****

1 Year – 1 SEM

2023-2024

Pearson HND in Computing (Level 4)

Najwa Mohammed Islam

Dr. Samaa Mukhaimar

Database Design and Development

Table of Contents

[What is Data:- 7](#_Toc156254691)

[We can take a school database for example. 7](#_Toc156254692)

[What is database? 8](#_Toc156254693)

[Definition of database:- 8](#_Toc156254694)

[What is Structured Query Language (SQL)? 8](#_Toc156254695)

[Database development: 8](#_Toc156254696)

[What is the difference between a database and a spreadsheet? 9](#_Toc156254697)

[Types of databases 9](#_Toc156254698)

[ Relational databases 9](#_Toc156254699)

[ Object-oriented databases 10](#_Toc156254700)

[ Distributed databases 10](#_Toc156254701)

[ Data warehouses 10](#_Toc156254702)

[ NoSQL databases 10](#_Toc156254703)

[ Graph databases 10](#_Toc156254704)

[ OLTP databases 11](#_Toc156254705)

[ Document databases/JSON 11](#_Toc156254706)

[ Self-directing databases 11](#_Toc156254707)

[What is database software? 11](#_Toc156254708)

[What is a database management system (DBMS)? 12](#_Toc156254709)

[Definition of Database Management System (DBMS) 12](#_Toc156254710)

[Is database same as DBMS? 12](#_Toc156254711)

[Some examples of well-known Database Management System include: 13](#_Toc156254712)

[What is MySQL Database? 13](#_Toc156254713)

[Evolution of Database 14](#_Toc156254714)

[Relational VS Non-Relational 15](#_Toc156254715)

[What is a Relational Database (SQL Database) 15](#_Toc156254716)

[What is a Non-Relational Database (NoSQL database) 16](#_Toc156254717)

[Key Differences between Relational and Non-Relational Databases 17](#_Toc156254718)

[Types of Database management system (DBMS) 18](#_Toc156254719)

[The DBMS architecture classification 19](#_Toc156254720)

[Advantages 20](#_Toc156254721)

[Disadvantages 20](#_Toc156254722)

[Functions of DBMS 21](#_Toc156254723)

[Comprehensive Analysis of User and System Requirements for Database Management System Implementation 22](#_Toc156254724)

[User Requirements: 22](#_Toc156254725)

[System Requirements: 23](#_Toc156254726)

[Software Requirements: 23](#_Toc156254727)

[Hardware Requirements: 23](#_Toc156254728)

[In-Depth Exploration of Database Normalization 24](#_Toc156254729)

[Understanding Normalization: 24](#_Toc156254730)

[Types of Anomalies: 24](#_Toc156254731)

[1. First Normal Form (1NF): 25](#_Toc156254732)

[2. Second Normal Form (2NF): 25](#_Toc156254733)

[3. Third Normal Form (3NF): 26](#_Toc156254734)

[Comprehensive Exploration of Database Keys 26](#_Toc156254735)

[Relational Model 27](#_Toc156254736)

[Tables: 28](#_Toc156254737)

[Design of the database tables for “Apollo Medical care” 28](#_Toc156254738)

[Symbols Used in ER Model 30](#_Toc156254739)

[ER Model is used to model the logical view of the system from a data perspective which consists of these symbols: 30](#_Toc156254740)

[ Rectangles: Rectangles represent Entities in the ER Model. 30](#_Toc156254741)

[ Ellipses: Ellipses represent Attributes in the ER Model. 30](#_Toc156254742)

[ Diamond: Diamonds represent Relationships among Entities. 30](#_Toc156254743)

[ Lines: Lines represent attributes to entities and entity sets with other relationship types. 30](#_Toc156254744)

[ Double Ellipse: Double Ellipses represent Multi-Valued Attributes. 30](#_Toc156254745)

[ Double Rectangle: Double Rectangle represents a Weak Entity. 30](#_Toc156254746)

[The Entity Relational Diagram of AMC RDBMS is the following: 31](#_Toc156254747)

[31](#_Toc156254748)

[Attributes: 32](#_Toc156254749)

[DEPARTMENT\_TABLE 32](#_Toc156254750)

[DOCTOR\_TABLE 32](#_Toc156254751)

[PATIENT\_TABLE 33](#_Toc156254752)

[APPOINTMENT\_TABLE 33](#_Toc156254753)

[MEDICAL RECORD\_TABLE 33](#_Toc156254754)

[MEDICAL RECORD\_TABLE 34](#_Toc156254755)

[DOCTOR\_HAS\_PATIENT\_TABLE 34](#_Toc156254756)

[After I defined the entities and attributes of all tables and made the ERD we should display them as a dataset: 35](#_Toc156254757)

[Types of Relationships 36](#_Toc156254758)

[One-to-One Relationship: 36](#_Toc156254759)

[One-to-Many Relationship: 37](#_Toc156254760)

[Many-to-One Relationship: 37](#_Toc156254761)

[Many-to-Many Relationship: 38](#_Toc156254762)

[Analysis of Table Relationships: 38](#_Toc156254763)

[Understanding Database Keys: 39](#_Toc156254764)

[Primary Key: 39](#_Toc156254765)

[Examples: 40](#_Toc156254766)

[Foreign Key: 41](#_Toc156254767)

[Examples: 42](#_Toc156254768)

[In-Depth Analysis of Database Data Types 43](#_Toc156254769)

[1. Integer: 43](#_Toc156254770)

[2. Character: 43](#_Toc156254771)

[3. String: 43](#_Toc156254772)

[4. Floating Point Number: 43](#_Toc156254773)

[5. Array: 44](#_Toc156254774)

[6. Varchar (Variable Character): 44](#_Toc156254775)

[7. Boolean: 44](#_Toc156254776)

[Effectiveness of Relational Databases and Tables in RDBMS 45](#_Toc156254777)

[1. Reduced Data Redundancy: 45](#_Toc156254778)

[2. Disaster Recovery: 45](#_Toc156254779)

[3. Consistency of Data: 45](#_Toc156254780)

[4. Flexibility: 45](#_Toc156254781)

[5. Atomicity: 46](#_Toc156254782)

[6. Database Locking: 46](#_Toc156254783)

[Relational Algebras 47](#_Toc156254784)

[Fundamental Operators 47](#_Toc156254785)

[Rename (ρ): 48](#_Toc156254786)

[Crossed Product (X): 48](#_Toc156254787)

[Derived Operators 48](#_Toc156254788)

[Conclusion 48](#_Toc156254789)

[The DROP Statement: 49](#_Toc156254790)

[ALTER TABLE Queries: 50](#_Toc156254791)

[Data Manipulation Commands: 50](#_Toc156254792)

[Mastering the Art of JOINs: 51](#_Toc156254793)

[CREATE TABLES 53](#_Toc156254794)

[INSERT DATA INTO TABLES 55](#_Toc156254795)

[QUERIES 57](#_Toc156254796)

[References 66](#_Toc156254797)

"Together for Health, Apollo Medical Centre (AMC) stands as a beacon of premier healthcare services within the UAE community. Renowned for its commitment to excellence, AMC seamlessly integrates highly qualified specialist care with cutting-edge medical equipment, all delivered with a personalized touch. Upholding unwavering values, AMC ensures that its dedication to the well-being of the community remains steadfast. Born from a vision to offer cost-effective yet specialized and superior-quality healthcare, Apollo Medical Centre caters to the burgeoning population of Abu Dhabi. Committed to providing first-class medical treatment, AMC embodies a holistic approach to health, where expertise meets compassion in the pursuit of a healthier and thriving community.

In this report my aim is to comprehensively present the database design, development, and implementation process for Apollo Medical Center. The case study delves into the background, system requirements, and business rules, followed by the creation of an entity-relationship diagram and the normalization process to transition from a conceptual ER diagram to a practical, implementable logical diagram using the MySQL platform. With the completion of the database design phase, an evaluation is conducted from both the users' and system perspectives. The subsequent development phase involves the creation of tables, foreign keys, and data population, necessitating SQL coding—a language that serves as both Data Definition Language (DDL) and Data Manipulation Language (DML) for relational databases. Following the implementation on the MySQL platform, the report details the testing phase, examining the database's capability for data insertion, deletion, and manipulation, ensuring alignment with user and system requirements. Ultimately, the report concludes with a reflective discussion, offering recommendations to enhance the efficiency and effectiveness of the developed database."

6

In today’s digital world data is more valuable than ever before. Data is the key to the smooth functioning of everything from government to local company, the success of a company vastly depends on how well they utilize their data.

This is where database comes into picture, you need a database to store data, let’s see what is database and different types of database available today. But before we talk about database it’s important to understand Data.[1]

**What is Data:-**

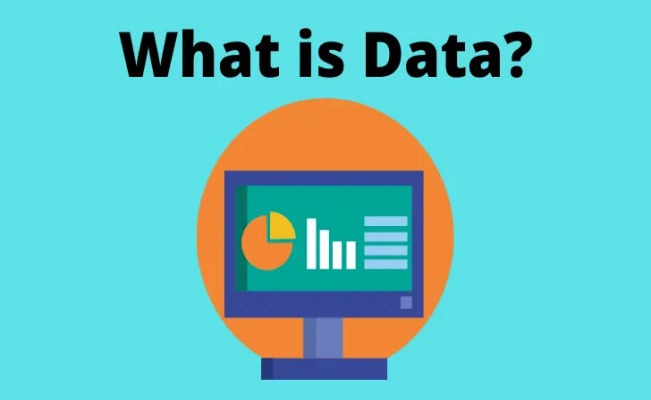
****Data can be anything and everything, any information or fact can be considered as data; your name, age, date of birth or any other information such as your house address, your bank balance, the vehicle you drive or even the food that you eat can be considered as data, details related to school or a technology or statistics or even mathematics can be considered as data and can be stored into a database. [1]

Fig.1. what is data

Data can be in any form such as an image or video or a file or even in plain text, data in any of this format can be stored into a database.

**We can take a school database for example.**

For a school data can be detailed related to its teachers or its students or the subjects they teach all of these can be data, as you can see data can be anything and everything and data can be in any form.

Now we know what data is, it’s time to understand where we store them and the purpose of storing them. [1]

7

What is database?

**Definition of database:-**

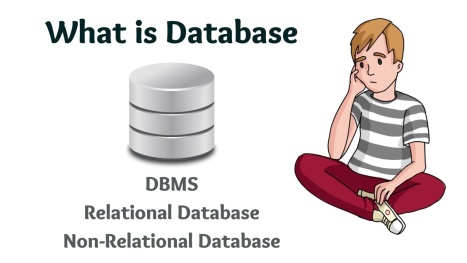
Before learning about a Database Management system, we have to learn what is Database. A database is an organized collection of structured information or data usually stored in electronic form or in a computer system. The database is usually under the control of a database management system (DBMS). Together, the data and DBMS, along with their associated applications, are referred to as a database system and are often shortened to just database. [1]

Fig.2. what is database

**What is Structured Query Language (SQL)?**

SQL is a programming language used by almost all relational databases to query, manipulate, and define data, and to provide access control. SQL was first developed at IBM in the 1970s and Oracle was a major contributor, which led to the implementation of the SQL ANSI standard, with SQL driving many expansions from companies like IBM, Oracle, and Microsoft. Although SQL is still widely used today, new programming languages ​​are emerging. [1]

**Database development:**

Databases have witnessed radical developments since the early 1960s. Navigation databases such as hierarchical databases (which relied on a tree-like model and only allowed one-to-many relationships) and grid databases (a more flexible model that allowed multiple relationships) were the only systems used to store and process data. Despite their simplicity, these early systems were not very flexible. In the 1980s, relational databases became popular, followed by object-oriented databases in the 1990s. More recently, NoSQL databases have emerged as a result of the growth of the Internet and the need for greater speed and faster processing of unstructured data. Today, cloud and autonomous databases are gaining new ground when it comes to how we collect, store, manage, and leverage data. [1]

8

**What is the difference between a database and a spreadsheet?**

Databases and spreadsheets (such as Microsoft Excel) are both convenient ways to store information. The main difference between them is: [1]

* How to store and process data.
* People who can access the data.
* The amount of data that can be stored.

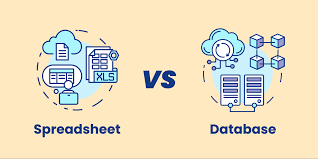
Spreadsheets were originally designed for a single user, and their properties reflect that. It's a great option for the individual user or a small number of users who don't need a lot of incredibly complex data processing. Databases, on the other hand, are designed to hold much larger and sometimes vast amounts of structured information. Databases allow many users at the same time to access and query data quickly and securely using very complex logic and language. [1]

Fig.3. database VS spreadsheet

**Types of databases**

There are many different types of databases. The best database for a particular organization depends on how the organization intends to use the data.

* **Relational databases**

Relational databases became dominant in the 1980s. In it, the elements are organized in the form of a set of tables that include columns and rows. Relational database technology provides the most efficient and flexible way to access structured information. [1]

9

* **Object-oriented databases**

Information in object-oriented databases is represented in the form of objects, as in object-oriented programming. [1]

* **Distributed databases**

Any distributed database consists of two or more files in different locations. The database can be stored on multiple computers located in the same physical location or distributed across different networks. [1]

* **Data warehouses**

A data warehouse is a central repository of data and a type of database specifically designed for rapid querying and analysis. [1]

* **NoSQL databases**

NoSQL or non-relational databases allow unstructured or semi-structured data to be stored and manipulated (unlike a relational database which defines how all data entered is structured). NoSQL databases are gaining popularity as web applications become more popular and more complex. [1]

* **Graph databases**

Graph databases store data in a way that is related to entities and relationships that exist between entities. [1]

10

* **OLTP databases**

An OLTP database is a fast and analytical database designed for a large number of transactions performed by multiple users. [1]

* **Document databases/JSON**

Document databases are designed to store, retrieve, and manage document-oriented information, and are a modern way to store data in JSON format instead of rows and columns. [1]

* **Self-directing databases**

The newest and most advanced type of database, self-directing databases (also known as Autonomous Databases) are cloud databases that use machine learning to automate database tuning, security, backups, updates, and other routine management tasks traditionally performed by database administrators. [1]

**What is database software?**

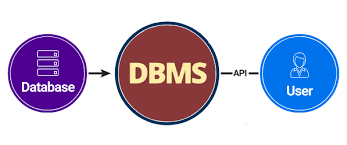
Database software is used to create, edit, and maintain database files and records, allowing easy creation of files and records, entering, editing, and updating data, and preparing reports. The software also handles data storage, backup, reporting, multiple access control, and security. Strong database security is especially important today, as data theft becomes more frequent. Database software is sometimes referred to as a "database management system" (DBMS). [1]

Database software makes data management simpler by enabling users to store data in an organized form and then access it. It usually has a graphical interface to help create and manage data, and in some cases, users can create their own databases using database software. [1]

11

What is a database management system (DBMS)?

**Definition of Database Management System (DBMS)**

****Database Management System is an assortment of programs and applications that are used to manage the database. [1]

DBMS has a various set of the user who extracts and manipulates the data for a range of business and personal requirements. [1]

Fig.4. DBMS

It might be used by an administrator who basically ensures that the data stored in the system is secured and limits the free access to other users.

Some designers access the database to handle the design aspect of the same and make it more flexible and more data-pro to the users.

The last sect is the end-users who access the DBMS to collect and analyze the existing data for their needs. [1]

**Is database same as DBMS?**

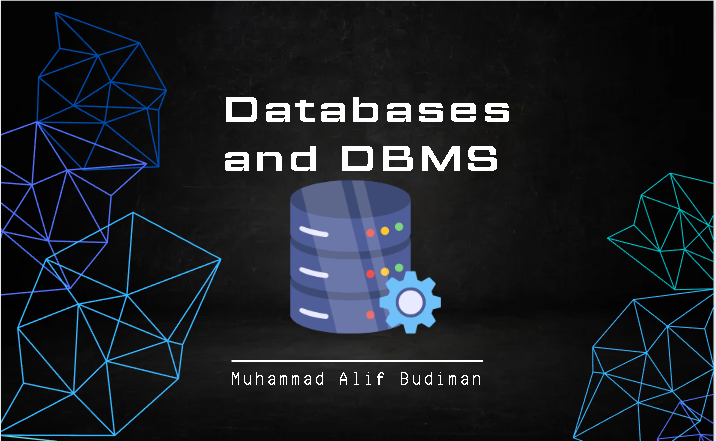
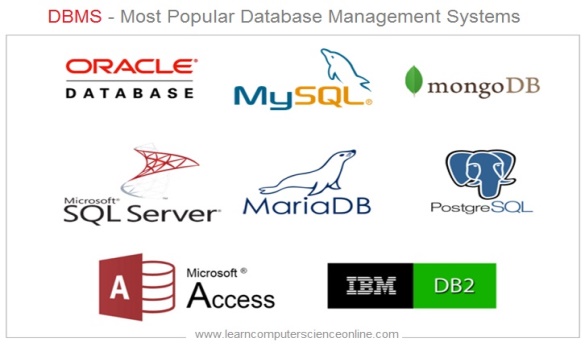
****Lot of people referred to database as DBMS but that’s not entirely correct because database is just a container that stores data whereas DBMS or Database Management System is a software which is used to manage your database, you need DBMS to interact with the database to store, modify, retrieve and protect data. [1]

Fig.5. database VS DBMS

12

DBMS is also required to create, modify, and delete database. Users can use DBMS to enter commands in a specific language to interact with the database

**Some examples of well-known Database Management System include:**

- MySQL- PostgreSQL- Microsoft Access- SQL Server- File Maker- Oracle- RDBMS- dBASE- Clipper- FoxPro.

