

DATABASE MANAGEMENT

ASSIGNMENT 4

INSTRUCTOR

Ahmet Soran

Samet Tonyalı

GROUP MEMBERS

Tacettin Batuhan Bostancı

Ayşe Şeyda Çalışkan

Mustafa Demiröz

Mehmet Anıl İrfanoğlu

Dhiya Ulhaq

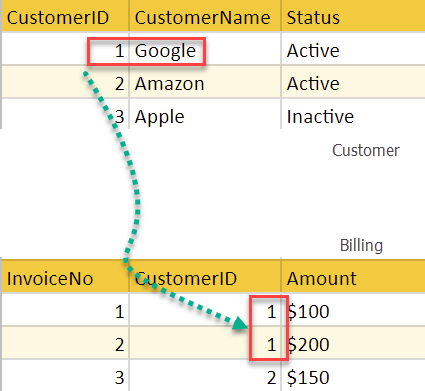
**INTRODUCTION**

In this article, we will talk about the database system in general, both about the things we learned in the lessons and the things we learned from the internet ourselves. Database systems are very important in terms of storing data. The reason for this is that we live in a digital world and we create data in many transactions we do. It is of great importance that the data we create is stored effectively. The reason for this is that if this much data is not stored well, serious performance and optimization problems may be encountered when this data is desired to be used again. A database system has been established in order to prevent such problems.

**DATABASE MANAGEMENT SYSTEMS**

In this part, we will mention about important topics which are needed to understand the logic of database systems.

**Relational Model**: In order to create a database, we have to know the connection between the data we want to keep. Defining the relationship between attributes that we symbolize in tables is of great importance for the system to work properly.



**DDL (Data Definition Language):** As the name suggests, DDL is called the type of code we write to determine how the data we have will be stored in the system. With the help of DDL, the tables where the data will be kept are prepared and which table has which attribute, which key primary key, which key foreign key, whether the stored data string or number, and the determination of such properties depends on the codes written with DDL logic.

**DML (Data Manipulation Language):** We need codes written in DML logic to add information to the database we use and to update the information in the database.

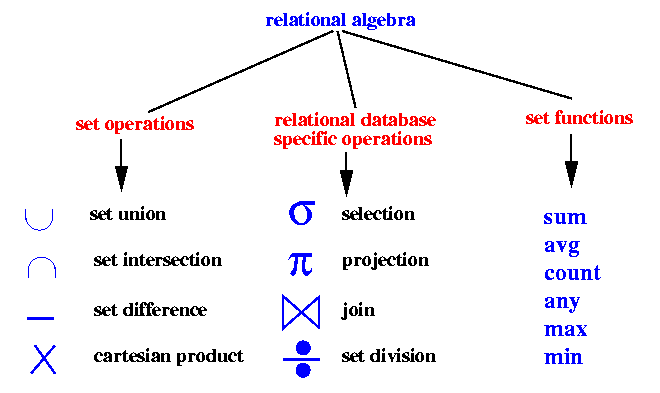
**DQL (Data Query Language):** In order to recall the information stored in the database for reuse, codes written with DQL logic are needed. With the Select statement, data extraction is performed and the Select statement is examined under the DQL heading.

**Query:** In order to run the codes for the database, it needs to be query. With the query, the desired operation is sent to the database as a request, and the database performs the required operation in line with this request, and the query structure is required for this operation.

**Attribute:** As we know, we store data in the database system and this data is stored under attributes. A separate column is used for each different attribute.

**Key:** The key, which has many types, is the element that is necessary to define attributes and makes each row different from each other.

**Relational Algebra:** As we know, the operations performed in the database are based on jset logic and it is called a topic that consists of mathematical terms used to explain the cluster logic. It is very easy to code mathematically prepared queries.

