

(Given below is interaction with Deepseek LLM of version deepseek-r1:1.5b)

Group members- Hrithik Raj, Akshit Sharma, Aditya Rawat, Dhimant Kaul

Microsoft Windows [Version 10.0.22631.5039]

(c) Microsoft Corporation. All rights reserved.

C:\Users\lenovo>ollama run deepseek-r1:1.5b

>>> Table Name: Disaster

...

... Disaster_ID (PK, SERIAL) – Unique identifier for each disaster.

...

... Disaster_Type (VARCHAR(100), NOT NULL) – Type of disaster (e.g., earthquake, flood, fire).

...

... Date (DATE, NOT NULL) – The date when the disaster occurred.

...

... Location (TEXT, NOT NULL) – Location of the disaster.

...

... Severity_Level (INT, CHECK 1-10) – Scale of severity from 1 (low) to 10 (high).

...

... Description (TEXT, NULL) – Additional details about the disaster.

...

... Status (VARCHAR(20), NOT NULL, CHECK 'Ongoing' OR 'Not Ongoing') – Whether the disaster is active or not.

...

... 2. Affected Area Entity

... Table Name: Affected_Area

...

... Disaster_ID (FK, INT, REFERENCES Disaster ON DELETE CASCADE) – Links to the related disaster.

...

... Area_Name (VARCHAR(255), NOT NULL) – Name of the affected area.

...

... Population_Affected (INT, CHECK >= 0, NULL) – Number of people affected.

...

... Damage_Estimation (NUMERIC, CHECK >= 0, NULL) – Estimated financial damage.

...

... Primary Key: (Disaster_ID, Area_Name)

...

... 3. Rescue Team Entity

... Table Name: Rescue_Team

...

... Team_ID (PK, SERIAL) – Unique identifier for each team.

...

... Team_Name (VARCHAR(100), NOT NULL) – Name of the rescue team.

...
... Specialization (TEXT, NOT NULL) – Expertise area (e.g., medical, firefighting, logistics).

...
... Contact_Number (VARCHAR(15), NOT NULL) – Contact number of the team.

...
... Status (VARCHAR(50), CHECK 'Active' OR 'Standby') – Availability status of the team.

...
... 4. Volunteer Entity

... Table Name: Volunteer

...
... Volunteer_ID (PK, SERIAL) – Unique identifier for each volunteer.

...
... Name (VARCHAR(100), NOT NULL) – Full name of the volunteer.

...
... Skills (TEXT, NOT NULL) – List of skills or expertise.

...
... Contact_Number (VARCHAR(15), NOT NULL) – Contact information.

...
... Availability (BOOLEAN, NOT NULL) – Indicates if the volunteer is available for duty.

...
... 5. Volunteer-Rescue Team Association

... Table Name: Volunteer_Rescue_Team

...
... Volunteer_ID (FK, INT, REFERENCES Volunteer ON DELETE CASCADE) – Volunteer assigned.

...
... Team_ID (FK, INT, REFERENCES Rescue_Team ON DELETE CASCADE) – Team assigned.

...
... Primary Key: (Volunteer_ID, Team_ID)

...
... Trigger:

...
... Prevents assigning an unavailable volunteer.

...
... 6. Disaster-Rescue Team Assignment

... Table Name: Disaster_Assigned_Teams

...
... Disaster_ID (FK, INT, REFERENCES Disaster ON DELETE CASCADE) – Disaster being handled.

...
... Team_ID (FK, INT, REFERENCES Rescue_Team ON DELETE CASCADE) – Assigned rescue team.

...

... Primary Key: (Disaster_ID, Team_ID)

...

... Trigger:

...

... Ensures only "Ongoing" disasters get assigned teams.

...

... Ensures only "Active" teams can be assigned.

...

... 7. Relief Center Entity

... Table Name: Relief_Center

...

... Relief_Center_ID (PK, SERIAL) – Unique ID for the relief center.

...

... Name (VARCHAR(255), NOT NULL) – Name of the relief center.

...

... Location (TEXT, NOT NULL) – Address or area of the center.

...

... Capacity (INT, CHECK > 0, NOT NULL) – Maximum people it can accommodate.

...

... Available_Resources (TEXT, NULL) – List of available resources.

...

... 8. Victim Entity

... Table Name: Victim

...

... Victim_ID (PK, SERIAL) – Unique identifier for each victim.

...

... Name (VARCHAR(100), NOT NULL) – Full name of the victim.

...

... Age (INT, CHECK >= 0, NULL) – Age of the victim.

...

... Gender (VARCHAR(10), CHECK 'Male' OR 'Female' OR 'Other', NULL) – Gender of the victim.

