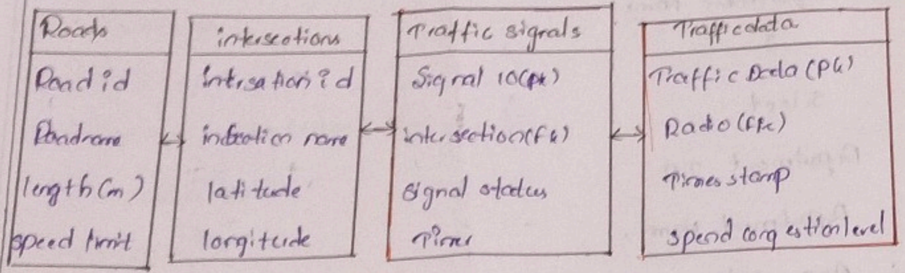
Question 4:-

ER Diagram Question, Traffic flow management system;

Scenario,

you are tasked with designing an Entity - Relationship(ER) diagram for a traffic flow management system(TFMS)

Task:- entity identification and Attributes.



Task-2

Relation ship Modeling:-

Relationships:-

1. Roads to intersection:-

* one Road op can connect to multiple intersections
* An intersection can host multiple traffic data entites

2. Intersection to traffic signals:-

* One intersection can host multiple traffic data entities

cardinality and optionality:-

1. Roads to intersection:

M. Polireddy

* one road can connect to zero or more intersections
* one intersection can connect to one or a more roads

2. Intersections to trafic Signals:-

* one intersection can have zero or more traffic signals
* one traffic signal must be associated to one a more roads intersection.

3. Roads to traffic data:-

* One load can have zero or more traffic data entities
* One traffic data entry must be associated with one road.

Task-4:- Justification and normalization.

1. Scalability

* The design allows for easy addition of new roads, intersections traffic signals, and trafficdata entities without modifying the size.

*Road time Data processing:-

* Real-time data integration is fanilitated by the trafic data.

3. Efficient Traffic Management

* The clear separation of entities

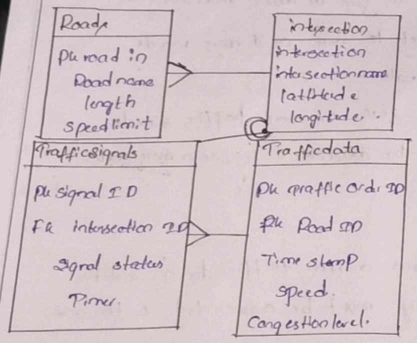
Deliverables:-

ER Diagram; provided above in plan text format

entities Definition : listed in task.

