Introduction to Database Systems – Part 1

General Concepts

What is a Database?

- Have you designed a database?
- Have you worked with a database?

Definition of Databases

- A **database** is a collection of related data.
- Implicit properties:
  - represents some **aspects of the real world**;
  - a **logically coherent collection** of data;
  - designed and built for a **specific purpose**.

Examples (Huge):

Amazon: – It has 244 million active customers, over 60 million items occupying many terabytes of data (clothing, sports, videos, office products).

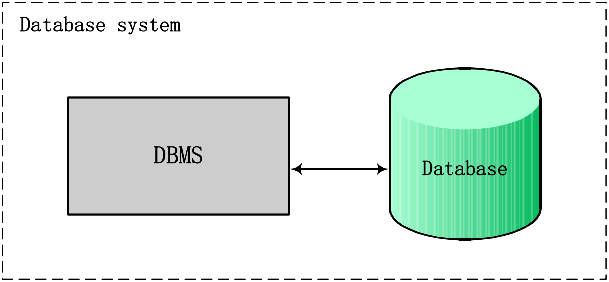
YouTube: – Over 1.3 billion users, 300 hours of videos added every minute, average of one billion mobile YouTube views per day

What is a Database Management System?

- A **database management system** (DBMS) is a collection of programs that enable users to create and maintain a database.
- It is a general-purpose software system that facilitates the process of
  - **defining:** specifying data types, structures and constraints;
  - **constructing:** storing data on some storage medium;
  - **manipulating:** retrieving and manipulating data;
  - **sharing:** using data by multiple users/programs simultaneously.
- Well-known relational DBMSs include Oracle, IBM DB2, Microsoft's Access, Microsoft's SQL Server, MySQL, postgreSQL, etc.

What is a Database System?

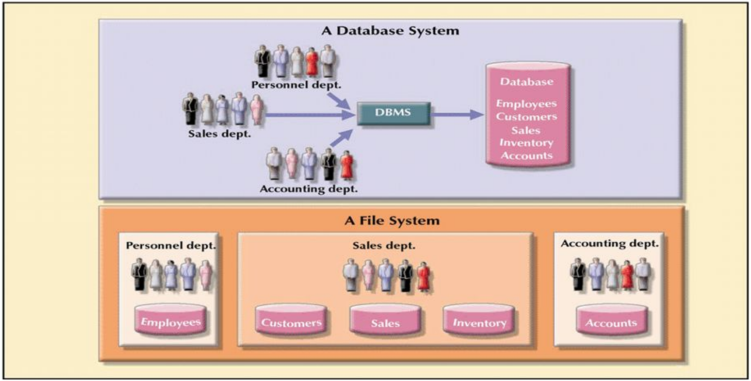
- A **database system** is part of information systems dealing with data retrieval and manipulation.
- It often refers to a DBMS plus a database.



- Main services a database system provides:
  - answer queries efficiently;
  - execute updates efficiently.

Why is a Database System Needed?

- Database system: an integrated collection of logically related data
- File system: many separate and unrelated files



Database Systems: Design, Implementation, & Management, 6th Edition, Rob & Coronel

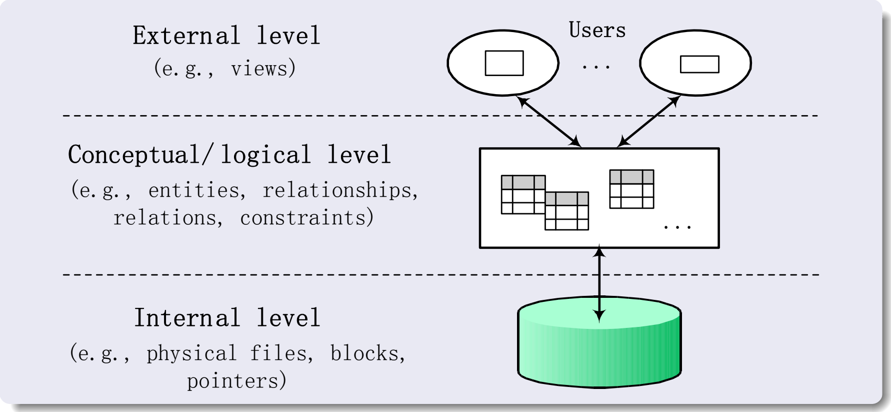
Why is a Database System Needed?

