다음은 학사관리 테이블에 대한 연습문제이다.

STUDENT(STU_NO, STU_NAME, STU_DEPT, STU_GRADE, STU_CLASS,
STU_GENDER, STU_HEIGHT, STU_WEIGHT)
ENROL(SUB_NO, STU_NO, ENR_GRADE)
SUBJECT(SUB_NO, SUB_NAME, SUB_PROF, SUB_GRADE, SUB_DEPT)

(문제 1) 기존 과목번호를 새로운 과목번호로 교체하는 프로시저를 생성하시오.
프로시저 이름: **p_sub_no_update**

<실행>

```
SQL> execute p_sub_no_update('501','201');
```

교체 할 과목번호가 없습니다.

	Line	Position	Message
			교체 할 과목번호가 없습니다.

```
SQL> execute p_sub_no_update('101','102');
```

교체 될 과목번호가 이미 있습니다.

	Line	Position	Message
			교체 될 과목번호가 이미 있습니다.

```
SQL> execute p_sub_no_update('101','501');
```

완료되었습니다.

```
CREATE OR REPLACE PROCEDURE p_sub_no_update(p_old_sub_no IN
VARCHAR2, p_new_sub_no IN VARCHAR2) AS
  v_old_sub_count NUMBER;
  v_new_sub_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_old_sub_count
  FROM SUBJECT
  WHERE SUB_NO = p_old_sub_no;

  SELECT COUNT(*) INTO v_new_sub_count
  FROM SUBJECT
  WHERE SUB_NO = p_new_sub_no;

  IF v_old_sub_count = 0 THEN
    DBMS_OUTPUT.PUT_LINE('교체 할 과목번호가 없습니다.');
  ELSIF v_new_sub_count > 0 THEN
```

```
    DBMS_OUTPUT.PUT_LINE('교체 될 과목번호가 이미 있습니다.');
  ELSE
    UPDATE SUBJECT
    SET SUB_NO = p_new_sub_no
    WHERE SUB_NO = p_old_sub_no;

    DBMS_OUTPUT.PUT_LINE('완료되었습니다.');
  END IF;
END;
```

(문제 2) 수강테이블을 추가하는 프로시저를 생성하시오.(무결성유지)
프로시저 이름 :**p_insert_enrol**

<실행>

SQL> execute p_insert_enrol('101','20141001',40);
무결성에 위배됩니다.

	Line	Position	Message
1			무결성에 위배됩니다.

SQL> execute p_insert_enrol('106','20213088',40);
학번 20153088으로 106번 과목이 40점으로 추가되었습니다.

	Line	Position	Message
1			학번 20213088으로 106번 과목이 40점으로 추가되었습니다.

```
CREATE OR REPLACE PROCEDURE p_insert_enrol(p_sub_no IN VARCHAR2,
p_stu_no IN VARCHAR2, p_enr_grade IN NUMBER) AS
  v_stu_count NUMBER;
  v_sub_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_stu_count
  FROM STUDENT
  WHERE STU_NO = p_stu_no;

  SELECT COUNT(*) INTO v_sub_count
  FROM SUBJECT
  WHERE SUB_NO = p_sub_no;

  IF v_stu_count = 0 OR v_sub_count = 0 THEN
    DBMS_OUTPUT.PUT_LINE('무결성에 위배됩니다.');
  ELSE
    INSERT INTO ENROL(SUB_NO, STU_NO, ENR_GRADE)
    VALUES (p_sub_no, p_stu_no, p_enr_grade);
```

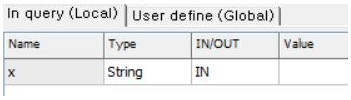
```
    DBMS_OUTPUT.PUT_LINE('학번 ' || p_stu_no || '으로 ' || p_sub_no || '번 과목
이 ' || p_enr_grade || '점으로 추가되었습니다.');
  END IF;
END;
```

(문제 3) 과목번호를 입력하면 해당 교사의 이름을 반환하는 함수를 작성하시오.
함수 이름 : **f_prof_return**

<실행>

SQL> variable x varchar2(10);

SQL> execute :x := f_prof_return(999);
과목번호가 없습니다.



SQL> execute :x := f_prof_return(107);

x
구봉규

```
CREATE OR REPLACE FUNCTION f_prof_return(p_sub_no IN VARCHAR2) RETURN
VARCHAR2 AS v_prof_name VARCHAR2(50);
BEGIN
  SELECT SUB_PROF INTO v_prof_name
  FROM SUBJECT
  WHERE SUB_NO = p_sub_no;

  IF v_prof_name IS NULL THEN
    DBMS_OUTPUT.PUT_LINE('과목번호가 없습니다.');
    RETURN NULL;
  ELSE
    RETURN v_prof_name;
  END IF;
END;
```

다음은 판매관리 테이블에 대한 실전문제이다.

