Update Stories, Phantoms, and Ghosts

Reutlingen, November 3, 2014

M. Laiho and D. A. Dervos

www.dbtechnet.org
Multi-granular locking scheme (MGL/LSCC)

- Sample variants of lock compatibility matrices

Lock granules:

database

(tablespace)

table

(extent)

page

row

Other locks on index ranges, schemas

1. Intent locks
   IS for S on row
   IX for X on row

2. Lock on row

SIX = S + IX

Shared locks (S) allow reading.
Exclusive locks (X) allow writing and are kept up to end of transaction eliminating lost updates.

Martti Laiho
Multi-granular locking scheme (MGL/LSCC)

- Sample variants of lock compatibility matrices

**Lock granules:**

- database
- (tablespace)
- table
- (extent)
- page
- row

1. Intent locks
   - IS for S on row
   - IX for X on row

2. Lock on row

**Other locks on index ranges, schemas**

**Shared locks (S)** allow reading.

**eXclusive locks (X)** allow writing and are kept up to end of transaction eliminating lost updates.

Martti Laiho
SQL Update Stories
MVCC: in simple terms

- The DB server makes use of timestamps to maintain a history chain of transaction (begin) and updated row instances
- Considering the above two, any one transaction may either read the latest committed version of each one row at read time (READ COMMITTED), or the latest committed version of each one row at its (transaction) begin time (SNAPSHOT)
- Consequently, readers and writers do not block each other: improved performance
On latest committed... (two possibilities)

A
B
C
On latest committed... (two possibilities)

\[
\begin{align*}
A \\
B \\
C \\
A' \\
t
\end{align*}
\]
On latest committed... (two possibilities)

\[ \begin{array}{c}
  A \\
  B \\
  C \\
  \vdots
\end{array} \]

\[ \begin{array}{c}
  A' \\
  B' \\
  C'
\end{array} \]
On latest committed... (two possibilities)

\[ X_{\text{action begins}} \]

\[ t \]

\[ A \]
\[ B \]
\[ C \]

\[ A' \]
\[ B' \]
\[ C' \]
On latest committed... (two possibilities)

- Xaction begins
- Reads (A, B, C)

A
B
C

A'
B'
C'
On latest committed... (two possibilities)

- A
- B
- C

<table>
<thead>
<tr>
<th>READ COMMITTED</th>
<th>SNAPSHOT ISOLATION</th>
</tr>
</thead>
</table>

- Xaction begins
- Reads (A, B, C)

A'
B'
C'
On latest committed... (two possibilities)

- Xaction begins
- Reads (A, B, C) → (A', B', C)
- A'  B'  C'

t
On latest committed... (two possibilities)

- Xaction begins
- Reads (A,B,C) -> (A',B',C) -> (A',B,C)

![Diagram showing transaction phases and read consistency](image)
MVCC implementation: Oracle

atomic update action:

1. copy the row
2. "lock" the row
3. update the row...
   ... Commit

Table

Temporary space

latest committed row version

original row location on a page of the table

scn: system change number

Martti Laiho
e.g. in MySQL/InnoDB:

- **READ UNCOMMITTED**: MVCC (read latest written)
- **READ COMMITTED**: MVCC (read latest committed)
- **REPEATABLE READ**: MVCC (snapshot isolation)
- **SERIALIZABLE**: MGL (long S- /X- locks plus intention locks, w. MV in the background)
MV-based Concurrency Control

e.g. in MySQL/InnoDB:

- **READ UNCOMMITTED**: MVCC (*read latest written*)
- **READ COMMITTED**: MVCC (*read latest committed*)
- **REPEATABLE READ**: MVCC (*snapshot isolation*)
- **SERIALIZABLE**: MGL (*long S- /X- locks plus intention locks, w. MV in the background*)

...still: room for further investigation (learn by verifying)

Martti Laiho
Phantoms and ghosts under SI

- **Phantoms**: rows inserted (or updated) by other transactions, following the current transaction's ‘begin’ time. They exist physically, yet they are not visible (as such) under SI.

- **Ghosts**: rows deleted by other transactions, following the current transaction's begin time (stamp). They do not exist physically, yet they appear to be present under SI (in a 'see but do not touch/update' fashion).
Phantoms and Ghosts
under MySQL/InnoDB's MVCC
Don't believe all that you read! For developing reliable applications, you need to experiment and verify yourself the services of your DBMS product. DBMS products differ in the way they support even the basics of SQL transaction services.

Martti Laiho
Thank You