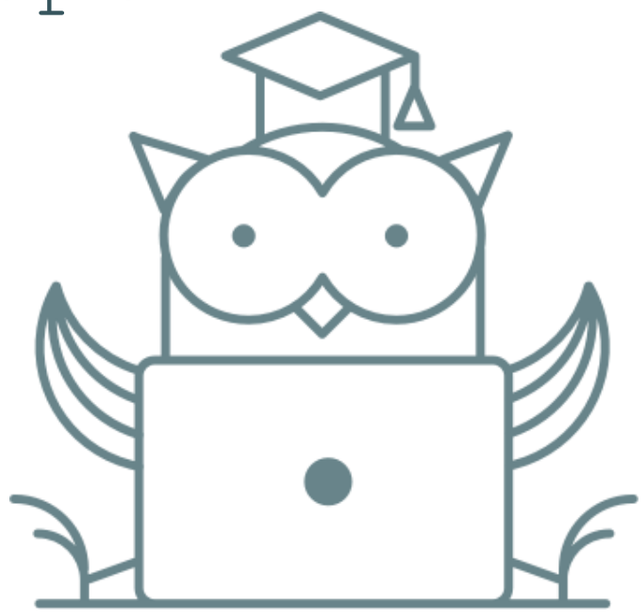
OTUS

ОНЛАЙН-ОБРАЗОВАНИЕ

```
let lesson = {
    id:         '07'
    theme:      'Стэк MEAN',
    date:       '20.08.2018',
    teacher: {
        name:       'Юрий Дворжецкий',
        position: 'Lead Developer'
    }
};
```



Как меня слышно && видно?

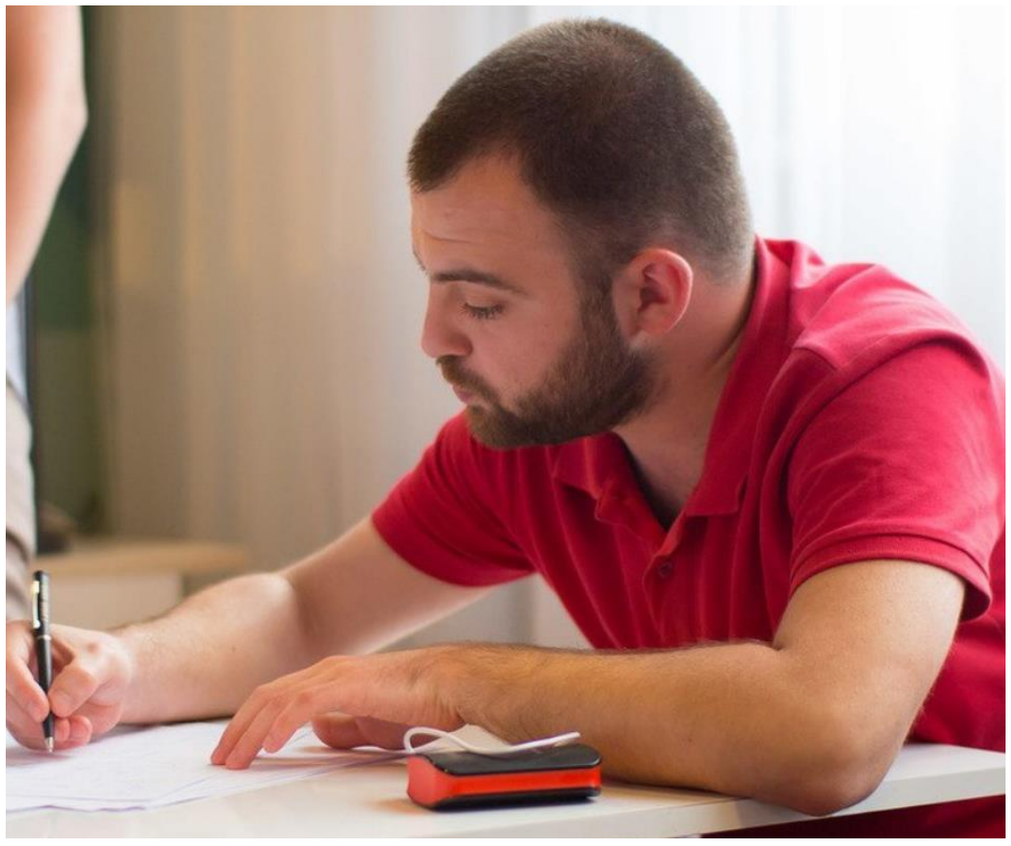
Если нет – напишите, если слышите – смайлик в чат.

Отвечаю на возникший вопрос

- А где Александр?

Я: Дворжецкий Юрий

- ведущий разработчик в Luxoft
- влюблён в JS, правда, женат на Java
- написал гигабайты кода в очень больших и маленьких проектах;
- провёл более 1500 часов курсов, тренингов и вебинаров;
- учил и подготовил более 600 разработчиков;
- почти побил мировой рекорд по отсутствию сна, но при этом отлично сдал проект.



Что сможем делать после вебинара?

- Хвастаться, что знаем как расшифровывается MEAN.
- Разрабатывать back-end на Express.
- Разбираться в NoSQL.
- Хранить данные в MongoDB.

О вебинаре:

- Несмотря на кажущуюся простоту – это очень важно.
- Если Вам кажется, что очень тяжело – не переживайте, это просто ☺
- Самым шиком будет использовать MongoDB в Вашем проекте и использовать правильно.

Заранее:

