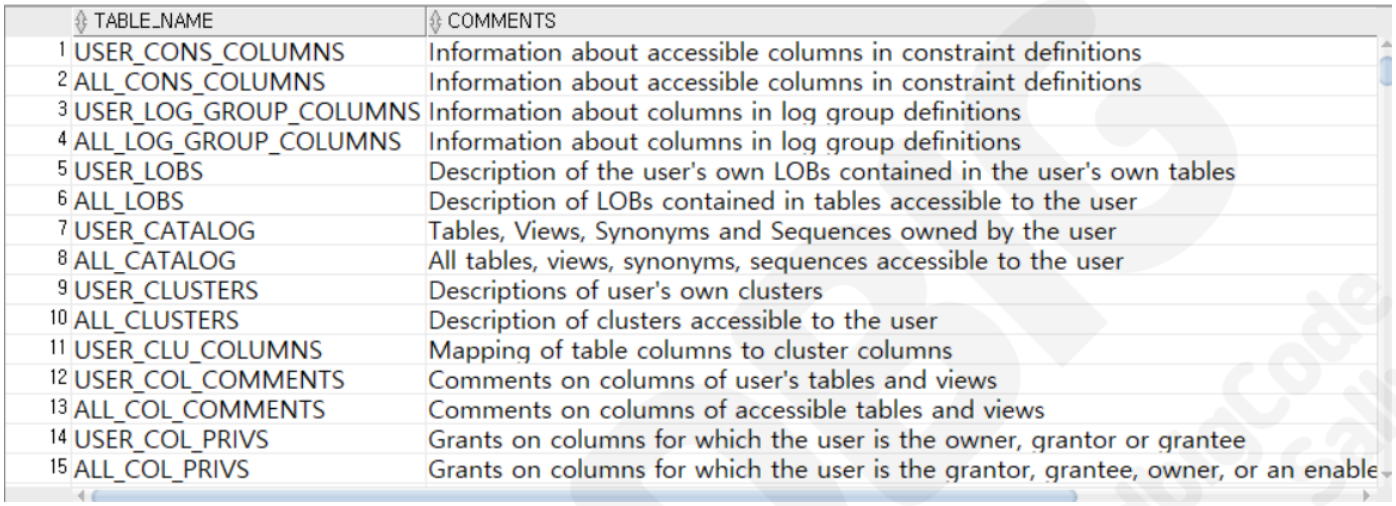
■STEP1. BASIC

Q001

- DICT을 이용하여 SCOTT계정에서 사용가능한 데이터 사전을 살펴보시오

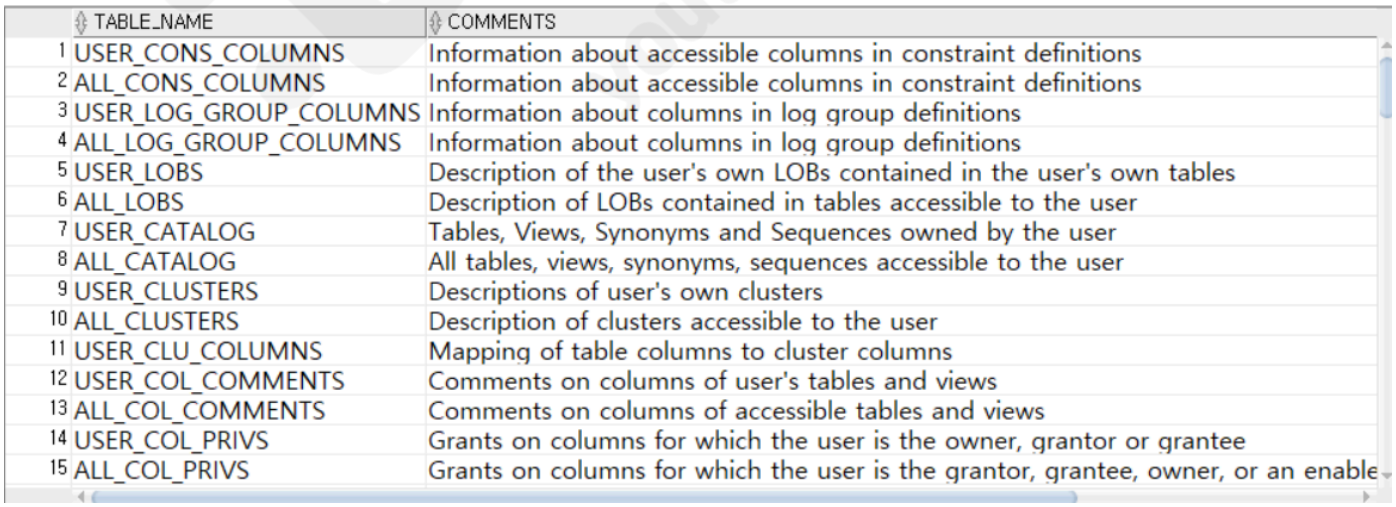


	TABLE_NAME	COMMENTS
1	USER_CONS_COLUMNS	Information about accessible columns in constraint definitions
2	ALL_CONS_COLUMNS	Information about accessible columns in constraint definitions
3	USER_LOG_GROUP_COLUMNS	Information about columns in log group definitions
4	ALL_LOG_GROUP_COLUMNS	Information about columns in log group definitions
5	USER_LOBS	Description of the user's own LOBs contained in the user's own tables
6	ALL_LOBS	Description of LOBs contained in tables accessible to the user
7	USER_CATALOG	Tables, Views, Synonyms and Sequences owned by the user
8	ALL_CATALOG	All tables, views, synonyms, sequences accessible to the user
9	USER_CLUSTERS	Descriptions of user's own clusters
10	ALL_CLUSTERS	Description of clusters accessible to the user
11	USER_CLU_COLUMNS	Mapping of table columns to cluster columns
12	USER_COL_COMMENTS	Comments on columns of user's tables and views
13	ALL_COL_COMMENTS	Comments on columns of accessible tables and views
14	USER_COL_PRIVS	Grants on columns for which the user is the owner, grantor or grantee
15	ALL_COL_PRIVS	Grants on columns for which the user is the grantor, grantee, owner, or an enable

```
SELECT * FROM DICT;
```