- Advantages of using a database system
  - **Data redundancy:** Data redundancy is controlled to ensure consistency and save the storage space.
  - **Data integrity:** Some integrity constraints can be enforced automatically by the DBMS.
  - **Data security:** Since the data is managed centrally, the DBMS ensures that the database access is through an authorized channel.

  In addition to the above, the database system also facilitates the following:

  Concurrent transactions; backup and recovery services; data independence; etc.

Three-level ANSI/SPARC Architecture



- **Note:** schemas at the three levels are *descriptions* of data; the stored data *actually* exists at the internal level (i.e., physical level) only.

Three-level ANSI/SPARC Architecture

- **External Schema**
  - perspective of the user / application
  - describes restructured parts of the database used in applications
- **Conceptual or Logical Schema**
  - perspective of a community of users
  - describes what data is stored in the database and relationships among data (independent from their physical storage structures).
- **Internal Schema**
  - perspective of the implementation / system realization
  - describes how data is stored in the database (e.g., physical storage structures).

Derived Principles – Data Independence

- **Logical data independence**: change the conceptual/logical schemas without having to change external schemas or application programs

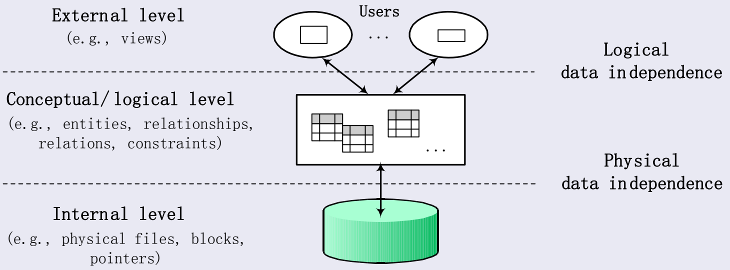
  Example: If adding or removing entities, external schemas that refer only to the remaining data should not be affected.

- **Physical data independence**: change the internal schemas without having to change the conceptual/logical schemas

  Example: If physical files were reorganised, we should not have to change the conceptual/logical schemas.

Derived Principles – Data Independence

- **Key idea**: When the schema is changed at some level,
  - the schema at the next higher level remains unchanged;
  - only the *mapping* between two levels is changed.



Historical Remarks I/II

- **Hierarchical Databases**
  - Oldest data model (1960s);
  - SABRE, a collaboration between IBM and American Airlines;
- **Network Databases**
  - Extension of hierarchical databases, from tree to network (late 1960s);
- **Relational Databases**
  - Edgar F. Codd,
    A Relational Model of Data for Large Shared Data Banks
  - System R and SQL

Historical Remarks II/II

- **Object-Oriented Databases**
  - Driven by object-oriented programming languages (1980s);
  - Designed to store and share complex, structured objects.
- **XML Databases**
  - XML is emerged as the standard for Web data exchange (1990s);
  - Suitable to sparse data, deeply nested data and mixed content.
- **NoSQL Databases**
  - Recent development in industry (since 2009);
  - We will discuss NoSQL databases at the end of this course.

Introduction to Database Systems – Part 2

Math Concepts

What are the Math Concepts behind Databases?

- Set
- Tuple
- Cartesian Product of Sets
- Relation

Set Notation

Container

Set Notation

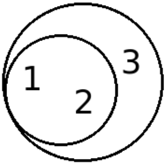
- We need set notation to represent formal definitions in this course.
- A **set** is a collection of distinct elements.
- Two basic properties of sets
  - The elements in a set have no order.
    e.g., $\{1, 2, 3\} = \{2, 3, 1\}$
  - Each element can not be in the set more than once.
    e.g., {Monday, Monday, Tuesday, Wednesday, Thursday, Friday, Friday} is Not a set. Note that **Multisets** allow to have duplicate elements.

Set Notation

- **Two ways of specifying a set**
  1. $\{x_1, \ldots, x_n\}$ (i.e., list all the elements in a set)
     - $\{ 2, 3, 4, 5 \}$
     - {Sydney, Melbourne, Canberra}
     - $\{\}$ or $\emptyset$, i.e., the *empty* set.
  2. $\{x|\varphi\}$ (i.e., describe the elements that satisfy a property φ)
     - {x | x is a student currently enrolled in COMP7240}
     - {x | x is an integer and x $>$ 0}

Set Operations

- **Membership**: $x \in A$ if x is in the set A; $x \notin A$ if x is not in the set A.