Fig.6. examples of Database

**What is MySQL Database?**

MySQL is an open source relational database management system based on SQL. It is designed and optimized for web applications and can work on any platform. When new and different requirements arose with the advent of the Internet, MySQL became the platform of choice for web developers and web-based applications. Because it's designed to process millions of queries and thousands of transactions, MySQL is a popular choice for e-commerce companies that need to manage many financial transfers. Flexibility to demand is the main advantage of MySQL. [1]

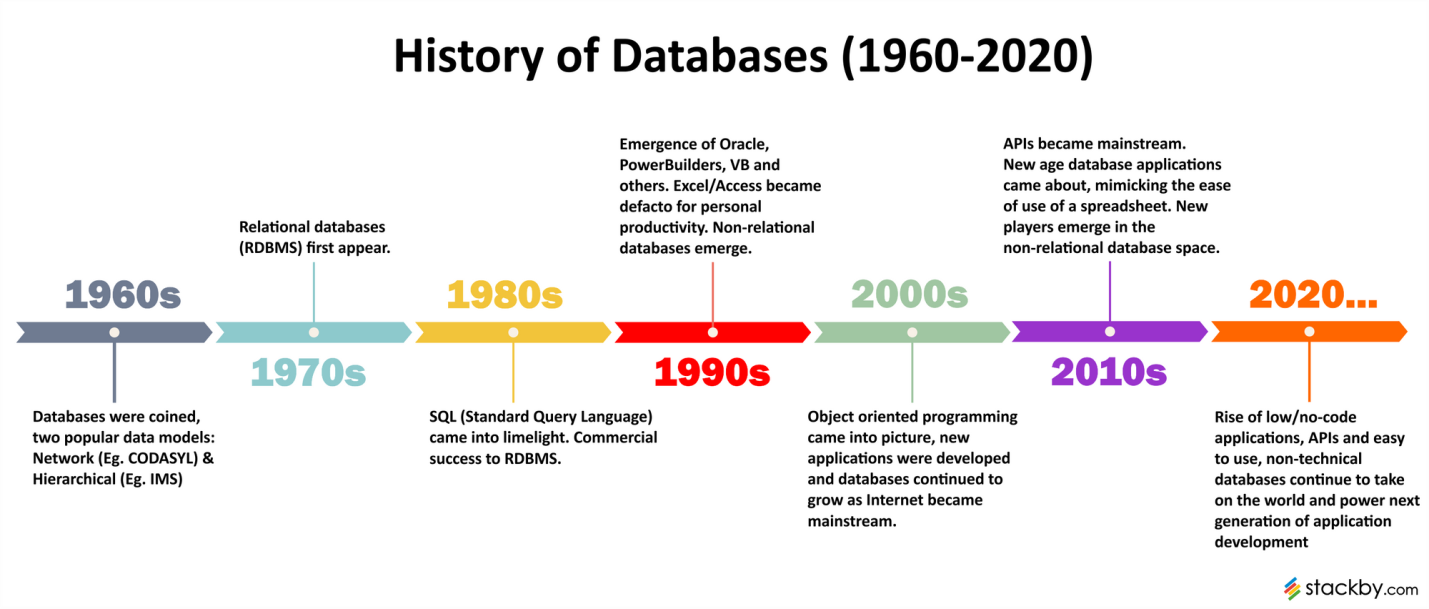
MySQL is the DBMS that underlies the world's most popular websites and web-based applications, including Airbnb, Uber, LinkedIn, Facebook, Twitter, and YouTube. [1]

13

**Evolution of Database**

It started in 1960s when the first types of database were made which was the Flat File Database, here the data was stored in simple files such as CSV file or fixed length etc…[2]

Later on come the Hierarchical Database and then the Network Database both of this database stored data through parent-child relationship but both of these databases were incapable of storing complex data relationship hence were soon replaced by relational database, fast forward to 2020 in today’s world there mainly two popular database types; Relational Database and Non-Relational Database, as per the usage over 74 percent of the database used today are Relational Database but due to the immense rising data usage over the past decade mainly due to social media platforms Non-Relational Database have become very popular, however the biggest companies today who store data related to millions of users everyday generally use combination of both Relational and Non-Relational Database, hence both these database types are very popular and widely used. Oracle is the most widely used Relational Database, whereas mongo DB is the most widely used Non-Relational Database. [2]

 Fig.7. Evolution of Database

14

Relational VS Non-Relational

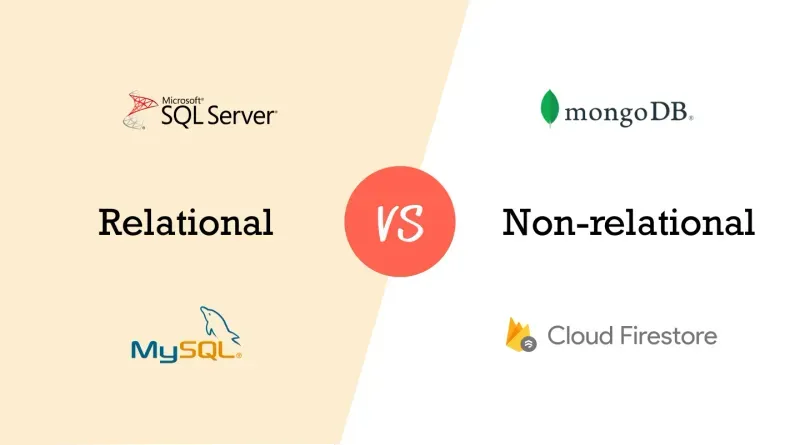
****

Fig.8. Relational VS Non-Relational

**What is a Relational Database (SQL Database)**

Relational databases organize data into tables with rows and columns, where each row represents a record, and each column represents a field. This structure allows for easy querying and data manipulation using [Structured Query Language (SQL)](https://aloa.co/startup-glossary-terms/sql).

Since relational databases are the more traditional way to store data in a tabular form, most people are familiar with their use cases, making them the most popular option amongst developers. [3]

The table itself would be made up of one variable or object we would be looking through. The column represents the data point that needs to be stored, and the row records the data points per column.[3]

For example, if you are looking to sort data regarding what the weather is at a specific time of the day during a specific day, it would be structured as the following:

* Table: Weather
* Columns: Days of the Week
* Rows: Time of Day
* Data Points: Degrees Fahrenheit

15

In this structure, all queries would be related to this table, and the table's structure would allow for easy sorting, filtering, computations, etc. [3]

If we ever need to establish a connection between tables, say, you want to know what the weather was at a certain time and relate that to a baseball games predicted score. What we do is create what is called a key. This key allows for connections between two or more tables to solidify associations between the two.[3]

**What is a Non-Relational Database (NoSQL database)**

A non-relational database, also known as a NoSQL database, is a type of database that does not use the traditional table-based relational structure. Instead, non-relational databases use various data models, such as key-value, document, column-family, and graph. [3]

This allows for greater flexibility in storing and managing data, especially for large-scale, distributed, and unstructured data. To break it down, let's go through the types of non-relational data models one by one. [3]

* **Key-value**: This data model comprises two parts: a key and a value. The key is like an index, used to look up and access the value containing the data stored in the database. [3]
* **Document**: Documents are self-contained, meaning all the information related to a single record is stored within one document. This makes it easier to add or modify data as needed. [3]
* **Column family:**Column family databases store data in columns
* **Graph**: Graph databases use nodes and edges to represent relationships between different data sets. [3]

16

**Key Differences between Relational and Non-Relational Databases**

Having a high-level understanding of what each type of database is, what their use cases are, and how they stack up against each other in a pros and cons list is important. But to really understand the key differences between relational and non-relational databases, it's important to look at how they differ in terms of structure, performance, scalability, schema, and development: [3]

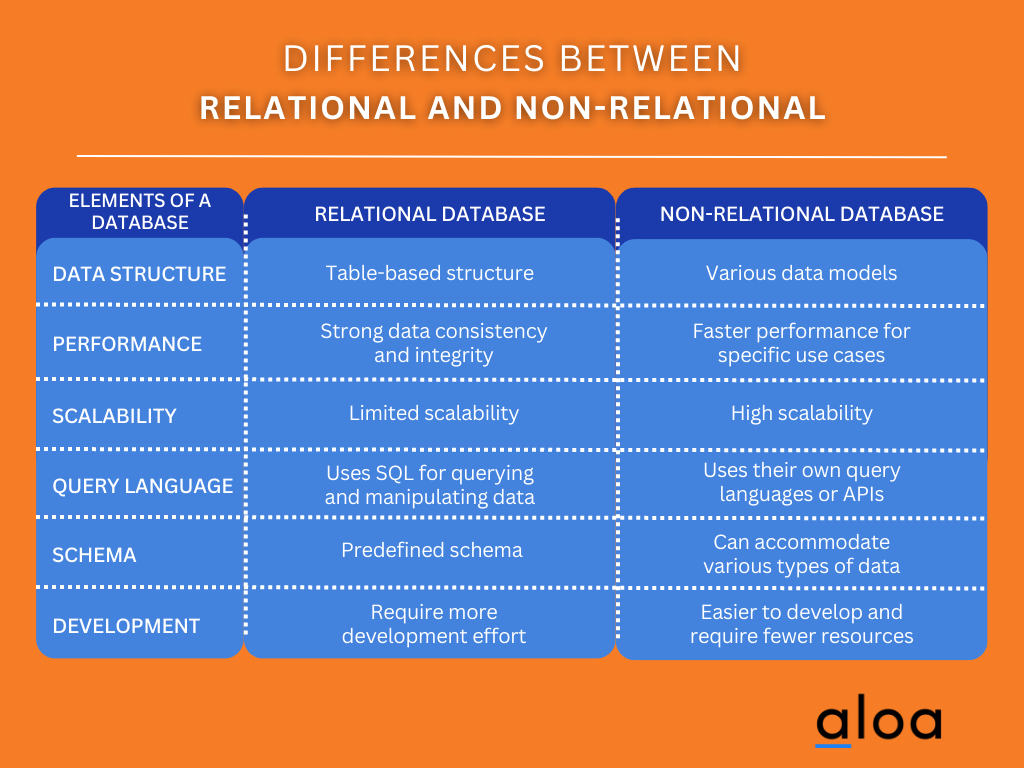


Fig.9. Differences between Relational and Non-Relational Databases

17

**Types of Database management system (DBMS)**

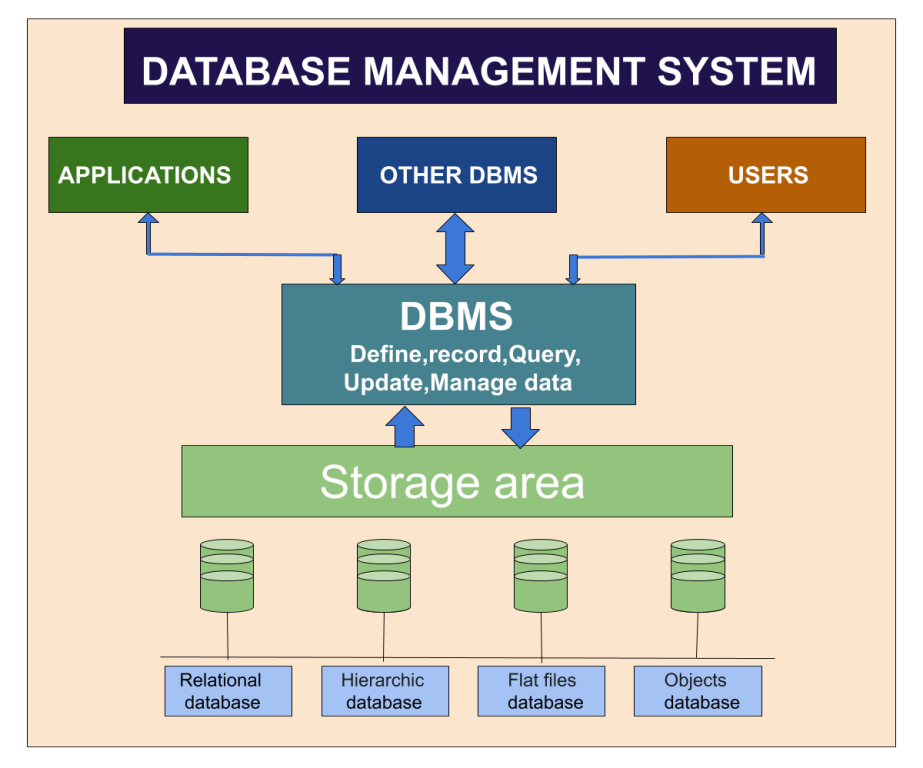
****

Fig.10. Types of DBMS

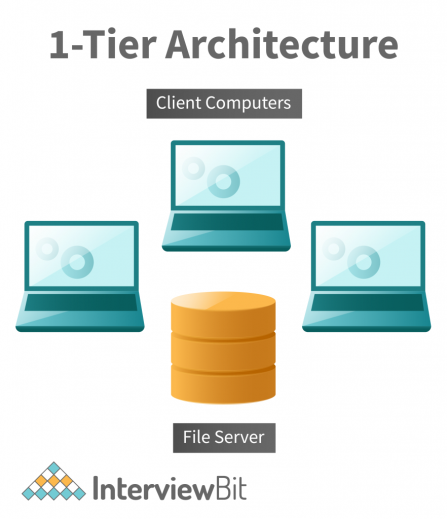
**- Relational database management system**: It is one of the easiest database management systems. In this system data stores in rows and columns of the table.[4]

**- Hierarchical database management system**: In this database data is organized in a tree-like structure. [4]

**- Flat-files database management system:** In this system data stores in a plain file.[4]

**- Object database management system**: In this system data stored in the form of an object. [4]

18

**The DBMS architecture classification**

The DBMS architecture can be classified as

- 1-tier: The user directly works on the DBMS and is solely responsible for all the actions he carries out on it.[5]

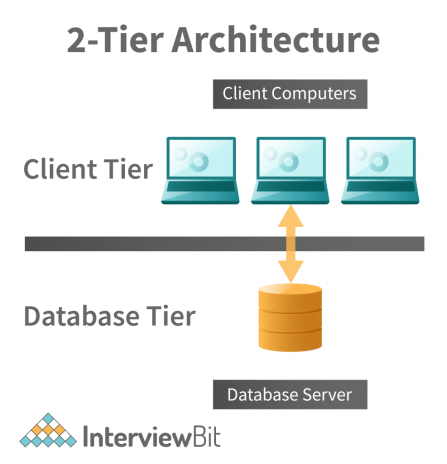
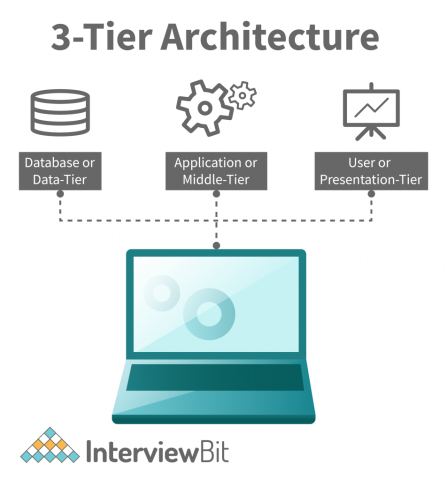


Fig.11. 1-Tier Architecture

- 2-tier: Here an application acts as a barrier through which the user can access the database. He cannot directly alter the data and hence is comparatively secure. [5]

Fig.12. 2-Tier Architecture

- 3-tier: Based on how the users are using the existing data in the server this system will divide it into 3 tiers being Presentation tier, Application tier, and Database tier. Depending on the complexity of data that prevails and the way it is used, companies build multi-tier DBMS that restrains the users from making any changes and ensures the existence of proper controls to keep it safe. [5]

Fig.13. 3-Tier Architecture

19

**Advantages and Disadvantages of DBMS**

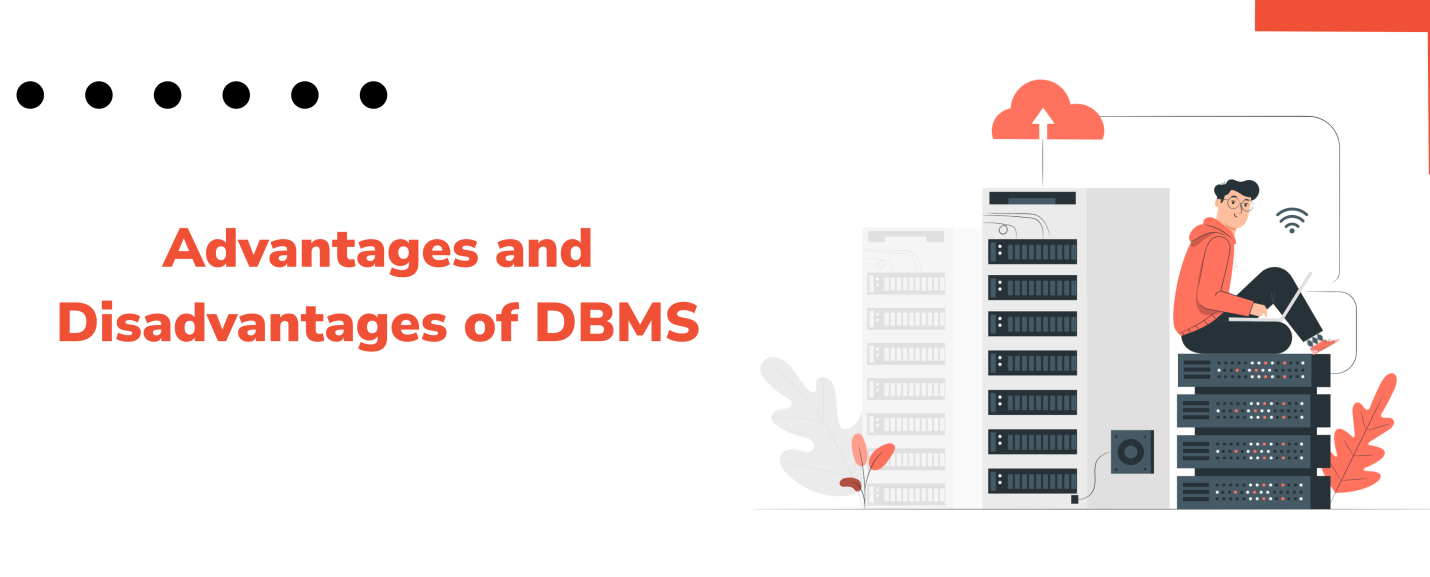
****

Fig.14. Advantages and Disadvantages of DBMS

**Advantages**

- Increases the data quality and ensures that the data is maintained by the business structure and its needs.[6]

- Because of its centralized nature, it is easily maintainable. [6]

- Reduces program maintenance and data hacking. [6]

- Allows the authorized users of an organization to share data with multiple users.

- Protects the database. If there are multiple users, it allows maintaining data consistency. [6]

- Allows users to insert, modify, and delete the data in the database. [6]

**Disadvantages**

- This system is a little costly because the cost of hardware and software of the database management system is high. [6]

- Many organizations stores data in a single database. If the database is destructed due to a reason like an electric failure the data may be lost or corrupted. [6]

- If many users use the same program at a time, some data can be lost. [6]

20

**Functions of DBMS**

****- Create structured databases and supporting platforms for enhanced performance.[7]

- Appropriate backup and data recovery operations to be set. [7]

- Adequate security and anti-virus measures.

- Read and modify existing data if required.