Q002

- DICTIONARY을 이용하여 SCOTT계정에서 사용가능한 데이터 사전을 살펴보시오

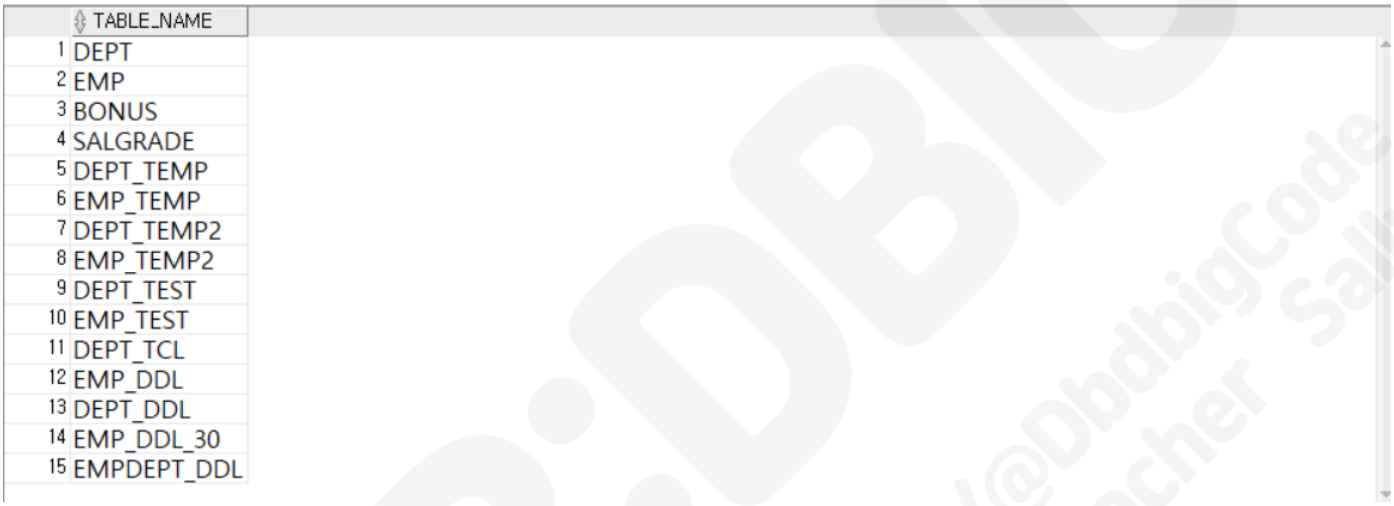


	TABLE_NAME	COMMENTS
1	USER_CONS_COLUMNS	Information about accessible columns in constraint definitions
2	ALL_CONS_COLUMNS	Information about accessible columns in constraint definitions
3	USER_LOG_GROUP_COLUMNS	Information about columns in log group definitions
4	ALL_LOG_GROUP_COLUMNS	Information about columns in log group definitions
5	USER_LOBS	Description of the user's own LOBs contained in the user's own tables
6	ALL_LOBS	Description of LOBs contained in tables accessible to the user
7	USER_CATALOG	Tables, Views, Synonyms and Sequences owned by the user
8	ALL_CATALOG	All tables, views, synonyms, sequences accessible to the user
9	USER_CLUSTERS	Descriptions of user's own clusters
10	ALL_CLUSTERS	Description of clusters accessible to the user
11	USER_CLU_COLUMNS	Mapping of table columns to cluster columns
12	USER_COL_COMMENTS	Comments on columns of user's tables and views
13	ALL_COL_COMMENTS	Comments on columns of accessible tables and views
14	USER_COL_PRIVS	Grants on columns for which the user is the owner, grantor or grantee
15	ALL_COL_PRIVS	Grants on columns for which the user is the grantor, grantee, owner, or an enable

```
SELECT * FROM DICTIONARY;
```

Q003

- USER접두어를 가진 데이터 사전 : 현재 오라클에 접속해 있는 사용자가 소유한 객체 정보가 보관되어 있음.
- SCOTT계정이 가지고 있는 객체 정보 살펴보기
- SCOTT계정이 가지고 있는 테이블 이름 알고 싶을때 유용함.

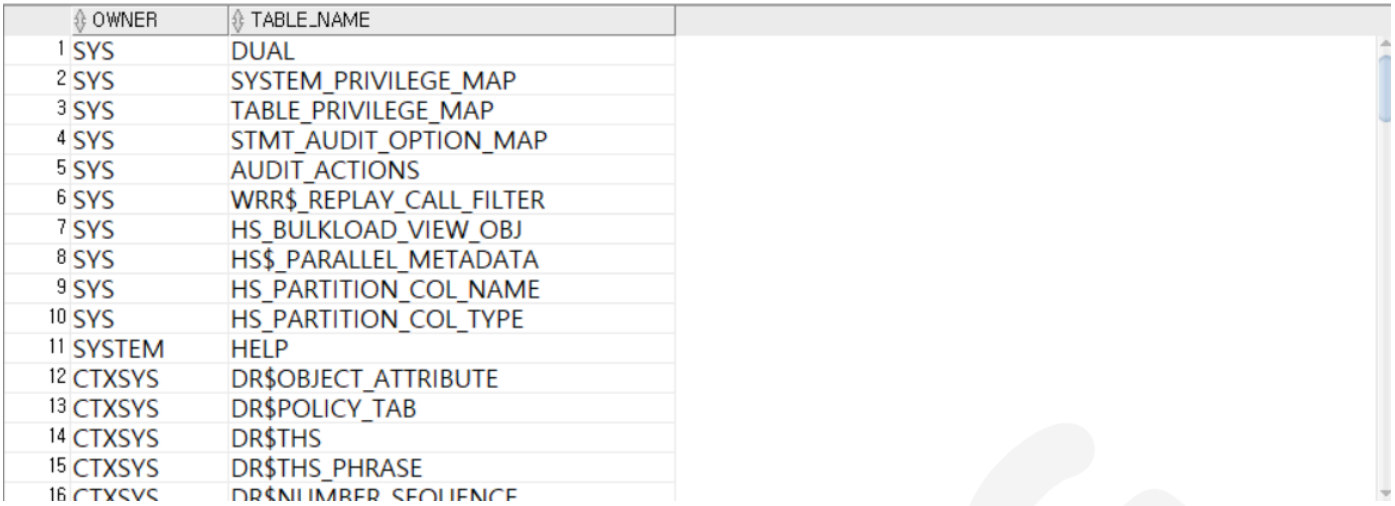


	TABLE_NAME
1	DEPT
2	EMP
3	BONUS
4	SALGRADE
5	DEPT_TEMP
6	EMP_TEMP
7	DEPT_TEMP2
8	EMP_TEMP2
9	DEPT_TEST
10	EMP_TEST
11	DEPT_TCL
12	EMP_DDL
13	DEPT_DDL
14	EMP_DDL_30
15	EMPDEPT_DDL

```
SELECT TABLE_NAME
  FROM USER_TABLES;
```

Q004

- ALL 접두어를 가진 데이터 사전 : 오라클데이터베이스에 접속해 있는 사용자가 소유한 객체 및 다른 소유자가 소유한 객체중 사용이 허락되어 있는 객체 정보
- 사용가능한 모든 테이블이 출력됨.



	OWNER	TABLE_NAME
1	SYS	DUAL
2	SYS	SYSTEM_PRIVILEGE_MAP
3	SYS	TABLE_PRIVILEGE_MAP
4	SYS	STMT_AUDIT_OPTION_MAP
5	SYS	AUDIT_ACTIONS
6	SYS	WRR$_REPLAY_CALL_FILTER
7	SYS	HS_BULKLOAD_VIEW_OBJ
8	SYS	HS$_PARALLEL_METADATA
9	SYS	HS_PARTITION_COL_NAME
10	SYS	HS_PARTITION_COL_TYPE
11	SYSTEM	HELP
12	CTXSYS	DR$OBJECT_ATTRIBUTE
13	CTXSYS	DR$POLICY_TAB
14	CTXSYS	DR$THS
15	CTXSYS	DR$THS_PHRASE
16	CTXSYS	DR$NUMBER_SEQUENCE

