

Consistency Checking for Transactional Databases

Senior Engineer @ Tencent Database (TDSQL)

Yuxing Chen



Content



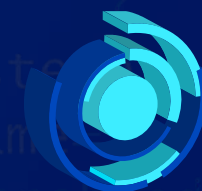
Background



Motivation



Solution



Summary

Background

TDSQL

Tencent Enterprise Distributed Database

Serve more than 500K enterprise customers

Applications cover billions of users





TDSQL

Tencent Enterprise Distributed Database

Serve more than 500K enterprise customers

Applications cover billions of users

Financial



Social
Network



Entertain
ment



Video



OLTP

OLAP

Enterprise
MySQL(CDB)

- MySQL Compatible
- Enhanced with enterprise-level features

TDSQL-C

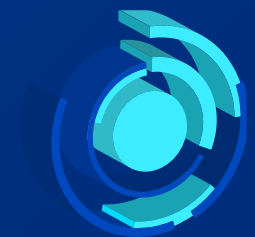
- Cloud native
- Shared storage
- MySQL and PG compatible

TDSQL

- Shared nothing
- MySQL compatible

TDSQL-A

- Shared nothing
- MPP + Column store
- Oracle Compatible



Isolation levels vs. Data anomalies

	P0:Dirty Write	P1:Dirty Read	P2:Non-repeatable Read	P3:Phantom
Read Uncommitted	Not Possible	Possible	Possible	Possible
Read Committed	Not Possible	Not Possible	Possible	Possible
Repeatable Read	Not Possible	Not Possible	Not Possible	Possible
Serializable	Not Possible	Not Possible	Not Possible	Not Possible

The stronger the levels, the less the anomalies

Anomaly occurrences
by ANSI SQL



Background

	<i>P0:Dirty Write</i>	<i>P1:Dirty Read</i>	<i>P4C:Cursor Lost Update</i>	<i>P4:Lost Update</i>	<i>P2:Non-repeatable Read</i>	<i>P3:Phantom</i>	<i>A5A: Read Skew</i>	<i>A5B: Write Skew</i>
<i>Read Uncommitted</i>	<i>Not Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>
<i>Read Committed</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>
<i>Cursor Stability</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>	<i>Possible</i>
<i>Repeatable Read</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>
<i>Snapshot Isolation</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Possible</i>	<i>Possible</i>
<i>Serializable</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>	<i>Not Possible</i>

More isolation levels and anomalies

Reference: Hal Berenson, Philip A. Bernstein, Jim Gray, Jim Melton, Elizabeth J. O'Neil, Patrick E. O'Neil: A Critique of ANSI SQL Isolation Levels. SIGMOD Conference 1995: 1-10

Motivation



Motivation

Non traditional isolation levels

MongoDB: Snapshot Isolation level

DB2: Cursor Stability level

TiDB: Optimistic level

Databricks: WriteSerializable level

SQL Server: Read Committed Snapshot Isolation level



Motivation

User:

1 Selection of a level for application, understanding of a new level:

want to know expected and unexpected anomalies in advance

2 Databases may not meet the level they claim or may not perform the same

Oracle claimed Serializable support but existing Write Skew

PostgreSQL does not allow Phantom but allow Write Skew at RR level

SQL Server does not allow Write Skew but allow Phantom at RR level



Motivation

Vendors:

1 Iteration developing yet regression test may be incomplete

PostgreSQL exists Write Skew from v9.1 to v12.3

2 How do we verify a newly developed database

Design flaw, program bugs, other optimization may yield unexpected anomalies.

Solutions



Current solutions

Jepsen/Elle

Pros:

- 1. Isolation verification*
- 2. Efficient checking*

Cons:

- 1. Random test and sometimes hard to construct SQL test cases*
- 2. No range queries, can not distinguish RR and SER levels*



Current solutions

Cobra

Pros:

- 1. test K-V serializability*
- 2. can catch the workload throughput*

Cons:

- 1. Random test and sometimes hard to construct SQL test cases*
- 2. No range queries, can not distinguish RR and SER levels*

Summary



Summary

TDSQL cares about the correctness of the databases.

Current tools of isolation verification still

Lack reproducibility and understandability

Lack predicate anomaly verifications

```

return
    ) {var c=b.nodeName.toLowerCase
function(d,filter.ID=
parentNode.disabled==a:b.disabled==a}}function pa(a){return
.lengthwhile(g--)c[e=f[g]]&&(c[e
.getElementsByName&&a}c=ga.support=
!!b&&"HTML"!==b.nodeName},m=ga
.nodeType&&g
.addEventListene?e.addEventListene
function(){return
.className="i",la
.querySelectorAll("enabled").length
(b){return"form"in
.disabled==a:bfunction(b){v
.querySelectorAll("enabled").length
){var c=b.nodeName.toLowerCase!!b&&"HTML"!==b.nodeName},m=ga
function(b){return"form"in b?b.parentNode
parentNode.disabled==a:b.disabled==a:b.isDisabled
.disabled
.length
.getElementsByName&&a}c=ga.support=

```

Thanks

Yuxing Chen
axingguchen@tencent.com

Reference:

Atul Adya, Barbara Liskov, Patrick E. O'Neil: Generalized Isolation Level Definitions. ICDE 2000: 67-78

Peter Alvaro, Kyle Kingsbury: Elle: Inferring Isolation Anomalies from Experimental Observations. Proc. VLDB Endow. 14(3): 268-280 (2020)

Cheng Tan, Changgeng Zhao, Shuai Mu, Michael Walfish: Cobra: Making Transactional Key-Value Stores Verifiably Serializable. OSDI 2020: 63-80