Данные в веб-приложении Модель. БД.

Данные

Данные веб-приложения

«Действия» над данными

- Хранение
- Передача
- Обработка
- Отображение

Отображение данных

- HTML-документ
- Плагины браузера

Передача данных

- Протокол HTTP
  - HTML, JSON, XML, JPG,….
- WebSocket (поверх HTTP)
  - Произвольные данные
- Прикладные протоколы над TCP-IP, реализуемые плагинами

Хранение

- «Сервер»
  - Оперативная память (сервер приложений / кеш)
  - Файлы
  - База данных
- «Клиент»
  - Оперативная память
  - Файлы «Cookie»
  - «Базы данных» в браузере:
    localStorage, sessionStorage, webSQL, indexedDB

Базы данных

- База данных - взаимосвязанная информация об объектах, которая организованна специальным образом и хранится на каком-либо носителе.
- СУБД - совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

СУБД

PostgreSQL

Задачи

- Обеспечение хранения в всей необходимой информации.
- Обеспечение возможности получения данных
- Сокращение избыточности и дублирования данных.
- Обеспечение целостности данных

Понятия из мира баз данных

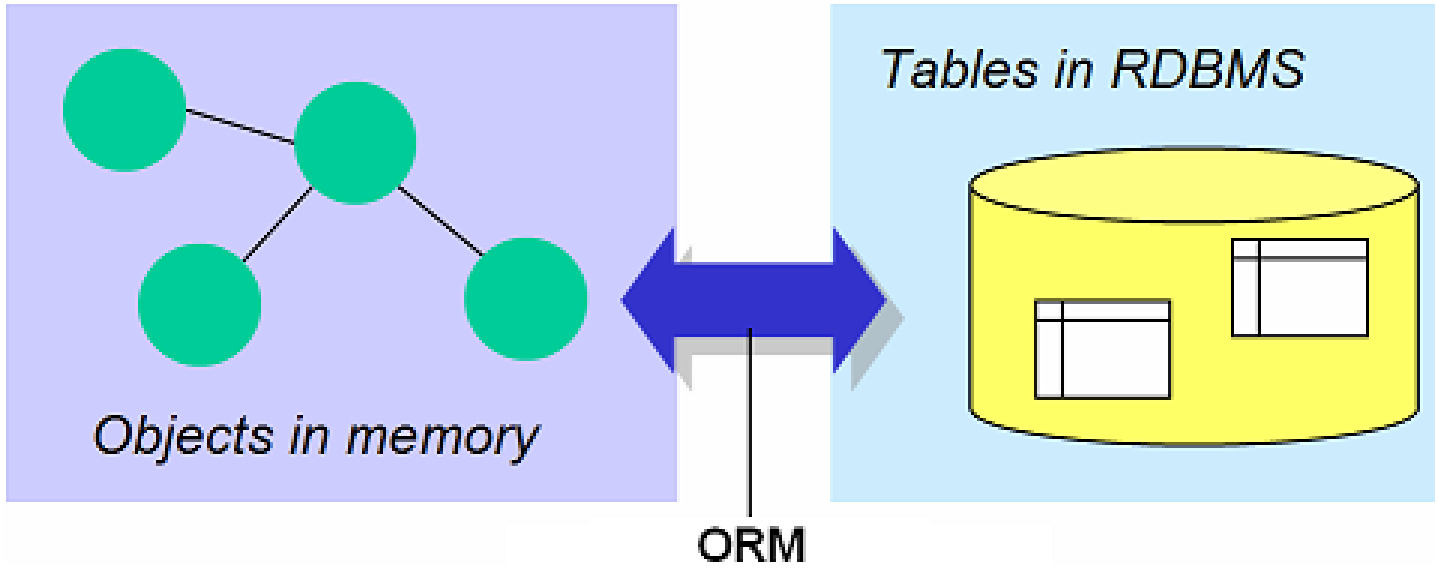
- Организация данных:
  - реляционная
  - документно-ориентированная
  - графовая
  - ключ-значение
- CRUD-операции
- Триггеры, хранимые процедуры
- ACID-транзакции
- Индексы
- Курсоры

