Data Warehousing for Environmental Applications

Learning Outcomes

After completing this topic, students will be able to:

- Explain the concept and role of data storage with applications to environmental data.
- Set up a data warehouse and manage its core functions.
- Download, search and upload environmental data from documented external sources.
- Apply SQL queries to the created environment data warehouse.

Introduction

Environmental data are collected through various measurements, each capturing only one aspect of the complex ecological system. Some datasets focus on the physical properties of air, others on meteorological conditions, and a third on ecosystem health or human activity levels. While each provides valuable information, none offers a comprehensive overview of the processes that affect the environment. Therefore, analytical work on environmental data almost always begins by integrating and harmonizing them into a unified view of environmental conditions.

In doing so, two main challenges are identified: finding relevant data, which is often distributed across multiple institutions, and storing it in a format suitable for future analysis. The data may arrive in different formats, timeframes, and processing stages, so they must be integrated into a system that supports reliable merging, aggregation, and comparison. This is precisely the function of a data warehouse: to combine heterogeneous data into a unified structure. By doing so, it offers a stable environment for analysis that supports complex queries, trend analysis, and insightful interpretation.

The data warehouse differs from typical transactional databases in its purpose and use. While transactional systems are optimized for quick, frequent data entry, data warehouses focus on long-term storage and analysis of large historical datasets. The emphasis is on analytical queries, trend analysis over time, and the integration of diverse data sources that help to understand complex phenomena and their interrelationships.

In environmental science, this method is essential because isolated measurements rarely capture ecological changes. Air quality, for instance, results from a mix of meteorological conditions, emission sources, geographic features, and seasonal shifts. Only by observing these

data together over an extended period can we accurately evaluate environmental progress, the effectiveness of measures implemented, and potential hazards.

Figure 4.1 illustrates a simplified data warehouse architecture that serves as a central hub for various data formats, ranging from traditional CSV files and semi-structured JSON/XML records to Parquet formats and data retrieved from APIs or other systems' databases. All these data sources are consolidated into a single system, where they are organized, linked, and prepared for analysis. This makes the data warehouse a reliable infrastructure for researchers, institutions, and decision-makers, supporting continuous upgrades and the development of evidence-based policies.

Once the data is loaded into the warehouse, it enables a variety of analyses, from basic statistical summaries and visualizations to the exploration of relationships between variables and the application of advanced machine learning and AI techniques. In the environmental field, this can help forecast pollution events, assess how weather conditions influence air quality, and analyze long-term trends in ecological processes.

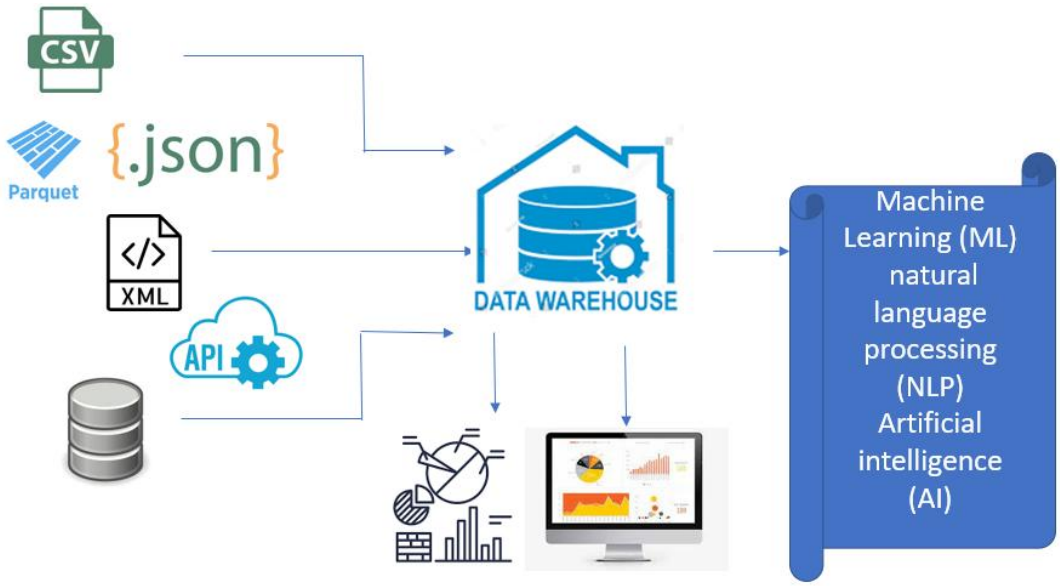

Figure 4.1 Simplified overview of data warehouse architecture

Technological Context

Today, a wide variety of technologies can be used to build data warehouses, each differing in scalability, management complexity, features, and overall costs. Selecting the right technology depends on the specific user needs, available resources, data types, and planned analytical

methods. At one end are local installations of traditional relational databases that offer complete control but require significant technical support and ongoing maintenance. At the other end are commercial cloud platforms, which provide high levels of automation, strong performance, and the ability to process large datasets, with different billing models and varying operational complexities.

For this release, Oracle Autonomous Database was chosen. Oracle Autonomous Database [1] allows the creation of data warehouses in a stable, reliable, and highly automated environment. As part of Oracle Cloud [2], Oracle Cloud is included in the always-free resources, making it accessible to beginners without additional infrastructure costs. Built-in automatic security, configuration, and upgrade management systems remove technical barriers that could otherwise impede less-experienced users, allowing them to focus on analysis and data interpretation.

Indeed, using this platform does not rule out the use of other solutions. The principles in this text also apply to various technologies, allowing readers to explore options such as Amazon Redshift [3], Google BigQuery [4], Microsoft Azure Synapse Analytics [5], or open-source systems like Apache Hive [6], depending on their specific needs and resources. Each of these technologies offers a unique approach to data storage, different scaling models, and its own benefits. However, the core concepts of data integration and analytical processing stay the same.

Create and Set Up a Data Warehouse

The process for installing the Oracle Autonomous Database data warehouse starts with signing in to your Oracle Cloud account. You can create a free account through the Oracle Academy program [7] or follow the instructions at [8]. After logging in, select the Oracle AI Database menu, then choose the Autonomous AI Database options, as shown in Figure 4.2. This provides an overview of all databases in the user's account.

Click the Create Autonomous Database link to set up a new database. The form collects essential details about the instance, including the database name, the cloud user account used for installation, and the work environment. It is advised to select a Lakehouse environment, which is tailored for building data warehouses and offers additional tools for database management, executing SQL queries, and applying machine learning techniques to datasets.

Next, enable always-free access and select the database version. The most common version is included by default, with an option to upgrade to the latest version. Then, define the administrator credentials by setting a password for the system administrator (ADMIN username), who initially has full system access and is responsible for creating new users, loading data, and

configuring the database. The password must meet security requirements. The final step is to choose how to access the instance network.

In an always-free environment, the most common feature is the ability to access the internet securely from any device. Secure access from any location, using a digital wallet and authentication mechanisms built into an autonomous database. This approach significantly simplifies work in educational settings by eliminating the need for a dedicated network or additional infrastructure components, such as VPN tunnels or IP address restrictions.

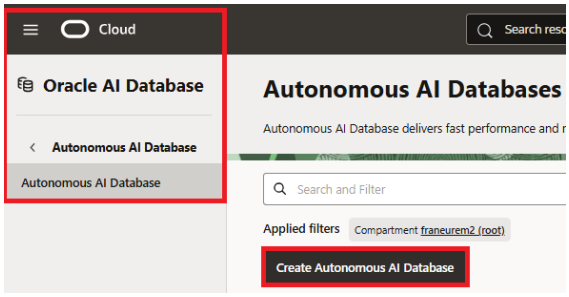

Figure 4.2 Access to the Oracle Autonomous Database platform

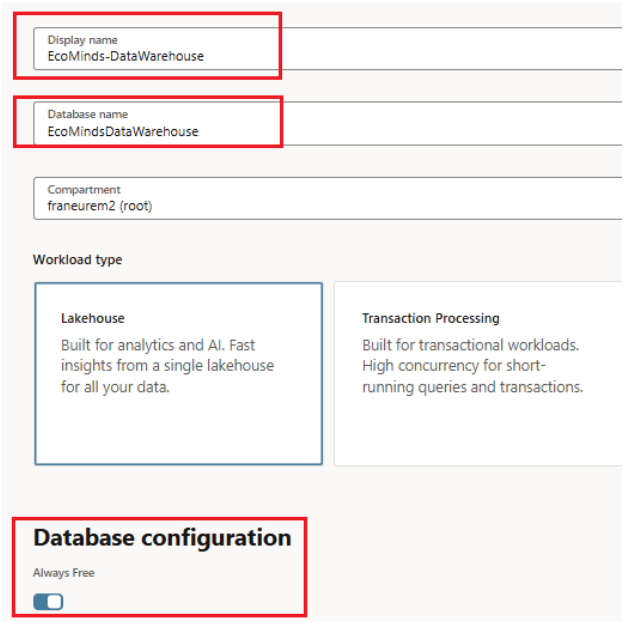

Figure 4.3 Data warehouse name and workload selection

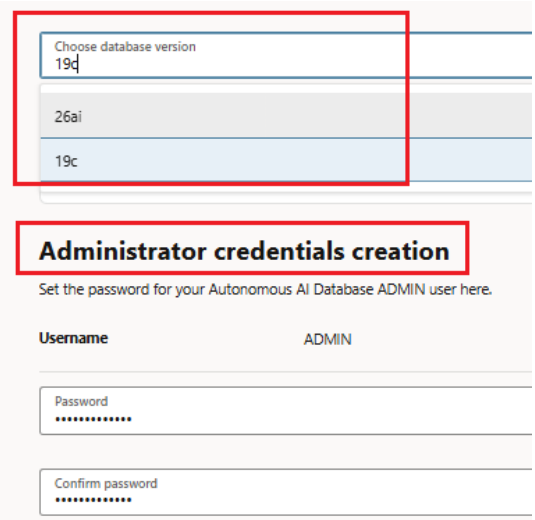

Figure 4.4 Data warehouse final configuration

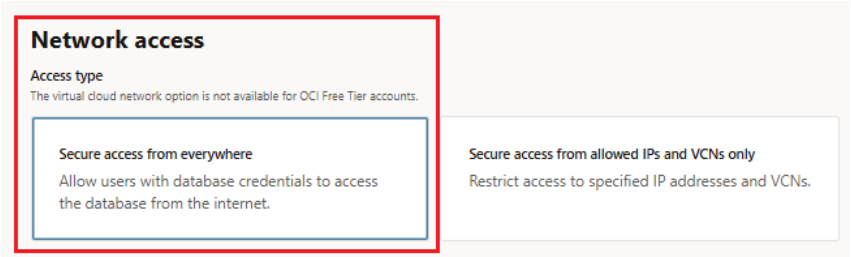

Figure 4.5 Network access selection

Authentication and Development Environment

All environmental data analyses in the data warehouse depend on SQL to retrieve, filter, aggregate, and join data to create comprehensive analytical views. For these queries to execute smoothly, researchers need access to a data warehouse and a reliable development environment.

