

Documentation of Steps – Weather Dataset

This document serves to record the procedures used when working with the dataset *WeatherHistory.csv* in Excel, SQL, and Power BI. The dataset was obtained from Kaggle.com. Each step is described, and some include screenshots. This work serves as a demonstration of my current data-handling skills and as a portfolio example for a junior data analyst position. This document is intended to formally document the procedures employed during work processes.

Step 1: Data Loading

I loaded the dataset *weatherHistory.csv* into Power Query in Excel. The dataset contains 12 columns and 96,453 rows.

Formatted Date	Summary	Precip Type	Apparent Temperature (C)	Humidity	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Load
01.04.2008 00:00:00+02:00	Partly Cloudy	rain	9.47222222222222	7.38888888888889	0.89	14.1287	255.0	15.8263000000000002
01.04.2008 1:00:00+02:00	Partly Cloudy	rain	9.30555555555556	7.22777777777778	0.89	14.2496	255.0	15.8263000000000002
01.04.2008 2:00:00+02:00	Mostly Cloudy	rain	9.37777777777778	9.37777777777778	0.89	14.3398	255.0	14.9569
01.04.2008 3:00:00+02:00	Partly Cloudy	rain	8.38888888888889	5.94444444444444	0.83	14.1036	269.0	15.8263000000000002
01.04.2008 4:00:00+02:00	Mostly Cloudy	rain	8.79555555555556	6.97777777777778	0.83	11.0446	279.0	15.8263000000000002
01.04.2008 5:00:00+02:00	Partly Cloudy	rain	9.22222222222222	7.11111111111111	0.85	11.9587	254.0	14.9569
01.04.2008 6:00:00+02:00	Partly Cloudy	rain	7.73333333333333	5.52222222222222	0.95	12.3649	259.0	9.9620000000000001
01.04.2008 7:00:00+02:00	Partly Cloudy	rain	8.77222222222222	6.52777777777778	0.89	14.1219	260.0	9.9620000000000001
01.04.2008 8:00:00+02:00	Partly Cloudy	rain	10.8222222222222	10.8222222222222	0.82	11.3183	259.0	9.9620000000000001
01.04.2008 9:00:00+02:00	Partly Cloudy	rain	11.7722222222222	11.7722222222222	0.72	12.5280000000000002	279.0	9.9620000000000001
01.04.2008 10:00:00+02:00	Partly Cloudy	rain	18.01666666666666	18.01666666666666	0.67	17.5851	290.0	11.2096
01.04.2008 11:00:00+02:00	Partly Cloudy	rain	17.14444444444444	17.14444444444444	0.54	18.7869	316.0	11.4471
01.04.2008 12:00:00+02:00	Partly Cloudy	rain	17.80000000000000	17.80000000000000	0.55	21.9440000000000002	281.0	11.2700000000000001
01.04.2008 13:00:00+02:00	Partly Cloudy	rain	17.33333333333333	17.33333333333333	0.51	20.6885	289.0	11.2700000000000001
01.04.2008 14:00:00+02:00	Partly Cloudy	rain	18.8777777777778	18.8777777777778	0.67	15.3790000000000002	262.0	11.4471
01.04.2008 15:00:00+02:00	Partly Cloudy	rain	18.9111111111111	18.9111111111111	0.46	20.4006	286.0	11.2700000000000001
01.04.2008 16:00:00+02:00	Partly Cloudy	rain	15.3888888888889	15.3888888888889	0.6	14.4095	251.0	11.2700000000000001
01.04.2008 17:00:00+02:00	Mostly Cloudy	rain	15.50000000000000	15.50000000000000	0.63	11.1570000000000001	230.0	11.4471
01.04.2008 18:00:00+02:00	Mostly Cloudy	rain	14.20000000000000	14.20000000000000	0.49	8.5168	161.0	11.2096
01.04.2008 19:00:00+02:00	Mostly Cloudy	rain	13.14444444444444	13.14444444444444	0.7	7.9151000000000001	139.0	11.2096
01.04.2008 20:00:00+02:00	Mostly Cloudy	rain	11.5499999999999	11.5499999999999	0.77	7.3899	147.0	11.0289
01.04.2008 21:00:00+02:00	Mostly Cloudy	rain	11.38333333333334	11.38333333333334	0.76	6.8266000000000005	160.0	9.9620000000000001
01.04.2008 22:00:00+02:00	Partly Cloudy	rain	10.11666666666667	10.11666666666667	0.79	6.1493	161.0	15.8263000000000002
01.04.2008 23:00:00+02:00	Mostly Cloudy	rain	10.2	10.2	0.77	5.9386000000000003	151.0	14.9569
01.04.2008 00:00:00+02:00	Partly Cloudy	rain	10.4222222222222	10.4222222222222	0.61	18.9810000000000002	150.0	15.8263000000000002
01.04.2008 1:00:00+02:00	Partly Cloudy	rain	9.91111111111111	7.94444444444444	0.66	17.1209	149.0	15.8263000000000002
01.04.2008 2:00:00+02:00	Mostly Cloudy	rain	11.38333333333334	11.38333333333334	0.8	20.8202	161.0	14.9569
01.04.2008 3:00:00+02:00	Partly Cloudy	rain	7.35555555555556	5.04444444444444	0.79	11.0768	180.0	15.8263000000000002
01.04.2008 4:00:00+02:00	Partly Cloudy	rain	6.11111111111111	4.81666666666667	0.82	6.6033	161.0	15.8263000000000002
01.04.2008 5:00:00+02:00	Partly Cloudy	rain	6.78888888888889	4.77222222222222	0.83	11.0768	151.0	14.9569
01.04.2008 6:00:00+02:00	Mostly Cloudy	rain	7.26111111111111	5.35555555555556	0.85	11.1794	141.0	6.1985
01.04.2008 7:00:00+02:00	Mostly Cloudy	rain	7.79999999999999	5.52777777777778	0.81	12.8226	150.0	8.25
01.04.2008 8:00:00+02:00	Mostly Cloudy	rain	9.87222222222225	7.93333333333334	0.78	11.7494	160.0	9.9620000000000001
01.04.2008 9:00:00+02:00	Mostly Cloudy	rain	12.2222222222222	12.2222222222222	0.72	20.4310000000000002	190.0	9.9620000000000001
01.04.2008 10:00:00+02:00	Mostly Cloudy	rain	15.0844444444444	15.0844444444444	0.61	17.5490000000000003	151.0	9.9620000000000001
01.04.2008 11:00:00+02:00	Mostly Cloudy	rain	17.35555555555556	17.35555555555556	0.52	22.7825	169.0	9.9620000000000001
01.04.2008 12:00:00+02:00	Mostly Cloudy	rain	18.80000000000000	18.80000000000000	0.49	21.8914000000000002	169.0	9.9620000000000001
01.04.2008 13:00:00+02:00	Mostly Cloudy	rain	20.04444444444444	20.04444444444444	0.4	20.3682	170.0	9.9620000000000001
01.04.2008 14:00:00+02:00	Mostly Cloudy	rain	21.0499999999999	21.0499999999999	0.4	20.9010000000000002	187.0	10.1521
01.04.2008 15:00:00+02:00	Mostly Cloudy	rain	21.3883333333333	21.3883333333333	0.37	25.8990000000000002	179.0	9.9620000000000001
01.04.2008 16:00:00+02:00	Mostly Cloudy	rain	20.11666666666667	20.11666666666667	0.4	25.8290000000000004	140.0	9.9620000000000001

