**DB Object 스크립트(PL-SQL)**

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | B-01 | 작성일 | 2025-02-06 |
| 요구사항명 | 관리자 정보 | 작성자 | 김우현 |
| DML | | | |
| --/\*추가\*/AdminInsert  create or replace procedure adminInsert(  pName Admin.AdminName%type,  pTel Admin.AdminTel%type,  pPw Admin.AdminPw%type  )  is  v\_AdminSeq NUMBER;  BEGIN  -- 관리자 정보 삽입  v\_AdminSeq := admin\_seq.NEXTVAL;  INSERT INTO Admin (AdminSeq, AdminName, AdminTel, AdminPw)  VALUES (v\_AdminSeq, pName, pTel, pPw);    DBMS\_OUTPUT.PUT\_LINE('추가된 관리자 번호: ' || v\_AdminSeq);  DBMS\_OUTPUT.PUT\_LINE('관리자 이름: ' || pName);  DBMS\_OUTPUT.PUT\_LINE('관리자 전화번호: ' || pTel);  DBMS\_OUTPUT.PUT\_LINE('관리자 비밀번호: ' || pPw);    EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --adminInsert(관리자이름, 관리자전화번호, 관리자비밀번호);  begin  adminInsert('김우현', '010-9193-3250', 1493870);  end;  /  --/\*조회\*/AdminRead  CREATE OR REPLACE PROCEDURE adminRead IS  CURSOR admin\_cursor IS  SELECT AdminSeq ,  AdminName,  AdminTel ,  AdminPw  FROM Admin;  v\_AdminSeq Admin.AdminSeq%TYPE;  v\_AdminName Admin.AdminName%TYPE;  v\_AdminTel Admin.AdminTel%TYPE;  v\_AdminPw Admin.AdminPw%TYPE;  BEGIN  OPEN admin\_cursor; -- 커서를 여는 부분  LOOP  FETCH admin\_cursor INTO v\_AdminSeq, v\_AdminName, v\_AdminTel, v\_AdminPw; -- 커서로부터 값을 읽어옴  EXIT WHEN admin\_cursor%NOTFOUND; -- 더 이상 읽을 것이 없으면 루프 종료  -- 결과 출력  DBMS\_OUTPUT.PUT\_LINE('관리자 번호: ' || v\_AdminSeq ||  ', 관리자 이름: ' || v\_AdminName ||  ', 전화번호: ' || v\_AdminTel ||  ', 비밀번호: ' || v\_AdminPw);  END LOOP;  CLOSE admin\_cursor; -- 커서를 닫는 부분  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --adminRead();  begin  adminRead();  end;  /  --/\*수정\*/AdminUpdate  CREATE OR REPLACE PROCEDURE adminUpdate(  p\_AdminSeq Admin.AdminSeq%TYPE,  p\_AdminName Admin.AdminName%TYPE,  p\_AdminTel Admin.AdminTel%TYPE,  p\_AdminPw Admin.AdminPw%TYPE  ) IS  BEGIN  -- 관리자 정보 업데이트  UPDATE Admin  SET AdminName = p\_AdminName,  AdminTel = p\_AdminTel,  AdminPw = p\_AdminPw  WHERE AdminSeq = p\_AdminSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('관리자 번호 ' || p\_AdminSeq || '가(이) 성공적으로 업데이트되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('관리자 번호 ' || p\_AdminSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --adminUpdate(관리자번호,관리자이름,관리자전화번호,관리자비밀번호);  begin  adminUpdate(1, '김우현', '010-9193-3250', 1493870);  end;  /  -- /\*삭제\*/AdminDELETE  CREATE OR REPLACE PROCEDURE AdminDELETE(  p\_AdminSeq Admin.AdminSeq%TYPE  ) IS  BEGIN  -- 관리자 정보 업데이트  UPDATE Admin  SET AdminName = '없음',  AdminTel = '없음',  AdminPw = 0  WHERE AdminSeq = p\_AdminSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('관리자 순서 번호 ' || p\_AdminSeq || '가(이) 성공적으로 초기화되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('관리자 순서 번호 ' || p\_AdminSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  -- AdminDELETE(관리자번호);  begin  AdminDELETE(1);  end;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | B-02 | 작성일 | 2025-02-06 |
| 요구사항명 | 기초정보(과정) 관리 | 작성자 | 김우현 |
| DML | | | |
| --/\*추가\*/CourseInsert  create or replace procedure CourseInsert(  pCourseName Course.courseName%type  )  is  v\_CourseSeq NUMBER;  BEGIN  -- 관리자 정보 삽입  v\_CourseSeq := course\_seq.NEXTVAL;  INSERT INTO Course (courseSeq,courseName)  VALUES (v\_CourseSeq,pCourseName);    DBMS\_OUTPUT.PUT\_LINE('추가된 과정 번호: ' || v\_CourseSeq);  DBMS\_OUTPUT.PUT\_LINE('추가된 과정 이름: ' || pCourseName);      EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --CourseInsert(과정명)  Begin  CourseInsert(q'[AWS와 Docker & Kubernetes를 활용한 Java Full-Stack 개발자 양성과정]');  end;  /  --/\*조회\*/CourseRead  CREATE OR REPLACE PROCEDURE CourseRead IS  CURSOR Course\_cursor IS  SELECT CourseSeq,  CourseName    FROM Course;  v\_CourseSeq Course.CourseSeq%TYPE;  v\_CourseName Course.CourseName%TYPE;    BEGIN  OPEN Course\_cursor; -- 커서를 여는 부분  LOOP  FETCH Course\_cursor INTO v\_CourseSeq, v\_CourseName; -- 커서로부터 값을 읽어옴  EXIT WHEN Course\_cursor%NOTFOUND; -- 더 이상 읽을 것이 없으면 루프 종료  -- 결과 출력  DBMS\_OUTPUT.PUT\_LINE('과정번호: ' || v\_CourseSeq ||  ', 과정이름: ' || v\_CourseName);  END LOOP;  CLOSE Course\_cursor; -- 커서를 닫는 부분  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --CourseRead()  Begin  CourseRead();  end;  /  --/\*수정\*/CourseUpdate  CREATE OR REPLACE PROCEDURE CourseUpdate(  p\_CourseSeq Course.CourseSeq%TYPE,  p\_CourseName Course.CourseName%TYPE    ) IS  BEGIN  -- 과정 정보 업데이트  UPDATE Course  SET  CourseName = p\_CourseName  WHERE CourseSeq = p\_CourseSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('과정 번호 ' || p\_CourseSeq || '가(이) 성공적으로 업데이트되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('과정 번호 ' || p\_CourseSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --CourseUpdate(1,과정명)  begin  CourseUpdate(1,q'[AWS와 Docker & Kubernetes를 활용한 Java Full-Stack 개발자 양성과정]');  end;  /  -- /\*삭제\*/CourseDelete  CREATE OR REPLACE PROCEDURE CourseDelete(  p\_CourseSeq Course.CourseSeq%TYPE  ) IS  BEGIN  -- 과정 정보 초기화  UPDATE Course  SET CourseName = '없음'  WHERE CourseSeq = p\_CourseSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('과정 번호 ' || p\_CourseSeq || '가(이) 성공적으로 초기화되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('과정 번호 ' || p\_CourseSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  -- CourseDelete(과정번호);  begin  CourseDelete(1);  end;  /  --2.과목명 관리  --/\*추가\*/SubjectInsert  create or replace procedure SubjectInsert(  p\_subjectName subject.subjectName%type,  p\_subjectEsn subject.subjectEsn%type  )  is  v\_subjectSeq NUMBER;  BEGIN  -- 과목 정보 삽입  v\_subjectSeq := subject\_seq.NEXTVAL;  INSERT INTO subject (subjectSeq,subjectName,subjectEsn)  VALUES (v\_subjectSeq,p\_subjectName,p\_subjectEsn);    DBMS\_OUTPUT.PUT\_LINE('추가된 과목 번호: ' || v\_CourseSeq);  DBMS\_OUTPUT.PUT\_LINE('추가된 과목 이름: ' || pCourseName);  DBMS\_OUTPUT.PUT\_LINE('추가된 과목 유형: ' || p\_subjectEsn);    EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --SubjectInsert(과목명,과목유형)  Begin  SubjectInsert(q'[java]','필수');  end;  /  --/\*조회\*/SubjectRead  CREATE OR REPLACE PROCEDURE SubjectRead IS  CURSOR Subject\_cursor IS  SELECT subjectSeq ,  subjectName,  subjectEsn    FROM Subject;  v\_SubjectSeq Subject.subjectSeq%TYPE;  v\_SubjectName Subject.subjectName%TYPE;  v\_SubjectEsn Subject.subjectEsn%TYPE;    BEGIN  OPEN Subject\_cursor; -- 커서를 여는 부분  LOOP  FETCH Subject\_cursor INTO v\_SubjectSeq, v\_SubjectName,v\_SubjectEsn; -- 커서로부터 값을 읽어옴  EXIT WHEN Subject\_cursor%NOTFOUND; -- 더 이상 읽을 것이 없으면 루프 종료  -- 결과 출력  DBMS\_OUTPUT.PUT\_LINE('과목번호: ' || v\_SubjectSeq ||  ', 과목이름: ' || v\_SubjectName ||  ', 과목유형: ' || v\_SubjectEsn);  END LOOP;  CLOSE Subject\_cursor; -- 커서를 닫는 부분  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --SubjectRead()  Begin  SubjectRead();  end;  /  --/\*수정\*/SubjectUpdate  CREATE OR REPLACE PROCEDURE SubjectUpdate(  p\_SubjectSeq Subject.SubjectSeq%TYPE,  p\_SubjectName Subject.SubjectName%TYPE,  p\_SubjectEsn Subject.SubjectEsn%TYPE  ) IS  BEGIN  -- 과정 정보 업데이트  UPDATE Subject  SET  SubjectName = p\_SubjectName,  SubjectEsn = p\_SubjectEsn  WHERE SubjectSeq = p\_SubjectSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('과목 번호 ' || p\_SubjectSeq || '가(이) 성공적으로 업데이트되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('과목 번호 ' || p\_SubjectSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --SubjectUpdate(1,과목명,과목유형(필수/선택))  begin  SubjectUpdate(1,q'[java]','필수');  end;  /  -- /\*삭제\*/SubjectDelete  CREATE OR REPLACE PROCEDURE SubjectDelete(  p\_SubjectSeq Subject.SubjectSeq%TYPE  ) IS  BEGIN  -- 과목 정보 초기화  UPDATE Subject  SET SubjectName = '없음',  SubjectEsn = '없음'  WHERE SubjectSeq = p\_SubjectSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('과목 번호 ' || p\_SubjectSeq || '가(이) 성공적으로 초기화되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('과목 번호 ' || p\_SubjectSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  -- SubjectDelete(과목번호);  begin  SubjectDelete(1);  end;  /  --3.