

DB Object 스크립트

PL-SQL SCRIPT

작성일자	2023-04-04 ~ 2023-04-07
쌍용교육센터 오라클 프로젝트 1조	
김대환, 유동현, 이동재, 이채린, 조연우	

INDEX

분류		페이지
프로시저	관리자 기능 프로시저	2
	교사 기능 프로시저	75
	교육생 기능 프로시저	108
트리거		140
뷰		149
인덱스		166

관리자 계정

요구사항번호	B-00	작성일	2023-04-04
요구사항명	관리자 로그인	작성자	1조
DML			
<pre> CREATE OR REPLACE PROCEDURE admin_P (p_ssn IN admin.ssn%TYPE) IS v_admin_seq admin.admin_seq%TYPE; v_admin_name admin.name%TYPE; v_admin_tel admin.tel%TYPE; BEGIN SELECT admin_seq, name, tel INTO v_admin_seq, v_admin_name, v_admin_tel FROM admin WHERE ssn = p_ssn; DBMS_OUTPUT.PUT_LINE('관리자 번호: ' v_admin_seq ', 관리자 이름: ' v_admin_name ', 관리자 전화번호: ' v_admin_tel); END admin_P; BEGIN admin_P(1111111); END;</pre>			

요구사항번호	B-01	작성일	2023-04-04
요구사항명	기초 정보 관리 기능	작성자	1조
DML			
<pre>--1.과정명 관리(과정명을 등록, 수정, 삭제, 조회) --등록 create or replace procedure curriculum_list_insert_p (p_seq in curriculum_list.curriculum_list_seq%type, p_name in curriculum_list.curs_name%type) is begin insert into curriculum_list(curriculum_list_seq,curs_name) values(p_seq, p_name); commit; dbms_output.put_line('INSERT SUCCESS'); end curriculum_list_insert_p; --수정 CREATE OR REPLACE PROCEDURE curriculum_list_update (curriculum_list_seq IN NUMBER, curs_name IN VARCHAR2) IS BEGIN UPDATE curriculum_list SET curs_name = curs_name</pre>			

```

WHERE curriculum_list_seq = curriculum_list_seq;
EXCEPTION WHEN OTHERS THEN
DBMS_OUTPUT.put_line('수정에 실패했습니다.');
```

ROLLBACK;

```

END curriculum_list_update;

--삭제
CREATE OR REPLACE PROCEDURE curriculum_list_delete_p(
    p_seq IN curriculum_list.curriculum_list_seq%TYPE
)
IS
BEGIN
    DELETE FROM curriculum_list
    WHERE curriculum_list_seq = p_seq;
    COMMIT;
END curriculum_list_delete_p;

--조회
select * from curriculum_list;
```

-2. 과목명 관리(과목명을 등록, 수정, 삭제, 조회)

--등록

create or replace procedure Subject_insert_p (p_seq in Subject.subject_seq%type, p_name in Subject.subject_name%type)

is

begin

insert into Subject(subject_seq,subject_name) values(p_seq, p_name);

commit;

dbms_output.put_line('INSERT SUCCESS');

end Subject_insert_p;

--수정

CREATE OR REPLACE PROCEDURE subject_update (subject_seq IN NUMBER, subject_name IN VARCHAR2)

IS

BEGIN

UPDATE subject

SET

subject_name = subject_name

WHERE subject_seq = subject_seq;

EXCEPTION WHEN OTHERS THEN

DBMS_OUTPUT.put_line('수정에 실패했습니다.');

ROLLBACK;

END subject_update;

```

--삭제
create or replace procedure subject_delete_p(p_seq in subject.subject_seq%type)
is
begin
    delete from subject
    where subject_seq = p_seq;
    commit;
end subject_delete_p;

--조회
select * from Subject;

--3. 강의실명 관리(강의실명을 등록, 수정, 삭제, 조회)
--등록
create or replace procedure Room_insert_p (p_seq in Room.room_seq%type, p_name in Room.room_name%type, p_limit in Room.student_limit%type)
is
begin
    insert into Room(room_seq,room_name,student_limit) values(p_seq, p_name, p_limit);
    commit;
    dbms_output.put_line('INSERT SUCCESS');
end Room_insert_p;

```


--수정

```
CREATE OR REPLACE PROCEDURE room_update (room_seq IN NUMBER, room_name IN VARCHAR2, student_limit in NUMBER)
```

```
IS
```

```
BEGIN
```

```
    UPDATE room
```

```
    SET
```

```
    room_name = room_name,
```

```
    student_limit = student_limit
```

```
    WHERE room_seq = room_seq;
```

```
    EXCEPTION WHEN OTHERS THEN
```

```
        DBMS_OUTPUT.put_line('수정에 실패했습니다.');
```

```
    ROLLBACK;
```

```
END room_update;
```

--삭제

```
create or replace procedure room_delete_p(p_seq in room.room_seq%type)
```

```
is
```

```
begin
```

```
    delete from room
```

```
    where room_seq = p_seq;
```

```
    commit;
```

```
end room_delete_p;
```

--조회

```
select * from room;
```

요구사항번호	B-02	작성일	2023-04-04
요구사항명	교사 계정 관리 기능	작성자	1조
DML			
<p>--1. 교사 정보 등록(이름, 주민번호 뒷자리, 전화번호, 강의 가능 과목을 입력하여 교사정보를 등록)</p> <pre> create or replace procedure Teacher_insert_p (p_seq in Teacher.teacher_seq%type, p_name in Teacher.name%type, p_ssn in Teacher.ssn%type, p_tel in Teacher.tel%type) is begin insert into Teacher(teacher_seq,name,ssn,tel) values(p_seq, p_name, p_ssn, p_tel); commit; dbms_output.put_line('INSERT SUCCESS'); end Teacher_insert_p; </pre> <p>--2. 교사 정보 수정</p> <pre> CREATE OR REPLACE PROCEDURE teacher_update (teacher_seq IN NUMBER, name in VARCHAR2, ssn in NUMBER, tel in VARCHAR2) IS BEGIN UPDATE teacher SET name = name, ssn = ssn, tel = tel WHERE teacher_seq = teacher_seq; EXCEPTION WHEN OTHERS THEN </pre>			

```
DBMS_OUTPUT.put_line('수정에 실패했습니다.');
```

```
ROLLBACK;
```

```
END teacher_update;
```

--3. 교사 정보 삭제

```
create or replace procedure teacher_delete_p(  
    p_teacher_seq in teacher.teacher_seq%type)  
is  
begin  
    delete from teacher  
    where teacher_seq = p_teacher_seq;  
    commit;  
end teacher_delete_p;
```

--4. 교사 정보 조회(이름, 주민번호 뒷자리, 전화번호, 강의 가능 과목)을 출력한다.

– 선언문

```
CREATE OR REPLACE PROCEDURE teacher_info_print_p AS  
BEGIN  
    FOR t IN (  
        SELECT t.name AS "이름", t.ssn AS "주민번호뒷자리", t.tel AS "전화번호", s.subject_name AS "강의가능과목"  
        FROM teacher t  
            INNER JOIN teaching_subject ts  
                ON t.teacher_seq = ts.teacher_seq  
            INNER JOIN subject s  
                ON ts.subject_seq = s.subject_seq  
    )
```

```

LOOP
    DBMS_OUTPUT.PUT_LINE('이름: ' || t."이름");
    DBMS_OUTPUT.PUT_LINE('주민번호뒷자리: ' || t."주민번호뒷자리");
    DBMS_OUTPUT.PUT_LINE('전화번호: ' || t."전화번호");
    DBMS_OUTPUT.PUT_LINE('강의가능과목: ' || t."강의가능과목");
    DBMS_OUTPUT.PUT_LINE('-----');
END LOOP;
END teacher_info_print_p;

-- 실행문
BEGIN
    teacher_info_print_p;
END;
```

요구사항번호	B-03	작성일	2023-04-04
요구사항명	개설 과정 관리	작성자	1조
DML			
<pre> --1. 과정등록 -- 과정번호, 과정명, 과정기간, 강의실명, 상세과정 create or replace procedure Open_curs_insert_p (p_seq in Open_curs.open_curs_seq%type, p_curriculum_list_seq in Open_curs.curriculum_list_seq%type, p_room_seq in Open_curs.room_seq%type, p_teacher_seq in Open_curs.teacher_seq%type, p_curs_name in Open_curs.curs_name%type, p_begin_date in Open_curs.begin_date%type, p_end_date in Open_curs.end_date%type, p_student_limit in Open_curs.student_limit%type) is begin insert into Open_curs(open_curs_seq,curriculum_list_seq,room_seq,teacher_seq,curs_name,begin_date,end_date,student_limit) values(p_seq, p_curriculum_list_seq, p_room_seq, p_teacher_seq, p_curs_name, p_begin_date, p_end_date, p_student_limit); commit; dbms_output.put_line('INSERT SUCCESS'); end Open_curs_insert_p; --2. 과정 정보 수정 CREATE OR REPLACE PROCEDURE open_curs_update (open_curs_seq in number, curs_name in varchar2) IS BEGIN UPDATE open_curs </pre>			

```
SET
  curs_name = curs_name
WHERE open_curs_seq = open_curs_seq;
EXCEPTION WHEN OTHERS THEN
  DBMS_OUTPUT.put_line('수정에 실패했습니다.');
```

ROLLBACK;

END open_curs_update;

--3. 개설 과정 삭제

```
CREATE OR REPLACE PROCEDURE open_curs_delete_p(p_seq in open_curs.open_curs_seq%type)
is
begin
  delete from open_curs
  where open_curs_seq = p_seq;
  commit;
end open_curs_delete_p;
```

--4. 과정 조회

--4-1 개설 과정번호, 과정명, 과정기간, 강의실명, 개설 과목 등록 여부, 교육생 수용 인원을 출력

```
create or replace procedure open_curs_select_p
is
begin
```

```

For c In (
select distinct oc.open_curs_seq as "개설 과정번호", oc.curs_name as "과정명",
CASE
    WHEN ROUND(MONTHS_BETWEEN(oc.end_date, oc.begin_date),1) - FLOOR(MONTHS_BETWEEN(oc.end_date, oc.begin_date)) >= 0.5
    THEN CEIL(MONTHS_BETWEEN(oc.end_date, oc.begin_date)) - 0.5
    ELSE FLOOR(MONTHS_BETWEEN(oc.end_date, oc.begin_date))
END || '개월' as "과정기간",
r.room_name as "강의실명",
CASE WHEN cs.subject_seq is not null THEN '과목 등록 완료' ELSE '과목 등록 미완료' END as "개설 과목 등록 여부",
oc.student_limit as "교육생 수용 인원"
from open_curs oc
inner join room r on oc.room_seq = r.room_seq
inner join curriculum_list cl on oc.curriculum_list_seq = cl.curriculum_list_seq
inner join curriculum_subject cs on cl.curriculum_list_seq = cs.curriculum_list_seq
)
LOOP
    DBMS_OUTPUT.PUT_LINE('개설 과정번호: ' || c."개설 과정번호" || ', 과정명: ' || c."과정명" || ', 과정기간: ' || c."과정기간" || ', 강의실명: ' || c."강의실명"
|| ', 개설 과목 등록 여부: ' || c."개설 과목 등록 여부" || ', 교육생 수용 인원: ' || c."교육생 수용 인원");
END LOOP;
end open_curs_select_p;

```



```

--실행
begin
open_curs_select_p;
end;

--4-2 개설 과정 선택시, 개설과목정보(과목명, 과목기간, 교재명, 교사명)을 출력
CREATE OR REPLACE PROCEDURE opencurs_opensubject_info(popen_curs_seq IN number)
AS
BEGIN
    FOR c IN (
        select
            sj.subject_name as "과목명",
            os.begin_date as "과목시작날짜",
            os.end_date as "과목종료날짜",
            b.subject as "교재명",
            t.name as "교사명"
        from open_curs oc
            inner join curriculum_subject cs on oc.curriculum_list_seq = cs.curriculum_list_seq
            inner join subject sj
                on cs.subject_seq = sj.subject_seq
            inner join open_subject os
                on sj.subject_seq = os.subject_seq and oc.open_curs_seq = os.open_curs_seq
    )

```

```

        inner join subject_book sb
        on sj.subject_seq = sb.subject_seq
        inner join book b
        on sb.book_seq = b.book_seq
        inner join teacher t
        on oc.teacher_seq = t.teacher_seq
        where oc.open_curs_seq = popen_curs_seq

    ) LOOP
        DBMS_OUTPUT.PUT_LINE('과목명: ' || c.과목명);
        DBMS_OUTPUT.PUT_LINE('과목기간(시작): ' || c.과목시작날짜);
        DBMS_OUTPUT.PUT_LINE('과목기간(끝): ' || c.과목종료날짜);
        DBMS_OUTPUT.PUT_LINE('교재명: ' || c.교재명);
        DBMS_OUTPUT.PUT_LINE('교사명: ' || c.교사명);
    END LOOP;
END opencurs_opensubject_info;

--4-3 개설 과정 선택시, 교육생 정보(이름, 주민번호 뒷자리, 전화번호, 등록일, 수료여부(수료,중도탈락))을 출력
CREATE OR REPLACE PROCEDURE open_curs_students_select_p(
    popen_curs_seq open_curs.open_curs_seq%type
)

```

```

IS
BEGIN
    FOR c IN (
    SELECT
        m.name AS 이름,
        m.ssn AS "주민번호 뒷자리",
        m.tel AS 전화번호,
        s.enroll_date AS 등록일,
        s.drop_out AS "수료 여부"
    FROM student s
        INNER JOIN open_curs oc
            ON s.curriculum_list_seq = oc.curriculum_list_seq
        INNER JOIN member m
            ON m.member_seq = s.member_seq
    WHERE open_curs_seq = popen_curs_seq
    )
    LOOP
        DBMS_OUTPUT.PUT_LINE('이름: ' || c.이름);
        DBMS_OUTPUT.PUT_LINE('주민번호 뒷자리: ' || c."주민번호 뒷자리");
        DBMS_OUTPUT.PUT_LINE('전화번호: ' || c.전화번호);
        DBMS_OUTPUT.PUT_LINE('등록일: ' || c.등록일);
        DBMS_OUTPUT.PUT_LINE("");
    
```

```
END LOOP;  
END open_curs_students_select_p;  
/  
  
-5 과정 수료 처리  
create or replace procedure student_update_p(p_seq in student.student_student_seq%type)  
is  
begin  
    update student set drop_out = '수료완료', drop_out_date = sysdate  
    where p_seq = seq;  
    commit;  
end teacher_delete_p;
```

요구사항번호	B-04	작성일	2023-04-04
요구사항명	개설 과목 관리	작성자	1조
DML			
<p>--5. 특정 개설 과정 선택 시 개설 과목 신규 등록을 할 수 있다.</p> <p>-- 개설 과목 정보에 대한 입력, 수정, 삭제, 조회 기능</p> <p>--입력</p> <pre> create or replace procedure Curriculum_subject_insert_p (p_seq in Curriculum_subject.curriculum_list_seq%type, p_subject_seq in Curriculum_subject.subject_seq%type) is begin insert into Curriculum_subject(curriculum_list_seq,subject_seq) values(p_seq, p_subject_seq); commit; dbms_output.put_line('INSERT SUCCESS'); end Curriculum_subject_insert_p; </pre> <p>--수정</p> <pre> CREATE OR REPLACE PROCEDURE curriculum_subject_update (curriculum_list_seq in number, subject_seq in number) IS BEGIN UPDATE curriculum_subject SET </pre>			

```

    subject_seq = subject_seq
    WHERE curriculum_list_seq = curriculum_list_seq and subject_seq = subject_seq;

    EXCEPTION WHEN OTHERS THEN
        DBMS_OUTPUT.put_line('수정에 실패했습니다.');
```

ROLLBACK;

END curriculum_subject_update;

--삭제

```

create or replace procedure curriculum_subject_delete_p(
    p_curriculum_list_seq in curriculum_subject.curriculum_list_seq%type
)
is
begin
    delete from curriculum_subject
    where curriculum_list_seq = p_curriculum_list_seq;
    commit;
end curriculum_subject_delete_p;
```

--조회

— 선언문

```

CREATE OR REPLACE PROCEDURE curriculum_subject_print_p
IS
```

```
BEGIN
  FOR subject_record IN (SELECT * FROM curriculum_subject)
  LOOP
    DBMS_OUTPUT.PUT_LINE('CURRICULUM_LIST_SEQ: ' || subject_record.CURRICULUM_LIST_SEQ);
    DBMS_OUTPUT.PUT_LINE('SUBJECT_SEQ: ' || subject_record.SUBJECT_SEQ);
  END LOOP;
END curriculum_subject_print_p;
-- 호출문
DECLARE
  v_curriculum_list_seq NUMBER := 1;
BEGIN
  curriculum_subject_print_p;
END;
```

