

# The Art of Technology



# Базы данных



# Определение

Умно: Совокупность данных, организованных в соответствии с концептуальной структурой, описывающей характеристики этих данных и взаимоотношения между ними.

Понятно: Хранилище любых данных, которые вы будете использовать в приложении.

# Системы управления базами данных

СУБД — это общий термин, относящийся ко всем видам абсолютно разных инструментов, от компьютерных программ до встроенных библиотек. Эти приложения управляют или помогают управлять наборами данных. Так как эти данные могут быть разного формата и размера, были созданы разные виды СУБД.



# Реляционные (SQL) БД

База данных, основанная на реляционной модели данных.

Понятие «реляционный» основано на англ. relation («отношение», «зависимость», «связь»).

Целью нормализации реляционной базы данных является устранение недостатков структуры базы данных, приводящих к избыточности, которая, в свою очередь, потенциально приводит к различным аномалиям и нарушениям целостности данных.

Клиент				Товар	
Id_кл	Фамилия	Имя	Отчество	Id_тов	Название
15	Иванов	Иван	Иванович	1	Шкаф
16	Петров	Петр	Петрович	2	Стул
17	Николаев	Николай	Николаевич	3	Стол

Заказ				
Id_зак	Клиент	Товар	Дата	Количество
1	15	1	15.09.2003	1
2	17	1	17.09.2003	2
3	15	2	20.09.2003	12

# Плюсы и минусы SQL

- Требуют однозначно определенной структуры хранения данных
- Любая реляционная (SQL) СУБД позволяет использовать язык SQL
- Позволяют масштабироваться вертикально (путем увеличения ресурсов)
- Высокая надежность, большое количество качественной документации
- Поддержка сложных структур данных

# Не реляционные (NoSQL) БД

NoSQL-СУБД не используют реляционную модель структуризации данных. Они допускают неограниченное формирование записей и хранение данных в виде ключ-значение.

```
{
  "_id" : "fred",
  "items" : [
    {
      "id" : "slingshot",
      "type" : "weapon",
      "damage" : 23,
      "ranged" : true
    },
    {
      "id" : "sword",
      "type" : "weapon",
      "damage" : 50,
      "ranged" : false
    }
  ]
}
```

# Плюсы и минусы NoSQL

- Можно хранить что угодно и как угодно
- Каждая NoSQL база данных реализует свой способ работы с данными
- Легко масштабируются как вертикально (за счет увеличения ресурсов), так и горизонтально (объединение в кластер)
- Не очень надежны, слабо изучены, мало документации
- Реализация сложной схемы хранения данных может быть затруднительна



# Типы данных

Тип данных поля можно рассматривать как набор характеристик, которые применяются ко всем значениям в этом поле.

- **INTEGER** (ЦЕЛОЕ ЧИСЛО)
- **DECIMAL** (ДЕСЯТИЧНОЕ ЧИСЛО)
- **CHAR** (или СИМВОЛ) .
- **VARCHAR** (ПЕРЕМЕННОЕ ЧИСЛО СИМВОЛОВ).

Важно правильно выбрать типы данных для организации базы данных с точки зрения оптимальной производительности и объема хранимых данных

# Другие специальные типы

- **DATA** (ДАТА) и **TIME** (ВРЕМЯ) — фактически почти стандартные типы (хотя точный формат их меняется).
- **MONEY** (ДЕНЬГИ)
- **BINARY** (ДВОИЧНЫЕ).

# Язык SQL

SQL (англ. structured query language — «язык структурированных запросов») — декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

# Операторы SQL

- **CREATE** создает объект БД (саму базу, таблицу, представление, пользователя и т. д.),
- **ALTER** изменяет объект,
- **DROP** удаляет объект;
  
- **SELECT** выбирает данные, удовлетворяющие заданным условиям,
- **INSERT** добавляет новые данные,
- **UPDATE** изменяет существующие данные,
- **DELETE** удаляет данные;
  
- **GRANT** предоставляет пользователю (группе) разрешения на определенные операции с объектом,
- **REVOKE** отзывает ранее выданные разрешения,
- **DENY** задает запрет, имеющий приоритет над разрешением;
  
- **COMMIT** применяет транзакцию,
- **ROLLBACK** откатывает все изменения, сделанные в контексте текущей транзакции;

# MySQL

MySQL— свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle. Продукт распространяется как под GNU General Public License, так и под собственной коммерческой лицензией.