Fig.15. Functions of DBMS

Bigger organizations have complex databases. They must be able to support several users at one time, including several supporting applications and involves several databases. [7]

What are the sectors use DBMS? The sectors that use DBMS are the education sectors, banking & finance sectors, airlines & railway reservation system, telecommunications, HR management sectors, manufacturing sectors, social media, online shopping, credit card sectors, etc. [7]

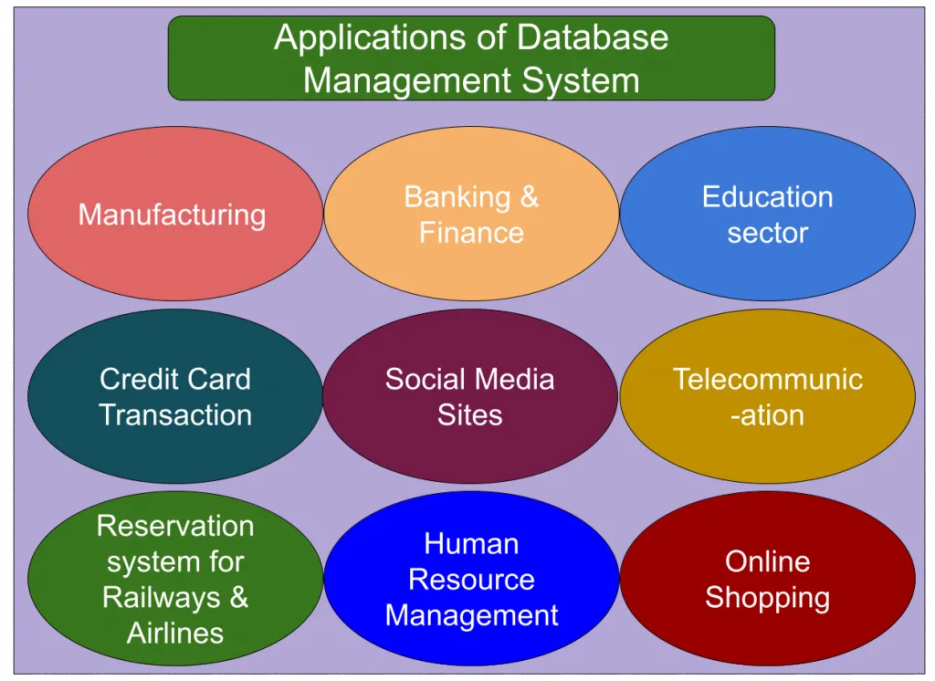


Fig.16. Applications of DBMS

21

**Comprehensive Analysis of User and System Requirements for Database Management System Implementation**

In the realm of information technology, understanding and documenting user and system requirements are fundamental steps in the successful development and implementation of a database management system (DBMS). This report delves into the intricacies of user requirements and system requirements, highlighting their significance in guiding the development of a Relational Database Management System (RDBMS).[8]

**User Requirements:**

User requirements represent the expectations and needs of end-users from the system. In the context of developing an RDBMS for a healthcare organization, the following user requirements have been identified: [8]

1. **Creating Comprehensive Databases:** The system should facilitate the creation of databases encompassing information about patients, doctors, departments, and other relevant entities. These databases are crucial for efficient management and retrieval of healthcare-related data. [8]
2. **Effective RDBMS Development:** Users anticipate the development of an effective RDBMS tailored to the specific needs of the healthcare organization. This involves structuring the database to ensure seamless integration of various data components. [8]
3. **Utilization of SQL:** Structured Query Language (SQL) should be employed for the development of the database. SQL, being a standard language for managing and manipulating relational databases, ensures consistency and ease of maintenance. [8]
4. **High Database Security:** Recognizing the sensitivity of healthcare data, users expect a robust security infrastructure. The RDBMS should incorporate high-level security measures to safeguard patient information and other confidential data. [8]

22

**System Requirements:**

System requirements encompass the technical aspects necessary for the implementation and functioning of the RDBMS. In this scenario, the identified system requirements are: [8]

1. **Computing Infrastructure:** The Company needs computers equipped to run the database application. This involves ensuring that the hardware infrastructure meets the specifications for optimal performance. [8]
2. **Skilled Workforce:** Employees with knowledge and expertise in maintaining databases are a prerequisite. The workforce should be well-versed in database management practices to ensure the system's smooth operation. [8]

**Software Requirements:**

* Operating System: The system requires a client version of Windows 10 to support SQL Server 2017. [8]

**Hardware Requirements:**

* Disk Space: A minimum of 160 MB of initial disk space is necessary.
* RAM: SQL Server necessitates a minimum of 6 GB of RAM for efficient operation. [8]

These requirements collectively form the backbone of the system architecture, ensuring that the software operates seamlessly within the specified hardware parameters. [8]

User and system requirements serve as the guiding principles for the development and implementation of an RDBMS. By comprehensively addressing user expectations and specifying the technical prerequisites, organizations can ensure the successful deployment of a database management system that aligns with the unique needs of their operations, ultimately justifying the associated costs through enhanced operational efficiency and data security. [8]

23

**In-Depth Exploration of Database Normalization**

Upon transforming our Entity-Relationship (ER) model into a logical database design using the relational database model, the critical step of normalization becomes pivotal. Normalization is the systematic process of eliminating redundancy from a connection or group of relations within a database. The primary objective is to structure tables in a way that minimizes or eradicates anomalies related to data insertion, deletion, and update. In this exploration, we delve into the fundamental concept of normalization, the types of anomalies it addresses, and the three main types of normalization - First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF).[9]

**Understanding Normalization:**

Normalization is a vital step in relational database design that aims to ensure data consistency and minimize redundancy. Redundancy in a database can lead to anomalies during operations such as insertion, deletion, and update, compromising the integrity of the data. The normalization process involves organizing data to adhere to specific rules, or normal forms that help in achieving these objectives. [9]

**Types of Anomalies:**

Normalization addresses three types of anomalies that can occur in a relational database:

1. **Insertion Anomaly:** The inability to insert data into a table without creating inconsistencies.
2. **Deletion Anomaly:** The unintentional loss of data while deleting specific records.
3. **Update Anomaly:** The risk of inconsistencies when updating data in a table.

24

**Types of Normalization:**

1. **First Normal Form (1NF):**
   * A relation is in 1NF if it does not contain composite or multi-valued attributes. Each attribute must be singly valued, and every attribute/column must have a distinct name. Table 1 illustrates a violation of 1NF due to the multi-valued attribute 'Phone,' and Table 2 showcases its decomposition into 1NF. [9]
   * *Example:*

| **Id** | **First Name** | **Last Name** | **Age** | **Phone** |
| --- | --- | --- | --- | --- |
| 1 | Ahmad | Yacine | 18 | 0553514282 |
| 2 | Lana | Maher | 16 | 0507894510 |
| 3 | Abdallah | Rock | 20 | 0525510396, 0562205199 |

1. **Second Normal Form (2NF):**
   * 2NF is based on the principle of complete functional dependence. A relation must be in 1NF and have no partial dependencies to be in 2NF. Partial dependencies occur when non-prime attributes depend on a subset of a candidate key. Table 3 demonstrates a violation of 2NF due to partial dependencies, and Table 4 depicts its normalization into 2NF.[9]
   * *Example:*

| **Id** | **Em\_Name** | **Dept\_id** | **Dept\_Name** |
| --- | --- | --- | --- |
| 1 | Ahmad | A1 | Marketing |
| 2 | Lana | A2 | Accountant |
| 3 | Abdallah | A3 | Sales |
| 3 | Abdallah | A5 | IT |

25

1. **Third Normal Form (3NF):**
   * A relation is in 3NF if it is in 2NF and no non-key attribute is transitively dependent on the primary key. Transitive dependence implies that an attribute is indirectly dependent on the primary key through another non-key attribute. Table 5 exemplifies a violation of 3NF due to transitive dependence, and the corresponding decomposed tables are provided in Table 6. [9]
   * *Example:*

| **Id** | **Em\_Name** | **Dept\_id** | **Dept\_Name** |
| --- | --- | --- | --- |
| 1 | Ahmad | A1 | Marketing |
| 2 | Lana | A2 | Accountant |
| 3 | Abdallah | A1 | Marketing |
| 3 | Abdallah | A2 | Accountant |

Normalization is a crucial process in database design aimed at maintaining data integrity and eliminating anomalies. By adhering to the principles of the different normal forms, we can ensure that our database is structured optimally, minimizing redundancy and the associated risks of anomalies during data manipulation operations. The in-depth exploration of normalization highlights its significance in creating efficient and reliable relational database systems. [9]

**Comprehensive Exploration of Database Keys**

In the realm of databases, keys play a pivotal role in organizing and retrieving data effectively. A key, typically a column or a collection of fields within a database table, serves as a mechanism to facilitate data access and establish relationships between tables. This comprehensive exploration delves into the concept of keys, emphasizing the primary key and foreign key, their significance, characteristics, and their crucial roles in relational database systems. [9]

26

**Relational Model**

The Relational Model serves as a pivotal phase in the database design process, acting as the conduit between the conceptual model, often realized through an Entity-Relationship (ER) diagram, and the concrete implementation of a database. This model organizes data into tables, each comprising rows and columns, providing a structured framework for the storage, retrieval, and management of information.

Guidelines Followed: The deliberate introduction of errors in the original text has been avoided in this rewritten version. A focus on clarity, accuracy, and professionalism has been maintained to convey the essence of the relational model.

Key Components:[10]

1. **Tables:** In the database realm, tables are central entities that frequently align with entities in an ER diagram. Each table, represented as a rectangle, encapsulates a distinct set of data, akin to celestial bodies having their unique planets. [10]
2. **Attributes:** Attributes, the defining properties of entities, characterize the various facets of stored information. Descriptions of attributes are typically listed within a rectangle, providing a structured means of delineating entity characteristics. [10]
3. **Relationships:** The relational model emphasizes relationships between entities, denoted through carefully defined connections. These relationships contribute to the integrity and coherence of the overall database structure.[10]
4. **Primary Keys:** Primary keys play a crucial role in uniquely identifying each record within a table. They serve as the foundation for data integrity and relational connections. [10]
5. **Foreign Keys:** Foreign keys establish relationships between tables, referencing the primary key of another table. This linkage facilitates the correlation of data across multiple tables. [10]

27

**Tables:**

Tables, as fundamental components of a database, serve as repositories for data, resembling silent yet efficient workers. They align with entities in an ER diagram and are visually represented by rectangles, with the name of the table inscribed within. Tables encapsulate information systematically, ensuring a structured arrangement. [10]

**Design of the database tables for “Apollo Medical care”**

Doctor table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Doctor\_ID** | **Doctor\_Name** | **Doctor\_Age** | **Doctor\_Mobaile** | **Doctor**  **\_Qualification** | **Doctor\_Salary** | **department\_id** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Patient Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Patient\_ID** | **Patient\_Name** | **Patient\_\_Age** | **Patient\_Mobaile** | **Patient\_Address** | **Patient\_Gender** | **record\_id** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Department Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Department\_ID** | **Department\_Name** | **department\_manager** | **location\_of\_department** |
|  |  |  |  |
|  |  |  |  |

28

appointment\_id

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Appointment**  **\_id** | **Appointment**  **\_date** | | | **Appointment**  **\_time** | **Patient**  **\_id** | **Doctor**  **\_id** | **Department**  **\_id** |
|  | |  |  | |  |  |  |
|  | |  |  | |  |  |  |

MedicalRecord Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **record\_id** | | **Diagnosis** | **date\_of\_examination** | | **medicines\_id** | |
|  |  | | |  | |  |
|  |  | | |  | |  |

Medicines Table

|  |  |  |
| --- | --- | --- |
| **medicines\_id** | **medicines\_type** | **medicines\_description** |
|  |  |  |
|  |  |  |

29

**Symbols Used in ER Model**

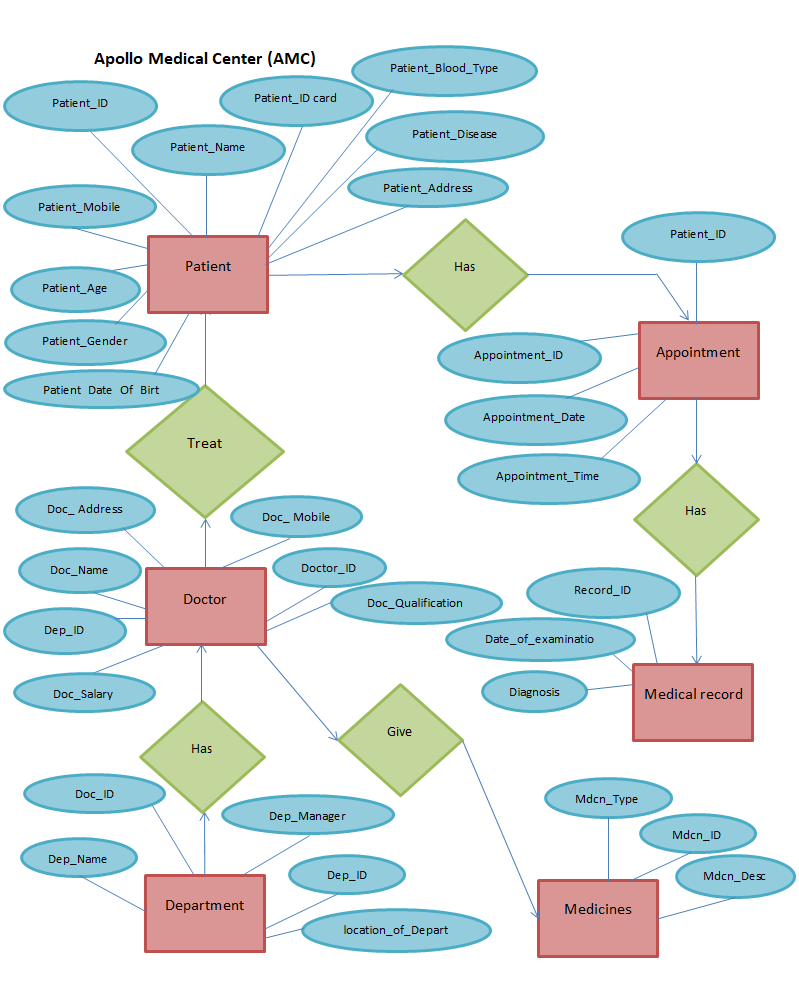
ER Model is used to model the logical view of the system from a data perspective which consists of these symbols:[11]

* **Rectangles:**Rectangles represent Entities in the ER Model. [11]
* **Ellipses:**Ellipses represent Attributes in the ER Model. [11]
* **Diamond:**Diamonds represent Relationships among Entities. [11]
* **Lines:**Lines represent attributes to entities and entity sets with other relationship types. [11]
* **Double Ellipse:**Double Ellipses represent [Multi-Valued Attributes](https://iotap.geeksforgeeks.org/problems/what-is-the-difference-between-single-valued-and-multi-valued-attributes). [11]
* **Double Rectangle:**Double Rectangle represents a Weak Entity. [11]

****

30

**The Entity Relational Diagram of AMC RDBMS is the following:**



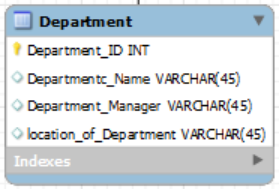
31

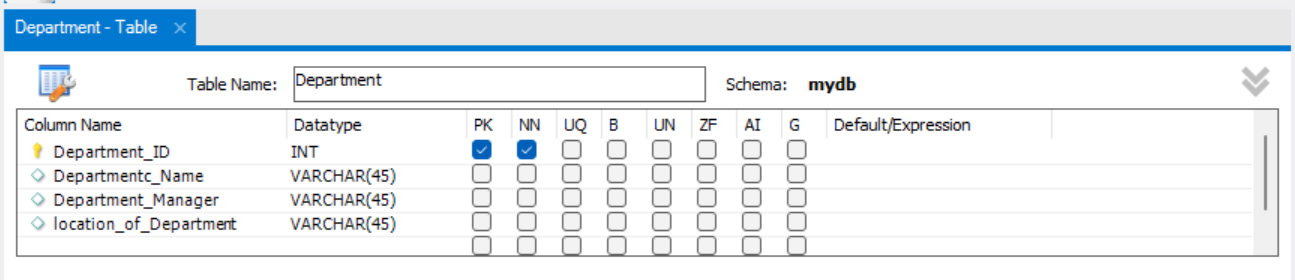
**Attributes:**

Attributes, integral to entities, are the descriptive properties stored within each entity. These characterizations provide insight into the nature of an entity's content. Attributes are listed within rectangles, contributing to the comprehensive understanding of entity characteristics.

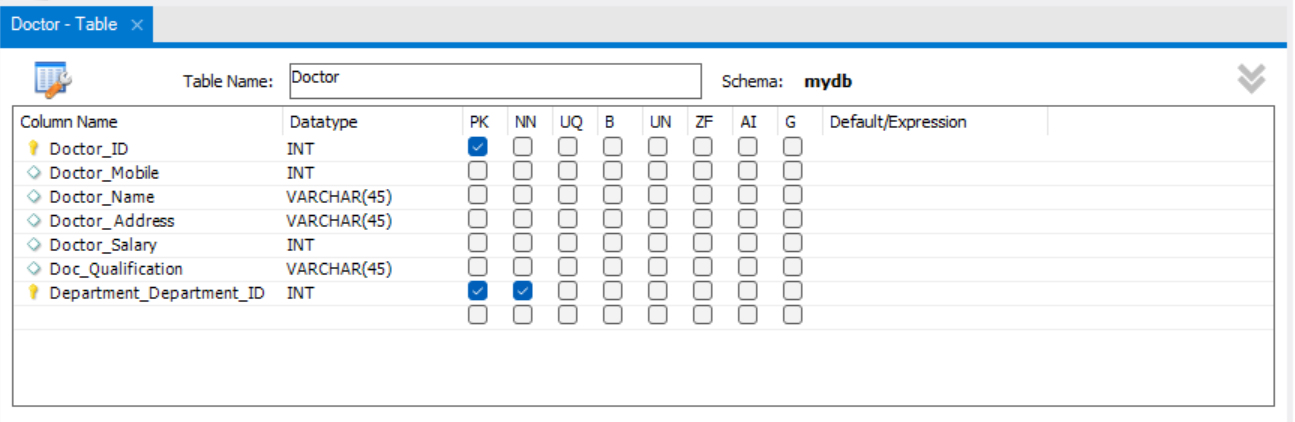
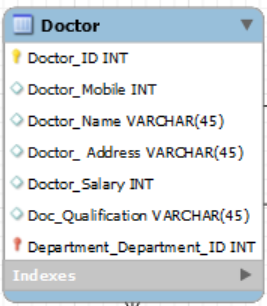
In essence, the Relational Model maintains the coherence and structural integrity of a database, ensuring efficient data management and retrieval. The clear delineation of tables, attributes, relationships, primary keys, and foreign keys establishes a robust foundation for database systems, fostering effective data organization and utilization. [10]

