Islamic University of Technology

RDBMS

CSE 4508

Lab Report 6

Name : Abdullah

Student ID : 200041126

Section : 1

Lab Group : 1B

Date of Performance : 25/10/23

Date of Submission : 26/10/23

Task A

In task A,

- I created 2 tables as required. I added CHECK constraint to make sure status can only be member, normal or regular.
- Then I wrote a PL/SQL block. In this block I run two queries.
- In the first query I updated customer data whose status are normal and total PURCHASEAMOUNT is between 4000 and 6000. I set their status to 'member'.
- After the query I calculated how many records are affected by this query. This is done by using SQL%ROWCOUNT.
- Then I wrote another query to update customer who are member or normal and their PURCHASEAMOUNT is total greater than 6000. I set their status to 'regular'.
- Then, again, I used SQL%ROWCOUNT to find out affected number of rows by the last query.

Table creation and PL/SQL procedure:

```
-- TASK A
DROP TABLE CustomerData;
CREATE TABLE CustomerData (
    userId INTEGER,
    name VARCHAR2(50) NOT NULL,
    status VARCHAR2(25) NOT NULL,
    CONSTRAINT cd_pk PRIMARY KEY (userId),
    CONSTRAINT st_chk CHECK (status = 'normal' OR status = 'member' OR status = 'regular')
);

ALTER TABLE CustomerData
ADD CONSTRAINT st_upd_chk
CHECK (status IN ('normal','member','regular'));

DROP TABLE PurchaseData;
CREATE TABLE PurchaseData (
    purchaseId INTEGER,
    purchaseAmount INTEGER NOT NULL,
    userId INTEGER,
    CONSTRAINT pd_pk PRIMARY KEY (purchaseId),
    CONSTRAINT pd_cd_fk FOREIGN KEY (userId) REFERENCES CustomerData (userId) ON DELETE CASCADE
);
```

```
SET SERVEROUTPUT ON;

DECLARE
    num_customer_status_changed INTEGER := 0;
BEGIN
    UPDATE CustomerData
    SET status = 'member'
    WHERE userId IN (
        SELECT cd.userId as selectedUserId
        FROM CustomerData cd, PurchaseData pd
        WHERE cd.userId = pd.userId AND cd.status = 'normal'
        GROUP BY cd.userId
        HAVING SUM(pd.purchaseAmount) BETWEEN 4000 AND 5999
    );

    num_customer_status_changed := num_customer_status_changed + SQL%ROWCOUNT;

    UPDATE CustomerData
    SET status = 'regular'
    WHERE userId IN (
        SELECT cd.userId as selectedUserId
        FROM CustomerData cd, PurchaseData pd
        WHERE cd.userId = pd.userId AND (cd.status = 'normal' OR cd.status = 'member')
        GROUP BY cd.userId
        HAVING SUM(pd.purchaseAmount) >= 6000
    );

    num_customer_status_changed := num_customer_status_changed + SQL%ROWCOUNT;

    DBMS_OUTPUT.PUT_LINE('Number of customer status changed: ' || num_customer_status_changed);
END;
/
```

Result:

```
SQL> SET SERVEROUTPUT ON;
SQL>
SQL> DECLARE
  2      num_customer_status_changed INTEGER := 0;
  3  BEGIN
  4      UPDATE CustomerData
  5      SET status = 'member'
  6      WHERE userId IN (
  7          SELECT cd.userId as selectedUserId
  8          FROM CustomerData cd, PurchaseData pd
  9          WHERE cd.userId = pd.userId AND cd.status = 'normal'
 10          GROUP BY cd.userId
 11          HAVING SUM(pd.purchaseAmount) BETWEEN 4000 AND 5999
 12      );
 13
 14      num_customer_status_changed := num_customer_status_changed + SQL%ROWCOUNT;
 15
 16      UPDATE CustomerData
 17      SET status = 'regular'
 18      WHERE userId IN (
 19          SELECT cd.userId as selectedUserId
 20          FROM CustomerData cd, PurchaseData pd
 21          WHERE cd.userId = pd.userId AND (cd.status = 'normal' OR cd.status = 'member')
 22          GROUP BY cd.userId
 23          HAVING SUM(pd.purchaseAmount) >= 6000
 24      );
 25
 26      num_customer_status_changed := num_customer_status_changed + SQL%ROWCOUNT;
 27
 28      DBMS_OUTPUT.PUT_LINE('Number of customer status changed: ' || num_customer_status_changed);
 29  END;
 30  /
Number of customer status changed: 3

PL/SQL procedure successfully completed.
```

