Chap13. 객체 종류

- 13-1 데이터베이스를 위한 데이터를 저장한 데이터사전
- 13-2 더 빠른 검색을 위한 인덱스
- 13-3 테이블처럼 사용하는 뷰
- 13-4 규칙에 따라 순번을 생성하는 시퀀스
- 13-5 공식 별칭을 지정하는 동의어

13-1 데이터베이스를 위한 데이터를 저장한 데이터사전

- 데이터 사전이란?
  - 데이터베이스를 구성하고 운영하는데 필요한 정보
  - 데이터 사전 뷰

접두어	설명
USER_XXXX	현재 데이터베이스에 접속한 사용자가 소유한 객체 정보
ALL_XXXX	현재 데이터베이스에 접속한 사용자가 소유한 객체 또는 다른 사용자가 소유한 객체 중 사용 허가를 받은 객체, 즉 사용 가능한 모든 객체 정보
DBA_XXXX	데이터베이스 관리를 위한 정보(데이터베이스 관리 권한을 가진 SYSTEM, SYS 사용자만 열람 가능)
V$_XXXX	데이터베이스 성능 관련 정보(X$_XXXX 테이블의 뷰)

13-1 데이터베이스를 위한 데이터를 저장한 데이터사전

- USER_ 접두어를 가진 데이터 사전
  - 접속한 사용자가 소유한 객체 정보
- ALL_ 접두어를 가진 데이터 사전
  - 접속한 사용자가 소유하거나, 사용을 허가받은 객체 정보
- DBA_ 접두어를 가진 데이터 사전
  - 데이터베이스 관리 권한을 가진 사용자가 조회가능

실습 13-1 SCOTT 계정에서 사용 가능한 데이터 사전 살펴보기(DICT 사용)

```
01  SELECT * FROM DICT;
```

실습 13-2 SCOTT 계정에서 사용 가능한 데이터 사전 살펴보기(DICTIONARY 사용)

```
01  SELECT * FROM DICTIONARY;
```

:: 결과 화면(실습 13-1, 13-2의 실행 결과가 같음)

TABLE_NAME	COMMENTS
USER_AW_PROP	Object properties in Analytic Workspaces owned by the user
USER_AW_PS	Pagespaces in Analytic Workspaces owned by the user
USER_BASE_TABLE_MVIEWS	All materialized views with log(s) owned by the user in the database
USER_CATALOG	Tables, Views, Synonyms and Sequences owned by the user

13-1 데이터베이스를 위한 데이터를 저장한 데이터사전

- USER_ 접두어를 가진 데이터 사전
  - 접속한 사용자가 소유한 객체 정보

실습 13-3 SCOTT 계정이 가지고 있는 객체 정보 살펴보기(USER_ 접두어 사용)

```
01  SELECT TABLE_NAME
02    FROM USER_TABLES;
```

접두어 뒤에 복수형 단어로 이름을 구성하고 있음

:: 결과 화면(일부 데이터만 표시함)

TABLE_NAME
DEPT
EMP
BONUS
SALGRADE

13-1 데이터베이스를 위한 데이터를 저장한 데이터사전

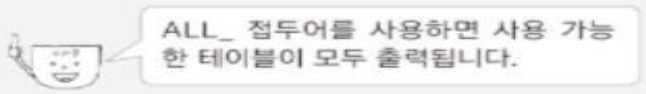
- ALL_ 접두어를 가진 데이터 사전
  - 접속한 사용자가 소유하거나, 사용을 허가받은 객체 정보

실습 13-4 SCOTT 계정이 사용할 수 있는 객체 정보 살펴보기(ALL_ 접두어 사용)

```
01  SELECT OWNER, TABLE_NAME
02    FROM ALL_TABLES;
```

:: 결과 화면(일부 데이터만 표시함)

	OWNER	TABLE_NAME
▶	SYS	DUAL
	SYS	SYSTEM_PRIVILEGE_MAP
	SYS	TABLE_PRIVILEGE_MAP
	SYS	STMT_AUDIT_OPTION_MAP
	SYS	AUDIT_ACTIONS
	SYS	WRR$_REPLAY_CALL_FILTER
	SYS	HS_BULKLOAD_VIEW_OBJ



열 이름	자료형	NULL 여부	설명
OWNER	VARCHAR2(30)	NOT NULL	테이블을 소유한 사용자 (ALL_TABLES에만 존재)
TABLE_NAME	VARCHAR(30)	NOT NULL	테이블 이름
TABLESPACE_NAME	VARCHAR(30)		테이블 스페이스 이름
NUM_ROWS	NUMBER		테이블에 저장된 행 수

