SOFAStack

分布式事务

产品简介

产品版本：AntStack Plus 1.11.0

文档版本：20221026

蚂蚁集团
ANT GROUP

法律声明

蚂蚁集团版权所有©2022，并保留一切权利。

未经蚂蚁集团事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

商标声明

蚂蚁集团 ANT GROUP及其他蚂蚁集团相关的商标均为蚂蚁集团所有。本文档涉及的第三方的注册商标，依法由权利人所有。

免责声明

由于产品版本升级、调整或其他原因，本文档内容有可能变更。蚂蚁集团保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在蚂蚁集团授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过蚂蚁集团授权渠道下载、获取最新版的用户文档。如因文档使用不当造成的直接或间接损失，本公司不承担任何责任。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.概述

分布式事务 DTX（Distributed Transaction-eXtended）是蚂蚁集团自主研发的金融级分布式事务中间件，用来保障在大规模分布式环境下业务活动的最终一致性。在蚂蚁集团内部被广泛地应用于交易、转账、红包等核心资金链路，服务于亿级用户的资金操作。

分布式事务可以与服务框架（SOFABoot、Spring Cloud、Dubbo 等）、数据源（数据访问代理、RDS、MySQL、OceanBase 等）、消息队列等蚂蚁集团中间件产品配合使用，轻松实现服务链路级事务、跨库事务、消息事务及各种组合。

分布式事务主要涉及发起方、参与者与事务管理器三个角色，具体描述如下：

- **发起方**：分布式事务的发起方负责启动分布式事务，通过调用参与者的服务，将参与者纳入到分布式事务当中，并决定整个分布式事务是提交还是回滚。一个分布式事务有且只能有一个发起方。
- **参与者**：参与者提供分支事务服务。当一个参与者被发起方调用后，该参与者会被纳入到该发起方启动的分布式事务中，成为该分布式事务的一个分支事务。一个分布式事务可以有多个参与者。
- **事务管理器**：事务管理器是一个独立的服务，用于协调分布式事务，包括创建主事务记录、分支事务记录，并根据分布式事务的状态，调用参与者提交或回滚方法。

2.产品优势

金融场景的全面覆盖

- **支付与转账**

  金融行业常见的支付、转账、账务等业务场景对于吞吐量有很高的要求。SOFAStack 的分布式事务产品在各类大促中的优异表现证明了性能不会成为瓶颈。

- **财富理财**

  这类场景中往往涉及的金额较大，所以对于产品的稳定性要求非常高，SOFAStack 的分布式事务产品拥有金融级的品质，可为业务的持续性与稳定性保驾护航。

- **保险与监管报送**

  参与方多、业务复杂度高是该类业务的典型特征，SOFAStack 的分布式事务产品历经十多年的演进历程，足以灵活应对各种场景，满足事务一致性要求，保证与各类业务完美结合。

政务领域支付更便捷

- **生活缴费**

  作为支付、转账场景的延伸，生活缴费在政务系统中不可或缺，例如水电费，电话费，上网资费等，都通过手机 APP 或者电脑端进行处理缴费。政务系统需要对缴费信息进行一致性处理，SOFAStack 的分布式事务产品可以保证关联信息同步修改，跨系统信息及时同步。

- **跨地域信息即刻同步**

  当前各地域政府机关往往有自己的数据库，人员流动、企业信息备案，都最初在本地进行登记备案。信息变更频繁的信息化时代，仅通过手工方式进行信息变更后的同步，会带来脏读和脏写的问题，采用 SOFAStack 的分布式事务产品可以保证政务机关的信息高效同步，精准一致。

泛互联网多领域场景

- **订单、会员卡、成长值、积分**

  以积分商城为例，使用会员卡余额购买商品，会涉及到扣减账户余额（数据库）、增加账户积分数量（数据库）、会员卡成长值提升、历史订单增加等服务。目前使用对账的方式来应对此类场景的性能较低，涉及业务扩展或改变时改造成本高。使用 SOFAStack 的分布式事务产品进行简单的改造接入，即可完成数据的同步。