MySQL является решением для малых и средних приложений. Входит в состав серверов WAMP, AppServ, LAMP и в портативные сборки серверов Денвер, XAMPP, VertrigoServ.



# Работа с MySQL

```
MySQL 8.0 Command Line Client - Unicode
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.0.12 MySQL Community Server - GPL

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE DATABASE IF NOT EXISTS testdatabase;
Query OK, 1 row affected (0.10 sec)

mysql> USE testdatabase;
Database changed
mysql> CREATE TABLE IF NOT EXISTS famous_people;
ERROR 1113 (42000): A table must have at least 1 column
mysql> CREATE TABLE IF NOT EXISTS famous_people (
  -> id INT UNSIGNED NOT NULL AUTO_INCREMENT primary key,
  -> first_name VARCHAR(30) NOT NULL,
  -> LAST_NAME VARCHAR(30) NOT NULL,
  -> money INT
  -> );
Query OK, 0 rows affected (0.28 sec)

mysql> select * from famous_people;
Empty set (0.00 sec)
```

# Создание БД и таблицы

```
mysql> CREATE DATABASE IF NOT EXISTS testdatabase;  
Query OK, 1 row affected (0.10 sec)  
  
mysql> USE testdatabase;  
Database changed  
  
mysql> CREATE TABLE IF NOT EXISTS famous_people (  
  id INT UNSIGNED NOT NULL AUTO_INCREMENT primary key,  
  first_name VARCHAR(30) NOT NULL,  
  last_name VARCHAR(30) NOT NULL,  
  money INT  
);  
Query OK, 0 rows affected (0.28 sec)  
  
mysql> SELECT * FROM famous_people;  
Empty set (0.00 sec)
```

1. Создаем базу данных
2. Подключаемся к ней
3. Создаем таблицу со столбцами:
  - Id – беззнаковое число, не может быть пустым, автоматически увеличивает свое значение, является первичным ключом;
  - first\_name – строка любых символов длиной до 30, не пустая;
  - LAST\_NAME – аналогично;
  - money – число, может быть пустым

В конце запрашиваем все данные из таблицы. Она уже существует, но еще пуста.

# КОМАНДА SELECT

В самой простой форме, команда SELECT просто инструктирует базу данных, чтобы извлечь информацию из таблицы.

**SELECT \***

**FROM Salespeople;**

Эта команда просто выводит все данные из таблицы.

Если вы укажете столбцы отдельно, вы можете получить их в том порядке, в котором хотите.

**WHERE** — предложение команды SELECT, которое позволяет вам устанавливать предикаты, условие которых может быть или верным или неверным для любой строки таблицы. Команда извлекает только те строки из таблицы, для которых такое утверждение верно.



# Первичный ключ

Уникальный столбец (или уникальная группа столбцов), используемый чтобы идентифицировать каждую строку и хранить все строки отдельно, называются — *первичными ключами таблицы*.

Принципы выбора первичного ключа:

- **Минимальность** - в качестве первичного ключа, как правило, выбирают тот, который имеет наименьший размер (физического хранения) и/или включает наименьшее количество атрибутов.
- **Уникальность** - в качестве первичного ключа стараются выбирать такой потенциальный ключ, который с наибольшей вероятностью не утратит уникальность.
- **Неизменяемость** - значение первичного ключа никогда не должно меняться. Изменение значения первичного ключа означает, что Вы меняете идентичность сущности, что не имеет смысла.

Если первичный ключ состоит из единственного атрибута, его называют **простым ключом**.

Если первичный ключ состоит из двух и более атрибутов, его называют **составным ключом**.

