10차 SQL 과제

1. CRATE TABLE

CREATE OR REPLACE PROCEDURE PROC\_INS\_TABLE(P\_OWNER IN VARCHAR2, P\_TABLE\_NAME IN VARCHAR2)

IS

L\_COLUMN\_CNT INTEGER;

L\_TABLESPACE\_NAME VARCHAR2(100);

L\_OWNER VARCHAR2(100);

BEGIN

SELECT TABLESPACE\_NAME INTO L\_TABLESPACE\_NAME FROM DBA\_TABLES WHERE OWNER = P\_OWNER AND TABLE\_NAME = P\_TABLE\_NAME;

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER, 'TABLE', P\_TABLE\_NAME, 1, 'CREATE TABLE '|| P\_OWNER || '.'|| P\_TABLE\_NAME);

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER, 'TABLE', P\_TABLE\_NAME, 2, '(');

FOR L\_CURSOR IN (SELECT \* FROM (SELECT OWNER, 'TABLE', TABLE\_NAME, COLUMN\_ID ,

COLUMN\_NAME

|| ' '

|| DATA\_TYPE

|| CASE WHEN DATA\_TYPE = 'DATE' OR DATA\_TYPE ='CLOB' THEN NULL

WHEN DATA\_TYPE = 'VARCHAR2' OR DATA\_TYPE ='RAW' THEN '('|| DATA\_LENGTH ||')'

WHEN DATA\_PRECISION IS NULL THEN NULL

WHEN DATA\_PRECISION IS NOT NULL AND DATA\_SCALE IS NOT NULL AND DATA\_SCALE != 0 THEN '('||DATA\_PRECISION||','||DATA\_SCALE||')'

WHEN DATA\_PRECISION IS NOT NULL AND DATA\_SCALE IS NOT NULL AND DATA\_SCALE = 0 THEN '('||DATA\_PRECISION||')'

END

|| DECODE(NULLABLE, 'N', ' NOT NULL')

|| DECODE(COLUMN\_ID, MAX(COLUMN\_ID) OVER(PARTITION BY P\_OWNER, P\_TABLE\_NAME),NULL,',')

AS TEXT

FROM DBA\_TAB\_COLUMNS

WHERE

OWNER = P\_OWNER

AND TABLE\_NAME = P\_TABLE\_NAME

ORDER BY COLUMN\_ID))

LOOP

INSERT INTO DDL\_SCRIPTS VALUES(L\_CURSOR.OWNER, 'TABLE', L\_CURSOR.TABLE\_NAME, L\_CURSOR.COLUMN\_ID+2, L\_CURSOR.TEXT);

COMMIT;

END LOOP;

SELECT MAX(LINE) INTO L\_COLUMN\_CNT

FROM DDL\_SCRIPTS

WHERE OBJECT\_NAME = P\_TABLE\_NAME

AND OWNER = P\_OWNER

GROUP BY OWNER, OBJECT\_NAME;

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER, 'TABLE', P\_TABLE\_NAME, L\_COLUMN\_CNT +1, ')');

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER, 'TABLE', P\_TABLE\_NAME, L\_COLUMN\_CNT +2, 'TABLESPACE '|| L\_TABLESPACE\_NAME ||';');

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('데이터 입력 성공!');

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN DBMS\_OUTPUT.PUT\_LINE(P\_OWNER || '에 ' || P\_TABLE\_NAME ||'이(가) 없습니다.');

DBMS\_OUTPUT.PUT\_LINE('OWNER 이름과 TABLE 이름을 다시 확인해주세요.');

END PROC\_INS\_TABLE;

2. CRATE INDEX

CREATE OR REPLACE PROCEDURE PROC\_INS\_INDEX(P\_OWNER IN VARCHAR2, P\_INDEX\_NAME IN VARCHAR2)

IS

L\_TABLESPACE\_NAME VARCHAR2(100);

L\_COLUMNS\_CNT INTEGER;

L\_INDEX\_CHECK VARCHAR2(100);

L\_COLUMN\_EXPRESSION VARCHAR2(4000);