- **担保交易**

  以电商抢购支付场景为例，秒杀抢购并发量高，性能要求高。通常流程尝试扣除用户可用资金，转移预冻结资金，增加中间账户可用资金（担保交易不能立即把钱打给商户，需要有一个中间账户来暂存），七天后需要将资金从中间账户划拨给商户。SOFAStack 的分布式事务产品可以支持大规模的抢购场景，保证客户成功支付，等到低锋期时，再慢慢消化支付数据，异步的执行资金到账流程，并且最终保证资金能顺利转入商户的账户中。

金融级品质的保障

- **金融级容灾保障**

  提供同城以及异地等多种模式以及多种级别的容灾能力，以业界最高规格的标准来保障客户业务的连续性。

- **无与伦比的性能**

相比传统二阶段模式，减少持有锁时间，大幅提升性能。特有的性能推进模式（Performance Bursting Mode）可以大幅提升吞吐量，曾在 2021 年双 11 中支撑 25.6 万笔/秒的支付操作。

- **使用简洁易于接入**

  蚂蚁集团多年沉淀的实操经验使产品具备了快速灵活的接入能力，易于使用与运维。

- **兼容性保障**

  分布式事务是一个抽象的基于 Service 层的概念，与底层事务实现无关，也就是说在分布式事务的范围内，无论是关系型数据库 MySQL、Oracle，还是 KV 存储 MemCache，或是列存数据库 HBase，只要将对它们的操作包装成分布式事务的参与者，就可以接入到分布式事务中。

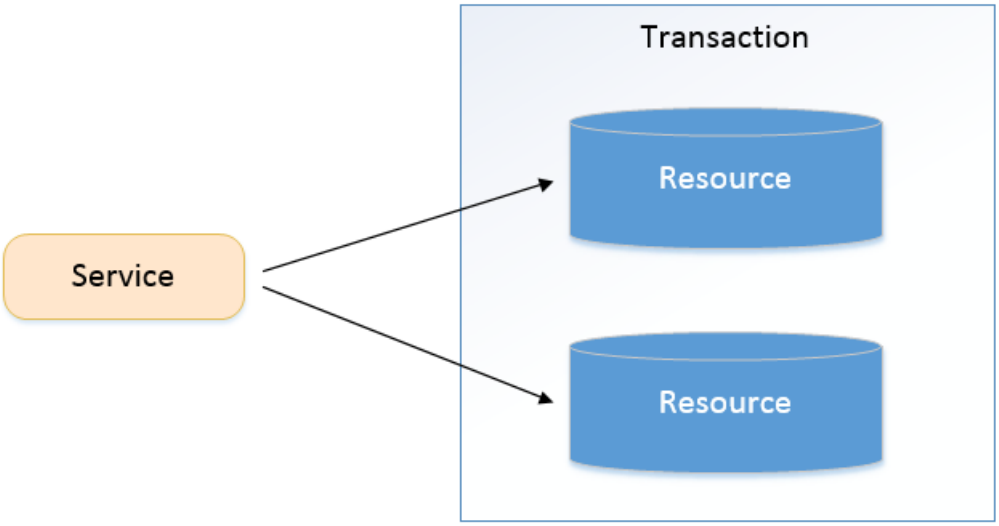
3.功能特性

分布式事务（Distributed Transaction-eXtended，简称 DTX）是蚂蚁集团自主研发的金融级分布式事务中间件，支持跨数据库、跨服务以及混合的方式处理分布式应用，具备多种接入模式和金融级配套功能，本文将主要介绍分布式事务的功能特性。