Relationship Description

Task 8:- ER Diagram Design:-



Question:- 2:-

Question: Top3 Department with Highest Average Salary

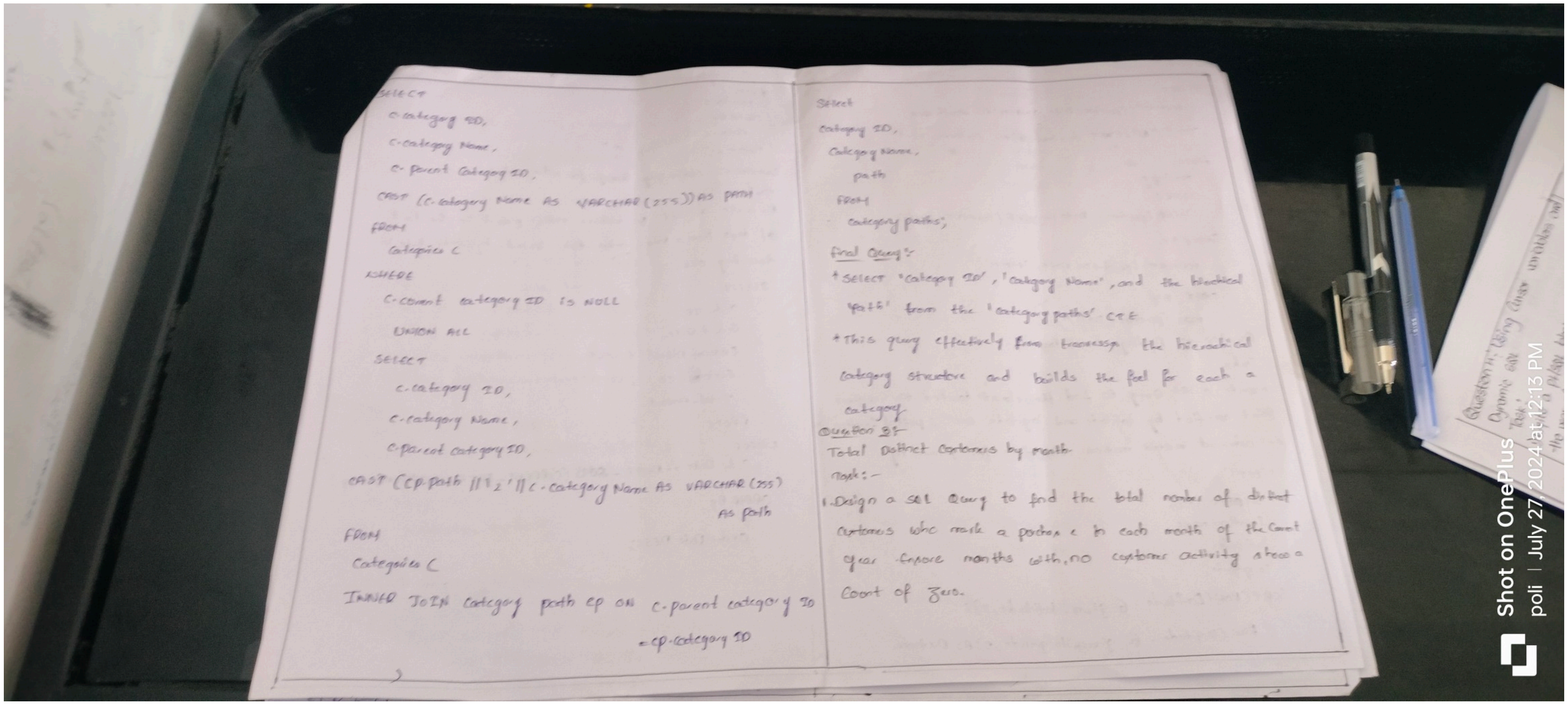
SQL Query

```
with Avg salaries As {
  Select
    d.department ID,
    D.Department Name,
    Avg (e.salary) As Avg salary
FROM
  Department i d
Group By
  d.department ID,
  d.department ID,
  }
Select
  Department ID,
  Department Name.
  )
  Select
  Department ID
  Department Name,
  Avg salary
  FROM
     Avg salaries
     ORDER BY
  Avg salary DESC NULLS LAST
LIMIT 3;
```

Question 2:- Retriving Hierarchical Paths SQL Query

```
with RECURSIVE category path As (
SELECT
```

Question 4:- Using Cursor
Dynamic SQL
Task:
1) Write a PL/SQL block
the user of coursor
dynamic SQL Declar
variables for que
ID, firstname, a
DECLARE
TYPE em
IS REC
V-EMP-
V-Sal



```
SELECT
  c.category ID,
  c.category Name,
  c.parent Category ID,
CAST (c.category Name AS VARCHAR(255)) AS PATH
FROM
  Categories C
WHERE
  C.coment category ID is NULL
   UNION ALL
SELECT
   c.category ID,
  c.category Name,
  c.parent category ID,
CAST (cp.path ||'2'|| c.category Name AS VARCHAR(255)
                                        AS path
FROM
 Categories C
INNER JOIN category path cp ON c.parent category ID
                        = cp.category ID
)
```

```
Select
Category ID,
 Category Name,
   path
 FROM
  category paths;
```

<u>final Query</u>:-

* select 'category ID', 'category Name', and the hirachical 'path' from the 'category paths' CTE
* This query effectively from traversp the hierachical category structure and builds the fool for each a category