There are two main ways to connect to an Oracle Autonomous Database. The first and simplest method is to use the Oracle Cloud web interface. Two-step authentication (2FA) is enabled during login, granting access to all necessary resources. After successfully logging in, select Database Actions, then SQL, to open an integrated web development environment for running SQL statements (Figures 4.6 and 4.7). The advantage of this method is that it requires no additional installation; because it can be accessed from any device and browser, it is particularly convenient for beginners.

Another way to access is through the Oracle SQL Developer desktop application, which is available for free download at [9]. SQL Developer provides advanced script management,

supports more complex projects, enables review of the database structure, and assists with administrative tasks. To connect using this application, you need to download a digital wallet from the Oracle Cloud console by clicking on Database Connection (Figure 4.8). The package includes the certificates and configuration files required to establish a secure mTLS connection to the data warehouse, ensuring mutual authentication between the client and the server. After uploading the digital wallet to SQL Developer (Figure 4.9), you can connect to the data warehouse and work with it in a standard desktop environment (Figure 4.10).

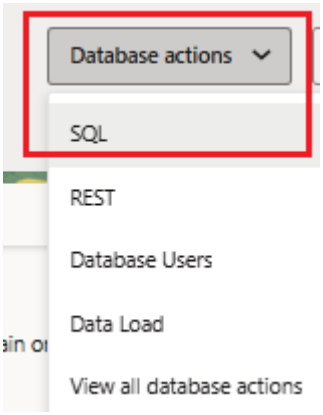

Figure 4.6 Running the SQL Developer web application

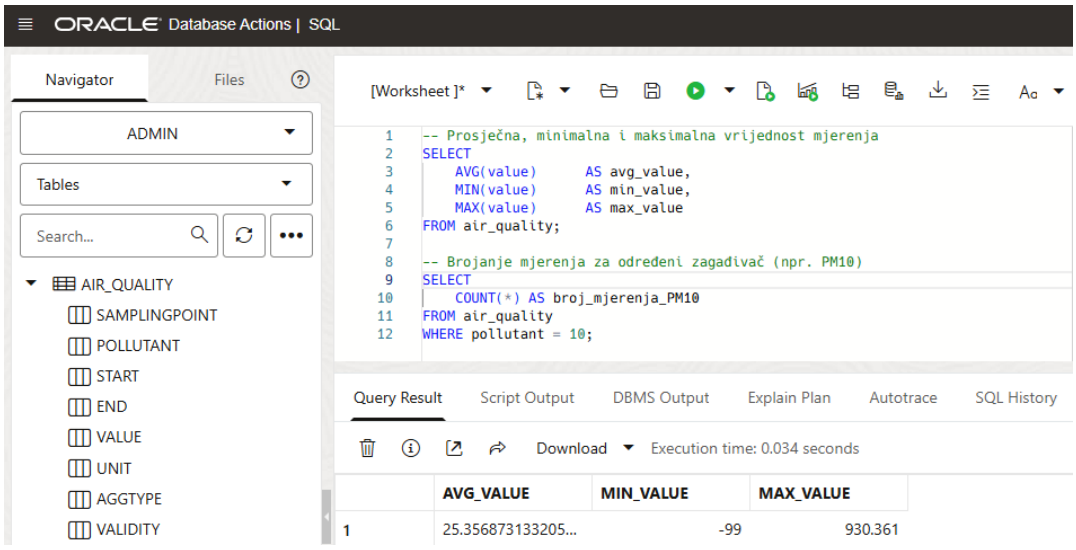

Figure 4.7: SQL Developer web application

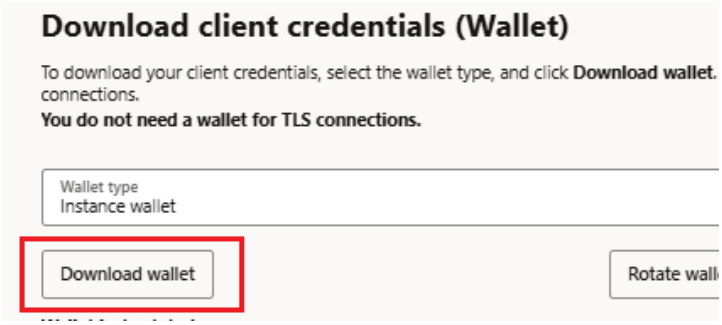

Figure 4.8 Digital wallet download

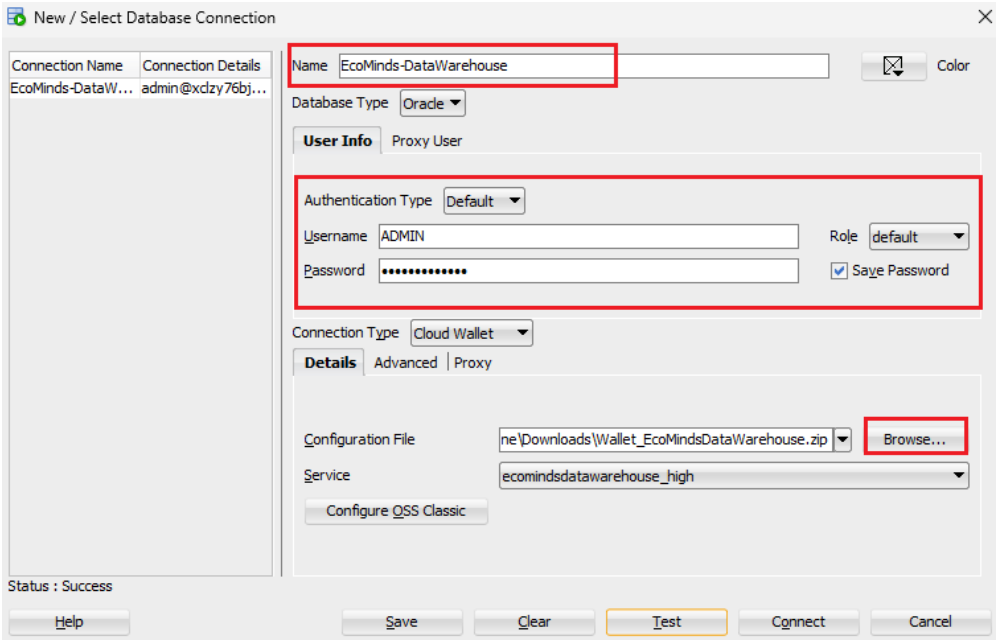

Figure 4.9 Setup Data Warehouse Connection from a SQL Developer desktop application

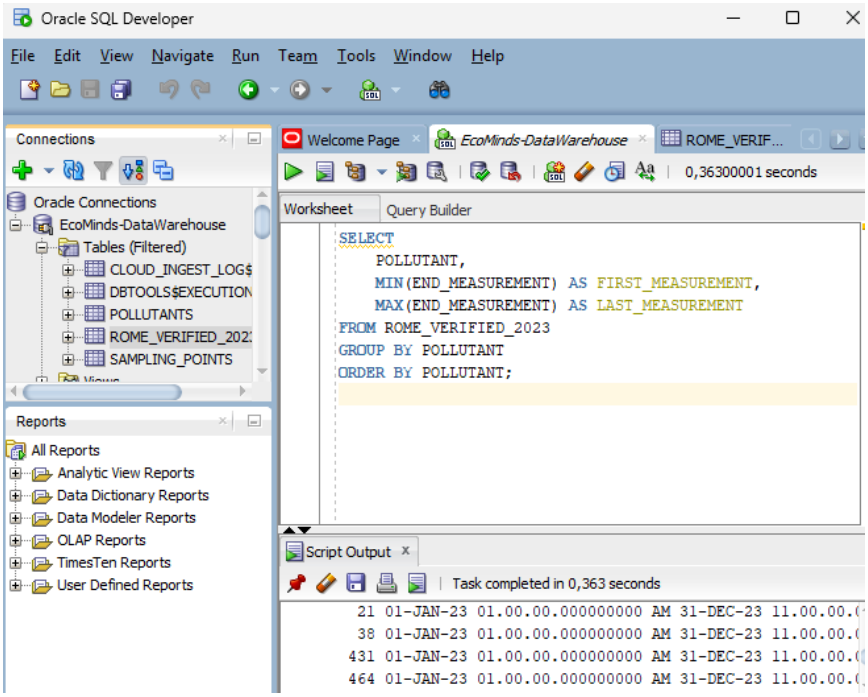

Figure 4.10: Work in the SQL Developer desktop application

User Rights Administration

After creating the database, the first available user in the system is ADMIN. In educational settings, especially during introductory exercises, it is temporarily allowed to use an ADMIN account to simplify data loading and basic SQL query execution. Although this approach is convenient for initial system familiarization, it is not considered acceptable in professional applications.

