CSC 261/461 – Database Systems
Lecture 26

Spring 2018
Announcement

• Term Paper:
  – We will not have a poster session this semester
  – Students need to present their work before the class
  – You can either create PowerPoint Slides or present the poster as it is.
  – When: One week from Today (April 30)
  – Duration: 12 minutes (We would also like to see demo of your findings) + 3 minutes to set up.
  – Wonderful opportunity to practice public speaking!

• Attendance:
  – All the students are expected to give (constructive) feedback.
  – Feedback will be shared anonymously with the presenters
  – You will get class participation points for your feedback
  – Bonus: You can use your laptop!

CSC261, Spring 2018, UR
Project 4 (Optional)

- Is out
- Try only if you want!
- I will share a few Notebooks
- Mainly covered during workshops
Transaction Characteristics in SQL

• SQL allows to specify three (3) characteristics of a transaction
  – Access mode
    • READ ONLY and READ WRITE
  – Diagnostics size
    • (Determines # of error conditions.)
  – Isolation level
    • Controls the extent to which a given transaction is exposed to actions of other transactions
Transaction Characteristics in SQL

Unrepeatable Read

Dirty read

Phantom read
## Transaction Characteristics in SQL

### Isolation Levels and Characteristics

<table>
<thead>
<tr>
<th>Isolation Level</th>
<th>Dirty Read</th>
<th>Unrepeatable Read</th>
<th>Phantom</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ UNCOMMITTED</td>
<td>Maybe</td>
<td>Maybe</td>
<td>Maybe</td>
</tr>
<tr>
<td>READ COMMITED</td>
<td>No</td>
<td>Maybe</td>
<td>Maybe</td>
</tr>
<tr>
<td>REPEATABLE READ</td>
<td>No</td>
<td>No</td>
<td>Maybe</td>
</tr>
<tr>
<td>SERIALIZABLE</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

**Example:** SET TRANSACTION ISOLATION LEVEL SERIALIZABLE
The recovery manager of a DBMS is responsible for ensuring transaction atomicity and durability

- Atomicity by undoing the actions of transitions that do not commit
- Durability by making sure that all actions of committed transactions survive system crashes.

The transaction manager of a DBMS controls the execution of transactions.

- Before reading and writing objects during normal execution, locks must be acquired (and released at some later time) according to a chosen locking protocol.
Stealing Frames and Forcing Pages

• Writing objects rises two important questions:

  – Can changes to an Object O made by a Transaction T be written to disk before T commits?
    • 'steal' the page.
    • Steal approach

  – When a transaction commits, must we ensure that all changes it has made to objects are immediately forced to disk?
    • If yes, we call that a force approach.
      • Easy: No Steal, force approach
      • But have drawbacks
Drawbacks

• **No-steal** assumes all pages modified by ongoing transactions can be accommodated in the buffer pool

• **Force** approach results in excessive page I/O

• Most system uses a **steal, no force** approach.
  – Allows writing dirty frames to disk
  – Do not enforce immediate writing back after commit.
Who handles recovery?

• Recovery Manager
• Handles:
  – Atomicity
    • By undoing actions that do not commit
  – Durability
    • Making sure committed transactions survive system crashes
Solution?

- **WAL** (Write-Ahead Log)
- Enables the recovery manager to:
  - **Undo** the actions of aborted and incomplete transactions
  - **Redo** the actions of committed transactions.
- **Example:**
  - Changes of transactions that did not commit prior to crash might have written to the disk (due to steal approach)
    - Changes can be identified from the log and undone.
  - A transactions that committed before the crash may have updates not written to the disk. (due to no-force)
    - Changes can be identified from the log and written to disk

Example of Recovery Algorithm: ARIES
Overview of ARIES

• Algorithms for Recovery and Isolation Exploiting Semantics, or ARIES
  – A recovery algorithm designed to work with a no-force, steal database approach
  – Used by IBM DB2, Microsoft SQL Server and many other database systems
Three main principles lie behind ARIES:

- **Write-ahead logging:**
  - Any change to an object is first recorded in the log, and the log must be written to stable storage before changes to the object are written to disk.

- **Redo:**
  - On restart after a crash, ARIES retraces the actions of a database before the crash and brings the system back to the exact state that it was in before the crash. Then it undoes the transactions still active at crash.

- **Undo:**
  - Changes made to the database while undoing transactions are logged to ensure such an action isn't repeated in the event of repeated restarts.

Acknowledgement

• Some of the slides in this presentation are taken from the slides provided by the authors.
• Many of these slides are taken from cs145 course offered by Stanford University.

• https://vladmihalcea.com/2014/01/05/a-beginners-guide-to-acid-and-database-transactions/

• Spark slides and material: Jonathan Carroll-Nellenback (CIRC, UofR)