Advanced Database Systems

Assignment 4

A Critique of ANSI SQL Isolation Levels

16CS30044: Sarthak Chakraborty

1.
INTRODUCTION

Isolation Levels specified in ANSI/ISO SQL-92

Introduction to Isolation

- Isolation guards against inconsistent database state due to concurrent transactions

- **Phenomena:** Operation subsequences that may lead to anomaly
  [Defined in ANSI/ISO SQL-92 Specifications]
  - **Dirty Read:** Reading a data item that never existed
  - **Non-repeatable Read:** Rereading data item shows inconsistent values
  - **Phantom:** Rereading data based on <search condition> returns different set of item

Dirty Read

> ***Transaction T1 modifies a data item. Another transaction T2 then reads that data item before T1 performs a COMMIT or ROLLBACK. If T1 then performs a ROLLBACK, T2 has read a data item that was never committed and so never really existed.***

P1

write1(x)
. . .
read2(x)
. . .
((c1 or a1) and (c2 or a2)
in any order)

Statement does not insist that T1 aborts

Loose Interpretation

A1

write1(x)
. . .
read2(x)
. . .
(a1 and c2 in any order)

ACTUAL ANOMALY !!

Non-Repeatable Read

> ***Transaction T1 reads a data item. Another transaction T2 then modifies or deletes that data item and commits. If T1 then attempts to reread the data item, it receives a modified value or discovers that the data item has been deleted.***

P2

read1(x)
. . .
write2(x)
. . .
((c1 or a1) and (c2 or a2)
in any order)

A2

read1(x)
write2(x)
commit2
read1(x)
commit1

Phantom

> ***Transaction T1 reads a set of data items satisfying some <search condition>. Transaction T2 then creates data items that satisfy T1's <search condition> and commits. If T1 then repeats its read with the same <search condition>, it gets a set of data items different from the first read.***

P3

read1(P)
. . .
write2(y in P)
. . .
((c1 or a1) and (c2 or a2)
in any order)

A2

read1(P)
write2(y inP)
commit2
read1(P)
commit1

Isolation Levels in ANSI/ISO SQL-92

Table 1. ANSI SQL Isolation Levels Defined in terms of the Three Original Phenomena			
Isolation Level	**P1 (or A1) Dirty Read**	**P2 (or A2) Fuzzy Read**	**P3 (or A3) Phantom**
ANSI READ UNCOMMITTED	Possible	Possible	Possible
ANSI READ COMMITTED	Not Possible	Possible	Possible
ANSI REPEATABLE READ	Not Possible	Not Possible	Possible
ANOMALY SERIALIZABLE	Not Possible	Not Possible	Not Possible

Execution subsequences that disallow the three phenomena are not fully serializable !

2.

ANALYSIS

Analysis of ANSI SQL Isolation Levels

Dirty Write - A New Phenomena

> ***Transaction T1 modifies a data item. Another transaction T2 then further modifies that data item before T1 performs a COMMIT or ROLLBACK. If T1 or T2 then performs a ROLLBACK, it is unclear what the correct data value should be.***

Po write1(x)
. . .
write2(x)
. . .
((c1 or a1) and (c2 or a2)
in any order)

Violates Database Consistency

Only ANSI SERIALIZABLE isolation level excludes this anomalous behaviour

REMARK

> *ANSI SQL isolation should be modified to require P0 for all isolation levels*

Phenomena or Anomaly: Which is Correct?

read1[x=50] -> write1[x=10] -> read2[x=10] -> read2[y=50] -> commit2 -> read1[y=50] -> write1[y=90] -> commit1

T1 transfering 40 from x to y, maintaining x+y=100, but T2 reads inconsistent state where x+y=60

P1 VIOLATED !!

- **A1 not violated**
  No transaction aborts
- **A2 not violated**
  No rereading
- **A3 not violated**
  No relevant predicate change

Phenomena or Anomaly: Which is Correct?

read1[x=50] -> read2[x=50] -> write2[x=10] -> read2[y=50] -> write2[y=90] -> commit2 -> read1[y=90] -> commit1

P2 VIOLATED !!

read1[P] -> write2[insert y in P] -> read2[z] -> write2[z] -> commit2 -> read1[z] -> commit1

P3 VIOLATED !!

