CALCULATED COLUMS AND DAX MEASURES

1 Calculated Columns

1.1 OWID Dataset compact.csv:

1) **Population Category**
Classifies each country based on total population size: Small (<10M), Medium (10M–100M), or Large (>100M).

```
Population Category =
SWITCH(
    TRUE(),
    OWID_Compact[population] <= 10000000, "Small",
    OWID_Compact[population] <= 100000000, "Medium",
    OWID_Compact[population] > 100000000, "Large"
```

2) **Population Density Category**
Groups countries by population density: Low (<100), Medium (100–500), or High (>500).

```
Population Density Category =
SWITCH(
    TRUE(),
    OWID_Compact[population_density] <= 100, "Low",
    OWID_Compact[population_density] <= 500, "Medium",
    OWID_Compact[population_density] > 500, "High"
)
```

3) **Demographic Structure**
Categorizes countries as having a young, balanced, or aging population based on median age and life expectancy.

```
Demographic Structure =
SWITCH(
    TRUE(),
    OWID_Compact[median_age] < 25 || OWID_Compact[life_expectancy] < 70,       "Young
 population",
    OWID_Compact[median_age] >= 25 && OWID_Compact[median_age] <= 40 &&
 OWID_Compact[life_expectancy] >= 70 && OWID_Compact[life_expectancy] <= 78, "Balanced
 population",
    OWID_Compact[median_age] > 40 || OWID_Compact[life_expectancy] > 78, "Aging
 population",
    "Unknown"
)
```

4) **GDP per Capita Category**
Segments countries by GDP per capita: Low (<4000), Middle (4000–15000), or High (>15000).

```
GDP per Capita Category =
SWITCH(
    TRUE(),
    OWID_Compact[gdp_per_capita] < 4000, "Low",
    OWID_Compact[gdp_per_capita] >= 4000 && OWID_Compact[gdp_per_capita] <= 15000, "Middle",
    OWID_Compact[gdp_per_capita] > 15000, "High",
    "Unknown"
```

1.2 OWID Dataset vaccinations_manufacturer.csv:

1) **Daily Doses**
Calculates the number of vaccine doses administered daily by subtracting the cumulative value from the previous day.

```
Daily Doses =
VAR CurrentCountry = OWID_Vacc[country]
VAR CurrentVaccine = OWID_Vacc[vaccine]
VAR CurrentDate = OWID_Vacc[date]
VAR PreviousDayValue =
    CALCULATE(
        MAX(OWID_Vacc[total_vaccinations]),
        FILTER(
            OWID_Vacc,
            OWID_Vacc[country] = CurrentCountry &&
            OWID_Vacc[vaccine] = CurrentVaccine &&
            OWID_Vacc[date] = CurrentDate - 1
        )
    )
RETURN
IF(
    ISBLANK(PreviousDayValue),
    BLANK(),
    OWID_Vacc[total_vaccinations] - PreviousDayValue
)
```

1.3 VAERS Dataset data.csv:

1) **Days from Vaccination to Death**
Measures the number of days between vaccination and death to analyze delay patterns in fatal events.

```
Days from Vaccination to Death =
VAR diff = DATEDIFF(VAERS_Data[VAX_DATE], VAERS_Data[DATEDIED], DAY)
RETURN IF(diff >= 0, diff, BLANK())
```

2) **Days from Onset to Death**
Calculates the time from symptom onset to death for a more clinical view of disease progression.

```
Days from Onset to Death =
VAR diff = DATEDIFF(VAERS_Data[ONSET_DATE], VAERS_Data[DATEDIED], DAY)
RETURN IF(diff >= 0, diff, BLANK())
```

3) **Age Category**
Groups patients into Child, Young Adult, Adult, or Senior based on their age in years.

```
Age Category =
SWITCH(
    TRUE(),
    ISBLANK(VAERS_Data[AGE_YRS]), BLANK(),
    VAERS_Data[AGE_YRS] <= 17, "Child",
    VAERS_Data[AGE_YRS] <= 44, "Young adult",
    VAERS_Data[AGE_YRS] <= 64, "Adult",
    VAERS_Data[AGE_YRS] >= 65, "Senior",
    BLANK()
)
```

