Islamic University of Technology

CSE-4308

Database Management Systems Lab

Lab Report-10

Name : Mukit Mahdin

ID : 200042170

Department : CSE

Program : SWE

Group : B

Task - 1:

Decrease the budget of the departments having a budget of more than 99999 by 10%. Show the number of departments that did not get affected.

Code:

```
SET SERVEROUTPUT ON
CREATE OR REPLACE PROCEDURE NUMBEROFROWS (ROWNUMBERS OUT NUMBER)
AS
BEGIN
	SELECT COUNT(*) INTO ROWNUMBERS FROM DEPARTMENT;
END;
/

CREATE OR REPLACE PROCEDURE DECREASEBUDGET(ROWNUMBERS OUT NUMBER)
AS
BEGIN
	UPDATE DEPARTMENT
	SET BUDGET=BUDGET - (BUDGET*0.1)
	WHERE BUDGET>99999;

	IF(SQL%FOUND) THEN
		ROWNUMBERS:=SQL%ROWCOUNT;
	END IF;
END;
/

DECLARE
	ROWS NUMBER;
	ROWS1 NUMBER;
BEGIN
	NUMBEROFROWS(ROWS);
	DECREASEBUDGET(ROWS1);
	DBMS_OUTPUT.PUT_LINE(ROWS-ROWS1);
END;
/
```

Explanation :

In the above task, NUMOFROWS and DECREASEBUDGET procedures are created. NUMOFROWS will return the number of total rows in the DEPARTMENT table and DECREASEBUDGET will decrease the budget and return the number of affected rows. To get unaffected rows, I just subtracted the number of affected rows from the total rows.

Task-2 :

Write a procedure that takes the day of the week and starting hour and ending hour as input and prints which instructors should be in the class during that time.

Code :

```
CREATE OR REPLACE PROCEDURE TAKINGCLASS (DAY IN VARCHAR, STARTHR IN  NUMBER,ENDHR
IN NUMBER)
AS
BEGIN
	FOR I IN(SELECT ID,NAME
			FROM INSTRUCTOR NATURAL JOIN TIME_SLOT
			WHERE TIME_SLOT.START_HR>=STARTHR AND TIME_SLOT.END_HR <=ENDHR  AND
DAY=TIME_SLOT.DAY) LOOP
				DBMS_OUTPUT.PUT_LINE(I.ID|| ' ' || I.NAME);
	END LOOP;
END;
/
BEGIN
	TAKINGCLASS('M',8,12);
END;
/
```

Explanation :

TAKINGCLASS procedure is taking the DAY, STARTHR, and ENDHR as input and it will print the ID and Name of the instructors who should be in class during that time.

Task-3:

Write a procedure to find the top N students based on the number of courses they are enrolled in. The procedure should take N as input and print the ID, name, department name, and the number of courses taken by the student.

```
CREATE OR REPLACE PROCEDURE MOSTNUMBEROFCOURSES (N IN NUMBER)
AS
BEGIN
	FOR I IN(SELECT ID, NAME, DEPT_NAME, COUNT(COURSE_ID) AS NUMOFCOURESE
			FROM STUDENT NATURAL JOIN COURSE
```

```
                WHERE ROWNUM<=N
                GROUP BY ID,NAME,DEPT_NAME
                 ORDER BY COUNT(COURSE_ID) DESC) LOOP
                        DBMS_OUTPUT.PUT_LINE(I.ID|| CHR(8) || I.NAME||CHR(8) ||
                         I.DEPT_NAME||CHR(8) || I.NUMOFCOURESE);
    END LOOP;
END;
/
BEGIN
    MOSTNUMBEROFCOURSES(8);
END;
/
```

Explanation :

MOSTNUMBEROFCOURSES procedure is taking the "n" as a parameter and after the operation, it will give the output as n number of rows ordered by NUMOFCOURSE DESCENDING and print the STUDENTID, NAME and Total Number of Courses taken.

Task-4 :

Create a trigger that automatically generates IDs for instructors when we insert data into INSTRUCTOR table.

Code :

```
CREATE SEQUENCE INSTRUCTOR_SEQ
MINVALUE 10001
MAXVALUE 99999
START WITH 10001
INCREMENT BY 1
CACHE 20;

CREATE OR REPLACE TRIGGER INSTRUCTOR_ID_GENERATOR
BEFORE INSERT ON INSTRUCTOR
FOR EACH ROW
BEGIN
    :NEW.ID:=INSTRUCTOR_SEQ.NEXTVAL;
END;
/
INSERT INTO INSTRUCTOR VALUES('0', 'MUKIT', 'SWE', '31256');
```

```
INSERT INTO INSTRUCTOR VALUES('0', 'SHUVRO','COMP. SCI','31257');
```

Explanation :

In the above task, the procedure should be created with necessary credentials. Here, INSTRUCTOR_ID_GENERATOR is the trigger and it will be triggered when a new instructor is inserted and the INSTRUCTOR_SEQ sequence will provide an unique ID using NEW and NEXTVAL.

Task-5 :

Create a trigger that will automatically assign an advisor to a newly admitted student of his/her own department. In case there are multiple teachers, the advisor should be selected based on the least number of students advised.

```
CREATE OR REPLACE TRIGGER INSTRUCTOR_ASSIGNMENT
BEFORE INSERT ON STUDENT
FOR EACH ROW
DECLARE
    INS_ID INSTRUCTOR.ID%TYPE;
BEGIN
    SELECT ID INTO INS_ID
    FROM(
        SELECT ID,COUNT(S_ID) AS STUDENTNUM
        FROM ADVISOR NATURAL JOIN INSTRUCTOR
        GROUP BY ID
        ORDER BY COUNT(S_ID) ASC
       )
    WHERE ROWNUM=1;
    INSERT INTO ADVISOR VALUES(:NEW.ID,INS_ID);
END;
/
INSERT INTO STUDENT VALUES('31256','MUKIT','SWE','123');
```

Explanation :

The trigger INSTRUCTOR_ASSIGNMENT is created to be triggered if an insertion operation is done in the STUDENT table. And to provide the condition given in the trigger it will invoke a query that will return the least student-assigned instructor and assign the inserted student to that particular instructor.