13-1 데이터베이스를 위한 데이터를 저장한 데이터사전

- DBA_ 접두어를 가진 데이터 사전
  - 데이터베이스 관리 권한을 가진 사용자가 조회가능

실습 13-3 SCOTT 계정이 가지고 있는 객체 정보 살펴보기(USER_ 접두어 사용)

```
01  SELECT TABLE_NAME
02    FROM USER_TABLES;
```

13-1 데이터베이스를 위한 데이터를 저장한 데이터사전

- DBA_ 접두어를 가진 데이터 사전
  - 데이터베이스 관리 권한을 가진 사용자가 조회가능

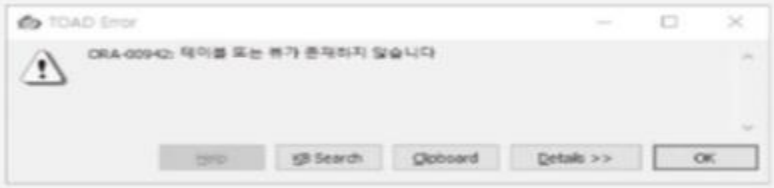
DBA_ 접두어를 가진 데이터 사전

DBA_ 접두어를 가진 데이터 사전은 데이터베이스 관리 권한을 가진 사용자만 조회할 수 있는 테이블로서 SCOTT 계정으로는 조회가 불가능합니다.

실습 13-5 SCOTT 계정으로 DBA_ 접두어 사용하기

```
01  SELECT * FROM DBA_TABLES;
```

:: 결과 화면



한 발 더 나가기! 왜 데이터베이스에 존재하는데 '존재하지 않습니다'라고 출력되나요?

실습 13-5의 결과 화면을 조금 더 자세히 살펴볼까요? DBA_TABLES는 분명 오라클 데이터베이스에 존재하지만 사용 권한이 없는 SCOTT 계정으로 조회하면 테이블이 존재하지 않는다는 오류 메시지가 나옵니다. 이는 사용 권한이 없는 사용자는 해당 개체의 존재 여부조차 확인할 수 없음을 의미합니다. '존재는 하지만 너에겐 권한이 없어'라는 식의 문구는 보안 문제를 일으킬 수 있습니다.

13-1 데이터베이스를 위한 데이터를 저장한 데이터사전

- DBA_ 접두어를 가진 데이터 사전
  - 데이터베이스 관리 권한을 가진 사용자가 조회가능

실습 13-6 SYSTEM 계정으로 DBA_ 접두어 사용하기(SYSTEM 계정으로 접속했을 때)

```
01  SELECT * FROM DBA_TABLES;
```

:: 결과 화면

OWNER	TABLE_NAME	TABLESPACE_NAME	CLUSTER_NAME	IOT_NAME	STATUS
SYS	ICOL$	SYSTEM	C_OBJ#		VALID
SYS	CON$	SYSTEM			VALID
SYS	UNDO$	SYSTEM			VALID

실습 13-7 DBA_ USERS를 사용하여 사용자 정보를 알아보기(SYSTEM 계정으로 접속했을 때)

```
01  SELECT *
02    FROM DBA_USERS
03   WHERE USERNAME = 'SCOTT';
```

:: 결과 화면

USERNAME	USER_ID	PASSWORD	ACCOUNT_STATUS	LOCK_DATE	EXPIRY_DATE	DEFAULT_TABLESPACE
SCOTT	84		OPEN		2018-09-02 오후 7:41:14	USERS

13-2 더 빠른 검색을 위한 인덱스

- 인덱스
  - 데이터 검색 성능 향상을 위해 테이블 열에 사용하는 객체
  - USER_INDEXES
  - USER_IND_COLUMNS

실습 13-8 SCOTT 계정이 소유한 인덱스 정보 알아보기(SCOTT 계정일 때)

```
01  SELECT *
02    FROM USER_INDEXES;
```

:: 결과 화면

INDEX_NAME	INDEX_TYPE	TABLE_OWNER	TABLE_NAME	TABLE_TYPE	UNIQUENESS	COMPRESSION
PK_DEPT	NORMAL	SCOTT	DEPT	TABLE	UNIQUE	DISABLED
PK_EMP	NORMAL	SCOTT	EMP	TABLE	UNIQUE	DISABLED

실습 13-9 SCOTT 계정이 소유한 인덱스 컬럼 정보 알아보기(SCOTT 계정일 때)

```
01  SELECT *
02    FROM USER_IND_COLUMNS;
```

:: 결과 화면

INDEX_NAME	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	COLUMN_LENGTH	CHAR_LENGTH	DESCEND
PK_DEPT	DEPT	DEPTNO	1	22	0	ASC
PK_EMP	EMP	EMPNO	1	22	0	ASC

고유키는 열 데이터의 중복을 허용하지 않는 제약 조건(constraint)입니다.

