Fall 2022

Project 1: BUAN 6320.005 Database Foundations for Business Analytics – Database Design Project

E-Commerce Database

Parthasarathi, Prriyamvradha (pxp220005)
Jawahar Vasagam, Premi (pxj220007)
Gopisetty, Jatin (jxg210091)
Moravaneni, Sai Priya (sxm220113)

Step 1: Choose a Dataset

The dataset chosen for the Database design project is an E-Commerce dataset.

- The dataset was a free resource and was chosen from Data world website
- The size of the dataset is 739KB. It comprises of 34 columns and 3455 rows
- The dataset is structured
- The datatypes of the columns are integer, floating point constants, variable character, and date
- 9 Tables were derived from this dataset
- The dataset does not contain any missing values
- The dataset can be used to derive Business insights to improve the e-Commerce business.

E-Commerce_Dataset
.xlsx

Step 2: Business Understanding

1. **Why has this data been gathered?**
    - This data has been gathered to help the company keep track of its business key performance indicators with regards to their customers, sales, products sold, profit margins and the orders.
    - Since there are many e-commerce websites for consumers to choose from, gathering this data will give insights for the company to keep stock of their products based on the orders since demand is usually high on e-commerce platforms
2. **What can be done with data? What can we achieve?**
    - We can take business decisions regarding the selling price after reviewing the demand of orders to see if they can generate more profit and give loyalty discounts to regular customers since it will be captivating to the consumers
    - Once we analyze the data and make the necessary changes to selling prices depending on trends and we will be able to edge out competitors
3. **What are some of the goals/targets we have regarding the business that we can achieve by investigating this data?**
    - We can increase customer retention by analyzing the data to modify the loyalty discounts
    - We can make changes to product prices depending on popularity and increase profits

4. **What insightful information can this data provide us that can be used to improve the business?**
    - We can find out which seller's products are preferred by customers
    - We can find the best-selling products
    - We can also see which payment type customers frequently use and introduce promotions with it
5. **Why are we studying this data?**
    - We are studying this data to find out how the company can increase its customer base and expand its business into other markets
    - We can use this data to forecast customer demand over time
6. **Are there any problems in our business (based on the given data)?**
    - The problem in our business is that some high selling items are not being sold effectively, i.e., they can be used to make more profit
    - Stocks for popular items are not being properly maintained
7. **Can we find any solutions to these problems by studying this data?**
    - We can use this data to keep correct stock of our products and avoid any last-minute emergencies
    - We can survey the markets which have more potential and consider selling more products in that sector to expand the business
8. **What are some of the things we can optimize/improve in our business by studying this data?**
    - If we decide to explore more sectors, we can use this data as a model to understand which customers, locations and sellers will give the best output for our business
    - We can also create better reports and dashboards using the existing data to understand which aspects of the business is working out best and work upon it

Step 3: Data Understanding

1. **What information each column of the data contains**

Column Name	Description
ORDER ID	Provides the unique identification identity allotted for each order.
ORDER DATE	Provides the date the order Is made by the customer.
UNITS IN ORDER	Provides the units in each order.
ORDER TOTAL	Provides the total price of the order made by the customer
SELLER ID	Provides the total price of the order made by the customer
SELLER CONTACT NAME	Provides the name of each seller.
SELLER ADDRESS	Provides the address of each seller.
SELLER ZIP CODE	Provides the zip code for each seller.
SELLER PHONE	Provides the contact number for the seller.

SELLER EMAIL	Provides the email address of each seller.
CUSTOMER ID	Provides a unique identification for each customer.
FIRST NAME	Provides the first name of the customers.
LAST NAME	Provides the last name of the customers.
ADDRESS	Provides the Address of the customers.
ZIP CODE	Provides the zip code of the customers.
PHONE NUMBER	Provides the phone number of the customers.
EMAIL ADDRESS	Provides the email address of the customers.
CITY ID	Provides a unique identification for each city.
CITY	Provides the City of the Customer and Seller
STATE ID	Provides a unique identification identity for each state.
STATE	Provides the State of the Customer and Seller
LOYALTY DISCOUNT	Provides the date the order Is made by the customer.
PRODUCT_ID	Provides a unique identification number for each product.
PRODUCT NAME	Provides the name of each product.
PRICE PER UNIT	Provides the price of each unit.
TRANSACTION ID	Provides a unique identification number for each transaction.
PAYMENT TYPE	Provides the type of payment each customer pays through.
PAYMENT TYPE ID	Provides the unique identifier for each payment type.

