E-DataSync: Distributed E-Commerce Inventory Management

Team Members:

Christopher Apton (apton@usc.edu)

Daniel Jang (hyundong@usc.edu)

Shahzaib Saqib Warraich (warraich@usc.edu)

Implementation Questions:

- **Which database are you using?**
  MySQL with two distributed relational databases.
- **What is the approach for distributed scaling of data?**
  Hashing on the product name (primary key of products table) which can give a unique primary key that distributes the data to the correct database amongst the two distributed relational databases.
- **Which application did you choose to implement?**
  An e-commerce distributed database inventory management system that will basically be used by database administrators and end-users for interaction using CLI (User Interface) and dropdowns/text fields (Web Application).

Planned Implementation:

An E-commerce inventory management system that allows database managers and end users to communicate and interact with the distributed databases. We will have **four** relational tables in the distributed databases including *Products*, *Suppliers*, *Orders, and Order_Detail.* We will be using the **product names** in the *Products* table to split (based on hash values) the data into multiple distributed databases. The purpose of this web application is to manage inventory and its audience is mainly targeted for inventory managers who are managing large e-commerce databases.

Relations:

Table #1: *Products* (real data), link: Kaggle

Columns: *product_name, category, sub_category, brand, sale_price, market_price, type, rating, description*

Primary key: product_name

Foreign key: brand, **reference** Suppliers (brand_name)

Table #2: *Suppliers* (synthetically generated data)

Columns: *brand_name, address, description, founding_year, number_of_products*

Primary key: brand_name

Table #3: *Orders* (synthetically generated data)

Columns: *order_id, date, customer_id, total_price*

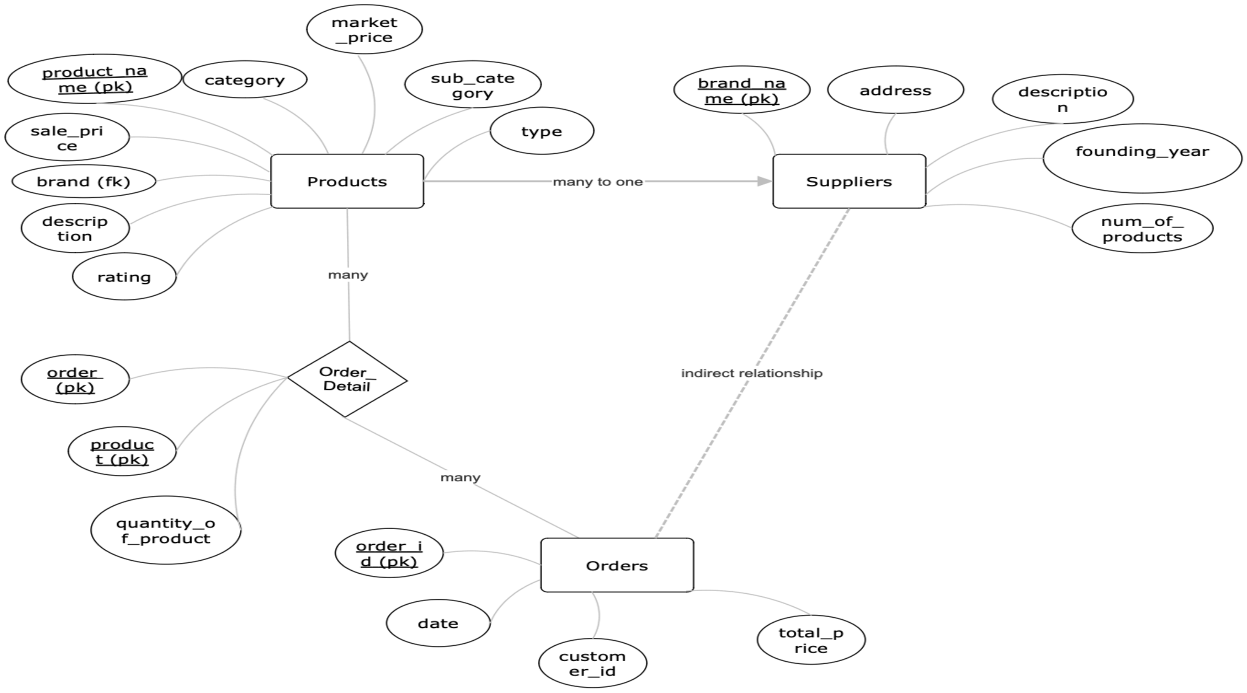
Primary key: order_id

Table #4: *Order_Detail* (synthetically generated data)

Columns: *order, product, quantity_of_product*

Primary keys: order, product → **composite**, **references** Orders (order_id), Products(product_name)

ER:



User Interface:

- The Database Manager selects the relevant database table (*products*, *suppliers*, or *orders*) they want to interact with.

- Command Line Interface (CLI) which allows DB Managers to either add, modify, or delete record(s) from the chosen database table.
- An additional option to get record(s) from the chosen database table.