Так, номер паспорта и серия паспорта не могут быть первичными ключами по отдельности, так как могут оказаться одинаковыми у двух и более людей. Но не бывает двух личных документов одного типа с одинаковыми серией и номером.

Также есть понятие

**Синтетический (суррогатный) первичный ключ** - дополнительное служебное поле, добавленное к уже имеющимся информационным полям таблицы, единственное предназначение которого — служить первичным ключом. Значение этого поля не образуется на основе каких-либо других данных из БД, а генерируется искусственно.

# Наполнение таблицы

```
mysql> INSERT INTO famous_people (id, first_name, last_name, money) VALUES (null, 'John', 'Cena', 35000000);
Query OK, 1 row affected (0.11 sec)
```

```
mysql> SELECT * FROM famous_people;
```

```
+-----+-----+-----+
| id | first_name | last_name | money |
+-----+-----+-----+
| 1 | John | Cena | 35000000 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

```
mysql> INSERT INTO famous_people (id, first_name, last_name, money) VALUES (null, 'Elon', 'Musk', 19600000000);
ERROR 1264 (22003): Out of range value for column 'money' at row 1
```

Два вопроса:

1. Что за null вместо id?
2. Почему Илон Маск не добавился в базу?

Давайте разбираться

# Значение null



# Значение NULL

**NULL** означает отсутствие, неизвестность информации.

**Значение NULL** не является значением в полном смысле слова: по определению оно означает отсутствие **значения** и не принадлежит ни одному типу **данных**.

Поэтому **NULL** не равно ни логическому значению FALSE, ни пустой строке, ни нулю.

## Сравнение с NULLом

Существуют специальные операторы IS NULL и IS NOT NULL



# 0 (False) и NULL наглядно



Photo by R SATO (@raysato)

# Таблица истинности

		Условие 1		
Условие 2	AND	TRUE	FALSE	NULL
	TRUE	TRUE	FALSE	NULL
	FALSE	FALSE	FALSE	FALSE
	NULL	NULL	FALSE	NULL

		Условие 1		
Условие 2	OR	TRUE	FALSE	NULL
	TRUE	TRUE	TRUE	TRUE
	FALSE	TRUE	FALSE	NULL
	NULL	TRUE	NULL	NULL

		Условие		
	NOT	TRUE	FALSE	NULL
	-	FALSE	TRUE	NULL

# Изменение таблицы

```
mysql> ALTER TABLE famous_people MODIFY money BIGINT;
```

```
Query OK, 1 row affected (0.60 sec)
```

```
Records: 1 Duplicates: 0 Warnings: 0
```

```
mysql> INSERT INTO famous_people (id, first_name, last_name, money) VALUES (null, 'Elon', 'Musk', 19600000000);
```

```
Query OK, 1 row affected (0.10 sec)
```

```
mysql> INSERT INTO famous_people (id, first_name, last_name) VALUES (null, 'Stephen', 'Hawking');
```

```
Query OK, 1 row affected (0.10 sec)
```

```
mysql> SELECT * FROM famous_people;
```

```
+---+-----+-----+-----+
| id | first_name | last_name | money |
+---+-----+-----+-----+
| 1 | John | Cena | 35000000 |
| 2 | Elon | Musk | 19600000000 |
| 3 | Stephen | Hawking | NULL |
+---+-----+-----+-----+
```

```
3 rows in set (0.00 sec)
```

INT: Целое число нормального размера. Диапазон со знаком от -2147483648 до 2147483647. Диапазон без знака от 0 до 4294967295.

BIGINT: Большое целое число. Диапазон со знаком от -9223372036854775808 до 9223372036854775807. Диапазон без знака от 0 до 18446744073709551615. [Документация по типам данных](#)





Make good choices today  
so you don't have regrets  
tomorrow.

*PictureQuotes.com*

# Функции

Запросы могут производить обобщенное групповое значение полей точно также как и значение одного поля. Это делает с помощью агрегатных функций. Агрегатные функции производят одиночное значение для всей группы таблицы. Имеется список этих функций:

- **COUNT** производит номера строк или не-NULL значения полей которые выбрал запрос.
- **SUM** производит арифметическую сумму всех выбранных значений данного поля.
- **AVG** производит усреднение всех выбранных значений данного поля.
- **MAX** производит наибольшее из всех выбранных значений данного поля.
- **MIN** производит наименьшее из всех выбранных значений данного поля.

