拉勾Offer工场

主讲人：ccmouse

2017-09-24

1

CHAPTER1

Leetcode真题解析

2

CHAPTER2

设计模式

3

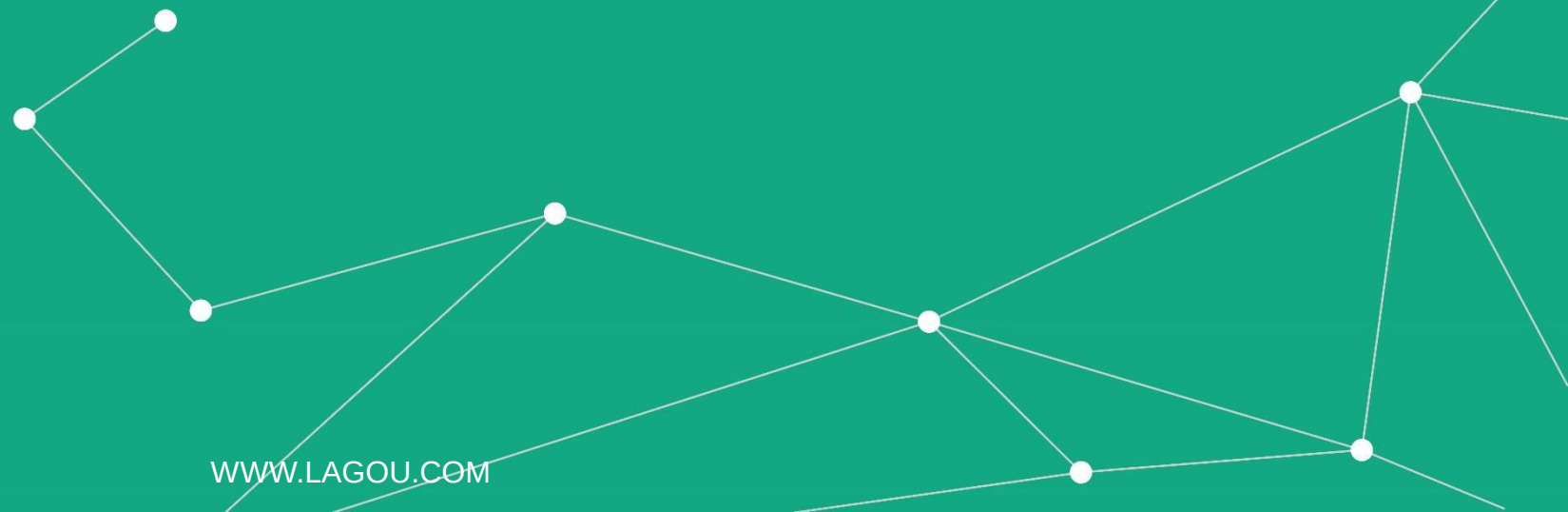
CHAPTER3

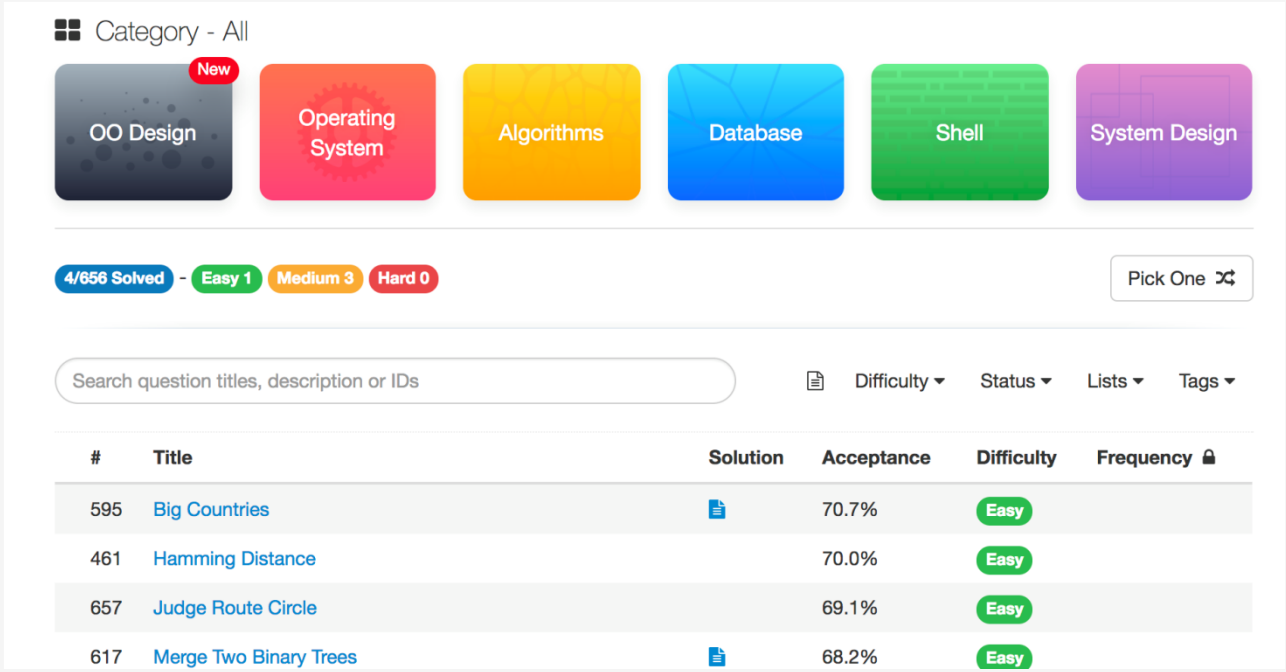
开放式问答

CHAPTER1

Leetcode真题解析

Talk is cheap. Show me the code.




Category - All
New
OO Design
Operating System
Algorithms
Database
Shell
System Design
4/656 Solved - Easy 1 Medium 3 Hard 0
Pick One
Search question titles, description or IDs
Difficulty
Status
Lists
Tags
Title Solution Acceptance Difficulty Frequency
595 Big Countries 70.7% Easy
461 Hamming Distance 70.0% Easy
657 Judge Route Circle 69.1% Easy
617 Merge Two Binary Trees 68.2% Easy

真题选讲

脑筋急转弯类

- 不用临时变量，交换两个变量的值
- 一个数组中含有1到n每个数各一遍，但是唯独缺了其中一个，把它找出来
- 一个数组中几乎所有的数都出现两遍，只有一个数出现一遍，把它找出来

脑筋急转弯类

- https://leetcode.com/problems/linked-list-cycle/description/
- https://leetcode.com/problems/search-a-2d-matrix-ii/description/

字符串，数组操作类

- https://leetcode.com/problems/reverse-words-in-a-string-iii/description/
- https://leetcode.com/problems/move-zeroes/description/
- https://leetcode.com/problems/longest-substring-without-repeating-characters/description/

计算机基础相关

- https://leetcode.com/problems/excel-sheet-column-title/description/
- https://leetcode.com/problems/regular-expression-matching/description/
- https://leetcode.com/problems/evaluate-reverse-polish-notation/description/

生成数据

- 生成排列，生成组合，生成下一排列，生成下一组合
- https://leetcode.com/problems/shuffle-an-array/description/
- https://leetcode.com/problems/generate-parentheses/description/

上下界类

- https://leetcode.com/problems/best-time-to-buy-and-sell-stock-ii/description/
- https://leetcode.com/problems/trapping-rain-water/description/