L\_COLUMN\_EXPRESSION\_LEN NUMBER;

L\_UNIQUENESS VARCHAR2(100);

L\_TABLE\_NAME VARCHAR2(100);

BEGIN

SELECT DISTINCT INDEX\_OWNER INTO L\_INDEX\_CHECK FROM DBA\_IND\_COLUMNS WHERE INDEX\_OWNER = P\_OWNER AND INDEX\_NAME = P\_INDEX\_NAME;

SELECT COUNT(\*) INTO L\_COLUMNS\_CNT FROM DBA\_IND\_COLUMNS WHERE INDEX\_OWNER = P\_OWNER AND INDEX\_NAME = P\_INDEX\_NAME;

FOR L\_INDEX\_INFO IN (SELECT OWNER, INDEX\_NAME, UNIQUENESS, TABLESPACE\_NAME, TABLE\_NAME

FROM DBA\_INDEXES

WHERE OWNER = P\_OWNER

AND INDEX\_NAME = P\_INDEX\_NAME)

LOOP

L\_TABLESPACE\_NAME := L\_INDEX\_INFO.TABLESPACE\_NAME;

L\_UNIQUENESS := L\_INDEX\_INFO.UNIQUENESS;

L\_TABLE\_NAME := L\_INDEX\_INFO.TABLE\_NAME;

END LOOP;

INSERT INTO DDL\_SCRIPTS (OWNER, OBJECT\_TYPE, OBJECT\_NAME, LINE, TEXT)

SELECT P\_OWNER AS OWNER, 'INDEX' AS OBJECT\_TYPE, P\_INDEX\_NAME AS OBJECT\_NAME, LINE, DECODE(LINE, 1, 'CREATE ' || DECODE(L\_UNIQUENESS, 'UNIQUE', 'UNIQUE ', NULL) || 'INDEX ' || P\_OWNER || '.' || P\_INDEX\_NAME,

2, 'ON ' || P\_OWNER || '.' || L\_TABLE\_NAME,

3, '(') AS TEXT

FROM (SELECT 1 LINE FROM DUAL

UNION ALL

SELECT 2 FROM DUAL

UNION ALL

SELECT 3 FROM DUAL);

FOR L\_CURSOR\_COLUMNS IN (SELECT A.INDEX\_OWNER, A.INDEX\_NAME, A.TABLE\_NAME, A.COLUMN\_NAME, A.COLUMN\_POSITION, A.DESCEND, B.COLUMN\_EXPRESSION AS COLUMN\_EXPRESSION

FROM DBA\_IND\_COLUMNS A, (SELECT COLUMN\_EXPRESSION, COLUMN\_POSITION FROM DBA\_IND\_EXPRESSIONS WHERE INDEX\_NAME = P\_INDEX\_NAME) B

WHERE A.INDEX\_OWNER = P\_OWNER

AND A.INDEX\_NAME = P\_INDEX\_NAME

AND A.COLUMN\_POSITION = B.COLUMN\_POSITION(+)

ORDER BY COLUMN\_POSITION)

LOOP

L\_COLUMN\_EXPRESSION\_LEN := LENGTH(L\_CURSOR\_COLUMNS.COLUMN\_EXPRESSION);

L\_COLUMN\_EXPRESSION :=SUBSTR(L\_CURSOR\_COLUMNS.COLUMN\_EXPRESSION,2,L\_COLUMN\_EXPRESSION\_LEN-2);

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER,

'INDEX',

P\_INDEX\_NAME,

L\_CURSOR\_COLUMNS.COLUMN\_POSITION+3,

NVL(L\_COLUMN\_EXPRESSION, L\_CURSOR\_COLUMNS.COLUMN\_NAME)

|| ' ' || L\_CURSOR\_COLUMNS.DESCEND

|| DECODE(L\_CURSOR\_COLUMNS.COLUMN\_POSITION,

L\_COLUMNS\_CNT,

' ',

','));

END LOOP;

INSERT INTO DDL\_SCRIPTS (OWNER, OBJECT\_TYPE, OBJECT\_NAME, LINE, TEXT)