СУБД

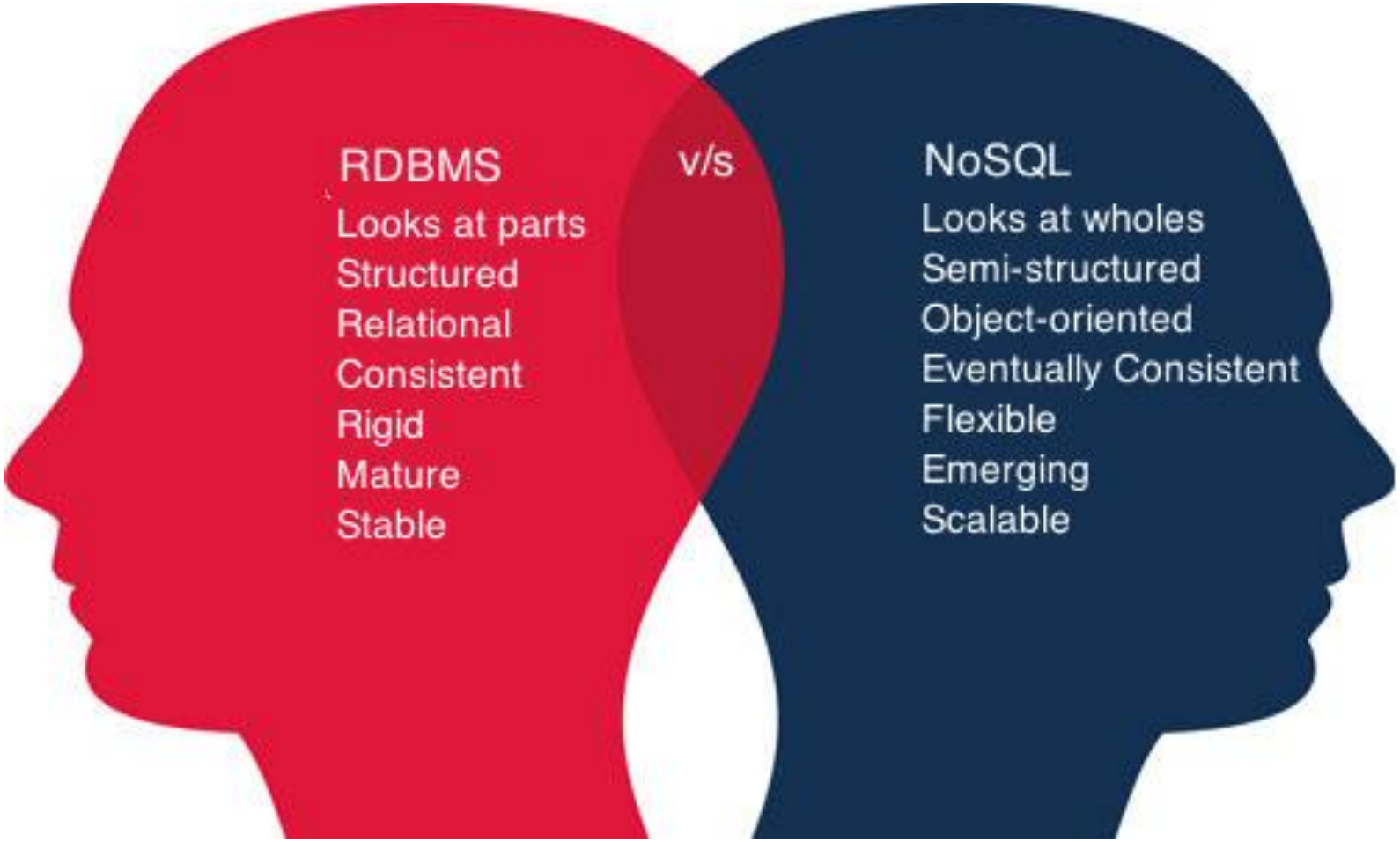
- Сервер БД
- Провайдер (драйвер) БД
- Журнал
- Пользователи и роли