13-2 더 빠른 검색을 위한 인덱스

인덱스 생성

```
CREATE INDEX 인덱스 이름
    ON 테이블 이름(열 이름1 ASC or DESC,
                   열 이름2 ASC or DESC,
                   ...                    );
```

실습 13-10 EMP 테이블의 SAL 열에 인덱스를 생성하기

```
01  CREATE INDEX IDX_EMP_SAL
02      ON EMP(SAL);
```

실습 13-11 생성된 인덱스 살펴보기(USER_IND_COLUMNS 사용)

```
01  SELECT * FROM USER_IND_COLUMNS;
```

:: 결과 화면(실습 13-11)

INDEX_NAME	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	COLUMN_LENGTH	CHAR_LENGTH	DESCEND
IDX_EMP_SAL	EMP	SAL	1	22	0	ASC
PK_DEPT	DEPT	DEPTNO	1	22	0	ASC
PK_EMP	EMP	EMPNO	1	22	0	ASC

13-2 더 빠른 검색을 위한 인덱스

인덱스 종류

방식	사용
단일 인덱스(single index)	CREATE INDEX IDX_NAME ON EMP(SAL);
복합 인덱스(concatenated index) 결합 인덱스(composite index) - 두 개 이상 열로 만들어지는 인덱스 - WHERE절의 두 열이 AND 연산으로 묶이는 경우	CREATE INDEX IDX_NAME ON EMP(SAL, ENAME, …);
고유 인덱스(unique index) - 열에 중복 데이터가 없을 때 사용 - UNIQUE 키워드를 지정하지 않으면 비고유 인덱스(non unique index)가 기본값	CREATE UNIQUE INDEX IDX_NAME ON EMP(EMPNO);
함수 기반 인덱스(function based index) - 열에 산술식 같은 데이터 가공이 진행된 결과로 인덱스 생성	CREATE INDEX IDX_NAME ON EMP(SAL*12 + COMM);
비트맵 인덱스(bitmap index) - 데이터 종류가 적고 같은 데이터가 많이 존재할 때 주로 사용	CREATE BITMAP INDEX IDX_NAME ON EMP(JOB);

13-2 더 빠른 검색을 위한 인덱스

- 인덱스 삭제
  - DROP

```
DROP INDEX 인덱스 이름;
```

기본 형식

실습 13-12 인덱스 삭제하기

```
01  DROP INDEX IDX_EMP_SAL;
```

실습 13-13 생성된 인덱스 살펴보기(USER_IND_COLUMNS 사용)

```
01  SELECT * FROM USER_IND_COLUMNS;
```

:: 결과 화면

INDEX_NAME	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	COLUMN_LENGTH	CHAR_LENGTH	DESCEND
PK_DEPT	DEPT	DEPTNO	1	22	0	ASC
PK_EMP	EMP	EMPNO	1	22	0	ASC