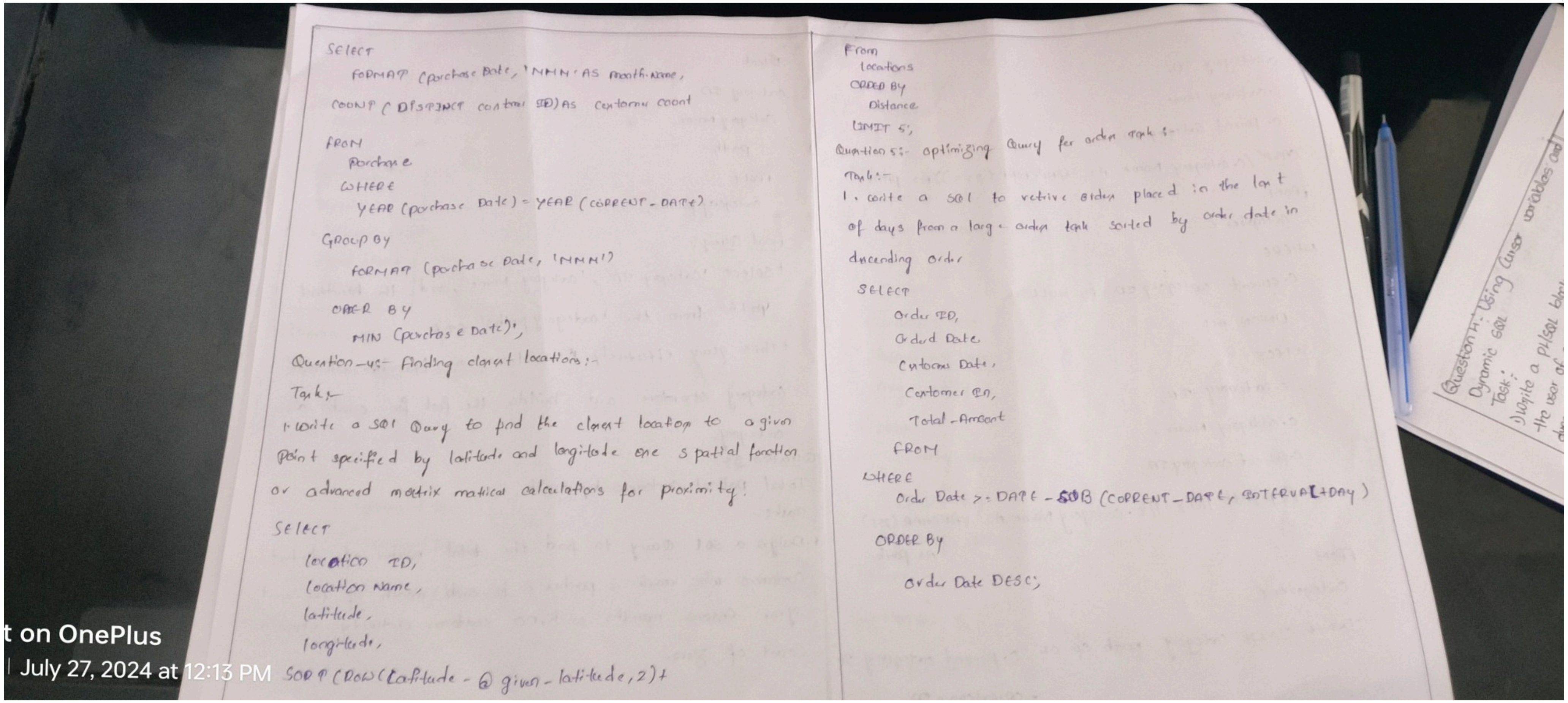
<u>Question 3</u>:-

Total Distinct customers by month.

Task:-

1. Design a SQL Query to find the total number of distinct customers who mark a porchase in each month of the current year ensure months with no customer activity show a count of zero.