Web Application:

- Create a homepage that allows users to select the relevant database table (*products*, *suppliers*, or *orders*) they want to interact with .
- Based on the selected database table, direct the user to another page that has four main buttons: *insert*, *modify*, *delete*, and *search*.
- Each of the four buttons will direct to a new page and prompt the user to interact with the interface.
    - Text fields for users to enter data.
    - Additional small buttons will be available: "*Go back*", "*Save*" and "*Empty*".
        - First one will go back to the homepage page without saving anything.
        - Second one will save all the entered records and send the request to the database table.
        - Third one will empty all fields if the user has either saved or entered some data fields.

Functionality of buttons:

- *Insert*
    - 1xN blank data fields. All of them are mandatory. Additional small buttons will be available: "*Go back*", "*Save*" and "*Empty*".
- *Modify*
    - The *unique product name* will be entered based on which the current record will be retrieved and displayed from the relevant database table. Additional small buttons will be available: "Go back", "Save" and "Empty".

○ Afterwards, 1xN blank data fields will be displayed and none of them will be mandatory. Additional small buttons will be available: “*Go back*”, “*Save*” and “*Empty*”.

- *Delete*

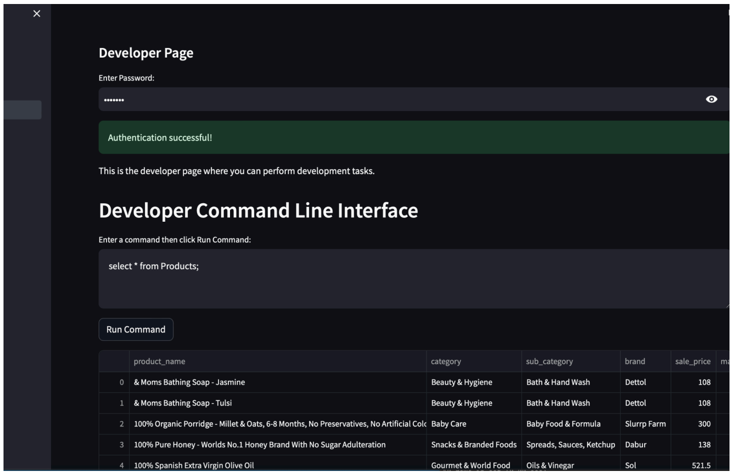
○ The *unique product name* will be entered and the corresponding record will be deleted. Additional small buttons will be available: “*Go back*”, “*Save*” and “*Empty*”.

- *Search*

○ The *unique product name* will be entered and the corresponding record will be retrieved from the databases. Additional small buttons will be available: “*Go back*”, “*Save*” and “*Empty*”.

Status of the project:

- Products table has been cleaned and distributed based on the product name to amongst the two distributed databases
- Suppliers table has been synthetically generated
- Hash function has been developed
- All primary key, foreign key, and data type constraints have been imposed
- Databases have been populated
- User Interface for the Database Managers has been developed
- Backend for querying the database has been developed



Developer Page

Enter Password:

•••••••

Authentication successful!

This is the developer page where you can perform development tasks.

Developer Command Line Interface

Enter a command then click Run Command:

```
select * from Products;
```

	product_name	category	sub_category	brand	sale_price	ma
0	& Moms Bathing Soap - Jasmine	Beauty & Hygiene	Bath & Hand Wash	Dettol	108	
1	& Moms Bathing Soap - Tulsi	Beauty & Hygiene	Bath & Hand Wash	Dettol	108	
2	100% Organic Porridge - Millet & Oats, 6-8 Months, No Preservatives, No Artificial Colc	Baby Care	Baby Food & Formula	Slurrp Farm	300	
3	100% Pure Honey - Worlds No.1 Honey Brand With No Sugar Adulteration	Snacks & Branded Foods	Spreads, Sauces, Ketchup	Dabur	138	
4	100% Spanish Extra Virgin Olive Oil	Gourmet & World Food	Oils & Vinegar	Sol	521.5	

Challenges:

One challenge we had was with port-forwarding. It was pretty difficult to port-forward the 2 databases from MySQL due to many different restrictions. First, the issue was handled by port forwarding on the correct port. Also, we needed to edit the configuration file for MySQL to add the port to it. Next, there needed to be users created specifically for people to access the databases. Afterward, we needed to create the two databases along with giving the user all necessary permissions to manipulate the databases.

Timeline:

Event	Due Date
In-person group meeting. Pitch in ideas. Discuss how we should tackle the project and work on proposal as a team	~~2/2~~
Clean dataset and generate synthetic data for our relational tables. Set up the distributed databases with constraints. Develop and test front and backend for CLI User Interface.	~~3/1~~
Make sure both of the deliverables are working to demonstrate in class. Show that we are able to use the main features on our web applications. Everything must be fully implemented at this time.	4/17
Finish up with the last details. Add up any additional features that we might be interested in. Make sure everything is working as intended and create a final report.	5/3