2. The data types of each column

Column Name	DATA TYPE
Order ID	DOUBLE
Order Date	DATE
Units in Order	INT
Order Total	FLOAT
Brand ID	DOUBLE
Brand Name	VARCHAR (45)
Seller ID	DOUBLE
Seller Contact Name	VARCHAR (45)
Seller Address	VARCHAR (45)
Seller Zip Code	DOUBLE
Seller Phone	DOUBLE
Seller Email	VARCHAR (45)
City	VARCHAR (45)
State ID	DOUBLE
STATE	VARCHAR (45)
Customer ID	DOUBLE
First Name	VARCHAR (45)

Last Name	VARCHAR (45)
Address	VARCHAR (45)
ZIP Code	DOUBLE
Phone Number	DOUBLE
Email Address	VARCHAR (45)
Loyalty Discount	FLOAT
Product ID	DOUBLE
Product Name	VARCHAR (45)
Price per Unit	FLOAT
Units in Stock	INT
Transaction ID	DOUBLE
Payment Type ID	DOUBLE
Payment Type	DOUBLE

3. **What are some of the values each column contains**

   There are many attributes in the dataset and the values and ranges for a few of them have been described below:

   - For the brand name, we have 10 individual values: Adidas, CavinKare, Nike, Jockey, Ekouser, Iris, Moyee, Puma, Lilly, and Zando
   - For the states, we have all 50 states in the USA
   - For payment type, we have 7 types: e-cheque, apple pay, zelle, credit card, debit card, cash on delivery, and PayPal
   - The order ID, brand ID, seller ID, customer ID, state ID, product ID, and transaction ID are all unique numbers

4. **Verify the data quality**

   - Verify the quality of the name of the columns
     - Do you need to change any of the column names? Propose proper column names if the names do not look good to you
       - No, we do not need to change any of the column names
   - Are there any missing values? If yes, then what columns and what percentage?
     - No, there are no missing values in the dataset
   - Are there any duplicate data?
     - Yes, there were duplicates in our database. We removed 952 duplicate values from the state sheet.

5. **Provide simple statistics of the data for each column**

   - For example: range, mode, mean, median, variance, counts (frequency)

Order Table

Column Name	Quality	Values	Range	Mean	Standard Deviation	Count	Median	Mode	Variance
Order ID	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Order Date	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Units in Order	Acceptable	Unique	9	5.547	2.86	3455	6	2	8.225
Order Total	Acceptable	Unique	1233.791	254.526	222.655	3455	189.72	79.38	49575.48

Seller Table

Column Name	Quality	Values	Range	Mean	Standard Deviation	Count	Median	Mode	Variance
Brand ID	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Brand Name	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Seller ID	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Seller Contact Name	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Seller Address	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Seller Zip Code	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Seller Phone	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Seller Email	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A

City and State Table

Column Name	Quality	Values	Range	Mean	Standard Deviation	Count	Median	Mode	Variance
City ID	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
City	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
State ID	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
State	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A

Customer Table

Column Name	Quality	Values	Range	Mean	Standard Deviation	Count	Median	Mode	Variance
Customer ID	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
First Name	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Last Name	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Address	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
ZIP Code	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Phone Number	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A

Email Address	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Loyalty Discount	Acceptable	Unique	0.1	0.05	0.0322	3455	0.05	0.1	0.001

Product Table

Column Name	Quality	Values	Range	Mean	Standard Deviation	Count	Median	Mode	Variance
Product ID	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Product Name	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Price per Unit	Acceptable	Unique	119.28	48.325	30.592	3455	41.86	74.16	935.906

Payment Table

Column Name	Quality	Values	Range	Mean	Standard Deviation	Count	Median	Mode	Variance
Transaction ID	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Payment Type ID	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A
Payment Type	Acceptable	Unique	N.A	N.A	N.A	3455	N.A	N.A	N.A

- **Describe what these values mean, especially if you found something interesting**
  - The E-commerce website can use these metrics to determine the average no of product sold per month and stock up the products accordingly. Additionally, the most popular brands can be identified and the stocks for that particular brand can be increased to improve the sales and profit.

6. **Try to understand the relationships between the columns of the data**

- **What relationships can you find between the columns?**

  Below relationships exists between the columns

  - Order and Customer – Each Order ID is associated with one Customer ID
  - Order and Transaction – Each Transaction ID is associated with one Order
  - Payment and Transaction – Each Transaction is associated with one Payment Type
  - Customer, City and State – Each Customer is associated with one City and State
  - City and State – Each State is associated with two or more cities
  - Product and Customer – Each Customer ID is associated with one Product
  - Seller and Product – Each Product is associated with one or more Seller.

7. **Are there any functional dependencies in the data?**

  - Yes, there are Functional dependencies in the data. The dependencies are explained in detail in step 4: 4a

Step 4: Design a Database

1. **Requirement Analysis**: Covered in Step 2
2. **Data Understanding**: Covered in Step 3
3. **Schema Design**:
   a. **Find entities, their attributes, their primary keys, and relationships between them**

Entities - The following are the entities in the E-Commerce dataset.

- Customer
- Product
- Order
- Seller
- Payment
- Payment Type
- City
- State
- Seller Details

Attributes - The following are the attributes in the E-Commerce dataset

CUSTOMER
CUSTOMER ID
FIRST NAME
LAST NAME
ADDRESS
ZIP CODE
PHONE NUMBER
EMAIL ADDRESS
CITY ID
LOYALTY DISCOUNT

CITY
CITY ID
CITY
STATE ID

STATE
STATE ID
STATE

SELLER
PRODUCT ID
SELLER ID

SELLER DETAILS
SELLER ID
BRAND NAME
SELLER CONTACT NAME
SELLER ADDRESS
SELLER ZIP CODE
SELLER PHONE
SELLER EMAIL
CITY ID

ORDER
ORDER ID
ORDER DATE
CUSTOMER ID
PRODUCT ID
TRANSACTION ID
UNITS IN ORDER
ORDER TOTAL

PRODUCT
PRODUCT ID
PRODUCT NAME
PRICE PER UNIT

PAYMENT
PAYMENT ID
PAYMENT TYPE ID

PAYMENT TYPE
TRANSACTION ID
PAYMENT TYPE ID

Keys Attributes - The Key attributes in the E-Commerce dataset are

<u>Primary Key</u> – Customer ID, Transaction ID, Product ID, Order ID, Seller ID, Payment Type ID, State ID