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

Q005

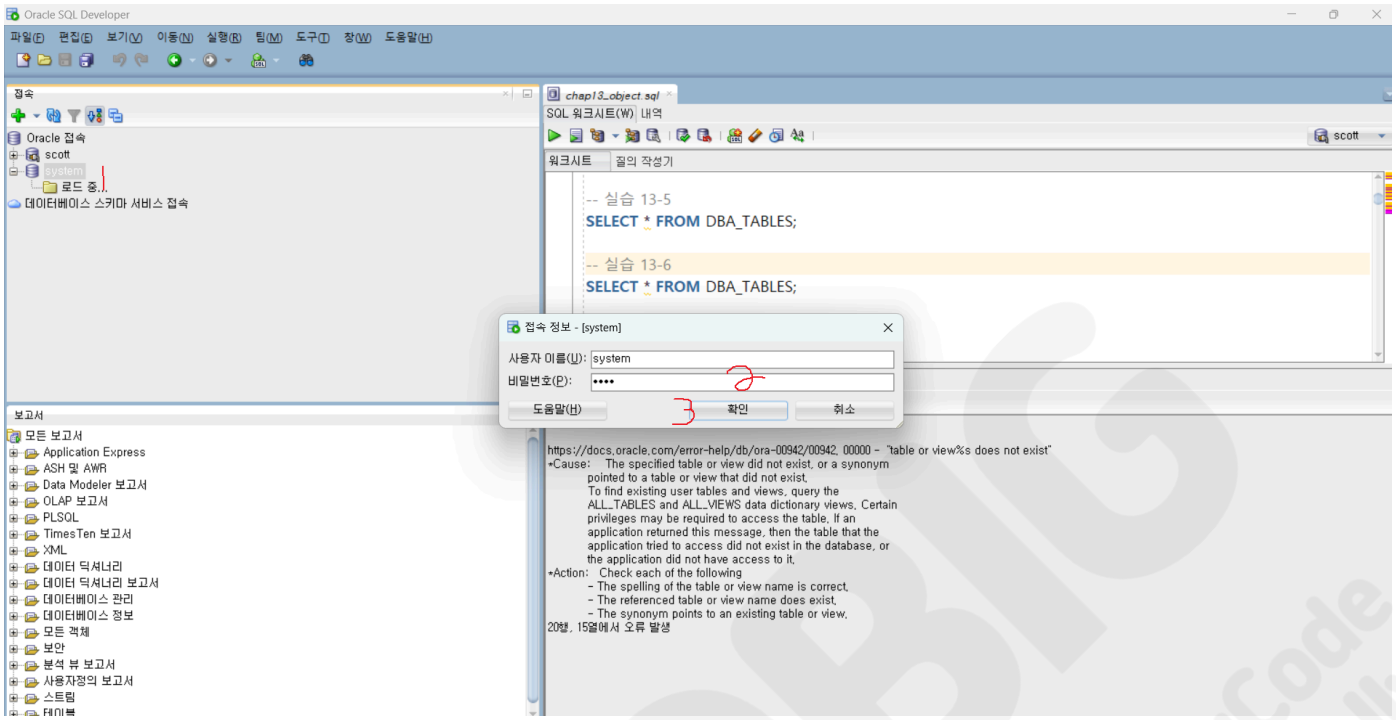
- 데이터베이스 관리 권한을 가진 사용자만 조회할 수 있는 테이블
- SCOTT계정으로는 조회가 불가능함.

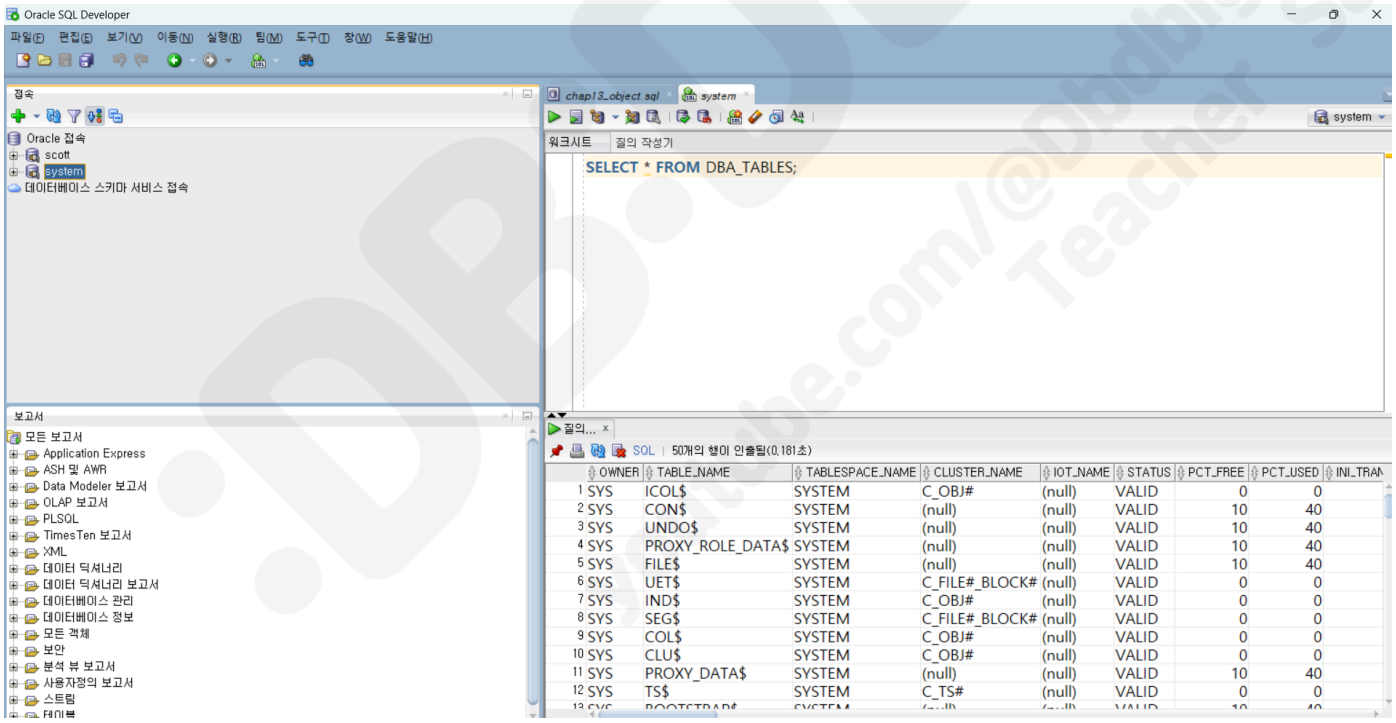
```
ORA-00942: table or view does not exist

https://docs.oracle.com/error-help/db/ora-00942/00942, 00000 - "table or view%s does not exist"
*Cause:    The specified table or view did not exist, or a synonym
           pointed to a table or view that did not exist.
           To find existing user tables and views, query the
           ALL_TABLES and ALL_VIEWS data dictionary views. Certain
           privileges may be required to access the table. If an
           application returned this message, then the table that the
           application tried to access did not exist in the database, or
           the application did not have access to it.
*Action:   Check each of the following
           - The spelling of the table or view name is correct.
           - The referenced table or view name does exist.
           - The synonym points to an existing table or view.
16행, 15열에서 오류 발생
```

```
SELECT * FROM DBA_TABLES;
```

