ASSIGNMENT-3

K. DHANUSH REDDY

192324087

CSA0653

27-07-2024

Question 1:-

ER Diagram :- Traffic Flow Managment System :

Scenario :-

You are tasked with designed an Entity-Relationship(ER) diagram for a Traffic Flow Managment System (TFMS)

Task 1 :- Entity identification and Attributes:

Based on the scenario provided for the Traffic Flow management system (TFMS), we can identify the following entities along with their attributes:

Roads	Intersections	Traffic signals	Traffic Data	Relationships
Road ID (PK) (unique identifier)	Intersection ID(PK) (unique identifier)	signal ID(PK) (unique identifier)	Traffic Data ID(PK) (unique traffic data)	Roads
Road Name (Name of the road)	Intersection Name (name of intersection)	Intersection(FK) (associated intersection)	Road ID (FK) (associated road)	Intersections
Length (length of the road meters)	Latitude (Geographic latitude)	Signal states (current status)	Timestamp (date was collected)	Traffic signals
Speed Limit (speed limit in km/h)	Longitude (Geographic longitude)	timer (countdown timer)	congestion level (low, medium, high)	Traffic Data

⇒ Task 2:- Relationship Modeling

1. Roads to intersections

* A road can connect to multiple intersections, but an intersection is associated with at least two roads

Relationship TYPE : One-to-many from Roads to intersections

Cardinality: One Road can connect to many intersections, each intersection connects to multiple Roads

2. Intersections to Traffic signals :

* An intersection can have one or more traffic signals

Relationship TYPE : One-to-Many from intersections to traffic signals

cardinality : one intersection can have many Traffic signals; each traffic signal is associated with one intersection.

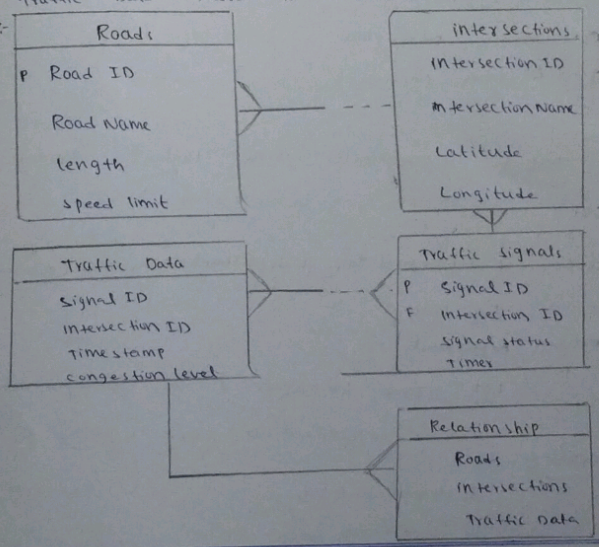
3. Roads TO Traffic Data :-

* A road can have multiple traffic data records collected over time

Relationship TYPE : one-to-many from Roads to Traffic Data

cardinality : one Road can have many traffic Data records, each Traffic Data record is associated with one Road

Task 3:-



Task 4 :- Justification and Normalization :-

1. Scalability : The design allows for easy addition of new roads, intersections, traffic signals and traffic data entries without modifying the traffic

2. Real-Time Data processing :-

A Real time traffic data integration is facilitated Traffic data

3. Efficient Traffic Managment :-

the clear seperation of entities

Deliverables :-

ER diagram : Provided above in plain text format entity

Definition : listed in task 1

Relationship Descriptions

Justification Document

2.

1. TOP 3 Departments with Highest Average salary

SQL

```
SELECT d.department ID, d.department name, AVG (e.salary) As
        Avg salary
FROM    Departments.d
        LEFT JOIN EMployees e IN d.department ID
            e Department ID
GROUP BY
        d.department ID,
        d.department Name
        SELECT Department ID,
               Department Name
               Avg salaries
        FROM   Avg salaries
        ORDER BY Avg salary DESC NULLS
               LAST LIMIT 3;
```

2. Retreiving Hierarchical category paths

SQL

```
WITH RECURSIVE category path As
SELECT c. category ID,
       c. category Name,
       c. parent category ID,
LAST (c. catogery name As VARCHAR (255)) As path
FROM
    categories c
```

Final Query :-

* Select 'category ID', 'category Name', and the hierarchical 'path' from the 'category paths' CTE

* This query effectively tranverses the hierarchical category structure and builds the fuel for each category.

Total Distintic customers by Month

```
SELECT
    Date_Format (order date, ('%Y - %m'))
As  month name,
    COUNT (Distinct CUSTOMER ID) As
    CUSTOMER COUNT
FROM
    Orders
Where
    Order date ≥ Date - sub (curdate(), Internal 1 year)
GROUP BY  Month Name
GROUP BY
    Month Name;
```

4. Finding closest locations :-

```
SELECT  location ID, location Name,
        latitude, Longitude,
        (637 * As (Radians (37.7749) x cos
    (Radians (catitude)) As (Radians (-122.4194)-
    Radians (longitude) As In (Radians (latitude)))
    As distance
```

5. Optimizing Query for order table :-

```
SELECT *
FROM  Orders
WHERE  order date >= Date - sub (cur date(),
    Interval day)
    ORDER BY  order Date DESC;
```

3. Handling Division operation :-

```
Declare
dividend  number = 100;
    divisor  Number )
    result  Number;
BEGIN  divisor = & divisor;
Begin
    result := dividend / divisor;
    DBMS - output - line ('Result' || Result);
Exeption
    is not allowed );
END;
END;
```

2. Updating rows with For ALL :-

```
Declare
    emp-id's  DBMS - SQL  Number - Table;
```

DBMS - SQL number - Table (101, 102, 103);

Salary - like DBMS - SQL - number - Table :=

DBMS - SQL Number - Table (1000, 2000, 3000);

Begin

for ALL in emp-ids . First emp-ids | AST

Update Employees

set salary = salary * salary - incc(i)

WHERE employee ID = emp- ids (i);

END;

3. IMplementing Nested Table procedure :-

create TYpe emp - table - type is

Table of employees %. Rows TYPE;

Create (or) replace procedure

get - department - employee (

D - department - id IN number.

D - employee out emp-table-type)

IS BEGIN

5. Designing pipecined function for sales Data:-

Create (or) Replace function get-sales-data(

P - month - Number;

P - 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

AND Extract (year from order date) = P-year

pipe row (sales . record);

END loop;