

E-Commerce Database Management System Project

Group 8

Contents

1	Introduction	3
2	Database Design and Implementation	3
2.1	Entity Relationship Diagram	3
2.2	Design Considerations	5
2.3	Relationships and Cardinalities	5
2.4	Logical Schema	8
2.5	Physical Schema Creation	9
2.6	Normalisation	9
3	Part 2: Data Generation and Management	11
3.1	Synthetic Data Generation	11
3.2	Data Import and Quality Assurance	12
4	Part 3: Data Pipeline Generation	13
4.1	Github Repository Creation	13
4.2	Automated Workflow	13
5	Part 4: Data Analysis and Reporting with Quarto in R	13
6	Conclusion	17
7	References	17
8	Appendices	17
8.1	Appendix 1	17
8.2	Appendix 2	18
8.3	Appendix 3	18

Masters Programmes: Group Assignment Cover Sheet

Student Numbers: Please list numbers of all group members	5586472, 5576643, 5523411, 5563941, 5575984, 5584180
Module Code:	IB9HP0
Module Title:	Data Management
Submission Deadline:	20th March 2024
Date Submitted:	20th March 2024
Word Count:	2000
Number of Pages:	15
Question Attempted: <i>(question number/title, or description of assignment)</i>	4 Questions E-Commerce Data Base Management System
Have you used Artificial Intelligence (AI) in any part of this assignment?	Yes
<p>Academic Integrity Declaration</p> <p>We're part of an academic community at Warwick. Whether studying, teaching, or researching, we're all taking part in an expert conversation which must meet standards of academic integrity. When we all meet these standards, we can take pride in our own academic achievements, as individuals and as an academic community.</p> <p>Academic integrity means committing to honesty in academic work, giving credit where we've used others' ideas and being proud of our own achievements.</p> <p>In submitting my work, I confirm that:</p> <ul style="list-style-type: none"> ▪ I have read the guidance on academic integrity provided in the Student Handbook and understand the University regulations in relation to Academic Integrity. I am aware of the potential consequences of Academic Misconduct. ▪ I declare that this work is being submitted on behalf of my group and is all our own, , except where I have stated otherwise. ▪ No substantial part(s) of the work submitted here has also been submitted by me in other credit bearing assessments courses of study (other than in certain cases of a resubmission of a piece of work), and I acknowledge that if this has been done this may lead to an appropriate sanction. ▪ Where a generative Artificial Intelligence such as ChatGPT has been used I confirm I have abided by both the University guidance and specific requirements as set out in the Student Handbook and the Assessment brief. I have clearly acknowledged the use of any generative Artificial Intelligence in my submission, my reasoning for using it and which generative AI (or AIs) I have used. Except where indicated the work is otherwise entirely my own. ▪ I understand that should this piece of work raise concerns requiring investigation in relation to any of points above, it is possible that other work I have submitted for assessment will be checked, even if marks (provisional or confirmed) have been published. ▪ Where a proof-reader, paid or unpaid was used, I confirm that the proof-reader was made aware of and has complied with the University's proofreading policy. <p>Upon electronic submission of your assessment you will be required to agree to the statements above</p>	

Load Necessary Libraries

```
library(DBI)
library(readr)
library(RSQLite)
library(dplyr)
library(stringr)
```

1 Introduction

This project simulates a real-world e-commerce data environment, including all stages of data management, from database configuration to data analysis and reporting. The database is created and managed with SQLite while GitHub Actions are used to automate processes enhancing workflow efficiency. In addition, Quarto is used in conjunction with R to conduct comprehensive data analysis and generate reports. Through this approach, technical skills and competencies necessary for effective data management are examined in-depth, mirroring the complexities and dynamics encountered in e-commerce.

2 Database Design and Implementation

2.1 Entity Relationship Diagram

Figure 1 shows the E-R diagram that simulates a real-world e-commerce data ecosystem, capturing the detailed relationships between entities and attributes essential for facilitating online transactions. In addition, it provides a comprehensive view of the e-commerce system, which serves as a platform for users to browse products, make purchases, and securely complete their payments.

2.1.1 Assumptions

- The company only distributes products within the United Kingdom (UK).
- The Currency used is Pound Sterling (GBP).
- Attributes formats will be aligned with UK standard formats such as date , addresses , names ...etc

2.1.2 Entities and Attributes

This section describes and illustrates the entities in the above ERD and their respective attributes.

- Customer

Shows us the users who have placed orders in the past. Attributes include name , email, phone number and address of the customer.

- Supplier

Vendors who supply products. This entity contains attributes of names, addresses, emails and status of whether the supplier is currently supplying product or not (Active/Inactive).

- Product

Describes all products in the stock and available for sale. Attributes include name, price, and availability of the products.

- Order Details

Emphasises all details related to placed orders. Some of its attributes includes billing and shipping addresses, type and status of payment, date and status of the order.