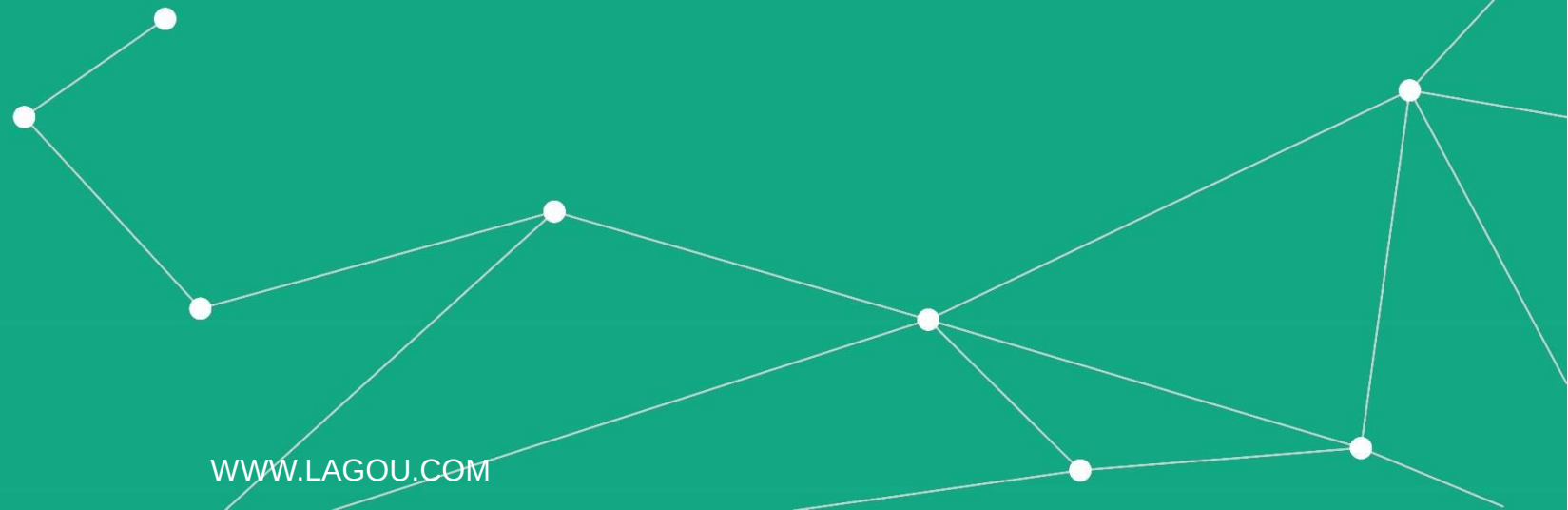
其它

- https://leetcode.com/problems/longest-increasing-path-in-a-matrix/description/
- https://leetcode.com/problems/lru-cache/description/

CHAPTER2

设计模式

变继承为组合



再谈Singleton模式

Singleton优缺点

- 确保全局至多只有一个对象
- 用于：构造缓慢的对象，需要统一管理的资源
- 缺点：很多全局状态，线程安全性

再谈Singleton模式

Singleton的创建

- 双重锁模式 Double checked locking
- 作为Java类的静态变量
- 使用框架提供的能力

变继承关系为组合关系

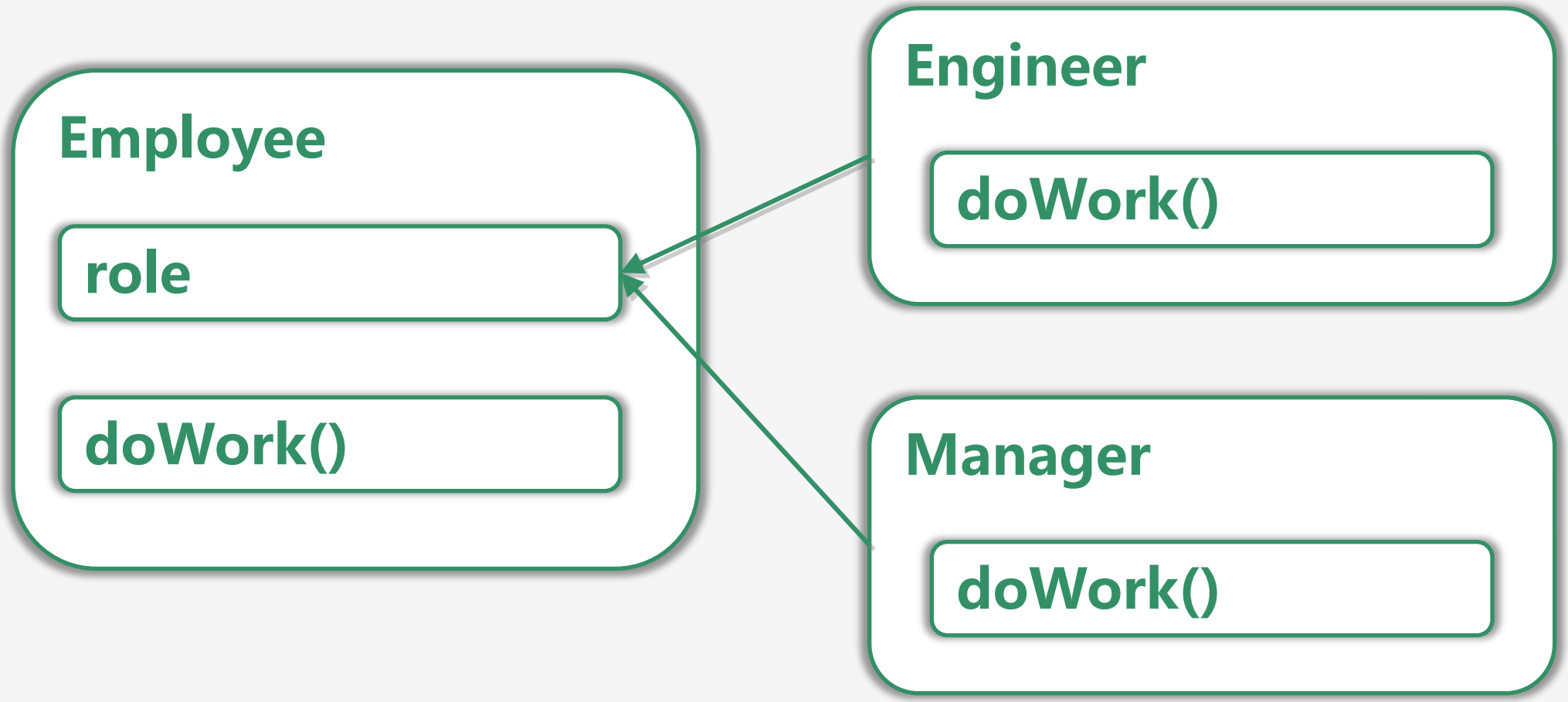
继承关系

- 描述is-a关系
- 不要用继承关系来实现复用
- 使用设计模式来实现复用

变继承关系为组合关系

如果Employee 升级成了 Manager?

State Pattern

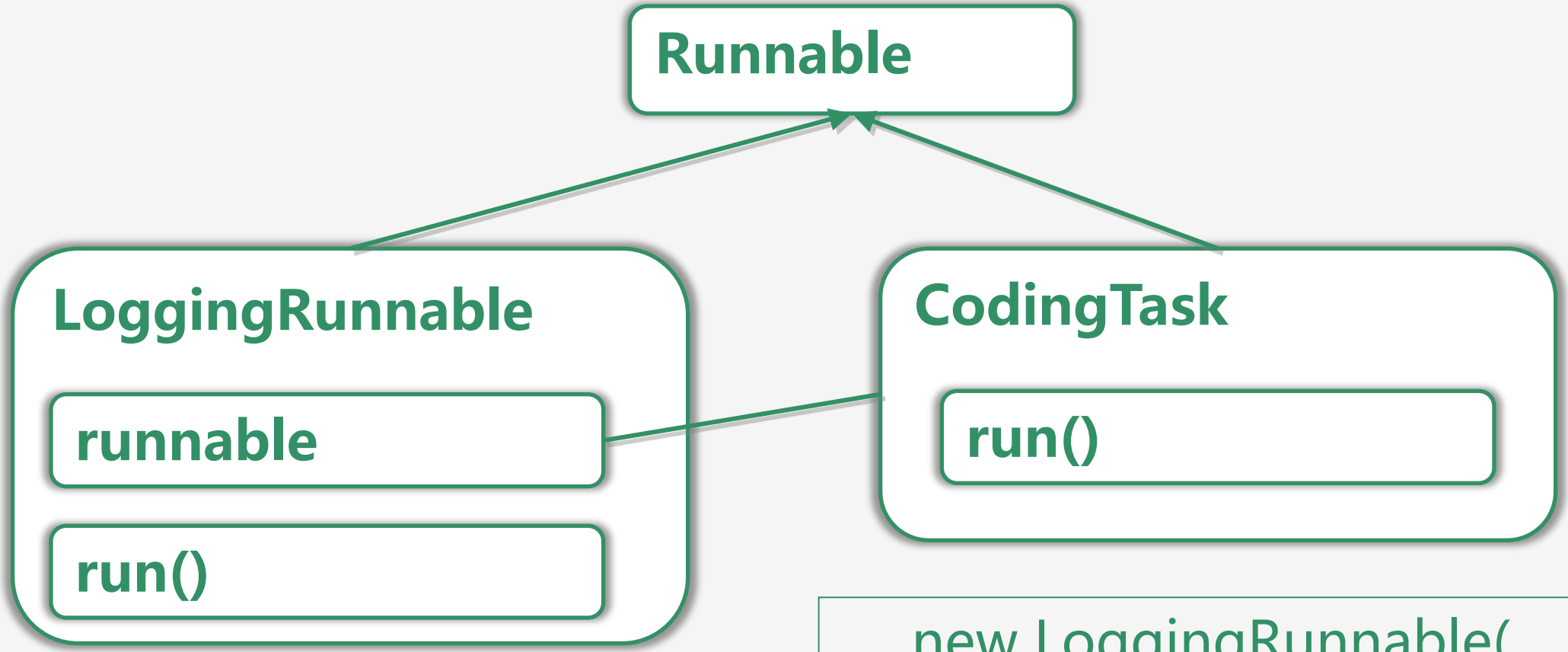


变继承关系为组合关系

```
interface Runnable {
    void run();
}
```

如何实现LoggingRunnable, TransactionalRunnable, ……

Decorator Pattern




```
new LoggingRunnable(
new CodingTask())
.run()
```

如何创建对象

使用 new 来创建的缺点

- 编译时必须决定创建哪个类的对象
- 参数意义不明确

Abstract Factory Pattern

比较

- task = new LoggingTask(new CodingTask());
- task = taskFactory.createCodingTask();

