

ICS 321 Fall 2011

Introduction to Database Systems

Asst. Prof. Lipyeow Lim
Information & Computer Science Department
University of Hawaii at Manoa

Group Discussion

- How would you implement an internet store without using any database software ?
 - List all the questions, challenges or difficulties you encounter as you discuss the high level steps.

Group Discussion Results

Data, Database, DBMS

- A **database** : a collection of related data.
 - Represents some aspect of the real world (aka universe of discourse).
 - Logically coherent collection of data
 - Designed and built for specific purpose
- **Data** are known facts that can be recorded and that have implicit meaning.
- A **data model** is a collection of concepts for describing data.
- A **schema** is a description of a particular collection of data, using the a given data model.

DBMS

- A **database management system (DBMS)** is a collection of programs that enables users to
 - **Create** new DBs and specify the structure using data definition language (DDL)
 - **Query** data using a query language or data manipulation language (DML)
 - **Store** very large amounts of data
 - Support **durability** in the face of failures, errors, misuse
 - Control **concurrent** access to data from many users

Types of Databases

- On-line Transaction Processing (**OLTP**)
 - Banking
 - Airline reservations
 - Corporate records
- On-line Analytical Processing (**OLAP**)
 - Data warehouses, data marts
 - Business intelligence (BI)
- Specialized databases
 - Multimedia
 - XML
 - Geographical Information Systems (GIS)
 - Real-time databases (telecom industry)
- Special Applications
 - Customer Relationship Management (CRM)
 - Enterprise Resource Planning (ERP)
- Hosted DB Services
 - Amazon, Salesforce

A Bit of History

- 1970 **Edgar F Codd** (aka “Ted”) invented the **relational model** in the seminal paper “A Relational Model of Data for Large Shared Data Banks”
 - Main concept: relation = a table with rows and columns.
 - Every relation has a schema, which describes the columns.
- Prior 1970, no standard data model.
 - Network model used by Codasyl
 - Hierarchical model used by IMS
- After 1970, IBM built System R as proof-of-concept for relational model and used **SQL** as the query language. SQL eventually became a standard.