In production environments, the ADMIN role should be reserved for administrative tasks such as creating users, granting privileges, and configuring the database. Because the ADMIN account has full rights, using it for daily operations introduces a security risk. Therefore, dedicated accounts or roles with limited access are created for users performing analytical or development tasks, in accordance with the principle of least privilege.

Note that the ADMIN does not have access to all specialized tools in Autonomous Database. Some features, such as Oracle Machine Learning (OML), graph analytics, and specific web components, require privileged user accounts. As a result, separate users with clearly defined access levels are usually created to work with these advanced tools, as shown in Figures 4.11-4.13.

Figure 4.11 User Admissions Selection

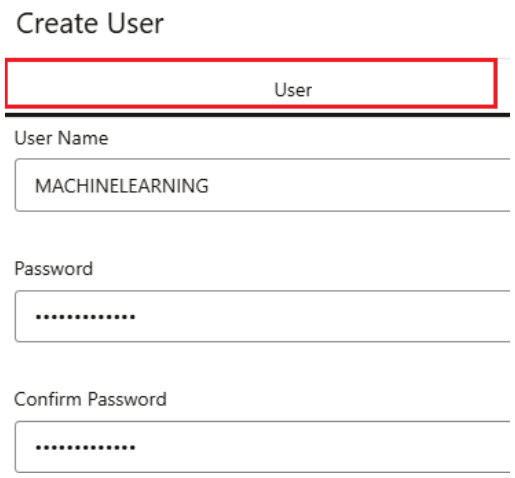

Figure 4.12 New User Entry

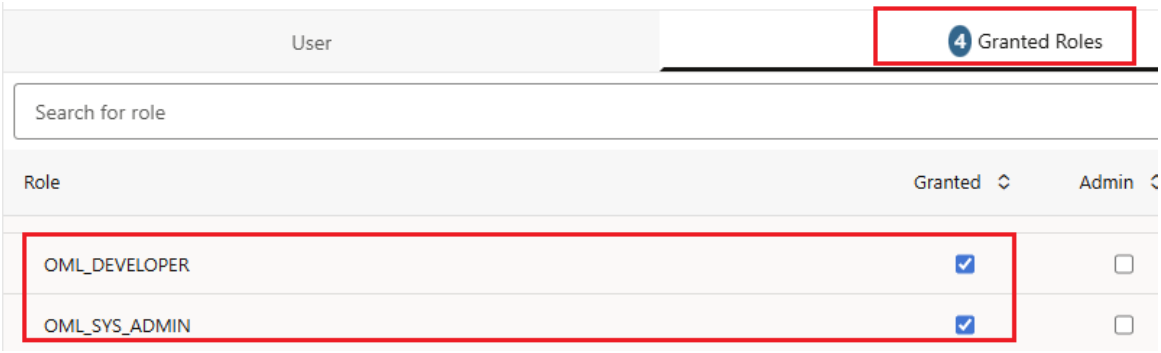

Figure 4.13 Defining user rights

Data Loading

Proper data preparation is essential for effective data warehouse operations. After configuring the data warehouse with the Autonomous Database instance, all data is loaded through a single Data Load interface accessible from Database Actions (Figures 4.14 and 4.15).

The simplest way to import data into a warehouse is to upload local files via a straightforward drag-and-drop process in the Data Load desktop within the work environment. The system automatically recognizes the structure, suggests column names and data types, and provides the option to create a new table. This approach works well for smaller datasets in CSV, JSON, or similar formats and doesn't need additional tools.

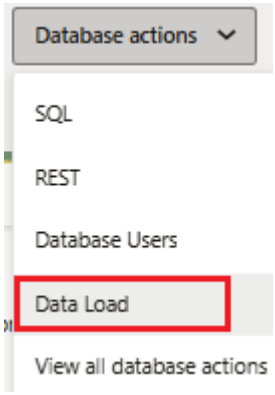

Figure 4.14 Running Data Load

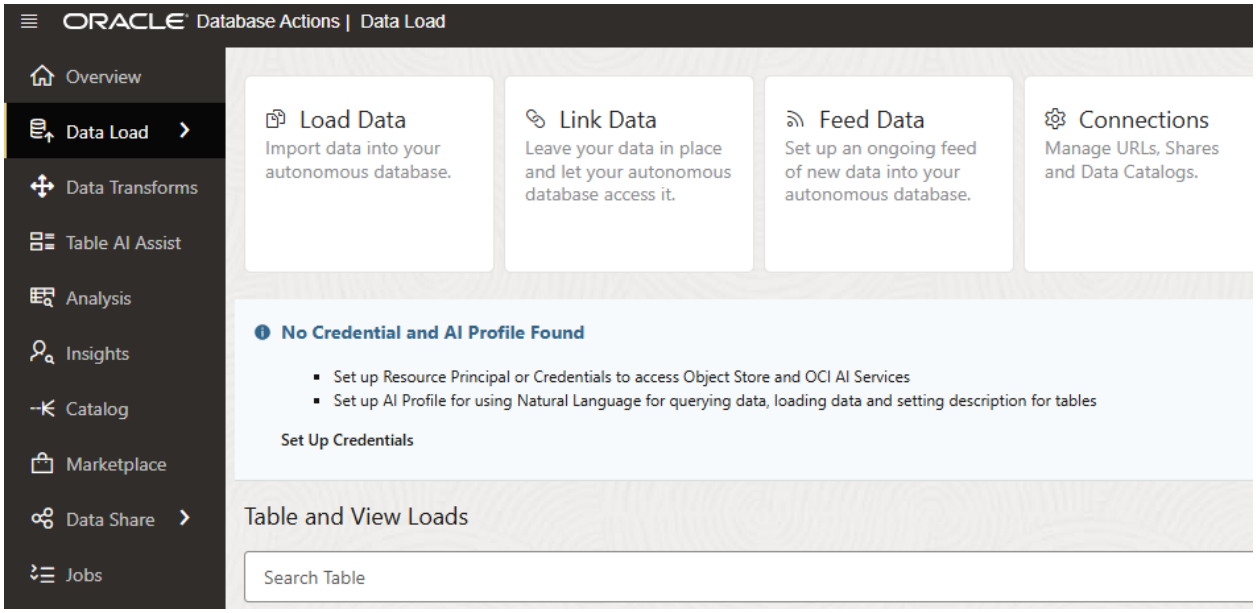

Figure 4.15 Data Load interface

For large files, especially in Parquet format, it is recommended to pre-load them into an Oracle Object Storage service on Oracle Cloud Infrastructure. In the specified repository, data is stored in containers called buckets, and up to 20 GB of workspace is available in permanently free mode, which is sufficient for various analytical applications.

A bucket is created unilaterally in Storage - Object Storage, with the bucket name set to default (Figures 4.16 and 4.17). Afterwards, the prepared files can be uploaded to and downloaded from the data warehouse (Figure 4.18).

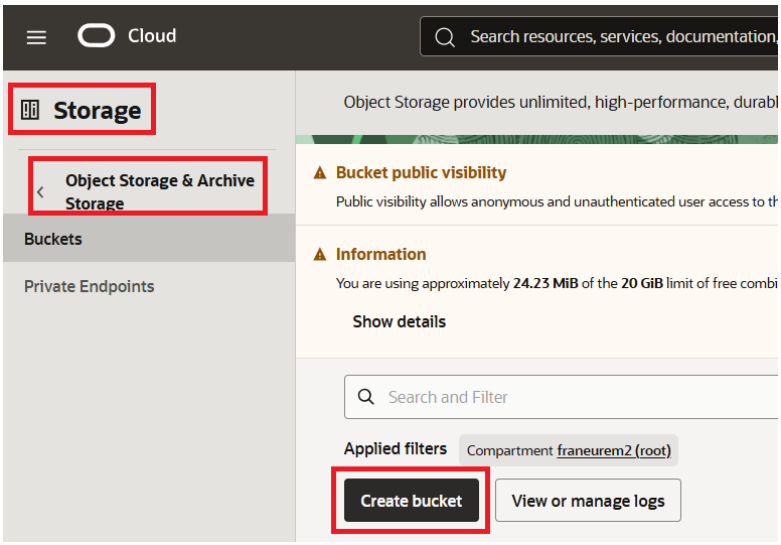

Figure 4.16 Creating a new Oracle Object Storage bucket.

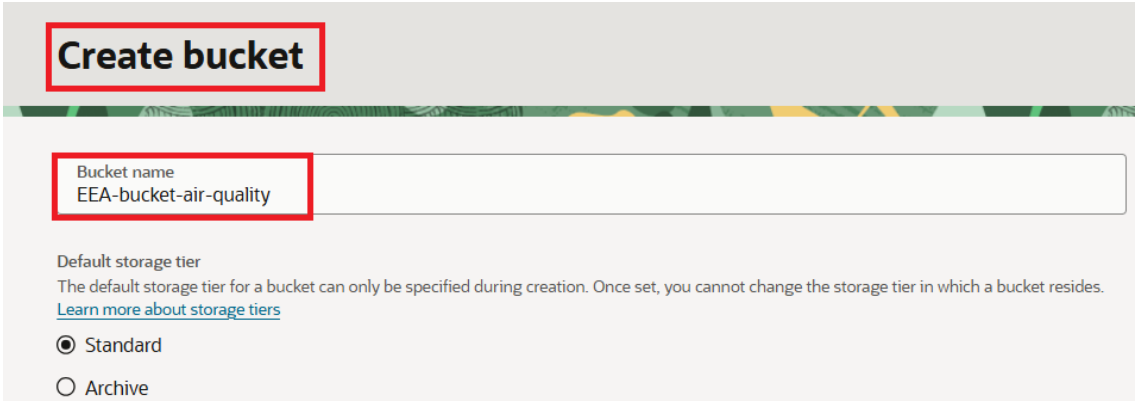

Figure 4.17 Creating a new Oracle Object Storage bucket.