강의실명 관리  /\*추가\*/  insert into clsRoom(clsRoomSeq,clsRoomName,clsRoomPpl) values (clsRoom\_seq.NEXTVAL, q'[제 1 강의실]','30');  --/\*추가\*/ClsRoomInsert  create or replace procedure ClsRoomInsert(  p\_clsRoomName subject.clsRoomName%type,  p\_clsRoomPpl subject.clsRoomPpl%type  )  is  v\_clsRoomSeq NUMBER;  BEGIN  -- 과목 정보 삽입  v\_clsRoomSeq := clsRoom\_seq.NEXTVAL;  INSERT INTO clsRoom (clsRoomSeq,clsRoomName,clsRoomPpl)  VALUES (v\_subjectSeq,p\_clsRoomName,p\_clsRoomPpl);    DBMS\_OUTPUT.PUT\_LINE('추가된 강의실 번호: ' || v\_subjectSeq);  DBMS\_OUTPUT.PUT\_LINE('추가된 강의실 이름: ' || p\_clsRoomName);  DBMS\_OUTPUT.PUT\_LINE('추가된 강의실 수용인원: ' || p\_clsRoomPpl);    EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --ClsRoomInsert(강의실명,강의실수용인원)  Begin  ClsRoomInsert(q'[제 1 강의실]','30');  end;  /  --/\*조회\*/ClsRoomRead  CREATE OR REPLACE PROCEDURE ClsRoomRead IS  CURSOR clsRoom\_cursor IS  SELECT distinct  c.clsRoomName ,  c.clsRoomPpl ,  p.processCount  FROM  clsRoom c  INNER JOIN  process p ON c.clsRoomSeq = p.clsRoomSeq;    v\_ClsRoomName clsRoom.clsRoomName%TYPE;  v\_ClsRoomPpl clsRoom.clsRoomPpl%TYPE;  v\_ProcessCount process.processCount%TYPE;    BEGIN  OPEN ClsRoom\_cursor; -- 커서를 여는 부분  LOOP  FETCH ClsRoom\_cursor INTO v\_ClsRoomName,v\_ClsRoomPpl,v\_ProcessCount; -- 커서로부터 값을 읽어옴  EXIT WHEN ClsRoom\_cursor%NOTFOUND; -- 더 이상 읽을 것이 없으면 루프 종료  -- 결과 출력  DBMS\_OUTPUT.PUT\_LINE('강의실이름: ' || v\_ClsRoomName ||  ', 강의실수용인원: ' || v\_ClsRoomPpl ||  ', 수강한인원: ' || v\_ProcessCount);  END LOOP;  CLOSE ClsRoom\_cursor; -- 커서를 닫는 부분  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --ClsRoomRead()  Begin  ClsRoomRead();  end;  /  --/\*수정\*/ClsRoomUpdate  CREATE OR REPLACE PROCEDURE ClsRoomUpdate(  p\_ClsRoomSeq ClsRoom.ClsRoomSeq%TYPE,  p\_ClsRoomName ClsRoom.ClsRoomName%TYPE,  p\_ClsRoomPpl ClsRoom.ClsRoomPpl%TYPE  ) IS  BEGIN  -- 강의실 정보 업데이트  UPDATE ClsRoom  SET  ClsRoomName = p\_ClsRoomName,  ClsRoomPpl = p\_ClsRoomPpl    WHERE ClsRoomSeq = p\_ClsRoomSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('강의실 번호 ' || p\_ClsRoomSeq || '가(이) 성공적으로 업데이트되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('강의실 번호 ' || p\_ClsRoomSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --ClsRoomUpdate(강의실번호,강의실명,강의실수용인원)  begin  ClsRoomUpdate(1,q'[제 1 강의실]',30);  end;  /  -- /\*삭제\*/ClsRoomDelete  CREATE OR REPLACE PROCEDURE ClsRoomDelete(  p\_ClsRoomSeq ClsRoom.ClsRoomSeq%TYPE  ) IS  BEGIN  -- 선택한 강의실 정보 초기화(삭제)  UPDATE ClsRoom  SET ClsRoomName = '없음',  ClsRoomPpl = 0  WHERE ClsRoomSeq = p\_ClsRoomSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('강의실 번호 ' || p\_ClsRoomSeq || '가(이) 성공적으로 초기화되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('강의실 번호 ' || p\_ClsRoomSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  -- ClsRoomDelete(강의실번호);  begin  ClsRoomDelete(1);  end;  /  --4.교재명 관리  --/\*추가\*/BookInsert  create or replace procedure BookInsert(  p\_BookName book.bookName%type,  p\_BookPublisher book.bookPublisher%type,  p\_BookAuthor book.bookAuthor%type,  p\_BookYear book.bookYear%type  )  is  v\_BookSeq NUMBER;  BEGIN  -- 교재 정보 추가등록  v\_BookSeq := book\_seq.NEXTVAL;  INSERT INTO book(bookSeq,bookName,bookPublisher,bookAuthor,bookYear)  VALUES (v\_BookSeq,p\_BookName,p\_BookPublisher,p\_BookAuthor,p\_BookYear);    DBMS\_OUTPUT.PUT\_LINE('추가등록된 교재 번호: ' || v\_BookSeq);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 교재 이름: ' || p\_BookName);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 교재 출판사: ' || p\_BookPublisher);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 교재 작가: ' || p\_BookAuthor);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 교재 발행연도: ' || p\_BookYear);    EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --BookInsert(교재명,출판사,작가,발행연도)  Begin  BookInsert(q'[최신Java 프로그래밍]', q'[21세기사]', q'[한정란]', to\_date('2024','YYYY'));  end;  /  --/\*조회\*/BookRead  CREATE OR REPLACE PROCEDURE BookRead IS  CURSOR Book\_cursor IS  SELECT  bookseq,  bookName,  bookPublisher,  bookAuthor,  bookYear  FROM  book;  v\_Bookseq book.bookseq%TYPE;  v\_BookName book.bookName%TYPE;  v\_BookPublisher book.bookPublisher%TYPE;  v\_BookAuthor book.bookAuthor%TYPE;  v\_BookYear book.bookYear%TYPE;  BEGIN  OPEN Book\_cursor; -- 커서를 여는 부분  LOOP  FETCH Book\_cursor INTO v\_Bookseq,v\_BookName,v\_BookPublisher,v\_BookAuthor,v\_BookYear; -- 커서로부터 값을 읽어옴  EXIT WHEN Book\_cursor%NOTFOUND; -- 더 이상 읽을 것이 없으면 루프 종료    -- 결과 출력  DBMS\_OUTPUT.PUT\_LINE('교재번호: ' || v\_BookSeq ||  ', 교재이름: ' || v\_BookName ||  ', 강의실수용인원: ' || v\_BookPublisher ||  ', 수강한인원: ' || v\_BookAuthor ||  ', 강의실수용인원: ' || v\_BookYear);  END LOOP;  CLOSE Book\_cursor; -- 커서를 닫는 부분  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --BookRead()  Begin  BookRead();  end;  /  update book  set bookName = q'[최신Java 프로그래밍]'  , bookPublisher = q'[21세기사]'  , bookAuthor = q'[한정란]'  , bookYear = to\_date('2024','YYYY')  where bookSeq = 1;  --/\*수정\*/BookUpdate  CREATE OR REPLACE PROCEDURE BookUpdate(  p\_bookSeq book.bookSeq%TYPE,  p\_bookName book.bookName%TYPE,  p\_bookPublisher book.bookPublisher%TYPE,  p\_bookAuthor book.bookAuthor%TYPE,  p\_bookYear book.bookYear%TYPE  ) IS  BEGIN  -- 교재 정보 업데이트  UPDATE book  SET  bookName = p\_bookName,  bookPublisher = p\_bookPublisher,  bookAuthor = p\_bookAuthor,  bookYear = p\_bookYear    WHERE bookSeq = p\_bookSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('교재 번호 ' || p\_bookSeq || '가(이) 성공적으로 업데이트되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('교재 번호 ' || p\_bookSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --BookUpdate()  Begin  BookUpdate(1,q'[최신Java 프로그래밍]',q'[21세기사]',q'[한정란]',to\_date('2024','YYYY'));  end;  /  -- /\*삭제\*/BookDelete  CREATE OR REPLACE PROCEDURE BookDelete(  p\_BookSeq book.bookSeq%TYPE  ) IS  BEGIN  -- 선택한 강의실 정보 초기화(삭제)  UPDATE book  SET bookName = '없음'  , bookPublisher = null  , bookAuthor = null  , bookYear = null  WHERE bookSeq = p\_BookSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('교재번호 ' || p\_BookSeq || '가(이) 성공적으로 초기화되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('교재번호 ' || p\_BookSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  -- BookDelete(교재번호);  begin  BookDelete(1);  end;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | B-03 | 작성일 | 2025-02-06 |
| 요구사항명 | 교사 계정 관리 기능 | 작성자 | 채상혁 |
| DML | | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | B-04 | 작성일 | 2025-02-06 |
| 요구사항명 | 개설 과정 관리 | 작성자 | 채상혁 |
| DML | | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | B-05 | 작성일 | 2025-02-06 |
| 요구사항명 | 개설 과목 관리 기능 | 작성자 | 김우현 |
| DML | | | |
| --/\*추가\*/  begin  SubjectRead();  BookRead();  prcSubject();    end;  /  create or replace procedure prcSubjectInsert(  p\_processSeq prcSubject.processSeq%type,  p\_subjectSeq prcSubject.subjectSeq%type,  p\_prcSubjectSDate prcSubject.prcSubjectSDate%type,  p\_prcSubjectEDate prcSubject.prcSubjectEDate%type  )  is  v\_prcSubjectSeq NUMBER;  BEGIN  -- 교재 정보 추가등록  v\_prcSubjectSeq := prcSubject\_seq.NEXTVAL;  INSERT INTO prcSubject(prcSubjectSeq,processSeq,subjectSeq,prcSubjectSDate,prcSubjectEDate)  VALUES (v\_prcSubjectSeq,p\_processSeq,p\_subjectSeq,p\_prcSubjectSDate,p\_prcSubjectEDate);    DBMS\_OUTPUT.PUT\_LINE('추가등록된 개설과정과목번호: ' || v\_prcSubjectSeq);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 개설과정번호: ' || p\_processSeq);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 과목번호: ' || p\_subjectSeq);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 과목시작날짜: ' || p\_prcSubjectSDate);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 과목종료날짜: ' || p\_prcSubjectEDate);    EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --prcSubjectInsert(교재명,출판사,작가,발행연도)  Begin  prcSubjectInsert(2,1, to\_date('24/08/07','YY-MM-DD'), to\_date('24/09/11','YY-MM-DD'));  end;  /  create or replace procedure teacherInsert(  p\_teacherName teacher.teacherName%type,  p\_teacherPw teacher.teacherPw%type,  p\_teacherTel teacher.teacherTel%type  )  is  v\_teacherSeq NUMBER;  BEGIN  -- 교재 정보 추가등록  v\_teacherSeq := teacher\_seq.NEXTVAL;  INSERT INTO teacher(teacherSeq,teacherName,teacherPw,teacherTel)  