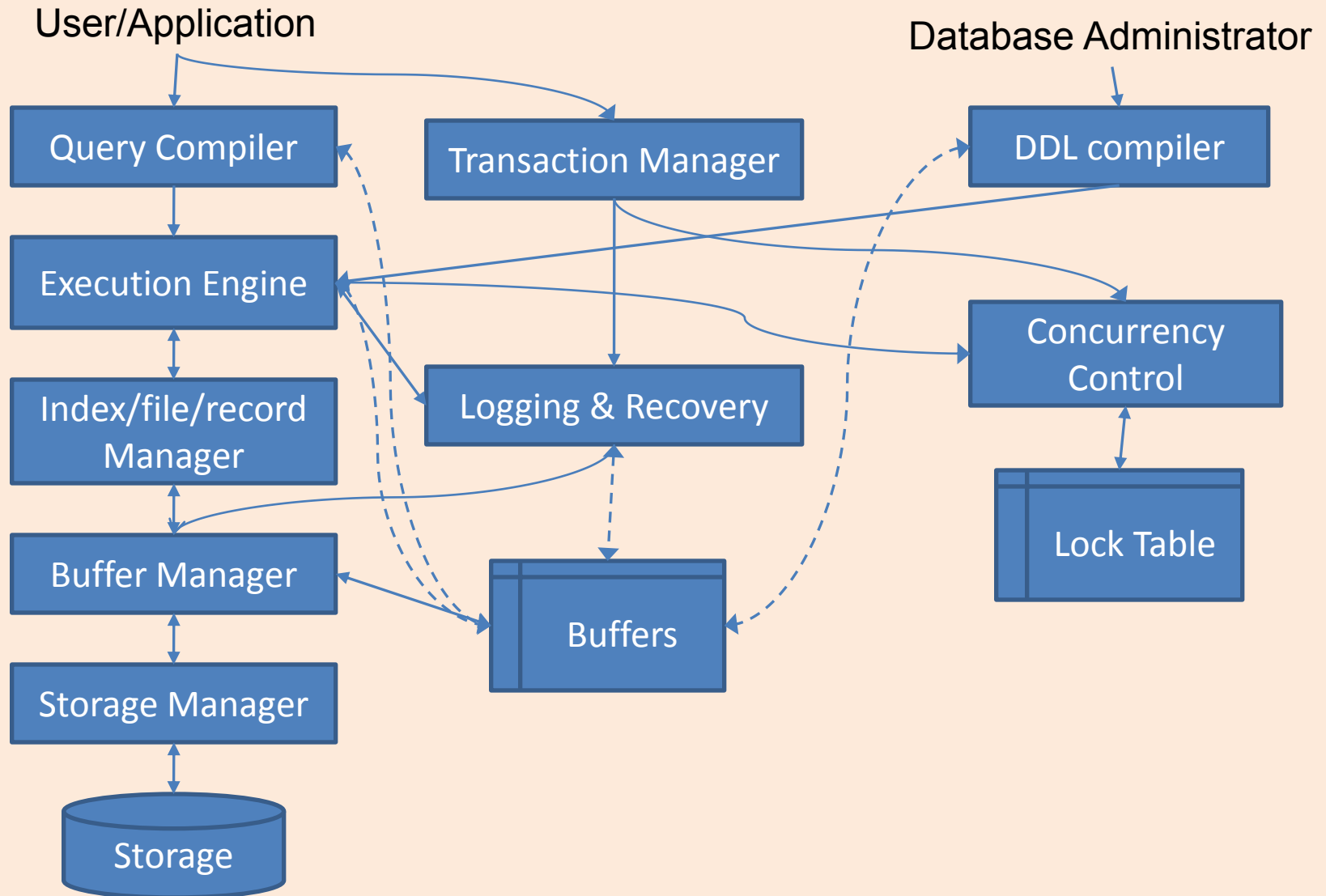
Files vs DBMS

- Swapping data between memory and files
- Difficult to add records to files
- Security & access control
- Do optimization manually
- Good for small data/files
- Run out of pointers (32bit)
- Code your own search algorithm
 - Search on different fields is difficult
- Must protect data from inconsistency due to concurrency
- Fault tolerance – crash recovery

Why use a DBMS ?

- Large datasets
- Concurrency/ multi-user
- Crash recovery
- Declarative query language
- No need to figure out what low level data structure
- Data independence and efficient access.
- Reduced application development time.
- Data integrity and security.
- Uniform data administration.

DBMS Components



Transaction: An Execution of a DB Program

- A transaction an *atomic* sequence of database actions (reads/writes).
- Each transaction, executed completely, must leave the DB in a consistent state if DB is consistent when the transaction begins.
 - Users can specify some simple integrity constraints on the data, and the DBMS will enforce these constraints.
 - Beyond this, the DBMS does not really understand the semantics of the data. (e.g., it does not understand how the interest on a bank account is computed).
 - Thus, ensuring that a transaction (run alone) preserves consistency is ultimately the *user's* responsibility!

Concurrency Control

- Concurrent execution of user programs is essential for good DBMS performance.
 - Because disk accesses are frequent, and relatively slow, it is important to keep the cpu humming by working on several user programs concurrently.
- Interleaving actions of different user programs can lead to inconsistency: e.g., check is cleared while account balance is being computed.
- DBMS ensures such problems don't arise: users can pretend they are using a single-user system.

ACID Properties

- **Atomicity** : all-or-nothing execution of transactions
- **Consistency**: constraints on data elements is preserved
- **Isolation**: each transaction executes as if no other transaction is executing concurrently
- **Durability**: effect of an executed transaction must never be lost

Ensuring Isolation

- Scheduling concurrent transactions
- DBMS ensures that execution of $\{T_1, \dots, T_n\}$ is equivalent to some serial execution $T_1' \dots T_n'$.
 - Before reading/writing an object, a transaction requests a lock on the object, and waits till the DBMS gives it the lock. All locks are released at the end of the transaction. (Strict 2PL locking protocol.)
 - **Idea:** If an action of T_i (say, writing X) affects T_j (which perhaps reads X), one of them, say T_i , will obtain the lock on X first and T_j is forced to wait until T_i completes; this effectively orders the transactions.
 - What if T_j already has a lock on Y and T_i later requests a lock on Y ? (Deadlock!) T_i or T_j is aborted and restarted!

Ensuring Atomicity

- DBMS ensures *atomicity* (all-or-nothing property) even if system crashes in the middle of a Xact.
- **Idea:** Keep a log (history) of all actions carried out by the DBMS while executing a set of Xacts:
 - **Before** a change is made to the database, the corresponding log entry is forced to a safe location. (WAL protocol; OS support for this is often inadequate.)
 - After a crash, the effects of partially executed transactions are undone using the log. (Thanks to WAL, if log entry wasn't saved before the crash, corresponding change was not applied to database!)

The Log

- The following actions are recorded in the log:
 - *Ti writes an object*: The old value and the new value.
 - Log record must go to disk before the changed page!
 - *Ti commits/aborts*: A log record indicating this action.
- Log records chained together by Xact id → easy to undo a specific Xact (e.g., to resolve a deadlock).
- Log is often *duplexed* and *archived* on “stable” storage.
- All log related activities (in fact, all CC related activities such as lock/unlock, dealing with deadlocks etc.) are handled transparently by DBMS.

Summary

- Definitions of data, databases, data models, schema
- When to use or not use a DBMS
- DBMS major components
- Transactions and concurrency
- ACID properties of transactions
- Techniques for ensuring ACID properties in DBMSs.