



ЯЗЫК СТРУКТУРИРОВАННЫХ ЗАПРОСОВ SQL

Лекция 6

Доцент каф. ВТиКГ Данилова Е.В.

План

1. Введение в SQL
2. Компонент DDL языка SQL
3. Компонент DML языка SQL



Введение в SQL

1

3

Начало 1970-х годов

Исследовательская лаборатория компании IBM

Экспериментальная реляционная СУБД IBM System R

Специальный язык SEQUEL

Structured English QUERy Language — «структурированный английский язык запросов»

Переименование SEQUEL в SQL

1986 год – принятие первого стандарта языка SQL
Институтом стандартизации ANSI (American National Standards Institute)

Официальное произношение [ˌes kjuːˈel] – эс-кью-эл

Многие англоязычные специалисты читают SQL как «сиквел», по-русски говорят «эс-ку-эль» или используют жаргонизм «скуль»

Стандартизация

Начало 1980-х годов – несколько вариантов СУБД от разных производителей со своей собственной реализацией языка запросов SQL.

Необходимость в разработке стандарта языка SQL.

Стандарт SQL разрабатывался Международной организацией по стандартизации (ISO) и Американским национальным институтом стандартов (ANSI).

1986 год – первая версия стандарта SQL: стандарт ANSI-86, или стандарт SQL-86, или SQL1 (неофициальное название).

1989 год – стандарт ANSI-89 (стандарт SQL-89).

1992 год – стандарт ANSI-92 (стандарт SQL-92).

1999 год – стандарт ANSI-99 (стандарт SQL-99 или SQL3).

2003 год – SQL:2003

2006, 2008, 2001, 2016, 2019 года

Определение 1

SQL – информационно-логический язык, предназначенный для описания, изменения и извлечения данных, хранимых в реляционной базе данных

Определение 2

SQL – мощный язык для управления СУБД и взаимодействия с ней

Возможности SQL

- **Определение данных.** SQL позволяет пользователю определить структуру и организацию хранимых данных и взаимоотношения между элементами сохраненных данных.
- **Выборка данных.** SQL дает пользователю или приложению возможность извлекать из базы содержащиеся в ней данные и пользоваться ими.
- **Обработка данных.** SQL позволяет пользователю или приложению изменять базу данных, т.е. добавлять в нее новые данные, а также удалять или обновлять уже имеющиеся в ней данные.

Возможности SQL

- **Управление доступом.** С помощью SQL можно ограничить возможности пользователя по выборке, добавлению и изменению данных и защитить их от несанкционированного доступа.
- **Совместное использование данных.** SQL применяется для координации совместного использования данных пользователями, работающими одновременно, с тем чтобы изменения, вносимые одним пользователем, не приводили к непреднамеренному уничтожению изменений, вносимых примерно в то же время иным пользователем.
- **Целостность данных.** SQL позволяет обеспечить целостность базы данных, защищая ее от разрушения из-за несогласованных изменений или отказа системы.

Определение 3

SQL – язык программирования (не тьюринг-полный)

Тьюринг-полный язык программирования – язык программирования, на котором можно реализовать любую вычислимую функцию

SQL – «подъязык» баз данных, содержит около сорока инструкций для решения задач управления БД.

Инструкции SQL могут быть встроены в другой язык (COBOL, С и др.), расширяют его, давая возможность получать доступ к базам данных.

Из языков С, С++ или Java инструкции SQL можно посылать СУБД в явном виде, используя интерфейс на уровне вызовов функций (call-level interface) или отправляя сообщения по вычислительной сети.

Определение 4

SQL – это слабо структурированный язык (по сравнению с высокоструктурированными языками: C, Pascal или Java)

SQL – это декларативный (описательный), а не процедурный язык. Описывает, что пользователь хочет от компьютера, а не как компьютер должен это сделать.

Отсутствуют инструкции:

- IF (для проверки выполнения условия);
- GOTO, DO или FOR (для управления потоком выполнения).

Инструкции SQL описывают, как организован набор данных или какие данные должны быть выбраны или добавлены в базу данных.

Последовательность шагов для решения этих задач определяется самой СУБД.

Формы языка SQL

■ Интерактивный SQL

Позволяет конечному пользователю в интерактивном режиме выполнять SQL-операторы. Все СУБД предоставляют инструментальные средства для работы с базой данных в интерактивном режиме.

■ Статический SQL

Может реализовываться как встроенный SQL или модульный SQL. Операторы статического SQL определены в момент компиляции программы.