Task B

In this task, we had two tables: one with player records and another for ranking methods.

I created a PL/SQL procedure that takes a date as input. Then, I crafted a query to determine, up to that specified date, which player held which rank.

We were also told to take input from user. So, I created a separate block of code to take user input and called the procedure there.

Below is the implementation of the procedure :

```
SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE printRanking
(givenDate IN VARCHAR2)
IS
    pid PlayerPoint.playerId%type;
    prnk Ranking.rankName%type;

    CURSOR playerList
    IS
        SELECT spp.playerId AS plid, rnk.rankName AS plrnk
        FROM
        (
            SELECT playerId, SUM(dTP) AS tDTP
            FROM PlayerPoint
            WHERE gPD <= TO_DATE(givenDate, 'YYYY-MM-DD')
            GROUP BY playerId
        ) spp, Ranking rnk
        WHERE spp.tDTP >= rnk.rLP AND spp.tDTP <= rnk.rHP
        ORDER BY spp.playerId ASC;
BEGIN
    OPEN playerList;
        LOOP
            FETCH playerList into pid, prnk;
            EXIT WHEN playerList%notfound;
            dbms_output.put_line('Player ' || pid || ' was a ' || prnk || ' till ' || givenDate);
        END LOOP;
    CLOSE playerList;
END printRanking;
/

SHOW ERROR;
```

```
DECLARE
    ugdate VARCHAR2(255);
BEGIN
    dbms_output.put_line('Enter a date in given format [YYYY-MM-DD]:');
    ugdate:='&ugdate';
    dbms_output.put_line(ugdate);
    dbms_output.put_line('----------------------------------------');

    printRanking(ugdate);
END;
/
```

Result:

```
SQL> DECLARE
  2       ugdate VARCHAR2(255);
  3   BEGIN
  4       dbms_output.put_line('Enter a date in given format [YYYY-MM-DD]:');
  5       ugdate:='&ugdate';
  6       dbms_output.put_line(ugdate);
  7       dbms_output.put_line('----------------------------------------');
  8
  9       printRanking(ugdate);
 10   END;
 11   /
Enter value for ugdate: 2022-12-31
old   5:      ugdate:='&ugdate';
new   5:      ugdate:='2022-12-31';
Enter a date in given format [YYYY-MM-DD]:
2022-12-31
----------------------------------------
Player 1 was a pupil till 2022-12-31
Player 2 was a newbie till 2022-12-31
Player 3 was a pupil till 2022-12-31

PL/SQL procedure successfully completed.

SQL>
```

```
SQL> DECLARE
  2      ugdate VARCHAR2(255);
  3  BEGIN
  4      dbms_output.put_line('Enter a date in given format [YYYY-MM-DD]:');
  5      ugdate:='&ugdate';
  6      dbms_output.put_line(ugdate);
  7      dbms_output.put_line('----------------------------------------');
  8
  9      printRanking(ugdate);
 10  END;
 11  /
Enter value for ugdate: 2019-12-31
old   5:      ugdate:='&ugdate';
new   5:      ugdate:='2019-12-31';
Enter a date in given format [YYYY-MM-DD]:
2019-12-31
----------------------------------------
Player 1 was a newbie till 2019-12-31
Player 2 was a newbie till 2019-12-31

PL/SQL procedure successfully completed.

SQL>
```