4) **Hospital Stay Category**
Categorizes hospital stays as Short (≤2 days), Medium (3–7 days), or Long (>7 days) based on HOSPDAYS.

```
Hospital Stay Category =
SWITCH(
    TRUE(),
    ISBLANK(VAERS_Data[HOSPDAYS]), BLANK(),
    VAERS_Data[HOSPDAYS] <= 2, "Short",
    VAERS_Data[HOSPDAYS] <= 7, "Medium",
    VAERS_Data[HOSPDAYS] > 7, "Long",
    BLANK()
)
```

1.4 VAERS Dataset symptoms:

Symptom Unpivoting (this is not a calculated column)
Converts columns SYMPTOM1–SYMPTOM5 into a single column called Symptom, allowing aggregation and ranking by frequency.

1.5 VAERS Dataset vax.csv:

1) **Dose Category**
Classifies the vaccine dose as First dose, Second dose, or Booster based on the dose sequence reported.

```
Dose Category =
```

```
SWITCH(
    TRUE(),
    VALUE(VAERS_VAX[VAX_DOSE_SERIES]) = 1, "First dose",
    VALUE(VAERS_VAX[VAX_DOSE_SERIES]) = 2, "Second dose",
    VALUE(VAERS_VAX[VAX_DOSE_SERIES]) >= 3, "Booster",
    "Unknown"
)
```

2 Dax Measures

1.1 OWID Dataset compact.csv:

A) Time Intelligence Measures

These measures compare current values to the same period last year (LY) or aggregate values year-to-date (YTD) for key COVID-19 indicators.

1) **Total Cases LY**

```
Total Cases LY =
CALCULATE(
    SUM(OWID_Compact[new_cases_smoothed]),
    SAMEPERIODLASTYEAR(dim_calendar[date])
)
```

2) **Total Cases YTD**

```
Total Cases YTD =
CALCULATE(
   SUM(OWID_Compact[new_cases_smoothed]),
   DATESYTD(dim_calendar[date])
   )
```

3) **Avg. Deaths per Million LY**

```
Avg. Deaths per Million LY =
CALCULATE(
AVERAGE(OWID_Compact[new_deaths_per_million]),
SAMEPERIODLASTYEAR(dim_calendar[date])
)
```

4) **Avg. Deaths per Million YTD**

```
Avg. Deaths per Million YTD =
CALCULATE(
    AVERAGE(OWID_Compact[new_deaths_per_million]),
    DATESYTD(dim_calendar[date])
)
```

5) **Total Tests LY**

```
Total Tests LY =
CALCULATE(
    SUM(OWID_Compact[new_tests_smoothed]),
    SAMEPERIODLASTYEAR(dim_calendar[date])
)
```

6) Total Tests YTD

```
Total Tests YTD =
CALCULATE(
    SUM(OWID_Compact[new_tests_smoothed]),
    DATESYTD(dim_calendar[date])
)
```

7) Total Vaccinations LY

```
Total Vaccinations LY =
CALCULATE(
    SUM(OWID_Compact[new_vaccinations_smoothed]),
    SAMEPERIODLASTYEAR(dim_calendar[date])
)
```

8) Total Vaccinations YTD

```
Total Vaccinations YTD =
CALCULATE(
    SUM(OWID_Compact[new_vaccinations_smoothed]),
    DATESYTD(dim_calendar[date])
)
```

9) Avg. Cases per Million LY

```
Avg. Cases per Million LY =
CALCULATE(
    AVERAGE(OWID_Compact[new_cases_smoothed_per_million]),
    SAMEPERIODLASTYEAR(dim_calendar[date])
)
```

10) Avg. Cases per Million YTD

```
Avg. Cases per Million YTD =
CALCULATE(
    AVERAGE(OWID_Compact[new_cases_smoothed_per_million]),
    DATESYTD(dim_calendar[date])
)
```

B) Aggregated Totals

These simple SUM functions return the total number of cases, deaths, vaccinations, and tests over the entire period.

