Query -1

1. procedure to update payment status

order_ID	cust_ID	order_date	order _Total	Parameter status.
1	101	2024-02-01	250.00	
2	102	2024-02-02	400.75	
3	103	2024-02-01	150.00	

Query-2

Total revenue
calling the get_total_revenue
function

Output

801.25

Task-6

Procedures, Functions, and loops in PL/SQL (Based on online Food Ordering system) case Study: online Food ordering system

objective :-

This will help in automating transactions improving database efficiency, and enforcing bussiness rules in a structural manner

Step-1:- Ensure the Necessary Tables Exists

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

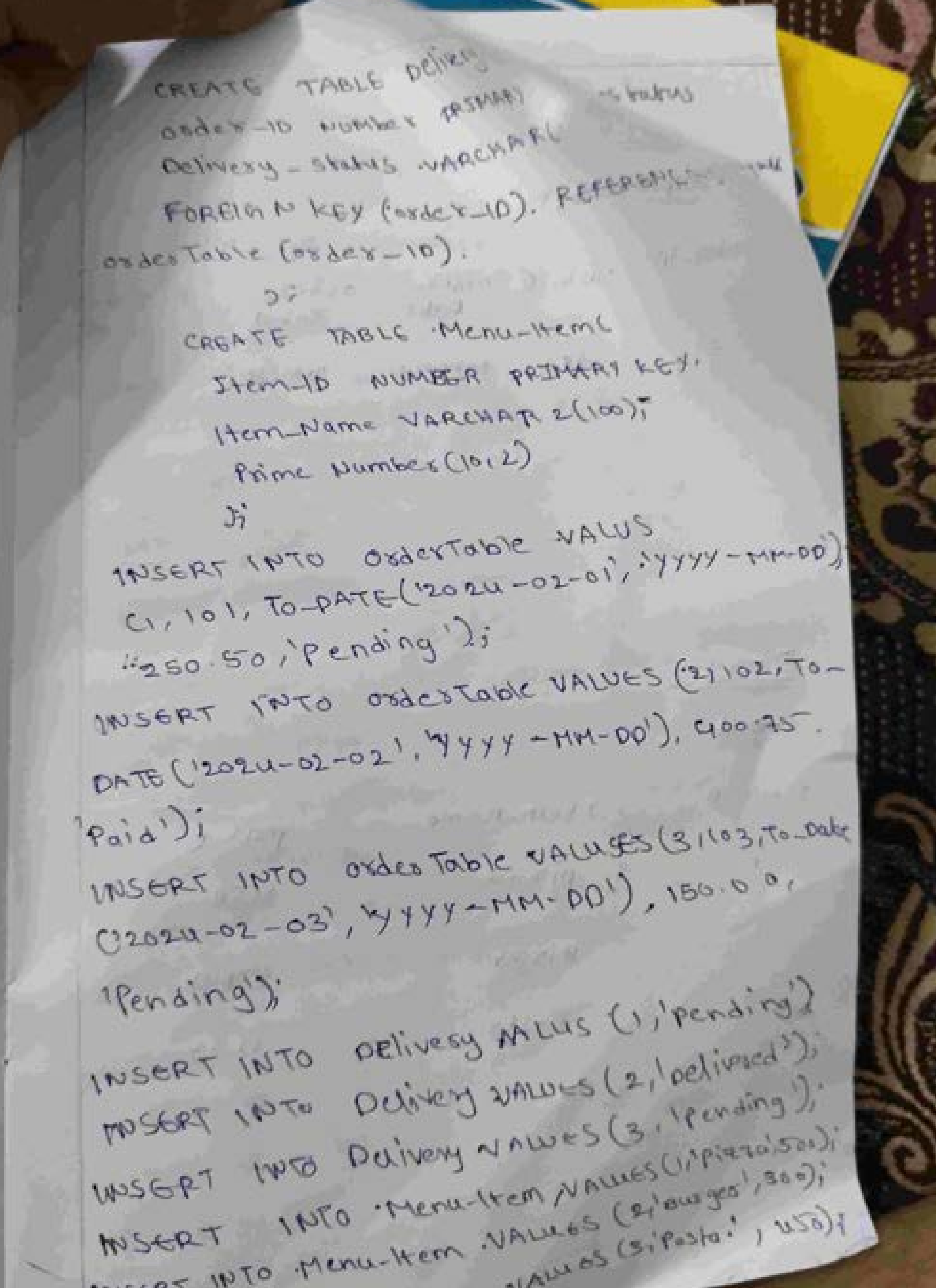
```
DROP TABLE OrderTable PURGE;
DROP TABLE Delivery PURGE;
DROP TABLE Menu-Item PURGE;

CREATE TABLE OrderTable (
  order_ID NUMBER PRIMARY KEY,
   Cust_ID Number;
   order_Date DATE,
   order_Total Number(10,2);
   Payment_status VARCHAR (20).
   );
```

