UNIT 09

데이터 이동

DB2는 데이터베이스 테이블에 저장된 데이터를 여러 가지 유형의 파일로 저장시키는 EXPORT 유틸리티를 제공합니다. 또한, 파일의 데이터를 테이블에 추가시키는 IMPORT와 LOAD 유틸리티를 제공합니다. LOAD 유틸리티는 대량의 데이터를 고속으로 처리하는데 사용됩니다.

DB2 9.7 운영자 가이드

Administrator Edition

- 데이터 파일의 유형
- EXPORT 유틸리티
- EXPORT 명령어
- IMPORT 유틸리티
- IMPORT 명령어
- LOAD 유틸리티
- LOAD 명령어
- LOAD QUERY 명령어
- LOAD 단계
- BUILD 단계
- DELETE 단계
- 백업 보류 상태
- 점검 보류 상태
- LOAD 시나리오
- Cursor Load

TOPIC 09 01 데이터 파일의 유형

Point

데이블로부터 데이터를 파일로 저장하거나, 파일의 데이터를 추가 입력하는 유틸리티에서 사용되는 파일의 유형은 ASC, DEL, IXF 등이 있습니다. IXF는 데이터와 컬럼에 대한 메타정보를 가지고 있으므로 목표 테이블을 생성할 수도 있습니다.

Tip

Host DB2로부터 데이터 추출 시에는 IXF포맷을 지원합니다.

1 EXPORT, IMPORT, LOAD 명령어는 테이블과 파일간의 데이터 이동을 지원합니다.

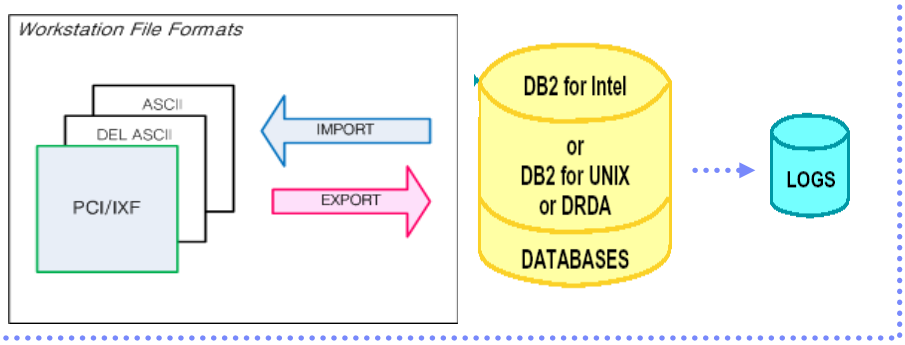


Figure 0901A ••• 데이터 이동에 사용되는 파일의 유형

2 5가지 유형의 데이터 파일과 CURSOR를 지원합니다.

유 형	설 명
ASC	Non-delimited ASCii 파일로, IMPORT, LOAD 에 사용됩니다.
DEL	DELimited ascii 파일로, EXPORT, IMPORT, LOAD 에 사용됩니다.
WSF	DB2 9.7에서는 더 이상 지원하지 않습니다.
IXF	Integrated eXchange Format으로 EXPORT, IMPORT, LOAD 에 사용됩니다.
CURSOR	SELECT문의 결과 집합을 저장한 구조체로 LOAD에 사용됩니다.

Tip

데이터 파일의 유형과 확장자는 일치하지 않아도 됩니다.

3 ASC파일 유형은 에디터로 편집이 가능한 파일이며, 각 컬럼에 대응되는 데이터의 값들은 그 시작 바이트와 종료 바이트의 위치가 동일합니다.

```
$ cat <데이터 파일명>.asc
AAAA      312       2006-01-01
BB        4538      2006-02-01
```

4 DEL 파일 유형도 에디터로 편집이 가능한 파일이며, 각 컬럼에 대응되는 데이터의 값들은 , (컴마 부호)와 " (쌍따옴표 부호) 등의 구분자에 의해 구별됩니다.

```
$ cat <데이터 파일명>.del
"AAAA", 312,"2006-01-01"
"BB", 45,"2006-02-01"
```

Tip

CURSOR는 유틸리티가 실행되는 동안 엔진에 의해 임시로 메모리 또는 시스템 임시 테이블스페이스에 생성되는 파일과 유사한 개념입니다.

5 IXF 파일 유형은 데이터와 그 데이터에 대한 속성을 함께 가진 파일입니다. 에디터로 편집은 할 수 없으며, 새로운 테이블을 생성하고 데이터를 입력할 때 사용됩니다.

6 CURSOR 유형은 소스 테이블에 SQL 쿼리문을 이용하여 조건에 맞는 결과 집합을 추출하여 데이터 파일을 생성하지 않은 채로 LOAD 유틸리티의 입력으로 사용하는 방법입니다.

TOPIC 09 02 EXPORT 유틸리티

Point

EXPORT 명령어를 이용하여 테이블의 데이터를 파일로 저장할 수 있습니다. 지원되는 파일의 유형은 DEL, WSF, IXF 입니다. 원격 데이터베이스에 대해서도 실행할 수 있으며, 데이터 파일은 클라이언트에 생성됩니다.

1 EXPORT 명령어에서 데이터를 추출할 SELECT문과 출력 파일명을 지정하면, SELECT문의 실행으로 생성된 결과 집합을 지정한 출력 파일로 저장합니다.

```
EXPORT TO kes.del OF DEL
MESSAGES kes.msgs
SELECT * FROM kes.empl ;

EXPORT TO kes.wsf OF WSF
MESSAGES kes.msgs
SELECT * FROM artists;

EXPORT TO kes.ixf OF IXF
MESSAGES kes.msgs
SELECT * FROM artists;
```

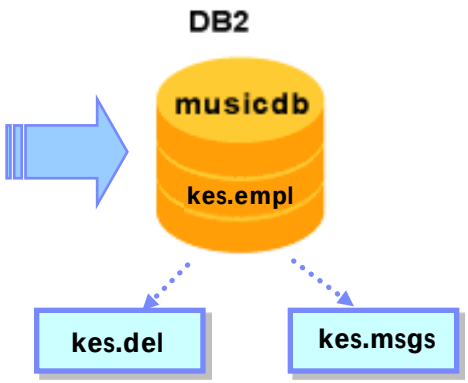


Figure 0902A DEL 유형의 파일로 데이터 내보내기

Tip

DB2 9.7이후 WSF형식은 더 이상 지원되지 않습니다.

2 데이터베이스에 접속하고 export 명령어로 DEL 유형의 파일에 데이터를 저장합니다.

```
$ db2 connect to <데이터베이스명>
$ db2 "export to <출력파일명> of del messages <메시지파일명> <select문>"
```

3 생성된 DEL 유형의 데이터 파일을 확인합니다.

```
$ cat <출력파일명>
```

Tip

메시지 파일에 기록된 오류 코드 중에서 SQL0012W, SQL0347W, SQL0360W, SQL0437W, SQL1824W 은 경고 메시지입니다. 다른 오류가 발생하면, 유틸리티는 실패합니다.

4 메시지 파일을 확인하여 오류가 있었는지 확인합니다.

```
$ cat <메시지파일명>
```

5 WSF 유형의 파일은 Lotus 1-2-3과 Symphony 제품이 사용하는 파일의 형식입니다.

```
$ db2 "export to <출력파일명> of ixf <SELECT문>"
```

Tip

255 글자 이상의 VARCHAR 또는 LOB 등의 LONG 유형의 컬럼이 포함된 경우에는 IXF 유형을 사용하도록 합니다.

6 IXF 유형의 파일로 데이터를 저장하면, 데이터와 컬럼의 속성 정보가 함께 저장되므로, 동일한 구조의 테이블을 생성하고 데이터도 함께 입력할 때 이용됩니다.

```
$ db2 "export to <출력파일명> of ixf <SELECT문>"
```

7 EXPORT 유틸리티는 ASC 유형의 출력 파일을 지원하지 않으므로 , CLP를 이용하여 원하는 SELECT문을 실행하고, 그 결과를 출력 파일로 저장하는 간접적인 방법을 사용합니다.

```
$ db2 -x -o "<SELECT문>" > <출력파일명>
```

TOPIC 09 03 EXPORT 명령어

Point

SELECT문을 이용하여 데이터를 조회하여 결과를 파일로 저장하는 명령어입니다. 지원되는 파일의 유형은 DEL, WSF, IXF 이며, 데이터 파일은 클라이언트에 생성됩니다.

Tip

SYSADM, DBADM 권한 또는 테이블에 대한 CONTROL, SELECT 특권을 가진 사용자가 실행합니다.

1 EXPORT 명령어의 형식은 다음과 같습니다.

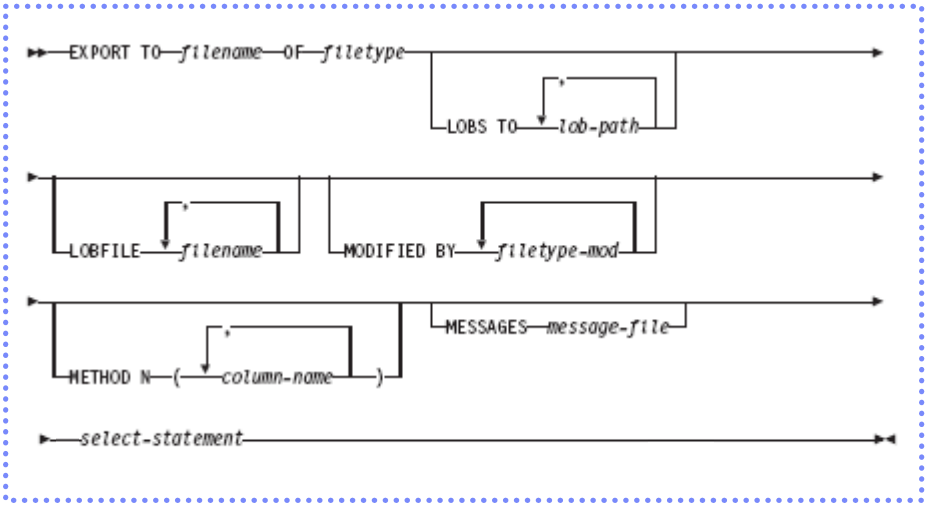


Figure 0903A • • • EXPORT 명령어

Tip

출력 파일의 유형은 OF 옵션으로 판별합니다. 입력 파일의 확장자는 무의미합니다.

