IBM® DB2 Universal Database™

Visual Explain 자습서

버전 8

IBM® DB2 Universal Database™

Visual Explain 자습서

버전 8

이 정보 및 이 정보가 지원하는 제품을 사용하기 전에 반드시 *주의사항*에 나와 있는 일반 정보를 읽으십시오.

IBM 서적을 주문하려면 온라인을 통하거나 한국 IBM 담당자에게 문의하십시오.

- 책을 온라인으로 주문하려면 IBM Publications Center(www.ibm.com/shop/publications/order)를 방문하십시오.
- 한국 IBM 담당자에게 문의하려면 IBM Directory of Worldwide Contacts(www.ibm.com/planetwide)를 방문하십시오.

미국이나 캐나다의 DB2 마케팅 및 판매 부서에서 DB2 책을 주문하려면 1-800-IBM-4YOU(426-4968)로 전화하십시오.

IBM에 정보를 보내는 경우, IBM은 귀하의 권리를 침해하지 않는 범위 내에서 적절하다고 생각하는 방식으로 귀하가 제공한 정보를 사용하거나 배포할 수 있습니다.

목차

이 자습서 정보

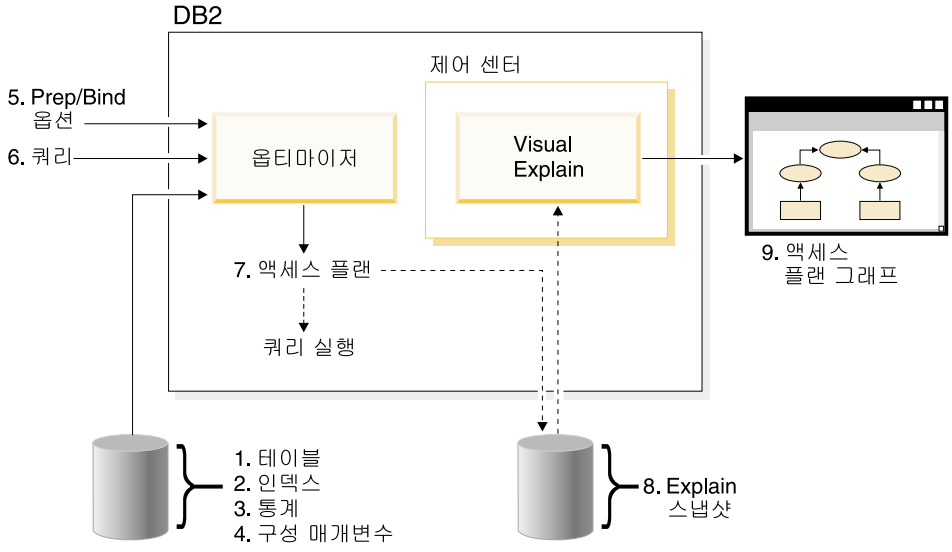
이 자습서는 DB2 Visual Explain의 기능에 대한 안내서를 제공합니다. 이 자습서의 레슨을 완료함으로써, Visual Explain을 사용하여 Explain된 SQL문에 대한 액세스 플랜을 그래프로 볼 수 있는 방법을 학습하게 됩니다. 또한, 보다 나은 성능을 위해 그래프에서 사용 가능한 정보를 사용하여 SQL 쿼리를 조정하는 방법에 대해서도 배우게 됩니다.

DB2에서는 해당 옵티마이저를 사용하여 SQL 쿼리를 검사하고 데이터에 액세스하는 최상의 방법을 판별합니다. 데이터에 대한 이러한 경로를 액세스 플랜이라고 합니다. DB2에서는 특정 SQL 쿼리를 수행하기 위해 선택한 액세스 플랜을 볼 수 있게 하여 옵티마이저가 수행한 내용을 알 수 있습니다. Visual Explain을 사용하여 액세스 플랜을 그래프로 표시할 수 있습니다. 그래프는 쿼리에 관련된 데이터베이스 오브젝트(예: 테이블 및 인덱스)의 비주얼 프리젠테이션입니다. 또한 해당 오브젝트에 대해 수행되는 조작(예: 스캔 및 정렬)도 포함되어 있으며 데이터의 흐름도 보여줍니다.

다음 조정 활동 중 하나 또는 모두를 수행하여 데이터에 대한 쿼리 액세스를 향상시킬 수 있습니다.

1. 테이블 설계 조정 및 테이블 데이터 재구성.
2. 적절한 인덱스 작성.
3. **runstats** 명령을 사용하여 옵티마이저에 현재 통계 제공.
4. 적절한 구성 매개변수 선택.
5. 적절한 바인드 옵션 선택.
6. 필수 데이터만 검색하도록 쿼리 설계.
7. 액세스 플랜에 대한 작업.
8. Explain 스냅샷 작성.
9. 액세스 플랜을 향상하기 위해 액세스 플랜 그래프 사용.

이러한 성능 관련 활동은 다음 그림에 나타난 사항에 해당됩니다(점선은 Visual Explain에 필요한 조치를 나타냅니다).



이 자습서에는 다음에 관한 레슨이 포함되어 있습니다.

- Explain 스냅샷 작성. 이것은 액세스 플랜 그래프 표시를 위한 요구사항입니다.
- 액세스 플랜 그래프 표시 및 조작.
- 성능 조정 활동 수행 및 액세스 플랜 향상 방법 설명.

주: 성능 조정은 단일 파티션 데이터베이스 환경에 대한 레슨과 파티션된 데이터베이스 환경에 대한 레슨으로 나뉘어져 있습니다.

레슨을 통해 작업하기 위해서 DB2 제공 샘플 데이터베이스를 사용하게 됩니다. 샘플 데이터베이스를 미리 작성하지 않은 경우, *관리 안내서*를 참조하십시오.

환경별 정보

이 아이콘으로 표시되는 정보는 단일 파티션 데이터베이스 환경에만 해당됩니다.

이 아이콘으로 표시되는 정보는 파티션된 데이터베이스 환경에만 해당됩니다.

레슨 1. Explain 스냅샷 작성

이 레슨에서는 Explain 스냅샷을 작성하게 됩니다. SQL Explain 기능은 정적 또는 동적 SQL문이 컴파일되는 환경에 대한 정보를 캡처하는 데 사용됩니다. 캡처된 정보를 사용하여 SQL문의 구조와 잠재적인 실행 성능을 이해할 수 있습니다. Explain 스냅샷은 SQL문이 Explain될 때 수집되는 압축 정보입니다. 이 정보는 2진 대형 오브젝트(BLOB)로서 EXPLAIN_STATEMENT 테이블에 저장되며, 다음과 같은 정보를 포함합니다.

- 액세스 플랜의 내부 표시로 해당 연산자와 액세스된 테이블 및 인덱스가 포함됩니다.
- 옵티마이저가 사용하는 결정 기준으로 데이터베이스 오브젝트에 대한 통계와 각 조작에 대한 누적 비용이 포함됩니다.

액세스 플랜 그래프를 표시하기 위해 Visual Explain에서는 Explain 스냅샷에 포함된 정보가 필요합니다.

Explain 테이블 작성

Explain 스냅샷을 작성하려면 다음 Explain 테이블이 사용자 ID에 존재하는지 확인해야 합니다.

- EXPLAIN_INSTANCE
- EXPLAIN_STATEMENT

존재 여부를 검사하려면 **DB2 list tables** 명령을 사용하십시오. 이러한 테이블이 없으면, 다음 지시사항을 따라 이러한 테이블을 작성해야 합니다.

1. DB2가 아직 시작되지 않았다면, **db2start** 명령을 발행하십시오.
2. DB2 CLP 프롬프트에서 사용하려는 데이터베이스로 연결하십시오. 이 자습서를 위해 **connect to sample** 명령을 사용하여 샘플 데이터베이스에 연결하십시오.
3. EXPLAIN.DDL 파일에 제공된 샘플 명령 파일을 사용하여 Explain 테이블을 작성하십시오. 이 파일은 sqllib\misc 디렉토리에 있습니다. 명령 파일을 실

행하려면, 이 디렉토리로 가서 **db2 -tf EXPLAIN.DDL** 명령을 발행하십시오. 이 명령 파일은 연결된 사용자 ID가 접두어로 붙은 Explain 테이블을 작성합니다. 이 사용자 ID는 데이터베이스에 대한 CREATETAB 특권 또는 SYSADM이나 DBADM 권한을 가져야 합니다.

Explain 스냅샷 사용

Visual Explain에 대해 학습하는 데 도움을 주기 위해 네 가지 샘플 스냅샷이 제공됩니다. 사용자 소유의 스냅샷 작성에 대한 정보는 다음 절에 나와 있으나 이 자습서를 위해 사용자 소유의 스냅샷을 작성할 필요는 없습니다.

- 동적 SQL문에 대한 Explain 스냅샷 작성
- 정적 SQL문에 대한 Explain 스냅샷 작성

샘플 스냅샷에 사용된 쿼리는 가장 많은 급여를 받는 관리자 급여의 90%를 초과하여 버는 관리자가 아닌 모든 사원의 이름, 부서 및 급여를 나열합니다.

```
SELECT S.ID,S.NAME,O.DEPTNAME,SALARY+COMM
FROM ORG O, STAFF S
  WHERE
  O.DEPTNUMB = S.DEPT AND
  S.JOB <> 'Mgr' AND
  S.SALARY+S.COMM > ALL( SELECT ST.SALARY*.9
                         FROM STAFF ST
                         WHERE ST.JOB='Mgr' )
ORDER BY S.NAME
```

쿼리는 다음의 두 부분으로 구성됩니다.

1. 서브쿼리(괄호 안의)는 각 관리자 급여의 90%를 구성하는 데이터 행을 생성합니다. 서브쿼리는 ALL로 규정되므로 이 테이블에서 가장 큰 값만이 검색됩니다.
2. 기본 쿼리는 부서 번호가 동일하고 JOB이 'Mgr'이 아니고 (급여+수당)이 서브쿼리에서 리턴된 값보다 큰 모든 행을 ORG 및 STAFF 테이블에서 조인합니다.

기본 쿼리에는 다음의 세 가지 술어(비교)가 포함되어 있습니다.

```
1.   O.DEPTNUMB = S.DEPT
2.   S.JOB <> 'Mgr'
3.   S.SALARY+S.COMM > ALL ( SELECT ST.SALARY*.9
                             FROM STAFF ST
                             WHERE ST.JOB='Mgr' )
```

이러한 술어는 각각 다음을 나타냅니다.

1. 부서 번호가 동일한 ORG 및 STAFF 테이블을 조인하는 Join 술어.
2. STAFF 테이블의 JOB 컬럼의 로컬 술어.
3. 서브쿼리 결과를 사용하는 STAFF 테이블의 SALARY 및 COMM 컬럼에 있는 로컬 술어.

샘플 스냅샷을 로드하려면, 다음을 수행하십시오.

1. DB2가 아직 시작되지 않았다면, **db2start** 명령을 발행하십시오.
2. Explain 테이블이 데이터베이스에 존재하는지 확인하십시오. 이렇게 하려면, Explain 테이블 작성에 있는 지시사항을 따르십시오.
3. 사용하려는 데이터베이스에 연결하십시오. 이 자습서의 경우, 샘플 데이터베이스에 연결하게 됩니다. 샘플 데이터베이스에 연결하려면, DB2 CLP 프롬프트에서 **connect to sample** 명령을 발행하십시오.

   미리 작성되어 있지 않은 경우 *관리 안내서*의 샘플 데이터베이스 설치에 관한 절을 참조하십시오.
4. 미리 정의된 스냅샷을 임포트하려면, DB2 명령 파일 VESAMPL.DDL을 실행하십시오.

   -  이 파일은 sqllib\samples\ve 디렉토리에 있습니다.
   -  이 파일은 sqllib\samples\ve\inter 디렉토리에 있습니다.

   명령 파일을 실행하려면, 이 디렉토리로 가서 **db2 -tf vesampl.ddl** 명령을 발행하십시오.

   - 이 명령 파일은 Explain 테이블을 작성하는 데 사용된 것과 동일한 사용자 ID를 사용해서 실행되어야 합니다.

- 이 명령 파일은 미리 정의된 스냅샷만 임포트합니다. 테이블이나 데이터를 작성하지 않습니다. 나중에 설명되는 성능 조정 활동(예: CREATE INDEX 및 runstats)은 샘플 데이터베이스의 테이블 및 데이터에서 실행됩니다.

이제 액세스 플랜 그래프를 표시하고 사용할 준비가 되었습니다.

동적 SQL문에 대한 Explain 스냅샷 작성

주: 사용자 참조를 위해 이 절에 Explain 스냅샷 작성 정보가 제공됩니다. 샘플 Explain 스냅샷과 함께 제공되므로 자습서를 통해 작업하기 위해 이 태스크를 완료할 필요는 없습니다.

동적 SQL문에 대한 Explain 스냅샷을 작성하려면 다음 단계를 수행하십시오.

1. DB2가 아직 시작되지 않았다면, **db2start** 명령을 발행하십시오.
2. Explain 테이블이 데이터베이스에 존재하는지 확인하십시오. 이렇게 하려면, Explain 테이블 작성에 있는 지시사항을 따르십시오.
3. DB2 CLP 프롬프트에서 사용하려는 데이터베이스로 연결하십시오. 예를 들어 샘플 데이터베이스에 연결하려면, **connect to sample** 명령을 발행하십시오.

   샘플 데이터베이스를 작성하려면, *관리 안내서*의 샘플 데이터베이스 설치에 관한 절을 참조하십시오.
4. DB2 CLP 프롬프트에서 다음 명령 중 하나를 사용하여 동적 SQL문에 대한 Explain 스냅샷을 작성하십시오.
   - SQL문을 실행하지 않고 Explain 스냅샷을 작성하려면, **set current explain snapshot=explain** 명령을 발행하십시오.
   - Explain 스냅샷을 작성하고 SQL문을 실행하려면, **set current explain snapshot=yes** 명령을 발행하십시오.

   이 명령은 Explain 특수 레지스터를 설정합니다. 일단 설정되면, 모든 후속 SQL 문은 영향을 받습니다. 자세한 정보는 *SQL 참조서*의 현재 Explain 스냅샷에 관한 절을 참조하십시오.
5. DB2 CLP 프롬프트에서 SQL문을 제출하십시오.
6. 스냅샷의 액세스 플랜 그래프를 보려면 Explain된 명령문 실행기록 창(제어 센터에서 사용 가능)을 새로 고친 후 스냅샷을 더블 클릭하십시오.

7. 선택적: 스냅샷 기능 작동을 해제하려면, SQL문을 제출한 후에 **set current explain snapshot=no** 명령을 발행하십시오.

정적 SQL문에 대한 Explain 스냅샷 작성

주: 사용자 참조를 위해 이 절에 Explain 스냅샷 작성 정보가 제공됩니다. 샘플 Explain 스냅샷과 함께 제공되므로 자습서를 통해 작업하기 위해 이 태스크를 완료할 필요는 없습니다.

정적 SQL문에 대한 Explain 스냅샷을 작성하려면, 다음 단계를 수행하십시오.

1. DB2가 아직 시작되지 않았다면, **db2start** 명령을 발행하십시오.
2. Explain 테이블이 데이터베이스에 존재하는지 확인하십시오. 이렇게 하려면, Explain 테이블 작성에 있는 지시사항을 따르십시오.
3. DB2 CLP 프롬프트에서 사용하려는 데이터베이스로 연결하십시오. 예를 들어 샘플 데이터베이스에 연결하려면, **connect to sample** 명령을 발행하십시오.
4. 응용프로그램을 바인드하거나 준비할 때 EXPLSNAP 옵션을 사용하여 정적 SQL문에 대한 Explain 스냅샷을 작성하십시오. 예를 들면, **bind** *your file* **explsnap yes** 명령을 발행하십시오.
5. 선택적: 스냅샷의 액세스 플랜 그래프를 보려면, Explain된 명령문 실행기록 창(제어 센터에서 사용 가능)을 새로 고친 후 스냅샷을 더블 클릭하십시오.

대응하는 API에 대해 EXPLSNAP 옵션을 사용하는 방법에 대해서는 *응용프로그램 개발 안내서*에서 이들 각각에 대한 절을 참조하십시오.

다음 레슨 내용

7 페이지의 『레슨 2. 액세스 플랜 그래프 표시 및 사용』에서는 액세스 플랜 그래프를 보고 그 내용을 이해하는 방법에 대해 배우게 됩니다.

레슨 2. 액세스 플랜 그래프 표시 및 사용

이 레슨에서는 액세스 플랜 그래프를 표시하고 사용하기 위해 액세스 플랜 그래프 창을 사용하게 됩니다. 액세스 플랜 그래프는 액세스 플랜의 그래픽 표현입니다. 여기에서 다음에 대한 세부사항을 볼 수 있습니다.

- 테이블(및 그와 관련된 컬럼)과 인덱스
- 연산자(예: 테이블 스캔, 정렬 및 조인)
- 테이블 스페이스 및 함수

다음을 수행하여 액세스 플랜 그래프를 표시할 수 있습니다.

- 이전에 Explain된 명령문 목록에서 선택.
- 패키지의 Explain 가능한 명령문 목록에서 선택.
- SQL문을 동적으로 Explain.

레슨 1에서 로드한 샘플 Explain 스냅샷의 액세스 플랜 그래프로 작업할 것이므로, 이전에 Explain된 명령문 목록에서 선택하게 됩니다. 액세스 플랜 그래프를 표시하는 다른 메소드에 대한 정보는 Visual Explain 도움말을 참조하십시오.

이전에 Explain된 SQL문 목록에서 선택하여 액세스 플랜 그래프 표시

이전에 Explain된 명령문 목록에서 선택하여 액세스 플랜 그래프를 표시하려면, 다음을 수행하십시오.

1. 제어 센터에서 샘플 데이터베이스를 찾을 때까지 오브젝트 트리를 펼치십시오.
2. 데이터베이스를 마우스 오른쪽 단추로 누른 후 팝업 메뉴에서 **Explain된 명령문 실행기록 표시**를 선택하십시오. Explain된 명령문 실행기록 창이 열립니다.
3. Explain 스냅샷이 있는 명령문에 대한 액세스 플랜 그래프는 표시만 할 수 있습니다. 규정된 명령문에는 **Explain 스냅샷** 컬럼에 YES 항목이 있습니다. 쿼리 번호 1로 식별되는 항목을 더블 클릭하십시오(**쿼리 번호** 컬럼을 찾기 위해 오른쪽으로 스크롤해야 할 수도 있습니다). 명령문에 대한 액세스 플랜 그래프 창이 열립니다.

주: 그래프는 아래쪽에서 위쪽으로 읽혀집니다. 쿼리의 첫 단계는 그래프의 맨 아래에 나열되며 마지막 단계는 맨 위에 나열됩니다.

액세스 플랜 그래프에서 기호 읽기

액세스 플랜 그래프는 액세스 플랜 구조를 트리로 보여줍니다. 트리의 노드는 다음을 나타냅니다.

- 테이블: 직사각형
- 인덱스: 다이아몬드꼴
- 연산자: 8각형. TQUEUE 연산자: 평행사변형
- 테이블 함수: 6각형.

연산자의 경우, 연산자 유형 오른쪽 대괄호 안의 수는 각 노드에 대해 고유한 ID입니다. 연산자 유형 아래의 수는 누적 비용입니다.

확대/축소 슬라이더를 사용하여 그래프의 일부 확대

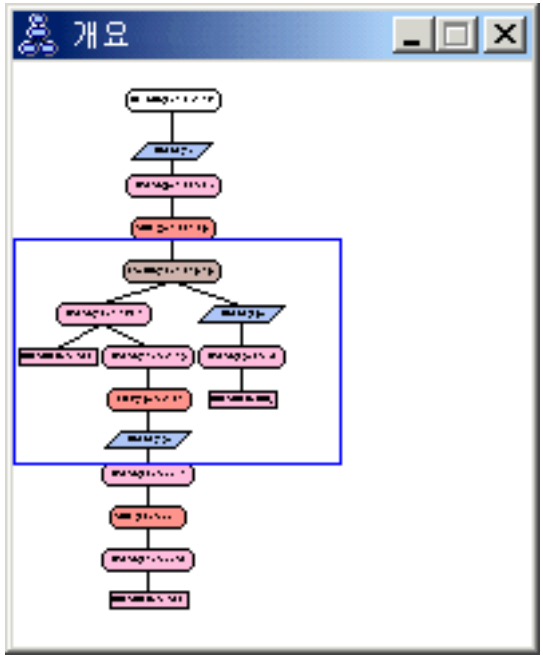
액세스 플랜 그래프를 표시할 때, 전체 그래프가 표시되므로 각 노드를 구별하는 세부사항이 표시되지 않을 수도 있습니다.

액세스 플랜 그래프 창에서 **확대/축소 슬라이더**를 사용하여 그래프의 일부를 확대하십시오.

1. 그래프 왼쪽의 확대/축소 슬라이더 막대에서 작은 스크롤 상자 위에 마우스 포인터를 놓으십시오.
2. 그래프가 원하는 확대 레벨에 올 때까지 마우스 오른쪽 단추를 누른 상태에서 슬라이더를 끄십시오.

그래프의 다른 부분을 보려면, 스크롤 막대를 사용하십시오.

크고 복잡한 액세스 플랜 그래프를 보려면, 그래프 개요 창을 사용하십시오. 이 창을 사용하여 보고 있는 그래프의 일부를 표시하거나 그래프를 확대하거나 스크롤할 수 있습니다. 확대/축소 상자의 섹션은 액세스 플랜에 표시됩니다.



그래프를 스크롤하려면, 그래프 개요 창에서 강조표시된 영역 위에 마우스 포인터를 놓고 마우스 왼쪽 단추를 누른 상태에서 원하는 액세스 플랜 그래프 부분이 표시될 때까지 마우스를 이동시키십시오.

그래프에서 오브젝트에 대한 세부사항 가져오기

액세스 플랜 그래프에서 오브젝트에 대한 자세한 정보에 액세스할 수 있습니다. 다음을 표시할 수 있습니다.

- 다음과 같은 오브젝트의 시스템 카탈로그 통계
  - 테이블, 인덱스 또는 테이블 함수
  - 비용, 등록 정보 및 입력 인수와 같은 연산자에 대한 정보
  - 내장 함수 또는 사용자 정의 함수
  - 테이블 스페이스
  - SQL문에서 참조된 컬럼
- 구성 매개변수 및 바인드 옵션에 대한 정보(최적화 매개변수).

테이블, 인덱스 및 테이블 함수에 대한 통계 가져오기

그래프에서 단일 테이블(직사각형), 인덱스(다이아몬드) 또는 테이블 함수(6각형)에 대한 카탈로그 통계를 보려면, 해당 노드를 더블 클릭하십시오. 시스템 카탈로그 테이블에 현재 존재하는 정보뿐 아니라 스냅샷 작성시 적용된 통계에 대한 정보도 표시하는 통계 창이 선택된 오브젝트에 대해 열립니다.

그래프에서 *여러* 테이블, 인덱스 또는 테이블 함수에 대한 카탈로그 통계를 보려면, 해당 항목(강조표시된)을 눌러 각각 선택한 후 **노드 -> 통계 표시**를 선택하십시오. 선택된 오브젝트마다 통계 창이 열립니다(창은 스택될 수 있으며 이들에 모두 액세스하기 위해서는 일부 끌어서 놓기가 필요할 수 있습니다).

Explain 컬럼의 **STATS_TIME** 항목에 **통계 갱신 안됨** 항목이 포함되면, 옵티마이저가 액세스 플랜을 작성할 때 통계가 존재하지는 않았습니다. 따라서 옵티마이저가 액세스 플랜을 작성하는 데 특정 통계를 필요로 하는 경우, 디폴트가 사용됩니다. 디폴트 통계가 옵티마이저에서 사용되었으면, Explain 컬럼에서 **(디폴트)**로 식별됩니다.

그래프에서 연산자에 대한 세부사항 가져오기

단일 연산자(8각형)의 카탈로그 통계를 보려면, 해당 노드를 더블 클릭하십시오. 선택한 연산자에 대해 다음과 같은 정보를 표시하는 연산자 세부사항 창이 열립니다.

- 계산된 누적 비용(I/O, CPU 지시사항 및 전체 비용)
- 지금까지의 카디낼리티(cardinality)(즉, 검색된 예상 행 수)
- 플랜에서 지금까지 액세스되고 조인된 테이블
- 지금까지 액세스된 테이블의 컬럼
- 예상된 선택성을 포함하여 지금까지 적용된 술어
- 각 연산자에 대한 입력 인수

여러 연산자의 세부사항을 보려면, 해당 항목(강조표시된)을 눌러 각각 선택한 후 **노드 -> 세부사항 표시**를 선택하십시오. 선택된 오브젝트마다 통계 창이 열립니다. (창은 스택될 수 있으며 이들에 모두 액세스하기 위해서는 일부 끌어서 놓기가 필요할 수 있습니다.)

함수에 대한 통계 가져오기

내장 함수 및 사용자 정의 함수에 대한 카탈로그 통계를 보려면, **명령문 -> 통계 표시 -> 함수**를 선택하십시오. 함수 창에 표시된 목록에서 하나 이상의 항목을 선택한 후 **확인**을 누르십시오. 선택한 함수마다 함수 통계 창이 열립니다.

테이블 스페이스에 대한 통계 가져오기

테이블 스페이스에 대한 카탈로그 통계를 보려면, **명령문 -> 통계 표시 -> 테이블 스페이스**를 선택하십시오. 테이블 스페이스 창에 표시된 목록에서 하나 이상의 항목을 선택한 후 **확인**을 누르십시오. 선택한 테이블 스페이스마다 테이블 스페이스 통계 창이 열립니다.

SQL문의 컬럼에 대한 통계 가져오기

SQL문에서 참조된 컬럼에 대한 통계를 가져오려면, 다음을 수행하십시오.

1. 액세스 플랜 그래프의 테이블을 더블 클릭하십시오. 테이블 통계 창이 열립니다.
2. **참조된 컬럼** 누름 단추를 누르십시오. 참조된 컬럼 창이 열리고 테이블의 컬럼이 나열됩니다.
3. 목록에서 하나 이상의 컬럼을 선택한 후 **확인**을 누르십시오. 선택한 컬럼마다 참조된 컬럼 통계 창이 열립니다.

구성 매개변수 및 바인드 옵션에 대한 정보 가져오기

구성 매개변수 및 바인드 옵션에 대한 정보(최적화 매개변수)를 보려면, 액세스 플랜 그래프 창에서 **명령문 -> 최적화 매개변수 표시**를 선택하십시오. 현재 값뿐 아니라 스냅샷 작성시 적용된 매개변수 값에 대한 정보도 표시하는 최적화 매개변수 창이 열립니다.

그래프 모양 변경

그래프가 표시되는 방법에 대한 다양한 특성을 변경하려면, 다음을 수행하십시오.

