**Medical Cooperative Information System**

The database can be used for work by a team of doctors. The table lists the name, gender, date of birth and home address of each of their patients. Whenever a doctor examines a patient who has made an appointment with him or comes to his home himself, he writes down the date and place where the examination is performed, symptoms, diagnosis and prescriptions for the patient, puts the patient's name, as well as his own name. If the doctor prescribes a medicine to the patient, the name of the medicine, the method of its administration, a verbal description of the intended effect and possible side effects are entered in the table. A package consisting of functions and procedures that allow:

1) determine the number of calls on this day based on a given date;

2) allow to determine the number of patients who fell ill with this disease;

3) to determine the order of the side effect of the prescribed medication;

4) provide the possibility of adding a new drug with a description of its properties in the database.

CONTENT

[Introduction 4](#_Toc90654587)

[Analysis of the subject area 6](#_Toc90654588)

[Functional modeling 8](#_Toc90654589)

[Conceptual modeling 10](#_Toc90654590)

[Relational database model 14](#_Toc90654591)

[Physical model of the database 15](#_Toc90654592)

[Implementation of tasks 21](#_Toc90654593)

[Conclusions 24](#_Toc90654594)

[List of used literature 25](#_Toc90654595)

Introduction

It is hard to imagine a day without mobile communication and electronic payments. One of the key components of modernity are databases, which play an important role in creating the modern world. These databases are used to store large amounts of information such as e-queues, social media profile data, student information on the KPI campus. They typically include schemas, tables, stored procedures, and other objects.

Database management systems (DBMS) are usually used to manage databases, which help to effectively manage and manipulate information. Some applications may also be part of a DBMS, such as MySQL, Microsoft SQL Server, Oracle. In my work, I chose Microsoft SQL Server as a convenient tool that uses the Transact-SQL language and provides all the necessary features for successful work.

Database design includes several stages:

1. Analysis of the problem, selection of methods of implementation of the task and drawing up a plan.

2. Analysis of objects, their composition, parameters and types.

3. Database modeling and schema creation.

4. Selection of application and DBMS, as well as creation of database and tables.

The main reason for the need for the ability to create databases in the medical field is the effective collection, management and analysis of a large amount of medical information. Databases allow doctors and other medical personnel to efficiently access information about patients, their medical history, treatment and diagnosis. It helps to make informed decisions and ensures reliable medical care. Also, they allow to effectively keep records of medicines, control over appointments and taking medicines by patients, which is important to prevent dangerous interactions and deficiencies in treatment. The correct creation and management of databases in medicine allows compliance with high standards of confidentiality of medical information, which is a particularly important aspect in health care. Modern databases facilitate the standardization and exchange of medical information between different medical institutions, which facilitates collaboration and provides consolidation of data for deeper analysis.

The task of work includes:

1. Study of the subject area.

2. Detailed analysis.

3. Construction of tables and diagrams based on the conducted analysis.

4. Implementation of procedures and triggers according to tasks.

5. Formulation of conclusions.

Analysis of the subject area

It is important to know that the database is created for the work of a team of doctors. One of the most important stages of building a database is the analysis of the subject area. As already mentioned above, the subject area of ​​my work is a medical cooperative.

The main tables in this database will be the "Physicians" and "Patients" tables. About patients, we need to know their name, address and age. Information about doctors: full name, specialty, position. Each patient is assigned to a specific doctor, for this a special table (doctor-patient) will be created, where the doctor and the patient, the patient's name, his gender, birthday and home address are marked. Also, it will be necessary to create a table of scheduled appointments, where there is information about the doctor, the patient, the location, the patient's symptoms, the diagnosis, the medication prescribed to the patient, the appointment date, and the doctor's call number for that day. We also have the "Drugs" table, which contains all the information about them, namely the name, method of application, side effects, and a description of the drug's action.

The main tasks to be solved within the software:

1. Build all the necessary tables.

2. Provide for the development of triggers that provide cascading changes in related tables.

List of main data requests:

1. According to the given date, determine the number of calls on this day;

2. They allow to determine the number of patients who fell ill with this disease;

3. According to the given medicine, determine its side effect order;

4. Provide the possibility of adding new drugs with their description

Functional modeling

A DFD diagram is a graphical tool for representing the flow of data and information processing in a system. It allows you to visualize how data moves through various processes in the system, how they are processed and interact with each other.