Tip

METHOD N 옵션은 WSF, IXF 유형에서만 지원됩니다.

2 옵션에 대한 설명은 다음과 같습니다.

모 드	설 명
TO <파일명>	출력 데이터 파일명을 지정합니다.
OF <파일의 유형>	ASC, DEL, WSF, IXF 중의 한 가지를 지정합니다.
LOBS TO <디렉토리명>	LOB 컬럼이 저장될 디렉토리명을 지정합니다.
LOBFILE <파일명>	LOB 컬럼이 저장될 파일명을 지정합니다. 3자리 숫자의 확장자가 자동적으로 생성됩니다.
MODIFIED BY <파일형식수정자>	다양한 옵션으로 데이터 파일의 형식을 제어합니다.
METHOD N	컬럼의 이름으로 데이터를 지정합니다.
MESSAGES	작업 결과를 기록하는 메시지 파일명을 지정합니다.
<SELECT문>	데이터를 추출할 SELECT문을 지정합니다.

Tip

DB2 9.7이후 WSF형식은 더 이상 지원되지 않습니다.

3 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.admin.cmd.doc/doc/r0008303.html 에서 MODIFIED BY 옵션에 사용되는 <파일형식수정자> 정보를 확인합니다.

TOPIC 09 04 IMPORT 유틸리티

Point

IMPORT 명령어를 이용하여 입력 파일의 데이터를 테이블에 추가할 수 있습니다. 지원되는 파일의 유형은 ASC, DEL, WSF, IXF 입니다. 원격 데이터베이스에 대해서도 실행할 수 있으며, 데이터 파일은 클라이언트에 있어야 됩니다.

Tip

INSERT, UPDATE, CREATE TABLE 등을 실행할 때와 동일하게 데이터베이스 로깅, 트리거 실행, 고유 인덱스 점검, 외부 키 점검, 컬럼 제약 조건 점검 등이 실행됩니다.

1 데이터 파일을 준비하여 IMPORT 명령어를 실행하면 파일의 데이터를 INSERT 문으로 테이블에 추가하거나 UPDATE 문으로 갱신합니다. IXF 유형을 사용하면 CREATE TABLE 문으로 테이블을 생성하고, INSERT 문으로 데이터를 추가합니다.

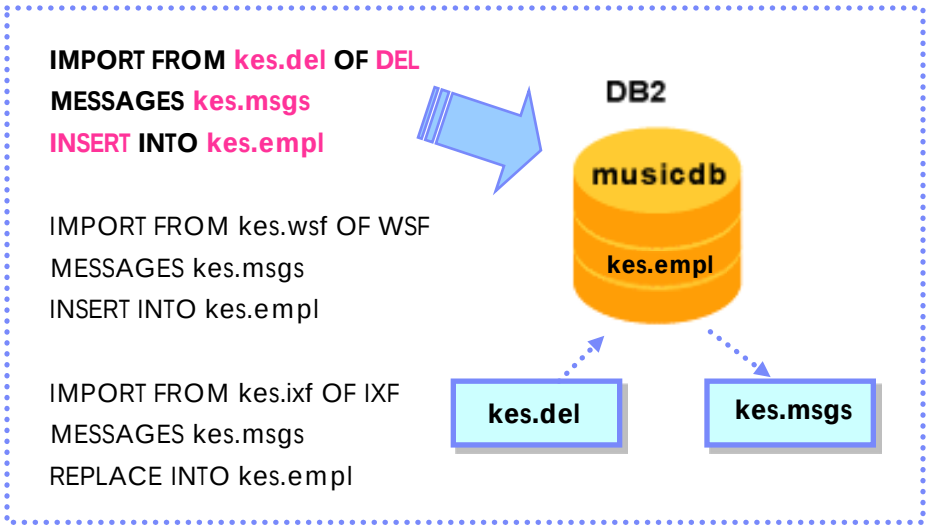


Figure 0904A • • • DEL 유형의 파일에서 데이터 가져오기

Tip

DB2 9.7이후 WSF형식은 더 이상 지원되지 않습니다.

2 ASC 유형의 입력 파일에서 데이터를 테이블로 저장합니다. 반드시 METHOD L 옵션을 이용하여 컬럼에 대응하는 필드의 위치를 지정해야 합니다.

```
$ db2 "import from <입력파일명> of asc METHOD L (<시작위치 1> <종료위치 1>, <시작위치 2> <종료위치 2>, ... , <시작위치 N> <종료위치 N>) messages <메시지파일명> <실행모드> into <목표테이블명>"
```

Tip

ASC, DEL 유형의 데이터 파일은 에디터를 이용하여 편집할 수 있습니다.

3 DEL 유형의 입력 파일에서 데이터를 테이블로 저장합니다.

```
$ db2 "import from <입력파일명> of del messages <메시지파일명> <실행모드> into <목표테이블명>"
```

Tip

Linux 또는 Unix에서 REPLACE 모드로 입력 파일의 이름을 /dev/null 로 지정하면 목표 테이블의 기존 데이터가 로깅 없이 삭제됩니다.

4 WSF 유형의 파일은 Lotus 1-2-3과 Symphony 제품이 사용하는 파일의 형식입니다. 목표 테이블에 데이터를 추가할 수 있습니다.

```
$ db2 "import from <입력파일명> of wsf messages <메시지파일명> <실행모드> into <목표테이블명>"
```

Tip

IMPORT 유틸리티가 실패하면, 마지막 COMMITCOUNT 지점으로 롤백됩니다. REPLACE 모드에서 COMMITCOUNT 옵션이 없다면, 모든 데이터는 삭제됩니다.

5 IXF 유형의 파일을 이용하면 목표 테이블을 생성하고 데이터를 추가할 수 있습니다. 목표 테이블이 이미 존재하면 데이터만 추가할 수도 있습니다.

```
$ db2 "import from <입력파일명> of ixf messages <메시지파일명> <실행모드> into <목표테이블명>"
```

TOPIC 09 05 IMPORT 명령어

Point

입력 파일의 데이터를 INSERT, UPDATE, CREATE TABLE 문을 이용하여 기존 또는 새로운 테이블로 저장하는 명령어입니다. ASC, DEL, WSF, IXF 유형이 지원되며, 데이터 파일은 클라이언트에 있어야 합니다.

Tip
SYSADM, DBADM 권한 또는 테이블에 대한 CONTROL, INSERT 특권을 가진 사용자가 실행합니다.

1 IMPORT 명령어의 형식은 다음과 같습니다.

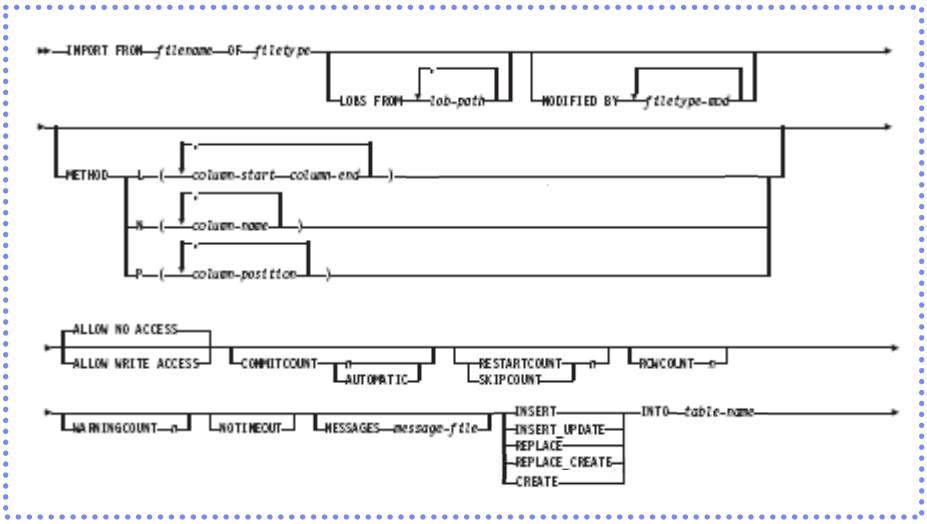


Figure 0905A IMPORT 명령어

Tip
입력 파일의 유형은 OF 옵션으로 판별합니다. 입력 파일의 확장자는 무의미합니다.

Tip
COMMITCOUNT 옵션으로 데이터가 큰 경우에 과다한 로그 파일 사용을 방지하는 것이 좋습니다.

Tip
RESTARTCOUNT n 옵션과 SKIPCOUNT n 옵션은 동일합니다.

Tip
RESTARTCOUNT m 옵션과 ROWCOUNT n 옵션을 함께 사용하면, m+1 번째부터 n 건의 데이터를 처리합니다.

Tip
INSERT_UPDATE 모드를 사용하려면, 반드시 목표 테이블에 기본 키가 정의되어 있어야 합니다.

Tip
CRETAE, CREATE_REPLACE 옵션은 IXF 유형에서 지원됩니다.

2 옵션에 대한 설명은 다음과 같습니다.