```
mysql> SELECT SUM(money) FROM famous_people;
+-----+ |
SUM(money)
| +-----+ |
1963500000 |
+-----+
1 row in set (0.00 sec)
```

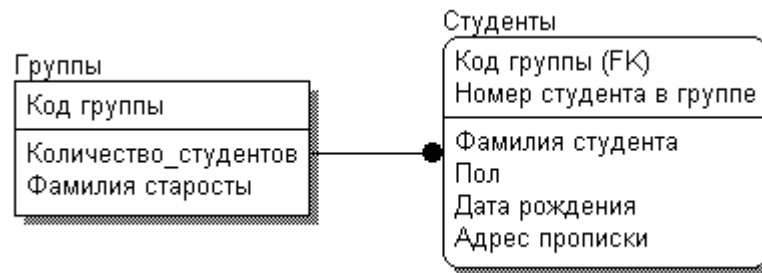
[Документация по функциям](#)

# СВЯЗИ



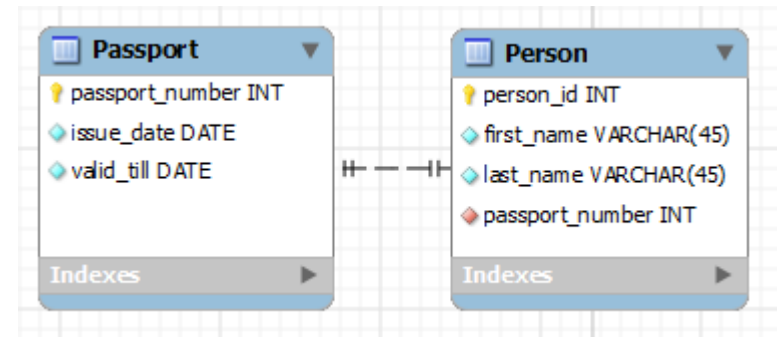
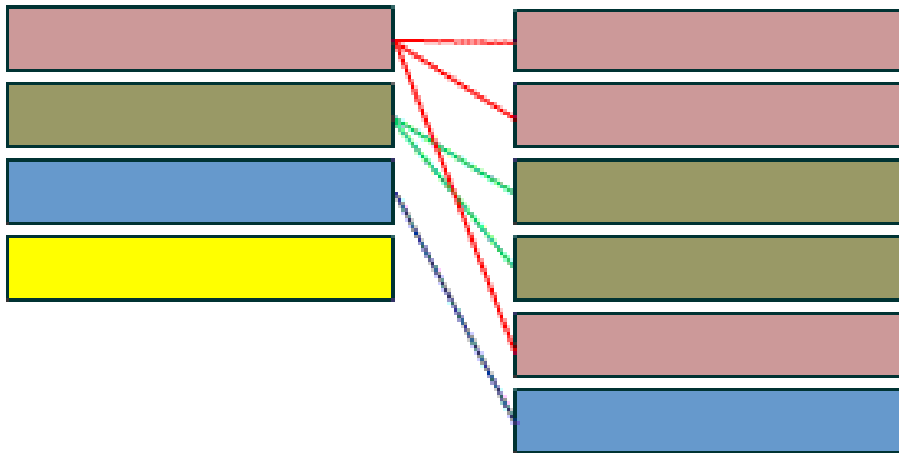
# Внешний и родительский ключ

Когда одно поле в таблице ссылается на другое, оно называется — внешним ключом (Foreign Key); а поле на которое оно ссылается, называется — родительским ключом (только первичный ключ). Каждое значение (каждая строка) внешнего ключа должно недвусмысленно ссылаться к одному и только этому значению (строке) родительского ключа.

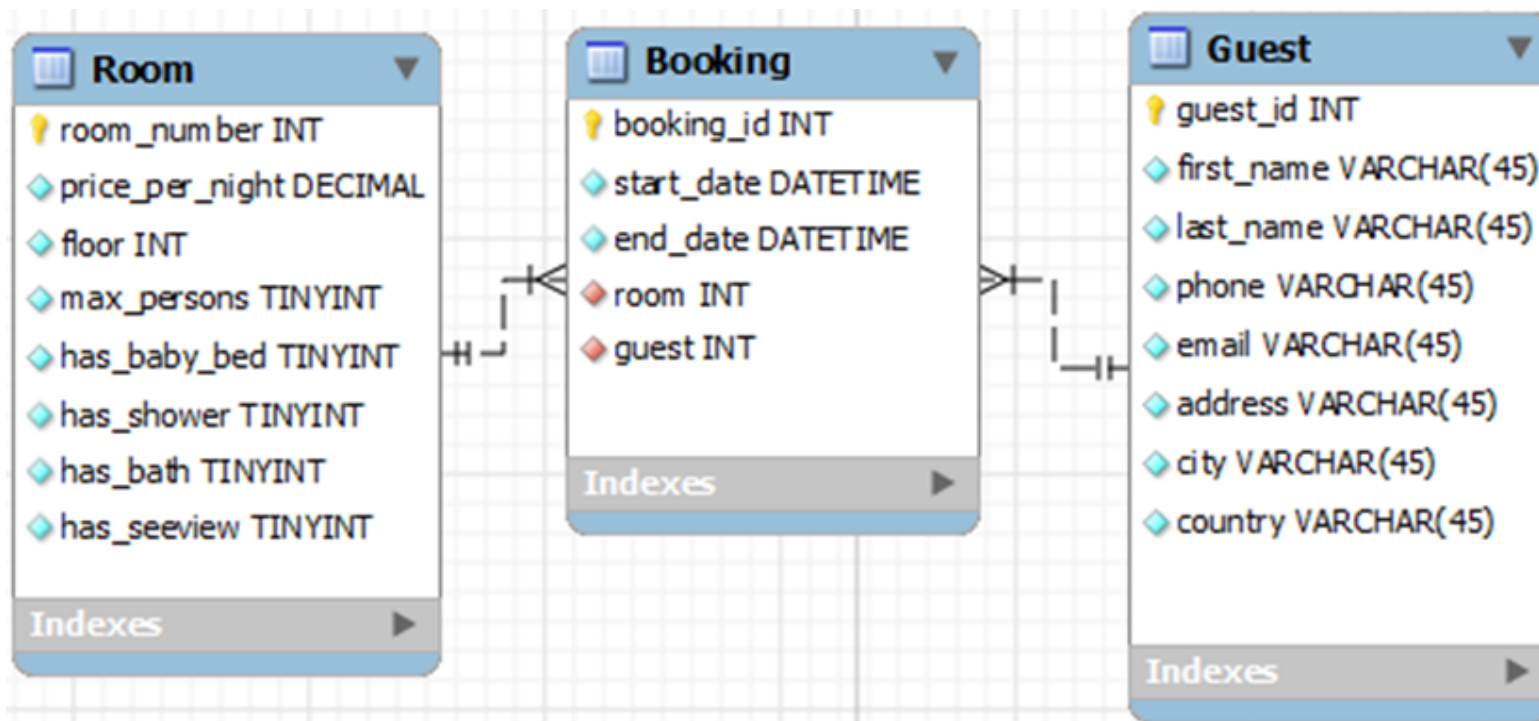


# Типы связей

Связь **«один ко многим»** самая распространенная. В этом типе связей у строки таблицы А может быть несколько совпадающих строк таблицы Б, но каждой строке таблицы Б может соответствовать только одна строка из А.



Связь «**многие ко многим**»: строке таблицы А может сопоставляться несколько строк таблицы Б, и наоборот.



# Связываем таблицы

