E-Commerce Feedback and Rating Analysis

BUSADM_816

~ Meenakshi Sambari

Table of Contents

1.1 Project Title

"**E-Commerce Feedback and Rating Analysis**"

The project aims to analyze feedback and ratings from e-commerce customers, offering insights into product and department performance to support data-driven decision-making.

1.2 Project Objective

The primary objective of this project is to design a comprehensive business intelligence (BI) solution to analyze women's feedback and ratings for an e-commerce platform. The analysis focuses on uncovering patterns, trends, and actionable insights to enhance product offerings, improve customer satisfaction, and optimize operational efficiency.

Specific objectives include:

1. Identifying trends in product ratings and customer feedback by demographic and categorical breakdowns.
2. Developing data visualizations to aid strategic decision-making by stakeholders.
3. Creating performance metrics for departments, divisions, and classes to identify improvement opportunities.
4. Supporting targeted marketing and product development through actionable insights.

1.3 Business Case Definition

E-commerce platforms thrive on customer satisfaction and operational efficiency. This project addresses the need to evaluate customer sentiments, identify high-performing departments or classes, and understand areas requiring improvement.

By analyzing ratings and feedback metrics across various dimensions such as age groups, divisions, and product classes, the project helps decision-makers:

1. Enhance product quality based on customer input.
2. Refine marketing strategies to target specific demographics.
3. Drive innovation in product development.

Key Business Questions:

- Which departments or product classes receive the highest ratings and feedback?
- What trends exist in customer preferences across age groups?
- Are certain divisions underperforming compared to others?
- How does positive feedback correlate with product ratings?

1.4 Key Performance Indicators (KPIs)

The following KPIs were derived from the visualizations created and represent critical metrics for monitoring and decision-making:

1. **Average Rating by Division, Department, and Class**
    - *Purpose:* Evaluate overall customer satisfaction trends at different organizational levels.
    - *Visualization:* Bar/column charts displaying average ratings per hierarchy.
2. **Average Rating by Age Group**
    - *Purpose:* Understand customer satisfaction across different demographics.
    - *Visualization:* Line or bar chart comparing age groups.
3. **Sum of Positive Feedback Count by Department**
    - *Purpose:* Highlight departments receiving the most positive feedback.
    - *Visualization:* Donut chart or grouped bar chart.
4. **Sum of Ratings by Department**
    - *Purpose:* Measure total customer engagement per department.
    - *Visualization:* Bar chart or table.
5. **Sum of Ratings by Division**
    - *Purpose:* Understand performance across divisions.
    - *Visualization:* Bar or column chart.
6. **Sum of Positive Feedback Count by Title**

- *Purpose:* Assess product titles receiving the highest positive feedback.
- *Visualization:* Pivot table or table.

7. **Sum of Ratings by Class**
    - *Purpose:* Analyze satisfaction trends across product classes.
    - *Visualization:* Pie chart or table.
8. **Sum of Positive Feedback Count and Ratings by Age Group**
    - *Purpose:* Identify correlations between age groups and their product feedback.
    - *Visualization:* Combined bar chart.

1.5 Scope and Limitations

Scope:

- Analyze structured data consisting of customer ratings, feedback counts, and demographic details.
- Create dashboards and reports with aggregated views of metrics such as average ratings and feedback counts.
- Provide actionable insights for stakeholders to improve e-commerce operations.

Limitations:

1. The analysis is constrained to the dataset provided and may not capture real-time trends.
2. Data quality issues like missing or inconsistent entries may affect accuracy.
3. The scope of insights is limited to available variables and does not account for external factors like competitor performance or macroeconomic trends.
4. The project assumes a static dataset and does not implement predictive analytics for future trends.

2.1.1 Source Data Description

The primary dataset used in this project is derived from an **e-commerce customer feedback dataset**, which includes key details about customer reviews, product categories, and demographic attributes. The dataset was structured and sourced as follows:

- **Dataset Name:** Women's E-commerce Clothing Reviews Dataset
- **Source:** Kaggle (Public Dataset) https://www.kaggle.com/datasets/nicapotato/womens-ecommerce-clothing-reviews/code
- **Data Format:** CSV (Comma-Separated Values)
- **Key Features:**
    - **Customer Feedback Details:** Textual reviews, ratings, positive feedback count, and recommendation indicators.
    - **Demographic Data:** Age of customers, allowing for segmentation analysis.
    - **Product Hierarchy:** Division, department, and class identifiers for hierarchical analysis.
    - **Product Metadata:** Product titles and clothing IDs for product-level insights.

2.1.2 Metadata Overview

The dataset includes the following key fields, which align with the data modeling process:

Field Name	Description
clothing_id	Unique identifier for each product.
age	Age of the customer providing the review.
title	Product title associated with the review.
review_text	Textual feedback provided by the customer.

Field Name	Description
rating	Numeric rating (1–5) assigned by the customer.
recommended_ind	Binary flag indicating whether the customer recommends the product (0 or 1).
positive_feedback_count	Count of positive feedback ("helpful votes") received for the review.
division_name	Identifier for the division under which the product is classified.
department_name	Identifier for the department within the division.
class_name	Identifier for the class within the department.

2.1.3 Importance of Source Systems

The dataset was designed to answer critical business questions, focusing on customer satisfaction and product performance. These source systems provide a robust starting point for creating a data warehouse, which supports slicing and dicing the data across various dimensions such as **age, ratings, and feedback counts**.

2.2 Data Quality Assessment

A thorough review of the source data revealed several data quality considerations:

1. **Completeness:**
    - Certain fields, such as division_id, department_id, and class_id, contained null values, which required data transformation during the ETL process.
2. **Consistency:**
    - The dataset ensured consistency in its numeric and categorical fields, but unstructured fields like review_text required additional preprocessing to maintain clarity.

3. **Accuracy:**
    - Fields like rating and positive_feedback_count were checked for outliers to ensure reliable insights.
4. **Integrity:**
    - Relationships between dimensions (division, department, class) and facts (ratings, feedback counts) were validated against predefined hierarchical structures.

2.3 Preprocessing of Source Data

Before integrating the data into the ETL pipeline, initial preprocessing steps were performed, including:

1. **Removing Duplicates:**
    - Ensuring no repeated records existed in the dataset, especially for clothing_id.
2. **Handling Missing Values:**
    - Null values in categorical fields like division_id were flagged and either removed or imputed with default values during the transformation phase.
3. **Data Validation:**
    - Numeric fields like rating and positive_feedback_count were validated to ensure they fell within acceptable ranges.
4. **Data Formatting:**
    - Converting date fields (if any) and numeric data into a standard format compatible with the database schema.

2.4 Data Integration with Source Systems

The source systems provided data that seamlessly integrated into the following dimensions and fact tables:

1. **Dimension Tables:**
    - division_dim

- department_dim
- class_dim

2. **Fact Table:**
    - product_fact containing metrics like rating and positive_feedback_count.

This structure facilitated multi-dimensional analysis and aggregation, forming the backbone for data visualization and reporting.

3.1 ETL Plan Overview

The ETL process consisted of the following stages:

1. **Extraction:** Reading the source CSV file containing raw data on customer reviews, product information, and demographics.
2. **Transformation:** Cleaning, preprocessing, and splitting the raw data into normalized tables to align with a dimensional modeling approach.
3. **Loading:** Populating the transformed data into a SQL database with proper relationships defined among dimension and fact tables.

3.2 Extraction Process

The extraction phase involved reading the raw data from a single CSV file (**Women's E-commerce Clothing Reviews**) using Python.

Steps:

1. **File Identification:**
    - The source file was a CSV named ecommerce_clothing_reviews.csv, containing all the raw data fields.
2. **Extraction Tools Used:**
    - **Python Libraries:** Pandas was used to read and analyze the CSV data.
    - **Validation:** Initial checks were performed to ensure the file had no encoding issues and all fields were properly delimited.

```
import pandas as pd
 # Load the CSV file into a Pandas DataFrame
data = pd.read_csv(‘cleaned_Womens_Clothing_E-Commerce_Reviews.csv')
 # Display the first few rows to understand the structure
print(data.head())
```

3.3 Transformation Process

The transformation phase was crucial to ensuring the data was ready for integration into the relational database.

Key Transformations:

1. **Normalization:**
    - The data was divided into multiple tables: division_dim, department_dim, class_dim, and product_fact.
    - This approach reduced data redundancy and maintained the integrity of relationships among different entities.
2. **Creating Unique Dimensions:**
    - Unique values for division_name, department_name, and class_name were extracted and assigned primary keys.

SQL for Dimension Creation:

```
CREATE TABLE division_dim (
    division_id INT AUTO_INCREMENT PRIMARY KEY,
    division_name VARCHAR(255) UNIQUE
);


CREATE TABLE department_dim (
    department_id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    department_name VARCHAR(255) UNIQUE
);

CREATE TABLE class_dim (
    class_id INT AUTO_INCREMENT PRIMARY KEY,
    class_name VARCHAR(255) UNIQUE
);
select count(*) from class_dim;
```

Fact Table Transformation:

- Data fields like clothing_id, age, rating, positive_feedback_count, and review_text were included in the product_fact table.
- Foreign keys were added to connect the product_fact table to the dimension tables.

```
CREATE TABLE product_fact (
    product_id INT AUTO_INCREMENT PRIMARY KEY,
    clothing_id INT NOT NULL,
    age INT,
    title VARCHAR(255),
    review_text TEXT,
    rating INT,
    recommended_ind TINYINT(1),
    positive_feedback_count INT,
    division_id INT,
    department_id INT,
    class_id INT,
    FOREIGN KEY (division_id) REFERENCES division_dim(division_id),
```

```
    FOREIGN KEY (department_id) REFERENCES
department_dim(department_id),
    FOREIGN KEY (class_id) REFERENCES class_dim(class_id)
);
```

Data Cleaning:

- **Null Value Handling:** Rows with missing foreign keys were flagged using the query

```
SELECT * FROM product_fact WHERE division_id IS NULL OR department_id
IS NULL OR class_id IS NULL;
```

Duplicate Removal: Duplicates in the clothing_id field were detected and removed using:

```
SELECT clothing_id, COUNT(*) FROM product_fact GROUP BY clothing_id
HAVING COUNT(*) > 1;
```

3.4 Loading Process

The final phase involved loading the transformed data into the SQL database.

Database Creation:

```
CREATE DATABASE women;
USE women;
```

Data Insertion:

- Python scripts were used to populate the tables with cleaned and transformed data.

```
import pandas as pd
import mysql.connector
```

```
# Database connection
conn = mysql.connector.connect(
    host='localhost',
    user='root',
    password='Pundarikam14@',
    database='women'
)
cursor = conn.cursor()

# Load the main CSV file
df = pd.read_csv('C:/Users/meena/Desktop/cleaned_Womens_Clothing_E-Commerce_Reviews.csv')

# Standardize column names
df.columns = df.columns.str.strip().str.lower()
print("Columns in DataFrame:", df.columns)

# Ensure expected columns exist
required_columns = ['division_name', 'department_name', 'class_name', 'clothing_id', 'age', 'title', 'review_text', 'rating', 'recommended_ind', 'positive_feedback_count']
missing_columns = [col for col in required_columns if col not in df.columns]
if missing_columns:
    raise KeyError(f"Missing columns in the CSV: {missing_columns}")

# Load dimension data
def load_dimension(df, table_name, column_name):
    unique_values = df[[column_name]].drop_duplicates()
```

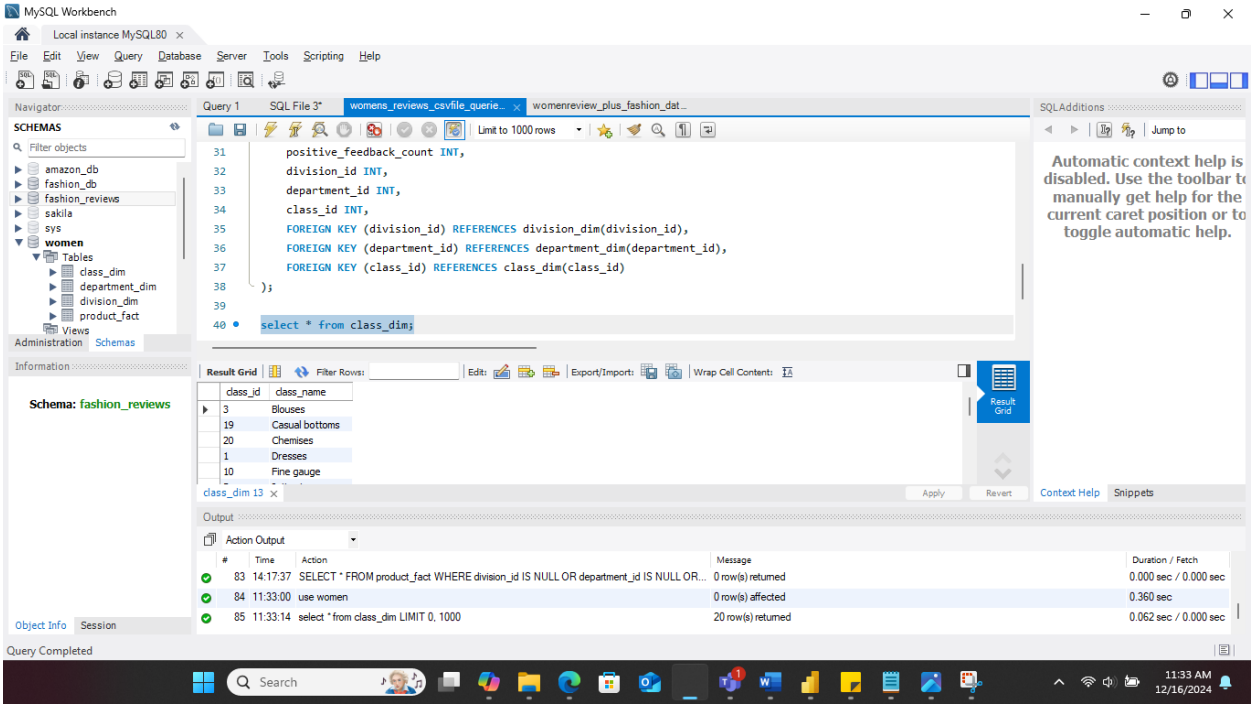
```
    for value in unique_values[column_name]:
        cursor.execute(f"INSERT IGNORE INTO {table_name} ({column_name}) VALUES (%s)", (value,))
    conn.commit()

# Load dimensions
load_dimension(df, 'division_dim', 'division_name')
load_dimension(df, 'department_dim', 'department_name')
load_dimension(df, 'class_dim', 'class_name')

# Load fact table
for _, row in df.iterrows():
    # Fetch foreign keys
    cursor.execute("SELECT division_id FROM division_dim WHERE division_name = %s", (row['division_name'],))
    division_id = cursor.fetchone()
    cursor.execute("SELECT department_id FROM department_dim WHERE department_name = %s", (row['department_name'],))
    department_id = cursor.fetchone()
    cursor.execute("SELECT class_id FROM class_dim WHERE class_name = %s", (row['class_name'],))
    class_id = cursor.fetchone()

    if division_id and department_id and class_id:
        # Insert into fact table
        cursor.execute('''
            INSERT INTO product_fact (clothing_id, age, title, review_text, rating, recommended_ind, positive_feedback_count, division_id, department_id, class_id)
```