모 드	설 명
FROM <입력파일명>	입력 데이터 파일명을 지정합니다.
OF <파일의 유형>	ASC, DEL, WSF, IXF 중의 한 가지를 지정합니다.
METHOD L / N / P	필드의 시작과 종료 위치 바이트 옵셋, 컬럼의 이름, 필드 번호로 컬럼별 데이터를 지정합니다.
COMMITCOUNT	n 건을 처리할 때마다 COMMIT을 실행합니다.
RESTARTCOUNT n	n+1 번째 데이터부터 처리합니다.
ROWCOUNT n	n 건의 데이터만 처리합니다.
WARNINGCOUNT n	n 개 이상의 경고가 발생하면 처리를 중단합니다.
MESSAGES	작업 결과를 기록하는 메시지 파일명을 지정합니다.
INSERT	기존의 목표 테이블에 데이터를 추가합니다.
INSERT_UPDATE	기존의 목표 테이블에 데이터를 추가 또는 갱신합니다.
REPLACE	기존의 목표 테이블의 데이터를 로깅 없이 truncate한 후, INSERT문으로 데이터를 추가합니다.
CREATE	목표 테이블을 생성하고 데이터를 입력합니다.
CREATE_REPLACE	REPLACE 또는 CREATE 옵션과 동일합니다.
INTO <테이블명>	목표테이블명을 지정합니다.

3 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.admin.cmd.doc/doc/r0008304.html 에서 MODIFIED BY 옵션에 사용되는 <파일형식수정자> 정보를 확인합니다.

TOPIC 09 06 LOAD 유틸리티

Point

파일로부터 대량의 데이터를 테이블에 고속으로 저장하는 유틸리티입니다. load 명령어와 set integrity 명령어가 사용됩니다. 지원되는 파일의 유형은 ASC, DEL, IXF 이며, 데이터 파일은 서버 또는 클라이언트에 존재할 수 있습니다.

Tip

입력된 데이터는 데이터베이스 로그 파일에 기록되지 않고, 목표 테이블과 관련된 트리거도 실행되지 않습니다.

1 데이터 파일을 준비하여 LOAD 명령어를 실행하면, 입력된 데이터는 목표 테이블과 인덱스에 반영되고, 고유 인덱스를 위반한 데이터는 목표 테이블에 입력되지 않습니다. LOAD 명령어는 내부적으로 LOAD, BUILD, DELETE 과정을 실행합니다.

단 계	설 명
LOAD	입력 파일의 데이터를 테이블스페이스 컨테이너로 직접 복사합니다. 컬럼과 데이터 유형이 호환되지 않거나, NULL을 허용하지 않는 컬럼에 NULL 값을 제공한 행은 복사되지 않습니다. 인덱스에 대한 정보를 함께 수집합니다.
BUILD	LOAD 단계에서 수집한 인덱스 정보를 이용하여 기존의 인덱스를 갱신하거나 재 생성합니다.
DELETE	고유 인덱스를 위반한 데이터를 점검하여 목표 테이블에서 제거하여 미리 정의된 예외 테이블에 입력합니다.

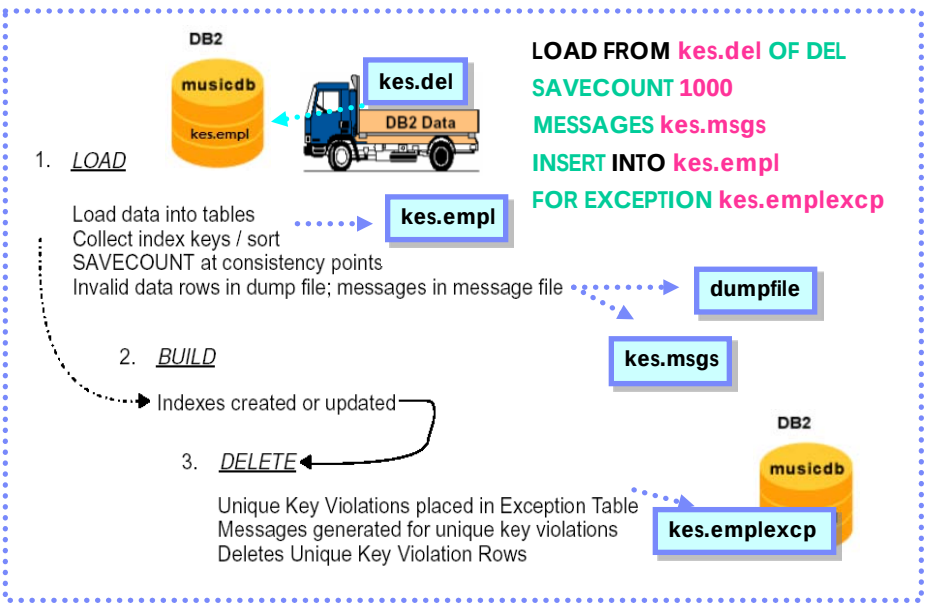


Figure 0906A LOAD 명령어의 3단계

Tip

예외 테이블을 사용하는 것을 권장합니다. 예외 테이블은 목표테이블과 다른 테이블스페이스에 생성하도록 합니다.

2 성공적으로 입력되지 못한 데이터는 예외 테이블에 저장합니다. 예외 테이블은 목표 테이블과 동일한 구조로 생성하며, 예외 데이터로 처리된 이유를 저장하기 위해 2개의 컬럼이 추가로 필요합니다.

```
$ db2 "create table <예외테이블> like <목표테이블>"
$ db2 "alter table <예외테이블> add column ts timestamp add column
    msg clob(32K)"
```

3 아카이브 로깅에서 load 명령어가 완료된 후에 목표 테이블이 속한 테이블스페이스는 '백업 보류 (Backup Pending)' 상태가 될 수 있습니다. backup db 명령어로 해결합니다.

4 load 명령어가 완료된 후에 목표 테이블에 외부키 또는 컬럼 제약 조건이 있으면, 목표 테이블은 '점검 보류 (Check Pending)' 상태가 됩니다. set integrity 명령어로 해결합니다.

TOPIC 09 07 LOAD 명령어

Point

LOAD 유틸리티에서 사용되는 명령어로 데이터 LOAD, 인덱스 BUILD, 고유 인덱스 위반 행 DELETE 등의 작업을 처리합니다. ASC, DEL, IXF 유형의 데이터 파일을 지원하며, 데이터 파일은 서버 또는 클라이언트에 존재할 수 있습니다.

Tip
SYSADM, DBADM 또는 LOAD 권한과 테이블에 대한 INSERT, DELETE 특권이 필요합니다.

Tip
데이터의 일관성을 보장을 위해 기본적으로 목표 테이블에 배타적 잠금을 적용합니다.

Tip
OF CURSOR 옵션을 이용하면, CURSOR 를 선언하여 입력 파일을 대신합니다.

Tip
데이터 파일이 클라이언트에 있으면 CLIENT 옵션을 사용합니다.

Tip
입력 파일의 유형은 OF 옵션으로 판별합니다. 입력 파일의 확장자는 무의미합니다.

Tip
입력 데이터 파일이 큰 경우에는 SAVECOUNT 옵션을 사용하면, 복사 중에 실패해도 마지막 복사 완료 지점부터 계속 복사할 수 있습니다.

Tip
REPLACE 옵션을 사용한 LOAD 는 TERMINATE 옵션으로 종료했을 때, 목표 테이블의 모든 데이터는 truncate 되어 남아있지 않습니다.

Tip
고유 인덱스, 외부키, 컬럼 제약 조건 등을 위반한 행을 예외테이블에 저장하여 확인하는 것이 권장됩니다.

Tip
아카이브 로깅 모드를 사용할 때는 COPY 옵션을 지정하도록 합니다.

1 LOAD 명령어의 형식은 다음과 같습니다.

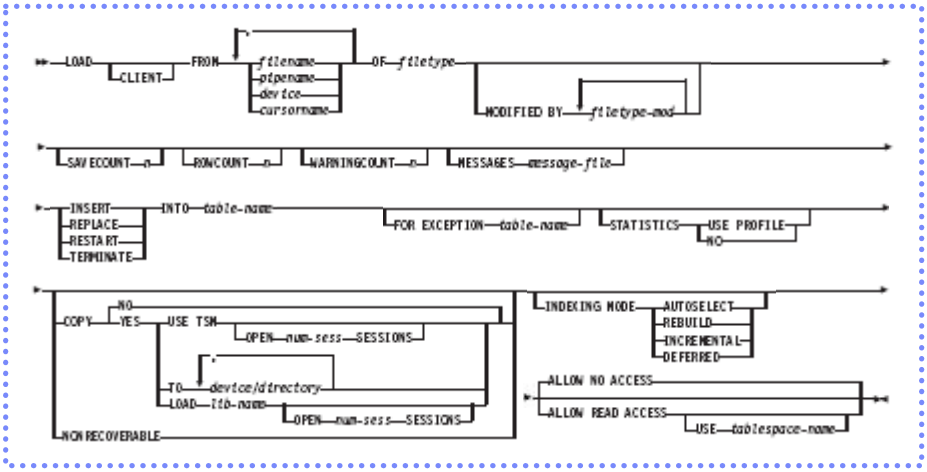


Figure 0907A LOAD 명령어

2 옵션에 대한 설명은 다음과 같습니다.

옵션	설명
CLIENT	클라이언트의 데이터 파일을 사용할 수 있습니다.
FROM <입력파일명>	입력 데이터 파일명을 지정합니다.
OF <파일의 유형>	ASC, DEL, IXF 중의 한 가지를 지정합니다.
METHOD L / N / P	데이터 파일의 필드를 구별하는 방법을 지정합니다.
SAVECOUNT n	n 건을 복사할 때마다 복사 완료 정보를 보관합니다.
ROWCOUNT n	n 건의 데이터만 처리합니다.
WARNINGCOUNT n	n 개 이상의 경고가 발생하면 처리를 중단합니다.
MESSAGES <파일명>	작업 결과를 기록하는 메시지 파일명을 지정합니다.
INSERT	목표 테이블에 데이터를 추가합니다.
REPLACE	목표 테이블의 데이터를 교체합니다.
RESTART / TERMINATE	중단된 LOAD 작업을 다시 시작하거나 종료합니다.
INTO <테이블명>	목표테이블명을 지정합니다.
FOR EXCEPTION <테이블명>	예외테이블명을 지정합니다.
COPY <모드>	백업 실행 여부를 지정합니다.
INDEXING MODE <모드>	인덱스 재생성 모드를 지정합니다.
ALLOW READ ACCESS	load 명령어가 실행되기 이전의 데이터에 대한 읽기 액세스를 허용합니다.

