


**CSIE30600/CSIEB0290**  
**Database Systems**

**Lecture 1:**  
**Introduction**



**Outline**

- Data is ubiquitous
- Basic Definitions
- Types of Databases and Database Applications
- Typical DBMS Functionalities
- Example of a Database (UNIVERSITY)
- Main Characteristics of the Database Approach
- Database Users
- Advantages of Using the Database Approach
- History of Database Systems
- Extending Database Capabilities
- When Not to Use Databases

CSIE30600/CSIEB0290 Database Systems Introduction 2

## Data & Technological Advances

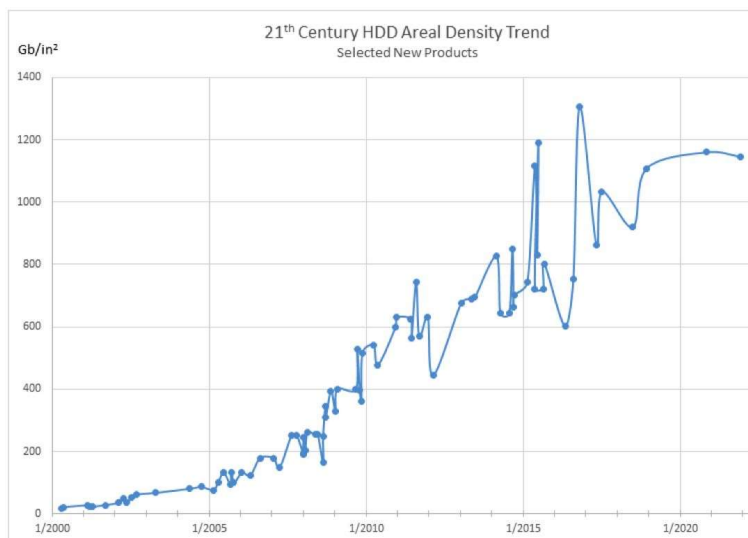


- Five classes of technological advances are changing our relationship with data:
- More **storage space**
  - allows us to keep more data
- Faster **processor (and memory) speeds**
  - allows us to access and process more data
- Better **networking**
  - allows us to share data more efficiently
- Different **“sensors”**
  - allows us to access new kinds of data
- Better **processing methods** (AI & machine learning)
  - allows us to process data more intelligently

CSIE30600/CSIEB0290 Database Systems

Introduction 3


## HDD Areal Density Growth



CSIE30600/CSIEB0290 Database Systems

Introduction 4

## HDD Cost (Historical)




**Hard drive cost per GB over time**

date	capacity	cost	\$/GB
1957	3.75 MB	\$34,500	\$9.2 million/GB
1989	40 MB	\$1,200	\$30,000/GB
1995	1 GB	\$850	\$850/GB
2004	250 GB	\$250	\$1/GB
2011	2 TB	\$70	\$0.035/GB
2018	4 TB	\$75	\$0.019/GB