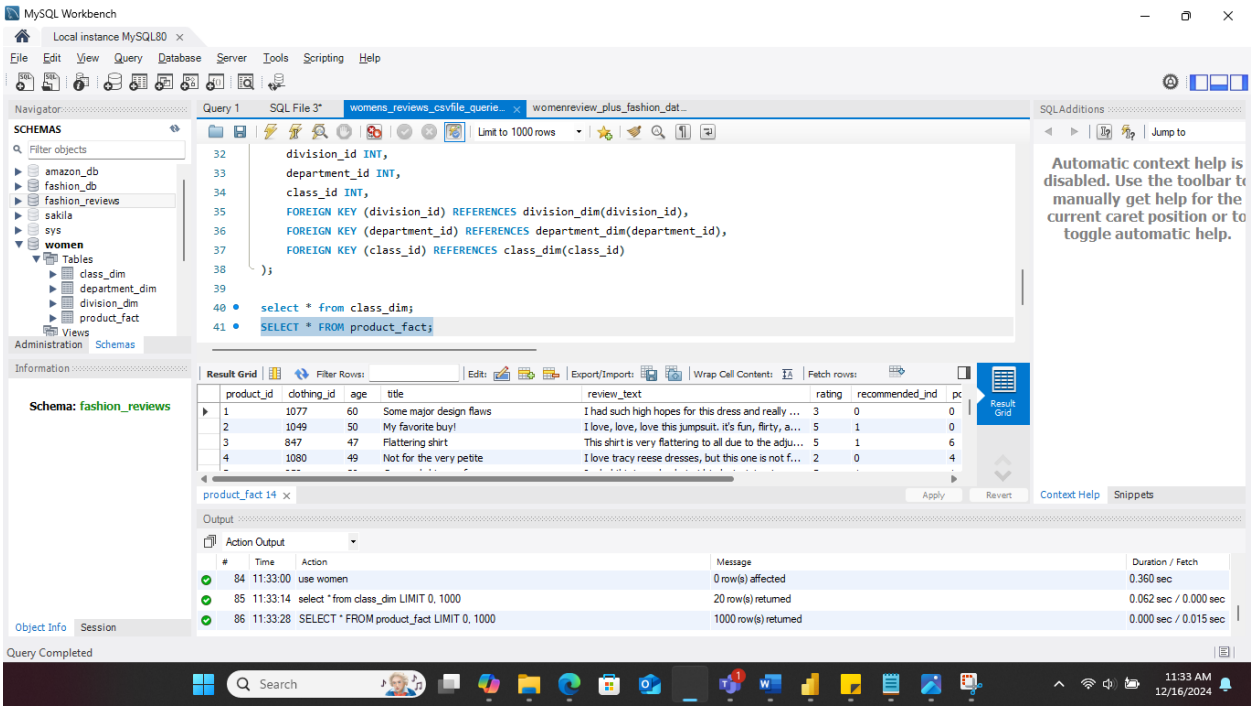
```
            VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
        ''', (
            row['clothing_id'], row['age'], row['title'], row['review_text'], row['rating'],
            row['recommended_ind'], row['positive_feedback_count'], division_id[0],
department_id[0], class_id[0]
        ))
        conn.commit()
    else:
        print(f"Skipping row due to missing foreign keys: {row.to_dict()}")

# Close connection
cursor.close()
conn.close()
```

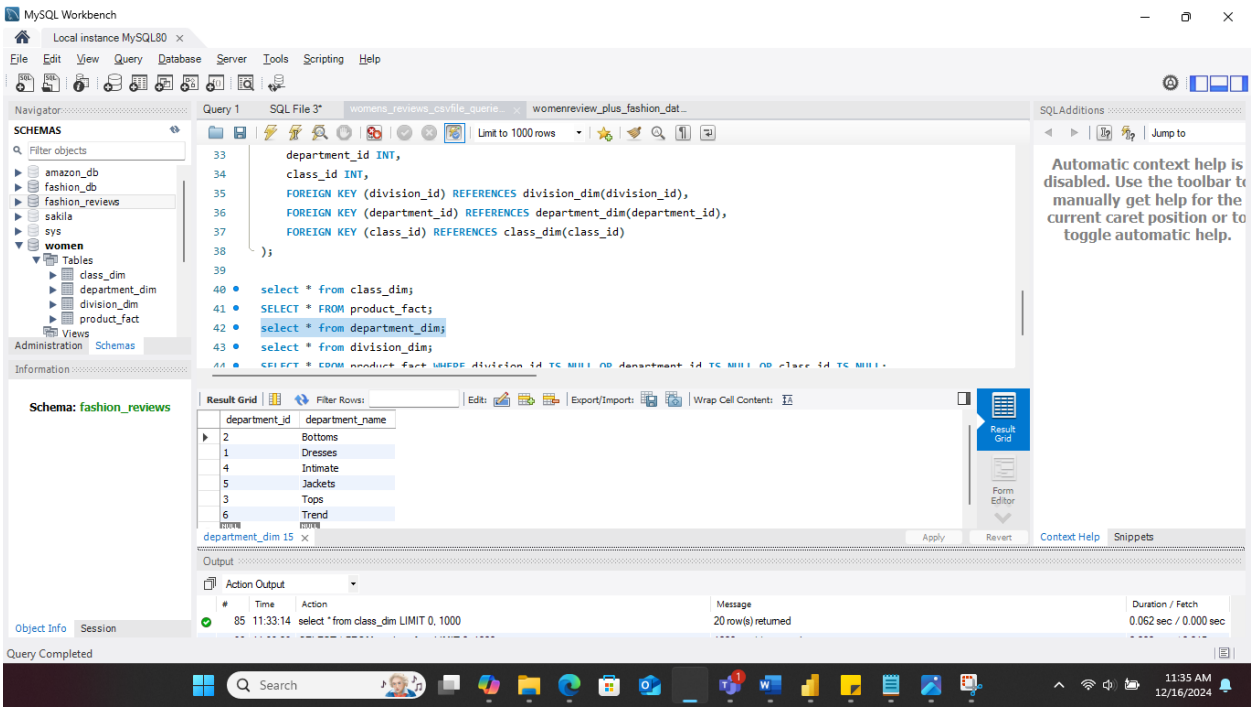
Verification:

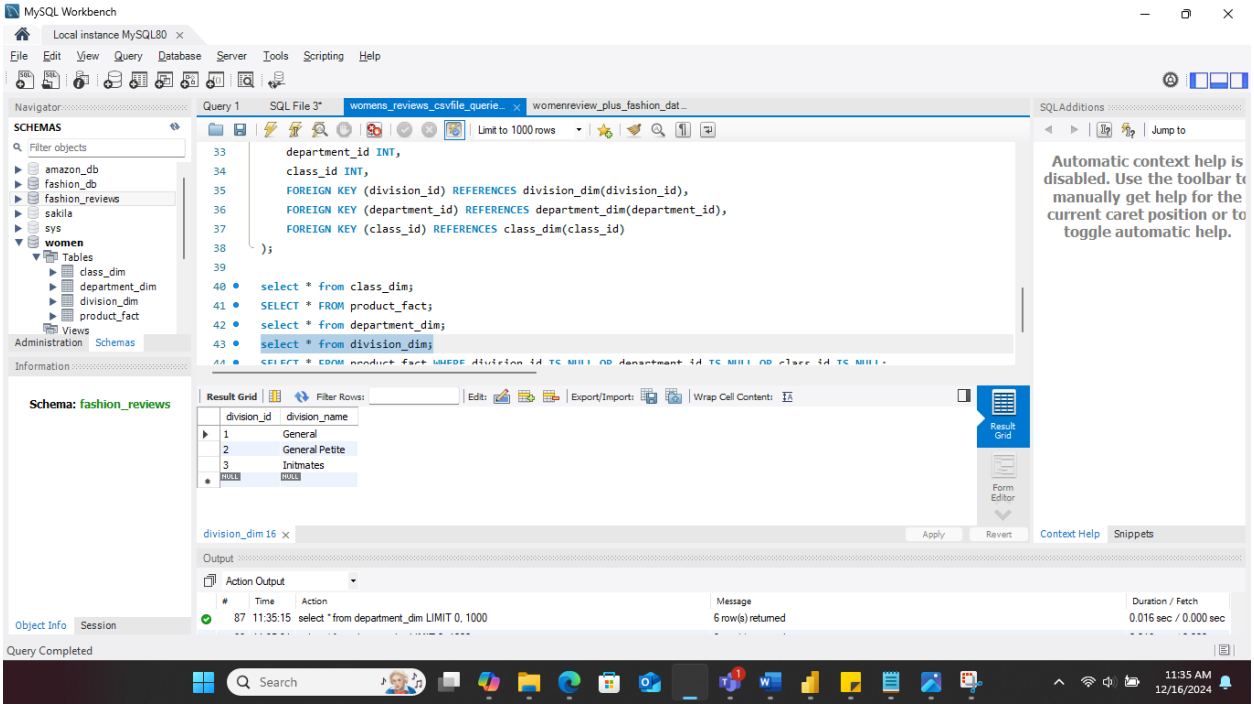
- Queries were executed to verify successful data insertion:

```
SELECT * FROM division_dim;
SELECT * FROM department_dim;
SELECT * FROM class_dim;
SELECT * FROM product_fact;
```


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
amazon_db
fashion_db
fashion_reviews
sakila
sys
women
Tables
class_dim
department_dim
division_dim
product_fact
Views
Administration Schemas
Information
Schema: fashion_reviews
Object Info Session
Query 1 SQL File 3* womens_reviews_csvfile_querie... womenreview_plus_fashion_dat...
Limit to 1000 rows
positive_feedback_count INT,
division_id INT,
department_id INT,
class_id INT,
FOREIGN KEY (division_id) REFERENCES division_dim(division_id),
FOREIGN KEY (department_id) REFERENCES department_dim(department_id),
FOREIGN KEY (class_id) REFERENCES class_dim(class_id)
);
select * from class_dim;
Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:
class_id class_name
3 Blouses
19 Casual bottoms
20 Chemises
1 Dresses
10 Fine gauge
class_dim 13
Apply Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
83 14:17:37 SELECT * FROM product_fact WHERE division_id IS NULL OR department_id IS NULL OR... 0 row(s) returned 0.000 sec / 0.000 sec
84 11:33:00 use women 0 row(s) affected 0.360 sec
85 11:33:14 select * from class_dim LIMIT 0, 1000 20 row(s) returned 0.062 sec / 0.000 sec
Query Completed
Search
11:33 AM
12/16/2024


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
amazon_db
fashion_db
fashion_reviews
sakila
sys
women
Tables
class_dim
department_dim
division_dim
product_fact
Views
Administration Schemas
Information
Schema: fashion_reviews
Object Info Session
Query 1 SQL File 3* womens_reviews_csvfile_querie... womenreview_plus_fashion_dat...
Limit to 1000 rows
division_id INT,
department_id INT,
class_id INT,
FOREIGN KEY (division_id) REFERENCES division_dim(division_id),
FOREIGN KEY (department_id) REFERENCES department_dim(department_id),
FOREIGN KEY (class_id) REFERENCES class_dim(class_id)
);
select * from class_dim;
SELECT * FROM product_fact;
Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content: Fetch rows:
product_id clothing_id age title review_text rating recommended_ind pc
1 1077 60 Some major design flaws I had such high hopes for this dress and really ... 3 0 0
2 1049 50 My favorite buy! I love, love, love this jumpsuit. it's fun, flirty, a... 5 1 0
3 847 47 Flattering shirt This shirt is very flattering to all due to the adju... 5 1 6
4 1080 49 Not for the very petite I love tracy reese dresses, but this one is not f... 2 0 4
product_fact 14
Apply Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
84 11:33:00 use women 0 row(s) affected 0.360 sec
85 11:33:14 select * from class_dim LIMIT 0, 1000 20 row(s) returned 0.062 sec / 0.000 sec
86 11:33:28 SELECT * FROM product_fact LIMIT 0, 1000 1000 row(s) returned 0.000 sec / 0.015 sec
Query Completed
Search
11:33 AM
12/16/2024





3.5 Tools and Technologies Used in ETL

The following tools and technologies were employed during the ETL process:

1. **Data Extraction:**
    - **Python:** For reading and preprocessing the CSV file.
    - **Pandas Library:** For cleaning and structuring the data.
2. **Data Transformation:**
    - **SQL Queries:** To create and normalize the database schema.
    - **Python:** To handle advanced cleaning and table creation logic.
3. **Data Loading:**
    - **MySQL Database:** For storing the processed data in structured tables.
    - **SQLAlchemy:** For connecting Python to the MySQL database.

4. Data Storage

This section describes the selection of the data storage solution and the implementation of a robust database structure to support efficient querying, data integrity, and scalability.

4.1 Selection of Data Storage Solution

Chosen Solution: MySQL Database

MySQL was chosen as the data storage solution for this project due to the following reasons:

1. **Relational Data Structure:**
    - The project required a structured, relational model to capture the relationships between products, divisions, departments, and classes.
2. **Scalability and Reliability:**
    - MySQL supports large datasets and provides reliable data management for enterprise-level applications.
3. **Ease of Use:**
    - MySQL offers a straightforward setup and compatibility with Python through libraries like SQLAlchemy, making the ETL process seamless.

4. **Foreign Key Support:**
    - Foreign key constraints in MySQL ensure referential integrity, which was crucial for maintaining consistent relationships between the fact and dimension tables.
5. **Compatibility with Tools:**
    - MySQL is widely used and integrates well with BI tools for visualization and reporting.

4.2 Database Structure

To support the analytical requirements of the project, we implemented a star schema model. This structure allowed for efficient querying and simplified data aggregation. The schema consisted of one fact table and three dimension tables, each carefully designed to capture specific attributes of the dataset.

Database Name:

- **women**: This database stored all the tables related to the project.

Fact Table: product_fact

The fact table contained quantitative data and foreign keys linking to the dimension tables. This table was central to the star schema and facilitated complex analysis.

- **Columns:**
    - product_id: Primary key.
    - clothing_id: Unique identifier for products.
    - age: Age of the reviewer.
    - title: Title of the review.
    - review_text: Full review text.
    - rating: Rating assigned by the customer.
    - recommended_ind: Indicator (1/0) for recommendation.
    - positive_feedback_count: Number of positive feedbacks received.

- Foreign Keys:
  - division_id → division_dim
  - department_id → department_dim
  - class_id → class_dim

SQL for product_fact:

```
CREATE TABLE product_fact (
    product_id INT AUTO_INCREMENT PRIMARY KEY,
    clothing_id INT NOT NULL,
    age INT,
    title VARCHAR(255),
    review_text TEXT,
    rating INT,
    recommended_ind TINYINT(1),
    positive_feedback_count INT,
    division_id INT,
    department_id INT,
    class_id INT,
    FOREIGN KEY (division_id) REFERENCES division_dim(division_id),
    FOREIGN KEY (department_id) REFERENCES
department_dim(department_id),
    FOREIGN KEY (class_id) REFERENCES class_dim(class_id)
);
```

Dimension Tables

1. **division_dim:**
   - Captured unique division information.

- Columns: division_id, division_name
- Each division_name was extracted from the raw data and normalized into a unique table.

