

ИСБД.

1. Введение

1. СУБД

Зачем нужны БД?

Почему нельзя «просто» использовать файлы?

Зачем нужны БД?

Просто хранить данные в файлах неудобно:

- Полное отсутствие гибкости (изменение структуры – изменение кода программы).
- Невозможность нормальной многопользовательской работы с данными.
- Вынужденная избыточность (проще создать еще одну копию данных, чем вносить изменения в десятки программ).

Решение — хранить данные и *метаданные* (данные о данных) вместе.

База данных

База данных — это файлы, снабжённые описанием хранимых в них данных и находящиеся под управлением специальных программных комплексов, называемых "Системы управления базами данных" (СУБД).

Классификация СУБД

По степени распределённости:

- *локальные;*
- *распределённые.*

Классификация СУБД

По способу доступа к БД.

- *Файл-серверные* — данные находятся на файл-сервере, СУБД — на каждом клиентском компьютере. Примеры — M\$ Access, dBase, FoxPro.
- *Клиент-серверные* — СУБД находятся на сервере вместе с данными. Примеры — Oracle, M\$ SQL Server, Caché.
- *Встраиваемые* — СУБД встраивается в приложение, хранит только его данные и не требует отдельной установки. Примеры — SQLite, BerkeleyDB.

Классификация СУБД (продолжение)

По модели данных:

- *Сетевые* — используют сетевую модель данных. Частный случай — графовые СУБД. Примеры — HypergraphDB, OrientDB.
- *Объектно-ориентированные* — используют ОО-модель данных. Пример — InterSystems Caché.
- *Реляционные и объектно-реляционные* — используют реляционную модель данных (возможно, с частичной поддержкой ООП). Примеры — Oracle, PostgreSQL.

2. Реляционные БД

Реляционные БД

Реляционная (relation — отношение, связь) или табличная модель данных была предложена в конце 60-х годов Эдгаром Коддом (IBM).

Ключевые особенности:

- 1) Данные хранятся в таблицах.
- 2) Каждая таблица состоит из строк, имеющих одну и ту же структуру, и имеет уникальное имя.

STUDENT

id	name	surname	gr_id
1	Григорий	Иванов	34
2	Григорий	Иванов	34
3	Иван	Сидоров	37

Ключевые особенности:

- 3) Строки имеют фиксированное число полей (столбцов) и значений (множественные поля и повторяющиеся группы недопустимы).
- 4) В каждой позиции таблицы на пересечении строки и столбца всегда имеется в точности одно значение или ничего.

STUDENT

id	name	surname	gr_id
1	Григорий	Иванов	34
2	Григорий	Иванов	34
3	Иван	Сидоров	37

Реляционные БД (продолжение)

- 5) Строки таблицы обязательно отличаются друг от друга хотя бы единственным значением.
- 6) Столбцам таблицы однозначно присваиваются имена.
- 7) В каждом из столбцов размещаются однородные значения данных (даты, фамилии, целые числа).

STUDENT

id	name	surname	gr_id
1	Григорий	Иванов	34
2	Григорий	Иванов	34
3	Иван	Сидоров	37

- 8) Содержание базы данных представляется в виде явных значений данных (не существует каких-либо специальных "связей" или указателей, соединяющих одну таблицу с другой).
- 9) При выполнении операций с таблицей ее строки и столбцы можно обрабатывать в любом порядке безотносительно к их содержанию.
- 10) Таблицы базы данных часто группируют в схемы.

Терминология

Переменная отношения/
Имя таблицы

Атрибут/
Колонка

STUDENT

id	name	surname	gr_id
1	Григорий	Иванов	34
2	Григорий	Иванов	34
3	Иван	Сидоров	37

Заголовок

Тело

Кортеж/Строка

- **Ключ** — минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр сущности.
- **Внешний ключ** — множество атрибутов сущности для организации ссылок на экземпляры другой сущности (должен соответствовать *первичному* ключу в другой сущности)

Таблицы

STUDENT

id	name	surname	gr_id
1	Григорий	Иванов	34
2	Григорий	Иванов	34
3	Иван	Сидоров	37

GROUP

id	name	class_leader
34	P3100	3
37	P3112	2
354	R4230	NULL

Целостность данных

Целостность (от англ. integrity) — корректность данных в любой момент времени.

Выделяют три группы правил целостности.