- Category and Sub-Category

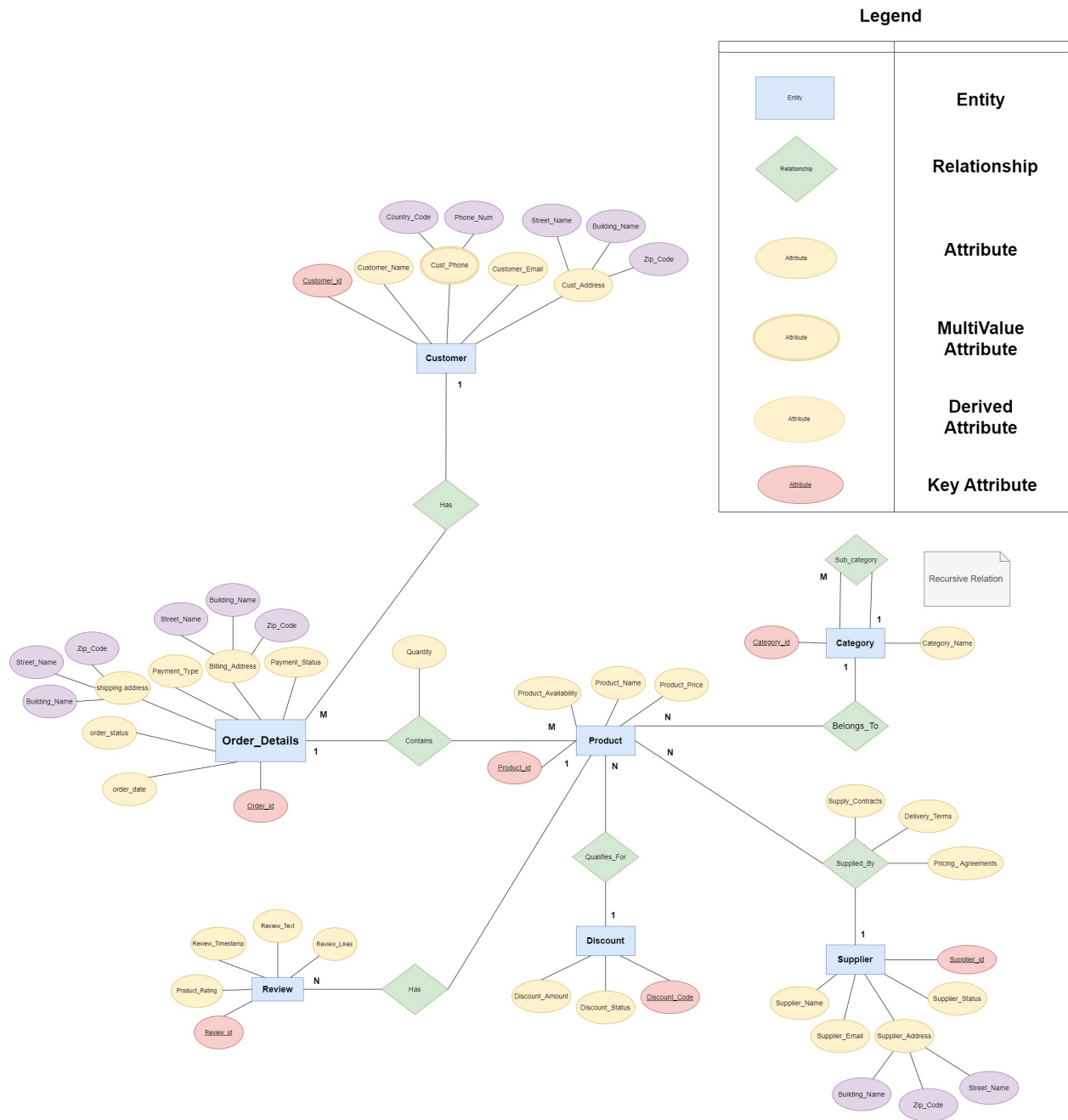


Figure 1: E-Commerce Entity Relationship Diagram

Category is the broad classification of products that share common features or are intended for a similar purpose. A sub-category is a more specific grouping of products within a category based on finer distinctions or attributes.

- Product Discounts

The voucher number or offer code to be applied to eligible products. The amount of discount it offers as well as the status of the discount are the attributes included.

- Reviews

Contains Written comments and rating of product bought by customers, the likes recieved as well as the time stamp of when the review was written.

2.2 Design Considerations

2.2.1 Absence of an Order Entity

The model intentionally skips direct order management. Instead, it focuses on product management and customer interactions through reviews and payment methods.

Order Entity not considered in this ER design in order to follow best practices by not having to include orderId as part of product table which might affect the overall performance of DB retrieval.

2.3 Relationships and Cardinalities

2.3.1 Order Detail Contains Products

This relation as can be seen in **Figure 2** indicates that each order detail contains multiple products that have been bought by the customer.

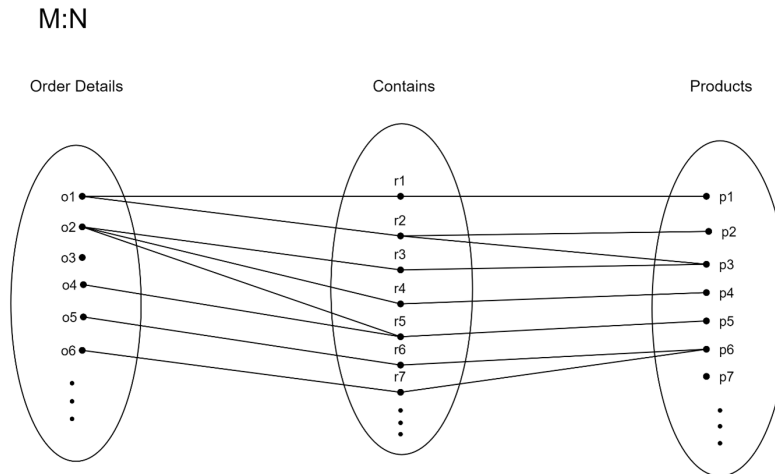


Figure 2: Order Details Contains Products

Associative Attribute (Quantity: The number of units of the product ordered in this line item.)

2.3.2 Customer Has Order Details

This relation will be created when customers orders one or more products. They will be linked with a particular order detail indicating the order status throughout the process. One customer can be associated with multiple order details at any given time as illustrated in **Figure 3**. Moreover, it aids in tracking an order's life cycle, allowing for updates, customer notifications, and management of the order fulfillment process.