SQL:

```
CREATE TABLE division_dim (
    division_id INT AUTO_INCREMENT PRIMARY KEY,
    division_name VARCHAR(255) UNIQUE
);
```

department_dim:

- Represented unique department details.
- Columns: department_id, department_name

SQL:

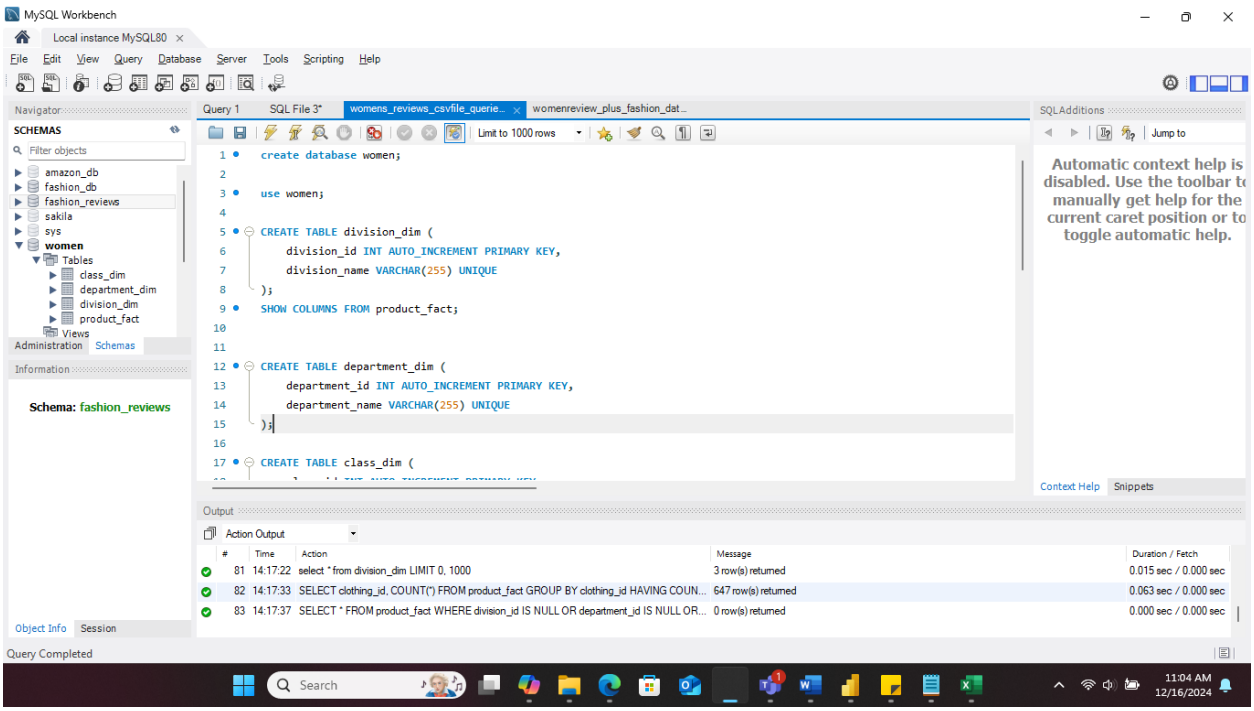
```
CREATE TABLE department_dim (
    department_id INT AUTO_INCREMENT PRIMARY KEY,
    department_name VARCHAR(255) UNIQUE
);
```

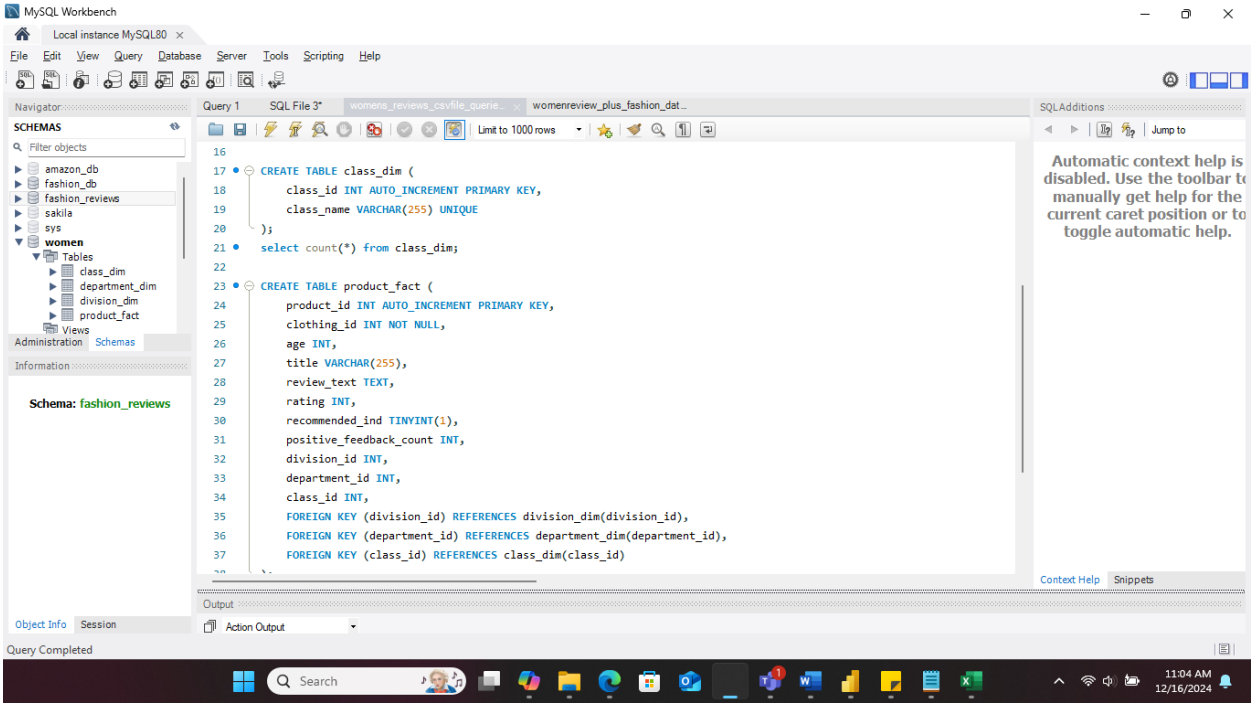
class_dim:

- Contained unique class information related to the product.
- Columns: class_id, class_name

SQL:

```
CREATE TABLE class_dim (
    class_id INT AUTO_INCREMENT PRIMARY KEY,
    class_name VARCHAR(255) UNIQUE
);
```





Normalization and Data Integrity

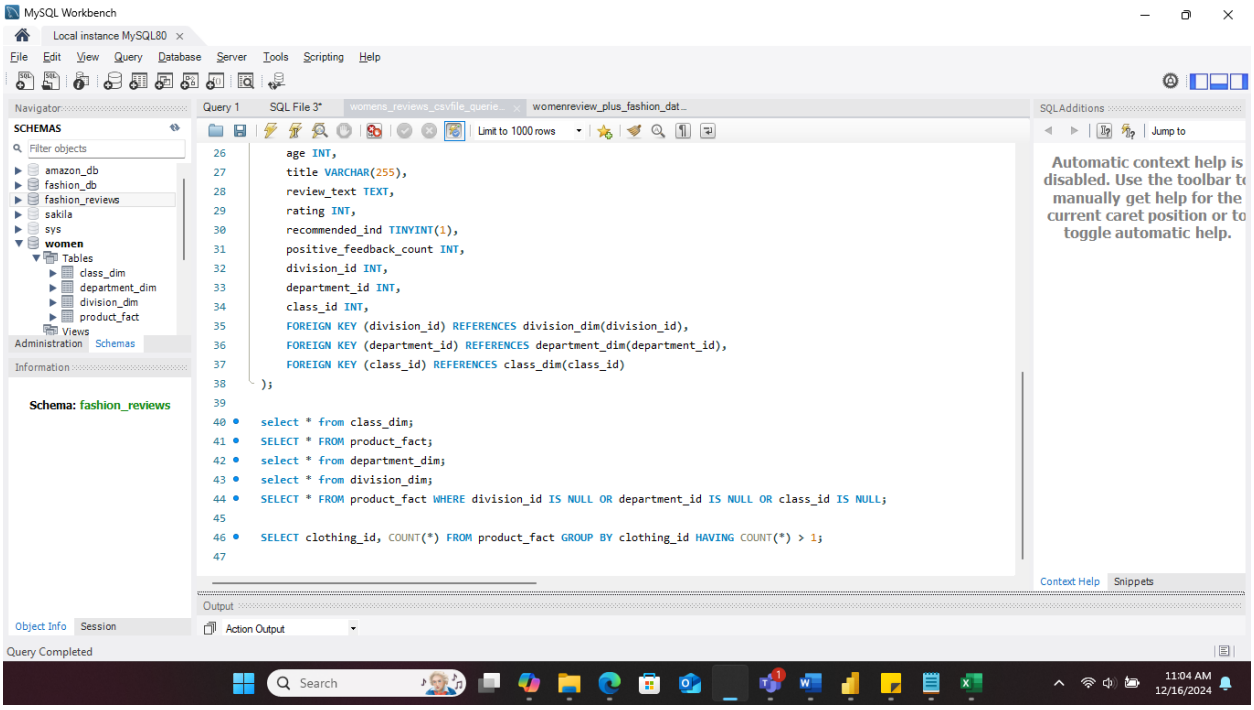
- **Normalization:**
  The data from the CSV file was divided into smaller, manageable tables

to reduce redundancy.
For example:

- division_name, department_name, and class_name were stored as unique rows in the respective dimension tables.
- The fact table used foreign keys (division_id, department_id, and class_id) to link to these tables.

- **Referential Integrity:**
  Foreign key constraints ensured valid relationships between the fact and dimension tables. This avoided data inconsistencies and errors during analysis.

- **Schema Verification**
- After creating the database and its tables, queries were run to verify the schema's integrity and to confirm that data relationships were correctly established.
- Example Queries:



SHOW COLUMNS FROM product_fact;

SELECT * FROM product_fact WHERE division_id IS NULL OR department_id IS NULL OR class_id IS NULL;

The schema ensures a star schema structure, enabling efficient analysis and integration with visualization tools like Power BI.

5. Data Visualization

This section details the choice of the visualization tool used in the project and the process of designing and developing insightful dashboards that align with the defined KPIs.

5.1 Choice of Data Visualization Tool

Selected Tool: Microsoft Power BI

For the visualization phase of the project, Microsoft Power BI was selected due to its intuitive interface, robust data modeling capabilities, and ability to create insightful dashboards. Power BI seamlessly integrates with SQL databases through the ODBC (Open Database Connectivity) driver, making it the ideal choice for this project.

Microsoft Power BI was chosen as the data visualization tool for the following reasons:

1. **Ease of Use:**
    - Power BI provides an intuitive drag-and-drop interface that simplifies the creation of complex visualizations.
2. **Interactivity:**
    - It allows users to build interactive reports and dashboards with slicers, filters, and drill-down functionalities.
3. **Data Connectivity:**
    - Power BI seamlessly connects to MySQL, enabling the integration of the structured data stored in the database.
4. **Advanced Customization:**
    - Offers a wide range of customization options for formatting, data labeling, and conditional formatting, ensuring that visuals are tailored to the business context.

5. **Support for KPIs:**
    - Power BI supports the creation of calculated measures, which were essential for aligning visualizations with project KPIs.
6. **Scalability:**
    - Power BI is scalable for both small projects and enterprise-level reporting, making it ideal for this use case.

5.2 Report Design / Dashboard Development

The dashboard was designed to provide actionable insights based on the KPIs defined earlier, ensuring that stakeholders could quickly identify trends and areas for improvement.

KPIs Implemented in Visualizations:

1. **Average Rating by Division, Department, and Class.**
2. **Average Rating by Age Group.**
3. **Sum of Positive Feedback Count by Department.**
4. **Sum of Rating by Department.**
5. **Sum of Rating by Division.**
6. **Sum of Positive Feedback Count by Title.**
7. **Sum of Rating by Class.**
8. **Sum of Positive Feedback Count and Rating by Age Group.**

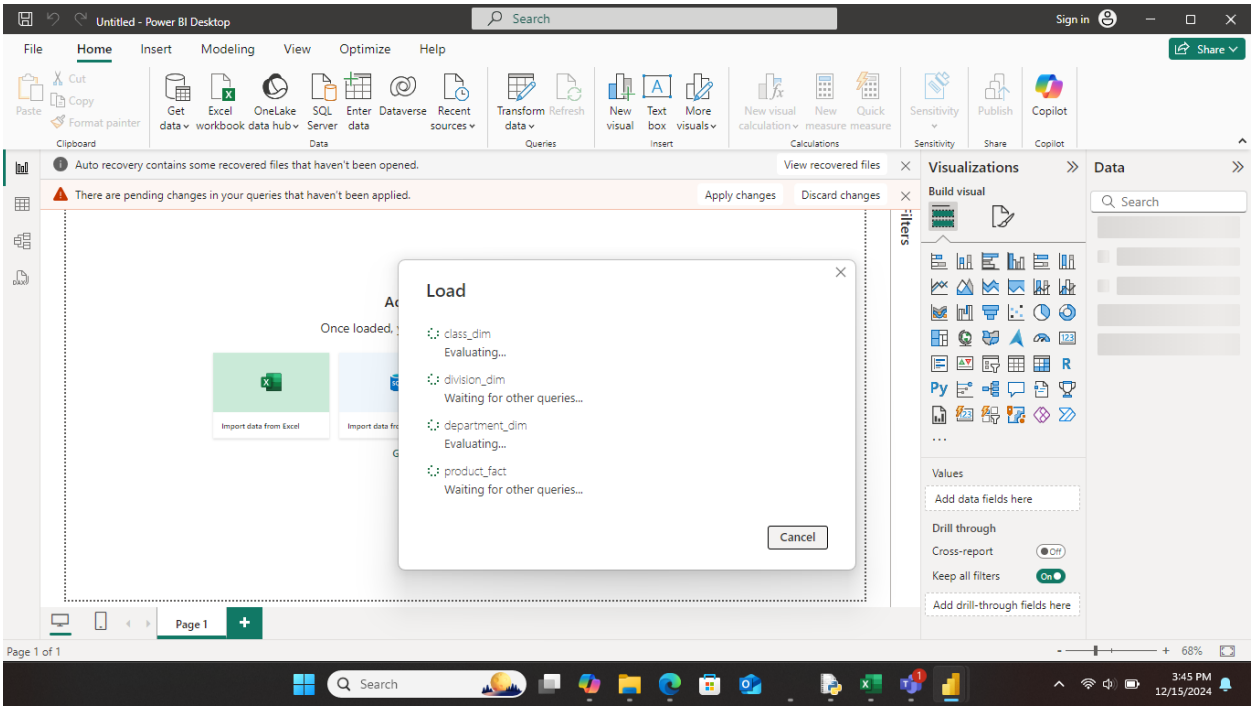
Dashboard Development

- **Connecting MySQL to Power BI Using ODBC:**

  After loading the transformed data into the MySQL database, we utilized the **ODBC driver** to establish a connection between the MySQL database and Power BI. This allowed for real-time access to the data stored in MySQL tables.

- Steps involved:
    - Configuring the **ODBC Data Source Administrator** to set up a connection to the MySQL database.