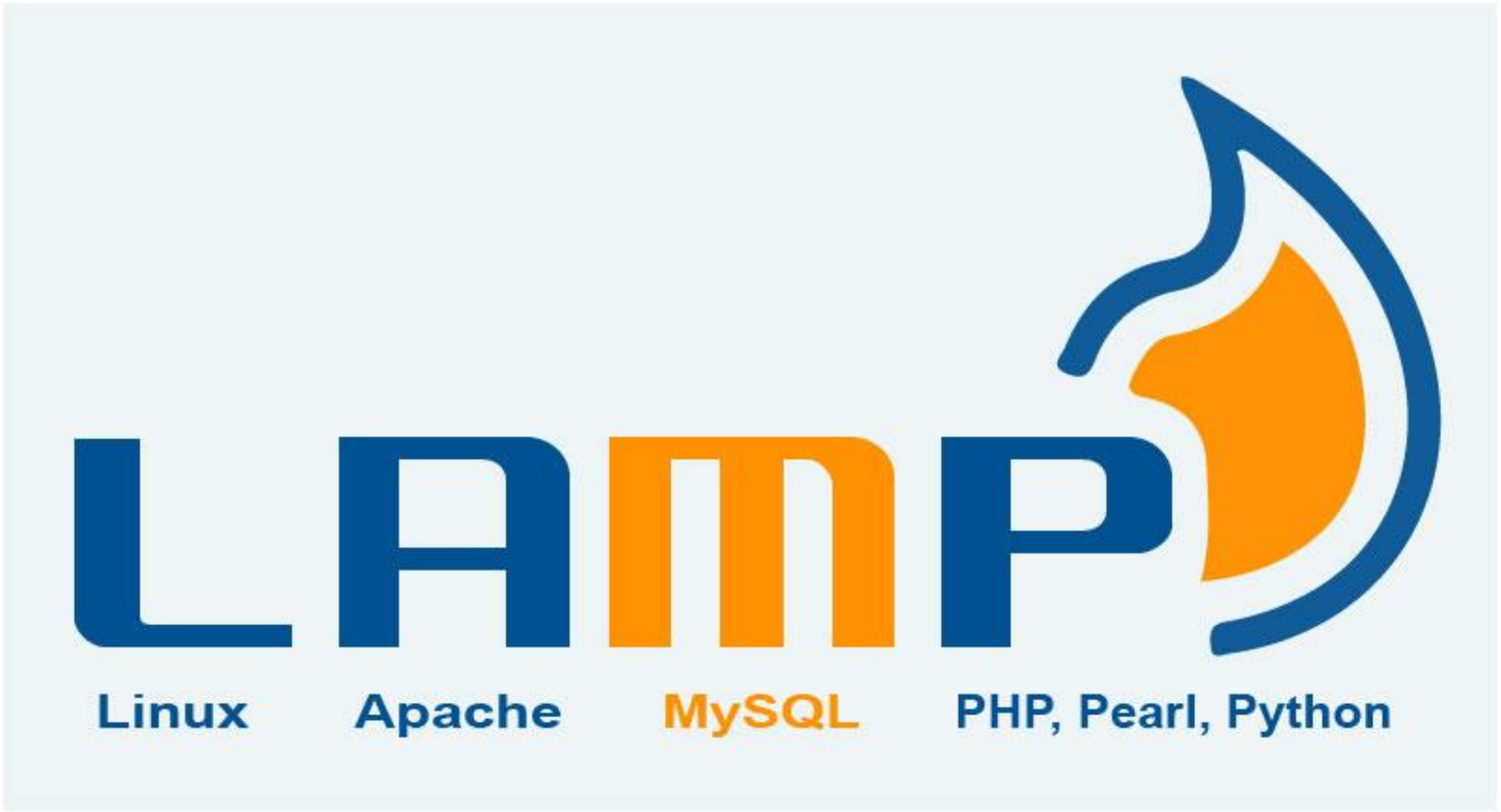
- Если хотите попробовать MongoDB просьба её скачать
- https://www.mongodb.com/download-center
- Вам нужен будет Community Server

