# COVID-19 Data Analysis

This repository contains SQL queries used for analyzing COVID-19 data, including cases, deaths, and vaccinations. The data is imported from two main tables: `CovidDeaths` and `CovidVaccinations`. Below is a description of each query and its purpose.

## Viewing Imported Data

### Query 1: Viewing CovidVaccinations Data

```sql

SELECT \* FROM covidde1..CovidVaccinations ORDER BY 3,4;

```

- \*\*Purpose\*\*: Display all data from the `CovidVaccinations` table, ordered by the third and fourth columns.

### Query 2: Viewing CovidDeaths Data

```sql

SELECT \* FROM covidde1..CovidDeaths ORDER BY 3,4;

```

- \*\*Purpose\*\*: Display all data from the `CovidDeaths` table, ordered by the third and fourth columns.

## COVID Deaths Analysis

### Query 3: Highest Total Cases per Location

```sql

SELECT location, population, MAX(CAST(total\_cases AS int)) AS highest FROM covidde1..CovidDeaths GROUP BY location, population ORDER BY 1,2;

```

- \*\*Purpose\*\*: Find the highest total cases per location and display them along with the population.

### Query 4: Select Data for Deaths Analysis

```sql

SELECT location, date, total\_cases, new\_cases, total\_deaths, population FROM covidde1..CovidDeaths ORDER BY 1,2;

```

- \*\*Purpose\*\*: Select specific columns for further analysis of COVID deaths data.

### Query 5: Calculate Percentage of Deaths

```sql

SELECT location, date, total\_cases, total\_deaths, (CAST(total\_deaths AS int) / CAST(CAST(total\_cases AS int) AS float) \* 100) AS percentage\_deaths FROM covidde1..CovidDeaths ORDER BY 1,2;

```

- \*\*Purpose\*\*: Calculate the percentage of deaths relative to total cases.

### Query 6: Calculate Percentage of Cases

```sql

SELECT location, date, population, total\_cases, CAST(CAST(total\_cases AS int) AS float) / population AS percentage\_cases FROM covidde1..CovidDeaths ORDER BY 1,2;

```

- \*\*Purpose\*\*: Calculate the percentage of cases relative to the population.

### Query 7: Countries with Highest Infection Rate

```sql

SELECT location, population, MAX(CAST(total\_cases AS int)) AS HighestInfectionCount, MAX(total\_cases / population) \* 100 AS percentage\_cases FROM covidde1..CovidDeaths GROUP BY location, population ORDER BY percentage\_cases DESC;

```

- \*\*Purpose\*\*: Identify countries with the highest infection rates relative to their population.

### Query 8: Countries with Highest Death Count per Population

```sql

SELECT location, population, MAX(CAST(total\_deaths AS int)) AS HighestDeathCount, (MAX(CAST(total\_deaths AS float)) / population) \* 100 AS percentage\_deaths FROM covidde1..CovidDeaths WHERE continent IS NOT NULL GROUP BY location, population ORDER BY percentage\_deaths DESC;

```

- \*\*Purpose\*\*: Identify countries with the highest death counts relative to their population.

### Query 9: Locations with Highest Death Count (without Continent)

```sql

SELECT location, MAX(CAST(total\_deaths AS int)) AS HighestDeathCount FROM covidde1..CovidDeaths WHERE continent IS NULL GROUP BY location ORDER BY HighestDeathCount DESC;

```

- \*\*Purpose\*\*: Find locations with the highest death counts where the continent is not specified.

### Query 10: Continents with Highest Death Counts

```sql

SELECT continent, MAX(CAST(total\_deaths AS int)) AS HighestDeathCount FROM covidde1..CovidDeaths WHERE continent IS NOT NULL GROUP BY continent ORDER BY HighestDeathCount DESC;

```

- \*\*Purpose\*\*: Identify continents with the highest death counts.

### Query 11: Global Numbers - Daily New Cases and Deaths

```sql

SELECT date, SUM(new\_cases) AS Cases\_worldwide, SUM(new\_deaths) AS Deaths\_worldwide, SUM(new\_deaths) / SUM(new\_cases) AS DeathPercentage FROM covidde1..CovidDeaths WHERE new\_cases IS NOT NULL AND new\_deaths IS NOT NULL AND new\_cases != 0 AND new\_deaths != 0 AND continent IS NOT NULL GROUP BY date ORDER BY 1;

```

- \*\*Purpose\*\*: Calculate daily global new cases, new deaths, and death percentage.

