INST 733

Database Design

Project Diary and Report:

"*The Coffee Bar*" located at the stamp student union at the University of Maryland, College Park

Submitted By:

Adarsh Srinivas (adarshs1@umd.edu)

Srikanth Jaikumar (sri14@umd.edu)

Introduction:

The Coffee Bar is a café located at the Stamp Student Union Building at the University of Maryland, College Park which is run as part of the "Adele's Restaurant" falling under the umbrella of dining services. It caters to the beverage consumption needs of both staff and students at the University. At any given point of time, the Coffee bar attracts a range of customers, resulting in a multitude of transactions occurring on a daily basis. In order to facilitate the provision of these services, we observed they are essentially involved in employee management, generation of reports, inventory logs, and a menu of food & beverage items for the customer to choose from. The presence of such activities greatly warrants a need for the development and maintenance of a database capable of encompassing the operation of the Coffee Bar. The system they have at present is software/vendor specific thus limiting their functionality of use which results in tasks being performed manually. For example, they do not at present have a cohesive inventory management system that can track the demand and supply of food items and other requisite products for their day to day operation. What we proposed is to develop a database that mirrors the operation of The Coffee Bar in terms of capturing all information that they are associated with on a daily basis.

Target Audience:

To begin with, the database will primarily be used by the Manager at The Coffee Bar/Adele's Restaurant named Andre Christie. In addition to that the audience will depend upon specific functionality/information that we store in our database. For Example, individuals responsible for inventory management who will be an employee among the staff will make use of the same to monitor and track inventory.

Logical Design:

The logical design comprises primarily of four major entities- **Employees, Inventory, Menu Items and Ordered Items**. The Employees entity captures all information required in order to be working as an employee at the coffee bar. It is supplemented by the 'employment_availability' entity which records information relating to the days and time slots in which an employee is available to work. The tables namely 'day_of_the_week' and 'Time' serve as lookup tables to achieve this purpose. The Inventory table is used to store all the raw items that are used in the day to day operation of the coffee bar. Cups, lids, straws, tazo tea bags are a few examples of the kind of information that is present in this table. Menu Items lists the various items that can be purchased at the coffee bar for consumption by its customers. The Ordered Items entity provides us with a list of all items that were ordered at the Coffee bar which is identified using a combination of the order id, product id and employee id. These four areas cover the majority of the functionality and operation of the Coffee bar which is why we chose to focus on the same.

We faced many problems in constructing the design for recording the information relating to availability of employees. Our initial thoughts on this were to store the availability in relation to the actual date and time on which an employee was available to work. However, we realized that this would result in a multitude of rows being stored at the back end for every week when the employees would be scheduled. To overcome this we created two lookup tables which would only store the day and the time slots on which the employees were expected to work. This then acted as a lookup when storing this information with regard to an employee. Another problem, we faced was in terms of partial dependency of product_category on the product name when we had the same coded in the Menu_items table itself. With the help of the Professor we were able to identify a solution by creating a new entity named 'Product_Categories' which held the Category ID and Product_Category.

