COVID-19 Analytics System (CAMS)

Database Design

Entities and related attributes:

- **Continent** (Continent ID (PK), Continent Name)
- **Country** (Country Id (PK), Country Name, Total Population, Continent Id (FK), Total Regions)
- **State/Region** (Region Id (PK), Region Name, Population, Country Id (FK))
- **Daily Covid-19 Reports** (Report Id (PK), Date, Total No. Of Cases, Total No. Of Deaths, Total No. Of Recoveries, Region Id (FK))
- **Vaccination Data** (Vaccination Id (PK), Date, Total No. Of Vaccinations, Region Id (FK))
- **Testing Data** (Testing Id (PK), Date, Total Tests, Positive Tests, Region Id (FK))
- **Public Health Measures** (Measures Id (PK), Measure Type, Start Date, End Date, Region Id (FK))

SQL Source Code

```
CREATE DATABASE CAMS_PROJECT

USE CAMS_PROJECT

-- PROCEDURE for creation of tables

      CREATE PROCEDURE CreateCOVID19Tables

      AS

      BEGIN

           -- Continent table

                 CREATE TABLE Continent (

                     ContinentID INT PRIMARY KEY IDENTITY(1,1),

                     ContinentName VARCHAR(255) NOT NULL

                 );


           -- Country table

                 CREATE TABLE Country (

                     CountryID INT PRIMARY KEY IDENTITY(1,1),

                     CountryName VARCHAR(255) NOT NULL,

                     TotalPopulation INT,

                     ContinentID INT,

                     FOREIGN KEY (ContinentID) REFERENCES Continent(ContinentID)

                 );


           -- Region table
```

```
    CREATE TABLE Region (

        RegionID INT NOT NULL IDENTITY(1,1),

        RegionName VARCHAR(255) NOT NULL,

        Population INT,

        CountryID INT

    );

      ALTER TABLE Region

      ADD CONSTRAINT PK_Region PRIMARY KEY (RegionID)

      ALTER TABLE Region

      ADD CONSTRAINT FK_Region_Country FOREIGN KEY (CountryID) REFERENCES
Country(CountryID)
```

-- DailyCovid19Reports table

```
    CREATE TABLE DailyCovid19Reports (

        ReportID INT NOT NULL IDENTITY(1,1),

        Date DATE NOT NULL,

        TotalCases INT,

        TotalDeaths INT,

        TotalRecoveries INT,

        RegionID INT

    );

      ALTER TABLE DailyCovid19Reports

      ADD CONSTRAINT PK_DailyCovid19Reports PRIMARY KEY (ReportID)

      ALTER TABLE DailyCovid19Reports

      ADD CONSTRAINT FK_DailyCovid19Reports_Region FOREIGN KEY (RegionID)
REFERENCES Region(RegionID)
```

-- VaccinationData table

```
    CREATE TABLE VaccinationData (

        VaccinationID INT PRIMARY KEY IDENTITY(1,1),

        Date DATE NOT NULL,

        TotalVaccinations INT,

        RegionID INT,

        FOREIGN KEY (RegionID) REFERENCES Region(RegionID)

    );
```