요구사항번호	B-05	작성일	2023-04-04
요구사항명	교육생 관리	작성자	1조
DML			
<pre> --입력 create or replace procedure Member_insert_p (p_seq in Member.member_seq%type, p_name in Member.name%type, p_ssn in Member.ssn%type, p_tel in Member.tel%type) is begin insert into Member(member_seq,name,ssn,tel) values(p_seq, p_name, p_ssn, p_tel); commit; dbms_output.put_line('INSERT SUCCESS'); end Member_insert_p; --출력(교육생 이름, 주민번호 뒷자리, 전화번호, 등록일, 수강신청 횟수) CREATE OR REPLACE PROCEDURE student_info IS BEGIN FOR c IN (SELECT mb.name AS "교육생 이름", mb.ssn AS "주민번호 뒷자리", mb.tel AS "전화번호", st.enroll_date AS "등록일", COUNT(en.enrollment_seq) AS "수강신청 횟수" FROM member mb </pre>			


```

        INNER JOIN student st ON mb.member_seq = st.member_seq
        INNER JOIN enrollment en ON mb.member_seq = en.member_seq
        GROUP BY mb.name, mb.ssn, mb.tel, st.enroll_date
    )
    LOOP
        DBMS_OUTPUT.PUT_LINE(c."교육생 이름" || ' ' || c."주민번호 뒷자리" || ' ' || c."전화번호" || ' ' || c."등록일" || ' ' || c."수강신청 횟수");
    END LOOP;
END;

--특정 교육생 선택 출력(교육생이 수강 신청한 또는 수강중인, 수강했던 개설 과정 정보(과정명, 과정기간(시작 년월일, 끝 년월일), 강의실, 수료 및 중도탈락
여부, 수료 및 중도탈락 날짜)
CREATE OR REPLACE PROCEDURE student_find_p(p_seq IN student.student_seq%type)
IS
    v_curs_name  open_curs.curs_name%type;
    v_begin_date open_curs.begin_date%type;
    v_end_date   open_curs.end_date%type;
    v_room_name  room.room_name%type;
    v_drop_out   student.drop_out%type;
    v_drop_out_date student.drop_out_date%type;
BEGIN
    SELECT oc.curs_name, oc.begin_date, oc.end_date, r.room_name, st.drop_out, st.drop_out_date
    INTO v_curs_name, v_begin_date, v_end_date, v_room_name, v_drop_out, v_drop_out_date

```

```

FROM member mb
INNER JOIN student st ON mb.member_seq = st.member_seq
INNER JOIN enrollment en ON mb.member_seq = en.member_seq
INNER JOIN open_curs oc ON en.open_curs_seq = oc.open_curs_seq
INNER JOIN room r ON oc.room_seq = r.room_seq
WHERE st.student_seq = p_seq;

DBMS_OUTPUT.PUT_LINE('과정명: ' || v_curs_name);
DBMS_OUTPUT.PUT_LINE('과정기간(시작): ' || v_begin_date);
DBMS_OUTPUT.PUT_LINE('과정기간(끝): ' || v_end_date);
DBMS_OUTPUT.PUT_LINE('강의실: ' || v_room_name);
DBMS_OUTPUT.PUT_LINE('수료 및 중도탈락 여부: ' || v_drop_out);
DBMS_OUTPUT.PUT_LINE('수료 및 중도탈락 날짜: ' || v_drop_out_date);
END student_find_p;

--수료 및 중도 탈락 처리(날짜 입력)
-- 선언문
CREATE OR REPLACE PROCEDURE update_student_drop_out_p (
    p_student_seq IN NUMBER
)
IS

```

```

BEGIN
    UPDATE student SET drop_out_date = SYSDATE WHERE student_seq = p_student_seq;
    COMMIT;
    DBMS_OUTPUT.PUT_LINE('Student with ID ' || p_student_seq || ' has been marked as dropped out.');
```

EXCEPTION

```

    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Error updating drop out date for student with ID ' || p_student_seq || ': ' || SQLERRM);
END;
```

– 호출문

```

DECLARE
    v_student_seq NUMBER := 31;
BEGIN
    update_student_drop_out_p(v_student_seq);
END;
```

-- 강의 예정인 과정, 강의 중인 과정, 강의 종료된 과정 중에서 선택한 과정을 신청한 교육생 정보를 확인할 수 있다.

```

CREATE OR REPLACE PROCEDURE student_list_by_curriculum_p(
    p_curriculum_list_seq IN curriculum_list.curriculum_list_seq%TYPE,
    p_names OUT SYS.ODCIVARCHAR2LIST,
    p_ssns OUT SYS.ODCIVARCHAR2LIST,
    p_tels OUT SYS.ODCIVARCHAR2LIST,
    p_drop_outs OUT SYS.ODCIVARCHAR2LIST
```

```

)
IS
BEGIN
    SELECT mb.name, mb.ssn, mb.tel, st.drop_out
    BULK COLLECT INTO p_names, p_ssns, p_tels, p_drop_outs
    FROM curriculum_list cl
    INNER JOIN student st ON cl.curriculum_list_seq = st.curriculum_list_seq
    INNER JOIN member mb ON st.member_seq = mb.member_seq
    WHERE cl.curriculum_list_seq = p_curriculum_list_seq;
END student_list_by_curriculum_p;

DECLARE
    names SYS.ODCIVARCHAR2LIST;
    ssns SYS.ODCIVARCHAR2LIST;
    tels SYS.ODCIVARCHAR2LIST;
    drop_outs SYS.ODCIVARCHAR2LIST;
BEGIN
    student_list_by_curriculum_p(1, names, ssns, tels, drop_outs);

    FOR i IN 1..names.COUNT LOOP
        DBMS_OUTPUT.PUT_LINE('Name: ' || names(i));
        DBMS_OUTPUT.PUT_LINE('SSN: ' || ssns(i));
    END LOOP;
END;

```

```

        DBMS_OUTPUT.PUT_LINE('Tel: ' || tels(i));
        DBMS_OUTPUT.PUT_LINE('Drop Out: ' || drop_outs(i));
    END LOOP;
END;

-- 교육생 검색을 할 수 있다.
create or replace procedure student_select_p (p_student_seq in Student.student_seq%type)
is
begin
    For c In (
        select
            m.name as "이름",
            m.ssn as "주민번호뒷자리",
            m.tel as "전화번호",
            s.student_seq as "교육생번호",
            s.curriculum_list_seq as "과정번호",
            s.enroll_date as "등록일",
            s.drop_out as "교육생현재상태",
            s.drop_out_date as "수료및중도탈락날짜"
        from student s
        inner join member m on s.member_seq = m.member_seq
    )

```

```

        where s.student_seq = p_student_seq
    )
    LOOP
        DBMS_OUTPUT.PUT_LINE('이름: ' || c."이름" || ', 주민번호뒷자리: ' || c."주민번호뒷자리" || ', 전화번호: ' || c."전화번호" || ', 교육생번호: ' || c."교육생번호"
|| ', 과정번호: ' || c."과정번호" || ', 등록일: ' || c."등록일" || ', 교육생현재상태: ' || c."교육생현재상태" || ', 수료 및 중도탈락 날짜: ' || c."수료및중도탈락날짜");
    END LOOP;
end student_select_p;

--실행
begin
student_select_p(교육생번호);
end;
```

요구사항번호	B-06	작성일	2023-04-05
요구사항명	시험 관리 및 성적 조회	작성자	유동현
DML			
<p>1. 시험 관리</p> <p>- 특정 개설 과정 선택 시 등록된 개설 과목 정보를 출력하고, 과목별 성적 등록 여부를 확인할 수 있다.</p> <pre> CREATE OR REPLACE PROCEDURE get_subject_scores(open_curs_seq NUMBER) AS subject_name VARCHAR2(100); begin_date DATE; end_date DATE; score_status VARCHAR2(20); BEGIN FOR subject_record IN (SELECT DISTINCT sj.subject_name, os.begin_date, os.end_date, CASE WHEN ss.attend_score IS NOT NULL AND ss.test_writescore IS NOT NULL AND ss.test_realscore IS NOT NULL THEN '성적 등록 완료' ELSE '성적 등록 미완료' END AS score_status FROM curriculum_subject cs </pre>			

```

        INNER JOIN subject sj ON cs.subject_seq = sj.subject_seq
        INNER JOIN subject_score ss ON sj.subject_seq = ss.subject_seq
        INNER JOIN open_subject os ON sj.subject_seq = os.subject_seq
    WHERE cs.curriculum_list_seq = open_curs_seq
) LOOP
    subject_name := subject_record.subject_name;
    begin_date := subject_record.begin_date;
    end_date := subject_record.end_date;
    score_status := subject_record.score_status;

    DBMS_OUTPUT.PUT_LINE('과목명: ' || subject_name);
    DBMS_OUTPUT.PUT_LINE('과목기간(시작): ' || begin_date);
    DBMS_OUTPUT.PUT_LINE('과목기간(끝): ' || end_date);
    DBMS_OUTPUT.PUT_LINE('과목별 성적 등록 여부: ' || score_status);
    DBMS_OUTPUT.PUT_LINE('-----');
END LOOP;
END;

-- 실행
BEGIN
    get_subject_scores(개설과정번호);
END;
```


2. 성적 조회

- 개설 과목별, 교육생 개인별로 성적 정보를 출력할 수 있다.
- 개설 과목별 성적 출력 시 과정명, 과정 기간, 강의실명, 과목명, 교사명, 교재명 등을 출력하고, 해당 과목을 수강한 모든 교육생의 성적 정보(이름, 주민번호 뒷자리, 필기, 실기)를 같이 출력한다.

```
CREATE OR REPLACE PROCEDURE print_open_subject_columns
IS
BEGIN
    FOR c IN (SELECT DISTINCT
                cl.curs_name as "과정명",
                oc.begin_date as "과정기간(시작)",
                oc.end_date as "과정기간(끝)",
                sj.subject_name as "과목명",
                t.name as "교사명",
                b.subject as "교재명",
                mb.name as "교육생 이름",
                mb.ssn as "주민번호 뒷자리",
                ss.test_writescore as "필기",
                ss.test_realscore as "실기"
            FROM open_subject os
            INNER JOIN subject sj ON sj.subject_seq = os.subject_seq
            INNER JOIN curriculum_subject cs ON sj.subject_seq = cs.subject_seq
            INNER JOIN curriculum_list cl ON cs.curriculum_list_seq = cl.curriculum_list_seq
            INNER JOIN open_curs oc ON cl.curriculum_list_seq = oc.curriculum_list_seq
            INNER JOIN teacher t ON oc.teacher_seq = t.teacher_seq
```

```

        INNER JOIN subject_book sb ON sj.subject_seq = sb.subject_seq
        INNER JOIN book b ON sb.book_seq = b.book_seq
        INNER JOIN student st ON cl.curriculum_list_seq = st.curriculum_list_seq
        INNER JOIN member mb ON st.member_seq = mb.member_seq
        INNER JOIN subject_score ss ON st.student_seq = ss.student_seq)
    LOOP
        DBMS_OUTPUT.PUT_LINE('과정명: ' || c."과정명" || ', 과정기간(시작): ' || c."과정기간(시작)" || ', 과정기간(끝): ' || c."과정기간(끝)" || ', 과목명: ' || c."과목명" ||
        ', 교사명: ' || c."교사명" || ', 교재명: ' || c."교재명" || ', 교육생 이름: ' || c."교육생 이름" || ', 주민번호 뒷자리: ' || c."주민번호 뒷자리" || ', 필기: ' || c."필기" || ', 실기:
        ' || c."실기");
    END LOOP;
END;

```

3. 교육생 개인별 성적 출력 시 교육생 정보(이름, 주민번호 뒷자리, 과정명, 과정 기간, 강의실명 등)를 출력하고, 교육생이 수강한 모든 과목에 대한 성적 정보(과목명, 과목 기간, 교사명, 필기, 실기)를 같이 출력한다.

```
CREATE OR REPLACE PROCEDURE print_personal_score
```

```
IS
```

```

    v_name member.name%TYPE;
    v_ssn member.ssn%TYPE;
    v_tel member.tel%TYPE;
    v_curs_name curriculum_list.curs_name%TYPE;
    v_enroll_date student.enroll_date%TYPE;
    v_drop_out_date student.drop_out_date%TYPE;
    v_room_seq open_curs.room_seq%TYPE;
    v_subject_name subject.subject_name%TYPE;

```

```
v_begin_date open_subject.begin_date%TYPE;
v_end_date open_subject.end_date%TYPE;
v_teacher_name teacher.name%TYPE;
v_test_writescore subject_score.test_writescore%TYPE;
v_test_realscore subject_score.test_realscore%TYPE;
CURSOR c_results IS
    SELECT
        mb.name AS "이름",
        mb.ssn AS "주민번호 뒷자리",
        mb.tel AS "전화번호",
        cl.curs_name AS "과정명",
        st.enroll_date AS "과정 시작날짜",
        st.drop_out_date AS "과정 종료날짜",
        oc.room_seq AS "강의실",
        sj.subject_name AS "과목명",
        os.begin_date AS "과목 시작날짜",
        os.end_date AS "과목 종료날짜",
        tc.name AS "교사명",
        sc.test_writescore AS "필기시험점수",
        sc.test_realscore AS "실기시험점수"
    FROM
        student st
```

```

INNER JOIN member mb ON st.member_seq = mb.member_seq
INNER JOIN curriculum_list cl ON st.curriculum_list_seq = cl.curriculum_list_seq
LEFT OUTER JOIN open_curs oc ON st.curriculum_list_seq = oc.curriculum_list_seq
LEFT OUTER JOIN open_subject os ON oc.open_curs_seq = os.open_curs_seq AND oc.teacher_seq = os.teacher_seq
INNER JOIN subject sj ON os.subject_seq = sj.subject_seq
INNER JOIN teacher tc ON os.teacher_seq = tc.teacher_seq
INNER JOIN subject_score sc ON st.student_seq = sc.student_seq AND st.curriculum_list_seq = sc.curriculum_list_seq AND sj.subject_seq =
sc.subject_seq;
BEGIN
    OPEN c_results;

    LOOP
        FETCH c_results INTO v_name, v_ssn, v_tel, v_curs_name, v_enroll_date, v_drop_out_date, v_room_seq, v_subject_name, v_begin_date, v_end_date,
v_teacher_name, v_test_writescore, v_test_realscore;
        EXIT WHEN c_results%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('교육생명: ' || v_name || ', ' || '과목명: ' || v_subject_name || ', ' || '전화번호: ' || v_tel || ', ' || '과정명: ' || v_curs_name || ', ' ||
'등록일: ' || v_enroll_date || ', ' || '강의실: ' || v_room_seq || ', ' || '주민번호: ' || v_ssn || ', ' || '과목시작일: ' || v_begin_date || ', ' || '과목종료일: ' || v_end_date ||
', ' || '선생님이름: ' || v_teacher_name || ', ' || '필기시험점수: ' || v_test_writescore || ', ' || '실기시험점수: ' || v_test_realscore);
    END LOOP;

    CLOSE c_results;

```

```
END;
```

—실행

```
BEGIN
```

```
    print_personal_score;
```

```
END;
```

요구사항번호	B-07	작성일	2023-04-05
요구사항명	출결 관리 및 출결 조회	작성자	유동현
DML			
<p>1. 개설 과정 선택 시 모든 교육생의 출결을 조회할 수 있다.</p> <pre> CREATE OR REPLACE PROCEDURE get_attendance_info (p_open_curs_seq IN NUMBER) IS v_open_curs_seq open_curs.open_curs_seq%TYPE; v_student_seq student.student_seq%TYPE; v_name member.name%TYPE; v_attendance_date attendance.attendance_date%TYPE; BEGIN FOR attendance_info IN (SELECT oc.open_curs_seq, st.student_seq, mb.name, at.attendance_date FROM open_curs oc INNER JOIN student st ON oc.curriculum_list_seq = st.curriculum_list_seq INNER JOIN attendance at ON st.student_seq = at.student_seq INNER JOIN member mb ON mb.member_seq = st.member_seq WHERE oc.open_curs_seq = p_open_curs_seq order by st.student_seq </pre>			

```

) LOOP
    v_open_curs_seq := attendance_info.open_curs_seq;
    v_student_seq := attendance_info.student_seq;
    v_name := attendance_info.name;
    v_attendance_date := attendance_info.attendance_date;

    DBMS_OUTPUT.PUT_LINE('과정번호: ' || v_open_curs_seq || ', 교육생번호: ' || v_student_seq || ', 교육생이름: ' || v_name || ', 출석날짜: ' ||
v_attendance_date);
    END LOOP;
END;

```

– 실행

```

DECLARE
BEGIN
    get_attendance_info(개설과정번호 );
END;

```

2. 출결 현황을 기간별(년, 월, 일)로 조회할 수 있다.

```

CREATE OR REPLACE PROCEDURE get_attendance_date_info (
    p_open_curs_seq IN NUMBER,
    p_start_date IN DATE,
    p_end_date IN DATE

```

```

)
IS
    v_open_curs_seq open_curs.open_curs_seq%TYPE;
    v_student_seq student.student_seq%TYPE;
    v_name member.name%TYPE;
    v_attendance_date attendance.attendance_date%TYPE;
BEGIN
    FOR rec IN (
        SELECT oc.open_curs_seq, st.student_seq, mb.name, at.attendance_date
        FROM open_curs oc
        INNER JOIN student st ON oc.curriculum_list_seq = st.curriculum_list_seq
        INNER JOIN attendance at ON st.student_seq = at.student_seq
        INNER JOIN member mb ON mb.member_seq = st.member_seq
        WHERE oc.open_curs_seq = p_open_curs_seq AND at.attendance_date BETWEEN p_start_date AND p_end_date
    ) LOOP
        DBMS_OUTPUT.PUT_LINE('과정번호: ' || rec.open_curs_seq || ', 교육생번호: ' || rec.student_seq || ', 교육생이름: ' || rec.name || ', 출결날짜: ' ||
rec.attendance_date);
    END LOOP;
END;

— 실행
DECLARE

```



```
BEGIN
```

```
    get_attendance_date_info(<개설과정번호>,<'시작날짜'>, <'끝날짜'>);
```

```
END;
```