VALUES (v\_teacherSeq,p\_teacherName,p\_teacherPw,p\_teacherTel);    DBMS\_OUTPUT.PUT\_LINE('추가등록된 교사번호: ' || v\_teacherSeq);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 교사이름: ' || p\_teacherName);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 교사비밀번호: ' || p\_teacherPw);  DBMS\_OUTPUT.PUT\_LINE('추가등록된 교사전화번호: ' || p\_teacherTel);    EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --teacherInsert(교사번호,교사이름,교사비밀번호,교사전화번호)  Begin  teacherInsert('구하늘',1156493,'010-6789-0123');  end;  /  CREATE OR REPLACE PROCEDURE prcSubjectRead AS  CURSOR subject\_cursor IS  SELECT  p.processSeq,  p.prcSubjectSeq,  p.subjectSeq,  s.subjectName,  p.prcSubjectSDate,  p.prcSubjectEDate,  t.teacherName,  b.bookName  FROM  prcSubject p  INNER JOIN  subject s ON p.subjectSeq = s.subjectSeq  INNER JOIN  process pr ON p.processSeq = pr.processSeq  INNER JOIN  teacher t ON pr.teacherSeq = t.teacherSeq  INNER JOIN  sbjectBook sb ON s.subjectSeq = sb.subjectSeq  INNER JOIN  book b ON sb.bookSeq = b.bookSeq;  rec subject\_cursor%ROWTYPE; -- 커서의 레코드 타입 정의  BEGIN  OPEN subject\_cursor; -- 커서 열기  LOOP  FETCH subject\_cursor INTO rec; -- 데이터 가져오기  EXIT WHEN subject\_cursor%NOTFOUND; -- 모든 데이터를 읽었는지 확인  DBMS\_OUTPUT.PUT\_LINE('개설과정번호: ' || rec.processSeq ||  ', 개설과정과목번호: ' || rec.prcSubjectSeq ||  ', 과목번호: ' || rec.subjectSeq ||  ', 과목명: ' || rec.subjectName ||  ', 과목시작날짜: ' || rec.prcSubjectSDate ||  ', 과목종료날짜: ' || rec.prcSubjectEDate ||  ', 교사명: ' || rec.teacherName ||  ', 교재명: ' || rec.bookName);  END LOOP;  CLOSE subject\_cursor; -- 커서 닫기  END prcSubjectRead;  /  BEGIN  prcSubjectRead();  END;  /  /\*조회\*/ --개설과정과목조회  CREATE OR REPLACE PROCEDURE prcSubjectRead IS  CURSOR prcSubject\_cursor IS  SELECT  prcSubjectseq,  processSeq,  subjectSeq,  prcSubjectSDate,  prcSubjectEDate  FROM  prcSubject;  v\_prcSubjectseq prcSubject.prcSubjectseq%TYPE;  v\_processSeq prcSubject.processSeq%TYPE;  v\_subjectSeq prcSubject.subjectSeq%TYPE;  v\_prcSubjectSDate prcSubject.prcSubjectSDate%TYPE;  v\_prcSubjectEDate prcSubject.prcSubjectEDate%TYPE;  BEGIN  OPEN prcSubject\_cursor; -- 커서를 여는 부분  LOOP  FETCH prcSubject\_cursor INTO v\_prcSubjectseq,v\_processSeq,v\_subjectSeq,v\_prcSubjectSDate,v\_prcSubjectEDate; -- 커서로부터 값을 읽어옴  EXIT WHEN prcSubject\_cursor%NOTFOUND; -- 더 이상 읽을 것이 없으면 루프 종료    -- 결과 출력  DBMS\_OUTPUT.PUT\_LINE('개설과정과목번호: ' || v\_prcSubjectseq ||  ', 과정번호: ' || v\_processSeq ||  ', 과목번호: ' || v\_subjectSeq ||  ', 개설과목시작일: ' || v\_prcSubjectSDate ||  ', 개설과목종료일: ' || v\_prcSubjectEDate);  END LOOP;  CLOSE prcSubject\_cursor; -- 커서를 닫는 부분  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --prcSubjectRead()  Begin  prcSubjectRead();  end;  /  /\*수정\*/--개설과정과목수정  CREATE OR REPLACE PROCEDURE prcSubjectUpdate(  p\_prcSubjectSeq prcSubject.prcSubjectSeq%TYPE,  p\_processSeq prcSubject.processSeq%TYPE,  p\_subjectSeq prcSubject.subjectSeq%TYPE,  p\_prcSubjectSDate prcSubject.prcSubjectSDate%TYPE,  p\_prcSubjectEDate prcSubject.prcSubjectEDate%TYPE  ) IS  BEGIN  -- 교재 정보 업데이트  UPDATE prcSubject  SET  processSeq = p\_processSeq,  subjectSeq = p\_subjectSeq,  prcSubjectSDate = p\_prcSubjectSDate,  prcSubjectEDate = p\_prcSubjectEDate    WHERE prcSubjectSeq = p\_prcSubjectSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('교재 번호 ' || p\_prcSubjectSeq || '가(이) 성공적으로 업데이트되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('교재 번호 ' || p\_prcSubjectSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  --prcSubjectUpdate()  Begin  prcSubjectUpdate(1,2,1, to\_date('24/08/07','YY-MM-DD'), to\_date('24/09/11','YY-MM-DD'));  end;  /  -- /\*삭제\*/prcSubjectDelete  CREATE OR REPLACE PROCEDURE prcSubjectDelete(  p\_prcSubjectSeq prcSubject.prcSubjectSeq%TYPE  ) IS  BEGIN  -- 선택한 강의실 정보 초기화(삭제)  UPDATE prcSubject  SET  prcSubjectSDate = null  , prcSubjectEDate = null  WHERE prcSubjectSeq = p\_prcSubjectSeq;  -- 변경된 행 수를 확인하고 출력  IF SQL%ROWCOUNT > 0 THEN  DBMS\_OUTPUT.PUT\_LINE('개설과정과목번호 ' || p\_prcSubjectSeq || '가(이) 성공적으로 초기화되었습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE('개설과정과목번호 ' || p\_prcSubjectSeq || '은(는) 존재하지 않거나 업데이트할 데이터가 없습니다.');  END IF;  EXCEPTION  WHEN OTHERS THEN  ROLLBACK; -- 오류 발생 시 롤백  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  -- prcSubjectDelete(개설과정과목번호);  begin  prcSubjectDelete(1);  end;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | B-06 | 작성일 | 2025-02-06 |
| 요구사항명 | 교육생 관리 기능 | 작성자 | 김우현 |
| DML | | | |
| /\*추가\*/--1. 학생 정보 입력  CREATE OR REPLACE PROCEDURE StudentInsert(  p\_studentName IN VARCHAR2,  p\_studentPw IN NUMBER,  p\_studentTel IN VARCHAR2,  p\_studentDate IN DATE  )  IS  v\_studentSeq NUMBER;  BEGIN  SELECT ststatus\_seq1.NEXTVAL INTO v\_studentSeq FROM DUAL;  INSERT INTO Student (studentSeq, studentName, studentPw, studentTel, studentDate)  VALUES (v\_studentSeq, p\_studentName, p\_studentPw, p\_studentTel, p\_studentDate);  COMMIT;  END;  /  select studentName as 이름, studentPw as 주민번호, studentTel as 전화번호, studentDate as 등록일  from Student;  BEGIN  StudentInsert('이영희', 2345678, '010-2345-6789', '2024-06-03');  END;  /  /\*조회\*/--2  CREATE OR REPLACE PROCEDURE studentRead  IS  CURSOR student\_cursor IS  SELECT studentName AS 이름, studentPw AS 주민번호, studentTel AS 전화번호, studentDate AS 등록일  FROM Student;    v\_studentName VARCHAR2(50);  v\_studentPw NUMBER;  v\_studentTel VARCHAR2(20);  v\_studentDate DATE;  BEGIN  OPEN student\_cursor;  LOOP  FETCH student\_cursor INTO v\_studentName, v\_studentPw, v\_studentTel, v\_studentDate;  EXIT WHEN student\_cursor%NOTFOUND;    DBMS\_OUTPUT.PUT\_LINE('이름: ' || v\_studentName);  DBMS\_OUTPUT.PUT\_LINE('주민번호: ' || v\_studentPw);  DBMS\_OUTPUT.PUT\_LINE('전화번호: ' || v\_studentTel);  DBMS\_OUTPUT.PUT\_LINE('등록일: ' || TO\_CHAR(v\_studentDate, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('-----------------------');  END LOOP;  CLOSE student\_cursor;  END;  /  BEGIN  studentRead();  END;  /  /\*조회\*/--3  CREATE OR REPLACE PROCEDURE studentInfoRead(  P\_studentSeq student.studentSeq%TYPE  )  IS  CURSOR student\_info\_cursor IS  SELECT  st.studentname AS 이름,  c.coursename AS 과정명,  pr.processSDate AS 과정시작날짜,  pr.processEDate AS 과정종료날짜,  clR.clsroomname AS 강의실,  CASE  WHEN stS.Status IS NULL THEN '수료중'  ELSE stS.Status  END AS 수료여부,  stS.stStatusDate AS 날짜  FROM Student st  INNER JOIN studentCls stcl ON st.studentSeq = stcl.studentSeq  INNER JOIN process pr ON stcl.processSeq = pr.processSeq  INNER JOIN Course c ON pr.courseSeq = c.courseSeq  INNER JOIN clsRoom clR ON pr.clsRoomSeq = clR.clsRoomSeq  LEFT JOIN stStatus stS ON st.studentSeq = stS.studentSeq  WHERE st.studentSeq = P\_studentSeq;    v\_name VARCHAR2(50);  v\_courseName VARCHAR2(100);  v\_startDate DATE;  v\_endDate DATE;  v\_classroom VARCHAR2(50);  v\_status VARCHAR2(20);  v\_statusDate DATE;  BEGIN  OPEN student\_info\_cursor;  LOOP  FETCH student\_info\_cursor INTO v\_name, v\_courseName, v\_startDate, v\_endDate, v\_classroom, v\_status, v\_statusDate;  EXIT WHEN student\_info\_cursor%NOTFOUND;    DBMS\_OUTPUT.PUT\_LINE('이름: ' || v\_name);  DBMS\_OUTPUT.PUT\_LINE('과정명: ' || v\_courseName);  DBMS\_OUTPUT.PUT\_LINE('과정 시작날짜: ' || TO\_CHAR(v\_startDate, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('과정 종료날짜: ' || TO\_CHAR(v\_endDate, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('강의실: ' || v\_classroom);  DBMS\_OUTPUT.PUT\_LINE('수료여부: ' || v\_status);  DBMS\_OUTPUT.PUT\_LINE('날짜: ' || TO\_CHAR(v\_statusDate, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('-----------------------');  END LOOP;  CLOSE student\_info\_cursor;  END;  /  BEGIN  studentInfoRead(1);  END;  /  --4  CREATE OR REPLACE PROCEDURE stInfoReadName(  P\_studentName student.studentName%TYPE  )  IS  CURSOR student\_info\_cursor IS  SELECT studentName AS 이름, studentPw AS 주민번호, studentTel AS 전화번호, studentDate AS 등록일  FROM student  WHERE studentName = P\_studentName;    v\_name student.studentName%TYPE;  v\_pw student.studentPw%TYPE;  v\_tel student.studentTel%TYPE;  