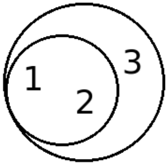


$1 \in \{1,2,3\}$ $\quad$ $3 \in \{1,2,3\}$

$2 \in \{1,2\}$ $\quad$ $3 \notin \{1,2\}$

Set Operations

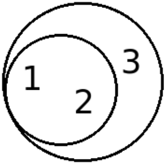
- **Equality**: If A and B have the same elements, we write $A = B$; otherwise we write $A \neq B$.
    - $\{x \mid x$ is an integer, $x > 1$ and $x < 6\ \} = \{2, 3, 4, 5\}$
    - If one set contains some element that is not in the other set, then they are different.



{1,2} $\neq$ {1,2,3}

Set Operations

- **Subset**: A is called a **subset** of B if every element of A is in B and we write $A \subseteq B$;
- **Proper subset**: A is called a **proper subset** of B if $A \subseteq B$ and A and B are not equal, and we write $A \subset B$.



$\{1,2\} \subseteq \{1,2,3\}$ $\quad$ $\{1,2\} \subseteq \{1,2\}$

$\{1,2\} \subset \{1,2,3\}$

Set Operations

- **Subset**: A is called a **subset** of B if every element of A is in B and we write $A \subseteq B$;
- **Proper subset**: A is called a **proper subset** of B if $A \subseteq B$ and A and B are not equal, and we write $A \subset B$.

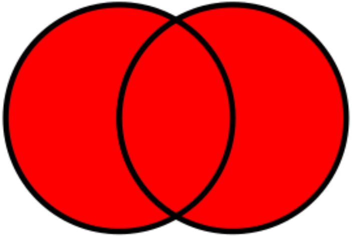
$$\subseteq \text{ means } \subset \text{ or } =$$

$$\{1,2\} \subseteq \{1,2,3\} \qquad \{1,2\} \subseteq \{1,2\}$$

$$\{1,2\} \subset \{1,2,3\}$$

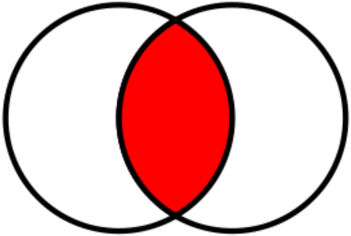
Set Operations

- **Union**: $A \cup B$ for the set containing everything in A and everything in B.
  - $\{3, 4, 5\} \cup \{3, 5, 7, 9\} = \{3, 4, 5, 7, 9\}$.



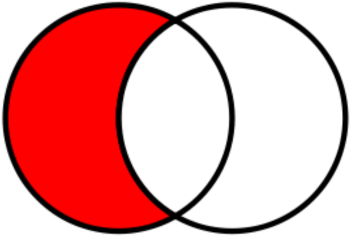
Set Operations

- **Intersection**: $A \cap B$ for the set of elements that are in both A and B
  - $\{3, 4, 5\} \cap \{3, 5, 7, 9\} = \{3, 5\}$.



Set Operations

- **Difference**: $A - B$ is the elements from A that are *not* in B
  - $\{3, 4, 5\} - \{3, 5, 7, 9\} = \{4\}$.



Set Operations – Exercise

- Let $A = \{1, 2, 3\}$ and $B = \{\mathit{true}, \mathit{false}\}$.
- Which of the following are correct?

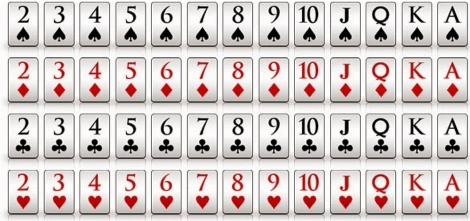
1. $\{2\} \in A$ — No! $\{2\} \subset A$ and $2 \in A$
2. $\mathit{true} \subset B$ — No! $\mathit{true} \in B$ and $\{\mathit{true}\} \subset B$
3. $\{2, 3\} \subseteq A \cup B$ — Yes! $A \cup B = \{1, 2, 3, \mathit{true}, \mathit{false}\}$
4. $2 \in A \cap B$ — No! $A \cap B = \{\}$
5. $2 \in A - \{1, 3, 5\}$ — Yes! $A - \{1, 3, 5\} = \{2\}$
6. $\{1, 4\} \subseteq A - B$ — No! $A - B = \{1, 2, 3\}$
7. $\emptyset \cap B = \emptyset$ — Yes! $\emptyset = \{\}$, the empty set