요구사항번호	B-08	작성일	2023-04-04
요구사항명	교육생 면접 및 선발	작성자	이채린
DML			
<p>1. 지원생 등록(면접을 지원한 지원생들의 이름, 지원일자, 면접 예정일자, 지원동기, 지원과정 등을 등록한다)</p> <pre> create or replace procedure MemberInterview_insert_p (p_member_seq in Member.member_seq%type, p_name in Member.name%type, p_ssn in Member.ssn%type, p_tel in Member.tel%type, p_interview_seq in Interview.interview_seq%type, p_motive in Interview.motive%type, p_enroll_subject in Interview.enroll_subject%type) is begin insert into Member(member_seq,name,ssn,tel) values(p_member_seq, p_name, p_ssn, p_tel); insert into Interview(interview_seq,member_seq,motive,enroll_subject) values(p_interview_seq, p_member_seq, p_motive, p_enroll_subject); commit; dbms_output.put_line('INSERT SUCCESS'); end MemberInterview_insert_p; </pre> <p>2. 교육생 선발</p> <pre> create or replace procedure Student_insert_p (p_seq in Student.student_seq%type, p_curriculum_list_seq in Student.curriculum_list_seq%type, p_enroll_date in Student.enroll_date%type, p_drop_out in Student.drop_out%type, p_drop_out_date in Student.drop_out_date%type, p_member_seq in Student.member_seq%type) is </pre>			

```
begin
  insert into Student(student_seq,curriculum_list_seq,enroll_date,drop_out,drop_out_date,member_seq) values(p_seq, p_curriculum_list_seq,
p_enroll_date, p_drop_out, p_drop_out_date, p_member_seq);
  commit;
  dbms_output.put_line('INSERT SUCCESS');
end Student_insert_p;
```

-실행

```
begin
```

```
Student_insert_p(교육생번호, 전체과정번호, 등록일, 교육생현재상태, 수료 및 중도탈락 날짜, 지원생번호);
```

```
end;
```

요구사항번호	B-09	작성일	2023-04-04
요구사항명	상담 관리	작성자	이채린
DML			
<pre>-- 특정 과정을 선택하고, 선택된 과정에 대해 작성된 상담 일지 전체 목록을 작성일자가 최신인 순으로 출력한다(작성일자, 상담 주제, 학생 이름, 상담 내용). CREATE OR REPLACE PROCEDURE counsel_select_p (p_open_curs_seq in open_curs.open_curs_seq%type) IS BEGIN FOR c IN (SELECT cs.counsel_date, tp.counsel_subject, m.name, cs.counsel_text FROM counsel cs INNER JOIN teacher t ON cs.teacher_seq = t.teacher_seq INNER JOIN open_curs o ON o.teacher_seq = t.teacher_seq INNER JOIN counsel_topic tp ON cs.counsel_topic_seq = tp.counsel_topic_seq INNER JOIN student s ON cs.student_seq = s.student_seq INNER JOIN member m ON s.member_seq = m.member_seq WHERE o.open_curs_seq = p_open_curs_seq ORDER BY cs.counsel_date DESC) LOOP DBMS_OUTPUT.PUT_LINE('작성일자: ' c.counsel_date ', 상담주제: ' c.counsel_subject ', 학생이름: ' c.name ', 상담내용: ' c.counsel_text); END LOOP;</pre>			

```
END counsel_select_p;
```

-호출

```
BEGIN
```

```
  counsel_select_p(<과정번호>);
```

```
END;
```

-- 지정된 과정의 전체 상담 일지를 수강생별로 조회할 수 있다.

```
CREATE OR REPLACE PROCEDURE counsel_student_select_p (p_open_curs_seq in open_curs.open_curs_seq%type, p_name in member.name%type)
```

```
IS
```

```
BEGIN
```

```
  FOR c IN (
```

```
    SELECT
```

```
      cs.counsel_date,
```

```
      tp.counsel_subject,
```

```
      m.name,
```

```
      cs.counsel_text
```

```
    FROM counsel cs
```

```
    INNER JOIN teacher t ON cs.teacher_seq = t.teacher_seq
```

```
    INNER JOIN open_curs o ON o.teacher_seq = t.teacher_seq
```

```
    INNER JOIN counsel_topic tp ON cs.counsel_topic_seq = tp.counsel_topic_seq
```

```
    INNER JOIN student s ON cs.student_seq = s.student_seq
```

```
    INNER JOIN member m ON s.member_seq = m.member_seq
```

```
WHERE o.open_curs_seq = p_open_curs_seq AND m.name = p_name
      ORDER BY cs.counsel_date DESC
)
LOOP
  DBMS_OUTPUT.PUT_LINE('작성일자: ' || c.counsel_date || ', 상담주제: ' || c.counsel_subject || ', 학생이름: ' || c.name || ', 상담내용: ' || c.counsel_text);
END LOOP;
END counsel_student_select_p;

--호출
begin
counsel_student_select_p(<과정번호>, '<학생이름>');
end;
```

요구사항번호	B-10	작성일	2023-04-04
요구사항명	과목별 교재 관리	작성자	이채린
DML			
<p>1. 교재 등록</p> <p>- 기초 정보 교재 목록에 교재를 추가할 수 있다.</p> <p>create or replace procedure Book_insert_p (p_seq in Book.book_seq%type, p_subject in Book.subject%type, p_author in Book.author%type, p_publisher in Book.publisher%type, p_pub_year in Book.pub_year%type)</p> <p>is</p> <p>begin</p> <p> insert into Book(book_seq,subject,author,publisher,pub_year) values(p_seq, p_subject, p_author, p_publisher, p_pub_year);</p> <p> commit;</p> <p> dbms_output.put_line('INSERT SUCCESS');</p> <p>end Book_insert_p;</p> <p>- 개설 과목에 교재 등록 시 기초정보 교재 목록에서 선택적으로 추가할 수 있다.</p> <p>create or replace procedure Subject_Book_insert_p (p_subject_seq in Subject_Book.subject_seq%type, p_book_seq in Subject_Book.book_seq%type)</p> <p>is</p> <p>begin</p> <p> insert into Subject_Book(subject_seq,book_seq) values (p_subject_seq, p_book_seq);</p> <p> commit;</p> <p> dbms_output.put_line('INSERT SUCCESS');</p>			

```
end;
```

2. 교재 조회

```
CREATE OR REPLACE PROCEDURE Subject_Book_select_p
```

```
IS
```

```
BEGIN
```

```
  FOR c IN (
```

```
    SELECT
```

```
      b.book_seq,
```

```
      s.subject_name,
```

```
      b.subject,
```

```
      b.author,
```

```
      b.publisher,
```

```
      b.pub_year
```

```
    FROM book b
```

```
    INNER JOIN subject_book sb ON b.book_seq = sb.book_seq
```

```
    INNER JOIN subject s ON sb.subject_seq = s.subject_seq
```

```
  )
```

```
  LOOP
```

```
    DBMS_OUTPUT.PUT_LINE('책 번호: ' || c.book_seq || ', 과목: ' || c.subject_name || ', 교재명: ' || c.subject || ', 저자: ' || c.author || ', 출판사: ' || c.publisher ||  
, 출판연도: ' || c.pub_year);
```

```
  END LOOP;
```



```
END Subject_Book_select_p;
```

3. 교재 수정

```
CREATE OR REPLACE PROCEDURE book_update (book_seq IN NUMBER, subject in VARCHAR2, author in VARCHAR2, publisher in VARCHAR2, pub_year in VARCHAR2)
```

```
IS
```

```
BEGIN
```

```
    UPDATE book
```

```
    SET
```

```
    subject = subject,
```

```
    author = author,
```

```
    publisher = publisher,
```

```
    pub_year = pub_year
```

```
    WHERE book_seq = book_seq;
```

```
    EXCEPTION WHEN OTHERS THEN
```

```
        DBMS_OUTPUT.put_line('수정에 실패했습니다.');
```

```
        ROLLBACK;
```

```
END book_update;
```

4. 교재 삭제

```
CREATE OR REPLACE PROCEDURE book_delete_p (
```

```
    p_seq in book.book_seq%type
```

```
)  
is  
begin  
  delete from book  
  where book_seq = p_seq;  
  commit;  
end book_delete_p;
```

요구사항번호	B-11	작성일	2023-04-04
요구사항명	수료생 취업 활동 관리	작성자	조연우
DML			
<p>1. 사후처리 등록</p> <p>- 선언</p> <pre> CREATE OR REPLACE PROCEDURE job_current_insert_p(pjob_field_seq job_current.job_field_seq%type, pstudent_seq job_current.student_seq%type, pcompany job_current.company%type, phire_date job_current.hire_date%type) IS BEGIN INSERT INTO job_current VALUES (jfSeq.nextVal, pjob_field_seq, pstudent_seq, pcompany, phire_date); END job_current_insert_p; / - 호출 BEGIN job_current_insert_p(4, 501, '카카오', '2021-10-31'); END; / </pre>			

2. 사후처리 조회 (교육생번호, 교육생, 교육과정 정보, 수료날짜, 취업현황)

- 선언

```
CREATE OR REPLACE PROCEDURE job_current_select_p
IS
BEGIN
    FOR c IN (
        SELECT
            s.student_seq AS "학생번호",
            m.name AS "이름",
            c.curs_name AS "과정명",
            f.field_name AS "분야",
            jc.company AS "회사명",
            jc.hire_date AS "입사일"
        FROM job_current jc
            INNER JOIN student s
                ON jc.student_seq = s.student_seq
            INNER JOIN Member m
                ON m.member_seq = s.member_seq
            INNER JOIN curriculum_list c
                ON s.curriculum_list_seq = c.curriculum_list_seq
            INNER JOIN job_field f
                ON jc.job_field_seq = f.job_field_seq
        WHERE jc.company IS NOT NULL
    )
```

```

LOOP
  DBMS_OUTPUT.PUT_LINE('학생번호: ' || c.학생번호);
  DBMS_OUTPUT.PUT_LINE('이름: ' || c.이름);
  DBMS_OUTPUT.PUT_LINE('과정명: ' || c.과정명);
  DBMS_OUTPUT.PUT_LINE('분야: ' || c.분야);
  DBMS_OUTPUT.PUT_LINE('회사명: ' || c.회사명);
  DBMS_OUTPUT.PUT_LINE('입사일: ' || c.입사일);
  DBMS_OUTPUT.PUT_LINE('');
END LOOP;
END job_current_select_p;
/

```

- 호출

```

BEGIN
  job_current_select_p;
END;
/

```

3. 관리자는 취업현황에 대한 재취업 지원 시스템을 제공하기 위해 미취업 교육생 목록을 열람할 수 있다. 단, 재취업 지원은 수료 후 12개월 이내의 교육생들에 한해서 지원한다,

— 선언

```

CREATE OR REPLACE PROCEDURE job_needed_select_p
IS
BEGIN
  FOR c IN (

```

```

SELECT
    s.student_seq AS "학생번호",
    m.name AS "이름",
    c.curs_name AS "과정명",
    f.field_name AS "분야",
    jc.company AS "회사명",
    jc.hire_date AS "입사일"
FROM job_current jc
    LEFT OUTER JOIN student s
        ON jc.student_seq = s.student_seq
    LEFT OUTER JOIN Member m
        ON m.member_seq = s.member_seq
    LEFT OUTER JOIN curriculum_list c
        ON s.curriculum_list_seq = c.curriculum_list_seq
    LEFT OUTER JOIN job_field f
        ON jc.job_field_seq = f.job_field_seq
WHERE jc.company IS NULL
    AND MONTHS_BETWEEN(SYSDATE, s.drop_out_date) <= 6
)
LOOP
DBMS_OUTPUT.PUT_LINE('학생번호: ' || c.학생번호);
DBMS_OUTPUT.PUT_LINE('이름: ' || c.이름);
DBMS_OUTPUT.PUT_LINE('과정명: ' || c.과정명);
DBMS_OUTPUT.PUT_LINE('분야: ' || c.분야);
DBMS_OUTPUT.PUT_LINE('회사명: ' || c.회사명);
DBMS_OUTPUT.PUT_LINE('입사일: ' || c.입사일);

```

```

    DBMS_OUTPUT.PUT_LINE('');
  END LOOP;
END job_needed_select_p;
/

```

- 호출

```

BEGIN
  job_needed_select_p;
END;
/

```

4. 관리자는 취업현황에 대한 수정, 삭제 기능을 사용할 수 있다.

- 직업 분야 수정

```

CREATE OR REPLACE PROCEDURE job_current_field_update_p (
  pfield IN job_current.job_field_seq%type,
  pstudent IN job_current.student_seq%type
)
IS
BEGIN
  UPDATE job_current SET job_field_seq = pfield WHERE student_seq = pstudent;
  DBMS_OUTPUT.PUT_LINE('수정됐습니다. ');
  COMMIT;
  EXCEPTION WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('수정에 실패했습니다. ');
    rollback;

```

```
END job_current_field_update_p;
```

```
/
```

- 회사명 수정

```
CREATE OR REPLACE PROCEDURE job_current_company_update_p (
```

```
    pcompany IN job_current.company%type,
```

```
    pstudent IN job_current.student_seq%type
```

```
)
```

```
IS
```

```
BEGIN
```

```
    UPDATE job_current SET company = pcompany WHERE student_seq = pstudent;
```

```
    DBMS_OUTPUT.PUT_LINE('수정됐습니다.');
```

```
    COMMIT;
```

```
    EXCEPTION WHEN OTHERS THEN
```

```
        DBMS_OUTPUT.PUT_LINE('수정에 실패했습니다.');
```

```
        rollback;
```

```
END job_current_company_update_p;
```

```
/
```

- 입사일 수정

```
CREATE OR REPLACE PROCEDURE job_current_hiredate_update_p (
```

```
    phire IN job_current.hire_date%type,
```

```
    pstudent IN job_current.student_seq%type
```

```
)
```

```
IS
```



```

BEGIN
    UPDATE job_current SET hire_date = phire WHERE student_seq = pstudent;
    DBMS_OUTPUT.PUT_LINE('수정됐습니다.');
```

COMMIT;

```

    EXCEPTION WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('수정에 실패했습니다.');
```

rollback;

```

END job_current_hiredate_update_p;
/
```

- 삭제

```

CREATE OR REPLACE PROCEDURE job_current_delete_p(
    pstudent_seq job_current.student_seq%type
)
IS
BEGIN
    DELETE FROM job_current WHERE student_seq = pstudent_seq;
END job_current_delete_p;
```

요구사항번호	B-12	작성일	2023-04-04
요구사항명	협력 기업 관리	작성자	조연우
DML			
<p>- 협력 기업 등록</p> <pre> CREATE OR REPLACE PROCEDURE job_announce_insert_p(pcity IN job_announce.city_seq%type, pjob_field IN job_announce.job_field_seq%type, pcontract_type IN job_announce.contract_type_seq%type, pcompany IN job_announce.company%type, ppeople IN job_announce.people%type, pbegindate IN job_announce.begin_date%type, penddate IN job_announce.end_date%type) IS BEGIN INSERT into job_announce VALUES (jobannounceseq.nextVal, pcity, pjob_field, pcontract_type, pcompany, ppeople, pbegindate, penddate); DBMS_OUTPUT.PUT_LINE('등록됐습니다.');</pre> <pre> COMMIT; EXCEPTION WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE('등록에 실패했습니다.');</pre> <pre> ROLLBACK;</pre>			

```
END job_announce_insert_p;
```

```
/
```

- 협력 기업 조회

- 선언

```
CREATE OR REPLACE PROCEDURE job_announce_select_p
```

```
IS
```

```
BEGIN
```

```
  FOR c IN (
```

```
    SELECT
```

```
    job_announce_seq AS 공고번호,
```

```
    c.city_name AS 지역,
```

```
    ct.contract_type AS 계약형태,
```

```
    f.field_name AS 분야,
```

```
    a.company AS 회사명,
```

```
    a.begin_date AS 모집시작일,
```

```
    a.end_date AS 모집종료일
```

```
  FROM job_announce a
```

```
    INNER JOIN city c
```

```
      ON c.city_seq = a.city_seq
```

```
    INNER JOIN job_field f
```

```
      ON a.job_field_seq = f.job_field_seq
```

```
    INNER JOIN contract_type ct
```

```
      ON a.contract_type_seq = ct.contract_type_seq
```

```

order by job_announce_seq
)
LOOP
  DBMS_OUTPUT.PUT_LINE('공고번호: ' || c.공고번호);
  DBMS_OUTPUT.PUT_LINE('지역: ' || c.지역);
  DBMS_OUTPUT.PUT_LINE('계약형태: ' || c.계약형태);
  DBMS_OUTPUT.PUT_LINE('분야: ' || c.분야);
  DBMS_OUTPUT.PUT_LINE('회사명: ' || c.회사명);
  DBMS_OUTPUT.PUT_LINE('모집시작일: ' || c.모집시작일);
  DBMS_OUTPUT.PUT_LINE('모집종료일: ' || c.모집종료일);
  DBMS_OUTPUT.PUT_LINE('');
END LOOP;
END job_announce_select_p;
/

- 호출
BEGIN
  job_announce_select_p;
END;
/

- 협력 기업 수정
CREATE OR REPLACE PROCEDURE job_announce_update_p(
  pannounce_seq IN job_announce.job_announce_seq%type,
  pcompany IN job_announce.company%type

```

```

)
IS
BEGIN
    UPDATE job_announce SET company = pcompany WHERE job_announce_seq = pannounce_seq;
    DBMS_OUTPUT.PUT_LINE('수정됐습니다.');
```

COMMIT;

```

    EXCEPTION WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('수정에 실패했습니다.');
```

ROLLBACK;

```

END job_announce_update_p;
/
```

- 협력 기업 삭제

```

CREATE OR REPLACE PROCEDURE job_announce_delete_p(
    pannounce_seq IN job_announce.job_announce_seq%type
)
IS
BEGIN
    DELETE FROM job_announce WHERE job_announce_seq = pannounce_seq;
    DBMS_OUTPUT.PUT_LINE('삭제됐습니다.');
```