Question 4: Using [illegible] variables and
Dynamic SQL
Task:



```
SElECT
   FORMAT (purchase Date, 'MMM' AS month-Name,
COUNT ( DISTINCT customer ID) AS customer count

FROM
   purchase
  WHERE
    YEAR (purchase Date) = YEAR (CURRENT-DATE)
GROUP BY
   FORMAT (purchase Date, 'MMM')
  ORDER BY
   MIN (purchase Date)';
```

Question-4:- Finding closest locations:-

Task:-

1. write a SQL Query to find the closest location to a given point specified by latitude and longitude one spatial function or advanced mathematical calculations for proximity!

```
SElECT
   location ID,
   location Name,
   latitude,
   longitude,
 SQRT (POW(Latitude - @ given-latitude, 2)+
From
  locations
 ORDER BY
   Distance
 LIMIT 5;
```

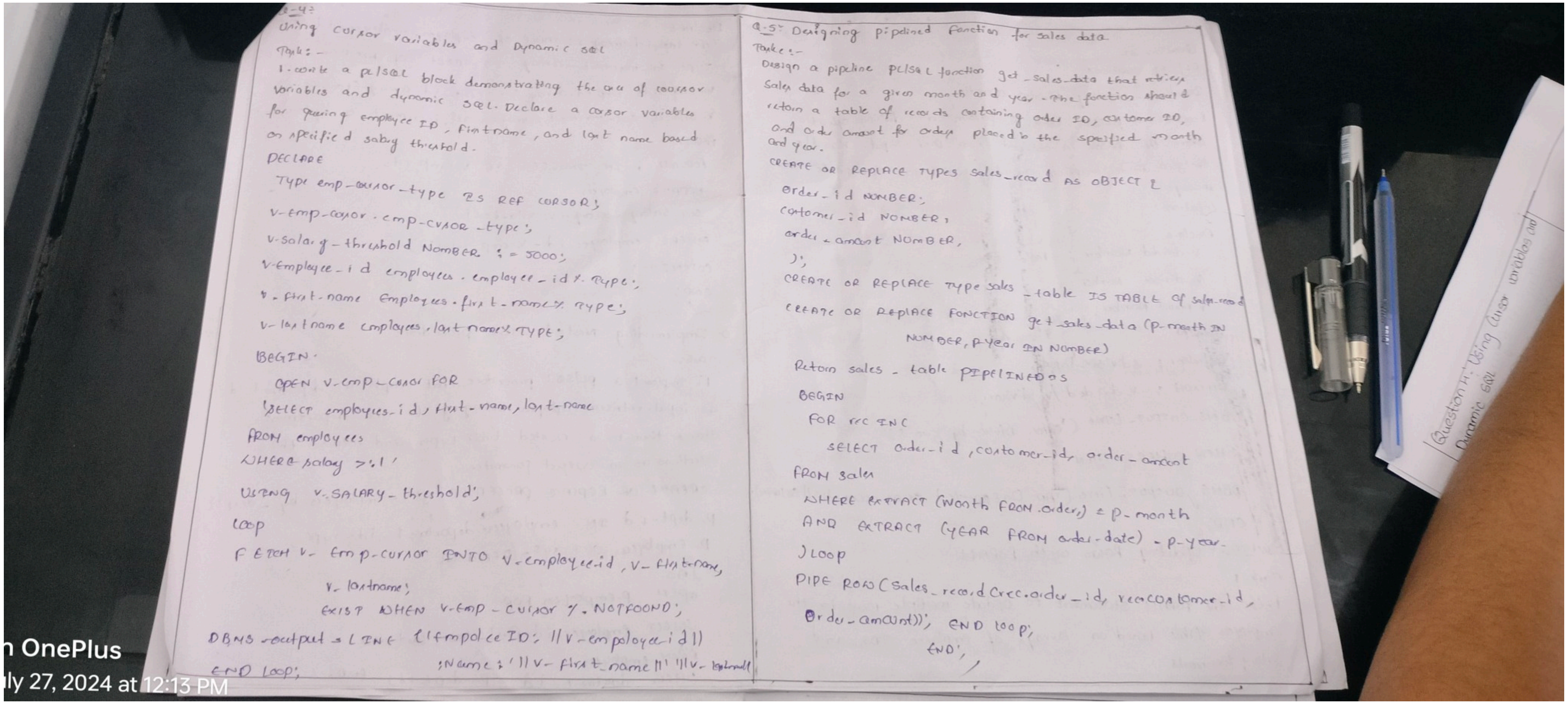
Question 5:- optimizing Query for order Task :-