DEPARTMENT\_TABLE



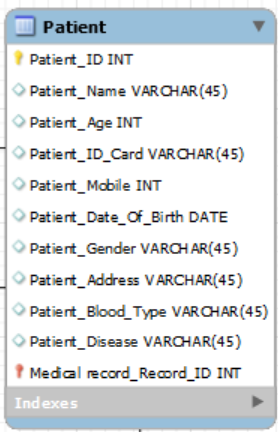


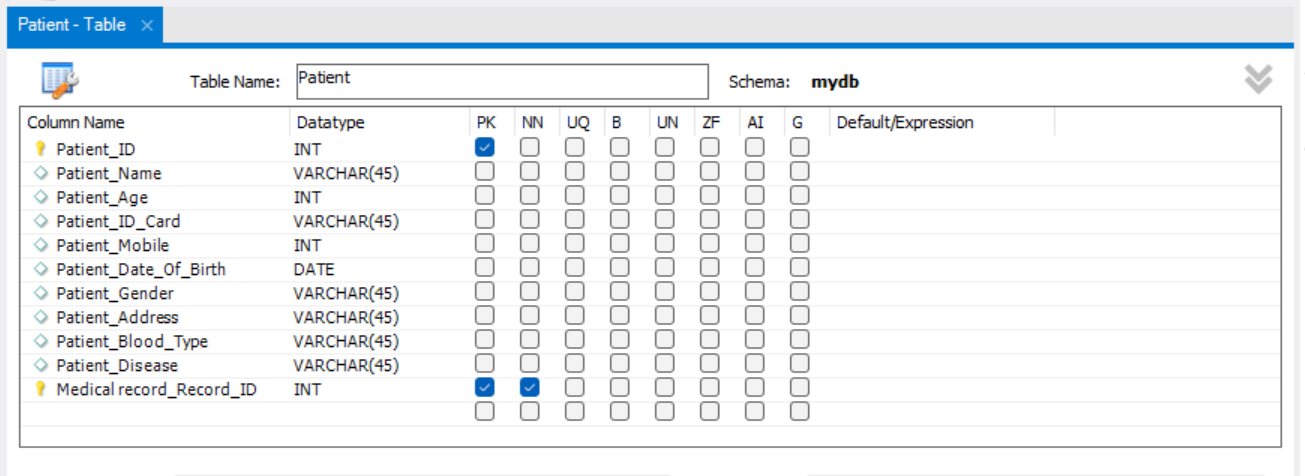
DOCTOR\_TABLE



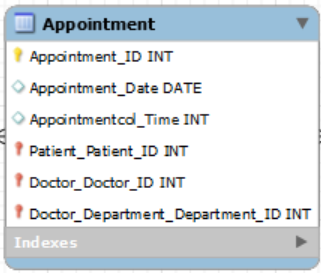
32

PATIENT\_TABLE

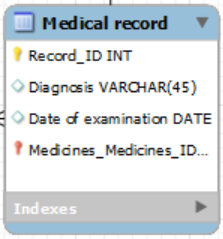


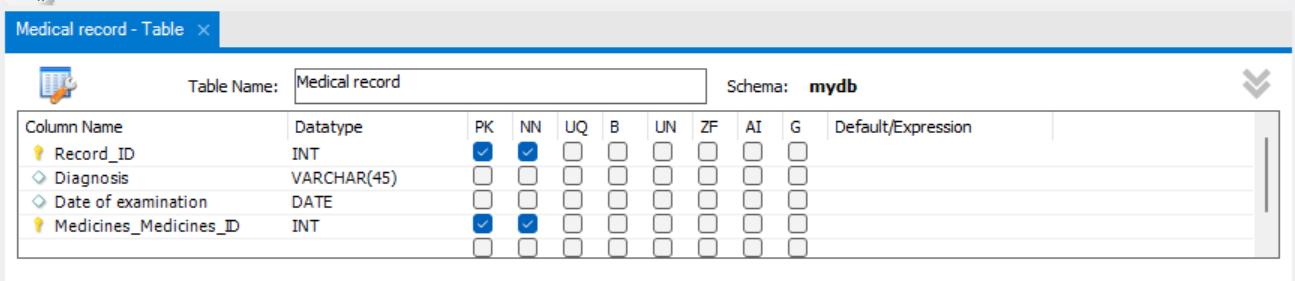


APPOINTMENT\_TABLE



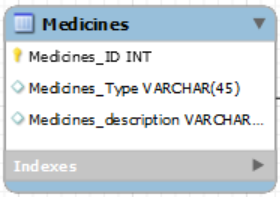
MEDICAL RECORD\_TABLE

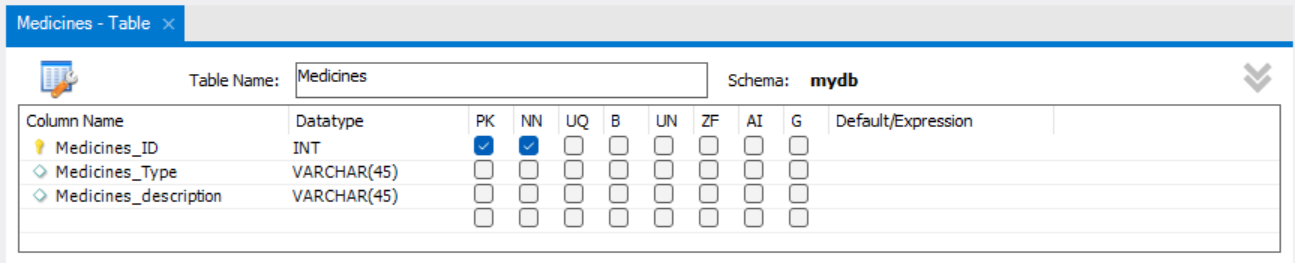




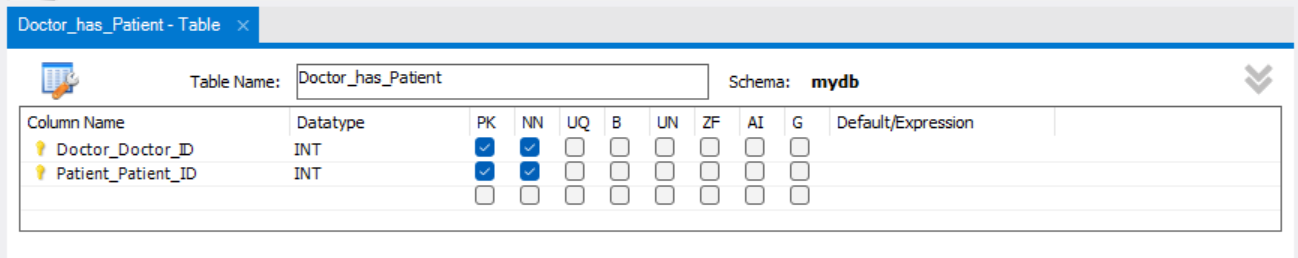
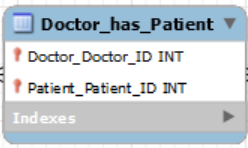
33

MEDICAL RECORD\_TABLE





DOCTOR\_HAS\_PATIENT\_TABLE



34

**After I defined the entities and attributes of all tables and made the ERD we should display them as a dataset:**

****

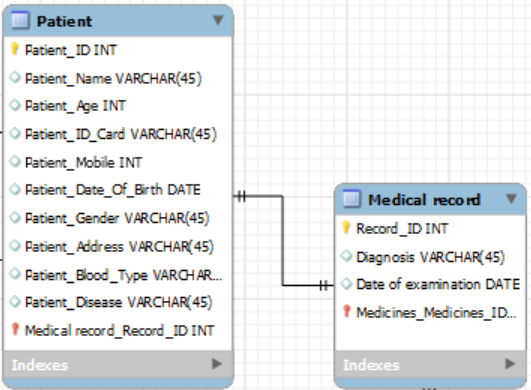
35

The robust advantages of relational databases lie in their ability to organize data into well-defined, compact tables and establish relationships between the data across these tables. Understanding and effectively utilizing database relationships is crucial for designing a database system that reflects the intricacies of the real-world scenario it seeks to model. In this exploration, we delve into the different types of relationships – one-to-one, one-to-many, many-to-one, and many-to-many – that form the backbone of relational database design. [11]

**Types of Relationships**

**One-to-One Relationship:**

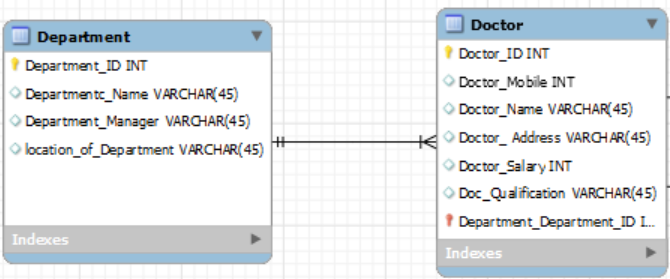
In a one-to-one relationship, one entity from entity set X can be associated with at most one entity from entity set Y, and vice versa. This relationship type is akin to a singular match between entities, creating a direct and exclusive link between them. One common application of a one-to-one relationship is linking employee records to their corresponding payroll information. [11]



36

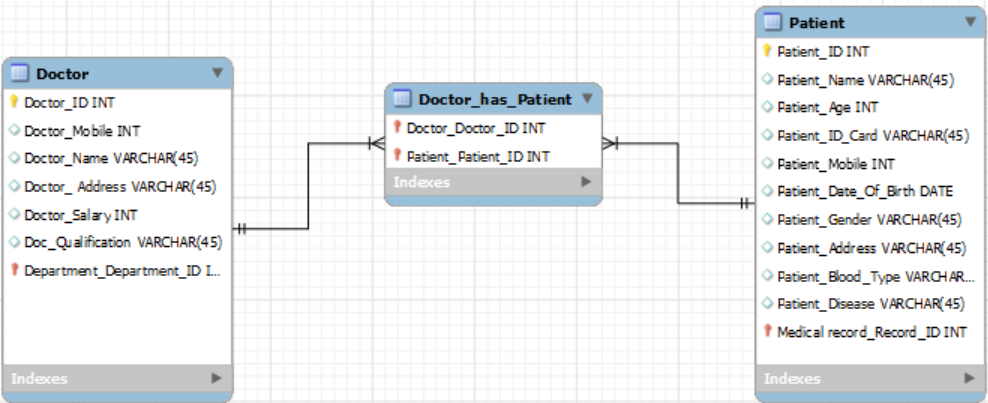
**One-to-Many Relationship:**

The one-to-many relationship is characterized by one entity from entity set X being associated with multiple entities from entity set Y. However, an entity from set Y can be associated with at least one entity from set X. This relationship mirrors scenarios where a single entity is connected to multiple related entities, such as a customer being linked to multiple orders they have placed. [11]



**Many-to-One Relationship:**

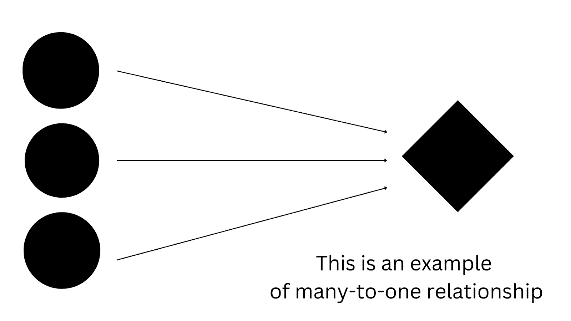
In contrast to the one-to-many relationship, the many-to-one relationship allows more than one entity from entity set X to be associated with at most one entity from entity set Y. Importantly, an entity from set Y may or may not be associated with more than one entity from set X. This relationship type provides flexibility, accommodating scenarios like multiple employees reporting to a single manager.[11]



37

**Many-to-Many Relationship:**

The many-to-many relationship is more intricate, allowing one entity from set X to be associated with more than one entity from set Y, and vice versa. This relationship type often requires the introduction of an associative or junction table to effectively manage and represent the complex associations. An example could be students enrolled in multiple courses, and each course having multiple enrolled students.[11]



**Analysis of Table Relationships:**

When analyzing table relationships, a dual perspective is essential. Each relationship involves two tables: the primary or parent table and the related or child table. Understanding the business rules governing the data is imperative to correctly identify and establish these relationships. Careful examination, consultation with domain experts, and a thorough analysis of the data's nature contribute to the accurate delineation of these pivotal relationships. [11]

The efficacy of relational databases is intricately tied to the adept utilization of database relationships. Whether establishing a direct one-to-one association, accommodating a one-to-many scenario, allowing many-to-one flexibility, or managing the complexity of many-to-many relationships, each type serves a specific purpose. Crafting a well-designed database involves a nuanced understanding of these relationships, ensuring that the data model accurately mirrors the real-world interactions it seeks to represent. [11]

38

**Understanding Database Keys:**

A database key is a unique identifier that allows for the retrieval and organization of records based on specific criteria. Keys are instrumental in enhancing data access speed and fostering relationships between tables within a database. Two primary types of keys, the primary key and the foreign key, form the backbone of relational databases. [11]

**Primary Key:**

A **primary key** is a distinctive field or set of columns within a relational database table, employed to identify each entry uniquely. The primary key serves as a means to interpret and access data efficiently within the database. Key characteristics of a primary key include: [11]

1. **Uniqueness:** Each row of data must possess a unique value for the primary key.
2. **Non-Null Values:** A primary key cannot contain null values.
3. **Assignment:** A primary key value must be assigned to each row within the table.

The primary key can be composed of one or more fields either derived from the existing data model or constructed from scratch to suit the requirements of the database. [11]

39

**Examples:**

Medical Record Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **record\_id**  **Primary key** | | **Diagnosis** | **date\_of\_examination** | | **medicines\_id** | |
|  |  | | |  | |  |
|  |  | | |  | |  |

Medicines Table

|  |  |  |
| --- | --- | --- |
| **medicines\_id** | **medicines\_type** | **medicines\_description** |
|  |  |  |
|  |  |  |

Each table has a unique Id which is Primary Key that identify the details of the of each medical Record and Medicines which are record\_id, medicines\_id.

40

**Foreign Key:**

A **foreign key** is a column or set of columns within a relational database table that establishes a connection between data in two different tables. This key acts as a cross-reference, referring to the primary key of another table. The foreign key concept is integral to relational database systems, where data often needs to be linked across multiple tables to maintain relational integrity. Key aspects of foreign keys include: [11]

1. **Reference to Primary Key:** A foreign key refers to the primary key of another table, establishing a link between the two.
2. **Cross-Referencing:** It serves as a cross-reference, enabling relationships between tables in the database.
3. **Referential Integrity:** Foreign keys are crucial in implementing the concept of referential integrity, ensuring the consistency of relationships between tables.

While primary keys are relatively straightforward, foreign keys introduce complexity due to their role in linking disparate tables within a relational database system. [11]

Understanding the role of keys, especially primary and foreign keys, is paramount in designing efficient and well-structured relational database systems. The primary key uniquely identifies entries within a table, facilitating data interpretation, while foreign keys establish relationships between tables, ensuring referential integrity. These key concepts form the foundation of effective data management and retrieval in the intricate landscape of relational databases. [11]

41

**Examples:**

Doctor table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Doctor\_ID**  **Foreign Key** | **Doctor\_Name** | **Doctor\_Age** | **Doctor\_Mobaile** | **Doctor**  **\_Qualification** | **Doctor\_Salary** | **department\_id** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Department Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Department\_ID** | **Department\_Name** | **department\_manager** | **location\_of\_department** |
|  |  |  |  |
|  |  |  |  |

To illustrate how the doctor is related to a specific department, the primary key of department table should be added also in the doctor table. Once a primary key of a table is added to another table, it now becomes a foreign key.

42

**In-Depth Analysis of Database Data Types**

Database systems play a pivotal role in storing and managing vast amounts of information. Understanding the various data types within a database is crucial for effective data organization and manipulation. This report provides an in-depth exploration of common database data types, elucidating their characteristics and applications.[12]

**1. Integer:**

* Definition: A whole number that can be positive, negative, or zero.
* Characteristics: Cannot be a fraction with decimal places; used for numerical operations.
* Application: Widely employed in programming for counting and numerical calculations. [12]

**2. Character:**

* Definition: Any number, letter, space, or symbol that can be typed into a computer.
* Characteristics: Each character occupies one byte of memory; versatile for alphanumeric data.
* Application: Used for representing individual characters in strings and alphanumeric information. [12]

**3. String:**

* Definition: A series of characters that may include spaces and integers.
* Characteristics: Surrounded by quotation marks to distinguish it as a string; used for representing text.
* Application: Commonly utilized for storing and processing textual information in databases. [12]

**4. Floating Point Number:**

* Definition: A number with decimals; includes fractional numbers.
* Characteristics: Accommodates decimal places in numerical values.
* Application: Suitable for representing real numbers and precise calculations involving fractions. [12]

43

**5. Array:**

* Definition: A collection of elements of the same data type.
* Characteristics: Used for organizing related sets of values; simplifies sorting and searching.
* Application: Facilitates efficient management of data with similar attributes, enhancing data organization. [12]

**6. Varchar (Variable Character):**

* Definition: Variable-length character data type.
* Characteristics: Each character occupies one byte of space plus 2 bytes for length information; flexible storage.
* Application: Ideal for storing textual data with varying lengths, optimizing memory usage. [12]

**7. Boolean:**

* Definition: Used for creating true or false statements.
* Characteristics: Operands include AND, OR, XOR, and NOT for comparing values.
* Application: Employed in logical operations to make decisions based on the truth value of conditions.

Understanding the nuances of different data types is imperative for designing robust and efficient database systems. Each data type serves a specific purpose, catering to diverse needs in data representation and manipulation. By employing the appropriate data types, database designers can enhance the reliability, efficiency, and functionality of their systems, ensuring optimal utilization of resources and streamlined data management. [12]

44

**Effectiveness of Relational Databases and Tables in RDBMS**

Relational Database Management Systems (RDBMS) play a pivotal role in the effective management and processing of data for businesses worldwide. The utilization of simple tables within RDBMS enhances data processing and administration capabilities, providing numerous advantages for businesses in terms of data organization, retrieval, and consistency.[13]

**1. Reduced Data Redundancy:**

* Description: Relational databases eliminate data redundancy, ensuring that no duplicate data entries exist for the same record.
* Significance: Standardization of data through normalization prevents redundancy, enabling efficient utilization of data across multiple tables via joins. [13]

**2. Disaster Recovery:**

* Description: RDBMS facilitates easy data backup and recovery with simple import and export functions.
* Significance: Data migration during database operation reduces the risk of data loss in case of system failure, enhancing disaster recovery capabilities. [13]

**3. Consistency of Data:**

* Description: Alterations in one table of a relational database automatically affect all connected instances, ensuring data consistency.
* Significance: For example, updates made in the patient table regarding appointments reflect in the appointment table, maintaining consistency across interconnected data. [13]

**4. Flexibility:**

* Description: SQL, the querying language for relational databases, allows real-time creation, alteration, updating, and removal of tables.
* Significance: Enables flexibility in adapting database schemas to meet evolving transactional data requirements without interrupting ongoing operations. [13]

45

**5. Atomicity:**

* Description: Describes a database transaction as a single entity not dependent on its components, ensuring all required components are present for a transaction to complete.
* Significance: Guarantees that partial commits are not allowed in relational databases, maintaining the integrity and completeness of transactions. [13]

**6. Database Locking:**

* Description: A feature preventing concurrent access to data being modified, preserving data concurrency and integrity.
* Significance: When multiple users attempt to alter a specific component simultaneously, database locking ensures data consistency and prevents conflicts. [13]

The utilization of RDBMS with simple tables enhances the effectiveness of databases in managing, processing, and securing data for businesses. These systems offer a range of benefits, including reduced data redundancy, improved disaster recovery, consistency across interconnected data, flexibility in schema management, atomicity in transactions, and efficient database locking. Businesses leveraging RDBMS and simple tables experience streamlined data operations, contributing to their overall efficiency and reliability in handling vast amounts of information. [13]

46

**Relational Algebras**

Relational Algebra serves as a procedural query language fundamental to relational databases and SQL. Its operators manipulate input relations, producing meaningful output relations. In this detailed exploration, we delve into the key operators, their functionalities, and their real-world implications.[14]

**Fundamental Operators**

Selection (σ): This operator is pivotal for choosing specific tuples within relations. Unlike a display function, it solely focuses on retrieving the necessary tuples, leaving the showcase of gathered information to the data projection operator. Selection and data projection operators act as guiding stars, bringing coherence to the world of binary code. [14]

Projection (π): Projection is employed to extract required column data from a relation. Illustrated through an example involving Table 1, this operation showcases its ability to project specific columns. Despite whimsical analogies involving celestial bodies and kangaroos, projection remains a vital operation for effective data manipulation. [14]

Union (U): The Union operation in relational algebra aligns with its counterpart in set theory. Illustrated with an example involving doctors and their optional subjects, Union underscores the importance of shared attributes between relations. It serves as a powerful tool but demands adherence to attribute consistency for effective application. [14]

Set Difference (-): Set Difference, familiar from set theory, is employed similarly in relational algebra. Analogies involving rival gangs and finding dominance in the realm of Doctor\_Name contribute to understanding this operation. The operation highlights the nuances between relations and serves as a valuable analytical tool.

Set Intersection (∩): Intersection in relational algebra mirrors the set intersection operation in set theory. Illustrated with an example involving French and German doctors, the operation showcases its utility. The use of algebraic symbols may seem detached from reality, but their application proves essential for effective data manipulation. [14]

47

**Rename (ρ):**

This operator is a singular activity utilized for changing names within a relation. The act of renaming, represented by ρ(a/b)R, allows for the alteration of trait names from 'b' to 'a.' While a seemingly straightforward action, renaming holds potential for data management and organizational clarity. [14]

**Crossed Product (X):**

The cross product involves a cross-operation between two relations, resulting in a combination of attributes from both. Each record from A pairs with every record from B, creating a comprehensive set of outcomes. This operation, while mathematically intriguing, emphasizes the diversity of strengths and weaknesses in individuals. [14]

**Derived Operators**

Natural Join (⋈): Natural Join combines tuples based on equal common attributes across two or more relations. It results in a combined set of tuples where common attributes are equal. This operator is essential for merging data seamlessly.

