1 Database Creation and Validation

To store the collected price data in a structured and persistent manner, a MySQL relational database was created programmatically using Python. This approach ensures reproducibility and allows the database to be easily deployed across different environments, such as a local machine or cloud infrastructure.

1.1 Database Schema

A database named `price_collection` was created, containing a single table named `PRICE`, as specified in the project requirements. Each row in the table represents the price offered by one seller for a given product on a specific date.

The table schema is defined as follows:

- `ID`: Auto-incremented primary key
- `Product`: Name of the product
- `Date`: Date on which the price was collected
- `Seller`: Seller or platform offering the product
- `Price`: Product price including delivery costs

To prevent duplicate entries, a unique constraint was enforced on the combination of `Product`, `Date`, and `Seller`.

1.2 Database Initialization

The database and table were created using a Python script that connects to the MySQL server and executes the corresponding SQL commands. The core SQL definition is shown below:

```
CREATE DATABASE IF NOT EXISTS price_collection;

CREATE TABLE IF NOT EXISTS PRICE (
    ID BIGINT AUTO_INCREMENT PRIMARY KEY,
    Product VARCHAR(255) NOT NULL,
    Date DATE NOT NULL,
    Seller VARCHAR(255) NOT NULL,
    Price DECIMAL(10,2) NOT NULL,
    UNIQUE (Product, Date, Seller)
);
```

This setup ensures that the database structure is created only once and can be safely reused across multiple executions of the ETL pipeline.

1.3 Sanity Testing

After creating the database, a sanity test was performed to verify correct functionality. The test involved inserting a sample record into the `PRICE` table and retrieving it using a `SELECT` query. Furthermore, the unique constraint was validated by attempting to insert a duplicate record with identical product, seller, and date values.

The successful execution of these tests confirmed that:

- The database connection was correctly established
- Data insertion and retrieval worked as expected
- Duplicate entries were effectively prevented

After verification, the test data was removed, leaving the database ready for use in the actual ETL process.