Q006

- (SYSTEM) SYSTEM 계정으로 DBA_ 접두어 사용하기






```
SELECT * FROM DBA_TABLES;
```

Q007

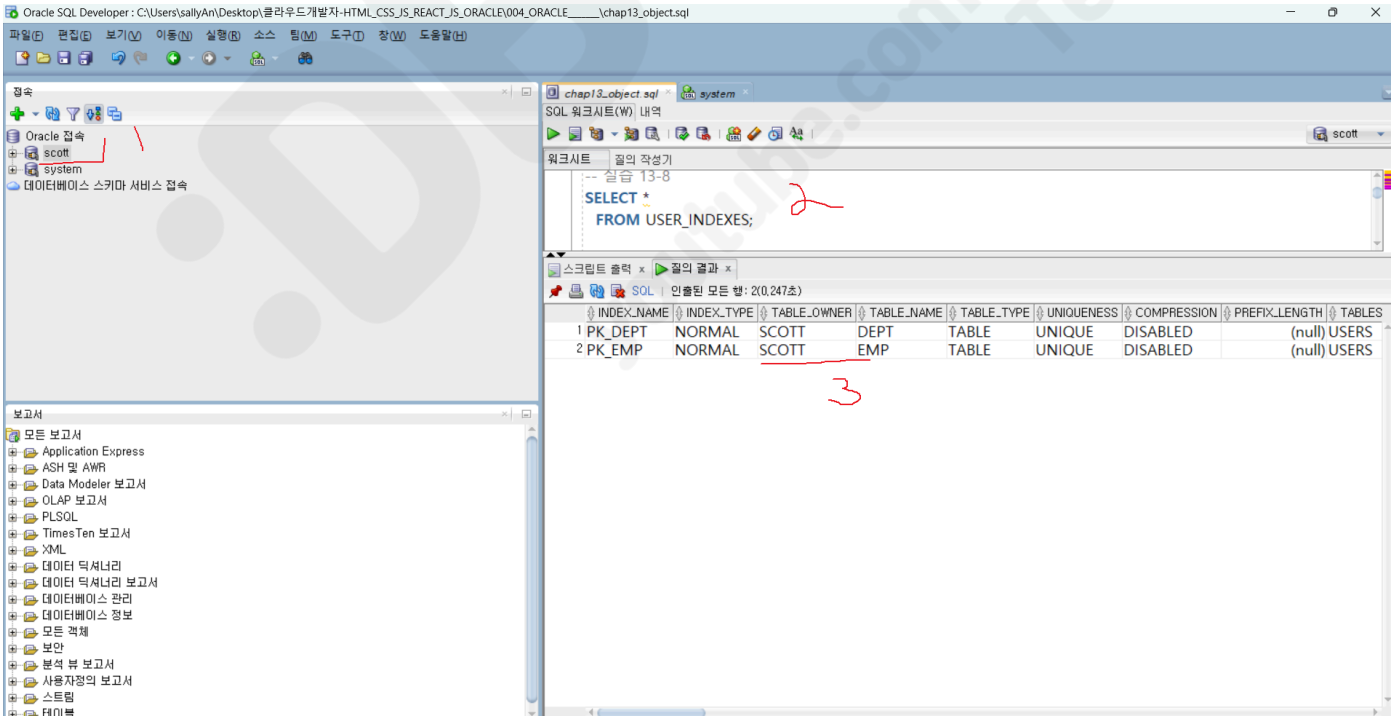
- (SYSTEM) DBA_USERS를 사용하여 사용자 정보 알아보기

	USERNAME	USER_ID	PASSWORD	ACCOUNT_STATUS	LOCK_DATE	EXPIRY_DATE	DEFAULT_TABLESPACE	TEMPORARY_TABLESP.
1	SCOTT	48 (null)		OPEN	(null)	25/06/19	USERS	TEMP

```
SELECT *
  FROM DBA_USERS
 WHERE USERNAME = 'SCOTT';
```

Q008

- 더 빠른 검색을 위한 인덱스
- (SCOTT계정) SCOTT계정이 소유한 인덱스 정보 알아보기



```sql
SELECT *
  FROM USER_INDEXES;
```

Q009

- (SCOTT계정) SCOTT계정이 소유한 인덱스 컬럼 정보 알아보기

	INDEX_NAME	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	CC
1	BIN$VPkj0okTR7aHBXnh7lBD5g==$0	BIN$9ZcWVQGGQoiox3RfVX0Ahw==$0	EMPNO	1	
2	PK_DEPT	DEPT	DEPTNO	1	
3	PK_EMP	EMP	EMPNO	1	

```sql
SELECT *
  FROM USER_IND_COLUMNS;
```

Q010

- EMP 테이블의 SAL열에 인덱스 생성하기

```
Index IDX_EMP_SAL이(가) 생성되었습니다.
```

```sql
CREATE INDEX IDX_EMP_SAL
    ON EMP(SAL);
```

Q011

- 생성된 인덱스 살펴보기

	INDEX_NAME	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	CC
1	BIN$VPkj0okTR7aHBXnh7lBD5g==$0	BIN$9ZcWVQGGQoiox3RfVX0Ahw==$0	EMPNO	1	
2	IDX_EMP_SAL	EMP	SAL	1	
3	PK_DEPT	DEPT	DEPTNO	1	
4	PK_EMP	EMP	EMPNO	1	

```sql
SELECT * FROM USER_IND_COLUMNS;
```

Q012

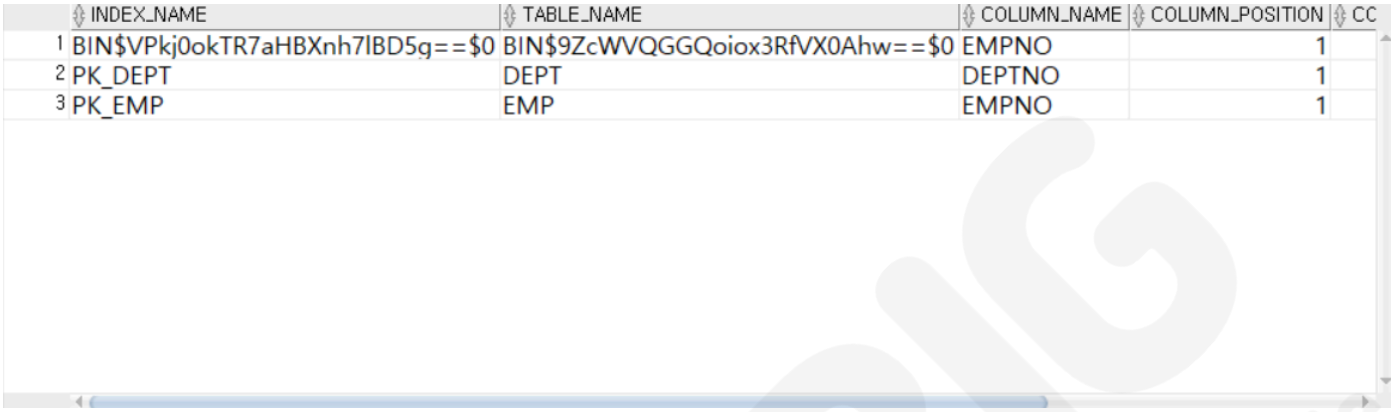
- 인덱스 삭제하기기

```
Index IDX_EMP_SAL이(가) 생성되었습니다.

Index IDX_EMP_SAL이(가) 삭제되었습니다.
```

```sql
DROP INDEX IDX_EMP_SAL;
```

Q013

- 생성된 인덱스 살펴보기



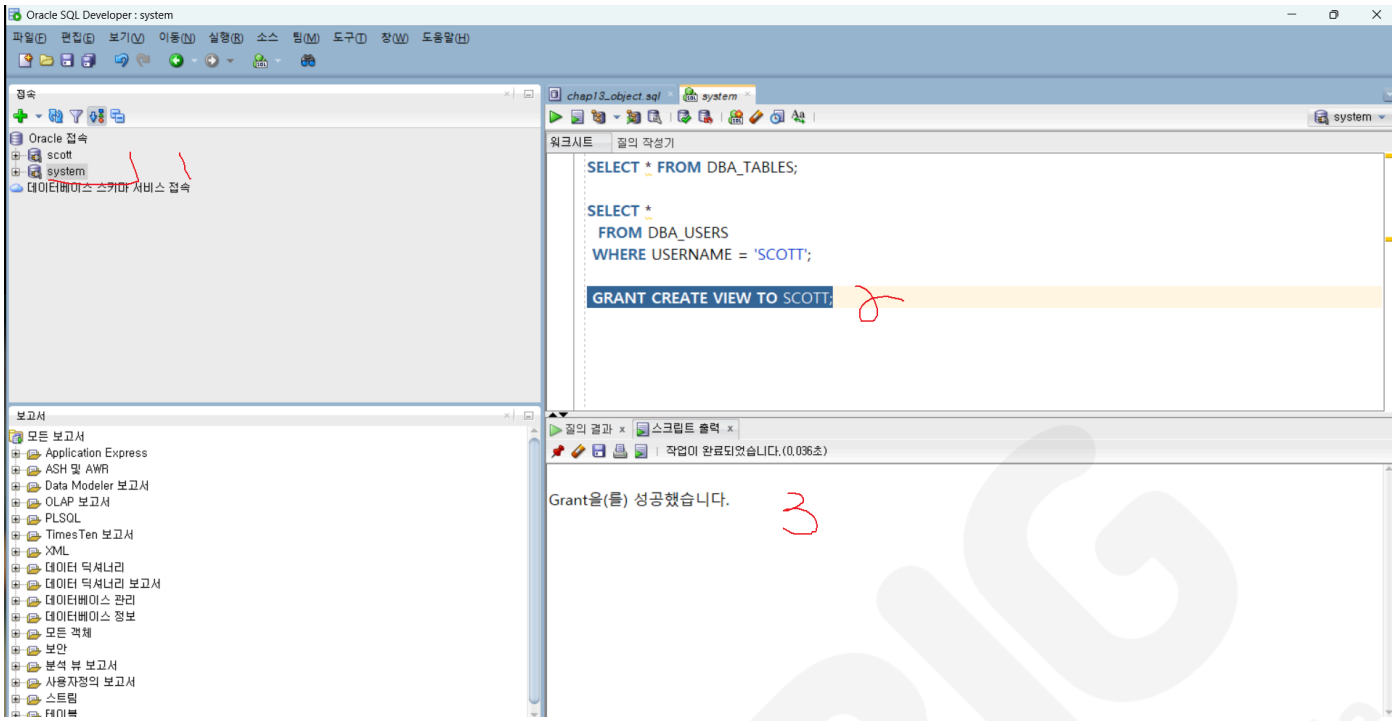
	INDEX_NAME	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	CC
1	BIN$VPkj0okTR7aHBXnh7lBD5g==$0	BIN$9ZcWVQGGQoiox3RfVX0Ahw==$0	EMPNO	1	
2	PK_DEPT	DEPT	DEPTNO	1	
3	PK_EMP	EMP	EMPNO	1	

```
SELECT * FROM USER_IND_COLUMNS;
```