Builder Pattern

比较

- employee = new Employee(
  oldEmployee.getName(), 15000);

- employee = Employee.fromExisting(oldEmployee)
  .withSalary(15000)
  .build();

- 不可变对象往往配合Builder使用

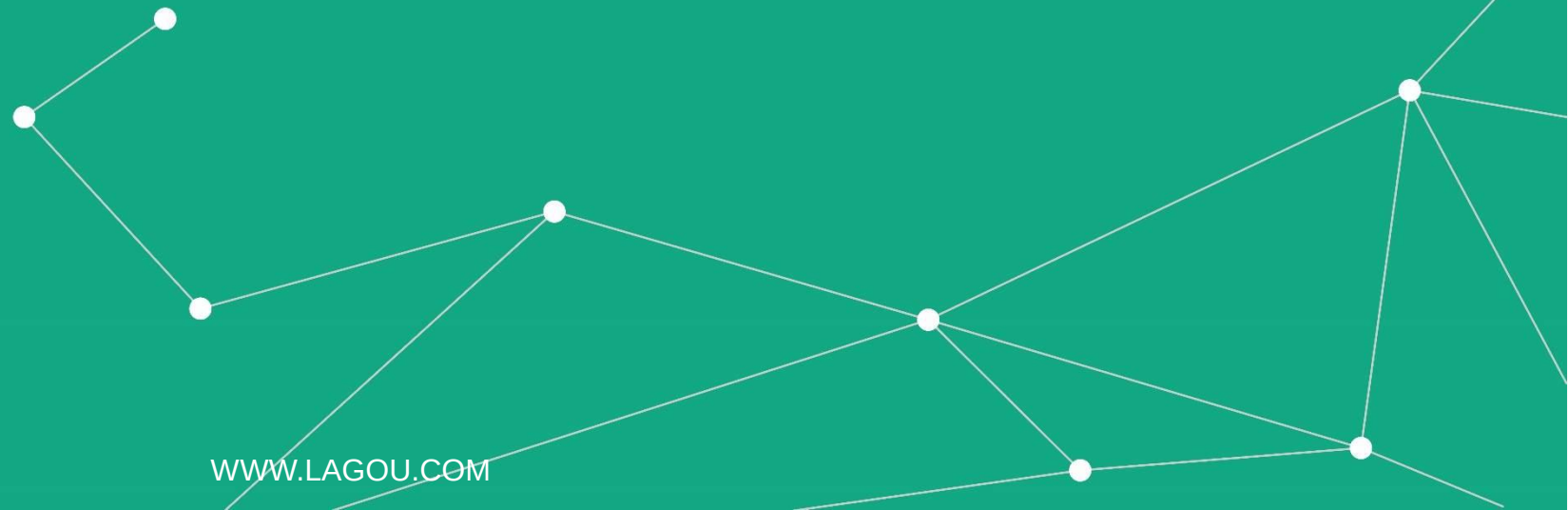
函数式编程　vs　函数指针 ？

python中的decorator

CHAPTER3

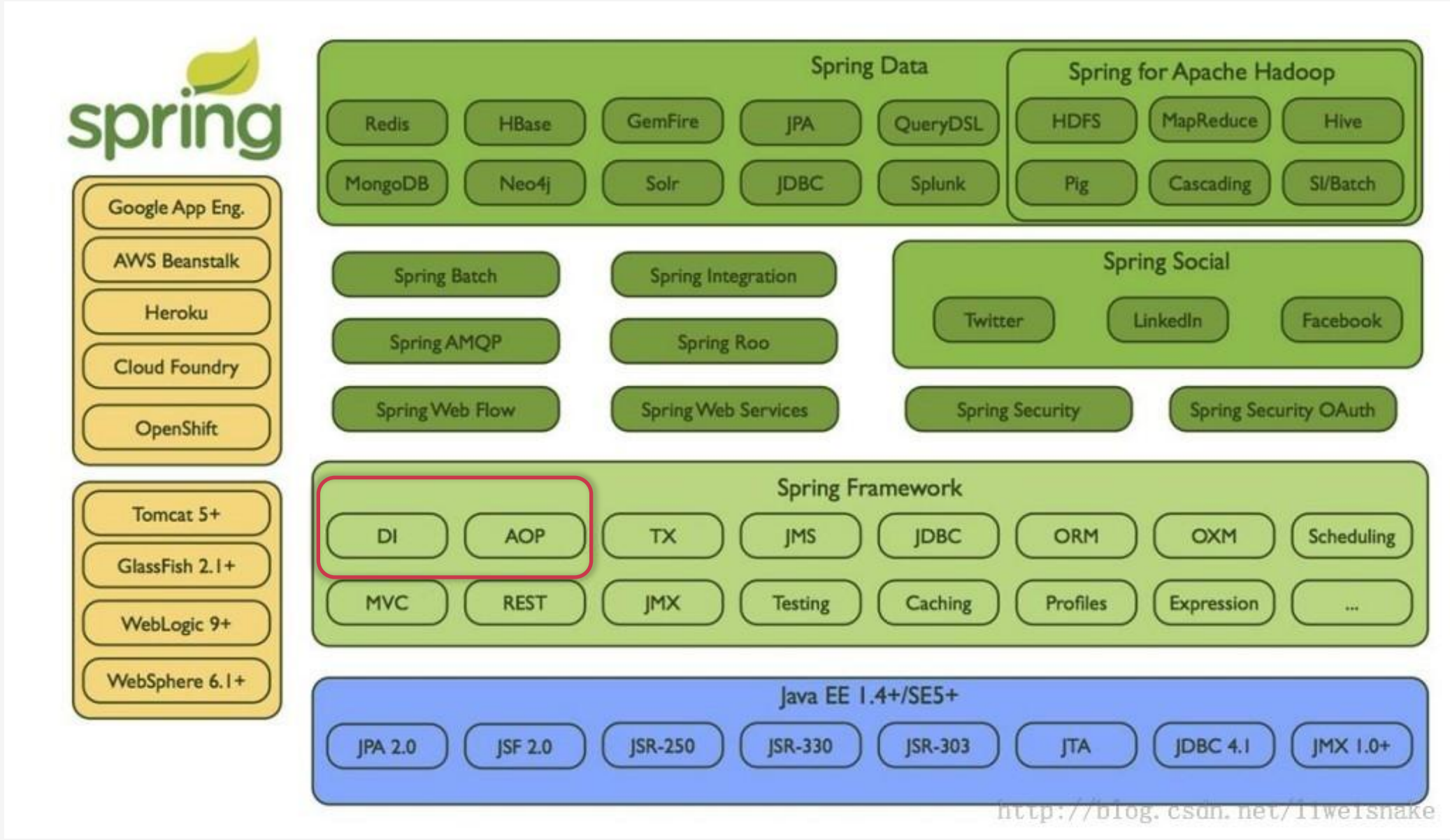
开放式问答

当我们聊简历时我们在聊什么



介绍一下xx框架？

- 尽量简单，结构化，不要过多修饰
- 与竞品的比较
- 你为何选择这个框架



NoSql

- Key-value: Memcached, Redis
- Document: CouchDB, MongoDB
- Column: Cassandra, HBase

NoSql技术的比较

- 功能：数据结构，Transaction支持，接口，持久性，Pub/Sub
- 性能
- Scalability：Replication & Failover
- https://kkovacs.eu/cassandra-vs-mongodb-vs-couchdb-vs-redis