REMARK

> *Strict interpretations A1, A2, and A3 have unintended weaknesses. The correct interpretations are the Loose ones. We assume in what follows that ANSI meant to define P1, P2, and P3.*

Isolation Levels with Four Phenomena

Table 3. ANSI SQL Isolation Levels Defined in terms of the four phenomena

Isolation Level	P 0 Dirty Write	P 1 Dirty Read	P 2 Fuzzy Read	P 3 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

3.
CURSOR STABILITY

Other Isolation Levels

Lost Update Anomaly

> ***The lost update anomaly occurs when transaction T1 reads a data item and then T2 updates the data item (possibly based on a previous read), then T1 (based on its earlier read value) updates the data item and commits.***

P4

read1(x)
. . .
write2(x)
. . .
write1(x)
. . .
commit1

read1[x=100], read2[x=100], write2[x=120], commit2, write1[x=130], commit1

- Update of x=120 lost
- Possible at READ COMMITTED level
- Forbidding P0 and P1 allows the history
- Forbidding P2 precludes P4

Cursor Stability

- Fetch data from a cursor
- Lock the current data item of the cursor
- Lock is held till cursor moves or is closed, possibly by commit
- To update a row, hold write lock until transaction commits, even after the cursor moves on

REPEATABLE READ >>
CURSOR STABILITY >>
READ COMMITTED

Widely Implemented

4.
SNAPSHOT ISOLATION

Other Isolation Levels

Snapshot Isolation

- Transaction reads from the snapshot of the data
- Read is never blocked for a transaction running in Snapshot Isolation
- Updates by concurrent transaction are not visible to each other
- T1 can commit if there is no other concurrent transaction T2 wrote data item that T1 also wrote and committed. Otherwise T1 aborts.
  FIRST COMMITTER WINS
- Type of multiversion concurrency control

Data Item Constraint Violation Anomaly

Read Skew

> ***Suppose transaction T1 reads x, and then a second transaction T2 updates x and y to new values and commits. If now T1 reads y, it may see an inconsistent state, and therefore produce an inconsistent state as output.***

A5A

read1(x) . . write2(x) . . write2(y) . . commit2 . . read1(y) . . (c1 or a1)

Data Item Constraint Violation Anomaly

Write Skew

> ***Suppose T1 reads x and y, which are consistent with some constraint, and then a T2 reads x and y, writes x, and commits. Then T1 writes y. If there were a constraint between x and y, it might be violated.***

A5B

read1(x) . . read2(y) . . write1(y) . . write2(x) . . (c1 and c2 can occur)

Snapshot Isolation

- First-committer-wins prevents Dirty Writes (**P0**)
- Timestamp mechanism in SI prevents dirty reads (**P1**)
- Transaction cannot see updates of other transactions, prevents **A2**
- **A5A** not possible in SI but possible in READ COMMITTED
- **A5B** can occur in SI. Forbidding **P2** also forbids **A5B**. Thus **P2** occurs in SI which does not occur in REPEATABLE READ
- SI does not experience **A3** since transaction updates are not seen by others, but REPEATABLE READ can experience **A3**
- Snapshot Isolation histories prohibit histories with anomaly A3, but allow A5B, while REPEATABLE READ does the opposite.