Еще несколько понятий

- ORM
- Model First vs Code First
- Миграции



БД: SQL vs NoSQL



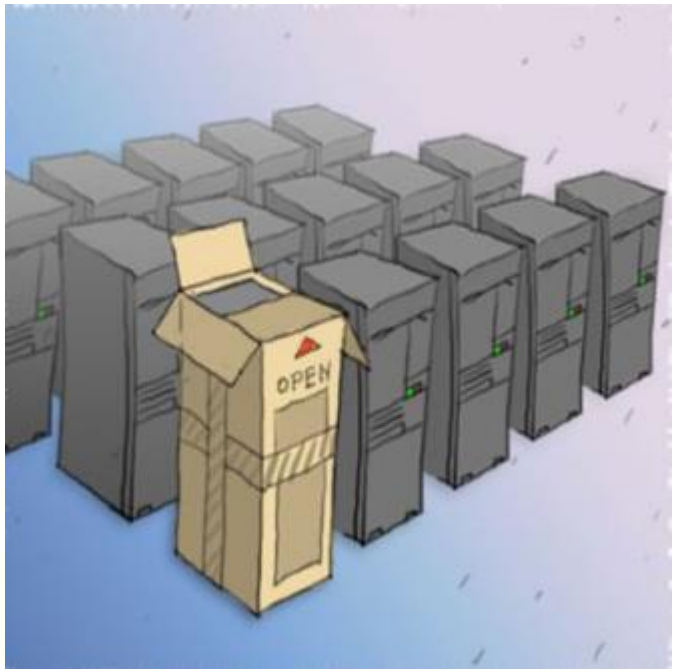
Реальные задачи веба

Чтение к записи – 10 к 1

Необходимость масштабирования

Масштабирование

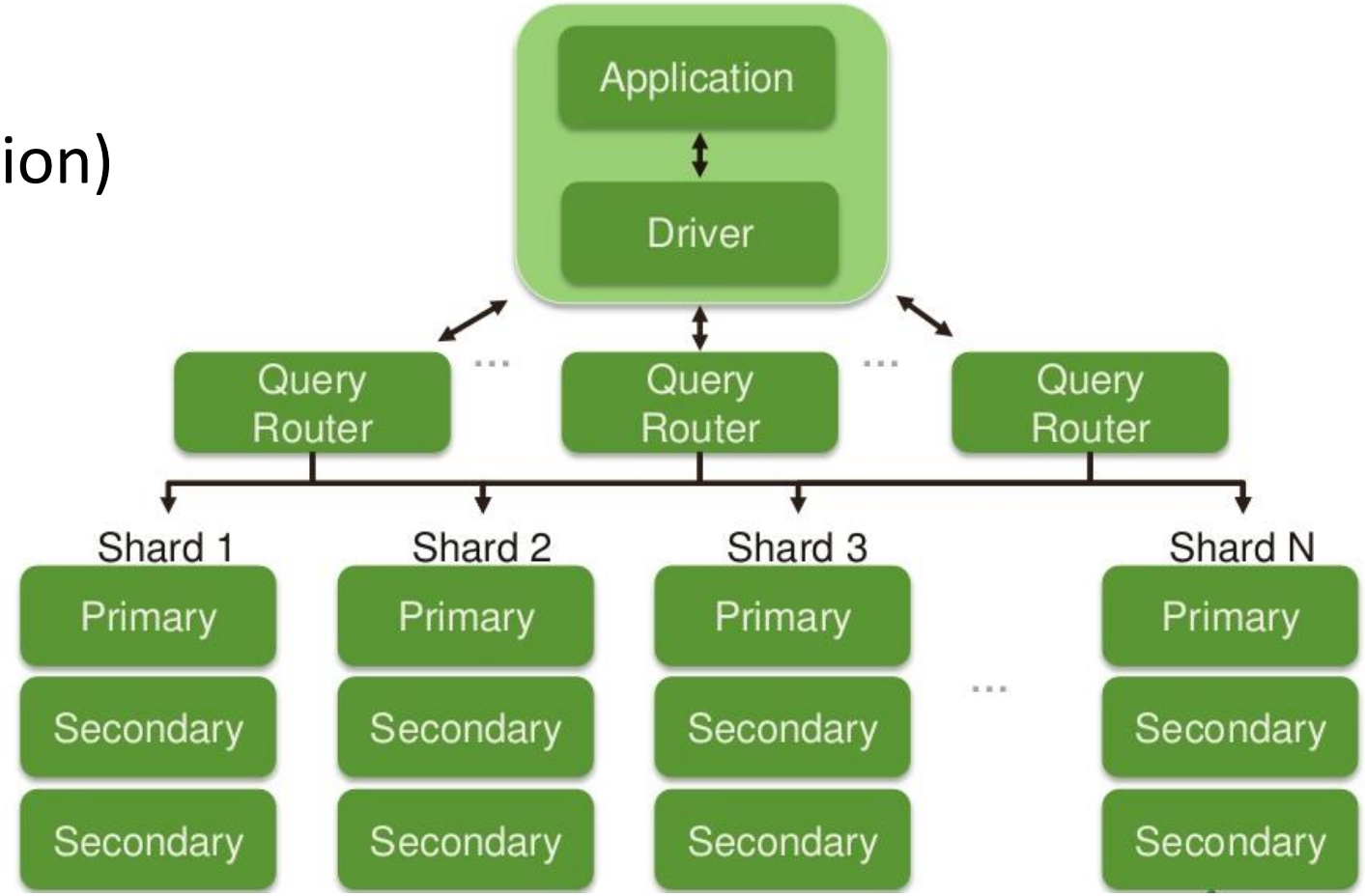
Вертикальное



Горизонтальное

Масштабирование

- Репликация (Replication)
- Шардинг (Sharding)

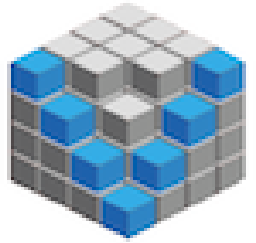


Проблемы SQL

- JOIN не масштабируем!
- Сложные проверки при выборке и вставке данных
- Семантический разрыв данных в бд и объектов в программе

Проблемы NoSQL

- Обеспечение целостности – на приложении
- Приходится обходиться без JOIN
- После SQL, кажется, что это НЕ ДОЛЖНО РАБОТАТЬ



Когда NoSQL?

- Данные сильно документно-ориентированы
- Главная задача— высокая масштабируемость по запросу.

Attention!

Technology first vs Architecture first

Чистый key-value

- Есть только ключ и значение