1. 액세스 플랜 그래프 창에서 **보기 -> 설정**을 선택하십시오. 액세스 플랜 그래프 설정 노트북이 열립니다.
2. 배경색을 변경하려면, 그래프 탭을 선택하십시오.

3. 다양한 연산자 색상을 변경하려면, 기본, 확장, 갱신 및 기타 탭을 사용하십시오.
4. 테이블, 인덱스 또는 테이블 함수 노드의 색상을 변경하려면, 피연산자 탭을 선택하십시오.
5. 연산자 노드에 표시된 정보 유형(지금까지 리턴된 예상 행 수인 비용 또는 카디낼리티(cardinality)의 유형)을 지정하려면, 연산자 탭을 선택하십시오.
6. 테이블 노드에 스키마 이름이 표시되는지 또는 사용자 ID가 표시되는지를 지정하려면, 연산자 탭을 선택하십시오.
7. 노드가 2차원 또는 3차원으로 표시되는지를 지정하려면, 노드 탭을 선택하십시오.
8. 그래프를 선택한 옵션으로 갱신한 후 설정값을 저장하려면, **적용**을 누르십시오.

다음 레슨 내용

단일 파티션 데이터베이스 환경에서 작업 중인 경우, 어떻게 다른 성능 조정 활동이 액세스 플랜을 변경하고 향상시킬 수 있는 지에 대해 배우게 되는 13 페이지의 『레슨 3. 단일 파티션 데이터베이스 환경에서 액세스 플랜 향상』으로 가십시오.

단일 파티션 데이터베이스 환경에서 작업 중인 경우, 어떻게 다른 성능 조정 활동이 액세스 플랜을 변경하고 향상시킬 수 있는 지에 대해 배우게 되는 14 페이지의 『인덱스와 통계 없이 쿼리 실행』으로 가십시오.

레슨 3. 단일 파티션 데이터베이스 환경에서 액세스 플랜 향상

이 레슨에서는 다양한 성능 조정 활동을 수행할 때 기본 쿼리에 대한 액세스 플랜 및 관련 창이 어떻게 변하는지 배우게 됩니다. 그림과 함께 제공되는 일련의 예를 사용하여, **runstats** 명령을 사용하고 해당 인덱스를 추가함으로써 단순 쿼리의 액세스 플랜에 대해 계산된 전체 비용이 향상될 수 있는 방법에 대해 배우게 됩니다.

Visual Explain에 익숙해지게 되면, 쿼리를 조정하는 다른 방법을 발견하게 됩니다.

액세스 플랜 그래프에 대한 작업

예에서처럼 네 개의 샘플 Explain 스냅샷을 사용하여, 어떻게 성능 조정이 데이터베이스 성능의 중요한 부분이 되는지를 배우게 됩니다.

Explain 스냅샷에 연관된 쿼리는 1 - 4로 번호가 지정되어 있습니다. 각 쿼리는 동일한 SQL문(레슨 1에 설명되어 있음)을 사용합니다.

```
SELECT S.ID,S.NAME,O.DEPTNAME,SALARY+COMM
FROM ORG O, STAFF S
  WHERE
  O.DEPTNUMB = S.DEPT AND
  S.JOB <> 'Mgr' AND
  S.SALARY+S.COMM > ALL( SELECT ST.SALARY*.9
                         FROM STAFF ST
                         WHERE ST.JOB='Mgr' )
ORDER BY S.NAME
```

그러나 각 쿼리 반복은 이전 실행보다 더 많은 성능 조정 기법을 사용합니다. 예를 들면, 쿼리 1에는 성능 조정이 없는 반면, 쿼리 4에는 가장 많습니다. 쿼리의 차이점에 대해서는 다음에 설명되어 있습니다.

쿼리 1 인덱스와 통계 없이 쿼리 실행

쿼리 2 쿼리에서 테이블 및 인덱스에 대한 현재 통계 수집

쿼리 3 쿼리에서 테이블을 조인하는 데 사용되는 컬럼에 대한 인덱스 작성

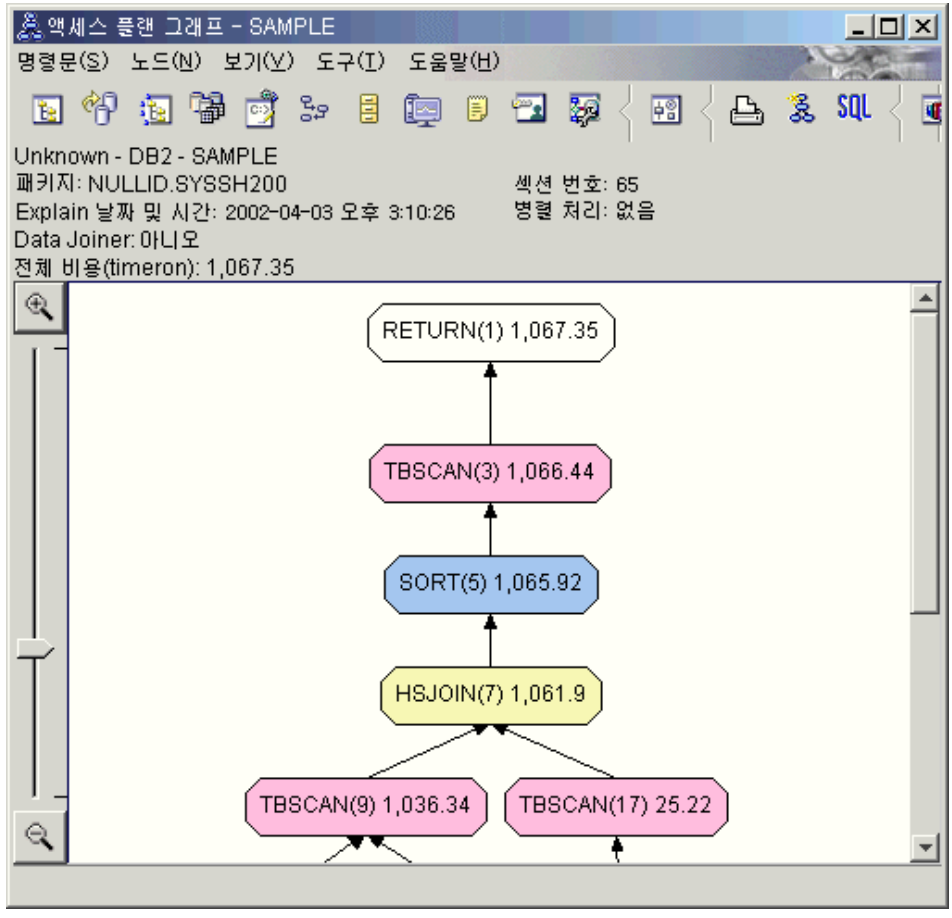
쿼리 4 테이블 컬럼에 추가 인덱스 작성

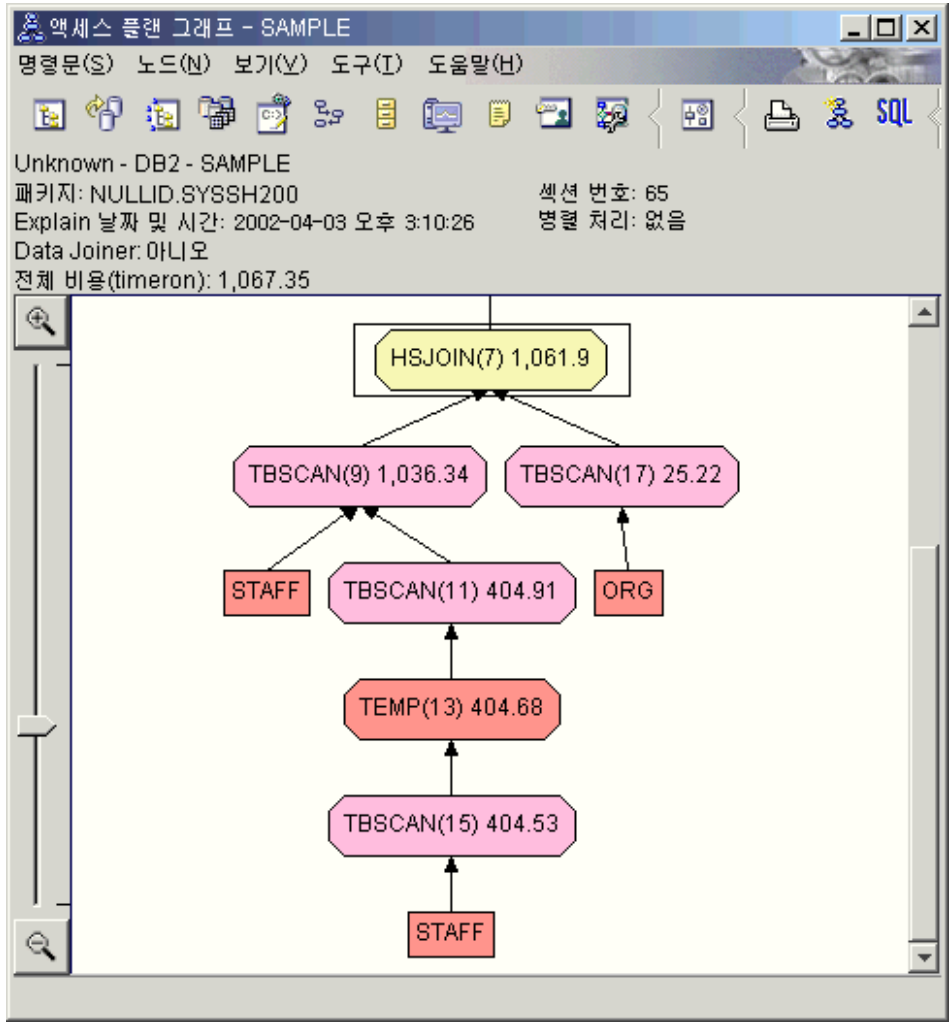
인덱스와 통계 없이 쿼리 실행

예에서 액세스 플랜은 인덱스와 통계 없이 SQL 쿼리에 대해 작성되었습니다.

쿼리(쿼리 1)에 대한 액세스 플랜 그래프를 보려면, 다음을 수행하십시오.

1. 제어 센터에서 샘플 데이터베이스를 찾을 때까지 오브젝트 트리를 펼치십시오.
2. 데이터베이스를 마우스 오른쪽 단추로 누른 후 팝업 메뉴에서 **Explain된 명령문 실행기록 표시**를 선택하십시오. Explain된 명령문 실행기록 창이 열립니다.
3. 쿼리 번호 1로 식별되는 항목을 더블 클릭하십시오(**쿼리 번호** 컬럼을 찾기 위해 오른쪽으로 스크롤해야 할 수 있습니다). 명령문에 대한 액세스 플랜 그래프 창이 열립니다.


액세스 플랜 그래프 - SAMPLE
명령문(S) 노드(N) 보기(V) 도구(T) 도움말(H)
SQL
Unknown - DB2 - SAMPLE
패키지: NULLID.SYSSH200
섹션 번호: 65
Explain 날짜 및 시간: 2002-04-03 오후 3:10:26
병렬 처리: 없음
Data Joiner: 아니오
전체 비용(timeron): 1,067.35
RETURN(1) 1,067.35
TBSCAN(3) 1,066.44
SORT(5) 1,065.92
HSJOIN(7) 1,061.9
TBSCAN(9) 1,036.34
TBSCAN(17) 25.22



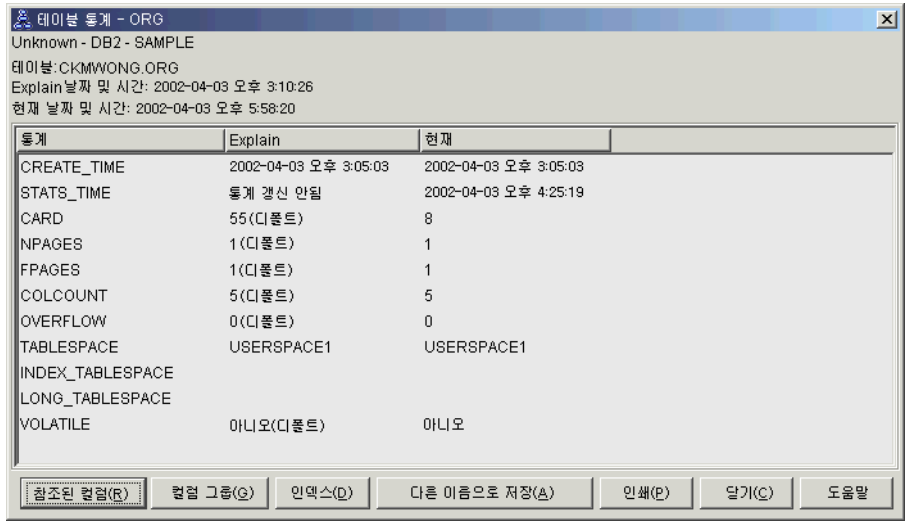
다음 질문에 응답하면 쿼리를 향상시키는 방법을 이해하는 데 도움이 됩니다.

1. 쿼리에 각 테이블에 대한 현재 통계가 존재합니까?

   쿼리에 각 테이블에 대한 현재 통계가 존재하는지 점검하려면, 액세스 플랜 그래프에서 각 테이블 노드를 더블 클릭하십시오. 열린 테이블 통계 창에는 스냅샷 작성시 통계가 수집되지 않은 경우 **Explain** 컬럼 아래의 **STATS_TIME** 행에 "통계 갱신 안됨"이 표시됩니다.

현재 통계가 없으면, 옵티마이저는 실제 통계와는 다를 수 있는 디폴트 통계를 사용합니다. 디폴트 통계는 테이블 통계 창의 **Explain** 컬럼 아래의 "디폴트"라는 단어로 식별됩니다.

ORG 테이블에 대한 테이블 통계 창의 정보에 따르면, 옵티마이저가 디폴트 통계(Explain 값 옆에 표시된 대로)를 사용했습니다. 디폴트 통계는 스냅샷 작성 시 실제 통계를 사용할 수 없으므로 사용되었습니다(**STATS_TIME** 행에 표시된 대로).



2. 이 액세스 플랜이 데이터 액세스에 가장 효과적인 메소드를 사용합니까?

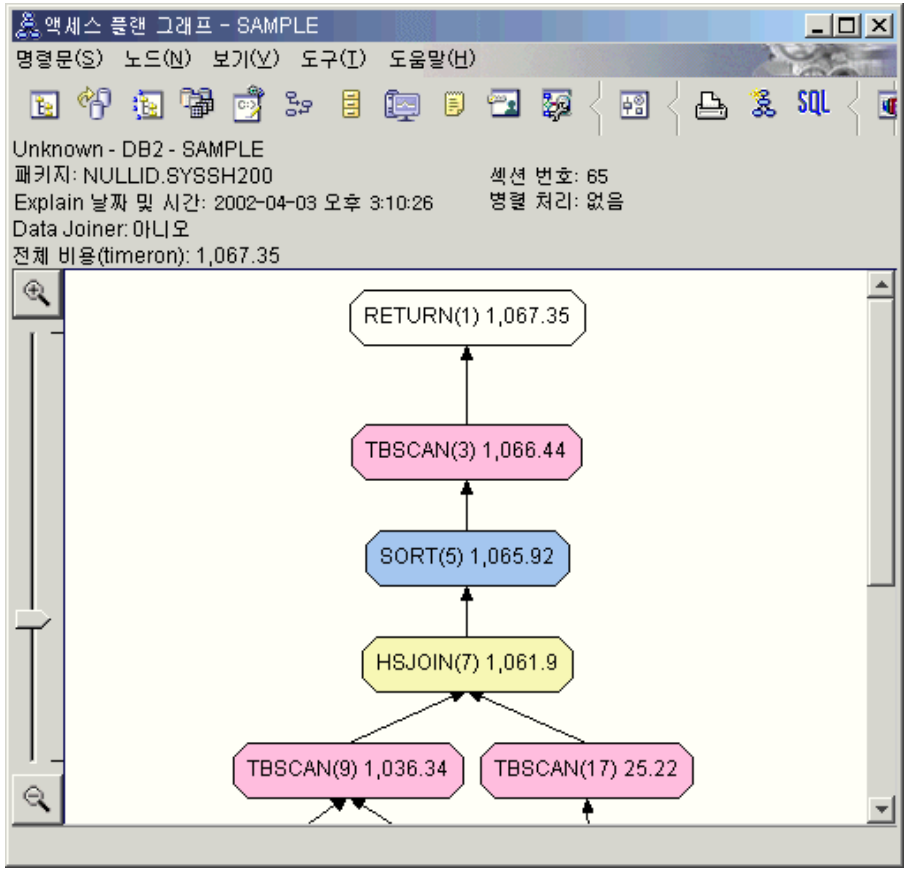
   이 액세스 플랜에는 인덱스 스캔이 아니라 테이블 스캔이 포함되어 있습니다. 테이블 스캔은 8각형으로 표시되며 TBSCAN 레이블이 붙어 있습니다. 인덱스 스캔이 사용되었으면, 이 스캔은 다이아몬드로 표시되며 IXSCAN 레이블이 붙어 있습니다. 소량의 데이터가 발췌되고 있는 경우 테이블에 대해 작성된 인덱스를 사용하는 것이 테이블 스캔보다 비용면에서 효율적입니다.

3. 이 액세스 플랜이 얼마나 효과적입니까?

   실제 통계를 사용하는 경우에만 액세스 플랜의 효과를 판별할 수 있습니다. 옵티마이저가 액세스 플랜의 디폴트 통계를 사용했으므로, 플랜이 얼마나 효과적인지 판별할 수 없습니다.

일반적으로, 나중에 개정된 액세스 플랜과 비교하기 위해 액세스 플랜에 대해 계산된 전체 비용을 기록해 두어야 합니다. 각 노드에 나열된 비용은 쿼리의 첫 단계에서 노드까지(경계 포함) 누적됩니다.

액세스 플랜 그래프 창에서 그래프의 맨 위에 있는 **RETURN (1)**에 표시된 전체 비용은 대략 1,067 timeron입니다. 계산된 전체 비용은 창의 맨 위 영역에도 표시됩니다.



4. 다음 내용은 무엇입니까?

   쿼리 2는 **runstats**가 실행된 후에 기본 쿼리에 대한 액세스 플랜을 조회합니다. **runstats** 명령을 사용하면 쿼리에서 액세스되는 모든 테이블의 현재 통계가 옵티마이저에 제공됩니다.

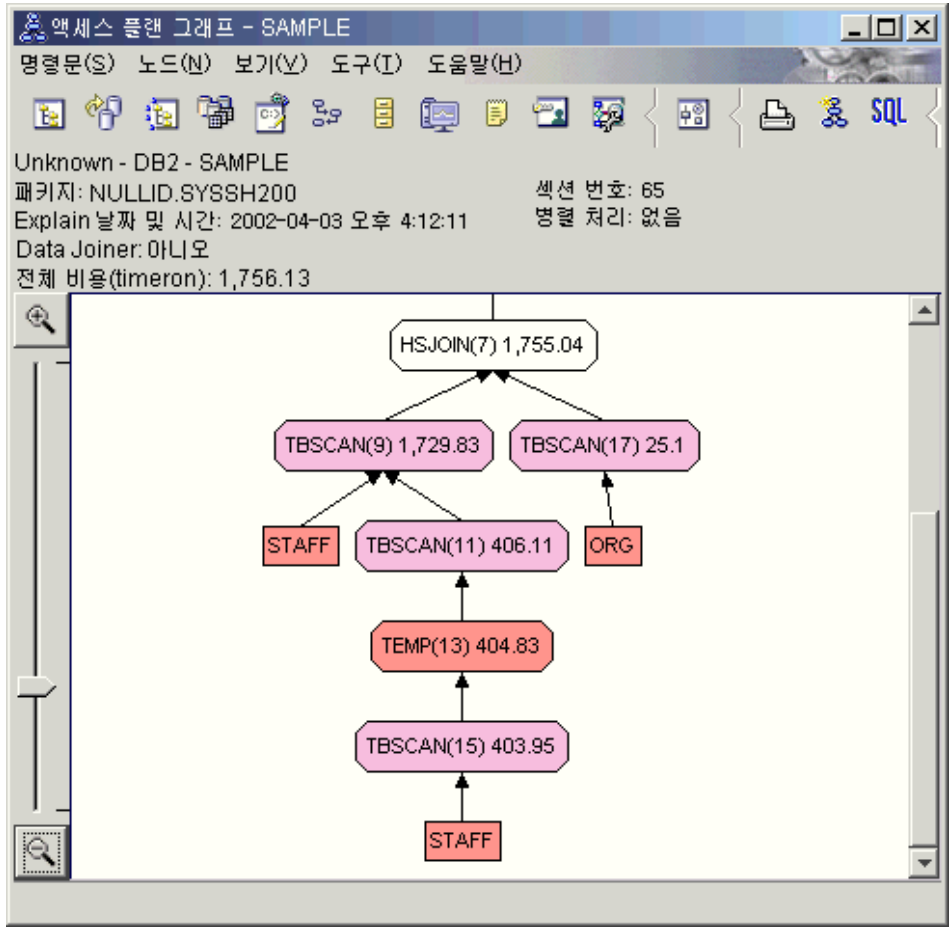
Runstats를 사용하여 테이블 및 인덱스에 대한 현재 통계 수집

이 예는 **runstats** 명령으로 현재 통계를 수집하여 쿼리 1에 설명된 액세스 플랜에 빌드합니다.

runstats 명령을 사용하여 테이블 및 인덱스에 대한 현재 통계를 수집하는 것이 바람직한데, 이는 마지막으로 **runstats** 명령을 실행한 이후 새로운 인덱스가 작성되었거나 중요한 갱신 활동이 발생한 경우에 특히 권장됩니다. 이는 옵티마이저에 최상의 액세스 플랜을 판별할 수 있는 가장 정확한 정보를 제공합니다. 현재 통계를 사용할 수 없으면, 옵티마이저는 정확하지 않은 디폴트 통계에 따라 비효율적인 액세스 플랜을 선택할 수 있습니다.

runstats 명령은 테이블 갱신 후 사용해야 하며, 그렇지 않을 경우 빈 테이블이 옵티마이저에 표시됩니다. 이러한 문제점은 연산자 세부사항 창에 대한 카디널리티가 0인 경우 입증됩니다. 이 경우, 테이블 갱신을 완료하고 **runstats** 명령을 재실행한 후 영향을 받는 테이블에 대한 Explain 스냅샷을 재작성하십시오.

Explain된 명령문 실행기록 창에서 쿼리(쿼리 2)에 대한 액세스 플랜 그래프를 보려면, 쿼리 번호 2로 식별되는 항목을 더블 클릭하십시오. 이 명령문 실행을 위한 액세스 플랜 그래프 창이 열립니다.



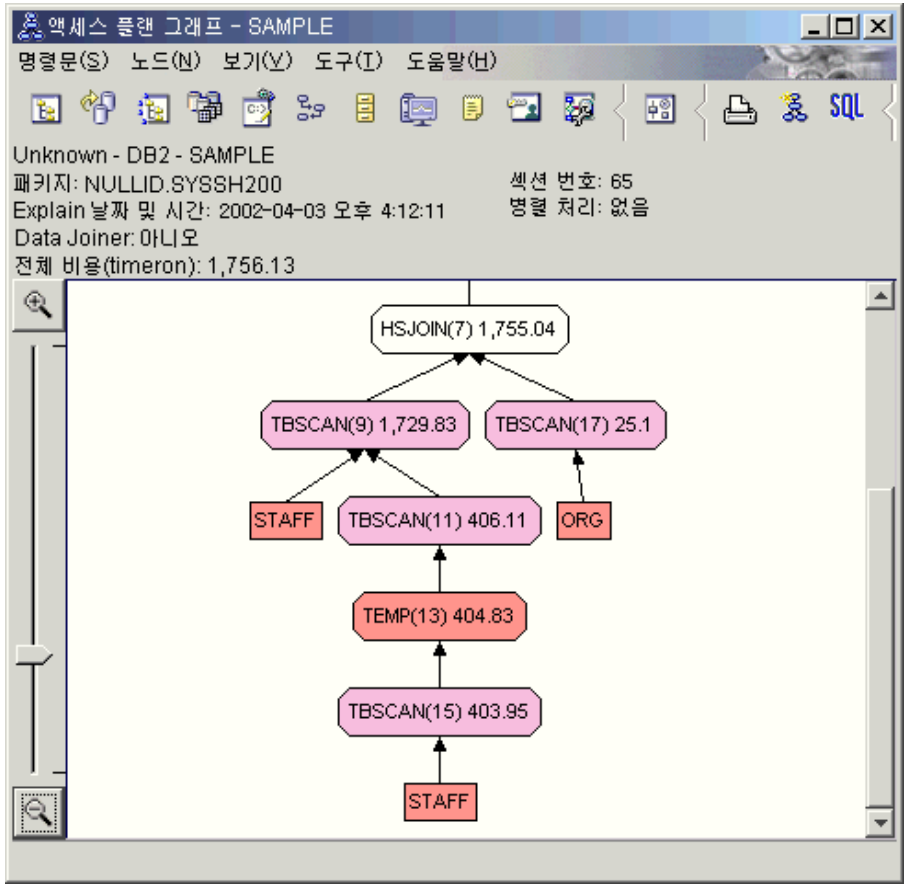
다음 질문에 응답하면 쿼리를 향상시키는 방법을 이해하는 데 도움이 됩니다.

1. 쿼리에 각 테이블에 대한 현재 통계가 존재합니까?

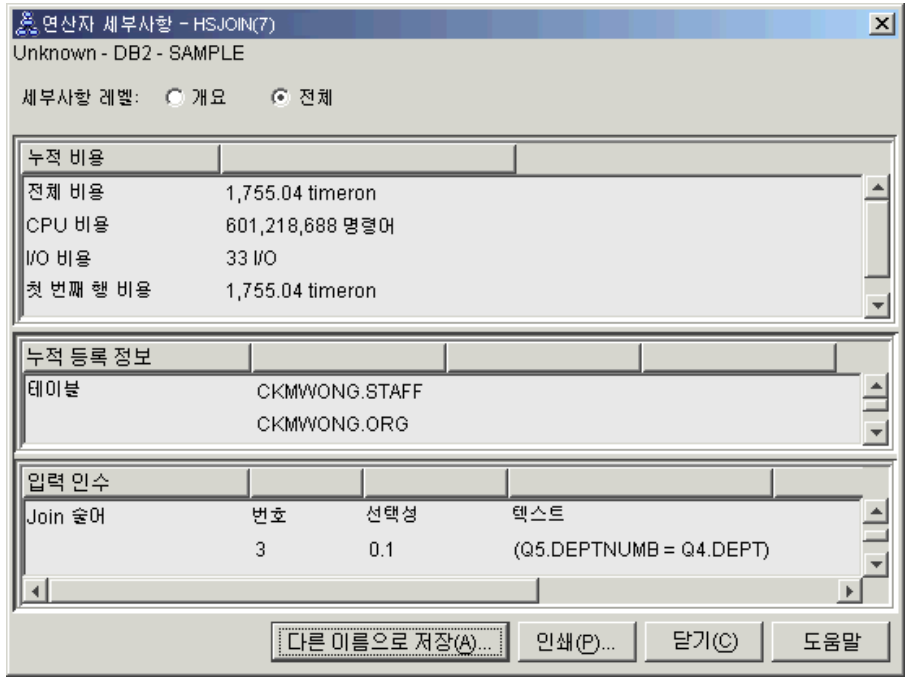
   ORG 테이블에 대한 테이블 통계 창은 옵티마이저가 실제 통계를 사용한다는 것을 보여줍니다(**STATS_TIME** 값은 통계가 수집된 실제 시간임). 통계의 정확성은 **runstats** 명령이 실행된 이후에 테이블의 내용에 현저한 변경사항이 있는지 여부에 따라 다릅니다.

