SNAPSHOT ISOLATION 隔离级别功能设计文档

修订历史

版本	修订日期	修订描述	作者	备注
Cedar 0.2	2016-06-16	SNAPSHOT ISOLATION 隔离级别功能设计文档	郭进伟 肖冰	

1 需求分析

ANSI SQL-92 根据异常现象定义了四种隔离级别，如下表。大多数的数据库系统按照这四种隔离级别对外提供不同要求的服务，但不同的数据库系统由于不同的并发控制机制会在同一名称的隔离级别上存在细微的差别。
本文介绍Cedar 0.2版本中新增的快照隔离级别（Snapshot Isolation）。该隔离级别对外的接口为REPEATALBE READ，即用户设置隔离级别为REPEATABLE READ时，系统内部对应的隔离级别为SNAPSHOT ISOLATION。

隔离级别	脏读	模糊读	幻读
READ UNCOMMITTED (读未提交)	可能	可能	可能
READ COMMITTED (读已提交)	不可能	可能	可能
REPEATABLE READ (可重复读)	不可能	不可能	可能
SERIALIZABLE (可串行化)	不可能	不可能	不可能

SNAPSHOT ISOLATION只关注事务开始之前提交的数据，不会看到任何在事务执行期间其他并发事务未提交的数据或者提交的修改。但是事务可以在执行期间看到自己之前未提交的修改。相对于READ COMMITTED，SNAPSHOT ISOLATION能够保证每一个事务访问到一个完全稳定的数据库视图。

快照隔离作为一种多版本并发控制方法能够保证读不被阻塞并且避免大多数的异常，可增强OLTP应用程序的并发性。它允许一些写倾斜的异常但是能够支持较好的系统性能，很早就被应用于Oracle，在PostgreSQL也有长期相应的研究，是数据库中比较流行的一个隔离级别。

Cedar有语句级的快照隔离，而没有事务级的快照隔离，即SELECT操作独立于事务本身。对于每一个SELECT语句而言，都可以看到数据库最新的快照。所以Cedar只支持READ COMMITTED事务隔离级别，且事务无法读到自己修改的但是没有提交的数据。

为了进一步保证事务执行的正确性，基于Cedar的增量数据存储和更新机制，该功能意在为其添加新的隔离级别支持。

2 适用场景

适用于需要提供事务级“快照隔离级别”的业务需求。

3 功能简述

1. 事务无论属于哪种隔离级别均可以读到自己修改的但是没有提交的数据。
2. 实现SNAPSHOT ISOLATION，使得该隔离级别下运行的事务能够避免脏读的同时避免模糊读。

4 设计思路

4.1 子功能模块划分

基于隔离级别的需求和Cedar 0.1 处理读请求的基础，SNAPSHOT ISOLATION隔离级别的实现需要解决以下几个问题:

首先Cedar的SELECT执行流程不涉及其所属事务，即读取操作没有关联其所属事务相关信息。所以第一个要解决的问题是执行读取操作执行过程中，**事务信息的添加和传递。**

其次，记录和传递事务信息是为了在不同的隔离级别下面对数据版本的读取有不同的控制。因此在**UPS处理读取请求**时，执行流程需要满足不同隔离级别需求。

最后，对于客户端而言，需要提供设置新的隔离级别的**SQL支持**，这里需要关系到客户端会话初始时分配的锁信息。

综上，本功能的子模块主要划分为三部分：**(1)**事务信息的传递； **(2)**SELECT语句在UPS上的执行； **(3)**SQL支持。

4.2 总体设计思路

在请求传递的过程中，需要在请求中添加事务信息。显式事务的执行流程如下：

1 . 通过START TRANSACTION开启一个新的事务；
2 . 执行读操作（ SELECT ）或者写操作；
3 . 通过COMMIT显式地提交事务，或者通过ROLLBACK显式地回滚事务，或者事务执行失败隐式地回滚事务。

为了实现SNAPSHOT ISOLATION隔离级别，MS端在处理SELECT操作时，需要将事务信息传递给UPS，事务信息的获取及传递流程如下：

1 . MS端收到客户端的START TRANSACTION请求后，将该请求转发给UPS端。UPS收到START TRANSACTION请求后，会为其分配新的事务（包括事务描述符、事务开始时间戳），并将事务信息返回给MS端，MS将该事务信息保存至本地；
2 . MS端收到客户端的SELECT请求后，首先从本地获取该客户端对应的事务信息，然后查询需要访问的CS集合，将事务信息打包进数据请求中，发送给相应的CS。CS端接收到数据请求后，获得其中的事务标识符，并将其打包进增量数据的请求中，发送给UPS；
3 . UPS端收到读取请求时，可以获取该读取请求相关的事务信息，根据事务信息获取指定的版本数据并返回。

在UPS端，读取多行记录的流程类似于读取单行记录。因此我们在描述读取流程的时候，可以不用区分读取请求的类型。在读取请求中加入事务信息（ ObTransID ）后，读取的执行流程为：

1 . UPS在接收到读取请求后，反序列化该读取请求，并获得该请求中的事务信息ObTransID；
2 . 为读取请求分配新的只读会话（ ROSessionCtx ），该只读会话与请求读取的事务没有任何关系；
3 . 判断ObTransID中的隔离级别。如果隔离级别是SNAPSHOT ISOLATION，并且请求读取的事务的开始时间戳为非0，则将该只读会话的开始时间戳设置为ObTransID中的事务开始时间戳；否则认为隔离级别为READ COMMITTED，将该只读会话的开始时间戳设置为当前已公开的最大事务号（该事务号为单调递增的时间戳）；
4 . 访问内存表，获取相应的记录，并且遍历每一个记录的数据块链表，根据事务标识符和事务开始时间戳获取相应的数据块。

5 参考文献

[1] ANSI X3.135-1992. American National Standard for Information Systems – Database language – SQL. , 1992.

[2] Gray J., Helland P., O'Neil P, et al. The Dangers of Replication and a Solution. In: Proc. of SIGMOD, pp. 173–182, 1996.

[3] PostgreSQL 9.5.1 Documentation.