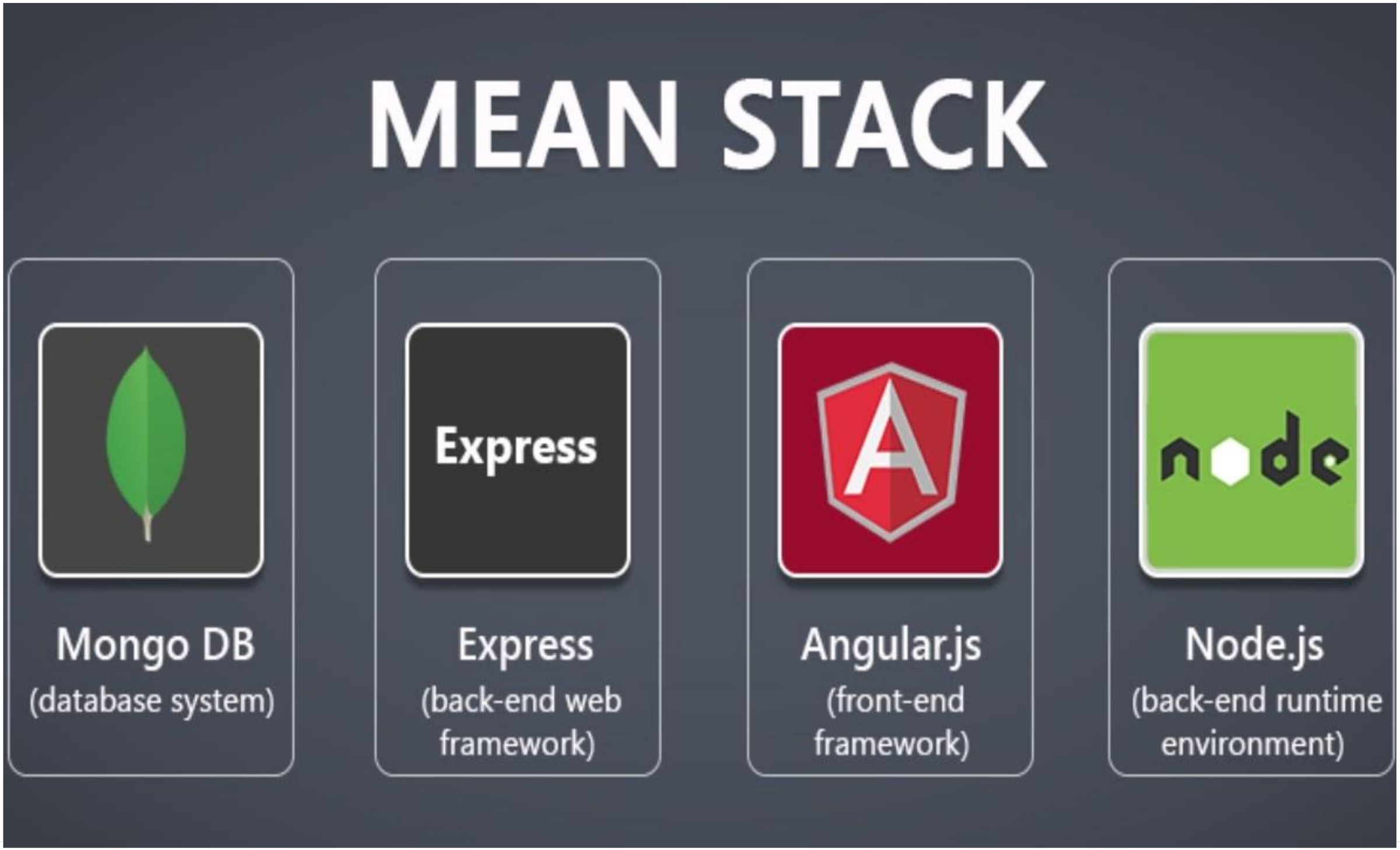
OTUS

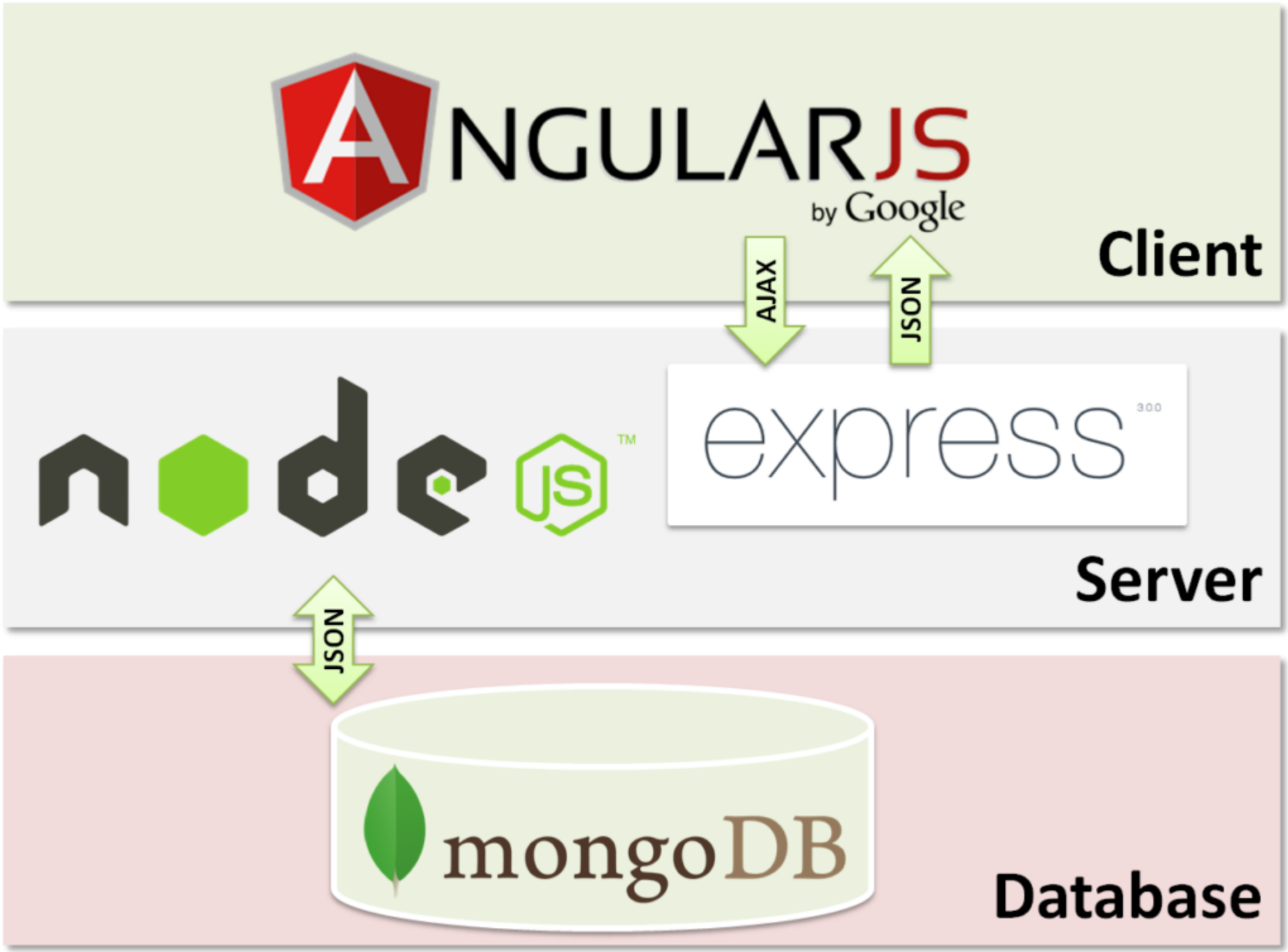
Поехали?

OTUS

MEAN


LAMP
Linux
Apache
MySQL
PHP, Pearl, Python


MEAN STACK
Mongo DB
(database system)
Express
Express
(back-end web framework)
Angular.js
(front-end framework)
Node.js
(back-end runtime environment)


ANGULARJS
by Google
Client
AJAX
JSON
express
Server
JSON
mongoDB
Database

M - MongoDB

1. NoSQL-База данных
2. Другие принципы работы (документ-ориентированная)
3. Интерфейс на JS (!)
4. Бесплатная (!) и платная
5. Эффективно работает с большими данными

E - Express

1. Фреймворк для написания back-end веб-приложений
2. Lightweight (прям совсем Lightweight)
3. Позволяет, как писать обработчики HTTP-запросов, так и выставлять наружу статический контент (веб-сервер)
4. Родной для NodeJS (!)

A - Angular

1. Angular – раньше был AngularJS, потом Angular2, Angular2, потом сразу Angular4 и Angular5, сейчас 6
2. Framework (!)
3. TypeScript (!)
4. Isomorphic apps (!)

N - NodeJS

1. Среда выполнения
2. Уже нам родная
3. Event-driven
4. JS (!)
5. Одна из самых популярных платформ для разработки веб-приложений

Вопросы?

OTUS

Express

Упражнение:
Установим Express?

// package.json

```
{
  "private": true,
  "dependencies": {
    "express": "4.16.3"
  }
}
```

Или

```
npm init
npm i express –save
```

Упражнение:
Установим Express?

```
// server.js
var express = require('express');
var app = express();

app.get('/', function(req, res){
    res.send('Hello World');
});

app.listen(3000);
```

и npm start
и откроем localhost:3000

DEMO

Упражнение:
Установим Express?

// server.js и npm start localhost:3000/name?name=Ivan

```
var express = require('express');
var app = express();

app.get('/hello', function(req, res){
    res.send('Hello ' + req.query.name);
});

app.listen(3000);
```

Упражнение:
Установим Express?

// server.js и npm start localhost:3000/hello/1

```
var express = require('express');
var app = express();

app.get('/hello/:id', function(req, res){
    res.send('Hello ' + req.params.id);
});

app.listen(3000);
```

Упражнение:
Установим Express?

// server.js и npm start localhost:3000/name?name=Ivan

```
var express = require('express');
var app = express();

app.get('/hello/:id?', function(req, res){
    res.send('Hello ' + req.params.id);
});

app.listen(3000);
```

Object

Упражнение:
Установим body-parser?

// package.json

```
{
  "private": true,
  "dependencies": {
    "body-parser": "1.18.3",
    "express": "4.16.3"
  }
}
```

```
var express = require('express');
var bodyParser = require('body-parser')
var app = express();

app.use(bodyParser.json())

app.post('/hello', function(req, res){
    res.status(201).send(req.body.name);
});

app.listen(3000);
```

```
app.use(function (req, res, next) {
  console.log('Time:', Date.now());
  next();
});
```

```
app.use('/user/:id', function(req, res, next) {
  console.log('Request URL:', req.originalUrl);
  next();
}, function (req, res, next) {
  console.log('Request Type:', req.method);
  next();
});
```

Object

```
var options = {
    etag: false,
    extensions: ['htm', 'html'],
    index: false, maxAge: '1d',
    redirect: false
};
app.use(express.static('public', options));


app.use(express.static('uploads'));
app.use(express.static('files'));
```

Что прочитать попробовать?