3 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.admin.cmd.doc/doc/r0008305.html 에서 MODIFIED BY 옵션에 사용되는 <파일형식수정자> 정보를 확인합니다.

TOPIC 0908 LOAD QUERY 명령어

Point

로드 유틸리티가 로드 조작 중에 데이터베이스 일관성을 위해 변경하는 테이블의 상태 값을 확인하는 명령어로 LOAD 명령어의 현재 실행 단계 및 LOAD 단계에서 복사 완료된 데이터의 건수와 재생성 또는 갱신이 완료된 인덱스의 개수도 확인할 수 있습니다.

Tip

권한 또는 특권이 필요하지 않습니다.

1 LOAD QUERY 명령어의 형식은 다음과 같습니다.

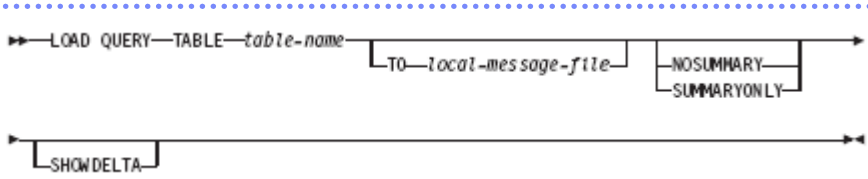


Figure 0908A ··· LOAD QUERY 명령어

2 옵션에 대한 설명은 다음과 같습니다.

옵션	설명
<테이블명>	load 명령어가 실행 중인 목표 테이블명을 지정합니다.
TO <메시지 파일명>	load 명령어가 현재까지 실행한 상황을 저장할 메시지 파일명을 지정합니다.
NOSUMMARY	load 명령어가 처리한 데이터 건수에 대한 요약 정보 (읽은 행 수, 건너뛴 행 수, 입력된 행 수, 거부된 행 수, 삭제된 행 수, 커미트된 행 수, 경고 개수)를 제공하지 않습니다.
SUMMARYONLY	load 명령어가 처리한 데이터 건수에 대한 요약 정보만 제공합니다.
SHOW DELTA	최근의 load query 명령어 실행 이후에 변경된 정보만 제공합니다.

Tip

load 명령어를 실행하고 있는 세션에서 실행하지 말고 새로운 세션에서 실행합니다.

3 데이터베이스에 접속한 후에 load query 명령어를 실행합니다.

```
$ db2 connect to <DB명>
$ db2 load query table <목표테이블명> TO <메시지파일명>
$ db2 load query table <목표테이블명> NOSUMMARY
$ db2 load query table <목표테이블명> SUMMARYONLY
$ db2 load query table <목표테이블명> summaryonly SHOWDELTA
```

```
$ db2 load query table kes.empl summaryonly

Number of rows read         - 453376
Number of rows skipped      - 0
Number of rows loaded       - 453376
Number of rows rejected     - 0
Number of rows deleted      - 0
Number of rows committed    - 408439
Number of warnings          - 0

Tablestate:
  Load in Progress
```

Figure 0908B ··· LOAD QUERY 명령어의 SUMMARYONLY 옵션

TOPIC 09 09 LOAD 단계

Point

LOAD 유틸리티의 첫 번째 과정으로 입력 파일의 데이터를 테이블스페이스 컨테이너로 직접 복사합니다. 컨테이너에 추가된 데이터는 데이터베이스 로그 파일에 기록되지 않으며, 데이터의 추가로 인한 INSERT 트리거도 실행되지 않습니다.

Tip

데이터는 SQL문을 이용해서 입력되는 것이 아니라, 디스크 수준에서 직접 복사됩니다.

Tip

MESSAGES 옵션을 이용하면, 특정 행이 유효하지 못한 데이터로 분류된 이유를 확인할 수 있습니다.

Tip

MODIFIED BY DUMPFILE 옵션을 이용하면 지정한 파일명으로 덤프 파일이 생성되어 유효하지 못한 데이터를 저장합니다. LOAD 작업이 완료된 후, 덤프 파일을 입력 파일로 이용하여 추가적인 LOAD 작업을 할 수 있습니다.

Tip

Not NULL 컬럼에 데이터파일로부터 값이 입력되지 않는 경우에는 Usedefaults 옵션을 사용하면 칼럼 Default 값이 자동 입력됩니다.

Tip

MODIFIED BY DUMPFILE 옵션은 ASC, DEL 유형의 입력 파일에만 사용할 수 있습니다. 덤프 파일명은 확장자를 한 개만 가질 수 있습니다.

1 입력 파일에 대해 다음과 같은 작업들을 실행합니다.

작 업	설 명
데이터 파일 확인	데이터베이스 서버의 지정한 디렉토리에서 입력 파일의 존재를 확인합니다. 로드 유틸리티를 실행하는 클라이언트에 데이터 파일이 있는 경우에는 CLIENT 옵션이 필요합니다.
데이터 복사	입력 파일의 데이터를 한 행씩 읽어들여 테이블스페이스 컨테이너로 직접 복사합니다. 데이터가 많은 경우에는 SAVECOUNT 옵션을 이용하여 내부적으로 복사 완료 중간 지점을 기록하게 하면, LOAD 유틸리티가 이 단계에서 실패한 경우에 마지막 복사 완료 중간 지점 이후부터 복사를 계속할 수 있습니다.
데이터 유형 점검	목표 컬럼의 데이터 유형과 대응되는 필드의 데이터 유형이 호환되지 못하는 행은 복사되지 않습니다. 유효하지 못한 데이터라고 표현합니다.
NULL 값 점검	NULL 값을 허용하지 않는 목표 컬럼에 NULL 값이 제공된 행은 복사되지 않습니다. 유효하지 못한 데이터로 처리하며, DUMPFILE 옵션을 이용하여 별도의 덤프 파일에 저장할 수 있습니다.
인덱스 정보 수집	목표 테이블에 존재하는 모든 인덱스에 대한 인덱스 키값과 대응되는 RID에 대한 정보를 미리 수집합니다.

2 목표 테이블에 데이터를 입력하는 모드는 2가지가 있습니다.

모 드	설 명
INSERT	목표 테이블에 입력 파일의 데이터를 추가합니다.
REPLACE	목표 테이블의 기존 데이터를 데이터베이스 로그 파일에 기록하지 않고 truncate한 후에 입력 파일의 데이터로 대체합니다.

3 load query table 명령어로 복사 작업의 진행 정도를 확인할 수 있습니다. 테이블스페이스의 상태는 '로드 진행 중 (Load in Progress)' 가 됩니다.

4 이 단계에서 실패하면, 목표 테이블은 '로드 보류 (Load Pending)' 상태가 됩니다. 로드 보류 상태는 다음의 2가지 모드를 이용하여 load 명령어를 다시 실행하여 해결합니다.

모 드	설 명
RESTART	LOAD 유틸리티를 계속합니다. SAVECOUNT 옵션을 사용했다면 마지막 복사 완료 중간 지점 이후부터 데이터 복사가 계속됩니다.
TERMINATE	LOAD 유틸리티를 취소합니다. INSERT 모드를 사용했다면 테이블은 load 명령어를 실행하기 이전의 상태로 복구되고, REPLACE 모드를 사용했다면 테이블의 데이터는 모두 삭제됩니다.

5 성공적으로 완료되면, 덤프 파일과 메시지 파일을 확인합니다. 유효하지 못한 데이터로 분류되어 테이블에 추가되지 못한 데이터와 그 원인을 확인할 수 있습니다.

```
$ cat <메시지파일명>
$ cat <덤프파일명>
```

TOPIC 09-10 BUILD 단계

Point

LOAD 유틸리티의 두 번째 과정으로 LOAD 단계에서 수집된 인덱스 정보를 이용하여 기존의 인덱스를 갱신하거나 재 생성합니다. LOAD 명령어에서 INDEXING MODE 옵션으로 제어합니다.

Tip

load 명령어를 완료한 후에 인덱스를 생성하는 것보다 인덱스를 미리 생성하여 load 명령어의 BUILD 단계에서 처리하는 것이 일반적으로 성능이 좋습니다.

Tip

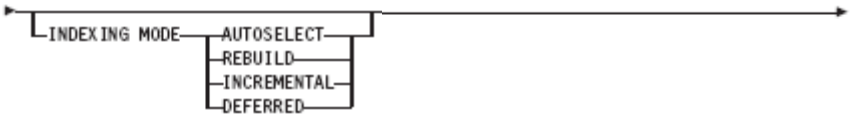
이 단계에서는 Index를 Build하기 위해 Temp공간을 사용합니다. Table의 크기에 비례하여, 그리고 동시에 수행되는 로드 작업 수를 감안하여 충분한 크기의 Temp공간을 미리 확보하십시오.

1 LOAD 단계에서 수집된 인덱스의 정보를 이용하여 새로 추가된 데이터를 인덱스에 반영하는 방법은 2 가지가 있습니다.

작 업	설 명
인덱스 갱신	목표 테이블에 존재하는 모든 인덱스에 새로 추가된 행에 대한 인덱스 데이터를 추가합니다.
인덱스 재생성	목표 테이블에 존재하는 모든 인덱스를 제거하고 다시 생성합니다.

2 인덱스의 재생성 여부는 4가지의 INDEXING MODE 옵션에 의해 결정됩니다.

작 업	설 명
AUTO	엔진이 UPDATE 와 REBUILD 모드 중에서 선택합니다.
UPDATE	목표 테이블에 존재하는 모든 인덱스에 추가된 데이터의 인덱스 정보를 추가합니다.
REBUILD	목표 테이블에 존재하는 모든 인덱스를 재 생성합니다.
DEFERRED	목표 테이블에 존재하는 모든 인덱스의 재생성 작업을 실행하지 않고 보류합니다. INDEXREC 데이터베이스 구성 변수의 값에 의해 재생성 시점이 결정됩니다.




```
load from kes.del of del messages kes.msgs insert into kes.empl
INDEXING MODE REBUILD ;
```

Figure 0910A ••• LOAD 명령어의 INDEXING MODE 옵션

3 load query table 명령어로 인덱스 재생성 또는 갱신 작업의 진행 정도를 확인할 수 있습니다. 테이블스페이스의 상태는 '로드 진행 중 (Load in Progress)' 가 됩니다.

