13. 시퀀스 인덱스 동의어

컴퓨터공학부

김은경

[학습 목표]

1. CREATE SEQUENCE 명령을 사용하여 시퀀스를 생성할 수 있다.
2. NEXTVAL과 CURRVAL이라는 의사열을 사용해서 각각 시퀀스를 생성하고 시퀀스의 현재 값을 확인할 수 있다.
3. ALTER SEQUENCE 명령을 사용하여 시퀀스를 수정할 수 있다.
4. CREATE INDEX 명령을 사용하여 인덱스를 생성할 수 있다.
5. CREATE SYNONYM 명령을 사용하여 동의어를 생성할 수 있다.

1. 시퀀스

시퀀스(Sequence)란?

- 자동으로 테이블의 행에 대한 유일한(Unique) 번호를 생성하는 DB 객체

시퀀스의 특징

- 여러 사용자가 공유 가능한 객체임
- 주로 기본 키 값을 생성하는데 사용됨
- 시퀀스 값을 액세스하는 효율성을 향상하기 위해서, 시퀀스 값을 미리 생성해서 메모리에 저장해 두고 사용할 수 있음
- 테이블과는 독립적으로 존재함
- 동일한 시퀀스가 여러 테이블에 대해 사용될 수 있음

1. 시퀀스

시퀀스 정의 방법

- 시퀀스를 정의하면 시퀀스의 초기값 등 여러 가지 옵션을 지정할 수 있음

```
CREATE  SEQUENCE   시퀀스_이름
                   [INCREMENT  BY  n]
                   [START  WITH  n]
                   [{MAXVALUE  n | NOMAXVALUE}]
                   [{MINVALUE  n | NOMINVALUE}]
                   [{CYCLE | NOCYCLE}]
                   [{CACHE n | NOCACHE}];
```

없다는 의미가 아니라, 너무 크거나 작아서 최대/최소 값 지정을 생략한다는 의미임

; 밑줄 표시한 것이 디폴트로 설정된 옵션임

1. 시퀀스

CREATE SEQUENCE 명령의 옵션들

옵 션	설 명
INCREMENT BY n	시퀀스 번호의 간격(Gap)을 n(정수)으로 설정함 (생략 시 '1'씩 증가함)
START WITH n	생성할 첫번째 시퀀스 번호를 n으로 지정함 (생략 시 '1'부터 생성함)
MAXVALUE n	생성할 시퀀스의 최대값을 n으로 설정함
NOMAXVALUE	오름차순용으로 10^27, 내림차순용으로 –1을 지정함 (최대값의 한계를 정하지 않겠다는 의미로 사용됨)
MINVALUE n	생성할 시퀀스의 최소값을 n으로 설정함
NOMINVALUE	오름차순용으로 1, 내림차순용으로 –(10^26)을 지정함 (최소값의 한계를 정하지 않겠다는 의미로 사용됨)
CYCLE | NOCYCLE	- 최대값 또는 최소값에 도달한 후, 값을 계속 생성할 지 여부를 지정함(디폴트 설정 : NOCYCLE) - CYCLE로 지정한 경우, 최대값 또는 최소값에 도달한 후 첫번째 시퀀스 번호부터 다시 생성함 - 단, 기본 키에 대해서는 CYCLE 옵션 지정이 불가능함
CACHE n | NOCACHE	얼마나 많은 시퀀스 값을 미리 생성해서 메모리에 저장하고 있을 지를 지정함 (디폴트 n 값은 20임. 즉, 20개의 시퀀스를 미리 생성해서 메모리에 저장해 둠으로써 시퀀스를 액세스하는 효율성을 향상시킴)

