Introduction to Database

Vũ Tuyet Trinh

1

Learning points

1. Course overview
2. Basic concepts on database
3. Data management
4. Relational data model

2

Learning objectives

- Upon completion of this lesson, students will be able to:
  - Recall the concepts of database, DBMS, data model, file system.
  - Identify the characteristics of database and file system approach in data management
  - Recall some basic concepts of relational data model.
  - Show some constraints of relational data model.

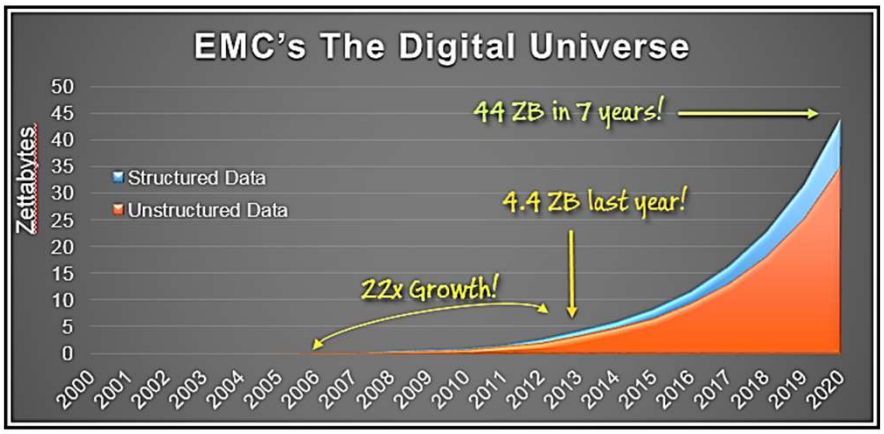
3

1. Introduction

- Major research field with long history (since the begining of computer)
- 90% applications use databases
- Hot jobs in startups, big coorporates
- Massive Industry: Oracle, IBM, Microsoft, Google, AWS

4

How big is our digital universe?



Source: http://hadoopsie.com/is-hadoop-better-than-sql-server/

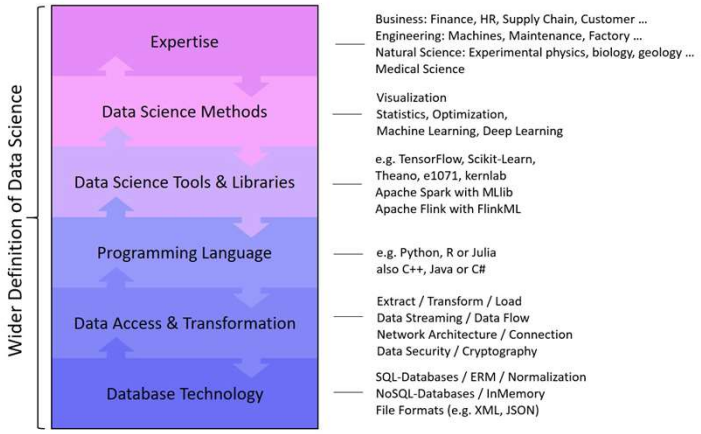
5

Data is the new oil of the 21st century

Source: https://www.economist.com

6

Data science knowledge stack



Source: https://data-science-blog.com

7

2. Basic concepts

2.1. Data
2.2. Database
2.3. Data model vs. schema vs. instance
2.4. Database management system (DBMS)
2.5. Database environment
2.6. Database users

8

2.1. Data

- Definitions

Wikipedia	Data is any sequence of one or more symbols given meaning by specific act(s) of interpretation.
Businessdictionary.com	Information in raw or unorganized form (e.g. alphabets, numbers, or symbols) that refer to, or represent, conditions, ideas, or objects. Data is limitless and present everywhere in the universe

- E.g. A specific student data: ID, Name, Age, Gender, Address,…

2.2. Database

- Definitions

Wikipedia	Database is **a shared collection of related data** designed **to meet the information needs** of an organization
Intro to CS	A database is a collection of information that is organized so that it can be **easily accessed, managed and updated**

