare clic su **Avanti** per aprire la finestra Selezionare convalida.

4. Nella finestra Selezionare convalida, scegliere se convalidare i documenti XML con una DTD.
   - Per convalidare:
     a. Fare clic su **Convalidare i documenti XML con la DTD**.
     b. Selezionare la DTD da utilizzare per la convalida dall'elenco **ID DTD**.

     Se non è stata importata alcuna DTD nel magazzino DTD del database, non è possibile convalidare i documenti XML.
   - Fare clic su **NON convalidare i documenti XML con la DTD** per continuare senza convalidare i documenti XML.
5. Fare clic su **Avanti** per aprire la finestra Specificare testo.
6. Immettere il nome prologo nel campo **Prologo** della finestra Specificare testo.

   ```
   <?xml version="1.0"?>
   ```

   Se si sta editando una DAD esistente, il prologo viene visualizzato automaticamente nel campo **Prologo**.
7. Immettere il tipo di documento XML nel campo **Tipo di documento** presente nella finestra Specificare testo.

   Se si sta editando una DAD esistente, il tipo di documento viene visualizzato automaticamente nel campo **Tipo di documento**.
8. Fare clic su **Avanti** per aprire la finestra Associazione RDB.
9. Selezionare un nodo di attributo o elemento da cui creare un'associazione facendo clic su di esso nel campo visualizzato a sinistra della finestra Associazione RDB.

   Associare gli elementi ed attributi del documento XML ai nodi di elemento e attributo che corrispondono ai dati DB2. Questi nodi forniscono un percorso dai dati XML ai dati DB2.
10. **Per aggiungere il nodo root:**
    a. Selezionare l'icona **Root**.
    b. Fare clic su **Nuovo elemento** per definire un nuovo nodo.
    c. Nella casella **Dettagli** specificare **Tipo nodo** come **Elemento**.
    d. Immettere il nome del nodo di livello superiore nel campo **Nome nodo**.
    e. Fare clic su **Aggiungere** per creare il nuovo nodo.

       E' stato creato un elemento o nodo root, che è principale rispetto a tutti gli altri nodi di attributo ed elemento nella struttura dell'associazione. Il nodo root ha elementi secondari di tabella e una condizione di collegamento.
    f. Aggiungere nodi di tabella per ciascuna tabella appartenente alla raccolta.
       1) Evidenziare il nome nodo root e selezionare **Nuovo elemento**.
       2) Nella casella **Dettagli** specificare **Tipo nodo** come **Tabella**.

3) Selezionare il nome della tabella dall'elenco **Nome tabella**. La tabella deve essere già presente.
4) Fare clic su **Aggiungere** per aggiungere il nodo di tabella.
5) Ripetere questi passi per ciascuna tabella.

g. Aggiungere una condizione di collegamento per i nodi di tabella.
1) Evidenziare il nome nodo root e selezionare **Nuovo elemento**.
2) Nella casella **Dettagli** specificare **Tipo nodo** come **Condizione**.
3) Nel campo **Condizione**, immettere la condizione di collegamento utilizzando la seguente sintassi:

```
table_name.table_column = table_name.table_column AND
table_name.table_column = table_name.table_column ...
```

4) Fare clic su **Aggiungere** per aggiungere la condizione.

11. **Per aggiungere un nodo di attributo o elemento:**
a. Per aggiungere un attributo o elemento secondario, fare clic su un nodo principale nel campo visualizzato a sinistra.
b. Fare clic su **Nuovo elemento**. Se non è stato selezionato un nodo principale, **Nuovo elemento** non è selezionabile.
c. Selezionare un tipo di nodo dal menu **Tipo nodo** nella casella **Dettagli**.

Nel menu **Tipo nodo** vengono visualizzati solo i tipi di nodo validi in questo punto del processo di associazione. **Elemento** o **Attributo**.

d. Specificare un nome nodo nel campo **Nome nodo**.
e. Fare clic su **Aggiungere** per aggiungere il nuovo nodo.
f. **Per associare il contenuto di un nodo di attributo o elemento ad una tabella relazionale:**
1) Specificare un nodo di testo.
a) Fare clic sul nodo secondario.
b) Fare clic su **Nuovo elemento**.
c) Nel campo **Tipo nodo** selezionare **Testo**.
d) Selezionare **Aggiungere** per aggiungere il nodo.
2) Aggiungere un nodo di tabella.
a) Selezionare il nodo creato e fare clic su **Nuovo elemento**.
b) Nel campo **Tipo nodo** selezionare **Tabella** e specificare un nome tabella per l'elemento.
c) Fare clic su **Aggiungere** per aggiungere il nodo.
3) Aggiungere un nodo di colonna.
a) Selezionare nuovamente il nodo di testo e fare clic su **Nuovo elemento**.
b) Nel campo **Tipo nodo** selezionare **Colonna** e specificare un nome colonna per l'elemento.
c) Fare clic su **Aggiungere** per aggiungere il nodo.

Limitazione: Non è possibile creare nuove colonne utilizzando il wizard di gestione. Se si specifica Colonna come tipo di nodo, è possibile selezionare solo una colonna già esistente nel database DB2.

4) Facoltativamente, aggiungere una condizione per la colonna.
a) Selezionare nuovamente il nodo di testo e fare clic su **Nuovo elemento**.

b) Nel campo **Tipo nodo** selezionare **Condizione** e la condizione con la seguente sintassi:

```
operator LIKE|<|>|= value
```

c) Fare clic su **Aggiungere** per aggiungere il nodo.

g. Continuare ad editare l'associazione RDB oppure fare clic su **Avanti** per aprire la finestra Specificare una DAD.

12. **Per eliminare un nodo:**
    a. Fare clic su un nodo nel campo visualizzato a sinistra.
    b. Fare clic su **Eliminare**.
    c. Continuare ad editare l'associazione RDB_node oppure fare clic su **Avanti** per aprire la finestra Specificare una DAD.
13. Immettere un nome file di output per la DAD modificata nel campo **Nome file** della finestra Specificare una DAD.
14. Fare clic su **Fine** per eliminare il nodo e ritornare alla finestra LaunchPad.

Shell dei comandi DB2: Il file DAD è un file XML che può essere creato utilizzando qualsiasi editor di testo. I seguenti passi illustrano alcuni esempi riportati nella sezione "File DAD" a pagina 224. Per ulteriori dettagli, fare riferimento a questi esempi.

1. Aprire un editor di testo.
2. Creare l'intestazione DAD:

```
<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM "path\dad.dtd"> --> il percorso e il
nome file della DTD
per
DAD
```

3. Inserire le tag <DAD></DAD>.
4. Dopo la tag <DAD>, specificare l'ID DTD che associa il file DAD alla DTD del documento XML.

```
<dtdid>path\dtd_name.dtd> --> il percorso e il
nome file della DTD per l'applicazione
```

5. Specificare se si desidera utilizzare una DTD per accertarsi che il documento XML è un documento valido. Ad esempio:

```
<validation>NO</validation>   --> specificare Sì o No
```

6. Utilizzare l'elemento <Xcollection> per definire il metodo di memorizzazione e accesso come raccolta XML. I metodi di memorizzazione e accesso consentono di definire se i dati XML sono memorizzati in una raccolta delle tabelle DB2.

```
<Xcollection>
</Xcollection>
```

7. Aggiungere le seguenti informazioni relative al prologo:

```
<prolog>?xml version="1.0"?</prolog>
```

   E' richiesto il testo esatto.
8. Aggiungere le tag <doctype></doctype>. Ad esempio:

```
<doctype>! DOCTYPE Order SYSTEM "c:\dxx\samples\dtd\getstart.dtd"</doctype>
```

9. Definire il nodo root utilizzando la tag <root_node>. Nella tag root_node, specificare gli elementi e gli attributi che costituiscono il documento XML.
10. Associare gli elementi ed attributi del documento XML ai nodi di elemento e attributo che corrispondono ai dati DB2. Questi nodi forniscono un percorso dai dati XML ai dati DB2.

a. Definire un element_node per il nodo root. Questo element_node contiene:
   - RDB_node che definisce table_node con una condizione di collegamento per specificare la raccolta
   - Elementi secondari
   - Attributi

   Per specificare i nodi di tabella e le condizioni:

   1) Creare un elemento RDB_node. Ad esempio:

      ```
      <RDB_node>
            </RDB_node>
      ```

   2) Definire un <table_node> per ogni tabella contenente i dati da includere nel documento XML. Ad esempio, se vi sono tre tabelle, ORDER_TAB, PART_TAB e SHIP_TAB, contenenti dati colonna da inserire nel documento, creare un nodo per ciascuna tabella. Ad esempio:

      ```
      <RDB_node>
      <table name="ORDER_TAB">
      <table name="PART_TAB">
      <table name="SHIP_TAB"></RDB_node>
      ```

   3) Facoltativamente, specificare una colonna chiave per ciascuna tabella quando si pianifica l'abilitazione di questa raccolta. L'attributo chiave non è obbligatorio per la composizione; tuttavia, quando si abilita una raccolta, il file DAD utilizzato deve supportare la composizione e la scomposizione. Ad esempio:

      ```
      <RDB_node>
      <table name="ORDER_TAB" key="order_key">
      <table name="PART_TAB" key="part_key">
      <table name="SHIP_TAB" key="date mode">
            </RDB_node>
      ```

   4) Definire una condizione di collegamento per le tabelle nella raccolta. La sintassi è:

      ```
      expression = expression AND
      expression = expression
      ```

      Ad esempio:

      ```
      <RDB_node>
      <table name="ORDER_TAB">
      <table name="PART_TAB">
      <table name="SHIP_TAB">
      <condition>
      order_tab.order_key = part_tab.order_key AND
                  part_tab.part_key = ship_tab.part_key
      </condition>
            </RDB_node>
      ```

b. Definire una tag <element_node> per ciascun elemento del documento XML che si associa ad una colonna della tabella DB2. Ad esempio:

   ```
   <element_node name="name">
   </element_node>
   ```

   Un nodo di elemento può disporre di uno dei seguenti tipi di elemento:

   - <text_node>: per specificare che il contenuto dell'elemento è associato a una tabella DB2; l'elemento non presenta elementi secondari
   - <attribute_node>: per specificare un attributo. I nodi dell'attributo vengono definiti nel passo successivo.

L'elemento text_node contiene un <RDB_node> il cui contenuto deve essere associato a una tabella DB2 e a un nome colonna.

Gli RDB_node vengono utilizzati per gli elementi di livello inferiore il cui contenuto deve essere associato a una tabella DB2. RDB_node presenta i seguenti elementi secondari:

- <table>: definisce la tabella corrispondente all'elemento
- <column>: definisce la colonna contenente l'elemento corrispondente e specifica il tipo di colonna come attributo
- <condition>: specifica facoltativamente una condizione per la colonna

Ad esempio, è possibile specificare un elemento XML <Tax> che viene associato a una colonna denominata TAX:

Documento XML:

```
<Tax>0.02</Tax>
```

In tal caso, il valore `0.02` deve essere un valore della colonna TAX.

```
<element_node name="Tax">
   <text_node>
<RDB_node>
         <table name="part_tab"/>
         <column name="tax"/>
      </RDB_node>
   </text_node>
</element_node>
```

In questo esempio, <RDB_node> specifica che il valore dell'elemento <Tax> è un valore di testo e che i dati vengono memorizzati nella colonna TAX della tabella PART_TAB.Fare riferimento ai file DAD di esempio illustrati nella sezione "File DAD" a pagina 224 per l'associazione RDB_node e alla DTD per il file DAD descritta nella sezione "Appendice A. DTD per il file DAD" a pagina 217, che fornisce la sintassi corretta per il file DAD.

c. Facoltativamente, aggiungere un attributo tipo a ciascun elemento <column> quando si pianifica l'abilitazione di questa raccolta. L'attributo tipo non è obbligatorio per la composizione; tuttavia, quando si abilita una raccolta, il file DAD utilizzato deve supportare la composizione e la scomposizione. Ad esempio:

```
<column name="tax" type="real"/>
```

d. Definire una tag <attribute_node> per ciascun attributo del documento XML che si associa ad una colonna della tabella DB2. Ad esempio:

```
<attribute_node name="key">
</attribute_node>
```

L'elemento attribute_node utilizza un <RDB_node> per associare il valore dell'attributo a una colonna o tabella DB2. <RDB_node> presenta i seguenti elementi secondari:

- <table>: definisce la tabella corrispondente all'elemento
- <column>: definisce la colonna contenente l'elemento corrispondente
- <condition>: specifica facoltativamente una condizione per la colonna

Ad esempio, è richiesto un attributo `key` per l'elemento <Order>. Il valore della chiave deve essere memorizzato in una colonna PART_KEY. Nel file

DAD, creare un elemento <attribute_node> per la chiave e indicare la tabella in cui verrà memorizzato il valore.

File DAD

```
<attribute_node name="key">
<RDB_node>
    <table name="part_tab">
    <column name="part_key"/>
<RDB_node>
</attribute_node>
```

Documento XML composto:

```
    <Order key="1">
```

11. Verificare se viene posizionata una tag </root_node> dopo l'ultima tag </element_node>.
12. Verificare se viene posizionata una tag </Xcollection> dopo la tag </root_node>.
13. Verificare se viene posizionata una tag </DAD> dopo la tag </Xcollection>.

Specifica di un foglio di stile per il documento XML

Durante la composizione di documenti, XML Extender supporta anche le istruzioni di elaborazione per i fogli di stile, mediante l'elemento <stylesheet>. Le istruzioni di elaborazione devono trovarsi all'interno dell'elemento root <Xcollection>, localizzate da <doctype> e <prolog>, come definito per la strutura del documento XML. Ad esempio:

```
<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM "c:\dtd\dad.dtd">
<DAD>
<SQL_stmt>
       ...
 </SQL_stmt>
<Xcollection>
...
<prolog>...</prolog>
<doctype>...</doctype>
<stylesheet>?xml-stylesheet type="text/css" href="order.css"?</stylesheet>
<root_node>...</root_node>
...

</Xcollection>
...
</DAD>
```

Scomposizione dei documenti XML con l'associazione RDB_node

Utilizzare l'associazione RDB_node per eseguire la scomposizione dei documenti XML. Questo metodo utilizza la tag <RDB_node> per specificare le tabelle DB2, le colonne e le condizioni per un nodo di attributo o elemento. <RDB_node> utilizza i seguenti elementi:

- <table>: definisce la tabella corrispondente all'elemento
- <column>: definisce la colonna contenente l'elemento corrispondente
- <condition>: specifica facoltativamente una condizione per la colonna

Gli elementi secondari utilizzati nella tag <RDB_node> dipendono dal contesto del nodo e rispettano le seguenti regole:

Se il tipo di nodo è:	Viene utilizzato l'elemento secondario RDB:		
	Tabella	**Colonna**	**Condizione[1]**
Elemento root	Y	S	Y
Attributo	Y	Y	facoltativa
Testo	Y	Y	facoltativa
(1) Condizione obbligatoria con più tabelle			

Utilizzo del wizard di gestione: Per creare una DAD per la scomposizione:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Operazioni con i file DAD** dalla finestra LaunchPad. Viene visualizzata la finestra Specificare una DAD.
3. Scegliere se editare un file DAD esistente oppure creare un nuovo file DAD.

   Per editare un file DAD esistente:

   a. Immettere il nome del file DAD nel campo **Nome file** oppure fare clic su **...** per selezionare un file DAD esistente.
   b. Verificare se il wizard riconosce il file DAD specificato.
      - Se il wizard riconosce il file DAD specificato, **Avanti** è selezionabile e l'associazione RDB_node di raccolta XML è visualizzata nel campo **Tipo**.
      - Se il wizard non riconosce il file DAD specificato, non è possibile selezionare **Avanti**. Immettere nuovamente il nome del file DAD nel campo **Nome file** oppure fare clic su **...** per selezionare un file DAD esistente. Continuare questi passi fino a quando **Avanti** diventa selezionabile.
   c. Fare clic su **Avanti** per aprire la finestra Selezionare convalida.

   Per creare una nuova DAD:

   a. Lasciare il campo **Nome file** vuoto.
   b. Selezionare Associazione RDB_node di raccolta XML dal menu **Tipo**.
   c. Fare clic su **Avanti** per aprire la finestra Selezionare convalida.
4. Nella finestra Selezionare convalida, scegliere se convalidare i documenti XML con una DTD.
   - Per convalidare:
     a. Fare clic su **Convalidare i documenti XML con la DTD**.
     b. Selezionare la DTD da utilizzare per la convalida dall'elenco **ID DTD**.

     Se non è stata importata alcuna DTD nel magazzino DTD del database, non è possibile convalidare i documenti XML.
   - Fare clic su **NON convalidare i documenti XML con la DTD** per continuare senza convalidare i documenti XML.
5. Fare clic su **Avanti** per aprire la finestra Specificare testo.
6. Se si sta eliminando solo un documento XML, ignorare il campo **Prologo**. Se si sta utilizzando il file DAD sia per la creazione che per l'eliminazione, immettere il nome prologo nel campo **Prologo** della finestra Specificare testo. Il prologo non è richiesto se si stanno eliminando documenti XML all'interno dei dati DB2.

   ```
   <?xml version="1.0"?>
   ```

Se si sta editando una DAD esistente, il prologo viene visualizzato automaticamente nel campo **Prologo**.

7. Se si sta eliminando solo un documento XML, ignorare il campo **Tipo di documento**. se si sta utilizzando il file DAD sia per la creazione che per l'eliminazione, immettere il tipo di documento del documento XML nel campo **Tipo di documento**.

   Se si sta editando una DAD esistente, il tipo di documento viene visualizzato automaticamente nel campo **Tipo di documento**.
8. Fare clic su **Avanti** per aprire la finestra Associazione RDB.
9. Selezionare un nodo di attributo o elemento da cui creare un'associazione facendo clic su di esso nel campo visualizzato a sinistra della finestra Associazione RDB.

   Associare gli elementi ed attributi del documento XML ai nodi di elemento e attributo che corrispondono ai dati DB2. Questi nodi forniscono un percorso dai dati XML ai dati DB2.
10. **Per aggiungere il nodo root:**
    a. Selezionare l'icona **Root**.
    b. Fare clic su **Nuovo elemento** per definire un nuovo nodo.
    c. Nella casella **Dettagli** specificare **Tipo nodo** come **Elemento**.
    d. Immettere il nome del nodo di livello superiore nel campo **Nome nodo**.
    e. Fare clic su **Aggiungere** per creare il nuovo nodo.

       E' stato creato un elemento o nodo root, che è principale rispetto a tutti gli altri nodi di attributo ed elemento nella struttura dell'associazione. Il nodo root ha elementi secondari di tabella e una condizione di collegamento.
    f. Aggiungere nodi di tabella per ciascuna tabella appartenente alla raccolta.
       1) Evidenziare il nome nodo root e selezionare **Nuovo elemento**.
       2) Nella casella **Dettagli** specificare **Tipo nodo** come **Tabella**.
       3) Selezionare il nome della tabella dall'elenco **Nome tabella**. La tabella deve essere già presente.
       4) Specificare una colonna chiave per la tabella nel campo **Chiave tabella**.
       5) Fare clic su **Aggiungere** per aggiungere il nodo di tabella.
       6) Ripetere questi passi per ciascuna tabella.
    g. Aggiungere una condizione di collegamento per i nodi di tabella.
       1) Evidenziare il nome nodo root e selezionare **Nuovo elemento**.
       2) Nella casella **Dettagli** specificare **Tipo nodo** come **Condizione**.
       3) Nel campo **Condizione** immettere la condizione di collegamento utilizzando la sintassi riportata di seguito:.

          ```
          table_name.table_column = table_name.table_column AND
          table_name.table_column = table_name.table_column ...
          ```
       4) Fare clic su **Aggiungere** per aggiungere la condizione.

    Adesso è possibile aggiungere attributi ed elementi secondari a questo nodo.
11. **Per aggiungere un nodo di attributo o elemento:**
    a. Per aggiungere un attributo o elemento secondario, fare clic su un nodo principale nel campo visualizzato a sinistra.

       Se non è stato selezionato un nodo principale, **Nuovo** non è selezionabile.
    b. Fare clic su **Nuovo elemento**.
    c. Selezionare un tipo di nodo dal menu **Tipo nodo** nella casella **Dettagli**.

Nel menu **Tipo nodo** vengono visualizzati solo i tipi di nodo validi in questo punto del processo di associazione. **Elemento** o **Attributo**.

d. Specificare un nome nodo nel campo **Nome nodo**.

e. Fare clic su **Aggiungere** per aggiungere il nuovo nodo.

f. **Per associare il contenuto di un nodo di attributo o elemento ad una tabella relazionale:**

1) Specificare un nodo di testo.
   a) Fare clic sul nodo secondario.
   b) Fare clic su **Nuovo elemento**.
   c) Nel campo **Tipo nodo** selezionare **Testo**.
   d) Selezionare **Aggiungere** per aggiungere il nodo.

2) Aggiungere un nodo di tabella.
   a) Selezionare il nodo creato e fare clic su **Nuovo elemento**.
   b) Nel campo **Tipo nodo** selezionare **Tabella** e specificare un nome tabella per l'elemento.
   c) Fare clic su **Aggiungere** per aggiungere il nodo.

3) Aggiungere un nodo di colonna.
   a) Selezionare nuovamente il nodo di testo e fare clic su **Nuovo elemento**.
   b) Nel campo **Tipo nodo** selezionare **Colonna** e specificare un nome colonna per l'elemento.
   c) Specificare un tipo di dati base per la colonna nel campo **Tipo**, per specificare di quale tipo deve essere la colonna in cui vengono memorizzati i dati non provvisti di tag.
   d) Fare clic su **Aggiungere** per aggiungere il nodo.

   Limitazione: Non è possibile creare nuove colonne utilizzando il wizard di gestione. Se si specifica Colonna come tipo di nodo, è possibile selezionare solo una colonna già esistente nel database DB2.

4) Facoltativamente, aggiungere una condizione per la colonna.
   a) Selezionare nuovamente il nodo di testo e fare clic su **Nuovo elemento**.
   b) Nel campo **Tipo nodo** selezionare **Condizione** e la condizione con la seguente sintassi:
   `operator LIKE|<|>|= value`
   c) Fare clic su **Aggiungere** per aggiungere il nodo.

E' possibile modificare questi nodi selezionando il nodo, modificando i campi della casella **Dettagli** e facendo clic su **Modificare**.

g. Continuare ad editare l'associazione RDB oppure fare clic su **Avanti** per aprire la finestra Specificare una DAD.

12. **Per eliminare un nodo:**
    a. Fare clic su un nodo nel campo visualizzato a sinistra.
    b. Fare clic su **Eliminare**.
    c. Continuare ad editare l'associazione RDB_node oppure fare clic su **Avanti** per aprire la finestra Specificare una DAD.
13. Immettere un nome file di output per la DAD modificata nel campo **Nome file** della finestra Specificare una DAD.
14. Fare clic su **Fine** per eliminare il nodo e ritornare alla finestra LaunchPad.

Shell dei comandi DB2: Il file DAD è un file XML che può essere creato utilizzando qualsiasi editor di testo. I seguenti passi illustrano alcuni esempi riportati nella sezione "File DAD" a pagina 224. Per ulteriori dettagli, fare riferimento a questi esempi.

1. Aprire un editor di testo.
2. Creare l'intestazione DAD:

   ```
   <?xml version="1.0"?>
   <!DOCTYPE DAD SYSTEM "path\dad.dtd"> --> il percorso e il
   nome file della DTD
   per
   DAD
   ```

3. Inserire le tag <DAD></DAD>.
4. Dopo la tag <DAD>, specificare l'ID DTD che associa il file DAD alla DTD del documento XML.

   ```
   <dtdid>path\dtd_name.dtd> --> il percorso e il
   nome file della DTD                                    per l'applicazione
   ```

5. Specificare se si desidera utilizzare una DTD per accertarsi che il documento XML è un documento valido. Ad esempio:

   ```
   <validation>NO</validation>   --> specificare Sì o No
   ```

6. Utilizzare l'elemento <Xcollection> per definire il metodo di memorizzazione e accesso come raccolta XML. I metodi di memorizzazione e accesso consentono di definire se i dati XML sono memorizzati in una raccolta delle tabelle DB2.

   ```
   <Xcollection>
   </Xcollection>
   ```

7. Aggiungere le seguenti informazioni relative al prologo:

   ```
   <prolog>?xml version="1.0"?</prolog>
   ```

   E' richiesto il testo esatto.
8. Aggiungere le tag <doctype></doctype>. Ad esempio:

   ```
   <doctype>! DOCTYPE Order SYSTEM "c:\dxx\samples\dtd\getstart.dtd"</doctype>
   ```

9. Definire root_node utilizzando le tag <root_node></root_node>. Nella tag root_node, specificare gli elementi e gli attributi che costituiscono il documento XML.
10. Dopo la tag <root_node>, associare gli elementi e gli attributi nel documento XML ai nodi degli elementi e attributi che corrispondono ai dati DB2. Questi nodi forniscono un percorso dai dati XML ai dati DB2.
    a. Definire un element_node root di livello superiore. Questo element_node contiene:
       - Nodi di tabella con una condizione di collegamento per specificare la raccolta.
       - Elementi secondari
       - Attributi

       Per specificare i nodi di tabella e le condizioni:

       1) Creare un elemento RDB_node. Ad esempio:

          ```
          <RDB_node>
                </RDB_node>
          ```

       2) Definire un <table_node> per ogni tabella contenente i dati da includere nel documento XML. Ad esempio, se vi sono tre tabelle, ORDER_TAB, PART_TAB e SHIP_TAB, contenenti dati colonna da inserire nel documento, creare un nodo per ciascuna tabella. Ad esempio:

```
<RDB_node>
<table name="ORDER_TAB">
<table name="PART_TAB">
<table name="SHIP_TAB"></RDB_node>
```

3) Definire una condizione di collegamento per le tabelle nella raccolta. La sintassi è:

```
expression = expression AND
expression = expression ...
```

Ad esempio:

```
<RDB_node>
<table name="ORDER_TAB">
<table name="PART_TAB">
<table name="SHIP_TAB">
<condition>
order_tab.order_key = part_tab.order_key AND
             part_tab.part_key = ship_tab.part_key
</condition>
      </RDB_node>
```

4) Specificare una chiave primaria per ogni tabella. La chiave primaria è costituita da una o più colonne denominate chiavi composte. Per specificare la chiave primaria, aggiungere una chiave di attributo all'elemento tabella di RDB_node. Il seguente esempio definisce una chiave primaria per ogni tabella in RDB_node di element_node root, Order:

```
<element_node name="Order">
<RDB_node>
         <table name="order_tab" key="order_key"/>
         <table name="part_tab" key="part_key price"/>
         <table name="ship_tab" key="date mode"/>
         <condition>
order_tab.order_key = part_tab.order_key AND
             part_tab.part_key = ship_tab.part_key
         </condition>
<RDB_node>
```

Le informazioni specificate per la scomposizione vengono ignorate durante la composizione di un documento XML.

L'attributo chiave è obbligatorio per la scomposizione e per l'abilitazione di una raccolta in quanto il file DAD utilizzato deve supportare la composizione e la scomposizione.

b. Definire una tag <element_node> per ciascun elemento del documento XML che si associa ad una colonna della tabella DB2. Ad esempio:

```
<element_node name="name">
</element_node>
```

Un nodo di elemento può disporre di uno dei seguenti tipi di elemento:

- <text_node>: per specificare che il contenuto dell'elemento è associato a una tabella DB2; l'elemento non presenta elementi secondari.
- <attribute_node>: per specificare un attributo; i nodi dell'attributo vengono definiti nel passo successivo.
- Elementi secondari

L'elemento text_node contiene un RDB_node il cui contenuto deve essere associato a una tabella DB2 e a un nome colonna.

Gli RDB_node vengono utilizzati per gli elementi di livello inferiore il cui contenuto deve essere associato a una tabella DB2. RDB_node presenta i seguenti elementi secondari:

- <table>: definisce la tabella corrispondente all'elemento
- <column>: definisce la colonna contenente l'elemento corrispondente
- <condition>: specifica facoltativamente una condizione per la colonna

Ad esempio, è possibile specificare un elemento XML <Tax> per il quale si desidera memorizzare il contenuto senza tag in una colonna denominata TAX:

Documento XML:

```
<Tax>0.02</Tax>
```

In tal caso, il valore `0.02` deve essere memorizzato nella colonna TAX.

Nel file DAD, specificare un <RDB_node> per associare l'elemento XML alla tabella o colonna DB2.

File DAD:

```
<element_node name="Tax">
   <text_node>
<RDB_node>
         <table name="part_tab"/>
         <column name="tax"/>
      </RDB_node>
   </text_node>
</element_node>
```

<RDB_node> specifica che il valore dell'elemento <Tax> è un valore di testo e che i dati vengono memorizzati nella colonna TAX della tabella PART_TAB.

c. Definire una tag <attribute_node> per ciascun attributo del documento XML che si associa ad una colonna della tabella DB2. Ad esempio:

```
<attribute_node name="key">
</attribute_node>
```

L'elemento attribute_node utilizza un RDB_node per associare il valore dell'attributo a una colonna o tabella DB2. RDB_node presenta i seguenti elementi secondari:

- <table>: definisce la tabella corrispondente all'elemento
- <column>: definisce la colonna contenente l'elemento corrispondente
- <condition>: specifica facoltativamente una condizione per la colonna

Ad esempio, è richiesta una chiave attributo per l'elemento <Order>. Il valore della chiave deve essere memorizzato in una colonna PART_KEY.

Documento XML:

```
    <Order key="1">
```

Nel file DAD, creare un elemento attribute_node per la chiave e indicare la tabella in cui verrà memorizzato il valore 1.

File DAD:

```
<attribute_node name="key">
<RDB_node>
    <table name="part_tab">
    <column name="part_key"/>
<RDB_node>
</attribute_node>
```

11. Specificare il tipo di colonna per RDB_node per ogni attribute_node e text_node. Sarà così garantito l'utilizzo del tipo di dati corretto per ogni colonna in cui verranno memorizzati i dati senza tag. Per specificare i tipi di colonna, aggiungere il tipo di attributo all'elemento colonna. Nel seguente esempio il tipo di colonna viene definito come INTEGER:

```
<attribute_node name="key">
<RDB_node>
        <table name="order_tab"/>
        <column name="order_key" type="integer"/>
     </RDB_node>
</attribute_node>
```

12. Verificare se viene posizionata una tag </root_node> dopo l'ultima tag </element_node>.
13. Verificare se viene posizionata una tag </Xcollection> dopo la tag </root_node>.
14. Verificare se viene posizionata una tag </DAD> dopo la tag </Xcollection>.

Abilitazione delle raccolte XML

L'abilitazione di una raccolta XML analizza il file DAD per identificare le tabella e le colonne relative ai documenti XML e registra le informazioni di controllo nella tabella XML_USAGE. L'abilitazione di una raccolta XML è facoltativa per:

- la scomposizione di un documento XML e la memorizzazione dei dati in nuove tabelle
- la composizione di un documento XML utilizzando i dati esistenti di più tabelle.

Se viene utilizzato lo stesso file DAD per la composizione e la scomposizione, è possibile abilitare la raccolta per entrambe le operazioni.

E' possibile abilitare una raccolta XML tramite il wizard di gestione XML Extender, utilizzando il comando **dxxadm** con l'opzione enable_collection o la procedura memorizzata XML Extender, dxxEnableCollection().

Utilizzo del wizard di gestione

Eseguire i passi riportati di seguito per abilitare una raccolta XML:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Operazioni con le raccolte XML** dalla finestra LaunchPad. Viene visualizzata la finestra Selezionare un'attività.
3. Fare clic su **Abilitare una raccolta** e quindi su **Avanti**. Viene visualizzata la finestra Abilitare una raccolta.
4. Selezionare il nome della raccolta che si desidera abilitare nel campo **Nome colonna** dal menu a discesa.
5. Immettere il nome del file DAD nel campo **Nome file DAD** oppure fare clic su **...** per selezionare un file DAD esistente.
6. Facoltativamente, immettere il nome di un table space precedentemente creato nel campo **Table space**.

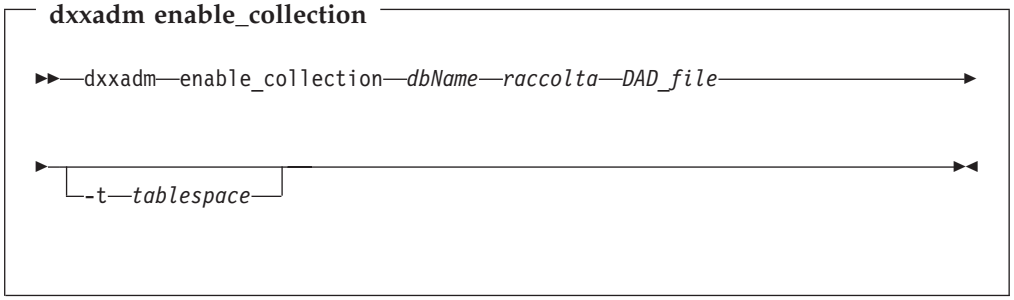
   Il table space conterrà le nuove tabelle DB2 generate per l'eliminazione.
7. Fare clic su **Fine** per abilitare la raccolta e ritornare alla finestra LaunchPad.

- Se la raccolta è stata abilitata con esito positivo, viene visualizzato un messaggio `Abilitazione raccolta terminata con esito positivo`.
- Se la raccolta non è stata abilitata con esito positivo, viene visualizzato un messaggio di errore. Continuare i passi precedenti fino a quando la raccolta è abilitata con esito positivo.

Shell dei comandi DB2

Per abilitare una raccolta XML, immettere il comando **dxxadm**:

Sintassi:



Parametri:

dbName
Il nome del database.

raccolta
Il nome della raccolta XML. Questo valore viene utilizzato come parametro per le procedure memorizzate della raccolta XML.

DAD_file
Il nome del file che contiene la DAD (document access definition).

tablespace
Un table space esistente che contiene nuove tabelle DB2 generate per la scomposizione. Se non viene specificato, viene utilizzato il table space predefinito.

Esempio: Il seguente esempio consente di abilitare una raccolta, denominata sales_ord nel database SALES_DB, utilizzando la shell dei comandi DB2. Il file DAD utilizza l'associazione SQL e viene descritto nella sezione "File DAD: raccolta XML - associazione SQL" a pagina 225.

```
dxxadm enable_collection SALES_DB sales_ord getstart.dad
```

Dopo aver abilitato la raccolta XML, è possibile comporre o scomporre i documenti XML utilizzando le procedure memorizzate XML Extender.

Disabilitazione delle raccolte XML

La disabilitazione di una raccolta XML, elimina il record nella tabella XML_USAGE che identifica le tabelle e le colonne come parte di una raccolta. Le tabelle di dati non vengono cancellate. Disabilitare una raccolta quando si desidera aggiornare la DAD ed è necessario riabilitare una raccolta, oppure per cancellare una raccolta.

E' possibile disabilitare una raccolta XML tramite il wizard di gestione XML Extender, utilizzando il comando **dxxadm** con l'opzione disable_collection o la procedura memorizzata XML Extender, dxxDisableCollection().

Utilizzo del wizard di gestione

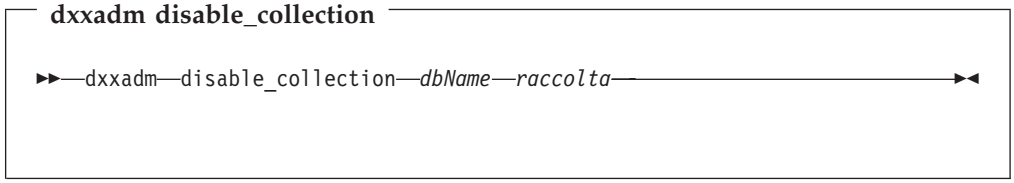
Eseguire i passi riportati di seguito per disabilitare una raccolta XML:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Operazioni con le raccolte XML** dalla finestra LaunchPad per visualizzare le attività relative alle raccolte XML Extender. Viene visualizzata la finestra Selezionare un'attività.
3. Fare clic su **Disabilitare una raccolta XML** e quindi su **Avanti** per disabilitare una raccolta XML. Viene visualizzata la finestra Disabilitare una raccolta.
4. Immettere il nome della raccolta che si desidera disabilitare nel campo **Nome raccolta**.
5. Fare clic su **Fine** per disabilitare la raccolta e ritornare alla finestra LaunchPad.
   - Se la raccolta è stata disabilitata con esito positivo, viene visualizzato un messaggio `Disabilitazione raccolta terminata con esito positivo`.
   - Se la raccolta non è stata disabilitata con esito positivo, viene visualizzata una casella di errore. Continuare i passi precedenti fino a quando la raccolta è disabilitata con esito positivo.

Shell dei comandi DB2

Per disabilitare una raccolta XML, immettere il comando **dxxadm**:

Sintassi:



Parametri:

dbName
: Il nome del database.

raccolta
: Il nome della raccolta XML. Questo valore viene utilizzato come parametro per le procedure memorizzate della raccolta XML.

Esempio:

```
dxxadm disable_collection SALES_DB sales_ord
```

Disabilitazione di un database per XML

Disabilitare il database se si desidera ripulire l'ambiente XML Extender e cancellare le procedure memorizzate, le tabelle di gestione, le UDF e e gli UDT XML Extender. XML Extender disabilita il database a cui si è collegati utilizzando l'istanza corrente.

Quando si disabilita un database per XML, XML Extender effettua le seguenti operazioni:

- Cancella tutti i tipi definiti dall'utente (UDT) e le funzioni definite dall'utente (UDF)
- Cancella le tabelle di controllo con i metadati per XML Extender

- Cancella lo schema db2xml.

Informazioni preliminari

Disabilitare le colonne o le raccolte XML del database che si desidera disabilitare.

Utilizzo del wizard di gestione

Eseguire i passi riportati di seguito per disabilitare un database per i dati XML:

1. Installare e avviare il wizard di gestione. Per ulteriori dettagli, consultare "Avvio del wizard di gestione" a pagina 57.
2. Fare clic su **Disabilitare database** dalla finestra LaunchPad per disabilitare il database corrente.

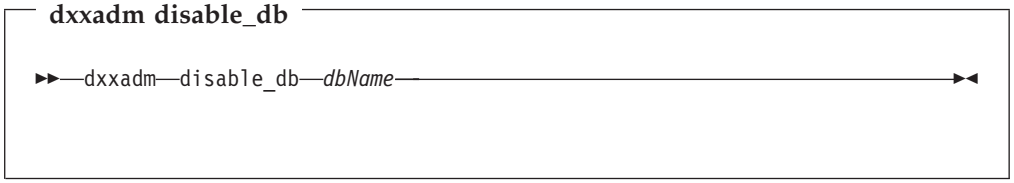
   Se un database non è correntemente abilitato, l'opzione **Abilitare un database** è selezionabile.

   Una volta disabilitato il database, si ritorna alla finestra LaunchPad.

Shell dei comandi DB2

Immettere **dxxadm** dalla riga comandi, specificando il database che si desidera disabilitare.

Sintassi:



Parametri:

dbName
Il nome del database che si desidera disabilitare.

Esempio: Per disabilitare un database esistente denominato SALES_DB, immettere:

```
dxxadm disable_db SALES_DB
```

Parte 3. Programmazione

Questa sezione descrive le tecniche di programmazione per la gestione dei dati XML.

Capitolo 5. Gestione dei dati della colonna XML

Quando si utilizzano le colonne XML, è necessario memorizzare un documento XML come dato della colonna. Questo metodo di memorizzazione e di accesso consente di conservare il documento XML integro, fornendo la possibilità di indicizzare, ricercare e aggiornare il documento e richiamare i relativi dati. Una colonna XML contiene documenti XML nel loro formato originario DB2 come dati di colonna. Una volta abilitato un database per XML, sono disponibili i seguenti UDT (user-defined type):

XMLCLOB
Il contenuto del documento XML memorizzato come CLOB (character document object) in DB2

XMLVARCHAR
Il contenuto del documento XML memorizzato come VARCHAR in DB2

XMLFile
Il documento XML memorizzato in un file su un sistema di file locali

E' possibile creare o modificare tabelle applicative utilizzando gli UDT XML come tipi di dati di colonna. Queste tabelle sono note come tabelle XML. Per informazioni sulla creazione o la modifica di una tabella per XML, consultare la sezione "Creazione o modifica di una tabella XML" a pagina 66.

Una volta abilitata una colonna per XML, è possibile iniziare la gestione del contenuto della colonna XML. Dopo la creazione della colonna XML è possibile effettuare le seguenti attività di gestione:

- Memorizzazione dei documenti XML in DB2
- Richiamo dei documenti o dati XML da DB2
- Aggiornamento dei documenti XML
- Cancellazione dei documenti o dati XML

Per eseguire queste attività, è possibile utilizzare due metodi:

- Il metodo relativo alle *Funzioni cast predefinite*, che convertono il tipo di base SQL negli UDT XML
- Le funzioni UDF (funzioni definite dall'utente) fornite da XML Extender

Questa pubblicazione descrive entrambi i metodi.

I nomi UDT e UDF

Il nome completo di una funzione DB2 è: *nome-schema.nome-funzione*, dove *nome-schema* è l'identificativo che fornisce un raggruppamento logico per gli oggetti SQL. Il nome schema delle UDF XML Extender è db2xml. Il nome schema db2xml è anche il qualificatore degli UDT di XML Extender. In questa pubblicazione, si fa riferimento solo al nome delle funzioni.

Il percorso della funzione è un elenco ordinato di nomi schema. DB2 utilizza l'ordine dei nomi schema dell'elenco per definire i riferimenti alle funzioni e agli UDT. E' possibile specificare il percorso della funzione specificando l'istruzione SQL SET CURRENT FUNCTION PATH. Essa imposta il percorso della funzione nel registro speciale CURRENT FUNCTION PATH.