The main elements of a DFD include: Processes (Represent operations or activities that perform data processing. They can include such elements as calculations, data transformations, receiving or sending information), Data Flows (Lines that represent data flows between processes , data storage and output. They illustrate the direction of information movement in the system), Data Repositories (Indicate the places where data is stored. These can be databases, files or any other means of information storage), External Entities (Represent external sources or data receivers that interact with the system (eg users, other systems, sensors, etc.)

The main rules of the DFD diagram:

1. Each process has at least 1 input and at least 1 output, because the process must receive data for processing and return this data.

2. Each process must interact with other elements of the system, transmitting or receiving data.

3. Each process, data flow, data store, and external entity must be uniquely labeled and identified.

Let's build a DFD diagram:

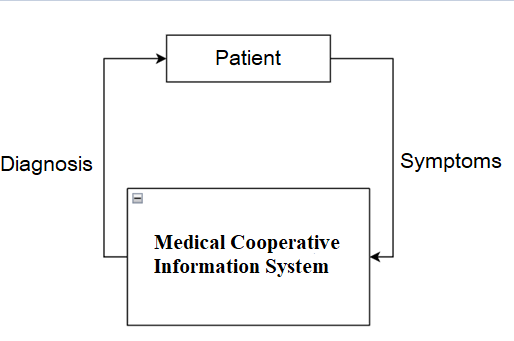


Fig. 2.1. (zero-level DFD diagram)

Conceptual modeling

The task of conceptual modeling is to describe the main objects of the database and the relationships between them. More detailed information can be seen in the following tables, which are built on the basis of database objects and the relationships between them.

Entity table:

|  |  |  |  |
| --- | --- | --- | --- |
| The name of the entity | Designation | Type | Primary key |
| Patient | Patient | Non-Associative | PatID |
| Doctor | Doctor | Non-Associative | DocID |
| Doctor-patient | DoctorForPatient | Associative | DocForPatID |
| Reception | Appointment | Associative | AppID |
| Medicine | Medicine | Non-Associative | MedID |

Tables of attributes:

|  |
| --- |
| Patient |

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Column Name | Condensed table | Nullable |
| Index | PatID | int | No |
| Name | PatName | nvarchar(50) | No |
| Surname | PatSurname | nvarchar(50) | No |
| Patronymic | PatPatronymic | nvarchar(50) | No |
| Address | PatAddress | nvarchar(50) | No |
| Age | PatAge | int | No |

|  |  |  |  |
| --- | --- | --- | --- |
| Doctor | | | |
| Attribute | Column Name | Condensed table | Nullable |
| Index | DocID | int | No |
| Name | DocName | nvarchar(50) | No |
| Surname | DocSurname | nvarchar(50) | No |
| Patronymic | DocPatronymic | nvarchar(50) | No |
| Specialty | DocSpecialty | nvarchar(50) | No |
| Position | DocPosition | nvarchar(50) | No |

|  |  |  |  |
| --- | --- | --- | --- |
| DoctorForPatient | | | |
| Attribute | Column Name | Condensed table | Nullable |
| Index | DocforpatId | int | No |
| Doctor index | DocId | int | No |
| Patient index | PatId | int | No |
| Name of the patient | DocforpatName | nvarchar(50) | No |
| Gender of the patient | DocforpatSex | nvarchar(50) | No |
| Patient's birthday | DocforpatBirth | date | No |
| Address | DocforpatAddress | nvarchar(50) | No |

|  |  |  |  |
| --- | --- | --- | --- |
| Appointment | | | |
| Attribute | Column Name | Condensed table | Nullable |
| Index | AppID | int | No |
| Date | AppDate | date | No |
| Place | AppPlace | nvarchar(50) | No |
| Symptom | AppSymptom | nvarchar(50) | No |
| Diagnosis | AppDiagnosis | nvarchar(50) | No |
| Patient index | PatId | int | No |

|  |  |  |  |
| --- | --- | --- | --- |
| Doctor index | DocId | int | No |
| Index of Medicines | MedId | int | No |
| Number of calls | DocNumCall | int | No |

|  |  |  |  |
| --- | --- | --- | --- |
| Medicine | | | |
| Attribute | Column Name | Condensed table | Nullable |
| Index | MedID | int | No |
| Name | MedName | nvarchar(50) | No |
| Method of reception | MedMethod | nvarchar(400) | No |
| Side effects | MedEffects | nvarchar(400) | No |
| Description of the intended | MedDescribe | nvarchar(400) | No |