v\_date student.studentDate%TYPE;  BEGIN  OPEN student\_info\_cursor;  LOOP  FETCH student\_info\_cursor INTO v\_name, v\_pw, v\_tel, v\_date;  EXIT WHEN student\_info\_cursor%NOTFOUND;    DBMS\_OUTPUT.PUT\_LINE('이름: ' || v\_name);  DBMS\_OUTPUT.PUT\_LINE('주민번호: ' || v\_pw);  DBMS\_OUTPUT.PUT\_LINE('전화번호: ' || v\_tel);  DBMS\_OUTPUT.PUT\_LINE('등록일: ' || TO\_CHAR(v\_date, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('-----------------------');  END LOOP;  CLOSE student\_info\_cursor;  END;  /  BEGIN  stInfoReadName('정수현');  END;  /  CREATE OR REPLACE PROCEDURE stInfoReadPw(  P\_studentPw student.studentPw%TYPE  )  IS  CURSOR student\_info\_cursor IS  SELECT studentName AS 이름, studentPw AS 주민번호, studentTel AS 전화번호, studentDate AS 등록일  FROM student  WHERE studentPw = P\_studentPw;    v\_name student.studentName%TYPE;  v\_pw student.studentPw%TYPE;  v\_tel student.studentTel%TYPE;  v\_date student.studentDate%TYPE;  BEGIN  OPEN student\_info\_cursor;  LOOP  FETCH student\_info\_cursor INTO v\_name, v\_pw, v\_tel, v\_date;  EXIT WHEN student\_info\_cursor%NOTFOUND;    DBMS\_OUTPUT.PUT\_LINE('이름: ' || v\_name);  DBMS\_OUTPUT.PUT\_LINE('주민번호: ' || v\_pw);  DBMS\_OUTPUT.PUT\_LINE('전화번호: ' || v\_tel);  DBMS\_OUTPUT.PUT\_LINE('등록일: ' || TO\_CHAR(v\_date, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('-----------------------');  END LOOP;  CLOSE student\_info\_cursor;  END;  /  BEGIN  stInfoReadPw('1234560');  END;  /  CREATE OR REPLACE PROCEDURE stInfoReadTel(  P\_studentTel student.studentTel%TYPE  )  IS  CURSOR student\_info\_cursor IS  SELECT studentName AS 이름, studentPw AS 주민번호, studentTel AS 전화번호, studentDate AS 등록일  FROM student  WHERE studentTel = P\_studentTel;    v\_name student.studentName%TYPE;  v\_pw student.studentPw%TYPE;  v\_tel student.studentTel%TYPE;  v\_date student.studentDate%TYPE;  BEGIN  OPEN student\_info\_cursor;  LOOP  FETCH student\_info\_cursor INTO v\_name, v\_pw, v\_tel, v\_date;  EXIT WHEN student\_info\_cursor%NOTFOUND;    DBMS\_OUTPUT.PUT\_LINE('이름: ' || v\_name);  DBMS\_OUTPUT.PUT\_LINE('주민번호: ' || v\_pw);  DBMS\_OUTPUT.PUT\_LINE('전화번호: ' || v\_tel);  DBMS\_OUTPUT.PUT\_LINE('등록일: ' || TO\_CHAR(v\_date, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('-----------------------');  END LOOP;  CLOSE student\_info\_cursor;  END;  /  BEGIN  stInfoReadTel('010-5678-9012');  END;  /  --5 /\*수정\*/  CREATE OR REPLACE PROCEDURE studentUpdate(  P\_studentSeq student.studentSeq%TYPE,  P\_studentName student.studentName%TYPE,  P\_studentPw student.studentPw%TYPE,  P\_studentTel student.studentTel%TYPE,  P\_studentDate student.studentDate%TYPE  )  IS  BEGIN  UPDATE student  SET studentName = P\_studentName,  studentPw = P\_studentPw,  studentTel = P\_studentTel,  studentDate = P\_studentDate  WHERE studentSeq = P\_studentSeq;  END;  /  BEGIN  studentUpdate(1, '김우현', 1520316, '010-4983-9012', '2025-02-04');  END;  /  --5 /\*수정\*/  CREATE OR REPLACE PROCEDURE stStatusUpdate(  P\_studentSeq stStatus.studentSeq%TYPE,  P\_status stStatus.status%TYPE,  P\_stStatusDate stStatus.stStatusDate%TYPE  )  IS  BEGIN  UPDATE stStatus  SET status = P\_status,  stStatusDate = P\_stStatusDate  WHERE stStatus.studentSeq = P\_studentSeq;  END;  /  BEGIN  stStatusUpdate(1, '중도탈락', '2025-02-04');  END;  /  /\*삭제\*/  --5  CREATE OR REPLACE PROCEDURE Studentdelete(  P\_studentSeq student.studentSeq%TYPE  )  IS  BEGIN  UPDATE student  SET studentName = NULL,  studentPw = NULL,  studentTel = NULL,  studentDate = NULL  WHERE studentSeq = P\_studentSeq;  END;  /  BEGIN  Studentdelete(1);  END;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | B-07 | 작성일 | 2025-02-06 |
| 요구사항명 | 시험 관리 및 성적 조회 가능 | 작성자 | 채상혁 |
| DML | | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | B-08 | 작성일 | 2025-02-06 |
| 요구사항명 | 출결 관리 및 출결 조회 기능 | 작성자 | 채상혁 |
| DML | | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | C-02 | 작성일 | 2025-02-06 |
| 요구사항명 | 강의 스케줄 조회 | 작성자 | 서희원 |
| DML | | | |
| /\* 교사 \*/  --a. 자신이 진행중인 강의스케줄 조회  CREATE OR REPLACE PROCEDURE teacherInfo (  pTeacherName IN VARCHAR2  )  IS  BEGIN  -- 강사의 강의 정보 조회 및 출력  FOR rec IN (  SELECT  t.teacherName AS 교사이름,  s.subjectName AS 과목명,  TO\_CHAR(ps.prcSubjectSDate, 'YYYY-MM-DD') AS 과목시작날짜,  TO\_CHAR(ps.prcSubjectEDate, 'YYYY-MM-DD') AS 과목종료날짜,  c.courseName AS 과정명,  TO\_CHAR(p.processSDate, 'YYYY-MM-DD') AS 과정시작날짜,  TO\_CHAR(p.processEDate, 'YYYY-MM-DD') AS 과정종료날짜,  b.bookName AS 교재명,  cr.clsRoomName AS 강의실명,  CASE  WHEN ps.prcSubjectSDate > SYSDATE THEN '강의예정'  WHEN ps.prcSubjectSDate <= SYSDATE AND ps.prcSubjectEDate >= SYSDATE THEN '강의중'  ELSE '강의종료'  END AS 강의진행여부,  p.processCount AS 수강인원  FROM teacher t  JOIN process p ON t.teacherSeq = p.teacherSeq  JOIN prcSubject ps ON p.processSeq = ps.processSeq  JOIN subject s ON ps.subjectSeq = s.subjectSeq  JOIN course c ON p.courseSeq = c.courseSeq  JOIN clsRoom cr ON p.clsRoomSeq = cr.clsRoomSeq  JOIN sbjectBook sb ON s.subjectSeq = sb.subjectSeq  JOIN book b ON sb.bookSeq = b.bookSeq  WHERE t.teacherName = pTeacherName  ORDER BY p.processSDate  ) LOOP  DBMS\_OUTPUT.PUT\_LINE('과정명: ' || rec.과정명);  DBMS\_OUTPUT.PUT\_LINE('교사이름: ' || rec.교사이름);  DBMS\_OUTPUT.PUT\_LINE('과목명: ' || rec.과목명);  DBMS\_OUTPUT.PUT\_LINE('과목시작날짜: ' || rec.과목시작날짜 || ' | 과목종료날짜: ' || rec.과목종료날짜);  DBMS\_OUTPUT.PUT\_LINE('강의실명: ' || rec.강의실명);  DBMS\_OUTPUT.PUT\_LINE('교재명: ' || rec.교재명);  DBMS\_OUTPUT.PUT\_LINE('과정시작날짜: ' || rec.과정시작날짜 || ' | 과정종료날짜: ' || rec.과정종료날짜);  DBMS\_OUTPUT.PUT\_LINE('수강인원: ' || rec.수강인원);  DBMS\_OUTPUT.PUT\_LINE('강의진행여부: ' || rec.강의진행여부);  DBMS\_OUTPUT.PUT\_LINE('--------------------------------------');  END LOOP;    EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('해당 강사의 강의 정보가 없습니다.');  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END teacherInfo;  /  BEGIN  teacherInfo('구하늘');  END;  /  -- 프로세스번호, 개설과정과목번호 insert하기  select \* from teacherprcVw  where 선생님번호 = 1 and 과정시작날짜 <= sysdate and 과정종료날짜 > sysdate; -- 선생님 선택, 현재진행중인프로세스만 나오게  CREATE OR REPLACE PROCEDURE getTeacherProcessInfo (  pTeacherSeq IN NUMBER -- 조회할 선생님번호  )  IS  BEGIN  -- teacherprcVw 뷰에서 특정 교사의 강의 정보 조회  FOR rec IN (  SELECT \*  FROM teacherprcVw  WHERE 선생님번호 = pTeacherSeq  ORDER BY 선생님번호, 과정번호, 과목번호  ) LOOP  DBMS\_OUTPUT.PUT\_LINE('선생님번호: ' || rec.선생님번호);  DBMS\_OUTPUT.PUT\_LINE('선생님이름: ' || rec.선생님이름);  DBMS\_OUTPUT.PUT\_LINE('과정번호: ' || rec.과정번호);  DBMS\_OUTPUT.PUT\_LINE('개설과정번호: ' || rec.개설과정번호);  DBMS\_OUTPUT.PUT\_LINE('과목번호: ' || rec.과목번호);  DBMS\_OUTPUT.PUT\_LINE('과목명: ' || rec.과목명);  DBMS\_OUTPUT.PUT\_LINE('출석점수: ' || rec.출석점수 ||  ', 필기점수: ' || rec.필기점수 ||  ', 실기점수: ' || rec.실기점수);  DBMS\_OUTPUT.PUT\_LINE('과목시작날짜: ' || rec.과목시작날짜 ||  ' ~ 과목종료날짜: ' || rec.과목종료날짜);  DBMS\_OUTPUT.PUT\_LINE('과정시작날짜: ' || rec.과정시작날짜 ||  ' ~ 과정종료날짜: ' || rec.과정종료날짜);  DBMS\_OUTPUT.PUT\_LINE('--------------------------------------');  END LOOP;  EXCEPTION  -- 데이터가 없는 경우  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('해당 선생님번호 (' || pTeacherSeq || ') 에 대한 강의 정보가 없습니다.');  -- 기타 오류 처리  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END getTeacherProcessInfo;  /  BEGIN  getTeacherProcessInfo(1); -- 특정 선생님의 강의 정보 조회  END;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | C-03 | 작성일 | 2025-02-06 |
| 요구사항명 | 배점 입출력 | 작성자 | 서희원 |
| DML | | | |