Snapshot Isolation

REPEATABLE READ >> << SNAPSHOT ISOLATION >> READ COMMITTED

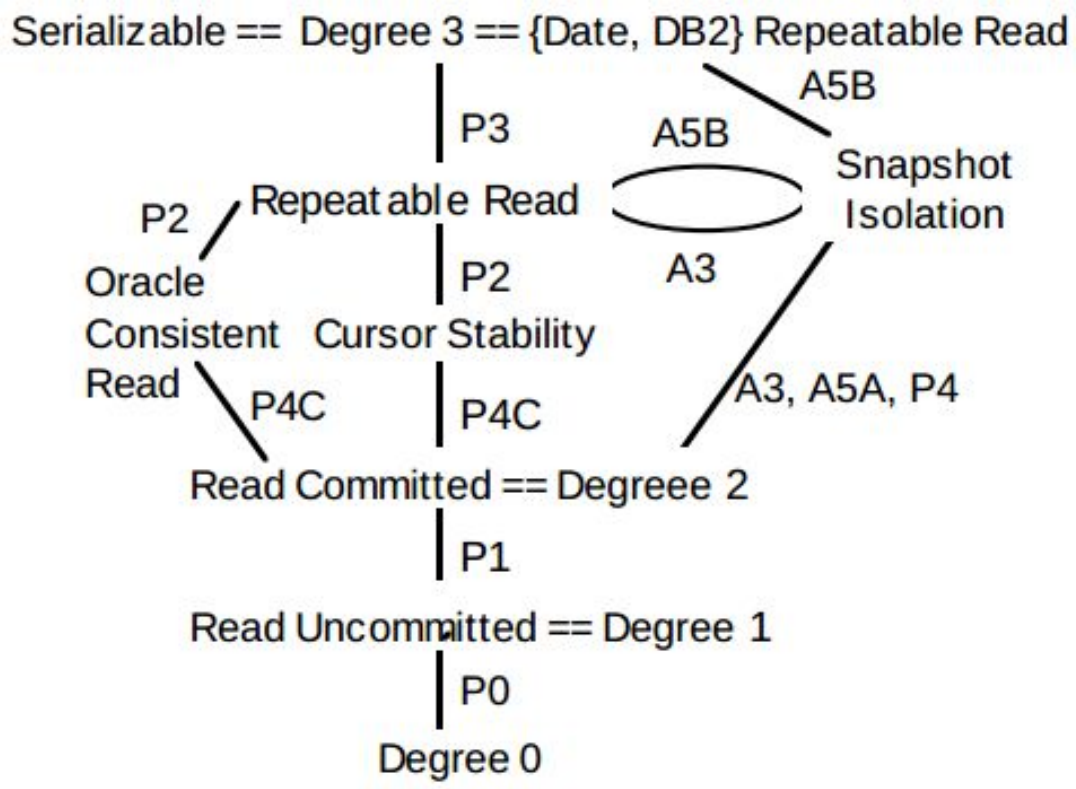
- Snapshot Isolation histories preclude anomalies **A1**, **A2** and **A3**. Therefore, in the anomaly interpretation of ANOMALY SERIALIZABLE

SNAPSHOT ISOLATION >> ANOMALY SERIALIZABLE

5.
CONCLUSION

Summary of the Analysis

Conclusion


Serializable == Degree 3 == {Date, DB2} Repeatable Read
A5B
P3
A5B
Snapshot
Isolation
P2
Repeatable Read
A3
Oracle
Consistent
Read
P2
Cursor Stability
P4C
P4C
A3, A5A, P4
Read Committed == Degreee 2
P1
Read Uncommitted == Degree 1
P0
Degree 0

Conclusion

Table 4. Isolation Types Characterized by Possible Anomalies Allowed.

Isolation level	P0 Dirty Write	P1 Dirty Read	P4C Cursor Lost Update	P4 Lost Update	P2 Fuzzy Read	P3 Phantom	A5A Read Skew	A5B Write Skew
READ UNCOMMITTED == Degree 1	Not Possible	Possible	Possible	Possible	Possible	Possible	Possible	Possible
READ COMMITTED == Degree 2	Not Possible	Not Possible	Possible	Possible	Possible	Possible	Possible	Possible
Cursor Stability	Not Possible	Not Possible	Not Possible	Sometimes Possible	Sometimes Possible	Possible	Possible	Sometimes Possible
REPEATABLE READ	Not Possible	Not Possible	Not Possible	Not Possible	Not Possible	Possible	Not Possible	Not Possible
Snapshot	Not Possible	Not Possible	Not Possible	Not Possible	Not Possible	Sometimes Possible	Not Possible	Possible
ANSI SQL SERIALIZABLE == Degree 3 == Repeatable Read Date, IBM, Tandem, ...	Not Possible	Not Possible	Not Possible	Not Possible	Not Possible	Not Possible	Not Possible	Not Possible

Thank You!