CSIE30600/CSIEB0290 Database Systems Introduction 5

## HDD Price per TB

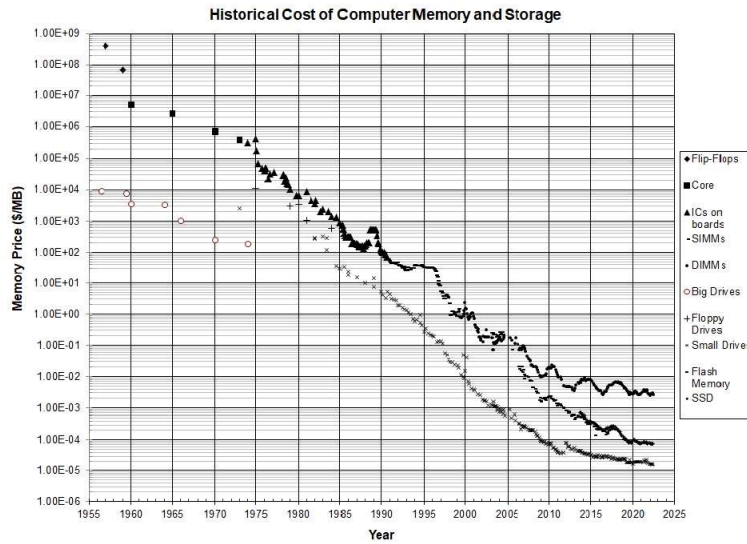


Price per TB	Price	Capacity	Warranty	Form Factor	Technology	Condition	Name
\$7.500	\$30	4 TB		Internal	SAS	Used	<a href="#">SEAGATE ST4000NM0023 Constellation ES.3 4TB 7200 RPM 128MB Cache SAS 6.0Gb/s 3.5-Inch Internal Hard Drive (Bare Drive)</a>
\$10.00	\$30	3 TB	2 years	Internal 3.5"	HDD	Used	<a href="#">(Old Model) Seagate 3TB Desktop HDD SATA 6Gb/s 64MB Cache 3.5-Inch Internal Bare Drive (ST3000DM001)</a>
\$10.00	\$60	6 TB	5 years	Internal 3.5"	HDD	Used	<a href="#">HDD - HGST Ultrastar (HUS726060ALE610) 6TB 7200RPM 128MB Cache SATA 6Gb/s 3.5-Inch Enterprise Hard Drive (for NAS, Desktop PC, Surveillance Storage) - 5 Year Warranty (Renewed)</a>
\$11.25	\$68	6 TB	3 months	Internal 3.5"	HDD	Used	<a href="#">HGST Ultrastar 7K6000 HUS726060ALE610 (0F23001) 6TB 7200 RPM SATA 6Gb/s 128MB Cache 3.5-Inch Enterprise Hard Drive (Renewed)</a>
\$11.25	\$180	16 TB	5 years	Internal 3.5"	HDD	Used	<a href="#">Toshiba MG08ACA16TE 16TB 7200RPM 512e 3.5" SATA Enterprise Desktop Hard Drive</a>
\$11.25	\$45	4 TB	3 years	Internal 3.5"	HDD	Used	<a href="#">Western Digital WD4000FYYZ ENTERPRISE 4TB 7200RPM 64MB Cache SATA 6.0Gb/s 3.5-Inch Internal Hard Drive</a>
\$11.25	\$68	6 TB	3 years	Internal 3.5"	HDD	Used	<a href="#">Seagate Constellation ES.3 (ST6000NM0044) 6TB 128MB Cache 7200RPM SATA 6.0Gb/s 3.5-Inch Internal Hard Drive - 3 Year Warranty (Renewed)</a>
\$11.25	\$68	6 TB	3 years	Internal 3.5"	HDD	New	<a href="#">MDD (MD6000GSA12872NAS) 6TB 7200RPM 128MB Cache SATA 6.0Gb/s 3.5-Inch Internal Hard Drive for NAS/Network Storage - 3 Years Warranty</a>
\$11.67	\$140	12 TB	2 years	Internal 3.5"	HDD	Used	<a href="#">WP Arsenal 12TB SATA 7200RPM 3.5-Inch DAS Hard Drive (Renewed)</a>
\$12.00	\$72	6 TB	3 years	Internal 3.5"	HDD	New	<a href="#">MaxDigitalData 6TB 7200RPM 128MB Cache SATA 6.0Gb/s 3.5" Internal Hard Drive for Surveillance (MD6000GSA12872DVR) - 3 Years Warranty</a>
\$12.12	\$48	4 TB	2 years	Internal 3.5"	HDD	New	<a href="#">MaxDigitalData 4TB 64MB Cache 7200PM SATA 6.0Gb/s 3.5" Internal Surveillance CCTV DVR Hard Drive (MD4000GSA6472DVR) - w/ 2 Year Warranty</a>
\$12.12	\$48	4 TB	2 years	Internal 3.5"	HDD	New	<a href="#">MaxDigital 4TB 7200RPM 64MB Cache SATA III 6.0Gb/s (Enterprise Storage) 3.5" Internal Hard Drive w/2 Year Warranty</a>
\$12.14	\$170	14 TB	2 years	Internal 3.5"	HDD	Used	<a href="#">HGST WD Ultrastar DC HC530 14TB SATA 6Gb/s 3.5-Inch Data Center HDD - WUJH721414AL E604 0F31152 (Renewed)</a>
\$12.25	\$49	4 TB	2 years	Internal 3.5"	HDD	New	<a href="#">MaxDigitalData 4TB 64MB Cache 5900PM SATA 6.0Gb/s 3.5" Internal Surveillance CCTV DVR Hard Drive (MD4000GSA6459DVR) - w/ 2 Year Warranty</a>