| -- 배점 insert  CREATE OR REPLACE PROCEDURE insertScoreAllot (  pPrcSubjectSeq IN NUMBER, -- 개설과정과목번호  pTeacherSeq IN NUMBER, -- 교사번호  pAttendAllot IN NUMBER, -- 출석 배점  pWritingAllot IN NUMBER, -- 필기 배점  pRealAllot IN NUMBER -- 실기 배점  )  IS  BEGIN  -- 배점 합이 100이 맞는지 확인  IF pAttendAllot + pWritingAllot + pRealAllot <> 100 THEN  DBMS\_OUTPUT.PUT\_LINE('오류: 출석, 필기, 실기 배점의 합이 100이 되어야 합니다.');  RETURN;  END IF;  -- scoreAllot 테이블에 데이터 삽입  INSERT INTO scoreAllot(scoreAllotSeq, prcSubjectSeq, teacherSeq, attendAllot, writingAllot, realAllot)  VALUES (scoreAllot\_Seq.NEXTVAL, pPrcSubjectSeq, pTeacherSeq, pAttendAllot, pWritingAllot, pRealAllot);  -- 변경 사항 저장  COMMIT;  -- 삽입 완료 메시지 출력  DBMS\_OUTPUT.PUT\_LINE('새로운 scoreAllot 데이터가 삽입되었습니다.');  DBMS\_OUTPUT.PUT\_LINE('개설과정과목번호: ' || pPrcSubjectSeq || ', 교사번호: ' || pTeacherSeq);  DBMS\_OUTPUT.PUT\_LINE('출석 배점: ' || pAttendAllot || ', 필기 배점: ' || pWritingAllot || ', 실기 배점: ' || pRealAllot);  EXCEPTION  -- 데이터 삽입 시 오류 처리  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  ROLLBACK;  END insertScoreAllot;  /  BEGIN  insertScoreAllot(1, 1, 20, 0, 80); -- 개설과정과목번호: 1, 교사번호: 1, 출석: 20, 필기: 0, 실기: 80  END;  /  -- 배점 update 실기  CREATE OR REPLACE PROCEDURE updateRealAllot (  pScoreAllotSeq IN NUMBER, -- 수정할 scoreAllotSeq 값  pRealAllot IN NUMBER -- 새로운 실기 배점 값  )  IS  v\_count NUMBER;  BEGIN  -- scoreAllotSeq 값이 존재하는지 확인  SELECT COUNT(\*)  INTO v\_count  FROM scoreAllot  WHERE scoreAllotSeq = pScoreAllotSeq;  IF v\_count = 0 THEN  DBMS\_OUTPUT.PUT\_LINE('오류: scoreAllotSeq = ' || pScoreAllotSeq || ' 해당 데이터가 존재하지 않습니다.');  RETURN;  END IF;  UPDATE scoreAllot  SET realAllot = pRealAllot  WHERE scoreAllotSeq = pScoreAllotSeq;  COMMIT;  DBMS\_OUTPUT.PUT\_LINE('scoreAllotSeq = ' || pScoreAllotSeq || ' 의 실기 배점이 ' || pRealAllot || ' 으로 업데이트되었습니다.');  EXCEPTION  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  ROLLBACK;  END updateRealAllot;  /  BEGIN  updateRealAllot(1, 80); -- scoreAllotSeq = 1의 실기 배점을 80으로 업데이트  END;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | C-04 | 작성일 | 2025-02-06 |
| 요구사항명 | 성적 입출력 | 작성자 |  |
| DML | | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | C-05 | 작성일 | 2025-02-06 |
| 요구사항명 | 출결 관리 및 출결 조회 | 작성자 | 박세원 |
| DML | | | |
| --날짜의 뷰 먼저 생성  create or replace view vwTotalDate  as  select  to\_date('2024-07-03', 'yyyy-mm-dd') + level - 1 as regdate  from dual  connect by level <= (to\_date('2025-08-24', 'yyyy-mm-dd')  - to\_date('2024-07-03', 'yyyy-mm-dd') + 1);  --1. 2번 교사가 강의한 1번 과정의 모든 교육생의 출결  CREATE OR REPLACE PROCEDURE pAttendanceTeacherTotal(  pteacherNum IN teacher.teacherseq%TYPE,  pprocessNum IN process.processseq%TYPE  )  IS  vteacherName teacher.teachername%TYPE;  vstudentName student.studentname%TYPE;  vdate DATE;  vresult VARCHAR2(50);  vstartdate date;  venddate date;    -- 교육생 목록 조회를 위한 커서  CURSOR student\_cursor IS  SELECT s.studentseq, s.studentname  FROM student s  JOIN studentcls sc ON s.studentseq = sc.studentseq  WHERE sc.processseq = pprocessNum;    -- 특정 교육생의 출결 조회를 위한 커서  CURSOR attendance\_cursor(pstudentNum student.studentseq%TYPE) IS  SELECT  v.regdate AS "날짜",  CASE  WHEN TO\_CHAR(v.regdate, 'd') = '1' THEN '일요일'  WHEN TO\_CHAR(v.regdate, 'd') = '7' THEN '토요일'  WHEN h.holidayseq IS NOT NULL THEN h.holidayname  WHEN h.holidayseq IS NULL AND t.attendanceseq IS NULL THEN '결석'  ELSE t.attendancest  END AS "상태"  FROM vwTotalDate v  LEFT JOIN attendance t  ON TO\_CHAR(v.regdate, 'yyyy-mm-dd') = TO\_CHAR(t.attendancedate, 'yyyy-mm-dd')  AND t.studentseq = pstudentNum  LEFT JOIN holiday h  ON TO\_CHAR(v.regdate, 'yyyy-mm-dd') = TO\_CHAR(h.holidaydate, 'yyyy-mm-dd')  WHERE v.regdate BETWEEN vstartdate AND venddate  ORDER BY v.regdate ASC;  BEGIN  -- 교사 이름  SELECT teachername INTO vteacherName  FROM teacher  WHERE teacherseq = pteacherNum;    -- 과정의 시작일과 종료일 가져오기  SELECT processsdate, processedate INTO vstartdate, venddate  FROM process  WHERE processseq = pprocessNum;  DBMS\_OUTPUT.PUT\_LINE(vteacherName || ' 강사의 ' || pprocessNum || '번 과정 출석부');  DBMS\_OUTPUT.PUT\_LINE('----------------------------------------------------');  DBMS\_OUTPUT.PUT\_LINE('조회할 기간 : ' || vstartdate || '~ ' || venddate);  DBMS\_OUTPUT.PUT\_LINE('----------------------------------------------------');  -- 학생 목록  FOR student\_rec IN student\_cursor LOOP  vstudentName := student\_rec.studentname;  DBMS\_OUTPUT.PUT\_LINE(vstudentName);  -- 특정 학생의 출결  OPEN attendance\_cursor(student\_rec.studentseq);    LOOP  FETCH attendance\_cursor INTO vdate, vresult;  EXIT WHEN attendance\_cursor%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE(' [' || TO\_CHAR(vdate, 'YYYY-MM-DD') || '] ' || vresult);  END LOOP;  CLOSE attendance\_cursor;  END LOOP;  END;  /  -- 매개변수 : 교사번호,과정  -- 호출문  BEGIN  pAttendanceTeacherTotal(2,1);  END;  /  ----------------------------------------------------------------------------------  --2. 2번 교사가 강의한 1번 과정의 월 입력시 모든 교육생의 출결  CREATE OR REPLACE PROCEDURE pAttendanceTeacherMonth(  pteacherNum IN teacher.teacherseq%TYPE,  pprocessNum IN process.processseq%TYPE,  pyear IN NUMBER,  pmonth IN NUMBER  )  IS  vteacherName teacher.teachername%TYPE;  vstudentName student.studentname%TYPE;  vdate DATE;  vresult VARCHAR2(50);    -- 교육생 목록 조회를 위한 커서  CURSOR student\_cursor IS  SELECT s.studentseq, s.studentname  FROM student s  JOIN studentcls sc ON s.studentseq = sc.studentseq  WHERE sc.processseq = pprocessNum;    -- 특정 교육생의 출결 조회를 위한 커서  CURSOR attendance\_cursor(pstudentNum student.studentseq%TYPE) IS  SELECT  v.regdate AS "날짜",  CASE  WHEN TO\_CHAR(v.regdate, 'd') = '1' THEN '일요일'  WHEN TO\_CHAR(v.regdate, 'd') = '7' THEN '토요일'  WHEN h.holidayseq IS NOT NULL THEN h.holidayname  WHEN h.holidayseq IS NULL AND t.attendanceseq IS NULL THEN '결석'  ELSE t.attendancest  END AS "상태"  FROM vwTotalDate v  LEFT JOIN attendance t  ON TO\_CHAR(v.regdate, 'yyyy-mm-dd') = TO\_CHAR(t.attendancedate, 'yyyy-mm-dd')  AND t.studentseq = pstudentNum  LEFT JOIN holiday h  ON TO\_CHAR(v.regdate, 'yyyy-mm-dd') = TO\_CHAR(h.holidaydate, 'yyyy-mm-dd')  WHERE EXTRACT(YEAR FROM v.regdate) = pyear  AND EXTRACT(MONTH FROM v.regdate) = pmonth  ORDER BY v.regdate ASC;  BEGIN  -- 교사 이름  SELECT teachername INTO vteacherName  FROM teacher  WHERE teacherseq = pteacherNum;  DBMS\_OUTPUT.PUT\_LINE(vteacherName || ' 강사의 ' || pprocessNum || '번 과정 출석부');  DBMS\_OUTPUT.PUT\_LINE('----------------------------------------------------');  DBMS\_OUTPUT.PUT\_LINE('조회할 월 : ' || pyear || '년 ' || pmonth || '월 ');  DBMS\_OUTPUT.PUT\_LINE('----------------------------------------------------');  -- 학생 목록  FOR student\_rec IN student\_cursor LOOP  vstudentName := student\_rec.studentname;  DBMS\_OUTPUT.PUT\_LINE(vstudentName);  -- 특정 학생의 출결  OPEN attendance\_cursor(student\_rec.studentseq);    LOOP  FETCH attendance\_cursor INTO vdate, vresult;  EXIT WHEN attendance\_cursor%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE(' [' || TO\_CHAR(vdate, 'YYYY-MM-DD') || '] ' || vresult);  END LOOP;  CLOSE attendance\_cursor;  END LOOP;  END;  /  -- 매개변수 : 교사번호,과정, 년  -- 호출문  BEGIN  pAttendanceTeacherMonth(2,1,2024,7);  END;  /  -------------------------------------------------------------------------------------  --3. 2번 교사가 강의한 1번 과정의 날짜 입력시 모든 교육생의 출결  CREATE OR REPLACE PROCEDURE pAttendanceTeacherDate(  pteacherNum IN teacher.teacherseq%TYPE,  pprocessNum IN process.processseq%TYPE,  pyear IN NUMBER,  pmonth IN NUMBER,  pdate IN NUMBER  )  IS  vteacherName teacher.teachername%TYPE;  vstudentName student.studentname%TYPE;  vdate DATE;  vresult VARCHAR2(50);    -- 교육생 목록 조회를 위한 커서  CURSOR student\_cursor IS  SELECT s.studentseq, s.studentname  FROM student s  JOIN studentcls sc ON s.studentseq = sc.studentseq  WHERE sc.processseq = pprocessNum;    -- 특정 교육생의 출결 조회를 위한 커서  CURSOR attendance\_cursor(pstudentNum student.studentseq%TYPE) IS  SELECT  v.regdate AS "날짜",  CASE  WHEN TO\_CHAR(v.regdate, 'd') = '1' THEN '일요일'  WHEN TO\_CHAR(v.regdate, 'd') = '7' THEN '토요일'  WHEN h.holidayseq IS NOT NULL THEN h.holidayname  WHEN h.holidayseq IS NULL AND t.attendanceseq IS NULL THEN '결석'  ELSE t.attendancest  END AS "상태"  FROM vwTotalDate v  LEFT JOIN attendance t  ON TO\_CHAR(v.regdate, 'yyyy-mm-dd') = TO\_CHAR(t.attendancedate, 'yyyy-mm-dd')  AND t.studentseq = pstudentNum  LEFT JOIN holiday h  ON TO\_CHAR(v.regdate, 'yyyy-mm-dd') = TO\_CHAR(h.holidaydate, 'yyyy-mm-dd')  WHERE EXTRACT(YEAR FROM v.regdate) = pyear  AND EXTRACT(MONTH FROM v.regdate) = pmonth  AND EXTRACT(DAY FROM v.regdate) = pdate  ORDER BY v.regdate ASC;  BEGIN  -- 교사 이름  SELECT teachername INTO vteacherName  FROM teacher  WHERE teacherseq = pteacherNum;  DBMS\_OUTPUT.PUT\_LINE(vteacherName || ' 강사의 ' || pprocessNum || '번 과정 출석부');  DBMS\_OUTPUT.PUT\_LINE('----------------------------------------------------');  DBMS\_OUTPUT.PUT\_LINE('조회할 날짜 : ' || pyear || '년 ' || pmonth || '월 ' || pdate || '일');  DBMS\_OUTPUT.PUT\_LINE('----------------------------------------------------');  -- 학생 목록  FOR student\_rec IN student\_cursor LOOP  vstudentName := student\_rec.studentname;  -- 특정 학생의 출결  OPEN attendance\_cursor(student\_rec.studentseq);    LOOP  FETCH attendance\_cursor INTO vdate, vresult;  EXIT WHEN attendance\_cursor%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE(vstudentName||' : '|| vresult);  END LOOP;  CLOSE attendance\_cursor;  END LOOP;  END;  /  -- 매개변수 : 교사번호,과정, 년, 월  -- 호출문  BEGIN  pAttendanceTeacherDate(2,1,2024,7,24);  END;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | D-01 | 작성일 | 2025-02-06 |
