ST207 Group Project Report:

An Insight into an E-Commerce Database

Candidate Numbers: [redacted]

Table of contents

1.0 Introduction

Our database application will be centred around an e-commerce marketplace, particularly focused on customer reviews on purchases. This will allow us to establish an application that can be used for multiple real-life purposes. To accomplish this we will be using a relational SQL database and creating queries with aggregation functions. The relationships between tables for Customers, Products, Reviews and Orders should allow for powerful queries that can highlight the most interesting and notable relationships in the dataset.

One purpose will be on the consumer side, providing customers with insights into which products have good/bad reviews, which are highly/poorly rated, etc. This will be done via queries retrieving the highest/lowest ratings for products grouped by category, brand etc.

The other application will be to do with market insights for companies on the supply side. Insights include ideas such as which products sell better at different times in the year and which brands have the highest average reviews.

2.0 Database modelling

Entities, relationships and usage operations
To build a database that gives sellers and buyers insights into products, we need to create tables to organise the data. At first glance, these tables have to be created: Customers, Products, Orders, and Reviews.

For the Customers table, we can use UserId as the primary key. The table can also have an attribute of Location which we will be using in queries later on.

The Products table will consist of ProductId as its primary key. It is worth mentioning that each ProductId will represent a particular product. There can be more than one order of this particular product on Amazon, but they are all identified using one ProductId. The reason for this is that we are not focusing on an e-commerce operations database, but rather on the application we have described earlier. Following this, it will also have Category, Brand, SubCategory, and ProdDescription as its attributes. These attributes essentially give us information or describe the products themselves.

The Reviews table will use UniqID as its primary key to identify each review. UserId, ProductId, and BillingId will be foreign keys in this table. Under this, we will have RTitle, RMonth, RRating, RContent, and HelpfulR as attributes of the Review entity. We will also use BillingId as a foreign key in this table to identify which review links to the particular purchase. Assuming that a customer only leaves one review for a purchase, we can state that this will be a one to one relationship as a particular review can only be linked to one purchase. Reviews also have a 1 to Many relationship with Products as a particular product can have reviews from several different purchases whereas one review can only be for one specific product.

For the last table, Orders, BillingId will be the primary key while UserId and ProductId are the foreign keys. We need both of these to be foreign keys because we need to link the specific order to the customer, and the product they have bought. Apart from that, we will also have ProdDescription as an attribute. We can assume that one customer can buy multiple products, which means they can have more than one order. On the other hand, a particular purchase of a product (and the BillingId corresponding to that purchase) can only be linked to one customer. This results in Customer to Orders being a one to many relationship. Products have a 1 to Many Relation with Orders as each purchase is linked to one product, but a type of product can be purchased several times.

As usual, all primary keys for each table have to be unique and can't be empty. The restrictions apply to all the keys UserId, BillingId, UniqID, and ProductId.