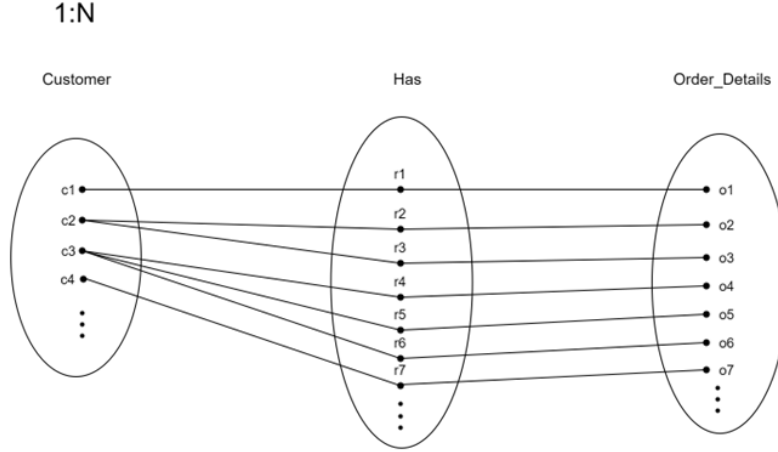


Figure 3: Customer Has Order Details

2.3.3 Product Belongs to Category

Figure 4 specify that each product is classified under a specific Category where products can belong to only one category. This enables customers to browse products by category and helps retailers manage product listings more efficiently.

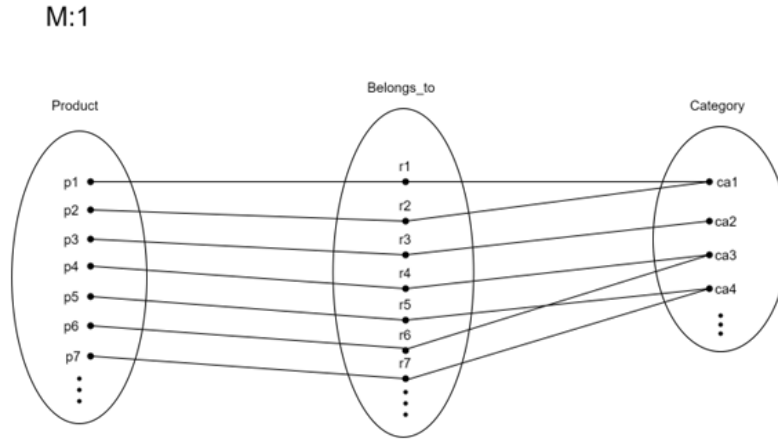


Figure 4: Product Belong to Category

2.3.4 Category Self-Reference Relation

As reflected in **Figure 5**, category can have multiple subcategories. For example, “Home Appliances” is the parent category containing subcategories like Washing Machines, Food Processors, Dishwashers etc.

2.3.5 Product Supplied By Supplier

This relation is important for inventory management, reflecting the real-world scenario where a single supplier might provide various products, but each product is supplied by only one specific supplier **Figure 6**. The relation helps track inventory sources, manage supplier relationships, and ensure product availability.

1:N

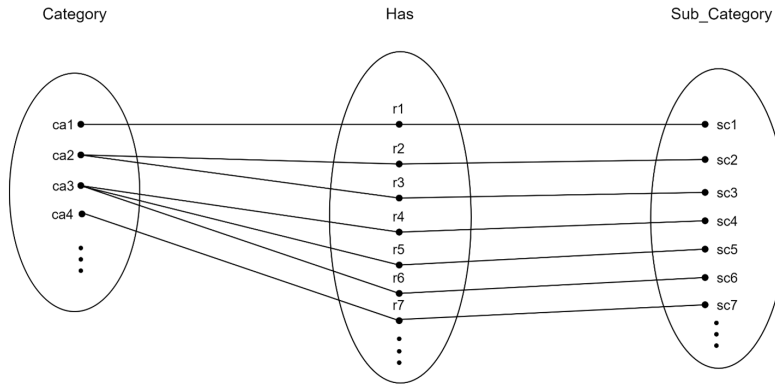


Figure 5: Category Has Sub-Categories

N:1

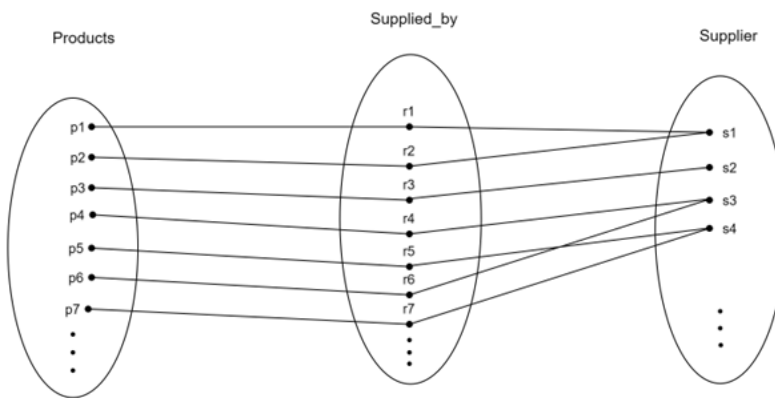


Figure 6: Products Supplied By Supplier

2.3.6 Product Qualifies For Discount

This relation signifies that a product can be eligible for a promotional discount. In this context and for simplicity the relation representing one discount code or voucher that is valid to apply on multiple eligible products **Figure 7**.

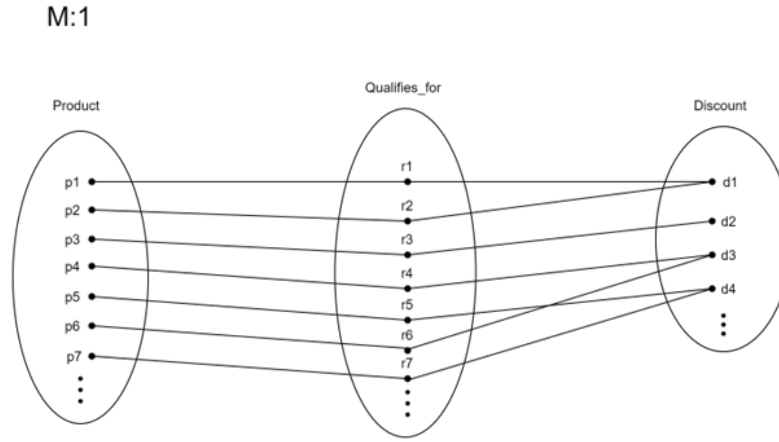


Figure 7: Product Qualifies For Discount

2.3.7 Product Has Reviews

A product's reviews are provided by Customers reflecting feedback or evaluation for a specific product or service. One product can have multiple reviews over time as shown in **Figure 8**.