SELECT P\_OWNER AS OWNER, 'INDEX' AS OBJECT\_TYPE, P\_INDEX\_NAME AS OBJECT\_NAME,LINE, DECODE(LINE, L\_COLUMNS\_CNT +5, 'TABLESPACE ' || L\_TABLESPACE\_NAME || ';', ')') AS TEXT

FROM (SELECT L\_COLUMNS\_CNT +4 LINE FROM DUAL

UNION ALL

SELECT L\_COLUMNS\_CNT +5 FROM DUAL);

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('데이터 입력 성공!');

EXCEPTION WHEN NO\_DATA\_FOUND

THEN DBMS\_OUTPUT.PUT\_LINE(P\_OWNER || '에 ' || P\_INDEX\_NAME ||'이(가) 없습니다.');

DBMS\_OUTPUT.PUT\_LINE('OWNER 이름과 INDEX 이름을 다시 확인해주세요.');

END PROC\_INS\_INDEX;

3. CREATE PACKAGE

(PACKAGE 선언부 생성)

CREATE OR REPLACE PACKAGE PAK\_INS\_TABLE\_INDEX

AS

PROCEDURE PROC\_INS\_TABLE(P\_OWNER IN VARCHAR2, P\_TABLE\_NAME IN VARCHAR2);

PROCEDURE PROC\_INS\_INDEX(P\_OWNER IN VARCHAR2, P\_INDEX\_NAME IN VARCHAR2);

PROCEDURE PROC\_INS\_TABLE\_INDEX(P\_OWNER IN VARCHAR2, P\_TABLE\_NAME IN VARCHAR2);

END PAK\_INS\_TABLE\_INDEX;

(PACKAGE 본문 생성)

CREATE OR REPLACE PACKAGE BODY PAK\_INS\_TABLE\_INDEX

IS

PROCEDURE PROC\_INS\_TABLE(P\_OWNER IN VARCHAR2, P\_TABLE\_NAME IN VARCHAR2)

IS

L\_COLUMN\_CNT INTEGER;

L\_TABLESPACE\_NAME VARCHAR2(100);

L\_OWNER VARCHAR2(100);

BEGIN

SELECT TABLESPACE\_NAME INTO L\_TABLESPACE\_NAME FROM DBA\_TABLES WHERE OWNER = P\_OWNER AND TABLE\_NAME = P\_TABLE\_NAME;

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER, 'TABLE', P\_TABLE\_NAME, 1, 'CREATE TABLE '|| P\_OWNER || '.'|| P\_TABLE\_NAME);

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER, 'TABLE', P\_TABLE\_NAME, 2, '(');

FOR L\_CURSOR IN (SELECT \* FROM (SELECT OWNER, 'TABLE', TABLE\_NAME, COLUMN\_ID ,

COLUMN\_NAME

|| ' '

|| DATA\_TYPE

|| CASE WHEN DATA\_TYPE = 'DATE' OR DATA\_TYPE ='CLOB' THEN NULL

WHEN DATA\_TYPE = 'VARCHAR2' OR DATA\_TYPE ='RAW' THEN '('|| DATA\_LENGTH ||')'

WHEN DATA\_PRECISION IS NULL THEN NULL

WHEN DATA\_PRECISION IS NOT NULL AND DATA\_SCALE IS NOT NULL AND DATA\_SCALE != 0 THEN '('||DATA\_PRECISION||','||DATA\_SCALE||')'

WHEN DATA\_PRECISION IS NOT NULL AND DATA\_SCALE IS NOT NULL AND DATA\_SCALE = 0 THEN '('||DATA\_PRECISION||')'

END

|| DECODE(NULLABLE, 'N', ' NOT NULL')

|| DECODE(COLUMN\_ID, MAX(COLUMN\_ID) OVER(PARTITION BY P\_OWNER, P\_TABLE\_NAME),NULL,',')

AS TEXT

FROM DBA\_TAB\_COLUMNS

WHERE

OWNER = P\_OWNER

AND TABLE\_NAME = P\_TABLE\_NAME

ORDER BY COLUMN\_ID))

