ASSIGNMENT 3

Name of the Student : Challa. Ravi teja

Register Number : 192325065

Name of the course : Database Management System.

Course code : CSA0563

Date of Submission : 27-07-24

Question - 1

ER - Diagram Question :- Traffic flow Management System.

Task - 1 : Entity Identification and Attributes.

Road (Root)	Intersection	Traffic data	Traffic Signals
Road (PK)	Intersection id (PK)	Signal id (PK)	Traffic Data ID (PK)
Road Name	Intersection Name	intersection id (FK)	Radio (FK)
length (m)	lattitude	Signal status	Timestamp
Speed limit (Km)	longitude	Timer	Speed congestional leve

Task - 2 : Relational Modelling.

Relationships.

1. Roads to intersection.

→ An intersection can be continued by Multiple loads

→ One Road up can connect to Multiple intersection.

2. Intersection to traffic Signals.

→ one Intersection can not Multiple traffic data entities.

3. Cardinality and optionality:

1. Roads to traffic data

→ one road can have zero or more data Entities

→ one traffic data entry Must be associated with one road.

2. Roads to intersection
→ one road can connect to zero or more intersections
→ one Intersection can connect to one or more roads
3. Intersection to trafficSignals
→ one Intersection can have zero (or) more traffic Signals
→ one traffic signal Must be associated with one Intersection.

Task-4 : Satistication and Normalization.

1. Scalability
* The design allows for any addition of new roads Intersection traffic status signals and traffic data entities modifying the size.

2. Real time Data processing.
* Real time traffic data Integration is faclitated by Traffic data.

3. Efficient Traffic Management
* The clear seperate of Entities.

Delivarables.
ER Diagram provided above in plan text format
Entity Defination: listed in Task 1
Relation ship Description.
Justification Document.

Task-5 :

Table 3 : ER Diagram Design.

Roads
(PK) Road ID
Road Name
Length
Speed limit

Intersection.
Intersection ID
Intersection Name
lattitude
longitude

Traffic Signal
(PK) Signal ID
(PK) Intersection ID
Signal status
Timer.

Traffic Data.
(PK) Traffic Data ID
(PK) Road ID
Timestamp.
Speed congestion level.

Question-2

Task-1 : Top 3 Departments with Highest Average Salary.

Sql query.

```
with AvgSalaries As {
    Select
      d.department ID
      d. department Name.
      AVG.(e.Salary) As Avg salary
      FROM Department.
      left Join employees e.d.depatrment = e.Department ID
```

```
GROUP BY(
    d.department ID,
    d.department Name.
   )
 SELECT
    Department ID
    Department Name;
 SELECT
    Department ID
    Department Name
    Avg Salary.
FROM
   Avg Salaries
ORDER BY
   Avg Salary DESCNULLSLAct
LIMIT 3;
```

<u>Task-2</u>: Retreiving Hierarchial categorypaths setquery

```
WITH RECURSIVE category path ASC
Select
C. Category ID
C. category Name,
C. parent category ID,
CAST (C.category Name As VARCHAR(255) Aspath.
FROM
   categories C
WHERE
   C.parent category ID is NULL
   UNIONALL
```

```
SELECT
   C.category ID,
   C.category Name
   C.parent category ID,
   CAST (Cppath || || category Name As VARCHAR(255) As path
FROM
   categories C
   INNER JOIN category path cp or C.parent category
                 ID = cp.category ID
  )
Select
   category ID,
   category Name,
   path
FROM
   category paths;
```

Final query:

* Select category ID category Name and the hierarichal 'path' The category paths CTE

* This query effectively Traverse the hieraichal category structure and build fuel for each category.

3. Total Distinte customers by month.

```
SELECT
   Date_Format (order date, (%Y - %M)
As month name 1
   COUNT (Distint Customer ID) As
   CUSTOMER COUNT
```

```
FROM
    orders
WHERE
    order date ≥ Date_sub(curdate(), interval, year)
GROUP by.
    Month Name
GROUP by
    Month Name;
```

4. Finding closest locations:

```
SELECT
   location ID,
   location Name,
   latitude,
   longitude,
   (637* AS os (Radius (37.7747)* cos
(Radius (latitude))Alos (Radius (-122.4174)
Radius (longitude) is in (Radius (latitude)))
  As distance.
```

5. Optimizing Query for order table.

```
SELECT * from orders
Where order date ≥ Date Sub (curdate(), Interval day)
Order By
Order Date Dase;
```

Question-3

Task-1: Handling Division operation:

```
  Declare
  dividend number; =100;
      divisor number;
      result number;
 Begin
    divisor; = &divisor;
 Begin
    result: = dividend/divisior;
   DBMS-output-line ('Result; '// result);
 Exception
   is not allowed');
 End;
 End;
```

Task-2: Updating rows with for All.

```
 Declare
  Emp-ids DBMS_SQL. number-Table: =
 DBMS-SqL. number_Table (101, 102, 103);
 Salary-line DBMS-SqL number. Table; =
 DBMS-SqL number_Table (1000, 2000, 3000);
Begin
 .For All in emp-ids. first_emp-ids_LAST
update Employees.
  Set Salary = Salary * Salary-incc(i)
WHERE Employee ID = emp-ids(i);
  END;
```

Task-3: implementing Nested_Table procedure.

```
Create Type emp_table_type is
Table of employees % Row Type;
Create (or) replace procedure
get_department_employee(
   D-department_id IN number'
   D-employees out emp-table-type)
IS BEGIN
SELECT *
Bulk collect INTO D-employees
FROM Employees.
WHERE Department ID=
  D-department-ID;
END;
```

Task-4:

```
DECLARE
  Type emp_cursor is REF cursor;
  emp-ref emp_cursor;
  emp-id
Employees Employee ID % TYPE;
   first Name.
Employees first Name % TVPE;
Salary_Thresold Number := 5000;
Sql-stmt varchar 2(500);
Begin
  Sql_stmt:='SELECT employee id, first Name, last Name
FROM Employees.
WHERE Salary > : salary;
Open emp-ref for Sql-stmt using salary.
Thresold;
loop
Fetch emp-ref into emp_id; Firstname
last Name;
Exit when emp-ref %. NOT FOUND;
DBMS - output. put_line (emp_id || "-" || first_name||" ||
last name;
END name;
close emp.ref;
END.
```

Task-5: Designing pipelined function for sales Data.

```
Create (or) Replace function get-sales-data (
P_month_number
D-year-number)
Return Sales_data-type
Pipelined is
Cursor Sales_Cursor IS
SELECT order_ID, customer ID, order Amount
FROM orders
WHERE extract (Month from order date) = P_month.
FROM orders
WHERE extract (year from order data) - P-year
Sales record
```