PRODUCT(P_CODE, P_NAME, P_COST, P_GROUP)
STOCK(P_CODE, S_QTY, S_LASTDATE)

CUSTOMER(C_CODE, C_NAME, C_CEO, C_ADDR, C_PHONE)
TRADE(T_SEQ, P_CODE, C_CODE, T_DATE, T_QTY, T_COST, T_TAX)

(문제 4) 상품을 삽입하는 프로시저를 생성하여라.
프로시저 이름 : p_product_insert

<실행>

SQL> execute p_product_insert('101','7.1채널 스피커',180000,'스피커');
기존 p_code가 있습니다.

	Line	Position	Message
1			기존 P_CODE가 있습니다.

SQL> execute p_product_insert('403','7.1채널 스피커',180000,'스피커');
완료되었습니다.

	Line	Position	Message
1			상품이 추가되었습니다.

```
CREATE OR REPLACE PROCEDURE p_product_insert(
  p_p_code IN VARCHAR2,
  p_p_name IN VARCHAR2,
  p_p_cost IN NUMBER,
  p_p_group IN VARCHAR2
) AS
  v_existing_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_existing_count
  FROM PRODUCT
  WHERE P_CODE = p_p_code;

  IF v_existing_count > 0 THEN
    DBMS_OUTPUT.PUT_LINE('기존 P_CODE가 있습니다.');
  ELSE
    INSERT INTO PRODUCT(P_CODE, P_NAME, P_COST, P_GROUP)
    VALUES (p_p_code, p_p_name, p_p_cost, p_p_group);

    DBMS_OUTPUT.PUT_LINE('상품이 추가되었습니다.');
  END IF;
END;
```

(문제 5) 거래테이블에 거래내역을 추가하여라.(재고확인)
프로시저 이름 : **p_trade_insert**

<실행>

먼저 시퀀스 t_seq를 생성한다.

```
SQL> create sequence t_seq
start with 12;
```

```
SQL> execute p_trade_insert('301','101',10);
```
상품재고가 10개 부족합니다.

```
SQL> execute p_trade_insert('101','101',10);
```

완료되었습니다.

작성 실패

(문제 6) 거래가 취소되었다. 거래내역을 삭제하여라.(재고복구)
프로시저 이름 : p_trade_delete

<실행>

SQL> execute p_trade_delete(15);
없는 거래번호입니다.

SQL> execute p_trade_delete(6);
완료되었습니다.

```
CREATE OR REPLACE PROCEDURE p_trade_delete(
  p_t_seq IN NUMBER
) AS
  v_거래수량 NUMBER;
  v_상품코드 VARCHAR2(50);
BEGIN
  BEGIN
    SELECT T_QTY, P_CODE INTO v_거래수량, v_상품코드
    FROM TRADE
    WHERE T_SEQ = p_t_seq;

    EXCEPTION
      WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('완료되었습니다.');
        RETURN;
  END;
```

```
    UPDATE STOCK
    SET S_QTY = S_QTY + v_거래수량
    WHERE P_CODE = v_상품코드;

    DELETE FROM TRADE
    WHERE T_SEQ = p_t_seq;

    DBMS_OUTPUT.PUT_LINE('존재하지 않는 거래번호
```

```
  입니다');
  END;
```

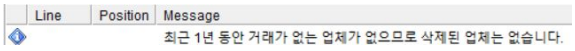
(문제 7) 최근 1년 동안 거래가 없는 업체 삭제하여라.

프로시저 이름 : p_trade_not_delete

<실행>

SQL> execute p_trade_not_delete();

※ 최근 1년 동안 거래가 없는 업체가 없으므로 삭제된 업체는 없음



Line	Position	Message
		최근 1년 동안 거래가 없는 업체가 없으므로 삭제된 업체는 없습니다.

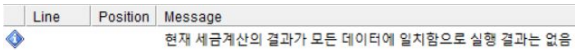
```
  CREATE OR REPLACE PROCEDURE p_trade_not_delete AS
    v_cutoff_date DATE;
    v_constraint_exists NUMBER;
  BEGIN
    v_cutoff_date := SYSDATE - 365;

    SELECT COUNT(*)
    INTO v_constraint_exists
    FROM USER_CONSTRAINTS
    WHERE TABLE_NAME = 'TRADE' AND CONSTRAINT_NAME =
  'TRA_C_CODE_FK';

    IF v_constraint_exists > 0 THEN
      EXECUTE IMMEDIATE 'ALTER TABLE TRADE DROP CONSTRAINT
  TRA_C_CODE_FK';
    END IF;
```

```
  DELETE FROM CUSTOMER
  WHERE C_CODE NOT IN (
    SELECT DISTINCT C_CODE
    FROM TRADE
    WHERE T_DATE > v_cutoff_date
  );

  IF v_constraint_exists > 0 THEN
    DELETE FROM TRADE
    WHERE C_CODE NOT IN (
      SELECT C_CODE
      FROM CUSTOMER
    );

    EXECUTE IMMEDIATE 'ALTER TABLE TRADE ADD CONSTRAINT
TRA_C_CODE_FK FOREIGN KEY (C_CODE) REFERENCES CUSTOMER(C_CODE) ON
DELETE CASCADE';
  END IF;

  IF SQL%ROWCOUNT = 0 THEN
    DBMS_OUTPUT.PUT_LINE('최근 1년 동안 거래가 없는 업체가 없으므로 삭제된 업
체는 없습니다.');
  ELSE
    DBMS_OUTPUT.PUT_LINE('최근 1년 동안 거래가 없는 업체가 삭제되었습니다.');
  END IF;
END;
```

