Database Project Group 2

Aaron Schapira i6211359 Arthur Goffinet i6215214
Martin Michaux i6220118

September 2020

1 Introduction

For this project, we were asked to design a database for a specific task. We chose to create a database management system for a grocery.

2 ER model

See figure 1 page 4.

3 Relational model

See figure 2 page 5.

4 Questions

- Describe the problem concisely. Why is a database a good idea for this task? Describe also the intended stakeholder of your database system.

  Answer:
  This is a basic case where we may implement a DBMS where we explore the different needs of a DBMS. In a grocery, a DBMS has its place thanks to the amount of information that are present inside of a grocery.
  Possible stakeholders for this task would be chains of shops such as Albert Heijn.

- Construct an entity relationship (ER) Your model should contain at least 5 entities and 5 relationships. Document your ER model. Highlight the design decisions - describe the role of each entity and the role of each relationship.

  Answer:

- EMPLOYEE: Entity that represents an employee
  e_ID : Attribute(primary key) that represents an ID given by the company
  Last_Name : Attribute that represents the last name of the employee
  First_Name : Attribute that represents the first name of the employee
  Email_Address : Attribute that represents the mail address of the employee
  Sex : Attribute that represents the gender of the employee
  Phone_Number : Attribute that represents the phone number of the employee
  Id_contract: Foreign key, link to key c_Id of Contract entity

- MANAGER: entity that represents a manager
  m_Id : Attribute that represents an ID given by the company
  Lead_Position : Attribute that represents the lead position of the manager
  Id_contract: Foreign key, link to key c_Id of Contract entity

- JOB CONTRACT: Entity that represents a job contract
  c_Id: Attribute(primary key) that represents the ID of the contract
  Salary : Attribute that represents the salary of the user
  Position: Attribute that represents the position of the user
  Start_date: Attribute that represents the beginning date of a contract

- GROCERY: Entity that represents a grocery
  g_Id : Attribute(primary key) that represents an ID for the grocery
  Grocery_Name : Attribute that represents the name of the product
  Category : Attribute that represents the category of the product
  Price : Attribute that represents the price of the product

- SHOPPING LIST: this is a Bridge table between Grocery and user entity
  User_Id : Attribute (foreign key from Contract User) that represents the ID of the user that uses the shopping list
  Id_Grocery : Attribute (foreign key from Contract Grocery) that represents the id of the grocery of the shopping list
  Quantity : Attribute that represents the number of grocery for each user entity

- User: entity that represents a user
  u_Id : Attribute(primary key) that represents the ID for the user
  Last_Name : Attribute that represents the last name of the employee
  First_Name : Attribute that represents the first name of the employee
  Email_Address : Attribute that represents the mail address of the employee
  Phone_Number : Attribute that represents the phone number of the user (10 digits)

- BRIDGE MARKET GROCERY: this is a Bridge table between Grocery and Market entity, it is a weak entity because it needs the user entity (total participation from this weak entity to the user entity)

m_Id : foreign key from Contract Market
g_Id : foreign key from Contract Grocery

- Market : Entity that represents a market
m_Id : Attribute(primary key) that represents the ID of the market
Id_Manager : Attribute(foreign key from Manager) that represents the ID of the manager of the market
Location : Attribute that represents the location of the market (GPS coordinates)
Market_Name : Attribute that represents the name of the market

- Map your ER model to a relational model. Demonstrate the functional dependencies of the tables. Normalize your tables (relations) based on your design decisions. In what normal form are the tables? Explain why did you decide to normalize each table in a certain normal form.
Answer:

  Since all our attributes are atomic and single valued, we can say that it is normalized to 1NF. For example, the name is divided in two different attributes.

  We broke our relations such that every partial key, with their dependent attributes is in a separate relation. So, we only kept attributes that depend totally on the primary key. Thus, since we do not have any dependencies, our relation model is normalized to 2NF. For example, the employee table, contains only attributes that are dependent on its primary key (e.g. the employee id).

  Finally we made several bridge tables to avoid repetition in the main table.

  By doing so, we made sure that no non-prime attribute is transitively depending on the primary key. (We again broke up the relation such that the attributes that are depending on not-key attributes appear in a separate table (definition from the slides)). For example, we have created a table for the contract to avoid redundancy of information about it in the employees and manager tables. This also avoid transitive dependency.

  Finally, by doing those changes, we made sure that this would make our database normalized to 3NF.

5 Select queries

1. This first query selects all grocery names that have been bought by the user id 1 ("Donald Trump").

2. This second query also finds all grocery names that have been bought by the user id 1 ("Donald Trump") but also the total price of those groceries.

Our database needs those 2 last queries in case we want to retrieve info about the grocery bought by a certain customer.