Fig. 1: Loaded WeatherHistory dataset in Power Query (Excel)

The column names are as follows: *Formatted Date*, *Summary*, *Precip Type*, *Temperature (°C)*, *Apparent Temperature (°C)*, *Humidity*, *Wind Speed (km/h)*, *Wind Bearing (degrees)*, *Visibility (km)*, *Loud Cover*, *Pressure (millibars)*, *Daily Summary*.

Step 2: Setting Data Types

After checking the data types in individual columns, I found that most of them were set incorrectly. Therefore, I adjusted the data types — I set the date column to *Date/Time*, numeric columns to *Decimal Number* or *Int64*, and descriptive text columns to *Text*.

While converting numeric values, I encountered an error caused by different decimal separators. To ensure proper conversion, I used the *Using Locale* function and selected *English (United States)*. This allowed the data to load correctly and without errors.

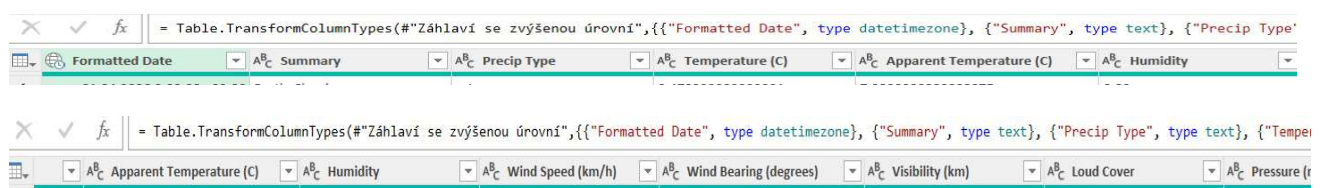


Fig. 2: Example of selected data types after loading the file

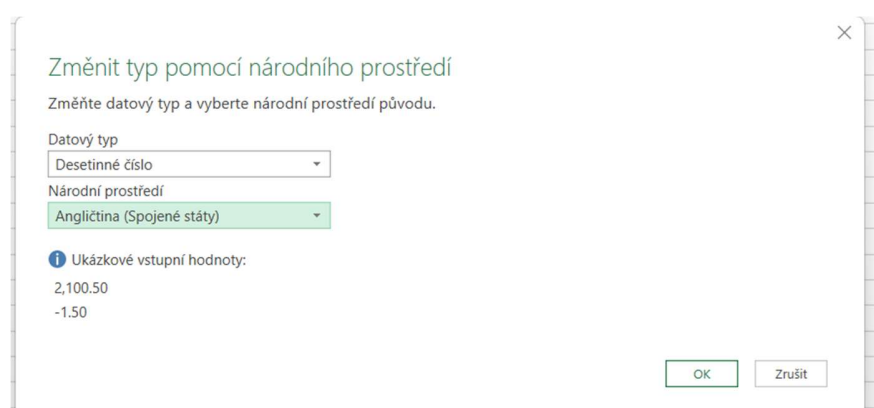


Fig. 3: Changing the locale for correct data type settings

Formatted Date	A ^B _C Summary	A ^B _C Precip Type	1.2 Temperature (C)	1.2 Apparent Temperature (C)	1.2 Humidity
A ^B _C Precip Type	1.2 Temperature (C)	1.2 Apparent Temperature (C)	1.2 Humidity	1.2 Wind Speed (km/h)	1 ² ₃ Wind Bearing (degrees)

Fig. 4: Example of selected data types after adjustment

Step 3: Data Cleaning

3.1 Removing Duplicate Values

The dataset contained 24 duplicate records, corresponding to 48 rows in total (the original record plus its duplicate). I identified the duplicates using the *Keep Duplicates* function and then removed them. The final dataset contains 96,429 rows.

