Lab Report of CSE464

Lab - 4

Submitted By:

Aklhak Hossain
2022-3-60-057
https://akhlak.com

Submitted To:

Antu Chowdhury
Lecturer
Department of CSE
East West University

PRE-LAB TASK:

CODE: CREATE STUDENT TABLE

```
CREATE TABLE STUDENTS (
  STUDENT_ID NUMBER CONSTRAINT STUDENT_PK PRIMARY KEY,
  NAME VARCHAR2(100) NOT NULL,
  SEMESTER_FEE NUMBER NOT NULL,
  CURRENT_SEMESTER NUMBER NOT NULL,
  DEPT_NAME VARCHAR2(100) NOT NULL
);
```

```
SQL> CREATE TABLE STUDENTS (
  STUDENT_ID NUMBER CONSTRAINT STUDENT_PK PRIMARY KEY,
  NAME VARCHAR2(100) NOT NULL,
  SEMESTER_FEE NUMBER NOT NULL,
  CURRENT_SEMESTER NUMBER NOT NULL,
  DEPT_NAME VARCHAR2(100) NOT NULL
);
  2    3    4    5    6    7
Table created.
```

LAB TASK 1:

CODE: INSERT STUDENT CODE:

```
CREATE OR REPLACE PROCEDURE INSERT_STUDENT (
  P_STUDENT_ID IN STUDENTS.STUDENT_ID%TYPE,
  P_NAME IN STUDENTS.NAME%TYPE,
  P_SEMESTER_FEE IN STUDENTS.SEMESTER_FEE%TYPE,
  P_CURRENT_SEMESTER IN STUDENTS.CURRENT_SEMESTER%TYPE,
  P_DEPT_NAME IN STUDENTS.DEPT_NAME%TYPE
) AS
BEGIN
  INSERT INTO STUDENTS (STUDENT_ID, NAME, SEMESTER_FEE, CURRENT_SEMESTER,
DEPT_NAME)
  VALUES (P_STUDENT_ID, P_NAME, P_SEMESTER_FEE, P_CURRENT_SEMESTER,
P_DEPT_NAME);
END;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE INSERT_STUDENT (
  P_STUDENT_ID IN STUDENTS.STUDENT_ID%TYPE,
  P_NAME IN STUDENTS.NAME%TYPE,
  P_SEMESTER_FEE IN STUDENTS.SEMESTER_FEE%TYPE,
  P_CURRENT_SEMESTER IN STUDENTS.CURRENT_SEMESTER%TYPE,
  P_DEPT_NAME IN STUDENTS.DEPT_NAME%TYPE
) AS
BEGIN
  INSERT INTO STUDENTS (STUDENT_ID, NAME, SEMESTER_FEE, CURRENT_SEMESTER, DEPT_NA
ME)
  VALUES (P_STUDENT_ID, P_NAME, P_SEMESTER_FEE, P_CURRENT_SEMESTER, P_DEPT_NAME);
END;
/
  2    3    4    5    6    7    8    9   10   11   12
Procedure created.
```

LAB TASK 2:

CODE: PROCEDURE UPDATE_FEE

```
CREATE OR REPLACE PROCEDURE UPDATE_FEE (
  P_STUDENT_ID IN  STUDENTS.STUDENT_ID%TYPE,
  P_AMOUNT     IN  NUMBER,
  P_NEW_FEE    OUT NUMBER
) AS
BEGIN
  UPDATE STUDENTS
     SET SEMESTER_FEE = SEMESTER_FEE + P_AMOUNT
   WHERE STUDENT_ID = P_STUDENT_ID;
  IF SQL%ROWCOUNT = 0 THEN
    RAISE_APPLICATION_ERROR(-20001, 'STUDENT NOT FOUND: ' || P_STUDENT_ID);
  END IF;
  SELECT SEMESTER_FEE
    INTO P_NEW_FEE
    FROM STUDENTS
   WHERE STUDENT_ID = P_STUDENT_ID;
END;
/
```

```
SQL> CREATE OR REPLACE PROCEDURE UPDATE_FEE (
  P_STUDENT_ID IN  STUDENTS.STUDENT_ID%TYPE,
  P_AMOUNT     IN  NUMBER,
  P_NEW_FEE    OUT NUMBER
) AS
BEGIN
  UPDATE STUDENTS
     SET SEMESTER_FEE = SEMESTER_FEE + P_AMOUNT
   WHERE STUDENT_ID = P_STUDENT_ID;
  IF SQL%ROWCOUNT = 0 THEN
    RAISE_APPLICATION_ERROR(-20001, 'STUDENT NOT FOUND: ' || P_STUDENT_ID);
  END IF;
  SELECT SEMESTER_FEE
    INTO P_NEW_FEE
    FROM STUDENTS
   WHERE STUDENT_ID = P_STUDENT_ID;
END;
/
  2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17
18
Procedure created.
```

LAB TASK 3:

```
CREATE OR REPLACE FUNCTION CALCULATE_SCHOLARSHIP (
  P_SEMESTER_FEE IN NUMBER
) RETURN NUMBER AS
BEGIN
  RETURN ROUND(P_SEMESTER_FEE * 0.15, 2);
END;
/
```

```
SQL> CREATE OR REPLACE FUNCTION CALCULATE_SCHOLARSHIP (
  P_SEMESTER_FEE IN NUMBER
) RETURN NUMBER AS
BEGIN
  RETURN ROUND(P_SEMESTER_FEE * 0.15, 2);
END;
/
  2    3    4    5    6    7
Function created.
```

