Procedures, functions and loops

Date:-12/9/25

Case study:- online food ordering system

Objective:- The objective of this task is to design, implement and execute PL/SQL Procedures, functions and loops to handle real-world business scenarios related to an online food ordering system. This will help in automating transactions, improving database efficiency and enforcing business rules as structured programmes

EP1:- Ensure the necessary tables exists:-

Before running the procedure and functions, create the required tables in your oracle Database

```
DROP TABLE order table PURGE;
DROP TABLE Delivery PURGE;
DROP TABLE Menu - Item PURGE;

CREATE TABLE order table (order - ID Number primary key, rest - ID Number, order - Date DATE, order - total Number (10,2), payment - status varchar (20));
CREATE TABLE Delivery (order - ID Number primary key, Delivery - status varchar2(20) foreign key (order - ID) References order table (order ID);
INSERT into order table values (1, 101, To_date ('2024-10-01', 'YYYY-MM-DD'), 2500.50, 'Ordering');
```

```
Discount Applied: 10%
Statement processed.
```

```
GET_TOTAL_REVENUE()
-------------------
             801.25
```

```
Payment status updated successfully
for order ID: 1
Statement processed.
```

- Employee (Employee_ID, Name, Department_ID, Job_Title, Hire_Date, Salary) 🔗
- Department (Department_ID, Manager_ID) 🔗
- Manager (Manager_ID, Name)

Insert into Delivery values (1, 'Pending');

Insert into Delivery values (2, 'Delivered');

Insert into Delivery values (3, 'Pending');

1. Procedure to update payment status

step2 : create a procedure

create OR Replace Procedure update_payment_status (p_order_ID IN Number, p_new_status IN varchar2(20))

update order table set payment status = p_new_status where order_ID = p_order_ID;

Commit;

DBMS_output.put_line ('Payment status updated for order ID: ' || p_order_ID);

End;

Expected output:

procedure created

step2:- Execution:

Begin

update_payment_status

End;

Expected output:

Payment status updated successfully for order ID: statement processed.

Query2:- function to calculate total revenue

step1:- create a function

Begin

select sum(order_total) Revenue number

```
Order ID: 1, Date: 01-FEB-24, Total:
250.5, Status: Paid
Statement processed.
```

order_ID	Cust_ID	order_Date	order_total	payment_statu
1	101	2024-02-01	250.50	Paid
2	102	2024-02-02	400.75	Paid
3	103	2024-02-03	150.00	Pending

Item_ID	Item_name	price
1	Pizza	450.00
2	Burger	450.00
3	Pasta	405.00

order table;
Return V_total_Revenue;
END;
/

Expected output:-
Function created

Query:- Procedure to apply Discount on menu Items

Step1:- create a procedure

```
create or Replace procedure Apply
   _Discount(
   discount_precent In Number
)
IS
Begin
Update menu_Item
  Set price = price -(price * discount
       _percent/100);
  commit;
  DBMS_output.put_line('Discount Applied :'||discount
     percent||'%');
End;
/
```

Excepected output
Procedure created

step2:- excution

```
Begin
  Apply_Discount(10);
End;
/
```

VEL TECH CSE	
EX	10
[illegible]	5
[illegible]	5
VIVA VOCE (5)	5
R[illegible] (5)	5
TOTAL (20)	20
SIGN WITH DATE	[illegible]

Result: Thus, the PL/SQL procedure function and loops or number theory experiment was successfully completed ...