COMMIT;

```
EXCEPTION WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('삭제에 실패했습니다.');
```

ROLLBACK;

```
END job_announce_delete_p;
/
```

요구사항번호	B-13	작성일	2023-04-04
요구사항명	추천 도서 관리	작성자	김대환
DML			
<pre>--1. 추천 도서 추가(추천도서번호(자동생성), 과목번호, 교사번호, 제목) -- 선언문 create or replace procedure comment_book_insert_p (p_subject_seq in comment_book.subject_seq%type, p_teacher_seq in comment_book.teacher_seq%type, p_book_name in comment_book.book_name%type) is begin insert into comment_book(comment_book_seq, subject_seq, teacher_seq, book_name) values (cbook_seq.nextVal, p_subject_seq, p_teacher_seq, p_book_name); commit; dbms_output.put_line('insert success'); end comment_book_insert_p; -- 호출문 DECLARE</pre>			

```

v_subject_seq comment_book.subject_seq%TYPE := 123;
v_teacher_seq comment_book.teacher_seq%TYPE := 456;
v_book_name comment_book.book_name%TYPE := 'My Book';
BEGIN
comment_book_insert_p(
    p_subject_seq => v_subject_seq,
    p_teacher_seq => v_teacher_seq,
    p_book_name => v_book_name
);
END;

--2. 추천 도서 수정하기
-- 선언문
CREATE OR REPLACE PROCEDURE comment_book_update_p(
    p_comment_book_seq comment_book.comment_book_seq%TYPE,
    p_book_name comment_book.book_name%TYPE
) IS
BEGIN
    UPDATE comment_book
    SET book_name = p_book_name
    WHERE comment_book_seq = p_comment_book_seq;

    COMMIT;

    DBMS_OUTPUT.PUT_LINE('Update success');
END comment_book_update_p;

```


— 호출문

BEGIN

comment_book_update_p(p_comment_book_seq => 121, p_book_name => '자바 개론');

END;

--3. 추천 도서 삭제하기

—선언문

CREATE OR REPLACE PROCEDURE comment_book_delete_p(

p_comment_book_seq comment_book.comment_book_seq%TYPE

)

IS

BEGIN

delete from comment_book where comment_book_seq = p_comment_book_seq;

commit;

END comment_book_delete_p;

— 호출문

DECLARE

v_comment_book_seq comment_book.comment_book_seq%TYPE := 10;

BEGIN

comment_book_delete_p(

p_comment_book_seq => v_comment_book_seq

);

END;

--4. 추천 도서 조회하기

— 선언하기

```
CREATE OR REPLACE PROCEDURE comment_book_select_p
```

```
IS
```

```
BEGIN
```

```
  FOR c IN (
```

```
    SELECT subject_seq, teacher_seq, book_name
```

```
    FROM comment_book
```

```
  )
```

```
  LOOP
```

```
    DBMS_OUTPUT.PUT_LINE('과목번호: ' || c.subject_seq || ', 교사번호: ' || c.teacher_seq || ', 책제목: ' || c.book_name);
```

```
  END LOOP;
```

```
END comment_book_select_p;
```

— 호출하기

```
BEGIN
```

```
  comment_book_select_p;
```

```
END;
```

요구사항번호	B-14	작성일	2023-04-05
요구사항명	만남의 광장 관리	작성자	김대환
DML			
<pre>--1. 게시물 조회 -- 선언문 CREATE OR REPLACE PROCEDURE board_select_p IS BEGIN FOR c IN (SELECT b.board_seq, b.student_seq, b.board_title, b.board_content, br.reply_seq, br.student_seq AS reply_student_seq, br.reply_content from board b inner join board_reply br on b.board_seq = br.board_seq) LOOP DBMS_OUTPUT.PUT_LINE('게시물번호: ' c.board_seq ', 교육생번호: ' c.student_seq ', 게시글 제목: ' c.board_title ', 게시글 본문: ' c.board_content ', 댓글번호: ' c.reply_seq ', 교육생번호(댓글): ' c.reply_student_seq ', 댓글내용: ' c.reply_content); END LOOP; END board_select_p; -- 호출문 begin board_select_p; end;</pre>			

--2. 게시물 수정

— 선언문

```
CREATE OR REPLACE PROCEDURE board_update_p (  
    p_board_seq board.board_seq%type,  
    p_board_title board.board_title%type  
)
```

IS

BEGIN

update board

set board_title = p_board_title

where board_seq = p_board_seq;

COMMIT;

DBMS_OUTPUT.PUT_LINE('Update success');

END board_update_p;

— 호출문

BEGIN

board_update_p(p_board_seq => 13, p_board_title => '수정할 제목');

END;

--3. 게시물 삭제

— 선언문

```
CREATE OR REPLACE PROCEDURE board_delete_p (  

```

```

    p_board_seq board.board_seq%type
)
IS
BEGIN
    delete from board where board_seq = p_board_seq;
    commit;
END board_delete_p;

```

— 호출문

```

DECLARE
    v_board_seq board.board_seq%TYPE := 10;
BEGIN
    board_delete_p(
        p_board_seq => v_board_seq
    );
END;

```

요구사항번호	B-15	작성일	2023-04-04
요구사항명	달란트 시장 관리	작성자	이동재
DML			
<pre> 1. 퀴즈 관리 --입력 create or replace procedure QUIZ_insert (pseq in number, pquiz_title in varchar2, pquiz_content in varchar2, psubject_seq in number, pquiz_answer in number) is begin insert into QUIZ values(pseq, pquiz_title, pquiz_content, psubject_seq, pquiz_answer); commit; dbms_output.put_line('INSERT SUCCESS'); end QUIZ_insert; --수정 CREATE OR REPLACE PROCEDURE quiz_update (pquiz_seq IN NUMBER, pquiz_title IN VARCHAR2) IS BEGIN UPDATE quiz SET quiz_title = pquiz_title WHERE quiz_seq = pquiz_seq; EXCEPTION WHEN OTHERS THEN DBMS_OUTPUT.put_line('수정에 실패했습니다.');</pre>			
ROLLBACK;			

```

END quiz_update;

--삭제
CREATE OR REPLACE PROCEDURE quiz_delete(pquiz_seq IN NUMBER)
IS
BEGIN
    DELETE FROM quiz
    WHERE quiz_seq = pquiz_seq;
    COMMIT;
END quiz_delete;

--조회
CREATE OR REPLACE PROCEDURE quiz_info IS
BEGIN
    FOR c IN (
        SELECT
            quiz_seq as "퀴즈번호",
            quiz_title as "퀴즈제목",
            quiz_content as "퀴즈내용",
            subject_seq as "과목번호",
            quiz_answer as "정답번호"
        FROM quiz)
    LOOP
        DBMS_OUTPUT.PUT_LINE('퀴즈번호: ' || c.퀴즈번호);
        DBMS_OUTPUT.PUT_LINE('퀴즈제목: ' || c.퀴즈제목);
        DBMS_OUTPUT.PUT_LINE('퀴즈내용: ' || c.퀴즈내용);
    
```

```
DBMS_OUTPUT.PUT_LINE('과목번호: ' || c.과목번호);  
DBMS_OUTPUT.PUT_LINE('정답번호: ' || c.정답번호);  
END LOOP;  
END quiz_info;
```

2. 포인트 관리

--입력

```
create or replace procedure point_insert (ppoint_seq in number, pstudent_seq in number, pppoint in number)  
is  
begin  
    insert into point values (ppoint_seq, pstudent_seq, pppoint);  
    commit;  
    dbms_output.put_line('INSERT SUCCESS');  
end point_insert;
```

--수정

```
CREATE OR REPLACE PROCEDURE point_update (pppoint in number, pstudent_seq in number)  
IS  
BEGIN  
    update point  
    set  
    ppoint = pppoint  
    where student_seq = pstudent_seq;  
    EXCEPTION WHEN OTHERS THEN  
    DBMS_OUTPUT.put_line('수정에 실패했습니다.');
```



```

ROLLBACK;
END point_update;

--삭제
CREATE OR REPLACE PROCEDURE point_delete(pstudent_seq IN NUMBER)
IS
BEGIN
    delete from point
    where student_seq = pstudent_seq;
    COMMIT;
END point_delete;

--조회
CREATE OR REPLACE PROCEDURE point_info
IS
BEGIN
    FOR c IN (
        SELECT
            student_seq as "학생번호",
            ppoint as "보유포인트"
        FROM point)
    LOOP
        DBMS_OUTPUT.PUT_LINE('학생번호: ' || c.학생번호);
        DBMS_OUTPUT.PUT_LINE('보유 포인트: ' || c.보유포인트);
    END LOOP;
END point_info;

```

3. 상품 관리

--입력

```
create or replace procedure prize_insert (pprize_seq in number, pprize_name in varchar2, pprize_point in number)
```

```
is
```

```
begin
```

```
    insert into PRIZE values (pprize_seq, pprize_name, pprize_point);
```

```
    commit;
```

```
    dbms_output.put_line('INSERT SUCCESS');
```

```
end prize_insert;
```

--수정

```
CREATE OR REPLACE PROCEDURE prize_update (pprize_point in number, pprize_seq in number)
```

```
IS
```

```
BEGIN
```

```
    update prize
```

```
    set prize_point = pprize_point
```

```
    where prize_seq = pprize_seq;
```

```
    EXCEPTION WHEN OTHERS THEN
```

```
        DBMS_OUTPUT.put_line('수정에 실패했습니다.');
```

```
        ROLLBACK;
```

```
END prize_update;
```

--삭제

```
CREATE OR REPLACE PROCEDURE prize_delete(pprize_seq IN NUMBER)
```

```
IS
BEGIN
    delete from prize
    where prize_seq = pprize_seq;
    COMMIT;
END prize_delete;

--조회
CREATE OR REPLACE PROCEDURE prize_info IS
BEGIN
    FOR c IN (SELECT
                prize_name as "상품명",
                prize_point as "상품가격"
                FROM prize)
    LOOP
        DBMS_OUTPUT.PUT_LINE('상품명: ' || c.상품명);
        DBMS_OUTPUT.PUT_LINE('상품가격: ' || c.상품가격);
    END LOOP;
END prize_info;
```

교사 계정

요구사항번호	C-00	작성일	2023-04-05
요구사항명	교사 로그인	작성자	1조
DML			
<pre> CREATE OR REPLACE PROCEDURE teacher_P (p_ssn IN teacher.ssn%TYPE) IS v_teacher_seq teacher.teacher_seq%TYPE; v_teacher_name teacher.name%TYPE; v_teacher_tel teacher.tel%TYPE; BEGIN SELECT teacher_seq, name, tel INTO v_teacher_seq, v_teacher_name, v_teacher_tel FROM teacher WHERE ssn = p_ssn; DBMS_OUTPUT.PUT_LINE('선생님 번호: ' v_teacher_seq ', 선생님 이름: ' v_teacher_name ', 선생님 전화번호: ' v_teacher_tel); END teacher_P; </pre>			

요구사항번호	C-01	작성일	2023-04-05
요구사항명	강의 스케줄 조회	작성자	이동재
DML			
<pre> --1. 강의 예정 CREATE OR REPLACE PROCEDURE curriculum_info IS BEGIN FOR c IN (SELECT * FROM Curriculum_list WHERE curriculum_list_Seq BETWEEN 7 AND 12) LOOP DBMS_OUTPUT.PUT_LINE('과정 번호: ' c.curriculum_list_seq); DBMS_OUTPUT.PUT_LINE('과정명: ' c.curs_name); END LOOP; END curriculum_info; --2. 강의 중 CREATE OR REPLACE PROCEDURE open_curs_info IS BEGIN FOR c IN (SELECT c.curriculum_list_seq as "과정번호", c.curs_name as "과정명", o.room_seq as "강의실", o.teacher_seq as "담당선생님번호", o.begin_date as "시작날짜", o.end_date as "종료날짜", </pre>			

```

        o.student_limit as "수강인원"
    FROM Curriculum_list c
    INNER JOIN open_curs o
        ON c.curriculum_list_seq = o.curriculum_list_seq)
LOOP
    -- 결과를 출력합니다.
    DBMS_OUTPUT.PUT_LINE('과정번호: ' || c.과정번호);
    DBMS_OUTPUT.PUT_LINE('과정명: ' || c.과정명);
    DBMS_OUTPUT.PUT_LINE('강의실: ' || c.강의실);
    DBMS_OUTPUT.PUT_LINE('담당선생님번호: ' || c.담당선생님번호);
    DBMS_OUTPUT.PUT_LINE('시작날짜: ' || c.시작날짜);
    DBMS_OUTPUT.PUT_LINE('종료날짜: ' || c.종료날짜);
    DBMS_OUTPUT.PUT_LINE('수강인원: ' || c.수강인원);
END LOOP;
END open_curs_info;

--3. 강의 종료
CREATE OR REPLACE PROCEDURE no_open_curs IS
BEGIN
    -- 결과를 출력합니다.
    FOR rec IN (SELECT c.curriculum_list_seq, c.curs_name
        FROM Curriculum_list c
        LEFT OUTER JOIN open_curs o
            ON c.curriculum_list_seq = o.curriculum_list_seq
        WHERE c.curriculum_list_Seq BETWEEN 13 AND 21
    )

```

```
ORDER BY c.curriculum_list_seq )
```

```
LOOP
```

```
  DBMS_OUTPUT.PUT_LINE('과정번호: ' || rec.curriculum_list_seq);
```

```
  DBMS_OUTPUT.PUT_LINE('과정명: ' || rec.curs_name);
```

```
END LOOP;
```

```
END no_open_curs;
```


요구사항번호	C-02	작성일	2023-04-04
요구사항명	배점 입출력	작성자	조연우
DML			
<p>1. 강의를 마친 과목에 대한 성적 처리를 위한 배점 입출력</p> <p>-교사는 자신이 특정 과목을 선택하고 해당 과목의 배점 정보를 출결, 필기, 실기로 구분해 등록할 수 있어야 한다. 시험 날짜, 시험 문제를 추가할 수 있어야 한다. (단, 출결은 최소 20점 이상이어야 하고, 출결, 필기, 실기의 합은 100점이 되도록 한다.)</p> <p>- 선언</p> <pre> CREATE OR REPLACE PROCEDURE score_insert_p(psubject IN score.subject_seq%type, pattend IN score.score_attend%type, pwritetest IN score.score_writetest%type, prealtest IN score.score_realtest%type) IS BEGIN INSERT INTO score VALUES (score_seq.nextVal, psubject, pattend, pwritetest, prealtest); DBMS_OUTPUT.PUT_LINE('등록에 성공했습니다.');</pre> <pre> COMMIT; EXCEPTION WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE('등록에 실패했습니다.');</pre> <pre> ROLLBACK; END score_insert_p;</pre>			

- 호출

```
BEGIN
  score_insert_p(42, 31, 31, 31);
END;
/
```

-출결, 필기, 실기의 배점 비중은 담당 교사가 과목별로 결정한다. (단, 출결은 최소 20점 이상이어야 하고, 출결, 필기, 실기의 합은 100점이 되도록 한다.)

```
CREATE OR REPLACE PROCEDURE test_insert_p(
  psubject IN test.subject_seq%type,
  ptype IN test.test_type%type,
  pname IN test.test_name%type
)
IS
BEGIN
  INSERT INTO test VALUES (test_seq.nextVal, psubject, ptype, pname);
  DBMS_OUTPUT.PUT_LINE('등록에 성공했습니다.');
```

COMMIT;

EXCEPTION WHEN OTHERS THEN

DBMS_OUTPUT.PUT_LINE('등록에 실패했습니다.');

ROLLBACK;

```
END test_insert_p;
/
```

- 호출

```
BEGIN
```

```
test_insert_p(42, '테스트', '테스트');
```

```
END;
```

```
/
```

2. 과목 목록 출력

- 과목 목록 출력 시 과목번호, 과정명, 과정기간(시작 연월일, 끝 연월일), 강의실, 과목명, 과목기간(시작 연월일, 끝 연월일), 교재명, 출결, 필기, 실기 배점 등 출력.