核心功能

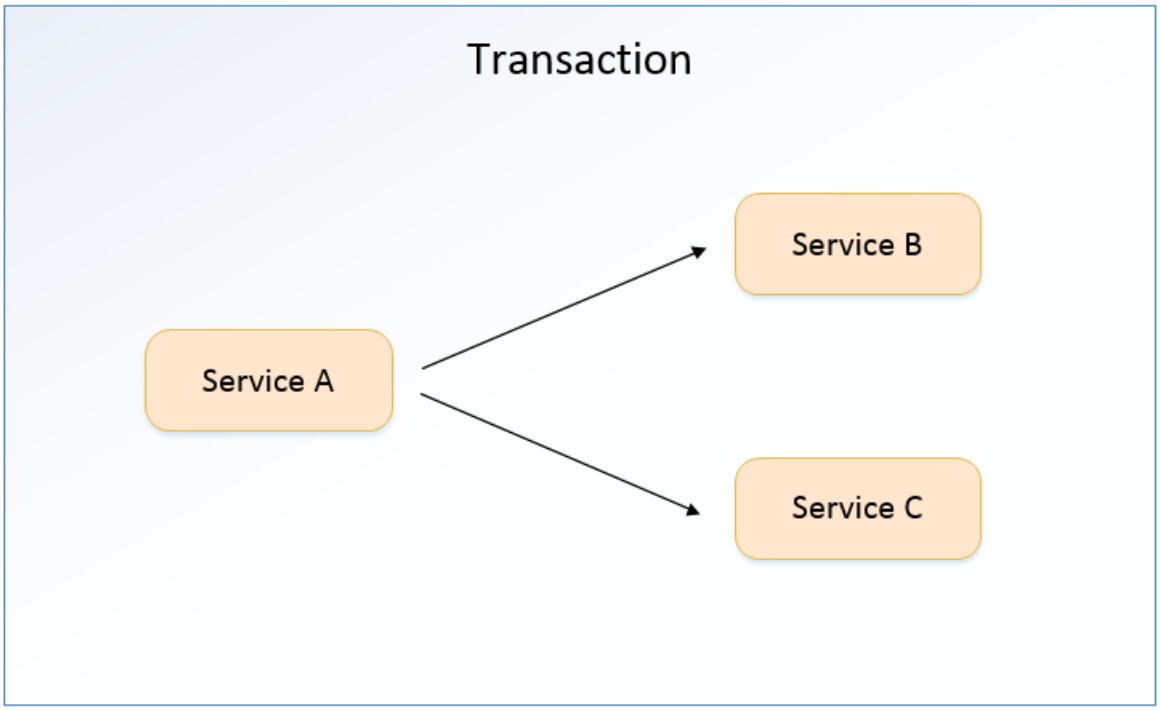
分布式事务拥有三大核心功能：跨数据库、跨服务以及混合分布式事务。

跨数据库分布式事务



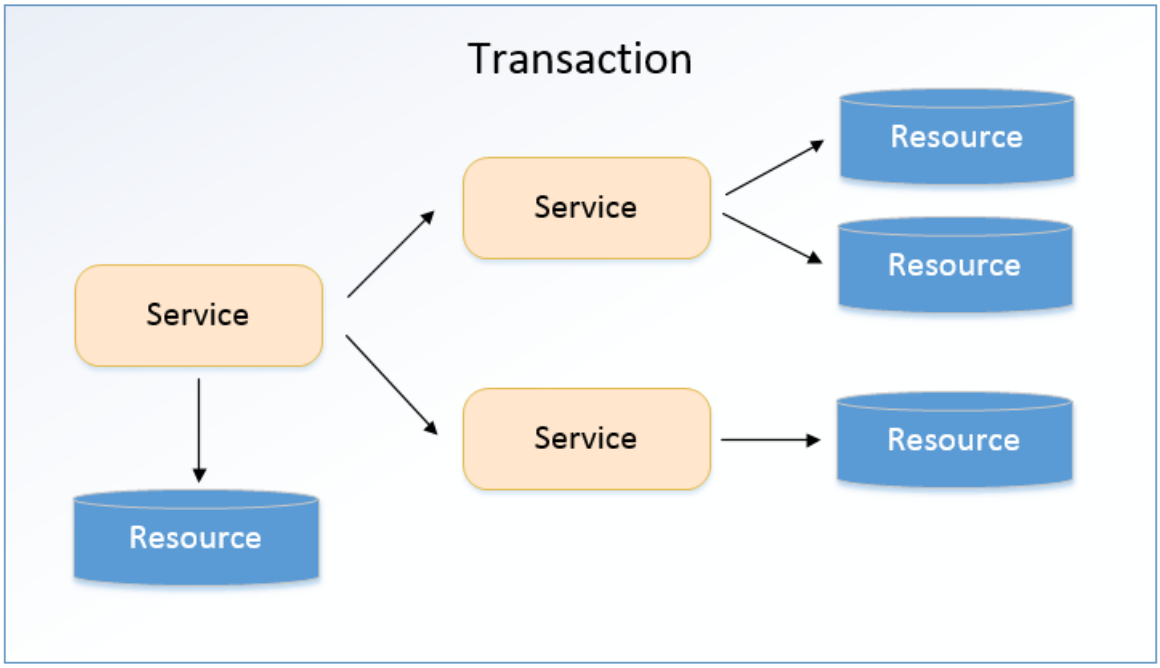
当业务规模增大，单库单表无法满足业务需求时，自然就会出现分库分表的情况。但是，单机事务又不能保证分库后的事务属性，分布式事务几乎无法避免。分布式事务可以让应用轻松具备跨库分布式事务处理能力，像使用单机数据库事务一样，透明地使用分布式事务。

