IBM® DB2® Universal Database

관리 안내서: 계획

버전 7

SA30-0990-01

IBM® DB2® Universal Database

관리 안내서: 계획

버전 7

SA30-0990-01

이 책의 정보와 지원하는 제품을 사용하기 전에 반드시 327 페이지의 『부록F. 주의사항』을 읽으십시오.

목차

이 책에 관하여

세 개의 볼륨으로 이루어진 관리 안내서는 Y2K 인증 DB2* 관계형 데이터베이스 관리 시스템(RDBMS) 제품을 사용하고 관리하는 데 필요한 정보를 제공합니다. 내용은 다음과 같습니다.

- 데이터베이스 설계에 관한 정보(*관리 안내서: 계획*에 있음)
- 데이터베이스의 구현 및 관리에 관한 정보(*관리 안내서: 구현*에 있음)
- 성능 개선을 위한 데이터베이스 환경의 구성 및 조정에 관련된 정보(*관리 안내서: 성능*에 있음)

이 책에서 다루는 여러 타스크는 서로 다른 인터페이스를 사용하여 수행될 수 있습니다.

- 그래픽 인터페이스로부터 데이터베이스를 액세스하고 조작할 수 있도록 해주는 **명령행 처리기**. 이 인터페이스에서 SQL문과 DB2 유틸리티 기능을 실행할 수도 있습니다. 이 책의 예제 대부분이 이 인터페이스를 사용하여 설명되어 있습니다. 명령행 처리기 사용에 대한 정보는 *Command Reference*를 참조하십시오.
- 응용프로그램 내에서 DB2 유틸리티 기능을 실행할 수 있도록 해주는 **API**(Application Programming Interface). Application Programming Interface 사용에 대해서는 *Administrative API Reference*에서 자세한 내용을 참조하십시오.
- 시스템 구성, 디렉토리 관리, 시스템 백업 및 복구, 작업 스케줄 작성, 미디어 관리와 같은 관리 타스크를 그래픽으로 수행할 수 있도록 해주는 **제어 센터**. 제어 센터에는 시스템간에 그래픽으로 데이터 복제를 설정할 수 있는 복제 관리가 있습니다. 그리고, 제어 센터는 그래픽 사용자 인터페이스를 통해 DB2 유틸리티 기능을 실행할 수 있습니다. 사용자 플랫폼에 따라 제어 센터를 호출하는 여러 가지 방법이 있습니다. 예를 들어, 명령행에서 `db2cc` 명령을 사용하여(OS/2에서) DB2 폴더에서 제어 센터 아이콘을 선택하거나 Windows 플랫폼에서 시작 패널을 사용하십시오. 좀더 자세한 내용을 보려면, 제어 센터 창의 **도움말** 풀다운 메뉴에서 **시작하기**를 선택하십시오. 제어 센터에서 **Visual Explain** 및 **성능 모니터** 도구가 호출됩니다.

관리 타스크를 실행하는 데 사용할 수 있는 기타 도구가 있습니다. 도구는 다음과 같습니다.

- 스크립트라고 불리는 작은 응용프로그램을 저장하는 스크립트 센터. 이 스크립트는 운영 체제 명령뿐 아니라 SQL 명령문 및 DB2 명령도 포함하고 있습니다.
- 다른 DB2 조작에서 기인한 메시지를 모니터하는 경고 센터
- 제어 센터, 경고 센터, 복제 관리의 설정값을 변경하는 도구 설정값
- 작업이 자동으로 수행되도록 스케줄을 잡는 저널
- 웨어하우스 오브젝트를 관리하는 Data Warehouse Center

이 책의 사용자

이 책은 기본적으로 지역 또는 원격 클라이언트에 의해 액세스될 데이터베이스를 설계, 구현 및 관리해야 하는 데이터베이스 관리자, 시스템 관리자, 보안 관리자 및 시스템 운영자를 대상으로 합니다. 또한, 프로그래머나 DB2 관계형 데이터베이스 관리 시스템의 관리 및 운영을 이해해야 하는 기타 다른 사용자에게도 도움이 됩니다.

이 책의 구성 방법

이 책에는 다음 주요 주제에 대한 정보가 들어 있습니다.

DB2 Universal Database의 세계

- 제1장 DB2 Universal Database 관리에서는 DB2 Universal Database에 대한 소개와 개요를 설명합니다.

데이터베이스 개념

- 제2장 기본 관계형 데이터베이스 개념에서는 저장 오브젝트 및 시스템 오브젝트를 포함하여 데이터베이스 오브젝트의 개요를 설명합니다.
- 제3장 연합 시스템에서는 단일 명령문에서 두 개 이상의 DBMS 또는 데이터베이스를 참조하는 사용자 SQL문과 응용프로그램을 지원하는 데이터베이스 관리 시스템(DBMS)인 연합 시스템을 설명합니다.

- 제4장 병렬 데이터베이스 시스템에서는 DB2로 사용 가능한 병렬 처리 유형에 대한 소개를 제공합니다.
- 제5장 데이터 웨어하우징 정보에서는 데이터 웨어하우징 및 데이터 웨어하우징 타스크의 개요를 제공합니다.
- 제6장 Spatial Extender 정보에서는 Spatial Extender 목적을 설명하고 처리하는 데이터를 설명하여 Spatial Extender를 소개합니다.

데이터베이스 설계

- 제7장 논리 데이터베이스 설계에서는 논리 데이터베이스 설계에 대한 개념 및 지침을 설명합니다.
- 제8장 물리적 데이터베이스 설계에서는 데이터 저장영역과 관련된 고려사항을 포함하여 물리 데이터베이스 설계에 대한 지침을 설명합니다.
- 제9장 분산 데이터베이스 설계에서는 단일 트랜잭션에서 다중 데이터베이스에 액세스하는 방법을 설명합니다.
- 제10장 트랜잭션 관리 프로그램 설계에서는 CICS와 같은 분산 트랜잭션 프로세싱 환경에서 데이터베이스를 사용하는 방법에 대해 설명합니다.
- 제11장 고가용성 및 장애 복구 지원 소개에서는 DB2에서 제공되는 고가용성 장애 복구 지원의 개요를 설명합니다.

부록 및 끝머리

- 부록A. 이름 지정 규칙에서는 데이터베이스 및 오브젝트를 이름 지정할 때 지켜야 할 규칙을 설명합니다.
- 부록B. 데이터베이스 이주 계획에서는 버전 7로의 데이터베이스 이주 정보를 설명합니다.
- 부록C. 릴리스간 비 호환성에서는 버전 7을 포함하여 릴리스간에 도입된 비 호환성에 대해 설명합니다.
- 부록D. 자국어 지원(NLS)에서는 국가, 언어 및 코드 페이지의 정보가 포함된 DB2 자국어 지원(NLS)을 소개합니다.
- 부록E. DB2 라이브러리 사용에서는 마법사, 온라인 도움말, 메시지 및 책을 포함하여 DB2 라이브러리 구조에 대한 정보를 제공합니다.

다른 볼륨의 관리 안내서에 대한 간단한 개요

관리 안내서: 구현

*관리 안내서: 구현*은 데이터베이스 설계의 구현을 설명합니다. 관리 안내서: 구현에 있는 특정 장 및 부록에 대해 다음에서 간략히 설명합니다.

제어 센터를 사용하여 관리

- "GUI 도구를 사용한 DB2 관리"에서는 데이터베이스 관리에 사용되는 그래픽 사용자 인터페이스(GUI) 도구를 설명합니다.

설계 구현

- "데이터베이스를 작성하기 전에"에서는 데이터베이스를 작성하기 전에 필요한 전제조건에 대해 설명합니다.
- "데이터베이스 작성"에서는 데이터베이스 및 관련된 데이터베이스 오브젝트의 작성과 연관된 타스크를 설명합니다.
- "데이터베이스 변경"에서는 데이터베이스를 변경하기 전에 수행해야 하는 작업과 데이터베이스나 관련된 데이터베이스 오브젝트의 수정 또는 삭제와 연관된 타스크를 설명합니다.

데이터베이스 보안

- "데이터베이스 액세스 제어"에서는 데이터베이스 자원에 대한 액세스를 제어하는 방법에 대해 설명합니다.
- "DB2 활동 감사"에서는 원하지 않거나 예상치 않은 데이터 액세스를 검출하고 모니터하는 방법에 대해 설명합니다.

데이터 이동

- "데이터의 이동을 위한 유틸리티"에서는 데이터를 이동시키며 직접 *데이터 이동 유틸리티 안내 및 참조서* 책으로 안내하는 여러 가지 방법을 한 페이지에 걸쳐 소개합니다.

복구

- ″데이터베이스 복구″는 데이터베이스 백업, 복원 및 롤 포워드의 개념을 한 페이지에 걸쳐 소개합니다. 보다 광범위한 정보는 *데이터 복구 및 고가용성 안내 및 참조서*를 참조하십시오.

부록 및 끝머리

- ″DCE 디렉토리 서비스 사용″에서는 DCE 디렉토리 서비스를 사용할 수 있는 방법에 대한 정보를 설명합니다.
- ″데이터베이스 복구를 위한 User Exit″에서는 데이터베이스 로그 파일과 함께 User Exit 프로그램을 사용하는 방법을 설명하며, 몇몇 User Exit 샘플 프로그램을 설명합니다.
- ″다중 데이터베이스 파티션 서버로 명령 발행″에서는 파티션된 데이터베이스 환경의 모든 파티션으로 명령을 송신하는 *db2_all*과 *rah* 쉘 스크립트를 사용하는 방법을 설명합니다.
- ″Windows NT용 DB2가 Windows NT 보안을 사용하여 작업하는 방법″에서는 DB2가 Windows NT 보안을 사용하여 작업하는 방법을 설명합니다.
- ″Windows NT 성능 모니터 사용″에서는 Windows NT 성능 모니터로 DB2를 등록하고, 성능 정보를 사용하는 것에 대한 정보를 제공합니다.
- ″Windows 2000 데이터베이스 파티션 서버 및 Windows NT 작업″에서는 Windows NT 또는 Windows 2000에서 데이터베이스 파티션과 작업할 수 있는 유틸리티에 대한 정보를 제공합니다.
- ″다중 논리 노드 구성″에서는 파티션된 데이터베이스 환경에서 다중 논리 노드를 구성하는 방법을 설명합니다.
- ″고속 노드간 통신″에서는 DB2 Universal Database와 함께 가상 인터페이스 아키텍처를 사용할 수 있는 방법을 설명합니다.
- ″LDAP(Lightweight Directory Access Protocol) 디렉토리 서비스″에서는 LDAP 디렉토리 서비스를 사용할 수 있는 방법에 대한 정보를 제공합니다.
- ″제어 센터 확장″에서는 새 조치를 포함한 새 도구 막대 버튼, 새 오브젝트 정의 및 새 조치 정의를 추가하여 제어 센터를 확장하는 방법에 대한 정보를 제공합니다.

관리 안내서: 성능

*관리 안내서: 성능*은 성능 문제를 설명합니다. 즉, 응용프로그램의 성능 및 DB2 Universal Database 제품의 자체의 성능에 대한 설정, 테스트 및 개선에 관련된 주제와 문제입니다. 관리 안내서: 구현에 있는 특정 장 및 부록에 대해 다음에서 간략히 설명합니다.

성능 개요

- "성능 요소"에서는 DB2 UDB 성능의 관리 및 개선에 대한 개념 및 고려사항을 소개합니다.
- "아키텍처 및 프로세스 개요"에서는 주요 DB2 Universal Database 아키텍처 및 프로세스를 소개합니다.

응용프로그램 성능 조정

- "응용프로그램 고려사항"에서는 응용프로그램을 설계할 때 데이터베이스 성능을 향상시키기 위한 몇몇 기술에 대해 설명합니다.
- "환경상의 고려사항"에서는 데이터베이스 환경을 설정할 때 데이터베이스 성능을 향상시키기 위한 몇몇 기술에 대해 설명합니다.
- "시스템 카탈로그 통계"에서는 사용자 데이터에 대한 통계를 수집하고 이를 사용하여 최적의 성능을 보장할 수 있는 방법을 설명합니다.
- "SQL 컴파일러의 이해"에서는 SQL 컴파일러로 컴파일할 때 SQL문에 발생하는 현상에 대해 설명합니다.
- "SQL Explain 기능"에서는 SQL 컴파일러가 데이터에 액세스하기 위해 취하는 선택 항목을 살펴볼 수 있는 Explain 기능에 대해 설명합니다.

시스템의 조정 및 구성

- "수행시의 성능"에서는 데이터베이스 관리 프로그램이 메모리를 사용하는 방법 및 런타임 성능에 영향을 주는 기타 고려사항에 대한 개요를 설명합니다.
- "조정자(Governor) 사용"에서는 조정자를 사용하여 데이터베이스 관리의 일부 양상을 제어하는 방법에 대해 설명합니다.
- "프로세서 추가를 통해 구성 조정"에서는 데이터베이스 시스템의 크기 증가에 관련된 일부 고려사항 및 타스크에 대해 설명합니다.

- ″데이터베이스 파티션에 걸친 데이터 재분산″에서는 파티션된 데이터베이스 환경에서 파티션에 데이터를 재분배하는 데 필요한 타스크에 대해 설명합니다.
- ″벤치마크 테스트″에서는 벤치마크 테스트의 개요 및 벤치마크 테스트 수행 방법을 설명합니다.
- ″DB2 구성″에서는 데이터베이스 관리 프로그램 및 데이터베이스 구성 파일과 구성 매개변수에 대한 값에 대해 설명합니다.

부록 및 끝머리

- ″DB2 레지스트리 및 환경 변수″에서는 프로파일 등록 값 및 환경 변수를 설명합니다.
- ″Explain 테이블 및 정의″에서는 DB2 Explain 기능에서 사용되는 테이블 및 해당 테이블 작성 방법을 설명합니다.
- ″SQL Explain 도구″에서는 DB2 explain 도구인 db2expln 및 dynexpln의 사용 방법을 설명합니다.
- ″db2exfmt -- Explain 테이블 형식 도구″는 DB2 Explain 도구를 사용하여 Explain 테이블 데이터를 형식화하는 방법을 설명합니다.

제1부 DB2 Universal Database의 세계

제1장 DB2 Universal Database 관리

DB2는 광범위한 하드웨어 구성을 수행할 수 있는 융통성을 제공합니다. 특정 DB2 제품 구성을 가진 사용자의 하드웨어 및 응용프로그램의 요구사항에 가장 잘 부합하는 방법을 선택할 수 있습니다.

또한, DB2는 데이터베이스 환경의 서로 다른 복잡한 레벨을 지원하여 각 환경에 대한 특정 고려사항 및 타스크가 있습니다. 이들은 *관리 안내서* 및 DB2 라이브러리의 다른 책에서 설명됩니다(307 페이지의 『부록E. DB2 라이브러리 사용』 참조). 어떤 경우, 이 책의 전체 절은 특정 환경에 대해서만 적합합니다. 이 책의 서문을 읽은 후에는(″이 책에 관하여″), *관리 안내서*(*관리 안내서: 구현* 및 *관리 안내서: 성능*)의 이 볼륨과 다른 볼륨의 어떤 장이 사용자의 비지니스 요구에 적합한지 이해하게 됩니다.

관계형 데이터베이스 관리 시스템(RDBMS)이나 DB2를 처음 사용하는 경우, ″기본 관계형 데이터베이스 개념″절이 도움이 됩니다. 이 개념과 친숙하거나 검토할 필요가 없는 경우, 이 절을 건너뛰고 다음과 같은 고급 주제를 다루는 절로 직접 이동하십시오.

- 연합 시스템. 이 절에서는 단일 명령문에서 두 개 이상의 DBMS 또는 데이터베이스를 참조하는 사용자 SQL문과 응용프로그램을 지원하는 데이터베이스 관리 시스템(DBMS)을 설명합니다.
- 병렬 데이터베이스 시스템. 이 절에서는 DB2로 사용 가능한 병렬 처리 유형에 대한 소개를 제공합니다. 데이터베이스 조회와 같은 타스크의 구성요소는 극적으로 성능을 향상시키기 위해 병렬로 실행될 수 있습니다.
- 분산 트랜잭션 프로세싱. 이 절에서는 단일 트랜잭션에서 여러 데이터베이스에 액세스할 수 있는 방법과 분산 트랜잭션 프로세싱 환경에서 데이터베이스를 사용할 수 있는 방법을 설명합니다.

DB2는 다음과 같이 대부분의 특정 데이터 관리 필요성을 처리할 수 있습니다.

- *복제*는 일반 기준으로 여러 원격 데이터베이스로 데이터를 복사하도록 허용합니다. 마스터 데이터베이스로부터의 갱신이 자동으로 다른 데이터베이스로 복사되

게 할 필요가 있는 경우, DB2의 복제 기능을 사용하여 어떤 데이터가 복사되어야 하는지, 데이터가 복사되어야 하는 데이터베이스 테이블은 어떤 것인지 그리고 얼마나 자주 갱신사항을 복사해야 하는지 지정할 수 있습니다. DB2의 복제 기능을 사용하려는 경우, *복제 안내 및 참조서*에서 자세한 내용을 참조하십시오. DB2 데이터 복제의 개념을 소개하고, 복제 환경을 계획, 구성 및 관리하는 방법을 설명합니다.

- *데이터 웨어하우징*에서는 "정보용 데이터"나 조작용 데이터에서 발췌된 후 일반 사용자 결정을 위해 변환되는 데이터의 저장을 작성할 수 있습니다. 예를 들어, 데이터 웨어하우징 도구는 조작용 데이터베이스에서 모든 판매 데이터를 복사하고, 데이터를 요약하기 위해 계산을 수행하고, 요약된 데이터를 개별 데이터베이스 내의 목표로 기록하십시오. 조작 데이터베이스에 영향을 미치지 않고 개별 데이터베이스(*웨어하우스*)를 조회할 수 있습니다. 데이터 웨어하우징에 대해서는 *Data Warehouse Center 관리 안내서*에서 자세한 내용을 참조하십시오.
- *지리학적 정보 시스템*(GIS)은 Spatial Extender에서 작성될 수 있습니다. GIS는 지리학적 기능에 대한 공간 정보를 생성하고 분석하게 하는 오브젝트, 데이터 및 응용프로그램의 복합체입니다. Spatial Extender에서 지리학적인 기능은 테이블 또는 뷰에서 행별로 표시되거나 그러한 행의 부분에 의해 표시될 수 있습니다. Spatial Extender 사용에 대해서는 *Spatial Extender 사용자 안내 및 참조서*에서 자세한 내용을 참조하십시오.

또한 *관리 안내서: 계획*에서는 DB2의 논리 데이터베이스 설계 및 실제 데이터베이스 설계 고려사항을 포함하여 데이터베이스 설계를 다룹니다. 또한 데이터베이스 이주 계획, 사용자 응용프로그램(*비 호환성*은 이미 있는 응용프로그램에서 사용될 경우, 다른 결과를 초래하여 응용프로그램을 변경하거나 성능을 감소시키게 되는 방법으로, 이전 릴리스의 DB2에서 작동하던 것과는 다르게 작동하는 DB2 Universal Database의 일부입니다.)에 영향을 미칠 수 있는 비 호환성을 식별하며, 자연 언어 지원(NLS) 활용과 같은 기타 계획 문제가 설명됩니다.

*관리 안내서: 구현*에서는 사용자 데이터베이스 설계를 구현하는 세부사항을 설명합니다. 데이터베이스, 데이터베이스 보안을 작성 및 변경하는 것과 제어 센터, DB2 그래픽 사용자 인터페이스를 사용하여 DB2를 관리하는 것이 주제에 포함됩니다.

*관리 안내서: 성능*에서는 응용프로그램의 성능 및 DB2 자체의 성능에 대한 설정, 테스트 및 개선에 관련된 주제와 문제를 다룹니다.

제2부 데이터베이스 개념

제2장 기본 관계형 데이터베이스 개념

이 절에서는 다음 주제를 다룹니다.

- 『데이터베이스 오브젝트 개요』
- 14 페이지의 『저장 오브젝트 개요』
- 21 페이지의 『시스템 오브젝트 개요』
- 23 페이지의 『데이터에 대한 비지니스 규칙』
- 27 페이지의 『데이터베이스 복구』
- 28 페이지의 『데이터베이스에서 테이블 재구성』
- 29 페이지의 『DB2 보안 개요』

데이터베이스 오브젝트 개요

이 절에서는 다음 키 데이터베이스 오브젝트의 개요를 제공합니다.

- 인스턴스
- 데이터베이스
- 노드 그룹
- 테이블
- 뷰
- 색인
- 스키마
- 시스템 카탈로그 테이블

10 페이지의 그림1에서는 이 일부 오브젝트들 간의 관계를 설명합니다. 또한, 테이블 공간에 저장된 테이블, 색인 및 long 데이터도 보여줍니다.

그림 1. 여러 데이터베이스 오브젝트간의 관계

인스턴스

인스턴스(종종 *데이터베이스 관리 프로그램*이라고 함)는 데이터를 관리하는 DB2 코드입니다. 데이터의 내용을 제어하고 할당된 시스템 자원을 관리합니다. 각 인스턴스는 완전한 환경입니다. 여기에는 주어진 병렬 데이터베이스 시스템용으로 정의된 모든 데이터베이스 파티션이 들어 있습니다(37 페이지의 『제4장 병렬 데이터베

이스 시스템』 참조). 하나의 인스턴스에는 다른 인스턴스가 액세스할 수 없는 고유의 데이터베이스가 있고, 데이터베이스 파티션은 같은 시스템 디렉토리를 공유합니다. 같은 머신(시스템)의 다른 인스턴스로부터 분리된 보안도 가지고 있습니다.

데이터베이스

*관계형 데이터베이스*는 테이블의 콜렉션으로서 데이터를 나타냅니다. 테이블은 정의된 컬럼 수와 임의 행 수로 구성됩니다. 각 데이터베이스는 데이터의 논리 및 물리 구조, 데이터베이스용으로 할당된 매개변수 값이 들어 있는 구성 파일 그리고 처리 중인 트랜잭션과 아카이브가능한 트랜잭션이 있는 복구 로그를 설명하는 시스템 카탈로그 테이블 세트를 포함합니다.

노드 그룹

노드 그룹은 하나 이상의 데이터베이스 파티션의 세트입니다. 데이터베이스에 테이블을 작성하려면, 테이블 공간이 저장될 노드 그룹을 먼저 작성한 다음 테이블이 저장될 테이블 공간을 작성합니다. 노드 그룹에 대해서는 38 페이지의 『노드 그룹 및 데이터 파티션』에서 자세한 내용을 참조하십시오. 데이터베이스 파티션의 정의는 37 페이지의 『제4장 병렬 데이터베이스 시스템』에서 자세한 내용을 참조하십시오. 테이블 공간에 대해서는 15 페이지의 『테이블 공간』에서 자세한 내용을 참조하십시오.

테이블

관계형 데이터베이스는 테이블의 콜렉션으로서 데이터를 나타냅니다. *테이블*은 컬럼과 행이 논리적으로 배열된 데이터로 이루어져 있습니다. 모든 데이터베이스와 테이블 데이터는 테이블 공간으로 지정됩니다. 테이블 공간에 대해서는 15 페이지의 『테이블 공간』에서 자세한 내용을 참조하십시오. 테이블의 데이터는 논리적으로 관련되고, 관계는 테이블간에 정의될 수 있습니다. 데이터는 *관계*를 호출한 수학적 규칙과 연산을 기초로 하여 볼 수 있고 조정될 수 있습니다.

테이블 데이터는 관계형 데이터베이스에서 데이터를 정의하고 조작하는 표준 언어인 SQL(*SQL 참조서* 참조)을 통해 액세스됩니다. 응용프로그램 또는 사용자가 데이터베이스의 데이터를 검색하는 데 *조회*가 사용됩니다. 조회시 다음 양식의 명령문을 작성하기 위해 SQL이 사용됩니다.

```
SELECT <data_name> FROM <table_name>
```

뷰

*뷰*는 데이터를 유지보수할 필요없이 데이터를 표시하는 효과적인 방법입니다. 뷰는 실제 테이블이 아니며 영구 저장영역을 요구하지 않습니다. "가상 테이블"이 작성되어 사용됩니다.

뷰는 근거한 테이블에 포함된 컬럼이나 행 모두나 일부를 포함할 수 있습니다. 예를 들어, 부서 테이블 및 사원 테이블을 뷰에서 조인하여, 특정 부서에 있는 모든 사원을 나열할 수 있습니다.

그림2는 테이블 및 뷰 사이의 관계를 나타냅니다.

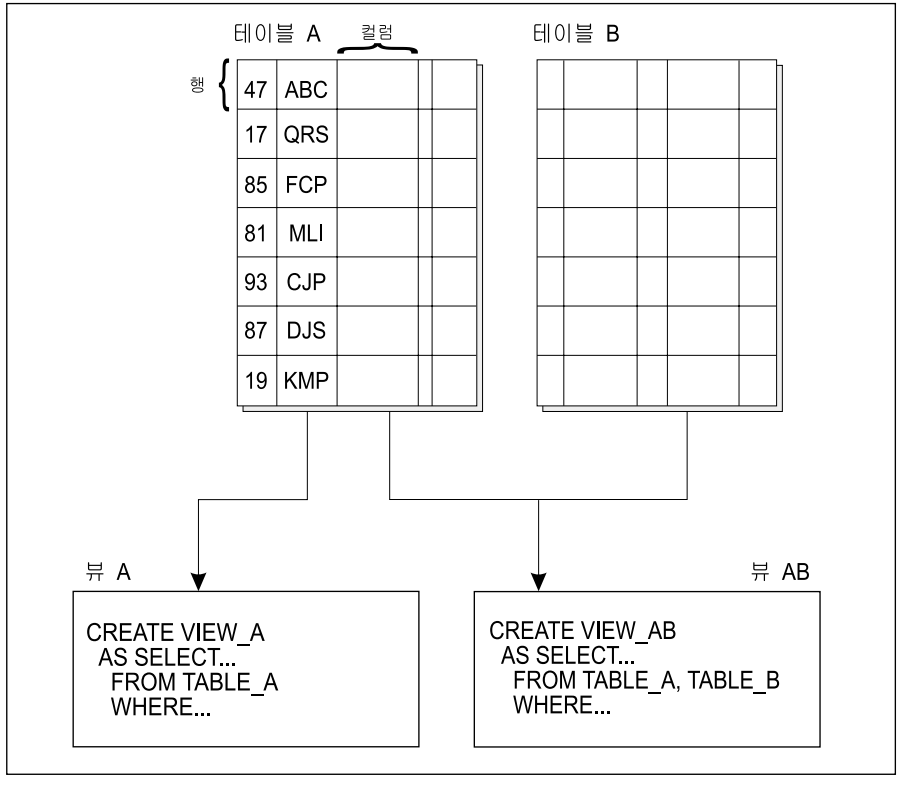

그림 2. 테이블 및 뷰 사이의 관계

색인

*색인*은 키 세트이며, 각각은 테이블에 있는 행을 지시합니다. 예를 들어, 그림3에 있는 테이블 A는 테이블에 있는 사원 번호에 근거한 색인을 갖습니다. 이 키 값은 테이블에서 행에 대한 포인터를 제공합니다. 예를 들어, 사원 번호 19는 사원 KMP를 지시합니다. 색인은 포인터를 통해 데이터에 대한 직접 경로를 작성하여 테이블에 있는 행에 대해 더 효과적인 액세스를 허용합니다.

SQL *최적화 알고리즘*은 테이블에 있는 데이터에 액세스하는 가장 효과적인 방법을 자동으로 선택합니다. 최적화 알고리즘은 데이터에 대한 가장 빠른 액세스 경로를 판별할 때 색인을 고려합니다.

고유 색인은 색인 키의 고유성을 보장하도록 작성될 수 있습니다. *색인 키*는 색인이 정의된 컬럼의 순서화된 콜렉션 또는 컬럼입니다. 고유 색인을 사용하여 색인 컬럼이나 컬럼에 있는 각 색인 키의 값이 고유하도록 보장합니다. 23 페이지의 『데이터에 대한 비지니스 규칙』에서는 키와 색인에 대한 자세한 내용을 설명합니다.

그림3은 색인 및 테이블 사이의 관계를 나타냅니다.

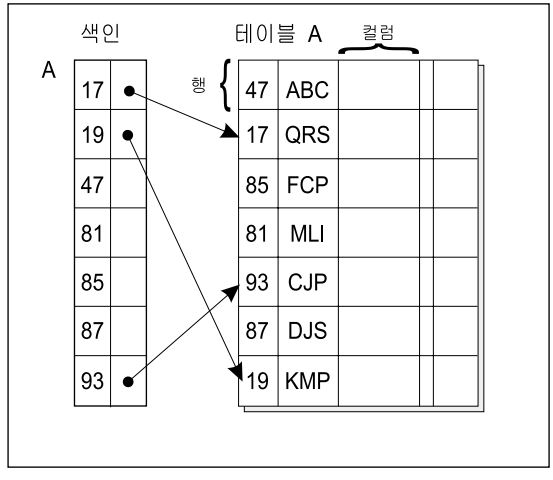

그림3. 색인 및 테이블 사이의 관계

스키마

*스키마*는 테이블 및 다른 데이터베이스 오브젝트를 그룹화하도록 돕는 사용자 ID와 같은 식별자입니다. 스키마는 개인에 의해 소유되며, 소유자는 데이터와 데이터 내의 오브젝트에 대한 액세스를 제어할 수 있습니다.

또한, 스키마는 데이터베이스의 오브젝트입니다. 스키마에 있는 첫 번째 오브젝트가 작성될 때 자동으로 작성될 수 있습니다. 그러한 오브젝트는 테이블, 색인, 뷰, 패키지, 구별 유형, 기능 또는 트리거와 같이 스키마 이름으로 규정될 수 있는 것입니다. 스키마가 자동으로 작성되는 경우 IMPLICIT_SCHEMA 권한을 가져야 하거나, 명시적으로 스키마를 작성할 수 있습니다.

스키마 이름은 두 부분으로 된 오브젝트 이름의 첫 번째 부분으로서 사용됩니다. 오브젝트가 작성되면, 특정 스키마로 지정할 수 있습니다. 스키마를 지정하지 않은 경우, 보통 오브젝트를 작성한 사람의 사용자 ID인 기본 스키마에 지정됩니다. 두 번째 부분은 오브젝트의 이름입니다. 예를 들어, Smith라는 사용자는 SMITH.PAYROLL이라는 테이블을 갖습니다.

시스템 카탈로그 테이블

각 데이터베이스는 데이터의 논리 및 물리 구조를 설명하는 *시스템 카탈로그 테이블* 세트를 포함합니다. DB2는 각 데이터베이스의 시스템 카탈로그 테이블의 확장 세트를 작성하고 유지보수합니다. 이 테이블에는 이 오브젝트에서 사용자가 가진 권한에 대한 보안 정보는 물론, 사용자 테이블, 뷰 및 색인과 같은 데이터베이스 오브젝트의 정의에 대한 정보가 들어 있습니다. 이 테이블은 데이터베이스가 작성될 때 작성되어 정상 조작의 과정 중 갱신됩니다. 사용자가 테이블을 명시적으로 작성하거나 삭제할 수는 없지만, 카탈로그 뷰를 사용하여 해당 내용을 조회하거나 볼 수는 있습니다.

저장 오브젝트 개요

다음과 같은 데이터베이스 오브젝트는 사용자 시스템에 데이터를 저장하는 방법과 성능이 개선될 수 있는 방법(데이터 액세스와 관련)을 정의하게 합니다.

- 테이블 공간
- 컨테이너

- 버퍼 풀

테이블 공간

데이터베이스는 *테이블 공간*이라는 부분으로 구성됩니다. 테이블 공간은 테이블을 저장할 장소입니다. 테이블 작성시 나머지 테이블 데이터와는 별개로 색인과 대형 오브젝트(LOB) 데이터와 같은 특정 오브젝트를 유지하도록 결정할 수 있습니다. 또한 테이블 공간은 하나 이상의 물리적 저장 장치에 걸쳐 있을 수 있습니다. 다음과 같은 그림은 테이블 공간에서 데이터를 전개하는 유연성의 일부를 보여줍니다.

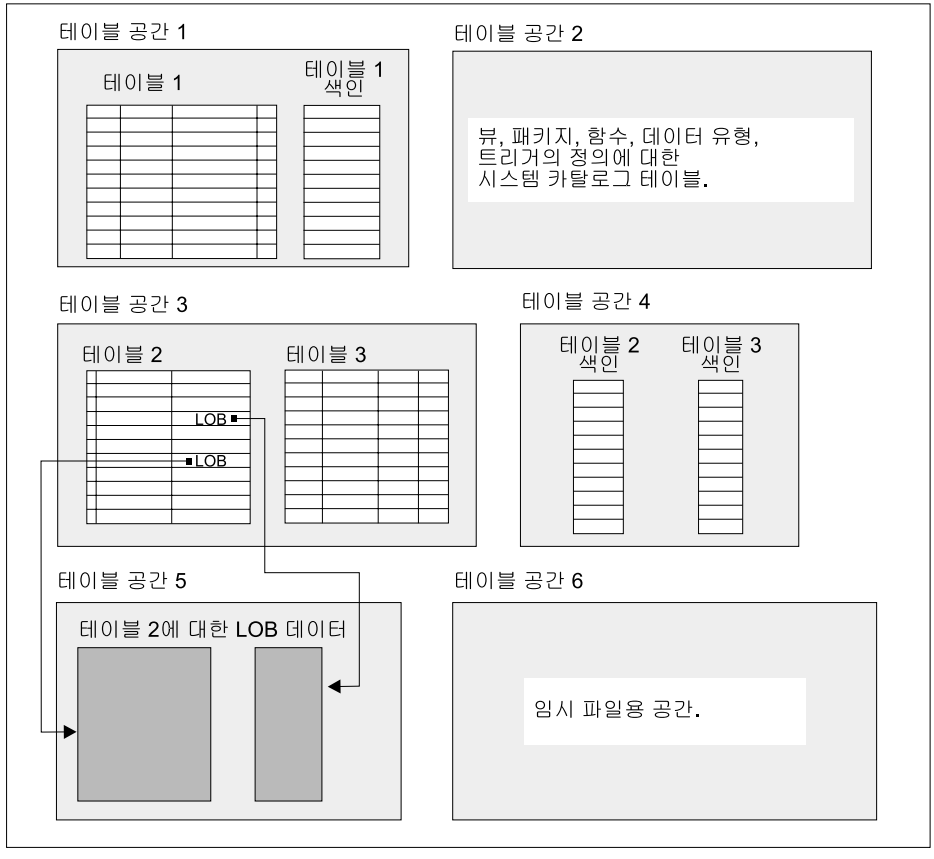

그림 4. 테이블 공간

테이블 공간은 노드 그룹에 상주합니다(11 페이지의 『노드 그룹』 참조). 테이블 공간 정의 및 속성은 데이터베이스 시스템 카탈로그에 기록됩니다(14 페이지의 『시스템 카탈로그 테이블』 참조).

컨테이너는 테이블 공간에 지정됩니다. *컨테이너*는 물리적 저장영역(예: 파일 또는 장치)의 할당입니다.

테이블 공간은 시스템 관리 공간(SMS) 또는 데이터베이스 관리 공간(DMS) 중 하나일 수 있습니다. SMS 테이블 공간의 경우, 각 컨테이너는 운영 체제의 파일 공간에 있는 디렉토리이며, 운영 체제의 파일 관리 프로그램이 저장영역 공간을 제어합니다. DMS 테이블 공간의 경우, 각 컨테이너는 사전 할당된 고정 *크기* 파일이나 디스크와 같은 물리 장치 중 하나이며, 데이터베이스 관리 프로그램은 저장영역 공간을 제어합니다.

17 페이지의 그림5는 테이블, 테이블 공간 및 두 개의 공간 유형 사이의 관계를 나타냅니다. 또한, 테이블 공간에 저장된 테이블, 색인 및 long 데이터도 보여줍니다.

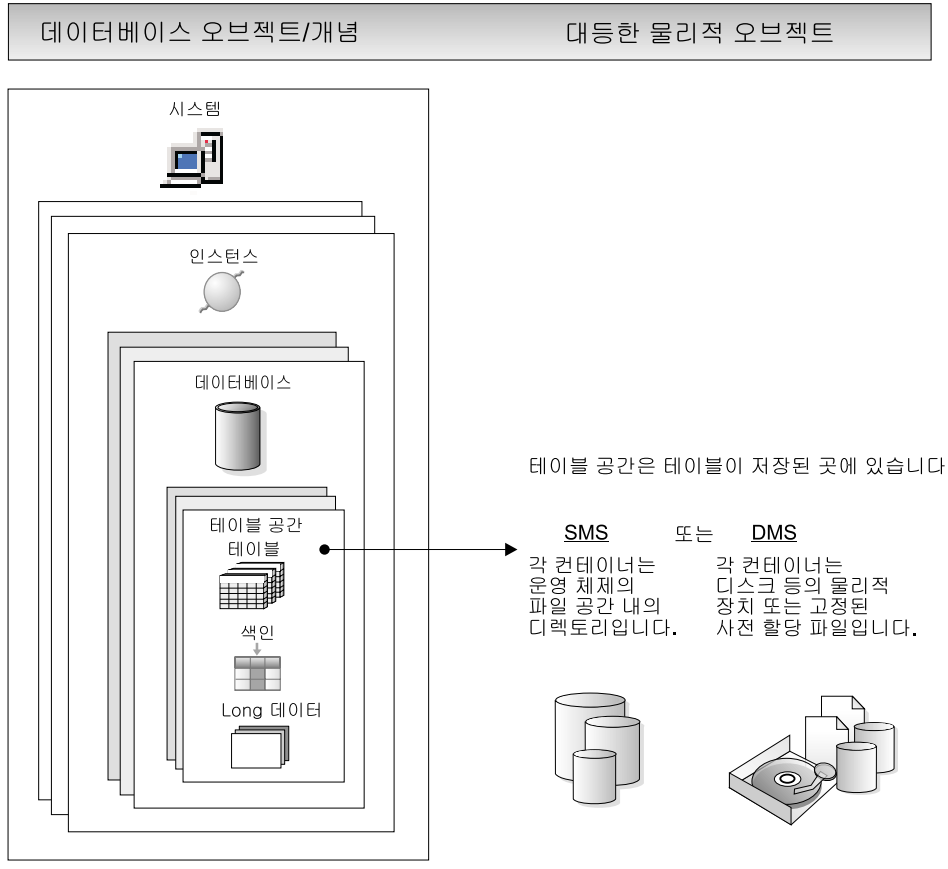

그림5. 테이블 공간 및 테이블

18 페이지의 그림6에서는 *regular*, *temporary* 및 *long*과 같은 세 가지 테이블 공간 유형을 보여줍니다.

사용자 데이터가 들어 있는 테이블은 일반 테이블 공간에 있습니다. 기본 사용자 테이블 공간은 USERSPACE1이라고 합니다. 또한 색인은 일반 테이블 공간에 저장됩니다. 시스템 카탈로그 테이블은 일반 테이블 공간에 있습니다. 기본 시스템 카탈로그 테이블 공간은 SYSCATSPACE라고 합니다.

다중 매체 오브젝트와 같이 long 필드 데이터나 long 오브젝트 데이터가 들어 있는 테이블은 long 테이블 공간에 있습니다.

*임시 테이블 공간*은 시스템이나 사용자로서 분류됩니다. *시스템 임시 테이블 공간*은 정렬, 테이블 재구성, 색인 작성 및 테이블 조인과 같은 SQL 조작 중 필요한 내부 임시 데이터

를 저장하는 데 사용됩니다. 임의의 시스템 임시 테이블 공간을 작성할 수 있지만, 대부분의 사용자 테이블이 사용하는 페이지 *크기*를 사용하여 오직 하나만 작성하도록 권장합니다. *기본 시스템 임시 테이블 공간*은 TEMPSPACE1이라고 합니다. *사용자 임시 테이블 공간*은 응용프로그램 임시 데이터를 저장하는 선언된 전역 임시 테이블을 저장하기 위해 사용됩니다. 사용자 임시 테이블 공간은 데이터베이스 작성시 기본값으로 작성되지 *않습니다*.

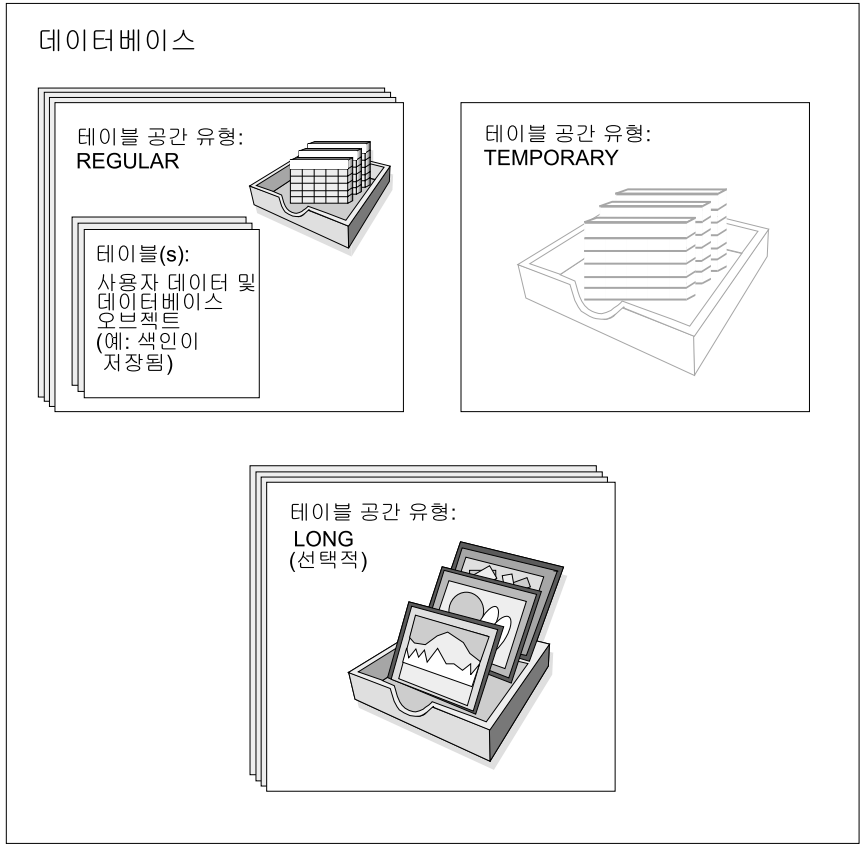

그림 6. 세 개의 테이블 공간 유형

컨테이너

*컨테이너*는 물리적 저장 장치입니다. 컨테이너는 디렉토리 이름, 장치 이름 또는 파일 이름에 의해 식별될 수 있습니다.

컨테이너는 테이블 공간에 지정됩니다. 단일 테이블 공간은 여러 컨테이너에 걸쳐 있을 수 있지만, 각 컨테이너는 오직 하나의 테이블 공간에만 속할 수 있습니다.

그림7은 데이터베이스 내의 테이블과 테이블 공간과의 관계, 연관된 컨테이너 및 디스크를 설명합니다.

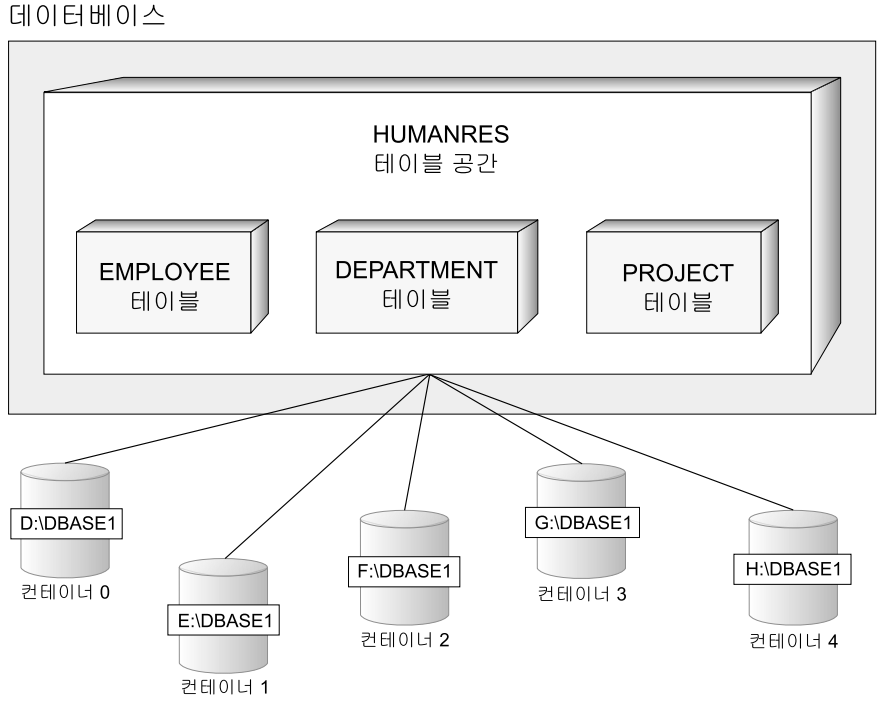

그림 7. 데이터베이스 내의 테이블 공간 및 테이블

EMPLOYEE, DEPARTMENT 및 PROJECT 테이블은 HUMANRES 테이블 공간 내에 있는데, 이는 컨테이너 0, 1, 2, 3 및 4에 걸쳐 있습니다. 이 예에서는 별도의 디스크에 존재하는 각 컨테이너를 보여줍니다.

테이블의 데이터는 라운드 로빈 유형으로 테이블 공간 내의 모든 컨테이너에 저장됩니다. 이것은 주어진 테이블 공간에 속하는 컨테이너간에 데이터의 균형을 이룹니다. 다른 컨테이너를 사용하기 전에 데이터베이스 관리 프로그램이 하나의 컨테이너에 기록하는 페이지의 수를 *extent 크기*라고 합니다.

버퍼 풀

버퍼 풀은 디스크에서 읽거나 수정 중인 테이블 및 색인 데이터 페이지를 캐쉬하도록 할당되는 기본 메모리의 양입니다. 버퍼 풀의 목적은 시스템의 성능을 향상시키는 것입니다. 데이터는 디스크보다는 메모리에서 더 빨리 액세스할 수 있으므로, 데이터베이스 관리 프로그램이 디스크에서 읽거나 쓰는(입출력) 횟수가 적을수록 성능은 향상됩니다. 대부분의 경우 오직 하나의 버퍼 풀이 필요하지만, 둘 이상의 버퍼 풀을 작성할 수 있습니다.

버퍼 풀의 구성은 가장 중요한 단일 조정 영역으로, 이는 느린 입출력으로 인한 지연을 줄일 수 있기 때문입니다.

21 페이지의 그림8은 버퍼 풀과 컨테이너간의 관계를 설명합니다.

데이터베이스 오브젝트/개념	대등한 물리적 오브젝트

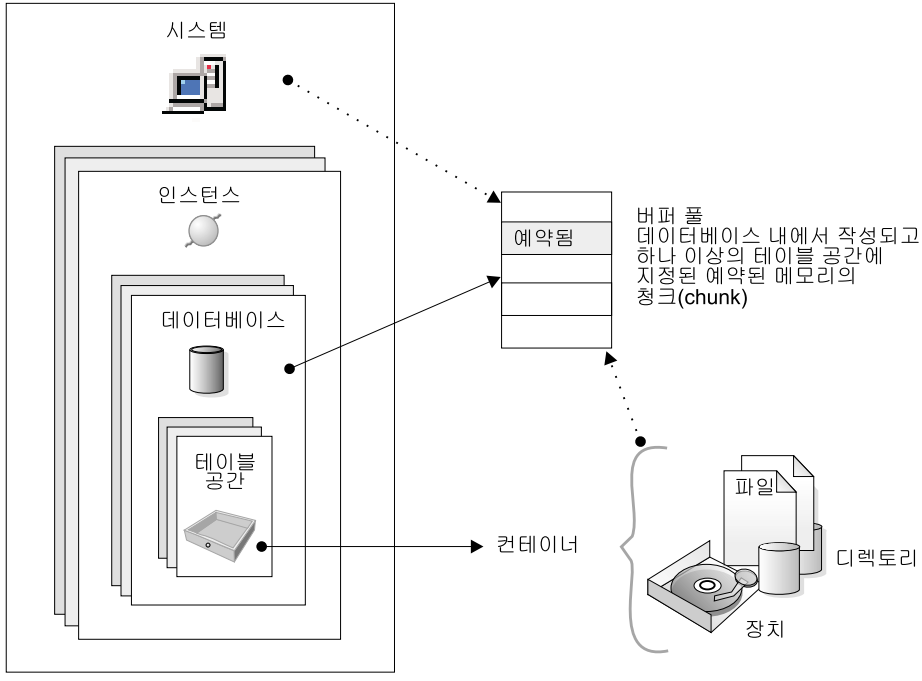

그림 8. 버퍼 풀 및 컨테이너

시스템 오브젝트 개요

DB2 인스턴스 또는 데이터베이스가 작성되면, 해당 구성 파일은 기본 매개변수 값으로 작성됩니다. 성능을 개선하기 위해 이 매개변수 값을 수정할 수 있습니다.

구성 매개변수

*구성 파일*에는 DB2 제품과 개별적 데이터베이스에 할당된 자원과 진단 레벨 같은 값을 정의하는 매개변수가 들어 있습니다. 각 DB2 인스턴스의 데이터베이스 관리 프로그램 구성 파일과 개별적 데이터베이스에 대한 데이터베이스 구성 파일의 두가지 유형의 구성 파일이 있습니다(23 페이지의 그림9 참조).

*데이터베이스 관리 프로그램 구성 파일*은 DB2 인스턴스가 작성될 때 작성됩니다. 인스턴스의 일부인 임의의 한 데이터베이스와 무관하게, 들어 있는 매개변수는 인스턴스 레벨에서 시스템 자원에 영향을 미칩니다. 여러 매개변수들의 값은 시스템 기본값에서 변경되어 사용자 시스템 구성에 따라 성능을 개선시키거나 용량을 증가시킬 수 있습니다.

마찬가지로 각 클라이언트 설치에 대해 하나의 데이터베이스 관리 프로그램 구성 파일이 있습니다. 이 파일에는 특정 워크스테이션의 클라이언트 인에이블러에 대한 정보가 들어 있습니다. 서버용으로 사용가능한 매개변수의 부속 집합은 클라이언트에 적용할 수 있습니다.

*데이터베이스 구성 파일*은 데이터베이스가 작성될 때 작성되며 데이터베이스가 상주하는 곳에 상주합니다. 데이터베이스당 하나의 구성 파일이 있습니다. 관련 매개변수는 다른 것 가운데, 해당 데이터베이스에 할당되는 자원량을 지정합니다. 많은 매개변수의 값은 성능을 개선시키거나 용량을 증가시키기 위해 변경될 수 있습니다. 특정 데이터베이스에서 활동 유형에 따라, 여러 가지 변경사항이 필요할 수 있습니다.

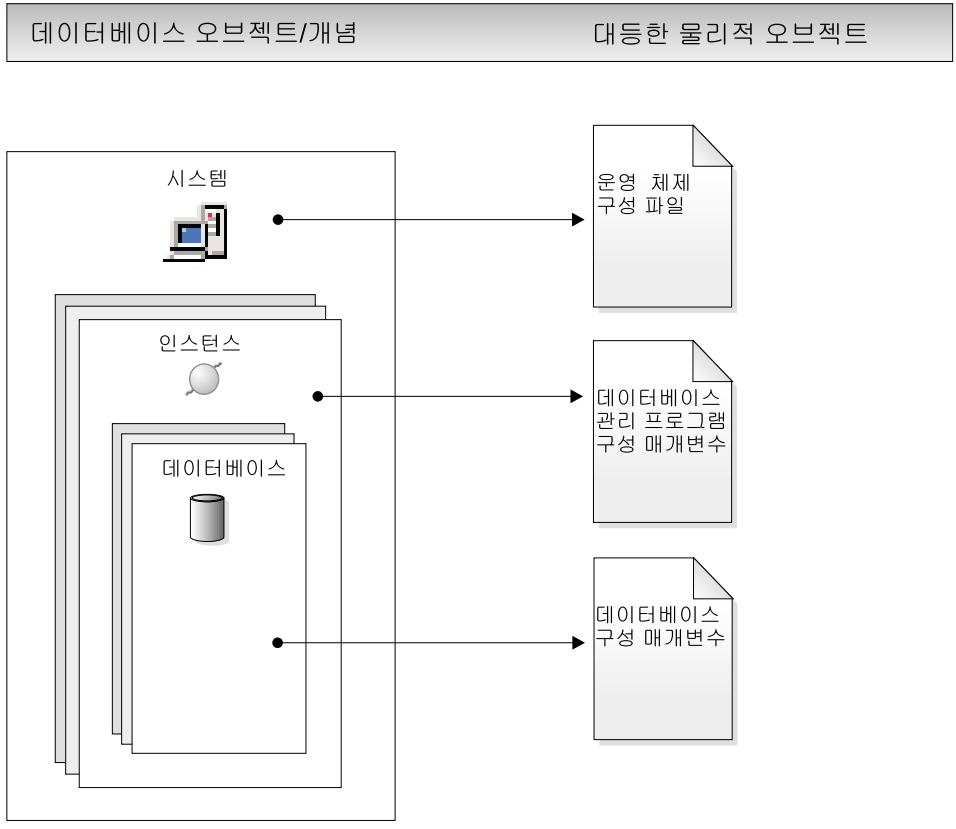

그림 9. 구성 매개변수 파일

데이터에 대한 비지니스 규칙

임의의 비지니스 내에서, 데이터는 특정 제한사항이나 규칙을 고수해야 합니다. 예를 들어, 사원 번호는 고유해야 합니다. DB2는 그러한 규칙을 실시하는 방법으로서 *제한조건*을 제공합니다.

DB2는 다음 컨테이너 유형을 제공합니다.

- NOT NULL 제한조건
- 고유 제한조건
- 기본 키 제한조건

- 외부 키 제한조건
- 점검 제한조건

NOT NULL 제한조건

NOT NULL 제한조건은 널(NULL) 값이 컬럼으로 입력되지 못하게 합니다.

고유 제한조건

고유 제한조건은 컬럼 세트 내의 값이 고유하고 테이블 내의 모든 행에 대해 널(NULL)이 아니도록 보장합니다. 예를 들어, DEPARTMENT 테이블 내의 일반적인 고유 제한조건은 부서 번호가 고유하고 널(NULL)이 아니어야 합니다.

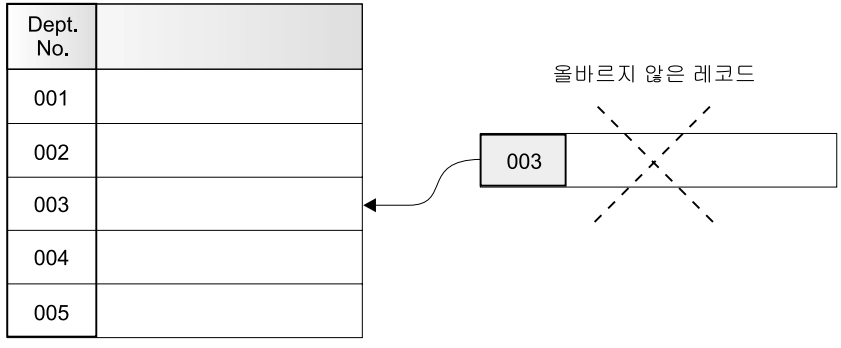

그림 10. 고유 제한조건의 중복 데이터 방지

데이터베이스 관리 프로그램은 데이터 무결성을 보장하며, 삽입 및 갱신 조작 중 제한조건을 수행합니다.

기본 키 제한조건

각 테이블은 하나의 기본 키를 가질 수 있습니다. 기본 키는 고유 제한조건으로서 동일한 등록 정보를 갖는 컬럼이나 컬럼의 조합입니다. 기본 키 및 외부 키 제한조건을 사용하여 테이블간의 관계를 정의할 수 있습니다.

기본 키는 테이블에서 행을 식별하기 위해 사용되므로 고유해야 하며, 극히 적은 추가 또는 삭제를 해야 합니다. 테이블은 둘 이상의 기본 키를 가질 수 없지만, 여러 고유 키를 가질 수 있습니다. 기본 키는 선택적이며, 테이블이 작성되거나 변경될 때 정의될 수 있습니다. 자료를 내보내기하거나 재구성할 때 데이터를 순서화하므로 기본 키가 유용합니다.

다음 테이블에서, DEPTNO 및 EMPNO는 DEPARTMENT 및 EMPLOYEE 테이블의 기본 키입니다.

표 1. DEPARTMENT 테이블

DEPTNO(기본 키)	**DEPTNAME**	**MGRNO**
A00	Spiffy Computer Service Division	000010
B01	Planning	000020
C01	Information Center	000030
D11	Manufacturing Systems	000060

표 2. EMPLOYEE 테이블

EMPNO (기본 키)	**FIRSTNAME**	**LASTNAME**	**WORKDEPT (외부 키)**	**PHONENO**
000010	Christine	Haas	A00	3978
000030	Sally	Kwan	C01	4738
000060	Irving	Stern	D11	6423
000120	Sean	O'Connell	A00	2167
000140	Heather	Nicholls	C01	1793
000170	Masatoshi	Yoshimura	D11	2890

외부 키 제한조건

외부 키 제한조건(참조 무결성 제한조건)은 사용자가 테이블 간과 테이블 내에서 필요한 관계를 정의할 수 있게 합니다.

예를 들어, 일반적인 외부 키 제한조건은 EMPLOYEE 테이블의 모든 사원이 DEPARTMENT 테이블에 정의된 대로 기존 부서의 구성원이어야 함을 나타냅니다.

이 관계를 설정하기 위해, 외부 키로서 EMPLOYEE 테이블에 부서 번호를 정의하고, 기본 키로서 DEPARTMENT 테이블에 있는 부서 번호를 정의합니다.

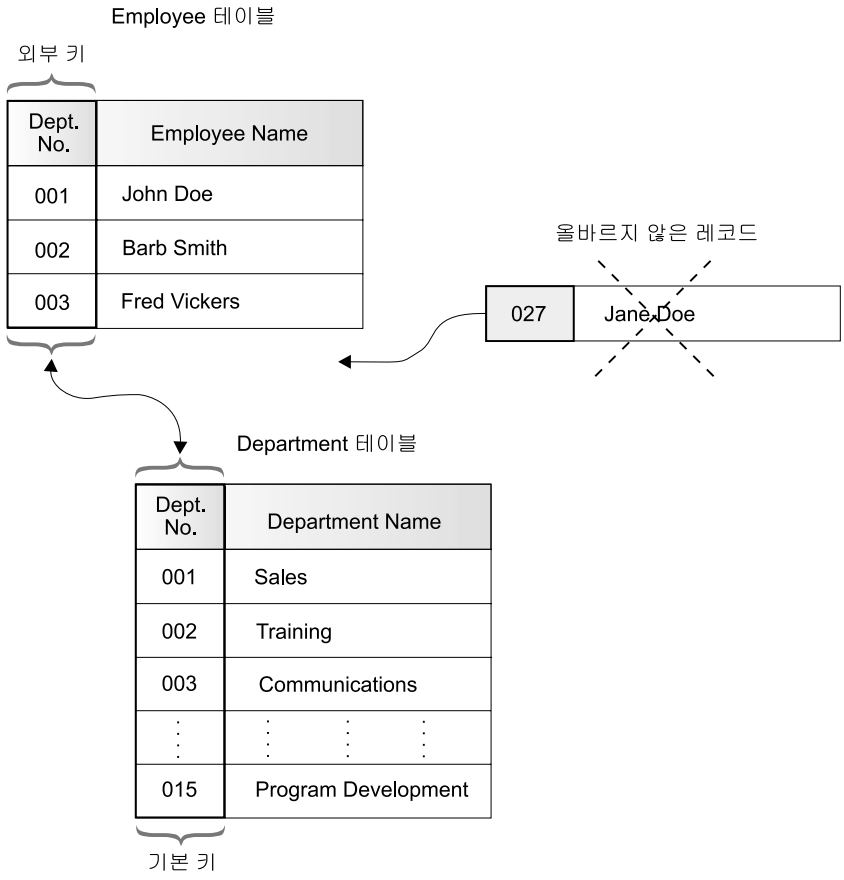

그림 11. 외부 및 기본 키 제한조건은 관계를 정의하며 데이터를 보호합니다.

점검 제한조건

점검 제한조건은 테이블의 모든 행의 하나 이상의 컬럼에 허용된 값을 지정하는 데이터베이스 규칙입니다.

예를 들어, EMPLOYEE 테이블에서, ″Sales″, ″Manager″ 또는 ″Clerk″이 되는 작업 컬럼의 유형을 정의할 수 있습니다. 이 제한조건 사용 시, 작업 컬럼의 유형에서 다른 값을 가진 임의의 레코드는 유효하지 않으므로 거부되며, 테이블에서 허용되는 데이터 유형에 대한 규칙을 실시합니다.

또한 데이터베이스에서 *트리거*를 사용할 수 있습니다. 트리거는 더 복잡하며 잠재적으로 제한조건보다 더 강력합니다. 지정된 기본 테이블에서 INSERT, UPDATE 또는 DELETE 절과 함께 실행되는 조치 또는 이러한 절에 의해 트리거되는 일련의 조치를 정의합니다. 사

용자는 무결성 또는 비지니스 규칙의 일반 양식을 지원하기 위해 트리거를 사용할 수 있습니다. 예를 들어, 트리거는 주문을 받아들이기 전에 고객의 신용 한도를 점검하거나 계정으로부터의 철회가 고객의 표준 철회 패턴과 맞지 않는 경우 경고를 보여주도록 은행 응용프로그램에서 사용될 수 있습니다. 트리거에 대해서는 *응용프로그램 개발 안내서*에서 자세한 내용을 참조하십시오.

데이터베이스 복구

데이터베이스가 하드웨어 또는 소프트웨어 오류(또는 둘다)로 인해 사용할 수 없게 되고 여러 장애 시나리오는 여러 가지 복구 조치를 필요로 하는 경우가 있습니다. 실패의 가능성에 대비하여 데이터베이스를 보호하도록 연습된 전략을 가져야 합니다.

*데이터 복구 및 고가용성 안내 및 참조서*에 백업 및 복구에 대한 상세한 정보와 탁월한 백업 및 복구 전략을 개발하는 방법이 설명되어 있습니다.

데이터베이스 백업의 개념은 다른 데이터 백업과 동일하며, 원본의 실패 또는 손상시 여러 매체에 데이터를 저장하고 복사합니다. 백업의 가장 간단한 경우는 더 이상 트랜잭션이 발생하지 않도록 확인하기 위해 데이터베이스를 종료한 후 단순히 백업하는 것과 관련됩니다.

데이터베이스가 손상된 경우, 백업 이미지에서 다시 빌드할 수 있습니다. 데이터베이스의 재구축을 복구라고 합니다.

트랜잭션이 진행되는 동안 데이터베이스가 손상된 경우, 데이터베이스가 재시작될 때 응급 복구라는 프로세스가 발생합니다. *응급 복구*는 데이터베이스를 일관되게 사용할 수 있는 상태로 다시 돌리는 프로세스입니다. 완료되지 않은 트랜잭션을 구간 복원하고, 파손이 발생했을 때 메모리에 남아 있던 확약된 트랜잭션을 완료하여 이루어집니다.

데이터베이스가 손상된 경우 두 가지 방법(버전 복구와 롤 포워드 복구)을 사용하여 복구할 수 있습니다.

*버전 복구*는 백업 조작 중 작성된 이미지를 사용하는 데이터베이스의 이전 버전의 복원입니다. 데이터베이스 백업을 사용하여 백업 작성시와 동일한 상태로 데이터베이스를 복원할 수 있습니다. 그러나 백업시부터 장애시까지의 모든 작업 단위가 유실됩니다.

백업이 이루어진 시점으로 데이터베이스를 복원하려면 롤 *포워드 복구*를 사용해야 합니다. 롤 포워드 복구 방법을 사용하려면, 데이터베이스를 백업하고, 로그를 아카이브해야 합니다(*logretain* 또는 *userexit* 데이터베이스 구성 매개변수 중 하나 또는 둘다를 작동시켜서 수행).

모든 데이터베이스에는 응용프로그램 또는 시스템 오류를 복구하는 데 사용되는 복구 로그를 포함합니다. 복구 로그는 데이터베이스 백업과 함께 오류가 발생했을 때의 시점으로 데이터베이스를 복구하는 데 사용할 수 있습니다. 로그 파일은 데이터베이스가 작성될 때 자동으로 작성됩니다. 로그 파일을 직접 수정할 수 없습니다.

또다른 중요한 복구 오브젝트는 복구 실행기록 파일입니다. 복구 실행기록 파일에는 데이터베이스의 전부 또는 일부가 주어진 시점으로 복구되어야 하는 경우에 사용될 수 있는 백업 정보의 요약이 들어 있습니다. 백업, 복원 및 로드 조작과 같은 복구 관련 이벤트를 추적하는 데 사용됩니다.

주: 백업 및 복구에 대한 모든 정보, Command Reference 및 API Reference의 해당 정보가 *데이터 복구 및 고가용성 안내 및 참조서*에 통합되어 있습니다. *데이터 복구 및 고가용성 안내 및 참조서*는 백업 및 복구 정보를 얻을 수 있는 1차적인 유일한 소스입니다.

데이터베이스에서 테이블 재구성

테이블은 많은 갱신 이후에 단편화되어 성능이 저하될 수 있습니다. 통계를 수집하고 가시적 성능 개선을 얻지 못한 경우, 테이블 데이터의 재구성이 도움이 될 수 있습니다. 테이블 데이터를 재구성할 때, 지정된 색인에 따라 실제 순서로 데이터를 재배열하며, 단편화된 데이터에서 계승된 여유 공간을 제거합니다. 이것은 데이터에 대한 더 빠른 액세스를 제공하므로, 성능을 개선시킬 수 있습니다.

테이블을 재구성하기 전에, REORGCHK 명령을 호출하고 테이블에서 통계를 수집하도록 권장합니다. 이 명령을 수행하면 테이블 데이터의 재구성이 적절한지의 여부를 판별하도록 돕습니다. REORGCHK 명령에 대해서는 *Command Reference*에서 자세한 내용을 참조하십시오.

DB2 보안 개요

데이터베이스 서버 및 연관된 데이터와 자원을 보호하기 위해, DB2는 외부 보안 서비스와 내부 액세스 제어 정보를 같이 사용합니다. 데이터베이스 서버에 액세스하려면, 데이터베이스 데이터 또는 자원으로 액세스하기 전에 여러 보안 검사를 통과해야 합니다. 데이터베이스 보안의 첫 단계는 *인증*이라고 하며, 사용자는 본인이 맞는지 증명해야 합니다. 두 번째 단계는 *권한 부여*라고 하며, 이 단계에서 데이터베이스 관리 프로그램은 유효한 사용자가 요청된 조치를 실행하거나 요청된 데이터에 액세스할 수 있는지 여부를 결정합니다.

인증

사용자의 인증은 DB2 밖의 보안 기능을 사용하여 완료됩니다. 보안 기능은 운영 체제와 다른 제품의 일부일 수 있으며, 어떤 경우에서는 없을 수도 있습니다. UNIX 기반 시스템에서, 보안 기능은 운영 체제 자체에 있습니다. DCE 보안 서비스는 분산 환경에 보안 기능을 제공하는 다른 제품입니다. Windows 95 또는 Windows 3.1 운영 체제에는 보안 기능이 없습니다.

보안 기능은 사용자를 인증하기 위해 사용자 ID 및 암호를 요구합니다. 사용자 ID는 보안 기능에 대해 사용자를 식별합니다. 올바른 암호를 제공하여(사용자 및 보안 기능에만 알려진 정보) 사용자 식별(사용자 ID에 해당)이 검증됩니다.

사용자를 인증한 후에는 다음을 수행하십시오.

- 사용자는 SQL 권한 부여 이름 또는 *authid*를 사용하여 DB2에서 식별해야 합니다. 이 이름은 사용자 ID 또는 맵된 값과 동일할 수 있습니다. 예를 들어 UNIX 기반 시스템에서, DB2 *authid*는 DB2 이름 지정 규칙을 따르는 UNIX 사용자 ID를 대문자로 전환하면 파생됩니다. DCE 보안 서비스 제품에서, DB2 *authid*는 DCE 레지스트리에 포함되어 있고 인증이 성공적으로 완료되는 곳으로부터 얻어집니다.

- 사용자가 속한 그룹 목록이 얻어집니다. 그룹 멤버십은 사용자에게 권한을 부여할 때 사용할 수 있습니다. 그룹은 DB2 권한 부여 이름에 맵을 해야 하는 보안 기능 엔터티입니다. 이 맵핑은 사용자 ID에 사용된 방식과 비슷한 방식으로 실행됩니다.

  DB2는 최대 64 그룹의 그룹 목록을 얻을 수 있습니다. 사용자가 65 그룹 이상의 구성원인 경우, 유효한 DB2 권한 부여 이름에 맵하는 첫 번째 64만 DB2 그룹 목록에 추가됩니다. 오류는 리턴되지 않으며, 첫 번째 64 다음의 그룹은 모두 DB2에 의해 무시됩니다.

DB2는 보안 기능을 사용하여 사용자를 다음 두 가지 방법 중 한 방법으로 인증합니다.

- DB2는 식별의 증거로서 성공적인 보안 시스템 로그인을 사용하고 다음을 허용합니다.
  - 지역 데이터에 액세스하는 지역 명령 사용
  - 서버가 클라이언트 인증을 신뢰하는 원격 연결 사용
- DB2는 사용자 ID와 암호를 같이 승인합니다. DB2는 보안 기능을 사용하여 사용자 ID 증거로 이 결합의 성공적인 유효성을 사용하며 다음을 허용합니다.
  - 서버가 인증을 증명해야 하는 원격 연결 사용
  - 로그인에 사용된 ID를 제외한 ID 하에서 사용자가 명령을 수행하려는 조작 사용

DB2 관리자는 프로파일 등록 변수 DB2CHGPWD_EEE를 통해 AIX 및 Windows NT EEE 시스템에서 다른 사용자가 암호를 변경할 수 있게 합니다. 이 변수의 기본값은 NOT SET(사용 안함)입니다. DB2CHGPWD_EEE는 다른 DB2 프로파일 변수가 사용하는 표준 부울 값을 승인합니다.

DB2 관리자는 모든 노드에 대한 암호가 Windows NT에서는 Windows NT 도메인 제어기를 사용하여, AIX에서는 NIS를 사용하여 중앙에서 유지보수되도록 해야 합니다.

주: 암호가 중앙에서 유지보수되지 않는 경우, DB2CHGPWD_EEE 변수를 사용하면 암호가 모든 노드에서 일치하지 않는 결과를 가져올 수 있습니다. 즉, "암호 변경" 기능을 사용하면, 사용자 암호는 연결되는 노드에서만 변경됩니다.

AIX의 DB2 UDB는 운영 체제에서 실패한 암호 시도를 기록하고 클라이언트가 LOGINRETRIES 매개변수에서 지정된 허용 가능한 로그인 시도 횟수를 초과하는 시기를 검출할 수 있습니다.

원격 클라이언트가 데이터베이스에 액세스하는 경우, 특히 관련된 시스템 항목 유효성 검사에 대해서는 *관리 안내서: 구현*의 "서버에 대한 인증 방식 선택"에서 자세한 내용을 참조하십시오.

권한 부여

권한 부여는 권한이 부여된 DB2 사용자에 대한 정보(사용자가 실행하는 데이터베이스 조작과 액세스한 데이터 오브젝트를 나타내는)를 DB2가 얻는 프로세스입니다. 각 사용자의 요청으로, 오브젝트 및 관련된 조작에 의존하는 둘 이상의 권한 부여 점검이 있습니다.

권한 부여는 DB2 기능을 사용하여 실행됩니다. DB2 테이블과 구성 파일을 사용하여 각 권한 부여 이름과 연관된 사용권한을 기록합니다. 인증된 사용자의 권한 부여 이름과 사용자가 속한 그룹 이름은 기록된 사용권한과 비교됩니다. 비교에 따라, DB2는 요청된 액세스의 허용 여부를 결정합니다.

DB2에 의해 기록된 특권과 권한 레벨의 두 가지 사용권한 유형이 있습니다. *특권*은 사용자가 데이터베이스 자원을 작성하거나 액세스할 수 있도록 하는 하나의 사용권한을 권한 부여 이름에 대해 정의합니다. 특권은 데이터베이스 카탈로그에 저장됩니다. *권한 레벨*은 특권을 그룹화하는 방법을 제공하고, 더 높은 레벨에서 데이터베이스 관리 프로그램 유지보수와 유틸리티 조작을 제어합니다. 데이터베이스 고유 권한은 데이터베이스 카탈로그에 저장됩니다. 시스템 권한은 그룹 멤버십과 연관되고, 주어진 인스턴스용 데이터베이스 관리 프로그램 구성 파일에 저장됩니다.

그룹은 각 사용자에 대해 개별적으로 권한 부여 또는 권한 취소 없이 사용자 콜렉션에 대한 권한 부여를 실행하는 데 있어서 편리한 방법을 제공합니다. 다르게 지정되지 않으면, 그룹 권한 부여 이름은 어디서도 권한 부여 이름으로도 사용될 수 있고 권한 부여 목적에 사용됩니다. 일반적으로, 그룹 멤버십은 동적 SQL과 비-데이터베이스 오브젝트 권한 부여(예: 인스턴스 레벨 명령 및 유틸리티)용으로 고려되지만, 정적 SQL용으로 고려되지 않습니다. 이 일반적인 경우에 대한 예외는

특권이 PUBLIC으로 부여될 때에 발생합니다. 이것은 정적 SQL이 처리될 때 고려됩니다. 그룹 멤버십이 적용되지 않는 특정한 경우는 적용 가능한 DB2 문서에 설명되어 있습니다.

*관리 안내서: 구현*의 "특권 권한 부여"에서 자세한 내용을 참조하십시오.

연합 데이터베이스 인증 및 권한 부여 개요

DB2 연합 데이터베이스 시스템은 여러 데이터베이스 관리 시스템에 있는 정보에 액세스할 수 있으므로, 추가 과정은 데이터를 안전하게 하는 데 필요할 수 있습니다.

인증 접근 방식을 계획할 때, 사용자가 DB2에서 뿐만 아니라, 데이터 소스에서도 인증 검사를 통과해야 한다는 사실을 기억하십시오. 연합 시스템에서 인증은 DB2 클라이언트 워크스테이션, DB2 서버, 데이터 소스(DB2, OS/390용 DB2, 기타 DRDA 서버, Oracle) 또는 DB2(클라이언트 또는 DB2 서버)와 데이터 소스의 결합에서 발생할 수 있습니다. DCE 환경에서도 데이터 소스에 사용자 ID와 암호가 필요하면 특정 과정이 필요할 수 있습니다. *관리 안내서: 구현*의 "연합 데이터베이스 인증 처리"에서 자세한 내용을 참조하십시오.

이와 같이, 사용자는 데이터 소스와 DB2에서 권한 부여 검사를 통과해야 합니다. 각 데이터 소스(DB2, Oracle, OS/390용 DB2 등)는 제어하에 오브젝트의 보안을 유지보수합니다. 사용자가 별명에 대해 조작을 수행하려면, 이 사용자는 별명이 참조하는 뷰 또는 테이블에 대한 권한 부여 검사를 통과해야 합니다.

제3장 연합 시스템

연합 데이터베이스 시스템 또는 *연합 시스템*은 단일 명령문에서 두 개 이상의 DBMS 또는 데이터베이스를 참조하는 사용자 SQL문과 응용프로그램을 지원하는 데이터베이스 관리 시스템(DBMS)입니다. 한 예로 서로 다른 두 개의 DB2 데이터베이스에서 테이블간의 조인을 들 수 있습니다. 이 유형의 명령문을 *분산 요청*이라고 합니다.

DB2 Universal Database 연합 시스템은 데이터베이스 및 DBMS에서 분산 요청에 대한 지원을 제공합니다. 예를 들어, DB2 테이블과 Oracle 뷰간에 UNION 조작을 수행할 수 있습니다. 지원되는 DBMS에는 DB2, DB2 계열의 구성원(예: OS/390용 DB2와 AS/400용 DB2) 및 Oracle이 있습니다.

DB2 연합 시스템은 데이터베이스 오브젝트에 대해 *위치 투명성*을 제공합니다. 정보(테이블 및 뷰)가 이동되면, 해당 정보에 대한 참조(*별명*이라고 함)는 정보를 요청한 응용프로그램을 변경하지 않고 갱신될 수 있습니다. DB2 연합 시스템은 또한 모든 DB2 SQL 통용어 또는 특정 최적화 기능을 지원하지 않는 DBMS에 대해 *보상*을 제공합니다. 그러한 DBMS에서 수행될 수 없는 조작은(순환 SQL과 같은) DB2에서 수행됩니다.

반자동 방식의 DB2 연합 시스템 기능: Oracle 오브젝트에 대한 참조가 있는 DB2 조회는 Oracle 응용프로그램이 같은 서버에 액세스하는 중에 제출될 수 있습니다. DB2 연합 시스템은 Oracle 또는 기타 DBMS 오브젝트에 대한 액세스를 독점하거나 제한하지 않습니다(무결성 및 잠금 제한조건을 넘음).

DB2 연합 시스템은 DB2 UDB 인스턴스, *연합 데이터베이스*로 작동하는 데이터베이스 및 하나 이상의 *데이터 소스*로 구성됩니다. 연합 데이터베이스에는 데이터 소스 및 특성을 식별하는 카탈로그 항목이 들어 있습니다. 데이터 소스는 DBMS 및 데이터로 구성됩니다. 응용프로그램은 기타 모든 DB2 데이터베이스처럼 연합 데이터베이스에 연결됩니다. 34 페이지의 그림12에서는 연합 데이터베이스 환경의 시각적 표현을 보여줍니다.

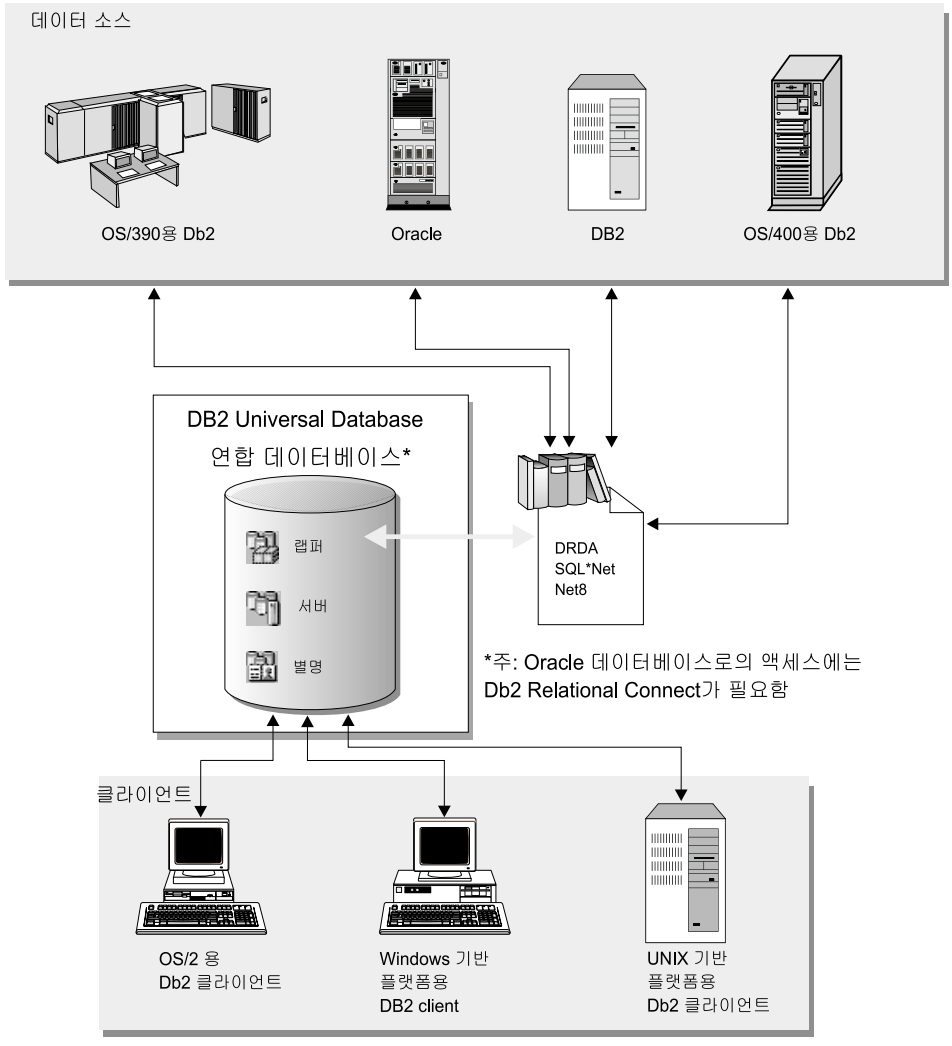

그림 12. 연합 데이터베이스 시스템

DB2 연합 데이터베이스 카탈로그 항목에는 오브젝트에 들어 있는 정보 및 사용되는 조건 등의 데이터 소스 오브젝트 정보가 들어 있습니다. 이 DB2 카탈로그는 오브젝트에 대한 정보를 많은 DBMS에 저장하므로, *전역 카탈로그*라고 합니다. 오브젝트 속성은 카탈로그에 저장됩니다. 참조되는 실제 DBMS, 데이터 소스와 통신할 때 사용되는 모듈 그리고 액세스되는 DBMS 데이터 오브젝트(예: 테이블)는 데이터베이스 외부에 있습니다. (한 가지

예외: 연합 데이터베이스는 연합 시스템에 대한 데이터 소스일 수 있습니다.) 제어 센터 또는 SQL DDL문을 사용하여 연합 오브젝트를 작성할 수 있습니다. 필수 연합 데이터베이스 오브젝트는 다음과 같습니다.

랩퍼 특정 클래스나 데이터 소스의 범주를 액세스하는 데 사용되는 모듈(dll, 라이브러리 등)을 식별합니다.

서버 데이터 소스를 정의합니다. 서버 데이터에는 랩퍼 이름, 서버 이름, 서버 유형, 서버 버전, 권한 부여 정보 및 서버 옵션이 있습니다.

별명 식별자는 특정 데이터 소스 오브젝트(테이블, 별명, 뷰)를 참조하는 연합 데이터베이스에 저장됩니다. 응용프로그램은 테이블 및 뷰를 참조하는 것처럼 조회에 있는 별명을 참조합니다.

사용자의 특정 필요에 따라 추가 오브젝트를 작성할 수 있습니다.

- 인증 문제를 처리하기 위한 사용자 맵핑
- 데이터 소스 유형 및 DB2 유형간의 관계를 사용자 정의하기 위한 데이터 유형 맵핑
- 지역 기능을 데이터 소스 기능으로 맵하기 위한 기능 맵핑
- 성능을 향상시키기 위한 색인 스펙

연합 시스템이 설정된 후 데이터 소스에 있는 정보는 하나의 대형 데이터베이스에 있는 것처럼 액세스될 수 있습니다. 사용자 및 응용프로그램은 하나의 연합 데이터베이스로 조회를 전송한 후 필요에 따라 DB2 계열 및 Oracle 시스템에서 데이터를 검색합니다. 사용자 및 응용프로그램은 조회에서 별명을 지정합니다. 이들 별명은 데이터 소스에 위치한 테이블 및 뷰에 대한 참조를 제공합니다. 일반 사용자 입장에서 별명(nickname)과 별명(alias)은 비슷합니다.

연합 시스템 성능에 영향을 주는 인수가 많습니다. 가장 중요한 인수는 데이터 소스와 해당 오브젝트에 대한 정확한 최신의 정보가 연합 데이터베이스 전역 카탈로그에 저장되어 있는지 확인하는 것입니다. 이 정보는 DB2 최적화에서 사용되고 의사 결정에 영향을 주어 데이터 소스에서 조작에 대한 평가가 이루어지도록 합니다. 연합 시스템 성능에 대해서는 *관리 안내서: 성능*에서 자세한 내용을 참조하십시오.

DB2 연합 시스템은 몇 가지 제한사항 하에서 작동합니다. 분산 요청은 읽기 전용 조작으로 제한됩니다. 이 외에, 별명에 대하여 유틸리티 조작(LOAD, REORG, REORGCHK, IMPORT, RUNSTATS 등)을 실행할 수 없습니다.

그러나, 사용자는 통과 기능을 사용하여 해당 데이터 소스에 관련된 SQL 통용어를 통해 DDL 및 DML문을 직접 데이터베이스 관리 프로그램에 제출할 수 있습니다.

연합 시스템은 병렬 처리 환경에서 작동합니다. 성능 이점은 연합 데이터베이스 조회가 의미에 있어서 지역 오브젝트(테이블, 뷰) 참조와 별명 참조로 나누어질 수 있는 extent에 의해 제한됩니다. 별명 데이터에 대한 요청은 순차적으로 처리됩니다. 지역 오브젝트는 병렬 처리될 수 있습니다. 예를 들어, Oracle 데이터 소스에서 A와 B가 지역 테이블이고 C와 D가 별명 참조 테이블인 `SELECT * FROM A, B, C, D` 조회가 주어지는 경우, 테이블 A와 B를 병렬 조인과 함께 조인할 수 있습니다. 결과는 별명 C 및 D와 순차적으로 조인됩니다.

연합 시스템 작동

DB2 Enterprise Edition(EE) 및 DB2 Enterprise - Extended Edition(EEE)은 연합 시스템을 지원할 수 있습니다. 연합 시스템을 작동시키려면, 다음을 수행하십시오.

1. 설치중에 DB2 EE 또는 EEE의 *DB2 데이터베이스에 대한 분산 조인* 설치 옵션을 선택하십시오.
2. 연합 시스템에서 Oracle 데이터베이스를 포함하는 경우, DB2 Relational Connect를 설치하십시오. *설치 및 구성 보충 설명서*에서 자세한 내용을 참조하십시오.
3. 데이터베이스 관리 프로그램 구성 매개변수를 *연합*하도록 "예"로 설정하십시오.
4. 랩퍼, 서버 및 별명(*관리 안내서: 구현*의 "데이터베이스 작성" 참조)을 작성하십시오.
5. 필요에 따라 추가 오브젝트를 작성하거나 옵션을 설정하십시오. (*관리 안내서: 구현*의 "설계 구현"에서 자세한 내용을 참조하십시오.)

제4장 병렬 데이터베이스 시스템

DB2는 데이터베이스 관리 프로그램을 병렬, 다중 노드 환경으로 확장시킵니다. *데이터베이스 파티션*은 고유의 데이터, 색인, 구성 파일 및 트랜잭션 로그로 이루어진 데이터베이스의 부분입니다. 데이터베이스 파티션은 종종 노드 또는 데이터베이스 노드라고도 합니다. (노드는 AIX용 DB2 Parallel Edition 버전 1 제품에서 사용된 용어입니다.)

*단일 파티션 데이터베이스*는 하나의 데이터베이스 파티션만이 있는 데이터베이스입니다. 데이터베이스의 모든 데이터가 해당 파티션에 저장됩니다. 이 경우 노드 그룹(11 페이지의 『노드 그룹』 참조)이 있으면, 다른 추가 기능을 제공하지 않습니다.

*파티션된 데이터베이스*는 두 개 이상의 데이터베이스 파티션을 가진 데이터베이스입니다. 테이블은 하나 이상의 데이터베이스 파티션에 위치할 수 있습니다. 테이블이 다중 파티션을 이루는 노드 그룹에 있으면, 하나의 파티션에 여러 행이 저장되고 나머지는 다른 파티션에 저장됩니다.

흔히, 단일 데이터베이스 파티션은 각 물리 노드에 있으며, 각 데이터베이스 파티션에서 데이터베이스 관리 프로그램은 시스템상의 프로세서를 사용하여 데이터베이스에서 전체 데이터의 해당 부분을 관리합니다.

데이터는 데이터베이스 파티션 전체에서 나누어져 있기 때문에, 다중 물리적 노드의 다중 프로세서를 사용하여 정보에 대한 요청에 부합할 수 있습니다. 데이터 검색 및 갱신 요청은 자동으로 하위 요청으로 파티션되며, 적용 가능한 데이터베이스 파티션간에 병렬 상태로 실행됩니다. 데이터베이스가 데이터베이스 파티션 전반에서 나뉜다는 사실을 SQL문을 발행하는 사용자는 명확히 알 수 있습니다.

해당 사용자의 *조정자* 노드로 알려진 데이터베이스 파티션을 통해 사용자 상호작용이 발생합니다. 조정자는 응용프로그램으로서 같은 데이터베이스 파티션에서 수

행되고, 원격 응용프로그램의 경우에는 해당 응용프로그램이 연결된 데이터베이스 파티션에서 수행됩니다. 모든 데이터베이스 파티션은 조정자 노드로서 사용될 수 있습니다.

노드 그룹 및 데이터 파티션

데이터베이스에서 하나 이상의 데이터베이스 파티션의 이름 지정된 부속 집합을 정의할 수 있습니다. 정의된 각 부속 집합을 노드 그룹이라고 합니다. 두 개 이상의 데이터베이스 파티션이 들어 있는 각 부속 집합은 *다중 파티션* 노드 그룹이라고 합니다. 다중 파티션 노드 그룹은 같은 인스턴스에 속한 데이터베이스 파티션으로만 정의될 수 있습니다.

39 페이지의 그림13은 5가지 파티션이 있는 데이터베이스의 예를 보여줍니다.

- 노드 그룹은 데이터베이스 파티션 중 하나를 제외한 모두를 포함합니다(노드 그룹 1).
- 노드 그룹에는 하나의 데이터베이스 파티션이 들어 있습니다(노드 그룹 2).
- 노드 그룹에는 두 개의 데이터베이스 파티션이 들어 있습니다(노드 그룹 3).
- 노드 그룹 2에 있는 데이터베이스 파티션은 노드 그룹 1에서 공유(및 겹침)됩니다.
- 노드 그룹 3에는 노드 그룹 1과 공유(및 겹침)되는 단일 데이터베이스 파티션이 있습니다.

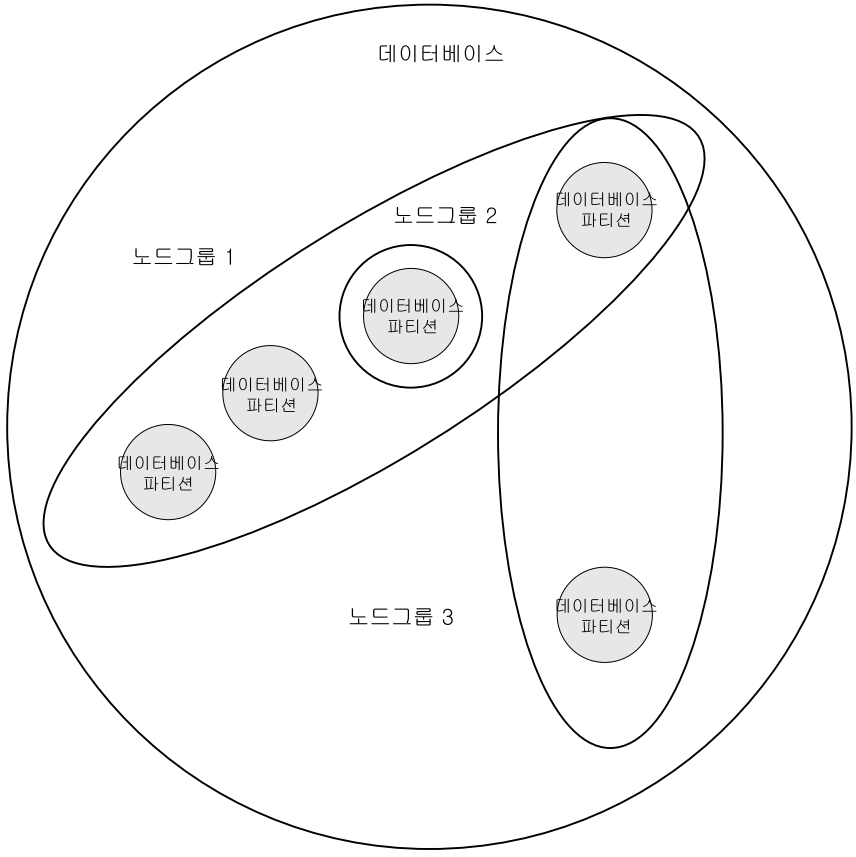

그림 13. 데이터베이스의 노드 그룹

CREATE NODEGROUP문을 사용하여 새 노드 그룹을 작성하십시오. *SQL 참조서*에서 자세한 내용을 참조하십시오. 데이터는 노드 그룹의 파티션 전체에 나누어집니다. 다중 파티션 노드 그룹을 사용하려면, 몇 가지 노드 그룹 설계 고려사항을 알아보아야 합니다. 111 페이지의 『노드 그룹 설계』에서 자세한 내용을 참조하십시오.