Query 3:-

Loop 1:- Mark all undelivered order as "Delayed"

order_ID	Delivery_status
1	Delayed
2	Delivered
3	Delayed



```
CREATE TABLE Delivery
order-ID Number PRIMARY          status
Delivery_status VARCHAR(
FOREIGN KEY (order_ID). REFERENCES
orderTable (order_ID).
    );
CREATE TABLE Menu-Item(
  Item_ID NUMBER PRIMARY KEY,
  Item_Name VARCHAR 2(100);
  Prime Number(10,2)
  );
INSERT INTO OrderTable VALUS
 (1, 101, TO_DATE('2024-02-01', 'YYYY-MM-DD'),
 250.50, 'Pending');
INSERT INTO orderTable VALUES (2, 102, To-
DATE ('2024-02-02', 'YYYY-MM-DD'), 400.75.
'Paid');
INSERT INTO orderTable VALUES (3, 103, To_Date
('2024-02-03', 'YYYY-MM-DD'), 150.00,
'Pending');
INSERT INTO Delivery VALUS (1, 'pending')
INSERT INTO Delivery VALUES (2, 'Delivered');
INSERT INTO Delivery VALUES (3, 'pending');
INSERT INTO Menu-Item VALUES (1, 'Pizza', 500);
INSERT INTO Menu-Item VALUES (2, 'burger', 300);
INSERT INTO Menuitem VALUES (3, 'Pasta', 450);
```

```
CREATE TABLE Deli[illegible]
order_ID NUMBER PRIMARY KEY,
Delivery_Status VARCHAR2(...),
FOREIGN KEY (order_ID) REFERENCES
orderTable (order_ID)
);

CREATE TABLE Menu_Item(
Item_ID NUMBER PRIMARY KEY,
Item_Name VARCHAR2(100),
Price NUMBER(10,2)
);

INSERT INTO OrderTable VALUES
(1, 101, TO_DATE('2024-02-01', 'YYYY-MM-DD'),
250.50, 'Pending');
INSERT INTO orderTable VALUES (2, 102, TO_
DATE('2024-02-02', 'YYYY-MM-DD'), 500.75,
'Paid');
INSERT INTO orderTable VALUES (3, 103, To_date
('2024-02-03', 'YYYY-MM-DD'), 150.00,
'Pending');
INSERT INTO Delivery VALUES (1, 'pending')
INSERT INTO Delivery VALUES (2, 'Delivered');
INSERT INTO Delivery VALUES (3, 'Pending');
INSERT INTO Menu-Item VALUES (1, 'Pizza', 500);
INSERT INTO Menu-Item VALUES (2, 'Burger', 300);
INSERT INTO Menu-item VALUES (3, 'Pasta', 350);
INSERT INTO Menu-item
```

```
CREATE TABLE Delivery(
order-ID Number PRIMARY KEY,
Delivery - status VARCHAR(
FOREIGN KEY (order_ID). REFERENCES
orderTable (order_ID).
      );
   CREATE TABLE Menu-Item(
    Item_ID NUMBER PRIMARY KEY,
    Item_Name VARCHAR 2(100);
     Prime Number(10,2)
      );
INSERT INTO orderTable VALUS
 (1, 101, To_DATE('2024-02-01', 'YYYY-MM-DD')
 250.50, 'Pending');
INSERT INTO orderTable VALUES ('2) 102, To-
DATE('2024-02-02', 'YYYY-MM-DD'), 400.75,
'Paid');
INSERT INTO orderTable VALUSES(3,103, To_Date
('2024-02-03', 'YYYY-MM-DD'), 150.00,
 'Pending');
INSERT INTO Delivery VALUS (1, 'Pending')
 INSERT INTO Delivery VALUES (2, 'Delivred');
INSERT INTO Delivery VALUES(3, 'Pending');
INSERT INTO Menu-Item VALUES(1,'Pizza',500);
INSERT INTO Menu-Item VALUES (2,'burger',300);
INSERT INTO Menu-Item VALUES (3,'Pasta', 450);
```