3. This third query selects all groceries that come from the markets in "Bruxelles". This kind of demand is useful for the a customer that asks for a specific grocery around its location.

4. This fourth query selects every employee with a salary between 1500 and 4000. This is useful in our database for the market (manager) that can retrieve info about the employee salary.

5. This last query finds every groceries and check how many are sold and how much they urn for the grocery id 1. We think it is useful for this database because it gives info about every grocery if they are sold or not. This is useful to know whether a market needs a refill for specific groceries or not.

6 Views

1. This is a view of all grocery name. We think is is useful for the end user because only the user will be only able to see the grocery, nothing else.

2. This is a view of what the client bought and how many it cost. It is useful for the end user (that specific client) because he only wants to see what groceries he bought.

3. This is a view of all groceries that come from the markets in "Bruxelles". We think it is useful in our database for the end user because he only wants to see the groceries he can buy (e.g. the groceries around its location).

4. This one creates a view of every employee with a salary between 1500 and 4000.

   This would be useful for a grocery company because it could search efficiency for a list of employees given specific salary constraints.

5. The last one creates a view of every grocery order, ordered by category, and within this sorting, it is sorted by price by descending order.

   Finally, we think this last view could be useful for a company because it gives a pretty good view of every grocery in its category with its price, plus they are ordered which is good for retrieving information.

7 Conclusion

We think that we made our database has a good structure, regarding the normalization handling. Then the different selects and views allow the client to see whatever information it needs to see.

For those reasons, we think our DBMS could be useful for a grocery company in condition to add function, procedure for atomizing the select, update, delete. For the second part, the implementation should not be too hard to implement because our ER model is well normalized.

NB: We also added a SQL Script of examples of deletions and updates.

