Seattle Air Quality Forecasting

This documentation outlines the process of creating a multivariate time series model (ARIMA_PLUS_XREG) for time-series forecasting using sample tables from the epa_historical_air_quality dataset. The dataset includes daily information on PM 2.5 levels, temperature, and wind speed across various cities in the United States.

Sample Tables Used:

- epa_historical_air_quality.pm25_nonfrm_daily_summary
- epa_historical_air_quality.wind_daily_summary
- epa_historical_air_quality.temperature_daily_summary

Dataset Overview:

The epa_historical_air_quality dataset contains daily records of PM 2.5 concentrations, temperature, and wind speed, which have been collected from different US cities.

Objectives

In this documentation, we will cover the following key steps for building and evaluating a time series model:

- **CREATE MODEL** Statement: Learn how to create a multivariate time series model using ARIMA_PLUS_XREG.
- **ML.ARIMA_EVALUATE** Function: Explore how to inspect ARIMA-related evaluation metrics and model diagnostics.
- **ML.ARIMA_COEFFICIENTS** Function: Review the model coefficients to understand the relationships between variables.
- **ML.FORECAST** Function: Use this function to forecast daily PM 2.5 values.
- **ML.EVALUATE** Function: Evaluate the accuracy of the model by comparing forecasts with actual data.
- **ML.EXPLAIN_FORECAST** Function: Retrieve and interpret various components of the time series (such as seasonality, trend, and feature attributions) to better explain the forecast results.

Each of these steps will guide you in building a robust time series forecasting model using the **ARIMA_PLUS_XREG** approach.

Steps to Set Up the Project for Time Series Forecasting

Step 1: Create a New BigQuery Project and Dataset in Google Cloud Platform (GCP)

1. Log into the Google Cloud Console.
2. Create a new project in GCP and then create a dataset for storing tables and Model.
3. Make sure your dataset permissions are configured for the necessary access.

Step 2: Create a Service Account and Generate a JSON Key

1. In GCP, navigate to IAM & Admin > Service Accounts.
2. Create a new service account for your project and assign appropriate roles (e.g., BigQuery Owner).
3. Generate and download a JSON key for this service account for use in dbt Cloud.

Step 3: Create a New Project in dbt Cloud IDE

1. In the dbt Cloud IDE, create a new project by logging into your dbt Cloud account.
2. During project setup, select BigQuery as your data warehouse.
3. Input the required BigQuery details such as your project ID, dataset, and JSON key information.
4. dbt Cloud IDE will handle most of the configuration and environment setup automatically.

Step 4: Connect Your BigQuery Project and Git Repository to dbt Cloud IDE

1. In the dbt Cloud IDE, configure the connection to your BigQuery project by providing the details of the service account JSON key and BigQuery project you created earlier.
2. Next, link your dbt Cloud project to your Git repository by configuring the repository settings within the IDE.
3. This will allow you to version control your models and configurations directly from dbt Cloud.

Steps to Complete Project Setup and Data Transformation in dbt Cloud

Step 5: Initialize the Project in dbt Cloud

1. After configuring the BigQuery connection in dbt Cloud, initialize your project within the dbt Cloud IDE.
2. dbt Cloud will automatically set up the necessary folders, configurations, and basic model structures for you.

Step 6: Commit and Sync the Project to Git (Initial Commit)

1. In the dbt Cloud IDE, make the first commit with your initialized project structure.
2. Use the commit option in the IDE to sync your dbt project to your connected Git repository.
3. Example commit message: "Initial commit: project setup"

Step 7: Generate a Git Branch

1. To work on your transformations, create a new Git branch within dbt Cloud.
2. You can do this from the IDE or your Git repository.
3. Example branch: feature/data-transformation

Step 8: Generate Views in dbt Cloud IDE for pm25_daily, temperature_daily, wind_speed_daily

1. In the dbt Cloud IDE, create new .sql files for the views representing each dataset (pm25_daily, temperature_daily, and wind_speed_daily).
2. Write SQL queries in each file to define the views using SELECT statements on the corresponding BigQuery tables.
3. Save these views in the appropriate folder under models.

models / pm25_daily.sql

```sql
{{ config(
    materialized='view'
) }}

SELECT
        avg(arithmetic_mean) AS pm25, date_local AS date
    FROM
        `bigquery-public-data.epa_historical_air_quality.pm25_nonfrm_daily_summary`
    WHERE
        city_name = 'Seattle'
        AND parameter_name = 'Acceptable PM2.5 AQI & Speciation Mass'
    GROUP BY date_local
```

models / temperature_daily.sql

```sql
SELECT
        avg(first_max_value) AS temperature, date_local AS date
    FROM
        `bigquery-public-data.epa_historical_air_quality.temperature_daily_summary`
    WHERE
        city_name = 'Seattle' AND parameter_name = 'Outdoor Temperature'
    GROUP BY date_local
```

models / wind_speed_daily.sql

```sql
SELECT
        avg(arithmetic_mean) AS wind_speed, date_local AS date
    FROM
        `bigquery-public-data.epa_historical_air_quality.wind_daily_summary`
    WHERE
        city_name = 'Seattle' AND parameter_name = 'Wind Speed - Resultant'
    GROUP BY date_local
```

Step 9: Transform the Data into a Single Table by Joining Multiple Views

1. Create a new model that joins the data from the views into a single table.
2. Use SQL JOIN statements to merge data from pm25_daily, temperature_daily, and wind_speed_daily views into one consolidated table.

models / seattle_air_quality_daily.sql

```
#standardSQL
{{ config(
    materialized='table'
) }}
SELECT
  P.date AS date, P.pm25, W.wind_speed, T.temperature
FROM {{ ref('pm25_daily')}} as P
JOIN {{ ref('wind_speed_daily')}} as W USING (date)
JOIN {{ ref('temperature_daily')}} as T USING (date)
```

Step 10: Push the Views and Table to BigQuery by Executing dbt run

1. Once the transformations are complete, run the dbt models to push the views and the final table to your BigQuery dataset.
2. In the dbt Cloud IDE, click on the Run button or execute the command dbt run to materialize the views and tables in BigQuery.
3. Verify that the tables and views have been successfully created and populated in your BigQuery dataset.