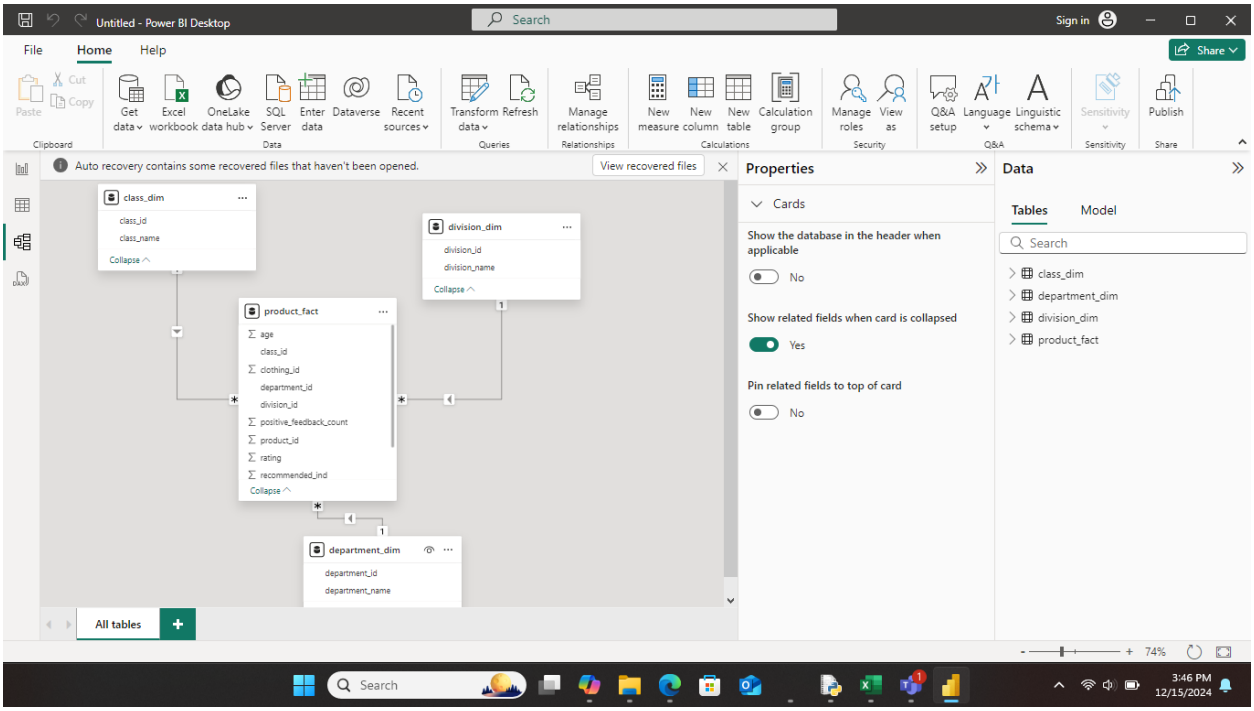
- Importing the tables (division_dim, department_dim, class_dim, and product_fact) into Power BI for further analysis and visualization.



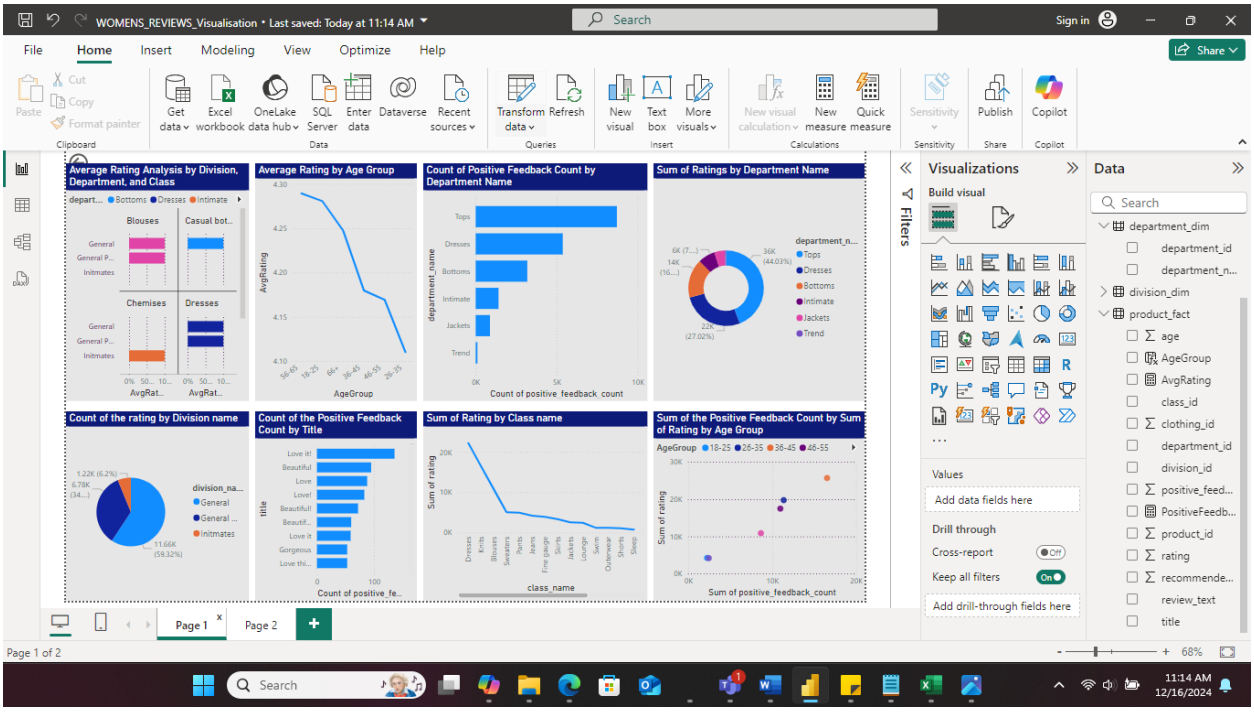
- **Data Modeling in Power BI:**

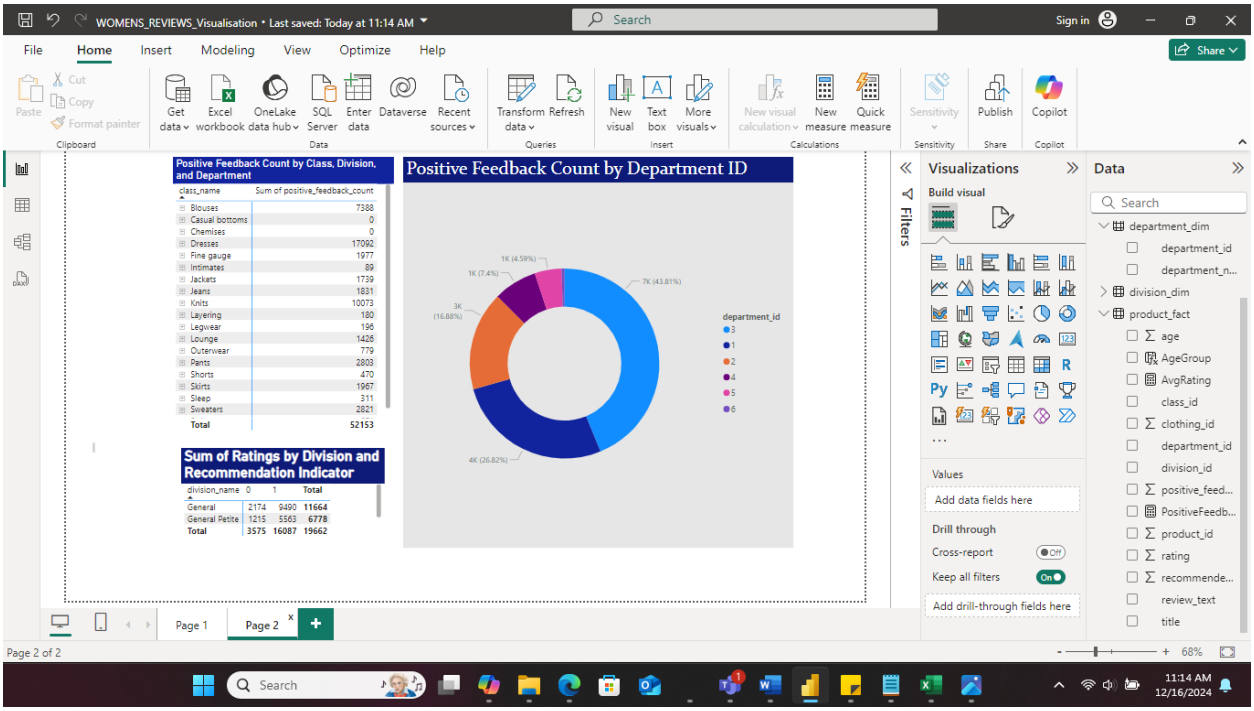
  Once the data was imported, relationships between the dimension tables and the fact table were defined in Power BI's **Model View**, replicating the star schema structure designed in MySQL.

- Key relationships:
  - product_fact.division_id → division_dim.division_id
  - product_fact.department_id → department_dim.department_id
  - product_fact.class_id → class_dim.class_id

- 

Main Page Visualizations





1. **Bar Chart: Average Rating Analysis by Division, Department, and Class**
    - **Objective:** Identify departments, division, class receiving the most positive feedback.
    - **Implementation:**
        - Used a **Stacked Bar Chart**.
        - X-Axis: AvgRating - A measure that I created
        - Y-Axis: division_name
        - Legend: department_name
2. **Line Chart: Average Rating by Age Group**
    - **Objective:** Analyze average ratings across different age groups.
    - **Implementation:**
        - Used a **Used a Line Chart**.
        - X-Axis: AgeGroup – A measure created by me in PowerBi
        - Y-Axis: Average Rating (Measure created in PowerBi).
3. **Bar Chart: Count of Positive Feedback Count by Department Name**

- **Objective:** Identify departments receiving the most positive feedback.
- **Implementation:**
  - Created a **Bar Chart**.
  - X-Axis: Positive Feedback Count - Count function
  - Y-Axis: department_name

4. **Pivot Table: Sum of Ratings by Department Name**
   - **Objective:** Visualize the proportion of ratings across departments.
   - **Implementation:**
     - Created a **Donut chart.**
     - Legend: department_name.
     - Values: Sum of Ratings.
5. **Bar Chart: Count of the rating by Division name**
   - **Objective:** Compare ratings across different divisions..
   - **Implementation:**
     - Created a **Pie Chart.**
     - Legend: Division Name.
     - Values: Sum of Rating.
6. **Bar Chart: Count of the Positive Feedback Count by Title**
   - **Objective:** Identify titles receiving the most positive feedback.
   - **Implementation:**
     - Created a **Bar Chart.**
     - X-axis: Title
     - Y-axis: Count of Positive Feedback Count.
7. **Line Chart: Sum of Rating by Class name**
   - **Objective:** Identify titles receiving the most positive feedback.
   - **Implementation:**

        - Created a **LineChart.**
        - X-axis: Title
        - Y-axis: Count of Positive Feedback Count.

8. **Scatter Plot: Sum of the Positive Feedback Count by Sum of Rating by Age Group**
    - **Objective:** Correlate positive feedback count and ratings across age groups.
    - **Implementation:**
        - Created a **Scatter plot.**
        - X-axis: Sum of positive_feedback_count
        - Y-axis: Sum of Rating.
        - Legend: AgeGroup.
9. **Table: Positive Feedback Count by Class, Division, and Department**
    - **Objective:** Provide detailed breakdowns for specific classes, divisions, and departments.
    - **Implementation:**
        - Created a **Table.**
        - Rows: class_name,division_name,department_name
        - Values: Sum of Positive feedback count.

10. **Pivot Table: Sum of Ratings by Division and Recommendation Indicator**
    - **Objective:** Analyze how division-wise ratings correlate with customer recommendations.
    - **Implementation:**
        - Created a **Pivot Table.**
        - Rows: division_name
        - Columns: Recommended_ind

        - Values: Sum of Ratings.

11. **Donut Chart: Positive Feedback Count by Department ID**
    - **Objective:** Visualize the proportion of positive feedback across departments.
    - **Implementation:**
        - Legend: department_id.
        - Values: Count of Positive feedback count.

Additional Features

- **Slicers:**
    - **Age Group Slicer:** Allows users to filter visuals by specific age groups.
    - **Division/Department/Class Slicers:** Enable filtering across multiple organizational levels.
    - **Rating Range Slicer:** Helps users focus on specific rating bands (e.g., low, medium, or high ratings).
- **Drill-Down and Hierarchy:**
    - Hierarchical relationships (Division > Department > Class) were built into visualizations to allow drill-down for granular insights.
- **Custom Tooltips:**
    - Tooltips were configured to show additional details, such as the percentage contribution of positive feedback or ratings when hovering over visuals.

Insights Derived

1. **Division-Level Analysis:**
    - The bar chart highlighted which divisions had the highest average ratings and the most positive feedback, helping to identify areas of success.
2. **Age Group Trends:**

- Visuals showed how different age groups rated products, aiding in targeted marketing and product development.

3. **Product-Level Analysis:**
   - Titles receiving high feedback scores indicated products that were well-received, which can inform inventory and promotional strategies.
4. **Recommendation Patterns:**
   - The pivot table revealed strong correlations between division performance and customer recommendations, enabling prioritization of top-performing divisions.

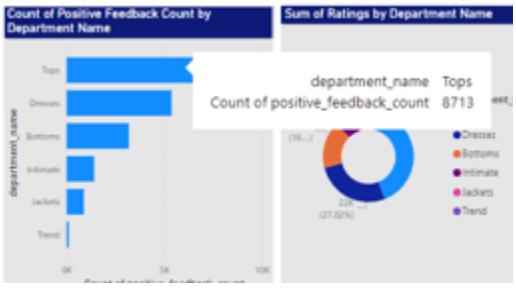
6. Conclusion and Summary

This section provides a holistic view of the project's outcomes, highlighting the key findings, actionable insights derived from the visualizations, and reflections on how the project addressed the business case objectives.

6.1 Key Findings

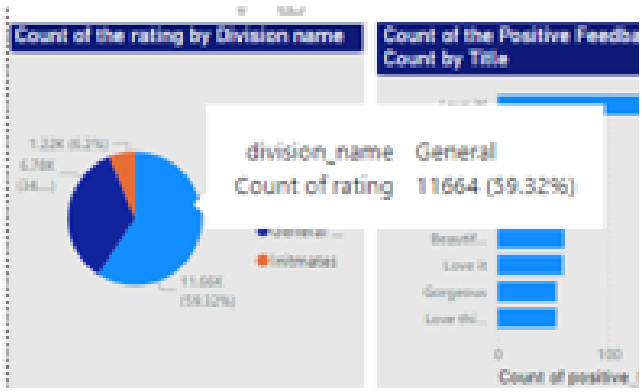
The analysis and data visualizations generated the following key findings:

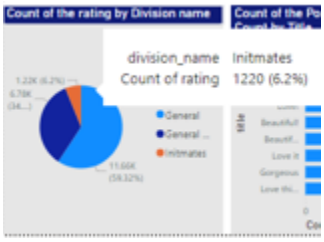
1. **Department Performance:**
   - Departments receiving the highest positive feedback were identified. This demonstrates customer satisfaction and product performance at a departmental level.
   - For example, **Tops** had the highest positive feedback count of **8713**, indicating its strong alignment with customer preferences.