2. 이 액세스 플랜이 데이터 액세스에 가장 효과적인 메소드를 사용합니까?

   쿼리 1과 같이 쿼리 2의 액세스 플랜은 인덱스 스캔(IXSCAN)이 아니라 테이블 스캔(TBSCAN)을 사용합니다. 현재 통계가 존재하는 경우에도 인덱스 스캔은 쿼리에서 사용된 컬럼에 인덱스가 없으므로 수행되지 않았습니다. 쿼리를 향상시키는 한 방법은 테이블을 조인하는 데 사용되는 컬럼(즉, Join 술어에서 사용되는 컬럼)의 인덱스를 옵티마이저에 제공하는 것입니다. 이 예에서는 이것이 처음 병합 스캔 조인 HSJOIN (7)입니다.



HSJOIN (7) 연산자에 대한 연산자 세부사항 창에서 **입력 인수** 아래의 **Join 술어** 섹션을 보십시오. 이 조인 조작에서 사용되는 컬럼은 **텍스트** 컬럼 아래에 나열되어 있습니다. 이 예에서, 이러한 컬럼은 DEPTNUMB 및 DEPT입니다.



3. 이 액세스 플랜이 얼마나 효과적입니까?

   최신 통계에 기초한 액세스 플랜은 항상 실제로 계산된 비용(timeron 단위)을 생성합니다. 쿼리 1에서 계산된 비용은 디폴트 통계에 기초하므로, 어느 것이 더 효과적인지 판별하기 위해 두 개의 액세스 플랜 그래프의 비용을 비교할 수 없습니다. 비용이 높거나 낮은지 여부는 관련되지 않습니다. 유효한 효과 측정을 얻기 위해서는 실제 통계에 따라 액세스 플랜 비용을 비교해야 합니다.

4. 다음 내용은 무엇입니까?

   쿼리 3에서는 DEPTNUMB 및 DEPT 컬럼에서 인덱스 추가 효과를 검색합니다. Join 술어에 사용되는 컬럼에 인덱스 추가는 성능을 향상시킬 수 있습니다.

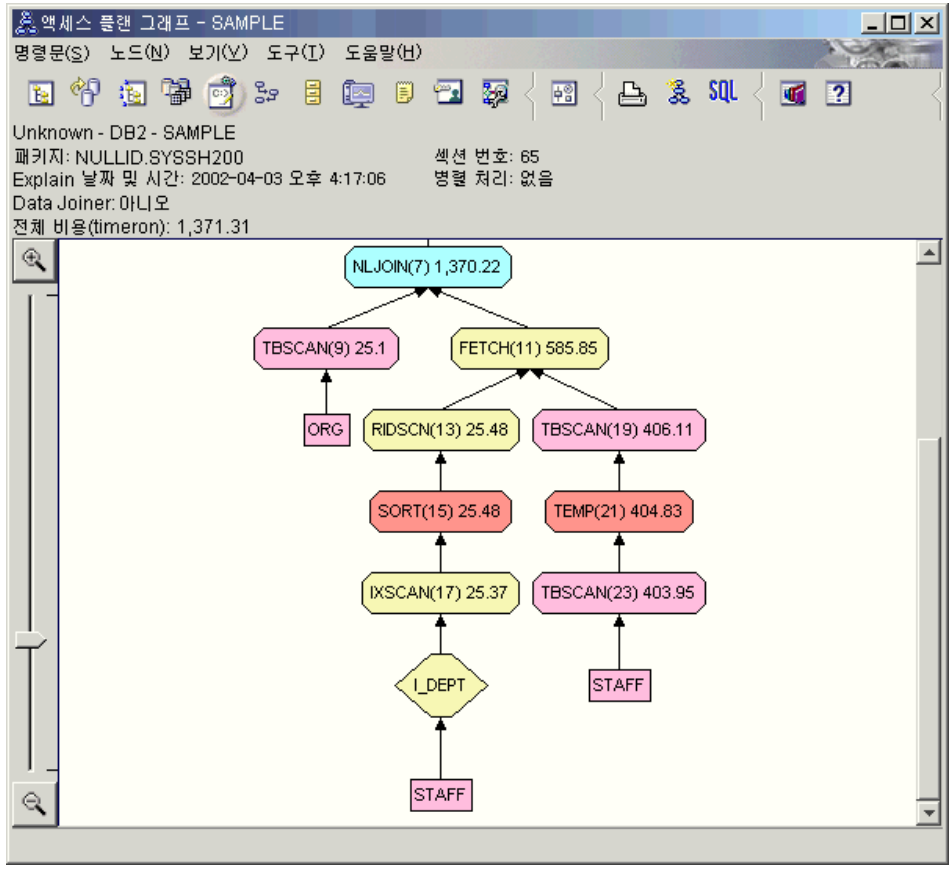
쿼리에서 테이블을 조인하는 데 사용되는 컬럼에 대한 인덱스 작성

이 예는 STAFF 테이블의 DEPT 컬럼과 ORG 테이블의 DEPTNUMB 컬럼에 대한 인덱스를 작성하여 쿼리 2에 설명된 액세스 플랜에서 빌드합니다.

주: 버전 8에서 권장되는 인덱스는 워크로드 성능 마법사를 사용하여 작성될 수 있습니다.

Explain된 명령문 실행기록 창에서 쿼리(쿼리 3)에 대한 액세스 플랜 그래프를 보려면, 쿼리 번호 3으로 식별되는 항목을 더블 클릭하십시오. 이 명령문 실행을 위한 액세스 플랜 그래프 창이 열립니다.

주: DEPTNUM에 대한 인덱스가 작성되었더라도 옵티마이저가 이를 사용하지 않았습니다.

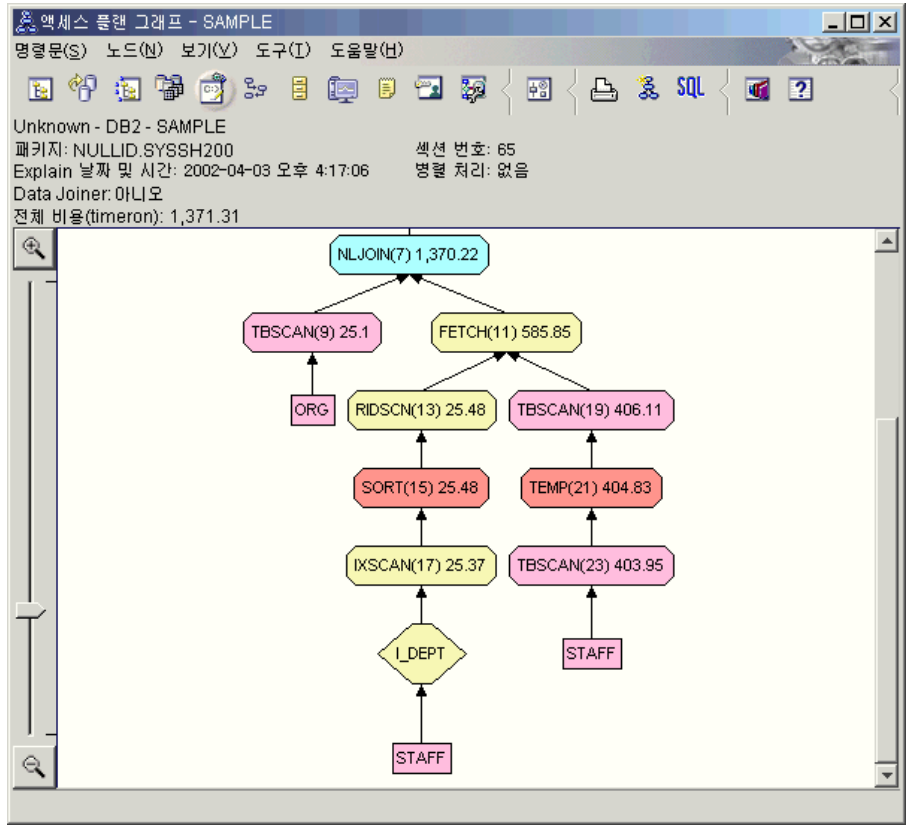


다음 질문에 응답하면 쿼리를 향상시키는 방법을 이해하는 데 도움이 됩니다.

1. 인덱스가 있는 액세스 플랜에서 무엇이 변경되었습니까?

중첩 루프 조인인 NLJOIN (7)은 쿼리 2에서 사용된 병합 스캔 조인 HSJOIN (7)을 대체합니다. 중첩 루프 조인을 사용하면, 이 유형의 조인에는 정렬 또는 임시 테이블이 필요하지 않으므로 병합 스캔보다 비용이 더 낮게 계산됩니다.

새로운 다이아몬드 모양의 노드인 **I_DEPT**가 STAFF 테이블 바로 위에 추가되었습니다. 이 노드는 DEPT에 작성된 인덱스를 나타내며, 검색할 행을 판별하기 위해 옵티마이저가 테이블 스캔 대신에 인덱스 스캔을 사용한다는 것을 보여줍니다.



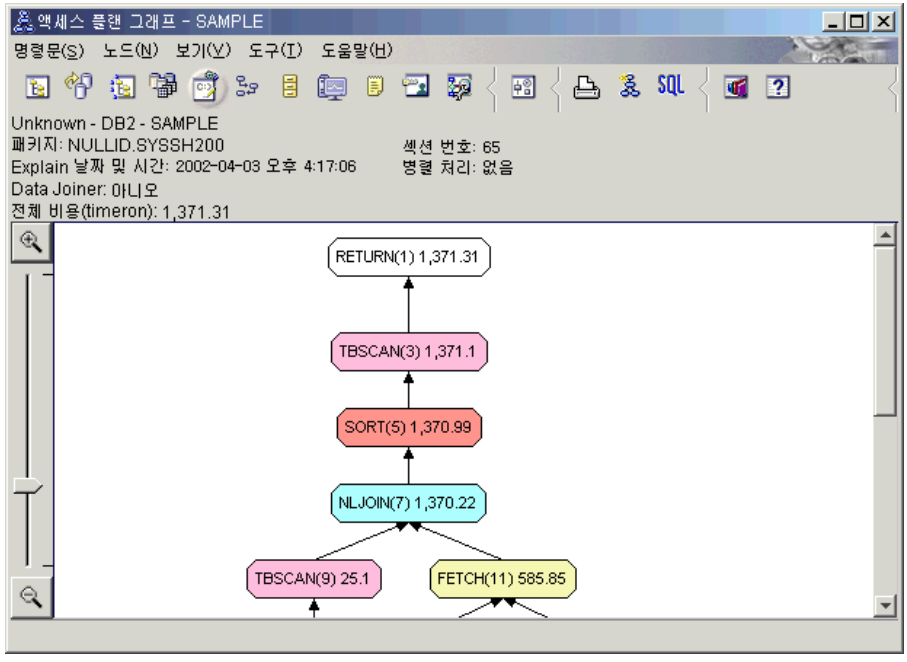
이 부분의 액세스 플랜 그래프에서는 새 인덱스(I_DEPT)가 DEPT 컬럼에 작성되었고 IXSCAN (17)이 STAFF 테이블을 액세스하는 데 사용되었다는 점에 유의하십시오. 쿼리 2에서는 테이블 스캔이 STAFF 테이블을 액세스하는 데 사용되었습니다.

2. 이 액세스 플랜이 데이터 액세스에 가장 효과적인 메소드를 사용합니까?

인덱스 추가 결과로서 IXSCAN 노드인 IXSCAN (17)이 STAFF 테이블에 액세스하는 데 사용되었습니다. 쿼리 2에는 인덱스가 없으므로 예에서 테이블 스캔이 사용되었습니다.

FETCH 노드인 FETCH (11)은 컬럼 DEPT를 검색하기 위해 인덱스 스캔을 사용하는 것외에도 옵티마이저가 포인터로 인덱스를 사용하여 STAFF 테이블에서 추가 컬럼을 검색했음을 보여줍니다. 이 경우, 인덱스 스캔과 페치의 조합은 이전 액세스 플랜에서 사용된 전체 테이블 스캔보다 비용이 덜 드는 것으로 계산되었습니다.

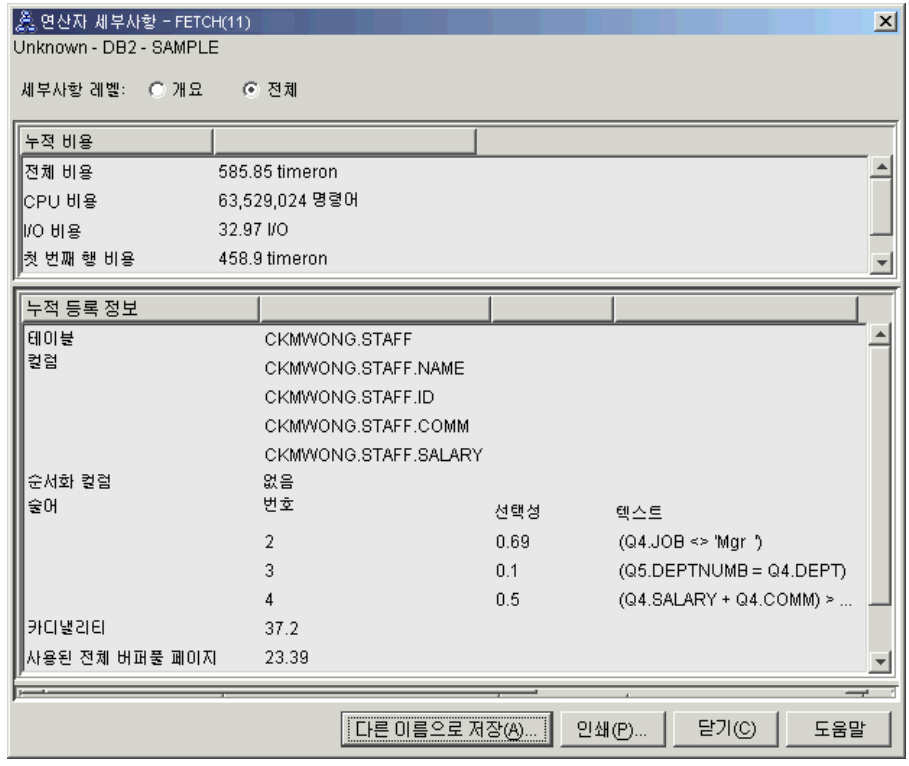
주: STAFF 테이블의 노드는 두 번 표시되어 DEPT에 대한 인덱스 및 FETCH 조작에 대한 관계를 모두 보여줍니다.

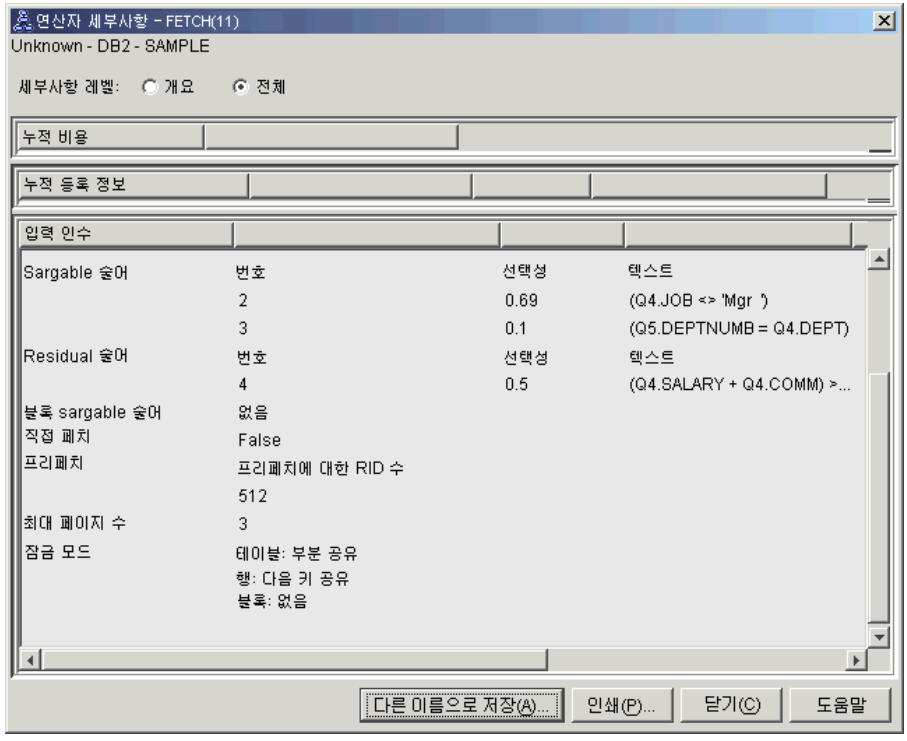


이 조회에 대한 액세스 플랜은 Join 술어에 관련된 컬럼에서의 인덱스 작성 효과를 보여줍니다. 인덱스는 로컬 술어의 응용프로그램 속도를 높일 수도 있습니다. 로컬 술어에서 참조된 컬럼에 인덱스 추가가 액세스 플랜에 어떤 영향을 줄 수 있는지를 알기 위해 이 조회의 각 테이블마다 로컬 술어를 살펴봅시다.

FETCH (11) 연산자에 대한 연산자 세부사항 창에서 **누적 등록 정보** 아래의 컬럼을 보십시오. 이 페치 조작에 대해 술어에 사용된 컬럼은 술어 섹션에 표시된 대로 JOB입니다.

주: 이 술어의 선택성은 .69입니다. 이는 이 술어를 사용하면 행의 69%가 다음 처리에 선택될 것임을 의미합니다.





FETCH (11) 연산자에 대한 연산자 세부사항 창에서는 이 조작에 사용되고 있는 컬럼을 보여줍니다. DEPTNAME이 **입력 인수** 아래에서 **검색된 컬럼** 옆의 첫 행에 나열되어 있다는 것을 알 수 있습니다.

3. 이 액세스 플랜이 얼마나 효과적입니까?

   이 액세스 플랜은 이전 예의 액세스 플랜보다 비용면에서 더 효율적입니다. 누적 비용은 쿼리 2의 대략 1,755 timeron에서 쿼리 3의 대략 959 timeron으로 줄었습니다.

   그러나 쿼리 3의 액세스 플랜은 STAFF 테이블의 인덱스 스캔 IXSCAN (17)과 FETCH (11)를 보여줍니다. 페치 조작과 조합된 인덱스 스캔은 전체 테이블 스캔보다 비용이 덜 들지만, 이는 검색되는 행마다 테이블이 한 번 액세스되고 인덱스가 한 번 액세스된다는 것을 의미합니다. STAFF 테이블에서의 이러한 이중 액세스를 줄여봅시다.

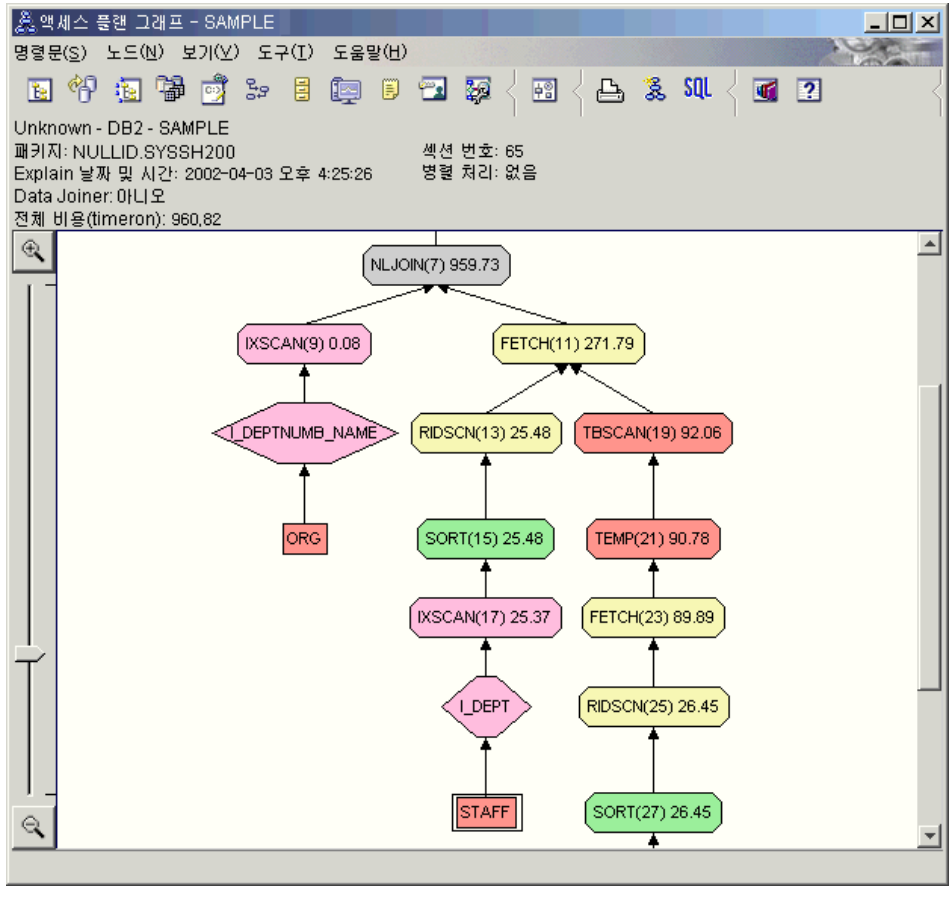
4. 다음 내용은 무엇입니까?

쿼리 4는 페치 및 인덱스 스캔을 페치가 없는 단일 인덱스 스캔으로 줄입니다. 추가 인덱스를 작성하면 액세스 플랜에 대해 계산된 비용이 줄어듭니다.

테이블 컬럼에 추가 인덱스 작성

이 예는 STAFF 테이블에서 JOB 컬럼에 인덱스를 작성하고 ORG 테이블에서 기존 인덱스에 DEPTNAME을 추가하여 쿼리 3에서 설명한 액세스 플랜에서 빌드합니다(별도의 인덱스를 추가하면, 추가 액세스가 발생합니다).

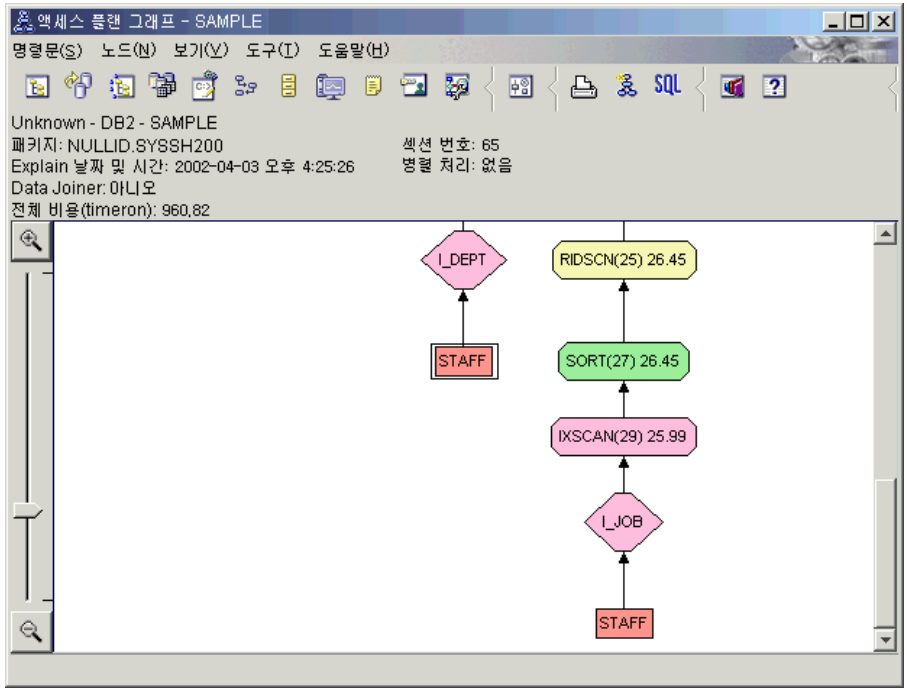
Explain된 명령문 실행기록 창에서 쿼리(쿼리 4)에 대한 액세스 플랜 그래프를 보려면, 쿼리 번호 4로 식별되는 항목을 더블 클릭하십시오. 이 명령문 실행을 위한 액세스 플랜 그래프 창이 열립니다.



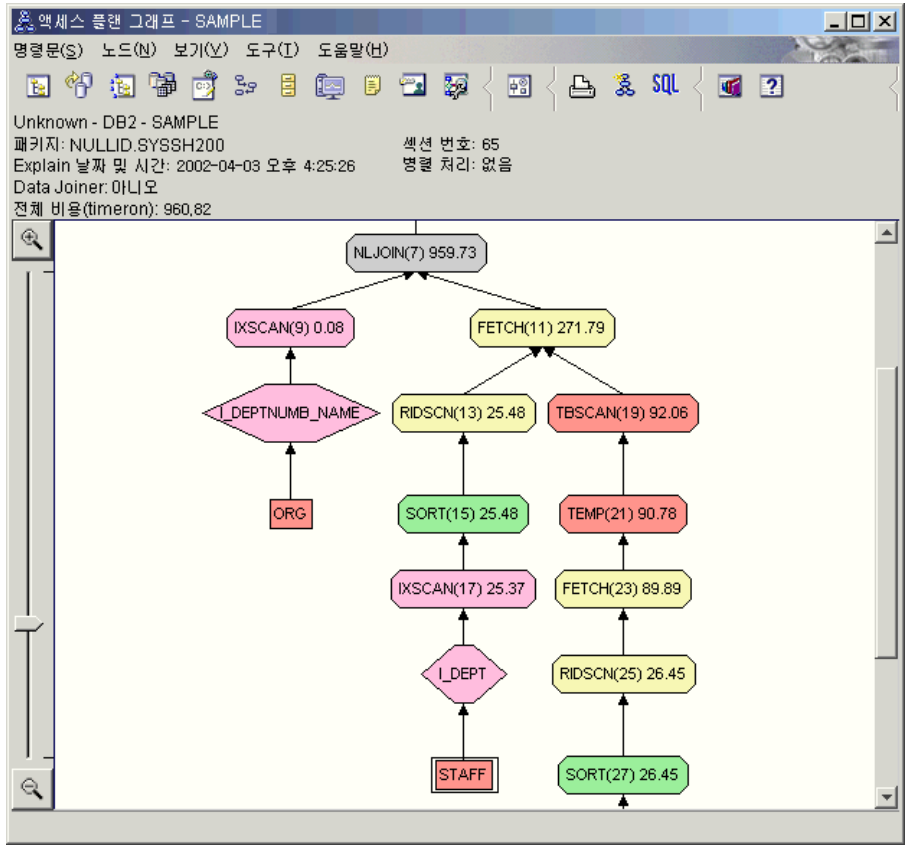
다음 질문에 응답하면 쿼리를 향상시키는 방법을 이해하는 데 도움이 됩니다.

