**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 | | | |
| 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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | B-01 | 작성일 | 2023-04-04 |
| 요구사항명 | 기초 정보 관리 기능 | 작성자 | 1조 |
| DML | | | |
| --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  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 | | | |
| --1. 교사 정보 등록(이름, 주민번호 뒷자리, 전화번호, 강의 가능 과목을 입력하여 교사정보를 등록)  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;  --2. 교사 정보 수정  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  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 | | | |
| --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  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 sutdent.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 | | | |
| --5. 특정 개설 과정 선택 시 개설 과목 신규 등록을 할 수 있다.  -- 개설 과목 정보에 대한 입력, 수정, 삭제, 조회 기능  --입력  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;  --수정  CREATE OR REPLACE PROCEDURE curriculum\_subject\_update (curriculum\_list\_seq in number, subject\_seq in number)  IS  BEGIN  UPDATE curriculum\_subject  SET  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 | | | |
| --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  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));  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 | | | |
| 1. 시험 관리  - 특정 개설 과정 선택 시 등록된 개설 과목 정보를 출력하고, 과목별 성적 등록 여부를 확인할 수 있다.  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  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 | | | |
| 1. 개설 과정 선택 시 모든 교육생의 출결을 조회할 수 있다.  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  ) 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 | | | |
| 1. 지원생 등록(면접을 지원한 지원생들의 이름, 지원일자, 면접 예정일자, 지원동기, 지원과정 등을 등록한다)  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;  2. 교육생 선발  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  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 | | | |
| -- 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;  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 | | | |
| 1. 교재 등록  - 기초 정보 교재 목록에 교재를 추가할 수 있다.  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)  is  begin  insert into Book(book\_seq,subject,author,publisher,pub\_year) values(p\_seq, p\_subject, p\_author, p\_publisher, p\_pub\_year);  commit;  dbms\_output.put\_line('INSERT SUCCESS');  end Book\_insert\_p;  - 개설 과목에 교재 등록 시 기초정보 교재 목록에서 선택적으로 추가할 수 있다.  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)  is  begin  insert into Subject\_Book(subject\_seq,book\_seq) values (p\_subject\_seq, p\_book\_seq);  commit;  dbms\_output.put\_line('INSERT SUCCESS');  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 | | | |
| 1. 사후처리 등록  – 선언  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;  /  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 | | | |
| - 협력 기업 등록  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.');  COMMIT;  EXCEPTION WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('등록에 실패했습니다.');  ROLLBACK;  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 | | | |
| --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  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 | | | |
| --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;  --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 | | | |
| 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.');  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 | | | |
| 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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | C-01 | 작성일 | 2023-04-05 |
| 요구사항명 | 강의 스케줄 조회 | 작성자 | 이동재 |
| DML | | | |
| --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",  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 | | | |
| 1. 강의를 마친 과목에 대한 성적 처리를 위한 배점 입출력  -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.)  – 선언  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.');  COMMIT;    EXCEPTION WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('등록에 실패했습니다.');  ROLLBACK;  END score\_insert\_p;  – 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 | | | |
| -교사는 자신이 강의를 마친 과목의 목록 중에서 특정 과목을 선택하면, 교육생 정보가 출력된다.  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 || ', 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 | | | |
| 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)   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  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 | | | |
| 1. 상담 일지 작성  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.');  end counsel\_insert;  2. 상담 일지 조회  CREATE OR REPLACE PROCEDURE print\_counsel IS  BEGIN  FOR c IN (SELECT  counsel\_seq as "상담번호",  teacher\_seq as "교사번호",  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 | | | |
| - 과정을 선택하면 해당 과정을 수강한 전체 교육생의 교육생 번호, 교육생 이름, 취업 분야, 취업 회사, 취업일을 출력한다.  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  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 | | | |
| -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  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 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 | | | |
| – 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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | C-09 | 작성일 | 2023-04-05 |
| 요구사항명 | 퀴즈 관리 | 작성자 | 유동현 |
| DML | | | |
| --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.');  return;  end if;  update quiz  set quiz\_title = new\_title,  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 | | | |
| 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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | D-01 | 작성일 | 2023-04-05 |
| 요구사항명 | 성적 조회 | 작성자 | 이동재 |
| DML | | | |
