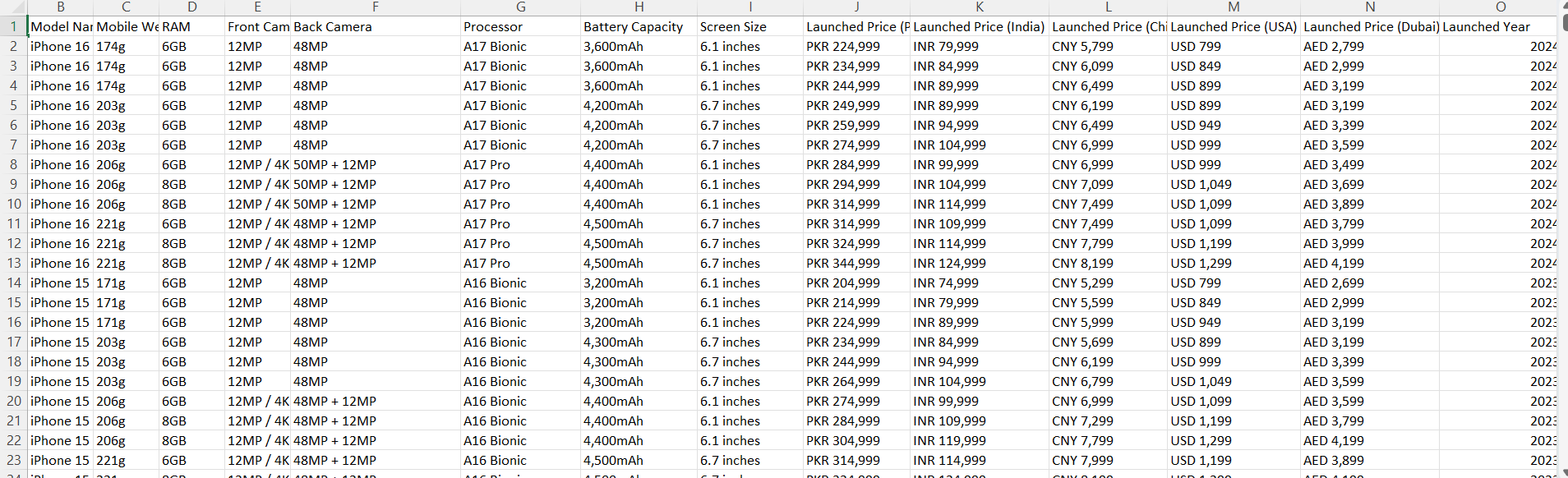
**ETL REPORT**

**Project Overview**

The objective of this ETL project was to prepare and integrate multi-source retail pricing data from various countries (e.g., China, India, Dubai, Pakistan, and the USA) for business intelligence purposes. The end goal was to visualize trends and insights using Tableau. The project follows a Star Schema with clearly defined fact and dimension tables. The original data was delivered as multiple CSV files representing entities derived from a normalized ERD.

1. Extract :

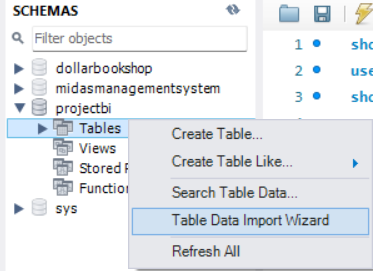
The extraction phase began with separating the dataset into several CSV files, each corresponding to a conceptual table in the ERD. These files were extracted from a zipped Excel archive and placed in a structured folder. Each file represented a distinct dimension or fact table such as BI Company, BI Product, BI USDPrice, and other country-specific pricing files.