- Запрос – по ключу
- В лучшем случае – примитивная работа со значениями
- Использование – кеш/обмен сообщений (MemcachedDB, Redis)

Документные БД. MongoDB

- 2009
- Разрабатывается 10gen
- Родилась из облачного хранилища
- SQL-like Манипуляция данными
- Скорость и горизонтальная масштабируемость key-value
- JSON-Документы
- Командная оболочка на JS
- Репликация
- Сегментирование (шардинг)

Документная модель данных

- Документы JSON
- Объединены в коллекции

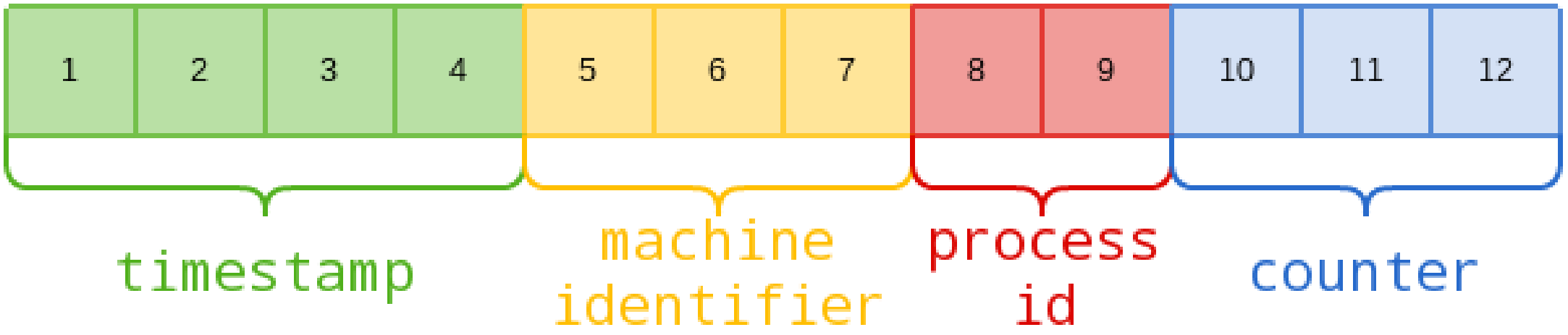
Пример документов

```
db.users.save({name:"Ivan", secondname:"Ivanovich",
surname:"Ivanov", login:"ivanov",
password:"qwerty", birth_date:"1975-08-24"});

db.users.save({name:"Petr", surname:"Petrov",
login:"petrov", password:"qwerty",
birth_date:"1972-09-11"});

db.users.save({name:"Aleksey", surname:"Alekseev",
login:"alex135", password:"qwerty", city:"Saint-
Petersburg"});
```