Таблиця зв’язків:

|  |  |  |
| --- | --- | --- |
| Communication | Type of Communication | Attributes |
| Patient - DoctorForPatient | 1:1 | PatID, PattId |
| Doctor - DoctorForPatient | 1:M | DocID, DocId |
| Patient - Appointment | 1:M | PatID; PatId |
| Doctor - Appointment | 1:M | DocID; DocId |
| Medicine - Appointment | 1:M | MedID; MedId |

Scheme:

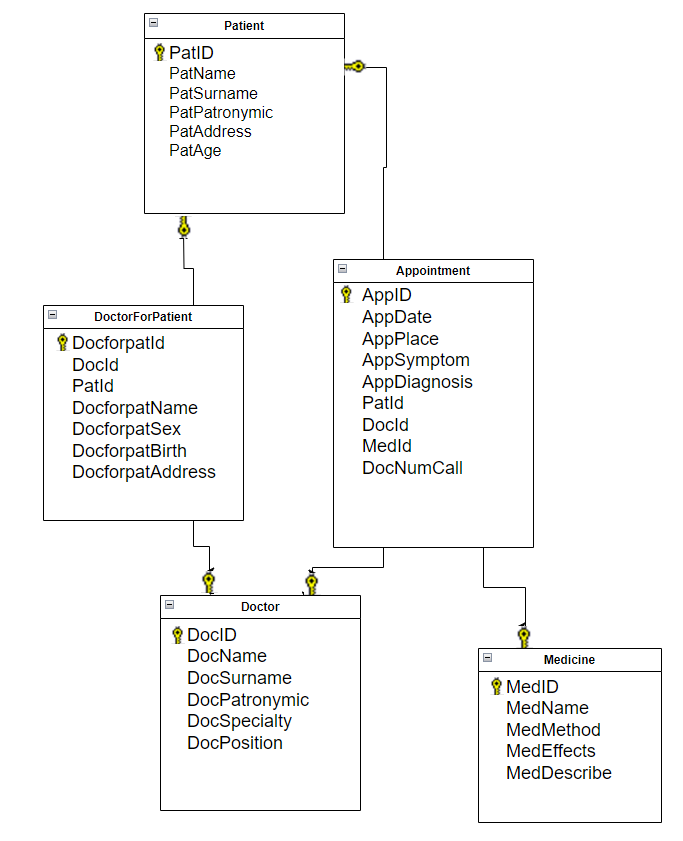


Fig. 3.1. (scheme in the form of an ER diagram)

Relational database model

A relational database model is an approach to data management that uses a tabular structure to represent and organize data. It is based on the mathematical concepts of set theory and relation theory. The main components of the relational model are structural, manipulative and holistic;

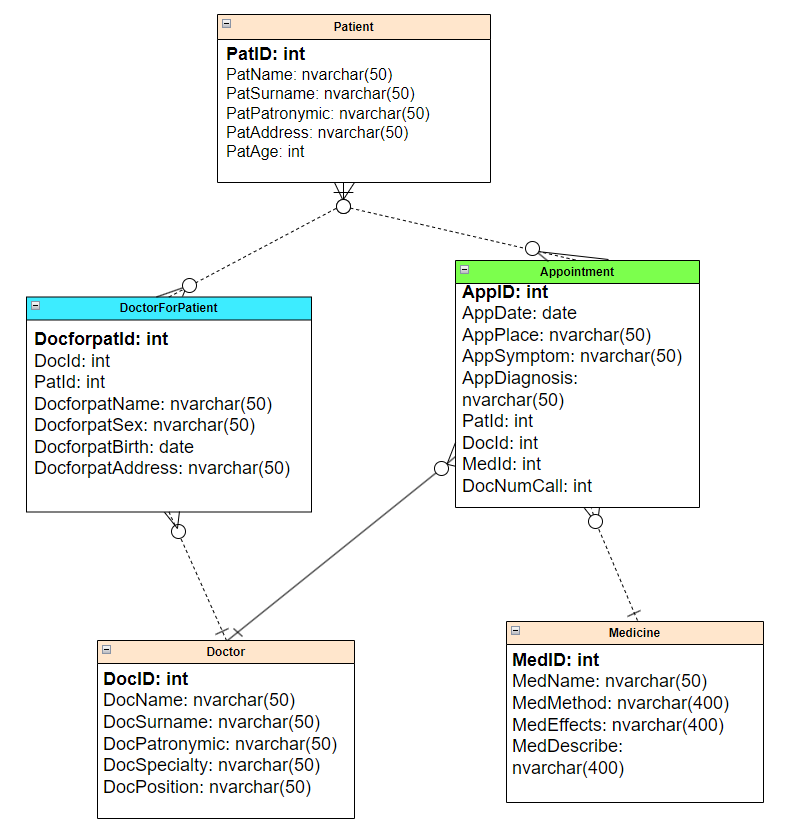
In detail, the structural component of the model establishes that the main data structure is a normalized n-ary table. The manipulation component of the model describes two key data processing tools — relational algebra and relational calculus. The integrity part of the model defines the requirements for the integrity of objects and the integrity of relationships.