1) **Total Cases**

```
Total Cases =
SUM(OWID_Compact[new_cases_smoothed])
```

2) **Total Deaths**

```
Total Deaths =
SUM(OWID_Compact[new_deaths])
```

3) **Total Vaccinations**

```
Total Vaccinations =
SUM(OWID_Compact[new_vaccinations_smoothed])
```

4) **Total Tests**

```
Total Tests =
SUM(OWID_Compact[new_tests_smoothed])
```

C) Aggregated Values for Analysis

These averages enable country-level and time-based comparisons across all major COVID indicators, healthcare capacity, and response effectiveness.

1) **Average New Cases Smoothed**

```
Average New Cases Smoothed =
AVERAGE(OWID_Compact[new_cases_smoothed])
```

2) **Average New Deaths**

```
Average New Deaths =
AVERAGE(OWID_Compact[new_deaths])
```

3) **Average New Cases Smoothed per Million**

```
Average New Cases Smoothed per Million =
AVERAGE(OWID_Compact[new_cases_smoothed_per_million])
```

4) **Average Daily Deaths per Million**

```
Average Daily Deaths per Million =
AVERAGE(OWID_Compact[new_deaths_per_million])
```

5) **Average ICU Patients**

```
Average ICU Patients =
```

```
AVERAGE(OWID_Compact[icu_patients])
```

6) Average Weekly Hospital Admissions per Million

```
Average Weekly Hospital Admissions per Million =
AVERAGE(OWID_Compact[weekly_hosp_admissions_per_million])
```

7) Average Stringency Index

```
Average Stringency Index =
AVERAGE(OWID_Compact[stringency_index])
```

8) Average Reproduction Rate

```
Average Reproduction Rate =
AVERAGE(OWID_Compact[reproduction_rate])
```

9) Average Last 10 Vaccination Values

```
Average Last 10 Vaccination Values =
VAR TablaFiltrada =
    FILTER(
        ADDCOLUMNS(
            OWID_Compact,
            "RankFecha",
            RANKX(
                FILTER(OWID_Compact, NOT
ISBLANK(OWID_Compact[people_fully_vaccinated_per_hundred])),
                OWID_Compact[date],
                ,
                DESC
            )
        ),
        [RankFecha] <= 10
    )
RETURN
AVERAGEX(
    TablaFiltrada,
    OWID_Compact[people_fully_vaccinated_per_hundred]
)
```

10) Average New Tests Smoothed

```
Average New Tests Smoothed =
AVERAGE(OWID_Compact[new_tests_smoothed])
```

11) Average New Tests Smoothed per Thousand

```
Average New Tests Smoothed per Thousand =
AVERAGE(OWID_Compact[new_tests_smoothed_per_thousand])
```

12) Average Positive Rate

```
Average Positive Rate =
AVERAGE(OWID_Compact[positive_rate])
```

13) Average Tests per Case

```
Average Tests per Case =
AVERAGE(OWID_Compact[tests_per_case])
```

14) Average Excess Mortality

```
Average Excess Mortality =
AVERAGE(OWID_Compact[excess_mortality])
```

15) Average Excess Mortality Cumulative

```
Average Excess Mortality Cumulative =
AVERAGE(OWID_Compact[excess_mortality_cumulative])
```

16) Average Excess Mortality Cumulative Absolute

```
Average Excess Mortality Cumulative Absolute =
AVERAGE(OWID_Compact[excess_mortality_cumulative_absolute])
```

17) Average Excess Mortality Cumulative per Million

```
Average Excess Mortality Cumulative per Million =
AVERAGE(OWID_Compact[excess_mortality_cumulative_per_million])
```

D) Epidemiological Ratios and Composite indicators

1) **Case Fatality Rate**
   Measures the proportion of confirmed COVID-19 cases that resulted in death, providing a basic mortality rate.

```
Case Fatality Rate =
DIVIDE(
SUM(OWID_Compact[total_deaths]),
SUM(OWID_Compact[total_cases])
)
```

2) **Vaccination vs Case Ratio**
   Compares vaccination coverage to new case incidence, suggesting whether high immunization correlates with lower infection rates.

```
Vaccination vs Case Ratio =
```

```
DIVIDE(
    [Average People Fully Vaccinated per Hundred],
    [Average New Cases Smoothed per Million]
)
```