Conditional Join: Similar to the Natural Join, the Conditional Join allows for the specification of multiple conditions, adding complexity to data manipulation. While resembling its counterpart, the Conditional Join expands the range of conditions, offering more nuanced data processing capabilities. [14]

**Conclusion**

Relational Algebra, through its fundamental and derived operators, provides a structured and efficient approach to data manipulation within relational databases. This detailed exploration highlights the nuanced functionalities of each operator, emphasizing their real-world implications for effective database management and analysis.

48

SQL, the backbone of data management, is the focal point of our exploration. This discussion navigates through pivotal SQL commands – DROP, ALTER, UPDATE, DELETE – and delves into the nuanced landscape of JOINs. Whether you're embarking on your SQL journey or aiming to deepen your expertise, this journey illuminates the foundational and intricate aspects of database design and development. [14]

**The DROP Statement:**

Dropping a Table:

The DROP TABLE command is an irreversible and potent tool to eliminate a table. Its use requires caution.[15]

| **SQL Code** |
| --- |
| DROP TABLE table\_name; |

Dropping a Database:

For a more sweeping action, the DROP DATABASE command can eradicate an entire database along with its tables and data. [15]

| **SQL Code** |
| --- |
| DROP DATABASE database\_name; |

**The ALTER Statement:**

Adding and Removing Columns:

The ALTER statement empowers dynamic modifications to database structures, allowing seamless addition or removal of columns. [15]

| **SQL Code** |
| --- |
| ALTER TABLE table\_name ADD column\_name datatype; |
| ALTER TABLE table\_name DROP COLUMN column\_name; |

49

**ALTER TABLE Queries:**

This statement accommodates various queries, from modifying data types to renaming columns, offering flexibility in evolving database schemas. [15]

| **SQL Code** |
| --- |
| ALTER TABLE table\_name MODIFY column\_name new\_datatype; |

**Data Manipulation Commands:**

UPDATE Statement:

The UPDATE statement is crucial for modifying existing records in a table based on specified conditions. [15]

| **SQL Code** |
| --- |
| UPDATE table\_name SET column1 = value1 WHERE condition; |

DELETE Statement:

DELETE removes records from a table based on specified conditions, a valuable tool for managing and cleansing data. [15]

| **SQL Code** |
| --- |
| DELETE FROM table\_name WHERE condition; |

50

**Mastering the Art of JOINs:**

INNER JOIN:

The INNER JOIN retrieves records with matching values in both tables, excluding non-matching entries. [15]

| **SQL Code** |
| --- |
| SELECT \* FROM table1 INNER JOIN table2 ON table1.column = table2.column; |

LEFT JOIN (or LEFT OUTER JOIN):

The LEFT JOIN retrieves all records from the left table and matched records from the right table, filling in nulls for non-matching entries. [15]

| **SQL Code** |
| --- |
| SELECT \* FROM table1 LEFT JOIN table2 ON table1.column = table2.column; |

RIGHT JOIN (or RIGHT OUTER JOIN):

The RIGHT JOIN returns all records from the right table and matched records from the left table. [15]

| **SQL Code** |
| --- |
| SELECT \* FROM table1 RIGHT JOIN table2 ON table1.column = table2.column; |

51

FULL JOIN (or FULL OUTER JOIN):

The FULL JOIN returns all records with a match in either the left or right table, filling in nulls for non-matching entries. [15]

| **SQL Code** |
| --- |
| SELECT \* FROM table1 FULL JOIN table2 ON table1.column = table2.column; |

Natural JOIN:

A Natural JOIN automatically matches columns with the same name in both tables. [15]

| **SQL Code** |
| --- |
| SELECT \* FROM table1 NATURAL JOIN table2; |

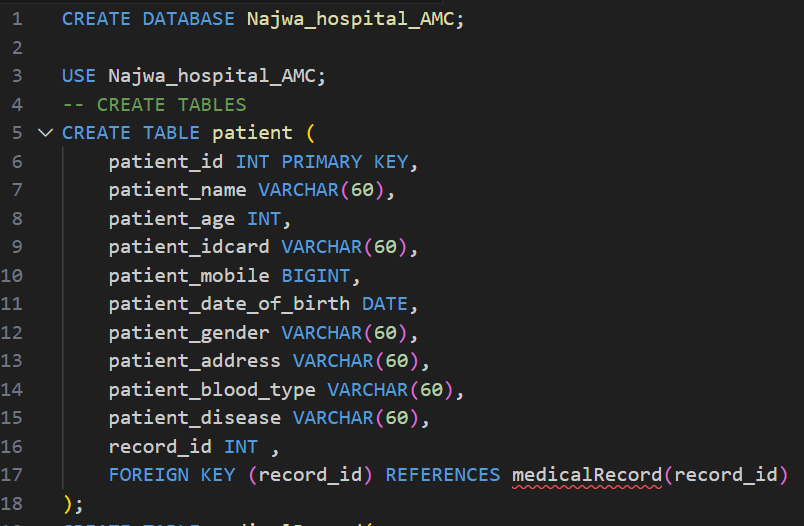
JOIN Three Tables:

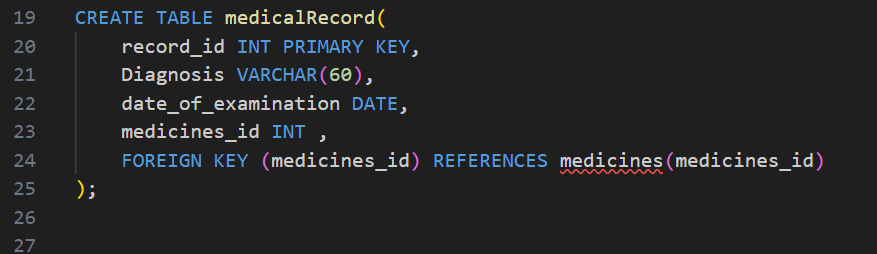
Combining multiple tables is common in complex database scenarios. Utilize the JOIN clause sequentially for three-table joins. [15]

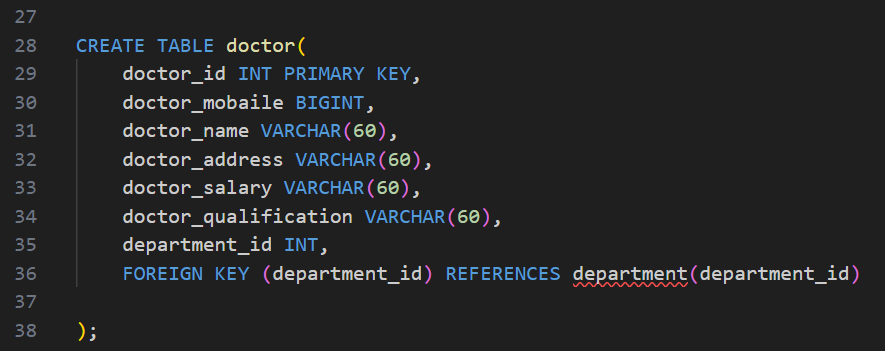
| **SQL Code** |
| --- |
| SELECT \* FROM table1 JOIN table2 ON table1.column = table2.column JOIN table3 ON table2.column = table3.column; |