-- TestingData table

```
    CREATE TABLE TestingData (

        TestingID INT PRIMARY KEY IDENTITY(1,1),

        Date DATE NOT NULL,
                      TotalTests INT,
                      PositiveTests INT,
                      RegionID INT,
                      FOREIGN KEY (RegionID) REFERENCES Region(RegionID)
                  );
            -- PublicHealthMeasures table
                  CREATE TABLE PublicHealthMeasures (
                      MeasuresID INT PRIMARY KEY IDENTITY(1,1),
                      MeasureType VARCHAR(255) NOT NULL,
                      StartDate DATE NOT NULL,
                      EndDate DATE NOT NULL,
                      RegionID INT,
                      FOREIGN KEY (RegionID) REFERENCES Region(RegionID)
                  );
        END;
EXEC CreateCOVID19Tables
-- Procdeure to display the data of specific table
        CREATE OR ALTER PROCEDURE DisplayTableData
        @TableName VARCHAR(255)
        AS
        BEGIN
            DECLARE @SqlQuery NVARCHAR(MAX)
            SET @SqlQuery = 'SELECT * FROM ' + QUOTENAME(@TableName);
            EXEC sp_executesql @SqlQuery;
        END;
-- Stored procedure to insert values into Continent table
        CREATE PROCEDURE InsertContinent
        @ContinentName VARCHAR(255)
        AS
        BEGIN
            INSERT INTO Continent (ContinentName)
            VALUES (@ContinentName);
        END;
-- Insert data into Continent table
        EXEC InsertContinent 'Asia'
        EXEC InsertContinent 'Africa'
        EXEC InsertContinent 'Europe'

        EXEC InsertContinent 'North America'

        EXEC InsertContinent 'South America'

EXEC DisplayTableData @TableName = 'Continent'

-- Stored procedure to insert values into Country table

        CREATE PROCEDURE InsertCountry

            @CountryName VARCHAR(255),

            @TotalPopulation INT,

            @ContinentID INT

        AS

        BEGIN

            INSERT INTO Country (CountryName, TotalPopulation, ContinentID)

            VALUES (@CountryName, @TotalPopulation, @ContinentID);

        END

-- Insert data into Country table

        EXEC InsertCountry 'China', 1444216107, 1;

        EXEC InsertCountry 'Pakistan', 243087662, 1;

        EXEC InsertCountry 'Nigeria', 211400708, 2;

        EXEC InsertCountry 'South Africa', 60041930, 2;

        EXEC InsertCountry 'Germany', 83132799, 3;

        EXEC InsertCountry 'France', 65273511, 3;

        EXEC InsertCountry 'United States', 332915073, 4;

        EXEC InsertCountry 'Canada', 38115509, 4;

        EXEC InsertCountry 'Brazil', 213993437, 5;

        EXEC InsertCountry 'Argentina', 45785022, 5;

EXEC DisplayTableData @TableName = 'Country'

-- Stored procedure to insert values into REGION table

        CREATE PROCEDURE InsertRegion

            @RegionName VARCHAR(255),

            @Population INT,

            @CountryName VARCHAR(255)

        AS

        BEGIN

            DECLARE @CountryID INT;

            SELECT @CountryID = CountryID

            FROM Country
```

```sql
        WHERE CountryName = @CountryName;

        INSERT INTO Region (RegionName, Population, CountryID)

        VALUES (@RegionName, @Population, @CountryID);

    END;

-- Insert regions for China

    EXEC InsertRegion 'Beijing', 21542000, 'China';

-- Insert regions for Pakistan

    EXEC InsertRegion 'Karachi', 15741000, 'Pakistan';

-- Insert regions for Nigeria

    EXEC InsertRegion 'Lagos', 14040000, 'Nigeria';

-- Insert regions for South Africa

    EXEC InsertRegion 'Johannesburg', 5796010, 'South Africa';

-- Same goes for other entries too

EXEC DisplayTableData @TableName = 'Region'

-- Stored procedure to insert values into DailyCovid19Report table

    CREATE PROCEDURE InsertDailyCovid19Report

        @Date DATE,

        @TotalCases INT,

        @TotalDeaths INT,

        @TotalRecoveries INT,

        @RegionName VARCHAR(255)

    AS

    BEGIN

        DECLARE @RegionID INT;

        SELECT @RegionID = RegionID

        FROM Region

        WHERE RegionName = @RegionName;

        INSERT INTO DailyCovid19Reports (Date, TotalCases, TotalDeaths, TotalRecoveries,
    RegionID)

        VALUES (@Date, @TotalCases, @TotalDeaths, @TotalRecoveries, @RegionID);

    END;


-- Insert reports for Beijing (RegionID = 1)

    EXEC InsertDailyCovid19Report '2023-02-01', 120, 7, 100, 'Beijing';

    EXEC InsertDailyCovid19Report '2023-02-02', 130, 8, 110, 'Beijing';

-- Insert reports for Shanghai (RegionID = 2)
```

```sql
       EXEC InsertDailyCovid19Report '2023-02-01', 45, 3, 40, 'Shanghai';
       EXEC InsertDailyCovid19Report '2023-02-02', 50, 4, 45, 'Shanghai';
-- Insert reports for Guangzhou (RegionID = 3)
       EXEC InsertDailyCovid19Report '2023-02-01', 75, 4, 70, 'Guangzhou';