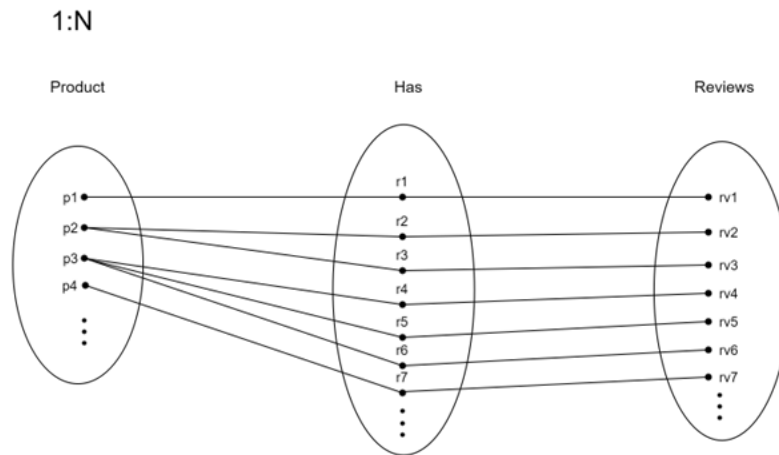


Figure 8: Product Has Reviews

2.4 Logical Schema

Legend¹

¹Italic attributes indicate foreign keys

2.4.1 Customers

Customers (Cust_ID, Cust_First_Name, Cust_Last_Name, Cust_Building_Name, Cust_Street_Name, Cust_Zip_Code, Cust_Email, Cust_Phone_Number, Cust_Country_Code)

2.4.2 Products

Products (Product_ID, *Discount_Code*, *Category_ID*, *Supplier_ID*, Product_Name, Product_Availability, Product_Price)

2.4.3 Suppliers

Suppliers (Supplier_ID, Supplier_Name, Supplier_Building_Name, Supplier_Street_Name, Supplier_Zip_Code, Supplier_Email, Supplier_Status)

2.4.4 Order_Details

Order_Details (Order_ID, *Customer_ID*, Order_Date, Shipping_Building_Name, Shipping_Street_Name, Shipping_Zip_Code, Order_Status, Payment_Type, Payment_Status, Billing_Building_Name, Billing_Street_Name, Billing_Zip_Code)

2.4.5 Discounts

Discounts (Discount_Code, *Product_ID*, Discount_Amount, Discount_Status)

2.4.6 Reviews

Reviews (Review_ID, *Product_ID*, Review_Timestamp, Product_Rating, Review_Text, Review_Likes)

2.4.7 Categories

Categories (Category_ID, *Sub_Cat_ID*, Category_Name)

2.4.8 Many to Many : Order_details - Product

Order_Items (Order_ID, Product_ID, Quantity)

2.5 Physical Schema Creation

```
#Creating the db schema in SQL
con <- dbConnect(RSQLite::SQLite(), "ecommerce.db")
sql_file <- readLines("dbScript.sql")
for (sql_command in sql_file) {
  if (sql_command!=""){
    print(sql_command)
    dbExecute(con,sql_command)
    print("-----DONE-----")
  }
}
```

2.6 Normalisation

The First Normal Form (1NF) mandates atomic values and unique rows via a primary key without duplicate columns. The tables were verified to meet these criteria. Second Normal Form (2NF) is achieved when a table in 1NF has all non-key attributes fully dependent on the primary key, thus avoiding partial dependencies on composite keys which has also been achieved. Finally, the team conducted thorough checks to ensure all tables comply with 3NF requirements, as detailed in subsequent sections.

2.6.1 Customers:

Looking at the table, we can identify the following dependencies:

$Cust_ID \rightarrow Cust_First_Name$
 $Cust_ID \rightarrow Cust_Last_Name$
 $Cust_ID \rightarrow Cust_Building_Number$
 $Cust_ID \rightarrow Cust_Building_Name$
 $Cust_ID \rightarrow Cust_Street_Name$
 $Cust_ID \rightarrow Cust_Country_Code$
 $Cust_ID \rightarrow Cust_Email$
 $Cust_ID \rightarrow Cust_Phone_Number$

Figure 9: Dependencies in Customers table

- Based on this analysis, the table appears to be in 3NF. Each non-key attribute depends only on the primary key ($Cust_ID$), and there are no transitive dependencies.

2.6.2 Order_Items:

$Product_ID, Order_ID \rightarrow Quantity$

Figure 10: Dependencies in Order_Items table

$Product_ID$ and $Order_ID$ combined are representing composite key, and $Quantity$ is functionally dependent on $Product_ID$ and $Order_ID$. There are no transitive dependencies here.

2.6.3 Order_Details:

$Order_ID \rightarrow Order_Date$
 $Order_ID \rightarrow Order_Status$
 $Order_ID \rightarrow Payment_Type$
 $Order_ID \rightarrow Payment_Status$
 $Order_ID \rightarrow Cust_ID$
 $Order_ID \rightarrow Shipping_Building_Name$
 $Order_ID \rightarrow Shipping_Street_Name$
 $Order_ID \rightarrow Shipping_Zip_Code$
 $Order_ID \rightarrow Billing_Building_Name$
 $Order_ID \rightarrow Billing_Street_Name$
 $Order_ID \rightarrow Billing_Zip_Code$

Figure 11: Dependencies in Order_Details table

Given that $Cust_ID$ is a foreign key in this table and that there is a separate Customers table where $Cust_ID$ serves as the primary key, it suggests that $Cust_ID$ is not functionally dependent on any other attribute within this table. It is merely referencing the primary key of another table.

Therefore, the dependency involving $Cust_ID$ does not violate 3rd Normal Form (3NF). The table appears to be in 3NF as all other attributes are functionally dependent only on the primary key ($Order_ID$)

2.6.4 Product Category

$Category_ID$ is the primary key and it uniquely identifies $Category_Name$.