并行计算

- HDFS
- Hadoop
- Spark

容器

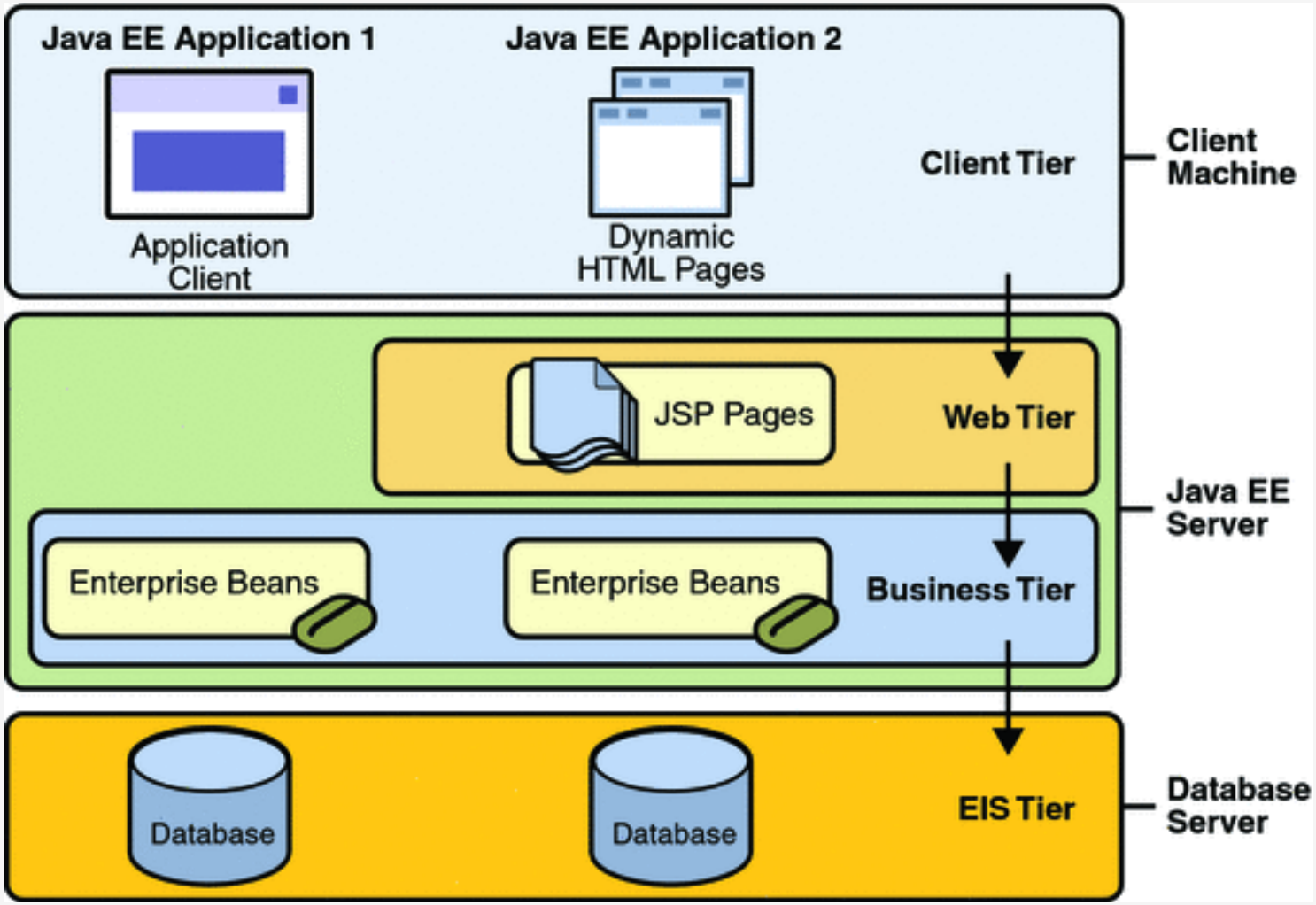
- Docker
- Kubernetes

前端技术

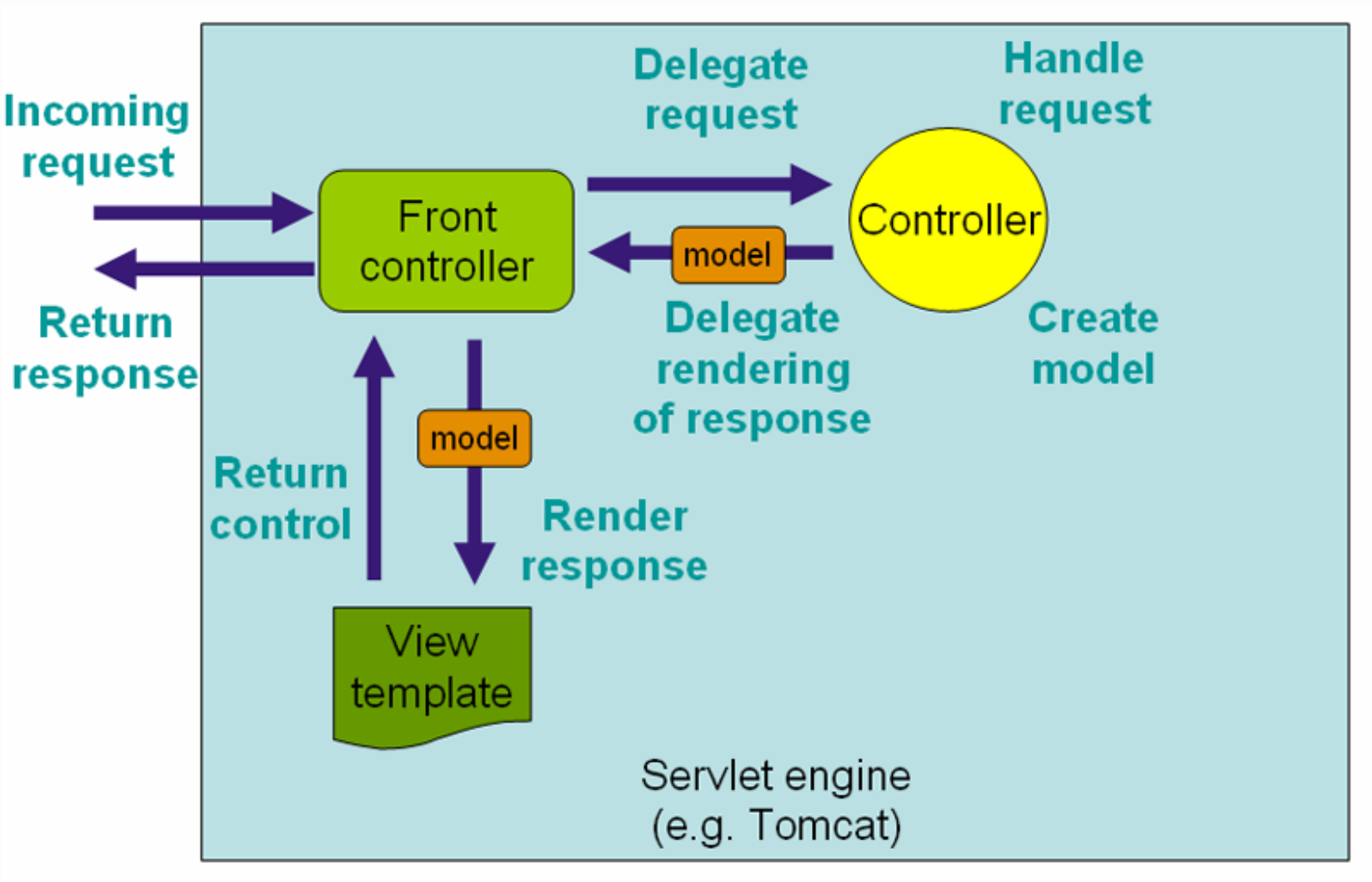
- React.js
- AngularJs
- ajax
- Node.js

服务器架构的演进

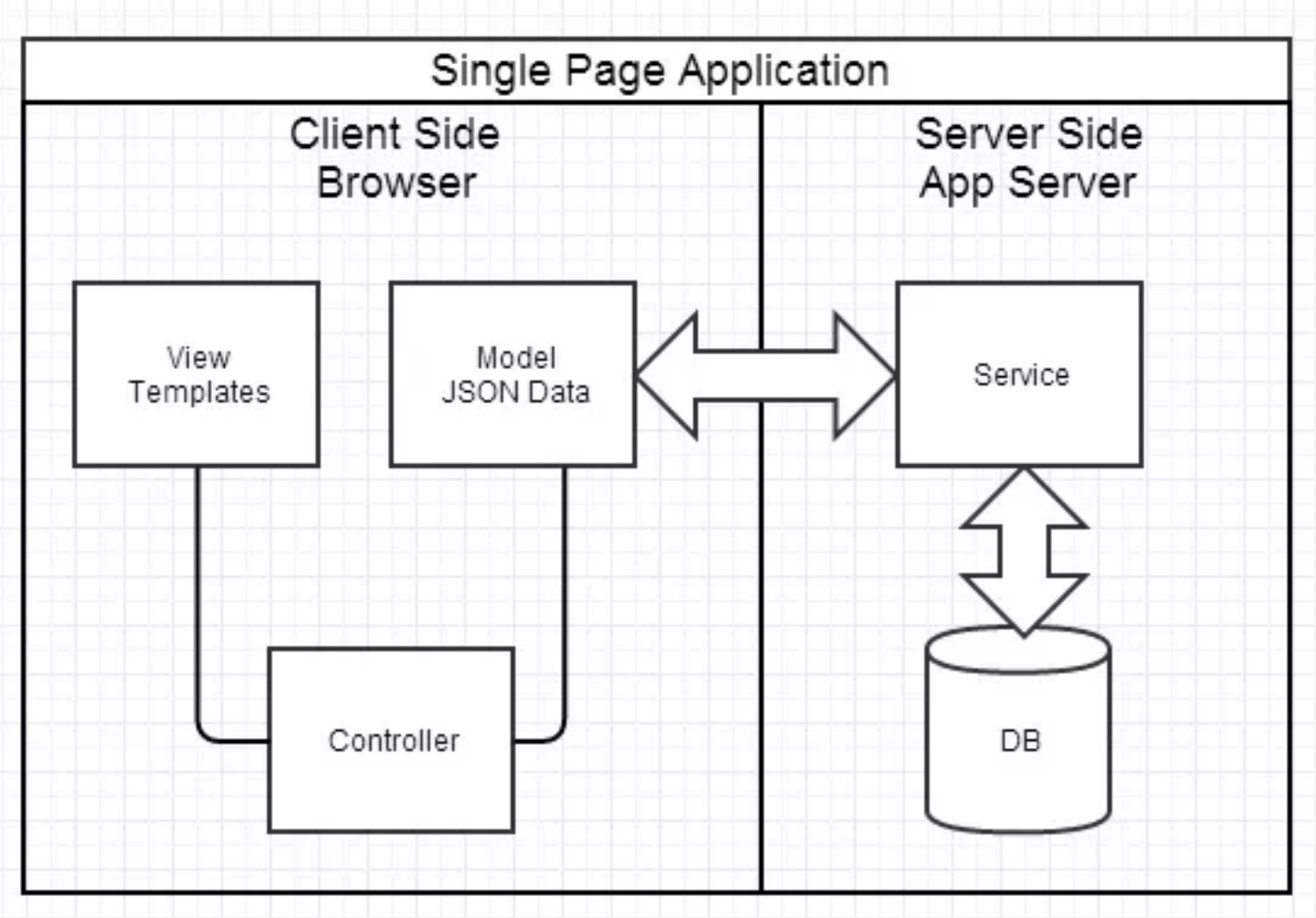
三层架构


Java EE Application 1
Java EE Application 2
Application Client
Dynamic HTML Pages
Client Tier
Client Machine
JSP Pages
Web Tier
