Chapter 1

Introduction

Slides by Silberschatz, Modifications by Rogers and Brown

DBMS – Database Management Systems

- Contain information about a particular enterprise
  - Collection of interrelated data
  - Set of programs to access the data
  - An environment that is convenient and efficient to use

DBMS – Database Management Systems

- Database Applications
  - Banking
    - Keeping track of transactions
  - Airlines
    - Scheduling and Reservations
  - Universities
    - Registration and Grading
  - Sales
    - Customers, Products and Purchases
  - Manufacturing
    - Production, Inventory, Ordering, Supply Chain tracking

University Database Example

- Going to be revisited for most of the semester
- Application program examples
  - Add new students, instructors and courses
  - Register students for courses and generate class rosters
  - Assign grades to students
  - Compute student grade point averages
  - Generate transcripts for alumni
- In early days, database applications were built directly on top of file systems

Drawbacks of Using File System to Store Data

- Data Redundancy and Inconsistency
  - Multiple file formats
  - Duplication of information across multiple files
- Difficulty in Accessing Data
  - Need to write a new computer program to accomplish each new task
- Data Isolation
  - Multiple files and formats
- Security
  - Hard to provide a user to access to some, but not all, data

Drawbacks of Using File System to Store Data

- Integrity Problems
  - Integrity constraints (e.g. account balance > 0) become buried in program code rather than stated explicitly
  - Hard to add new constraints or change existing ones
- Atomicity of updates
  - Failures may leave database in an inconsistent state with partial updates carried out
  - Ex : Transfer of funds from one account should either complete or not happen at all

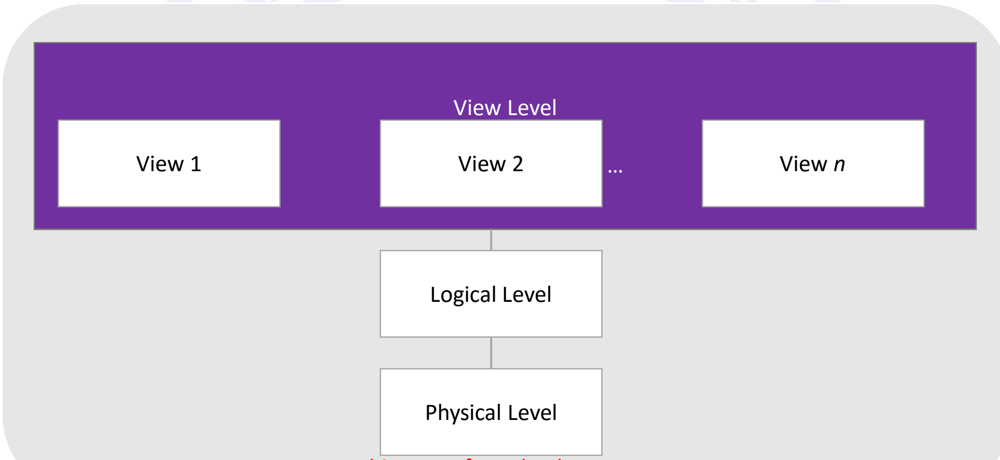
Drawbacks of Using File System to Store Data

- Concurrent Access by Multiple Users
  - Concurrent Access is needed for performance
  - Uncontrolled concurrent access can lead to inconsistencies
    - Ex. Two people reading a balance (say $100) and each withdrawing money (say $55) at the same time
- Database Management Systems offer solutions to all of these problems

Levels of Abstraction

- Physical Level : Describes how a record (e.g. a student) is stored on the file system
- Logical Level : Describes how data is stored in the database and the relationships among the different data points
  - E.g.
```
type instructor = record
    ID : string;
    name : string;
    dept_name : string;
    salary : string;
end;
```
- View Level : application programs hide details of data types.
  - Views can also hide information such as salary for security purposes

View of Data



Architecture for a database system

Instances and Schemas

- Similar to types and variables in programming languages
- Schema – The logical structure of the database
  - Ex. The database consists of information about a set of customers and accounts and the relationships between them
  - Analogous to type information of a variable in a program
  - Physical Schema : database design at the physical level
  - Logical Schema : database design at the logical level
- Instance – The actual content of the database at a particular point in time
  - Analogous to the value of a variable