2.2. Database

- Logically coherent
- Internally consistent
- Specific purpose
- Representation of the real world
  - Entities (e.g., Students, Courses)
  - Relationships (e.g., Tam is enrolled in C++)

Example

[Example] A course management system

Entities	Relationships
• Students • Courses • Teachers	• Students take in some courses • Course are given by some teachers

2.3. Model vs. Schema vs. Instance

Data Model	• Set of concepts used to describe the structure of a database: data types, relationships, constraints, semantics,
Schema	• Data structure fulfilled all features of the parts of the real world which is of interest to the users
Instance	• Data itself

13

Example

Data Model	**type** <type_name> = **record** <field_name> : <data_type>; <field_name> : <data_type>; … **end**;
Schema	**type** student = **record** ID : string; fullName: string; Birthday: date; Address: string ; Class: string; **end**;
Instance	(« Stud001 », « Nguyen », 1/4/1983, «1 Dai Co Viet », « 1F VN K50 »)

14

2.4. Database Management System (DBMS)

Wikipedia	A software to facilitate the creation and maintenance of a database
Techtarget	The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data

15

2.4. Database Management System (DBMS)

"More than 80 % of real world computer applications are associated with databases"

* Korth & Silberschatz. Database System Concepts.

16

Functions of DBMS

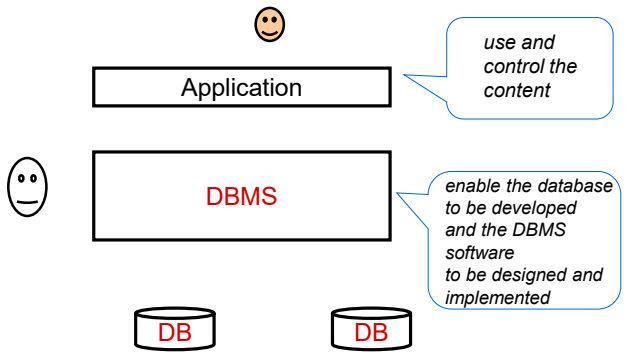
- Defining ~ specifying types of data
- Constructing ~ storing & populating
- Manipulating ~ querying, updating, reporting

MySQL · Microsoft Access · Microsoft SQL Server · ORACLE DATABASE · IBM DB2 · PostgreSQL

17

2.5. Database Environment

- A database environment is a collective system of components that regulates the management, the use of data, and the data itself
  - Hardware
  - Software
  - Data
  - Users
  - Procedures/Manuals



18

2.6. Database Users

- Database administrators
  - authorize access to the database
  - co-ordinate and monitoring its use
  - acquire software, and hardware resources, controlling its use and monitoring efficiency of operations.

19

2.6. Database Users

- Database Designers
  - define the content, the structure, the constraints, and functions or transactions against the database.
  - communicate with the end-users and understand their needs.
- End-users
  - use the data for queries, reports and some of them actually update the database content.
  - Casual end users
  - Naive users
  - Sophisticated end users