■ Динамический SQL

Позволяет формировать операторы SQL во время выполнения программы.

■ Встроенный SQL

Позволяет включать операторы SQL в код программы на другом языке программирования.

Функции SQL

- **SQL – интерактивный язык запросов.** Для получения данных и вывода их на экран пользователи вводят команды SQL в интерактивных программах. Это удобный способ выполнения специальных запросов.
- **SQL – язык программирования баз данных.** Чтобы получить доступ к базе данных, программисты вставляют в свои прикладные программы команды SQL. Эта методика используется как в программах, написанных пользователями, так и в служебных программах баз данных (например, в таких, как генераторы отчетов).
- **SQL – язык администрирования баз данных.** Администратор базы данных, находящейся на рабочей станции или на сервере, использует SQL для определения структуры базы данных и управления доступом к данным.

Функции SQL

- **SQL – язык создания приложений «клиент/сервер».** В программах для персональных компьютеров SQL используется как средство организации связи по сети с серверами баз данных, в которых хранятся совместно используемые данные.
- **SQL – язык доступа к данным в Интернете.** Веб-серверы Интернета, взаимодействующие с корпоративными данными и серверами приложений Интернета, используют SQL в качестве стандартного языка доступа к корпоративным базам данных, зачастую путем внедрения SQL-доступа в языки сценариев (Python, PHP, Perl).

Функции SQL

- **SQL – язык распределенных баз данных.** В системах управления распределенными базами данных SQL помогает распределять данные между несколькими соединенными вычислительными системами. Программное обеспечение каждой системы посредством SQL связывается с другими системами, посылая им запросы на доступ к данным.
- **SQL – язык шлюзов баз данных.** В вычислительных сетях с различными СУБД SQL часто используется в шлюзовой программе, которая позволяет СУБД одного типа связываться с СУБД другого типа.

Преимущества SQL

- Независимость от конкретных СУБД
- Межплатформенная переносимость
- Наличие стандартов
- Поддержка крупных производителей ПО (IBM, Microsoft и др.)
- Построение на реляционной модели
- Высокоуровневая структура, напоминающая естественный язык
- Возможность выполнения специальных интерактивных запросов
- Обеспечение программного доступа к базам данных
- Возможность различного представления данных

Преимущества SQL

- Полноценность в качестве языка, предназначенного для работы с базами данных
- Возможность динамического определения данных
- Поддержка архитектуры клиент/сервер
- Поддержка приложений уровня предприятия
- Расширяемость и поддержка объектно-ориентированных технологий
- Возможность доступа к данным в Интернет
- Интеграция с языком Java (протокол JDBC)
- Поддержка открытого кода
- Промышленная инфраструктура

Терминология

- **Инструкция** – содержит описание набора данных на языке SQL.

Инструкция = предложение1 + предложение2 + ...

Инструкции содержат специальные ключевые слова

```
SELECT AVG (AMOUNT)
FROM ORDERS
WHERE CUST = 2103;
```

```
SELECT CUST, SUM (AMOUNT)
FROM ORDERS
GROUP BY CUST;
```



SELECT

INSERT

UPDATE

MERGE

DELETE

CREATE TABLE

DROP TABLE

ALTER TABLE

CREATE VIEW

DROP VIEW

CREATE INDEX

DROP INDEX

CREATE SCHEMA

DROPE SCHEMA

CREATE DOMAIN

DROP DOMAIN

Управление доступом

GRANT	Предоставляет пользователю привилегии доступа
REVOKE	Отменяет пользовательские привилегии доступа
CREATE ROLE	Добавляет в базу данных новую роль
GRANT ROLE	Предоставляет роль, содержащую привилегии доступа
DROP ROLE	Удаляет роль из базы данных

Управление транзакциями

COMMIT	Завершает текущую транзакцию
ROLLBACK	Отменяет текущую транзакцию
SET TRANSACTION	Определяет характеристики доступа к данным в текущей транзакции
START TRANSACTION	Явно начинает новую транзакцию
SAVEPOINT	Устанавливает точку восстановления транзакции