dbt run success

timeseries-01

about 1 hour ago

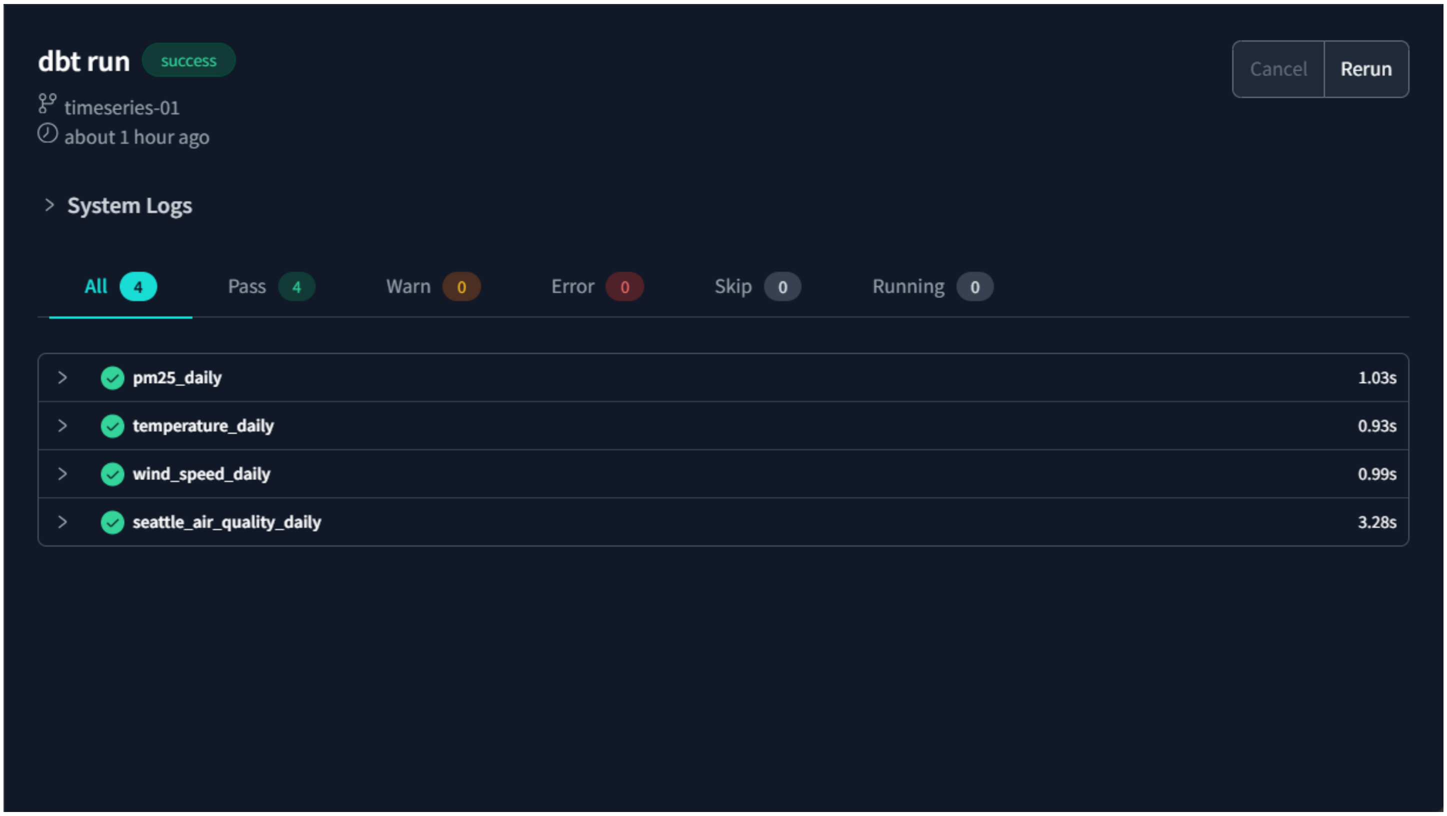
Cancel Rerun

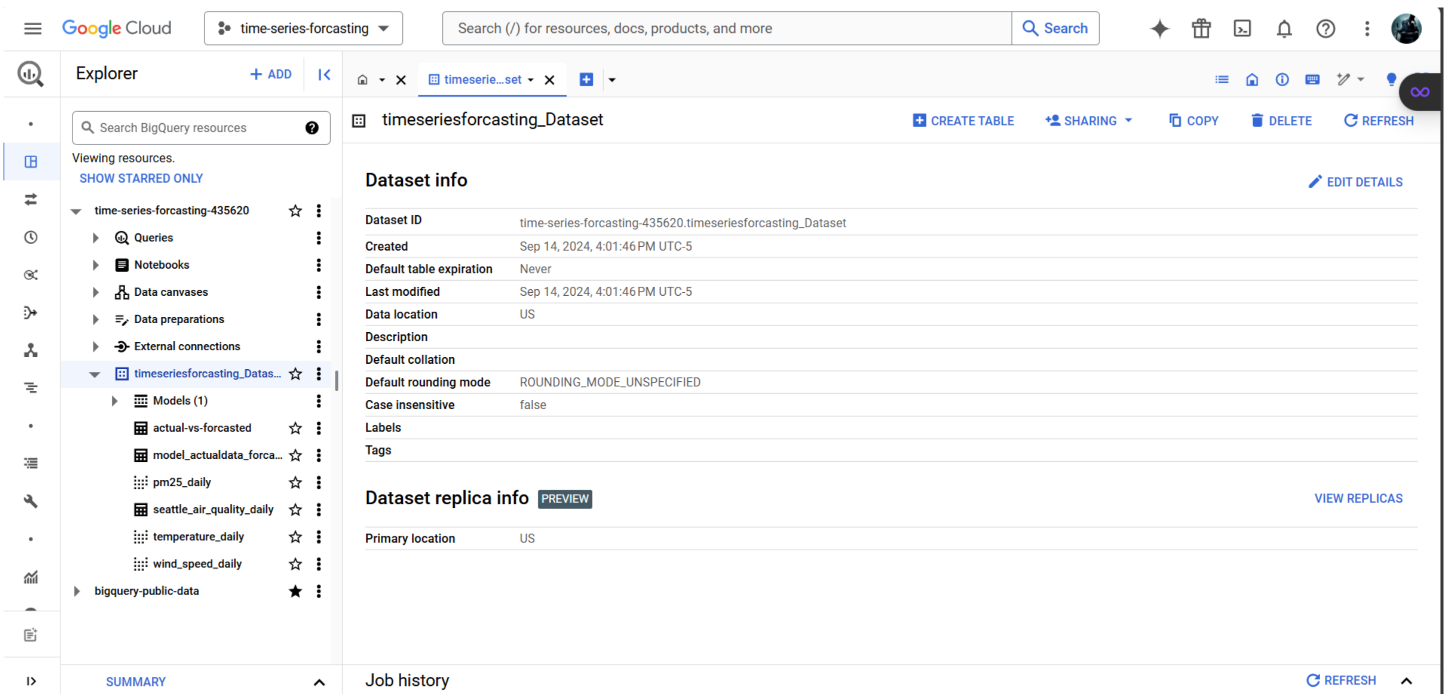
System Logs

Summary Details

```
01:48:02 Running dbt...
01:48:02 [WARNING]: Configuration paths exist in your dbt_project.yml file which do not apply to any resources.
There are 1 unused configuration paths:
- models.my_new_project.example
01:48:02 Found 4 models, 474 macros

01:48:03 Concurrency: 6 threads (target='default')


01:48:07 Finished running 3 view models, 1 table model in 0 hours 0 minutes and 4.90 seconds (4.90s).

01:48:07 Completed successfully

01:48:07 Done. PASS=4 WARN=0 ERROR=0 SKIP=0 TOTAL=4
Running dbt...
```