병렬 처리의 유형

데이터베이스 조회와 같은 타스크의 구성요소는 극적으로 성능을 향상시키기 위해 병렬로 실행될 수 있습니다. 타스크 속성, 데이터베이스 구성 및 하드웨어 환경은 모두 DB2가 병렬로 타스크를 수행하는 방식을 결정합니다. 이러한 고려사항은 서로 관련되며, 데이터베이스의 실제 및 논리 설계에서 작업할 때 함께 고려되어야 합니다. 이 절에서는 DB2가 지원하는 다음과 같은 병렬 처리 유형을 설명합니다.

- 입출력
- 조회
- 유틸리티

입출력 병렬 처리

하나의 테이블 공간에 대하여 복수의 컨테이너가 존재할 때, 데이타베이스 관리 프로그램은 *병렬 I/O*를 활용할 수 있습니다. 병렬 I/O는 동시에 둘 이상의 입출력 장치에 쓰거나, 읽는 프로세스를 나타내며, 단위 처리량을 상당히 개선시킬 수 있습니다.

입출력 병렬 처리는 44 페이지의 『하드웨어 환경』에서 설명된 각 하드웨어 환경의 구성요소입니다. 53 페이지의 표3에는 입출력 병렬 처리에서 가장 적합한 하드웨어 환경이 나열되어 있습니다.

조회 병렬 처리

조회 병렬 처리에는 조회간 병렬 처리와 조회 내 병렬 처리의 두 가지 유형이 있습니다.

*조회간 병렬 처리*는 다중 응용프로그램이 동시에 데이터베이스를 조회하는 기능입니다. 각 조회는 다른 조회와 상관없이 실행되지만, DB2는 모든 조회를 동시에 실행합니다. DB2는 이 병렬 처리 유형을 항상 지원합니다.

*조회 내 병렬 처리*는 *파티션 내 병렬 처리* 및 *파티션간 병렬 처리* 중 하나 또는 둘다를 사용하여 동시에 하나의 조회 부분을 처리하는 것입니다.

조회 병렬 처리 용어는 이 책 전체에서 사용됩니다.

파티션 내 병렬 처리

*파티션 내 병렬 처리*는 하나의 조회를 여러 부분으로 분해하는 기능입니다. (일부 유틸리티도 이 병렬 처리 유형을 실행합니다. 43 페이지의 『유틸리티 병렬 처리』에서 자세한 내용을 참조하십시오.)

파티션 내 병렬 처리는 색인 작성, 데이터베이스 로드 또는 SQL 조회와 같이 일반적으로 단일 데이터베이스 조작으로 간주되는 조작을 여러 부분으로 세분하며, 그 중 많은 부분 또는 모두를 *단일 데이터베이스 파티션 내에서* 병렬로 수행할 수 있습니다.

그림14에서는 병렬로 수행 가능하며, 조회가 연속 방식으로 수행될 경우보다 더 빨리 리턴되는 네 부분으로 분해된 조회를 나타냅니다. 각 부분은 서로 복사됩니다. 파티션 내 병렬 처리를 사용하려면, 데이터베이스를 적합하게 구성해야 합니다. 사용자가 직접 병렬 처리 정도를 선택하거나 시스템이 임의로 선택하게 할 수 있습니다. 병렬 처리 정도는 병렬로 수행하는 조회 부분의 수입니다.

53 페이지의 표3에는 파티션 내 병렬 처리에 가장 적합한 하드웨어 환경이 나열되어 있습니다.

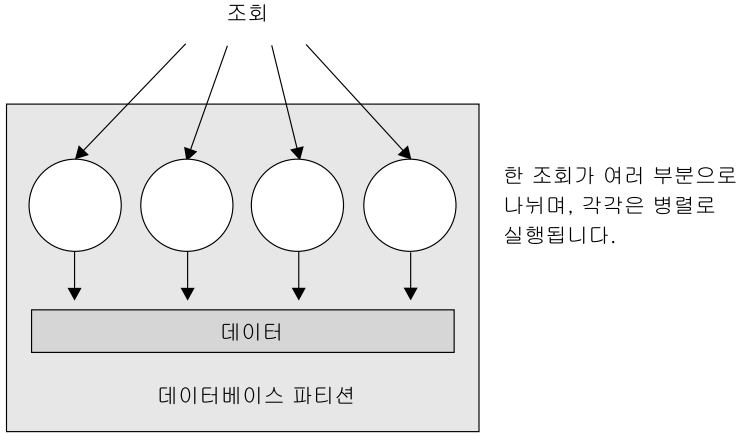

그림 14. 파티션 내 병렬 처리

파티션간 병렬 처리

*파티션간 병렬 처리*는 하나 이상의 머신에서 파티션된 데이터베이스의 다중 파티션에 하나의 조회를 여러 부분으로 분해하는 기능입니다. 조회는 병렬로 수행됩니다. (일부 유틸리티도 이 병렬 처리 유형을 실행합니다. 43 페이지의 『유틸리티 병렬 처리』에서 자세한 내용을 참조하십시오.)

파티션간 병렬 처리는 색인 작성, 데이터베이스 로드 또는 SQL 조회와 같이 일반적으로 단일 데이터베이스 조작으로 간주되는 조작을 여러 부분으로 세분하며, 이 중 많은 부분 또는 모두를 *하나 이상의 머신에서 파티션된 데이터베이스의 다중 파티션 전반에서* 병렬로 수행할 수 있습니다.

그림15에서는 병렬로 수행 가능하며, 단일 파티션에서 조회가 연속 방식으로 수행될 경우보다 더 빨리 리턴되는 네 부분으로 분해된 조회를 나타냅니다.

병렬 처리 정도는 작성하는 파티션 수와 노드 그룹의 정의 방식에 따라 결정됩니다.

53 페이지의 표3에는 파티션간 병렬 처리에 가장 적합한 하드웨어 환경이 나열되어 있습니다.

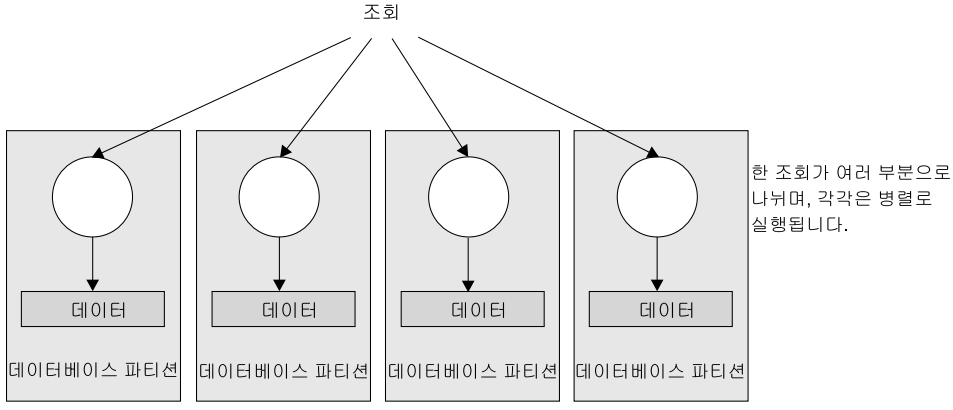

그림 15. 파티션간 병렬 처리

동시에 파티션 내 및 파티션간 병렬 처리

파티션 내 병렬 처리와 파티션간 병렬 처리를 동시에 사용할 수 있습니다. 이 조합은 두가지 병렬 처리 차원을 제공하여 조회가 처리되는 속도가 훨씬 빨라지게 합니다.

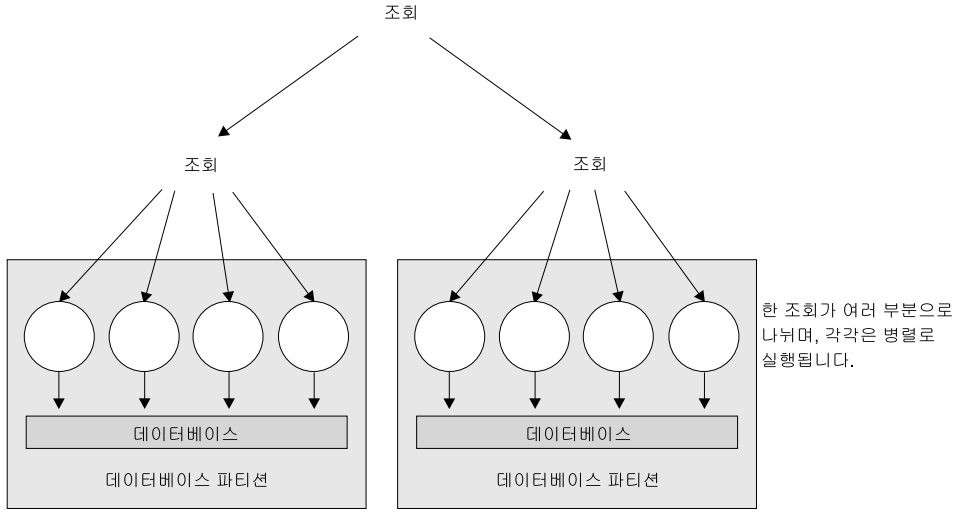

그림 16. 동시에 파티션간 및 파티션 내 병렬 처리

유틸리티 병렬 처리

DB2 유틸리티는 파티션 내 병렬 처리를 사용할 수 있습니다. 또한 파티션간 병렬 처리를 사용할 수도 있습니다. 다중 데이터베이스 파티션이 존재하면, 유틸리티는 병렬로 각 파티션에서 실행합니다.

로드 유틸리티는 파티션 내 병렬 처리와 입출력 병렬 처리를 사용할 수 있습니다. 데이터 로드는 CPU 집중적인 타스크입니다. 로드 유틸리티는 데이터의 해석 및 형식화와 같은 타스크에 대한 다중 프로세서를 사용합니다. 또한, 병렬 입출력 서버를 사용하여 데이터를 컨테이너에 병렬로 기록할 수도 있습니다. 로드 유틸리티에 대해 병렬 처리를 작동하도록 설정하는 방법에 대해서는 *데이터 이동 유틸리티 안내 및 참조서*에서 자세한 내용을 참조하십시오.

파티션된 데이터베이스 환경에서, 자동 로드 프로그램 유틸리티는 테이블이 상주하는 각 데이터베이스 파티션에서 LOAD 명령을 병렬 호출하여 파티션 내, 파티션간 및 입출력 병렬 처리를 사용합니다. 자동 로드 유틸리티에 대해서는 *데이터 이동 유틸리티 안내 및 참조서*에서 자세한 내용을 참조하십시오.

색인 작성시 데이터의 스캐닝과 연속 정렬은 병렬로 발생합니다. DB2는 색인 작성시 입출력 병렬 처리와 파티션 내 병렬 처리를 이용합니다. 이는 CREATE INDEX 명령문을 발행할 때, 재시작 동안(색인이 유효하지 않다고 표시될 경우) 및 데이터의 재구성 동안 색인 작성의 속도를 높이는 데 유용합니다.

데이터의 백업 및 복원은 입출력 바인드 타스크입니다. DB2는 백업 및 복원 조작을 실행할 경우 입출력 병렬 처리와 파티션 내 병렬 처리를 이용합니다. 백업은 병렬로 다중 테이블 공간 컨테이너로부터 읽고, 병렬로 복수의 백업 미디어를 비동기적으로 기록하여 입출력 병렬 처리를 이용합니다. 이들 유틸리티에 대해 병렬 처리를 작동하도록 설정하는 방법에 대해서는 *Command Reference*에 있는 BACKUP DATABASE 명령 및 RESTORE DATABASE 명령을 참조하십시오.

하드웨어 환경

이 절에서는 다음 하드웨어 환경의 개요를 제공합니다.

- 단일 프로세서상의 단일 파티션(단일프로세서)
- 다중 프로세서가 있는 단일 파티션(SMP)
- 다중 파티션 구성
  - 하나의 프로세서가 있는 파티션(MPP)
  - 다중 프로세서가 있는 파티션(SMP의 클러스터)
  - 논리 데이터베이스 파티션(또한 AIX용 DB2 Parallel Edition 버전 1의 다중 논리 노드 또는 MLN이라고도 함)

용량 및 확장성이 각 환경에서 설명됩니다. *용량*은 데이터베이스에 액세스할 수 있는 사용자 수와 응용프로그램 수를 나타냅니다. 이것은 메모리, 에이전트, 잠금, 입출력 및 저장영역 관리에 의해 결정되는 큰 부분에 있습니다. *확장성*은 데이터베이스의 규모를 확대하고 동일한 조작 특성과 응답 시간을 계속 표시하는 기능입니다.

단일 프로세서상의 단일 파티션

이 환경은 메모리와 디스크로 이루어져 있지만 하나의 CPU만 들어 있습니다(45 페이지의 그림17 참조). 이것은 독립형 데이터베이스, 클라이언트/서버 데이터베이

스, 직렬 데이터베이스, 단일 프로세서 시스템 그리고 단일 노드 또는 병렬이 아닌 환경과 같은 다양한 이름으로 나타냅니다.

이 환경의 데이터베이스는 데이터 및 시스템 자원(단일 프로세서 또는 CPU 포함)이 단일 데이터베이스 관리 프로그램에 의해 관리되는 부서나 소규모 지사의 필요에 부합됩니다.

53 페이지의 표3에서는 이 하드웨어 구성을 사용하는 데 가장 적합한 병렬 처리 유형이 나열되어 있습니다.

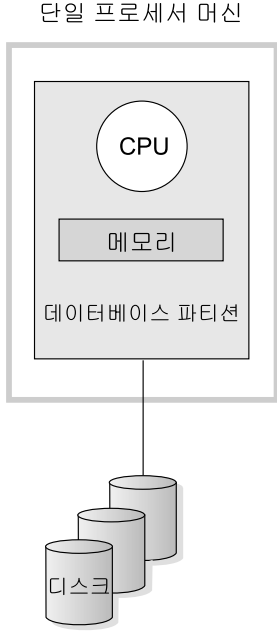

그림 17. 단일 프로세서상의 단일 파티션

용량 및 확장성

이 환경에서 더 많은 디스크를 추가할 수 있습니다. 각 디스크에 대해 하나 이상의 입출력 서버를 사용하면 동시에 둘 이상의 입출력 조작을 수행할 수 있습니다.

단일 프로세서 시스템은 프로세서가 다룰 수 있는 디스크 공간량에 따라 제한됩니다. 워크로드가 증가하면, 추가할 수 있는 메모리 또는 디스크와 같은 기타 구성요

소와 관계없이 사용자 요청을 빠르게 처리할 수 없습니다. 최대 용량 또는 가변성에 도달하면, 다중 프로세서가 있는 단일 파티션 시스템으로의 이동을 고려할 수 있습니다.

다중 프로세서가 있는 단일 파티션

이 환경은 같은 머신(47 페이지의 그림18 참조) 안에서 동일한 성능을 가진 여러 개의 프로세서로 이루어져 있으며, *대칭 다중 프로세서(SMP)* 시스템이라고 합니다. 디스크 공간 및 메모리 같은 자원을 *공유*합니다.

사용 가능한 다중 프로세서의 경우, 다른 데이터베이스 조작은 더 빨리 완료될 수 있습니다. DB2는 처리 속도를 향상시키기 위해 사용 가능한 프로세서간에 단일 조회 작업을 나눌 수도 있습니다. 데이터 로드, 테이블 공간 백업 및 복원 그리고 기존 데이터에서의 색인 작성과 같은 다른 데이터베이스 조작은 다중 프로세서를 이용할 수 있습니다.

53 페이지의 표3에서는 이 하드웨어 구성을 사용하는 데 가장 적합한 병렬 처리 유형이 나열되어 있습니다.

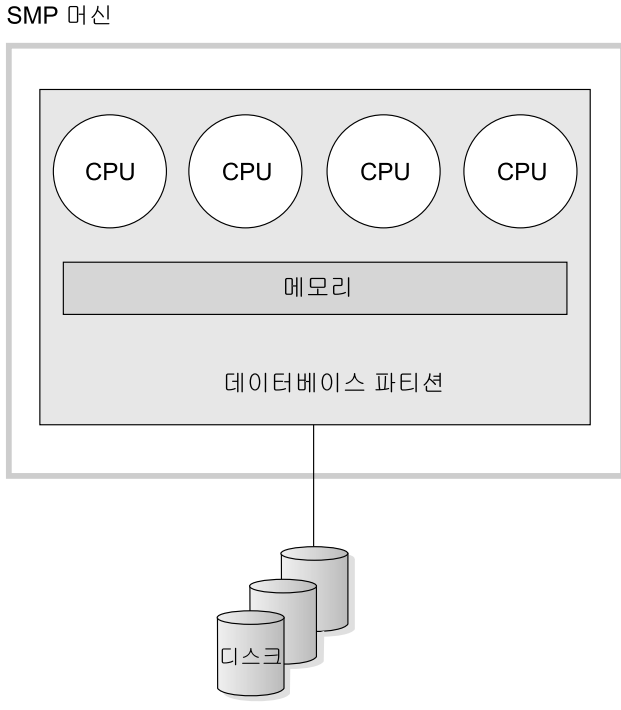

그림 18. 단일 파티션 데이터베이스 대칭 다중 프로세서 시스템

용량 및 확장성

이 환경에 프로세서를 추가할 수 있습니다. 그러나 여러 프로세서가 같은 데이터에 액세스할 수 있으므로, 이 환경에서 비지니스 조작이 계속 증가하면 한계가 나타날 수 있습니다. 공유 메모리 및 공유 디스크의 경우, 모든 데이터베이스 데이터를 효과적으로 공유합니다.

디스크 수를 증가시켜 프로세서와 연관된 데이터베이스 파티션의 입출력 용량을 증가시킬 수 있습니다. 특별히 입출력 요청을 다루는 입출력 서버를 설정할 수 있습니다. 각 디스크에 대해 하나 이상의 입출력 서버를 사용하면 동시에 둘 이상의 입출력 조작을 수행할 수 있습니다.

최대 용량 또는 확장성에 도달하면, 다중 파티션이 있는 시스템으로의 이동을 고려할 수 있습니다.

다중 파티션 구성

데이터베이스를 자체 머신에서 다중 파티션으로 나눌 수 있습니다. 다중 데이터베이스 파티션이 있는 다중 머신은 함께 그룹화될 수 있습니다. 이 절에서는 다음 파티션 구성을 설명합니다.

- 하나의 프로세서로 시스템에서 파티션
- 다중 프로세서로 시스템에서 파티션
- 논리 데이터베이스 파티션

하나의 프로세서가 있는 파티션

이 환경에는 여러 개의 데이터베이스 파티션이 있습니다. 각 파티션은 자체 머신에 상주하며, 자체 프로세서, 메모리 및 디스크를 갖습니다(49 페이지의 그림19). 모든 머신은 통신 기능에 의해 연결됩니다. 이 환경은 클러스터, 단일 프로세서의 클러스터, MPP(massively parallel processing) 환경 및 어떤 것도 공유하지 않는 구성을 포함하여 여러 가지 다른 이름으로 나타냅니다. 후자의 이름은 이 환경의 자원 배열을 정확히 나타냅니다. SMP 환경과는 달리, MPP 환경에는 공유된 메모리 또는 디스크가 없습니다. MPP 환경은 메모리 및 디스크의 공유로 야기된 제한사항을 제거합니다.

파티션된 데이터베이스 환경에서 데이터베이스는 둘 이상의 파티션에 물리적으로 나뉘지만 논리적으로는 하나로 남아 있습니다. 데이터가 파티션으로 나누어진 사실을 대부분의 사용자는 명확히 알 수 있습니다. 데이터베이스 관리 프로그램 간에 작업이 나뉠 수 있으며, 각 파티션의 각 데이터베이스 관리 프로그램은 자체 데이터베이스 부분에 대해 작업합니다.

53 페이지의 표3에서는 이 하드웨어 구성을 사용하는 데 가장 적합한 병렬 처리 유형이 나열되어 있습니다.

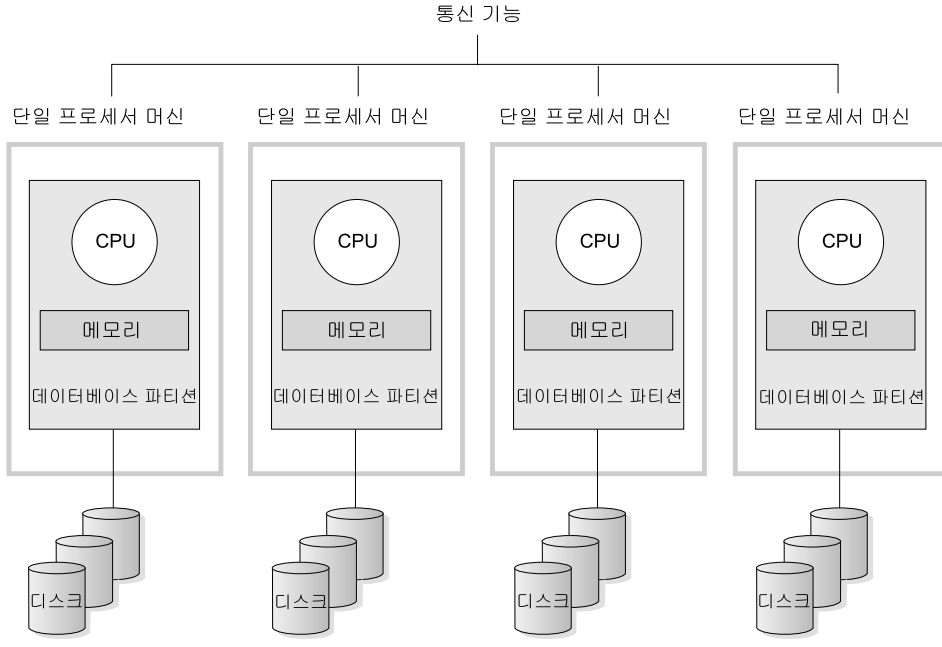

그림 19. 대용량의 병렬 처리 시스템

용량 및 확장성: 이 환경에서는 더 많은 데이터베이스 파티션(노드)을 구성에 추가할 수 있습니다. 일부 플랫폼에서, 예를 들어 RS/6000 SP에서 최대 수는 512 노드입니다. 그러나, 많은 수의 머신과 인스턴스를 관리하다보면 한계가 생길 수 있습니다.

최대 용량 또는 가변성에 도달하면, 각 파티션에 다중 프로세서가 있는 시스템으로의 이동을 고려할 수 있습니다.

다중 프로세서가 있는 파티션

각 파티션에 단일 프로세서가 있는 구성에 대한 대안으로 하나의 파티션에 다중 프로세서가 있는 구성이 있습니다. 이를 *SMP 클러스터*(50 페이지의 그림20)라고 합니다.

이 구성은 SMP 병렬 처리와 MPP 병렬 처리의 이점을 결합합니다. 다중 프로세서를 통해 조회가 단일 파티션에서 실행될 수 있습니다. 또한, 다중 파티션을 통해 조회가 병렬로 실행될 수도 있습니다.

53 페이지의 표3에서는 이 하드웨어 구성을 사용하는 데 가장 적합한 병렬 처리 유형이 나열되어 있습니다.

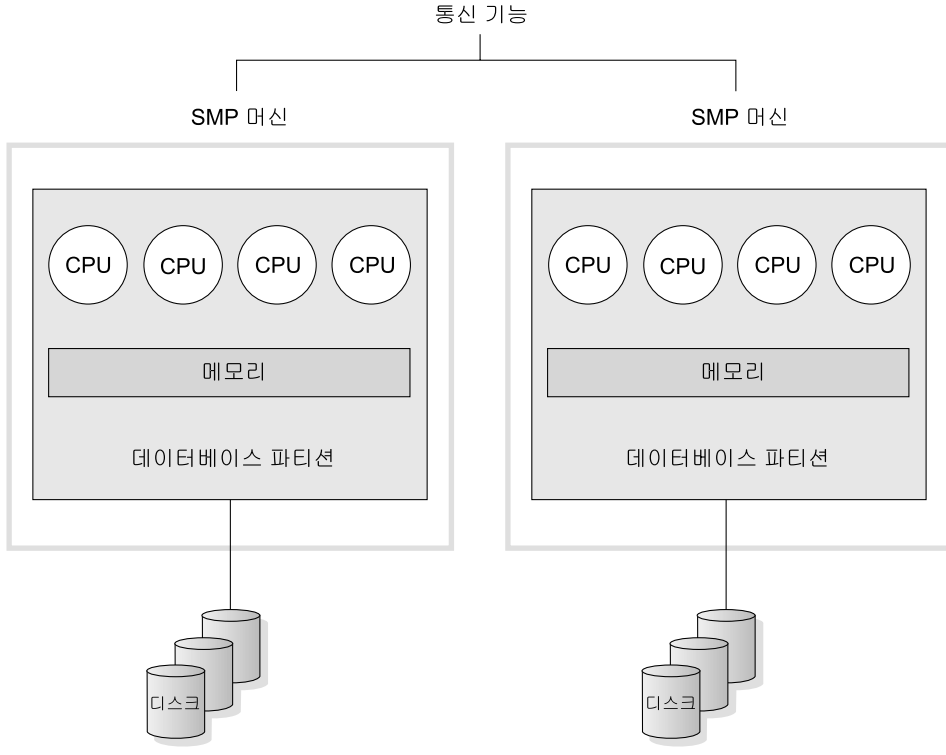

그림 20. SMP의 클러스터

용량 및 확장성: 이 환경에서는 더 많은 데이터베이스 파티션을 추가할 수 있으며, 기존 데이터베이스 파티션에 더 많은 프로세서를 추가시킬 수 있습니다.

논리 데이터베이스 파티션

논리 데이터베이스 파티션은 전체 머신 제어가 제공되지 않는다는 점에서 물리적 파티션과는 다릅니다. 머신은 자원을 공유하지만, 데이터베이스 파티션은 자원을 공유하지 않습니다. 프로세서는 공유되지만, 디스크와 메모리는 공유되지 않습니다.

논리 데이터베이스 파티션은 확장성을 제공합니다. 다중 논리 파티션에서 수행되는 다중 데이터베이스 관리 프로그램은 단일 데이터베이스 관리 프로그램이 사용하는 것보다 사용 가능한 자원을 더 완벽하게 사용할 수 있습니다. 51 페이지의 그림

21에서는 더 많은 파티션을 추가하여 SMP 머신에서 더 많은 확장성을 얻을 수 있으며, 특히 많은 프로세서가 있는 머신의 경우에 특히 그렇습니다. 데이터베이스의 파티션을 나누면, 각 파티션을 개별적으로 관리하고 복구할 수 있습니다.

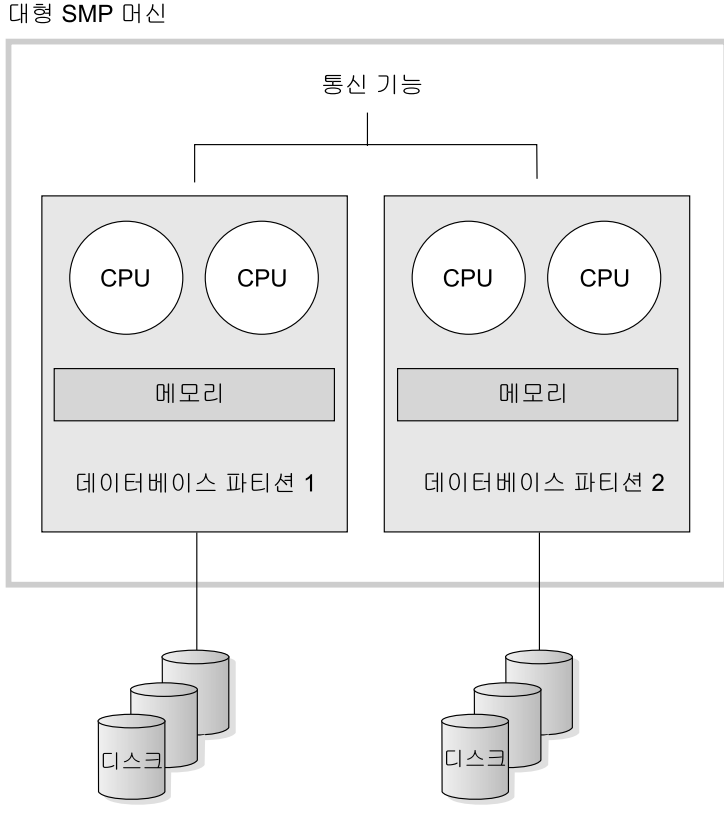

그림 21. 파티션된 데이터베이스, 대칭 다중 프로세서 시스템

52 페이지의 그림22에서는 그림21의 구성을 늘려 처리 능력을 증가시킬 수 있다는 사실을 보여줍니다.

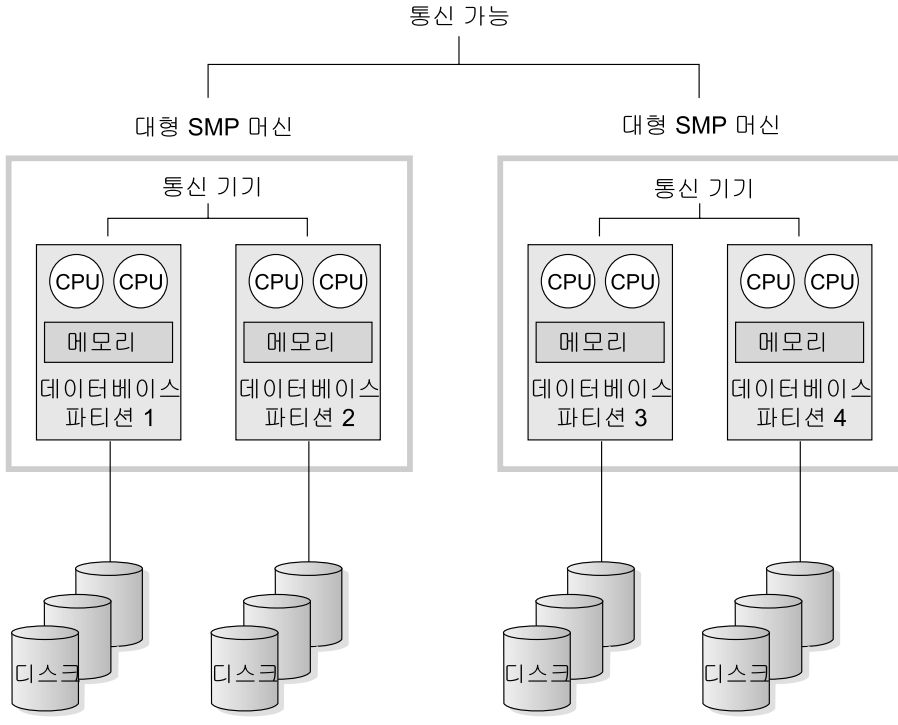

그림 22. 파티션된 데이터베이스, 같이 클러스터된 대칭 다중 프로세서 시스템

53 페이지의 표3은 이 하드웨어 환경을 사용하는 데 가장 적합한 병렬 처리 유형입니다.

주: 둘 이상의 파티션이 같은 머신(프로세서 수에 관계없이)에 공존하는 기능은 고가용성 구성 및 장애 복구 전략 설계에서 융통성을 높인다는 점을 유의하십시오. 머신 고장의 경우, 데이터베이스 파티션이 자동으로 이동되어 같은 데이터베이스의 다른 파티션이 이미 들어 있는 다른 머신에서 재시작될 수 있습니다. 205 페이지의 『제11장 고가용성 및 장애 복구 지원 소개』에서 자세한 내용을 참조하십시오.

각 하드웨어 환경에 가장 적합한 병렬 처리의 요약

다음 테이블에서는 다양한 하드웨어 환경에 가장 적합한 병렬 처리의 유형을 요약합니다.

표 3. 각 하드웨어 환경에서 가능한 병렬 처리의 유형

하드웨어 환경	입출력 병렬 처리	조회 내 병렬 처리	
		파티션 내 병렬 처리	파티션간 병렬 처리
단일 파티션, 단일 프로세서	예	아니오(1)	아니오
단일 파티션, 다중 프로세서(SMP)	예	예	아니오
다중 파티션, 하나의 프로세서(MPP)	예	아니오(1)	예
다중 파티션, 다중 프로세서(SMP 클러스터)	예	예	예
논리 데이터베이스 파티션	예	예	예
주: (1) 병렬 처리 수준 설정에는(구성 매개변수 중 하나를 사용하여) 단일 프로세서 시스템에서보다, 특히 실행하는 조회가 CPU를 완전히 사용하지 못하는 경우(예를 들어, 조회가 입출력 바운드일 경우)에 훨씬 유리합니다.			

제5장 데이터 웨어하우징 정보

DB2 Universal Database는 데이터 웨어하우스 처리를 자동화하는 Data Warehouse Center 구성요소를 제공합니다. Data Warehouse Center를 사용하여 웨어하우스에 포함할 데이터를 정의할 수 있습니다. 그런 다음 Data Warehouse Center를 사용하여 웨어하우스에서 자동으로 데이터의 새로 고침을 스케줄할 수 있습니다.

이 절에서는 데이터 웨어하우스 및 데이터 웨어하우징 타스크의 개요를 제공합니다. *Data Warehouse Center 관리 안내서* 및 Data Warehouse Center 온라인 도움말에서 웨어하우징에 대한 자세한 내용과 Data Warehouse Center 사용에 대한 자세한 내용을 참조하십시오.

데이터 웨어하우징이란?

조작 데이터(사용자 비즈니스의 매일 트랜잭션을 수행하는 데이터)가 들어 있는 시스템에는 비지니스 분석에 유용한 정보가 들어 있습니다. 예를 들어, 분석자는 이상이 있는지 보기 위해 또는 장래 판매를 계획하기 위해 언제 어떤 지역에서 어떤 제품이 팔리는지에 대한 정보를 사용할 수 있습니다.

그러나, 직접 조작 데이터에 액세스하는 분석자에게는 여러 가지 문제점이 있습니다.

- 조작 데이터베이스를 조회하는 전문 기술을 가지고 있지 않을 수 있습니다. 예를 들어, IMS 데이터베이스 조회는 데이터 조작 언어의 특수화된 유형을 사용하는 응용프로그램을 요구합니다. 일반적으로, 조작 데이터베이스를 조회하는 전문 기술이 있는 프로그래머는 데이터베이스 및 관련 응용프로그램을 유지보수하는 데 전념하는 작업을 갖습니다.
- 성능은 은행의 데이터베이스와 같이 여러 조작 데이터베이스에서 중요합니다. 시스템은 임시 조회를 작성하는 사용자를 핸들할 수 없습니다.
- 일반적으로 조작 데이터는 비지니스 분석자가 사용하기에 최적의 형식으로 되어 있지 않습니다. 예를 들어, 제품, 지역 및 계절별로 요약된 판매 데이터는 원시 데이터보다는 분석하기에 더 유용합니다.

데이터 웨어하우징은 이 문제점을 해결합니다. *데이터 웨어하우징*에서는 조작용 데이터에서 발췌된 후 일반 사용자 결정을 위해 변환되는 *정보용 데이터*의 저장을 작성합니다. 예를 들어, 데이터 웨어하우스 도구는 조작용 데이터베이스에서 모든 판매 데이터를 복사하고, 데이터를 요약하기 위해 계산을 수행하고, 요약된 데이터를 조작 데이터에서 각 데이터베이스 내의 목표로 기록하십시오. 일반 사용자는 조작 데이터베이스에 영향을 미치지 않고 각 데이터베이스(*웨어하우스*)를 조회할 수 있습니다.

다음 절에서는 데이터 웨어하우스를 작성하고 유지보수하는 데 사용할 오브젝트(주제 영역, 웨어하우스 소스, 웨어하우스 목표, 에이전트, 에이전트 사이트, 단계 및 프로세스)를 설명합니다.

주제 영역

*주제 영역*은 비지니스의 논리 영역에 관련된 프로세스를 식별하고 그룹화합니다. 예를 들어, 마케팅 및 판매 데이터의 웨어하우스를 빌드하는 중이라면, 판매 주제 영역 및 마케팅 주제 영역을 정의할 수 있습니다. 그런 다음 판매 주제 영역 아래에서 판매와 관련된 프로세스를 추가할 수 있습니다. 유사하게, 마케팅 주제 영역 아래에서 마켓팅 데이터에 관련된 정의를 추가할 수 있습니다.

웨어하우스 소스

*웨어하우스 소스*는 웨어하우스에 데이터를 제공할 테이블 및 파일을 식별합니다. Data Warehouse Center는 웨어하우스 소스에서 권장 스펙을 사용하여 데이터에 액세스하고 선택할 수 있습니다. 소스는 사용자 웨어하우스와 연결성을 가진 거의 모든 관계형 또는 비관계형 소스(테이블, 뷰 또는 파일)일 수 있습니다.

웨어하우스 목표

*웨어하우스 목표*는 일반 사용자가 사용할 수 있도록 변환된 데이터가 들어 있는 데이터베이스 테이블 또는 파일입니다. 웨어하우스 소스와 같이, 웨어하우스 목표는 데이터를 Data Warehouse Center 단계에 제공할 수 있습니다.

웨어하우스 에이전트 및 에이전트 사이트

Data Warehouse Center *에이전트*는 데이터 소스 및 목표 웨어하우스 간의 데이터 흐름을 관리합니다. 에이전트는 Windows NT, AIX, OS/2, OS/390, OS/400

및 SUN Solaris 운영 체제에서 사용 가능합니다. 에이전트는 ODBC(Open Database Connectivity) 드라이버나 DB2 CLI를 사용하여 여러 데이터베이스와 통신합니다.

여러 에이전트는 소스와 목표 웨어하우스간의 데이터 전송을 핸들할 수 있습니다. 사용하는 에이전트의 수는 웨어하우스를 통해 이동하려는 데이터의 볼륨 및 기존 연결성 구성에 좌우됩니다. 에이전트의 추가 인스턴스는 동일한 에이전트를 요구하는 다중 프로세스가 동시에 실행중인 경우 생성될 수 있습니다.

에이전트는 지역 또는 원격이 될 수 있습니다. *지역 웨어하우스 에이전트*는 웨어하우스 서버와 동일한 머신에 설치된 에이전트입니다. *원격 웨어하우스 에이전트*는 웨어하우스 서버와의 연결성을 가진 또다른 머신에 설치된 에이전트입니다.

*에이전트 사이트*는 에이전트 소프트웨어가 설치된 워크스테이션의 논리 이름입니다. 에이전트 사이트 이름은 TCP/IP 호스트 이름과 같지 않습니다. 단일 실제 머신은 하나의 TCP/IP 호스트 이름만 가질 수 있습니다. 그러나, 단일 머신에 다중 에이전트 사이트를 정의할 수 있습니다. 논리 이름은 각 에이전트 사이트를 식별합니다.

기본 VW 에이전트 사이트라고 하는 *기본 에이전트 사이트*는 웨어하우스 제어 데이터베이스의 설치 중 Data Warehouse Center가 정의하는 Windows NT에 있는 논리 에이전트입니다.

단계 및 프로세스

*단계*는 정의하는 Data Warehouse Center에 있는 논리 엔터티입니다.

- 출력 테이블 또는 파일의 구조
- 출력 테이블 또는 파일을 처리하는 메카니즘(SQL 또는 프로그램)
- 출력 테이블 또는 파일이 처리되는 스케줄

단계에서는 SQL문을 사용하거나 프로그램을 호출하여 데이터를 이동하며 변환합니다. 단계를 수행할 때, 웨어하우스 소스와 웨어하우스 목표간의 데이터 전송 및 해당 데이터의 변환이 발생합니다.

*프로세스*에는 변환 및 이동 타스크를 수행하는 단계들이 들어 있습니다. 일반적으로, 프로세스는 데이터베이스 테이블 또는 파일일 수 있는 하나 이상의 웨어하우스 소스에서 데이터를 발췌하여 웨어하우스 데이터베이스에서 웨어하우스 목표를 처리합니다. 그러나, 임의의 웨어하우스 소스 또는 목표를 지정하지 않는 프로그램을 시작하는 프로세스를 정의할 수도 있습니다.

필요시 단계를 수행하거나 설정 시간에 수행할 단계를 스케줄할 수 있습니다. 오직 한번만 수행하도록 단계를 스케줄하거나 매주 금요일마다 반복 수행하도록 단계를 스케줄할 수 있습니다. 또한 순서대로 수행할 단계를 스케줄하여 한 단계가 수행을 완료하면 다음 단계가 수행을 시작하게 할 수 있습니다. 또다른 단계의 (성공 또는 실패) 완료시 수행할 단계를 스케줄할 수 있습니다. 프로세스를 스케줄하는 경우, 프로세스의 첫 번째 단계는 스케줄 시간에 수행합니다.

단계 또는 프로세스가 수행하면, 다음에 의해 데이터를 저장할 수 있습니다.

- 새로운 데이터로 웨어하우스 목표에서 모든 데이터 바꾸기
- 새로운 데이터를 기존 데이터에 추가
- 데이터의 각 편집판 추가

다음과 같은 타스크를 수행하기 위해 Data Warehouse Center를 원함을 가정하십시오.

1. 여러 데이터베이스에서 데이터를 추출합니다.
2. 데이터를 단일 형식으로 변환합니다.
3. 데이터를 데이터 웨어하우스 내의 테이블에 씁니다.

각 단계를 포함한 프로세스를 작성합니다. 각 단계는 데이터베이스에서 데이터를 추출하거나 올바른 형식으로 변환하는 것과 같이 별도의 타스크를 수행합니다. 그런 다음 또다른 단계를 사용하여 변환된 데이터가 들어 있는 목표 테이블을 처리하기 위해 또다른 단계를 사용합니다.

다음 절에서는 Data Warehouse Center에서 찾는 다양한 단계 유형을 설명합니다. 단계에 대해서는 *Data Warehouse Center 관리 안내서*에서 자세한 내용을 참조하십시오.

SQL 단계

SQL 단계는 SQL SELECT문을 사용하여 웨어하우스 소스에서 데이터를 추출하고, INSERT문을 생성하여 데이터를 웨어하우스 목표로 삽입합니다.

프로그램 단계

AS/400용 DB2 프로그램, OS/390용 DB2, UDB용 DB2 프로그램, Visual Warehouse 5.2 DB2 프로그램, OLAP Server 프로그램, 파일 프로그램 및 복제 등과 같은 여러 가지 유형의 프로그램 단계가 있습니다. 이들 단계는 사전정의된 프로그램 및 유틸리티를 수행합니다.

변환기 단계

변환기 단계는 데이터를 변환하기 위해 사용할 수 있는 통계 또는 웨어하우스 변환기를 지정하는 사용자 정의 함수 및 저장 프로시듀어입니다. 변환기를 사용하여 데이터를 삭제, 전환 및 피벗하고, 기본 키 및 기간 테이블을 생성하고 다양한 통계를 계산합니다.

변환기 단계에서 통계 또는 웨어하우스 변환기 중 하나를 지정합니다. 프로세스를 수행할 때 변환기는 데이터를 하나 이상의 웨어하우스 목표에 기록합니다.

사용자 정의 프로그램 단계

*사용자 정의 프로그램 단계*는 Data Warehouse Center가 시작하게 하려는 응용 프로그램을 나타내는 Data Warehouse Center 내의 논리 엔터티입니다. 웨어하우스 에이전트는 다음과 같이 사용자 정의 프로그램 단계를 시작할 수 있습니다.

- 웨어하우스 목표의 처리 중
- 웨어하우스 목표의 처리 이후
- 자체에 의해

예를 들어, 다음과 같은 프로세스를 수행할 사용자 정의 프로그램을 작성할 수 있습니다.

1. 테이블에서 데이터 내보내기
2. 데이터 조작
3. 중간 출력 자원이나 웨어하우스 목표에 데이터 쓰기

웨어하우징 타스크

다음과 같은 타스크와 관련된 데이터 웨어하우스 작성합니다.

- 사용자 웨어하우스에서 사용할 프로세스를 식별하고 그룹화하는 주제 영역 정의.
- 소스 데이터(또는 조작 데이터) 검색 및 웨어하우스 소스 정의.
- 웨어하우스로서 사용할 데이터베이스 작성 및 웨어하우스 목표 정의.
- 프로세스를 정의하여 웨어하우스 데이터베이스의 형식으로 소스 데이터를 이동하고 변환하는 방법 지정.
- 정의한 단계를 테스트하고 자동으로 수행하도록 스케줄.
- 보안을 정의하여 웨어하우스 관리, 데이터베이스 사용 모니터링.
- DB2 Warehouse Manager 패키지가 있는 경우, 웨어하우스에서 데이터의 정보 카탈로그 작성. 정보 카탈로그는 비지니스 메타데이터가 들어 있는 데이터베이스입니다. 메타데이터는 사용자가 조직에서 사용가능한 데이터 및 정보를 식별하고 찾도록 돕습니다. 웨어하우스의 일반 사용자는 카탈로그를 검색하여 어떤 테이블을 조회할지 판별할 수 있습니다.
- 웨어하우스에서 테이터의 스타 스키마 모델 정의. 스타 스키마는 다중 차원 테이블(비지니스의 측면 설명) 및 하나의 사실 테이블(비지니스에 대한 사실이 들어 있음)로 구성된 특수화된 설계입니다. 예를 들어, 소프트 드링크를 제조하는 경우 어떤 차원 테이블은 제품, 시장 및 시간입니다. 사실 테이블에는 계절별로 각 지역에서 순서화된 제품에 대한 트랜잭션 정보가 들어 있습니다.
- 순서 정보로 차원 테이블에서 세부사항을 조합하기 위해 사실 테이블 및 차원 테이블 조인. 예를 들어, 제품 차원을 사실 테이블에 조인하여 각 제품이 주문에 대해 패키지되는 방법에 대한 정보를 추가할 수 있습니다.

*비지니스 인텔리전스 자습서*를 사용하거나 *DB2 Universal Database 빠른 안내*를 보거나 *Data Warehouse Center 관리 안내서*를 읽고 이 타스크와 다른 타스크에 대한 자세한 내용을 참조하십시오.

제6장 Spatial Extender 정보

이 절에서는 Spatial Extender의 목적을 설명하고 처리하는 데이터를 설명하여 Spatial Extender를 소개합니다. Spatial Extender 사용에 대해서는 *Spatial Extender 사용자 안내 및 참조서*에서 자세한 내용을 참조하십시오.

Spatial Extender 목적

Spatial Extender를 사용하여 지리학적 기능에 대한 공간 정보를 생성하고 분석하게 하는 오브젝트, 데이터 및 응용프로그램을 복합한 *지리학적 정보 시스템*(GIS)을 작성하십시오. *지리학적 기능*은 지구 표면을 형성하는 오브젝트와 지구를 차지하는 오브젝트를 포함합니다. 이 기능들은 자연 환경(강, 숲, 언덕 및 사막 등)과 문화 환경(도시, 주택, 사무실 건물, 경계표 등) 모두로 구성됩니다.

*공간 정보*는 다음과 같은 것을 포함합니다.

- 환경에 대한 지리학적 기능의 위치(예: 병원 및 클리닉이 위치한 도시 내의 지점 또는 국지 지진대의 도시 거주지의 근접도)
- 지리학적 기능이 서로 관련된 방식(예: 특정 수계가 특정 지역을 둘러싸거나 수계의 지류에 걸쳐 있는 지역에 있는 특정 다리에 대한 정보)
- 하나 이상의 지리학적 기능에 적용하는 측정 방법(예를 들어, 사무실 건물과 관련 구획선 또는 조류 보존 경계선간의 거리)

자체나 기존 데이터베이스 출력과 결합한 공간 정보는 프로젝트를 설계하고 비지니스 및 규정 결정을 돕습니다. 예를 들어, 복지 지역 관리 프로그램은 복지 지원자 및 수용자가 실제로 해당 지역이 서비스하는 영역 내에 살고 있는지 확인하는 것이 필요합니다. Spatial Extender는 서비스 영역의 위치와 지원자 및 수용자의 주소에서 이 정보를 얻습니다.

또는 레스토랑 체인의 소유자가 근처 도시에서 운영하려고 한다고 가정하십시오. 새로운 레스토랑을 열 장소를 결정하려면, 소유자가 다음과 같은 질문에 응답할 필요가 있습니다. 이 도시의 어디에 레스토랑을 빈번하게 찾을 사람들이 집중되어 있는가? 주요 고속도로는 어디에 있는가? 범죄 비율은 어디가 가장 낮은가? 경쟁 레

스토랑은 어디에 위치해 있는가? Spatial Extender는 그러한 질문에 응답할 수 있도록 비주얼 표시장치에 공간 정보를 생산할 수 있으며, 주요 데이터베이스 관리 시스템은 표시장치를 설명할 텍스트 및 레이블을 생성할 수 있습니다.

그래픽 기능을 표시하는 데이터

이 절에서는 공간 정보를 확보하기 위해 생성, 저장 및 조작하는 데이터의 개요를 제공합니다. 다루는 주제는 다음과 같습니다.

- 지리학적 기능이 표시되는 방식
- 공간 데이터의 특성
- 공간 데이터를 생성하는 방식

데이터가 지리학적 기능을 나타내는 방식

Spatial Extender에서 지리학적인 기능은 테이블 또는 뷰에서 행별로 표시되거나 그러한 행의 부분에 의해 표시될 수 있습니다. 예를 들어, 사무실 건물 및 거주지와 같은 두 개의 지리학적 기능을 고려하십시오. 63 페이지의 그림23에서, 각 BRANCHES 테이블 행은 은행의 지사를 나타냅니다. 전체로서 얻은 CUSTOMERS 테이블의 각 행은 변형으로서 은행의 고객을 나타냅니다. 그러나 특별히 고객의 주소가 들어 있는 셀인 각 행의 일부는 고객의 거주지를 나타내는 것으로 볼 수 있습니다.

이 테이블에는 은행의 지사 및 고객을 식별하고 설명하는 데이터가 들어 있습니다. 그러한 데이터를 *속성 데이터*라고 합니다.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141

CUSTOMERS

ID	LAST NAME	FIRST NAME	ADDRESS	CITY	STATE	ZIP	CHECKING	SAVINGS
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	CA	95141	A	A

그림 23. 지리학적 기능을 나타내는 테이블 행, 주소 데이터가 지리학적 기능을 나타내는 테이블 행. BRANCHES 테이블에 있는 데이터 행은 은행의 지사를 나타냅니다. CUSTOMERS 테이블에 있는 주소 데이터의 셀은 고객의 거주지를 나타냅니다.

속성 데이터의 부속 집합(지사 및 고객 주소를 나타내는 값)은 공간 정보를 생성하는 값으로 변환될 수 있습니다. 예를 들어, 그림에 표시된 것처럼 지사 주소는 92467 Airzone Blvd., San Jose CA 95141이고, 고객 주소는 9 Concourt Circle, San Jose CA 95141입니다. Spatial Extender는 이 주소를 각각의 주변 환경과 관련하여 지사와 고객의 집이 있는 장소를 나타내는 값으로 변환시킬 수 있습니다. 그림24에서는 그러한 값이 들어 있는 새 컬럼이 있는 BRANCHES 및 CUSTOMERS 테이블을 보여줍니다.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP	LOCATION
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141	

CUSTOMERS

ID	LAST NAME	FIRST NAME	ADDRESS	CITY	STATE	ZIP	LOCATION	CHECKING	SAVINGS
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	CA	95141		A	A

그림 24. 추가된 공간 컬럼이 있는 테이블. 각 테이블에서, LOCATION 컬럼은 주소에 해당하는 좌표를 포함합니다.

주소 및 유사한 식별자가 공간 정보의 시작 지점으로 사용될 때, *소스 데이터*라고 합니다. 소스 정보에서 얻은 값은 공간 정보를 생성하므로, 이 값을 *공간 데이터*라고 합니다. 다음 절에서는 공간 데이터를 설명하며 관련된 데이터 유형을 소개합니다.

공간 데이터의 특성

많은 공간 데이터는 좌표로 구성됩니다. *좌표*는 참조 지점과 관련된 위치를 나타내는 숫자입니다. 예를 들어, 위도는 적도에 상대적인 위치를 나타내는 좌표입니다. 경도는 그리니치 자오선에 상대적인 위치를 나타내는 좌표입니다. 그러므로, Yellowstone National Park의 위치는 위도(적도의 44.45도 북쪽)와 경도(그리니치 자오선의 110.40도 서쪽)로 정의됩니다.

위도, 경도, 관련 참조 지점 및 기타 연관된 매개변수를 모두 *좌표 시스템*으로 나타냅니다. 이도 및 경도 이외의 값에 근거한 좌표 시스템도 있습니다. 이 좌표 시스템은 자체의 위치 측정, 참조 지점 및 기타 구분 매개변수를 갖습니다.

가장 단순한 공간 데이터 항목은 단일 지리학적 기능의 위치를 정의하는 두 개의 좌표로 구성됩니다. *데이터 항목* 항목은 관계 테이블에서 셀을 차지하는 값으로 나타냅니다. 더 확장된 공간 데이터 항목은 길 또는 강과 같은 선형 경로를 정의하는 여러 좌표로 구성됩니다. 세 번째 종류는 토지 1 구획 또는 범람원의 테두리와 같은 영역의 둘레를 정의하는 좌표로 구성됩니다.

각 공간 데이터 항목은 공간 데이터 유형이 인스턴스입니다. 위치를 표시하는 두 개의 좌표에 대한 데이터 유형은 ST_Point이며, 선형 경로를 정의하는 좌표의 데이터 유형은 ST_LineString이며, 둘레를 정의하는 좌표의 유형은 ST_Polygon입니다. 데이터 유형의 다른 공간 데이터과 함께 이 유형은 단일 계층 구조에 속하는 구조화된 유형입니다.

공간 데이터를 확보하는 곳

다음에 의해 공간 데이터를 확보할 수 있습니다.

- 속성 데이터에서 공간 테이터 얻기
- 다른 공간 데이터에서 공간 테이터 얻기
- 공간 데이터 가져오기

속성 데이터를 소스 데이터로서 사용

속성 데이터(주소와 같은)에서 공간 데이터를 얻는 것을 *공간형 변환*이라고 합니다. 63 페이지의 그림24에서는 공간 데이터용으로 지정된 두 개의 컬럼을 표시합니다. 하나는 BRANCHES 테이블에 있는 컬럼이며, 다른 하나는 CUSTOMERS

테이블에 있는 컬럼입니다. Spatial Extender는 이 테이블에서 주소를 공간형 변환하며 결과 출력(주소에 해당하는 좌표)을 컬럼에 위치시킨다고 상상하십시오. 그림25에서는 이 결과를 보여줍니다.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP	LOCATION
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141	1653 3094

CUSTOMERS

ID	LAST NAME	FIRST NAME	ADDRESS	CITY	STATE	ZIP	LOCATION	CHECKING	SAVINGS
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	CA	95141	953 1527	A	A

그림 25. 소스 데이터에서 얻은 공간 데이터를 포함하는 테이블. CUSTOMERS 테이블의 LOCATION 컬럼에는 공간형 변환 프로그램이 ADDRESS, CITY, STATE 및 ZIP 컬럼에 있는 주소에서 얻은 좌표가 들어 있습니다. 유사하게 BRANCHES 테이블의 LOCATION 컬럼에는 공간형 변환 프로그램이 ADDRESS, CITY, STATE 및 ZIP 컬럼에 있는 주소에서 얻은 좌표가 들어 있습니다.

Spatial Extender는 *geocoder*라는 함수를 사용하여 속성 데이터를 공간형 변환하고 결과 공간 데이터를 컬럼에 위치시킵니다.

기타 공간 데이터를 소스 데이터로서 사용

공간 데이터는 속성 데이터뿐만 아니라 다른 공간 데이터에서도 생성될 수 있습니다. 예를 들어, 지사가 BRANCHES 테이블에 정의된 은행에서는 얼마나 많은 고객이 각 지사의 5마일 내에 위치하는가를 알고자 한다고 가정하십시오. 은행이 데이터베이스에서 이 정보를 얻기 전에, 각 지사 주위의 5마일 반경 내에 있는 지역의 정의를 데이터베이스에 제공해야 합니다. Spatial Extender 함수, ST_Buffer는 그러한 정의를 작성할 수 있습니다. 입력과 같이 각 지사의 좌표를 사용하여, 이 함수는 원하는 지역의 둘레를 구분하는 좌표를 생성할 수 있습니다. 66 페이지의 그림26에서는 ST_Buffer가 제공하는 정보가 있는 BRANCHES 테이블을 보여줍니다.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP	LOCATION	SALES_AREA
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141	1653 3094	1002 2001, 1192 3564, 2502 3415, 1915 3394, 1002 2001

그림 26. 기존의 공간 데이터에서 얻은 새로운 공간 데이터를 포함하는 테이블. LOCATION 컬럼에 있는 좌표로부터 ST_Buffer 함수에 의해 얻은 SALES_AREA 컬럼에 있는 좌표

ST_Buffer 이외에, Spatial Extender는 기존의 공간 데이터에서 새로운 공간 데이터를 얻는 다른 여러 가지 함수를 제공합니다.

공간 데이터 가져오기

공간 데이터를 확보하는 세 번째 방법은 Spatial Extender가 지원하는 형식 중 하나로 되어 있는 파일에서 공간 데이터를 가져오는 것입니다. 이 파일에는 보통 맵에 적용되는 데이터가 들어 있습니다(조사 트랙, 범람원, 지진 단층 등). 사용자가 생성하는 공간 데이터와 결합하여 그러한 데이터를 사용하여, 사용 가능한 지리학적 정보를 늘릴 수 있습니다. 예를 들어, 공용 작업 부서가 거주 집단의 취약한 위험 요소를 판별하는 것이 필요한 경우, ST_Buffer를 사용하여 그 거주 집단 주위의 지역을 정의한 후 범람원 및 지진 단층으로 데이터를 가져와 이 문제점 영역 중 어느 것이 그 지역과 중첩되는지 알 수 있습니다.

제3부 데이터베이스 설계

제7장 논리 데이터베이스 설계

이 장에서는 데이터베이스 설계 과정 중 다음과 같은 단계에 대해 설명합니다.

- 70 페이지의 『데이터베이스에 기록할 데이터 결정』
- 71 페이지의 『관계의 각 유형에 대한 테이블 정의』
- 74 페이지의 『모든 테이블에 대한 컬럼 정의 제공』
- 77 페이지의 『하나 이상의 컬럼을 기본 키로 식별』
- 80 페이지의 『같은 값이 동일한 엔터티를 표현하는지 확인』
- 81 페이지의 『테이블 정규화 고려』
- 87 페이지의 『제한조건 강제 계획』
- 95 페이지의 『기타 데이터베이스 설계 고려사항』

데이터베이스 설계의 목적은 사용자의 환경을 이해하기 쉽고 확장시 기준이 될 수 있는 환경으로 만드는 것입니다. 또한, 사용자가 데이터의 일관성과 무결성을 유지 보수하는 데 도움이 되도록 설계하는 것입니다. 이는 불필요한 부분을 줄이고 데이터베이스의 갱신중에 발생할 수 있는 예외사항을 제거함으로써 실현될 수 있습니다.

이들 단계는 *논리* 데이터베이스 설계의 일부입니다. 데이터베이스 설계는 선형적으로 발생하는 프로세스가 아닙니다. 설계를 마친 뒤에도 다시 반복해야 할 경우가 있습니다.

데이터베이스 설계의 *물리적* 구현은 97 페이지의 『제8장 물리적 데이터베이스 설계』 및 *관리 안내서: 구현*의 "사용자 설계 구현"에 설명되어 있습니다.

데이터베이스에 기록할 데이터 결정

데이터베이스 설계 개발 과정의 첫 번째 단계는 데이터베이스 테이블에 저장될 데이터의 유형을 식별하는 것입니다. 데이터베이스에는 구조 또는 비지니스상의 *엔터티*와 엔터티간의 관계에 대한 정보가 들어 있습니다. 관계형 데이터베이스에서, *엔터티*는 *테이블*로 나타냅니다.

*엔터티*는 정보를 저장할 대상이 되는 사용자, 오브젝트 또는 개념입니다. 샘플 테이블에 설명된 일부 엔터티는 사원, 부서 및 프로젝트입니다. (샘플 데이터베이스에 대해서는 *SQL 참조서*에서 자세한 내용을 참조하십시오.)

샘플 사원 테이블에서, 엔터티 "사원"에는 사원 번호, 이름, 작업 부서 및 임금과 같은 *속성* 또는 특성이 있습니다. 이러한 특성은 *컬럼* EMPNO, FIRSTNME, LASTNAME, WORKDEPT 및 SALARY로 표시됩니다.

엔터티 "사원"의 *발생*은 사원 한 명에 대한 모든 컬럼의 값으로 구성됩니다. 각 사원에게는 엔터티 "사원"의 발생을 식별할 수 고유 사원 번호(EMPNO)가 있습니다. 테이블에서의 각 행은 엔터티 또는 관계의 발생을 나타냅니다. 예를 들어, 다음 표에서 첫 번째 행의 값은 Haas라는 사원에 대한 것입니다.

표 4. 사원 엔터티의 발생 및 속성

EMPNO	FIRSTNME	LASTNAME	WORKDEPT	JOB
000010	Christine	Haas	A00	President
000020	Michael	Thompson	B01	Manager
000120	Sean	O'Connell	A00	Clerk
000130	Dolores	Quintana	C01	Analyst
000030	Sally	Kwan	C01	Manager
000140	Heather	Nicholls	C01	Analyst
000170	Masatoshi	Yoshimura	D11	Designer

멀티미디어와 같은 기존에 사용되지 않던 데이터베이스 응용프로그램도 추가로 지원해야 합니다. 문서, 비디오 또는 혼합된 미디어, 영상 및 음성과 같은 멀티미디어 오브젝트를 지원하기 위한 속성도 고려해야 합니다.

테이블에서, 행의 각 컬럼은 해당 행의 다른 모든 컬럼과 어떠한 방식으로든 관련되어 있습니다. 다음은 샘플 테이블에 나타난 관계 중 일부입니다.

- 사원은 부서에 할당되어 있습니다.
  - Dolores Quintana는 부서 C01에 할당되어 있습니다.
- 사원은 작업을 수행합니다.
  - Dolores는 분석가로서 일합니다.
- 부서는 다른 부서에 보고합니다.
  - 부서 C01은 부서 A00에 보고합니다.
  - 부서 B01은 부서 A00에 보고합니다.
- 사원은 프로젝트를 수행중입니다.
  - Dolores와 Heather는 둘다 프로젝트 IF1000을 수행중입니다.
- 사원은 부서를 관리합니다.
  - Sally는 부서 C01을 관리합니다.

″사원″ 및 ″부서″는 엔터티이고, Sally Kwan은 ″사원″ 발생의 부분이며, C01은 ″부서″ 발생의 부분입니다. 테이블의 각 행에 있는 동일한 컬럼에는 동일한 관계가 적용됩니다. 예를 들어, 테이블의 하나의 행은 Sally Kwan이 C01 부서를 관리한다는 관계를 나타내고, 또다른 행은 Sean O'Connell이 A00 부서의 직원이라는 관계를 나타냅니다.

테이블에 들어 있는 정보는 표현될 관계, 요구되는 융통성의 정도 및 원하는 데이터 검색 속도에 종속됩니다.

엔터프라이즈 내에서 엔터티 관계를 식별하는 것 외에도, 해당 데이터에 적용할 비지니스 규칙과 같은 다른 유형의 정보도 식별해야 합니다.

관계의 각 유형에 대한 테이블 정의

관계의 몇 가지 유형은 데이터베이스에서 정의될 수 있습니다. 사원과 부서간의 가능한 관계를 고려해 보십시오. 사원은 한 부서에서만 일할 수 있으므로, 이 관계는 사원에 대해 *단일 값*입니다. 반대로, 한 부서에서는 여러 사원이 일할 수 있으므로, 이 관계는 부서에 대해 *복수 값*입니다. 사원(단일 값)과 부서(복수 값)간의 관계는 *일대다* 관계입니다. 다음과 같은 관계 유형이 이 절에서 설명됩니다.

- 72 페이지의『일대다 및 다대일 관계』

- 73 페이지의『다대다 관계』
- 73 페이지의『일대일 관계』

일대다 및 다대일 관계

일대다 및 다대일 관계 각각에 대한 테이블을 정의하려면, 다음을 수행하십시오.

1. 관계의 다수(many)측이 동일한 엔터티가 되도록 모든 관계를 그룹화하십시오.
2. 그룹에 속한 모든 관계에 대해 단일 테이블을 정의하십시오.

다음 예에서, 첫 번째 및 두 번째 관계의 "다수"측이 "사원"이므로 사원 테이블 EMPLOYEE를 정의할 수 있습니다.

표 5. 다대일 관계

엔터티	관계	엔터티
사원	할당됨	부서에
사원	일함	작업에서
부서	보고함	(관리) 부서에

세 번째 관계에서 "부서"는 "다수"측이므로 부서 테이블 DEPARTMENT를 정의할 수 있습니다.

다음은 이 관계가 테이블에서 표시되는 방법을 보여줍니다.

EMPLOYEE 테이블

EMPNO	WORKDEPT	JOB
000010	A00	President
000020	B01	Manager
000120	A00	Clerk
000130	C01	Analyst
000030	C01	Manager
000140	C01	Analyst
000170	D11	Designer

DEPARTMENT 테이블

DEPTNO	ADMRDEPT
C01	A00
D01	A00
D11	D01

다대다 관계

양방향으로 복수 값인 관계를 다대다 관계라고 합니다. 사원이 두 개 이상의 프로젝트에서 일할 수 있고, 프로젝트가 두 명 이상의 사원에 할당될 수 있습니다. ″Dolores Quintana가 무슨 일을 하고 있는가?″ 및 ″누가 프로젝트 IF1000을 수행하는가?″라는 두 질문의 답은 여러 개가 될 수 있습니다. 다대다 관계는 다음 예에서와 같이 각 엔터티(″사원″ 및 ″프로젝트″)에 대한 컬럼으로 테이블에 표시될 수 있습니다.

다음 테이블은 다대다 관계(한 사원이 여러 프로젝트를 수행할 수 있고, 한 프로젝트가 여러 사원에게 할당될 수 있음)가 어떻게 표현되는지를 나타냅니다.

사원 활동(EMP_ACT) 테이블

EMPNO	PROJNO
000030	IF1000
000030	IF2000
000130	IF1000
000140	IF2000
000250	AD3112

일대일 관계

일대일 관계는 양방향으로 단일 값인 관계를 말합니다. 관리자는 하나의 부서를 관리하고, 부서에는 한 명의 관리자만 있습니다. ″부서 C01의 관리자는 누구인가?″ 및 ″Sally Kwan은 어느 부서를 관리하는가?″라는 두 질문의 답은 각각 하나씩입니다. 관계는 DEPARTMENT 테이블 또는 EMPLOYEE 테이블 중 하나에 지

정될 수 있습니다. 모든 부서에는 관리자가 있지만 모든 사원이 관리자는 아니므로, 관리자를 다음 예에서처럼 DEPARTMENT 테이블에 추가하는 것이 가장 논리적입니다.

다음은 일대일 관계가 테이블에서 어떻게 표시되는지를 나타냅니다.

DEPARTMENT 테이블

DEPTNO	MGRNO
A00	000010
B01	000020
D11	000060

모든 테이블에 대한 컬럼 정의 제공

관계 테이블에서 컬럼을 정의하려면, 다음과 같이 하십시오.

1. 컬럼의 이름을 선택하십시오.

   테이블의 각 컬럼은 테이블내에서 고유한 이름을 갖습니다. 컬럼 이름을 선택하는 것은 215 페이지의 『부록A. 이름 지정 규칙』에서 자세하게 다룹니다.

2. 어떤 종류의 데이터가 컬럼에 대해 유효한지를 정하십시오.

   데이터 유형 및 *길이*는 데이터 유형 및 컬럼에 대한 유효한 최대 길이를 지정합니다. 데이터 유형은 데이터베이스 관리 프로그램에 의해 제공된 유형 중에서 선택하거나 사용자 정의 유형을 작성하여 선택할 수도 있습니다. DB2에서 제공하는 데이터 유형 및 사용자 정의 유형에 대해서는 *SQL 참조서*에서 자세한 내용을 참조하십시오.

   데이터 유형 범위의 예로는 숫자, 문자열, 2바이트(또는 그래픽) 문자열, 날짜 및 시간, 2진 문자열이 있습니다.

   대형 오브젝트(LOB) 데이터 유형은 문서, 비디오, 영상 및 음성과 같은 멀티미디어 오브젝트를 지원합니다. 이들 오브젝트는 다음과 같은 데이터 유형으로 구현됩니다.

   - *2진 대형 오브젝트*(BLOB) 문자열 BLOB의 예로는 사원의 사진, 음성 및 비디오가 있습니다.

- *문자 대형 오브젝트*(CLOB) 문자열에서는 문자 순서가 1바이트, 멀티 바이트 또는 둘의 결합일 수 있습니다. CLOB의 예로는 사원의 이력서가 있습니다.
- *2바이트 문자 대형 오브젝트*(DBCLOB) 문자열. 여기서 연속 문자들은 2바이트 문자입니다. DBCLOB의 예로는 일본어 신상 명세서가 있습니다.

대형 오브젝트 지원에 대해서는 *SQL 참조서*에서 자세한 내용을 참조하십시오.

사용자 정의 유형(UDT)은 기존의 유형에서 파생된 유형입니다. 기존의 유형에서 파생되며 기존 유형과 특성을 공유하지만, 그럼에도 불구하고 개별적이고 호환 불가능한 유형을 정의할 때 필요합니다.

*구조화 유형*은 데이터베이스에 정의된 구조가 있는 사용자 정의 유형입니다. 이 유형에는 각각 데이터 유형을 가지는 *속성*이라는 시퀀스가 포함됩니다. 구조화 유형은 *상위 유형*이라는 다른 구조화 유형의 *부속 유형*으로서 정의될 수 있습니다. 부속 유형은 해당 상위 유형의 모든 속성을 계승하며, 부속 유형에 대한 추가 속성을 정의할 수 있습니다. 공통된 상위 유형과 연관된 구조화 유형 세트를 *유형 계층*이라 하며, 상위 유형이 없는 상위 유형을 유형 계층의 *루트 유형*이라 합니다.

구조화 유형을 테이블 또는 뷰의 유형으로 사용할 수 있습니다. 오브젝트 식별자와 함께 구조화 유형 속성의 이름 및 데이터 유형이 해당 *입력된 테이블* 또는 *입력된 뷰* 컬럼의 이름 및 데이터 유형이 됩니다. 입력된 테이블 또는 입력된 뷰의 행을 구조화 유형의 인스턴스 표현이라고 간주하면 됩니다.

구조화 유형을 테이블 또는 뷰 컬럼의 데이터 유형으로 사용할 수 없습니다. 또한, 전체 구조화 유형 인스턴스에서 응용프로그램의 호스트 변수를 검색하는 기능도 지원되지 않습니다.

*참조 유형*은 구조화 유형과 동등한 유형입니다. 구별 유형과 유사하게, 참조 유형은 내장된 데이터 유형 중 하나와 공통 표현을 공유하는 스칼라 유형입니다. 유형 계층의 모든 유형이 이러한 동일한 표현을 공유합니다. 참조 유형 표현은 유형 계층의 루트 유형이 작성될 때 정의됩니다. 참조 유형을 사용할 때, 구조화 유형이 유형 매개변수로서 지정됩니다. 이러한 매개변수를 참조의 *목표 유형*이라 부릅니다.

참조 목표는 항상 입력된 테이블 또는 뷰의 행입니다. 참조 유형이 사용된 경우, 해당 참조 유형에는 *범위*가 정의되어 있습니다. 범위는 참조 값의 목표 행이 들어 있는 테이블(*목표 테이블*이라 함) 또는 뷰(*목표 뷰*라 함)를 나타냅니다. 목표 테이블 또는 뷰는 참조 유형의 목표 유형과 동일한 유형이어야 합니다. 범위가 지정된 참조 유형의 인스턴스는 *목표 행*이라는 입력된 테이블 또는 유형 뷰의 행을 고유하게 식별합니다.

사용자 정의 함수(UDF)는 사용자 정의 유형간의 비교 또는 변환 작업을 위해 루틴을 호출하는 것 외에도 여러 가지 이유로 사용될 수 있습니다. UDF는 SQL 내장 함수에 의해 제공되는 지원 기능에 확장 및 추가되어 내장 함수가 사용되는 곳이면 어디든지 사용될 수 있습니다. 다음은 UDF의 두 가지 유형입니다.

- 외부 함수 : 프로그래밍 언어로 작성됨
- 원시 함수 : 다른 UDF를 호출하는 데 사용됨

예를 들어, 두 개의 수치 데이터 유형이 유럽형 구두 *크기*와 미국형 구두 *크기*인 경우를 가정하십시오. 두 유형 모두 구두 *크기*를 표현하지만, 측정 기준이 달라 서로 비교할 수 없기 때문에 호환될 수 없습니다. 사용자 정의 함수가 호출되어 하나의 구두 *크기*를 다른 *크기*로 변환시킬 수 있습니다.

사용자 정의 유형, 구조화 유형, 참조 유형 및 사용자 정의 함수에 대해서는 *SQL 참조서*에서 자세한 내용을 참조하십시오.

3. 기본값이 필요한 컬럼을 정하십시오.

   일부 컬럼은 다음과 같은 이유로 모든 행에 의미 있는 값을 가질 수 없습니다.

   - 컬럼 값은 행에 적용될 수 없습니다.

     예를 들어, 사원의 중간 이름이 들어 있는 컬럼은 중간 이름이 없는 사원에게는 적용될 수 없습니다.

   - 값을 적용할 수는 있지만, 아직 알려지지 않은 값입니다.

     예를 들어, MGRNO 컬럼에는 유효한 관리자 번호가 들어 있지 않을 수도 있는데, 이는 부서의 이전 관리자는 전출되었으나 새로운 관리자가 아직 결정되지 않았기 때문입니다.

두 경우 모두, 널(NULL) 값(컬럼 값을 알 수 없거나 적용할 수 없음을 나타내는 특수 값)을 허용하는 방안과 데이터베이스 관리 프로그램 또는 응용프로그램이 널(NULL) 값이 아닌 기본값을 할당하도록 하는 방안 중 하나를 선택할 수 있습니다.

널(NULL) 값 및 기본값에 대해서는 *SQL 참조서*에서 자세한 내용을 참조하십시오.

하나 이상의 컬럼을 기본 키로 식별

*키*는 특정 행(들)을 식별하거나 액세스하기 위해 사용될 수 있는 컬럼 세트입니다. 키는 테이블, 색인 또는 참조 제한조건의 설명으로 식별됩니다. 동일한 컬럼은 둘 이상의 키의 일부일 수 있습니다.

*고유 키*는 두 개의 관련 값이 동일하지 않도록 제한되는 키입니다. 고유 키의 컬럼은 널(NULL) 값을 포함할 수 없습니다. 예를 들어, 사원 번호 컬럼이 고유 키로 정의될 수 있는데, 이는 컬럼의 각 값이 한 명의 사원만을 식별하기 때문입니다. 두 명의 사원이 동일한 사원 번호를 가질 수는 없습니다.

키의 고유성을 부여하는데 사용되는 메카니즘을 *고유 색인*이라고 합니다. 고유 색인은 각 값이 고유한 행을 식별하는(기능적으로 판별) 컬럼이나 컬럼의 순서화된 콜렉션입니다. 고유 색인은 널(NULL) 값을 포함할 수 있습니다.

*기본 키*는 테이블에서 정의된 고유 키 중의 하나이지만, 첫 번째로 중요한 키로 선택됩니다. 하나의 테이블에는 기본 키가 하나 있습니다.

기본 키에 대해 자동으로 *1차 색인*이 작성됩니다. 데이터베이스 관리 프로그램은 1차 색인을 사용하여 테이블 행에 효과적으로 액세스하고, 기본 키의 고유성을 강력하게 시행할 수 있습니다. 또한 조회를 처리할 때 효과적으로 데이터에 액세스하도록 기본 키가 아닌 컬럼에서 색인을 정의할 수 있습니다.

테이블에 "자연스러운" 고유 키가 없는 경우 또는 도착 시간순으로 고유 행을 구별하는 경우, 키의 일부로서 시간소인을 사용하면 도움이 됩니다(79 페이지의 『식별 컬럼 정의』도 참조).

일부 샘플 테이블에 대한 기본 키는 다음과 같습니다.

테이블	키 컬럼
사원 테이블	EMPNO
부서 테이블	DEPTNO
프로젝트 테이블	PROJNO

다음 예는 기본 키 컬럼이 포함된 PROJECT 테이블의 일부를 보여줍니다.

표 6. PROJECT 테이블의 기본 키

PROJNO(기본 키)	**PROJNAME**	**DEPTNO**
MA2100	Weld Line Automation	D01
MA2110	Weld Line Programming	D11

테이블의 모든 컬럼에 중복된 값이 들어 있으면, 기본 키를 하나의 컬럼만으로 정의할 수 없습니다. 두 개 이상의 컬럼으로 된 키를 *복합 키*라고 합니다. 컬럼 값의 조합으로 고유 엔터티를 정의해야 합니다. 복합 키를 쉽게 정의할 수 없으면, 고유 값을 갖는 새로운 컬럼을 작성하는 것을 고려해 보아야 합니다.

다음 예에서는 두 개 이상의 컬럼을 포함하는 기본 키(복합 키)를 보여줍니다.

표 7. EMP_ACT 테이블의 복합 기본 키

EMPNO (기본 키)	**PROJNO (기본 키)**	**ACTNO (기본 키)**	**EMPTIME**	**EMSTDATE (기본 키)**
000250	AD3112	60	1.0	1982-01-01
000250	AD3112	60	.5	1982-02-01
000250	AD3112	70	.5	1982-02-01

후보 키 컬럼 식별

후보 키를 식별하려면, 고유 엔터티를 정의하는 컬럼의 최소 번호를 선택하십시오. 둘 이상의 후보 키가 있을 수도 있습니다. 25 페이지의 표2에는 여러 개의 후보 키가 있습니다. EMPNO, PHONENO 및 LASTNAME 컬럼은 각각 고유하게 사원을 식별합니다.

후보 키 풀에서 기본 키를 선택할 때의 기준은 지속성, 고유성 및 안정성입니다.

- 지속성은 각 행의 기본 키 값이 항상 존재함을 의미합니다.
- 고유성은 각 행의 키 값이 다른 모든 것과는 다름을 의미합니다.

- 안정성은 기본 키 값이 결코 변경되지 않음을 의미합니다.

예의 세 가지 후보 키 중에서 EMPNO만이 이 모든 기준을 충족시킵니다. 사원이 회사에 입사할 당시에 전화번호를 가지고 있지 않을 수도 있습니다. 성은 변경될 수 있으며, 한때 고유했을지라도 보장할 수는 없습니다. 사원 번호 컬럼을 기본 키로 선택하는 것이 가장 바람직합니다. 사원은 일단 고유 번호를 할당받으면 회사에 계속 남아 있는 한 일반적으로 이 번호는 갱신되지 않습니다. 각 사원은 각기 사원 번호를 가지고 있어야 하므로 사원 번호 컬럼의 값은 지속적입니다.

식별 컬럼 정의

*식별 컬럼*은 테이블의 각 행에 대한 고유한 숫자 값을 자동으로 생성하는 DB2에 대한 방법을 제공합니다. 테이블은 식별 속성으로 정의된 단일 컬럼을 가질 수 있습니다. 식별 컬럼의 예는 주문 번호, 사원 번호, 재고 번호 및 변환 기록 번호를 포함합니다.

식별 컬럼 값은 항상 또는 기본값으로 생성될 수 있습니다.

- *항상 생성*으로 정의되는 식별 컬럼은 DB2에 의해 고유성이 보장됩니다. 관련 값은 DB2에 의해 항상 생성되며, 응용프로그램은 명시적 값을 제공하도록 허용되지 않습니다.
- *기본값으로 생성*으로 정의된 식별 컬럼은 식별 컬럼에 대한 값을 명시적으로 제공하는 방법을 응용프로그램에 제공합니다. 값이 제공되지 않는다면, DB2가 값을 생성하지만, 이 경우 값의 고유성을 보장할 수 없습니다. DB2는 생성하는 값 세트에 대해서만 고유성을 보장합니다. 기본값으로 생성은 기본 테이블의 내용을 복사하거나 테이블을 언로드하고 재로드하는 데이터 전파에 사용됨을 의미합니다.

식별 컬럼은 고유한 기본 키 값을 생성하는 타스크에 아주 적합합니다. 응용프로그램은 식별 컬럼을 사용하여 응용프로그램이 데이터베이스의 외부에서 자체 고유한 카운터를 생성할 때 초래할 수 있는 동시성 및 성능 문제점을 피할 수 있습니다. 예를 들어, 하나의 공통 응용프로그램 레벨 구현은 카운터가 들어 있는 1행 테이블을 유지보수하기 위한 것입니다. 각 트랜잭션은 이 테이블을 잠그며, 숫자를 증가시킨 후, 확약합니다. 즉, 한번에 오직 하나의 트랜잭션만이 카운터를 증가시킬 수 있습니다. 반면, 카운터가 식별 컬럼을 통해 유지보수되는 경우, 카운터가 트랜

잭션에 의해 잠겨지지 않으므로 동시성의 레벨이 더 높아질 수 있습니다. 카운터를 증가시킨 하나의 미확약 트랜잭션은 카운터를 증가시키지 못하도록 차후 트랜잭션을 방지하지 않습니다.

식별 컬럼의 카운터는 트랜잭션과 별개로 증가됩니다. (또는 감소됩니다.) 주어진 트랜잭션이 식별 카운터를 두 번 증가시키는 경우, 동일한 식별 카운터를 동시에 증가시키는 다른 트랜잭션이 있을 수 있으므로(즉, 행을 동일한 테이블에 추가) 생성되는 두 개의 숫자에 갭이 있습니다. 응용프로그램이 연속적인 숫자 범위를 가져야 하는 경우, 응용프로그램은 식별 컬럼을 가진 테이블에서 독점 잠금을 취해야 합니다. 이 결정은 초래되는 동시성의 손실에 대해 숙고되어야 합니다. 더우기, 주어진 식별 컬럼은 숫자에서 갭을 생성할 수 있는데, 그 이유는 식별 컬럼의 값을 생성한 트랜잭션이 구간 복원되거나, 캐쉬된 모든 값이 지정되기 전에 값의 범위를 캐쉬한 데이터베이스가 비활성화되기 때문입니다.

식별 컬럼으로 생성되는 순차 번호는 다음과 같은 추가 등록 정보를 갖습니다.

- 값은 0의 스케일을 가진 완전한 숫자 데이터 유형일 수 있습니다. 즉 0의 스케일을 가진 SMALLINT, INTEGER, BIGINT 또는 DECIMAL이 있습니다. (단일 및 배정밀도 부동 소수점은 근사한 숫자 데이터 유형으로 간주됩니다.)
- 연속 값은 지정된 정수 증분으로 차이가 납니다. 기본 증분은 1입니다.
- 식별 컬럼의 카운터 값은 회복가능합니다. 실패한 경우, 카운터 값은 로그에서 재구성되므로, 고유 값 생성이 계속됩니다.
- 식별 컬럼 값은 더 나은 성능을 제공하기 위해 캐쉬될 수 있습니다.

같은 값이 동일한 엔터티를 표현하는지 확인

동일한 세트의 엔터티에 대한 속성을 묘사하는 테이블을 두 개 이상 가질 수 있습니다. 예를 들어, EMPLOYEE 테이블에서는 사원이 지정된 부서의 번호를 보여주고, DEPARTMENT 테이블에서는 각 부서 번호에 지정된 관리자를 보여줍니다. 두 세트의 속성을 동시에 검색하기 위해, 다음 예에서처럼 일치하는 컬럼에서 두 테이블을 조인할 수 있습니다. WORKDEPT 및 DEPTNO의 값은 동일한 엔터티를 나타내며, DEPARTMENT 및 EMPLOYEE 테이블간의 *조인 경로*를 나타냅니다.

DEPARTMENT 테이블

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT
D21	관리 지원	000070	D01

EMPLOYEE 테이블

EMPNO	FIRSTNAME	LASTNAME	WORKDEPT	JOB
000250	Daniel	Smith	D21	Clerk

두 개 이상의 테이블에서 하나의 엔터티에 대한 정보를 검색할 때, 같은 값이 동일한 엔터티를 나타내는지 확인하십시오. 연결되는 컬럼에 다른 이름이 있거나(이전 예에서의 WORKDEPT 및 DEPTNO처럼) 동일한 이름이 있을 수 있습니다(부서 및 프로젝트 테이블에서의 DEPTNO 컬럼처럼).

테이블 정규화 고려

*정규화*는 테이블 테이터의 중복과 불일치를 줄입니다. 정규화는 키가 아닌 모든 컬럼이 기본 키 컬럼에 따라 달라지는 컬럼 세트로 테이블을 축소시키는 프로세스입니다. 그렇지 않은 경우에는 갱신하는 동안에 데이터가 일치하지 않을 수도 있습니다.

이 절에서는 첫 번째, 두 번째, 세 번째 및 네 번째 정규 양식에 대한 규칙을 간단히 검토합니다. 데이터베이스 설계에 관한 여러 책에서 다루는 테이블의 5번째 정규 양식에 대해서는 다루지 않습니다.

양식 설명

첫 번째

테이블의 각 행 및 컬럼 위치에 여러 개가 아닌 *하나의* 값만 존재합니다(82 페이지의 『첫 번째 정규 양식』 참조).

두 번째

키의 일부가 아닌 각 컬럼은 키에 따라 좌우됩니다(82 페이지의 『두 번째 정규 양식』 참조).

세 번째

키가 아닌 각 컬럼은 다른 키가 아닌 컬럼과 독립적이며, 키에 대해서만 종속됩니다(84 페이지의 『세 번째 정규 양식』 참조).

네 번째

어느 행에도 엔터티에 대한 두 개 이상의 독립적인 복수 값 사실이 들어 있지 않습니다(86 페이지의 『네 번째 정규 양식』 참조).

첫 번째 정규 양식

각 셀에 값 세트가 아닌 오직 하나의 값이 있는 경우, 테이블은 *첫 번째 정규 양식*으로 되어 있습니다. 첫 번째 정규 양식으로 된 테이블은 상위 정규 양식에 대한 기준을 반드시 충족시키지 않습니다.

예를 들어, 다음 테이블은 WAREHOUSE 컬럼에 각각의 PART 발생에 대한 값이 여러 개 들어 있기 때문에 첫 번째 정규 양식에 위배됩니다.

표 8. 첫 번째 정규 양식에 위배되는 테이블

PART(기본 키)	**WAREHOUSE**
P0010	Warehouse A, Warehouse B, Warehouse C
P0020	Warehouse B, Warehouse D

다음 예에서는 첫 번째 정규 양식으로 된 테이블을 보여줍니다.

표 9. 첫 번째 정규 양식을 따르는 테이블

PART(기본 키)	**WAREHOUSE(기본 키)**	**QUANTITY**
P0010	Warehouse A	400
P0010	Warehouse B	543
P0010	Warehouse C	329
P0020	Warehouse B	200
P0020	Warehouse D	278

두 번째 정규 양식

키의 일부가 아닌 각 컬럼이 *전체* 키에 종속되는 경우 테이블은 *두 번째 정규 양식*으로 되어 있습니다.

다음의 예에서처럼 키가 아닌 컬럼이 복합 키의 부품에 종속될 때 두 번째 정규 양식에 위배됩니다.

표 10. 두 번째 정규 양식에 위배되는 테이블

PART(기본 키)	**WAREHOUSE (기본 키)**	**QUANTITY**	**WAREHOUSE_ADDRESS**
P0010	Warehouse A	400	1608 New Field Road
P0010	Warehouse B	543	4141 Greenway Drive
P0010	Warehouse C	329	171 Pine Lane
P0020	Warehouse B	200	4141 Greenway Drive
P0020	Warehouse D	278	800 Massey Street

기본 키는 PART 및 WAREHOUSE 컬럼이 함께 구성된 복합 키입니다. WAREHOUSE_ADDRESS 컬럼은 WAREHOUSE의 값에만 종속적이기 때문에, 이 테이블은 두 번째 정규 양식의 규칙에 위배됩니다.

이 설계의 문제점은 다음과 같습니다.

- 웨어하우스 주소는 해당 웨어하우스에 저장된 부품이 나오는 모든 레코드마다 반복됩니다.
- 웨어하우스의 주소가 변경되면, 웨어하우스에 저장된 부품에 관한 모든 행도 갱신되어야 합니다.
- 중복되는 데이터로 인해, 동일한 웨어하우스의 주소가 레코드마다 서로 다르게 표시되어 데이터가 일치하지 않을 수도 있습니다.
- 웨어하우스에 저장된 부품이 없을 때에는 웨어하우스 주소를 기록하는 행이 없을 수 있습니다.

솔루션은 다음과 같은 두 개의 테이블로 테이블을 나누는 것입니다.

표 11. 두 번째 정규 양식을 따르는 PART_STOCK 테이블

PART(기본 키)	**WAREHOUSE(기본 키)**	**QUANTITY**
P0010	Warehouse A	400
P0010	Warehouse B	543
P0010	Warehouse C	329
P0020	Warehouse B	200

표 11. 두 번째 정규 양식을 따르는 PART_STOCK 테이블 (계속)

PART(기본 키)	WAREHOUSE(기본 키)	QUANTITY
P0020	Warehouse D	278

표 12. 두 번째 정규 양식을 따르는 WAREHOUSE 테이블

WAREHOUSE(기본 키)	WAREHOUSE_ADDRESS
Warehouse A	1608 New Field Road
Warehouse B	4141 Greenway Drive
Warehouse C	171 Pine Lane
Warehouse D	800 Massey Street

두 번째 정규 양식으로 된 두 테이블을 만들 때에는 성능상 고려해야 할 점이 있습니다. 부품의 위치에 관한 보고서를 작성하는 응용프로그램에서는 관련 정보를 검색하기 위해 두 테이블을 모두 조인해야 합니다.

성능 고려사항에 대해서는 *관리 안내서: 성능*의 "응용프로그램 성능 조정"에서 자세한 내용을 참조하십시오.

세 번째 정규 양식

키가 아닌 각 컬럼이 키가 아닌 다른 컬럼에는 독립적이면서 해당 키에만 종속적인 경우, 이 테이블은 세 번째 정규 양식으로 된 것입니다.

다음 예에서 첫 번째 테이블에는 컬럼 EMPNO 및 WORKDEPT가 들어 있습니다. 여기에 컬럼 DEPTNAME이 추가된다고 가정해 보십시오(85 페이지의 표14 참조). 새 컬럼은 WORKDEPT에 좌우되지만, 기본 키는 EMPNO입니다. 테이블은 이제 세 번째 정규 양식에 위배됩니다. 한 명의 사원, 즉 John Parker에 대한 DEPTNAME을 변경한다고 해서 해당 부서에 있는 다른 사원의 부서 이름이 변경되지는 않습니다. 이제 부서 번호 E11에 서로 다른 두 개의 부서 이름이 사용된다는 점에 유의하십시오. 그 결과 생기는 불일치가 테이블의 갱신된 버전에 표시됩니다.