20

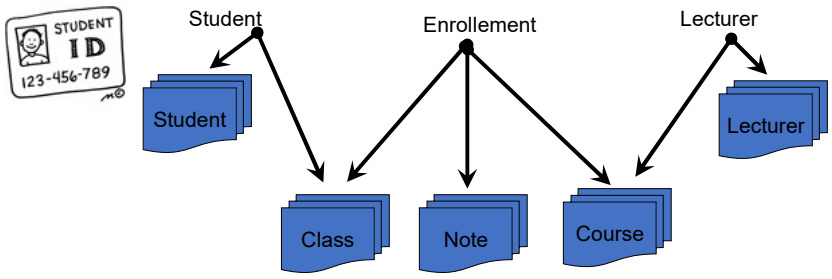
3. Data management

3.1. File management system approach

3.2. Database management system approach

21

3.1. File management system approach

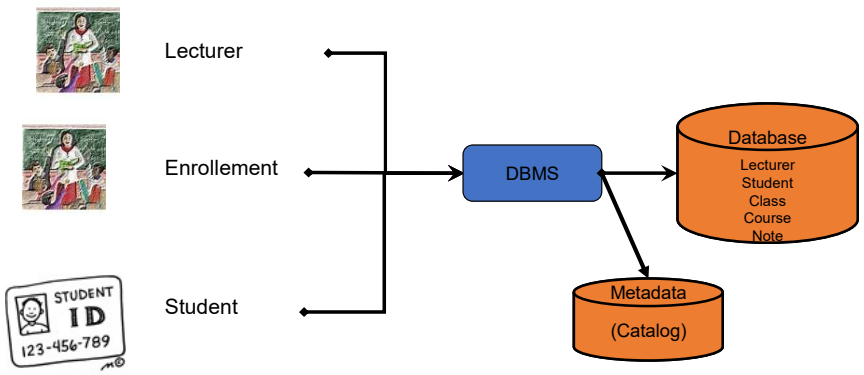


22

3.1. File management system approach

- Limitations
  - Uncontrolled redundancy
  - Inconsistent data
  - Inflexibility
  - Limited data sharing
  - Poor enforcement of standards
  - Low programmer productivity
  - Excessive program maintenance
  - Excessive data maintenance

3.2. Database approach



3.2. Database approach

- Controlled redundancy: consistency of data & integrity constraints
- Integration of data: self-contained & represents semantics of application
- Data and operation sharing: multiple interfaces
- Flexibility: data independence, data accessibility, reduced program maintenance
- Services & Controls
  - Security & privacy controls
  - backup & recovery
  - enforcement of standards
- Ease of application development

3.2. Database approach

- Characteristics of Database Approach
  - Self-describing
    - Catalog (or meta-data) stores the description of the database
    - Allow the DBMS software to work with different DBs
  - Data Abstraction:
    - Data model used to hide storage details
    - Present the users with a conceptual view of the DB
  - Sharing of data
    - Support multiple view of a DB
    - Allow concurrent access on a DB

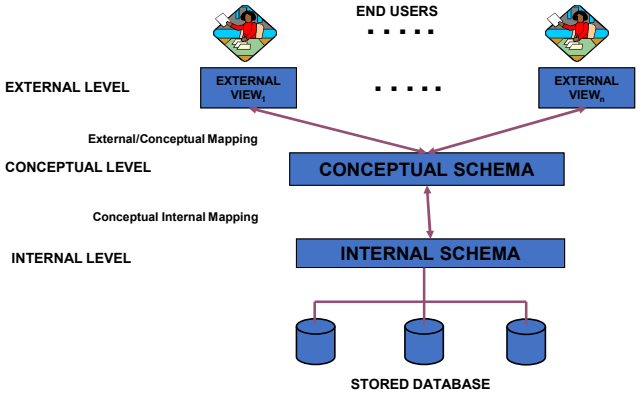
3.2. Database approach

- Characteristics of Database Approach
  - Persistence
    - store data on secondary storage
  - Retrieval
    - a declarative query language
    - a procedural database programming language
  - Performance
    - retrieve and store data quickly
    - deal with large volume of data

27

3.2. Database approach

- 3-tier Schema Model (ANSI-SPARC Architecture)



28

“checkpoint”

- Basic concepts
  - Data
  - Database
  - Data model vs. schema vs. Instance
  - Database management system (DBMS)
  - Database environment
  - Database users
- Data management
  - File management system approach
  - Database management system approach

29

4. Relational data model

4.1. Some data models

4.2. Database Basic concepts

4.3. Constraints

4.4. An example

30

4.1 Some data models

- Hierarchical database model
- Network model
- Object-oriented database model
- Relational model
- Entity-relationship model
- Document model
- …

31

Relational data model

- Is very simple model, was first introduced by Ted Codd of IBM Research in 1970
- Used by most of commercial database systems
- Query with high-level languages
- Efficient implementations
- Based on mathematical theory, closed to file structure and data structure, there are three sets of terminology:

Relation	Table	File
Tuple	Row	Record
Attribute	Column	Field

32

4.2. Basic concepts

Relations	• are saved in the format of tables, which have rows and columns.
Relation instance/state	• actual contents at given point in time. The lowercase letters q, r, s denote relation states.
Database	• a set of named relations (or tables).