All 4 Pass 4 Warn 0 Error 0 Skip 0 Running 0

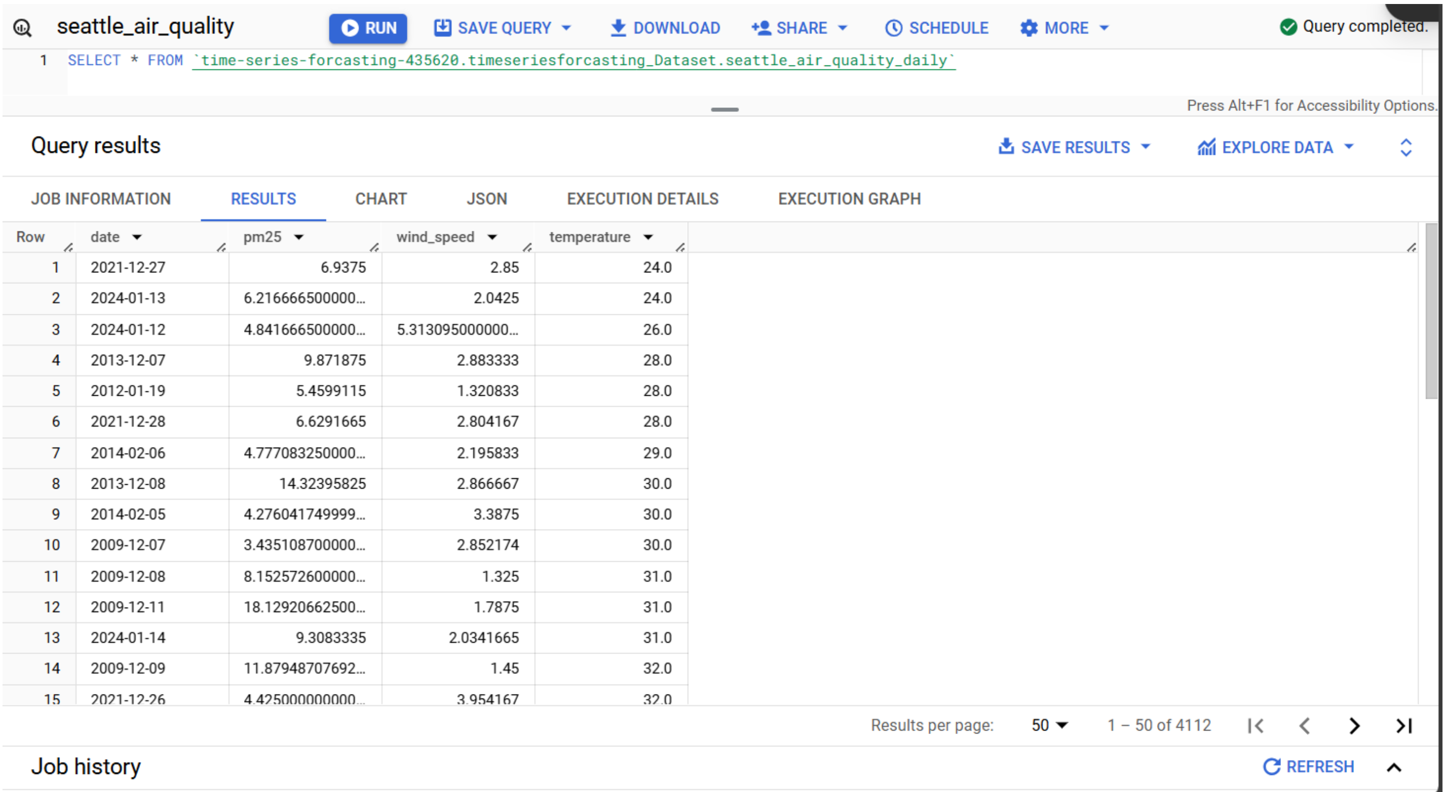




Visualizing Seattle Air Quality Data with Looker Studio

Step 11: Query the Seattle Air Quality Table in BigQuery Query Editor

1. Open the BigQuery Query Editor in Google Cloud Console.
2. Write a SQL query to retrieve air quality data specific to Seattle from the table created in the previous step.
3. Run the query to ensure the correct data is retrieved for analysis.



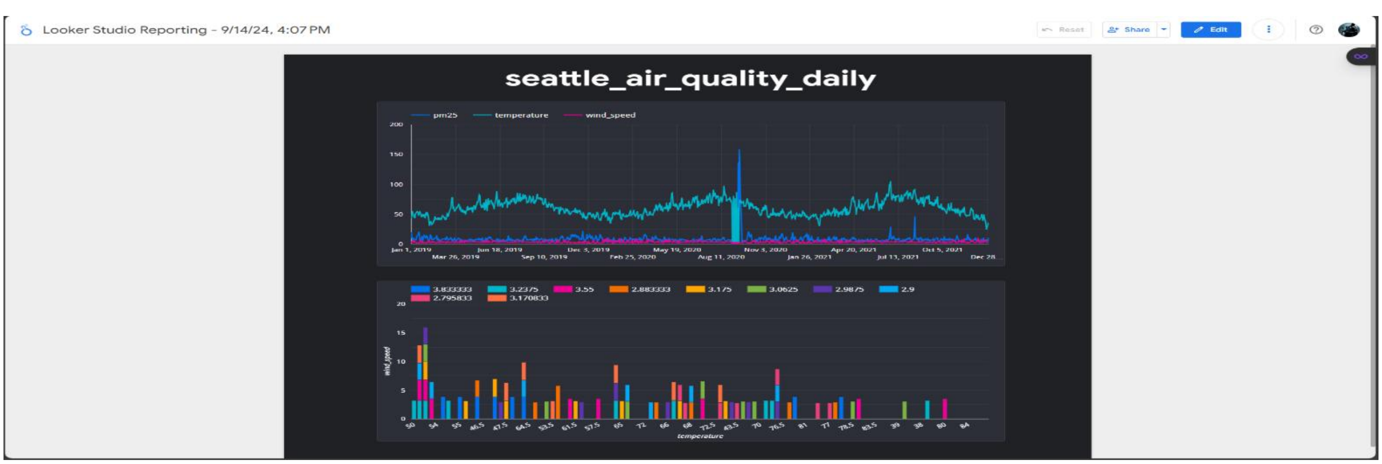
Row	date	pm25	wind_speed	temperature
1	2021-12-27	6.9375	2.85	24.0
2	2024-01-13	6.216666500000...	2.0425	24.0
3	2024-01-12	4.841666500000...	5.313095000000...	26.0
4	2013-12-07	9.871875	2.883333	28.0
5	2012-01-19	5.4599115	1.320833	28.0
6	2021-12-28	6.6291665	2.804167	28.0
7	2014-02-06	4.777083250000...	2.195833	29.0
8	2013-12-08	14.32395825	2.866667	30.0
9	2014-02-05	4.276041749999...	3.3875	30.0
10	2009-12-07	3.435108700000...	2.852174	30.0
11	2009-12-08	8.152572600000...	1.325	31.0
12	2009-12-11	18.12920662500...	1.7875	31.0
13	2024-01-14	9.3083335	2.0341665	31.0
14	2009-12-09	11.87948707692...	1.45	32.0
15	2021-12-26	4.425000000000...	3.954167	32.0