Программный SQL

DECLARE

Определяет курсор запроса

EXPLAIN

Возвращает описание плана доступа к данным в запросе

OPEN

Открывает курсор для получения результатов запроса

FETCH

Извлекает строку из результатов запроса

CLOSE

Закрывает курсор

PREPARE

Подготавливает инструкцию SQL к динамическому выполнению

EXECUTE

Динамически выполняет инструкцию SQL

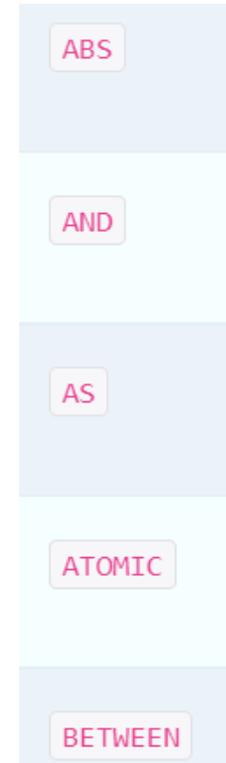
DESCRIBE

Описывает подготовленный запрос

Существует набор зарезервированных (а также незарезервированных) ключевых слов, которые используются в инструкциях SQL.

В соответствии со стандартом, зарезервированные ключевые слова нельзя использовать для именования объектов базы данных: таблицы, столбцы и пользователи.

Зарезервированные
ключевые слова



Незарезервированные
ключевые
слова



- **Команда** – ключевое слово, описывающее действие, выполняемое инструкцией.

CREATE, INSERT, DELETE, COMMIT и др.

- **Предложение** – описывает данные, с которыми работает инструкция, содержит уточняющую информацию о действии, выполняемом инструкцией. Может быть несколько предложений для одной команды.

Каждое предложение также начинается с ключевого слова WHERE (где), FROM (откуда), INTO (куда), HAVING (имеющий)

Одни предложения в инструкции являются обязательными, другие – нет.

Конкретная структура и содержимое предложения могут изменяться.

Многие предложения содержат имена таблиц или столбцов; некоторые могут содержать дополнительные ключевые слова, константы и выражения.



Каждое предложение SQL – это либо запрос данных из базы, либо обращение к базе данных, которое приводит к изменению данных в базе.

В соответствии с тем, какие изменения происходят в базе данных, различают следующие типы запросов:

- запросы на создание/изменение в базе данных новых/существующих объектов;
- запросы на получение данных;
- запросы на добавление новых данных (записей);
- запросы на удаление данных;
- обращения к СУБД.

Запросы делятся на:

- запросы, оперирующие самими таблицами (создание и изменение таблиц);
- запросы, оперирующие с отдельными записями (или строками таблиц) или их наборами (вставка новой строки, изменение значений полей строки или набора строк, удаление строки или набора строк, возвращение строки или набора строк).

Группы команд

- Команды определения данных (Data Definition Language, DDL)
- Команды манипуляции данными (Data Manipulation Language, DML)
- Команды определения доступа к данным (Data Control Language, DCL)
- Команды управления транзакциями (Transaction Control Language, TCL)

СУБД Microsoft Access поддерживает язык запросов Jet SQL, который частично соответствует стандарту ANSI-89 (уровень 1) и ANSI-92 (уровень 1).

При описании синтаксиса конструкций языка SQL используются следующие соглашения:

- в квадратные скобки [] заключены необязательные элементы (если элемент пропущен, то сами квадратные скобки в операторе SQL не используются);
- вертикальная черта | , разделяющая два элемента, указывает на то, что в операторе используется либо один элемент, либо второй (сама вертикальная черта в текст оператора также не включается);
- В фигурные скобки { } заключаются элементы, разделенные вертикальной чертой (иначе невозможно понять, какие именно элементы предложены на выбор);
- троеточие означает, что далее в операторе следует (в зависимости от контекста) выражение, либо повторяются элементы, указанные перед тремя точками.



Компонент DDL языка SQL

2

Инструкция CREATE TABLE.

Данная инструкция используется для создания новых таблиц.

Синтаксис:

```
CREATE [TEMPORARY] TABLE таблица (  
    поле_1 тип [(размер)] [NOT NULL] [WITH COMPRESSION |  
    WITH COMP] [индекс_1]  
    [, поле_2 тип [(размер)] [NOT NULL] [WITH  
    COMPRESSION | WITH COMP] [индекс_2]  
    [, ...]]  
    [, CONSTRAINT составной_индекс [, ...]])
```

Инструкция CREATE TABLE.

Описание параметров:

таблица имя создаваемой таблицы

поле_1 тип [(размер)] описание одного или нескольких полей создаваемой таблицы; описание включает имя поля, его тип и размер поля в знаках (только для текстовых и двоичных полей);

индекс1, индекс_2 предложение CONSTRAINT для создания простых индексов.

составной_индекс Предложение CONSTRAINT для создания составных индексов.

TEMPORARY Если указано ключевое слово TEMPORARY, то создается временная таблица, доступная только в текущем сеансе работы. При завершении сеанса, такая таблица автоматически удаляется.

