Date: 14 - 02 - 2026
Assignment No.: 6

Name: Varghese K James
Reg No.: 3122245001192

Excercise 6 - PL/SQL - Cursors, Stored Procedures & Functions

Aim:

- To create procedures and functions using implicit and explicit cursors.

Syntax for Commands Used

Anonymous Blocks

Used for executing PL/SQL code without storing it in the database schema.

```
DECLARE
   -- Variable declarations
BEGIN
   -- Executable statements
END;
/
```

Procedures

Procedures are stored blocks that can take parameters.

```
CREATE OR REPLACE PROCEDURE procedure_name (parameter_name IN
datatype) IS
   -- Local declarations
BEGIN
   -- Executable statements
END;
/
```

Functions

Functions are similar to procedures but must return a single value.

```
CREATE OR REPLACE FUNCTION function_name (parameter_name IN
datatype)
RETURN return_datatype IS
   -- Local declarations
BEGIN
   -- Executable statements
   RETURN value;
END;
/
```

Explicit Cursors

Used to handle multiple rows returned by a query.

```
CURSOR cursor_name IS
    SELECT ... FROM table WHERE condition;

OPEN cursor_name;
FETCH cursor_name INTO record_variable;
IF cursor_name%NOTFOUND THEN
    -- logic for empty result
END IF;
CLOSE cursor_name;
```

Implicit Cursors

```
UPDATE table_name SET col = val WHERE condition;

IF SQL%NOTFOUND THEN
    -- Logic if no rows were affected
END IF;
```