	Formatted Date	A ^B _C Summary	A ^B _C Precip Type
1	02.08.2010 16:00:00	Partly Cloudy	rain
2	02.08.2010 17:00:00	Partly Cloudy	rain
3	02.08.2010 15:00:00	Partly Cloudy	rain
4	02.08.2010 18:00:00	Partly Cloudy	rain
5	02.08.2010 14:00:00	Partly Cloudy	rain
6	02.08.2010 13:00:00	Partly Cloudy	rain
7	02.08.2010 12:00:00	Clear	rain
8	02.08.2010 19:00:00	Clear	rain
9	02.08.2010 11:00:00	Clear	rain
10	02.08.2010 10:00:00	Clear	rain
11	02.08.2010 9:00:00	Clear	rain
12	02.08.2010 20:00:00	Clear	rain
13	02.08.2010 8:00:00	Clear	rain
14	02.08.2010 21:00:00	Clear	rain
15	02.08.2010 22:00:00	Partly Cloudy	rain
16	02.08.2010 7:00:00	Clear	rain
17	02.08.2010 23:00:00	Clear	rain
18	02.08.2010 0:00:00	Clear	rain
19	02.08.2010 1:00:00	Clear	rain
20	02.08.2010 2:00:00	Clear	rain
21	02.08.2010 6:00:00	Clear	rain
22	02.08.2010 3:00:00	Clear	rain
23	02.08.2010 4:00:00	Clear	rain
24	02.08.2010 5:00:00	Clear	rain

Table 1: List of 24 Duplicate Rows

3.2 Checking Value Ranges

I checked the value ranges and missing values in each column. The data in the columns generally make sense — they fall within the expected ranges and do not contain negative values where they shouldn't.

However, in the Pressure column, I found unrealistic values where pressure was recorded as 0 in 1,288 rows. I decided to replace these values with null to prevent their inclusion in calculations.

In the Loud Cover column, all values are 0.

Out of the 12 columns, 8 contain numeric values — their minimum and maximum values are summarized in the tables below.

	ABC 123	Column	ABC 123	Minimum	ABC 123	Maximum
1		Temperature (C)		-21,82222222		39,90555556
2		Apparent Temperature (C)		-27,71666667		39,34444444
3		Humidity		0		1
4		Wind Speed (km/h)		0		63,8526
5		Wind Bearing (degrees)		0		359
6		Visibility (km)		0		16,1
7		Loud Cover		0		0
8		Pressure (millibars)		0		1046,38

Table 2: Value ranges in numeric columns before correcting unrealistic values in the Pressure column.

	ABC 123	Column	ABC 123	Minimum	ABC 123	Maximum
1		Temperature (C)		-21,82222222		39,90555556
2		Apparent Temperature (C)		-27,71666667		39,34444444
3		Humidity		0		1
4		Wind Speed (km/h)		0		63,8526
5		Wind Bearing (degrees)		0		359
6		Visibility (km)		0		16,1
7		Loud Cover		0		0
8		Pressure (millibars)		973,78		1046,38

Table 3: Final table of value ranges in numeric columns.

The dataset also contains three text columns (*Summary*, *Daily Summary*, and *Precip Type*). The number of unique categories in each column is summarized in the table below. In the *Precip Type* column, there are three possible values — *snow*, *rain*, and *null*. The handling of *null* values in this column is discussed in the next section.

	ABC 123 Column	ABC 123 DistinctCount
1	Summary	27
2	Daily Summary	214
3	Precip Type	3

Table 4: Summary of the number of categories in the text columns

The last column, *Formatted Date*, contains date values — the date ranges are summarized in the table below. From the data, it is evident that data collection took place over a period of 10 years.

	ABC 123 Column	ABC 123 Minimum	ABC 123 Maximum
1	Formatted Date	01.01.2006 0:00:00	31.12.2016 23:00:00

Table 5: Summary of date ranges in the *Formatted Date* column

3.3 Checking Missing Values

I found that the dataset is fairly complete. *Null* values appeared only in the *Precip Type* column, where 517 *null* entries were identified. Since this column records the type of precipitation, I assumed that these *null* values indicate that no precipitation occurred on that day.

Therefore, I replaced the *null* values with *no precipitation*. After this adjustment, the column contains three possible values — *snow*, *rain*, and *no precipitation*

```
e.SelectRows("#Změněný typ", each true)
```