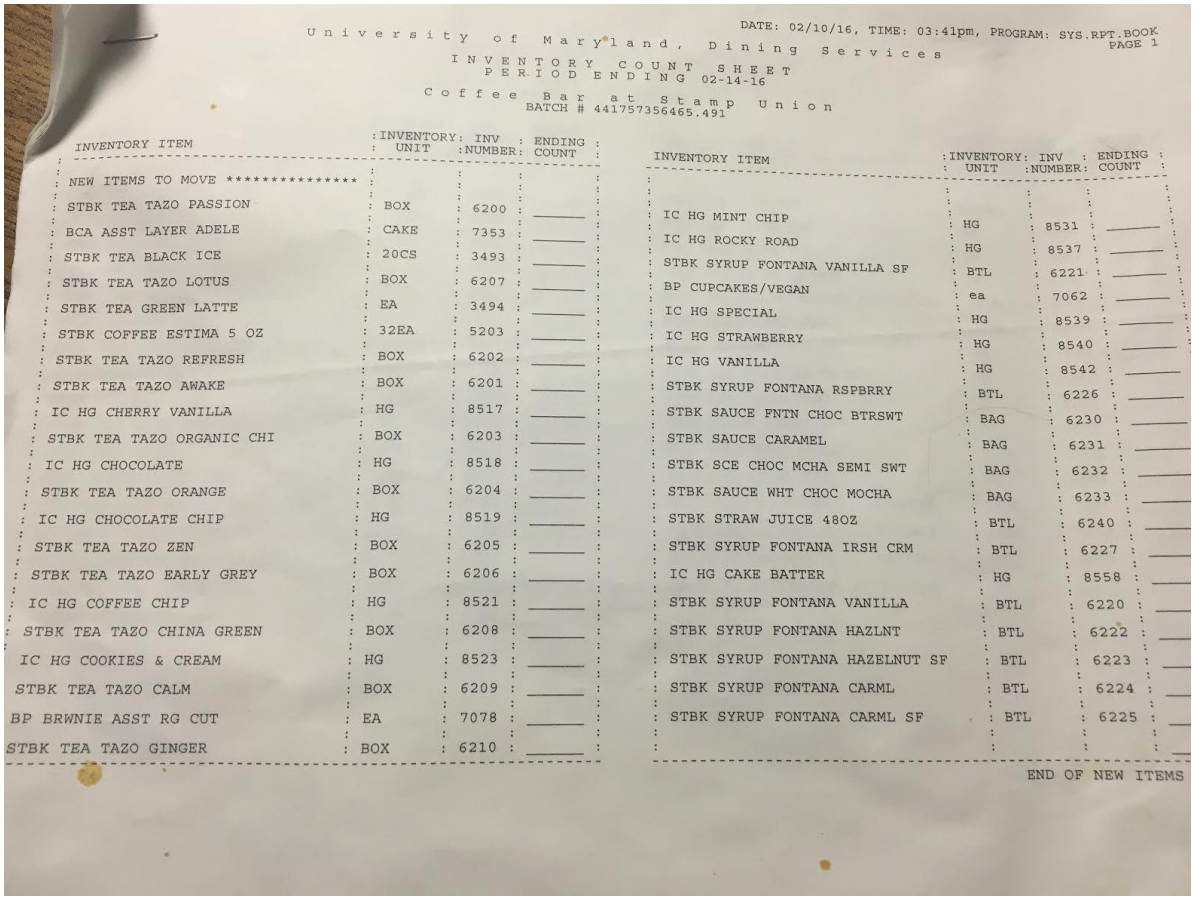
Physical Design:

In order to implement the design, we wrote manual ‘Create Table’ and ‘Insert Into Table’ queries. We did not rely on the workbench interface for creation or insertion of our data as it was our understanding that the manual process would promote more learning. One of the problems we ran into in this process was assigning the data types according to the type of data that the field would hold. For example, we needed the field Time_Stamp of ordered items to hold a value of date as well as time. We realized we can achieve this dual purpose by making use of the timestamp datatype supported by mysql which stored both date and time. We also learnt the syntax that is associated with providing a foreign key reference to a field present in another table using the create table query itself.

The sample queries document that was provided by the Professor in class was also of immense help to us as it helped in formulating queries by looking at examples. We also made several alterations to the design of our database after it was physically implemented in order to accommodate the changes we made to our ERD to be in keeping with the requirements defined. This was achieved using the ALTER and MODIFY commands.

Sample Data:

The data that we worked with was provided to us by the manager at the coffee bar. This pertained to inventory reports, example schedule generated and snapshot of the menu.