| 요구사항명 | 교육생 로그인 | 작성자 | 박세원 |
| DML | | | |
| /\* 조회, 호출문의 매개변수 : 학생 번호 , 학생 비밀번호 \*/  CREATE OR REPLACE PROCEDURE pStudentLogin(  pstudentNum IN STUDENT.studentseq%TYPE,  pstudentPW IN STUDENT.studentpw%TYPE  )  IS  vstudentName student.studentname%TYPE;  vstudentTel student.studenttel%TYPE;  vstudentDate student.studentdate%TYPE;  vcourseName course.coursename%TYPE;  vprocessStart process.processsdate%TYPE;  vprocessEnd process.processedate%TYPE;  vclassroom clsroom.clsroomname%TYPE;  vcount NUMBER;  -- 학생 정보  CURSOR student\_cursor  IS  SELECT  s.studentname,  s.studenttel,  s.studentdate,  c.coursename,  p.processsdate,  p.processedate,  cr.clsroomname  FROM student s  INNER JOIN studentcls sc ON s.studentseq = sc.studentseq  INNER JOIN process p ON sc.processseq = p.processseq  INNER JOIN course c ON c.courseseq = p.courseseq  INNER JOIN clsroom cr ON cr.clsroomseq = p.clsroomseq  WHERE s.studentseq = pstudentNum;  BEGIN  -- 1. 학생 존재 확인, 및 비밀번호 확인  SELECT COUNT(\*) INTO vcount  FROM student  WHERE studentseq = pstudentNum AND studentpw = pstudentPW;    IF vcount = 0 THEN  DBMS\_OUTPUT.PUT\_LINE('로그인 실패: 학번 또는 비밀번호가 틀렸습니다.');  RETURN;  END IF;  -- 2. 학생 정보 출력  OPEN student\_cursor;    FETCH student\_cursor INTO  vstudentName, vstudentTel, vstudentDate,  vcourseName, vprocessStart, vprocessEnd, vclassroom;    IF student\_cursor%NOTFOUND THEN  DBMS\_OUTPUT.PUT\_LINE('학생 정보가 존재하지 않습니다.');  ELSE  DBMS\_OUTPUT.PUT\_LINE(' 로그인 성공!');  DBMS\_OUTPUT.PUT\_LINE('--------------------------------------');  DBMS\_OUTPUT.PUT\_LINE('이름: ' || vstudentName);  DBMS\_OUTPUT.PUT\_LINE('전화번호: ' || vstudentTel);  DBMS\_OUTPUT.PUT\_LINE('등록일: ' || TO\_CHAR(vstudentDate, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('과정명: ' || vcourseName);  DBMS\_OUTPUT.PUT\_LINE('과정 시작일: ' || TO\_CHAR(vprocessStart, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('과정 종료일: ' || TO\_CHAR(vprocessEnd, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('강의실: ' || vclassroom);  DBMS\_OUTPUT.PUT\_LINE('--------------------------------------');  END IF;    CLOSE student\_cursor;  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('로그인 실패: 학번 또는 비밀번호가 틀렸습니다.');  WHEN OTHERS THEN  DBMS\_OUTPUT.PUT\_LINE('오류 발생!');  END pStudentLogin;  /  -- 호출문의 매개변수 : 학생 번호  BEGIN  pStudentLogin(1,2345678);  END;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | D-02 | 작성일 | 2025-02-06 |
| 요구사항명 | 교육생 개인 성적 정보 확인 기능 | 작성자 | 박세원 |
| DML | | | |
| /\* 조회, 호출문의 매개변수 : 학생 번호 \*/  CREATE OR REPLACE PROCEDURE pStudentInfo(  p\_studentseq IN NUMBER  )  as  BEGIN  FOR rec IN (  SELECT  pc.subjectseq AS 과목번호,  sj.subjectname AS 과목명,  pc.prcsubjectsdate || ' ~ ' || pc.prcsubjectedate AS "과목 진행 기간",  b.bookname AS 교재명,  t.teachername AS 교사명,  '(' || sa.attendallot || ', ' || sa.writingallot || ' ,' || sa.realallot || ')' AS "과목별 배점정보(출석, 필기, 실기)",  '('||score.attendancescore||', '||score.writingscore ||' ,'|| score.realscore ||')' as "과목별 성적정보(출석, 필기, 실기)",  score.totalscore AS "과목별 총점수",  test.testtype AS 시험유형,  test.testdate AS 시험날짜,  test.testcontext AS 시험문제  FROM prcsubject pc  INNER JOIN subject sj ON sj.subjectseq = pc.subjectseq  INNER JOIN sbjectbook sb ON sb.subjectseq = pc.subjectseq  INNER JOIN book b ON b.bookseq = sb.bookseq  INNER JOIN process p ON p.processseq = pc.processseq  INNER JOIN teacher t ON t.teacherseq = p.teacherseq  INNER JOIN scoreallot sa ON sa.prcsubjectseq = pc.prcsubjectseq  INNER JOIN test ON test.teacherseq = t.teacherseq AND pc.subjectseq = test.subjectseq  INNER JOIN studentcls scl ON scl.processseq = p.processseq  INNER JOIN student st ON st.studentseq = scl.studentseq  INNER JOIN score ON score.subjectseq = pc.subjectseq AND score.studentseq = st.studentseq  WHERE st.studentseq = p\_studentseq  AND pc.processseq = (SELECT processseq FROM studentcls WHERE studentseq = p\_studentseq)  AND test.teacherseq = (SELECT teacherseq FROM process WHERE processseq = (SELECT processseq FROM studentcls WHERE studentseq = p\_studentseq))  AND test.testdate BETWEEN p.processsdate AND p.processedate  ) LOOP  DBMS\_OUTPUT.PUT\_LINE('과목번호: ' || rec.과목번호);  DBMS\_OUTPUT.PUT\_LINE('과목명: ' || rec.과목명);  DBMS\_OUTPUT.PUT\_LINE('과목 진행 기간: ' || rec."과목 진행 기간");  DBMS\_OUTPUT.PUT\_LINE('교재명: ' || rec.교재명);  DBMS\_OUTPUT.PUT\_LINE('교사명: ' || rec.교사명);  DBMS\_OUTPUT.PUT\_LINE('과목별 배점정보(출석, 필기, 실기): ' || rec."과목별 배점정보(출석, 필기, 실기)");  DBMS\_OUTPUT.PUT\_LINE('과목별 성적정보(출석, 필기, 실기): ' || rec."과목별 성적정보(출석, 필기, 실기)");  DBMS\_OUTPUT.PUT\_LINE('과목별 총점수: ' || rec."과목별 총점수");  DBMS\_OUTPUT.PUT\_LINE('시험유형: ' || rec.시험유형);  DBMS\_OUTPUT.PUT\_LINE('시험날짜: ' || rec.시험날짜);  DBMS\_OUTPUT.PUT\_LINE('시험문제: ' || rec.시험문제);  DBMS\_OUTPUT.PUT\_LINE('---------------------------------------------------');  END LOOP;  END pStudentInfo;  /  -- 호출문의 매개변수 : 학생 번호  BEGIN  pStudentInfo(27);  END;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | D-03 | 작성일 | 2025-02-06 |
| 요구사항명 | 교육생 출결관리 기능 | 작성자 | 박세원 |
| DML | | | |
| -- 출결 조회하기, 호출문의 매개변수 : 학생 번호,년,월,일  CREATE OR REPLACE PROCEDURE pStudentAttendance(  pstudentNum IN attendance.studentseq%TYPE  )  IS  CURSOR cur\_attendance IS  SELECT  attendancedate AS 날짜,  TO\_CHAR(attendancestime, 'HH24:MI') AS 등원시간,  TO\_CHAR(attendanceetime, 'HH24:MI') AS 하원시간  FROM attendance  WHERE studentseq = pstudentNum  ORDER BY attendancedate;  v\_attendancedate attendance.attendancedate%TYPE;  v\_attendancestime VARCHAR2(5);  v\_attendanceetime VARCHAR2(5);  BEGIN  OPEN cur\_attendance;    LOOP  FETCH cur\_attendance INTO  v\_attendancedate, v\_attendancestime, v\_attendanceetime;    EXIT WHEN cur\_attendance%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE('날짜: ' || TO\_CHAR(v\_attendancedate, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('등원 시간: ' || v\_attendancestime);  DBMS\_OUTPUT.PUT\_LINE('하원 시간: ' || v\_attendanceetime);  DBMS\_OUTPUT.PUT\_LINE('-------------------------------------');  END LOOP;  CLOSE cur\_attendance;  END;  /  -- 호출문의 매개변수 : 학생 번호  BEGIN  pStudentAttendance(1);  END;  /  -----------------------------------------------------------------------------------  -- 특정 날짜(2024-07-24) 출결 조회하기  CREATE OR REPLACE PROCEDURE pStudentAttendanceDate(  pstudentNum IN attendance.studentseq%TYPE,  pyear IN NUMBER,  pmonth IN NUMBER,  pdate IN NUMBER  )  IS  CURSOR cur\_attendance IS  SELECT  attendancedate AS 날짜,  TO\_CHAR(attendancestime, 'HH24:MI') AS 등원시간,  TO\_CHAR(attendanceetime, 'HH24:MI') AS 하원시간  FROM attendance  WHERE studentseq = pstudentNum  AND EXTRACT(year FROM attendancedate) = pyear  AND EXTRACT(MONTH FROM attendancedate) = pmonth  AND EXTRACT(day FROM attendancedate) = pdate  ORDER BY attendancedate;  vattendancedate attendance.attendancedate%TYPE;  vattendancestime VARCHAR2(5);  vattendanceetime VARCHAR2(5);  BEGIN  OPEN cur\_attendance;    LOOP  FETCH cur\_attendance INTO  vattendancedate, vattendancestime, vattendanceetime;    EXIT WHEN cur\_attendance%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE('-------------------------------------');  DBMS\_OUTPUT.PUT\_LINE('날짜: ' || TO\_CHAR(vattendancedate, 'YYYY-MM-DD'));  