표 13. 갱신 이전의 정규화되지 않은 EMPLOYEE_DEPARTMENT 테이블

EMPNO (기본 키)	**FIRSTNAME**	**LASTNAME**	**WORKDEPT**	**DEPTNAME**
000290	John	Parker	E11	Operations
000320	Ramlal	Mehta	E21	Software Support
000310	Maude	Setright	E11	Operations

표 14. 갱신 이후의 정규화되지 않은 EMPLOYEE_DEPARTMENT 테이블. 테이블의 정보가 서로 일치하지 않음.

EMPNO (기본 키)	**FIRSTNAME**	**LASTNAME**	**WORKDEPT**	**DEPTNAME**
000290	John	Parker	E11	Installation Mgmt
000320	Ramlal	Mehta	E21	Software Support
000310	Maude	Setright	E11	Operations

이 테이블은 WORKDEPT 및 DEPTNAME 컬럼을 추가한 새로운 테이블을 작성해서 정규화될 수 있습니다. 부서 이름을 변경하는 것과 같은 갱신은 이제 훨씬 더 쉬우며, 새로운 테이블만이 갱신될 필요가 있습니다.

사원 이름뿐만 아니라, 부서 이름을 리턴하는 SQL 조회는 두 개의 테이블을 조인해야 하기 때문에 작성하기가 더 복잡합니다. 이 조회는 단일 테이블을 조회하는 것보다 수행 시간이 더 걸릴 수도 있습니다. WORKDEPT 컬럼은 두 테이블에 표시되어야 하므로 더 많은 저장영역 공간이 필요합니다.

다음 테이블은 정규화한 결과로서 정의됩니다.

표 15. EMPLOYEE_DEPARTMENT 테이블을 정규화한 후의 EMPLOYEE 테이블

EMPNO(기본 키)	**FIRSTNAME**	**LASTNAME**	**WORKDEPT**
000290	John	Parker	E11
000320	Ramlal	Mehta	E21
000310	Maude	Setright	E11

표 16. EMPLOYEE_DEPARTMENT 테이블을 정규화한 후의 DEPARTMENT 테이블

DEPTNO(기본 키)	**DEPTNAME**
E11	Operations
E21	Software Support

네 번째 정규 양식

엔터티에 대한 독립적인 복수 값 사실이 두 개 이상 들어 있는 행이 없으면 이 테이블은 네 번째 정규 양식입니다.

사원, 기술 및 언어와 같은 엔터티를 고려하십시오. 사원이 여러 개의 기술을 가지고 있고 여러 가지 언어를 알고 있을 수 있습니다. 여기에는 두 개의 관계, 즉 사원과 기술간의 관계와 사원과 언어간의 관계가 있습니다. 테이블은 다음 예에서처럼 양쪽 관계를 나타내면, 네 번째 정규 양식으로 되어 있지 않습니다.

표 17. 네 번째 정규 양식에 위배되는 테이블

EMPNO(기본 키)	**SKILL(기본 키)**	**LANGUAGE(기본 키)**
000130	데이터 모델링	영어
000130	데이터베이스 설계	영어
000130	응용프로그램 설계	영어
000130	데이터 모델링	스페인어
000130	데이터베이스 설계	스페인어
000130	응용프로그램 설계	스페인어

대신, 관계가 두 개의 테이블로 표시되어야 합니다.

표 18. 네 번째 정규 양식에 따르는 EMPLOYEE_SKILL

EMPNO(기본 키)	**SKILL(기본 키)**
000130	데이터 모델링
000130	데이터베이스 설계
000130	응용프로그램 설계

표 19. 네 번째 정규 양식에 따르는 EMPLOYEE_LANGUAGE

EMPNO(기본 키)	**LANGUAGE(기본 키)**
000130	영어
000130	스페인어

그러나, 속성이 상호 종속적이라면(즉, 사원이 특정 언어를 특정 기술에만 적용시키는 경우), 테이블은 파티션되지 *않아야* 합니다.

데이터베이스 설계시 네 번째 정규 양식으로 된 테이블에 모든 데이터를 배치하는 것이 좋으며, 그 결과가 성능 레벨에 적합한지 판별하십시오. 적합하지 않다면, 세 번째 정규 양식으로 테이블에 있는 데이터를 재배열한 후, 성능을 재평가할 수 있습니다.

제한조건 강제 계획

*제한조건*은 데이터베이스 관리 프로그램이 강제로 시행하는 규칙입니다. 이 절에서는 네 가지 유형의 제한조건 조절에 대해 설명합니다.

고유 제한조건 테이블에 있는 키 값이 고유한지 확인하십시오. 기본 키를 작성하는 컬럼의 모든 변경사항에 대한 고유성이 점검됩니다.

참조 무결성 삽입, 갱신 및 삭제 조작에 대한 참조 제한조건 강제. 이는 모든 외부 키의 모든 값이 유효한 데이터베이스의 상태를 말합니다.

테이블 점검 제한조건 테이블이 작성되거나 변경될 때 변경된 데이터가 명시된 조건을 위반하지 않음을 확인.

트리거 지정된 테이블에서 갱신, 삭제 또는 삽입 조작이 호출되면 수행되는 조치 세트 정의.

고유 제한조건

*고유 제한조건*은 키 값이 테이블 내에서 고유함을 보장하는 규칙입니다. 고유 제한조건에서 키를 구성하는 각 컬럼은 NOT NULL로 정의되어야 합니다. 고유 제

한조건은 PRIMARY KEY절이나 UNIQUE절을 사용하여 CREATE TABLE문 또는 ALTER TABLE문에서 정의됩니다.

한 테이블의 고유 제한조건 수에는 제한이 없지만, 하나의 고유 제한조건만을 테이블의 기본 키로 정의할 수 있습니다. 또한, 테이블은 동일한 컬럼 세트에 둘 이상의 고유 제한조건을 가질 수 없습니다.

고유 제한조건이 정의되면, 데이터베이스 관리 프로그램은 고유 색인을 작성하고(필요할 경우), 이를 기본 색인 또는 고유 시스템 요구 색인으로 지정합니다. 제한조건은 고유 색인을 통해 강화됩니다. 일단 컬럼에 고유 색인이 설정되면, 갱신이 끝날 때까지 다중 행 갱신중의 고유성이 유지되는지 점검하십시오.

고유 제한조건은 또한 참조 제한조건의 상위 키로 사용될 수 있습니다.

참조 무결성

데이터베이스 관리 프로그램은 주어진 속성이나 테이블 컬럼의 모든 값이 일부 다른 테이블이나 컬럼에 존재하도록 요구하는 *참조 제한조건*을 통해 참조 무결성을 유지보수합니다. 예를 들어, 참조 제한조건에서는 EMPLOYEE 테이블의 모든 사원이 DEPARTMENT 테이블에 존재하는 부서 내에 있어야 합니다. 존재하지 않는 부서에 사원이 속해 있을 수는 없습니다.

참조 무결성이 유지보수되도록 하고 최적화 알고리즘이 이러한 특수 관계를 이해하여 조회를 보다 효과적으로 처리하도록 하기 위해 데이터베이스에 참조 제한조건을 작성할 수 있습니다. 참조 무결성을 계획할 때, 데이터베이스 테이블간의 모든 관계를 식별하십시오. 기본 키와 참조 제한조건을 정의함으로써 관계를 식별할 수 있습니다.

다음과 같은 관련 테이블을 고려하십시오.

표 20. DEPARTMENT 테이블

DEPTNO(기본 키)	**DEPTNAME**	**MGRNO**
A00	Spiffy Computer Service Div.	000010
B01	Planning	000020
C01	Information Center	000030

표 20. DEPARTMENT 테이블 (계속)

DEPTNO(기본 키)	**DEPTNAME**	**MGRNO**
D11	Manufacturing Systems	000060

표 21. EMPLOYEE 테이블

EMPNO (기본 키)	**FIRSTNAME**	**LASTNAME**	**WORKDEPT (외부 키)**	**PHONENO**
000010	Christine	Haas	A00	3978
000030	Sally	Kwan	C01	4738
000060	Irving	Stern	D11	6423
000120	Sean	O'Connell	A00	2167
000140	Heather	Nicholls	C01	1793
000170	Masatoshi	Yoshimura	D11	2890

참조 무결성을 이해하는 데 유용한 다음과 같은 여러가지 개념이 이 테이블과 관련하여 설명됩니다.

*고유 키*는 행에 있는 어떤 값도 다른 행에서 중복되지 않은 컬럼이나 컬럼 세트입니다. 테이블에 대한 기본 키로서 하나의 고유 키를 정의할 수 있습니다. 고유 키는 또한 외부 키에 의해 참조될 때는 *상위 키*라고도 합니다.

*기본 키*는 테이블 정의의 일부인 고유 키입니다. 각 테이블은 하나의 기본 키만을 가질 수 있습니다. 앞의 테이블에서 DEPTNO와 EMPNO는 DEPARTMENT 및 EMPLOYEE 테이블 각각에 대한 기본 키입니다.

*외부 키*는 동일한 테이블이나 다른 테이블의 고유 키 또는 기본 키를 나타내는 컬럼이거나 테이블의 컬럼 세트입니다. 외부 키는 테이블간의 참조 무결성을 강화하기 위해 고유 키 또는 기본 키와의 관계를 설정하는 데 사용됩니다. EMPLOYEE 테이블의 컬럼 WORKDEPT는 DEPARTMENT 테이블의 기본 키인 DEPNO를 참조하므로 외부 키입니다.

*복합 키*는 둘 이상의 컬럼을 가지는 키입니다. 기본 키 및 외부 키 모두 복합 키가 될 수 있습니다. 예를 들어, 부서가 국 번호와 부서 번호의 조합으로 고유하게 식별되면, DEPARTMENT 테이블의 키를 작성하는 데 두 컬럼이 필요합니다.

*상위 키*는 참조 제한조건의 기본 키 또는 고유 키입니다. *기본 키 제한조건*은 상위 키 컬럼 세트가 지정되지 않은 경우에 참조 제한조건의 기본 상위 키입니다.

*상위 테이블*이란 동일한 테이블 또는 다른 테이블에 있는 하나 이상의 외부 키에 관련된 상위 키를 가지고 있는 테이블입니다. 테이블은 여러 관계에서 상위 테이블이 될 수 있습니다. 예를 들어, DEPTNO의 기본 키를 가지고 있는 DEPARTMENT 테이블이 외부 키 WORKDEPT가 들어 있는 EMPLOYEE 테이블의 상위 테이블이 됩니다.

*상위 행*이란 상위 키 값이 종속 테이블의 하나 이상의 외부 키 값에 일치하는 상위 테이블 행입니다. 상위 테이블의 행이 반드시 상위 행이 되는 것은 아닙니다. DEPARTMENT 테이블의 네 번째 행(D11)은 EMPLOYEE 테이블의 세 번째 및 6번째 행의 상위 행입니다. DEPARTMENT 테이블의 두 번째 행(B01)은 다른 행의 상위 행이 될 수 없습니다.

*종속 테이블*은 하나 이상의 외부 키가 들어 있는 테이블을 말합니다. 종속 테이블도 상위 테이블이 될 수 있습니다. 테이블은 여러 관계에서 종속 테이블일 수 있습니다. EMPLOYEE 테이블에는 외부 키 WORKDEPT가 들어 있고, WORKDEPT는 기본 키를 가지고 있는 DEPARTMENT 테이블에 종속적입니다.

*종속 행*은 상위 키 값에 대응하는 널(NULL) 값이 아닌 외부 키 값을 갖는 종속 테이블 행입니다. 외부 키 값은 종속 행에서 상위 행으로의 참조를 나타냅니다. 외부 키에 널(NULL) 값을 사용할 수 있으므로, 종속 테이블의 행이 반드시 종속 행은 아닙니다.

테이블이 종속 테이블이거나 종속 테이블의 하위이면, 테이블은 테이블의 *하위*가 됩니다. 하위 테이블에는 일부 테이블의 상위 키로 거꾸로 추적될 수 있는 외부 키가 있습니다.

*참조 순환*은 테이블이 자체에 연결되는 경로입니다. 테이블이 직접 자체에 연결될 때, 이를 *자체 참조* 테이블이라고 합니다. EMPLOYEE 테이블에 각 사원의 관리자의 EMPNO가 들어 있는 MGRID라는 또다른 컬럼이 들어 있으면, EMPLOYEE 테이블은 자체 참조 테이블이 됩니다. MGRID는 EMPLOYEE 테이블에 대한 외부 키입니다.

자체 참조 테이블은 동일한 관계에 있는 사위 및 종속 테이블입니다. 자체 참조 행은 자신의 상위 행 및 종속 행이 되는 행입니다. 이러한 상황에서 존재하는 제한조건을 자체 참조 제한조건이라 합니다. 예를 들어, 자체 참조 테이블의 행에 있는 외부 키에 대한 값이 해당 행의 고유 키 값과 일치하면, 행은 자체 참조 행이 됩니다.

*참조 제한조건*은 지정된 외부 키의 널(NULL)이 아닌 값이 고유 키 또는 지정된 테이블 값으로 나타날 경우에만 유효하다는 단정입니다. 참조 제한조건의 목적은 데이터베이스 관계를 유지보수하고 데이터 입력 규칙을 준수하도록 하는 것입니다.

SQL 조작에 대한 포함

참조 제한조건의 시행에는 테이블이 상위 테이블인지 또는 종속 테이블인지에 따라 달라지는 일부 SQL 조작에 대한 특수 포함이 있습니다. 이 섹션에서는 SQL INSERT, DELETE, UPDATE 및 DROP 조작에 대한 유지보수 참조 무결성의 효과에 대해 설명합니다.

DB2는 시스템간의 참조 제한조건을 자동으로 시행하지는 *않습니다*. 결과적으로, 시스템간에 참조 제한조건을 시행하려면 응용프로그램에 필요한 논리가 들어 있어야 합니다.

이 절에는 다음과 같은 내용이 들어 있습니다.

- 『INSERT 규칙』
- 92 페이지의 『DELETE 규칙』
- 93 페이지의 『UPDATE 규칙』

INSERT 규칙: 종속 테이블에 어떠한 조치도 취하지 않고 상위 테이블에 수시로 행을 삽입할 수 있습니다. 그러나 삽입 중인 행의 외부 키 값과 동등한 상위 키 값을 가진 상위 테이블에 행이 있는 경우, 즉 외부 키 값이 널(NULL)이 아니면, 종속 테이블에 하나의 행만을 삽입할 수 있습니다. 값의 구성요소가 널(NULL)일 경우, 복합 외부 키의 값은 널(NULL)이 됩니다.

이 규칙은 외부 키가 정의되는 경우가 내재되어 있습니다.

참조 제한조건을 가진 테이블에 행을 삽입하려고 할 때, 널(NULL)이 아닌 외부 키 값이 상위 키에 나타나지 않으면 INSERT 조작은 허용되지 않습니다. 두 개 이상의 행을 삽입하려는 도중에 INSERT 조작이 하나의 행에 대해 실패하게 되면, 어떤 행도 삽입되지 않습니다.

DELETE 규칙: 상위 테이블에서 행을 삭제할 때, 데이터베이스 관리 프로그램은 종속 테이블에 외부 키 값과 일치하는 임의의 종속 행이 있는지 점검합니다. 종속 행이 발견되면, 여러 조치를 취할 수 있습니다. 사용자는 종속 테이블을 작성할 때 *삭제* 규칙을 지정함으로써, 어떠한 조치가 취해질지를 결정할 수 있습니다.

기본 키가 삭제되는 경우 종속 테이블(외부 키가 들어 있는 테이블)에 대한 삭제 규칙은 다음과 같습니다.

RESTRICT
종속 행이 발견되는 경우에는 상위 테이블의 어떠한 행도 삭제되지 않도록 합니다. 상위 행 및 종속 행 모두를 제거해야 할 경우에는, 종속 행을 먼저 삭제하십시오. 상위 행을 먼저 삭제하는 것은 참조 제한조건에 위배되며 허용되지도 않습니다.

NO ACTION
모든 참조 제한조건이 적용된 후에 모든 하위 행에 대해 상위 행의 존재를 강제하십시오.

CASCADE
상위 테이블의 행을 지우면 자동으로 종속 테이블의 관련 행도 지워집니다. 이 규칙은 종속 테이블의 행이 상위 테이블의 행 없이는 아무런 의미가 없을 때 유용합니다.

상위 행을 먼저 삭제하면 기본 키를 참조하는 종속 행이 자동으로 삭제됩니다. 종속 행을 먼저 삭제하지 않아도 됩니다. 이 종속 행의 일부가 자체 종속 행을 가지고 있으면, 이러한 관계에 대한 삭제 규칙이 적용됩니다. DB2는 연쇄 삭제를 관리합니다.

SET NULL 상위 테이블의 행을 삭제하면 종속 행의 외부 키 값이 널(NULL)로 설정됩니다. 행의 다른 부분은 변경되지 않습니다.

테이블이 작성될 때 삭제 규칙이 명시적으로 정의되지 않으면, NO ACTION 규칙이 적용됩니다.

삭제 조작에 포함될 수 있는 테이블을 연속 삭제라고 합니다. 다음의 제한사항이 연속 삭제 관계에 적용됩니다.

- 테이블은 둘 이상의 테이블의 참조 순환 관계에서 자체 테이블로 연속 삭제될 수 없습니다.
- 테이블이 둘 이상의 종속 관계를 통해 다른 테이블로 연속 삭제될 때, 이러한 관계에는 동일한 삭제 규칙 CASCADE 또는 NO ACTION이 적용되어야 합니다.
- 자체 참조 테이블이 CASCADE 관계에 있는 다른 테이블의 종속 테이블일 때에는 자체 참조 관계에 대한 삭제 규칙도 CASCADE여야 합니다.

사용자는 상위 테이블에 아무런 조치도 취하지 않고 종속 테이블로부터 수시로 행을 삭제할 수 있습니다. 예를 들어, 부서 사원 관계에서 사원이 퇴직하게 되면 부서 테이블에 아무런 영향도 주지 않고 사원 테이블에서 해당 행을 삭제하게 됩니다.(사원-부서의 역 관계에서 부서 관리자 ID는 사원 테이블의 상위 키를 참조하는 외부 키가 됩니다. 관리자가 퇴직하면 부서 테이블에 영향을 미칩니다.)

UPDATE 규칙: DB2는 상위 행에 대한 고유 키 갱신을 방지합니다. 사용자가 종속 테이블의 외부 키를 갱신하고 외부 키가 널(NULL)이 아닌 경우, 외부 키는 관계에서 상위 테이블의 일부 상위 키 값과 일치해야 합니다. 참조 제한조건이 UPDATE 조작에 의해 위배되면, 오류가 발생하고 행이 갱신되지 않습니다.

다음은 상위 키의 컬럼 값이 갱신되는 경우입니다.

- 종속 테이블의 행이 키의 원래 값과 일치하면, 갱신 규칙이 RESTRICT인 경우 갱신은 거부됩니다.
- 종속 테이블의 행이 갱신 명령문이 완료될 때(트리거 이후를 제외하여) 일치하는 상위 키가 없으면, 갱신 규칙이 NO ACTION인 경우 갱신이 거부됩니다.

상위 행에 있는 상위 키의 값을 갱신하려면, 먼저 다음 방법에 의해 종속 테이블에 있는 하위 행에 대한 관계를 제거해야 합니다.

- 하위 행을 삭제하십시오. 또는,
- 다른 유효한 키 값을 포함할 수 있도록 종속 테이블의 외부 키를 갱신하십시오.

키 값에 대한 종속성이 없는 행에는 참조 관계의 상위가 더 이상 없으며 갱신될 수 있습니다.

외부 키의 일부가 갱신되는 중이고 외부 키 값의 어떠한 부분도 널(NULL)이 아니면, 외부 키의 새로운 값이 상위 테이블의 고유 키 값으로 나타나야 합니다. 주어진 고유 키에 대해 종속적인 외부 키가 없으면, 즉 고유 키가 들어 있는 행이 상위 행이 *아니면*, 고유 키의 일부가 갱신될 수도 있습니다. 그러나 사용자는 중복 행은 허용되지 않는 고유 키로 작업하고 있기 때문에, 이 경우에 갱신 목적으로 둘 이상의 행이 선택될 수는 없습니다.

테이블 점검 제한조건

설계 과정에서 식별된 비지니스 규칙이 테이블 점검 제한을 통해 시행됩니다. *테이블 점검 제한조건*은 테이블의 각 행에 대해 적용되는 검색 조건을 지정합니다. 이들 제한조건은 갱신 또는 삽입 명령문이 테이블에 대해 적용될 때 자동으로 활성화됩니다. 이들 제한조건은 CREATE TABLE 또는 ALTER TABLE문을 통해 정의됩니다.

테이블 점검 제한조건은 유효성 검사에 사용될 수 있습니다. 예를 들어, 부서 번호의 값은 10 - 100 사이에 와야 합니다. 사원의 직함은 ″Sales″, ″Manager″ 또는 ″Clerk″일 수 있습니다. 또는 회사에서 근무한지 9년 이상된 사원은 $40,500가 넘게 수입을 올려야 합니다.

IMPORT 및 LOAD 명령에 대한 테이블 점검 제한조건의 영향에 대해서는 *데이터 이동 유틸리티 안내 및 참조서*에서 자세한 내용을 참조하십시오.

트리거

*트리거*는 지정된 테이블에 대해 삭제, 삽입 또는 갱신 조작이 수행될 때마다 실행되는 정의된 조치 세트입니다. 트리거는 비지니스 규칙을 지원하도록 돕기 위해 정의될 수 있습니다. 또한, 트리거는 자동으로 요약을 갱신하거나 데이터를 감사하는

데 사용될 수 있습니다. 트리거는 데이터베이스에 저장되기 때문에 모든 응용프로그램에서 조치를 코드화할 필요가 없습니다. 일단 코드화된 트리거는 데이터베이스에 저장되어 있다가 응용프로그램에서 데이터베이스를 사용할 때(필요에 따라) DB2에 의해 자동으로 호출됩니다. 이로써, 데이터와 관련된 비지니스 규칙은 항상 시행됩니다. 비지니스 규칙이 변하는 경우, 트리거만을 수정할 필요가 있습니다.

트리거된 SQL문에서 사용자 정의 함수(UDF)가 호출될 수 있습니다. 이를 사용함으로써 트리거가 작동될 때 트리거 조치는 SQL이 아닌 조작을 수행할 수 있습니다. 예를 들어, 경고 메커니즘으로 전자우편이 전송될 수 있습니다. 트리거에 대해서는 *관리 안내서: 구현*의 "트리거 작성" 및 *응용프로그램 개발 안내서*에서 자세한 내용을 참조하십시오.

기타 데이터베이스 설계 고려사항

데이터베이스 설계시, 사용자가 어느 테이블에 액세스할 수 있어야 하는지를 고려해야 합니다. 테이블에 대한 액세스는 권한 부여를 통해 권한 부여되거나 권한 취소됩니다. 권한의 최상위 레벨은 시스템 관리 권한(SYSADM)입니다. SYSADM 권한을 가진 사용자는 데이터베이스 관리자 권한(DBADM)을 비롯한 기타 권한을 할당할 수 있습니다.

감사 활동, *실행기록 데이터*, *요약 테이블*, *보안*, *데이터 입력* 및 *병렬 처리 기능*과 같이 설계중에 고려해야 할 기타 사항이 있습니다.

*감사*를 목적으로 하는 경우, 지정된 기간 동안 사용자의 데이터에 대한 모든 갱신 내용을 기록해야 합니다. 예를 들어, 사원의 급여가 변경될 때마다 감사 테이블을 갱신하려 할 수도 있습니다. 이 테이블에 대한 갱신은 적절한 트리거가 정의된 경우 자동으로 이루어질 수 있습니다. 감사 활동은 DB2 감사 기능을 통해 수행될 수 있습니다. *관리 안내서: 구현*의 "DB2 활동 감사"에서 자세한 내용을 참조하십시오.

성능상 이유로, 기본 데이터를 *실행기록*으로 유지보수하는 동안 선택된 데이터량만을 액세스할 수 있습니다. 데이터가 말소되기 전에 사용 가능해야 하는 데이터의 연도 또는 월 수와 같이, 이러한 실행기록 데이터를 유지보수하는 데 필요한 사항을 설계시에 고려해야 합니다.

또한 *요약* 정보를 이용하려고 할 수 있습니다. 예를 들어, 모든 사원 정보가 들어 있는 테이블이 있을 수 있습니다. 그러나, 이 정보를 팀 또는 부서별로 별도의 테이블에 나누고자 할 수도 있습니다. 이 경우, 원래의 테이블 데이터에 기초한 각 팀 또는 부서에 대한 요약 테이블을 사용하면 도움이 됩니다. 요약 테이블에 대해서는 *관리 안내서: 구현*의 "요약 테이블 작성"에서 자세한 내용을 참조하십시오.

*보안*에 관한 사항도 설계시에 고려해야 합니다. 예를 들어, 사용자가 보안 테이블을 통해 특정 유형의 데이터에 액세스할 수 있도록 지원할 수 있습니다. 여러 가지 데이터 유형에 대한 액세스 레벨과 이 데이터에 액세스할 수 있는 사람을 정의할 수 있습니다. 사원 및 급여 데이터와 같은 기밀 데이터는 엄격한 보안 제한사항을 갖습니다. 보안 및 권한 부여에 대해서는 *관리 안내서: 구현*의 "데이터베이스 액세스 제어"에서 자세한 내용을 참조하십시오.

*구조화 유형*과 연관되어 있는 테이블을 작성할 수 있습니다. 이러한 입력된 테이블을 사용하면 해당 테이블간에 관계가 정의되어 있는 *유형 계층*이라는 계층 구조를 설정할 수 있습니다. 유형 계층은 단일 루트 유형, 상위 유형, 부속 유형으로 구성됩니다.

참조 유형 표현은 유형 계층의 루트 유형이 작성될 때 정의됩니다. 참조 목표는 항상 입력된 테이블 또는 뷰의 행입니다.

유형 행 또는 테이블이 포함된 설계 구현에 대해서는 *관리 안내서: 구현*의 "설계 구현"에서 자세한 내용을 참조하십시오. 계층 구조 내에 있는 입력된 테이블간에 데이터를 이동시키는 작업에 대해서는 *데이터 이동 유틸리티 안내 및 참조서*에서 자세한 내용을 참조하십시오.

비지니스가 많아지면, DB2 Enterprise - Extended Edition에 의해 제공된 추가 용량 및 실행 성능을 필요로 할 수 있습니다. 이 환경에서, 사용자의 데이터베이스는 여러 머신 또는 시스템에 파티션되고, 전체 데이터베이스 일부의 저장영역 및 검색에 책임이 있습니다. 각 파티션(또는 노드)은 병렬로 작업하여 SQL 또는 유틸리티 조작을 조절합니다.

병렬 조작과 관련된 문제 및 고려사항에 대해서는 이 책에서 다루고 있습니다.

제8장 물리적 데이터베이스 설계

논리 데이터베이스 설계를 완료한 후에(69 페이지의 『제7장 논리 데이터베이스 설계』 참조), 데이터베이스와 테이블이 상주하는 물리적 환경에 대해 여러가지 요소를 고려해야 합니다. 여기에는 데이터베이스를 지원하고 관리하기 위해 작성될 파일에 대한 이해, 데이터를 저장하기 위해 얼마나 많은 공간이 필요한지에 대한 이해, 데이터를 저장하는 데 필요한 테이블 공간을 어떻게 사용할지의 결정 등이 포함됩니다.

이 절에는 다음 주제를 다룹니다.

- 『데이터베이스 디렉토리』
- 100 페이지의 『테이블에 대한 공간 요구사항 추정』
- 109 페이지의 『추가 공간 요구사항』
- 111 페이지의 『노드 그룹 설계』
- 120 페이지의 『테이블 공간 설계 및 선택』
- 146 페이지의 『연합 데이터베이스 설계 고려사항』

데이터베이스 디렉토리

DB2는 데이터베이스를 작성할 때 별도의 서브디렉토리를 작성하여 제어 파일(로그 헤더 파일 등)을 저장하고 컨테이너를 기본 테이블 공간에 할당합니다. 데이터베이스에 관련된 오브젝트가 항상 데이터베이스 디렉토리에 저장되는 것은 아닙니다. 장치에 저장되는 경우를 포함하여 여러 위치에 저장될 수 있습니다.

데이터베이스는 DB2INSTANCE 환경 변수로 정의된 인스턴스 또는 명시적으로 접속된 인스턴스(ATTACH 명령을 사용하여)에 작성됩니다. *관리 안내서: 구현*의 "데이터베이스 관리 프로그램의 다중 인스턴스 사용"에서 인스턴스에 대한 소개에 대한 자세한 내용을 참조하십시오.

UNIX 기반 시스템에서 사용되는 이름 지정 스킴은 다음과 같습니다.

```
specified_path/$DB2INSTANCE/NODEnnnn/SQL00001
```

OS/2 및 Windows 운영 체제에서 사용되는 이름 지정 스킴은 다음과 같습니다.

```
D:\$DB2INSTANCE\NODEnnnn\SQL00001
```

여기서,

- specified_path는 인스턴스를 설치할 사용자 지정 위치입니다(선택적).
- NODEnnnn은 파티션된 데이터베이스 환경에서의 노드 식별자입니다. 첫 번째 노드는 NODE0000입니다.
- "D:"는 루트 디렉토리가 위치한 볼륨을 식별하는 "드라이브 이름"입니다.

SQL00001에는 작성된 첫 번째 데이터베이스와 관련된 오브젝트가 들어 있으며, 후속 데이터베이스에는 SQL00002 등과 같은 더 높은 번호가 주어집니다.

서브디렉토리는 사용자가 데이터베이스를 작성할 때 접속되는 데이터베이스 관리 프로그램 인스턴스와 이름이 동일한 디렉토리에 작성됩니다. OS/2 및 Windows 운영 체제에서, "드라이브 이름"으로 식별되는 볼륨의 루트 디렉토리 아래에 서브디렉토리가 작성됩니다. 이들 인스턴스 및 데이터베이스 서브디렉토리는 CREATE DATABASE 명령으로 지정된 경로 내에 작성되고, 데이터베이스 관리 프로그램은 이를 자동으로 유지보수합니다. 사용자의 플랫폼에 따라 각 인스턴스는 인스턴스 소유자에 의해 소유되며, 인스턴스 소유자는 해당 인스턴스의 데이터베이스에 대한 시스템 관리자(SYSADM) 권한을 갖습니다.

잠재적인 문제점을 피하기 위해서는 동일한 이름 지정 스킴을 사용하는 디렉토리를 작성하지 말고, 데이터베이스 관리 프로그램이 이미 작성해 놓은 디렉토리를 조작하지 마십시오.

데이터베이스 파일

다음과 같은 파일이 데이터베이스와 연관됩니다.

파일 이름	설명
SQLDBCON	이 파일에는 데이터베이스에 대한 조정 매개변수 및 플래그가 저장됩니다. 데이터베이스 구성 매개변수 변경에 대해서는 *관리 안내서: 성능*에서 자세한 내용을 참조하십시오.

SQLOGCTL.LFH

이 파일은 좀더 쉽게 모든 데이터베이스 로그 파일을 추적하고 제어하는 데 사용됩니다.

Syyyyyyy.LOG

0000000에서 9999999까지 번호가 매겨진 데이터베이스 로그 파일. 이들 파일의 번호는 *logprimary* 및 *logsecond* 데이터베이스 구성 매개변수에 의해 제어됩니다. 각 파일의 크기는 *logfilsiz* 데이터베이스 구성 매개변수에 의해 제어됩니다.

순환 로그에서는 파일이 재사용되고 동일한 번호로 남아 있습니다. 아카이브 로그에서의 파일 번호는 로그가 아카이브되는 순서로 증가하며, 새로운 로그가 할당됩니다. 번호가 9999999에 이르면 다시 랩됩니다.

기본값으로, 이들 로그 파일은 SQLOGDIR이라는 디렉토리에 저장됩니다. SQLOGDIR은 SQL*nnnnn* 서브디렉토리에 있습니다.

SQLINSLK 이 파일은 데이터베이스가 데이터베이스 관리 프로그램의 유일한 인스턴스에 의해 사용됨을 보장하도록 돕습니다.

SQLTMPLK 이 파일은 데이터베이스가 데이터베이스 관리 프로그램의 유일한 인스턴스에 의해 사용됨을 보장하도록 돕습니다.

SQLSPCS.1 이 파일에는 데이터베이스에 있는 모든 테이블 공간의 정의와 현재 상태가 들어 있습니다.

SQLSPCS.2 이 파일은 SQLSPCS.1의 백업 사본입니다. 이들 파일 중 하나가 없으면, 데이터베이스에 액세스할 수 없게 됩니다.

SQLBP.1 이 파일에는 데이터베이스에 있는 모든 버퍼 풀의 정의가 들어 있습니다.

SQLBP.2 이 파일은 SQLBP.1의 백업 사본입니다. 이들 파일 중 하나가 없으면, 데이터베이스에 액세스할 수 없게 됩니다.

DB2RHIST.ASC

이 파일은 데이터베이스 실행기록 파일입니다. 백업 및 복원 조작과 같이 데이터베이스에서 관리 조작의 실행기록을 유지합니다.

DB2RHIST.BAK

이 파일은 DB2RHIST.ASC의 백업 사본입니다.

주:

1. 이들 파일을 직접 변경하지 *마십시오*. 문서화된 API를 사용하거나, 명령행 처리기 및 제어 센터를 비롯하여 API를 구현하여 도구에 의해서만, 간접적으로 액세스될 수 있습니다.
2. 위의 파일을 이동시키지 마십시오.
3. 위의 파일을 제거하지 마십시오.
4. 데이터베이스 또는 테이블 공간을 백업하는 유일한 방법은 **sqlubkp**(백업 데이터베이스) API를 사용하는 것입니다(해당 API의 명령행 처리기 및 제어 센터 구현 포함).

테이블에 대한 공간 요구사항 추정

데이터베이스 오브젝트의 *크기*를 계산하는 것은 부정확합니다. 컬럼 유형 및 행 길이가 다양하므로 디스크 단편화, 여유 공간 및 가변 길이 컬럼의 사용으로 인한 오버헤드는 *크기* 계산을 어렵게 합니다. 데이터베이스 *크기*를 *초기*에 추정하고 나면, 테스트용 데이터베이스를 작성하여 몇 개의 대표적인 데이터로 이를 사용해 보십시오.

제어 센터에서, 다양한 데이터베이스 오브젝트의 *크기* 요구사항을 결정하도록 돕기 위해 설계된 몇 개의 유틸리티에 액세스할 수 있습니다.

- 오브젝트를 선택한 후 "*크기 계산*" 유틸리티를 사용할 수 있습니다. 이 유틸리티를 사용하여 테이블과 같은 기존 오브젝트의 현재 *크기*를 알 수 있습니다. 그런 다음, 오브젝트를 변경할 수 있으며 유틸리티는 예측된 새 오브젝트 값을 계산합니다. 유틸리티는 장래 성장을 고려하여 저장영역 요구사항을 계산하도록 돕습니다. 오브젝트의 단일 예측 이상을 제공합니다. 또한, 현재 값에 근거한 가장 작은 *크기*와 가능한 가장 큰 *크기* 둘다에 대한 오브젝트의 가능한 *크기* 범위를 제공합니다.
- "관련 항목 표시" 창을 사용하여 오브젝트간의 관계를 결정할 수 있습니다.

- 인스턴스 및 요청 ″DDL 생성″에서 임의의 데이터베이스 오브젝트를 선택할 수 있습니다. 이 기능은 **db2look** 유틸리티를 사용하여 데이터베이스에 대한 데이터 정의 명령문을 생성합니다. 이 유틸리티에 대해서는 *Command Reference*에서 자세한 내용을 참조하십시오.

각 경우에 ″SQL 표시″ 또는 ″명령 표시″ 단추 중 하나가 사용 가능합니다. 또한 결과 SQL 또는 명령을 스크립트 파일로 저장하여 나중에 사용할 수도 있습니다. 이 모든 유틸리티는 사용자를 돕는 온라인 도움말을 갖습니다.

물리 데이터베이스 요구사항을 계획해 나가면서 이들 기능을 염두에 두십시오.

데이터베이스의 크기를 계산할 때, 다음과 같은 사항을 고려해야 합니다.

- 『시스템 카탈로그 테이블』
- 102 페이지의 『사용자 테이블 데이터』
- 104 페이지의 『Long 필드 데이터』
- 105 페이지의 『대형 오브젝트(LOB) 데이터』
- 106 페이지의 『색인 공간』

다음과 관련된 공간 요구사항은 설명되지 않습니다.

- 지역 데이터베이스 디렉토리 파일
- 시스템 데이터베이스 디렉토리 파일
- 다음을 포함한 운영 체제에서 필요한 파일 관리 오버헤드
  - 파일 블록 크기
  - 디렉토리 제어 공간

시스템 카탈로그 테이블

데이터베이스가 작성될 때 시스템 카탈로그 테이블이 작성됩니다. 시스템 테이블은 데이터베이스 오브젝트 및 특권이 데이터베이스에 추가되면서 커집니다. 초기에는 약 3.5MB의 디스크 공간을 사용합니다.

카탈로그 테이블에 할당되는 공간량은 테이블 공간의 유형 및 카탈로그 테이블이 들어 있는 테이블 공간의 extent 크기에 따라 다릅니다. 예를 들어, extent 크기가

32인 DMS 테이블 공간이 사용될 경우에는 초기에 20MB의 공간이 카탈로그 테이블 공간에 할당됩니다. 120 페이지의 『테이블 공간 설계 및 선택』에서 자세한 내용을 참조하십시오.

주: 다중 파티션이 있는 데이터베이스의 경우, 카탈로그 테이블은 CREATE DATABASE가 발행된 파티션에만 상주합니다. 해당 파티션에만 카탈로그 테이블용 디스크 공간이 있으면 됩니다.

사용자 테이블 데이터

기본적으로, 테이블 데이터는 4KB 페이지씩 저장됩니다. 각 페이지(페이지 *크기*와 무관)에는 데이터베이스 관리 프로그램에 대한 76바이트 오버헤드가 들어 있습니다. 이는 4KB 페이지에 4005바이트 길이를 초과하는 행이 없더라도, 사용자 데이터(또는 행)를 보유하기 위해 4020바이트를 남겨 놓습니다. 하나의 행이 여러 페이지에 걸쳐질 수는 *없습니다*. 4KB 페이지 *크기*를 사용할 때에는 최대 500 컬럼을 사용할 수 있습니다.

테이블 데이터 페이지에는 LONG VARCHAR, LONG VARGRAPHIC, BLOB, CLOB 또는 DBCLOB 데이터 유형으로 정의된 컬럼에 대한 데이터가 들어 있지 *않습니다*. 그러나, 테이블 데이터 페이지 내의 행에는 이 컬럼에 대한 설명자가 들어 있습니다. 이러한 데이터 유형이 들어 있는 테이블 오브젝트의 공간 요구사항을 계산하는 방법에 대해서는 104 페이지의 『Long 필드 데이터』 및 105 페이지의 『대형 오브젝트(LOB) 데이터』에서 자세한 내용을 참조하십시오.

행은 보통 first-fit 순서로 테이블에 삽입됩니다. 파일은 새로운 행을 보유할 정도의 *크기*를 가진 공간 중 첫 번째로 사용 가능한 공간을 검색합니다(여유 공간 맵을 통해). 행이 *갱신*될 때, *갱신*된 행이 수록될 페이지에 충분한 공간이 남아 있어야만 해당 행이 *갱신*됩니다. 이러한 *경우*, 레코드는 *갱신*된 행의 테이블 파일에서 새로운 위치를 가리키는 원래 행의 위치에 작성됩니다.

ALTER TABLE APPEND ON문을 호출한 *경우*, 데이터는 항상 추가되며 해당 데이터 페이지의 사용 가능한 공간에 대한 정보는 보존되지 않습니다. 이 명령문에 대해서는 *SQL 참조서*에서 자세한 내용을 참조하십시오.

데이터베이스의 각 사용자 테이블에서 4KB 페이지 수는 다음과 같이 산정됩니다.

```
ROUND DOWN(4020/(average row size + 10)) = records_per_page
```

이 결과를 다음에 삽입합니다.

```
(number_of_records/records_per_page) * 1.1 = number_of_pages
```

여기서, 평균 행 *크기*는 평균 컬럼 *크기*의 합이며(각 컬럼의 *크기*에 대해서는 *SQL 참조서*에서 CREATE TABLE문을 참조하십시오), ″1.1″ 인수는 오버헤드용입니다.

주: 이 공식은 단지 추정치만을 제공합니다. 단편화 및 오버플로우 레코드로 인해 레코드 길이가 다양한 경우 추정의 정확도가 줄어듭니다.

또한, 8KB, 16KB 또는 32KB 페이지 *크기*를 갖는 버퍼 풀 또는 테이블 공간을 작성할 수 있는 옵션이 있습니다. 특정 *크기*의 테이블 공간 내에서 작성된 모든 테이블에는 일치하는 페이지 *크기*가 있습니다. 단일 테이블이나 색인 오브젝트는 32KB 페이지 *크기*를 가정할 때 512GB만큼 클 수 있습니다. 8KB, 16KB 또는 32KB 페이지 *크기*를 사용할 때에는 최대 1012 컬럼을 사용할 수 있습니다. 컬럼의 최대 수는 4KB 페이지 *크기*에 대해 500입니다. 페이지 *크기*에 따라 최대 행 길이도 변합니다.

- 페이지 *크기*가 4KB이면, 행 길이는 최대 4005바이트일 수 있습니다.
- 페이지 *크기*가 8KB이면, 행 길이는 최대 8101바이트일 수 있습니다.
- 페이지 *크기*가 16KB이면, 행 길이는 최대 16,293바이트일 수 있습니다.
- 페이지 *크기*가 32KB이면, 행 길이는 최대 32,677바이트일 수 있습니다.

페이지 *크기*가 클수록 색인의 레벨 수가 줄어들 가능성이 높아집니다. 임의의 행을 읽고 쓰는 OLTP(온라인 거래 처리) 응용프로그램을 사용하여 작업하는 경우, 페이지 *크기*가 작으면 불필요한 행에 소비되는 버퍼 공간이 더 적기 때문에 보다 바람직합니다. 한번에 많은 수의 연속 행에 액세스하는 DSS 응용프로그램을 사용하여 작업하는 경우, 페이지 *크기*가 크면 특정 행 번호를 읽는 데 필요한 입출력 요청 수가 줄어들기 때문에 보다 바람직합니다. 255로 페이지 *크기*를 나눈 값보다 행 *크기*가 작을 때 예외가 발생합니다. 이 경우, 각 페이지마다 소모되는 공간이 있게 됩니다. (페이지당 최대 255 행만 가능합니다.) 이러한 소모 공간을 줄이려면, 보다 적은 페이지 *크기*를 사용하는 것이 바람직합니다.

백업을 다른 페이지 *크기*로 복원할 수 없습니다.

756 컬럼 이상을 나타내는 IXF 데이터 파일을 가져올 수 없습니다. 테이블로 데이터 가져오기와 IXF 데이터 파일에 대한 자세한 내용은 *데이터 이동 유틸리티 안내 및 참조서*에서 참조하십시오.

선언된 임시 테이블은 "사용자 임시" 테이블 공간 유형에서만 작성될 수 있습니다. 기본 사용자 임시 테이블 공간이 없습니다. 임시 테이블은 LONG 데이터를 가질 수 없습니다. 응용프로그램이 데이터베이스에서 연결 해제될 때 테이블은 내재적으로 삭제되며, 공간 요구사항의 계산을 고려해야 합니다.

Long 필드 데이터

Long 필드 데이터는 다른 데이터 유형과는 다르게 구성된 개별 테이블 오브젝트에 저장됩니다(102 페이지의 『사용자 테이블 데이터』 및 105 페이지의 『대형 오브젝트(LOB) 데이터』 참조).

데이터는 512바이트의 "2의 제곱"배 *크기*의 세그먼트로 파티션된 32KB 영역에 저장됩니다(즉, 이 세그먼트의 *크기*는 512바이트, 1024바이트, 2048바이트 등으로 32 768바이트까지 가능합니다).

Long 필드 데이터 유형(LONG VARCHAR 또는 LONG VARGRAPHIC)은 사용 가능한 공간을 쉽게 재생시킬 수 있는 방법으로 저장됩니다. 할당 및 사용 가능한 공간 정보는 4KB의 할당 페이지에 저장되며, 이는 오브젝트 전반에 걸쳐 나타납니다.

오브젝트 공간 중 사용하고 있지 않은 공간의 양은 long 필드 데이터의 *크기*와 이 *크기*가 모든 데이터 발생에 대해 상대적으로 일정한지 여부에 따라 달라집니다. 256바이트 이상의 데이터 항목인 경우에는 사용되지 않는 이 공간이 long 필드 데이터 *크기*의 50퍼센트까지 될 수 있습니다.

문자 데이터가 페이지 *크기*보다 작고 나머지 데이터의 레코드와 길이가 맞을 경우, LONG VARCHAR 또는 LONG VARGRAPHIC 대신 CHAR, GRAPHIC, VARCHAR 또는 VARGRAPHIC 데이터 유형을 사용해야 합니다.

대형 오브젝트(LOB) 데이터

대형 오브젝트(LOB) 데이터는 다른 데이터 유형과는 달리 구성된 두 개의 개별 테이블 오브젝트에 저장됩니다(102 페이지의 『사용자 테이블 데이터』 및 104 페이지의 『Long 필드 데이터』 참조).

LOB 데이터에서 필요한 공간을 예측하려면, 이러한 데이터 유형으로 정의된 데이터를 저장하는 데 사용되는 두 개의 테이블 오브젝트를 고려해야 합니다.

- **LOB 데이터 오브젝트**

  데이터는 1024바이트의 ″2의 제곱″배 *크기*의 세그먼트로 파티션된 64MB 영역에 저장됩니다. (즉, 이 세그먼트의 *크기*는 1024바이트, 2048바이트, 4069바이트 등으로 64MB까지 가능합니다.)

  LOB 데이터에 의해 사용되는 디스크 양을 줄이기 위해 CREATE TABLE 및 ALTER TABLE문의 *lob-options*절에 COMPACT 옵션을 지정할 수 있습니다. COMPACT 옵션은 LOB 데이터를 더 작은 세그먼트로 분할함으로써 필요한 디스크 공간량을 최소합니다. 이 프로세스는 데이터 압축을 포함하지 않으며, 1KB 정도의 최소 공간을 사용할 뿐입니다. COMPACT 옵션 사용은 LOB 값에 추가할 때 성능이 저감된 결과를 가져올 수 있습니다.

  LOB 데이터 오브젝트에 포함된 사용 가능한 공간의 양은 삽입될 LOB 값의 *크기*뿐만 아니라 갱신 및 삭제 활동에도 영향을 받습니다.

- **LOB 할당 오브젝트**

  할당 및 사용 가능한 공간 정보는 실제 데이터와는 구별되어 4KB의 할당 페이지에 저장됩니다. 이들 4KB 페이지의 수는 대형 오브젝트(LOB) 데이터에 할당된 데이터량(사용하고 있지 않은 공간도 포함)에 따라 달라집니다. 오버헤드는 64GB마다 4KB이고, 여기에 8MB마다 4KB페이지씩 더해지는 방식을 계산됩니다.

문자 데이터가 페이지 *크기*보다 작고 나머지 데이터의 레코드와 길이가 맞을 경우, BLOB, CLOB 또는 DBCLOB 대신 CHAR, GRAPHIC, VARCHAR 또는 VARGRAPHIC 데이터 유형을 사용해야 합니다.

색인 공간

각 색인에 대해, 필요한 공간은 다음과 같이 추정될 수 있습니다.

```
(평균 색인 키 크기 + 8) * 행 수 * 2
```

까지입니다. 여기서,

- "평균 색인 키 *크기*"란 색인 키에 있는 각 컬럼의 바이트 수를 말합니다. 서로 다른 데이터 유형의 컬럼에 대한 바이트 계수를 계산하는 방법에 대해서는 *SQL 참조서*의 CREATE TABLE문을 참조하십시오. (VARCHAR 및 VARGRAPHIC 컬럼의 평균 컬럼 *크기*를 추정하려면, 현재 데이터 *크기*의 평균에 1바이트를 더하여 사용하십시오. 최대 선언 *크기* 값은 사용하지 마십시오.)
- "2"라는 요소는 비 리프 페이지 및 여유 공간과 같은 오버헤드용 값입니다.

주: 널(NULL)을 허용하는 모든 컬럼에 대해 널(NULL) 표시기용으로 1바이트를 여분으로 추가하십시오.

색인 작성시 임시 공간이 필요합니다. 색인 작성중에 필요한 최대 임시 공간량은 다음과 같이 추정됩니다.

```
(평균 색인 키 크기 + 8) * 행 수 * 3.2
```

여기서 "3.2"의 인수는 색인 오버헤드이며, 색인 작성 중 분류를 위한 공간이 필요합니다.

주: 비고유 색인의 경우, 4바이트만 있으면 중복 키 항목을 저장합니다. 위의 측정값에는 중복 값이 없습니다. 색인 저장에 필요한 공간이 위 수식에 의해 과대 측정되었습니다.

다음 두 가지 공식은 리프 페이지의 수를 계산하기 위해 사용될 수 있습니다. (두 번째는 더 정확한 계산을 제공합니다.) 이 계산의 정확도는 평균이 실제 데이터를 얼마나 반영하는가에 의해 *크게* 좌우됩니다.

주: SMS 테이블 공간의 최소 요구 공간은 12KB입니다. DMS 테이블 공간의 최소 공간은 extent입니다.

- 리프 페이지당 평균 키 수는 대략 다음과 같습니다.

```
(.9 * (U - (M*2))) * (D + 1)
----------------------------
      K + 6 + (4 * D)
```

까지입니다. 여기서,

- U는 페이지에서 사용 가능한 공간은 페이지 *크기*에서 100을 뺀 것과 같습니다. 페이지 *크기*가 4096인 경우, U는 3996입니다.
- M = U / (8 + *minimumKeySize*)
- D = 키 값마다 중복된 평균 수
- K = *averageKeySize*

minimumKeySize 및 *averageKeysize*는 각 널(NULL) 입력 가능 키 부분에 대해 여분의 바이트와 각 변수 길이 키 부분에 대해 추가의 바이트를 가지고 있어야 한다는 점을 유의하십시오.

포함 컬럼이 있는 경우, *minimumKeySize* 및 *averageKeySize*로 설명되어야 합니다.

색인 작성중에 기본값 10퍼센트보다 더 여유 공간 비율 값이 지정된 경우, .9는 (100 - pctfree)/100 값으로 대체될 수 있습니다.

- 리프 페이지당 평균 키 수에 대한 보다 정확한 계산은 다음과 같습니다.

  ```
  L = 리프 페이지 수 = X / (리프 페이지당 평균 키 수)
  ```

  여기서, X는 테이블의 총 행 수입니다.

  다음과 같이 색인의 원래 *크기*를 산정할 수 있습니다.

  ```
  (L + 2L/(리프 페이지의 평균 키 수)) * 페이지 크기
  ```

  DMS 테이블 공간의 경우, 테이블의 모든 색인에 대한 *크기*를 같이 추가하고, 색인이 상주하는 테이블 공간의 extent *크기* 배수에서 반올림하십시오.

  INSERT/UPDATE 활동으로 인한 색인 확장 추가 공간을 제공해야 하는데, 이 결과에 페이지가 파티션될 수 있습니다.

색인의 레벨 수 뿐 아니라 색인의 원래 *크기*에 대한 보다 정확한 수치를 구하려면, 다음 계산식을 사용하십시오(포함 컬럼이 색인 정의에 사용되는 경우 이 수치는 특히 중요할 수 있습니다). 비 리프 페이지의 평균 키 수는 대략 다음과 같습니다.

```
(.9 * (U - (M*2))) * (D + 1)
----------------------------
    K + 12 + (8 * D)
```

까지입니다. 여기서,

- U는 페이지에서 사용 가능한 공간은 페이지 *크기*에서 100을 뺀 것과 같습니다. 페이지 *크기*가 4096인 경우, U는 3996입니다.
- D는 비 리프 페이지의 키 값당 평균 중복 수입니다. (이 값은 리프 페이지의 값보다 훨씬 적으며 0으로 설정하여 단순화할 수 있습니다.)
- M = U / (8 + 비 리프 페이지의 *minimumKeySize*)
- K = 비 리프 페이지의 *averageKeySize*

포함 컬럼 있는 경우를 제외하고는 *minimumKeySize*와 *averageKeySize*는 리프 페이지의 경우와 동일합니다. 포함 컬럼은 비 리프 페이지에 저장되지 않습니다.

색인 작성시 최대 10 퍼센트의 여유 공간 비율이 비 리프 페이지에 남게 되므로, 해당 값이 .9보다 크지 않은 경우 .9를 (100 - pctfree)/100으로 대체해서는 안 됩니다.

비 리프 페이지 수는 다음과 같이 산정할 수 있습니다.

```
if L > 1 then {P++; Z++}
While (Y > 1)
{
   P = P + Y
   Y = Y / N
  Z++
}
```

까지입니다. 여기서,

- P는 페이지 수(초기에는 0)입니다.
- L은 리프 페이지 수입니다.
- N은 각 비 리프 페이지당 키 수입니다.

– Y = L / N

– Z은 색인 트리의 레벨 수(초기에는 1)입니다.

페이지 총 수는 다음과 같습니다.

```
T = (L + P + 2) * 1.0002
```

추가적인 0.02 퍼센트는 공간 맵 페이지를 포함하는 오버헤드용입니다.

색인을 작성하는 데 필요한 공간량은 다음과 같이 산정됩니다.

```
T * pagesize
```

추가 공간 요구사항

다음과 같은 추가 공간이 필요합니다.

- 『로그 파일 공간』
- 110 페이지의 『임시 작업 공간』

로그 파일 공간

로그 파일에 필요한 공간량의 범위(바이트 단위)는

```
( logprimary * (logfilsiz + 2 ) * 4096 ) + 8192
```

에서

```
( (logprimary + logsecond) * (logfilsiz + 2 ) * 4096 ) + 8192
```

까지입니다. 여기서,

- *logprimary*는 데이터베이스 구성 파일에서 정의된 1차 로그 파일 수입니다.
- *logsecond*는 데이터베이스 구성 파일에서 정의된 2차 로그 파일 수입니다.
- *logfilsiz*는 데이터베이스 구성 파일에서 정의된 각 로그 파일에 있는 페이지 수입니다.
- 2는 각 로그 파일에 필요한 헤더 페이지 수입니다.
- 4096은 한 페이지의 바이트 수입니다.
- 8192는 로그 제어 파일의 크기(바이트 단위)입니다.

logprimary, *logsecond* 및 *logfilsiz* 구성 매개변수에 대한 정보는 *관리 안내서: 성능*을 참조하십시오.

주: 사용 중인 총 로그 공간 크기는 32GB를 초과할 수 없습니다.

로그 파일 공간의 한계는 데이터베이스 관리 프로그램 수행시 필요로 하는 2차 로그 파일의 실제 수에 따라 달라집니다. 결코 이 상한선을 요구해서는 안되며, 다만 대용량 작업의 특별한 경우에만 사용될 수 있습니다.

데이터베이스가 롤 포워드 복구에 대해 작동 가능화된 경우에는 특수 로그 공간 요구사항을 고려해야 합니다.

- *logretain* 구성 매개변수가 작동되면, 로그 파일이 로그 경로 디렉토리에 아카이브됩니다. 온라인 디스크 공간은 사용자가 로그 파일을 다른 위치로 이동시키지 않는 한 결국 채워집니다.
- *userexit* 구성 매개변수가 작동되면, User Exit 프로그램이 아카이브 로그 파일을 다른 위치로 이동시킵니다. 다음 사항을 위해 충분한 로그 공간이 필요합니다.
  - User Exit 프로그램에 의해 이동되기를 기다리는 온라인 아카이브 로그
  - 차후에 사용하기 위해 형식화되고 있는 새 로그 파일

임시 작업 공간

일부 SQL문은 처리(예: 메모리에서 수행될 수 없는 정렬 조작용 작업 파일)를 위해 임시 테이블이 필요합니다. 이 임시 테이블은 디스크 공간을 요구합니다. 필수 공간량은 조회, 리턴된 테이블 크기에 의해 좌우되며, 예측할 수 없습니다.

데이터베이스 시스템 모니터와 조회 테이블 공간 API를 사용하면 정상적인 조작 과정에서 사용되는 작업 공간량을 추적하는 데 도움이 됩니다.

노드 그룹 설계

노드 그룹은 데이터베이스에 속하는 것으로서 정의된 하나 이상의 노드의 이름 지정된 세트입니다. 데이터베이스 시스템 구성의 일부인 각 데이터베이스 파티션은 *db2nodes.cfg*라고 하는 노드 구성 파일에 이미 정의되어 있어야 합니다. 노드 그룹은 하나의 데이터베이스 파티션에서 데이터베이스 시스템에 대해 정의된 전체 데이터베이스 파티션까지 포함할 수 있습니다.

CREATE NODEGROUP문을 사용하여 새 노드 그룹을 작성하며, ALTER NODEGROUP문을 사용하여 수정할 수 있습니다. 노드 그룹에서 하나 이상의 데이터베이스 파티션을 추가하거나 삭제할 수 있습니다. 노드 그룹을 수정하기 전에 데이터베이스 파티션이 `db2nodes.cfg` 파일에 정의되어 있어야 합니다. 테이블 공간은 노드 그룹 내에 상주합니다. 테이블은 테이블 공간 내에 상주합니다.

노드 그룹이 작성되거나 수정될 때에는 *파티션 맵*이 노드 그룹과 연관됩니다. 데이터베이스 관리 프로그램은 *파티션 키* 및 해싱 알고리즘과 함께 파티션 맵을 사용하여 노드 그룹의 어떤 데이터베이스 파티션에 주어진 데이터 행을 저장할 것인지 결정합니다. 파티션 맵에 대해서는 114 페이지의 『파티션 맵』에서, 파티션 키에 대해서는 115 페이지의 『파티션 키』에서 자세한 내용을 참조하십시오.

파티션되지 않은 데이터베이스에서는 파티션 키 또는 파티션 맵이 필요 없습니다. 파티션되지 않은 데이터베이스를 사용할 경우에는 노드 그룹 설계에 대해 고려하지 않아도 됩니다. *데이터베이스 파티션*은 사용자 데이터, 색인, 구성 파일, 트랜잭션 로그로 이루어진 데이터베이스 일부입니다. 데이터베이스 관리 프로그램은 데이터베이스가 작성될 때 작성된 기본 노드 그룹을 사용합니다. IBMCATGROUP은 시스템 카탈로그가 들어 있는 테이블 공간에 대한 기본 노드 그룹입니다. IBMTEMPGROUP은 시스템 임시 테이블 공간의 기본 노드 그룹입니다. IBMDEFAULTGROUP은 사용자가 선택한 사용자 정의 테이블이 들어 있는 테이블 공간에 대한 기본 노드 그룹입니다. 선언된 임시 테이블의 사용자 임시 테이블 공간은 IBMDEFAULTGROUP이나 사용자 작성 노드 그룹으로 작성될 수 있지만, IBMTEMPGROUP으로는 작성될 수 없습니다.

다중 파티션 노드 그룹을 사용하는 경우, 다음가 같은 설계시 중요사항을 고려하십시오.

- 다중 파티션 노드 그룹에서는 파티션 키의 상위 집합일 경우에만 고유 색인을 작성할 수 있습니다.
- 데이터베이스의 데이터베이스 파티션 수에 따라, 하나 이상의 단일 파티션 노드 그룹 및 하나 이상의 복수 파티션 노드 그룹을 보유할 수 있습니다.
- 각 데이터베이스 파티션에는 고유한 파티션 번호가 지정되어야 합니다. 하나 이상의 노드 그룹에서 같은 데이터베이스 파티션을 찾을 수 있습니다.
- 시스템 카탈로그 테이블이 들어 있는 데이터베이스 파티션 복구가 신속하게 이루어지도록 하려면, 동일한 데이터베이스 파티션에 사용자 테이블을 놓지 마십시오. IBMCATGROUP 노드 그룹의 데이터베이스 파티션을 포함하지 않은 노드 그룹에 사용자 테이블을 위치지정하면 완료됩니다.

더 큰 테이블과의 공동 *배치*를 이용하려는 경우가 아니라면, 작은 테이블을 단일 파티션 노드 그룹에 배치해야 합니다. 공동 배치는 같은 데이터베이스 파티션의 관련 데이터가 들어 있는 다른 테이블과는 구별되는 행의 위치입니다. 공동 배치된 테이블로 DB2는 조인 전략을 더 효율적으로 사용할 수 있습니다. 공동 배치된 테이블은 단일 파티션 노드 그룹에 상주할 수 있습니다. 테이블이 다중 파티션 노드 그룹에 상주하는 경우, 테이블은 공동 배치된 것으로 간주되며 파티션 키에서 같은 컬럼 수를 가지고 있고, 대응하는 컬럼의 데이터 유형은 호환적인 파티션입니다. 공동 배치된 테이블에서 동일한 파티션 키 값을 갖는 행은 동일한 데이터베이스 파티션에 배치됩니다. 테이블은 동일한 노드 그룹의 별도의 테이블 공간에 있을 수도 있고, 공동 배치된 것으로 간주될 수도 있습니다.

너무 많은 데이터베이스 파티션에 중간 크기의 테이블을 확장시키지 않도록 해야 합니다. 예를 들어, 100MB의 테이블은 32 파티션 노드 그룹에서 보다 16 파티션 노드 그룹에서 더 잘 수행됩니다.

노드 그룹을 사용하여 온라인 거래 처리(OLTP) 테이블을 결정 지원(DSS) 테이블에서 분리하면 OLTP 거래의 성능이 역으로는 영향을 받지 않습니다.

노드 그룹 설계 고려사항

논리 데이터베이스 설계와 처리되는 데이터량은 데이터베이스이 파티션될 필요가 있는지 여부를 제시합니다. 이 절에서는 데이터베이스 파티션에 관련된 다음 주제들을 다룹니다.

- 『데이터 파티션』
- 114 페이지의 『파티션 맵』
- 115 페이지의 『파티션 키』
- 118 페이지의 『테이블 배열』
- 118 페이지의 『파티션 호환성』
- 119 페이지의 『복제된 요약 테이블』

데이터 파티션

DB2는 데이터베이스의 몇몇 데이터베이스 파티션에 데이터를 저장할 수 있도록 파티션된 저장영역 모델을 지원합니다. 즉, 데이터는 물리적으로는 둘 이상의 데이터베이스 파티션에 저장되지만, 마치 같은 장소에 있는 것처럼 액세스할 수 있습니다. 파티션된 데이터베이스에서 데이터에 액세스하는 응용프로그램과 사용자는 데이터의 물리적 위치를 알 필요가 없습니다.

데이터는 물리적으로 파티션되어 있어도, 논리적으로는 하나의 전체 단위로 사용되고 관리됩니다. 사용자는 파티션 키를 선언하여 데이터의 파티션 방법을 선택할 수 있습니다. 또한 사용자는 데이터가 저장될 테이블 공간 및 관련 노드 그룹을 선택함으로써 테이블 데이터가 어떤 데이터베이스 파티션에, 몇 개의 파티션에 걸쳐 분배될 것인지 결정할 수 있습니다. 그리고, 갱신할 수 있는 파티션 맵이 해싱 알고리즘과 함께 사용되어 파티션 키 값과 데이터베이스 파티션간의 맵핑을 지정합니다. 이를 통해 각 데이터 행의 위치와 검색 방법이 결정됩니다. 결과적으로, 작은 테이블이 하나 이상의 데이터베이스 파티션에 저장될 수 있도록 하면서, 큰 테이블의 워크로드를 파티션된 데이터베이스에 분배할 수 있습니다. 각 데이터베이스 파티션에는 저장된 데이터에 대한 지역 색인이 있어 지역 데이터 액세스 성능을 증진시킵니다.

모든 테이블을 데이터베이스의 모든 데이터베이스 파티션에서 파티션해야 한다는 제약은 없습니다. DB2는 *부분 디클러스터링*을 지원하는데, 이 기능은 테이블 및 테이블 공간을 시스템(즉, 노드 그룹)의 데이터베이스 파티션 부속 집합에 나눌 수 있는 기능입니다.

각 데이터베이스 파티션에 테이블을 배치할 때 고려해야 할 또다른 사항은 요약 테이블을 사용한 다음 해당 테이블을 복제하는 것입니다. 필요한 정보가 들어 있

는 요약 테이블을 작성한 다음 각 노드로 복제할 수 있습니다. 119 페이지의 『복제된 요약 테이블』에서 자세한 내용을 참조하십시오.

파티션 맵

파티션된 데이터베이스 환경에서, 데이터베이스 관리 프로그램에는 어느 데이터베이스 파티션에 테이블 행이 저장되어 있는지 알 수 있는 방법이 있어야 합니다. 데이터베이스 관리 프로그램은 필요로 하는 데이터를 찾는 장소를 알고, *파티션 맵*이라는 맵을 사용하여 데이터를 찾아야 합니다.

파티션 맵은 다중 파티션 노드 그룹의 경우 4 096 항목을 포함하고, 단일 파티션 노드 그룹의 경우 단일 항목을 포함하는 내부적으로 생성된 배열입니다. 단일 파티션 노드 그룹의 경우, 파티션 맵은 데이터베이스 테이블의 모든 행이 저장된 데이터베이스 파티션의 파티션 번호가 들어 있는 하나의 항목만을 보유합니다. 다중 파티션 노드 그룹의 경우, 노드 그룹 파티션 번호는 원형식으로 지정됩니다. 도시의 지도가 격자를 사용하여 각 부분으로 구분되듯이, 데이터베이스 관리 프로그램은 *파티션 키*를 사용하여 데이터가 저장된 위치(데이터베이스 파티션)를 판별합니다.

예를 들어, 4개의 데이터베이스 파티션(0 - 3의 번호로 지정)에 작성된 데이터베이스가 있다고 가정해 보십시오. 이 데이터베이스의 IBMDEFAULTGROUP 노드 그룹에 대한 파티션 맵은 다음과 같습니다.

```
0 1 2 3 0 1 2 ...
```

노드 그룹이 데이터베이스 파티션 1과 2를 사용하여 데이터베이스에 작성되면, 해당 노드 그룹에 대한 파티션 맵은 다음과 같습니다.

```
1 2 1 2 1 2 1 ...
```

데이터베이스에 로드될 테이블에 대한 파티션 키가 1에서 500 000 사이의 정수일 경우, 파티션 키는 0에서 4 095 사이의 파티션 번호로 해쉬됩니다. 이 숫자는 해당 행의 데이터베이스 파티션을 선택하는 색인으로 파티션 맵에 사용됩니다.

115 페이지의 그림27은 파티션 키 값(c1, c2, c3)을 갖는 행이 파티션 2에 맵핑된 것으로서, 차례로 데이터베이스 파티션 n5를 참조합니다.

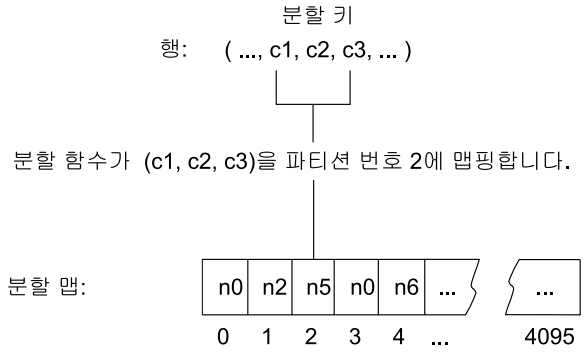

그림 27. 파티션 맵을 사용한 데이터 분배

파티션 맵을 사용하면 파티션이 설정된 데이터베이스에서 데이터가 저장되는 곳을 용통성 있게 제어할 수 있습니다. 향후에 데이터베이스의 데이터베이스 파티션 데이터 분배를 변경할 필요가 있을 경우, 데이터 재분배 유틸리티를 사용하면 됩니다. 이 유틸리티를 사용하면, 데이터 분배를 다시 조절할 수 있습니다. 이 유틸리티에 대해서는 *관리 안내서: 성능*의 "데이터베이스 파티션에서의 데이터 재분배"를 참조하십시오.

테이블 파티션 정보 확보(**sqlugtpi**) API를 사용하면, 열람할 수 있는 파티션 맵의 사본을 얻을 수 있습니다. 이 API에 관한 자세한 정보는 *Administrative API Reference*를 참조하십시오.

파티션 키

*파티션 키*는 특정 데이터 행이 저장될 파티션을 결정하는 데 사용되는 컬럼(또는 컬럼 그룹)입니다. 파티션 키는 CREATE TABLE문을 사용하여 테이블에서 정의됩니다. 노드 그룹에서 둘 이상의 데이터베이스 파티션에 분포된 테이블 공간의 테이블에 대해 파티션 키가 정의되어 있지 않을 경우, 기본 키의 첫 번째 컬럼에서 기본값에 따라 작성됩니다. 기본 키가 지정되어 있지 않을 경우, 테이블에 대해 정의된 기본 파티션 키는 첫 번째 long이 아닌 필드 컬럼이 됩니다. (*Long*은 모든 long 데이터 유형 및 모든 대형 오브젝트(LOB) 데이터 유형을 포함합니다.) 단일 파티션 노드 그룹과 연관된 테이블 공간에서 테이블을 작성하고 있으며 파티션 키를 갖고자 할 경우, 파티션 키를 명시적으로 정의해야 합니다. 파티션 키는 기본값에 의해 작성되지 않습니다.

어떠한 컬럼도 기본 파티션 키에 대한 요구사항을 충족시킬 수 없는 경우, 파티션 키 없이 테이블이 작성됩니다. 파티션 키 없는 테이블은 단일 파티션 노드 그룹에

서만 허용됩니다. ALTER TABLE문을 사용하여 나중에 파티션 키를 추가하거나 삭제할 수 있습니다. 단일 파티션 노드 그룹과 연관된 테이블 공간의 테이블에서만 파티션 키를 변경할 수 있습니다.

올바른 파티션 키를 선택하는 것이 중요합니다. 다음을 고려해야 합니다.

- 테이블에 액세스하는 방법
- 조회 워크로드의 특성
- 데이터베이스 시스템이 채용하고 있는 조인 전략

공동 배치가 주요한 고려사항이 아닐 경우, 테이블에 적합한 파티션 키를 선택하는 것이 노드 그룹의 모든 데이터베이스 파티션에 데이터를 균등하게 분배하는 방법입니다. 노드 그룹과 연관된 테이블 공간의 각 테이블에 대한 파티션 키가 테이블을 배열할 것인지 여부를 결정합니다. 다음과 같은 경우 테이블은 공동 배치된 것으로 간주됩니다.

- 동일한 노드 그룹에 있는 테이블 공간에 테이블이 배치될 경우
- 각 테이블의 파티션 키가 동일한 수의 컬럼을 가지고 있는 경우
- 해당 컬럼의 데이터 유형이 파티션 호환 가능한 것일 경우

이들 문자는 공동 배치된 테이블 행은 같은 파티션 키 값을 사용하여 같은 파티션에 위치지정됩니다. 파티션 호환성에 대해서는 118 페이지의 『파티션 호환성』에서 자세한 내용을 참조하십시오. 테이블 공동 배치에 대해서는 118 페이지의 『테이블 배열』에서 자세한 내용을 참조하십시오.

파티션 키가 적합하지 않을 경우, 데이터 분배가 한편으로 치우칠 수 있습니다. 데이터가 치우쳐 분배된 컬럼과 구별 값의 적은 수를 가진 컬럼을 파티션 키로 선택해서는 안 됩니다. 구별 값 수는 노드 그룹의 모든 데이터베이스 파티션에 행이 균등하게 분배될 만큼 되어야 합니다. 파티션 해싱 알고리즘을 적용할 때 들어가는 노력의 양은 파티션 키의 크기에 비례합니다. 파티션 키는 17 컬럼 이상일 수 없으며, 컬럼 수가 적을수록 다 나은 성능을 제공합니다. 파티션 키에 불필요한 컬럼이 들어가서는 안 됩니다.

다음 요점은 파티션 키를 정의할 때 고려되어야 합니다.

- Long 데이터 유형(LONG VARCHAR, LONG VARGRAPHIC, BLOB, CLOB 또는 DBCLOB)만 들어 있는 다중 파티션 테이블을 작성할 수 없습니다.
- 파티션 키 정의는 변경될 수 없습니다.
- 파티션 키는 가장 자주 조인되는 컬럼을 포함해야 합니다.
- 파티션 키는 GROUP BY절에 종종 참여하는 컬럼으로 구성되어야 합니다.
- 모든 고유 키 또는 기본 키는 파티션 키 컬럼 모두를 포함해야 합니다.
- 온라인 거래 처리(OLTP) 환경에서 파티션 키에 있는 모든 컬럼은 상수 또는 호스트 변수와 함계 등호(=) 술어를 사용하여 거래에 참여해야 합니다. 예를 들어, 다음과 같은 거래에서 자주 사용되는 사원 번호 *emp_no*가 있다고 가정해 보십시오.

  ```
  UPDATE emp_table SET ... WHERE
  emp_no = host-variable
  ```

  이 경우, EMP_NO 컬럼은 EMP_TABLE의 좋은 단일 컬럼 파티션 키를 작성합니다.

DB2_UPDATE_PART_KEY 레지스트리 변수를 NO(아니오)로 설정하면 표의 행에 대한 파티션 키 컬럼 값을 갱신할 수 없으며, 파티션 키 컬럼 값을 삭제하거나 삽입만 할 수 있습니다.

*해쉬 파티션*은 파티션된 테이블의 각 행의 배치를 결정하는 방법입니다. 이 방법은 다음과 같이 작동됩니다.

1. 해쉬 알고리즘은 파티션 키 값에 적용되고, 0과 4095 사이의 파티션 번호를 생성합니다.
2. 파티션 맵은 노드 그룹이 작성될 때 작성됩니다. 각 파티션 번호는 파티션 맵을 채우기 위해 원형식으로 계속 반복됩니다. 파티션 맵에 대해서는 114 페이지의 『파티션 맵』에서 자세한 내용을 참조하십시오.
3. 파티션 번호는 파티션 맵의 색인으로 사용됩니다. 파티션 맵의 해당 위치에 있는 숫자는 행이 저장된 데이터베이스 파티션의 번호입니다.

테이블 배열

둘 이상의 테이블이 특정 조회에 대한 응답에서 자주 데이터를 제공함을 알 수 있습니다. 이 경우, 사용자는 관련 데이터를 이들 테이블에서 가능하면 서로 가까운 곳에 위치지정하려고 합니다. 데이터베이스가 물리적으로 둘 이상의 데이터베이스 파티션으로 나누어지는 환경에서는 파티션된 테이블의 관련된 부분을 가능한 한 가까이 둘 수 있는 방법이 있어야 합니다. 바로 이러한 기능을 *테이블 배열*이라고 합니다.

테이블은 같은 노드 그룹에 저장될 경우와 파티션 키가 서로 호환성이 있는 경우에 공동 배치됩니다. 두 테이블을 동일한 노드 그룹에 배치할 경우, 공통된 파티션 맵을 사용하게 됩니다. 테이블은 다른 테이블 공간에 있습니다. 그러나 테이블 공간은 같은 노드 그룹과 연결되어야 합니다. 각 파티션 키의 해당 컬럼의 테이터 유형은 *파티션 호환 가능*한 것이어야 합니다. 파티션 호환성에 대해서는 『파티션 호환성』에서 자세한 내용을 참조하십시오.

DB2는 조인 또는 부속 조회를 위해 둘 이상의 테이블에 액세스할 때, 조인할 데이터가 동일한 데이터베이스 파티션에 위치지정되어 있는지를 인식하는 기능이 있습니다. 이 경우, DB2는 데이터베이스 파티션간에 데이터를 이동시키지 않고, 데이터가 저장된 데이터베이스 파티션에서 조인 또는 부속 조회가 수행되도록 선택할 수 있습니다. 데이터베이스 파티션에서 조인 또는 부속 조회를 수행하는 기능은 매우 성능을 높여주는 기능입니다. *관리 안내서: 성능*의 "조인"에서 자세한 내용을 참조하십시오.

파티션 호환성

파티션 키의 해당 컬럼의 기본 데이터 유형이 비교되며, *파티션 호환 가능*이 선언될 수 있습니다. 파티션 호환 가능한 데이터 유형은 동일한 값을 갖는 각 유형의 두 변수가 동일한 해싱 알고리즘에 의해 동일한 파티션 번호에 맵핑된다는 특성을 갖습니다.

파티션 호환성에는 다음과 같은 특성이 있습니다.

- 기본 데이터 유형은 동일한 기본 데이터 유형의 다른 유형과 호환됩니다.
- DATE, TIME 및 TIMESTAMP 데이터 유형에 내부 형식이 사용됩니다. 이들은 서로 호환되지 않으며, 아무것도 CHAR와 호환되지 않습니다.

- 파티션 호환성은 NOT NULL 또는 FOR BIT DATA 정의로 된 컬럼에 의해 영향받지 않습니다.
- 호환 가능한 데이터 유형의 널(NULL) 값은 동일하게 취급되며, 호환 가능하지 않은 데이터 유형은 동일하게 취급되지 않습니다.
- 사용자 정의 유형의 기본 데이터 유형이 파티션 호환성을 분석하는 데 사용됩니다.
- 파티션 키의 동일한 값의 소수는 스케일과 정밀도 면에서 다르더라도 동일하게 취급됩니다.
- 해싱 알고리즘은 문자열의 뒤 공백(CHAR, VARCHAR, GRAPHIC 또는 VARGRAPHIC)을 무시합니다.
- BIGINT, SMALLINT 및 INTEGER는 호환 가능한 데이터 유형입니다.
- REAL 및 FLOAT는 호환 가능한 데이터 유형입니다.
- 다른 길이의 CHAR 또는 VARCHAR는 호환 가능한 데이터 유형입니다.
- GRAPHIC 및 VARGRAPHIC은 호환 가능한 데이터 유형입니다.
- 파티션 호환성은 LONG VARCHAR, LONG VARGRAPHIC, CLOB, DBCLOB 및 BLOB 데이터 유형에는 적용되지 않는데, 파티션 키로서 지원되지 않기 때문입니다.

복제된 요약 테이블

*요약 테이블*은 테이블의 데이터를 판별하는 데에도 사용되는 조회에 의해 정의된 테이블입니다. 요약 테이블을 사용하여 조회 성능을 높일 수 있습니다. DB2가 조회의 일부를 요약 테이블을 사용하여 해석할 수 있다고 판단한 경우, 데이터베이스 관리 프로그램는 요약 테이블을 사용하기 위해 조회를 재작성합니다.

파티션된 데이터베이스 환경에서는 요약 테이블을 복제할 수 있습니다. *복제된 요약 테이블*을 사용하여 조회 성능을 높일 수 있습니다. 복제된 요약 테이블은 단일 구획 노드 그룹에서 작성될 수 있는 테이블에 근거하지만, 노드 그룹 내의 모든 데이터베이스 파티션에서 복제하려고 합니다. 복제된 요약 테이블을 작성하려면, CREATE TABLE문을 REPLICATED 키워드와 함께 호출하십시오.

요약 테이블에 대해서는 *관리 안내서: 구현*의 "요약 테이블 작성"에서 자세한 내용을 참조하십시오.

복제된 요약 테이블을 사용하면, 일반적으로 배열되지 않은 테이블을 공동 배열할 수 있습니다. 대형 사실 테이블과 소형 차원 테이블을 가지고 있는 조인의 경우, 복제된 요약 테이블은 특히 유용합니다. 필요한 추가 저장영역 및 모든 복제본을 갱신해야 할 경우의 부담을 최소화하려면, 복제되는 테이블은 작고 자주 갱신되지 않아야 합니다.

주: 또한, 자주 갱신되지 않는 대형 테이블을 복제하는 경우도 고려해야 하는데, 공동 배열을 통해 얻을 수 있는 성능상의 이점으로 인해 일회 복제시의 부담이 상쇄됩니다.

복제된 테이블을 정의하는 데 사용되는 부속 선택 절에 적절한 술어를 지정하면, 선택한 컬럼이나 선택한 행 중 한 가지 또는 둘다를 복제할 수 있습니다.

복제된 요약 테이블에 대해서는 *SQL 참조서*의 CREATE TABLE문을 참조하십시오. 조합 조인에 대해서는 *관리 안내서: 구현*의 "조합 조인"에서 자세한 내용을 참조하십시오.

테이블 공간 설계 및 선택

테이블 공간은 데이터베이스와 해당 데이터베이스에 저장된 테이블간의 간접성 레벨을 제공하는 저장영역 모델입니다. 테이블 공간은 노드 그룹에 상주합니다. 사용자는 데이터베이스의 위치와 테이블 데이터를 컨테이너에 직접 할당할 수 있습니다. (디렉토리 이름, 장치 이름 또는 파일 이름이 컨테이너로 사용될 수 있습니다.) 이로 인해 성능이 개선되고, 구성에 융통성이 있게 되며, 좀더 나은 무결성을 제공할 수 있습니다.

*관리 안내서: 구현*의 "테이블 공간 작성" 또는 "테이블 공간 변경"에서 테이블 공간 작성이나 변경에 대한 자세한 내용을 참조하십시오.

테이블 공간은 노드 그룹에 상주하기 때문에, 테이블을 수용하기 위해 선택된 테이블 공간은 노드 그룹의 데이터베이스 파티션에서 테이블 데이터가 분산되는 양식을 정의합니다. 단일 테이블 공간이 여러 개의 컨테이너에 걸쳐 있을 수 있습니다. 다중 컨테이너가(하나 이상의 테이블 공간으로부터) 같은 물리적 디스크(또는 드라이브)에서 작성될 수 있습니다. 성능을 높이려면, 각각의 컨테이너가 서로 다

른 디스크를 사용해야 합니다. 그림28은 데이터베이스 내의 테이블과 테이블 공간과의 관계, 데이터베이스와 연관된 컨테이너를 나타냅니다.

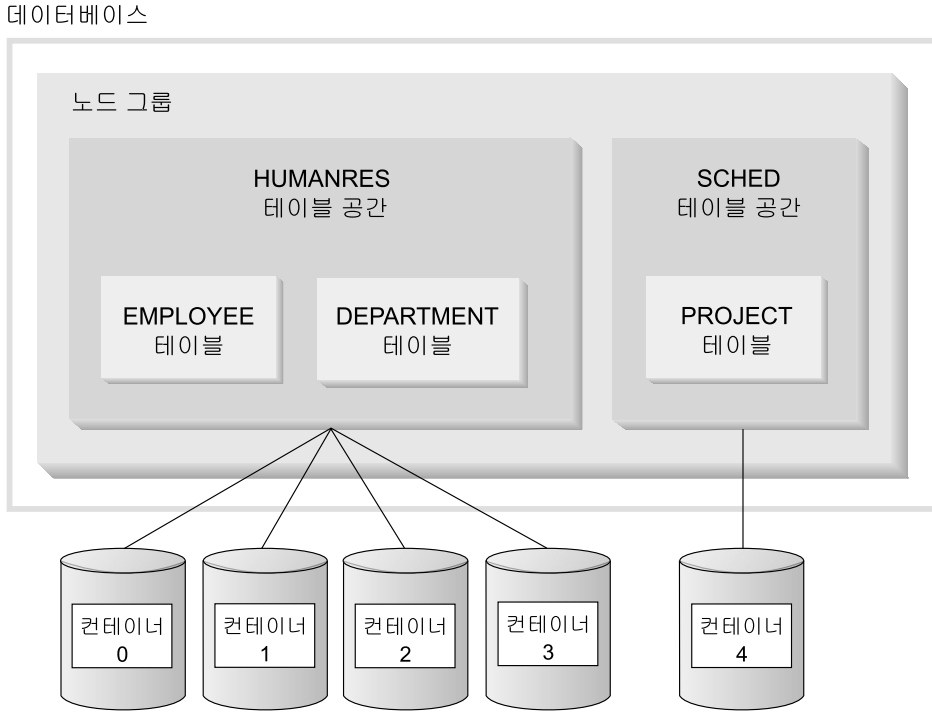

그림 28. 데이터베이스 내의 테이블 공간 및 테이블

EMPLOYEE 및 DEPARTMENT 테이블은 HUMANRES 테이블 공간 내에 있는데, 이는 컨테이너 0, 1, 2 및 3에 걸쳐 있습니다. PROJECT 테이블은 컨테이너 4의 SCHED 테이블 공간에 있습니다. 이 예에서는 별도의 디스크에 존재하는 각 컨테이너를 보여줍니다.

데이터베이스 관리 프로그램은 가능한 한 컨테이너간에 균형 있게 데이터를 로드하려고 노력합니다. 결과적으로, 모든 컨테이너가 데이터를 저장하는 데 사용됩니다. 다른 컨테이너를 사용하기 전에 데이터베이스 관리 프로그램이 컨테이너에 기록하는 페이지의 수를 *extent 크기*라고 합니다. 데이터베이스 관리 프로그램은 항상 첫 번째 컨테이너부터 테이블 데이터를 저장하지는 않습니다.

122 페이지의 그림29에서는 extent 크기가 두 개의 4 페이지이면서 각각 적은 수의 할당 extent를 가지고 있는 4개의 컨테이너가 있는 HUMANRES 테이블 공간을 보여줍니다.

DEPARTMENT 및 EMPLOYEE 테이블은 둘다 7 페이지를 가지고 있고 4개의 컨테이너 모두에 걸쳐 있습니다.

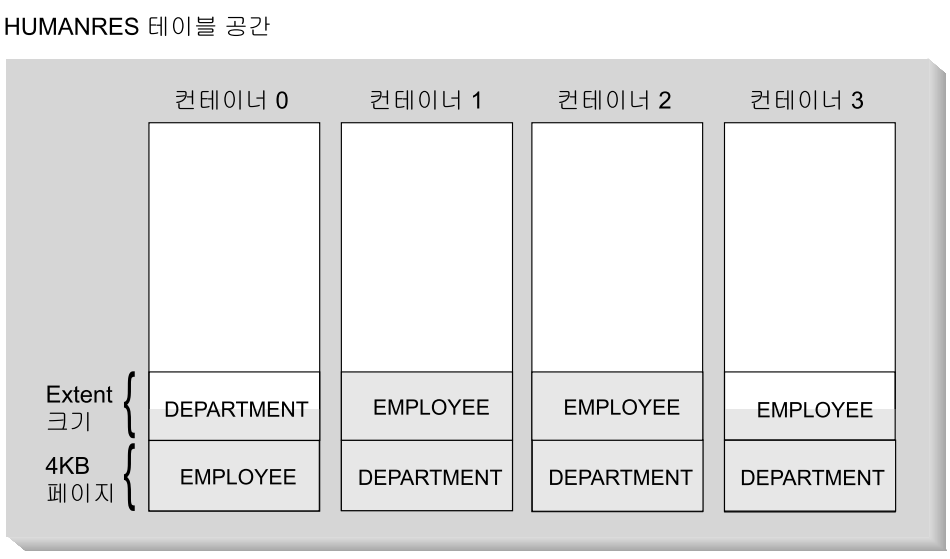

그림 29. 컨테이너 및 Extent

데이터베이스에는 최소한 세 개의 테이블 공간이 있어야 합니다.

- 하나의 *카탈로그 테이블 공간*. 여기에는 데이터베이스에 대한 모든 시스템 카탈로그 테이블이 들어 있습니다. 이 테이블 공간을 SYSCATSPACE라 하며 이는 삭제될 수 없습니다. IBMCATGROUP은 테이블 공간에 대한 기본 노드 그룹입니다.
- 하나 이상의 *사용자 테이블 공간*. 여기에는 모든 사용자 정의 테이블이 들어 있습니다. 기본적으로, 하나의 테이블 공간 USERSPACE1만이 작성됩니다.
  IBMDEFAULTGROUP은 이 테이블 공간에 대한 기본 노드 그룹입니다.

  테이블을 작성할 때 테이블 공간 이름을 지정하지 않으면 의도한 결과가 나오지 않을 수 있습니다. 테이블 공간 이름을 지정하지 않으면, 테이블은 다음 규칙에 따라 위치지정됩니다. 사용자 작성 테이블 공간이 있는 경우, 이 테이블용으로 충분한 가장 작은 페이지 *크기*를 가진 테이블 공간을 선택하십시오. 그렇지 않으면, 페이지 *크기*가 테이블용으로 충분한 경우 USERSPACE1을 사용하십시오. 충분히 큰 페이지 *크기*를 가진 테이블 공간이 없는 경우, 테이블은 작성되지 않습니다.

  테이블의 페이지 *크기*는 행 *크기*나 컬럼 수 중 하나로 판별됩니다. 행으로 허용 가능한 최대 길이는 테이블이 작성된 테이블 공간의 페이지 *크기*에 의해 좌우됩니다. 페이지 크기로 사용할 수 있는 값은 4KB(기본값), 8KB, 16KB 및 32KB입니다. 기본 테이블에

대한 페이지 *크기*가 하나인 테이블 공간을 사용하고, long 또는 LOB 데이터에 대한 페이지 *크기*가 다른 테이블 공간을 사용할 수 있습니다. DMS는 테이블 공간을 확장시키는 테이블을 지원하는 반면, SMS는 지원하지 않는다는 점을 기억하십시오. 컬럼의 수 또는 행 *크기*가 테이블 공간의 페이지 *크기*를 초과하면, 오류가 리턴됩니다(SQLSTATE 42997).

- 임시 테이블이 들어 있는 하나 이상의 *임시 테이블 공간*. 임시 테이블 공간은 *시스템 임시 테이블 공간* 또는 *사용자 임시 테이블 공간*일 수 있습니다. 데이터베이스는 최소한 하나의 시스템 임시 테이블 공간을 가져야 합니다. 기본값으로 TEMPSPACE1이라고 하는 하나의 시스템 임시 테이블 공간은 데이터베이스 작성 시간에 작성됩니다. IBMTEMPGROUP은 이 테이블 공간에 대한 기본 노드 그룹입니다. 사용자 임시 테이블 공간은 데이터베이스 작성시 기본값으로 작성되지 *않습니다*.

  데이터베이스가 두 개 이상의 임시 테이블 공간을 사용하면서 새 임시 오브젝트가 필요한 경우, 최적화 알고리즘은 이 오브젝트에 적절한 페이지 *크기*를 선택합니다. 그런 다음, 이 오브젝트는 이 페이지 *크기*로 임시 테이블 공간에 할당됩니다. 페이지 *크기*가 같은 임시 테이블 공간이 두 개 이상인 경우, 테이블 공간은 라운드 로빈 방식으로 선택됩니다.

  4KB 기본값보다 더 큰 페이지 공간으로 정의된 테이블 공간에 있는 테이블에 대해 조회를 실행하는 경우(예를 들어, 1012 컬럼에서의 ORDER BY), 이 중 일부는 실패할 수 있습니다. 이것은 대형 페이지 *크기*로 정의된 임시 테이블 공간의 부족으로 발생합니다. 대형 페이지 *크기*(8KB, 16KB 또는 32KB)의 임시 테이블 공간을 작성해야 합니다. 모든 DML문은 사용자 데이터의 가장 큰 페이지 *크기*와 동일한 페이지 *크기*의 임시 테이블 공간이 있지 않는 한 실패할 수 있습니다.

  대부분의 사용자 테이블 공간에서 사용되는 페이지 *크기*와 동일한 페이지 *크기*의 SMS 임시 테이블 공간 하나를 정의해야 합니다. 이는 일반적인 환경 및 워크로드에 적합해야 합니다. 또한 138 페이지의 『임시 테이블 공간에 대한 권고사항』을 참조하십시오.