University of Maryland, Dining Services

DATE: 02/10/16, TIME: 03:41pm, PROGRAM: SYS.RPT.BOOK
PAGE 1

INVENTORY COUNT SHEET
PERIOD ENDING 02-14-16

Coffee Bar at Stamp Union
BATCH # 441757356465.491

INVENTORY ITEM	INVENTORY UNIT	INV NUMBER	ENDING COUNT
NEW ITEMS TO MOVE **********			
STBK TEA TAZO PASSION	BOX	6200	
BCA ASST LAYER ADELE	CAKE	7353	
STBK TEA BLACK ICE	20CS	3493	
STBK TEA TAZO LOTUS	BOX	6207	
STBK TEA GREEN LATTE	EA	3494	
STBK COFFEE ESTIMA 5 OZ	32EA	5203	
STBK TEA TAZO REFRESH	BOX	6202	
STBK TEA TAZO AWAKE	BOX	6201	
IC HG CHERRY VANILLA	HG	8517	
STBK TEA TAZO ORGANIC CHI	BOX	6203	
IC HG CHOCOLATE	HG	8518	
STBK TEA TAZO ORANGE	BOX	6204	
IC HG CHOCOLATE CHIP	HG	8519	
STBK TEA TAZO ZEN	BOX	6205	
STBK TEA TAZO EARLY GREY	BOX	6206	
IC HG COFFEE CHIP	HG	8521	
STBK TEA TAZO CHINA GREEN	BOX	6208	
IC HG COOKIES & CREAM	HG	8523	
STBK TEA TAZO CALM	BOX	6209	
BP BRWNIE ASST RG CUT	EA	7078	
STBK TEA TAZO GINGER	BOX	6210	

INVENTORY ITEM	INVENTORY UNIT	INV NUMBER	ENDING COUNT
IC HG MINT CHIP	HG	8531	
IC HG ROCKY ROAD	HG	8537	
STBK SYRUP FONTANA VANILLA SF	BTL	6221	
BP CUPCAKES/VEGAN	ea	7062	
IC HG SPECIAL	HG	8539	
IC HG STRAWBERRY	HG	8540	
IC HG VANILLA	HG	8542	
STBK SYRUP FONTANA RSPBRRY	BTL	6226	
STBK SAUCE FNTN CHOC BTRSWT	BAG	6230	
STBK SAUCE CARAMEL	BAG	6231	
STBK SCE CHOC MCHA SEMI SWT	BAG	6232	
STBK SAUCE WHT CHOC MOCHA	BAG	6233	
STBK STRAW JUICE 480Z	BTL	6240	
STBK SYRUP FONTANA IRSH CRM	BTL	6227	
IC HG CAKE BATTER	HG	8558	
STBK SYRUP FONTANA VANILLA	BTL	6220	
STBK SYRUP FONTANA HAZLNT	BTL	6222	
STBK SYRUP FONTANA HAZELNUT SF	BTL	6223	
STBK SYRUP FONTANA CARML	BTL	6224	
STBK SYRUP FONTANA CARML SF	BTL	6225	

END OF NEW ITEMS

Figure: Inventory Count sheet that we made use of to load the data

Inserting the data into the tables proved to be a challenge as we had to condense the data we had obtained in the format that we required first. The data was provided to us was primarily in the form of printed documents with the menu at the coffee bar sent as a snapshot. We worked on segregating this information based on the fields that we had defined and came up with a final list of values which we inserted into the database.

Queries:

Query name	Req. A	Req. B	Req. C	Req. D
Query 1	X	X		X
Query 2		X		
Query 3			X	
Query 4	X	X		X
Query 5	X	X	X	X
Query 6	X	X	X	X
Query 7	X	X	X	X
Query 8	X	X	X	X
Query 9	X	X	X	X
Query 10	X	X	X	X
Query 11	X	X	X	X

Table: Table detailing Query Requirements

Please find mentioned below a concise description of the function of each query that we have coded in the database:-

Query 1: Provides a list of employees who have requested for leave

Query 2: Check on particular items in the Inventory that have fallen below a given level as set by the user

The following query considers the below mentioned items in the inventory

1) STBK TEA TAZO ORANGE

2) STBK TEA TAZO ZEN

3) STBK TEA TAZO REFRESH

Query 3: Checks the value of existing stock in the Inventory

Query 4: Checks who the immediate supervisor of an employee is

Query 5: Checks the total number of items that were sold belonging to a particular category And a particular date

Query 6: Lists the total number of items that have been sold so far belonging to a particular category

Query 7: Lists the menu items that are being sold in High Frequency with quantity level set by user

Query 8: Provides breakdown of the amount that has been generated by the cafe on a day-to-day basis

Query 9: Generate the Date and Amount in the week the cafe earned the highest revenue in terms of sales

Query 10: Shows the revenue that was generated by the cafe on a particular Day as set by the user

Query 11: Provides the list of employees who are available to work along with time slots. It also takes into account the Date set by user in order to avoid stale entries from the employee_leave table interfering with current data.