- 선언

```
CREATE OR REPLACE PROCEDURE all_subject_selecct_p
```

```
IS
```

```
BEGIN
```

```
FOR c IN (
```

```
select
```

```
o.subject_seq as "과목번호",
```

```
c.curs_name as "과정명",
```

```
c.begin_date as "과정기간(시작)",
```

```
c.end_date as "과정기간(종료)",
```

```
room_seq as "강의실",
```

```
s.subject_name as "과목이름",
```

```
o.begin_date as "과목시작일",
```

```
o.end_date as "과목종료일",
```

```
b.subject as "교재",
```

```
e.score_attend as "출석점수",
```

```

e.score_writeTest as "필기점수",
e.score_realTest as "실기점수"
from subject s
  inner join open_subject o
    on s.subject_seq = o.subject_seq
    inner join open_curs c
      on c.open_curs_seq = o.open_curs_seq
      inner join subject_book t
        on t.subject_seq = s.subject_seq
        inner join book b
          on t.book_seq = b.book_seq
          inner join score e
            on e.subject_seq = s.subject_seq
)
LOOP
DBMS_OUTPUT.PUT_LINE('과목번호: ' || c.과목번호);
DBMS_OUTPUT.PUT_LINE('과정명: ' || c.과정명);
DBMS_OUTPUT.PUT_LINE('과정기간(시작): ' || c."과정기간(시작)");
DBMS_OUTPUT.PUT_LINE('과정기간(종료): ' || c."과정기간(종료)");
DBMS_OUTPUT.PUT_LINE('강의실: ' || c.강의실);
DBMS_OUTPUT.PUT_LINE('과목이름: ' || c.과목이름);
DBMS_OUTPUT.PUT_LINE('과목시작일: ' || c.과목시작일);
DBMS_OUTPUT.PUT_LINE('과목종료일: ' || c.과목종료일);
DBMS_OUTPUT.PUT_LINE('교재: ' || c.교재);
DBMS_OUTPUT.PUT_LINE('출석점수: ' || c.출석점수);
DBMS_OUTPUT.PUT_LINE('필기점수: ' || c.필기점수);

```

```
DBMS_OUTPUT.PUT_LINE('실기점수: ' || c.실기점수);  
DBMS_OUTPUT.PUT_LINE('');  
END LOOP;  
END all_subject_selecct_p;  
/  

```

- 출력

```
BEGIN  
    all_subject_selecct_p;  
END;  
/  

```

요구사항번호	C-03	작성일	2023-04-05
요구사항명	성적 입출력	작성자	이채린
DML			
<p>-교사는 자신이 강의를 마친 과목의 목록 중에서 특정 과목을 선택하면, 교육생 정보가 출력된다.</p> <pre> create or replace procedure score_select_p (p_subject_seq in Subject_score.subject_seq%type, p_teacher_seq in Teacher.teacher_seq%type) is begin For c In (select s.subject_seq, d.student_seq, m.name, r.attend_score, r.test_writescore, r.test_realscore from subject s inner join teaching_subject e on s.subject_seq = e.subject_seq inner join teacher t on t.teacher_seq = e.teacher_seq inner join subject_score r on r.subject_seq = s.subject_seq inner join student d on d.student_seq = r.student_seq inner join member m on d.member_seq = m.member_seq where s.subject_seq = p_subject_seq /*과목번호*/ and t.teacher_seq = p_teacher_seq/*선생번호*/) LOOP DBMS_OUTPUT.PUT_LINE('과목번호: ' c.subject_seq ', 학생번호: ' c.student_seq ', 학생이름: ' c.name ', 출결점수: ' c.attend_score ', </pre>			

```
필기점수: ' || c.test_writescore || ', 실기점수: ' || c.test_realscore);
```

```
END LOOP;
```

```
END score_select_p;
```

```
--호출
```

```
--set serveroutput on;
```

```
begin
```

```
score_select_p(3, 6);
```

```
end;
```

-특정 교육생 정보를 선택하면, 해당 교육생의 시험 점수를 입력할 수 있어야 한다. (이때, 출결, 필기, 실기 점수를 구분해서 입력할 수 있어야 한다.)

-- 강의를 마친 과정 -> 과목번호

— 해당 과목에 해당하는 교육생 번호

```
create or replace procedure Subject_score_update_p (p_student_seq in Subject_score.subject_seq%type, p_subject_seq in  
Subject_score.Subject_seq%type, p_attend_score in Subject_score.attend_score%type, p_test_writescore in Subject_score.test_writescore%type,  
p_test_realscore in Subject_score.test_realscore%type)
```

```
is
```

```
begin
```

```
    update Subject_score set attend_score = p_attend_score where student_seq = p_student_seq and subject_seq = p_subject_seq;
```

```
    update Subject_score set test_writescore = p_test_writescore where student_seq = p_student_seq and subject_seq = p_subject_seq;
```

```
    update Subject_score set test_realscore = p_test_realscore where student_seq = p_student_seq and subject_seq = p_subject_seq;
```

```
    commit;
```

```
    dbms_output.put_line('UPDATE SUCCESS');
```

```
end Subject_score_update_p;
```

--호출

begin

Subject_score_update_p(학생번호, 과목번호, 출결점수, 필기점수, 실기점수);

end;

-과목 목록 출력 시 과목번호, 과정명, 과정기간(시작 연월일, 끝 연월일), 강의실, 과목명, 과목기간(시작 연월일, 끝 연월일), 교재명, 출결, 필기, 실기 배점, 성적 등록 여부 등이 출력된다.

CREATE OR REPLACE PROCEDURE Subject_info_select_p

IS

BEGIN

FOR c IN (

select

o.subject_seq as "과목번호",

c.curs_name as "과정명",

c.begin_date as "과정기간(시작)",

c.end_date as "과정기간(종료)",

c.room_seq as "강의실",

s.subject_name as "과목이름",

o.begin_date as "과목시작일",

o.end_date as "과목종료일",

b.subject as "교재",

e.score_attend as "출석점수",

e.score_writeTest as "필기점수",

e.score_realTest as "실기점수"


```

from subject s
  inner join open_subject o on s.subject_seq = o.subject_seq
  inner join open_curs c on c.open_curs_seq = o.open_curs_seq
  inner join subject_book t on t.subject_seq = s.subject_seq
  inner join book b on t.book_seq = b.book_seq
  inner join score e on e.subject_seq = s.subject_seq
)
LOOP
  DBMS_OUTPUT.PUT_LINE('과목번호: ' || c."과목번호" || ', 과정명: ' || c."과정명" || ', 과정기간(시작): ' || c."과정기간(시작)" || ', 과정기간(종료): ' ||
c."과정기간(종료)"
  || ', 강의실: ' || c."강의실" || ', 과목이름: ' || c."과목이름" || ', 과목시작일: ' || c."과목시작일" || ', 과목종료일: ' || c."과목종료일" || ', 교재: ' || c."교재"
  || ', 출석점수: ' || c."출석점수" || ', 필기점수: ' || c."필기점수" || ', 실기점수: ' || c."실기점수");
END LOOP;
END Subject_info_select_p;

```

-호출

```

begin
Subject_info_select_p;
end;

```

-특정 과목을 과목번호로 선택 시 교육생 정보(이름, 전화번호, 수료 또는 중도탈락, 수료 또는 중도탈락 날짜) 및 성적이 출결, 필기, 실기 점수로 구분되어서 출력되어야 한다.

-과정을 중도 탈락해서 성적 처리가 제외된 교육생이더라도 교육생 명단에는 출력되어야 한다. 중도 탈락 여부를 확인할 수 있도록 해야 한다.

```

create or replace procedure score_student_info_select_p (p_subject_seq in Subject_score.subject_seq%type)

```

```

is

```

```

begin
  For c In (
    select
      s.subject_seq,
      m.name,
      m.tel,
      d.drop_out,
      d.drop_out_date,
      r.attend_score,
      r.test_writescore,
      r.test_realscore
    from subject s
      inner join teaching_subject e on s.subject_seq = e.subject_seq
      inner join teacher t on t.teacher_seq = e.teacher_seq
      inner join subject_score r on r.subject_seq = s.subject_seq
      inner join student d on d.student_seq = r.student_seq
      inner join member m on d.member_seq = m.member_seq
      where s.subject_seq = p_subject_seq
  )
  LOOP
    DBMS_OUTPUT.PUT_LINE('과목번호: ' || c.subject_seq || ', 교육생이름: ' || c.name || ', 전화번호: ' || c.tel || ', 교육생현재상태: ' || c.drop_out || ', 수료 및
중도탈락 날짜: ' || c.drop_out_date || ', 출결점수: ' || c.attend_score || ', 필기점수: ' || c.test_writescore || ', 실기점수: ' || c.test_realscore);
  
```

```
END LOOP;  
END score_student_info_select_p;
```

-실행

```
begin  
score_student_info_select_p(1);  
end;
```

요구사항번호	C-04	작성일	2023-04-05
요구사항명	출결 관리 및 출결 조회	작성자	이채린
DML			
<p>1. 교사가 강의한 과정에 한해 선택하는 경우 모든 교육생의 출결을 조회할 수 있어야 한다. (모든 출결 조회는 근태 상황을 구분할 수 있어야 한다.(정상, 지각, 조퇴, 외출, 병가, 기타)</p> <p>2.</p> <pre> create or replace procedure curriculum_attendance_select_p (p_teacher in Teacher.teacher_seq%type, p_open_curs_seq in Open_curs.open_curs_seq%type) is begin For c In (select oc.open_curs_seq as "개설과정번호", st.student_seq as "교육생번호", mb.name as "교육생이름", at.attendance_date as "출석날짜", CASE WHEN se.sick_or_etc IS NULL THEN '출석' ELSE se.sick_or_etc END AS "근태상황" from open_curs oc inner join teacher t on t.teacher_seq = oc.teacher_seq inner join curriculum_list cl on cl.curriculum_list_seq = oc.curriculum_list_seq inner join student st on cl.curriculum_list_seq = st.curriculum_list_seq inner join member mb on mb.member_seq = st.member_seq inner join attendance at on st.student_seq = at.student_seq </pre>			

```

left outer join sick_etc se on at.attendance_seq = se.attendance_seq
where t.teacher_seq = p_teacher and oc.open_curs_seq = p_open_curs_seq
)
LOOP
    DBMS_OUTPUT.PUT_LINE('개설과정번호: ' || c."개설과정번호" || ', 교육생번호: ' || c."교육생번호" || ', 교육생이름: ' || c."교육생이름" || ', 출석날짜: ' ||
c."출석날짜" || ', 근태상황: ' || c."근태상황");
END LOOP;
END curriculum_attendance_select_p;

```

-실행

```

begin
curriculum_attendance_select_p(1, 1);
end;

```

2. 출결 현황을 기간별(년, 월, 일) 조회할 수 있어야 한다.

```

create or replace procedure attendance_date_select_p (p_teacher in Teacher.teacher_seq%type, p_date1 in Attendance.attendance_date%type,
p_date2 in Attendance.attendance_date%type)
is
begin
    For c In (
        select
        oc.open_curs_seq as "개설과정번호",
        st.student_seq as "교육생번호",
        mb.name as "교육생이름",
        at.attendance_date as "출석날짜",

```

```

CASE WHEN se.sick_or_etc IS NULL THEN '출석'
ELSE se.sick_or_etc
END AS "근태상황"
from open_curs oc
  inner join teacher t on t.teacher_seq = oc.teacher_seq
  inner join curriculum_list cl on cl.curriculum_list_seq = oc.curriculum_list_seq
  inner join student st on cl.curriculum_list_seq = st.curriculum_list_seq
  inner join member mb on mb.member_seq = st.member_seq
  inner join attendance at on st.student_seq = at.student_seq
  left outer join sick_etc se on at.attendance_seq = se.attendance_seq
  where t.teacher_seq = p_teacher and at.attendance_date between p_date1 and p_date2
)
LOOP
  DBMS_OUTPUT.PUT_LINE('개설과정번호: ' || c."개설과정번호" || ', 교육생번호: ' || c."교육생번호" || ', 교육생이름: ' || c."교육생이름" || ', 출석날짜: ' ||
c."출석날짜" || ', 근태상황: ' || c."근태상황");
END LOOP;
END attendance_date_select_p;

begin
attendance_date_select_p(교사번호, 조회시작날짜, 조회끝날짜);
end;

3. 특정(특정 과정, 특정 인원) 출결 현황을 조회할 수 있어야 한다.

create or replace procedure student_attendance_select_p (p_curriculum_list_seq in Curriculum_list.curriculum_list_seq%type, p_student_seq in
Student.student_seq%type)

```

```

is
begin
  For c In (
    select
      c.curriculum_list_seq as "과정번호",
      d.student_seq as "교육생번호",
      mb.name as "교육생이름",
      at.attendance_date as "출석날짜",
      CASE WHEN se.sick_or_etc IS NULL THEN '출석'
      ELSE se.sick_or_etc
      END AS "근태상황"
    from teaching_subject s
      inner join teacher t on s.teacher_seq = t.teacher_seq
      inner join curriculum_subject c on c.subject_seq = s.subject_seq
      inner join subject_score e on s.subject_seq = e.subject_seq
      inner join student d on e.student_seq = d.student_seq
      inner join member mb on mb.member_seq = d.member_seq
      inner join attendance at on at.student_seq = d.student_seq
      left outer join sick_etc se on at.attendance_seq = se.attendance_seq
    where c.curriculum_list_seq = p_curriculum_list_seq
    and d.student_seq = p_student_seq
    GROUP BY c.curriculum_list_seq, d.student_seq, mb.name, at.attendance_date, se.sick_or_etc
    ORDER BY at.attendance_date
  )
  LOOP
    DBMS_OUTPUT.PUT_LINE('과정번호: ' || c."과정번호" || ', 교육생번호: ' || c."교육생번호" || ', 교육생이름: ' || c."교육생이름" || ', 출석날짜: ' || c."출석날짜" || ',

```

```
근태상황: ' || c."근태상황");
```

```
END LOOP;
```

```
END student_attendance_select_p;
```

```
-실행
```

```
begin
```

```
student_attendance_select_p(3, 1);
```

```
end;
```


요구사항번호	C-05	작성일	2023-04-05
요구사항명	상담일지 작성 및 조회	작성자	이동재
DML			
<p>1. 상담 일지 작성</p> <pre> create or replace procedure counsel_insert (pteacher_seq in number, pstudent_seq in number, pcounsel_topic_seq in number, pcounsel_date in date, pcounsel_text in varchar2) is begin insert into counsel values (cnsl_seq.nextVal, pteacher_seq, pstudent_seq, pcounsel_topic_seq, pcounsel_date, pcounsel_text); commit; dbms_output.put_line('상담일지가 등록 되었습니다.');</pre> <p>end counsel_insert;</p> <p>2. 상담 일지 조회</p> <pre> CREATE OR REPLACE PROCEDURE print_counsel IS BEGIN FOR c IN (SELECT counsel_seq as "상담번호", teacher_seq as "교사번호",</pre>			

```

        student_seq as "교육생번호",
        counsel_topic_seq as "상담주제번호",
        counsel_date as "상담일자",
        counsel_text as "상담내용"
    FROM counsel)
LOOP
    -- 결과를 출력합니다.
    DBMS_OUTPUT.PUT_LINE('상담번호: ' || c.상담번호);
    DBMS_OUTPUT.PUT_LINE('교사번호: ' || c.교사번호);
    DBMS_OUTPUT.PUT_LINE('교육생번호: ' || c.교육생번호);
    DBMS_OUTPUT.PUT_LINE('상담주제번호: ' || c.상담주제번호);
    DBMS_OUTPUT.PUT_LINE('상담일자: ' || c.상담일자);
    DBMS_OUTPUT.PUT_LINE('상담내용: ' || c.상담내용);
END LOOP;
END print_counsel;

```

요구사항번호	C-06	작성일	2023-04-05
요구사항명	교육생 사후처리 조회	작성자	이동재
DML			
<p>- 과정을 선택하면 해당 과정을 수강한 전체 교육생의 교육생 번호, 교육생 이름, 취업 분야, 취업 회사, 취업일을 출력한다.</p> <pre> CREATE OR REPLACE PROCEDURE job_current_info (pcurriculum_list_seq IN NUMBER) IS BEGIN FOR c IN (SELECT j.student_seq AS "학생 번호", m.name AS "이름", m.tel AS "전화번호", f.field_name AS "취업 분야", j.company AS "회사", j.hire_date AS "취업일" FROM job_current j LEFT OUTER JOIN student s ON j.student_seq = s.student_seq LEFT OUTER JOIN member m ON m.member_seq = s.member_seq LEFT OUTER JOIN curriculum_list c ON s.curriculum_list_seq = c.curriculum_list_seq LEFT OUTER JOIN job_field f ON j.job_field_seq = f.job_field_seq </pre>			

```

        WHERE c.curriculum_list_seq = pcurriculum_list_seq
    )
LOOP
    DBMS_OUTPUT.PUT_LINE('학생 번호: ' || c."학생 번호");
    DBMS_OUTPUT.PUT_LINE('이름: ' || c."이름");
    DBMS_OUTPUT.PUT_LINE('전화번호: ' || c."전화번호");
    DBMS_OUTPUT.PUT_LINE('취업 분야: ' || c."취업 분야");
    DBMS_OUTPUT.PUT_LINE('회사: ' || c."회사");
    DBMS_OUTPUT.PUT_LINE('취업일: ' || c."취업일");
END LOOP;
END job_current_info;

```

- 학생을 선택하면 해당 학생에 대한 사후 처리를 출력한다.

```

CREATE OR REPLACE PROCEDURE job_info_student(
    pname IN VARCHAR2
) IS
BEGIN
    FOR c IN (
        SELECT
            s.student_seq AS "학생번호",
            m.name AS "이름",
            m.tel AS "전화번호",
            j.job_field_seq AS "분야 번호",
            f.field_name AS "분야",
            j.company AS "회사",

```

```

        j.hire_date AS "고용날짜"
FROM job_current j
INNER JOIN student s
    ON j.student_seq = s.student_seq
INNER JOIN member m
    ON m.member_seq = s.member_seq
INNER JOIN job_field f
    ON f.job_field_seq = j.job_field_seq
WHERE m.name = pname
) LOOP
    DBMS_OUTPUT.PUT_LINE('학생번호: ' || c.학생번호);
    DBMS_OUTPUT.PUT_LINE('이름: ' || c.이름);
    DBMS_OUTPUT.PUT_LINE('전화번호: ' || c.전화번호);
    DBMS_OUTPUT.PUT_LINE('분야 번호: ' || c."분야 번호");
    DBMS_OUTPUT.PUT_LINE('분야: ' || c.분야);
    DBMS_OUTPUT.PUT_LINE('회사: ' || c.회사);
    DBMS_OUTPUT.PUT_LINE('고용날짜: ' || c.고용날짜);
END LOOP;
END job_info_student;