Per XML Extender, si consiglia di aggiungere lo schema db2xml al percorso di funzione. In tal modo è possibile immettere i nomi UDF e UDT di XML Extender privi del prefisso db2xml. L'esempio riportato di seguito illustra come aggiungere lo schema db2xml al percorso di funzione:

```
SET CURRENT FUNCTION PATH = db2xml, CURRENT FUNCTION PATH
```

Importante: Non aggiungere db2xml come primo schema nel percorso della funzione se ci si collega come db2xml in quanto db2xml viene automaticamente impostato come primo schema. Tale operazione genera una condizione di errore poiché il percorso della funzione inizierà con due schemi db2xml.

Memorizzazione dati

Utilizzando XML Extender è possibile inserire documenti XML integri in una colonna XML. Se si definiscono tabelle laterali, XML Extender automaticamente aggiorna tali tabelle. Quando si memorizza direttamente un documento XML, XML Extender memorizza il tipo di base come tipo XML.

Panoramica delle attività:

1. Accertarsi di aver creato o aggiornato il file DAD.
2. Determinare il tipo di dati da utilizzare quando si memorizza il documento.
3. Scegliere un metodo per la memorizzazione dei dati nella tabella DB2 (funzioni cast o UDF).
4. Specificare un'istruzione SQL INSERT che specifichi la colonna e la tabella XML in cui contenere il documento XML.

XML Extender fornisce due metodi per la memorizzazione dei documenti XML: le funzioni cast predefinite e le funzioni UDF di memorizzazione. La sezione Tabella 8 illustra quando utilizzare ciascun metodo.

Tabella 8. Le funzioni di memorizzazione XML Extender

Tipo di base	Memorizzare in DB2 come...		
	XMLVARCHAR	**XMLCLOB**	**XMLFILE**
VARCHAR	XMLVARCHAR()	N/D	XMLFileFromVarchar()
CLOB	N/D	XMLCLOB()	XMLFileFromCLOB()
FILE	XMLVarcharFromFile()	XMLCLOBFromFile()	XMLFILE

Utilizzare una funzione cast predefinita

Per ciascun UDT, esiste una *funzione cast predefinita* per eseguire un tipo di base SQL nell'UDT. E' possibile utilizzare le funzioni cast fornite da XML Extender nella clausola VALUES per inserire i dati. La sezione Tabella 9 illustra le funzioni cast fornite:

Tabella 9. Le funzioni cast predefinite di XML Extender

Cast utilizzato nella clausola SELECT	Tipo restituito	Descrizione
XMLVARCHAR(VARCHAR)	XMLVARCHAR	Immissione dal buffer di memoria di VARCHAR
XMLCLOB(CLOB)	XMLCLOB	Immissione dal buffer di memoria di CLOB o del relativo indicatore di posizione

Tabella 9. Le funzioni cast predefinite di XML Extender (Continua)

Cast utilizzato nella clausola SELECT	Tipo restituito	Descrizione
XMLFILE(VARCHAR)	XMLFILE	Memorizzazione solo del nome file

Esempio: La seguente istruzione inserisce un tipo VARCHAR cast nel tipo XMLVARCHAR:

```
INSERT INTO sales_tab
VALUES('123456', 'Sriram Srinivasan', db2xml.XMLVarchar(:xml_buff))
```

Utilizzo di una UDF di memorizzazione:

Per ciascun UDT di XML Extender, esiste un'UDF di memorizzazione per eseguire l'importazione di dati in DB2 da una risorsa diversa dal relativo tipo di base. Ad esempio, se si desidera importare un documento file XML in DB2 come XMLCLOB, è possibile utilizzare la funzione XMLCLOBFromFile().

La sezione Tabella 10 illustra le funzioni di memorizzazione fornite da XML Extender.

Tabella 10. UDF di memorizzazione di XML Extender

UDF (user-defined function) di memorizzazione	Tipo restituito	Descrizione
XMLVarcharFromFile()	XMLVARCHAR	Legge un documento XML da un file situato sul server e restituisce il valore di tipo XMLVARCHAR.
XMLCLOBFromFile()	XMLCLOB	Legge un documento XML da un file situato sul server e restituisce il valore di tipo XMLCLOB.
XMLFileFromVarchar()	XMLFILE	Legge un documento XML dalla memoria come VARCHAR, lo scrive in un file esterno e restituisce il nome file come tipo XMLFILE.
XMLFileFromCLOB()	XMLFILE	Legge un documento XML dalla memoria come CLOB o indicatore di posizione CLOB, lo scrive in un file esterno e restituisce il nome file come tipo XMLFILE.

Esempio: la seguente istruzione memorizza un record in una tabella XML utilizzando la funzione XMLCLOBFromFile() come XMLCLOB.

```
EXEC SQL INSERT INTO sales_tab(ID, NAME, ORDER)
    VALUES( '1234', 'Sriram Srinivasan,
      XMLCLOBFromFile('c:\dxx\samples\cmd\getstart.xml'))
```

Il precedente esempio esegue l'importazione dell'oggetto XML dal file denominato `c:\dxx\samples\cmd\getstart.xml` nella colonna ORDER della tabella SALES_TAB.

Richiamo dati

Utilizzando XML Extender, è possibile richiamare un documento intero o il contenuto degli elementi e gli attributi. Quando si richiama direttamente una colonna XML, XML Extender restituisce l'UDT come tipo di colonna. Per informazioni dettagliate sul richiamo dei dati consultare le seguenti sezioni:

- "Richiamo di un documento intero"
- "Richiamo del contenuto degli elementi e dei valori degli attributi" a pagina 102

XML Extender fornisce due metodi di richiamo dati: le funzioni cast predefinite e la UDF di sovraccarico Content(). La sezione Tabella 11 illustra quando utilizzare ciascun metodo.

Tabella 11. Le funzioni di richiamo di XML Extender

Tipo XML	Richiamare da DB2 come...		
	VARCHAR	**CLOB**	**FILE**
XMLVARCHAR	VARCHAR	N/D	Content()
XMLCLOB	N/D	XMLCLOB	Content()
XMLFILE	N/D	Content()	FILE

Richiamo di un documento intero

Panoramica delle attività:

1. Accertarsi di aver memorizzato il documento XML in una tabella XML e stabilire quali dati richiamare.
2. Scegliere un metodo per il richiamo dei dati nella tabella DB2 (funzioni cast o UDF).
3. Se si utilizzano le UDF Content() sovraccaricate, determinare i tipi di dati associati ai dati richiamati e i tipi di dati da esportare.
4. Specificare un'interrogazione SQL che indichi la colonna e la tabella XML da cui richiamare il documento XML.

XML Extender fornisce de metodi per il richiamo dei dati:

Utilizzare una funzione cast predefinita

Utilizzare la funzione cast predefinita fornita da DB2 per gli UDT per convertire un UDT XML in un tipo di base SQL e quindi utilizzarlo. E' possibile utilizzare le funzioni cast fornite da XML Extender nell'istruzione SELECT per richiamare i dati. La sezione Tabella 12 illustra le funzioni cast fornite:

Tabella 12. Le funzioni cast predefinite di XML Extender

Cast utilizzato nella clausola select	Tipo restituito	Descrizione
varchar(XMLVARCHAR)	VARCHAR	documento XML in VARCHAR
clob(XMLCLOB)	CLOB	documento XML in CLOB
varchar(XMLFile)	VARCHAR	nome file XML in VARCHAR

Esempio: il seguente esempio richiama XMLVARCHAR e lo memorizza come tipo di dati VARCHAR:

```
EXEC SQL SELECT db2xml.varchar(order) from sales_tab
```

Utilizzare la UDF di sovraccarico Content()

Utilizzare la UDF Content() per richiamare il contenuto del documento dalla memoria esterna oppure per esportare il documento dalla memoria interna in un *file esterno* del server DB2.

Ad esempio, si dispone del documento XML memorizzato come XMLFILE e si desidera eseguirlo nella memoria, è possibile utilizzare la UDF Content() che richiama un tipo di dati XMLFILE come input e restituire un CLOB.

La UDF Content() esegue due differenti funzioni di richiamo, a seconda del tipo di dati specificato. Tale funzione:

Richiama un documento dalla memoria esterna e lo memorizza in quella interna.

E' possibile utilizzare Content() per richiamare il documento XML in un buffer di memoria o in un indicatore di posizione CLOB quando il documento viene memorizzato come file esterno.. Utilizzare la seguente sintassi di funzione, dove *xmlobj* è la colonna XML per la quale viene eseguita l'interrogazione:

XMLFILE to CLOB: Richiama i dati da un file e li esporta in un indicatore di posizione CLOB.

```
Content(xmlobj XMLFile)
```

Richiama un documento dalla memoria interna e lo esporta in un file esterno

E' inoltre possibile utilizzare Content() per richiamare un documento XML memorizzato in DB2 come tipo di dati XMLCLOB e per esportarlo in un file del sistema file del server database. Quindi restituisce il nome file di tipo VARCHAR. Utilizzare la seguente sintassi di funzione, dove *xmlobj* è la colonna XML per la quale viene eseguita l'interrogazione *filename* è il file esterno. *XML type* può essere un tipo di dati XMLVARCHAR o XMLCLOB.

tipo XML **nel file esterno:** Richiama il contenuto XML memorizzato come tipo di dati XML e lo esporta in un file esterno.

```
Tipo XML Content(xmlobj,
nomefile varchar(512))
```

Dove:

xmlobj Il nome della colonna XML da cui richiamare il contenuto XML; *xmlobj* può essere di tipo XMLVARCHAR o XMLCLOB.

filename
Il nome del file in cui memorizzare i dati XML.

Nell'esempio riportato di seguito, un piccolo segmento deI programma C con *SQL integrata* illustra come un documento XML viene richiamato da un file nella memoria. In questo esempio la colonna BOOK è di tipo XMLFILE.

```
EXEC SQL BEGIN DECLARE SECTION;
             SQL TYPE IS CLOB_LOCATOR xml_buff;
          EXEC SQL END DECLARE SECTION;
          EXEC SQL CONNECT TO SALES_DB
          EXEC SQL DECLARE c1 CURSOR FOR
              SELECT Content(order) from sales_tab
                        EXEC SQL OPEN c1;
          do {
```

```
    EXEC SQL FETCH c1 INTO :xml_buff;
    if (SQLCODE != 0) {
       break;
       }
    else {
      /* do whatever you need to do with the XML doc in buffer */
      }
    }
EXEC SQL CLOSE c1;
EXEC SQL CONNECT RESET;
```

Richiamo del contenuto degli elementi e dei valori degli attributi

E' possibile richiamare (estrarre) il contenuto di un *elemento* o un valore *attributo* da uno o più documenti XML (ricerca documenti della raccolta o del documento singolo). XML Extender fornisce funzioni di estrazione definite dall'utente che è possibile specificare nella clausola SQL SELECT per ciascun tipo di dati SQL.

Il richiamo del contenuto e dei valori degli elementi e degli attributi è utile nello sviluppo delle applicazioni, in quanto è possibile accedere ai dati XML come dati relazionali. Ad esempio, si supponga di disporre di 1000 documenti XML memorizzati nella colonna ORDER nella tabella SALES_TAB. E' possibile richiamare i nomi di tutti i clienti che hanno ordinato gli articoli utilizzando la seguente istruzione SQL con le UDF di estrazione nella clausola SELECT per richiamare le seguenti informazioni:

```
SELECT extractVarchar(Order, '/Order/Customer/Name') from sales_order_view
   WHERE price > 2500.00
```

In questo esempio, l'UDF di estrazione richiama l'elemento <customer> dalla colonna ORDER come tipo di dati VARCHAR. Il percorso di ubicazione è `/Order/Customer/Name` (consultare la sezione "Percorso di ubicazione" a pagina 44 per la sintassi del percorso di ubicazione). Inoltre, il numero dei valori restituiti viene ridotto utilizzando la clausola WHERE, in cui si specifica che solo il contenuto dell'elemento <customer> con un elemento secondario <ExtendedPrice> ha un valore maggiore di 2500.00.

Per estrarre il contenuto dell'elemento o i valori dell'attributo: Utilizzare le UDF di estrazione elencate in Tabella 13 a pagina 103 utilizzando la seguente sintassi come tabella o *funzione scalare*:

extract*retrieved_datatype*(*xmlobj*, *path*)

Dove:

retrieved_datatype
: E' il tipo di dati restituito dalla funzione di estrazione; i tipi di dati sono i seguenti:
  - INTEGER
  - SMALLINT
  - DOUBLE
  - REAL
  - CHAR
  - VARCHAR
  - CLOB
  - DATE
  - TIME

- TIMESTAMP
- FILE

xmlobj Il nome della colonna XML da cui estrarre l'elemento o l'attributo. Questa colonna deve essere definita come uno dei seguenti UDT XML:

- XMLVARCHAR
- XMLCLOB come LOCATOR
- XMLFILE

percorso

Il percorso di ubicazione dell'elemento o dell'attributo nel documento XML (ad esempio `/Order/Customer/Name`). Per informazioni sulla sintassi del percorso di ubicazione, consultare la sezione "Percorso di ubicazione" a pagina 44.

Importante: Notare che le UDF di estrazione supportano percorsi di ubicazione che presentano predicati con attributi ma non elementi. Ad esempio, il seguente predicato è supportato:

```
'/Order/Part[@color="black "]/ExtendedPrice'
```

Mentre il seguente predicato non è supportato:

```
'/Order/Part/Shipment/[Shipdate < "11/25/00"]'
```

Tabella 13 illustra le funzioni di estrazione, sia in formato scalare sia in formato tabella:

Tabella 13. Le funzioni di estrazione XML Extender

Funzione scalare	Funzione di tabella	Nome colonna restituita (funzione tabella)	Tipo restituito
extractInteger()	extractIntegers()	returnedInteger	INTEGER
extractSmallint()	extractSmallints()	returnedSmallint	SMALLINT
extractDouble()	extractDoubles()	returnedDouble	DOUBLE
extractReal()	extractReals()	returnedReal	REAL
extractChar()	extractChars()	returnedChar	CHAR
extractVarchar()	extractVarchars()	returnedVarchar	VARCHAR
extractCLOB()	extractCLOBs()	returnedCLOB	CLOB
extractDate()	extractDates()	returnedDate	DATE
extractTime()	extractTimes()	returnedTime	TIME
extractTimestamp()	extractTimestamps()	returnedTimestamp	TIMESTAMP

Esempio di funzione scalare:

Nel seguente esempio, un valore viene restituito quando il valore attributo della chiave è ″1″. Il valore viene estratto come un intero automaticamente convertito in un tipo DECIMAL.

```
CREATE TABLE t1(key decimal(3,2));
INSERT into t1 values
SELECT * from table(db2xml.extractInteger(db2xml.XMLFile
   ('c:\dxx\samples\xml\getstart.xml'), '/Order/[@key="1"]'));
SELECT * from t1;
```

Esempio di funzione di tabella:

Nel seguente esempio, ogni valore chiave per l'ordine di acquisto viene estratto come INTEGER

```
SELECT * from table(db2xml.extractIntegers(db2xml.XMLFile
   ('c:\dxx\samples\xml\getstart.xml'), '/Order/@key')) as x;
```

Aggiornamento dei dati XML

Con XML Extender, è possibile aggiornare l'intero documento XML sostituendo i dati di colonna XML oppure è possibile aggiornare i valori degli elementi o attributi specificati.

Panoramica delle attività:

1. Accertarsi di aver memorizzato il documento XML in una tabella XML e stabilire quali dati richiamare.
2. Scegliere un metodo per l'aggiornamento dei dati nella tabella DB2 (funzioni cast o UDF).
3. Specificare un'interrogazione SQL che specifica la tabella e la colonna XML da aggiornare.

Importante: quando si aggiorna una colonna abilitata per XML, XML Extender automaticamente aggiorna le tabelle laterali in base alle modifiche apportate. Tuttavia, non aggiornare queste tabelle senza aggiornare il documento XML originale che è stato memorizzato nella colonna XML modificando l'elemento XML o il valore attributo corrispondente. I seguenti aggiornamenti possono causare problemi di incongruenza dati.

Per aggiornare un documento XML:

Utilizzare uno dei seguenti metodi:

Utilizzare una funzione cast predefinita

Per ciascun UDT (user-defined type), esiste una funzione cast predefinita per eseguire il tipo di base nell'UDT. E' possibile utilizzare le funzioni cast fornite da XML Extender per aggiornare il documento XML. La Tabella 9 a pagina 98 illustra le funzioni cast predefinite e specifica la colonna ORDER con un UDT differente, fornito da XML Extender.

Esempio: Aggiorna il tipo XMLVARCHAR, dal tipo VARCHAR cast presumendo che `xml_buf` sia una variabile host definita come tipo VARCHAR.

```
UPDATE sales_tab VALUES('123456', 'Sriram Srinivasan',
  db2xml.XMLVarchar(:xml_buff))
```

Utilizzo di una UDF di memorizzazione

Per ciascun UDT di XML Extender, esiste un'UDF di memorizzazione per eseguire l'importazione di dati in DB2 da una risorsa diversa dal relativo tipo di base. E' possibile utilizzare un'UDF di memorizzazione per aggiornare l'intero documento XML mediante la relativa sostituzione.

Esempio: il seguente esempio esegue l'aggiornamento di un documento XML utilizzando la funzione XMLVarcharFromFile():

```
UPDATE sales_tab
       set order = XMLVarcharFromFile('c:\dxx\samples\cmd\getstart.xml')
   WHERE sales_person = 'Sriram Srinivasan'
```

L'esempio precedente aggiorna l'oggetto XML dal file `c:\dxx\samples\cmd\getstart.xml` nella colonna ORDER nella tabella SALES_TAB.

Consultare Tabella 10 a pagina 99 per un elenco delle funzioni di memorizzazione fornite da XML Extender.

Per aggiornare elementi e attributi specifici di un documento XML:

Utilizzare la UDF Update() per aggiornare un valore alla volta, in un documento, anziché aggiornare l'intero documento. Se si utilizza la UDF, specificare un percorso di ubicazione e il valore dell'elemento o dell'attributo rappresentato dal percorso di ubicazione da sostituire. Per informazioni sulla sintassi del percorso di ubicazione, consultare la sezione "Percorso di ubicazione" a pagina 44. Non è necessario modificare il documento XML: XML Extender effettua automaticamente le modifiche.

La UDF Update aggiorna l'intero file XML e ricostruisce il file in base alle informazioni ricevute dal parser XML. Consultare la sezione "Elaborazione del documento XML con la funzione Update" a pagina 167 per informazioni su come la UDF Update elabora il documento ed esempi di documenti prima e dopo l'aggiornamento.

Sintassi:

```
Update(xmlobj, path, value)
```

Dove:

xmlobj Il nome della colonna XML per la quale aggiornare il valore dell'elemento o attributo.

percorso
Il percorso di ubicazione dell'elemento o dell'attributo di cui eseguire l'aggiornamento. Per informazioni sulla sintassi del percorso di ubicazione, consultare la sezione "Percorso di ubicazione" a pagina 44. Per le considerazioni relative a ricorrenze multiple, consultare la sezione "Ricorrenze multiple" a pagina 169.

valore E' il valore che deve essere aggiornato.

Esempio: la seguente istruzione aggiorna il valore dell'elemento <Customer> nella stringa dei caratteri IBM, utilizzando la UDF Update():

```
UPDATE sales_tab
   set order = Update(order, '/Order/Customer/Name', 'IBM')
   WHERE sales_person = 'Sriram Srinivasan'
```

Ricorrenze multiple:

Quando un percorso di ubicazione viene fornito in UDF Update(), il contenuto di ciascun elemento o attributo con un percorso corrispondente viene aggiornato con il valore fornito. Ciò significa che, se un documento presenta ricorrenze multiple di percorsi di ubicazione, la funzione Update sostituisce i valori esistenti con il valore fornito nel parametro *value*.

Ricerca dei documenti XML

La tecnica di ricerca dei dati XML è simile a quella per il richiamo: entrambe le tecniche richiamano i dati per effettuare ulteriori manipolazioni utilizzando la clausola WHERE che definisce i predicati come criteri di richiamo.

XML Extender fornisce vari metodi di ricerca documenti XML in una colonna XML, a seconda dell'applicazione utilizzata. Inoltre fornisce la funzione di ricerca struttura documenti e restituisce i risultati in base al contenuto degli elementi e dei valori attributo. E' possibile ricercare una vista della colonna XML e delle relative tabelle laterali, ricercare direttamente le tabelle laterali per migliorare le prestazioni oppure utilizzare le UDF di estrazione con le clausole WHERE. Inoltre è possibile utilizzare Text Extender e ricercare i dati di colonna all'interno del contenuto strutturale relativo a una stringa di testo.

Con XML Extender è possibile utilizzare gli indici nelle colonne delle tabelle laterali contenenti i valori attributo o il contenuto degli elementi XML estratti dai documenti XML, per una ricerca con velocità elevata. Specificando il tipo di dati di un elemento o attributo, è possibile ricercare il tipo di dati SQL generico oppure effettuare ricerca delimitate. Nell'esempio degli ordini di acquisto, è possibile ricercare tutti gli ordini con prezzo superiore a 2500.00.

Inoltre, è possibile utilizzare DB2 UDB Text Extender per effettuare ricerche di testo strutturale o ricerche di testo completo. Ad esempio, è possibile disporre di una colonna RESUME che contiene il riepilogo in formato XML. Si desidera conoscere il nome delle applicazioni con funzioni Java. E' possibile utilizzare DB2 Text Extender per ricercare nei documenti XML tutte le ricorrenze in cui l'elemento <skill> contiene la stringa di caratteri JAVA.

Le seguenti sezioni descrivono i metodi di ricerca:

- "Ricerca di documenti XML in base alla struttura"
- "Utilizzo di Text Extender per la ricerca di testo strutturale" a pagina 108

Ricerca di documenti XML in base alla struttura

Utilizzando le funzioni di ricerca XML Extender, è possibile ricercare dati XML in una colonna basata sulla struttura del documento, con elementi e attributi. Per ricercare i dati di colonna utilizzare un'istruzione SELECT in vari modi e restituire una *serie di risultati* sulla base delle corrispondenze agli elementi e attributi del documento. E' possibile ricercare i dati di colonna utilizzando i seguenti metodi:

- Ricerca con interrogazione diretta nelle tabelle laterali
- Ricerca da una *vista di unione*
- Ricerca con le UDF di estrazione
- Ricerca di elementi o attributi con ricorrenza multipla

Questi metodi sono descritti nelle sezioni riportate di seguito e utilizzano esempi con il seguente scenario. La tabella applicativa SALES_TAB presenta una colonna XML denominata ORDER. Questa colonna presenta tre tabelle laterali, ORDER_SIDE_TAB, PART_SIDE_TAB e SHIP_SIDE_TAB. La vista predefinita, sales_order_view, è stata specificata quando è stata abilitata la colonna ORDER e le tabelle vengono unite utilizzando la seguente istruzione CREATE VIEW:

```
CREATE VIEW sales_order_view(invoice_num, sales_person, order,
                             order_key, customer, part_key, price, date)
AS
     SELECT sales_tab.invoice_num, sales_tab.sales_person, sales_tab.order,
```

```
        order_side_tab.order_key, order_side_tab.customer,
        part_side_tab.part_key, ship_side_tab.date
  FROM sales_tab, order_side_tab, part_side_tab, ship_side_tab
  WHERE sales_tab.invoice_num = order_side_tab.invoice_num
        AND sales_tab.invoice_num = part_side_tab.invoice_num
        AND sales_tab.invoice_num = ship_side_tab.invoice_num
```

Ricerca con interrogazione diretta nelle tabelle laterali

L'interrogazione diretta con la ricerca di interrogazioni secondarie fornisce la migliore prestazione della ricerca strutturale quando le tabelle laterali sono indicizzate. E' possibile utilizzare un'interrogazione o un'interrogazione secondaria per ricercare correttamente le tabelle laterali.

Esempio: la seguente istruzione utilizza un'interrogazione e un'interrogazione secondaria per ricercare direttamente una tabella laterale:

```
SELECT sales_person from sales_tab
   WHERE invoice_num in
     (SELECT invoice_num from part_side_tab
      WHERE price > 2500.00)
```

In questo esempio, invoice_num è la chiave primaria nella tabella SALES_TAB.

Ricerca da una vista di unione

E' possibile che XML Extender effettui la creazione di una vista predefinita che unisce la tabella applicativa e le tabelle laterali utilizzando un ID univoco. E' possibile utilizzare questa vista predefinita oppure qualsiasi vista che unisce una tabella applicativa e le tabelle laterali, per ricercare dati di colonna e per effettuare interrogazioni nelle tabelle laterali. Questo metodo fornisce una vista virtuale singola della tabella applicativa e delle relative tabelle laterali. Tuttavia, tenere presente che la complessità di un'interrogazione è direttamente proporzionale al numero di tabelle laterali create.

Suggerimento: è possibile utilizzare *root_id* o DXXROOT_ID (creato da XML Extender), per unire le tabelle quando si crea la propria vista.

Esempio: la seguente istruzione ricerca la vista

```
SELECT sales_person from sales_order_view
   WHERE price > 2500.00
```

L'istruzione SQL restituisce i valori sales_person della tabella della vista di unione sales_order_view che presenta gli ordini di articoli con un prezzo superiore a 2500.00.

Ricerca con le UDF di estrazione

E' inoltre possibile utilizzare le UDF di estrazione di XML Extender per ricercare elementi o attributi, nel caso in cui non sono stati creati indici o tabelle laterali per la tabella applicativa. L'utilizzo delle UDF di estrazione per eseguire la scansione dei dati XML rende l'operazione molto complessa inoltre è necessario utilizzarle con le clausole WHERE che riducono il numero dei documenti XML inclusi nella ricerca.

Esempio: la seguente istruzione ricerca un'UDF di estrazione di XML Extender:

```
SELECT sales_person from sales_tab
     WHERE extractVarchar(order, '/Order/Customer/Name')
    like '%IBM%'
AND invoice_num > 100
```

In questo esempio, la UDF di estrazione estrae gli elementi </Order/Customer/Name> con il valore IBM.

Ricerca di elementi o attributi con ricorrenza multipla

Per la ricerca di elementi o attributi con ricorrenza multipla, utilizzare la clausola DISTINCT per evitare valori duplicati.

Esempio: la seguente istruzione ricerca la clausola DISTINCT:

```
SELECT sales_person from sales_tab
            WHERE invoice_num in
              (SELECT DISTINCT invoice_num from part_side_tab
               WHERE price > 2500.00 )
```

In questo esempio, il file DAD specifica `/Order/Part/Price` con ricorrenza multipla e crea la tabella laterale PART_SIDE_TAB. La tabella PART_SIDE_TAB può presentare più di una riga con lo stesso invoice_num. Con DISTINCT vengono restituiti solo i valori univoci.

Utilizzo di Text Extender per la ricerca di testo strutturale

Quando si ricerca la struttura del documento XML, XML Extender ricerca i valori degli attributi e dell'elemento che sono stati convertiti nei tipi di dati generali, ma non ricerca il testo. E' possibile utilizzare DB2 UDB Text Extender per la ricerca di testo completo o strutturale in una colonna abilitata per XML. Text Extender supporta la ricerca del documento XML in DB2 UDB versione 6.1 o successiva. Text Extender è disponibile su sistemi operativi Windows, su AIX e Sun Solaris.

Ricerca di testo strutturale
: La ricerca di stringhe di testo che si basano sulla struttura ad albero del documento XML. Ad esempio, se si dispone della struttura del documento `/Order/Customer/Name` e si desidera ricercare la stringa di caratteri "IBM" all'interno dell'elemento secondario <Customer>, è possibile utilizzare la ricerca di testo strutturale. E' possibile che il documento contenga la stringa IBM in un elemento secondario <Comment> o come nome di parte del prodotto. Il testo strutturale viene ricercato solo negli elementi specificati per la stringa. In questo esempio, vengono rilevati solo i documenti che presentano IBM nell'elemento secondario </Order/Customer/Name>; i documenti che presentano IBM in altri elementi e non nell'elemento secondario </Order/Customer/Name> non vengono restituiti.

Ricerca testo completo
: Ricerca stringhe di testo in qualsiasi punto della struttura del documento, senza tener conto degli elementi o degli attributi. Nell'esempio precedente, vengono restituiti tutti i documenti che presentano la stringa IBM senza considerare il punto in cui ricorre.

Per utilizzare la ricerca Text Extender, è necessario installare DB2 Text Extender e abilitare il database e le tabelle come descritto di seguito. Per le modalità di utilizzo della ricerca Text Extender, consultare il capitolo relativo alla ricerca con le UDF di Text Extender della pubblicazione *DB2 Universal Database Text Extender Administration and Programming*.

Abilitazione di una colonna XML per Text Extender

Si supponga di disporre di un database abilitato per XML, effettuare le seguenti operazioni per abilitare Text Extender per la ricerca del contenuto di una colonna abilitata per XML. Ad esempio, il database è denominato SALES_DB, la tabella ORDER e i nomi della colonna XML sono XVARCHAR e XCLOB:

1. Consultare il file `install.txt` sul CD degli extender per le modalità di installazione di Text Extender.
2. Immettere il comando **txstart** da una delle seguenti ubicazioni:
   - Nei sistemi operativi UNIX, immettere il comando dal prompt dei comandi del proprietario dell'istanza.
   - Su Windows NT, immettere il comando dalla finestra dei comandi in cui DB2INSTANCE è specificato.
3. Aprire la finestra della riga comandi di Text Extender. Questo esempio specifica un database denominato SALES_DB e una tabella denominata ORDER, con due colonne XML denominate XVARCHAR e XCLOB. E' necessario eseguire i programmi di esempio in `dxx\samples\c`.
4. Collegamento al database. Nel prompt dei comandi **db2tx**, immettere:
   ```
   'connect to SALES_DB'
   ```
5. Abilitare il database per Text Extender.

   Nel prompt dei comandi **db2tx**, immettere:
   ```
   'enable database'
   ```
6. Abilitare le colonne della tabella XML per Text Extender, definendo i tipi di dati del documento XML, la lingua, le code page e altre informazioni relative alla colonna.
   - Per la colonna VARCHAR XVARCHAR, immettere:
     ```
     'enable text column order xvarchar function db2xml.varchartovarchar handle
     varcharhandle ccsid 850 language us_english format xml indextype precise
         indexproperty sections_enabled
     documentmodel (Order) updateindex update'
     ```
   - Per la colonna CLOB XCLOB, immettere:
     ```
     'enable text column order xclob function db2xml.clob handle clobhandle
      ccsid 850 language us_english indextype precise updateindex update'
     ```
7. Verificare lo stato dell'indice.
   - Per la colonna XVARCHAR, immettere: `get index status order handle varcharhandle`
   - Per la colonna XCLOB, immettere: `get index status order handle clobhandle`
8. Definire il modello del documento XML in un file INI del modello documento denominato `desmodel.ini`. Questo file è ubicato in: `/db2tx/txins000` per UNIX e `\instance\\db2tx\txins000` per Windows NT e le sezioni in un file di inizializzazione. Ad esempio, per `textmodel.ini`:
   ```
   ;list of document models
   [MODELS]
   modelname=Order

   ; an 'Order' document model definition
   ; left side = section name identifier
   ; right side = section name tag

   [Order]
   Order = /Order
   Order/Customer/Name = /Order/Customer/Name
   Order/Customer/Email = /Order/Customer/Email
   Order/Part/@color = /Order/Part/@color
   Order/Part/Shipment/ShipMode = /Order/Part/Shipment/ShipMode
   ```

Ricerca del testo utilizzando Text Extender

La funzione di ricerca di Text Extender è strettamente correlata alla ricerca strutturale del documento XML Extender. Il metodo consigliato è la creazione di

un'interrogazione che effettua ricerche negli elementi o negli attributi del documento e utilizza Text Extender per ricercare il contenuto dell'elemento o i valori dell'attributo.

Esempio: le seguenti istruzioni ricercano un testo del documento XML con Text Extender. Dalla finestra Comandi DB2, immettere:

```
'connect to SALES_DB'
'select xvarchar from order where db2tx.contains(varcharhandle,
 'model Order section(Order/Customer/Name) "Motors"')=1'
'select xclob from order where db2tx.contains(clobhandle,
 'model Order section(Order/Customer/Name) "Motors"')=1'
```

Le ricerche UDF Contains() di Text Extender.

Questo esempio non riporta tutte le operazioni necessarie per utilizzare Text Extender nella ricerca di dati di colonna. Per ulteriori informazioni sui concetti e le funzioni della ricerca Text Extender , consultare il capitolo relativo alla ricerca con le UDF di Text Extender della pubblicazione *DB2 Universal Database Text Extender Administration and Programming*.

Cancellazione dei documenti XML

Utilizzare l'istruzione SQL DELETE per cancellare una riga del documento XML da una colonna XML. E' possibile specificare clausole WHERE per definire ulteriormente i documenti da cancellare.

Esempio: le seguenti istruzioni cancellano tutti i documento con valore <ExtendedPrice> superiore a 2500.00.

```
DELETE from sales_tab
       WHERE invoice_num in
          (SELECT invoice_num from part_side_tab
           WHERE price > 2500.00)
```

Limitazioni durante il richiamo di funzioni da JDBC

Quando si utilizzano indicatori di parametro in funzioni, una limitazione JDBC richiede che l'indicatore di parametro per la funzione sia collegato al tipo di dati della colonna in cui verranno inseriti i dati restituiti. La logica di selezione della funzione non conosce il tipo di dati in cui verrà convertito l'argomento e non è in grado di risolvere il riferimento.

Ne risulta che JDBC non può risolvere il seguente codice:

```
db2xml.XMLdefault_casting_function(lunghezza)
```

Per fornire un tipo all'indicatore di parametro, ad esempio VARCHAR, è possibile utilizzare la specifica CAST; in questo modo la logica di selezione della funzione può procedere:

```
db2xml.XMLdefault_casting_function(CAST(? AS cast_type(lunghezza))
```

Esempio 1: Nel seguente esempio, l'indicatore di parametro viene emesso come VARCHAR. Il parametro che viene trasmesso è un documento XML, emesso come VARCHAR(1000) e inserito nella colonna ORDER.

```
String query = "insert into sales_tab(invoice_num, sales_person, order) values
    (?,?,db2xml.XMLVarchar(cast (? as varchar(1000))))";
```

Esempio 2: Nel seguente esempio, l'indicatore di parametro viene emesso come VARCHAR. Il parametro che viene trasmesso è un nome file e il suo contenuto viene convertito in VARCHAR e inserito nella colonna ORDER.

```
String query = "insert into sales_tab(invoice_num, sales_person, order) values
    (?,?,db2xml.XMLVarcharfromFILE(cast (? as varchar(1000))))";
```

Capitolo 6. Gestione dei dati delle raccolte XML

Una raccolta XML è una serie du tabelle relazioni che contengono i dati associati ai documenti XML. Questo metodo di accesso e memorizzazione consente di comporre un documento XML dai dati esistenti, scomporre un documento XML e utilizzare XML come metodo di interscambio.

Le tabelle relazionali possono essere tabelle nuove che XML Extender genera durante la scomposizione dei documenti XML o tabelle esistenti che presentano i dati che devono essere utilizzati con XML Extender per generare documenti XML. I dati delle colonne di queste tabelle non contengono tag XML ma contengono il contenuto e i valori associati agli elementi e agli attributi, rispettivamente. Le procedure memorizzate rappresentano metodi di memorizzazione e di accesso per salvare, richiamare, aggiornare, ricercare e cancellare i dati della raccolta XML.

I limiti di parametro utilizzati dalle procedure memorizzate della raccolta XML sono descritti in "Appendice D. Limiti di XML Extender" a pagina 241.

E' possibile aumentare le dimensioni CLOB per i risultati di procedure memorizzate, come descritto nella sezione "Incremento del limite CLOB" a pagina 174.

Per informazioni sulla gestione della raccolta XML consultare le seguenti sezioni:

- "Composizione dei documenti XML dai dati DB2"
- "Scomposizione dei documenti XML in dati DB2" a pagina 120

Composizione dei documenti XML dai dati DB2

La composizione è la creazione di una serie di documenti XML che si basa sui dati relazionali contenuti in una raccolta XML. E' possibile comporre documenti XML utilizzando procedure memorizzate. Per utilizzare queste procedure memorizzate, è necessario creare un file DAD che specifica l'associazione tra il documento XML e la struttura della tabella DB2. Le procedure memorizzate utilizzano il file DAD per comporre il documento XML. Per informazioni sulla creazione di un file DAD consultare la sezione "Pianificazione per le raccolte XML" a pagina 46.

Informazioni preliminari

- Associare la struttura del documento XML alle tabelle relazionali che contengono il contenuto dell'elemento e i valori dell'attributo.
- Selezionare un metodo di associazione: associazione SQL o associazione RDB_node.
- Preparare il file DAD. Per ulteriori informazioni consultare la sezione "Pianificazione per le raccolte XML" a pagina 46.
- Facoltativamente, abilitare la raccolta XML.

Composizione del documento XML

XML Extender fornisce due procedure memorizzate, dxxGenXML() e dxxRetrieveXML(), per comporre i documenti XML.

dxxGenXML()

Questa procedura memorizzata viene utilizzata per le applicazioni che

eseguono aggiornamenti occasionali o per le applicazioni che non sono in grado di eseguire la completa gestione dei dati XML. La procedura memorizzata dxxGenXML() non richiede una raccolta abilitata ma utilizza un file DAD.

La procedura memorizzata dxxGenXML() crea documenti XML utilizzando i dati memorizzati nelle tabelle della raccolta XML specificati dall'elemento <Xcollection> nel file DAD. Questa procedura inserisce ciascun documento XML come riga all'interno di una *tabella dei risultati*. E' inoltre possibile posizionare il cursore sulla tabella dei risultati ed eseguire la lettura sequenziale della serie di risultati. La tabella dei risultati viene creata dall'applicazione e presenta sempre una colonna di tipo VARCHAR, CLOB, XMLVARCHAR o XMLCLOB.

Inoltre, se si specifica YES per l'elemento di convalida nel file DAD, XML Extender aggiunge la colonna DXX_VALID di tipo INTEGER alla tabella dei risultati e inserisce il valore 1 per un documento XML valido e 0 per un documento non valido.

La procedura memorizzata dxxGenXML() consente inoltre di specificare il numero massimo di righe che sono state generate nella tabella dei risultati. In tal modo si riduce il tempo di elaborazione. La procedura memorizzata restituisce il numero effettivo di righe contenute nella tabella con i codici di ritorno e i messaggi di errore.

La procedura memorizzata corrispondente per la scomposizione è dxxShredXML() la quale rileva anche il file DAD come parametro di input e non richiede che la raccolta XML sia abilitata.

Per comporre una raccolta XML: dxxGenXML()

Comprende il richiamo di una procedura memorizzata utilizzando la seguente procedura:

```
dxxGenXML(CLOB(100K)     DAD,             /* input */
          char(resultTabName) resultTabName,   /* input */
               integer        overrideType,    /* input */
               varchar(1024)  override,        /* input */
               integer        maxRows,         /* input */
               integer        numRows,         /* output */
               long           returnCode,      /* output */
               varchar(1024)  returnMsg)       /* output */
```

Per la sintassi completa e gli esempi consultare la sezione "dxxGenXML()" a pagina 183.

Esempio: Nel seguente esempio viene composto un documento XML:

```
#include "dxx.h" #include "dxxrc.h"

            EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
     SQL TYPE is CLOB(100K) dad;              /* DAD */
     SQL TYPE is CLOB_FILE  dadFile;          /* dad file */
               char    result_tab[32];   /* name of the result table */
               char    override[2];      /* override, will set to NULL*/
               short   overrideType;     /* defined in dxx.h */
               short   max_row;          /* maximum number of rows */
               short   num_row;          /* actual number of rows */
               long    returnCode;       /* return error code */
               char    returnMsg[1024];  /* error message text */
     short                   dad_ind;
               short   rtab_ind;
               short   ovtype_ind;
```

```
                short   ov_inde;
                short   maxrow_ind;
                short   numrow_ind;
                short   returnCode_ind;
                short   returnMsg_ind;

        EXEC SQL END DECLARE SECTION;

                /* create table */
             EXEC CREATE TABLE xml_order_tab (xmlorder XMLVarchar);

             /* read data from a file to a CLOB */
             strcpy(dadfile.name,"c:\dxx\samples\dad\getstart_xcollection.dad");
             dadfile.name_length = strlen("c:\dxx\samples\dad\getstart_xcollection.dad");
             dadfile.file_options = SQL_FILE_READ;
             EXEC SQL VALUES (:dadfile) INTO :dad;
                strcpy(result_tab,"xml_order_tab");
                override[0] = '\0';
                overrideType = NO_OVERRIDE;
                max_row = 500;
                num_row = 0;
                returnCode = 0;
                msg_txt[0] = '\0';
                collection_ind = 0;
             dad_ind = 0;
                rtab_ind = 0;
                ov_ind = -1;
                ovtype_ind = 0;
                maxrow_ind = 0;
                numrow_ind = -1;
                returnCode_ind = -1;
                returnMsg_ind = -1;

                /* Call the store procedure */
             EXEC SQL CALL dxxGenXML(:dad:dad_ind;
                        :result_tab:rtab_ind,
                        :overrideType:ovtype_ind,:override:ov_ind,
                        :max_row:maxrow_ind,:num_row:numrow_ind,
                        :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

La tabella dei risultati che deriva dal richiamo della procedura memorizzata contiene 250 righe in quanto l'interrogazione SQL specificata nel file DAD ha generato 250 documenti XML.

dxxRetrieveXML()

Questa procedura memorizzata viene utilizzata per le applicazioni che eseguono aggiornamenti regolari. Poiché vengono eseguite sempre le stesse attività, è importante che le prestazioni siano ottimali. L'abilitazione di una raccolta XML e l'utilizzo del nome di raccolta nella procedura memorizzata migliora le prestazioni.

La procedura memorizzata dxxRetrieveXML() corrisponde alla procedura memorizzata dxxGenXML(), l'unica differenza è che la prima rileva il nome di una raccolta XML abilitata anziché un file DAD. Quando si abilita una raccolta XML, un file DAD viene memorizzato nella tabella XML_USAGE, XML Extender richiama tale file e dxxRetrieveXML() prosegue come la procedura memorizzata dxxGenXML().

dxxRetrieveXML() consente l'utilizzo del file DAD sia nella composizione sia nella scomposizione. E' inoltre possibile utilizzare questa procedura memorizzata per il richiamo di documento XML scomposti.

La procedura memorizzata corrispondente per la scomposizione è dxxInsertXML() la quale rileva anche il nome di una raccolta XML abilitata.

Per comporre una raccolta XML: dxxRetrieveXML()

Comprende il richiamo di una procedura memorizzata utilizzando la seguente procedura:

```
dxxRetrieveXML(char(collectionName) collectionName,  /* input */
               char(resultTabName) resultTabName,   /* input */
               integer        overrideType,    /* input */
               varchar(1024)  override,        /* input */
               integer        maxRows,         /* input */
               integer        numRows,         /* output */
               long           returnCode,      /* output */
               varchar(1024)  returnMsg)       /* output */
```

Per la sintassi completa e gli esempi consultare la sezione "dxxRetrieveXML()" a pagina 186.

Esempio: di seguito è riportato un esempio di un richiamo di dxxRetrieveXML(). In questo esempio il nome della tabella dei risultati creata è XML_ORDER_TAB e la tabella presenta una colonna di tipo XMLVARCHAR.