13-3 테이블처럼 사용하는 뷰

- 뷰란?
  - SELECT문을 저장한 객체

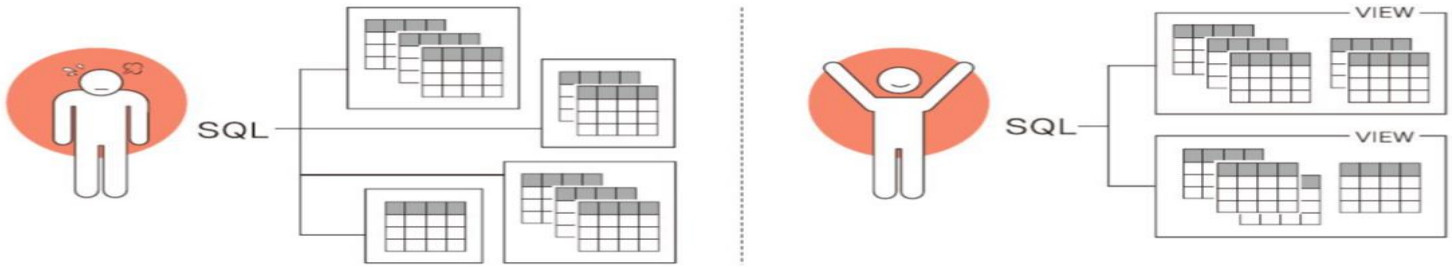
```
SELECT EMPNO, ENAME, JOB, DEPTNO
  FROM EMP
 WHERE DEPTNO = 20;
```

뷰	서브쿼리

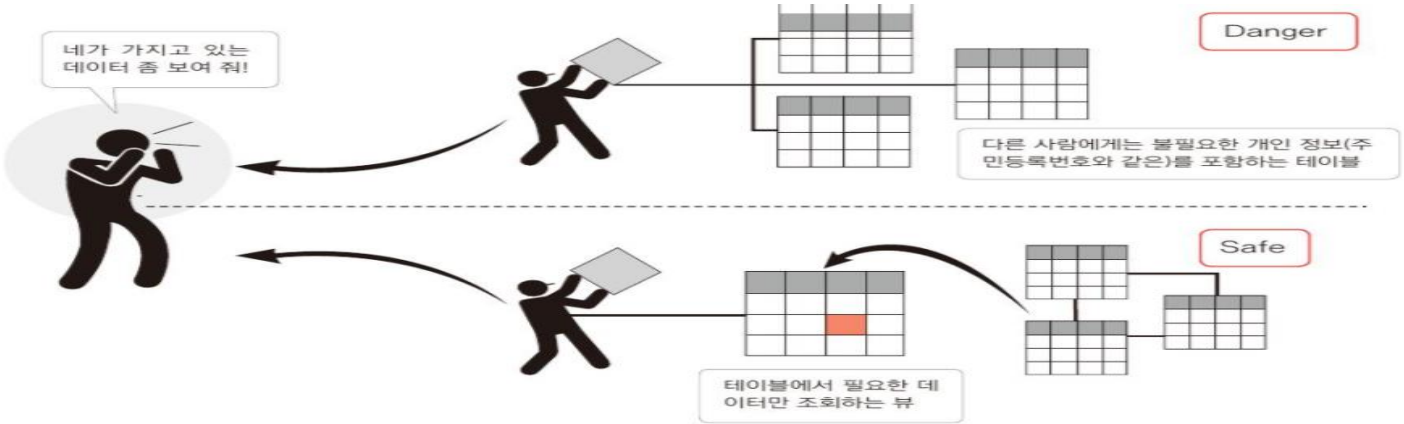
SELECT *
  FROM VW_EMP20;
	SELECT *
	  FROM(SELECT EMPNO, ENAME, JOB, DEPTNO
	  FROM EMP
	  WHERE DEPTNO = 20);

13-3 테이블처럼 사용하는 뷰

- 뷰의 사용 목적
  - 편의성 : SELECT문의 복잡도 완화



  - 보안성 : 테이블의 일부 데이터만 노출



13-3 테이블처럼 사용하는 뷰

뷰의 생성

```
CREATE [OR REPLACE] [FORCE | NOFORCE] VIEW 뷰 이름 (열 이름1, 열 이름2, …)
    AS (저장할SELECT문)
[WITH CHECK OPTION [CONSTRAINT 제약 조건]]
[WITH READ ONLY [CONSTRAINT 제약 조건]];
```

요소	설명
❶ OR REPLACE	같은 이름의 뷰가 이미 존재할 경우에 현재 생성할 뷰로 대체하여 생성(선택)
❷ FORCE	뷰가 저장할 SELECT문의 기반 테이블이 존재하지 않아도 강제로 생성(선택)
❸ NOFORCE	뷰가 저장할 SELECT문의 기반 테이블이 존재할 경우에만 생성(기본값)(선택)
❹ 뷰 이름	생성할 뷰 이름을 지정(필수)
❺ 열 이름	SELECT문에 명시된 이름 대신 사용할 열 이름 지정(생략 가능)(선택)
❻ 저장할 SELECT문	생성할 뷰에 저장할 SELECT문 지정(필수)
❼ WITH CHECK OPTION	지정한 제약 조건을 만족하는 데이터에 한해 DML 작업이 가능하도록 뷰 생성(선택)
❽ WITH READ ONLY	뷰의 열람, 즉 SELECT만 가능하도록 뷰 생성(선택)

13-3 테이블처럼 사용하는 뷰

뷰의 생성

실습 13-14 뷰를 생성하기 위해 계정 변경 접속하기(SQL*PLUS)

```
01  SQLPLUS SYSTEM/oracle

02  GRANT CREATE VIEW TO SCOTT;
```