2.6.5 Product Discount

Analyzing the functional dependencies, it is found that $Discount_Code$ is the primary key and it uniquely identifies $Discount_Amount$ and $Discount_Status$. Each $Discount_Code$ corresponds to a specific Dis-

Category_ID \rightarrow Category_Name
Category_ID \rightarrow Sub_Cat_ID

Figure 12: Dependencies in Category table

Discount_Code \rightarrow Discount_Amount
Discount_Code \rightarrow Discount_Status

Figure 13: Dependencies in Discounts table

count_Amount and Discount_Status, ensuring that the table adheres to the principles of 3rd Normal Form (3NF). There are also no transitive dependencies or non-key attributes determining other attributes within the table.

2.6.6 Products

Product_ID \rightarrow Product_Name
Product_ID \rightarrow Product_Price
Product_ID \rightarrow Product_Availability
Product_ID \rightarrow Supplier_ID
Product_ID \rightarrow Discount_Code
Product_ID \rightarrow Category_ID

Figure 14: Dependencies in Products table

The table also contains foreign keys, namely Supplier_ID, Discount_Code, and Category_ID. These foreign keys are merely referencing the primary key of their respective tables and are not functionally dependent on any other attribute within this table. Examining the non key attributes' functional dependencies, Product_ID emerges as the primary key, in which Product_Name, Product_Price and Product_Availability appears to be functionally dependent on it.

2.6.7 Reviews

In the Reviews table, Review_ID acts as the primary key, ensuring uniqueness. Attributes like Review_Timestamp, Product_Rating, Review_Text and Review_Likes are dependent on Review_ID.

2.6.8 Suppliers

In the Suppliers table, Supplier_ID acts as the primary key. Attributes such as Supplier_Name, Supplier_Building_Name, Supplier_Building_Number, Supplier_Street_Name, Supplier_Zip_Code, Supplier_Email and Supplier_Status are dependent on Supplier_ID.

Given these functional dependencies where each attribute seems to be functionally dependent on the primary key, with no non-key attributes determining other, all tables appears to adhere to the principles of 3rd Normal Form (3NF).

3 Part 2: Data Generation and Management

3.1 Synthetic Data Generation

After the agreement on the schema mentioned in the previous section, the team started to generate synthetic data that to some extent, imitated realistic e-commerce as much as possible.

```

Review_ID → Review_Timestamp
Review_ID → Product_Rating
Review_ID → Review_Text
Review_ID → Review_Likes

```

Figure 15: Dependencies in Reviews table

```

Supplier_ID → Supplier_Name
Supplier_ID → Supplier_Building_Name
Supplier_ID → Supplier_Building_Number
Supplier_ID → Supplier_Street_Name
Supplier_ID → Supplier_Zip_Code
Supplier_ID → Supplier_Email
Supplier_ID → Supplier_Status

```

Figure 16: Dependencies in Suppliers table

ChatGPT has been used as the main tool for this step as an alternative to Mockaroo, as the former produces more structural and logical data than the latter. [Appendix 1,2]. For tables consisting of foreign keys, R code has been used for assignment based on logic set by the team Appendix 3.

3.2 Data Import and Quality Assurance

To enhance data quality for e-commerce analysis and validate the csv data generated, we use R to perform 4 main checks [Refer data_validate.R]:

- Primary key integrity check
- Duplicate entry check
- Email id pattern check
- Phone number pattern check

```

[1] "CHECKING FILE : Customers"
[1] "---No invalid emails exist"
[1] "---No invalid phone numbers exist"
[1] "---No duplicate entries found in the data"
[1] "CHECKING FILE : Order_Details"
[1] "---No duplicate entries found in the data"
[1] "CHECKING FILE : Order_Item"
[1] "CHECKING FILE : Product_Category"
[1] "---No duplicate entries found in the data"
[1] "CHECKING FILE : Product_Discounts"
[1] "---No duplicate entries found in the data"
[1] "CHECKING FILE : Products"
[1] "---No duplicate entries found in the data"
[1] "CHECKING FILE : Reviews"
[1] "---No duplicate entries found in the data"
[1] "CHECKING FILE : Suppliers"
[1] "---No invalid emails exist"
[1] "---No duplicate entries found in the data"

```

Figure 17: Data validation of CSV files

Once the data has gone through all the checks, the updated csv files will be used to import into SQL. Importing the data into SQL is also done in R [Refer data_update.R]. This method focused on refining datasets for accurate analysis by identifying and eliminating invalid entries and duplicates, thereby maintaining the

dataset's uniqueness and reliability before inserting into Data Base.

4 Part 3: Data Pipeline Generation

To efficiently collaborate on this assignment, we have uploaded the project to Github. This was done so that the team could track changes, follow progress, collaborate on bug fixes etc. Moving the project to Github also allowed for implementation of version control and continuous integration.

4.1 Github Repository Creation

The pipeline generation process consists of two phases. In the first phase, github repository has been created^[1] followed by integrating the posit cloud project. Additionally, all teams members have been added as collaborators.

4.2 Automated Workflow

Second phase was the automating our workflow. Using Github Workflows, we were able to automate the process of validating data, updating data and data analysis. The workflow is triggered on pushing new data to the project. Once all 3 steps are completed successfully, the new files are committed and pushed to the project.

Figures 10, 11 shows the updated analysis after 1st run of workflow (49) and then after second run of workflow (150).

