нетология
центр онлайн-образования

УПРАВЛЕНИЕ ТАБЛИЦАМИ И БАЗАМИ ДАННЫХ

ДАЦКОВ АЛЕКСЕЙ / РУКОВОДИТЕЛЬ РАЗРАБОТКИ ПРОЕКТА «БЕРИТО»

ДАЦКОВ АЛЕКСЕЙ

alex.smap@gmail.com

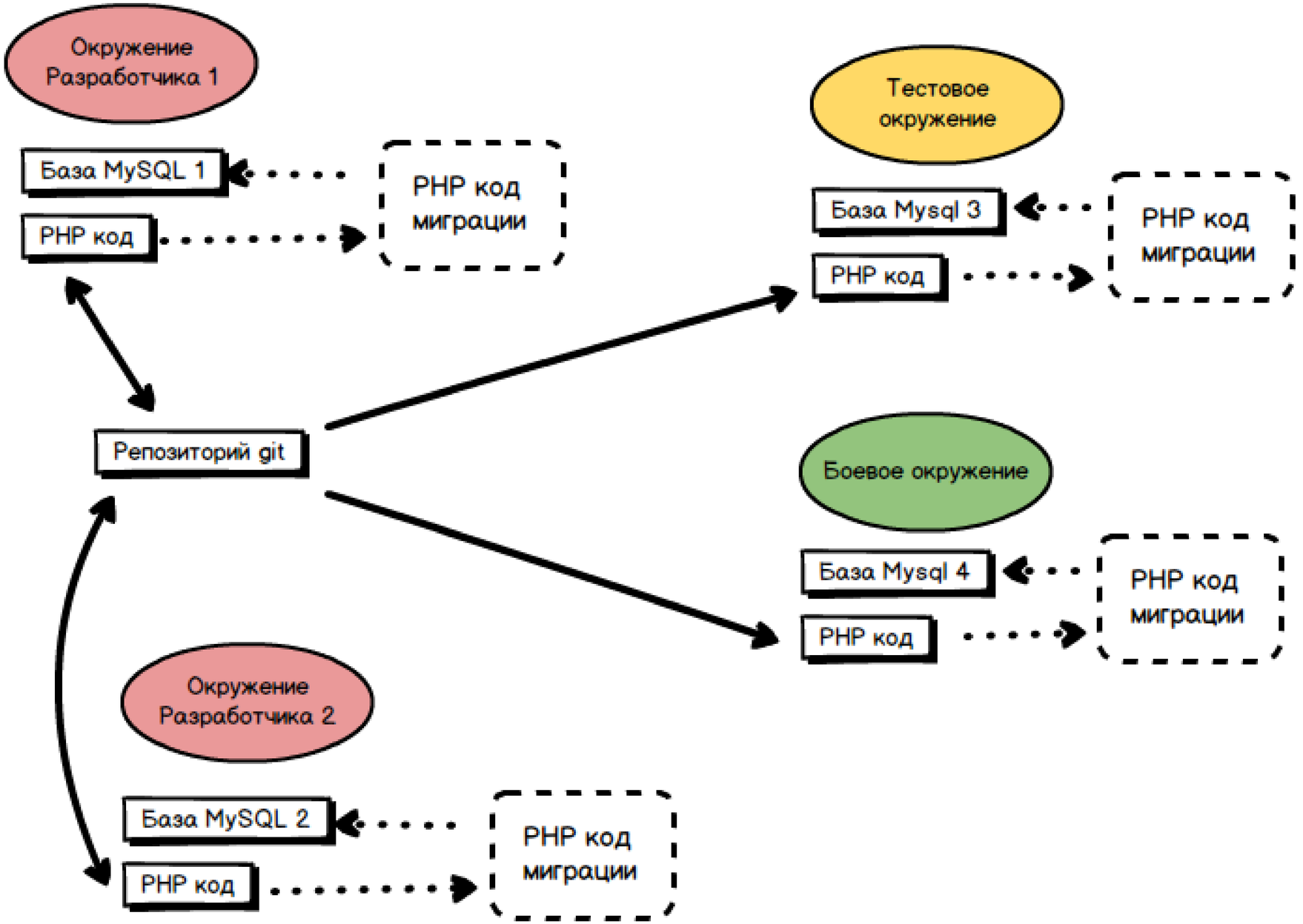
alex.smap

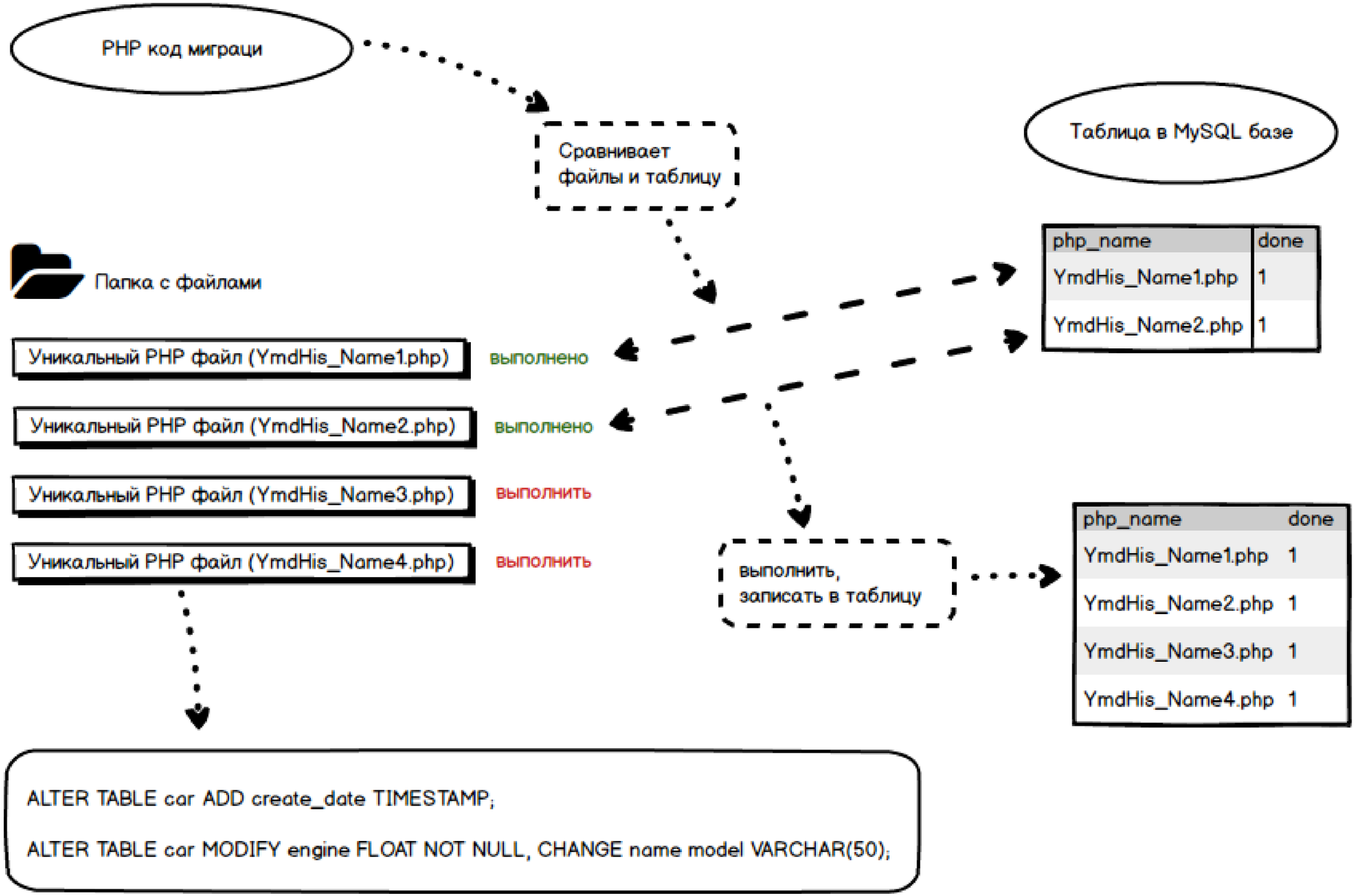
vk.com/alex.smap

fb.com/alex.smap

ПЛАН ЗАНЯТИЯ

- Мотивация. Миграция
- Пользователи и права доступа
- SQL запросы


Окружение Разработчика 1
База MySQL 1
PHP код
PHP код миграции
Тестовое окружение
База Mysql 3
PHP код
PHP код миграции
Репозиторий git
Боевое окружение
База Mysql 4
PHP код
PHP код миграции
Окружение Разработчика 2
База MySQL 2
PHP код
PHP код миграции