跨服务的分布式事务



在基于 SOA（Service-Oriented Architecture，面向服务架构）的分布式应用环境下，系统按照功能解耦，拆分成不同的微服务，一项业务往往会涉及多个服务之间的调用。因此，为了保障业务完整，需要保证各调用服务间的数据一致性。分布式事务同样支持跨服务的分布式事务，并且可以很方便的与 SOFABoot、Spring Cloud、Dubbo 等框架打通，实现业务链路级别的分布式事务。

混合分布式事务



在一个大规模的分布式应用环境下，除了常见的微服务、数据库资源之外，还会涉及到消息队列、缓存等系统资源的使用。同时，依然需要保证这些资源间访问的数据的一致性。分布式事务支持在同一个分布式事务中引入数据库、服务、消息队列等不同类型的资源，并且支持混合接入模式。

接入模式

SOFAStack 的分布式事务产品随着业务的多样性发展而演变沉淀了多种接入模式，针对科技金融下多种业务场景灵活适配。

TCC 模式

TCC（Try-Confirm-Cancel）是一种高性能的分布式事务接入方案，该模式提供了更多的灵活性，几乎可满足任何您能想到的事务场景。TCC 模式提供自定义补偿型事务、自定义资源预留型事务、消息事务等场景，用户可以介入两阶段提交的过程，以达到特殊场景下的自定义优化及特殊功能的实现。

FMT 模式

为了解决 TCC 模式的易用性问题，分布式事务推出了框架管理事务模式（Framework-managed transactions，简称 FMT）。FMT 是一种无侵入的分布式事务解决方案，该模式解决了分布式事务的易用性问题，最大的特点是易于使用、快速接入以及对业务代码无侵入。

SAGA 模式

基于 Hector & Kenneth 发表论文 Sagas（1987）理论的长事务解决方案，在 Saga 模式中，业务流程中每个参与者都提交本地事务，当出现某一个参与者失败则补偿前面已经成功的参与者。该模式适用于业务流程长、业务流程多、参与者包含其它公司或老系统服务等场景。优势是一阶段提交本地事务无锁，事件驱动架构，参与者可异步执行，高并发高吞吐；补偿服务易于实现或老系统本身就有补偿（冲正）服务；支持服务编排、有可视化的设计器和执行轨迹监控。

XA 模式

高性能的标准 XA 事务方案，基于标准 XA 实现，消除与 JEE 开发相关的日益复杂的问题，帮助传统企业的业务无缝上云以及服务化拆分。并可以与蚂蚁集团自研数据库 OceanBase 共同打造实时数据一致性的整体解决方案。