- Различные Middleware
- Публикацию веб-страниц (веб-сервер)

Несколько слов про express

Если Вы будете использовать webpack

А вместе с ним dev-server

То там внутри есть express

dev-server.before

Object

Упражнение:

Сделать хранилище пользователей с корректным управлением id

GET /person/1
response: {id: 1, name: "Ivan“, age: 15}

GET /person/create?name=Irina&age=18
response: {id: 2, name: “Irina“, age: 18}

Что из этого и для чего?

GET /person/:id

POST /person

PUT /person/:id

PATCH /person/:id

DELETE /person/:id

Что из этого и для чего?

GET /person/ - все

GET /person/:id – получение конкретного

POST /person - создание

PUT /person/:id – изменение/замена

PATCH /person/:id – изменение/обновление

DELETE /person/:id - удаление

Object

Мой любимый вопрос на собеседованиях

В чём различие POST и PUT ?

Идемпотентность

Мой любимый вопрос на собеседованиях

POST неидемпотентная

PUT идемпотентная

Что прочитать попробовать?

http://www.restapitutorial.com/lessons/httpmethods.html

Twitter API, Яндекс-Диск, Яндекс-ПДД, Яндекс-* API

SoapUI (REST)

Postman (Chrome extension)

Вкладка Network в Chrome на сайтах

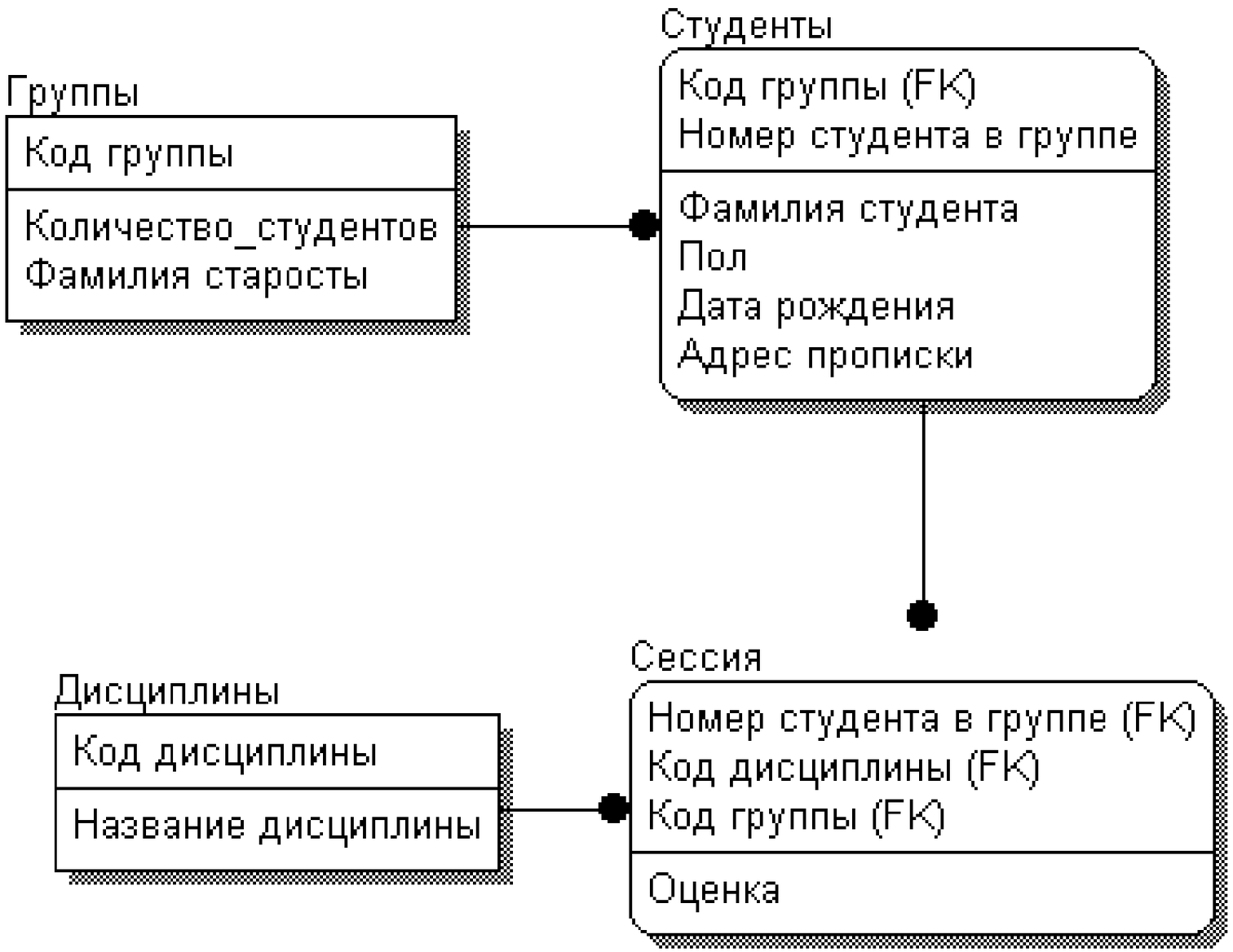
OTUS

NoSQL

OTUS

SQL и NoSQL БД

SQL DBs (RDB)


Группы
Код группы
Количество_студентов
Фамилия старосты
Студенты
Код группы (FK)
Номер студента в группе
Фамилия студента
Пол
Дата рождения
Адрес прописки
Дисциплины
Код дисциплины
Название дисциплины
Сессия
Номер студента в группе (FK)
Код дисциплины (FK)
Код группы (FK)
Оценка

SQL DBs (RDB)

- Под SQL базами понимают реляционные базы данных, потому что запросы к ним пишутся на SQL языке

- Данные строго структурированы в виде таблиц

- У них есть транзакции

- Часто они развёрнуты в кластере (горизонтальное масштабирование) обеспечивающим бОльшую надёжность (хотя для них достаточно тяжеловато)

- Вертикально масштабируются прекрасно.