LAB TASK 4:

```
CREATE TABLE BACKUP_STUDENTS (
  STUDENT_ID       NUMBER        NOT NULL,
  NAME             VARCHAR2(100) NOT NULL,
  SEMESTER_FEE     NUMBER        NOT NULL,
  CURRENT_SEMESTER NUMBER        NOT NULL,
  DEPT_NAME        VARCHAR2(100) NOT NULL,
  DELETED_AT       DATE,
  DELETED_BY       VARCHAR2(50)
);
```

```
SQL> CREATE TABLE BACKUP_STUDENTS (
  STUDENT_ID       NUMBER        NOT NULL,
  NAME             VARCHAR2(100) NOT NULL,
  SEMESTER_FEE     NUMBER        NOT NULL,
  CURRENT_SEMESTER NUMBER        NOT NULL,
  DEPT_NAME        VARCHAR2(100) NOT NULL,
  DELETED_AT       DATE,
  DELETED_BY       VARCHAR2(50)
);
  2    3    4    5    6    7    8    9
Table created.
```

LAB TASK 5:

```
CREATE OR REPLACE TRIGGER TRG_STUDENTS_BDEL
BEFORE DELETE ON STUDENTS
FOR EACH ROW
BEGIN
  INSERT INTO BACKUP_STUDENTS (
    STUDENT_ID, NAME, SEMESTER_FEE, CURRENT_SEMESTER, DEPT_NAME,
    DELETED_AT, DELETED_BY
  ) VALUES (
    :OLD.STUDENT_ID, :OLD.NAME, :OLD.SEMESTER_FEE, :OLD.CURRENT_SEMESTER,
:OLD.DEPT_NAME,
    SYSDATE, USER
  );
END;
/
```

```
SQL> CREATE OR REPLACE TRIGGER TRG_STUDENTS_BDEL
BEFORE DELETE ON STUDENTS
FOR EACH ROW
BEGIN
  INSERT INTO BACKUP_STUDENTS (
    STUDENT_ID, NAME, SEMESTER_FEE, CURRENT_SEMESTER, DEPT_NAME,
    DELETED_AT, DELETED_BY
  ) VALUES (
    :OLD.STUDENT_ID, :OLD.NAME, :OLD.SEMESTER_FEE, :OLD.CURRENT_SEMESTER, :OLD.DE
PT_NAME,
    SYSDATE, USER
  );
END;
/
  2    3    4    5    6    7    8    9   10   11   12   13
Trigger created.
```

LAB TASK 6:

```
CREATE OR REPLACE TRIGGER BACKUP_STUDENT_BEFORE_UPDATE
BEFORE UPDATE ON STUDENTS
FOR EACH ROW
BEGIN
  INSERT INTO BACKUP_STUDENTS (
    STUDENT_ID, NAME, SEMESTER_FEE, CURRENT_SEMESTER, DEPT_NAME,
    DELETED_AT, DELETED_BY
  ) VALUES (
    :OLD.STUDENT_ID, :OLD.NAME, :OLD.SEMESTER_FEE, :OLD.CURRENT_SEMESTER,
:OLD.DEPT_NAME,
    SYSDATE, USER
  );
END;
/
```

```
SQL> CREATE OR REPLACE TRIGGER BACKUP_STUDENT_BEFORE_UPDATE
BEFORE UPDATE ON STUDENTS
FOR EACH ROW
BEGIN
  INSERT INTO BACKUP_STUDENTS (
    STUDENT_ID, NAME, SEMESTER_FEE, CURRENT_SEMESTER, DEPT_NAME,
    DELETED_AT, DELETED_BY
  ) VALUES (
    :OLD.STUDENT_ID, :OLD.NAME, :OLD.SEMESTER_FEE, :OLD.CURRENT_SEMESTER, :OLD.DE
PT_NAME,
    SYSDATE, USER
  );
END;
/
  2    3    4    5    6    7    8    9   10   11   12   13
Trigger created.

SQL>
```

LAB TASK 7:

```
CREATE OR REPLACE TRIGGER BACKUP_STUDENT_AFTER_INSERT
AFTER INSERT ON STUDENTS
FOR EACH ROW
BEGIN
  INSERT INTO BACKUP_STUDENTS (
    STUDENT_ID, NAME, SEMESTER_FEE, CURRENT_SEMESTER, DEPT_NAME,
    DELETED_AT, DELETED_BY
  ) VALUES (
    :NEW.STUDENT_ID, :NEW.NAME, :NEW.SEMESTER_FEE, :NEW.CURRENT_SEMESTER,
:NEW.DEPT_NAME,
    SYSDATE, USER
  );
END;
/
```

```
SQL> CREATE OR REPLACE TRIGGER BACKUP_STUDENT_AFTER_INSERT
AFTER INSERT ON STUDENTS
FOR EACH ROW
BEGIN
  INSERT INTO BACKUP_STUDENTS (
    STUDENT_ID, NAME, SEMESTER_FEE, CURRENT_SEMESTER, DEPT_NAME,
    DELETED_AT, DELETED_BY
  ) VALUES (
    :NEW.STUDENT_ID, :NEW.NAME, :NEW.SEMESTER_FEE, :NEW.CURRENT_SEMESTER, :NEW.DE
PT_NAME,
    SYSDATE, USER
  );
END;
/
  2    3    4    5    6    7    8    9   10   11   12   13
Trigger created.
```