金融级配套功能

沉淀蚂蚁集团历练多年的精粹，全方位的金融企业级能力为科技金融保驾护航。

隔离性保障

多层次全链路的纵深防御能力，金融级数据隔离能力，高度保障数据安全。

性能推进模式（Performance Bursting）

在日常高性能运转的基础上，通过性能推进模式来提升处理能力，在不改变底层基础设施的前提下从容应对峰值事务交易。

云端恢复机制

提供完备的云端服务功能，帮助业务应用在出现异常事务时进行自动恢复，避免业务损失。

4.应用场景

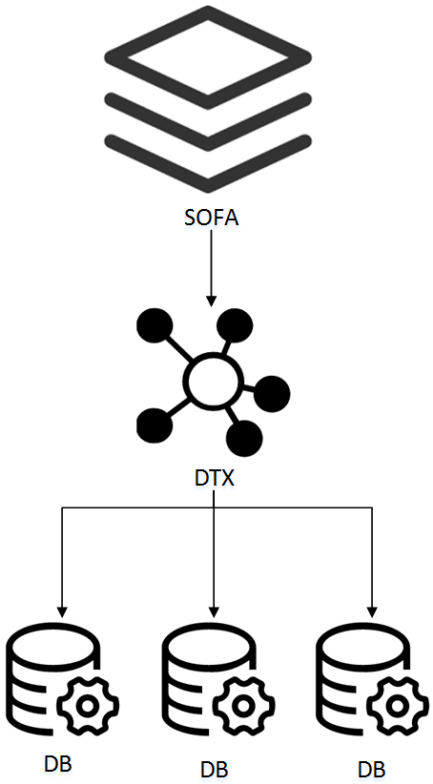
分布式事务可应用在多个涉及数据库操作的领域，尤其在金融领域可以做到全场景的覆盖与落地验证，包括：

- 支付与转账、账务：对于吞吐量有很高的要求
- 金融与理财：往往涉及的金额较大，所以对于产品的稳定性要求非常高
- 保险与监管报送：参与方多，业务复杂度高是该类业务的典型特征

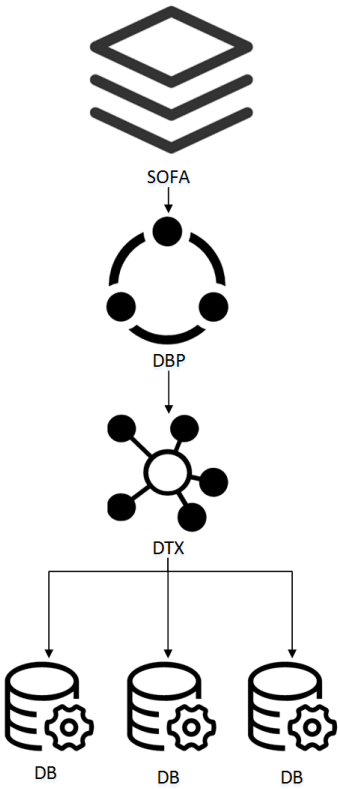
典型的应用场景如下：

分库分表后的跨数据库分布式事务

例如支付宝的交易服务，随着业务规模的增大，单个交易流水表已经不能满足业务需求，需要通过分库分表实现数据水平拆分。但是水平拆分后，单表的数据被分散到多个库的表中，原来对单表多行数据进行的变更，可能会变为对多库多表的数据变更，即单机本地事务变成了分布式事务。



使用分布式事务能够轻易获得分布式事务处理能力。如果交易服务使用数据访问代理来分库分表，虽然数据访问代理本身不支持分布式事务，但是分布式事务可以轻松和数据访问代理集成，使得数据访问代理具备分布式事务的处理能力，解决分库分表后的跨库分布式事务问题。