3) **Stringency Effectiveness**

Evaluates the relationship between government restrictions and new case trends; higher ratios may imply more effective containment.

```
Stringency Effectiveness =
DIVIDE(
    [Average Stringency Index],
    [Average New Cases Smoothed per Million]
)
```

4) **Pandemic Efficiency Score**

Combines vaccination coverage, test positivity, mortality, and testing intensity into a single index to assess pandemic management quality.

```
Pandemic Efficiency Score =
VAR Vaccination = DIVIDE([Average People Fully Vaccinated per Hundred], 100)
VAR Positivity = 1 - DIVIDE([Average Positive Rate], 0.2) // peor si >20%
VAR Mortality = 1 - DIVIDE([Average New Deaths per Million], 20) // peor si >20
muertes/millón
VAR Testing = DIVIDE([Average Tests per Case], 10) // mejor si >10

RETURN
100 * DIVIDE(
    Vaccination + Positivity + Mortality + Testing,
    4
)
```

E) Ranking by Metric

These measures rank countries based on key health, socioeconomic, and pandemic performance metrics.

1) **Rank by Avg Daily Cases**

```
Rank by Avg Daily Cases =
RANKX(
ALL(OWID_Compact[Country]),
[Average New Cases Smoothed],
,
DESC
)
```

2) **Rank by Avg Daily Deaths per Million**

```
Rank by Avg Daily Deaths per Million =
```

```
RANKX(
    ALL(OWID_Compact[Country]),
    [Average Daily Deaths per Million],
    ,
    DESC
)
```

3) **Rank by Total Vaccinations**

```
Rank by Total Vaccinations =
RANKX(
    ALL(OWID_Compact[Country]),
    CALCULATE(SUM(OWID_Compact[new_vaccinations_smoothed])),
    ,
    DESC
)
```

4) **Rank by Hospital Capacity**

```
Rank by Hospital Capacity =
RANKX(
    ALL(OWID_Compact[Country]),
    CALCULATE(AVERAGE(OWID_Compact[hospital_beds_per_thousand])),
    ,
    DESC
)
```

5) **Rank by GDP per Capita**

```
Rank by GDP per Capita =
RANKX(
    ALL(OWID_Compact[Country]),
    CALCULATE(AVERAGE(OWID_Compact[gdp_per_capita])),
    ,
    DESC
)
```

6) **Rank by Population**

```
Rank by Population =
RANKX(
    ALL(OWID_Compact[Country]),
    CALCULATE(SUM(OWID_Compact[population])),
    ,
    DESC
)
```

F) Conditional Formating / Color Logic

Used to assign color indicators in visual tables based on whether values have increased or decreased compared to the previous year.

1) Color Avg. Daily Cases per Million

```
Color Avg. Daily Cases per Million =
IF(
    [Avg. Daily Cases per Million] > [Avg. Daily Cases per Million LY],
    1,
    0
)
```

2) Color Avg. Daily Deaths per Million

```
Color Avg. Daily Deaths per Million =
IF(
    [Avg. Daily Deaths per Million] > [Avg. Daily Deaths per Million LY],
    1,
    0
)
```

3) Color Total Cases

```
Color Total Cases =
IF(
    [Total Cases] > [Total Cases LY],
    1,
    0
)
```

4) Color Total Tests

```
Color Total Tests =
IF(
    [Total Tests] > [Total Tests LY],
    0,
            1
        )
```

5) Color Total Vaccinations

```
Color Total Vaccinations =
IF(
    [Total Vaccinations] > [Total Vaccinations LY],
    0,
    1
)
```

1.2 OWID Dataset vaccinations_manufacturer.csv:

A) Aggregated and Comparative Measures

These measures summarize or compare the total number of administered vaccine doses by vaccine or technology type.