Q014

- 테이블처럼 사용하는 뷰
- 뷰를 SELECT문의FROM 절에 사용하면 특정데이터를 조회하는 것과 같은 효과를 얻을 수 있음.
- 편리성 : SELECT 문의 복잡도를 완화하기 위해
- 보완성 : 테이블의 특정열을 노출하고 싶지 않을때

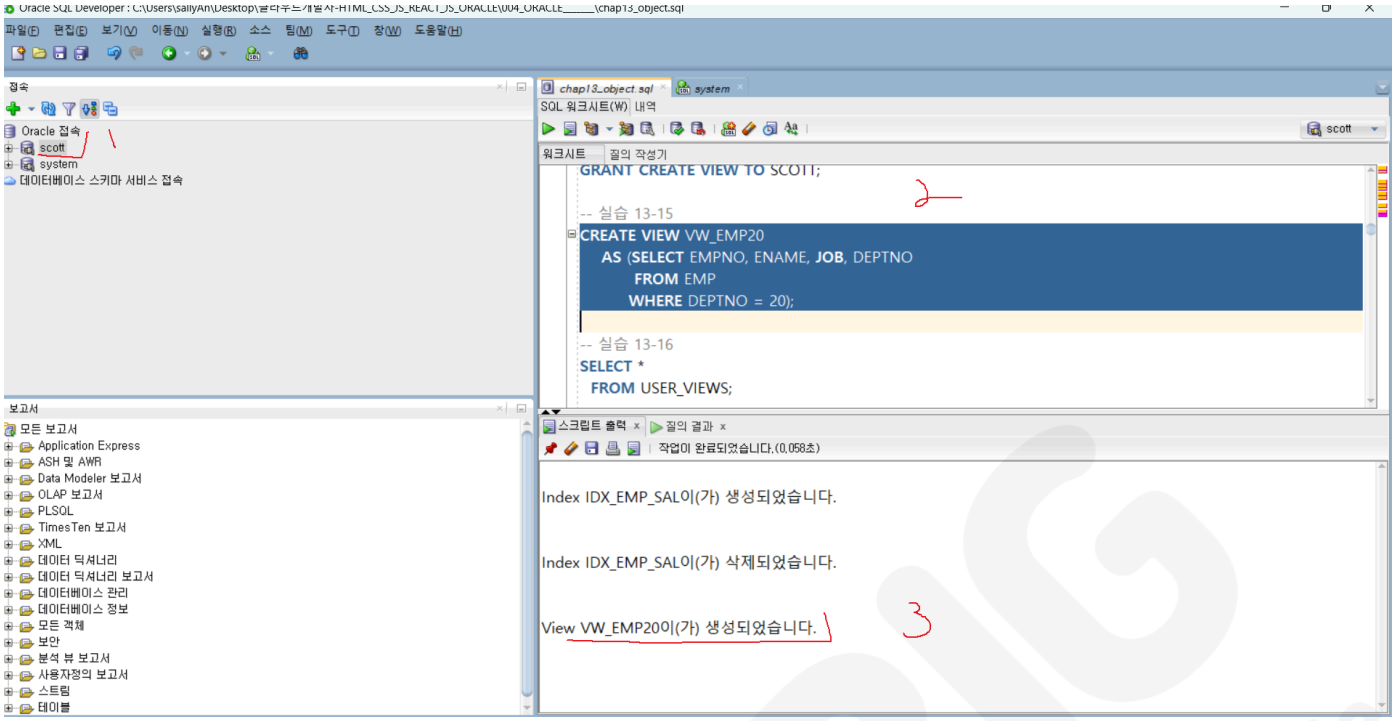
1. 뷰를 생성하기위해 SYSTEM으로 계정변경
2. SCOTT에 VIEW 생성권한 주기




```
SQLPLUS SYSTEM/1234

GRANT CREATE VIEW TO SCOTT;
```

Q015

- SCOTT 계정으로 VIEW 생성하기
- EMP 테이블에서 DEPTNO가 20인 EMPNO, ENAME, JOB, DEPTNO 열의 데이터로 VW_EMP20 라는 VIEW를 생성하시오오




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

Q016

- VIEW가 잘 만들어 졌는지 USER_VIEWS에서서 조회하시오.

	VIEW_NAME	TEXT_LENGTH	TEXT
1	VW_EMP20	80	(SELECT EMPNO, ENAME, JOB, DEPTNO FROM EMP WHERE DEPTNO = 2(

```
SELECT *
  FROM USER_VIEWS;
```

Q017

- (SCOTT계정) 생성한 뷰의 내용을 확인하시오.

	VIEW_NAME	TEXT_LENGTH	TEXT
1	VW_EMP20	80	(SELECT EMPNO, ENAME, JOB, DEPTNO FROM EMP WHERE DEPTNO = 2(

```
SELECT VIEW_NAME, TEXT_LENGTH, TEXT
  FROM USER_VIEWS;
```

Q018

- VW_EMP20 의 생성한 뷰를 조회하시오.

	EMPNO	ENAME	JOB	DEPTNO
1	7566	JONES	MANAGER	20
2	7902	FORD	ANALYST	20
3	7369	SMITH	CLERK	20
4	7788	SCOTT	ANALYST	20
5	7876	ADAMS	CLERK	20

```
SELECT *
  FROM VW_EMP20;
```

Q019

- VW_EMP20 뷰를 삭제 하시오.

View VW_EMP20이(가) 생성되었습니다.

View VW_EMP20이(가) 삭제되었습니다.

```
DROP VIEW VW_EMP20;
```

Q020

- 인라인뷰 : 일회성으로 만들어서 사용하는 뷰
- SELECT에서 사용되는 서브쿼리, WITH절에서 미리 이름을 정의해사용하는 SELECT문이 해당됨.
- ROWNUM을 이용하면 열의 데이터가 숫자로 출력되는 것을 확인 할수 있음.

	ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	1	7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
2	2	7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
3	3	7782	CLARK	MANAGER	7839	81/05/09	2450	(null)	10
4	4	7566	JONES	MANAGER	7839	81/04/01	2975	(null)	20
5	5	7654	MARTIN	SALESMAN	7698	81/09/10	1250	1400	30
6	6	7499	ALLEN	SALESMAN	7698	81/02/11	1600	300	30
7	7	7844	TURNER	SALESMAN	7698	81/08/21	1500	0	30
8	8	7900	JAMES	CLERK	7698	81/12/11	950	(null)	30
9	9	7521	WARD	SALESMAN	7698	81/02/23	1250	500	30
10	10	7902	FORD	ANALYST	7566	81/12/11	3000	(null)	20
11	11	7369	SMITH	CLERK	7902	80/12/09	800	(null)	20
12	12	7788	SCOTT	ANALYST	7566	82/12/22	3000	(null)	20
13	13	7876	ADAMS	CLERK	7788	83/01/15	1100	(null)	20
14	14	7934	MILLER	CLERK	7782	82/01/11	1300	(null)	10

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

Q021

- EMP 테이블을 SAL열 기준 내림차순으로 정렬 및 ROWNUM의 열 데이터 번호를 이용하여 조회하시오.

	ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	1	7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
2	12	7788	SCOTT	ANALYST	7566	82/12/22	3000	(null)	20
3	10	7902	FORD	ANALYST	7566	81/12/11	3000	(null)	20
4	4	7566	JONES	MANAGER	7839	81/04/01	2975	(null)	20
5	2	7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
6	3	7782	CLARK	MANAGER	7839	81/05/09	2450	(null)	10
7	6	7499	ALLEN	SALESMAN	7698	81/02/11	1600	300	30
8	7	7844	TURNER	SALESMAN	7698	81/08/21	1500	0	30
9	14	7934	MILLER	CLERK	7782	82/01/11	1300	(null)	10
10	9	7521	WARD	SALESMAN	7698	81/02/23	1250	500	30
11	5	7654	MARTIN	SALESMAN	7698	81/09/10	1250	1400	30
12	13	7876	ADAMS	CLERK	7788	83/01/15	1100	(null)	20
13	8	7900	JAMES	CLERK	7698	81/12/11	950	(null)	30
14	11	7369	SMITH	CLERK	7902	80/12/09	800	(null)	20

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

Q022

- 인라인뷰 (서브쿼리) 를 이용하여
- EMP 테이블을 SAL열 기준 내림차순으로 정렬 및 ROWNUM의 열 데이터 번호를 이용하여 조회하시오.

	ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	1	7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
2	2	7788	SCOTT	ANALYST	7566	82/12/22	3000	(null)	20
3	3	7902	FORD	ANALYST	7566	81/12/11	3000	(null)	20
4	4	7566	JONES	MANAGER	7839	81/04/01	2975	(null)	20
5	5	7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
6	6	7782	CLARK	MANAGER	7839	81/05/09	2450	(null)	10
7	7	7499	ALLEN	SALESMAN	7698	81/02/11	1600	300	30
8	8	7844	TURNER	SALESMAN	7698	81/08/21	1500	0	30
9	9	7934	MILLER	CLERK	7782	82/01/11	1300	(null)	10
10	10	7521	WARD	SALESMAN	7698	81/02/23	1250	500	30
11	11	7654	MARTIN	SALESMAN	7698	81/09/10	1250	1400	30
12	12	7876	ADAMS	CLERK	7788	83/01/15	1100	(null)	20
13	13	7900	JAMES	CLERK	7698	81/12/11	950	(null)	30
14	14	7369	SMITH	CLERK	7902	80/12/09	800	(null)	20

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

Q023

- 인라인뷰(WITH 절)을 이용하여
- EMP 테이블을 SAL열 기준 내림차순으로 정렬 및 ROWNUM의 열 데이터 번호를 이용하여 조회하시오.

	ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	1	7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
2	2	7788	SCOTT	ANALYST	7566	82/12/22	3000	(null)	20
3	3	7902	FORD	ANALYST	7566	81/12/11	3000	(null)	20
4	4	7566	JONES	MANAGER	7839	81/04/01	2975	(null)	20
5	5	7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
6	6	7782	CLARK	MANAGER	7839	81/05/09	2450	(null)	10
7	7	7499	ALLEN	SALESMAN	7698	81/02/11	1600	300	30
8	8	7844	TURNER	SALESMAN	7698	81/08/21	1500	0	30
9	9	7934	MILLER	CLERK	7782	82/01/11	1300	(null)	10
10	10	7521	WARD	SALESMAN	7698	81/02/23	1250	500	30
11	11	7654	MARTIN	SALESMAN	7698	81/09/10	1250	1400	30
12	12	7876	ADAMS	CLERK	7788	83/01/15	1100	(null)	20
13	13	7900	JAMES	CLERK	7698	81/12/11	950	(null)	30
14	14	7369	SMITH	CLERK	7902	80/12/09	800	(null)	20

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

Q024

- 인라인뷰 (서브쿼리) 를 이용하여
- EMP 테이블을 SAL열 기준 내림차순으로 정렬 및 ROWNUM의 열 데이터 번호를 이용하여
- 마지막으로 급여가 높은 상위 세면의 데이터만 출력하시오.

	ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	1	7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
2	2	7902	FORD	ANALYST	7566	81/12/11	3000	(null)	20
3	3	7788	SCOTT	ANALYST	7566	82/12/22	3000	(null)	20

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

Q025

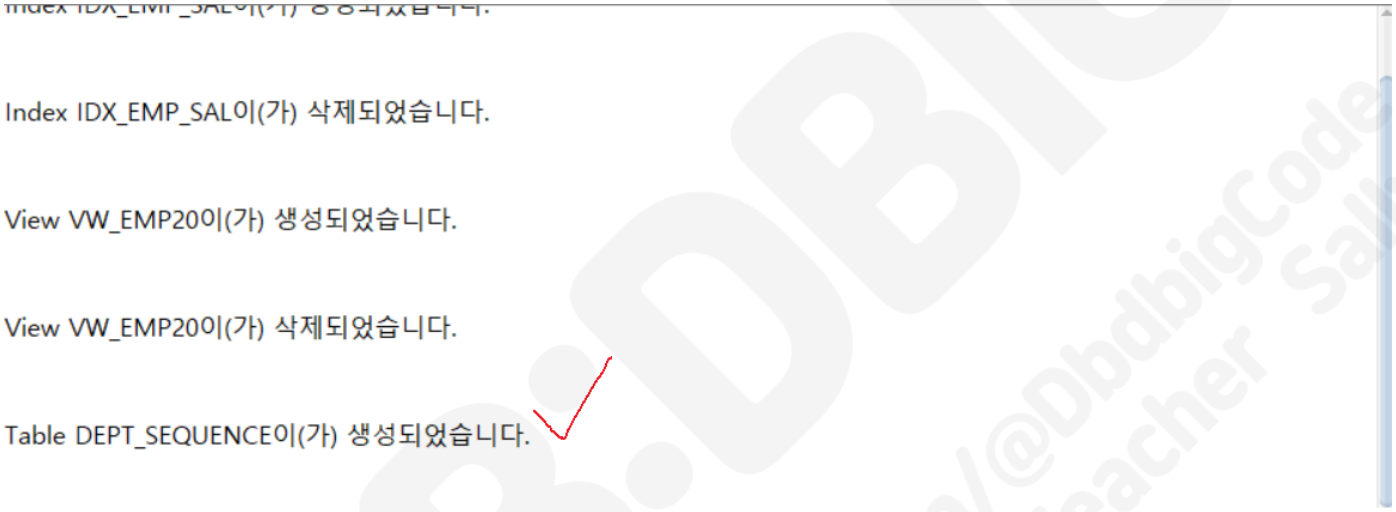
- 인라인뷰 (WITH 절절) 를 이용하여
- EMP 테이블을 SAL열 기준 내림차순으로 정렬 및 ROWNUM의 열 데이터 번호를 이용하여
- 마지막으로 급여가 높은 상위 세면의 데이터만 출력하시오.

	ROWNUM	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	1	7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
2	2	7902	FORD	ANALYST	7566	81/12/11	3000	(null)	20
3	3	7788	SCOTT	ANALYST	7566	82/12/22	3000	(null)	20

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

Q026

- DEPT 테이블을 이용하여 테이블 열구성은 갖고 데이터가 없는 DEPT_SEQUENCE 테이블을 생성하시오.




```sql
CREATE TABLE DEPT_SEQUENCE
    AS SELECT *
         FROM DEPT
        WHERE 1 <> 1;
```

Q027

- 기존의 부서번호는 10으로 시작해서 10씩증가하고 최대는 90, 최소는 0, 반복안하는 SEQ_DEPT_SEQUENCE 시퀀스를 작성하시오.

index IDX_EMP_SAL이(가) 삭제되었습니다.

