**Лабораторная работа №5**

**Знакомство с языком программирования SQL. Основы написания запросов к базам данных**

**Цель работы**: ознакомиться с языком программирования SQL. Научиться писать запросы БД.

**SQL** (Structured Query Language — «язык структурированных запросов») — декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

**SQL** — это язык программирования структурированных запросов (SQL, Structured Query Language), который используется в качестве эффективного способа сохранения данных, поиска их частей, обновления, извлечения из базы и удаления.

Группы операторов SQL

1. DDL (Data Definition Language)

Представляют собой группу операторов для определения данных. Они работают с целыми таблицами. Такие операторы SQL используются в тех случаях, когда нужно внести в базу новую таблицу или, напротив, удалить старую. Они включают в себя следующие командные слова:

CREATE — создание нового объекта в существующей базе.

ALTER — изменение существующего объекта.

DROP — удаление объекта из базы.

1. DML (Data Manipulation Language)

Эти операторы языка SQL предназначены для манипуляции данными. С их помощью меняется наполнение таблиц. Они позволяют изменять значение строк, столбцов и прочих атрибутов. Такие операторы SQL, например, позволяют удалить информацию о сотруднике, который больше не работает в компании, или исправить данные действующих специалистов. Эти операторы SQL представлены следующими командными словами:

SELECT — позволяет выбрать данные в соответствии с необходимым условием.

INSERT — осуществляют добавление новых данных.

UPDATE — производит замену существующих данных.

DELETE — удаление информации.

Существуют и другие группы, но пока остановимся на этих.

Основные операторы для начала работы:

SELECT — выбор данных.

FROM — источник информации, откуда брать данные.

JOIN — добавление таблиц.

WHERE — при каком условии.

GROUP BY — сформируй группу данных по заданному признаку.

ORDER BY — сортировка данных по нужному признаку.

LIMIT —количество результатов.

; — конец предложения

Пример SQL запроса, который создает таблицу со столбцами id, name, gender. При этом указываются атрибуты этих столбцов integer, text, text – типы данных, которые будут содержаться. Primary key – этот столбец должен быть всегда уникальным. Not null столбец всегда должен содержать значение.

CREATE TABLE students (

id INTEGER PRIMARY KEY,

name TEXT NOT NULL,

gender TEXT NOT NULL

);

Читается как: СОЗДАТЬ ТАБЛИЦУ студенты, где (

Столбец id ТИП ЧИСЛО ОСНОВНОЙ КЛЮЧ,

Столбец name ТИП ТЕКСТ НЕ НОЛЬ,

Столбец gender ТИП ТЕКСТ НЕ НОЛЬ,

)

Запрос, который добавляет в таблицу строку со значениями id=1, name=Ryan, gender=M

INSERT INTO students VALUES (1, 'Ryan', 'M');

Читается как: ДОБАВИТЬ В таблицу студенты ЗНАЧЕНИЯ (1, 'Ryan', 'M').

Запрос, который возвращает данные из таблицы, в соответствии с условием

SELECT \* FROM students WHERE gender = 'F';

Читается как: ВЫБРАТЬ ВСЕ ИЗ таблицы студенты, ГДЕ пол = F.

Для обновления используется запрос вида:

UPDATE students SET name = 'Lisa', gender = 'F' WHERE id = 1;

Читается как ОБНОВИТЬ таблицу студенты УСТАНОВИТЬ столбец имя = Лиза, пол = F, ГДЕ id = 1;

Использование JOIN.

Операция LEFT JOIN создает левое внешнее соединение. С помощью левого внешнего соединения выбираются все записи первой (левой) таблицы, даже если они не соответствуют записям во второй (правой) таблице.