...

... Contact_Details (TEXT, NULL) – Contact info or emergency contact.

...

... Status (VARCHAR(50), CHECK 'Rescued' OR 'Safe' OR 'Injured' OR 'Missing' OR 'Deceased', NOT NULL) – Condition status

... .

...

... Disaster_ID (FK, INT, REFERENCES Disaster ON DELETE SET NULL) – Associated disaster.

...

... Relief_Center_ID (FK, INT, REFERENCES Relief_Center ON DELETE SET NULL) – Assigned relief center.

...

... Constraint:

...

... If Status = 'Deceased' or 'Missing', then Relief_Center_ID must be NULL.

...

... 9. Donation Entity

... Table Name: Donation

...

... Donation_ID (PK, SERIAL) – Unique ID of the donation.

...

... Donor_Name (VARCHAR(255), NOT NULL) – Name of the donor.

...

... Amount (NUMERIC, CHECK > 0, NOT NULL) – Donation amount.

...

... Date (DATE, NOT NULL) – Date of donation.

...

... Purpose (TEXT, NOT NULL) – Purpose or description of the donation.

...

... Relief_Center_ID (FK, INT, REFERENCES Relief_Center ON DELETE SET NULL) – Associated relief center.

...

... 10. Government Agency Entity

... Table Name: Government_Agency

...

... Agency_ID (PK, SERIAL) – Unique identifier.

...

... Agency_Name (VARCHAR(255), NOT NULL) – Name of the agency.

...

... Contact_Number (VARCHAR(15), NOT NULL) – Contact info.

...

... Responsibilities (TEXT, NULL) – Description of duties.

...

... given is the database schemas of our project natural disaster management system

... following this given below are some questions of which you have generate SQL queries:

... 1. List volunteers assigned to rescue teams despite being unavailable

... 2. Find disasters with total damage exceeding \$1M across all affected areas

... 3. List relief centers at full capacity with empty resources

... 4. Find rescue teams managed by multiple government agencies

... 5. Show deceased victims linked to a relief center

... 6. List volunteers with "First Aid" skills not in any team

... 7. Count active teams assigned to ongoing disasters

... 8. Find donors contributing >\$5000 across all purposes

... 9. Disasters with no assigned rescue teams

... 10. Affected areas with above-average population impact

<think>

Okay, so I'm trying to figure out how to answer these 10 questions based on the given database schema and tables.

Let me take each question one by one and think through what needs to be done.

Starting with question 1: "List volunteers assigned to rescue teams despite being unavailable."

Hmm, so I need to

find volunteers whose team is assigned, but their status is not available (i.e., 'Standby'). How do I do that?

Well, looking at the tables, there's a Volunteer table and a Rescue_Team table. The association is through Team_ID

in both. So maybe I can join these two tables on Team_ID, then filter where the volunteer's status is 'Standby'.

That should give me all volunteers assigned to teams that are not available.

Question 2: "Find disasters with total damage exceeding \$1M across all affected areas." The table for this seems

to be Disaster and its associated Affected_Area. I need the sum of Damage_Estimation from each Affected_Area. So,

group by Disaster_ID and calculate a SUM on Damage_Estimation. If that sum is more than 1,000,000, then include it.

Question 3: "List relief centers at full capacity with empty resources." Full capacity means

Capacity equals the maximum possible, which might be defined in the database as

something like not null or highest value. But wait, I should check if the table has a Max function.

If so, filter for Capacity = Max(Capacity). For empty resources, I think

that would mean Available_Resources is NULL. So combine those two conditions.

Question 4: "Find rescue teams managed by multiple government agencies." Each Rescue

Team has a Volunteer_ID and belongs to a Government Agency via Volunteer_ID. So

maybe join the Rescue_Team and Government_Agency tables on Volunteer_ID, then group by Disaster_ID (since each team is associated with one disaster). Then count how many

agencies are there for each team. If the count is more than 1, include it.

Question 5: "Show deceased victims linked to a relief center." Look at the Victim table and see if their Status is 'Deceased'. Also, check if their Relief_Center_ID matches

the Donation table's RELI_Center_ID. So I need to join these tables on both sides for the relation.

Question 6: "List volunteers with 'First Aid' skills not in any team." The Volunteer entity has Skills and Availability. But to get all the skills, maybe I need a way to

list unique skills across all entries. Then filter out those without 'First Aid'. Alternatively, if each volunteer's skills are a set, perhaps using a U function or

something similar. Wait, but since it's a TEXT field, maybe using an OR operator on all skills would give me all the skills used by the volunteers. Then I can exclude any that don't have 'First Aid'.

