Islamic University of Technology

RDBMS

CSE 4508

Lab Report 6

Name : Amina

Student ID : 200041155

Section : 1

Lab Group : 1B (Shifted from 1A)

Date of Performance : 25/10/23

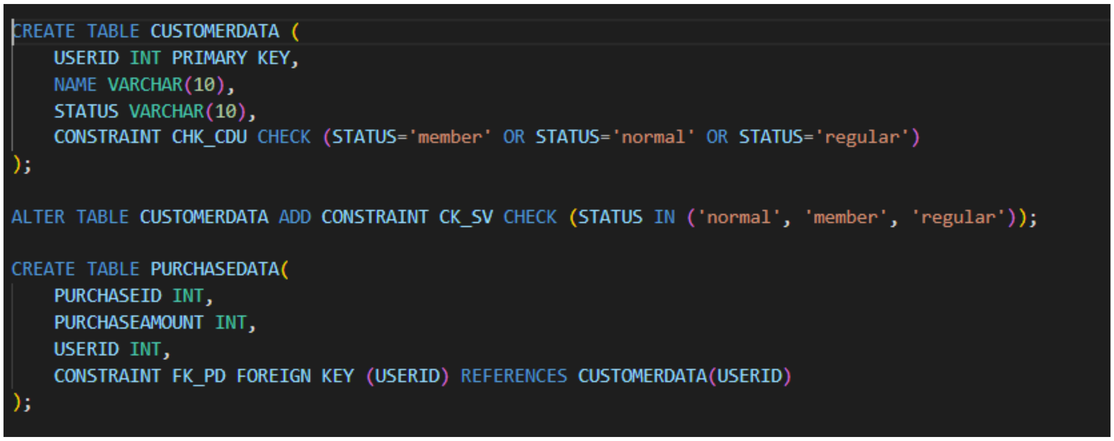
Date of Submission : 26/10/23

Task A

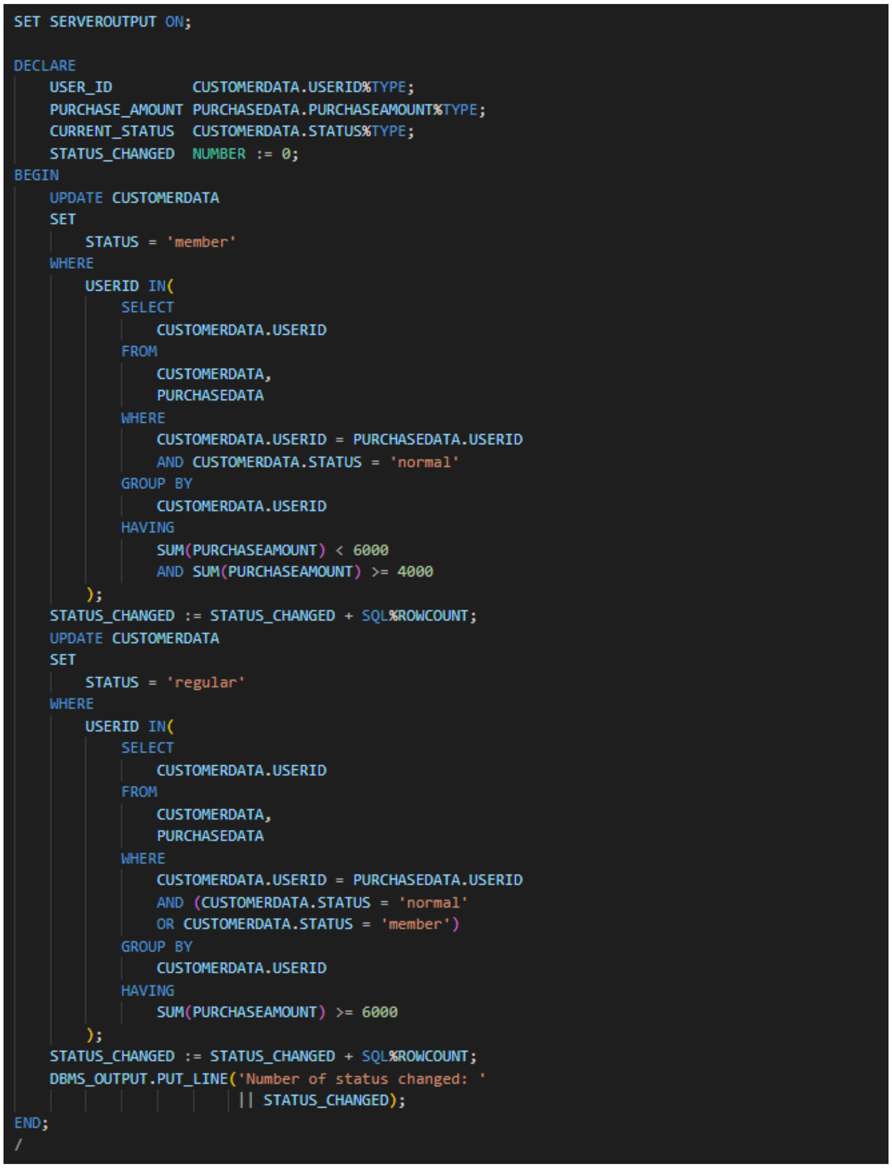
In task A,

1. I created two tables as needed. In one of these tables, I made sure that the "status" column can only have values like "member," "normal," or "regular" by adding a constraint.
2. I then wrote a PL/SQL block, which is like a script in the Oracle database language. Inside this block, I ran two queries.
3. In the first query, I updated customer records with a "normal" status and a total purchase amount between 4000 and 6000. I changed their status to "member."
4. In the second query, I updated customer records that have a "member" or "normal" status and a total purchase amount greater than 6000. I changed their status to "regular."
5. After each query, I counted the number of rows that were changed using SQL%ROWCOUNT.

Table creation and PL/SQL procedure:

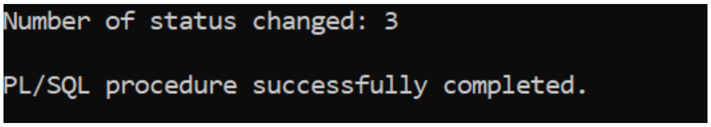


```
CREATE TABLE CUSTOMERDATA (
    USERID INT PRIMARY KEY,
    NAME VARCHAR(10),
    STATUS VARCHAR(10),
    CONSTRAINT CHK_CDU CHECK (STATUS='member' OR STATUS='normal' OR STATUS='regular')
);

ALTER TABLE CUSTOMERDATA ADD CONSTRAINT CK_SV CHECK (STATUS IN ('normal', 'member', 'regular'));

CREATE TABLE PURCHASEDATA(
    PURCHASEID INT,
    PURCHASEAMOUNT INT,
    USERID INT,
    CONSTRAINT FK_PD FOREIGN KEY (USERID) REFERENCES CUSTOMERDATA(USERID)
);
```



```