```
#include "dxx.h" #include "dxxrc.h"

            EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
              char    collection[32];   /* dad buffer */
              char    result_tab[32];   /* name of the result table */
              char    override[2];      /* override, will set to NULL*/
              short   overrideType;     /* defined in dxx.h */
              short   max_row;          /* maximum number of rows */
              short   num_row;          /* actual number of rows */
              long    returnCode;       /* return error code */
              char    returnMsg[1024];  /* error message text */
              short   dadbuf_ind;
              short   rtab_ind;
              short   ovtype_ind;
              short   ov_inde;
              short   maxrow_ind;
              short   numrow_ind;
              short   returnCode_ind;
              short   returnMsg_ind;

EXEC SQL END DECLARE SECTION;

              /* create table */
              EXEC SQL CREATE TABLE xml_order_tab (xmlorder XMLVarchar);

              /* initialize host variable and indicators */
              strcpy(collection,"sales_ord");
              strcpy(result_tab,"xml_order_tab");
              override[0] = '\0';
              overrideType = NO_OVERRIDE;
              max_row = 500;
              num_row = 0;
              returnCode = 0;
              msg_txt[0] = '\0';
              collection_ind = 0;
              rtab_ind = 0;
              ov_ind = -1;
              ovtype_ind = 0;
              maxrow_ind = 0;
              numrow_ind = -1;
              returnCode_ind = -1;
              returnMsg_ind = -1;

              /* Call the store procedure */
```

```
EXEC SQL CALL dxxRetrieve(:collection:collection_ind;
       :result_tab:rtab_ind,
       :overrideType:ovtype_ind,:override:ov_ind,
       :max_row:maxrow_ind,:num_row:numrow_ind,
       :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

Sostituzione dinamica dei valori del file DAD

Nelle interrogazioni dinamiche è possibile utilizzare due parametri facoltativi per sostituire le condizioni del file DAD: *override* e *overrideType*. Sulla base del parametro di input *overrideType*, l'applicazione può sostituire i valori tag di <SQL_stmt> per l'associazione SQL oppure le condizioni di RDB_nodes per la relativa associazione nel file DAD.

Questi parametri presentano i seguenti valori e regole:

overrideType
Un parametro di input obbligatorio (IN) che indica il tipo del parametro *override*. *overrideType* presenta i seguenti valori:

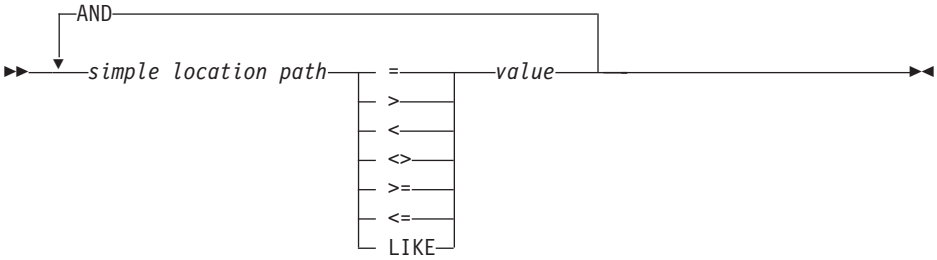
NO_OVERRIDE
Specifica di non sostituire una condizione del file DAD.

SQL_OVERRIDE
Specifica la sostituzione di una condizione del file DAD con un'istruzione SQL.

XML_OVERRIDE
Specifica la sostituzione di una condizione del file DAD con una condizione basata su XPath.

override
Questo parametro è un parametro di input facoltativo (IV) che specifica la condizione di sostituzione per il file DAD. La sintassi del valore di input corrisponde al valore specificato in *overrideType*.

- Se si specifica NO_OVERRIDE, il valore di input è una stringa NULL.
- Se si specifica SQL_OVERRIDE, il valore di input è un'istruzione SQL valida. ***Obbligatorio:*** Se si utilizza SQL_OVERRIDE e un'istruzione SQL, è necessario utilizzare lo schema di associazione SQL nel file DAD. L'istruzione SQL di input sostituisce l'istruzione SQL specificata dall'elemento <SQL_stmt> nel file DAD.
- Se si utilizza XML_OVERRIDE, il valore di input è una stringa che contiene una o più espressioni. ***Obbligatorio:*** Se si utilizza XML_OVERRIDE e un'espressione, è necessario utilizzare lo schema di associazione RDB_node nel file DAD. L'espressione XML di input sostituisce la condizione RDB_node specificata nel file DAD. L'espressione utilizza la seguente sintassi:



Dove:

simple location path
Un percorso di ubicazione semplice che utilizza la sintassi definita da XPATH; per la sintassi consultare Tabella 5 a pagina 45.

operatori
Per separare l'operatore dalle altre parti dell'espressione è possibile utilizzare uno spazio.

valore
Un valore numerico o una stringa tra apici singole.

Gli spazi accanto alle operazioni sono facoltativi; quelli accanto all'operatore LIKE sono obbligatori.

Quando si specifica il valore XML_OVERRIDE, la condizione relativa a RDB_node in test_node o attribute_node che corrisponde al percorso di ubicazione semplice viene sostituita dall'espressione specificata.

XML_OVERRIDE non è completamente compatibile con XPath. Il percorso di ubicazione semplice viene utilizzato solo per identificare l'elemento o l'attributo associato a una colonna.

Esempi:

I seguenti esempi mostrano la sostituzione dinamica mediante SQL_OVERRIDE e XML_OVERRIDE. La maggior parte degli esempi di procedura memorizzata illustrati in questa pubblicazione utilizzano NO_OVERRIDE.

Esempio: la procedura memorizzata che utilizza SQL_OVERRIDE.

```
include "dxx.h"
include "dxxrc.h"

            EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
              char    collection[32];   /* dad buffer */
              char    result_tab[32];   /* name of the result table */
          char    override[256];   /* override, SQL_stmt */
          short   overrideType;      /* defined in dxx.h */
              short   max_row;          /* maximum number of rows */
              short   num_row;          /* actual number of rows */
              long    returnCode;       /* return error code */
              char    returnMsg[1024];  /* error message text */
              short   rtab_ind;
              short   ovtype_ind;
              short   ov_inde;
              short   maxrow_ind;
              short   numrow_ind;
              short   returnCode_ind;
              short   returnMsg_ind;

EXEC SQL END DECLARE SECTION;

              /* create table */
     EXEC CREATE TABLE xml_order_tab (xmlorder XMLVarchar);

              /* initialize host variable and indicators */
              strcpy(collection,"sales_ord");
              strcpy(result_tab,"xml_order_tab");
          sprintf(override,"%s %s %s %s %s %s %s",
                    "SELECT o.order_key, customer, p.part_key, quantity, price,",
                    "tax, ship_id, date, mode ",
                    "FROM order_tab o, part_tab p,",
```

```
                "table(select substr(char(timestamp(generate_unique())),16",
                "as ship_id,date,mode from ship_tab)as s",
                "WHERE p.price > 50.00 and s.date >'1998-12-01' AND",
                "p.order_key = o.order_key and s.part_key = p.part_key");
        overrideType = SQL_OVERRIDE;
             max_row = 500;
             num_row = 0;
             returnCode = 0;
             msg_txt[0] = '\0';
             collection_ind = 0;
             rtab_ind = 0;
         ov_ind = 0;
             ovtype_ind = 0;
             maxrow_ind = 0;
             numrow_ind = -1;
             returnCode_ind = -1;
             returnMsg_ind = -1;

             /* Call the store procedure */
             EXEC SQL CALL dxxRetrieve(:collection:collection_ind;
                    :result_tab:rtab_ind,
               :overrideType:ovtype_ind,:override:ov_ind,
                    :max_row:maxrow_ind,:num_row:numrow_ind,
                    :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

In questo esempio, l'elemento <xcollection> nel file DAD deve presentare un elemento <SQL_stmt>. Il parametro *override* sostituisce il valore <SQL_stmt>, modificando il prezzo in un valore superiore a 50.00, la data viene modificata in un valore superiore a 1998-12-01.

Esempio: la procedura memorizzata che utilizza XML_OVERRIDE.

```
include "dxx.h"
include "dxxrc.h"

           EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
             char    collection[32];   /* dad buffer */
             char    result_tab[32];   /* name of the result table */
         char    override[256];   /* override, SQL_stmt */
         short   overrideType;    /* defined in dxx.h */
             short   max_row;          /* maximum number of rows */
             short   num_row;          /* actual number of rows */
             long    returnCode;       /* return error code */
             char    returnMsg[1024];  /* error message text */
             short   dadbuf_ind;
             short   rtab_ind;
             short   ovtype_ind;
             short   ov_inde;
             short   maxrow_ind;
             short   numrow_ind;
             short   returnCode_ind;
             short   returnMsg_ind;

EXEC SQL END DECLARE SECTION;

             /* create table */
     EXEC CREATE TABLE xml_order_tab (xmlorder XMLVarchar);

             /* initialize host variable and indicators */
             strcpy(collection,"sales_ord");
             strcpy(result_tab,"xml_order_tab");
         sprintf(override,"%s %s",
                "/Order/Part/Price > 50.00 AND ",
                "Order/Part/Shipment/ShipDate > '1998-12-01'");
        overrideType = XML_OVERRIDE;
```

```
        max_row = 500;
        num_row = 0;
        returnCode = 0;
        msg_txt[0] = '\0';
        collection_ind = 0;
        rtab_ind = 0;
    ov_ind = 0;
        ovtype_ind = 0;
        maxrow_ind = 0;
        numrow_ind = -1;
        returnCode_ind = -1;
        returnMsg_ind = -1;

        /* Call the store procedure */
        EXEC SQL CALL dxxRetrieve(:collection:collection_ind;
               :result_tab:rtab_ind,
          :overrideType:ovtype_ind,:override:ov_ind,
               :max_row:maxrow_ind,:num_row:numrow_ind,
               :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

In questo esempio, l'elemento <collection> nel file DAD presenta un RDB_node per element_node root. Il valore *override* si basa sul contenuto XML. XML Extender converte il percorso di ubicazione semplice nella colonna DB2 associata.

Scomposizione dei documenti XML in dati DB2

Scomporre un documento XML significa disgregare i dati all'interno di un documento XML e memorizzarli nelle tabelle relazionali. XML Extender fornisce le procedure memorizzate per scomporre i dati XML dei documenti XML di origine nelle tabelle relazionali. Per utilizzare queste procedure memorizzate, è necessario creare un file DAD che specifica l'associazione tra il documento XML e la struttura della tabella DB2. Le procedure memorizzate utilizzano il file DAD per scomporre il documento XML. Per informazioni sulla creazione di un file DAD, consultare la sezione "Pianificazione per le raccolte XML" a pagina 46.

Abilitazione di una raccolta XML per la scomposizione

Nella maggior parte dei casi, è necessario abilitare una raccolta XML prima di utilizzare le procedure memorizzate. Nei seguenti casi, è obbligatorio abilitare una raccolta XML:

- Quando si scompongono i documenti XML in nuove tabelle, è necessario abilitare una raccolta XML in modo che tutte le relative tabelle vengano create da XML Extender una volta abilitata la raccolta.
- Quando è rilevante conservare la sequenza degli elementi e degli attributi con ricorrenza multipla. XML Extender conserva solo l'ordine sequenziale degli elementi o degli attributi con ricorrenza multipla per le tabelle create durante l'abilitazione di una raccolta. Con la scomposizione dei documenti XML nelle tabelle relazionali esistenti, l'ordine della sequenza non è sicuro che vanga conservato.

Se si desidera che il file DAD venga trasferito automaticamente quando nel database esistono già delle tabelle, non è necessario abilitare la raccolta XML.

Limiti della scomposizione delle dimensioni della tabella

La scomposizione utilizza l'associazione RDB_node per specificare come un documento XML si scompone in tabelle DB2, estraendo i valori attributo ed

elemento nelle righe della tabella. I valori di ciascun documento XML vengono memorizzati in uno o più tabelle DB2. Una tabella può presentare massimo 1024 righe scomposte da ciascun documento.

Ad esempio, se un documento XML si scompone in cinque tabelle, ogni tabella può presentare massimo 1024 righe per quel determinato documento. Se la tabella presenta righe per più documenti, è possibile che contenga massimo 1024 righe per ciascun documento. Se la tabella presenta 20 documenti, è possibile che contenga 20.480 righe, 1024 per ciascun documento.

L'utilizzo di elementi con ricorrenza multipla (elementi con percorsi di ubicazione multipla nella struttura XML) incide sul numero di righe. Ad esempio, un documento che contiene un elemento <Part> che ricorre 20 volte, è necessario che sia scomposto in 20 righe in una tabella. Quando si utilizzano elementi con ricorrenza multipla, considerare questa restrizione delle dimensioni di tabella.

Informazioni preliminari

- Associare la struttura del documento XML alle tabelle relazionali che contengono il contenuto degli elementi e i valori dell'attributo.
- Preparare il file DAD, utilizzando l'associazione RDB_node. Per ulteriori dettagli, consultare "Pianificazione per le raccolte XML" a pagina 46.
- Facoltativamente, abilitare la raccolta XML.

Scomposizione del documento XML

XML Extender fornisce due procedure memorizzate, dxxShredXML() e dxxInsertXML,per la scomposizione di documenti XML.

dxxShredXML()

Questa procedura memorizzata viene utilizzata per le applicazioni che eseguono aggiornamenti occasionali o per le applicazioni che non sono in grado di eseguire la completa gestione dei dati XML. La procedura memorizzata dxxShredXML() non richiede una raccolta abilitata ma utilizza un file DAD.

La procedura memorizzata dxxShredXML() rileva due parametri di input, un file DAD e il documento XML da scomporre e restituisce due parametri di output: un codice di ritorno e un messaggio di errore.

La procedura memorizzata dxxShredXML() inserisce un documento XML in una raccolta XML a seconda della specifica <Xcollection> nel file DAD di input. Si presume che le tabelle utilizzate in <Xcollection> del file DAD esistano e che le colonne soddisfano i tipi di dati specificati nell'associazione DAD. In caso contrario, viene restituito un messaggio di errore. La procedura memorizzata dxxShredXML() quindi scompone il documento XML e inserisce i dati XML privi di tag nelle tabelle specificate nel file DAD.

La procedura memorizzata corrispondente per la composizione è dxxGenXML() la quale rileva anche il file DAD come parametro di input e non richiede che la raccolta XML sia abilitata.

Per scomporre una raccolta XML: dxxShredXML()

Comprende il richiamo di una procedura memorizzata utilizzando la seguente procedura:

```
dxxShredXML(CLOB(100K)    DAD,            /* input */
            CLOB(1M)       xmlobj,          /* input */
              long           returnCode,      /* output */
              varchar(1024)  returnMsg)       /* output */
```

Per la sintassi completa e gli esempi consultare la sezione "dxxShredXML()" a pagina 190.

Esempio: di seguito è riportato un esempio di un richiamo di dxxShredXML():

```
#include "dxx.h" #include "dxxrc.h"

           EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
             SQL TYPE is CLOB dad;           /* DAD*/
             SQL TYPE is CLOB_FILE  dadFile; /* DAD file*/
             SQL TYPE is CLOB xmlDoc;        /* input XML document */
             SQL TYPE is CLOB_FILE xmlFile;  /* input XMLfile */
             long                  returnCode;      /* error code */
             char    returnMsg[1024];  /* error message text */
     short                 dad_ind;
             short                 xmlDoc_ind;
             short   returnCode_ind;
             short   returnMsg_ind;
EXEC SQL END DECLARE SECTION;

             /* initialize host variable and indicators */
        strcpy(dadFile.name,"c:\dxx\samples\dad\getstart_xcollection.dad");
        dadFile.name_length=strlen("c:\dxx\samples\dad\getstart_xcollection.dad");
             dadFile.file_option=SQL_FILE_READ;
        strcpy(xmlFile.name,"c:\dxx\samples\cmd\getstart_xcollection.xml");
        xmlFile.name_length=strlen("c:\dxx\samples\cmd\getstart_xcollection.xml");
             xmlFile.file_option=SQL_FILE_READ;
             SQL EXEC VALUES (:dadFile) INTO :dad;
             SQL EXEC VALUES (:xmlFile) INTO :xmlDoc;
             returnCode = 0;
             returnMsg[0] = '\0';
     dad_ind = 0;
             xmlDoc_ind = 0;
             returnCode_ind = -1;
             returnMsg_ind = -1;

             /* Call the store procedure */
        EXEC SQL CALL db2xml.dxxShredXML(:dad:dad_ind;
                          :xmlDoc:xmlDoc_ind,
                  :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

dxxInsertXML()

Questa procedura memorizzata viene utilizzata per le applicazioni che eseguono aggiornamenti regolari. Poiché vengono eseguite sempre le stesse attività, è importante che le prestazioni siano ottimali. L'abilitazione di una raccolta XML e l'utilizzo del nome di raccolta nella procedura memorizzata migliora le prestazioni. La procedura memorizzata dxxInsertXML() corrisponde alla procedura memorizzata dxxShredXML(), l'unica differenza è che la prima rileva una raccolta XML abilitata come primo parametro di input.

La procedura memorizzata dxxInsertXML() inserisce un documento XML in una raccolta XML abilitata, associata a un file DAD. Il file DAD contiene le specifiche relative alle tabelle della raccolta e all'associazione. Le tabelle della raccolta vengono selezionate o create in base alle specifiche contenute in <Xcollection>. La procedura memorizzata dxxInsertXML() quindi

scompone il documento XML in base all'associazione e inserisce i dati XML privi di tag nelle tabelle della raccolta XML.

La procedura memorizzata corrispondente per la composizione è dxxRetrieveXML() la quale rileva il nome di una raccolta XML abilitata.

Per scomporre una raccolta XML: dxxInsertXML()

Comprende il richiamo di una procedura memorizzata utilizzando la seguente procedura:

```
dxxInsertXML(char(collectionName) collectionName,   /* input */
             CLOB(1M)       xmlobj,           /* input */
               long           returnCode,      /* output */
               varchar(1024)  returnMsg)       /* output */
```

Per la sintassi completa e gli esempi consultare la sezione "dxxInsertXML()" a pagina 192.

Esempio: di seguito è riportato un esempio di richiamo di dxxInsertXML():

```
#include "dxx.h" #include "dxxrc.h"

            EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
              char                  collection[64];  /* name of an XML collection */
              SQL TYPE is CLOB_FILE xmlFile;  /* input XMLfile */
         SQL TYPE is CLOB      xmlDoc;           /* input XML doc */
              long                  returnCode;      /* error code */
              char    returnMsg[1024];  /* error message text */
              short                 collection_ind;
              short                 xmlDoc_ind;
              short   returnCode_ind;
              short   returnMsg_ind;
EXEC SQL END DECLARE SECTION;

              /* initialize host variable and indicators */
              strcpy(collection,"sales_ord")
         strcpy(xmlobj.name,"c:\dxx\samples\cmd\getstart_xcollection.xml");
         xmlobj.name_length=strlen("c:\dxx\samples\cmd\getstart_xcollection.xml");
              xmlobj.file_option=SQL_FILE_READ;
         SQL EXEC VALUES (:xmlFile) INTO (:xmlDoc);
              returnCode = 0;
              returnMsg[0] = '\0';
              collection_ind = 0;
              xmlobj_ind = 0;
              returnCode_ind = -1;
              returnMsg_ind = -1;

              /* Call the store procedure */
         EXEC SQL CALL db2xml.dxxInsertXML(:collection:collection_ind;
                              :xmlDoc:xmlDoc_ind,
                    :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

Accesso a una raccolta XML

E' possibile aggiornare, cancellare, ricercare e richiamare le raccolte XML. Tuttavia, tenere presente che lo scopo dell'utilizzo di una raccolta XML è quello di memorizzare o richiamare i dati, privi di tag, contenuti nelle tabelle del database. I dati contenuti nelle tabelle esistenti del database sono elementi differenti dai documenti XML in entrata; le operazioni di aggiornamento, di cancellazione e di ricerca si basano sull'accesso SQL regolare a questo tabelle. Se i dati derivano dalla scomposizione dei documenti XML in entrata, nessun documento XML esisterà più.

XML Extender consente di eseguire operazioni sui dati da una vista della raccolta XML. Utilizzando le istruzioni UPDATE e DELETE, è possibile modificare i dati utilizzati per la composizione dei documenti XML, e quindi, aggiornare la raccolta XML.

Considerazioni:

- Per aggiornare un documento, non cancellare una riga contenente la chiave primaria della tabella che rappresenta la riga della chiave esterna delle altre tabelle della raccolta. Quando si cancella la riga della chiave primaria e della chiave esterna, viene cancellato anche il documento.
- Per sostituire o cancellare gli elementi e i valori dell'attributo, è possibile cancellare e inserire righe nelle tabelle di livello inferiore senza cancellare il documento.
- Per cancellare un documento, cancellare la riga che compone il primo element_node specificato nel file DAD.

Aggiornamento dei dati in una raccolta XML

XML Extender consente di aggiornare i dati privi di tag memorizzati nelle tabelle della raccolta XML. L'aggiornamento dei valori della tabella della raccolta XML aggiorna anche il testo di un elemento XML o il valore di un attributo XML. E' inoltre possibile che gli aggiornamenti eseguano la cancellazione di un'istanza di dati da attributi o elementi con ricorrenza multipla.

Da un punto di vista SQL, la modifica del valore dell'elemento o dell'attributo è un'operazione di aggiornamento e la cancellazione di un'istanza di un elemento o un attributo è un'operazione di cancellazione. Dal punto di vista XML, finché esiste il testo dell'elemento o il valore dell'attributo di element_node root, esiste anche il documento XML che rappresenta, quindi, un'operazione di aggiornamento.

Requisiti: per aggiornare i dati in una raccolta XML, rispettare le seguente regole.

- Specificare la relazione chiave primaria-esterna tra le tabelle della raccolta se le tabelle esistenti presentano questo relazione. In caso contrario, accertarsi che siano presenti colonne che è possibile unire.
- Includere la condizione di collegamento specificata nel file DAD:
  - Per l'associazione SQL, nell'elemento <SQL_stmt>
  - Per l'associazione RDB_node, in RDB_node del nodo elemento root

Aggiornamento dei valori dell'attributo o dell'elemento

In una raccolta XML, il testo dell'elemento e il valore dell'attributo vengono associati nelle colonne nelle tabelle del database. Senza tener conto se i dati della colonna esistevano precedentemente o se derivano dalla scomposizione dei documenti XML in entrata, sostituire questi dati utilizzando la tecnica di aggiornamento SQL regolare.

Per aggiornare un valore dell'elemento o dell'attributo, specificare una clausola WHERE nell'istruzione SQL UPDATE che contiene la condizione di collegamento specificata nel file DAD.

Ad esempio:

```
UPDATE SHIP_TAB
    set MODE = 'BOAT'
  WHERE MODE='AIR' AND PART_KEY in
    (SELECT PART_KEY from PART_TAB WHERE ORDER_KEY=68)
```

Il valore dell'elemento <ShipMode> viene aggiornato da AIR in BOAT nella tabella SHIP_TAB, dove la chiave è 68.

Cancellazione delle istanze di elemento e di attributo

Per aggiornare i documenti XML composti eliminando gli elementi o gli attributi a ricorrenza multipla, cancellare una riga contenente il valore del campo che corrisponde al valore dell'attributo o dell'elemento, utilizzando la clausola Where. Finché non si cancella la riga che contiene i valori del primo element_node, la cancellazione dei valori dell'elemento è considerata un'aggiornamento del documento XML.

Ad esempio, nella seguente istruzione DELETE, si esegue la cancellazione di un elemento <shipment> specificando un valore univoco di uno dei relativi elementi secondari.

```
DELETE from SHIP_TAB
   WHERE DATE='1999-04-12'
```

Se si specifica un valore DATE viene cancellata la riga che corrisponde a questo valore. Il documento composto originariamente conteneva due elementi <shipment>, ora ne contiene uno.

Cancellazione di un documento XML da una raccolta XML

E' possibile cancellare un documento XML composto da una raccolta. Ciò significa che se si dispone di una raccolta XML che compone più documenti XML, è possibile cancellare uno di questi documenti composti.

Per cancellare il documento, cancellare una riga della tabella che compone il primo element_node specificato nel file DAD. Questa tabella contiene la chiave primarie relativa alla tabella della raccolta di livello superiore e la chiave esterna relative alle tabelle di livello inferiore.

Ad esempio la seguente istruzione DELETE specifica il valore della colonna della chiave primaria.

```
DELETE from order_tab
   WHERE order_key=1
```

ORDER_KEY è la chiave primaria della tabella ORDER_TAB e il primo element_node quando viene composto il documento XML. La cancellazione di questa riga cancella un documento XML generato durante la composizione. Quindi, dal punto di vista XML, un documento XML viene cancellato dalla raccolta XML.

Richiamo dei documenti XML da una raccolta XML

Il richiamo dei documenti XML da una raccolta XML è simile alla composizione di documenti dalla raccolta.

Per richiamare documenti XML, utilizzare la procedura memorizzata, dxxRetrieveXML(). Per la sintassi e gli esempi, consultare la sezione "dxxRetrieveXML()" a pagina 186.

Considerazioni sul file DAD: quando si scompongono i documenti XML in una raccolta XML, è possibile perdere l'ordine dei valori degli attributi e degli elementi a ricorrenza multipla, a mano che non si specifichi l'ordine nel file DAD. Per conservare questo ordine, utilizzare lo schema dell'associazione RDB_node. Questo

schema dell'associazione consente di specificare un attributo orderBy per la tabella contenente l'elemento root nel relativo RDB_node.

Ricerca di una raccolta XML

Questa sezione descrive la ricerca di una raccolta XML per le seguenti attività:

- **Creazione di documenti XML utilizzando criteri di ricerca:**

  Questa attività rappresenta una composizione che utilizza una condizione. E' possibile specificare i criteri di ricerca utilizzando quelli riportati di seguito:

  – Specificare la condizione in text_node e attribute_node del file DAD

  – Specificare il parametri *overwrite* quando si utilizzano le procedure memorizzate dxxGenXML() e dxxRetrieveXML().

  Ad esempio, se è stata abilitata una raccolta XML, sales_ord, utilizzando il file DAD, `order.dad` e si desidera sostituire il prezzo utilizzando i dati che derivano dal Web, è possibile sostituire il valore dell'elemento DAD <SQL_stmt> come segue:

```
            EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
       ...

EXEC SQL END DECLARE SECTION;

         float    price_value;

              /* create table */
              EXEC SQL CREATE TABLE xml_order_tab (xmlorder XMLVarchar);

              /* initialize host variable and indicators */
              strcpy(collection,"sales_ord");
              strcpy(result_tab,"xml_order_tab");
         overrideType = SQL_OVERRIDE;
         max_row = 20;
              num_row = 0;
              returnCode = 0;
              msg_txt[0] = '\0';
         override_ind = 0;
         overrideType_ind = 0;
              rtab_ind = 0;
              maxrow_ind = 0;
              numrow_ind = -1;
              returnCode_ind = -1;
              returnMsg_ind = -1;

         /* get the price_value from some place, such as form data */
         price_value = 1000.00         /* for example*/

         /* specify the overwrite */
         sprintf(overwrite,
                "SELECT o.order_key, customer, p.part_key, quantity, price,
                        tax, ship_id, date, mode
         FROM order_tab o, part_tab p,
table (select substr(char(timestamp(generate_unique())),16)
                     as ship_id, date, mode from ship_tab)as s
                   WHERE p.price > %d and s.date >'1996-06-01' AND
                       p.order_key = o.order_key and s.part_key = p.part_key",
                       price_value);

              /* Call the store procedure */
        EXEC SQL CALL db2xml.dxxRetrieve(:collection:collection_ind,
                     :result_tab:rtab_ind,
```

```
:overrideType:overrideType_ind,:overwrite:overwrite_ind,
     :max_row:maxrow_ind,:num_row:numrow_ind,
     :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

La condizione prezzo > 2500.00 in `order.dad` viene sostituita da > ?, dove ? si basa sulla variabile di input *price_value*.

- **Ricerca dei dati XML scomposti:**

  E' possibile utilizzare operazioni di interrogazione SQL regolari per ricercare le tabelle della raccolta. E' possibile unire le tabelle della raccolta oppure utilizzare le interrogazioni secondarie e quindi effettuare ricerche di testo strutturale nelle colonne di testo. Con i risultati della ricerca strutturale, è possibile utilizzare questi dati per richiamare o creare il documento XML specificato.

Parte 4. Riferimento

Questa sezione fornisce le informazioni sulla sintassi relativa al comando di gestione XML Extender, agli UDT (user-defined data types), alle UDF (user-defined functions) e alle procedure memorizzate. Vengono inoltre forniti messaggi per le attività di individuazione problemi.

Capitolo 7. Comando di gestione XML Extender: dxxadm

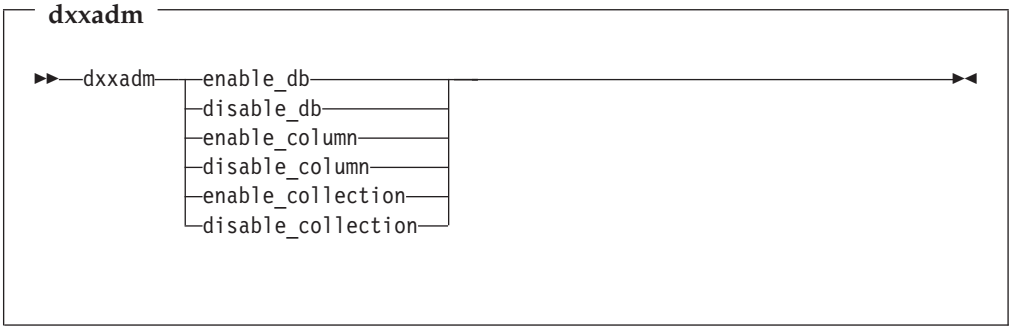
XML Extender fornisce un comando di gestione, **dxxadm**, per completare le seguenti attività di gestione:E' possibile effettuare le seguenti attività di gestione XML Extender richiamando **dxxadm** con differenti opzioni di comando:

- Abilitazione o disabilitazione di un database per XML Extender
- Abilitazione o disabilitazione di una colonna XML
- Abilitazione o disabilitazione di una raccolta XML

E' inoltre possibile utilizzare le procedure memorizzate o il wizard di gestione XML Extender per eseguire le attività di gestione.

Sintassi di alto livello

Il seguente diagramma di sintassi fornisce la sintassi di alto livello del comando **dxxadm**. Nelle seguenti sezioni vengono fornite le descrizioni di ciascuna opzione.



Opzioni del comando di gestione

Le seguenti sezioni descrivono ciascuna opzione di comando di **dxxadm** disponibili per i programmatori di sistema:

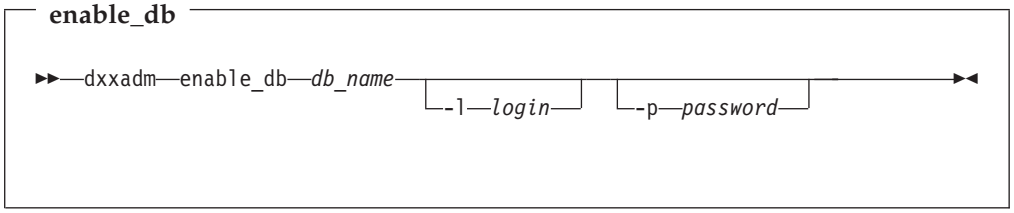
enable_db

Scopo

Stabilisce un collegamento e abilita un database per l'utilizzo con XML Extender. Una volta abilitato il database, XML Extender crea i seguenti oggetti:

- Gli UDT (tipi definiti dall'utente) XML Extender.
- Gli UDF (funzioni definite dall'utente) XML Extender.
- La tabella di riferimento DTD di XML Extender, DTD_REF, che memorizza i DTD e le relative informazioni. Per una completa descrizione della tabella DTD_REF, consultare la sezione "Tabella di riferimento DTD" a pagina 193.
- La tabella di utilizzo XML Extender, XML_USAGE, che memorizza le informazioni comuni per ciascuna colonna abilitata per XML e per ciascuna raccolta. Per una completa descrizione della tabella XML_USAGE, consultare la sezione "Tabella di utilizza XML" a pagina 193.

Formato



Parametri

Tabella 14. Parametri di enable_db

Parametro	Descrizione
db_name	Il nome del database in cui sono ubicati i dati XML.
-l *login*	L'ID utente, che è facoltativo, viene utilizzato per collegarsi al database, quando specificato. Se non viene specificato, viene utilizzato quello corrente.
-p *password*	La parola d'ordine, che è facoltativa, viene utilizzata per collegarsi al database, quando specificato. Se non viene specificata, viene utilizzata quella corrente.

Esempi

Il seguente esempio abilita il database SALES_DB.

```
dxxadm enable_db SALES_DB
```

disable_db

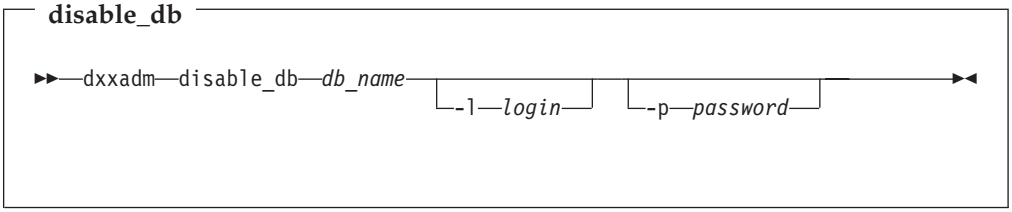
Scopo

Stabilisce un collegamento e disabilita il database abilitato per XML. Il database, una volta disabilitato, non può più essere utilizzato da XML Extender. Quando XML Extender disabilita il database, vengono cancellati i seguenti oggetti:

- Gli UDT (tipi definiti dall'utente) XML Extender.
- Gli UDF (funzioni definite dall'utente) XML Extender.
- La tabella di riferimento DTD di XML Extender, DTD_REF, che memorizza i DTD e le relative informazioni. Per una completa descrizione della tabella DTD_REF, consultare la sezione "Tabella di riferimento DTD" a pagina 193.
- La tabella di utilizzo XML Extender, XML_USAGE, che memorizza le informazioni comuni per ciascuna colonna abilitata per XML e per ciascuna raccolta. Per una completa descrizione della tabella XML_USAGE, consultare la sezione "Tabella di utilizza XML" a pagina 193.

Importante: è necessario disabilitare tutte le colonne XML prima di disabilitare un database. XML Extender non può disabilitare un database che contiene colonne o raccolte abilitate per XML.

Formato



Parametri

Tabella 15. Parametri di disable_db

Parametro	Descrizione
db_name	Il nome del database in cui sono ubicati i dati XML
-l *login*	L'ID utente, che è facoltativo, viene utilizzato per collegarsi al database, quando specificato. Se non viene specificato, viene utilizzato quello corrente.
-p *password*	La parola d'ordine, che è facoltativa, viene utilizzata per collegarsi al database, quando specificato. Se non viene specificata, viene utilizzata quella corrente.

Esempi

Il seguente esempio disabilita il database SALES_DB.

```
dxxadm disable_db SALES_DB
```

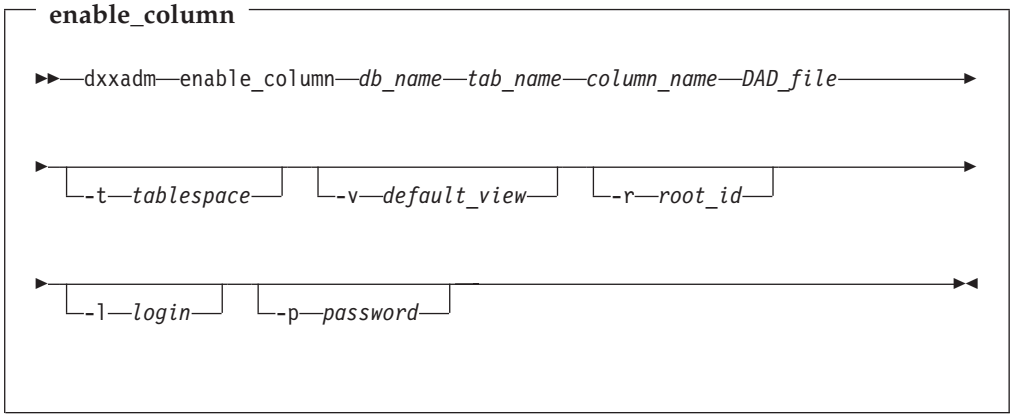
enable_column

Scopo

Stabilisce un collegamento con un database e abilita una colonna XML in modo che possa contenere gli UDT XML Extender. Quando si abilita una colonna, XML Extender completa le seguenti attività:

- Determina se la tabella XML presenta una chiave primaria; in caso contrario, XML Extender modifica la tabella XML e aggiunge una colonna denominata DXXROOT_ID.
- Crea le tabelle laterali specificate nel file DAD con una colonna contenente un identificativo univoco per ciascuna riga della tabella XML. Questa colonna è il parametro *root_id* specificato dall'utente o il parametro DXXROOT_ID specificato da XML Extender.
- Crea una vista predefinita per la tabella XML e le relative tabelle laterali, utilizzando, facoltativamente, un nome specificato dall'utente.

Formato



Parametri

Tabella 16. Parametri di enable_column

Parametro	Descrizione
db_name	Il nome del database in cui sono ubicati i dati XML.
tab_name	Il nome della tabella in cui sono ubicate le colonne XML.
column_name	Il nome della colonna XML.
DAD_file	Il nome del file DAD che associa il documento XML alle tabelle laterali e alla colonna XML.
-t *tablespace*	Il table space è facoltativo e contiene le tabelle laterali associate alla colonna XML. Se non viene specificato, viene utilizzato il table space predefinito.
-v *default_view*	Il nome della vista predefinita, facoltativa, che unisce le tabelle laterali e la colonna XML.

Tabella 16. Parametri di enable_column (Continua)

Parametro	Descrizione
-r *root_id*	Il nome della chiave primaria nella tabella della colonna da utilizzare come root_id per le tabelle laterali. Il parametro root_id è facoltativo.
-l *login*	L'ID utente, che è facoltativo, viene utilizzato per collegarsi al database, quando specificato. Se non viene specificato, viene utilizzato quello corrente.
-p *password*	La parola d'ordine, che è facoltativa, viene utilizzata per collegarsi al database, quando specificato. Se non viene specificata, viene utilizzata quella corrente.

Esempi

Il seguente esempio abilita una colonna XML.

```
dxxadm enable_column SALES_DB SALES_TAB ORDER -v sales_order_view -r INVOICE_NUMBER
```

disable_column

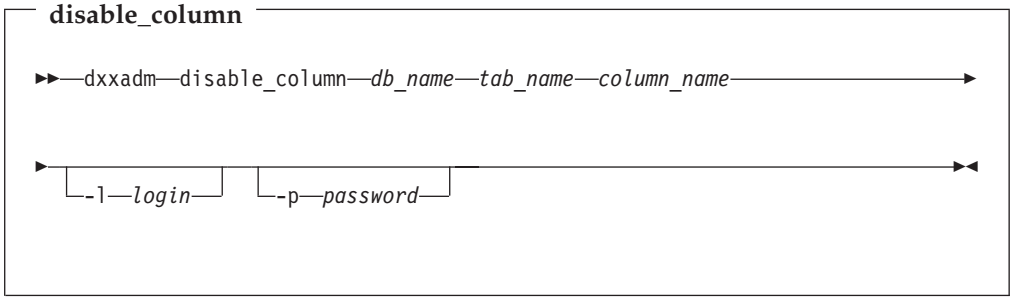
Scopo

Stabilisce un collegamento con un database e disabilita la colonna abilitata per XML. La colonna, una volta disabilitata, non può più contenere i tipi di dati XML. Quando si disabilita una colonna abilitata per XML, vengono effettuate le seguenti azioni:

- La voce relativa all'utilizzo della colonna XML viene cancellata dalla tabella XML_USAGE.
- Il valore USAGE_COUNT nella tabella DTD_REF si riduce.
- Tutti i trigger associati alla colonna vengono cancellati.
- Tutte le tabelle laterali associate alla colonna vengono cancellate.

Importante: è necessario disabilitare una colonna XML prima di cancellare una tabella XML. Se si cancella una tabella XML è la relativa colonna XML non è disabilitata, XML Extender conserva le tabelle laterali create e la voce colonna XML nella tabella XML_USAGE.

Formato



Parametri

Tabella 17. Parametri di disable_column

Parametro	Descrizione
db_name	Il nome del database in cui sono ubicati i dati.
tab_name	Il nome della tabella in cui sono ubicate le colonne XML.
column_name	Il nome della colonna XML.
-l *login*	L'ID utente, che è facoltativo, viene utilizzato per collegarsi al database, quando specificato. Se non viene specificato, viene utilizzato quello corrente.
-p *password*	La parola d'ordine, che è facoltativa, viene utilizzata per collegarsi al database, quando specificato. Se non viene specificata, viene utilizzata quella corrente.

Esempi

Il seguente esempio disabilita una colonna abilitata per XML.