Создадим вторую таблицу, где будут указаны профессии

CREATE TABLE job (

id INTEGER PRIMARY KEY,

name TEXT NOT NULL

);

Добавим несколько записей в таблицу job

INSERT INTO job VALUES (1, 'none');

INSERT INTO job VALUES (2, 'driver');

INSERT INTO job VALUES (3, 'server');

Отредактируем первую таблицу, вставив столбец, который будет использоваться для соединения.

bysy INTEGER NOT NULL

Теперь можно получить результирующую таблицу

SELECT \* FROM students

LEFT JOIN job ON students.bysy=job.id;

Читается как: ВЫБРАТЬ ВСЕ ИЗ таблицы студенты ПРИСОЕДЕНИТЬ таблицу работа где столбец занятость из таблицы студенты равен столбцу id из таблицы работа.

Мы можем выбирать не все столбцы из таблицы, а только некоторые, указывая их

SELECT students.id, students.gender FROM students;

**База данных (БД)** –это совокупность массивов и файлов данных, организованная по определённым правилам, предусматривающим стандартные принципы описания, хранения и обработки данных независимо от их вида.

Для выполнения работы рекомендуется использовать онлайн компилятор SQL. https://www.mycompiler.io/new/sql

При желании можно использовать СУБД для реляционных БД, например PHP MyAdmin или MySQL.

Задание на лабораторную работу:

1. Создать таблицу, содержащую список членов семьи (друзей, знакомых, кого угодно минимум четыре строки), пол, возраст.

*CREATE TABLE name (*

*id INT PRIMARY KEY,*

*name VARCHAR(50),*

*gender VARCHAR(10),*

*age INT*

*);*

*INSERT INTO name (id, name, gender, age) VALUES*

*(1, 'Николай', 'Мужской', 17),*

*(2, 'Нурислам', 'Мужской', 18),*

*(3, 'Иван', 'Мужской', 25),*

*(4, 'Артем', 'Мужской', 100)*

1. *;*Изменить значения строки в таблице членов семьи, где id = 3.

*UPDATE name*

*SET name = 'Сергей', gender = 'Мужской', age = 12*

*WHERE id = 3;*

1. Создать таблицу, содержащую список возможных увлечений/хобби.

*CREATE TABLE Hobbies (*

*id INT PRIMARY KEY,*

*hobby VARCHAR(50)*

*);*

*INSERT INTO Hobbies (id, hobby) VALUES*

*(1, 'Игры'),*

*(2, 'Музыка'),*

*(3, 'Гулять'),*

*(4, 'Тикток');*

1. Соединить добавить членам семьи увлечения/хобби из второй таблицы. Отобразить результат. Не показывать пользователю поле, используемое для соединения таблиц.

*ALTER TABLE names ADD hobby\_id INT;*

*UPDATE names SET hobby\_id = 1 WHERE id = 1;*

*UPDATE names SET hobby\_id = 2 WHERE id = 2;*

*UPDATE names SET hobby\_id = 3 WHERE id = 3;*

*UPDATE names SET hobby\_id = 4 WHERE id = 4;*

*SELECT*

*f.name,*

*f.gender,*

*f.age,*

*h.hobby*

*FROM*

*Names f*

*JOIN*

*Hobbies h ON f.hobby\_id = h.id;*

1. Сгруппировать полученную таблицу по увлечениям.

*SELECT*

*Hobbies.hobby,*

*COUNT(Names.id) AS member\_count*

*FROM*

*Names*

*JOIN*

*Hobbies ON names.hobby\_id = Hobbies.id*

*GROUP BY*

*Hobbies.hobby;*

1. Удалить строку из таблицы члены семьи.

*DELETE FROM name*

*WHERE id = 4;*

**Вопросы для самопроверки**

1) Что такое SQL? Назовите основные функции и свойства.

2) Группы операторов SQL. Принцип разделения, выполняемые функции.

3) Виды баз данных, их функции и особенности. Сфера применения разных типов БД.

**1) Что такое SQL? Назовите основные функции и свойства.**

**SQL (Structured Query Language)** — это стандартный язык программирования, используемый для управления и манипуляции реляционными базами данных. SQL позволяет пользователям выполнять различные операции с данными, такие как создание, чтение, обновление и удаление (CRUD).