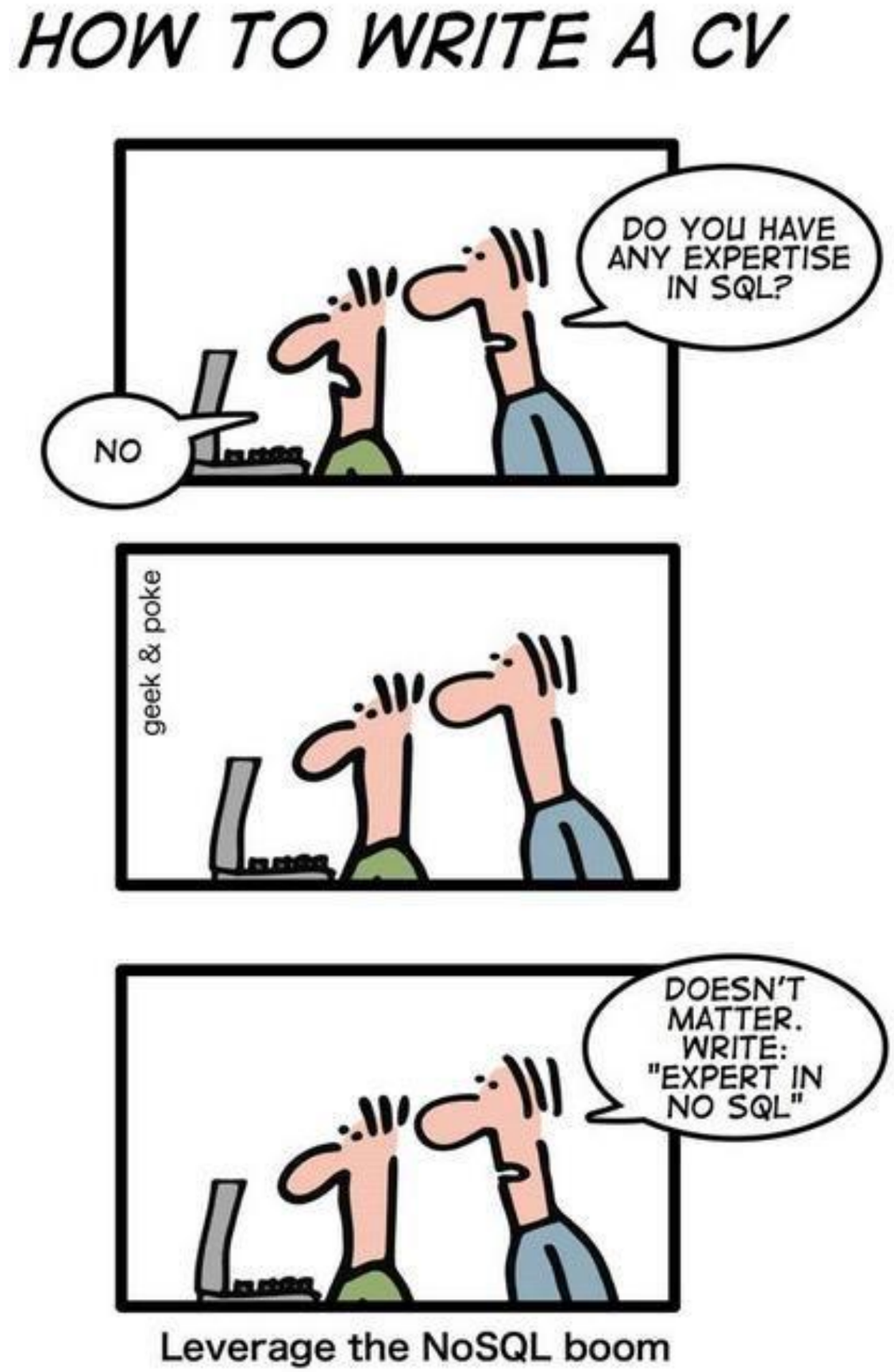
SQL DBs (RDB)

- SQL DB появились давно, как и реляционные БД
- Поэтому все БД и используют SQL
- Они очень популярны (хотя когда я был в детском саду, уже тогда были нереляционные БД)
- Многие технологии выстроены вокруг них (*DBC, ORM)

Где используются SQL

- Там где действительно важны транзакции. Банковские операции и т.д.
- Если данные реально представляют собой таблицы, а структура меняется редко.

NoSQL (not only SQL)



NoSQL

- NoSQL – это «buzzword» ☺ общее название для баз данных, не использующих реляционную модель

- Но мы то знаем, что SQL и реляционная модель – это так-то разные вещи)

- Они были давно, но использовались ну в ооооочень специфичных ситуациях.

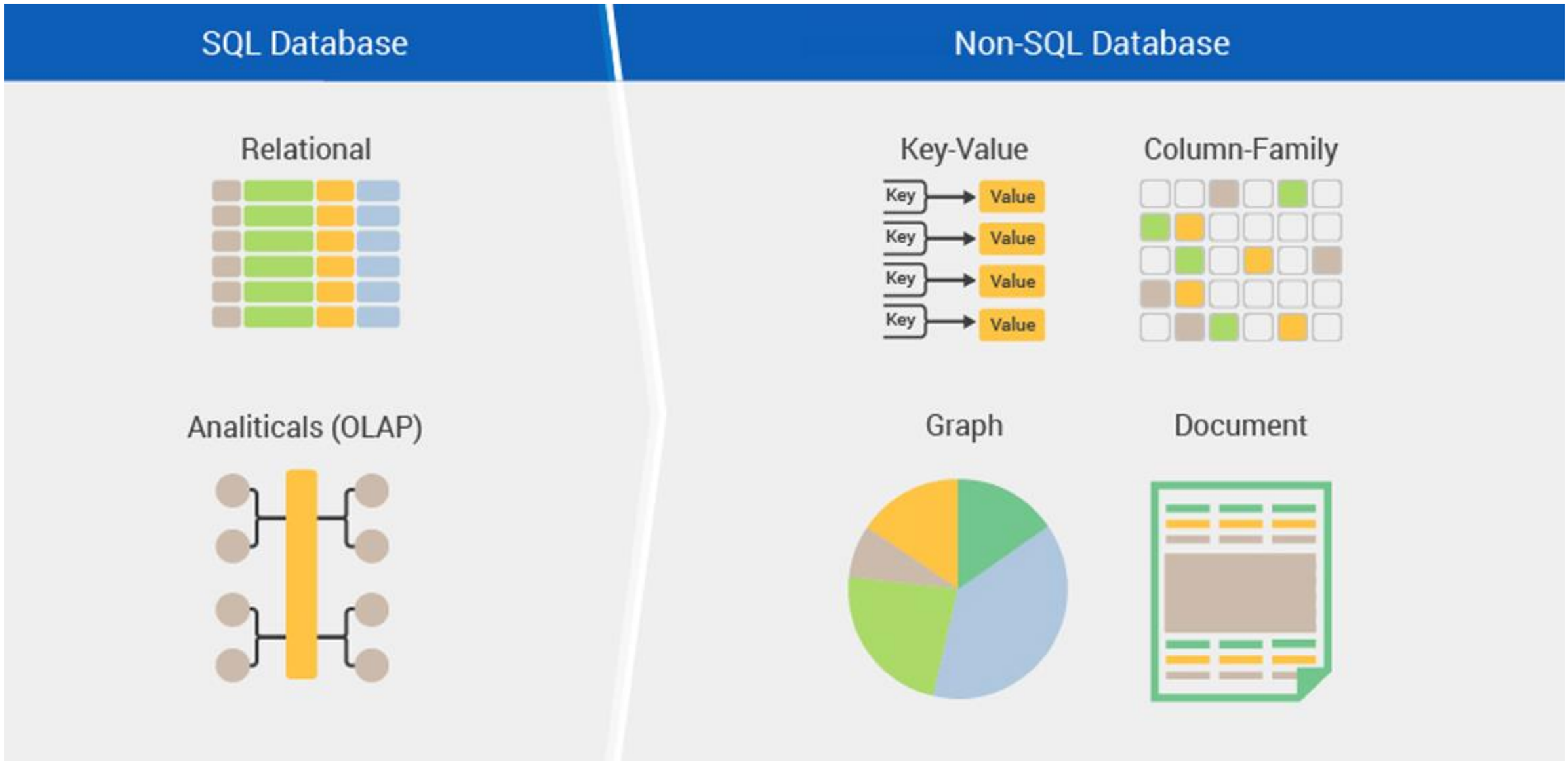
- Получили широкое распространение с популярностью BigData

- Ну и сейчас в NoSQL БД записали, то что не является БД как таковыми – например, распределённые кэши.