NOT NULL Означает, что соответствующее поле обязательно должно содержать значение.

WITH COMPRESSION (синоним WITH COMP) Атрибут WITH COMPRESSION используется только с полями типа CHARACTER или MEMO. Если этот атрибут указан, то при сохранении поля выполняется сжатие Unicode.



Компонент DML языка SQL

3

DML (Data Manipulation Language) – язык манипулирования данными

Операторы языка DML:

- INSERT – добавить строки в таблицу;
- SELECT – отобразить строки из таблиц.
- UPDATE – изменить строки в таблице;
- DELETE – удалить строки в таблице;
- COMMIT – зафиксировать внесенные изменения;
- ROLLBACK – отменить внесенные изменения;

INSERT

Базовая структура оператора INSERT

INSERT INTO имя_таблицы VALUES
(значение_атрибута_1, значение_атрибута_2, ...)

Пример:

```
INSERT INTO Аптека VALUES (23, 'Аптека24/7',  
'Новосибирск', 'Ломоносова', 4, 'ЛО-0120-72-  
009832')
```

Особенности:

- содержимое строки/кортежа заключается в скобки;
- значения атрибутов символьного/строкового типа и типа дата/время заключаются в апострофы (');
- значения атрибутов разделяются запятыми.

INSERT

Примеры в MS Access:

Добавление строки в таблицу «Аптека»

```
INSERT INTO Аптека VALUES (23, "Аптека24",  
"Новосибирск", "Ломоносова", 4, "ЛО-  
0120-72-009832")
```

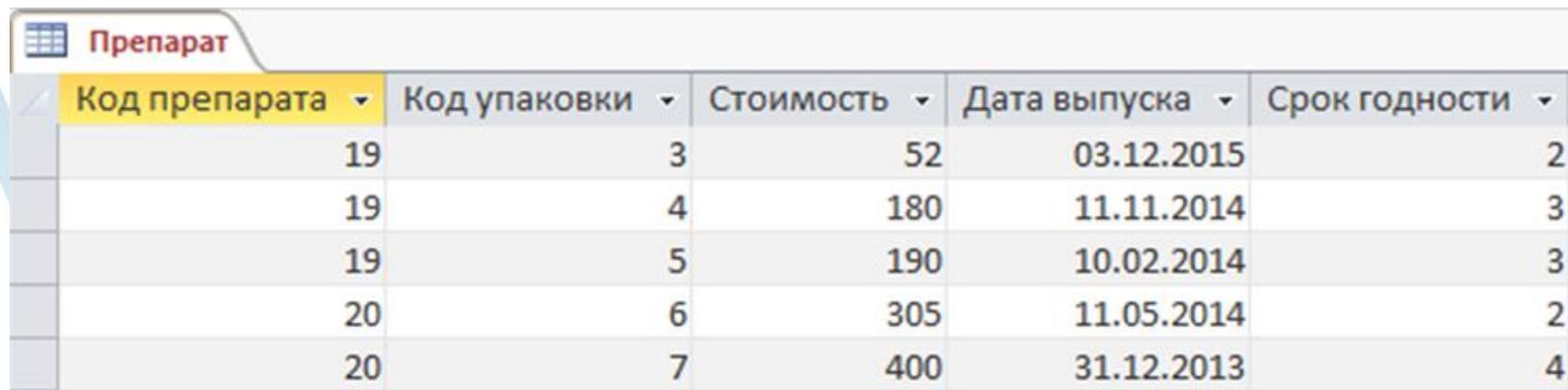
	Код аптеки	Название ап	Город	Улица	Номер дом	Лицензия
+	16	Вилаб	Хабаровск	Промышленн	5	ЛО-34-32-563288
+	17	Эколайф-ДВ	Хабаровск	Суворова	51	ЛО-10-00-009573
+	18	Семейная апт	Хабаровск	Кочнева	14	ЛО-00-17-772309
+	19	Фарком	Хабаровск	Космическая	3а	ЛО-51-94-918400
+	20	Артлайф	Хабаровск	Комсомольск	53	ЛО-01-82-503193
+	21	Фармаком	Уссурийск	Потемкина	4	ЛР-0000-54-9876
+	22	Аптифарм	Владивосток	Уссури	4	ЛО-0000-54-97748
+	23	Аптека24	Новосибирск	Ломоносова	4	ЛО-0120-72-00983

INSERT

Примеры в MS Access:

Добавление строки в таблицу «Препарат»:

```
INSERT INTO Препарат VALUES (19, 3, 52,  
#12/03/2015#, 2)
```



Код препарата	Код упаковки	Стоимость	Дата выпуска	Срок годности
19	3	52	03.12.2015	2
19	4	180	11.11.2014	3
19	5	190	10.02.2014	3
20	6	305	11.05.2014	2
20	7	400	31.12.2013	4

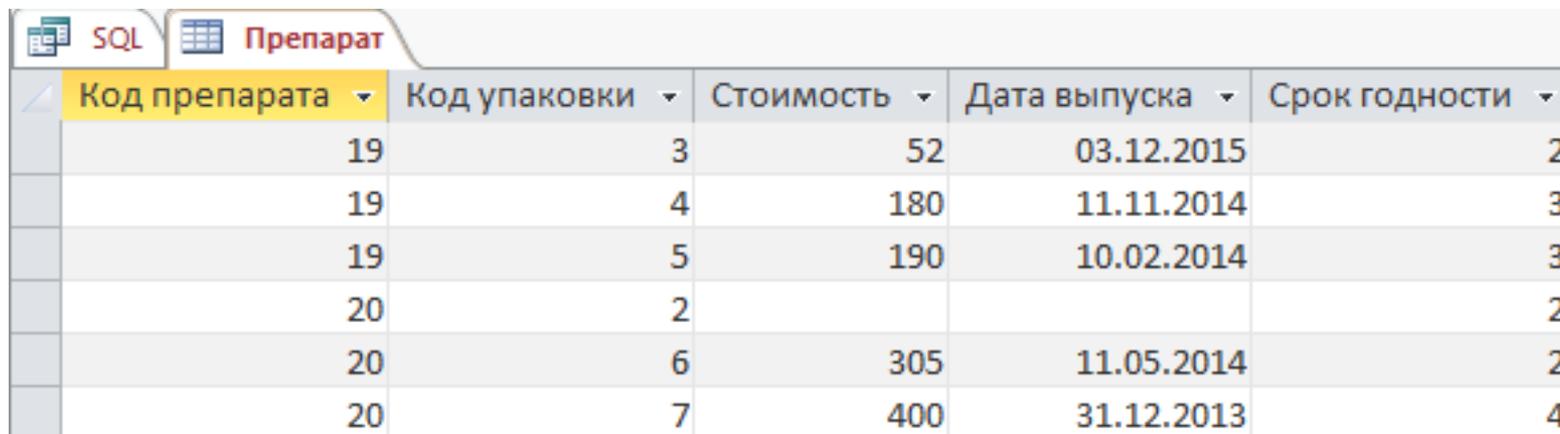
INSERT

Если значение одного или нескольких атрибутов нужно оставить пустым, то используется предикат NULL.

Примеры в MS Access:

Добавление строки с пустыми атрибутами в таблицу «Препарат»:

```
INSERT INTO Препарат VALUES (20, 2, null, null, 2);
```



Код препарата	Код упаковки	Стоимость	Дата выпуска	Срок годности
19	3	52	03.12.2015	2
19	4	180	11.11.2014	3
19	5	190	10.02.2014	3
20	2			2
20	6	305	11.05.2014	2
20	7	400	31.12.2013	4

UPDATE

Пример в MS Access:

UPDATE Аптека

SET [Номер дома] = 150 WHERE Город = "Новосибирск"

	Код аптеки	Название ап	Город	Улица	Номер дом	Лицензия
+	16	Вилаб	Хабаровск	Промышленн	5	ЛО-34-32-563288
+	17	Эколайф-ДВ	Хабаровск	Суворова	51	ЛО-10-00-009573
+	18	Семейная апт	Хабаровск	Кочнева	14	ЛО-00-17-772309
+	19	Фарком	Хабаровск	Космическая	3а	ЛО-51-94-918400
+	20	Артлайф	Хабаровск	Комсомольск	53	ЛО-01-82-503193
+	21	Фармаком	Уссурийск	Потемкина	4	ЛР-0000-54-9876
+	22	Аптифарм	Владивосток	Уссури	4	ЛО-0000-54-97748
+	23	Аптека24	Новосибирск	Ломоносова	150	ЛО-0120-72-00983

UPDATE

Если требуется изменить значение более одного атрибута в строке, нужно в инструкции отделять их друг от друга запятой.