Question 7: "Count active teams assigned to ongoing disasters." Looking at the tables, Active is a constraint in the Rescue_Team table, so each team has Status = 'Active'. So for each active team (Volunteer_ID exists), check if their Assigned status is 'Ongoing' (either from Assignment Table or Discount Constraints). Maybe using a subquery where we find all teams and then join with their assigned status. Alternatively, use an outer query to select distinct Team_ID from Rescue_Team where Status = 'Active' and count them.

Question 8: "Find donors contributing >\$5000 across all purposes." The Donation table has Donor_Name, Amount (as a numeric), and Purpose. So I need the sum of all Amounts in each purpose row. Group by Purpose and calculate SUM(Amount). If that's greater than \$5000, include it.

Question 9: "Disasters with no assigned rescue teams." From the tables, we have Assignment Table (volunteers) but I don't see a direct table for assignments. Alternatively, checking if any Rescue_Team has an empty Status, which would be 'Standby'. So in the Rescue_Team table, check for any Status where Status is not null and equals 'Standby'.

Question 10: "Affected areas with above-average population impact." The Average function might be used here. Then filter areas where their Damage exceeds this average.

Wait, let me make sure I didn't miss any table names or check for all possible joins. For example, question 5 uses both Victim and Donation tables. Also, when counting Active teams assigned to Ongoing disasters, perhaps using a SELECT DISTINCT on Team_ID from Rescue_Team where Status = 'Active' then count.

I think that's the approach, but I might have missed some nuances or constraints in the database schema.

</think>

To address each query based on the provided database schemas and tables, here are the SQL responses:

```
1. **List volunteers assigned to rescue teams despite being unavailable:**  
``sql  
SELECT v.*  
FROM Volunteer v  
JOIN Volunteer_Rescue_Team vrt ON v.Volunteer_ID = vrt.Volunteer_ID  
WHERE v.Availability = FALSE;
```

...

2. **Find disasters with total damage exceeding \$1M across all affected areas.**

```
```sql
SELECT d.Disaster_ID, SUM(a.Summation) AS Total_Damage
FROM disaster d
JOIN Affected_Area a ON d.Affected_Area_ID = a.ID
GROUP BY d.Disaster_ID
HAVING SUM(a.Summation) > 1000000;
```
```

3. **List relief centers at full capacity with empty resources.**

```
```sql
SELECT *
FROM Relief_Center
WHERE Capacity = 0
AND (Available_Resources = " OR Available_Resources IS NULL);
```
```

4. **Find rescue teams managed by multiple government agencies.**

```
```sql
SELECT count(g.Government_Agency_ID) AS Number_of_Agencies
FROM Rescue_Team r
JOIN Government_Agency g ON r.Volunteer_ID = g.Agency_ID
GROUP BY r.Disaster_ID, r.Volunteer_ID;
```
```

5. **Show deceased victims linked to a relief center.**

```
```sql
SELECT *
FROM Victim
WHERE Status = 'Deceased' AND Relief_Center_ID IS NOT NULL;
```
```

6. **List volunteers with "First Aid" skills not in any team.**

```
```sql
SELECT v.Volunteer_ID, v.Skills
FROM Volunteer v
LEFT JOIN (SELECT Union of all Skills from Volunteer) AS All_Skills ON v.Skills = 'First Aid';
```
```

7. **Count active teams assigned to ongoing disasters.**

```
```sql
```

```

SELECT d.Disaster_ID, COUNT(dat.Team_ID) AS Team_Count
FROM Disaster d
JOIN Disaster_Assigned_Teams dat ON d.Disaster_ID = dat.Disaster_ID
WHERE d.Status = 'Ongoing'
GROUP BY d.Disaster_ID;
...

```

8. **\*\*Find donors contributing >\$5000 across all purposes:\*\***

```

```sql
SELECT Donor_Name, SUM(Amount) AS Total_Donation
FROM Donation
GROUP BY Donor_Name
HAVING SUM(Amount) > 5000;
...

```

9. ****Disasters with no assigned rescue teams:****

```

```sql
SELECT d.*
FROM Disaster d
LEFT JOIN Disaster_Assigned_Teams dat ON d.Disaster_ID = dat.Disaster_ID
WHERE dat.Team_ID IS NULL;
...

```

10. **\*\*Affected areas with above-average population impact:\*\***

```

```sql
SELECT Area_Name, Population_Affected
FROM Affected_Area
WHERE Population_Affected > (
    SELECT AVG(Population_Affected)
    FROM Affected_Area
);
...

```

END

From the above interaction we found that Queries 2,4, and 6 are not correct. They have key issues :

- Schemas Ignorance
- Logical error
- Syntax blunder