Принципы хранения баз даных

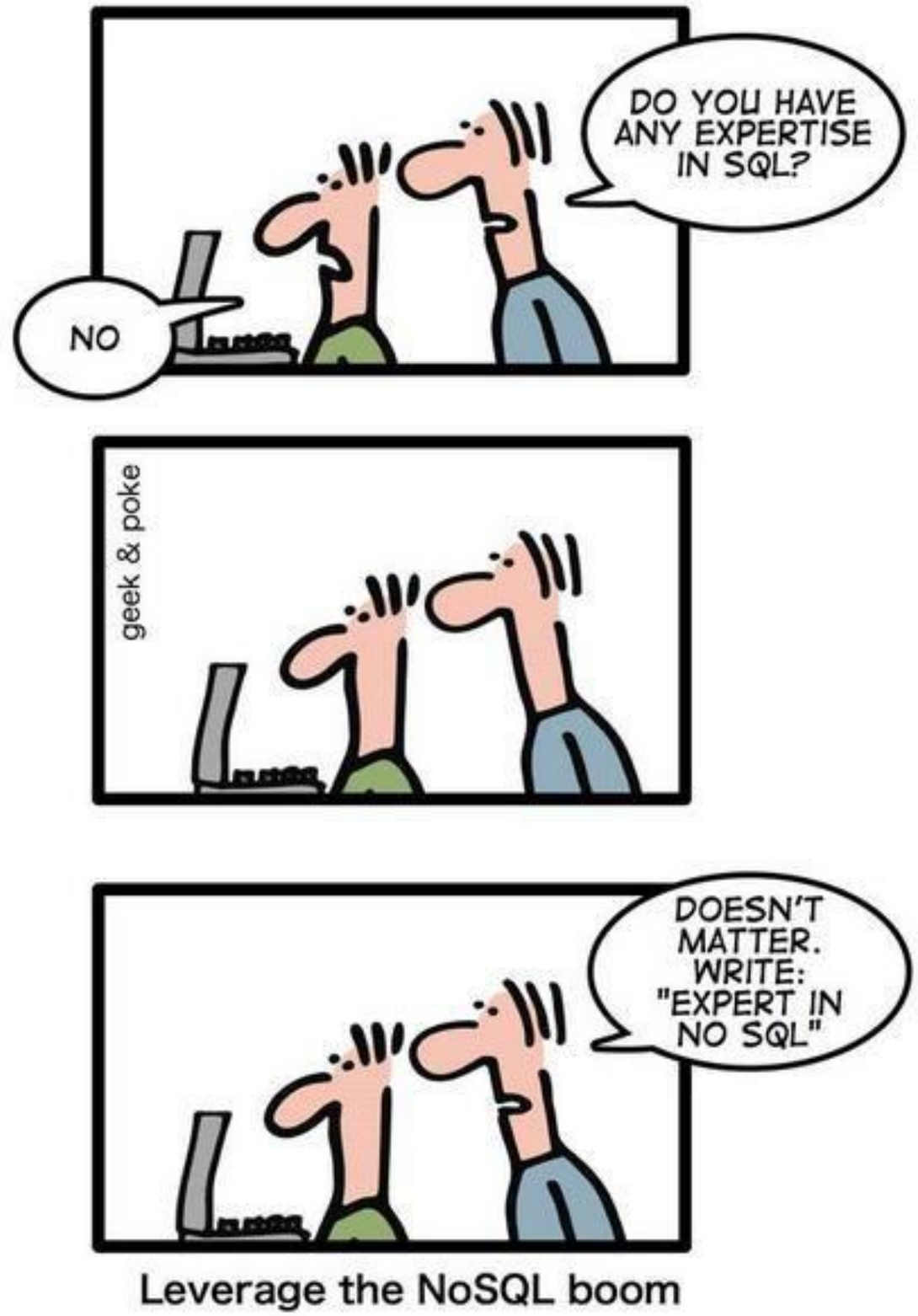
- Каждая БД выбирает свои принципы хранения данных.
- Часто даже одни и те же принципы хранения разительно отличаются в разных БД.
- Ну и принципы одно – а интерфейс доступа к БД – другое.

Принципы хранения данных



Вопросы?

NoSQL


HOW TO WRITE A CV
DO YOU HAVE ANY EXPERTISE IN SQL?
NO
geek & poke
DOESN'T MATTER. WRITE: "EXPERT IN NO SQL"
Leverage the NoSQL boom

NoSQL

- NoSQL – это «buzzword» ☺ общее название для баз данных, не использующих реляционную модель

- Но мы то знаем, что SQL и реляционная модель – это так-то разные вещи)

- Они были давно, но использовались ну в ооооочень специфичных ситуациях.

