# БД 2. Курс. Конспект. #

## 1. Что такое БД ##

База данных - это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе.

БД обычно управляется системой управления базами данных (СУБД).

Также БД может представлять собой контейнер, в котором хранятся таблицы и другие структуры SQL.

---

## 2. Что такое СУБД ##

Система управления базами данных (СУБД, DBMS – Data Base Management System) - совокупность языковых и программных средств, предназначенных для создания, ведения и совместного использования базы данных многими пользователями.

---

\*\*\*Основные функции СУБД:\*\*\*

+ Проектирование БД – определение состава и структуры данных, подлежащих хранению в БД, создание схемы данных;

+ Создание БД – первоначальная загрузка данных, сопровождаемая максимально возможной проверкой их правильности;

+ Извлечение данных из БД – отбор и выдача данных по заданным критериям пользователя;

+ Модификация структуры БД - включение/удаление таблиц БД.

+ Актуализация данных (добавление, обновление, удаление) с обеспечением целостности БД.

+ Экспорт данных в другие БД и импорт данных из них.

+ Защита и резервное копирование БД.

---

## 3. Что такое SQL ##

SQL (сокращение от англ. Structured Query Language) — это язык запросов, который применяют, чтобы работать с базами данных, структурированных особым образом.

Главные задачи SQL — составлять запросы так, чтобы находить среди большого объёма информации ту, что нужна для конкретных целей, сортировать её, структурировать и представлять в наиболее простом и понятном виде.

---

## 4. Типы БД ##

### I. Простейшие типы БД ###

#### 1. Простые структуры данных (текстовые файлы) ####

Информация об объектах собирается в простых по структуре файлах различных форматов – txt, csv и др. Для разделения полей применяются пробелы, табуляция, запятые, точка с запятой и двоеточие.

#### 2. Иерархические БД ####

В отличие от текстовых файлов здесь между хранимыми объектами устанавливаются связи. Объекты делятся на родителей (основные классы или категории объектов) и потомков (экземпляры этих классов или категорий). При этом у каждого потомка может быть не более одного родителя.

#### 3. Сетевые БД ####

Развивает иерархический подход за счет моделирования сложных отношений между объектами. Здесь потомки могут иметь более одного родителя, однако ограничения иерархического подхода сохраняются.

### II. Реляционные БД ###

Данные формируются в таблицы из строк и столбцов. В строках приводятся сведения об объектах (значения свойств), а в столбцах — сами свойства объектов (поля).

#### 4. SQL БД ####

### III. NoSQL БД ###

#### 5. Базы данных «ключ-значение» ####

Данные хранятся в виде словаря, где указателем выступает ключ.

#### 6. Документная БД ####

Данные здесь хранятся в структурированных форматах – XML, JSON, BSON. Тем не менее, сохраняется адресный доступ к данным по ключу. При этом содержимое документа может иметь различный набор свойств.

#### 7. Графовая БД ####

Предназначены для моделирования сложных отношений с помощью теории графов, где связями выступают ребра графа, а сами объекты – это узлы или вершины.

#### 8. Колоночная БД ####

Записи в таких базах хранятся не по строкам, а по столбцам (колонкам). Вместо таблиц здесь используются колоночные семейства. Они содержат ключи, указывающие на формат строки записи информации об объекте. Каждая строка имеет свой набор свойств, что позволяет хранить в рамках одного семейства разно структурированные данные.

#### 9. База данных временных рядов ####

Данные группируются по временным меткам. Базы временных рядом чаще ориентированы на запись, чем на построение сложных аналитических запросов.

### IV. Комбинированные типы ###

#### 10. NewSQL БД ####

Данный тип решений для хранения информации стремится обеспечить компромисс между масштабируемостью и согласованностью при сохранении реляционного подхода.

#### 11. Многомодельные БД ####

Такие БД сочетают в себе несколько подходов к организации данных одновременно. Это обеспечивает функциональное разнообразие при разработке систем с их использованием.