```
dxxadm disable_column SALES_DB SALES_TAB ORDER
```

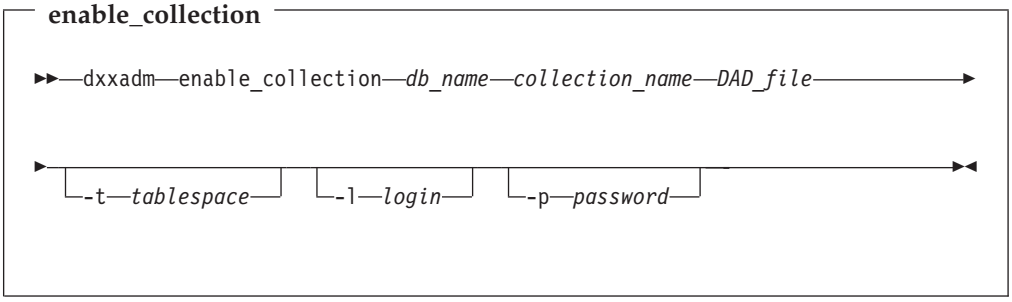
enable_collection

Scopo

Stabilisce un collegamento con un database e abilita una raccolta XML in base al parametro DAD specificato. Quando si abilita una raccolta, XML Extender effettua le seguenti attività:

- Crea una voce di utilizzo raccolta XML nella tabella XML_USAGE.
- Per l'associazione RDB_node, crea le tabelle della raccolta specificate con il parametro DAD se le tabelle non esistono nel database.

Formato



Parametri

Tabella 18. Parametri enable_collection

Parametro	Descrizione
db_name	Il nome del database in cui sono ubicati i dati.
collection_name	Il nome della raccolta XML.
DAD_file	Il nome del file DAD che associa il documento XML alle tabelle relazionali della raccolta.
-t *tablespace*	Il nome del table space, che è facoltativo ed è associato alla raccolta. Se non viene specificato, viene utilizzato il table space predefinito.
-l *login*	L'ID utente, che è facoltativo, viene utilizzato per collegarsi al database, quando specificato. Se non viene specificato, viene utilizzato quello corrente.
-p *password*	La parola d'ordine, che è facoltativa, viene utilizzata per collegarsi al database, quando specificato. Se non viene specificata, viene utilizzata quella corrente.

Esempi

Il seguente esempio abilita una raccolta XML.

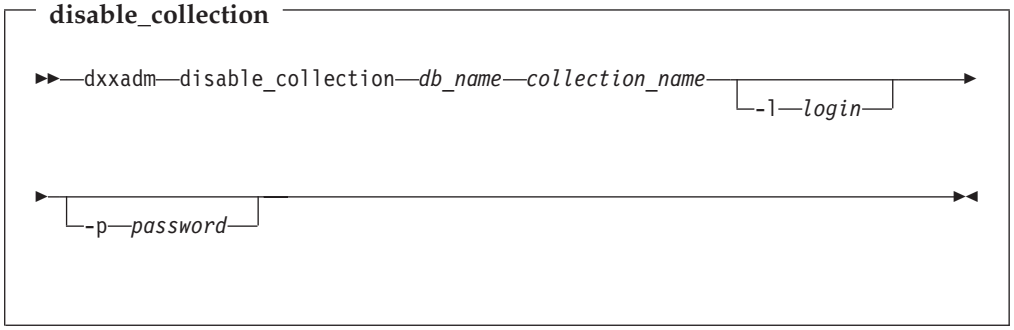
```
dxxadm enable_collection SALES_DB sales_ord getstart_xcollection.dad -t orderspace
```

disable_collection

Scopo

Stabilisce un collegamento con un database e disabilita la raccolta abilitata per XML. Non è più possibile utilizzare il nome della raccolta nelle procedure memorizzate per la composizione (dxxRetrieveXML) e la scomposizione (dxxInsertXML). Quando si disabilita una raccolta XML, la voce della raccolta associata viene cancellata dalla tabella XML_USAGE. Tenere presente che la disabilitazione della raccolta non cancella le tabelle della raccolta create durante la fase di abilitazione raccolta.

Formato



Parametri

Tabella 19. Parametri di disable_collection

Parametro	Descrizione
db_name	Il nome del database in cui sono ubicati i dati.
collection_name	Il nome della raccolta XML.
-l *login*	L'ID utente, che è facoltativo, viene utilizzato per collegarsi al database, quando specificato. Se non viene specificato, viene utilizzato quello corrente.
-p *password*	La parola d'ordine, che è facoltativa, viene utilizzata per collegarsi al database, quando specificato. Se non viene specificata, viene utilizzata quella corrente.

Esempi

Il seguente esempio disabilita una raccolta XML.

```
dxxadm disable_collection SALES_DB sales_ord
```

Capitolo 8. XML Extender - UDT

Gli UDT (tipi definiti dall'utente) XML Extender sono tipi di dati utilizzati per le colonne e le raccolte XML. Il nome dello schema di tutti gli UDT è db2xml. XML Extender crea gli UDT per memorizzare e richiamare i documenti XML. La Tabella 20 contiene una panoramica degli UDT.

Tabella 20. UDT XML Extender

Colonna UDT	Tipo di dati origine	Descrizione
XMLVARCHAR	VARCHAR(*varchar_len*)	Memorizza un intero documento XML come VARCHAR nel DB2.
XMLCLOB	CLOB(*clob_len*)	Memorizza un intero documento XML come CLOB nel DB2.
XMLFILE	VARCHAR(512)	Specifica il nome del file server locale. Se viene specificato XMLFILE per la colonna XML, XML Extender memorizza il documento XML in un server file esterno. Text Extender non può essere abilitato con XMLFILE. L'utente è responsabile della congruenza, tra DB2 e il contenuto dei file, e delle tabelle laterali create per gli indici.

varchar_len e *clob_len* variano in base al sistema operativo.

Per DB2 UDB, *varchar_len* = 3K e *clob_len* = 2G.

Questi UDT vengono utilizzati solo per specificare le colonne di applicazione; non sono validi per le tabelle laterali create da XML Extender.

Capitolo 9. XML Extender - UDF

XML Extender fornisce le funzioni per la memorizzazione, la ricerca e l'aggiornamento dei documenti XML e per l'estrazione di elementi o attributi XML. Utilizzare le funzioni XML definite dall'utente (UDF) per le colonne XML e non per le raccolte XML. Tutte le funzioni UDF presentano il nome di schema db2xml, che è possibile omettere.

I quattro tipi di funzione XML Extender sono: le funzioni di memorizzazione, di richiamo, di estrazione e una funzione di aggiornamento.

funzioni di memorizzazione
Le funzioni di memorizzazione inseriscono i documenti XML in un database DB2. Per la sintassi e gli esempi, consultare la sezione "Funzioni di memorizzazione" a pagina 142.

funzioni di richiamo
Le funzioni di richiamo consentono il richiamo dei documenti XML dalle colonne XML di un database DB2. Per la sintassi e gli esempi, consultare la sezione "Funzioni di richiamo" a pagina 147.

funzioni di estrazione
Le funzioni di estrazione consentono l'estrazione e la conversione del contenuto dell'elemento o del valore attributo di un documento XML nel tipo di dati specificato dal nome della funzione. XML Extender fornisce una serie di funzioni di estrazione per vari tipi di dati SQL. Per la sintassi e gli esempi, consultare la sezione "Funzioni di estrazione" a pagina 152.

funzione di aggiornamento
La funzione Update() modifica il contenuto dell'elemento o il valore attributo e restituisce una copia di un documento XML con un valore aggiornato specificato dal percorso di ubicazione. La funzione Update() consente al programmatore dell'applicazione di specificare l'elemento o l'attributo da aggiornare. Per la sintassi e gli esempi, consultare la sezione "Funzione di aggiornamento" a pagina 166.

Tabella 21 fornisce un riepilogo delle funzioni XML Extender.

Tabella 21. Le funzioni UDF XML Extender

Tipo	Funzione
Funzioni di memorizzazione	XMLVarcharFromFile()
	XMLCLOBFromFile()
	XMLFileFromVarchar()
	XMLFileFromCLOB()
Funzioni di richiamo	Content(): richiamo da XMLFile in un CLOB
	Content(): richiamo da XMLVarchar in un server file esterno
	Content(): richiamo da XMLCLOB in un server file esterno

Tabella 21. Le funzioni UDF XML Extender (Continua)

Tipo	Funzione
Funzioni di estrazione	extractInteger() ed extractIntegers()
	extractSmallint() ed extractSmallints()
	extractDouble() ed extractDoubles()
	extractReal() ed extractReals()
	extractChar() ed extractChars()
	extractVarchar() ed extractVarchars()
	extractCLOB() ed extractCLOBs()
	extractDate() ed extractDates()
	extractTime() ed extractTimes()
	extractTimestamp() ed extractTimestamps()
Funzione di aggiornamento	Update()

Quando si utilizzano indicatori di parametro in UDF, una limitazione JDBC richiede che l'indicatore per UDF sia collegato al tipo di dati della colonna in cui verranno inseriti i dati restituiti. Per informazioni sul collegamento degli indicatori di parametro, consultare la sezione "Limitazioni durante il richiamo di funzioni da JDBC" a pagina 110.

Funzioni di memorizzazione

Utilizzare le funzioni di memorizzazione per inserire documenti XML in un database DB2. E' possibile utilizzare le funzioni cast predefinite di un UDT direttamente nelle istruzioni INSERT o SELECT, come descritto in "Memorizzazione dati" a pagina 98. Inoltre, XML Extender fornisce le UDF per accedere ai documenti XML dalle origini anziché del tipo di dati di base UDT e per convertire tali documenti nell'UDT desiderato.

XML Extender fornisce le seguenti funzioni di memorizzazione:

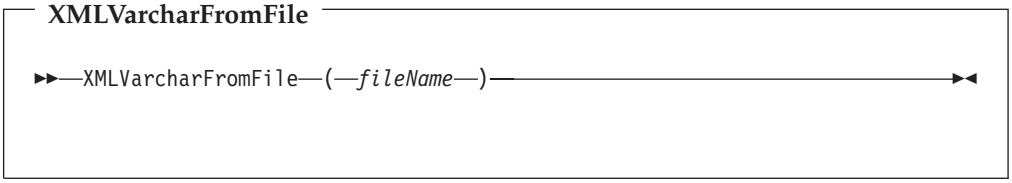
- "XMLVarcharFromFile()" a pagina 143
- "XMLCLOBFromFile()" a pagina 144
- "XMLFileFromVarchar()" a pagina 145
- "XMLFileFromCLOB()" a pagina 146

XMLVarcharFromFile()

Scopo

Legge un documento XML da un server file e restituisce il documento come tipo XMLVARCHAR.

Sintassi



Parametri

Tabella 22. parametro XMLVarcharFromFile

Parametro	Tipo di dati	Descrizione
fileName	VARCHAR(512)	Il nome completo del server file.

Tipo restituzione

XMLVARCHAR

Esempio

Di seguito è riportato un esempio di lettura di un documento XML da un server file e del relativo inserimento in una colonna XML come tipo XMLVARCHAR.

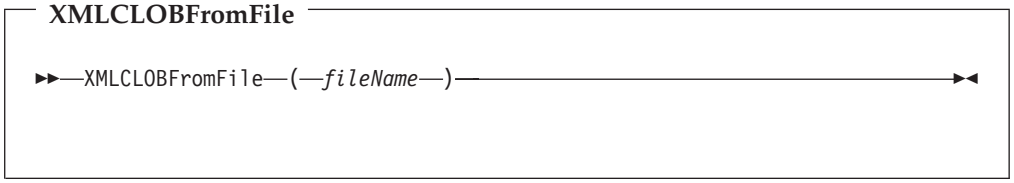
```
EXEC SQL INSERT INTO sales_tab(ID, NAME, ORDER)
   VALUES('1234', 'Sriram Srinivasan',
           XMLVarcharFromFile('c:\dxx\samples\cmd\getstart.xml'))
```

In questo esempio, un record viene inserito nella tabella SALES_TAB. La funzione XMLVarcharFromFile() importa il documento XML da un file in DB2 e lo memorizza come XMLVARCHAR.

XMLCLOBFromFile()

Scopo
Legge un documento XML da un server file e lo restituisce come tipo XMLCLOB.

Sintassi



Parametri

Tabella 23. parametro XMLCLOBFromFile

Parametro	Tipo di dati	Descrizione
fileName	VARCHAR(512)	Il nome completo del server file.

Tipo restituzione
XMLCLOB come LOCATOR

Esempio
Di seguito è riportato un esempio di lettura di un documento XML da un server file e del relativo inserimento in una colonna XML come tipo XMLCLOB.

```
EXEC SQL INSERT INTO sales_tab(ID, NAME, ORDER)
   VALUES('1234', 'Sriram Srinivasan',
          XMLCLOBFromFile('c:\dxx\samples\cmd\getstart.xml'))
```

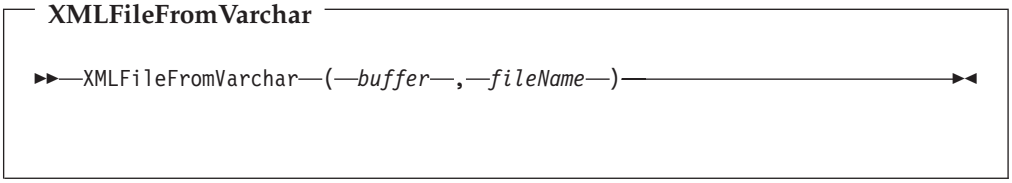
La colonna ORDER nella tabella SALES_TAB viene definita come tipo XMLCLOB. L'esempio precedente illustra come la colonna ORDER viene inserita nella tabella SALES_TAB.

XMLFileFromVarchar()

Scopo

Legge un documento XML dalla memoria come VARCHAR, lo scrive in un server file esterno e restituisce il percorso e nome file come tipo XMLFILE.

Sintassi



Parametri

Tabella 24. Parametri XMLFileFromVarchar

Parametro	Tipo di dati	Descrizione
buffer	VARCHAR(3K)	Il buffer della memoria.
fileName	VARCHAR(512)	Il nome completo del server file.

Tipo restituzione

XMLFILE

Esempio

Di seguito sono riportati esempi di lettura di un documento XML dalla memoria come VARCHAR, la relativa scrittura in un server file esterno e dell'inserimento del percorso e nome file come tipo XMLFILE in una colonna XML.

```
EXEC SQL BEGIN DECLARE SECTION;
      struct { short len; char data[3000]; } xml_buff;
EXEC SQL END DECLARE SECTION;

EXEC SQL INSERT INTO sales_tab(ID, NAME, ORDER)
  VALUES('1234', 'Sriram Srinivasan',
       XMLFileFromVarchar(:xml_buf, 'c:\dxx\samples\cmd\getstart.xml'))
```

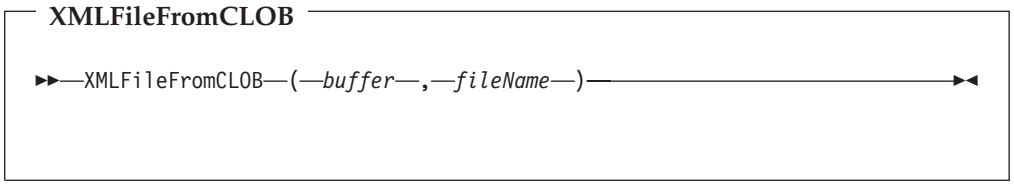
La colonna ORDER nella tabella SALES_TAB viene definita come tipo XMLFILE. L'esempio precedente illustra che se si dispone di un documento XML nel buffer, è possibile memorizzarlo in un server file.

XMLFileFromCLOB()

Scopo

Legge un documento XML come indicatore di posizione CLOB, lo scrive in un server file esterno e restituisce il percorso e nome file come tipo XMLFILE.

Sintassi



Parametri

Tabella 25. Parametri XMLFileFromCLOB()

Parametri	Tipo di dati	Descrizione
buffer	CLOB come LOCATOR	Il buffer che contiene il documento XML.
fileName	VARCHAR(512)	Il nome completo del server file.

Tipo restituzione

XMLFILE

Esempio

Di seguito sono riportati esempi di lettura di un documento XML come indicatore di posizione CLOB, la relativa scrittura in un server file esterno e dell'inserimento del percorso e nome file come tipo XMLFILE in una colonna XML.

```
EXEC SQL BEGIN DECLARE SECTION;
     SQL TYPE IS CLOB_LOCATOR xml_buff;
EXEC SQL END DECLARE SECTION;

EXEC SQL INSERT INTO sales_tab(ID, NAME, ORDER)
   VALUES('1234', 'Sriram Srinivasan',
           XMLFileFromCLOB(:xml_buf, 'c:\dxx\samples\cmd\getstart.xml'))
```

La colonna ORDER nella tabella SALES_TAB viene definita come tipo XMLFILE. Se si dispone di un documento XML nel buffer, è possibile memorizzarlo in un server file.

Funzioni di richiamo

E' possibile utilizzare le funzioni cast predefinite per convertire un UDT XML nel tipo di dati base come descritto in "Richiamo di un documento intero" a pagina 100. XML Extender inoltre fornisce una *funzione di sovraccarico* Content(), utilizzata per il richiamo.

La funzione Content() fornisce i seguenti tipi di richiamo:

- **Richiamo dalla memoria esterna sul server ad una variabile host sul client.**

  E' possibile utilizzare Content() per richiamare un documento XML, memorizzato come server file esterno, in un buffer di memoria. E' possibile consultare la sezione "Content(): richiamo da XMLFILE in CLOB" a pagina 148 per informazioni su tale operazione.
- **Richiamo dalla memoria interna in un server file sterno.**

  E' inoltre possibile utilizzare Content() per richiamare un documento XML memorizzato in DB2 e memorizzarlo in un server file del file system del server DB2. Le seguenti funzioni Content() vengono utilizzate per memorizzare le informazioni nei server file esterni:

  – "Content(): richiamo da XMLVARCHAR in un server file" a pagina 149

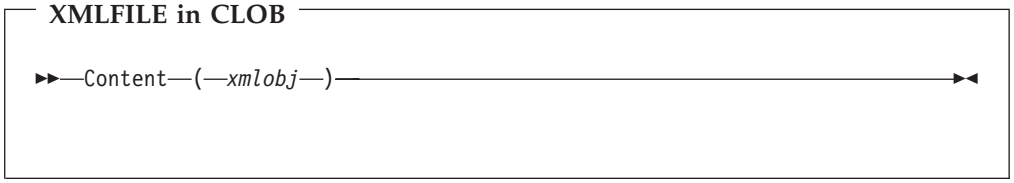
  – "Content(): richiamo da XMLCLOB in un server file esterno" a pagina 151

Negli esempi riportati nella seguente sezione viene utilizzata la shell dei comandi DB2 in cui non è necessario immettere "DB2" all'inizio di ciascun comando.

Content(): richiamo da XMLFILE in CLOB

Scopo
Richiama i dati da un server file e li memorizza in CLOB LOCATOR.

Sintassi



Parametri

Tabella 26. parametro XMLFILE in CLOB

Parametro	Tipo di dati	Descrizione
xmlobj	XMLFILE	Il documento XML.

Tipo restituzione
CLOB (*clob_len*) come LOCATOR

clob_len per DB2 UDB è 2G.

Esempio
Di seguito è riportato l'esempio del richiamo di dati da un server file e della relativa memorizzazione in un indicatore di posizione CLOB.

```
EXEC SQL BEGIN DECLARE SECTION;
     SQL TYPE IS CLOB_LOCATOR xml_buff;
EXEC SQL END DECLARE SECTION;

EXEC SQL CONNECT TO SALES_DB

EXEC SQL DECLARE c1 CURSOR FOR

     SELECT Content(order) from sales_tab
     WHERE sales_person = 'Sriram Srinivasan'

EXEC SQL OPEN c1;

do {
   EXEC SQL FETCH c1 INTO :xml_buff;
   if (SQLCODE != 0) {
                   break;
      }
   else {
     /* con doc XML nel buffer */
     }
   }

EXEC SQL CLOSE c1;

EXEC SQL CONNECT RESET;
```

La colonna ORDER della tabella SALES_TAB è di tipo XMLFILE, quindi la funzione UDF Content() richiama i dati da un server file e li memorizza in un indicatore di posizione CLOB.

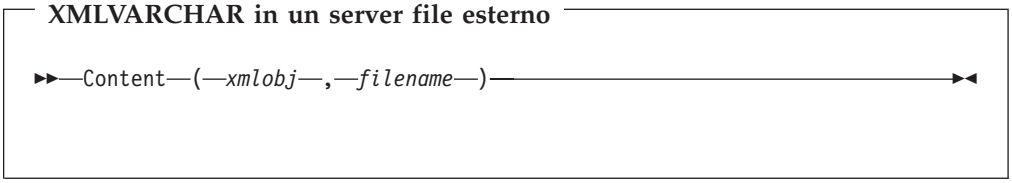
Content(): richiamo da XMLVARCHAR in un server file

Scopo

Richiama il contenuto XML memorizzato come tipo XMLVARCHAR e lo memorizza in un server file esterno.

Sintassi

XMLVARCHAR in un server file esterno



Importante: se un file con il nome specificato esiste già, la funzione Content ne sovrascriverà il contenuto.

Nota:

Parametri

Tabella 27. Parametri XMLVarchar in un server file esterno

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR	Il documento XML.
filename	VARCHAR(512)	Il nome completo del server file.

Tipo restituzione

VARCHAR(512)

Esempio

Di seguito è riportato un esempio di richiamo del contenuto XML memorizzato come tipo XMLVARCHAR e della relativa memorizzazione in un server file esterno.

```
CREATE table app1 (id int NOT NULL, order db2xml.XMLVarchar);
INSERT into app1 values (1, '<?xml version="1.0"?>
   <!DOCTYPE SYSTEM c:\dxx\samples\dtd\getstart.dtd"->
     <Order key="1">
        <Customer>
          <Name>American Motors</Name>
          <Email>parts@am.com</Email>
        </Customer>
        <Part color="black">
          <key>68</key>
          <Quantity>36</Quantity>
          <ExtendedPrice>34850.16</ExtendedPrice>
          <Tax>6.000000e-02</Tax>
          <Shipment>
              <ShipDate>1998-08-19</ShipDate>
              <ShipMode>AIR </ShipMode>
          </Shipment>
          <Shipment>
              <ShipDate>1998-08-19</ShipDate>
              <ShipMode>BOAT </ShipMode>
          </Shipment>
```

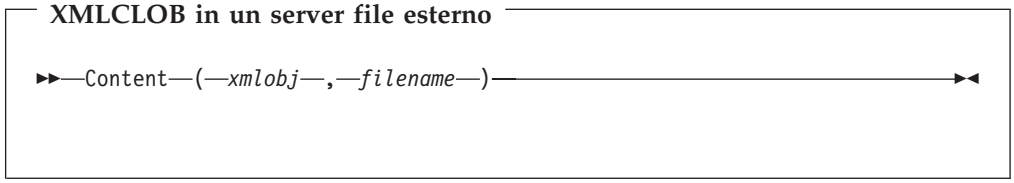
```
    </Order>');

SELECT db2xml.Content(order, 'c:\dxx\samples\cmd\getstart_.dad')
  from app1 where ID=1;
```

Content(): richiamo da XMLCLOB in un server file esterno

Scopo
Richiama il contenuto XML memorizzato come tipo XMLCLOB e lo memorizza in un server file esterno.

Sintassi

XMLCLOB in un server file esterno



Importante: se un file con il nome specificato esiste già, la funzione Content ne sovrascriverà il contenuto.

Parametri

Tabella 28. Parametri XMLCLOB in un server file esterno

Parametro	Tipo di dati	Descrizione
xmlobj	XMLCLOB come LOCATOR	Il documento XML.
filename	VARCHAR(512)	Il nome completo del server file.

Tipo restituzione
VARCHAR(512)

Esempio
Di seguito è riportato un esempio di richiamo del contenuto XML memorizzato come XMLCLOB e della relativa memorizzazione in un server file esterno.

```
CREATE table app1 (id int NOT NULL, order db2xml.XMLCLOB not logged);

INSERT into app1 values (1, '<?xml version="1.0"?>
  <!DOCTYPE SYSTEM c:\dxx\samples\dtd\getstart.dtd"->
    <Order key="1">
      <Customer>
          <Name>American Motors</Name>
          <Email>parts@am.com</Email>
      </Customer>
      <Part color="black">
          <key>68</key>
          <Quantity>36</Quantity>
          <ExtendedPrice>34850.16</ExtendedPrice>
          <Tax>6.000000e-02</Tax>
          <Shipment>
              <ShipDate>1998-08-19</ShipDate>
              <ShipMode>AIR </ShipMode>
          </Shipment>
          <Shipment>
              <ShipDate>1998-08-19</ShipDate>
              <ShipMode>BOAT </ShipMode>
          </Shipment>
    </Order>');

SELECT db2xml.Content(order, 'c:\dxx\samples\cmd\getstart.xml')
from app1 where ID=1;
```

Funzioni di estrazione

Le funzioni di estrazione estraggono il contenuto dell'elemento o il valore attributo da un documento XML e restituiscono i tipi di dati SQL richiesti. XML Extender fornisce una serie di funzioni di estrazione per vari tipi di dati SQL. Le funzioni di estrazione richiedono due tipi di parametri di input. Il primo parametro è l'UDT XML Extender, un UDT XML. Il secondo parametro è il percorso di ubicazione che specifica l'elemento o l'attributo XML. Ciascuna funzione di estrazione restituisce il valore o il contenuto specificato dal percorso di ubicazione.

Poiché alcuni valori dell'elemento o attributo presentano più ricorrenze, le funzioni di estrazione restituiscono un valore scalare o di tabella; l'ultimo è denominato funzione di tabella.

XML Extender fornisce le seguenti funzioni di estrazione:

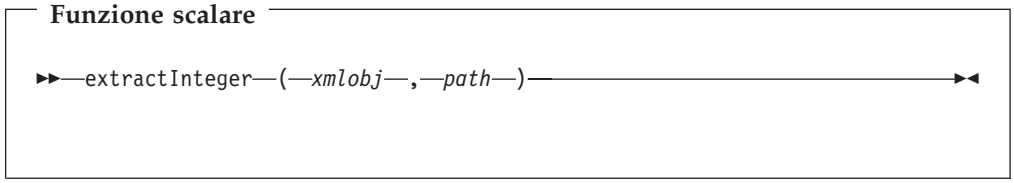
Negli esempi riportati nella seguente sezione viene utilizzata la shell dei comandi DB2 in cui non è necessario immettere "DB2" all'inizio di ciascun comando.

extractInteger() ed extractIntegers()

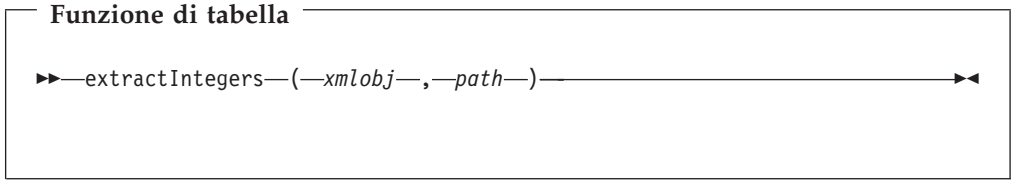
Scopo
Estrae il contenuto dell'elemento o il valore attributo da un documento XML e restituisce i dati come tipo INTEGER.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 29. Parametri delle funzioni extractInteger ed extractIntegers

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione
INTEGER

Nome colonna restituito (funzione tabella)
returnedInteger

Esempio
Esempio di funzione scalare:

Nel seguente esempio, un valore viene restituito quando il valore attributo della chiave è ″1″. Il valore viene estratto come un intero automaticamente convertito in un tipo DECIMAL.

```
CREATE TABLE t1(key decimal(3,2));
INSERT into t1 values
SELECT * from table(db2xml.extractInteger(db2xml.XMLFile
   ('c:\dxx\samples\xml\getstart.xml'), '/Order/[@key="1"]'));
SELECT * from t1;
```

Esempio di funzione di tabella:

Nel seguente esempio, ogni valore chiave per l'ordine di acquisto viene estratto come INTEGER

```
SELECT * from table(db2xml.extractIntegers(db2xml.XMLFile
   ('c:\dxx\samples\xml\getstart.xml'), '/Order/@key')) as x;
```

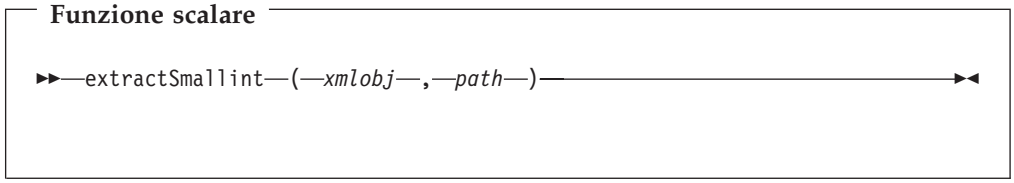
extractSmallint() ed extractSmallints()

Scopo

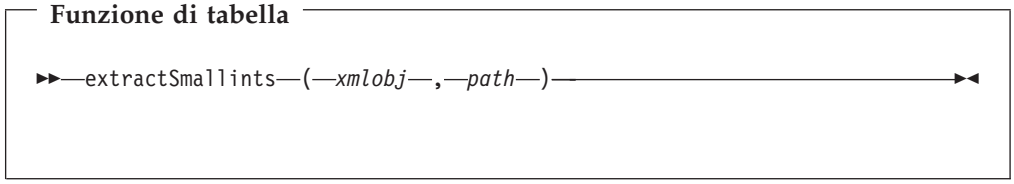
Estrae il contenuto dell'elemento o il valore attributo da un documento XML e restituisce i dati come tipo SMALLINT.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 30. Parametri delle funzioni extractSmallint ed extractSmallints

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione

SMALLINT

Nome colonna restituito (funzione tabella)

returnedSmallint

Esempio

Nel seguente esempio, il valore di Quantity in tutti gli ordini di vendita viene estratto come tipo SMALLINT

```
SELECT * from table(db2xml.extractSmallints(db2xml.File
   ('c:\dxx\samples\xml\getstart.xml'), '/Order/Part/Quantity')) as x;
```

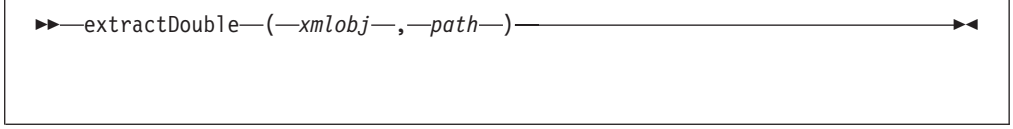
extractDouble() ed extractDoubles()

Scopo

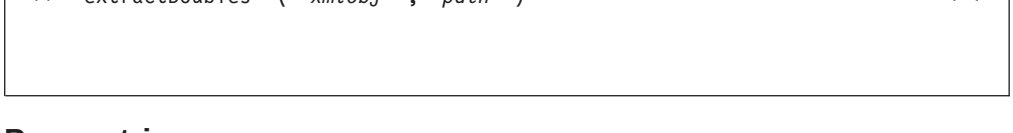
Estrae il contenuto dell'elemento o il valore attributo da un documento XML e restituisce i dati come tipo DOUBLE.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 31. Parametri delle funzioni extractDouble ed extractDoubles

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione

DOUBLE

Nome colonna restituito (funzione tabella)

returnedDouble

Esempio

Esempio di funzione scalare:

Il seguente esempio converte automaticamente il prezzo di un ordine da un tipo DOUBLE in un tipo DECIMAL.

```
CREATE TABLE t1(price, DECIMAL(5,2));
INSERT into t1 values (db2xml.extractDouble(db2xml.XMLFile
   ('c:\dxx\samples\xml\getstart.xml'),
    '/Order/Part[@color="black "]/ExtendedPrice'));
SELECT * from t1;
```

Esempio di funzione di tabella:

Nel seguente esempio, il valore di ExtendedPrice in tutte le parti dell'ordine di vendita viene estratto come tipo DOUBLE.

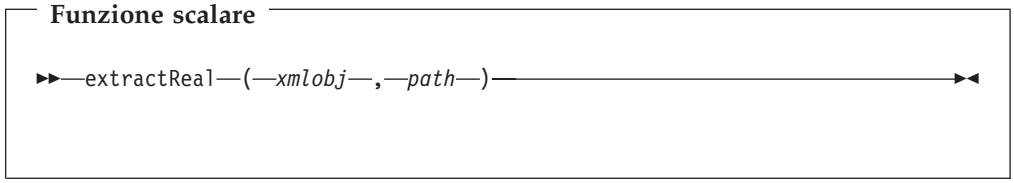
```
SELECT * from table(db2xml.extractDoubles(db2xml.XMLFile
   ('c:\dxx\samples\xml\getstart.xml'), '/Order/Part/ExtendedPrice')) as x;
```

extractReal() ed extractReals()

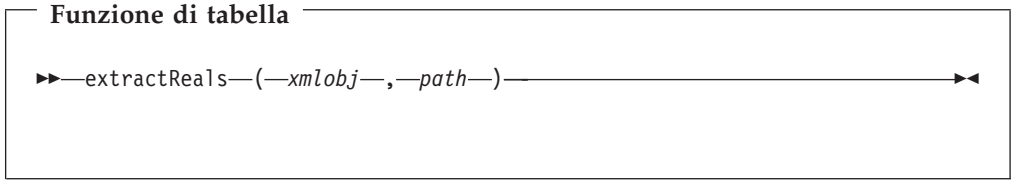
Scopo
Estrae il contenuto dell'elemento o il valore attributo da un documento XML e restituisce i dati come tipo REAL.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 32. Parametri delle funzioni extractReal ed extractReals

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione
REAL

Nome colonna restituito (funzione tabella)
returnedReal

Esempio
Nel seguente esempio, ciascun valore di Tax viene estratto come tipo REAL.

```
SELECT * from table(db2xml.extractReals(db2xml.XMLFile
   ('c:\dxx\samples\xml\getstart.xml'), '/Order/Part/Tax')) as x;
```

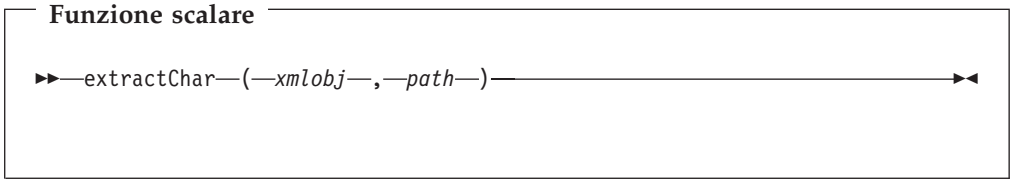
extractChar() ed extractChars()

Scopo

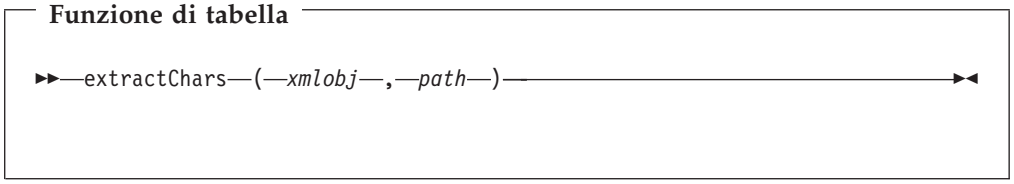
Estrae il contenuto dell'elemento o il valore attributo da un documento XML e restituisce i dati come tipo CHAR.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 33. Parametri delle funzioni extractChar ed extractChars

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione

CHAR

Nome colonna restituito (funzione tabella)

returnedChar

Esempio

Nel seguente esempio il valore di Color viene estratto come tipo CHAR.

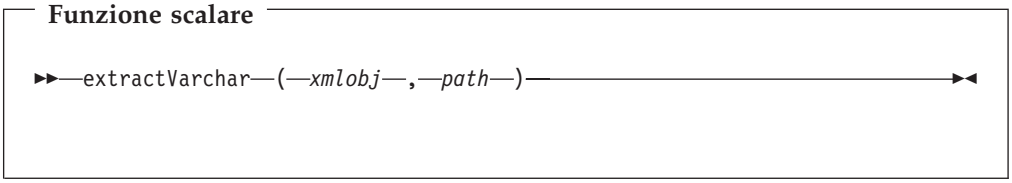
```
SELECT * from table(db2xml.extractChars(Order,
   ('c:\dxx\samples\xml\getstart.xml'), '/Order/Part/@Color')) as x;
```

extractVarchar() ed extractVarchars()

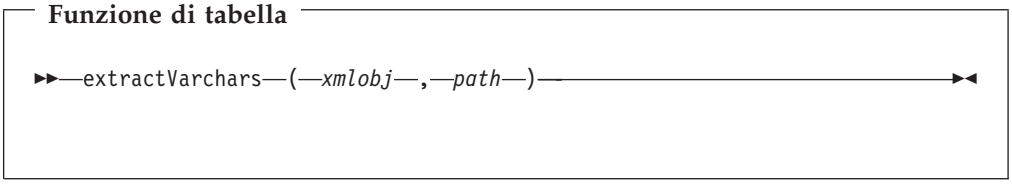
Scopo
Estrae il contenuto dell'elemento o il valore attributo da un documento XML e restituisce i dati come tipo VARCHAR.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 34. Parametri delle funzioni extractVarchar ed extractVarchars

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione
VARCHAR(4K)

Nome colonna restituito (funzione tabella)
returnedVarchar

Esempio
In un database con più di 1000 documenti XML memorizzati nella colonna ORDER della tabella SALES_TAB, è possibile ricercare tutti i clienti che hanno ordinato articoli con ExtendedPrice superiore a 2500.00. La seguente istruzione SQL utilizza la funzione UDF di estrazione nella clausola SELECT:

```
SELECT extractVarchar(Order, '/Order/Customer/Name') from sales_order_view
   WHERE price > 2500.00
```

La funzione UDF extractVarchar() utilizza la colonna ORDER come input e il percorso di ubicazione `/Order/Customer/Name` come identificativo select. La funzione UDF restituisce i nomi dei clienti. Con la clausola WHERE, la funzione di estrazione considera solo gli ordini con ExtendedPrice superiore a 2500.00.

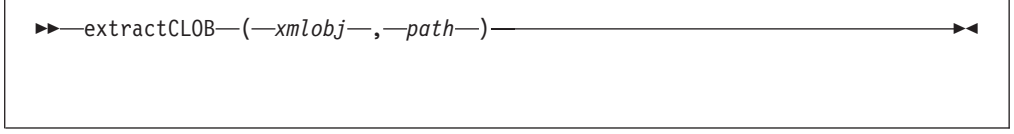
extractCLOB() ed extractCLOBs()

Scopo

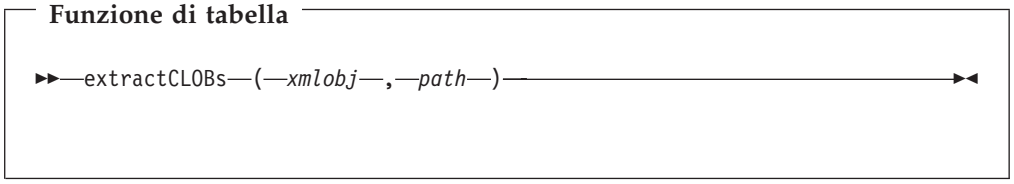
Estrae un frammento di documenti XML, con markup dell'elemento e degli attributi, contenuto degli elementi e degli attributi, incluso elementi secondari. Questa funzione è diversa dalle altre funzioni di estrazione; queste ultime restituiscono solo il contenuto di elementi e attributi. Le funzioni extractClob(s) dovrebbero essere utilizzate per estrarre frammenti di documenti, mentre extractVarchar(s) e extractChar(s) per estrarre valori semplici.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 35. Parametri delle funzioni extractCLOB ed extractCLOBs

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione

CLOB(10K)

Nome colonna restituito (funzione tabella)

returnedCLOB

Esempio

In questo esempio, tutti gli elementi vengono estratti da un ordine di acquisto.