4 이 단계에서 실패하면, 목표 테이블은 '로드 보류 (Load Pending)' 상태가 됩니다. 로드 보류 상태는 다음의 2가지 모드를 이용하여 해결합니다.

모 드	설 명
RESTART	LOAD 유틸리티를 계속합니다. 마지막으로 실패한 인덱스의 재생성 또는 갱신 작업부터 다시 시작합니다.
TERMINATE	LOAD 유틸리티를 취소합니다. INSERT 모드를 사용했다면 테이블은 load 명령어를 실행하기 이전의 상태로 복구되고, REPLACE 모드를 사용했다면 테이블의 데이터는 모두 삭제됩니다.

5 성공적으로 완료되면, 메시지 파일을 확인합니다.

```
$ cat <메시지파일명>
```

TOPIC 09 11 DELETE 단계

Point

LOAD 유틸리티의 세 번째 과정으로 LOAD 단계에서 추가된 데이터 중에서 고유 인덱스의 규정을 위반한 중복 행들을 점검하여 예외 테이블로 입력합니다.

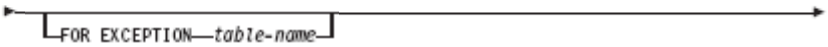
1 LOAD 단계에서 복사된 데이터에 대해서 다음과 같이 점검합니다.

작업	설명
기본키 확인	테이블에 일차키가 정의되어 있으면, 고유 인덱스가 존재합니다.
고유키 확인	테이블에 고유키가 정의되어 있으면, 고유 인덱스가 존재합니다.
고유 인덱스 확인	테이블에 일차키 또는 고유키가 정의되어 있지 않아도, 고유 인덱스가 존재할 수 있습니다.
고유 인덱스 위반 점검	LOAD 단계에서 복사된 데이터 중에서 고유 인덱스의 규정을 위반한 행을 점검합니다. 복사된 순서대로 점검하므로, 중복된 데이터는 최초의 한 건을 제외하고 모두 위반행으로 처리됩니다.
위반 데이터 제거	고유 인덱스 규정을 위반한 데이터를 테이블스페이스 컨테이너에서 제거합니다.
예외 테이블 입력	EXCEPTION 옵션으로 예외 테이블을 제공하면, 위반한 행은 미리 정의된 예외 테이블에 입력됩니다. 예외 테이블을 조회하여 데이터가 입력되지 못한 이유를 확인할 수 있습니다.

Tip

예외 테이블을 지정하지 않아도 이 단계는 실행되지만, 중복행에 대한 정보는 확인할 수 없습니다.

2 FOR EXCEPTION 옵션으로 고유 인덱스를 위반한 행을 저장할 예외 테이블명을 지정합니다.




```
load from kes.del of del messages kes.msgs insert into kes.empl
FOR EXCEPTION kes.emplexcp ;
```

Figure 0911A LOAD 명령어의 FOR EXCEPTION 옵션

3 load query table 명령어로 고유 인덱스 위반 행 점검 작업의 진행 정도를 확인할 수 있습니다. 테이블스페이스의 상태는 '삭제 진행 중 (Delete in Progress)' 가 됩니다.

4 이 단계에서 실패하면, 목표 테이블은 '삭제 보류 (Delete Pending)' 상태가 됩니다. 삭제 보류 상태는 RESTART 또는 TERMINATE 옵션으로 해결합니다.

모드	설명
RESTART	LOAD 유틸리티를 계속합니다. 마지막으로 실패한 DELETE 작업부터 다시 시작합니다.
TERMINATE	LOAD 유틸리티를 취소합니다. INSERT 모드를 사용했다면 테이블은 load 명령어를 실행하기 이전의 상태로 복구되고, REPLACE 모드를 사용했다면 테이블의 데이터는 모두 삭제됩니다.

5 성공적으로 완료되면, 예외 테이블을 이용하여 고유 인덱스를 위반한 행을 확인합니다.

```
$ db2 "select * from <예외테이블명>"
```

TOPIC 09-12 백업 보류 상태

Point

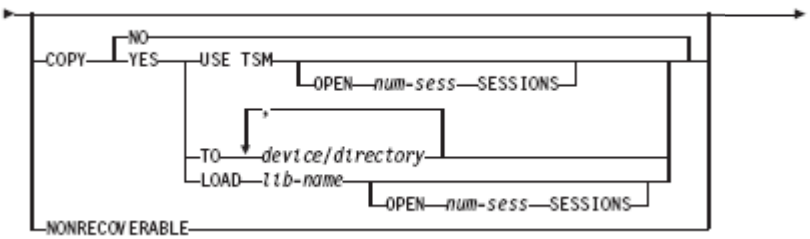
아카이브 로깅 모드에서 LOAD 명령어를 실행하면, 목표 테이블이 속한 테이블스페이스가 '백업 보류' 상태가 됩니다. LOAD 명령어의 COPY 옵션으로 LOAD 중에 백업 이미지를 생성하거나 BACKUP DB 명령어로 테이블스페이스의 백업을 실행합니다.

Tip

순환 로깅 모드를 사용했다면, '백업 보류 상태' 가 발생하지 않습니다.

1 load 명령어의 COPY 옵션으로 제어합니다. 기본 옵션은 COPY NO 입니다.

모 드	설 명
COPY YES	load 명령어를 실행하면서 추가된 데이터에 대한 백업 이미지를 생성합니다.
COPY NO	load 명령어를 실행하면서 추가된 데이터에 대한 백업 이미지를 생성하지 않으므로, load 명령어가 완료되면, 목표 테이블이 속한 테이블스페이스는 '백업 보류' 상태가 됩니다.
NONRECOVERABLE	로드 작업을 '복구 불가능' 상태로 표시합니다. 롤포워드 복구 시에 '복구 불가능'으로 표시된 로드 작업은 무시되고, 해당 시점 이후에 목표 테이블과 관련된 모든 트랜잭션은 무시됩니다. 롤포워드 복구가 완료되면, drop table 문으로 목표 테이블을 제거해야 합니다.




```
load from kes.del of del insert into kes.empl  COPY YES TO /back;
load from kes.del of del insert into kes.empl  COPY NO;
load from kes.del of del insert into kes.empl  ;
load from kes.del of del insert into kes.empl  NONRECOVERABLE;
```

Figure 0912A ... LOAD 명령어의 COPY 옵션

2 COPY YES 옵션을 지정하면, load 명령어를 실행하는 동안 추가된 데이터에 대한 백업 이미지를 생성하며, 완료한 후에 '백업 보류' 상태가 되지 않습니다.

3 COPY 옵션이 기본값인 NO 였다면, load 명령어를 완료한 후에 목표 테이블이 속한 테이블스페이스는 '백업 보류 (Backup Pending)' 상태가 됩니다. backup db 명령어에서 TABLESPACE 옵션을 이용하여 해당 테이블스페이스를 백업하여 해결합니다.

```
$ db2 backup db online tablespace <목표테이블스페이스명>
```

4 NONRECOVERABLE 옵션을 지정하면, load 명령어를 완료한 후에 '백업 보류' 상태가 되지 않습니다. ROLLFORWARD 복구시에 rollforward db 명령어로 로그 파일을 재적용할 때, 목표 테이블에 LOAD 유틸리티로 추가한 데이터는 복구가 불가능합니다.

TOPIC 09-13 점검 보류 상태

Point

LOAD 명령어는 테이블의 외부키와 점검 제한 조건을 점검하지 않으므로, LOAD 명령어가 완료된 후에 목표 테이블은 '점검 보류 (Check Pending)' 상태가 됩니다. 예외 테이블을 이용하여 SET INTEGRITY 문으로 해결합니다.

Tip

외부키 또는 점검에 대한 점검 제한 조건이 없는 목표테이블은 '점검 보류 상태' 가 되지 않습니다.

1 set integrity 문은 테이블의 외부키와 점검 제한 조건을 만족하지 않는 행을 검출하여 테이블에서 삭제하고 예외 테이블에 저장합니다.

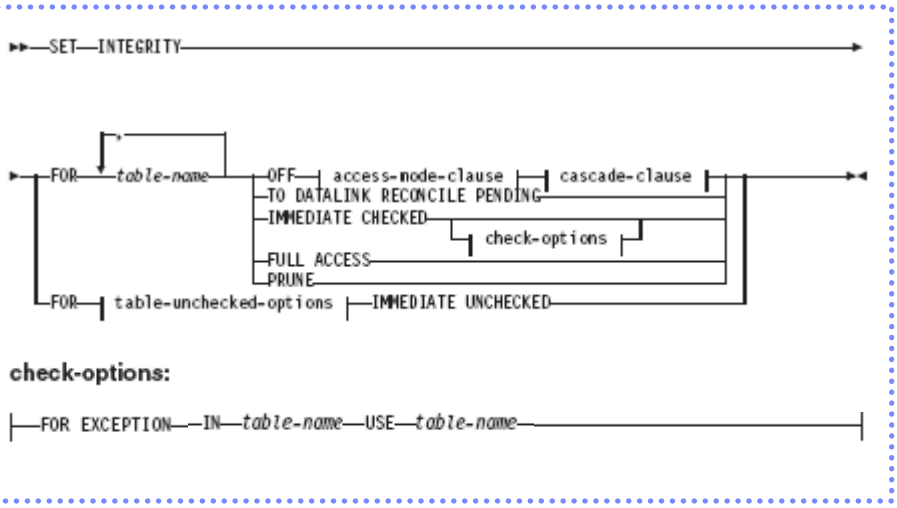


Figure 0913A SET INTEGRITY 문

2 옵션에 대한 설명은 다음과 같습니다.