Tuple	• A single row of a table, which contains a single record for that relation. • The letters t, u, v denote tuples.
Cardinality	• Is the number of tuples in a relation.
Degree (arity)	• Is the number of attributes in a relation.

33

Relational schema

~ structural description of relations in database

- A relation schema R of degree n, denoted by **$R(A_1, A_2, ..., A_n)$**, is made up of a relation name R and a list of attributes A_1, A_2, ..., A_n
- Each attribute A_i has values belong to domain D_i of A_i, denoted by **$dom(A_i)$**
- An n-tuple t in a relation r(R) is denoted by **$t = <v_1, v_2, ..., v_n>$**, where v_i is the value corresponding to attribute A_i. Both $t[A_i]$ and $t.A_i$ (and sometimes t[i]) refer to the value v_i in t for attribute A_i

34

4.2. Basic concepts

- Notice that the uppercase letters Q, R, S denote relation names.
- A relation (or relation state) r of the relation schema **$R(A_1, A_2, ..., A_n)$**, also denoted by r(R), is a set of n-tuples **$r = \{t_1, t_2, ..., t_m\}$**. Each n-tuple t is an ordered list of n-values **$t = <v_1, v_2, ..., v_n>$**, where each value v_i, $1 \le i \le n$, is an element of $dom(A_i)$ or is a special NULL value.
- A relation (or relation state) r(R) is a mathematical relation of degree n on the domains $dom(A_1)$, $dom(A_2)$, ..., $dom(A_n)$, which is a subset of the Cartesian product of the domains that define R:
  - $r(R) \subseteq (dom(A_1) \times dom(A_2) \times ... \times dom(A_n))$

4.2. Basic concepts

- An example
  - student(student_id, first_name, last_name, dob, gender, address, note, clazz_id)

Relation name

Attributes

student

student_id	first_name	last_name	dob	gender	address	note	clazz_id
1234	David	Beckham	12/21/1997	Male	London, UK		1
1238	Theresa	May	08/06/1998	Female	London, UK		1
1452	David	Cameron	07/06/1997	Male	Bangor, UK		1
1497	Tony	Blair	03/01/1999	Male	Bath, UK		2
1516	John	Major	03/01/1998	Male	Bradford		2
1542	Margaret	Thatcher	05/08/1997	Female	Cambridge		2

Tuples

Cardinality = 6

Degree = 8

4.3. Constraints

4.3.1. Introduction

4.3.2. Types of constraints

4.3.3. An example

37

4.3.1. Introduction

- Every relation has some conditions that must hold for it to be a valid relation.
- These conditions are called Relational Integrity Constraints.
- Provide a way of ensuring that changes made to the database by authorized users do not result in a loss of data consistency.

38

4.3.2. Types of constraints

- Key constraints
- Domain constraints
- Referential integrity constraints

Key constraints

- A key is an attribute or a set of attributes in the relation, which can identify a tuple uniquely.
- Key constraints force that:
  - in a relation with a key, no two tuples can have identical values for key attributes.
  - a key can not have NULL values.
  - Key constraints are also referred to as Entity Constraints.

Some types of key

- Superkey: An attribute, or a set of attributes, that uniquely identifies a tuple within a relation.
- Candidate Key:
  - Superkey (K) such that no proper subset is a superkey within the relation
  - In each tuple of the relation, values of K uniquely identify that tuple (uniqueness)
  - No proper subset of K has the uniqueness property (irreducibility)
- Primary Key: Candidate key selected to identify tuples uniquely within a relation. Each key attribute of primary key has its name underlined.

Some types of key (cont.)

- Alternate Keys: Candidate keys that are not selected to be the primary key.
- Minimal key: a minimal set of attributes that can be used to identify a single tuple.
- Foreign Key:
  - Attribute, or set of attributes, within one relation that matches candidate key of some relation
  - Used to model relationships between relations
  - Each key attribute of foreign key has its name italic

Domain constraints

- Attributes have specific values in real-world scenario. Every attribute is bound to have a specific range of values.
- Within each tuple, the value of each attribute A must be an atomic value from the domain dom(A).
- The data types associated with domains
  - standard numeric data types for integers (short integer, integer, and long integer) and real numbers (float, double precision float).
  - Characters, Booleans, fixed-length strings, and variable-length strings, date, time, timestamp, and money, or other special data types.
  - a subrange of values from a data type .
  - an enumerated data type in which all possible values are explicitly listed.