1. 시퀀스

✓ 시퀀스 정의 예

- test_dept 테이블의 기본 키 값을 생성하는데 사용될 시퀀스인 dept_id_seq를 생성하시오. 이때, 첫번째 시퀀스 값은 100이고, 최대 900까지, 100씩 증가시키며, 기본 키로 사용할 것이므로 최대값에 도달한 후에는 다시 첫번째 시퀀스 번호부터 생성하지 않아야 되며, 또 시퀀스 값을 미리 생성해서 메모리에 저장하지 않도록 하시오.

```
CREATE   TABLE    test_dept (
   id              Number(3)    PRIMARY KEY,
   name          Varchar2(20) NOT NULL,
   location      Varchar2(15));
```

```
CREATE   SEQUENCE dept_id_seq
                        INCREMENT   BY   100
                        START   WITH   100
                        MAXVALUE    900
                        NOCYCLE
                        NOCACHE;
```

1. 시퀀스

시퀀스 확인 방법

- 데이터 사전의 USER_SEQUENCES 테이블에 시퀀스에 관한 정보가 저장됨
- NOCACHE로 생성된 시퀀스에 한해서, 증가 없이 다음에 사용 가능한 시퀀스 값을 LAST_NUMBER 속성에서 확인할 수 있음
- 즉, LAST_NUMBER 열의 값이 다음에 사용 가능한 시퀀스 번호임

예)

```
SELECT   sequence_name,   min_value,   max_value,
         increment_by,   last_number
FROM   user_sequences;
```

SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1	900	100	100

2. 시퀀스 활용

의사열 NEXTVAL과 CURRVAL

- **의사열(Pseudo Column)이란?**
  - 시퀀스에 의해서 자동으로 생성되는 가상의 열을 의미하며, NEXTVAL과 CURRVAL이 있음

- **NEXTVAL**
  - 다음에 사용 가능한 시퀀스 값을 생성함
    → NEXTVAL에 의해서 새 시퀀스 값이 생성된 다음,
    그 값이 CURRVAL에 저장됨

- **CURRVAL**
  - 가장 최근에 생성된 현재의 시퀀스 값을 반환함

2. 시퀀스 활용

의사열 NEXTVAL과 CURRVAL

- **의사열 사용 형식**
  - 시퀀스 이름 뒤에 점(Dot)을 찍은 다음 NEXTVAL이나 CURRVAL을 기술함

시퀀스_이름.NEXTVAL 또는 시퀀스_이름.CURRVAL

- **주의사항**
  - CURRVAL이 참조되기 전에 반드시 NEXTVAL이 생성되어야 함
  - 의사열의 값을 확인할 때는 더미 테이블인 dual을 이용함

2. 시퀀스 활용

✓ 의사열 사용 예

- 앞에서 생성한 dept_id_seq 시퀀스를 이용해서, 다음에 사용할 새로운 시퀀스 값을 하나 생성한 다음, 현재 시퀀스 값이 얼마인지 확인하기

① NEXTVAL을 이용해서 새로운 시퀀스 값 생성하기

```
SELECT   dept_id_seq.NEXTVAL
FROM   dual;
```

NEXTVAL
100

② CURRVAL을 이용해서 현재 시퀀스 값 확인하기

```
SELECT   dept_id_seq.CURRVAL
FROM   dual;
```

CURRVAL
100

2. 시퀀스 활용

의사열 사용 규칙

- **의사열이 사용 가능한 경우**
  - 서브쿼리의 일부가 아닌 SELECT 문장의 SELECT 절
  - INSERT 문장에서 서브쿼리의 SELECT 절
  - INSERT 문장의 VALUES 절
  - UPDATE 문장의 SET 절

- **의사열이 사용 불가능한 경우**
  - 뷰의 SELECT 절
  - DISTINCT 키워드를 사용한 SELECT 문장
  - GROUP BY, HAVING, ORDER BY를 이용한 SELECT 문장
  - SELECT, DELETE, UPDATE 문장에서의 서브쿼리
  - CREATE TABLE, ALTER TABLE 명령문의 DEFAULT 표현식

2. 시퀀스 활용

✓ 시퀀스 사용 예

- 앞에서 생성한 test_dept 테이블에 '연구부'라는 부서를 하나 삽입하되, 부서번호는 시퀀스를 사용해서 삽입하기

① dept_id_seq 시퀀스를 이용해서 행 삽입

```
INSERT  INTO  test_dept(id,  name)
VALUES  (dept_id_seq.NEXTVAL,  '연구부');
```