파티션된 데이터베이스 환경에서, 카탈로그 노드에는 세 개의 테이블 공간 모두가 포함되고 다른 데이터베이스 파티션에는 TEMPSPACE1과 USERSPACE1이 각각 포함됩니다.

둘다 단일 데이터베이스에서 사용될 수 있는 두 가지 유형의 테이블 공간이 있습니다.

- 124 페이지의 『시스템 관리 공간』: 운영 체제의 파일 관리 프로그램은 저장영역 공간을 제어합니다.
- 129 페이지의 『데이터베이스 관리 공간(DMS) 테이블 공간』: 데이터베이스 관리 프로그램은 저장영역 공간을 제어합니다.

시스템 관리 공간

시스템 관리 공간(SMS) 테이블 공간에서 운영 체제의 파일 시스템 관리 프로그램은 테이블이 저장될 공간을 할당하고 관리합니다. 일반적으로, 저장영역 모델은 파일 시스템 공간에 저장된, 테이블 오브젝트를 나타내는 여러 파일로 구성됩니다. 사용자는 파일의 위치를 결정하며, DB2는 파일의 이름을 제어하고, 파일 시스템은 이를 관리합니다. 각 파일에 기록되는 데이터의 양을 제어함으로써, 데이터베이스 관리 프로그램은 테이블 공간 컨테이너에 데이터를 균등하게 분배합니다. SMS 테이블 공간이 기본 테이블 공간입니다.

각 테이블에는 이와 연관된 SMS 실제 파일이 적어도 하나씩 들어 있습니다. 이들 파일의 목록 및 내용의 설명은 127 페이지의 『SMS 실제 파일』에서 자세한 내용을 참조하십시오.

SMS 테이블 공간에서, 파일은 오브젝트가 커질 때마다 한번에 한 페이지씩 확장됩니다. 삽입 기능을 개선해야 할 경우, 다중 페이지 파일 할당 기능을 작동시키는 방안을 고려할 수 있습니다. 이렇게 하면, 시스템은 한번에 두 페이지 이상씩 파일을 할당하거나 확장할 수 있습니다. 다중 페이지 파일 할당 기능을 작동시키려면 **db2empfa**를 실행해야 합니다. 파티션이 나누어진 데이타베이스에서, 이 유틸리티는 각 데이타베이스 파티션에서 실행되어야 합니다. 일단 다중 페이지 파일 할당 기능 작동을 시도하면, 해제할 수 없습니다. **db2empfa**에 대해서는 *Command Reference*에서 자세한 내용을 참조하십시오.

CREATE DATABASE 명령 또는 CREATE TABLESPACE문에서 MANAGED BY SYSTEM 옵션을 사용하여 명시적으로 SMS 테이블 공간을 정의해야 합니다. SMS 테이블 공간을 설계할 때 두 가지 키 인수를 고려해야 합니다.

- 테이블 공간에 대한 컨테이너.

  사용자가 테이블 공간용으로 사용하려는 컨테이너 수를 지정해야 합니다. SMS 테이블 공간이 작성된 후에는 컨테이너를 추가하거나 삭제할 수 없기 때문에 사용하려는 컨테이너를 모두 식별하는 것은 매우 중요합니다. 파티션된 데이터베이스 환경에서, 새로운 파티션이 SMS 테이블 공간용 노드 그룹에 추가될 때, ALTER TABLESPACE문을 새로운 파티션에 대한 컨테이너를 추가하는 데 사용할 수 있습니다.

SMS 테이블 공간에 사용된 각각의 컨테이너는 절대 또는 상대 디렉토리 이름을 식별합니다. 이들 디렉토리는 각각 서로 다른 파일 시스템(또는 물리적 디스크)에 위치될 수 있습니다. 테이블 공간의 최대 *크기*는 다음과 같이 제한될 수 있습니다.

```
number of containers * (maximum file system size
   supported by the operating system)
```

이 공식은 각 컨테이너에 맵핑된 파일 시스템이 명확하고, 각 파일 시스템마다 사용 가능한 최대 공간이 있음을 전제로 합니다. 물리적으로 이러한 경우가 발생하지 않을 수도 있고 최대 테이블 공간 *크기*가 더 작을 수도 있습니다.

주: 컨테이너를 정의할 때에는 주의해야 합니다. 컨테이너에 기존 파일 또는 디렉토리가 있는 경우, 오류(SQL0298N)가 리턴됩니다.

- 테이블 공간에 대한 extent *크기*.

  extent *크기*는 테이블 공간이 작성될 때에만 지정될 수 있습니다. 이는 나중에 변경될 수 없기 때문에 extent *크기*에 적절한 값을 선택하는 것이 중요합니다. 137 페이지의 『extent *크기* 선택』에서 자세한 내용을 참조하십시오.

  테이블 공간을 작성할 때 extent *크기*를 지정하지 않으면, 데이터베이스 관리 프로그램은 *dft_extent_sz* 데이터베이스 구성 매개변수(이 매개변수에 관한 자세한 내용은 *관리 안내서: 성능* 참조)로 정의된 기본 extent *크기*를 사용하여 테이블 공간을 작성합니다. 이 구성 매개변수는 데이터베이스가 작성될 때 제공된 정보를 근거로 하여 초기에 설정됩니다. *dft_extent_sz* 매개변수가 CREATE DATABASE 명령에 지정되어 있지 않을 경우, 기본 extent *크기*는 32에 설정됩니다.

컨테이너 수에 대한 적절한 값과 테이블 공간에 대한 extent *크기*를 선택하려면, 다음을 이해해야 합니다.

- 운영 체제가 논리 파일 시스템의 *크기*에 대해 가지고 있는 한계

  예를 들어, 일부 운영 체제의 한계값은 2GB입니다. 따라서, 64GB 테이블 오브젝트를 원할 경우, 적어도 32개의 컨테이너가 필요합니다.

  테이블 공간을 작성할 때, 다른 파일 시스템에 상주하는 컨테이너를 지정할 수 있으며, 그 결과 데이터베이스에 저장될 수 있는 데이터의 양이 증가됩니다.

- 데이터베이스 관리 프로그램이 테이블 공간과 연관된 데이터 파일 및 컨테이너를 관리하는 방법

  첫 번째 테이블 데이터 파일(SQL00001.DAT)이 테이블 공간에 지정된 첫 번째 컨테이너에서 작성되면, 이 파일은 extent *크기*로 커집니다. 이 *크기*에 이르면, 데이터베이스 관리 프로그램이 그 다음 컨테이너에 있는 SQL00001.DAT에 데이터를 기록합니다. 이 프로세스는 모든 컨테이너가 SQL00001.DAT 파일을 포함할 때까지 계속됩니다. 이 때 데이터베이스 관리 프로그램은 첫 번째 컨테이너로 리턴합니다. 이 프로세스(*스트라이핑*이라고 함)는 컨테이너가 가득 차거나 (SQL0289N) 또는 운영 체제로부터 더 이상의 공간을 할당할 수 없을 때까지 (디스크 공간 부족 오류) 컨테이너 디렉토리를 통해 계속됩니다. 스트라이핑은 색인(SQL*nnnnn*.INX), long 필드(SQL*nnnnn*.LF)와 LOB(SQL*nnnnn*.LB 및 SQL*nnnnn*.LBA) 파일에 대해서도 사용됩니다.

  주: 컨테이너가 가득차게 되면 SMS 테이블 공간도 가득찹니다. 따라서, 각 컨테이너에 동일한 양의 공간을 할당하는 것이 중요합니다.

  데이터를 좀더 균등하게 컨테이너에 배분하기 위해 데이터베이스 관리 프로그램은 컨테이너의 번호를 갖는 테이블 ID(위의 예에서 1)를 사용하여 처음 사용할 컨테이너를 결정합니다. 컨테이너는 0에서 시작하여 순차적으로 번호를 매깁니다.

  SMS 테이블 공간에 사용된 파일에 대해서는 127 페이지의 『SMS 실제 파일』에서 자세한 내용을 참조하십시오.

SMS 실제 파일

다음 파일은 SMS 테이블 공간 디렉토리 컨테이너에 있습니다.

파일 이름 **설명**

SQLTAG.NAM

각 컨테이너 서브디렉토리에 이와 같은 파일 중 하나가 들어 있으며, 이들 파일은 사용자가 데이터베이스에 연결될 때 데이터베이스가 완전하고 일관성 있음을 확인하기 위해 데이터베이스 관리 프로그램에 의해 사용됩니다.

SQLxxxxx.DAT

테이블 파일. LONG VARCHAR, LONG VARGRAPHIC, BLOB, CLOB 또는 DBCLOB 데이터를 제외한 모든 테이블이 행이 여기에 저장됩니다.

SQLxxxxx.LF

LONG VARCHAR 또는 LONG VARGRAPHIC 데이터(″long 필드 데이터″라고도 함)가 들어 있는 파일. 이 파일은 LONG VARCHAR 또는 LONG VARGRAPHIC 컬럼이 테이블에 존재하는 경우에만 작성됩니다.

SQLxxxxx.LB

BLOB, CLOB 또는 DBCLOB 데이터(″LOB 데이터″라고도 함)가 들어 있는 파일. 이 파일은 BLOB, CLOB 또는 DBCLOB 컬럼이 테이블에 존재하는 경우에만 작성됩니다.

SQLxxxxx.LBA

SQL*xxxxx*.LB 파일에 관한 할당 및 사용 가능한 공간 정보가 들어 있는 파일.

SQLxxxxx.INX

테이블에 대한 색인 파일. 해당 테이블에 대한 모든 색인이 이 파일 하나에 저장됩니다. 이는 색인이 정의된 경우에만 작성됩니다.

주: 색인이 삭제되면, 색인 파일이 삭제될 때까지 공간이 색인(.INX) 파일로부터 물리적으로 해제되지 않습니다. 색인 파일은 테이블의 모든 색인이 삭제(및 확약)되거나 테이블이 재구

성되면 삭제됩니다. 색인 파일이 삭제되지 않으면, 공간은 일단 삭제가 확약되어 해제된 것으로 표시되었다가, 후에 색인 작성 또는 색인 유지보수에 다시 사용됩니다.

SQLxxxxx.DTR

DAT 파일의 재구성에 대한 임시 데이터 파일. 테이블을 재구성할 때, 재구성 유틸리티(REORG TABLE 명령을 통해)는 시스템 임시 테이블 공간 중 하나의 공간에 있는 테이블을 작성합니다. 이러한 임시 테이블 공간은 사용자 정의 테이블용으로 사용된 컨테이너가 아닌 다른 컨테이너를 사용하도록 정의될 수 있습니다.

SQLxxxxx.LFR

LF 파일의 재구성에 대한 임시 데이터 파일. 테이블을 재구성할 때, 재구성 유틸리티(REORG TABLE 명령을 통해)는 시스템 임시 테이블 공간 중 하나의 공간에 있는 테이블을 작성합니다. 이러한 임시 테이블 공간은 사용자 정의 테이블용으로 사용된 컨테이너가 아닌 다른 컨테이너를 사용하도록 정의될 수 있습니다.

SQLxxxxx.RLB

LB 파일의 재구성에 대한 임시 데이터 파일. 테이블을 재구성할 때, 재구성 유틸리티(REORG TABLE 명령을 통해)는 시스템 임시 테이블 공간 중 하나의 공간에 있는 테이블을 작성합니다. 이러한 임시 테이블 공간은 사용자 정의 테이블용으로 사용된 컨테이너가 아닌 다른 컨테이너를 사용하도록 정의될 수 있습니다.

SQLxxxxx.RBA

LBA 파일의 재구성에 대한 임시 데이터 파일. 테이블을 재구성할 때, 재구성 유틸리티(REORG TABLE 명령을 통해)는 시스템 임시 테이블 공간 중 하나의 공간에 있는 테이블을 작성합니다. 이러한 임시 테이블 공간은 사용자 정의 테이블용으로 사용된 컨테이너가 아닌 다른 컨테이너를 사용하도록 정의될 수 있습니다.

주:

1. 이들 파일을 직접 변경하지 *마십시오*. 문서화된 API를 사용하거나, 명령행 처리기 및 제어 센터를 비롯하여 API를 구현하여 도구에 의해서만, 간접적으로 액세스될 수 있습니다.

2. 위의 파일을 이동시키지 마십시오.
3. 위의 파일을 제거하지 마십시오.
4. 데이터베이스 또는 테이블 공간을 백업하는 유일한 방법은 **sqlubkp**(백업 데이터베이스) API를 사용하는 것입니다(해당 API의 명령행 처리기 및 제어 센터 구현 포함).

데이터베이스 관리 공간(DMS) 테이블 공간

DMS(데이터베이스 관리 공간) 테이블 공간에서, 데이터베이스 관리 프로그램은 저장영역 공간을 제어합니다. 저장영역 모델은 DB2에 의해 그 공간이 관리되는 한정된 수의 장치로 구성됩니다. 관리자는 어떤 장치를 사용할 것인지 결정하고, DB2는 장치의 공간을 관리합니다. 테이블 공간은 본질적으로 데이터베이스 관리 프로그램의 요구를 최대한 충족시키기 위해 설계된 특수 목적의 파일 시스템을 구현한 것입니다. 테이블 공간 정의에는 데이터가 저장될 수 있는 테이블 공간에 속해 있는 장치 또는 파일의 목록이 들어 있습니다.

사용자 정의 테이블 및 데이터가 있는 DMS 테이블 공간은 다음과 같이 정의될 수 있습니다.

- 정상 테이블 및 색인 데이터를 저장할 *일반* 테이블 공간
- Long 필드 또는 LOB 데이터를 저장할 *long* 테이블 공간

DMS 테이블 공간 및 컨테이너를 설계할 때 다음을 고려해야 합니다.

- 데이터베이스 관리 프로그램은 스트라이핑을 사용하여 모든 컨테이너에 걸쳐 데이터 분배가 고르게 되도록 합니다.
- 일반 테이블 공간 공간의 최대 *크기*는 4KB 페이지의 경우 64GB, 8KB 페이지의 경우 128GB, 32KB 페이지의 경우 512GB입니다. Long 테이블 공간의 최대 *크기*는 2TB입니다.
- SMS 테이블 공간과는 달리, DMS 테이블 공간을 구성하는 컨테이너는 *크기*가 동일할 필요가 없습니다. 그러나 이것은 컨테이너에서 고르지 못한 스트라이핑과 부분 최적 성능을 초래하므로 보통 권장되지 않습니다. 컨테이너가 가득차면, DMS 테이블 공간은 다른 컨테이너에서 사용 가능한 공간을 사용합니다.
- 공간은 사전 할당되기 때문에 먼저 공간이 사용 가능한 상태가 되어야 테이블 공간을 작성할 수 있습니다. 장치 컨테이너를 사용할 때 컨테이너 정의를 위한

충분한 공간과 함께 장치도 존재해야 합니다. 각 장치는 장치에 대해 정의된 하나의 컨테이너만을 가질 수 있습니다. 공간의 낭비를 피하기 위해서는 장치 크기와 컨테이너 크기가 동일해야 합니다. 예를 들어, 장치에 5 000 페이지가 할당되고 장치 컨테이너에 3 000 페이지가 할당되도록 정의된 경우에는 장치의 2 000 페이지는 사용되지 않습니다.

- 모든 컨테이너의 하나의 페이지는 오버헤드용으로 예약된 것이며, 나머지 페이지는 한번에 하나의 extent씩 사용됩니다. 전체 extent만 사용되므로, 최적의 공간 관리를 위해 다음 공식을 사용하여 컨테이너를 할당할 때 사용할 적절한 크기를 판별할 수 있습니다.

  ```
  (extent_size * n) + 1
  ```

  여기서 *extent_size*는 테이블 공간의 각 extent 크기이고, *n*은 컨테이너에 저장하려는 extents의 수입니다.
- 테이블 공간의 세 개의 extent는 오버헤드용으로 예약된 것입니다.
- 최소한 두 개의 extent가 사용자 테이블 데이터를 저장하는 데 필요합니다. (이들 extent는 하나의 테이블에 대한 일반 데이터용으로, 자체 extent가 필요로 하는 색인, long 필드 또는 대형 오브젝트(LOB) 데이터용이 아닙니다.)
- 장치 컨테이너는 물리적 볼륨이 아닌 ″문자 특수 인터페이스″와 함께 논리 볼륨을 사용해야 합니다.
- DMS 테이블 공간을 가진 장치 대신 파일을 사용할 수 있습니다. 파일과 장치 사이에 작동상의 차이는 없지만, 파일은 파일 시스템과 연관된 런타임 오버헤드 때문에 파일의 효율성이 떨어질 수 있습니다. 파일은 다음과 같은 경우에 유용합니다.
  - 장치가 직접 지원되지 않는 경우
  - 장치를 사용할 수 없는 경우
  - 최대 성능이 필요하지 않은 경우
  - 장치를 설정하고자 하지 않는 경우
- 워크로드에는 LOB 또는 LONG VARCHAR가 관련되어 있으며 파일 시스템 캐쉬에서 성능 이점을 얻을 수 있습니다. LOB 및 LONG VARCHAR는 DB2 버퍼 풀로 버퍼되지 않음을 기억하십시오.

- 일부 운영 체제에서는 *크기*가 2GB를 넘는 물리적 장치를 설치할 수 있습니다. 물리적 장치를 다중 논리 장치로 파티션하여 운영 체제에서 허용되는 *크기*보다 큰 컨테이너가 없도록 하는 것을 고려해야 합니다.

DMS 테이블 공간에 컨테이너 추가

ALTER TABLESPACE문은 기존의 테이블 공간에 컨테니너를 추가하여 저장영역을 넓힐 수 있습니다. 그런 다음, 테이블 공간의 내용이 모든 컨테이너간에 걸쳐 재조정됩니다. 재조정하는 동안에도 테이블 공간에 대한 액세스는 제한되지 않습니다. 둘 이상의 컨테이너를 추가해야 할 경우에는, 하나의 ALTER TABLESPACE문 또는 동일한 트랜잭션 내에 동시에 추가하면 데이터베이스 관리 프로그램이 컨테이너를 두 번 이상 재조정할 필요가 없도록 할 수 있습니다.

LIST TABLESPACE CONTAINERS 또는 LIST TABLESPACES 명령을 사용하여 테이블 공간의 컨테이너가 얼마나 채워졌는지 점검할 수 있습니다. 기존의 컨테이너가 거의 또는 완전히 채워지기 전에, 새 컨테이너를 추가해야 합니다. 재조정이 완료되어야만, 모든 컨테이너의 새 공간을 사용할 수 있습니다.

기존의 컨테이너보다 작은 컨테이너를 추가하면 데이터가 치우쳐 분산됩니다. 프리페치 데이터와 같은 병렬 입출력 조작이 일어나서 같은 *크기*의 컨테이너에서 달리 실행하는 것보다 비효율적으로 실행됩니다.

테이블 공간 설계 고려사항

이 절에서는 다음 주제를 다룹니다.

- 132 페이지의『테이블 공간 입출력 고려사항』
- 134 페이지의『테이블 공간을 버퍼 풀에 맵핑』
- 135 페이지의『테이블 공간을 노드 그룹에 맵핑』
- 135 페이지의『테이블을 테이블 공간에 맵핑』
- 137 페이지의『extent *크기* 선택』
- 138 페이지의『임시 테이블 공간에 대한 권고사항』
- 140 페이지의『카탈로그 테이블 공간에 대한 권고사항』
- 141 페이지의『워크로드 고려사항』
- 142 페이지의『SMS 또는 DMS 테이블 공간 선택』

- 143 페이지의『RAID 장치에 데이터 배치시 성능 최적화』

테이블 공간 입출력 고려사항

테이블 공간의 유형과 설계는 테이블 공간에서 이루어진 입출력의 효율성을 결정합니다. 다음은 테이블 공간의 설계와 사용을 둘러싼 추가 문제를 고려하기 전에 이해해야 할 개념입니다.

큰 블록 읽기 한 번의 요청에 몇 페이지(보통 extent)가 검색되는 읽기 방식. 동시에 몇 페이지를 읽는 것은 각 페이지를 별도로 읽는 것보다 더 효율적입니다.

프리페치 조회가 페이지를 참조하기 전에 페이지를 읽는 방식. 전반적인 목적은 응답 시간을 줄이는 것입니다. 조회 실행시 페이지 프리페치가 비동기식으로 이루어질 수 있을 경우 응답 시간을 줄일 수 있습니다. CPU 또는 입출력 서브시스템이 최대 기능을 내며 작동할 때 응답 시간이 최적이 됩니다.

페이지 정리 페이지를 읽고 변경할 때, 이들 페이지는 데이터베이스 버퍼 풀에 쌓이게 됩니다. 페이지를 읽을 때, 버퍼 풀 페이지로 읽어들여야 합니다. 버퍼 풀이 변경된 페이지로 가득 찰 경우, 새 페이지를 읽기 전에 이들 변경된 페이지 중 하나가 디스크에 기록되어야 합니다. 버퍼 풀이 가득 차지 않게 하기 위해, 페이지 정리자 에이전트는 변경된 페이지를 기록하여 장래 읽기 요청을 위해 버퍼 풀 페이지가 사용될 수 있도록 합니다.

DB2는 큰 블록 읽기가 도움이 될 때마다 이를 수행합니다. 보통, 완전히 순차적이거나 부분적으로 순차적이라는 특성을 갖는 데이터를 검색할 때 큰 블록 읽기가 발생합니다. 한번에 읽기 조작하는 데이터의 양은 extent 크기에 따라 다릅니다. extent 크기가 클수록, 더 많은 페이지를 한번에 읽을 수 있습니다.

extent가 디스크에 저장되는 양식은 입출력 효율성에 영향을 줍니다. 장치 컨테이너를 사용하는 DMS 테이블 공간에서, 데이터는 디스크에서 연속적으로 존재하며 최소한의 찾기 시간과 디스크 대기시간으로 읽을 수 있습니다. 그러나 파일이 사용 중일 경우, 데이터는 파일 시스템에 의해 나누어져 둘 이상의 디스크 위치에 저장됩니다. 파일이 한번에 한 페이지씩 확장되어 분할이 더 잘 일어날 수 있는 SMS 테이블 공간을 사용할 때 이러한 현상이 자주 발생할 수 있습니다. DMS 테이블

공간에서 사용되는 큰 파일을 사전에 할당하면 디스크에 인접하며, 특히 파일이 정리 파일 공간에 할당되는 경우에 해당됩니다.

CREATE TABLESPACE문의 PREFETCHSIZE 매개변수를 조정하여 프리페치의 등급을 제어할 수 있습니다. (데이터베이스의 모든 테이블 공간의 기본값은 *dft_prefetch_sz* 데이터베이스 구성 매개변수에 의해 설정됩니다.) PREFETCHSIZE 매개변수는 프리페치가 트리거될 때마다 얼마나 많은 페이지를 읽을 것인지 DB2에 알립니다. CREATE TABLESPACE문에서 PREFETCHSIZE를 여러 EXTENTSIZE 매개변수에 설정함으로써, 여러 extent를 병렬로 읽도록 할 수 있습니다. (데이터베이스의 모든 테이블 공간의 기본값은 *dft_extent_sz* 데이터베이스 구성 매개변수에 의해 설정됩니다.) EXTENTSIZE 매개변수는 다음 컨테이너로 건너뛰기 전에 컨테이너에 쓰여지는 4KB 페이지의 수를 지정합니다.

예를 들어, 세 개의 장치에 사용된 테이블 공간이 있다고 가정하십시오. PREFETCHSIZE를 EXTENTSIZE의 세 배로 설정할 경우, DB2는 각 장치로부터 병렬로 큰 블록 읽기를 수행하여, 입출력 양을 크게 늘릴 수 있습니다. 이 경우, 각 장치는 별도의 물리적 장치이며, 제어기는 각 장치로부터의 데이터 스트림을 처리할 만큼 충분한 대역폭을 가지고 있는 것으로 가정됩니다. DB2는 조회 속도, 버퍼 풀 사용 및 기타 요소에 의거하여 런타임 프리페치 매개변수를 동적으로 조정할 수 있음에 유의하십시오.

일부 파일 시스템은 자체의 프리페치(AIX의 JFS(Journaled File System)의 경우처럼)를 사용합니다. 어떤 경우에는 파일 시스템 프리페치는 DB2 프리페치보다 공격적으로 설정됩니다. 이 결과, 파일 컨테이너의 SMS 및 DMS 테이블 공간에 대한 프리페치는 장치를 통한 DMS 테이블 공간의 프리페치보다 우수해지게 할 수 있습니다. 파일 시스템에서 일어나는 추가 프리페치 레벨 결과와 같기 때문에 오해의 소지가 있습니다. DMS 테이블 공간은 어떤 동등한 구성보다 기능이 우수합니다.

프리페치(또는 읽기)를 효과적으로 수행하려면, 데이터를 읽어들일 빈 버퍼 풀 페이지가 충분히 있어야 합니다. 예를 들어, 테이블 공간으로부터 세 개의 extent를 읽어들이고, 읽히고 있는 각 페이지에 대해 변경된 페이지는 버퍼 풀에서 기록되는 병렬 프리페치 요청이 있을 수 있습니다. 프리페치 요청은 조회를 계속할 수 없는 지점까지 느려질 수 있습니다. 프리페치 요청을 충족시킬 만큼 충분한 수의 페

이지 정리자가 구성되어 있어야 합니다. 데이터베이스가 사용하는 각 디스크에 대해 적어도 하나의 페이지 정리자가 정의되어야 합니다. 이 주제에 대해서는 *관리 안내서: 성능*에서 자세한 내용을 참조하십시오.

테이블 공간을 버퍼 풀에 맵핑

각 테이블 공간은 특정 버퍼 풀과 연관됩니다. 기본 버퍼 풀은 IBMDEFAULTBP입니다. 다른 버퍼 풀이 테이블 공간과 연관될 경우, 버퍼 풀이 반드시 존재해야 하며(이것은 CREATE BUFFERPOOL문으로 정의됨), 테이블 공간이 작성될 때(CREATE TABLESPACE문 사용) 연관 관계가 정의됩니다. 테이블 공간과 버퍼 풀의 연관 관계는 ALTER TABLESPACE문을 사용하여 변경할 수 있습니다.

둘 이상의 버퍼 풀이 있으면, 사용자는 데이터베이스에 의해 사용된 메모리를 구성하여 전체 성능을 높일 수 있습니다. 예를 들어, 둘 이상의 대형 테이블을 사용자가 임의로 액세스하는 테이블 공간의 경우에 버퍼 풀의 *크기*를 제한할 수 있는데, 데이터 페이지 캐쉬가 도움이 되지 않기 때문입니다. 온라인 거래 응용프로그램의 테이블 공간은 더 큰 버퍼 풀과 연관될 수 있으므로, 응용프로그램이 사용하는 데이터 페이지는 더 오래 캐쉬되어, 더 빠른 응답 시간을 가져올 수 있습니다. 새로운 버퍼 풀을 구성할 때에는 주의해야 합니다. 이 주제에 대해서는 *관리 안내서: 성능*의 "데이터베이스 버퍼 풀 관리"를 참조하십시오.

주: 데이터베이스에서 8KB, 16KB 또는 32KB의 페이지 *크기*가 필요하다고 판별한 경우, 이들 페이지 *크기* 중 하나로 테이블 공간이 같은 페이지 *크기*의 버퍼 풀에 맵되어야 합니다.

모든 버퍼 풀이 사용할 수 있는 저장영역은 데이터베이스 관리 프로그램이 데이터베이스를 시작할 때에도 사용할 수 있어야 합니다. 정의된 모든 버퍼 풀에 필요한 저장영역을 DB2가 확보하지 못하면, 데이터베이스 관리 프로그램은 기본 버퍼 풀(각각 4KB, 8KB, 16KB 및 32KB 페이지 *크기*)을 시동하고 경고를 발행합니다.

파티션된 데이터베이스 환경에서, 데이터베이스의 모든 파티션에 대해 동일한 *크기*의 버퍼 풀을 작성할 수 있습니다. 또한, 다른 파티션에서 다른 *크기*의 버퍼 풀을 작성할 수도 있습니다. CREATE BUFFERPOOL문에 대해서는 *SQL 참조서*에서 자세한 내용을 참조하십시오.

테이블 공간을 노드 그룹에 맵핑

파티션된 데이터베이스 환경에서, 각 테이블 공간은 특정 노드 그룹과 연관됩니다. 이 경우, 테이블 공간의 특성이 노드 그룹의 각 노드에 적용될 수 있습니다. 노드 그룹은 반드시 존재해야 하며(이것은 CREATE NODEGROUP문으로 정의됨), 테이블 공간이 작성될 때(CREATE TABLESPACE문 사용) 테이블 공간과 노드 그룹간의 연관 관계가 정의됩니다.

테이블 공간과 노드 그룹의 연관 관계를 ALTER TABLESPACE문을 사용하여 변경할 수 없습니다. 노드 그룹 안에서 개별 파티션용 테이블 공간 스펙만 변경할 수 있습니다. 단일 파티션 환경에서, 각 테이블 공간이 기본 노드 그룹과 연관됩니다. 임시 테이블 공간이 정의되는 경우를 제외하고는 테이블 공간을 정의할 때 기본 노드 그룹은 IBMDEFAULTGROUP이며, IBMTEMPGROUP이 사용됩니다. CREATE NODEGROUP문에 대해서는 *SQL 참조서*에서 자세한 내용을 참조하십시오. 노드 그룹 및 물리적 데이터베이스 설계에 대해서는 111 페이지의 『노드 그룹 설계』에서 자세한 내용을 참조하십시오.

테이블을 테이블 공간에 맵핑

테이블을 테이블 공간에 맵하는 방법을 판별할 때 다음을 고려해야 합니다.

- 테이블의 파티션

  최소한, 선택한 테이블 공간은 사용자가 원하는 파티션이 있는 노드 그룹에 있어야 합니다.

- 테이블에 있는 데이터의 양

  여러 소형 테이블을 한 테이블 공간에 저장하기로 한 경우, 해당 테이블 공간에 대해 SMS를 사용하십시오. 입출력 및 공간 관리가 효율적이라는 DMS의 장점은 작은 테이블의 경우 그다지 중요하지 않습니다. 작은 테이블의 경우에는 한 번에 한 페이지씩, 필요할 때에만 공간을 할당할 수 있는 SMS가 보다 매력적입니다. 테이블 중 하나의 테이블이 더 *크거나* 테이블의 데이터에 액세스를 더 빨리 해야 하는 경우, 작은 extent *크기*의 DMS 테이블 공간이 고려되어야 합니다.

  각각의 대형 테이블에 대해 개별적인 테이블 공간을 사용하고 모든 소형 테이블을 단일 테이블 공간으로 그룹화하려는 경우도 있습니다. 이러한 분리를 통해 테이블 공간 사용에 근거한 적절한 extent *크기*를 선택할 수 있습니다. (137 페이지의 『extent *크기* 선택』에서 자세한 내용을 참조하십시오.)

- 테이블에 있는 데이터의 유형

  예를 들어, 가끔씩 사용되는 실행기록 데이터가 들어 있는 테이블이 있는데, 일반 사용자가 기꺼이 이 데이터에 대한 조회의 응답을 좀더 오랜 시간 기다리려고 하는 경우도 있습니다. 이러한 상황에서, 사용자는 실행기록 테이블에 대해 서로 다른 테이블 공간을 사용하고, 액세스 속도가 좀더 느리고 좀더 저렴한 가격의 물리적 장치에 이 테이블 공간을 할당할 수 있습니다.

  또한 데이터 고가용성과 빠른 응답 시간을 요구하는 필수 테이블 일부를 식별할 수도 있습니다. 이러한 중요한 데이터 요구사항을 지원하는 데 도움을 줄 수 있는 빠른 물리적 장치에 할당된 테이블 공간 안에 이들 테이블을 위치시킬 수 있습니다.

  DMS 테이블 공간을 사용할 때 테이블을 세 가지 서로 다른 테이블 공간에 분배할 수 있습니다. 하나는 색인 데이터를 위한 것이고, 또다른 하나는 LOB 및 long 필드 데이터를 위한 것이며, 마지막 하나는 일반 테이블 데이터를 위한 것입니다. 이는 사용자가 데이터에 가장 적합한 테이블 공간 특성과 테이블 공간을 지원하는 물리적 장치를 선택할 수 있도록 합니다. 예를 들어, 사용할 수 있는 장치 중 가장 빠른 장치에 색인 데이터를 위치시키게 되면, 성능면에서 매우 많이 개선될 수 있습니다. DMS 테이블 공간에 테이블을 분할하는 경우, 롤 포워드 복구를 사용할 수 있으면 해당 테이블 공간을 모두 백업한 후 복원하는 방안을 고려해야 합니다. SMS 테이블 공간은 이러한 유형의 데이터 분산을 테이블 공간에서 지원하지 않습니다.

- 관리 문제점

  관리 기능 중 일부는 데이터베이스 또는 테이블 레벨 대신에 테이블 공간 레벨에서 수행될 수 있습니다. 예를 들어, 데이터베이스 대신 테이블 공간을 백업할 경우 시간과 자원을 보다 효율적으로 사용할 수 있습니다. 이는 대용량의 변경사항이 있는 테이블 공간은 자주 백업하는 반면, 변경사항이 매우 적은 테이블 공간은 가끔씩 백업하도록 합니다.

  데이터베이스 또는 테이블 공간을 복원할 수 있습니다. 서로 상관없는 테이블이 테이블 공간을 공유하지 않는 경우에는, 데이터베이스의 더 작은 부분을 복원하고 비용을 줄일 수 있는 옵션을 갖습니다.

좋은 접근 방식은 관련 테이블을 테이블 공간 세트로 그룹화하는 것입니다. 이들 테이블은 참조 제한조건을 통해 또는 기타 비지니스 제한조건을 통해 관련될 수 있습니다.

자주 특정 테이블 공간을 삭제하고 재정의하는 것이 필요한 경우, 자체 테이블 공간에 테이블을 정의하는데, 그 이유는 테이블을 삭제하는 것보다 DMS 테이블을 삭제하는 것이 더 효율적이기 때문입니다.

extent 크기 선택

테이블 공간에 대한 extent *크기*는 데이터가 다음 컨테이너에 기록되기 전에 하나의 컨테이너에 기록될 테이블 데이터의 페이지 수를 나타냅니다. extent *크기*를 선택할 때 다음을 고려해야 합니다.

- 테이블 공간의 테이블 *크기* 및 유형

  DMS 테이블 공간 내의 공간은 한번에 한 extent 만큼 테이블에 할당됩니다. 테이블에 extent가 할당되면서 extent가 가득 차면, 새로운 extent가 할당됩니다.

  테이블은 다음의 개별 테이블 오브젝트로 구성됩니다.

  – 데이터 오브젝트. 일반 컬럼 데이터가 저장된 장소입니다.

  – 색인 오브젝트. 테이블에서 정의된 모든 색인은 여기에 저장됩니다.

  – Long 필드 오브젝트. 테이블에 하나 이상의 LONG 컬럼이 있는 경우, 이것은 long 필드 데이터가 저장되는 곳입니다.

  – 두 개의 LOB 오브젝트. 테이블에 하나 이상의 LOB 컬럼이 있으면, 이들 두 테이블 오브젝트에 저장됩니다.

    - LOB 데이터용 하나의 테이블 오브젝트

    - LOB 데이터를 설명하는 메타데이터용 두 번째 테이블 오브젝트

  각 테이블 오브젝트는 따로 저장되고 필요에 따라 새로운 extent를 할당합니다. 또한, 각 테이블 오브젝트는 테이블 오브젝트에 속하는 테이블 공간의 모든 extent를 설명하는 *extent map*이라는 메타 데이터 오브젝트와 한쌍이 됩니다. 또한, extent 맵 공간은 한번에 한 extent만큼 할당됩니다.

  그러므로 테이블의 공간 시작 할당은 각 테이블 오브젝트에 대해 두 개의 extent가 됩니다. 테이블 공간에 소형 테이블이 많이 있으면, 상대적으로 적은 데이터

량을 저장하기 위해 할당된 상대적으로 큰 공간량을 가지게 됩니다. 이 경우, 작은 extent *크기*를 지정하거나 한번에 한 페이지를 할당하는 SMS 테이블 공간을 사용해야 합니다.

한편, 고속으로 성장하는 초대형 테이블을 가지고 있고 작은 extent *크기*로 DMS 테이블 공간을 사용 중이면, 추가 extent의 잦은 할당으로 인해 불필요한 오버헤드가 발생할 수 있습니다.

- 테이블에 대한 액세스 유형

  테이블에 대한 액세스에 많은 양의 조회와 대용량 데이터를 처리하는 트랜잭션이 포함되어 있으면, 테이블에서 데이터를 프리페치함으로써 성능을 상당히 향상시킬 수 있습니다. (데이터 프리페치 및 extent *크기*와의 관계에 대해서는 *관리 안내서: 성능*에서 자세한 내용을 참조하십시오.)

- 필수 extent의 최소 수

  테이블 공간의 5개 extent용 컨테이너에 충분한 공간이 없으면, 테이블 공간이 작성되지 않습니다.

임시 테이블 공간에 대한 권고사항

대부분의 일반 테이블 공간에서 사용되는 페이지 *크기*와 동일한 페이지 *크기*로 단일 SMS 임시 테이블 공간을 정의하는 것이 바람직합니다. 이는 일반적인 환경 및 워크로드에 적합해야 합니다. 그러나, 다른 임시 테이블 공간 구성 및 워크로드를 사용하면 더 나을 수 있습니다. 다음 요점을 고려해야 합니다.

- 대부분의 경우, 임시 테이블은 일괄처리 및 순차적으로 액세스됩니다. 즉, 행의 일괄처리가 삽입되거나 순차 행의 일괄처리가 페치됩니다. 그러므로, 더 큰 페이지 *크기*를 사용하면, 주어진 데이터량을 읽기 위해 더 적은 수의 논리 또는 물리 페이지 입출력 요청이 필요하므로 더 나은 성능 특성을 초래합니다. 이는 항상 평균 임시 테이블 행 *크기*가 255로 나누어진 페이지 *크기*보다 작을 경우는 아닙니다. 페이지 *크기*에 관계없이 모든 페이지에 최대 255행이 올 수 있습니다. 예를 들어, 15바이트 행의 임시 테이블이 필요한 조회는 255의 행이 4KB 페이지에 전부 들어가므로 4KB 임시 테이블 공간 페이지 *크기*에서 더 잘 처리됩니다. 8KB(또는 그 이상) 페이지 *크기*를 사용하면, 각 임시 테이블에서 최소한 4KB(또는 그 이상)의 공간이 낭비되며, 필수 입출력 요청의 수를 줄이지 않습니다.

- 데이터베이스에서 일반 테이블 공간의 50%가 같은 페이지 *크기*를 사용하면, 임시 테이블 공간을 같은 페이지 *크기*로 정의하는 것이 더 좋습니다. 이렇게 하면, 임시 테이블 공간은 대부분 또는 모든 일반 테이블 공간과 동일한 버퍼 풀 공간을 공유할 수 있게 됩니다. 이는 다시 버퍼 풀 조정을 단순화합니다.
- 임시 테이블 공간을 사용하여 테이블을 재구성하는 경우, 임시 테이블 공간의 페이지 *크기*는 테이블의 페이지 *크기*와 일치해야 합니다. 이러한 이유로, 임시 테이블 공간을 사용하여 재구성할 수 있는 기존의 테이블에서 사용되는 서로 다른 각 페이지 *크기*에 대해 정의된 임시 테이블 공간이 있는지 확인해야 합니다.

  테이블 "inplace"를 재구성하여 임시 테이블 공간 없이 재구성을 수행할 수도 있습니다. 즉, 직접 목표 테이블 공간에 재구성할 수 있습니다. 물론, 이 "inplace" 재구성 작업에서는 목표 테이블 공간에 여유 공간이 있어야 합니다. 테이블 재구성에 대해서는 *관리 안내서: 성능*에서 자세한 내용을 참조하십시오.
- 일반적으로, 페이지 *크기*가 다른 임시 테이블 공간이 있으면, 최적화 알고리즘은 대부분의 경우 가장 큰 버퍼 풀의 임시 테이블 공간을 선택합니다. 이 경우, 임시 테이블 공간 중 하나에 충분히 큰 버퍼 풀을 지정하고 나머지는 작은 버퍼 풀을 지정하는 것이 바람직합니다. 이와 같은 버퍼 풀 지정은 기본 메모리를 효율적으로 활용할 수 있게 해 줍니다. 예를 들어, 카탈로그 테이블 공간이 4KB 페이지를 사용하고 나머지 테이블 공간이 8KB를 사용하면, 최적의 임시 테이블 공간 구성은 큰 버퍼 풀의 단일 8KB 임시 테이블 공간과 작은 버퍼 풀의 단일 4KB 테이블 공간일 수 있습니다.

  주: 카탈로그 테이블 공간은 4KB의 페이지 *크기*를 사용하도록 제한됩니다. 그러므로, 데이터베이스 관리 프로그램은 카탈로그 테이블 재구성을 작동시키기 위해 항상 4KB 임시 테이블 공간이 있도록 강제 수행합니다.
- 일반적으로, 단일 페이지 *크기*의 임시 테이블 공간을 둘 이상 정의해도 별 이득은 없습니다.
- SMS는 다음과 같은 이유로 항상 임시 테이블 공간에 대해 DMS보다는 더 나은 선택이 됩니다.
  - SMS에서 디스크 공간은 필요에 따라 할당되지만, DMS에서는 사전에 할당됩니다. 사전 할당은 어려울 수 있습니다. 임시 테이블 공간은 매우 큰 최대 저장영역 요구사항이 있지만, 더 작은 평균 저장영역 요구사항이 있는 임시

데이터를 보관합니다. DMS에서는 최대 저장영역 요구사항이 사전에 할당되어야 하는 반면, SMS에서는 여분의 디스크 공간이 충분한 시간 중에 다른 목적으로 사용될 수 있습니다.

- 데이터베이스 관리 프로그램은 이들을 디스크로 기록하기 보다는 임시 테이블 페이지를 메모리에 보관하려고 합니다. 결과적으로, DMS의 성능 장점은 덜 중요해집니다.
- SMS 컨테이너는 파일 시스템의 버퍼링을 이용할 수 있습니다. DMS 컨테이너는 이렇게 할 수 없습니다.

카탈로그 테이블 공간에 대한 권고사항

SMS 테이블 공간은 다음과 같은 이유로 데이터베이스 카탈로그에 권장됩니다.

- 데이터베이스 카탈로그는 여러 가지 크기의 여러 테이블로 구성됩니다. DMS 테이블 공간을 사용할 때에는, 각 테이블 오브젝트에 대해 최소한 두 개의 extent가 할당됩니다. 선택된 extent 크기에 따라, 상당량의 공간이 할당되어 사용되지 않는 공간이 생길 수 있습니다. DMS 테이블 공간을 사용할 때, 작은 extent 크기(두 페이지 또는 세 페이지)가 선택되어야 합니다. 그렇지 않으면, SMS 테이블 공간이 사용되어야 합니다.
- 카탈로그 테이블에는 대형 오브젝트(LOB) 컬럼이 있습니다. LOB 데이터는 다른 데이터와 함께 버퍼 풀에 보관되지 않지만, 필요할 때마다 디스크에서 읽습니다. 디스크에서 LOB를 읽는 것은 성능을 저하시킵니다. 보통 파일 시스템은 SMS 테이블 공간 또는 파일 컨테이너에 구축된 DMS 테이블 공간을 사용하여 자체적으로 데이터를 저장(또는 캐쉬화)할 장소를 가지고 있기 때문에, 앞에서 LOB가 참조됐을 때 입출력을 피할 수 있게 합니다.

이러한 점을 고려할 때, 카탈로그에 대해 SMS 테이블 공간을 선택하는 것이 좀 더 낫습니다.

고려해야 할 또다른 점은 향후에 카탈로그 테이블 공간을 확장할 필요가 있을 것인가 하는 점입니다. 일부 플랫폼은 SMS 컨테이너의 기본 저장영역의 확대를 지원하고, SMS 테이블 공간을 확대하기 위해 재지정된 복원을 사용할 수 있지만, DMS 테이블 공간을 사용할 경우 새 컨테이너를 쉽게 추가할 수 있습니다.

워크로드 고려사항

사용자의 환경에서 DB2가 관리하는 워크로드의 기본 유형은 사용할 테이블 공간의 유형 및 지정할 페이지 크기를 선택하는 데 영향을 줍니다. 온라인 거래 처리(OLTP) 워크로드는 데이터에 임의로 액세스하여 보통 작은 데이터 세트를 리턴하는 트랜잭션으로 특성화됩니다. 액세스가 임의로 이루어지고 하나의 페이지 또는 몇몇 페이지와 관련될 경우, 프리페치가 가능하지 않습니다.

이 경우, 장치 컨테이너를 사용하는 DMS 테이블 공간이 가장 우수한 성능을 발휘합니다. 최대의 성능이 필요하지 않을 경우, OLTP 워크로드에 대해 파일 컨테이너 또는 SMS 테이블 공간이 있는 DMS 테이블 공간을 선택하는 것이 합리적입니다. 순차적 입출력이 거의 없거나 전혀 없을 경우, CREATE TABLESPACE 문에서의 EXTENTSIZE 및 PREFETCHSIZE 매개변수 설정값은 입출력 효율성에 영향을 주지 않습니다.

조회 워크로드는 데이터에 순차적으로 또는 부분 순차적으로 액세스하여 보통 대형 데이터 세트를 리턴하는 트랜잭션으로 특징지워집니다. 여러 장치 컨테이너를 사용하고 각 컨테이너가 별도의 디스크에 있는 DMS 테이블 공간에서 효율적인 프리페치가 이루어질 가능성이 가장 큽니다. CREATE TABLESPACE문의 PREFETCHSIZE 매개변수 값은 EXTENTSIZE 매개변수 값에 장치 컨테이너 수를 곱한 값으로 설정되어야 합니다. 이 경우, DB2는 모든 컨테이너로부터 동시에 프리페치할 수 있습니다.

파일 시스템이 자체의 프리페치를 갖지 않을 경우, 조회 워크로드에 대한 합리적인 대안은 파일을 사용하는 것입니다. 파일은 파일 컨테이너를 사용하는 DMS 유형 또는 SMS 유형입니다. SMS를 사용할 경우 입출력을 병렬 처리하려면, 물리적 디스크를 분리할 디렉토리 컨테이너 맵이 있어야 합니다.

혼합 워크로드의 목적은 OLTP 워크로드에 대한 하나의 입출력 요청을 가능한 한 효율적으로 하여, 조회 워크로드의 병렬 입출력의 효율성을 극대화하는 것입니다.

테이블 공간의 페이지 크기를 결정할 때 고려해야 할 사항은 다음과 같습니다.

- 임의의 행을 읽고 쓰는 OLTP 응용프로그램의 경우, 페이지 크기가 작으면 불필요한 행에 소비되는 버퍼 공간이 적기 때문에 보다 바람직합니다.

- 한번에 많은 수의 연속 행에 액세스하는 DSS 응용프로그램의 경우, 페이지 *크기*가 크면 특정 행 번호를 읽는 데 필요한 입출력 요청 수가 줄어들기 때문에 보다 바람직합니다. 그러나, 이 경우 예외가 있습니다. 행 *크기*가 다음 보다 작으면

  ```
  pagesize / 255
  ```

  각 페이지에서 낭비되는 공간이 생깁니다(하나의 페이지당 최대 255 행이 있음). 이 경우, 보다 작은 페이지 *크기*를 사용하는 것이 더 바람직합니다.
- 페이지 *크기*가 더 크면 색인의 레벨 수를 줄일 수 있습니다.
- 페이지 *크기*가 더 크면 보다 길이가 긴 행이 지원됩니다.
- 기본 4KB 페이지에서 테이블이 500 컬럼으로 제한되는 반면, 더 큰 페이지 크기(8KB, 16KB 및 32KB)는 1012 컬럼을 지원합니다.
- 테이블 공간의 최대 *크기*는 테이블 공간의 페이지 *크기*에 비례합니다. 그 한계는 *SQL 참조서*에 있습니다.

SMS 또는 DMS 테이블 공간 선택

데이터 저장에 사용할 테이블 공간의 유형을 결정할 때 고려해야 할 몇 가지 사항이 있습니다.

SMS 테이블 공간의 장점

- 공간이 요구될 때까지 시스템에 의해 할당되지 않습니다.
- 데이터베이스를 작성하면 컨테이너를 사전 정의할 필요가 없으므로 초기 작업이 줄어듭니다.

DMS 테이블 공간의 장점

- 테이블 공간의 *크기*는 ALTER TABLESPACE문을 사용하여 컨테이너를 추가함으로써 증가될 수 있습니다. 기존의 데이터는 최적의 입출력 효율성을 보존할 수 있도록 새로운 컨테이너 세트에서 자동으로 재조정됩니다.
- 테이블은 저장되는 다음 데이터의 유형에 따라 다중 테이블 공간에 파티션될 수 있습니다.
  - Long 필드 및 LOB 데이터
  - 색인

– 일반 테이블 데이터

사용자가 성능상의 이유로 사용자의 테이블 데이터를 분리시키거나 테이블의 데이터 저장량을 증가시키려고 할 수도 있습니다. 예를 들어, 64GB의 정규 테이블 데이터, 64GB의 색인 데이터 및 2TB의 long 데이터가 들어 있는 테이블을 가질 수 있습니다. 8KB 페이지를 사용하는 경우, 테이블 데이터 및 색인 데이터는 128GB가 될 수 있습니다. 16KB 페이지를 사용하는 경우, 256GB가 될 수 있습니다. 32KB 페이지를 사용하는 경우, 테이블 데이터 및 색인 데이터는 512GB가 될 수 있습니다.

- 디스크에서 데이터의 위치는 운영 체제가 제어를 허용하면 제어될 수 있습니다.
- 모든 테이블 데이터가 단일 테이블 공간 내에 있으면, 테이블을 삭제하고 다시 정의하는 것보다 훨씬 적은 오버헤드로 테이블 공간을 삭제하고 다시 정의할 수 있습니다.
- 일반적으로, 잘 조정된 DMS 테이블 공간 세트는 SMS 테이블 공간보다 성능이 뛰어납니다.

주: Solaris 및 PTX(IBM NUMA-Q)에서, 원시 장치가 있는 DMS 테이블 공간은 성능이 중요한 워크로드에 특히 권장됩니다.

일반적으로, 소규모의 개인용 데이터베이스는 SMS 테이블 공간으로 관리하는 것이 가장 좋습니다. 반면에 대규모이면서 계속 커지는 데이터베이스의 경우에는 임시 테이블 공간 및 카탈로그 테이블 공간용으로 SMS 테이블 공간을 사용하고, 각 테이블용으로 다중 컨테이너를 가지고 있는 개별 DMS 테이블 공간을 사용하게 됩니다. 그리고, long 필드 데이터 및 색인을 자체 테이블 공간에서 저장하려고 할 수 있습니다.

장치 컨테이너가 있는 DMS 테이블 공간을 사용하려고 선택한 경우, 사용자가 환경을 조정하고 관리해야 합니다. *관리 안내서: 성능*의 "DMS 장치를 위한 성능상의 고려사항"에서 자세한 내용을 참조하십시오.

RAID 장치에 데이터 배치시 성능 최적화

이 절에서는 데이터를 RAID(Redundant Array of Independent Disks) 장치에 놓을 때 성능을 최적화하는 방법을 설명합니다. 일반적으로, RAID 장치를 사용하는 각 테이블 공간에 대해 다음을 수행해야 합니다.

- 테이블 공간(RAID 장치 사용)에 대해 단일 컨테이너를 정의하십시오.
- 테이블 공간의 EXTENTSIZE를 RAID 스트라이프 *크기*와 같거나 그 배수가 되도록 설정하십시오.
- 테이블 공간의 PREFETCHSIZE는 다음과 같아야 합니다.
  - RAID 병렬 장치(또는 이 제품의 전체 배수)의 수로 곱한 RAID 스트라이프 *크기*
  - EXTENTSIZE의 배수
- DB2_PARALLEL_IO 레지스트리 변수(DB2_PARALLEL_IO 참조)를 테이블 공간에 대한 병렬 입출력을 작동 가능하게 하십시오.
- DB2_STRIPED_CONTAINERS 레지스트리 변수
  (145 페이지의 DB2_STRIPED_CONTAINERS 참조)를 사용하여 extent 경계가 테이블 공간에 정렬되어 있는지 확인하십시오.

DB2_PARALLEL_IO

테이블 공간 컨테이너에서 데이터를 읽거나 또는 쓰는 경우, DB2는 데이터베이스에 있는 컨테이너의 수가 1보다 크면 병렬 입출력을 사용할 수 있습니다. 그러나, 단일 컨테이너 테이블 공간에 대해 병렬 입출력을 작동 가능하게 하는 것이 더 이득이 되는 경우도 있습니다. 예를 들어, 컨테이너가 둘 이상의 물리적 디스크로 구성된 단일 RAID 장치에 작성되면, 병렬 읽기 쓰기 호출을 실행할 수 있습니다.

컨테이너가 하나인 테이블 공간에 대해 병렬 입출력을 강제하려면,
DB2_PARALLEL_IO 레지스트리 변수를 사용할 수 있습니다. 이 변수는 모든 테이블 공간을 의미하는 "*"(별표)로 설정되거나 쉼표로 구분된 테이블 공간 ID 목록으로 설정될 수 있습니다. 예를 들면, 다음과 같습니다.

```
db2set DB2_PARALLEL_IO=*        {turn parallel I/O on for all table spaces}
db2set DB2_PARALLEL_IO=1,2,4,8  {turn parallel I/O on for table spaces 1, 2,
                                 4, and 8}
```

레지스트리 변수를 설정한 후 DB2는 중지되어야 하고(**db2stop**) 새 변수가 적용되도록 재시작되어야 합니다(**db2start**).

DB2_STRIPED_CONTAINERS

현재, DMS 테이블 공간 컨테이너(장치 또는 파일)를 작성할 때 하나의 페이지 태그는 컨테이너 시작시에 저장됩니다. 나머지 페이지는 DB2에서 데이터 저장영역에 사용될 수 있으며 extent *크기* 블록으로 그룹화됩니다.

테이블 공간 컨테이너에서 RAID 장치를 사용하는 경우, RAID 스트라이프 *크기*와 같거나 배수가 되는 extent *크기*로 테이블 공간을 작성하는 것이 바람직합니다. 그러나, 하나의 페이지 컨테이너 태그로 인해 extent는 RAID 스트라이프로 정렬되지 않으며 입출력 요청중에 최적의 물리적 디스크 수 이상을 액세스해야 합니다.

DMS 테이블 공간 컨테이너는 *태그*가 자체(전체) extent에 있도록 작성될 수 있습니다. 이렇게 하면, 앞에서 설명된 문제점을 피할 수 있지만 컨테이너에서 여분의 오버헤드 extent가 필요하게 됩니다. 이 방식으로 컨테이너를 작성하려면, DB2 레지스트리 변수인 DB2_STRIPED_CONTAINERS를 ″ON″으로 설정한 후 인스턴스를 중지시킨 다음 재시작해야 합니다.

```
db2set DB2_STRIPED_CONTAINERS=ON
db2stop
db2start
```

작성되는 모든 DMS 컨테이너(CREATE TABLESPACE 또는 ALTER TABLESPACE문을 사용하여)에는 전체 extent를 취하는 *태그*가 있습니다. 기존의 컨테이너는 변경되지 않은 상태로 남습니다.

이 속성을 갖는 컨테이너 작성을 중지시키려면, 변수를 재설정한 후(기본값은 널(NULL)) 인스턴스를 중지시킨 다음 재시작하십시오.

```
db2set DB2_STRIPED_CONTAINERS=
db2stop
db2start
```

제어 센터 및 LIST TABLESPACE CONTAINERS 명령은 컨테이너가 스트라이프 컨테이너로 작성되는지의 여부를 나타내지 않습니다. 컨테이너의 작성 방법에 따라 ″파일″ 또는 ″장치″를 계속 사용합니다. 컨테이너가 스트라이프된 컨테이너로 작성되었음을 검증하기 위해 DB2DART의 /DTSF 옵션을 사용하여 테이블 공간

및 컨테이너 정보를 덤프하고 문제의 컨테이너에 대한 유형 필드를 살펴볼 수 있습니다. 조회 컨테이너 API(**sqlbftcq** 및 **sqlbtcq**)는 유형을 표시하는 단순 응용 프로그램을 작성할 수 있습니다.

연합 데이터베이스 설계 고려사항

연합 데이터베이스를 설계할 때, 다음 설계 주제를 고려하십시오.

- 공간 요구사항
- 네트워크 우선순위

일반적으로, 연합 데이터베이스에서 액세스 가능한 데이터는 해당 데이터베이스에 저장되지 않습니다. 데이터 소스 테이블 및 뷰에 대한 참조는 시스템 카탈로그에 저장되지만, 실제 데이터는 일반적으로 데이터 소스에 위치되어 있습니다. 그런 경우, 연합 데이터베이스는 일반 데이터베이스보다 저장영역 공간을 덜 필요로할 수 있습니다. 이 일반 규칙은 공동 배치 시스템 차이점 또는 데이터 소스에서의 함수 부족으로 인해 조회를 지역적으로 실행해야 하는 경우에 적용되지 않습니다. 이 경우, 테이블은 DB2에서 처리될 수 있도록 구체화됩니다.

연합 시스템 데이터의 대부분은 일반적으로 네트워크상에 위치한 하나 이상의 데이터 소스에 위치해 있으므로, DB2 및 네트워크 시스템에 지정된 자원을 변경하는 것에 대해 고려해 보십시오. 데이터베이스 관리 프로그램 자체보다는 DB2 시스템에서 네트워크에 더 많은 자원을 할당한 후, 성능이 향상되는 것을 볼 수 있습니다.

제9장 분산 데이터베이스 설계

트랜잭션은 *작업 단위(UOW)*로서 DB2에서 일반적으로 참조됩니다. 작업 단위는 응용프로그램 프로세스 내에서 복구할 수 있는 조작의 순서입니다. 데이터베이스가 일관된 상태에 있도록 하기 위해 데이터베이스 관리 프로그램이 사용됩니다. 데이터베이스에 대한 읽기 또는 쓰기는 하나의 작업 단위(UOW) 내에서 수행됩니다.

예를 들면, 은행 거래는 저축 예금 구좌에서 당좌 예금 구좌로의 자금 이체가 관련될 수 있습니다. 응용프로그램이 저축 예금 구좌에서 금액을 뺀 후에는 두 구좌가 일치하지 않게 되며, 금액이 당좌 예금 구좌에 더해질 때까지는 일치하지 않게 됩니다. 두 단계가 완료되면, 일관성 지점에 도달합니다. 변경사항이 확약되며 다른 응용프로그램에 사용 가능합니다.

작업 단위는 첫 번째 SQL문이 데이터베이스에 대해 발행될 때 시작됩니다. 응용프로그램은 COMMIT 또는 ROLLBACK문을 발행함으로써 작업 단위(UOW)를 종료해야 합니다. COMMIT문은 작업 단위 내에 작성된 모든 변경사항을 영구화합니다. ROLLBACK문은 데이터베이스로부터 이러한 변경사항을 제거합니다. 응용프로그램이 이러한 명령문 중 하나도 없이 정상적으로 종료하면 작업 단위(UOW)는 자동으로 확약됩니다. 응용프로그램이 작업 단위 중간에 비정상적으로 종료하면 작업 단위는 자동으로 구간 복원됩니다. COMMIT 또는 ROLLBACK은 일단 발행되면 중단될 수 없습니다. 일부 멀티 스레드 응용프로그램이나 Windows와 같은 일부 운영 체제를 사용하여, 응용프로그램이 이러한 명령문 없이 정상적으로 종료하면 작업 단위는 자동으로 구간 복원됩니다. 응용프로그램이 항상 명시적으로 확약되거나 구간 복원이 작업 단위(UOW)를 완료하도록 권장합니다. 작업 단위(UOW)의 일부가 성공적으로 완료되지 않는 경우에는 갱신사항이 구간 복원되어 참여하는 테이블을 트랜잭션이 시작되기 전의 상태로 두어야 합니다. 이는 어느 요청도 유실되거나 중복되지 않도록 합니다.

다음 주제 항목에서는 추가 정보를 제공합니다.

• 148 페이지의『한 트랜잭션에서 단일 데이터베이스 사용』

- 149 페이지의『단일 트랜잭션에서 다중 데이터베이스 사용』
- 155 페이지의『기타 구성 고려사항』
- 159 페이지의『2단계 확약 프로세스 이해』
- 162 페이지의『2단계 확약 중의 문제점 복구』

응용프로그램 개발 안내서 및 *CLI Guide and Reference*에서 분산 데이터베이스를 사용하는 응용프로그램을 작성하는 자세한 내용을 참조하십시오.

한 트랜잭션에서 단일 데이터베이스 사용

트랜잭션의 가장 간단한 형식은 한 작업 단위 내에서 하나의 데이터베이스에 대해서만 읽고 쓰는 것입니다. 이 데이터베이스 유형을 *원격 작업 단위(UOW)*라고 합니다.

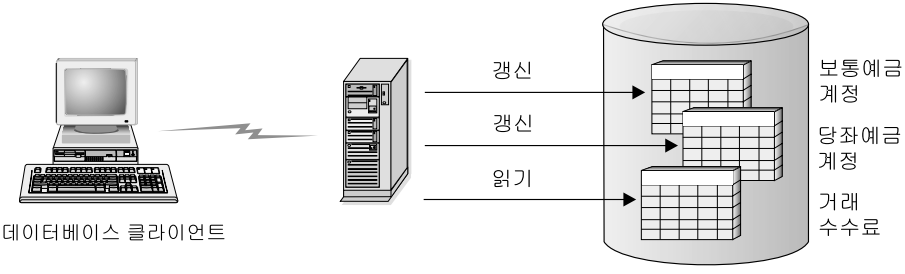

그림 30. 한 트랜잭션에서 단일 데이터베이스 사용

그림30에서는 은행 수수료 스케줄뿐만 아니라 당좌 예금 구좌와 저축 예금 구좌 테이블이 들어 있는 데이터베이스에 액세스하는 펀드 이체 응용프로그램을 수행하는 데이터베이스 클라이언트를 나타냅니다. 응용프로그램은 다음과 같아야 합니다.

- 사용자 인터페이스에서 이체할 금액을 받아들이십시오.
- 저축 예금 구좌에서 금액을 빼고 새로운 잔액을 결정하십시오.
- 수수료 스케줄을 읽어서 주어진 잔액을 가진 저축 예금 구좌에 대해 거래 수수료를 결정하십시오.
- 저축 예금 구좌에서 거래 수수료를 빼십시오.
- 이체 금액을 당좌 예금 구좌에 더하십시오.
- 거래(작업 단위(UOW))를 확약하십시오.

응용프로그램을 설정하려면, 다음을 수행해야 합니다.

1. 같은 데이터베이스 내에 저축 예금 구좌, 당좌 예금 구좌 및 은행 수수료 스케줄에 대한 테이블을 작성하십시오(*관리 안내서: 구현*에서 "설계 구현" 참조).
2. 물리적으로 원격일 경우, *설치 및 구성 보충 설명서* 매뉴얼에 설명된 대로 데이터베이스 서버를 설정하여 적절한 통신 프로토콜을 사용하십시오.
3. 물리적으로 원격일 경우, *빠른 시작* 매뉴얼에 설명된 대로 위의 데이터베이스 서버에서 데이터베이스를 식별하기 위해 노드 및 데이터베이스를 카탈로그화하십시오.
4. 응용프로그램을 사전 처리 컴파일하여 유형 1 연결을 지정하십시오. 즉, *응용프로그램 개발 안내서*에 설명된 대로 PRECOMPILE PROGRAM 명령에 CONNECT 1(기본값)을 지정하십시오.

단일 트랜잭션에서 다중 데이터베이스 사용

단일 트랜잭션에서 다중 데이터베이스를 사용하면, 트랜잭션에서 갱신중인 데이터베이스 수에 따라 환경을 설정하고 관리하는 데 필요한 요구사항이 달라집니다.

단일 데이터베이스 갱신

데이터가 다중 데이터베이스에 걸쳐 분산되어 있는 경우, 하나 이상의 데이터베이스를 읽는 동안 다른 하나의 데이터베이스를 갱신하고자 할 수 있습니다. 이러한 유형의 액세스는 단일 작업 단위(트랜잭션) 내에서 수행될 수 있습니다.

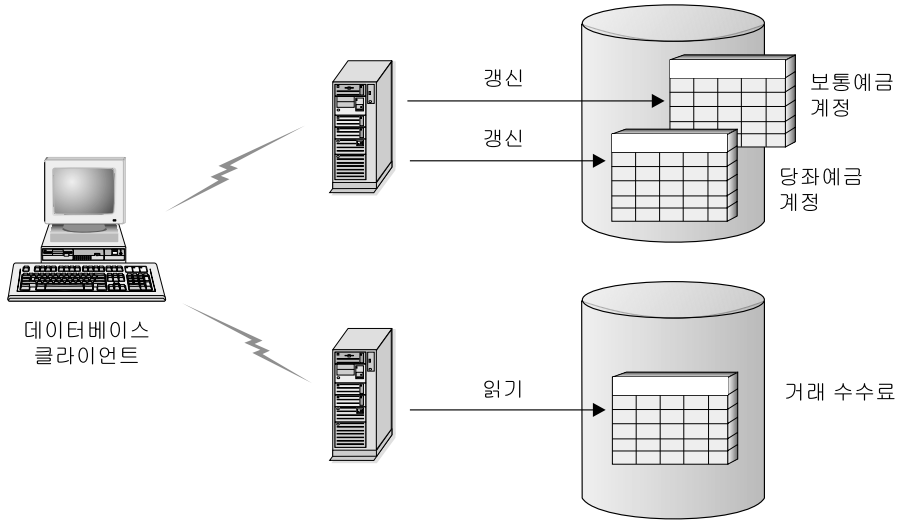

그림 31. 단일 트랜잭션에서 다중 데이터베이스 사용

그림31에서는 당좌 예금 구좌와 저축 예금 구좌를 포함하는 데이터베이스 서버와 은행 수수료 스케줄을 포함하는 데이터베이스 서버에 액세스하는 펀드 이체 응용프로그램을 수행하는 데이터베이스 클라이언트를 나타냅니다. 이 예는 데이터베이스의 수와 테이블의 위치를 제외하면 148 페이지의 그림30에서 제공되는 예와 비슷합니다.

이 환경의 펀드 이체 응용프로그램을 설정하려면, 다음을 수행해야 합니다.

1. 해당 데이터베이스에서 필요한 테이블을 작성하십시오(*관리 안내서: 구현*에서 "설계 구현" 참조).
2. 물리적으로 원격일 경우, *설치 및 구성 보충 설명서* 매뉴얼에 설명된 대로 데이터베이스 서버를 설정하여 적절한 통신 프로토콜을 사용하십시오.
3. 물리적으로 원격일 경우, *빠른 시작* 매뉴얼에 설명된 대로 위의 데이터베이스 서버에서 데이터베이스를 식별하기 위해 노드 및 데이터베이스를 카탈로그화하십시오.
4. *응용프로그램 개발 안내서*에 설명된 대로 응용프로그램을 사전 처리 컴파일하여 유형 2 연결을 지정하고(PRECOMPILE PROGRAM 명령에 CONNECT 2 지정), 1단계 확약을 지정하십시오(PRECOMPILE PROGRAM 명령에 SYNCPOINT ONEPHASE 지정).

데이터베이스가 호스트나 AS/400 데이터베이스 서버에 위치한 경우, 이 서버들에 대한 연결성을 위해 DB2 Connect를 요구합니다. DB2 Connect *빠른 시작* 책 중 하나에서 단계에 대한 자세한 내용을 참조하십시오. *DB2 Connect 사용자 안내서*에서 DB2 Connect에 대한 자세한 내용을 참조하십시오.

다중 데이터베이스 갱신

데이터가 다중 데이터베이스에 걸쳐 분산되어 있는 경우, 단일 트랜잭션 내에서 다중 데이터베이스를 읽고 갱신하고자 할 수 있습니다. 이러한 유형의 데이터베이스 액세스를 *다중 사이트 갱신*이라고 합니다.

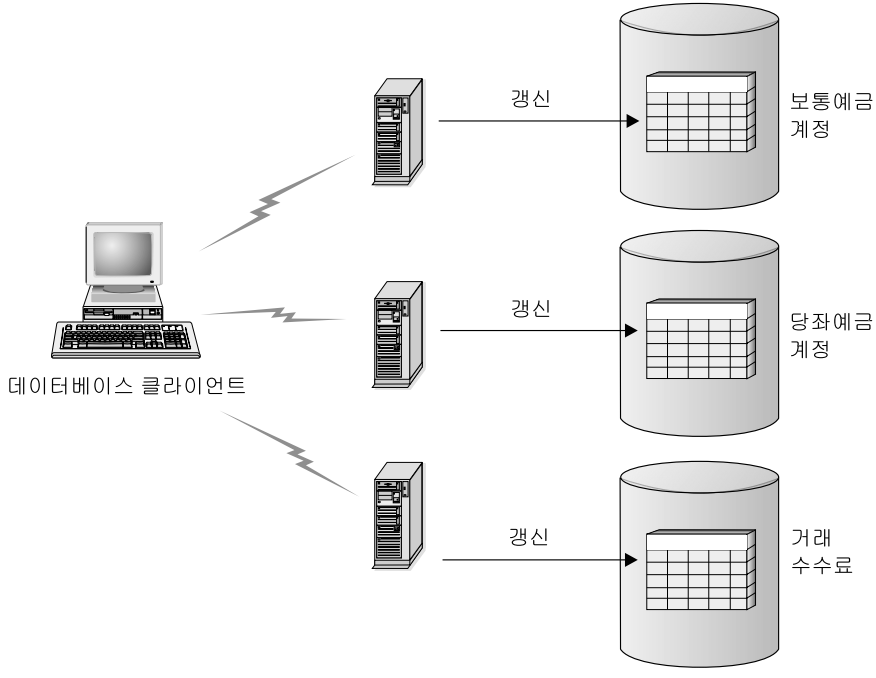

그림 32. 단일 트랜잭션에서 다중 데이터베이스 갱신

그림32에서는 당좌 예금 구좌, 저축 예금 구좌, 은행 수수료 스케줄을 각각 포함하는 세 데이터베이스 서버에 액세스하는 펀드 이체 응용프로그램을 수행하는 데이터베이스 클라이언트를 나타냅니다.

이 환경의 펀드 이체 응용프로그램을 설정하려면, 두 가지 옵션이 있어야 합니다.

1. 트랜잭션 관리 프로그램(TM)일 경우:

a. 해당 데이터베이스에서 필요한 테이블을 작성하십시오(*관리 안내서: 구현*에서 "설계 구현" 참조).

b. 물리적으로 원격일 경우, *설치 및 구성 보충 설명서* 매뉴얼에 설명된 대로 데이터베이스 서버를 설정하여 적절한 통신 프로토콜을 사용하십시오.

c. 물리적으로 원격일 경우, *빠른 시작* 매뉴얼에 설명된 대로 위의 데이터베이스 서버에서 데이터베이스를 식별하기 위해 노드 및 데이터베이스를 카탈로그화하십시오.

d. *응용프로그램 개발 안내서*에 설명된 대로 응용프로그램을 사전 처리 컴파일하여 유형 2 연결을 지정하고(PRECOMPILE PROGRAM 명령에 CONNECT 2 지정), 2단계 확약을 지정하십시오(PRECOMPILE PROGRAM 명령에 SYNCPOINT TWOPHASE 지정).

e. 『DB2 트랜잭션 관리 프로그램 사용』에 설명된 대로 DB2 트랜잭션 관리 프로그램(TM)을 구성하십시오.

2. 트랜잭션 관리 프로그램이 아닌 경우:

a. 해당 데이터베이스에서 필요한 테이블을 작성하십시오(*관리 안내서: 구현*에서 "설계 구현" 참조).

b. 물리적으로 원격일 경우, *설치 및 구성 보충 설명서* 매뉴얼에 설명된 대로 데이터베이스 서버를 설정하여 적절한 통신 프로토콜을 사용하십시오.

c. 물리적으로 원격일 경우, *빠른 시작* 매뉴얼에 설명된 대로 위의 데이터베이스 서버에서 데이터베이스를 식별하기 위해 노드 및 데이터베이스를 카탈로그화하십시오.

d. *응용프로그램 개발 안내서*에 설명된 대로 응용프로그램을 사전 처리 컴파일하여 유형 2 연결을 지정하고(PRECOMPILE PROGRAM 명령에 CONNECT 2 지정), 1단계 확약을 지정하십시오(PRECOMPILE PROGRAM 명령에 SYNCPOINT ONEPHASE 지정).

DB2 트랜잭션 관리 프로그램 사용

데이터베이스 관리 프로그램은 단일 작업 단위(UOW) 안에서 여러 데이터베이스 갱신을 조정하는 데 사용할 수 있는 트랜잭션 관리 프로그램 기능을 제공합니다. 데이터베이스 클라이언트는 자동으로 작업 단위(UOW)를 자동으로 조정하고 *트랜잭션 관리 프로그램 데이터베이스*를 사용하여 각 트랜잭션을 등록하고 완료 상태를 추적합니다.

IBM TXSeries, BEA Tuxedo 또는 Microsoft Transaction Server와 같은 XA 준수 트랜잭션 관리 프로그램을 사용하는 경우, 통합 관련 지침은 167 페이지의 『제10장 트랜잭션 관리 프로그램 설계』에서 자세한 내용을 참조하십시오.

UNIX용 DB2 UDB 기본 시스템, Windows 운영 체제 또는 OS/2를 사용하여 트랜잭션을 조정할 때, 특정 구성 요구사항을 완수해야 합니다. 통신에 TCP/IP만을 사용하고 DB2 UDB 및 OS/390용 DB2가 트랜잭션과 관련된 유일한 데이터베이스 서버인 경우, 구성이 단순해집니다.

TCP/IP 연결을 사용하는 DB2 UDB 및 OS/390용 DB2: 다음과 같은 각 명령문이 사용자 환경에서 참이면, 다중 사이트 갱신의 구성 단계는 간단합니다.

- 원격 데이터베이스 서버와의 모든 통신(OS/390용 DB2 UDB 포함)은 배타적으로 TCP/IP를 사용합니다.
- Unix용 DB2 UDB 기본 시스템, Windows 운영 체제, 시스템 또는 OS/390은 트랜잭션에 포함된 유일한 데이터베이스 서버입니다.
- DB2 Connect 동기 지점 관리 프로그램(SPM)은 구성되지 않습니다.

  DB2 Connect 동기 지점 관리 프로그램은 DB2 인스턴스 작성 시에 자동으로 구성되며 다음 경우에 필요합니다.

  – 다중 사이트 갱신을 위해 SNA 연결이 호스트 또는 AS/400 데이터베이스 서버에 사용되는 경우

  – XA 준수 트랜잭션 관리 프로그램(IBM TXSeries CICS 등)이 2단계 확약을 조정하는 경우

    이는 호스트 또는 AS/400 데이터베이스 서버의 SNA 및 TCP/IP 연결 모두에 적용됩니다. 167 페이지의 『제10장 트랜잭션 관리 프로그램 설계』에서 자세한 내용을 참조하십시오. 사용자 환경에서 DB2 Connect 동기 지점 관리 프로그램을 요구하지 않는 경우, DB2 Connect 서버에서 `db2 update dbm cfg using spm_name NULL`을 발행하여 해제시킬 수 있습니다. 그런 다음 DB2를 중지하고 재시작하십시오.

트랜잭션 관리 프로그램 데이터베이스로 사용될 데이터베이스는 데이터베이스 관리 프로그램 구성 매개변수인 *tm_database*에 의해 데이터베이스 클라이언트에서 결

정됩니다. 이 구성 매개변수에 대해서는 *관리 안내서: 성능*의 "DB2 구성"에서 자세한 내용을 참조하십시오. 이 구성 매개변수를 설정할 때에는 다음 인수를 고려하십시오.

- 트랜잭션 관리 프로그램 데이터베이스는 다음과 같습니다.
  - UNIX용 DB2 UDB 기본 시스템, Windows 운영 체제 또는 OS/2 데이터베이스
  - OS/390용 DB2 버전 5 또는 이후의 데이터베이스

    이것은 트랜잭션 관리 프로그램 데이터베이스로서 사용하도록 권장되는 데이터베이스 서버입니다. OS/390 시스템은 보통 워크스테이션 서버보다 더 안전하므로 잘못하여 전원이 차단되거나 재부팅될 가능성이 더 적습니다. 그러므로, 재동기화시 사용되는 복구 로그가 보다 안전합니다.
- 1ST_CONN의 값이 *tm_database* 구성 매개변수용으로 지정된 경우, 응용프로그램이 연결하는 첫 번째 데이터베이스가 트랜잭션 관리 프로그램 데이터베이스로서 사용됩니다.

  1ST_CONN을 사용할 때에는 주의해야 합니다. 관련된 모든 데이터베이스가 올바로 카탈로그화되었는지 확인하기 쉬운 경우에만 이 구성을 사용해야 합니다. 즉, 다음과 같은 경우입니다.
  - 트랜잭션을 시작하는 데이터베이스 클라이언트가 트랜잭션 관리 프로그램 데이터베이스를 포함한 관련 데이터베이스가 들어 있는 인스턴스에 있습니다.
  - 데이터베이스에 대한 액세스를 카탈로그화하고 관리하기 위해 DCE 디렉토리 서비스를 사용 중입니다.

  응용프로그램이 트랜잭션 관리 프로그램 데이터베이스로 사용 중인 데이터베이스로부터 연결을 해제하려고 시도하는 경우, 사용자는 경고 메세지를 수신하며 작업 단위(UOW)가 확약될 때까지 연결이 유지된다는 것을 참조하십시오.

기타 환경: 사용자 환경에서,

- 원격 데이터베이스 서버와의 통신에 TCP/IP만이 사용되지 않는 경우(예를 들어, NETBIOS가 사용되는 경우)
- MVS용 DB2 버전 3 또는 버전 4, AS/400용 DB2 또는 VM&VSE용 DB2에 액세스한 경우
- SNA를 사용하여 OS/390용 DB2에 액세스한 경우

- DB2 Connect 동기 지점 관리 프로그램을 사용하여 호스트 또는 AS/400 데이터베이스 서버에 액세스한 경우

다중 사이트 갱신을 위한 구성 단계가 더 복잡해집니다.

트랜잭션 관리 프로그램 데이터베이스로 사용될 데이터베이스는 데이터베이스 관리 프로그램 구성 매개변수인 *tm_database*에 의해 데이터베이스 클라이언트에서 결정됩니다. 이 구성 매개변수에 대해서는 *관리 안내서: 성능*의 "DB2 구성"에서 자세한 내용을 참조하십시오. 이 구성 매개변수를 설정할 때에는 다음 인수를 고려하십시오.

- 트랜잭션 관리 프로그램 데이터베이스는 Unix용 DB2 UDB 기본 시스템, Windows 운영 체제 또는 OS/2 데이터베이스입니다.
- 1ST_CONN의 값이 *tm_database* 구성 매개변수용으로 지정된 경우, 응용프로그램이 연결하는 첫 번째 데이터베이스가 트랜잭션 관리 프로그램 데이터베이스로서 사용됩니다.

  1ST_CONN을 사용할 때에는 주의해야 합니다. 관련된 모든 데이터베이스가 올바로 카탈로그화되었는지 확인하기 쉬운 경우에만 이 구성을 사용해야 합니다. 즉, 다음과 같은 경우입니다.

  – 트랜잭션을 시작하는 데이터베이스 클라이언트가 트랜잭션 관리 프로그램 데이터베이스를 포함한 관련 데이터베이스가 들어 있는 인스턴스에 있습니다.
  – 데이터베이스에 대한 액세스를 카탈로그화하고 관리하기 위해 DCE 디렉토리 서비스를 사용 중입니다.

  응용프로그램이 트랜잭션 관리 프로그램 데이터베이스로 사용 중인 데이터베이스로부터 연결을 해제하려고 시도하는 경우, 사용자는 경고 메세지를 수신하며 작업 단위(UOW)가 확약될 때까지 연결이 유지된다는 것을 참조하십시오.

기타 구성 고려사항

환경을 설정할 때에는 다음 구성 매개변수를 고려해야 합니다. 이들 매개변수 설정에 대해서는 *DB2 Connect 사용자 안내서*에서 자세한 내용을 참조하십시오.

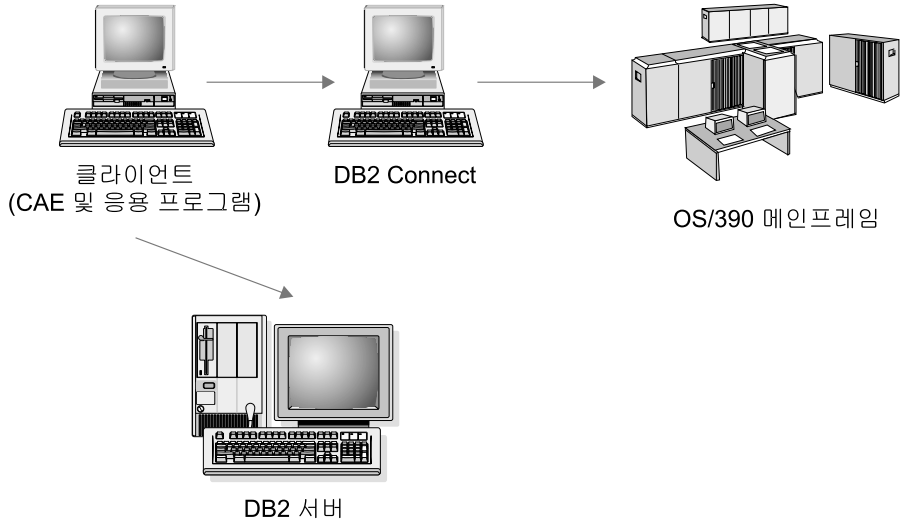

그림 33. 구성 고려사항

데이터베이스 관리 프로그램 구성 매개변수

- *tm_database*

  이 매개변수는 각 DB2 인스턴스에 대한 트랜잭션 관리 프로그램(TM) 데이터베이스 이름을 식별합니다.

- *spm_name*

  이 매개변수는 DB2 Connect 동기 지점 관리 프로그램 인스턴스의 이름을 데이터베이스 관리 프로그램으로 식별합니다. 재동기화에 성공하려면, 이름이 네트워크에서 고유해야 합니다.

- *resync_interval*

  이 매개변수는 DB2 트랜잭션 관리 프로그램, DB2 서버 데이터베이스 관리 프로그램 그리고 DB2 Connect 동기 지점 관리 프로그램 또는 DB2 UDB 동기 지점 관리 프로그램이 미해결의 2단계 확약 중 이상 실패 트랜잭션의 복구를 재시도해야 하는 초 단위의 시간 간격을 나타냅니다.

- *spm_log_file_sz*

  이 매개변수는 SPM 로그 파일의 크기(4KB 페이지)를 지정합니다.

- *spm_max_resync*

  이 매개변수는 재동기화 조작을 동시적으로 수행하는 에이전트의 수를 식별합니다.

- *spm_log_path*

  이 매개변수는 SPM 로그 파일에 대한 로그 경로를 나타냅니다.

데이터베이스 구성 매개변수

- *maxappls*

  이 매개변수는 사용 중인 응용프로그램의 최대 허용 수를 지정합니다. 이 매개변수의 값은 연결된 응용프로그램의 수와 2단계 확약 또는 구간 복원을 완료하는 프로세스에서 동시에 있을 수 있는 동일한 응용프로그램의 수와 한번에 존재할 수 있는 2단계 확약 중 이상 실패 트랜잭션의 기대 수를 합한 값과 같거나 커야 합니다. 2단계 확약 중 이상 실패 트랜잭션에 대해서는 162 페이지의 『2단계 확약 중의 문제점 복구』에서 자세한 내용을 참조하십시오.

- *autorestart*

  이 데이터베이스 구성 매개변수는 필요할 때 RESTART DATABASE 루틴이 자동으로 호출될지 여부를 지정합니다. 기본값은 예입니다(즉, 사용 가능).

  2단계 확약 중 이상 실패 트랜잭션이 들어 있는 데이터베이스에서는 시동을 위해 데이터베이스 조작을 재시작해야 합니다. *자동 재시작*이 사용 불가능한 경우, 데이터베이스로의 최종 연결이 삭제되면 다음 연결은 실패할 것이고 명시적 RESTART DATABASE가 다시 수행되어야 합니다. 이 조건은 2단계 확약 중 이상 실패 트랜잭션이 트랜잭션 관리 프로그램의 재동기 조작에 의해 제거되거나 관리자가 시작한 경험적 조작을 통해 제거될 때까지 존재합니다. RESTART DATABASE 명령이 발행될 때, 2단계 확약 중 이상 실패 트랜잭션이 데이터베이스에 있는 경우 메시지가 리턴됩니다. 그러면, 관리자는 LIST INDOUBT TRANSACTIONS 명령 및 다른 명령행 처리기 명령을 사용하여 2단계 확약 중 이상 실패 트랜잭션에 관한 정보를 알아낼 수 있습니다.

이 구성 매개변수에 대해서는 *관리 안내서: 성능*에서 자세한 내용을 참조하십시오.

다중 사이트 갱신의 LAN 기반 DB2 Universal Database 서버에 액세스하는 호스트 또는 AS/400 응용프로그램

DB2 Universal Database는 TCP/IP 연결성을 사용하는 호스트 또는 AS/400 데이터베이스 클라이언트로부터 다중 사이트 갱신을 지원하지 않습니다. 이 상황에서, SNA(Systems Network Architecture) 연결성만이 지원됩니다. DB2 동기 지점 관리 프로그램이 다중 사이트 갱신에 필요합니다. DB2 Connect는 이 시나리오에서 사용되지 않습니다.

호스트 또는 AS/400 데이터베이스 클라이언트로부터 액세스 중인 데이터베이스 서버는 DB2 동기 지점 관리 프로그램이 있는 워크스테이션에 대해 지역화될 필요가 없습니다. 호스트 또는 AS/400 데이터베이스 클라이언트는 DB2 동기 지점 관리 프로그램 워크스테이션을 임시 게이트웨이로 사용하여 DB2 UDB 서버에 연결할 수 있습니다. 이는 실제 DB2 UDB 서버가 조직에서 원격지에 있을 동안 DB2 동기 지점 관리 프로그램 워크스테이션을 보호 환경에 고립시킬 수 있게 합니다. 이는 또한 DB2 common server 버전 2 데이터베이스가 호스트 또는 AS/400 데이터베이스 클라이언트에서 시작되는 다중 사이트 갱신에 관련될 수 있게 합니다.

단계는 다음과 같습니다.

- 호스트나 AS/400 응용프로그램에 의해 직접 액세스될 워크스테이션에서,
  1. 호스트 또는 AS/400 데이터베이스 클라이언트와 함께 다중 사이트 갱신 지원을 제공하기 위해 DB2 Universal Database Enterprise Edition 또는 Enterprise - Extended Edition을 설치하십시오.
  2. 같은 시스템에 데이터베이스 인스턴스를 작성하십시오. 예를 들어, 기본 인스턴스 DB2를 사용하거나, 다음 명령을 사용하여 새로운 인스턴스를 작성할 수 있습니다.

     ```
     db2icrt myinstance
     ```

  3. 필요에 따라 사용권 정보를 제공하십시오.
  4. 레지스트리 값 DB2COMM이 APPC 값을 포함하는지 확인하십시오.
  5. 필요에 따라 SNA 통신을 구성하십시오. 지원되는 IBM SNA 제품이 사용될 때, DB2 동기 지점 관리 프로그램용으로 필요한 SNA 프로파일은 *spm_name* 데이터베이스 관리 프로그램 구성 매개변수의 값에 근거하여 자동으로 작성됩니다. 지원되는 다른 SNA 스택은 수동 구성을 요구합니다. 자세한 내용은 *설치 및 구성 보충 설명서*를 참조하십시오.
  6. *spm_name* 데이터베이스 관리 프로그램 구성 매개변수용으로 지정되는 값을 결정하십시오. 이 매개변수는 DB2 인스턴스 작성 시 머신의 머신 TCP/IP 호스트 이름의 파생물로 사전 구성됩니다. 이것이 승인가능하며 사용자 환경 내에서 고유하면, 변경하지 마십시오.

7. 필요한 경우, UPDATE DATABASE MANAGER CONFIGURATION 명령을 사용하여 DB2 Universal Database 서버에서 *spm_name*을 갱신하십시오.
8. 해당하는 경우, 이 DB2 워크스테이션이 원격 DB2 UDB 서버에 연결하는 데 필요한 통신을 구성하십시오.
9. 원격 DB2 UDB 서버가 이 DB2 서버에 연결하는 데 필요한 통신을 구성하십시오.
10. SPM을 최초로 시작하려면 DB2 Universal Database에서 데이터베이스 관리 프로그램을 중단하고 재시작하십시오.

    이 DB2 동기 지점 관리 프로그램 워크스테이션으로부터 원격 DB2 UDB 서버에 연결할 수 있어야 합니다.

- 호스트 또는 AS/400 데이터베이스 클라이언트에 의해 액세스될 각 원격 DB2 UDB 서버에서,
  1. 원격 DB2 UDB 워크스테이션이 이 DB2 UDB 동기 지점 관리 프로그램에 연결하는 데 필요한 통신을 구성하십시오.
  2. 데이터베이스 관리 프로그램을 중지한 다음 재시작하십시오.

2단계 확약 프로세스 이해

160 페이지의 그림34에서는 다중 사이트 갱신과 관련된 단계를 설명합니다. 2단계 확약 프로세스중 오류가 발생하는 경우, 트랜잭션 관리 방법에 대한 이해는 문제를 해결하는 데 도움이 됩니다.