       EXEC InsertDailyCovid19Report '2023-02-02', 80, 5, 75, 'Guangzhou';
-- Same goes for other entries too.
EXEC DisplayTableData @TableName = 'DailyCovid19Reports'
-- Stored procedure to insert values into VaccinationData  table
       CREATE PROCEDURE InsertVaccinationData
           @Date DATE,
           @TotalVaccinations INT,
           @RegionName VARCHAR(50)
       AS
       BEGIN
           SET NOCOUNT ON;
           DECLARE @RegionID INT;
           SELECT @RegionID = RegionID
           FROM Region
           WHERE RegionName = @RegionName;
           INSERT INTO VaccinationData (Date, TotalVaccinations, RegionID)
           VALUES (@Date, @TotalVaccinations, @RegionID);
       END;
-- Insert data for RegionID = 1
       EXEC InsertVaccinationData '2023-02-01', 1200, 'Beijing';
-- Insert data for RegionID = 2
       EXEC InsertVaccinationData '2023-02-01', 1700, 'Shanghai';
-- Insert data for RegionID = 3
       EXEC InsertVaccinationData '2023-02-01', 2200, 'Guangzhou';
-- Insert data for RegionID = 4
       EXEC InsertVaccinationData '2023-02-01', 2800, 'Karachi';
-- Same goes for other entries too.
EXEC DisplayTableData @TableName = 'VaccinationData'
-- Create stored procedure for inserting data into TestingData
       CREATE PROCEDURE InsertTestingData
           @Date DATE,
           @TotalTests INT,
           @PositiveTests INT,
           @RegionName VARCHAR(50)
       AS
       BEGIN
           DECLARE @RegionID INT;
           -- Get RegionID based on RegionName
           SELECT @RegionID = RegionID FROM Region WHERE RegionName = @RegionName;
           -- Insert data into TestingData
           INSERT INTO TestingData (Date, TotalTests, PositiveTests, RegionID)
           VALUES (@Date, @TotalTests, @PositiveTests, @RegionID);
       END
-- Testing Data for Country 1 (China)
       EXEC InsertTestingData '2023-02-01', 1000, 100, 'Beijing';
       EXEC InsertTestingData '2023-02-01', 1100, 110, 'Shanghai';
       EXEC InsertTestingData '2023-02-02', 1000, 100, 'Guangzhou';
-- Testing Data for Country 2 (Pakistan)
       EXEC InsertTestingData '2023-02-01', 800, 80, 'Karachi';
       EXEC InsertTestingData '2023-02-02', 850, 85, 'Lahore';
       EXEC InsertTestingData '2023-02-02', 800, 80, 'Islamabad';
-- Testing Data for Country 3 (Nigeria)
       EXEC InsertTestingData '2023-02-01', 600, 60, 'Lagos';
       EXEC InsertTestingData '2023-02-02', 650, 65, 'Kano';
       EXEC InsertTestingData '2023-02-01', 500, 50, 'Ibadan';
-- Same goes for other entries too.
EXEC DisplayTableData @TableName = 'TestingData'
-- Procedure for insertion in PublicHealthMeasures.
       CREATE PROCEDURE InsertPublicHealthMeasures
           @MeasureType VARCHAR(255),
           @StartDate DATE,
           @EndDate DATE,
           @RegionName VARCHAR(255)
       AS
       BEGIN
           DECLARE @RegionID INT;
           SELECT @RegionID = RegionID
           FROM Region
           WHERE RegionName = @RegionName;

           INSERT INTO PublicHealthMeasures (MeasureType, StartDate, EndDate, RegionID)

           VALUES (@MeasureType, @StartDate, @EndDate, @RegionID);

       END;


-- Insert public health measures data for Beijing (RegionID = 1)

       EXEC InsertPublicHealthMeasures 'Stay-at-Home Order', '2023-02-01', '2023-02-15', 'Beijing';

       EXEC InsertPublicHealthMeasures 'Curfew', '2023-02-16', '2023-02-28', 'Beijing';

-- Insert public health measures data for Shanghai (RegionID = 2)

       EXEC InsertPublicHealthMeasures 'Business Restrictions', '2023-02-01', '2023-02-15', 'Shanghai';

       EXEC InsertPublicHealthMeasures 'Public Transportation Limits', '2023-02-16', '2023-02-28', 'Shanghai';

-- Insert public health measures data for Guangzhou (RegionID = 3)

       EXEC InsertPublicHealthMeasures 'Outdoor Gathering Limits', '2023-02-01', '2023-02-15', 'Guangzhou';

       EXEC InsertPublicHealthMeasures 'School Closures', '2023-02-16', '2023-02-28', 'Guangzhou';

-- Insert public health measures data for Karachi (RegionID = 4)

       EXEC InsertPublicHealthMeasures 'Workplace Sanitization', '2023-02-01', '2023-02-15', 'Karachi';

       EXEC InsertPublicHealthMeasures 'Telecommuting Mandate', '2023-02-16', '2023-02-28', 'Karachi';

-- Insert public health measures data for Lahore (RegionID = 5)

       EXEC InsertPublicHealthMeasures 'Public Event Cancellations', '2023-02-01', '2023-02-15', 'Lahore';

       EXEC InsertPublicHealthMeasures 'Remote Learning', '2023-02-16', '2023-02-28', 'Lahore';

-- Insert public health measures data for Islamabad (RegionID = 6)

       EXEC InsertPublicHealthMeasures 'Travel Restrictions', '2023-02-01', '2023-02-15', 'Islamabad';

       EXEC InsertPublicHealthMeasures 'Quarantine Protocols', '2023-02-16', '2023-02-28', 'Islamabad';

EXEC DisplayTableData @TableName = 'PublicHealthMeasures'
```