(Last updated at 2022-07-25 06:35:40 UTC)

CSIE30600/CSIEB0290 Database Systems Introduction 6

# Cost of Memory & Storage



CSIE30600/CSIEB0290 Database Systems

Introduction 7

# SSD vs HDD

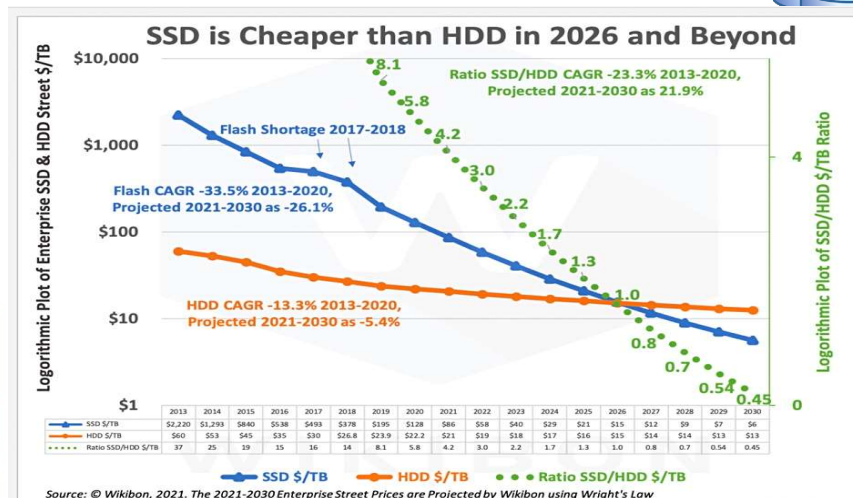
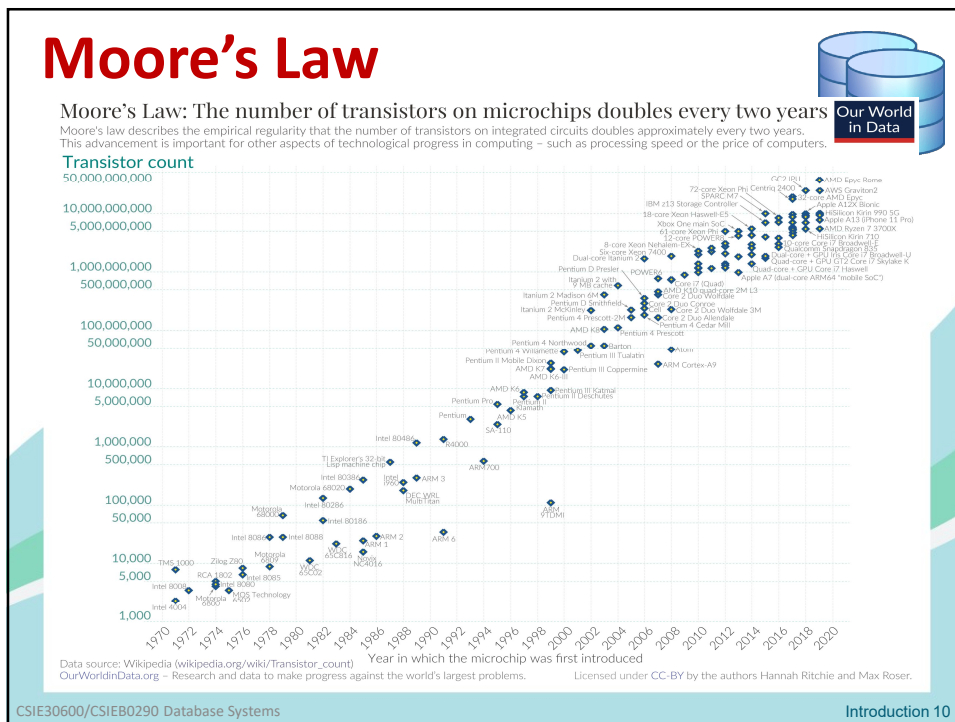
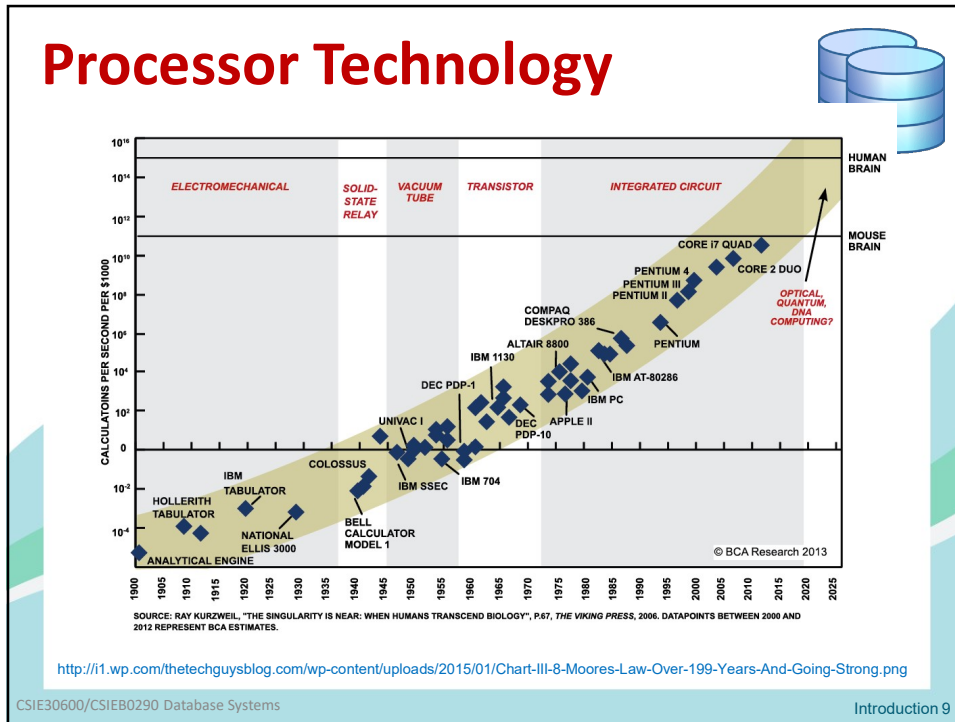
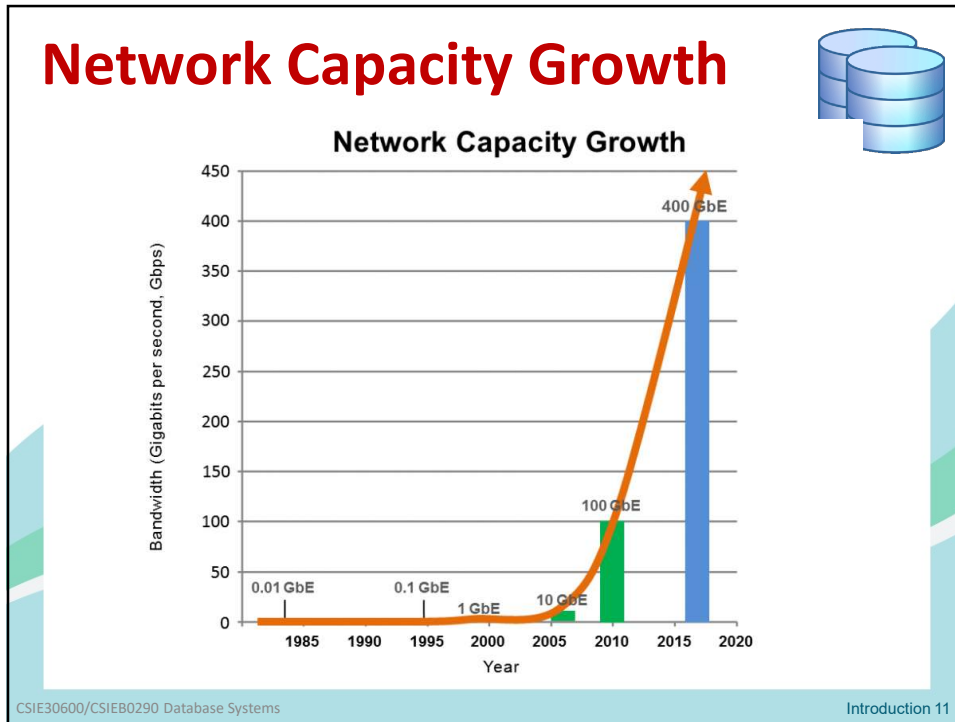