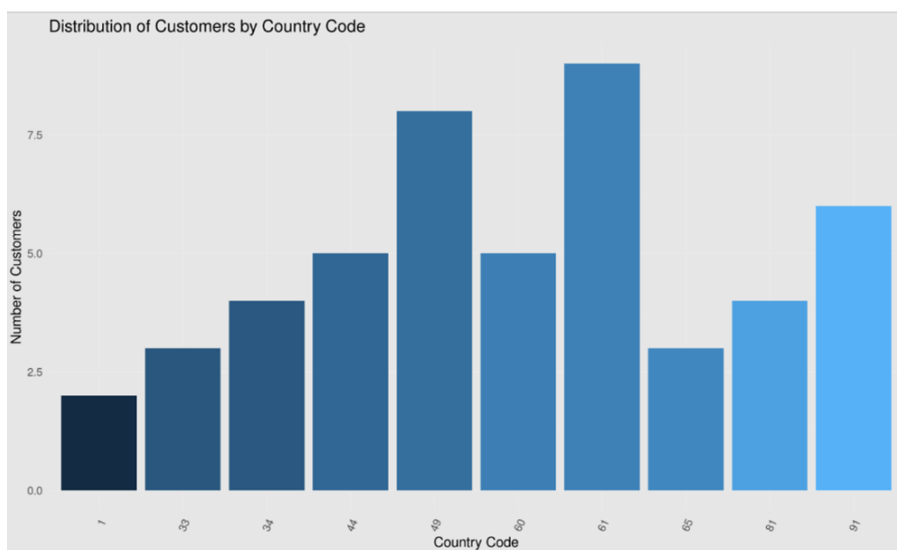


Figure 18: The histogram of customers data for 50 records

Figure 12 illustrate console output showing the record count in the sql database after data update using the workflow.

In **Figure 13** the snippet is Timestamps that showed as an evidence of a working workflow

5 Part 4: Data Analysis and Reporting with Quarto in R

Information regarding product ratings, sales revenue, customer distribution, and order status was obtained through the establishment of a connection to an SQLite database and the execution of SQL queries. Subse-

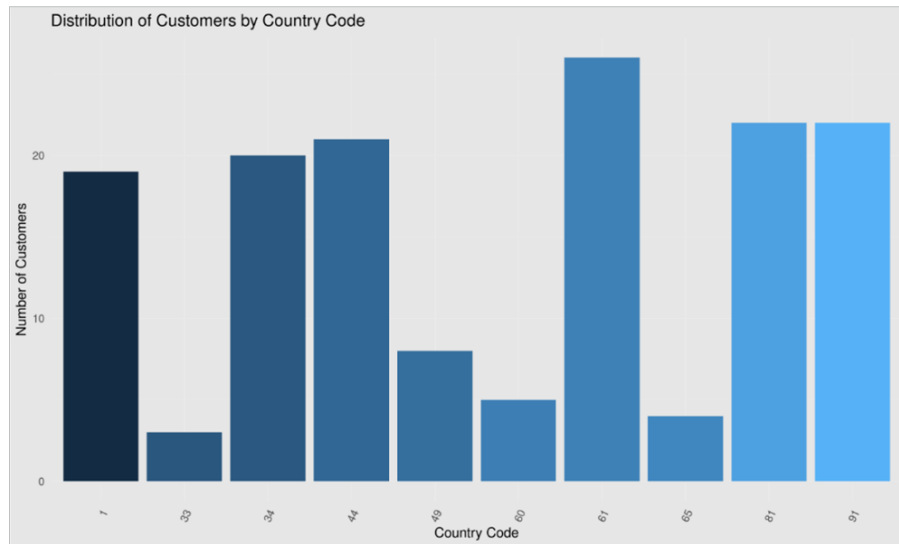


Figure 19: The histogram after workflow run and inserted additional 100 records

```
sqlite> .quit
/cloud/project$ sqlite3 ecommerce.db
SQLite version 3.31.1 2020-01-27 19:55:54
Enter ".help" for usage hints.
sqlite> select count(*) from Customers
...> ;
49
sqlite> .quit
/cloud/project$ sqlite3 ecommerce.db
SQLite version 3.31.1 2020-01-27 19:55:54
Enter ".help" for usage hints.
sqlite> select count(*) Customers;
1
sqlite> select count(*) from Customers;
150
sqlite> █
```

