Meeting 1 11am UK time 10NOV24

Team contract

- I downloaded it and filled it in with what we had already been discussing the days before.
- Established the team roles (which will rotate every 10 days).
- Discussed the written content and made sure everyone was on board and comfortable.

We do not need the actual data. We are only building the concept of the database design

Choosing the application & client profile

- Option 1 - Census Data (eg. NSO Malta)
  - ↳ Big data captured by statistical offices.
  - ↳ Collected from people. (easier collection)
  - ↳ We can create tables from different sets of data.
  - ↳ May be too complex for this application.
  - ↳ Would need internal permission to use from work.

- Option 2 - Workforce Intelligence (HR) (eg. Kenvue NJ)
  - ↳ Administering the data at a company.
  - ↳ Multiple tables eg. Table 1 has addresses, cities, states
    Table 2 has pay/salary info
    Table 3 has managerial info
    These are linked using SQL to create a database.
  - ↳ Analytics are happening in one house and survey data information/collection is happening in another house. The connections between all this are the APIs. Then connecting the information of the people is coming from PostgreSQL server.
  - ↳ PostgreSQL to Qualtrics to Snowflake (everything happening through API integrations). The data is being passed that way back and forth.
  - ↳ Complex idea.

- Option 3 - School
  - ↳ Students get positive & negative points for their behaviour.
  - ↳ Consequences such as detention, suspension, reflection.
  - ↳ Used to track academic progress with behaviour.
  - ↳ Teachers log in misbehaviour in class, lateness, work ethic, etc.
  - ↳ Centralised cloud access for remote access.

- Option 4 - Heath Survey for England
  - ↳ Tracks economic status, academic status, how many children, whether they drink or smoke, etc.
  - ↳ Huge dataset.
  - ↳ Conducted in the UK every year.
  - ↳ Publicly available data.
  - ↳ Similar to census data.
  - ↳ But given that it's government data, what would they need in terms of database design? The database we would create would just be linking tables but for what reason? What are we solving? What would our design be improving?

- Option 5 - Retail (eg. clothes shop)
  - ↳ You can have stores in different locations
  - ↳ You are writing in a table/csv that you sold this many products to this person.
Was it paid by cash or card?
Date of transaction?
  - ↳ We can house all that information
  - ↳ eg. if you have a membership with a store, you'd have the products that you purchased linked, as well as a username, ID, Name & surname, email, etc. So if you purchased 5 items today, they'd be recorded on your profile.
  - ↳ You'd have all of your users in one table & all of their personal information data, then you'd have the purchases table & you can link them together. You can also have a locations table to link to which store/location the purchase was made.
  - ↳ Then you get ads targeted to you based on what you bought (or vouchers to encourage you to buy more).

To do before next meeting (14NOV24 5:30pm UK time)

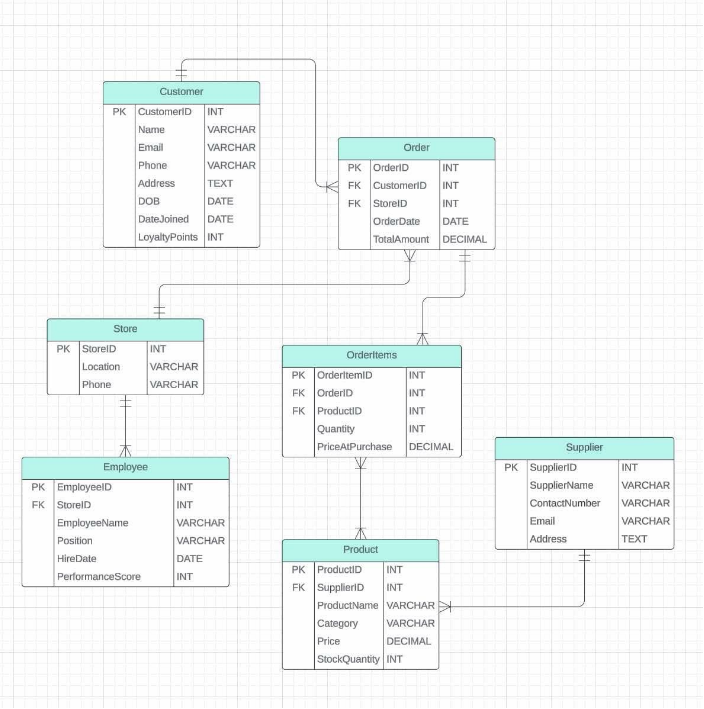
- Create an overall basic design with variables such as customer ID, Store location, etc.
- Develop a concept of how these tables would be linked.
- Send email to tutor with team contract by evening.

Meeting 2 5:30pm UK time 14NOV24

We first shared our screens in turns to show what we each have done so far.