Demo Script File

```
set pagesize 100;
-- making the table
set echo off;
set serveroutput off;
set feedback off;

@TABLE_CREATION.sql
@SAILOR_VALUES.sql

set serveroutput on;
set feedback on;
set echo on;
set verify off;
-- Done.

-- Q1
declare
    v_type boat.type%type := '&type';
    v_color boat.color%type := '&color';
    v_found boolean := false;

    cursor c_boats is
        select * from boat
        where type = v_type and color = v_color;
    r_boat c_boats%rowtype;
begin
    dbms_output.put_line('Checking availability for ' || v_type || ' boats in ' ||
v_color || '...');

    open c_boats;
    loop
        fetch c_boats into r_boat;
        exit when c_boats%notfound;
        v_found := true;
        dbms_output.put_line('id: ' || r_boat.boat_id || ' | name: ' || r_boat.boat_name
|| ' | capacity: ' || r_boat.capacity || ' | price: $' || r_boat.price);
    end loop;
    close c_boats;

    if not v_found then
        dbms_output.put_line('No ' || v_type || ' boats found with color ' || v_color ||
'.');
    end if;
end;
/

-- Q2
declare
    -- eg 31-03-05
    v_input_date varchar2(20) := '&sail_date';
    v_total_people number;

    cursor c_count is
        select sum(no_of_people + no_of_children)
        from reservation
        where sail_date = to_date(v_input_date, 'dd-mm-rr');
begin
    open c_count;
    fetch c_count into v_total_people;
    close c_count;

    if v_total_people is null or v_total_people = 0 then
        dbms_output.put_line('No passengers found for the date: ' || v_input_date);
    else
        dbms_output.put_line('Total passengers sailed on ' || v_input_date || ': ' ||
v_total_people);
    end if;
end;
/

-- Q3
create or replace procedure get_reservation_details(p_resv_id in
reservation.resv_id%type) is
    cursor c_resv is
        select t.tourist_name, b.boat_name, b.type, s.sailor_name, r.no_of_people,
r.no_of_children
        from reservation r
        join tourist t on r.tourist_id = t.tourist_id
        join boat b on r.boat_id = b.boat_id
        join sailor s on r.sailor_id = s.sailor_id
        where r.resv_id = p_resv_id;

    r_data c_resv%rowtype;
begin
    open c_resv;
    fetch c_resv into r_data;

    if c_resv%notfound then
        dbms_output.put_line('Error: reservation id ' || p_resv_id || ' does not
exist.');
    else
        dbms_output.put_line('Tourist name: ' || r_data.tourist_name);
        dbms_output.put_line('Boat name: ' || r_data.boat_name || ' (' || r_data.type ||
')');
        dbms_output.put_line('Sailor name: ' || r_data.sailor_name);
        dbms_output.put_line('Adults: ' || r_data.no_of_people || ', children: ' ||
r_data.no_of_children);
        dbms_output.put_line('Total passengers: ' || (r_data.no_of_people +
r_data.no_of_children));
    end if;
    close c_resv;
end;
/

begin
    get_reservation_details('R013');
end;
/

-- Q4
create or replace procedure generate_trip_bill(p_resv_id in reservation.resv_id%type) is
    cursor c_bill_data is
        select t.tourist_name, r.sail_date, b.boat_name, b.type, s.sailor_name, b.price
        from reservation r
        join tourist t on r.tourist_id = t.tourist_id
        join boat b on r.boat_id = b.boat_id
        join sailor s on r.sailor_id = s.sailor_id
        where r.resv_id = p_resv_id;

    r_bill c_bill_data%rowtype;
    v_extra_charge number := 450.00;
    v_safety_fee number := 150.00;
    v_total_amount number;
    v_discount_pct number := 0;
    v_discount_amt number;
    v_final_payable number;
begin
    open c_bill_data;
    fetch c_bill_data into r_bill;

    if c_bill_data%notfound then
        dbms_output.put_line('error: reservation details not found for id ' ||
p_resv_id);
    else
        v_total_amount := r_bill.price + v_extra_charge + v_safety_fee;
        if v_total_amount > 500 and v_total_amount < 2000 then
            v_discount_pct := 5;
        elsif v_total_amount >= 2000 and v_total_amount < 5000 then
            v_discount_pct := 10;
        elsif v_total_amount >= 5000 then
            v_discount_pct := 20;
        end if;

        v_discount_amt := v_total_amount * (v_discount_pct / 100);