PHP код миграци
Сравнивает
файлы и таблицу
Таблица в MySQL базе
Папка с файлами
Уникальный PHP файл (YmdHis_Name1.php)
выполнено
Уникальный PHP файл (YmdHis_Name2.php)
выполнено
Уникальный PHP файл (YmdHis_Name3.php)
выполнить
Уникальный PHP файл (YmdHis_Name4.php)
выполнить
выполнить,
записать в таблицу
php_name done
YmdHis_Name1.php 1
YmdHis_Name2.php 1
php_name done
YmdHis_Name1.php 1
YmdHis_Name2.php 1
YmdHis_Name3.php 1
YmdHis_Name4.php 1
ALTER TABLE car ADD create_date TIMESTAMP;
ALTER TABLE car MODIFY engine FLOAT NOT NULL, CHANGE name model VARCHAR(50);


нетология
центр онлайн-образования
ПОЛЬЗОВАТЕЛИ MYSQL

СОЗДАЕМ ПОЛЬЗОВАТЕЛЯ И НАДЕЛЯЕМ ЕГО ПРАВАМИ

1. Создание пользователя:

```
CREATE USER 'user'@'localhost' IDENTIFIED BY 'password’;
```

2. Наделением его правами:

```
GRANT ALL PRIVILEGES ON test.* TO 'user'@'localhost';
```

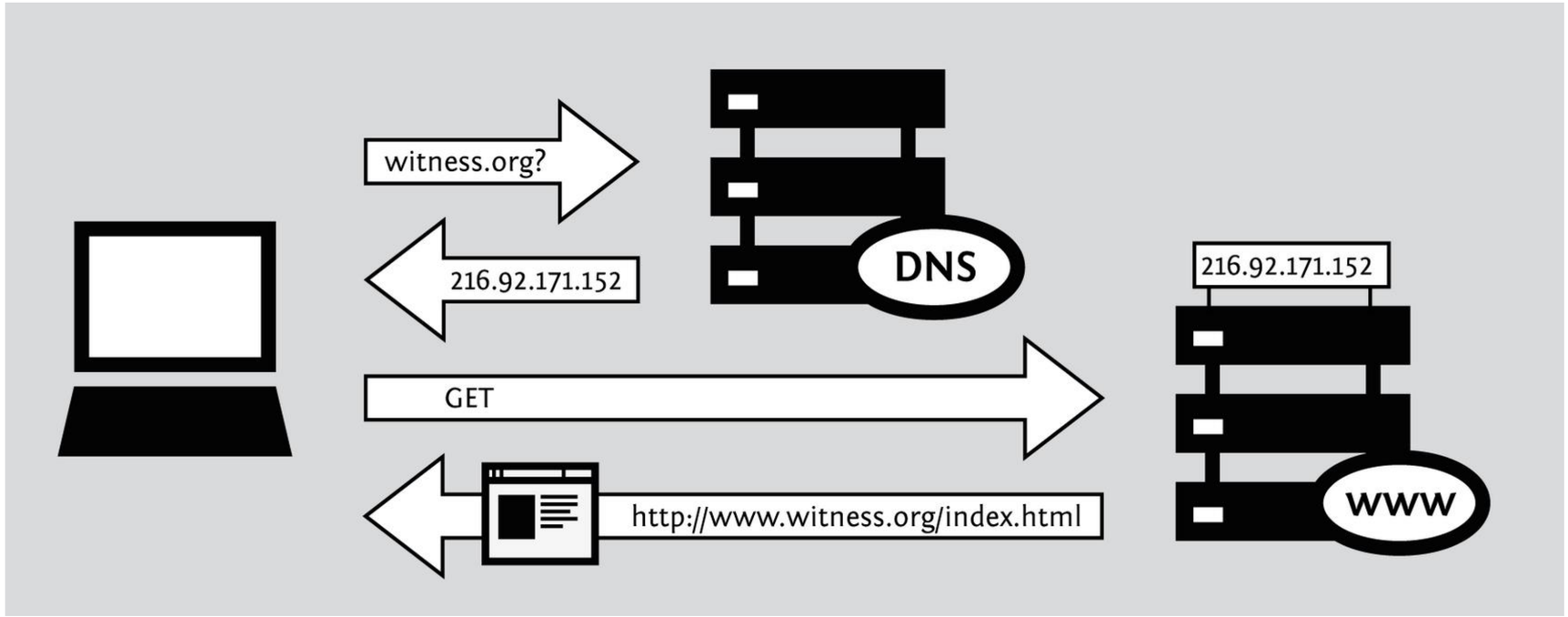
```
GRANT SELECT, UPDATE ON test2.* TO 'user'@'localhost';
```