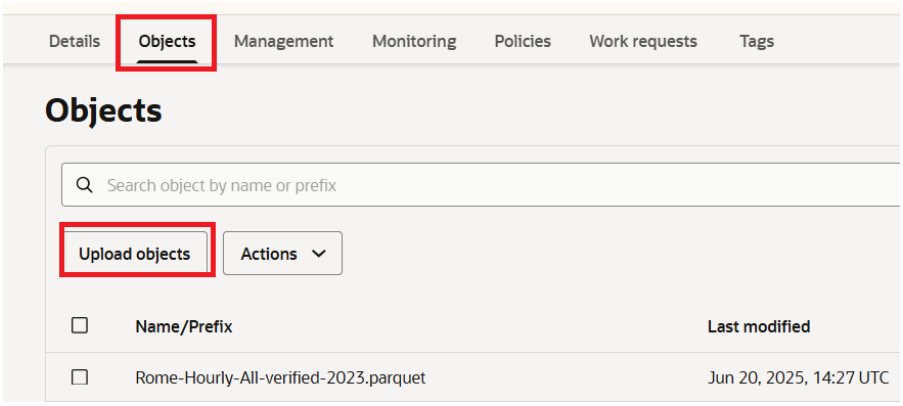

Figure 4.18 Uploading files to a bucket

When files are stored in a bucket, their database import is rerun from the same Data Load interface. Instead of using the local file, the Cloud Store source (Figure 4.19) is selected, and the

object's URL is entered from the bucket. Below, the system automatically maps the data and creates its table.

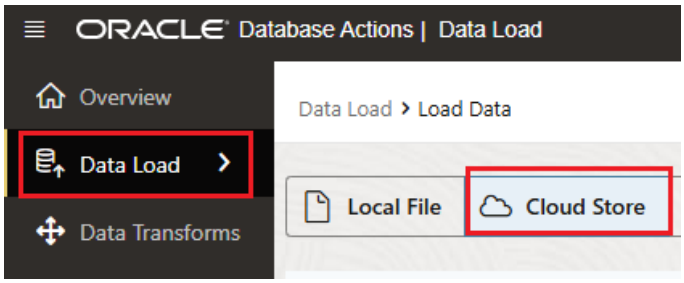

Figure 4.19 Loading data from Object Storage with the Data Load module

Data Modeling

After loading the data, the next step is to model it to support future analysis. Emphasis is placed on designing clear and understandable table structures in the database to ensure efficient execution of SQL queries.

Initially, tables are usually created directly from uploaded files. Each dataset, such as air quality measurements or meteorological records, should be organized into a separate table, with each row representing an individual measurement. These tables are often referred to as raw tables because they present the source data structure without further transformations. While fundamental analyses can be performed on them, their structure can be further refined to enable easier and more reliable query execution.

For more complex analyses, it is helpful to standardize column names, date formats, and location or measurement-station identifiers in advance. A consistent structure facilitates table merging and reduces the risk of interpretation errors. In some cases, basic denormalization can be used, such as adding information about the measurement station from the auxiliary table directly into the measurement table, which simplifies queries that would otherwise require multiple joins.

One of the most common and conceptually simple ways to organize data is the so-called star schema shown in Figure 4.20. It uses a central table, such as a measurement table, as the primary data source, while descriptive data on measurement locations, stations, or other categories are stored in smaller tables. These tables, called dimension tables, are joined to the central table as needed, making analytical queries more straightforward to write.

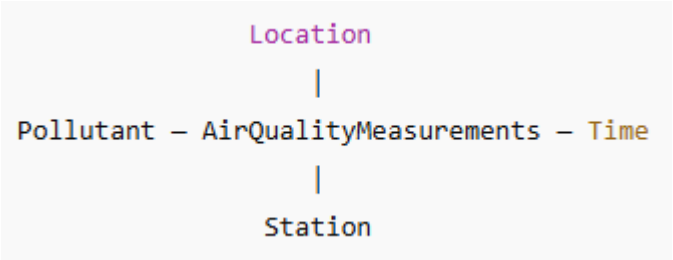

Figure 4.20 Simple star scheme

Data Visualization

After running the SQL query, you can display the results as charts using the built-in Charts module. Access this module from View all database actions, then go to the Data Studio tab, which contains Charts as shown in Figure 4.21. This tool lets you create simple visualizations directly from SQL query results without requiring external applications or additional installations. It enables rapid, straightforward data interpretation, particularly for analyzing trends, comparing datasets, or identifying anomalies.

For example, we can imagine a table of hourly measurements of the concentration of a specific hemispheric air mass over an extended period. In this case, it is often necessary to focus only on daily measurements. In the example shown, a bar graph is used: on the X-axis, measurement times for the selected day are displayed, while on the Y-axis, their corresponding values are shown (Figure 4.22).

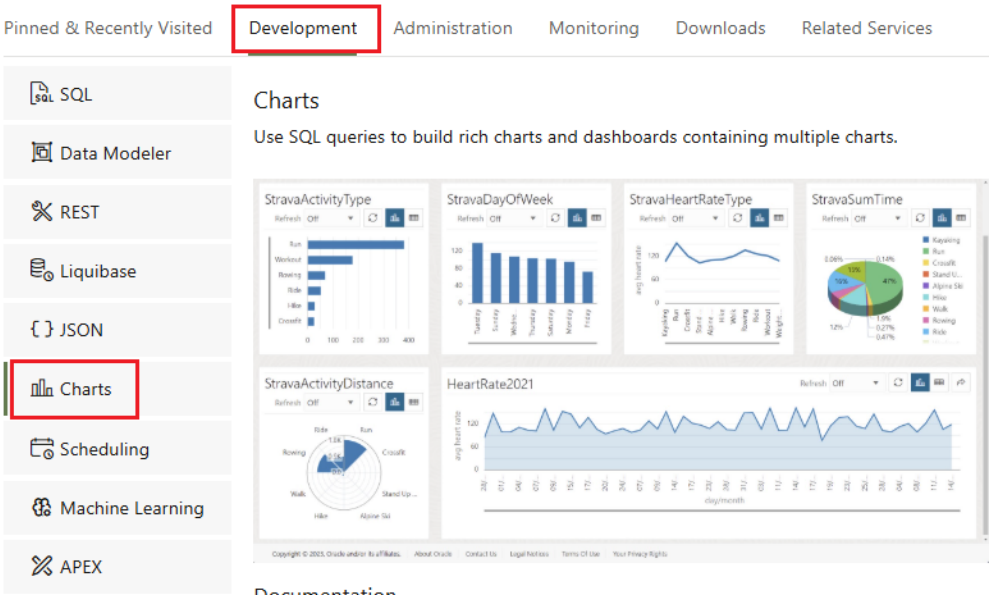

Figure 4.21 Startup Charts Module

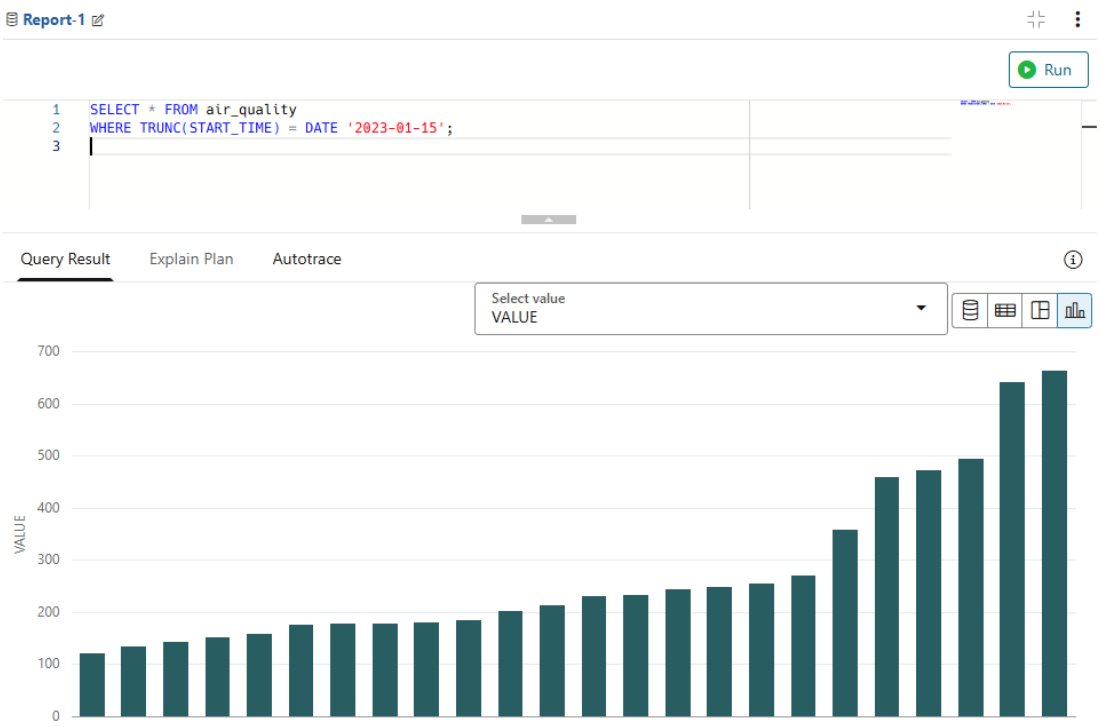

Figure 4.22: Simple data visualization

In-Database Machine Learning

A data warehouse with an Oracle Autonomous Database includes an embedded machine learning system, Oracle Machine Learning (OML), that allows you to train and deploy models directly on data stored in the database. Unlike methods that export data to external tools, OML enables data preparation, modeling, and evaluation within the database, eliminating the need for additional infrastructure and large-scale data transfers.

Access the OML environment by choosing Database Actions - View all database actions, then clicking the Development tab and the Machine Learning menu (Figure 4.23). This opens the OML workspace, as shown in Figure 4.24.