---

## 4. Нужна БД ##

+ Для структурированности

+ Для быстрого доступа

+ Для безопасности

---

## 5. Категории языка SQL ##

### 5.1. Data Definition Language (DDL) ###

Группа операторов определения данных (для создания, изменения, удаления объектов БД, операторы CREATE, ALTER, DROP)

### 5.2. Data Manipulation Language (DML) ###

Группа операторов для манипуляции данными (добавление, изменение, удаление, выгрузка данных из БД, операторы SELECT, INSERT, UPDATE, DELETE)

### 5.3. Data Query Language (DQL) ###

Операторы для запроса данных , запрос состоит из предложений SELECT, FROM и WHERE

### 5.4. Data Control Language (DCL) ###

Группа операторов определения доступа к данным (предоставление, отзыв, запрет доступа, операторы GRANT, REVOKE, DENY)

---

## 6. Реляционная БД ##

Это набор данных с предопределенными связями между ними. Эти данные организованны в виде набора таблиц, состоящих из столбцов и строк. В таблицах хранится информация об объектах, представленных в базе данных.

Состоит из таблиц:

Каждый столбец - имеет «тип и имя»

Каждая строка- отдельная запись или элемент данных в таблице.

Поле в таблице (внешний ключ) может содержать ссылки содержащие в других таблицах. Что позволяет их соединять.

Высокоорганизованная структура и гибкость позволяет адаптироваться к различным типам данных

\*\*Примеры:\*\*

MySQL

PostgreSQL

MariaDB

SQLite

### 6.1. Primary KEY (первичный ключ) ###

Оператор SQL PRIMARY KEY (Первичный ключ) — это параметр, который устанавливается для однозначной идентификации той или иной записи в таблице. Значения SQL PRIMARY KEY должны быть всегда уникальны, а так же не содержать значений NULL.

В каждой таблице может быть только один первичный ключ, но в БД их будет столько же, сколько и таблиц в БД.

\*\*Свойства Primary key:\*\*

Уникальность и минимальность.

Первичный ключ должен сохранять свою уникальность со временем

Таблица может иметь только один первичный ключ, а в таблице этот первичный ключ может состоять из одного или нескольких столбцов (полей).

### 6.2. Foreign key (внешний ключ) ###

Foreign key- внешний ключ для соединения двух таблиц.

Это поле (несколько полей) в одной таблице, которое ссылается на первичный ключ primary в другой таблице.

Таблица с foreign дочерняя

Таблица с primary родительская

Внешний ключ нужен для того, чтобы связать две разные SQL-таблицы между собой.

---

## 7. Нереляционная БД ##

Не используется табличная схема строк и столбцов. Хранение данных json, пара ключ - значение или граф - ребра и вершины.

Термин No SQL - применяется к БД не использующим язык запросов SQL.

\*\*Примеры:\*\*

MongoDB, Coachbase, Redis

---

## 8. Сущность ##

Сущность в таблицах - объект который хранится в БД. Каждая таблица представляет одну сущность.

---

## 9. JOIN ##

Оператор для объединения данных из нескольких таблиц с общим ключом.

Горизонтальное объединение таблиц.

---

## 10. UNION ##

Объединяет 2 набора строк из нескольких запросов по вертикали.

Оба запроса должны возвращать одинаковое количество столбцов

---

## 11. DBeaver ##

Это свободный кроссплатформенный менеджер баз данных для Linux, Windows, MacOS. Много возможностей, небольшой вес, портабельность, свободное ПО. Повелитель БД) проникает в любую структуру в базе данных и показывает что внутри, помогает навести порядок, внести изменения и предоставит прочие полезности.

---

## 12. Особенности реляционных БД: ##

+ Модель данных в реляционных БД определена заранее и является строго типизированной

+ Данные хранятся в таблицах, состоящих из столбцов и строк

+ На пересечении каждого столбца и строчки допускается только одно значение

+ Каждый столбец проименован и имеет определенный тип, которому следуют значения со всех строк в данном столбец

+ Столбцы располагаются в определённом порядке, который определяется при создании таблицы

+ В таблице может не быть ни одной строчки, но обязательно должен быть хотя бы один столбец

+ Запросы к базе данных возвращают результат в виде таблиц

---

## 13. Особенности нереляционных БД: ##

+ Высокая гибкость

Работы разбиваются на отдельные этапы и привлекаются несколько специалистов.

Можно создавать документы, заранее не устанавливая структуру. (Для каждого файла своя структура).

Во время работы можно добавлять новые поля.

+ Эффективность

Легко оптимизируется под хранение определенных данных и шаблонов.

+ Масштабируемость

Предусмотрено несколько кластеров для разделения информации и добавления любого кол-ва серверов.

Отлично подходит для часто меняющихся масштабных хранилищ.

---

## 14. Критерии оценки БД: ##

+ Доступ к информации конечного пользователя в нужное время и в нужном месте.