② test_dept 테이블의 데이터 검색

```
SELECT  *
FROM    test_dept;
```

ID	NAME	LOCATION
100	연구부	

3. 시퀀스 오류

시퀀스 오류 원인

① 롤백(ROLLBACK)이 발생한 경우
- 시퀀스를 포함한 문장을 롤백하면, 커밋 이후 롤백 이전에 생성된 시퀀스 번호를 모두 잃게 되므로 이후 시퀀스에 간격이 생김

② 시스템이 손상된 경우
- CACHE 옵션을 설정해서 미리 시퀀스를 생성해서 메모리에 저장해 둔 경우, 시스템의 손상으로 비정상적으로 종료하게 되면 미리 생성한 시퀀스 값을 모두 잃어버리게 되므로, 이후 시퀀스에 간격이 생김

③ 동일한 시퀀스가 다중 테이블에서 사용된 경우
- 시퀀스 값이 불규칙적으로 변할 수 있음

4. 시퀀스 수정

시퀀스 수정 방법

- START WITH 옵션을 제외하고, 시퀀스를 생성할 때 지정한 여러 옵션의 값을 변경할 수 있음

```
ALTER  SEQUENCE  시퀀스_이름
                 [INCREMENT  BY  n]
                 [{MAXVALUE  n | NOMAXVALUE}]
                 [{MINVALUE  n | NOMINVALUE}]
                 [{CYCLE | NOCYCLE}]
                 [{CACHE n | NOCACHE}];
```

4. 시퀀스 수정

시퀀스 수정 지침

- 시퀀스 생성자나 ALTER 권한을 가진 사용자만 수정할 수 있음
- 시퀀스를 수정하면, 수정 이후 생성되는 시퀀스 번호에만 영향을 미침
- 시퀀스를 생성할 때 다른 시작 번호부터 다시 생성하려면, 기존 시퀀스를 삭제하고 다시 생성해야 함
- 유효성 검사를 자동으로 수행함

  예) 만약 수정된 MAXVALUE가 현재 시퀀스 번호보다 작은 경우, 수정이 허용되지 않음

4. 시퀀스 수정

✓ 시퀀스 수정 예

- 시퀀스 dept_id_seq의 증가치를 100에서 50으로 수정하고, 최대값을 2000으로 변경하기

```
ALTER   SEQUENCE dept_id_seq
        INCREMENT   BY   50
        MAXVALUE    2000;
```

```
Sequence altered.
```

```
SELECT   sequence_name,   min_value,   max_value,
           increment_by,   last_number
FROM   user_sequences;
```

SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
DEPT_ID_SEQ	1	2000	50	150

4. 시퀀스 수정

→ 변경한 시퀀스를 이용해서 test_dept 테이블에 '개발부' 행을 하나 삽입한 후, 현재의 시퀀스 값을 확인하기 :

```
INSERT INTO   test_dept(id, name)
VALUES    (dept_id_seq.NEXTVAL, '개발부');
SELECT * FROM   test_dept;
```

ID	NAME	LOCATION
100	연구부	
150	개발부	

```
SELECT   dept_id_seq.CURRVAL
FROM   dual;
```

CURRVAL
150

5. 시퀀스 삭제

시퀀스 삭제 방법

- 더 이상 필요 없는 시퀀스의 이름을 명시함

```
DROP  SEQUENCE  시퀀스_이름;
```

✓ 주의사항

- 한 번 제거된 시퀀스는 더 이상 참조될 수 없음

5. 시퀀스 삭제

✓ 시퀀스 삭제 예

- 앞에서 생성한 dept_id_seq 시퀀스를 삭제한 다음, 이 시퀀스의 현재 값을 확인하시오.

```
DROP   SEQUENCE   dept_id_seq;
```

```
Sequence dropped.
```

```
SELECT    dept_id_seq.CURRVAL
FROM      dual;
```

```
SELECT   dept_id_seq.CURRVAL
         *

ERROR at line 1:
ORA-02289: sequence does not exist
```

6. 인덱스

인덱스(Index)란?