1. 추가 인덱스 작성 결과로 이 액세스 플랜을 어떻게 변경했습니까?

   옵티마이저가 나중에 이 액세스 플랜을 더 세분화하기 위해 STAFF 테이블(**I_JOB** 레이블이 붙은 다이어몬드로 표시)에서 JOB 컬럼에 작성된 인덱스를 이용했습니다.



   액세스 플랜 그래프 중간 부분에서, ORG 테이블의 경우 이전 인덱스 스캔 및 페치가 인덱스 스캔 IXSCAN (9)만으로 변경되었다는 점에 유의하십시오. ORG 테이블의 인덱스에 DEPTNAME 컬럼을 추가하면 옵티마이저가 페치에 관련된 여분의 액세스를 제거하게 할 수 있습니다.



2. 이 액세스 플랜이 얼마나 효과적입니까?

   이 액세스 플랜은 이전 예의 액세스 플랜보다 비용면에서 더 효율적입니다. 누적 비용은 쿼리 3의 대략 1,370 timeron에서 쿼리 4의 대략 959 timeron으로 줄었습니다.

다음 레슨 내용

성능을 향상시키기 위해 수행할 수 있는 추가 단계에 대한 자세한 정보를 찾으려면, *관리 안내서*를 참조하십시오. 그런 후 사용자 조치의 영향에 액세스하기 위해 Visual Explain으로 되돌아갈 수 있습니다.

레슨 4. 파티션된 데이터베이스 환경에서 액세스 플랜 향상

이 레슨에서는 다양한 성능 조정 활동을 수행할 때 기본 쿼리에 대한 액세스 플랜 및 관련 창이 어떻게 변하는지 배우게 됩니다. 그림과 함께 제공되는 일련의 예를 사용하여, **runstats** 명령을 사용하고 해당 인덱스를 추가함으로써 단순 쿼리의 액세스 플랜에 대해 계산된 전체 비용이 향상될 수 있는 방법에 대해 배우게 됩니다.

Visual Explain에 익숙해지게 되면, 쿼리를 조정하는 다른 방법을 발견하게 됩니다.

액세스 플랜 그래프에 대한 작업

예에서처럼 네 개의 샘플 Explain 스냅샷을 사용하여, 어떻게 성능 조정이 데이터베이스 성능의 중요한 부분이 되는지를 배우게 됩니다.

Explain 스냅샷에 연관된 쿼리는 1 - 4로 번호가 지정되어 있습니다. 각 쿼리는 동일한 SQL문(레슨 1에 설명되어 있음)을 사용합니다.

```
SELECT S.ID,S.NAME,O.DEPTNAME,SALARY+COMM
FROM ORG O, STAFF S
  WHERE
  O.DEPTNUMB = S.DEPT AND
  S.JOB <> 'Mgr' AND
  S.SALARY+S.COMM > ALL( SELECT ST.SALARY*.9
                         FROM STAFF ST
                         WHERE ST.JOB='Mgr' )
ORDER BY S.NAME
```

그러나 각 쿼리 반복은 이전 실행보다 더 많은 성능 조정 기법을 사용합니다. 예를 들면, 쿼리 1에는 성능 조정이 없는 반면, 쿼리 4에는 가장 많습니다. 쿼리의 차이점에 대해서는 다음에 설명되어 있습니다.

쿼리 1 인덱스와 통계 없이 쿼리 실행

쿼리 2 쿼리에서 테이블 및 인덱스에 대한 현재 통계 수집

쿼리 3 쿼리에서 테이블을 조인하는 데 사용되는 컬럼에 대한 인덱스 작성

쿼리 4 테이블 컬럼에 추가 인덱스 작성

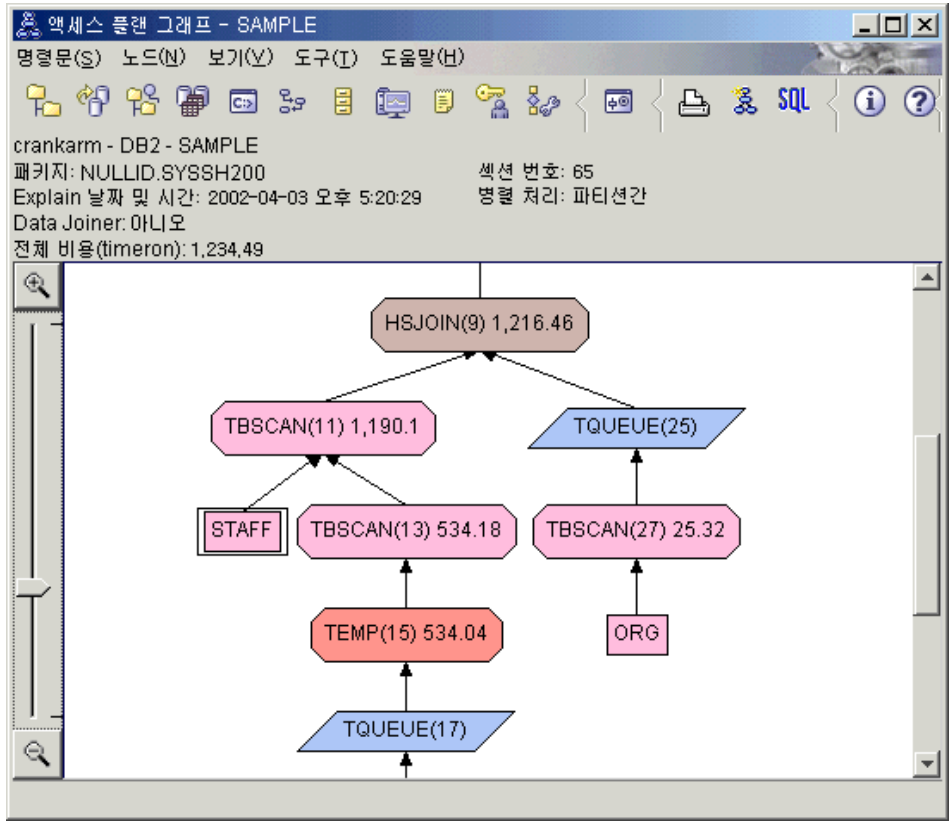
이 예는 파티션간 병렬 처리를 사용하여 7개의 물리적 노드로 RS/6000 SP 머신에서 생성되었습니다.

인덱스와 통계 없이 쿼리 실행

이 예에서 액세스 플랜은 인덱스와 통계 없이 SQL 쿼리에 대해 작성되었습니다.

쿼리(쿼리 1)에 대한 액세스 플랜 그래프를 보려면, 다음을 수행하십시오.

1. 제어 센터에서 샘플 데이터베이스를 찾을 때까지 오브젝트 트리를 펼치십시오.
2. 데이터베이스를 마우스 오른쪽 단추로 누른 후 팝업 메뉴에서 **Explain된 명령문 실행기록 표시**를 선택하십시오. Explain된 명령문 실행기록 창이 열립니다.
3. 쿼리 번호 1로 식별되는 항목을 더블 클릭하십시오(**쿼리 번호** 컬럼을 찾기 위해 오른쪽으로 스크롤해야 할 수 있습니다). 명령문에 대한 액세스 플랜 그래프 창이 열립니다.



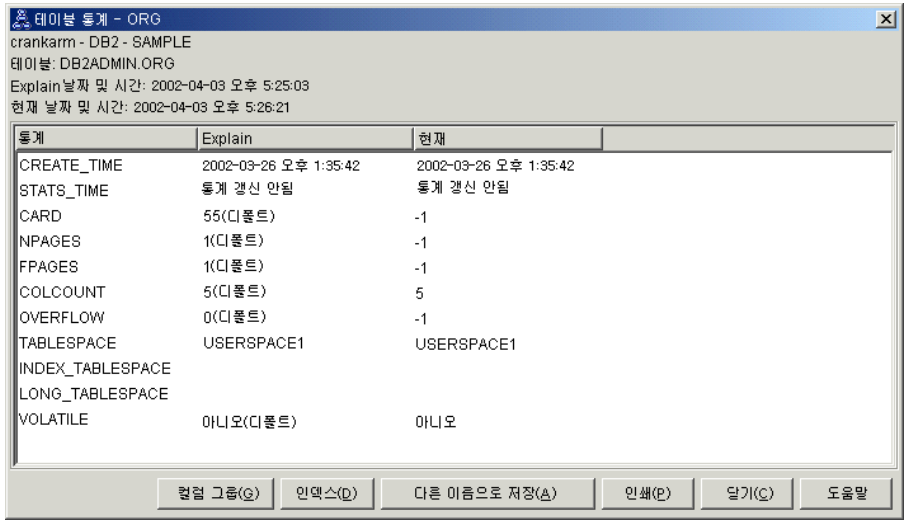
다음 질문에 응답하면 쿼리를 향상시키는 방법을 이해하는 데 도움이 됩니다.

1. 쿼리에서 테이블마다 현재 통계가 존재합니까?

   쿼리에서 테이블마다 현재 통계가 존재하는지 점검하려면, 액세스 플랜 그래프에서 각 테이블 노드를 더블 클릭하십시오. 열린 해당 테이블 통계 창에는 **Explain** 컬럼 아래의 **STATS_TIME** 행에 스냅샷 작성시 통계가 수집되지 않았음을 알리는 "통계 갱신 안됨"이 표시됩니다.

   현재 통계가 없으면, 옵티마이저는 실제 통계와는 다를 수 있는 디폴트 통계를 사용합니다. 디폴트 통계는 테이블 통계 창의 **Explain** 컬럼 아래의 "디폴트"라는 단어로 식별됩니다.

ORG 테이블에 대한 테이블 통계 창의 정보에 따라, 옵티마이저는 디폴트 통계를 사용했습니다(Explain 값 옆에 표시된 대로). 디폴트 통계는 스냅샷 작성 시 실제 통계를 사용할 수 없으므로 사용되었습니다(**STATS_TIME** 행에 표시된 대로).



2. 이 액세스 플랜이 데이터 액세스에 가장 효과적인 메소드를 사용합니까?

   이 액세스 플랜에는 인덱스 스캔이 아니라 테이블 스캔이 포함되어 있습니다. 테이블 스캔은 8각형으로 표시되며 TBSCAN 레이블이 붙어 있습니다. 인덱스 스캔이 사용되었으면, 이 스캔은 다이아몬드로 표시되며 IXSCAN 레이블이 붙어 있습니다. 소량의 데이터가 발췌되고 있는 경우 테이블에 대해 작성된 인덱스를 사용하는 것이 테이블 스캔보다 비용면에서 효율적입니다.

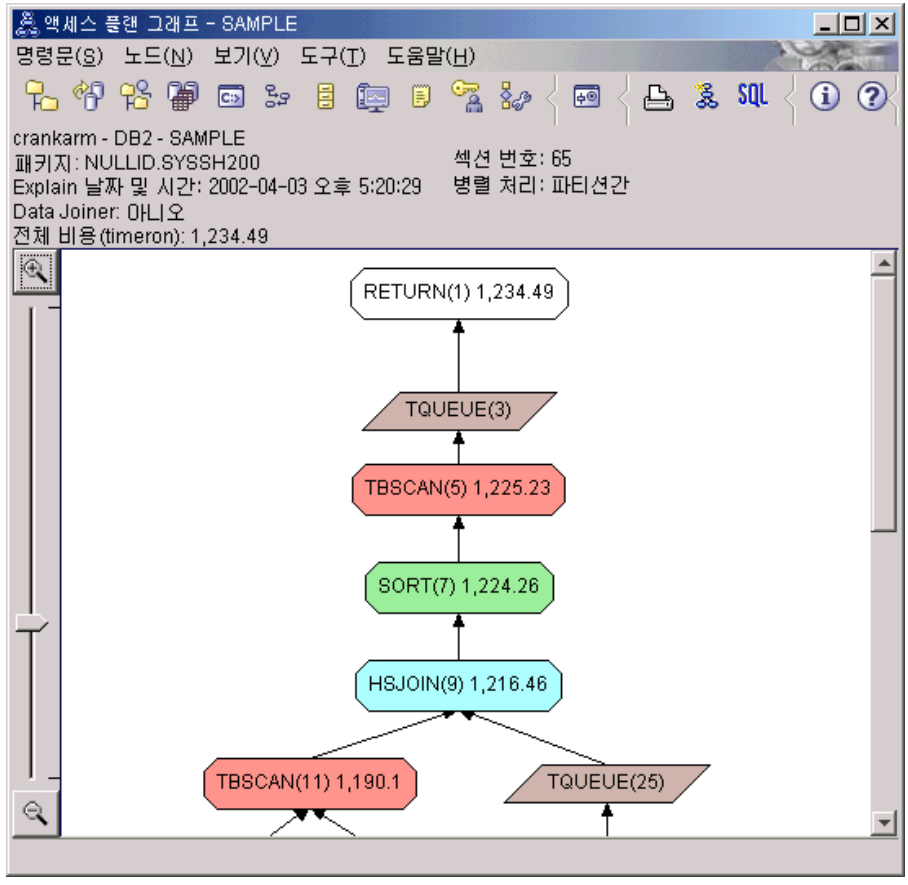
3. 이 액세스 플랜이 얼마나 효과적입니까?

   실제 통계를 사용하는 경우에만 액세스 플랜의 효과를 판별할 수 있습니다. 옵티마이저가 액세스 플랜의 디폴트 통계를 사용했으므로, 플랜이 얼마나 효과적인지 판별할 수 없습니다.

   일반적으로, 나중에 개정된 액세스 플랜과 비교하기 위해 액세스 플랜에 대해 계산된 전체 비용을 기록해 두어야 합니다. 각 노드에 나열된 비용은 쿼리의 첫 단계에서 노드까지(경계 포함) 누적됩니다.

주: 파티션된 데이터베이스의 경우, 이는 대부분의 자원을 사용하는 노드의 누적 비용입니다.

액세스 플랜 그래프 창에서 그래프의 맨 위에 있는 **RETURN (1)**에 표시된 전체 비용은 대략 1,234 timeron입니다. 계산된 전체 비용은 창의 맨 위 영역에도 표시됩니다.



4. 다음 내용은 무엇입니까??

   쿼리 2는 **runstats**가 실행된 후에 기본 쿼리에 대한 액세스 플랜을 조회합니다. **runstats** 명령을 사용하면 쿼리에서 액세스되는 모든 테이블의 현재 통계가 옵티마이저에 제공됩니다.

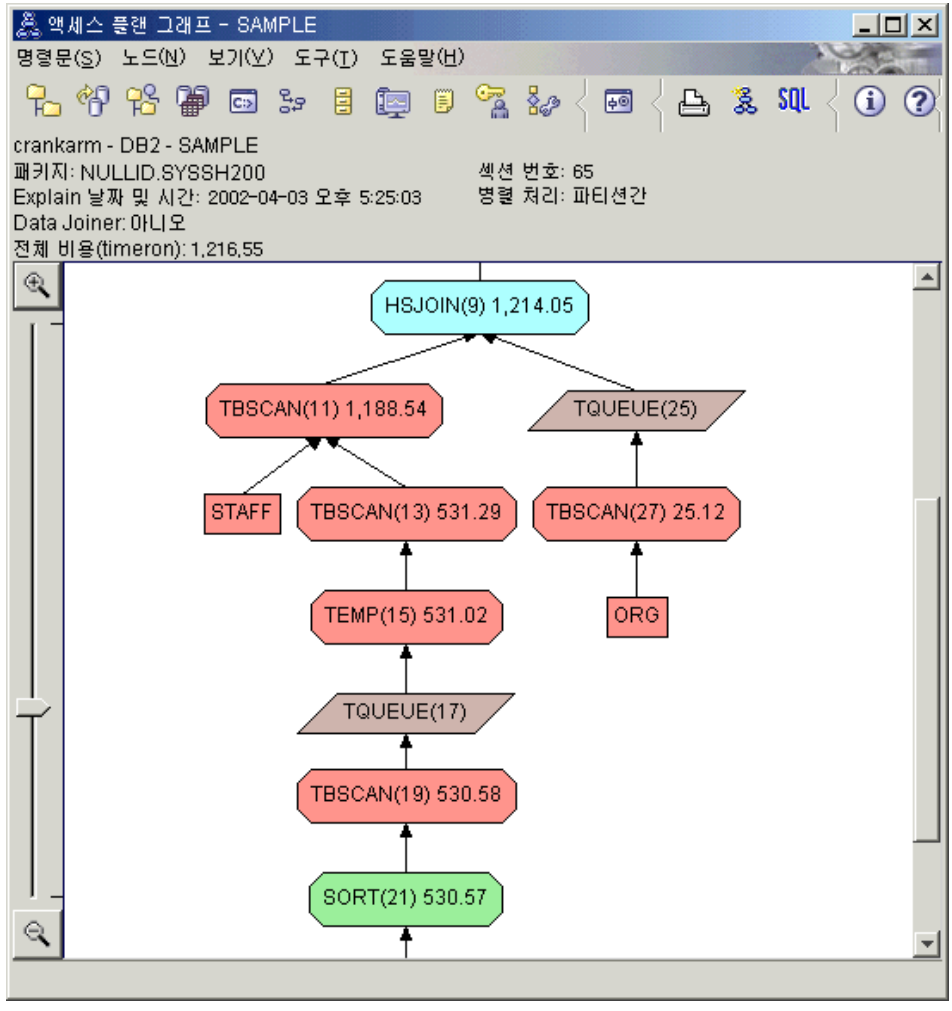
Runstats를 사용하여 테이블 및 인덱스에 대한 현재 통계 수집

이 예는 **runstats** 명령으로 현재 통계를 수집하여 쿼리 1에 설명된 액세스 플랜에 빌드합니다.

runstats 명령을 사용하여 테이블 및 인덱스에 대한 현재 통계를 수집하는 것이 바람직한데, 이는 마지막으로 **runstats** 명령을 실행한 이후 새로운 인덱스가 작성되었거나 중요한 갱신 활동이 발생한 경우에 특히 권장됩니다. 이는 옵티마이저에 최상의 액세스 플랜을 판별할 수 있는 가장 정확한 정보를 제공합니다. 현재 통계를 사용할 수 없으면, 옵티마이저는 정확하지 않은 디폴트 통계에 따라 비효율적인 액세스 플랜을 선택할 수 있습니다.

runstats 명령은 테이블 갱신 후 사용해야 하며, 그렇지 않을 경우 빈 테이블이 옵티마이저에 표시됩니다. 이러한 문제점은 연산자 세부사항 창에 대한 카디널리티가 0인 경우 입증됩니다. 이 경우, 테이블 갱신을 완료하고, **runstats** 명령을 재실행한 후 영향을 받는 테이블에 대한 Explain 스냅샷을 재작성하십시오.

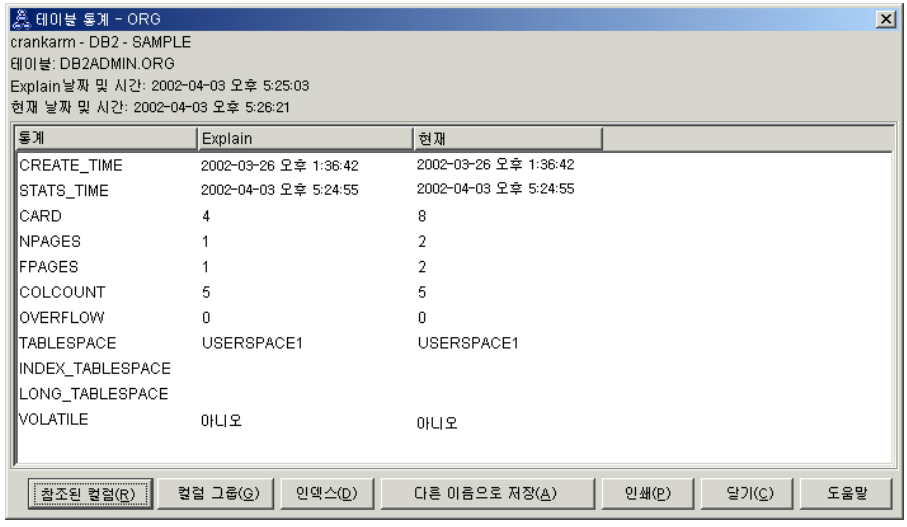
Explain된 명령문 실행기록 창에서 쿼리(쿼리 2)에 대한 액세스 플랜 그래프를 보려면, 쿼리 번호 2로 식별되는 항목을 더블 클릭하십시오. 이 명령문 실행을 위한 액세스 플랜 그래프 창이 열립니다.



다음 질문에 응답하면 쿼리를 향상시키는 방법을 이해하는 데 도움이 됩니다.

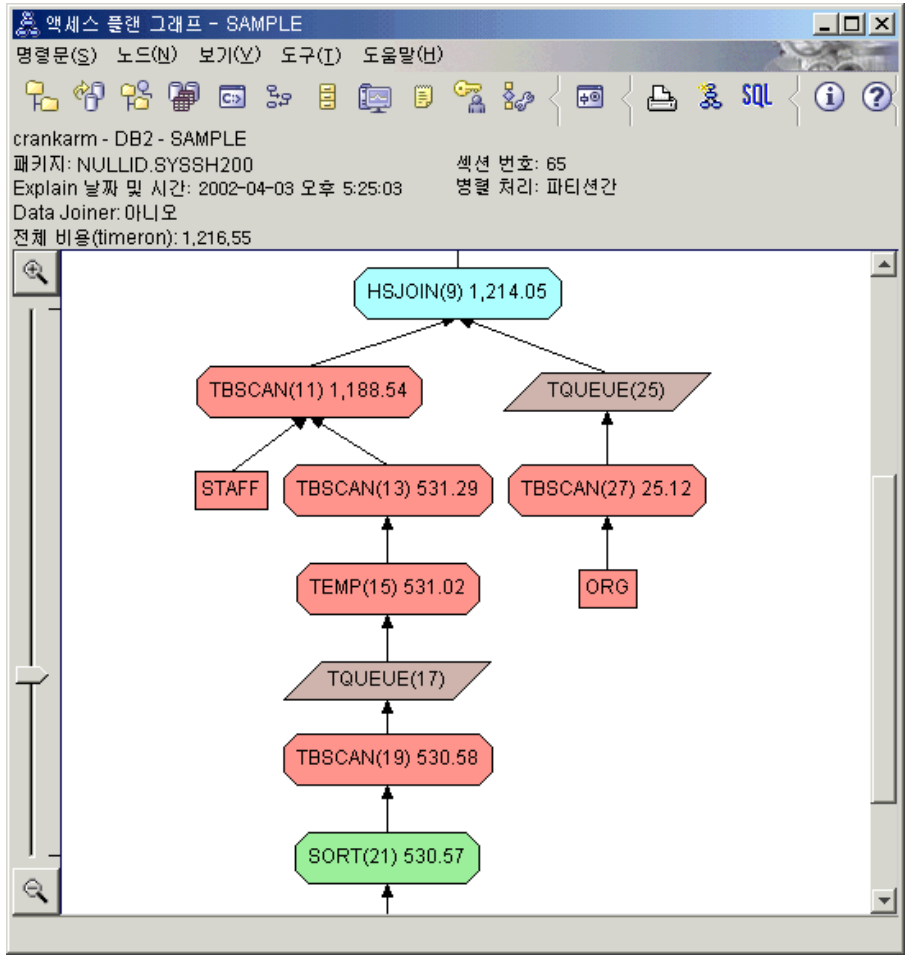
1. 쿼리에서 테이블마다 현재 통계가 존재합니까?

   ORG 테이블에 대한 테이블 통계 창은 옵티마이저가 실제 통계를 사용한다는 것을 보여줍니다(**STATS_TIME** 값은 통계가 수집된 실제 시간임). 통계의 정확성은 **runstats** 명령이 실행된 이후에 테이블의 내용에 현저한 변경사항이 있는지 여부에 따라 다릅니다.

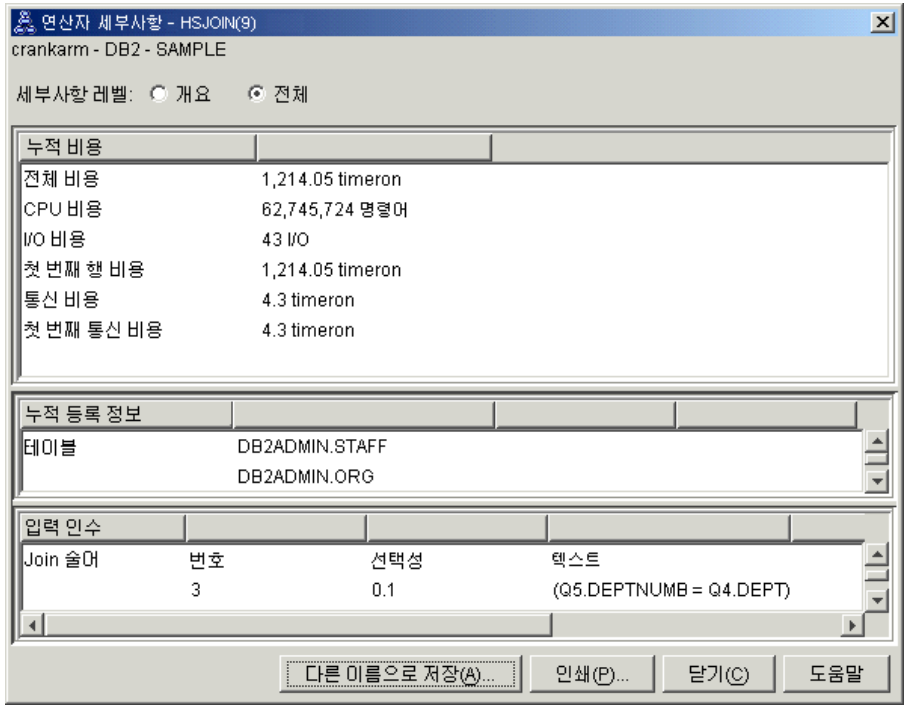


2. 이 액세스 플랜이 데이터 액세스에 가장 효과적인 메소드를 사용합니까?

   쿼리 1과 같이 쿼리 2의 액세스 플랜은 인덱스 스캔(IXSCAN)이 아니라 테이블 스캔(TBSCAN)을 사용합니다. 현재 통계가 존재하는 경우에도 인덱스 스캔은 쿼리에서 사용된 컬럼에 인덱스가 없으므로 수행되지 않았습니다. 쿼리를 향상시키는 한 방법은 테이블을 조인하는 데 사용되는 컬럼(즉, Join 술어에서 사용되는 컬럼)의 인덱스를 옵티마이저에 제공하는 것입니다. 이 예에서는 이것이 처음 병합 스캔 조인 HSJOIN (9)입니다.