:: 결과 화면

실습 13-15 뷰 생성하기(토드)

```
01  CREATE VIEW VW_EMP20
02     AS (SELECT EMPNO, ENAME, JOB, DEPTNO
03           FROM EMP
04          WHERE DEPTNO = 20);
```

실습 13-18 생성한 뷰 조회하기

```
01  SELECT *
02    FROM VW_EMP20;
```

:: 결과 화면

EMPNO	ENAME	JOB	DEPTNO
7369	SMITH	CLERK	20
7566	JONES	MANAGER	20
7788	SCOTT	ANALYST	20
7876	ADAMS	CLERK	20
7902	FORD	ANALYST	20

실습 13-16 생성한 뷰 확인하기(토드)

```
01  SELECT *
02    FROM USER_VIEWS;
```

:: 결과 화면

VIEW_NAME	TEXT_LENGTH	TEXT	TYPE_TEXT_LENGTH	TYPE_TEXT
VW_EMP20	80	(WIDEMEMO)		

1분 복습

부서 번호가 30인 사원 정보의 모든 열을 출력하는 VM_EMP30ALL 뷰를 작성하는 다음 SQL문의 빈칸을 채워 보세요.

```
CREATE [1]  VW_EMP30ALL
  AS (SELECT *
        FROM EMP
       WHERE [2]  );
```

13-3 테이블처럼 사용하는 뷰

뷰 삭제

DROP

실습 13-19 뷰 삭제하기

```
01  DROP VIEW VW_EMP20;
```

:: 결과 화면

VIEW_NAME	TEXT_LENGTH	TEXT	TYPE_TEXT_LENGTH	TYPE_TEXT	OID_TEXT_LENGTH

뷰는 실제 데이터가 아닌 SELECT문만 저장하므로 뷰를 삭제해도 테이블이나 데이터가 삭제되는 것은 아닙니다.

한 발 더 나가기! 뷰와 데이터 조작어

뷰는 SELECT문만 저장하는 객체이기 때문에 데이터 삽입·수정·삭제 같은 데이터 조작어 사용이 불가능할 것이라 생각할 수 있지만, 의외로 뷰에도 데이터 조작어를 직접 사용할 수 있는 경우가 있습니다. 하지만 뷰를 통한 테이블 데이터 조작이 가능하려면 여러 가지 조건을 만족해야 하고 테이블을 설계할 때 누락된 내용이 있으면 뷰를 통한 데이터 조작으로 인해 적합하지 않은 데이터가 생길 수도 있으므로 자주 사용하는 편은 아닙니다.

이는 뷰의 주 목적이 물리적 데이터를 저장하지 않고 SELECT문만 저장함으로써 테이블의 데이터를 열람하는 것이기 때문입니다. 이 책에서 다루지는 않지만 데이터를 따로 저장하는 것이 허용되는 구체화 뷰(materialized view)도 존재합니다.

13-3 테이블처럼 사용하는 뷰

인라인 뷰를 사용한 TOP-N SQL문

CREATE문을 통해 객체로 만들어지는 뷰 외에 SQL문에서 일회성으로 만들어서 사용하는 뷰를 인라인 뷰(inline view)라고 합니다. SELECT문에서 사용되는 서브쿼리, WITH절에서 미리 이름을 정의해 두고 사용하는 SELECT문 등이 이에 해당합니다.
이 인라인 뷰와 ROWNUM을 사용하면 ORDER BY절을 통해 정렬된 결과 중 최상위 몇 개 데이터만을 출력하는 것이 가능합니다. ROWNUM을 알아보기 위해 다음 SELECT문을 실행하여 결과를 확인해 보세요.

실습 13-20 ROWNUM을 추가로 조회하기

```
01  SELECT ROWNUM, E.*
02    FROM EMP E;
```

:: 결과 화면

ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7369	SMITH	CLERK	7902	1980/12/17	800		20
2	7499	ALLEN	SALESMAN	7698	1981/02/20	1600	300	30
3	7521	WARD	SALESMAN	7698	1981/02/22	1250	500	30

실습 13-21 EMP 테이블을 SAL 열 기준으로 정렬하기

```
01  SELECT ROWNUM, E.*
02    FROM EMP E
03  ORDER BY SAL DESC;
```

:: 결과 화면

ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
9	7839	KING	PRESIDENT		1981/11/17	5000		10
13	7902	FORD	ANALYST	7566	1981/12/03	3000		20
8	7788	SCOTT	ANALYST	7566	1987/04/19	3000		20

ROWNUM은 의사 열(pseudo column)이라고 하는 특수 열입니다. 의사 열은 데이터가 저장되는 실제 테이블에 존재하지는 않지만 특정 목적을 위해 테이블에 저장되어 있는 열처럼 사용 가능한 열을 뜻합니다.