Tuple Notation

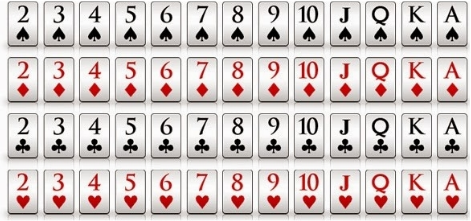
Tuple Notation

- A **tuple** is an ordered list of *n* elements.
  - $(1, 2, 3, 4, 5)$
  - $(Melbourne, Sydney, Canberra)$
- Two tuples are **equal** if they have the same elements in the same order.
  - $(1, 2, 3) \neq (2, 3, 1)$ (i.e., the order does matter!)
- The same element *can* be in a tuple twice.
  - (Monday, Monday, Tuesday, Wednesday, Thursday, Friday, Friday) is a tuple.
- Ordered pairs are special cases of tuples.

Cartesian Product of Sets

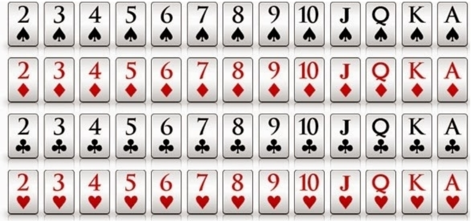


Cartesian Product of Sets



$$\{2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10,\ J,\ Q,\ K,\ A\}$$

Cartesian Product of Sets



$$\{2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10,\ J,\ Q,\ K,\ A\}$$

$$\{\spadesuit, \blacklozenge, \clubsuit, \heartsuit\}$$

Cartesian Product of Sets

- The Cartesian product operation takes an ordered list of sets, and returns a set of tuples.
- **Cartesian product** $D_1 \times ... \times D_n$ is the set of all possible combinations of values from the sets $D_1, ..., D_n$.
- It contains all the tuples with the first element from the first set, the second element from the second set, ...
- For example, $A \times B = \{(a, b) \mid a \in A \text{ and } b \in B\}$.
  If $A = \{2, 3\}$ and $B = \{Clubs, Diamonds, Hearts, Spades\}$
  Then $A \times B = \{(2, Clubs), (2, Diamonds), (2, Hearts), (2, Spades),$
  $(3, Clubs), (3, Diamonds), (3, Hearts), (3, Spades)\}$.
  $(2, Clubs) \in A \times B$, $(Spades, 3) \notin A \times B$, $(4, Hearts) \notin A \times B$
  $\{(3, Clubs), (3, Diamonds), (3, Hearts), (3, Spades)\} \subseteq A \times B$

Relation Notation

$$\{2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10,\ \mathrm{J},\ \mathrm{Q},\ \mathrm{K},\ \mathrm{A}\} \times \{\spadesuit, \blacklozenge, \clubsuit, \heartsuit\}$$

Relation Notation

$$\{2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10,\ J,\ Q,\ K,\ A\}$$

$$\{\spadesuit, \diamondsuit, \clubsuit, \heartsuit\}$$

Relation Notation

- A **relation** is a subset of a Cartesian product of sets.
- **Example**
  - Let $X = \{Canberra, Paris, Tokyo, Kyoto\}$, and $Y = \{Australia, France, Japan\}$
  - Let $R = \{(a, b) | a \in X, b \in Y \text{ and } a \text{ is a city in } b\}$.
  - It is easy to see that R is a relation
    - $R \subseteq X \times Y$.
    - $(Canberra, Australia) \in R$, $(Paris, France) \in R$ but $(Tokyo, France) \notin R$, $(France, Japan) \notin R$

Relation Notation

- A **relation** is a subset of a Cartesian product of sets.
- **Example**
  - Let $\mathbb{Z} = \{..., -1, 0, 1, 2, ...\}$, the set of all integers
  - Let $R = \{(x, y) \mid x \in \mathbb{Z}, y \in \mathbb{Z}$ and $x < y\}$.
  - It is easy to see that R is a relation.
    - $R \subseteq \mathbb{Z} \times \mathbb{Z}$.
    - $(0, 1) \in R$, $(-4, -2) \in R$
      but $(0, 0) \notin R$, $(100, -2) \notin R$.