- *Целостность по сущностям.*
- *Целостность по ссылкам.*
- *Целостность, определяемая пользователем.*

Целостность данных

- *Целостность по сущностям.* Не допускается, чтобы какой-либо атрибут, участвующий в первичном ключе, принимал неопределенное значение.
- *Целостность по ссылкам.* Значение внешнего ключа должно либо:
 - быть равным значению первичного ключа ассоциируемой сущности;
 - быть полностью неопределенным.

Целостность данных

- *Целостность, определяемая пользователем:*
 - уникальность тех или иных атрибутов;
 - диапазон значений (экзаменационная оценка от 2 до 5);
 - принадлежность набору значений (пол "М" или "Ж").

Сколько ошибок на слайде?

STUDENT

id	name	surname	gr_id
1	Григорий	Иванов	34
2	Иван	Сидоров	34
2	Петр	Петров	37
2	Иван	Сидоров	34

GROUP

id	name	class_leader
'Vasya'	P3100	3
37	P3112	2
34	R4230	4

Ответ

STUDENT

id	name	surname	gr_id
1	Григорий	Иванов	34
2	Иван	Сидоров	34
2	Петр	Петров	37
4	Иван	Сидоров	34

GROUP

id	name	class_leader
'Vasya'	P3100	3
37	P3112	2
354	R4230	4

3. Язык SQL

О языке

- *SQL (структурированный язык запросов)* — декларативный язык программирования.
- Применяется для получения, создания, модификации и управления данными в реляционной базе данных.
- Разработан в 1970-е гг. в компании IBM.

- Язык используется в большинстве реализаций реляционных БД.
- Стандартизирован: SQL-86
SQL-89
SQL-92
SQL-99
SQL-2003
SQL-2006
SQL-2008
...

Особенности языка

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

Различают следующие **типы запросов**:

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

Операторы SQL

Операторы определения данных (*Data Definition Language, DDL*):

- CREATE создает объект БД (саму базу, таблицу, представление, пользователя и т. д.);
- ALTER изменяет объект;
- DROP удаляет объект.

Операторы SQL (продолжение)

Операторы манипуляции данными (*Data Manipulation Language, DML*):

- SELECT считывает данные, удовлетворяющие заданным условиям;
- INSERT добавляет новые данные;
- UPDATE изменяет существующие данные;
- DELETE удаляет данные.

Операторы SQL (продолжение)

Операторы определения доступа к данным (*Data Control Language, DCL*):

- GRANT предоставляет пользователю (роли) разрешения на определенные операции с объектом;
- REVOKE отзывает ранее выданные разрешения;

Оператор SELECT

Возвращает набор данных из БД, удовлетворяющих заданному условию.

Формат запроса:

SELECT список полей FROM список таблиц

WHERE условия...

Оператор SELECT

Основные ключевые слова:

- WHERE — используется для определения, какие строки должны быть выбраны или включены в GROUP BY.
- GROUP BY — используется для объединения строк с общими значениями в элементы меньшего набора строк.
- HAVING — используется для определения, какие строки после GROUP BY должны быть выбраны.
- ORDER BY — используется для определения, какие столбцы используются для сортировки результирующего набора данных.

Агрегатные функции

- COUNT(*)
- COUNT(A)
- SUM(A)
- AVG(A)
- MIN(A)
- MAX(A)

A — некоторый атрибут

Параметр GROUP BY

- Позволяет группировать строки и использовать агрегатные функций (MAX, SUM, AVG, ...).
- Необходимо, чтобы в SELECT были заданы только требуемые в выходном потоке столбцы, перечисленные в GROUP BY и/или агрегированные значения.
- Пример использования:

Запрос возвращает список партнеров с общей суммой их продаж:

```
SELECT Partner, SUM(SaleAmount) FROM Sales  
GROUP BY Partner;
```

Параметр HAVING

- Позволяет указывать условия на результат агрегатных функций (MAX, SUM, AVG, ...).
- Аналогичен WHERE за исключением того, что строки отбираются не по значениям столбцов, а строятся из значений столбцов, указанных в GROUP BY, и значений агрегатных функций, вычисленных для каждой группы, образованной GROUP BY.
- Необходимо, чтобы в SELECT были заданы только требуемые в выходном потоке столбцы, перечисленные в GROUP BY и/или агрегированные значения.
- Если параметр GROUP BY в SELECT не задан, HAVING применяется к «группе» всех строк таблицы, полностью дублируя WHERE.