Name: Ememobong Edem Ekpenyong

URN: 6676807

Group: 36

1. Collaboration Approach:

Our collaboration approach was to work together as a team in understanding the business problem and identification of tasks. Everyone passively worked together on all tasks, but each member took the lead and played an active role on his or her assigned tasks.

2. Individuals' Roles in the Group:

Patrick Oramah:

1. Creation of the OrderGroup Table as per business requirements
2. Develop the Stored Procedure, prCreateOrderItem

Ibukun Omojola:

1. Normalization of OrderItem Table to 3NF.
2. Contributed to the implementation of the BI solution in areas of creating the Data Source, Data Source View.

Judith Etiobhio:

1. Normalization of the Product Table to 3NF.
2. Created the Stored Procedure, PrCreateOrderGroup

3. My Contribution:

My assigned tasks:

1. Normalization of CustomerCity Table to 3NF and creating suitable model constraints.
2. Contributed to the implementation of the Business Intelligence solution in the areas of creating the Cube, setting up hierarchies, and developing custom calculated measures.

Normalization of the CustomerCity Table:

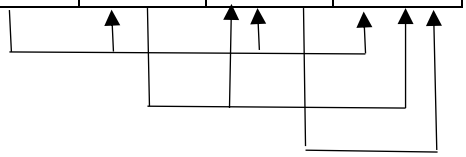
They were 9 attributes in the **dbo.CustomerCity** relation with 51,992 rows and no null values.

The Id attribute was identified as the primary key because the values are unique and not null. The Id column contains a total of 51,992 unique rows. This implies that there is no partial dependency in the CustomerCity table.

Therefore, the CustomerCity table as shown below is in 2NF – meaning it is in 1NF and there are no partial dependencies. However, there are transitive dependencies which will be addressed in 3NF.

CustomerCity Table

Id	**Gender**	**FirstName**	**LastName**	**DateReg**	**City**	**County**	**Region**	**Country**



The aim of 3NF is to achieve 2NF and remove all transitive dependencies. I identified three transitive dependencies in the CustomerCity table as described below.

A customer's City determines the County. City -> County

A customer's County determines the Region. County -> Region

A customer's Region determines the Country. Region -> Country

Interestingly, there was a debate in my group on whether to split the CustomerCity table into two tables consisting of only the Customer Table and CustomerAddress Table (with City, County, Region and Country in a single table) as this will result in lesser join operations when retrieving data or to split into five tables consisting of Customer, City, County, Region, Country tables. At the end, we agreed to decompose the CustomerAddress table into City, County, Region and Country based on the evidence that functional dependencies were identified between them.

Customer Table

Id	**Gender**	**FirstName**	**LastName**	**DateRegistered**	**City(fk)**

City Table

City	**County(fk)**

County Table

County	**Region(fk)**

Region Table

Region	**Country(fk)**

Country Table

Country

Model Constraints:

The Country column was made the primary key in the Country table.

The Region table was created with the Region column as the primary key and the Country as the foreign key referencing the Country column in the Country table.

The County table was created with the County column as the primary key and the Region as the foreign key referencing the Region column in the Region table.

The City table was created with the City column as the primary key and the County as the foreign key referencing the County table.

The CustomerCity table provided was renamed to Customer Table with Id, Gender, FirstName, LastName, DateRegistered and City as the columns, City was made the foreign key referencing the City column in the City table.

Indexing:

Clustered index was applied to the Id column on the Customer table because the values of the Id attribute are increasing and unique. With a clustered index, the values will be sorted and ordered. Hence, a suitable choice of indexing for the Id column.

Similarly, clustered index was applied to the primary keys in the City, County, Region and Country tables as clustered indexes are very fast in executing search operations.

Business Intelligence Solution:

The first step was to create a Data Source which is a link to our database, followed by the creation of the Data Source View (DSV). The DSV extracts the schema from our database and the Day, Month, Year attributes were created from the OrderCreateDate attribute as Named Calculations. Also, FullName attribute was created for ease of identifying a customer as different customers may have the same FirstName or LastName. Alternatively, a customer can be identified by the Customer's Id.

Order Item and Order Group were selected as measures because they contain numeric attributes that evaluates the organization's performance.

Product, ProductVariant, Customer and ProductGroup were selected as dimensions as these tables contain the information required to satisfy the data requirements requested by the business.

The following hierarchies were defined to be used for dicing and slicing the cube.

1. CustomerLocation: Country -> Region -> Cities
2. Calendar: Year -> Month -> Day
3. ProductVariant: ProductCode -> VariantCode

All the calculations requested for in deliverable 2 have been calculated and saved under Measures. The values can be gotten by doing a drag-and-drop on the Cube browser.