1) **Sum of Vaccinations**
   Calculates the latest cumulative dose total per country and vaccine to compare absolute values over time.

```
Sum of Vaccinations =
SUMX(
    SUMMARIZE(
        OWID_Vacc,
        OWID_Vacc[country],
        OWID_Vacc[vaccine],
        "MaxVac", MAX(OWID_Vacc[total_vaccinations])
    ),
    [MaxVac]
)
```

2) **% Vaccination Share (by vaccine)**
   Shows the percentage contribution of a specific vaccine to the total doses within the current filter context.

```
% Vaccination Share =
DIVIDE(
    [Sum of Vaccinations],
    CALCULATE([Sum of Vaccinations], REMOVEFILTERS(OWID_Vacc[vaccine]))
)
```

3) **% Vaccination Share (by technology)**
   Similar to the previous one, but based on the vaccine technology category (e.g., mRNA, viral vector).

```
% Vaccination Share =
DIVIDE(
    [Sum of Vaccinations],
    CALCULATE([Sum of Vaccinations], REMOVEFILTERS(OWID_Vacc[vaccine]))
)
```

B) Ranking and Daily Intensity

Measures designed to identify leading vaccines and quantify their daily use.

1) **Vaccine Rank**
   Ranks vaccines by total doses administered within the filter context; useful to identify the most used vaccine per country.

```
Vacuna Rank =
RANKX(
    ALL(OWID_Vacc[vaccine]),
    [Sum of Vaccinations],
    ,
    DESC
)
```

2) **Average Daily Doses**
Calculates the average number of daily doses administered for each vaccine or technology.

```
Average Daily Doses =
AVERAGE(OWID_Vacc[Daily Doses])
```

C) Log-Transformed Scaling

Useful for visualization when data is highly skewed.

1) **Log Vaccinations**
Applies a log base 10 transformation to total doses to improve readability and comparative scaling across vaccines.

```
Log Vaccinations =
VAR dosis = [Sum of Vaccinations]
RETURN
IF(
    dosis <= 0,
    BLANK(),  -- evita errores con log(0)
    LOG(dosis, 10)
)
```

1.3 VAERS Dataset data.csv:

A) Averages for Clinical Insights

These measures compute the average of key continuous variables for severity and timing analysis.

1) **Average Age**
Calculates the mean age of patients who reported adverse events.

```
Average Age =
AVERAGE(VAERS_Data[AGE_YRS])
```

2) **Average Days VAX to Symptoms**
Measures the average delay between vaccination and the onset of symptoms.

```
Average Days VAX to Symptoms =
AVERAGE(VAERS_Data[NUMDAYS])
```

3) **Average Days Vax to Death**
Computes the average time from vaccination to reported death.

```
Average Days Vax to Death =
AVERAGE(VAERS_Data[Days from Vaccination to Death])
```

4) **Average Days Onset to Death**
Tracks the average number of days between symptom onset and death.

```
Average Days Onset to Death =
AVERAGE(VAERS_Data[Days from Onset to Death])
```

5) **Average Hospital Days**
Indicates the average duration of hospitalization for reported cases.

```
Average Hospital Days =
AVERAGE(VAERS_Data[HOSPDAYS])
```

B) Death-Related Metrics

These measures count or calculate the proportion of deaths within specific clinical or reporting conditions.

1) **Total Deaths Reported**
Counts all reported deaths in the VAERS dataset.

```
Total Deaths Reported =
CALCULATE(
    COUNTROWS(VAERS_Data),
    VAERS_Data[DIED] = "Y"
)
```

2) **% Deaths Hospitalized**
Percentage of hospitalized patients who died.

```
% Deaths Hospitalized =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_Data),
        VAERS_Data[HOSPITAL] = "Y",
        VAERS_Data[DIED] = "Y"
    ),
    CALCULATE(COUNTROWS(VAERS_Data),
        VAERS_Data[HOSPITAL] = "Y"
    )
)
```

3) **% Deaths with Birth Defect**
Percentage of death cases that also reported a birth defect.

```
% Deaths with Birth Defect =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_Data),
```

```
        VAERS_Data[BIRTH_DEFECT] = "Y",
        VAERS_Data[DIED] = "Y"
    ),
    CALCULATE(COUNTROWS(VAERS_Data),
        VAERS_Data[BIRTH_DEFECT] = "Y"
    )
)
```

4) **% Deaths with ER Visit**
   Percentage of patients who died after visiting the emergency room.

```
% Deaths with ER Visit =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_Data),
        VAERS_Data[ER_VISIT] = "Y",
        VAERS_Data[DIED] = "Y"
    ),
    CALCULATE(COUNTROWS(VAERS_Data),
        VAERS_Data[ER_VISIT] = "Y"
    )
)
```

5) **% Deaths**
   Overall percentage of cases in the dataset that resulted in death.

```
% Deaths =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_Data), VAERS_Data[DIED] = "Y"),
    COUNTROWS(VAERS_Data)
)
```

C) Disability and Hospitalization Rates

These measures indicate the proportion of adverse events that led to long-term disability or hospitalization.