LOOP

INSERT INTO DDL\_SCRIPTS VALUES(L\_CURSOR.OWNER, 'TABLE', L\_CURSOR.TABLE\_NAME, L\_CURSOR.COLUMN\_ID+2, L\_CURSOR.TEXT);

COMMIT;

END LOOP;

SELECT MAX(LINE) INTO L\_COLUMN\_CNT

FROM DDL\_SCRIPTS

WHERE OBJECT\_NAME = P\_TABLE\_NAME

AND OWNER = P\_OWNER

GROUP BY OWNER, OBJECT\_NAME;

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER, 'TABLE', P\_TABLE\_NAME, L\_COLUMN\_CNT +1, ')');

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER, 'TABLE', P\_TABLE\_NAME, L\_COLUMN\_CNT +2, 'TABLESPACE '|| L\_TABLESPACE\_NAME ||';');

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('데이터 입력 성공!');

EXCEPTION

WHEN NO\_DATA\_FOUND

THEN DBMS\_OUTPUT.PUT\_LINE(P\_OWNER || '에 ' || P\_TABLE\_NAME ||'이(가) 없습니다.');

DBMS\_OUTPUT.PUT\_LINE('OWNER 이름과 TABLE 이름을 다시 확인해주세요.');

END PROC\_INS\_TABLE;

PROCEDURE PROC\_INS\_INDEX(P\_OWNER IN VARCHAR2, P\_INDEX\_NAME IN VARCHAR2)

IS

L\_TABLESPACE\_NAME VARCHAR2(100);

L\_COLUMNS\_CNT INTEGER;

L\_INDEX\_CHECK VARCHAR2(100);

L\_COLUMN\_EXPRESSION VARCHAR2(4000);

L\_COLUMN\_EXPRESSION\_LEN NUMBER;

L\_UNIQUENESS VARCHAR2(100);

L\_TABLE\_NAME VARCHAR2(100);

BEGIN

SELECT DISTINCT INDEX\_OWNER INTO L\_INDEX\_CHECK FROM DBA\_IND\_COLUMNS WHERE INDEX\_OWNER = P\_OWNER AND INDEX\_NAME = P\_INDEX\_NAME;

SELECT COUNT(\*) INTO L\_COLUMNS\_CNT FROM DBA\_IND\_COLUMNS WHERE INDEX\_OWNER = P\_OWNER AND INDEX\_NAME = P\_INDEX\_NAME;

FOR L\_INDEX\_INFO IN (SELECT OWNER, INDEX\_NAME, UNIQUENESS, TABLESPACE\_NAME, TABLE\_NAME

FROM DBA\_INDEXES

WHERE OWNER = P\_OWNER

AND INDEX\_NAME = P\_INDEX\_NAME)

LOOP

L\_TABLESPACE\_NAME := L\_INDEX\_INFO.TABLESPACE\_NAME;

L\_UNIQUENESS := L\_INDEX\_INFO.UNIQUENESS;

L\_TABLE\_NAME := L\_INDEX\_INFO.TABLE\_NAME;

END LOOP;

INSERT INTO DDL\_SCRIPTS (OWNER, OBJECT\_TYPE, OBJECT\_NAME, LINE, TEXT)

SELECT P\_OWNER AS OWNER, 'INDEX' AS OBJECT\_TYPE, P\_INDEX\_NAME AS OBJECT\_NAME, LINE, DECODE(LINE, 1, 'CREATE ' || DECODE(L\_UNIQUENESS, 'UNIQUE', 'UNIQUE ', NULL) || 'INDEX ' || P\_OWNER || '.' || P\_INDEX\_NAME,

2, 'ON ' || P\_OWNER || '.' || L\_TABLE\_NAME,

3, '(') AS TEXT

FROM (SELECT 1 LINE FROM DUAL

UNION ALL

SELECT 2 FROM DUAL

UNION ALL

SELECT 3 FROM DUAL);

