1.1.0.数据

1.1.1.0.数据概述

1.1.1.1.0.数据概述

- 1.1.1.1.1.0.数据概述
- 1.1.1.1.2.0.数据分类
- 1.1.1.1.3.0.数据一致性、数据一致性模型、分布式事务、数据一致性算法

1.1.1.2.0.数据（结构化）存储设计

概念	关系模型	文档模型	键值模型	列宽模型	图模型
数据单元	表（Table）/ 行（Row）	集合（Collection）/ 文档（Document）	键（Key）/ 值（Value）	表（Table）/ 行（Row）	顶点（Vertex）/ 边（Edge）
结构	固定模式（Schema）	灵活模式（Schema-less）	无模式	灵活模式（Schema-less）	灵活模式（Schema-less）
关系	通过外键（Foreign Key）表示关系	通过内嵌（Embedding）或引用（Reference）表示关系	无原生关系支持	通过共同的行键（Row Key）表示关系	顶点（Vertex）和边（Edge）天生表示关系

1.1.1.3.0.缓存

- 1.1.1.3.1.0.缓存概述
- 1.1.1.3.2.0.缓存设计
- 1.1.1.3.3.0.缓存实现

2.1.0.设计模式(Java开发)

2.1.1.0.单例模式

- 2.1.1.1.0.单例模式

- 饿汉式：线程安全，调用效率高，但是不能延时加载。
- 懒汉式：线程不安全，在多线程不能正常工作，调用效率不高，但是可以延时加载。
- 静态内部类：线程安全，调用效率高，但是可以延时加载。
- 枚举：线程安全，调用效率高，不能延时加载，可以防止反序列化创建新的对象。

2.1.2.0.建造者模式

- 2.1.2.1.0.建造者模式

- 建造者模式（Builder Pattern）是一种对象构建的设计模式，它将复杂对象的构建过程与其表示分离，使得同样的构建过程可以创建不同的表示。

2.1.3.0.工厂模式

- 2.1.3.1.0.工厂模式

- 简单工厂：一个工厂类根据传入的参数决定创建哪一种产品类的实例。
- 工厂方法：定义一个用于创建对象的接口，让子类决定实例化哪一个类。工厂方法使一个产品类的实例化延迟到其工厂的子类。
- 抽象工厂：提供一个创建一系列相关或相互依赖对象的接口，而无需指定它们具体的类。

2.1.4.0.责任链模式

- 2.1.4.1.0.责任链模式

- 责任链模式（Chain of Responsibility Pattern）是一种行为设计模式，允许将请求沿着处理者链进行发送。收到请求后，每个处理者均可对请求进行处理，或将其传递给链上的下个处理者。

2.1.5.0.观察者模式

- 2.1.5.1.0.观察者模式

- 观察者模式（Observer Pattern）是一种行为设计模式，允许定义一种订阅机制，可在顾客对象事件发生时通知多个“观察”该对象的其他对象。

2.1.6.0.迭代器模式

- 2.1.6.1.0.迭代器模式

- 迭代器模式（Iterator Pattern）是一种行为设计模式，让程序能将遍历操作移到另一个对象上来。

3.1.0.微信：1.0.0.0.一个U盘

3.1.1.0.微信的架构

- 客户端（Client）：
- 接入层（Gateway）：
- 逻辑层（Logic Service）：
- 存储层（Storage）：
- 基础服务（Infrastructure）：

3.1.2.0.微信的一个U盘

- 客户端（Client）：
- 接入层（Gateway）：
- 逻辑层（Logic Service）：
- 存储层（Storage）：
- 基础服务（Infrastructure）：