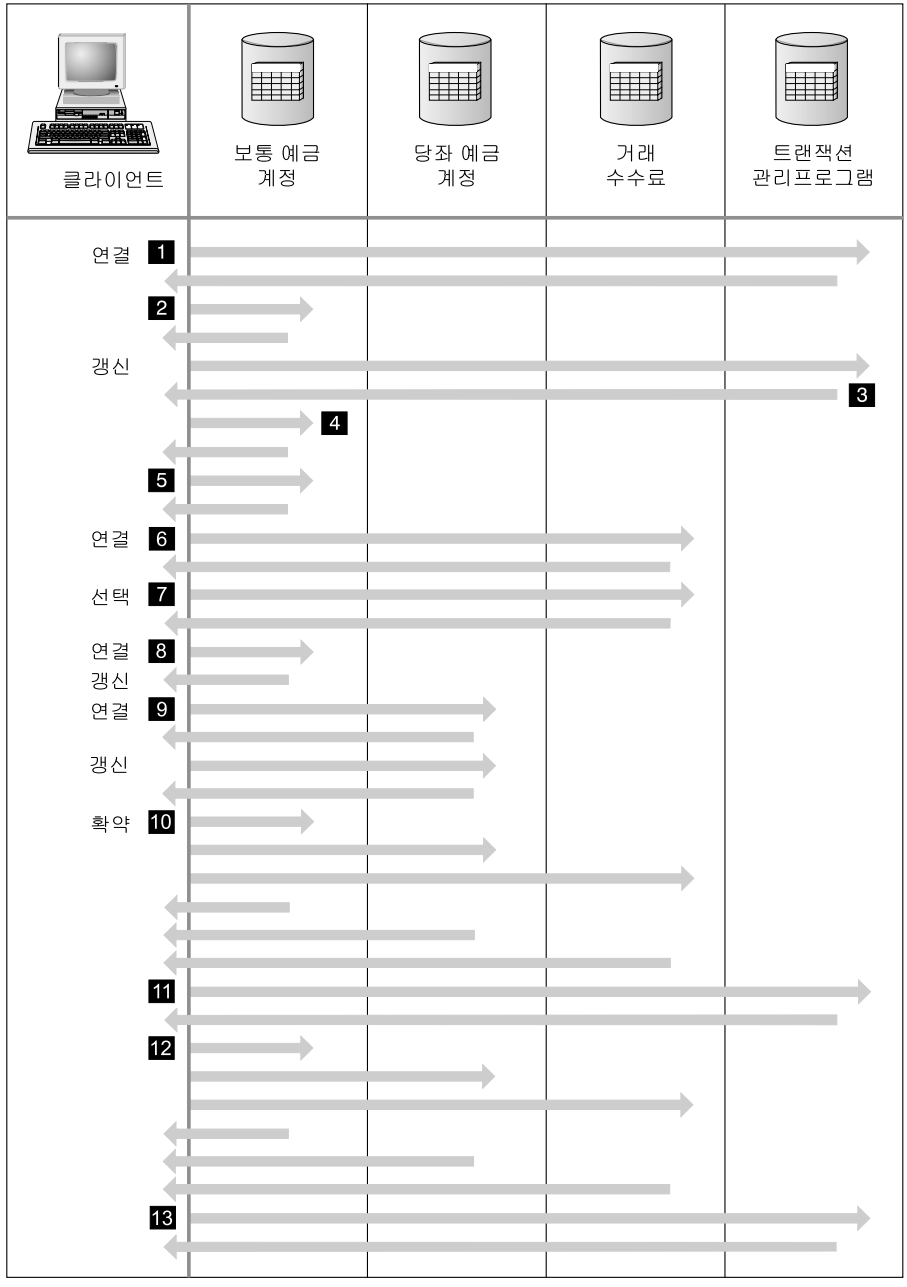

그림 34. 다중 데이터베이스 갱신

0 응용프로그램은 2단계 확약 준비가 되었습니다. 사전 처리 컴파일 옵션을 사용하여 이를 수행할 수 있습니다(자세한 정보는 *응용프로그램 개발 안내서* 참조). 또

한 DB2 CLI(콜 레벨 인터페이스) 구성을 통해서도 이를 수행할 수 있습니다(자세한 정보는 *CLI Guide and Reference* 참조).

1 데이터베이스 클라이언트는 SAVINGS_DB 데이터베이스에 연결하려는 경우, 먼저 내부적으로 트랜잭션 관리 프로그램(TM) 데이터베이스에 연결합니다. TM 데이터베이스는 데이터베이스 클라이언트에 확인 정보를 리턴합니다. 데이터베이스 관리 프로그램 구성 매개변수 *tm_database*가 1ST_CONN으로 설정되면, SAVINGS_DB는 이 응용프로그램 인스턴스의 지속기간 동안 트랜잭션 관리 프로그램 데이터베이스가 됩니다.

2 SAVINGS_DB 데이터베이스로 연결되고 확인됩니다.

3 데이터베이스 클라이언트는 SAVINGS_ACCOUNT 테이블에서 갱신을 시작합니다. 이것은 작업 단위(UOW)로 시작합니다. TM 데이터베이스는 데이터베이스 클라이언트에 응답하여 작업 단위(UOW)에 트랜잭션 ID를 제공합니다. 작업 단위(UOW) 등록은 작업 단위(UOW)의 첫 번째 SQL문이 수행될 때 발생하며, 연결 설정 동안에 발생하는 것은 아닙니다.

4 트랜잭션 ID를 수신한 후, 데이터베이스 클라이언트는 SAVINGS_ACCOUNT 테이블이 들어 있는 데이터베이스와 함께 작업 단위(UOW)를 등록합니다. 응답이 다시 클라이언트에 전송되어 작업 단위(UOW)가 성공적으로 등록되었음을 알립니다.

5 SAVINGS_DB 데이터베이스에 나타난 SQL문은 정상적인 방식으로 처리됩니다. 각 명령문에 대한 응답이 프로그램에서 embedded SQL문으로 작업하는 중에 SQLCA에서 리턴됩니다. (SQLCA는 *응용프로그램 개발 안내서* 및 *SQL 참조서*에 설명되어 있습니다.)

6 트랜잭션 ID는 작업 단위(UOW) 내의 해당 데이터베이스에 처음 액세스할 때 TRANSACTION_FEE 테이블이 들어 있는 FEE_DB 데이터베이스에 등록됩니다.

7 FEE_DB에 대한 모든 SQL문은 정상적인 방법으로 처리됩니다.

8 추가 SQL문은 적절하게 연결을 설정함으로써 SAVINGS_DB 데이터베이스에 대해 실행될 수 있습니다. 작업 단위(UOW)는 이미 SAVINGS_DB 데이터베이스 **4** 로 등록이 되어 있어, 데이터베이스 클라이언트는 등록 단계를 다시 실행할 필요가 없습니다.

9 CHECKING_DB 데이터베이스에 연결하고 사용하는 것은 **6** 및 **7** 에서 설명한 동일한 규칙을 따릅니다.

10 데이터베이스 클라이언트가 작업 단위(UOW)가 확약될 것을 요청하는 경우, *준비* 메시지는 작업 단위(UOW)의 모든 관련된 데이터베이스로 송신됩니다. 각 데이터베이스는 ″PREPARED″ 레코드를 관련 로그 파일에 쓰며, 데이터베이스 클라이언트에 응답합니다.

11 데이터베이스 클라이언트가 모든 데이터베이스로부터 긍정 응답을 수신한 후, 트랜잭션 관리 프로그램 데이터베이스로 작업 단위(UOW)는 지금 확약될 준비가 되어 있습니다(PREPARED)라는 메세지를 송신합니다. 트랜잭션 관리 프로그램 데이터베이스는 ″PREPARED″ 레코드를 로그 파일에 쓰고 응답을 보내 확약 프로세스의 두 번째 단계가 시작될 수 있음을 클라이언트에 알립니다.

12 확약 프로세스의 2단계중에, 데이터베이스 클라이언트는 모든 관련된 데이터베이스로 메세지를 송신하여 데이터베이스에 확약을 알립니다. 각 데이터베이스는 ″COMMITTED″ 레코드를 해당 로그 파일에 기록하고 이 작업 단위(UOW)에 대해 보유된 잠금을 해제합니다. 데이터베이스는 변경사항 확약을 완료하면 응답을 클라이언트에 보냅니다.

13 데이터베이스 클라이언트가 모든 참여 데이터베이스로부터 긍정적인 응답을 수신한 후, 트랜잭션 관리 프로그램 데이터베이스로 메시지를 보내어 작업 단위(UOW)가 완료되었음을 알립니다. 그러면 트랜잭션 관리 프로그램 데이터베이스는 ″COMMITTED″ 레코드를 관련 로그 파일에 기록하면, 작업 단위(UOW)가 완료되었음을 나타내며 클라이언트에 완료했다고 응답합니다.

2단계 확약 중의 문제점 복구

오류 조건을 복구하는 것은 응용프로그램 프로그래밍, 시스템 관리, 데이터베이스 관리 및 시스템 조작과 연관된 일상적인 작업입니다. 여러 원격 서버에 걸쳐 데이터베이스를 분배하면 네트워크 또는 통신 실패의 결과로 오류의 발생 가능성이 높아집니다. 데이터 무결성을 보장하기 위해 데이터베이스 관리 프로그램은 159 페이지의 『2단계 확약 프로세스 이해』(**10** , **11** 및 **12**)에서 설명한 대로 2단계 확약 프로세스를 제공합니다. 다음은 데이터베이스 관리 프로그램이 2단계 확약 프로세스 동안 오류를 처리하는 방법을 설명합니다.

- **첫 번째 단계 오류**

  데이터베이스가 작업 단위(UOW)를 확약하기 위한 준비에 실패했다는 것을 수신하는 경우, 데이터베이스 클라이언트는 확약 프로세스의 두 번째 단계 중 작

업 단위(UOW)를 구간 복원합니다. 이 경우에는 준비 메세지가 트랜잭션 관리 프로그램 데이터베이스에 전송되지 *않습니다*.

두 번째 확약 단계 동안, 클라이언트는 첫 번째 단계동안 성공적으로 확약을 준비한 모든 참여 데이터베이스로 구간 복원 메세지를 전송합니다. 그런 다음 각 데이터베이스는 "ABORT" 레코드를 해당 로그 파일에 기록하고 이 작업 단위(UOW)에 대해 보유된 잠금을 해제합니다.

- **두 번째 단계 오류**

  이 단계에서의 오류 처리는 두 번째 단계가 트랜잭션을 확약하느냐 또는 구간 복원하느냐에 달려 있습니다. 두 번째 단계는 첫 번째 단계에서 오류가 발생한 경우에만 트랜잭션을 구간 복원합니다.

  참여하고 있는 데이터베이스 중 하나가 작업 단위(UOW) 확약에 실패할 경우(통신 실패의 가능성이 있음), 트랜잭션 관리 프로그램 데이터베이스는 실패한 데이터베이스에 대해 확약을 재시도합니다. 그러나, 응용프로그램은 확약이 SQLCA를 통해 성공적이었음을 알립니다. DB2는 데이터베이스 서버 내의 미확약 트랜잭션이 확약됨을 보장합니다. 데이터베이스 관리 프로그램 구성 매개변수 *resync_interval*(*관리 안내서: 성능*의 "DB2 구성" 참조)은 작업 단위(UOW)를 확약하려는 시도 사이에서 트랜잭션 관리 프로그램 데이터베이스가 대기하게 될 시간을 지정하는 데 사용됩니다. 작업 단위(UOW)가 확약될 때까지 모든 잠금은 데이터베이스 서버에서 보류됩니다.

  트랜잭션 관리 프로그램 데이터베이스가 실패하는 경우, 데이터베이스가 다시 시작될 때 작업 단위를 재동기화합니다. 재동기화 프로세스는 모든 *2단계 확약 중 이상 실패 트랜잭션*, 즉 1단계를 끝마치고 2단계 확약을 완료하지 않은 트랜잭션을 완료하려고 시도합니다. 트랜잭션 관리 프로그램 데이터베이스과 연관된 데이터베이스 관리 프로그램은 다음에 의해 재동기화를 수행합니다.

  1. 확약 프로세스의 첫 번째 단계 동안 확약할 『PREPARED』라고 표시된 데이터베이스에 연결
  2. 해당 데이터베이스에서 2단계 확약 중 이상 실패 트랜잭션을 확약하려고 시도(2단계 확약 중 이상 실패 트랜잭션을 찾을 수 없으면, 데이터베이스 관리 프로그램에서는 데이터베이스가 확약 프로세스의 2단계 동안 정상적으로 트랜잭션을 확약했다고 가정합니다.)

3. 모든 2단계 확약 중 이상 실패 트랜잭션이 참여 데이터베이스에서 확약된 후, 트랜잭션 관리 프로그램 데이터베이스에서 2단계 확약 중 이상 실패 트랜잭션을 확약

   참여 데이터베이스 중 하나가 실패하여 재시작되면, 이 데이터베이스의 데이터베이스 관리 프로그램은 트랜잭션이 구간 복원되어야 하는지 여부를 판별하기 위해 트랜잭션 관리 프로그램 데이터베이스에 조회하여 이 트랜잭션의 상태를 알아봅니다. 트랜잭션이 로그에 없으면, 데이터베이스 관리 프로그램에서는 트랜잭션이 구간 복원되었고, 이 데이터베이스의 2단계 확약 중 이상 실패 트랜잭션을 구간 복원할 것이라고 가정합니다. 그렇지 않으면, 데이터베이스는 트랜잭션 관리 프로그램 데이터베이스로부터의 확약 요청을 기다립니다.

   트랜잭션 프로세싱 모니터(XA 준수 트랜잭션 관리 프로그램)가 트랜잭션을 조정한 경우, 데이터베이스는 항상 트랜잭션 프로세싱 모니터에 의거하여 재동기화를 시작합니다.

몇 가지 이유로 인해 사용자가 2단계 확약 중 이상 실패 트랜잭션을 자동으로 해결하기 위해 트랜잭션 관리 프로그램을 기다릴 수 없는 경우, 사용자가 2단계 확약 중 이상 실패 트랜잭션을 수동으로 해결하기 위해 취할 수 있는 조치가 있습니다. 이러한 수동 프로세스를 때때로 "경험적 결정"이라고 합니다. 2단계 확약 중 이상 실패 트랜잭션의 수동 복구에 대해서는 180 페이지의 『경험적 결정』에서 자세한 내용을 참조하십시오.

AUTORESTART=OFF인 경우 2단계 확약 중 이상 실패 트랜잭션 재동기화

DB2 Universal Database 2단계 확약 환경 구성시 고려사항은 155 페이지의 『기타 구성 고려사항』에서 자세한 내용을 참조하십시오.

특히, *자동 재시작* 데이터베이스 구성 매개변수가 OFF이고, TM 또는 RM 데이터베이스에 2단계 확약 중 이상 실패 트랜잭션이 있을 경우, 재동기화 프로세스를 시작하기 위해 RESTART DATABASE 명령이 필요합니다. 명령행 처리기에서 RESTART DATABASE 명령을 발행할 때, 다른 세션을 사용하십시오. 동일한 세션에서 다른 데이터베이스를 다시 시작할 경우, 이전 호출에 의해 형성된 연결이 삭제되며, 다시 한번만 재시작해야 합니다. LIST INDOUBT TRANSACTIONS

명령에 의해 더 이상 2단계 확약 중 이상 실패 트랜잭션이 리턴되지 않는 경우, TERMINATE 명령을 발행하여 연결을 삭제하십시오.

제10장 트랜잭션 관리 프로그램 설계

2단계 확약 트랜잭션에 참여하고자 하는 DB2 데이터베이스 이외의 다른 자원이 있는 경우, XA 준수 트랜잭션 관리 프로그램으로 사용자의 데이터베이스를 사용하고자 할 수 있습니다. 사용자의 트랜잭션이 DB2 데이터베이스만을 액세스하는 경우는 151 페이지의 『다중 데이터베이스 갱신』에 설명되어 있는 대로 DB2 트랜잭션 관리 프로그램을 사용해야 합니다.

다음 주제 항목은 IBM TXSeries CICS, IBM TXSeries Encina, BEA Tuxedo 또는 Microsoft Transaction Server와 같은 XA 준수 트랜잭션 관리 프로그램과 함께 데이터베이스 관리 프로그램을 사용하는 데 도움이 됩니다.

- 168 페이지의 『X/Open 분산 트랜잭션 프로세싱 모델』
- 173 페이지의 『데이터베이스를 자원 관리 프로그램으로 설정』
- 173 페이지의 『xa_open 및 xa_close 문자열 사용법』
- 173 페이지의 『DB2 버전 7의 새 xa_open 문자열 형식』
- 175 페이지의 『TPM 및 TP_MON_NAME 값』
- 179 페이지의 『이전 DB2 버전의 xa_open 문자열 형식』
- 179 페이지의 『호스트 또는 AS/400 데이터베이스 서버 갱신』
- 180 페이지의 『데이터베이스 연결 고려사항』
- 180 페이지의 『경험적 결정』
- 184 페이지의 『보안 고려사항』
- 185 페이지의 『구성 고려사항』
- 186 페이지의 『지원된 XA 함수』
- 189 페이지의 『XA 인터페이스 문제점 판별』
- 190 페이지의 『DB2 UDB를 사용하기 위한 XA 트랜잭션 관리 프로그램 구성』
- 196 페이지의 『Microsoft Transaction Server 구성』

XA 준수 트랜잭션 관리 프로그램을 사용하거나 하나를 구현하는 경우, 기술 지원 웹 사이트에서 자세한 내용을 참조하십시오.

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

거기에서 한번만, "DB2 Universal Database"를 선택한 다음, XA 준수 트랜잭션 관리 프로그램에서 사용 가능한 최신 정보에 대한 키워드 "XA"를 사용하여 웹 사이트를 검색하십시오.

X/Open 분산 트랜잭션 프로세싱 모델

X/Open 분산 트랜잭션 프로세싱(DTP) 모델은 서로 관계가 있는 세가지 구성요소를 포함합니다.

- 169 페이지의『응용프로그램(AP)』
- 170 페이지의『트랜잭션 관리 프로그램(TM)』
- 171 페이지의『자원 관리 프로그램(RM)』

그림35에서는 이 모델을 예시하며 이 구성요소 간에 관계를 표시합니다.

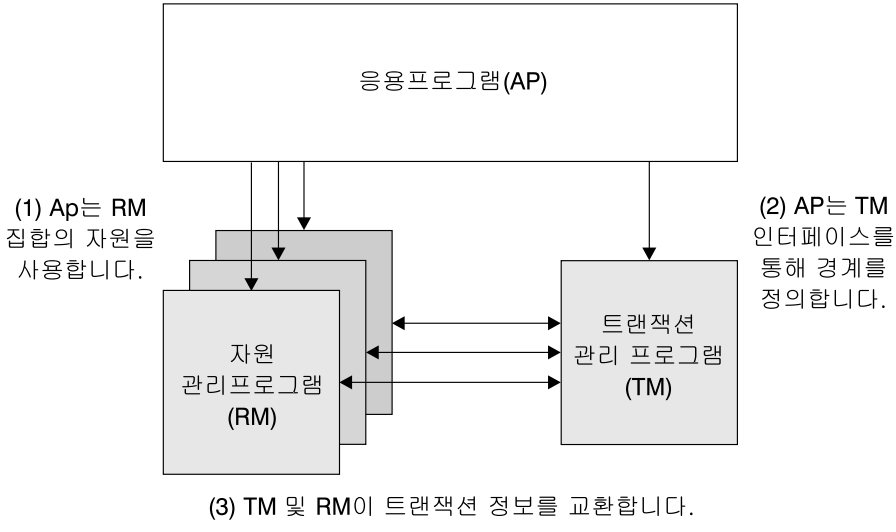

그림 35. X/Open 분산 트랜잭션 프로세싱(DTP) 모델

응용프로그램(AP)

응용프로그램(AP)은 트랜잭션 경계를 정의하고, 트랜잭션을 구성하는 응용프로그램 고유의 조치를 정의합니다.

예를 들어, CICS* 응용프로그램은 데이터베이스 및 CICS 임시 데이터 대기행렬과 같은 자원 관리 프로그램(RM)을 액세스하고 데이터를 조작하기 위해 프로그래밍 논리를 사용하고자 할 수 있습니다. 각 액세스 요청은 해당 RM에 고유한 함수 호출을 통해 해당 자원 관리 프로그램에 전달됩니다. DB2의 경우, 이것은 각 SQL문에 대해 DB2 사전 처리 컴파일러가 생성한 함수 호출이거나 프로그래머가 API를 사용하여 직접 코딩한 데이터베이스 호출입니다.

트랜잭션 관리 프로그램(TM)은 대개 사용자의 응용프로그램을 수행하는 트랜잭션 프로세싱(TP) 모니터를 포함하고 있습니다. TP 모니터는 응용프로그램이 트랜잭션을 시작하고 종료할 수 있도록 하며, 응용프로그램을 실행하고자 하는 여러 사용자간에 응용프로그램 스케줄링과 작업량 조정을 수행하는 API를 제공합니다. 분산 트랜잭션 프로세싱(DTP) 환경 내의 응용프로그램은 실제로 사용자 응용프로그램과 TP 모니터의 조합입니다.

유효한 온라인 트랜잭션 프로세싱(OLTP) 환경을 손쉽게 하기 위해, TP 모니터는 시동시에 많은 서버 프로세스를 미리 할당한 후, 많은 사용자 트랜잭션중에서 프로세스를 스케줄링하여 재사용합니다. 이는 보다 적은 수의 서버 프로세스와 해당 RM 프로세스로 보다 많은 수의 동시 사용자를 지원할 수 있도록 함으로써 시스템 자원을 보존합니다. 또한 이렇게 프로세스를 재사용하게 되면, 각 사용자 트랜잭션 또는 프로그램당 TM과 RM에서 한 프로세스를 시작해야 하는 오버헤드를 피할 수 있습니다. (한 프로그램은 하나 이상의 트랜잭션을 호출합니다.) 이것은 서버 프로세스가 TM과 RM에 대한 실제적인 "사용자 프로세스"임을 의미하기도 합니다. 이것은 보안 관리 및 응용프로그램 프로그래밍과 연관됩니다. 세부사항은 184 페이지의 『보안 고려사항』을 참조하십시오.

다음과 같은 유형의 트랜잭션이 TP 모니터에서 가능합니다.

- 비 XA 트랜잭션

이 트랜잭션은 TM에 정의되지 않은 RM과 관련되므로, TM의 2단계 확약 프로토콜 하에서는 조정되지 않습니다. 이것은 응용프로그램이 XA 인터페이스를 지원하지 않는 RM을 액세스할 필요가 있는 경우에 필요할 수 있습니다. TP 모니터는 단순히 응용프로그램과 작업량 조정의 유효한 스케줄링을 제공합니다. TM은 XA 처리를 위해 RM을 명시적으로 "열지" 않기 때문에, RM은 응용프로그램을 비 DTP 환경에서 수행되는 다른 응용프로그램으로 취급합니다.

- 전역 트랜잭션

  이들 트랜잭션은 TM에 정의된 RM과 관련되므로 TM의 2단계 확약 제어하에 있습니다. 전역 트랜잭션은 하나 이상의 RM이 관련될 수 있는 작업 단위(UOW)입니다. *트랜잭션 분기*는 전역 트랜잭션을 지원하는 TM과 RM 사이의 작업의 일부입니다. 전역 트랜잭션은 TM에 의해 조정되는 하나 이상의 응용프로그램 프로세스를 통해 여러 RM을 액세스할 때 다중 트랜잭션 분기를 가질 수 있습니다.

  느슨하게 연결된 전역 트랜잭션은 다수의 각 응용프로그램 프로세스가 별도의 전역 트랜잭션에 있는 것처럼 RM을 액세스할 때 존재하지만, 그러한 응용프로그램은 TM의 조정하에 있습니다. 각 응용프로그램 프로세스는 RM 내에 자체의 트랜잭션 분기를 갖습니다. 트랜잭션 분기는 AP, TM 또는 RM 중 어느 하나가 확약 또는 구간 복원을 요청할 때 모두 완료됩니다. 분기 사이에 자원 교착 상태가 발생하지 않도록 하는 것은 응용프로그램의 책임입니다. (SYNCPOINT (TWOPHASE) 옵션을 사용하여 준비된 응용프로그램에 대해 DB2 트랜잭션 관리 프로그램이 수행한 트랜잭션 조정은 이러한 느슨하게 연결된 전역 트랜잭션과 거의 동일하다는 점을 알아두십시오. 151 페이지의 『다중 데이터베이스 갱신』에서 자세한 내용을 참조하십시오.)

  밀접하게 연결된 전역 트랜잭션은 여러 응용프로그램 프로세스가 RM에서 같은 트랜잭션 분기 하에서 차례로 작업할 때 존재합니다. RM에 대해 두 응용프로그램 프로세스는 하나의 엔터티입니다. RM은 자원 교착 상태가 트랜잭션 분기 안에서 발생하지 않도록 해야 합니다.

트랜잭션 관리 프로그램(TM)

트랜잭션 관리 프로그램(TM)은 트랜잭션에 식별자를 할당하고, 트랜잭션의 진행을 모니터하며, 트랜잭션 완료와 실패에 대해 책임을 집니다. 트랜잭션 분기 식별자(XID로 알려져 있음)는 TM에 의해 할당되어 전역 트랜잭션과 RM 안의 특정 분

기를 식별합니다. 이것은 TM에서의 로그와 RM에서의 로그 사이의 상관관계 토큰입니다. XID는 2단계 확약이나 구간 복원, 시스템 시동시 *재동기화 조작(resync* 라고도 함) 수행 또는 필요할 경우 관리자가 *경험적 조작*(수동 *개입*이라고도 함)을 수행하는 데 필요합니다.

TP 모니터는 시동된 후, TM에게 응용프로그램 서버 세트가 정의한 모든 RM을 열라고 요청합니다. TM은 DTP 처리를 위해 RM이 시작될 수 있도록 RM에 **xa_open** 호출을 전달합니다. 이 시동 절차의 일부로서 TM은 재동기화를 수행하여 모든 *2단계 확약 중 이상 실패 트랜잭션*을 복구합니다. 2단계 확약 중 이상 실패 트랜잭션은 불확실 상태로 남아 있던 전역 트랜잭션입니다. 이것은 TM이나 최소한 하나의 RM이 2단계 확약 프로토콜의 첫 번째 단계(즉, 준비 단계)를 성공적으로 완료한 후 사용할 수 없게 될 때 발생합니다. RM은 다시 사용할 수 없게 될 때, TM이 RM 로그와 자체의 로그를 통합할 수 있을 때까지 트랜잭션 분기를 확약할 것인지 구간 복원할 것인지를 모르게 됩니다. 재동기화 조작을 수행하기 위해, TM은 각 RM에 대해 한 번 이상 **xa_recover** 호출을 발행하여 모든 2단계 확약 중 이상 실패 트랜잭션을 식별합니다. TM은 응답을 자체의 로그에 있는 정보와 비교하여 RM에 해당 트랜잭션을 **xa_commit**할 것인지 또는 **xa_rollback** 할 것인지를 알려야 하는지 결정합니다. RM이 이미 관리자에 의한 경험적 조작을 통해 2단계 확약 중 이상 실패 트랜잭션의 분기를 확약하거나 구간 복원한 경우, TM은 해당 RM에 대해 **xa_forget** 호출을 발행하여 재동기화 조작을 완료합니다.

사용자 응용프로그램이 확약 또는 구간 복원을 요청할 때, TM이 관련되는 모든 RM 사이에서 확약 및 구간 복원을 조정할 수 있도록 TP 모니터 또는 TM이 제공하는 API를 사용해야 합니다. 예를 들어, CICS 응용프로그램이 CICS SYNCPOINT 요청을 발행하여 트랜잭션을 확약할 때, CICS XA TM(Encina Server에서 구현)은 **xa_end**, **xa_prepare**, **xa_commit** 또는 **xa_rollback**과 같은 XA 호출을 차례로 발행하여 RM이 트랜잭션을 확약하거나 구간 복원하도록 요청합니다. TM은 하나의 RM만 관련되거나 RM이 분기가 읽기 전용이라고 응답하는 경우, 2단계 확약 대신 1단계를 선택하여 사용할 수 있습니다.

자원 관리 프로그램(RM)

자원 관리 프로그램(RM)은 데이터베이스와 같은 공유 자원에 대한 액세스를 제공합니다.

데이터베이스의 자원 관리 프로그램으로서의 DB2는 XA 준수 TM에서 조정되고 있는 *전역 트랜잭션*에 참여할 수 있습니다. XA 인터페이스가 요구할 때, 데이터베이스 관리 프로그램은 xa_switch_t 유형의 *db2xa_switch* 외부 C 변수를 제공하여 TM에 XA 스위치 구조를 되돌려 줍니다. 이 데이터 구조는 TM에 의해 호출될 다양한 XA 루틴의 주소와 RM의 운영 특성을 포함하고 있습니다. 데이터베이스 관리 프로그램이 지원하는 XA 함수에 대해서는 186 페이지의 『지원된 XA 함수』에서 자세한 내용을 참조하십시오.

RM이 각 전역 트랜잭션에 등록하는 방법에는 *정적* 등록 및 *동적* 등록의 두 가지 방법이 있습니다.

- 정적 등록은 특정 RM이 트랜잭션에 의해 사용되느냐 사용되지 않느냐의 여부에 상관없이 서버 응용프로그램에 대해 정의된 모든 RM에 매 트랜잭션마다 **xa_start**, **xa_end** 및 **xa_prepare** 호출을 발행하도록 요구합니다. 모든 RM이 모든 트랜잭션에 관련되지 않고 비효율성이 정의된 RM의 수와 비례하는 경우 이것은 무효합니다.
- 동적 등록(DB2에서 사용)은 유연성이 있으며 효과적입니다. RM은 자원에 대한 요청을 받을 때에만 **ax_reg** 호출을 사용하여 TM에 등록합니다. 단 하나의 RM이 정의되어 있을 때 또는 **ax_reg** 및 **xa_start**호출이 TM에서와 유사한 경로를 가지고 있기 때문에, 모든 RM이 모든 트랜잭션에 의해 사용될 때에도 이 방법에는 성능상의 불리한 점이 없음에 유의하십시오.

XA 인터페이스는 TM과 RM 사이에 양방향 통신을 제공합니다. 그것은 응용프로그램 개발자가 코딩하는 보통의 응용프로그램 인터페이스가 아니라, 두 DTP 소프트웨어 구성요소 사이의 시스템 레벨 인터페이스입니다. 그러나, 응용프로그램 개발자는 프로그래밍과 DTP 소프트웨어 구성요소가 부과하는 제한사항에 익숙해야 합니다. X/Open XA 인터페이스 프로그래밍 고려사항에 대해서는 *응용프로그램 개발 안내서*에서 자세한 내용을 참조하십시오.

XA 인터페이스가 다르지 않더라도, 각 XA 준수 TM은 RM을 통합하는 방식이 고유한 제품을 가질 수 있습니다. DB2 제품을 자원 관리 프로그램으로서 특정 트랜잭션 관리 프로그램과 통합하는 데 대한 정보는 적절한 TM 제품 데이터를 참조하십시오. 대부분의 일반 TP 모니터에 대한 통합 정보가 190 페이지의 『DB2 UDB를 사용하기 위한 XA 트랜잭션 관리 프로그램 구성』에 제공됩니다.

데이터베이스를 자원 관리 프로그램으로 설정

각 데이터베이스는 트랜잭션 관리 프로그램(TM)의 별도의 자원 관리 프로그램(RM)로 정의되며, 데이터베이스는 xa_open 문자열로 식별되어야 합니다. DB2의 xa_open 문자열 형식에 대해서는 『xa_open 및 xa_close 문자열 사용법』에서 자세한 내용을 참조하십시오.

xa_open 및 xa_close 문자열 사용법

데이터베이스 관리 프로그램 xa_open 문자열은 두 개의 승인된 형식을 갖습니다. 하나의 형식은 DB2 버전 7에 새로운 것입니다. 두 번째 형식은 이전 DB2 버전에 의해 사용되며 이전 레벨과 호환성을 갖습니다. 새로운 구현은 새로운 형식을 사용해야 하며, 이전 구현은 가능할 때 새로운 형식으로 이주되어야 합니다. DB2의 장래 버전은 이전 xa_open 문자열 형식을 지원하지 않을 수 있습니다. 179 페이지의 『이전 DB2 버전의 xa_open 문자열 형식』에서 원래의 xa_open 문자열 형식에 대한 자세한 내용을 참조하십시오.

데이터베이스를 자원 관리 프로그램으로 설정할 때에는 xa_close 문자열이 필요하지 않습니다. 이 문자열이 제공되더라도 데이터베이스 관리 프로그램에 의해 무시됩니다.

DB2 버전 7의 새 xa_open 문자열 형식

다음 xa_open 문자열 형식은 DB2 버전 7에 새로운 것입니다.

```
parm_id1 = <parm value>,parm_id2 = <parm value>, ...
```

어떤 순서로 이 매개변수를 지정해야 하는가는 중요하지 않습니다. *parm_id*에 대한 유효한 값이 다음 테이블에 기술됩니다.

표 22. parm_id에 대한 유효한 값

매개변수 이름	값	필수?	대소문자 구별?	기본값
DB	데이타베이스 별명	예	아니오	없음
데이터베이스에 액세스하기 위해 응용프로그램이 사용하는 데이터베이스 별명.				
UID	사용자 ID	아니오	예	없음
데이터베이스에 연결하는 권한을 가진 사용자 ID. 암호가 지정된 경우 필수.				
PWD	암호	아니오	예	없음

표 22. parm_id에 대한 유효한 값 (계속)

매개변수 이름	값	필수?	대소문자 구별?	기본값
사용자 ID와 연관된 암호. 사용자 ID가 지정된 경우 필수.				
TPM	트랜잭션 프로세싱 모니터 이름	아니오	아니오	없음
사용 중인 TP 모니터의 이름. 175 페이지의 『TPM 및 TP_MON_NAME 값』에서 지원되는 값에 대한 자세한 내용을 참조하십시오. 이 매개변수는 다중 TP 모니터가 단일 DB2 인스턴스를 사용하도록 허용하기 위해 지정될 수 있습니다. 지정된 값은 *tp_mon_name* 데이터베이스 관리 프로그램 구성 매개변수에 지정된 값을 겹쳐씁니다.				
AXLIB	TP 모니터의 **ax_reg** 및 **ax_unreg** 함수가 들어 있는 라이브러리.	아니오	예	없음
이 값은 DB2가 사용하여 필수 **ax_reg** 및 **ax_unreg** 함수의 주소를 얻습니다. TPM 매개변수에 근거하여 가정된 값을 겹쳐쓰기하거나, TPM용 목록에 표시되지 않은 TP 모니터에 의해 사용될 수 있습니다.				
CHAIN_END	xa_end 체인 플래그. 유효한 값은 T, F 또는 값 없음입니다.	아니오	아니오	F
XA_END 체인은 DB2가 네트워크 흐름을 줄이기 위해 사용할 수 있는 최적화입니다. TP 모니터 환경이 **xa_prepare**가 **xa_end**에 대한 호출 바로 다음에 오는 동일한 스레드 또는 프로세스 내에서 호출됨을 보증할 수 있는 경우 그리고 CHAIN_END가 설정된 경우, xa_end 플래그는 **xa_prepare** 명령과 연결되어 하나의 네트워크 흐름을 줄입니다. T 값은 CHAIN_END가 설정되어 있음을 의미하며, F 값은 CHAIN_END가 해제되어 있음을 의미하며, 지정되지 않은 값은 CHAIN_END가 설정되어 있음을 의미합니다. 이 매개변수는 지정된 TPM 값에서 파생된 설정을 겹쳐쓰는 데 사용될 수 있습니다.				
SUSPEND_CURSOR	제어의 트랜잭션 스레드가 일시 중지될 때 커서가 유지되는지 여부를 지정합니다. 유효한 값은 T, F 또는 값 없음입니다.	아니오	아니오	F

표 22. parm_id에 대한 유효한 값 (계속)

매개변수 이름	값	필수?	대소문자 구별?	기본값
트랜잭션 분기를 일시 중지시키는 TP 모니터는 다른 트랜잭션의 일시 중지된 스레드 또는 프로세스를 재사용할 수 있습니다. 이러한 상황에서, 커서가 닫히므로 새로운 트랜잭션은 커서를 계승하지 않습니다. 일시 중지된 트랜잭션이 재개되면, 응용프로그램은 다시 커서를 확보해야 합니다. SUSPEND_CURSOR가 설정되면, 임의의 열린 커서는 닫히지 않지만, 스레드 또는 프로세스는 다른 트랜잭션용으로 재사용될 수 없습니다. 일시 중지된 트랜잭션의 재개만이 허용됩니다. T 값은 SUSPEND_CURSOR가 설정되어 있음을 의미하며, F 값은 SUSPEND_CURSOR가 해제되어 있음을 의미하며, 지정되지 않은 값은 SUSPEND_CURSOR가 설정되어 있음을 의미합니다. 이 매개변수는 지정된 TPM 값에서 파생된 설정을 겹쳐쓰는 데 사용될 수 있습니다.				
HOLD_CURSOR	커서가 트랜잭션 확약에서 보류되는지 여부를 지정합니다. 유효한 값은 T, F 또는 값 없음입니다.	아니오	아니오	F
TP 모니터는 보통 다중 응용프로그램의 스레드나 프로세스를 재사용합니다. 새로 로드된 응용프로그램이 이전 응용프로그램에 의해 열린 커서를 계승하지 않음을 보장하기 위해, 커서가 확약 이후에 닫힙니다. HOLD_CURSOR가 설정되면, 커서는 트랜잭션 확약에서 보류됩니다. T 값은 HOLD_CURSOR가 설정되어 있음을 의미하며, F 값은 HOLD_CURSOR가 해제되어 있음을 의미하며, 지정되지 않은 값은 HOLD_CURSOR가 설정되어 있음을 의미합니다. 이 매개변수는 지정된 TPM 값에서 파생된 설정을 겹쳐쓰는 데 사용될 수 있습니다.				

TPM 및 TP_MON_NAME 값

xa_open 문자열 TPM 매개변수 및 *tp_mon_name* 데이터베이스 관리 프로그램 구성 매개변수는 TP 모니터가 사용 중인 DB2에 나타내기 위해 사용됩니다. *tp_mon_name* 값은 전체 DB2 인스턴스에 적용합니다. TPM 매개변수는 특정 XA 자원 관리 프로그램에만 적용합니다. TPM 값은 *tp_mon_name* 매개변수를 겹쳐쓰기합니다. TPM 및 *tp_mon_name* 매개변수의 유효한 값은 다음과 같습니다.

표 23. TPM 및 tp_mon_name에 대한 유효한 값

TPM 값	TP 모니터 제품	내부 설정
CICS	IBM TxSeries CICS	AXLIB=libEncServer (for Windows) =/usr/lpp/encina/lib/libEncServer (for UNIX based systems) HOLD_CURSOR=T CHAIN_END=T SUSPEND_CURSOR=F

표 23. TPM 및 tp_mon_name에 대한 유효한 값 (계속)

TPM 값	**TP 모니터 제품**	**내부 설정**
ENCINA	IBM TxSeries Encina Monitor	AXLIB=libEncServer (for Windows) =/usr/lpp/encina/lib/libEncServer (for UNIX based systems) HOLD_CURSOR=F CHAIN_END=T SUSPEND_CURSOR=F
MQ	IBM MQSeries	AXLIB=mqmax (for Windows) =/usr/mqm/lib/libmqmax.a (for AIX) =/opt/mqm/lib/libmqmax.a (for Solaris) HOLD_CURSOR=F CHAIN_END=F SUSPEND_CURSOR=F
CB	IBM 구성요소 브로커	AXLIB=somtrx1i (for Windows) =libsomtrx1 (for UNIX based systems) HOLD_CURSOR=F CHAIN_END=T SUSPEND_CURSOR=F
SF	IBM San Francisco	AXLIB=ibmsfDB2 HOLD_CURSOR=F CHAIN_END=T SUSPEND_CURSOR=F
TUXEDO	BEA Tuxedo	AXLIB=libtux HOLD_CURSOR=F CHAIN_END=F SUSPEND_CURSOR=F
MTS	Microsoft Transaction Server	MTS용 DB2를 구성하는 것이 필요하지 않습니다. MTS는 자동으로 DB2의 ODBC 드라이버에 의해 검출됩니다.
JTA	Java 트랜잭션 API	IBM WebSphere와 같은 EJS(Enterprise Java Servers)용 DB2를 구성하는 것이 필요하지 않습니다. DB2의 JDBC 드라이버는 자동으로 이 환경을 검출합니다.

예

1. Windows NT에서 IBM TxSeries CICS를 사용 중입니다. TxSeries 문서는 `libEncServer:C` 값으로 *tp_mon_name*을 구성할 필요가 있음을 나타냅니다. 이것은 여전히 채택 가능한 형식이지만, DB2 UDB 또는 DB2 Connect 버전 7 사용시 다음의 옵션을 갖습니다.
   - CICS의 *tp_mon_name* 지정(이 시나리오에 권장됨)

     ```
     db2 update dbm cfg using tp_mon_name CICS
     ```

     지역 —> 자원 —> 제품 —> XAD —> 자원 제품 초기설정 문자열에서 CICS에 정의된 각 데이터베이스의 경우, 다음을 지정하십시오.

     ```
     db=dbalias,uid=userid,pwd=password
     ```
   - 지역 —> 자원 —> 제품 —> XAD —> 자원 제품 초기설정 문자열에서 CICS에 정의된 각 데이터베이스의 경우, 다음을 지정하십시오.

     ```
     db=dbalias,uid=userid,pwd=password,tpm=cics
     ```
2. Windows NT에서 IBM MQSeries를 사용 중입니다. MQSeries 문서는 `mqmax` 값으로 *tp_mon_name*을 구성할 필요가 있음을 나타냅니다. 이것은 여전히 채택 가능한 형식이지만, DB2 UDB 또는 DB2 Connect 버전 7 사용시 다음의 옵션을 갖습니다.
   - MQ의 *tp_mon_name* 지정(이 시나리오에 권장됨)

     ```
     db2 update dbm cfg using tp_mon_name MQ
     ```

     지역 —> 자원 —> 제품 —> XAD —> 자원 제품 초기설정 문자열에서 CICS에 정의된 각 데이터베이스의 경우, 다음을 지정하십시오.

     ```
     uid=userid,db=dbalias,pwd=password
     ```
   - 지역 —> 자원 —> 제품 —> XAD —> 자원 제품 초기설정 문자열에서 CICS에 정의된 각 데이터베이스의 경우, 다음을 지정하십시오.

     ```
     uid=userid,db=dbalias,pwd=password,tpm=mq
     ```
3. IBM TxSeries CICS 및 IBM MQSeries 둘다를 WIndows NT에서 사용 중입니다. 단일 DB2 인스턴스를 사용 중입니다. 이 시나리오에서, 다음과 같이 구성합니다.

a. 지역 —> 자원 —> 제품 —> XAD —> 자원 제품 초기설정 문자열에서 CICS에 정의된 각 데이터베이스의 경우, 다음을 지정하십시오.

```
pwd=password,uid=userid,tpm=cics,db=dbalias
```

b. 대기행렬 관리 프로그램 등록 정보에서 자원으로서 정의된 각 데이터베이스의 경우, XaOpenString을 다음과 같이 지정하십시오.

```
db=dbalias,uid=userid,pwd=password,tpm=mq
```

4. Windows NT에서 사용자 자신의 XA 준수 트랜잭션 관리 프로그램(XA TM)을 Windows NT에서 개발하는 중이며, 사용자는 라이브러리 ″myaxlib″가 필수 함수 **ax_reg** 및 **ax_unreg**를 가지고 있음을 DB2에 알리려고 합니다. 라이브러리 ″myaxlib″는 PATH문에 지정된 디렉토리에 있습니다. 다음 옵션을 갖습니다.

   • myaxlib의 *tp_mon_name* 지정

   ```
   db2 update dbm cfg using tp_mon_name myaxlib
   ```

   그리고 XA TM에 정의된 각 데이터베이스의 경우, xa_open 문자열 지정

   ```
   db=dbalias,uid=userid,pwd=password
   ```

   • XA TM에 정의된 각 데이터베이스의 경우, xa_open 문자열 지정

   ```
   db=dbalias,uid=userid,pwd=password,axlib=myaxlib
   ```

5. Windows NT에서 사용자 자신의 XA 준수 트랜잭션 관리 프로그램(XA TM)을 Windows NT에서 개발하는 중이며, 사용자는 라이브러리 ″myaxlib″가 필수 함수 **ax_reg** 및 **ax_unreg**를 가지고 있음을 DB2에 알리려고 합니다. 라이브러리 ″myaxlib″는 PATH문에 지정된 디렉토리에 있습니다. 또한 XA END 체인을 사용할 수 있게 하려고 합니다. 다음 옵션을 갖습니다.

   • XA TM에 정의된 각 데이터베이스의 경우, xa_open 문자열 지정

   ```
   db=dbalias,uid=userid,pwd=password,axlib=myaxlib,chain_end=T
   ```

   • XA TM에 정의된 각 데이터베이스의 경우, xa_open 문자열 지정

   ```
   db=dbalias,uid=userid,pwd=password,axlib=myaxlib,chain_end
   ```

이전 DB2 버전의 xa_open 문자열 형식

DB2의 이전 버전은 여기에서 설명한 xa_open 문자열 형식을 사용했습니다. 이 형식은 호환성 이유로 여전히 지원됩니다. 응용프로그램은 가능할 때 새로운 형식으로 이주되어야 합니다(173 페이지의 『DB2 버전 7의 새 xa_open 문자열 형식』 참조).

각 데이터베이스는 트랜잭션 관리 프로그램(TM)의 별도의 자원 관리 프로그램(RM)으로 정의되며, 데이터베이스는 다음 구문을 갖는 xa_open 문자열로 식별되어야 합니다.

```
"database_alias<,userid,password>"
```

*database_alias*는 데이터베이스 별명을 지정하는 데 필요합니다. 별명은 사용자가 데이터베이스 작성 후 명시적으로 별명을 카탈로그화하지 않았다면 데이터베이스 이름과 같습니다. 사용자 이름과 암호는 선택적이며, 인증 방법에 따라 데이터베이스에 인증 정보를 제공하는 데 사용됩니다.

데이터베이스를 자원 관리 프로그램으로 설정할 때에는 xa_close 문자열이 필요하지 않습니다. 이 문자열이 제공되더라도 데이터베이스 관리 프로그램에 의해 무시됩니다.

호스트 또는 AS/400 데이터베이스 서버 갱신

XA 트랜잭션 관리 프로그램의 아키텍처에 따라 호스트 및 AS/400 데이터베이스 서버를 갱신할 수 있습니다. 다른 프로세스에서 확약 순서를 지원하려면, DB2 Connect concentrator가 사용 가능해야 합니다. DB2 Connect EE concentrator를 사용 가능하게 하려면, 데이터베이스 관리 프로그램 구성 매개변수 *max_logicagents*를 *maxagents*보다 더 큰 값으로 설정하십시오. DB2 Connect EE concentrator는 DB2 버전 7.1 클라이언트가 다른 프로세스로부터 XA 확약 순서를 지원하도록 요구합니다. 이 환경에서 사용할 수 있는 SQL문에 대해서는 *응용프로그램 개발 안내서*에서 자세한 내용을 참조하십시오. *DB2 Connect 사용자 안내서*에서 concentrator에 대한 자세한 내용을 참조하십시오.

호스트 또는 AS/400 데이터베이스 서버를 갱신하려면 DB2 동기 지정 관리 프로그램(SPM)이 구성된 DB2 Connect가 필요합니다. 명령어에 대해서는 *빠른 시작* 책 중 하나를 참조하십시오.

데이터베이스 연결 고려사항

이 절에서는 다음 주제를 다룹니다.

- 『RELEASE문』
- 『파티션된 데이터베이스를 액세스하는 트랜잭션』

RELEASE문

RELEASE문을 사용하여 데이터베이스로의 연결을 해제한 경우, 해당 데이터베이스로 다시 연결하려면 SET CONNECTION이 아닌 CONNECT문을 사용해야 합니다.

파티션된 데이터베이스를 액세스하는 트랜잭션

파티션된 데이터베이스 환경에서, 데이터베이스 파티션 전체에 걸쳐 사용자 데이터를 분산할 수 있습니다. 데이터베이스에 액세스하는 응용프로그램은 데이터베이스 파티션(조정자 노드) 중 하나에 연결하여 요청을 전송합니다. 서로 다른 응용프로그램은 서로 다른 데이터베이스 파티션으로 연결될 수 있으며, 동일한 응용프로그램이 서로 다른 연결에 대해 서로 다른 데이터베이스 파티션을 선택할 수 있습니다.

파티션된 데이터베이스 환경에서 데이터베이스에 대한 트랜잭션의 경우, 모든 액세스는 동일한 데이터베이스 파티션을 통해 이루어져야 합니다. 즉, 트랜잭션이 시작될 때부터 트랜잭션이 확약될 때까지(확약 당시 포함) 동일한 데이터베이스 파티션을 사용해야 합니다.

파티션된 데이터베이스에 대한 모든 트랜잭션은 연결 해제되기 전에 확약되어야 합니다.

경험적 결정

XA 준수 트랜잭션 관리 프로그램(트랜잭션 프로세싱 모니터)은 159 페이지의 『2단계 확약 프로세스 이해』에 설명되어 있는 대로 DB2 트랜잭션 관리 프로그램이 사용하는 것과 유사한 2단계 확약 프로세스를 사용합니다. 두 가지 환경간의 주요한 차이점은 TP 모니터가 DB2 트랜잭션 관리 프로그램 및 트랜잭션 관리 프로그램 데이터베이스 대신에 트랜잭션을 기록하고 제어하는 기능을 제공한다는 점입니다.

DB2 트랜잭션 관리 프로그램에 대해 논의된 것과 유사한 오류(162 페이지의 『2단계 확약 중의 문제점 복구』 참조)가 XA 준수 트랜잭션 관리 프로그램을 사용할 때 발생할 수 있습니다. DB2 트랜잭션 관리 프로그램과 마찬가지로, XA 준수 트랜잭션 관리 프로그램은 2단계 확약 중 이상 실패 트랜잭션의 재동기화를 시도합니다.

몇 가지 이유로 인해 사용자가 2단계 확약 중 이상 실패 트랜잭션을 자동으로 해결하기 위해 트랜잭션 관리 프로그램을 기다릴 수 없는 경우, 사용자가 2단계 확약 중 이상 실패 트랜잭션을 수동으로 해결하기 위해 취할 수 있는 조치가 있습니다. 이러한 수동 프로세스를 때때로 ″경험적 결정″이라고 합니다.

LIST INDOUBT TRANSACTIONS 명령(WITH PROMPTING 옵션 사용)이나 관련된 API 세트를 사용하여 2단계 확약 중 이상 실패 트랜잭션을 조회, 확약, 구간 복원할 수 있습니다. 또한, 로그 레코드를 제거하고 로그 공간을 해제시켜 경험적으로 확약 또는 구간 복원된 트랜잭션을 ″잊을(forget)″ 수 있습니다. Unix 기반 시스템, Windows 운영 체제 또는 OS/2에서 2단계 확약 중 이상 실패 트랜잭션을 획득하려면, 데이터베이스에 연결하고 LIST INDOUBT TRANSACTIONS WITH PROMPTING 명령이나 동일한 API를 발행하십시오. *Command Reference* 또는 *Administrative API Reference*에서 이 명령이나 관련된 관리 API에 대한 자세한 내용을 참조하십시오.

호스트 또는 AS/400 데이터베이스 서버에 대한 2단계 확약 중 이상 실패 트랜잭션 정보의 경우, 다음 두 가지 선택사항이 있습니다.

- 호스트 또는 AS/400 서버에서 직접 2단계 확약 중 이상 실패 정보를 획득할 수 있습니다.

  OS/390용 DB2에서 직접 2단계 확약 중 이상 실패 정보를 획득하려면, DISPLAY THREAD TYPE(INDOUBT) 명령을 호출하십시오. RECOVER 명령을 사용하여 경험적 결정을 하십시오. OS/400용 DB2에서 직접 2단계 확약 중 이상 실패 정보를 획득하려면, **wrkcmtdfn** 명령을 호출하십시오.

- 호스트 또는 AS/400 데이터베이스 서버에 액세스하는 데 사용되는 DB2 Connect 서버에서 2단계 확약 중 이상 실패 정보를 획득할 수 있습니다.

  DB2 Connect 서버에서 2단계 확약 중 이상 실패 정보를 획득하려면, *spm_name* 데이터베이스 관리 프로그램 구성 매개변수의 값에 의해 표시된 DB2 인스턴스

에 연결하여 DB2 동기 지점 관리프로그램에 우선 연결하십시오. 그런 다음 LIST DRDA INDOUBT TRANSACTIONS WITH PROMPTING 명령을 발행하여 2단계 확약 중 이상 실패 트랜잭션을 표시하고 경험적 결정을 작성하십시오.

*극도의 경고*로 이 명령을 사용하고, 마지막 수단으로서 사용하십시오. 최선의 전략은 트랜잭션 관리 프로그램이 재동기화 프로세스를 추진하기를 기다리는 것입니다. 참여 데이터베이스 중 하나의 트랜잭션을 수동으로 확약 또는 구간 복원하면 데이터 무결성 문제점이 발생할 수 있으며, 또 다른 참여 데이터베이스에 대해서는 반대되는 조치가 취해집니다. 데이터 무결성 문제점으로부터 복구하려면, 응용프로그램 논리를 이해하고, 변경되거나 구간 복원된 데이터를 식별한 다음 데이타베이스의 특정 시점 복구를 수행하거나 수동으로 데이타베이스 변경사항을 실행 취소하거나 다시 적용해야 합니다.

트랜잭션 관리 프로그램이 재동기화 프로세스를 초기화하기를 기다릴 수 없고 2단계 확약 중 이상 실패 트랜잭션에 의해 점유되고 있는 자원을 해제시켜야 하는 경우, 경험적 조작이 필수적입니다. 이러한 상황은 확장된 기간 동안 트랜잭션 관리 프로그램이 재동기화 수행에 사용될 수 없으며, 긴급하게 필요한 자원을 2단계 확약 중 이상 실패 트랜잭션이 점유하고 있는 경우에 발생할 수 있습니다. 2단계 확약 중 이상 실패 트랜잭션은 트랜잭션 관리 프로그램 또는 자원 관리 프로그램이 사용 불가능하게 되기 전에 이 트랜잭션과 연관된 자원을 점유합니다. 데이터베이스 관리 프로그램의 경우, 이러한 자원에는 테이블과 색인에 대한 잠금, 로그 공간 및 트랜잭션에 의해 점유된 저장영역 등이 포함됩니다. 또한 각 2단계 확약 중 이상 실패 트랜잭션은 데이터베이스에 의해 처리될 수 있는 동시 트랜잭션의 최대 수를 감소시킵니다(하나씩).

경험적 조작을 수행할 아주 간단한 방법은 없지만, 다음은 일반적인 지침을 제공합니다.

1. 모든 트랜잭션을 완료하기 위해 필요한 데이터베이스에 연결하십시오.
2. LIST INDOUBT TRANSACTIONS 명령을 사용하여 2단계 확약 중 이상 실패 트랜잭션을 표시하십시오. *xid*는 전역 트랜잭션 ID를 나타내며, 트랜잭션 관리 프로그램과 트랜잭션에 참여하는 다른 자원 관리 프로그램에 의해 사용되는 *xid*와 동일합니다.

3. 2단계 확약 중 이상 실패 트랜잭션 각각에 대해, 응용프로그램과 운영 환경에 관한 사용자의 지식을 사용하여 다른 참여 자원 관리 프로그램을 결정하십시오.
4. 트랜잭션 관리 프로그램이 사용 가능한지 판별하십시오.
   - 트랜잭션 관리 프로그램이 사용 가능하고 자원 관리 프로그램에 의해 그 안에 있는 2단계 확약 중 이상 실패 트랜잭션이 두 번째 확약 단계시 또는 이전 재동기 프로세스에 대해 사용할 수 없게 된다면, 트랜잭션 관리 프로그램의 로그를 점검하여 다른 자원 관리 프로그램에 대해 어떤 조치가 취해졌는지 알아내야 합니다. 그런 다음, 데이터베이스에 대해 동일한 조치를 취해야 합니다. 즉, LIST INDOUBT TRANSACTIONS 명령을 사용하여 트랜잭션을 경험적인 방법으로 확약하거나 구간 복원해야 합니다.
   - 트랜잭션 관리 프로그램이 *사용 불가능*한 경우, 다른 참여 자원 관리 프로그램 내의 트랜잭션 상태를 사용하여 사용자가 취해야 할 조치를 알아내야 할 것입니다.
     - 적어도 하나의 다른 자원 관리 프로그램이 트랜잭션을 확약한 경우, 모든 자원 관리 프로그램 내의 트랜잭션을 경험적인 방법으로 확약해야 합니다.
     - 적어도 하나의 다른 자원 관리 프로그램이 트랜잭션을 구간 복원한 경우, 트랜잭션을 경험적인 방법으로 구간 복원해야 합니다.
     - 트랜잭션이 모든 참여 자원 관리 프로그램에서 "준비된"(2단계 확약 중 이상 실패) 상태라면, 트랜잭션을 경험적인 방법으로 구간 복원해야 합니다.
     - 하나 이상의 다른 자원 관리 프로그램이 사용 불가능한 경우, 경험적인 방법으로 트랜잭션을 구간 복원해야 합니다.

경험적인 방법으로 확약되거나 구간 복원된 트랜잭션으로 인해 로그가 가득 찬 상태가 발생했음이 LIST INDOUBT TRANSACTIONS 명령의 출력에 나타난 경우를 제외하고는 경험적 잊기 기능을 수행하지 마십시오. 경험적 잊기 기능은 2단계 확약 중 이상 실패 트랜잭션이 점유하고 있는 로그 공간을 해제시킵니다. 이는 트랜잭션 관리 프로그램이 이 2단계 확약 중 이상 실패 트랜잭션에 대해 재동기화 조작을 수행하는 경우, 이 자원 관리 프로그램에 트랜잭션에 대한 로그 레코드가 없기 때문에 다른 자원 관리 프로그램의 확약 또는 구간 복원을 잘못 결정할

가능성이 있음을 의미합니다. 일반적으로 "누락" 로그 레코드는 자원 관리 프로그램이 트랜잭션을 구간 복원했음을 의미합니다.

보안 고려사항

TP 모니터는 서버 프로세스 세트를 사전 할당하고 이 서버 프로세스 ID 하에서 다른 사용자의 트랜잭션을 실행시킵니다. 데이터베이스에게는 각각의 서버 프로세스가 많은 작업 단위(UOW)를 가지고 서버 프로세스와 연관된 동일한 ID 하에서 모두가 수행되는 하나의 커다란 응용프로그램으로 나타납니다.

예를 들어, CICS를 사용하는 AIX 환경에서, TXSeries CICS 지역이 시작될 때, 이는 정의된 AIX 사용자 이름과 연관됩니다. 또한 모든 CICS 응용프로그램 서버 프로세스는 보통 "cics"로 정의되어 있는 이러한 TXSeries CICS "마스터" ID 하에서 실행되고 있습니다. CICS 사용자는 해당 DCE 로그인 ID 하에서 CICS 트랜잭션을 불러올 수 있으며, CICS에 있는 동안 CESN 사인온 트랜잭션을 사용하여 해당 ID를 변경할 수 있습니다. 어느 경우에도 일반 사용자 ID를 RM에서 사용할 수 없습니다. 결과적으로, CICS 응용프로그램 프로세스는 많은 사용자를 대신하여 트랜잭션을 수행중일 수 있으나, RM에게는 마치 동일한 "cics" ID의 여러 작업 단위(UOW)를 갖는 단일 프로그램인 것처럼 보여집니다. 선택적으로, xa_open 문자열에 사용자 ID와 암호를 지정할 수 있으며, 그 사용자 ID는 "cics" ID를 대신하여 데이터베이스로 연결하는 데 사용됩니다.

데이터베이스에 액세스하는 데 일반 사용자의 특권이 아닌 바인더 특권이 사용되므로 정적 SQL에 크게 영향을 주지 않습니다. 그러나, 이는 데이터베이스 패키지의 EXECUTE 특권이 일반 사용자 ID가 아닌 서버 ID에 권한 부여되어야 함을 의미합니다.

런타임 액세스 인증이 수행되는 동적문의 경우, 데이터베이스 오브젝트의 액세스 특권은 해당 오브젝트의 실제 사용자가 아닌 서버의 ID에 권한 부여되어야 합니다. 특정 사용자의 액세스를 제어하기 위해 데이터베이스에 의존하는 대신, TP 모니터 시스템에 따라 어떤 사용자가 어떤 프로그램을 사용하는지 판별해야 합니다. 서버 ID는 SQL 사용자가 요구하는 모든 특권을 권한 부여 받습니다.

누가 데이터베이스 테이블 또는 뷰에 액세스했는지 판별하려면, 다음 단계를 수행하십시오.

1. SYSCAT.PACKAGEDEP 카탈로그 뷰에서, 테이블 또는 뷰에 종속되는 모든 패키지 목록을 얻으십시오.
2. 설치에 사용된 이름 지정 규칙을 사용하여 이러한 패키지에 대응되는 서버 프로그램(예: CICS 프로그램)의 이름을 판별하십시오.
3. 이러한 프로그램을 호출할 수 있는 클라이언트 프로그램(예: CICS 트랜잭션 ID)을 판별한 다음, TP 모니터 로그(예: CICS 로그)를 사용하여 누가 언제 이 트랜잭션 또는 프로그램을 수행했는지를 판별하십시오.

구성 고려사항

TP 모니터 환경을 설정할 때에는 다음 구성 매개변수를 고려해야 합니다.

- *tp_mon_name*

  이 데이터베이스 관리 프로그램 구성 매개변수는 사용 중인 TP 모니터 제품의 이름을 식별합니다(예를 들어, ″CICS″ 또는 ″ENCINA″).

- *tpname*

  이 데이터베이스 관리 프로그램 구성 매개변수는 APPC 통신 프로토콜을 사용할 때 데이터베이스 서버로 할당 요청을 발행할 때 데이터베이스 클라이언트가 사용해야 하는 원격 트랜잭션 프로그램 이름을 식별합니다. 이 값은 서버에서 구성 파일에 설정되며, SNA 트랜잭션 프로그램에서 구성된 트랜잭션 프로세서(TP) 이름과 같아야 합니다. 자세한 정보는 *빠른 시작* 매뉴얼을 참조하십시오.

- *tm_database*

  DB2는 XA 환경에서 트랜잭션을 조정하지 않으므로, 이 데이터베이스 관리 프로그램 구성 매개변수는 XA 협력 트랜잭션에는 사용되지 않습니다.

- *maxappls*

  이 데이터베이스 구성 매개변수는 허용된 실행중인 응용프로그램의 최대 수를 지정합니다. 이 매개변수의 값은 2단계 확약 또는 구간 복원을 완료하는 프로세스에서 동시에 있을 수 있는 응용프로그램의 수와 연결된 응용프로그램을 합한 수와 같거나 커야 합니다. 그런 다음, 이 합계는 어느 시점에 존재해야 하는 2단계 확약 중 이상 실패 트랜잭션의 예상 수만큼 증가되어야 합니다. 2단계 확약 중 이상 실패 트랜잭션에 대해서는 162 페이지의 『2단계 확약 중의 문제점 복구』에서 자세한 내용을 참조하십시오.

TP 모니터 환경(예를 들어, TXSeries CICS)의 경우, *maxappls* 매개변수의 값을 증가시켜야 합니다. 이것은 모든 TP 모니터 프로세스를 수용할 수 있도록 돕습니다.

- *autorestart*

  이 데이터베이스 구성 매개변수는 필요할 때 RESTART DATABASE 루틴이 자동으로 호출될지 여부를 지정합니다. 기본값은 예입니다(즉, 사용 가능).

  2단계 확약 중 이상 실패 트랜잭션이 들어 있는 데이터베이스에서는 시동을 위해 데이터베이스 조작을 재시작해야 합니다. *자동 재시작*이 사용 불가능한 경우, 데이터베이스로의 최종 연결이 삭제되면 다음 연결은 실패할 것이고 명시적 RESTART DATABASE가 다시 수행되어야 합니다. 이 조건은 2단계 확약 중 이상 실패 트랜잭션이 트랜잭션 관리 프로그램의 재동기 조작에 의해 제거되거나 관리자가 시작한 경험적 조작을 통해 제거될 때까지 존재합니다. RESTART DATABASE 명령이 발행될 때, 2단계 확약 중 이상 실패 트랜잭션이 데이터베이스에 있는 경우 메시지가 리턴됩니다. 그러면, 관리자는 LIST INDOUBT TRANSACTIONS 명령 및 다른 명령행 처리기 명령을 사용하여 2단계 확약 중 이상 실패 트랜잭션에 관한 정보를 알아낼 수 있습니다.

지원된 XA 함수

DB2 Universal Database는 다음 예외 조항을 가지고 *X/Open CAE 스펙 분산 트랜잭션 프로세싱*에 정의된 XA91 스펙을 지원합니다.

- 비동기 서비스

  XA 스펙은 인터페이스가 나중에 요청의 결과를 점검할 수 있는 비동기 서비스를 사용할 수 있도록 허용합니다. 데이터베이스 관리 프로그램은 요청이 동기 모드에서 호출되도록 요구합니다.

- 정적 등록

  XA 인터페이스를 사용하면 RM을 등록하는 두 가지 방법(정적 등록 및 동적 등록)이 허용됩니다. DB2 Universal Database는 보다 고급이며 효율적인 동적 등록만을 지원합니다. 171 페이지의 『자원 관리 프로그램(RM)』에서 이 두 가지 방법에 대한 자세한 내용을 참조하십시오.

- 이주 연관

DB2 Universal Database는 제어 스레드간의 트랜잭션 이주를 지원하지 않습니다.

xa_open 및 xa_close 문자열 사용법에 대한 정보는 173 페이지의 『xa_open 및 xa_close 문자열 사용법』을 참조하십시오.

XA 전환 사용 및 배치

XA 인터페이스가 요구할 때, 데이터베이스 관리 프로그램은 xa_switch_t 유형의 *db2xa_switch* 외부 C 변수를 제공하여 TM에 XA 스위치 구조를 되돌려 줍니다. 다양한 XA 함수의 주소 외에도 다음과 같은 필드가 리턴됩니다.

필드	값
name	데이터베이스 관리 프로그램의 프로그램 이름. 예를 들어, AIX용 DB2.
flags	TMREGISTER | TMNOMIGRATE DB2 Universal Database가 동적 등록을 사용하며 TM이 이주 연관을 사용해서는 안됨을 명시합니다. 비동기 조작이 지원되지 않음을 암시합니다.
버전	0이어야 합니다.

DB2 Universal Database XA 전환 사용

XA 아키텍처의 경우, 자원 관리 프로그램(RM)은 RM의 **xa_** 루틴에 XA 트랜잭션 관리 프로그램(XA) 액세스를 제공하는 *전환*을 제공해야 합니다. RM 전환은 xa_switch_t라는 구조를 사용합니다. 전환에는 RM의 이름, RM의 XA 진입점으로의 널(NULL)이 아닌 포인터, 플래그 및 버전 번호가 들어 있습니다.

UNIX 기반 시스템 및 OS/2: 다음 두 가지 방법을 통해 DB2 UDB의 전환을 취득할 수 있습니다.

- 추가 레벨의 간접 방법을 통해 참조합니다. C 프로그램에서, 이 방법은 매크로 정의로 수행될 수 있습니다.

  ```
  #define db2xa_switch (*db2xa_switch)
  ```

  *db2xa_switch*를 사용하기에 앞서
- **db2xacic**를 호출하여

DB2 UDB는 *db2xa_switch* 구조의 주소를 리턴하는 이 API를 제공합니다. 이것은 다음과 같이 프로토타입된 함수입니다.

```
struct xa_switch_t * SQL_API_FN  db2xacic( )
```

어느 방법을 사용하든, 응용프로그램을 libdb2(UNIX 기반 시스템) 또는 db2api.lib(OS/2)와 링크해야 합니다.

Windows NT: *xa_switch* 구조 포인터인 *db2xa_switch*는 DLL 데이터와 같이 내보내집니다. 이 구조를 사용하는 Windows NT 응용프로그램이 반드시 다음 세 가지 중 하나의 방법으로 이를 참조해야 함을 암시합니다.

- 추가 레벨의 간접 방법을 통해 참조합니다. C 프로그램에서, 이 방법은 매크로 정의로 수행될 수 있습니다.

  ```
  #define db2xa_switch (*db2xa_switch)
  ```

  *db2xa_switch*를 사용하기에 앞서
- Microsoft Visual C++ 컴파일러를 사용할 경우, *db2xa_switch*를 다음과 같이 정의합니다.

  ```
  extern __declspec(dllimport) struct xa_switch_t db2xa_switch
  ```
- **db2xacic**를 호출하여

  DB2 UDB는 *db2xa_switch* 구조의 주소를 리턴하는 이 API를 제공합니다. 이것은 다음과 같이 프로토타입된 함수입니다.

  ```
  struct xa_switch_t * SQL_API_FN  db2xacic( )
  ```

어느 방법을 사용하든, 응용프로그램을 db2api.lib와 링크해야 합니다.

C 코드 예: 다음 코드는 DB2 UDB 플랫폼에서 C 프로그램을 통해 *db2xa_switch*에 액세스할 수 있는 다양한 방법을 설명합니다. 사용자 응용프로그램을 적절한 라이브러리와 반드시 연결하도록 하십시오.

```
#include <stdio.h>
#include <xa.h>
struct xa_switch_t * SQL_API_FN  db2xacic( );
#ifdef DECLSPEC_DEFN
extern __declspec(dllimport) struct xa_switch_t db2xa_switch;
```

```
   #else
   #define db2xa_switch (*db2xa_switch)
   extern struct xa_switch_t db2xa_switch;
   #endif

main( )
   {
      struct xa_switch_t *foo;
      printf ( "%s \n", db2xa_switch.name );
      foo = db2xacic();
      printf ( "%s \n", foo->name );
      return ;
   }
```

XA 인터페이스 문제점 판별

TM으로부터의 XA 요청중에 오류가 발견되면, 응용프로그램이 TM에서 오류 코드를 받을 수 없는 경우도 있습니다. 사용자 프로그램이 이상 종료되거나 TP 모니터 또는 TM으로부터 암호화된 리턴 코드를 받을 경우, 진단 레벨 3 이상이 유효할 때 XA 오류 정보를 보고하는 첫 번째 오류 서비스 로그를 검사해야 합니다. 첫 번째 오류 서비스 로그에 대해서는 *문제점 해결 안내서*에서 자세한 내용을 참조하십시오.

또한 콘솔 메세지, TM 오류 파일 또는 사용 중인 외부 트랜잭션 프로세싱 소프트웨어에 대한 제품별 정보도 검토해야 합니다.

데이터베이스 관리 프로그램은 모든 XA-specific 특정 오류를 SQLCODE -998(트랜잭션 또는 경험적 오류) 및 해당 이유 코드와 함께 첫 번째 오류 서비스 로그에 기록합니다. 다음은 일반적인 오류 몇 가지입니다.

- xa_open 문자열의 유효하지 않은 구문
- 다음 중 하나의 결과로 인해 열기 문자열에 지정된 데이터베이스로의 연결 실패
  - 데이터베이스가 카탈로그화되지 않았습니다.
  - 데이터베이스가 시작되지 않았습니다.
  - 서버 응용프로그램의 사용자 이름 또는 암호가 해당 데이터베이스로의 연결에 대한 권한을 부여받지 못했습니다.
- 통신 오류

다음 예는 누락된 xa_open 문자열로 인해 AIX에서 발생한 xa_open 오류 로그입니다.

```
Tue Apr  4 15:59:08 1995
toop pid(83378) process (xatest) XA DTP Support      sqlxa_open Probe:101
DIA4701E Database "" could not be opened for distributed transaction
processing.
String Title : XA Interface SQLCA  pid(83378)
SQLCODE = -998  REASON CODE: 4  SUBCODE: 1
Dump File : /u/toop/diagnostics/83378.dmp Data : SQLCA
```

DB2 UDB를 사용하기 위한 XA 트랜잭션 관리 프로그램 구성

다음 절에서는 특정 제품이 DB2를 자원 관리 프로그램으로 사용하도록 구성하는 방법을 설명합니다. 다음 중 원하는 방법을 사용할 수 있습니다.

- 『IBM TXSeries CICS 구성』
- 『IBM TXSeries Encina 구성』
- 193 페이지의 『BEA Tuxedo 구성』
- 196 페이지의 『Microsoft Transaction Server 구성』

IBM TXSeries CICS 구성

IBM TXSeries CICS가 DB2를 자원 관리 프로그램으로 사용하는 방법에 관한 정보는 *IBM TXSeries CICS Administration Guide*를 참조하십시오. TXSeries 문서는 http://www.transarc.com/Library/documentation/websphere/WAS-EE/en_US/html/에서 온라인으로 볼 수 있습니다.

호스트 및 AS/400 데이터베이스 서버는 CICS 협력 트랜잭션에 참여할 수 있습니다.

IBM TXSeries Encina 구성

다음은 Encina Monitor 및 DB2 Universal Database 서버의 통합 또는 DB2 Connect를 통해 액세스할 때 MVS용 DB2, OS/390용 DB2, AS/400용 DB2 또는 VSE 및 VM용 DB2에 필요한 API 및 구성 매개변수입니다. TXSeries 문서는 http://www.transarc.com/Library/documentation/websphere/WAS-EE/en_US/html/에서 온라인으로 볼 수 있습니다.

호스트 및 AS/400 데이터베이스 서버는 Encina 협력 트랜잭션에 참여할 수 있습니다.

DB2 구성

DB2를 구성하려면, 다음과 같이 하십시오.

1. 각 데이터베이스 이름이 DB2 데이터베이스 디렉토리에 정의되어 있어야 합니다. 해당 데이터베이스가 원격 데이터베이스인 경우, 노드 디렉토리 항목 또한 정의해야 합니다. 클라이언트 구성 지원 프로그램(CCA) 또는 DB2 명령행 처리기(CLP)를 사용하여 구성을 수행할 수 있습니다. 예를 들면, 다음과 같습니다.

   ```
   DB2 CATALOG DATABASE inventdb AS inventdb AT NODE host1 AUTH SERVER
   DB2 CATALOG TCPIP NODE host1 REMOTE hostname1 SERVER svcname1
   ```

2. DB2 클라이언트는 Encina를 처리 중임을 알 경우 Encina용 해당 내부 처리를 최적화합니다. *tp_mon_name* 데이터베이스 관리 프로그램 구성 매개변수를 ENCINA로 설정하여 이를 지정할 수 있습니다. 기본 활동은 특수 최적화가 없습니다. *tp_mon_name*이 설정된 경우, 응용프로그램은 작업 단위(UOW)를 수행하는 스레드가 작업을 종료되는 즉시 해당 작업을 확약하는지 확인해야 합니다. 다른 작업 단위(UOW)는 시작할 수 없습니다. 사용자 환경이 이러한 환경이 *아닐* 경우, *tp_mon_name* 값이 NONE인지(또는 CLP를 통해, 값이 NULL로 설정되었는지) 확인하십시오. 매개변수는 제어 센터나 CLP를 통해 갱신될 수 있습니다. CLP 명령은 다음과 같습니다.

   ```
   db2 update dbm cfg using tp_mon_name ENCINA
   ```

각 자원 관리 프로그램용으로 Encina 구성

각 자원 관리 프로그램(RM)용으로 Encina를 구성하려면, 자원 관리 프로그램이 응용프로그램의 트랜잭션용으로 등록되기 전에 관리자는 각 DB2 데이터베이스에 대해 열기 문자열, 닫기 문자열 및 제어 스레드 협정을 자원 관리 프로그램으로서 정의해야 합니다. Enconcole 전화면 인터페이스 또는 Encina 명령행 인터페이스를 사용하여 구성을 수행할 수 있습니다. 예를 들면, 다음과 같습니다.

```
monadmin create rm inventdb -open "db=inventdb,uid=user1,pwd=password1"
```

각 DB2 데이터베이스마다 하나의 자원 관리 프로그램 구성이 있으며, 각 자원 관리 프로그램(RM) 구성은 rm 이름(″논리 RM 이름″)을 소유해야 합니다. 상황을 단순화하려면, 이 이름을 데이터베이스 이름과 동일하게 설정해야 합니다.

xa_open 문자열은 데이터베이스로의 연결을 설정하는 데 필요한 정보를 수록합니다. 문자열 내용은 RM-specific별로 고유합니다. DB2 UDB의 xa_open 문자열은 열릴 데이터베이스의 별명을 수록하고 선택에 따라 연결과 관련된 사용자 ID 및 암호를 수록합니다. 여기에 정의된 데이터베이스 이름은 모든 데이터베이스 액세스에 필요한 일반 데이터베이스 디렉토리에도 카탈로그화되어야 한다는 점을 유의하십시오. DB2의 xa_open 문자열에 대한 정보는 173 페이지의 『데이터베이스를 자원 관리 프로그램으로 설정』을 참조하십시오.

DB2는 xa_close 문자열을 사용하지 않습니다.

제어 스레드 협정은 응용프로그램 에이전트 스레드가 한번에 하나의 트랜잭션을 처리할 수 있는지 여부를 판별합니다. DB2 UDB는 생략시 값인 TMXA_SERIALIZE_ALL_OPERATIONS를 지원하는데, 여기서 스레드는 트랜잭션이 완료된 후에만 사용할 수 있습니다.

OS/390용 DB2, MVS용 DB2, AS/400용 DB2 또는 VSE&VM용 DB2에 액세스하는 경우, DB2 동기 지점 관리 프로그램을 사용해야 합니다. 구성에 대한 지침은 *OS/2 및 Windows용 DB2 Connect Enterprise Edition 빠른 시작* 매뉴얼을 참조하십시오.

Encina 응용프로그램에서 DB2 데이터베이스 참조

Encina 응용프로그램에서 DB2 데이터베이스를 참조하려면, 다음을 수행하십시오.

1. Encina Scheduling Policy API를 사용하여 하나의 TP 모니터 응용프로그램 프로세스에서 실행할 수 있는 응용프로그램 에이전트 수를 지정하십시오. 예를 들면, 다음과 같습니다.

   ```
   rc = mon_SetSchedulingPolicy (MON_EXCLUSIVE)
   ```

   DB2(DB2 Universal Database, 호스트 또는 AS/400 데이터베이스 서버)의 경우, 기본 설정값인 MON_EXCLUSIVE를 사용해야 합니다. 그러면, 다음을 보장할 수 있습니다.

   - 트랜잭션 수행 주기 동안 응용프로그램 프로세스가 잠깁니다.
   - 응용프로그램은 단일 스레드로 작업합니다.

주: ODBC 또는 DB2 콜 레벨 인터페이스(CLI)를 사용하는 경우, 다중 스레드 지원을 작동 불가능으로 설정해야 합니다. CLI 구성 매개변수 DISABLEMULTITHREAD = 1(다중 스레드 작동 불가능)을 설정하여 이를 수행할 수 있습니다. DB2 Universal Database의 기본값은 DISABLEMULTITHREAD = 0(다중 스레드 작동 가능)입니다. *CLI Guide and Reference*에서 자세한 내용을 참조하십시오.

2. Encina RM Registration API를 사용하여 Encina가 응용프로그램 프로세스에서 RM을 참조할 때 사용할 XA 전환 및 논리 RM 이름을 제공하십시오. 예를 들면, 다음과 같습니다.

```
rc = mon_RegisterRmi ( &db2xa_switch,   /* xa switch */
                       "inventdb",      /* logical RM name */
                       &rmiId );        /* internal RM ID */
```

   XA 전환은 TM이 호출할 수 있는 RM의 XA 루틴 주소를 포함하며, RM이 제공하는 기능도 지정합니다. DB2 Universal Database의 XA 전환은 db2xa_switch이며, DB2 클라이언트 라이브러리에 상주합니다(Windows 운영 체제 및 OS/2의 db2app.dll과 Unix 기반 시스템의 libdb2).

   논리 RM 이름은 Encina가 사용하는 이름으로서, Encina 하에서 실행되는 SQL 응용프로그램이 사용하는 실제 데이터베이스 이름이 아닙니다. 실제 데이터베이스 이름은 Encina RM 등록 API의 xa_open 문자열에 지정됩니다. 논리 RM 이름이 이 예의 데이터베이스 이름과 동일한 이름으로 설정됩니다.

   세 번째 매개변수는 TM이 이 연결을 참조하기 위해 사용하는 내부 식별자 또는 핸들을 리턴합니다.

주: TM-XA 인터페이스를 통해 DB2와 함께 트랜잭션 프로세싱용 Encina-nested를 사용할 경우, 현재 DB2 XA 인터페이스는 Encina 중첩 트랜잭션을 지원하지 않는다는 점을 주의하십시오. 가능한 경우 이 트랜잭션 사용을 피하십시오. 사용이 불가피한 경우에는 반드시 SQL 작업이 Encina 트랜잭션 계열의 한 구성원에서만 수행되도록 하십시오.

BEA Tuxedo 구성

Tuxedo가 DB2를 자원 관리 프로그램으로 사용하도록 구성하려면, 다음 단계를 수행하십시오.

1. 해당 제품 관련 책에 지정된 대로 Tuxedo를 설치하십시오. 로그 파일 및 환경 변수를 포함하여 모든 기본 Tuxedo 구성을 수행하도록 하십시오.

   또한, 컴파일러 및 DB2 Software Developer's Kit(SDK)이 필요합니다. 필요하다면, 이 프로그램을 설치하십시오.

2. Tuxedo 서버 ID에서, DB2INSTANCE 환경 변수를 Tuxedo가 사용할 데이터베이스가 들어 있는 인스턴스를 참조하도록 설정하십시오. PATH 변수가 DB2 프로그램 디렉토리를 포함하도록 설정하십시오. Tuxedo 서버 ID가 DB2 데이터베이스로 연결할 수 있는지 확인하십시오.

3. *tp_mon_name* 데이터베이스 관리 프로그램 구성 매개변수를 TUXEDO 값으로 갱신하십시오.

4. DB2에 대한 정의를 Tuxedo 자원 관리 프로그램 정의 파일에 추가하십시오. 다음 예에서, UDB_XA는 DB2에 대해 지역적으로 정의된 Tuxedo 자원 관리 프로그램 이름이고 *db2xa_switch*는 xa_switch_t 유형의 구조에 대한 DB2 정의 이름입니다.

   - AIX의 경우, ${TUXDIR}/udataobj/RM 파일에 다음 정의를 추가하십시오.

     ```
     # DB2 UDB
     UDB_XA:db2xa_switch:-L${DB2DIR} /lib -ldb2
     ```

     여기서, {TUXDIR}은 Tuxedo를 설치한 디렉토리이고, {DB2DIR}은 DB2 인스턴스 디렉토리입니다.

   - Windows NT의 경우, %TUXDIR%\udataobj\rm 파일에 다음 정의를 추가하십시오.

     ```
     # DB2 UDB
     UDB_XA;db2xa_switch;%DB2DIR%\lib\db2api.lib
     ```

     여기서, %TUXDIR%은 Tuxedo를 설치한 디렉토리이고 %DB2DIR%은 DB2 인스턴스 디렉토리입니다.

5. DB2에 대한 Tuxedo 트랜잭션 모니터를 구축하십시오.

   - AIX의 경우,

     ```
     ${TUXDIR}/bin/buildtms -r UDB_XA -o ${TUXDIR}/bin/TMS_UDB
     ```

     여기서, {TUXDIR}은 Tuxedo를 설치한 디렉토리입니다.

- Windows NT의 경우,

```
%TUXDIR%\bin\buildtms -r UDB_XA -o %TUXDIR%\bin\TMS_UDB
```

6. 응용프로그램 서버를 구축하십시오. 다음 예에서, `-r` 옵션은 자원 관리 프로그램 이름을 지정하고, `-f` 옵션(한 번 이상 사용됨)은 응용프로그램 서비스가 들어 있는 파일을 지정하고, `-s` 옵션은 이 서버에 대한 응용프로그램 서비스 이름을 지정하며, `-o` 옵션은 출력 서버 파일 이름을 지정합니다.

   - AIX의 경우,

```
${TUXDIR}/bin/buildserver -r UDB_XA -f svcfile.o -s SVC1,SVC2
   -o UDBserver
```

     여기서, {TUXDIR}은 Tuxedo를 설치한 디렉토리입니다.

   - Windows NT의 경우,

```
%TUXDIR%\bin\buildserver -r UDB_XA -f svcfile.o -s SVC1,SVC2
   -o UDBserver
```

     여기서, %TUXDIR%은 Tuxedo를 설치한 디렉토리입니다.

7. Tuxedo 구성 파일이 DB2 서버를 참조하도록 설정하십시오. UDBCONFIG 파일의 *GROUPS 섹션에 다음과 유사한 항목을 추가하십시오.

```
UDB_GRP   LMID=simp GRPNO=3
  TMSNAME=TMS_UDB TMSCOUNT=2
  OPENINFO="UDB_XA:db=sample,uid=db2_user,pwd=db2_user_pwd"
```

   여기서, TMSNAME 매개변수는 이전에 구축한 트랜잭션 모니터 서버 프로그램을 지정하고, OPENINFO 매개변수는 자원 관리 프로그램 이름을 지정합니다. 그 다음에는 인증에 사용되는 데이터베이스 이름 및 DB2 사용자와 암호가 옵니다.

   이전에 구축한 응용프로그램 서버는 Tuxedo 구성 파일의 *SERVERS절에서 참조됩니다.

8. 응용프로그램이 OS/390용 DB2, OS/400용 DB2 또는 VM&VSE용 DB2에 상주하는 데이터를 액세스하는 경우, DB2 Connect XA concentrator가 필요합니다. *DB2 Connect 사용자 안내서*에서 구성 세부사항 및 제한사항에 대한 자세한 내용을 참조하십시오.
9. Tuxedo를 시작하십시오.

```
tmboot -y
```

명령이 완료되면, Tuxedo 메세지는 서버가 시작되었음을 알립니다. 그리고 DB2 명령 LIST APPLICATIONS ALL을 발행하면, Tuxedo 구성 파일 UDBCONFIG의 UDB 그룹 내 TMSCOUNT 매개변수에 의해 지정된 두 가지 연결이 표시됩니다(이 경우).

Microsoft Transaction Server 구성

DB2 UDB V5.2 이상 버전은 Microsoft Transaction Server(MTS) 버전 2.0과 완전히 통합될 수 있습니다. Windows 32비트 운영 체제의 MTS에서 실행되는 응용프로그램은 MTS를 사용하여 다른 MTS 준수 자원 관리 프로그램뿐 아니라 다중 DB2 UDB, 호스트 및 AS/400 데이터베이스 서버를 통해 2단계 확약을 조정할 수 있습니다.

주: IBM 웹 사이트에서는 DB2 MTS 지원의 설치 및 구성을 도와줄 추가 기술 정보가 제공됩니다. 사용자의 URL을 다음으로 설정하십시오. http://www.ibm.com/software/data/db2/library/ 그리고 키워드 "MTS"로 DB2 Universal Database "Technote"를 검색하십시오.

DB2에 MTS 지원 작동

Microsoft Transaction Server 지원은 자동으로 작동 가능해집니다. *tp_mon_name* 데이터베이스 관리 프로그램 구성 매개변수를 MTS로 설정할 수 있지만, 이것은 필요한 것이 아니므로 무시됩니다.

MTS 소프트웨어 전제조건

MTS를 지원하려면 DB2 CAE 버전 5.2 이상이 필요하고, MTS는 Hotfix 0772 이상 릴리스를 사용하는 버전 2.0이어야 합니다.

Windows 32비트 운영 체제에 DB2 ODBC 드라이버를 설치하면 자동으로 레지스트리에 새 키워드가 추가됩니다.

```
HKEY_LOCAL_MACHINE\software\ODBC\odbcinit.ini\IBM DB2 ODBC Driver:
Keyword Value Name: CPTimeout
Data Type: REG_SZ
Value: 60
```

설치 및 구성

다음은 MTS 설치 및 구성과 관련된 고려사항 개요입니다. DB2의 MTS 지원을 사용하려면, 다음을 수행해야 합니다.

1. MTS를 설치하고 MTS 응용프로그램이 실행되는 같은 머신에 DB2 클라이언트를 설치하십시오.
2. 호스트 또는 AS/400 데이터베이스 서버가 다중 사이트 갱신에 관련될 경우,
   a. 지역 머신 또는 원격 머신에 DB2 Connect Enterprise Edition(EE)을 설치하십시오. DB2 Connect EE를 사용하여 호스트 또는 AS/400 데이터베이스 서버가 다중 사이트 갱신 트랜잭션에 참가할 수 있습니다.
   b. DB2 Connect EE 서버는 다중 사이트 갱신에 대해 사용 가능한지 확인하십시오. 다중 사이트 갱신을 위한 DB2 Connect 작동에 관한 내용은 사용자 플랫폼에 대한 DB2 Connect Enterprise Edition 빠른 시작 매뉴얼을 참조하십시오.

DB2 CLI/ODBC 응용프로그램을 수행할 때, 다음과 같은 구성 키워드(db2cli.ini 파일에 설정된)를 기본값에서 변경하지 않아야 합니다.

- CONNECTYPE 키워드(기본값 1)
- MULTICONNECT 키워드(기본값 1)
- DISABLEMULTITHREAD 키워드(기본값 0)
- CONNECTIONPOOLING 키워드(기본값 0)
- KEEPCONNECTION 키워드(기본값 0)

MTS 지원을 사용하기 위해 작성된 DB2 CLI 응용프로그램은 위의 키워드에 해당하는 속성 값을 변경해서는 안 됩니다. 그 밖에 응용프로그램은 다음 속성의 기본값을 변경하지 않아야 합니다.

- SQL_ATTR_CONNECT_TYPE 속성(기본값 SQL_CONCURRENT_TRANS)
- SQL_ATTR_CONNECTON_POOLING 속성(기본값 SQL_CP_OFF)

설치 확인

1. DB2 UDB, 호스트 또는 AS/400 서버에 액세스하도록 DB2 클라이언트 및 DB2 Connect EE를 구성하십시오.

2. DB2 CAE 머신으로부터 DB2 UDB 데이터베이스 서버로의 연결을 확인하십시오.
3. DB2 CLP를 사용하여 DB2 Connect 머신으로부터 사용자의 호스트 또는 AS/400 데이터베이스 서버로의 연결을 확인하고 몇몇 조회를 실행하십시오.
4. DB2 Connect 게이트웨이를 통한 DB2 CAE 머신으로부터 사용자의 호스트 또는 AS/400 데이터베이스 서버로의 연결을 확인하고 몇몇 조회를 실행하십시오.

지원되는 DB2 데이터베이스 서버

다음 서버는 MTS 협력 트랜잭션을 사용한 다중 사이트 갱신을 지원합니다.

- DB2 Universal Database Enterprise Edition 버전 6 이상
- DB2 Universal Database Enterprise - Extended Edition 버전 6 이상
- OS/390용 DB2
- MVS용 DB2
- AS/400용 DB2
- VM 및 VSE용 DB2

MTS 트랜잭션 시간종료 및 DB2 연결 작동

MTS Explorer 도구에 트랜잭션 시간종료 값을 설정할 수 있습니다. 온라인 *MTS 관리자 안내서*에서 자세한 내용을 참조하십시오.

트랜잭션이 트랜잭션 시간종료 값(기본값은 60초)보다 오래 걸리는 경우, MTS는 관련된 모든 자원 관리 프로그램에 비동기식으로 중단을 발행하고 전체 트랜잭션은 중단됩니다.

DB2 서버로 연결된 경우, 중단은 DB2 구간 복원 요청으로 변환됩니다. 다른 모든 데이터베이스 요청과 마찬가지로, 연결상에서 구간 복원 요청을 직렬화하여 데이터베이스 서버에서의 데이터 무결성을 보장할 수 있습니다.

결과는 다음과 같습니다.

- 해당 연결이 대기 상태인 경우, 구간 복원이 즉시 실행됩니다.
- 오래 실행하는 SQL문을 처리 중인 경우, SQL문이 완료될 때까지 구간 복원 요청이 대기합니다.

연결 풀링

연결 풀링은 응용프로그램이 연결 풀로부터의 연결을 사용하도록 하여 각 사용마다 연결을 재설정할 필요가 없게 합니다. 일단 연결이 작성되어 풀에 배치되면, 응용프로그램은 전체 연결 프로세스를 수행하지 않고 해당 연결을 재사용할 수 있습니다. 응용프로그램이 ODBC 데이터 소스로부터 연결 해제되면 연결이 풀링되고, 해당 속성이 동일한 새 연결로 이어집니다.

연결 풀링은 ODBC 드라이버 관리 프로그램 2.x의 기능이었습니다. MTS와 함께 제공된 최신 ODBC 드라이버 관리 프로그램(버전 3.5)을 통해 연결 풀링은 구성이 약간 변경되었으며 트랜잭션용 MTS COM 오브젝트의 ODBC 연결 작동이 새로와졌습니다(200 페이지의 『동일한 트랜잭션에 참여하는 COM 오브젝트간에 ODBC 연결 재사용』 참조).

ODBC 드라이버 관리 프로그램 3.5를 사용하려면 ODBC 드라이버는 연결 풀링을 활성화하기 전에 레지스트리에 새 키워드를 등록해야 합니다. 키워드는 다음과 같습니다.

```
Key Name: SOFTWARE\ODBC\ODBCINST.INI\IBM DB2 ODBC DRIVER
Name: CPTimeout
Type: REG_SZ
Data: 60
```

32비트 Windows 운영 체제용 DB2 ODBC 드라이버 버전 6 이후 버전은 연결 풀링을 완전 지원하므로 이 키워드가 등록됩니다. 버전 5.2 클라이언트는 Fix Pack 3(WR09024) 이상을 설치해야 합니다.