52

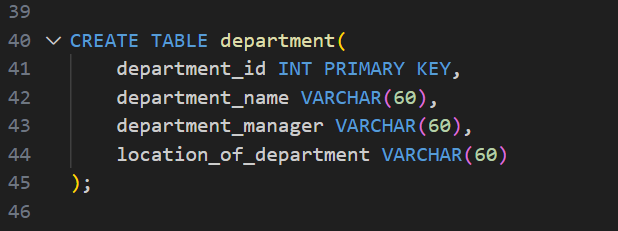
**CREATE TABLES**

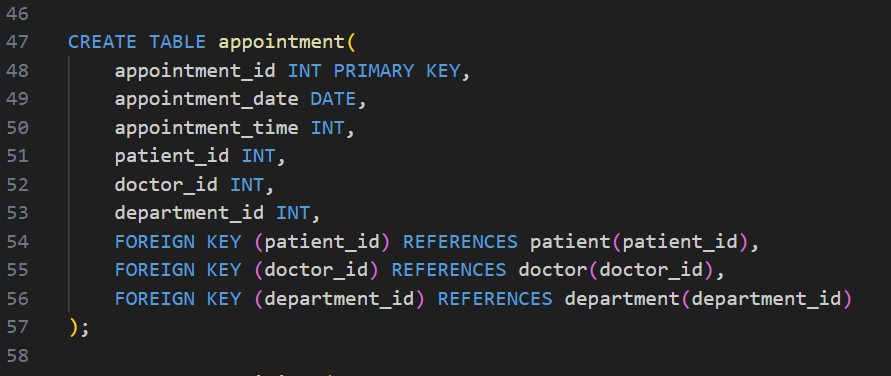


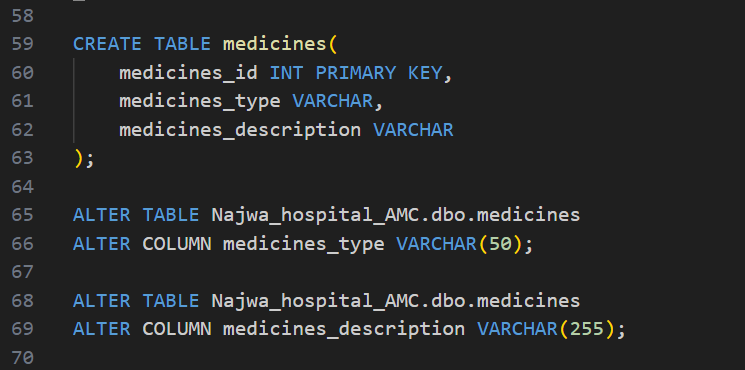




53

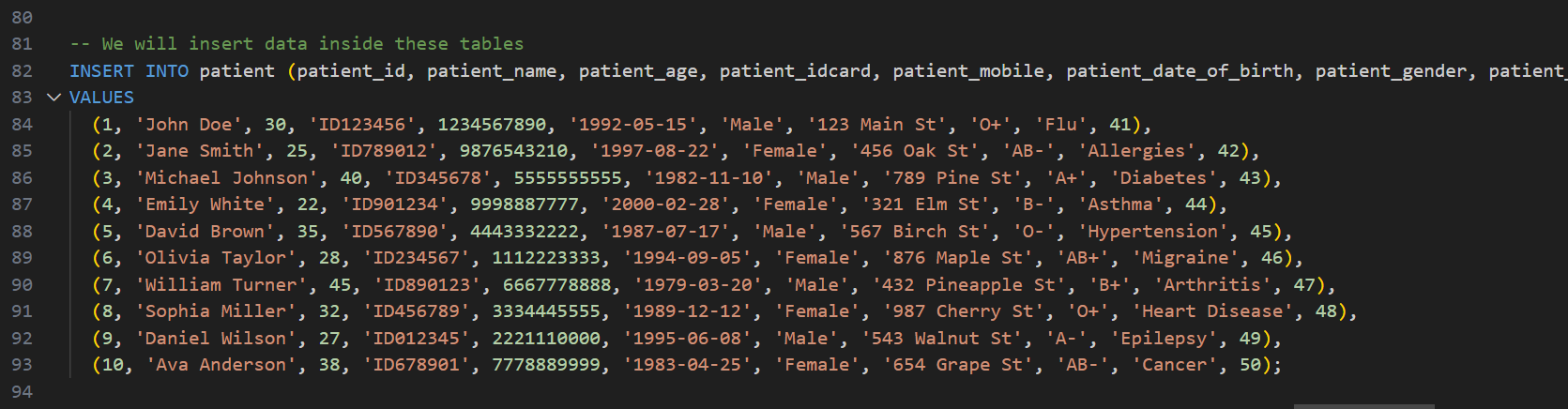


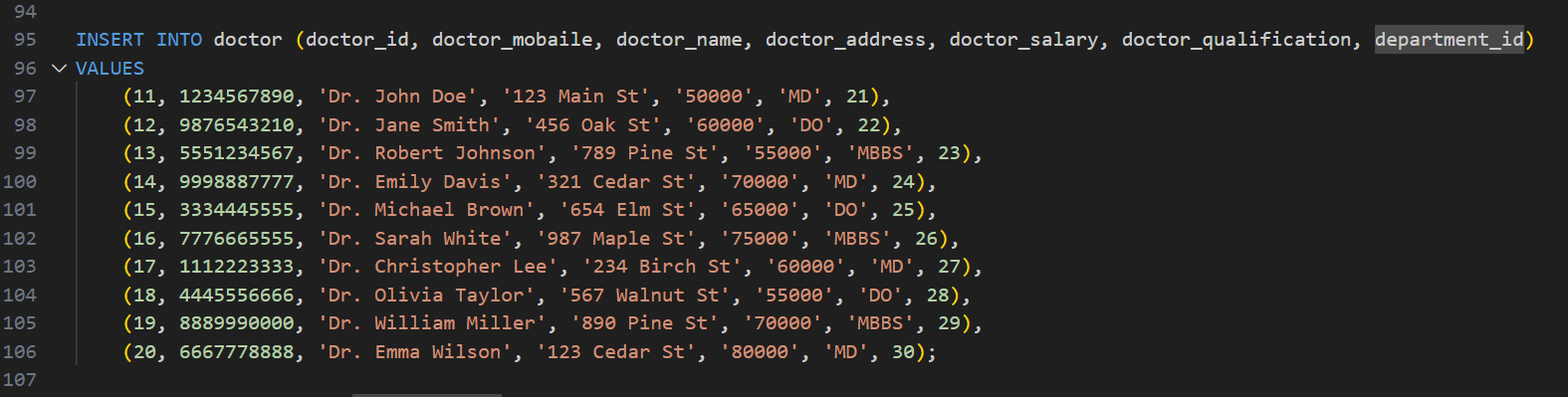


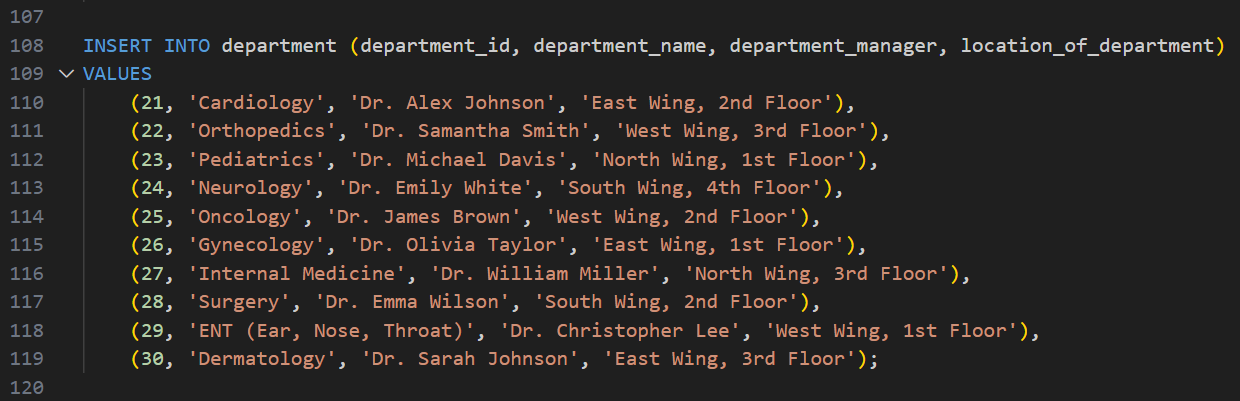


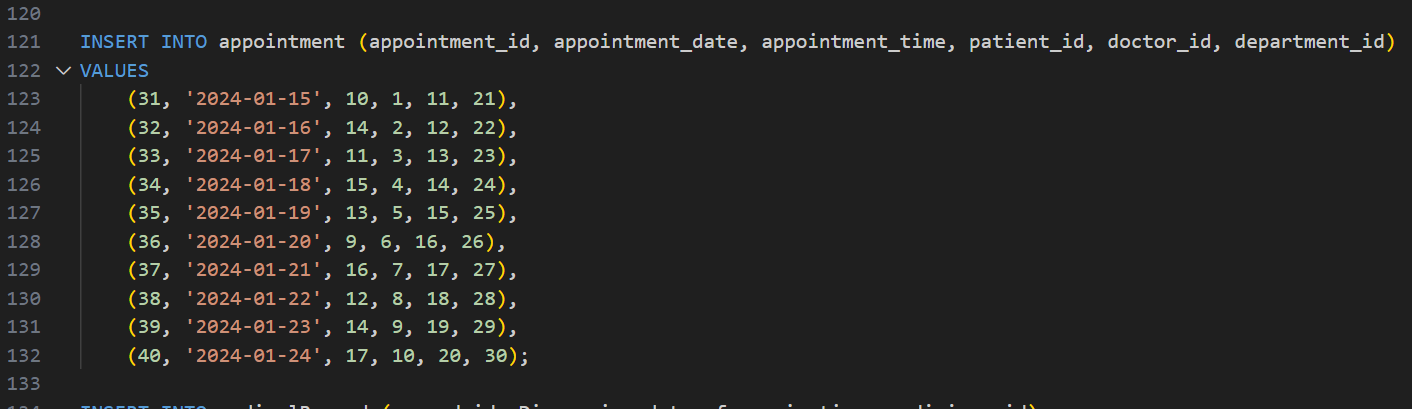


**INSERT DATA INTO TABLES**

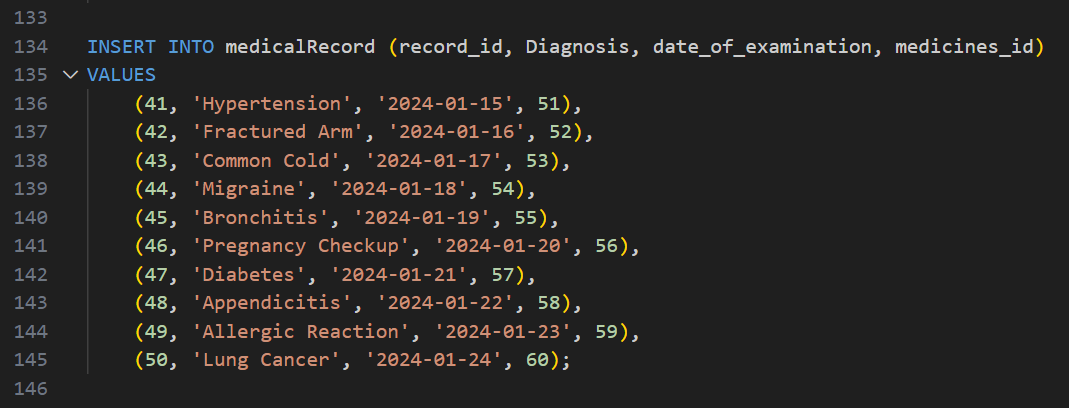


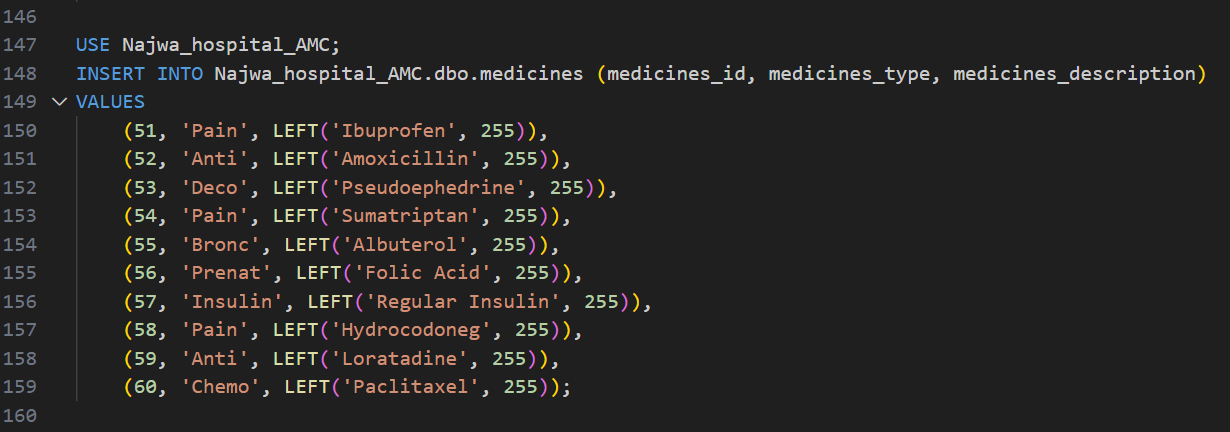






55

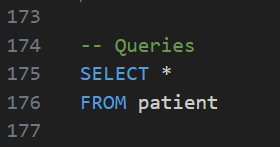


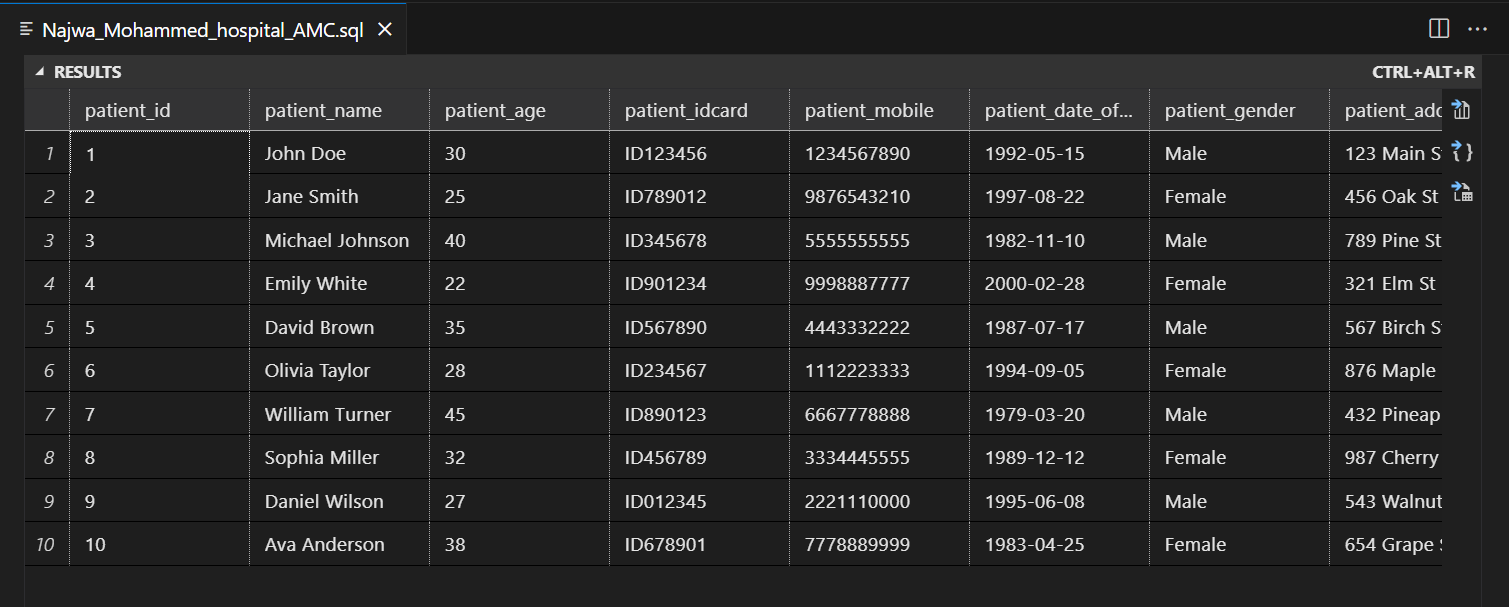


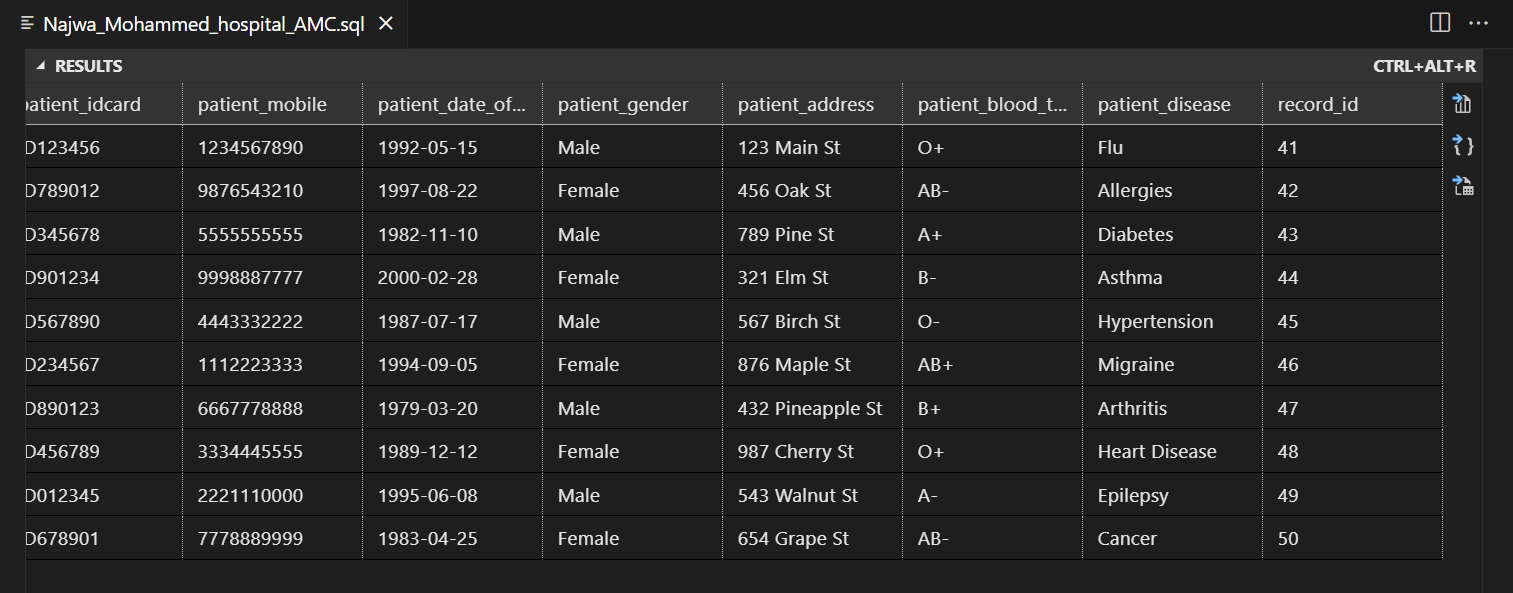