The simplest way to work in OML is through the OML Notebooks environment (see Figure 4.24), which allows you to write and execute SQL and Python scripts directly in your web browser. It offers features for classification, regression, clustering, anomaly detection, and automated model training.

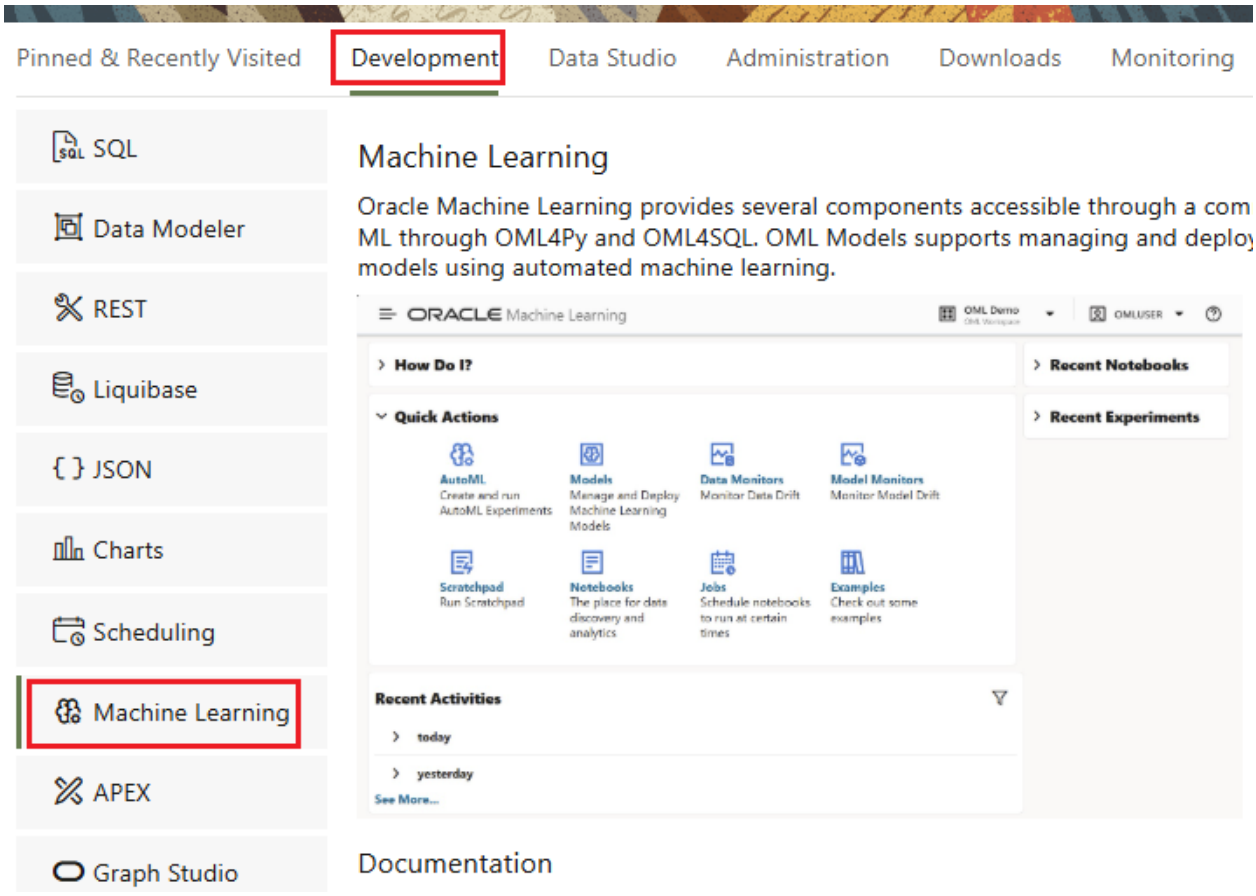

Figure 4.23 OML launch

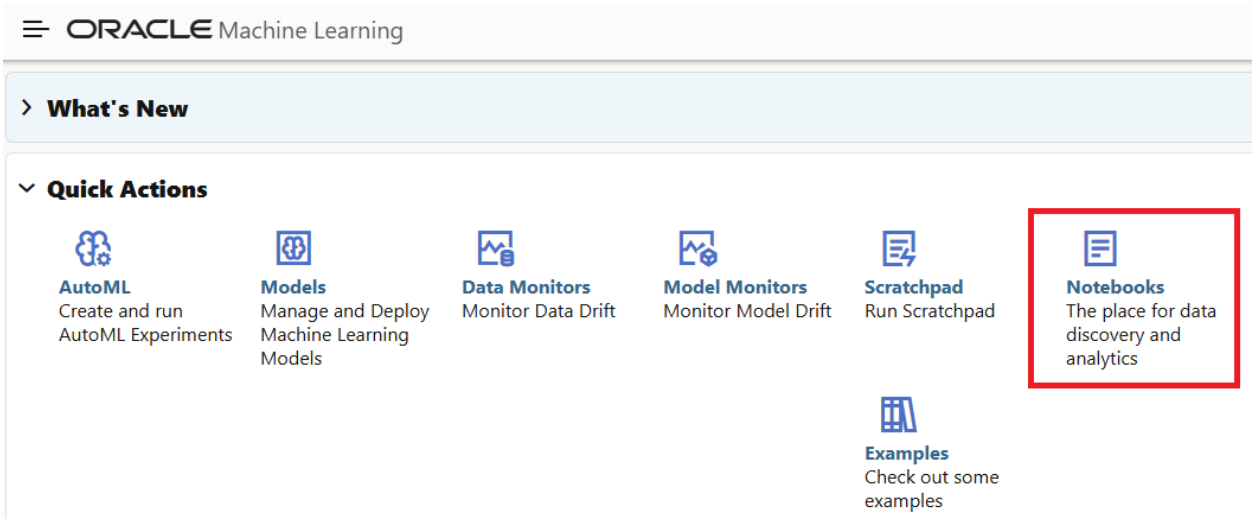

Figure 4.24 OML interface

Case Study: Air Pollution Analysis

This case study presents real-world air quality data utilized by European environmental agencies, research institutions, and analytical systems. These hourly measurements are provided by the European Environment Agency (EEA).

The example shows the whole process of working with environmental datasets. The first part covers data collection and storage methods in the data warehouse. Later steps include data analysis related to environmental issues, such as monitoring nitrogen dioxide (NO_2) or particulate matter ($PM_{2.5}$).

European Air Quality Data Sources

The air quality data can be downloaded from the EEA portal [10], which collects, verifies, and publishes measurements from national environmental monitoring networks. The EEA system offers access to detailed hourly, daily, and yearly concentration records for pollutants such as nitrogen dioxide (NO_2), ozone, PM2.5, and other substances, all essential for assessing air quality.

Measuring sites are located throughout Europe and follow standardized methods in accordance with European air quality monitoring directives. Before publication, data undergo validation to remove errors, logical inconsistencies, and missing records. The processed data is available via the official EEA service [11], including options for Parquet and CSV downloads.

After selecting the geographic area (Figure 4.25), time, and the desired measurement interval, the system generates a package containing the specified measurements. The downloaded file typically contains multiple Parquet records, one for each measurement station within the selected area. This layout allows for detailed analysis; however, in analytical settings, it is often helpful to consolidate all records into a single table. This can be done using available Parquet file merge tools, such as Agents for Data [12], which enable users to drag and combine multiple files into a single dataset (Figure 4.26).

It is also important to emphasize the need to understand the structure of downloaded data, their attributes and meanings, and the published links that facilitate the study of metadata, data dictionaries, vocabularies, and technical documentation for published API interfaces.

.

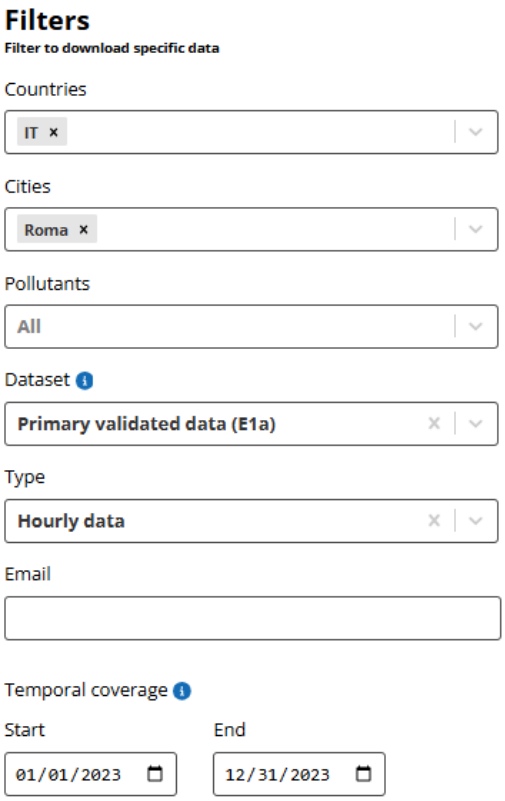

Figure 4.25 Download EEA data

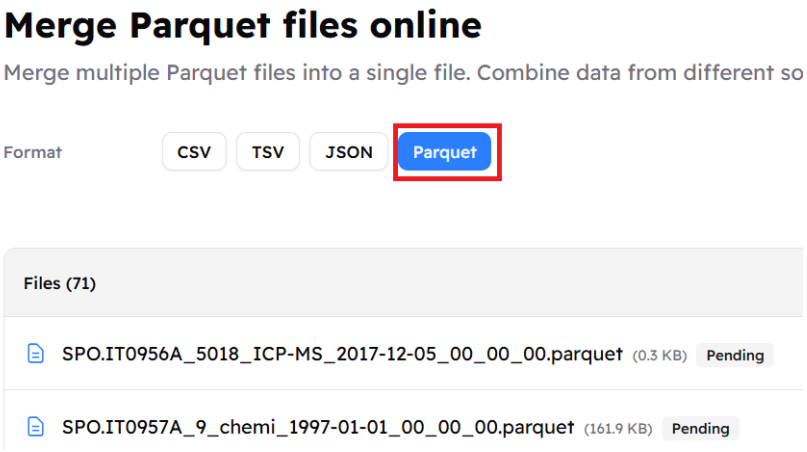

Figure 4.26 Merge multiple Parquet files on the Agents for the Data platform

Store EEA Measurements in Oracle Object Storage

After downloading the data from the EEA, the next step is to store it in a pre-configured container on the Oracle Object Storage service. Inside the container, select Objects, then click Upload Objects and upload the measure parquet file (Fig. 4.27).