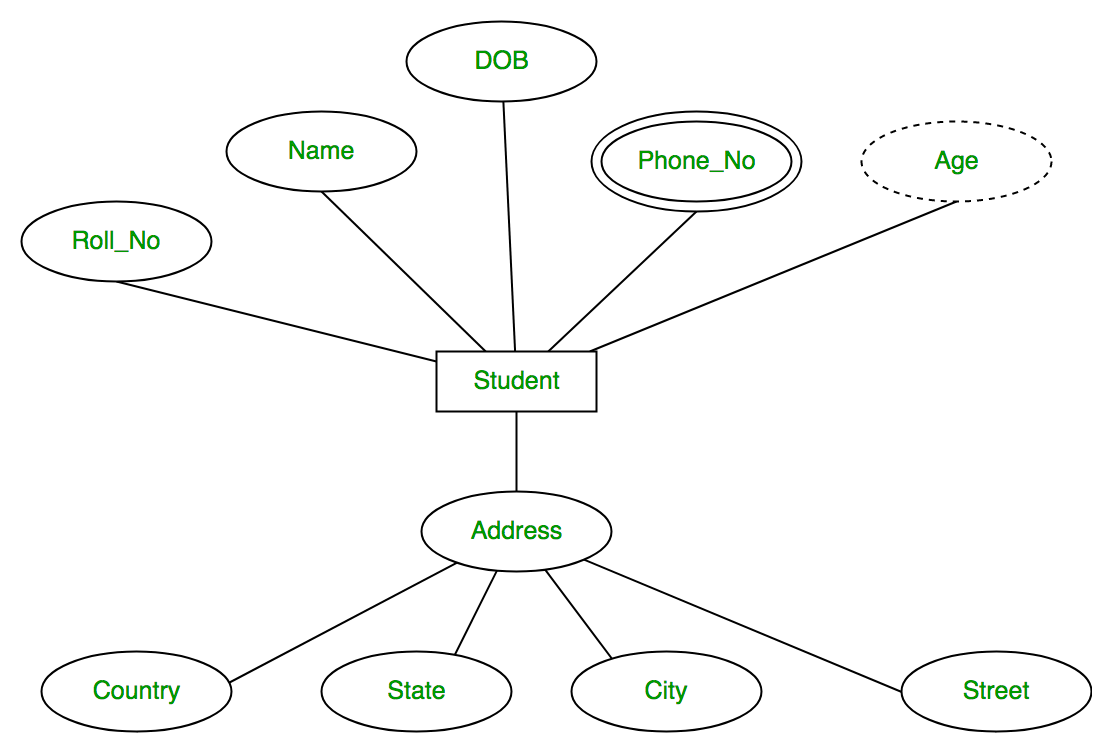


**Join:** Join operation is also of great importance, since all operations in the database are based on cluster logic. join operation is used to benefit from the relationships between data. Join operation, which has many types, is one of the most important operations in the database system.

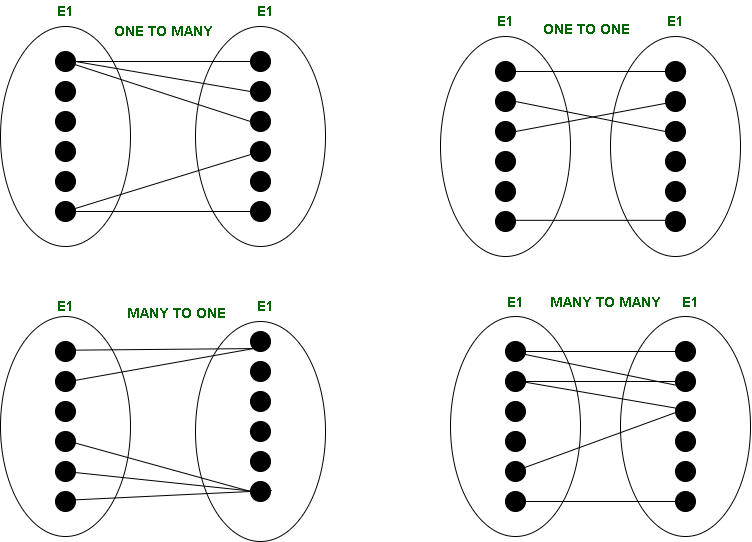
**View:** As we know, there are many tables in the database system to store data. The structure that consists of the information contained in these tables and stores data from many tables and displays them to the user is called a view. The difference between view and table is that tables are actually where data is stored and physically data is stored in tables. On the other hand, view is a virtual structure prepared for displaying the data stored in tables as desired and the data in it is connected to the tables.

**SQL Injection Attacks:** As we know, we keep a lot of data in the database and it is very important that some of this data cannot be accessed by other users. malicious people can exploit vulnerabilities in database logic for their own benefit. To prevent this, enormous security measures have been taken by database developers.

**ER Diagram:** It is of great importance to determine where and which data will be stored in order to store data in the database system. ER is used for database solution to be created in case of a problem. Thanks to this, ER is used to explain which attributes will be kept under which table, which tables will be in the link, and how the relationship between tables will be. The quality of the ER design will provide great convenience while designing the database in the future. Considering the database system to be used to solve real problems, this problem was tried to be solved systematically by using ER.



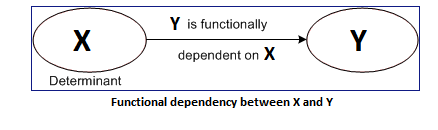
**Cardinality:** As we know, there are relations among the entities in the database. The rule used to determine how this relation will take place is called cardinality. This topic should be known in order to understand how the connection between entities will be. To give an example, let's assume that it is a student entity and a course entity. As we know, there can be a student and a student can have more than one course. In this case, the cardinality between the student and the course is one to many.



**Primary Key:** As we know, there are tables in the database. Each row in these tables keeps a record of a different event. In order to distinguish between these rows, the attributes that make each row unique should be known. The purpose of this method is that the attributes that make these rows different from each other are of great importance in terms of using the rows without mixing them up in future operations.