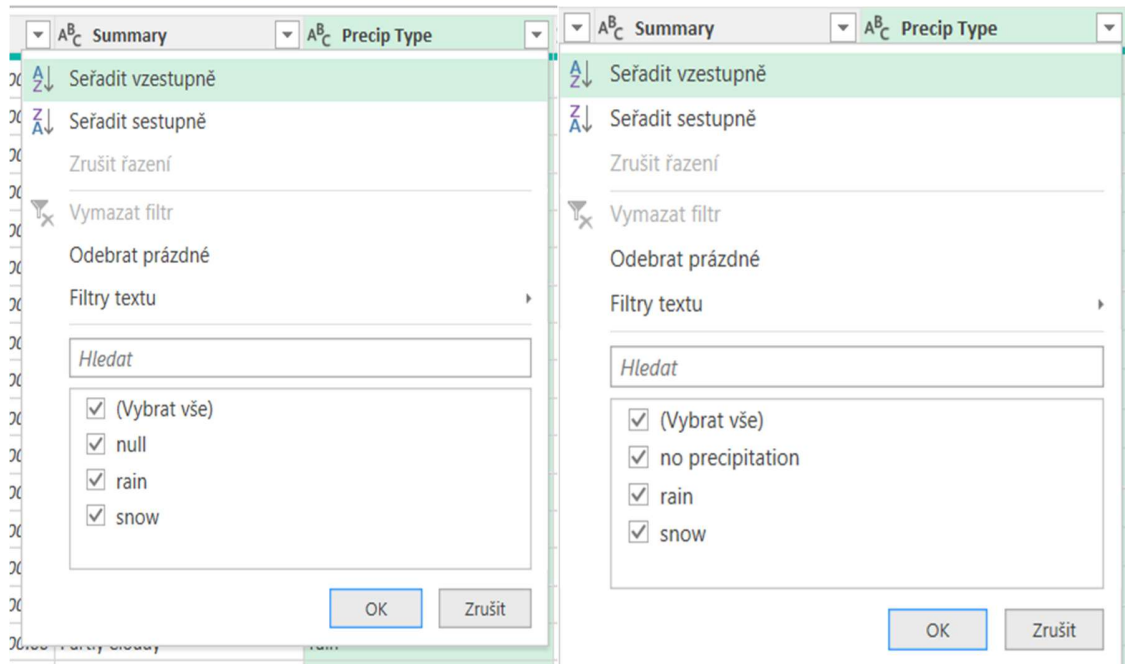


Table 6: Replaced null values in the Precip Type column

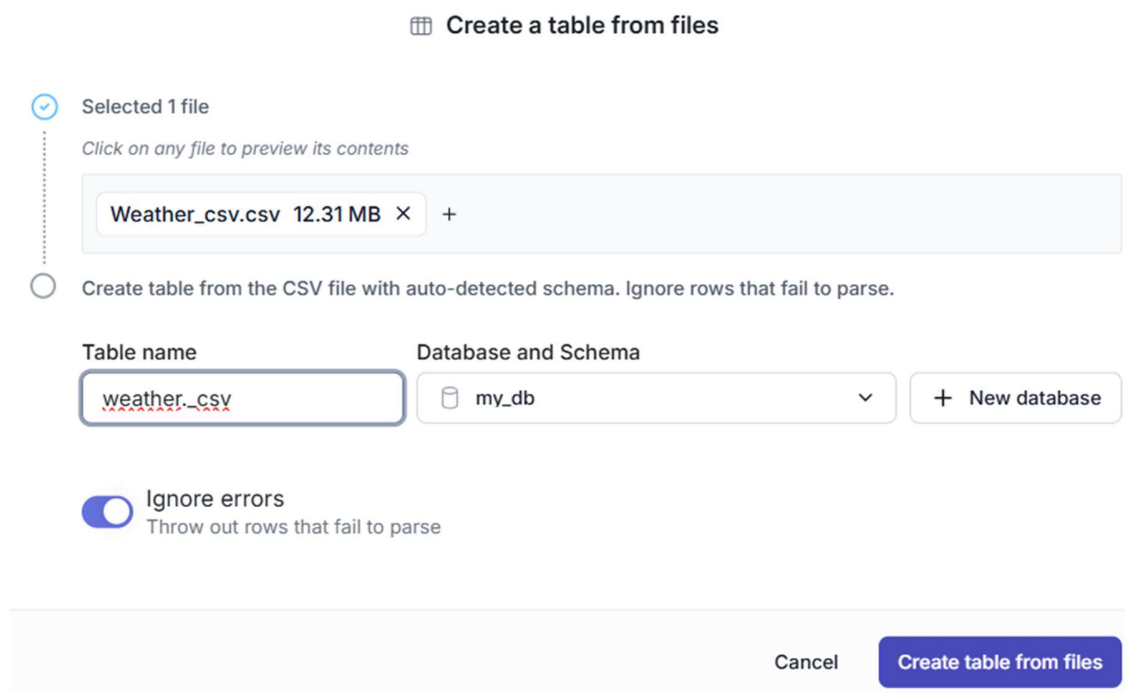
The dataset is now cleaned and verified in terms of data types, duplicates, missing and unrealistic values, as well as categorical variables. It is ready to be saved and further processed in SQL or Power BI. The file was saved in CSV format.

Step 4: SQL Analysis (MotherDuck)

After cleaning and exporting the dataset from Excel, I imported it into the MotherDuck SQL environment. The purpose of this step was to explore the data using SQL queries and prepare analytical summaries that were later visualized in Power BI.

4.1 Loading the Dataset

The cleaned dataset was uploaded to MotherDuck in .csv format.



The screenshot shows the 'Create a table from files' interface in MotherDuck. At the top, there's a header 'Create a table from files' with a grid icon. Below it, a status bar indicates 'Selected 1 file' and provides a link to 'Click on any file to preview its contents'. A file named 'Weather_csv.csv' (12.31 MB) is shown with a close button and a plus sign to add more files. Below the file list, there are two radio buttons: the first is selected and labeled 'Create table from the CSV file with auto-detected schema. Ignore rows that fail to parse.' Below this, there are two input fields: 'Table name' with the value 'weather_csv' and 'Database and Schema' with a dropdown menu showing 'my_db'. To the right of the dropdown is a '+ New database' button. Below these fields is a toggle switch for 'Ignore errors' which is turned on, with the text 'Throw out rows that fail to parse' underneath. At the bottom right, there are two buttons: 'Cancel' and 'Create table from files'.

Fig. 5: Uploading the weather_csv file to MotherDuck.

4.2 Checking Table Structure

To confirm column names and data types, I used:

```
PRAGMA table_info('weather_csv');
```

The table contained **12 columns**, consistent with the original dataset.



```
1 PRAGMA table_info('weather_csv');
```

12 rows returned in 40ms, queued for 37ms

	cid	T name	T type	T _F notnull	T dflt_value	T _F pk
1	0	Formatted Date	VARCHAR	false	NULL	false
2	1	Summary	VARCHAR	false	NULL	false
3	2	Precip Type	VARCHAR	false	NULL	false
4	3	Temperature (C)	VARCHAR	false	NULL	false
5	4	Apparent Temperature (C)	VARCHAR	false	NULL	false
6	5	Humidity	VARCHAR	false	NULL	false
7	6	Wind Speed (km/h)	VARCHAR	false	NULL	false
8	7	Wind Bearing (degrees)	BIGINT	false	NULL	false
9	8	Visibility (km)	VARCHAR	false	NULL	false
10	9	Loud Cover	BIGINT	false	NULL	false
11	10	Pressure (millibars)	VARCHAR	false	NULL	false
12	11	Daily Summary	VARCHAR	false	NULL	false

Fig. 6: Information about the **weather_csv** table after it was uploaded

Using the **PRAGMA** command, I identified that most columns were imported with incorrect data types. Additionally, decimal commas were used instead of decimal points. I resolved these issues by applying the **CAST** command to set the correct data types and using the **REPLACE()** function to substitute commas with dots.