View VW_EMP20이(가) 생성되었습니다.

View VW_EMP20이(가) 삭제되었습니다.

Table DEPT_SEQUENCE이(가) 생성되었습니다.

Sequence SEQ_DEPT_SEQUENCE이(가) 생성되었습니다.

```
CREATE SEQUENCE SEQ_DEPT_SEQUENCE
    INCREMENT BY 10
    START WITH 10
    MAXVALUE 90
    MINVALUE 0
    NOCYCLE
    CACHE 2;
```

Q028

- 생성한 시퀀스 SEQ_DEPT_SEQUENCE 를 조회하시오.

	SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	CYCLE_FLAG	ORDER_FLAG	CACHE_SIZE	LAST_NUMBER
1	SEQ_DEPT_SEQUENCE	0	90	10	N	N	2	10

```
SELECT *
  FROM USER_SEQUENCES;
```

Q029

- DEPT_SEQUENCE 테이블에 SEQ_DEPT_SEQUENCE 시퀀스를 이용하여 다음과 같이 데이터를 삽입하시오.

	DEPTNO	DNAME	LOC
1	10	DATABASE	SEOUL

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

SELECT * FROM DEPT_SEQUENCE ORDER BY DEPTNO;
```

Q030

- 가장 마지막으로 생성한 시퀀스를 확인하시오.

	CURRVAL
1	10

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

Q031

- 시퀀스에서 생성한 순번을 사용하여 INSERT문을 실행하시오.

	DEPTNO	DNAME	LOC
1	10	DATABASE	SEOUL
2	20	DATABASE	SEOUL

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

SELECT * FROM DEPT_SEQUENCE ORDER BY DEPTNO;
```

Q032

- SEQ_DEPT_SEQUENCE 시퀀스를 최대값 99, 증가값을 3, CYCLE옵션을 주어 다음과 같이 수정하시오.

```
Sequence SEQ_DEPT_SEQUENCE이(가) 생성되었습니다.


1 행 이(가) 삽입되었습니다.

>>Query Run In:질의 결과

1 행 이(가) 삽입되었습니다.

>>Query Run In:질의 결과 1

Sequence SEQ_DEPT_SEQUENCE이(가) 변경되었습니다.
```

```
ALTER SEQUENCE SEQ_DEPT_SEQUENCE
    INCREMENT BY 3
    MAXVALUE 99
    CYCLE;
```

Q033

- 옵션을 수정한 시퀀스를 조회하시오.

	SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	CYCLE_FLAG	ORDER_FLAG	CACHE_SIZE	LAST_NUMBER
1	SEQ_DEPT_SEQUENCE	0	99	3	Y	N	2	23

```sql
SELECT *
  FROM USER_SEQUENCES;
```

Q034

- 수정한 시퀀스로 다음과 같이 데이터를 삽입하시오.

	DEPTNO	DNAME	LOC
1	10	DATABASE	SEOUL
2	20	DATABASE	SEOUL
3	23	DATABASE	SEOUL

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

SELECT * FROM DEPT_SEQUENCE ORDER BY DEPTNO;
```

Q035

- SEQ_DEPT_SEQUENCE 시퀀스의 최대값 도달 후 수행결과를 확인하시오.

	DEPTNO	DNAME	LOC
1	10	DATABASE	SEOUL
2	20	DATABASE	SEOUL
3	23	DATABASE	SEOUL
4	26	DATABASE	SEOUL

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

SELECT * FROM DEPT_SEQUENCE ORDER BY DEPTNO;
```

Q036

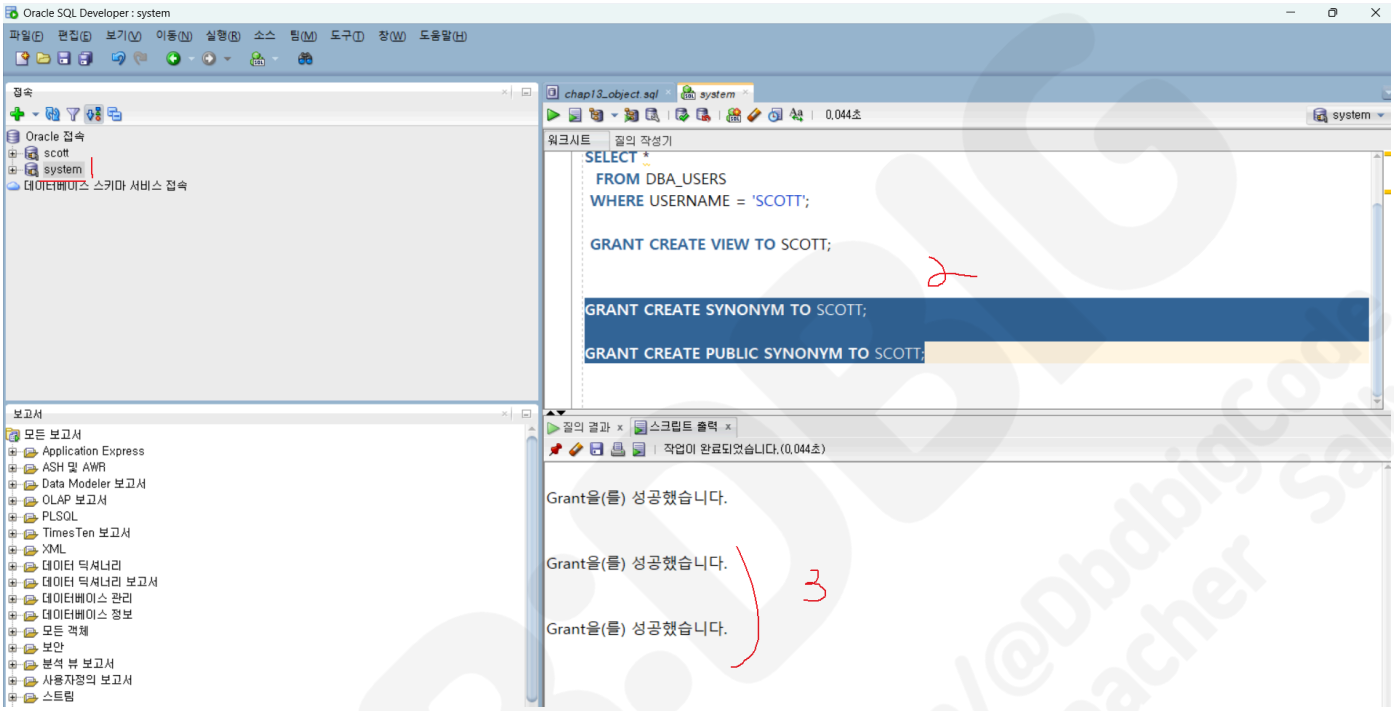
- SEQ_DEPT_SEQUENCE 시퀀스 삭제후 확인하시오.