SET SERVEROUTPUT ON;

DECLARE
    USER_ID         CUSTOMERDATA.USERID%TYPE;
    PURCHASE_AMOUNT PURCHASEDATA.PURCHASEAMOUNT%TYPE;
    CURRENT_STATUS  CUSTOMERDATA.STATUS%TYPE;
    STATUS_CHANGED  NUMBER := 0;
BEGIN
    UPDATE CUSTOMERDATA
    SET
        STATUS = 'member'
    WHERE
        USERID IN(
            SELECT
                CUSTOMERDATA.USERID
            FROM
                CUSTOMERDATA,
                PURCHASEDATA
            WHERE
                CUSTOMERDATA.USERID = PURCHASEDATA.USERID
                AND CUSTOMERDATA.STATUS = 'normal'
            GROUP BY
                CUSTOMERDATA.USERID
            HAVING
                SUM(PURCHASEAMOUNT) < 6000
                AND SUM(PURCHASEAMOUNT) >= 4000
        );
    STATUS_CHANGED := STATUS_CHANGED + SQL%ROWCOUNT;
    UPDATE CUSTOMERDATA
    SET
        STATUS = 'regular'
    WHERE
        USERID IN(
            SELECT
                CUSTOMERDATA.USERID
            FROM
                CUSTOMERDATA,
                PURCHASEDATA
            WHERE
                CUSTOMERDATA.USERID = PURCHASEDATA.USERID
                AND (CUSTOMERDATA.STATUS = 'normal'
                OR CUSTOMERDATA.STATUS = 'member')
            GROUP BY
                CUSTOMERDATA.USERID
            HAVING
                SUM(PURCHASEAMOUNT) >= 6000
        );
    STATUS_CHANGED := STATUS_CHANGED + SQL%ROWCOUNT;
    DBMS_OUTPUT.PUT_LINE('Number of status changed: '
                         || STATUS_CHANGED);
END;
/
```

Result:



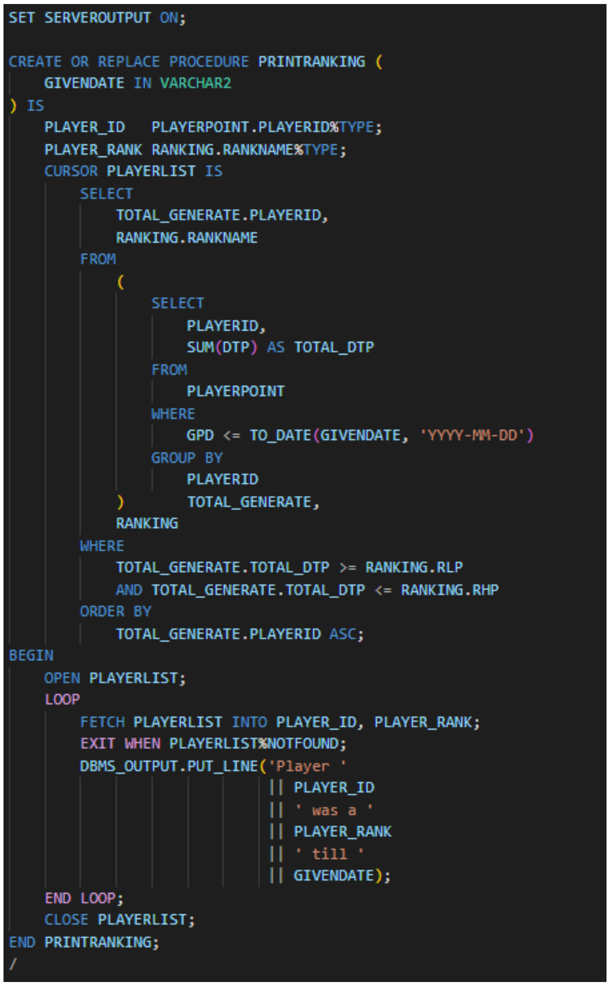
```
Number of status changed: 3

PL/SQL procedure successfully completed.
```