| 1. 교육생이 로그인에 성공하면 교육생 개인의 정보와 교육생이 수강한 과정명, 과정 기간(시작 연월일, 끝 연월일), 강의실이 타이틀로 출력된다.  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  ) 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 | | | |
| 1. 매일 출석을 기록할 수 있어야 한다.(출근 1회, 퇴근 1회)  -- 3번교육생  --출결  — 선언문  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  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\_detatil\_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 | | | |
| 1. 교육생은 자신의 상담일지를 조회할 수 있다.  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;  2. 상담 날짜, 상담 교사, 상담 학생, 상담 내용이 출력된다.  CREATE OR REPLACE PROCEDURE counsel\_detail\_info(pstudent\_seq IN number)  AS  BEGIN  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 | | | |
| 1.취업 활동 등록  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.');  end job\_current\_update;  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\_hiredate\_update\_p; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | D-05 | 작성일 | 2023-04-05 |
| 요구사항명 | 교사 추천 도서 | 작성자 | 이동재 |
| DML | | | |
| 1.교사가 등록한 추천 도서 정보를 조회할 수 있다.  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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | D-06 | 작성일 | 2023-04-06 |
| 요구사항명 | 만남의 광장 | 작성자 | 김대환 |
| DML | | | |
| 1. 게시물 등록  – 선언문  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.  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 | | | |
| 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;  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 | | | |
| 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;    -- 구매에 필요한 포인트가 학생의 현재 포인트보다 크다면 INSERT 작업을 취소합니다.  IF v\_prize\_point > v\_ppoint THEN  RAISE\_APPLICATION\_ERROR(-20001, '포인트가 부족합니다.');  END IF;    -- 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 | | | |
| 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.');  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.');  END IF;  END; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 분류 | 트리거-3 | 작성일 | 2023-04-05 |
| 목적 | 성적 입력시 출결 30점, 필기 30점, 실기 40점 이하여야 한다. | 작성자 | 이채린 |
| PL/SQL | | | |
| 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.');  elsif :new.test\_writescore > 30 then  raise\_application\_error (-21002, '필기점수는 최대 30점입니다.');  elsif :new.test\_realscore > 40 then  raise\_application\_error (-21003, '실기점수는 최대 40점입니다.');  end if;    end trg\_subject\_score; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 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 | | | |
| 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.');  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 | | | |
| 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.');  RAISE\_APPLICATION\_ERROR(-20001, '종강한 과목에 대한 성적만 입력할 수 있습니다.');  ROLLBACK;  ELSE  DBMS\_OUTPUT.PUT\_LINE('성적 입력이 완료됐습니다.');  COMMIT;  END IF;  END trg\_subject\_completed;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 분류 | 트리거-6 | 작성일 | 2023-04-06 |
| 목적 | 무단 지각, 결석, 조퇴시, panalty 부여  누적 penalty가 15점 이상 될 시 강제퇴소(중도탈락) 처리 | 작성자 | 김대환 |
| PL/SQL | | | |
| 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;  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 | | | |
| 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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 분류 | 뷰-2 | 작성일 | 2023-04-05 |
| 목적 | 교육생정보조회 (관리자) | 작성자 | 유동현 |
| PL/SQL | | | |
| 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  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 | | | |
| 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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 분류 | 뷰-4 | 작성일 | 2023-04-05 |
| 목적 | 사후처리 조회 (교육생번호, 교육생, 교육과정 정보, 수료날짜, 취업현황) | 작성자 | 김대환 |
| PL/SQL | | | |
| 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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 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 | | | |
| 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  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 | | | |
| 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  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 | | | |
| 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  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 | | | |
| 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  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 | | | |
| 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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 분류 | 뷰-10 | 작성일 | 2023-04-05 |
| 목적 | 출결현황 조회 | 작성자 | 유동현 |
| PL/SQL | | | |
| 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; | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 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 | | | |
| 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  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 | | | |
| CREATE INDEX ssn\_index ON member(ssn);  학생 로그인 속도 향상을 위한 인덱스 | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 분류 | 인덱스-2 | 작성일 | 2023-04-05 |
| 인덱스명 | member\_name\_index | 작성자 | 조연우 |
| PL/SQL | | | |
| CREATE INDEX member\_name\_index ON member(name);  학생명 검색 조회 시 검색 속도 향상을 위한 인덱스 | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 분류 | 인덱스-3 | 작성일 | 2023-04-05 |
| 인덱스명 | test\_date\_index | 작성자 | 조연우 |
| PL/SQL | | | |
| CREATE INDEX test\_date\_index ON test\_date(test\_date);  시행 날짜별 시험 조회 속도 향상을 위한 인덱스 | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 분류 | 인덱스-4 | 작성일 | 2023-04-05 |
| 인덱스명 | attendance\_date\_index | 작성자 | 조연우 |
| PL/SQL | | | |
| CREATE INDEX attendance\_date\_index ON attendance(attendance\_date);  날짜별 출결 조회를 위한 인덱스 | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 분류 | 인덱스-5 | 작성일 | 2023-04-05 |
| 인덱스명 | subject\_name\_index | 작성자 | 조연우 |
| PL/SQL | | | |
| CREATE INDEX subject\_name\_index ON subject(subject\_name);  과목 이름을 통한 검색 속도를 향상하기 위한 인덱스 | | | |