```
mysql> CREATE TABLE IF NOT EXISTS business (  
-> id INT UNSIGNED NOT NULL AUTO_INCREMENT primary key,  
-> business_type VARCHAR(30)  
-> );  
Query OK, 0 rows affected (0.26 sec)
```

```
mysql> INSERT INTO business (id, business_type) VALUES (null, 'Science');  
Query OK, 1 row affected (0.12 sec)
```

```
mysql> INSERT INTO business (id, business_type) VALUES (null, 'Wrestling');  
Query OK, 1 row affected (0.09 sec)
```

```
mysql> ALTER TABLE famous_people ADD business_id INTEGER NOT NULL;  
Query OK, 0 rows affected (0.26 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

```
mysql> UPDATE famous_people SET business_id=2 WHERE first_name='John';  
Query OK, 1 row affected (0.07 sec)  
Rows matched: 1 Changed: 1 Warnings: 0
```

```
mysql> UPDATE famous_people SET business_id=1 WHERE id IN (2, 3);  
Query OK, 2 rows affected (0.04 sec)  
Rows matched: 2 Changed: 2 Warnings: 0
```

```
mysql> SELECT * FROM famous_people;  
+----+-----+-----+-----+-----+  
| id | first_name | last_name | money | business_id |  
+----+-----+-----+-----+-----+  
| 1 | John | Cena | 35000000 | 2 |  
| 2 | Elon | Musk | 1960000000 | 1 |  
| 3 | Stephen | Hawking | NULL | 1 |  
+----+-----+-----+-----+-----+  
3 rows in set (0.00 sec)
```

Создаем новую таблицу и  
наполняем ее данными

Проставляем необходимые  
ID

# Используем связи

```
mysql> SELECT person.first_name, person.last_name, job.business_type
-> FROM famous_people person
-> INNER JOIN business job ON person.business_id=job.id;
+-----+-----+-----+
| first_name | last_name | business_type |
+-----+-----+-----+
| John | Cena | Wrestling |
| Elon | Musk | Science |
| Stephen | Hawking | Science |
+-----+-----+-----+
3 rows in set (0.00 sec)
```

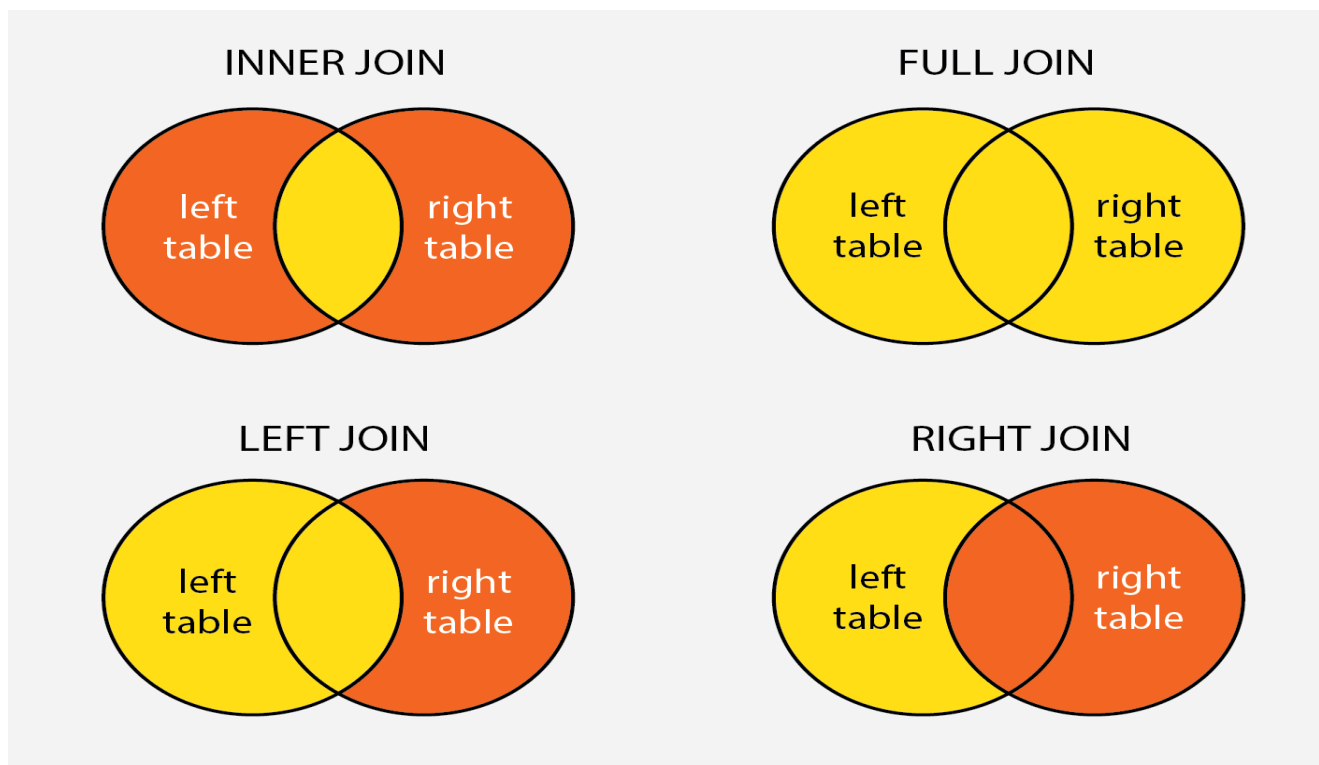
Одна из наиболее важных особенностей запросов SQL — это их способность определять связи между многочисленными таблицами и выводить информацию из них в терминах этих связей, всю внутри одной команды.