Task B

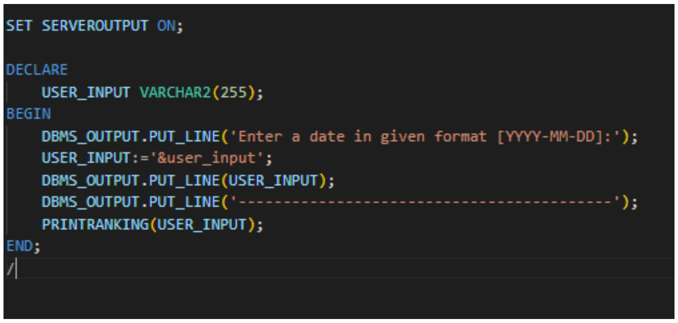
In this task, we had two tables: one with player records and the other with ranking information.

I made a special function using PL/SQL that takes a date as input. Then, I wrote a query to figure out which player had which rank name up until that date.

Implementation :

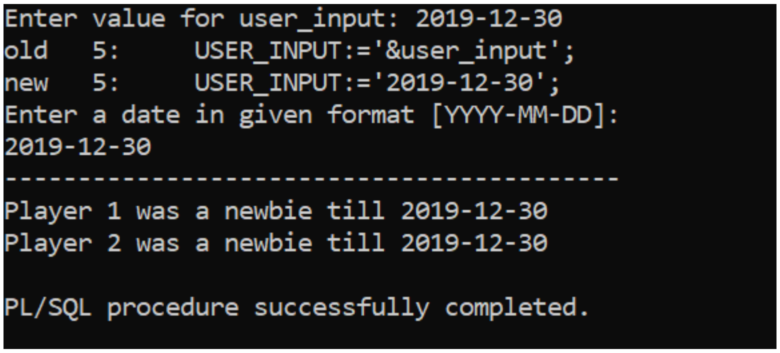


```
SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE PRINTRANKING (
    GIVENDATE IN VARCHAR2
) IS
    PLAYER_ID   PLAYERPOINT.PLAYERID%TYPE;
    PLAYER_RANK RANKING.RANKNAME%TYPE;
    CURSOR PLAYERLIST IS
        SELECT
            TOTAL_GENERATE.PLAYERID,
            RANKING.RANKNAME
        FROM
            (
                SELECT
                    PLAYERID,
                    SUM(DTP) AS TOTAL_DTP
                FROM
                    PLAYERPOINT
                WHERE
                    GPD <= TO_DATE(GIVENDATE, 'YYYY-MM-DD')
                GROUP BY
                    PLAYERID
            )   TOTAL_GENERATE,
            RANKING
        WHERE
            TOTAL_GENERATE.TOTAL_DTP >= RANKING.RLP
            AND TOTAL_GENERATE.TOTAL_DTP <= RANKING.RHP
        ORDER BY
            TOTAL_GENERATE.PLAYERID ASC;
BEGIN
    OPEN PLAYERLIST;
    LOOP
        FETCH PLAYERLIST INTO PLAYER_ID, PLAYER_RANK;
        EXIT WHEN PLAYERLIST%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('Player '
                             || PLAYER_ID
                             || ' was a '
                             || PLAYER_RANK
                             || ' till '
                             || GIVENDATE);
    END LOOP;
    CLOSE PLAYERLIST;
END PRINTRANKING;
/
```



```
SET SERVEROUTPUT ON;

DECLARE
    USER_INPUT VARCHAR2(255);
BEGIN
    DBMS_OUTPUT.PUT_LINE('Enter a date in given format [YYYY-MM-DD]:');
    USER_INPUT:='&user_input';
    DBMS_OUTPUT.PUT_LINE(USER_INPUT);
    DBMS_OUTPUT.PUT_LINE('------------------------------------------');
    PRINTRANKING(USER_INPUT);
END;
/
```

Result:



```
Enter value for user_input: 2019-12-30
old   5:     USER_INPUT:='&user_input';
new   5:     USER_INPUT:='2019-12-30';
Enter a date in given format [YYYY-MM-DD]:
2019-12-30
------------------------------------------
Player 1 was a newbie till 2019-12-30
Player 2 was a newbie till 2019-12-30

PL/SQL procedure successfully completed.
```