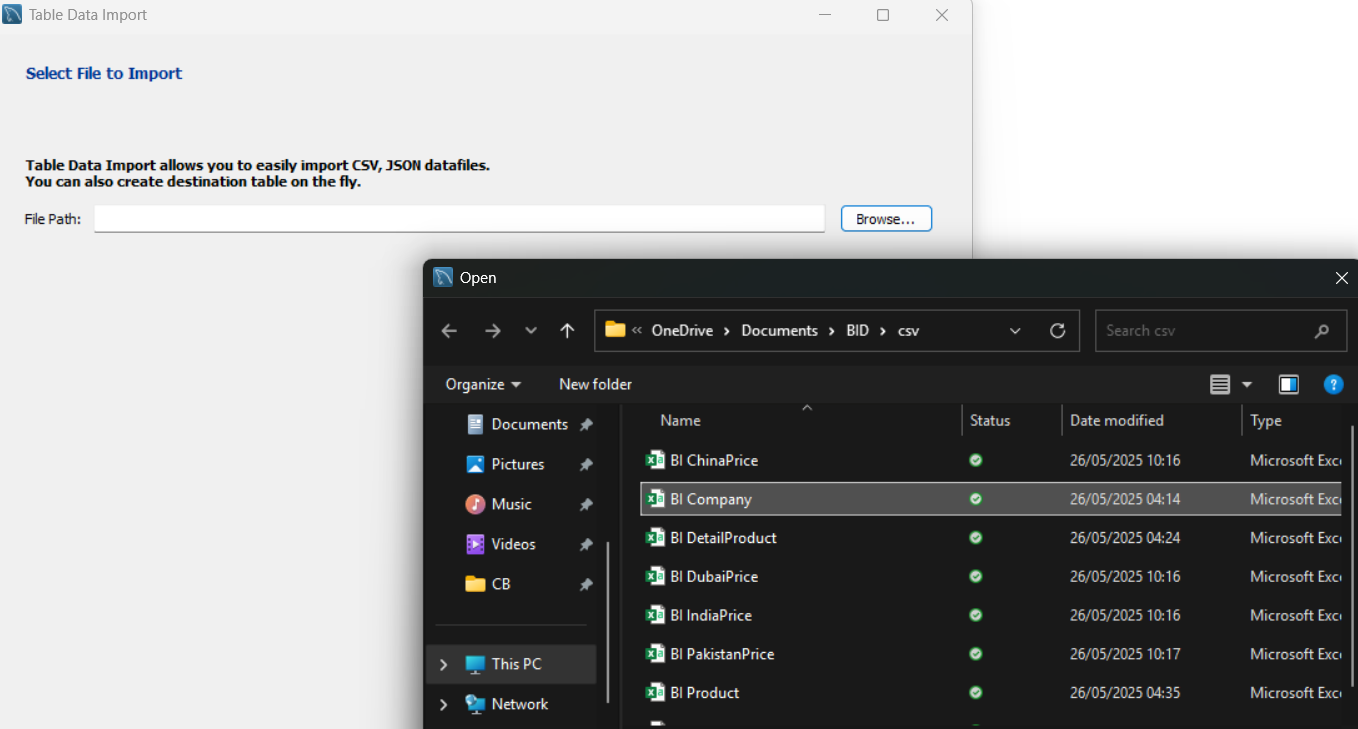
From:

To:

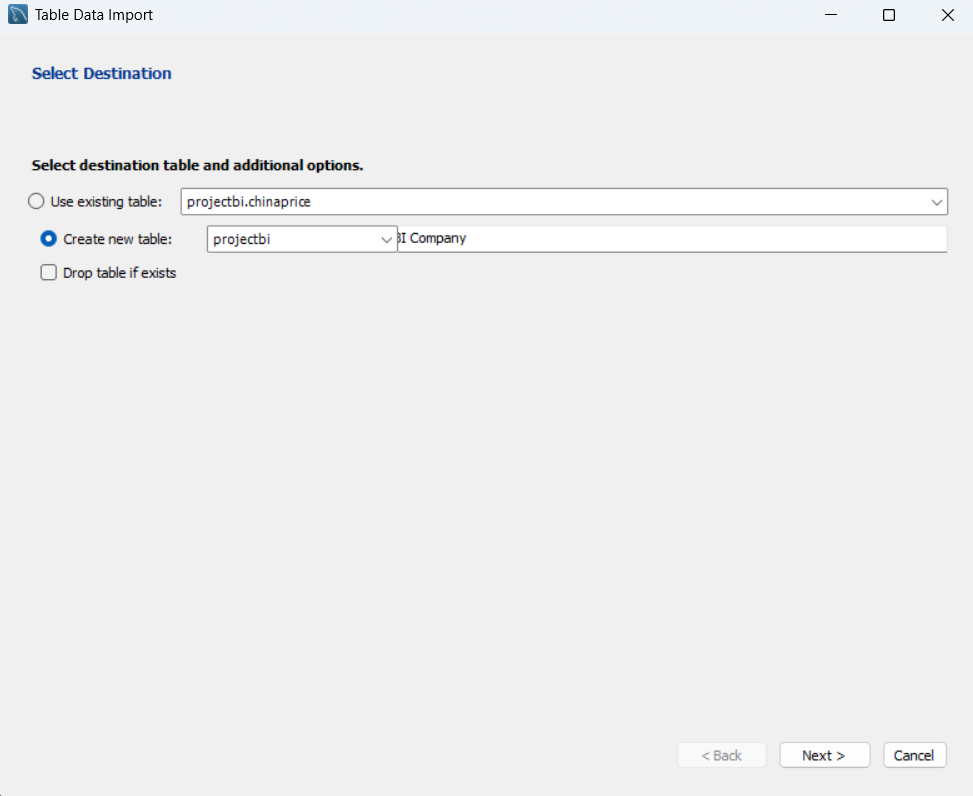
During extraction, it was crucial to ensure that the character encoding was preserved, as some files contained special characters that could corrupt the data if not read correctly. We used ISO-8859-1 encoding when importing the files into MySQL, ensuring the files were properly parsed and header formats maintained.