↳ Kieron worked on how the basic design would look like using tables
ie. the four tables consisted of the customer, store location, purchase details, . . .
& used MySQL Workbench

↳ I showed my own database design idea using Lucidchart: (using Crow's Foot Notation)



Customer

PK	CustomerID	INT
	Name	VARCHAR
	Email	VARCHAR
	Phone	VARCHAR
	Address	TEXT
	DOB	DATE
	DateJoined	DATE
	LoyaltyPoints	INT

Order

PK	OrderID	INT
FK	CustomerID	INT
FK	StoreID	INT
	OrderDate	DATE
	TotalAmount	DECIMAL

Store

PK	StoreID	INT
	Location	VARCHAR
	Phone	VARCHAR

OrderItems

PK	OrderItemID	INT
FK	OrderID	INT
FK	ProductID	INT
	Quantity	INT
	PriceAtPurchase	DECIMAL

Supplier

PK	SupplierID	INT
	SupplierName	VARCHAR
	ContactNumber	VARCHAR
	Email	VARCHAR
	Address	TEXT

Employee

PK	EmployeeID	INT
FK	StoreID	INT
	EmployeeName	VARCHAR
	Position	VARCHAR
	HireDate	DATE
	PerformanceScore	INT

Product

PK	ProductID	INT
FK	SupplierID	INT
	ProductName	VARCHAR
	Category	VARCHAR
	Price	DECIMAL
	StockQuantity	INT

↳ Naveed presented syntax of how relational tables would be written in SQL code.

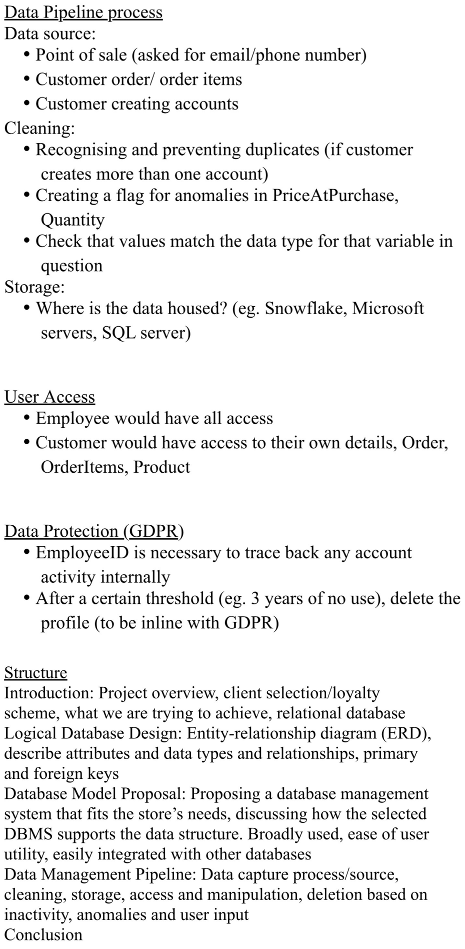
We decided to move forward with my design since my colleagues said the tables and the relationships are well-presented and organised. We would just need to translate it into SQL code

How shall we structure the report? Some important points:

- We can include a table to explain each flow
- Who has access to what? eg. customer has access to Order, OrderItems and Product BUT the employee has access to all the tables
- For storage, we need cloud servers like Snowflake (Snowflake is basically SQL online), because this is something that needs to be accessed by multiple stores & also customers.
- For the data source, we would have a problem tracking customers if they do NOT have an account with the store. Even if they use credit cards to purchase and the system links all the purchases made with the card in question... what about customers who pay in cash? Those customers are definitely NOT traceable.

  ∴ We set up our design only for a store's loyalty program/scheme

Below are some more notes I wrote during the team meeting:



Data Pipeline process

Data source:
- Point of sale (asked for email/phone number)
- Customer order/ order items
- Customer creating accounts

Cleaning:
- Recognising and preventing duplicates (if customer creates more than one account)
- Creating a flag for anomalies in PriceAtPurchase, Quantity
- Check that values match the data type for that variable in question

Storage:
- Where is the data housed? (eg. Snowflake, Microsoft servers, SQL server)

User Access
- Employee would have all access
- Customer would have access to their own details, Order, OrderItems, Product

Data Protection (GDPR)
- EmployeeID is necessary to trace back any account activity internally
- After a certain threshold (eg. 3 years of no use), delete the profile (to be inline with GDPR)

Structure

Introduction: Project overview, client selection/loyalty scheme, what we are trying to achieve, relational database
Logical Database Design: Entity-relationship diagram (ERD), describe attributes and data types and relationships, primary and foreign keys
Database Model Proposal: Proposing a database management system that fits the store's needs, discussing how the selected DBMS supports the data structure. Broadly used, ease of user utility, easily integrated with other databases
Data Management Pipeline: Data capture process/source, cleaning, storage, access and manipulation, deletion based on inactivity, anomalies and user input
Conclusion

This word document was sent to the other team members after the meeting.

Plan of action before next meeting (20NOV24 5:30pm UK time)

- Someone will take over writing the first version of the content.
- The others will add and amend the content one at a time.
- Find some academic sources

Meeting 3 4pm UK time 27 NOV 24

The write-up has been edited by all 3 members and is at its final stages.
I have 2 suggestions:
- Should we add a conclusion?
- Should we expand on the data pipeline process?

Change "Introduction" to "Overview"
Change "System Robustness" to "System Rules and Security"

Explained why I changed the quotations to paraphrased sentences (since it is advised on the Referencing Guide)

Agreed on Final version
- Ready to Submit