Restrictions

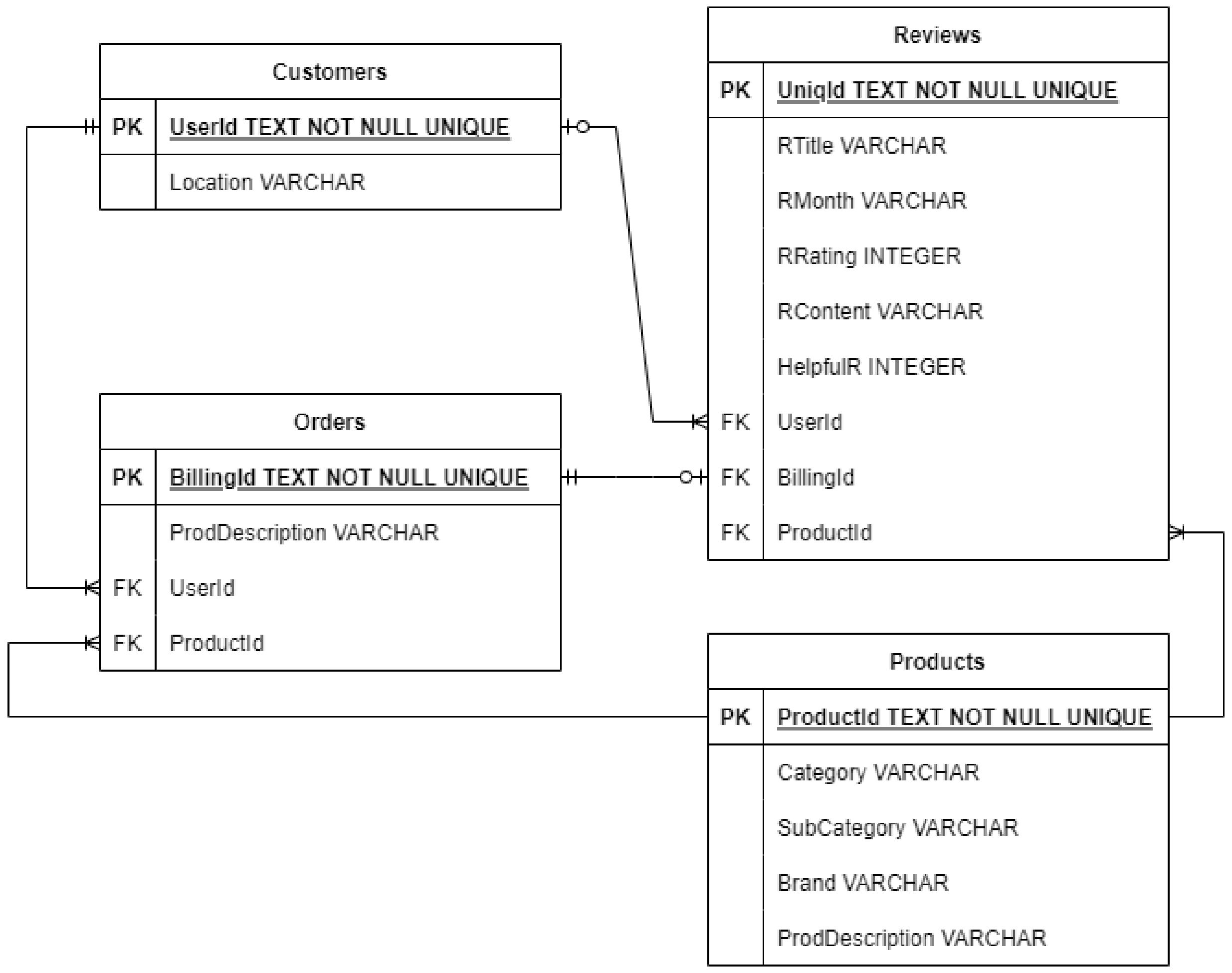
We have included a few restrictions to our database model to better mimic a real-life database for customer reviews. Some of these restrictions are built into our database through the use of triggers and database creation commands.

Firstly, we have simple primary and foreign key restrictions. By definition, primary keys must be unique and non-empty. While creating our database, we used the PRIMARY KEY command to ensure that these restrictions stay in place. Furthermore, we also have to ensure that each foreign key value in our database must point to a unique primary key value in a different table. We have used a trigger (see Trigger 3) to enforce this restriction in our database.

We also take into account restrictions for values that are entered into our database. For instance, we have included a restriction that the Review Ratings attribute in our database can only take a value between 1 and 5. This restriction already existed in our original dataset and we have created a trigger (see Trigger 1) to ensure that any future entries will abide by it.

ER model

Most of the model has been explained above. But, it is worth noting a few more things about the attributes, entities and relationships in our database. For starters, the only partial relationship in our database is between Orders and Reviews, and Customers and Reviews. This is because a customer who has ordered a product does not necessarily have to leave a review. We also have a composite attribute in Category which can be divided into Subcategories. None of the entities in our model are weak entities.



3.0 Dataset

The data is mostly consistent, so we have only identified a few things to change. First, we changed the names of most of the attributes for them to identify with more meaningful names.

Following the name changes, the main modification we made was adding new columns. The original dataset did not have the two columns Location and ProductId. We have added Location to give us more room to interpret our queries later on for market insights. We have also included ProductId so that it is clear what reviews relate to what product. All the attributes mentioned above are synthetic records that we have decided to add. Another major change we did was manipulating the RMonth column to represent months only. We decided that the review's month was the only relevant factor when it came to finding out at which times of the year products were in high demand. When it came to the HelpfulR column, we found that it would be easier if all of them were in number format, so we have changed this from the initial mix of text and numbers format. We also changed the Billing ID column to be unique for each review in line with our 1 to 1 assumptions for this relationship.

We also removed a couple of columns that we did not think were relevant to our application for simplicity. Since we are not panning our focus to the operations of Amazon, we have removed columns like manufacturer response, crawl time stamp etc. The remaining changes to the data were removing records with missing data, or unverified purchases as these would not be useful for our application.

