同济大学 TONGJI UNIVERSITY

OCEANBASE

蚂蚁集团 ANT GROUP

OceanBase 事务管理探究

小组调研报告

计算机科学与技术、信息安全

关佶红教授

2024年6月17日

同济大学 TONGJI UNIVERSITY

OCEANBASE

蚂蚁集团 ANT GROUP

01 OcenBase 简介

02 OcenBase 事务概念

03 OcenBase 事务管理特点

04 OcenBase 事务管理源码探究

05 OcenBase 总结、创新与展望

同济大学 TONGJI UNIVERSITY

OCEANBASE

蚂蚁集团 ANT GROUP

01 OcenBase 简介

01 OcenBase 简介

- 2010年自主研发
- 旨在解决传统关系型数据库在分布式环境下面临的扩展性和一致性挑战

- **架构设计：**
  - 基于无共享架构的关系型数据库管理系统（RDBMS）。
  - 在能源、政府、交通等多个行业广泛应用。
- **性能表现：**
  - 2019年TPC-C基准测试中荣获第一名，处理速度是第二名Oracle的两倍。
  - 2019年11月处理了每秒6100万次的交易记录。
- **实际应用：**
  - 广泛应用于支付宝和阿里巴巴多个系统。
  - 服务于南京银行、浙商银行、人保健康险等多个外部客户。