        v_final_payable := v_total_amount - v_discount_amt;

        update payment
        set amount = v_final_payable, p_status = 'completed'
        where resv_id = p_resv_id;

        if sql%notfound then
            dbms_output.put_line('warning: payment record not found for update.');
        end if;


dbms_output.put_line('************************************************************');
        dbms_output.put_line('Reservation Number: ' || p_resv_id || ' Tourist Name: ' ||
r_bill.tourist_name);
        dbms_output.put_line('Sail Date : ' || TO_CHAR(r_bill.sail_date, 'DD-Mon-YYYY'));
        dbms_output.put_line('Boat Name : ' || UPPER(r_bill.boat_name) || ' Boat Type: '
|| UPPER(r_bill.type));
        dbms_output.put_line('Sailor Name : ' || UPPER(r_bill.sailor_name));

dbms_output.put_line('************************************************************');
        dbms_output.put_line('Sno  Description                Amount');
        dbms_output.put_line('1.   Boat Trip Charge           ' || TO_CHAR(r_bill.price,
'9990.00'));
        dbms_output.put_line('2.   Extra Passenger Charges    ' ||
TO_CHAR(v_extra_charge, '9990.00'));
        dbms_output.put_line('3.   Safety Equipment Fee       ' || TO_CHAR(v_safety_fee,
'9990.00'));

dbms_output.put_line('------------------------------------------------------------');
        dbms_output.put_line('Total Amount = $ ' || TO_CHAR(v_total_amount, '9990.00'));

dbms_output.put_line('------------------------------------------------------------');
        dbms_output.put_line('Discount (' || v_discount_pct || '%) = $ ' ||
TO_CHAR(v_discount_amt, '9990.00'));

dbms_output.put_line('------------------------------------------------------------');
        dbms_output.put_line('Amount to be Paid = $ ' || TO_CHAR(v_final_payable,
'9990.00'));

dbms_output.put_line('************************************************************');
        dbms_output.put_line('🎉 Special Offer! Get 15% off on your next Island
Cruise!');

dbms_output.put_line('************************************************************');
    end if;
    close c_bill_data;
end;
```

```
/

begin
    generate_trip_bill('R013');
end;
/

-- Q5: Recommend Boat Procedure
create or replace procedure recommend_boat(p_budget in number, p_type in boat.type%type)
is
    cursor c_rec is
        select b.boat_id, b.boat_name, b.price
        from boat b
        join (
            select boat_id, count(*) as resv_count
            from reservation
            group by boat_id
        ) r_count on b.boat_id = r_count.boat_id
        where b.type = p_type
        and b.price <= p_budget
        order by r_count.resv_count desc
        fetch first 1 row only;

    r_boat c_rec%rowtype;
    v_trips number;
begin
    open c_rec;
    fetch c_rec into r_boat;

    if c_rec%notfound then
        dbms_output.put_line('no suitable ' || p_type || ' boat found within your budget
of $' || p_budget);
    else
        v_trips := floor(p_budget / r_boat.price);
        dbms_output.put_line('Recommended boat: ' || r_boat.boat_name || ' (id: ' ||
r_boat.boat_id || ')');
        dbms_output.put_line('Price per trip: $' || r_boat.price);
        dbms_output.put_line('With your budget of $' || p_budget || ', you can afford '
|| v_trips || ' trip(s).');
    end if;
    close c_rec;
end;
/

begin
    recommend_boat(320, 'CAR');
end;
/

-- Q6
create or replace function get_top_tourist(p_boat_id in boat.boat_id%type)
return varchar2 is
    cursor c_top is
        select t.tourist_name, count(*) as cnt
        from reservation r
        join tourist t on r.tourist_id = t.tourist_id
        where r.boat_id = p_boat_id
        group by t.tourist_name
        order by cnt desc
        fetch first 1 row only;

    v_name tourist.tourist_name%type;
    v_count number;
begin
    open c_top;
    fetch c_top into v_name, v_count;

    if c_top%notfound then
        close c_top;
        return 'No bookings exist for this boat.';
    else
        close c_top;
        return v_name || ' (reservations: ' || v_count || ')';
    end if;
end;
/

begin
    dbms_output.put_line('Top Tourist for B001: ' || get_top_tourist('B001'));
end;
/
```