**Основные функции SQL:**

* **Создание и изменение структуры базы данных:** Используя команды **CREATE**, **ALTER**, **DROP**, можно создавать и изменять таблицы, индексы и другие объекты базы данных.
* **Манипуляция данными:** Команды **INSERT**, **UPDATE**, **DELETE** позволяют добавлять, изменять и удалять данные в таблицах.
* **Запрос данных:** Команда **SELECT** используется для извлечения данных из одной или нескольких таблиц, с возможностью фильтрации, сортировки и группировки.
* **Управление правами доступа:** Команды **GRANT** и **REVOKE** позволяют управлять правами пользователей на доступ к данным и объектам базы данных.
* **Транзакции:** SQL поддерживает транзакции, что позволяет выполнять группы операций как единое целое, обеспечивая целостность данных.

**Свойства SQL:**

* **Стандартизированный язык:** SQL является стандартом ANSI и ISO, что обеспечивает совместимость между различными системами управления базами данных (СУБД).
* **Декларативный подход:** Пользователи описывают, что они хотят получить, а не как это сделать, что упрощает работу с данными.
* **Поддержка реляционной модели:** SQL работает с реляционными базами данных, где данные организованы в таблицы с четкими связями между ними.

**2) Группы операторов SQL. Принцип разделения, выполняемые функции.**

SQL-операторы можно разделить на несколько групп в зависимости от их назначения:

1. **DML (Data Manipulation Language)** — Язык манипуляции данными:
   * **INSERT**: добавление новых записей.
   * **UPDATE**: изменение существующих записей.
   * **DELETE**: удаление записей.
   * **SELECT**: извлечение данных.
2. **DDL (Data Definition Language)** — Язык определения данных:
   * **CREATE**: создание новых объектов (таблиц, индексов и т.д.).
   * **ALTER**: изменение структуры существующих объектов.
   * **DROP**: удаление объектов.
3. **DCL (Data Control Language)** — Язык управления данными:
   * **GRANT**: предоставление прав доступа.
   * **REVOKE**: отзыв прав доступа.
4. **TCL (Transaction Control Language)** — Язык управления транзакциями:
   * **COMMIT**: подтверждение транзакции.
   * **ROLLBACK**: откат транзакции.
   * **SAVEPOINT**: создание точки сохранения в транзакции.

**3) Виды баз данных, их функции и особенности. Сфера применения разных типов БД.**

Существует несколько видов баз данных, каждая из которых имеет свои функции, особенности и области применения:

1. **Реляционные базы данных (RDBMS)**:
   * **Функции:** Хранение данных в таблицах с четкими связями между ними, поддержка SQL.
   * **Особенности:** Данные организованы в строки и столбцы, поддержка транзакций и целостности данных.
   * **Сфера применения:** Финансовые системы, системы управления запасами, CRM-системы.
2. **Нереляционные базы данных (NoSQL)**:
   * **Функции:** Хранение данных в формате, отличном от таблиц (документы, графы, ключ-значение и т.д.).
   * **Особенности:** Гибкость схемы, высокая производительность при работе с большими объемами данных.
   * **Сфера применения:** Социальные сети, системы рекомендаций, большие данные.
3. **Объектно-ориентированные базы данных (OODBMS)**:
   * **Функции:** Хранение данных в виде объектов, как в объектно-ориентированном программировании.
   * **Особенности:** Поддержка сложных данных и их взаимосвязей.
   * **Сфера применения:** CAD/CAM системы, мультимедийные приложения.
4. **Графовые базы данных**:
   * **Функции:** Хранение данных в виде графов, где узлы представляют объекты, а ребра — связи между ними.
   * **Особенности:** Эффективное выполнение запросов, связанных с отношениями между данными.
   * **Сфера применения:** Социальные сети, рекомендательные