**Normalization:** Normalization is perhaps the most important issue in database design. The reason why we see it as the most important issue is whether the conditions necessary for the proper functioning of the database can be understood by normalization. When the ER diagram is designed, we must put it into the Relational model in accordance with the normalization requirements. If we talk about the reasons why normalization is important, thanks to normalization, unnecessary data duplication in the database is prevented. Another important issue is that data valid for more than one row should be kept in a separate entity. The reason for this is that if an update is desired, all rows will need to be updated and the risk of error will increase and optimization problems will arise. Another important point is that as it is known, a single data can be kept in a cell in the database system. The problem caused by this is consider a hospital and you need to save the patients who are in the hospital in the database. If only one data is kept in a cell, it is important how to keep all patients. Therefore, in such cases, another entity is created and the ids that make the rows of the other two entities unique are included in this entry, and thus the relation between them is defined.

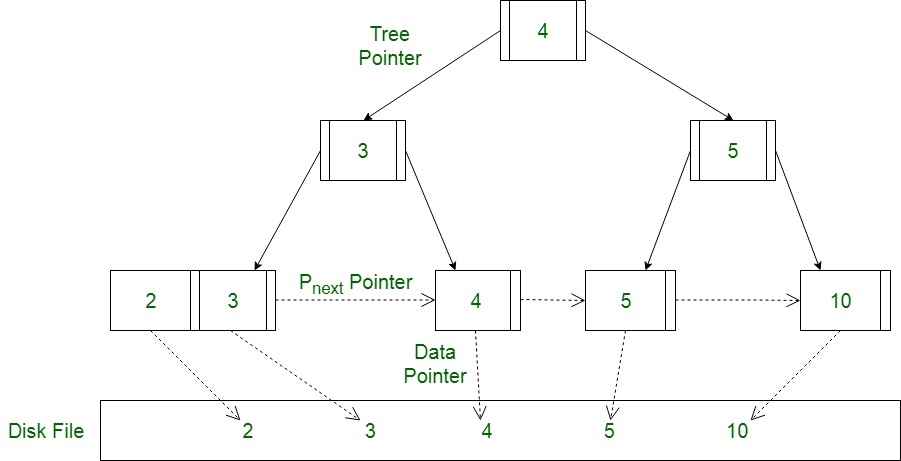
**Functional Dependency:** This title is also of great importance when designing a database. The reason why it is important is to determine which attribute is connected to which attribute, and the information in functional dependency is used. The importance of explaining the relationship between attributes is that in order to select candidate key, each attribute must have an attribute that can uniquely define each attribute, and this can be achieved by applying the rules specified under the functional dependency topic.



**Designing Desktop Application with Java and MYSQL:** There are many important steps to create a desktop application using database. To mention these sequentially, the application interface should be designed first. Thanks to this interface, the user will be able to select the desired transactions and the software in the backend section will make the request as a result of the selected transactions. In order to use the database in the backend section, the database must first be connected to the application. In order to do this, we need our username and password information that we use to access the data we have created in the database. There are special binding codes to connect the database to the desktop application, and when we add these codes to our project by editing these codes as necessary, we can access the service provided by the database. After the connection is completed, the queries that will perform the necessary operations are entered into the system and a desktop application that can respond to the calls made at the frontend can be developed. One of the important points here is to be able to take precautions against SQL attacks. It is important to use a system to prevent SQL attacks in the parts where the user needs to enter data. If you are using java to create your desktop application, you can set up a SQL attack-resistant system with the service offered by PreparedStatement.

**Indexing:** One of the important issues for the database is indexing. The reason for this is that the number of elements in the database sometimes exceeds millions, and in this case, it is really difficult to find an element in the database. In order to solve this problem, the methods mentioned under the topic of indexing are used. Generally, a table is created for indexing and this table consists of only 2 columns. The first of these columns keeps the id number of the data and the second column contains a pointer that stores the data where the data with the id kept in this first column is stored.

**B+ Tree:** As we mentioned about indexing, we called indexing to keep the ids and pointers of those ids in order to facilitate the search speed. We can see one of the most beautiful applications of indexing in B+ tree. A B + tree is similar to a binary tree in some features. The part that looks like a binary tree is listed, while ids smaller than root are kept on the left and ids larger than root are kept on the right. The biggest difference of a B + tree from a binary tree is that the B + tree does not have the data itself, but only the id of the data and the pointer of the physical location where the data is located. Leaf nodes in the B + tree are the pointers of the places where the data are located and these pointers are saved as a LinkedList. In the nodes that are the parent of the leaf nodes, the ids of the rows given these pointers are stored and among these nodes there are also pointers that show the location of the required data.



**CONCLUSION**

In this assignment, we made a brief summary of the topics we covered during the semester, such as the relational model, ER Diagram, normalization, functional dependency, indexing, and B+ tree. In addition, we tried to learn the use of Java and MySQL programs to design a desktop application that we worked on within the scope of the project assignment and we also mentioned this part in the homework.