Figure 4 - SSD/HDD Pricing Ratio 2013 - 2030

Source: © Wikibon, 2021.





## New Data from Sensor Web

Traffic Monitor

Satellite-borne Imaging device

Environmental Monitor

Strain Gauge

Airborne Imaging Device

Health Monitor

Industrial Process Monitor

Webcam

Stored Sensor Data

- All sensors reporting position
- All connected to the web
- All with metadata registered
- All readable remotely
- Some controllable remotely

CSIE30600/CSIEB0290 Database Systems Introduction 12

## Data Everywhere



- Airline flight management system
- Financial data
- Commercial store (eg, WalMart) data
- Department of Motor Vehicles
- Surveillance video
- University student records
- Baseball results
- Web sites
- Medical records
- ...

## Need Effective Data Management



- Effective management can make an organization's data a valuable **asset**(資產).
- Ineffective policies can make an organization's data a **liability**(負債).
- **Big data analytics** is becoming the **gold mine** of the 21<sup>st</sup> century.
- The paradigm has been extended from **database systems** to **data science**.

## Basic Concepts



- **Data**: Known facts that can be recorded and have an implicit meaning.
- **Database**: Collection of interrelated data
- **Mini-World** or **Universe of Discourse (UoD)**: Some part of the real world about which data is stored in a database.
- **Database Management System (DBMS)**: A collection of programs to facilitate the creation and maintenance of a database.
- **Database System** = DBMS + Database
- A database system contains **information** about a particular **enterprise**.
- A database system provides an **environment** that is both **convenient** and **efficient** to use.

## Database Management System (DBMS)



- DBMS is:
  - A collection of software programs
  - General purpose
- DBMS enables users to:
  - **Define** DB
  - **Construct** DB
  - **Change** (or **update**) DB
  - **Query** the data in DB
  - **Share** DB
- DBMS maintains the **integrity** of DB





## DB Engine Ranking

395 systems in ranking, August 2022

Rank			DBMS	Database Model	Score		
Aug 2022	Jul 2022	Aug 2021			Aug 2022	Jul 2022	Aug 2021
1.	1.	1.	Oracle +	Relational, Multi-model	1260.80	-19.50	-8.46
2.	2.	2.	MySQL +	Relational, Multi-model	1202.85	+7.98	-35.37
3.	3.	3.	Microsoft SQL Server +	Relational, Multi-model	944.96	+2.83	-28.39
4.	4.	4.	PostgreSQL +	Relational, Multi-model	618.00	+2.13	+40.95
5.	5.	5.	MongoDB +	Document, Multi-model	477.66	+4.68	-18.88
6.	6.	6.	Redis +	Key-value, Multi-model	176.39	+2.77	+6.51
7.	7.	7.	IBM Db2	Relational, Multi-model	157.23	-3.99	-8.24
8.	8.	8.	Elasticsearch	Search engine, Multi-model	155.08	+0.75	-2.01
9.	9.	10.	Microsoft Access	Relational	146.50	+1.41	+31.66
10.	10.	9.	SQLite +	Relational	138.87	+2.20	+9.06
11.	11.	11.	Cassandra +	Wide column	118.15	+3.74	+4.49
12.	12.	12.	MariaDB +	Relational, Multi-model	113.89	+1.37	+14.92
13.	13.	23.	Snowflake +	Relational	103.12	+3.97	+56.58
14.	14.	13.	Splunk	Search engine	97.44	-0.76	+6.84
15.	16.	16.	Amazon DynamoDB +	Multi-model	87.26	+3.32	+12.36
16.	15.	15.	Microsoft Azure SQL Database	Relational, Multi-model	86.18	+1.28	+11.02
17.	17.	14.	Hive +	Relational	78.66	-0.82	-5.27
18.	18.	17.	Teradata +	Relational, Multi-model	69.07	-1.85	+0.25
19.	19.	18.	Neo4j +	Graph	59.35	+0.94	+2.40
20.	20.	20.	Solr	Search engine, Multi-model	55.78	+0.08	+4.71

