

Database Systems

Jan Chomicki
University at Buffalo

Plan of the course

- 1 Database Management Systems
- 2 Relational data model
- 3 Indexing
- 4 Query processing and optimization
- 5 Database design
- 6 Selected issues in contemporary DBMS
- 7 Transaction processing

What is a database

Database

- a computer representation of the information needed for an application
- persistent, long-lived
- typically shared by many users and programs

Example applications

- e-commerce
- human resources, payroll
- medical records
- content management
- public utilities and services

Database management system (DBMS)

DBMS

- a software system that makes it possible to build, modify and efficiently access large databases
- supports one or more **data models**
- supports multiple languages and interfaces for **data definition, manipulation, and querying**
- application-independent

Additional DBMS functions

- integrity maintenance
- concurrency control
- resiliency
- access control
- meta-data
- support for distribution, heterogeneity, multimedia, Web access,...

Common data model

Data model

- a view of data *shared* by the programs and the users interacting with a database
- a mathematical *abstraction*
- supported by the DBMS

Basic data model constructs

Relational	Relations
Object-oriented	Objects, classes, attributes,...
XML	Labelled trees
Graph	Nodes, edges
NoSQL	Key-value pairs

Schema vs. instance

Schema

- captures and describes the structure of the data
- time-independent

FLIGHT

NUMBER	AIRLINE
--------	---------

Instance

- captures the current state of the data
- time-dependent
- only one exists at any given time

FLIGHT

NUMBER	AIRLINE
72	Delta
97	Delta
1210	United

Schema concepts

- relation schemas
- attributes
- integrity constraints
- views, triggers, procedures,...

Instance concepts

- relation instances
- tuples
- attribute values

XML

In XML, data is **self-describing**, so schema is not required.

```
<flight>
  <number>72</number>
  <airline>Delta</airline>
</flight>
<flight>
  <airline>Delta</airline>
  <number>97</number>
  <status>canceled</status>
</flight>
```

But even in XML schema can be useful:

- data validation
- query optimization

DBMS languages

Data definition (DDL)

- define/modify database schemas

Data manipulation (DML)

- create and update instances
- various kinds of updates:
 - ▶ incremental: **insert (1001,American) into FLIGHT**
 - ▶ bulk: **copy all Continental flights to United flights**
- transactions

Query languages

- retrieve information from database instances

Query languages

Simple lookup queries

```
SELECT FLIGHT.AIRLINE
FROM FLIGHT
WHERE FLIGHT.NUMBER=72
```

Complex queries (join)

```
SELECT TERMINAL.TNAME
FROM FLIGHT, TERMINAL
WHERE FLIGHT.NUMBER=72
AND FLIGHT.AIRLINE=TERMINAL.AIRLINE
```

Complex queries (aggregation)

```
SELECT AIRLINE, COUNT(NUMBER)
FROM FLIGHT
GROUP BY AIRLINE
```

Storage (disk)

- large capacity
- persistent
- random access
- blocked
- buffering

Key-based access

- primary keys (uniquely identifying):

```
SELECT TERMINAL.TNAME
FROM TERMINAL
WHERE TERMINAL.AIRLINE='Delta'
```
- secondary keys:

```
SELECT FLIGHT.NUMBER
FROM FLIGHT
WHERE FLIGHT.AIRLINE='Delta'
```

Levels of abstraction

Conceptual database

- the global view of the whole application
- uses a high-level data model, e.g., relational
- example: university database
- created/maintained by the database administrator

External database

- views of users and application programs, selective
- uses a high-level data model: relational, XML
- examples: parking, registration, portals

Physical (internal) database

- how the database is actually stored on disk(s)
- hidden from the users
- uses lower-level concepts: files, indexes,...
- created/maintained by the database administrator using high-level tools

Query optimization

A DBMS can evaluate a query in many different ways.

An efficient way (if one exists) is picked by the **query optimizer** module when the query is submitted, based on:

- physical schema
- available special-purpose algorithms
- cost analysis

Transactions

Transaction properties

- **A**tomicity: all-or-nothing execution
- **C**onsistency: database consistency is preserved
- **I**solation: concurrently executing transactions have no effect on one another
- **D**urability: results survive failures.

Major DBMS components

Relational query processor

- query parsing and authorization
- query rewrite
- query optimizer
- plan executor
- DDL and utility processing

Transactional storage manager

- access methods
- buffer manager
- lock manager
- log manager

Client communications manager

Architecture

- local/remote client protocols
- two-tier: client-server
- three-tier: a middle tier between client and server

Functions

- authentication
- connection state handling
- forwarding client requests
- shipping the results back

Other components

Process manager

- allocating/deallocating *DBMS workers* to clients
- processes and threads:
 - ▶ OS process per DBMS worker
 - ▶ thread per DBMS worker
 - ▶ process pool

Shared components and utilities

- catalog manager
- memory manager
- ...