同济大学 TONGJI UNIVERSITY

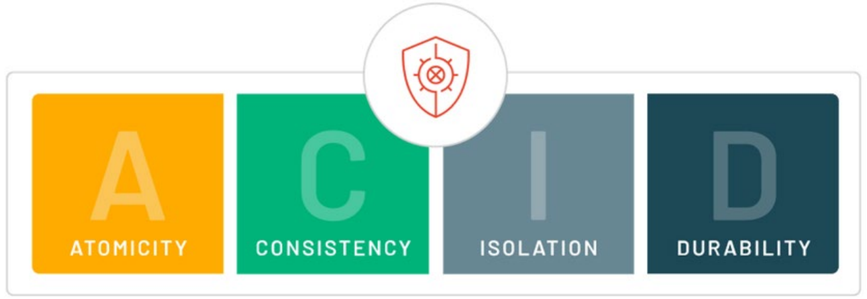
OCEANBASE

蚂蚁集团 ANT GROUP

02 OcenBase 事务概念

02 OceanBase 中事务的概念

数据库事务特性

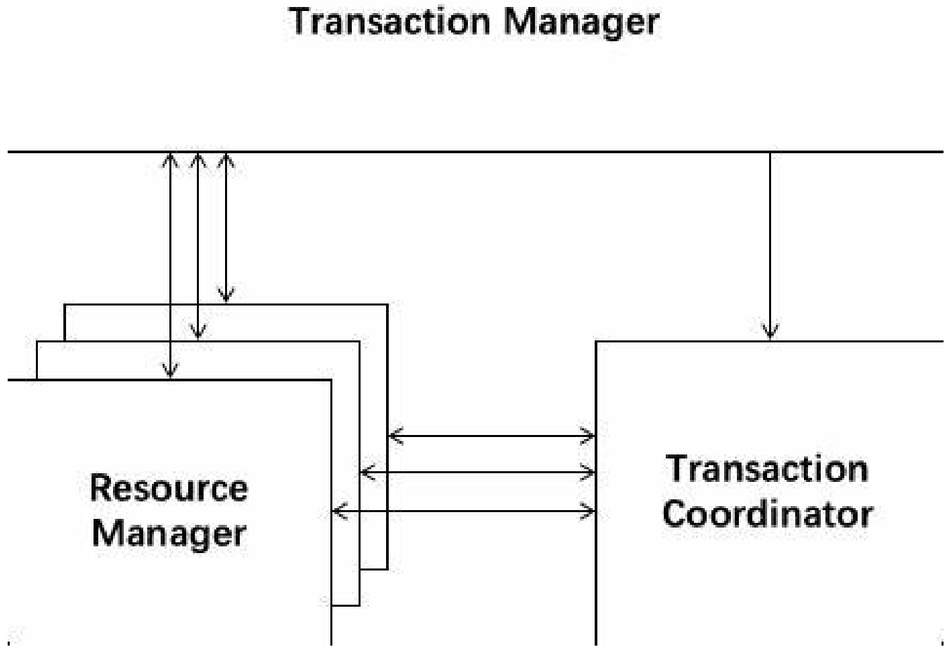


- **原子性（Atomicity）**
  - 确保每个事务要么完全完成，要么完全不发生。
  - OceanBase通过优化的两阶段提交协议（2PC）保持原子性，协调者不再写日志，变成无持久化状态的状态机，事务状态由参与者的持久化状态决定。
- **一致性（Consistency）**
  - OceanBase利用Multi-Paxos协议在多个数据副本之间保证一致性。
  - 通过多轮决策和投票过程，确保分布式环境下的数据一致性。
- **隔离性（Isolation）**
  - OceanBase通过不同级别的隔离选项管理并发操作。
  - 支持Oracle兼容模式（Read Committed和Serializable）和MySQL兼容模式（Read Committed, Serializable, Repeatable Read）。
- **持久性（Durability）**
  - OceanBase依赖重做日志（Redo Log）和预写日志（WAL）机制确保数据持久化。
  - 使用Paxos协议将数据复制到多个副本，提高了分布式数据处理的可靠性

02 OceanBase 中事务的概念

数据库事务实现基础

- **语句级回滚**
  - 当一条语句执行过程中没有报错，该语句的修改是成功的；若报错，则会回滚该语句的所有操作。
  - 语句级回滚的效果相当于该语句未曾执行过，保证了数据库事务的原子性。回滚涉及的元素包括全局索引、触发器、行锁等。
- **两阶段提交协议（2PC协议）**
  - 2PC是一个经典的强一致、中心化的原子提交协议。
  - Coordinator负责提议，Participant根据提议反结果，协调者根据反馈决定提交或回滚。
  - **OceanBase实现：**OceanBase通过原生的两阶段提交协议，确保分布式事务的原子性。



02 OceanBase 中事务的概念

事务的作用

- **提高了恢复到正常状态的方法**
  - 在系统故障情况下，确保事务要么完整提交，要么完全回滚，保护数据库完整性。
- **保持数据库一致性**
  - 利用分布式架构和一致性协议，维护跨节点的数据一致性。
- **处理并发访问**
  - 通过隔离级别和锁机制管理并发访问，确保高并发处理中的一致性。

- **OceanBase 的特定优势**
  - 分布式事务管理: 采用高效的分布式协议和算法，优化跨节点事务的性能和一致性。
  - 灵活的隔离级别: 提供多种隔离级别，允许用户根据具体需求和性能考虑调整事务的隔离程度。

02 OceanBase 中事务的概念

事务的结构

- OceanBase的事务结构遵循典型的数据库事务流程，即一个数据库事务包含一条或者多条 DML 语句。整个事务流程包括开启事务、语句执行、和结束事务的步骤。

开启事务

```
START TRANSACTION
    [transaction_characteristic [, transaction_characteristic] ...]

transaction_characteristic: {
    WITH CONSISTENT SNAPSHOT /* 在事务开始时创建一个一致性视图 */
  | READ WRITE /* 将事务设置为读写模式 */
  | READ ONLY /* 将事务设置为只读模式 */
}

BEGIN [WORK]

SET autocommit = {0 | 1} /* 设置自动提交的状态 */
```

02 OceanBase 中事务的概念

事务的结构

语句执行

- 在事务的执行过程中，OceanBase 在每个涉及的分区创建事务上下文。这个上下文记录了语句执行过程中的数据快照版本号以及对该分区所做的修改。

结束事务

- 显式结束事务:
  - COMMIT: 将事务中的所有更改永久保存到数据库中。
  - ROLLBACK: 撤销事务中的所有操作。
- 隐式提交事务:
  - 执行DDL（数据定义语言）操作（如CREATE、DROP等）时，OceanBase会隐式提交当前事务，随后的操作将在新事务中执行。
- 断开连接时回滚事务:
  - 在事务执行过程中，如果客户端断开连接，OceanBase会隐式发起一个ROLLBACK请求，回滚当前未完成的事务。