To import multiple CSV files into MySQL using MySQL Workbench, start by creating a database (e.g., BIProject) and define a table schema that matches the structure of one CSV file, such as BI\_Company. You can open the CSV in Excel or a text editor to check the column names and data types. In MySQL Workbench, right-click on the schema and choose **“Table Data Import Wizard”**. Select the CSV file, choose to create a new table (or map it to an existing one), and complete the import.

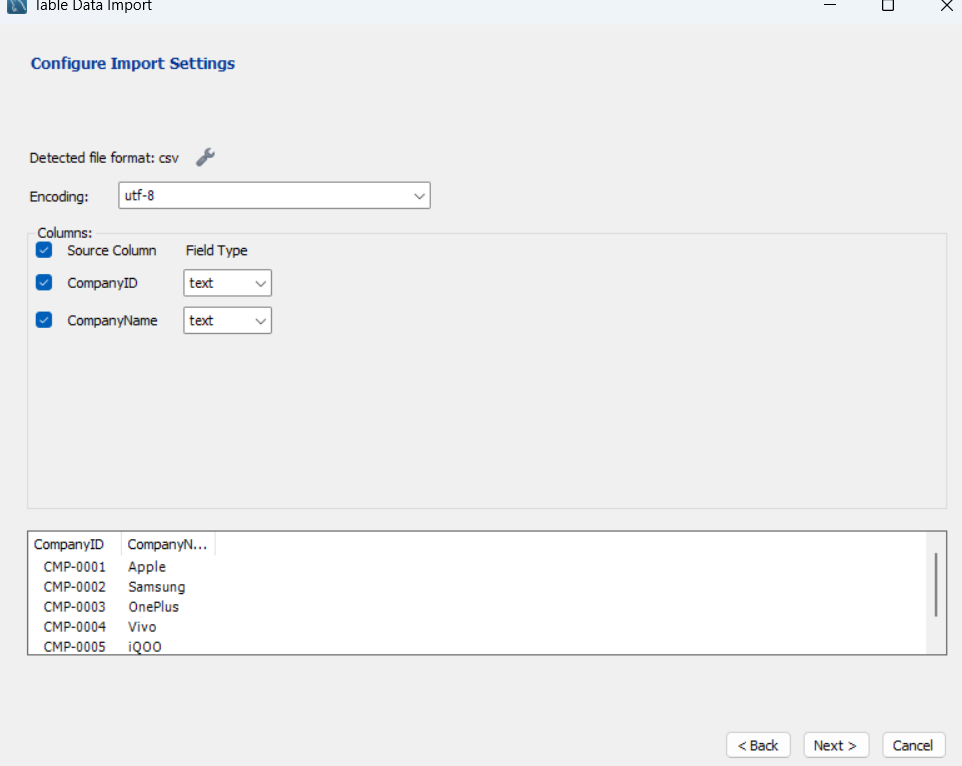




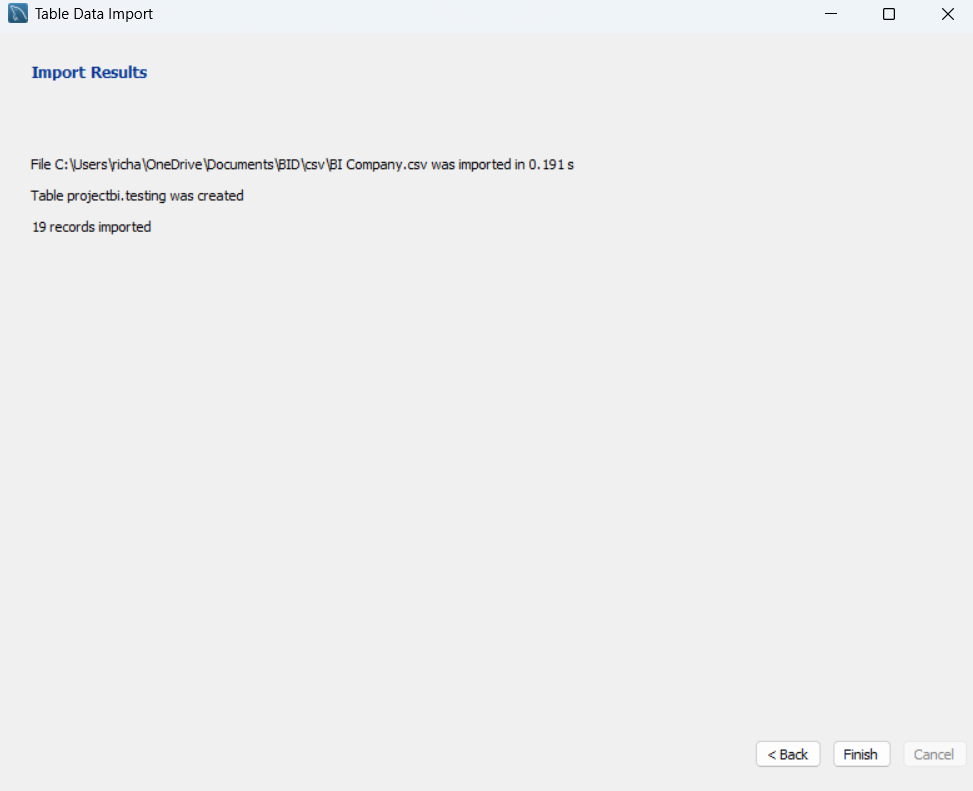
You can also change the name of the table that you are importing here:



Once you are done you can edit the fielt type of your data:



Once you are done this pop-up will appear:



When this pop-up appears, then the CSV file that you were importing is succesfully imported.

Once the data is loaded, use SQL statements to clean formatting (e.g., remove commas from numeric values) and set proper data types using ALTER TABLE. After importing all CSVs this way, define relationships between tables by adding primary keys and foreign keys with ALTER TABLE ... ADD CONSTRAINT. Repeat the import and setup process for the remaining