Введение в базы данных

Основные определения, SQL и пример на SQLite

Лазар В. И., Козлова Е. Р.

11 января 2025 г.

План занятия

1. Основные определения
2. SQL и SQLite
3. Конструкция SELECT
4. Задания

Что такое база данных?

- **База данных (БД)** — это организованная совокупность данных, структурированная таким образом, чтобы упростить хранение, обработку и поиск информации.
- **Таблица** — одна из основных структур хранения данных. Состоит из строк (записей) и столбцов (полей).
- **Первичный ключ (Primary Key)** — уникальный идентификатор для каждой строки в таблице.
- **СУБД (DBMS)** — система управления базами данных (например, MySQL, PostgreSQL, SQLite).

Основные свойства баз данных

- **Целостность данных**
- **Безопасность и надёжность**
- **Высокая производительность**
- **Масштабируемость**

Что такое SQL?

- **SQL (Structured Query Language)** — язык структурированных запросов, позволяющий взаимодействовать с базой данных.
- Используется для:
  - создания и модификации схемы БД (`CREATE`, `ALTER`, `DROP`);
  - управления данными (`INSERT`, `UPDATE`, `DELETE`);
  - выборки данных (`SELECT`).
- **SQLite** — лёгкая встраиваемая СУБД, которая хранит всю базу в одном файле.

Типы данных в SQL

- `CHAR(size)` - строка фиксированного размера size
- `VARCHAR(size)` - строка переменной длины длины не более size
- `BOOL/BOOLEAN`
- `SMALLINT`, `MEDIUMINT`, `INT/INTEGER` - целые числа (16, 32 и 64 бита сответственно)
- `FLOAT`, `DOUBLE` - числа с плавающей точкой размером 32 и 64 бита соответственно
- `DATE`, `TIME`, `DATETIME`, `TIMESTAMP`

Также в любой* колонке могут встречаться **NULL**-значения.

Общая форма SELECT

Основной синтаксис запроса:

Синтаксис

```
SELECT <столбцы или выражения>
FROM <название_таблицы>
[WHERE <условие>]
[GROUP BY <столбцы>]
[HAVING <условие_для_групп>]
[ORDER BY <столбцы> [ASC|DESC]]
[LIMIT <количество_строк>];
```

- **SELECT** указывает, какие столбцы (или вычисления) нужно вывести.
- **FROM** указывает, из какой таблицы получаем данные.
- **WHERE** — условие фильтрации строк.
- **GROUP BY** — группировка по одному или нескольким столбцам.
- **HAVING** — условие для отфильтрованных групп.
- **ORDER BY** — сортировка (по возрастанию или убыванию).
- **LIMIT** — ограничение количества возвращаемых строк.

Пример 1: Простой запрос

Структура таблицы `employees`:

- `id` (PRIMARY KEY)
- `name`
- `position`
- `salary`

Запрос:

```
SELECT name, position
FROM employees;
```

Описание: Этот запрос вернёт столбцы `name` и `position` из таблицы `employees`.

Пример 2: Фильтрация (WHERE)

```
SELECT name, salary
FROM employees
WHERE salary > 50000;
```

- Возвращает только тех сотрудников, у которых `salary` превышает 50000.

Математические операции

- Стандартные математические операции: `+`, `-`, `*`, `/`, `%`
- Битовые операции: `&`, `^`, `|`, $\sim$
- Операции сравнения: `>`, `<`, `<=`, `>=`, `=`, `<>`, `BETWEEN`
- Логические операции: `AND`, `OR`, `NOT`, `ALL`, `ANY/SOME`, `IN`, `LIKE`, `IS NULL`, `IS NOT NULL`
- Приведение типов: `cast(column_name as type)`

Пример 3: Использование математических операций

```
SELECT DISTINCT department
FROM employees
WHERE (name IN ('Саша', 'Маша', 'Петя')
       AND
       department = 'IT')
      OR salary / work_hours >= 5000;
```

DISTINCT - уникальные значения в колонке

Пример 4: Сортировка (ORDER BY) и LIMIT

```
SELECT name, salary
FROM employees
ORDER BY salary DESC
LIMIT 3;
```

- Сортирует сотрудников по убыванию `salary`.
- Показывает только первые 3 строки (самые высокие зарплаты).

`ASC` - сортировка по возрастанию, `DESC` - сортировка по убыванию

Aliasing (псевдонимы)

- **Aliasing** позволяет давать временные имена столбцам или таблицам.
- Удобно при использовании вычисляемых полей или при работе с длинными названиями таблиц.

Пример: Использование псевдонима столбца

```
SELECT name AS employee_name,
       salary * 1.2 AS new_salary
FROM employees;
```

- `AS` задаёт псевдоним для отображаемого столбца.
- В результате в итоговой выборке столбец будет называться `employee_name`, а второй — `new_salary`.

Пример: Псевдоним таблицы

```
SELECT e.name, e.position
FROM employees AS e;
```

Аггрегирующие функции

- `MIN` - минимальное значение в колонке
- `MAX` - максимальное значение в колонке
- `AVG` - среднее значение в колонке
- `SUM` - сумма значений в колонке
- `COUNT` - количество значений в колонке

Важно: все аггрегирующие функции игнорируют **NULL**-значения кроме `COUNT`.

Пример 5: Группировка (GROUP BY) и HAVING

```
SELECT position, AVG(salary) AS avg_salary
FROM employees
GROUP BY position
HAVING AVG(salary) > 40000;
```

- Группирует сотрудников по должности (`position`).
- Считает среднюю зарплату по каждой должности.
- Оставляет только те должности, где средняя зарплата больше 40000.

Пример 6: Простые вложенные запросы

```
SELECT *
FROM employees
WHERE salary = (SELECT MAX(salary) FROM employees);
```

Средства работы с SQLite

- SQLiteStudio
- PyCharm Professional Edition
- Visual Studio Code (SQL plugin)
- Vim/Neovim (dadbod plugin)

Задания для factbook.db

- Подключитесь к БД
- Сколько в ней таблиц?
- Для каждой таблицы в БД напишите запрос, получающий первые 10 строк любых двух колонок из этой таблицы
- Напишите запрос, получающий топ-20 столиц по населению
- Напишите запрос, получающий последние 10 строк любых двух колонок из таблицы facts
- Напишите запрос, получающий новую колонку `Population_%`, содержащую информацию о доле населения в городе относительно населения всех городов представленных в таблице
- Напишите запрос, получающий все строки, содержащие **NULL** хотя бы в одной колонке
- Выведите топ-10 стран по населению (facts_id - идентификатор страны)

Задания для jobs.db

- Подключитесь к БД
- Напишите запрос, получающий все категории занятости (Major_category), где сумма по всем сферам долей занятости женщин более 5
- Напишите запрос, получающий процентную долю мужчин в каждой сфере занятости (Major)
- Напишите запрос, получающий топ-10 сфер занятости по доли занятости женщин в этой сфере при условии что женщин не более 0.8 и не менее 0.5 от общего числа
- Напиште запрос, получающий все сферы занятости, где доля женщин более половины от общего числа либо менее, но при условии, что сфера занятости попадает в категорию `Engineering`
- Напишите запрос, получающий категории занятости, для которых показатель женской занятости (из задания 1) находится в промежутке от 3 до 5