02 OceanBase 中事务的概念

活跃事务

- 活跃事务是指事务已经开启，但还没有提交或者回滚的事务。活跃事务所做的修改在提交前都是临时的，其它的事务无法看到。

Savepoint

Savepoint 是 OceanBase 数据库提供的可以由用户定义的一个事务内的执行标记。
用户可以通过在事务内定义若干标记并在需要时将事务恢复到指定标记时的状态。

当执行 ROLLBACK TO 命令时，OceanBase 数据库内部会执行以下操作：

1. 将事务内的所有大于该 Savepoint 对应"sql sequence"的修改全部回滚，并释放对应的行锁。
2. 删除该 Savepoint 之后创建的所有 Savepoint。

ROLLBACK TO 命令执行成功后，事务仍然可以继续操作。

02 OceanBase 中事务的概念

Redo日志

Redo 日志是一种物理日志，它记录了数据库对于数据的全部修改历史，具体的说记录的是一次写操作后的结果。从某个持久化的数据版本开始逐条回放 Redo 日志可以还原出数据的最新版本。

日志作用

宕机恢复：在事务提交前将 Redo 日志持久化，保证事务的原子性和持久性。如果 observer进程退出或所在的服务器宕机，重启 OBServer 节点会扫描并回放本地的 Redo 日志用于恢复数据。宕机时未持久化的数据会随着 Redo 日志的回放而重新产生。

日志类型

每个分区的所有日志要求在逻辑上连续有序。机器上同一租户的所有 Tablet 产生的日志会写入到同一个日志流，日志流中的日志存在全序关系。

日志产生

在 V3.x 及之后的版本中，OceanBase 数据库新增了即时写日志功能，当事务内数据超过2 MB时，生成 Redo 日志，提交到 Clog 模块。以 2 MB 为单位主要是出于性能考虑。

02 OceanBase 中事务的概念

Redo日志

日志回放

Redo 日志的回放是 OceanBase 数据库提供高可用能力的基础。日志同步到 Follower 副本后，副本会将日志按照 transaction_id 哈希到同一个线程池的不同任务队列中进行回放。

日志容灾

当某一分区的 Leader 所在的机器发生故障或由于负载过高无法提供服务时，可以重新将另一个机器上的副本选为新的 Leader。因为它们拥有相同的日志和数据，新 Leader 可以继续提供服务。只要发生故障的副本不超过一半，OceanBase 数据库都可以持续提供服务。

控制回收

日志文件中记录了数据库的所有修改，因此回收的前提是日志相关的数据都已经成功持久化到磁盘上。
如果数据还未持久化就回收了日志，故障后数据就无法被复原。

03 OceanBase 数据库的锁机制

并发性定义：数据并发性和数据一致性两种语义称为并发控制模式，即为 ACID 中的 I(Isolation level)。数据库的并发性体现在数据库允许用户通过事务并发地访问与修改用一个数据，数据一致性用户观察区域内为一致的数据库状态。

并发控制模型

- OceanBase 数据库、支持快照读和读已提交两种隔离级别，并在分布式语义上隔离级别保证外部一致性。

多版本数据和事务表

- **OceanBase** 选择了多版本来作为存储，并让事务对于全局，维护两个版本号，读版本号和提交版本号。
- 内存中存在一个事务表，事务表中记录了每个事务的id、状态以及版本。事务开始和提交时会通过全局时间戳缓存服务（Global Timestamp Cache）获取时间戳作为读时间戳和作为提交时间戳参考的一部分。

提交请求处理

- OceanBase 的分布式事务有三个状态，RUNNING、PREPARE 和 COMMIT。
- OceanBase 选择在事务中维护本地提交版本号。
- 事务的全局提交版本号是由所有分区本地提交版本号的最大值决定的。最后事务的全局提交版本号 一定大于等于本地分区的本地提交版本号，从而保证每个分区的数据读写安全。