Domain constraints (cont.)

- Domain constraints
  - Null value
    - Represents value for an attribute that is currently unknown or not applicable for any tuple;
    - deals with incomplete or exceptional data;
    - represents the absence of a value and is not the same as zero or spaces

Referential integrity constraints

- Referential integrity constraints work on the concept of Foreign Keys. A foreign key is a key attribute of a relation that can be referred in other relation.
- Referential integrity constraint states that if a relation refers to a key attribute of a different or same relation, then that key element must exist.

45

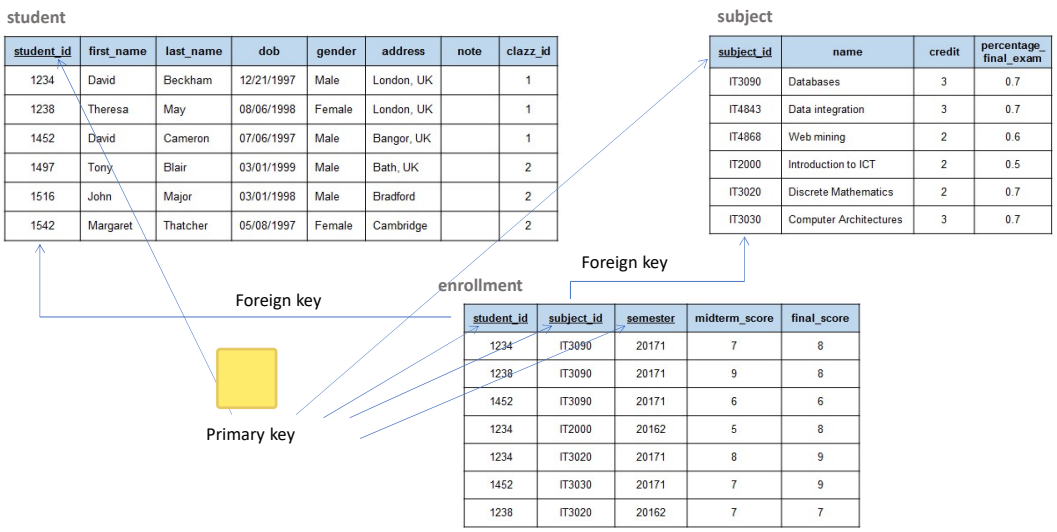
4.4. An example

student(**<u>student_id</u>**, first_name, last_name, dob, gender, address, note, *clazz_id*)

subject(**<u>subject_id</u>**, name, credit, percentage_final_exam)

enrollment(***<u>student_id, subject_id, semester,</u>*** midterm_score, final_score)

46



47

Summary

- Basic concepts
  - Database vs. DBMS. Database system/Database environment
  - Data Model vs. Schema vs. Instance
- Relational data model
  - Relations, relation instance/state, relation schema
  - Database, tuple
  - Cardinality, degree
- Constraints
  - Key constraints
  - Domain constraints
  - Referential integrity constraints

48

Keywords and descriptions

Keyword	Description
Database	A shared collection of related data designed to meet the information needs of an organization
DBMS	A software to facilitate the creation and maintenance of a database
Data model	A set of concepts used to describe the structure of a database: data types, relationships, constraints, semantics, …
Schema	A data structure fulfilled all features of the parts of the real world which is of interest to the users
Instance	The data itself (in the context of data model and database schema)

49

Keywords and descriptions

Keyword	Description
Data model	A set of concepts used to describe the structure of a database: data types, relationships, constraints, semantics…
Relation	Is thought of as a table of values, each row in the table represents a collection of related data values.
Key	An attribute or a set of attributes in the relation, which can identify a tuple uniqu ely.
Integrity constraints	Provide a way of ensuring that changes made to the database by authorized user s do not result in a loss of data consistency.

50