To convert date and time information stored as text into a proper timestamp format, I used the **STRPTIME()** function. This function parses a string according to a specified date-time format (in this case '%d.%m.%Y %H:%M') and returns a SQL-compatible **TIMESTAMP** value.


```

1 CREATE OR REPLACE TABLE weather_clean AS
2 SELECT
3     STRPTIME("Formatted Date", '%d.%m.%Y %H:%M') AS datetime,
4     "Summary",
5     "Precip Type",
6     "Wind Bearing (degrees)",
7     CAST(REPLACE("Temperature (C)", ',', '.') AS DOUBLE) AS temperature_c,
8     CAST(REPLACE("Apparent Temperature (C)", ',', '.') AS DOUBLE) AS apparent_temp_c,
9     CAST(REPLACE("Humidity", ',', '.') AS DOUBLE) AS humidity,
10    CAST(REPLACE("Wind Speed (km/h)", ',', '.') AS DOUBLE) AS wind_speed_kmh,
11    CAST(REPLACE("Visibility (km)", ',', '.') AS DOUBLE) AS visibility_km,
12    CAST(REPLACE("Pressure (millibars)", ',', '.') AS DOUBLE) AS pressure_mb,
13    "Loud Cover",
14    "Daily Summary"
15 FROM weather_csv;

```

Fig. 7: Query for adjusting data types in the **weather_csv** dataset.

After verifying the updated data types with the **PRAGMA** command, I confirmed that all conversions were correct.

```

1 PRAGMA table_info('weather_clean');

```

12 rows returned in 44ms, queued for 62ms

	123 cid	T name	T type	T _F notnull	T dflt_value	T _F pk
1	0	datetime	TIMESTAMP	false	NULL	false
2	1	Summary	VARCHAR	false	NULL	false
3	2	Precip Type	VARCHAR	false	NULL	false
4	3	Wind Bearing (degrees)	BIGINT	false	NULL	false
5	4	temperature_c	DOUBLE	false	NULL	false
6	5	apparent_temp_c	DOUBLE	false	NULL	false
7	6	humidity	DOUBLE	false	NULL	false
8	7	wind_speed_kmh	DOUBLE	false	NULL	false
9	8	visibility_km	DOUBLE	false	NULL	false
10	9	pressure_mb	DOUBLE	false	NULL	false
11	10	Loud Cover	BIGINT	false	NULL	false
12	11	Daily Summary	VARCHAR	false	NULL	false

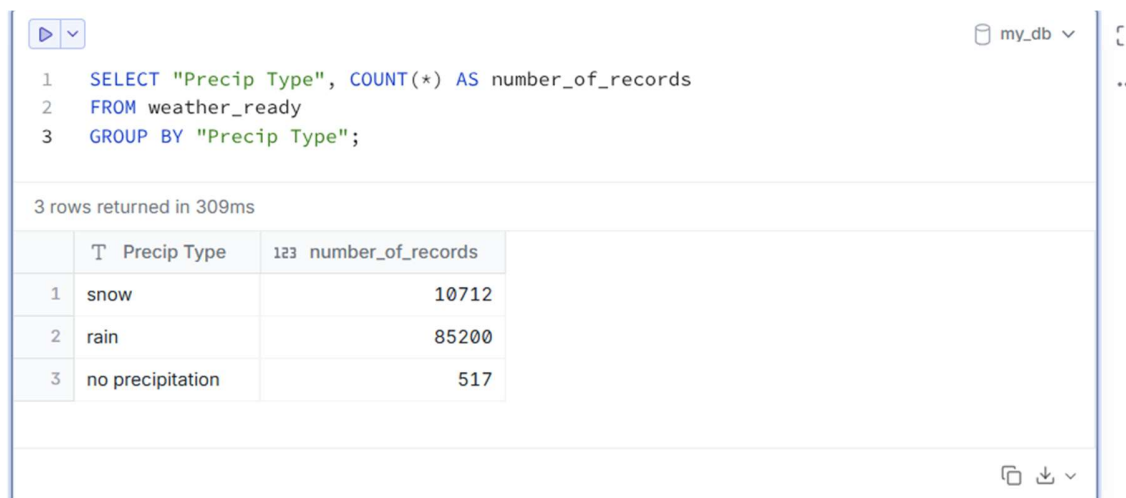
Fig. 8: Table information for **weather_clean** after data type adjustments

4.3 Exploratory Queries

I created a series of 10 analytical SQL queries, each focusing on a specific question related to weather behavior. These queries were used to prepare data summaries for visualization in Power BI.