读写请求处理

- 在写入数据的时候，若发现本行的多版本有正在执行的事务，则会把这次请求放入锁管理器中等待， 由此实现等待队列，通过锁或超时来唤醒此写请求。
- 在OceanBase中，读取时使用读版本号更新本地最大读时间戳。对于RUNNING状态的事务，可跳过读数据。对于PREPARE状态的事务，若本地时间戳大于读时间戳，则无需读取；否则需等待确认。

03

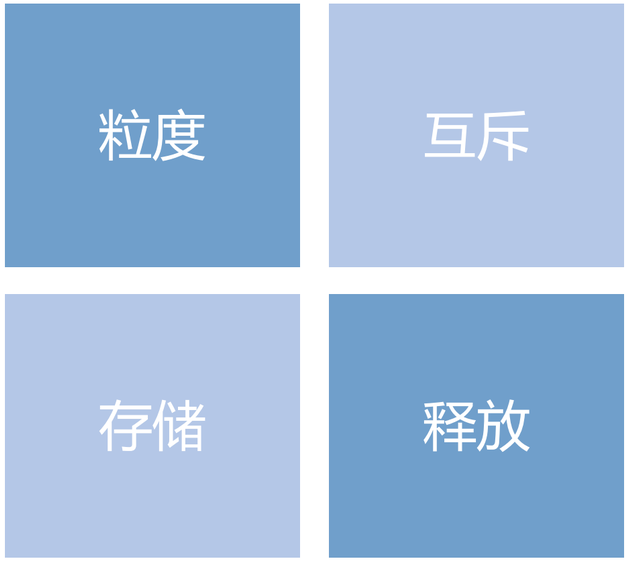
OcenBase 事务管理特点

03 OceanBase 数据库的锁机制

锁机制

- OceanBase 数据库现在**不支持表锁**，只支持行锁，且只存在互斥行锁。

- OceanBase 数据库的**锁存储**在行上，这种存储方式减少了内存中所需要维护的锁数据结 构带来的开销。

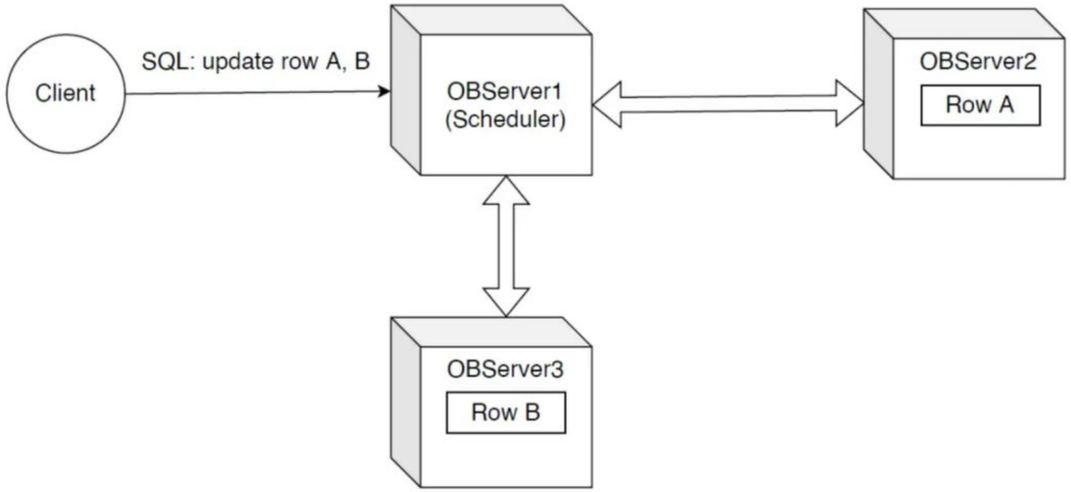


- OceanBase 数据库使用了**多版本两阶段锁**，事务的修改非原地修改，而产生新版本。不需行锁仍可以维护并发控制能力。

- OceanBase 用户回滚至 SAVEPOINT 后，事务内部会将之后所有涉及数据的行锁**根据数据库锁机制的互斥机制进行释放。**

LCL 死锁检测

- **LCL 死锁检测方案**采用的是一种经过特殊设计的边跟踪算法（edge-chasing algorithm）。
- 在 LCL 死锁检测方案中，每个节点维护两个状态，分别称为深度值以及令牌值，为防止多杀，一个节点维护的令牌值数量不能多于一个，令牌值之间可以比较，并使大的令牌值覆盖小的令牌值。
- 如此可以保证 在一个环路中，只有最大令牌值的节点可以探测到死锁，可以避免多杀的问题。



