8. Базы данных

Базы данных (БД) — это организованные хранилища данных, которые позволяют эффективно записывать, извлекать и управлять информацией. Они делятся на **реляционные (SQL)** и **нереляционные (NoSQL)**.

---

1. SQL (Реляционные БД)

Примеры: **PostgreSQL, MySQL, Oracle, SQL Server**.

Основные SQL-запросы

- `SELECT` – выборка данных.

```
SELECT * FROM users WHERE age > 18;
```

- `JOIN` – объединение таблиц.
  - `INNER JOIN` – только совпадающие строки.
  - `LEFT JOIN` – все строки из левой таблицы + совпадения.

```
SELECT u.name, o.order_id
FROM users u
LEFT JOIN orders o ON u.id = o.user_id;
```

- `GROUP BY` – группировка с агрегатными функциями (`COUNT`, `SUM`, `AVG`).

```
SELECT department, AVG(salary)
FROM employees
GROUP BY department;
```

Индексы

- Ускоряют поиск по столбцам (как оглавление в книге).
- Замедляют вставку/обновление (индекс нужно перестраивать).
- Пример:

```
CREATE INDEX idx_user_email ON users(email);
```

Транзакции (ACID)

- **Atomicity (Атомарность)** – транзакция либо выполняется целиком, либо откатывается.
- **Consistency (Согласованность)** – БД всегда в валидном состоянии.
- **Isolation (Изолированность)** – параллельные транзакции не мешают друг другу.
- **Durability (Долговечность)** – после фиксации изменения сохраняются даже при сбое.

Пример в SQL:

```
BEGIN TRANSACTION;
  UPDATE accounts SET balance = balance - 100 WHERE user_id = 1;
```

```
  UPDATE accounts SET balance = balance + 100 WHERE user_id = 2;
COMMIT; -- или ROLLBACK в случае ошибки
```

---

2. NoSQL (Нереляционные БД)

MongoDB (Документная БД)

- Хранит данные в **JSON-подобных документах** (BSON).
- Гибкая схема (нет строгих таблиц).
- Пример:

```
db.users.insertOne({ name: "Alice", age: 25, hobbies: ["coding", "music"] });
```

Redis (Ключ-значение + кэш)

- Хранит данные в памяти (очень быстро).
- Подходит для кэширования, сессий, очередей.
- Пример:

```
SET user:1 "Alice"
GET user:1  # Вернет "Alice"
```

---

3. JDBC (Java Database Connectivity)

API для работы с БД в Java.

Основные классы

- `Connection` – соединение с БД.
- `Statement` – выполнение SQL-запросов (уязвим к SQL-инъекциям).
- `PreparedStatement` – безопасный запрос с параметрами.

Пример:

```
try (Connection conn = DriverManager.getConnection(url, user, password)) {
    String sql = "SELECT * FROM users WHERE age > ?";
    PreparedStatement stmt = conn.prepareStatement(sql);
    stmt.setInt(1, 18);
    ResultSet rs = stmt.executeQuery();
    while (rs.next()) {
        System.out.println(rs.getString("name"));
    }
}
```

---

4. Миграции БД

Инструменты для управления изменениями схемы БД.

Flyway

- Простота: SQL-скрипты в папке `src/main/resources/db/migration`.
- Порядок версий: `V1__Create_table.sql`, `V2__Add_index.sql`.
- Интеграция с Spring Boot.

Liquibase

- Гибкость: поддерживает XML, YAML, JSON.
- Возможность отката изменений.
- Пример (`changelog.xml`):

```
<changeSet id="1" author="me">
    <createTable tableName="users">
        <column name="id" type="INT" autoIncrement="true"/>
        <column name="name" type="VARCHAR(255)"/>
    </createTable>
</changeSet>
```

Когда что использовать?

Критерий	SQL (PostgreSQL)	NoSQL (MongoDB/Redis)
Структура данных	Строгая схема (таблицы)	Гибкая схема (документы)
Масштабируемость	Вертикальное	Горизонтальное (шардинг)
Скорость записи	Средняя	Очень высокая (Redis)
Пример использования	Банковские системы	Кэш, логи, чаты

JDBC – низкоуровневое подключение к БД, **Hibernate/JPA** – более высокоуровневая ORM.
Flyway/Liquibase – выбор зависит от сложности проекта (Flyway проще, Liquibase мощнее).