1. Design and implement a *normalized OLTP database* based on the r=extracted Accidents and Vehicles data. **You will then develop 2 Views and 5 useful SQL queries. At least two of the 5 SQL queries must use one or more of these views. At least two of the 5 SQL queries must employ aggregations. At least three of the 5 SQL queries must involve JOINS ort sub-queries**

Utilizing the Schema provided, please try out these queries

VIEWS QUERIES

1. View_Accident Summary

This view provides a summary of accidents, including details about the driver and vehicle involved.

```
CREATE VIEW View_AccidentSummary AS
SELECT
    ad.Accident_Index,
    ad.Date,
    ad.Time,
    ad.Accident_Severity,
    ad.Number_of_Casualties,
    d.Age_Band_of_Driver,
    d.Sex_of_Driver,
    v.Vehicle_Type,
    v.Make,
    v.Model
FROM Accident_Details ad
JOIN Driver d ON ad.Accident_Index = d.Accident_Index
JOIN Vehicle v ON ad.Accident_Index = v.Accident_Index;
```

2:. View: View_LocationWeather

This view combines accident details with location and external condition information to analyze environmental factors.

```
CREATE VIEW View_LocationWeather AS
SELECT
    ad.Accident_Index,
    ad.Date,
    l.Latitude,
    l.Longitude,
```

```
    l.Road_Type,
    ec.Weather_Conditions,
    ec.Light_Conditions,
    ec.Road_Surface_Conditions
FROM Accident_Details ad
JOIN Location l ON ad.Accident_Index = l.Accident_Index
JOIN External_Conditions ec ON ad.Accident_Index = ec.Accident_Index;
```

OTHER QUERIES

Query 1 : List Accident Summary by Severity

Uses the View_AccidentSummary view and involves an aggregation.

```
SELECT
    Accident_Severity,
    COUNT(*) AS Total_Accidents,
    AVG(Number_of_Casualties) AS Avg_Casualties
FROM View_AccidentSummary
GROUP BY Accident_Severity;
```

Query 2: Count of Accidents by Vehicle Type and Gender of Driver

Uses the View_AccidentSummary view and employs aggregation.

```
SELECT

    Vehicle_Type,

    Sex_of_Driver,

    COUNT(*) AS Accident_Count

FROM View_AccidentSummary

GROUP BY Vehicle_Type, Sex_of_Driver;
```

Query 3: Detailed Accident Information on Specific Date

Involves a join and uses the View_LocationWeather.

```
SELECT
    aws.Accident_Index,
    aws.Date,
    aws.Time,
```

```
    aws.Vehicle_Type,
    lw.Weather_Conditions,
    lw.Light_Conditions
FROM View_AccidentSummary aws
JOIN View_LocationWeather lw ON aws.Accident_Index =
lw.Accident_Index
WHERE aws.Date = '2023-01-01';
```

Query 4: Find Accidents with Specific Weather and Road Conditions

Involves sub-queries

```
SELECT
    ad.Accident_Index,
    ad.Date,
    ad.Time,
    ec.Weather_Conditions,
    ec.Road_Surface_Conditions
FROM Accident_Details ad
WHERE EXISTS (
    SELECT 1 FROM External_Conditions ec
    WHERE ad.Accident_Index = ec.Accident_Index
    AND ec.Weather_Conditions = 'Rainy'
    AND ec.Road_Surface_Conditions = 'Wet'
);
```

Query 5: Average Number of Casualties in Urban vs Rural Areas

Involves a join and aggregation

```
SELECT

    ec.Urban_or_Rural_Area,

    AVG(ad.Number_of_Casualties) AS Avg_Casualties

FROM Accident_Details ad

JOIN External_Conditions ec ON ad.Accident_Index = ec.Accident_Index

P)HMP0-In{
```