03 OceanBase 数据库一致性

一致性定义：数据库通过维护每次更改，产生新的版本，从而做到读写不互斥,这被称为多版本并发控制（MVCC）。 对于不同的事务版本，我们需要为这种数据多版本来定义语义，保证用户看到一个一致的数据库状态，即数 据的一致性快照。需要注意的是，此处的数据一致性不等价于ACID中的C(Consistency)。

多版本读一致性

为了支持数据读写不互斥，OceanBase数据库存储了多个版本的数据。为了处理多版本数据的语义，我　们需要维护多版本一致性。OceanBase数据库的多版本一致性是通过读版本和数据版本来保证的，通过给读　取版本号，返回小于读取版本号的所有提交数据，来定义多版本一致性。

未提交事务

不能读到非本事务的未提交事务,否则若对应事务回滚，就会产生脏读（dirtyread）。

事务一致性快照

要读取小于读取版本号的所有提交数据，来保证一个用户可理解的一致性点，否则我们就会产生会返回一半事务（fracturedread）。

读写不互斥

在满足未提交事务与事务一致性点的前提下，依旧要保证读写不互斥。

03 多版本读一致性使用

OCEANBASE

同济大学 TONGJI UNIVERSITY

弱一致性读

OceanBase 数据库的弱一致性读依旧提供了事务的一致性快照，不会返回未提交事务和一半事务的情况。

强一致性读

OceanBase数据库的强一致性读分为两种，分别是事务级别读版本号和语句级别版本号， 分别提供给快照读和读已提交两个隔离级别使用，需要提供返回事务一致性点的能力。