Instances and Schemas

- Physical Data Independence
  - The ability to modify the physical schema without changing the logical schema
    - Applications depend on the logical schema
    - In general, interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others

Data Models

- A collection of tools for describing
  - Data
  - Data Relationships
  - Data Semantics
  - Data Constraints
- Relational Model
- Entity – Relationship Model (mainly used in Database design)
- Object Based Model (Object oriented and Object relational)
- Semi-Structured Model (XML)
- Older Models – Network and Hierarchical

Relational Model

- Will cover in detail in Chapter 2
- Example of tabular data in a relational model

Columns

Rows

ID	Name	Dept_Name	Salary
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
14061	Mozart	Music	59000
42305	Buchholz	Computer Science	78000
49675	Hemmingway	English	73000
85720	Crick	Biology	72000

The *Instructor* Table

A Sample Relational Database

ID	Name	Dept_Name	Salary
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
42305	Buchholz	Computer Science	78000
14061	Mozart	Music	59000
49675	Hemmingway	English	73000
85720	Crick	Biology	72000

The *Instructor* Table

Dept_Name	Building	Budget
Computer Science	Shannon	100000
Biology	Curie	90000
Music	Van Halen	80000
Finance	Wang	130000
Physics	Watson	90000

The *Department* Table

Data Manipulation Language (DML)

- Language for accessing and manipulating the data organized by the appropriate data model
  - DML is also known as 'Query Language'
- Two typical classes of languages
  - Procedural : user specifies what data is required and how to get to the data
  - Declarative : user specifies what data is required without specifying how to get to the data
- SQL is the most widely used query language

Data Definition Language (DDL)

- Specification notation used for defining the structure of the database
  - Ex.

```
create table instructor (
        ID              char(5),
        name            varchar (20),
        dept_name       varchar (20),
        salary          numeric (8,2))
```

- The DDL compiler generates a set of tables stored in a data directory

Data Definition Language (DDL)

- Data Dictionary contains metadata (i.e. data about data)
  - Database Schema
  - Integrity Constraints
    - Primary Key (ID that uniquely identifies information)
    - Referential Integrity (references constraints in DML)
      - Ex. : dept_name value in any instructor tuple must appear in a department relation
    - Authorization

Structured Query Language (SQL)

- Most widely used non-procedural language
  - Ex. : find the name of the instructor with ID = 42305

```
select name
from instructor
where instructor.ID = '42305'
```

  - What is this doing?

```
select instructor.ID, department.dept_name
from instructor, department
where instructor.dept_name =
     department.dept_name and
     department.budget > 95000
```

Structured Query Language (SQL)

- Application programs generally access databases through one of
  - Language extensions to allow embedded SQL
  - Application program interfaces (e.g. ODBC / JDBC) which allow SQL queries to be sent directly to the database
- Will be covered in Chapters 3 – 5

Database Design

- The process of designing the general structure of the database
  - Logical design
    - Deciding on the database schema
    - Database design requires that we find a “good” collection of relation schemas
      - Is a business decision – what attributes should be recorded in the database?
      - Is a computer science decision – what relational schemas should we have and how should the attributes be distributed among the various relational schemas
  - Physical design
    - Deciding on the physical layout of the database

Database Design

- What, if any, problems exist in the following design?

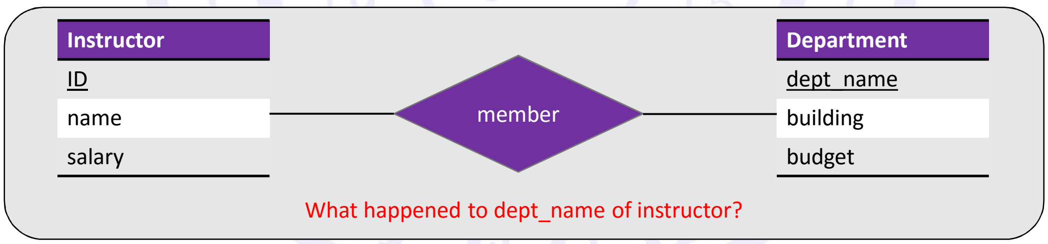
ID	Name	Salary	Dept_Name	Building	Budget
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Wang	120000
32343	El Said	60000	History	Painter	50000
98345	Cook	75000	Comp Sci	Shannon	100000
10101	Srinivasan	65000	Comp Sci	Shannon	100000
14061	Mozart	59000	Music	Van Halen	80000
85720	Crick	72000	Biology	Curie	90000
33454	Gold	87000	Chemistry	Watson	85000