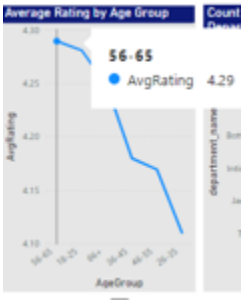
2. **Division-Wise Ratings:**

- The division-level breakdown of average ratings revealed that **General division** consistently performed well across products, with an average rating of **11664**.
- Lower-performing divisions, such as **Intimates**, had room for improvement, as indicated by an average rating below the project-defined threshold of **1120**.





3. **Age Group Analysis:**
   - The age group analysis showed that customers in the **56- 65 age group** category had the highest average ratings for products, highlighting a target demographic for marketing efforts.



4. **Positive Feedback Distribution:**
   - Positive feedback was heavily concentrated in certain titles and classes. For instance, **[Love it!]** received the most positive feedback, accounting for **136 counts** of the total.

5. **Recommendation Trends:**
    - The pivot table revealed that **Division - General** had the highest percentage of "Recommended" products, correlating strongly with higher overall ratings.

6. **Class-Level Trends:**
    - Classes like **Class - Dresses** showed a consistent trend in receiving both high ratings and positive feedback, indicating customer preference for products in these categories.

6.2 Insights for Decision-Making

The findings from the analysis provided actionable insights to guide decision-making:

1. **Optimizing Product Offerings:**
    - Products in classes and titles with high feedback and ratings should be prioritized in future inventory and marketing plans.
    - Low-performing products identified through class-level and title-level breakdowns can either be improved or discontinued.
2. **Targeted Marketing Strategies:**

- The age group analysis helps refine marketing strategies, focusing on the most responsive demographic.
- Departments and divisions with high positive feedback can be leveraged in promotional campaigns to maximize returns.

3. **Resource Allocation:**
    - Resource allocation can be improved by prioritizing high-performing divisions and classes for future investments in product development and promotions.
4. **Improving Customer Satisfaction:**
    - Lower-rated divisions and products highlighted opportunities to improve customer satisfaction by addressing specific pain points in quality or design.
5. **Enhancing Recommendations:**
    - By identifying patterns in products that are frequently recommended, stakeholders can enhance recommendation systems and tailor them to customer needs.

6.3 Reflection on Business Case Addressed

The project successfully addressed the initial business case, which was to utilize customer feedback data for improving product offerings and aligning them with customer preferences. Here's how:

1. **Data-Driven Insights:**
    - The dashboards provided granular insights into how products performed across divisions, departments, and classes, aligning with the objective to identify strong and weak areas in the product portfolio.
2. **Enhanced Customer Understanding:**
    - The age group and feedback analyses deepened the understanding of customer demographics and preferences, supporting the business goal of personalized marketing.
3. **Streamlined Decision-Making:**

- With clear visualizations, stakeholders were equipped to make faster, more informed decisions on product lifecycle management, inventory optimization, and strategic planning.

4. **Scalability for Future Use Cases:**
    - The modular approach to visualization and data processing ensures scalability. Additional datasets or KPIs can be incorporated into the system with minimal effort.

7. Documentation

This section documents the technical details of the project, including the data modeling diagram and the specific transformation steps undertaken during the ETL process. These details are essential for understanding the project's structure and processes, ensuring reproducibility and scalability.

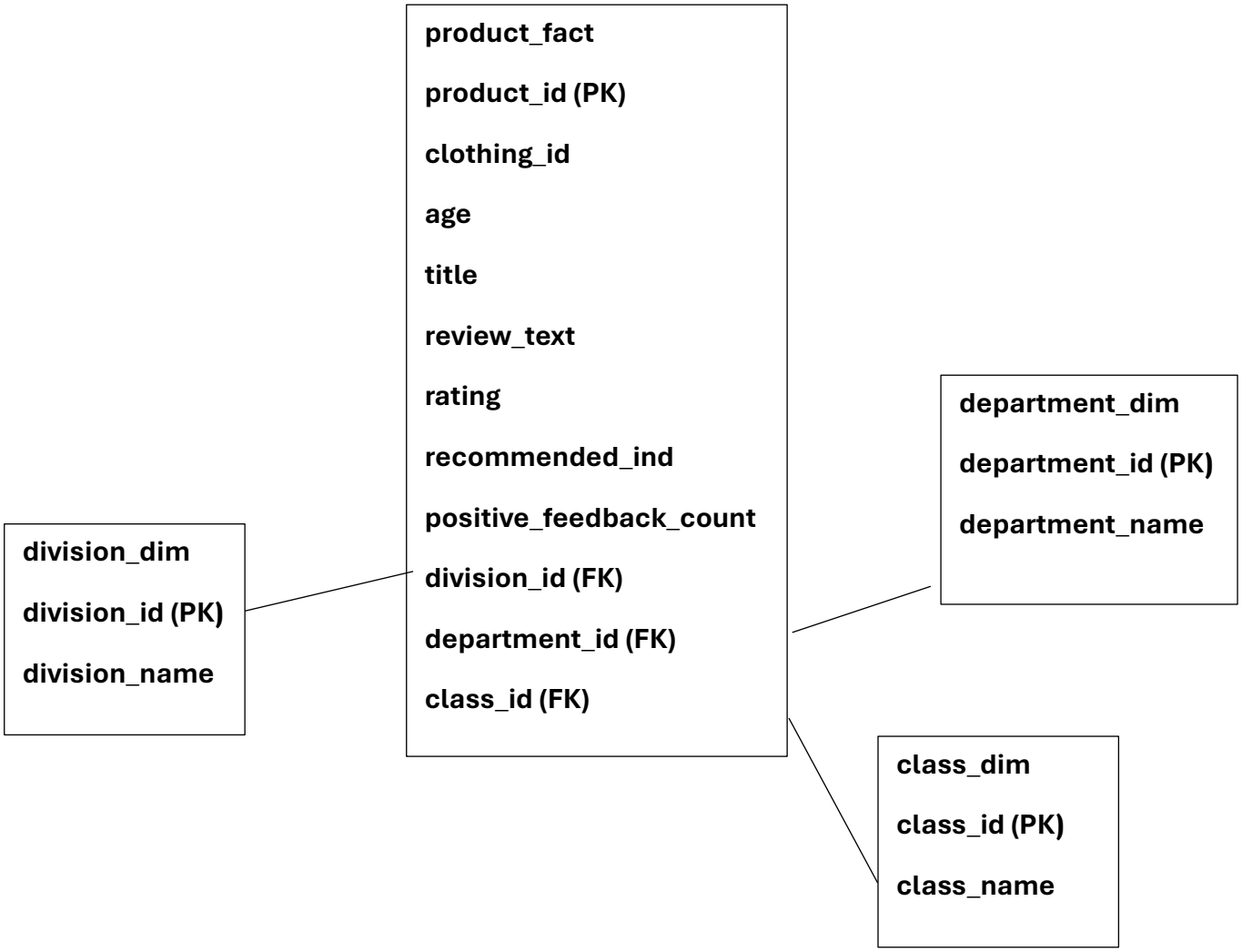
7.1 Data Modeling Diagram

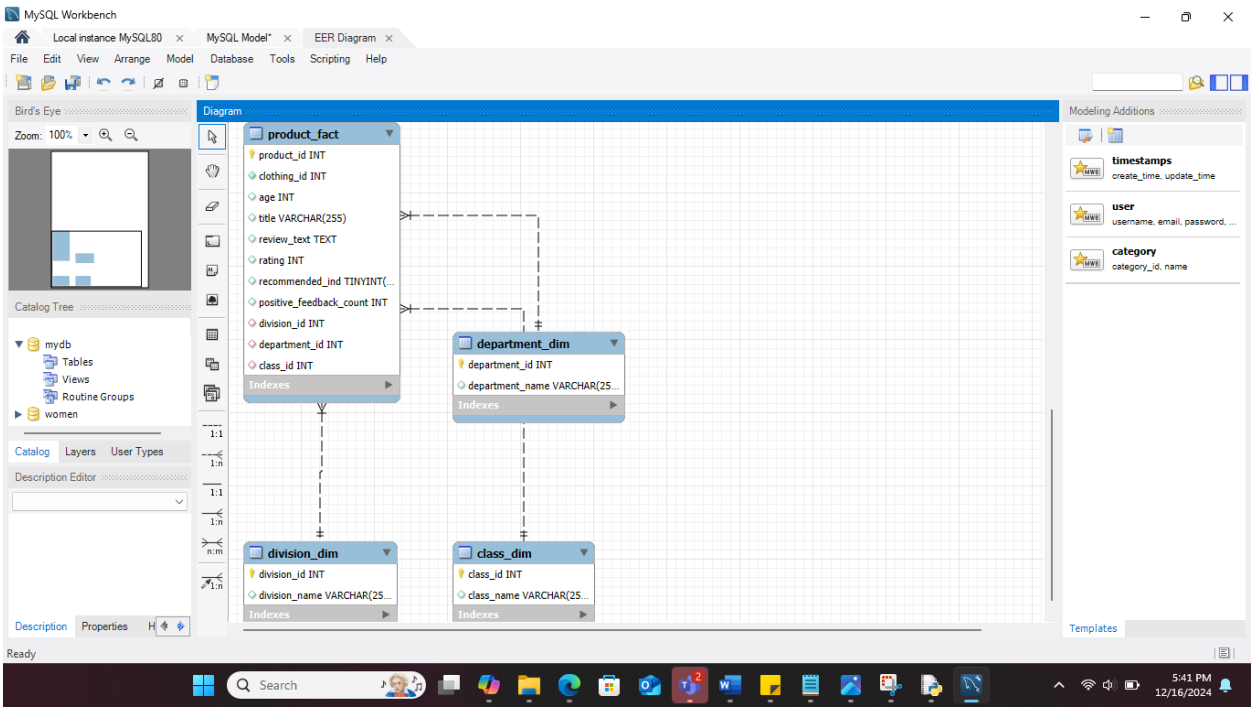
The **data modeling diagram** represents the relationships between the fact table and dimension tables, as well as the overall schema structure designed to support efficient querying and analysis.

1. **Database Structure:**
    - The schema followed a star schema design, which is optimal for analytical workloads. The central fact table (product_fact) is surrounded by dimension tables (division_dim, department_dim, and class_dim).
    - Key attributes included:
        - **Fact Table (product_fact)**:
            - Stores product-level details, including review ratings, customer feedback, and related dimensions.
            - Primary Key: product_id.
            - Foreign Keys: division_id, department_id, class_id.
        - **Dimension Tables**:

- division_dim: Captures division-level information with a unique identifier (division_id) and division name.
- department_dim: Contains department-level details, including department name and identifier (department_id).
- class_dim: Represents product classification with fields for class name and unique identifier (class_id).

STAR SCHEMA:





2. **Entity Relationships:**
    - **One-to-Many Relationships**:
        - A single division can have multiple departments.
        - Each department can have multiple product classes.
    - The foreign keys in the product_fact table ensure the integrity of data across dimensions.
3. **Visualization of the Data Model:**
    - A diagram was created to visually depict the database structure, showing:
        - **Dimension Tables**: division_dim, department_dim, class_dim.
        - **Fact Table**: product_fact.
        - **Relationships**: Arrows connecting the dimension tables to the fact table, indicating the direction of relationships.

7.2 Transformation Details

The transformation step was critical to ensuring data integrity, consistency, and readiness for analysis. Below are the key transformation activities:

1. **Splitting the CSV Data:**
    - The original dataset (CSV file) was normalized into separate tables to reduce redundancy and improve query performance.
    - Data was split into:
        - division_dim for division-level data.
        - department_dim for department-level data.
        - class_dim for class-level data.
        - product_fact for transactional product and feedback details.
2. **Data Cleaning:**
    - Null and duplicate values were addressed during the transformation:
        - Rows with missing division_id, department_id, or class_id were flagged and reviewed.
        - Duplicate clothing_id entries in the product_fact table were identified and removed using the query:

```
SELECT clothing_id, COUNT(*)
FROM product_fact
GROUP BY clothing_id
HAVING COUNT(*) > 1;
```

- Inconsistent data entries, such as misspelled class names or duplicate titles, were corrected.
- **Key Generation and Assignment:**
- Surrogate keys (division_id, department_id, class_id) were auto-generated for dimension tables.

- Foreign keys in the product_fact table were populated by matching records with their corresponding dimension entries.
- **Normalization Process:**
- Division, department, and class names were extracted from the original dataset and grouped into distinct dimension tables:
    - division_dim: Created to store unique division names.
    - department_dim: Created to store unique department names.
    - class_dim: Created to store unique class names.

7.2 **Data Transformation Scripts:**

- SQL queries were written to populate the normalized tables

```
INSERT INTO division_dim (division_name)
SELECT DISTINCT division_name
FROM division_dim;

INSERT INTO department_dim (department_name)
SELECT DISTINCT department_name
FROM department_dim;

INSERT INTO class_dim (class_name)
SELECT DISTINCT class_name
FROM class_dim;
```

-
- Relationships between dimensions and facts were maintained by linking their primary and foreign keys.

6. **Loading Data into the Fact Table:**

- Once the dimension tables were populated, the product_fact table was loaded with complete data, ensuring that all foreign key references pointed to valid dimension records.

7. **Validation:**

- Queries were executed to validate data integrity:
    - Ensuring no null values existed in foreign key fields:

```
SELECT *
FROM product_fact
WHERE division_id IS NULL OR department_id IS NULL OR class_id IS NULL;
```

Verifying accurate counts of records in each table to match the input dataset:

```
SELECT COUNT(*) FROM division_dim;
SELECT COUNT(*) FROM department_dim;
SELECT COUNT(*) FROM class_dim;
SELECT COUNT(*) FROM product_fact;
```

8. Project Implementation Challenges and Solutions

Throughout the implementation of the project, various challenges were encountered. These challenges required strategic approaches and innovative solutions to ensure the project's success.

Challenges:

1. **Data Quality Issues:**
    - The original dataset contained null values, duplicate entries, and inconsistent naming conventions.
    - Solution: Rigorous data cleaning and validation steps were implemented using SQL queries. Null values were identified and managed by referencing related tables. Duplicates were removed after identifying unique records using aggregation queries.

2. **Normalization of Data:**
    - Splitting the dataset into multiple normalized tables introduced complexity in managing relationships and ensuring no data was lost during transformation.
    - Solution: Careful design of primary and foreign key relationships ensured data integrity. SQL queries with robust validations were executed to confirm accurate data mapping.
3. **ETL Process Optimization:**
    - Extracting, transforming, and loading data in a structured and efficient manner posed challenges, especially with large datasets.
    - Solution: Python scripts were optimized with batch processing techniques, and the database was indexed to improve performance during data loading.
4. **Database Design Challenges:**
    - Ensuring scalability and efficient querying required thoughtful schema design.
    - Solution: A star schema was chosen for its analytical efficiency, balancing simplicity and performance.
5. **Visualization Constraints:**
    - Designing dashboards to accommodate a wide range of KPIs and ensuring clarity of insights was challenging.
    - Solution: Iterative dashboard development with feedback loops from stakeholders ensured relevant and impactful visualizations.
6. **Configuring ODBC Driver:**

    Downloaded and installed the 64-bit ODBC driver for MySQL to enable compatibility with Power BI.

    Configured a DSN (Data Source Name) in the ODBC Data Source Administrator to ensure successful connectivity.