Figure 20: Record count after data update using workflow

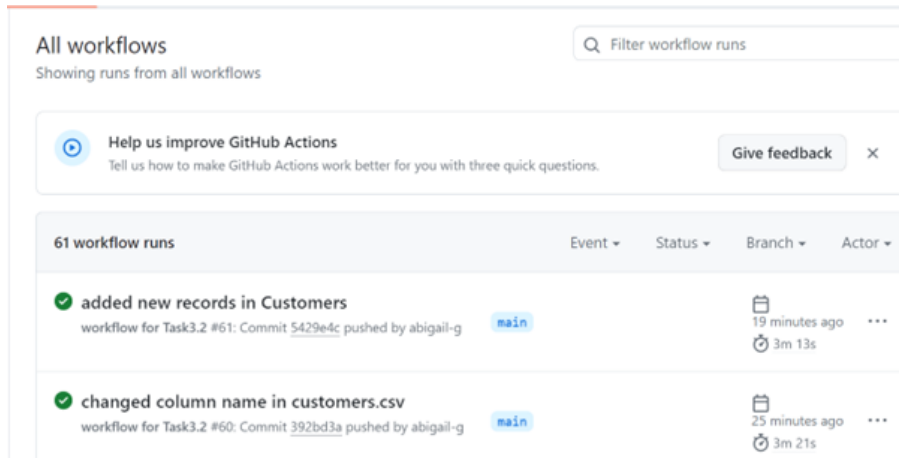


Figure 21: Workflow Timestamps

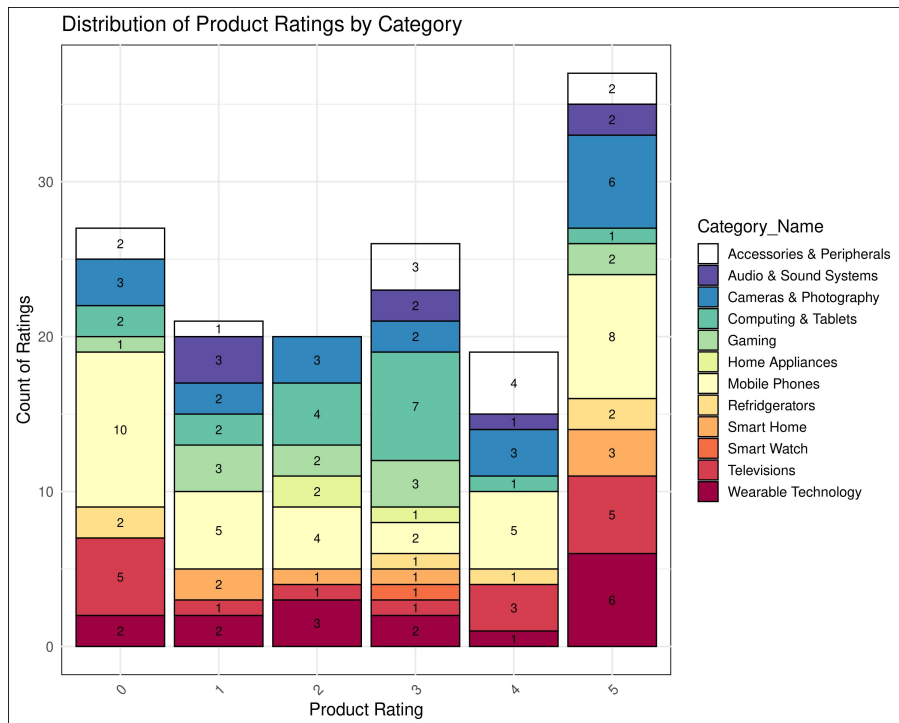


Figure 22: Distribution of Product Ratings by Category

quently, data visualisation techniques were used to identify the distribution and patterns within the data, offering insights into e-commerce operational dimensions.

The graph in **Figure 14** shows the distribution of product ratings from 0 to 5 coloured by category name.

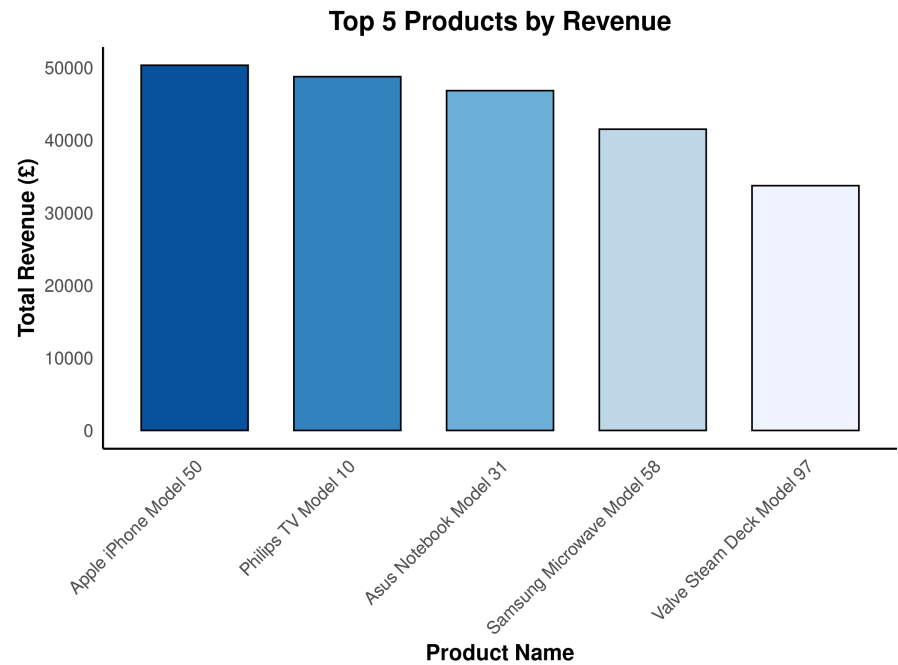


Figure 23: Top 5 Products by Revenue

Figure 15 reflecting Top 5 Products Sorted by total Revenue.

After joining the “Order_Items”, “Discounts”, and “Products” tables, the derived value, which is the “Total Revenue”, is calculated by multiplying “Order_Item” quantity and “Product_Price” and deducting it with “Discount_Amount”.

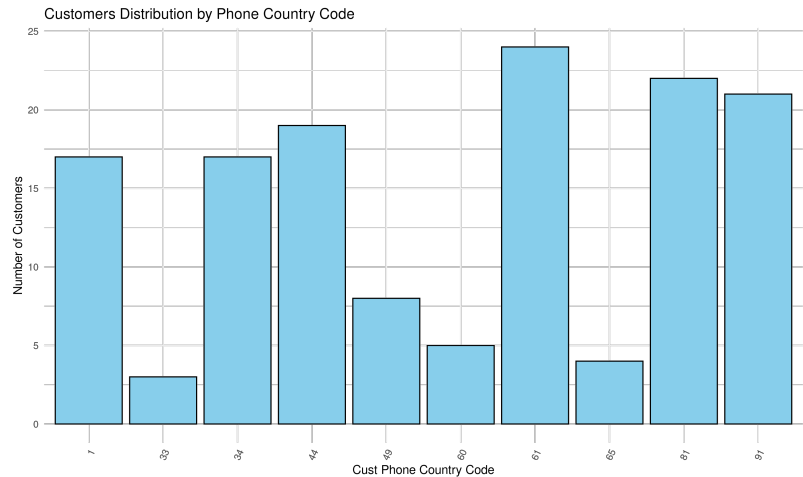


Figure 24: Distribution of Customers by Phone Country Code

The graph in **Figure 16** shows the distribution of customers by phone country codes.

Figure 17 displays the order status count of shipped, processing, delivered and cancelled orders.