모 드	설 명
FOR <테이블명>	목표 테이블명을 지정합니다.
IMMEDIATE CHECKED	즉시 점검하도록 요청합니다.
FOR EXCEPTION	예외 테이블이 사용되는 것을 알려줍니다.
FOR <테이블명>	예외 행이 검출된 테이블명입니다.
USE <테이블명>	예외 테이블명입니다.

Tip

예외 테이블은 일반적 용어이지 않고, load 명령어에서 사용하던 것을 그대로 사용하도록 합니다.

3 '점검 보류' 상태에 있는 테이블에 대해 set integrity 문을 실행합니다.

```
db2 "set integrity for <목표테이블명> immediate checked for exception
    in <목표테이블명> use <예외테이블명>"
```

4 예외 테이블을 이용하여 외부키와 점검 제한 조건을 위반한 행을 확인합니다.

```
$ db2 "select * from <예외테이블명>"
```

TOPIC 09 14 LOAD 시나리오

Point

LOAD 명령어로 입력 파일과 예외 테이블을 이용하여 실행합니다. '백업 보류' 상태에서는 BACKUP DB 명령어가 필요하며, '점검 보류' 상태에서는 SET INTEGRITY 명령어가 필요합니다. 실행 후에는 메시지 파일, 덤프 파일, 예외 테이블을 확인합니다.

Tip

load 명령어를 실행할 때는 목표 테이블이 반드시 존재해야 합니다.

Tip

인덱스를 따라 생성하는 것이 좋습니다. 시나리오에서는 alter table 문에 의하여 인덱스가 생성되었습니다.

Tip

kes.dept 테이블과 kes.empl 테이블은 RI 관계를 가지고 있습니다.

1 목표 테이블을 생성하고, 기본 데이터를 입력하는 SQL문을 작성하여, kes.sql 로 저장합니다.

```
$ vi kes.sql
DROP TABLE kes.empl;
DROP TABLE kes.dept;

CREATE TABLE kes.dept (
       id            smallint       not null
     , name          varchar(20)    not null
     , budget        int
) IN  ts01;

ALTER TABLE kes.dept ADD
      CONSTRAINT dept_pk01 PRIMARY KEY (id);

CREATE TABLE kes.empl (
       id            smallint       not null
     , name          varchar(30)    not null
     , sex           char(1)
     , mydept        smallint
     , salary        smallint
     , email         varchar(30)    not null
     , hiredate      date
) in ts02;

ALTER TABLE kes.empl ADD CONSTRAINT empl_pk01
      PRIMARY KEY(id);
ALTER TABLE kes.empl ADD CONSTRAINT empl_uk01
      UNIQUE (email);
ALTER TABLE kes.empl ADD CONSTRAINT empl_fk01
      FOREIGN KEY(mydept) REFERENCES kes.dept ;
ALTER TABLE kes.empl ADD CONSTRAINT empl_cc01
      CHECK (sex = 'M' or sex ='F');

INSERT INTO kes.dept VALUES
       (1,'총무팀',1000), (2,'기술지원팀', 2000) , (3,'POST팀',1500);
INSERT INTO kes.empl VALUES
       (1,'KES','F',1,100,'kes@kr.ibm.com','1993-01-30')
     , (2,'KHY','F',3,250,'khy@kr.ibm.com','1992-03-17')
     , (3,'JHS','F',2,300,'jhs@kr.ibm.com','1997-02-03')
     , (4,'JJY','M',2,280,'jjy@kr.ibm.com','1998-07-22');
```

2 CLP를 이용하여 kes.sql 파일의 SQL문을 실행합니다.

```
$ db2 connect to sample
$ db2 –tvf kes.sql
```

TOPIC 09 14 LOAD 시나리오

Point

LOAD 명령어로 입력 파일과 예외 테이블을 이용하여 실행합니다. '백업 보류' 상태에서는 BACKUP DB 명령어가 필요하며, '점검 보류' 상태에서는 SET INTEGRITY 명령어가 필요합니다. 실행 후에는 메시지 파일, 덤프 파일, 예외 테이블을 확인합니다.

Tip

1 행의 데이터는 문제가 없습니다.

Tip

2 행의 데이터는 첫 번째 칼럼의 값이 empl_pk01에 위반되므로 입력되지 않습니다. load 명령어의 DELETE 단계에서 예외 테이블인 kes.emplexcp 에 저장됩니다.

Tip

3 행의 데이터는 문제가 없습니다.

Tip

4 행에 있는 데이터는 여섯 번째 칼럼의 값이 empl_uk01에 위반되므로 입력되지 않습니다. load 명령어의 DELETE 단계에서 예외 테이블인 kes.emplexcp 에 저장됩니다.

Tip

5 행에 있는 데이터는 세 번째 칼럼의 값이 empl_cc01에 위반되므로 입력되지 않습니다. set integrity 문을 실행하면 예외 테이블인 kes.emplexcp 에 저장됩니다.

Tip

6 행의 데이터는 문제가 없습니다.

Tip

7 행의 데이터는 다섯 번째 칼럼의 값이 문자형이므로 컬럼의 데이터 유형과 호환되지 않습니다. NULL 값을 허용하는 컬럼이므로, NULL 값으로 변경되어 입력됩니다.

Tip

8 행의 데이터는 두 번째 칼럼의 값이 NULL 값이므로 입력되지 않고, 덤프 파일인 kes.dmp에 저장됩니다.

Tip

9 행에 있는 데이터는 네 번째 칼럼의 값이 empl_fk01에 위반되므로 입력되지 않습니다. set integrity 문을 실행하면 예외 테이블에 kes.emplexcp 에 저장됩니다.

Tip

10 행의 데이터는 첫 번째 칼럼의 값이 문자형이고, NULL 값으로 변경될 수 없으므로 입력되지 않고,덤프 파일인 kes.dmp에 저장됩니다.

3 입력 데이터를 준비하여 DEL 유형의 /work/kes.del 파일로 저장합니다.

```
$ vi /work/kes.del
11,"이문세","M", 1, 100, "lms@kr.ibm.com" , "2002-03-07"
11,"김경호","M", 2, 200, "kkh@kr.ibm.com", "2001-04-25"
13,"이기찬","M", 1, 300, "lkc@kr.ibm.com", "2002-02-19"
14,"김현정","F", 3, 400, "lkc@kr.ibm.com", "2002-07-17"
15,"김건모","m" ,2, 500, "kkm@kr.ibm.com", "2001-08-02"
16,"제이","F", 1, 120, "j@kr.ibm.com", "2000-05-08"
17,"양희은","F", 2, "130", "yhe@kr.ibm.com", "2002-10-20"
18,  ,"M", 2, 140, "god@kr.ibm.com", "2001-11-29"
19,"신화","M", 4 , 150, "sh@kr.ibm.com", "2001-04-07"
"20","엄정화","F", 1, 160, "ejw@kr.ibm.com", "2001-04-28"
```

4 추가되지 못한 행들을 확인하기 위해 예외 테이블인 kes.emplexcp를 미리 생성합니다.

```
$ db2 "CREATE TABLE kes.emplexcp LIKE kes.empl"
$ db2 "ALTER TABLE kes.emplexcp ADD COLUMN ts timestamp"
$ db2 "ALTER TABLE kes.emplexcp ADD COLUMN msg clob(32K)"
```

5 목표 테이블의 인덱스를 확인합니다.

```
$ db2 DESCRIBE INDEXES FOR TABLE kes.empl SHOW DETAIL
```

인덱스 스키마	인덱스 이름	규칙	고유한 컬럼수	컬럼 이름
KES	EMPL_PK01	P	1	+ID
KES	EMPL_UK01	U	1	+EMAIL

6 목표 테이블의 현재 데이터를 확인합니다.

```
$ db2 "SELECT id, name, mydept, sex, email FROM kes.empl"
```

ID	NAME	MYDEPT	SEX	EMAIL
1	KES	1	F	kes@kr.ibm.com
2	KHY	3	F	khy@kr.ibm.com
3	JHS	2	F	jhs@kr.ibm.com
4	JJY	2	M	jjy@kr.ibm.com

4 레코드가 선택됨.

```
$ db2 "SELECT id, name, mydept, sex, email FROM kes.emplexcp"
```

ID	NAME	MYDEPT	SEX	EMAIL

0 레코드가 선택됨.

TOPIC 09 14 LOAD 시나리오

Point

LOAD 명령어로 입력 파일과 예외 테이블을 이용하여 실행합니다. '백업 보류' 상태에서는 BACKUP DB 명령어가 필요하며, '점검 보류' 상태에서는 SET INTEGRITY 명령어가 필요합니다. 실행 후에는 메시지 파일, 덤프 파일, 예외 테이블을 확인합니다.

Tip

덤프 파일과 메시지 파일은 load 명령어가 실행되면서 생성됩니다.

7 kes.del 파일에서 kes.empl 테이블로 데이터를 저장할 load 명령어를 작성하여 kes.db2라는 파일에 저장합니다.

```
$ vi kes.db2
LOAD FROM /work/kes.del OF DEL
MODIFIED BY dumpfile=/work/kes.dmp
SAVECOUNT 10000
MESSAGES /work/kes.msgs
INSERT INTO kes.empl
FOR EXCEPTION kes.emplexcp;
```

8 CLP를 이용하여 kes.db2 파일의 load 명령어를 실행합니다.

```
$ db2 –stvf kes.db2
```

Tip

load query 명령어는 load 명령어가 실행되고 있는 세션이 아닌 새로운 세션에서 실행합니다.

Tip

OS shell 에서 while 문을 이용하여 일정한 시간 간격으로 load query 의 결과를 확인합니다.