CSV files, following the same steps. This approach ensures a clean, structured database ready for analysis or visualization in tools like Tableau. This will be done on the next step, which is Transform

1. Transform:

While editing the dataset during the transformation phase, several issues were encountered:

1. **Data Type Mismatches:** Many price columns were incorrectly read as strings due to inconsistent formatting (e.g., currency symbols, commas). These were corrected by cleaning and converting them into appropriate float data types.
2. **Attribute Renaming (Schema Alignment):** To maintain consistency with the finalized ERD, original column names from the source dataset were renamed using ALTER TABLE CHANGE COLUMN statements.
3. **Standardizing Data Formats:** Columns such as LaunchedYear were explicitly cast into appropriate types (INTEGER) using ALTER TABLE to enable consistent filtering, grouping, and temporal analysis.

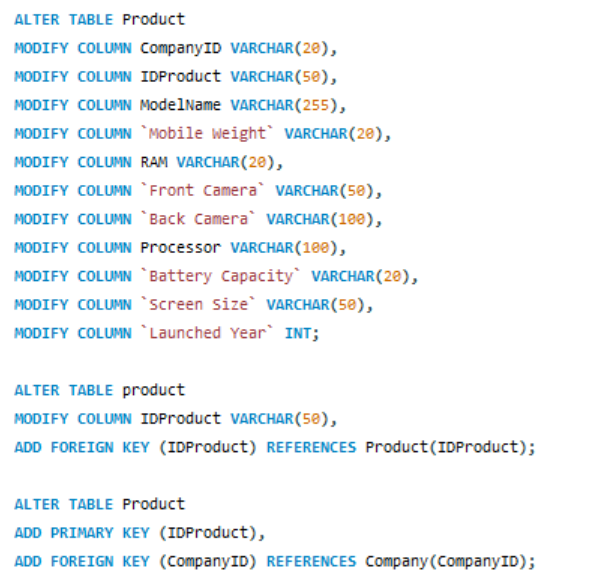


Fig.2 Attribute Renaming and Standardizing Data Formats

1. **Foreign Key and Primary Key Constraints:** Defined with ADD FOREIGN KEY and ADD PRIMARY KEY statements to ensure referential integrity across tables.