ROWNUM 외에 인덱스와 밀접하게 연관된 ROWID도 대표적인 의사 열입니다.

13-3 테이블처럼 사용하는 뷰

실습 13-22 인라인 뷰(서브쿼리 사용)

```
01  SELECT ROWNUM, E.*
02    FROM (SELECT *
03            FROM EMP E
04           ORDER BY SAL DESC) E;
```

실습 13-23 인라인 뷰(WITH절 사용)

```
01  WITH E AS (SELECT * FROM EMP ORDER BY SAL DESC)
02  SELECT ROWNUM, E.*
03    FROM E;
```

:: 결과 화면(실습 13-22, 13-23의 실행 결과가 동일)

ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7839	KING	PRESIDENT		1981/11/17	5000		10
2	7902	FORD	ANALYST	7566	1981/12/03	3000		20
3	7788	SCOTT	ANALYST	7566	1987/04/19	3000		20
4	7566	JONES	MANAGER	7839	1981/04/02	2975		20

13-3 테이블처럼 사용하는 뷰

실습 13-24 인라인 뷰로 TOP-N 추출하기(서브쿼리 사용)

```
01  SELECT ROWNUM, E.*
02    FROM (SELECT *
03            FROM EMP E
04           ORDER BY SAL DESC) E
05   WHERE ROWNUM <= 3;
```

실습 13-25 인라인 뷰로 TOP-N 추출하기(WITH절 사용)

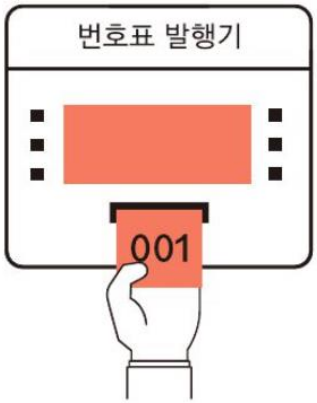
```
01  WITH E AS (SELECT * FROM EMP ORDER BY SAL DESC)
02  SELECT ROWNUM, E.*
03    FROM E
04   WHERE ROWNUM <= 3;
```

:: 결과 화면(실습 13-24, 13-25의 실행 결과가 동일)

ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7839	KING	PRESIDENT		1981/11/17	5000		10
2	7788	SCOTT	ANALYST	7566	1987/04/19	3000		20
3	7902	FORD	ANALYST	7566	1981/12/03	3000		20

13-4 규칙에 따라 순번을 생성하는 시퀀스

- 시퀀스란? 오라클 데이터베이스에서 특정 규칙에 맞는 연속 숫자를 생성하는 객체
  - 번호 생성기



13-4 규칙에 따라 순번을 생성하는 시퀀스

시퀀스 생성

```
CREATE SEQUENCE 시퀀스 이름 -1
[INCREMENT BY n] -2
[START WITH n] -3
[MAXVALUE n | NOMAXVALUE] -4
[MINVALUE n | NOMINVALUE] -5
[CYCLE | NOCYCLE] -6
[CACHE n | NOCACHE] -7
```

번호	설명
1	생성할 시퀀스 이름 지정. 아래 절(2 ~ 7)들을 지정하지 않았을 경우 1부터 시작하여 1만큼 계속 증가하는 시퀀스가 생성(필수)
2	시퀀스에서 생성할 번호의 증가 값(기본값은 1)(선택)
3	시퀀스에서 생성할 번호의 시작 값(기본값은 1)(선택)
4	시퀀스에서 생성할 번호의 최댓값 지정, 최댓값은 시작 값(START WITH) 이상, 최솟값(MINVALUE)을 초과값으로 지정. NOMAXVALUE로 지정하였을 경우 오름차순이면 10^{27}, 내림차순일 경우 -1로 설정(선택)
5	시퀀스에서 생성할 번호의 최솟값 지정, 최솟값은 시작 값(START WITH) 이하, 최댓값(MAXVALUE) 미만 값으로 지정. NOMINVALUE로 지정하였을 경우 오름차순이면 1, 내림차순이면 10^{-26}으로 설정(선택)
6	시퀀스에서 생성한 번호가 최댓값(MAXVALUE)에 도달했을 경우 CYCLE이면 시작 값(START WITH)에서 다시 시작, NOCYCLE이면 번호 생성이 중단되고, 추가 번호 생성을 요청하면 오류 발생(선택)
7	시퀀스가 생성할 번호를 메모리에 미리 할당해 놓은 수를 지정, NOCACHE는 미리 생성하지 않도록 설정. 옵션을 모두 생략하면 기본값은 20(선택)