<https://db-engines.com/en/ranking>

CSIE30600/CSIEB0290 Database Systems Introduction 17

## DB Engines Ranking Trend

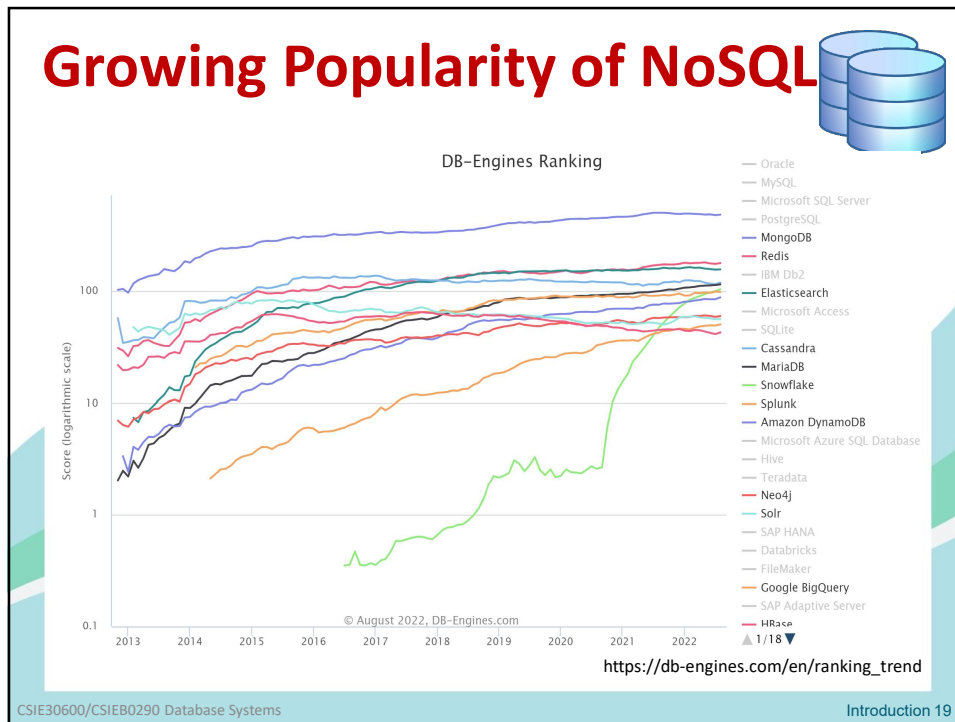
August 2022

DB-Engines Ranking

© August 2022, DB-Engines.com

[https://db-engines.com/en/ranking\\_trend](https://db-engines.com/en/ranking_trend)

CSIE30600/CSIEB0290 Database Systems Introduction 18



## Main Goals of DB Course

- To learn **why**, **when**, and **where** databases are useful.
- To understand **how** to **use** a **DBMS**
  - How to design and create DB, data models, SQL, ...
- To study how a DBMS **works**
  - Properties of disks and files, software to manage reading/writing, algorithms to answer user queries, transaction management, ...
- To explore new concepts in data science/analytics
  - Big data, NoSQL/NewSQL/Distributed SQL
  - Data science, data analytics
  - Streaming data management