Fig. 4.1. (Martin diagram)

Physical model of the database

Let's start creating tables in the Microsoft SQL Server database. Next, the program code for creating database tables and establishing relations between relations will be given.

Patient Table

|  |
| --- |
| Patient |

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Column Name | Condensed table | Nullable |
| Index | PatID | int | No |
| Name | PatName | nvarchar(50) | No |
| SurName | PatSurname | nvarchar(50) | No |
| Patronymic | PatPatronymic | nvarchar(50) | No |
| Address | PatAddress | nvarchar(50) | No |
| Age | PatAge | int | No |

Code:

CREATE TABLE Patient(

PatID INT IDENTITY PRIMARY KEY,

PatSurname NVARCHAR(50) NOT NULL,

PatName NVARCHAR(50) NOT NULL,

PatPatronymic NVARCHAR(50) NOT NULL,

PatAddress NVARCHAR(50) NOT NULL,

PatAge INT NOT NULL,)

Doctor table

|  |  |  |  |
| --- | --- | --- | --- |
| Doctor | | | |
| Attribute | Column Name | Condensed table | Nullable |
| Index | DocID | int | No |
| Name | DocName | nvarchar(50) | No |
| Surname | DocSurname | nvarchar(50) | No |
| Patronymic | DocPatronymic | nvarchar(50) | No |
| Specialty | DocSpecialty | nvarchar(50) | No |
| Position | DocPosition | nvarchar(50) | No |

Код:

CREATE TABLE Doctor(

DocID INT IDENTITY PRIMARY KEY,

DocName NVARCHAR(50) NOT NULL,

DocSurname NVARCHAR(50) NOT NULL,

DocPatronymic NVARCHAR(50) NOT NULL,

DocSpecialty NVARCHAR(50) NOT NULL,

DocPosition NVARCHAR(50) NOT NULL,

)

DoctorForPatient table

|  |  |  |  |
| --- | --- | --- | --- |
| DoctorForPatient | | | |
| Attribute | Column Name | Condensed table | Nullable |
| Index | DocforpatId | int | No |
| Doctor index | DocId | int | No |
| Patient index | PatId | int | No |
| Name of the patient | DocforpatName | nvarchar(50) | No |
| Gender of the patient | DocforpatSex | nvarchar(50) | No |
| Patient's birthday | DocforpatBirth | date | No |
| Address | DocforpatAddress | nvarchar(50) | No |

Code:

CREATE TABLE DoctorForPatient(

DocforpatID INT IDENTITY PRIMARY KEY,

DocId INT NOT NULL,

PatId INT NOT NULL,

DocforpatName NVARCHAR(50) NOT NULL,

DocforpatSex NVARCHAR(50) NOT NULL,

DocforpatBirth date NOT NULL,

DocforpatAddress NVARCHAR(50) NOT NULL

)

Appointment table.

|  |  |  |  |
| --- | --- | --- | --- |
| Appointment | | | |
| Attribute | Column Name | Condensed table | Nullable |
| Index | AppID | int | No |
| Date | AppDate | date | No |
| Place | AppPlace | nvarchar(50) | No |
| Symptom | AppSymptom | nvarchar(50) | No |
| Diagnosis | AppDiagnosis | nvarchar(50) | No |
| Patient index | PatId | int | No |

|  |  |  |  |
| --- | --- | --- | --- |
| Doctor index | DocId | int | No |
| Index of Medicines | MedId | int | No |
| Number of calls | DocNumCall | int | No |

Code:

CREATE TABLE Appointment(

AppID INT IDENTITY PRIMARY KEY,

AppDate DATE NOT NULL,

AppPlace NVARCHAR(50) NOT NULL,

AppSymptom NVARCHAR(50) NOT NULL,

AppDiagnosis NVARCHAR(50) NOT NULL,

PatId INT NOT NULL,

DocId INT NOT NULL,

MedId INT NOT NULL,

DocNumCall INT NOT NULL

)