Output Spool File

```
SQL>
SQL> -- Q1
SQL> declare
  2      v_type boat.type%type := '&type';
  3      v_color boat.color%type := '&color';
  4      v_found boolean := false;
  5
  6      cursor c_boats is
  7          select * from boat
  8          where type = v_type and color = v_color;
  9
 10      r_boat c_boats%rowtype;
 11  begin
 12      dbms_output.put_line('Checking availability for ' || v_type || ' boats in ' ||
v_color || '...');
 13
 14      open c_boats;
 15      loop
 16          fetch c_boats into r_boat;
 17          exit when c_boats%notfound;
 18          v_found := true;
 19          dbms_output.put_line('id: ' || r_boat.boat_id || ' | name: ' ||
r_boat.boat_name || ' | capacity: ' || r_boat.capacity || ' | price: $' || r_boat.price);
 20      end loop;
 21      close c_boats;
 22
 23      if not v_found then
 24          dbms_output.put_line('No ' || v_type || ' boats found with color ' ||
v_color || '.');
 25      end if;
 26  end;
 27  /
Enter value for type: LUX
Enter value for color: White
Checking availability for LUX boats in White...
id: B001 | name: Sea Queen | capacity: 10 | price: $500
id: B004 | name: Pearl Boat | capacity: 8 | price: $600
id: B013 | name: Marine Star | capacity: 9 | price: $680


PL/SQL procedure successfully completed.

SQL>
SQL> -- Q2
SQL> declare
  2      -- eg 31-03-05
  3      v_input_date varchar2(20) := '&sail_date';
  4      v_total_people number;
  5
  6      cursor c_count is
  7          select sum(no_of_people + no_of_children)
  8          from reservation
  9          where sail_date = to_date(v_input_date, 'dd-mm-rr');
 10  begin
 11      open c_count;
 12      fetch c_count into v_total_people;
 13      close c_count;
 14
 15      if v_total_people is null or v_total_people = 0 then
 16          dbms_output.put_line('No passengers found for the date: ' || v_input_date);
 17      else
 18          dbms_output.put_line('Total passengers sailed on ' || v_input_date || ': '
|| v_total_people);
 19      end if;
 20  end;
 21  /
Enter value for sail_date: 31-03-05
Total passengers sailed on 31-03-05: 16


PL/SQL procedure successfully completed.

SQL>
SQL> -- Q3
SQL> create or replace procedure get_reservation_details(p_resv_id in
reservation.resv_id%type) is
  2      cursor c_resv is
  3          select t.tourist_name, b.boat_name, b.type, s.sailor_name, r.no_of_people,
r.no_of_children
  4          from reservation r
  5          join tourist t on r.tourist_id = t.tourist_id
  6          join boat b on r.boat_id = b.boat_id
  7         join sailor s on r.sailor_id = s.sailor_id
  8         where r.resv_id = p_resv_id;
  9
 10     r_data c_resv%rowtype;
 11  begin
 12     open c_resv;
 13     fetch c_resv into r_data;
 14
 15     if c_resv%notfound then
 16         dbms_output.put_line('Error: reservation id ' || p_resv_id || ' does not
exist.');
 17     else
 18         dbms_output.put_line('Tourist name: ' || r_data.tourist_name);
 19         dbms_output.put_line('Boat name: ' || r_data.boat_name || ' (' ||
r_data.type || ')');
 20         dbms_output.put_line('Sailor name: ' || r_data.sailor_name);
 21         dbms_output.put_line('Adults: ' || r_data.no_of_people || ', children: ' ||
r_data.no_of_children);
 22         dbms_output.put_line('Total passengers: ' || (r_data.no_of_people +
r_data.no_of_children));
 23     end if;
 24     close c_resv;
 25  end;
 26  /

Procedure GET_RESERVATION_DETAILS compiled

SQL>
SQL> begin
  2     get_reservation_details('R013');
  3  end;
  4  /
Tourist name: Rajesh
Boat name: River Queen (CAR)
Sailor name: Ravi
Adults: 3, children: 0
Total passengers: 3


PL/SQL procedure successfully completed.

SQL>
SQL> -- Q4
SQL> create or replace procedure generate_trip_bill(p_resv_id in
reservation.resv_id%type) is
  2     cursor c_bill_data is
  3         select t.tourist_name, r.sail_date, b.boat_name, b.type, s.sailor_name,
b.price
  4         from reservation r
  5         join tourist t on r.tourist_id = t.tourist_id
  6         join boat b on r.boat_id = b.boat_id
  7         join sailor s on r.sailor_id = s.sailor_id
  8         where r.resv_id = p_resv_id;
  9
 10     r_bill c_bill_data%rowtype;
 11     v_extra_charge number := 450.00;
```

```
    v_safety_fee number := 150.00;
    v_total_amount number;
    v_discount_pct number := 0;
    v_discount_amt number;
    v_final_payable number;
begin
    open c_bill_data;
    fetch c_bill_data into r_bill;

    if c_bill_data%notfound then
        dbms_output.put_line('error: reservation details not found for id ' || p_resv_id);
    else
        v_total_amount := r_bill.price + v_extra_charge + v_safety_fee;

        if v_total_amount > 500 and v_total_amount < 2000 then
            v_discount_pct := 5;
        elsif v_total_amount >= 2000 and v_total_amount < 5000 then
            v_discount_pct := 10;
        elsif v_total_amount >= 5000 then
            v_discount_pct := 20;
        end if;

        v_discount_amt := v_total_amount * (v_discount_pct / 100);
        v_final_payable := v_total_amount - v_discount_amt;

        update payment
        set amount = v_final_payable, p_status = 'completed'
        where resv_id = p_resv_id;

        if sql%notfound then
            dbms_output.put_line('warning: payment record not found for update.');
        end if;

        dbms_output.put_line('************************************************************');
        dbms_output.put_line('Reservation Number: ' || p_resv_id || ' Tourist Name: ' || r_bill.tourist_name);