DBMS\_OUTPUT.PUT\_LINE('등원 시간: ' || vattendancestime);  DBMS\_OUTPUT.PUT\_LINE('하원 시간: ' || vattendanceetime);  DBMS\_OUTPUT.PUT\_LINE('-------------------------------------');  END LOOP;  CLOSE cur\_attendance;  END;  /  -- 호출문의 매개변수 : 학생 번호,년,월,일  BEGIN  pStudentAttendanceDate(1,2024,7,24);  END;  /  --출석 입력하기, 등원 추가, 호출문의 매개변수 : 학생 번호  CREATE OR REPLACE PROCEDURE pInsertAttendance(  pstudentNum IN attendance.studentseq%TYPE  )  IS  vstudentName student.studentname%TYPE;  vcount number;  BEGIN  select  studentname into vstudentName  from student where studentseq = pstudentNum;    -- 출석을 했는가 예외처리  SELECT COUNT(\*)  INTO vcount  FROM attendance  WHERE studentseq = pstudentNum  AND attendancedate = TRUNC(SYSDATE);    -- 이미 출석을 한 경우 종료  IF vcount > 0 THEN  DBMS\_OUTPUT.PUT\_LINE(vstudentName || ' 학생은 이미 출석이 기록되어 있습니다.');  RETURN;  END IF;      INSERT INTO attendance (  attendanceseq,  studentseq,  attendancedate,  attendancestime,  attendanceetime,  attendancest  ) VALUES (  attendance\_seq.NEXTVAL,  pstudentNum,  TRUNC(SYSDATE), -- 현재 날짜 (시간 00:00:00)  SYSDATE, -- 현재 시간으로 출석 시간 기록  NULL, -- 퇴근  '출석'  );  -- 변경 사항 저장  COMMIT;    DBMS\_OUTPUT.PUT\_LINE(vstudentname||' 학생, '|| '출석 등록 완료 (' || TO\_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS') || ')');    EXCEPTION  WHEN OTHERS THEN  -- 에러 발생 시 롤백  ROLLBACK;  DBMS\_OUTPUT.PUT\_LINE('오류 발생');  END ;  /  -- 호출문의 매개변수 : 학생 번호  BEGIN  pInsertAttendance(1);  END;  /  ---------------------------------------------------------------------------------  -- 수업종료 입력하기, 하원 추가  CREATE OR REPLACE PROCEDURE pUpdateAttendance(  pstudentNum IN attendance.studentseq%TYPE  )  IS  vattendancestime DATE;  vstudentName varchar2(50);  vstatus VARCHAR2(10);  vcount number;  BEGIN  select  max(studentname) into vstudentName  from student where studentseq = pstudentNum;      -- 하원을 했는가 예외처리  SELECT COUNT(\*)  INTO vcount  FROM attendance  WHERE studentseq = pstudentNum  AND attendancedate = TRUNC(SYSDATE)  and attendanceetime is not null;    -- 이미 하원을 한 경우 종료  IF vcount > 0 THEN  DBMS\_OUTPUT.PUT\_LINE(vstudentName || ' 학생은 이미 하원이 기록되어 있습니다.');  RETURN;  END IF;      SELECT min(attendancestime)  INTO vattendancestime  FROM attendance  WHERE studentseq = pstudentNum  AND attendancedate = TRUNC(SYSDATE)  AND attendanceetime IS NULL;  -- 등원 시간이 NULL -> 지각(하원은 등록했기때문)  IF vattendancestime IS NULL THEN  vstatus := '지각';  -- 정상 출석  ELSIF TO\_CHAR(vattendancestime, 'HH24:MI') <= '09:00'  AND TO\_CHAR(SYSDATE, 'HH24:MI') >= '18:00' THEN  vstatus := '정상';    -- 지각 (등원이 09:00 이후)  ELSIF TO\_CHAR(vattendancestime, 'HH24:MI') > '09:00' THEN  vstatus := '지각';  -- 조퇴 (하원이 18:00 이전)  ELSIF TO\_CHAR(SYSDATE, 'HH24:MI') < '18:00' THEN  vstatus := '조퇴';  END IF;  --업데이트  UPDATE attendance  SET attendanceetime = SYSDATE,  attendancest = vstatus  WHERE studentseq = pstudentNum  AND attendancedate = TRUNC(SYSDATE)  AND attendanceetime IS NULL;  --저장  COMMIT;  DBMS\_OUTPUT.PUT\_LINE(vstudentname ||' 학생, '|| ' 퇴근 등록 완료 (' || TO\_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS') || ')');  DBMS\_OUTPUT.PUT\_LINE('출석 상태: ' || vstatus);  EXCEPTION  WHEN NO\_DATA\_FOUND THEN  DBMS\_OUTPUT.PUT\_LINE('출석 기록 없음: ' || vstudentName || '의 오늘 출석 정보가 없습니다.');  WHEN OTHERS THEN  ROLLBACK;  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END;  /  -- 호출문의 매개변수 : 학생 번호  BEGIN  pUpdateAttendance(1);  END;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | D-04 | 작성일 | 2025-02-06 |
| 요구사항명 | 교육생 출결 조회 기능 | 작성자 | 박세원 |
| DML | | | |
| --날짜의 뷰 먼저 생성  create or replace view vwTotalDate  as  select  to\_date('2024-07-03', 'yyyy-mm-dd') + level - 1 as regdate  from dual  connect by level <= (to\_date('2025-08-24', 'yyyy-mm-dd')  - to\_date('2024-07-03', 'yyyy-mm-dd') + 1);    ----------------------------------------------------------------------------------  --1. 1번 학생의 출결전체 조회  -- 선언문  CREATE OR REPLACE PROCEDURE pAttendanceStudentTotal(  pstudentNum in STUDENT.studentseq%type  )  is  vname student.studentname%type;  vdate date;  vresult varchar2(50);  vstartdate date;  venddate date;    cursor vcursor  is  select  v.regdate as "날짜",  case  when to\_char(v.regdate, 'd') = '1' then '일요일'  when to\_char(v.regdate, 'd') = '7' then '토요일'  when h.holidayseq is not null then h.holidayname  when h.holidayseq is null and t.attendanceseq is null then '결석'  else t.attendancest  end as "상태"  from vwTotalDate v  left outer join attendance t  on to\_char(v.regdate, 'yyyy-mm-dd') = to\_char(t.attendancedate, 'yyyy-mm-dd')  AND t.studentseq = pstudentNum  left outer join holiday h  on to\_char(v.regdate, 'yyyy-mm-dd') = to\_char(h.holidaydate, 'yyyy-mm-dd')  WHERE v.regdate BETWEEN vstartdate AND venddate  order by v.regdate asc;    begin  select studentname into vname from student where studentseq=pstudentNum;    SELECT processsdate,processedate  INTO vstartdate, venddate  FROM process  where processseq=1;  dbms\_output.put\_line(vname||' 학생의 출석부');  dbms\_output.put\_line('----------------------------------------------------');  dbms\_output.put\_line('조회할 기간 : '|| TO\_CHAR(vstartdate, 'YYYY-MM-DD') || ' ~ ' || TO\_CHAR(venddate, 'YYYY-MM-DD'));    open vcursor;    loop  fetch vcursor into vdate,vresult;  exit when vcursor%notfound;    DBMS\_OUTPUT.PUT\_LINE('['||TO\_CHAR(vdate, 'YYYY-MM-DD')||'] ' || vresult);    end loop;  close vcursor;  end;  /  -- 호출문의 매개변수 : 학생 번호  BEGIN  pAttendanceStudentTotal(1);  END;  /  --------------------------------------------------------------------------------  --2. 1번 학생의 7월달 출결 조회  -- 선언문  CREATE OR REPLACE PROCEDURE pAttendanceStudentMonth(  pstudentNum in STUDENT.studentseq%type,  pyear in number,  pmonth in number  )  is  vname student.studentname%type;  vdate date;  vresult varchar2(50);  vstartdate date;  venddate date;  cursor vcursor  is  select  v.regdate as "날짜",  case  when to\_char(v.regdate, 'd') = '1' then '일요일'  when to\_char(v.regdate, 'd') = '7' then '토요일'  when h.holidayseq is not null then h.holidayname  when h.holidayseq is null and t.attendanceseq is null then '결석'  else t.attendancest  end as "상태"  from vwTotalDate v  left outer join attendance t  on to\_char(v.regdate, 'yyyy-mm-dd') = to\_char(t.attendancedate, 'yyyy-mm-dd')  AND t.studentseq = pstudentNum  left outer join holiday h  on to\_char(v.regdate, 'yyyy-mm-dd') = to\_char(h.holidaydate, 'yyyy-mm-dd')  WHERE extract(year from v.regdate) = pyear  and extract(month from v.regdate) = pmonth  order by v.regdate asc;    begin  select studentname into vname from student where studentseq=pstudentNum;    dbms\_output.put\_line(vname||' 학생의 출석부');  dbms\_output.put\_line('----------------------------------------------------');  dbms\_output.put\_line('조회할 월 : '||pyear || '년 ' || pmonth|| '월 ');    open vcursor;    loop  fetch vcursor into vdate,vresult;  exit when vcursor%notfound;    DBMS\_OUTPUT.PUT\_LINE('['||TO\_CHAR(vdate, 'YYYY-MM-DD')||'] ' || vresult);    end loop;  close vcursor;  end;  /  -- 매개변수 : 학생 번호, 년, 월  -- 호출문  BEGIN  pAttendanceStudentMonth(1,2024,7);  END;  /  ---------------------------------------------------------------------------------  --3. 1번 학생의 날짜별 출결 조회  -- 선언문  CREATE OR REPLACE PROCEDURE pAttendanceStudentDate(  pstudentNum in STUDENT.studentseq%type,  pyear in number,  pmonth in number,  pdate in number  )  is  vname student.studentname%type;  vdate date;  vresult varchar2(50);  cursor vcursor  is  select  v.regdate as "날짜",  case  when to\_char(v.regdate, 'd') = '1' then '일요일'  when to\_char(v.regdate, 'd') = '7' then '토요일'  when h.holidayseq is not null then h.holidayname  when h.holidayseq is null and t.attendanceseq is null then '결석'  else t.attendancest  end as "상태"  from vwTotalDate v  left outer join attendance t  on to\_char(v.regdate, 'yyyy-mm-dd') = to\_char(t.attendancedate, 'yyyy-mm-dd')  AND t.studentseq = pstudentNum  left outer join holiday h  on to\_char(v.regdate, 'yyyy-mm-dd') = to\_char(h.holidaydate, 'yyyy-mm-dd')  WHERE extract(year from v.regdate) = pyear  and extract(month from v.regdate) = pmonth  and extract(day from v.regdate) = pdate  order by v.regdate asc;    begin  select studentname into vname from student where studentseq=pstudentNum;    dbms\_output.put\_line(vname||' 학생의 출석부');  dbms\_output.put\_line('----------------------------------------------------');  dbms\_output.put\_line('조회할 날짜 : '||pyear || '년 ' || pmonth|| '월 '|| pdate|| '일 ');    open vcursor;    loop  fetch vcursor into vdate,vresult;  exit when vcursor%notfound;    DBMS\_OUTPUT.PUT\_LINE('['||TO\_CHAR(vdate, 'YYYY-MM-DD')||'] ' || vresult);    end loop;  close vcursor;  end;  /  -- 매개변수 : 학생 번호, 년, 월  -- 호출문  BEGIN  pAttendanceStudentDate(1,2024,7,24);  END;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | E-01 | 작성일 | 2025-02-06 |
