🔹 **Project Objective (My Point of View)**

The goal of this project is to **analyze and compare the financial health of countries over time** using key economic indicators such as:

- GDP
- Domestic Credit (Private Sector)
- Inflation
- Interest Rate

Since raw economic data is spread across multiple tables and years, I transformed the data, normalized it, created a **financial health score**, and then analyzed **ranking, trends, stability, and volatility** of countries.

🔹 **Phase 1: Data Cleaning & Transformation (Unpivoting)**

🔸 **Why this phase?**

The original tables store years as columns (2015–2024).
For proper analysis and visualization, I converted this into a **long format** (year as rows).

✅ **GDP View**

```
CREATE VIEW gdp_view AS

SELECT country_name, country_code, 'gdp' AS indicator, 2015 AS year, `2015` AS value FROM gdp

UNION ALL

SELECT country_name, country_code, 'gdp', 2016, `2016` FROM gdp

...

UNION ALL

SELECT country_name, country_code, 'gdp', 2024, `2024` FROM gdp;
```

📌 **What I did**

- Converted GDP data from wide to long format
- Added an indicator column to identify GDP

✅ **Domestic Credit (Private), Inflation & Interest Rate Views**

I repeated the same unpivot logic for:

- domestic_credit_private
- inflation
- interest_rate

📌 **Why same logic?**
Because all indicators follow the same year-based structure.

🔹 **Phase 2: Creating Master Financial Dataset**

✅ **Finance Master View**

```
CREATE VIEW finance_master AS

SELECT * FROM gdp_view

UNION ALL

SELECT * FROM domestic_credit_private_view

UNION ALL

SELECT * FROM inflation_view

UNION ALL

SELECT * FROM interest_rate_view;
```

📌 **Purpose**

- Combine all indicators into **one unified dataset**
- Makes normalization and scoring easy

✅ **Basic Analysis (Min & Max)**

```
SELECT indicator, MIN(value), MAX(value)

FROM finance_master

GROUP BY indicator;
```

📌 **Why this is important**

- Needed for **normalization**
- Each indicator has different units (GDP %, Inflation %, etc.)

🔹 **Phase 3: Normalization & Financial Scoring**

✅ **Normalized Values**

```
CREATE VIEW finance_normalized AS
SELECT
   fm.country_name,
   fm.country_code,
   fm.indicator,
   fm.year,
   fm.value,
   (fm.value - stats.min_value) /
   (stats.max_value - stats.min_value) AS normalized_value
FROM finance_master fm
JOIN (
   SELECT indicator, MIN(value) min_value, MAX(value) max_value
   FROM finance_master
   GROUP BY indicator
) stats
ON fm.indicator = stats.indicator;
```

📌 **Why normalization?**

- To bring all indicators into **0–1 range**
- Makes fair comparison possible

✅ **Final Score Adjustment**

```
CREATE VIEW finance_normalized_final AS
SELECT
  country_name,
  country_code,
  indicator,
  year,
  value,
  normalized_value,
  CASE
    WHEN indicator IN ('inflation', 'interest_rate')
      THEN 1 - normalized_value
    ELSE normalized_value
  END AS final_score
FROM finance_normalized;
```

📌 **Logic**

- High inflation & interest rate are **bad**
- So I reversed them using 1 - normalized_value

🔹 **Phase 4: Financial Health Score & Interpretation**

✅ **Financial Health Score**

```
CREATE VIEW financial_health_score AS
SELECT
  country_name,
  country_code,
  year,
  SUM(
    CASE
      WHEN indicator = 'gdp' THEN final_score * 0.30
      WHEN indicator = 'domestic_credit_private' THEN final_score * 0.20
      WHEN indicator = 'inflation' THEN final_score * 0.15
      WHEN indicator = 'interest_rate' THEN final_score * 0.15
    END
  ) AS financial_normalized_final,
  GROUP_CONCAT( ... ) AS indicator_status
FROM finance_normalized_final
GROUP BY country_name, country_code, year;
```

📌 **What this does**

- Combines all indicators into **one final score**
- Assigns **economic meaning** like:
  - *GDP: Strong Economy*
  - *Inflation: High*

🔹 **Phase 5: Country Ranking**

✅ **Ranking Countries by Year**

```
CREATE VIEW country_financial_rank AS
SELECT
    country_name,
    country_code,
    indicator_status,
    year,
    financial_normalized_final,
    RANK() OVER (
        PARTITION BY year
        ORDER BY financial_normalized_final DESC
    ) AS financial_rank
FROM financial_health_score;
```

📌 **Why ranking?**

- To compare countries **within the same year**
- Helps identify **top and bottom performers**

🔹 **Phase 6: Trend Analysis (Year-on-Year)**

✅ **Financial Trend**

```
CREATE VIEW country_financial_trend AS
SELECT
    country_name,
    country_code,
    year,
    financial_normalized_final,
    year_change,
    CASE
```

```
        WHEN year_change > 0 THEN 'Economy Improved'
        WHEN year_change < 0 THEN 'Economy Declined'
        WHEN year_change = 0 THEN 'No Change'
        ELSE 'First Year (No Previous Data)'
    END AS result
FROM (
    SELECT
        country_name,
        country_code,
        year,
        financial_normalized_final,
        financial_normalized_final -
        LAG(financial_normalized_final) OVER (
            PARTITION BY country_code
            ORDER BY year
        ) AS year_change
    FROM financial_health_score
) t;
```

📌 **Why this matters**

- Shows **economic improvement or decline**
- Helps identify growth patterns

🔹 **Additional Insights**

🔸 **Country Comparison**

```
SELECT country_name, year, financial_normalized_final
FROM financial_health_score
ORDER BY year, financial_normalized_final DESC;
```

📌 Compare countries within the same year

🔸 **Long-Term Growth**

```
SELECT country_name, SUM(year_change) AS total_change
FROM country_financial_trend
GROUP BY country_name
ORDER BY total_change DESC;
```

📌 Shows which country improved the most overall

🔸 **Stability vs Volatility**

```
SELECT country_name, STDDEV(year_change) AS volatility
FROM country_financial_trend
GROUP BY country_name
ORDER BY volatility DESC;
```

📌 Identifies:

- Stable economies
- Highly volatile economies

✅ **Final Outcome**

Through this project, I:

- Cleaned & transformed raw economic data
- Built a **financial health scoring system**
- Ranked countries
- Analyzed trends, growth, and volatility
- Prepared clean data for **Tableau visualization**