9. Future Enhancements and Recommendations

While the project achieved its objectives, there is potential for further enhancements to improve usability, scalability, and analytical depth.

Future Enhancements:

1. **Integration of Additional Data Sources:**
    - Incorporate more datasets, such as customer demographics or sales trends, to provide richer insights.
2. **Automated ETL Pipeline:**
    - Implement an automated ETL process using tools like Apache Airflow or Alteryx to improve efficiency and reduce manual intervention.
3. **Real-Time Analytics:**
    - Upgrade the solution to support real-time data ingestion and reporting for time-sensitive decision-making.
4. **Advanced Analytical Techniques:**
    - Use machine learning models for predictive analytics, such as predicting product performance or customer sentiment trends.
5. **Enhanced Visualizations:**
    - Introduce interactive features like drill-down capabilities and dynamic filtering in dashboards for better user experience.
6. **Cloud Migration:**
    - Host the database and visualization tools on cloud platforms (e.g., AWS or Azure) for improved scalability and collaboration.

Recommendations:

- Maintain consistent data quality checks during new data ingestion.
- Regularly review and update dashboards to reflect evolving business needs.
- Train users in dashboard usage and interpretation to maximize value.

10. References

This section lists the resources and tools used during the project, including datasets, technologies, and any documentation or tutorials consulted.

1. **Datasets:**
    - Kaggle: "Women's Clothing Reviews Dataset"
2. **Technologies:**
    - MySQL for database management. https://dev.mysql.com/doc/
    - Python for ETL processes. 3.13.1 Documentation

      https://pandas.pydata.org/pandas-docs/stable/
    - Power BI for visualization. Power BI documentation - Power BI | Microsoft Learn
3. **Documentation and Tutorials:**
    - Official MySQL Documentation. https://dev.mysql.com/doc/
    - Python ETL Tutorials and Libraries Documentation. Towards Data Science

      pandas documentation — pandas 2.2.3 documentation
    - Power BI Official Guide. Power BI documentation - Power BI | Microsoft Learn
4. **Additional References:**
    - Articles and forums on database normalization and data modeling best practices.

Data Database Normalization:base

- Database Normalization Explained
- Normalization in DBMS

Data Modeling Best PracticesModeling Best

- Data Modeling Best Practices
- Data Modeling Techniques

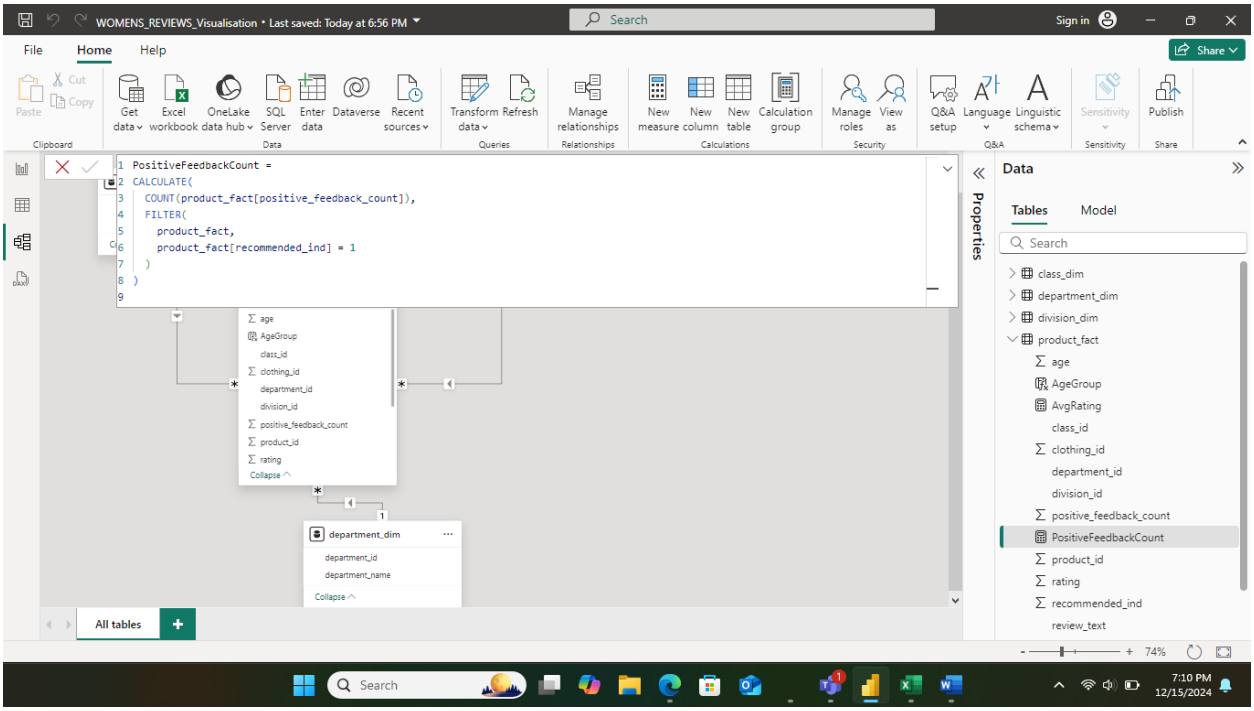
11. Appendices

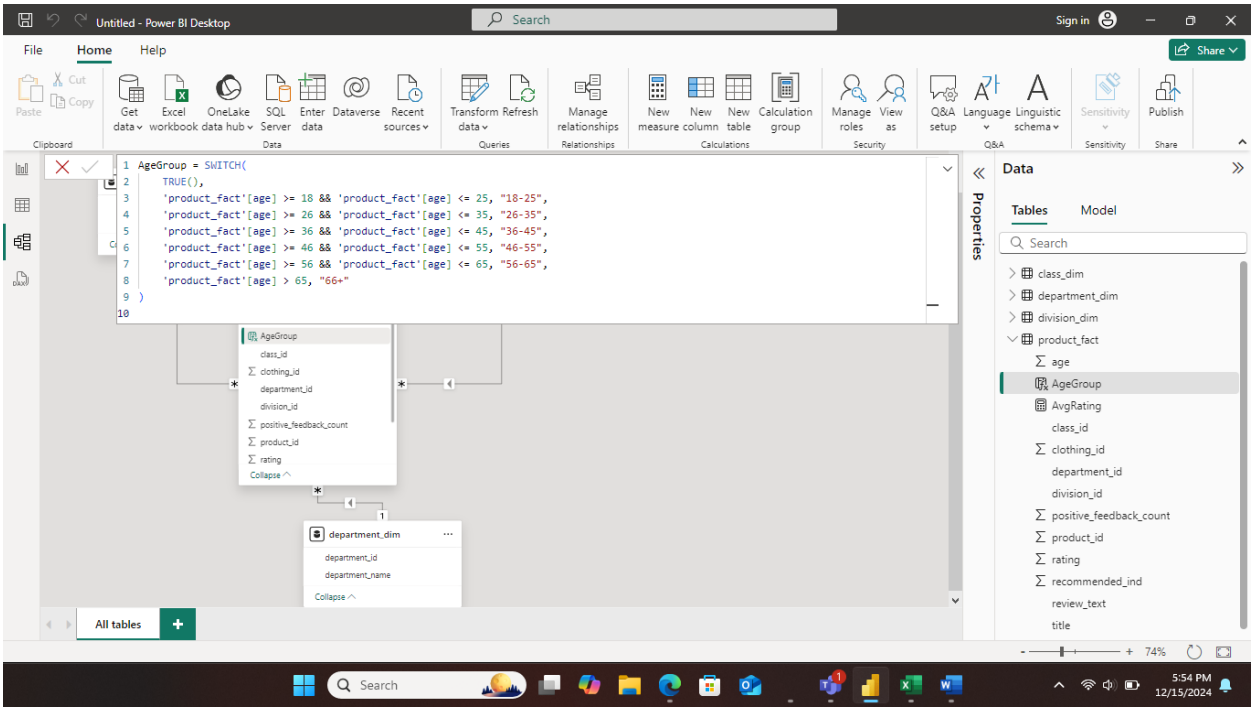
The appendices provide supplementary materials that support the project, including additional charts, code snippets, and a glossary of terms.

11.1 Additional Charts and Graphs

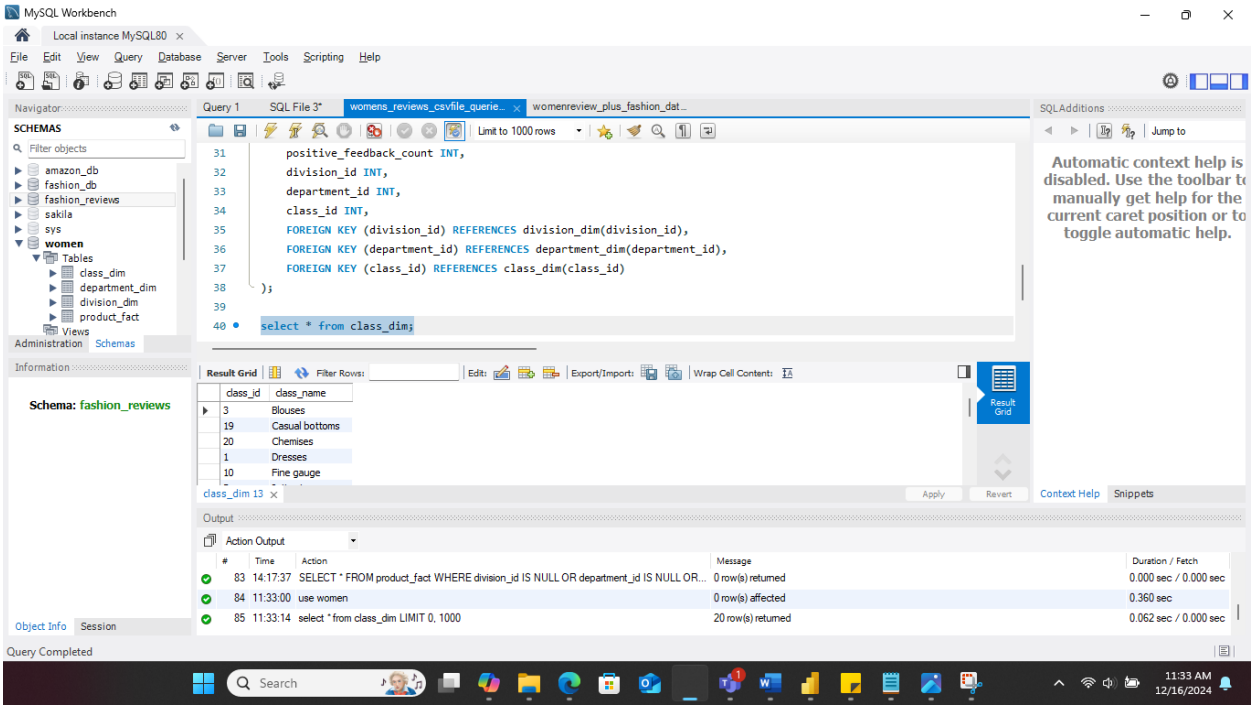
- **Additional Visualizations:**

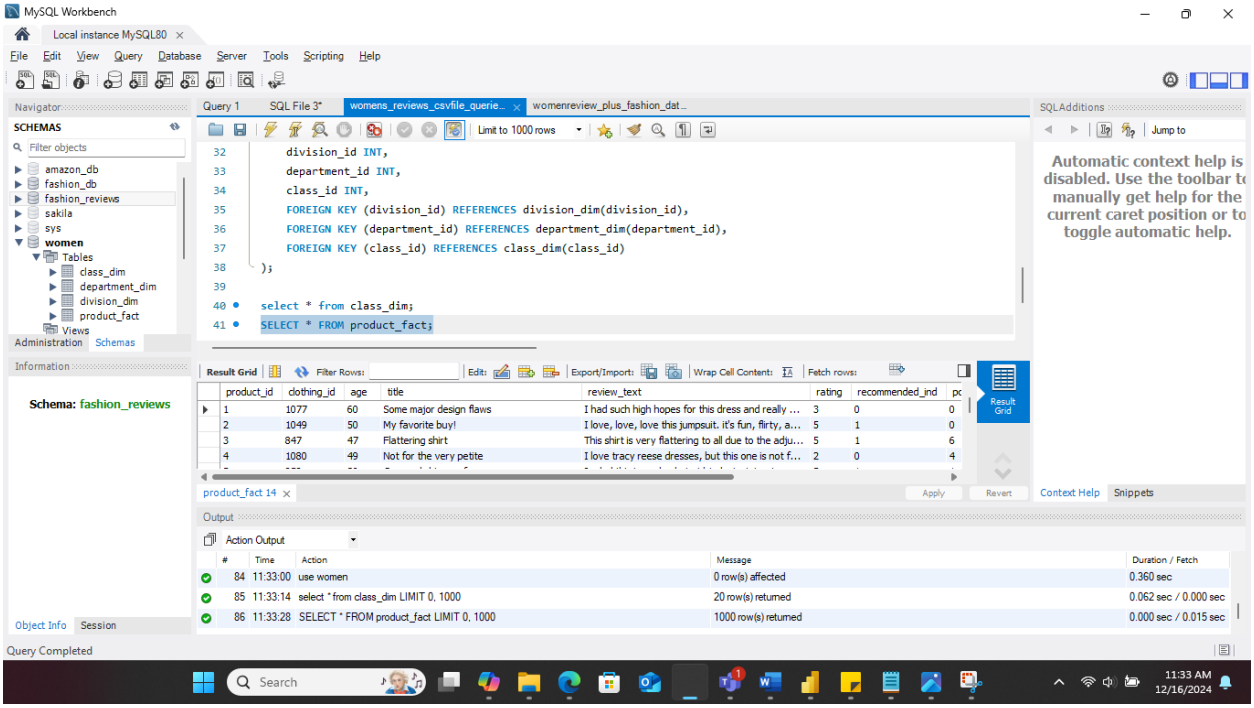
Measure creations:

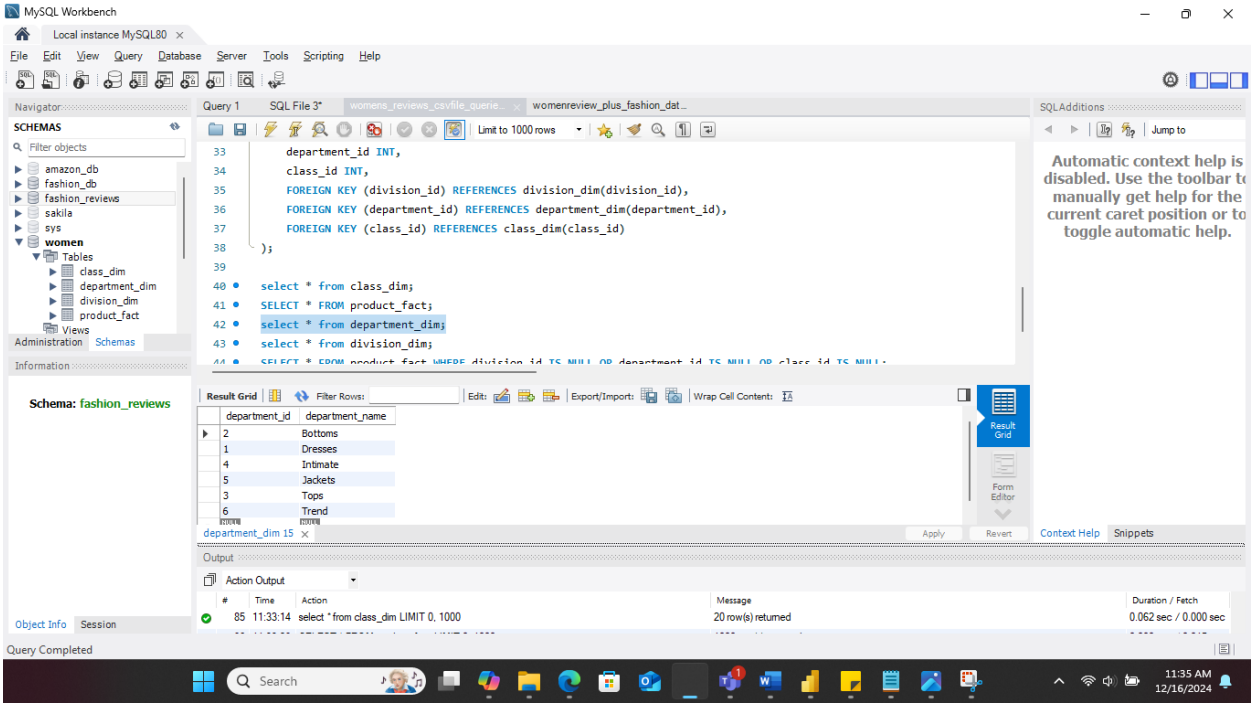


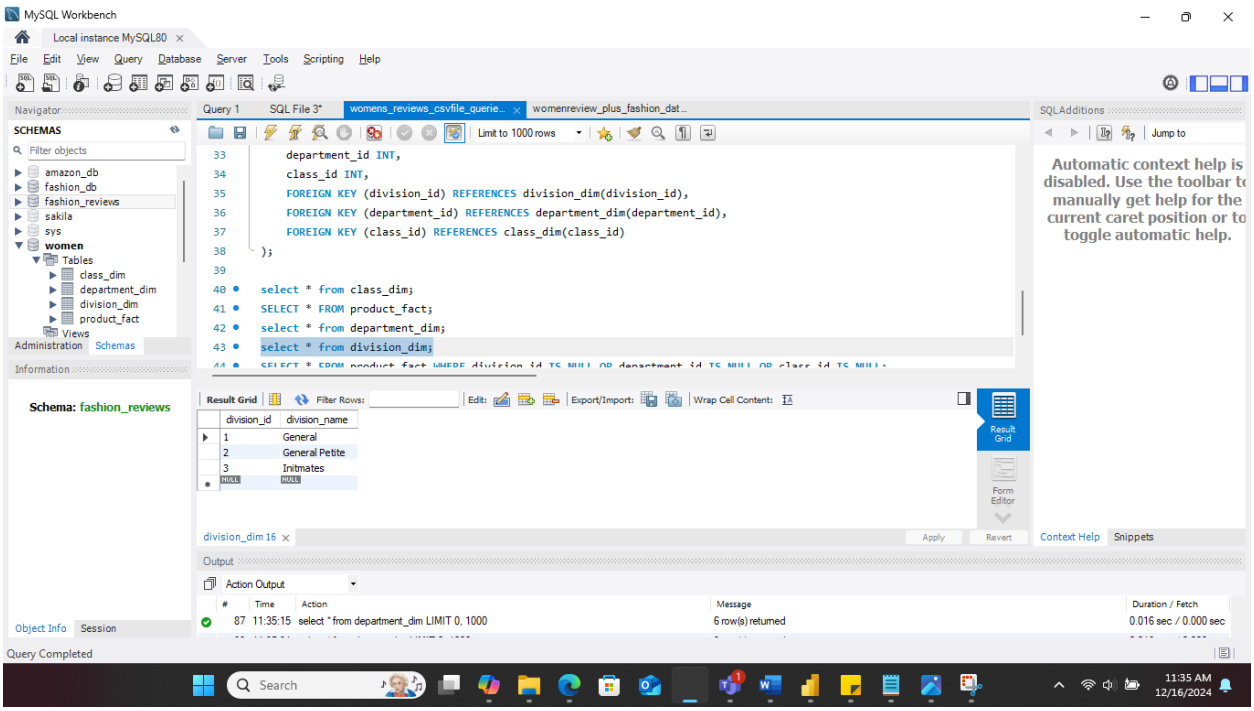


11.2 Code Snippets (if applicable)

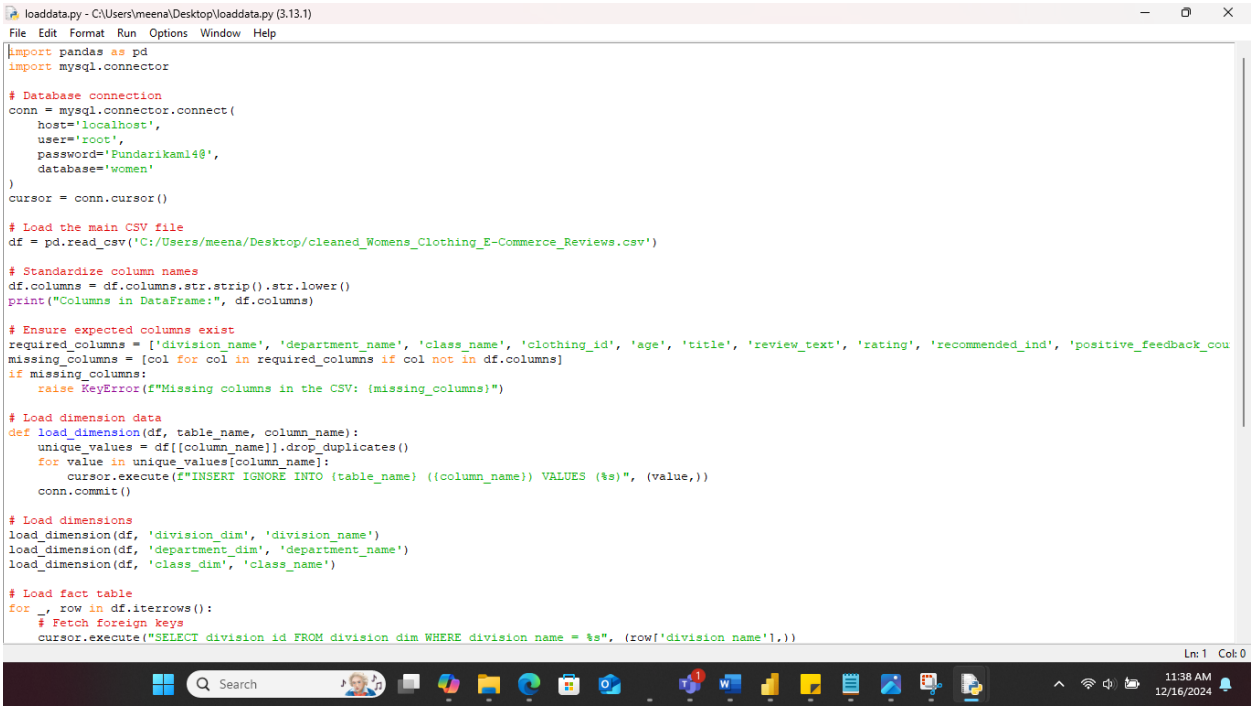


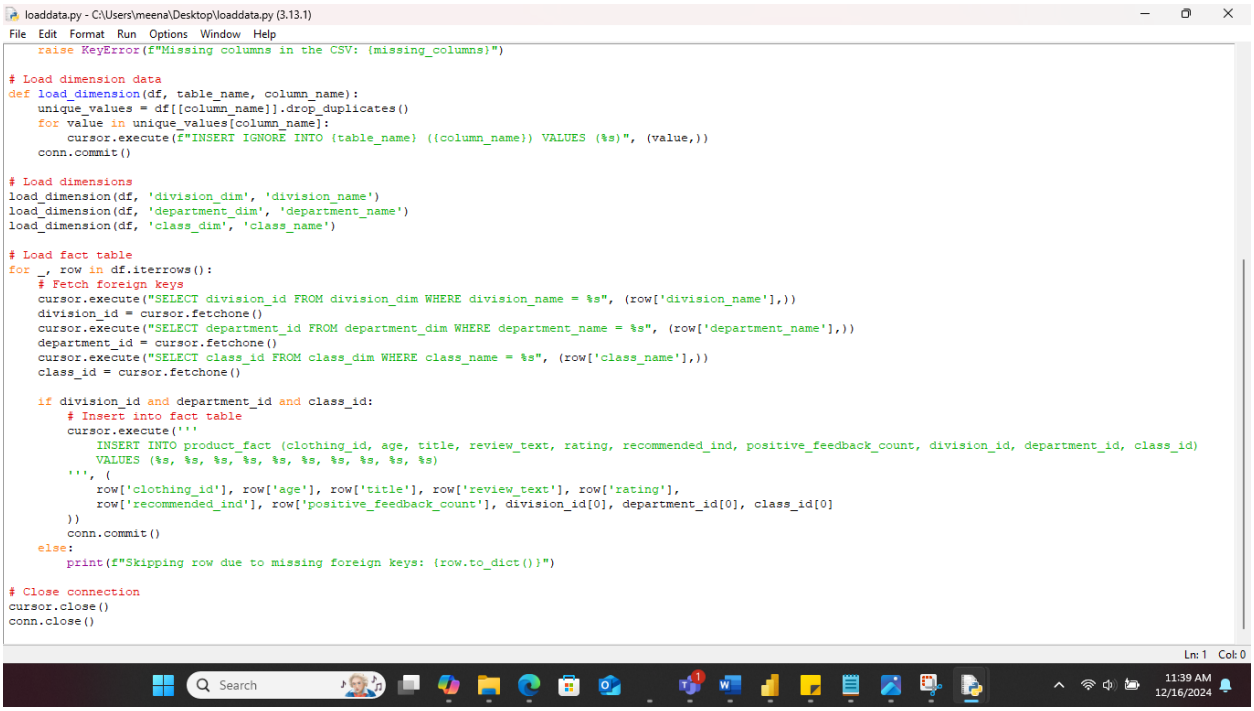

MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
amazon_db
fashion_db
fashion_reviews
sakila
sys
women
Tables
class_dim
department_dim
division_dim
product_fact
Views
Administration Schemas
Information
Schema: fashion_reviews
Object Info Session
Query 1 SQL File 3* womens_reviews_csvfile_querie... womenreview_plus_fashion_dat...
Limit to 1000 rows
32 division_id INT,
33 department_id INT,
34 class_id INT,
35 FOREIGN KEY (division_id) REFERENCES division_dim(division_id),
36 FOREIGN KEY (department_id) REFERENCES department_dim(department_id),
37 FOREIGN KEY (class_id) REFERENCES class_dim(class_id)
38);
39
40 select * from class_dim;
41 SELECT * FROM product_fact;
Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content: Fetch rows:
product_id clothing_id age title review_text rating recommended_ind pc
1 1077 60 Some major design flaws I had such high hopes for this dress and really ... 3 0 0
2 1049 50 My favorite buy! I love, love, love this jumpsuit. it's fun, flirty, a... 5 1 0
3 847 47 Flattering shirt This shirt is very flattering to all due to the adju... 5 1 6
4 1080 49 Not for the very petite I love tracy reese dresses, but this one is not f... 2 0 4
product_fact 14
Apply Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
84 11:33:00 use women 0 row(s) affected 0.360 sec
85 11:33:14 select * from class_dim LIMIT 0, 1000 20 row(s) returned 0.062 sec / 0.000 sec
86 11:33:28 SELECT * FROM product_fact LIMIT 0, 1000 1000 row(s) returned 0.000 sec / 0.015 sec
Query Completed
Search
11:33 AM
12/16/2024


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
amazon_db
fashion_db
fashion_reviews
sakila
sys
women
Tables
class_dim
department_dim
division_dim
product_fact
Views
Administration Schemas
Information
Schema: fashion_reviews
Object Info Session
Query 1 SQL File 3* womens_reviews_csvfile_querie... womenreview_plus_fashion_dat...
Limit to 1000 rows
33 department_id INT,
34 class_id INT,
35 FOREIGN KEY (division_id) REFERENCES division_dim(division_id),
36 FOREIGN KEY (department_id) REFERENCES department_dim(department_id),
37 FOREIGN KEY (class_id) REFERENCES class_dim(class_id)
38);
39
40 select * from class_dim;
41 SELECT * FROM product_fact;
42 select * from department_dim;
43 select * from division_dim;
Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:
department_id department_name
2 Bottoms
1 Dresses
4 Intimate
5 Jackets
3 Tops
6 Trend
department_dim 15
Apply Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
85 11:33:14 select * from class_dim LIMIT 0, 1000 20 row(s) returned 0.062 sec / 0.000 sec
Query Completed
Search
11:35 AM
12/16/2024