HSJOIN (9) 연산자에 대한 연산자 세부사항 창에서 **입력 인수** 아래의 **Join 술어** 섹션을 보십시오. 이 조인 조작에서 사용되는 컬럼은 **텍스트** 컬럼 아래에 나열되어 있습니다. 이 예에서, 이러한 컬럼은 DEPTNUMB 및 DEPT입니다.



3. 이 액세스 플랜이 얼마나 효과적입니까?

   최신 통계에 기초한 액세스 플랜은 항상 실제로 계산된 비용(timeron 단위)을 생성합니다. 쿼리 1에서 계산된 비용은 디폴트 통계에 기초하므로, 어느 것이 더 효과적인지 판별하기 위해 두 개의 액세스 플랜 그래프의 비용을 비교할 수 없습니다. 비용이 높거나 낮은지 여부는 관련되지 않습니다. 유효한 효과 측정을 얻기 위해서는 실제 통계에 따라 액세스 플랜 비용을 비교해야 합니다.

4. 다음 내용은 무엇입니까?

   쿼리 3에서는 DEPTNUMB 및 DEPT 컬럼에서 인덱스 추가 효과를 검색합니다. Join 술어에 사용되는 컬럼에 인덱스 추가는 성능을 향상시킬 수 있습니다.

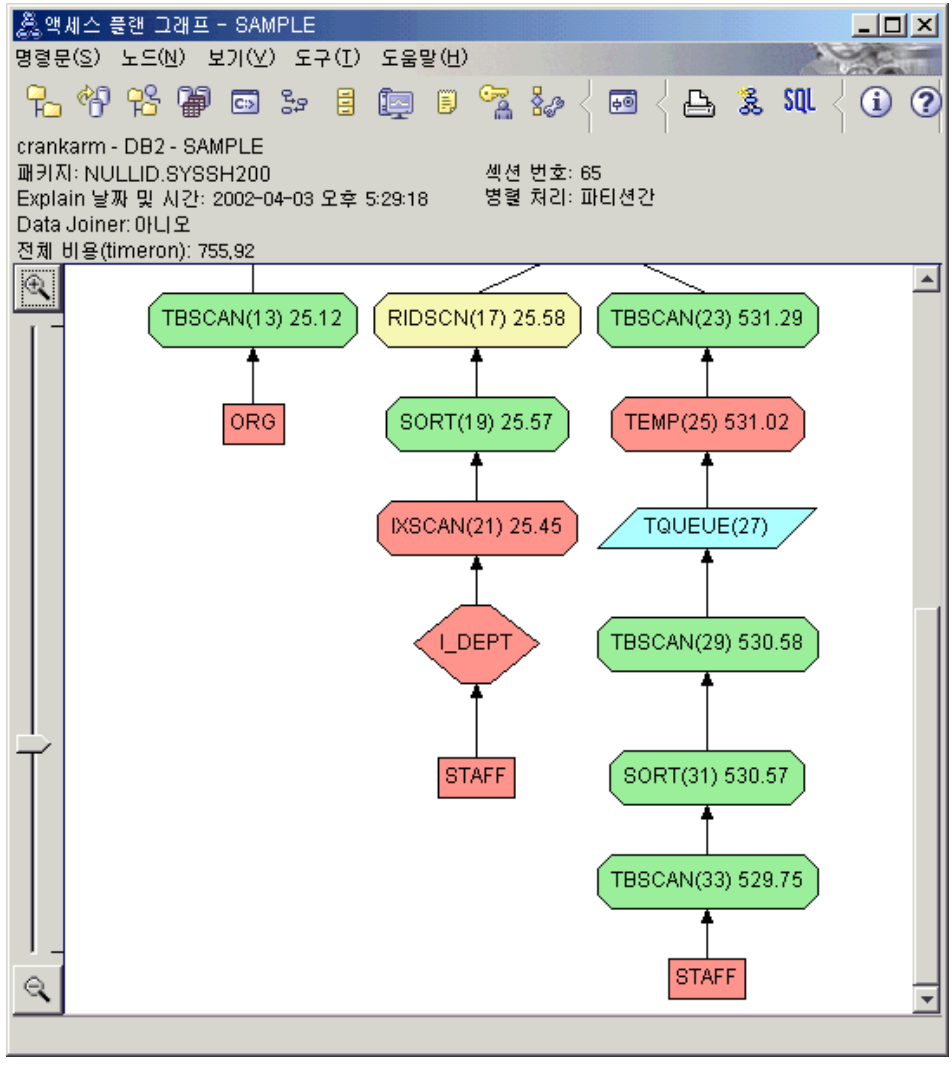
쿼리에서 테이블을 조인하는 데 사용되는 컬럼에 대한 인덱스 작성

이 예는 STAFF 테이블의 DEPT 컬럼과 ORG 테이블의 DEPTNUMB 컬럼에 인덱스를 작성하여 쿼리 2에 설명된 액세스 플랜에서 빌드합니다.

주: 버전 8에서 권장 인덱스는 워크로드 성능 마법사를 사용하여 작성될 수 있습니다.

Explain된 명령문 실행기록 창에서 쿼리(쿼리 3)에 대한 액세스 플랜 그래프를 보려면, 쿼리 번호 3으로 식별되는 항목을 더블 클릭하십시오. 이 명령문 실행을 위한 액세스 플랜 그래프 창이 열립니다.

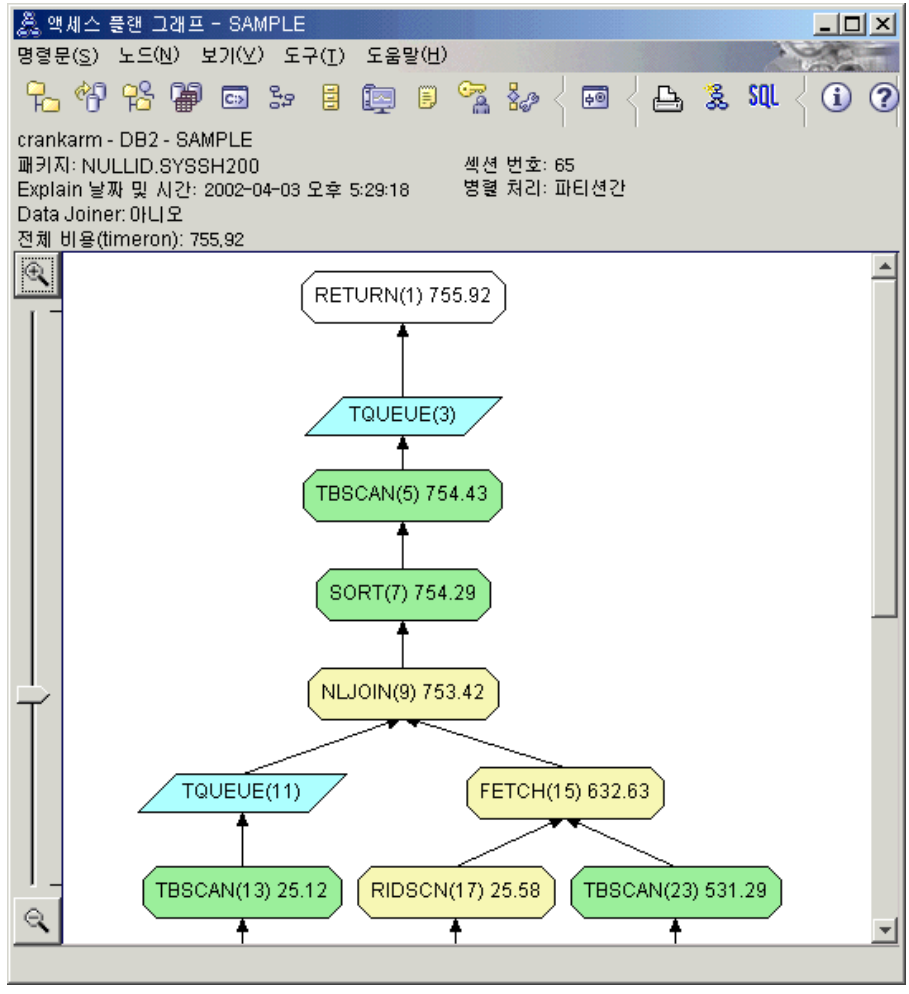
주: DEPTNUM에 대한 인덱스가 작성되었더라도 옵티마이저가 이를 사용하지 않았습니다.



다음 질문에 응답하면 쿼리를 향상시키는 방법을 이해하는 데 도움이 됩니다.

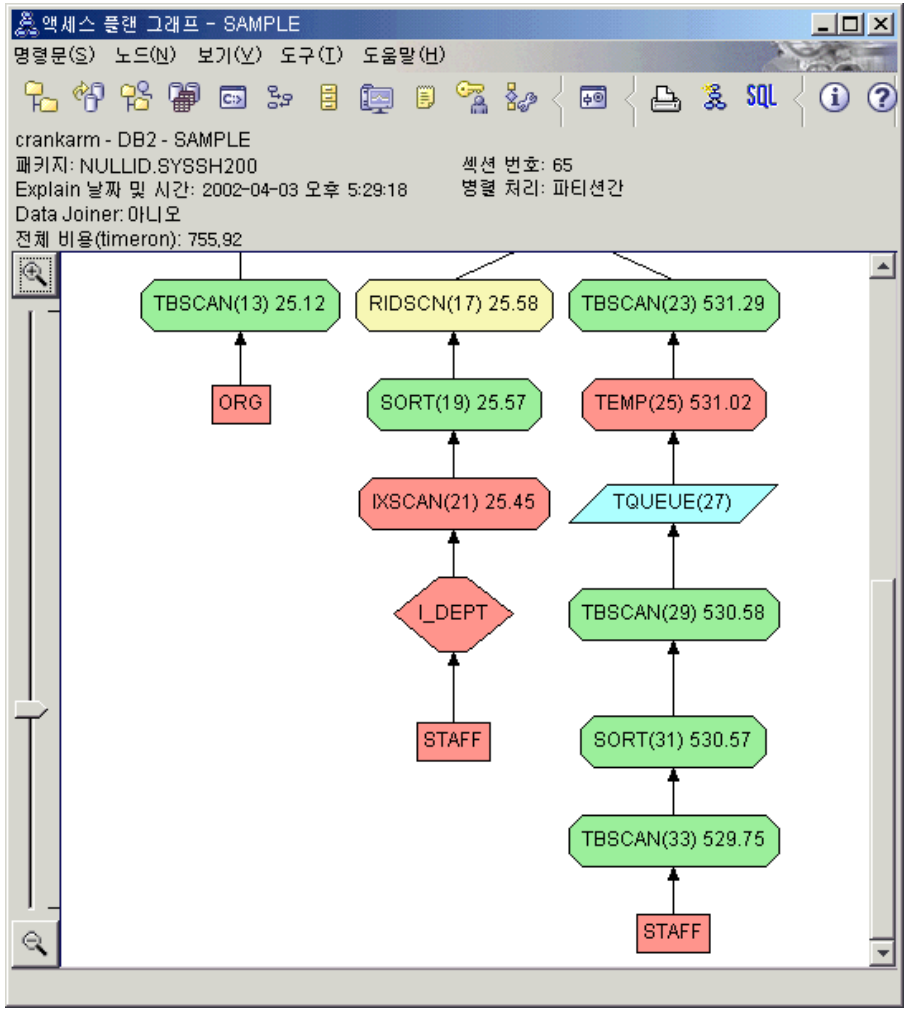
1. 인덱스가 있는 액세스 플랜에서 무엇이 변경되었습니까?

   새로운 다이아몬드 모양의 노드인 **I_DEPT**가 STAFF 테이블 바로 위에 추가되었습니다. 이 노드는 DEPT에 작성된 인덱스를 나타내며, 검색할 행을 판별하기 위해 옵티마이저가 테이블 스캔 대신에 인덱스 스캔을 사용한다는 것을 보여줍니다.

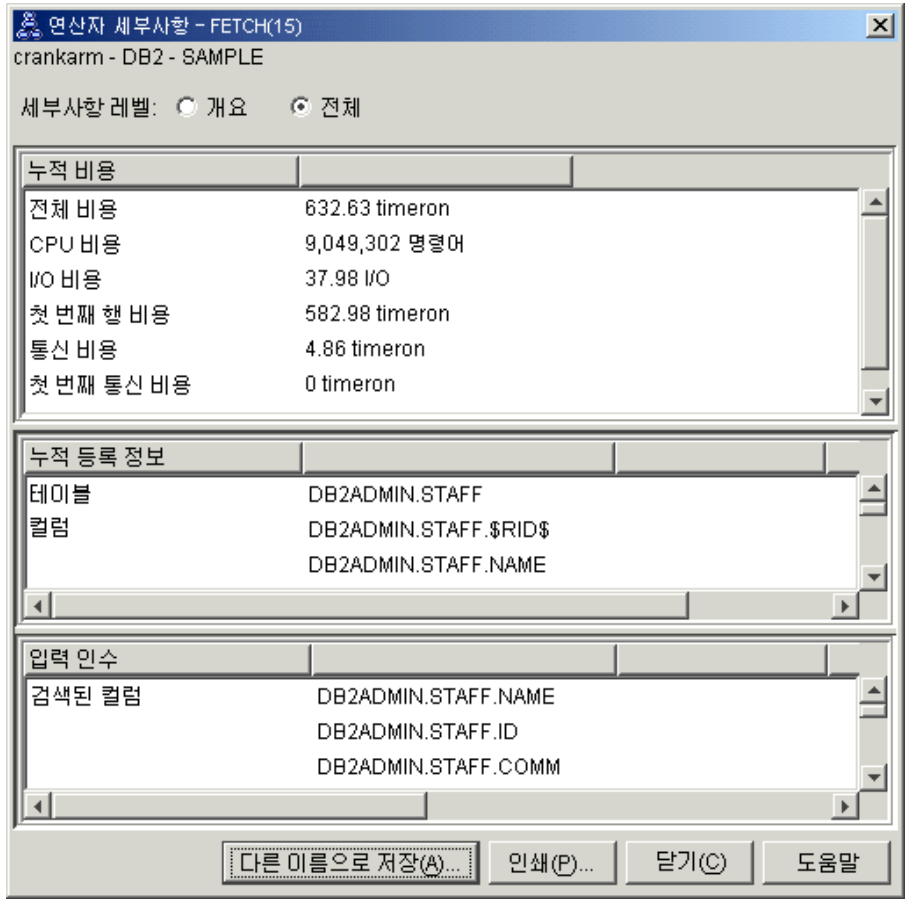


2. 이 액세스 플랜이 데이터 액세스에 가장 효과적인 메소드를 사용합니까?

   이 쿼리에 대한 액세스 플랜은 결과가 FETCH (15) 및 IXSCAN (21)인 ORG 테이블의 DEPTNUMB 컬럼과 STAFF 테이블의 DEPT 컬럼에 대한 인덱스 작성 효과를 보여줍니다. 쿼리 2에는 이 인덱스가 없으므로 예에서 테이블 스캔이 사용되었습니다.



FETCH (15) 연산자에 대한 연산자 세부사항 창에서는 이 조작에 사용되고 있는 컬럼을 보여줍니다.



인덱스와 페치의 조합은 이전 액세스 플랜에서 사용된 전체 테이블 스캔보다 비용이 덜 드는 것으로 계산되었습니다.

3. 이 액세스 플랜이 얼마나 효과적입니까?

   이 액세스 플랜은 이전 예의 액세스 플랜보다 비용면에서 더 효율적입니다. 누적 비용은 쿼리 2의 대략 1,214 timeron에서 쿼리 3의 대략 755 timeron으로 줄었습니다.

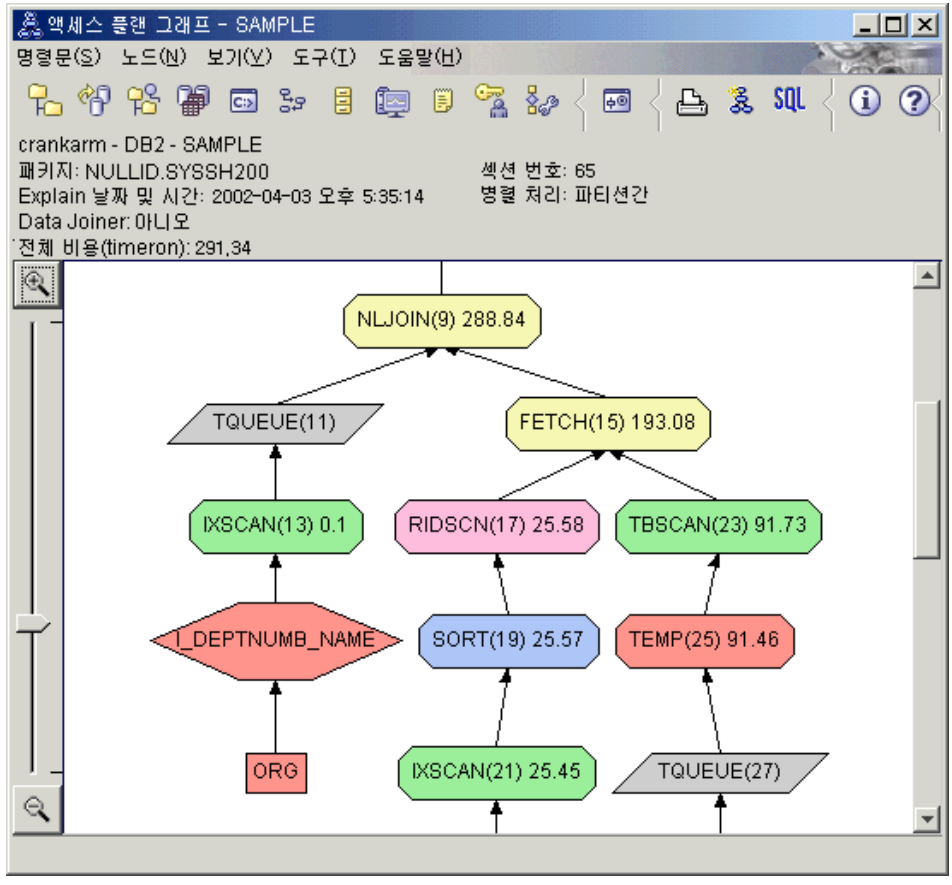
4. 다음 내용은 무엇입니까?

   쿼리 4는 페치 및 인덱스 스캔을 페치가 없는 단일 인덱스 스캔으로 줄입니다. 추가 인덱스를 작성하면, 액세스 플랜에 대해 계산된 비용이 줄어듭니다.

테이블 컬럼에 추가 인덱스 작성

이 예는 STAFF 테이블에서 JOB 컬럼에 인덱스를 작성하고 ORG 테이블에서 기존 인덱스에 DEPTNAME을 추가함으로써 쿼리 3에서 설명한 액세스 플랜에서 빌드합니다(별도의 인덱스를 추가하면, 추가 액세스가 발생합니다).

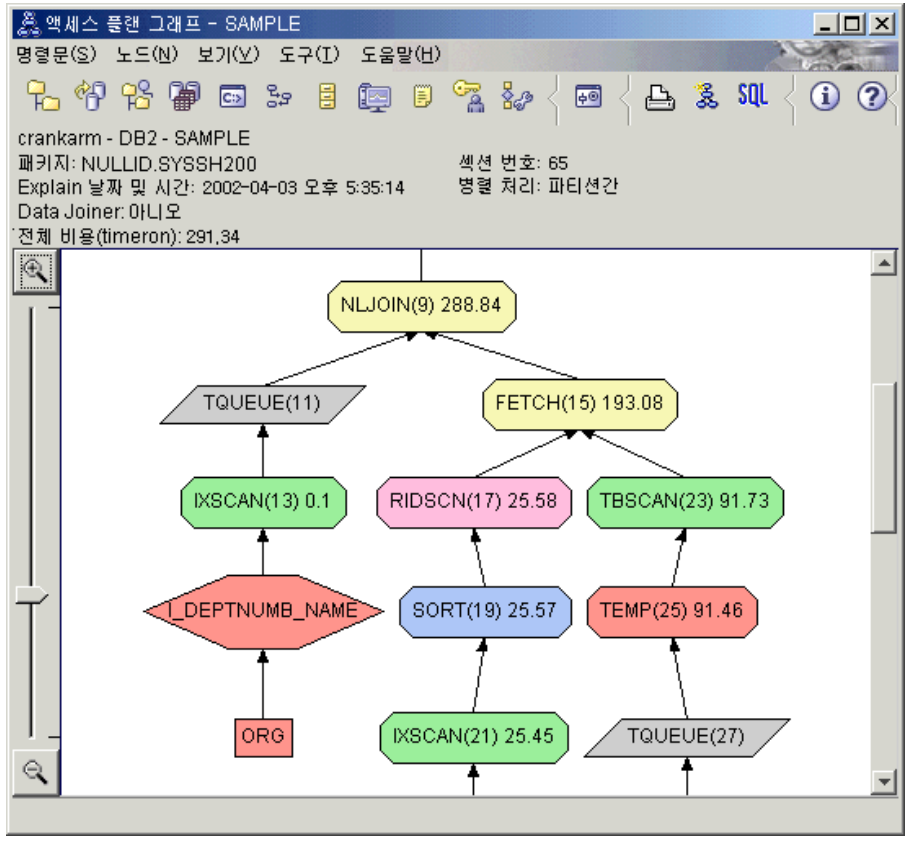
Explain된 명령문 실행기록 창에서 쿼리(쿼리 4)에 대한 액세스 플랜 그래프를 보려면, 쿼리 번호 4로 식별되는 항목을 더블 클릭하십시오. 이 명령문 실행을 위한 액세스 플랜 그래프 창이 열립니다.



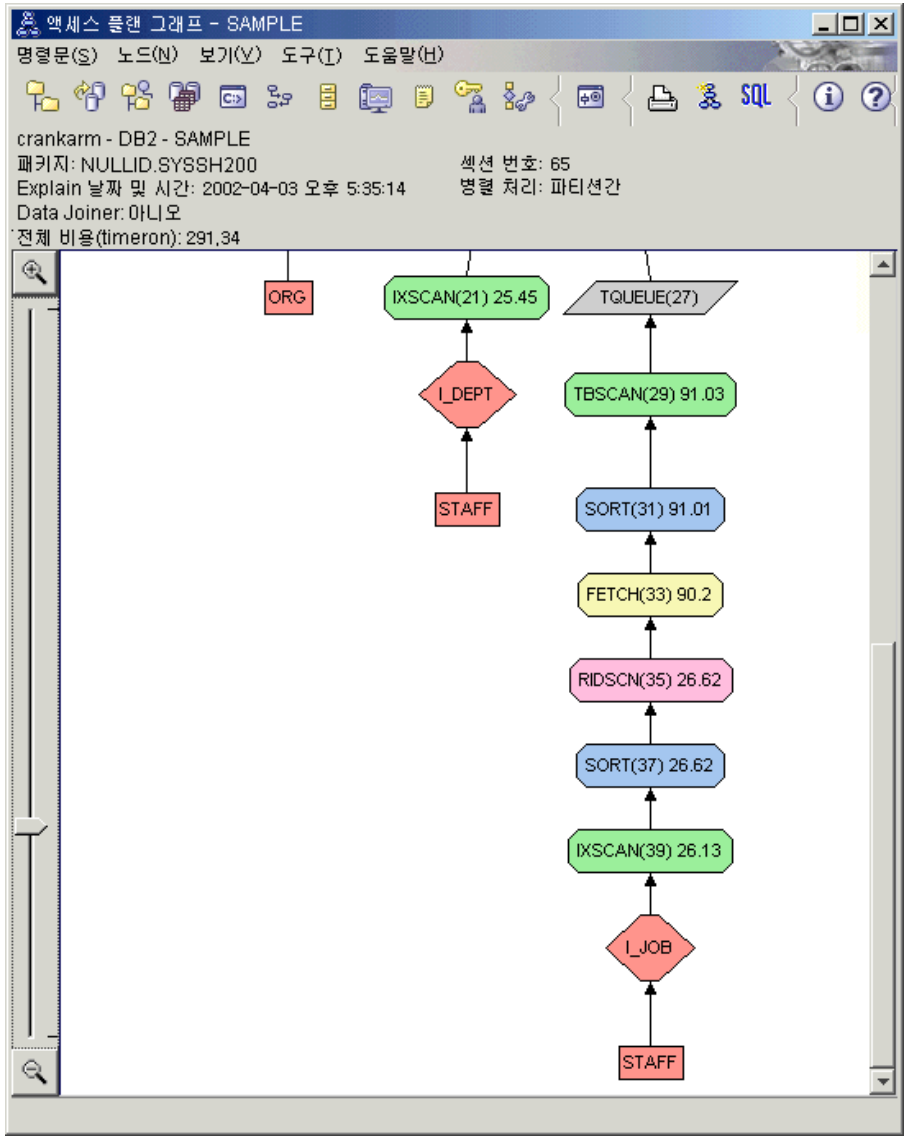
다음 질문에 응답하면 쿼리를 향상시키는 방법을 이해하는 데 도움이 됩니다.

1. 추가 인덱스 작성 결과로 이 액세스 플랜을 어떻게 변경했습니까?

액세스 플랜 그래프의 중간 부분에서, ORG 테이블의 경우 이전 테이블 스캔이 인덱스 스캔 IXSCAN (9)으로 변경되었다는 점에 유의하십시오. ORG 테이블의 인덱스에 DEPTNAME 컬럼을 추가하면 옵티마이저가 테이블 스캔에 관련된 액세스를 세분화할 수 있습니다.



액세스 플랜 그래프 맨 아래 부분에서, STAFF 테이블의 경우 이전 인덱스 스캔 및 페치가 인덱스 스캔 IXSCAN (39)만으로 변경되었다는 점에 유의하십시오. STAFF 테이블에 JOB 인덱스를 작성하면 옵티마이저가 페치에 관련된 여분의 액세스를 제거할 수 있습니다.



2. 이 액세스 플랜이 얼마나 효과적입니까?

   이 액세스 플랜은 이전 예의 액세스 플랜보다 비용면에서 더 효율적입니다. 누적 비용은 쿼리 3의 대략 753 timeron에서 쿼리 4의 대략 288 timeron으로 줄었습니다.

다음 레슨 내용

성능을 향상시키기 위해 수행할 수 있는 추가 단계에 대한 자세한 정보를 찾으려면, *관리 안내서*를 참조하십시오. 그런 후 사용자 조치의 영향에 액세스하기 위해 Visual Explain으로 되돌아갈 수 있습니다.

부록 A. Visual Explain 개념

액세스 플랜

Explain 가능한 SQL문을 해결하는 데 있어 특정 데이터가 필요합니다. *액세스 플랜*은 이러한 데이터에 액세스하기 위한 조작의 순서를 자세히 기술합니다. 이를 이용하여 선택한 테이블, 인덱스 또는 컬럼에 관한 통계치, 연산자 등록 정보, 테이블 스페이스 및 함수 통계와 같은 전역 정보, 최적화와 관련된 구성 매개변수 등을 볼 수 있습니다. Visual Explain을 사용하여 그래픽 양식의 SQL문에 대한 액세스 플랜을 볼 수 있습니다.