Query 1 — Number of Records by Precipitation Type

Counts the number of weather records for each type of precipitation.



```
1 SELECT "Precip Type", COUNT(*) AS number_of_records
2 FROM weather_ready
3 GROUP BY "Precip Type";
```

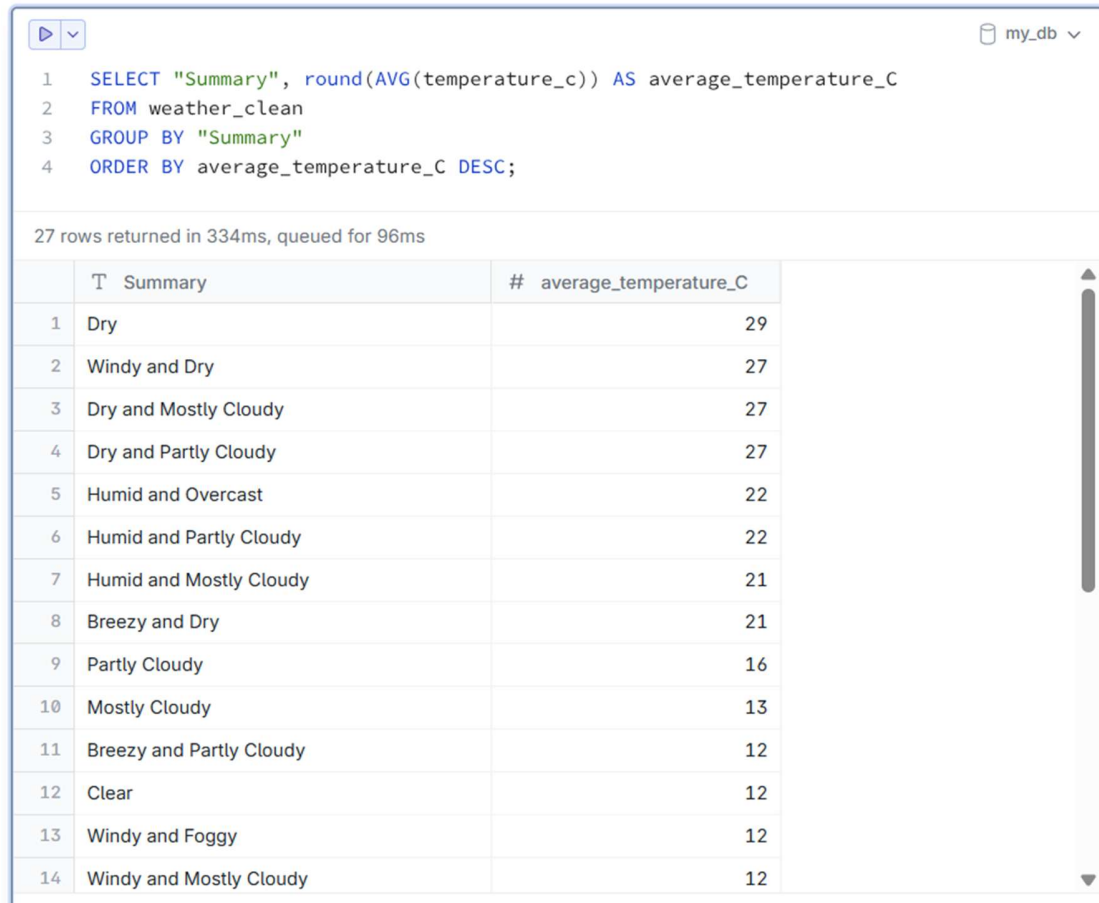
3 rows returned in 309ms

	T	Precip Type	number_of_records
1		snow	10712
2		rain	85200
3		no precipitation	517

Fig. 9: Query 1

Query 2 — Average Temperature by Weather Condition

Calculates the average temperature for each weather condition (summary).



The screenshot shows a database query interface. At the top, there is a play button and a dropdown menu. The SQL query is as follows:

```
1 SELECT "Summary", round(AVG(temperature_c)) AS average_temperature_C
2 FROM weather_clean
3 GROUP BY "Summary"
4 ORDER BY average_temperature_C DESC;
```

Below the query, it states "27 rows returned in 334ms, queued for 96ms". The results are displayed in a table with two columns: "Summary" and "average_temperature_C". The table is sorted in descending order of average temperature.

	Summary	average_temperature_C
1	Dry	29
2	Windy and Dry	27
3	Dry and Mostly Cloudy	27
4	Dry and Partly Cloudy	27
5	Humid and Overcast	22
6	Humid and Partly Cloudy	22
7	Humid and Mostly Cloudy	21
8	Breezy and Dry	21
9	Partly Cloudy	16
10	Mostly Cloudy	13
11	Breezy and Partly Cloudy	12
12	Clear	12
13	Windy and Foggy	12
14	Windy and Mostly Cloudy	12

Fig. 10: Query 2

Query 3 — Average Yearly Temperature (2006-2016)

Calculates the average temperature for each year in the dataset (2006–2016).



The screenshot shows a database query interface. At the top right, there is a dropdown menu labeled 'my_db'. The query is as follows:

```
1 SELECT "YEAR"(datetime) AS year, ROUND(AVG(temperature_c), 2) AS average_temperature_
2 FROM weather_ready
3 GROUP BY year,
4 ORDER BY year;
```

Below the query, it states: '11 rows returned in 381ms, queued for 71ms'. The results are displayed in a table with the following columns: 'year' and 'average_temperature_C'.

	year	average_temperature_C
1	2006	11.22
2	2007	12.14
3	2008	12.16
4	2009	12.27
5	2010	11.17
6	2011	11.52
7	2012	11.99
8	2013	11.94
9	2014	12.53
10	2015	12.31
11	2016	11.99

Fig. 11: Query 3

Query 4 — Average Temperature by Year and Month

Calculates the average temperature grouped by year and month.