- 포인터를 사용해서 행의 검색을 촉진할 수 있는 DB 객체

인덱스의 특징

- 테이블 행에 대한 직접적이고 빠른 액세스를 제공함
- 인덱스는 오라클 서버에 의해서 자동으로 생성되거나, 사용자에 의해 명시적으로 생성될 수 있음
- 인덱스는 오라클 서버에 의해서 자동으로 사용되고 유지됨
- 인덱스는 테이블과는 논리적, 물리적으로 독립적임
- 기본 테이블에 영향을 주지않고 생성하거나 제거할 수 있음
- 기본 테이블을 제거하면, 인덱스도 자동으로 제거됨
- 인덱스를 너무 많이 생성하면 오히려 DML 처리의 효율을 저하시키게 됨

6. 인덱스

인덱스가 필요한 경우

① 열이 WHERE 절이나 조인 조건에서 자주 사용되는 경우

② 열이 광범위한 값을 포함하는 경우

③ 열이 많은 수의 널 값을 포함하는 경우

→ 널 값에 대해서는 인덱스가 생성되지 않으므로, 널 값이 많을수록 인덱스의 크기가 작아짐

④ 둘 또는 그 이상의 열들이 WHERE 절 또는 조인 조건에서 자주 함께 사용되는 경우

⑤ 테이블이 대형이고, 대부분의 질의가 행의 2~4%보다 적게 읽어 들일 것으로 예상되는 경우

6. 인덱스

인덱스가 불필요한 경우

① 테이블 사이즈가 작은 경우

② 해당 열이 질의의 조건으로 자주 사용되지 않는 경우

③ 테이블이 자주 갱신되는 경우

→ 인덱스 유지를 위해 DML의 효율이 나빠짐

④ 대부분의 질의가 행의 2~4% 이상을 읽어 들일 것으로 예상되는 경우

→ 인덱스를 생성하는 대신 테이블 전체를 검색하는 것이 좋음

7. 인덱스 생성

인덱스 생성 방법

① 자동 인덱스 생성

- 테이블을 생성할 때 PRIMARY KEY나 UNIQUE 제약 조건이 정의된 열에 대해서 오라클 서버가 유일한 인덱스를 자동으로 생성함

② 수동 인덱스 생성

- 검색 속도의 향상을 위해 사용자가 CREATE INDEX 명령을 사용해서 명시적으로 특정 열에 대해 유일하지 않는 인덱스를 생성할 수 있음
- 하나 이상의 열에 대해서 하나의 인덱스를 생성할 수 있음
- 널 값에 대해서는 인덱스가 생성되지 않음

7. 인덱스 생성

수동 인덱스 생성 명령

- ON 절에 어떤 테이블의 어떤 열(들)에 대해 인덱스를 생성할 지를 명시함

```
CREATE  INDEX  인덱스_이름
ON  테이블 이름(열 이름1[, 열 이름2, . . .];
```

✓ 수동 인덱스 생성 예

- course 테이블의 cour_name 열에 대한 검색이 빈번히 발생하므로, 이 검색 속도를 향상시킬 수 있도록 인덱스를 생성하시오.

```
CREATE  INDEX  cour_name_idx
ON  course(cour_name);
```

8. 인덱스 확인

인덱스 확인 방법

- 데이터 사전의 USER_INDEXES 및 USER_IND_COLUMNS 뷰에 저장된 인덱스 정보를 확인함

① USER_INDEXES 데이터 사전 뷰

- 인덱스 이름(index_name)과 인덱스의 유일성(uniqueness) 정보 등을 포함함

② USER_IND_COLUMNS 데이터 사전 뷰

- 인덱스 이름(index_name), 테이블 이름(table_name), 열 이름 (column_name), 열 위치(column_position) 등의 정보를 포함함