С помощью связей за один запрос можно получить информацию сразу из нескольких таблиц. Правда, это будут большие запросы.



# Соединения (JOIN)

JOIN – оператор SQL, который является реализацией операции соединения реляционной алгебры. JOIN предназначен для соединения двух или более таблиц базы данных по совпадающему условию.



# Большие запросы

- Привет.
- Привет.
- Как там ваши дела с Кристиной? Еще не поженились?
- Нет, мы расстались.
- А что случилось?
- Мне надоело, у нее были слишком большие запросы.
- Например какие?
- Ну например `update instance inner join (select group.id as group_id, (select message.id from message inner join thread on thread.id = message.thread_id where location_id = @location_id and language_id = @language_id and concat(group_key, '.') like concat(group.`key`, '.%') order by message.created desc limit 1) as last_message_id, (select count(*) from thread where location_id = @location_id and language_id = @language_id and concat(group_key, '.') like concat(group.`key`, '.%')) as thread_count, (select if(sum(thread.message_count) is null, 0, sum(thread.message_count)) from thread where location_id = @location_id and language_id = @language_id and concat(group_key, '.') like concat(group.`key`, '.%')) as message_count from group where @group_key like concat(`key`, '.%')) as statistics on statistics.group_id = instance.group_id set instance.message_id = statistics.last_message_id, instance.thread_count = statistics.thread_count, instance.message_count = statistics.message_count where instance.location_id = @location_id and instance.language_id = @language_id;`

# Удаляем таблицу и базу

```
mysql> DROP TABLE famous_people;
Query OK, 0 rows affected (0.30 sec)
```

```
mysql> SELECT * FROM famous_people;
ERROR 1146 (42S02): Table 'testdatabase.famous_people' doesn't exist
```

```
mysql> DROP DATABASE testdatabase;
Query OK, 1 row affected (0.28 sec)
```

```
mysql> SELECT * FROM famous_people;
ERROR 1046 (3D000): No database selected
```


SQL инъекция — это один из самых доступных способов взлома сайта. Суть таких инъекций – внедрение в данные (передаваемые через GET, POST запросы или значения Cookie) произвольного SQL кода. Если сайт уязвим и выполняет такие инъекции, то по сути есть возможность творить с БД что угодно.



# Вопросы ?



Thanks for your  
attention

 [artezio\\_software](#)

 [info@artezio.com](mailto:info@artezio.com)

**[www.artezio.com](http://www.artezio.com)**