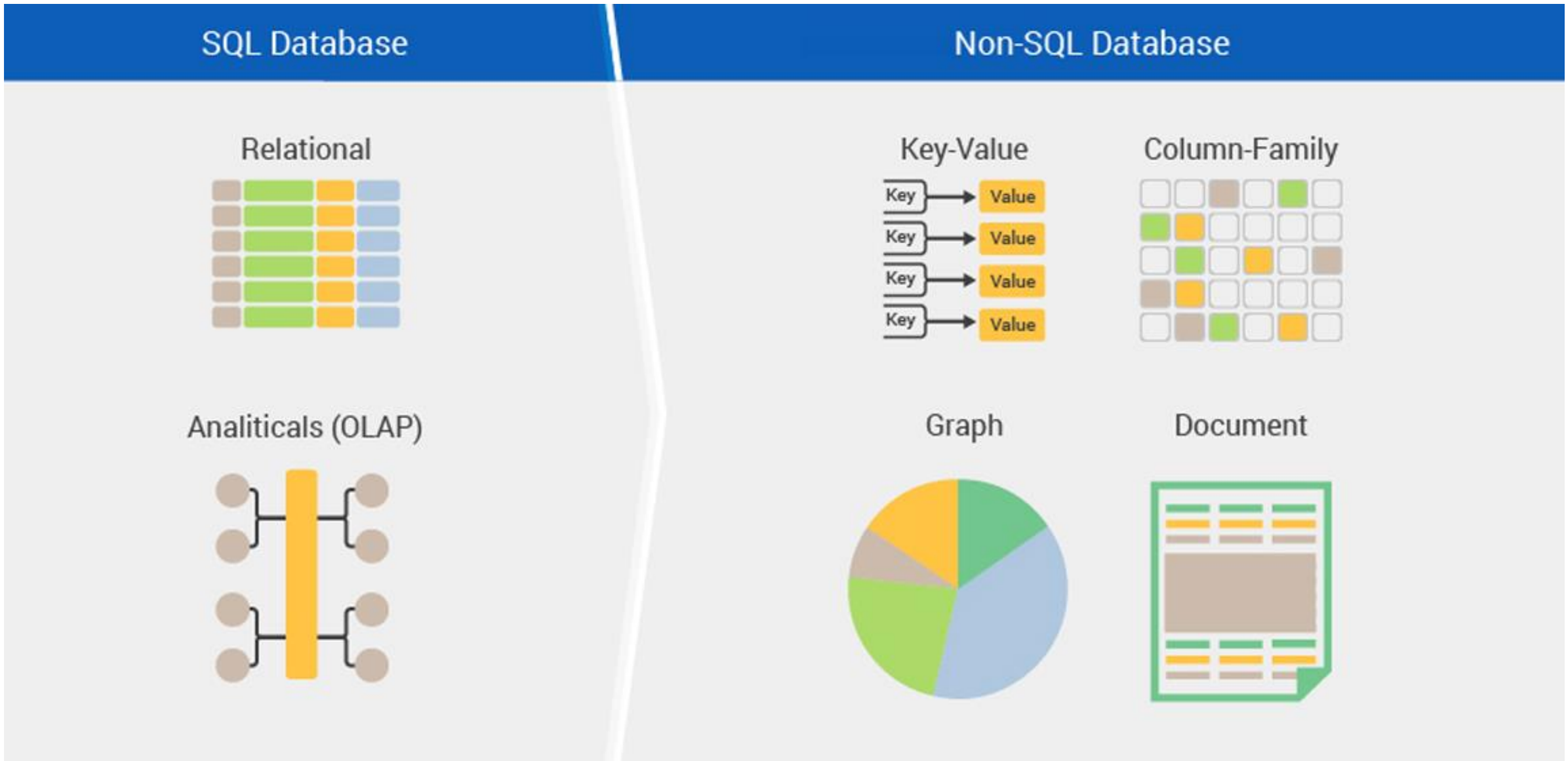
- Получили широкое распространение с популярностью BigData

- Ну и сейчас в NoSQL БД записали, то что не является БД как таковыми – например, распределённые кэши.

Принципы хранения баз даных

- Каждая БД выбирает свои принципы хранения данных.
- Часто даже одни и те же принципы хранения разительно отличаются в разных БД.
- Ну и принципы одно – а интерфейс доступа к БД – другое.

Принципы хранения данных



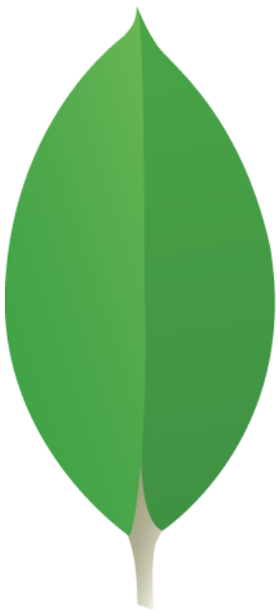
Документ-ориентированные

- У Вас есть структурированный документ (JSON)
- От записи к записи он может сильно разнится (описание товаров в интернет-магазине)*
- Документы могу содержать ссылки на другие документы
- А самое страшное, что они могу редактироваться и запросы могут делаться по полям (опять же. Которые могут отсутствовать)

Документ-ориентированные

- MongoDB – сейчас покажу
- Couchbase
- CouchDB
- ElasticSearch – основная фишка – полнотекстовый поиск
- Informix – прям для документов
- Solr
- PostgreSQL ☺

OTUS



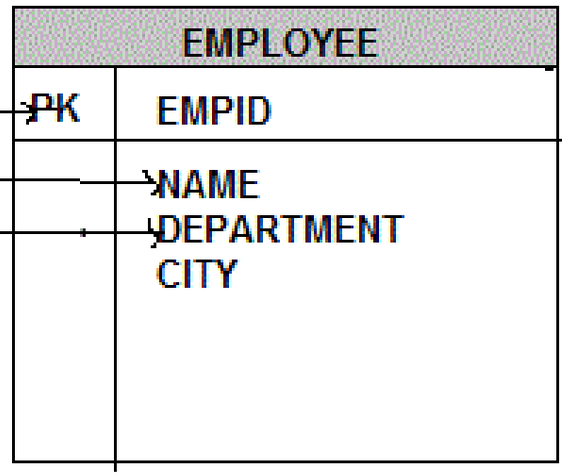
mongoDB.®

M - MongoDB

1. Масштабируемая.
2. Высокопроизводительная
3. Интерфейс на JS (!)
4. Простая в использовании
5. Эффективно работает с большими данными
6. Хранит документы (в нереляционном виде)
7. Никаких JOIN-ов, нет транзацкий между документами
8. Не подходит для денег