Each uploaded object receives its own URL tag, which can be viewed in the object's properties (see Figure 4.28). This tag is used when importing data into the data warehouse later.

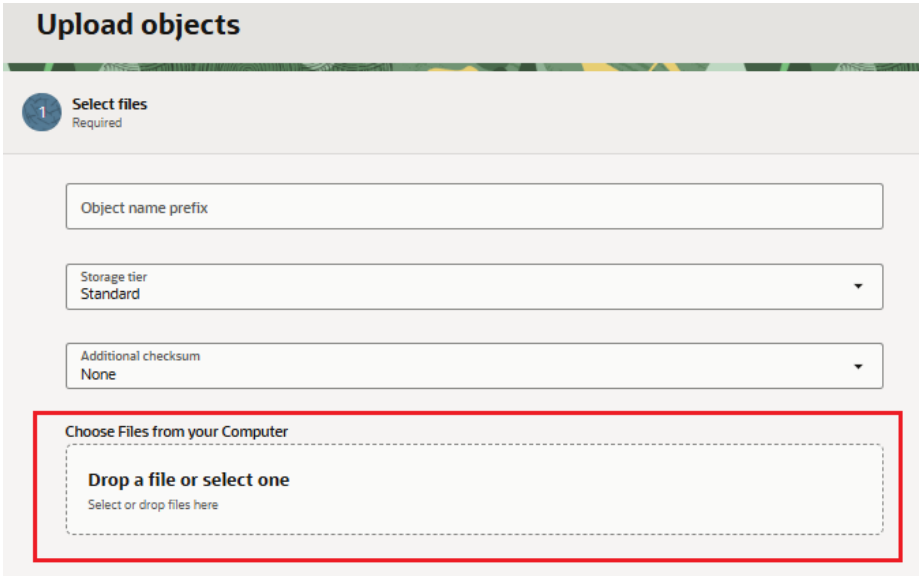

Figure 4.27 Uploading the EEA file to a bucket

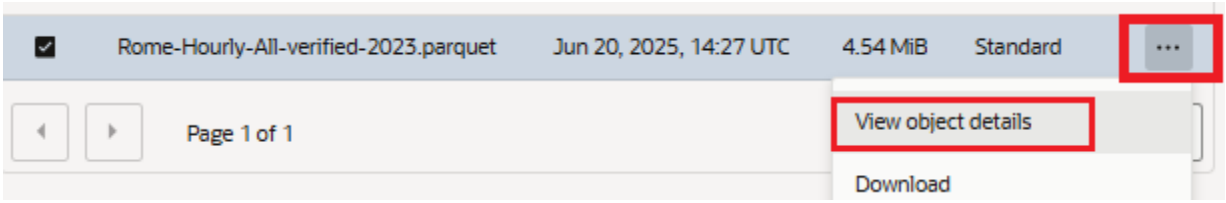

Figure 4.28: Generating object URL tags

Import Measurements into the Data Warehouse

The next step opens the Data Load interface in your Autonomous Database environment and selects the Cloud Store tab. The data source field contains the previously copied URL of the parquet object stored in the bucket (Figure 4.29).

The left panel displays the structure of available data, which can be selected and easily dragged onto the desktop for import. The upload process starts by clicking the Start button (Figure 4.30).

Before importing, it's recommended to open the Settings link to review the file structure and set key upload options: create a new table or update an existing one, map columns, and select the user schema (Figure 4.31).

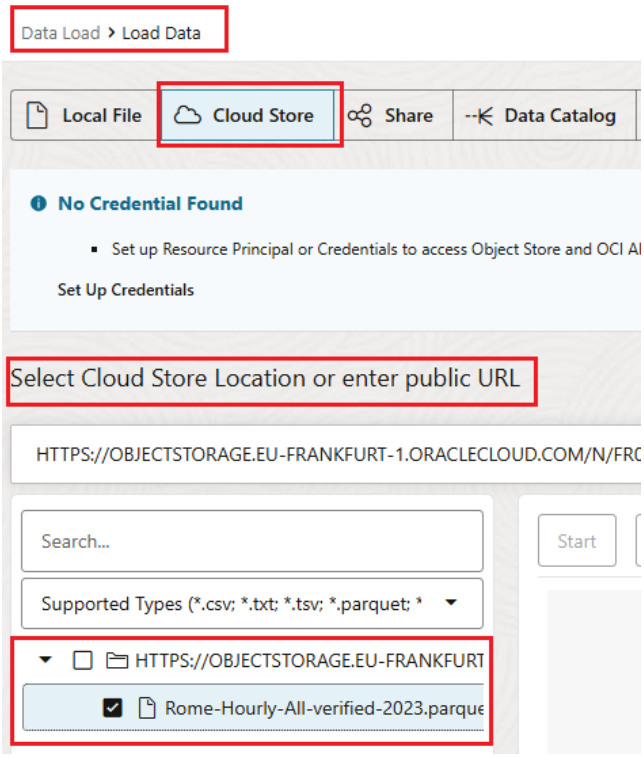

Figure 4.29 Preparation for loading measurement data

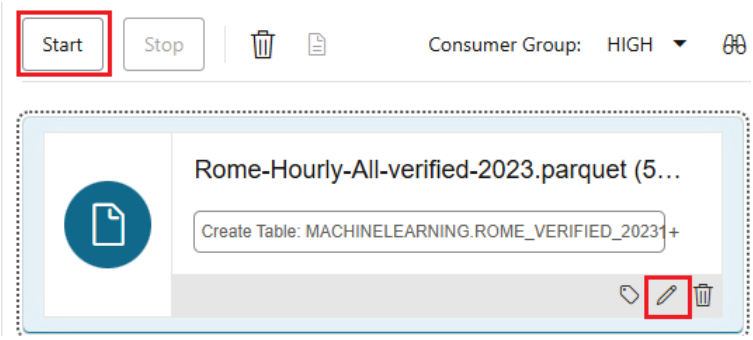

Figure 4.30 Preparation for loading measurement data

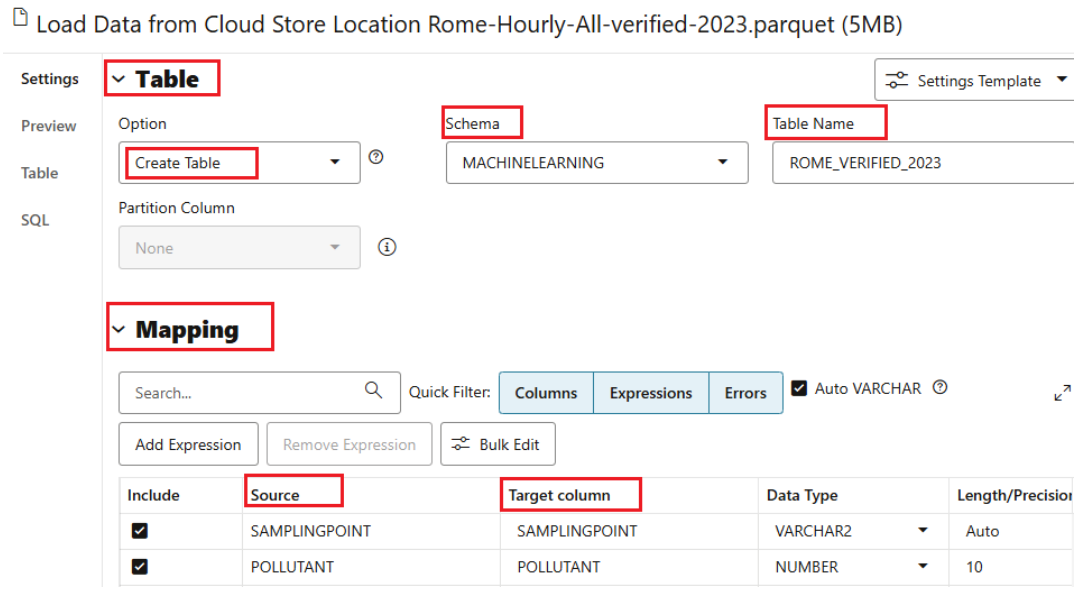

Figure 4.31: Adjusting transfer parameters

Examples of Initial Data Analysis

Below are examples of basic queries that let you view table content and understand the available attributes. This initial review helps clarify the data structure, measurement range, and how pollutants and measuring stations are identified.