Java EE Server
Enterprise Beans
Enterprise Beans
Business Tier
Database
Database
EIS Tier
Database Server

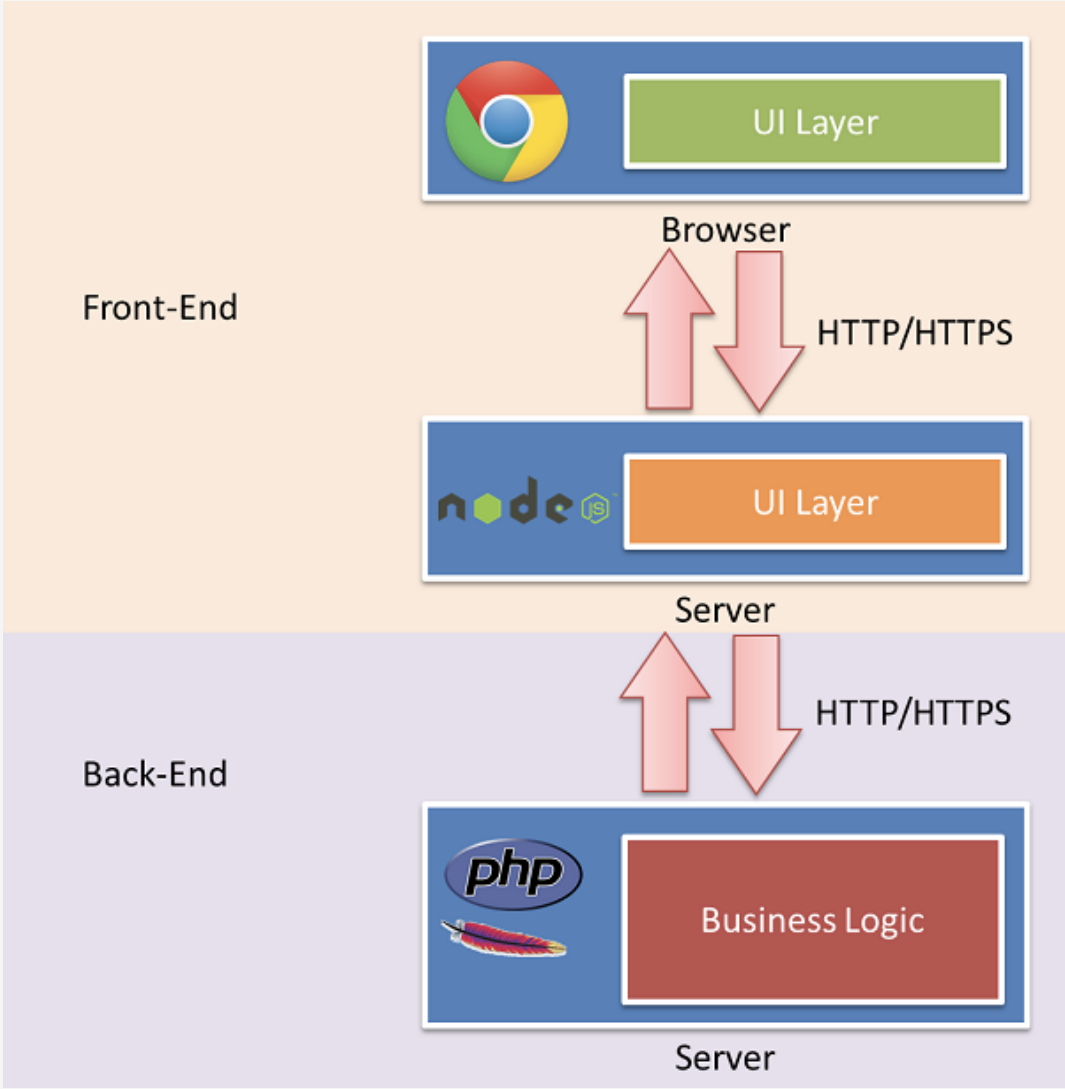
MVC


Incoming request
Delegate request
Handle request
Front controller
Controller
model
Return response
Delegate rendering of response
Create model
model
Return control
Render response
View template
Servlet engine (e.g. Tomcat)