| 요구사항명 | 해당 학생 자격증관련 기업 목록 조회 | 작성자 | 서희원 |
| DML | | | |
| -- 특정 학생이 갖고 있는 자격증을 요구하는 기업을 출력한다.  /\* 해당 학생자격증관련 기업 목록 \*/  CREATE OR REPLACE PROCEDURE studentEnter (  pstudentName IN VARCHAR2  )  IS  CURSOR vcursor IS  SELECT  e.enterName,  e.enterBuseo,  e.enterLocation,  e.enterSalary  FROM studentCrtf sc  INNER JOIN student s ON sc.studentSeq = s.studentSeq  INNER JOIN crtf c ON c.crtfSeq = sc.crtfSeq  INNER JOIN enter e ON c.crtfSeq = e.crtfSeq  WHERE s.studentName = pstudentName;  BEGIN  FOR rec IN vcursor LOOP  dbms\_output.put\_line('기업명: ' || rec.enterName);  dbms\_output.put\_line('부서: ' || rec.enterBuseo);  dbms\_output.put\_line('위치: ' || rec.enterLocation);  dbms\_output.put\_line('연봉: ' || rec.enterSalary);  dbms\_output.put\_line('--------------------------------');  END LOOP;  END studentEnter;  /  BEGIN  studentEnter('강나라');  END;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | E-02 | 작성일 | 2025-02-06 |
| 요구사항명 | 기업 목록 추가, 조회, 수정, 삭제 | 작성자 | 서희원 |
| DML | | | |
| /\* 기업 \*/  select \* from enter;  insert into enter values (enter\_Seq.NEXTVAL,1,5,'테스트','테스트','테스트',1);  /\*추가\*/  create or replace procedure enterInsert(  pcrtfSep number,  ptechSep number,  penterName varchar2,  penterBuseo varchar2,  penterLocation varchar2,  penterSalary number  )  is  begin  insert into enter values (enter\_Seq.NEXTVAL,pcrtfSep,ptechSep,penterName,penterBuseo,penterLocation,penterSalary);  end enterInsert;  /  begin  enterInsert(1,5,'테스트','테스트','테스트',1);  enterSelect;  end;  /  /\* 조회 \*/  select \* from enter;  create or replace procedure enterSelect  is  begin  for rec in (select \* from enter) loop  dbms\_output.put\_line('기업 : ' || rec.enterName);  dbms\_output.put\_line('부서 : ' || rec.enterBuseo);  dbms\_output.put\_line('위치 : ' || rec.enterLocation);  dbms\_output.put\_line('연봉 :' ||rec.enterSalary);  dbms\_output.put\_line('--------------------------------');  end loop;  end enterSelect;  /  begin  enterSelect;  end;  /  rollback;  /\*수정\*/  create or replace procedure enterUpdate(  ptechSep number,  penterName varchar2,  penterBuseo varchar2,  penterLocation varchar2,  penterSalary number  )  is  begin  update enter set enterName = penterName, enterBuseo = penterBuseo, enterLocation = penterLocation, enterSalary = penterSalary where enterSeq= ptechSep;  end enterUpdate;  /  begin  -- 수정할 기업 번호, 기업명, 부서, 위치, 연봉 입력  enterUpdate(100,'테스트','테스트','테스트',1);  end;  /  /\*삭제\*/  create or replace procedure enterDelete(  ptechSep number  )  is  begin  delete from enter where enterSeq = ptechSep;  end enterDelete;  /  begin  -- 삭제할 기업 번호  enterDelete(1000);  end;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | E-03 | 작성일 | 2025-02-06 |
| 요구사항명 | 공휴일 목록 추가, 조회, 수정, 삭제 | 작성자 | 박세원 |
| DML | | | |
| * 학원휴일을 holiday 테이블에 추가할 수 있다다, 이미 휴일이 있는 날짜는 예외처리를 한다. * Holiday의 휴일명을 변경할 수 있다, 없는 휴일이라면 예외처리를 한다.   /\*추가\*/  CREATE OR REPLACE PROCEDURE pInsertHoliday (  pyear in number,  pmonth in number,  pday in number  )  is  vcount NUMBER;  vholiday\_date date;  vholiday\_name varchar(50);  BEGIN  vholiday\_date := TO\_DATE(pyear || '-' || pmonth || '-' || pday, 'YYYY-MM-DD');  SELECT COUNT(\*) INTO vcount  FROM holiday  WHERE holidaydate = vholiday\_date;    select holidayname into vholiday\_name  from holiday  where holidaydate = vholiday\_date;  -- 날짜가 존재하면 예외 처리  IF vcount > 0 THEN  DBMS\_OUTPUT.PUT\_LINE(TO\_CHAR(vholiday\_date, 'YYYY-MM-DD') ||' : 이날은 '|| vholiday\_name||'입니다! 날짜를 올바르게 입력해주세요.');  ELSE  INSERT INTO holiday VALUES (holiday\_seq.NEXTVAL, TO\_DATE('2024-07-15', 'YYYY-MM-DD'), '학원휴일');  DBMS\_OUTPUT.PUT\_LINE(TO\_CHAR(vholiday\_date, 'YYYY-MM-DD') ||' : 휴일이 정상적으로 추가되었습니다.');  END IF;  END pInsertHoliday;  /  -- 호출하기 매개변수 : 년,월,일  begin  pInsertHoliday(2024,8,15);  end;  /  /\*수정\*/  CREATE OR REPLACE PROCEDURE pUpdateHoliday (  pyear in number,  pmonth in number,  pday in number,  pholiday\_name in varchar2  )  is  vcount NUMBER;  vholiday\_date date;  BEGIN  vholiday\_date := TO\_DATE(pyear || '-' || pmonth || '-' || pday, 'YYYY-MM-DD');  SELECT COUNT(\*) INTO vcount  FROM holiday  WHERE holidaydate = vholiday\_date;  -- 날짜가 존재하면 수정  IF vcount > 0 THEN  UPDATE holiday  SET holidayname = pholiday\_name  WHERE holidaydate = vholiday\_date;    DBMS\_OUTPUT.PUT\_LINE(TO\_CHAR(vholiday\_date, 'YYYY-MM-DD') || ' 휴일명이 "' || pholiday\_name || '"으로 변경되었습니다.');    ELSE  DBMS\_OUTPUT.PUT\_LINE(TO\_CHAR(vholiday\_date, 'YYYY-MM-DD') || ' 해당 날짜의 휴일 정보가 없습니다!');  END IF;  END pUpdateHoliday;  /  -- 호출하기 매개변수 : 년,월,일,공휴일명  begin  pUpdateHoliday(2024,7,15,'학원 쉬는날');  end;  /  /\*삭제\*/  CREATE OR REPLACE PROCEDURE pDeleteHoliday(  pyear in number,  pmonth in number,  pday in number  ) IS  vholiday\_date date;  vcount NUMBER;  BEGIN  vholiday\_date := TO\_DATE(pyear || '-' || pmonth || '-' || pday, 'YYYY-MM-DD');    SELECT COUNT(\*) INTO vcount  FROM holiday  WHERE holidaydate = vholiday\_date;      -- 날짜가 존재하면 삭제  IF vcount > 0 THEN  delete from holiday  WHERE holidaydate = vholiday\_date;    DBMS\_OUTPUT.PUT\_LINE(TO\_CHAR(vholiday\_date, 'YYYY-MM-DD') || '"의 휴일이 삭제되었습니다.');    ELSE  DBMS\_OUTPUT.PUT\_LINE(TO\_CHAR(vholiday\_date, 'YYYY-MM-DD') || ' 해당 날짜의 휴일 정보가 없습니다!');  END IF;  END pDeleteHoliday;  /  -- 호출하기 매개변수 : 년,월,일  begin  pDeleteHoliday(2024,7,14);  end;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 요구사항번호 | E-04 | 작성일 | 2025-02-06 |
| 요구사항명 | 학생이 보유한 자격증 | 작성자 | 박세원 |
| DML | | | |
| /\* 학생이 보유한 자격증 \*/  --pl/sql  CREATE OR REPLACE PROCEDURE pstudent\_certifications IS  -- 변수 선언  vStudentName student.studentName%TYPE;  vCrtfName crtf.crtfName%TYPE;  -- 커서 선언  CURSOR student\_cursor IS  SELECT  s.studentName AS studentName,  c.crtfName AS crtfName  FROM studentCrtf sc  INNER JOIN crtf c ON sc.crtfSeq = c.crtfSeq  INNER JOIN student s ON s.studentSeq = sc.studentSeq;  BEGIN  dbms\_output.put\_line('====================================================================');  dbms\_output.put\_line(' 학생 전체 목록과 보유한 자격증');  dbms\_output.put\_line('====================================================================');    OPEN student\_cursor;  -- 커서에서 데이터 하나씩 가져오기  LOOP  FETCH student\_cursor INTO vStudentName, vCrtfName;  -- 커서 끝에 도달하면 종료  EXIT WHEN student\_cursor%NOTFOUND;  -- 학생 이름과 자격증 출력  DBMS\_OUTPUT.PUT\_LINE('학생 이름: ' || vStudentName || ', 보유한 자격증: ' || vCrtfName);  END LOOP;  -- 커서 닫기  CLOSE student\_cursor;  EXCEPTION  WHEN OTHERS THEN  -- 오류 발생 시 예외 처리  DBMS\_OUTPUT.PUT\_LINE('오류 발생: ' || SQLERRM);  END pstudent\_certifications;  /  -- 호출하기, 매개변수 없음  begin  pstudent\_certifications();  end;  / | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 인덱스번호 | Index03 | 작성일 | 2023-04-09 |
| 인덱스명 | idxPracQ | 작성자 | 최영진 |
| DML | | | |
| --실기시험번호별 문제를 조회하는 인덱스  create index idxPracQ on tblPrac(PracTNum); | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 인덱스번호 | Index04 | 작성일 | 2023-04-09 |
| 인덱스명 | idxTask | 작성자 | 최영진 |
| DML | | | |
| --교육생별 과제를 조회하는 인덱스  create index idxTask on tblTask(stNum); | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 인덱스번호 | Index05 | 작성일 | 2023-04-09 |
| 인덱스명 | idxStudent | 작성자 | 김수민 |
| DML | | | |
| --교육생별 과제를 조회하는 인덱스  create index idxStudent on tblStudent(stNum); | | | |