What is a database

- a computer representation of the information relevant to an application.
- persistent.
- typically shared by many users and programs.

Examples:

- library database.
- credit record database.
- utilities database.
- e-commerce site database.

Textbook, chapter 1

Database management system (DBMS)

- a software system that makes it possible to build and efficiently access large databases.
- provides a single *data model*.
- supports multiple languages and interfaces for *data definition*, *manipulation*, and *querying*.

Additional functions of a DBMS:

- consistency checking.
- concurrency control.
- resiliency.
- access control.
- meta-data.
- support for distribution, heterogeneity, multimedia, Web access, data analysis,...

Common data model

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

Data model	*Basic notions*
Relational	Relations
Object-oriented	Objects, classes, attributes,...
XML	Labelled trees (graphs)
Entity-Relationship	Entities, relationships,...

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
82	Delta
1210	American

Relational data model

Schema concepts:

- relation schemas
- attributes
- integrity constraints

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>82</number>
  <status>canceled</status>
</flight>
```

Still, even in XML schema can be useful:

- data validation
- query optimization

DBMS languages

Data definition (DDL):

- define database schemas.

Data manipulation (DML):

- create and update instances
- various kinds of updates:
  - incremental: *"insert (1001,American) into* **FLIGHT***"*.
  - bulk: *"copy all NW flights to Delta flights"*.
- transactions

Query languages:

- retrieve information from database instances

Query languages

FLIGHT

NUMBER	AIRLINE

TERMINAL

AIRLINE	TNAME

Simple lookup queries:

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

Complex queries:

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

Join

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

FLIGHT

NUMBER	AIRLINE
72	Delta
82	Delta
1210	American

TERMINAL

AIRLINE	TNAME
American	A
Delta	B
United	B

Aggregation queries

Computing aggregate values, e.g., the number of flights for each airline.

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

The result:

Delta	2
American	1

Efficient database access

Storage (disk):

- large capacity.
- persistent.
- random access.
- blocked.

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:

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

External:

- the views of users and application programs.
- uses a high-level data model, e.g., relational, XML.
- selective.
- examples: payroll, parking, registration.

Physical (internal):

- how the database is actually stored on disk.
- hidden from the users.
- uses lower-level concepts: files, indices,...
- created and maintained by the database administrator.

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.