13-4 규칙에 따라 순번을 생성하는 시퀀스

- 시퀀스 사용
  - CURRVAL
    - 시퀀스 마지막 생성 번호
  - NEXTVAL
    - 다음 번호 생성
- 시퀀스 수정 : ALTER
- 시퀀스 삭제 : DROP

13-4 규칙에 따라 순번을 생성하는 시퀀스

- 시퀀스 생성
  - CURRVAL
    - 시퀀스 마지막 생성 번호
  - NEXTVAL
    - 다음 번호 생성

실습 13-26 DEPT 테이블을 사용하여 DEPT_SEQUENCE 테이블 생성하기

```
01  CREATE TABLE DEPT_SEQUENCE
02      AS SELECT *
03           FROM DEPT
04          WHERE 1 <> 1;

05  SELECT * FROM DEPT_SEQUENCE;
```

실습 13-27 시퀀스 생성하기

```
01  CREATE SEQUENCE SEQ_DEPT_SEQUENCE
02   INCREMENT BY 10
03   START WITH 10
04   MAXVALUE 90
05   MINVALUE 0
06   NOCYCLE
07   CACHE 2;
```

실습 13-28 생성한 시퀀스 확인하기

```
01  SELECT *
02    FROM USER_SEQUENCES;
```

:: 결과 화면(실습 13-28 결과)

SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	CYCLE_FLAG	ORDER_FLAG	CACHE_SIZE	LAST_NUMBER
SEQ_DEPT_SEQUENCE	10	90	10	N	N	2	10

13-4 규칙에 따라 순번을 생성하는 시퀀스

시퀀스 사용

실습 13-29 시퀀스에서 생성한 순번을 사용한 INSERT문 실행하기

```
01  INSERT INTO DEPT_SEQUENCE (DEPTNO, DNAME, LOC)
02  VALUES (SEQ_DEPT_SEQUENCE.NEXTVAL, 'DATABASE', 'SEOUL');

03  SELECT * FROM DEPT_SEQUENCE ORDER BY DEPTNO;
```

:: 결과 화면

DEPTNO	DNAME	LOC
10	DATABASE	SEOUL

실습 13-30 가장 마지막으로 생성된 시퀀스 확인하기

```
01  SELECT SEQ_DEPT_SEQUENCE.CURRVAL
02    FROM DUAL;
```

:: 결과 화면

CURRVAL
10

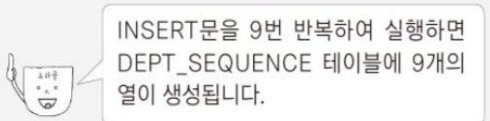
실습 13-31 시퀀스에서 생성한 순번을 반복 사용하여 INSERT문 실행하기

```
01  INSERT INTO DEPT_SEQUENCE (DEPTNO, DNAME, LOC)
02  VALUES (SEQ_DEPT_SEQUENCE.NEXTVAL, 'DATABASE', 'SEOUL');

03  SELECT * FROM DEPT_SEQUENCE ORDER BY DEPTNO;
```

:: 결과 화면

DEPTNO	DNAME	LOC
10	DATABASE	SEOUL
20	DATABASE	SEOUL
30	DATABASE	SEOUL
40	DATABASE	SEOUL
50	DATABASE	SEOUL
60	DATABASE	SEOUL
70	DATABASE	SEOUL
80	DATABASE	SEOUL
90	DATABASE	SEOUL



시퀀스 최댓값 90에 도달 후 다시 INSERT문을 실행하면 시퀀스는 번호를 더 생성하지 못합니다.