4.1.0.数据存储方式

4.1.1.0.基本原理

- 4.1.1.1.0.数据分类
- 4.1.1.2.0.数据一致性
- 4.1.1.3.0.数据存储

4.1.2.0.分布式存储

- 4.1.2.1.0.分布式存储概述
- 4.1.2.2.0.分布式存储实现

5.1.0.数据传输方式

5.1.1.0.基本原理

- 5.1.1.1.0.数据分类
- 5.1.1.2.0.数据传输

5.1.2.0.分布式传输

- 5.1.2.1.0.分布式传输概述
- 5.1.2.2.0.分布式传输实现

6.1.0.数据处理方式

6.1.1.0.基本原理

- 6.1.1.1.0.数据分类
- 6.1.1.2.0.数据处理

6.1.2.0.分布式处理

- 6.1.2.1.0.分布式处理概述
- 6.1.2.2.0.分布式处理实现

7.1.0.数据安全

7.1.1.0.基本原理

- 7.1.1.1.0.数据分类
- 7.1.1.2.0.数据安全

7.1.2.0.分布式安全

- 7.1.2.1.0.分布式安全概述
- 7.1.2.2.0.分布式安全实现

8.1.0.分布式锁

8.1.1.0.基本原理

- 8.1.1.1.0.分布式锁概述
- 8.1.1.2.0.分布式锁实现

8.1.2.0.分布式锁的应用场景

- 8.1.2.1.0.分布式锁的应用场景概述
- 8.1.2.2.0.分布式锁的应用场景实现

9.1.0.分布式事务

9.1.1.0.基本原理

- 9.1.1.1.0.分布式事务概述
- 9.1.1.2.0.分布式事务实现

9.1.2.0.分布式事务的应用场景

- 9.1.2.1.0.分布式事务的应用场景概述
- 9.1.2.2.0.分布式事务的应用场景实现

10.1.0.分布式算法

10.1.1.0.基本原理

- 10.1.1.1.0.分布式算法概述
- 10.1.1.2.0.分布式算法实现

10.1.2.0.分布式算法的应用场景

- 10.1.2.1.0.分布式算法的应用场景概述
- 10.1.2.2.0.分布式算法的应用场景实现

11.1.0.分布式存储

11.1.1.0.基本原理

- 11.1.1.1.0.分布式存储概述
- 11.1.1.2.0.分布式存储实现

11.1.2.0.分布式存储的应用场景

- 11.1.2.1.0.分布式存储的应用场景概述
- 11.1.2.2.0.分布式存储的应用场景实现

12.1.0.分布式计算

12.1.1.0.基本原理

- 12.1.1.1.0.分布式计算概述
- 12.1.1.2.0.分布式计算实现

12.1.2.0.分布式计算的应用场景

- 12.1.2.1.0.分布式计算的应用场景概述
- 12.1.2.2.0.分布式计算的应用场景实现

13.1.0.分布式数据库

13.1.1.0.基本原理

- 13.1.1.1.0.分布式数据库概述
- 13.1.1.2.0.分布式数据库实现

13.1.2.0.分布式数据库的应用场景

- 13.1.2.1.0.分布式数据库的应用场景概述
- 13.1.2.2.0.分布式数据库的应用场景实现

14.1.0.分布式消息

14.1.1.0.基本原理

- 14.1.1.1.0.分布式消息概述
- 14.1.1.2.0.分布式消息实现

14.1.2.0.分布式消息的应用场景

- 14.1.2.1.0.分布式消息的应用场景概述
- 14.1.2.2.0.分布式消息的应用场景实现

15.1.0.分布式搜索

15.1.1.0.基本原理

- 15.1.1.1.0.分布式搜索概述
- 15.1.1.2.0.分布式搜索实现

15.1.2.0.分布式搜索的应用场景

- 15.1.2.1.0.分布式搜索的应用场景概述
- 15.1.2.2.0.分布式搜索的应用场景实现

16.1.0.分布式日志

16.1.1.0.基本原理

- 16.1.1.1.0.分布式日志概述
- 16.1.1.2.0.分布式日志实现

16.1.2.0.分布式日志的应用场景

- 16.1.2.1.0.分布式日志的应用场景概述
- 16.1.2.2.0.分布式日志的应用场景实现

17.1.0.分布式监控

17.1.1.0.基本原理

- 17.1.1.1.0.分布式监控概述
- 17.1.1.2.0.分布式监控实现

17.1.2.0.分布式监控的应用场景

- 17.1.2.1.0.分布式监控的应用场景概述
- 17.1.2.2.0.分布式监控的应用场景实现

18.1.0.分布式配置

18.1.1.0.基本原理

- 18.1.1.1.0.分布式配置概述
- 18.1.1.2.0.分布式配置实现

18.1.2.0.分布式配置的应用场景

- 18.1.2.1.0.分布式配置的应用场景概述
- 18.1.2.2.0.分布式配置的应用场景实现

19.1.0.分布式测试

19.1.1.0.基本原理

- 19.1.1.1.0.分布式测试概述
- 19.1.1.2.0.分布式测试实现

19.1.2.0.分布式测试的应用场景

- 19.1.2.1.0.分布式测试的应用场景概述
- 19.1.2.2.0.分布式测试的应用场景实现

20.1.0.分布式部署

20.1.1.0.基本原理

- 20.1.1.1.0.分布式部署概述
- 20.1.1.2.0.分布式部署实现

20.1.2.0.分布式部署的应用场景

- 20.1.2.1.0.分布式部署的应用场景概述
- 20.1.2.2.0.分布式部署的应用场景实现

Generated by Mermaid Chart in Chrome