<u>Foreign Key</u> – Customer ID, City ID, Seller ID, State ID, Product ID, Transaction ID, Payment Type ID,

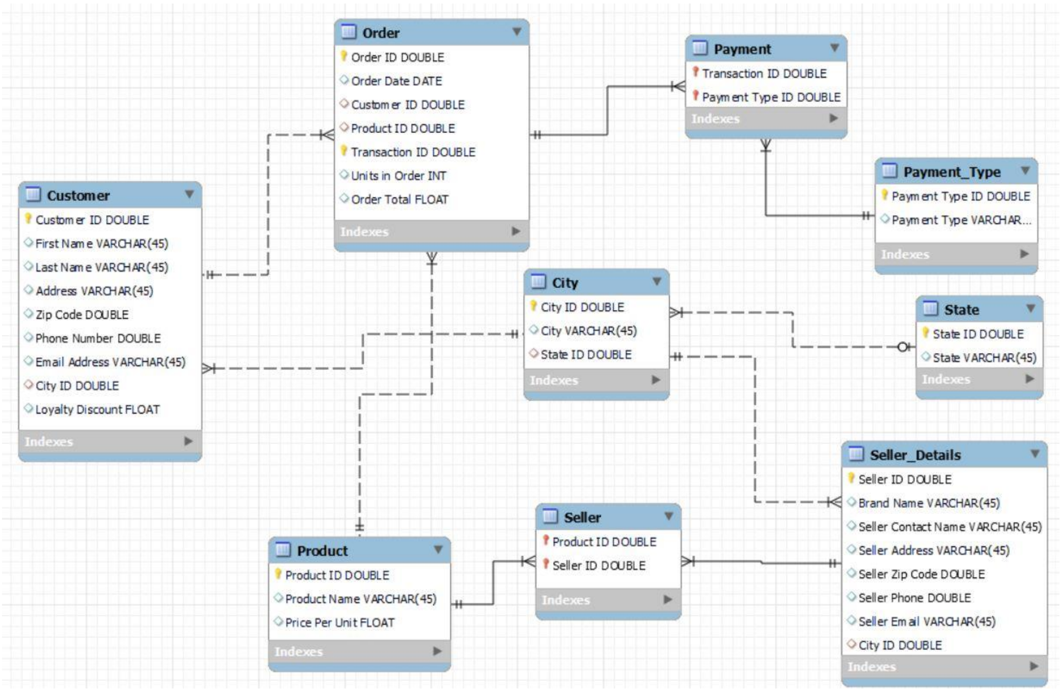
Relationships -

Below are three different relationships identified in the E-Commerce dataset.

- One to one relationship
- One to many relationship
- Zero to many relationships

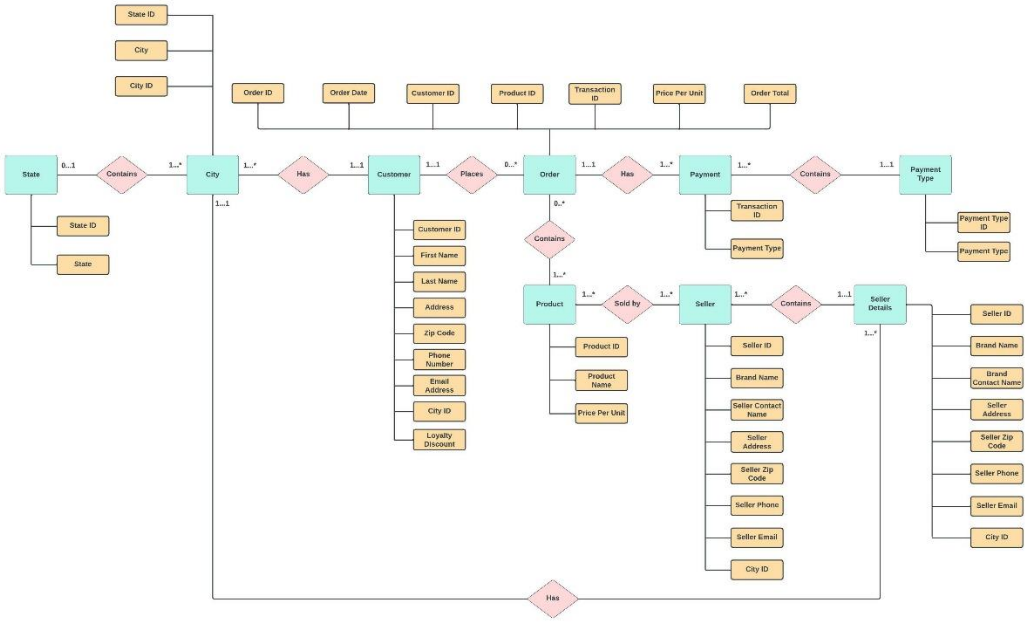
b. Model all the constraints you believe should be there in your schema