SEQUE...	MIN_VA...	MAX_VA...	INCREM...	CYCLE_...	ORDER_...	CACHE_...	LAST_N...

```
DROP SEQUENCE SEQ_DEPT_SEQUENCE;

SELECT * FROM USER_SEQUENCES;
```

Q037

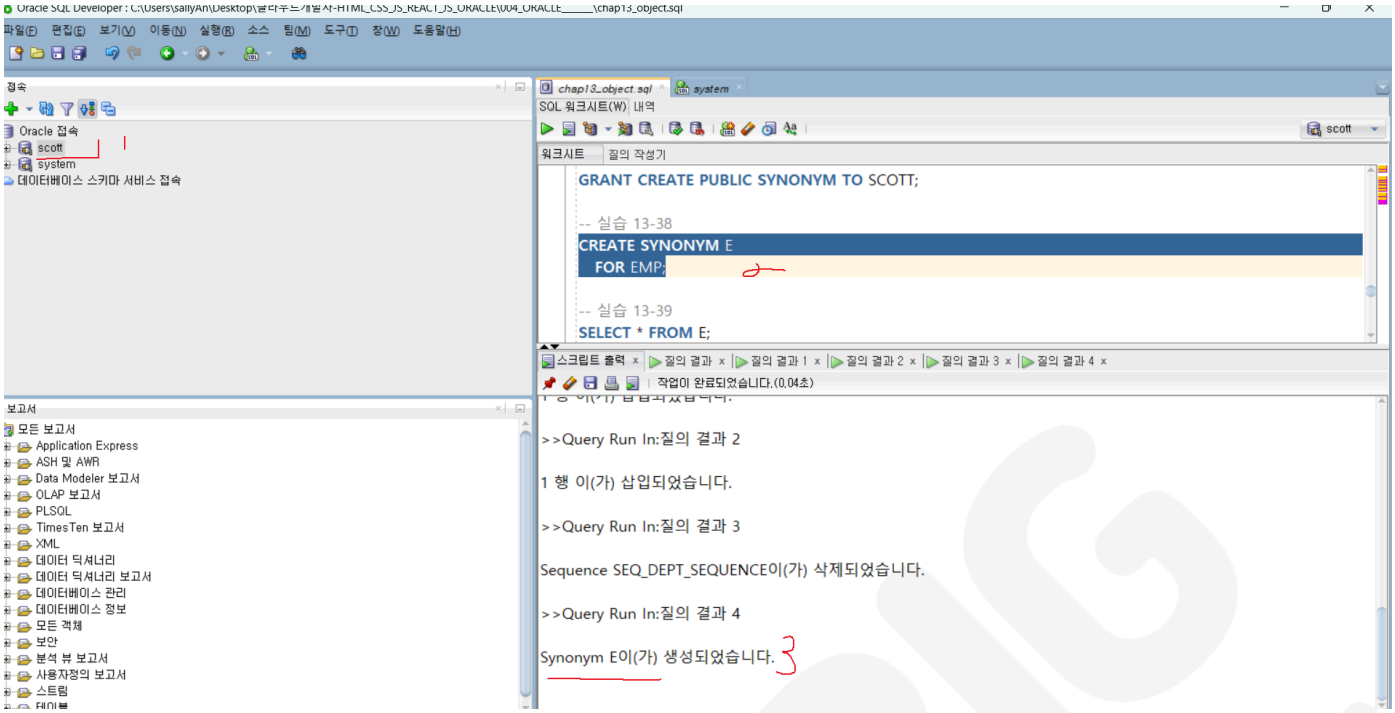
- SYNONYM? 동의어
- 테이블, 뷰, 시퀀스 등 객체 이름 대신에 사용할 수 있는 다른 이름을 뷰여하는 객체
- 테이블 이름이 너무길어 사용이 불편할때 좀더 간단하고 짧은 이름을 하나 더 만들어주기 위해 사용함.
- (SYSTEM 계정) SCOTT에 CREATE SYNONYM 생성권한,
  PUBLIC 데이터베이스 내 모든 사용자가 사용할수 있도록 설정하는 권한 부여




```
SQLPLUS SYSTEM/oracle

GRANT CREATE SYNONYM TO SCOTT;

GRANT CREATE PUBLIC SYNONYM TO SCOTT;
```

Q038

- (SCOTT) EMP 테이블의 동의어를 E로 생성하시오.




```
CREATE SYNONYM E
   FOR EMP;
```

Q039

- E 테이블 전체 내용을 조회하시오.

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
2	7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
3	7782	CLARK	MANAGER	7839	81/05/09	2450	(null)	10
4	7566	JONES	MANAGER	7839	81/04/01	2975	(null)	20
5	7654	MARTIN	SALESMAN	7698	81/09/10	1250	1400	30
6	7499	ALLEN	SALESMAN	7698	81/02/11	1600	300	30
7	7844	TURNER	SALESMAN	7698	81/08/21	1500	0	30
8	7900	JAMES	CLERK	7698	81/12/11	950	(null)	30
9	7521	WARD	SALESMAN	7698	81/02/23	1250	500	30
10	7902	FORD	ANALYST	7566	81/12/11	3000	(null)	20
11	7369	SMITH	CLERK	7902	80/12/09	800	(null)	20
12	7788	SCOTT	ANALYST	7566	82/12/22	3000	(null)	20
13	7876	ADAMS	CLERK	7788	83/01/15	1100	(null)	20
14	7934	MILLER	CLERK	7782	82/01/11	1300	(null)	10

```
SELECT * FROM E;
```

Q040

- E 동의어를 삭제하시오.

```
1 행 이(가) 삽입되었습니다.

>>Query Run In:질의 결과 3

Sequence SEQ_DEPT_SEQUENCE이(가) 삭제되었습니다.

>>Query Run In:질의 결과 4

Synonym E이(가) 생성되었습니다.


Synonym E이(가) 삭제되었습니다.
```

```
DROP SYNONYM E;
```

■STEP2. EX

EX001

1. EMP 테이블과 같은 구조의 데이터를 저장하는 EMPIDX 테이블을 생성하시오.
2. 생성한 EMPIDX 테이블의 EMPNO열에 IDX_EMPIDX_EMPNO 인덱스를 생성하시오.
3. 인덱스가 잘 생성되었는지 적절한 데이터 사전뷰를 통해 확인하시오.

Table EMPIDX이(가) 생성되었습니다.

Index IDX_EMPIDX_EMPNO이(가) 생성되었습니다.

	INDEX_NAME	INDEX_TYPE	TABLE_OWNER	TABLE_NAME	TABLE_TYPE	UNIQUENESS	COMPRESSION	PREFIX_LENG
1	IDX_EMPIDX_EMPNO	NORMAL	SCOTT	EMPIDX	TABLE	NONUNIQUE	DISABLED	(n

EX002

1. EMPIDX 테이블의 급여(SAL) 가 1500 초과인 사원들만 출력하는 EMPIDX_OVER15K 뷰를 생성하시오.
2. 만약 이뷰가 존재한다면 새로운 내용으로 대체가능하게 작성하시오.
3. EMPIDX_OVER15K 뷰는 EMPNO, ENAME, JOB, DEPTNO, SAL , COMM 이 있다면 O 존재하지 않으면 X 열을 가지고 있습니다.

	EMPNO	ENAME	JOB	DEPTNO	SAL	COMM
1	7839	KING	PRESIDENT	10	5000	X
2	7698	BLAKE	MANAGER	30	2850	X
3	7782	CLARK	MANAGER	10	2450	X
4	7566	JONES	MANAGER	20	2975	X
5	7499	ALLEN	SALESMAN	30	1600	O
6	7902	FORD	ANALYST	20	3000	X
7	7788	SCOTT	ANALYST	20	3000	X

EX003

1. DEPT 테이블과 같은 결과 행 구성ㅇ르 가지는 DEPTSEQ 테이블을 작성하시오.
2. DEPTSEQ 테이블에 사용할 SEQ_DEPTSEQ 시퀀스를 작성하시오.
   시작값1 , 증가1, 최대값 99, 최소값 1, 부서번호 최대값에서 생성중단, 캐시 없음

3. DEPTSEQ 를 이용하여 다음과 같이 세개 부서를 차례대로 추가하시오.

Table DEPTSEQ이(가) 생성되었습니다.

Sequence SEQ_DEPTSEQ이(가) 생성되었습니다.

	DEPTNO	DNAME	LOC
1	10	ACCOUNTING	NEW YORK
2	20	RESEARCH	DALLAS
3	30	SALES	CHICAGO
4	40	OPERATIONS	BOSTON
5	1	DATABASE	SEOUL
6	2	WEB	BUSAN
7	3	MOBILE	ILSAN