8. 인덱스 확인

✓ 인덱스 확인 예

- course 테이블에 대해서 생성된 인덱스의 이름과, 인덱스가 생성된 열 이름, 열의 위치 그리고 그 인덱스의 유일성을 확인하시오.

```
SELECT  ic.index_name, ic.column_name, ic.column_position  col_pos,
        ix.uniqueness
FROM    user_indexes ix, user_ind_columns ic
WHERE   ic.index_name = ix.index_name
AND     ic.table_name = 'COURSE';
```

두 개의 데이터 사전 테이블을 조인해서 인덱스 이름과 인덱스의 유일성 등을 한꺼번에 검색할 수 있음

INDEX_NAME	COLUMN_NAME	COL_POS	UNIQUENESS
COURSE_C_NUM_PK	COUR_ID	1	UNIQUE
COUR_NAME_IDX	COUR_NAME	1	NONUNIQUE

9. 인덱스 삭제

인덱스 삭제 방법

- DROP INDEX 명령을 사용해서 데이터 사전에서 인덱스를 삭제함

```
DROP INDEX 인덱스_이름;
```

✓ 주의사항

- 인덱스 소유자나 DROP ANY INDEX 권한을 가진 사용자만 삭제할 수 있음
- 인덱스는 수정될 수 없으며, 수정이 필요한 경우 삭제하고 다시 생성해야 함

9. 인덱스 삭제

✓ 인덱스 삭제 예

- 데이터 사전에서 cour_name_idx 인덱스를 제거한 다음, 제대로 삭제되었는지 확인하시오.

```
drop index  cour_name_idx
```

```
Index dropped.
```

```
SELECT  index_name, column_name
FROM    user_ind_columns
WHERE   table_name = 'COURSE';
```

INDEX_NAME	COLUMN_NAME
COURSE_C_NUM_PK	COUR_ID

10. 동의어

동의어(Synonym)란?

- DB 객체에 부여한 다른 이름, 즉 별칭을 의미함

동의어의 용도

- 뷰처럼 긴 이름을 가진 객체의 이름을 단축할 수 있음
- 다른 사용자가 소유한 테이블을 참조할 때 동의어를 사용하면, 객체에 대한 접근을 단순화시킬 수 있음
  (원래 테이블 이름 앞에 "소유자명"을 붙어야 하는데, 동의어를 사용하면 생략할 수 있음)

동의어 종류

① 개별 동의어 : 동의어를 생성한 사용자만 사용할 수 있는 동의어

② 공용 동의어 : 모든 DB 사용자가 사용할 수 있는 동의어

11. 동의어 생성

동의어 생성 방법

- FOR 절에 동의어를 생성하는 대상 객체의 이름을 명시함

```
CREATE [PUBLIC] SYNONYM 동의어_이름
FOR 객체_이름;
```

✓ PUBLIC 옵션을 지정하면 모든 사용자가 접근 가능한 동의어를 생성함

11. 동의어 생성

✓ 동의어 생성 예

- 앞에서 생성한 뷰인 empview300에 e_300이라는 간단한 이름을 하나 생성한 다음, 그 동의어를 사용해서 그 뷰의 모든 데이터를 검색하시오.

```
CREATE   SYNONYM   e_300
FOR      empview300;

SELECT   *
FROM     e_300;
```

단축된 동의어를 사용하면 명령문이 훨씬 간단해 짐

EMP_ID	EMP_NAME	JOB	DEPT_LOC
6351	진대준	부장	서울
7910	이영희	경리	서울
7920	김마리아	사무직	서울
7696	이혜성	사무직	서울

✓ DBA가 권한을 부여해야 동의어를 생성할 수 있음(일반 사용자는 생성 불가함)

12. 동의어 삭제

동의어 삭제 방법

- 삭제할 동의어의 이름을 지정해서 삭제함
- 공용 동의어의 경우 DBA가 PUBLIC 옵션을 지정해서 삭제할 수 있음

DROP [PUBLIC] SYNONYM 동의어_이름;

✓ **동의어 삭제 예 :**

CREATE SYNONYM e_300;

Q & A