You can view all pollutants listed in the measurement table with this SQL query:

```
-------------------------------------------------------------------------------
-- 1. List all pollutants that were measured
--    Each pollutant is identified by its numeric code.
-------------------------------------------------------------------------------
SELECT DISTINCT POLLUTANT
FROM ROME_VERIFIED_2023
ORDER BY POLLUTANT;
```

Then, it is helpful to compile a list of all measuring stations:

```
-------------------------------------------------------------------------------
-- 2. List all measurement stations (sampling points)
--    Each SAMPLINGPOINT represents a unique monitoring location.
-------------------------------------------------------------------------------
SELECT DISTINCT SAMPLINGPOINT
FROM ROME_VERIFIED_2023
ORDER BY SAMPLINGPOINT;
```

It is also possible to identify the range of measured values for all reported pollution:

```
-------------------------------------------------------------------------------
-- 3. Determine the measurement range for each pollutant
```

```
--    Shows the first and last measurement timestamp per pollutant.
-------------------------------------------------------------------------------
SELECT
    POLLUTANT,
    MIN(END_MEASUREMENT) AS FIRST_MEASUREMENT,
    MAX(END_MEASUREMENT) AS LAST_MEASUREMENT
FROM ROME_VERIFIED_2023
GROUP BY POLLUTANT
ORDER BY POLLUTANT;
```

As in the previous query, it is possible to display pollution measurement ranges grouped by measurement station.

```
-------------------------------------------------------------------------------
-- 4. Determine the measurement range for each sampling point
--    Shows when each station started and stopped reporting measurements.
-------------------------------------------------------------------------------
SELECT
    SAMPLINGPOINT,
    MIN(END_MEASUREMENT) AS FIRST_MEASUREMENT,
    MAX(END_MEASUREMENT) AS LAST_MEASUREMENT
FROM ROME_VERIFIED_2023
GROUP BY SAMPLINGPOINT
ORDER BY SAMPLINGPOINT;
```

Link Measurement to Pollution Descriptions

Unfortunately, it is not possible to identify which pollutants are involved from the air quality measurement table (ROME_VERIFIED_2023), as it only provides their numerical codes. The problem is solved by linking this table to another that contains detailed information on air pollutants and their labels. This data source, necessary to create the additional table, is available in the EEA dictionary at [13]. The pollution list can be easily copied into an Excel file and then uploaded to a data warehouse using a Data Load environment. In this example, the data is imported into the POLLUTANTS table, as shown in Figure 4.32.

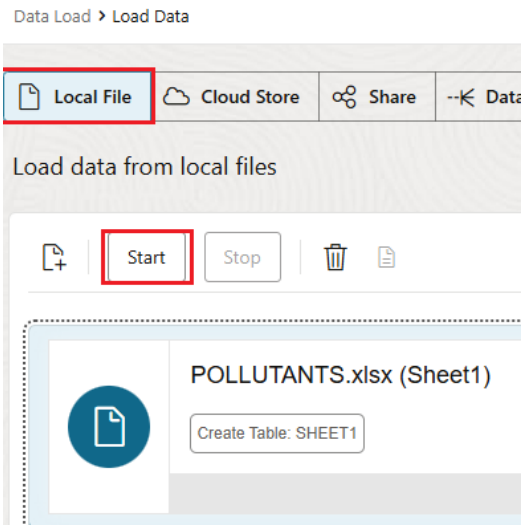

Figure 4.32 Import of air pollution codes

You can now connect measurements and pollution types in one SQL query.

```
SELECT
    DISTINCT m.POLLUTANT AS POLLUTANT_CODE,    -- numeric code as stored in
measurements
    p.LABEL,                                    -- official pollutant name
    p.NOTATION,                                 -- notation (e.g., SO2, PM10)
    p.MANDATORYUNIT                            -- recommended measurement unit
FROM ROME_VERIFIED_2023 m
LEFT JOIN POLLUTANTS p
       ON m.POLLUTANT = p.POLLUTANT_ID
ORDER BY POLLUTANT_CODE;
```

Additionally, we can evaluate the annual average NO_2 concentration across all measurement stations.

```
-------------------------------------------------------------------------------
-- Annual NO2 averages (pollutant = 8), excluding negative values.
-- Joined with POLLUTANTS for pollutant name, symbol and unit.
-------------------------------------------------------------------------------
SELECT m.SAMPLINGPOINT, m.POLLUTANT AS POLLUTANT_CODE, p.LABEL AS POLLUTANT_NAME,
       p.NOTATION AS POLLUTANT_SYMBOL, p.MANDATORYUNIT AS UNIT,
       EXTRACT(YEAR FROM m.END_MEASUREMENT) AS YEAR, AVG(m.VALUE) AS AVG_VALUE,
       COUNT(*) AS COUNT_MEASUREMENTS, MIN(m.END_MEASUREMENT) AS FIRST_MEASUREMENT,
       MAX(m.END_MEASUREMENT) AS LAST_MEASUREMENT
FROM ROME_VERIFIED_2023 m
LEFT JOIN POLLUTANTS p ON m.POLLUTANT = p.POLLUTANT_ID
WHERE m.POLLUTANT = 8          -- filter for NO2
  AND m.VALUE >= 0             -- exclude negative measurements
GROUP BY m.SAMPLINGPOINT, m.POLLUTANT, p.LABEL, p.NOTATION, p.MANDATORYUNIT,
         EXTRACT(YEAR FROM m.END_MEASUREMENT)
ORDER BY m.SAMPLINGPOINT, YEAR;
```

Running the previous query indicates that the measuring station is marked IT/SPO".IT1837A_8_chemi_2006-12-17_00:00:00 records an average annual NO level of 47,074 μg/m³, which is hazardous to health (the upper limit is 40,00 μg/m³).

Link Measurements to Measurement Station locations

After confirming that individual measuring stations record NO_2 exceedances, it is essential to obtain additional information on their locations. Table ROME_VERIFIED_2023 contains only the identifiers of the measuring stations; therefore, one cannot directly determine their exact locations. This issue is addressed by adding another table to the data warehouse that stores descriptive data for the measuring stations, including their names, geographic coordinates, and station types.

The source of such data is the EEA portal [10] mentioned above, which provides a complete list of European measurement stations. The list is downloaded as a CSV file by clicking the Metadata link, then selecting Download CSV (see Figure 4.29). The downloaded CSV file is then uploaded back to the data warehouse using the Data Load workbench (see Figure 4.30). This process creates a new table, in this example called SAMPLING_POINTS, containing all relevant information about the measuring cells (see Figure 4.31).

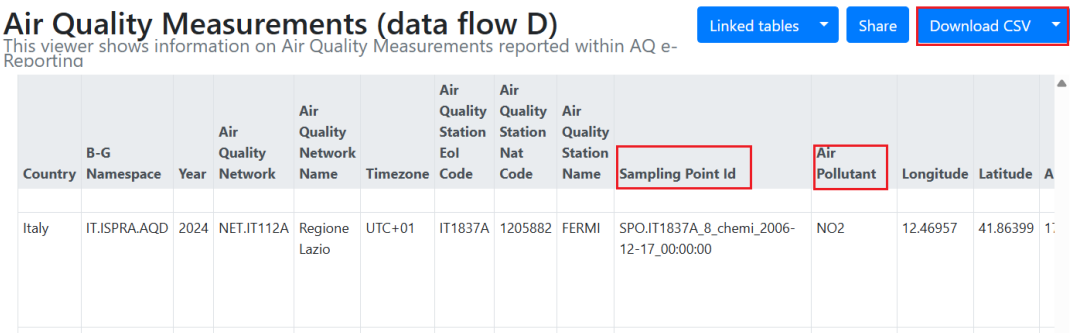

Figure 4.29 Download of measurement station data

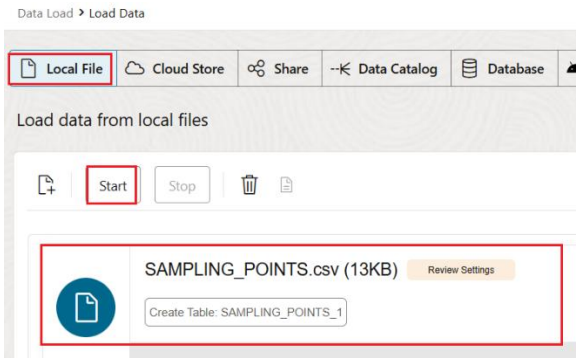

Figure 4.30 Loading of measurement station data

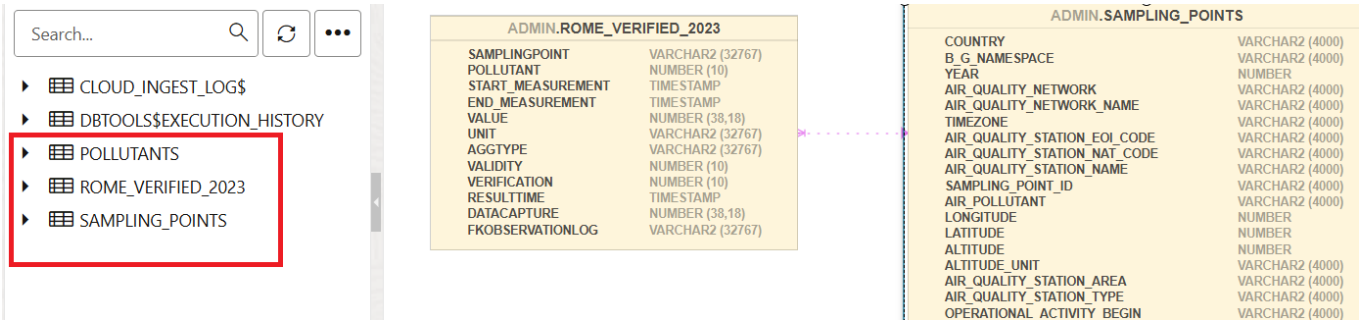

Figure 4.31 Air pollution tables

After analyzing the tables generated from the data import, it is determined that the table ROME_VERIFIED_2023 can be linked to the table POLLUTANTS through the column POLLUTANT in ROME_VERIFIED_2023 and the column POLLUTANT_ID in POLLUTANTS.

Additionally, the ROME_VERIFIED_2023 table can be linked to the SAMPLING_POINTS table via the SAMPLINGPOINT column in ROME_VERIFIED_2023 and the SAMPLING_POINT_ID column in SAMPLING_POINTS.

However, when attempting to connect, a problem arises because the values in the SAMPLINGPOINT column of the ROME_VERIFIED_2023 table do not directly match those in the SAMPLING_POINTS table. Specifically, in the ROME_VERIFIED_2023 table, the value might be IT/SPO.In IT1837A_8_chemi_2006-12-17_00:00:00, the same location exists only as SPO in the SAMPLING_POINTS table.IT1837A_8_chemi_2006-12-17_00:00:00.