9 다른 세션을 열고, load query 명령어를 이용하여 load 명령어의 진행 상태를 확인합니다.

```
$ db2 connect to sample
```

```
   데이터베이스 연결 정보

 데이터베이스 서버                        = DB2/AIX64 8.2.3
 SQL 권한 부여 ID                         = POST01
 로컬 데이터베이스 별명                   = SAMPLE
```

```
$ db2 load query table kes.empl summaryonly
```

```
------------------------------------------------------------------
SQL3532I  로드 유틸리티가 현재 "LOAD" 단계에 있습니다.

읽은 행 수                               = 10
건너뛴 행 수                             = 0
로드된 행 수                             = 8
거부된 행 수                             = 2
삭제된 행 수                             = 0
커미트된 행 수                           = 0
경고 수                                  = 3

테이블 상태:
  점검 보류
  로드 진행
------------------------------------------------------------------
SQL3532I  로드 유틸리티가 현재 "BUILD" 단계에 있습니다.
SQL3533I  로드 유틸리티가 현재 "2"의 인덱스 "2"을(를) 작성 중입니다.
------------------------------------------------------------------
SQL3532I  로드 유틸리티가 현재 "DELETE" 단계에 있습니다.
SQL3534I  로드 삭제 단계는 거의 "100"퍼센트 완료되었습니다.
------------------------------------------------------------------
SQL3532I  로드 유틸리티가 현재 "UNKNOWN" 단계에 있습니다.
테이블 상태:
  점검 보류
  로드 진행
```

TOPIC 09 14 LOAD 시나리오

Point

LOAD 명령어로 입력 파일과 예외 테이블을 이용하여 실행합니다. '백업 보류' 상태에서는 BACKUP DB 명령어가 필요하며, '점검 보류' 상태에서는 SET INTEGRITY 명령어가 필요합니다. 실행 후에는 메시지 파일, 덤프 파일, 예외 테이블을 확인합니다.

Tip

LOAD 단계에서는 10 개의 행 중에서 8 개의 행이 입력되었습니다.

- 7 행의 5번째 컬럼의 값이 NULL로 변환되어 입력되었습니다.
- 8 행의 두 번째 컬럼의 값이 누락되어 입력되지 못했습니다.
- 10 행의 첫 번째 컬럼의 값이 데이터 유형이 맞지 않고, NULL 값도 허용되지 않으므로 입력되지 못했습니다.
- 8 행과 10 행의 데이터는 덤프파일인 kes.dmp에 저장되었습니다.
- 거부된 행 수는 2건으로 8행과 10 행입니다.

Tip

BUILD 단계에서는 2 개의 인덱스가 다시 생성되었습니다.

- KES.EMPL_PK01
- KES.EMPL_UK01

Tip

DELETE 단계에서는 8 개의 행 중에서 2 개의 행이 삭제되었습니다.

- 2 행의 첫 번째 필드의 값이 empl_pk01에 위반되므로 삭제되었습니다.
- 4 행의 여섯 번째 필드의 값이 empl_uk01에 위반되므로 삭제되었습니다.
- 2 행과 4 행의 데이터는 예외테이블인 kes.emplexcp에 저장되었습니다.
- 삭제된 행수는 2건으로 2행과 4행입니다.

Tip

load 명령어로 입력된 행 수는 6건으로 1, 3, 5, 6, 7, 9 행의 데이터가 목표 테이블인 kes.empl 테이블에 입력되었습니다.

10 load 명령어가 완료되면, 메시지 파일을 확인하여 유효하지 못한 데이터로 분류되어 테이블에 추가되지 못한 데이터와 그 이유를 확인합니다.

```
$ cat kes.msgs

SQL3109N  유틸리티가 파일 "/data/post01/work/kes.del"에서 데이터를
          로드하기 시작합니다.

SQL3500W  유틸리티가 "04/22/2006 01:50:01.243151"에서 "LOAD" 단계를
          시작 중입니다.

SQL3519W  일관성 지점 로드 시작. 입력 레코드 계수 = "0".
SQL3520W  일관성 지점 로드에 성공했습니다.

SQL3117W  행 "F0-7", 컬럼 "5"의 필드 값을 SMALLINT 값으로 변환할 수
          없습니다.  널(NULL)이 로드되었습니다.
SQL3116W  행 "F0-8", 컬럼 "2"의 필드 값이 누락되었으나, 목표 컬럼이
          널(NULL) 입력 가능하지 않습니다.
SQL3185W  입력 파일의 "F0-8" 행에서 데이터를 처리하는 동안 이전의
          오류가 발생했습니다.
SQL3120W  행 "F0-10", 컬럼 "1"의 필드 값을 INTEGER 값으로 변환할 수
          는 없지만, 목표 컬럼이 널(NULL) 입력 가능하지 않습니다.
          행이 로드되지 않았습니다.
SQL3185W  입력 파일의 "F0-10" 행에서 데이터를 처리하는 동안 이전의
          오류가 발생했습니다.

SQL3227W  레코드 토큰 "F0-7"은(는) 사용자 레코드 번호 "7"을(를) 참조합
SQL3227W  레코드 토큰 "F0-8"은(는) 사용자 레코드 번호 "8"을(를) 참조합
SQL3227W  레코드 토큰 "F0-10"은(는) 사용자 레코드 번호 "10"을(를) 참조

SQL3110N  유틸리티가 처리를 완료했습니다. 입력 파일에서 "10"개의 행
          을 읽었습니다.
SQL3519W  일관성 지점 로드 시작. 입력 레코드 계수 = "10".
SQL3520W  일관성 지점 로드에 성공했습니다.
SQL3515W  유틸리티가 "04/22/2006 01:50:01.481270"에 "LOAD" 단계를 완료

SQL3500W  유틸리티가 "04/22/2006 01:50:01.493816"에서 "BUILD" 단계
          를 시작 중입니다.
SQL3213I  인덱싱 모드는 "REBUILD"입니다.
SQL3515W  유틸리티가 "04/22/2006 01:50:01.720362"에 "BUILD" 단계를
          완료했습니다.

SQL3500W  유틸리티가 "04/22/2006 01:50:02.192915"에서 "DELETE" 단계
          를 시작 중입니다.
SQL3509W  유틸리티가 테이블에서 "2"개의 행을 삭제했습니다.
SQL3515W  유틸리티가 "04/22/2006 01:50:02.261254"에 "DELETE" 단계를
          완료했습니다.

SQL3107W  메시지 파일에 적어도 하나의 경고 메시지가 있습니다.

읽은 행 수                                  = 10
건너뛴 행 수                                = 0
로드된 행 수                                = 8
거부된 행 수                                = 2
삭제된 행 수                                = 2
커미트된 행 수                              = 10
```

TOPIC 09 14 LOAD 시나리오

Point

LOAD 명령어로 입력 파일과 예외 테이블을 이용하여 실행합니다. '백업 보류' 상태에서는 BACKUP DB 명령어가 필요하며, '점검 보류' 상태에서는 SET INTEGRITY 명령어가 필요합니다. 실행 후에는 메시지 파일, 덤프 파일, 예외 테이블을 확인합니다.

Tip

load 명령어에서 거부된 경우 8 행과 10 행의 데이터는 덤프 파일인 kes.dmp에서 확인됩니다.

Tip

load 명령어에서 거부된 경우 2 행과 4 행의 데이터는 예외 테이블인 kes.emplexcp에서 확인됩니다.

Tip

12번 또는 13번은 load 명령어가 실패한 경우에만 실행합니다.

Tip

REPLACE 모드를 지정한 경우에는 모든 데이터가 삭제됩니다.

11 덤프 파일을 확인하면 유효하지 못한 데이터가 저장된 것을 확인합니다.

```
$ cat kes.dmp.load.000
18,    ,"M", 2, 140, "god@kr.ibm.com", "2001-11-29"
"20","엄정화","F", 1, 160, "ejw@kr.ibm.com", "2001-04-28"
```

12 예외 테이블을 이용하여 고유 인덱스를 위반한 행이 입력되었는지 확인합니다.

```
$ db2 "SELECT SMALLINT(id) id, SUBSTR(name,1,6) name, mydept, sex,
  SUBSTR(email,1,15) email, SUBSTR(msg,1,30) msg FROM
  kes.emplexcp"

ID     NAME   MYDEPT SEX EMAIL           MSG
------ ------ ------ --- --------------- ------------------------------
    14 김현정      3 F   lkc@kr.ibm.com  00001I0000500002
    11 김경호      2 M   kkh@kr.ibm.com  00001I0000500001
2 레코드가 선택됨.
```

13 load 명령어를 실행 도중에 오류가 발생하면, RESTART 옵션으로 마지막 실패 지점 이후부터 계속할 수 있습니다. kes.db2 파일에서 INSERT옵션을 RESTART 옵션으로 변경하여 다시 실행합니다.

```
$ vi kes.db2
LOAD FROM /work/kes.del OF DEL
SAVECOUNT 10000
MODIFIED BY dumpfile /work/kes.dmp
MESSAGES /work/kes.msgs
RESTART INTO kes.empl
FOR EXCEPTION kes.emplexcp;
$ db2 –stvf kes.db2
```

14 load 명령어를 실행 도중에 오류가 발생하면, TERMINATE 옵션으로 load 명령어를 취소할 수 있습니다. kes.db2 파일에서 INSERT옵션을 TERMINATE 옵션으로 변경하여 다시 실행합니다. 테이블의 데이터는 load 명령어를 실행하기 전으로 복구됩니다.

```
$ vi kes.db2
LOAD FROM /work/kes.del OF DEL
SAVECOUNT 10000
MODIFIED BY dumpfile /work/kes.dmp
MESSAGES /work/kes.msgs
TERMINATE INTO kes.empl
FOR EXCEPTION kes.emplexcp;
$ db2 –svtf kes.db2
```

TOPIC 09 14 LOAD 시나리오

Point

LOAD 명령어로 입력 파일과 예외 테이블을 이용하여 실행합니다. '백업 보류' 상태에서는 BACKUP DB 명령어가 필요하며, '점검 보류' 상태에서는 SET INTEGRITY 명령어가 필요합니다. 실행 후에는 메시지 파일, 덤프 파일, 예외 테이블을 확인합니다.

Tip

아카이브 로깅 모드이고, 기존 백업 COPY NO 옵션이 사용된 경우에만 해당 테이블스페이스가 백업 보류 상태가 됩니다.

Tip

테이블스페이스가 정상 상태인 경우에는 백업이 필요하지 않습니다.