Task:-

1. write a SQL to retrive orders placed in the last of days from a large order task sorted by order date in descending order

```
 SELECT
     Order ID,
     Orderd Date
     Customer Date,
      Customer ID,
      Total-Amount
     FROM
 WHERE
    Order Date >= DATE - SUB (CURRENT-DATE, INTERVAL +DAY )
  ORDER BY
     Order Date DESC;
```

Question 4:- Using Cursor variables and
Dynamic SQL
Task :
1) Write a PL/SQL bl
the user of



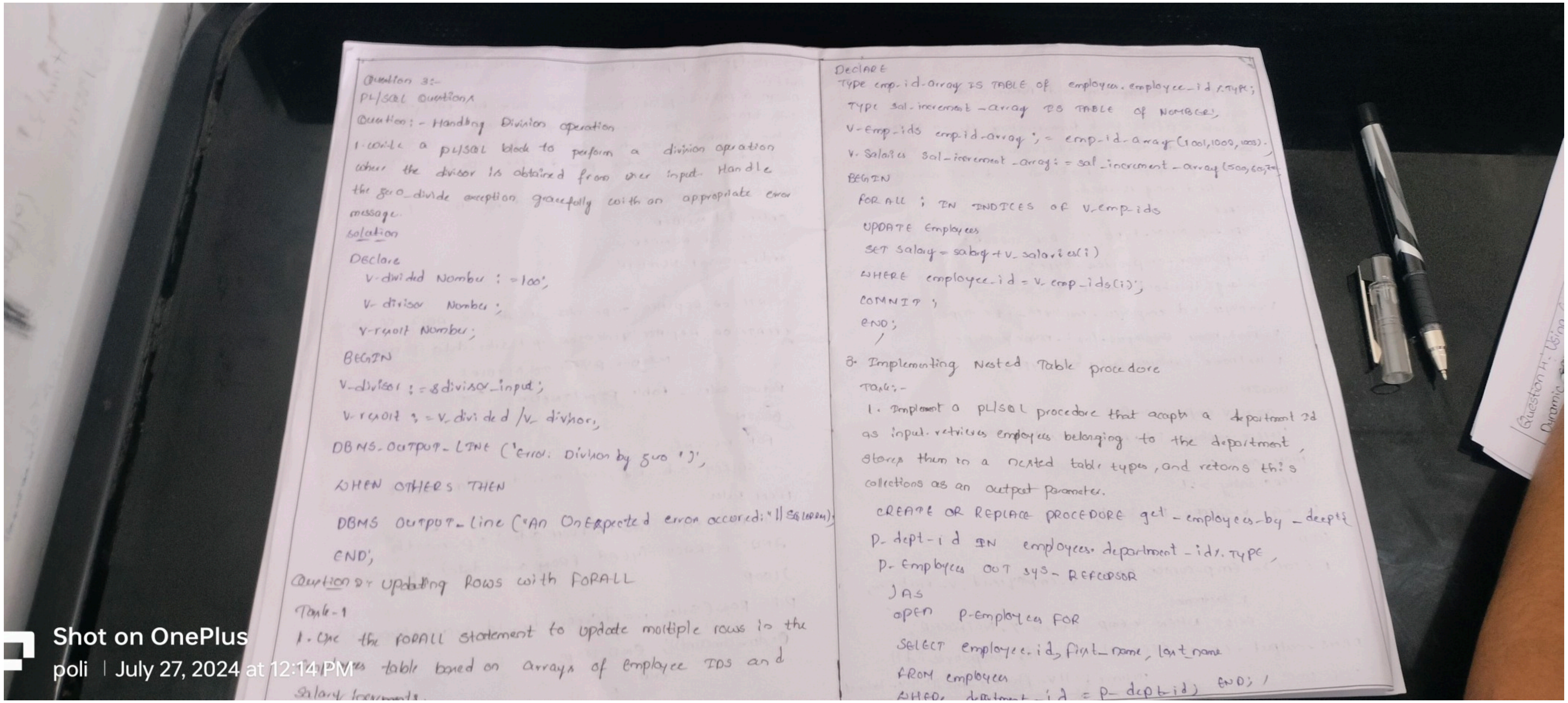
Q-4:-
Using cursor variables and Dynamic SQL
Task:-
1. write a PL/SQL block demonstrating the use of cursor variables and dynamic SQL. Declare a cursor variables for querying employee ID, first name, and last name based on specified salary threshold.

```
DECLARE
 Type emp-cursor-type IS Ref CURSOR;
 v-emp-cursor . emp-cursor -type;
 v-salary-threshold NUMBER := 5000;
 v-Employee-id employees . employee-id %. TYPE;
 v-first-name Employees . first-name% TYPE;
 v-last name employees . last name% TYPE;
BEGIN.
 OPEN v-emp-cursor FOR
 'SELECT employees-id, first-name, last-name
FROM employees
WHERE salary >:1'
USING v-SALARY-threshold;
loop
FETCH v- emp-cursor INTO v-employee-id , v- first-name,
          v- lastname;
          EXIST WHEN v-emp - cursor %. NOTFOUND;
DBMS-output . LINE ('Empolee ID:' || v-empoloyee-id ||
END Loop;              ;Name:' || v- first_name ||' '|| v- lastname||
```