13-4 규칙에 따라 순번을 생성하는 시퀀스

시퀀스 수정

기본 형식

```
ALTER SEQUENCE 시퀀스 이름
[INCREMENT BY n]
[MAXVALUE n ¦ NOMAXVALUE]
[MINVALUE n ¦ NOMINVALUE]
[CYCLE ¦ NOCYCLE]
[CACHE n ¦ NOCACHE]
```

실습 13-32 시퀀스 옵션 수정하기

```
01  ALTER SEQUENCE SEQ_DEPT_SEQUENCE
02   INCREMENT BY 3
03   MAXVALUE 99
04   CYCLE;
```

실습 13-33 옵션을 수정한 시퀀스 조회하기

```
01  SELECT *
02    FROM USER_SEQUENCES;
```

:: 결과 화면(실습 13-33 결과)

SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	CYCLE_FLAG	ORDER_FLAG	CACHE_SIZE	LAST_NUMBER
SEQ_DEPT_SEQUENCE	0	99	3	Y	N	2	93

실습 13-34 수정한 시퀀스를 사용하여 INSERT문 실행하기

```
01  INSERT INTO DEPT_SEQUENCE (DEPTNO, DNAME, LOC)
02  VALUES (SEQ_DEPT_SEQUENCE.NEXTVAL, 'DATABASE', 'SEOUL');

03  SELECT * FROM DEPT_SEQUENCE ORDER BY DEPTNO;
```

:: 결과 화면

DEPTNO	DNAME	LOC
10	DATABASE	SEOUL
20	DATABASE	SEOUL
30	DATABASE	SEOUL
40	DATABASE	SEOUL
50	DATABASE	SEOUL
60	DATABASE	SEOUL
70	DATABASE	SEOUL
80	DATABASE	SEOUL
90	DATABASE	SEOUL
93	DATABASE	SEOUL

수정된 시퀀스의 증가 값 때문에 새로 추가

13-5 공식 별칭을 지정하는 동의어

- 동의어란?
  - 객체 이름 대신 사용할 수 있는 다른 이름

```
CREATE [PUBLIC] SYNONYM 동의어 이름
       (1)              (2)
FOR [사용자.][객체 이름];
     (3)     (4)
```

동의어는 SELECT문의 SELECT절, FROM절에서 사용한 열 또는 테이블 별칭과 유사하지만, 오라클 데이터베이스에 저장되는 객체이기 때문에 일회성이 아니라는 점에서 차이가 납니다. 동의어 생성 역시 권한을 따로 부여해야 하기 때문에 다음과 같이 SQL*PLUS에서 SYSTEM에 접속하여 SCOTT 계정에 동의어 생성 권한을 부여해 보세요. PUBLIC SYNONYM 권한도 따로 부여해 주어야 합니다.

요소	설명
① PUBLIC	동의어를 데이터베이스 내 모든 사용자가 사용할 수 있도록 설정. 생략할 경우 동의어를 생성한 사용자만 사용 가능(PUBLIC으로 생성되어도 본래 객체의 사용 권한이 있어야 사용 가능)(선택)
② 동의어 이름	생성할 동의어 이름(필수)
③ 사용자.	생성할 동의어의 본래 객체 소유 사용자를 지정. 생략할 경우 현재 접속한 사용자로 지정(선택)
④ 객체이름	동의어를 생성할 대상 객체 이름(필수)

13-5 공식 별칭을 지정하는 동의어

- 동의어 생성
  - 객체 이름 대신 사용할 수 있는 다른 이름

실습 13-38 EMP 테이블의 동의어 생성하기

```
01  CREATE SYNONYM E
02      FOR EMP;
```

실습 13-39 E 테이블 전체 내용 조회하기

```
01  SELECT * FROM E;
```

:: 결과 화면

EMPNO	EN...	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	1980/12/17	800		20
7499	ALLEN	SALESMAN	7698	1981/02/20	1600	300	30
7521	WARD	SALESMAN	7698	1981/02/22	1250	500	30
7566	JONES	MANAGER	7839	1981/04/02	2975		20
7654	MARTIN	SALESMAN	7698	1981/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	1981/05/01	2850		30
7782	CLARK	MANAGER	7839	1981/06/09	2450		10
7788	SCOTT	ANALYST	7566	1987/04/19	3000		20
7839	KING	PRESIDENT		1981/11/17	5000		10
7844	TURNER	SALESMAN	7698	1981/09/08	1500	0	30
7876	ADAMS	CLERK	7788	1987/05/23	1100		20
7900	JAMES	CLERK	7698	1981/12/03	950		30
7902	FORD	ANALYST	7566	1981/12/03	3000		20
7934	MILLER	CLERK	7782	1982/01/23	1300		10

13-5 공식 별칭을 지정하는 동의어

- 동의어 삭제 : DROP

실습 13-40 동의어 삭제하기

```
01  DROP SYNONYM E;
```

동의어를 삭제하면 E 동의어로 SELECT를 할 수 없지만 EMP 테이블 이름과 데이터에는 아무 영향을 주지 않습니다.