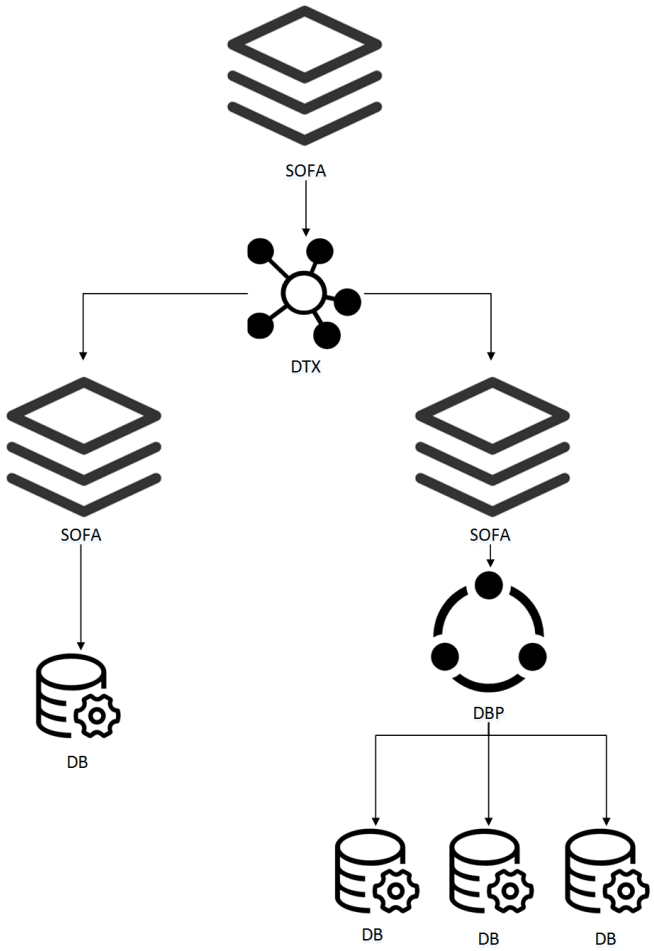


跨服务的分布式事务

例如，支付宝核心链路上的三个服务为：交易、支付、账务。当用户发起一笔交易时：

1. 首先访问交易服务，创建交易订单。
2. 然后交易服务调用支付服务为该交易创建支付订单，执行收款动作。
3. 最后支付服务调用账务服务记录账户流水和记账。

为了保证三个服务一起完成一笔交易（要么同时成功，要么同时失败），可以使用分布式事务将这三个服务放在一个分布式事务中，从而保证服务间调用的数据一致性。



混合分布式事务

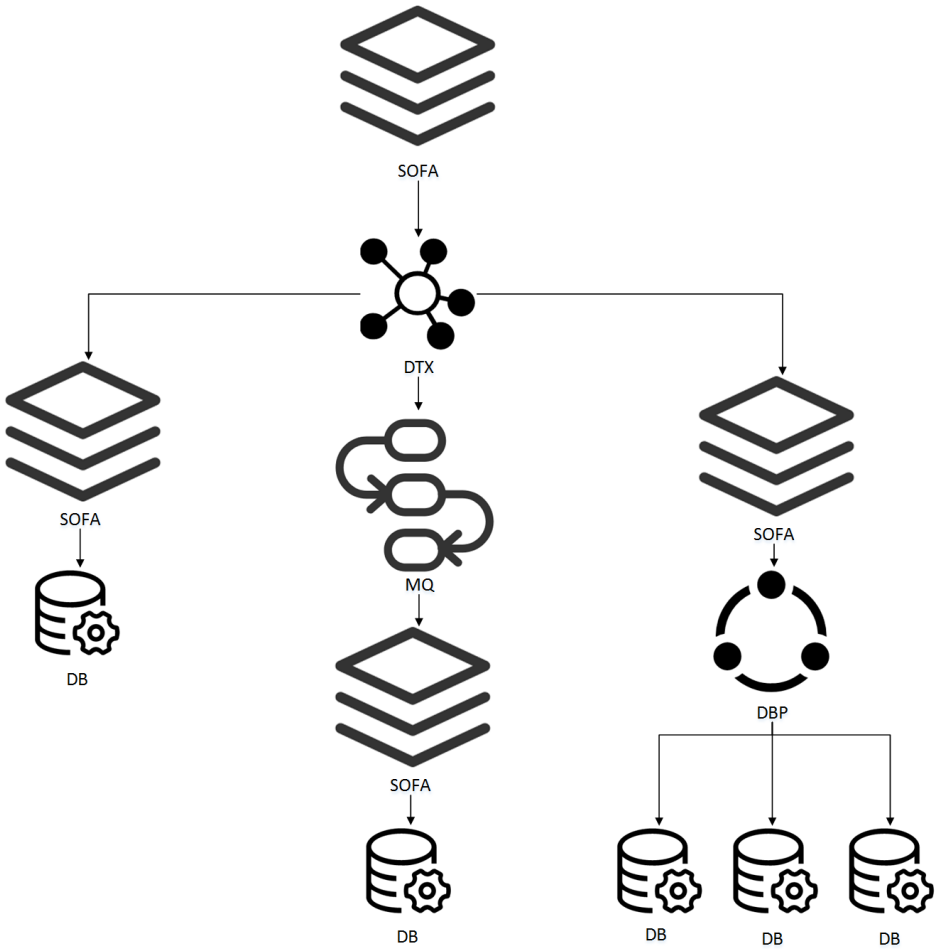
有些系统在使用数据库保证系统内数据一致的同时，也会使用消息队列作为和其他系统间的消息传递，完成不同系统间的数据一致。