只读事务

OceanBase数据库的只读语句也是要求提供强一致性读相同的能力，需要提供返回事务一致性点的能力。

弱一致性读

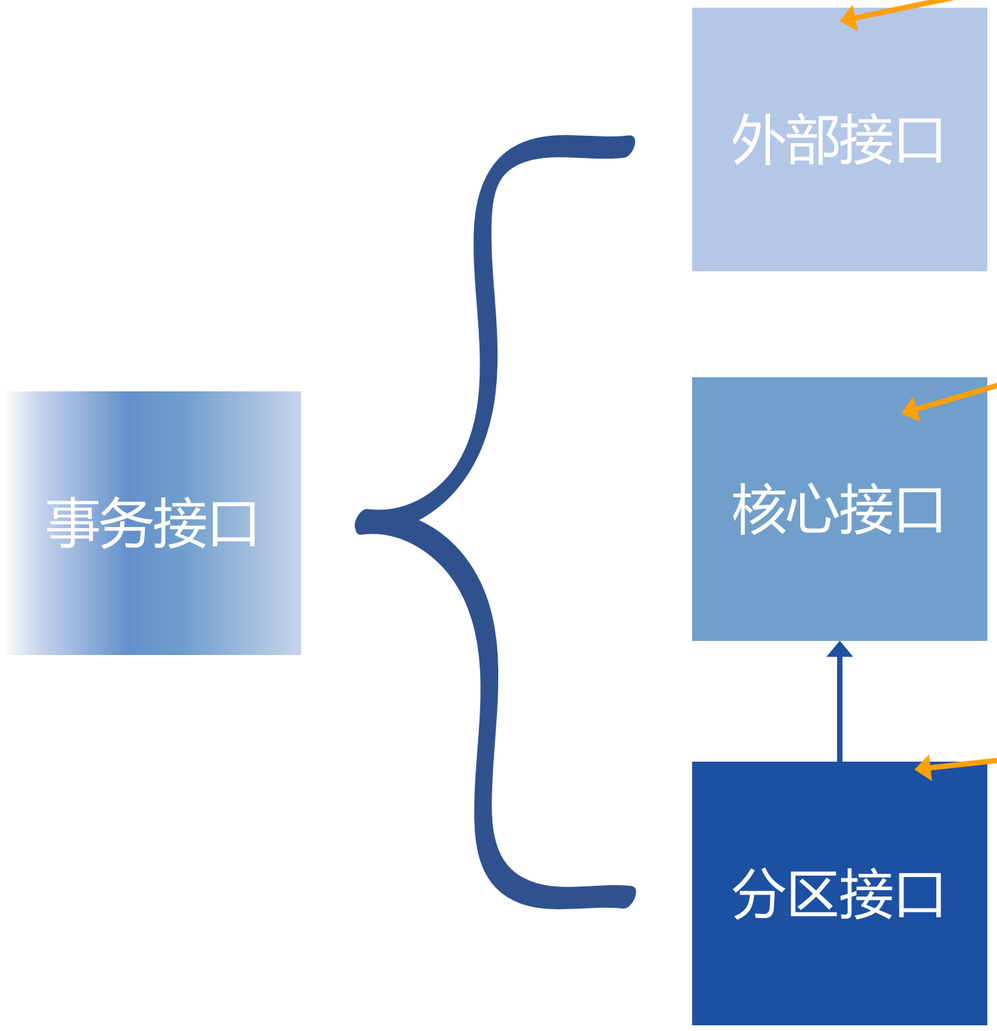
OceanBase数据库需要提供可以备份到一个事务一致性快照，防止备份了多余、未提交的事务或者没有备份需要备份的事务。

实现并发控制的核心之一，就是多版本读一致性在数据库内部的普遍适用性

04 OcenBase 事务管理源码探究

04 OceanBase 源码探究(事务)

接口代码分析



OcenBase协议层进行封装，外部调用
ob_sql_trans_control.cpp

```
int ObSqlTransControl::end_trans(
    storage::ObPartitionService* ps, ObSQLSessionInf
```