4.0 Database creation

Database technology

We used SQLite with DB Browser as we thought it to be very useful considering our application. It allowed for a simple connection to the database and good functionality. This is especially important as part of our application is focused on giving customers insights into products. Designing a relational database with SQL as the query language we believe allows for the greatest simplicity for end-users. The way DB Browser compacts the database files is another reason we chose to use it. With our dataset coming from Amazon there is potential for our database to handle vast amounts of data, so we must manage the size of the database files, so they do not grow too quickly and hamper performance. The suite of import and export tools available with DB Browser was also appealing to us. The other part of our application was to offer insights on the producer side. Considering these large firms will have analysts on payroll, they will likely be running queries at a far higher level than the customers and may want to use different analytical applications, so the ease of exporting databases as SQL dump files will be helpful.

Structure

Since we had to insert data from our relatively large dataset, our database creation process differed slightly. The tables were first created with the primary key restrictions directly in place. Following that, we had to make a few temporary tables before inserting the data. These temporary tables were in place to make sure that the data we were inserting followed the rules of our database.

Triggers

Based on the restrictions mentioned earlier, and more, we have put some triggers into our database and tested them. When it came to the primary key restrictions, most conditions were already set while creating the table with the UNIQUE and NOT NULL commands, so there aren't any triggers for these. But we have tested to check in case.

Trigger 1 focuses on customer ratings. Ratings can only take a value between 1 and 5, any numbers out of this range should be rejected. Trigger 2 only allows each review to correlate to one order. This trigger is in place because it would be an unreliable system if one customer could write multiple reviews on a single order.

Trigger 3 is in place for the foreign key violations, and to maintain referential identity. Since we have several foreign keys in our database, these triggers are necessary whenever we're inserting new records. For instance, in the Products table, the BillingId foreign key must point to somewhere in the Orders table. However, there are instances where a primary key doesn't need to have a foreign key elsewhere. An example of this would be with the Orders and Reviews table. A customer might have placed an order but chose not to leave a review, therefore, the foreign key BillingId in Reviews can be null. Before inserting the data, we can notice that we need to insert the data in order of Customers, Products, Orders, then Reviews, otherwise, trigger 3 will come into play because the foreign keys in the later tables do not match a primary key.

In our database, we are not allowing for any deletion of records because keeping all records would give us the historical breakdown of products and help us with our queries. Therefore, the last trigger, trigger 4, is in place to prevent any records from being deleted.

As explained earlier, when inserting our dataset into the database, we did not encounter any errors because we had cleaned up the data through the temporary tables before inserting them into the main tables.

Indexes

We have created some indexes for our database that we believe would be useful for database users by speeding up queries that a database user might be interested in.

Index 1 is used to search through the Helpful Reviews column in the Reviews table. This index helps the system search for reviews that customers find helpful while also helping the business gain an understanding of the general customer sentiment towards specific products.

Index 2 is used to search through the Locations column in the Customers table. This can be particularly useful for database users if they would like to analyse how Amazon is performing in certain areas of the world. Are certain products more popular in a certain country? Is business particularly performing well in a different country? These are all questions that can be answered by making queries using the location index.

Index 3 is a unique index that is made using the Billing ID column of the Orders table. This can be used to search through several orders to find information that may be relevant to database users. An example would be searching for particular products that are common in a large number of orders.

Index 4 is a composite index made on the category and subcategory columns in the Product table. The primary use of this index is searching across different categories and analysing which products in certain categories are performing the best in terms of both orders and positive reviews.

Views

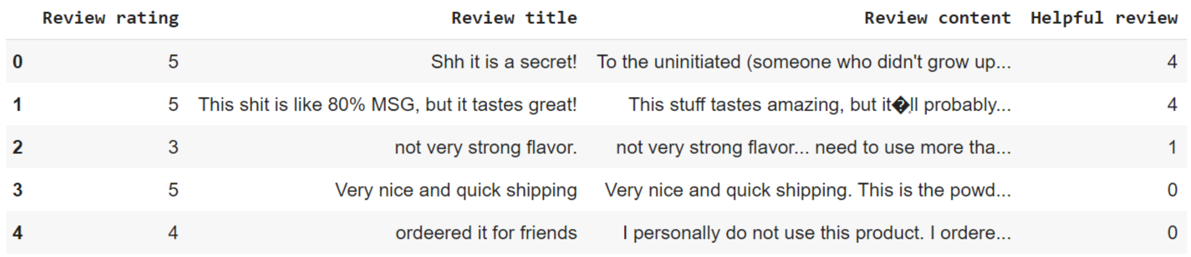
View 1 contains review content, as well as the location of the customers who other users have determined, have left a helpful review. This could be useful to our application as it helps to filter out bogus reviews (an increasingly common problem on Amazon) therefore allowing for more accurate analysis.