3. Обновление всех прав доступа:

```
FLUSH PRIVILEGES;
```

ШАБЛОН

```
GRANT [тип прав][,тип прав]

ON [название базы данных или *].[название таблицы или *]

TO '[имя пользователя]'@'[домен]';
```


witness.org?
216.92.171.152
DNS
GET
http://www.witness.org/index.html
216.92.171.152
WWW

РАЗЛИЧНЫЕ ПРАВА ДОСТУПА

ALL PRIVILEGES – полный доступ к заданной базе данных

CREATE – позволяет создавать новые таблицы или базы данных.

DROP – позволяет удалять таблицы или базы данных.

DELETE – позволяет удалять строки из таблиц.

INSERT – позволяет добавлять строки в таблицу.

SELECT – поволит использовать команду Select для чтения из баз данных.

UPDATE – позволит редактировать строки таблиц.

GRANT OPTION – позволит назначать или удалять права доступа для других пользователей.

СОЗДАЕМ И УДАЛЯЕМ БАЗЫ ДАННЫХ

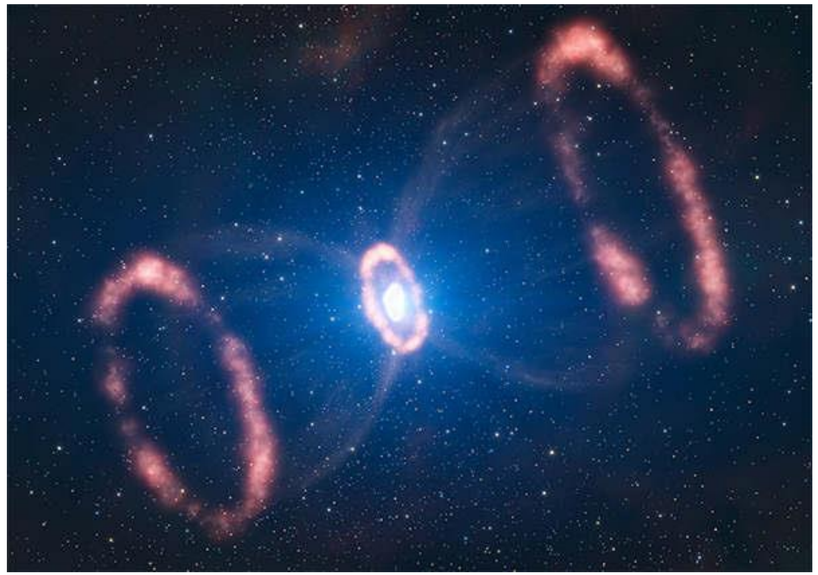
```
SHOW DATABASES;

CREATE DATABASE db_name;
RENAME DATABASE db_name TO new_db_name;

DROP DATABASE db_name;
DROP DATABASE IF EXISTS db_name;
```

нетология
центр онлайн-образования

УПРАВЛЕНИЕ ТАБЛИЦАМИ

СОЗДАНИЕ ТАБЛИЦЫ CREATE TABLE

```
CREATE TABLE `students` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NULL,
  `estimation`float NOT NULL,
  `budget` tinyint(4) NOT NULL DEFAULT '0',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```