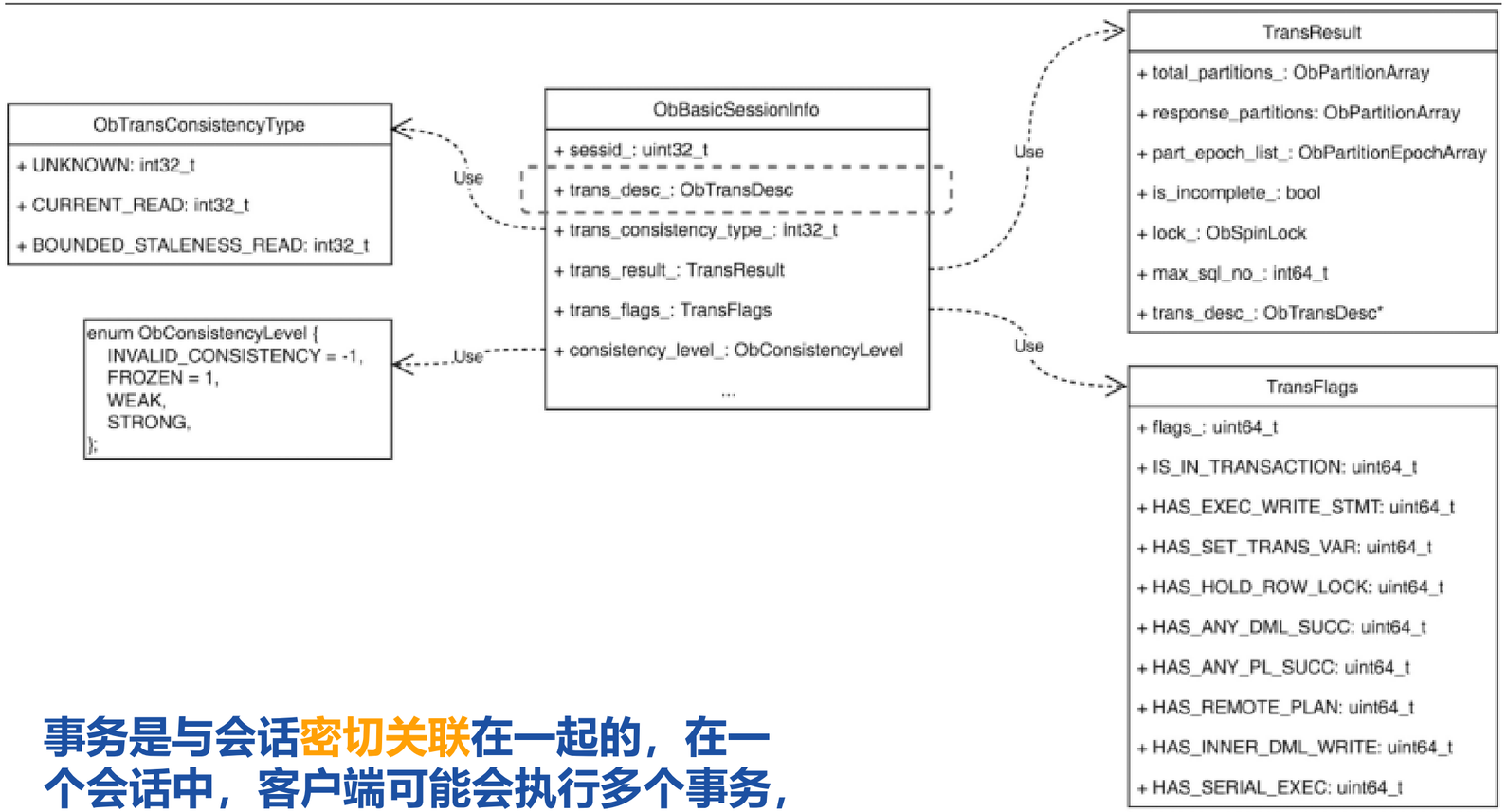
storage/trans/ob_trans_service.cpp

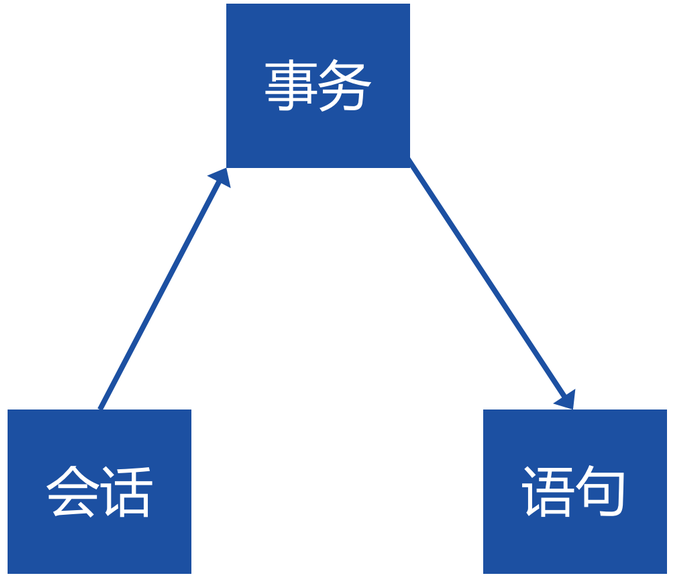
```
//事务开始，创建事务唯一id
int start_trans(const uint64_t tenant_id, const uint64_t
    const int64_t expired_time, const uint32_t session_
        ObTransDesc& trans_desc){
```