```

요구사항번호	C-07	작성일	2023-04-05
요구사항명	교사 추천 도서 등록 및 조회	작성자	조연우
DML			
<p>-1. 추천 도서 추가(추천도서번호(자동생성), 과목번호, 교사번호, 제목)</p> <p>— 선언문</p> <pre> create or replace procedure comment_book_insert_p (p_subject_seq in comment_book.subject_seq%type, p_teacher_seq in comment_book.teacher_seq%type, p_book_name in comment_book.book_name%type) is begin insert into comment_book(comment_book_seq, subject_seq, teacher_seq, book_name) values (cbook_seq.nextVal, p_subject_seq, p_teacher_seq, p_book_name); commit; dbms_output.put_line('insert success'); end comment_book_insert_p; / </pre> <p>— 호출문</p> <pre> DECLARE v_subject_seq comment_book.subject_seq%TYPE := 123; </pre>			

```

v_teacher_seq comment_book.teacher_seq%TYPE := 456;
v_book_name comment_book.book_name%TYPE := 'My Book';
BEGIN
    comment_book_insert_p(
        p_subject_seq => v_subject_seq,
        p_teacher_seq => v_teacher_seq,
        p_book_name => v_book_name
    );
END;
/

--2. 추천 도서 수정하기
-- 선언문
CREATE OR REPLACE PROCEDURE comment_book_update_p(
    p_comment_book_seq comment_book.comment_book_seq%TYPE,
    p_book_name comment_book.book_name%TYPE
) IS
BEGIN
    UPDATE comment_book
    SET book_name = p_book_name
    WHERE comment_book_seq = p_comment_book_seq;
    COMMIT;
    DBMS_OUTPUT.PUT_LINE('Update success');
END comment_book_update_p;
/

```

— 호출문

BEGIN

comment_book_update_p(p_comment_book_seq => 121, p_book_name => '자바 개론');

END;

/

--3. 추천 도서 조회하기

— 선언하기

CREATE OR REPLACE PROCEDURE comment_book_select_p

IS

BEGIN

FOR c IN (

SELECT subject_seq, teacher_seq, book_name

FROM comment_book

)

LOOP

DBMS_OUTPUT.PUT_LINE('과목번호: ' || c.subject_seq || ', 교사번호: ' || c.teacher_seq || ', 책제목: ' || c.book_name);

END LOOP;

END comment_book_select_p;

/

— 호출하기

BEGIN

comment_book_select_p;

END;

요구사항번호	C-08	작성일	2023-04-05
요구사항명	만남의 광장	작성자	조연우
DML			
<pre> - 선언 create or replace PROCEDURE board_select_p IS BEGIN FOR c IN (SELECT b.board_seq, b.student_seq, b.board_title, b.board_content, br.reply_seq, br.student_seq AS reply_student_seq, br.reply_content from board b inner join board_reply br on b.board_seq = br.board_seq) LOOP DBMS_OUTPUT.PUT_LINE('게시물번호: ' c.board_seq ', 교육생번호: ' c.student_seq ', 게시글 제목: ' c.board_title ', 게시글 본문: ' c.board_content ', 댓글번호: ' c.reply_seq ', 교육생번호(댓글): ' c.reply_student_seq ', 댓글내용: ' c.reply_content); END LOOP; END board_select_p; - 호출 begin board_select_p; end; </pre>			

요구사항번호	C-09	작성일	2023-04-05
요구사항명	퀴즈 관리	작성자	유동현
DML			
<pre> --1. 퀴즈 관리 --입력 insert into QUIZ values(42, '자바의 이해', '다음중 정답은?' CHR(13) CHR(10) '1. 1번 2. 2번 3. 3번 4. 4번 5. 5번', 1, qanswer_seq.nextVal); --수정 create or replace procedure update_quiz_title(quiz_id number, new_title quiz.quiz_title%type, new_content quiz.quiz_content%type, new_subseq quiz.subject_seq%type, new_answer quiz.quiz_answer%type) is begin if new_title is null or new_content is null or new_subseq is null or new_answer is null then dbms_output.put_line('수정할 항목이 전달되지 않았습니다.');</pre>			
<pre> return; end if; update quiz set quiz_title = new_title,</pre>			

```

    quiz_content = new_content,
    subject_seq = new_subseq,
    quiz_answer = new_answer
where quiz_seq = quiz_id;

dbms_output.put_line('퀴즈가 수정되었습니다. ');
exception
when others then
    dbms_output.put_line('수정에 실패했습니다. ');
    rollback;
end;

--실행
declare
begin
    update_quiz_title(<변경할퀴즈번호>,<새로운퀴즈제목>,<새로운퀴즈내용>,<새로운과목번호>,<새로운퀴즈정답>)
end;

--삭제
create or replace procedure pdelete_quiz(
    quiz_seq number
)
is
begin
    delete from quiz where quiz_seq = quiz_seq;
end;

```

—실행

declare

begin

pdelete_quiz(삭제할 퀴즈번호);

end;

--조회

create or replace procedure print_quiz_columns

is

begin

for c in (select quiz_seq, quiz_title, quiz_content, subject_seq, quiz_answer from quiz) loop

dbms_output.put_line('퀴즈번호: ' || c.quiz_seq || ', 퀴즈제목: ' || c.quiz_title || ', 퀴즈내용: ' || c.quiz_content || ', 과목번호: ' || c.subject_seq || ', 정답번호: ' || c.quiz_answer);

end loop;

end;

—실행

begin

print_quiz_columns;

end;

교육생 계정

요구사항번호	D-00	작성일	2023-04-05
요구사항명	교육생 로그인	작성자	이동재
DML			
<pre> CREATE OR REPLACE PROCEDURE member_P (p_ssn IN member.ssn%TYPE) IS v_member_seq member.member_seq%TYPE; v_member_name member.name%TYPE; v_member_tel member.tel%TYPE; BEGIN SELECT member_seq, name, tel INTO v_member_seq, v_member_name, v_member_tel FROM member WHERE ssn = p_ssn; DBMS_OUTPUT.PUT_LINE('학생 번호: ' v_member_seq ', 학생 이름: ' v_member_name ', 학생 전화번호: ' v_member_tel); END member_P; </pre>			

요구사항번호	D-01	작성일	2023-04-05
요구사항명	성적 조회	작성자	이동재
DML			
<p>1. 교육생이 로그인에 성공하면 교육생 개인의 정보와 교육생이 수강한 과정명, 과정 기간(시작 연월일, 끝 연월일), 강의실이 타이틀로 출력된다.</p> <pre> CREATE OR REPLACE PROCEDURE student_private_info(pname IN VARCHAR2) AS BEGIN FOR c IN (SELECT mb.name AS "이름", mb.ssn AS "주민번호", mb.tel AS "전화번호", cl.curs_name AS "과정명", st.enroll_date AS "과정시작날짜", st.drop_out_date AS "과정종료날짜", oc.room_seq AS "강의실" FROM student st INNER JOIN member mb ON st.member_seq = mb.member_seq INNER JOIN curriculum_list cl ON st.curriculum_list_seq = cl.curriculum_list_seq LEFT OUTER JOIN open_curs oc ON st.curriculum_list_seq = oc.curriculum_list_seq WHERE mb.name = pname </pre>			

```

) LOOP
  DBMS_OUTPUT.PUT_LINE('이름: ' || c.이름);
  DBMS_OUTPUT.PUT_LINE('주민번호: ' || c.주민번호);
  DBMS_OUTPUT.PUT_LINE('전화번호: ' || c.전화번호);
  DBMS_OUTPUT.PUT_LINE('과정 명: ' || c.과정명);
  DBMS_OUTPUT.PUT_LINE('과정 시작날짜: ' || c.과정시작날짜);
  DBMS_OUTPUT.PUT_LINE('과정 종료날짜: ' || c.과정종료날짜);
  DBMS_OUTPUT.PUT_LINE('강의실: ' || c.강의실);
END LOOP;
END student_private_info;

```

2. 성적 정보는 과목별로 목록 형태로 출력된다.

```

CREATE OR REPLACE PROCEDURE student_private_score(pssn IN number)
AS
BEGIN
  FOR c IN (
    SELECT
      mb.name as "이름",
      sj.subject_name as "과목명",
      sc.attend_score as "출결점수",
      sc.test_writescore as "필기시험점수",
      sc.test_realscore as "실기시험점수"
    from student st
      inner join subject_score sc
        on st.student_seq = sc.student_seq

```



```

        inner join member mb
        on st.member_seq = mb.member_seq
        inner join subject sj
        on sc.subject_seq = sj.subject_seq
        where mb.ssn = pssn
    ) LOOP
        DBMS_OUTPUT.PUT_LINE('이름: ' || c.이름);
        DBMS_OUTPUT.PUT_LINE('과목명: ' || c.과목명);
        DBMS_OUTPUT.PUT_LINE('출결점수: ' || c.출결점수);
        DBMS_OUTPUT.PUT_LINE('필기시험점수: ' || c.필기시험점수);
        DBMS_OUTPUT.PUT_LINE('실기시험점수: ' || c.실기시험점수);
    END LOOP;
END student_private_score;

```

3. 출력될 정보는 과목번호, 과목명, 과목 기간(시작 연월일, 끝 연월일), 교재명, 교사명, 과목별 배점 정보(출결, 필기, 실기 배점), 과목별 성적 정보(출결, 필기, 실기 점수), 과목별 시험날짜, 시험문제가 출력되어야 한다.

```

CREATE OR REPLACE PROCEDURE student_private_score_info(pstudent_seq IN number)
AS
BEGIN
    FOR c IN (
        select
        distinct
        s.subject_seq as "과목번호",
        s.subject_name as "과목명",
        os.begin_date as "과목시작일자",

```

```

os.end_date as "과목종료일자",
b.subject as "교재명",
t.name as "교사명",
sc.score_attend as "출결배점",
sc.score_writetest as "필기시험배점",
sc.score_realtest as "실기시험배점",
ss.attend_score as "출결점수",
ss.test_writescore as "필기점수",
ss.test_realscore as "실기점수",
td.test_date as "과목별시험날짜",
q.quiz_content as "시험문제"
from Subject s
  inner join open_subject os
    on s.subject_seq = os.subject_seq
    inner join subject_book sb
      on sb.subject_seq = s.subject_seq
      inner join book b
        on b.book_seq = sb.book_seq
        inner join teacher t
          on os.teacher_seq = t.teacher_seq
          inner join score sc
            on s.subject_seq = sc.subject_seq
            inner join subject_score ss
              on s.subject_seq = ss.subject_seq
              inner join student std
                on std.student_seq = ss.student_seq

```

```

        inner join test_date td
        on td.subject_seq = s.subject_seq
        inner join quiz q
        on q.quiz_seq = td.quiz_seq
        where std.student_seq = pstudent_seq
) LOOP
    DBMS_OUTPUT.PUT_LINE('과목번호: ' || c.과목번호);
    DBMS_OUTPUT.PUT_LINE('과목명: ' || c.과목명);
    DBMS_OUTPUT.PUT_LINE('과목시작일자: ' || c.과목시작일자);
    DBMS_OUTPUT.PUT_LINE('과목종료일자: ' || c.과목종료일자);
    DBMS_OUTPUT.PUT_LINE('교재명: ' || c.교재명);
    DBMS_OUTPUT.PUT_LINE('교사명: ' || c.교사명);
    DBMS_OUTPUT.PUT_LINE('출결배점: ' || c.출결배점);
    DBMS_OUTPUT.PUT_LINE('필기시험배점: ' || c.필기시험배점);
    DBMS_OUTPUT.PUT_LINE('실기시험배점: ' || c.실기시험배점);
    DBMS_OUTPUT.PUT_LINE('출결점수: ' || c.출결점수);
    DBMS_OUTPUT.PUT_LINE('필기점수: ' || c.필기점수);
    DBMS_OUTPUT.PUT_LINE('실기점수: ' || c.실기점수);
    DBMS_OUTPUT.PUT_LINE('과목별시험날짜: ' || c.과목별시험날짜);
    DBMS_OUTPUT.PUT_LINE('시험문제: ' || c.시험문제);
END LOOP;
END student_private_score_info;

```

요구사항번호	D-02	작성일	2023-04-05
요구사항명	출결 관리 및 출결 조회	작성자	김대환
DML			
<p>1. 매일 출석을 기록할 수 있어야 한다.(출근 1회, 퇴근 1회)</p> <p>-- 3번교육생</p> <p>--출결</p> <p>— 선언문</p> <pre> CREATE OR REPLACE PROCEDURE attendance_insert_p (p_student_seq attendance.student_seq%type) IS BEGIN insert into attendance(attendance_seq, student_seq, attendance_date) values (attendance_seq.nextVal, p_student_seq, sysdate); commit; dbms_output.put_line('insert success'); END attendance_insert_p; — 호출문 DECLARE v_student_seq attendance.student_seq%type := 23; -- 학생번호 23번 출석 </pre>			

```

BEGIN
    attendance_insert_p(p_student_seq => v_student_seq);
END;

--출근
-- 선언문
CREATE OR REPLACE PROCEDURE attendance_detail_insert_p (
    p_attendance_seq attendance_detail.attendance_seq%type,
    p_check_in attendance_detail.check_in%type,
    p_check_out attendance_detail.check_out%type
)
IS
BEGIN
    insert into attendance_detail(attendance_detail_seq, attendance_seq, check_in, check_out)
        values (attendance_detail_seq.nextVal, p_attendance_seq, p_check_in, p_check_out);

    commit;

    dbms_output.put_line('insert success');

END attendance_detail_insert_p;

-- 호출문
declare
    l_attendance_seq attendance_detail.attendance_seq%type := 1001;
    l_check_in attendance_detail.check_in%type := to_date('09:00:52');

```

```

    l_check_out attendance_detail.check_out%type := to_date(null);
begin
    attendance_detail_insert_p(l_attendance_seq, l_check_in, l_check_out);
end;

--퇴근
-- 선언문
CREATE OR REPLACE PROCEDURE attendance_detail_update_p (
    p_attendance_seq attendance_detail.attendance_seq%type,
    p_check_out attendance_detail.check_out%type
)
IS
BEGIN
    update attendance_detail
    set check_out = p_check_out
    where attendance_seq = p_attendance_seq;

    commit;

    dbms_output.put_line('update success');

END attendance_detail_update_p;

-- 호출문
declare
    v_attendance_seq attendance_detail.attendance_seq%type := 52023;

```

```

v_check_out attendance_detail.check_out%type := to_date('18:00:00');
begin
    attendance_detail_update_p(v_attendance_seq, v_check_out);
end;

```

2. 본인의 출결 현황을 기간별(전체, 월, 일) 조회할 수 있어야 한다.

-- 3번교육생 전체

— 선언문

```

CREATE OR REPLACE PROCEDURE get_attendance_date_p (
    p_student_seq IN attendance.student_seq%TYPE
)

```

IS

CURSOR c_attendance_dates IS

SELECT attendance_date

FROM attendance

WHERE student_seq = p_student_seq;

l_attendance_date attendance.attendance_date%TYPE;

BEGIN

FOR r_attendance_date IN c_attendance_dates LOOP

l_attendance_date := r_attendance_date.attendance_date;

DBMS_OUTPUT.PUT_LINE('Attendance Date: ' || TO_CHAR(l_attendance_date, 'YYYY-MM-DD'));

END LOOP;

END;

— 호출문

```

DECLARE
    v_student_seq attendance.student_seq%TYPE := 123; -- 조회하고자 하는 학생 번호를 지정
BEGIN
    get_attendance_date_p(p_student_seq => v_student_seq);
END;

-- 3번교육생 22년 10월
-- 선언문
CREATE OR REPLACE PROCEDURE get_attendance_date2_p (
    p_student_seq IN attendance.student_seq%TYPE,
    p_year IN NUMBER,
    p_month IN NUMBER
)
IS
BEGIN
    FOR c IN (
        SELECT attendance_date
        FROM attendance
        WHERE student_seq = p_student_seq
        AND attendance_date LIKE ('%' || p_year || '/' || LPAD(p_month, 2, '0') || '%')
    )
    LOOP
        DBMS_OUTPUT.PUT_LINE('Attendance Date: ' || TO_CHAR(c.attendance_date, 'YY-MM-DD'));
    END LOOP;
END get_attendance_date2_p;

```


— 호출문

DECLARE

 v_student_seq attendance.student_seq%TYPE := 3;

 v_year NUMBER := 23;

 v_month NUMBER := 04;

BEGIN

 get_attendance_date2_p(p_student_seq => v_student_seq, p_year => v_year, p_month => v_month);

END;

-- 3번교육생 22년 10월 31일

— 선언문

CREATE OR REPLACE PROCEDURE get_attendance_date2_p (

 p_student_seq IN attendance.student_seq%TYPE,

 p_year IN NUMBER,

 p_month IN NUMBER,

 p_day IN NUMBER,

 p_result OUT VARCHAR2

)

IS

 v_attendance_date DATE;

BEGIN

 SELECT attendance_date INTO v_attendance_date

 FROM attendance

 WHERE student_seq = p_student_seq

 AND TO_CHAR(attendance_date, 'YYYY/MM/DD') = TO_CHAR(TO_DATE(p_year || '/' || p_month || '/' || p_day, 'YYYY/MM/DD'), 'YYYY/MM/DD');

```

    p_result := '출석' ;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        p_result := '결석';
END;

-- 호출문
DECLARE
    v_result VARCHAR2(10);
BEGIN
    get_attendance_date2_p(3, 2022, 10, 31, v_result);
    DBMS_OUTPUT.PUT_LINE(v_result);
END;

