7. PL/SQL PROCEDURE FOR EB BILL PREPARATION USING CURSORS

AIM

To write a PL / SQL procedure for preparing electricity bill using cursors.

ALGORITHM

Step – I : Create a user-defined type Addr as follows:

```
SQL> create or replace type Addr as object (DN
     varchar2(10), SN varchar2(15), Loc varchar2(15),
     City varchar2(20), PIN number(6))
     /
```

Create tables EB_Customer and EB_Bill as follows:

```
SQL> create table EB_Customer (ServiceNo Number (4), Name
     Varchar2 (30) Constraint NN_Name Not Null, Address
     Addr, DepAmt Number (9, 2), Constraint PK_SerNo
     Primary Key (ServiceNo), Constraint CHK_DepAmt Check
     (DepAmt > 0));

SQL> create table EB_Bill (Reading_Dt Date, Service_No
     Number(4), Prev_Read Number(5), Curr_Read Number(5),
     Tot_Units Number(5), Amount Number(9,2), Constraint
     FK_SerNo Foreign Key (Service_No) References
     EB_Customer (ServiceNo) On Delete Cascade);
```

Step – II : Insert records in the EB_Customer and EB_Bill tables by using the Insert commands as given below:

```
SQL> Insert Into EB_Customer Values (&ServiceNo, '&Name',
     Addr ('&DN', '&SN', '&Loc', '&City', &PIN),
     &DepAmt);

SQL> Insert Into EB_Bill (Reading_Dt, Service_No,
     Prev_Read, Curr_Read) Values (SysDate, &Service_No,
     &Prev_Read, &Curr_Read);
```

Step – III : Type ed in the SQL Editor to open a notepad window. Key in the PL/SQL procedure as given below and save it using a filename (say E:\SKS\MC9217\Cursor_Demo.txt).

Step – IV : In the PL / SQL procedure, declare a cursor object, which selects records from EB_Bill table where Tot_Units is Null.

Step – V : Open the cursor object.

Step – VI : Fetch a record and store it in the tuple variable. Find the total_units consumed, and calculate the bill amount by using the table given below:

<table>
<tr><td colspan="4">Total units consumed ≤ 100</td></tr>
<tr><td>Units</td><td>Amount Per Unit</td><td>Units</td><td>Amount Per Unit</td></tr>
<tr><td>1 – 50</td><td>0.65</td><td>51 – 100</td><td>0.75</td></tr>
<tr><td colspan="4">Total units consumed > 100</td></tr>
<tr><td>Units</td><td>Amount Per Unit</td><td>Units</td><td>Amount Per Unit</td></tr>
<tr><td>1 – 50</td><td>0.75</td><td>201 – 600</td><td>2.20</td></tr>
<tr><td>51 – 100</td><td>0.85</td><td rowspan="2">Above 600</td><td rowspan="2">3.05</td></tr>
<tr><td>101 – 200</td><td>1.50</td></tr>
<tr><td colspan="3">Standard Charges</td><td>Rs. 10.00</td></tr>
<tr><td colspan="3">Minimum EB Charges</td><td>Rs. 40.00</td></tr>
</table>

Step – VII : Update the Tot_Units and Amount attributes of the EB_Bill table.

Step – VIII: Repeat Steps VI and VII for each record in the EB_Bill table where Tot_Units is Null.

Step – IX : Close the cursor object.

Step – X : Execute the PL / SQL procedure as shown below:

```
SQL> @ E:\SKS\MC9217\Cursor_Demo.txt
```

PL / SQL PROCEDURE

```
declare
 total_units eb_bill.tot_units%type;
 bill_amt eb_bill.amount%type;
 eb_row eb_bill%rowtype;
 cursor eb is select * from eb_bill where tot_units is null;
begin
 open eb;
 loop
  fetch eb into eb_row;
  total_units := eb_row.curr_read - eb_row.prev_read;
  if total_units <= 100 then
   if total_units <= 50 then
    bill_amt := total_units * 0.65;
   else
    bill_amt := 50 * 0.65 + (total_units - 50) * 0.75;
   end if;
  else
   if total_units <= 50 then
    bill_amt := total_units * 0.75;
   elsif total_units <= 100 then
    bill_amt := 50 * 0.75 + (total_units - 50) * 0.85;
   elsif total_units <= 200 then
    bill_amt := 50 * 0.75 + 50 * 0.85 + (total_units - 100) * 1.5;
   elsif total_units <=600 then
    bill_amt :=50*0.75 + 50*0.85 + 100*1.5 + (total_units-200)*2.2;
   else
    bill_amt :=50*0.75+50*0.85+100*1.5+400*2.2+(total_units-600)*3.05;
   end if;
  end if;
  bill_amt := bill_amt + 10;
  if bill_amt < 40 then
   bill_amt := 40;
  else
   bill_amt := round (bill_amt, 0);
  end if;
  update eb_bill set tot_units = total_units, amount = bill_amt
  where service_no = eb_row.service_no and tot_units is null;
  exit when eb%NOTFOUND;
 end loop;
 close eb;
end;
/
```

OUTPUT

```
SQL> @ E:\SKS\MC9217\Cursor_Demo.txt

      PL/SQL procedure successfully completed.
```

RESULT: Thus, the above program was executed successfully.