Пример в MS Access:

UPDATE Аптека

SET [Название аптеки] = "Аптека 24/7",

[Номер дома] = 300,

Улица = "Менделеева"

WHERE Город = "Новосибирск";

Аптека						
	Код аптеки	Название аптеки	Город	Улица	Номер дома	Лицензия
+	20	Артлайф	Хабаровск	Комсомольск	53	ЛО-01-82-503193
+	21	Фармаком	Уссурийск	Потемкина	4	ЛР-0000-54-9876
+	22	Аптифарм	Владивосток	Уссури	4	ЛО-0000-54-97748
+	23	Аптека 24/7	Новосибирск	Менделеева	300	ЛО-0120-72-00983

DELETE

Базовая структура оператора DELETE

DELETE FROM имя_таблицы

WHERE значение_атрибута = значение

Пример в MS Access:

DELETE FROM Аптека WHERE Город = "Новосибирск"

Аптека						
	Код аптеки	Название аптеки	Город	Улица	Номер дома	Лицензия
+	16	Вилаб	Хабаровск	Промышленн	5	ЛО-34-32-563288
+	17	Эколайф-ДВ	Хабаровск	Суворова	51	ЛО-10-00-009573
+	18	Семейная аптека	Хабаровск	Кочнева	14	ЛО-00-17-772309
+	19	Фарком	Хабаровск	Космическая	3а	ЛО-51-94-918400
+	20	Артлайф	Хабаровск	Комсомольск	53	ЛО-01-82-503193
+	21	Фармаком	Уссурийск	Потемкина	4	ЛР-0000-54-9876
+	22	Аптифарм	Владивосток	Уссури	4	ЛО-0000-54-97748

COMMIT

Все изменения, внесенные в таблицу, физически не сохраняются на носителе информации до тех пор, пока не будет выполнена команда COMMIT или не произведен выход из SQL.

Базовая структура оператора COMMIT

COMMIT имя_таблицы

ROLLBACK

Пока не была выполнена команда COMMIT для сохранения изменений в таблице, ее можно вернуть в предыдущее состояние с помощью команды ROLLBACK.

Базовая структура оператора COMMIT

ROLLBACK

SELECT

Базовая структура оператора SELECT:

SELECT имя_атрибута FROM имя_таблицы WHERE условие

Особенности.

- Оператор SELECT является самым важным для пользователя и самым сложным оператором SQL. Он предназначен для выборки данных из таблиц, т.е. реализует одно из основных назначений базы данных – предоставлять информацию пользователю.
- Результатом выполнения оператора SELECT всегда является таблица.
- Любой оператор реляционной алгебры может быть выражен подходящим образом сформулированным оператором SELECT.
- Сложность оператора SELECT определяется тем, что он содержит в себе все возможности реляционной алгебры, а также дополнительные возможности, которых в реляционной алгебре нет.

SELECT

Примеры в MS Access.