```

요구사항번호	D-03	작성일	2023-04-05
요구사항명	상담일지 조회	작성자	이동재
DML			
<p>1. 교육생은 자신의 상담일지를 조회할 수 있다.</p> <pre> CREATE OR REPLACE PROCEDURE counsel_info(pstudent_seq IN number) AS BEGIN FOR c IN (select teacher_seq as "상담선생님번호", counsel_date as "상담일자", counsel_text as "상담내용" from Counsel where student_seq = pstudent_seq) LOOP DBMS_OUTPUT.PUT_LINE('상담선생님번호: ' c.상담선생님번호); DBMS_OUTPUT.PUT_LINE('상담일자: ' c.상담일자); DBMS_OUTPUT.PUT_LINE('상담내용: ' c.상담내용); END LOOP; END counsel_info; </pre> <p>2. 상담 날짜, 상담 교사, 상담 학생, 상담 내용이 출력된다.</p> <pre> CREATE OR REPLACE PROCEDURE counsel_detail_info(pstudent_seq IN number) AS BEGIN </pre>			

```

FOR c IN (
  select
    c.counsel_date as "상담일자",
    t.name as "선생님이름",
    m.name as "교육생이름",
    c.counsel_text as "상담내용"
  from counsel c
    inner join teacher t
      on c.teacher_seq = t.teacher_seq
    inner join student s
      on s.student_seq = c.student_seq
    inner join Counsel_topic n
      on n.counsel_topic_seq = c.counsel_topic_seq
    right outer join member m
      on s.member_seq = m.member_seq
    where s.student_seq = pstudent_seq

) LOOP
  DBMS_OUTPUT.PUT_LINE('상담일자: ' || c.상담일자);
  DBMS_OUTPUT.PUT_LINE('선생님이름: ' || c.선생님이름);
  DBMS_OUTPUT.PUT_LINE('교육생이름: ' || c.교육생이름);
  DBMS_OUTPUT.PUT_LINE('상담내용: ' || c.상담내용);
END LOOP;
END counsel_detail_info;

```

요구사항번호	D-04	작성일	2023-04-05
요구사항명	사후처리 입력	작성자	1조
DML			
<p>1.취업 활동 등록</p> <pre> create or replace procedure job_current_update (pjob_field_seq in number, pstudent_seq in number, pcompany in varchar2, phire_date in date) is begin update job_current set JOB_FIELD_SEQ = job_field_seq, company= pcompany, hire_date=phire_date where student_seq = pstudent_seq; commit; dbms_output.put_line('취업활동 내역이 등록 되었습니다.');</pre> <p>end job_current_update;</p>			

2. 취업 활동 조회

- 선언

```
CREATE OR REPLACE PROCEDURE job_current_select_p
```

```
IS
```

```
BEGIN
```

```
  FOR c IN (
```

```
    SELECT
```

```
      s.student_seq AS "학생번호",
```

```
      m.name AS "이름",
```

```
      c.curs_name AS "과정명",
```

```
      f.field_name AS "분야",
```

```
      jc.company AS "회사명",
```

```
      jc.hire_date AS "입사일"
```

```
FROM job_current jc
```

```
  INNER JOIN student s
```

```
    ON jc.student_seq = s.student_seq
```

```
    INNER JOIN Member m
```

```
      ON m.member_seq = s.member_seq
```

```
      INNER JOIN curriculum_list c
```

```
        ON s.curriculum_list_seq = c.curriculum_list_seq
```

```
        INNER JOIN job_field f
```

```
          ON jc.job_field_seq = f.job_field_seq
```

```
WHERE jc.company IS NOT NULL
```

```
)
```

```
LOOP
```

```

DBMS_OUTPUT.PUT_LINE('학생번호: ' || c.학생번호);
DBMS_OUTPUT.PUT_LINE('이름: ' || c.이름);
DBMS_OUTPUT.PUT_LINE('과정명: ' || c.과정명);
DBMS_OUTPUT.PUT_LINE('분야: ' || c.분야);
DBMS_OUTPUT.PUT_LINE('회사명: ' || c.회사명);
DBMS_OUTPUT.PUT_LINE('입사일: ' || c.입사일);
DBMS_OUTPUT.PUT_LINE('');
END LOOP;
END job_current_select_p;
/

```

3. 취업 활동 수정

- 직업 분야 수정

```

CREATE OR REPLACE PROCEDURE job_current_field_update_p (
    pfield IN job_current.job_field_seq%type,
    pstudent IN job_current.student_seq%type
)
IS
BEGIN
    UPDATE job_current SET job_field_seq = pfield WHERE student_seq = pstudent;
    DBMS_OUTPUT.PUT_LINE('수정됐습니다.');
```

COMMIT;

EXCEPTION WHEN OTHERS THEN

```

    DBMS_OUTPUT.PUT_LINE('수정에 실패했습니다.');
```

rollback;

```

END job_current_field_update_p;

```

- 회사명 수정

```
CREATE OR REPLACE PROCEDURE job_current_company_update_p (  
    pcompany IN job_current.company%type,  
    pstudent IN job_current.student_seq%type  
)  
IS  
BEGIN  
    UPDATE job_current SET company = pcompany WHERE student_seq = pstudent;  
    DBMS_OUTPUT.PUT_LINE('수정됐습니다.');
```

COMMIT;

EXCEPTION WHEN OTHERS THEN

DBMS_OUTPUT.PUT_LINE('수정에 실패했습니다.');

rollback;

END job_current_company_update_p;

/

- 입사일 수정

```
CREATE OR REPLACE PROCEDURE job_current_hiredate_update_p (  
    phire IN job_current.hire_date%type,  
    pstudent IN job_current.student_seq%type  
)  
IS  
BEGIN  
    UPDATE job_current SET hire_date = phire WHERE student_seq = pstudent;
```



```
DBMS_OUTPUT.PUT_LINE('수정됐습니다.');
```

```
COMMIT;
```

```
EXCEPTION WHEN OTHERS THEN
```

```
DBMS_OUTPUT.PUT_LINE('수정에 실패했습니다.');
```

```
rollback;
```

```
END job_current_hiredat_update_p;
```

요구사항번호	D-05	작성일	2023-04-05
요구사항명	교사 추천 도서	작성자	이동재
DML			
<p>1.교사가 등록한 추천 도서 정보를 조회할 수 있다.</p> <pre> CREATE OR REPLACE PROCEDURE recommend_book(psubject_seq IN number) AS BEGIN FOR c IN (select sj.subject_name as "과목이름", cb.teacher_seq as "교사번호", cb.book_name as "책제목" from comment_book cb inner join subject sj on cb.subject_seq = sj.subject_seq where cb.subject_seq = psubject_seq) LOOP DBMS_OUTPUT.PUT_LINE('과목이름: ' c.과목이름); DBMS_OUTPUT.PUT_LINE('교사번호: ' c.교사번호); DBMS_OUTPUT.PUT_LINE('책제목: ' c.책제목); END LOOP; END recommend_book; </pre>			

요구사항번호	D-06	작성일	2023-04-06
요구사항명	만남의 광장	작성자	김대환
DML			
<p>1. 게시물 등록</p> <p>– 선언문</p> <pre> CREATE OR REPLACE PROCEDURE Board_insert_p(p_student_seq board.student_seq%type, p_board_title board.board_title%type, p_board_content board.board_content%type) IS BEGIN insert into Board(Board_seq, student_seq, Board_title, Board_content) values (board_seq.nextVal, p_student_seq, p_board_title, p_board_content); commit; dbms_output.put_line('insert success'); END Board_insert_p; -- 호출문 DECLARE v_student_seq board.student_seq%type := 1234; -- 적절한 값을 넣어줍니다. </pre>			

```
v_board_title board.board_title%type := '제목입니다'; -- 적절한 값을 넣어줍니다.  
v_board_content board.board_content%type := '내용입니다'; -- 적절한 값을 넣어줍니다.
```

```
BEGIN
```

```
Board_insert_p(v_student_seq, v_board_title, v_board_content);
```

```
END;
```

2. 게시물 조회

— 선언문

```
CREATE OR REPLACE PROCEDURE Board_select_all_p
```

```
IS
```

```
BEGIN
```

```
FOR rec IN (SELECT * FROM Board)
```

```
LOOP
```

```
dbms_output.put_line('Board_seq: ' || rec.Board_seq || ', Student_seq: ' || rec.Student_seq || ', Board_title: ' || rec.Board_title || ', Board_content: ' ||  
rec.Board_content);
```

```
END LOOP;
```

```
END;
```

— 호출문

```
BEGIN
```

```
Board_select_all_p;
```

```
END;
```

3. 게시물 수정

— 제목 수정

— 선언문

```

CREATE OR REPLACE PROCEDURE board_update_p (
    p_board_seq board.board_seq%type,
    p_board_title board.board_title%type
)
IS
BEGIN
    update board
    set board_title = p_board_title
    where board_seq = p_board_seq;

    COMMIT;

    DBMS_OUTPUT.PUT_LINE('Update success');
END board_update_p;

-- 호출문
BEGIN
    board_update_p(p_board_seq => 13, p_board_title => '수정할 제목');
END;

-- 내용수정
-- 선언문
CREATE OR REPLACE PROCEDURE board_update2_p (
    p_board_seq board.board_seq%type,
    p_board_content board.board_content%type
)

```

```

IS
BEGIN
    update board
    set board_content = p_board_content
    where board_seq = p_board_seq;

    COMMIT;

    DBMS_OUTPUT.PUT_LINE('Update success');
END board_update2_p;

```

— 호출문

```

BEGIN
    board_update2_p(p_board_seq => 13, p_board_content => '수정할 내용');
END;

```

4. 게시물 삭제

— 선언문

```

CREATE OR REPLACE PROCEDURE board_delete_p (
    p_board_seq board.board_seq%type
)
IS
BEGIN
    delete from board where board_seq = p_board_seq;
    commit;
END board_delete_p;

```

— 호출문

DECLARE

v_board_seq board.board_seq%TYPE := 13;

BEGIN

board_delete_p(

p_board_seq => v_board_seq

);

END;

5. 답글 작성

create or replace procedure Board_Reply_insert (

pBoard_seq in number,

pstudent_seq in number,

preply_content in varchar2

)

is

begin

insert into Board_Reply (reply_seq, Board_seq, student_seq, reply_content) values (boardreplyseq.nextVal, pBoard_seq, pstudent_seq, preply_content);

commit;

dbms_output.put_line('게시판 답글 등록이 완료 되었습니다.');

end Board_Reply_insert;

요구사항번호	D-07	작성일	2023-04-05
요구사항명	달란트 시장	작성자	유동현
DML			
<pre> 1. 퀴즈 조회 — 선언문 CREATE OR REPLACE PROCEDURE quiz_print_p IS BEGIN FOR quiz_record IN (SELECT quiz_seq as "퀴즈번호", quiz_title as "퀴즈제목", quiz_content as "퀴즈내용", subject_seq as "과목번호", quiz_answer as "정답번호" FROM quiz) LOOP DBMS_OUTPUT.PUT_LINE('퀴즈번호: ' quiz_record."퀴즈번호"); DBMS_OUTPUT.PUT_LINE('퀴즈제목: ' quiz_record."퀴즈제목"); DBMS_OUTPUT.PUT_LINE('퀴즈내용: ' quiz_record."퀴즈내용"); DBMS_OUTPUT.PUT_LINE('과목번호: ' quiz_record."과목번호"); DBMS_OUTPUT.PUT_LINE('정답번호: ' quiz_record."정답번호"); DBMS_OUTPUT.PUT_LINE('-----'); END LOOP; END quiz_print_p; — 호출문 DECLARE BEGIN quiz_print_p; END;</pre>			

2. 퀴즈 응시

```
CREATE OR REPLACE PROCEDURE INSERT_QUIZ_CORRECT(  
  p_quiz_num IN NUMBER,  
  p_answer IN VARCHAR2,  
  p_is_correct IN NUMBER,  
  p_student_num IN NUMBER  
) AS  
BEGIN  
  INSERT INTO QUIZ_Correct (correct_Seq, quiz_seq, quiz_input, quiz_check, student_seq)  
  VALUES (quizcorrectSeq.nextVal, p_quiz_num, p_answer, p_is_correct, p_student_num);  
  COMMIT;  
END;  
/
```

3. 포인트 조회

— 선언문

```
CREATE OR REPLACE PROCEDURE point_print_p(  
  student_seq_in IN point.student_seq%TYPE  
) AS  
  point_record point%ROWTYPE;  
BEGIN  
  SELECT *  
  INTO point_record  
  FROM point  
  WHERE student_seq = student_seq_in;
```

```

    DBMS_OUTPUT.PUT_LINE('POINT_SEQ: ' || point_record.POINT_SEQ);
    DBMS_OUTPUT.PUT_LINE('STUDENT_SEQ: ' || point_record.STUDENT_SEQ);
    DBMS_OUTPUT.PUT_LINE('PPOINT: ' || point_record.PPOINT);
END point_print_p;

```

— 호출문

```

DECLARE
v_student_seq point.student_seq%TYPE := 10;
BEGIN
point_print_p(v_student_seq);
END;

```

4. 상품 조회

- 선언

```

CREATE OR REPLACE PROCEDURE prize_select_p
IS
    vrow prize%rowtype;
BEGIN
FOR c IN(
    SELECT
        prize_seq AS "상품 번호",
        prize_name AS "상품 이름",
        prize_point AS "가격"
    FROM prize
)

```

```
LOOP
vrow := c;
DBMS_OUTPUT.PUT_LINE('상품 번호: ' || c."상품 번호");
DBMS_OUTPUT.PUT_LINE('상품 이름: ' || c."상품 이름");
DBMS_OUTPUT.PUT_LINE('가격: ' || c.가격);
DBMS_OUTPUT.PUT_LINE('');
END LOOP;
END prize_select_p;
/

- 호출
BEGIN
    prize_select_p;
END;
/
```

그 외 오브젝트(트리거, 뷰, 인덱스)

분류	트리거-1	작성일	2023-04-05
목적	학생이 포인트로 상품을 구매하면 포인트를 차감	작성자	이동재
PL/SQL			
<pre> CREATE OR REPLACE TRIGGER buy_prize BEFORE INSERT ON student_prize_list FOR EACH ROW DECLARE v_prize_point NUMBER; v_ppoint NUMBER; BEGIN -- 상품의 포인트를 조회합니다. SELECT prize_point INTO v_prize_point FROM prize WHERE prize_seq = :NEW.prize_seq; -- 학생의 현재 포인트를 조회합니다. SELECT ppoint INTO v_ppoint FROM point WHERE student_seq = :NEW.student_seq; </pre>			

```
-- 구매에 필요한 포인트가 학생의 현재 포인트보다 크다면 INSERT 작업을 취소합니다.  
IF v_prize_point > v_ppoint THEN  
    RAISE_APPLICATION_ERROR(-20001, '포인트가 부족합니다.');
```



```
-- 학생의 포인트에서 상품의 포인트를 차감합니다.  
UPDATE point  
SET ppoint = ppoint - v_prize_point  
WHERE student_seq = :NEW.student_seq;  
END buy_prize;
```

분류	트리거-2	작성일	2023-04-05
목적	출결은 최소 20점 이상이어야 하고, 출결, 필기, 실기의 합은 100점이 되도록 한다.	작성자	이동재
PL/SQL			
<pre> CREATE OR REPLACE TRIGGER trg_score_check BEFORE INSERT OR UPDATE ON score FOR EACH ROW DECLARE v_total_score NUMBER; BEGIN -- score_attend가 20점 미만인 경우 INSERT/UPDATE를 막습니다. IF :NEW.score_attend < 20 THEN RAISE_APPLICATION_ERROR(-20001, 'score_attend는 최소 20점 이상이어야 합니다.');</pre>			
<pre> END IF; -- score_attend, score_writetest, score_realtest 의 합이 100점을 초과하는 경우 INSERT/UPDATE를 막습니다. v_total_score := :NEW.score_attend + :NEW.score_writetest + :NEW.score_realtest; IF v_total_score > 100 THEN RAISE_APPLICATION_ERROR(-20002, 'score_attend, score_writetest, score_realtest 의 합은 100점 이하여야 합니다.');</pre>			
<pre> END IF; END;</pre>			

분류	트리거-3	작성일	2023-04-05
목적	성적 입력시 출결 30점, 필기 30점, 실기 40점 이하여야 한다.	작성자	이채린
PL/SQL			
<pre> create or replace trigger trg_subject_score before insert or update on Subject_score for each row begin dbms_output.put_line('트리거 실행'); if :new.attend_score > 30 then raise_application_error (-21001, '출결점수는 최대 30점입니다.');</pre>			
<pre> elsif :new.test_writescore > 30 then raise_application_error (-21002, '필기점수는 최대 30점입니다.');</pre>			
<pre> elsif :new.test_realscore > 40 then raise_application_error (-21003, '실기점수는 최대 40점입니다.');</pre>			
<pre> end if; end trg_subject_score;</pre>			

분류	트리거-4	작성일	2023-04-05
목적	open_curs 테이블에 teacher_seq가 존재한 teacher만 counsel테이블에 insert문을 작성할 수 있도록 하고, 또한 student테이블에 curriculum_list_seq가 open_curs테이블에 open_cusr_seq에 존재할 경우에만 insert문을 작성할 수 있도록 하고, student 테이블에 drop_out = '교육이수중'인 사람만 insert문을 작성할 수 있도록 권한부여	작성자	김대환
PL/SQL			
<pre> CREATE OR REPLACE TRIGGER trg_counsel_insert BEFORE INSERT ON counsel FOR EACH ROW DECLARE v_open_curs_seq open_curs.open_curs_seq%type; v_teacher_seq NUMBER; v_curriculum_list_seq open_curs.curriculum_list_seq%type; BEGIN SELECT COUNT(*) INTO v_teacher_seq FROM open_curs oc WHERE oc.open_curs_seq = v_open_curs_seq AND oc.teacher_seq = :NEW.teacher_seq; IF v_teacher_seq = 0 THEN RAISE_APPLICATION_ERROR(-20005, '해당 강사는 개설 과목의 담당 강사가 아닙니다.');</pre>			
END IF;			

```

SELECT oc.curriculum_list_seq INTO v_curriculum_list_seq
FROM open_curs oc
WHERE oc.open_curs_seq = v_open_curs_seq;

IF v_curriculum_list_seq NOT BETWEEN 1 AND 6 THEN
    RAISE_APPLICATION_ERROR(-20002, '수강하는 과정과 개설한 강좌의 과정이 일치하지 않습니다.');
```