Below is the schema diagram of the E-Commerce dataset



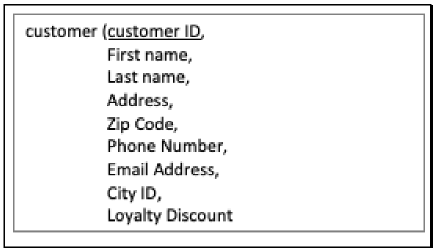
c. Draw and ER diagram of your dataset

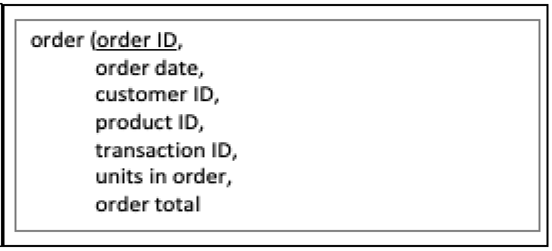
Below is the Entity Relationship Diagram for E-Commerce dataset.

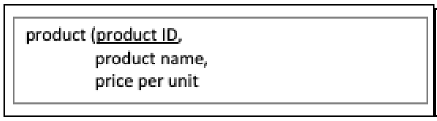


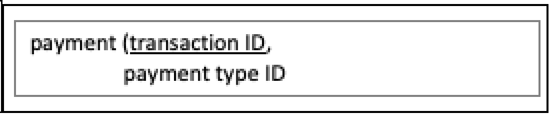
d. Translating ER Diagram into Relations

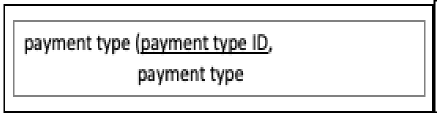
From the ER diagram above, the following relations can be derived:

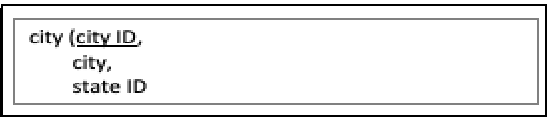

customer (customer ID,
First name,
Last name,
Address,
Zip Code,
Phone Number,
Email Address,
City ID,
Loyalty Discount

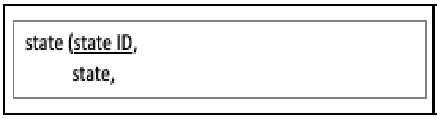

order (order ID,
order date,
customer ID,
product ID,
transaction ID,
units in order,
order total

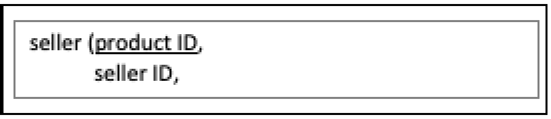

product (product ID,
product name,
price per unit

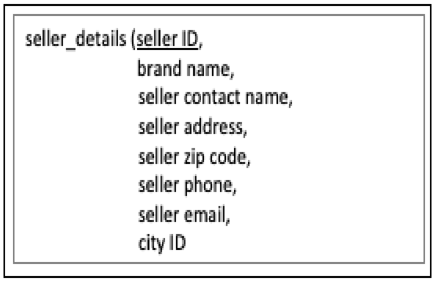

payment (transaction ID,
payment type ID


payment type (payment type ID,
payment type


city (city ID,
city,
state ID


state (state ID,
state,


seller (product ID,
seller ID,



```
seller_details (seller ID,
                brand name,
                seller contact name,
                seller address,
                seller zip code,
                seller phone,
                seller email,
                city ID
```

4. Schema Normalization

a. Find all the functional dependencies you can from your schema

Customer Table:

{First Name, Last Name, Address, Zip Code, Phone Number, Email Address, City ID} --> Customer ID

City Table:

{City, State ID} --> City ID

State Table:

{State} --> State ID

Product Table:

{Product Name, Price Per Unit} --> Product ID

Seller Table:

{Seller ID} --> Product ID

Seller Details:

{Brand Name, Seller Contact Name, Seller Address, Seller Zip Code, Seller Phone, Seller Email, City ID} --> Seller ID

Order Details:

{Order Date, Customer ID, Product ID, Transaction ID, Units in Order, Order Total} --> Order ID

Payment Table:

{Payment Type ID} --> Transaction ID

Payment Type Table:

{Payment Type} --> Payment Type ID

b. **Check if the keys you have chosen for your relations are minimal**

Customer Table:

Set of all Attributes

A = {Customer ID, First Name, Last Name, Address, Zip Code, Phone Number, Email Address, City ID}

Functional Dependencies

F = Customer ID --> {First Name, Last Name, Address, Zip Code, Phone Number, Email Address, City ID}

X = {Customer ID}

Using F, X+= {Customer ID, First Name, Last Name, Address, Zip Code, Phone Number, Email Address, City ID}

No more attributes can be added to X+

So {Customer ID} is the Key

Similarly, {Order ID}, {Product ID}, {Seller ID}, {City ID}, {State ID}, {Transaction ID} and {Payment Type ID} are the keys for the respective tables.

c. **Check if your schema is in BCNF (Boyce-Codd Normal Form)**

A table complies with BCNF if it is in 3NF and for every functional dependency there should be the super key of the table.