Объекты MongoDB

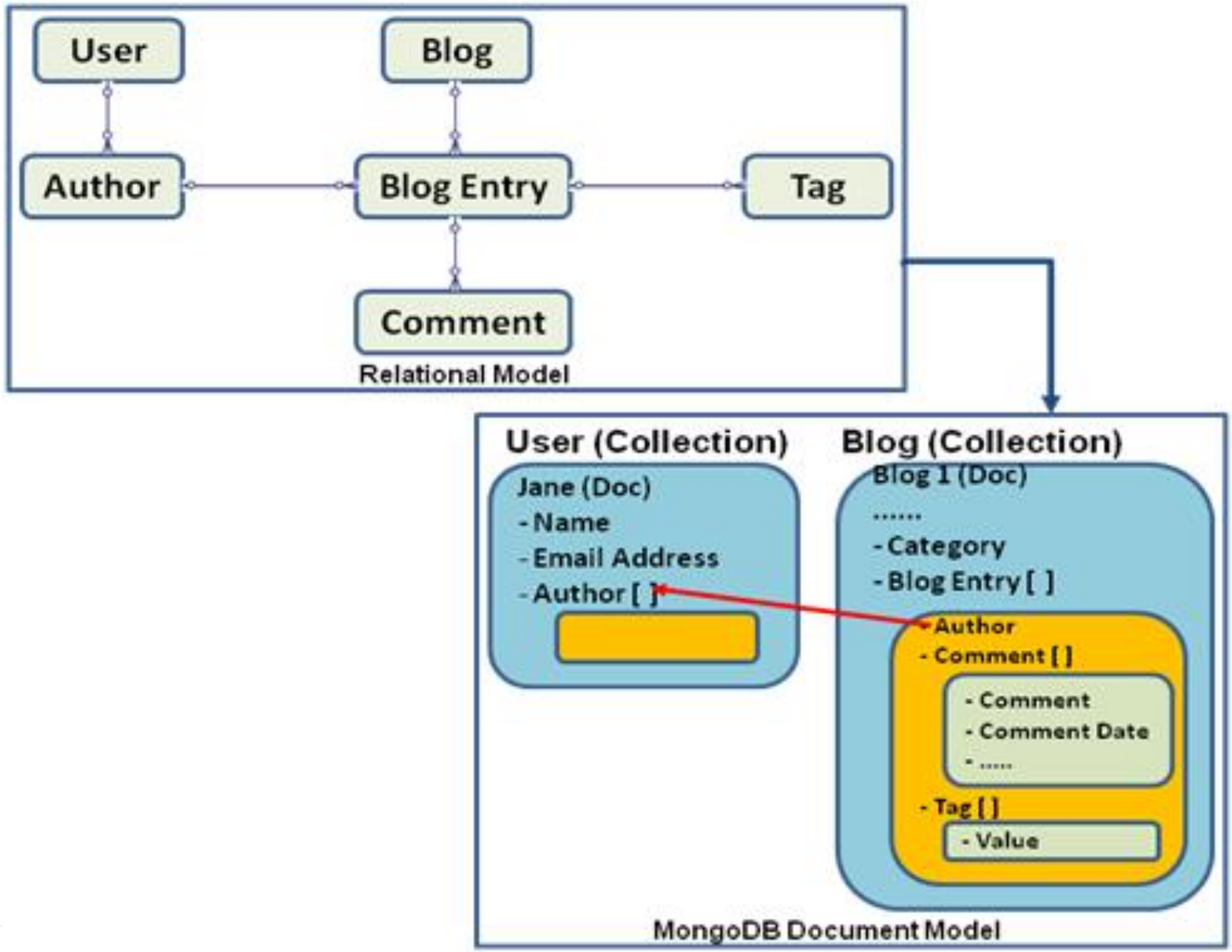
```
Public class Employee
{
    private int empId;
    private String name;
    private String department;
    ...
    ...
}
```



No Impedance Mismatch

```
// your application code
class Foo { int x; string [] tags;}

// mongo document for Foo
{ x: 1, tags: ['abc','xyz'] }
```


User
Blog
Author
Blog Entry
Tag
Comment
Relational Model
User (Collection)
Jane (Doc)
- Name
- Email Address
- Author [ ]
Blog (Collection)
Blog 1 (Doc)
.....
- Category
- Blog Entry [ ]
- Author
- Comment [ ]
- Comment
- Comment Date
- .....
- Tag [ ]
- Value
MongoDB Document Model

Database (MongoDB) – Database (SQL)

1. Database – база данных
2. Состоит из коллекций
3. Сама создаётся, если к ней обращаются

Collection (MongoDB) – Table (SQL)

1. Collection – коллекция
2. Состоит из документов
3. Индексируется причём хитрыми индексами

Document (MongoDB) – Row (SQL)

1. Document - документ
2. По сути дела – просто JSON
3. Хранится в формате JSONB
4. Есть _id – вроде primary key
5. Поддерживает отношения с другим документами – по ссылки или embedded

DEMO

Object

```
// Пробуем?
db.products.insert(
   { _id: 10, item: "box", qty: 20 }
)

// Ещё?
```

// Пробуем?

```
db.products.find({ _id: 10})
```

```
// Пробуем?


db.products.find({ _id: 10})

db.products.find({ item: “box”})
```

```
// дадада
db.products.find({ _id: { $gt: 5 }})

// проекция
db.products.find(
   { _id: { $gt: 5 } }, { item: 1, qty: 1 }
)
```

```
// Пробуем?

db.products.find({ _id: 10})

db.products.find({ _id: { $gt: 5 }})

db.products.find({ item: “box”})

db.products.find(
   { _id: { $gt: 5 } }, { item: 1, qty: 1 }
)
```

OTUS

// Пробуем?

```
db.products.find().sort({ name: 1 }).limit( 5 )
```

```
db.products.find().limit(5).sort( { name: 1 } )
```

```
// Пробуем?


db.prodcuts.update(
    { id: { $gte: 100 } },
    { $set: { "weight" : 100 } },
    {
      multi: true,
      $addToSet: {items: "box"}
    }
)

$set, $unset, $push, $pull, $pop, $addToSet,
$incr, $decr ...
```

```
db.collection.findAndModify({
    query: <document>,
    sort: <document>,
    remove: <boolean>,
    update: <document>,
    new: <boolean>,
    fields: <document>,
    upsert: <boolean>,
    bypassDocumentValidation: <boolean>,
    writeConcern: <document>,
    collation: <document>,
    arrayFilters: [ <filterdocument1>, ... ]
});
```

```
{
  "private": true,
  "dependencies": {
    "express": "4.16.3",
    "mongodb": "3.1.3"
  }
}
```

mongodb

1. https://mongodb.github.io/node-mongodb-native/
2. Официальный драйвер
3. Кстати, не единственный, есть ещё mongoose

```
const express = require('express');
const app = express();
const MongoClient = require('mongodb').MongoClient;

MongoClient.connect("mongodb://localhost:27017",
    function (err, client) {
        if(err) return console.log(err);

        // !!!
        app.get('/hello', function (req, res) {
            res.send('Hello ' + req.query.name);
        });

        app.listen(3000, function (){
            console.log('We are live on ' + port);
        });
});
```

mongoose

1. https://mongoosejs.com/
2. Так-то более удобный
3. Имеет подержку схем
4. И все почему-то пользуются им

mongoose

```
const mongoose = require('mongoose');
mongoose.connect('mongodb://localhost/test');

const Cat = mongoose.model('Cat', { name: String });

const kitty = new Cat({ name: 'Zildjian' });
kitty.save().then(() => console.log('meow'));
```

Упражнение:

Сделать хранилище пользователей на основе MongoDB

GET /person/1
response: {id: 1, name: "Ivan“, age: 15}

GET /person/create?name=Irina&age=18
response: {id: 2, name: “Irina“, age: 18}

Что прочитать попробовать?

https://docs.mongodb.com/manual/reference/method/js-collection/

Репликация

«Транзакции» в mongoDB

Чем NoSQL отличается от SQL

MonogoDB Compass

Домашнее задание

Написать приложение, которое будет запрашивать RSS рассылку, парсить XML и сохранять документы в БД.

Пример: Граббер сайта - который запускаем из командной строки, он читает новости с sports.ru и сохраняет в DB

или твиты Трампа.

Object

Опрос

https://otus.ru/polls/1693/

Спасибо
за внимание!