Query-4

output (cust, ID=102)

order_ID	cust_ID	order_ Date	order_ Total	Payment status
2	102	2024-02-02	400.75	paid

Query-5

output:

Item_ID	Item_name	price
1	pizza	450
2	Burger	270
3	Pasta	405

1. procedure to update payment -status

<u>step-1</u>:- Create a procedure

```
   CREATE OR REPLACE PROCEDURE update-paym
-status (
    P_order _ID IN NUMBER,
    P_new_status IN VARCHAR 2
      )AS
      BEGIN
      UPDATE order.Table
       SET Payment _status = P_ new _status
     WHERE order _ID = P_order_ID;
  COMMIT;
   DBMS_output.put _LINE ('Payment status
updated successfully for order ID ;'P_order_ID);
  END;
```

<u>Expected output</u>:-

·Procedure Created

<u>step-2</u>:- <u>Execution</u>

```
    BEGIN
     Update _payment_ Status (1, 'Paid');
    END;
```

<u>output</u>

```
Payment status updated successfully
for order ID: 1
statement processed
```

Query-2:- Function o calculate Total Revenue

step-1:- create a Function.

```
CREATE OR REPLACE FUNCTION GET_Total_
Revenue
 RETURN   NUMBER AS V_Total_Revenue number.
 BEGIN
   SELECT SUM(order_Total) INTO V_Total_Revenue
     FROM orderTable;
      Return V_Total_Revenue
       END;
```

Expected output;
Function created

Step-2:- Execution

```
GET_Total_REVENUE;
  801.25
```

Query-3:- Loop:- Mark All undelivered orders as "Delayed"

```
 DECLARE

   v-order_ID orderTable-order_ID %. TYPE;
CURSOR cur IS SELECT order_ID . FROM
DElivery WHERE Delivery_status="Pending';
 BEGIN.
    open Cur;
     Loop
      FETCH cur INTO v-order-ID;
       EXIT WHEN cur%.NOT Found;
```

```
UPDATE Delivery
  SET Delivery_status = ~~&~~ 'Delayed'
  WHERE order_ID = V-order_ID;
 DBMS_output.put_line ('order' ||V-order_ID||
                              'marked as Delayed');
   END LOOP;
   close cur;
      COMMIT;
       END;
```

Expected output;

1(rows) updated

Query-4:- procedure to get order Details by customer ID

Step-1:- Create a procedure

```
  CREATE OR REPLACE PROCEDURE Get_custo-
mer_order .
       (P-cust_ID . IN NUMBER ') - AS
  BEGIN
     For order_rec IN(SELECT order_ID,order_Dat
order_Total ,payment_status . FROM order
.Table
  WHERE cust=ID = P.cust_ID) LOOP
 DBMS_output.put_LINE ('order_ID;'|| 'order_rec
 order_ID.|| ,Date;'||order_rec-order-total,
 Total ;||.order_rec.order_Total ;status
 :''order_rec.payment_status);
```

Expected output

```
orderID =1, Date: 2024-02-01, Total =250.5,
  Payment : paid
statement processed
```

Query-5 :- procedure to Apply Discount on Menu Items

step-1 :- Create a procedure

```
CREATE OR REPLACE PROCEDURE Apply_Discount
 (discount_percent IN NUMBER )
IS BEGIN
  UPDATE Menu_Item
 set price= price_(price* discount_percent /100);
  COMMIT
  DBMS_output . put_LINE ('Discount Applied'||
                     "discount_percent" %);
  END;
```

VEL TECH - CSE	
EX NO.	
PERFORMANCE (5)	
RESULT AND ANALYSIS (3)	
VIVA VOCE (3)	
RECORD (4)	
TOTAL (15)	
SIGN WITH DATE	