옵티마이저는 Explain 가능한 SQL문이 컴파일될 때마다 액세스 플랜을 작성합니다. 이 작업은 정적 SQL문에서는 prep/bind할 때, 동적 SQL문에서는 실행시에 일어납니다.

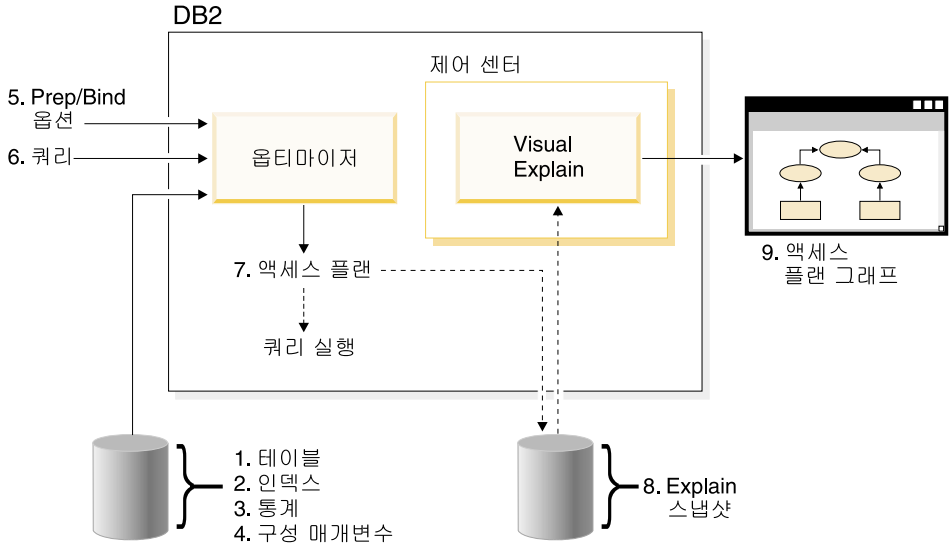
액세스 플랜이란 사용 가능한 정보를 근거로 한 일종의 *예측*입니다. 옵티마이저가 다음과 같은 정보를 근거로 예측을 합니다.

- 시스템 카탈로그 테이블에 있는 통계치(만약 현재 통계치가 아닌 경우, runstats 명령을 사용하여 이를 갱신하십시오.)
- 구성 매개변수
- 바인드 옵션
- 쿼리 최적화 클래스

액세스 플랜과 관련된 비용 정보는 쿼리 수행을 위한 자원 사용에 관한 옵티마이저의 *최상의 예측*입니다. 쿼리의 실제 경과 시간은 DB2 외부 요인(예를 들어, 동시에 실행되는 다른 응용프로그램 수 등)에 따라 달라집니다. 실제 경과 시간은 쿼리 실행 중에 성능 모니터링을 이용하여 측정할 수 있습니다.

액세스 플랜 그래프

Visual Explain은 아래 그림에 나와 있는 것과 같이 액세스 플랜 그래프를 작성하기 위해 많은 소스의 정보를 사용합니다. 다양한 입력에 기초하여 옵티마이저가 액세스 플랜을 선택하고, Visual Explain이 이를 *액세스 플랜 그래프*에 표시합니다. 그래프의 노드는 테이블과 인덱스 및 그에 대한 조작을 나타냅니다. 노드간의 링크는 데이터 흐름을 나타냅니다.



다음 태스크 목록은 위의 그림에 나타난 사항에 해당합니다(점선은 Visual Explain에 필요한 단계를 나타냅니다).

1. 사용자 테이블 설계 튜닝 및 테이블 데이터 재구성.
2. 적절한 인덱스 작성.
3. runstats 명령을 사용하여 옵티마이저에 현재 통계 제공.
4. 적절한 구성 매개변수 선택.
5. 적절한 바인드 옵션 선택.
6. 필요한 데이터만 검색하도록 쿼리 설계.
7. 액세스 플랜 작성.
8. Explain 스냅샷 작성.
9. 액세스 플랜 그래프 표시 및 사용.

예를 들어, Visual Explain을 사용하려면 먼저 명령문에 의해 사용된 인덱스와 테이블에 대해 **runstats** 명령을 사용하여 현재 통계를 갱신하십시오. 이러한 통계, 구성 매개변수, 바인드 옵션 및 쿼리 자체를 옵티마이저를 사용하여 패키지 바인드시 액세스 플랜과 Explain 스냅샷을 작성합니다. Visual Explain은 결과 Explain 스냅샷을 사용하여 이 명령문에 대한 액세스 플랜 그래프를 표시합니다.

액세스 플랜 그래프 노드

액세스 플랜 그래프는 노드들이 표시된 트리로 구성되어 있습니다. 이러한 노드는 다음을 나타냅니다.

- 테이블: 직사각형
- 인덱스: 다이아몬드꼴
- 연산자: 8각형, TQUEUE 연산자: 평행사변형
- 테이블 함수: 6각형

클러스터링

갱신이 거듭되면, 데이터 페이지의 행들이 위치를 바꾸게 되어 인덱스와 데이터 페이지간의 *클러스터* 등급이 낮아질 수 있습니다. 선택된 인덱스와 관련된 테이블을 재구성하면 데이터가 다시 클러스터링됩니다. 클러스터화 인덱스는 기본 테이블에 있는 데이터로의 순차적 액세스를 향상시키므로, 범위 술어가 있는 컬럼에 특히 유용합니다. 유사한 값들이 같은 데이터 페이지에 있으므로 페이지 페치 수가 훨씬 적어집니다.

일반적으로 한 테이블 내에서 한 인덱스만 높은 등급으로 클러스터화될 수 있습니다.

어떤 인덱스의 클러스터화 등급을 점검하려면, 해당 노드를 더블 클릭하여 인덱스 통계 창을 표시하십시오. 클러스터 비율 또는 클러스터 인수 값이 이 창에 표시됩니다. 이 값이 낮으면, 테이블 데이터의 재구성을 고려하십시오.

자세한 정보는 *관리 안내서*의 테이블 데이터 재구성에 관한 절을 참조하십시오.

컨테이너

*컨테이너*는 데이터의 물리적인 저장 장소를 말합니다. 컨테이너는 테이블 스페이스와 연관되어 있으며 파일이나 디렉토리 또는 디바이스를 말합니다.

비용

Visual Explain의 컨텍스트에서 *비용*은 명령문(또는 명령문의 요소)에 대한 액세스 플랜을 실행하는 데 필요한 전체 자원의 추정 사용량을 의미합니다. 비용은 CPU 비용(명령어 수)과 I/O(검색 및 페이지 전송 수)를 조합하여 추산됩니다.

비용의 단위는 *timeron*입니다. timeron은 어떠한 실제 경과 시간과도 직접적으로 일치하지는 않지만, 데이터베이스 관리 프로그램이 동일한 쿼리를 위해 두 개의 액세스 플랜을 실행할 때 필요한 자원(비용)의 소요량을 예측하여 제공합니다.

액세스 플랜 그래프의 각 연산자 노드에 표시되는 비용은 액세스 플랜 실행이 시작될 때부터 특정 연산자가 실행될 때까지의 누적된 비용입니다. 이 비용은 시스템의 워크로드나 사용자에게 데이터 행을 리턴하는 비용과 같은 요소는 포함되지 않습니다.

커서 블로킹

*커서 블로킹*은 데이터베이스 관리 프로그램이 한 블록의 행들을 한번의 조작으로 검색하도록 해서 오버헤드를 줄이는 기술입니다. 이 행들은 처리되는 동안 캐시에 저장됩니다. 응용프로그램이 OPEN CURSOR 요청을 발행하면 캐시가 할당되었다가 커서가 닫히면 캐시 할당이 해제됩니다. 모든 행이 처리되면 다른 블록의 행이 검색됩니다.

PREP 또는 **BIND** 명령의 BLOCKING 옵션을 다음 매개변수와 함께 사용하여 커서 블로킹의 유형을 지정하십시오.

UNAMBIG
명확한 커서만 블로킹됩니다(디폴트값).

ALL 앰비규어스(ambiguous) 커서 및 명확한 커서 모두 블로킹됩니다.

NO 커서들이 블로킹되지 않습니다.

자세한 내용은 *관리 안내서*의 커서 블로킹에 관한 절을 참조하십시오.

데이터베이스 관리 스페이스(DMS) 테이블 스페이스

데이터베이스에는 데이터베이스 관리 스페이스(DMS)와 시스템 관리 스페이스(SMS)라는 두 가지 유형의 테이블 스페이스가 있습니다.

DMS 테이블 스페이스는 데이터베이스 관리 프로그램이 관리되며, 요구사항에 맞도록 설계 및 조정됩니다.

DMS 테이블 스페이스 정의에는 DMS 테이블 스페이스 형식으로 데이터베이스 데이터가 저장된 파일(또는 디바이스) 목록이 포함됩니다.

사전 할당된 파일(또는 디바이스)을 기존 DMS 테이블 스페이스에 추가함으로써 그 스토리지 용량을 증가시킬 수 있습니다. 데이터베이스 관리 프로그램은 해당 테이블 스페이스에 속한 모든 컨테이너에 있는 기존 데이터의 균형을 자동으로 다시 조정합니다.

DMS 테이블 스페이스와 SMS 테이블 스페이스는 동일한 데이터베이스에 공존할 수 있습니다.

동적 SQL

*동적 SQL*문은 프로그램이 실행되는 동안 응용프로그램 내에서 준비되고 실행되는 SQL문입니다. 동적 SQL에서는 다음을 수행할 수 있습니다.

- CLI나 CLP를 사용하여 대화식으로 SQL문을 발행
- SQL 소스는 응용프로그램에 삽입되어 있는 호스트 언어 변수에 포함되어 있음

동적 SQL문을 실행할 때, DB2는 현재 카탈로그 통계와 구성 매개변수를 근거로 액세스 플랜을 작성합니다. 이 액세스 플랜은 응용프로그램의 한 명령문을 실행한 후 다음 명령문 실행시에 바뀔 수 있습니다.

동적 SQL에 대한 대안은 정적 SQL입니다.

Explain 스냅샷

Visual Explain을 사용하여 Explain 스냅샷의 내용을 살펴볼 수 있습니다.

*Explain 스냅샷*은 SQL문이 Explain될 때 수집되는 압축 정보입니다. 이 정보는 EXPLAIN_STATEMENT 테이블에 2진 대형 오브젝트(BLOB)로 저장되며 다음과 같은 정보가 들어 있습니다.

- 액세스 플랜의 내부 표현으로 연산자 및 액세스되는 테이블과 인덱스 등이 포함됨.
- 옵티마이저가 사용하는 결정 기준으로서, 데이터베이스 오브젝트에 대한 통계와 각 조작에 대한 누적 비용이 포함됨.

Explain 스냅샷은 SQL문의 액세스 플랜을 그래픽으로 표시할 경우에 필요합니다. Explain 스냅샷이 작성되어 있는지 확인하려면 다음을 수행하십시오.

1. Explain 테이블은 Explain 스냅샷을 저장하기 위한 데이터베이스 관리 프로그램에 존재해야 합니다. 이들 테이블을 작성하는 방법에 대한 정보는 온라인 도움말의 Explain 테이블 작성을 참조하십시오.
2. 정적 SQL문이 들어 있는 패키지의 경우, 패키지를 바인드하거나 준비할 때 EXPLSNAP 옵션을 ALL 또는 YES로 설정하십시오. 패키지에 있는 Explain 가능한 SQL문 각각에 대해 Explain 스냅샷을 얻게 될 것입니다. **BIND** 및 **PREP** 명령에 대한 자세한 내용은 *Command Reference*를 참조하십시오.
3. 동적 SQL문의 경우, 이를 발행하는 응용프로그램을 바인드할 때 EXPLSNAP 옵션을 ALL로 설정하십시오. 대화식으로 발행하려면, CURRENT EXPLAIN SNAPSHOT 특수 레지스터를 YES 또는 EXPLAIN으로 설정해야 합니다. 자세한 내용은 *SQL 참조서*의 현재 Explain 스냅샷에 관한 절을 참조하십시오.

Explain 가능한 명령문

*Explain 가능한 명령문*은 Explain 조작이 수행될 수 있는 SQL문입니다.

Explain 가능한 SQL문은 다음과 같습니다.

- SELECT
- INSERT
- UPDATE

- DELETE
- VALUES

Explain된 명령문

*Explain된 명령문*은 그에 대한 Explain 조작이 수행된 SQL문입니다. Explain된 명령문은 Explain된 명령문 실행기록 창에 표시됩니다.

피연산자

피연산자란 연산이 수행되는 대상입니다. 예를 들면, 테이블이나 인덱스는 TBSCAN과 IXSCAN 같은 다양한 연산자의 피연산자입니다.

연산자

*연산자*는 SQL문의 액세스 플랜이 수행될 때, 데이터에 대해 수행되어야 하는 조치이거나 인덱스 및 테이블로부터의 출력입니다.

다음과 같은 연산자가 액세스 플랜 그래프에 나타날 수 있습니다.

DELETE
테이블에서 행을 지웁니다.

EISCAN
행의 스트림을 줄이기 위해 사용자 정의 인덱스를 스캔합니다.

FETCH
특정 레코드 ID를 사용하여 테이블에서 컬럼을 페치합니다.

FILTER
한 개 이상의 조건으로 데이터를 필터합니다.

GRPBY
지정된 컬럼이나 함수의 공통값에 따라 행을 구분하고, 집합 함수를 평가합니다.

HSJOIN
둘 이상의 테이블이 조인 컬럼에서 해시된 해시 조인을 표시합니다.

INSERT

테이블에 행을 삽입합니다.

IXAND

두 개 이상의 인덱스를 스캔하여 나온 행 ID(RID)에 대해 AND 작업을 수행합니다.

IXSCAN

선택 가능한 시작/중지 조건으로 테이블의 인덱스를 스캔하여 일련의 순서화된 행을 산출합니다.

MSJOIN

병합 조인을 표현하며 외부 및 내부 테이블이 모두 조인-술어 순서로 되어야 합니다.

NLJOIN

중첩 루프 조인으로서 각 외부 테이블 행마다 한 번씩 내부 테이블에 액세스합니다.

RETURN

쿼리로부터 사용자에게 데이터를 리턴함을 의미합니다.

RIDSCN

한 개 이상의 인덱스로부터 얻어진 일련의 행 ID(RID)를 스캔합니다.

SHIP 원격 데이터베이스 소스의 데이터를 검색합니다. 페더레이티드 시스템에서 사용됩니다.

SORT 지정된 컬럼의 순서로 행을 정렬하며, 선택사항으로 중복된 항목을 제거할 수 있습니다.

TBSCAN

데이터 페이지에서 직접 필요한 모든 데이터를 읽어 행을 검색합니다.

TEMP

(여러 번) 다시 읽을 수 있도록 데이터를 임시 테이블에 저장합니다.

TQUEUE

데이터베이스 에이전트들 사이에 테이블 데이터를 전송합니다.

UNION

여러 테이블의 행들을 이어 붙입니다.

UNIQUE

특정 컬럼에 대해 중복된 값을 가진 행들을 제거합니다.

UPDATE

테이블에 있는 행들을 갱신합니다.

CMPEXP

연산자 이름: CMPEXP

의미: 중간 또는 최종 결과를 얻는 데 필요한 표현식 계산.

(이 연산자는 디버그 모드용입니다.)

DELETE

연산자 이름: DELETE

의미: 테이블에서 행 삭제.

이 연산자는 필수 연산입니다. 액세스 플랜의 비용을 개선하려면, 일련의 행이 삭제되도록 지정하는 다른 연산자(스캔, 조인)에 집중하십시오.

성능 제안

- 한 테이블에서 모든 행을 삭제하려면, DROP TABLE 명령문이나 **LOAD REPLACE** 명령을 사용해 보십시오.

EISCAN

연산자 이름: EISCAN

의미: 이 연산자는 행 스트림을 줄이기 위해 사용자 정의 인덱스를 스캔합니다. 스캔시 사용자 제공 탐색 범위 생성 함수에서 여러 개의 시작/중지 조건이 사용됩니다.

이 연산은 기본 테이블을 사용하기 전에(술어에 따라) 규정 행의 범위를 좁히는 데 사용됩니다.

성능 제안

- 데이터베이스가 갱신됨에 따라 인덱스가 단편화되어, 필요한 양보다 많은 인덱스 페이지를 차지하게 됩니다. 이것은 인덱스를 삭제하고 다시 만들거나 인덱스를 재구성하여 정정할 수 있습니다.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

FETCH

연산자 이름: FETCH

의미: 특정 행 ID(RID)를 사용하여 테이블에서 컬럼 페치.

성능 제안

- 인덱스 키를 확장하여 페치된 컬럼을 포함시켜 데이터 페이지를 액세스하지 못하게 하십시오.
- 페치와 관련된 인덱스를 찾은 후, 노드를 더블 클릭하십시오. 통계 창이 표시됩니다. 해당 인덱스의 클러스터링 등급이 높은지 확인하십시오.
- 페치로 인한 입출력(I/O)이 테이블의 페이지 수보다 크면 버퍼 크기를 늘리십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

  크기와 빈도에 관한 통계는 술어의 선택성에 관한 정보를 제공하며, 이것은 인덱스 스캔이 테이블 스캔 대신에 사용될 수 있는 시기를 결정합니다. 이러한 통계치를 갱신하려면, WITH DISTRIBUTION절이 있는 테이블의 **runstats** 명령을 사용하십시오.

FILTER

연산자 이름: FILTER

의미: 술어에서 제공하는 기준에 따라 데이터가 필터되도록 하는 레지듀얼(Residual) 술어의 응용프로그램.

성능 제안

- 필요한 데이터만 검색하는 술어를 사용했는지 확인하십시오. 예를 들어, 원하는 테이블의 일부를 나타내는 술어에 대한 선택성 값이 리턴되었는지 확인하십시오.
- 최적화 클래스가 적어도 3이 되어 옵티마이저가 서브쿼리 대신 조인을 사용하도록 하십시오. 만약 이것이 불가능하면, 직접 SQL 쿼리문을 재작성해서 서브쿼리를 제거하십시오. 예를 들어, *관리 안내서*의 SQL 컴파일러에 의한 쿼리 재작성에 관한 절을 참조하십시오.

GENROW

연산자 이름: GENROW

의미: 테이블, 인덱스 또는 연산자로부터의 입력 없이도 테이블 행을 생성하는 내장 함수.

GENROW 기능은 옵티마이저가 데이터 행들을 생성하기 위해 사용할 수도 있습니다(예를 들어, INSERT문이나 조인으로 변형되는 몇 가지 IN-목록 처리시).

GENROW 기능으로 생성된 테이블에 대해 예측된 통계를 열람하려면, 그 노드를 더블 클릭하십시오.

GRPBY

연산자 이름: GRPBY

의미: 지정된 컬럼 또는 함수의 공통 값에 따라 행 그룹화. 이 연산은 한 그룹의 값들을 만들어내거나 또는 집합 함수를 평가할 때 사용됩니다.

GROUP BY 컬럼이 지정되지 않은 경우, SELECT 목록에 총계 함수가 있으면 GRPBY 연산자를 사용할 수 있는데, 총계가 수행되는 동안 전체 테이블이 하나의 그룹으로 취급된다는 것을 나타냅니다.

성능 제안

- 이 연산자는 필수 연산입니다. 액세스 플랜의 비용을 개선하려면, 일련의 행을 그룹으로 정의하는 다른 연산자(스캔, 조인)에 집중하십시오.
- GROUP BY절 없이 단일 집계 함수를 포함하는 SELECT문의 성능을 향상시키려면 다음을 수행하십시오.
  - MIN(C) 집계 함수의 경우, C에 오름차순의 인덱스를 작성하십시오.
  - MAX(C) 집계 함수의 경우, C에 내림차순의 인덱스를 작성하십시오.

HSJOIN

연산자 이름: HSJOIN

의미: 테이블의 내용을 미리 정렬하지 않고도 직접 조인이 가능하도록 테이블의 규정된 행이 해시되는 해시 조인.

FROM절에서 두 개 이상의 테이블이 참조될 경우에는 조인이 반드시 필요합니다. 해시 조인은 두 개의 다른 테이블의 컬럼에 해당하는 조인 술어가 있을 때마다 가능해집니다. 조인 술어들의 데이터 유형은 정확히 동일해야 합니다. 해시 조인은 NLJOIN을 사용하는 경우처럼 재작성된 서브쿼리로부터 발생할 수도 있습니다.

해시 조인을 위해 입력 테이블을 재정렬할 필요는 없습니다. 이 조인은 해시 조인의 내부 테이블을 스캔하고 조인 컬럼 값을 해시하여 찾아보기 테이블을 생성함으로 수행됩니다. 그런 다음 외부 테이블을 읽고 그 조인 컬럼 값을 해시한 다음 내부 테이블에 대해 생성된 찾아보기 테이블을 점검하는 작업이 수행됩니다.

자세한 내용은 *관리 안내서*의 조인 개념에 관한 절을 참조하십시오.

성능 제안

- 조인할 행의 수를 줄이려면, 로컬 술어(즉, 한 테이블을 참조하는 술어)를 사용하십시오.
- 정렬 힙(heap)의 *크기*를 메모리에 해시 찾아보기 테이블을 보유할 수 있을 정도로 *크게* 늘리십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

INSERT

연산자 이름: INSERT

의미: 테이블에 행 삽입.

이 연산자는 필수 연산입니다. 액세스 플랜의 비용을 줄이려면, 삽입될 일련의 행을 정의하는 다른 연산자(스캔, 조인)에 집중하십시오.

IXAND

연산자 이름: IXAND

의미: 동적 비트맵 기술을 사용하여 여러 인덱스 스캔의 결과를 AND 처리하는 것. 이 연산자는 대상이 되는 테이블 사용을 최소화하기 위하여 AND 처리가 여러 개의 인덱스에 적용되도록 합니다.

이 연산자는 다음과 같은 경우 수행됩니다.
- 기본 테이블을 사용하기 전에 관련된 행의 범위를 좁히려는 경우
- 복수 인덱스에 적용되는 술어들을 함께 AND 처리하는 경우
- 스타 조인에서 사용되는 세미 조인의 결과를 함께 AND 처리하는 경우

성능 제안

- 데이터베이스가 갱신됨에 따라 인덱스가 단편화되어, 필요한 양보다 많은 인덱스 페이지를 차지하게 됩니다. 이것은 인덱스를 삭제하고 다시 만들거나 인덱스를 재구성하여 정정할 수 있습니다.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.
- 일반적으로, 인덱스 스캔은 규정화된 행이 적을 때 가장 효과가 있습니다. 규정화된 행의 수를 예측하기 위해 옵티마이저는 술어가 참조하는 컬럼에 관한 통계치를 사용합니다. 어떤 값이 다른 값보다 자주 나타난다면, **runstats** 명령의 WITH DISTRIBUTION절을 사용하여 분산 통계를 구하는 것이 중요합니다. 비정상 분산 통계를 이용하여, 옵티마이저는 자주 나타나는 값과 자주 나타나지 않는 값들을 구분할 수 있습니다.

- IXAND의 사용에는 시작 키와 중지 키가 중요하므로, IXAND는 한 컬럼으로 되어 있는 인덱스에 가장 효과적입니다.
- 스타 조인의 경우, 사실(Fact) 테이블 및 관련된 차원 테이블에서 가장 선택적인 컬럼 각각에 대해 단일 컬럼 인덱스를 작성합니다.

IXSCAN

연산자 이름: IXSCAN

의미: 행 스트림을 줄이기 위한 인덱스 스캔. 이 스캔은 시작/중지 조건을 가질 수 있으며, 인덱스의 컬럼을 참조하는 인덱스 가능한 술어에도 적용될 수 있습니다.

이 연산은 기본 테이블을 사용하기 전에(술어에 따라) 규정 행의 범위를 좁히는 데 사용됩니다.

자세한 내용은 *관리 안내서*의 인덱스 스캔에 관한 절을 참조하십시오.

성능 제안

- 데이터베이스가 갱신됨에 따라 인덱스가 단편화되어, 필요한 양보다 많은 인덱스 페이지를 차지하게 됩니다. 이것은 인덱스를 삭제하고 다시 만들거나 인덱스를 재구성하여 정정할 수 있습니다.
- 두 개 이상의 테이블이 사용될 경우, 외부 테이블의 조인 컬럼에 있는 인덱스를 제공하여 인덱스를 통해 내부 조인 테이블에 액세스하는 것을 더 효율적으로 할 수 있습니다.

  인덱스에 대한 자세한 지침은 Visual Explain에 대한 온라인 도움말을 참조하십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 갱신하십시오.
- 일반적으로, 인덱스 스캔은 규정된 행이 적을 때 가장 효과가 있습니다. 규정된 행의 수를 예측하기 위해 옵티마이저는 술어가 참조하는 컬럼에 관한 통계치를 사용합니다. 어떤 값이 다른 값보다 자주 나타난다면, **runstats** 명령의 WITH DISTRIBUTION절을 사용하여 분산 통계를 구하는 것이 중요합니다. 비정상 분산 통계를 이용하여, 옵티마이저는 자주 나타나는 값과 자주 나타나지 않는 값들을 구분할 수 있습니다.

MSJOIN

연산자 이름: MSJOIN

의미: 외부 및 내부 테이블 모두의 규정된 행이 조인-술어 순으로 있어야 하는 병합 조인. 병합 조인을 *병합 스캔 조인* 또는 *정렬 병합 조인*이라고도 합니다.

FROM절에서 두 개 이상의 테이블이 참조될 경우에는 조인이 반드시 필요합니다. 두 개의 다른 테이블의 컬럼에 해당하는 조인 술어가 있을 경우, 병합 조인이 가능합니다. 이것은 또한 재작성된 서브쿼리로부터 발생할 수도 있습니다

병합 조인을 할 때에는 테이블들이 일반적으로 한번만 스캔되므로 컬럼을 조인할 때 입력이 일정한 순서대로 이루어져야 합니다. 순서화된 입력은 인덱스나 정렬된 테이블에 액세스함으로써 이루어집니다.

자세한 내용은 *관리 안내서*의 조인 개념에 관한 절을 참조하십시오.

성능 제안

- 조인할 행의 수를 줄이려면, 로컬 술어(즉, 한 테이블을 참조하는 술어)를 사용하십시오.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

NLJOIN

연산자 이름: NLJOIN

의미: 외부 테이블의 각 행마다 한 번씩 내부 테이블을 스캔하는(보통 인덱스 스캔으로) 중첩 루프 조인.

FROM절에서 두 개 이상의 테이블이 참조될 경우에는 조인이 반드시 필요합니다. 중첩 루프 조인에는 조인 술어가 필요없지만 있으면 더 잘 수행됩니다.