+ Открытость и гибкость запросов

+ Надежность

+ Поддержка и распространенность

+ Интеграция с ПО

+ Спектр возможностей

+ Стоимость

---

## 15. Уровни проектирования БД: ##

+ Концептуальное

+ Логическое

+ Физическое

---

## 16. Типы связей: ##

+ Один ко многим

предполагает, что одному атрибуту первой таблицы соответствует несколько атрибутов второй таблицы;

+ Один к одному (1:1)

предполагает, что одному атрибуту первой таблицы соответствует только один атрибут второй таблицы, и наоборот;

+ Многие ко многим

предполагает, что одному атрибуту первой таблицы соответствует несколько атрибутов второй таблицы, и наоборот.

---

## 17. Нормализация ##

Нормализация – это процесс удаления избыточных данных.

Также нормализацию можно рассматривать и с позиции проектирования базы данных, в таком случае мы можем сформулировать определение нормализации следующим образом.

Нормализация – это метод проектирования базы данных, который позволяет привести базу данных к минимальной избыточности.

Нормализация - это процесс удаления избыточных данных для устранения аномалий, повышения производительности, повышения удобства управления данными

### 17.1. Первая нормальная форма ###

+ Не дублировать строки

+ В каждой ячейке таблицы хранится одно атомарное значение

+ В столбце хранятся данные одного типа

+ Отсутствуют массивы и списки

### 17.2. Вторая нормальная форма ###

+ Таблица должна находится в первой нормальной форме

+ Таблица должна иметь ключ

+ Все неключевые столбцы должны зависеть от полного ключа.

### 17.3. Третья нормальная форма ###

+ Таблица должна быть во второй нормальной форме

+ Все колонки в таблице зависят от первичного ключа и не зависят друг от друга

---

## 18. Что необходимо для вертикального объединения (UNION): ##

+ Одинаковое кол-во колонок

+ Совместимые типы данных (т.е. строка под строкой, число под числом, дата под датой и т.д.)

Есть следующие виды вертикального объедения Union all, union, expect, intersect.

---

## 19. Ограничения SQL ##

— это правила, применяемые к столбцам данных таблицы.

Они используются, чтобы ограничить типы данных, которые могут храниться в таблице.

Это обеспечивает точность и надежность данных в базе данных.

Ограничения могут применяться либо на уровне столбцов, либо на уровне таблицы.

Ограничения на уровне столбца применяются только к одному столбцу, тогда как ограничения уровне таблицы применяются ко всей таблице.

+ NOT NULL — столбец не может иметь значение NULL.

+ DEFAULT — задает значение по умолчанию для столбца, если оно не указано.

+ UNIQUE — все значения в столбце должны быть разными.

+ PRIMARY Key — уникальная идентификация каждой строки/записи в таблице базы данных.

+ FOREIGN Key — уникально идентифицирует строку/запись в любой другой таблице базы данных.

+ CHECK — ограничение CHECK обеспечивает, чтобы все значения в столбце удовлетворяли определенным условиям.

+ INDEX — используется для быстрого создания данных базы данных.

Ограничения могут указываться при создании таблицы с помощью оператора CREATE TABLE или вы можете использовать оператор ALTER TABLE для создания ограничений уже после создания таблицы.

---

## 20. Подзапрос (SubQueries) ##

это SQL-запрос, вложенный в более крупный запрос.

---

## 21. ФУНКЦИИ ЯЗЫКА SQL: ##

+ COUNT() — функция возвращающая количество записей (строк) таблицы.

+ AVG() — функция возвращающая среднее значение столбца. Данная функция применима только для числовых столбцов.

+ MIN() — функция возвращающая минимальное значение столбца.

+ MAX() — функция возвращающая максимальное значение столбца таблицы.

+ SUM() — функция, возвращающая сумму значений столбца таблицы.

+ ROUND() — функция для округления десятичных чисел.

+ UCASE() — функция, возвращающая значения столбца или столбцов в верхнем регистре букв.

\*\*В СУБД MS SQL Server аналогом UCASE() является функция UPPER\*\*

+ LCASE() — функция, возвращающая значения столбца или столбцов в нижнем регистре букв.

\*\*В СУБД MS SQL Server аналогом оператора SQL LCASE() является функция LOWER\*\*

+ LEN() — функция, возвращающая длину значения в поле записи. Функция исключает из подсчета конечные пробелы.

+ MID() — функция, выводящая определенное количество символов текстового поля таблицы.

+ NOW() — функция, возвращающая системное время и дату.