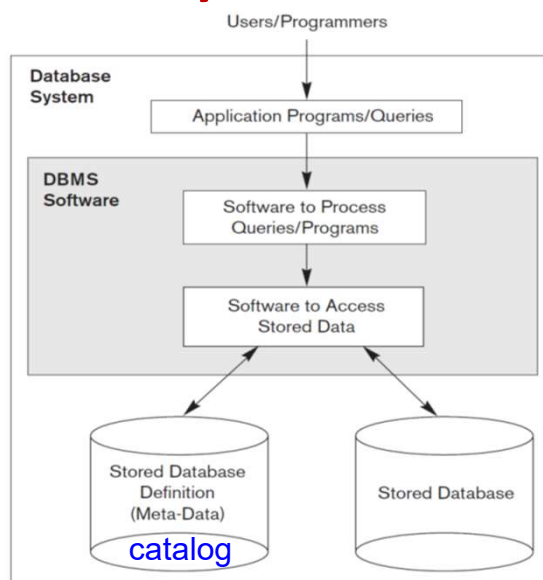
CSIE30600/CSIEB0290 Database Systems Introduction 20

## Types of Databases and Applications



- Traditional Applications:
  - Numeric and Textual Databases
- More Recent Applications:
  - Multimedia Databases (images, audio, video, ...)
  - Geographic Information Systems (GIS)
  - Data Warehouses
  - Real-time and Active Databases
  - Many other applications
- New Trends: cloud DB, big data analytics, IoT, AI/ML
- First part: focuses on **traditional applications**
- *A number of recent applications are described later in the class and book.*

## Database System Environment



**Figure 1.1**  
A simplified database system environment.

## Typical DBMS Functionality



- **Define** a particular database in terms of its data types, structures, and constraints
- **Construct** or **Load** the initial database contents on a secondary storage medium
- **Manipulating** the database:
  - Retrieval: Querying, generating reports
  - Modification: Insertions, deletions and updates
  - Generally known as CRUD
  - Accessing the database with Web/cloud applications
- **Processing** and **Sharing** by a set of concurrent users and application programs – yet, keeping all data valid and consistent

## Typical DBMS Functionality



- Other features:
  - **Protection** or **Security** measures to prevent unauthorized access
  - “Actively” take **internal actions** on data
  - **Presentation** and **Visualization** of data
  - **Maintaining** the database and associated programs over the lifetime of the database application
    - Called database, software, and system maintenance

## Application Activities on DB



- Applications interact with a database by generating
  - **Queries**: that access different parts of data and formulate the result of a request
  - **Transactions**: that may read some data and “update” certain values or generate new data and store that in the database
- Applications must not allow unauthorized users to access data
- Applications must keep up with changing user requirements against the database

## Database Applications



- **Banking**: all transactions
- **Airlines**: reservations, schedules
- **Universities**: registration, grades
- **Sales**: customers, products, purchases
- **Online retailers**: order tracking, customized recommendations
- **Manufacturing**: production, inventory, orders, supply chain
- **Human resources**: employee records, salaries, tax deductions
- **Internet & Web**: search, data management, ...
- **Social media**: content management, relationships, ...
- **Databases touch all aspects of our lives**

## Purpose of DB Systems



- In the early days, database applications were built directly on top of **file systems**
- Drawbacks of using file systems :
  - **Data redundancy** and **inconsistency**
    - Multiple file formats, duplication in different files
  - **Difficulty in accessing** data
    - Need to write a new program for each new task
  - **Data isolation** — multiple files and formats
  - **Integrity** problems
    - Integrity constraints (e.g. account balance > 0) become “buried” in program code rather than being stated explicitly
    - Hard to add new constraints or change existing ones

## Purpose of DB Systems



- Drawbacks of using file systems (cont.)
  - **Atomicity** of updates
    - Failures may leave database in an inconsistent state
    - Example: Transfer of funds from one account to another
  - **Concurrent access** by multiple users
    - Concurrent accessed needed for performance
    - Uncontrolled concurrent accesses can lead to inconsistencies
      - Example: Two people reading and updating at the same time
  - **Security problems**
    - Hard to provide user access to some, but not all, data
- Database systems offer solutions to **ALL** the problems above.