Queries

1. **Queries to drop procedure, table or database itself**
    - **Dropping Procedure:**

```
DROP PROCEDURE IF EXISTS InsertVaccinationData;
DROP PROCEDURE IF EXISTS InsertTestingData;
DROP PROCEDURE IF EXISTS InsertPublicHealthMeasures;
DROP PROCEDURE IF EXISTS InsertContinent;
DROP PROCEDURE IF EXISTS InsertCountry;
DROP PROCEDURE IF EXISTS InsertDailyCovid19Reports;
```

- **Dropping Tables:**

```
-- Procedure to drop tables if they exist
CREATE PROCEDURE DropTablesIfExists
AS
BEGIN
    IF OBJECT_ID('PublicHealthMeasures', 'U') IS NOT NULL
    BEGIN
        DROP TABLE PublicHealthMeasures;
    END
    IF OBJECT_ID('TestingData', 'U') IS NOT NULL
    BEGIN
        DROP TABLE TestingData;
    END
IF OBJECT_ID('VaccinationData', 'U') IS NOT NULL
    BEGIN
        DROP TABLE VaccinationData;
    END
    IF OBJECT_ID('DailyCovid19Reports', 'U') IS NOT NULL
    BEGIN
        DROP TABLE DailyCovid19Reports;
    END
    IF OBJECT_ID('Region', 'U') IS NOT NULL
    BEGIN
        DROP TABLE Region;
    END
    IF OBJECT_ID('Country', 'U') IS NOT NULL
    BEGIN
        DROP TABLE Country;
    END
    IF OBJECT_ID('Continent', 'U') IS NOT NULL
    BEGIN
        DROP TABLE Continent;
    END
END;
EXEC DropTablesIfExists
```

- **Dropping Database:**

```
-- This drops the entire database along with all its objects.
```

```
DROP DATABASE IF EXISTS CAMS_PROJECT;
```

2. **Queries to alter a table.**

- **Alter Table:**

```
-- Adding a new column
ALTER TABLE Continent
ADD TotalCountries INT;
-- Dropping a column
ALTER TABLE Continent
DROP COLUMN TotalCountries;

-- Renaming Column
ALTER TABLE Continent
ALTER COLUMN ContinentName NVARCHAR(255);
```

3. **Queries to rename procedure, table or column.**

- **Rename Table:**

```
-- Renaming the table
EXEC sp_rename 'Region', 'StateProvince';
```

- **Rename Column:**

```
-- Renaming the column
EXEC sp_rename 'Country.CountryName', 'CountryOfficialName', 'COLUMN';
```

- **Rename Procedure:**

```
-- Renaming Procedure
EXEC sp_rename 'InsertRegion', 'InsertStateProvince';
```

4. **Queries to delete and update in a table.**

- **Deletion:**

```
-- Procedure for deletion
CREATE PROCEDURE DeleteData
    @TableName VARCHAR(255),
    @Condition NVARCHAR(MAX)
AS
BEGIN
    DECLARE @SqlQuery NVARCHAR(MAX);
    SET @SqlQuery = 'DELETE FROM ' + QUOTENAME(@TableName) + ' WHERE ' + @Condition;
    EXEC sp_executesql @SqlQuery;
END;

EXEC DeleteData @TableName = 'Country', @Condition = 'CountryID = 1';
```

- **Updation:**

```
-- Procedure for updation
CREATE PROCEDURE UpdateData
    @TableName VARCHAR(255),
    @SetClause NVARCHAR(MAX),
    @Condition NVARCHAR(MAX)
AS
BEGIN
    DECLARE @SqlQuery NVARCHAR(MAX);
    SET @SqlQuery = 'UPDATE ' + QUOTENAME(@TableName) + ' SET ' + @SetClause + '
WHERE ' + @Condition;
    EXEC sp_executesql @SqlQuery;
END;

EXEC UpdateData @TableName = 'Country', @SetClause = 'TotalPopulation = 250000000',
@Condition = 'CountryName = ''Pakistan''';
```

In above queries, simple SQL operations has been performed. In the upcoming section I have utilize SQL concepts like Joins, Aggregate Functions, Subqueries, Order By, Group By and Logical, Comparison Operators both combine and individual to take a deep insight in the COVID-19 analysis.