Step 12: Explore Data with Looker Studio

1. In the BigQuery Query Editor results pane, click the Explore data button.
2. Select Explore with Looker Studio. Looker Studio will open in a new tab.
3. Once in Looker Studio, complete the following setup:
4. In the SETUP panel, below the Chart panel, navigate to the Metric section.
5. Add the pm25_value, temp_value, and wind_speed fields as metrics.
6. Remove the default metric, Record Count.
7. Set a custom date range for the analysis. For example, choose the date range from Jan. 1, 2019 to Dec. 31, 2021 to limit the time series for analysis.

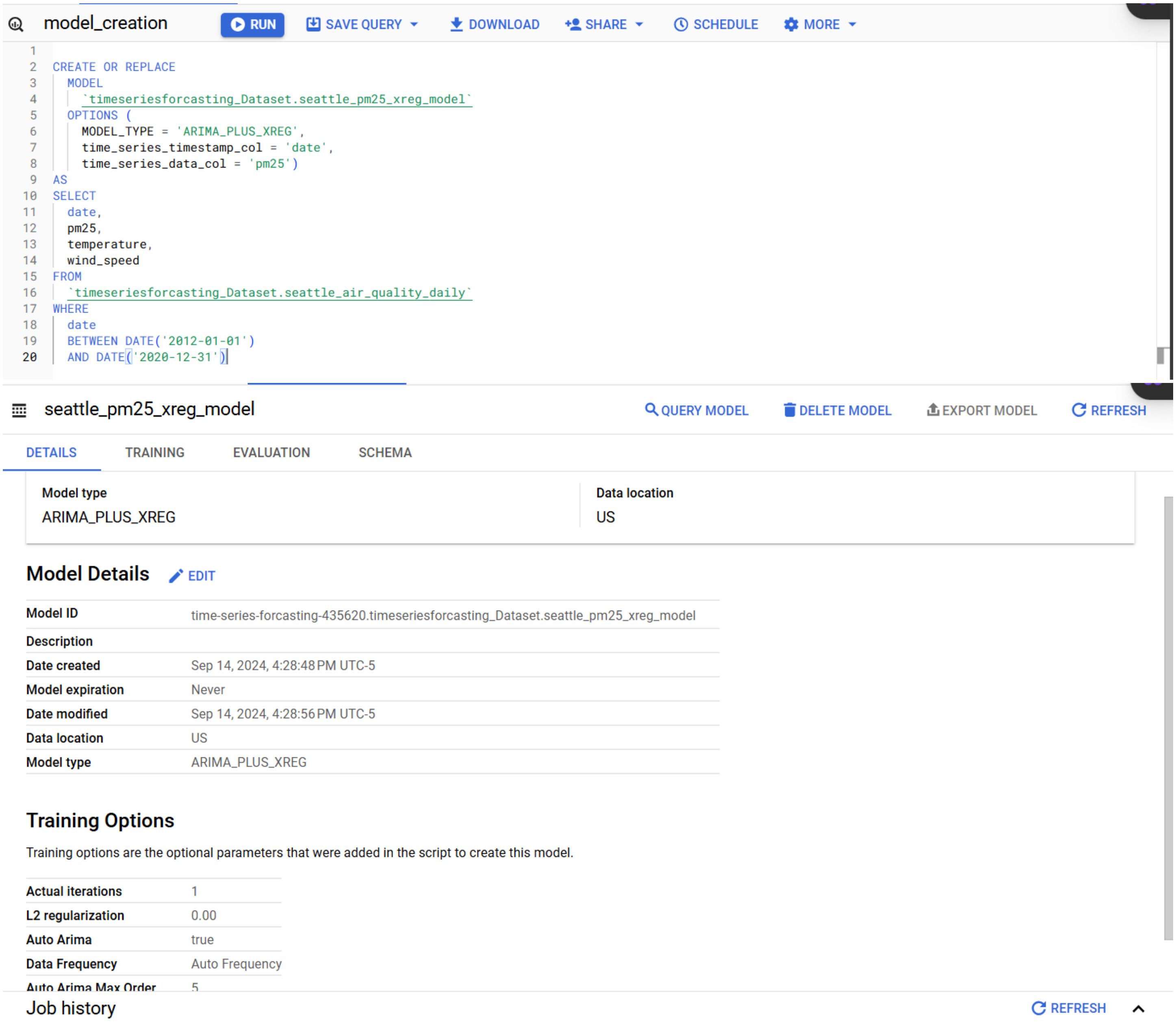
Step 13: Interpret the Plot

1. After completing the setup in Looker Studio, a time series plot will be generated.
2. The plot will show the relationship between PM 2.5 levels, temperature, and wind speed over time.
3. Upon inspection, you should observe a weekly seasonal pattern in the input time series, providing insight into air quality trends in Seattle over the specified period.
4. The visualizations created can now be used to analyze trends and provide further insights into the air quality data.