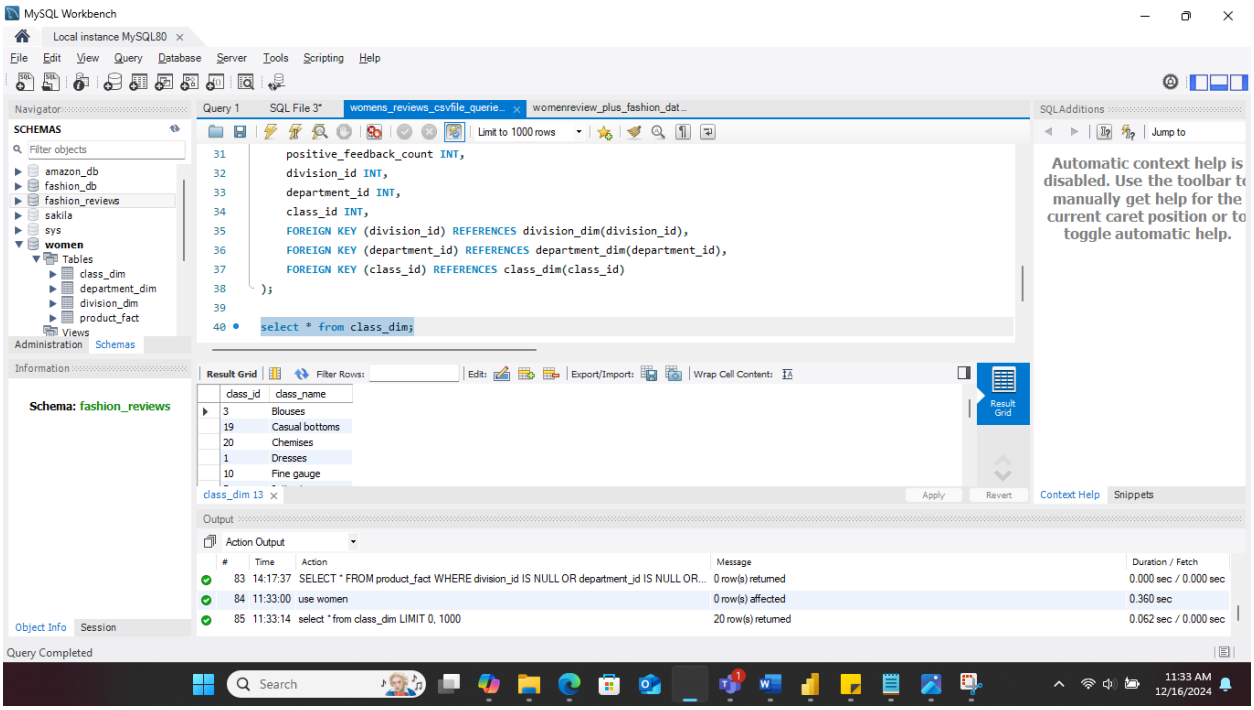


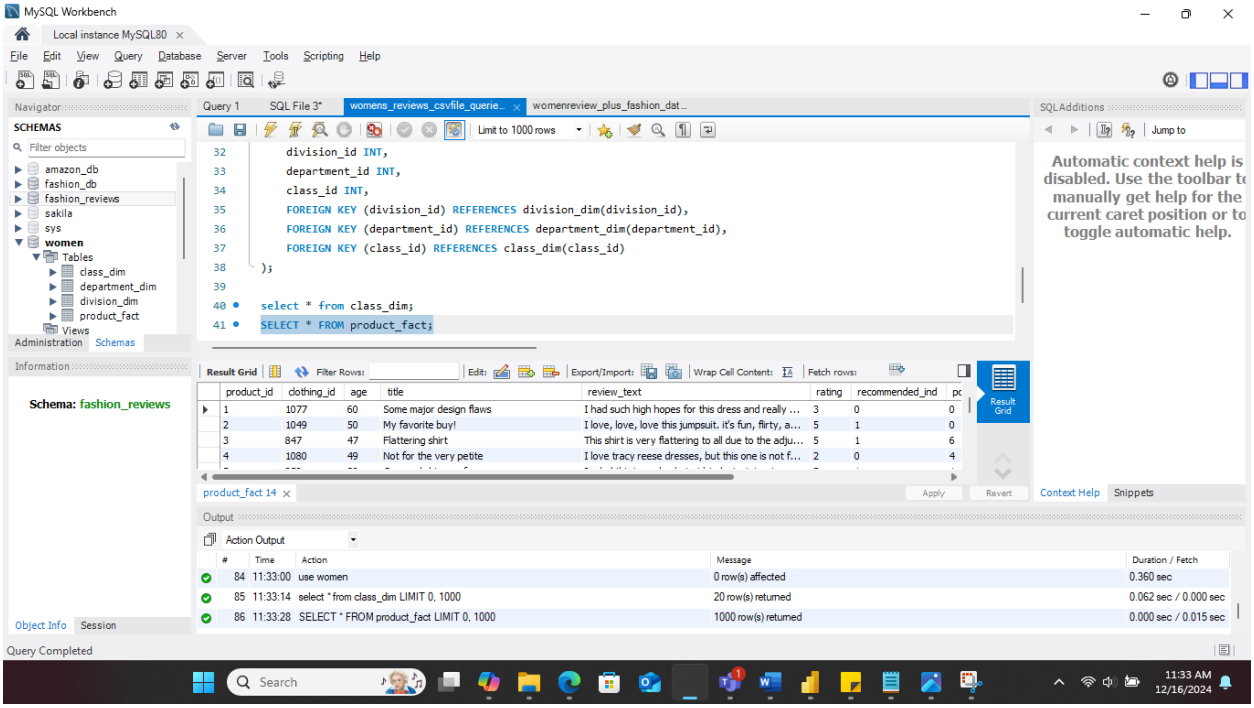
- Key Python scripts for ETL processes:

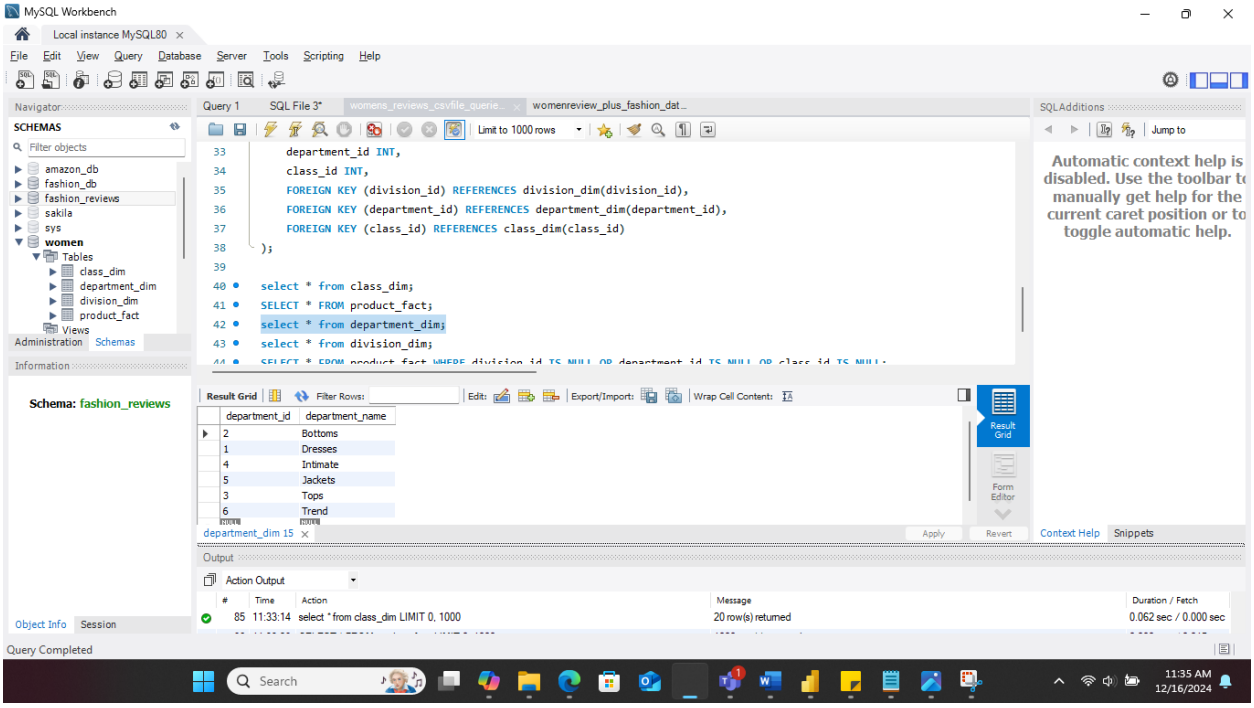


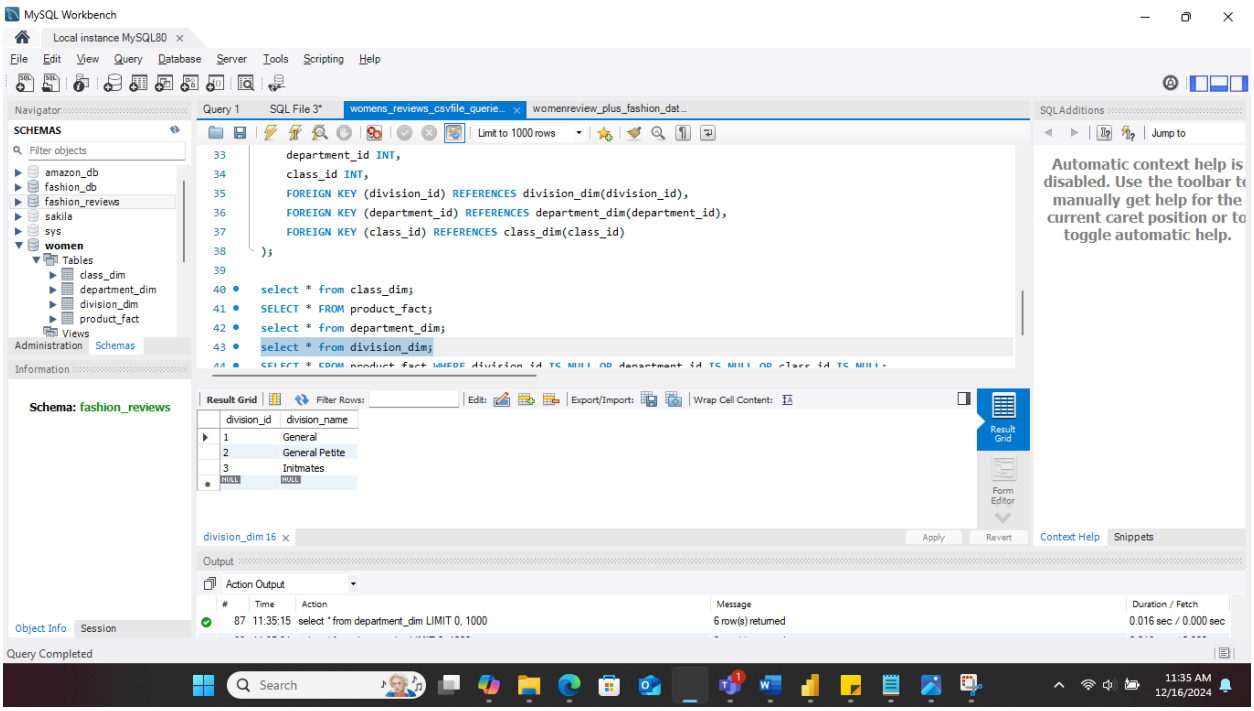

loaddata.py - C:\Users\meena\Desktop\loaddata.py (3.13.1)
File Edit Format Run Options Window Help
raise KeyError(f"Missing columns in the CSV: {missing_columns}")
# Load dimension data
def load_dimension(df, table_name, column_name):
unique_values = df[[column_name]].drop_duplicates()
for value in unique_values[column_name]:
cursor.execute(f"INSERT IGNORE INTO {table_name} ({column_name}) VALUES (%s)", (value,))
conn.commit()
# Load dimensions
load_dimension(df, 'division_dim', 'division_name')
load_dimension(df, 'department_dim', 'department_name')
load_dimension(df, 'class_dim', 'class_name')
# Load fact table
for _, row in df.iterrows():
# Fetch foreign keys
cursor.execute("SELECT division_id FROM division_dim WHERE division_name = %s", (row['division_name'],))
division_id = cursor.fetchone()
cursor.execute("SELECT department_id FROM department_dim WHERE department_name = %s", (row['department_name'],))
department_id = cursor.fetchone()
cursor.execute("SELECT class_id FROM class_dim WHERE class_name = %s", (row['class_name'],))
class_id = cursor.fetchone()
if division_id and department_id and class_id:
# Insert into fact table
cursor.execute('''
INSERT INTO product_fact (clothing_id, age, title, review_text, rating, recommended_ind, positive_feedback_count, division_id, department_id, class_id)
VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
''', (
row['clothing_id'], row['age'], row['title'], row['review_text'], row['rating'],
row['recommended_ind'], row['positive_feedback_count'], division_id[0], department_id[0], class_id[0]
))
conn.commit()
else:
print(f"Skipping row due to missing foreign keys: {row.to_dict()}")
# Close connection
cursor.close()
conn.close()
Ln: 1 Col: 0
Search
11:39 AM
12/16/2024

Loaded data:


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
amazon_db
fashion_db
fashion_reviews
sakila
sys
women
Tables
class_dim
department_dim
division_dim
product_fact
Views
Administration Schemas
Information
Schema: fashion_reviews
Object Info Session
Query 1 SQL File 3* womens_reviews_csvfile_querie... womenreview_plus_fashion_dat...
Limit to 1000 rows
positive_feedback_count INT,
division_id INT,
department_id INT,
class_id INT,
FOREIGN KEY (division_id) REFERENCES division_dim(division_id),
FOREIGN KEY (department_id) REFERENCES department_dim(department_id),
FOREIGN KEY (class_id) REFERENCES class_dim(class_id)
);
select * from class_dim;
Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:
class_id class_name
3 Blouses
19 Casual bottoms
20 Chemises
1 Dresses
10 Fine gauge
class_dim 13
Apply Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
83 14:17:37 SELECT * FROM product_fact WHERE division_id IS NULL OR department_id IS NULL OR... 0 row(s) returned 0.000 sec / 0.000 sec
84 11:33:00 use women 0 row(s) affected 0.360 sec
85 11:33:14 select * from class_dim LIMIT 0, 1000 20 row(s) returned 0.062 sec / 0.000 sec
Query Completed
Search
11:33 AM
12/16/2024


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
amazon_db
fashion_db
fashion_reviews
sakila
sys
women
Tables
class_dim
department_dim
division_dim
product_fact
Views
Administration Schemas
Information
Schema: fashion_reviews
Object Info Session
Query 1 SQL File 3* womens_reviews_csvfile_querie... womenreview_plus_fashion_dat...
Limit to 1000 rows
32 division_id INT,
33 department_id INT,
34 class_id INT,
35 FOREIGN KEY (division_id) REFERENCES division_dim(division_id),
36 FOREIGN KEY (department_id) REFERENCES department_dim(department_id),
37 FOREIGN KEY (class_id) REFERENCES class_dim(class_id)
38);
39
40 select * from class_dim;
41 SELECT * FROM product_fact;
Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content: Fetch rows:
product_id clothing_id age title review_text rating recommended_ind pc
1 1077 60 Some major design flaws I had such high hopes for this dress and really ... 3 0 0
2 1049 50 My favorite buy! I love, love, love this jumpsuit. it's fun, flirty, a... 5 1 0
3 847 47 Flattering shirt This shirt is very flattering to all due to the adju... 5 1 6
4 1080 49 Not for the very petite I love tracy reese dresses, but this one is not f... 2 0 4
product_fact 14
Apply Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
84 11:33:00 use women 0 row(s) affected 0.360 sec
85 11:33:14 select * from class_dim LIMIT 0, 1000 20 row(s) returned 0.062 sec / 0.000 sec
86 11:33:28 SELECT * FROM product_fact LIMIT 0, 1000 1000 row(s) returned 0.000 sec / 0.015 sec
Query Completed
Search
11:33 AM
12/16/2024


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
amazon_db
fashion_db
fashion_reviews
sakila
sys
women
Tables
class_dim
department_dim
division_dim
product_fact
Views
Administration Schemas
Information
Schema: fashion_reviews
Object Info Session
Query 1 SQL File 3* womens_reviews_csvfile_querie... womenreview_plus_fashion_dat...
Limit to 1000 rows
33 department_id INT,
34 class_id INT,
35 FOREIGN KEY (division_id) REFERENCES division_dim(division_id),
36 FOREIGN KEY (department_id) REFERENCES department_dim(department_id),
37 FOREIGN KEY (class_id) REFERENCES class_dim(class_id)
38);
39
40 select * from class_dim;
41 SELECT * FROM product_fact;
42 select * from department_dim;
43 select * from division_dim;
Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:
department_id department_name
2 Bottoms
1 Dresses
4 Intimate
5 Jackets
3 Tops
6 Trend
department_dim 15
Apply Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
85 11:33:14 select * from class_dim LIMIT 0, 1000 20 row(s) returned 0.062 sec / 0.000 sec
Query Completed
Search
11:35 AM
12/16/2024



11.3 Glossary of Terms

- **ETL (Extract, Transform, Load):** A process of extracting data from source systems, transforming it into a usable format, and loading it into a data storage solution.
- **Star Schema:** A database schema design optimized for analytics, with a central fact table connected to multiple dimension tables.
- **Dashboard:** A data visualization tool that displays key metrics and KPIs for easy analysis.
- **Normalization:** The process of organizing data to reduce redundancy and improve efficiency.
- **KPI (Key Performance Indicator):** A measurable value indicating the success of a project or process.