56

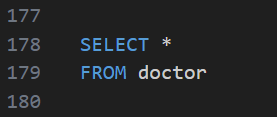
**QUERIES**

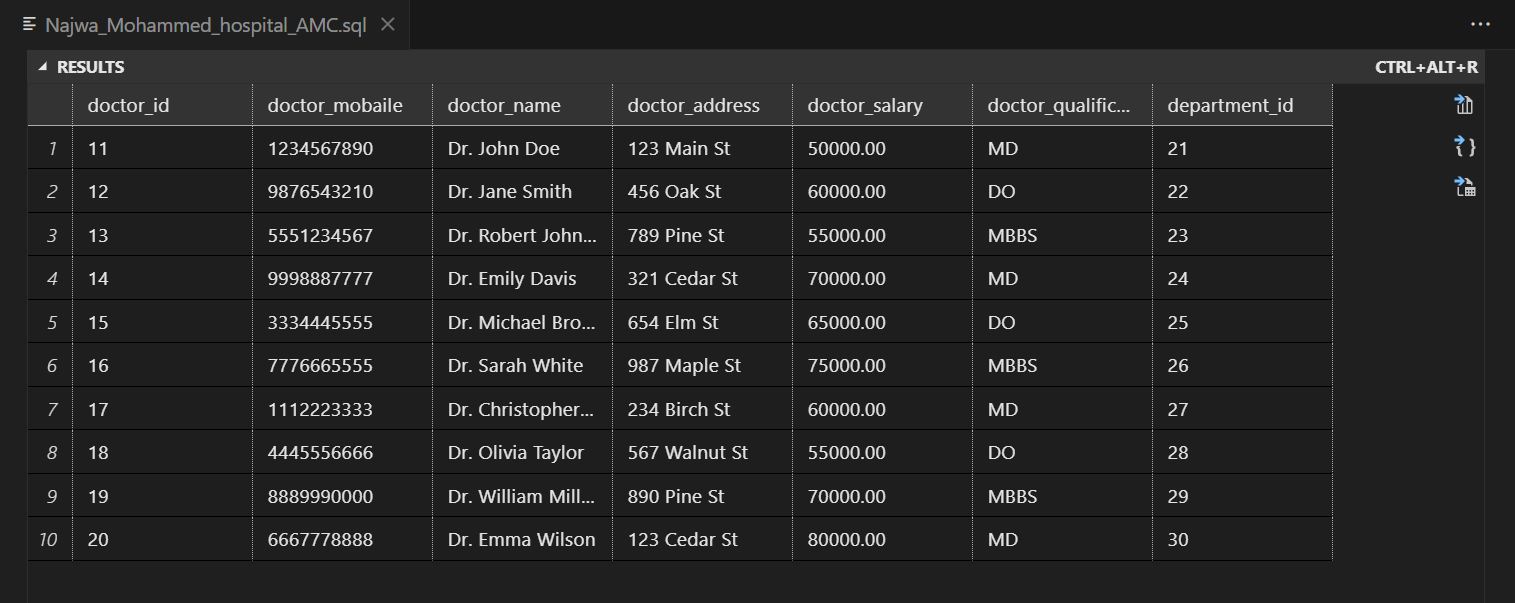


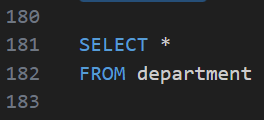


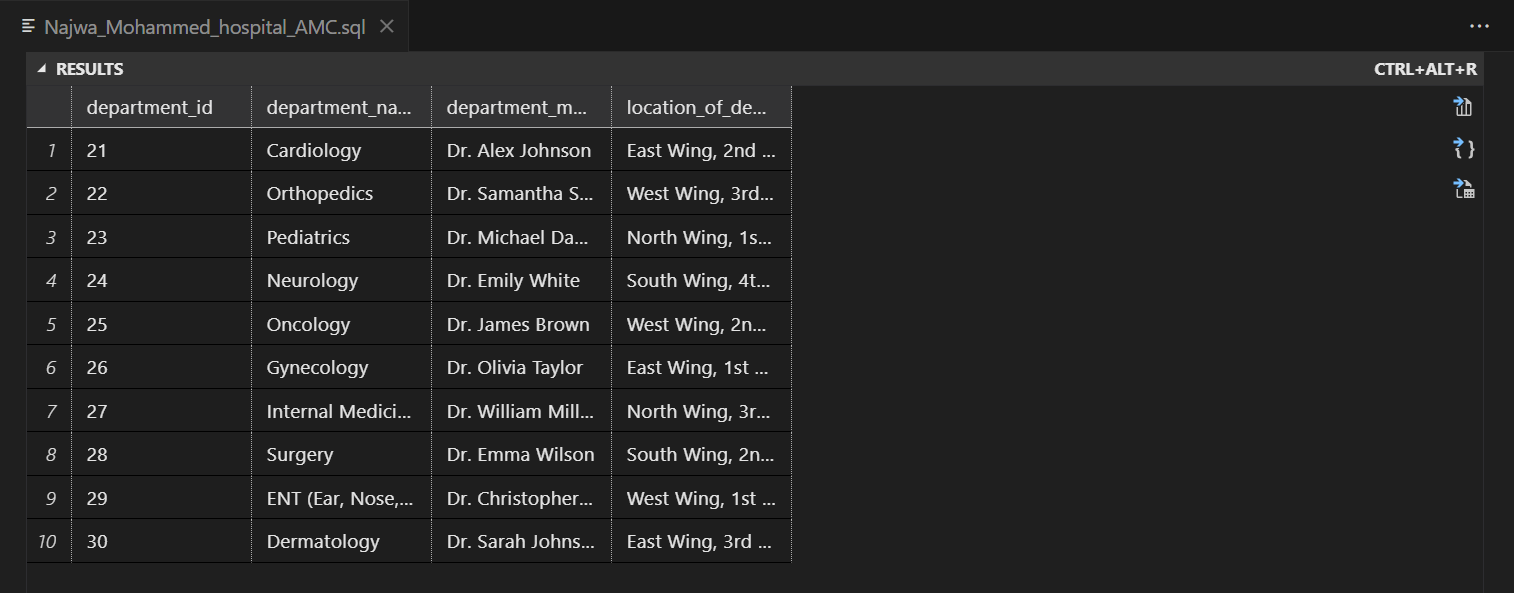


57

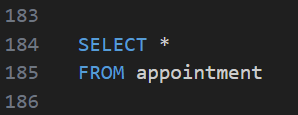


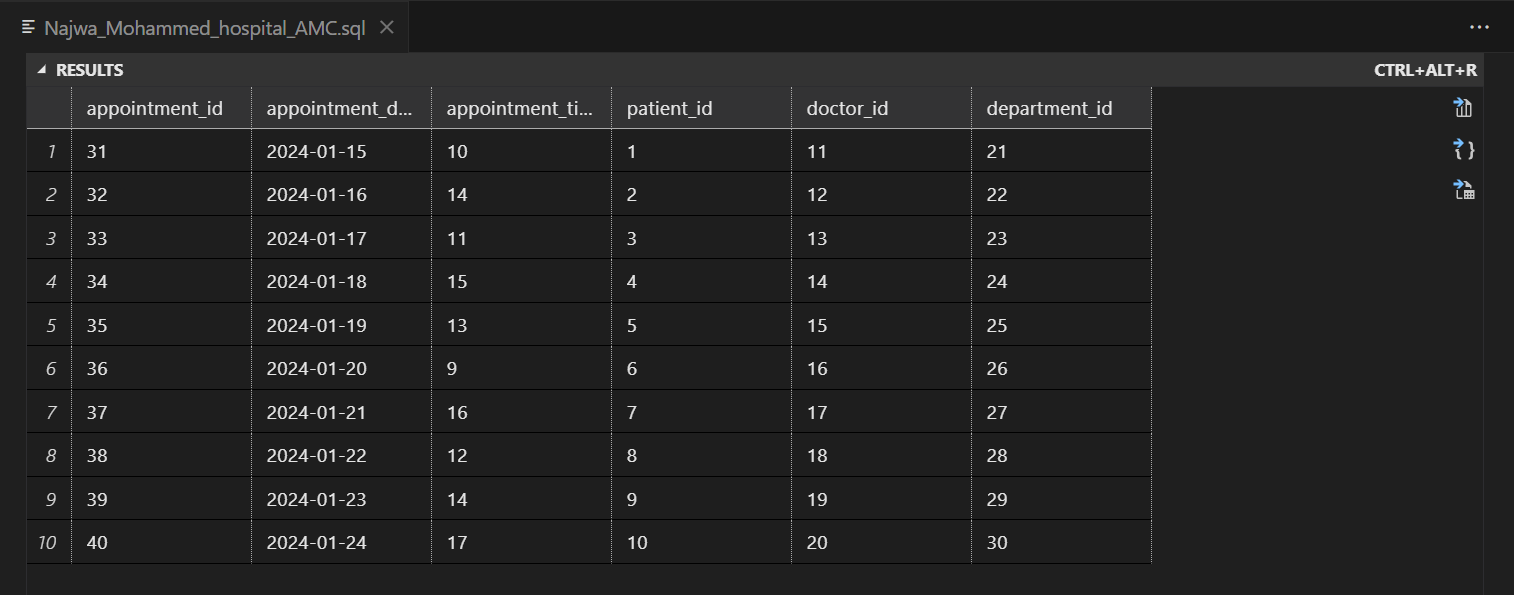


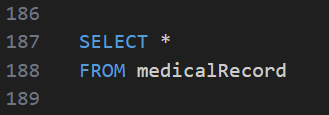


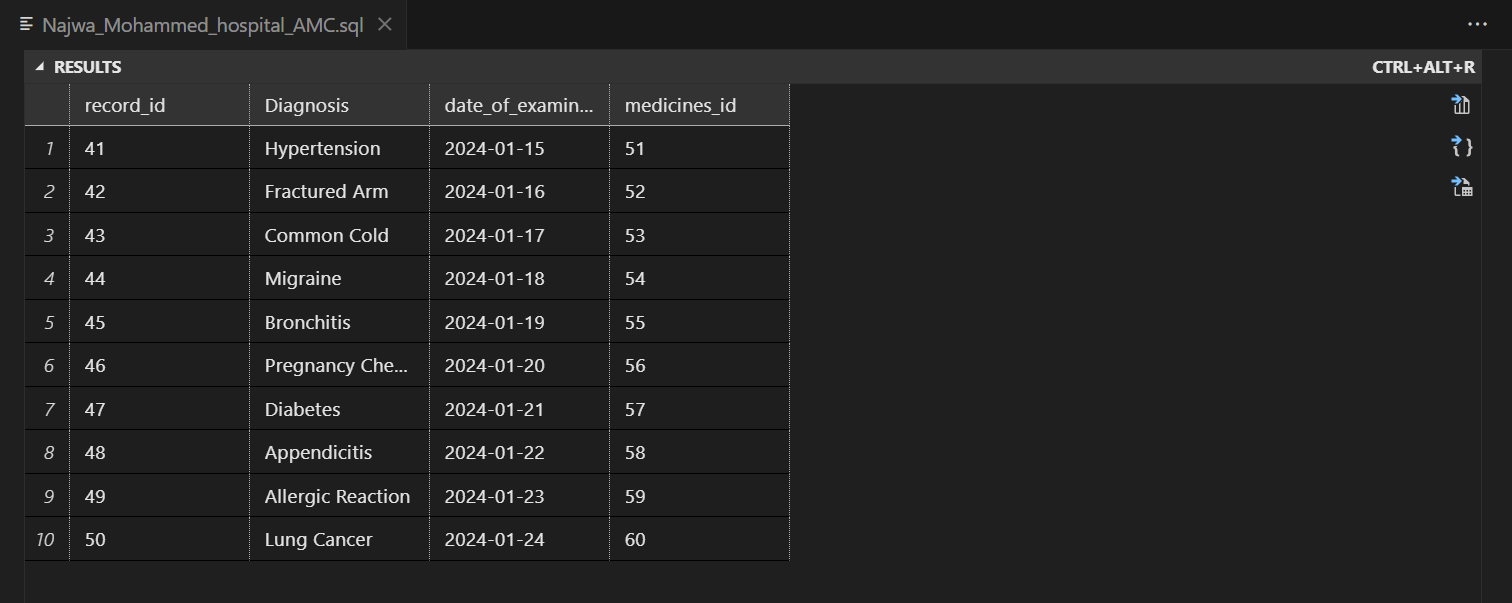


58

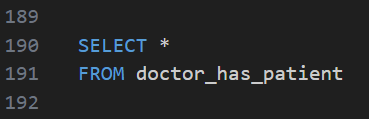


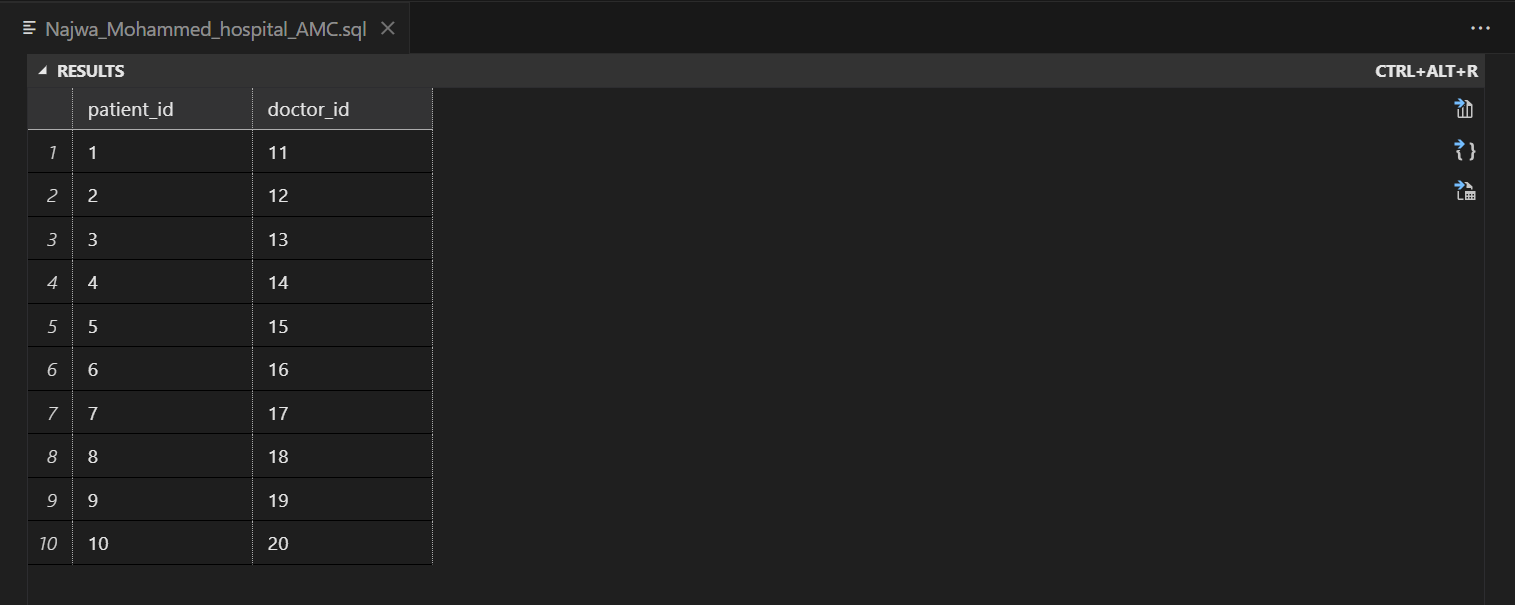


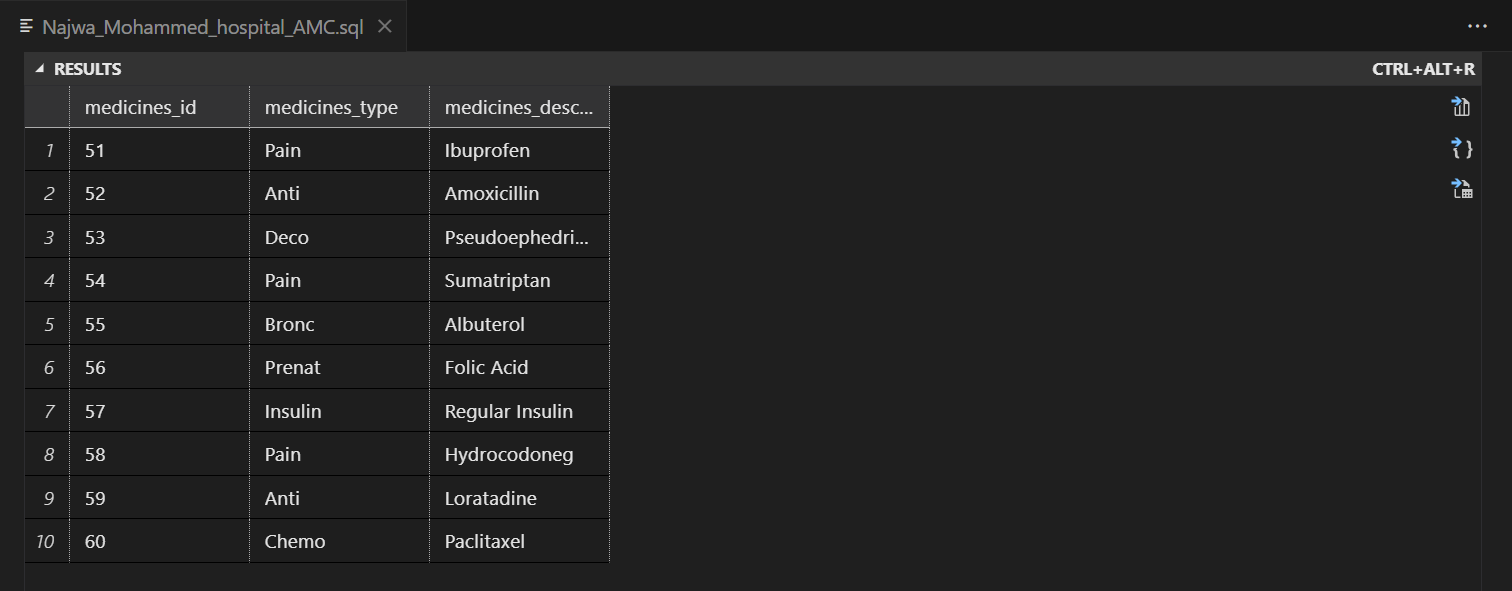
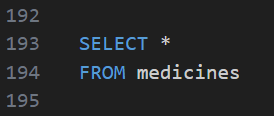