Step 14: Create Your Time Series Model

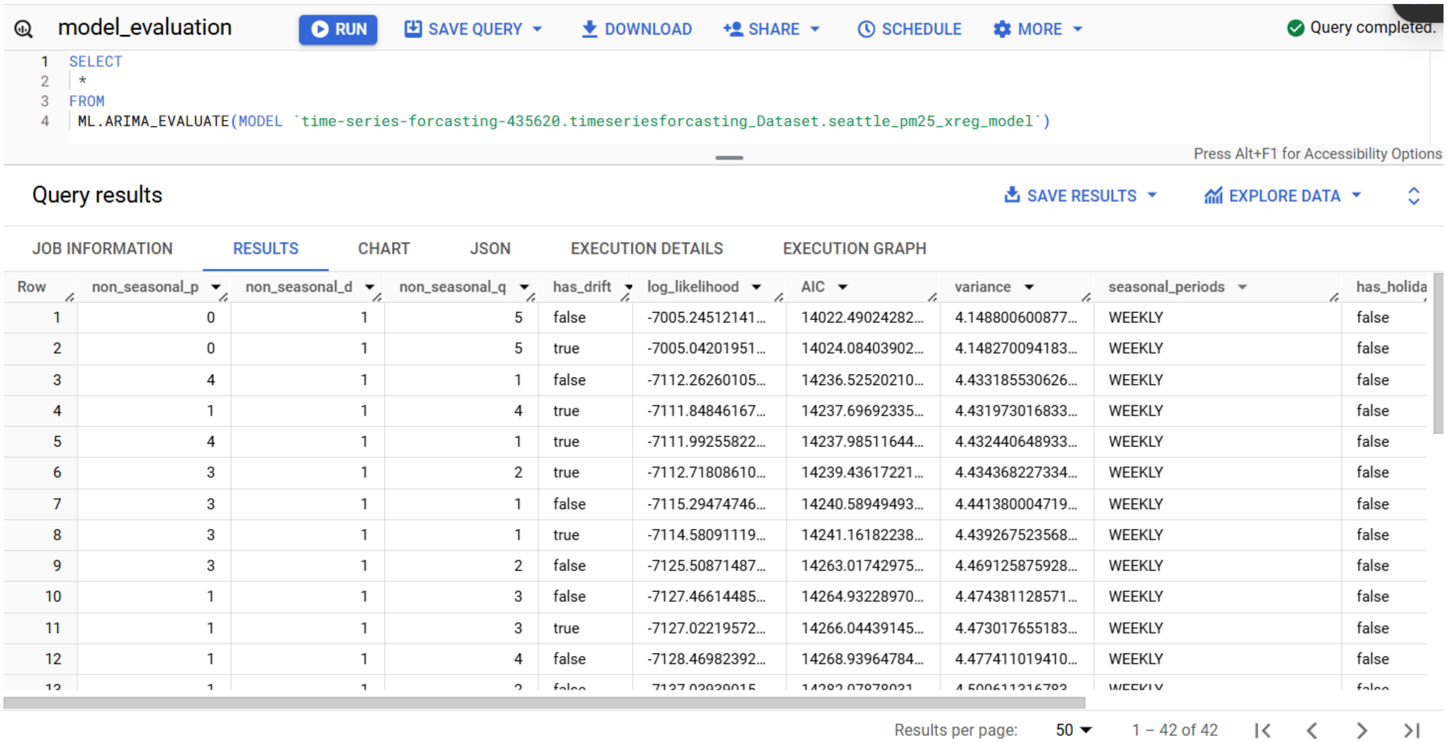
To begin, you'll create a time series model using the Seattle air quality data. Use the following GoogleSQL query to create a model that forecasts PM2.5 levels:



This query creates and trains a model named seattle_pm25_xreg_model using ARIMA with external regressors (ARIMA_PLUS_XREG). By default, auto_arima=TRUE tunes the ARIMA hyperparameters by fitting dozens of models and selecting the best one based on the Akaike Information Criterion (AIC).

Step 15: Inspect the Evaluation Metrics of the Models

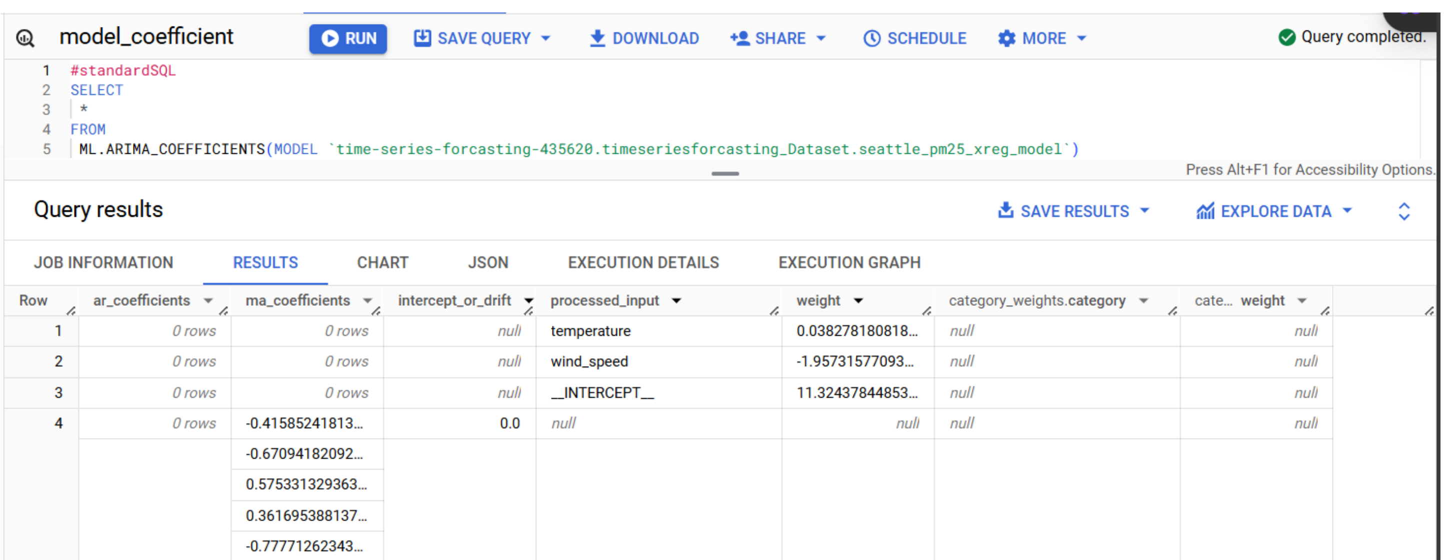
Once the model is created, you can inspect the evaluation metrics of the various models that were evaluated during hyperparameter tuning. Use the following query to see the evaluation results:



After running the query, the results will show various evaluation metrics such as AIC, log_likelihood, variance, and ARIMA model parameters like non_seasonal_p, non_seasonal_d, and non_seasonal_q.

Step 16: Inspect the Coefficients of Your Model

To retrieve the coefficients of your ARIMA model, run the following query:

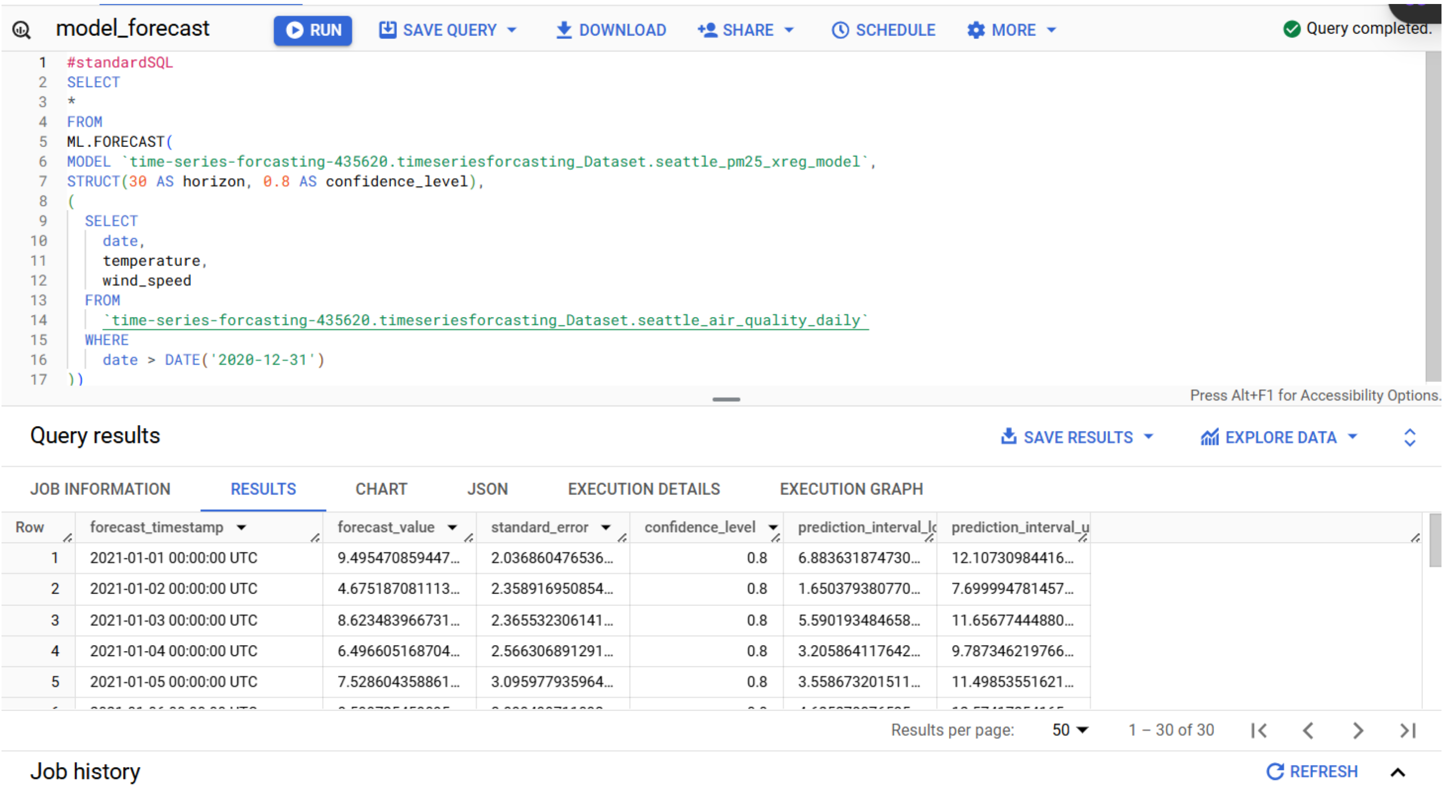


The results include:

1. ar_coefficients: Autoregressive (AR) part coefficients.
2. ma_coefficients: Moving-average (MA) part coefficients.
3. intercept_or_drift: The constant term of the model.

Step 17: Use Your Model to Forecast the Time Series

Next, you can forecast future PM2.5 values using the trained model. Use the following query to forecast 30 future time points with an 80% confidence level:

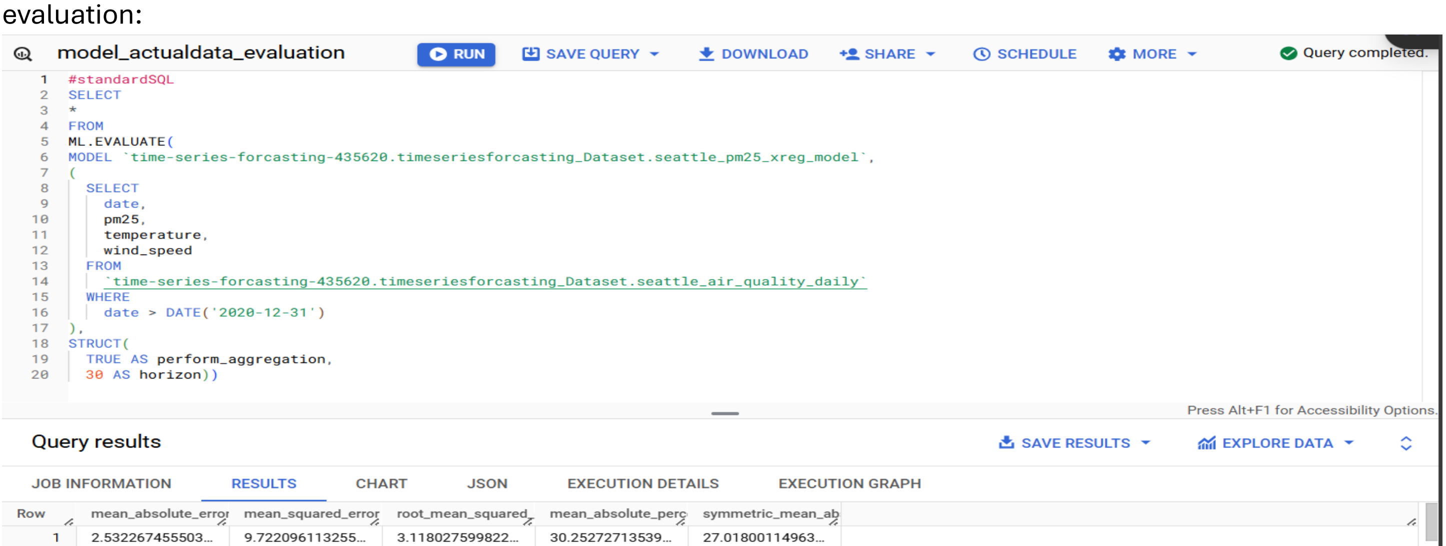


The forecast results include:

1. forecast_timestamp: Predicted future date.
2. forecast_value: Predicted PM2.5 value.
3. prediction_interval_lower_bound: Lower bound of the prediction interval.
4. prediction_interval_upper_bound: Upper bound of the prediction interval.