중첩 루프 조인은 다음 중 하나로 수행됩니다.

- 외부 테이블에서 사용된 행마다 내부 테이블을 스캔.
- 외부 테이블에서 사용된 행마다 내부 테이블의 인덱스를 검색.

자세한 내용은 *관리 안내서*의 조인 개념에 관한 절을 참조하십시오.

성능 제안

- 중첩 루프 조인은 내부 테이블(NLJOIN 연산자 오른쪽에 있는 테이블)의 조인-술어 컬럼에 인덱스가 있으면 보다 효율적으로 수행됩니다. 내부 테이블이 IXSCAN이 아니라 TBSCAN인지 확인하십시오. 그렇다면, 조인 컬럼에 인덱스를 추가해 보십시오.

  조인을 보다 효율적으로 하는 또 다른 방법은 외부 테이블이 정렬되도록 조인 컬럼에 대한 인덱스를 작성하는 것입니다.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

관련 정보:

- 스타 조인.

PIPE

연산자 이름: PIPE

의미: 행을 변경하지 않고 다른 연산자로 행 전송.

(이 연산자는 디버그 모드용입니다.)

RETURN

연산자 이름: RETURN

의미: 쿼리에서 사용자에게 데이터를 리턴. 이것은 액세스 플랜 그래프의 마지막 연산자이며 액세스 플랜의 최종 합계와 비용을 보여줍니다.

이 연산자는 필수 연산입니다.

성능 제안

- 필요한 데이터만 검색하는 술어를 사용했는지 확인하십시오. 예를 들어, 원하는 테이블의 일부를 나타내는 술어에 대한 선택성 값이 리턴되었는지 확인하십시오.

RIDSCN

연산자 이름: RIDSCN

의미: 하나 이상의 인덱스에서 확보한 행 ID(RID) 목록 스캔.

이 연산자는 옵티마이저에서 다음의 경우에 고려됩니다.

- 술어가 OR 키워드로 연결되거나, IN 술어가 있습니다. 인덱스 OR 처리(ORing)라는 기술을 사용하여, 같은 테이블의 여러 인덱스에 액세스한 결과를 조합할 수 있습니다.
- 단일 인덱스에 액세스하는 경우, 목록 프리페치를 사용하는 것이 좋습니다. 기본 행을 사용하기 전에 행 ID를 정렬하는 것이 I/O를 더욱 효율적으로 만들기 때문입니다.

RQUERY

연산자 이름: SHIP

의미: 리모트 데이터 소스에서 데이터를 검색하기 위해 페더레이티드 시스템에서 사용되는 연산자. 이 연산자는 SHIP 연산자가 쿼리 결과를 검색하기 위해 SQL SELECT문을 리모트 데이터 소스로 보낼 때 옵티마이저에서 고려합니다. SELECT문은 데이터 소스가 제공하는 SQL문을 사용하여 생성되며, 데이터 소스에서 허용하는 유효 쿼리를 포함할 수 있습니다.

성능 제안 관리 안내서 볼륨 2, 페더레이티드 데이터베이스 쿼리 및 네트워크 조정 정보의 제4장을 참조하십시오.

SORT

연산자 이름: SORT

의미: 테이블의 행을 하나 이상의 컬럼 순서로 정렬하는 것을 의미하며, 선택적으로 중복된 항목을 삭제할 수도 있습니다.

요청된 순서로 정렬된 인덱스가 없거나 인덱스 스캔보다 정렬 비용이 저렴한 경우, 정렬이 필요합니다. 일반적으로 정렬은 필요한 행이 페치되면 최종 작업으로 수행되거나, 조인이나 그룹으로 나누기 전에 데이터를 정렬하기 위해 사용됩니다.

행이 많거나 정렬된 데이터를 파이프할 수 없는 경우, 임시 테이블을 만들 때 비용이 많이 필요합니다.

정렬에 관한 자세한 내용은 *관리 안내서*를 참조하십시오.

성능 제안

- 정렬 컬럼에 대해 인덱스를 추가하는 것을 고려해 보십시오.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.
- 필요한 데이터만 검색하는 술어를 사용했는지 확인하십시오. 예를 들어, 원하는 테이블의 일부를 나타내는 술어에 대한 선택성 값이 리턴되었는지 확인하십시오.
- 시스템 임시 테이블 스페이스의 프리페치 *크기*가 적절한지, 즉 입출력이 바운드되지 않았는지 점검하십시오(이를 점검하려면, **명령문 -> 통계 표시 -> 테이블 스페이스**를 선택하십시오).
- 대용량의 정렬이 자주 필요하면, 다음 구성 매개변수의 값을 늘리는 것을 고려해 보십시오.
  - 정렬 힙 *크기*(sortheap). 이 매개변수를 변경하려면, 제어 센터에서 데이터베이스를 오른쪽 마우스 단추로 선택한 후, 그 팝업 메뉴에서 *구성*을 선택하십시오. 노트북이 나타나면 **성능** 탭을 선택하십시오.

- 정렬 힙 임계값(sheapthres). 이 매개변수를 변경하려면, 제어 센터 창에서 데이터베이스 인스턴스를 오른쪽 마우스 단추로 선택하고, 그 팝업 메뉴에서 *구성*을 선택하십시오. 노트북이 나타나면 **성능** 탭을 선택하십시오.

• 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

TBSCAN

연산자 이름: TBSCAN

의미: 데이터베이스에서 직접 필요한 모든 데이터를 읽어 행을 검색하는 테이블 스캔(관련 스캔).

이 유형의 스캔은 다음과 같은 경우처럼 인덱스 스캔시에 옵티마이저가 선택합니다.

• 스캔된 범위의 값들이 자주 나타날 때(즉, 테이블의 대부분에 액세스해야 할 경우)
• 테이블이 작을 때
• 인덱스의 클러스터링이 낮을 때
• 인덱스가 없을 때

테이블 및 인덱스 스캔에 관한 자세한 내용은 *관리 안내서*를 참조하십시오

성능 제안

• 테이블이 크고 대부분의 테이블 행이 액세스되지 않는 경우, 인덱스 스캔이 테이블 스캔보다 효율적입니다. 이런 경우 옵티마이저가 인덱스 스캔을 사용할 가능성을 높이려면, 선택적 술어가 있는 컬럼에 인덱스를 추가해 보십시오.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.

• 인덱스가 있으나 사용되지 않는 경우에는 각 선두 컬럼에 선택적 술어가 있는지 점검하십시오. 이 술어가 있는 경우, 인덱스에 대한 클러스터링 등급이 높은지 확인하십시오(이 통계치를 보려면, 정렬 아래의 테이블에 대한 테이블 통계 창을 열고 인덱스 통계 창을 보기 위해 *인덱스* 누름 단추를 누르십시오).

- 테이블 스페이스의 프리페치 크기가 적절한지, 즉 I/O에 치중되어 있는지 확인하십시오(이를 점검하려면, **명령문 -> 통계 표시 -> 테이블 스페이스**를 선택하십시오).

  자세한 내용은 *관리 안내서*의 버퍼 풀로 데이터 프리페칭에 관한 절을 참조하십시오 .

- 통계치가 현재의 것이 아닌 경우, runstats 명령을 사용하여 갱신하십시오.

  수량 및 빈도에 관한 통계는 술어의 선택성에 관한 정보를 제공합니다. 예를 들면, 이들 통계는 인덱스 스캔을 테이블 스캔 대신 사용할 시기를 결정하는 데 사용할 수 있습니다. 통계치를 갱신하려면, WITH DISTRIBUTION절을 포함하는 테이블에서 **runstats** 명령을 사용하십시오.

TEMP

연산자 이름: TEMP

의미: 다른 연산자가 다시 읽을 수 있도록(여러 번) 임시 테이블에 데이터를 저장하는 조치. 이 테이블은 SQL문이 처리되기 전에 제거되지 않았다면 SQL문이 처리된 후에 제거됩니다.

이 연산자는 중간 결과를 저장하거나 서브쿼리를 판단하는 데 필요합니다. 경우(명령문이 갱신될 수 있는 경우와 같이)에 따라 반드시 필요하기도 합니다.

TQUEUE

연산자 이름: TQUEUE

의미: 쿼리를 처리하는 에이전트가 여러 개 있는 경우 한 데이터베이스 에이전트에서 다른 데이터베이스 에이전트로 테이블 데이터를 전달하는 데 사용되는 테이블 큐. 병렬 처리될 경우, 쿼리 하나를 처리하는 데 여러 개의 데이터베이스 에이전트가 사용됩니다.

테이블 큐 유형은 다음과 같습니다.

- **로컬:** 테이블 큐가 단일 노드 내의 데이터베이스 에이전트 사이에서 데이터를 전달하는 데 사용됩니다. 로컬 테이블 큐는 파티션내 병렬 처리에 사용됩니다.

- **로컬이 아님:** 테이블 큐가 서로 다른 노드의 데이터베이스 에이전트 사이에서 데이터를 전달하는 데 사용됩니다.

UNION

연산자 이름: UNION

의미: 여러 테이블로부터의 행 스트림의 병합.

이 연산자는 필수 연산입니다. 액세스 플랜 비용을 줄이려면, 병합할 일련의 행을 정의하는 다른 연산자(스캔, 조인)에 집중하십시오.

UNIQUE

연산자 이름: UNIQUE

의미: 지정된 컬럼에 중복 값이 있는 행 삭제.

성능 제안

- 이 연산자는 적절한 컬럼에 고유한 인덱스가 있는 경우에만 필수 연산이 아닙니다.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.

UPDATE

연산자 이름: UPDATE

의미: 테이블 행의 데이터 갱신.

이 연산자는 필수 연산입니다. 액세스 플랜 비용을 줄이려면, 병합할 일련의 행을 정의하는 다른 연산자(스캔, 조인)에 집중하십시오.

옵티마이저

*옵티마이저*는 SQL 컴파일러의 한 부분으로 DML SQL문을 위한 액세스 플랜을 선택하는 일을 합니다. 이것은 여러 액세스 플랜의 실행 비용을 모델링하여 예상 비용이 가장 적은 것을 선택합니다.

패키지

*패키지*는 데이터베이스에 저장된 오브젝트로서, 응용프로그램의 한 소스 파일에 연관된 SQL문을 처리하는 데 필요한 정보를 가지고 있습니다. 패키지는 다음 방법으로 생성됩니다.

- **PREP** 명령으로 소스 파일을 프리컴파일.
- **BIND** 명령으로 프리컴파일러에 의해 만들어진 바인드 파일을 바인딩.

술어

술어는 탐색 조건의 한 요소로서 비교 조작을 나타내거나 암시합니다. 술어는 WHERE나 HAVING으로 시작하는 절에 포함되어 있습니다.

예를 들면, 다음의 SQL문에서:

```
SELECT * FROM SAMPLE
  WHERE NAME = 'SMITH' AND
  DEPT = 895 AND YEARS > 5
```

술어들은 다음과 같습니다. NAME = 'SMITH'; DEPT = 895; and YEARS > 5.

술어는 가장 효율적인 것부터 비효율적인 것 순으로 다음 중 하나의 범주에 속합니다.

1. 인덱스 스캔 범위를 제한(범위를 좁힘)하는 시작 및 중지 조건(이 조건들을 범위 구분 술어라고도 합니다).
2. 인덱스-페이지(sargable 인덱스라고도 함) 술어는 술어에 포함된 컬럼이 인덱스 키의 일부이므로 인덱스에서 검사됩니다.
3. 데이터-페이지(sargable 데이터라고도 함) 술어는 인덱스에서 검사될 수는 없고, 행이 버퍼에 있는 동안 검사될 수 있습니다.

4. 레지듀얼(Residual) 술어는 단순히 기본 테이블을 사용하는 것 이상의 입/출력을 필요로 하며, 데이터가 버퍼 페이지로부터 복사된 이후에 사용되어야 합니다. 이들은 서브쿼리를 포함하는 술어나, 테이블로부터 별도의 파일에 저장된 LONG VARCHAR 또는 LOB 데이터를 읽는 술어를 포함합니다.

술어를 설계할 때는 가능한 적은 행이 추출되도록 최고의 선택성을 목표로 해야 합니다.

다음의 술어 유형이 가장 효율적이며 가장 일반적으로 사용됩니다.

- *단순 동등 조인 술어*는 병합 조인에 필요합니다.
  이것은 table1.column = table2.column의 형태를 가지고 있으며, 테이블들이 조인될 수 있도록 다른 두 테이블의 컬럼들을 비교할 수 있도록 합니다.
- *로컬 술어*는 하나의 테이블에만 적용됩니다.

자세한 내용은 *관리 안내서*의 데이터 액세스 개념 및 최적화에 관한 절을 참조하십시오.

쿼리 최적화 클래스

*쿼리 최적화 클래스*는 쿼리 컴파일시의 최적화 기술 및 쿼리 재작성 규칙 세트입니다.

1차 쿼리 최적화 클래스는 다음과 같습니다.

1 제한된 최적화. 메모리 및 프로세싱 자원이 심하게 부족할 때 유용합니다. 버전 1에서 제공되는 최적화와 거의 비슷합니다.

2 약간의 최적화. 특히 매우 복잡한 쿼리의 경우, 버전 1의 최적화 비용보다는 높고 레벨 3 이상보다는 비용이 적게 드는 최적화 레벨을 지정합니다.

3 중간 최적화. MVS/ESA용 DB2의 쿼리 최적화 특성과 가장 유사합니다.

5 일반 최적화. 단순 트랜잭션 및 복잡한 쿼리가 혼합된 상황에서 쓰는 것이 좋습니다.

7 일반 최적화. 최적화 5단계와 같으나 복잡한 동적 SQL 쿼리에 대한 쿼리 최적화의 정도를 줄이지 않는다는 점에서 다릅니다.

특별한 상황에서만 사용되는 다른 쿼리 최적화 클래스는 다음과 같습니다.

0 최소 최적화. 최적화가 거의 또는 전혀 필요없을 때만 사용하십시오(즉, 인덱스가 잘 작성된 테이블에 대한 단순한 쿼리의 경우).

9 최대 최적화. 많은 메모리와 프로세싱 자원을 사용합니다. 5단계로 불충분할 경우에만 사용하십시오(즉, 5단계로 잘 수행되지 않는 매우 복잡하고 오래 수행되는 쿼리의 경우).

일반적으로, 정적 쿼리 및 오래 걸릴 것이 예상되는 쿼리에는 높은 최적화 클래스를, 동적으로 시작되는 단순 쿼리나 자주 사용되지 않는 쿼리에는 낮은 최적화 클래스를 사용하십시오.

동적 SQL문에 쿼리 최적화를 설정하려면, 명령행 처리기에 다음 명령을 입력하십시오.

```
SET CURRENT QUERY OPTIMIZATION = n;
```

여기서, 'n'은 원하는 쿼리 최적화 클래스입니다.

정적 SQL문에 대해 쿼리 최적화를 설정하려면, **BIND**나 **PREP** 명령에서 QUERYOPT 옵션을 사용하십시오.

자세한 내용은 *관리 안내서*의 최적화 클래스 조정에 관한 절을 참조하십시오.

술어의 선택성

*선택성*이란, 행이 술어를 만족시킬(즉, 참일) 가능성을 가리킵니다.

예를 들면, 1,000,000개의 행을 가진 테이블에 대해 수행되는 술어의 선택성이 0.01(1%)인 경우, 그 술어가 (1,000,000의 1%인) 약 10,000개의 행을 리턴하고 990,00개의 행은 버린다는 것을 의미합니다.

(선택성이 0.10 이하인) 매우 선택적인 술어가 바람직합니다. 그러한 술어는 다음 연산자가 작업할 행을 적게 리턴하므로, 쿼리를 수행하는 데 필요한 CPU 및 입/출력 수가 적어집니다.

예

1,000,000개의 행을 가진 테이블이 있고, 원래 쿼리에 추가로 정렬을 해야 하는 'ORDER BY'절이 있다고 가정해 보십시오. 선택성이 0.01인 술어에서 정렬 작업은 10,000개의 행에 대해 필요할 것입니다. 그러나 0.50의 덜 선택적인 술어로 예상된 500,000행에 대해 정렬이 수행되어야 하므로 더 많은 CPU와 I/O 시간이 필요합니다.

스타 조인

사실(fact) 테이블(큰 중심 테이블)이 두 개 이상의 차원(dimension) 테이블(사실 테이블에 컬럼 값의 설명이 들어 있는 더 작은 테이블)로 조인될 때 그 조인 세트를 스타 조인이라고 합니다.

스타 조인은 다음의 세 가지 주요 부분으로 구성됩니다.
- 세미 조인
- 세미 조인의 결과에 대한 AND 작업을 인덱스 처리
- 세미 조인 완료

이것은 IXAND 연산자를 제공하는 둘 이상의 조인으로 나타납니다.

세미 조인은 조인의 결과가 내부 및 외부 테이블 컬럼의 조인이 아닌 내부 테이블의 행 ID(RID)로만 나타나는 특수 형태의 조인입니다.

스타 조인은 세미 조인을 사용하여 행 ID를 인덱스 ANDing 연산자로 제공합니다. 인덱스 ANDing 연산자는 다양한 조인의 필터링 결과를 축적합니다. 인덱스 ANDing 연산자로부터 나온 출력은 인덱스 ORing 연산자로 제공되는 데, 이 연산자는 행 ID를 정렬하고 인덱스 ANDing 연산자를 제공하는 조인으로부터 발생할 수 있는 중복 행을 없앱니다. 그런 다음 이 사실(fact) 테이블의 행은 페치 연산자를 사용하여 페치됩니다. 마지막으로 감소된 사실(fact) 테이블은 모든 차원 테이블로 조인되어 조인이 완료됩니다.

성능 제안
- 각 차원 테이블 조인에 대한 사실(fact) 테이블에 인덱스를 작성하십시오.

- 정렬 힙 임계값이 인덱스 ANDing 연산자의 비트 필터를 할당할 만큼 충분히 높은지 확인하십시오. 스타 조인의 경우, 이 값은 12MB 또는 3000 4K 페이지 이상이어야 합니다. 파티션 내 병렬 처리의 경우, 비트 필터는 dbheap의 경우와 동일한 공유 메모리 세그먼트로부터 할당되지만 sortheap(및 인스턴스에서의 sheapthres)으로 바운드됩니다. 따라서 공유 메모리는 sortheap 및 sheapthres에 의해 제어되며 12MB가 넘어야 합니다.
- 차원 테이블에 대해 필터링 술어를 적용하십시오. 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

정적 SQL

정적 SQL 명령문은 응용프로그램 안에 내장되어 있습니다. 이 모든 명령문들은 프리컴파일되어 응용프로그램이 실행되기 전에 *패키지* 안에 바인드되어야 합니다.

DB2는 이 명령문들을 컴파일할 때, 프리컴파일되고 바인드될 당시의 카탈로그 통계 및 구성 매개변수에 의해 명령문마다 액세스 플랜을 작성합니다.

이들 액세스 플랜은 응용프로그램이 실행될 때 항상 사용되며, 패키지가 다시 바인드될 때까지는 변하지 않습니다.

정적 SQL 대신 동적 SQL을 사용할 수도 있습니다.

시스템 관리 스페이스(SMS) 테이블 스페이스

데이터베이스에는 시스템 관리 스페이스(SMS)와 데이터베이스 관리 스페이스(DMS)라는 두 가지 유형의 테이블 스페이스가 있습니다.

SMS 테이블 스페이스는 운영 체제에 의해 관리되며, 운영 체제는 테이블 스페이스가 작성될 때 할당되는 공간에 데이터베이스 데이터를 저장합니다. 테이블 스페이스 정의에는 이 데이터가 저장되는 하나 이상의 디렉토리 경로 목록이 들어 있습니다.

파일 시스템은 저장 미디어의 할당 및 관리를 담당합니다.

SMS 테이블 스페이스와 DMS 테이블 스페이스는 동일한 데이터베이스에 공존할 수 있습니다.

테이블 스페이스

대형 데이터베이스의 경우, *테이블 스페이스*라는 별도로 관리되는 부분으로 분할하면 관리하기가 쉽습니다.

테이블 스페이스는 데이터의 위치를 특정한 논리 디바이스나 디바이스의 한 부분에 배정할 수 있도록 합니다. 예를 들면, 테이블을 만들 때 인덱스나 길거나 큰 오브젝트(LOB)를 가진 긴 컬럼은 테이블의 나머지와 다른 곳에 저장되도록 할 수 있습니다.

테이블 스페이스는 성능 향상을 위해 한 개 이상의 물리 스토리지 디바이스(컨테이너)에 분산시킬 수 있습니다. 그러나 한 테이블 스페이스에 있는 디바이스나 컨테이너들은 비슷한 성능 특성을 갖는 것이 좋습니다.

테이블 스페이스는 시스템 관리 스페이스(SMS)나 데이터베이스 관리 스페이스(DMS), 두 가지 방식으로 관리할 수 있습니다.

Visual Explain

주: 버전 6의 경우, 더 이상 명령 행에서 Visual Explain을 호출할 수 없습니다. 그러나 제어 센터에 있는 다양한 데이터베이스 오브젝트로부터 호출할 수는 있습니다. 이 버전에 대해 문서에서는 여전히 Visual Explain의 이름을 사용합니다.

Visual Explain을 통해 Explain된 SQL문에 대한 액세스 플랜을 그래프로 볼 수 있습니다. 그래프에서 사용 가능한 정보를 사용하여 더 나은 성능을 위해 사용자의 SQL 쿼리를 튜닝할 수 있습니다.

액세스 플랜 그래프는 다음과 같은 세부사항을 나타냅니다.
- 테이블(및 그와 관련된 컬럼)과 인덱스
- 연산자(테이블 스캔, 정렬 및 조인과 같은)
- 테이블 스페이스 및 함수

Visual Explain을 사용하여 다음을 수행할 수도 있습니다.

- 최적화시 사용된 통계를 볼 수 있습니다. 그런 다음 이러한 통계를 현재 카탈로그 통계와 비교하여 패키지 리바인딩으로 성능이 향상되는지 여부를 판별하는 데 도움을 줄 수 있습니다.
- 인덱스가 테이블 액세스에 사용되었는지 여부를 판별할 수 있습니다. 인덱스가 사용되지 않은 경우, Visual Explain은 어떤 컬럼이 인덱스되면 이득이 되는지 판별하는 데 도움을 줄 수 있습니다.
- 쿼리에 대한 액세스 플랜 그래프의 이전 및 이후 버전을 비교함으로써 다양한 튜닝 기술의 수행 효과를 볼 수 있습니다.
- 전체 예상 비용 및 검색된 행의 수(카디널리티)를 포함하여 액세스 플랜에서 각 조작에 관한 정보를 확보할 수 있습니다.

부록 B. Visual Explain 연산자의 영문자순 목록

CMPEXP

연산자 이름: CMPEXP

의미: 중간 또는 최종 결과를 얻는 데 필요한 표현식 계산.

(이 연산자는 디버그 모드용입니다.)

DELETE

연산자 이름: DELETE

의미: 테이블에서 행 삭제.

이 연산자는 필수 연산입니다. 액세스 플랜의 비용을 개선하려면, 일련의 행이 삭제되도록 지정하는 다른 연산자(스캔, 조인)에 집중하십시오.

성능 제안

- 한 테이블에서 모든 행을 삭제하려면, DROP TABLE 명령문이나 **LOAD REPLACE** 명령을 사용해 보십시오.

EISCAN

연산자 이름: EISCAN

의미: 이 연산자는 행 스트림을 줄이기 위해 사용자 정의 인덱스를 스캔합니다. 스캔시 사용자 제공 탐색 범위 생성 함수에서 여러 개의 시작/중지 조건이 사용됩니다.

이 연산은 기본 테이블을 사용하기 전에(술어에 따라) 규정 행의 범위를 좁히는 데 사용됩니다.

성능 제안

- 데이터베이스가 갱신됨에 따라 인덱스가 단편화되어, 필요한 양보다 많은 인덱스 페이지를 차지하게 됩니다. 이것은 인덱스를 삭제하고 다시 만들거나 인덱스를 재구성하여 정정할 수 있습니다.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

FETCH

연산자 이름: FETCH

의미: 특정 행 ID(RID)를 사용하여 테이블에서 컬럼 페치.

성능 제안

- 인덱스 키를 확장하여 페치된 컬럼을 포함시켜 데이터 페이지를 액세스하지 못하게 하십시오.
- 페치와 관련된 인덱스를 찾은 후, 노드를 더블 클릭하십시오. 통계 창이 표시됩니다. 해당 인덱스의 클러스터링 등급이 높은지 확인하십시오.
- 페치로 인한 입출력(I/O)이 테이블의 페이지 수보다 크면 버퍼 크기를 늘리십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

  크기와 빈도에 관한 통계는 술어의 선택성에 관한 정보를 제공하며, 이것은 인덱스 스캔이 테이블 스캔 대신에 사용될 수 있는 시기를 결정합니다. 이러한 통계치를 갱신하려면, WITH DISTRIBUTION절이 있는 테이블의 **runstats** 명령을 사용하십시오.

FILTER

연산자 이름: FILTER

의미: 술어에서 제공하는 기준에 따라 데이터가 필터되도록 하는 레지듀얼(Residual) 술어의 응용프로그램.

성능 제안

- 필요한 데이터만 검색하는 술어를 사용했는지 확인하십시오. 예를 들어, 원하는 테이블의 일부를 나타내는 술어에 대한 선택성 값이 리턴되었는지 확인하십시오.
- 최적화 클래스가 적어도 3이 되어 옵티마이저가 서브쿼리 대신 조인을 사용하도록 하십시오. 만약 이것이 불가능하면, 직접 SQL 쿼리문을 재작성해서 서브쿼리를 제거하십시오. 예를 들어, *관리 안내서*의 SQL 컴파일러에 의한 쿼리 재작성에 관한 절을 참조하십시오.

GENROW

연산자 이름: GENROW

의미: 테이블, 인덱스 또는 연산자로부터의 입력 없이도 테이블 행을 생성하는 내장 함수.

GENROW 기능은 옵티마이저가 데이터 행들을 생성하기 위해 사용할 수도 있습니다(예를 들어, INSERT문이나 조인으로 변형되는 몇 가지 IN-목록 처리시).

GENROW 기능으로 생성된 테이블에 대해 예측된 통계를 열람하려면, 그 노드를 더블 클릭하십시오.

GRPBY

연산자 이름: GRPBY

의미: 지정된 컬럼 또는 함수의 공통 값에 따라 행 그룹화. 이 연산은 한 그룹의 값들을 만들어내거나 또는 집합 함수를 평가할 때 사용됩니다.

GROUP BY 컬럼이 지정되지 않은 경우, SELECT 목록에 총계 함수가 있으면 GRPBY 연산자를 사용할 수 있는데, 총계가 수행되는 동안 전체 테이블이 하나의 그룹으로 취급된다는 것을 나타냅니다.