### Query 12: Global Numbers - Total New Cases and Deaths

```sql

SELECT SUM(new\_cases) AS Cases\_worldwide, SUM(new\_deaths) AS Deaths\_worldwide, SUM(new\_deaths) / SUM(new\_cases) AS DeathPercentage FROM covidde1..CovidDeaths WHERE new\_cases IS NOT NULL AND new\_deaths IS NOT NULL AND new\_cases != 0 AND new\_deaths != 0 AND continent IS NOT NULL ORDER BY 1;

```

- \*\*Purpose\*\*: Calculate total new cases, new deaths, and death percentage globally.

## COVID Vaccinations Analysis

### Query 13: Join Deaths and Vaccinations Data

```sql

SELECT \* FROM covidde1..CovidDeaths dea JOIN covidde1..CovidVaccinations vac ON dea.location = vac.location AND dea.date = vac.date;

```

- \*\*Purpose\*\*: Join `CovidDeaths` and `CovidVaccinations` tables to analyze combined data.

### Query 14: Total Population vs. Vaccinations with Rolling Count

```sql

SELECT dea.continent, dea.location, dea.date, dea.population, vac.new\_vaccinations, SUM(CAST(vac.new\_vaccinations AS bigint)) OVER (Partition BY dea.location ORDER BY dea.location, dea.date) AS VaccinationRollingCount FROM covidde1..CovidDeaths dea JOIN covidde1..CovidVaccinations vac ON dea.location = vac.location AND dea.date = vac.date WHERE dea.continent IS NOT NULL ORDER BY 2,3;

```

- \*\*Purpose\*\*: Calculate the rolling count of vaccinations relative to the total population.

## Using Common Table Expressions (CTEs) and Temporary Tables

### Query 15: Using CTE to Create a Refined Table

```sql

WITH PopvsVac (Continent, Location, Date, Population, New\_Vaccinations, VaccinationRollingCount) AS (

SELECT dea.continent, dea.location, dea.date, dea.population, vac.new\_vaccinations, SUM(CAST(vac.new\_vaccinations AS bigint)) OVER (Partition BY dea.location ORDER BY dea.location, dea.date) AS VaccinationRollingCount

FROM covidde1..CovidDeaths dea

JOIN covidde1..CovidVaccinations vac ON dea.location = vac.location AND dea.date = vac.date

WHERE dea.continent IS NOT NULL

)

SELECT \*, (VaccinationRollingCount / Population) \* 100 AS Percentage\_vaccinated\_rolling FROM PopvsVac;

```

- \*\*Purpose\*\*: Create a refined dataset using a CTE to calculate the rolling percentage of vaccinations.

### Query 16: Using Temporary Table for Vaccination Data

```sql

DROP TABLE IF EXISTS #PercentPopulationVaccinated;

CREATE TABLE #PercentPopulationVaccinated (

Continent NVARCHAR(255),

Location NVARCHAR(255),

Date DATETIME,

Population NUMERIC,

New\_vaccinations NUMERIC,

VaccinationRollingCount NUMERIC

);

INSERT INTO #PercentPopulationVaccinated

SELECT dea.continent, dea.location, dea.date, dea.population, vac.new\_vaccinations, SUM(CAST(vac.new\_vaccinations AS bigint)) OVER (Partition BY dea.location ORDER BY dea.location, dea.date) AS VaccinationRollingCount

FROM covidde1..CovidDeaths dea

JOIN covidde1..CovidVaccinations vac ON dea.location = vac.location AND dea.date = vac.date;

SELECT \*, (VaccinationRollingCount / Population) \* 100 AS Percentage\_vaccinated\_rolling FROM #PercentPopulationVaccinated;

```

- \*\*Purpose\*\*: Use a temporary table to store and calculate the percentage of population vaccinated.

### Query 17: Creating a View for Data Visualization

```sql

CREATE VIEW PercentPopulationVaccinated AS

SELECT dea.continent, dea.location, dea.date, dea.population, vac.new\_vaccinations, SUM(CAST(vac.new\_vaccinations AS bigint)) OVER (Partition BY dea.location ORDER BY dea.location, dea.date) AS VaccinationRollingCount

FROM covidde1..CovidDeaths dea

JOIN covidde1..CovidVaccinations vac ON dea.location = vac.location AND dea.date = vac.date

WHERE dea.continent IS NOT NULL;

```

- \*\*Purpose\*\*: Create a view to store vaccination data for easier visualization and analysis.

### Query 18: Selecting Data from the View

```sql

SELECT \* FROM PercentPopulationVaccinated;

```

- \*\*Purpose\*\*: Select all data from the created view for analysis or visualization purposes.