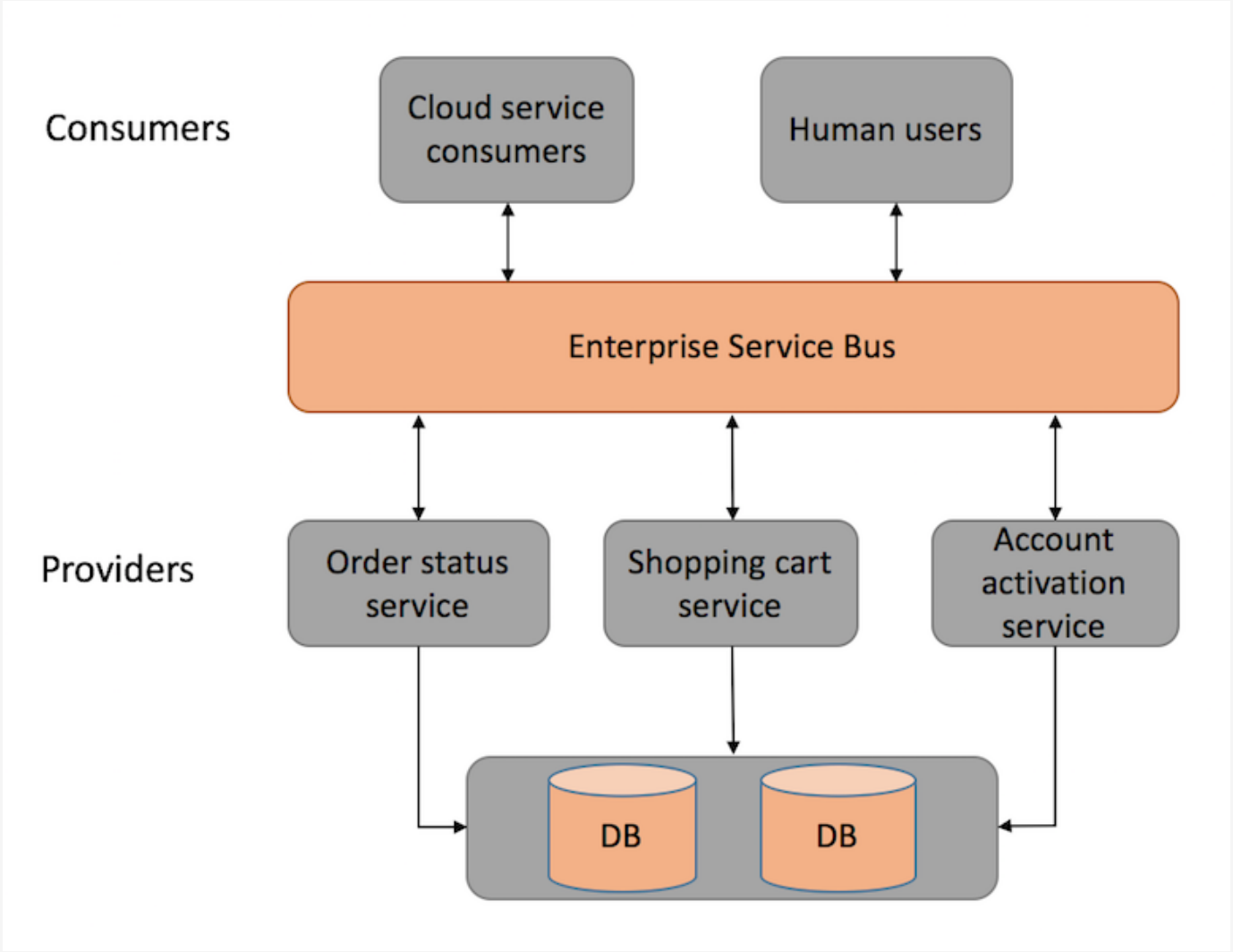
基于Ajax的前后端分离



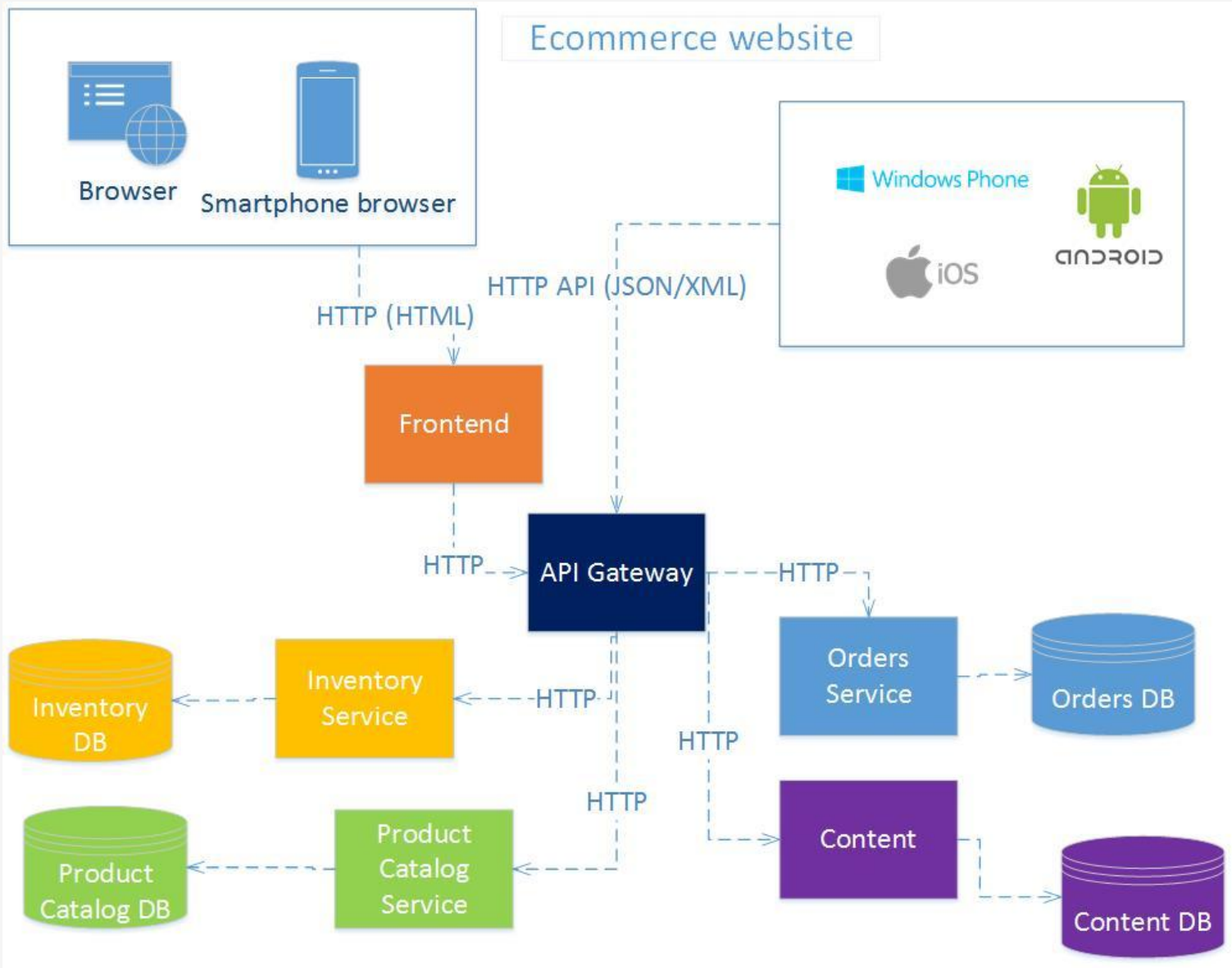
基于Node.js的前后端分离



SOA

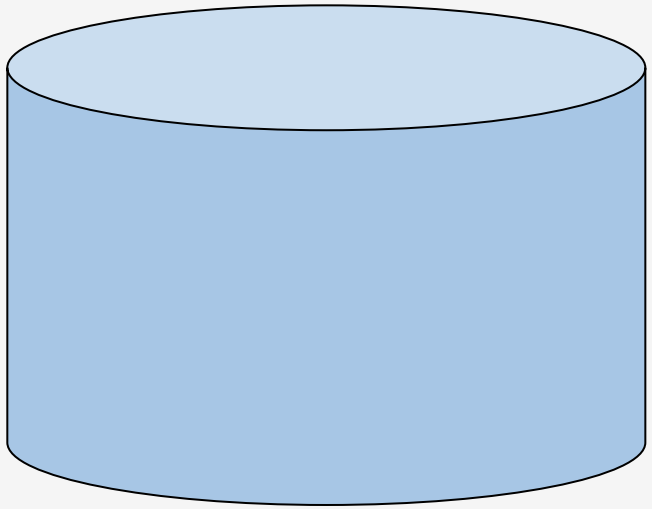

Consumers
Cloud service consumers
Human users
Enterprise Service Bus
Providers
Order status service
Shopping cart service
Account activation service
DB
DB

微服务

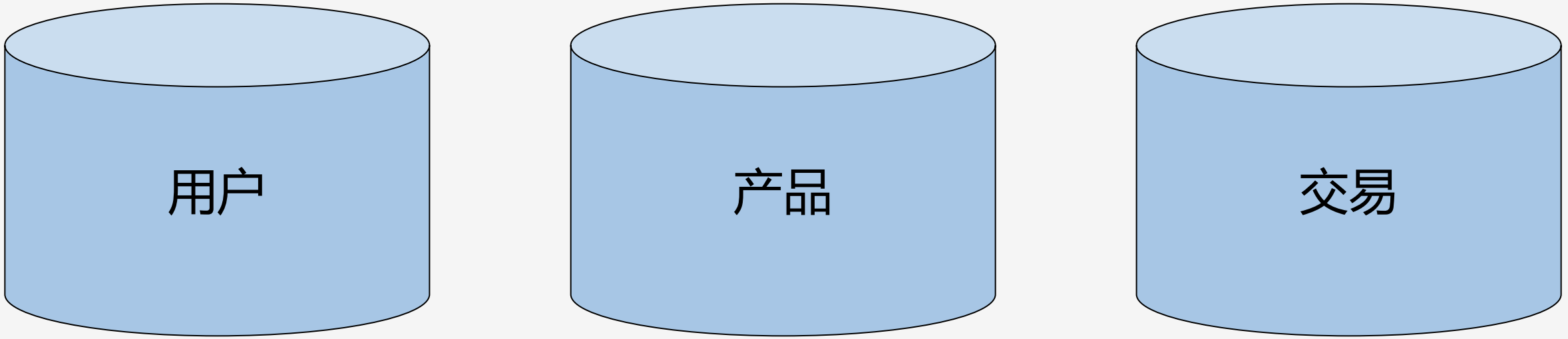

Ecommerce website
Browser
Smartphone browser
Windows Phone
iOS
android
HTTP (HTML)
HTTP API (JSON/XML)
Frontend
HTTP
API Gateway
HTTP
Orders Service
Orders DB
Inventory DB
Inventory Service
HTTP
HTTP
HTTP
Content
Content DB
Product Catalog DB
Product Catalog Service