Design Approaches

- Normalization Theory – Chapter 8
  - Formalize what designs are bad and test for them
- Entity Relationship Model – Chapter 7
  - Models an enterprise as a collection of entities and relationships
    - Entity is a thing or object in the enterprise that is distinguishable from other objects
      - Are described by a set of attributes
    - Relationships are an association among several entities
  - Represented diagrammatically by an entity-relationship diagram

The Entity-Relationship Model

- Models an enterprise as a collection of entities and relationships
  - An entity is a thing or object in the enterprise that is distinguishable from all other objects in the enterprise
    - An entity is described by a set of attributes
  - A relationship is an association among several entities
- Can be represented diagrammatically by an entity-relationship diagram



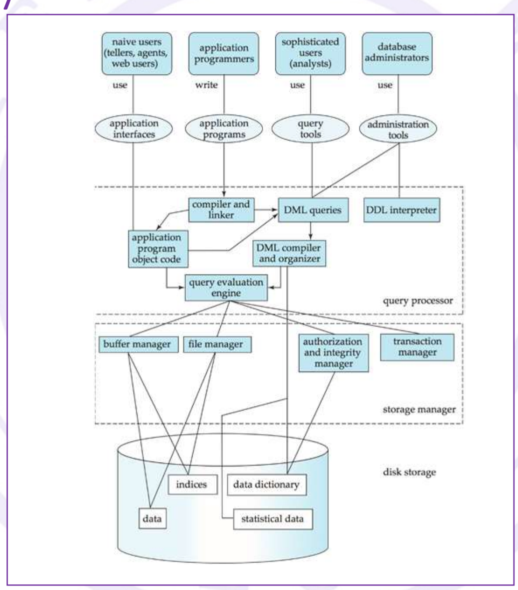
Object-Relational Data Models

- Relational model: flat, “atomic” values
  - Atomic data is indivisible and irreducible
  - Atomic transactions ensure that a transaction either occurs or is discarded
- Object-Relational Data Models
  - Extend the relational data model by including object orientation and constructs to deal with added data values
  - Allow attributes of tuples to have complex types, including non-atomic values such as nested relations
  - Preserve relational foundations, in particular the declarative access to data, while extending modeling power
  - Provide upward compatibility with existing relational languages

XML : Extensible Markup Language

- Defined by the WWW Consortium (W3C)
- Originally intended as a document markup language not a database language
- Has the ability to specify new tags, and create nested tag structures making it a great way to exchange data
- Has become the basis for all new generation data interchange formats
- Wide variety of tools exist for parsing, browsing and querying XML documents and data

Database System Internals

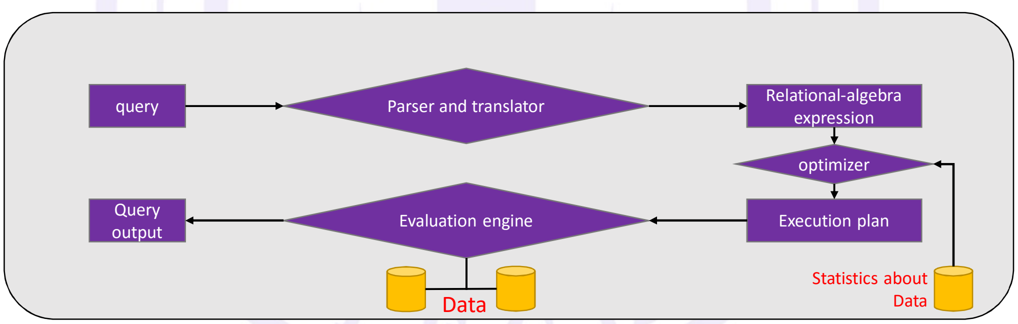


Storage Management

- Storage Manager is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system
- Is responsible for the following tasks
  - Interaction with the file manager
  - Efficient storing, retrieving and updating of data
- Issues
  - Storage Access
  - File Organization
  - Indexing and Hashing