Q-5:- Designing pipelined function for sales data
Task:-
Design a pipeline PL/SQL function get_sales-data that retrieves sales data for a given month and year. The function should return a table of records containing order ID, customer ID, and order amount for orders placed in the specified month and year.

```
CREATE OR REPLACE TYPES sales_record AS OBJECT (
 order-id NUMBER;
 customer-id NUMBER,
 order - amount NUMBER,
 );
CREATE OR REPLACE Type sales -table IS TABLE Of sales-record
CREATE OR REPLACE FUNCTION get_sales_data (p-month IN
              NUMBER, P-Year IN NUMBER)
 Return sales - table PIPELINED IS
 BEGIN
  FOR rec IN (
   SELECT order-id , customer-id, order - amount
 FROM sales
  WHERE EXTRACT (Month FROM .order,) = p- month
  AND EXTRACT (YEAR FROM order-date) = p-year.
 ) Loop
 PIPE ROW (Sales_record (rec.order_id, rec.customer-id,
  Order-amount)); END loop;
            END;
```

n OnePlus
ly 27, 2024 at 12:13 PM



Question 3:-
PL/SQL Questions
Question:- Handling Division operation
1. write a PL/SQL block to perform a division operation where the divisor is obtained from user input. Handle the zero_divide exception gracefully with an appropriate error message.
solution

```
Declare
  v-divided Number := 100;
  v-divisor Number;
  v-result Number;
BEGIN
v-divisor := &divisor_input;
v-result := v-divided / v-divisor;
DBMS-OUTPUT-LINE ('Error: Division by zero');
WHEN OTHERS THEN
DBMS OUTPUT-Line ("An Unexpected error occured:" || SQLERRM);
END;
```

Question 2: Updating Rows with FORALL
Task-1
1. Use the FORALL statement to update multiple rows in the employees table based on arrays of employee IDs and salary increments.

```
DeclARE
TYPE emp-id-array IS TABLE of employees.employee_id%TYPE;
 TYPE sal-increment-array IS TABLE of NUMBER;
 v-emp-ids emp-id-array := emp-id-array (1001,1002,1003).
 v. Salaries sal-increment-array := sal_increment-array (500,600,700);
 BEGIN
  FOR ALL i IN INDICES of v-emp-ids
  UPDATE Employees
  SET salary = salary + v-salaries(i)
  WHERE employee-id = v-emp-ids(i);
  COMMIT;
  END;
  /
```

3. Implementing Nested Table procedure
Task:-
1. Implement a PL/SQL procedure that accepts a department id as input. retrieves employees belonging to the department, stores them in a nested table types, and returns this collections as an output parameter.

```
  CREATE OR REPLACE PROCEDURE get-employees-by-dept(
  P-dept-id IN employees.department-id%TYPE,
  P-Employees OUT SYS-REFCURSOR
   )AS
   open p-employees FOR
   SELECT employee.id, first_name, last_name
   FROM employees
   WHERE department-id = p-dept-id)   END; /
```