数据访问架构的演进

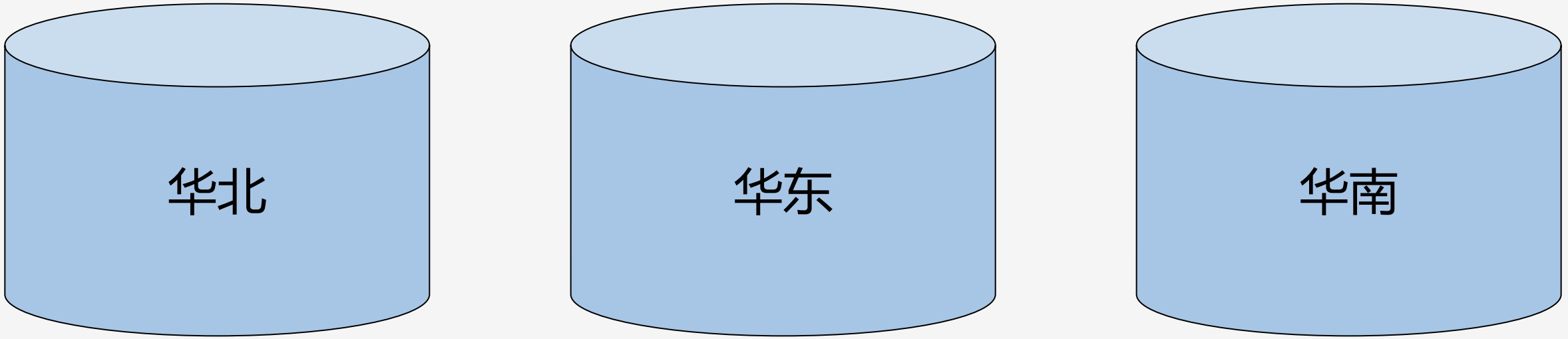
单个数据库



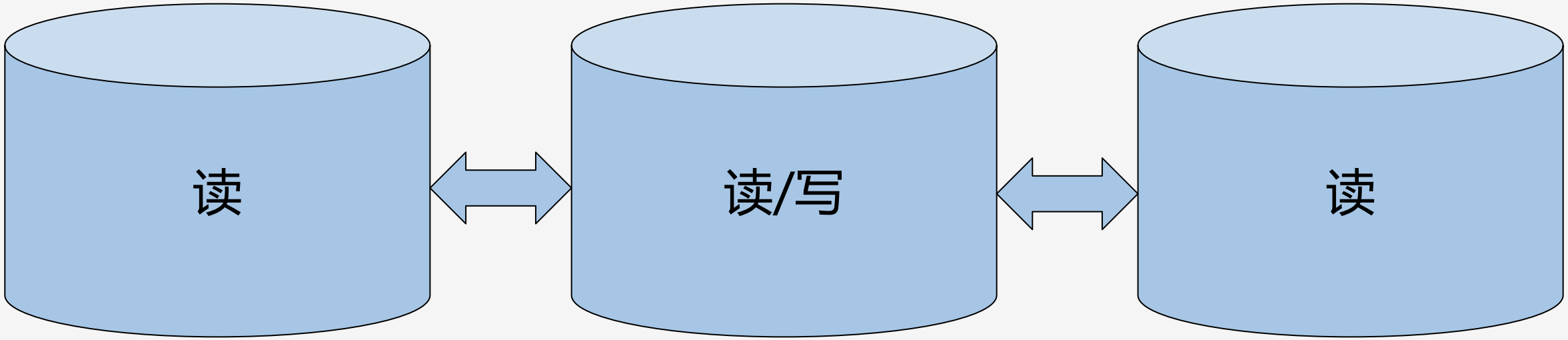
根据领域拆分数据库



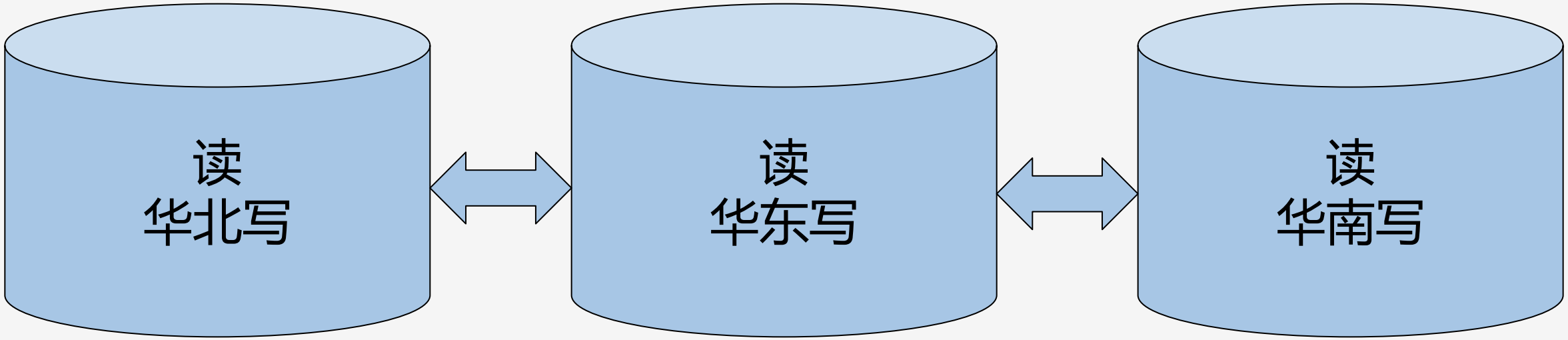
根据分区拆分数据库



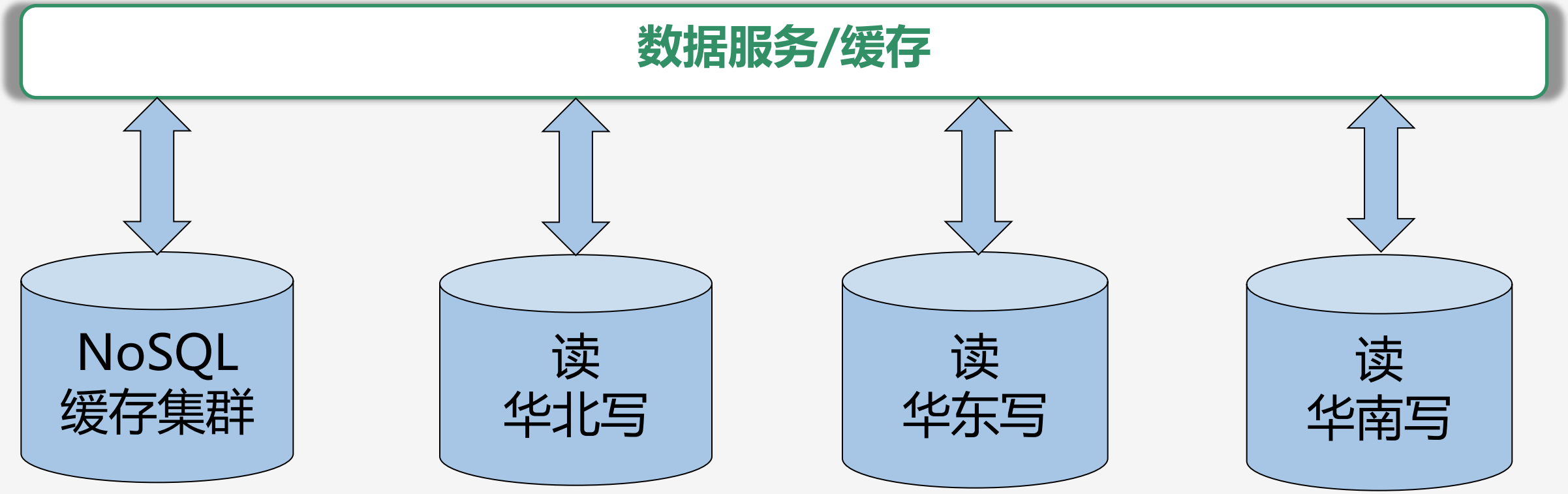
数据库读写分离



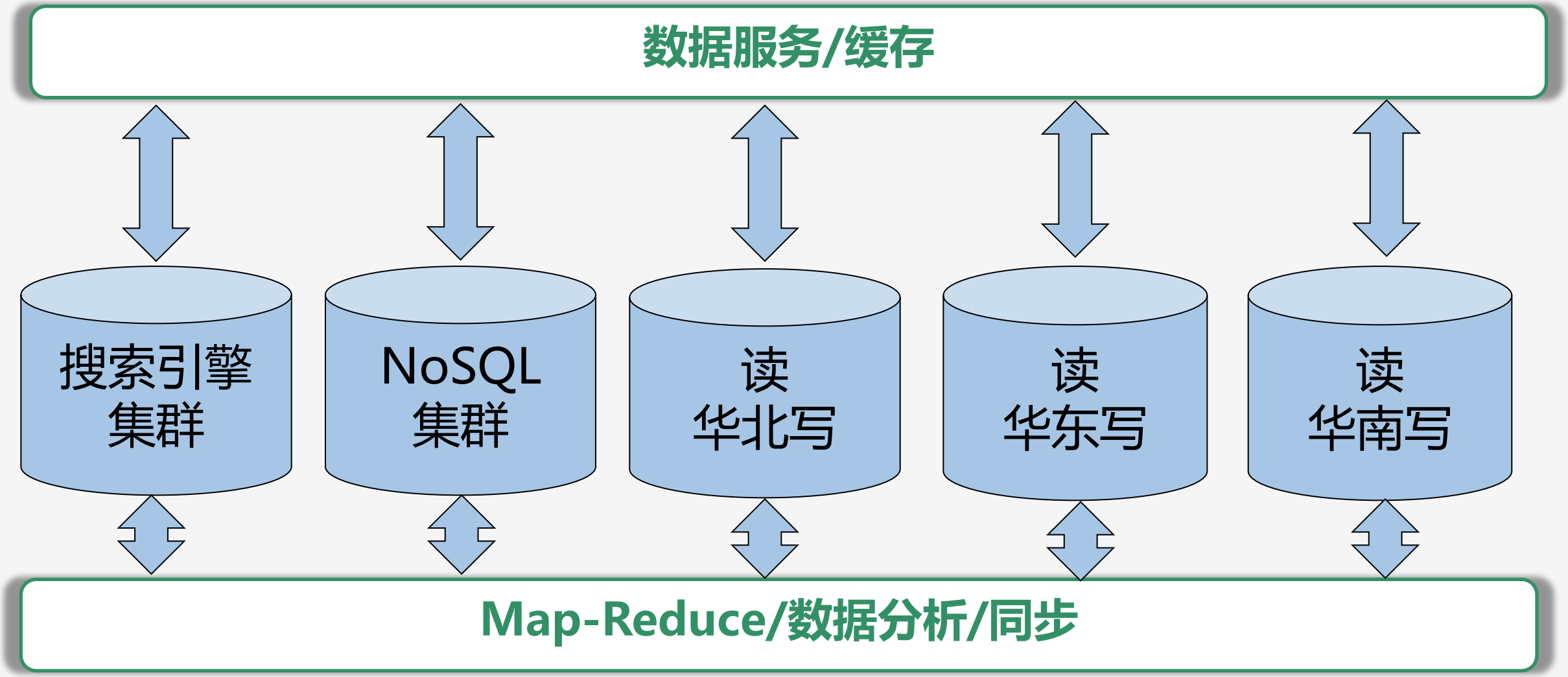
数据库读写分离+区域



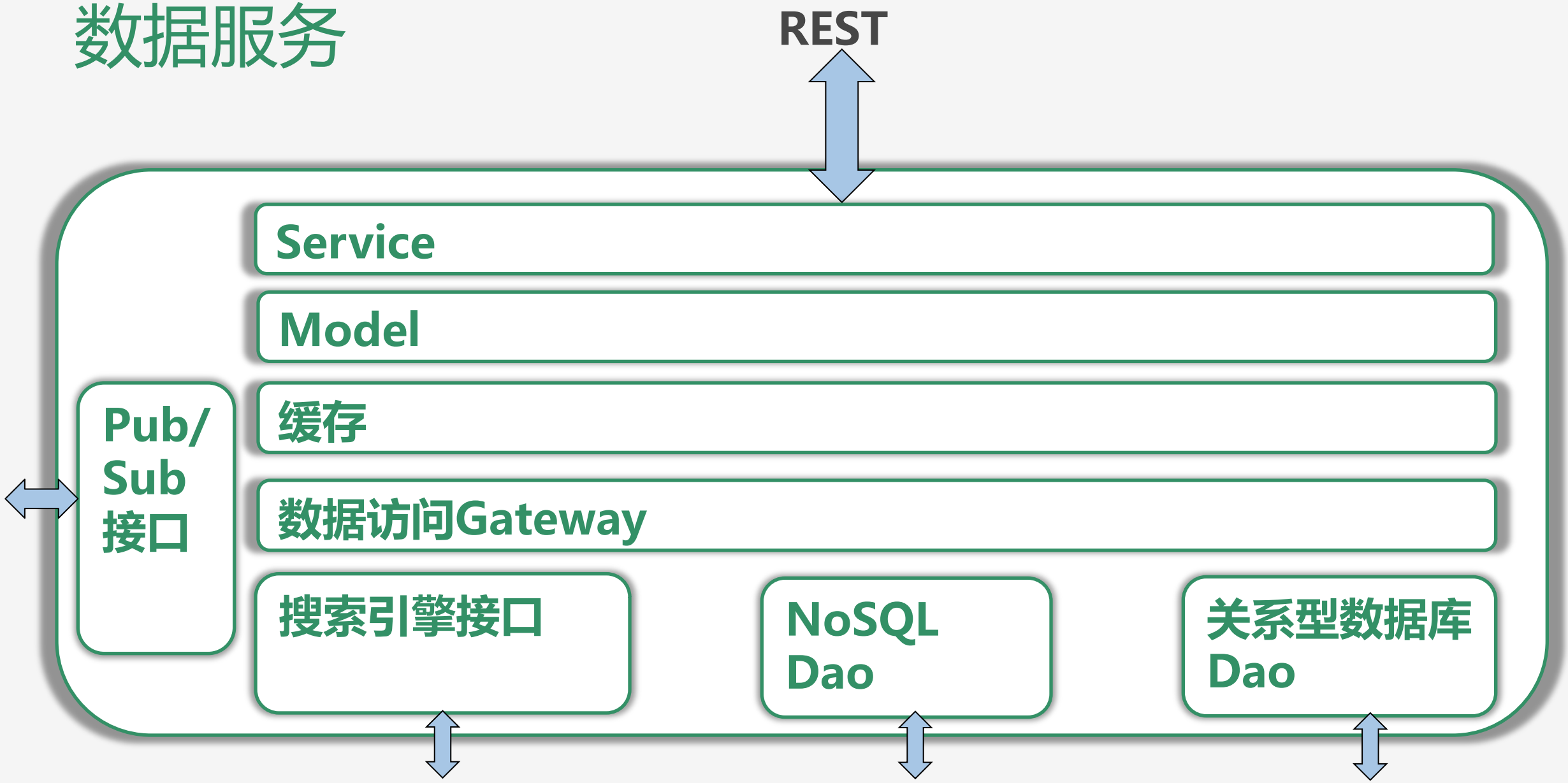
加入缓存/数据服务器



添加搜索引擎



数据服务


REST
Service
Model
缓存
数据访问Gateway
Pub/
Sub
接口
搜索引擎接口
NoSQL
Dao
关系型数据库
Dao

并行计算之Map-Reduce

Map-Reduce过程

- 将数据分块，并行的发给每个Mapper（系统）
- Map: Mapper对每个记录产生一个或多个Key-Value Pair（用户）
- Shuffle: 讲数据根据Key组装起来，形成Key -> Value List（系统）
- Reduce: 每个Key的Value List分发给Reducer进行计算（用户）
- 将每个Key对应的Reduce结果输出（系统）

例：用户视频质量统计

- 输入：videoId -> videoDetail (包括userId)
- 输出：userId -> 该用户视频质量因子

例：用户视频质量统计

- Map: videoId -> videoDetail ➔ userId -> videoDetail
- Reduce: userId -> videoDetailList ➔ userId -> score

同学加油

添加拉勾课程咨询老师薇薇微信，获得更多课程信息；

关注互联网offer之路，获取海量互联网求职干货。

Thank You