(문제 8) 기존 세금 계산이 틀렸다면 수정하여라.
프로시저 이름 : p_tax_check

```
execute p_tax_check();
```

※ 현재 세금계산의 결과가 모든 데이터에 일치함으로 실행 결과는 없음




```
CREATE OR REPLACE PROCEDURE p_tax_check AS
  v_tax_rate NUMBER := 0.1;
BEGIN
  UPDATE TRADE
  SET T_TAX = ROUND(T_COST * v_tax_rate, 2);

  DBMS_OUTPUT.PUT_LINE('현재 세금계산의 결과가 모든 데이터에 일치함으로 실행
```

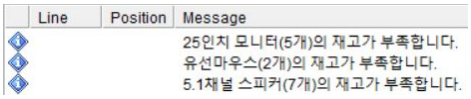
결과는 없음');
END;

(문제 9) 적정재고보다 현 재고량 보다 적은 상품 알아내기(적정재고는 재고가 10개 미만이거나 지난 3개월의 월별 평균 판매량)
프로시저 이름 : p_stock_fill

<실행>

SQL> execute p_stock_fill;

25인치 모니터(5개)의 재고가 부족합니다.
5.1채널 스피커(7개)의 재고가 부족합니다.

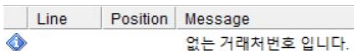



```
CREATE OR REPLACE PROCEDURE p_stock_fill AS
  v_monthly_avg_sales_threshold NUMBER := 10;
BEGIN
  FOR product IN (SELECT P.P_NAME, S.S_QTY
                  FROM PRODUCT P
                  JOIN STOCK S ON P.P_CODE = S.P_CODE
                  WHERE S.S_QTY < 10 OR
                        (MONTHS_BETWEEN(SYSDATE, S.S_LASTDATE) <= 3 AND
(SELECT NVL(AVG(T.T_QTY), 0) FROM TRADE T WHERE T.P_CODE = P.P_CODE)
< v_monthly_avg_sales_threshold))
  LOOP
    DBMS_OUTPUT.PUT_LINE(product.P_NAME || '(' || product.S_QTY || '개)의 재
고가 부족합니다.');
  END LOOP;
END;
```

(문제 10) 기존 거래처 번호를 새로운 거래처 번호를 수정하는 프로시저를 생성하여라.
프로시저 이름 : p_c_code_update

<실행>

SQL> execute p_c_code_update('999','501');
없는 거래처번호입니다.



SQL> execute p_c_code_update('102','101');
교체될 거래처번호가 이미 있습니다.

Line	Position	Message
		없는 거래처번호 입니다.

SQL> execute p_c_code_update('101','501');
완료되었습니다. ???

Line	Position	Message
		없는 거래처번호 입니다.

```
CREATE OR REPLACE PROCEDURE p_c_code_update(
  p_old_c_code IN VARCHAR2,
  p_new_c_code IN VARCHAR2
) AS
  v_old_c_code_exists NUMBER;
  v_new_c_code_exists NUMBER;
BEGIN
  SELECT COUNT(*)
  INTO v_old_c_code_exists
  FROM CUSTOMER
  WHERE C_CODE = p_old_c_code;

  SELECT COUNT(*)
  INTO v_new_c_code_exists
  FROM CUSTOMER
  WHERE C_CODE = p_new_c_code;

  IF v_old_c_code_exists = 0 THEN
    DBMS_OUTPUT.PUT_LINE('없는 거래처번호 입니다.');
  ELSIF v_new_c_code_exists > 0 THEN
    DBMS_OUTPUT.PUT_LINE('교체될 거래처번호가 이미 있습니다.');
  ELSE
    UPDATE TRADE
    SET C_CODE = p_new_c_code
    WHERE C_CODE = p_old_c_code;

    UPDATE CUSTOMER
    SET C_CODE = p_new_c_code
    WHERE C_CODE = p_old_c_code;

    DBMS_OUTPUT.PUT_LINE('완료되었습니다.');
  END IF;
END;
```

(문제 11) 상품코드를 입력받아 재고량을 반환하는 함수를 작성하여라.
함수 이름 : f_stock_return

<실행>

```
SQL> variable y NUMBER
execute :y := f_stock_return(501);
```

상품코드가 없습니다.

```
SQL> execute :y := f_stock_return(501);
```

y
20

```
CREATE OR REPLACE FUNCTION f_stock_return(
  p_p_code IN VARCHAR2
) RETURN NUMBER AS
  v_stock_qty NUMBER;
BEGIN
  SELECT S_QTY
  INTO v_stock_qty
  FROM STOCK
  WHERE P_CODE = p_p_code;

  RETURN v_stock_qty;
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('상품코드가 없습니다.');
    RETURN NULL;
END;
```