View 2 has review and product information on products that had a review of 4 or 5 out of 5. This makes queries concerning only the highest rated products and identifying anomalously high reviews, easier.

5.0 Queries

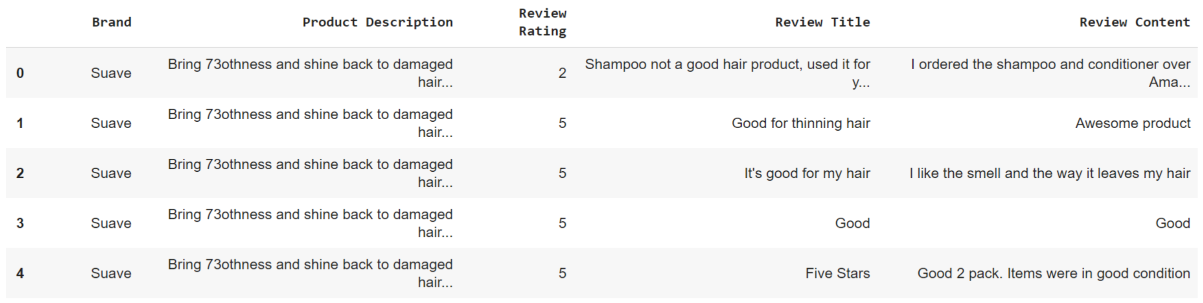
(The query outputs in this report are not complete because some results returned up to 80 rows, refer to the notebook for the complete results)

Query 1: Let’s say that we have a customer who is looking at a particular product with the ProductId of ‘153’. They are unsure whether they want to buy this product so they send a query to retrieve all the reviews of the product. The query will rank reviews by how helpful they are using HelpfulR.



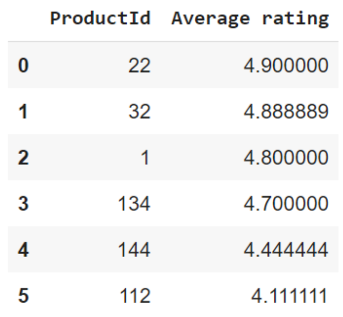
	Review rating	Review title	Review content	Helpful review
0	5	Shh it is a secret!	To the uninitiated (someone who didn't grow up...	4
1	5	This shit is like 80% MSG, but it tastes great!	This stuff tastes amazing, but it�ll probably...	4
2	3	not very strong flavor.	not very strong flavor... need to use more tha...	1
3	5	Very nice and quick shipping	Very nice and quick shipping. This is the powd...	0
4	4	ordeered it for friends	I personally do not use this product. I ordere...	0

Query 2: Similar to query 1, the idea behind this query is to act as a guide for a customer. If a customer is unsure what product they would like (e.g. for a gift), they can search through products that have had reviews with a generally positive sentiment



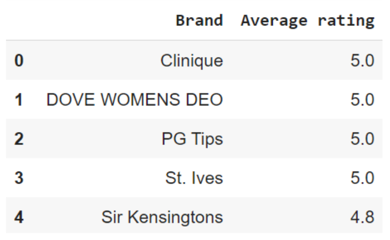
	Brand	Product Description	Review Rating	Review Title	Review Content
0	Suave	Bring 73othness and shine back to damaged hair...	2	Shampoo not a good hair product, used it for y...	I ordered the shampoo and conditioner over Ama...
1	Suave	Bring 73othness and shine back to damaged hair...	5	Good for thinning hair	Awesome product
2	Suave	Bring 73othness and shine back to damaged hair...	5	It's good for my hair	I like the smell and the way it leaves my hair
3	Suave	Bring 73othness and shine back to damaged hair...	5	Good	Good
4	Suave	Bring 73othness and shine back to damaged hair...	5	Five Stars	Good 2 pack. Items were in good condition

Query 3: This query is an extension of the previous query where customers can view the most highly rated products from amongst a particular subcategory - in this case, Hair Care.