15 목표 테이블이 속한 테이블스페이스의 상태를 확인합니다.

```
$ db2 "SELECT SUBSTR(tbspace,1,18) tablespace FROM syscat.tables
  WHERE tabschema = 'KES' AND tabname = 'EMPL' "

TABLESPACE
------------------
TS02
1 레코드가 선택됨.
```

```
$ db2 LIST TABLESPACES

테이블 스페이스 ID                    = 4
이름                                  = TS02
유형                                  = 데이터베이스 관리 스페이스
내용                                  = 임의의 데이터
상태                                  = 0x0020
  세부사항 설명:
    백업 보류
```

16 '백업 보류' 상태를 해결하려면, BACKUP DB 명령어로 테이블스페이스 ts02를 백업합니다.

```
$ db2 "BACKUP DB sample TABLESPACE(ts02) ONLINE "
백업이 완료되었습니다. 이 백업 이미지에 대한 시간소인은 20060422011035
```

17 BACKUP DB 명령어를 실행한 세션의 데이터베이스 접속은 자동적으로 해제됩니다. 다시 데이터베이스에 접속합니다.

```
$ db2 CONNECT TO sample

   데이터베이스 연결 정보

데이터베이스 서버                    = DB2/AIX64 8.2.3
SQL 권한 부여 ID                     = POST01
로컬 데이터베이스 별명               = SAMPLE
```

18 목표 테이블의 데이터를 확인합니다.

```
$ db2 "SELECT * FROM kes.empl"

ID         NAME                          SEX MYDEPT SALARY EMAIL
---------- ----------------------------- --- ------ ------ ----------
SQL0668N  이유 코드 "1"(으)로 인해 테이블 "KES.EMPL"에서 조작이 허용되
않습니다.  SQLSTATE=57016
```

TOPIC 09 14 LOAD 시나리오

Point

LOAD 명령어로 입력 파일과 예외 테이블을 이용하여 실행합니다. '백업 보류' 상태에서는 BACKUP DB 명령어가 필요하며, '점검 보류' 상태에서는 SET INTEGRITY 명령어가 필요합니다. 실행 후에는 메시지 파일, 덤프 파일, 예외 테이블을 확인합니다.

Tip

외부키 또는 일반에 대한 점검 제한 조건이 있는 경우에만, 테이블이 '점검 보류' 상태가 됩니다.

Tip

SET INTEGRITY 명령어로 LOAD 명령어가 입력했던 6 개의 행 중에서 2 개의 행이 삭제되었습니다.

- 5 행의 세 번째 필드의 값이 empl_cc01에 위반되므로 삭제되었습니다.
- 9 행의 네 번째 필드의 값이 empl_fk01에 위반되므로 삭제되었습니다.
- 삭제된 행 수는 2건으로 5행과 9행입니다.
- kes.empl 테이블에 남은 행 수는 4 건으로 1, 3, 6, 7 행의 데이터가 목표 테이블인 kes.empl에 남았습니다.

Tip

SET INTEGRITY 명령으로 삭제된 행인 5행과 9행의 데이터가 예외 테이블인 kes.emplexcp에 추가되었습니다. LOAD 명령어에서 삭제된 행인 2행과 4행의 데이터를 포함해 총 4건의 데이터가 예외 테이블인 kes.emplexcp에서 확인됩니다.

19 SQL0668N 은 '점검 보류' 상태를 의미합니다. 정확한 에러 메시지를 확인합니다.

```
$ db2 "? SQL0668N"
SQL0668N 이유 코드 "<reason-code>"(으)로 인해 테이블
         "<table-name>"에서 조작이 허용되지 않습니다.
설명:
 테이블 "<table-name>"에 대한 액세스가 제한됩니다.  원인은
"<reason-code>"에 기초합니다.
1 테이블이 점검 보류 상태입니다.  테이블의 무결성이 적용되지
않았으므로 테이블 내용이 유효하지 않을 수 있습니다.  종속
테이블이 점검 보류 상태이면 점검 보류 상태가 아닌 상위 테이블이나
기본 테이블에서 조작을 수행할 경우에도 이 오류를 수신할 수
있습니다.
```

20 시스템 카탈로그에서 테이블에 대한 정보를 확인하면, 테이블에 대한 제한 조건이 점검되지 않았다는 것을 확인할 수 있습니다.

```
$ db2 "SELECT const_checked FROM syscat.tables WHERE
  tabschema = 'KES' AND tabname = 'EMPL' "

CONST_CHECKED
--------------------------------
NNYYYYYYYYYYYYYYYYYYYYYYYYYYYY
1 레코드가 선택됨.
```

21 '점검 보류' 상태를 해결하려면, SET INTEGRITY 명령어로 외부키와 점검 제한 조건을 위반한 행을 점검합니다. 위반한 행은 예외 테이블에 저장됩니다.

```
$ db2 "SET INTEGRITY FOR kes.empl IMMEDIATE CHECKED FOR
  EXCEPTION IN kes.empl USE kes.emplexcp"
SQL3602W  데이터 점검 처리시 제한조건 위반이 발견되어 예외 테이블로
이동시켰습니다.  SQLSTATE=01603
```

22 예외 테이블을 이용하여 외부키와 점검 제한 조건을 위반한 행이 입력되었는지 확인합니다.

```
$ db2 "SELECT SMALLINT(id) id, SUBSTR(name,1,6) name, mydept, sex,
  SUBSTR(email,1,15) email, SUBSTR(msg,1,30) msg FROM
  kes.emplexcp"

ID     NAME   MYDEPT SEX EMAIL           MSG
------ ------ ------ --- --------------- ------------------------------
    14 김현정      3 F   lkc@kr.ibm.com  00001I0000500002
    11 김경호      2 M   kkh@kr.ibm.com  00001I0000500001
    15 김건모      2 m   kkm@kr.ibm.com  00001K00018KES.EMPL.EMPL_CC01
    19 신화        4 M   sh@kr.ibm.com   00001F00018KES.EMPL.EMPL_FK01
4 레코드가 선택됨.
```

TOPIC 09 14 LOAD 시나리오

Point

LOAD 명령어로 입력 파일과 예외 테이블을 이용하여 실행합니다. '백업 보류' 상태에서는 BACKUP DB 명령어가 필요하며, '점검 보류' 상태에서는 SET INTEGRITY 명령어가 필요합니다. 실행 후에는 메시지 파일, 덤프 파일, 예외 테이블을 확인합니다.

Tip

kes.del 파일에 있던 10 건의 데이터 중에서 1, 3, 6, 7 행의 데이터만 입력되었습니다. kes.empl 테이블에 원래 있던 4 건을 포함하여 8 건의 데이터가 목표 테이블인 kes.empl 에서 확인됩니다.

23 목표 테이블의 최종 데이터를 확인합니다.

```
$ db2 "SELECT id, name, mydept, sex, email FROM kes.empl ORDER
  BY id"

ID          NAME                           MYDEPT SEX EMAIL
----------- ------------------------------ ------ --- ---------------
          1 KES                                 1 F   kes@kr.ibm.com
          2 KHY                                 3 F   khy@kr.ibm.com
          3 JHS                                 2 F   jhs@kr.ibm.com
          4 JJY                                 2 M   jjy@kr.ibm.com
         11 이문세                              1 M   lms@kr.ibm.com
         13 이기찬                              1 M   lkc@kr.ibm.com
         16 제이                                1 F   j@kr.ibm.com
         17 양희은                              2 F   yhe@kr.ibm.com
8 레코드가 선택됨.
```

TOPIC 09 15 Cursor Load

Point

사용자 정의 커서로부터 데이터를 입력 받아 로드하는 기능을 Cursor Load라고 합니다. 데이터 이동 적재를 위해 중간 SAM 파일을 생성하지 않고 직접 로드할 수 있어 사용자 편의성을 제공합니다.

Tip

SELECT구문은 Join을 포함하여 어떤 종류의 조회 문도 사용될 수 있습니다. 테이블도 동일 데이터베이스 내 테이블이나 또는 다른 데이터베이스에 Federation된 Nickname 오브젝트도 사용 가능합니다.

Tip

하나의 작업(cursorload.sql)에서는 100개까지 Cursor를 선언할 수 있으며, 임의의 이름을 부여할 수 있습니다.

Tip

이기종 데이터의 Federation을 위해Infosphere Federation Server 제품을 설치하십시오.

Tip

Federation구성 방법은 Infosphere Federation Server 매뉴얼을 참조하십시오.

1 먼저 사용자 정의 커서를 선언한 후, Cursor로부터 데이터를 로드합니다.

```
$cat cursorload.sql

DECLARE cur1 CURSOR for select * from  ORA10.CUSTOMER WITH UR;
LOAD FROM cur1 OF CURSOR INSERT INTO DB2.CUSTOMER;
DECLARE cur2 CURSOR for select * from  ORA10.ORDERS WITH UR;
LOAD FROM cur2 OF CURSOR INSERT INTO DB2.ORDERS;
DECLARE cur3 CURSOR for select * from  ORA10.CUSTOMER WITH UR;
LOAD FROM cur3 OF CURSOR INSERT INTO DB2.CUSTOMER;
DECLARE cur4 CURSOR for select * from  ORA10.CUSTOMER WITH UR;
LOAD FROM cur4 OF CURSOR INSERT INTO DB2.CUSTOMER;

$db2 –stvf cursorload.sql
```

2 이 기종 데이터베이스로부터 데이터를 이관하고자 하는 경우에는 Federation을 구성하고, 미리 사용될 테이블들에 대해 아래와 같이 Nickname을 정의합니다.

```
$cat createnick.sql

CREATE NICKNAME ORA10.CUSTOMER for ora10.DBOWN.CUSTOMER;
CREATE NICKNAME ORA10.ORDER for ora10.DBOWN.ORDERS;
CREATE NICKNAME ORA10.PRODUCT for ora10.DBOWN.PRODUCT;
CREATE NICKNAME ORA10.LINEITEM for ora10.DBOWN.LINEITEM;

$db2 –stvf createnick.sql
```