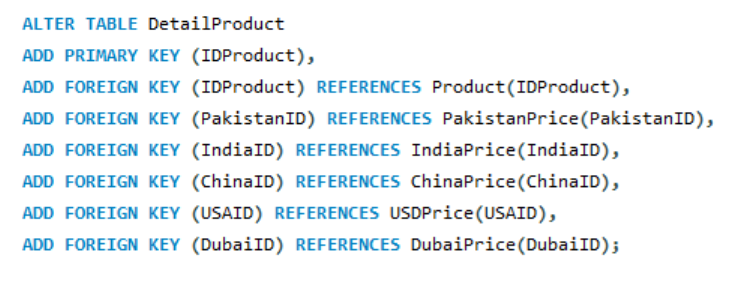


Fig. 3 Alter Table For Foreign Key

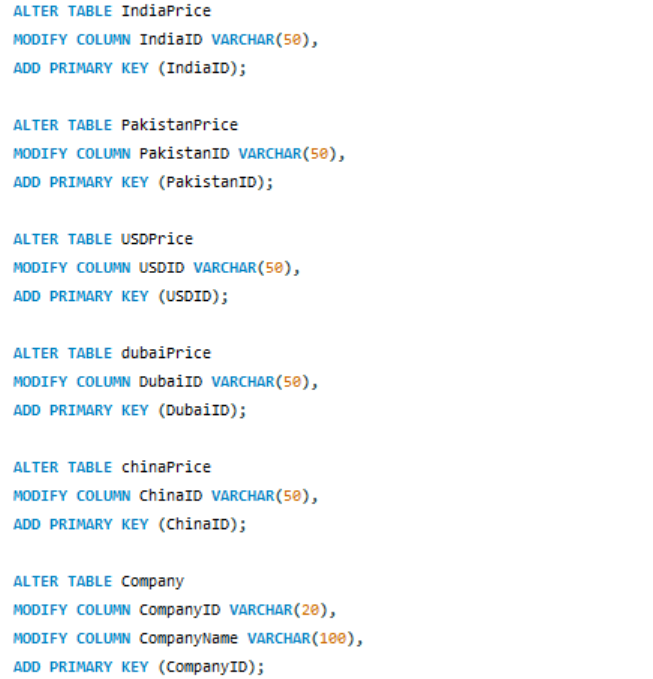


Fig. 4 Alter Table For Primary Key

1. **Cleaning and Validation:** Duplicates and null entries were identified and removed. String-to-float conversion errors and other data inconsistencies were addressed through validation scripts.
2. **Data Enrichment and Aggregation:** Although minimal, some data was aggregated (e.g., average prices per country) and enriched with additional flags or codes as needed for improved analysis during visualization.
3. Load:

The final phase of the ETL process involved loading the cleaned and transformed data into the final reporting schema. As the structure of the MySQL data warehouse closely mirrored the ERD and followed a Star Schema layout, there was no need for schema restructuring. Data from staging tables were loaded into production tables using straightforward INSERT INTO ... SELECT \* FROM ... queries. All fact and dimension tables were loaded in a sequence that respected their relational dependencies—dimension tables first, followed by fact tables. This sequencing ensured that all foreign key relationships could be successfully enforced during or after the data load.

1. ETL Tools:

The entire ETL process was executed using SQL scripts written and run directly in MySQL. This approach provided full control over each stage of the pipeline and enabled efficient debugging and tracking of changes. Supplementary data inspection and cleaning were performed using spreadsheet tools and, in some cases, Python (e.g., for character encoding inspection and data preview). Visualization was performed in Tableau, which connected directly to the MySQL database for live querying and dashboard generation. This SQL-based ETL process was well-suited to the scope of the project. It allowed for a transparent, cost-effective, and scalable data pipeline without the need for external ETL platforms such as Talend or Pentaho. The simplicity and precision of SQL also made it easier to implement data integrity rules and custom validation logic.

**Challenges and Solutions**

Throughout the ETL process, several challenges were encountered. One of the most significant was data inconsistency in terms of formatting and encoding. Currency fields were particularly problematic due to mixed use of symbols and thousands separators. These were resolved using REPLACE() functions followed by type casting. Another common challenge was duplicate key values, which prevented the enforcement of primary key constraints. This was addressed by identifying duplicates using aggregation queries and removing them before applying constraints. Finally, missing foreign key references created issues when establishing relationships between tables. These were resolved by either deleting orphaned records or inserting the missing references into the appropriate parent tables with placeholder data.

**Conclusion**

This ETL project successfully transformed a multi-sheet Excel dataset into a clean, relational data warehouse within MySQL. The structured data enabled robust dashboard development in Tableau, supporting business intelligence use cases such as cross-country price comparisons, company-level performance tracking, and product-level analysis. The project demonstrated the effectiveness of using SQL as a lightweight but powerful ETL tool and laid a strong foundation for scaling to larger datasets and more advanced analytics in the future.