_id

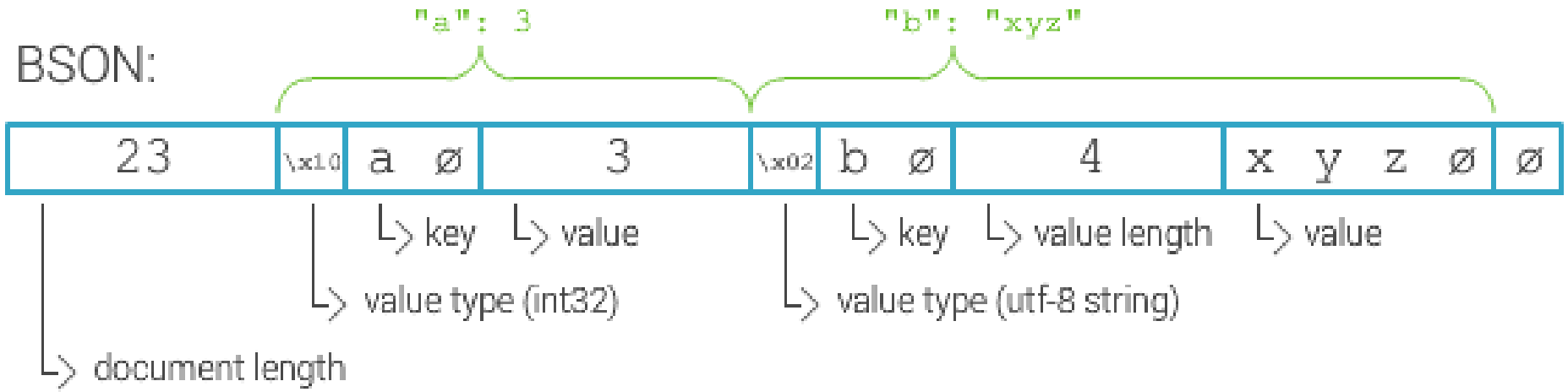

1
2
3
4
5
6
7
8
9
10
11
12
timestamp
machine identifier
process id
counter

BSON

JSON:

```
{
   "a": 3,
   "b": "xyz"
}
```

BSON:



Запросы

```
db.users.find({"login":"ivanov"});
db.users.find({"birth_date":{"$gt":"1969-06-06"}});
db.users.find({ $query : { "birth_date" : {"$lt":"1999-09-09", "$gt":"1936-05-02"}}, $orderby : { "login" : 1 } } );
```

Обновление данных

```
db.users.update({"login":"ivanov"},{$set:{"login":
"sidorov"}});
```

- $set
- $unset
- $addToSet

Удаление данных

```
db.users.remove({"login":"ivanov"});
```

- `db.users.drop()`

Индексы

```
for(var i=0; i<200000; i++){
     db.numbers.save({num:i});
}
db.numbers.find({num:{“$gt”:199995}}).explain()
db.numbers.ensureIndex({num:1})
db.numbers.getIndexes()
```

Проектирование модели данных

- Отказ от встраивания (имитация реляционности)
- Полное встраивание (дублирование данных)
- Тонкая балансировка ☺

Беда монги

«Есть только две трудные задачи в области информатики: инвалидация кеша и придумывание названий.»

Фил Карлтон

Монга – это КЕШ.

Как использовать в NodeJS?