1. **Query to check the total population by a continent.**

```
SELECT Continent.ContinentName, SUM(Country.TotalPopulation) AS TotalPopulation
FROM Continent
JOIN Country  ON Continent.ContinentID = Country.ContinentID
GROUP BY Continent.ContinentName;
```

2. **Query to check Total Cases, Deaths, and Recoveries by Date.**

```
SELECT Date,
       SUM(TotalCases) AS TotalCases,
       SUM(TotalDeaths) AS TotalDeaths,
       SUM(TotalRecoveries) AS TotalRecoveries
FROM DailyCovid19Reports
GROUP BY Date
ORDER BY Date;
```

3. **Query to check the Average Testing Positivity Rate by Region.**

```
SELECT r.RegionName,
       AVG((td.PositiveTests / td.TotalTests)) * 100 AS PositivityRate
FROM Region r
JOIN TestingData td ON r.RegionID = td.RegionID
GROUP BY r.RegionName;
```

4. **Query to check Maximum Vaccinations Administered in a Single Day.**

```
SELECT MAX(TotalVaccinations) AS MaxVaccinations
FROM VaccinationData ;
```

5. **Query to Check Total Public Health Measures Implemented per Country.**

```
SELECT co.CountryName,
       COUNT(phm.MeasuresID) AS TotalMeasuresImplemented
FROM Country co
JOIN Region r ON co.CountryID = r.CountryID
JOIN PublicHealthMeasures phm ON r.RegionID = phm.RegionID
GROUP BY co.CountryName;
```

6. **Query to Check Top 5 Countries by Total Cases.**

```
SELECT TOP (5) c.CountryName,
       SUM(d.TotalCases) AS TotalCases
FROM Country c
JOIN Region r ON c.CountryID = r.CountryID
JOIN DailyCovid19Reports d ON r.RegionID = d.RegionID
GROUP BY c.CountryName
ORDER BY TotalCases DESC
```

7. **Query to Total Cases and Vaccinations by Country.**

```
SELECT c.CountryName,
       SUM(d.TotalCases) AS TotalCases,
       SUM(v.TotalVaccinations) AS TotalVaccinations
FROM Country c
JOIN Region r ON c.CountryID = r.CountryID
JOIN DailyCovid19Reports d ON r.RegionID = d.RegionID
JOIN VaccinationData v ON r.RegionID = v.RegionID
```

```
GROUP BY c.CountryName
      ORDER BY TotalCases DESC;
```

8. **Query to Check Daily Case Trends by Country.**

```
SELECT CountryName,
       d.Date,
       d.TotalCases
FROM Country c
JOIN Region r ON c.CountryID = r.CountryID
JOIN DailyCovid19Reports d ON r.RegionID = d.RegionID
ORDER BY c.CountryName, d.Date;
```

9. **Query to identify Countries with Missing Vaccination Data.**

```
SELECT c.CountryName
FROM Country c
LEFT JOIN Region r ON c.CountryID = r.CountryID
LEFT JOIN VaccinationData v ON r.RegionID = v.RegionID
WHERE v.RegionID IS NULL
      ORDER BY c.CountryName;
```

10. **Query to identify regions with both high vaccination rates and low positivity rates.**

```
SELECT c.CountryName,
       r.RegionName,
       SUM(v.TotalVaccinations) / r.Population * 100 AS VaccinationRate,
       AVG(CAST(d.PositiveTests / d.TotalTests AS NUMERIC(5,2))) * 100 AS
PositiveTestRate
FROM Country c
JOIN Region r ON c.CountryID = r.RegionID
JOIN VaccinationData v ON r.RegionID = v.RegionID
JOIN TestingData d ON r.RegionID = d.RegionID
WHERE VaccinationRate > 70 AND PositiveTestRate < 5
ORDER BY CountryName, RegionName;
```