The reasoning behind the selection of our queries was to try and incorporate SQL concepts such as inner join, left join, aggregate functions, sorting functions learnt in class with the our business requirements at hand. Majority of our queries comprise of usage of inner join with multiple tables. There is one query where we made use of a **LEFT JOIN** in order to only select the mismatched rows that were belonging to the employee_availability table which was joined with the employee_leave table. This was done to provide the user with a list of all employees who are available to work and have not applied for a leave. We made use of aggregate functions such as **SUM, MAX** and **COUNT** in relation with calculating metrics for the revenue generated by the coffee bar. The use of **GROUP BY** clause helped us understand how it is to be used in conjunction with an aggregate function in order to group similar rows to form a single group. Moreover, we also learnt how to use a **HAVING** clause with **GROUP BY** in order to check a condition on a calculated value in the **SELECT** statement. The use of **ORDER BY** clause was especially useful in formatting the output in terms of sorting the data so that it would promote easy readability for the output obtained from queries. We made use of aliases to better represent the column names of output obtained from the queries. We have also implemented an update trigger which will be fired every time an Update takes place on the Inventory table. This trigger will help us log the old quantity of an item and the updated new quantity that was made to the stock and by whom was the same made along with a timestamp. The information pertaining to this is stored in the 'update_inventory_log' table.

User Interactions:

We have used LucidChart to design the forms and user interactions of our database. We have created two data input and two data retrieval forms.

DATA INPUT:

1. **Employee Availability Form:**

The Employees of the Coffee Bar will be required to fill in their availability to work using this form. The employees will have to submit their availability in terms of the days they are available to work and the time slots on which they can work. It is important to note that the employee must fill out the form again for different days. This has been designed so taking into consideration that employees could be available to work at different time periods on different days. Moreover, this form of data capture is in keeping with the design of our database wherein the data will be stored in the 'employee_availability' table.

2. **Leave Request Form:**

This form pertains to those employees who wish to apply for a leave. In order to do so, they must mention their employee ID, the date on which they wish to take leave, the day that date falls under along with the time slots on which they are not available. This form will help capture the leave information and will be recorded in the 'employee_leave' table as part of our database design.

DATA RETRIEVAL:

1. **Inventory Count View:**

The inventory count view is designed such that the user only has to enter what the business terms as a 'PAR' value. In other words, it provides a list of those items that have fallen below a certain level that is set by the manager of the coffee bar. This level is matched with the item quantity that is present at the back end in the inventory table.

2. **Employee Availability View:**

The Employee Availability view allows the manager to filter records based on the selection of days and slots as required by him. Once they have selected the same, the form will go on to reflect all employees that are available to work on the selected days and time slots. For example, if the manager requires an employee to work on Friday from 5:30PM- 9PM. They have the

option of selecting the day and time through this form and viewing all employees who are available to work on that particular day and time.

Problems Faced and Lessons learnt:

We faced a problem with one of the queries wherein we had to select the maximum value from the result of a group by statement. The use of MAX(SUM(product_cost*frequency_count)) resulted in an invalid use of aggregate function error message. We overcame this problem with the help of subquery which we obtained through the round robin discussion in class. A minor problem that we faced was in terms of ambiguous use of similar column names belonging to different tables through joins which we overcame by the use of referencing by assigning a variable to the table names. Another important point of learning was the use of DATE function. The timestamp datatype in mysql stores the date and time fields together when inserting a date value. However, some of our queries involved checking only the DATE portion of a field, ignoring the time. We were able to strip the field to only select the DATE by making use of the DATE function. In the process of coding the trigger we faced a problem of specifying a delimiter character, specifically '//' which we solved through the lecture in class discussing the use of the same when implementing a trigger. While implementing the physical design, exporting and importing to MySQL Workbench and sharing our work in .sql files was a problem. We learnt from Dr. Lawley during class interactions that importing the file to a new schema solves this problem as it sometimes interferes with certain constraints of an existing database.

Future Improvements on Database:

The payment system was not tracked in this database as including an entity for payments made the database complex and outside the scope of the project. In the future, a payment entity could

be included in the ERD where the payment system could be tracked. Customers accessing the The Coffee Bar are allowed to pay through cash, credit/debit cards, Terp bucks and Terrapin express. These different types of payments could be reflected in the database. Also, employees earning tips apart from their hourly wage could be incorporated in the payment entity which would make the database more robust.