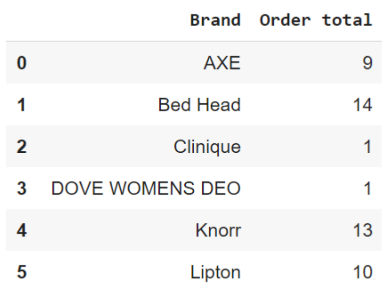
	ProductId	Average rating
0	22	4.900000
1	32	4.888889
2	1	4.800000
3	134	4.700000
4	144	4.444444
5	112	4.111111

Query 4: This query retrieves the average review for each brand across the dataset. This has an application for both customers and producers, as customers can see the average quality of a product they might be about to buy, and producers can see which of their competitors is most highly favoured at any given time



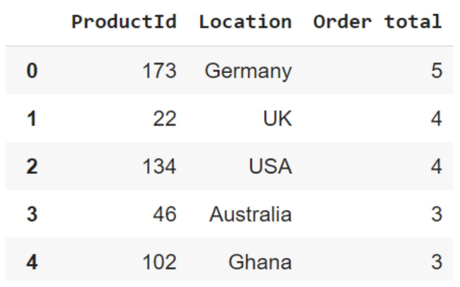
	Brand	Average rating
0	Clinique	5.0
1	DOVE WOMENS DEO	5.0
2	PG Tips	5.0
3	St. Ives	5.0
4	Sir Kensingtons	4.8

Query 5: This query retrieves useful information for companies. It counts the number of orders for each brand. It will give companies a good idea of who has the largest market share



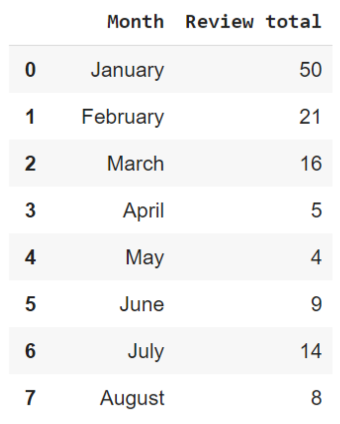
	Brand	Order total
0	AXE	9
1	Bed Head	14
2	Clinique	1
3	DOVE WOMENS DEO	1
4	Knorr	13
5	Lipton	10

Query 6: This query counts the number of orders for each product grouped by country. It will be useful as it gives insights to producers on where their products are selling the best.



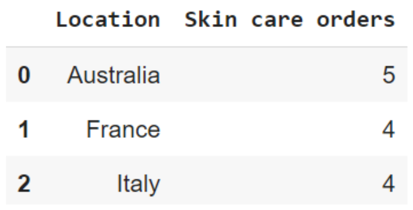
	ProductId	Location	Order total
0	173	Germany	5
1	22	UK	4
2	134	USA	4
3	46	Australia	3
4	102	Ghana	3

Query 7: This query calculates the number of reviews by month. This is a producer application as this query gives us an insight into when customers buy products.



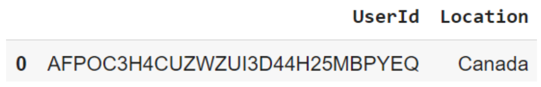
	Month	Review total
0	January	50
1	February	21
2	March	16
3	April	5
4	May	4
5	June	9
6	July	14
7	August	8

Query 8: This gives a viewpoint from a business standpoint showing the top 3 countries where Skin Care products are purchased on Amazon.



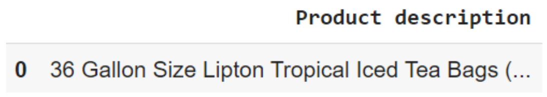
	Location	Skin care orders
0	Australia	5
1	France	4
2	Italy	4

Query 9: This shows an update query where we update the location of a particular customer who may have moved countries. We then use a query to showcase the new location.



	UserId	Location
0	AFPOC3H4CUZWZUI3D44H25MBPYEQ	Canada

Query 10: Here we have an update query that is designed to work for companies on Amazon. If producers want to make changes to their products, it should be reflected in the database with this query.



	Product description
0	36 Gallon Size Lipton Tropical Iced Tea Bags (...

6.0 Conclusion

Practicalities of our results

We made this database to serve both customers and companies on Amazon. Queries 1-4 are focused on customer applications. As we described in the query applications, customers will be able to use the database on a day-to-day basis, so it has a practical aspect from the customer's point of view. Queries 4-8 are focused on company applications. Similarly, we can see that the queries would come in handy for companies who are curious about their competitor's activities or companies who are planning to expand to new locations. However, we do think that there could be a wider potential for our database to serve these companies better.

Improvements

While we are very proud of the design of our database there are some areas of the project we could have expanded on when thinking of the application. For example, it could have been interesting to explore some different database tools, maybe some with more advanced visualization capabilities to help display the insights from the data for companies. We also had to drop a few columns where the majority of data was missing or inconsistent, we could have explored these areas in more depth. As we did with the locations column we also could have created more synthetic entities which would have increased the number of relations and overall database complexity. Overall, we think the way our database application had purposes for both the consumer and producer side may have stopped us from creating queries of the highest complexity as it also had to be accessible for consumers.