        dbms_output.put_line('Sail Date : ' || TO_CHAR(r_bill.sail_date, 'DD-Mon-YYYY'));
        dbms_output.put_line('Boat Name : ' || UPPER(r_bill.boat_name) || ' Boat Type: ' || UPPER(r_bill.type));
        dbms_output.put_line('Sailor Name : ' || UPPER(r_bill.sailor_name));
        dbms_output.put_line('************************************************************');
        dbms_output.put_line('Sno  Description                Amount');
        dbms_output.put_line('1.   Boat Trip Charge           ' || TO_CHAR(r_bill.price, '9990.00'));
        dbms_output.put_line('2.   Extra Passenger Charges    ' || TO_CHAR(v_extra_charge, '9990.00'));
        dbms_output.put_line('3.   Safety Equipment Fee       ' || TO_CHAR(v_safety_fee, '9990.00'));
        dbms_output.put_line('------------------------------------------------------------');
        dbms_output.put_line('Total Amount = $ ' || TO_CHAR(v_total_amount, '9990.00'));
```

```
 57
dbms_output.put_line('------------------------------------------------------------');
 58          dbms_output.put_line('Discount (' || v_discount_pct || '%) = $ ' ||
TO_CHAR(v_discount_amt, '9990.00'));
 59
dbms_output.put_line('------------------------------------------------------------');
 60          dbms_output.put_line('Amount to be Paid = $ ' || TO_CHAR(v_final_payable,
'9990.00'));
 61
dbms_output.put_line('************************************************************');
 62          dbms_output.put_line('🎉 Special Offer! Get 15% off on your next Island
Cruise!');
 63
dbms_output.put_line('************************************************************');
 64      end if;
 65      close c_bill_data;
 66  end;
 67  /

Procedure GENERATE_TRIP_BILL compiled

SQL>
SQL> begin
  2      generate_trip_bill('R013');
  3  end;
  4  /
************************************************************
Reservation Number: R013 Tourist Name: Rajesh
Sail Date : 12-Mar-2005
Boat Name : RIVER QUEEN Boat Type: CAR
Sailor Name : RAVI
************************************************************
Sno  Description                Amount
1.   Boat Trip Charge             320.00
2.   Extra Passenger Charges      450.00
3.   Safety Equipment Fee         150.00
------------------------------------------------------------
Total Amount = $   920.00
------------------------------------------------------------
Discount (5%) = $    46.00
------------------------------------------------------------
Amount to be Paid = $   874.00
************************************************************
🎉 Special Offer! Get 15% off on your next Island Cruise!
************************************************************


PL/SQL procedure successfully completed.

SQL>
SQL> -- Q5: Recommend Boat Procedure
SQL> create or replace procedure recommend_boat(p_budget in number, p_type in
boat.type%type) is
  2      cursor c_rec is
  3          select b.boat_id, b.boat_name, b.price
  4          from boat b
  5          join (
  6              select boat_id, count(*) as resv_count
  7              from reservation
  8              group by boat_id
  9          ) r_count on b.boat_id = r_count.boat_id
 10          where b.type = p_type
 11          and b.price <= p_budget
 12          order by r_count.resv_count desc
 13          fetch first 1 row only;
 14
 15      r_boat c_rec%rowtype;
 16      v_trips number;
 17  begin
 18      open c_rec;
 19      fetch c_rec into r_boat;
 20
 21      if c_rec%notfound then
 22          dbms_output.put_line('no suitable ' || p_type || ' boat found within your
budget of $' || p_budget);
 23      else
 24          v_trips := floor(p_budget / r_boat.price);
 25          dbms_output.put_line('Recommended boat: ' || r_boat.boat_name || ' (id: ' ||
r_boat.boat_id || ')');
 26          dbms_output.put_line('Price per trip: $' || r_boat.price);
 27          dbms_output.put_line('With your budget of $' || p_budget || ', you can
afford ' || v_trips || ' trip(s).');
 28      end if;
 29      close c_rec;
 30  end;
 31  /

Procedure RECOMMEND_BOAT compiled

SQL>
SQL> begin
  2      recommend_boat(320, 'CAR');
  3  end;
  4  /
Recommended boat: River Queen (id: B008)
Price per trip: $320
With your budget of $320, you can afford 1 trip(s).


PL/SQL procedure successfully completed.

SQL>
SQL> -- Q6
SQL> create or replace function get_top_tourist(p_boat_id in boat.boat_id%type)
  2  return varchar2 is
  3      cursor c_top is
  4          select t.tourist_name, count(*) as cnt
  5          from reservation r
  6          join tourist t on r.tourist_id = t.tourist_id
  7          where r.boat_id = p_boat_id
  8          group by t.tourist_name
  9          order by cnt desc
 10          fetch first 1 row only;
 11
 12      v_name tourist.tourist_name%type;
 13      v_count number;
 14  begin
 15      open c_top;
 16      fetch c_top into v_name, v_count;
 17
 18      if c_top%notfound then
 19          close c_top;
 20          return 'No bookings exist for this boat.';
 21      else
 22          close c_top;
 23          return v_name || ' (reservations: ' || v_count || ')';
 24      end if;
 25  end;
 26  /

Function GET_TOP_TOURIST compiled

SQL>
SQL> begin
  2      dbms_output.put_line('Top Tourist for B001: ' || get_top_tourist('B001'));
  3  end;
  4  /
Top Tourist for B001: Anitha (reservations: 2)


PL/SQL procedure successfully completed.

SQL> SPOOL off;
```

Learning Outcomes

- Learned how to create procedures and functions in PL/SQL
- Understood the proper use of cursors, both implicit and explicit
- Familiarised with the concepts of control flow in PL/SQL