```
SELECT returnedCLOB as part
   from table(db2xml.extractCLOBs(db2xml.XMLFile('c:\dxx\samples\xml\getstart.xml'),
   '/Order/Part')) as x;
```

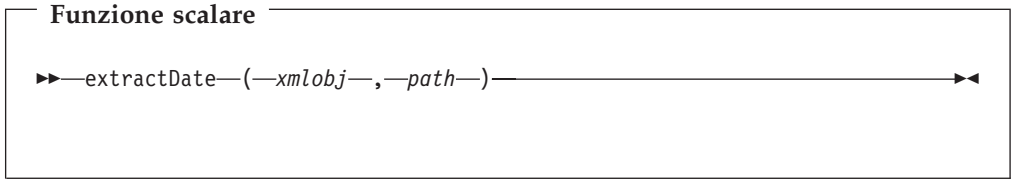
extractDate() ed extractDates()

Scopo

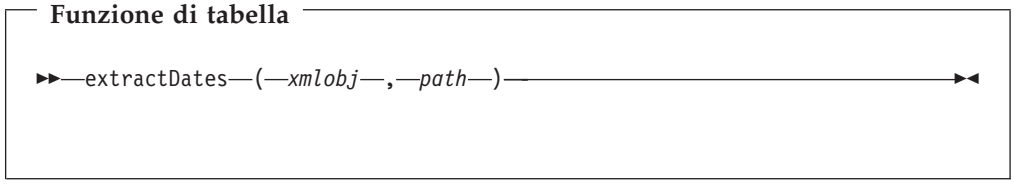
Estrae il contenuto dell'elemento o il valore attributo da un documento XML e restituisce i dati come tipo DATE.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 36. Parametri delle funzioni extractDate ed extractDates

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione

DATE

Nome colonna restituito (funzione tabella)

returnedDate

Esempio

Nel seguente esempio il valore di ShipDate viene estratto come tipo DATE.

```
SELECT * from table(db2xml.extractDates(db2xml.XMLFile
   ('c:\dxx\samples\xml\getstart.xml'), '/Order/Part/Shipment/ShipDate')) as x;
```

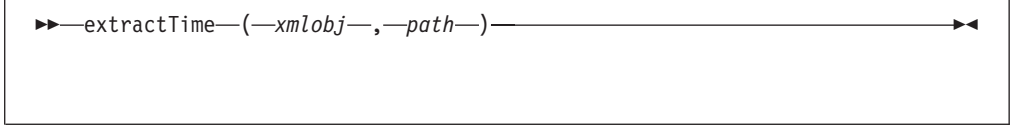
extractTime() ed extractTimes()

Scopo

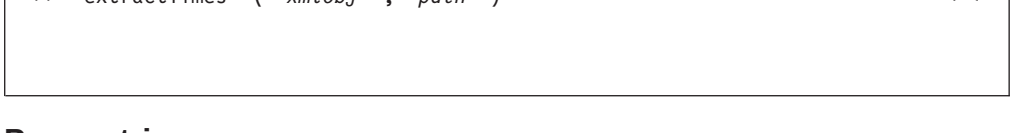
Estrae il contenuto dell'elemento o il valore attributo da un documento XML e restituisce i dati come tipo TIME.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 37. Parametri delle funzioni extractTime ed extractTimes

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione

TIME

Nome colonna restituito (funzione tabella)

returnedTime

Esempio

Questo esempio utilizza i file di esempio del manuale. Esegue la ricerca nel file XML `book.xml` dell'ora in cui è stato fissato il prezzo dei manuali e restituisce i valori come tipo TIME.

File XML:

```
<?xml version="1.0">
<DOCTYPE book SYSTEM "c:\dxx\samples\book.dtd">
<book>
<chapter id="1" date="07/01/97">
<section>This is a section in Chapter One.</section>
<chapter id="2" date="01/02/1997">
<section>This is a section in Chapter Two.</section>
</chapter>
<price date="12/22/1998" time="11.12.13" timestamp="1998-12-22-11.12.13.888888">
38.281
</price>
</book>
```

Esempio di estrazione:

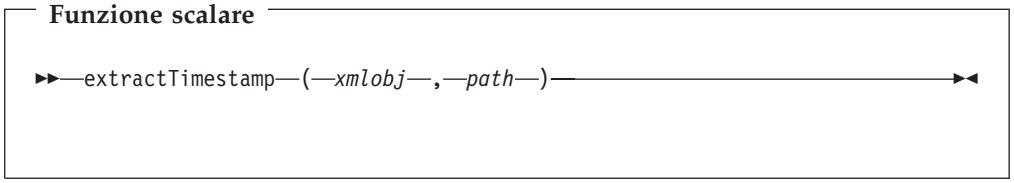
```
CREATE TABLE t1(SaleTime TIME);
INSERT INTO t1 values(db2xml.extractTime(doc, '/book/price/@time'));
SELECT * from t1;
```

extractTimestamp() ed extractTimestamps()

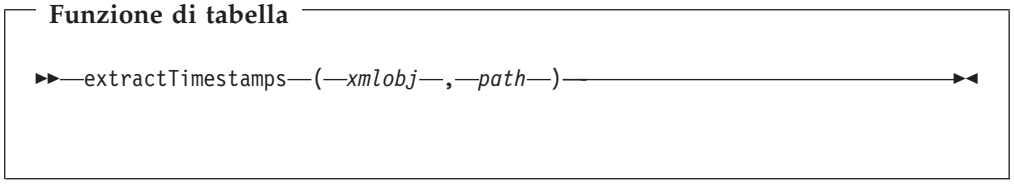
Scopo
Estrae il contenuto dell'elemento o il valore attributo da un documento XML e restituisce i dati come tipo TIMESTAMP.

Sintassi

Funzione scalare



Funzione di tabella



Parametri

Tabella 38. Parametri delle funzioni extractTimestamp ed extractTimestamps

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLFILE o XMLCLOB	Il nome della colonna.
path	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.

Tipo restituzione
TIMESTAMP

Nome colonna restituito (funzione tabella)
returnedTimestamp

Esempio
Questo esempio utilizza i file di esempio del manuale. Esegue la ricerca nel file XML `book.xml` dell'ora che indica il momento in cui è stato fissato il prezzo di ciascun manuale ed estrae il valore come tipo TIMESTAMP.

File XML:

```
<?xml version="1.0">
<DOCTYPE book SYSTEM "c:\dxx\samples\book.dtd">
<book>
<chapter id="1" date="07/01/97">
<section>This is a section in Chapter One.</section>
<chapter id="2" date="01/02/1997">
<section>This is a section in Chapter Two.</section>
</chapter>
<price date="12/22/1998" time="11.12.13" timestamp="1998-12-22-11.12.13.888888">
```

```
38.281
</price>
</book>
```

Esempio di estrazione:

```
CREATE TABLE t1(SaleTime TIMESTAMP);
INSERT INTO t1 values(db2xml.extractTimestamp(doc, '/book/price/@timestamp'));
SELECT * from t1;
```

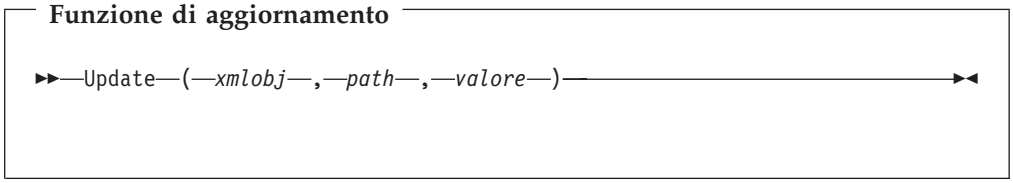
Funzione di aggiornamento

La funzione Update() esegue l'aggiornamento di un valore di elemento o attributo specificato in uno o più documenti XML memorizzati nella colonna XML. E' anche possibile utilizzare le funzioni cast predefinite per convertire un tipo di base SQL nell'UDT XML, come descritto in "Aggiornamento dei dati XML" a pagina 104.

Scopo

Sostituisce il nome della colonna di un UDT XML, il percorso di ubicazione e una stringa del valore di aggiornamento con un UDT XML corrispondente al primo parametro di input. Con la funzione Update() è possibile specificare l'elemento o l'attributo da aggiornare.

Sintassi



Parametri

Tabella 39. I parametri della funzione UDF Update

Parametro	Tipo di dati	Descrizione
xmlobj	XMLVARCHAR, XMLCLOB come LOCATOR	Il nome della colonna.
percorso	VARCHAR	Il percorso dell'ubicazione dell'elemento o dell'attributo.
valore	VARCHAR	La stringa di aggiornamento.

Importante: Notare che UDF Update supporta percorsi di ubicazione che presentano predicati con attributi, ma non elementi. Ad esempio, il seguente predicato è supportato:

```
'/Order/Part[@color="black "]/ExtendedPrice'
```

Mentre il seguente predicato non è supportato:

```
'/Order/Part/Shipment/[Shipdate < "11/25/00"]'
```

Tipo restituito

Tipo di dati	Tipo restituito
XMLVARCHAR	XMLVARCHAR
XMLCLOB come LOCATOR	XMLCLOB

Esempio

Il seguente esempio aggiorna l'ordine di acquisto gestito dall'addetto alle vendite Sriram Srinivasan.

```
UPDATE sales_tab
   set order = Update(order, '/Order/Customer/Name', 'IBM')
   WHERE sales_person = 'Sriram Srinivasan'
```

In questo esempio, il contenuto di `/Order/Customer/Name` viene sostituito con `IBM`.

Utilizzo

Quando si utilizza la funzione Update per modificare un valore in uno o più documenti XML documents, tale funzione sostituisce i documenti XML all'interno della colonna XML. In base alle emissioni del parser XML, alcune parti del documento originale vengono conservate, mentre altre parti vengono perse o modificate. La seguente sezione descrive le modalità di elaborazione del documento e fornisce esempi su come appare il documento prima e dopo gli aggiornamenti.

Elaborazione del documento XML con la funzione Update

Quando la funzione Update sostituisce documenti XML, deve ricostruire il documento in base all'emissione del parser XML. Tabella 40 descrive attraverso esempi come vengono gestite le parti del documento. Per gli esempi di confronto dei documenti XML prima e dopo un aggiornamento, consultare la sezione "Esempi" a pagina 169.

Tabella 40. Regole della funzione Update

Elemento o tipo di nodo	Esempio di codice del documento XML	Stato dopo l'aggiornamento
Dichiarazione XML	`<?xml version='1.0' encoding='utf-8' standalone='yes' >`	La dichiarazione XML viene conservata: • Le informazioni sulla versione vengono conservate. • La dichiarazione di codifica viene conservata e appare quando specificato nel documento originale. • La dichiarazione autonoma viene conservata e appare quando specificato nel documento originale. • Dopo l'aggiornamento, per delineare i valori vengono utilizzati singoli apici.

Tabella 40. Regole della funzione Update (Continua)

Elemento o tipo di nodo	Esempio di codice del documento XML	Stato dopo l'aggiornamento
Dichiarazione DOCTYPE	`<!DOCTYPE books SYSTEM "http://dtds.org/books.dtd" > <!DOCTYPE books PUBLIC "local.books.dtd" "http://dtds.org/books.dtd" > <!DOCTYPE books> -Any of <!DOCTYPE books ( S ExternalID ) ? [ internal-dtd-subset ] > -Such as <!DOCTYPE books [ <!ENTITY mydog "Spot"> ] >? [ internal-dtd-subset ] >`	La dichiarazione del tipo di documento viene conservata: • Il nome dell'elemento root è supportato. • `ExternalID` pubblici e di sistema vengono conservati e appaiono quando specificato nel documento originale. • Il sottoinsieme DTD interno NON viene conservato. Le entità vengono sostituite; i valori predefiniti per gli attributi vengono elaborati e appaiono nei documenti di emissione. • Dopo l'aggiornamento, per delineare i valori URI pubblici e di sistema vengono utilizzati doppi apici. • Il parser XML4c corrente non riporta una dichiarazione XML che non contiene un `ExternalID` o un sottoinsieme DTD interno. Dopo l'aggiornamento, in questo caso la dichiarazione DOCTYPE potrebbe mancare.
Istruzioni di elaborazione	`<?xml-stylesheet title="compact" href="datatypes1.xsl" type="text/xsl"?>`	Le istruzioni di elaborazione vengono conservate.
Commenti	`<!-- comment -->`	I commenti vengono conservati quando si trovano all'interno dell'elemento root. I commenti esterni all'elemento root vengono eliminati.

Elementi	`<books>`
	contenuto
	`</books>`
		Gli elementi vengono conservati.

Attributi	`id='1' date='01/02/1997"`	Gli attributi di elementi vengono conservati. • Dopo l'aggiornamento, per delineare i valori vengono utilizzati doppi apici. • I dati all'interno di attributi vengono saltati. • Le entità vengono sostituite.

Tabella 40. Regole della funzione Update (Continua)

Elemento o tipo di nodo	Esempio di codice del documento XML	Stato dopo l'aggiornamento
Nodi di testo	`Questo capitolo riguarda my dog &mydoc;.` `Questo capitolo riguarda my dog Spot.`	I nodi di testo (contenuto dell'elemento) vengono conservati. • I dati all'interno dei nodi di testo vengono saltati. • Le entità vengono sostituite.

Ricorrenze multiple

Quando un percorso di ubicazione viene fornito in UDF Update(), il contenuto di ciascun elemento o attributo con un percorso corrispondente viene aggiornato con il valore fornito. Ciò significa che, se un documento presenta ricorrenze multiple di percorsi di ubicazione, la funzione Update sostituisce i valori esistenti con il valore fornito nel parametro *value*.

E' possibile specificare un predicato nel parametro *path* per fornire percorsi di ubicazione distinti, in modo da evitare aggiornamenti non intenzionali. Notare che UDF Update supporta percorsi di ubicazione che presentano predicati con attributi, ma non elementi. Per ulteriori informazioni, consultare "Parametri" a pagina 166.

Esempi

Gli esempi seguenti mostrano istanze di un documento XML prima e dopo un aggiornamento.

Esempio 1:

Prima:

```
<?xml version='1.0'   encoding='utf-8'    standalone="yes"?>
<!DOCTYPE book PUBLIC "public.dtd" "system.dtd">
<?pitarget option1='value1' option2='value2'?>
<!-- comment -->
<book>
  <chapter id="1" date='07/01/1997'>
    <!-- first section -->
<section>This is a section in Chapter One.</section>
</chapter>
<chapter id="2" date="01/02/1997">
<section>This is a section in Chapter Two.</section>
    <footnote>A footnote in Chapter Two is here.</footnote>
</chapter>
  <price date="12/22/1998" time="11.12.13"
       timestamp="1998-12-22-11.12.13.888888">38.281</price>
</book>
```

- Contiene uno spazio in bianco nella dichiarazione XML
- Specifica un'istruzione di elaborazione
- Contiene un commento al di fuori del nodo root
- Specifica PUBLIC ExternalID
- Contiene un commento all'interno del nodo root

Dopo:

```
<?xml version='1.0' encoding='utf-8' standalone='yes'?>
<!DOCTYPE book PUBLIC "public.dtd" "system.dtd">
<?pitarget option1='value1' option2='value2'?><book>
  <chapter id="1" date="07/01/1997">
        <!-- first section -->
<section>This is a section in Chapter One.</section>
</chapter>
<chapter id="2" date="01/02/1997">
<section>This is a section in Chapter Two.</section>
        <footnote>A footnote in Chapter Two is here.</footnote>
</chapter>
  <price date="12/22/1998" time="11.12.13"
       timestamp="1998-12-22-11.12.13.888888">60.02</price>
</book>
```

- Lo spazio in bianco all'interno della markup è stato eliminato
- Le istruzioni di elaborazione sono state conservate
- Il commento al di fuori del nodo root non è stato conservato
- PUBLIC ExternalID viene conservato
- Il commento all'interno del nodo root viene conservato
- Il valore modificato è il valore dell'elemento <price>

Esempio 2:

Prima:

```
<?xml version='1.0'     ?>
<!DOCTYPE book>
<!-- comment -->
<book>
  ...
</book>
```

Contiene la dichiarazione DOCTYPE senza un ExternalID o un sottoinsieme DTD interno. Non supportato.

Dopo:

```
<?xml version='1.0'?>
<book>
  ...
</book>
```

La dichiarazione DOCTYPE non è riportata dal parser XML e non viene conservata.

Esempio 3:

Prima:

```
<?xml version='1.0'     ?>
<!DOCTYPE book [ <!ENTITY myDog "Spot"> ]>
<!-- comment -->
<book>
  <chapter id="1" date='07/01/1997'>
       <!-- first section -->
       <section>This is a section in Chapter
         One about my dog &;myDog;.</section>
       ...
</chapter>
  ...
</book>
```

- Contiene uno spazio in bianco nella markup
- Specifica un sottoinsieme DTD interno
- Specifica l'entità nel nodo di testo

Dopo:

```
<?xml version='1.0'?>
<!DOCTYPE book>
<book>
  <chapter id="1" date="07/01/1997">
       <!-- first section -->
       <section>This is a section in Chapter
         One about my dog Spot.</section>
          ...
</chapter>
  ...
</book>
```

- Lo spazio in bianco nella markup è stato eliminato
- Il sottoinsieme DTD interno non viene conservato
- L'entità nel nodo di testo viene risolta e sostituita

Capitolo 10. XML Extenderprocedure memorizzate

XML Extender fornisce le procedure memorizzate per la gestione delle colonne e le raccolte XML. Queste procedure memorizzate possono essere richiamate dal client DB2. L'interfaccia client può essere integrata in SQL, ODBC o JDBC. Per informazioni sulle modalità di richiamo delle procedure memorizzate fare riferimento alla relativa sezione della pubblicazione *DB2 UDB Administration Guide*.

Le procedure memorizzate utilizzano lo schema db2xml, che è il nome schema di XML Extender

XML Extender fornisce tre tipi di procedure memorizzate:

- "Procedure memorizzate di gestione" a pagina 175, che consentono il completamento delle attività di gestione
- Le "Procedure memorizzate di composizione" a pagina 182, che creano documenti XML utilizzando i dati contenuti nelle tabelle esistenti nel database
- Le "Procedure di memorizzazione di scomposizione" a pagina 189, che scompongono i documenti XML in entrata e i dati memorizzati nelle tabelle nuove o esistenti del database.

I limiti di parametro utilizzati dalle procedure memorizzate della raccolta XML sono descritti in "Appendice D. Limiti di XML Extender" a pagina 241.

Specifica dei file di inclusione

Accertarsi di includere i file di intestazione esterni di XML Extender nel programma che richiama le procedure memorizzate. I file di intestazione sono ubicati nella directory `DXX_INSTALL/include`. `DXX_INSTALL` è la directory di installazione per XML Extender. La directory di intestazione varia in base al sistema operativo. I file di intestazione sono:

dxx.h I tipi di dati e le costanti definite per XML

dxxrc.h Il codice di ritorno XML Extender

La sintassi per l'inclusione di tali file di intestazione è:

```
#include "dxx.h"
#include "dxxrc.h"
```

Accertarsi che il percorso dei file di inclusione sia specificato nel makefile con l'opzione di compilazione.

Richiamo delle procedure memorizzate di XML Extenders

Per richiamare XML Extender generalmente si utilizza la seguente sintassi:

```
CALL db2xml.function_entry_point
```

Dove:

db2xml
Specifica la libreria delle procedure memorizzate XML Extender. Su sistemi

operativi UNIX, la libreria è memorizzata nell'indirizzario sqllib/function. Su sistemi operativi Windows, la libreria è memorizzata nell'indirizzario *DXX_INSTALL*/bin.

function_entry_point
Specifica gli argomenti inoltrati alla procedura memorizzata.

Nell'istruzione CALL, gli argomenti inoltrati alla procedura memorizzata devono essere variabili host e non costanti o espressioni. Le variabili host possono presentare indicatori nulli. Consultare gli esempi per il richiamo delle procedure memorizzate nelle directory DXX_INSTALL/samples/c e DXX_INSTALL/samples/cli e nelle seguenti sezioni di questa pubblicazione: "Composizione del documento XML" a pagina 32 e "Capitolo 6. Gestione dei dati delle raccolte XML" a pagina 113.

Nella directory DXX_INSTALL/samples/c, i file di codice SQC vengono forniti per richiamare le procedure memorizzate della raccolta XML utilizzando SQL integrata. Nella directory DXX_INSTALL/samples/cli, i file di esempio descrivono come richiamare le procedure memorizzate utilizzando CLI (Call Level Interface).

Incremento del limite CLOB

Il limite predefinito per il parametro CLOB quando questo viene trasmesso a una procedura memorizzata è di 1 MB. E' possibile aumentare il limite completando i seguenti passi:

1. Rilasciare ogni procedura memorizzata. Ad esempio:

```
db2   "drop procedure db2xml.dxxShredXML"
```

2. Creare una nuova procedura con il limite CLOB incrementato. Ad esempio:

```
db2   "create procedure db2xml.dxxShredXML(in     dadBuf      clob(100K),
                                           in     XMLObj      clob(10M),
                                           out    returnCode  integer,
                                           out    returnMsg   varchar(1024)
                                           )
       external name 'db2xml.dxxShredXML'
       language C
       parameter style DB2DARI
       not deterministic
       fenced
       null call;
```

Informazioni preliminari

Eseguire il bind del database con i file di bind DB2 CLI e delle procedure memorizzate XML Extender. E' possibile utilizzare un file dei comandi di esempio, getstart_prep.cmd, per eseguire il bind dei file. Questo file dei comandi è ubicato nella directory DXX_INSTALL/samples/cmd.

1. Collegamento al database. Ad esempio:

```
db2 "connect to SALES_DB"
```

2. Entrare nella directory DXX_INSTALL/bnd ed eseguire il bind di XML Extender sul database.

```
db2 "bind @dxxbind.lst"
```

3. Entrare nella directory sqllib/bnd ed eseguire il bind di CLI sul database.

```
db2 "bind @db2cli.lst"
```

4. Terminare la connessione.

```
db2 "terminate"
```

Procedure memorizzate di gestione

Queste procedure memorizzate vengono utilizzate per le attività di gestione, quali l'abilitazione e la disabilitazione di una raccolta o colonna XML. Tali procedure vengono richiamate dal wizard di gestione e dal comando di gestione **dxxadm** di XML Extender.

dxxEnableDB()

Scopo

Abilita il database. Una volta abilitato il database, XML Extender crea i seguenti oggetti:

- Gli UDT (tipi definiti dall'utente) XML Extender.
- Gli UDF (funzioni definite dall'utente) XML Extender.
- La tabella di riferimento DTD di XML Extender, DTD_REF, che memorizza i DTD e le relative informazioni. Per una completa descrizione della tabella DTD_REF, consultare la sezione "Tabella di riferimento DTD" a pagina 193.
- La tabella di utilizzo XML Extender, XML_USAGE, che memorizza le informazioni comuni per ciascuna colonna abilitata per XML e per ciascuna raccolta. Per una completa descrizione della tabella XML_USAGE, consultare la sezione "Tabella di utilizza XML" a pagina 193.

```
dxxEnableDB(char(dbName) dbName,       /* input */
            long         returnCode,   /* output */
            varchar(1024) returnMsg)   /* output */
```

Parametri

Tabella 41. Parametri di dxxEnableDB()

Parametro	Descrizione	Parametro IN/OUT
dbName	Il nome del database.	IN
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

dxxDisableDB()

Scopo

Disabilita il database. Quando XML Extender disabilita il database, vengono cancellati i seguenti oggetti:

- Gli UDT (tipi definiti dall'utente) XML Extender.
- Gli UDF (funzioni definite dall'utente) XML Extender.
- La tabella di riferimento DTD di XML Extender, DTD_REF, che memorizza i DTD e le relative informazioni. Per una completa descrizione della tabella DTD_REF, consultare la sezione "Tabella di riferimento DTD" a pagina 193.
- La tabella di utilizzo XML Extender, XML_USAGE, che memorizza le informazioni comuni per ciascuna colonna abilitata per XML e per ciascuna raccolta. Per una completa descrizione della tabella XML_USAGE, consultare la sezione "Tabella di utilizza XML" a pagina 193.

Importante: è necessario disabilitare tutte le colonne XML prima di disabilitare un database. XML Extender non può disabilitare un database che contiene colonne o raccolte abilitate per XML.

```
dxxDisableDB(char(dbName)     dbName,       /* input */
             long             returnCode,     /* output */
             varchar(1024)    returnMsg)      /* output */
```

Parametri

Tabella 42. Parametri di dxxDisableDB()

Parametro	Descrizione	Parametro IN/OUT
dbName	Il nome del database.	IN
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

dxxEnableColumn()

Scopo

Abilita una colonna XML. Quando si abilita una colonna, XML Extender completa le seguenti attività:

- Determina se la tabella XML presenta una chiave primaria; in caso contrario, XML Extender modifica la tabella XML e aggiunge una colonna denominata DXXROOT_ID.
- Crea le tabelle laterali specificate nel file DAD con una colonna contenente un identificativo univoco per ciascuna riga della tabella XML. Questa colonna è il parametro *root_id* specificato dall'utente o il parametro DXXROOT_ID specificato da XML Extender.
- Crea una vista predefinita per la tabella XML e le relative tabelle laterali, utilizzando, facoltativamente, un nome specificato dall'utente.

```
dxxEnableColumn(char(dbName) dbName,      /* input */
                char(tbName) tbName,      /* input */
                char(colName) colName,     /* input */
                    CLOB(100K)     DAD,         /* input */
                char(tablespace) tablespace,  /* input */
                char(defaultView) defaultView, /* input */
                char(rootID) rootID,      /* input */
             long           returnCode,       /* output */
             varchar(1024)  returnMsg)        /* output */
```

Parametri

Tabella 43. Parametri di dxxEnableColumn()

Parametro	Descrizione	Parametro IN/OUT
dbName	Il nome del database.	IN
tbName	Il nome della tabella contenente la colonna XML.	IN
colName	Il nome della colonna XML.	IN
DAD	Il CLOB contenente il file DAD.	IN
tablespace	Il table space che contiene le tabelle laterali diverse dal table space predefinito. Se non viene specificato, viene utilizzato il table space predefinito.	IN
defaultView	Il nome della vista predefinita che unisce la tabella applicativa e le tabelle laterali.	IN
rootID	Il nome della chiave primaria singola della tabella applicativa da utilizzare come *root_id* per la tabella laterale.	IN
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

dxxDisableColumn()

Scopo

Disabilita la colonna abilitata per XML. La colonna XML, una volta disabilitata, non può più contenere i tipi di dati XML.

```
dxxDisableColumn(char(dbName) dbName,      /* input */
                 char(tbName) tbName,      /* input */
                 char(colName) colName,     /* input */
            long           returnCode,       /* output */
            varchar(1024)  returnMsg)        /* output */
```

Parametri

Tabella 44. Parametri di dxxDisableColumn()

Parametro	Descrizione	Parametro IN/OUT
dbName	Il nome del database.	IN
tbName	Il nome della tabella contenente la colonna XML.	IN
colName	Il nome della colonna XML.	IN
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

dxxEnableCollection()

Scopo

Abilita una raccolta XML associata a una tabella applicativa.

```
dxxEnableCollection(char(dbName) dbName,      /* input */
                    char(colName) colName,     /* input */
                    CLOB(100K)     DAD,         /* input */
                    char(tablespace) tablespace,  /* input */
             long           returnCode,       /* output */
             varchar(1024)  returnMsg)        /* output */
```

Parametri

Tabella 45. Parametri di dxxEnableCollection()

Parametro	Descrizione	Parametro IN/OUT
dbName	Il nome del database.	IN
colName	Il nome della raccolta XML.	IN
DAD	Il CLOB contenente il file DAD.	IN
tablespace	Il table space che contiene le tabelle laterali diverse dal table space predefinito. Se non viene specificato, viene utilizzato il table space predefinito.	IN
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

dxxDisableCollection()

Scopo

Disabilita una raccolta abilitata per XML, eliminando i contrassegni che identificano le tabelle e le colonne come parte di una raccolta.

```
dxxDisableCollection(char(dbName) dbName,      /* input */
                     char(colName) colName,     /* input */
             long           returnCode,       /* output */
             varchar(1024)  returnMsg)        /* output */
```

Parametri

Tabella 46. Parametri di dxxDisableCollection()

Parametro	Descrizione	Parametro IN/OUT
dbName	Il nome del database.	IN
colName	Il nome della raccolta XML.	IN
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

Procedure memorizzate di composizione

Le procedure memorizzate di composizione dxxGenXML() e dxxRetrieveXML() consentono la creazione dei documenti XML utilizzando i dati contenuti nelle tabelle esistenti del database. La procedura memorizzata dxxGenXML() rileva un file DAD come input; non richiede una raccolta XML abilitata. La procedura memorizzata dxxRetrieveXML() rileva un nome di raccolta XML abilitata come input.

dxxGenXML()

Scopo

Crea documenti XML utilizzando i dati memorizzati nelle tabelle della raccolta XML specificati da <Xcollection> nel file DAD e inserisce ciascun documento XML come riga nella tabella dei risultati. E' inoltre possibile posizionare il cursore sulla tabella dei risultati ed eseguire la lettura sequenziale della serie di risultati.

Per maggiore flessibilità, dxxGenXML() consente all'utente di specificare il numero massimo di righe che devono essere create nella tabella dei risultati. In tal modo viene ridotto il periodo di attesa risultati durante un processo di prova. La procedura memorizzata restituisce il numero delle righe effettive contenute nella tabella e le eventuali informazioni sugli errori, compresi i codici e i messaggi di errore.

Per supportare interrogazioni dinamiche, dxxGenXML() rileva il parametro di input *override*. Sulla base del parametro di input *overrideType*, l'applicazione può sostituire SQL_stmt per l'associazione SQL oppure le condizioni di RDB_node per la relativa associazione nel file DAD. ll parametro di input *overrideType* è utilizzato per definire ulteriormente il tipo del parametro *override*. Per dettagli relativi al parametri *override*, consultare la sezione "Sostituzione dinamica dei valori del file DAD" a pagina 117.

```
dxxGenXML(CLOB(100K)     DAD,             /* input */
          char(resultTabName) resultTabName,   /* input */
          integer        overrideType     /* input */
               varchar(1024)  override,         /* input */
               integer        maxRows,          /* input */
               integer        numRows,          /* output */
             long           returnCode,       /* output */
             varchar(1024)  returnMsg)        /* output */
```

Parametri

Tabella 47. Parametri di dxxGenXML()

Parametro	Descrizione	Parametro IN/OUT
DAD	Il CLOB contenente il file DAD.	IN
resultTabName	Il nome della tabella dei risultati, già esistente. La tabella contiene solo una colonna di tipo XMLVARCHAR o XMLCLOB.	IN
overrideType	L'indicatore del tipo del seguente parametro *override*: • **NO_OVERRIDE**: nessuna sostituzione. • **SQL_OVERRIDE**: sostituzione con SQL_stmt. • **XML_OVERRIDE**: sostituzione con una condizione basata su XPath.	IN

Tabella 47. Parametri di dxxGenXML() (Continua)

Parametro	Descrizione	Parametro IN/OUT
override	Sostituisce la condizione nel file DAD. Il valore di input si basa sul valore *overrideType*. • **NO_OVERRIDE**: una stringa NULL. • **SQL_OVERRIDE**: un'istruzione SQL valida. L'utilizzo di questo *overrideType* richiede che nel file DAD sia utilizzata l'associazione SQL. L'istruzione SQL di input sostituisce SQL_stmt nel file DAD. • **XML_OVERRIDE**: una stringa che contiene una o più espressioni delimitate da doppi apici e separate da ″AND″. L'utilizzo di questo *overrideType* richiede che nel file DAD sia utilizzata l'associazione RDB_node.	IN
maxRows	Il numero massimo di righe contenuto nella tabella dei risultati.	IN
numRows	Il numero effettivo di righe generato nella tabella dei risultati.	OUT
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

Esempi

Nel seguente esempio il nome della tabella dei risultati creata è XML_ORDER_TAB e la tabella presenta una colonna di tipo XMLVARCHAR.

```
#include "dxx.h"
#include "dxxrc.h"

EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
              SQL TYPE is CLOB(100K)  dad;              /* DAD */
              SQL TYPE is CLOB_FILE   dadFile;          /* dad file */
              char    result_tab[32];    /* name of the result table */
              char    override[2];       /* override, will set to NULL*/
              short   overrideType;      /* defined in dxx.h */
              short   max_row;           /* maximum number of rows */
              short   num_row;           /* actual number of rows */
              long    returnCode;        /* return error code */
              char                  returnMsg[1024]; /* error message text */
              short       dad_ind;
              short   rtab_ind;
              short   ovtype_ind;
              short   ov_inde;
              short   maxrow_ind;
```

```
                short   numrow_ind;
                short                   returnCode_ind;
                short                   returnMsg_ind;

EXEC SQL END DECLARE SECTION;

                /* create table */
                EXEC CREATE TABLE xml_order_tab (xmlorder XMLVarchar);

                /* read data from a file to a CLOB */
                strcpy(dadfile.name,"e:\dxx\dad\litem3.dad");
                dadfile.name_length = strlen("e:\dxx\dad\litem3.dad");
                dadfile.file_options = SQL_FILE_READ;
                EXEC SQL VALUES (:dadfile) INTO :dad;
                strcpy(result_tab,"xml_order_tab");
                override[0] = '\0';
                overrideType = NO_OVERRIDE;
                max_row = 500;
                num_row = 0;
                returnCode = 0;
                msg_txt[0] = '\0';
                collection_ind = 0;
                dad_ind = 0;
                rtab_ind = 0;
                ov_ind = -1;
                ovtype_ind = 0;
                maxrow_ind = 0;
                numrow_ind = -1;
                returnCode_ind = -1;
                returnMsg_ind = -1;

                /* Call the store procedure */
                EXEC SQL CALL dxxGenXML(:dad:dad_ind;
                       :result_tab:rtab_ind,
                       :overrideType:ovtype_ind,:override:ov_ind,
                       :max_row:maxrow_ind,:num_row:numrow_ind,
                                :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

dxxRetrieveXML()

Scopo

Abilita un file DAD per l'utilizzo sia nella composizione che nella scomposizione. E' inoltre possibile utilizzare la procedura memorizzata dxxRetrieveXML() per richiamare i documenti XML scomposti. Come input, dxxRetrieveXML() rileva un buffer contenente il file DAD, il nome della tabella dei risultati creata e il numero massimo di righe da restituire e restituisce una serie di risultati della tabella dei risultati, il numero di righe effettivo in essa contenute, un codice di errore e il testo del messaggio.

Per supportare interrogazioni dinamiche, dxxRetrieveXML() rileva il parametro di input *override*. Sulla base del parametro di input *overrideType*, l'applicazione può sostituire SQL_stmt per l'associazione SQL oppure le condizioni di RDB_node per la relativa associazione nel file DAD. ll parametro di input *overrideType* è utilizzato per definire ulteriormente il tipo del parametro *override*. Per dettagli relativi al parametri *override*, consultare la sezione "Sostituzione dinamica dei valori del file DAD" a pagina 117.

I requisiti del file DAD relativi a dxxRetrieveXML() corrispondono ai requisiti di dxxGenXML(). L'unica differenza è che il file DAD non è un parametro di input per dxxRetrieveXML(), ma è il nome di una raccolta XML abilitata.

```
dxxRetrieveXML(char(collectionName) collectionName,   /* input */
               char(resultTabName) resultTabName,    /* input */
                 integer        overrideType,     /* input */
                 varchar(1024)  override,         /* input */
                 integer        maxRows,          /* input */
                 integer        numRows,          /* output */
               long           returnCode,       /* output */
               varchar(1024)  returnMsg)        /* output */
```

Parametri

Tabella 48. Parametri di dxxRetrieveXML()

Parametro	Descrizione	Parametro IN/OUT
collectionName	Il nome di una raccolta XML abilitata	IN
resultTabName	Il nome della tabella dei risultati, già esistente. La tabella contiene solo una colonna di tipo XMLVARCHAR o XMLCLOB.	IN
overrideType	L'indicatore del tipo del seguente parametro *override*: • **NO_OVERRIDE**: nessuna sostituzione. • **SQL_OVERRIDE**: sostituzione con SQL_stmt. • **XML_OVERRIDE**: sostituzione con una condizione basata su XPath.	IN

Tabella 48. Parametri di dxxRetrieveXML() (Continua)

Parametro	Descrizione	Parametro IN/OUT
override	Sostituisce la condizione nel file DAD. Il valore di input si basa sul valore *overrideType*. • **NO_OVERRIDE**: una stringa NULL. • **SQL_OVERRIDE**: un'istruzione SQL valida. L'utilizzo di questo *overrideType* richiede che nel file DAD sia utilizzata l'associazione SQL. L'istruzione SQL di input sostituisce SQL_stmt nel file DAD. • **XML_OVERRIDE**: una stringa che contiene una o più espressioni delimitate da doppi apici e separate da ″AND″. L'utilizzo di questo *overrideType* richiede che nel file DAD sia utilizzata l'associazione RDB_node.	IN
maxRows	Il numero massimo di righe contenuto nella tabella dei risultati.	IN
numRows	Il numero effettivo di righe generato nella tabella dei risultati.	OUT
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

Esempi

Di seguito è riportato un esempio di un richiamo di dxxRetrieveXML(). In questo esempio il nome della la tabella dei risultati creata è XML_ORDER_TAB e la tabella presenta una colonna di tipo XMLVARCHAR.

```
#include "dxx.h"
#include "dxxrc.h"

EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
              char    collection[32];    /* dad buffer */
              char    result_tab[32];    /* name of the result table */
              char    override[2];       /* override, will set to NULL*/
              short   overrideType;      /* defined in dxx.h */
              short   max_row;           /* maximum number of rows */
              short   num_row;           /* actual number of rows */
              long    returnCode;        /* return error code */
              char                  returnMsg[1024]; /* error message text */
              short   dadbuf_ind;
              short   rtab_ind;
              short   ovtype_ind;
              short   ov_inde;
```

```
                short   maxrow_ind;
                short   numrow_ind;
                short                   returnCode_ind;
                short                   returnMsg_ind;

EXEC SQL END DECLARE SECTION;

                /* create table */
                EXEC SQL CREATE TABLE xml_order_tab (xmlorder XMLVarchar);

                /* initialize host variable and indicators */
                strcpy(collection,"sales_ord");
                strcpy(result_tab,"xml_order_tab");
                override[0] = '\0';
                overrideType = NO_OVERRIDE;
                max_row = 500;
                num_row = 0;
                returnCode = 0;
                msg_txt[0] = '\0';
                collection_ind = 0;
                rtab_ind = 0;
                ov_ind = -1;
                ovtype_ind = 0;
                maxrow_ind = 0;
                numrow_ind = -1;
                returnCode_ind = -1;
                returnMsg_ind = -1;

                /* Call the store procedure */
                EXEC SQL CALL dxxRetrieve(:collection:collection_ind;
                       :result_tab:rtab_ind,
                       :overrideType:ovtype_ind,:override:ov_ind,
                       :max_row:maxrow_ind,:num_row:numrow_ind,
                                :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

Procedure di memorizzazione di scomposizione

Le procedure memorizzate di scomposizione dxxInsertXML() e dxxShredXML() vengono utilizzate per scomporre i documenti XML in entrata e per memorizzare i dati nelle tabelle nuove o esistenti del database. La procedura memorizzata dxxInsertXML() rileva un nome di raccolta XML abilitata come input. La procedura memorizzata dxxShredXML() rileva un file DAD come input; non richiede una raccolta XML abilitata.

dxxShredXML()

Scopo

La procedura memorizzata dxxShredXML() è associata alla procedura memorizzata dxxGenXML(). Per utilizzare la procedura memorizzata dxxShredXML(), è necessario che esistano tutte le tabelle specificate nel file DAD e che tutte le colonne e i relativi tipi di dati in esso specificati siano congruenti con le tabelle esistenti. La procedura memorizzata dxxShredXML() non richiede alcuna relazione di chiavi primarie-esterne tra le tabelle di collegamento create da XML Extender durante il processo di abilitazione raccolta. Tuttavia, è necessario che le colonne con condizione di collegamento specificate in RDB_node della radice element_node siano presenti nelle tabelle.

```
dxxShredXML(CLOB(100K)    DAD,            /* input */
            CLOB(1M)       xmlobj,           /* input */
            long           returnCode,       /* output */
            varchar(1024)  returnMsg)        /* output */
```

Parametri

Tabella 49. Parametri di dxxShredXML()

Parametro	Descrizione	Parametro IN/OUT
DAD	Il CLOB contenente il file DAD.	IN
xmlobj	Un oggetto documento XML di tipo XMLCLOB.	IN
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

Esempi

Di seguito è riportato un esempio di un richiamo di dxxShredXML().