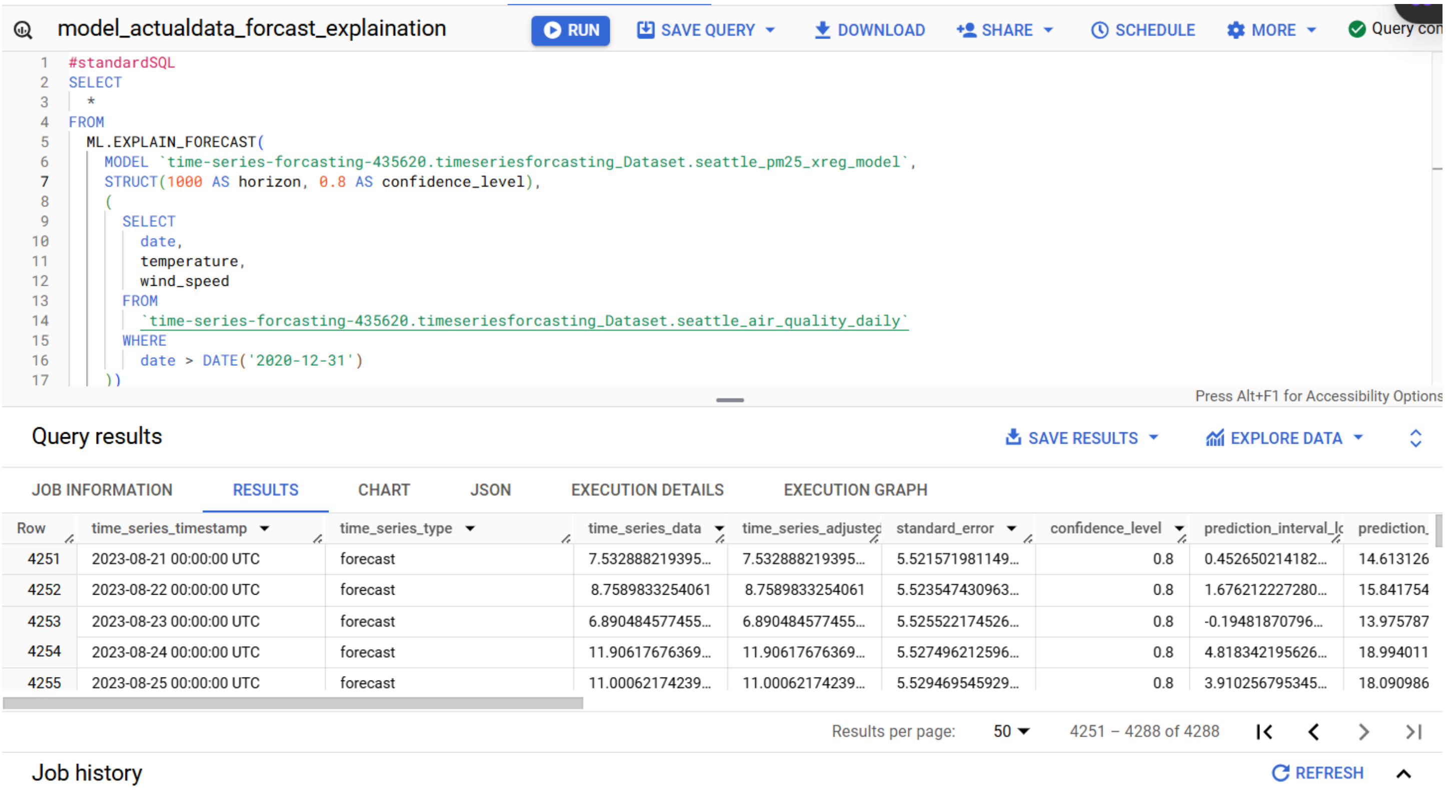
Step 18: Evaluate Forecasting Accuracy with Actual Data

To evaluate how accurate your forecasts are, compare them with actual data. Use this query to perform the evaluation:



Step 19: Explain the Forecasting Results

To understand the components behind the time series forecast, use the ML.EXPLAIN_FORECAST function. This query breaks down the forecast into components like trends, seasonal periods, and residuals:



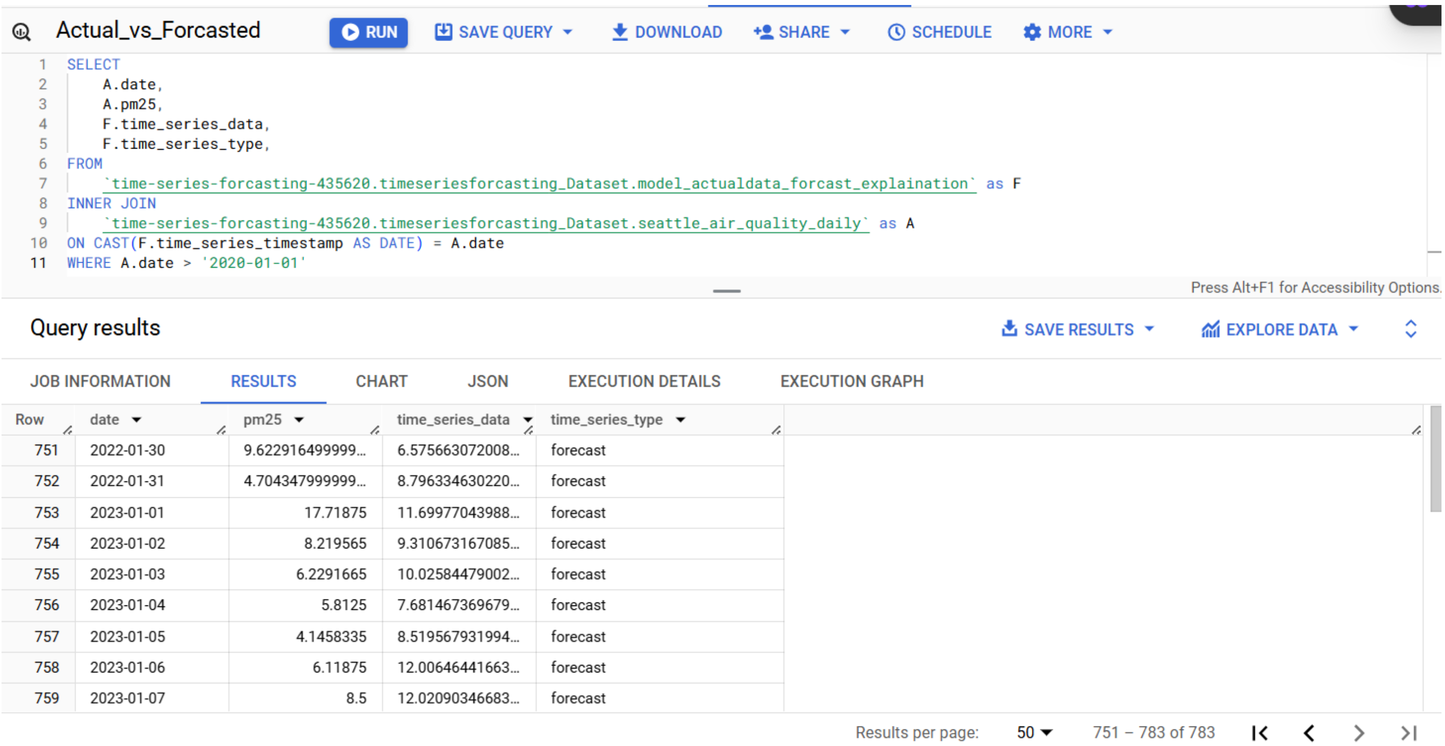
The output includes:

1. trend: The underlying trend in the data.
2. seasonal_period_*: Seasonal patterns at different frequencies (yearly, weekly, daily).
3. spikes_and_dips: Sudden changes in the time series.