Выбор всех записей из таблицы «Аптека»:

```
SELECT * FROM Аптека;
```

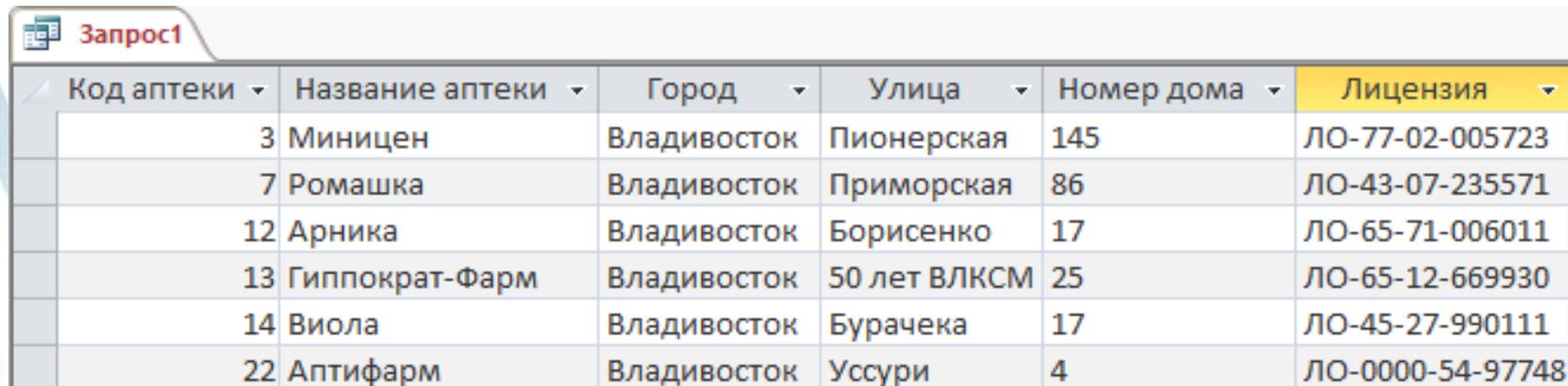
Код аптеки	Название аптеки	Город	Улица	Номер дома	Лицензия
1	Здоровье	Хабаровск	Ленина	15	ЛО-77-02-005584
2	Дальфарма	Хабаровск	Серышева	28	ЛО-57-34-006912
3	Миницен	Владивосток	Пионерская	145	ЛО-77-02-005723
4	Эвалар	Уссурийск	Карла Маркса	6	ЛО-86-05-005358
5	Новая аптека	Москва	Сущевский ва.	23	ЛО-86-05-045678
6	Лекарь	Москва	Новоарбатска	12	ЛО-86-05-014582
7	Ромашка	Владивосток	Приморская	86	ЛО-43-07-235571
8	Радуга	Москва	Башиловская	29	ЛО-04-46-048092
9	Фармаком	Москва	Магаданская	8	ЛО-11-34-095741
10	Флория	Москва	Серпуховский	17	ЛО-65-71-139633
11	Онфарм	Москва	Салтыковская	7	ЛО-92-44-900023
12	Арника	Владивосток	Борисенко	17	ЛО-65-71-006011
13	Гиппократ-Фарм	Владивосток	50 лет ВЛКСМ	25	ЛО-65-12-669930
14	Виола	Владивосток	Бурачека	17	ЛО-45-27-990111
15	Аптека 27	Хабаровск	Дзержинскогс	56	ЛО-78-12-669930
16	Вилаб	Хабаровск	Промышленн	5	ЛО-34-32-563288
17	Эколайф-ДВ	Хабаровск	Суворова	51	ЛО-10-00-009573
18	Семейная аптека	Хабаровск	Кочнева	14	ЛО-00-17-772309
19	Фарком	Хабаровск	Космическая	3а	ЛО-51-94-918400
20	Артлайф	Хабаровск	Комсомольск	53	ЛО-01-82-503193
21	Фармаком	Уссурийск	Потемкина	4	ЛР-0000-54-9876
22	Аптифарм	Владивосток	Уссури	4	ЛО-0000-54-97748

SELECT

Примеры в MS Access.

Выбор записей из таблицы «Аптека» для города Владивостока:

```
SELECT * FROM Аптека WHERE Город = "Владивосток";
```



The screenshot shows a query window titled "Запрос1" (Query1) displaying a table of pharmacy records. The table has six columns: "Код аптеки" (Pharmacy Code), "Название аптеки" (Pharmacy Name), "Город" (City), "Улица" (Street), "Номер дома" (House Number), and "Лицензия" (License). The data is filtered for the city of Vladivostok. The license numbers are underlined in the original image.

Код аптеки	Название аптеки	Город	Улица	Номер дома	Лицензия
3	Миницен	Владивосток	Пионерская	145	<u>ЛО-77-02-005723</u>
7	Ромашка	Владивосток	Приморская	86	<u>ЛО-43-07-235571</u>
12	Арника	Владивосток	Борисенко	17	<u>ЛО-65-71-006011</u>
13	Гиппократ-Фарм	Владивосток	50 лет ВЛКСМ	25	<u>ЛО-65-12-669930</u>
14	Виола	Владивосток	Бурачека	17	<u>ЛО-45-27-990111</u>
22	Аптифарм	Владивосток	Уссури	4	<u>ЛО-0000-54-97748</u>

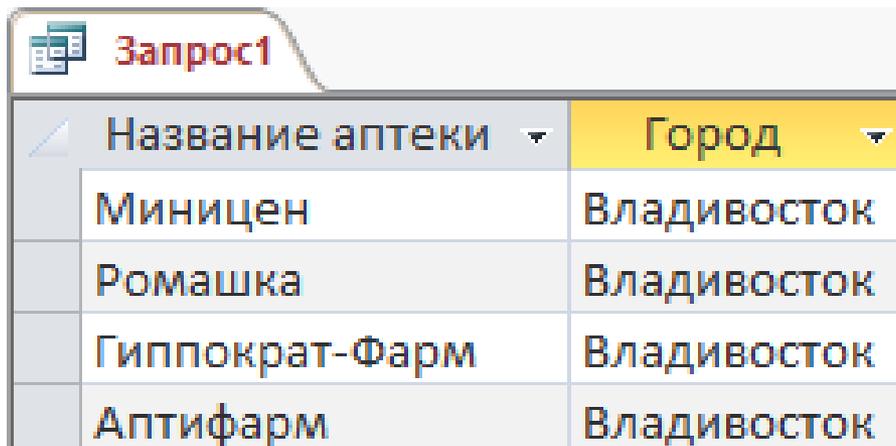
SELECT

В качестве условия в разделе WHERE можно использовать сложные логические выражения, использующие поля таблиц, константы, сравнения (>, <, = и т.д.), скобки, союзы AND и OR, отрицание NOT.