- Через провайдер
- Через ORM (Mongoose)

mongoose

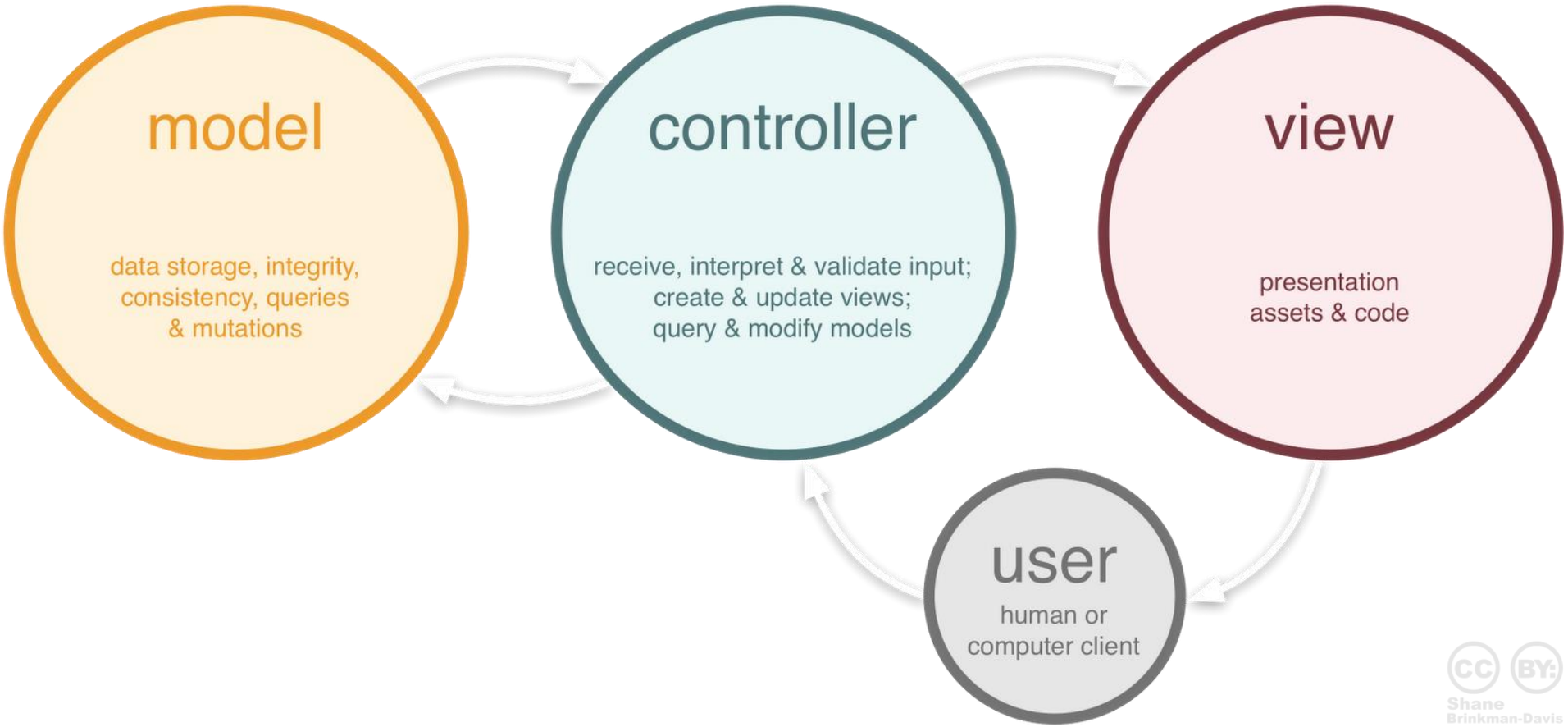
elegant mongodb object modeling for node.js