```
CREATE TABLE `[Название таблицы]` (
...
) ENGINE=[тип таблицы] DEFAULT CHARSET=[кодировка];
```

column datatype [NULL | NOT NULL]

[DEFAULT default_value]
[AUTO_INCREMENT]
[UNIQUE KEY | PRIMARY KEY]
[COMMENT ‘string’],

Тип	Тип данных	Использование	Диапазоны
Целочисленное	TINYINT	Очень маленькое целое число	Диапазон числа со знаком от –128 до 127. Диапазон числа без знака (UNSIGNED) от 0 до 255.
	SMALLINT	Маленькое целое число	Диапазон числа со знаком от –32768 до 32767. Диапазон числа без знака (UNSIGNED) от 0 до 65535.
	MEDIUMINT	Среднее целое число	Диапазон числа со знаком от –8388608 до 8388607. Диапазон числа без знака (UNSIGNED) от 0 до 16777215.
	INT	Целое число	Диапазон числа со знаком от –2147483648 до 2147483647. Диапазон числа без знака (UNSIGNED) от 0 до 4294967295.
	BIGINT	Большое целое число	Диапазон числа со знаком от –9223372036854775808 до 9223372036854775807. Диапазон числа без знака (UNSIGNED) от 0 до 18446744073709551615.
Вещественные	FLOAT	Малое (одинарной точности) число с плавающей запятой. Не может быть числом без знака	Диапазоны от –3.402823466E+38 до –1.175494351E-38, 0 и 1.175494351E-38 до 3.402823466E+38. Если количество знаков после запятой не установлено или <= 24 это число с плавающей запятой одинарной точности.
	DOUBLE	Нормальное (двойной точности) число с плавающей запятой. Не может быть числом без знака	Диапазоны от -1.7976931348623157E+308 до -2.2250738585072014E-308, 0 и 2.2250738585072014E-308 до 1.7976931348623157E+308. Если количество знаков после запятой не установлени или 25 <= количество знаков <= 53 означает числоа с плавающей запятой двойной точности.

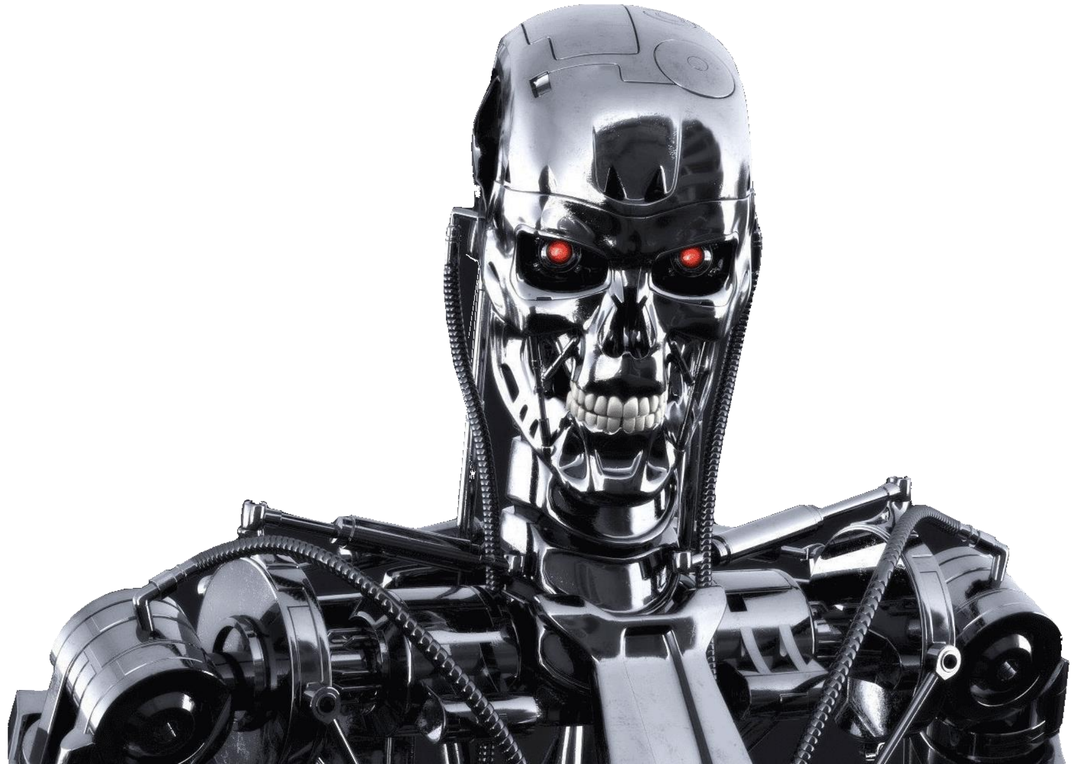
Временные	DATE	Дата	Дата в диапазоне от «1000-01-01» до «9999-12-31». MySQL хранит поле типа DATE в виде «YYYY-MM-DD» (ГГГГ-ММ-ДД).
	DATETIME	Дата и время	Допустимые диапазоны от «1000-01-01 00:00:00» до «9999-12-31 23:59:59». MySQL хранит поле типа DATETIME в виде «YYYY-MM-DD HH:MM:SS» (ГГГГ-ММ-ДД ЧЧ-ММ-СС).