The screenshot shows a SQL query execution interface. At the top, there is a play button and a dropdown menu. The query is as follows:

```
1 SELECT "YEAR"(datetime) AS year, "MONTH"(datetime) AS month, ROUND(AVG(temperature_c)
2 FROM weather_ready
3 GROUP BY year, month
4 ORDER BY year, month;
```

Below the query, it states "132 rows returned in 424ms, queued for 43ms". The results are displayed in a table with the following columns: `year`, `month`, and `average_temperature_C`. The table shows data for the year 2006, grouped by month from 1 to 8.

	year	month	average_temperature_C
1	2006	1	-1.67
2	2006	2	-0.06
3	2006	3	4.53
4	2006	4	12.63
5	2006	5	15.67
6	2006	6	19.33
7	2006	7	23.58
8	2006	8	19.49

Fig. 12: Query 4

Query 5 — Average Humidity by Weather Condition

Calculates the average humidity for each weather condition.



The screenshot shows a database query interface. At the top, there is a play button and a dropdown menu. The SQL query is displayed in a monospace font. Below the query, it states '27 rows returned in 439ms, queued for 121ms'. The results are shown in a table with two columns: 'Summary' and 'average_humidity_percent'. The table is sorted in descending order of average humidity.

```
1 SELECT "Summary", ROUND(AVG(humidity * 100), 1) AS average_humidity_percent
2 FROM weather_ready
3 GROUP BY "Summary"
4 ORDER BY average_humidity_percent DESC;
```

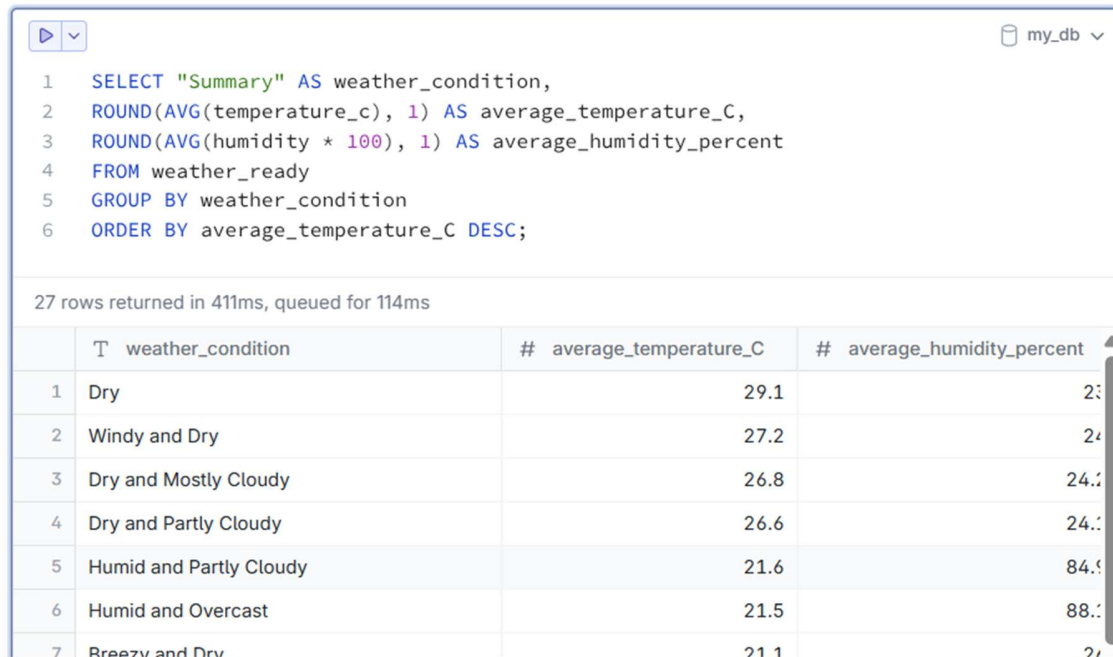
27 rows returned in 439ms, queued for 121ms

	T Summary	# average_humidity_percent
1	Foggy	95.1
2	Rain	94.7
3	Breezy and Foggy	93.9
4	Windy and Foggy	90
5	Light Rain	88.8
6	Humid and Overcast	88.1
7	Humid and Mostly Cloudy	87.4
8	Drizzle	86.8

Fig. 13: Query 5

Query 6 — Average Temperature and Average Humidity by Weather Condition

Calculates the average temperature and average humidity for each weather condition.



```
1 SELECT "Summary" AS weather_condition,
2 ROUND(AVG(temperature_c), 1) AS average_temperature_C,
3 ROUND(AVG(humidity * 100), 1) AS average_humidity_percent
4 FROM weather_ready
5 GROUP BY weather_condition
6 ORDER BY average_temperature_C DESC;
```

27 rows returned in 411ms, queued for 114ms

	T weather_condition	# average_temperature_C	# average_humidity_percent
1	Dry	29.1	23.1
2	Windy and Dry	27.2	24.1
3	Dry and Mostly Cloudy	26.8	24.1
4	Dry and Partly Cloudy	26.6	24.1
5	Humid and Partly Cloudy	21.6	84.9
6	Humid and Overcast	21.5	88.1
7	Breezy and Dry	21.1	27.1

Fig. 14: Query 6

Query 7 — Min and Max Yearly Temperatures (2006–2016)

Shows the minimum and maximum temperatures recorded for each year from 2006 to 2016.