{First Name, Last Name, Address, Zip Code, Phone Number, Email Address, City ID} --> Customer ID – All the columns are in the same Table and Customer ID is the Key column. Therefore, there is one super key for the functional dependency

{City, State ID} --> City ID - All the columns are in the same Table and City ID is the Key column. Therefore, there is one super key for the functional dependency

{State} --> State ID - All the columns are in the same Table and State ID is the Key column. Therefore, there is one super key for the functional dependency

{Product Name, Price Per Unit} --> Product ID - All the columns are in the same Table and Product ID is the Key column. Therefore, there is one super key for the functional dependency

{Seller ID} --> Product ID - All the columns are in the same Table and Product ID is the Key column. Therefore, there is one super key for the functional dependency

{Brand Name, Seller Contact Name, Seller Address, Seller Zip Code, Seller Phone, Seller Email, City ID} --> Seller ID - All the columns are in the same Table and Seller ID is the Key column. Therefore, there is one super key for the functional dependency

{Order Date, Customer ID, Product ID, Transaction ID, Units in Order, Order Total} --> Order ID - All the columns are in the same Table and Order ID is the Key column. Therefore, there is one super key for the functional dependency

{Payment Type ID} --> Transaction ID - All the columns are in the same Table and Transaction ID is the Key column. Therefore, there is one super key for the functional dependency

{Payment Type} --> Payment Type ID - All the columns are in the same Table and Payment Type ID is the Key column. Therefore, there is one super key for the functional dependency

d. **If your schema violates BCNF, bring it to BCNF by decomposing it**
Schema is already in BCNF

e. **Update your ER diagram with the latest schema**

Since the schema is already in BCNF, the ER diagram is not updated

5. **Create your database in MySQL using the latest version of your schema**

E-Commerce Database has been created with the below SQL command

CREATE SCHEMA IF NOT EXISTS `ecommerce_database` DEFAULT CHARACTER SET utf8

6. **Import the data into your database**

   1. If there are errors while importing, document these errors in your report and mention how you dealt with them

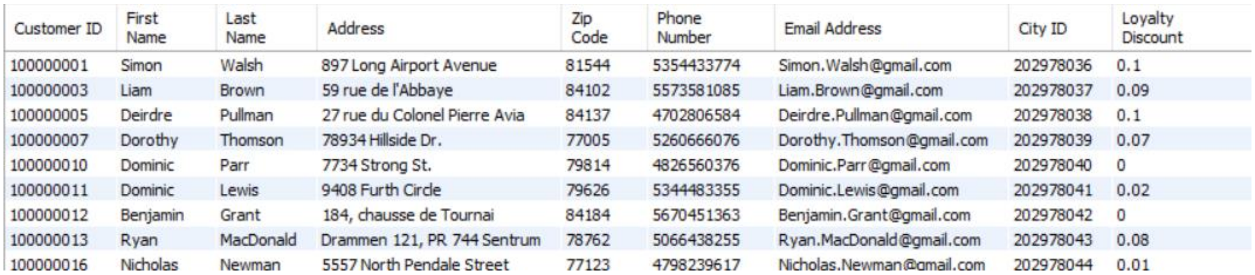
   Errors Encountered while importing the Tables

   - Five values in the column Price Per Units had redundant values such as “N/A”. Due to this, the five rows were getting rejected, while loading the data into the database. The reason being, the datatype of the column was set to decimal, but we were trying to load character values. We solved this error by populating the correct value for the product based on the previous records.
   - While loading the Order Table, the column Order Date was getting rejected as it was in ‘YYYY/MM/DD” format in the Dataset. The error was resolved by changing the Date format to “YYYY-MM-DD"

Step 5: Data Cleaning and Database Testing

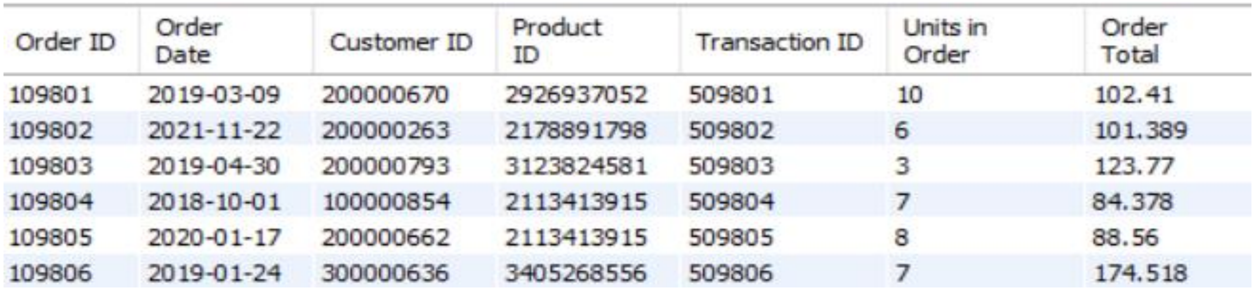
- Data has now been loaded into the database - “ecommerce_database”
- Each table in the database has been validated. The data has been correctly populated into the corresponding columns.