	TIMESTAMP	Дата и время	Диапазон от «1970-01-01 00:00:00» до, примерно, 2037 года. MySQL может хранить поле типа TIMESTAMP в видах «YYYYMMDDHHMMSS» (TIMESTAMP(14)), «YYMMDDHHMMSS» (TIMESTAMP(12)), «YYYYMMDD» (TIMESTAMP(8)) и др.
	TIME	Время	Диапазон от «-838:59:59» до «838:59:59». MySQL хранит поле TIME в виде «HH:MM:SS», но позволяет присваивать значения столбцам TIME с использованием либо строки или числа.
	YEAR	Год в 2- или 4- хцифровом виде (4 цифры по-умолчанию)	Если вы используете 4 цифра, то допустимые значения 1901-2155, и 0000. Если 2 цифры, то 1970-2069 (70-69). MySQL хранит значения поля YEAR в формате «YYYY».
Текстовые	CHAR	Строка фиксированной длины, которая справа дополняются пробелами до указанной длины, при хранении.	Диапазон длины от 1 до 255 символов. Завершающие пробелы удаляются, когда значение извлекается. Значения CHAR сортируются и сравниваются без учета регистра в зависимости от кодировки по умолчанию, если не установлен флаг BINARY.
	VARCHAR	Строка переменной длины. Примечание: конечные пробелы удаляются при сохранении. При хранении используется только то количество символов, которое необходимо, плюс 1 байт для записи длины.	Диапазон длины от 1 до 21 844 (UTF-8) символов. Значения VARCHAR сортируются и сравниваются без учета регистра, если не установлен флаг BINARY. Примечания: VARCHAR медленее из-за того, что граница у него плавающая, CHAR идёт блоками, из-за чего можно точно предсказать где заканчивается одна запись и начинается другая.
	TINYTEXT, TEXT, MEDIUMTEXT, LONGTEXT	Строка переменной длины. Примечание: Информация TEXT хранится вне основных данных (внутри данных - ссылка на расположение). Отсюда уменьшенная скорость доступа и возможные ограничения. При хранении используется только то количество символов, которое необходимо, плюс байты для записи ссылки: TINYTEXT - 1 байт, TEXT- 2 байта, MEDIUMTEXT - 3 байта, LONGTEXT - 4 байта.	TINYTEXT - 255 байт, TEXT- 64 Кб, MEDIUMTEXT - 16 Мб, LONGTEXT - 4 Гб Рекомендации: используем если база MySQL < 5.0.3 или когда нужен объем текста больше 64 кб.

ИНФОРМАЦИЯ О ТАБЛИЦЕ