Step 20: Compare pm25 from Seattle Air Quality Table with time_series_data from Model Actual Data Forecast Explanation

In this step, we compare the actual pm25 data from the bqml_tutorial.seattle_air_quality_daily table with the time_series_data that we obtained from the ML.EXPLAIN_FORECAST function. This comparison helps evaluate how well the forecasted model aligns with the actual observed data, providing insights into the accuracy of the predictions.

Use the following GoogleSQL query to compare the pm25 values from the air quality table and the time_series_data from the model's actual forecast explanation:




```sql
SELECT
    A.date,
    A.pm25,
    F.time_series_data,
    F.time_series_type,
FROM
    `time-series-forcasting-435620.timeseriesforcasting_Dataset.model_actualdata_forcast_explaination` as F
INNER JOIN
    `time-series-forcasting-435620.timeseriesforcasting_Dataset.seattle_air_quality_daily` as A
ON CAST(F.time_series_timestamp AS DATE) = A.date
WHERE A.date > '2020-01-01'
```

Row	date	pm25	time_series_data	time_series_type
751	2022-01-30	9.622916499999...	6.575663072008...	forecast
752	2022-01-31	4.704347999999...	8.796334630220...	forecast
753	2023-01-01	17.71875	11.69977043988...	forecast
754	2023-01-02	8.219565	9.310673167085...	forecast
755	2023-01-03	6.2291665	10.02584479002...	forecast
756	2023-01-04	5.8125	7.681467369679...	forecast
757	2023-01-05	4.1458335	8.519567931994...	forecast
758	2023-01-06	6.11875	12.00646441663...	forecast
759	2023-01-07	8.5	12.02090346683...	forecast

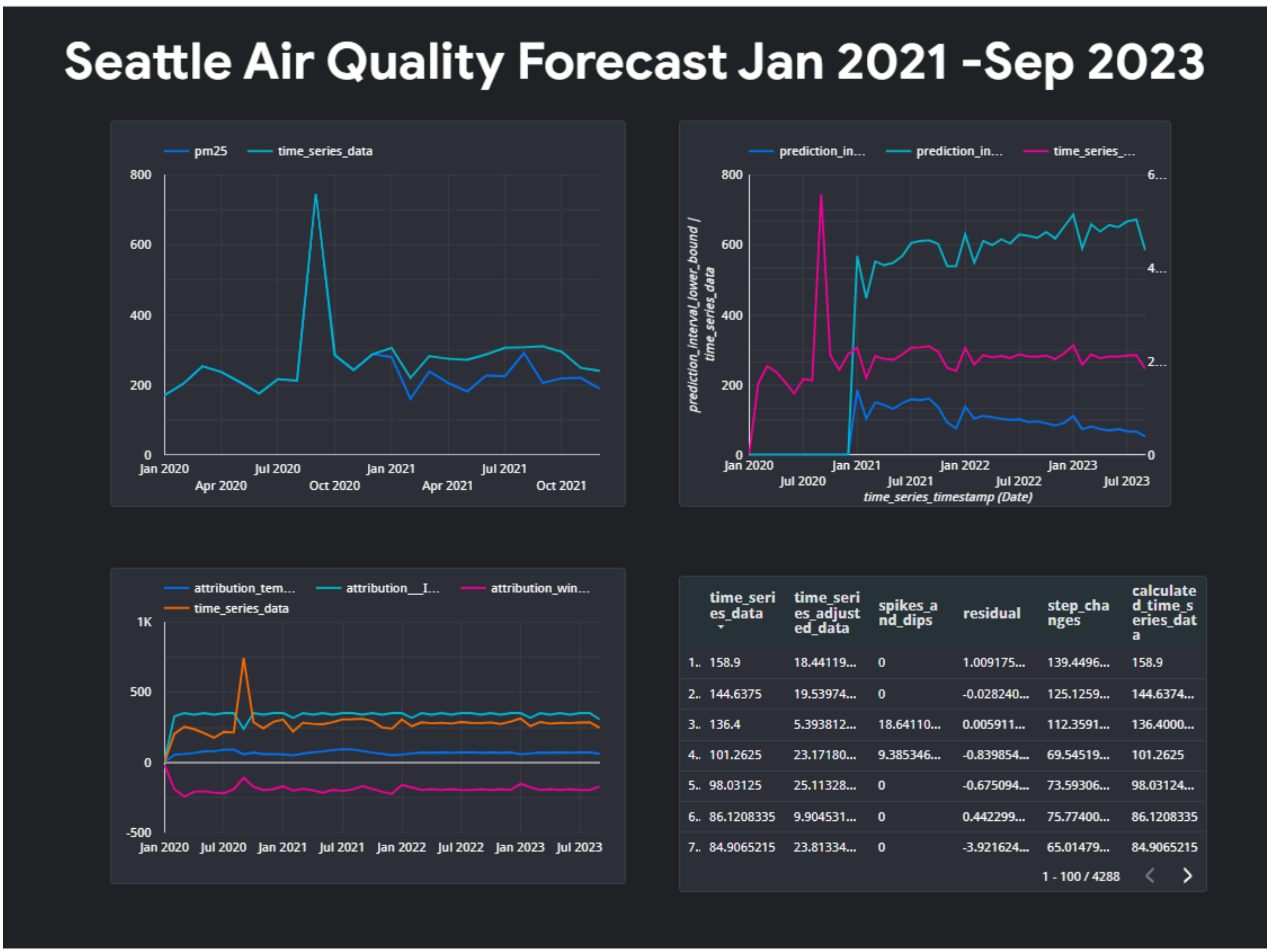
Explore Data with Looker Studio

Step 21: Open Looker Studio:

After querying the Seattle Air Quality table in BigQuery, click the Explore data button.

Select Explore with Looker Studio. Looker Studio will open in a new tab.

1. Set Up Your Data:
    - In Looker Studio, ensure you are using the dataset generated from your BigQuery model.
    - In the SETUP panel, go to the Metric section.
2. Add Metrics:
    - Add the fields for pm25, temperature, and wind_speed.
    - Remove the default metric Record Count to focus on the relevant data.
3. Customize Date Range:
    - Set a custom date range for your data, such as January 1, 2019, to December 31, 2021. This helps in analyzing a specific time period and makes the time series more manageable.
4. Generate Charts:
    - Use Looker Studio to create various charts to visualize the data.



Links:

Git Repository: https://github.com/Dp1997-Cloud/time-series-forcasting

Looker report 1: https://lookerstudio.google.com/reporting/7eb64f89-e097-4a5d-82f3-e92c8a7c9636

Looker Report 2: https://lookerstudio.google.com/reporting/51db085a-d098-4935-a345-990d21d3c6be