기본값 60은 연결이 해제되기 전에 60초간 풀링됨을 의미합니다.

업무량이 많은 환경에서는 CPTimeout 값을 증가시켜(Microsoft는 때로 특정 확경에서 10분을 제안함) 실제 연결 및 연결 해제가 너무 많이 발생하지 않도록 하는 것이 좋습니다. 연결 및 연결 해제가 너무 많으면 시스템 메모리 및 통신 스택 자원을 포함한 시스템 자원이 너무 많이 소모되기 때문입니다.

그리고, 동일한 연결이 다중 프로세서 머신에서 동일한 트랜잭션의 오브젝트 간에 사용되는지 확인하려면, "프로세서당 다중 풀" 지원을 해제해야 합니다. 이를 수행하려면, 다음과 같은 레지스트리 설정을 `odbcpool.reg`라는 파일로 복사하고, 일

반 텍스트 파일로서 저장하고 **odbcpool.reg** 명령을 발행하십시오. Windows 운영 체제는 이 레지스트리 설정을 가져옵니다.

```
REGEDIT4
[HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBCINST.INI\ODBC Connection Pooling]
"NumberOfPools"="1"
```

이 키워드를 1로 설정하지 않고, MTS는 여러 풀에 연결을 풀링할 수 있으므로 동일한 연결을 재사용하지 않습니다.

ADO 2.1 이상을 사용하여 MTS 연결 풀링

MTS COM 오브젝트가 ADO를 사용하여 데이터베이스에 액세스하는 경우, OLEDB 자원 풀링을 해제하여 MSDASQL(Microsoft OLEDB provider for ODBC)이 ODBC 연결 풀링을 방해하지 않습니다. 이 기능은 ADO 2.0에서 OFF로 초기화되지만, ADO 2.1에서는 ON으로 초기화됩니다. OLEDB 자원 폴링을 해제하려면, 다음 행을 oledb.reg라고 하는 파일로 복사하여, 일반 텍스트 파일로서 저장하고 **oledb.reg** 명령을 발행하십시오. Windows 운영 체제는 이 레지스트리 설정을 가져옵니다.

```
REGEDIT4
[HKEY_CLASSES_ROOT\CLSID\{c8b522cb-5cf3-11ce-ade5-00aa0044773d}]
@="MSDASQL"
"OLEDB_SERVICES"=dword:fffffffc
```

동일한 트랜잭션에 참여하는 COM 오브젝트간에 ODBC 연결 재사용

MTS COM 오브젝트에 있는 ODBC 연결의 연결 풀링은 자동으로 활성화됩니다 (COM 오브젝트 처리 가능 여부와 상관 없음).

동일한 트랜잭션에 여러 MTS COM 오브젝트가 참여하는 경우, 다음과 같은 방식으로 둘 이상의 COM 오브젝트 사이에 연결을 재사용할 수 있습니다.

동일한 ODBC 데이터 소스에 연결되며 동일한 트랜잭션에 참여하는 두 COM 오브젝트인 COM1 및 COM2가 있다고 가정하면, 다음과 같습니다.

COM1이 연결되어 해당 작업을 수행한 후, 연결 해제되며 해당 연결은 풀링됩니다. 그러나, 이 연결은 동일한 트랜잭션의 다른 COM 오브젝트용으로 예약됩니다. 현재 트랜잭션이 종료된 후에만 다른 트랜잭션은 이 연결을 사용할 수 있습니다.

동일한 트랜잭션에서 COM2가 호출되면, 해당 트랜잭션에는 풀링된 연결이 주어집니다. MTS는 동일한 트랜잭션에 참여하는 COM 오브젝트에만 연결이 제공될 수 있도록 합니다.

반면, COM1이 명시적으로 연결 해제되지 않은 경우, 트랜잭션이 종료될 때까지 연결을 고정해둡니다. 동일한 트랜잭션에서 COM2가 호출되면, 별도의 연결이 주어집니다. 따라서 이 트랜잭션은 하나가 아닌 두 개의 연결을 둡니다.

다음과 같은 이유로 동일한 트랜잭션에 참여하는 COM 오브젝트의 이러한 연결 재사용 기능을 사용하는 것이 좋습니다.

- 이 기능은 클라이언트와 서버 모두에서 자원을 덜 사용합니다. 하나의 연결만이 필요합니다.
- 이 기능은 동일한 트랜잭션(동일한 데이터베이스 서버 및 동일한 데이터에 액세스하는)에 참여하는 두 연결이 서로를 잠그게 될 가능성을 배제하는데, DB2 서버는 MTS COM 오브젝트로부터의 서로 다른 연결을 별도의 트랜잭션으로 간주하기 때문입니다.

TCP/IP 통신 조정

동시에 너무 많은 실제 연결 및 연결 해제가 발생하는 워크로드가 높은 환경에서 작은 CPTimeout 값을 사용하면, TCP/IP 스택에 자원 제한조건이 발생할 수 있습니다.

이 문제를 줄이려면 TCP/IP 레지스트리 항목을 사용하십시오. 이에 대해서는 *Windows NT Resource Guide*, Volume 1에 설명되어 있습니다. 레지스트리 키 값은 HKEY_LOCAL_MACHINE —> SYSTEM —> CurrentControlSet —> Services —> TCPIP —> 매개변수에 위치합니다.

기본값 및 권장 값은 다음과 같습니다.

이름	기본값	권장 값
KeepAlive 시간	7200000(2시간)	같음
KeepAlive 간격	1000(1초)	10000(10초)
TcpKeepCnt	120(2분)	240(4분)
TcpKeepTries	20(20회 재시도)	같음
TcpMaxConnectAttempts	3	6

이름	기본값	권장 값
TcpMaxConnectRetransmission	3	6
TcpMaxDataRetransmission	5	8
TcpMaxRetransmissionAttempts	7	10
레지스트리 값이 정의되지 않은 경우, 해당 값을 작성하십시오.		

MTS "BANK" 샘플 응용프로그램을 사용하여 DB2 테스트

MTS와 함께 제공되는 "BANK" 샘플 프로그램을 사용하여 클라이언트 제품 및 MTS의 설정을 테스트할 수 있습니다.

다음 단계를 수행하십시오.

1. \Program Files\Common Files\ODBC\Data Sources\ MTSSamples.dsn 파일을 다음과 같이 변경하십시오.

   ```
   [ODBC]
   DRIVER=IBM DB2 ODBC DRIVER
   UID=your_user_id
   PWD=your_password
   DSN=your_database_alias
   Description=MTS Samples
   ```

   여기서,

   - *your_user_id* 및 *your_password*는 호스트에 연결하는 데 사용된 ID 및 암호입니다.
   - *your_database_alias*는 데이터베이스 서버에 연결하는 데 사용된 데이터베이스 별명입니다.

2. 제어 패널에서 ODBC 관리로 가서, **System DSN** 탭을 선택한 후, 데이터 소스를 추가하십시오.

   a. IBM ODBC 드라이버를 선택한 후 **완료**를 선택하십시오.

   b. 목록 별명 목록이 제공되면, 이전에 지정된 별명을 선택하십시오.

   c. **확인**을 선택하십시오.

3. 위와 같이, DB2 CLP를 사용하여 ID *your_user_id*를 사용하여 DB2 데이터베이스에 연결하십시오.

   a. db2cli.lst 파일을 바인드하십시오.

```
db2 bind @C:\sqllib\bnd\db2cli.lst blocking all grant public
```

b. 유틸리티를 바인드하십시오.

서버가 DRDA 호스트 서버인 경우, 연결 중인 호스트(OS/390, AS/400 또는 VSE&VM)에 따라 ddcsmvs.lst, ddcs400.lst 또는 ddcsvm.lst를 바인드하십시오. 예를 들면, 다음과 같습니다.

```
db2 bind @C:\sqllib\bnd\@ddcsmvs.lst blocking all grant public
```

그렇지 않으면, db2ubind.lst 파일을 바인드하십시오.

```
db2 bind @C:\sqllib\bnd\@db2ubind.lst blocking all grant public
```

c. 그런 다음, 다음과 같이 MTS 샘플 응용프로그램용 샘플 테이블 및 데이터를 작성하십시오.

```
db2 create table account (accountno int, balance int)
db2 insert into account values(1, 1)
```

4. DB2 클라이언트에서, 데이터베이스 관리 프로그램 구성 매개변수 *tp_mon_name*이 MTS로 설정되어 있는지 확인하십시오.

5. ″BANK″ 응용프로그램을 수행하십시오. **계정** 단추를 선택한 후 **Visual C++** 옵션을 선택한 다음, 요청을 제출하십시오. 다른 옵션은 SQL 서버에 고유한 SQL을 사용하며 작동하지 않을 수 있습니다.

제11장 고가용성 및 장애 복구 지원 소개

e-business 성공 여부는 트랜잭션 처리 시스템의 인터럽트 없는 가용성에 따라 달라지며, 트랜잭션 처리 시스템의 가용성은 1주일 24시간 내내(『24 x 7』) 사용할 수 있어야 하는 DB2와 같은 데이터베이스 관리 시스템에 의해 좌우됩니다.

고가용성

고가용성(HA)은 실행되어 항상 고객이 사용할 수 있는 시스템을 설명하는 데 사용되는 용어입니다. 다음 경우 고가용성을 달성할 수 있습니다.

- 최대 작동 기간 동안 성능이 저하되지 않고(또는 가용성이 손실되지 않고) 트랜잭션이 효율적으로 처리되어야 합니다. 파티션된 데이터베이스 환경에서 DB2는 파티션 내, 파티션 간 병렬 처리 모두를 이용하여 트랜잭션을 효율적으로 처리할 수 있습니다. SMP 환경에서 *파티션 내 병렬 처리*를 사용하여 복잡한 SQL 문의 여러 구성요소를 동시에 처리할 수 있습니다. 반면에, 파티션된 데이터베이스 환경에서 *파티션 간 병렬 처리*는 모든 관계 노드에서 조회를 동시에 처리하는 것을 말합니다. 각 노드는 테이블의 행 부속 집합을 처리합니다. 병렬 처리에 대한 자세한 정보는 39 페이지의 『병렬 처리의 유형』을 참조하십시오.
- 시스템은 하드웨어 또는 소프트웨어 장애가 발생하거나 피해가 발생했을 때 신속히 복구할 수 있어야 합니다. DB2는 고급 연속 체크포인트 시스템과 신속한 응급 복구가 가능한 병렬 복구 기능을 가지고 있습니다.

  신속 복구 기능은 적소에 검증된 백업 및 복구 전략을 수립하는 것에 따라 달라질 수도 있습니다. 복구 전략에 대한 자세한 정보는 *데이터 복구 및 고가용성 안내 및 참조서*를 참조하십시오.
- 엔터프라이즈 데이터베이스에 동력을 공급하는 소프트웨어는 지속적으로 실행되어 트랜잭션 처리에 사용할 수 있어야 합니다. 데이터베이스 관리 프로그램이 계속 실행되도록 하려면 데이터베이스 관리 프로그램에 장애가 발생할 경우 다른 데이터베이스 관리 프로그램이 대신 역할을 수행할 수 있는지 확인해야 합

니다. 이것을 장애 복구라고 합니다. *장애 복구 지원* 기능으로 하드웨어에 장애가 발생할 때 한 시스템에서 다른 시스템으로 워크로드를 자동으로 이동시킬 수 있게 됩니다.

로그 파일을 끊임없이 롤 포워드하는 다른 머신에 데이터베이스 사본을 보관함으로써, 장애 복구 보호를 달성할 수 있습니다. *로그 제공*은 전체 로그 파일을 아카이브 장치에서 또는 1차 데이터베이스에 대해 실행 중인 UserExit 프로그램을 통해 대기 머신에 복사하는 프로세스입니다. 이 접근 방법을 사용하면 DB2 복원 유틸리티나 분할 미러 기능을 사용하여 1차 데이터베이스가 대기 머신으로 복원됩니다. 새롭게 일시 정지된 입출력 지원을 사용하여 새 데이터베이스를 신속히 초기화할 수 있습니다(208 페이지의 『온라인 분할 미러 및 일시 정지된 입출력 지원을 통한 고가용성』 참조). 대기 머신의 2차 데이터베이스는 로그 파일을 연속적으로 롤 포워드합니다. 1차 데이터베이스에 장애가 발생하면, 나머지 로그 파일이 대기 머신으로 복사됩니다. 로그 끝까지 롤 포워드하고 중지 작업 후에 모든 클라이언트는 대기 머신의 2차 데이터베이스로 다시 연결됩니다.

시스템에 추가할 수 있는 플랫폼 특정 소프트웨어를 통해서도 장애 복구를 지원할 수 있습니다. 예를 들면, 다음과 같습니다.

- AIX에서 고가용성 클러스터 다중 처리, 향상된 확장성

  HACMP/ES에 대한 자세한 정보는 *데이터 복구 및 고가용성 안내 및 참조서* 또는 『IBM DB2 Universal Database Enterprise Edition for AIX and HACMP/ES』라는 백서를 참조하십시오. 이 백서는 『DB2 UDB 및 DB2 Connect 온라인 지원』 웹 사이트(http://www.ibm.com/software/data/pubs/papers/)에서 사용할 수 있습니다.

- Windows NT 또는 Windows 2000에서 Microsoft Cluster Server

  MSCS에 대한 자세한 정보는 *데이터 복구 및 고가용성 안내 및 참조서*를 참조하십시오.

- Solaris 운영 환경에서 Sun Cluster 또는 VERITAS Cluster Server

  Sun Cluster 2.x에 대한 정보는 *데이터 복구 및 고가용성 안내 및 참조서*를 참조하십시오. Sun Cluster 3.0에 대한 정보는 『DB2 and High Availability on Sun Cluster 3.0』이라는 백서를 참조하십시오. 이 백서는 『DB2 UDB 및 DB2 Connect 온라인 지원』 웹 사이트(http://www.ibm.com/software/data/pubs/papers/)에서 사용할 수 있습니다. VERITAS Cluster Server에 대한 정보는

『DB2 and High Availability on VERITAS Cluster Server』라는 백서를 참조하십시오. 이 백서도 『DB2 UDB 및 DB2 Connect 온라인 지원』 웹 사이트에서 사용할 수 있습니다.

- Hewlett-Packard에서 Multi-Computer/ServiceGuard

  HP MC/ServiceGuard에 대한 정보는 『IBM DB2 EE v.7.1 Implementation and Certification With Hewlett-Packard's MC/ServiceGuard High Availability Software』를 참조하십시오. 이 백서는 『DB2 UDB 및 DB2 Connect 온라인 지원』 웹 사이트(http://www.ibm.com/software/data/pubs/papers/)에서 사용할 수 있습니다.

장애 복구 전략은 대개 시스템 클러스터를 기반으로 합니다. *클러스터*는 하나의 시스템으로서 함께 작업하는 연결된 시스템 그룹입니다. 각 프로세서는 클러스터 내의 노드라고 합니다. 클러스터링을 사용하면 장애가 발생할 때 장애가 발생한 서버의 워크로드를 분담하여 서버가 상호 백업을 수행할 수 있습니다.

IP 주소 인계(또는 IP 인계)는 서버가 가동 중지될 때 특정 머신에서 다른 머신으로 서버 IP 주소를 전송하는 기능입니다. 클라이언트 응용프로그램에 두 개의 머신은 서로 다른 시간에 동일한 서버로 나타납니다.

장애 복구 소프트웨어는 시스템 간에 *하트비트(heartbeat) 모니터링* 또는 *활성유지 패킷*을 사용하여 가용성을 확인합니다. 하트비트(heartbeat) 모니터링에는 클러스터의 모든 노드 간에 계속적으로 통신을 유지보수하는 시스템 서비스가 수반됩니다. 하트비트(heartbeat)가 검출되지 않으면, 백업 시스템에 대한 장애 복구가 시작됩니다. 일반 사용자는 시스템에 장애가 발생했음을 알지 못합니다.

시장에서 가장 일반적인 두 개의 장애 복구 전략은 *유휴 대기*와 *상호 인계*이며, 이 용어들과 연관된 구성은 벤더에 따라 다른 용어와 연관될 수 있습니다.

유휴 대기

이 구성에서, 한 시스템은 DB2 인스턴스를 실행하는 데 사용되며, 두 번째 시스템은 『유휴』 또는 대기 모드로 첫 번째 시스템과 관련하여 운영체제 또는 하드웨어에 장애가 발생할 경우 인스턴스를 인계 받을 준비가 되어 있습니다. 필요할 때까지 대기 시스템이 대기 상태이므로 전체 시스템 성능에는 영향을 주지 않습니다.

상호 인계

이 구성에서 각 시스템은 다른 시스템에 대해 계획된 백업입니다. 백업 시스템은 장애 복구에 이어 추가로 작업을 수행해야 하므로 전체 시스템 성능에 영향을 줄 수 있습니다. 고유 작업 이외에 장애가 발생한 시스템이 수행하고 있던 작업도 수행해야 합니다.

장애 복구 전략을 사용하여 인스턴스, 파티션 또는 다중 논리 노드의 장애를 복구할 수 있습니다.

온라인 분할 미러 및 일시 정지된 입출력 지원을 통한 고가용성

*일시 정지된 입출력*은 온라인 분할 미러 조절을 위한 전체 구현을 제공하여 즉, 데이터베이스를 종료하지 않고 미러를 분할하여 연속적인 시스템 가용성을 제공합니다. *분할 미러*는 데이터가 들어 있는 디스크를 미러링하고 사본이 필요할 때 미러를 분할하여 작성할 수 있는 데이터베이스 『즉석』 사본입니다. *디스크 미러링*은 모든 데이터를 두 개의 별도 하드 디스크에 작성하는 프로세스입니다. 하나의 디스크가 다른 디스크의 미러입니다. 미러 분할은 미러 백업 사본을 작성하는 프로세스입니다.

DB2 백업 유틸리티를 사용하여 대형 데이터베이스를 백업하지 않는 경우, 일시 정지된 입출력과 분할 미러 기능을 사용하여 미러링된 이미지에서 백업을 작성할 수 있습니다. 이 접근 방법은 다음을 제공합니다.

- 생산 머신으로부터 백업 작업 오버헤드 제거
- 시스템을 복제하는 신속한 방법 제공
- 신속한 대기 장애 복구 구현 제공. 초기 복원 작업은 없으며, 롤 포워드 작업 속도가 너무 느리거나 오류가 발생하면 재초기화가 신속히 진행됩니다.

db2inidb 명령은 분할 미러를 사용할 수 있도록 초기화합니다.

- 복제 데이터베이스 작성 시

  예를 들면, 1차 데이터베이스 읽기 전용 복제본을 사용하여 보고서를 작성할 수 있습니다.
- 대기 데이터베이스로서
- 백업 이미지로

이 명령은 분할 미러에 대해서만 발행할 수 있으며, 분할 미러를 사용하기 전에 **db2inidb**를 실행해야 합니다(*데이터 복구 및 고가용성 안내 및 참조서* 참조).

파티션된 데이터베이스 환경에서 파티션의 분할 이미지를 사용하기 전에, 모든 파티션에서 **db2inidb** 명령을 실행해야 합니다. 모든 파티션에서 동시에 도구를 실행할 수 있습니다.

복제 데이터베이스 작성

데이터베이스 복제는 1차(live) 데이터베이스에 대한 오프라인 『백업』을 나타낼 수 있습니다. 그러나 복제된 데이터베이스는 백업할 수 없습니다. 원래 시스템에서 이 이미지를 복원하고 원래 시스템에서 작성된 로그 파일을 통해 롤 포워드하십시오.

데이터베이스를 복제하려면 다음 단계를 따르십시오.

1. 1차 데이터베이스에서 입출력을 일시 정지하십시오.

   ```
   db2 set write suspend for database
   ```

2. 적절한 운영 체제 명령을 사용하여 1차 데이터베이스에서 미러를 분할하십시오.
3. 1차 데이터베이스에서 입출력을 재개하십시오.

   ```
   db2 set write resume for database
   ```

4. 다른 머신에서 미러링된 데이터베이스에 접속하십시오.
5. 데이터베이스 인스턴스를 시작하십시오.

   ```
   db2start
   ```

6. 미러링된 데이터베이스를 1차 데이터베이스의 복제본으로 초기화하십시오.

   ```
   db2inidb database_alias as snapshot
   ```

   주: 이 명령은 분할이 발생할 때 이동 중인 트랜잭션을 구간 복원합니다.

분할 미러를 대기 데이터베이스로 사용

미러링된(대기) 데이터베이스가 로그를 통해 연속해서 롤 포워드할 때 1차 데이터베이스에 의해 작성 중인 새 로그는 1차 시스템에서 연속해서 페치됩니다. 분할 미러를 대기 데이터베이스로 사용하려면 다음 단계를 따르십시오.

1. 1차 데이터베이스에서 입출력을 일시 정지하십시오.

```
db2 set write suspend for database
```

2. 적절한 운영 체제 레벨 명령을 사용하여 1차 데이터베이스에서 미러를 분할하십시오.
3. 1차 데이터베이스에서 입출력을 재개하십시오.

   ```
   db2 set write resume for database
   ```

4. 미러링된 데이터베이스를 다른 인스턴스에 접속하십시오.
5. 미러링된 데이터베이스를 롤 포워드 보류 상태로 두십시오.

   ```
   db2inidb database_alias as standby
   ```

   DMS 테이블 공간(데이터베이스 관리 공간)만 있는 경우, 전체 데이터베이스를 백업하여 생산 데이터베이스에서 백업을 수행하는 오버헤드를 줄일 수 있습니다.
6. UserExit 프로그램을 설정하여 1차 시스템에서 로그 파일을 검색하십시오.
7. 데이터베이스를 로그 끝까지 롤 포워드하십시오.
8. 로그 파일 검색을 계속하고 1차 데이터베이스가 중지될 때까지 로그 끝까지 데이터베이스를 롤 포워드하십시오.

분할 미러를 백업 이미지로 사용

분할 미러를 『백업 이미지』로 사용하려면 다음 단계를 따르십시오.

1. 1차 데이터베이스에서 입출력을 일시 정지하십시오.

   ```
   db2 set write suspend for database
   ```

2. 적절한 운영 체제 명령을 사용하여 1차 데이터베이스에서 미러를 분할하십시오.
3. 1차 데이터베이스에서 입출력을 재개하십시오.

   ```
   db2 set write resume for database
   ```

4. 1차 시스템에서 장애가 발생하여 백업에서 복원을 해야 합니다.
5. 운영 체제 레벨 명령을 사용하여 1차 시스템에 대해 분할 데이터를 복사하십시오. 롤 포워드 복구에 1차 로그가 필요하므로 분할 로그는 복사하지 마십시오.
6. 1차 데이터베이스 인스턴스를 시작하십시오.

```
db2start
```

7. 1차 데이터베이스를 초기화하십시오.

```
db2inidb database_alias as mirror
```

8. 1차 데이터베이스를 로그 끝까지 롤 포워드하십시오.

제4부 부록 및 끝머리

부록A. 이름 지정 규칙

다음에 대한 정보가 필요한 경우, 해당 이름 지정 규칙을 설명하는 섹션으로 이동하십시오.

- 『일반 이름 지정 규칙』
- 『오브젝트 이름 지정 규칙』
- 220 페이지의 『연합 시스템에서 대소문자 구별 값이 보존되는 방법』

일반 이름 지정 규칙

별도로 지정한 경우를 제외하고 모든 이름에는 다음 문자가 포함될 수 있습니다.

- A - Z. 대부분의 이름에 사용할 때 A에서 Z까지의 문자는 소문자에서 대문자로 변환됩니다.
- 0 - 9
- @, #, $ 및 _(밑줄)

이름을 숫자 또는 밑줄 문자로 시작할 수 없습니다.

SQL 예약어를 테이블, 뷰, 컬럼, 색인 및 권한 부여 ID명으로 사용하지 마십시오. SQL 예약어 목록은 *SQL 참조서*를 참조하십시오.

운영 체제에 따라, DB2에 대한 작업을 하는 위치에 따라 별도로 작용하는 기타 특수 문자가 있습니다. 그러나 그런 특수 문자가 작용하는 동안 작용에 대한 보증은 없습니다. 데이터베이스의 오브젝트 이름을 지정할 때 이런 기타 특수 문자는 사용하지 않는 것이 좋습니다.

오브젝트 이름 지정 규칙

모든 오브젝트는 일반 이름 지정 규칙을 따릅니다. 또한 아래 표시된 제한사항이 추가로 있는 오브젝트도 있습니다.

표 24. 데이터베이스, 데이터베이스 별명 및 인스턴스 이름 지정 규칙

오브젝트	지침
• 데이터베이스 • 데이터베이스 별명 • 인스턴스	• 데이터베이스 이름은 이들이 카탈로그화되어 있는 위치 안에서 고유해야 합니다. DB2의 UNIX 기반 구현에서 이 위치는 디렉토리 경로인데 반해, Windows 구현에서는 드라이브 이름입니다. • 데이터베이스 별명 이름은 시스템 데이터베이스 디렉토리 안에서 고유해야 합니다. 새로운 데이터베이스가 생성될 때, 데이터베이스 이름에 대해 기본값으로 별명이 따르게 됩니다. 결과적으로, 해당 이름을 가진 데이터베이스가 없다고 하더라도, 데이터베이스 별명으로 존재하는 이름을 사용하여 데이터베이스를 생성할 수 없습니다. • 데이터베이스, 데이터베이스 별명 및 인스턴스 이름은 8바이트까지 쓸 수 있습니다. • Windows NT 및 Windows 2000 시스템에서 인스턴스 이름은 서비스 이름과 동일할 수 없습니다. **주:** 데이터베이스를 통신 환경에서 사용하려고 하는 경우, 잠재적인 문제점을 피하기 위해서는 데이터베이스 이름에 특수 문자 @, # 및 $를 사용하지 마십시오. 또한, 이러한 문자가 모든 키보드에서 사용 가능한 것이 아닐 수 있으므로, 다른 언어에서 해당 데이터베이스를 사용할 계획이 있다면, 이러한 문자를 사용하는 것은 피해 주십시오.

표 25. 데이터베이스 오브젝트 이름 지정 규칙

오브젝트	지침
• 별명 • 버퍼 풀 • 컬럼 • 이벤트 모니터 • 색인 • 메소드 • 노드 그룹 • 스키마 • 저장 프로시듀어 • 테이블 • 테이블 공간 • 트리거 • UDF • UDT • 뷰	다음 경우를 *제외*하고 최대 18바이트까지 가능합니다. • 테이블 이름(뷰 이름, 요약 테이블 이름, 별명 이름 및 상관 이름 포함)은 최대 128바이트까지 가능함 • 컬럼 이름(30바이트까지 가능함) • 스키마 이름(30바이트까지 가능함) • 오브젝트 이름에는 다음 사항도 포함됩니다. – 유효한 강조 문자(예: ö) – 멀티 바이트 공간을 제외한 멀티 바이트 문자(멀티 바이트 환경의 경우)

스키마 이름에 대한 추가 정보

- 스키마 이름이 18바이트를 초과하는 테이블은 복제할 수 없습니다.
- 사용자 정의 유형(UDT)에는 8바이트를 초과하는 스키마 이름이 있을 수 없습니다.
- 스키마 이름 SYSCAT, SYSFUN, SYSIBM, SYSSTAT는 예약어이므로 사용할 수 없습니다.
- 향후에 발생할 수도 있는 이주 문제점을 피하려면 SYS로 시작되는 스키마 이름을 사용하지 마십시오. 데이터베이스 관리 프로그램은 SYS로 시작하는 스키마 이름을 사용하여 트리거, 사용자 정의 유형 또는 사용자 정의 함수를 작성하지 못하게 합니다.
- 스키마 이름으로 SESSION을 사용하지 않는 것이 좋습니다. 선언된 임시 테이블은 SESSION으로 규정되어야 합니다. 그러므로 응용프로그램 논리가 지나치

게 복잡할 수 있는 경우에, 지속 테이블의 이름과 동일한 이름을 갖는 임시 테이블을 응용프로그램이 선언하게 하는 것이 가능합니다. 선언된 임시 테이블을 처리할 때를 제외하고는 스키마 SESSION의 사용을 피하십시오.

표 26. 사용자, 사용자 ID 및 그룹 이름 지정 규칙

오브젝트	지침
• 그룹 이름 • 사용자 이름 • 사용자 ID	• 그룹 이름은 8자까지 쓸 수 있습니다. • UNIX 기반 시스템에서 사용자 ID는 8자까지 쓸 수 있습니다. • Windows에서 사용자 이름은 30자까지 쓸 수 있습니다. Windows NT 및 Windows 2000에서는 현재 20자까지 가능합니다. • DCE 인증을 사용할 때 사용자 이름은 8자까지 가능합니다. • DCE 또는 클라이언트 인증을 사용하지 않는 경우, 사용자 이름과 암호를 명시적으로 지정하면 8자를 넘는 사용자 이름으로 Windows NT 및 Windows 2000에 연결하는 Windows이외의 32비트 클라이언트가 지원됩니다. • 이름 및 ID는 다음과 같을 수 없습니다. – *SQL 참조서*에 나열된 USERS, ADMINS, GUESTS, PUBLIC, LOCAL 또는 모든 SQL 예약어 – IBM, SQL 또는 SYS로 시작 – 강조 문자 포함 UNIX 기반 시스템에서는 그룹과 사용자가 같은 이름을 가질 수 있습니다. GRANT문의 경우, 그룹을 참조한 것인지, 사용자를 참조한 것인지의 여부를 지정해야 합니다. REVOKE 문에 대해서는 사용자 또는 그룹을 지정하는 것은 서로 다른 GRANTEETYPE 값을 갖는 GRANTEE에 대해 권한 카탈로그 테이블에 다중 행이 있는지 여부에 따라 달라집니다. Windows NT, 지역 그룹, 전역 그룹 및 사용자가 같은 이름을 가질 수 없습니다. OS/2에서는 그룹과 사용자가 같은 이름을 가질 수 없습니다. **주:** 1. 사용자 ID와 암호를 대소문자 구별하는 운영 체제도 있습니다. 이러한 경우가 있는지 운영 체제 문서를 점검해야 합니다. 2. CONNECT 또는 ATTACH에서 리턴된 권한 부여 ID는 8자로 절단됩니다. 생략부호(...)가 권한 부여 ID에 추가되며, SQLWARN 필드에는 절단을 표시하는 경고가 포함됩니다. 자세한 정보는 *SQL 참조서*의 CONNECT문을 참조하십시오.

암호에 대한 추가 정보

암호 유지보수 타스크를 수행할 때 필요할 수 있습니다. 그러한 타스크는 서버에서 요구되고, 많은 사용자가 서버 환경에 대해 작업할 수 없거나 쉽지가 않으므로, 이러한 타스크를 수행하면 위험한 상황이 올 수도 있습니다. DB2 UDB는 서버에서 요구하지 않아도 암호를 갱신하고 확인할 수 있는 방법을 제공합니다. 예를 들어, OS/390용 DB2 버전 5는 이러한 사용자 암호 변경 방법을 지원합니다. 오류 메시지 SQL1404N『암호가 만기됨』이 수신되면, 다음과 같이 암호를 변경하기 위해 CONNECT문을 사용하십시오.

```
CONNECT TO <database> USER <userid> USING <password>
   NEW <new_password> CONFIRM <new_password>
```

DB2 클라이언트 구성 지원 프로그램(CCA)의『암호 변경』대화 상자를 사용하여 암호를 변경할 수도 있습니다. 암호를 변경하는 이 방법에 대한 자세한 내용은 *SQL 참조서* 및 CCA 온라인 도움말에서 참조하십시오.

표 27. 연합 데이터베이스 오브젝트 이름 지정 규칙

오브젝트	지침
• 함수 맵핑 • 색인 스펙 • 별명 • 서버 • 유형 맵핑 • 사용자 맵핑 • 랩퍼	• 별명, 맵핑, 색인 스펙, 서버 및 랩퍼 이름은 128바이트를 초과할 수 없습니다. • 서버와 별명 옵션 및 옵션 설정값은 255바이트로 제한됩니다. • 연합 데이터베이스 오브젝트의 이름에는 다음 사항이 포함됩니다. – 유효한 강조 문자(예: ö) – 멀티 바이트 공간을 제외한 멀티 바이트 문자(멀티 바이트 환경의 경우)

오브젝트 이름에서 분리 식별자 사용

키워드를 사용할 수 있습니다. 만일 SQL 키워드로도 해석될 수 있는 구문에서 키워드가 사용되면, 이것은 분리 식별자로 지정되어야 합니다.

분리 식별자를 사용하여 이러한 이름 지정 규칙에 어긋나는 오브젝트를 작성할 수 있습니다. 그러나, 이후에 오브젝트를 사용하면 오류가 발생할 수 있습니다. 예를 들어, 이름에 + 또는 - 기호가 포함된 컬럼을 작성하고 나서 그 컬럼을 색인에 사

용하면, 테이블 재구성시 문제가 발생합니다. 분리 식별자에 대한 정보는 *SQL 참조서*의 "SQL 식별자" 절을 참조하십시오.

연합 시스템에서 대소문자 구별 값이 보존되는 방법

분산 요청에서, 데이터 소스에 대소문자가 구별되는 식별자 및 암호를 지정할 필요가 있습니다. 데이터 소스로 전달될 때 대소문자가 올바른지 확인하려면, 이 지침을 따르십시오.

- 식별자 및 암호를 필수 대소문자로 단어를 지정하고 큰 따옴표로 묶으십시오.
- 사용자 ID를 지정할 경우, 데이터 소스에 대해 *fold_id* 서버 옵션을 "n"(『아니오, 대소문자를 변경하지 마십시오』)으로 설정하십시오. 암호를 지정할 경우, 데이터 소스에 대해 *fold_pw* 서버 옵션을 "n"으로 설정하십시오.

  사용자 ID와 암호에 대해 다른 방법을 사용할 수 있습니다. 데이터 소스에서 사용자 ID를 소문자로 해야 할 경우, 이 사용자 ID를 어떠한 문자로든지 지정하고 *fold_id* 서버 옵션을 "l"(『이 ID를 소문자로 데이터 소스에 보냅니다』)로 설정할 수 있습니다. 데이터 소스에서 ID를 대문자로 해야 할 경우, 이 ID를 어떠한 문자로든지 지정하고 *fold_id*를 "u"(『이 ID를 대문자로 데이터 소스에 보냅니다』)로 설정할 수 있습니다. 같은 방법으로, 데이터 소스에서 암호를 소문자나 대문자로 해야 할 경우, *fold_pw* 서버 옵션을 "l" 또는 "u"로 설정하여 이 요구사항을 충족할 수 있습니다.

  서버 옵션에 대한 자세한 정보는 *관리 안내서: 구현*의 "서버 옵션을 사용한 데이터 소스 정의 지원 및 인증 처리 용이화"를 참조하십시오.
- 운영 체제의 명령 프롬트에서 대소문자를 구별하는 식별자 또는 암호를 큰 따옴표로 묶어 표시할 경우, 시스템이 큰 따옴표를 올바로 분석하는지 확인해야 합니다. 이를 수행하려면, 다음을 수행하십시오.

  – UNIX 기반 운영 체제에서, 명령문을 작은 따옴표로 묶어 표시하십시오.

  – Windows NT 운영 체제에서, 각 따옴표 앞에 백슬래시를 표시하십시오.

  예를 들어, DB2 데이터 소스에 있는 여러 분리 식별자는 대소문자를 구별합니다. NORBASE라는 데이터 소스에 상주하는 CS 뷰, `"my_schema"."wkly_sal"`을 위한 DB2에 대해 별명 NICK1을 작성하려 한다고 가정하십시오.

  UNIX 기반 시스템에 대한 명령 프롬프트에서, 다음과 같이 입력해야 합니다.

```
db2 'create nickname nick1 for norbase."my_schema"."wkly_sal"'
```

Windows NT 명령 프롬프트에서, 다음과 같이 입력해야 합니다.

```
db2 create nickname nick1 for norbase.\"my_schema\".\"wkly_sal\"
```

DB2 명령 프롬프트(대화식 모드)에서 명령문을 입력하거나 응용프로그램에서 명령문을 지정할 경우, 작은 따옴표나 백슬래시를 사용하지 않아도 됩니다. 예를 들어, Unix 기반 시스템 또는 Windows NT 중 하나의 DB2 명령 프롬프트에서 다음과 같이 입력합니다.

```
create nickname nick1 for norbase."my_schema"."wkly_sal"
```

부록B. 데이터베이스 이주 계획

이 절에서는 이주 프로세스의 개요를 제공합니다. DB2 UDB 버전 6은 버전 7로 이주될 필요가 없음을 기억하십시오. DB2 UDB 버전 5.*x* 데이터베이스를 이주하는 세부사항 정보는 사용자 운영 체제의 *빠른 시작* 매뉴얼에 있습니다.

데이터베이스를 이주할 때 다음과 같이 수행됩니다.

- 다음 데이터베이스 엔터티가 이주됩니다.
  - 데이터베이스 구성 파일
  - 데이터베이스 시스템 카탈로그 테이블
  - 데이터베이스 디렉토리
  - 데이터베이스 로그 파일 헤더
- 시스템 카탈로그 테이블은 다음과 같이 변경됩니다.
  - 새로운 컬럼이 추가됩니다.
  - 새로운 테이블이 생성됩니다.
  - 카탈로그 보기의 세트가 이주되면 SYSCAT 스키마에 새로운 카탈로그 뷰 세트가 생성됩니다.
  - 갱신 가능 카탈로그 뷰 세트가 SYSSTAT 스키마에 생성됩니다.
  - 일반적 목적의 스칼라 함수 세트가 유지되면, SYSFUN 스키마에 일반적 목적의 새로운 스칼라 함수가 생성됩니다. SYSFUN.DIFFERENCE 스칼라 함수만이 삭제되고 데이터베이스가 이주되는 동안 새로운 함수가 다시 생성됩니다.
- 데이터베이스 실행기록 파일 및 음영이 데이터베이스 디렉토리에 생성됩니다. 이 파일에는 데이터베이스가 복원되어야 할 경우 사용할 수 있는 백업 요약 정보가 들어 있으며, 데이터베이스에서 특정 조작이 수행될 때마다 갱신됩니다. 백업의 요약 정보가 또한 백업을 위해 보존되고 테이블 공간의 조작을 복구합니다.

이주시 고려사항

데이터베이스 관리 프로그램의 이전 버전으로 작성된 데이터베이스를 올바로 이주시키려면, 다음 사항을 고려해야 합니다.

- 『이주 제한사항』
- 225 페이지의 『보안 및 권한 부여』
- 225 페이지의 『저장영역에 대한 요구사항』
- 225 페이지의 『릴리스간 비 호환성』

이주 제한사항

데이터베이스를 버전 7에 이주하기 전에 주의해야 할 전제조건 또는 제한사항이 있습니다.

- 이주는 V5.x 또는 V6에서만 지원됩니다. DB2 V1.2 Parallel Edition에서는 지원되지 않습니다. DB2(데이터베이스 관리 프로그램)의 초기 버전은 V7으로 이주하려고 하기 전에 반드시 V5.x 또는 V6로 이주해야 합니다.
- V7 클라이언트에서 V7 서버의 데이터베이스로 이주 명령이 지원됩니다. 그러나, 이전 DB2 클라이언트에서 V7 서버로 이주 명령을 발행하는 것은 지원되지 않습니다.
- 플랫폼간의 이주는 지원되지 않습니다.
- 데이터베이스에 있는 사용자 오브젝트는 오브젝트 규정자로서 V7 예약 스키마 이름을 가질 수 없습니다. 예약된 스키마 이름으로는 SYSCAT, SYSSTAT 및 SYSFUN이 있습니다.
- BIGINT, REAL, DATALINK 또는 REFERENCE 이름을 사용하는 사용자 정의 구별 유형은 데이터베이스를 이주시키기 전에 재명명되어야 합니다.
- 다음 상태 중 하나에 있는 데이터베이스는 이주될 수 없습니다.
  - 백업 보류
  - 롤 포워드 보류
  - 정상 상태가 아닌 하나 이상의 테이블 공간
  - 트랜잭션 불일치

- 하위 레벨(V5.x 또는 V6) 데이터베이스 백업의 복원은 지원되지만, 하위 레벨 로그를 롤 포워드하는 것은 지원되지 않습니다.

보안 및 권한 부여

데이터베이스를 이주시키기 위해서는 SYSADM 권한이 필요합니다.

저장영역에 대한 요구사항

이주시키는 동안 구 카탈로그와 새로운 카탈로그 모두를 위한 공간이 요구됩니다. 요구된 디스크 공간의 양은 데이터베이스 오브젝트의 수 및 크기뿐만 아니라, 데이터베이스의 복잡성에 따라 다양하게 결정됩니다. 이들 오브젝트에는 모든 테이블 및 뷰가 포함됩니다. 데이터베이스 카탈로그가 현재 차지하고 있는 디스크 공간을 최소 두 번에 걸쳐 사용 가능하게 만들어야 합니다. 디스크 공간이 충분하지 않을 경우, 이주는 실패합니다.

SYSCAT 테이블 공간이 SMS 유형의 테이블 공간인 경우, 로그 파일과 연관되는 데이터베이스 구성 매개변수 갱신을 고려해야 합니다. *logfilsiz*, *logprimary* 및 *logsecond*의 값을 증가시켜 이 로그 파일 공간이 부족하지 않게 해야 합니다(이유 코드 3인 SQL1704N). 위와 같이 코드를 받게될 경우, 로그 공간 매개변수를 증가시키고, MIGRATE DATABASE 명령을 다시 발행하십시오.

릴리스간 비 호환성

데이터베이스를 이주하려고 계획할 때 제품의 두 버전 간의 비 호환성 영향을 고려하십시오.

버전 7의 개선된 기능을 이용하려면, 데이터베이스를 이주시킨 다음 데이터베이스 및 데이터베이스 관리 프로그램 구성을 조정해야 합니다. 이 조정 작업에 도움이 되기 위해서 이주 전과 후의 구성 매개변수 값을 기록하고 비교할 수 있습니다. (GET DATABASE CONFIGURATION 명령 및 GET DATABASE MANAGER CONFIGURATION 명령에 대해서는 *Command Reference*에서 자세한 내용을 참조하십시오.)

데이터베이스 이주

다음은 데이터베이스를 이주시킬 때 거쳐야 하는 단계입니다. 데이터베이스 관리 프로그램은 이주가 시작되기 전에 시동되어 있어야 합니다.

PRE-MIGRATION:

주: 사전 이주(pre-migration)는 앞의 릴리스에서 반드시 수행해야 하는 단계입니다(즉, 새로운 릴리스로 이주하거나 설치하기 전의 현재 릴리스에서).

1. 224 페이지의 『이주 제한사항』에 부속된 해결되지 않은 문제점이 있는지 확인하십시오.
2. 모든 응용프로그램 및 일반 사용자를 이주 중인 각 데이터베이스에서 연결 해제하십시오. (필요한 경우 LIST APPLICATIONS 명령 또는 FORCE APPLICATIONS 명령을 사용하십시오.)
3. 데이터베이스가 이주될 수 있는지 여부를 판별하기 위해 DB2CKMIG 사전 이주 유틸리티를 사용하십시오. (이 유틸리티 사용에 대한 자세한 내용은 사용자 플랫폼의 *빠른 시작* 책에서 참조하십시오.) Windows NT 또는 OS/2에서, 설치 중 이 도구를 실행하도록 프롬프트되지만, Unix 기반 시스템에서 이 도구는 인스턴스 이주 중 자동으로 호출됩니다.
4. 데이터베이스를 백업하십시오.

   이주는 다시 제자리로 되돌릴 수 있는 과정이 아닙니다. 버전 6에서 예약된 스키마 이름이 변경되기 전에 데이터베이스를 백업할 경우, DB2 버전 7을 사용하여 백업된 데이터베이스를 복원할 수 없습니다. 데이터베이스를 복원하려면, 데이터베이스 관리 프로그램의 이전 버전을 사용해야 합니다.

   주의! 데이터베이스 백업을 가지고 있지 않고 이주에 실패되면, DB2 UDB 버전 7 또는 데이터베이스 관리 프로그램의 이전 버전을 사용하여 데이터베이스를 복원할 방법이 전혀 없습니다.

   백업이 수행된 시점과 버전 7로의 업그레이드가 완료된 시점 간에 이루어진 모든 데이터베이스 트랜잭션은 복원할 수 없다는 것도 알아야 합니다. 즉, 버전 7에 설치와 이주가 완료된 다음에 데이터베이스가 복원될 필요가 있는 경우(버전 7 레벨로), 버전 7 설치 전에 기록된 로그는 롤 포워드 복구에서 사용될 수 없습니다.

MIGRATION:

5. 다음 중 하나를 사용하여 데이터베이스를 이주하십시오.
   - MIGRATE DATABASE 명령
   - 데이터베이스 백업 전체를 복원하는 경우 RESTORE DATABASE 명령
   - sqlemgdb - Database API 이주

   OS/2: 구성/설치/분배(CID) 아키텍처 환경에서 작동하는 DB2CIDMG 이주 유틸리티는 OS/2용 DB2에서만 사용할 수 있습니다. 이를 사용하면, 원격 무인 설치 및 LAN 기반 워크스테이션상에서의 구성이 가능해집니다. CID 이주를 사용하기 위해서는 LAN상에 NetView DM/2를 가지고 있어야 합니다.

   Unix 기반 시스템: 사용자 플랫폼의 *빠른 시작* 책은 주어진 인스턴스에서 모든 데이터베이스를 이주하려고 하지 않는 경우 무엇을 수행해야 하는지를 설명합니다.

POST-MIGRATION:

6. 선택적으로, DB2UIDDL 유틸리티를 사용하여 사용자 자신의 스케줄에서 고유 색인의 단계별 이주의 관리를 용이하게 하십시오. 버전 5에서 작성된 DB2 버전 5 데이터베이스는 이 도구가 지연된 고유성 점검을 이용하도록 요구하지 않는데, 이는 버전 5에서 작성된 모든 고유한 색인이 이미 이 시맨틱스를 가지기 때문입니다. 그러나, 이전에 버전 5로 이주된 데이터베이스의 경우, DB2UIDDL 유틸리티를 사용하여 고유 색인을 변경하려고 하는 경우를 제외하고는 이 시맨틱스는 자동이 아닙니다. 이 유틸리티는 사용자 테이블에서 고유 색인에 대해 CREATE UNIQUE INDEX문을 생성하여 파일에 씁니다. 이 파일을 DB2 CLP 명령 파일로 수행하면, 결과적으로 고유 색인이 버전 7 시맨틱스로 변환됩니다. *빠른 시작* 책 중 하나에서 이 유틸리티에 대한 자세한 내용을 참조하십시오.
7. 선택적으로, SQL 조회 성능에 특히 중요한 테이블에 대해 RUNSTATS 명령을 발행하십시오. 이전 통계는 이주된 데이터베이스에 보유되며, RUNSTATS 명령을 호출할 때를 제외하고는 갱신되지 않습니다.
8. 선택적으로, 모든 패키지를 재검증하거나, 패키지를 맨 처음 사용할 때, 패키지 재검증 작업이 내재적으로 일어나게 하기 위해 DB2RBIND 유틸리티를 사용할 수 있습니다.

9. 선택적으로, 버전 7에서 Explain 테이블을 사용하려고 계획하는 경우 Explain 테이블을 이주하십시오. *관리 안내서: 성능*의 "SQL Explain 기능"에서 자세한 내용을 참조하십시오.
10. 버전 7의 향상된 부분의 이점을 취하려면 데이터베이스 및 데이터베이스 관리 프로그램 구성 매개변수를 조정하십시오.

부록C. 릴리스간 비 호환성

이 절에서는 DB2 Universal Database와 DB2 이전 릴리스 사이에 존재하는 비 호환성을 설명합니다.

*비 호환성*은 이전 DB2 릴리스에서와는 다르게 작업하는 DB2 Universal Database의 일부입니다. 기존 응용프로그램에서 사용되는 경우, 예상치 못한 결과를 생성하거나 응용프로그램에 대한 변경을 필요로 하거나 성능을 저하시킵니다. 이 문맥에서, "응용프로그램"은 다음을 나타냅니다.

- 응용프로그램 코드
- 써드 파티 유틸리티
- 상호작용 SQL 조회
- 명령 또는 API 호출

DB2 Universal Database 버전 6 및 버전 7에서 소개된 비 호환성이 설명됩니다. 다음과 같은 범주에 따라 비 호환성이 그룹화됩니다.

- 시스템 카탈로그 뷰
- 응용프로그램 프로그래밍
- SQL
- 데이터베이스 보안 및 조정
- 유틸리티 및 도구
- 연결성 및 공존
- 구성 매개변수

각 비 호환성 섹션은 비 호환성에 대한 설명, 비 호환성의 증상 또는 결과, 가능한 해결책을 포함합니다. 비 호환성이 적용되는 운영 체제를 식별하는 각 비 호환성 설명의 시작부분에는 표시기가 있습니다.

WIN DB2로 지원되는 Microsoft Windows 플랫폼

UNIX DB2로 지원되는 UNIX 기반 플랫폼

OS/2 OS/2

주: DB2 Parallel Edition 버전 1.2와 함께 제공되는 클라이언트를 포함하여, DB2 Universal Database 버전 6, 버전 1.x 및 버전 2.x 클라이언트에서는 더 이상 지원되지 않습니다.

DB2 Universal Database 계획된 비 호환성

이 절에서는 DB2 Universal Database의 사용자가 새로운 새로운 응용프로그램을 코드화하거나 기존 응용프로그램을 수정할 때 기억해야 하는 장래 비 호환성을 설명합니다. 이것은 DB2 UDB의 장래 버전으로의 이주를 용이하게 합니다.

DB2 Universal Database의 향후 버전에 있는 읽기 전용 뷰

WIN	UNIX	OS/2

변경

시스템 카탈로그 뷰는 읽기 전용 뷰입니다. SYSSTAT 뷰는 계속 갱신 가능합니다.

증상

SYSCAT 뷰의 컬럼에 대해 작업하는 데 사용된 UPDATE문이 실패합니다.

설명

도구 및 응용프로그램은 SYSCAT 뷰에 정의된 대로 컬럼을 갱신하여 카탈로그에 있는 값을 변경하도록 코드화되어 있습니다.

해결

SYSSTAT 뷰에 정의된 대로 컬럼을 갱신하여 카탈로그를 변경하도록 도구 또는 응용프로그램을 변경하십시오.

DB2 Universal Database의 향후 버전에서 PK_COLNAMES 및 FK_COLNAMES

WIN	UNIX	OS/2

변경

SYSCAT.REFERENCES 컬럼 PK_COLNAMES 및 FK_COLNAMES는 더 이상 사용할 수 없습니다.

증상

컬럼이 없으므로 오류가 리턴됩니다.

설명

도구 또는 응용프로그램이 사용하지 않는 PK_COLNAMES 및 FK_COLNAMES 컬럼을 사용하도록 코드화되어 있습니다.

해결

도구 또는 응용프로그램이 SYSCAT.KEYCOLUSE 뷰를 대신 사용하도록 변경하십시오.

DB2 Universal Database의 향후 버전에서 더 이상 사용할 수 없는 COLNAMES

WIN	UNIX	OS/2

변경

SYSCAT.INDEXES 컬럼 COLNAMES는 더 이상 사용 가능하지 않습니다.

증상

컬럼이 없으므로 오류가 리턴됩니다.

설명

도구 또는 응용프로그램이 사용하지 않는 COLNAMES 컬럼을 사용하도록 코드화되어 있습니다.

해결

도구 또는 응용프로그램이 SYSCAT.INDEXCOLUSE 뷰를 대신 사용하도록 변경하십시오.

DB2 Universal Database 버전 7 비 호환성

이 절은 DB2 Universal Database 버전 7에 소개된 비 호환성을 식별합니다.

응용프로그램 프로그래밍

Query Patroller Universal Client

WIN	UNIX	OS/2

변경: CAE(Client Application Enabler)의 이 새 버전은 Query Patroller 서버 버전 7에서만 작업하는데, 그 이유는 새 저장 프로시듀어가 있기 때문입니다. CAE는 모든 응용프로그램이 결국 데이터베이스에 액세스하기 위해 통과해야 하는 DB2에 대한 응용프로그램 인터페이스입니다.

증상: 이 CAE가 백 레벨 서버에 대해 수행되면, 메시지 SQL29001이 리턴됩니다.

오브젝트 변환 함수 및 구조화 유형

WIN	UNIX	OS/2

변경: SQLDA에 작성한 변경사항과 관련된 버전 7 서버와 버전 7 이전 클라이언트 간에 드물게 나타나는 비 호환성이 있습니다. *응용프로그램 개발 안내서*에서 설명한 것처럼, 두 번째 SQLVAR의 바이트 8은 이제 값 X'00' 및 X'01' 이외에 값 X'12'에서도 얻을 수 있습니다. 새 값을 기대하지 않는 응용프로그램은 이 확장으로 영향 받을 수 있습니다.

해결: 장래 릴리스에서 이 필드에 대한 다른 확장이 있을 수 있으므로, 개발 프로그램은 명시적으로 정의된 값에 대해서만 테스트하도록 권장됩니다.

JVM이 사용하는 클래스 및 Jar 파일의 버전

WIN	UNIX	OS/2

변경: 이전에, Java 저장 프로시듀어 또는 사용자 정의 함수(UDF)가 시작된 경우, JVM(Java Virtual Machine)이 CLASSPATH에서 제공하는 모든 파일

(sqllib/function에 있는 파일 포함)을 잠궜습니다. 데이터베이스 관리 프로그램이 중지될 때까지 JVM이 이 파일들을 사용했습니다. 저장 프로시듀어 또는 UDF를 수행하는 환경에 따라(즉, *keepdari* 데이터베이스 관리 프로그램 구성 매개변수의 값에 의해 좌우되며, 저장 프로시듀어가 분리되어 있는지 여부에 따라), 클래스를 새로 고치면 데이터베이스 관리 프로그램을 중지시키지 않고 클래스 및 jar 파일을 바꾸게 합니다. 이것은 이전 작동과 다릅니다.

Jar 명령의 설치, 바꾸기 및 제거의 변경된 기능

WIN	UNIX	OS/2

변경: 이전에, jar의 설치는 모든 DARI(Database Application Remote Interface) 프로세스를 삭제했습니다. 이러한 방식으로, 새 저장 프로시듀어 클래스는 다음 호출에서 채택되도록 보장되었습니다. 현재, 어떠한 jar 명령도 DARI 프로세스를 삭제하지 않습니다. 새로 설치되거나 바뀐 jar의 클래스가 채택되는지 확인하려면, 명시적으로 SQLEJ.REFRESH_CLASSES 명령을 발행해야 합니다.

DARI 프로세스를 삭제하지 않고 도입된 또다른 비 호환성은 *keepdari* 데이터베이스 관리 프로그램 구성 매개변수의 값을 "YES"로 설정한 분리 저장 프로시듀어의 경우, 클라이언트가 jar 파일의 다른 버전을 가져올 수 있음을 의미합니다. 다음 시나리오를 고려하십시오.

1. 사용자 A는 jar를 바꾸며 클래스를 새로 고치지 않습니다.
2. 그런 다음 사용자 A는 jar에서 저장 프로시듀어를 호출합니다. 이 호출이 동일한 DARI 프로세스를 사용한다고 가정하면, 사용자 A는 jar 파일의 이전 버전을 가져옵니다.
3. 사용자 B는 동일한 저장 프로시듀어를 호출합니다. 이 호출은 새로 작성한 클래스 로드 프로그램이 jar 파일의 새 버전을 채택할 것임을 의미하는 새 DARI를 사용합니다.

다시 말해서, 클래스가 jar 조작 이후에 새로 고쳐지지 않으면, jar의 다른 버전으로부터의 저장 프로시듀어는 어떤 DARI 프로세스가 사용되는가에 따라 호출될 수 있습니다. 이것은 (DARI 프로세스를 삭제하여) 새 클래스가 항상 사용되도록 보장한 이전 작동과는 다릅니다.

32비트 응용프로그램 비 호환성

	UNIX	

변경: 32비트 실행 파일(DB2 응용프로그램)은 새 64비트 데이터베이스 엔진에 대해서는 수행되지 않습니다.

증상: 응용프로그램이 링크하는 데 실패합니다. 64비트 DB2 응용프로그램 라이브러리에 대해 32비트 오브젝트를 링크하도록 시도할 때, 운영 체제 링크 프로그램 오류 메시지가 표시됩니다.

해결: 응용프로그램은 64비트 실행파일로서 다시 컴파일되어야 하며, 새 64비트 DB2 라이브러리에 대해 다시 링크되어야 합니다.

스크래치 패드의 필드 길이 변경

WIN	UNIX	OS/2

변경: UDF로 전달되는 스크래치 패드의 길이 필드를 변경하는 임의의 사용자 정의 함수(UDF)는 이제 SQLCODE -450를 받습니다.

증상: 스크래치 패드의 길이 필드를 변경하는 UDF가 실패합니다. 호출하는 명령문은 스키마와 채워진 함수의 특정 이름과 함께 SQLCODE -450을 받습니다.

해결: 스크래치 패드의 길이 필드를 변경하지 않도록 UDF 본문을 다시 쓰십시오.

SQL

스키마 SESSION에 의해 규정된 일반 테이블을 사용하는 응용프로그램

WIN	UNIX	OS/2

변경: 스키마 SESSION은 임시 테이블용으로 허용된 유일한 스키마이며, SESSION 규정 테이블이 임시 테이블을 참조할 수 있음을 나타내기 위해 이제는 DB2에 의해 사용됩니다. 그러나, SESSION은 임시 테이블용으로 예약된 키워드가 아니며, 일반 기본 테이블용 스키마로서 사용될 수 있습니다. 그러므로, 응용프

로그램은 SESSION.T1 real 테이블 및 SESSION.T1이 기존의 임시 테이블을 동시에 선언했음을 알 수 있습니다. 패키지가 바인드될 때, "SESSION"으로 규정된 (명시적 또는 암시적으로) 테이블 참조를 포함하는 정적 명령문이 있으면, 이 명령문의 섹션이나 패키지 중 어느 것도 카탈로그에 저장되지 않습니다. 대신, 이 섹션은 런타임에 점진적으로 바인드될 필요가 있습니다. 이것은 캐쉬된 동적 SQL 캐쉬에 섹션 사본을 위치시키며, 여기서 캐쉬된 사본은 응용프로그램의 고유한 인스턴스에만 개별적입니다. 런타임에, 테이블 이름과 일치하는 선언된 임시 테이블이 존재하며, 동일한 이름의 영구 기본 테이블이 존재하더라도 선언된 임시 테이블이 사용됩니다.

증상: 버전 6 이전에서, SESSION에 의해 규정된 테이블과 관련된 정적 명령문이 있는 패키지는 항상 영구 기본 테이블을 참조합니다. 해당 명령문의 관련 종속성 레코드와 마찬가지로 패키지, 섹션을 바인드할 때, 카탈로그에 저장됩니다. 버전 7에서, 이 명령문은 바인드 시간에 바인드되지 않으며, 런타임에 동일한 이름의 선언된 임시 테이블로 분석됩니다. 그러므로, 다음 상황이 일어날 수 있습니다.

- 버전 5에서 이주. 버전 5에 그러한 패키지가 존재하는 경우, 다시 버전 6에서 바인드되며, 정적 명령문은 이제 점진적으로 바인드됩니다. 캐쉬된 동적 섹션이 다른 응용프로그램 간에 공유될 수 있는 것만 제외하고(동일한 응용프로그램 실행 파일의 여러 가지 인스턴스이더라도) 캐쉬된 동적 SQL과 같이 점진적으로 바인드된 섹션이 작동하므로 이것은 성능에 영향을 미칠 수 있습니다.
- 버전 6에서 버전 7로 이주. 버전 6에 그러한 패키지가 존재하는 경우, 반드시 다시 버전 7에서 바인드되지는 않습니다. 대신, 명령문은 원래의 바인드 시간에 카탈로그에 저장된 섹션을 사용하여, 일반 정적 SQL로서 여전히 실행합니다. 그러나, 이 패키지가 리바인드되는 경우(암시적이거나 명시적으로), SESSION 규정 테이블 참조가 있는 패키지에 있는 명령문은 더 이상 저장되지 않으며 점증적인 바인딩을 요구하지 않습니다. 이것은 성능을 저하시킬 수 있었습니다.

요약하기 위해, SESSION 규정 테이블을 참조하는 정적 명령문으로 버전 7에서 바인드하는 모든 패키지는 더 이상 정적 SQL과 같이 수행하지 않는데, 그 이유는 패키지에서 점증적인 바인딩을 요구하기 때문입니다. 사실 응용프로그램 프로세스가 기존의 SESSION 규정 테이블, 뷰 또는 별명과 동일한 이름을 가진 테이블에 대한 DECLARE GLOBAL TEMPORARY TABLE문을 발행한 경우, 선언된 임시 테이블을 참조하도록 이 오브젝트들에 대한 참조가 항상 이루어집니다.

해결: 가능하면, 영구 테이블의 스키마 이름을 변경하여 "SESSION"이 아니도록 하십시오. 그렇지 않으면, 의지가 되는 것은 없지만 발생할 수 있는 선언된 임시 테이블과의 가능한 충돌과 성능 암시를 인지하십시오.

다음 조회는 응용프로그램이 임시 테이블을 사용하는 경우 영향을 받을 수 있는 테이블, 뷰 및 별명을 식별하는 데 사용될 수 있습니다.

```
select tabschema, tabname from SYSCAT.TABLES where tabschema = 'SESSION'
```

다음 조회는 카탈로그에 저장된 정적 섹션이 있는 버전 7 바운드 패키지를 식별하기 위해 사용될 수 있으며, 패키지가 리바인드되는 경우 작동이 변경될 수 있습니다(버전 6에서 버전 7로 이동할 때에만 관련됨).

```
select pkgschema, pkgname, bschema, bname from syscat.packagedep
   where bschema = 'SESSION' and btype in ('T', 'V', 'I')
```

유틸리티 및 도구

Solaris의 데이터 링크 파일 관리 프로그램 및 파일 시스템 필터

	UNIX	

변경: 데이터 링크 파일 관리 프로그램 및 파일 시스템 필터는 Solaris OS 2.5.1에서 지원되지 않습니다.

AIX 및 Solaris의 db2set

	UNIX	

변경: 명령 "db2set -ul (user level)" 및 관련된 함수가 AIX 또는 Solaris로 이식되지 않습니다.

데이터 링크 파일 시스템 및 Norton 유틸리티**

WIN		

변경: Windows NT 데이터 링크 파일 시스템은 Norton 유틸리티와 호환되지 않습니다.

증상: DLFS에 의해 제어되는 드라이브에서 파일을 삭제하면 커널 예외가 발생합니다. 오류 0x1E(처리할 수 없는 커널 모드 예외), 0xC00000005 예외(액세스 위반)

설명: DLFS 필터 드라이버를 로드한 후에 Norton 유틸리티 드라이버를 로드했기 때문에 이 액세스 위반이 발생합니다.

해결: Norton 유틸리티 드라이버를 로드한 후에 DLFSD 드라이버를 로드하여 이 문제를 해결할 수 있습니다. 시작을 눌러 DLFSD 드라이버 시작을 수동으로 변경한 후 설정 —> 제어판 —> 장치 —> DLFSD를 선택하고 수동으로 설정하십시오.

시스템 시작 시 DLFSD 드라이버와 DLFM 서비스를 로드하는 배치 파일을 작성할 수 있습니다. 배치 파일의 내용은 다음과 같습니다.

```
net start dlfsd
net start "dlfm service"
```

이 배치 파일 이름을 start_dlfs.bat로 지정하고 **WINNT\Profiles\Administrator\Start Menu\Programs\Startup** 디렉토리에 복사하십시오. 관리자만 DLFS 필터 드라이버와 DLFM 서비스를 로드할 수 있는 권한을 갖습니다.

연결성 및 공존

32비트 클라이언트 비 호환성

WIN	UNIX	OS/2

변경: 32비트 클라이언트는 인스턴스에 첨부할 수 없으며 64비트 서버에서 데이터베이스에 연결할 수 없습니다.

증상: 클라이언트 및 서버 둘다 버전 7 코드를 수행하는 경우, SQL1434N이 리턴되며, 그렇지 않으면 첨부 또는 연결이 SQLCODE -30081로 실패합니다.

해결: 64비트 클라이언트를 사용하십시오.

DB2 Universal Database 버전 6 비 호환성

이 절은 DB2 Universal Database 버전 6에 소개된 비 호환성을 식별합니다.

시스템 카탈로그 뷰

DB2 Universal Database 버전 6에서 시스템 카탈로그 뷰

WIN	UNIX	OS/2

변경: 시스템 카탈로그 뷰에서 새로운 코드 즉, 입력된 테이블에 대한 ″U″와 입력된 뷰에 대한 ″W″가 소개되었습니다.

증상: 테이블에 대한 ″T″ 및 뷰에 대한 ″V″ 유형 코드를 사용하여 시스템 카탈로그의 테이블 및 뷰에 대해 검색하는 조회는 더 이상 입력된 테이블 및 뷰를 찾지 않습니다.

설명: TABLES, PACKAGEDEP, TRIGDEP 및 VIEWDEP 이름을 갖는 시스템 카탈로그 뷰를 포함하는 여러 시스템 카탈로그는 한 문자 유형 코드를 갖는 TYPE 또는 BTYPE 이름의 컬럼을 가집니다. 버전 5.2에서 유형 코드 ″T″가 모든 테이블에 대해 사용되고, ″V″가 모든 뷰에 대해 사용되었습니다. 버전 6의 경우, 입력되지 않은 테이블은 유형 코드 ″T″를 계속가지며, 입력된 테이블은 새로운 유형 코드 ″U″를 가지게 됩니다. 마찬가지로, 입력되지 않은 뷰는 계속 ″V″ 유형 코드를 가지고, 입력된 뷰는 새로운 유형 코드 ″W″를 가지게 됩니다. 또한, 사용자가 직접 작성하지 않았지만 테이블 계층구조를 구현하기 위해 시스템이 사용하는, 계층구조 테이블이라고 하는 새로운 종류의 테이블이 ″H″ 유형 코드가 있는 시스템 카탈로그 테이블에 나타납니다.

해결: 입력된 테이블 및 뷰에 대한 코드를 인식하도록 도구 또는 응용프로그램을 변경하십시오. 도구 또는 응용프로그램이 테이블의 논리 뷰를 필요로 할 경우, 유형 코드 ″T″, ″U″, ″V″ 및 ″W″를 사용해야 합니다. 도구 또는 응용프로그램이 계층구조 테이블과 테이블의 물리 뷰를 필요로 할 경우에는 유형 코드 ″T″ 및 ″H″를 사용해야 합니다.

DB2 Universal Database 버전 6의 1차 및 외부 키 컬럼 이름

WIN	UNIX	OS/2

변경: 두 개의 SYSCAT.REFERENCES 컬럼 PK_COLNAMES 및 FK_COLNAMES의 데이터 유형을 VARCHAR(320)에서 VARCHAR(640)로 변경합니다.

증상: 기본 키 또는 외부 키 컬럼 이름이 절단되거나, 정확하지 않거나, 생략되었습니다.

설명: 길이가 18바이트를 초과하는 컬럼 이름이 기본 키 또는 외부 키에 사용될 때, 컬럼 이름 목록이 이 두 컬럼에 저장된 형식은 그대로 보존될 수 없습니다. 길이(*n*)가 18바이트를 초과하는 컬럼 뒤에 오는 20바이트 공백 구분 컬럼 이름은 오른쪽으로 *n*-18바이트만큼 이동됩니다. 마찬가지로, 컬럼 이름 목록이 640바이트를 초과하면 컬럼은 빈 문자열을 포함하게 됩니다.

해결: SYSCAT.KEYCOLUSE 뷰에는 고유 키뿐만 아니라 기본 키, 외부 키를 작성하며, SYSCAT.REFERENCES의 컬럼 대신 사용되어야 하는 컬럼의 목록이 들어 있습니다. 또는 사용자는 컬럼 이름의 길이를 18바이트로 제한하거나 컬럼 목록의 전체 길이를 640바이트로 제한할 수 있습니다.

DB2 Universal Database 버전 6의 SYSCAT.VIEWS 컬럼 텍스트

WIN	UNIX	OS/2

변경: SYSCAT.VIEWS 컬럼 TEXT의 뷰 텍스트는 이제 여러 행으로 파티션되지 않습니다. 데이터 유형은 VARCHAR(3600)에서 CLOB(64K)로 변경됩니다.

증상: 완전 뷰 텍스트는 도구나 응용프로그램에서 제공되지 않습니다.

설명: TEXT 컬럼에서 한번에 3600(또는 아마도 3900)바이트 이하가 리턴될 것을 예상하고 코드화된 도구 또는 응용프로그램은 이 필드의 증가된 크기를 처리하지 않습니다. 다중 행을 검색하고 SEQNO 필드를 사용하여 뷰 텍스트를 재구성하는 매커니즘은 더 이상 사용되지 않습니다. SEQNO 값은 항상 1입니다.

해결: 도구 또는 응용프로그램이 3600바이트를 넘는 TEXT 컬럼의 값을 처리할 수 있도록 변경합니다. 또는, 뷰 TEXT가 3600바이트 내에 맞춰지도록 재작성될 수 있습니다.

DB2 Universal Database 버전 6의 SYSCAT.STATEMENTS 컬럼 TEXT

WIN	UNIX	OS/2

변경: SYSCAT.STATEMENTS 컬럼 TEXT의 명령문 텍스트는 이제 여러 행으로 파티션되지 않습니다. 데이터 유형은 VARCHAR(3600)에서 CLOB(64K)로 변경됩니다.

증상: 완전 명령문 텍스트는 도구 또는 응용프로그램에서 제공되지 않습니다.

설명: TEXT 컬럼에서 한번에 3600(또는 아마도 3900)바이트 이하가 리턴될 것을 예상하고 코드화된 도구 또는 응용프로그램은 이 필드의 증가된 *크기*를 처리하지 않습니다. 다중 행을 검색하고 SEQNO 필드를 사용하여 명령문 텍스트를 재구성하는 매커니즘은 더 이상 사용되지 않습니다. SEQNO 값은 항상 1입니다.

해결: 도구 또는 응용프로그램이 3600바이트를 넘는 TEXT 컬럼의 값을 처리할 수 있도록 변경합니다. 또는 명령문 TEXT가 3600바이트 내에 맞춰지도록 재작성될 수 있습니다.

DB2 Universal Database 버전 6의 SYSCAT.INDEXES 컬럼 COLNAMES

WIN	UNIX	OS/2

변경: SYSCAT.INDEXES 컬럼 COLNAMES 데이터 유형은 VARCHAR(320)에서 VARCHAR(640)로 변경됩니다.

증상: 컬럼 이름이 색인에서 빠져 있습니다.

설명: VARCHAR(320) 데이터 유형의 컬럼에서 검색하도록 코드화되어 있는 도구나 응용프로그램은 이 필드의 증가된 *크기*를 처리할 수 없습니다.

해결: SYSCAT.INDEXCOLUSE에는 색인을 구성하는 컬럼 목록이 있으며 COLNAMES 컬럼 대신 사용되어야 합니다. 또는 색인에서 컬럼을 삭제하거나 컬럼 이름의 크기를 줄여 컬럼 이름 목록(선행 + 또는 -가 있는)이 320바이트 이내에 맞춰지게 하십시오.

DB2 Universal Database 버전 6의 SYSCAT.CHECKS 컬럼 TEXT

WIN	UNIX	OS/2

변경: CHECKS 컬럼 TEXT 데이터 유형은 CLOB(32K)에서 CLOB(64K)로 변경됩니다.

증상: 점검 제한조건 절이 완전하지 않습니다.

설명: CLOB(32K) 데이터 유형의 컬럼에서 검색하도록 코드화되어 있는 도구나 응용프로그램은 이 필드의 증가된 크기를 처리할 수 없습니다.

해결: 도구 또는 응용프로그램이 32KB를 넘는 TEXT 컬럼의 값을 처리할 수 있도록 변경합니다. 또는 32바이트에 맞도록 더 적은 문자를 사용하여 점검 제한조건 절을 다시 작성하십시오.

DB2 Universal Database 버전 6에서 BIGINT에 대한 컬럼 데이터 유형

WIN	UNIX	OS/2

변경: 여러 시스템 카탈로그 뷰 컬럼이 INTEGER에서 BIGINT로 변경된 데이터 유형을 가지고 있습니다.

증상: 값은 예상된 것보다, 특히 통계 정보보다 훨씬 더 작습니다. (또는 큽니다.)

설명: INTEGER 데이터 유형의 컬럼에서 검색하도록 코드화되어 있는 도구나 응용프로그램은 이 필드의 증가된 크기를 처리할 수 없습니다.

해결: 도구 또는 응용프로그램이 INTEGER 필드에 저장될 수 있는 최대값보다 더 크거나 또는 최소값보다 더 작은 값을 처리할 수 있도록 변경하십시오. 또는 값을 INTEGER 필드에 표시될 수 있는 값을 벗어나게 하는 기본 구조 또는 SQL 코드를 변경하십시오.

DB2 Universal Database 버전 6의 컬럼 불일치

WIN	UNIX	OS/2

변경: 새로운 컬럼은 SYSCAT 뷰 정의에 있는 뷰의 끝에 삽입되지 않습니다.

증상: 여러 개의 컬럼 불일치 또는 컬럼 데이터 유형 불일치로 재 사전처리가 실패했습니다.

설명: 새로운 컬럼은 시스템 카탈로그 뷰에 소개되어 특별한 조회 환경에서 유용한 위치에 배치되는데, 특히 짧은 컬럼은 매우 긴 컬럼 앞에 배치되고 REMARKS 컬럼은 항상 마지막 컬럼이 됩니다.

해결: ″SELECT *″를 코드화하는 대신 선택 목록에 컬럼 이름을 명시하십시오.

DB2 Universal Database 버전 6의 SYSCAT.COLUMNS 및 SYSCAT.ATTRIBUTES

WIN	UNIX	OS/2

변경: SYSCAT.COLUMNS 및 SYSCAT.ATTRIBUTES는 이제 계승된 컬럼과 속성에 대한 항목을 포함합니다.

증상: 입력된 테이블 또는 뷰의 컬럼을 검색하기 위한 SYSCAT.COLUMNS에 대한 조회 및 구조화 유형의 속성을 검색하기 위한 SYSCAT.ATTRIBUTES에 대한 조회는 조회의 주제가 서브테이블, 서브뷰 또는 부속 유형일 경우, 버전 5.2에서보다 버전 6에서 더 많은 행을 리턴할 수 있습니다.

설명: 버전 5.2의 경우, 지정된 테이블, 뷰 또는 구조화 유형에 대해 COLUMN 및 ATTRIBUTES 카탈로그는 테이블, 뷰 또는 유형에 의해 소개된 컬럼 및 속성에 대한 항목만 포함했었습니다. 상위테이블 또는 상위유형에서 계승된 컬럼과 속성은 이 카탈로그에 표시되지 않았습니다. 그러나, 버전 6에서는 COLUMNS 및 ATTRIBUTES 카탈로그가 계승된 컬럼과 속성에 대한 항목을 포함합니다.

해결: COLUMNS 및 ATTRIBUTES 카탈로그에서 새로운 항목을 인식하도록 도구 또는 응용프로그램을 변경하십시오.

DB2 Universal Database 버전 6에서 더 이상 지원되지 않는 OBJCAT 뷰

WIN	UNIX	OS/2

변경: 버전 5.2의 OBJCAT 스키마에 있는 순환 카탈로그 뷰는 더 이상 DB2 Universal Database 제품의 일부로 제공되지 않습니다.

증상: OBJCAT 카탈로그 뷰에 대해 작성된 조회는 더 이상 올바로 수행되지 않습니다.

해결: OBJCAT 뷰에 원래 있던 대부분의 정보는 일반적인 SYSCAT 카탈로그 뷰로 통합됩니다. 대부분의 경우, 시스템 카탈로그 뷰에서 정보를 구할 수 있습니다. 버전 5.2로부터 이주할 경우, OBJCAT 카탈로그 뷰가 이미 있으면 이들 뷰는 삭제되어야 합니다. `sqllib` 디렉토리의 `misc` 서브디렉토리하에 있는 `objcatdp.db2`라고 하는 CLP 스크립트를 실행시켜 삭제를 수행할 수 있습니다.

또한 버전 5.2에서 지원되는 카탈로그 뷰에 대응하는 사용자의 OBJCAT 뷰 세트를 작성할 수도 있습니다.

버전 5.2에서, *SQL 참조서*의 ″부록 E″에서는 사용자에게 OBJCAT 카탈로그 뷰가 임시적인 것이며 향후 릴리스에서 지원되지 않을 것이라는 것을 경고했습니다.

DB2 Universal Database 버전 6에서 변경된 종속성 코드

WIN	UNIX	OS/2

변경: 시스템 카탈로그 뷰에서, 이전에 코드 ″H″로 표시된 계층적 종속성이 이제는 코드 ″O″로 표시됩니다.

증상: 카탈로그 뷰에서 코드 ″H″로 계층적 종속성을 검색하는 조회는 더 이상 올바로 작업하지 않습니다.

설명: PACKAGEDEP, TRIGDEP 및 VIEWDEP라는 시스템 카탈로그 뷰를 포함한 여러 시스템 카탈로그는 BTYPE라는 컬럼을 가집니다. 버전 5.2에서, OBJCAT 뷰는 코드 ″H″에 의해 계층적 종속성을 표시했습니다. 버전 6에서, 이 종속성은 코드 ″0″으로 표시됩니다.

해결: 이 조회를 개정하여 코드 ″0″을 검색하십시오.

DB2 Universal Database 버전 6의 SYSIBM 기본 카탈로그 테이블

WIN	UNIX	OS/2

변경: 다음은 SYSCAT 뷰 대신 사용될 수 있는 SYSIBM 기본 카탈로그 테이블에 대한 변경사항입니다.

- 삭제된 필드(SYSCAT 뷰에는 계속 남아 있음):
  - SYSSTMT.SEQNO
  - SYSVIEWS.SEQNO
- 카탈로그 테이블 이름 바꾸기: SYSTRIGDEP가 SYSDEPENDENCIES로 변경되었습니다. 이와 마찬가지로, 컬럼 BCREATOR 및 DCREATOR가 BSCHEMA 및 DSCHEMA로 각각 재명명되었습니다. 뷰 SYSCAT.TRIGDEP는 변경되지 않았습니다.
- 삭제된 필드(SYSCAT 뷰에 없음):
  - SYSATTRIBUTES.DEFAULT_VALUE
  - SYSATTRIBUTES.NULLS
  - SYSCOLUMNS.SERVERTYPE
  - SYSDATATYPES.REFREP_TYPENAME
  - SYSDATATYPES.REFREP_TYPESCHEMA
  - SYSDATATYPES.REFREP_LENGTH
  - SYSDATATYPES.REFREP_SCALE
  - SYSDATATYPES.REFREP_CODEPAGE
  - SYSINDEXES.TEXT

    (뷰에는 있었지만, 단지 향후 사용을 위해 예약됨)
  - SYSPLANDEP.PUBLICPRIV
  - SYSSECTION.SEQNO
  - SYSTABAUTH.UPDATE_BY_COLS
  - SYSTABAUTH.REF_BY_COLS

  - SYSTABLES.MINPDLENGTH
  - SYSTABLESPACES.READONLY
  - SYSTABLESPACES.REMOVABLEMEDIA
- 데이터 유형 변경:
  - SYSSECTION.SECTION이 VARCHAR(3600)에서 CLOB(10M)로 변경됨
  - SYSPLANDEP.COLUSAGE가 VARCHAR(3000) 2진 데이터용에서 BLOB(5K)로 변경됨

응용프로그램 프로그래밍

DB2 Universal Database 버전 6의 VARCHAR 데이터 유형

WIN	UNIX	OS/2

변경: 버전 6에서 VARCHAR (VARGRAPHIC) 데이터 유형의 최대 가능 크기가 4000자(2000 2바이트 문자)에서 32672자(16336 2바이트 문자)로 증가했습니다.

증상: VARCHAR(VARGRAPHIC) 데이터 유형에 대해 4000바이트의 고정 길이 버퍼를 사용하는 응용프로그램은 4000바이트를 넘는 VARCHAR 필드를 너무 작은 버퍼에 페치할 경우 겹쳐쓰여지거나 절단될 가능성이 있습니다. CLI 기능 - SQLGetTypeInfo()가 이제 VARCHAR의 크기를 32672로 리턴합니다. 테이블 DDL에서 이 값을 사용하는 CLI 응용프로그램은 충분한 페이지 크기 테이블 공간이 사용 가능하지 않기 때문에 오류가 생길 수 있습니다. 102 페이지의 『사용자 테이블 데이터』에서 테이블 공간 페이지 크기에 대한 자세한 내용을 참조하십시오.

해결: 응용프로그램을 코드화할 때, 우선 결과 세트의 컬럼을 설명하고(DESCRIBE Statement 사용), DESCRIBE Statement에서 리턴된 길이에 근거한 크기를 갖는 버퍼를 사용하십시오.

DB2 Universal Database 버전 6에서 위치지정된 UPDATE 및 DELETE Java 프로그래밍

WIN	UNIX	OS/2

변경: 버전 6에서 Java를 사용하여 프로그래밍할 때, 위치지정된 UPDATE 및 DELETE문은 커서 패키지를 바인드한 사용자의 기본 권한 부여 ID를 사용합니다. 이는 패키지를 수행하는 사용자의 권한 부여 ID가 사용된 버전 5.2와는 다릅니다.

증상: 위치지정된 UPDATE 및 DELETE문이 있는 패키지는 패키지를 바인드한 사용자의 권한 부여 ID가 충분한 권한을 가지고 있지 않기 때문에 수행되지 않을 수 있습니다.

해결: 패키지를 바인드하는 사용자의 권한 부여 ID는 패키지에 있는 위치지정된 UPDATE 및 DELETE문을 수행하는 데 충분한 권한이 부여되어야 합니다. 올바른 특권을 권한 부여하고 패키지를 다시 바인드하십시오.

DB2 Universal Database 버전 6에서 FOR UPDATE절의 구문 변경

WIN	UNIX	OS/2

변경: 버전 5.2에서, SELECT 문에 있는 FOR UPDATE 절은 차후 위치지정된 UPDATE문에서 갱신될 수 있는 컬럼을 식별하기 위해 SQLJ 프로그램에 사용될 수 있습니다. 버전 6에 대해 구문이 변경됩니다.

증상: SELECT문에 FOR UPDATE절을 포함할 경우, 오류 메세지 SQJ0204E가 나타납니다.

해결: SELECT문에서 FOR UPDATE절을 제거하십시오. 반복기 선언 절을 통해 갱신 가능한 반복기를 지정하십시오. 예를 들면, 다음과 같습니다.

```
#sql public iterator DelByName implements sqlj.runtime.ForUpdate(String EmpNo)
   with updateColumns = (salary);
```

어떤 컬럼이 갱신 가능한지 명시적으로 식별하려고 하면, WITH절과 함께 사용된 `updateColumns` 키워드를 통해 지정해야 합니다.

위치지정된 iterator 선언에 대해서는 *응용프로그램 개발 안내서*에서 자세한 내용을 참조하십시오.

DB2 Universal Database 버전 6의 문자 이름 크기

WIN	UNIX	OS/2

변경: DB2 Universal Database 버전 6은 128바이트 테이블, 뷰 및 별명 이름과 30바이트 컬럼 이름을 지원합니다. 이전에는 이들 엔터티 각각에 대해 18바이트 이름을 지원했었습니다.

USER 및 CURRENT SCHEMA 특수 레지스터는 CHAR(8)이었지만 이제는 VARCHAR(128)입니다. CURRENT EXPLAIN MODE 특수 레지스터는 CHAR(8)이었으며 이제는 VARCHAR(254)입니다. TYPE_SCHEMA 및 TABLE_SCHEMA 내장 함수의 출력은 CHAR(8)이며, 이제 VARCHAR(128)입니다.

증상: 버전 6 전에 개발된 응용프로그램이 더 긴 한계를 활용하지 않는 버전 6 데이터베이스에 대해 수행된 경우, 응용프로그램 활동은 전혀 변경되지 않아야 합니다. 그러나, 더 긴 이름을 사용하는 버전 6 데이터베이스에 대해 이 응용프로그램을 수행하면, 이 응용프로그램이 코드화된 방법에 따라 특정한 결과를 가져올 수 있습니다.

예를 들어, 다음과 같습니다.

- 테이블 또는 컬럼 이름(일반적으로 카탈로그 뷰로부터)을 18바이트로 정의된 호스트 변수에 FETCH하는 기존 응용프로그램을 고려하십시오. 버전 6까지는 18바이트가 테이블이나 컬럼 이름의 크기에 대한 한계이므로, SQLCA의 sqlwarn1 비트를 점검하려 하지 않는 수가 있습니다. 이 응용프로그램은 (적당하지 않게) 결코 절단되지 않을 것이라고 가정합니다.
- SELECT의 DESCRIBE로부터 sqllen 필드를 기준으로 sqldata 필드의 크기가 할당된 SQLDA에 테이블 또는 컬럼 이름(일반적으로 카탈로그 뷰로부터)을 FETCH하는 응용프로그램을 고려하십시오. 이것은 테이블 또는 컬럼 이름의 크기가 늘어났더라도 올바른(절단되지 않은) 결과가 응용프로그램에 리턴되는 결과를 가져옵니다. 컬럼 이름은 18바이트로 제한하며, 리턴되는 더 긴 이름은 예기치 않은 방식으로 처리될 수 있다고 가정하고 다른 응용프로그램 논리가 작동하는 경우, 더 긴 컬럼 이름은 18바이트에서 절단됩니다.

- SQLCA 토큰 필드(sqlerrmc)가 70바이트로 제한되기 때문에, 이는 테이블에 행을 삽입하려고 하는 기존 응용프로그램에 영향을 줄 수 있습니다. 오류 메세지 SQL0204N이 생겼을 때, 이 응용프로그램은 SQLCA sqlerrmc 필드에서 테이블 이름을 결정하고 이 오브젝트 이름에 따라 몇 가지 조작을 수행합니다. 이전 DB2 버전 사용 시, 테이블 또는 스키마 식별자는 전체 테이블 이름이 SQLCA에 포함됨을 보장합니다. 이것은 버전 6에서의 경우가 아닙니다.
- 백 레벨 API를 사용하는 응용프로그램은 테이블 이름의 처음 18바이트만 가져옵니다.
- 스키마 함수(SQLTables() 또는 SQLColumns() 및 기타)를 사용하는 기존 CLI 및 ODBC 응용프로그램은 18-byte바이트를 초과하는 이름을 지원하는 서버에 연결할 경우 영향을 받게 됩니다. 절단 경고가 있더라도 응용프로그램은 이 경고에 대해 점검하지 않고 절단된 이름을 사용하여 계속 수행합니다.

해결: 이러한 유형의 문제점을 해결하는 가장 좋은 방법은 응용프로그램이 긴 테이블 이름 및 컬럼 이름을 처리하도록 다시 코드화하는 것입니다. 그렇지 않으면, 이 응용프로그램이 18바이트보다 긴 이름을 사용하는 버전 6 데이터베이스에 대해 수행하지 않도록 하십시오.

DB2 Universal Database 버전 6에서 PC/IXF 형식 변경

WIN	UNIX	OS/2

변경: DB2 Universal Database 버전 6은 128바이트 테이블, 뷰 및 별명 이름과 30바이트 컬럼 이름을 지원합니다. 이전에는 이들 엔터티 각각에 대해 18바이트 이름을 지원했었습니다.

증상: DB2 Universal Database 버전 5는 DB2 Universal Database 버전 6 클라이언트에 의해 내보낸 PC/IXF를 가져올 수 없습니다. 또한, PC/IXF 파일(DB2 Universal Database 버전 6에서 내보낸 파일)은 DB2 Universal Database 버전 5 데이터베이스에 로드될 수 없습니다(오류 SQL3059N).

해결: PC/IXF 데이터를 가져오거나 로드할 때 DB2 Universal Database의 호환가능한 버전을 사용하십시오.

DB2 Universal Database 버전 6의 Non-doubled SQLVAR에 있는 SQL NAME

WIN	UNIX	OS/2

변경: DB2 Universal Database 버전 6은 30바이트 컬럼 이름을 지원합니다. 원래는 18바이트 이름을 지원했었습니다. 버전 5에서, 문서화된 작동은 ″0x00″가 non-doubled SQLVAR에 대한 SQLNAME 필드의 30번째 바이트에 배치되는 것이었습니다. ″AS″절에 지정된 시스템 생성 이름 및 사용자 지정 컬럼 이름의 경우 ″0x00″가 30번째 바이트에 배치됩니다.

버전 6에서, 시스템이 작성한 이름의 경우에만 ″0xFF″를 30번째 컬럼에 리턴합니다.

증상: SQLNAME의 30번째 바이트로 사용자 지정 컬럼 이름인지 또는 시스템 작성 이름인지를 판별하는 모든 응용프로그램은 사용자 지정 컬럼 이름이 30자일 경우, 예기치 않은 논리 점검을 받게 될 수 있습니다. 이것은 드물게 발생해야 합니다.

해결: 이 응용프로그램은 SQLNAME 필드의 길이가 30바이트 미만일 경우, SQLNAME 필드의 30번째 바이트에 있는 ″0xFF″에 대해서만 점검하도록 수정되어야 합니다. 이 경우, 이름은 사용자 생성 이름입니다.

DB2 Universal Database 버전 6에서 사용하지 않는 DB2 CLI/ODBC 구성 키워드

WIN		

변경: DB2 UDB의 새 버전으로 이주할 때, `db2cli.ini` 파일에 있는 선택적 키워드 세트를 지정하여 DB2 CLI/ODBC 드라이버의 작동을 변경할 수 있습니다.

버전 6에서, TRANSLATEDLL 및 TRANSLATEOPTION 키워드는 불필요해집니다.

증상: 이들 키워드가 여전히 존재하더라도 무시됩니다. 이들 설정값 제거에 근거한 수행이 나타날 수 있습니다.

해결: 사용자의 환경에 적당한 키워드 및 설정값을 결정하기 위해 유효한 매개변수의 새로운 목록을 검토할 필요가 있습니다. *CLI Guide and Reference*에서 이 키워드에 대한 자세한 내용을 참조하십시오.

DB2 Universal Database 버전 6의 이벤트 모니터 출력 스트림 형식

WIN	UNIX	OS/2

변경: 이벤트 모니터 출력 스트림에는 버전 제어가 없습니다. 결과적으로, 18바이트가 넘는 테이블 이름에 대한 지원을 추가하면 출력 스트림 형식으로 이동할 필요가 있습니다.

증상: 이벤트 모니터 출력 스트림을 분석하는 응용프로그램은 더 이상 제대로 작업하지 않습니다.

해결: 다음과 같은 두 가지 옵션이 있습니다.

- 새로운 데이터 스트림을 사용하도록 응용프로그램을 갱신하십시오.
- 레지스트리 변수를 설정하십시오.

  ```
  DB2OLDEVMON=evmonname1,evmonname2,...
  ```

  여기서 *evmonname*은 이전 데이터 형식으로 작성하려는 이벤트 모니터의 이름입니다. 이벤트 모니터의 새로운 모든 필드는 이전 형식으로 액세스할 수 없다는 것을 유의하십시오.

SQL

DB2 Universal Database 버전 6의 DATALINK 컬럼

	UNIX	

변경: DB2 Universal Database 버전 6에 삽입된 DATALINK 값은 컬럼 값 설명자에서 4바이트 추가 공간을 요구합니다.

