**阅读笔记**

根据周二的讨论，方向开始从CI/CD的学习转到分布式事务的回滚上。

首先是对分布式事务处理技术的了解。

传统的事务处理技术主要应用于数据库系统。然而随着计算机科学技术的发展，事务处理的概念已经被引入了更为广泛的分布式网络计算环境。

事务具有ACID特性。

原子性（Atomic）：事物保持原子性。一个事务内的所有操作要么全部‘提交’，要么全部‘滚回’。

一致性（Consistent）：事务所产生的结果具有一致性。无论事务成功与否，事务结束后系统将处于一致状态。

隔离性（Isolated）:事务具有隔离性。事务的中间状态对于其他事务来说是不可见的。

持久性（Durable）：一个事务一旦完成，它所产生的结果是持久的，不可能再被修改。

事务具有ACID特性，任何满足ACID特点的工作单元都可以被称为事务。事务只能以两种方式结束：提交或滚回。事务提交意味着该事务中的所有操作都成功执行。事务提交后，所有相关操作所做的修改就永久地保存下来，系统进入一个新的一致状态。如果事务中的部分操作失败，则整个事务被视为失败，事务滚回，由相关操作所做的修改必须全部撤销，系统恢复到食物开始时的一致状态。

在分布式计算环境中，一个事务访问多个不同的独立资源，它们可能分布在不同的应用中，或者不同的进程中，甚至不同的计算机上，称这种事务为分布式事务。分布式事务所涉及的整个系统也具有ACID的特性。微服务也是分布式的，一个容器可以承载多个微服务，可以借助分布式事务的思想进行回滚操作。

事务管理器与两阶段提交协议

图示

描述已自动生成

如果所有资源都返回‘提交’，那么事务管理器决定提交事务，并依次向各个资源发出正式提交的信息；如果有一个或者更多的资源返回‘回滚’，那么事务管理器将决定回滚事务并通知所有的资源‘回滚’。（在容器中的多个微服务，是否也以这样的方式处理？）

事务的恢复机制

事务恢复的处理方法取决于事务出错的时机与位置（微服务也是如此）

如果事务的错误发生在两阶段提交协议开始之前，则利用两阶段提交协议就可以保证事务以回滚结束。用事务日志等辅助手段撤销已执行的事务操作，将事物恢复到初始状态，即前一个状态。称这种恢复为向后恢复。

当事务中的错误发生在两阶段提交协议开始之后事务提交之前，那么情况就会比较复杂。在这种情况下，一般采取向前恢复的机制，即恢复出错部件，重新执行事务操作将它推进到下一个一致状态。

下周准备把分布式事务与微服务结合起来看一下。

**学习笔记**

本周在进行项目实训，重点放在开发健身俱乐部管理系统中（mybatis+springboot+html）

设计了数据库物理模型，编写基本前端页面，通过controller实现view和model层的交互，实现了数据库基本的增删改查。