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

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

var kitty = new Cat({ name: 'Zildjian' });
kitty.save(function (err) {
  if (err) // ...
  console.log('meow');
});
```

MVC


model
data storage, integrity, consistency, queries & mutations
controller
receive, interpret & validate input; create & update views; query & modify models
view
presentation assets & code
user
human or computer client
CC BY:
Shane Brinkman-Davis

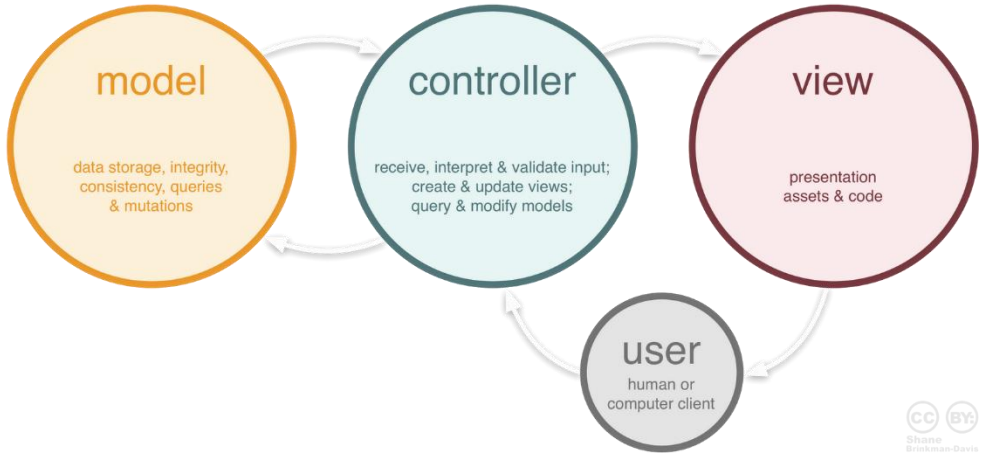
«Тощий контроллер, толстая модель»

FSUC: Fat Stupid Ugly Controllers

Controller

Обработка запросов пользователя

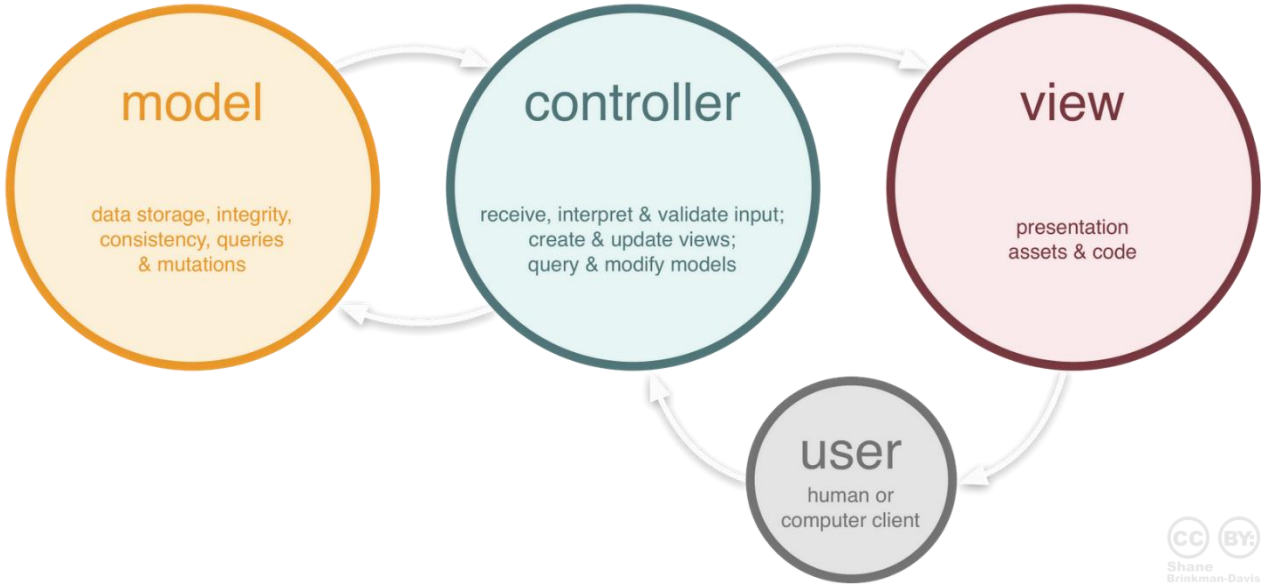
- Маршрутизация (Routing)
- Получение http-запросов
- Обращение к модели (бизнес логике) для получения данных
- Генерация представления (передача данных шаблонизатору)
- Отправка http-ответов



View

Представление данных

- Тонкий клиент: шаблон на языке шаблонизатора (например, jade, ejs,…)
- Толстый клиент



Model

- Данные приложения
- *Бизнес логика (манипулирование данными, ограничения и правила их использования)*