Примеры в MS Access.

Выбор записей из таблицы «Аптека» в соответствии со сложным условием:

```
SELECT [Название аптеки], Город FROM Аптека WHERE Город = "Владивосток" and NOT ([Номер дома]>10 and [Номер дома]<20);
```



The screenshot shows a query window titled 'Запрос1' (Query1) displaying the results of a SELECT query. The table has two columns: 'Название аптеки' (Pharmacy Name) and 'Город' (City). The 'Город' column is highlighted in yellow. The results show four rows, all with 'Владивосток' as the city.

Название аптеки	Город
Миницен	Владивосток
Ромашка	Владивосток
Гиппократ-Фарм	Владивосток
Аптифарм	Владивосток

SELECT

В объявлении WHERE могут использоваться следующие предикаты:

BETWEEN – задание диапазона значений;

IS NULL – проверка на пустое значение атрибута;

LIKE – проверка подобия символьных строк;

IN – проверка присутствия значения атрибута в перечисленном наборе;

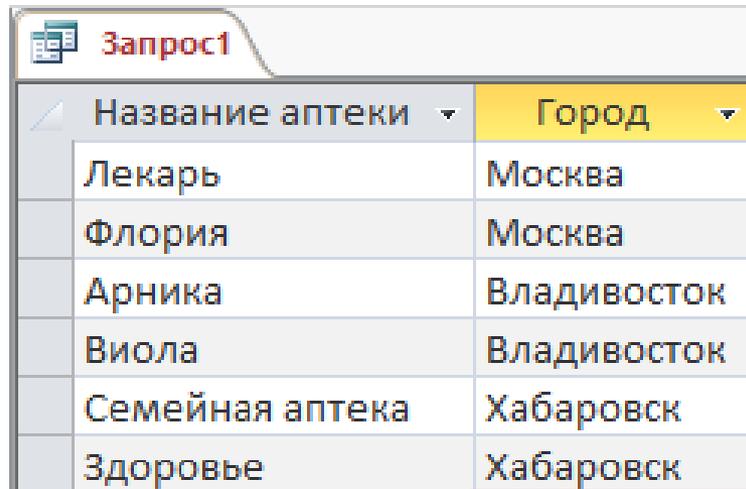
EXISTS – проверка того, имеет ли атрибут какое-либо значение.

SELECT

Примеры в MS Access.

Выбор полей из таблицы «Аптека» для которых «Номер дома» находится в диапазоне от 10 до 20:

```
SELECT [Название аптеки], Город  
FROM Аптека  
WHERE [Номер дома] BETWEEN 10 AND 20;
```



Запрос1

Название аптеки	Город
Лекарь	Москва
Флория	Москва
Арника	Владивосток
Виола	Владивосток
Семейная аптека	Хабаровск
Здоровье	Хабаровск

Аналоги операций реляционной алгебры

UNION – объединение:

запрос_1 UNION запрос_2

CROSS – декартово произведение:

запрос_1 CROSS запрос_2

MINUS – разность:

запрос_1 MINUS запрос_2

Data Control Language, DCL

- GRANT – предоставление пользователю (группе) разрешения на определенные операции с объектом;
- REVOKE – отзыв ранее выданных разрешений;
- DENY - запрет, имеющий приоритет над разрешением

Transaction Control Language, TCL

- **COMMIT** – применение транзакции;
- **ROLLBACK** – отмена всех изменений, сделанный в контексте текущей транзакции;
- **SAVEPOINT** – деление транзакции на более мелкие участки

Список литературы

1. Современный SQL. URL: <https://modern-sql.com/>
2. Коннолли, Томас; Бегг, Каролин. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. 3-е издание. : Пер. с англ. — М. : Издательский дом "Вильямс", 2017. — 1440 с. : ил.
3. Дейт, К. Д. Введение в системы баз данных : Пер. с англ. / К. Д. Дейт. - 8-е изд. - Москва : Диалектика, 2019. - 1328 с. : ил.
4. Стандарты ISO. URL: <https://standards.iso.org/iso/>