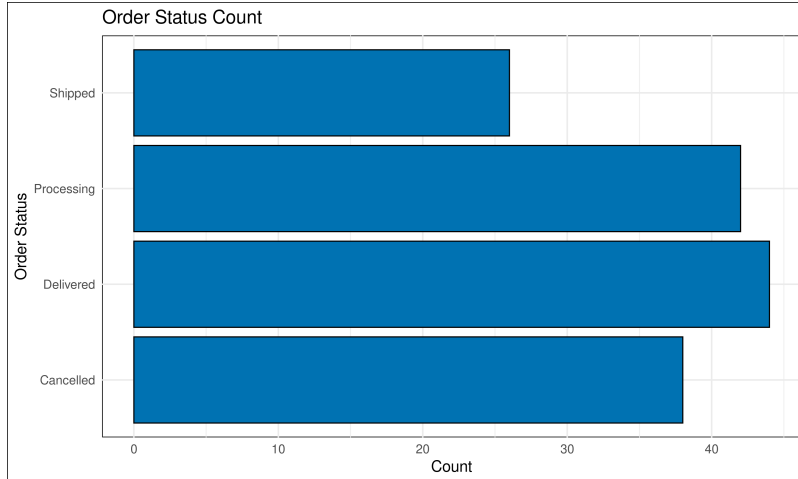


Figure 25: Order Status Count

6 Conclusion

Upon completion of this project, valuable insights are gained regarding the challenges and methodologies associated with an e-commerce context, including the design of databases, analysis of data, and presentation of findings in a clear and impactful manner. Ultimately, this project serves as a reference for future e-commerce projects in particular and data-driven projects in general, providing valuable insight into the modern data management landscape.

7 References

1. To be redirected to the project repository click [here](#)

8 Appendices

8.1 Appendix 1

ChatGPT 4

You

```
CREATE TABLE 'Products' (
  'Product_ID' VARCHAR(250) PRIMARY KEY,
  'Product_Name' VARCHAR(250) NOT NULL,
  'Product_Price' FLOAT(10,2) NOT NULL,
  'Product_Availability' VARCHAR(25) );
```

generate a csv file with 150 observations, each product ID needs to be P with random 7 digits, product price must range between 50 to 2000. The product name must be from these category including televisions, computing & tablets, mobile phones, home appliances, audio&sound systems, cameras and photography, gaming (PS5, Xbox etc), Smart home, Wearable Technology, Accessories & Peripherals , every column must be unique, the product name must be unique

ChatGPT

The CSV file with 150 observations has been generated successfully. Each product ID consists of a "P" followed by 7 random digits, with product prices ranging between \$50 and \$2000. The product names are unique and categorized into televisions, computing & tablets, mobile phones, home appliances, audio & sound systems, cameras and photography, gaming, smart home, wearable technology, and accessories & peripherals. The availability of each product is also noted.


You can download the CSV file from the following link: [products.csv](#). [↗]

Figure 26: ChatGPT_1

8.2 Appendix 2

ChatGPT 4 ▾

You

 Customers.csv
Spreadsheet

can you generate more data, from 50 to 150, do not duplicate, make sure that each columns are distinct

ChatGPT

The uploaded CSV file, "Customers.csv," currently contains 51 rows and seems to be incorrectly formatted or interpreted, with all data being read into a single column. This might be due to the delimiters or the structure of the file itself.

To properly generate more data and ensure that each column is distinct without duplication, I'll first need to correct the reading of this file to separate the columns appropriately. Let's try to read the file again with a more suitable delimiter. [↩]

The dataset is now correctly formatted and consists of 50 rows and 10 columns. Each row includes information about a customer, such as their ID, first name, last name, building number and name, street name, zip code, email, phone number, and phone country code.

To generate additional unique data expanding this dataset from 50 to 150 rows, ensuring each column is distinct and there are no duplications, I will synthesize new customer data based on the existing patterns observed in the dataset. Let's proceed with generating this additional data. [↩]

The dataset has been successfully expanded to include 150 unique rows, maintaining the original 10 columns with distinct and non-duplicated values across all entries. This enriched dataset now comprises a wider variety of customer information, ready for any intended analysis or processing. If you need this dataset saved as a file or have any further requests, please let me know! [↩]

Figure 27: ChatGPT_2

8.3 Appendix 3

```
88 # To apply the foreign key into the table |
89 # Define a function to assign Category_ID based on keywords in Product_Name
90 - assign_category_id <- function(Product_Name) {
91   if (grepl("TV|Television", Product_Name, ignore.case = TRUE)) {
92     return("CAT1")
93   } else if (grepl("Laptop|Tablet|Computing|Book|Surface|Monitor", Product_Name, ignore.case = TRUE)) {
94     return("CAT2")
95   } else if (grepl("Phone|Galaxy|Mi|P Series|OnePlus", Product_Name, ignore.case = TRUE)) {
96     return("CAT3")
97   } else if (grepl("Washing Machine|Home Appliance|Vacuum|Dishwasher", Product_Name, ignore.case = TRUE)){
98     return("CAT4")
99   } else if (grepl("Headphones|Speakers|Sound System|Earbuds|Speaker|Technica|Soundbar", Product_Name, ignore.case = TRUE)) {
100     return("CAT5")
101   } else if (grepl("Camera|Photography|GoPro|Mirrorless|Nikon|Camcorder|Compact", Product_Name, ignore.case = TRUE)) {
102     return("CAT6")
103   } else if (grepl("Xbox|PS|Gaming|Switch", Product_Name, ignore.case = TRUE)) {
104     return("CAT7")
105   } else if (grepl("Smart Home|Echo|Smart Lock|Steam Deck|Hue Light", Product_Name, ignore.case = TRUE)) {
106     return("CAT8")
107   } else if (grepl("Wearable|Quest|Tracker|Gear|Band|Glasses", Product_Name, ignore.case = TRUE)) {
108     return("CAT9")
109   } else if (grepl("Keyboard|Mouse|Peripheral|Thermostat", Product_Name, ignore.case = TRUE)) {
110     return("CAT10")
111   } else if (grepl("Refrigerator", Product_Name, ignore.case = TRUE)){
112     return("CAT11")
113   } else if (grepl("Microwave", Product_Name, ignore.case = TRUE)){
114     return("CAT12")
115   } else if (grepl("Smart Tech", Product_Name, ignore.case = TRUE)){
116     return("CAT13")
117   } else if (grepl("Watch", Product_Name, ignore.case = TRUE)) {
118     return("CAT14")
119   } else if (grepl("Projectors", Product_Name, ignore.case = TRUE)) {
```

Figure 28: Data Update