Medicine table

|  |  |  |  |
| --- | --- | --- | --- |
| Medicine | | | |
| Attribute | Column Name | Condensed table | Nullable |
| Index | MedID | int | No |
| Name | MedName | nvarchar(50) | No |
| Method of reception | MedMethod | nvarchar(400) | No |
| Side effects | MedEffects | nvarchar(400) | No |
| Description of the intended | MedDescribe | nvarchar(400) | No |

Code:

CREATE TABLE Medicine(

MedID INT IDENTITY PRIMARY KEY,

MedName NVARCHAR(50) NOT NULL,

MedMethod NVARCHAR(400) NOT NULL,

MedEffects NVARCHAR(400) NOT NULL,

MedDescribe NVARCHAR(400) NOT NULL

)

Implementing cascading communication triggers:

Patient - DoctorForPatient:

ALTER TABLE DoctorForPatient

ADD CONSTRAINT DocforpatPatientForeign

FOREIGN KEY (PatId) REFERENCES Patient(PatID)

ON DELETE CASCADE;

Doctor - DoctorForPatient:

ALTER TABLE DoctorForPatient

ADD CONSTRAINT DocforpatDoctorForeign

FOREIGN KEY (DocId) REFERENCES Doctor(DocID)

ON DELETE CASCADE;

Patient - Appointment:

ALTER TABLE Appointment

ADD CONSTRAINT AppointmentPatientForeign

FOREIGN KEY (PatId) REFERENCES Patient(PatID)

ON DELETE CASCADE;

Doctor - Appointment:

ALTER TABLE Appointment

ADD CONSTRAINT AppointmentDoctorForeign

FOREIGN KEY (DocId) REFERENCES Doctor(DocID)

ON DELETE CASCADE;

Medicine - Appointment:

ALTER TABLE Appointment

ADD CONSTRAINT MedicinePatientForeign

FOREIGN KEY (MedId) REFERENCES Medicine(MedID)

ON DELETE CASCADE;

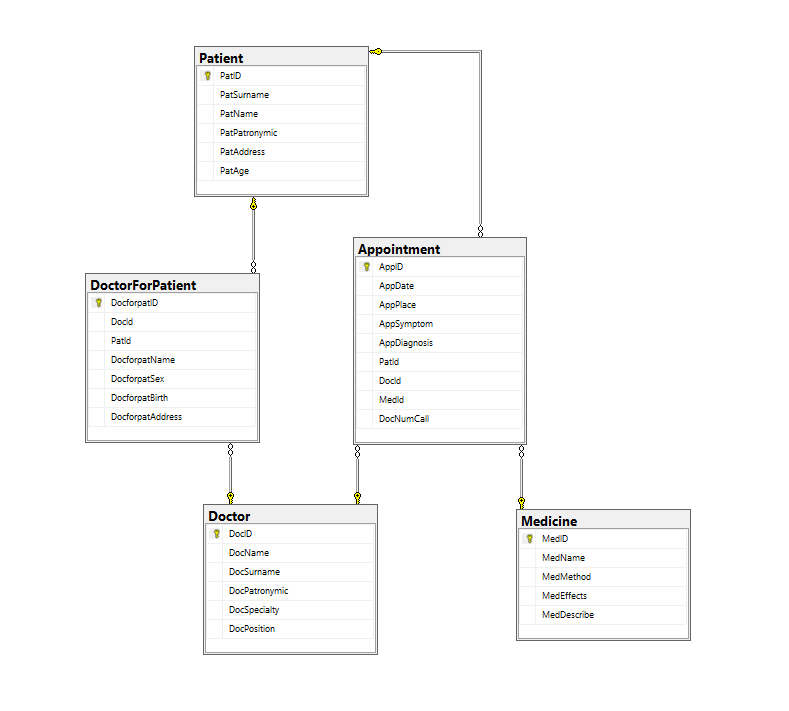


Figure 5.1 (ER diagram from SQL Server Management Studio)

Implementation of tasks

In this point, the necessary requests will be implemented in the form of procedures for the operation of the database.

1) determine the number of calls on this day based on a given date;

GO

CREATE PROC CallPerDay

@Day date

AS

SELECT SUM(DocNumCall)

AS Calls FROM Appointment

WHERE AppDate = @Day

EXEC CallPerDay '2023-12-15'

Performance result:

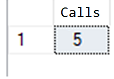


Fig. 6.1.