```
#include "dxx.h"
#include "dxxrc.h"

EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
             SQL TYPE is CLOB dad;           /* DAD*/
             SQL TYPE is CLOB_FILE  dadFile; /* DAD file*/
             SQL TYPE is CLOB xmlDoc;        /* input XML document */
             SQL TYPE is CLOB_FILE xmlFile;  /* input XMLfile */
             long                  returnCode;      /* error code */
             char                  returnMsg[1024]; /* error message text */
             short       dad_ind;
             short                 xmlDoc_ind;
             short                 returnCode_ind;
             short                 returnMsg_ind;
EXEC SQL END DECLARE SECTION;

             /* initialize host variable and indicators */
             strcpy(dadFile.name,"c:\dxx\samples\dad\getstart_xcollection.dad");
             dadFile.name_length=strlen("c:\dxx\samples\dad\getstart_xcollection.dad");
             dadFile.file_option=SQL_FILE_READ;
             strcpy(xmlFile.name,"c:\dxx\samples\cmd\getstart.xml");
             xmlFile.name_length=strlen("c:\dxx\samples\cmd\getstart.xml");
             xmlFile.file_option=SQL_FILE_READ;
             SQL EXEC VALUES (:dadFile) INTO :dad;
             SQL EXEC VALUES (:xmlFile) INTO :xmlDoc;
             returnCode = 0;
```

```
returnMsg[0] = '\0';
dad_ind = 0;
xmlDoc_ind = 0;
returnCode_ind = -1;
returnMsg_ind = -1;

/* Call the store procedure */
EXEC SQL CALL db2xml.dxxShredXML(:dad:dad_ind;
                :xmlDoc:xmlDoc_ind,
                :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

dxxInsertXML()

Scopo

Rileva due parametri di input, il nome di una raccolta XML abilitata e il documento XML da scomporre e restituisce due parametri di output, un codice di ritorno e un messaggio di errore.

```
dxxInsertXML(char(collectionName) collectionName,   /* input */
             CLOB(1M)       xmlobj,           /* input */
             long           returnCode,       /* output */
             varchar(1024)  returnMsg)        /* output */
```

Parametri

Tabella 50. Parametri dxxInsertXML()

Parametro	Descrizione	Parametro IN/OUT
collectionName	Il nome di una raccolta XML abilitata	IN
xmlobj	Un oggetto documento XML di tipo CLOB.	IN
returnCode	Il codice di ritorno della procedura memorizzata.	OUT
returnMsg	Il testo del messaggio restituito in caso di errore.	OUT

Esempi

Nel seguente esempio, la chiamata dxxInsertXML() scompone il documento XML di input `e:\xml\order1.xml` e inserisce i dati nelle tabelle della raccolta SALES_ORDER a seconda dell'associazione specificata nel file DAD con la quale è stata abilitata.

```
#include "dxx.h"
#include "dxxrc.h"

EXEC SQL INCLUDE SQLCA;
EXEC SQL BEGIN DECLARE SECTION;
             char                  collection[64];  /* name of an XML collection */
             SQL TYPE is CLOB_FILE xmlDoc;          /* input XML document */
             long                  returnCode;      /* error code */
             char                  returnMsg[1024]; /* error message text */
             short                 collection_ind;
             short                 xmlDoc_ind;
             short                 returnCode_ind;
             short                 returnMsg_ind;
EXEC SQL END DECLARE SECTION;

             /* initialize host variable and indicators */
             strcpy(collection,"sales_ord")
             strcpy(xmlobj.name,"c:\dxx\samples\cmd\getstart.xml");
             xmlobj.name_length=strlen("c:\dxx\samples\cmd\getstart.xml");
             xmlobj.file_option=SQL_FILE_READ;
             returnCode = 0;
             returnMsg[0] = '\0';
             collection_ind = 0;
             xmlobj_ind = 0;
             returnCode_ind = -1;
             returnMsg_ind = -1;

             /* Call the store procedure */
             EXEC SQL CALL db2xml.dxxInsertXML(:collection:collection_ind;
                           :xmlobj:xmlobj_ind,
                           :returnCode:returnCode_ind,:returnMsg:returnMsg_ind);
```

Capitolo 11. Tabelle di supporto gestione

Quando si abilita un database, vengono create una tabella di riferimento DTD, DTD_REF, e una tabella XML_USAGE. La tabella DTD_REF contiene informazioni relative a tutti i DTD. La tabella XML_USAGE memorizza le informazioni comuni relative a ciascuna colonna abilitata per XML.

I limiti dei parametri elencati nelle tabelle di supporto vengono trattati in "Appendice D. Limiti di XML Extender" a pagina 241.

Tabella di riferimento DTD

XML Extender inoltre fornisce un magazzino DTD XML. Quando si abilita per XML un database, viene creata una tabella di riferimento DTD, DTD_REF. Ciascuna riga di questa tabella rappresenta una DTD con informazioni sui metadati aggiuntive. Gli utenti possono accedere a questa tabella e inserire le relative DTD. Le DTD della tabella DTD_REF vengono utilizzate per convalidare documenti XML e per consentire alle applicazioni la definizione di un file DAD. Il nome schema della tabella è db2xml. Una tabella DTD_REF può presenta le colonne riportate in Tabella 51.

Tabella 51. tabella DTD_REF

Nome colonna	Tipo di dati	Descrizione
DTDID	VARCHAR(128)	Chiave primaria (univoca e non nulla). Utilizzata per identificare la DTD. Quando viene specificata la chiave primaria nel file DAD, questo file deve rispettare lo schema definito dalla DTD.
CONTENT	XMLCLOB	Il contenuto della DTD.
USAGE_COUNT	INTEGER	Il numero di colonne e raccolte XML del database che utilizzano questa DTD per definire i relativi file DAD.
AUTHOR	VARCHAR(128)	L'autore della DTD, informazione facoltativa.
CREATOR	VARCHAR(128)	L'ID utente che esegue il primo inserimento. La colonna CREATOR è facoltativa.
UPDATOR	VARCHAR(128)	L'ID utente che esegue l'ultimo aggiornamento. La colonna UPDATOR è facoltativa.

Restrizione: la DTD può essere modificata dall'applicazione solo quando USAGE_COUNT è uguale a zero.

Tabella di utilizza XML

Memorizza le informazioni comuni per ciascuna colonna abilitata per XML. Il nome schema della tabella XML_USAGE è db2xml e la relativa chiave primaria è (*table_name*, *col_name*). Una tabella XML_USAGE viene creata quando si abilita il database con le colonne elencate in Tabella 52 a pagina 194.

Tabella 52. tabella XML_USAGE

Nome colonna	Descrizione
table_schema	Per la colonna XML, il nome di schema della tabella utente che contiene una colonna XML. Per la raccolta XML, il valore di ″DXX_COLL″ come nome di schema predefinito.
table_name	Per la colonna XML, il nome della tabella utente che contiene una colonna XML. Per la raccolta XML, il valore ″DXX_COLLECTION,″ che identifica l'entità come raccolta.
col_name	Il nome della colonna o della raccolta XML. Fa parte della chiave composta insieme a table_name.
DTDID	L'ID della DTD nella tabella DTD_REF per la definizione del file DAD. La chiave esterna.
DAD	Il contenuto del file DAD associato alla colonna.
default_view	Memorizza il nome della vista predefinita se ne esiste una.
trigger_suffix	Non NULL. Per i nomi di trigger univoci.
Convalida	1 per Sì, 0 per No.
access_mode	1 per raccolta XML, 0 per colonna XML

Restrizione: la DTD può essere modificata dall'applicazione solo quando USAGE_COUNT è uguale a zero.

Capitolo 12. Informazioni diagnostiche

Tutte le istruzioni SQL incorporate e le chiamate CLI (Command Line Interface) del programma, comprese quelle che richiamano le UDF XML Extender, generano dei codici che indicano se l'istruzione SQL incorporata o la chiamata CLI è stata eseguita correttamente.

Il programma è in grado di richiamare le informazioni che integrano questi codici. Queste comprendono le informazioni SQLSTATE e i messaggi di errore. E' possibile utilizzare queste informazioni di diagnostica per isolare e risolvere i problemi nel programma.

Talvolta l'origine di un problema non è facilmente diagnosticabile. In questi casi, per isolare e risolvere il problema, potrebbe essere necessario fornire informazioni al servizio di assistenza. XML Extender fornisce una funzione di traccia che registra l'attività di XML Extender. Le informazioni di traccia possono essere un importante aiuto per l'assistenza tecnica IBM. E' consigliabile utilizzare la funzione di traccia solo seguendo le istruzioni fornite dal supporto di assistenza IBM.

Questo capitolo descrive come accedere a queste informazioni di diagnostica. Esso descrive:

- Come gestire i codici di ritorno UDF XML Extender
- Come controllare la traccia

Inoltre elenca e descrive i codici SQLSTATE e i messaggio di errore che potrebbero essere restituiti da XML Extender.

Gestione dei codici di ritorno UDF

Le istruzioni SQL incorporate restituiscono i codici nei campi SQLCODE, SQLWARN e SQLSTATE della struttura SQLCA. Questa struttura viene definita in un file SQLCA INCLUDE. Per ulteriori informazioni sulla struttura SQLCA e sul file SQLCA INCLUDE, consultare il manuale *DB2 Application Development Guide*.

Le chiamate CLI DB2 restituiscono i valori SQLCODE e SQLSTATE che possono essere richiamati utilizzando la funzione SQLError. Per ulteriori dettagli su come richiamare le informazioni sugli errori con la funzione SQLError, consultare il manuale *CLI Guide and Reference*.

Un valore SQLCODE 0 indica che l'istruzione è stata eseguita correttamente (con possibili condizioni di avvertenza). Un valore SQLCODE positivo indica che l'istruzione è stata eseguita con esito positivo ma con un'avvertenza. Le istruzioni SQL incorporate restituiscono informazioni sull'avvertenza associata a 0 o sui valori SQLCODE positivi nel campo SQLWARN. Un valore SQLCODE negativo indica che si è verificato un errore.

Il DB2 associa un messaggio a ciascun valore SQLCODE. Se una UDF XML Extender rileva una condizione di errore o avvertenza, trasmette le relative informazioni al DB2 perché vengano incluse nel messaggio SQLCODE.

I valori SQLSTATE contengono i codici che integrano i messaggi SQLCODE. Consultare “Codici SQLSTATE” per la descrizione di tutti i codici SQLSTATE restituiti da XML Extender.

Le istruzioni SQL incorporate e le chiamate CLI DB2 che richiamano le UDF XML Extender potrebbero restituire messaggi SQLCODE e valori SQLSTATE univoci per queste UDF, tuttavia DB2 restituisce questi valori seguendo la stessa procedura utilizzata per le altre istruzioni SQL incorporate o per le altre chiamate CLI DB2. Quindi il modo di accedere a questi valori è uguale a quello delle istruzioni SQL incorporate o delle chiamate CLI DB2 che non avviano le UDF DB2 XML Extender.

Consultare “Codici SQLSTATE” per i valori SQLSTATE e per il numero dei messaggi associati che possono essere restituiti da XML Extender Consultare “Messaggi” a pagina 200 per informazioni su ciascun messaggio.

Gestione dei codici di ritorno delle procedure memorizzate

XML Extender fornisce i codici di ritorno di supporto nella risoluzione dei problemi relativi alle procedure memorizzate. Quando si riceve un codice di ritorno da una procedura memorizzata, controllare il seguente file che associa il codice di ritorno al numero di un messaggio di errore XML Extender e alla costante simbolica.

```
DXX_INSTALL/include/dxxrc.h
```

E’ possibile consultare i numeri dei messaggi di errore nella sezione “Messaggi” a pagina 200 e utilizzare le informazioni descritte nella spiegazione.

Codici SQLSTATE

La Tabella 53 elenca e descrive i valori SQLSTATE restituiti da XML Extender. La descrizione di ciascun valore SQLSTATE comprende la sua rappresentazione simbolica. La tabella elenca anche i numeri di messaggio associati a ciascun valore SQLSTATE. Consultare “Messaggi” a pagina 200 per informazioni su ciascun messaggio.

Tabella 53. Codici SQLSTATE e numeri di messaggio associati

SQLSTATE	Num. messaggio	Descrizione
00000	DXX*nnnn*I	Non si è verificato alcun errore.
01HX0	DXXD003W	L’elemento o l’attributo specificato nell’espressione del percorso non esiste nel documento XML.
38X00	DXXC000E	XML Extender non è riuscito ad aprire il file specificato.
38X01	DXXA072E	XML Extender ha tentato di eseguire automaticamente il bind del database prima di abilitarlo.
	DXXC001E	XML Extender non è riuscito a trovare il file specificato.
38X02	DXXC002E	XML Extender non è riuscito a leggere i dati del file specificato.
38X03	DXXC003E	XML Extender non è riuscito a scrivere i dati nel file specificato.

Tabella 53. Codici SQLSTATE e numeri di messaggio associati (Continua)

SQLSTATE	Num. messaggio	Descrizione
	DXXC011E	XML Extender non è riuscito a scrivere i dati nel file di controllo della traccia.
38X04	DXXC004E	XML Extender non è riuscito a utilizzare l'indicatore di posizione specificato.
38X05	DXXC005E	La dimensione file è maggiore della dimensione XMLVarchar e XML Extender non è riuscito a importare tutti i dati dal file.
38X06	DXXC006E	La dimensione file è maggiore della dimensione XMLCLOB e XML Extender non è riuscito a importare tutti i dati dal file.
38X07	DXXC007E	Il numero di byte nell'indicatore di posizione LOB non corrisponde alla dimensione del file.
38X08	DXXD001E	Una funzione di estrazione scalare ha utilizzato un percorso di ubicazione che ricorre più volte. Una funzione scalare può utilizzare solo un percorso di ubicazione che non abbia ricorrenze multiple.
38X09	DXXD002E	L'espressione del percorso presenta errori di sintassi.
38X10	DXXG002E	XML Extender non è riuscito ad assegnare la memoria dal sistema operativo.
38X11	DXXA009E	Questa procedura memorizzata viene utilizzata solo per la colonna XML.
38X12	DXXA010E	Durante l'abilitazione della colonna, XML Extender non è riuscito a trovare l'ID DTD specificato per DTD nel file DAD.
38X14	DXXD000E	Si è tentato di memorizzare un documento non valido in una tabella. La convalida non ha avuto esito positivo.
38X15	DXXA056E	L'elemento di convalida nel file DAD (document access definition) manca o non è valido.
	DXXA057E	L'attributo nome di una tabella laterale nel file DAD (document access definition) manca o non è valido.
	DXXA058E	L'attributo nome di una colonna nel file DAD (document access definition) manca o non è valido.

Tabella 53. Codici SQLSTATE e numeri di messaggio associati (Continua)

SQLSTATE	Num. messaggio	Descrizione
	DXXA059E	L'attributo tipo di una colonna nel file DAD (document access definition) manca o non è valido.
	DXXA060E	L'attributo percorso di una colonna nel file DAD (document access definition) manca o non è valido.
	DXXA061E	L'attributo multi_occurrence di una colonna nel file DAD (document access definition) manca o non è valido.
	DXXQ000E	Manca un elemento obbligatorio nel file DAD (document access definition).
38X16	DXXG004E	E' stato inoltrato un valore nullo per un parametro obbligatorio a una procedura memorizzata XML.
38X17	DXXQ001E	L'istruzione SQL nel file DAD (document access definition) o quella di sostituzione non è valida. E' necessaria un'istruzione SELECT per la creazione di documenti XML.
38X18	DXXG001E	XML Extender ha rilevato un errore interno.
	DXXG006E	XML Extender ha rilevato un errore interno durante l'utilizzo di CLI.
38X19	DXXQ002E	La memoria o lo spazio su disco del sistema sta esaurendo. Non è disponibile spazio sufficiente per contenere i documenti XML derivanti.
38X20	DXXQ003W	L'interrogazione SQL definita dall'utente genera un numero di documenti superiore al numero massimo specificato. Viene restituito solo il numero di documenti specificato.
38X21	DXXQ004E	La colonna specificata non è una delle colonne comprese nei risultati dell'interrogazione SQL.
38X22	DXXQ005E	L'associazione dell'interrogazione SQL a XML è errata.
38X23	DXXQ006E	L'elemento attribute_node nel file DAD (document access definition) non presenta un attributo nome.
38X24	DXXQ007E	L'elemento attribute_node nel file DAD (document access definition) non presenta un elemento colonna o un RDB_node.
38X25	DXXQ008E	L'elemento text_node nel file DAD (document access definition) non presenta un elemento colonna.

Tabella 53. Codici SQLSTATE e numeri di messaggio associati (Continua)

SQLSTATE	Num. messaggio	Descrizione
38X26	DXXQ009E	Impossibile trovare la tabella dei risultati specificata nel catalogo di sistema.
38X27	DXXQ010E	E' necessario che RDB_node di attribute_node o di text_node presenti una tabella.
	DXXQ011E	E' necessario che RDB_node di attribute_node o di text_node presenti una colonna.
	DXXQ017E	Il documento XML generato da XML Extender è troppo grande per rientrare nella colonna della tabella dei risultati.
38X28	DXXQ012E	XML Extender non è riuscito a trovare l'elemento previsto durante l'elaborazione della DAD.
	DXXQ016E	E' necessario definire tutte le tabelle nel RDB_node dell'elemento iniziale del file DAD (document access definition). Le tabelle dell'elemento secondario devono corrispondere alle tabelle definite nell'elemento iniziale. Il nome tabella in questo RDB_node non è ubicato nell'elemento iniziale.
38X29	DXXQ013E	La tabella o la colonna dell'elemento deve presentare un nome nel file DAD (document access definition).
	DXXQ015E	La condizione dell'elemento condizione nel file DAD (document access definition) presenta un formato errato.
38X30	DXXQ014E	L'elemento element_node nel file DAD (document access definition) non presenta un attributo nome.
	DXXQ018E	Manca la clausola ORDER BY nell'istruzione SQL di un file DAD (document access definition) che associa SQL a XML.
38X31	DXXQ019E	L'elemento objids non presenta alcun elemento colonna nel file DAD (document access definition) che associa SQL a XML.
38X36	DXXA073E	Il bind del database non era stato eseguito quando l'utente ha tentato di abilitarlo.
38X37	DXXG007E	La locale del sistema operativo non corrisponde alla code page DB2.
38X38	DXXG008E	Impossibile rilevare la locale del sistema operativo del server nella tabella delle code page.

Tabella 53. Codici SQLSTATE e numeri di messaggio associati (Continua)

SQLSTATE	Num. messaggio	Descrizione
38x33	DXXG005E	Il parametro non è supportato in questo release, verrà supportato nel release successivo.
38x34	DXXG000E	E' stato specificato un nome file non corretto.

Messaggi

XML Extender fornisce messaggi di errore di supporto nella risoluzione dei problemi.

Messaggi di errore

XML Extender genera i seguenti messaggi quando completa un'operazione o rileva un errore.

DXXA000I **Abilitazione della colonna** <*nome colonna*> **in corso. Attendere.**

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA001S **Si è verificato un errore imprevisto nel build** <*ID build*>**, file** <*nome file*>**, riga** <*numero riga*>**.**

Spiegazione: Si è verificato un errore non previsto.

Risposta dell'utente: Se l'errore persiste, contattare l'assistenza tecnica. Quando l'errore viene notificato, includere l'intero testo del messaggio, il file di traccia e la spiegazione su come riprodurre il problema.

DXXA002I **Collegamento al database** <*database*> **in corso.**

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA003E **Impossibile collegarsi al database** <*database*>**.**

Spiegazione: E' possibile che il database specificato non esista o sia danneggiato.

Risposta dell'utente:

1. Accertarsi che il database venga specificato correttamente.
2. Verificare che il database esista e sia accessibile.
3. Stabilire se il database è danneggiato. In tal caso, richiedere al responsabile del database di ripristinarlo da un backup.

DXXA004E **Impossibile abilitare il database** <*database*>**.**

Spiegazione: E' possibile che il database sia già abilitato o danneggiato.

Risposta dell'utente:

1. Stabilire se il database è già abilitato.
2. Stabilire se il database è danneggiato. In tal caso, richiedere al responsabile del database di ripristinarlo da un backup.

DXXA005I **Abilitazione del database** <*database*> **in corso. Attendere.**

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA006I **Il database** <*database*> **è stato abilitato correttamente.**

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA007E **Impossibile disabilitare il database** <*database*>**.**

Spiegazione: Il database non può essere disabilitato da XML Extender se contiene colonne o raccolte XML.

Risposta dell'utente: Eseguire il backup dei dati importanti, disabilitare le colonne o le raccolte XML e aggiornare o cancellare le tabelle in modo da eliminare tutti i tipi di dati XML dal database.

DXXA008I Disabilitazione della colonna <*nome colonna*> **in corso. Attendere.**

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA009E La tag Xcolumn non è specificata nel file DAD.

Spiegazione: Questa procedura memorizzata viene utilizzata solo per la colonna XML.

Risposta dell'utente: Accertarsi che la tag Xcolumn venga specificata correttamente nel file DAD.

DXXA010E Impossibile trovare l'ID DTD <*id dtd*>**.**

Spiegazione: Durante l'abilitazione della colonna, XML Extender non è riuscito a trovare l'ID DTD specificato per DTD nel file DAD.

Risposta dell'utente: Accertarsi che venga specificato il valore corretto per l'ID DTD nel file DAD.

DXXA011E L'inserimento nella tabella DB2XML.XML_USAGE non ha avuto esito positivo.

Spiegazione: Durante l'abilitazione della colonna, XML Extender non è riuscito ad inserire un record nella tabella DB2XML.XML_USAGE.

Risposta dell'utente: Accertarsi che la tabella DB2XML.XML_USAGE esista e che non contenga già un record con lo stesso nome.

DXXA012E Il tentativo di aggiornamento della tabella DB2XML.DTD_REF non ha avuto esito positivo.

Spiegazione: Durante l'abilitazione della colonna, XML Extender non è riuscito ad aggiornare la tabella DB2XML.DTD_REF.

Risposta dell'utente: Accertarsi che la tabella DB2XML.DTD_REF esista. Verificare se la tabella è danneggiata o se l'ID utente di gestione dispone dell'autorizzazione appropriata per aggiornare la tabella.

DXXA013E Il tentativo di modifica della tabella <*nome tabella*> **non ha avuto esito positivo.**

Spiegazione: Durante l'abilitazione della colonna, XML Extender non è riuscito a modificare la tabella specificata.

Risposta dell'utente: Controllare i privilegi necessari per modificare la tabella.

DXXA014E L'ID root specificato: <*id root*> **non è una chiave primaria singola della tabella** <*nome tabella*>**.**

Spiegazione: L'ID root specificato non è una chiave o non è una chiave singola della tabella *nome tabella*.

Risposta dell'utente: Accertarsi che l'ID root specificato rappresenti la chiave primaria singola della tabella.

DXXA015E La colonna DXXROOT_ID esiste già nella tabella <*nome tabella*>**.**

Spiegazione: La colonna DXXROOT_ID esiste, ma non è stata creata da XML Extender.

Risposta dell'utente: Specificare una colonna primaria per l'opzione ID root durante l'abilitazione della colonna, utilizzando un nome colonna differente.

DXXA016E La tabella di input <*nome tabella*> **non esiste.**

Spiegazione: XML Extender non è riuscito a trovare la tabella specificata nel catalogo di sistema.

Risposta dell'utente: Assicurarsi che la tabella esista nel database e che sia stata specificata correttamente.

DXXA017E La colonna di input <*nome colonna*> **non esiste nella tabella specificata** <*nome tabella*>**.**

Spiegazione: XML Extender non è riuscito a trovare la colonna nel catalogo di sistema.

Risposta dell'utente: Accertarsi che una tabella utente contenga la colonna.

DXXA018E La colonna specificata non è abilitata per i dati XML.

Spiegazione: Durante la disabilitazione della colonna, XML Extender non è riuscito a trovare la colonna nella tabella DB2XML.XML_USAGE indicante che la colonna non è abilitata. Se la colonna non è abilitata per XML, non occorre disabilitarla.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA019E Un parametro di input richiesto per abilitare la colonna è nullo.

Spiegazione: Un parametro di input per la procedura memorizzata enable_column() è nullo.

Risposta dell'utente: Controllare tutti i parametri di input per la procedura memorizzata enable_column().

DXXA020E Impossibile trovare le colonne nella tabella <*nome tabella*>**.**

Spiegazione: Durante la creazione della vista predefinita, XML Extender non è riuscito a trovare le colonne nella tabella specificata.

Risposta dell'utente: Accertarsi che i nomi tabella e colonna vengano specificati correttamente.

DXXA021E Impossibile creare la vista predefinita <*vista predefinita*>**.**

Spiegazione: Durante l'abilitazione di una colonna, XML Extender non è riuscito a creare la vista specificata.

Risposta dell'utente: Accertarsi che il nome della vista predefinita sia univoco. Se esiste già una vista con lo stesso nome, specificare un nome univoco per la vista predefinita.

DXXA022I La colonna <*nome colonna*> **è stata abilitata.**

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna risposta.

DXXA023E Impossibile trovare il file DAD.

Spiegazione: Durante la disabilitazione di una colonna, XML Extender non è riuscito a trovare il file DAD.

Risposta dell'utente: Accertarsi che vengano specificati i nomi database, tabella e colonna corretti.

DXXA024E XML Extender ha rilevato un errore interno durante l'accesso alle tabelle di catalogo del sistema.

Spiegazione: XML Extender non è riuscito ad accedere alla tabella di catalogo del sistema.

Risposta dell'utente: Accertarsi che il database risulti in uno stato stabile.

DXXA025E Impossibile cancellare la vista predefinita <*vista predefinita*>**.**

Spiegazione: Durante la disabilitazione di una colonna, XML Extender non è riuscito a cancellare la vista predefinita.

Risposta dell'utente: Accertarsi che l'ID utente di gestione per XML Extender disponga dei privilegi necessari per cancellare la vista predefinita.

DXXA026E Impossibile cancellare la tabella laterale <*tabella laterale*>**.**

Spiegazione: Durante la disabilitazione di una colonna, XML Extender non è riuscito a cancellare la tabella specificata.

Risposta dell'utente: Accertarsi che l'ID utente di gestione per XML Extender disponga dei privilegi necessari per cancellare la tabella laterale.

DXXA027E Non è stato possibile disabilitare la colonna.

Spiegazione: XML Extender non è riuscito a disabilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA028E Non è stato possibile disabilitare la colonna.

Spiegazione: XML Extender non è riuscito a disabilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA029E Non è stato possibile disabilitare la colonna.

Spiegazione: XML Extender non è riuscito a disabilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA030E Non è stato possibile disabilitare la colonna.

Spiegazione: XML Extender non è riuscito a disabilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA031E Impossibile reimpostare il valore della colonna DXXROOT_ID nella tabella applicativa su NULL.

Spiegazione: Durante la disabilitazione di una colonna, XML Extender non è riuscito a impostare il valore DXXROOT_ID su NULL nella tabella applicativa.

Risposta dell'utente: Accertarsi che l'ID utente di gestione per XML Extender disponga dei privilegi necessari per modificare la tabella applicativa.

DXXA032E La riduzione del valore USAGE_COUNT nella tabella DB2XML.XML_USAGE non ha avuto esito positivo.

Spiegazione: Durante la disabilitazione della colonna, XML Extender non è riuscito a ridurre il valore della colonna USAGE_COUNT.

Risposta dell'utente: Accertarsi che la tabella DB2XML.XML_USAGE esista e che l'ID utente di gestione per XML Extender disponga dei privilegi necessari per aggiornarla.

DXXA033E Il tentativo di cancellazione di una riga dalla tabella DB2XML.XML_USAGE non ha avuto esito positivo.

Spiegazione: Durante la disabilitazione di una colonna, XML Extender non è riuscito a cancellare la riga corrispondente nella tabella DB2XML.XML_USAGE.

Risposta dell'utente: Accertarsi che la tabella DB2XML.XML_USAGE esista e che l'ID utente di gestione per XML Extender disponga dei privilegi necessari per aggiornarla.

DXXA034I XML Extender ha disabilitato la colonna <*nome colonna*> **con esito positivo.**

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA035I XML Extender sta disabilitando il database <*database*>**. Attendere.**

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA036I XML Extender ha disabilitato il database <*database*> **con esito positivo.**

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA037E Il nome del table space specificato supera i 18 caratteri.

Spiegazione: Il nome del table space non può superare i 18 caratteri alfanumerici.

Risposta dell'utente: Specificare un nome inferiore a 18 caratteri.

DXXA038E Il nome vista predefinito specificato supera i 18 caratteri.

Spiegazione: Il nome della vista predefinita non può superare i 18 caratteri alfanumerici.

Risposta dell'utente: Specificare un nome inferiore a 18 caratteri.

DXXA039E Il nome ROOT_ID specificato supera i 18 caratteri.

Spiegazione: Il nome ROOT_ID non può superare i 18 caratteri alfanumerici.

Risposta dell'utente: Specificare un nome inferiore a 18 caratteri.

DXXA046E Impossibile creare la tabella laterale <*tabella laterale*>**.**

Spiegazione: Durante l'abilitazione di una colonna, XML Extender non è riuscito a creare la tabella laterale specificata.

Risposta dell'utente: Accertarsi che l'ID utente di gestione per XML Extender disponga dei privilegi necessari per creare la tabella laterale.

DXXA047E Non è stato possibile abilitare la colonna.

Spiegazione: XML Extender non è riuscito ad abilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA048E Non è stato possibile abilitare la colonna.

Spiegazione: XML Extender non è riuscito ad abilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA049E Non è stato possibile abilitare la colonna.

Spiegazione: XML Extender non è riuscito ad abilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA050E Non è stato possibile abilitare la colonna.

Spiegazione: XML Extender non è riuscito ad abilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA051E Non è stato possibile disabilitare la colonna.

Spiegazione: XML Extender non è riuscito a disabilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA052E Non è stato possibile disabilitare la colonna.

Spiegazione: XML Extender non è riuscito a disabilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA053E Non è stato possibile abilitare la colonna.

Spiegazione: XML Extender non è riuscito ad abilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA054E Non è stato possibile abilitare la colonna.

Spiegazione: XML Extender non è riuscito ad abilitare una colonna in quanto un trigger interno non è stato eseguito regolarmente. Cause possibile:

Risposta dell'utente: Utilizzare la funzione di traccia per creare un file di traccia ed eliminare il problema. Se il problema persiste, contattare l'assistenza tecnica e fornire il file di traccia.

DXXA056E Il valore di convalida <*valore convalida*> **nel file DAD non è valido.**

Spiegazione: L'elemento di convalida nel file DAD (document access definition) manca o non è valido.

Risposta dell'utente: Accertarsi che l'elemento di convalida venga specificato correttamente nel file DAD.

DXXA057E Un nome tabella laterale <*nome tabella*> **nel file DAD non è valido.**

Spiegazione: L'attributo nome di una tabella laterale nel file DAD (document access definition) manca o non è valido.

Risposta dell'utente: Accertarsi che l'attributo nome di una tabella laterale venga specificato correttamente nel file DAD.

DXXA058E Un nome colonna <*nome colonna*> **nel file DAD non è valido.**

Spiegazione: L'attributo nome di una colonna nel file DAD (document access definition) manca o non è valido.

Risposta dell'utente: Accertarsi che l'attributo nome di una colonna venga specificato correttamente nel file DAD.

DXXA059E Il tipo <*tipo*> **di colonna** <*nome colonna*> **nel file DAD non è valido.**

Spiegazione: L'attributo tipo di una colonna nel file DAD (document access definition) manca o non è valido.

Risposta dell'utente: Accertarsi che l'attributo tipo di una colonna venga specificato correttamente nel file DAD.

DXXA060E L'attributo percorso <*percorso ubicazione*> **di** <*nome colonna*> **nel file DAD non è valido.**

Spiegazione: L'attributo percorso di una colonna nel file DAD (document access definition) manca o non è valido.

Risposta dell'utente: Accertarsi che l'attributo

percorso di una colonna venga specificato correttamente nel file DAD.

DXXA061E L'attributo multi_occurrence <*multi_occurrence*> di <*nome colonna*> nel file DAD non è valido.

Spiegazione: L'attributo multi_occurrence di una colonna nel file DAD (document access definition) manca o non è valido.

Risposta dell'utente: Accertarsi che l'attributo multi_occurrence di una colonna venga specificato correttamente nel file DAD.

DXXA062E Impossibile richiamare il numero di colonna per <*nome colonna*> nella tabella <*nome tabella*>.

Spiegazione: XML Extender non è riuscito a richiamare il numero di colonna per *nome colonna* nella tabella *nome tabella* dal catalogo di sistema.

Risposta dell'utente: Accertarsi che la tabella applicativa venga definita correttamente.

DXXA063I Abilitazione della raccolta <*nome raccolta*> in corso. Attendere.

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA064I Disabilitazione della raccolta <*nome raccolta*> in corso. Attendere.

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA065E L'operazione di richiamo della procedura memorizzata <*nome procedura*> non ha avuto esito positivo.

Spiegazione: Esaminare la libreria condivisa db2xml e verificare che l'autorizzazione sia corretta.

Risposta dell'utente: Accertarsi che il client disponga dell'autorizzazione appropriata per eseguire la procedura memorizzata.

DXXA066I XML Extender ha disabilitato la raccolta <*nome raccolta*> con esito positivo.

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna risposta.

DXXA067I XML Extender ha abilitato la raccolta <*nome raccolta*> con esito positivo.

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna risposta.

DXXA068I XML Extender ha attivato la funzione di traccia con esito positivo.

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna risposta.

DXXA069I XML Extender ha disattivato la funzione di traccia con esito positivo.

Spiegazione: Si tratta di un messaggio informativo.

Risposta dell'utente: Non è richiesta alcuna risposta.

DXXA070W Il database è già stato abilitato.

Spiegazione: Il comando enable database è stato eseguito sul database abilitato.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA071W Il database è già stato disabilitato.

Spiegazione: Il comando disable database è stato eseguito sul database disabilitato.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXA072E XML Extender non ha trovato i file di bind. Eseguire il bind del database prima di abilitarlo.

Spiegazione: XML Extender ha tentato di eseguire automaticamente il bind del database prima di abilitarlo.

Risposta dell'utente: Eseguire il bind del database prima di abilitarlo.

DXXA073E Non è stato eseguito il bind del database. Eseguire il bind del database prima di abilitarlo.

Spiegazione: Il bind del database non era stato eseguito quando l'utente ha tentato di abilitarlo.

Risposta dell'utente: Eseguire il bind del database prima di abilitarlo.

DXXA074E Tipo di parametro errato. La procedura memorizzata prevede l'utilizzo del parametro STRING.

Spiegazione: La procedura memorizzata prevede l'utilizzo del parametro STRING.

Risposta dell'utente: Specificare STRING come tipo di parametro di input.

DXXA075E Tipo di parametro errato. Il parametro di input deve essere di tipo LONG.

Spiegazione: La procedura memorizzata prevede l'utilizzo del parametro di input di tipo LONG.

Risposta dell'utente: Specificare LONG come tipo di parametro di input.

DXXA076E ID di istanza di traccia XML Extender non valido.

Spiegazione: Impossibile avviare una traccia con l'ID di istanza fornito.

Risposta dell'utente: Accertarsi che l'ID di istanza sia un ID utente AS/400 valido.

DXXC000E Impossibile aprire il file specificato.

Spiegazione: XML Extender non è riuscito ad aprire il file specificato.

Risposta dell'utente: Accertarsi che l'ID utente dell'applicazione disponga delle autorizzazioni di scrittura e lettura sul file.

DXXC001E Il file specificato non è stato trovato.

Spiegazione: XML Extender non è riuscito a trovare il file specificato.

Risposta dell'utente: Accertarsi che il file esista e che il percorso sia stato specificato correttamente.

DXXC002E Impossibile leggere il file.

Spiegazione: XML Extender non è riuscito a leggere i dati del file specificato.

Risposta dell'utente: Accertarsi che l'ID utente dell'applicazione disponga dell'autorizzazione di lettura sul file.

DXXC003E Impossibile scrivere nel file specificato.

Spiegazione: XML Extender non è riuscito a scrivere i dati nel file specificato.

Risposta dell'utente: Accertarsi che l'ID utente dell'applicazione disponga dell'autorizzazione di scrittura sul file.

DXXC004E Impossibile utilizzare l'indicatore di posizione LOB: rc=<*rc indicatore*>**.**

Spiegazione: XML Extender non è riuscito a utilizzare l'indicatore di posizione specificato.

Risposta dell'utente: Accertarsi che l'indicatore di posizione LOB sia stato impostato correttamente.

DXXC005E La dimensione del file di input è maggiore della dimensione di XMLVarchar.

Spiegazione: La dimensione file è maggiore della dimensione XMLVarchar e XML Extender non è riuscito a importare tutti i dati dal file.

Risposta dell'utente: Utilizzare il tipo di colonna XMLCLOB.

DXXC006E Il file di input supera il limite LOB DB2.

Spiegazione: La dimensione file è maggiore della dimensione XMLCLOB e XML Extender non è riuscito a importare tutti i dati dal file.

Risposta dell'utente: Scomporre il file in oggetti più piccoli oppure utilizzare una raccolta XML.

DXXC007E Impossibile richiamare i dati dal file nell'indicatore di posizione LOB.

Spiegazione: Il numero di byte nell'indicatore di posizione LOB non corrisponde alla dimensione del file.

Risposta dell'utente: Accertarsi che l'indicatore di posizione LOB sia stato impostato correttamente.

DXXC008E Impossibile rimuovere il file <*nome file*>**.**

Spiegazione: Il file presenta una violazione di accesso condiviso o è ancora aperto.

Risposta dell'utente: Chiudere il file o arrestare i processi in corso relativi al file. E' possibile che sia necessario arrestare e riavviare il DB2.

DXXC009E Impossibile creare il file nell'indirizzario <*indirizzario*>**.**

Spiegazione: XML Extender non è riuscito a creare un file nell'indirizzario *indirizzario*.

Risposta dell'utente: Accertarsi che l'indirizzario esista, che l'ID utente dell'applicazione disponga delle autorizzazioni sull'indirizzario e che il file system abbia spazio sufficiente per il file.

DXXC010E Errore verificatosi in fase di scrittura nel file <*nome file*>**.**

Spiegazione: Si è verificato un errore durante la scrittura nel file *nome file*.

Risposta dell'utente: Accertarsi che il file system abbia spazio sufficiente per il file.

DXXC011E Impossibile scrivere nel file di controllo traccia.

Spiegazione: XML Extender non è riuscito a scrivere i dati nel file di controllo della traccia.

Risposta dell'utente: Accertarsi che l'ID utente dell'applicazione disponga dell'autorizzazione di scrittura sul file.

DXXC012E Impossibile creare il file temporaneo.

Spiegazione: Non è possibile creare il file nell'indirizzario temporaneo di sistema.

Risposta dell'utente: Accertarsi che l'ID utente dell'applicazione disponga delle autorizzazioni sull'indirizzario temporaneo di sistema e che il file system abbia spazio sufficiente per il file.

DXXD000E Un documento XML non valido è stato rifiutato.

Spiegazione: Si è tentato di memorizzare un documento non valido in una tabella. La convalida non ha avuto esito positivo.

Risposta dell'utente: Controllare il documento e la relativa DTD utilizzando un editor in grado di visualizzare i caratteri non validi invisibili. Per correggere questo errore, disattivare la funzione di convalida nel file DAD.

DXXD001E <*percorso ubicazione*> **ricorre più volte.**

Spiegazione: Una funzione di estrazione scalare ha utilizzato un percorso di ubicazione che ricorre più volte. Una funzione scalare può utilizzare solo un percorso di ubicazione che non abbia ricorrenze multiple.

Risposta dell'utente: Utilizzare una funzione di tabella (e una 's' alla fine del nome della funzione scalare).

DXXD002E Si è verificato un errore di sintassi accanto alla posizione <*posizione*> **nel percorso di ricerca.**

Spiegazione: L'espressione del percorso presenta errori di sintassi.

Risposta dell'utente: Correggere l'argomento del percorso di ricerca dell'interrogazione. Consultare la documentazione per informazioni sulla sintassi corretta.

DXXD003W Percorso non trovato. Viene restituito un valore nullo.

Spiegazione: L'elemento o l'attributo specificato nell'espressione del percorso non esiste nel documento XML.

Risposta dell'utente: Verificare che il percorso specificato sia corretto.

DXXG000E Il nome file <*nome file*> **non è valido.**

Spiegazione: E' stato specificato un nome file non corretto.

Risposta dell'utente: Specificare un nome file corretto e rieseguire l'operazione.

DXXG001E Si è verificato un errore interno nel build <*ID build*>**, file** <*nome file*>**, riga** <*numero riga*>**.**

Spiegazione: XML Extender ha rilevato un errore interno.

Risposta dell'utente: Contattare l'assistenza tecnica. Quando l'errore viene notificato, includere tutti i messaggi, il file di traccia e la spiegazione su come riprodurre l'errore.

DXXG002E Il sistema ha esaurito la memoria.

Spiegazione: XML Extender non è riuscito ad assegnare la memoria dal sistema operativo.

Risposta dell'utente: Chiudere alcune applicazioni e rieseguire l'operazione. Se il problema persiste, consultare la documentazione relativa al proprio sistema operativo. Alcuni sistemi operativi richiedono di riavviare il sistema.

DXXG004E Parametro null non valido.

Spiegazione: E' stato inoltrato un valore nullo per un parametro obbligatorio a una procedura memorizzata XML.

Risposta dell'utente: Controllare tutti i parametri obbligatori nell'elenco degli argomenti per il richiamo della procedura memorizzata.

DXXG005E Parametro non supportato.

Spiegazione: Il parametro non è supportato in questo release, verrà supportato nel release successivo.

Risposta dell'utente: Impostare questo parametro su NULL.

DXXG006E Errore interno CLISTATE=<*clistate*>**, RC=**<*cli_rc*>**, build** <*ID build*>**, file** <*nome file*>**, riga** <*numero riga*> **CLIMSG=**<*messaggio CLI*>**.**

Spiegazione: XML Extender ha rilevato un errore interno durante l'utilizzo di CLI.

Risposta dell'utente: Contattare l'assistenza tecnica. E' possibile che questo errore sia stato causato da un'immissione non corretta. Quando l'errore viene

notificato, includere tutti i messaggi di output, il file di traccia e la spiegazione su come riprodurre l'errore. Se possibile, inviare le DAD, i documenti XML e le definizioni di tabella da applicare.

DXXG007E Locale <*locale*> incongruente con code page DB2 <*code_page*>.

Spiegazione: La locale del sistema operativo non corrisponde alla code page DB2.

Risposta dell'utente: Correggere la locale del sistema operativo del server e riavviare DB2.

DXXG008E Locale <*locale*> non supportata.

Spiegazione: Impossibile rilevare la locale del sistema operativo del server nella tabella delle code page.

Risposta dell'utente: Correggere la locale del sistema operativo del server e riavviare DB2.

DXXQ000E <*Elemento*> non è presente nel file DAD.

Spiegazione: Manca un elemento obbligatorio nel file DAD (document access definition).

Risposta dell'utente: Aggiungere l'elemento mancante al file DAD.

DXXQ001E Istruzione SQL non valida per la creazione di XML.

Spiegazione: L'istruzione SQL nel file DAD (document access definition) o quella di sostituzione non è valida. E' necessaria un'istruzione SELECT per la creazione di documenti XML.

Risposta dell'utente: Correggere l'istruzione SQL.

DXXQ002E Impossibile generare lo spazio di memoria per conservare i documenti XML.