Therefore, the record in ROME_VERIFIED_2023 includes a country prefix (IT/), which is missing in the SAMPLING_POINTS table. Since SQL JOIN requires an exact match of values, these tables cannot be linked directly.

The problem is resolved by using the SUBSTR (SAMPLINGPOINT, 4) function in the JOIN clause, which removes the first three characters (IT/) from the value in the ROME_VERIFIED_2023 table. Subsequently, the values are matched, enabling correct binding of the SAMPLINGPOINT and SAMPLING_POINT_ID columns and allowing all three tables to be combined in a single query.

```
SELECT
    -- measurement data
    r.start_measurement, r.end_measurement, r.value, r.unit,

    -- pollutant information
    p.label AS pollutant_label,

    -- sampling point location data
    sp. country, sp.municipality, sp.air_quality_station_name,
```

```
    sp.air_quality_station_area, sp.air_quality_station_type,
    sp.latitude, sp.longitude

FROM rome_verified_2023 r
JOIN pollutants p
      ON p.pollutant_id = r.pollutant
JOIN sampling_points sp
      ON sp.sampling_point_id = SUBSTR(r.samplingpoint, 4)
WHERE r.value >= 0;
```

The aggregate query that links all three tables and computes annual NO_2 pollution averages (label 8) can also be finalized. This query displays pollution data and location attributes for measuring stations simultaneously, facilitating the interpretation of measurement results.

```
SELECT
    -- pollutant info
    p.label AS pollutant_label,

    -- sampling point (location data)
    sp.country, sp.municipality, sp.air_quality_station_name,
    sp.air_quality_station_area, sp.air_quality_station_type,
    sp.latitude, sp.longitude,

    -- year and aggregated value
    EXTRACT(YEAR FROM r.start_measurement) AS measurement_year,
    AVG(r.value) AS avg_pollutant_value

FROM rome_verified_2023 r
JOIN pollutants p ON p.pollutant_id = r.pollutant
JOIN sampling_points sp ON sp.sampling_point_id = SUBSTR(r.samplingpoint, 4)

WHERE r.value >= 0
  AND r.pollutant = 8

GROUP BY
    p.label,
    sp.country, sp.municipality, sp.air_quality_station_name,
    sp.air_quality_station_area, sp.air_quality_station_type,
    sp.latitude, sp.longitude,
    EXTRACT(YEAR FROM r.start_measurement)

ORDER BY measurement_year, sp.country, sp.municipality;
```

The results of the previous query are shown in Table 1, which indicates that the NO_2 mean annual limit value is 40 µg/m³ (micrograms per cubic meter) at the Fermi measuring station in Rome, on Piazza Enrico Fermi. The coordinates for this station (41.86399° N, 12.46957° E) can be viewed on Google Maps and in Street View (Figure 4.32).

Table 1: Overview of annual average NO_2 concentrations$_2$ per measuring station in Rome (2023)

POLLUTANT	COUNTRY	MUNICIPALITY	AIR QUALITY STATION	AREA	AIR QUALITY STATION TYPE	LATITUDE	LONGITUDE	YEAR	VALUE
Nitrogen (air)	Italy	CORSO FRANCIA - Roma (RM)	C.SO FRANC	urban	car	41,94745	12,46956	2023	35,61
Nitrogen (air)	Italy	LARGO MAGNA GRECIA - Roma (RM)	L.GO MAGNA GRECIA	urban	car	41,88307	12,50895	2023	33,96
Nitrogen (air)	Italy	LARGO PERESTRELLO - Roma (RM)	L.GO PERESTRELLO	urban	background	41,88602	12,54162	2023	26,22
Nitrogen (air)	Italy	LGO BELOLLI - Roma (RM)	CINECITTA	urban	background	41,85772	12,56865	2023	25,77
Nitrogen (air)	**Italy**	**P.ZZA E. FERMI, FRONTE CIVICO16 - Roma (RM)**	**FERMS**	**urban**	**car**	**41,86399**	**12,46957**	**2023**	**47,07**
Nitrogen (air)	Italy	Piazza Cairoli adiacente via dei Giubbonari - Roma (RM)	ARENUL	urban	background	41,89397	12,47539	2023	33,00
Nitrogen (air)	Italy	VIA AURELIA KM 20 CASTEL DI GUIDO - Roma (RM)	CASTEL DI GUIDÓ	rural	background	41,88945	12,26633	2023	8,17
indicates that the NO_2 mean annual limit value is 40 µg/m³ (micrograms per cubic meter) at the Fermi measuring station located in	Italy	VIA DELLA MELORIA, FRONTE CIVICO 27 - Roma (RM)	CIPRO	urban	background	41,90636	12,44759	2023	30,04
Nitrogen (air)	Italy	VIA MONTE CANDA, ADIACENTE CIVICO 12 - Roma (RM)	BUFALOTTA	urban	background	41,94765	12,53367	2023	29,94
Nitrogen (air)	Italy	VIA PAOLO EMILIO PAVOLINI 69/71 - Roma (R)	MALAGROT	suburban	background	41,87478	12,34559	2023	16,90
Nitrogen (air)	Italy	VIA SALARIA - Roma (RM)	VILLA ADA	urban	background	41,93289	12,50695	2023	19,76
Nitrogen (air)	Italy	VIA TENUTA DEL CAVALIERE 102 - Roma (RM)	TENUTA DEL CAVALIER	suburban	background	41,92947	12,65848	2023	23,89
Nitrogen (air)	Italy	VIA TIBURTINA, ADIACENTE CIVICO 654 - Roma (RM)	TIBURNIA	urban	car	41,91027	12,54885	2023	35,50



Figure 4.32 Google Street View at Fermi measuring station

Practical Exercises

- Connect to the Oracle Cloud Console.
- Open a new Autonomous Database instance to realize the data warehouse from.
- Create a bucket in Oracle Object Storage.
- Connect to the EEA headquarters and review the documentation and descriptions of data models (Medatdata, Vocabulary links).
- Download the parquet file with current air pollution measurements for your country and place it in your data tank.
- Load data into the data warehouse (measures from the container, pre-prepared pollution list files (Excel file POLLUTANTS, csv file SAMPLING_POINTS)
- Analyze measurements for PM2.5 particles. What are the sources of this pollution, and how does it affect people's health?
- Identify the locations where the highest levels of PM2.5 pollution were measured.
- Compare measured values with limit values.

Assessment

What are the key properties and benefits of data warehouses?

A. Enables analysis of large historical data sets
B. Optimized for rapid Real-Time Transactions
C. Unifies and connects data from multiple sources
D. Used as a replacement for reporting jobs on transactional databases

If you want to prepare meteorological data in a data warehouse to implement a simple star schema, what minimum set of tables would be required? (One correct response.)

A. Single Measurement Table and Single Measurement Location Table
B. Single measurement table, measurement sensor table and system operator table
C. Only one measurement table because all the information is already in it
D. All of the foregoing

What property best describes the so-called "raw" (raw) tables in the environmental data model?

A. Represent source, raw data, taken from external systems, without any further transformations
B. Contains temporary data downloaded from external systems
C. Store data from external systems that are designed for quick access and analysis
D. Aggregate query execution and statistics calculation are optimized

What steps best describe the data warehouse modeling process in the context of the environmental data case study presented?

A. Download Data from External Sources
B. Overview of structure and understanding of data (metadata, vocabulary), identification of key attributes and data cleansing
C. Table Association
D. All of the following

What are the core functions of the Data Load interface while working with Oracle Automatic Database Data Warehouse?

A. Automatic SQL Query Generation for Data Analysis
B. Importing data from an Object Storage bucket
C. Manage user accounts and security settings.
D. Optimize analytic SQL queries.
E. Import Local Files

References / Further Reading

[1] "Autonomous AI Database", Oracle. [Online]. Available: https://www.oracle.com/autonomous-database/ (accessed Dec. 9, 2025).

[2] "Oracle Cloud Infrastructure (OCI)", Oracle. [Online]. Available: https://www.oracle.com/cloud/ (accessed Dec. 9, 2025).

[3] "Amazon Redshift," AWS. [Online]. Available: https://aws.amazon.com/redshift/ (accessed Dec. 9, 2025).

[4] "BigQuery | AI data platform | Lakehouse | EDW", Google Cloud. [Online]. Available: https://cloud.google.com/bigquery (accessed Dec. 9, 2025).

[5] "Azure Synapse Analytics", Microsoft Azure. [Online]. Available: https://azure.microsoft.com/en-us/products/synapse-analytics (accessed Dec. 9, 2025).

[6] "Apache Hive", Apache Software Foundation. [Online]. Available: https://hive.apache.org/ (accessed Dec. 9, 2025).

[7] "Solutions Cloud Program", Oracle Academy. [Online]. Available: https://academy.oracle.com/en/solutions-cloud-program.html (accessed Dec. 9, 2025).

[8] "Oracle Cloud Free Tier", Oracle. [Online]. Available: https://www.oracle.com/cloud/free (accessed Dec. 9, 2025).

[9] "Oracle SQL Developer – Downloads". Available: https://www.oracle.com/database/sqldeveloper/technologies/download (accessed Dec. 9, 2025).

[10] European Environment Agency, Available: https://www.eea.europa.eu/en (accessed Dec. 9, 2025).

[11] European Environment Agency, Data Download Service, All Available: https://eeadmz1-downloads-webapp.azurewebsites.net (accessed Dec. 9, 2025).

[12] Agents for Data, "Parquet Merge Tool", Available: https://www.agentsfordata.com/parquet/merge (accessed Dec. 9, 2025).

[13] European Environment Information and Observation Network, Air Quality Pollutant Vocabulary, Available: https://dd.eionet.europa.eu/vocabulary/aq/pollutant (accessed Dec. 9, 2025).