Challenges:

In the Business Intelligence part, the cube crashed due to memory issues when trying to do a drill down for Order calculations, Sales and Quantity by product, by customer, by day on the lab computer.

4. Design decisions:

For the Business Intelligence solution, we used two Measures, OrderItem and OrderGroup tables because these tables contain the numeric columns required to evaluate the business performance.

In the Product Table, considering that the Product Group does not have a dependency with any attribute, we combined the ProductCode with the ProductGroup and created a composite primary key with both attributes as primary key resulting in a many-to-many relationship.

In the PrCreateOrderGroup Stored Procedure, we assumed that orders cannot be created in the past or future, therefore, OrderCreateDate will always be the current date.

5. What I Learnt:

Before this project, I was only able to write simple select queries in SQL but now I understand normalization and its different stages, I can write complex queries, learnt how to write Stored Procedures and how to design a Business Intelligence solution using Microsoft SSAS.

6. Appendix: Evidence of Collaboration (with MS Teams links)

Minutes of the meetings for Database Coursework Group 36

MEETING 1

Date: 24th-October-2021

Location: IFH lab

Task: Problem understanding and task allocation:

- The team discussed the task assignment and agreed to learn about the various database normalisation techniques.
- Brainstormed some idea that were put forward

The team decided to divide the coursework deliverables into different tasks:

Patrick: 1. Creation of the OrderGroup Table

2. Develop the Stored Procedure prCreateOrderItem

Ememobong: 1. Normalization of CustomerCity Table to 3NF

2. BI solution implementation

Judith: 1. Normalization of the Product Table to 3NF

2. Stored Procedure for PrCreateOrderGroup

Ibukun: 1. Normalization of OrderItem Table to 3NF

2. BI solution implementation

Action points: Patrick recommended some videos on YouTube. We decided to watch it to enhance our understanding of the problem for the next meeting.

MEETING 2

Date: 11th November 2021

Location: MS Teams

Task: Identify functional dependencies of MumsNet.com database:

- The four tables were reviewed with the aim of identifying all functional dependencies. To achieve, different select queries were written and the results discussed with the team members.
- The team decided to assign the tables to every team member because of time constraint

Ememobong - CustomerCity table

Judith – Product table

Ibukun- Order Item table

Patrick: Order Group table

Action points: Normalise the allocated tables to 3NF by first identifying and removing all the dependencies.

MEETING 3

Date: 17th-November-2021

Location: MS Teams

Task: Normalization and Stored Procedure

- The team discussed and brainstormed on the tasks and its requirements.
- We watched a few YouTube videos on how to write a Stored Procedure with TRY/CATCH error handling and data validation.

Action points: Develop the PrCreateOrderGroup and prCreateOrderItem Stored Procedures.

MEETING 4

Date: **22nd-November-2021**

Location: **MS Teams**

Task: **Report writing and Business Intelligence Solution Implementation**

- The team worked together on the cube and tried to replicate the lab tutorials in a bid to familiarise themselves.

Action points: Every team member practises more by creating the Data Source, Data Source View and the Cube with its measures, dimension and hierarchies.

MEETING 5

Date: **1st-December-2021**

Location: **IFH lab**

Task: **Business Intelligence Solution Implementation**

Attendance: Ememobong Ekpenyong, Judith Etiobhio, Patrick Oramah, Ibukun Omojola

Action Points:

- Using our database, we tried to create a data source from the .bak file.
- Created a data source view with some Named Calculations
- Designed the cube with its respective measures, dimensions and hierarchies.
- Answer the questions asked in deliverable 2 by creation Calculated Measures.

Teams links and Group pictures:

https://surreyac-my.sharepoint.com/:v:/g/personal/ee00475_surrey_ac_uk/EW4HXNqTbJRDrq2LwFnGTtYB6DKOLeWCPwLgkocF7Ms9xg

https://surreyac-my.sharepoint.com/:v:/g/personal/ze00013_surrey_ac_uk/EaqaqwNHllNKr8oOP7y-JEsBNuc6HcsaHJjdBB9jlhq2qQ

https://surreyac-my.sharepoint.com/:v:/g/personal/ee00475_surrey_ac_uk/ET0dRlR26rJLuCvTYav-cv8B91XhjtMnsaxg-leDMl3zQQ

https://surreyac-my.sharepoint.com/:v:/g/personal/io00261_surrey_ac_uk/ERRd2OSgfQNOi1NaYJGQ4CIBWdMnBGsPTQoye91zX4gi5A