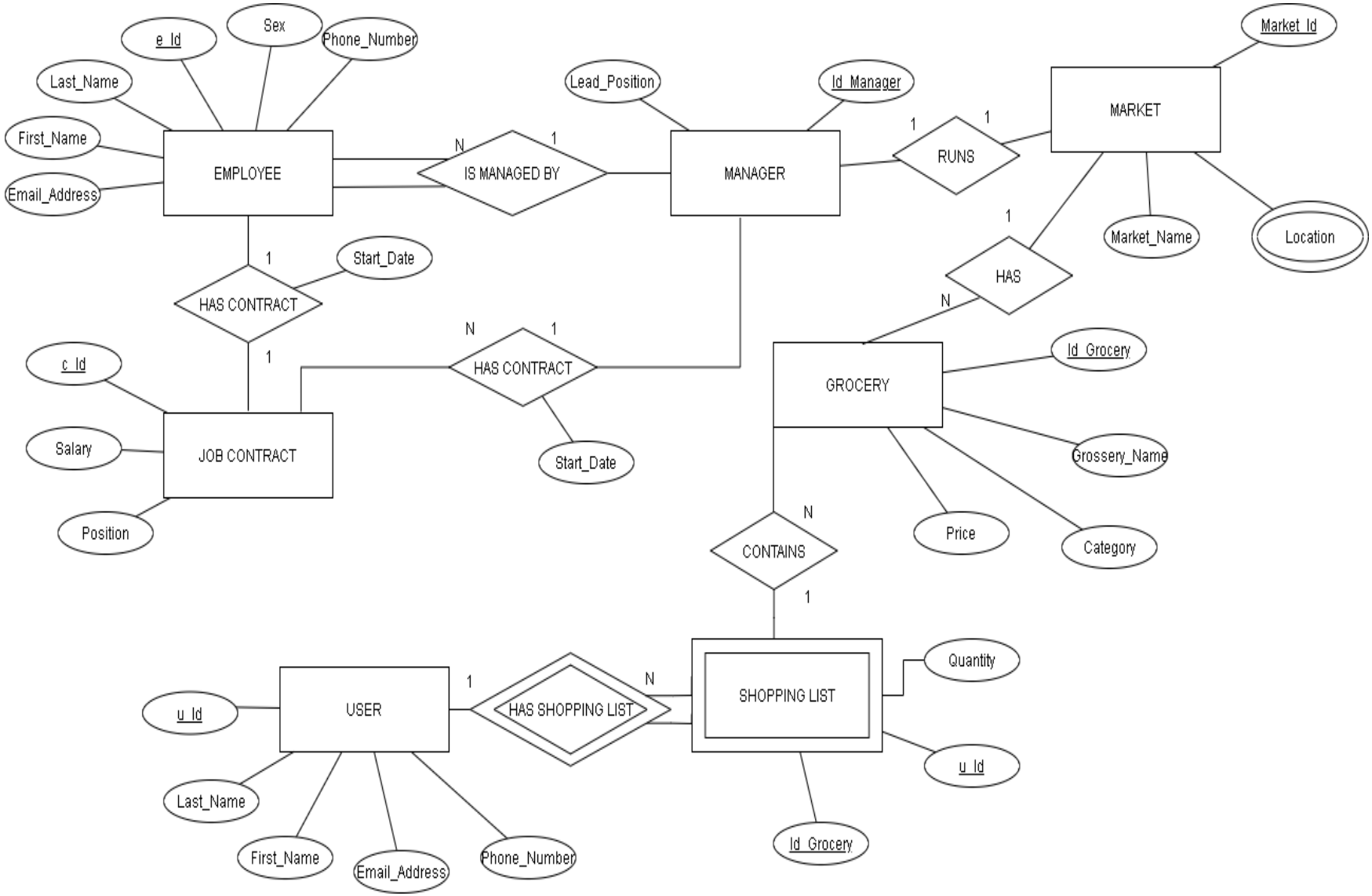

Figure 1: ER Model

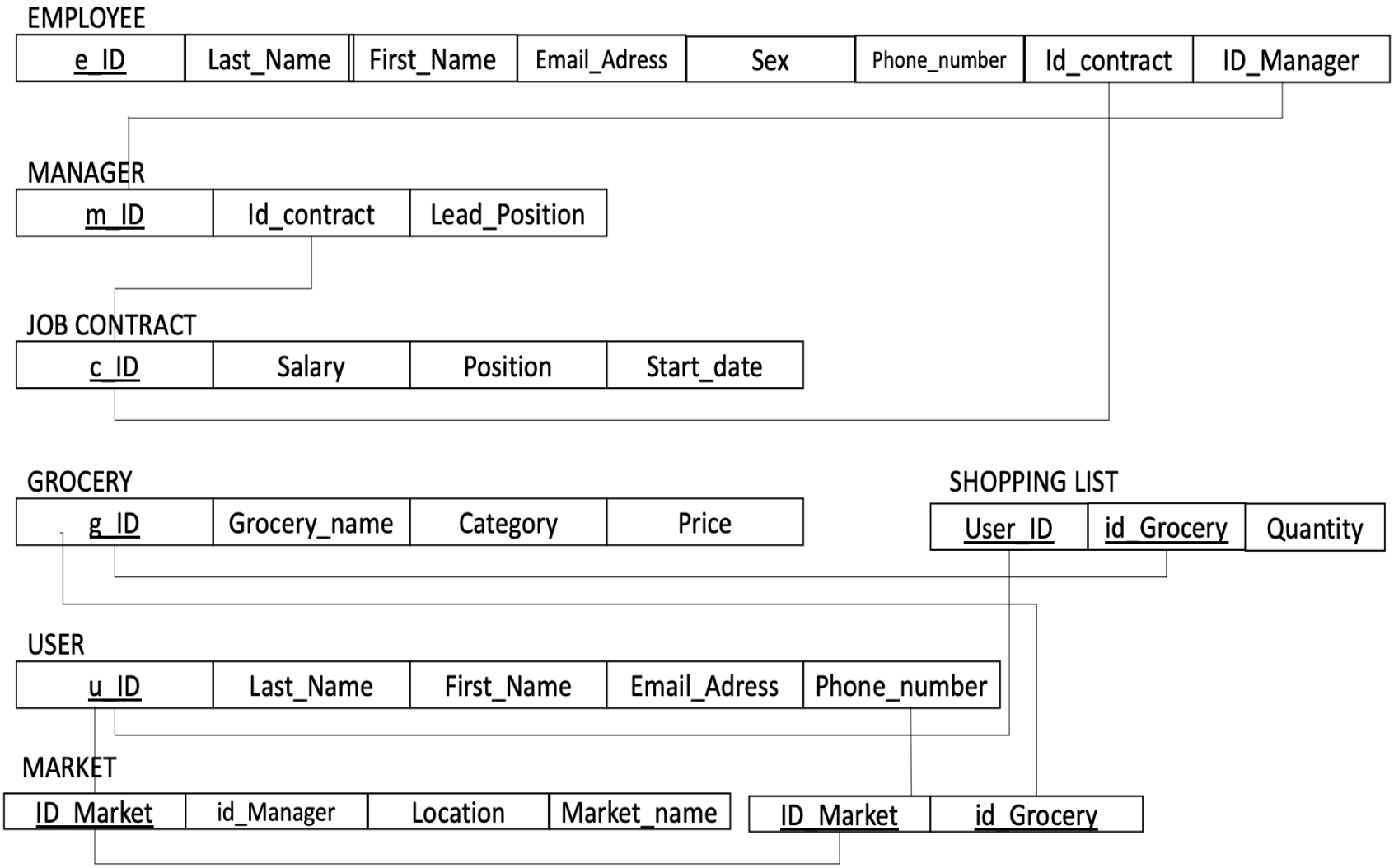

Figure 2: Relational model