2) allow to determine the number of patients who fell ill with this disease;

GO

CREATE PROC IllCount

@Diagnosis NVARCHAR(50)

AS

SELECT COUNT(PatId)

AS Num FROM Appointment

WHERE AppDiagnosis = @Diagnosis

EXEC IllCount 'Blood Cancer'

Performance result:

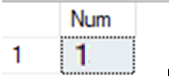


Fig. 6.2.

3) to determine the order of the side effect of the prescribed medication;

GO

CREATE PROC MedEffect

@Effect NVARCHAR(400)

AS

SELECT \* FROM Medicine

WHERE Medicine.MedEffects = @Effect

EXEC MedEffect 'Possible side effects may include: Irritation of the gastrointestinal tract. Headache. Gastrointestinal disturbances. Allergic reactions.'

Performance result:

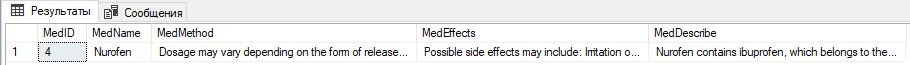


Fig. 6.3.

4) provide the possibility of adding new drugs with a description of their properties in the database.

GO

CREATE PROC MedicineAdd

@MedName NVARCHAR(50),

@MedMethod NVARCHAR(400),

@MedEffects NVARCHAR(400),

@MedDescribe NVARCHAR (400)

AS

INSERT INTO

Med (MedName, MedMethod, MedEffects, MedDescribe)

VALUES (@MedName, @MedMethod, @MedEffects, @MedDescribe)

EXEC AddPryp 'No-Spa',

'Always follow the doctors recommendations or the instructions provided with the medication.',

'Nausea. Dizziness. Allergic reactions. Gastrointestinal disturbances.',

'No-Spa contains drotaverine hydrochloride, a smooth muscle relaxant. It is used to relieve smooth muscle spasms, particularly in the gastrointestinal and genitourinary tracts.'

Performance result:

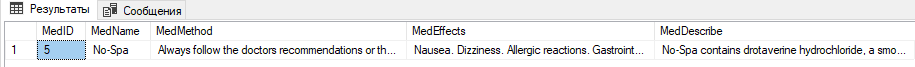


Fig. 6.4.

Conclusions

This work contributed to the development of a database for a medical cooperative using Microsoft SQL Server software and the Transact-SQL language. In the process of performing the work, I increased my theoretical knowledge and successfully put it into practice.

First, an analysis of the subject area, in particular of the medical cooperative, was carried out. Prototypes of tables were developed, which were further implemented in the process of the following stages. Tables were created for entities, attributes and relationships, which were carefully analyzed and further implemented when creating a database. Secondly, I studied the system of constructing entity-relationship diagrams (DFD), their basic rules and successfully used this knowledge in practice. A database schema was also constructed and a Martin diagram was created. Thirdly, the code was written, which was used to implement the creation of tables in the database and the execution of the necessary queries.

I completed the following tasks:

1. Determination of the number of calls for a certain date.

2. Determining the number of patients with a certain disease.

3. Finding side effects for a given medication.

4. The possibility of adding new drugs with the indication of their properties in the database.

Cascading triggers were also implemented to ensure cascading changes in tables.

As a result of this course work, a database of the medical cooperative was created. It works well, both on a small scale and on a large scale. Medicine is constantly developing, and along with it, the effective storage of accumulated information must develop, and therefore, the infrastructure of databases in this area must also develop.

List of used literature

1. <https://support.microsoft.com/uk-ua/office/%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D1%96-%D0%B2%D1%96%D0%B4%D0%BE%D0%BC%D0%BE%D1%81%D1%82%D1%96-%D0%BF%D1%80%D0%BE-%D0%B1%D0%B0%D0%B7%D0%B8-%D0%B4%D0%B0%D0%BD%D0%B8%D1%85-a849ac16-07c7-4a31-9948-3c8c94a7c204>
2. [https://www.w3schools.com/sql/sql\_intro.asp](https://www.w3schools.com/sql/sql_intro.asp%20)
3. <https://uk.wikipedia.org/wiki/%D0%94%D1%96%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%B0_%D0%BF%D0%BE%D1%82%D0%BE%D0%BA%D1%96%D0%B2_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85>
4. [https://www.postgresql.org/docs/books/](https://www.postgresql.org/docs/books/%20)
5. <https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85>