Customer Table



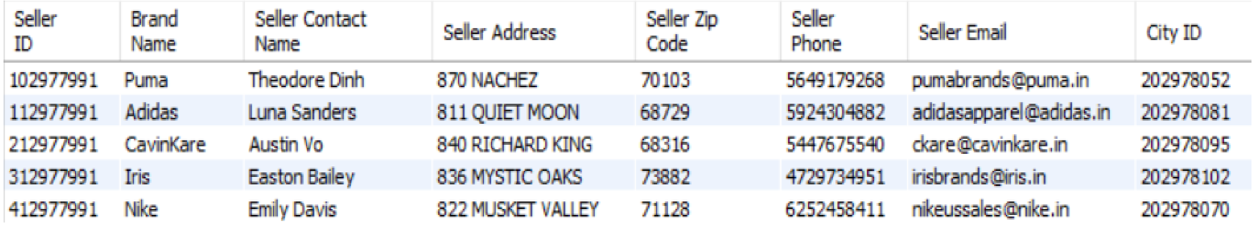
Customer ID	First Name	Last Name	Address	Zip Code	Phone Number	Email Address	City ID	Loyalty Discount
100000001	Simon	Walsh	897 Long Airport Avenue	81544	5354433774	Simon.Walsh@gmail.com	202978036	0.1
100000003	Liam	Brown	59 rue de l'Abbaye	84102	5573581085	Liam.Brown@gmail.com	202978037	0.09
100000005	Deirdre	Pullman	27 rue du Colonel Pierre Avia	84137	4702806584	Deirdre.Pullman@gmail.com	202978038	0.1
100000007	Dorothy	Thomson	78934 Hillside Dr.	77005	5260666076	Dorothy.Thomson@gmail.com	202978039	0.07
100000010	Dominic	Parr	7734 Strong St.	79814	4826560376	Dominic.Parr@gmail.com	202978040	0
100000011	Dominic	Lewis	9408 Furth Circle	79626	5344483355	Dominic.Lewis@gmail.com	202978041	0.02
100000012	Benjamin	Grant	184, chausse de Tournai	84184	5670451363	Benjamin.Grant@gmail.com	202978042	0
100000013	Ryan	MacDonald	Drammen 121, PR 744 Sentrum	78762	5066438255	Ryan.MacDonald@gmail.com	202978043	0.08
100000016	Nicholas	Newman	5557 North Pendale Street	77123	4798239617	Nicholas.Newman@gmail.com	202978044	0.01

Order Table



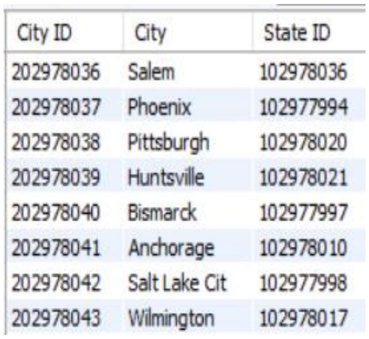
Order ID	Order Date	Customer ID	Product ID	Transaction ID	Units in Order	Order Total
109801	2019-03-09	200000670	2926937052	509801	10	102.41
109802	2021-11-22	200000263	2178891798	509802	6	101.389
109803	2019-04-30	200000793	3123824581	509803	3	123.77
109804	2018-10-01	100000854	2113413915	509804	7	84.378
109805	2020-01-17	200000662	2113413915	509805	8	88.56
109806	2019-01-24	300000636	3405268556	509806	7	174.518

Seller Table



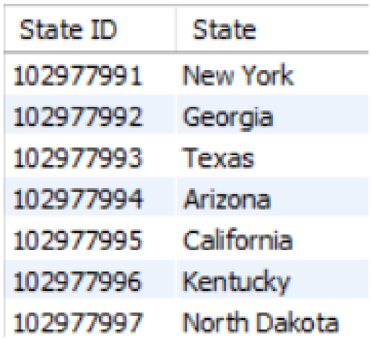
Seller ID	Brand Name	Seller Contact Name	Seller Address	Seller Zip Code	Seller Phone	Seller Email	City ID
102977991	Puma	Theodore Dinh	870 NACHEZ	70103	5649179268	pumabrands@puma.in	202978052
112977991	Adidas	Luna Sanders	811 QUIET MOON	68729	5924304882	adidasapparel@adidas.in	202978081
212977991	CavinKare	Austin Vo	840 RICHARD KING	68316	5447675540	ckare@cavinkare.in	202978095
312977991	Iris	Easton Bailey	836 MYSTIC OAKS	73882	4729734951	irisbrands@iris.in	202978102
412977991	Nike	Emily Davis	822 MUSKET VALLEY	71128	6252458411	nikeussales@nike.in	202978070

City Table



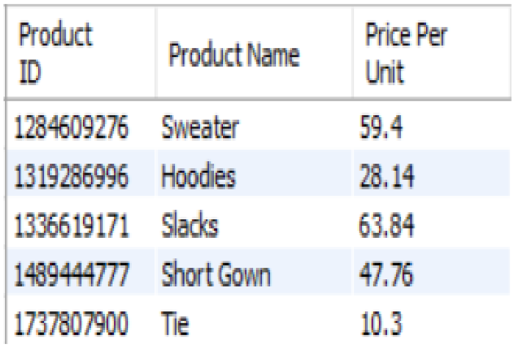
City ID	City	State ID
202978036	Salem	102978036
202978037	Phoenix	102977994
202978038	Pittsburgh	102978020
202978039	Huntsville	102978021
202978040	Bismarck	102977997
202978041	Anchorage	102978010
202978042	Salt Lake Cit	102977998
202978043	Wilmington	102978017