Spiegazione: La memoria o lo spazio su disco del sistema sta esaurendo. Non è disponibile spazio sufficiente per contenere i documenti XML derivanti.

Risposta dell'utente: Limitare il numero di documenti da generare. Ridurre la dimensione di ciascun documento eliminando i nodi di elemento o attributo non necessari dal file DAD.

DXXQ003W Il risultato supera il limite massimo.

Spiegazione: L'interrogazione SQL definita dall'utente genera un numero di documenti superiore al numero massimo specificato. Viene restituito solo il numero di documenti specificato.

Risposta dell'utente: Non è richiesta alcuna azione. Se sono necessari tutti i documenti, specificare zero come numero massimo di documenti.

DXXQ004E La colonna <*nome colonna*> non è compresa nei risultati dell'interrogazione.

Spiegazione: La colonna specificata non è una delle colonne comprese nei risultati dell'interrogazione SQL.

Risposta dell'utente: Modificare il nome colonna specificato nel file DAD in una delle colonne comprese nei risultati dell'interrogazione SQL. Altrimenti, modificare l'interrogazione SQL in modo da includere la colonna specificata nei risultati.

DXXQ004W L'ID DTD non è stato trovato nella DAD.

Spiegazione: Nella DAD, VALIDATION è YES ma l'elemento DTDID non è stato specificato. Non viene eseguito alcun controllo di convalida.

Risposta dell'utente: Non è richiesta alcuna azione. Se occorre eseguire la convalida, specificare l'elemento DTDID nel file DAD.

DXXQ005E Associazione relazionale errata. L'elemento <*nome elemento*> si trova ad un livello più basso della relativa colonna secondaria <*nome colonna*>.

Spiegazione: L'associazione dell'interrogazione SQL a XML è errata.

Risposta dell'utente: Accertarsi che le colonne comprese nei risultati dell'interrogazione SQL siano disposte in un ordine decrescente nella gerarchia di riferimento. Inoltre, verificare che vi sia una chiave candidata come singola colonna per ogni livello. Se questa chiave non è disponibile in una tabella, l'interrogazione ne deve generare una per la tabella che utilizza un'espressione e la funzione integrata DB2 generate_unique().

DXXQ006E Un elemento attribute_node non ha nome.

Spiegazione: L'elemento attribute_node nel file DAD (document access definition) non presenta un attributo nome.

Risposta dell'utente: Accertarsi che per ogni attribute_node sia stato specificato un nome nel file DAD.

DXXQ007E Attribute_node <*nome attributo*> non presenta elementi colonna o RDB_node.

Spiegazione: L'elemento attribute_node nel file DAD (document access definition) non presenta un elemento colonna o un RDB_node.

Risposta dell'utente: Accertarsi che per ogni attribute_node sia stato specificato un elemento colonna o RDB_node nel file DAD.

DXXQ008E Un elemento text_node non presenta elementi di colonna.

Spiegazione: L'elemento text_node nel file DAD (document access definition) non presenta un elemento colonna.

Risposta dell'utente: Accertarsi che per ogni text_node sia stato specificato un elemento colonna nel file DAD.

DXXQ009E La tabella dei risultati <*nome tabella*> **non esiste.**

Spiegazione: Impossibile trovare la tabella dei risultati specificata nel catalogo di sistema.

Risposta dell'utente: Creare una tabella dei risultati prima di richiamare la procedura memorizzata.

DXXQ010E RDB_node di <*nome nodo*> **non presenta una tabella nel file DAD.**

Spiegazione: E' necessario che RDB_node di attribute_node o di text_node presenti una tabella.

Risposta dell'utente: Specificare la tabella di RDB_node per attribute_node o text_node nel file DAD.

DXXQ011E L'elemento RDB_node di <*nome nodo*> **non presenta una colonna nel file DAD.**

Spiegazione: E' necessario che RDB_node di attribute_node o di text_node presenti una colonna.

Risposta dell'utente: Specificare la colonna di RDB_node per attribute_node o text_node nel file DAD.

DXXQ012E Errori in DAD.

Spiegazione: XML Extender non è riuscito a trovare l'elemento previsto durante l'elaborazione della DAD.

Risposta dell'utente: Controllare che DAD sia un documento XML valido e che contenga tutti gli elementi richiesti dalla DTD DAD. Consultare la documentazione XML Extender per la DTD DAD.

DXXQ013E L'elemento colonna o tabella non presenta alcun nome nel file DAD.

Spiegazione: La tabella o la colonna dell'elemento deve presentare un nome nel file DAD (document access definition).

Risposta dell'utente: Specificare il nome dell'elemento tabella o colonna nel file DAD.

DXXQ014E Un elemento element_node non ha nome.

Spiegazione: L'elemento element_node nel file DAD (document access definition) non presenta un attributo nome.

Risposta dell'utente: Accertarsi che per ogni elemento element_node sia stato specificato un nome nel file DAD.

DXXQ015E Il formato della condizione non è valido.

Spiegazione: La condizione dell'elemento condizione nel file DAD (document access definition) presenta un formato errato.

Risposta dell'utente: Accertarsi che il formato della condizione sia valido.

DXXQ016E Il nome tabella in questo RDB_node non è definito nell'elemento iniziale del file DAD.

Spiegazione: E' necessario definire tutte le tabelle nel RDB_node dell'elemento iniziale del file DAD (document access definition). Le tabelle dell'elemento secondario devono corrispondere alle tabelle definite nell'elemento iniziale. Il nome tabella in questo RDB_node non è ubicato nell'elemento iniziale.

Risposta dell'utente: Accertarsi che la tabella del nodo RDB venga definita nell'elemento iniziale del file DAD.

DXXQ017E La colonna nella tabella dei risultati <*nome tabella*> **è troppo piccola.**

Spiegazione: Il documento XML generato da XML Extender è troppo grande per rientrare nella colonna della tabella dei risultati.

Risposta dell'utente: Cancellare la tabella dei risultati. Creare un'altra tabella dei risultati con una colonna più grande. Rieseguire la procedura memorizzata.

DXXQ018E La clausola ORDER BY non è presente nell'istruzione SQL.

Spiegazione: Manca la clausola ORDER BY nell'istruzione SQL di un file DAD (document access definition) che associa SQL a XML.

Risposta dell'utente: Editare il file DAD. Aggiungere una clausola ORDER BY che contenga le colonne di identificazione entità.

DXXQ019E L'elemento objids non presenta alcun elemento colonna nel file DAD.

Spiegazione: L'elemento objids non presenta alcun elemento colonna nel file DAD (document access definition) che associa SQL a XML.

Risposta dell'utente: Editare il file DAD. Aggiungere le colonne chiave come elementi secondari dell'elemento objids.

DXXQ020I XML è stato generato con esito positivo.

Spiegazione: I documenti XML richiesti sono stati generati correttamente dal database.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXQ021E La tabella <*nome tabella*> **non contiene la colonna** <*nome colonna*>**.**

Spiegazione: La tabella non contiene la colonna specificata nel database.

Risposta dell'utente: Specificare un altro nome colonna nel file DAD o aggiungere la colonna specificata nel database della tabella.

DXXQ022E Il tipo di colonna <*nome colonna*> **di** <*nome tabella*> **deve essere** <*nome tipo*>**.**

Spiegazione: Il tipo di colonna non è valido.

Risposta dell'utente: Correggere il tipo di colonna nella DAD.

DXXQ023E La lunghezza della colonna <*nome colonna*> **di** <*nome tabella*> **non può superare il limite consentito** <*limite lunghezza*>**.**

Spiegazione: La lunghezza definita per la colonna nella DAD supera il limite consentito.

Risposta dell'utente: Correggere la lunghezza della colonna nella DAD.

DXXQ024E Impossibile creare la tabella <*nome tabella*>**.**

Spiegazione: La tabella specificata non può essere creata.

Risposta dell'utente: Accertarsi che l'ID utente che sta creando la tabella disponga dell'autorizzazione appropriata.

DXXQ025I Scomposizione XML eseguita correttamente.

Spiegazione: Un documento XML è stato scomposto e memorizzato in una raccolta con esito positivo.

Risposta dell'utente: Non è richiesta alcuna azione.

DXXQ026E Impossibile caricare i dati XML <*nome xml*> **nella colonna** <*nome colonna*>**.**

Spiegazione: La dimensione dei dati del documento XML è maggiore di quella della colonna specificata.

Risposta dell'utente: Aumentare la lunghezza della colonna utilizzando l'istruzione ALTER TABLE oppure ridurre la dimensione dei dati editando il documento XML.

DXXQ028E Impossibile trovare la raccolta <*nome raccolta*> **nella tabella XML_USAGE.**

Spiegazione: Non è possibile trovare un record per la raccolta nella tabella XML_USAGE.

Risposta dell'utente: Verificare che la raccolta sia stata abilitata.

DXXQ029E Impossibile trovare DAD nella tabella XML_USAGE per la raccolta <*nome raccolta*>**.**

Spiegazione: Non è possibile trovare un record DAD per la raccolta nella tabella XML_USAGE.

Risposta dell'utente: Verificare che la raccolta sia stata abilitata correttamente.

DXXQ030E Sintassi del valore XML_override errata.

Spiegazione: Il valore XML_override non è stato specificato correttamente nella procedura memorizzata.

Risposta dell'utente: Accertarsi che la sintassi del valore XML_override sia corretta.

DXXQ031E Il nome tabella non può superare la lunghezza massima consentita dal DB2.

Spiegazione: Il nome tabella specificato dall'elemento di condizione nella DAD è troppo lungo.

Risposta dell'utente: Modificare la lunghezza del nome tabella nella DAD.

DXXQ032E Il nome colonna non può superare la lunghezza massima consentita dal DB2.

Spiegazione: Il nome colonna specificato dall'elemento di condizione nella DAD è troppo lungo.

Risposta dell'utente: Modificare la lunghezza del nome colonna nella DAD.

DXXQ033E Identificativo <*tipo identificativo*> **non valido.**

Spiegazione: La stringa non è un identificativo SQL DB2 valido.

Risposta dell'utente: Correggere la stringa nella DAD

per renderla conforme alle regole definite per l'identificativo SQL DB2.

DXXQ034E Elemento di condizione nel RDB_node iniziale della DAD non valido: <*condizione*>

Spiegazione: L'elemento di condizione deve essere una clausola WHERE valida costituita da condizioni di collegamento valide collegate dalla congiunzione AND.

Risposta dell'utente: Per informazioni sulla sintassi corretta delle condizioni di collegamento in una DAD, consultare la documentazione XML Extender.

DXXQ035E Condizione di collegamento nel RDB_node iniziale della DAD non valida: <*condizione*>

Spiegazione: I nomi colonna nell'elemento di condizione di RDB_node devono essere definiti con i nomi tabella se DAD specifica più tabelle.

Risposta dell'utente: Per informazioni sulla sintassi corretta delle condizioni di collegamento in una DAD, consultare la documentazione XML Extender.

DXXQ036E Nome di schema specificato in una tag della condizione DAD di lunghezza maggiore di quella consentita.

Spiegazione: E stato rilevato un errore durante l'analisi del testo in una tag della condizione interna al file DAD. Il testo della condizione contiene un ID specificato da un nome di schema troppo lungo.

Risposta dell'utente: Correggere il testo delle tag della condizione nella DAD (document access definition).

DXXQ037E Impossibile generare <*element*> **con ricorrenze multiple.**

Spiegazione: Il nodo dell'elemento e i relativi elementi secondari non presentano alcuna associazione al database, ma il relativo valore multi_occurrence corrisponde a YES.

Risposta dell'utente: Correggere la DAD impostando il valore multi_occurrence su NO oppure creare un RDB_node in uno dei relativi elementi secondari.

Traccia diagnostica

XML Extender fornisce una funzione di traccia che registra l'attività dei server XML Extender. E' consigliabile utilizzare la funzione di traccia solo seguendo le istruzioni fornite dal supporto di assistenza IBM.

La funzione di traccia registra le informazioni in un file del server riguardo a una grande varietà di eventi quali l'entrata o l'uscita da un componente XML Extender oppure la restituzione di un codice d'errore da parte di un componente DB2 XML Extender. Poiché essa registra le informazioni per molti eventi, la funzione traccia dovrebbe essere utilizzata soltanto se necessario, ad esempio quando si esamina la causa di condizioni di errore. Inoltre, occorre limitare il numero di applicazioni attive durante l'utilizzo della funzione traccia. Limitare il numero di applicazioni attive, può agevolare l'isolamento della causa di un problema.

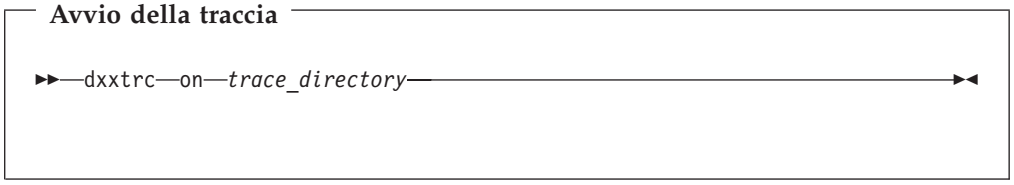
Utilizzare il comando **dxxtrc** per avviare o interrompere la funzione di traccia. E' possibile eseguire il comando da una riga comandi di un server AIX, Windows NT o Solaris. Per eseguire il comando, occorre avere l'autorizzazione SYSADM, SYSCTRL o SYSMINT.

Avvio della traccia

Scopo

Registra l'attività del server XML Extender. Per avviare la traccia, applicare l'opzione `on` al comando **dxxtrc**, insieme al nome dell'indirizzario che contiene il file di traccia. Quando la traccia è attivata, il file, `dxx`*`INSTANCE`*`.trc`, viene posizionato nell'indirizzario specificato. *INSTANCE* rappresenta il valore di DB2INSTANCE. Ogni istanza DB2 ha un proprio file di registrazione.

Formato



Parametri

Tabella 54. Parametri della traccia

Parametro	Descrizione
trace_directory	Il nome dell'indirizzario in cui viene posizionato `dxx`*`INSTANCE`*`.trc`. Obbligatorio, nessun valore predefinito.

Esempi

Il seguente esempio illustra l'avvio della funzione di traccia per un'istanza db2inst1 su AIX. Il file di traccia, `dxxdb2inst1.trc`, viene posizionato nell'indirizzario `/home/db2inst1/sqllib/log`.

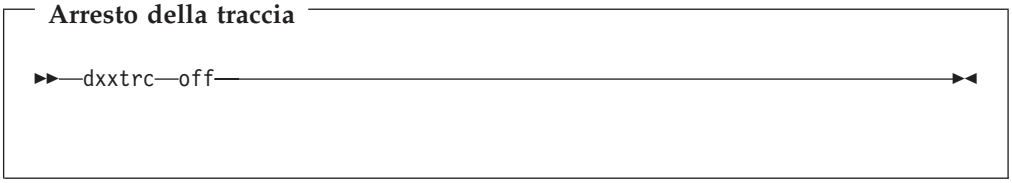
```
dxxtrc on /home/db2inst1/sqllib/log
```

Arresto della traccia

Scopo

Disattiva la traccia. Non viene registrata alcuna informazione sulla traccia. Poiché l'esecuzione della funzione di traccia può influire sulle prestazioni, si consiglia di disattivare la traccia in un ambiente di produzione.

Formato



Esempi

Questo esempio indica che la funzione di traccia viene disattivata.

```
dxxtrc off
```

Parte 5. Appendici

Appendice A. DTD per il file DAD

Questa sezione descrive le DTD (Document Type Declaration) per il file DAD (Document Access Definition). Il file DAD è un documento XML strutturato ad albero che richiede una DTD. Il nome del file DTD è `dxxdad.dtd`. La Figura 13 illustra la DTD per il file DAD. Gli elementi di questo file vengono descritti subito dopo la figura.

```
<?xml encoding="US-ASCII"?>

  <!ELEMENT DAD (dtdid?, validation, (Xcolumn | Xcollection))>
  <!ELEMENT dtdid (#PCDATA)>
  <!ELEMENT validation (#PCDATA)>
  <!ELEMENT Xcolumn (table*)>
  <!ELEMENT table (column*)>
  <!ATTLIST table name CDATA #REQUIRED
                              key CDATA #IMPLIED
                              orderBy CDATA #IMPLIED>
  <!ELEMENT column EMPTY>
  <!ATTLIST column
                      name CDATA #REQUIRED
                      type CDATA #IMPLIED
                      path CDATA #IMPLIED
                       multi_occurrence CDATA #IMPLIED>
  <!ELEMENT Xcollection (SQL_stmt?, objids?, prolog, doctype, root_node)>
  <!ELEMENT SQL_stmt (#PCDATA)>
  <!ELEMENT objids (column+)>
  <!ELEMENT prolog (#PCDATA)>
  <!ELEMENT doctype (#PCDATA | RDB_node)*>
  <!ELEMENT root_node (element_node)>
  <!ELEMENT element_node (RDB_node*,
                                        attribute_node*,
                                        text_node?,
                                        element_node*,
                                        namespace_node*,
                                        process_instruction_node*,
                                        comment_node*)>
  <!ATTLIST element_node
                     name CDATA #REQUIRED
                     ID CDATA #IMPLIED
                     multi_occurrence CDATA "NO"
                     BASE_URI CDATA #IMPLIED>
  <!ELEMENT attribute_node (column | RDB_node)>
  <!ATTLIST attribute_node
                     name CDATA #REQUIRED>
  <!ELEMENT text_node (column | RDB_node)>
  <!ELEMENT RDB_node (table+, column?, condition?)>
  <!ELEMENT condition (#PCDATA)>
  <!ELEMENT comment_node (#PCDATA)>
  <!ELEMENT namespace_node (EMPTY)>
  <!ATTLIST namespace_node
                      name CDATA #IMPLIED
                     value CDATA #IMPLIED>
  <!ELEMENT process_instruction_node (#PCDATA)>
```

Figura 13. DTD per il file DAD

Il file DAD ha quattro elementi principali:

- DTDID

- convalida
- Xcolumn
- Xcollection

Xcolumn e Xcollection hanno attributi ed elementi secondari che semplificano l'associazione dei dati XML alle tabelle referenziali DB2. Il seguente elenco descrive gli elementi principali e i relativi attributi ed elementi secondari. Gli esempi di sintassi sono descritti nella sezione Figura 13 a pagina 217.

DTDID
Specifica l'ID della DTD memorizzata nella tabella DTD_REF. `DTDID` fa riferimento alla DTD che convalida i documenti XML o consente di eseguire l'associazione tra le tabelle della raccolta XML e i documenti XML. `DTDID` deve essere specificato per le raccolte XML. Per le colonne XML, è facoltativo ed è richiesto solo se si desidera creare delle tabelle laterali per l'indicizzazione degli elementi o attributi oppure per la convalida dei documenti XML di input. `DTDID` deve essere uguale al SYSTEM ID specificato nel `doctype` dei documenti XML.

Sintassi: `<!ELEMENT dtdid (#PCDATA)>`

validation
Indica se convalidare o meno il documento XML con la DTD per DAD. Se si specifica YES, occorre specificare anche `DTDID`.

Sintassi: `<!ELEMENT validation(#PCDATA)>`

Xcolumn
Definisce lo schema di indice per una colonna XML. Questo schema è costituito da zero o più tabelle.

Sintassi: `<!ELEMENT Xcolumn (table*)>Xcolumn` ha un elemento secondario, `table`.

table Definisce una o più tabelle relazionali per l'indicizzazione degli elementi o attributi dei documenti memorizzati in una colonna XML.

Sintassi:

```
<!ELEMENT table (column+)>
  <!ATTLIST table name CDATA #REQUIRED
key CDATA #IMPLIED
orderBy CDATA #IMPLIED>
```

L'elemento `table` presenta un attributo:

attributo nome
Specifica il nome della tabella laterale

L'elemento `table` ha un elemento secondario:

attributo chiave
La chiave primaria singola della tabella

attributo orderBy
I nomi delle colonne che determinano l'ordine sequenziale dei valori dell'attributo o del testo dell'elemento a ricorrenza multipla quando vengono creati i documenti XML.

elemento colonna
Specifica la colonna della tabella che contiene il valore di un percorso di ubicazione del tipo specificato.

Sintassi:

```
<!ATTLIST column
                      name CDATA #REQUIRED
                       type CDATA #IMPLIED
                       path CDATA #IMPLIED
                        multi_occurrence CDATA #IMPLIED>
```

L'elemento `column` ha i seguenti attributi:

attributo nome
Specifica il nome della colonna. E' il nome alias del percorso di ubicazione che identifica un elemento o attributo.

attributo tipo
Definisce il tipo di dati della colonna. Può essere un qualsiasi tipo di dati SQL.

attributo percorso
Indica il percorso di ubicazione di un attributo o elemento XML e deve essere un percorso di ubicazione semplice come specificato nella tabella 3.1.a.

attributo multi_occurrence
Specifica se questo elemento o attributo può essere utilizzato più volte all'interno di un documento XML. I valori possono essere YES e NO.

Xcollection
Definisce l'associazione tra i documenti XML e una raccolta XML di tabelle relazionali.

Sintassi: `<!ELEMENT Xcollection(SQL_stmt*, prolog, doctype, root_node)>Xcollection` presenta i seguenti elementi secondari:

SQL_stmt
Specifica l'istruzione SQL utilizzata da XML Extender per definire la raccolta. In modo specifico, l'istruzione seleziona i dati XML dalle tabelle della raccolta XML e li utilizza per creare i documenti XML nella raccolta. Il valore di questo elemento deve essere un'istruzione SQL valida. Viene utilizzato per la composizione ed è consentito un solo elemento `SQL_stmt`. Per la scomposizione, è possibile specificare più valori per `SQL_stmt` per eseguire la creazione e l'inserimento della tabella necessaria.

Sintassi: `<!ELEMENT SQL_stmt  #PCDATA >`

prolog Il testo per il prologo XML. Lo stesso prologo è fornito a tutti i documenti dell'intera raccolta. Il valore di `prolog` è fisso.

Sintassi: `<!ELEMENT prolog  #PCDATA>`

doctype
Definisce il testo per la definizione del tipo di documento XML.

Sintassi: `<!ELEMENT doctype  #PCDATA | RDB_node>doctype` può essere specificato in uno dei seguenti modi:

- Definire un valore esplicito. Questo valore è fornito a tutti i documenti dell'intera raccolta.
- Durante la scomposizione, specificare l'elemento secondario, `RDB_node`, che può essere associato e memorizzato come i dati della colonna di una tabella.

`doctype` presenta un elemento secondario:

RDB_node
Non ancora implementato.

root_node
Definisce il nodo root virtuale. `root_node` deve avere un elemento secondario obbligatorio, `element_node`, che può essere utilizzato una sola volta. L'`element_node` di `root_node` è in effetti il `root_node` del documento XML.

Sintassi: `<!ELEMENT root_node(element_node)>`

element_node
Indica un elemento XML. Questo deve essere definito nella DTD specificata per la raccolta. Per l'associazione `RDB_node`, `element_node` root deve avere un `RDB_node` per specificare tutte le tabelle contenenti dati XML per tutti i relativi nodi secondari. Può presentare zero o più `attribute_node` ed `element_node` secondari oltre a zero o più `text_node`. Per gli elementi differenti dall'elemento root non è necessario specificare alcun RDB_node.

Sintassi:

Un `element_node` viene definito dai seguenti elementi secondari:

RDB_node
(Facoltativo) Specifica le tabelle, le colonne e le condizioni per i dati XML. RDB_node per un elemento deve essere definito solo per l'associazione RDB_node. In tal caso, è necessario specificare una o più tabelle. La colonna non è necessaria poiché il contenuto dell'elemento viene specificato da text_node. La condizione è facoltativa, dipende dalla DTD e dalla condizione dell'interrogazione.

nodi secondari
(Facoltativo) Un `element_node` può anche avere i seguenti nodi secondari:

element_node
Indica gli elementi secondari dell'elemento XML corrente

attribute_node
Indica gli attributi dell'elemento XML corrente

text_node
Indica il testo CDATA dell'elemento XML corrente

attribute_node
Indica un attributo XML. E' il nodo che definisce l'associazione tra un attributo XML e i dati della colonna di una tabella relazionale.

Sintassi:

L'elemento `attribute_node` deve disporre delle definizioni per un attributo `name` e per un elemento secondario `column` o `RDB_node`. `attribute_node` presenta il seguente attributo:

name Il nome dell'attributo.

`attribute_node` presenta i seguenti elementi secondari:

Column
Utilizzato per l'associazione SQL. La colonna deve essere specificata nella clausola SELECT di `SQL_stmt`.

RDB_node
Utilizzato per l'associazione RDB_node. Il nodo definisce l'associazione tra questo attributo e i dati della colonna nella tabella relazionale. E' necessario specificare la tabella e la colonna. La condizione è facoltativa.

text_node
Rappresenta il contenuto del testo di un elemento XML. E' il nodo che definisce l'associazione tra un elemento XML e i dati della colonna di una tabella relazionale.

Sintassi: Deve essere definito da un elemento secondario `column` o `RDB_node`:

Column
Richiesto per l'associazione SQL. In questo caso, la colonna deve essere specificata nella clausola SELECT di `SQL_stmt`.

RDB_node
Richiesto per l'associazione RDB_node. Il nodo definisce l'associazione tra questo contenuto del testo e i dati della colonna nella tabella relazionale. La `tabella` e la `colonna` devono essere specificate. La condizione è facoltativa.

Appendice B. Esempi

Quest'appendice illustra gli oggetti di esempio utilizzati negli esempi riportati nella presente pubblicazione.

DTD XML

La seguente DTD viene utilizzata per il documento `getstart.xml` cui si fa riferimento in questa pubblicazione ed è riportata nella sezione Figura 15 a pagina 224.

```
<!xml encoding="US-ASCII"?>

<!ELEMENT Order (Customer, Part+)>
<!ATTLIST Order key CDATA #REQUIRED>
<!ELEMENT Customer (Name, Email)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Email (#PCDATA)>
<!ELEMENT Part (key,Quantity,ExtendedPrice,Tax, Shipment+)>
<!ELEMENT key (#PCDATA)>
<!ELEMENT Quantity (#PCDATA)>
<!ELEMENT ExtendedPrice (#PCDATA)>
<!ELEMENT Tax (#PCDATA)>
<!ATTLIST Part color CDATA #REQUIRED>
<!ELEMENT Shipment (ShipDate, ShipMode)>
<!ELEMENT ShipDate (#PCDATA)>
<!ELEMENT ShipMode (#PCDATA)>
```

Figura 14. DTD XML di esempio: getstart.dtd

Documento XML: getstart.xml

Il seguente documento XML, `getstart.xml`, è il documento XML di esempio utilizzato negli esempi riportati in questa pubblicazione. Tale documento contiene tag XML per eseguire un ordine di acquisto.

```
<?xml version="1.0"?>
<!DOCTYPE Order SYSTEM "c:\dxx\samples\dtd\getstart.dtd">
    <Order key="1">
        <Customer>
          <Name>American Motors</Name>
          <Email>parts@am.com</Email>
        </Customer>
        <Part color="black">
          <key>68</key>
          <Quantity>36</Quantity>
          <ExtendedPrice>34850.16</ExtendedPrice>
          <Tax>6.000000e-02</Tax>
          <Shipment>
              <ShipDate>1998-08-19</ShipDate>
              <ShipMode>BOAT </ShipMode>
          </Shipment>
          <Shipment>
              <ShipDate>1998-08-19</ShipDate>
              <ShipMode>AIR </ShipMode>
          </Shipment>
  </Part>
  <Part color="red   ">
    <key>128</key>
    <Quantity>28</Quantity>
    <ExtendedPrice>38000.00</ExtendedPrice>
    <Tax>7.000000e-02</Tax>
          <Shipment>
      <ShipDate>1998-12-30</ShipDate>
      <ShipMode>TRUCK </ShipMode>
          </Shipment>
  </Part>
</Order>
```

Figura 15. Documento XML di esempio: `getstart.xml`

File DAD

Le seguenti sezioni contengono i file DAD (document access definition) che associano i dati XML alle tabelle relazionali DB2, utilizzando le modalità di accesso colonna XML o raccolta XML.

- "File DAD: colonna XML"
- La sezione "File DAD: raccolta XML - associazione SQL" a pagina 225 illustra un file DAD per una raccolta XML che utilizza un'associazione SQL.
- La sezione "File DAD: XML - associazione RDB_node" a pagina 226 illustra un file DAD per una raccolta XML che utilizza l'associazione RDB_node.

File DAD: colonna XML

Questo file DAD contiene l'associazione per una colonna XML, definendo la tabella, le tabelle laterali e le colonne in cui contenere i dati XML.

```
<?xml version="1.0"?>
<!DOCTYPE Order SYSTEM "c:\dxx\dad.dtd">
<DAD>
   <dtdid>c:\dxx\samples\dtd\getstart.dtd</dtdid>
  <validation>YES</validation>

<Xcolumn>
     <table name="order_side_tab">
   <column name="order_key"
           type="integer"
           path="/Order/@key"
           multi_occurrence="NO"/>
   <column name="customer"
           type="varchar(50)"
           path="/Order/Customer/Name"
           multi_occurrence="NO"/>
</table>
     <table name="part_side_tab">
         <column name="price"
            type="decimal(10,2)"
            path="/Order/Part/ExtendedPrice"
            multi_occurrence="YES"/>
</table>
     <table name="ship_side_tab">
         <column name="date"
            type="DATE"
            path="/Order/Part/Shipment/ShipDate"
            multi_occurrence="YES"/>
</table>

</Xcolumn>

</DAD>
```

Figura 16. File DAD di esempio per una colonna XML

File DAD: raccolta XML - associazione SQL

Questo file DAD contiene un'istruzione SQL che specifica le condizioni, le tabelle e le colonne DB2 in cui contenere i dati XML.

```
<?xml version="1.0"?>
<!DOCTYPE DAD SYSTEM "c:\dxx\dtd\dad.dtd">
<DAD>
<validation>NO</validation>
<Xcollection>
<SQL_stmt>SELECT o.order_key, customer_name, customer_email, p.part_key, color, quantity,
price, tax, ship_id, date, mode from order_tab o, part_tab p,
table (select substr(char(timestamp(generate_unique())),16)
  as ship_id, date, mode, part_key from ship_tab) s
      WHERE o.order_key = 1 and
                p.price > 20000 and
            p.order_key = o.order_key and
            s.part_key = p.part_key
         ORDER BY order_key, part_key, ship_id</SQL_stmt>
<prolog>?xml version="1.0"?</prolog>
<doctype>! DOCTYPE Order SYSTEM "c:\dxx\samples\dtd\getstart.dtd"</doctype>
```

Figura 17. File DAD di esempio per una raccolta XML che utilizza l'associazione SQL (Numero 1 di 2)

```
    <root_node>
<element_node name="Order">
<attribute_node name="key">
                 <column name="order_key"/>
</attribute_node>
           <element_node name="Customer">
    <element_node name="Name">
      <text_node><column name="customer_name"/></text_node>
</element_node>
    <element_node name="Email">
      <text_node><column name="customer_email"/></text_node>
</element_node>
</element_node>
           <element_node name="Part">
    <attribute_node name="color">
      <column name="color"/>
</attribute_node>
    <element_node name="key">
      <text_node><column name="part_key"/></text_node>
</element_node>
<element_node name="Quantity">
      <text_node><column name="quantity"/></text_node>
</element_node>
           <element_node name="ExtendedPrice">
      <text_node><column name="price"/></text_node>
</element_node>
<element_node name="Tax">
      <text_node><column name="tax"/></text_node>
</element_node>
    <element_node name="Shipment" multi_occurence="YES">
              <element_node name="ShipDate">
        <text_node><column name="date"/></text_node>
</element_node>
              <element_node name="ShipMode">
        <text_node><column name="mode"/></text_node>
</element_node>
</element_node>
</element_node>
</element_node>
    </root_node>
</Xcollection>
</DAD>
```

Figura 17. File DAD di esempio per una raccolta XML che utilizza l'associazione SQL (Numero 2 di 2)

File DAD: XML - associazione RDB_node

Questo file DAD utilizza elementi <RDB_node> per definire le condizioni, le colonne e le tabelle DB2 in cui contenere i dati XML.

```
<?xml version="1.0"?>
<!DOCTYPE Order SYSTEM "c:\dxx\dtd\dad.dtd">
<DAD>
   <dtdid>c:\dxx\samples\dtd\getstart.dtd</dtdid>
  <validation>YES</validation>
<Xcollection>
<prolog>?xml version="1.0"?</prolog>
<doctype>! DOCTYPE Order SYSTEM "c:\dxx\samples\dtd\getstart.dtd"</doctype>
    <root_node>
<element_node name="Order">
        <RDB_node>
        <table name="order_tab"/>
            <table name="part_tab"/>
        <table name="ship_tab"/>
            <condition>
order_tab.order_key = part_tab.order_key AND
part_tab.part_key = ship_tab.part_key
            </condition>
      </RDB_node>
<attribute_node name="key">
        <RDB_node>
        <table name="order_tab"/>
                 <column name="order_key"/>
      </RDB_node>
</attribute_node>
           <element_node name="Customer">
   <text_node>
        <RDB_node>
        <table name="order_tab"/>
                    <column name="customer"/>
      </RDB_node>
   </text_node>
</element_node>
           <element_node name="Part">
        <RDB_node>
            <table name="part_tab"/>
        <table name="ship_tab"/>
            <condition>
part_tab.part_key = ship_tab.part_key
            </condition>
      </RDB_node>
<attribute_node name="key">
        <RDB_node>
            <table name="part_tab"/>
    <column name="part_key"/>
      </RDB_node>
</attribute_node>
```

Figura 18. File DAD di esempio per una raccolta XML che utilizza l'associazione RDB_node (Numero 1 di 3)

```
          <element_node name="Quantity">
   <text_node>
        <RDB_node>
            <table name="part_tab"/>
                    <column name="quantity"/>
      </RDB_node>
   </text_node>
</element_node>
          <element_node name="ExtendedPrice">
   <text_node>
        <RDB_node>
            <table name="part_tab"/>
            <column name="price"/>
            <condition>
                 price > 2500.00
            </condition>
      </RDB_node>
   </text_node>
</element_node>
<element_node name="Tax">
   <text_node>
        <RDB_node>
            <table name="part_tab"/>
         <column name="tax"/>
      </RDB_node>
   </text_node>
</element_node>
```

Figura 18. File DAD di esempio per una raccolta XML che utilizza l'associazione RDB_node (Numero 2 di 3)

```
          <element_node name="shipment">
        <RDB_node>
        <table name="ship_tab"/>
            <condition>
                    part_key = part_tab.part_key
            </condition>
      </RDB_node>
              <element_node name="ShipDate">
   <text_node>
        <RDB_node>
        <table name="ship_tab"/>
                        <column name="date"/>
            <condition>
                          date > "1966-01-01"
            </condition>
      </RDB_node>
   </text_node>
</element_node>
              <element_node name="ShipMode">
   <text_node>
        <RDB_node>
        <table name="ship_tab"/>
                        <column name="mode"/>
      </RDB_node>
   </text_node>
</element_node>
              <element_node name="Comment">
   <text_node>
        <RDB_node>
        <table name="ship_tab"/>
                        <column name="comment"/>
      </RDB_node>
   </text_node>
</element_node>
           </element_node> <! -- end of element Shipment>
         </element_node> <! -- end of element Part --->
        </element_node> <! -- end of element Order --->
    </root_node>
</Xcollection>
</DAD>
```

Figura 18. File DAD di esempio per una raccolta XML che utilizza l'associazione RDB_node (Numero 3 di 3)

Appendice C. Considerazioni relative alle code page

E' importante accertarsi che i documenti XML e gli altri file correlati siano adeguatamente codificati per ciascun client o server che accede ai file. E', quindi, importante conoscere i presupposti di XML Extender quando si elabora un file e quando, successivamente, vengono gestite conversioni di code page. Le considerazioni primarie sono le seguenti:

- Assicurarsi che la code page del client che richiama un documento XML dal DB2 corrisponda alla codifica del documento.
- Assicurarsi che, quando il documento viene elaborato da un parser XML, la dichiarazione di codifica del documento XML sia anch'essa congruente con la codifica.

Le seguenti sezioni trattano in dettaglio queste considerazioni, forniscono la preparazione a possibili problemi e descrivono il modo in cui XML Extender e DB2 supportano le code page quando i documenti vengono trasmessi dal client al server e al database.

Terminologia

I seguenti termini vengono utilizzati nella presente sezione:

codifica del documento
L'effettiva code page di un documento XML.

dichiarazione di codifica del documento
Il nome della code page specificata nella dichiarazione XML. Ad esempio:
```
<?xml version="1.0" encoding="ibm037"?>
```

documento congruente
Un documento in cui la codifica corrisponde alla code page di dichiarazione codifica del documento.

documento incongruente
Un documento in cui la codifica non corrisponde alla code page di dichiarazione codifica del documento.

variabile (di ambiente) di registro DB2CODEPAGE
Specifica la code page dei dati presentati al DB2 da un'applicazione client di database. DB2 riceve la code page del client dalla locale del sistema operativo del client, a meno che non sia impostata questa variabile. Per il DB2, se impostato, questo valore sostituisce la locale del sistema operativo del client.

code page client
La code page dell'applicazione. Se la variabile DB2CODEPAGE è impostata, la code page del client è rappresentata dal valore di DB2CODEPAGE. Diversamente, la code page del client sarà la locale del sistema operativo del client.

code page server, oppure, code page di locale del sistema operativo server
La locale del sistema operativo su cui è installato il database DB2.

code page database:
La codifica dei dati memorizzati, determinata al momento della creazione

del database. Se non specificamente impostato mediante la clausola USING CODESET, il valore predefinito è impostato sulla locale del sistema operativo del server.

Presupposti di code page DB2 e XML Extender

Quando DB2 riceve o invia un documento XML, non esegue il controllo della dichiarazione di codifica. Controlla invece la code page del client per verificare se corrisponde a quella del database. Se le code page sono differenti, DB2 converte i dati del documento XML in modo che risultino compatibili con la code page:

- del database, quando si importa il documento o un frammento di esso in una tabella di database
- del database, quando si esegue la scomposizione di un documento in una o più tabelle di database
- del client, quando si esporta il documento dal database e lo si sottopone al client
- del server, quando si elabora un file con un UDF che restituisce dati in un file sul file system del server

Quando un documento XML viene importato nel database, viene generalmente importato come documento XML da memorizzare in una colonna XML, oppure per la scomposizione di una raccolta XML, con i contenuti di elemento e attributo che verranno salvati come dati DB2. Quando si importa un documento, DB2 converte la codifica del documento nella codifica del database. DB2 presuppone che il documento abbia la code page specificata nella colonna "Code page di origine" riportata nella tabella seguente. Tabella 55 sintetizza le conversioni operate da DB2 quando si importa un documento XML.

Tabella 55. Utilizzo di UDF e procedure memorizzate quando si importa il file XML nel database

Metodo di elaborazione	Code page di origine per la conversione	Code page di destinazione per la conversione	Commenti
Inserimento file DTD nella tabella DTD_REF	Code page client	Code page database	
Abilitare colonna o procedure memorizzate di raccolta, oppure comandi di gestione, che importano il file DAD	Code page client	Code page database	
UDF: • XMLVarcharFromFile() • XMLCLOBFromFile() • Content(): richiamo da XMLFILE in CLOB	Code page server	Code page database	
Procedure memorizzate di scomposizione	Code page client	Code page database	Il documento da scomporre si presuppone abbia la code page del client. I dati della scomposizione sono memorizzati in tabelle nella code page del database.

Quando un documento XML viene esportato dal database, ciò avviene generalmente: in base ad una richiesta di presentazione di un documento da parte del client, in base a un'interrogazione del contenuto di un documento XML oppure durante la composizione di un documento da dati DB2. Quando si esporta un documento, DB2 converte la codifica del documento nella codifica del client o del server, in base a dove ha avuto origine la richiesta e in base a dove devono essere presentati i dati. Tabella 56 riassume le conversioni eseguite da DB2 quando si esporta un documento XML.

Tabella 56. Utilizzo di UDF e procedure memorizzate quando si esporta il file XML dal database

Metodo di elaborazione	Conversione	Commenti
UDF • XMLFileFromVarchar() • XMLFileFromCLOB()	Code page del database in code page del client al momento della presentazione dei dati al client	
UDF • Content(): richiamo da XMLVARCHAR in un server file	Code page del database in code page del server	
Procedure memorizzate di composizione: tabelle risultanti memorizzate nella tabella dei risultati, che può essere interrogata e esportata.	Code page del database in code page del client quando l'insieme dei risultati viene presentato al client.	Durante la composizione di documenti, XML Extender copia la dichiarazione di codifica, specificata mediante la tag nella DAD, nel documento appena creato. Quando viene presentato, il documento deve corrispondere alla code page del client.

Considerazioni relative alla dichiarazione di codifica

La dichiarazione di codifica specifica la codifica del documento XML e viene visualizzata nell'istruzione della dichiarazione XML. Quando si utilizza XML Extender, è importante assicurarsi che la codifica del documento corrisponda alla code page del client o del server, in base a dove è ubicato il file.

Dichiarazioni di codifica legale

Nei documenti XML è possibile utilizzare qualsiasi dichiarazione di codifica, seguendo alcune norme. In questa sezione vengono definite tali norme, oltre alle dichiarazioni di codifica supportate.

Le codifiche trasportabili raccomandate per i dati XML sono UTF-8 e UTF-16, in conformità alla specifica XML. Se si utilizzano queste codifiche, la propria applicazione può interoperare tra diverse società. Se si utilizzano le codifiche elencate in Tabella 57 a pagina 234, l'applicazione può essere condivisibile su sistemi operativi IBM. Se si utilizzano altre codifiche, sarà meno probabile poter trasportare i propri dati.