例如会员注册服务和邮件发送服务，当用户注册会员成功，需要给用户发送一封邮件，告诉用户注册成功，并提示用户激活该会员。但要注意两点：

- 如果用户注册成功，一定要给用户发送一封邮件；
- 如果用户注册失败，一定不能给用户发送邮件。

因此，这同样需要会员服务和邮件服务保证原子性（要么都执行，要么都不执行）。不一样的是，邮件服务只是一种被动型的业务，并不影响用户是否能够注册成功，它只需要在用户注册成功以后发送邮件给用户即可，邮件服务不需要参与到会员服务的活动决策中。因此，可以使用消息队列来解耦会员和邮件服务。

对于此种业务场景，分布式事务可以将会员服务、消息队列组成分布式事务模型，保证事务原子性。然后通过消息队列的可靠特性，确保消息一定能够被邮件服务消费，从而使得会员与邮件服务在同一个分布式事务中。同时，邮件服务也不会影响会员服务的执行过程，只在会员服务执行成功后被动接收发送邮件的请求。


SOFA
DTX
SOFA
DB
MQ
SOFA
DB
SOFA
DBP
DB
DB
DB

5.基础术语

中文	英文	释义
事务	transaction	事务是指作为单个逻辑工作单元执行的一系列操作，要么完全执行，要么完全不执行。
分布式事务	distributed transaction	事务的发起者、资源及资源管理器和事务协调者分别位于不同的分布式系统的不同节点之上。
分支事务	action	一个分布式事务可能包含多个数据库本地事务，在分布式事务框架下，分支事务可能是一个分库上执行的 SQL 语句，或是一个自定义模式服务的调用。
发起方	initiator	分布式事务的发起方负责启动分布式事务，通过调用参与者的服务，将参与者纳入到分布式事务当中，并决定整个分布式事务是提交还是回滚。一个分布式事务有且只能有一个发起方。
参与者	participant	参与者提供分支事务服务。当一个参与者被发起方调用，则被纳入到该发起方启动的分布式事务中，成为该分布式事务的一个分支事务。一个分布式事务可以有多个参与者。
事务管理器	transaction manager	事务管理器是一个独立的服务，用于协调分布式事务，包括创建主事务记录、分支事务记录，并根据分布式事务的状态，调用参与者提交或回滚方法。
主事务记录	activity record	又叫 Activity 记录，是整个分布式事务的主体。其最核心的数据结构是事务号（TX_ID）和事务状态（STATE），它是在启动分布式事务时持久化写入数据库的，它的状态决定了这个分布式事务的状态。
分支事务记录	action record	又叫 Action 记录，用于标识分支事务。它记录了该提供该分支事务的参与者的信息，其中包括参与者的唯一标识等。通过分支事务信息，事务管理器就可以对参与者进行提交或者回滚操作。