State Table



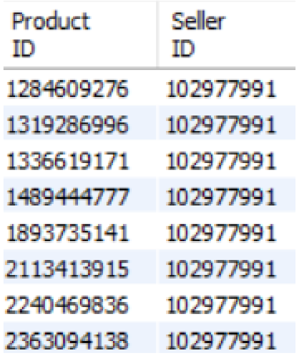
State ID	State
102977991	New York
102977992	Georgia
102977993	Texas
102977994	Arizona
102977995	California
102977996	Kentucky
102977997	North Dakota

Product Table



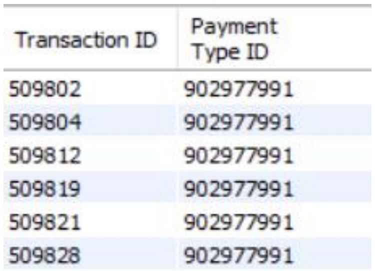
Product ID	Product Name	Price Per Unit
1284609276	Sweater	59.4
1319286996	Hoodies	28.14
1336619171	Slacks	63.84
1489444777	Short Gown	47.76
1737807900	Tie	10.3

Seller Table



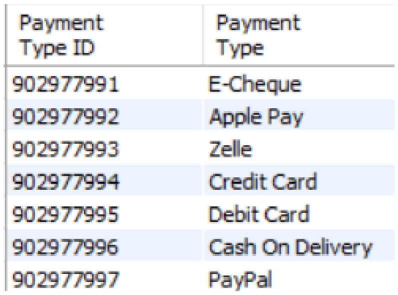
Product ID	Seller ID
1284609276	102977991
1319286996	102977991
1336619171	102977991
1489444777	102977991
1893735141	102977991
2113413915	102977991
2240469836	102977991
2363094138	102977991

Payment Table



Transaction ID	Payment Type ID
509802	902977991
509804	902977991
509812	902977991
509819	902977991
509821	902977991
509828	902977991

Payment Type Table



Payment Type ID	Payment Type
902977991	E-Cheque
902977992	Apple Pay
902977993	Zelle
902977994	Credit Card
902977995	Debit Card
902977996	Cash On Delivery
902977997	PayPal

Statistics:

- The Statistics for each of the columns have been captured in Step 3

Missing Values:

- There are no missing values identified in any of the Tables. This has been verified using the ISNULL () function in SQL. The SQL used is appended in the Appendix.

Data Errors

- There are no Data Errors or any values that do not seem to be valid

Datatype Validation:

- All the values in each of the columns are of the same datatype. The same has been validated using the REGEXP function in SQL. The function returns 1 if it contains numerical or floating-point values and returns 0 if contains characters. The SQL used is appended in the Appendix.

Constraints

- Each of the constraints defined are working as expected and are fetching the desired output

Example1:

In this Example, the Order Table, Customer Table, City Table, State Table, Payment Table and Payment Type Table are joined to fetch the record for the corresponding Order ID. The SQL used is appended in the Appendix.

Example 2:

In this Example, the Product Table, Seller Table, Seller Details Table, City Table and State Table are joined to fetch the seller details for the corresponding Product. The SQL used is appended in the Appendix.

Example 3:

In this Example, Order Table and Product Table are joined to fetch the Product Details for the corresponding Order ID. The SQL used is appended in the Appendix.

Conclusion

The ecommerce dataset gives us a lot of useful information that can be transformed into useful business ideas. Using SQL was useful in querying the data to examine areas where improvements can be made to the business. The data did not need any cleaning and it did not have any null values. We carried out schema normalization to the data and it removed the redundancy issues that this dataset would have otherwise faced.

Appendix

1. **Missing Values SQL Query**

```
SELECT *

FROM `CUSTOMER`

WHERE `Customer ID` IS NULL

        or `FIRST NAME` IS NULL

   or `LAST NAME` IS NULL

   or `ADDRESS` IS NULL

   or `ZIP CODE` IS NULL

   or `PHONE NUMBER` IS NULL

   or `EMAIL ADDRESS` IS NULL

   or `CITY ID` IS NULL

   or `LOYALTY DISCOUNT` IS NULL.


SELECT *

FROM `SELLER`

WHERE `PRODUCT ID` IS NULL

        or `SELLER ID` IS NULL.


SELECT *

FROM `PRODUCT`

WHERE `PRODUCT ID` IS NULL

        or `PRODUCT NAME` IS NULL

   or `PRICE PER UNIT` IS NULL.


SELECT *

FROM `CITY`
```

```
WHERE `CITY ID` IS NULL
         or `CITY` IS NULL
    or `STATE ID` IS NULL.

SELECT *
FROM `STATE`
WHERE `STATE ID` IS NULL
         or `STATE` IS NULL.

SELECT *
FROM `SELLER_DETAILS`
WHERE `SELLER ID` IS NULL
         or `BRAND NAME` IS NULL
    or `SELLER CONTACT NAME` IS NULL
    or `SELLER ADDRESS` IS NULL
    or `SELLER ZIP CODE` IS NULL
    or `SELLER PHONE` IS NULL
    or `SELLER EMAIL` IS NULL
    or `CITY ID` IS NULL.

SELECT *
FROM `ORDER`
WHERE `ORDER ID` IS NULL
         or `ORDER DATE` IS NULL
    or `CUSTOMER ID` IS NULL
    or `PRODUCT ID` IS NULL
    or `TRANSACTION ID` IS NULL
```

```
       or `UNITS IN ORDER` IS NULL

       or `ORDER TOTAL` IS NULL.


SELECT *

FROM `PAYMENT_TYPE`

WHERE `TRANSACTION ID` IS NULL

                or `PAYMENT TYPE ID` IS NULL.


SELECT *

FROM `PAYMENT`

WHERE `TRANSACTION ID` IS NULL

                or `PAYMENT TYPE ID` IS NULL.
```