share/partition_table.cpp
sql/ob_sql_partition_location_cache.cpp

04 OceanBase 源码探究(事务)

会话代码分析





事务是与会话密切关联在一起的，在一个会话中，客户端可能会执行多个事务，不过在会话存续的任一时刻仅有一个事务运行（称为活跃事务）。

OceanBase 管理事务的核心就是维持会话、事务、语句的状态以及它们之间的关联，并根据 SQL 语句在各个节点上的执行情况适时推进它们的状态变化

同济大学 TONGJI UNIVERSITY

OCEANBASE

蚂蚁集团 ANT GROUP

05

总结与创新

05 总结与创新

- 与前期调研不同的地方

事务概念与特性

版本更迭与完善

OceanBase 数据库 V4.2.1
2023-10-13

OceanBase 数据库 V4.3.1
2024-05-22

- 深入调研了事务的开始与结束以及对应的**源码**部分，解释了事务的结构，包括开启事务、语句执行和结束事务的**步骤**，并结合 OceanBase 的具体实现进行了说明。

事务管理特点

- 多版本并发控制 (MVCC) + 行锁机制: OceanBase 使用 MVCC 和行锁机制来管理并发事务，实现了读写不互斥，从而提高了并发处理能力。深入研究了**读一致性**的使用
- Redo 日志和 Paxos 协议: 深入调研了使用 **Redo 日志和 Paxos 协议**来保证事务的持久性和多副本一致性，调研了Redo日志的多个应用类型。

事务源码探究

- 分析了 OceanBase 事务的**接口**，包括外部接口、核心接口和分区接口，
- 分析了会话中与事务相关的**关键属性和数据结构**。从快照管理、事务提交、并发控制和全局时间戳等方面进行了源码分析.
- 分析了 OceanBase 的 clog **日志模块**，包括日志的提交、同步、持久化和回放机制，并解释了其在事务持久性和多副本一致性方面的作用。

在版本更新的基础上进行深入调研

参考文献

[1] 杨冬青, 李红燕等. 数据库系统概念（第 7 版）. 机械工业出版社, 2021.

[2] 彭煜玮, 杨传辉, 杨志丰. OceanBase 源码解析. 机械工业出版社, 2023.

[3] Zhenkun Yang ,Chuanhui Yang ,Fusheng Han. OceanBase: A 707 Million tpmC Distributed Relational Database System. 2022,PVLDB,15,12,3385-3397.

[4] 李凯, and 韩富晟. "OceanBase 内存事务引擎." 华东师范大学学报 (自然科学版) 2014, no. 5 (2014): 147.

[5] 阳振坤. "OceanBase 关系数据库架构." 华东师范大学学报 (自然科学版) 2014, no. 5 (2014): 141.

[6] CSDN. 第四章：OceanBase 集群技术架构（分布式事务、MVCC、事务隔离级别）[OL].(2023-03-15)[2023-12-10].https://blog.csdn.net/qq_43714918/article/details/129557961.

[7] 周欢, 樊秋实, and 胡华梁. "OceanBase 一致性与可用性分析." 华东师范大学学报 (自然科学版) 2014, no. 5 (2014): 103.

[8] Kai, L. I., and H. A. N. Fu-Sheng. "Memory transaction engine of OceanBase." Journal of East China Normal University (Natural Science) 2014, no. 5 (2014): 147.

[9] Xu, Liang-yu, Xiao-fang Zhang, Li-jun Zhang, and Jin-tao Gao. "Design and Implementation of Trigger on OceanBase." In Advances in Computer Communication and Computational Sciences: Proceedings of IC4S 2017, Volume 2, pp. 121-132. Springer Singapore, 2019.

[10] Huan, Z. H. O. U., F. A. N. Qiu-Shi, and H. U. Hua-Liang. "Consistency and availability in OceanBase." Journal of East China Normal University (Natural Science) 2014, no. 5 (2014): 103.

[11] 祝君, 刘柏众, 余晟隽, 宫学庆, and 周敏奇. "面向 OceanBase 的存储过程设计与实现." 华东师范大学学报 (自然科学版) 2016, no. 5 (2016): 144.

[12] OceanBase版本发布记录 https://www.oceanbase.com/product/oceanbase-database-community-rn/releaseNote

同济大学 TONGJI UNIVERSITY

OCEANBASE

蚂蚁集团 ANT GROUP

OceanBase 事务管理探究

小组调研报告

计算机科学与技术、信息安全

关佶红教授

2024年6月17日