Query Processing

- Parsing and Translation
- Optimization
- Evaluation



Query Processing

- Alternative ways of evaluating a given query
  - Equivalent expressions
  - Different algorithms for each operation
- Cost difference between a good and a bad way of evaluating a query can be enormous
- Need to translate the cost of operations
  - Depends critically on statistical information about relations which the database must maintain
  - Need to estimate statistics for intermediate results to compute cost of complex expressions

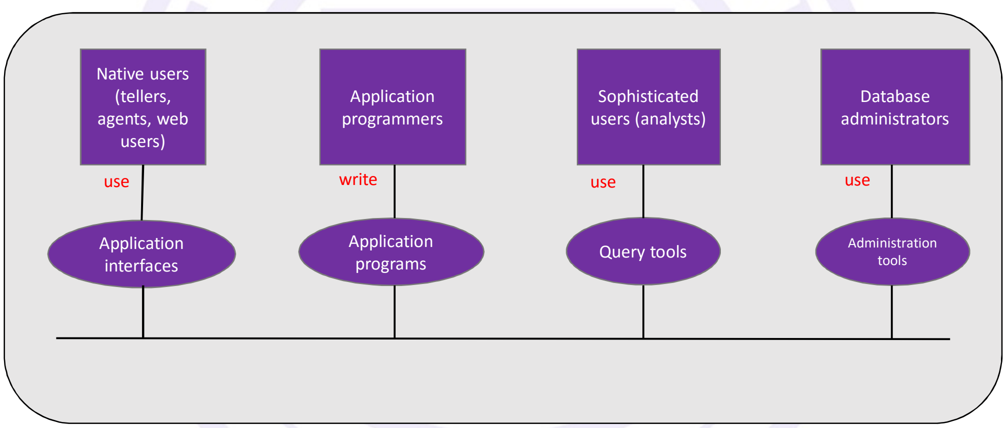
Transaction Management

- What if the system fails?
- What if more than one use is concurrently updating the same data?
- A transaction is a collection of operations that performs a single logical function in a database operation
- Transaction-management component ensures that the database remains in a consistent (correct) state despite system failures (e.g. power failures and operating system crashes) and transaction failures
- Concurrency-control manager controls the interaction among the concurrent transactions, to ensure the consistency of the database

Database Architecture

- The architecture of a database system is greatly influenced by the underlying computer system on which the database is running
  - Centralized
  - Client – Server
  - Parallel (multi-processor)
  - Distributed (multi-machine)

Database Users and Administrators



History of Database Systems

- 1950's and early 1960's
  - Data processing using magnetic tapes for storage
    - Tapes provide only sequential access
  - Punch cards used for input
- Late 1960's and 1970's
  - Hard disks allow direct access to data
  - Network and Hierarchical data models in widespread use
  - High-Performance (for the era) transaction processing

History of Database Systems

- Late 1960's and 1970's (cont)
  - Ted Codd defines the relational data model
    - Would win the ACM Turing award for his work
    - Is a mathematical model
    - IBM Research begins work on System R Prototype
    - UC-Berkeley begins Ingres Prototype
- 1980's
  - Research relational prototypes evolve to commercial systems
    - SQL becomes the national standard
  - Parallel and distributed database systems
  - Object-oriented database systems

History of Database Systems

- 1990’s
  - Large decision support and data-mining applications
  - Large multi-terabyte data warehouses
  - Emergence of Web commerce (Web 1.0)
- Early 2000’s
  - XML and XQuery standards
  - Automated database administration

History of Database Systems

- Later 2000’s to 2010’s
  - Giant data storage systems
    - Google Big Table
    - Yahoo Pnuts
    - Amazon Azure
    - …

End of Chapter 1