증상: 버전 5.2에서 작성된 DATALINK 컬럼이 갱신될 때, 새로운 컬럼 값을 저장하기 위해 데이터 페이지상에 4바이트가 추가로 필요합니다. 그 결과, 데이터 페

이지에 갱신을 완료하기 위한 충분한 공간이 없을 수 있으며 새로운 페이지로 이동되어야 하는 경우가 있습니다. 이것은 갱신 수행으로 공간을 다 써버리게 할 수 있습니다.

해결: 시스템에 더 많은 공간을 추가하여 갱신을 수행할 수 있도록 해야 합니다.

DB2 Universal Database 버전 6의 SYSFUN 문자열 함수 시그니처

WIN	UNIX	OS/2

변경: SYSFUN 스키마에 있는 문자열 함수는 이제 SYSIBM 스키마(내장 함수)에 정의된 향상된 버전을 가집니다. 이들 함수명은 LCASE, LTRIM, RTRIM 및 UCASE입니다.

증상: 명령문을 준비하거나 뷰를 작성할 때, 이들 함수 중에서 리턴된 데이터 유형은 버전 6에서와 다를 수 있습니다. 이것은 내장된 함수(SYSIBM 스키마 아래)가 일반적으로 SYSFUN 스키마에 있는 함수보다 앞에서 분석되기 때문에 발생합니다.

해결: 필요한 조치가 없습니다. 내장 함수는 보통 SYSFUN 스키마에 있는 기능보다 우선합니다. 이전 버전의 작동은 SYSFUN이 SYSIBM을 우선하도록 SQL 경로를 전환하여 재저장될 수 있지만, 성능이 저하됩니다. 스키마 이름 SYSFUN으로 함수 이름을 규정화하여 이전 버전 함수를 호출할 수 있습니다.

이들 함수를 참조하는 이주된 패키지, 뷰, 요약 테이블, 트리거 및 제한조건은 패키지를 명시적으로 바인드하거나 뷰, 요약 테이블, 트리거 또는 제한조건을 재작성하는 것과 같은 명백한 조치를 취하는 경우를 제외하고는 SYSFUN 스키마의 버전을 계속 사용합니다.

SET INTEGRITY가 SET CONSTRAINTS 대체

WIN	UNIX	OS/2

변경: SET CONSTRAINTS문이 SET INTEGRITY문으로 대체되었습니다. 역호환이 가능하도록 DB2 버전 6과 버전 7에서는 두 개의 명령문이 모두 사용됩니다.

DB2 Universal Database 버전 6에서 새로운 무결성 상태로 SYSTABLE 컬럼 변경

WIN	UNIX	OS/2

변경: SYSCAT.TABLES의 CONST_CHECKED 컬럼에 있는 ″U″ 상태는 SET INTEGRITY ... OFF문이 수행될 때 다르게 변합니다.

증상: 버전 6 이전 버전에서 CONST_CHECKED 컬럼에 있는 ″U″ 상태는 SET INTEGRITY ... OFF문이 수행될 때 ″N″으로 변경되었습니다. 이제 ″U″ 상태는 ″W″ 상태로 변합니다.

해결: 필요한 조치가 없습니다. CONST_CHECKED 컬럼에 있는 새로운 ″W″ 상태는 제한조건의 유형이 이전에 사용자에 의해 점검되고, 테이블의 일부 데이터가 무결성에 대해 점검될 필요가 있을 것이라는 것을 나타내는 데 사용됩니다.

″N″ 상태로는 데이터베이스 관리 프로그램이 아직 검증되지 않은 이전 데이터가 있는지 확신할 수 없습니다. 차후의 SET INTEGRITY ... IMMEDIATE CHECKED INCREMENTAL문에서 데이터베이스 관리 프로그램은 오류를 리턴해야 하는데, 새로운 변경사항만을 점검하는 경우 데이터 무결성을 보장할 수 없기 때문입니다. 반면, ″W″ 상태는 다시 ″U″ 상태로 변경되어(INCREMENTAL 옵션이 지정되는 경우) 사용자가 여전히 테이블에서 데이터 무결성에 대해 책임이 있음을 나타낼 수 있습니다. INCREMENTAL 옵션이 지정되지 않으면, 데이터베이스 관리 프로그램은 전체 처리를 선택하여 ″W″ 상태를 ″Y″ 상태로 변경하고 데이터 무결성 유지보수 책임을 가정합니다.

데이터베이스 보안 및 조정

DB2 Universal Database 버전 6에서 클라이언트를 사용하여 데이터베이스 작성

WIN	UNIX	OS/2

변경: 클라이언트가 데이터베이스를 작성하기 위해 사용하는 방식

증상: 데이터베이스를 작성하기 위해 백 레벨 클라이언트를 사용하면 오류를 초래할 수 있습니다.

해결: 데이터베이스를 작성하기 위해 클라이언트를 사용할 때에는 클라이언트 및 서버가 같은 레벨의 DB2 코드를 수행 중인지 확인하십시오.

DB2 Universal Database 버전 6의 계층구조에 필요한 SELECT 특권

WIN 32비트	UNIX	OS/2

변경: 테이블에서의 ONLY 키워드의 스펙은 이제 사용자가 지정된 입력된 테이블의 모든 서브테이블에 대해 SELECT 특권을 가질 것을 요구합니다. 마찬가지로, 뷰에서의 ONLY 키워드의 스펙은 사용자가 지정된 입력된 테이블의 모든 서브뷰에 대해 SELECT 특권을 가질 것을 요구합니다. DB2의 이전 버전은 단지 지정된 테이블 또는 뷰에 대해서만 SELECT 특권을 요구했었습니다.

증상: 두 개의 가능한 증상이 있습니다.

- 패키지가 바인드된 권한 부여 ID가 이름 지정된 입력 테이블(또는 뷰)의 서브테이블에 대해 SELECT 특권이 부족할 경우, FROM절에 ONLY 키워드가 지정된 SQL문이 들어 있는 패키지를 리바인드할 때 권한 부여 오류(SQLCODE -551, SQLSTATE 42501)가 발생합니다.
- 뷰 또는 트리거의 정의에서 FROM절에 ONLY 키워드가 있으면, 뷰 또는 트리거는 계속 정상적으로 작업합니다. 그러나, 뷰 또는 트리거의 정의는 작성자가 이름 지정된 테이블(또는 뷰)의 모든 서브테이블에 대해 SELECT 특권을 보유하는 경우를 제외하고는, 새로운 뷰 또는 트리거를 작성하는 데 더 이상 사용될 수 없습니다.

해결: 패키지를 리바인드하거나 새로운 뷰 또는 트리거를 작성할 필요가 있는 권한 부여 ID는 ONLY 키워드에 따라 지정된 테이블(또는 뷰)의 모든 서브테이블(및 서브뷰)에 대해 SELECT 특권을 권한 부여받아야 합니다.

DB2 Universal Database 버전 6에서 사용하지 않는 프로파일 레지스트리 및 환경 변수

WIN	UNIX	OS/2

변경: 다음 프로파일 레지스트리 또는 환경 변수는 사용되지 않습니다.

- DB2_VECTOR

해결: 이 변수는 더 이상 필요하지 않습니다.

유틸리티 및 도구

DB2 Universal Database 버전 6에서의 현재 Explain 모드

WIN	UNIX	OS/2

변경: ″CURRENT EXPLAIN MODE″ 특수 레지스터의 유형은 CHAR(8)에서 VARCHAR(254)로 변경되었습니다.

증상: 응용프로그램이 유형을 계속 CHAR(8)로 가정할 경우, 이 값은 254에서 8바이트로 절단될 수 있습니다.

해결: 특수 레지스터를 읽는 모든 호스트 변수의 유형을 CHAR(8)에서 VARCHAR(254)로 재정의하십시오.

이 변경은 ″CURRENT EXPLAIN MODE″ 특수 레지스터에 대해 두 개의 새로운 값을 제공하기 위해 필요합니다. 이 새 값은 ″EVALUATE INDEXES″ 및 ″RECOMMEND INDEXES″입니다.

DB2 Universal Database 버전 6의 USING 및 SORT BUFFER 매개변수

WIN	UNIX	OS/2

변경: 버전 6에서, LOAD 명령의 USING 및 SORT BUFFER 매개변수는 더 이상 지원되지 않습니다. 매개변수들이 무시됩니다.

증상: USING 및 SORT BUFFER 매개변수가 로드 유틸리티에 의해 무시되며 더 이상 지원되지 않는다는 경고 메세지를 받게 됩니다.

해결: 경고 메세지를 무시하십시오. 추가 정보는 *데이터 이동 유틸리티 안내 및 참조서*에서 자세한 내용을 참조하십시오.

연결성 및 공존

DB2 Universal Database 버전 6에서 RUMBA를 PCOMM으로 대체

WIN		

변경: 버전 6에서, Windows NT, Windows 98 및 Windows 95에서만(Windows 3.1에서는 아님) RUMBA가 PCOMM으로 대체됩니다.

증상: 없음.

해결: 없음.

구성 매개변수

사용하지 않는 데이터베이스 구성 매개변수

WIN	UNIX	OS/2

변경: 다음은 사용하지 않는 데이터베이스 구성 매개변수입니다.

- DL_NUM_BACKUP(NUM_DB_BACKUP 데이터베이스 구성 매개변수로 대체)

해결: 사용자 응용프로그램에서 이 매개변수에 대한 모든 참조를 제거하십시오.

부록D. 자국어 지원(NLS)

이 절에는 DB2에서 제공하는 자국어 지원(NLS) 정보와 국가/지역, 언어 및 코드 페이지(코드 세트), 응용프로그램과 데이터베이스에 있는 DB2 NLS 구성 및 사용 방법에 대한 정보도 들어 있습니다.

자국어 지원(NLS)

DB2 버전 7은 영어, 프랑스어, 독어, 이태리어, 스페인어, 브라질 포르투갈어, 일본어, 한국어, 중국어, 대만어, 덴마크어, 핀란드어, 노르웨이어, 스웨덴어, 체코어, 네덜란드어, 헝가리어, 폴란드어, 터키어, 러시아어, 불가리아어, 슬로베니아어로 제공됩니다.

국가/지역 코드 및 코드 페이지 지원

258 페이지의 표28에서는 데이터베이스 서버로 지원하는 언어와 코드 세트를 보여주고, 이 값이 데이터베이스 관리 프로그램에서 사용된 국가/지역 코드 및 코드 페이지에 맵핑되는 방법도 보여줍니다.

다음은 테이블 컬럼 각각에 대한 설명입니다.

- **코드 페이지**는 운영 체제 코드 세트에서 맵핑될 때의 IBM 정의 코드 페이지를 보여줍니다.
- **그룹**은 코드 페이지가 1바이트(″S″)인지 아니면 2바이트(″D″)인지를 보여줍니다. ″-n″은 영문자-숫자 조합을 작성하는 데 사용되는 번호입니다. 대응하는 조합은 연결 및 변환이 DB2에서 허용되는 곳을 보여줍니다. 예를 들어, 모든 ″S-1″ 그룹을 함께 작동시킬 수 있습니다.
- **코드 세트**는 지원 언어와 관련된 코드 세트를 보여줍니다. 코드 세트는 DB2 코드 페이지로 맵됩니다.
- **Tr.**은 지역 식별자를 보여줍니다.

- **국가/지역 코드**는 내부적으로 특정 지역 지원을 제공하기 위해 데이터베이스 관리 프로그램이 사용하는 코드를 보여줍니다.
- **로케일**은 데이터베이스 관리 프로그램이 지원하는 로케일 값을 보여줍니다.
- **OS**는 언어 및 코드 세트를 지원하는 운영 체제를 보여줍니다.
- **국가/지역 이름**은 지역, 국가(들)의 이름을 보여줍니다.

표 28. 지원 언어 및 코드 페이지

코드 페이지	그룹	코드 세트	Tr.	국가/지역 코드	로케일	OS	국가/지역 이름
437	S-1	IBM-437	AL	355	-	OS2	Albania
850	S-1	IBM-850	AL	355	-	OS2	Albania
819	S-1	ISO8859-1	AL	355	sq_AL	AIX	Albania
850	S-1	IBM-850	AL	355	Sq_AL	AIX	Albania
819	S-1	iso88591	AL	355	-	HP	Albania
1051	S-1	roman8	AL	355	-	HP	Albania
819	S-1	ISO8859-1	AL	355	-	Sun	Albania
1252	S-1	1252	AL	355	-	WIN	Albania
1275	S-1	1275	AL	355	-	Mac	Albania
37	S-1	IBM-37	AL	355	-	HOST	Albania
1140	S-1	IBM-1140	AL	355	-	HOST	Albania
864	S-6	IBM-864	AA	785	-	OS2	Arabic Countries
1046	S-6	IBM-1046	AA	785	Ar_AA	AIX	Arabic Countries
1089	S-6	ISO8859-6	AA	785	ar_AA	AIX	Arabic Countries
1089	S-6	iso88596	AA	785	ar_SA.iso88596	HP	Arabic Countries
1256	S-6	1256	AA	785	-	WIN	Arabic Countries
420	S-6	IBM-420	AA	785	-	HOST	Arabic Countries
437	S-1	IBM-437	AU	61	-	OS2	Australia
850	S-1	IBM-850	AU	61	-	OS2	Australia
819	S-1	ISO8859-1	AU	61	en_AU	AIX	Australia
850	S-1	IBM-850	AU	61	En_AU	AIX	Australia
819	S-1	iso88591	AU	61	-	HP	Australia
1051	S-1	roman8	AU	61	-	HP	Australia
819	S-1	ISO8859-1	AU	61	en_AU	Sun	Australia
819	S-1	ISO8859-1	AU	61	en_AU	SCO	Australia
1252	S-1	1252	AU	61	-	WIN	Australia
1275	S-1	1275	AU	61	-	Mac	Australia
37	S-1	IBM-37	AU	61	-	HOST	Australia
1140	S-1	IBM-1140	AU	61	-	HOST	Australia

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
437	S-1	IBM-437	AT	43	-	OS2	Austria
850	S-1	IBM-850	AT	43	-	OS2	Austria
819	S-1	ISO8859-1	AT	43	ge_AT	AIX	Austria
850	S-1	IBM-850	AT	43	Ge_AT	AIX	Austria
819	S-1	iso88591	AT	43	-	HP	Austria
1051	S-1	roman8	AT	43	-	HP	Austria
819	S-1	ISO8859-1	AT	43	de_AT	SCO	Austria
819	S-1	ISO-8859-1	AT	43	de_AT	Linux	Austria
819	S-1	ISO8859-1	AT	43	de_AT	Sun	Austria
1252	S-1	1252	AT	43	-	WIN	Austria
1275	S-1	1275	AT	43	-	Mac	Austria
37	S-1	IBM-37	AT	43	-	HOST	Austria
1140	S-1	IBM-1140	AT	43	-	HOST	Austria
915	S-5	ISO8859-5	BY	375	-	OS2	Belarus
915	S-5	ISO8859-5	BY	375	be_BY	AIX	Belarus
1131	S-5	IBM-1131	BY	375	-	OS2	Belarus
1251	S-5	1251	BY	375	-	WIN	Belarus
1283	S-5	1283	BY	375	-	Mac	Belarus
1025	S-5	IBM-1025	BY	375	-	HOST	Belarus
274	S-1	IBM-274	BE	32	-	HOST	Belgium
437	S-1	IBM-437	BE	32	-	OS2	Belgium
850	S-1	IBM-850	BE	32	-	OS2	Belgium
819	S-1	ISO8859-1	BE	32	nl_BE	AIX	Belgium
850	S-1	IBM-850	BE	32	Nl_BE	AIX	Belgium
819	S-1	iso88591	BE	32	-	HP	Belgium
819	S-1	ISO8859-1	BE	32	fr_BE	SCO	Belgium
819	S-1	ISO8859-1	BE	32	nl_BE	SCO	Belgium
819	S-1	ISO-8859-1	BE	32	nl_BE	Linux	Belgium
819	S-1	ISO8859-1	BE	32	nl_BE	Sun	Belgium
1252	S-1	1252	BE	32	-	WIN	Belgium
1275	S-1	1275	BE	32	-	Mac	Belgium
500	S-1	IBM-500	BE	32	-	HOST	Belgium
1148	S-1	IBM-1148	BE	32	-	HOST	Belgium
855	S-5	IBM-855	BG	359	-	OS2	Bulgaria
915	S-5	ISO8859-5	BG	359	-	OS2	Bulgaria
915	S-5	ISO8859-5	BG	359	bg_BG	AIX	Bulgaria
915	S-5	iso88595	BG	359	bg_BG.iso88595	HP	Bulgaria
1251	S-5	1251	BG	359	-	WIN	Bulgaria
1283	S-5	1283	BG	359	-	Mac	Bulgaria
1025	S-5	IBM-1025	BG	359	-	HOST	Bulgaria

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
850	S-1	IBM-850	BR	55	-	OS2	Brazil
850	S-1	IBM-850	BR	55	-	AIX	Brazil
819	S-1	ISO8859-1	BR	55	pt_BR	AIX	Brazil
819	S-1	ISO8859-1	BR	55	-	HP	Brazil
819	S-1	ISO8859-1	BR	55	pt_BR	SCO	Brazil
819	S-1	ISO8859-1	BR	55	pt_BR	Sun	Brazil
819	S-1	ISO-8859-1	BR	55	pt_BR	Linux	Brazil
1252	S-1	1252	BR	55	-	WIN	Brazil
37	S-1	IBM-37	BR	55	-	HOST	Brazil
1140	S-1	IBM-1140	BR	55	-	HOST	Brazil
850	S-1	IBM-850	CA	1	-	OS2	Canada
850	S-1	IBM-850	CA	1	En_CA	AIX	Canada
819	S-1	ISO8859-1	CA	1	en_CA	AIX	Canada
819	S-1	iso88591	CA	1	fr_CA.iso88591	HP	Canada
1051	S-1	roman8	CA	1	fr_CA.roman8	HP	Canada
819	S-1	ISO8859-1	CA	1	en_CA	SCO	Canada
819	S-1	ISO8859-1	CA	1	fr_CA	SCO	Canada
819	S-1	ISO8859-1	CA	1	en_CA	Sun	Canada
819	S-1	ISO8859-1	CA	1	en_CA	Sun	Canada
819	S-1	ISO-8859-1	CA	1	en_CA	Linux	Canada
1252	S-1	1252	CA	1	-	WIN	Canada
1275	S-1	1275	CA	1	-	Mac	Canada
37	S-1	IBM-37	CA	1	-	HOST	Canada
1140	S-1	IBM-1140	CA	1	-	HOST	Canada
863	S-1	IBM-863	CA	2	-	OS2	Canada (French)
1381	D-4	IBM-1381	CN	86	-	OS2	China (PRC)
1386	D-4	GBK	CN	86	-	OS2	China (PRC)
1383	D-4	IBM-eucCN	CN	86	zh_CN	AIX	China (PRC)
1386	D-4	GBK	CN	86	Zh_CN.GBK	AIX	China (PRC)
1383	D-4	hp15CN	CN	86	zh_CN.hp15CN	HP	China (PRC)
1383	D-4	eucCN	CN	86	zh_CN	SCO	China (PRC)
1383	D-4	eucCN	CN	86	zh_CN.eucCN	SCO	China (PRC)
1383	D-4	gb2312	CN	86	zh	Sun	China (PRC)
1381	D-4	IBM-1381	CN	86	-	WIN	China (PRC)
1386	D-4	GBK	CN	86	-	WIN	China (PRC)
935	D-4	IBM-935	CN	86	-	HOST	China (PRC)
1388	D-4	IBM-1388	CN	86	-	HOST	China (PRC)
5488**	D-4		CN	86	-		China (PRC)

** Code page 5488 can only be used with the LOAD or IMPORT utilities to move data from code page 5488 to a DB2 Unicode database, or to EXPORT from a DB2 Unicode database to code page 5488. For more information, see the Data Movement Utilities Guide and Reference section of the Version 7.2 FixPak 4 Release Notes.

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/지역 코드	로케일	OS	국가/지역 이름
852	S-2	IBM-852	HR	385	-	OS2	Croatia
912	S-2	ISO8859-2	HR	385	hr_HR	AIX	Croatia
912	S-2	iso88592	HR	385	hr_HR.iso88592	HP	Croatia
912	S-2	ISO8859-2	HR	385	hr_HR.ISO8859-2	SCO	Croatia
912	S-2	ISO-8859-2	HR	385	hr_HR	Linux	Croatia
1250	S-2	1250	HR	385	-	WIN	Croatia
1282	S-2	1282	HR	385	-	Mac	Croatia
870	S-2	IBM-870	HR	385	-	HOST	Croatia
852	S-2	IBM-852	CZ	421	-	OS2	Czech Republic
912	S-2	ISO8859-2	CZ	421	cs_CZ	AIX	Czech Republic
912	S-2	iso88592	CZ	421	cs_CZ.iso88592	HP	Czech Republic
912	S-2	ISO8859-2	CZ	421	cs_CZ.ISO8859-2	SCO	Czech Republic
912	S-2	ISO-8859-2	CZ	421	cs_CZ	Linux	Czech Republic
1250	S-2	1250	CZ	421	-	WIN	Czech Republic
1282	S-2	1282	CZ	421	-	Mac	Czech Republic
870	S-2	IBM-870	CZ	421	-	HOST	Czech Republic
850	S-1	IBM-850	DK	45	-	OS2	Denmark
819	S-1	ISO8859-1	DK	45	da_DK	AIX	Denmark
850	S-1	IBM-850	DK	45	Da_DK	AIX	Denmark
819	S-1	iso88591	DK	45	da_DK.iso88591	HP	Denmark
1051	S-1	roman8	DK	45	da_DK.roman8	HP	Denmark
819	S-1	ISO8859-1	DK	45	da	SCO	Denmark
819	S-1	ISO8859-1	DK	45	da_DA	SCO	Denmark
819	S-1	ISO8859-1	DK	45	da_DK	SCO	Denmark
819	S-1	ISO8859-1	DK	45	da	Sun	Denmark
819	S-1	ISO8859-1	DK	45	da	Sun	Denmark
819	S-1	ISO-8859-1	DK	45	da_DK	Linux	Denmark
1252	S-1	1252	DK	45	-	WIN	Denmark
1275	S-1	1275	DK	45	-	Mac	Denmark
277	S-1	IBM-277	DK	45	-	HOST	Denmark
1142	S-1	IBM-1142	DK	45	-	HOST	Denamrk
922	S-10	IBM-922	EE	372	-	OS2	Estonia
922	S-10	IBM-922	EE	372	Et_EE	AIX	Estonia
1257	S-10	1257	EE	372	-	WIN	Estonia
1122	S-10	IBM-1122	EE	372	-	HOST	Estonia

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
437	S-1	IBM-437	FI	358	-	OS2	Finland
850	S-1	IBM-850	FI	358	-	OS2	Finland
819	S-1	ISO8859-1	FI	358	fi_FI	AIX	Finland
850	S-1	IBM-850	FI	358	Fi_FI	AIX	Finland
819	S-1	iso88591	FI	358	fi_FI.iso88591	HP	Finland
819	S-1	ISO8859-1	FI	358	fi	SCO	Finland
819	S-1	ISO8859-1	FI	358	fi_FI	SCO	Finland
819	S-1	ISO8859-1	FI	358	sv_FI	SCO	Finland
819	S-1	ISO8859-1	FI	358	-	Sun	Finland
819	S-1	ISO-8859-1	FI	358	fi_FI	Linux	Finland
1051	S-1	roman8	FI	358	-	HP	Finland
1252	S-1	1252	FI	358	-	WIN	Finland
1275	S-1	1275	FI	358	-	Mac	Finland
278	S-1	IBM-278	FI	358	-	HOST	Finland
1143	S-1	IBM-1143	FI	358	-	HOST	Finland
855	S-5	IBM-855	MK	389	-	OS2	FYR Macedonia
915	S-5	ISO8859-5	MK	389	-	OS2	FYR Macedonia
915	S-5	ISO8859-5	MK	389	mk_MK	AIX	FYR Macedonia
915	S-5	iso88595	MK	389	-	HP	FYR Macedonia
1251	S-5	1251	MK	389	-	WIN	FYR Macedonia
1283	S-5	1283	MK	389	-	Mac	FYR Macedonia
1025	S-5	IBM-1025	MK	389	-	HOST	FYR Macedonia
437	S-1	IBM-437	FR	33	-	OS2	France
850	S-1	IBM-850	FR	33	-	OS2	France
819	S-1	ISO8859-1	FR	33	fr_FR	AIX	France
850	S-1	IBM-850	FR	33	Fr_FR	AIX	France
819	S-1	iso88591	FR	33	fr_FR.iso88591	HP	France
1051	S-1	roman8	FR	33	fr_FR.roman8	HP	France
819	S-1	ISO8859-1	FR	33	fr	Sun	France
819	S-1	ISO8859-1	FR	33	fr	SCO	France
819	S-1	ISO8859-1	FR	33	fr_FR	SCO	France
819	S-1	ISO-8859-1	FR	33	fr_FR	Linux	France
1252	S-1	1252	FR	33	-	WIN	France
1275	S-1	1275	FR	33	-	Mac	France
297	S-1	IBM-297	FR	33	-	HOST	France
1147	S-1	IBM-1147	FR	33	-	HOST	France

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
437	S-1	IBM-437	DE	49	-	OS2	Germany
850	S-1	IBM-850	DE	49	-	OS2	Germany
819	S-1	ISO8859-1	DE	49	de_DE	AIX	Germany
850	S-1	IBM-850	DE	49	De_DE	AIX	Germany
819	S-1	iso88591	DE	49	de_DE.iso88591	HP	Germany
1051	S-1	roman8	DE	49	de_DE.roman8	HP	Germany
819	S-1	ISO8859-1	DE	49	de	SCO	Germany
819	S-1	ISO8859-1	DE	49	de_DE	SCO	Germany
819	S-1	ISO8859-1	DE	49	de	Sun	Germany
819	S-1	ISO-8859-1	DE	49	de_DE	Linux	Germany
1252	S-1	1252	DE	49	-	WIN	Germany
1275	S-1	1275	DE	49	-	Mac	Germany
273	S-1	IBM-273	DE	49	-	HOST	Germany
1141	S-1	IBM-1141	DE	49	-	HOST	Germany
819	S-1	ISO8859-1	DE	49	De_DE.88591	SINIX	Germany
819	S-1	ISO8859-1	DE	49	De_DE.6937	SINIX	Germany
813	S-7	ISO8859-7	GR	30	-	OS2	Greece
869	S-7	IBM-869	GR	30	-	OS2	Greece
813	S-7	ISO8859-7	GR	30	el_GR	AIX	Greece
813	S-7	iso88597	GR	30	el_GR.iso88597	HP	Greece
813	S-7	ISO8859-7	GR	30	el_GR.ISO8859-7	SCO	Greece
813	S-7	ISO-8859-7	GR	30	gr_GR	Linux	Greece
737	S-7	737	GR	30	-	WIN	Greece
1253	S-7	1253	GR	30	-	WIN	Greece
1280	S-7	1280	GR	30	-	Mac	Greece
423	S-7	IBM-423	GR	30	-	HOST	Greece
875	S-7	IBM-875	GR	30	-	HOST	Greece
852	S-2	IBM-852	HU	36	-	OS2	Hungary
912	S-2	ISO8859-2	HU	36	hu_HU	AIX	Hungary
912	S-2	iso88592	HU	36	hu_HU.iso88592	HP	Hungary
912	S-2	ISO8859-2	HU	36	hu_HU.ISO8859-2	SCO	Hungary
912	S-2	ISO-8859-2	HU	36	hu_HU	Linux	Hungary
1250	S-2	1250	HU	36	-	WIN	Hungary
1282	S-2	1282	HU	36	-	Mac	Hungary
870	S-2	IBM-870	HU	36	-	HOST	Hungary
850	S-1	IBM-850	IS	354	-	OS2	Iceland
819	S-1	ISO8859-1	IS	354	is_IS	AIX	Iceland
850	S-1	IBM-850	IS	354	Is_IS	AIX	Iceland
819	S-1	iso88591	IS	354	is_IS.iso88591	HP	Iceland
1051	S-1	roman8	IS	354	is_IS.roman8	HP	Iceland
819	S-1	ISO8859-1	IS	354	is	SCO	Iceland
819	S-1	ISO8859-1	IS	354	is_IS	SCO	Iceland
819	S-1	ISO8859-1	IS	354	-	Sun	Iceland
819	S-1	ISO-8859-1	IS	354	is_IS	Linux	Iceland
1252	S-1	1252	IS	354	-	WIN	Iceland
1275	S-1	1275	IS	354	-	Mac	Iceland
871	S-1	IBM-871	IS	354	-	HOST	Iceland
1149	S-1	IBM-1149	IS	354	-	HOST	Iceland

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
437	S-1	IBM-437	IE	353	-	OS2	Ireland
850	S-1	IBM-850	IE	353	-	OS2	Ireland
819	S-1	ISO8859-1	IE	353	en_IE	AIX	Ireland
850	S-1	IBM-850	IE	353	En_IE	AIX	Ireland
819	S-1	iso88591	IE	353	-	HP	Ireland
1051	S-1	roman8	IE	353	-	HP	Ireland
819	S-1	ISO8859-1	IE	353	en_IE	Sun	Ireland
819	S-1	ISO8859-1	IE	353	en_IE.ISO8859-1	SCO	Ireland
819	S-1	ISO-8859-1	IE	353	en_IE	Linux	Ireland
1252	S-1	1252	IE	353	-	WIN	Ireland
1275	S-1	1275	IE	353	-	Mac	Ireland
285	S-1	IBM-285	IE	353	-	HOST	Ireland
1146	S-1	IBM-1146	IE	353	-	HOST	Ireland
806	S-12	IBM-806	IN	91	hi_IN	-	India
1137	S-12	IBM-1137	IN	91	-	HOST	India
862	S-8	IBM-862	IL	972	-	OS2	Israel
916	S-8	ISO8859-8	IL	972	iw_IL	AIX	Israel
856	S-8	IBM-856	IL	972	Iw_IL	AIX	Israel
916	S-8	ISO-8859-8	IL	972	iw_IL	Linux	Israel
1255	S-8	1255	IL	972	-	WIN	Israel
424	S-8	IBM-424	IL	972	-	HOST	Israel
437	S-1	IBM-437	IT	39	-	OS2	Italy
850	S-1	IBM-850	IT	39	-	OS2	Italy
819	S-1	ISO8859-1	IT	39	it_IT	AIX	Italy
850	S-1	IBM-850	IT	39	It_IT	AIX	Italy
819	S-1	iso88591	IT	39	it_IT.iso88591	HP	Italy
1051	S-1	roman8	IT	39	it_IT.roman8	HP	Italy
819	S-1	ISO8859-1	IT	39	it	SCO	Italy
819	S-1	ISO8859-1	IT	39	it_IT	SCO	Italy
819	S-1	ISO8859-1	IT	39	it	Sun	Italy
819	S-1	ISO-8859-1	IT	39	it_IT	Linux	Italy
1252	S-1	1252	IT	39	-	WIN	Italy
1275	S-1	1275	IT	39	-	Mac	Italy
280	S-1	IBM-280	IT	39	-	HOST	Italy
1144	S-1	IBM-1144	IT	39	-	HOST	Italy

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
932	D-1	IBM-932	JP	81	-	OS2	Japan
942	D-1	IBM-942	JP	81	-	OS2	Japan
943	D-1	IBM-943	JP	81	-	OS2	Japan
954	D-1	IBM-eucJP	JP	81	ja_JP	AIX	Japan
943*	D-1	IBM-943	JP	81	Ja_JP	AIX	Japan
954	D-1	eucJP	JP	81	ja_JP.eucJP	HP	Japan
5039	D-1	SJIS	JP	81	ja_JP.SJIS	HP	Japan
954	D-1	eucJP	JP	81	ja	SCO	Japan
954	D-1	eucJP	JP	81	ja_JP	SCO	Japan
954	D-1	eucJP	JP	81	ja_JP.EUC	SCO	Japan
954	D-1	eucJP	JP	81	ja_JP.eucJP	SCO	Japan
954	D-1	eucJP	JP	81	ja	Sun	Japan
943	D-1	IBM-943	JP	81	ja_JP.PCK	Sun	Japan
954	D-1	EUC-JP	JP	81	ja_JP	Linux	Japan
943	D-1	IBM-943	JP	81	-	WIN	Japan
930	D-1	IBM-930	JP	81	-	HOST	Japan
939	D-1	IBM-939	JP	81	-	HOST	Japan
5026	D-1	IBM-5026	JP	81	-	HOST	Japan
5035	D-1	IBM-5035	JP	81	-	HOST	Japan
1390	D-1		JP	81	-	HOST	Japan
1399	D-1		JP	81	-	HOST	Japan
1394**	D-1		JP	81	-		Japan
949	D-3	IBM-949	KR	82	-	OS2	Korea, South
970	D-3	IBM-eucKR	KR	82	ko_KR	AIX	Korea, South
970	D-3	eucKR	KR	82	ko_KR.eucKR	HP	Korea, South
970	D-3	eucKR	KR	82	ko_KR.eucKR	SGI	Korea, South
970	D-3	5601	KR	82	ko	Sun	Korea, South
1363	D-3	1363	KR	82	-	WIN	Korea, South
933	D-3	IBM-933	KR	82	-	HOST	Korea, South
1364	D-3	IBM-1364	KR	82	-	HOST	Korea, South
437	S-1	IBM-437	Lat	3	-	OS2	Latin America
850	S-1	IBM-850	Lat	3	-	OS2	Latin America
819	S-1	ISO8859-1	Lat	3	-	AIX	Latin America
850	S-1	IBM-850	Lat	3	-	AIX	Latin America
819	S-1	iso88591	Lat	3	-	HP	Latin America
819	S-1	ISO8859-1	Lat	3	-	Sun	Latin America
819	S-1	ISO-8859-1	Lat	3	-	Linux	Latin America
1051	S-1	roman8	Lat	3	-	HP	Latin America
1252	S-1	1252	Lat	3	-	WIN	Latin America
1275	S-1	1275	Lat	3	-	Mac	Latin America
284	S-1	IBM-284	Lat	3	-	HOST	Latin America
1145	S-1	IBM-1145	Lat	3	-	HOST	Latin America

* On AIX 4.3 or later the code page is 943. If you are using AIX 4.2 or earlier, the code page is 932.

** Code page 1394 can only be used with the LOAD or IMPORT utilities to move data from code page 1394 to a DB2 Unicode database, or to EXPORT from a DB2 Unicode database to code page 1394. For more information, see the Data Movement Utilities Guide and Reference section of the Version 7.2 FixPak 4 Release Notes.

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
921	S-10	IBM-921	LV	371	-	OS2	Latvia
921	S-10	IBM-921	LV	371	Lv_LV	AIX	Latvia
1257	S-10	1257	LV	371	-	WIN	Latvia
1112	S-10	IBM-1112	LV	371	-	HOST	Latvia
921	S-10	IBM-921	LT	370	-	OS2	Lithuania
921	S-10	IBM-921	LT	370	Lt_LT	AIX	Lithuania
1257	S-10	1257	LT	370	-	WIN	Lithuania
1112	S-10	IBM-1112	LT	370	-	HOST	Lithuania
437	S-1	IBM-437	NL	31	-	OS2	Netherlands
850	S-1	IBM-850	NL	31	-	OS2	Netherlands
819	S-1	ISO8859-1	NL	31	nl_NL	AIX	Netherlands
850	S-1	IBM-850	NL	31	Nl_NL	AIX	Netherlands
819	S-1	iso88591	NL	31	nl_NL.iso88591	HP	Netherlands
1051	S-1	roman8	NL	31	nl_NL.roman8	HP	Netherlands
819	S-1	ISO8859-1	NL	31	nl	SCO	Netherlands
819	S-1	ISO8859-1	NL	31	nl_NL	SCO	Netherlands
819	S-1	ISO8859-1	NL	31	nl	Sun	Netherlands
819	S-1	ISO-8859-1	NL	31	nl_NL	Linux	Netherlands
1252	S-1	1252	NL	31	-	WIN	Netherlands
1275	S-1	1275	NL	31	-	Mac	Netherlands
37	S-1	IBM-37	NL	31	-	HOST	Netherlands
1140	S-1	IBM-1140	NL	31	-	HOST	Netherlands
850	S-1	IBM-850	NZ	64	-	OS2	New Zealand
850	S-1	IBM-850	NZ	64	En_NZ	AIX	New Zealand
819	S-1	ISO8859-1	NZ	64	en_NZ	AIX	New Zealand
819	S-1	ISO8859-1	NZ	64	-	HP	New Zealand
819	S-1	ISO8859-1	NZ	64	en_NZ	SCO	New Zealand
819	S-1	ISO8859-1	NZ	64	en_NZ	Sun	New Zealand
1252	S-1	1252	NZ	64	-	WIN	New Zealand
37	S-1	IBM-37	NZ	64	-	HOST	New Zealand
1140	S-1	IBM-1140	NZ	64	-	HOST	New Zealand
850	S-1	IBM-850	NO	47	-	OS2	Norway
819	S-1	ISO8859-1	NO	47	no_NO	AIX	Norway
850	S-1	IBM-850	NO	47	No_NO	AIX	Norway
819	S-1	iso88591	NO	47	no_NO.iso88591	HP	Norway
1051	S-1	roman8	NO	47	no_NO.roman8	HP	Norway
819	S-1	ISO8859-1	NO	47	no	SCO	Norway
819	S-1	ISO8859-1	NO	47	no_NO	SCO	Norway
819	S-1	ISO8859-1	NO	47	no	Sun	Norway
819	S-1	ISO-8859-1	NO	47	no_NO	Linux	Norway
1252	S-1	1252	NO	47	-	WIN	Norway
1275	S-1	1275	NO	47	-	Mac	Norway
277	S-1	IBM-277	NO	47	-	HOST	Norway
1142	S-1	IBM-1142	NO	47	-	HOST	Norway

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/지역 코드	로케일	OS	국가/지역 이름
852	S-2	IBM-852	PL	48	-	OS2	Poland
912	S-2	ISO8859-2	PL	48	pl_PL	AIX	Poland
912	S-2	iso88592	PL	48	pl_PL.iso88592	HP	Poland
912	S-2	ISO8859-2	PL	48	pl_PL.ISO8859-2	SCO	Poland
912	S-2	ISO-8859-2	PL	48	pl_PL	Linux	Poland
1250	S-2	1250	PL	48	-	WIN	Poland
1282	S-2	1282	PL	48	-	Mac	Poland
870	S-2	IBM-870	PL	48	-	HOST	Poland
860	S-1	IBM-860	PT	351	-	OS2	Portugal
850	S-1	IBM-850	PT	351	-	OS2	Portugal
819	S-1	ISO8859-1	PT	351	pt_PT	AIX	Portugal
850	S-1	IBM-850	PT	351	Pt_PT	AIX	Portugal
819	S-1	iso88591	PT	351	pt_PT.iso88591	HP	Portugal
1051	S-1	roman8	PT	351	pt_PT.roman8	HP	Portugal
819	S-1	ISO8859-1	PT	351	pt	SCO	Portugal
819	S-1	ISO8859-1	PT	351	pt_PT	SCO	Portugal
819	S-1	ISO8859-1	PT	351	pt	Sun	Portugal
819	S-1	ISO-8859-1	PT	351	pt_PT	Linux	Portugal
1252	S-1	1252	PT	351	-	WIN	Portugal
1275	S-1	1275	PT	351	-	Mac	Portugal
37	S-1	IBM-37	PT	351	-	HOST	Portugal
1140	S-1	IBM-1140	PT	351	-	HOST	Portugal
852	S-2	IBM-852	RO	40	-	OS2	Romania
912	S-2	ISO8859-2	RO	40	ro_RO	AIX	Romania
912	S-2	iso88592	RO	40	ro_RO.iso88592	HP	Romania
912	S-2	ISO8859-2	RO	40	ro_RO.ISO8859-2	SCO	Romania
912	S-2	ISO-8859-2	RO	40	ro_RO	Linux	Romania
1250	S-2	1250	RO	40	-	WIN	Romania
1282	S-2	1282	RO	40	-	Mac	Romania
870	S-2	IBM-870	RO	40	-	HOST	Romania
866	S-5	IBM-866	RU	7	-	OS2	Russia
915	S-5	ISO8859-5	RU	7	-	OS2	Russia
915	S-5	ISO8859-5	RU	7	ru_RU	AIX	Russia
915	S-5	iso88595	RU	7	ru_RU.iso88595	HP	Russia
915	S-5	ISO8859-5	RU	7	ru_RU.ISO8859-5	SCO	Russia
915	S-5	ISO-8859-5	RU	7	ru_RU	Linux	Russia
1251	S-5	1251	RU	7	-	WIN	Russia
1283	S-5	1283	RU	7	-	Mac	Russia
1025	S-5	IBM-1025	RU	7	-	HOST	Russia
855	S-5	IBM-855	SP	381	-	OS2	Serbia/Montenegro
915	S-5	ISO8859-5	SP	381	-	OS2	Serbia/Montenegro
915	S-5	ISO8859-5	SP	381	sr_SP	AIX	Serbia/Montenegro
915	S-5	iso88595	SP	381	-	HP	Serbia/Montenegro
1251	S-5	1251	SP	381	-	WIN	Serbia/Montenegro
1283	S-5	1283	SP	381	-	Mac	Serbia/Montenegro
1025	S-5	IBM-1025	SP	381	-	HOST	Serbia/Montenegro

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
852	S-2	IBM-852	SK	422	-	OS2	Slovakia
912	S-2	ISO8859-2	SK	422	sk_SK	AIX	Slovakia
912	S-2	iso88592	SK	422	sk_SK.iso88592	HP	Slovakia
912	S-2	ISO8859-2	SK	422	sk_SK.ISO8859-2	SCO	Slovakia
1250	S-2	1250	SK	422	-	WIN	Slovakia
1282	S-2	1282	SK	422	-	Mac	Slovakia
870	S-2	IBM-870	SK	422	-	HOST	Slovakia
852	S-2	IBM-852	SI	386	-	OS2	Slovenia
912	S-2	ISO8859-2	SI	386	sl_SI	AIX	Slovenia
912	S-2	iso88592	SI	386	sl_SI.iso88592	HP	Slovenia
912	S-2	ISO8859-2	SI	386	sl_SI.ISO8859-2	SCO	Slovenia
912	S-2	ISO-8859-2	SI	386	sl_SI	Linux	Slovenia
1250	S-2	1250	SI	386	-	WIN	Slovenia
1282	S-2	1282	SI	386	-	Mac	Slovenia
870	S-2	IBM-870	SI	386	-	HOST	Slovenia
437	S-1	IBM-437	ZA	27	-	OS2	South Africa
850	S-1	IBM-850	ZA	27	-	OS2	South Africa
819	S-1	ISO8859-1	ZA	27	en_ZA	AIX	South Africa
850	S-1	IBM-850	ZA	27	En_ZA	AIX	South Africa
819	S-1	iso88591	ZA	27	-	HP	South Africa
1051	S-1	roman8	ZA	27	-	HP	South Africa
819	S-1	ISO8859-1	ZA	27	-	Sun	South Africa
819	S-1	ISO8859-1	ZA	27	en_ZA.ISO8859-1	SCO	South Africa
1252	S-1	1252	ZA	27	-	WIN	South Africa
1275	S-1	1275	ZA	27	-	Mac	South Africa
285	S-1	IBM-285	ZA	27	-	HOST	South Africa
1146	S-1	IBM-1146	ZA	27	-	HOST	South Africa
437	S-1	IBM-437	ES	34	-	OS2	Spain
850	S-1	IBM-850	ES	34	-	OS2	Spain
819	S-1	ISO8859-1	ES	34	es_ES	AIX	Spain
850	S-1	IBM-850	ES	34	Es_ES	AIX	Spain
819	S-1	iso88591	ES	34	es_ES.iso88591	HP	Spain
1051	S-1	roman8	ES	34	es_ES.roman8	HP	Spain
819	S-1	ISO8859-1	ES	34	es	Sun	Spain
819	S-1	ISO8859-1	ES	34	es	SCO	Spain
819	S-1	ISO8859-1	ES	34	es_ES	SCO	Spain
819	S-1	ISO-8859-1	ES	34	es_ES	Linux	Spain
1252	S-1	1252	ES	34	-	WIN	Spain
1275	S-1	1275	ES	34	-	Mac	Spain
284	S-1	IBM-284	ES	34	-	HOST	Spain
1145	S-1	IBM-1145	ES	34	-	HOST	Spain

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
437	S-1	IBM-437	SE	46	-	OS2	Sweden
850	S-1	IBM-850	SE	46	-	OS2	Sweden
819	S-1	ISO8859-1	SE	46	sv_SE	AIX	Sweden
850	S-1	IBM-850	SE	46	Sv_SE	AIX	Sweden
819	S-1	iso88591	SE	46	sv_SE.iso88591	HP	Sweden
1051	S-1	roman8	SE	46	sv_SE.roman8	HP	Sweden
819	S-1	ISO8859-1	SE	46	sv	SCO	Sweden
819	S-1	ISO8859-1	SE	46	sv_SE	SCO	Sweden
819	S-1	ISO8859-1	SE	46	sv	Sun	Sweden
819	S-1	ISO-8859-1	SE	46	sv_SE	Linux	Sweden
1252	S-1	1252	SE	46	-	WIN	Sweden
1275	S-1	1275	SE	46	-	Mac	Sweden
278	S-1	IBM-278	SE	46	-	HOST	Sweden
1143	S-1	IBM-1143	SE	46	-	HOST	Sweden
437	S-1	IBM-437	CH	41	-	OS2	Switzerland
850	S-1	IBM-850	CH	41	-	OS2	Switzerland
819	S-1	ISO8859-1	CH	41	de_CH	AIX	Switzerland
850	S-1	IBM-850	CH	41	De_CH	AIX	Switzerland
819	S-1	iso88591	CH	41	-	HP	Switzerland
1051	S-1	roman8	CH	41	-	HP	Switzerland
819	S-1	ISO8859-1	CH	41	de_CH	SCO	Switzerland
819	S-1	ISO8859-1	CH	41	fr_CH	SCO	Switzerland
819	S-1	ISO8859-1	CH	41	it_CH	SCO	Switzerland
819	S-1	ISO8859-1	CH	41	de_CH	Sun	Switzerland
819	S-1	ISO-8859-1	CH	41	de_CH	Linux	Switzerland
1252	S-1	1252	CH	41	-	WIN	Switzerland
1275	S-1	1275	CH	41	-	Mac	Switzerland
500	S-1	IBM-500	CH	41	-	HOST	Switzerland
1148	S-1	IBM-1148	CH	41	-	HOST	Switzerland
938	D-2	IBM-938	TW	88	-	OS2	Taiwan
948	D-2	IBM-948	TW	88	-	OS2	Taiwan
950	D-2	big5	TW	88	-	OS2	Taiwan
950	D-2	big5	TW	88	Zh_TW	AIX	Taiwan
964	D-2	IBM-eucTW	TW	88	zh_TW	AIX	Taiwan
950	D-2	big5	TW	88	zh_TW.big5	HP	Taiwan
964	D-2	eucTW	TW	88	zh_TW.eucTW	HP	Taiwan
950	D-2	big5	TW	88	big5	Sun	Taiwan
964	D-2	cns11643	TW	88	zh_TW	Sun	Taiwan
950	D-2	big5	TW	88	-	WIN	Taiwan
937	D-2	IBM-937	TW	88	-	HOST	Taiwan
874	S-20	TIS620-1	TH	66	-	OS2	Thailand
874	S-20	TIS620-1	TH	66	Th_TH	AIX	Thailand
874	S-20	tis620	TH	66	th_TH.tis620	HP	Thailand
874	S-20	TIS620-1	TH	66	-	WIN	Thailand
838	S-20	IBM-838	TH	66	-	HOST	Thailand

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
857	S-9	IBM-857	TR	90	-	OS2	Turkey
920	S-9	ISO8859-9	TR	90	tr_TR	AIX	Turkey
920	S-9	iso88599	TR	90	tr_TR.iso88599	HP	Turkey
920	S-9	ISO8859-9	TR	90	tr_TR.ISO8859-9	SCO	Turkey
920	S-9	ISO-8859-9	TR	90	tr_TR	Linux	Turkey
1254	S-9	1254	TR	90	-	WIN	Turkey
1281	S-9	1281	TR	90	-	Mac	Turkey
1026	S-9	IBM-1026	TR	90	-	HOST	Turkey
437	S-1	IBM-437	GB	44	-	OS2	U.K.
850	S-1	IBM-850	GB	44	-	OS2	U.K.
819	S-1	ISO8859-1	GB	44	en_GB	AIX	U.K.
850	S-1	IBM-850	GB	44	En_GB	AIX	U.K.
819	S-1	iso88591	GB	44	en_GB.iso88591	HP	U.K.
1051	S-1	roman8	GB	44	en_GB.roman8	HP	U.K.
819	S-1	ISO8859-1	GB	44	en_UK	Sun	U.K.
819	S-1	ISO8859-1	GB	44	en_GB	SCO	U.K.
819	S-1	ISO8859-1	GB	44	en	SCO	U.K.
819	S-1	ISO-8859-1	GB	44	en_GB	Linux	U.K.
1252	S-1	1252	GB	44	-	WIN	U.K.
1275	S-1	1275	GB	44	-	Mac	U.K.
285	S-1	IBM-285	GB	44	-	HOST	U.K.
1146	S-1	IBM-1146	GB	44	-	HOST	U.K.
819	S-1	88591	GB	44	En_GB.88591	SINIX	U.K.
819	S-1	ISO8859-1	GB	44	En_GB.6937	SINIX	U.K.
1125	S-5	IBM-1125	UA	380	-	OS2	Ukraine
1124	S-5	IBM-1124	UA	380	uk_UA	AIX	Ukraine
1251	S-5	1251	UA	380	-	WIN	Ukraine
1123	S-5	IBM-1123	UA	380	-	HOST	Ukraine
437	S-1	IBM-437	US	1	-	OS2	USA
850	S-1	IBM-850	US	1	-	OS2	USA
819	S-1	ISO8859-1	US	1	en_US	AIX	USA
850	S-1	IBM-850	US	1	En_US	AIX	USA
819	S-1	iso88591	US	1	en_US.iso88591	HP	USA
1051	S-1	roman8	US	1	en_US.roman8	HP	USA
819	S-1	ISO8859-1	US	1	en_US	Sun	USA
819	S-1	ISO8859-1	US	1	en_US	SGI	USA
819	S-1	ISO8859-1	US	1	en_US	SCO	USA
819	S-1	ISO-8859-1	US	1	en_US	Linux	USA
1252	S-1	1252	US	1	-	WIN	USA
1275	S-1	1275	US	1	-	Mac	USA
37	S-1	IBM-37	US	1	-	HOST	USA
1140	S-1	IBM-1140	US	1	-	HOST	USA
1163	S-11	IBM-1163	VN	84	-	OS2	Vietnam
1163	S-11	IBM-1163	VN	84	vi_VN	AIX	Vietnam
1258	S-11	1258	VN	84	-	WIN	Vietnam
1164	S-11	IBM-1164	VN	84	-	HOST	Vietnam

표 28. 지원 언어 및 코드 페이지 (계속)

코드 페이지	그룹	코드 세트	Tr.	국가/ 지역 코드	로케일	OS	국가/ 지역 이름
The following map to Arabic Countries (AA):							
					/* Arabic (Saudi Arabia) */		
					/* Arabic (Iraq) */		
					/* Arabic (Egypt) */		
					/* Arabic (Libya) */		
					/* Arabic (Algeria) */		
					/* Arabic (Morocco) */		
					/* Arabic (Tunisia) */		
					/* Arabic (Oman) */		
					/* Arabic (Yemen) */		
					/* Arabic (Syria) */		
					/* Arabic (Jordan) */		
					/* Arabic (Lebanon) */		
					/* Arabic (Kuwait) */		
					/* Arabic (United Arab Emirates) */		
					/* Arabic (Bahrain) */		
					/* Arabic (Qatar) */		
다음은 미국 영어(US)에 대응됩니다.							
					/* English (Jamaica) */		
					/* English (Caribbean) */		
다음은 라틴 아메리카(Lat)에 대응됩니다.							
					/* Spanish (Mexican) */		
					/* Spanish (Guatemala) */		
					/* Spanish (Costa Rica) */		
					/* Spanish (Panama) */		
					/* Spanish (Dominican Republic) */		
					/* Spanish (Venezuela) */		
					/* Spanish (Colombia) */		
					/* Spanish (Peru) */		
					/* Spanish (Argentina) */		
					/* Spanish (Ecuador) */		
					/* Spanish (Chile) */		
					/* Spanish (Uruguay) */		
					/* Spanish (Paraguay) */		
					/* Spanish (Bolivia) */		

주:

1. Solaris 코드 페이지 950은 다음 IBM 950의 문자를 지원하지 않습니다.

코드 범위	설명	**Sun Big-5**	**IBM Big-5**
C6A1-C8FE	기호	예약 영역	기호
F9D6-F9FE	ETen 확장	예약 영역	ETen 확장
F286-F9A0	IBM 선택 문자	예약 영역	IBM 선택

2. Euro 기호 지원은 이 버전의 DB2 UDB와 함께 제공됩니다. Microsoft Windows ANSI 코드 페이지는 Euro 기호를 0x80 위치에 포함시키기 위해 Microsoft의 최근 정의에 따라 수정되었습니다. 이 위치는 이전에 정의되지 않았습니다. 또한, 코드 페이지 850의 정의가 변경되어 문자 Dotless i(0xD5 위치에 있는)가 Euro 기호로 대체되었습니다. DB2 UDB는 이들 코드 페이지의 새로운 정의를 기본값으로 사용하여 Euro 기호 지원을 제공합니다. 이 구현은 Euro 기호 지원을 필요로 하는 현재 DB2 UDB에 대한 적절한 기본값으로, 다른 고객에게는 영향을 주지 않아야 합니다. 그러나, 계속해서 이들 코드 페이지의 이전 정의를 사용하려면, 다음 파일을 복사하면 됩니다.

   - 12520850.cnv
   - 08501252.cnv
   - IBM00850.ucs
   - IBM01252.ucs

   설치 완료 후에 위의 파일을

   ```
   sqllib/conv/alt/
   ```

   디렉토리에서 다음 디렉토리로 복사합니다.

   ```
   sqllib/conv/
   ```

   Euro 이외의 버전을 복사하기 전에 기존의 `IBM01252.usc` 및 `IBM00850.ucs` 파일을 백업할 수도 있습니다. 파일을 복사하고 나면, DB2 UDB로부터의 Euro 통화 기호 지원이 없어집니다.

DB2 관리 서버(DAS)를 위한 로케일 설정

DB2 관리 서버(DAS) 인스턴스 로케일이 DB2 인스턴스 로케일과 호환 가능한지 확인하십시오. 호환되지 않으면, DB2 인스턴스가 DB2 관리 서버(DAS)와 통신할 수 없습니다.

LANG 환경 변수가 DB2 관리 서버(DAS)의 사용자 프로파일에 설정되어 있지 않은 경우, DB2 관리 서버(DAS)는 기본 시스템 로케일로 시작됩니다. 기본 시스템 로케일이 정의되어 있지 않으면, DB2 관리 서버(DAS)는 코드 페이지 819로 시작합니다. DB2 인스턴스가 DBCS 로케일 중 하나를 사용하는 경우, DB2 관리 서버(DAS)는 코드 페이지 819로 시작하며 인스턴스는 DB2 관리 서버(DAS)와 통신할 수 없습니다. DB2 관리 서버(DAS) 로케일과 DB2 인스턴스 로케일은 호환 가능해야 합니다.

예를 들면, 중국어 Linux 시스템에서 LANG=zh_CN이 DB2 관리 서버(DAS)의 사용자 프로파일에 설정되어야 합니다.

UNIX 기반 플랫폼에 지원되는 변환 로케일

UNIX 기반 플랫폼에서 DB2 제품 메시지와 라이브러리를 서로 다른 언어로 설치할 수 있습니다. DB2 설치 유틸리티는 아래 테이블에서와 같이 메시지 카탈로그 파일 세트를 주어진 플랫폼에 가장 일반적으로 사용되는 로케일에 배치합니다. 274 페이지의 표29에는 AIX, HP-UX 및 Solaris용 정보가 제공됩니다. 275 페이지의 표30에는 Linux, Linux/390, SGI 및 Dynix용 정보가 제공됩니다.

시스템이 동일한 코드 페이지를 사용하지만 274 페이지의 표29 및 275 페이지의 표30에 제공된 로케일 이름과 다른 이름을 사용하는 경우, 해당 메시지 디렉토리로의 링크를 작성하면 변환된 메시지를 계속 볼 수 있습니다.

예를 들어, AIX 머신의 기본 로케일이 ja_JP.IBM-eucJP이고, ja_JP.IBM-eucJP의 코드 페이지가 954이면, 다음 명령을 발행하여 /usr/lpp/db2_07_01/msg/ja_JP.IBM-eucJP에서 /usr/lpp/db2_07_01/msg/ja_JP로 링크를 작성할 수 있습니다.

ln -s /usr/lpp/db2_07_01/msg/ja_JP /usr/lpp/db2_07_01/msg/ja_JP.IBM-eucJP

이 명령의 실행 후에는 모든 DB2 메시지가 일본어로 나타납니다.

표 29. AIX, HP-UX, Solaris를 위한 로케일 디렉토리

	AIX		**HP-UX**		**Solaris**	
언어	**로케일**	**코 드 페이지**	**로케일**	**코 드 페이지**	**로케일**	**코 드 페이지**
프랑스어	fr_FR Fr_FR	819 850	fr_FR.iso88591 fr_FR.roman8	819 1051	fr	819
독일어	de_DE De_DE	819 850	de_DE.iso88591 de_DE.roman8	819 1051	de	819
이탈리아어	it_IT It_IT	819 850	it_IT.iso88591 it_IT.roman8	819 1051	it	819
스페인어	es_ES Es_ES	819 850	es_ES.iso88591 es_ES.roman8	819 1051	es	819
브라질 포르투갈어	pt_BR	819			pt_BR	819
일본어	ja_JP Ja_JP	954 932	ja_JP.eucJP	954	ja	954
한국어	ko_KR	970	ko_KR.eucKR	970	ko	970
중국어	zh_CN Zh_ CN.GBK	1383 1386	zh_CN.hp15CN	1383	zh	1383
대만어	zh_TW Zh_TW	964 950	zh_TW.eucTW zh_TW.big5	964 950	zh_TW zh_TW.BIG5	964 950
덴마크어	da_DK Da_DK	819 850	da_DK.iso88591 da_DK.roman8	819 1051	da	819
핀란드어	fi_FI Fi_FI	819 850	fi_FI.iso88591 fi_FI.roman8	819 1051	fi	819
노르웨이어	no_NO No_NO	819 850	no_NO.iso88591 no_NO.roman8	819 1051	no	819
스웨덴어	sv_SE Sv_SE	819 850	sv_SE.iso88591 sv_SE.roman8	819 1051	sv	819
체코어	cs_CZ	912				

표 29. AIX, HP-UX, Solaris를 위한 로케일 디렉토리 (계속)

	AIX		HP-UX		Solaris	
언어	로케일	코 드 페이지	로케일	코 드 페이지	로케일	코 드 페이지
헝가리어	hu_HU	912				
폴란드어	pl_PL	912				
네덜란드어	nl_NL Nl_NL	819 850				
터키어	tr_TR	920				
러시아어	ru_RU	915				
불가리아어	bg_BG	915	bg_BG.iso88595	915		
슬로베니아어	sl_SI	912	sl_SI.iso88592	912	sl_SI	912

표 30. Linux, Linux/390, SGI, Dynix를 위한 로케일 디렉토리

	Linux		Linux/390		SGI		Dynix	
언어	로케일	코 드 페 이 지	로케일	코 드 페 이 지	로케일	코 드 페 이 지	로케일	코 드 페 이 지
프랑스어	fr	819	fr	819			fr	819
독일어	de	819	de	819			de	819
이탈리아어							es	819
스페인어								
브라질 포르투갈어								
일본어	ja_JP.ujis	954	ja_JP.ujis	954			ja_JP.EUC	954
한국어	ko	970	ko	970	ko_KO.euc	970		
중국어	zh zh_CN.GBK	1386	zh zh_CN.GBK	1386				
대만어	zh_TW.Big5	950	zh_TW.Big5	950				
덴마크어								
핀란드어								
노르웨이어								
스웨덴어								
체코어								
헝가리어								
폴란드어								

표 30. Linux, Linux/390, SGI, Dynix를 위한 로케일 디렉토리 (계속)

	Linux		Linux/390		SGI		Dynix	
언어	로케일	코드 페이지	로케일	코드 페이지	로케일	코드 페이지	로케일	코드 페이지
네덜란드어							nl	819
터키어								
러시아어								
불가리아어								
슬로베니아어								

제어 센터 및 문서 파일 세트

제어 센터, 제어 센터 도움말 및 문서 파일 세트는 목표 워크스테이션의 다음 디렉토리에 있습니다.

- AIX용 DB2:
  - /usr/lpp/db2_07_01/cc/%L
  - /usr/lpp/db2_07_01/java/%L
  - /usr/lpp/db2_07_01/doc/%L
  - /usr/lpp/db2_07_01/qp/%L
  - /usr/lpp/db2_07_01/spb/%L
- HP-UX용 DB2:
  - /opt/IBMdb2/V7.1/cc/%L
  - /opt/IBMdb2/V7.1/java/%L
  - /opt/IBMdb2/V7.1/doc/%L
- Linux용 DB2:
  - /usr/IBMdb2/V7.1/cc/%L
  - /usr/IBMdb2/V7.1/java/%L
  - /usr/IBMdb2/V7.1/doc/%L
- Solaris용 DB2:
  - /opt/IBMdb2/V7.1/cc/%L

- /usr/IBMdb2/V7.1/java/%L
- /opt/IBMdb2/V7.1/doc/%L

제어 센터 파일 세트는 유니코드 코드 페이지에 있습니다. 문서 및 제어 센터 도움말 파일 세트는 브라우저가 인식하는 코드 세트에 있습니다. 시스템이 제공된 로케일 이름과 다른 이름을 사용하는 경우, 해당 언어 디렉토리로의 링크를 작성하면 제어 센터 변환 버전을 계속 실행하여 변환된 도움말 버전을 볼 수 있습니다.

예를 들어, AIX 머신 기본 로케일이 ja_JP.IBM-eucJP이면 다음 명령을 발행하여 /usr/lpp/db2_07_01/cc/ja_JP.IBM-eucJP에서 /usr/lpp/db2_07_01/cc/ja_JP로, /usr/lpp/db2_07_01/doc/ja_JP.IBM-eucJP에서 /usr/lpp/db2_07_01/doc/ja_JP로의 링크를 작성할 수 있습니다.

- **ln -s /usr/lpp/db2_07_01/cc/ja_JP /usr/lpp/db2_07_01/cc/ja_JP.IBM-eucJP**
- **ln -s /usr/lpp/db2_07_01/doc/ja_JP /usr/lpp/db2_07_01/doc/ja_JP.IBM-eucJP**

이 명령을 실행한 이후에 제어 센터 및 도움말은 일본어로 나타납니다.

주: Linux/390 또는 NUMA-Q에서 실행되는 경우에는 웹 제어 센터가 지원되지 않습니다. 클라이언트 워크스테이션에서 웹 제어 센터를 사용하여 이들 플랫폼에서 데이터베이스를 관리할 수 있습니다.

코드 페이지 값 도출

데이터베이스에 연결이 생길 때, *응용프로그램 코드 페이지*를 사용 중인 환경에서 얻을 수 있습니다. DB2CODEPAGE 레지스트리 변수가 설정된 경우, 해당 값은 응용프로그램 코드 페이지로 간주됩니다. DB2가 운영 체제로부터 코드 페이지 정보를 자동으로 도출하기 때문에 일반적으로 DB2CODEPAGE 레지스트리 변수는 설정하지 않아도 됩니다. DB2CODEPAGE 레지스트리 변수를 틀린 값으로 설정하면 예기치 않은 결과가 야기될 수 있습니다.

응용프로그램이 CLI 인터페이스를 사용하도록 코드화된 경우, CLI 코드 계층은 일부 경우 로케일 설정을 사용합니다(사용자가 DB2CODEPAGE 레지스트리 변수를 설정한 경우에도 해당).

*데이터베이스 코드 페이지*를 데이터베이스가 작성될 때 지정된 값(명시적으로 또는 기본적으로)에서 얻을 수 있습니다. 예를 들어, 다음은 *사용 중인 환경*이 다른 운영 환경에서 결정되는 방법을 설명합니다.

UNIX UNIX 기반 환경 시스템에서, 사용 중인 환경은 언어, 지역 및 코드 세트에 대한 정보까지 포함하여 로케일 설정값에서 결정됩니다.

OS/2 OS/2에서, 1차 및 2차 코드 페이지는 CONFIG.SYS 파일에서 지정됩니다. **chcp** 명령을 사용하여 코드 페이지를 표시하고, 주어진 세션 내에서 동적으로 변경할 수 있습니다.

Macintosh Macintosh 운영 체제의 경우, DB2CODEPAGE 레지스트리 변수가 설정되지 않은 경우, Macintosh 코드 페이지를 설치된 스크립트의 Regional 버전 코드에서 얻을 수 있습니다.

Windows 모든 Windows 32비트 운영 체제의 경우, DB2CODEPAGE 레지스트리 변수가 설정되지 않으면 코드 페이지는 레지스트리의 ANSI 코드 페이지 설정값에서 얻을 수 있습니다.

코드 페이지 값에 대한 환경 맵의 완전한 목록은 258 페이지의 표28에서 자세한 내용을 참조하십시오.

문자 세트

데이터베이스 관리 프로그램은 일반적으로 응용프로그램에 사용 가능한 문자 세트를 제한하지 않습니다. *응용프로그램 개발 안내서*에서 DB2가 지원하는 멀티 바이트 문자 세트(MBCS)에 대한 자세한 내용을 참조하십시오.

식별자용 문자 세트

데이터베이스 이름에서 사용될 수 있는 기본 문자 세트는 1바이트 대소문자 라틴 문자(A...Z, a...z), 아라비아 숫자(0...9) 및 밑줄 문자(_)로 구성됩니다. 호스트 데이터베이스 제품과의 호환성을 제공하기 위해 세 개의 특수 문자(#, @ 및 $)가 이

목록에 추가되었습니다. 특수 문자 #, @ 및 $는 NLS 호스트(EBCDIC) 불변 문자 세트에는 포함되지 않기 때문에 NLS 환경에서는 주의하여 사용하십시오. 사용 중인 코드 페이지에 따라 확장 문자 세트의 문자를 사용할 수도 있습니다. 다중 코드 페이지 환경에서 데이터베이스를 사용할 경우, 모든 코드 페이지가 확장 문자 세트에서 사용하려고 하는 어떠한 영문자라도 지원한다는 것을 확인해야 합니다.

데이터베이스 오브젝트(테이블 및 뷰)를 이름 지정할 때 프로그램 레이블, 호스트 변수, 커서 및 확장 문자 세트의 요소(예를 들어, 구별할 수 있는 표시가 있는 문자)를 사용할 수 있습니다. 정확히 어떤 문자가 사용 가능한지는 사용 중인 코드 페이지에 의해 좌우됩니다. 다중 코드 페이지 환경에서 데이터베이스를 사용할 경우, 모든 코드 페이지가 확장 문자 세트에서 사용하려고 하는 어떠한 영문자라도 지원한다는 것을 확인해야 합니다. *SQL 참조서*에서 SQL문에서 사용될 수 있는 확장 문자 이외의 문자를 가진 분리 식별자에 대한 자세한 내용을 참조하십시오.

DBCS 식별자용 확장 문자 세트 정의

DBCS 환경에서, 확장 문자 세트는 기본 문자 세트의 모든 문자로 구성되어 있고, 다음도 추가됩니다.

- 2바이트 공간을 제외한 DBCS 코드 페이지의 모든 2바이트 문자가 유효합니다.
- 2바이트 공간은 특수 문자입니다.
- 혼합 코드 페이지에서 사용 가능한 1바이트 문자. 다음과 같이 다양한 범주에 할당됩니다.

범주	복합 코드 페이지 내의 유효한 코드 포인트
숫자	x30-39
문자	x23-24, x40-5A, x61-7A, xA6-DF(코드 페이지 932 및 942의 경우 A6-DF)
특수 문자	다른 모든 유효한 1바이트 문자 코드 포인트

코딩 SQL문

SQL문의 코딩은 언어에 종속되지 않습니다. SQL 키워드는 대문자, 소문자 또는 혼합 문자로 인쇄될 수 있습니다. SQL문에서의 프로그램 레이블과 마찬가지로 데이터베이스 오브젝트 및 호스트 변수의 이름은 『DBCS 식별자용 확장 문자 세트 정의』에 설명된 확장 문자 세트 외의 문자를 포함할 수 없습니다.

양방향 CCSID 지원

다음의 BiDi 속성은 서로 다른 플랫폼에서의 양방향 데이터에 대한 올바른 처리에 필요합니다.

```
- Text type (LOGICAL or VISUAL)
- Shaping (SHAPED or UNSHAPED)
- Orientation (RIGHT-TO-LEFT or LEFT-TO-RIGHT)
- Numeral shape (ARABIC or HINDI)
- Symmetric swapping (YES or NO)
```

서로 다른 플랫폼에서의 기본값은 같지 않으므로, DB2 데이터가 플랫폼 사이에 이동될 때 문제점이 발생합니다. 예를 들어, Windows 운영 체제는 LOGICAL UNSHAPED 데이터를 사용하며, OS/390은 보통 SHAPED VISUAL 데이터를 사용합니다. 그러므로, 양방향 속성에 대한 지원이 없으면 OS/390용 DB2 Universal Database로부터 Windows 32비트 운영 체제에서 DB2 UDB로의 데이터 송신은 올바르지 않게 표시될 수 있습니다.

양방향 특정 CCSID

DB2는 특수 양방향 코드화 문자 세트 식별자(CCSID)를 통한 양방향 데이터 속성을 지원합니다. 다음과 같은 양방향 CCSI가 정의되어 DB2 UDB와 함께 구현되었습니다.

CCSID (dec)	CCSID (hex)	Code Page	String Type
00420	x'01A4'	420	4
00424	x'01A8'	424	4
08612	x'21A4'	420	5
08616	x'21A8'	424	10
00856	x'0358'	856	5
00862	x'035E'	862	4
00864	x'0360'	864	5
00916	x'0394'	916	5
01046	x'0416'	1046	5
01089	x'0441'	1089	5
01255	x'04E7'	1255	5
01256	x'04E8'	1256	5
62208	x'F300'	856	4
62209	x'F301'	862	10
62210	x'F302'	916	4
62211	x'F303'	424	5
62213	x'F305'	862	5
62215	x'F307'	1255	4

```
62218   x'F30A'   864      4
62220   x'F30C'   856      6
62221   x'F30D'   862      6
62222   x'F30E'   916      6
62223   x'F30F'  1255      6
62224   x'F310'   420      6
62225   x'F311'   864      6
62226   x'F312'  1046      6
62227   x'F313'  1089      6
62228   x'F314'  1256      6
62229   x'F315'   424      8
62230   x'F316'   856      8
62231   x'F317'   862      8
62232   x'F318'   916      8
62233   x'F319'   420      8
62234   x'F31A'   420      9
62235   x'F31B'   424      6
62236   x'F31C'   856     10
62237   x'F31D'  1255      8
62238   x'F31E'   916     10
62239   x'F31F'  1255     10
62240   x'F320'   424     11
62241   x'F321'   856     11
62242   x'F322'   862     11
62243   x'F323'   916     11
62244   x'F324'  1255     11
62245   x'F325'   424     10
62246   x'F326'  1046      8
62247   x'F327'  1046      9
62248   x'F328'  1046      4
62249   x'F329'  1046     12
62250   x'F32A'   420     12
```

여기서, CDRA 문자열 유형이 다음과 같이 정의됩니다.

String Type	Text Type	Numerical Shape	Orientation	Shaping	Symmetrical Swapping
4	Visual	Passthru	LTR	Shaped	OFF
5	Implicit	Arabic	LTR	Unshaped	ON
6	Implicit	Arabic	RTL	Unshaped	ON
7(*)	Visual	Arabic	Contextual(*)	Unshaped-Lig	OFF
8	Visual	Arabic	RTL	Shaped	OFF
9	Visual	Passthru	RTL	Shaped	ON
10	Implicit	Passthru	Contextual-L	Unshaped	ON
11	Implicit	Passthru	Contextual-R	Unshaped	ON
12	Implicit	Arabic	RTL	Shaped	ON

주: (*) 필드 방향은 첫 번째 영문자가 라틴 문자일 때 왼쪽에서 오른쪽으로(LTR) 이고, 양방향(RTL) 문자일 때 오른쪽에서 왼쪽으로(RTL)입니다. 문자 형태가 없지만(unshaped), LamAlef 이음이 보존되고 구성 요소로 분리되지 않습니다.

양방향 지원의 DB2 Universal Database 구현

양방향 배치 변환은 새로운 CCSID 정의를 사용하여 DB2 Universal Database에서 구현됩니다. 새로운 BiDi 고유의 CCSID의 경우, 배치 변환은 코드 페이지 변환 대신 또는 변환에 추가로 수행됩니다. 이 지원을 사용하려면, DB2BIDI 레지스트리 변수를 YES로 설정해야 합니다. 기본값은 이 변수를 설정하지 않는 것입니다. 이 변수는 모든 변환에 대해 서버에 의해 사용되며, 서버가 시작될 때에만 설정할 수 있습니다. DB2BIDI를 YES로 설정하면, 추가 점검 및 배치 변환으로 성능에 약간 영향을 줄 수도 있습니다.

DRDA 이외의 환경에서 특정 양방향 CCSID를 지정하려면, 위의 표에서 클라이언트 특성이 일치하는 CCSID를 선택하고 DB2CODEPAGE를 해당되는 값에 설정하십시오. 이미 데이터베이스에 연결되어 있으면, TERMINATE 명령을 발행하여 다시 연결함으로서 DB2CODEPAGE의 새로운 설정값이 적용되도록 해야 합니다. 클라이언트 플랫폼의 문자열 유형 또는 코드 페이지에 대해 올바르지 않은 CCSID를 선택하면, 예기치 않은 결과가 야기될 수 있습니다. 호환되지 않는 CCSID(예: 아라비아 문자 데이터베이스에 대한 연결 또는 반대 방향으로의 연결을 위한 Hebrew CCSID)를 선택하거나, 서버에 대해 DB2BIDI를 설정하지 않은 경우, 연결하려고 하면 오류 메세지가 수신됩니다.

DRDA 환경에서, HOST EBCDIC 플랫폼은 이러한 양방향 CCSID도 지원할 경우, DB2CODEPAGE 값을 설정해야 합니다. 그러나, HOST 플랫폼에서 이러한 CCSID가 지원되지 않으면, 연결되어 있는 HOST 데이터베이스 서버에 대해 CCSID 겹쳐쓰기를 지정해야 합니다. 이것은 DRDA 환경에서 코드 페이지 변환과 배치 변환이 데이터 수신기에 의해 수행되기 때문에 반드시 필요합니다. 그러나, HOST 서버가 이러한 양방향 CCSID를 지원하지 않으면, DB2 UDB로부터 수신하는 데이터에 대해 배치 변환을 수행하지 않습니다. CCSID 겹쳐쓰기를 사용할 경우, DB2 UDB 클라이언트는 아웃바운드 데이터에 대해서도 배치 변환을 수행합니다. *DB2 Connect 사용자 안내서*에서 CCSID 겹쳐쓰기 설정에 대한 자세한 내용을 참조하십시오.

CCSID 겹쳐쓰기는 HOST EBCDIC 플랫폼이 클라이언트이고 DB2 UDB가 서버일 경우에는 지원되지 않습니다.

양방향 지원의 DB2 Connect 구현

DB2 Connect와 서버의 데이터베이스간에 데이터가 교환될 때 수신 데이터에서 변환을 수행하는 것은 주로 수신기입니다. 일반 코드 페이지 변환 외에 양방향 배치 변환에도 동일한 규칙이 일반적으로 적용됩니다. DB2 Connect에는 서버 데이터베이스에서 수신한 데이터 외에도 서버 데이터베이스로 보내려고 하는 데이터에 대해 양방향 배치 변환을 수행할 수 있는 선택적 기능이 있습니다.

DB2 Connect가 서버 데이터베이스로 출력하는 데이터에 대해 양방향 배치 변환을 수행하기 위해, 서버 데이터베이스의 양방향 CCSID가 대체되어야 합니다. 이것은 서버 데이터베이스에 대한 DCS 데이터베이스 디렉토리 항목의 PARMS 필드에 있는 BIDI 매개변수를 사용하여 수행할 수 있습니다.

주: DB2 Connect가 DB2 호스트 데이터베이스에 송신하려고 하는 데이터에 대해 배치 변환을 수행하려 할 경우, CCSID를 겹쳐쓸 필요가 없더라도 여전히 DCS 데이터베이스 디렉토리 PARMS 필드에 BIDI 매개변수를 추가해야 합니다. 이러한 경우, 사용자가 제공해야 하는 CCSID는 기본 DB2 호스트 데이터베이스 CCSID입니다.

BIDI 매개변수는 기본 서버 데이터베이스 양방향 CCSID를 겹쳐쓸 양방향 CCSID와 함께 PARMS 필드에서 9번째 매개변수로 지정됩니다.

```
",,,,,,,,BIDI=xyz"
```

여기서, *xyz*는 CCSID 겹쳐쓰기입니다.

주: BIDI 매개변수를 유효하게 하기 위해 레지스트리 변수 DB2BIDI를 YES로 설정해야 합니다.

지원되는 양방향 CCSID는 해당 문자열 유형과 함께 280 페이지의 『양방향 특정 CCSID』에 있습니다.

이 기능의 사용은 예와 함께 설명됩니다.

CCSID 62213(양방향 문자열 유형 5)을 수행하는 Hebrew DB2 클라이언트를 가지고 있으며, CCSID 00424(양방향 문자열 유형 4)를 수행하는 DB2 호스트 데이터베이스에 액세스하려고 한다고 가정하십시오. 그러나, DB2 호스트 데이터베이스에 있는 데이터는 CCSID 08616(양방향 문자열 유형 6)에 근거합니다.

여기에서는 두 가지 문제점이 있습니다. 첫 번째 문제점은 DB2 호스트 데이터베이스가 CCSID 00424와 08616의 양방향 문자열 유형에서 차이점을 알지 못한다는 것입니다. 두 번째 문제점은 DB2 호스트 데이터베이스가 DB2 클라이언트 CCSID(62213)를 인식하지 않는다는 것입니다. CCSID 62213과 같은 코드 페이지를 기준으로 하는 CCSID 00862만 지원합니다.

DB2 호스트 데이터베이스에 송신된 데이터가 반드시 양방향 문자열 유형 6 형식으로 시작하도록 해야 하며, 또한 DB2 Connect가 DB2 호스트 데이터베이스에서 수신하는 데이터에 대해 양방향 변환을 수행해야 하는 것을 알게 해야 합니다. DB2 호스트 데이터베이스에 대해 다음과 같은 카탈로그 명령을 사용해야 합니다.

```
db2 catalog dcs database nydb1 as telaviv parms ",,,,,,,,BIDI=08616"
```

이 명령은 DB2 호스트 데이터베이스 CCSID 00424를 08616으로 겹쳐쓰도록 DB2 Connect에 지시합니다. 이를 겹쳐쓸 때 다음과 같은 처리가 수반됩니다.

1. DB2 Connect가 CCSID 00862를 사용하여 DB2 호스트 데이터베이스에 연결합니다.
2. DB2 Connect가 DB2 호스트 데이터베이스에 송신하는 데이터에 대해 양방향 배치 변환을 수행합니다. CCSID 62213(양방향 문자열 유형 5)에서 CCSID 62221(양방향 문자열 유형 6)로 변환합니다.
3. DB2 Connect가 DB2 호스트 데이터베이스에서 수신하는 데이터에 대해 양방향 배치 변환을 수행합니다. CCSID 08616(양방향 문자열 유형 6)에서 CCSID 62213(양방향 문자열 유형 5)으로 변환합니다.

주: 어떤 경우에는 양방향 CCSID를 사용하면 SQL 조회가 수정되어 DB2 서버에 의해 인식되지 않게 될 수도 있습니다. 특히, 다른 문자열 유형을 사용할 수 있을 때 IMPLICIT CONTEXTUAL 및 IMPLICIT RIGHT-TO-LEFT CCSID 사용을 피해야 합니다. CONTEXTUAL CCSID는 SQL 조회에 따

옴표로 표시된 문자열이 들어 있을 경우 예상치 않은 결과가 발생할 수 있습니다. SQL문에 따옴표로 표시되는 문자열의 사용을 피하고 가능하면 호스트 변수를 대신 사용하십시오.

특정 양방향 CCSID가 이러한 권장사항으로 수정될 수 없는 문제점을 일으키고 있는 경우, DB2BIDI를 NO로 설정해야 합니다.

조합 순서

데이터베이스 관리 프로그램은 *조합 순서*를 사용하여 문자 데이터를 비교합니다. 즉, 특정 문자를 다른 문자보다 높게, 낮게 또는 같게 배열해야 하는지 결정하는 문자 세트의 순서를 말합니다. 예를 들어, 조합 순서를 특정 문자의 대소문자를 동등하게 배열할 때 사용합니다.

데이터베이스에 대한 조합 순서는 데이터베이스 작성시 지정되며, 나중에 수정될 수 없습니다.

데이터베이스 관리 프로그램은 API를 사용하여 데이터베이스를 사용자 조합 순서로 작성되게 합니다. *응용프로그램 개발 안내서*에서 사용자 정의 조합 순서 테이블 구현에 대한 자세한 내용을 참조하십시오.

주: 2진 데이터용 속성 및 BLOB 데이터로 정의된 문자열은 2진 정렬 순서로 정렬됩니다.

일반 관심사항

일단 조합 순서가 정의되면, 해당 데이터베이스에 대한 다음의 모든 문자 비교는 조합 순서대로 수행됩니다. 2진 데이터용 또는 BLOB 데이터로서 정의된 문자 데이터를 제외하고, 조합 순서는 SQL 비교 및 ORDER BY절용으로 사용되고 색인 및 통계 설정시에도 사용됩니다. *SQL 참조서*에서 "문자열 비교"를 참조하여 데이터베이스 조합 순서가 사용되는 방법에 대한 자세한 내용을 참조하십시오.

다음은 문제점이 발생할 수 있는 경우입니다.

- 응용프로그램은 다른 조합 순서를 사용하여 정렬된 응용프로그램의 데이터와 데이터베이스에서 정렬된 데이터를 병합합니다.

- 응용프로그램이 하나의 데이터베이스에서 정렬된 데이터와 다른 데이터베이스에서 정렬된 데이터를 병합하지만, 데이터베이스는 서로 다른 조합 순서를 갖습니다.
- 응용프로그램은 정렬된 데이터가 적절한 조합 순서로 되어 있지 않다고 가정합니다. 예를 들어, 영문자보다 낮게 배열된 수는 특정 조합 순서에 맞을 수도 틀릴 수도 있습니다.

마지막으로 기억해야 할 점은 문자 코드 포인트를 직접 비교하여 정렬된 결과는 식별 조합 순서를 사용하여 지시된 조회의 결과와 일치한다는 것입니다.

연합 데이터베이스 관심사항

데이터베이스 조합 순서를 선택하면 연합 시스템 성능에 영향을 줄 수 있습니다. 데이터 소스가 DB2 연합 데이터베이스와 동일한 조합 순서를 사용할 경우, DB2는 데이터 소스에 대한 문자 데이터와 관련하여 명령 종속 처리를 푸시다운할 수 있습니다. 데이터 소스 조합 순서가 DB2 조합 순서와 일치하지 않을 경우, 데이터가 검색되고 문자 데이터에 대한 모든 명령 종속 처리가 지역적으로 수행됩니다. (성능을 저하시킬 수 있습니다.)

데이터 소스 및 DB2가 동일한 조합 순서를 갖는지 판별하려면, 다음 사항을 고려하십시오.

- 자국어 지원.

  조합 순서는 서버에 지원되는 언어와 관련됩니다. DB2 NLS 정보를 데이터 소스 NLS 정보와 비교해 보십시오.

- 데이터 소스 특성.

  일부 데이터 소스는 대소문자를 구별하지 않는 조합 순서를 사용하여 작성되며, 이는 명령 종속 조작에서 DB2로부터 다른 결과를 낼 수 있습니다.

- 사용자 정의.

  일부 데이터 소스는 조합 순서에 대해 여러 옵션을 제공하거나 조합 순서를 사용자 정의할 수 있습니다.

DB2 연합 데이터베이스에서 액세스할 데이터 소스 혼합에 따라 DB2 연합 데이터베이스에 대한 조합 순서를 선택하십시오. 예를 들면, 다음과 같습니다.

- DB2 데이터베이스가 DB2와 같은 코드 페이지(NLS)를 사용하여 Oracle 데이터베이스를 가장 많이 액세스할 경우, 데이터베이스 작성시 식별 순서를 지정하십시오. (Oracle 데이터베이스는 동일한 조합 순서를 사용합니다.)
- DB2 데이터베이스가 DB2 UDB 데이터베이스만 액세스할 경우에는 조합 순서 값을 일치시키는지 확인하십시오.

sqlecrea - 데이터베이스 작성 API의 설명 아래에 있는 예에 대해서는 *Administrative API Reference*를 참조하여 MVS 조합 순서 설정에 대한 자세한 내용을 얻으십시오. 이들 예에는 EBCIDIC 500, 37 및 5026/5035 코드 페이지에 대한 배열 테이블이 포함되어 있습니다.

DB2 데이터베이스에 대한 조합 순서를 설정한 다음에는 각 데이터 소스 서버에 대해 *collating_sequence* 서버 옵션을 설정했는지 확인하십시오. 이 옵션은 지정된 데이터 소스 서버의 조합 순서가 DB2 데이터베이스의 조합 순서와 일치하는지를 나타냅니다.

조합 순서가 일치하는 경우 collating_sequence 옵션을 ″Y″로 설정하십시오. 이러한 설정은 DB2 최적화 알고리즘이 데이터 소스에서 성능을 향상시킬 수 있는 명령 종속 처리를 고려할 수 있게 합니다. 단, 데이터 소스 조합 순서가 DB2 데이터베이스 조합 순서와 같지 않을 경우, 부정확한 결과를 받을 수 있습니다. 예를 들어, 사용자의 플랜이 병합 조인을 사용할 때 DB2 최적화 알고리즘은 가능한 한 많은 데이터 소스에 대해 조작 명령을 푸시다운합니다. 데이터 소스 조합 순서가 동일하지 않으면, 조인 결과는 정확한 결과 집합을 갖게 되지 않을 수 있습니다.

조합 순서가 일치하지 않으면 collating_sequence 옵션을 ″N″으로 설정하십시오. 데이터 소스 조합 순서가 DB2와 다르거나 데이터 소스 배열 조작이 대소문자를 구분하지 않을 때 이 값을 사용하십시오. 예를 들어, 영문 코드 페이지를 사용하는 대소문자 비 구분 데이터 소스에서 TOLLESON, ToLLeSoN 및 tolleson은 모두 동일하게 간주됩니다. 데이터 소스의 조합 순서가 DB2 조합 순서와 동일하다고 확신하지 않으면 collating_sequence 옵션을 ″N″으로 설정하십시오.

타이 문자 정렬

특수 모음(″선행 모음″), 음조 표시 및 기타 특수 타이 문자를 올바로 정렬하려는 경우, CREATE DATABASE 명령에서 COLLATE USING NLSCHAR 절을 사용하여 데이터베이스를 작성해야 합니다. 구문은 *Command Reference*를 참조하십시오.

날짜 시간 값

날짜 시간 데이터 유형은 아래에 설명되어 있습니다. 날짜 시간 값을 산술 및 문자 연산에 사용할 수도 있고 어떤 문자열과는 호환되기도 하지만, 날짜 값은 문자열 또는 수가 아닙니다.

날짜

*날짜*는 세 부분의 값(년, 월, 일)으로 되어 있습니다. 연도의 범위는 0001에서 9999까지입니다. 월의 범위는 1에서 12까지입니다. 일의 범위는 1에서 월이 속한 *x*까지로 됩니다.

날짜의 내부 표현은 4바이트 문자열로 되어 있습니다. 각각의 바이트는 두 개의 압축된 10진수로 구성되어 있습니다. 처음 2바이트는 연도, 세 번째 바이트는 월, 네 번째 바이트는 일을 표시합니다.

SQLDA에서 설명한 대로 DATE 컬럼은 10바이트로, 값을 표현하는 문자열에 적당한 길이입니다.

시간

*시간*은 24시간으로 정해진 세 부분의 값(시, 분, 초)으로 되어 있습니다. 시 부분의 범위는 0에서 24까지입니다. 기타 부분의 범위는 0에서 59까지입니다. 시가 24일 경우, 분과 초 스펙은 0이 됩니다.

시간의 내부 표현은 3바이트 문자열로 되어 있습니다. 각각의 바이트는 두 개의 압축된 10진수로 구성되어 있습니다. 첫 번째 바이트는 시, 두 번째 바이트는 분, 마지막 바이트는 초를 표시합니다.

SQLDA에서 설명한 대로 TIME 컬럼은 8바이트로, 값을 표현하는 문자열에 적당한 길이입니다.

시간소인

*시간소인*은 시간이 마이크로초 스펙을 포함할 때를 제외하고는, 위에서 정의된 것처럼 날짜 및 시간을 지시하는 7부분의 값(연, 월, 일, 시, 분, 초 및 마이크로초)으로 되어 있습니다.

시간소인의 내부 표현은 3바이트 문자열로 되어 있습니다. 각각의 바이트는 두 개의 압축된 10진수로 구성되어 있습니다. 처음 4바이트는 일, 다음 3바이트는 시간, 마지막 3바이트는 마이크로초를 표시합니다.

SQLDA에서 설명한 대로 TIMESTAMP 컬럼은 26바이트로, 값을 표현하는 문자열에 적당한 길이입니다.

날짜 시간 값의 문자열 표현

데이터 유형이 DATE, TIME 또는 TIMESTAMP인 값은 SQL 사용자에게 잘 보이는 내부 형식으로 표현됩니다. 날짜, 시간, 시간소인은 문자열로 표현되기도 하지만, 이런 표현은 DATE, TIME 또는 TIMESTAMP인 데이터 유형을 갖는 상수 또는 변수가 없기 때문에 직접적으로 SQL 사용자를 고려하게 됩니다. 그러므로, 날짜 시간 값은 검색을 위해 문자열 변수에 반드시 할당되야 합니다. 프로그램을 사전 처리 컴파일하거나 데이터베이스에 바인드할 때, ″F″ 형식 옵션을 스펙하여 겹쳐쓰지 않는다면, 문자열 표현은 클라이언트 국가/지역 코드와 관련된 날짜 시간 값의 정상적인 기본 형식이 됩니다. 292 페이지의 표33에서 다양한 국가/지역 코드에 대한 문자열 형식의 목록에 대해 참조하십시오.

날짜 시간 값의 유효한 문자열 표현을 내부 날짜 시간 값을 가지고 조작할 때에는, 조작이 수행되기 전 문자열 표현을 날짜, 시간 또는 시간소인의 내부 형식으로 전환시켜야 합니다. 날짜 시간 값의 유효한 문자열 표현은 다음 절에 정의되어 있습니다.

날짜 문자열

날짜의 문자열 표현은 수로 시작하는 문자열로 최소 8자 길이를 갖습니다. 뒤 공백이 포함될 수 있으며, 월과 일 부분에서는 선행 0이 생략되기도 합니다.

날짜에 대한 유효한 문자열 형식이 290 페이지의 표31에 나열되어 있습니다. 각 형식은 이름으로 식별하고 해당 약어 및 사용 예가 들어 있습니다.