```
SHOW TABLES;
DESCRIBE table_name;
```

УДАЛЕНИЕ ТАБЛИЦЫ DROP TABLE

```
DROP TABLE tbl_name;

DROP TABLE IF EXISTS tbl_name;
```



ОЧИСТКА ТАБЛИЦЫ TRUNCATE

```
TRUNCATE TABLE table_name;

DELETE FROM table_name;
```

УПРАВЛЕНИЕ ТАБЛИЦЕЙ ALTER TABLE

```
CREATE TABLE cars (engine INTEGER, name VARCHAR(10));
```

Переименовываем таблицу:

```
ALTER TABLE cars RENAME car;
```

Изменяем тип поля:

```
ALTER TABLE car MODIFY engine FLOAT NOT NULL;
```

Изменяем название поля:

```
ALTER TABLE car CHANGE name model VARCHAR(50);
ALTER TABLE car MODIFY engine FLOAT NOT NULL, CHANGE name model VARCHAR(50);
```

Добавляем поле:

```
ALTER TABLE car ADD create_date TIMESTAMP;
```

Удаляем поле:

```
ALTER TABLE car DROP COLUMN create_date;
```

Добавляем ключ:

```
ALTER TABLE car ADD id INT UNSIGNED NOT NULL AUTO_INCREMENT, ADD PRIMARY KEY (id);
```

Меняем местами поля:

```
ALTER TABLE car CHANGE id id INT UNSIGNED NOT NULL AUTO_INCREMENT FIRST;
ALTER TABLE car CHANGE id id INT UNSIGNED NOT NULL AUTO_INCREMENT AFTER id;
```

нетология
центр онлайн-образования

ИНДЕКСЫ

ЧТО ТАКОЕ ИНДЕКС?

Индекс — это отсортированный набор значений.

В MySQL индексы всегда строятся для какой-то конкретной колонки. Например, когда мы делаем колонку **id** нашим **первичным ключом**, она автоматически получает индекс.

ЗАЧЕМ НУЖНЫ ИНДЕКСЫ?

Индексы применяются для быстрого поиска строк.

Без индекса чтение таблицы осуществляется по всей таблице начиная с первой записи, пока не будут найдены соответствующие строки.

Если таблица содержит индекс, то MySQL может быстро определить позицию для поиска в середине файла данных без просмотра всех данных.

Для таблицы, содержащей 1000 строк, это будет как минимум в 100 раз быстрее по сравнению с последовательным перебором всех записей.

КАК РАБОТАЕТ ИНДЕКС?

Как мы помним, индекс — это отсортированный набор значений.

А если мы знаем, что данные отсортированы, мы можем использовать алгоритм бинарного поиска, или проще говоря, поиск путем «деления пополам».

Собственно, MySQL этим и занимается.

ДОБАВЛЯЕМ ИНДЕКСЫ

```
CREATE INDEX age ON random_people(age);
CREATE INDEX age_iq ON random_people(age, iq);

CREATE UNIQUE INDEX name ON random_people(name);


CREATE INDEX [название индекса] ON [название таблицы]([поле][,поле]);
```

А В ЧЕМ ПОДВОХ?

Индексы – это дополнительные операции записи на диск.

При каждом обновлении или добавлении данных в таблицу, происходит также запись и обновление данных в индексе.

Создавайте только необходимые индексы, чтобы не расходовать зря ресурсы сервера.

ОСОЗНАЕМ ПРОИСХОДЯЩЕЕ EXPLAIN

EXPLAIN поможет получить информацию о том, как происходило выполнение запроса, и какие индексы были задействованы.

id	select_type	table	type	key	key_len	ref	rows	Extra
1	PRIMARY		ALL				24	Using temporary; Using filesort
2	DERIVED	dm	index	emp_no	4		24	Using index; Using temporary; Using filesort
2	DERIVED	m	eq_ref	PRIMARY	4	employees.dm.emp_no	1	
2	DERIVED	de	ref	dept_no	12	employees.dm.dept_no	1	Using index
2	DERIVED	e	eq_ref	PRIMARY	4	employees.de.emp_no	1	Using index

The query had to use a temp table. Typically used when Group By and Order By clauses list columns in different orders

5 row(s) returned in 0.0009 second(s).

нетология
центр онлайн-образования

Спасибо за внимание!

ДАЦКОВ АЛЕКСЕЙ


alex.smap@gmail.com


alex.smap


vk.com/alex.smap


fb.com/alex.smap