The screenshot shows a database query interface. At the top, there is a play button and a dropdown menu labeled 'my_db'. Below this is the SQL query:

```
1 SELECT "YEAR"(datetime) AS year, ROUND(MIN(temperature_c), 1) AS min_temperature_C,  
2 ROUND (MAX(temperature_c),1) AS max_temperature_C  
3 FROM weather_ready  
4 GROUP BY year,  
5 ORDER BY year DESC;
```

Below the query, it states '11 rows returned in 449ms, queued for 90ms'. The results are displayed in a table with 4 columns: an index, 'year', 'min_temperature_C', and 'max_temperature_C'.

	year	min_temperature_C	max_temperature_C
1	2016	-10.1	34.8
2	2015	-13.1	37.2
3	2014	-13.3	33.9
4	2013	-9	37.9
5	2012	-21.8	38.9
6	2011	-11.7	37.8
7	2010	-15.5	34.9
8	2009	-16.7	36.1
9	2008	-11.1	37.8
10	2007	-10.2	39.9
11	2006	-14.1	34

Fig. 15: Query 7

Query 8 — Average Visibility by Weather Condition

Calculates the average visibility for each weather condition.



```
1 SELECT "Summary" AS weather_condition, ROUND(AVG(visibility_km), 1) AS average_visibi
2 FROM weather_ready
3 GROUP BY weather_condition
4 ORDER BY average_visibility_km DESC;
```

27 rows returned in 392ms, queued for 129ms

	T weather_condition	# average_visibility_km
1	Partly Cloudy	11.8
2	Breezy and Mostly Cloudy	11.5
3	Windy and Partly Cloudy	11.5
4	Dangerously Windy and Partly Cloudy	11.4

Fig. 16: Query 8

Query 9 — Average Comfort Gap by Month

Calculates the average difference between actual and apparent temperature (comfort gap) for each month.



```
1 SELECT "MONTH"(datetime) AS month,
2 ROUND(AVG(ABS(temperature_c - apparent_temp_c)), 2) AS average_comfort_gap_C
3 FROM weather_ready
4 GROUP BY month
5 ORDER BY month;
```

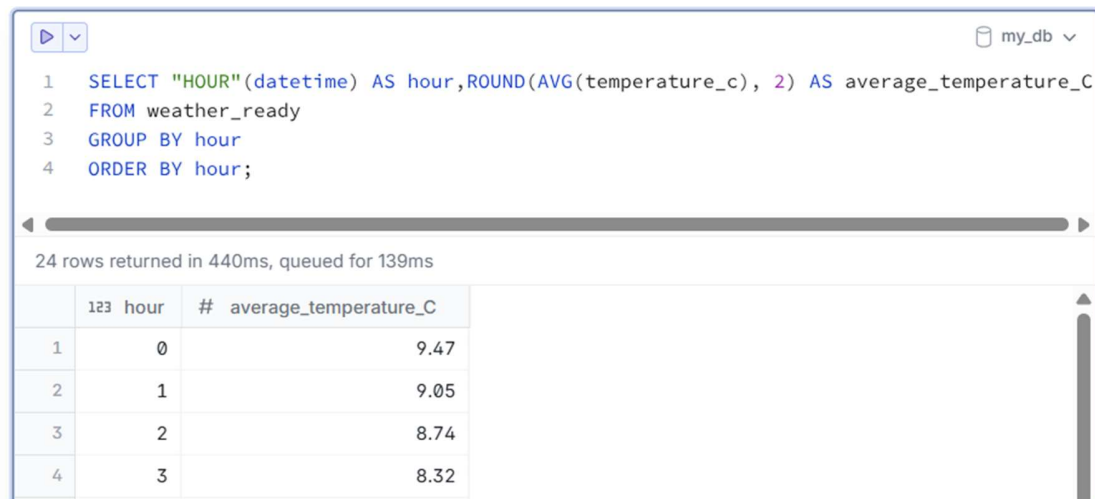
12 rows returned in 493ms, queued for 126ms

	123 month	# average_comfort_gap_C
1	1	2.75
2	2	2.73
3	3	1.83

Fig. 17: Query 9

Query 10 — Average Temperature by Hour of Day

Shows how the average temperature changes throughout the day, grouped by hour.



```
1 SELECT 'HOUR'(datetime) AS hour, ROUND(AVG(temperature_c), 2) AS average_temperature_C
2 FROM weather_ready
3 GROUP BY hour
4 ORDER BY hour;
```

24 rows returned in 440ms, queued for 139ms

	hour	average_temperature_C
1	0	9.47
2	1	9.05
3	2	8.74
4	3	8.32

Fig. 18: Query 10

The prepared **SELECT** queries were exported as **CSV** files, then uploaded to **Power BI** for visualization.











Název	Stav	Datum změny	Typ	Velikost
 Query 1 - Number of Records by Precipitation Type	✓	12.10.2025 10:22	Textový soubor s oddělovači Microsoft Excelu	1 kB
 Query 2 - Average temperature by weather condition	✓	12.10.2025 10:38	Textový soubor s oddělovači Microsoft Excelu	1 kB
 Query 3 - Average yearly temperature	✓	12.10.2025 14:55	Textový soubor s oddělovači Microsoft Excelu	1 kB
 Query 4 - Average temperature by year and month	✓	12.10.2025 15:21	Textový soubor s oddělovači Microsoft Excelu	2 kB
 Query 5 - Average humidity by weather condition	✓	12.10.2025 16:35	Textový soubor s oddělovači Microsoft Excelu	1 kB
 Query 6 - Temperature vs humidity by weather condition	✓	12.10.2025 17:06	Textový soubor s oddělovači Microsoft Excelu	1 kB
 Query 7 - Min and max temperature by year	✓	21.10.2025 17:27	Textový soubor s oddělovači Microsoft Excelu	1 kB
 Query 8 - Average visibility by weather condition	✓	21.10.2025 18:19	Textový soubor s oddělovači Microsoft Excelu	1 kB
 Query 9 - Comfort gap (difference between real and perceived temperature)	✓	21.10.2025 18:33	Textový soubor s oddělovači Microsoft Excelu	1 kB
 Query 10 - Average temperature by hour of day	✓	21.10.2025 19:14	Textový soubor s oddělovači Microsoft Excelu	1 kB

Fig. 19: Queries 1–10 in CSV format, prepared for export to Power BI

Step 5 – Power BI visualization

The visualizations are available in a separate Power BI file.