FOR L\_CURSOR\_COLUMNS IN (SELECT A.INDEX\_OWNER, A.INDEX\_NAME, A.TABLE\_NAME, A.COLUMN\_NAME, A.COLUMN\_POSITION, A.DESCEND, B.COLUMN\_EXPRESSION AS COLUMN\_EXPRESSION

FROM DBA\_IND\_COLUMNS A, (SELECT COLUMN\_EXPRESSION, COLUMN\_POSITION FROM DBA\_IND\_EXPRESSIONS WHERE INDEX\_NAME = P\_INDEX\_NAME) B

WHERE A.INDEX\_OWNER = P\_OWNER

AND A.INDEX\_NAME = P\_INDEX\_NAME

AND A.COLUMN\_POSITION = B.COLUMN\_POSITION(+)

ORDER BY COLUMN\_POSITION)

LOOP

L\_COLUMN\_EXPRESSION\_LEN := LENGTH(L\_CURSOR\_COLUMNS.COLUMN\_EXPRESSION);

L\_COLUMN\_EXPRESSION :=SUBSTR(L\_CURSOR\_COLUMNS.COLUMN\_EXPRESSION,2,L\_COLUMN\_EXPRESSION\_LEN-2);

INSERT INTO DDL\_SCRIPTS VALUES(P\_OWNER,

'INDEX',

P\_INDEX\_NAME,

L\_CURSOR\_COLUMNS.COLUMN\_POSITION+3,

NVL(L\_COLUMN\_EXPRESSION, L\_CURSOR\_COLUMNS.COLUMN\_NAME)

|| ' ' || L\_CURSOR\_COLUMNS.DESCEND

|| DECODE(L\_CURSOR\_COLUMNS.COLUMN\_POSITION,

L\_COLUMNS\_CNT,

' ',

','));

END LOOP;

INSERT INTO DDL\_SCRIPTS (OWNER, OBJECT\_TYPE, OBJECT\_NAME, LINE, TEXT)

SELECT P\_OWNER AS OWNER, 'INDEX' AS OBJECT\_TYPE, P\_INDEX\_NAME AS OBJECT\_NAME,LINE, DECODE(LINE, L\_COLUMNS\_CNT +5, 'TABLESPACE ' || L\_TABLESPACE\_NAME || ';', ')') AS TEXT

FROM (SELECT L\_COLUMNS\_CNT +4 LINE FROM DUAL

UNION ALL

SELECT L\_COLUMNS\_CNT +5 FROM DUAL);

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('데이터 입력 성공!');

EXCEPTION WHEN NO\_DATA\_FOUND

THEN DBMS\_OUTPUT.PUT\_LINE(P\_OWNER || '에 ' || P\_INDEX\_NAME ||'이(가) 없습니다.');

DBMS\_OUTPUT.PUT\_LINE('OWNER 이름과 INDEX 이름을 다시 확인해주세요.');

END PROC\_INS\_INDEX;

PROCEDURE PROC\_INS\_TABLE\_INDEX(P\_OWNER IN VARCHAR2, P\_TABLE\_NAME IN VARCHAR2)

IS

L\_COLUMN\_CHECK NUMBER;

BEGIN

SELECT COUNT(\*) INTO L\_COLUMN\_CHECK FROM DBA\_INDEXES WHERE OWNER = P\_OWNER AND TABLE\_NAME = P\_TABLE\_NAME;

IF L\_COLUMN\_CHECK = 0 THEN

RAISE NO\_DATA\_FOUND;

END IF;

PROC\_INS\_TABLE(P\_OWNER, P\_TABLE\_NAME);

FOR L\_CURSOR IN (SELECT DISTINCT INDEX\_NAME, TABLE\_NAME FROM DBA\_IND\_COLUMNS WHERE INDEX\_OWNER = P\_OWNER AND TABLE\_NAME = P\_TABLE\_NAME)

LOOP

PROC\_INS\_INDEX(P\_OWNER, L\_CURSOR.INDEX\_NAME);

END LOOP;

EXCEPTION WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('OWNER 이름 또는 TABLE 이름을 다시 확인해 주세요.');

END PROC\_INS\_TABLE\_INDEX;

END PAK\_INS\_TABLE\_INDEX;