성능 제안

- 이 연산자는 필수 연산입니다. 액세스 플랜의 비용을 개선하려면, 일련의 행을 그룹으로 정의하는 다른 연산자(스캔, 조인)에 집중하십시오.
- GROUP BY절 없이 단일 집계 함수를 포함하는 SELECT문의 성능을 향상시키려면 다음을 수행하십시오.
  - MIN(C) 집계 함수의 경우, C에 오름차순의 인덱스를 작성하십시오.
  - MAX(C) 집계 함수의 경우, C에 내림차순의 인덱스를 작성하십시오.

HSJOIN

연산자 이름: HSJOIN

의미: 테이블의 내용을 미리 정렬하지 않고도 직접 조인이 가능하도록 테이블의 규정된 행이 해시되는 해시 조인.

FROM절에서 두 개 이상의 테이블이 참조될 경우에는 조인이 반드시 필요합니다. 해시 조인은 두 개의 다른 테이블의 컬럼에 해당하는 조인 술어가 있을 때마다 가능해집니다. 조인 술어들의 데이터 유형은 정확히 동일해야 합니다. 해시 조인은 NLJOIN을 사용하는 경우처럼 재작성된 서브쿼리로부터 발생할 수도 있습니다.

해시 조인을 위해 입력 테이블을 재정렬할 필요는 없습니다. 이 조인은 해시 조인의 내부 테이블을 스캔하고 조인 컬럼 값을 해시하여 찾아보기 테이블을 생성함으로 수행됩니다. 그런 다음 외부 테이블을 읽고 그 조인 컬럼 값을 해시한 다음 내부 테이블에 대해 생성된 찾아보기 테이블을 점검하는 작업이 수행됩니다.

자세한 내용은 *관리 안내서*의 조인 개념에 관한 절을 참조하십시오.

성능 제안

- 조인할 행의 수를 줄이려면, 로컬 술어(즉, 한 테이블을 참조하는 술어)를 사용하십시오.
- 정렬 힙(heap)의 *크기*를 메모리에 해시 찾아보기 테이블을 보유할 수 있을 정도로 *크게* 늘리십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

INSERT

연산자 이름: INSERT

의미: 테이블에 행 삽입.

이 연산자는 필수 연산입니다. 액세스 플랜의 비용을 줄이려면, 삽입될 일련의 행을 정의하는 다른 연산자(스캔, 조인)에 집중하십시오.

IXAND

연산자 이름: IXAND

의미: 동적 비트맵 기술을 사용하여 여러 인덱스 스캔의 결과를 AND 처리하는 것. 이 연산자는 대상이 되는 테이블 사용을 최소화하기 위하여 AND 처리가 여러 개의 인덱스에 적용되도록 합니다.

이 연산자는 다음과 같은 경우 수행됩니다.
- 기본 테이블을 사용하기 전에 관련된 행의 범위를 좁히려는 경우
- 복수 인덱스에 적용되는 술어들을 함께 AND 처리하는 경우
- 스타 조인에서 사용되는 세미 조인의 결과를 함께 AND 처리하는 경우

성능 제안

- 데이터베이스가 갱신됨에 따라 인덱스가 단편화되어, 필요한 양보다 많은 인덱스 페이지를 차지하게 됩니다. 이것은 인덱스를 삭제하고 다시 만들거나 인덱스를 재구성하여 정정할 수 있습니다.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.
- 일반적으로, 인덱스 스캔은 규정화된 행이 적을 때 가장 효과가 있습니다. 규정화된 행의 수를 예측하기 위해 옵티마이저는 술어가 참조하는 컬럼에 관한 통계치를 사용합니다. 어떤 값이 다른 값보다 자주 나타난다면, **runstats** 명령의 WITH DISTRIBUTION절을 사용하여 분산 통계를 구하는 것이 중요합니다. 비정상 분산 통계를 이용하여, 옵티마이저는 자주 나타나는 값과 자주 나타나지 않는 값들을 구분할 수 있습니다.

- IXAND의 사용에는 시작 키와 중지 키가 중요하므로, IXAND는 한 컬럼으로 되어 있는 인덱스에 가장 효과적입니다.
- 스타 조인의 경우, 사실(Fact) 테이블 및 관련된 차원 테이블에서 가장 선택적인 컬럼 각각에 대해 단일 컬럼 인덱스를 작성합니다.

IXSCAN

연산자 이름: IXSCAN

의미: 행 스트림을 줄이기 위한 인덱스 스캔. 이 스캔은 시작/중지 조건을 가질 수 있으며, 인덱스의 컬럼을 참조하는 인덱스 가능한 술어에도 적용될 수 있습니다.

이 연산은 기본 테이블을 사용하기 전에(술어에 따라) 규정 행의 범위를 좁히는 데 사용됩니다.

자세한 내용은 *관리 안내서*의 인덱스 스캔에 관한 절을 참조하십시오.

성능 제안

- 데이터베이스가 갱신됨에 따라 인덱스가 단편화되어, 필요한 양보다 많은 인덱스 페이지를 차지하게 됩니다. 이것은 인덱스를 삭제하고 다시 만들거나 인덱스를 재구성하여 정정할 수 있습니다.
- 두 개 이상의 테이블이 사용될 경우, 외부 테이블의 조인 컬럼에 있는 인덱스를 제공하여 인덱스를 통해 내부 조인 테이블에 액세스하는 것을 더 효율적으로 할 수 있습니다.

  인덱스에 대한 자세한 지침은 Visual Explain에 대한 온라인 도움말을 참조하십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 갱신하십시오.
- 일반적으로, 인덱스 스캔은 규정된 행이 적을 때 가장 효과가 있습니다. 규정된 행의 수를 예측하기 위해 옵티마이저는 술어가 참조하는 컬럼에 관한 통계치를 사용합니다. 어떤 값이 다른 값보다 자주 나타난다면, **runstats** 명령의 WITH DISTRIBUTION절을 사용하여 분산 통계를 구하는 것이 중요합니다. 비정상 분산 통계를 이용하여, 옵티마이저는 자주 나타나는 값과 자주 나타나지 않는 값들을 구분할 수 있습니다.

MSJOIN

연산자 이름: MSJOIN

의미: 외부 및 내부 테이블 모두의 규정된 행이 조인-술어 순으로 있어야 하는 병합 조인. 병합 조인을 *병합 스캔 조인* 또는 *정렬 병합 조인*이라고도 합니다.

FROM절에서 두 개 이상의 테이블이 참조될 경우에는 조인이 반드시 필요합니다. 두 개의 다른 테이블의 컬럼에 해당하는 조인 술어가 있을 경우, 병합 조인이 가능합니다. 이것은 또한 재작성된 서브쿼리로부터 발생할 수도 있습니다

병합 조인을 할 때에는 테이블들이 일반적으로 한번만 스캔되므로 컬럼을 조인할 때 입력이 일정한 순서대로 이루어져야 합니다. 순서화된 입력은 인덱스나 정렬된 테이블에 액세스함으로써 이루어집니다.

자세한 내용은 *관리 안내서*의 조인 개념에 관한 절을 참조하십시오.

성능 제안

- 조인할 행의 수를 줄이려면, 로컬 술어(즉, 한 테이블을 참조하는 술어)를 사용하십시오.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

NLJOIN

연산자 이름: NLJOIN

의미: 외부 테이블의 각 행마다 한 번씩 내부 테이블을 스캔하는(보통 인덱스 스캔으로) 중첩 루프 조인.

FROM절에서 두 개 이상의 테이블이 참조될 경우에는 조인이 반드시 필요합니다. 중첩 루프 조인에는 조인 술어가 필요없지만 있으면 더 잘 수행됩니다.

중첩 루프 조인은 다음 중 하나로 수행됩니다.

- 외부 테이블에서 사용된 행마다 내부 테이블을 스캔.
- 외부 테이블에서 사용된 행마다 내부 테이블의 인덱스를 검색.

자세한 내용은 *관리 안내서*의 조인 개념에 관한 절을 참조하십시오.

성능 제안

- 중첩 루프 조인은 내부 테이블(NLJOIN 연산자 오른쪽에 있는 테이블)의 조인-술어 컬럼에 인덱스가 있으면 보다 효율적으로 수행됩니다. 내부 테이블이 IXSCAN이 아니라 TBSCAN인지 확인하십시오. 그렇다면, 조인 컬럼에 인덱스를 추가해 보십시오.

  조인을 보다 효율적으로 하는 또 다른 방법은 외부 테이블이 정렬되도록 조인 컬럼에 대한 인덱스를 작성하는 것입니다.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.
- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

관련 정보:

- 스타 조인.

PIPE

연산자 이름: PIPE

의미: 행을 변경하지 않고 다른 연산자로 행 전송.

(이 연산자는 디버그 모드용입니다.)

RETURN

연산자 이름: RETURN

의미: 쿼리에서 사용자에게 데이터를 리턴. 이것은 액세스 플랜 그래프의 마지막 연산자이며 액세스 플랜의 최종 합계와 비용을 보여줍니다.

이 연산자는 필수 연산입니다.

성능 제안

- 필요한 데이터만 검색하는 술어를 사용했는지 확인하십시오. 예를 들어, 원하는 테이블의 일부를 나타내는 술어에 대한 선택성 값이 리턴되었는지 확인하십시오.

RIDSCN

연산자 이름: RIDSCN

의미: 하나 이상의 인덱스에서 확보한 행 ID(RID) 목록 스캔.

이 연산자는 옵티마이저에서 다음의 경우에 고려됩니다.

- 술어가 OR 키워드로 연결되거나, IN 술어가 있습니다. 인덱스 OR 처리(ORing)라는 기술을 사용하여, 같은 테이블의 여러 인덱스에 액세스한 결과를 조합할 수 있습니다.
- 단일 인덱스에 액세스하는 경우, 목록 프리페치를 사용하는 것이 좋습니다. 기본 행을 사용하기 전에 행 ID를 정렬하는 것이 I/O를 더욱 효율적으로 만들기 때문입니다.

RQUERY

연산자 이름: SHIP

의미: 리모트 데이터 소스에서 데이터를 검색하기 위해 페더레이티드 시스템에서 사용되는 연산자. 이 연산자는 SHIP 연산자가 쿼리 결과를 검색하기 위해 SQL SELECT문을 리모트 데이터 소스로 보낼 때 옵티마이저에서 고려합니다. SELECT문은 데이터 소스가 제공하는 SQL문을 사용하여 생성되며, 데이터 소스에서 허용하는 유효 쿼리를 포함할 수 있습니다.

성능 제안 관리 안내서 볼륨 2, 페더레이티드 데이터베이스 쿼리 및 네트워크 조정 정보의 제4장을 참조하십시오.

SORT

연산자 이름: SORT

의미: 테이블의 행을 하나 이상의 컬럼 순서로 정렬하는 것을 의미하며, 선택적으로 중복된 항목을 삭제할 수도 있습니다.

요청된 순서로 정렬된 인덱스가 없거나 인덱스 스캔보다 정렬 비용이 저렴한 경우, 정렬이 필요합니다. 일반적으로 정렬은 필요한 행이 페치되면 최종 작업으로 수행되거나, 조인이나 그룹으로 나누기 전에 데이터를 정렬하기 위해 사용됩니다.

행이 많거나 정렬된 데이터를 파이프할 수 없는 경우, 임시 테이블을 만들 때 비용이 많이 필요합니다.

정렬에 관한 자세한 내용은 *관리 안내서*를 참조하십시오.

성능 제안

- 정렬 컬럼에 대해 인덱스를 추가하는 것을 고려해 보십시오.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.
- 필요한 데이터만 검색하는 술어를 사용했는지 확인하십시오. 예를 들어, 원하는 테이블의 일부를 나타내는 술어에 대한 선택성 값이 리턴되었는지 확인하십시오.
- 시스템 임시 테이블 스페이스의 프리페치 *크기*가 적절한지, 즉 입출력이 바운드되지 않았는지 점검하십시오(이를 점검하려면, **명령문 -> 통계 표시 -> 테이블 스페이스**를 선택하십시오).
- 대용량의 정렬이 자주 필요하면, 다음 구성 매개변수의 값을 늘리는 것을 고려해 보십시오.
  - 정렬 힙 *크기*(sortheap). 이 매개변수를 변경하려면, 제어 센터에서 데이터베이스를 오른쪽 마우스 단추로 선택한 후, 그 팝업 메뉴에서 *구성*을 선택하십시오. 노트북이 나타나면 **성능** 탭을 선택하십시오.

- 정렬 힙 임계값(sheapthres). 이 매개변수를 변경하려면, 제어 센터 창에서 데이터베이스 인스턴스를 오른쪽 마우스 단추로 선택하고, 그 팝업 메뉴에서 *구성*을 선택하십시오. 노트북이 나타나면 **성능** 탭을 선택하십시오.

- 통계치가 현재 값이 아닌 경우, runstats 명령을 사용하여 통계치를 갱신하십시오.

TBSCAN

연산자 이름: TBSCAN

의미: 데이터베이스에서 직접 필요한 모든 데이터를 읽어 행을 검색하는 테이블 스캔(관련 스캔).

이 유형의 스캔은 다음과 같은 경우처럼 인덱스 스캔시에 옵티마이저가 선택합니다.

- 스캔된 범위의 값들이 자주 나타날 때(즉, 테이블의 대부분에 액세스해야 할 경우)
- 테이블이 작을 때
- 인덱스의 클러스터링이 낮을 때
- 인덱스가 없을 때

테이블 및 인덱스 스캔에 관한 자세한 내용은 *관리 안내서*를 참조하십시오

성능 제안

- 테이블이 크고 대부분의 테이블 행이 액세스되지 않는 경우, 인덱스 스캔이 테이블 스캔보다 효율적입니다. 이런 경우 옵티마이저가 인덱스 스캔을 사용할 가능성을 높이려면, 선택적 술어가 있는 컬럼에 인덱스를 추가해 보십시오.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.

- 인덱스가 있으나 사용되지 않는 경우에는 각 선두 컬럼에 선택적 술어가 있는지 점검하십시오. 이 술어가 있는 경우, 인덱스에 대한 클러스터링 등급이 높은지 확인하십시오(이 통계치를 보려면, 정렬 아래의 테이블에 대한 테이블 통계 창을 열고 인덱스 통계 창을 보기 위해 *인덱스* 누름 단추를 누르십시오).

- 테이블 스페이스의 프리페치 크기가 적절한지, 즉 I/O에 치중되어 있는지 확인하십시오(이를 점검하려면, **명령문 -> 통계 표시 -> 테이블 스페이스**를 선택하십시오).

  자세한 내용은 *관리 안내서*의 버퍼 풀로 데이터 프리페칭에 관한 절을 참조하십시오 .

- 통계치가 현재의 것이 아닌 경우, runstats 명령을 사용하여 갱신하십시오.

  수량 및 빈도에 관한 통계는 술어의 선택성에 관한 정보를 제공합니다. 예를 들면, 이들 통계는 인덱스 스캔을 테이블 스캔 대신 사용할 시기를 결정하는 데 사용할 수 있습니다. 통계치를 갱신하려면, WITH DISTRIBUTION절을 포함하는 테이블에서 **runstats** 명령을 사용하십시오.

TEMP

연산자 이름: TEMP

의미: 다른 연산자가 다시 읽을 수 있도록(여러 번) 임시 테이블에 데이터를 저장하는 조치. 이 테이블은 SQL문이 처리되기 전에 제거되지 않았다면 SQL문이 처리된 후에 제거됩니다.

이 연산자는 중간 결과를 저장하거나 서브쿼리를 판단하는 데 필요합니다. 경우(명령문이 갱신될 수 있는 경우와 같이)에 따라 반드시 필요하기도 합니다.

TQUEUE

연산자 이름: TQUEUE

의미: 쿼리를 처리하는 에이전트가 여러 개 있는 경우 한 데이터베이스 에이전트에서 다른 데이터베이스 에이전트로 테이블 데이터를 전달하는 데 사용되는 테이블 큐. 병렬 처리될 경우, 쿼리 하나를 처리하는 데 여러 개의 데이터베이스 에이전트가 사용됩니다.

테이블 큐 유형은 다음과 같습니다.

- **로컬:** 테이블 큐가 단일 노드 내의 데이터베이스 에이전트 사이에서 데이터를 전달하는 데 사용됩니다. 로컬 테이블 큐는 파티션내 병렬 처리에 사용됩니다.

- **로컬이 아님:** 테이블 큐가 서로 다른 노드의 데이터베이스 에이전트 사이에서 데이터를 전달하는 데 사용됩니다.

UNION

연산자 이름: UNION

의미: 여러 테이블로부터의 행 스트림의 병합.

이 연산자는 필수 연산입니다. 액세스 플랜 비용을 줄이려면, 병합할 일련의 행을 정의하는 다른 연산자(스캔, 조인)에 집중하십시오.

UNIQUE

연산자 이름: UNIQUE

의미: 지정된 컬럼에 중복 값이 있는 행 삭제.

성능 제안

- 이 연산자는 적절한 컬럼에 고유한 인덱스가 있는 경우에만 필수 연산이 아닙니다.

  인덱스에 대한 자세한 내용은 Visual Explain에 대한 온라인 도움말에서 적절한 인덱스 작성을 참조하십시오.

UPDATE

연산자 이름: UPDATE

의미: 테이블 행의 데이터 갱신.

이 연산자는 필수 연산입니다. 액세스 플랜 비용을 줄이려면, 병합할 일련의 행을 정의하는 다른 연산자(스캔, 조인)에 집중하십시오.

부록 C. DB2 개념

데이터베이스

관계형 데이터베이스는 데이터를 테이블의 콜렉션으로 나타냅니다. 테이블은 정의된 컬럼 세트 및 임의의 행 수로 구성됩니다. 각 테이블의 데이터는 논리적으로 관련되므로 테이블간의 관계를 정의할 수 있습니다. 수학적 원리 및 관계 연산(예: INSERT, SELECT 및 UPDATE)에 따라 데이터를 보고 조작할 수 있습니다.

데이터베이스는 데이터는 물론 그 구조에 대한 설명을 포함한다는 점에서 자체 설명됩니다. 데이터베이스에는 데이터의 논리적 및 물리적 구조를 설명하는 시스템 카탈로그 테이블 세트, 데이터베이스와 연관된 매개변수 값을 포함하는 구성 파일 및 아카이브할 수 있는 트랜잭션과 진행 중인 트랜잭션을 기록하는 복구 로그가 포함됩니다.

데이터베이스는 로컬 또는 리모트일 수 있습니다. 로컬 데이터베이스는 실제로 사용 중인 워크스테이션에 있는 반면, 다른 머신의 데이터베이스는 리모트로 간주됩니다.

스키마

스키마는 데이터베이스 오브젝트(예: 테이블, 뷰, 인덱스 및 별명) 세트를 그룹화하는 데 사용되는 고유 ID입니다. 즉, PAYROLL 테이블을 작성 중인 경우 다른 사용자가 이미 같은 이름의 테이블을 작성했는지 알아내기 위해 데이터베이스를 검색하는 작업은 지루한 일입니다. 각 오브젝트의 이름은 해당 스키마 내에서만 고유해야 합니다.

대부분의 데이터베이스 오브젝트에는 두 파트로 된 오브젝트 이름(첫 번째 파트는 스키마 이름이 되고 두 번째 파트는 오브젝트의 이름임)이 있습니다. 오브젝트를 작성할 때 특정 스키마에 오브젝트를 지정할 수 있습니다. 스키마를 지정하지 않

으면, 오브젝트는 대개 오브젝트를 작성한 사용자의 사용자 ID인 디폴트 스키마에 지정됩니다. 예를 들면, Smith라는 사용자는 테이블 SMITH.PAYROLL을 가질 수 있습니다.

스키마는 데이터베이스의 오브젝트가 되기도 합니다. 스키마는 스키마의 첫 번째 오브젝트가 작성될 때 작성됩니다. 스키마는 개인이 소유할 수 있으며 소유자는 스키마내의 데이터와 오브젝트에 대한 액세스를 제어할 수 있습니다.

테이블

관계형 데이터베이스는 데이터를 테이블의 콜렉션으로 나타냅니다. 테이블은 컬럼 및 행에 논리적으로 배열된 데이터(일반적으로 레코드)로 구성됩니다.

각 테이블에는 이름이 있으며, 테이블내의 각 컬럼에는 이름이 있습니다. 특정 순서가 테이블 행 사이에서 유지되지만, 컬럼의 값에 따라 결정된 순서로 행을 검색할 수 있습니다. 테이블의 데이터는 논리적으로 관련됩니다. 모든 데이터베이스 및 테이블 데이터는 테이블 스페이스에 지정됩니다.

부록 D. 주의사항

IBM은 다른 국가에서는 이 자료에 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 언급하는 것이 해당 IBM 제품, 프로그램 또는 서비스만을 사용할 수 있다는 것을 의미하지는 않습니다. IBM의 지적 재산권을 침해하지 않는 한, 기능상 동등한 제품, 프로그램 또는 서비스를 대신 사용할 수 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

2바이트(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, 변경사항은 최신판에 통합됩니다. IBM은 이 책에서 설명한 제품 및(또는) 프로그램을 사전 통지없이 언제든지 개선 및(또는) 변경할 수 있습니다.

이 정보에서 언급되는 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 사이트 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 및 (ii) 교환된 정보의 상호 이용을 목적으로 본 프로그램에 관한 정보를 얻고자 하는 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 들어, 사용권 지불 등)에 따라 사용할 수 있습니다.

이 정보에 기술된 사용권 프로그램 및 사용 가능한 모든 사용권 자료는 IBM이 IBM 기본 계약, IBM 프로그램 사용권 계약(IPLA) 또는 이와 동등한 계약에 따라 제공한 것입니다.

본 문서에 포함된 모든 성능 데이터는 제한된 환경에서 산출된 것입니다. 따라서, 다른 운영 환경에서 얻어진 결과는 상당히 다를 수 있습니다. 일부 성능은 개발 레벨 상태의 시스템에서 측정되었을 수 있으므로, 이러한 측정치가 일반적으로 사용되고 있는 시스템에서도 동일하게 나타날 것이라고는 보증할 수 없습니다. 또한, 일부 성능은 추정을 통해 추측되었을 수도 있으므로, 실제 결과는 다를 수 있습니다. 이 책의 사용자는 해당 데이터를 사용자의 특정 환경에서 검증해야 합니다.

비IBM 제품에 관한 정보는 해당 제품의 공급업체, 공개 자료 또는 다른 기타 범용 소스로부터 얻은 것입니다. IBM에서는 이러한 제품을 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 언급은 특별한 통지없이 변경될 수 있습니다.

이 정보에는 일상의 비즈니스 운영에서 사용되는 자료 및 보고서에 대한 예제가 들어 있습니다. 이 예제에는 가능한 완벽하게 설명하기 위하여 개인, 회사, 상표 및 제품의 이름이 포함될 수 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 들어 있습니다. 샘플 응용프로그램의 작성 기준이 된 운영 플랫폼의 응용프로그램 프로그래밍 인터페이스에 부합하는 응용프로그램의 개발, 사용, 마케팅 또는 배포를 목적으로 이들 샘플 프로그램을 복사, 수정 및 배포할 수 있으며 IBM에 대한 지불 의무는 없습니다. 이러한 예제가 모든 조건하에서 철처히 테스트된 것은 아닙니다. 따라서, IBM은 이들 프로그램의 신뢰성, 서비스 가능성 또는 기능에 대해 어떠한 보증도 하지 않습니다.

이러한 샘플 프로그램 또는 파생 제품의 각 사본이나 일부에는 다음과 같은 저작권 표시가 반드시 포함되어야 합니다.

© (*귀하의 회사명*) (연도). 이 코드의 일부는 IBM Corp. 샘플 프로그램에서 파생됩니다. © Copyright IBM Corp. _연도 *입력*_. All rights reserved.

상표

다음 용어는 미국 또는 기타 국가에서 사용되는 IBM Corporation의 상표이며, 이러한 용어는 DB2 UDB 문서 라이브러리에 있는 최소 하나의 문서에서 사용되었습니다.

ACF/VTAM
AISPO
AIX
AIXwindows
AnyNet
APPN
AS/400
BookManager
C Set++
C/370
CICS
Database 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eServer
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
iSeries
LAN Distance
MVS
MVS/ESA
MVS/XA
Net.Data
NetView
OS/390
OS/400
PowerPC
pSeries
QBIC
QMF
RACF
RISC System/6000
RS/6000
S/370
SP
SQL/400
SQL/DS
System/370
System/390
SystemView
Tivoli
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WebSphere
WIN-OS/2
z/OS
zSeries

다음 용어는 기타 회사의 상표 또는 등록상표이며, DB2 UDB 문서 라이브러리의 최소 하나의 문서에서 사용되었습니다.

Microsoft, Windows, Windows NT 및 Windows 로고는 미국 또는 기타 국가에서 사용되는 Microsoft Corporation의 상표입니다.

Intel 및 Pentium은 미국 및 기타 국가에서 사용되는 Intel Corporation의 상표입니다.

Java 및 모든 Java 관련 상표는 미국 및 기타 국가에서 사용되는 Sun Microsystems, Inc.의 상표입니다.

UNIX는 미국 및 기타 국가에서 Open Group의 등록 상표입니다.

기타 회사, 제품 또는 서비스 이름은 해당 회사의 상표 또는 서비스표입니다.

색인

[하]

C

D

E

F

G

H

I

L

M

N

P

R

S

T

U

V

IBM에 문의

미국에서는 다음 번호로 IBM에 문의하십시오.

- 고객 서비스를 받으려면 1-800-237-5511
- 사용 가능한 서비스 옵션을 알려면 1-888-426-4343
- DB2 마케팅 및 판매에 대해서는 1-800-IBM-4YOU(426-4968)

캐나다에서는 다음 번호로 IBM에 문의하십시오.

- 고객 서비스를 받으려면 1-800-IBM-SERV(1-800-426-7378)
- 사용 가능한 서비스 옵션을 알려면 1-800-465-9600
- DB2 마케팅 및 판매에 대해서는 1-800-IBM-4YOU(1-800-426-4968)

해당 국가 및 지역의 IBM 지사를 찾으려면 IBM의 Directory of Worldwide Contacts(www.ibm.com/planetwide)를 확인하십시오.

제품 정보

DB2 Universal Database 제품 관련 정보는 전화 또는 www.ibm.com/software/data/db2/udb의 WWW(World Wide Web)에서 사용 가능합니다.

이 사이트에서는 기술 라이브러리, 책 주문, 클라이언트 다운로드, 뉴스 그룹, FixPak, 뉴스, 웹 자원에 대한 링크와 관련된 최신 정보를 제공합니다.

미국에 거주하는 분은 다음 번호 중 하나를 선택하여 문의하십시오.

- 제품을 주문하거나 일반 정보를 얻으려면 1-800-IBM-CALL(1-800-426-2255).
- 책에 대한 주문은 1-800-879-2755.

미국 이외의 지역에서는 IBM에 문의하는 방법에 대한 정보는 www.ibm.com/planetwide의 IBM Worldwide 페이지를 참조하십시오.

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™
Visual Explain 자습서
버전 8

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.