표 31. 날짜의 문자열 표현에 대한 형식

형식명	약어	날짜 형식	예
국제 표준화 기구	ISO	yyyy-mm-dd	1991-10-27
IBM USA 표준	USA	mm/dd/yyyy	10/27/1991
IBM 유럽 표준	EUR	dd.mm.yyyy	27.10.1991
일본 산업 표준 서력 기원	JIS	yyyy-mm-dd	1991-10-27
장소 정의(지역)	LOC	데이터베이스 국가/지역 코드에 따라 다름	--

시간 문자열

시간의 문자열 표현은 수로 시작하는 문자열로 최소 4자 길이를 갖습니다. 뒤가 공백이 될 수도 있으며, 시간과 초 부분에서 0으로 시작되면 0이 생략되기도 합니다. 초를 생략할 경우, 내재적인 0초 스펙인 것으로 간주합니다. 그래서, 13.30은 13.30.00과 같습니다.

시간에 대한 유효한 문자열 형식이 표32에 나열되어 있습니다. 각 형식은 이름으로 식별하고 해당 약어 및 사용 예가 들어 있습니다.

표 32. 시간의 문자열 표현에 대한 형식

형식명	약어	시간 형식	예
국제 표준회 기구	ISO	hh.mm.ss	13.30.05
IBM USA 표준	USA	hh:mm AM 또는 PM	1:30 PM
IBM 유럽 표준	EUR	hh.mm.ss	13.30.05
일본 산업 표준 서력 기원	JIS	hh:mm:ss	13:30:05
장소 정의(지역)	LOC	응용프로그램 국가/지역 코드에 따라 다름	--

주:

1. ISO, EUR 또는 JIS 형식에서 .ss(또는 :ss)는 선택적입니다.
2. USA 시간 문자열 형식에서, 내재적인 00분 스펙을 지시하는 분 스펙은 생략될 수도 있습니다. 따라서, 1 PM은 1:00 PM과 같습니다.

3. USA 시간 문자열 형식에서, 시간 스펙은 12를 초과할 수 없으며, 00:00 AM의 특수한 경우를 제외하고 0이 될 수 없습니다. 24시간제의 ISO 형식을 사용한 USA 형식과 24시간제 사이의 대응은 다음과 같습니다.
   - 12:01 AM에서 12:59 AM은 00.01.00에서 00.59.00까지와 상응합니다.
   - 01:00 AM에서 11:59 AM은 01.00.00에서 11.59.00까지와 상응합니다.
   - 12:00 PM(정오)에서 11:59 PM은 12.00.00에서 23.59.00까지와 상응합니다.
   - 12:00 AM(자정)은 24.00.00과 상응하며, 00:00 AM(자정)은 00.00.00과 상응합니다.

시간소인 문자열

시간소인의 문자열 표현은 수로 시작하는 문자열로 최소 16자 길이를 갖습니다. 시간소인의 완전한 문자열 표현은 *yyyy-mm-dd-hh.mm.ss.nnnnnn* 형식입니다. 뒤 공백이 포함될 수 있으며, 선행 0은 시간소인의 월, 일 또는 시 부분에서 생략될 수 있으며, 마이크로초는 절단되거나 완전히 생략될 수 있습니다. 마이크로초 부분의 임의 자리를 생략하도록 선택한 경우, 내재적인 0 스펙이 가정됩니다. 그래서, 1991-3-2-8.30.00는 1991-03-02-08.30.00.000000에 대응합니다.

문자 세트 고려사항

날짜와 시간소인 문자열에는 1바이트 문자와 수만 들어 있어야 합니다.

날짜 및 시간 형식

날짜와 시간 형식의 문자열 표현은 응용프로그램의 국가/지역 코드 연관 날짜의 기본 형식입니다. 프로그램을 사전 처리 컴파일하거나 데이터베이스에 바인드할 때, ″F″ 형식 옵션을 지정하여 이 기본 형식을 겹쳐쓸 수 있습니다.

다음은 날짜 및 시간의 입출력 형식에 대한 설명입니다.

- 입력 시간 형식
  - 기본 입력 시간 형식이 없습니다.
  - 국가/지역 코드를 입력해 모든 시간 형식을 할 수 있습니다.
- 출력 시간 형식
  - 기본 출력 시간 형식은 지역 시간 형식과 같습니다.
- 입력 날짜 형식

- 기본 입력 날짜 형식이 없습니다.
- 데이터의 지역 형식이 ISO, JIS, EUR 또는 USA 날짜 형식과 충돌하는 곳에서, 지역 형식은 날짜 입력으로 간주됩니다. 예를 들어, 표33의 영국 항목을 참조하십시오.

• 출력 날짜 형식
- 기본 출력 날짜 형식은 표33에서 볼 수 있습니다.

  주: 또한, 표33에서는 다양한 국가/지역 코드에 대한 문자 형식이 나열되어 있습니다.

표 33. 국가/지역 코드로 날짜 및 시간 형식

국가/지역 코드	지역 날짜 형식	지역 시간 형식	기본 출력 날짜 형식	입력 날짜 형식
355 알바니아	yyyy-mm-dd	JIS	LOC	LOC, USA, EUR, ISO
785 아랍	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
001 오스트레일리아(1)	mm-dd-yyyy	JIS	LOC	LOC, USA, EUR, ISO
061 오스트레일리아	dd-mm-yyyy	JIS	LOC	LOC, USA, EUR, ISO
032 벨기에	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
055 브라질	dd.mm.yyyy	JIS	LOC	LOC, EUR, ISO
359 불가리아	dd.mm.yyyy	JIS	EUR	LOC, USA, EUR, ISO
001 캐나다	mm-dd-yyyy	JIS	USA	LOC, USA, EUR, ISO
002 캐나다(프랑스)	dd-mm-yyyy	ISO	ISO	LOC, USA, EUR, ISO
385 크로아티아	yyyy-mm-dd	JIS	ISO	LOC, USA, EUR, ISO
042 체코	yyyy-mm-dd	JIS	ISO	LOC, USA, EUR, ISO
045 덴마크	dd-mm-yyyy	ISO	ISO	LOC, USA, EUR, ISO
358 핀란드	dd/mm/yyyy	ISO	EUR	LOC, EUR, ISO

표 33. 국가/지역 코드로 날짜 및 시간 형식 (계속)

국가/지역 코드	지역 날짜 형식	지역 시간 형식	기본 출력 날짜 형식	입력 날짜 형식
389 FYR 마케도니아	dd.mm.yyyy	JIS	EUR	LOC, USA, EUR, ISO
033 프랑스	dd/mm/yyyy	JIS	EUR	LOC, EUR, ISO
049 독일	dd/mm/yyyy	ISO	ISO	LOC, EUR, ISO
030 그리스	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
036 헝가리	yyyy-mm-dd	JIS	ISO	LOC, USA, EUR, ISO
354 아이슬랜드	dd-mm-yyyy	JIS	LOC	LOC, USA, EUR, ISO
091 인도	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
972 이스라엘	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
039 이탈리아	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
081 일본	mm/dd/yyyy	JIS	ISO	LOC, USA, EUR, ISO
082 한국	mm/dd/yyyy	JIS	ISO	LOC, USA, EUR, ISO
001 라틴 아메리카(1)	mm-dd-yyyy	JIS	LOC	LOC, USA, EUR, ISO
003 라틴 아메리카	dd-mm-yyyy	JIS	LOC	LOC, EUR, ISO
031 네덜란드	dd-mm-yyyy	JIS	LOC	LOC, USA, EUR, ISO
047 노르웨이	dd/mm/yyyy	ISO	EUR	LOC, EUR, ISO
048 폴란드	yyyy-mm-dd	JIS	ISO	LOC, USA, EUR, ISO
351 포르투갈	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
086 중국	mm/dd/yyyy	JIS	ISO	LOC, USA, EUR, ISO
040 루마니아	yyyy-mm-dd	JIS	ISO	LOC, USA, EUR, ISO
007 러시아	dd/mm/yyyy	ISO	LOC	LOC, EUR, ISO
381 세르비아/몬테니그로	yyyy-mm-dd	JIS	ISO	LOC, USA, EUR, ISO

표 33. 국가/지역 코드로 날짜 및 시간 형식 (계속)

국가/지역 코드	지역 날짜 형식	지역 시간 형식	기본 출력 날짜 형식	입력 날짜 형식
042 슬로바키아	yyyy-mm-dd	JIS	ISO	LOC, USA, EUR, ISO
386 슬로베니아	yyyy-mm-dd	JIS	ISO	LOC, USA, EUR, ISO
034 스페인	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
046 스웨덴	dd/mm/yyyy	ISO	ISO	LOC, EUR, ISO
041 스위스	dd/mm/yyyy	ISO	EUR	LOC, EUR, ISO
088 대만	mm-dd-yyyy	JIS	ISO	LOC, USA, EUR, ISO
066 대만(2)	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
090 터키	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
044 영국	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO
001 미국	mm-dd-yyyy	JIS	USA	LOC, USA, EUR, ISO
084 베트남	dd/mm/yyyy	JIS	LOC	LOC, EUR, ISO

주:

1. 기본 C 로케일을 사용하는 국가/지역에는 국가/지역 코드 001이 지정됩니다.
2. 불교 연대의 yyyy는 양력에 543년을 합한 것과 같습니다(태국만).

DB2 UDB에서의 유니코드 지원

여기서는 DB2 UDB에서 유니코드 지원 레벨을 설명합니다.

소개

유니코드 문자 코드화 표준은 사용 중인 거의 모든 세계 언어의 문자를 포함하는 고정 길이의 코드화 체계입니다.

유니코드는 두 개의 인코딩 양식(8비트와 16비트)을 사용합니다. 기본 인코딩 양식은 16비트로 각 문자가 16비트(2바이트)이며 보통 U+hhhh로 표시됩니다. 여기서, hhhh는 문자의 16진수 코드 포인트입니다. 65 000+ 코드 요소가 세계 주요 언어의 대부분의 문자를 충분히 암호화하는 한편, 유티코드 표준은 백만 개나 더

많은 문자를 암호화할 수 있는 확장 메커니즘을 제공하기도 합니다. 확장 메커니즘은 상/하위 대리 문자 쌍을 사용하여 하나의 확장 문자를 인코드합니다. 첫 번째(또는 상위) 대리 문자는 U+D800에서 U+DBFF 사이의 코드 값을 가지며, 두 번째(또는 하위) 대리 문자는 U+DC00에서 U+DFFF 사이의 코드 값을 갖습니다.

국제 표준화 기구(ISO) 및 국제 전기 기술 위원회(IEC) 10646 표준(ISO/IEC 10646)은 16비트(2바이트) 버전(UCS-2)과 32비트(4바이트) 버전(UCS-4)을 갖는 UCS(Universal Multiple-Octet Coded Character Set)를 지정합니다. UCS-2는 대리 문자가 없는 유니코드 16비트와 동일합니다. ISO/IEC 10646는 두 개의 UCS-2 문자를 사용하여 일부 UCS-4 문자를 암호화하기 위해 확장 기술도 정의합니다. UTF-16이라는 이 확장은 대리 문자가 있는 유니코드 16비트 인코딩 양식과 동일합니다. 요약하면 UTF-16 문자 레퍼토리는 모든 UCS-2 문자에 대리 문자 쌍을 통해 액세스할 수 있는 백만 개의 문자를 더한 것입니다.

16비트 유니코드 문자를 바이트로 일련화할 때 일부 프로세서는 최상위 비트를 처음 위치에 배치하고(빅 엔디안(big-endian)순서라고 함) 다른 프로세서는 최하위 비트를 먼저 배치합니다(리틀 엔디안(little-endian) 순서라고 함). 유니코드의 기본 바이트 순서는 빅 엔디안입니다.

유니코드에 대한 자세한 정보는 유니코드 표준 서적 최종판과 유니코드 컨소시엄 웹 사이트(www.unicode.org)에 있습니다.

UTF-8

16비트 유니코드 문자는 바이트 지향 ASCII 기반 응용프로그램과 파일 시스템에 중요한 문제점을 제기합니다. 예를 들어, 비유니코드 인식 응용프로그램은 대문자 ‘A’(U+0041)의 선행 8 제로 비트를 1바이트 ASCII NULL 문자로 잘못 해석할 수 있습니다.

UTF-8(UCS 변환 형식 8)은 고정 길이 유니코드 문자를 가변 길이 ASCII-safe 바이트 문자열로 변환하는 알고리즘 변환입니다. UTF-8에서 ASCII 및 제어 문자는 보통 1바이트 코드로 표시되지만 다른 문자는 2바이트 이상입니다. 296 페이지의 표34에서 UTF-8 형식의 각 UCS-2 문자에 대한 바이트 수를 결정할 수 있습니다.

표 34. UTF-8 비트 분산

코드 값 (2진)	UTF-16 (2진)	첫 번째 바이트 (2진)	두 번째 바이트 (2진)	세 번째 바이트 (2진)	네 번째 바이트 (2진)
00000000 0xxxxxxx	00000000 0xxxxxxx	0xxxxxxx			
00000yyy yyxxxxxx	00000yyy yyxxxxxx	110yyyyy	10xxxxxx		
zzzzyyyy yyxxxxxx	zzzzyyyy yyxxxxxx	1110zzzz	10yyyyyy	10xxxxxx	
uuuuu zzzzyyyy yyxxxxxx	110110ww wwzzzzyy 110111yy yyxxxxxx	11110uuu (여기서 uuuuu = wwww+1)	10uuzzzz	10yyyyyy	10xxxxxx

위의 각 내용에서 u's, w's, x's, y's 및 z's의 연속 문자는 문자의 비트 표시입니다. 예를 들어, U+0080은 2진에서 11000010 10000000으로 변환되며, 대리 문자 쌍 U+D800 U+DC00은 2진에서 11110000 10010000 10000000으로 변환됩니다.

DB2 UDB에서의 유니코드 구현

DB2 UDB는 UCS-2 즉, 대리가 없는 유니코드를 지원합니다.

버전 7 FixPak 4 이전의 DB2 버전에서 DB2 UDB는 대리 쌍의 두 문자를 두 개의 독립 유니코드 문자로 취급합니다. 따라서 대리 쌍을 UTF-16/UCS-2에서 UTF-8로 변환하면 두 개의 3바이트 순서가 됩니다(표34의 두 번째 마지막 행 참조). DB2 UDB 버전 7.2 FixPak 4에서부터 DB2 UDB는 UTF-16/UCS-2와 UTF-8간 변환 시 대리 쌍을 인식하며 UTF-16 대리 쌍이 하나의 UTF-8 4바이트 순서가 됩니다(표34의 마지막 행 참조).

DB2 UDB는 COMBINING ACUTE ACCENT 문자(U+0301)와 같은 (비공백) 문자를 포함하여 각 유니코드 문자를 개별 문자로 취급합니다. 따라서 DB2 UDB는 문자 LATIN SMALL LETTER A WITH ACUTE(U+00E1)가 LATIN SMALL

LETTER A(U+0061) 다음에 문자 COMBINING ACUTE ACCENT(U+0301)가 오는 것과 동일하다고 인식하지 않습니다.

코드 페이지/CCSID 번호

IBM의 UCS-2 코드 페이지는 코드 페이지 1200으로 등록된 상태입니다. 즉, 새로운 문자가 코드 페이지에 추가될 때 코드 페이지 번호는 변경되지 않습니다. 코드 페이지 1200은 항상 유니코드의 현재 버전을 참조합니다.

Unicode 2.0 및 ISO/IEC 10646-1에 의해 정의된 대로, UCS 표준의 특정 버전도 IBM에 CCSID 13488로 등록되어 있습니다. 이 CCSID는 일본 EUC 및 대만 EUC 데이터베이스에 그래픽 문자열 데이터를 저장하기 위해 DB2 UDB에 의해 내부적으로 사용되었습니다. CCSID 13488 및 코드 페이지 1200은 모두 UCS-2를 참조하며, "2바이트"(DBCS) 공간의 값을 제외하고는 같은 방법으로 처리됩니다.

CP/CCSID	Single-byte (SBCS) space	Double-byte (DBCS) space
1200	N/A	U+0020
13488	N/A	U+3000

주: UCS-2 데이터베이스에서, U+3000은 아무 특별한 의미도 없습니다.

변환 테이블에 관하여는, 코드 페이지 1200이 CCSID 13488의 상위 세트이므로 동일한(상위 세트) 테이블이 둘다에 대해 사용됩니다.

IBM에서, UTF-8이 증가하는 문자 세트와 함께 CCSID 1208로 등록되어 있습니다(때로는 코드 페이지 1208로서 참조되기도 함). 새로운 문자가 표준에 추가될 때, 이 번호(1208) 역시 변경되지 않습니다.

MBCS 코드 페이지 번호는 데이터베이스 코드 페이지 번호인 1208이고, 데이터베이스의 문자열 데이터의 코드 페이지입니다. UCS-2의 2바이트 코드 페이지 번호는 데이터베이스의 그래픽 문자열 데이터의 코드 페이지인 1200입니다. 유니코드 데이터베이스가 작성될 때 CHAR, VARCHAR, LONG VARCHAR 및 CLOB 데이터는 UTF-8에 저장되고, GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC 및 DBCLOB 데이터는 UCS-2에 저장됩니다.

유니코드 데이터베이스 작성

기본 데이터베이스는 데이터베이스를 작성하는 응용프로그램의 코드 페이지에서 작성됩니다. 따라서, 유니코드(UTF-8) 클라이언트에서 데이터베이스를 작성할 경우(예

를 들어, AIX의 UNIVERSAL 로케일 또는 클라이언트의 DB2CODEPAGE 레지스트리 변수가 1208로 설정된 경우), 데이터베이스는 유니코드 데이터베이스로 작성됩니다. 또는, CODESET 이름으로 "UTF-8"을 명시적으로 지정하고, DB2 UDB가 지원하는 유효한 TERRITORY 코드를 사용할 수 있습니다.

예를 들어, 미국의 지역 코드를 사용하여 유니코드 데이터베이스를 작성하려면, 다음 명령을 발행하십시오.

```
DB2 CREATE DATABASE dbname USING CODESET UTF-8 TERRITORY US
```

sqlecrea API를 사용하여 유니코드 데이터베이스를 작성하려면, *sqledbcountryinfo*에 값을 적절하게 설정해야 합니다. 예를 들어, SQLDBCODESET을 UTF-8에 설정하고, SQLDBLOCALE을 유효한 모든 지역 코드(예: US)에 설정하십시오.

UCS-2 유니코드 데이터베이스의 기본 조합 순서는 IDENTITY이며, 문자를 코드 포인트별로 정렬합니다. 따라서 기본값으로 모든 유니코드 문자가 코드 포인트에 따라 정렬되고 비교됩니다. 대부분의 유니코드 문자의 경우, UTF-8 및 UCS-2로 인코드할 때 2진 조합 순서는 동일합니다. 그러나 인코드하려면 대리 문자 쌍이 필요한 확장 문자가 있는 경우, UTF-8 인코딩에서는 문자가 끝쪽으로 조합되지만 UCS-2 인코딩에서는 동일한 문자가 중간에서 조합되며 두 개의 대리 문자를 분리할 수 있습니다. 그 이유는 UTF-8로 인코드할 때 확장 문자가 U+FFFF UTF-8 인코딩보다 큰 11110xxx 10xxxxxx 10xxxxxx 10xxxxxx의 4바이트 2진 코드 값을 갖기 때문입니다. 그러나 UCS-2에서는 동일한 확장 문자가 두 개의 UCS-2 대리 문자로 인코드되고 DB2 UDB가 각 대리 문자를 독립적으로 취급하여 조합합니다.

날짜 또는 시간 형식, 십진수 분리자 등과 같이 문화적 차이가 있는 모든 매개변수는 클라이언트의 현재 지역에 기준을 두고 있습니다.

유니코드 데이터베이스는 DB2 UDB가 지원하는 모든 코드 페이지의 연결을 허용합니다. 클라이언트의 코드 페이지와 UTF-8간의 코드 페이지 문자 변환은 데이터베이스 관리 프로그램에 의해 자동으로 수행됩니다. 그래픽 문자열 유형의 데이터는 항상 UCS-2로 되어 있으며, 코드 페이지 변환을 통과하지 않습니다. 명령행

처리기(CLP) 환경은 예외입니다. CLP에서 선택 그래픽 문자열(UCS-2) 데이터를 선택할 경우, 리턴된 그래픽 문자열 데이터는 (CLP에 의해) UCS-2에서 클라이언트 환경의 코드 페이지로 변환됩니다.

모든 클라이언트가 문자 레퍼토리, 입력 방법 및 환경이 지원하는 글꼴에 의해 제한되지만, UCS-2 데이터베이스 자체는 모든 UCS-2 문자를 승인하고 저장합니다. 따라서, 모든 클라이언트가 일반적으로 UCS-2 문자의 부속 집합으로 작업하지만, 데이터베이스 관리 프로그램은 UCS-2 문자의 전체 레퍼토리를 허용합니다.

문자가 지역 코드 페이지에서 UTF-8로 변환될 때 바이트 수가 확장될 수 있습니다. ASCII 문자에 대해서는 확장이 일어나지 않지만, 다른 UCS-2 문자는 두세 가지의 인수에 의해 확장됩니다. UTF-8 형식의 각 UCS-2 문자의 바이트 수는 295 페이지의 『UTF-8』에 있는 테이블에서 결정될 수 있습니다.

데이터 유형

DB2 UDB에서 지원되는 모든 데이터 유형은 UCS-2 데이터베이스에서도 지원됩니다. 특히, 그래픽 문자열 데이터는 UCS-2 데이터베이스에 대해 지원되며, UCS-2/유니코드에 저장됩니다. SBCS 클라이언트를 포함하는 모든 클라이언트는 UCS-2 데이터베이스에 연결될 때 UCS-2/유니코드에 있는 그래픽 문자열 데이터 유형으로 작업할 수 있습니다.

UCS-2 데이터베이스는 문자열 데이터가 바이트 수로 측정된 MBCS 데이터베이스와 같습니다. UTF-8의 문자열 데이터로 작업할 때 각 문자가 1바이트라고 가정해서는 안 됩니다. 멀티 바이트 UTF-8 암호화에서, 각 ASCII 문자는 1바이트이지만 비 ASCII 문자는 각각 1 - 4바이트를 차지합니다. CHAR 필드를 정의할 때 이것을 고려해야 합니다. 비 ASCII 문자에 대한 ASCII 비율에 따라, *n* 바이트 크기의 CHAR 필드는 *n*/4에서 *n*자까지 포함할 수 있습니다.

그래픽 문자열 UCS-2 데이터 유형과 대비하여 문자열 UTF-8 암호화 사용은 또한 전체 저장영역 요구사항에 영향을 미칩니다. 주로 ASCII 문자를 사용하고 그 사이에 약간의 비 ASCII 문자를 사용하는 경우, UTF-8 데이터의 저장은 저장영역 요구사항이 한 문자당 1바이트에 가깝기 때문에 좋은 대안이 될 수도 있습니다. 반면에, 주로 사용된 문자가 3바이트 UTF-8 순서로 확장하는 비 ASCII 문자(예: 표의 문자)일 경우, 모든 UCS-2 문자가 UTF-8 형식에서 상응하는 각 문

자에 대해 3이나 4바이트보다는 실제로 2바이트를 필요로 하기 때문에 UCS-2 그래픽 문자열 형식이 더 좋은 대안이 될 수있습니다.

MBCS 환경에서 LENGTH, SUBSTR, POSSTR, MAX, MIN 등과 같이 문자열에 대해 작동하는 SQL 스칼라 함수는 "문자" 수보다는 "바이트" 수에 대해 작동합니다. 이 작동은 UCS-2 데이터베이스에서 동일하게 수행되지만, UCS-2 데이터베이스에 대한 오프셋과 길이를 지정할 때 특히 주의해야 합니다. 왜냐하면, 이들 값이 항상 데이터베이스 코드 페이지의 문맥에 정의되기 때문입니다. 즉, UCS-2 데이터베이스의 경우 아 오프셋은 UTF-8에 정의되어야 합니다. 일부 1바이트 문자가 UTF-8에서 2바이트 이상을 요구하기 때문에 1바이트 데이터베이스에 대해 유효한 SUBSTR 색인이 UCS-2 데이터베이스에 대해서는 유효하지 않을 수 있습니다. 부정확한 색인을 지정할 경우, SQLCODE -191(SQLSTATE 22504) 오류가 발생합니다. *SQL 참조서*에서 이 함수들의 작동에 대한 자세한 내용을 참조하십시오.

SQL CHAR 데이터 유형은 사용자 프로그램에서 C 언어의 `char` 데이터 유형에 의해 지원됩니다. SQL GRAPHIC 데이터 유형은 사용자 프로그램에서 `sqldbchar`에 의해 지원됩니다. UCS-2 데이터베이스의 경우, `sqldbchar` 데이터가 항상 빅 엔디안(큰 바이트 먼저) 형식으로 되어 있다는 것을 주지하십시오. 응용프로그램이 UCS-2 데이터베이스에 연결될 때, 문자열 데이터는 DB2 UDB에 의해 응용프로그램 코드 페이지와 UTF-8 사이에서 변환되지만, 그래픽 문자열 데이터는 항상 UCS-2로 되어 있습니다.

식별자

UCS-2 데이터베이스에서 모든 식별자는 멀티 바이트 UTF-8입니다. 따라서, DB2 UDB가 확장 문자 세트(예: 강조 문자 또는 멀티 바이트 문자)에 있는 문자 사용을 허용한 식별자에 모든 UCS-2 문자를 사용할 수 있습니다. 215 페이지의 『부록A. 이름 지정 규칙』에서 어떤 식별자가 확장 문자의 사용을 허용하는가에 대한 자세한 내용을 참조하십시오.

클라이언트는 환경이 지원하는 모든 문자를 입력할 수 있으며, 식별자에 있는 모든 문자는 데이터베이스 관리 프로그램에 의해 UTF-8로 변환됩니다. UCS-2 데이터베이스에 대한 식별자에 자국어 문자를 지정할 때 두 가지 점을 고려해야 합니다.

- 각 비 ASCII 문자는 2바이트 또는 4바이트를 요구합니다. 따라서, *n*바이트 식별자는 비 ASCII 문자에 대한 ASCII 문자 비율에 따라 *n*/4과 *n* 문자 사이에서만 보유할 수 있습니다. 한 개 또는 두 개의 비 ASCII(예: 강조 문자)만 가지고 있을 경우, 한계는 *n* 문자에 더 가깝습니다. 반면에 완전히 비 ASCII(예: 일본어)인 식별자의 경우, *n*/4 - *n*/3 문자만 사용될 수 있습니다.
- 식별자가 다른 클라이언트 환경에서 입력되는 경우, 식별자는 이 클라이언트에 사용할 수 있는 문자의 공통 부속 집합을 사용하여 정의되어야 합니다. 예를 들어, UCS-2 데이터베이스가 라틴-1, 아랍어 및 일본어 환경에서 액세스되는 경우, 모든 식별자는 현실적으로 ASCII로 제한되어야 합니다.

유니코드 리터럴

유니코드 리터럴은 두 가지 방법으로 지정될 수 있습니다.

- 그래픽 문자열 상수와 같이, G'...' 또는 N'....' 형식 사용은 *SQL 참조서*에서 "언어 요소" 장의 "그래픽 문자열 상수" 절에서 설명됩니다. 이 방법으로 지정된 모든 리터럴은 데이터베이스 관리 프로그램에 의해 응용프로그램 코드 페이지에서 16비트 유니코드로 변환됩니다.
- 유니코드 16진 문자열로 UX'....' 또는 GX'....' 형식 사용. UX 또는 GX 뒤의 따옴표 사이에 지정된 상수는 빅 엔디안(big-endian)순서로 여러 개의 4자리 16진수여야 합니다. 각 4자리 그룹은 한 개의 16비트 유니코드 코드 포인트를 나타냅니다. 대리 문자는 항상 쌍으로 나타나므로 상위 및 하위 대리 문자를 표시하려면 4자리 그룹이 필요하다는 점을 참고하십시오.

명령행 처리기(CLP)를 사용할 때, 지역 응용프로그램 코드 페이지에 UCS-2 문자가 있을 경우에는 첫 번째 방법이 보다 쉽습니다(예를 들어, 코드 페이지 850을 사용하는 터미널로부터 코드 페이지 850자를 입력할 경우). 두 번째 방법은 응용프로그램 코드 페이지 레퍼토리 밖에 있는 문자에 대해 사용되어야 합니다(예를 들어, 코드 페이지 850을 사용하는 터미널로부터 일본어 문자를 지정할 경우).

UCS-2 데이터베이스에서의 패턴 일치

패턴 일치는 기존 MBCS 데이터베이스의 작동이 UCS-2 데이터베이스의 작동과 조금 다른 영역입니다.

DB2 UDB의 MBCS 데이터베이스의 경우, 현재 작동은 다음과 같습니다. 일치 표현에 MBCS 데이터가 들어 있을 경우, 패턴은 SBCS와 비 SBCS 문자를 모두 포함할 수 있습니다. 이 패턴에 있는 특수 문자는 다음과 같이 해석됩니다.

- SBCS 밑줄은 하나의 SBCS 문자를 참조합니다.
- DBCS 밑줄은 하나의 MBCS 문자를 참조합니다.
- 퍼센트(SBCS 또는 DBCS)는 0 이상의 SBCS 또는 비 SBCS 문자열을 참조합니다.

유니코드 데이터베이스에서, ″1바이트″ 및 ″2바이트″ 문자 사이에 실제적으로 아무 구별이 없습니다. 모든 16비트 문자는 2바이트를 차지합니다. UTF-8 형식이 유니코드 문자의 ″혼합 바이트″ 암호화이기는 하지만, UTF-8에 있는 SBCS와 MBCS 문자간에 실제적인 구별은 없습니다. UTF-8 형식에서의 바이트 수와 관계없이 모든 문자는 유니코드 문자입니다. 문자열 또는 그래픽 문자열 표현식을 지정할 때, 밑줄은 하나의 유니코드 문자를 참조하고, 퍼센트는 0 이상의 유니코드 문자열을 참조합니다. 대리 쌍을 일치시키려면 두 개의 밑줄이 있어야 합니다.

클라이언트측에서, 문자열 표현식은 클라이언트의 코드 페이지에 있으며, 데이터베이스 관리 프로그램에 의해 UTF-8로 변환됩니다. SBCS 클라이언트 코드 페이지는 DBCS 퍼센트 부호 또는 DBCS 밑줄을 가지고 있지 않지만, 지원되는 모든 코드 페이지에 1바이트 퍼센트 부호(U+0025에 해당)와 1바이트 밑줄(U+005F에 해당)이 들어 있습니다. UCS-2 데이터베이스에 대한 특수 문자의 해석은 다음과 같습니다.

- SBCS 밑줄(U+0025에 해당)은 그래픽 문자열 표현식에서 하나의 UCS-2 문자를 참조하거나, 문자열 표현식에서 하나의 UTF-8 문자를 참조합니다.
- SBCS 퍼센트 기호(U+005F에 해당하는)는 그래픽 문자열 표현식에서 0 이상의 UCS-2 문자열을 참조하거나, 문자열 표현식에서 0 이상의 UTF-8 문자열을 참조합니다.

DBCS 코드 페이지는 DBCS 퍼센트 부호(U+FF05에 해당) 및 DBCS 밑줄(U+FF3F에 해당)도 지원합니다. 이들 문자는 UCS-2 데이터베이스에 대해 특별한 의미를 갖지 않습니다.

한 문자를 밑줄 및 퍼센트 부호 문자의 특수한 의미를 수정하기 위해 사용되도록 지정하는 선택적 ″escape-표현식″의 경우, ASCII 문자 또는 2바이트 UTF-8 순

서로 확장하는 문자만 지원됩니다. 3바이트 UTF-8 값으로 확장하는 Escape 문자를 지정할 경우, 오류 메세지를 받게 됩니다(오류 SQL0130N, SQLSTATE 22019).

가져오기/내보내기/로드 고려사항

이 절에 설명된 대로 UCS-2 데이터베이스에 대해 DEL, ASC 및 PC/IXF 파일 형식이 지원됩니다. WSF 형식은 지원되지 않습니다.

UCS-2 데이터베이스에서 ASCII 구분(DEL) 파일로 내보낼 때, 모든 문자 데이터가 응용프로그램 코드 페이지로 변환됩니다. 문자열과 그래픽 문자열 데이터는 모두 클라이언트의 동일한 SBCS 또는 MBCS 코드 페이지로 변환됩니다. 이것은 모든 데이터베이스의 내보내기에 대해 예상된 작동이며, 전체 구분 ASCII 파일이 한 개의 코드 페이지만 가질 수 있기 때문에 변경할 수 없습니다. 따라서, 구분된 ASCII 파일을 내보낼 경우, 응용프로그램 코드 페이지에 이미 있는 UCS-2 문자만 저장됩니다. 다른 문자는 응용프로그램 코드 페이지에 대한 기본 대체 문자로 교체됩니다. UTF-8 클라이언트(코드 페이지 1208)의 경우, 모든 UCS-2 문자가 UTF-8 클라이언트에 의해 지원되므로 데이터가 유실되지 않습니다.

ASCII 파일(DEL 또는 ASC)에서 UCS-2 데이터베이스로 가져올 때, 문자열 데이터는 응용프로그램 코드 페이지에서 UTF-8로 변환되고, 그래픽 문자열 데이터는 응용프로그램 코드 페이지에서 UCS-2로 변환됩니다. 이때 유실되는 데이터는 없습니다. 다른 코드 페이지 아래에 저장되어 있는 ASCII 데이터를 가져오려 할 경우, IMPORT 명령을 발행하기 전에 데이터 파일 코드 페이지를 변경해야 합니다. 이를 수행하기 위한 한 가지 방법은 DB2CODEPAGE를 ASCII 데이터 파일의 코드 페이지로 설정하는 것입니다.

SBCS 및 MBCS 클라이언트에 대한 유효 ASCII 분리 문자의 범위는 이들 클라이언트에 대해 DB2 UDB가 현재 지원하는 범위와 동일합니다. UTF-8 클라이언트에 대한 유효 분리 문자의 범위는 X'01' - X'7F'이며, 일반 제한사항이 적용됩니다. 이들 제한사항의 완전한 목록은 *데이터 이동 유틸리티 안내 및 참조서*에 있는 "가져오기/내보내기/로드 유틸리티 파일 형식" 부록을 참조하십시오.

UCS-2 데이터베이스에서 PC/IXF 파일로 내보낼 때, 문자열 데이터는 클라이언트의 SBCS/MBCS 코드 페이지로 변환됩니다. 그래픽 문자열 데이터는 변환되지 않으며 UCS-2(코드 페이지 1200)에 저장됩니다. 이때 유실되는 데이터는 없습니다.

PC/IXF 파일에서 UCS-2 데이터베이스로 가져올 때, 문자열 데이터는 PC/IXF 헤더에 저장된 SBCS/MBCS 코드 페이지에 있다고 간주되고, 그래픽 문자열 데이터는 PC/IXF 헤더에 저장된 DBCS 코드 페이지에 있다고 간주됩니다. 문자열 데이터는 가져오기 유틸리티에 의해 PC/IXF 헤더에 지정되어 있는 코드 페이지에서 클라이언트의 코드 페이지로 변환된 다음, 클라이언트 코드 페이지에서 UTF-8(INSERT문 사용으로)로 변환됩니다. 그래픽 문자열 데이터는 가져오기 유틸리티에 의해 PC/IXF 헤더에 지정되어 있는 DBCS 코드 페이지에서 직접 UCS-2(코드 페이지 1200)로 변환됩니다.

로드 유틸리티는 데이터를 직접 데이터베이스에 배치하고, 기본값으로, ASC 또는 DEL 파일에 있는 데이터가 데이터베이스의 코드 페이지에 있다고 간주합니다. 따라서, 기본값으로 ASCII 파일에 대해 코드 페이지 변환이 일어나지 않습니다. 데이터 파일의 코드 페이지가 명시적으로 지정되어 있으면(코드 페이지 수정자를 사용하여), 로드 유틸리티는 데이터를 로드하기 전에 이 정보를 사용하여 지정 코드 페이지에서 데이터베이스 코드 페이지로 변환합니다. PC/IXF 파일의 경우, 로드 유틸리티는 항상 IXF 헤더에 지정된 코드 페이지에서 데이터베이스 코드 페이지(CHAR의 경우 1208, GRAPHIC의 경우 1200)로 변환합니다.

DBCLOB 파일의 코드 페이지는 항상 UCS-2의 1200입니다. CLOG 파일의 코드 페이지는 가져오기, 로드 또는 내보내기 중인 데이터 파일의 코드 페이지와 동일합니다. 예를 들어, PC/IXF 형식을 사용하여 로드하거나 가져올 경우, CLOB 파일은 PC/IXF 헤더가 지정하는 코드 페이지에 있다고 간주됩니다. DBCLOB 파일이 ASC 또는 DEL 형식일 경우, CLOB 데이터는 데이터베이스의 코드 페이지에 있다고 간주되며(그렇지 않고 코드 페이지 수정자를 사용하여 명시적으로 지정한 경우를 제외하고는), 가져오기 유틸리티는 클라이언트 응용프로그램 코드 페이지에 있다고 간주됩니다.

nochecklengths 수정자는 항상 UCS-2 데이터베이스용으로 지정되는데, 그 이유는 다음과 같습니다.

- 모든 SBCS는 DBCS 코드 페이지가 없는 데이터베이스에 연결될 수 없습니다.
- UTF-8 형식의 문자열은 보통 클라이언트 코드 페이지에 있는 문자열과는 다른 길이를 갖습니다.

*데이터 이동 유틸리티 안내 및 참조서*에서 로드, 가져오기, 내보내기 유틸리티에 대한 자세한 내용을 참조하십시오.

비 호환성

UCS-2 데이터베이스에 연결된 응용프로그램에서 그래픽 문자열 데이터는 항상 UCS-2(코드 페이지 1200)입니다. 비 UCS-2 데이터베이스에 연결된 응용프로그램의 경우, 그래픽 문자열 데이터는 이 응용프로그램 DBCS 코드 페이지에 있거나, 응용프로그램 코드 페이지가 SBCS일 경우에는 그래픽 문자열 데이터가 허용되지 않습니다. 예를 들어, 932 클라이언트가 일본어의 비 UCS-2 데이터베이스에 연결되어 있을 때, 그래픽 문자열 데이터는 코드 페이지 301에 있습니다. 932 클라이언트 응용프로그램이 UCS-2 데이터베이스에 연결된 경우, 그래픽 문자열 데이터는 UCS-2입니다.

부록E. DB2 라이브러리 사용

DB2 Universal Database 라이브러리는 온라인 도움말, 책(PDF 및 HTML) 및 샘플 프로그램이 HTML 형식으로 구성됩니다. 이 절에서는 제공되는 정보 및 액세스하는 방법에 대해 설명합니다.

제품 정보에 온라인으로 액세스하려면 정보 센터를 이용할 수 있습니다. 자세한 내용은 323 페이지의 『정보 센터로 정보에 액세스』를 참조하십시오. 웹에서 타스크 정보, DB2 책, 문제점 해결 정보, 샘플 프로그램 및 DB2 정보를 볼 수 있습니다.

DB2 PDF 파일 및 인쇄된 책

DB2 정보

다음의 표는 DB2 책을 네 개의 범주로 나눕니다.

DB2 안내 및 참조 정보

이 책에는 모든 플랫폼에 공통적인 DB2 정보가 들어 있습니다.

DB2 설치 및 구성 정보

이들 책은 특정 플랫폼에서의 DB2에 대한 것입니다. 예를 들어, OS/2, Windows 및 UNIX 기반 플랫폼에서의 DB2용으로 각각 다른 *빠른 시작* 책이 있습니다.

HTML 형식의 크로스 플랫폼 샘플 프로그램

이들 샘플은 응용프로그램 개발 클라이언트와 함께 설치된 샘플 프로그램의 HTML 버전입니다. 이들은 단지 정보용으로서 실제 프로그램을 바꾸지는 않습니다.

릴리스 정보

이러한 파일에는 DB2 책에 포함되지 않은 최신 정보가 포함되어 있습니다.

설치 매뉴얼, 릴리스 정보 및 자습서는 제품 CD-ROM의 HTML 디렉토리에서 볼 수 있습니다. 대부분의 책은 단지 보기용으로 제품 CD-ROM에서 HTML 형식으로 제공되고 보기와 인쇄용으로 DB2 책 CD-ROM에서 Adobe Acrobat(PDF) 형식으로 제공됩니다. 또한 IBM에서 인쇄된 책을 주문할 수 있습니다. 318 페이지의 『인쇄된 책 주문』을 참조하십시오. 다음 표에는 주문할 수 있는 책을 보여줍니다.

OS/2 및 Windows 플랫폼에서는 `sqllib\doc\html` 디렉토리에 HTML 파일을 설치할 수 있습니다. DB2 정보는 여러 언어로 번역되었습니다. 그러나 모든 정보가 모든 언어로 번역된 것은 아닙니다. 정보를 특정 언어로 사용할 수 없을 경우에는 영문으로 제공됩니다.

UNIX 플랫폼에서는 `doc/%L/html` 디렉토리에 여러 언어 버전의 HTML 파일을 설치할 수 있습니다. 여기서 *%L*은 로케일을 나타냅니다. 자세한 내용은 *빠른 시작* 책을 참조하십시오.

다음의 여러 가지 방법으로 DB2 책을 구하고 정보에 액세스할 수 있습니다.

- 322 페이지의 『온라인 정보 보기』
- 326 페이지의 『온라인 정보 검색』
- 318 페이지의 『인쇄된 책 주문』
- 317 페이지의 『PDF 책 인쇄』

표 35. DB2 정보

이름	설명	문서 번호 **PDF** 파일 이름	**HTML** 디렉토리
	DB2 안내 및 참조 정보		
관리 안내서	*관리 안내서: 계획*에서는 데이터베이스의 개념에 대한 개요, 논리적 또는 물리적인 데이터베이스 설계와 같은 설계에 대한 정보, 그리고 고가용성에 대한 정보를 제공합니다. *관리 안내서: 구현*에서는 사용자의 설계 구현, 데이터베이스 액세스, 감사, 백업 및 복구와 같은 구현에 대한 정보를 제공합니다. *관리 안내서: 성능*에서는 데이터베이스의 환경, 응용프로그램 성능 평가 및 조정에 대한 정보를 제공합니다. 사용자는 문서 번호 SBOF-8934를 사용하여 세 권으로 된 *관리 안내서* 책을 주문할 수 있습니다.	SA30-0990 db2d1x70 SA30-0988 db2d2x70 SA30-0989 db2d3x70	db2d0
Administrative API Reference	데이터베이스를 관리하는 데 사용할 수 있는 DB2 API와 데이터 구조에 대해 설명합니다. 또한 응용프로그램에서 API를 호출하는 방법에 대해 설명합니다.	SC09-2947 db2b0x70	db2b0
응용프로그램 빌드 안내서	환경 설정 정보와 Windows, OS/2 및 UNIX 기반 플랫폼에서 DB2 응용프로그램을 컴파일, 링크 및 수행하는 방법에 대한 단계별 지침을 제공합니다.	SA30-0991 db2axx70	db2ax
APPC, CPI-C, and SNA Sense Codes	DB2 Universal Database 제품을 사용할 때 발생할 수 있는 APPC, CPI-C 및 SNA 감지 코드에 대한 일반 정보를 제공합니다. HTML 형식으로만 사용할 수 있습니다.	문서 번호 없음 db2apx70	db2ap

표 35. DB2 정보 (계속)

이름	설명	문서 번호 **PDF** 파일 이름	**HTML** 디렉토리
응용프로그램 개발 안내서	Embedded SQL 또는 Java(JDBC 및 SQLJ)를 사용하여 DB2 데이터베이스에 액세스하는 응용프로그램을 개발하는 방법에 대해 설명합니다. 저장 프로시듀어 작성, 사용자 정의 함수(UDF) 작성, 사용자 정의 유형 작성, 트리거 사용, 파티션된 환경 또는 연합 시스템에서 응용프로그램을 개발하는 등의 다양한 주제가 다루어집니다.	SA30-0992 db2a0x70	db2a0
CLI Guide and Reference	Microsoft ODBC 스펙과 호환 가능한 DB2 콜 레벨 인터페이스 및 호출 가능 SQL 인터페이스를 사용하여 DB2 데이터베이스에 액세스하는 응용프로그램을 개발하는 방법에 대해 설명합니다.	SC09-2950 db2l0x70	db2l0
Command Reference	명령행 처리기를 사용하는 방법 및 데이터베이스를 관리하기 위해 사용할 수 있는 DB2 명령에 대해 설명합니다.	SC09-2951 db2n0x70	db2n0
연결성 보충 설명서	AS/400용 DB2, OS/390용 DB2, MVS용 DB2 또는 VM용 DB2를 DB2 Universal Database 서버와의 DRDA 응용프로그램 리퀘스터(AR)로 사용하는 방법에 대한 참조 정보 및 설치 정보를 제공합니다. 또한 DB2 Connect 응용프로그램 리퀘스터(AR)와 함께 DRDA 응용프로그램 서버를 사용하는 방법에 대해서도 상세히 설명합니다. HTML 및 PDF 형식으로만 사용할 수 있습니다.	문서 번호 없음 db2h1x70	db2h1
데이터 이동 유틸리티 안내 및 참조서	가져오기, 내보내기, 로드, 자동 로드 프로그램 및 DPROP와 같이 데이터 이동을 용이하게 해 주는 DB2 UDB 유틸리티의 사용 방법에 대해 설명합니다.	SA30-0994 db2dmx70	db2dm
Data Warehouse Center 관리 안내서	Data Warehouse Center를 사용하여 데이터 웨어하우스를 빌드 및 유지보수하는 방법에 대한 정보를 제공합니다.	SA30-1000 db2ddx70	db2dd
Data Warehouse Center 응용프로그램 통합 안내서	프로그래머들이 Data Warehouse Center 및 Information Catalog Manager를 응용프로그램과 통합하는 데 도움을 주는 정보를 제공합니다.	SA30-1001 db2adx70	db2ad

표 35. DB2 정보 (계속)

이름	설명	문서 번호 **PDF** 파일 이름	**HTML** 디렉토리
DB2 Connect 사용자 안내서	DB2 Connect 제품에 대한 개념, 프로그래밍 및 일반 사용 정보를 제공합니다.	SA30-0993 db2c0x70	db2c0
DB2 Query Patroller Administration Guide	DB2 Query Patroller 시스템의 조작 개요, 특정 조작 및 관리 정보, 관리 그래픽 사용자 인터페이스 유틸리티에 대한 타스크 정보를 제공합니다.	SC09-2958 db2dwx70	db2dw
DB2 Query Patroller User's Guide	DB2 Query Patroller의 도구 및 함수를 사용하는 방법에 대해 설명합니다.	SC09-2960 db2wwx70	db2ww
용어집	DB2에서 사용되는 용어와 그 구성요소에 대한 정의를 제공합니다. HTML 형식과 *SQL 참조서*에서 사용할 수 있습니다.	문서 번호 없음 db2t0x70	db2t0
Image, Audio 및 Video Extenders 관리 및 프로그래밍	DB2 Extender에 대한 일반 정보와, 이미지, 오디오 및 비디오(IAV) Extenders 관리 및 구성에 대한 정보, 그리고 IAV Extenders를 사용한 프로그래밍에 대한 정보를 제공합니다. 여기에는 참조 정보, 진단 정보(메시지 포함) 및 샘플도 들어 있습니다.	SA30-1043 dmbu7x70	dmbu7
Information Catalog Manager Administration Guide	정보 카탈로그 관리에 대한 지침을 제공합니다.	SC26-9995 db2dix70	db2di
Information Catalog Manager Programming Guide and Reference	Information Catalog Manager에 대한 아키텍처 인터페이스에 대한 정의를 제공합니다.	SC26-9997 db2bix70	db2bi
Information Catalog Manager 사용자 안내서	Information Catalog Manager 사용자 인터페이스 사용에 대한 정보를 제공합니다.	SA30-1002 db2aix70	db2ai
설치 및 구성 보충 설명서	플랫폼 특정 DB2 클라이언트의 플랜, 설치 및 설정에 대해 설명합니다. 또한 바인딩, 클라이언트 및 서버 통신의 설정, DB2 GUI 도구, DRDA AS, 분산 설치 및 분산 요청(DR)의 구성 및 이기종 데이터 소스에 액세스 등에 대한 정보가 들어 있습니다.	GA30-0975 db2iyx70	db2iy

표 35. DB2 정보 (계속)

이름	설명	문서 번호 **PDF** 파일 이름	**HTML** 디렉토리
메시지 참조서	DB2, Information Catalog Manager 및 Data Warehouse Center가 발행하는 메시지와 코드를 나열하고 수행해야 할 조치에 대해 설명합니다. 문서 번호(SBOF-8932)를 사용하여 두 권으로 된 메시지 참조서 책을 모두 주문할 수 있습니다.	볼륨 1 GA30-0986 db2m1x70 볼륨 2 SA30-0987 db2m2x70	db2m0
OLAP Integration Server Administration Guide	OLAP Integration Server의 관리 프로그램 구성요소를 사용하는 방법에 대해 설명합니다.	SC27-0782 db2dpx70	n/a
OLAP Integration Server Metaoutline User's Guide	표준 OLAP Metaoutline 인터페이스(Metaoutline Assistant가 아닌)를 사용하여 OLAP Metaoutlines를 작성하고 처리하는 방법에 대해 설명합니다.	SC27-0784 db2upx70	n/a
OLAP Integration Server Model User's Guide	표준 OLAP 모델 인터페이스(Model Assistant가 아닌)를 사용하여 OLAP 모델을 작성하는 방법에 대해 설명합니다.	SC27-0783 db2lpx70	n/a
OLAP 설치 및 사용자 안내서	OLAP Starter Kit에 대한 구성 및 설치 정보를 제공합니다.	SA30-1074 db2ipx70	db2ip
Excel용 OLAP Spreadsheet Add-in 사용자 안내서	Excel 스프레드시트 프로그램을 사용하여 OLAP 데이터를 분석하는 방법에 대해 설명합니다.	SA30-1094 db2epx70	db2ep
Lotus 1-2-3용 OLAP Spreadsheet Add-in 사용자 안내서	Lotus 1-2-3 스프레드시트 프로그램을 사용하여 OLAP 데이터를 분석하는 방법에 대해 설명합니다.	SA30-1093 db2tpx70	db2tp
복제 안내 및 참조서	DB2와 함께 제공된 IBM 복제 도구에 관한 플랜, 구성, 관리 및 사용 정보를 제공합니다.	SA30-1003 db2e0x70	db2e0
Spatial Extender 사용자 안내 및 참조서	Spatial Extender 설치, 구성, 관리, 프로그래밍 및 문제점 해결에 대한 정보를 제공합니다. 또한 공간 데이터 개념에 대한 설명을 제공하고 Spatial Extender에 대한 특정 참조 정보(메시지 및 SQL)를 제공합니다.	SA30-1045 db2sbx70	db2sb

표 35. DB2 정보 (계속)

이름	설명	문서 번호 **PDF** 파일 이름	**HTML** 디렉토리
SQL 시작하기	SQL 개념을 소개하고 많은 구조와 타스크에 관한 예를 제공합니다.	SA30-0996 db2y0x70	db2y0
SQL 참조서, 볼륨 1 및 볼륨 2	SQL 구문, 의미 및 규칙에 대해 설명합니다. 또한 릴리스 간 비 호환성, 제품 제한사항 및 카탈로그 뷰에 대한 정보도 들어 있습니다. SBOF-8933 문서 번호를 사용하여 *SQL 참조서*를 주문할 수 있습니다.	볼륨 1 SA30-0997 db2s1x70 볼륨 2 SA30-0998 db2s2x70	db2s0
시스템 모니터 안내 및 참조서	데이터베이스와 데이터베이스 관리자에 대한 다른 종류의 정보를 수집하는 방법에 대해 설명합니다. 이 책은 데이터베이스 활동을 이해하고 성능을 향상시키며 문제점의 원인을 판별하기 위한 정보를 사용하는 방법에 대해 설명합니다.	SA30-0995 db2f0x70	db2f0
Text Extender 관리 및 프로그래밍	DB2 Extenders에 대한 일반 정보, Text Extenders 관리 및 구성에 관한 정보, Text Extenders를 사용한 프로그래밍에 대한 정보를 제공합니다. 여기에는 참조 정보, 진단 정보(메시지 포함) 및 샘플도 들어 있습니다.	SA30-1044 desu9x70	desu9
문제점 해결 안내서	오류의 소스를 판별하고 문제점으로부터 복구하며 DB2 고객 서비스와 상담하여 진단 도구를 사용하는 것을 도와줍니다.	GA30-0704 db2p0x70	db2p0
새로운 기능	DB2 Universal Database 버전 7의 새로운 특성, 기능 및 향상된 내용에 대해 설명합니다.	SA30-0999 db2q0x70	db2q0
	DB2 설치 및 구성 정보		
OS/2 및 Windows용 DB2 Connect Enterprise Edition 빠른 시작	OS/2 및 Windows 32비트 운영 체제에서 DB2 Connect Enterprise Edition에 대한 플랜, 설치, 이주 및 구성 정보를 제공합니다. 또한 지원되는 많은 클라이언트에 대한 설치 및 설정 정보도 들어 있습니다.	GA30-0974 db2c6x70	db2c6

표 35. DB2 정보 (계속)

이름	설명	문서 번호 **PDF** 파일 이름	**HTML** 디렉토리
UNIX용 DB2 Connect Enterprise Edition 빠른 시작	UNIX 기반 플랫폼에서의 DB2 Connect Enterprise Edition에 대한 플랜, 이주, 설치, 구성 및 타스크 정보를 제공합니다. 또한 지원되는 많은 클라이언트에 대한 설치 및 설정 정보도 들어 있습니다.	GA30-0973 db2cyx70	db2cy
DB2 Connect Personal Edition 빠른 시작	OS/2 및 Windows 32비트 운영 체제에서 DB2 Connect Personal Edition에 관한 플랜, 설치, 이주 및 구성 정보를 제공합니다. 또한 지원되는 모든 클라이언트에 대한 설치 및 설정 정보도 들어 있습니다.	GA30-0981 db2c1x70	db2c1
DB2 Connect Personal Edition Quick Beginnings for Linux	지원되는 모든 Linux 분산에서 DB2 Connect Personal Edition에 대한 플랜, 설치, 이주 및 구성 정보를 제공합니다.	GC09-2962 db2c4x70	db2c4
DB2 Data Links Manager 빠른 시작	AIX 및 Windows 32비트 운영 체제용 DB2 Data Links Manager에 대한 플랜, 설치, 구성 및 타스크 정보를 제공합니다.	GA30-0980 db2z6x70	db2z6
UNIX용 DB2 Enterprise - Extended Edition 빠른 시작	UNIX 기반 플랫폼에서의 DB2 Enterprise - Extended Edition 플랜, 설치 및 구성 정보를 제공합니다. 또한 지원되는 많은 클라이언트에 대한 설치 및 설정 정보도 들어 있습니다.	GA30-0978 db2v3x70	db2v3
Windows용 DB2 Enterprise - Extended Edition 빠른 시작	Windows 32비트 운영 체제용 DB2 Enterprise - Extended Edition에 대한 플랜, 설치 및 구성 정보를 제공합니다. 또한 지원되는 많은 클라이언트에 대한 설치 및 설정 정보도 들어 있습니다.	GA30-0977 db2v6x70	db2v6
OS/2용 DB2 빠른 시작	OS/2 운영 체제에서의 DB2 Universal Database에 대한 플랜, 설치, 이주 및 구성 정보를 제공합니다. 또한 지원되는 많은 클라이언트에 대한 설치 및 설정 정보도 들어 있습니다.	GA30-0982 db2i2x70	db2i2
UNIX용 DB2 빠른 시작	UNIX 기반 플랫폼에서의 DB2 Universal Database에 대한 플랜, 설치, 이주 및 구성 정보를 제공합니다. 또한 지원되는 많은 클라이언트에 대한 설치 및 설정 정보도 들어 있습니다.	GA30-0984 db2ixx70	db2ix

표 35. DB2 정보 (계속)

이름	설명	문서 번호 **PDF** 파일 이름	**HTML** 디렉토리
Windows용 DB2 빠른 시작	Windows 32비트 운영 체제에서의 DB2 Universal Database에 대한 플랜, 설치, 이주 및 구성 정보를 제공합니다. 또한 지원되는 많은 클라이언트에 대한 설치 및 설정 정보도 들어 있습니다.	GA30-0985 db2i6x70	db2i6
DB2 Personal Edition 빠른 시작	OS/2 및 Windows 32비트 운영 체제에서의 DB2 Universal Database Personal Edition에 대한 플랜, 설치, 이주 및 구성 정보를 제공합니다.	GA30-0983 db2i1x70	db2i1
DB2 Personal Edition Quick Beginnings for Linux	지원되는 모든 Linux 분산에서 DB2 Universal Database Personal Edition에 대한 플랜, 설치, 이주 및 구성 정보를 제공합니다.	GC09-2972 db2i4x70	db2i4
DB2 Query Patroller 설치 안내서	DB2 Query Patroller에 대한 설치 정보를 제공합니다.	GA30-0976 db2iwx70	db2iw
DB2 Warehouse Manager 설치 안내서	웨어하우스 에이전트, 웨어하우스 변환기 및 Information Catalog Manager에 대한 설치 정보를 제공합니다.	GA30-1027 db2idx70	db2id
	HTML 형식의 크로스 플랫폼 샘플 프로그램		
HTML 형식의 샘플 프로그램	DB2가 지원하는 모든 플랫폼에서 프로그래밍 언어에 대한 샘플 프로그램이 HTML 형식으로 제공됩니다. 이 샘플 프로그램은 정보용으로만 제공됩니다. 모든 샘플을 모든 프로그래밍 언어로 사용할 수 있는 것은 아닙니다. HTML 샘플은 DB2 응용프로그램 개발 클라이언트가 설치될 때만 사용할 수 있습니다. 프로그램에 대한 자세한 내용은 *응용프로그램 빌드 안내서*를 참조하십시오.	문서 번호 없음	db2hs
	릴리스 정보		
DB2 Connect 릴리스 정보	DB2 Connect 책에는 포함될 수 없었던 최신 정보를 제공합니다.	#2를 참조하십시오.	db2cr
DB2 설치 정보	DB2 책에는 포함될 수 없었던 최신 설치 특정 정보를 제공합니다.	제품 CD-ROM에서만 사용할 수 있습니다.	

표 35. DB2 정보 (계속)

이름	설명	문서 번호 PDF 파일 이름	HTML 디렉토리
DB2 릴리스 정보	DB2 책에는 포함될 수 없었던 모든 DB2 제품 및 기능에 대한 최신 정보를 제공합니다.	#2를 참조하십시오.	db2ir

주:

1. 파일 이름의 6번째 자리에 있는 문자 *x*는 책의 언어 버전을 나타냅니다. 예를 들어, 파일 이름 db2d0e70은 *관리 안내서* 책의 영문 버전을 나타내며 db2d0k70은 같은 책의 한글 버전을 나타냅니다. 다음 문자는 파일 이름의 6번째 자리에 사용되어 언어 버전을 나타냅니다.

언어	식별자
브라질 포르투갈어	b
불가리아어	u
체코어	x
덴마크어	d
네덜란드어	q
영어	e
핀란드어	y
프랑스어	f
독일어	g
그리스어	a
헝가리어	h
이탈리아어	i
일본어	j
한국어	k
노르웨이어	n
폴란드어	p
포르투갈어	v
러시아어	r
중국어	c
슬로베니아어	l
스페인어	z
스웨덴어	s
대만어	t
터키어	m

2. DB2 책에 포함되어 있지 않을 수 있는 최신 정보는 릴리스 정보에서 HTML 형식과 ASCII 파일로 사용할 수 있습니다. HTML 버전은 정보 센터와 제품 CD-ROM에서 사용할 수 있습니다. ASCII 파일을 보려면 다음을 수행하십시오.
   - UNIX 기반 플랫폼의 경우에는 Release.Notes 파일을 참조하십시오. 이 파일은 DB2DIR/Readme/%L 디렉토리에 있으며, 여기서 %L은 로케일 이름을 나타내고 DB2DIR은 다음과 같습니다.
     - AIX에서는 /usr/lpp/db2_07_01
     - HP-UX, PTX, Solaris 및 Silicon Graphics IRIX에서는 /opt/IBMdb2/V7.1
     - Linux에서는 /usr/IBMdb2/V7.1
   - 다른 플랫폼의 경우에는 RELEASE.TXT 파일을 참조하십시오. 이 파일은 제품이 설치된 디렉토리에 있습니다. OS/2 플랫폼에서는 **IBM DB2** 폴더를 더블 클릭한 다음 **릴리스 정보** 아이콘을 더블 클릭할 수 있습니다.

PDF 책 인쇄

인쇄된 책의 사본을 원하는 경우, DB2 책 CD-ROM에 있는 PDF 파일을 인쇄할 수 있습니다. Adobe Acrobat Reader를 사용하여 책 전체나 특정 페이지를 인쇄할 수 있습니다. 라이브러리에 있는 각 책의 파일 이름에 대한 자세한 내용은 309 페이지의 표35를 참조하십시오.

Adobe 웹 사이트 http://www.adobe.com에서 Adobe Acrobat Reader의 최신 버전을 얻을 수 있습니다.

PDF 파일은 파일 확장자가 PDF인 DB2 책 CD-ROM에 들어 있습니다. PDF 파일에 액세스하려면 다음을 수행하십시오.

1. DB2 책 CD-ROM을 삽입하십시오. UNIX 기반의 플랫폼에서는 DB2 책 CD-ROM을 마운트합니다. 마운트 프로시듀어에 대한 자세한 내용은 *빠른 시작* 책을 참조하십시오.
2. Acrobat Reader를 시작하십시오.
3. 다음 위치 중 하나에서 원하는 PDF 파일을 여십시오.
   - OS/2 및 Windows 플랫폼에서

x:\doc*language* 디렉토리. 여기서 *x*는 CD-ROM 드라이브를 나타내며 *language*는 사용자 언어를 나타내는 2문자 국가 코드를 나타냅니다. 예를 들어, 영문인 경우에는 EN입니다.

- UNIX 기반 플랫폼에서

  /*cdrom*/doc/*%L* 디렉토리. 여기서 */cdrom*은 CD-ROM의 마운트 지점을 나타내고 *%L*은 원하는 로케일의 이름을 나타냅니다.

또한 PDF 파일을 CD-ROM에서 지역이나 네트워크 드라이브로 파일을 복사하고 거기서 읽을 수도 있습니다.

인쇄된 책 주문

인쇄된 DB2 책은 책 주문 번호(SBOF)를 사용하여 세트나 낱권으로 주문할 수 있습니다. 인쇄본을 주문하려면 IBM 협력업체 또는 영업 대표에게 문의하십시오. 또한 웹 페이지 http://www.elink.ibmlink.ibm.com/pbl/pbl에서도 책을 주문할 수 있습니다.

두 종류의 책 세트를 사용할 수 있습니다. SBOF-8935는 DB2 Warehouse Manager에 대한 참조 및 사용에 관한 정보를 제공합니다. SBOF-8931은 다른 모든 DB2 Universal Database 제품과 기능에 대한 참조 및 사용 정보를 제공합니다. 각 SBOF의 내용은 다음 표에 나열되어 있습니다.

표 36. 인쇄된 책 주문

SBOF 번호	포함된 책	
SBOF-8931	• 관리 안내서: 계획 • 관리 안내서: 구현 • 관리 안내서: 성능 • Administrative API Reference • 응용프로그램 빌드 안내서 • 응용프로그램 개발 안내서 • CLI Guide and Reference • Command Reference • 데이터 이동 유틸리티 안내 및 참조서 • Data Warehouse Center 관리 안내서 • Data Warehouse Center 응용프로그램 통합 안내서 • DB2 Connect 사용자 안내서 • 설치 및 구성 보충 설명서 • Image, Audio, and Video Extenders 관리 및 프로그래밍 • 메시지 참조서, 볼륨 1 및 2	• OLAP Integration Server Administration Guide • OLAP Integration Server Metaoutline User's Guide • OLAP Integration Server Model User's Guide • OLAP Integration Server User's Guide • OLAP 설치 및 사용자 안내서 • Excel용 OLAP Spreadsheet Add-in 사용자 안내서 • Lotus 1-2-3용 OLAP Spreadsheet Add-in 사용자 안내서 • 복제 안내 및 참조서 • Spatial Extender Administration and Programming Guide • SQL 시작하기 • SQL 참조서, 볼륨 1 및 2 • 시스템 모니터 안내 및 참조서 • Text Extender 관리 및 프로그래밍 • 문제점 해결 안내서 • 새로운 기능
SBOF-8935	• Information Catalog Manager Administration Guide • Information Catalog Manager 사용자 안내서 • Information Catalog Manager Programming Guide and Reference	• Query Patroller Administration Guide • Query Patroller User's Guide

DB2 온라인 문서

온라인 도움말 액세스

온라인 도움말은 모든 DB2 구성요소에서 사용 가능합니다. 다음의 표에서는 다양한 도움말 유형에 대해 설명합니다.

도움말 유형	내용	액세스하는 방법
명령 도움말	명령행 처리기의 명령 구문에 대해 설명합니다.	대화식 모드의 명령행 처리기에서 다음을 입력하십시오. `? command` 여기서 *command*는 키워드이거나 전체 명령입니다. 예를 들어, `? catalog`는 모든 CATALOG 명령에 대한 도움말을 표시하고, `? catalog database`는 CATALOG DATABASE 명령에 대한 도움말을 표시합니다.
클라이언트 구성 지원 프로그램 도움말 *명령 센터 도움말* *제어 센터 도움말* *Data Warehouse Center 도움말* *이벤트 분석기 도움말* *Information Catalog Manager 도움말* *위성 관리 센터 도움말* *스크립트 센터 도움말*	창 또는 노트북에서 수행할 수 있는 타스크에 대해 설명합니다. 도움말은 알아야 할 개요와 전제조건 정보를 포함하고, 창 또는 노트북 제어를 사용하는 방법에 대해 설명합니다.	창이나 노트북에서 **도움말** 버튼을 누르거나 **F1** 키를 누르십시오.

도움말 유형	내용	액세스하는 방법

메시지 도움말	메시지의 원인과 사용자가 취해야 할 조치에 대해 설명합니다.	대화식 모드의 명령행 처리기에서 다음을 입력하십시오.
		`? XXXnnnnn`
		여기서 *XXXnnnnn*은 유효한 메시지 식별자입니다.
		예를 들어, `? SQL30081`은 SQL30081 메시지에 대한 도움말을 표시합니다.
		한 번에 한 화면씩 메시지 도움말을 보려면 다음을 입력하십시오.
		`? XXXnnnnn | more`
		파일에 메시지 도움말을 저장하려면 다음을 입력하십시오.
		`? XXXnnnnn > filename.ext`
		여기서 *filename.ext*는 메시지 도움말을 저장하려는 파일입니다.

SQL 도움말	SQL문의 구문에 대해 설명합니다.	대화식 모드의 명령행 처리기에서 다음을 입력하십시오. `help statement` 여기서 *statement*는 SQL문입니다. 예를 들어, `help SELECT`는 SELECT문에 대한 도움말을 표시합니다. **주:** SQL 도움말은 UNIX 기반 플랫폼에서 사용할 수 없습니다.
SQLSTATE 도움말	SQL 상태와 클래스 코드에 대해 설명합니다.	대화식 모드의 명령행 처리기에서 다음을 입력하십시오. `? sqlstate` 또는 `? class code` 여기서 *sqlstate*는 유효한 5자리 숫자로 된 SQL 상태이고 *class code*는 SQL 상태의 처음 2자리 숫자입니다. 예를 들어, `? 08003`은 08003 SQL 상태에 대한 도움말을 표시하고, `? 08`은 08 클래스 코드에 대한 도움말을 표시합니다.

온라인 정보 보기

이 제품에 들어 있는 책은 HTML(Hypertext Markup Language) 소프트카피 형식으로 제공됩니다. 소프트카피 형식은 정보를 검색할 수 있게 하고 관련된 정보로 링크하는 하이퍼텍스트를 제공합니다. 또한 사이트에서 라이브러리를 공유하는 것도 더 쉬워집니다.

HTML 버전 3.2 스펙을 따르는 브라우저로 온라인 책 또는 샘플 프로그램을 볼 수 있습니다.

온라인 책 또는 샘플 프로그램을 보려면 다음을 수행하십시오.

- DB2 관리 도구를 수행할 경우, 정보 센터를 사용하십시오.
- 브라우저에서 **파일 —> 페이지 열기**를 누르십시오. 열린 페이지에 DB2 정보에 대한 설명과 링크가 들어 있습니다.
  - UNIX 기반 플랫폼의 경우, 다음 페이지를 여십시오.

    INSTHOME/sqllib/doc/*%L*/html/index.htm

    여기서 *%L*은 로케일 이름입니다.
  - 기타 플랫폼에서는 다음 페이지를 여십시오.

    sqllib\doc\html\index.htm

    이 경로는 DB2가 설치된 드라이브에 있습니다.

    정보 센터를 설치하지 않은 경우, **DB2 정보** 아이콘을 더블 클릭하여 페이지를 열 수 있습니다. 사용하는 시스템에 따라 주 제품 폴더나 Windows 시작 메뉴에 아이콘이 있습니다.

Netscape 브라우저 설치

웹 브라우저를 설치하지 않은 경우, 제품 상자에 있는 Netscape CD-ROM에서 Netscape를 설치할 수 있습니다. 설치하는 방법에 대한 자세한 지침을 보려면 다음을 수행하십시오.

1. Netscape CD-ROM을 삽입하십시오.
2. UNIX 기반 플랫폼에서는 CD-ROM을 마운트하십시오. 마운트 프로시듀어에 대한 자세한 내용은 *빠른 시작* 책을 참조하십시오.

3. 설치 지침의 경우에는 CDNAV*nn*.txt 파일을 참조하십시오. 여기서 *nn*은 2문자로 된 언어 식별자입니다. 파일은 CD-ROM의 루트 디렉토리에 있습니다.

정보 센터로 정보에 액세스

정보 센터는 DB2 제품 정보에 대한 빠른 액세스를 제공합니다. 정보 센터는 DB2 관리 도구를 사용할 수 있는 모든 플랫폼에서 사용할 수 있습니다.

정보 센터 아이콘을 더블 클릭하여 정보 센터를 열 수 있습니다. 사용하는 시스템에 따라 아이콘은 주 제품 폴더나 Windows **시작** 메뉴의 정보 폴더에 있습니다.

또한 DB2 Windows 플랫폼에서 도구 모음이나 **도움말** 메뉴를 사용하여 정보 센터에 액세스할 수 있습니다.

정보 센터는 6개 유형의 정보를 제공합니다. 적절한 탭을 눌러 해당 유형에 제공되는 주제를 보십시오.

타스크 DB2를 사용하여 수행할 수 있는 주요 타스크.

참조 키워드, 명령 및 API와 같은 DB2 참조 정보.

책 DB2 책.

문제점 해결 오류 메시지의 범주와 복구 조치.

샘플 프로그램 DB2 응용프로그램 개발 클라이언트와 함께 제공되는 샘플 프로그램. DB2 응용프로그램 개발 클라이언트를 설치하지 않은 경우, 이 탭은 표시되지 않습니다.

웹 월드 와이드 웹에서의 DB2 정보. 이 정보에 액세스하려면 시스템에서 웹으로의 연결을 갖고 있어야 합니다.

목록 중 하나에서 항목을 선택하면 정보 센터가 표시기를 시작하여 정보를 표시합니다. 표시기는 사용자가 선택하는 정보의 종류에 따라 시스템 도움말 표시기, 편집기 또는 웹브라우저가 될 수 있습니다.

정보 센터는 찾기 기능을 제공하므로 목록을 찾지 않고도 특정 주제를 찾을 수 있습니다.

전체 텍스트 검색의 경우, **DB2 온라인 정보 검색** 검색 양식으로 연결된 정보 센터의 하이퍼텍스트 링크를 따라 검색하십시오.

HTML 검색 서버는 보통 자동으로 시작됩니다. HTML 정보에서 검색 기능이 작동하지 않으면 다음 방법 중 하나를 사용하여 검색 서버를 시작할 수 있습니다.

Windows의 경우

시작을 누르고 **프로그램 —> IBM DB2 —> 정보 —> HTML 검색 서버 시작**을 선택하십시오.

OS/2의 경우

OS/2용 DB2 폴더를 더블 클릭한 다음 **HTML 검색 서버 시작** 아이콘을 더블 클릭하십시오.

HTML 정보 검색시 그 외의 다른 문제가 발생한 경우에는 릴리스 정보를 참조하십시오.

주: 검색 기능은 Linux, PTX 및 Silicon Graphics IRIX 환경에서는 사용할 수 없습니다.

DB2 마법사 사용

마법사는 한 번에 한 단계씩 각 타스크를 수행하게 함으로써 특정 관리 타스크를 완료하는 데 도움을 줍니다. 마법사는 제어 센터 및 클라이언트 구성 지원 프로그램을 통해 사용할 수 있습니다. 다음 표에서는 마법사를 나열하고 해당 기능에 대해 설명합니다.

주: 데이터베이스 작성, 색인 작성, 다중 사이트 갱신 구성 및 성능 구성 마법사는 파티션된 데이터베이스 환경에서 사용할 수 있습니다.

마법사	도움 내용	액세스하는 방법
데이터베이스 추가	클라이언트 워크스테이션의 데이터베이스를 카탈로그화합니다.	클라이언트 구성 지원 프로그램에서 **추가**를 누르십시오.
데이터베이스 백업	백업 플랜의 결정, 작성 및 스케줄합니다.	제어 센터에서 백업하려는 데이터베이스를 마우스 오른쪽 단추로 누른 다음 **백업 —> 마법사를 사용한 데이터베이스**를 선택하십시오.
다중 사이트 갱신 구성	다중 사이트 갱신, 분산 트랜잭션 또는 2단계 확약을 구성합니다.	제어 센터에서 **데이터베이스** 폴더를 마우스 오른쪽 단추로 누른 다음 **다중 사이트 갱신**을 선택하십시오.

마법사	도움 내용	액세스하는 방법
데이터베이스 작성	데이터베이스 작성 및 일부 기본 구성 타스크를 수행합니다.	제어 센터에서 **데이터베이스** 폴더를 마우스 오른쪽 단추로 누른 다음 **작성 —> 마법사를 사용한 데이터베이스**를 선택하십시오.
테이블 작성	기본 데이터 유형의 선택 및 테이블의 기본 키를 작성합니다.	제어 센터에서 **테이블** 아이콘을 마우스 오른쪽 단추로 누른 다음 **작성 —> 마법사를 사용한 테이블**을 선택하십시오.
테이블 공간 작성	새로운 테이블 공간을 작성합니다.	제어 센터에서 **테이블 공간** 아이콘을 마우스 오른쪽 단추로 선택한 다음 **작성 —> 마법사를 사용한 테이블 공간**을 선택하십시오.
색인 작성	사용자의 모든 조회를 작성하고 삭제하기 위해 색인화합니다.	제어 센터에서 **색인** 아이콘을 마우스 오른쪽 단추로 누른 다음 **작성 —> 마법사를 사용한 색인**을 선택하십시오.
성능 구성	비지니스 요구에 맞게 구성 매개변수를 갱신하여 데이터베이스의 성능을 조정합니다.	제어 센터에서 조정하려는 데이터베이스를 마우스 오른쪽 단추로 누른 다음 **마법사를 사용한 성능 구성**을 선택하십시오. 파티션된 데이터베이스 환경에 대해 데이터베이스 파티션 뷰에서 조정하려는 첫 번째 데이터베이스 파티션을 마우스 오른쪽 단추로 누른 다음 **마법사를 사용한 성능 구성**을 선택하십시오.
데이터베이스 복원	실패 후에 데이터베이스를 복구합니다. 사용할 백업 종류 및 재작동할 로그를 이해하는 데 도움을 줍니다.	제어 센터에서 복원하려는 데이터베이스를 마우스 오른쪽 단추로 누른 다음 **복원 —> 마법사를 사용한 데이터베이스**를 선택하십시오.

문서 서버 설정

기본값으로 DB2 정보는 지역 시스템에 설치됩니다. 이는 DB2 정보에 액세스해야 하는 모든 사람이 동일한 파일을 설치해야 함을 의미합니다. DB2 정보를 단일 위치에 저장하려면 다음 단계를 수행하십시오.

1. 모든 파일과 서브디렉토리를 지역 시스템의 \sqllib\doc\html에서 웹 서버로 복사하십시오. 각 책은 책을 구성하는 데 필요한 모든 HTML 및 GIF 파일이 들어 있는 서브디렉토리를 가집니다. 디렉토리 구조가 변경되지 않게 하십시오.
2. 새로운 위치에 있는 파일을 찾도록 웹 서버를 구성하십시오. 자세한 내용은 *설치 및 구성 보충 설명서*의 부록 NetQuestion을 참조하십시오.
3. Java 버전의 정보 센터를 사용 중인 경우, 모든 HTML 파일에 대한 기본 URL을 지정할 수 있습니다. 책 목록에 대한 URL을 사용해야 합니다.
4. 책 파일을 볼 수 있게 되면 다음과 같이 자주 보는 주제 항목에 대해서는 책갈피를 설정할 수 있습니다. 다음 페이지를 책갈피로 설정해 두면 도움이 됩니다.
   - 책 목록
   - 자주 사용되는 책의 목차
   - ALTER TABLE 주제와 같은 자주 참조하는 항목
   - 검색 양식

DB2 Universal Database 온라인 문서 파일을 중앙 머신에서 제공하는 방법에 대한 자세한 내용은 *설치 및 구성 보충 설명서*의 부록 NetQuestion을 참조하십시오.

온라인 정보 검색

HTML 파일에서 정보를 찾으려면 다음 방법 중 하나를 사용하십시오.

- 맨 위 프레임에서 **검색**을 누르십시오. 특정 주제를 찾으려면 검색 양식을 사용하십시오. 이 기능은 Linux, PTX 또는 Silicon Graphics IRIX 환경에서는 사용할 수 없습니다.
- 맨 위 프레임에서 **색인**을 누르십시오. 책에서 특정 주제를 찾으려면 색인을 사용하십시오.
- 책에서 특정 주제를 찾으려면 목차나 도움말의 색인 또는 HTML 책을 표시하고 웹 브라우저의 찾기 기능을 사용하십시오.
- 특정 주제로 빨리 리턴하려면 웹 브라우저의 책갈피 기능을 사용하십시오.
- 특정 주제를 찾으려면 정보 센터의 검색 기능을 사용하십시오. 자세한 내용은 323 페이지의 『정보 센터로 정보에 액세스』를 참조하십시오.

부록F. 주의사항

IBM은 다른 국가에서 이 책에 기술된 제품, 서비스 또는 기능을 제공하지 않을 수도 있습니다. 현재 사용할 수 있는 제품 및 서비스에 대한 정보는 한국 IBM 담당자에게 문의하십시오. 이 책에서 IBM 제품, 프로그램 또는 서비스를 언급했다고 해서 해당 IBM 제품, 프로그램 또는 서비스만을 사용할 수 있다는 것을 의미하지는 않습니다. IBM의 지적 재산권을 침해하지 않는 한, 기능상으로 동등한 제품, 프로그램 또는 서비스를 대신 사용할 수도 있습니다. 그러나 비IBM 제품, 프로그램 또는 서비스의 운영에 대한 평가 및 검증은 사용자의 책임입니다.

IBM은 이 책에서 다루고 있는 특정 내용에 대해 특허를 보유하고 있거나 현재 특허 출원 중일 수 있습니다. 이 책을 제공한다고 해서 특허에 대한 사용권까지 부여하는 것은 아닙니다. 사용권에 대한 의문사항은 다음으로 문의하십시오.

135-270
서울특별시 강남구 도곡동 467-12, 군인공제회관빌딩
한국 아이.비.엠 주식회사
고객만족센터
전화번호: 080-023-8080

2 바이트(DBCS) 정보에 관한 사용권 문의는 한국 IBM 고객만족센터에 문의하거나 다음 주소로 서면 문의하시기 바랍니다.

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

다음 단락은 현지법과 상충하는 영국이나 기타 국가에서는 적용되지 않습니다. IBM은 타인의 권리 비침해, 상품성 및 특정 목적에의 적합성에 대한 묵시적 보증을 포함하여(단, 이에 한하지 않음) 묵시적이든 명시적이든 어떠한 종류의 보증없이 이 책을『현상태대로』제공합니다. 일부 국가에서는 특정 거래에서 명시적 또는 묵시적 보증의 면책사항을 허용하지 않으므로, 이 사항이 적용되지 않을 수도 있습니다.

이 정보에는 기술적으로 부정확한 내용이나 인쇄상의 오류가 있을 수 있습니다. 이 정보는 주기적으로 변경되며, 이 변경사항은 최신판에 통합됩니다. IBM은 이 책에서 설명한 제품 및/또는 프로그램을 사전 통고 없이 언제든지 개선 및/또는 변경할 수 있습니다.

이 정보에서 비IBM의 웹 사이트는 단지 편의상 제공된 것으로, 어떤 방식으로든 이들 웹 사이트를 옹호하고자 하는 것은 아닙니다. 해당 웹 사이트의 자료는 본 IBM 제품 자료의 일부가 아니므로 해당 웹 사이트의 사용으로 인한 위험은 사용자 본인이 감수해야 합니다.

IBM은 귀하의 권리를 침해하지 않는 범위 내에서 적절하다고 생각하는 방식으로 귀하가 제공한 정보를 사용하거나 배포할 수 있습니다.

(i) 독립적으로 작성된 프로그램 및 기타 프로그램(이 프로그램 포함)간의 정보 교환 (ii) 교환된 정보의 상호 이용을 목적으로 정보를 원하는 프로그램 사용권자는 다음 주소로 문의하십시오.

135-270
서울특별시 강남구 도곡동 467-12, 군인공제회관빌딩
한국 아이.비.엠 주식회사
고객만족센터

이러한 정보는 해당 조항 및 조건(예를 들면, 사용권 지불 포함)에 따라 사용할 수 있습니다.

이 정보에 기술된 사용권 프로그램 및 사용가능한 모든 사용권 자료는 IBM이 IBM 기본 계약, IBM 프로그램 사용권 계약(IPLA) 또는 이와 동등한 계약에 따라 제공한 것입니다.

여기에 있는 모든 성능 데이터는 제한된 환경에서 산출된 것입니다. 따라서, 다른 운영 환경에서 얻어진 결과는 상당히 다를 수 있습니다. 일부 측정치는 개발 레벨 시스템에서 작성되었을 수 있으며, 이러한 측정치가 일반적으로 사용가능한 시스템에서도 동일하다고는 보장하지 않습니다. 더우기, 일부 측정치는 추정을 통해 추측되었을 수도 있으므로 실제 결과는 다를 수 있습니다. 이 책의 사용자는 본인의 특정 환경에 적용할 수 있는 데이터를 검증해야 합니다.

비IBM 제품에 관한 정보는 해당 제품의 공급업체, 공개자료 또는 기타 범용 소스로부터 얻은 것입니다. IBM에서는 이러한 제품들을 테스트하지 않았으므로, 비IBM 제품과 관련된 성능의 정확성, 호환성 또는 배상 청구에 대해서는 확신할 수 없습니다. 비IBM 제품의 성능에 대한 의문사항은 해당 제품의 공급업체에 문의하십시오.

IBM이 제시하는 방향 또는 의도에 관한 모든 언급은 특별한 통지없이 변경될 수 있습니다.

이 정보에는 일상의 업무에서 사용되는 자료와 보고의 예제가 포함되어 있을 수 있습니다. 가능한 완벽하게 설명하기 위해 개인, 회사, 상표 및 제품의 이름이 예제에 들어 있습니다. 이들 이름은 모두 가공의 것이며, 실제 기업의 이름 및 주소와 유사하더라도 이는 전적으로 우연입니다.

저작권:

이 정보에는 여러 가지 운영 플랫폼에서의 프로그래밍 기법을 보여주는 원어로 된 샘플 응용프로그램이 포함되어 있을 수 있습니다. 샘플 응용프로그램의 작성 기준이 된 운영 플랫폼의 응용프로그램 프로그래밍 인터페이스에 부합하는 응용프로그램을 개발, 사용, 마케팅 또는 배포를 목적으로 이들 샘플 프로그램을 복사, 수정 및 배포할 수 있으며 IBM에 대한 지불 의무는 없습니다. 이들 예제 프로그램은 모든 조건에서 철저하게 검사된 것은 아닙니다. 따라서, IBM은 이들 프로그램의 신뢰성, 서비스 가능성 또는 기능에 대해 어떠한 보증도 하지 않습니다.

각 사본이나 이들 샘플 프로그램의 일부 또는 파생본에는 다음과 같은 저작권 주의사항을 포함시켜야 합니다.

상표

별표(*)로 표시된 다음의 용어는 전세계에서 IBM의 상표입니다.

ACF/VTAM
AISPO
AIX
AIX/6000
AIXwindows
AnyNet
APPN
AS/400
BookManager CICS
C Set++
C/370
DATABASE 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 ConnectDB2 Extenders
DB2 OLAP Server
DB2 Universal Database
Distributed Relational
Database Architecture
DRDA
eNetwork
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
LAN DistanceMVS
MVS/ESA
MVS/XA
Net.Data
OS/2OS/390
OS/400
PowerPC
QBIC
QMF
RACF
RISC System/6000
RS/6000S/370
SP
SQL/DS
SQL/400
System/370
System/390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WIN-OS/2

다음 용어는 기타 회사의 상표 또는 등록상표입니다.

Microsoft, Windows 및 Windows NT는 Microsoft Corporation의 상표 또는 등록상표입니다.

Java 또는 모든 Java 관련 상표와 로고 및 Solaris는 미국 또는 기타 국가에서 사용되는 Sun Microsystems, Inc.의 상표입니다.

Tivoli 및 NetView는 미국 또는 기타 국가에서 사용되는 Tivoli Systems Inc.의 상표입니다.

UNIX는 미국 또는 기타 국가에서 X/Open Company Limited가 독점적인 사용권을 가진 등록상표입니다.

두 개의 별표(**)가 붙은 기타 회사 이름, 제품 이름 또는 서비스 이름은 해당 회사의 상표이거나 서비스표입니다.

색인

[라]

[마]

[바]

[사]

[아]

[자]

[차]

[카]

[타]

[파]

[하]

[숫자]

C

D

E

F

G

H

I

IBM에 문의

기술적인 문제가 발생한 경우에는 DB2 고객 지원 센터에 문의하기 전에 *문제점 해결 안내서*에서 제안한 조치를 검토하고 실행해 보십시오. 이것은 DB2 고객 지원 부서로 하여금 사용자를 보다 더 잘 지원할 수 있도록 사용자가 모을 수 있는 정보를 제공합니다.

DB2 Universal Database 제품에 대한 정보나 주문은 그 지역의 IBM 영업 대표나 공인 IBM 소프트웨어 재판매업자에게 문의하십시오.

미국에 사시는 분은 다음 번호 중 하나를 선택하여 전화하십시오.

- 고객 지원을 받으려면, 1-800-237-5511.
- 사용가능한 서비스 옵션을 알려면, 1-888-426-4343.

제품 정보

미국에 사시는 분은 다음 번호 중 하나를 선택하여 전화하십시오.

- 제품 주문이나 일반 정보를 얻으려면 1-800-IBM-CALL(1-800-426-2255) 또는 1-800-3IBM-OS2(1-800-342-6672).
- 책에 대한 주문은, 1-800-879-2755.

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

DB2 World Wide Web 페이지에서는 새로운 소식, 제품 설명, 교육 일정 등에 관한 현재 DB2 정보를 제공합니다.

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

DB2 제품 및 서비스 기술 라이브러리는 빈도 높은 질문(FAQ), 수정사항(fixes), 책 및 최신 DB2 기술 정보에 대한 액세스를 제공합니다.

주: 이러한 정보는 영어로만 제공됩니다.

http://www.elink.ibmlink.ibm.com/pbl/pbl/

여기에서는 책을 웹 사이트에서 주문할 수 있는 방법을 제공합니다.

http://www.ibm.com/education/certify/

IBM 웹 사이트에서 기술 전문 인증 프로그램은 DB2를 포함하여 다른 IBM 제품의 기술 전문 인증 테스트 정보를 제공합니다.

ftp.software.ibm.com

anonymous로 로그온하십시오. /ps/products/db2 디렉토리에서, DB2와 많은 관련 제품에 관한 데이터, 수정사항, 도구 등을 찾을 수 있습니다.

comp.databases.ibm-db2, bit.listserv.db2-l

이러한 인터넷 뉴스 그룹으로 사용자는 DB2 제품에 대한 자신의 사용 경험을 토론할 수 있습니다.

Compuserve에서, GO IBMDB2

이 명령을 입력하여 IBM DB2 계열 포럼을 액세스하십시오. 모든 DB2 제품이 이러한 포럼을 통해 지원됩니다.

미국 외 지역에서 IBM에 연락하는 방법에 관한 정보는 *IBM Software Support Handbook*의 Appendix A를 참조하십시오. 이 문서에 액세스하려면, 웹 사이트 http://www.ibm.com/support/로 가서 페이지 맨 밑에 있는 IBM Software Support Handbook 링크를 누르십시오.

주: 일부 국가에서는 IBM-authorized 공인 딜러는 IBM 지원 센터 대신 해당 딜러 지원 부서에 연락해야 합니다.

IBM 한글 지원에 관한 설문

FAX : (02) 3787-0123

보내 주시는 의견은 더 나은 고객 지원 체제를 위한 귀중한 자료가 됩니다.
독자 여러분의 좋은 의견을 기다립니다.

책 제목: IBM® DB2® Universal Database
관리 안내서: 계획
버전 7

책 번호: SA30-0990-01

성 명		직위/담당업무	
회 사 명		부 서 명	
주 소			
전화번호		팩스번호	
전자우편 주소			
사용중인 시스템	o 중대형 서버 o UNIX 서버 o PC 및 PC 서버		

1. IBM에서 제공하는 한글 책자와 영문 책자 중 어느 것을 더 좋아하십니까? 그 이유는 무엇입니까?
   ☐ 한글 책자 ☐ 영문 책자
   (이유:)

2. 본 책자와 해당 소프트웨어에서 사용된 한글 용어에 대한 귀하의 평가 점수는?
   ☐ 수 ☐ 우 ☐ 미 ☐ 양 ☐ 가

3. 본 책자와 해당 소프트웨어에서 번역 품질에 대한 귀하의 평가 점수는?
   ☐ 수 ☐ 우 ☐ 미 ☐ 양 ☐ 가

4. 본 책자의 인쇄 상태에 대한 귀하의 평가 점수는?
   ☐ 수 ☐ 우 ☐ 미 ☐ 양 ☐ 가

5. 한글 소프트웨어 및 책자가 지원되는 분야에 대해 귀하는 어떻게 생각하십니까?
   ☐ 한글 책자를 늘려야 함 ☐ 현재 수준으로 만족
   ☐ 그다지 필요성을 느끼지 않음

6. IBM은 인쇄물 형식(hardcopy)과 화면 형식(softcopy)의 두 종류로 책자를 제공합니다. 어느 형식을 더 좋아하십니까?
   ☐ 인쇄물 형식(hardcopy) ☐ 화면 형식(softcopy) ☐ 둘 다

☜ IBM 한글 지원 서비스에 대해 기타 제안사항이 있으시면 적어주십시오.

☺ 설문에 답해 주셔서 감사합니다.
귀하의 의견은 저희에게 매우 소중한 것이며, 고객 여러분들께 보다 좋은 제품을 제공해 드리기 위해 최선을 다하겠습니다.

SA30-0990-01

Spine information:

IBM® DB2® Universal Database

관리 안내서: 계획

버전 7