1) **% Disabled**
   Percentage of reports indicating the patient suffered a lasting disability.

```
% Disabled =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_Data), VAERS_Data[DISABLE] = "Y"),
    COUNTROWS(VAERS_Data)
)
```

2) **% Hospitalized**
   Percentage of reported cases that required hospitalization.

```
% Hospitalized =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_Data), VAERS_Data[HOSPITAL] = "Y"),
    COUNTROWS(VAERS_Data)
```

```
)
```

1.4 VAERS Dataset symptoms:

A) Symptom-Based Severity Rates

These measures assess the proportion of cases involving disability, hospitalization, or death for each symptom.

1) **Disability by Symptom (%)**
   Percentage of symptom reports where the patient developed a disability.

```
Disability by Symptom (%) =

DIVIDE(
    CALCULATE(COUNTROWS(VAERS_Symptoms), VAERS_Data[DISABLE] = "Y"),
    COUNTROWS(VAERS_Symptoms)
)
```

2) **Hospitalization by Symptom (%)**
   Percentage of symptom reports that resulted in hospitalization.

```
Hospitalization by Symptom (%) =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_Symptoms), VAERS_Data[HOSPITAL] = "Y"),
    COUNTROWS(VAERS_Symptoms)
)
```

3) **Mortality by Symptom (%)**
   Percentage of symptom reports associated with a fatal outcome.

```
Mortality by Symptom (%) =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_Symptoms),(VAERS_Data[DIED]) = "Y"),
    COUNTROWS(VAERS_Symptoms)
)
```

B) Symptom Counts

Basic measures for counting total symptoms and deaths among those symptoms.

1) **Symptom Count**
   Total number of individual symptom reports.

```
Symptom Count = COUNTROWS(VAERS_Symptoms)
```

2) **Symptom Death Count**
   Number of symptom entries where the patient died.

```
Symptom Death Count =
CALCULATE(
    COUNTROWS(VAERS_Symptoms),
    VAERS_Data[DIED] = "Y"
)
```

1.5 VAERS Dataset vax.csv:

A) Dose Distribution Metrics

Measures to understand how many individuals received one or multiple doses.

1) **Average Number of Doses**
   Average number of doses received by individuals who reported an adverse event.

```
Average Number of Doses =
AVERAGE(
    VAERS_VAX[VAX_DOSE_SERIES])
```

2) **% Only First Dose**
   Percentage of patients who received only the first dose of the vaccine.

```
% Only First Dose =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_VAX), VALUE(VAERS_VAX[VAX_DOSE_SERIES]) = 1),
    COUNTROWS(VAERS_VAX)
)
```

3) **% Up to Second Dose**
   Percentage of individuals who received either one or two doses (no booster).

```
% Up to Second Dose =
DIVIDE(
    CALCULATE(COUNTROWS(VAERS_VAX), VALUE(VAERS_VAX[VAX_DOSE_SERIES]) <= 2),
    COUNTROWS(VAERS_VAX)
)
```

B) Aggregated Total and Log-Transformed Scaling

Basic SUM to quantify the total dose count and used for scaling charts when the data has outliers or large ranges.

1) **Total VAERS Vaccinations**
   Sum of all doses administered as reported in VAERS events.

```
Total VAERS Vaccinations =
```

```
SUM(
    VAERS_VAX[VAX_DOSE_SERIES]
```

2) **Log VAERS Vaccinations**
Applies a log10 transformation to the total doses to improve visualization scale.

```
Log VAERS Vaccinations =
VAR dosis = [Total VAERS Vaccinations]
RETURN
IF(
    dosis <= 0,
    BLANK(),  -- evita errores con log(0)
    LOG(dosis, 10)
)
```