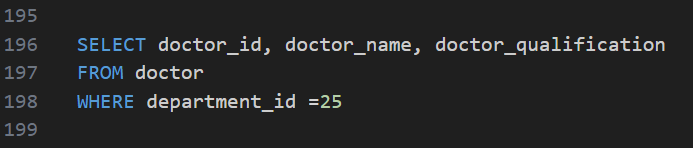
59

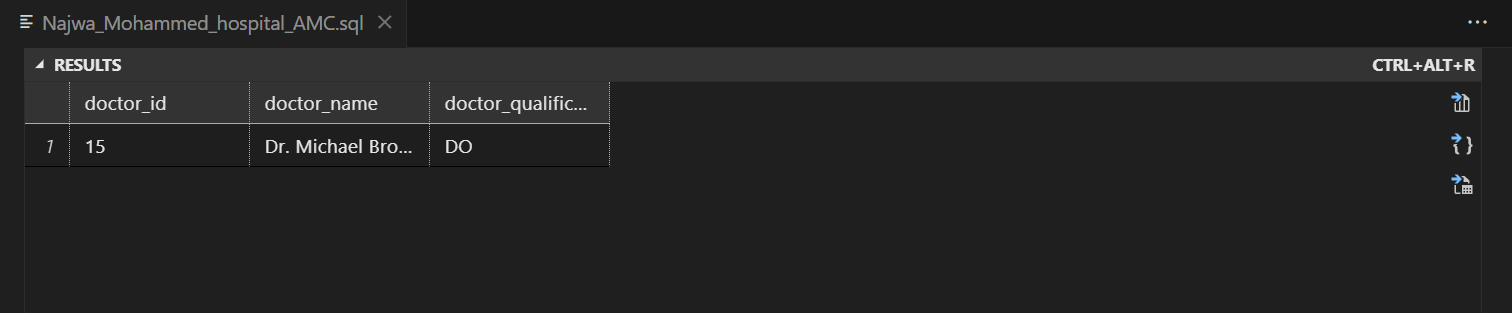


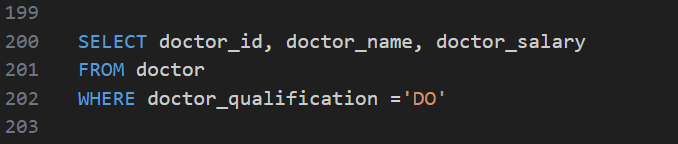


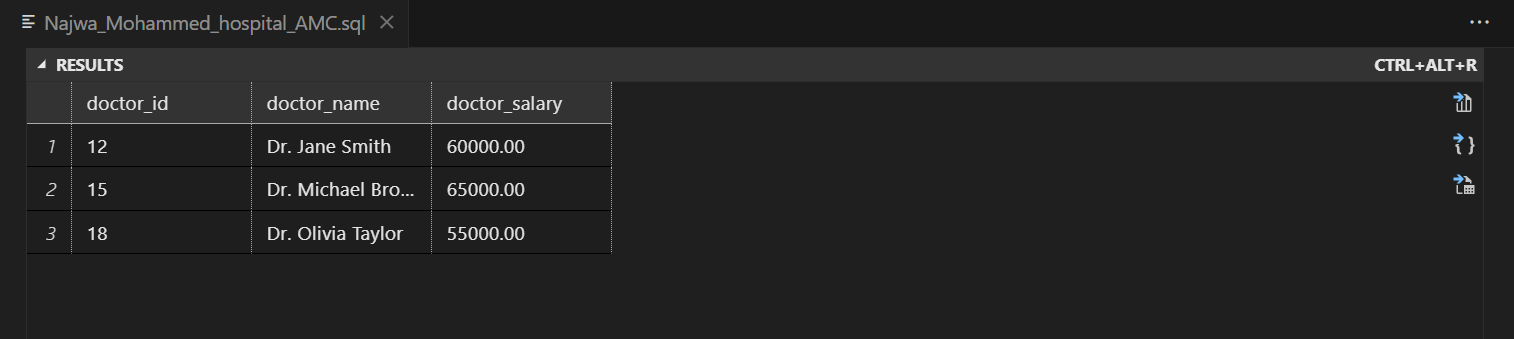


60

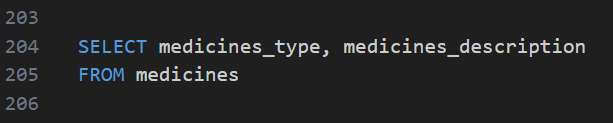


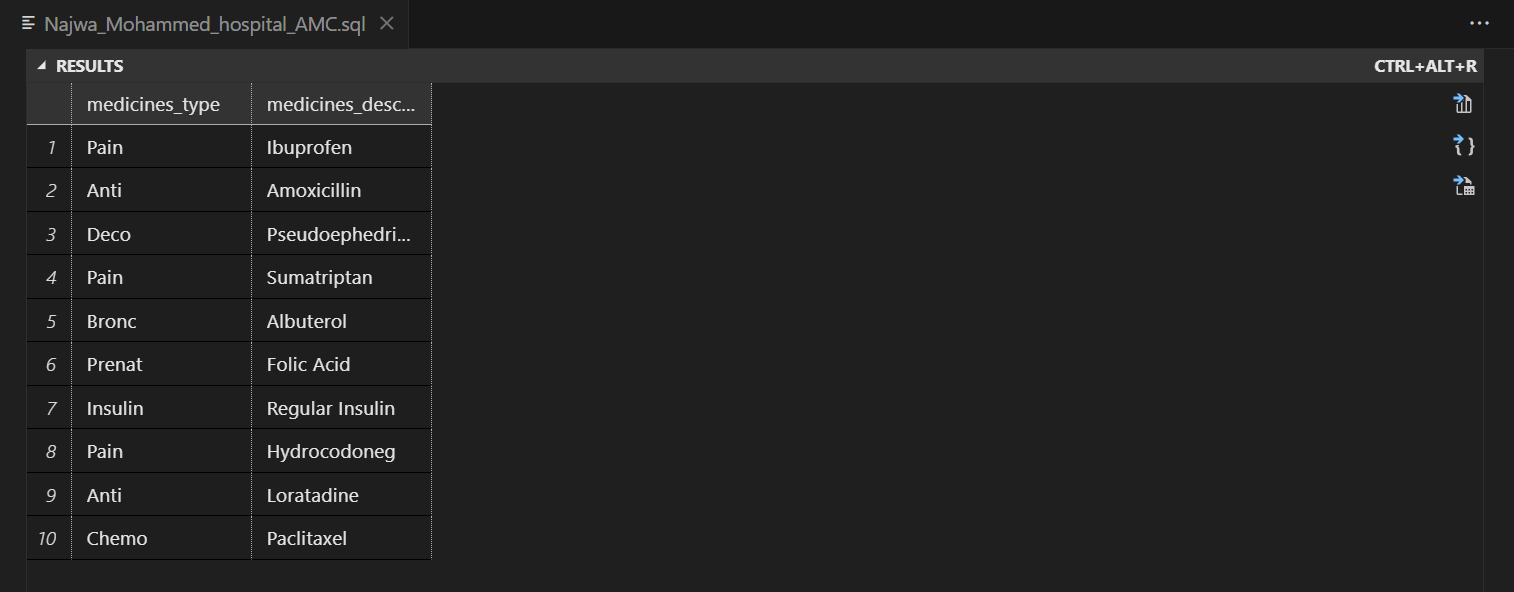


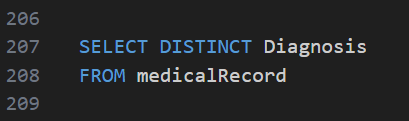


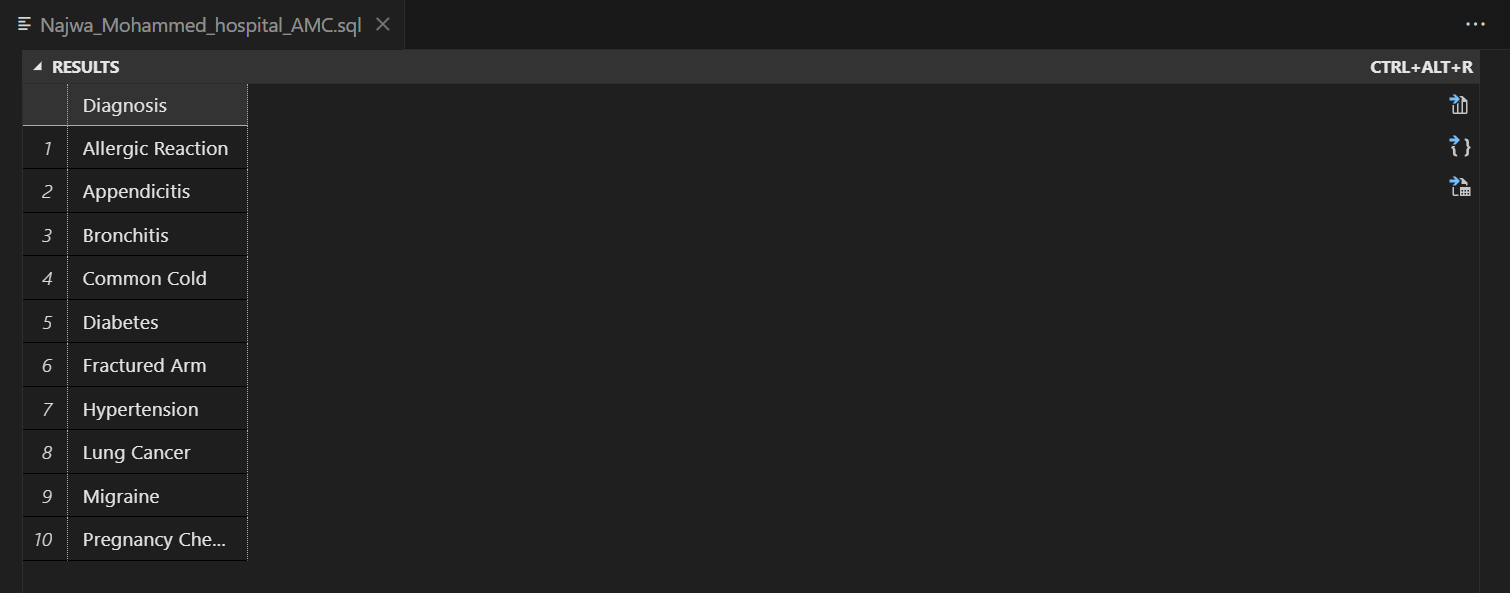


61

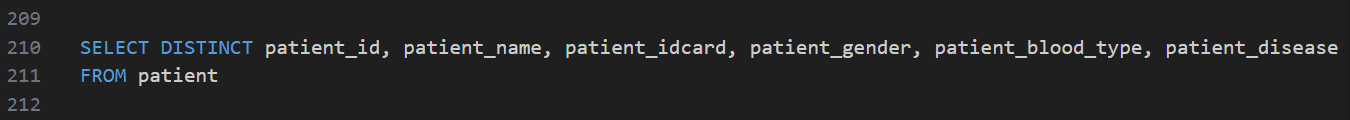


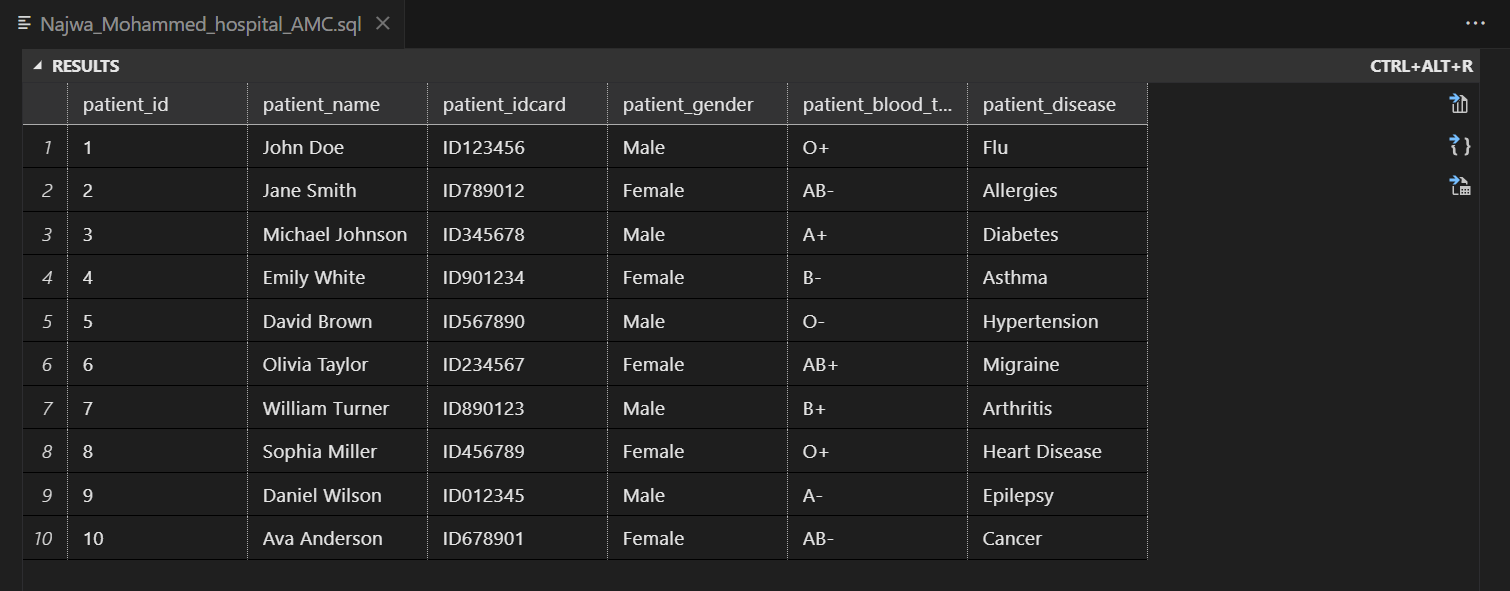


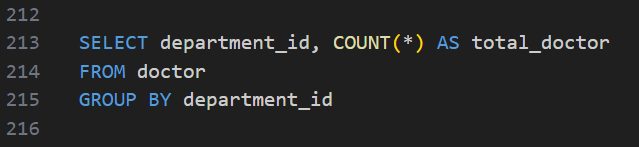


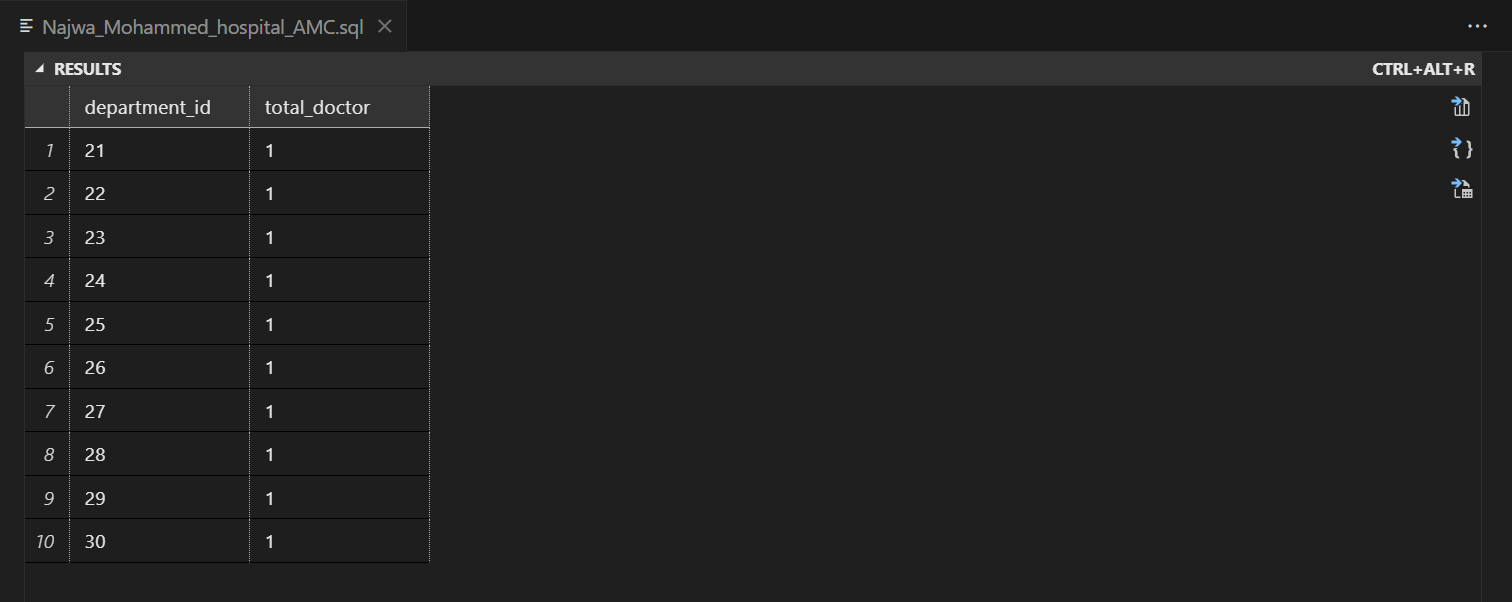


62

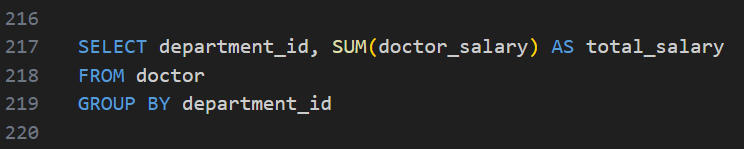


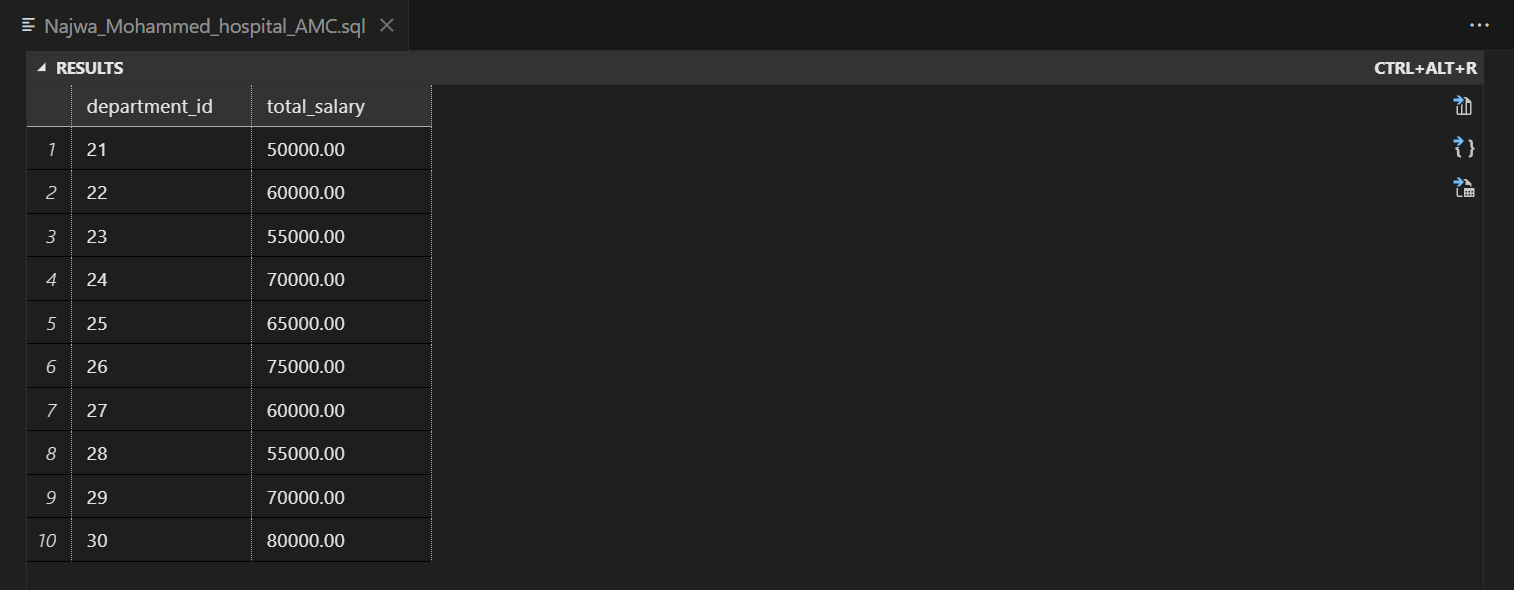


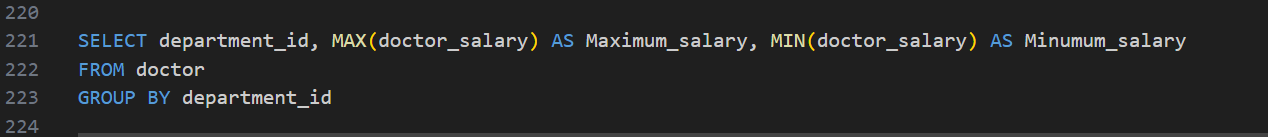


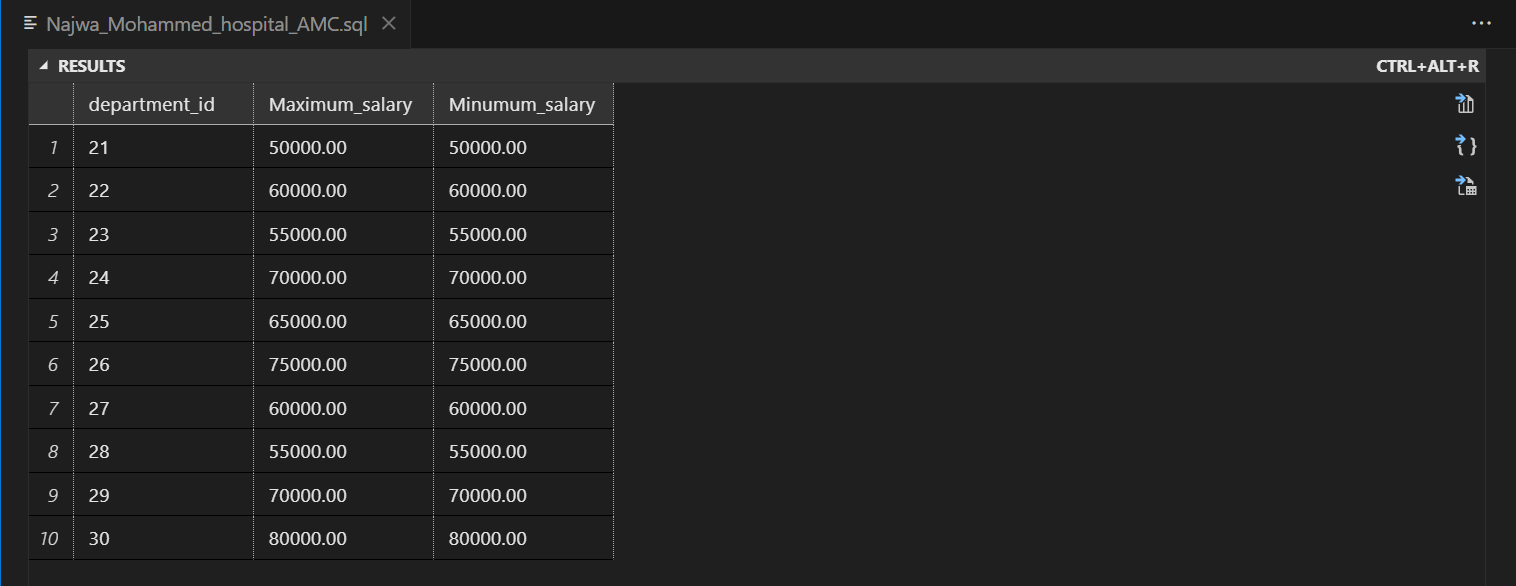


63

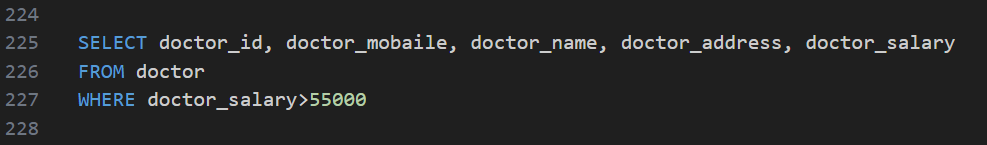


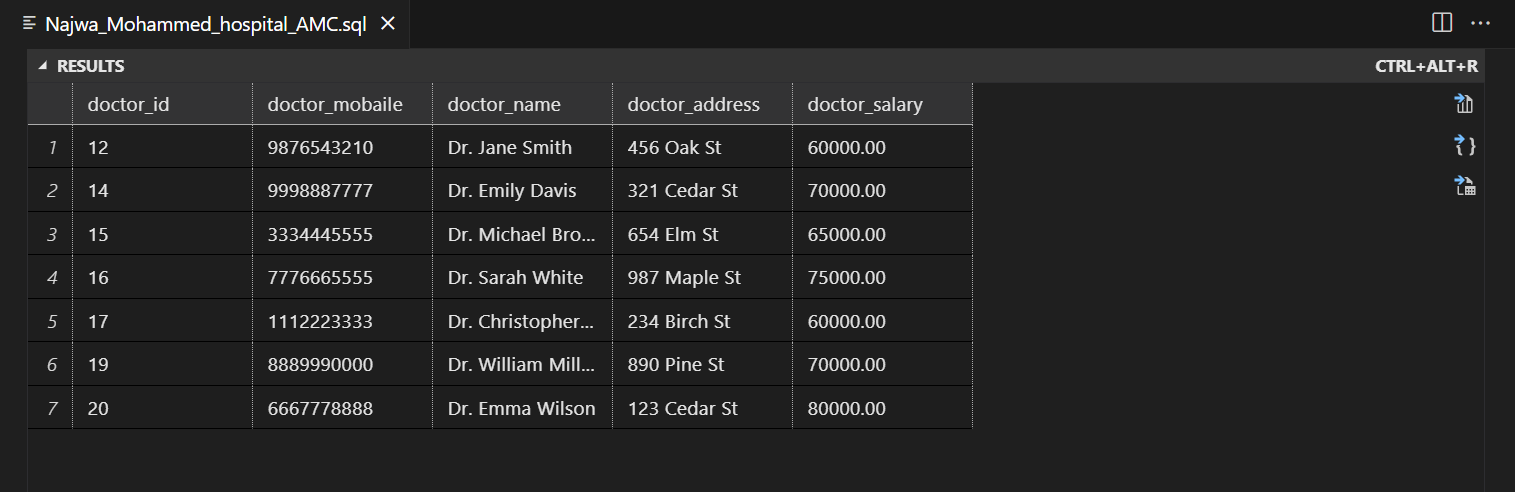


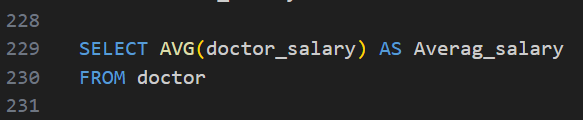


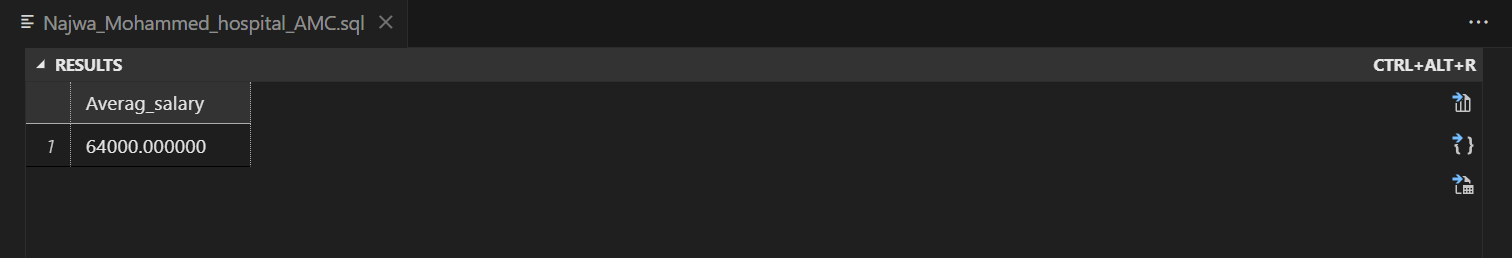


64

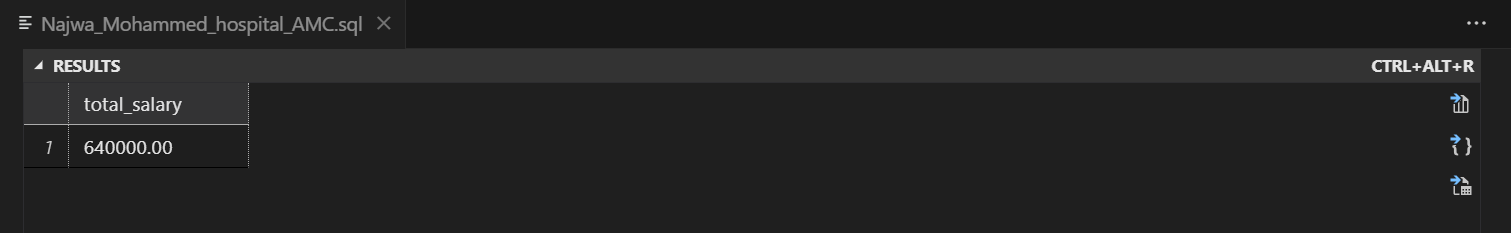












65

References

[1] “What is a database?” Accessed: Jan. 15, 2024. [Online]. Available: https://www.oracle.com/ae/database/what-is-database/

[2] “History of Data Models and Databases.” Accessed: Jan. 15, 2024. [Online]. Available: http://graphdatamodeling.com/GraphDataModeling/History.html

[3] “Relational vs. Non-Relational Database: Pros & Cons.” Accessed: Jan. 15, 2024. [Online]. Available: https://aloa.co/blog/relational-vs-non-relational-database-pros-cons, https://aloa.co/blog/relational-vs-non-relational-database-pros-cons

[4] “Types of Databases.” Accessed: Jan. 16, 2024. [Online]. Available: https://www.c-sharpcorner.com/UploadFile/65fc13/types-of-database-management-systems/

[5] “Database Architecture in DBMS: 1-Tier, 2-Tier and 3-Tier.” Accessed: Jan. 16, 2024. [Online]. Available: https://www.guru99.com/dbms-architecture.html

[6] “Advantages and Disadvantages of DBMS.” Accessed: Jan. 16, 2024. [Online]. Available: https://www.faceprep.in/data-structures/advantages-and-disadvantages-of-dbms/

[7] “Functions of DBMS - javatpoint.” Accessed: Jan. 16, 2024. [Online]. Available: https://www.javatpoint.com/functions-of-dbms

[8] karie\_barrett@qat.com, “Guide to Writing Data Requirements,” QAT Global. Accessed: Jan. 16, 2024. [Online]. Available: https://qat.com/guide-writing-data-requirements/

[9] “Understanding the Basics of Database Normalization.” Accessed: Jan. 16, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2023/03/understanding-the-basics-of-database-normalization/

[10] “DBMS: Database Management Systems Explained,” BMC Blogs. Accessed: Jan. 16, 2024. [Online]. Available: https://www.bmc.com/blogs/dbms-database-management-systems/

[11] “Introduction of ER Model,” GeeksforGeeks. Accessed: Jan. 16, 2024. [Online]. Available: https://www.geeksforgeeks.org/introduction-of-er-model/

[12] “Types of Data Analysis,” Chartio. Accessed: Jan. 16, 2024. [Online]. Available: https://chartio.com/learn/data-analytics/types-of-data-analysis/

[13] “What is a relational database? | IBM.” Accessed: Jan. 16, 2024. [Online]. Available: https://www.ibm.com/topics/relational-databases

[14] “Introduction of Relational Algebra in DBMS,” GeeksforGeeks. Accessed: Jan. 16, 2024. [Online]. Available: https://www.geeksforgeeks.org/introduction-of-relational-algebra-in-dbms/

[15] avcontentteam, “SQL Skill Test | SQL Quiz to Test a Data Science Professional (Updated 2023),” Analytics Vidhya. Accessed: Jan. 16, 2024. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/01/46-questions-on-sql-to-test-a-data-science-professional-skilltest-solution/

66