2. **Datatype Validation SQL Query**

```
  SELECT *
  FROM `CUSTOMER`
  WHERE `CUSTOMER ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
             AND `FIRST NAME` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'
    AND `LAST NAME` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'
    AND `ADDRESS` REGEXP '^ [0-9]+\\.?[0-9]*$' = '1'
    AND `ZIP CODE` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
    AND `PHONE NUMBER` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
    AND `EMAIL ADDRESS` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'
    AND `CITY ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
    AND `LOYALTY DISCOUNT` REGEXP '^[0-9]+\\.?[0-9]*$' = '0';

SELECT *
FROM `CITY`
WHERE `CITY ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
  AND `CITY` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'
  AND `STATE ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0' ;

SELECT *
FROM `STATE`
```

```
WHERE `STATE ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
  AND `STATE` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'  ;

SELECT *
FROM `SELLER_DETAILS`
WHERE `SELLER ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
            AND `BRAND NAME` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'
     AND `SELLER CONTACT NAME` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'
     AND `SELLER ADDRESS` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'
     AND `SELLER ZIP CODE` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
     AND `SELLER PHONE` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
     AND `SELLER EMAIL` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'
     AND `CITY ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0';

SELECT *
FROM `SELLER`
WHERE `PRODUCT ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
  AND `SELLER ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0' ;

SELECT *
FROM `PRODUCT`
WHERE `PRODUCT ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
          AND `PRODUCT NAME` REGEXP '^[0-9]+\\.?[0-9]*$' = '1'
          AND `PRICE PER UNIT` REGEXP '^[0-9]+\\.?[0-9]*$' = '0';

SELECT *
FROM `PAYMENT_TYPE`
WHERE `TRANSACTION ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
  AND `PAYMENT TYPE ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0' ;

SELECT *
FROM `PAYMENT`
WHERE `TRANSACTION ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
          AND `PAYMENT TYPE ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'  ;

SELECT *
FROM `ORDER`
WHERE `ORDER ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
            AND `ORDER DATE` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
     AND `CUSTOMER ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
     AND `PRODUCT ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
```

```
    AND `TRANSACTION ID` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
    AND `UNITS IN ORDER` REGEXP '^[0-9]+\\.?[0-9]*$' = '0'
    AND `ORDER TOTAL` REGEXP '^[0-9]+\\.?[0-9]*$' = '0';
```

3. Constraints Validation SQL Query

```
SELECT o.`Order ID`, o.`Order Date`, c.`Customer ID`, c.`First Name`, c.`Last Name`,
c.`Address`,
         c.`Zip Code`,c.`Phone Number`, c.`Email Address`,ci.`City`, st.`State`,
c.`Loyalty Discount`, o.`Units in Order`, o.`Order Total`,
    pay.`Transaction ID`, pt.`Payment Type` AS `Payment Mode`
FROM `CUSTOMER` AS c
JOIN `ORDER` AS o
ON o.`Customer ID` = c.`Customer ID`
JOIN `CITY` AS ci
ON ci.`City ID` = c.`City ID`
JOIN `STATE` AS st
ON st.`State ID` = ci.`State ID`
JOIN `PAYMENT`AS pay
ON o.`Transaction ID` = pay.`Transaction ID`
JOIN `PAYMENT_TYPE` AS pt
ON pay.`Payment Type ID` = pt.`Payment Type ID`
WHERE o.`Order ID` = '109805';

SELECT o.`Order ID`,p.`Product ID`, p.`Product Name`, p.`Price Per Unit`
FROM  `ORDER` AS o
JOIN `PRODUCT` AS p
ON o.`Product ID` = p.`Product ID`
WHERE o.`Order ID` = '109805';

SELECT DISTINCT sd.`Seller ID`, sd.`Brand Name`, sd.`Seller Contact Name`,
    sd.`Seller Address`, sd.`Seller Zip Code`, sd.`Seller Phone`, sd.`Seller Email`,
ci.`City`, st.`State`
FROM `SELLER` AS s
JOIN `PRODUCT` AS p
ON p.`Product ID` = s.`Product ID`
JOIN `SELLER_DETAILS` AS sd
ON s.`Seller ID` = sd.`Seller ID`
JOIN `CITY` AS ci
ON ci.`City ID` = sd.`City ID`
JOIN `STATE` AS st
```

```
ON st.`State ID` = ci.`State ID`
WHERE sd.`Brand Name` = 'CavinKare';
```