Per tutti i sistemi operativi, sono supportate le seguenti dichiarazioni di codifica. L'elenco seguente descrive il senso di ciascuna colonna:

- **Codifica** specifica la stringa di codifica da utilizzare nella dichiarazione XML.
- **OS** evidenzia il sistema operativo su cui DB2 supporta la specifica code page.

- **Code page** descrive la code page definita da IBM associata alla codifica specificata.

Tabella 57. Dichiarazioni di codifica supportate da XML Extender

Categoria	Codifica	OS	Code page
Unicode	UTF-8	AIX, SUN, Linux	1208
	UTF-16	AIX, SUN, Linux	1200
ASCII	iso-8859-1	AIX, Linux, Sun	819
	ibm-1252	Windows NT	1252
	iso-8859-2	AIX, Linux, Sun	912
	iso-8859-5	AIX, Linux	915
	iso-8859-6	AIX	1089
	iso-8859-7	AIX, Linux	813
	iso-8859-8	AIX, Linux	916
	iso-8859-9	AIX, Linux	920
MBCS	gb2312	Windows NT	1386
	ibm-932, shift_jis78	AIX	932
	Shift_JIS	solo AIX 4.3, Windows NT	943
	IBM-eucCN	AIX, Sun	1383
	IBM-eucJP, EUC-JP	AIX, Linux, Sun	954, 33722
	euc-tw, IBM-eucTW	AIX, Sun	964
	euc-kr, IBM-eucKR	AIX	970
	big5	AIX, Sun, Windows NT	950

La stringa di codifica deve essere compatibile con la code page della destinazione del documento. Se un documento deve essere restituito da un server a un client, la stringa di codifica deve essere compatibile con la code page del client. Consultare "Codifiche congruenti e dichiarazioni di codifica" per le conseguenze derivanti da codifiche non compatibili. Fare riferimento al seguente URL per un elenco delle code page supportate dal parser XML utilizzato da XML Extender:

`http://www.ibm.com/software/data/db2/extenders/xmlext/moreinfo/encoding.html`

Codifiche congruenti e dichiarazioni di codifica

Quando un documento XML viene elaborato o scambiato con un altro sistema, è importante che la dichiarazione di codifica corrisponda all'effettiva codifica del documento. La verifica che la codifica di un documento sia congruente con il client risulta una funzione utile perché gli strumenti XML, come i programmi di analisi, generano un errore per un'entità che include un dichiarazione di codifica differente da quella specificata nella dichiarazione.

Figura 19 a pagina 235 descrive client che hanno code page congruenti con la codifica del documento e con la codifica dichiarata.

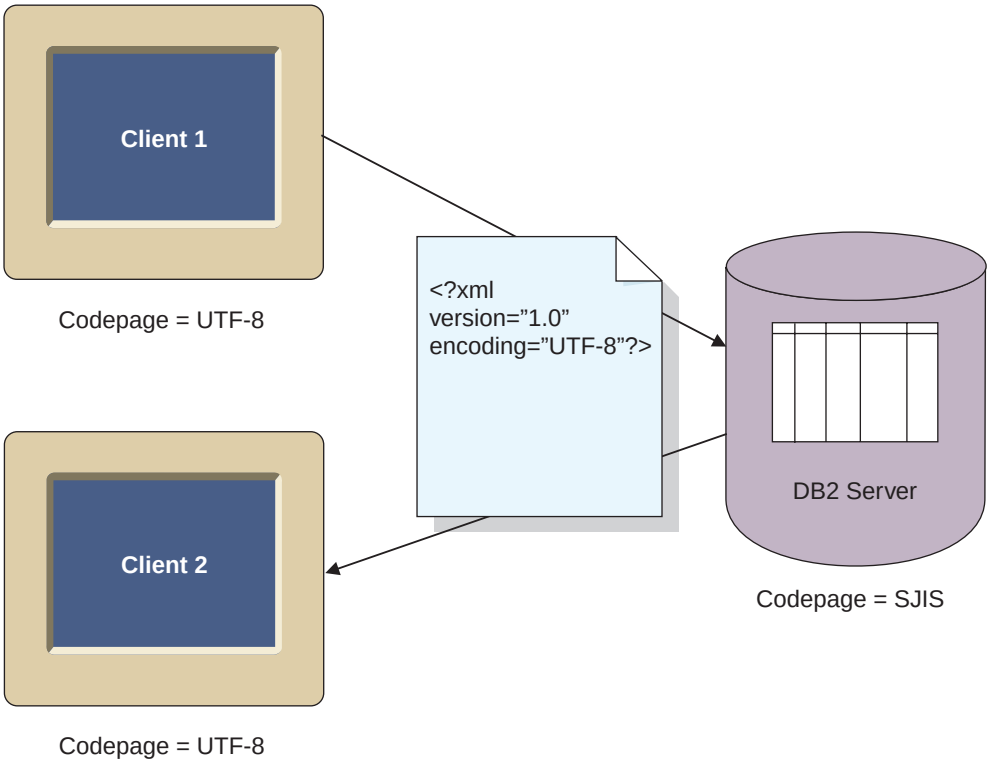


Figura 19. Client con code page corrispondenti

Nel caso in cui le code page risultino differenti, è possibile che si verifichino le seguenti condizioni:

- La conversione può causare la perdita dei dati, soprattutto se la code page di origine è Unicode e quella di destinazione è differente. Unicode contiene la serie completa di conversioni caratteri. Se un file viene convertito da UTF-8 in una code page che non supporta tutti i caratteri utilizzati nel documento, i dati potrebbero essere persi durante la conversione.
- La codifica dichiarata del documento XML potrebbe non essere più congruente con quella del documento effettivo, se questo viene richiamato da un client con una code page diversa da quella della codifica dichiarata del documento.

La Figura 20 a pagina 236 illustra un ambiente in cui le code page dei client non sono congruenti.

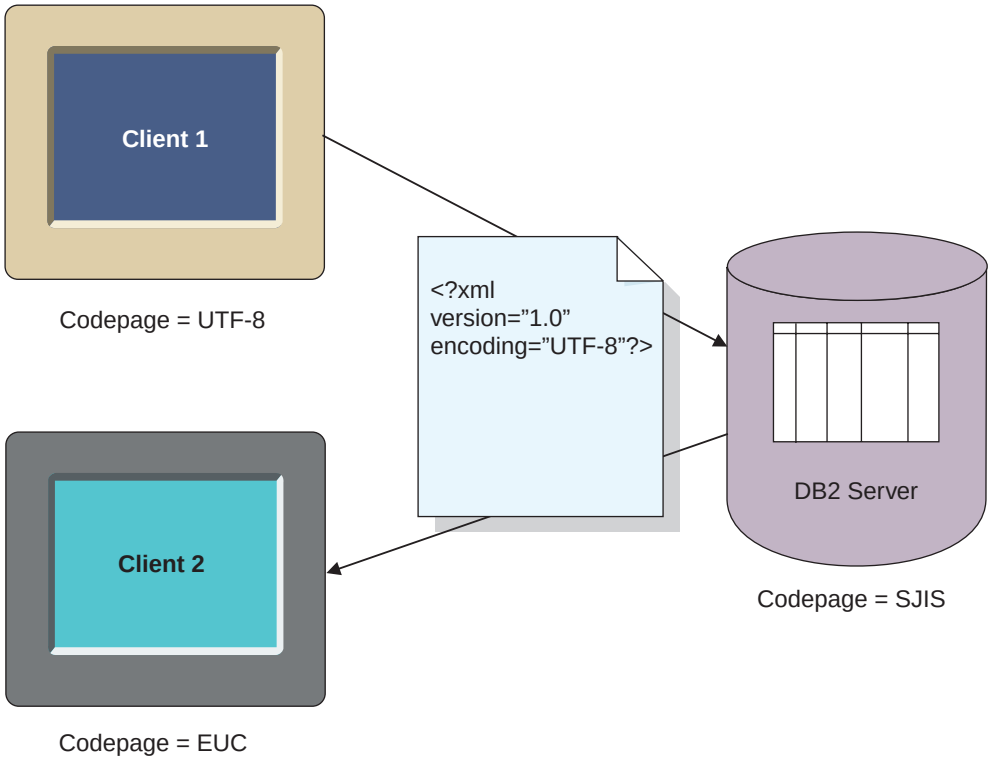


Figura 20. I client presentano code page non corrispondenti.

Client2 riceve il documento in EUC ma il documento avrà una dichiarazione di codifica UTF-8.

Dichiarazione di una codifica

Il valore predefinito della dichiarazione di codifica è UTF-8 e l'omissione della dichiarazione di codifica indica che il documento è in UTF-8.

Per dichiarare un valore di codifica:

Nella dichiarazione del documento XML specificare la dichiarazione di codifica con il nome della code page del client. Ad esempio:

```
<?xml version="1.0" encoding="UTF-8" ?>
```

Scenari di conversione

XML Extender elabora documenti XML quando:

- Memorizza e richiama dati di colonne XML, utilizzando il metodo di accesso e di memorizzazione colonne XML
- Compone e scompone documenti XML

I documenti vengono sottoposti alla conversione di code page quando sono trasferiti da un client o un server, a un database. Incongruenze o danni ai documenti XML possono verificarsi spesso durante le conversioni da code page del client, del server e del database. Quando si sceglie la dichiarazione di codifica del documento e quando si stabiliscono i client e i server che possono importare o esportare documenti dal database, tenere presente le conversioni descritte nelle tabelle precedenti e gli scenari riportati di seguito.

I seguenti scenari illustrano possibili situazioni di conversione:

Scenario 1: Questo scenario rappresenta una configurazione con codifiche congruenti, nessuna conversione DB2 e un documento importato dal server. La dichiarazione di codifica del documento è UTF-8, il server è UTF-8 e il database è UTF-8.

1. Il documento viene importato nel DB2 utilizzando la UDF XMLClobfromFile.
2. Il documento viene estratto nel server.
3. DB2 non ha bisogno di convertire il documento poiché la code page del server e quella del database sono identiche. La codifica e la dichiarazione sono congruenti.

Scenario 2: Questo scenario rappresenta una configurazione con codifiche congruenti, conversione DB2 e un documento importato dal server ed esportato nel client. La codifica e la dichiarazione del documento è SJIS, la code page del client è SJIS, le code page del server e del database sono UTF-8.

1. Il documento viene importato in DB2 utilizzando la UDF XMLClobfromfile del server.
2. DB2 converte il documento da SJIS e lo memorizza in UTF-8. La codifica e la dichiarazione di codifica nel database sono incongruenti.
3. Un client che utilizza SJIS richiede il documento per una presentazione sul browser web.
4. DB2 converte il documento in SJIS, cioè la code page del client. Adesso la codifica e la dichiarazione del documento sono congruenti sul client.

Scenario 3: Questo scenario rappresenta una configurazione con codifiche non congruenti, conversione DB2 e un documento importato dal server ed esportato in un client. La dichiarazione di codifica del documento è SJIS per il documento in entrata. La code page del server è SJIS, le code page del client e del database sono UTF-8.

1. Il documento viene importato nel database utilizzando una UDF di memorizzazione.
2. DB2 converte il documento da SJIS in UTF-8. La codifica e la dichiarazione non sono congruenti.
3. Un client con code page UTF-8 richiede il documento per una presentazione su un browser web.
4. DB2 non esegue la conversione perché le code page del client e del database sono uguali.
5. La dichiarazione e la codifica del documento sono incompatibili poiché la prima è SJIS mentre la seconda è UTF-8. Il documento non può essere elaborato da un parser XML o da altri strumenti di elaborazione XML.

Scenario 4: Questo scenario rappresenta una configurazione con dati persi, conversione DB2 e un documento importato da un server UTF-8: la dichiarazione di codifica del documento è UTF-8, il server è UTF-8 e il database è SJIS.

1. Il documento viene importato nel DB2 utilizzando la UDF XMLClobfromFile.
2. DB2 converte la codifica in SJIS. Quando il documento viene importato, il documento memorizzato nel database potrebbe essere corrotto poiché i caratteri rappresentati in UTF-8 potrebbero non avere una rappresentazione in SJIS.

Scenario 5:

Questo scenario descrive una configurazione con una limitazione di Windows NT. Su Windows NT, le locali del sistema operativo non possono essere impostate su UTF-8; tuttavia, DB2 consente al client di impostare la code page su UTF-8 mediante `db2set DB2CODEPAGE=1208`. In questo scenario, il client e il server si trovano sulla stessa macchina. Il client è UTF-8, ma il server non può essere impostato su UTF-8; la sua code page è 1252. Il documento è codificato come 1252 e la dichiarazione di codifica è ibm-1252. La code page del database è UTF-8.

1. Il documento viene importato dal server mediante una UDF di memorizzazione e convertito da 1252 in 1208.
2. Il documento viene esportato dal DB2 mediante la UDF Content() del client.
3. DB2 converte il documento da UTF-8 a 1252, anche se il client potrebbe aspettare 1208 dato che il client si trova sulla stessa macchina del server e quest'ultimo è impostato su 1208.

Come evitare documenti XML incompatibili

Nelle sezioni precedenti è stato trattato il caso di un documento XML con codifica incongruente, che presenta, cioè, la dichiarazione di codifica in conflitto con la codifica del documento. Le codifiche incompatibili possono provocare la perdita di dati oppure rendere inutilizzabili i documenti XML.

Utilizzare una delle seguenti raccomandazioni per assicurare la compatibilità della codifica di documenti XML con la code page del client prima di sottoporre il documento a un processore XML, ad esempio un parser:

- Quando si esporta un documento da un database utilizzando le UDF di XML Extender, provare una delle tecniche seguenti: (si presuppone che XML Extender abbia esportato il file, nella code page del server, nel file system sul server).
  - Convertire il documento nella code page della codifica dichiarata
  - Sostituire la codifica, se lo strumento dispone della funzione di sostituzione
  - Modificare manualmente la dichiarazione di codifica del documento esportato nella codifica effettiva del documento (cioè, la code page del server)
- Quando si esporta un documento dal database utilizzando le procedure memorizzate XML Extender, provare una delle tecniche seguenti: (si presuppone che il client stia eseguendo l'interrogazione della tabella dei risultati, in cui è memorizzato il documento composto)
  - Convertire il documento nella code page della codifica dichiarata
  - Sostituire la codifica, se lo strumento dispone della funzione di sostituzione
  - Prima di interrogare la tabella dei risultati, accertarsi che il client imposti la variabile di ambiente DB2CODEPAGE per forzare la code page client su una code page che sia compatibile con la dichiarazione di codifica del documento XML.
  - Modificare manualmente la dichiarazione di codifica del documento esportato nella codifica effettiva del documento (cioè, la code page del client)
- **Limitazione all'utilizzo di Unicode e Windows NT:** Su Windows NT, la locale del sistema operativo non può essere impostata su UTF-8. Per importare o esportare documenti, utilizzare le seguenti norme:
  - Quando si importano file e DTD con codifica UTF-8, impostare la code page del client su UTF-8 utilizzando:

    ```
    db2set DB2CODEPAGE=1208
    ```

    Usare questa tecnica durante:

    - L'inserimento di un DTD nella tabella db2xml.DTD_REF

- L'abilitazione di una colonna o di una raccolta
- La scomposizione di procedure memorizzate

– Quando si utilizzano UDF Content() o XMLxxxfromFile per importare documenti XML, i documenti devono essere codificati nella code page della locale di sistema operativo del server, che non può essere UTF-8.
– Quando si esporta un file XML dal database, impostare la code page client, mediante il seguente comando, in modo che DB2 codifichi in UTF-8 i dati risultanti:

```
db2set DB2CODEPAGE=1208
```

Usare questa tecnica durante:

- L'interrogazione della tabella dei risultati dopo la composizione
- L'estrazione di dati da una colonna XML mediante UDF di estrazione

– Quando si utilizzano UDF Content() o XMLxxxfromFile per esportare documenti XML in file sul file system del server, i documenti risultanti vengono codificati nella code page della locale di sistema operativo del server, che non può essere UTF-8.

Appendice D. Limiti di XML Extender

Il file DAD, le procedure memorizzate e le tabelle XML Extender presentano i seguenti limiti.

Tabella 58. Limiti per XML Extender

Valore	Limite
Tabella in una raccolta XML di scomposizione	1024 righe da ciascun documento XML scomposto
Parametri di procedure memorizzate:	
CLOB di documento XML	1 MB[2]
CLOB DAD (Document Access Definition)	100 KB[2]
collectionName	30
colName	30
dbName	18
defaultView	128
rootID	128
resultTabName	128
tablespace	128
tbName	128[1]
colonne di tabella db2xml.DTD_REF	
AUTHOR	128
CREATOR	128
UPDATOR	128
DTDID	128
CLOB	100 KB

Note:

1. Se *tbName* è qualificato da un nome di schema, l'intero nome (incluso il carattere separatore) non deve essere più lungo di 128 caratteri.
2. Tale dimensione può essere modificata. Per informazioni, consultare la sezione "Incremento del limite CLOB" a pagina 174.

I nomi possono subire un'espansione quando DB2 li converte dalla code page del client alla code page del database. Un nome potrebbe rientrare nel limite di dimensione sul client ma superare tale limite quando la procedura memorizzata riceve il nome convertito. Per ulteriori informazioni, fare riferimento alla sezione "National Language Support Application Development" del capitolo "Programming in Complex Environments" di *DB2 Application Development Guide*.

Informazioni particolari

Queste informazioni sono state sviluppate per i prodotti e i servizi offerti negli Stati Uniti. E' possibile che negli altri paesi l'IBM non offra i prodotti, le funzioni o i servizi illustrati in questo documento. Consultare il rappresentante IBM locale per informazioni sui prodotti o sui servizi disponibili nel proprio paese. Ogni riferimento relativo a prodotti, programmi o servizi IBM non implica che solo quei prodotti, programmi o servizi IBM possono essere utilizzati. In sostituzione a quelli forniti dall'IBM, è possibile usare prodotti, programmi o servizi funzionalmente equivalenti che non comportino violazione dei diritti di proprietà intellettuale o di altri diritti dell'IBM. E' comunque responsabilità dell'utente valutare e verificare la possibilità di utilizzare altri prodotti, programmi o servizi non IBM.

L'IBM può avere brevetti o domande di brevetti in corso relativi a quanto trattato nella presente pubblicazione. La fornitura di questa pubblicazione non implica la concessione di alcuna licenza su di essi. Chi desiderasse ricevere informazioni relative alle licenze può rivolgersi per iscritto a:

IBM Director of Commercial Relations
IBM Corporation
Schoenaicher Str. 220
D-7030 Boeblingen
Deutschland

Il seguente paragrafo non è valido per il Regno Unito o per tutti i paesi le cui leggi nazionali siano in contrasto con le disposizioni in esso contenute: L'INTERNATIONAL BUSINESS MACHINES CORPORATION FORNISCE QUESTA PUBBLICAZIONE NELLO STATO IN CUI SI TROVA SENZA ALCUNA GARANZIA, ESPLICITA O IMPLICITA, IVI INCLUSE EVENTUALI GARANZIE DI COMMERCIABILITÀ' ED IDONEITÀ' AD UNO SCOPO SPECIFICO. Alcuni stati non consentono la rinuncia a garanzie esplicite o implicite in determinate transazioni, quindi, la presente dichiarazione potrebbe non essere a voi applicabile.

Questa pubblicazione potrebbe contenere imprecisioni tecniche o errori tipografici. Le informazioni incluse in questo documento vengono modificate su base periodica; tali modifiche verranno incorporate nelle nuove edizioni della pubblicazione. L'IBM si riserva il diritto di apportare miglioramenti e/o modifiche al prodotto o al programma descritto nel manuale in qualsiasi momento e senza preavviso.

Coloro che detengono la licenza su questo programma e desiderano avere informazioni allo scopo di consentire: (i) uno scambio di informazioni tra programmi indipendenti e altri (compreso questo) e (ii) l'uso reciproco di tali informazioni, dovrebbero rivolgersi a:

IBM Corporation
J74/G4
555 Bailey Avenue
P.O. Box 49023
San Jose, CA 95161-9023
U.S.A.

Queste informazioni possono essere rese disponibili secondo condizioni contrattuali appropriate, compreso, in alcuni casi, il pagamento di un addebito.

Il programma su licenza descritto in questo manuale e tutto il materiale su licenza ad esso relativo sono forniti dall'IBM nel rispetto delle condizioni previste dalla licenza d'uso.

Le informazioni relative a prodotti non IBM sono state ottenute dai fornitori di tali prodotti. L'IBM non ha verificato tali prodotti e non può garantire l'accuratezza delle prestazioni. Eventuali commenti relativi alle prestazioni dei prodotti non IBM devono essere indirizzati ai fornitori di tali prodotti.

Le dichiarazioni relative a futuri intenti o obiettivi IBM sono soggette a modifiche senza preavviso.

LICENZA RELATIVA AI DIRITTI D'AUTORE:

Queste informazioni contengono programmi applicativi di esempio in lingua originale che illustrano le tecniche di programmazione su diverse piattaforme operative. Potete copiare, modificare e distribuire questi esempi di programmi sotto qualsiasi forma senza alcun pagamento alla IBM, allo scopo di sviluppare, utilizzare, commercializzare o distribuire i programmi applicativi in modo conforme alle API (Application Programming Interface) a seconda della piattaforma operativa per cui tali esempi di programmi sono stati scritti. Questi esempi non sono stati testati approfonditamente tenendo conto di tutte le condizioni possibili. La IBM, quindi, non può garantire o assicurare l'affidabilità, la praticità o il funzionamento di questi programmi.

Marchi

I seguenti termini sono marchi dell'IBM (International Business Machines Corporation) negli Stati Uniti o in altri paesi o in entrambi:

DB2
DB2 Extenders
DB2 UDB
(Universal Database)
IBM
IMS
Net.Data
OS/2
OS/390
OS/400
VTAM

Java e tutti i marchi a base Java sono marchi della Sun Microsystems, Inc. negli Stati Uniti e in altri paesi.

Microsoft, Windows, Windows NT ed il logo Windows sono marchi della Microsoft Corporation negli Stati Uniti e/o in altri paesi.

UNIX è un marchio registrato negli Stati Uniti e/o in altri paesi concesso su licenza esclusivamente tramite la X/Open Company Limited.

Nomi di altri prodotti, società e servizi possono essere marchi di altre società.

Glossario

API. Vedere anche *application programming interface*.

application programming interface (API).

(1) Un'interfaccia funzionale fornita dal sistema operativo o da un programma su licenza ordinabile separatamente. Un'API consente a un programma applicativo scritto in un linguaggio di alto livello di utilizzare dati specifici o funzioni del sistema operativo o del programma su licenza.

(2) In DB2, una funzione interna all'interfaccia, ad esempio l'API di richiamo del messaggio di errore.

(3) In DB2, una funzioni interna dell'interfaccia. Ad esempio, l'API del messaggio di richiamo errore.

Associazione RDB_node. L'ubicazione del contenuto di un elemento XML o il valore di un attributo XML definiti da RDB_node. XML Extender utilizza questa associazione per stabilire dove memorizzare o richiamare i dati XML.

Associazione SQL. Una definizione della relazione del contenuto di un elemento XML o del valore di un attributo XML con i dati relazionali, utilizzando uno o più istruzioni SQL e il modello dei dati XSLT. XML Extender utilizza questa definizione per determinare dove memorizzare o richiamare i dati XML. L'associazione SQL è definita dall'elemento SQL_stmt nel file DAD.

attribute_node. Una rappresentazione di un attributo di un elemento.

attributo. Vedere *attributo XML*.

attributo XML. Qualsiasi attributo specificato da ATTLIST nell'elemento XML nella DTD. XML Extender utilizza il percorso di ubicazione per identificare un attributo.

browser. Vedere *browser Web*.

browser Web. Un programma client che inizializza richieste a un server Web e visualizza le informazioni restituite dal server.

character large object (CLOB). Una stringa di caratteri a byte singolo, della lunghezza massima di 2 GB. I CLOB hanno una code page associata. Gli oggetti di testo che contengono caratteri a byte singolo, vengono memorizzati in un database DB2 come CLOB.

chiave esterna. Una chiave che fa parte della definizione di una restrizione di riferimento e che consiste di una o più colonne di una tabella dipendente.

chiave primaria. Una chiave univoca che fa parte della definizione di una tabella. Una chiave primaria e a chiave principale predefinita di una definizione di restrizione referenziale.

CLOB. Acronimo di Character large object.

colonna XML. Una colonna della tabella applicativa che è stata abilitata per gli UDT di XML Extender.

comporre. Per generare documenti XML dai dati relazionali contenuti in una raccolta XML.

condizione. Un specificazione per i criteri di selezione dei dati XML o per l'unione delle tabelle della raccolta XML.

convalida. Il processo di utilizzo di una DTD per verificare che il documento XML sia valido e per consentire ricerche strutturate nei dati XML. La DTD viene memorizzata nel magazzino DTD.

DAD. Vedere *Document access definition*.

datalink. Un tipo di dati DB2 che abilita i riferimenti logici dal database a un file memorizzato esternamente al database.

dati colonna. I dati memorizzati all'interno di una colonna DB2. Il tipo dei dati pò essere qualsiasi tipo supportato da DB2.

DBCLOB. Double-byte character large object.

Document Access Definition (DAD). Utilizzato per definire lo schema di indicizzazione di una colonna XML o lo schema di associazione di una raccolta XML. E' possibile utilizzare tale definizione per abilitare una colonna XML Extender di una raccolta XML formattata XML.

documento con formato regolare. Un documento XML che non contiene la DTD. Sebbene nella specifica XML, un documento con una DTD valida deve anche essere con formato regolare.

documento valido. Un documento XML con una DTD associata. Perché sia valido, il documento XML non può violare le regole sintattiche specificate nella relativa DTD.

Document type definition (DTD). Una serie di dichiarazioni relative agli elementi e agli attributi XML. La DTD definisce gli elementi utilizzati nel documento XML, l'ordine in cui vengono utilizzati e gli elementi che possono contenere altri elementi. E' possibile associare una DTD a un file DAD (document access definition) per convalidare i documenti XML.

double-byte character large object (DBCLOB). Una stringa caratteri dalla lunghezza massima di 2 GB, con caratteri a doppio byte o una combinazione di caratteri a doppio e singolo byte. I DBCLOB hanno una code page associata. Gli oggetti di testo che comprendono caratteri a doppio byte sono memorizzati in un database DB2 come DBCLOB.

DTD. (1) . (2) Vedere *Document type definition*.

DTD_REF table. Tabella di riferimento DTD.

EDI. Acronimo di Electronic Data Interchange.

Electronic Data Interchange (EDI). Uno standard per l'interscambio di dati elettronici per applicazioni B2B (business-to-business).

element_node. Una rappresentazione di un elemento. Un element_node può essere un elemento root o un elemento secondario.

elemento. Vedere *elemento XML*.

elemento root. L'elemento principale di un documento XML.

elemento XML. Un ELEMENT o una tag XML come specificato nella DTD XML. XML Extender utilizza il percorso di ubicazione per identificare un elemento.

espressione di percorso. Vedere *percorso di ubicazione*.

Extensible Stylesheet language (XSL). Un linguaggio utilizzato per i fogli di stile. XSL comprende due parti: un linguaggio per la trasformazione di documenti XML e un vocabolario XML per la specifica della formattazione semantica.

Extensive Stylesheet Language Transformation (XSLT). Un linguaggio utilizzato per trasformare i documenti XML in altri documenti XML. XSLT è progettato per l'utilizzo come parte di XSL che è un linguaggio di fogli di stile per XML.

file esterno. Un file che esiste in un file system esterno a DB2.

file system locale. Un file system presente in DB2

funzione cast. Una funzione utilizzata per convertire le istanze di un tipo dati (origine) in istanze di tipo dati differente (destinazione). Di solito, una funzione cast ha lo stesso nome del tipo di dati di destinazione. It Presenta un unico argomento il cui tipo è il tipo dati origine; il tipo di dati restituiti è il tipo dati di destinazione.

funzione cast predefinita. Esegue il tipo di base SQL in un UDT.

funzione scalare. Un'operazione SQL che produce un valore singolo da un altro valore ed è espressa come nome funzione seguito da un elenco di argomenti racchiusi tra parentesi.

funzione sovraccaricata. Un nome funzione per il quale esistono più istanze di funzione.

Id root. Un identificativo univoco che associa tutte le tabelle laterali alla tabella applicativa.

indicatore di posizione. Un'indicatore che è possibile utilizzare per individuare un oggetto. In DB2, l'indicatore di posizione LOB (large object block) è il tipo di dati che individua i LOB.

indice. Una serie di puntatori memorizzati in modo logico in base ai valori di una chiave. Gli indici consentono un accesso rapido ai dati e possono far rispettare l'univocità delle righe in una tabella.

indice del testo strutturale. Per creare un indice delle stringhe di testo nella struttura ad albero del documento XML, utilizzando DB2 Text Extender.

indicizzazione con struttura B. Lo schema di indice originario fornito dal motore DB2. Crea voci di indice con struttura B. Supporta i tipi di dati di base DB2.

interrogazione. Una richiesta di informazioni dal database basata su condizioni specifiche; ad esempio, la richiesta di un elenco di tutti i clienti in una tabella di clienti il cui saldo è superiore a 1000.

interrogazione secondaria. Un'istruzione SELECT completa utilizzata all'interno di una condizione di ricerca di un'istruzione SQL.

interscambio dati. La condivisione di dati tra applicazioni. XML supporta l'interscambio di dati senza elaborare la prima trasformazione di dati da un formato proprietario.

Java Database Connectivity (JDBC). Un'API (application programming interface) con le stesse caratteristiche di ODBC (Open Database Connectivity) progettata in modo specifico per l'utilizzo da parte di applicazioni database Java. Inoltre, per i database privi di driver JDBC, JDBC include un ponte JDBC-ODBC, che rappresenta un meccanismo di conversione da JDBC in ODBC; JDBC introduce l'API JDBC nelle applicazioni database Java e la converte in ODBC. JDBC è stato sviluppato dalla Sun Microsystems, Inc. e da vari partner e fornitori.

JDBC. Acronimo di Java Database Connectivity.

large object (LOB). Una sequenza di byte della lunghezza massima di 2 GB. Un LOB può essere di tre tipi: BLOB *(binary large object)*, CLOB *(character large object)* o DBCLOB *(double-byte character large object)*.

LOB. Acronimo di Large object.

Magazzino DTD. Una tabella DB2, denominata DTD_REF, in cui ciascuna riga della tabella rappresenta una DTD con le informazioni sui metadata relazionali.

metodo di accesso e memorizzazione. Associa i documenti XML a un database DB2 mediante due metodi di memorizzazione e di accesso principali: colonne XML e raccolte XML. Vedere anche *colonna XML* e *raccolta XML*.

modello dati XPath. La struttura ad albero utilizzata per creare modelli e per navigare in un documento XML utilizzando i nodi.

nodo. Nella suddivisione in partizioni del database, sinonimo di partizione database.

nodo database relazionale (RDB_node). Un nodo che contiene una o più definizione di elemento relative alle tabelle, e alle colonne e condizioni facoltative. Le tabelle e le colonne vengono utilizzate per definire come i dati XML sono memorizzati nel database. La condizione specifica i criteri di selezione dei dati XML o le modalità di unione delle tabelle della raccolta XML.

ODBC. Acronimo di Open Database Connectivity.

oggetto. Nella programmazione a oggetti, un'astrazione che consiste nei dati e nelle operazioni ad essi associati.

oggetto. Una raccolta di oggetti database quali tabelle, viste, indici o trigger. Consente una classificazione logica degli oggetti database.

oggetto XML. Equivalente di un documento XML.

Open Database Connectivity. Un'API standard per l'accesso ai dati dei sistemi di gestione database relazionali e non relazionali. Utilizzando questa API, le applicazioni del database possono accedere ai dati memorizzati nei sistemi di gestione database su vari computer anche se ciascun sistema di gestione utilizza un'interfaccia di programmazione e un formato di memorizzazione dati differenti. ODBC si basa sulla specifica CLI (call level interface) di X/Open SQL Access Group ed è stato sviluppato dalla Digital Equipment Corporation (DEC), Lotus, Microsoft e Sybase. Contrario di *Java Database Connectivity*.

origine dati. Un programma di gestione dei dati relazionali e non relazionali, locale o remoto, che supporta l'accesso ai dati tramite un driver ODBC che, a sua volta, supporta le API ODBC.

partizione. Una suddivisione della memoria con dimensione fissa.

percorso di ubicazione. Una sequenza di tag XML che identificano un elemento o un attributo XML. Il percorso di ubicazione identifica la struttura del documento XML, indicando il contesto dell'elemento e dell'attributo. Il percorso con singola barra (/) indica che il contesto è un documento intero. Il percorso di ubicazione è utilizzato nelle UDF di estrazione per identificare gli elementi e gli attributi da estrarre. Nel file DAD è utilizzato per specificare l'associazione tra un elemento o attributo XML e una colonna DB2 durante la definizione dello schema di indicizzazione relativo alla colonna XML. E' inoltre utilizzato da Text Extender per la ricerca del testo strutturale.

percorso di ubicazione assoluto. Il nome del percorso completo di un oggetto. Il nome di percorso assoluto inizia al livello superiore o all'indirizzario ″root″, contrassegnato dal carattere della barra (/) o della barra retroversa (\).

percorso di ubicazione semplice. Una sequenza di nomi di tipi di elementi collegati da una barra (/).

più ricorrenze. Indica se è possibile utilizzare più volte un elemento o attributo di colonna in un documento. Più ricorrenze è specificato nel file DAD.

predicato. Un elemento di una condizione di ricerca che esprime o sottintende un'operazione di confronto.

primo element_node. La rappresentazione dell'elemento root del documento XML nel file DAD.

procedura. Vedere *procedura memorizzata*.

procedure memorizzate. Un blocco di procedure e di istruzioni SQL incorporate memorizzate in un database che possono essere richiamate per nome. Le procedure memorizzate consentono di eseguire un programma applicativo in due parti. Una parte viene eseguita sul client e l'altra parte sul server. Ciò consente ad una chiamata di produrre più accessi al database.

Raccolta XML. Una raccolta di tabelle relazionali che presenta i dati per comporre o scomporre documenti XML o per la scomposizione.

RDB_node. Nodo database relazionale.

ricerca sezione. Consente la ricerca del testo all'interno di una sezione che può essere definita dall'applicazione. Per supportare la ricerca del testo strutturale, una sezione può essere definita dal percorso di ubicazione abbreviato di Xpath.

ricerca testo completo. Con DB2 Text Extender, la ricerca di stringhe di testo in qualsiasi punto senza tener conto della struttura del documento.

schema di associazione. Una definizione della rappresentazione dei dati XML in un database relazionale. Lo schema di associazione è specificato nel file DAD. XML Extender fornisce due tipi di schemi di associazione: *associazione SQL* e *associazione del nodo database relazionale (RDB_node)*.

scomporre. Raccoglie i documenti XML delle raccolte di tabelle relazionali in una raccolta XML.

serie di risultati. Una serie di righe restituite da una procedura memorizzata.

SQL integrata. Le istruzioni SQL codificate all'interno di un programma applicativo. Vedere *SQL statica*.

SQL statica. Le istruzioni SQL interne al programma e preparate durante il processo di preparazione del programma prima della sua esecuzione. Dopo essere stata preparata, un'istruzione SQL non viene modificata, anche se è possibile modificare i valori delle variabili host specificate dall'istruzione.

tabella dei risultati. Una tabella che contiene righe come risultato di un'interrogazione SQL o di un'esecuzione di una procedura memorizzata.

tabella di riferimento DTD (tabella DTD_REF). Una tabella che contiene le DTD utilizzate per convalidare i documenti XML e per consentire alle applicazioni la definizione di un file DAD. Gli utenti possono inserire le proprie DTD nella tabella DTD_REF. Questa tabella viene creata quando un database è abilitato per XML.

tabella laterale. Tabelle aggiuntive create da XML Extender per migliorare le prestazioni durante le ricerca di elementi o attributi presenti in una colonna XML.

tabella metadati. Vedere *tabella di supporto gestione*.

tabella utente. Una tabella creata e utilizzata da un'applicazione.

tabella XML. La tabella applicativa che include una o più colonne XML Extender.

tabelle di supporto gestione. Una tabella utilizzata da DB2 extender per elaborare le richieste dell'utente relative agli oggetti XML. Alcune tabelle di supporto di gestione identificano le tabelle e le colonne utente abilitate per un Extender. Altre tabelle di supporto di gestione contengono informazioni di attributi relative agli oggetti nelle colonne attivate. Sinonimo di tabella metadati.

table space. Un termine astratto che indica una raccolta di contenitori in cui vengono memorizzati gli oggetti database. Un table space fornisce un livello di azione indiretta tra un database e le tabelle memorizzate all'interno del database. Un table space:

- Dispone di spazio sulle unità di memorizzazione supporti.
- Dispone di tabelle create al suo interno. Queste tabelle utilizzano spazio nei contenitori riservato al table space. E' possibile memorizzare i dati, l'indice, i campi e le sezioni LOB di una tabella nello stesso table space oppure ripartirli in table space separati.

tag XML. Una tag di linguaggio markup XML valida, in modo particolare l'elemento XML. Le tag dei termini e gli elementi vengono utilizzati scambievolmente.

text_node. Una rappresentazione del testo CDATA di un elemento.

tipo di dati. Un attributo delle colonne e dei caratteri.

tipo esterno. Vedere *user-defined type*.

UDF. Vedere *user-defined function*.

UDT. Vedere *user-defined type*.

UDT XML. Un UDT DB2 fornita da XML Extender.

uniform resource locator (URL). Un indirizzo che richiama un server HTTP e facoltativamente una directory e file, ad esempio: http://www.ibm.com/data/db2/extenders.

UNION. Un'operazione SQL che combina i risultati di due istruzioni Select. UNION viene spesso utilizzato per unire elenchi di valori ottenuti da varie tabelle.

unione. Un'operazione relazionale che consente di richiamare i dati da due o più tabelle in base alla corrispondenza dei valori delle colonne.

URL. Acronimo di Uniform resource locator.

user-defined function (UDF). Una funzione definita nel sistema di gestione del database a cui è possibile far riferimento nelle interrogazioni SQL. Può essere una delle seguenti funzioni:

- Una funzione esterna in cui il corpo della funzione è scritto in un linguaggio di programmazione i cui argomenti sono valori scalari e produce un risultato scalare per ciascun richiamo.
- Una funzione originata, implementata da un'altra funzione interna o definita dall'utente già conosciuta nel DBMS. Questa funzione può essere una funzione scalare o una funzione di colonna (aggregante); restituisce un valore singolo da una serie di valori (ad esempio, MAX o AVG).

user-defined type (UDT). Un tipo di dati che non è nativo del manager database e che viene creato da un utente. Vedere *tipo esterno*.

vista di unione. Una vista DB2 creata dall'istruzione "CREATE VIEW" che unisce una o più tabelle.

vista predefinita. Una rappresentazione di dati in cui vengono unite la tabella XML e tutte le relative tabelle laterali.

XML. Acronimo di eXtensible Markup Language.

XML Path Language. Il linguaggio per l'indirizzamento di parti di un documento XML. XML Path Language è progettato per l'utilizzo con XSLT.

Ciascun percorso di ubicazione può essere espresso utilizzando la sintassi definita per XPath.

XML UDF. Una UDF DB2 fornita da XML Extender.

XPath. Il linguaggio per l'indirizzamento di parti di un documento XML.

XSL. Acronimo di XML Stylesheet Language.

XSLT. Acronimo di XML Stylesheet Language Transformation.

Indice analitico

D

E

F

G

I

J

L

M

N

O

P

R

S

T

U

V

W

X

Riservato ai commenti del lettore

IBM® DB2® Universal Database
XML Extender - Gestione e programmazione
Versione 7

Commenti relativi alla pubblicazione in oggetto potranno contribuire a migliorarla. Sono graditi commenti pertinenti alle informazioni contenute in questo manuale ed al modo in cui esse sono presentate. Si invita il lettore ad usare lo spazio sottostante citando, ove possibile, i riferimenti alla pagina ed al paragrafo.

Si prega di non utilizzare questo foglio per richiedere informazioni tecniche su sistemi, programmi o pubblicazioni e/o per richiedere informazioni di carattere generale.

Per tali esigenze si consiglia di rivolgersi al punto di vendita autorizzato o alla filiale IBM della propria zona oppure di chiamare il ″Supporto Clienti″ IBM al numero verde 167-017001.

I suggerimenti ed i commenti inviati potranno essere usati liberamente dall'IBM e dalla Selfin e diventeranno proprietà esclusiva delle stesse.

Commenti:

Si ringrazia per la collaborazione.

Per inviare i commenti è possibile utilizzare uno dei seguenti modi.

- Spedire questo modulo all'indirizzo indicato sul retro.
- Inviare un fax al numero: +39-081-660236
- Spedire una nota via email a: translationassurance@selfin.it

Se è gradita una risposta dalla Selfin, si prega di fornire le informazioni che seguono:

Nome

Indirizzo

Società

Numero di telefono

Indirizzo e-mail

Indicandoci i Suoi dati, Lei avrà l'opportunità di ottenere dal responsabile del Servizio di Translation Assurance della Selfin S.p.A. le risposte ai quesiti o alle richieste di informazioni che vorrà sottoporci. I Suoi dati saranno trattati nel rispetto di quanto stabilito dalla legge 31 dicembre 1996, n.675 sulla "Tutela delle persone e di altri soggetti rispetto al trattamento di dati personali". I Suoi dati non saranno oggetto di comunicazione o di diffusione a terzi; essi saranno utilizzati "una tantum" e saranno conservati per il tempo strettamente necessario al loro utilizzo.

Selfin S.p.A.
Translation Assurance

Via F. Giordani, 7

80122 NAPOLI

Printed in Denmark by IBM Danmark A/S