END IF;

```

SELECT COUNT(*) INTO v_teacher_seq
FROM teacher t
WHERE t.teacher_seq = :NEW.teacher_seq;

IF v_teacher_seq = 0 THEN
    RAISE_APPLICATION_ERROR(-20003, '등록되지 않은 강사입니다.');
```

END IF;

```

SELECT COUNT(*) INTO v_curriculum_list_seq
FROM student s
WHERE s.student_seq = :NEW.student_seq
AND s.drop_out = '교육이수중';

IF v_curriculum_list_seq = 0 THEN
    RAISE_APPLICATION_ERROR(-20004, '접근 권한이 없는 학생입니다.');
```

END IF;

END;

분류	트리거-5	작성일	2023-04-05
목적	과목 종강일 전 성적 입력 시도 시 입력 무효화	작성자	조연우
PL/SQL			
<pre> CREATE OR REPLACE TRIGGER trg_subject_completed AFTER INSERT ON subject_score DECLARE edate date; BEGIN SELECT os.end_date into edate FROM open_subject os JOIN subject_score ss ON os.subject_seq = ss.subject_seq WHERE sysdate >= os.end_date; IF SQL%ROWCOUNT = 0 THEN DBMS_OUTPUT.PUT_LINE('종강한 과목에 대한 성적만 입력할 수 있습니다.');</pre>			
<pre> RAISE_APPLICATION_ERROR(-20001, '종강한 과목에 대한 성적만 입력할 수 있습니다.');</pre>			
<pre> ROLLBACK; ELSE DBMS_OUTPUT.PUT_LINE('성적 입력이 완료되었습니다.');</pre>			
<pre> COMMIT; END IF; END trg_subject_completed; /</pre>			

분류	트리거-6	작성일	2023-04-06
목적	무단 지각, 결석, 조퇴시, penalty 부여 누적 penalty가 15점 이상 될 시 강제퇴소(중도탈락) 처리	작성자	김대환
PL/SQL			
<pre> CREATE OR REPLACE TRIGGER attendance_penalty_trigger AFTER INSERT OR UPDATE ON attendance_detail FOR EACH ROW DECLARE v_student_seq student.student_seq%TYPE; v_attendance_seq attendance_detail.attendance_seq%TYPE; v_check_in attendance_detail.check_in%TYPE; v_check_out attendance_detail.check_out%TYPE; v_penalty student.penalty%TYPE; v_total_penalty NUMBER; BEGIN v_attendance_seq := :NEW.attendance_seq; SELECT student_seq INTO v_student_seq FROM student WHERE student_seq = (SELECT student_seq FROM attendance WHERE attendance_seq = v_attendance_seq); IF v_student_seq IS NULL THEN RETURN; END IF; v_check_in := :NEW.check_in; </pre>			

```

v_check_out := :NEW.check_out;

IF v_check_in IS NULL AND v_check_out IS NULL THEN
    v_penalty := 3;
ELSIF v_check_in IS NOT NULL AND to_date(v_check_in, 'HH24:MI:SS') > to_date('09:10:00', 'HH24:MI:SS') THEN
    v_penalty := 1;
ELSIF v_check_out IS NOT NULL AND to_date(v_check_out, 'HH24:MI:SS') < to_date('17:50:00', 'HH24:MI:SS') THEN
    v_penalty := 1;
ELSE
    v_penalty := 0;
END IF;

SELECT penalty INTO v_total_penalty FROM student WHERE student_seq = v_student_seq;

IF v_total_penalty IS NULL THEN
    v_total_penalty := 0;
END IF;

v_total_penalty := v_total_penalty + v_penalty;

IF v_total_penalty >= 15 THEN
    UPDATE student SET drop_out = '중도탈락' WHERE student_seq = v_student_seq;
END IF;

UPDATE student SET penalty = v_total_penalty WHERE student_seq = v_student_seq;
END;

```

분류	뷰-1	작성일	2023-04-05
목적	성적조회	작성자	김대환
PL/SQL			
<pre> CREATE VIEW course_scores AS SELECT DISTINCT cl.curs_name as "과정명", oc.begin_date as "과정기간(시작)", oc.end_date as "과정기간(끝)", sj.subject_name as "과목명", t.name as "교사명", b.subject as "교재명", mb.name as "교육생 이름", mb.ssn as "주민번호 뒷자리", ss.test_writescore as "필기", ss.test_realscore as "실기" FROM open_subject os INNER JOIN subject sj ON sj.subject_seq = os.subject_seq INNER JOIN curriculum_subject cs ON sj.subject_seq = cs.subject_seq INNER JOIN curriculum_list cl ON cs.curriculum_list_seq = cl.curriculum_list_seq INNER JOIN open_curs oc ON cl.curriculum_list_seq = oc.curriculum_list_seq INNER JOIN teacher t ON oc.teacher_seq = t.teacher_seq INNER JOIN subject_book sb ON sj.subject_seq = sb.subject_seq INNER JOIN book b ON sb.book_seq = b.book_seq INNER JOIN student st ON cl.curriculum_list_seq = st.curriculum_list_seq INNER JOIN member mb ON st.member_seq = mb.member_seq INNER JOIN subject_score ss ON st.student_seq = ss.student_seq; SELECT * FROM course_scores; </pre>			

분류	뷰-2	작성일	2023-04-05
목적	교육생정보조회 (관리자)	작성자	유동현
PL/SQL			
<pre> CREATE VIEW student_scores_view AS SELECT mb.name AS "이름", mb.ssn AS "주민번호 뒷자리", mb.tel AS "전화번호", cl.curs_name AS "과정명", st.enroll_date AS "과정 시작날짜", st.drop_out_date AS "과정 종료날짜", oc.room_seq AS "강의실", sj.subject_name AS "과목명",counsel os.begin_date AS "과목 시작날짜", os.end_date AS "과목 종료날짜", tc.name AS "교사명", sc.test_writescore AS "필기시험점수", sc.test_realscore AS "실기시험점수" FROM student st INNER JOIN member mb ON st.member_seq = mb.member_seq INNER JOIN curriculum_list cl ON st.curriculum_list_seq = cl.curriculum_list_seq LEFT OUTER JOIN open_curs oc ON st.curriculum_list_seq = oc.curriculum_list_seq LEFT OUTER JOIN open_subject os ON oc.open_curs_seq = os.open_curs_seq AND oc.teacher_seq = os.teacher_seq INNER JOIN subject sj ON os.subject_seq = sj.subject_seq </pre>			

```
INNER JOIN teacher tc ON os.teacher_seq = tc.teacher_seq
INNER JOIN subject_score sc ON st.student_seq = sc.student_seq AND st.curriculum_list_seq = sc.curriculum_list_seq AND sj.subject_seq =
sc.subject_seq;

SELECT * FROM student_scores_view;
```


분류	뷰-3	작성일	2023-04-05
목적	지정된 과정의 전체 상담 일지를 수강생별로 조회	작성자	김대환
PL/SQL			
<pre> CREATE VIEW counsel_info AS SELECT c.counsel_date AS "상담 날짜", tp.counsel_subject AS "상담 주제", m.name AS "학생 이름", c.counsel_text AS "상담 내용" FROM counsel c INNER JOIN teacher t ON c.teacher_seq = t.teacher_seq INNER JOIN open_curs o ON o.teacher_seq = c.teacher_seq INNER JOIN counsel_topic tp ON c.counsel_topic_seq = tp.counsel_topic_seq INNER JOIN student s ON c.student_seq = s.student_seq INNER JOIN member m ON s.member_seq = m.member_seq; SELECT * FROM counsel_info WHERE open_curs_seq = <과정 번호> AND name = <'학생 이름'> ORDER BY counsel_date DESC; </pre>			

분류	뷰-4	작성일	2023-04-05
목적	사후처리 조회 (교육생번호, 교육생, 교육과정 정보, 수료날짜, 취업현황)	작성자	김대환
PL/SQL			
<pre> CREATE VIEW job_info AS SELECT s.student_seq AS "학생 번호", m.name AS "이름", c.curs_name AS "과정명", f.field_name AS "분야", jc.company AS "회사명", jc.hire_date AS "입사일" FROM job_current jc INNER JOIN student s ON jc.student_seq = s.student_seq INNER JOIN Member m ON m.member_seq = s.member_seq INNER JOIN curriculum_list c ON s.curriculum_list_seq = c.curriculum_list_seq INNER JOIN job_field f ON jc.job_field_seq = f.job_field_seq WHERE jc.company IS NOT NULL; SELECT * FROM job_info; </pre>			

분류	뷰-5	작성일	2023-04-05
목적	관리자는 취업현황에 대한 재취업 지원 시스템을 제공하기 위해 미취업 교육생 목록을 열람할 수 있다. (최근 n개월 내)	작성자	김대환
PL/SQL			
<pre> CREATE OR REPLACE VIEW jobless_info AS SELECT s.student_seq AS "학생 번호", m.name AS "이름", c.curs_name AS "과정명", f.field_name AS "분야", jc.company AS "회사명", jc.hire_date AS "입사일" FROM job_current jc LEFT OUTER JOIN student s ON jc.student_seq = s.student_seq LEFT OUTER JOIN Member m ON m.member_seq = s.member_seq LEFT OUTER JOIN curriculum_list c ON s.curriculum_list_seq = c.curriculum_list_seq LEFT OUTER JOIN job_field f ON jc.job_field_seq = f.job_field_seq WHERE jc.company IS NULL; SELECT * FROM jobless_info j </pre>			

```
JOIN student s  
ON j."학생 번호" = s.student_seq  
WHERE MONTHS_BETWEEN(SYSDATE, s.drop_out_date) <= 12;
```

분류	뷰-6	작성일	2023-04-05
목적	강의 상세정보 출력	작성자	유동현
PL/SQL			
<pre> CREATE VIEW curriculum_subject_info AS SELECT c.curriculum_list_seq as "과정번호", o.curs_name as "과정명", p.subject_seq as "과목번호", t.subject_name as "과목이름", p.begin_date as "과정기간(시작)", p.end_date as "과정기간(끝)", o.room_seq as "강의실", b.subject as "과목명", o.student_limit as "수강정원" FROM curriculum_list c INNER JOIN open_curs o ON c.curriculum_list_seq = o.curriculum_list_seq INNER JOIN student s ON s.curriculum_list_seq = c.curriculum_list_seq INNER JOIN open_subject p ON o.open_curs_seq = p.open_curs_seq INNER JOIN subject t ON t.subject_seq = p.subject_seq </pre>			

```
INNER JOIN subject_book k
  ON k.subject_seq = t.subject_seq
INNER JOIN book b
  ON k.book_seq = b.book_seq;

select * from curriculum_subject_info ;
```

분류	뷰-7	작성일	2023-04-05
목적	과목 목록 출력	작성자	김대환
PL/SQL			
<pre> CREATE VIEW subject_score_info AS SELECT o.subject_seq as "과목번호", c.curs_name as "과정명", c.begin_date as "과정기간(시작)", c.end_date as "과정기간(종료)", room_seq as "강의실", s.subject_name as "과목이름", o.begin_date as "과목시작일", o.end_date as "과목종료일", b.subject as "교재", e.score_attend as "출석점수", e.score_writeTest as "필기점수", e.score_realTest as "실기점수" FROM subject s INNER JOIN open_subject o ON s.subject_seq = o.subject_seq INNER JOIN open_curs c ON c.open_curs_seq = o.open_curs_seq INNER JOIN subject_book t ON t.subject_seq = s.subject_seq </pre>			

```
INNER JOIN book b
ON t.book_seq = b.book_seq
INNER JOIN score e
ON e.subject_seq = s.subject_seq;

select *
from subject_score_info;
```


분류	뷰-8	작성일	2023-04-05
목적	과목상세정보 조회	작성자	유동현
PL/SQL			
<pre> CREATE VIEW subject_info AS SELECT o.subject_seq as "과목번호", c.curs_name as "과정명", c.begin_date as "과정기간(시작)", c.end_date as "과정기간(종료)", room_seq as "강의실", s.subject_name as "과목이름", o.begin_date as "과목시작일", o.end_date as "과목종료일", b.subject as "교재", e.score_attend as "출석점수", e.score_writeTest as "필기점수", e.score_realTest as "실기점수" FROM subject s INNER JOIN open_subject o ON s.subject_seq = o.subject_seq INNER JOIN open_curs c ON c.open_curs_seq = o.open_curs_seq INNER JOIN subject_book t </pre>			

```
    ON t.subject_seq = s.subject_seq  
INNER JOIN book b  
    ON t.book_seq = b.book_seq  
INNER JOIN score e  
    ON e.subject_seq = s.subject_seq;  
  
select * from subject_info;
```

분류	뷰-9	작성일	2023-04-05
목적	출결 현황 조회	작성자	김대환
PL/SQL			
<pre> CREATE VIEW teacher_attendance_info AS SELECT c.curriculum_list_seq as "과정번호", c.subject_seq as "과목번호", d.student_seq as "학생번호", a.attendance_date as "출석날짜" FROM teaching_subject s INNER JOIN teacher t ON s.teacher_seq = t.teacher_seq INNER JOIN curriculum_subject c ON c.subject_seq = s.subject_seq INNER JOIN subject_score e ON s.subject_seq = e.subject_seq INNER JOIN student d ON e.student_seq = d.student_seq INNER JOIN attendance a ON a.student_seq = d.student_seq; select * from teacher_attendance_info; </pre>			

분류	뷰-10	작성일	2023-04-05
목적	출결현황 조회	작성자	유동현
PL/SQL			
<pre> CREATE VIEW attendance_info AS SELECT c.curriculum_list_seq AS "과정번호", c.subject_seq AS "과목번호", d.student_seq AS "학생번호", a.attendance_date AS "출석날짜" FROM teaching_subject s INNER JOIN teacher t ON s.teacher_seq = t.teacher_seq INNER JOIN curriculum_subject c ON c.subject_seq = s.subject_seq INNER JOIN subject_score e ON s.subject_seq = e.subject_seq INNER JOIN student d ON e.student_seq = d.student_seq INNER JOIN attendance a ON a.student_seq = d.student_seq; SELECT * FROM attendance_info; </pre>			

분류	뷰-11	작성일	2023-04-05
목적	성적조회 출력될 정보는 과목번호, 과목명, 과목 기간(시작 연월일, 끝 연월일), 교재명, 교사명, 과목별 배점 정보(출결, 필기, 실기 배점), 과목별 성적 정보(출결, 필기, 실기 점수), 과목별 시험날짜, 시험문제	작성자	김대환
PL/SQL			
<pre> CREATE VIEW subject_score_view AS SELECT DISTINCT s.subject_seq AS "과목번호", s.subject_name AS "과목명", os.begin_date AS "과목시작일자", os.end_date AS "과목종료일자", b.subject AS "교재명", t.name AS "교사명", sc.score_attend AS "출결배점", sc.score_writetest AS "필기시험 배점", sc.score_realtest AS "실기시험 배점", ss.attend_score AS "출결 점수", ss.test_writescore AS "필기 점수", ss.test_realscore AS "실기 점수", td.test_date AS "과목별 시험날짜", q.quiz_content AS "시험문제" FROM Subject s INNER JOIN open_subject os ON s.subject_seq = os.subject_seq </pre>			

```
INNER JOIN subject_book sb ON sb.subject_seq = s.subject_seq
INNER JOIN book b ON b.book_seq = sb.book_seq
INNER JOIN teacher t ON os.teacher_seq = t.teacher_seq
INNER JOIN score sc ON s.subject_seq = sc.subject_seq
INNER JOIN subject_score ss ON s.subject_seq = ss.subject_seq
INNER JOIN student std ON std.student_seq = ss.student_seq
INNER JOIN test_date td ON td.subject_seq = s.subject_seq
INNER JOIN quiz q ON q.quiz_seq = td.quiz_seq;

SELECT * FROM subject_score_view WHERE "교육생 번호" = 10;
```

분류	인덱스-1	작성일	2023-04-05
인덱스명	ssn_index	작성자	조연우
PL/SQL			
<pre>CREATE INDEX ssn_index ON member(ssn);</pre> <p>학생 로그인 속도 향상을 위한 인덱스</p>			

분류	인덱스-2	작성일	2023-04-05
인덱스명	member_name_index	작성자	조연우
PL/SQL			
<pre>CREATE INDEX member_name_index ON member(name);</pre> <p>학생명 검색 조회 시 검색 속도 향상을 위한 인덱스</p>			

분류	인덱스-3	작성일	2023-04-05
인덱스명	test_date_index	작성자	조연우
PL/SQL			
<pre>CREATE INDEX test_date_index ON test_date(test_date);</pre> <p>시행 날짜별 시험 조회 속도 향상을 위한 인덱스</p>			

분류	인덱스-4	작성일	2023-04-05
인덱스명	attendance_date_index	작성자	조연우
PL/SQL			
<pre>CREATE INDEX attendance_date_index ON attendance(attendance_date);</pre> <p>날짜별 출결 조회를 위한 인덱스</p>			

분류	인덱스-5	작성일	2023-04-05
인덱스명	subject_name_index	작성자	조연우
PL/SQL			
<pre>CREATE INDEX subject_name_index ON subject(subject_name);</pre> <p>과목 이름을 통한 검색 속도를 향상하기 위한 인덱스</p>			