## Example of a Database



- **Mini-world** for the example:
  - Part of a UNIVERSITY environment.
- **Some mini-world *entities***:
  - STUDENTs
  - COURSEs
  - SECTIONs (of COURSEs)
  - DEPARTMENTs
  - INSTRUCTORs



## Example of a Database



- **Some mini-world *relationships***:
  - SECTIONs *are of specific* COURSEs
  - STUDENTs *take* SECTIONs
  - COURSEs *have prerequisite* COURSEs
  - INSTRUCTORs *teach* SECTIONs
  - COURSEs *are offered by* DEPARTMENTs
  - STUDENTs *major in* DEPARTMENTs
- Note: The above entities and relationships are typically expressed in a **conceptual data model**, such as the **ENTITY-RELATIONSHIP** data model (to be discussed later in class)

# Example of a Database



**STUDENT**

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

**Figure 1.2**

A database that stores student and course information.

**COURSE**

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

**GRADE\_REPORT**

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

**SECTION**

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

**PREREQUISITE**

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

# Main Characteristics



- **Self-describing:** A **DBMS catalog (meta-data)** stores the description of the database. (next slide)
- **Program-data Independence:** Allows changing storage structures w/o changing DBMS access programs.
- **Data abstraction:** **Data models** hide storage details and present the users with a **conceptual view** of the DB.
- **Multiple views:** Each user may see a different view of the database.
- **Data sharing:** among multiple users
- **Transactions, concurrent access , recovery , OLTP**



# A Simplified DB Catalog



**Figure 1.3**  
An example of a database catalog for the database in Figure 1.2.

### RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

### COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
....	....	....
....	....	....
....	....	....
Prerequisite_number	XXXXNNNN	PREREQUISITE

*Note:* Major\_type is defined as an enumerated type with all known majors. XXXXNNNN is used to define a type with four alphabetic characters followed by four numeric digits.

# Database Users



- Users may be divided into
  - Those who actually use and control the database content, and those who design, develop and maintain database applications (called “Actors on the Scene” 幕前使用者), and
  - Those who design and develop the DBMS software and related tools, and the computer systems operators (called “Workers Behind the Scene” 幕後工作者).

## Database Users



- Actors on the scene
  - **Database administrators:**
    - Responsible for authorizing access to the database, for coordinating and monitoring its use, acquiring software and hardware resources, controlling its use and monitoring efficiency of operations.
  - **Database Designers:**
    - Responsible to define the content, the structure, the constraints, and functions or transactions against the database. They must communicate with the end-users and understand their needs.

## Database Administrator (DBA)



- Coordinates all the activities of the database system; must have a good understanding of the enterprise's information resources and needs.
- Database administrator's **duties:**
  - Schema definition
  - Storage structure and access method definition
  - Schema and physical organization modification
  - Granting user authority to access the database
  - Specifying integrity constraints
  - Acting as liaison with users
  - Monitoring performance and responding to changes in requirements

## End-users



- Actors on the scene (continued)
  - **End-users**: use the data for queries, reports and some of them update the database content.
- End-users can be categorized into:
  - **Casual**: access db occasionally when needed
  - **Naïve** or **Parametric**: they make up a large section of the end-user population.
    - They use previously well-defined functions in the form of “**canned transactions**” against the database.
    - Examples are bank-tellers or reservation clerks who do this activity for an entire shift of operations.

## End-users (cont.)



- **Sophisticated**:
  - Business analysts, scientists, engineers, others thoroughly familiar with the system capabilities.
  - Many use tools in the form of **software packages** that work closely with the stored database.
- **Stand-alone**:
  - Mostly maintain personal databases using ready-to-use packaged applications.
  - An example is a tax program user that creates its own internal database.
  - Another example is a user that maintains an address book

## System Analysts and Application Programmers



- **System analysts:** Analyze problem, determine the requirements of the users, develop specifications.
- **Application programmers:** Design and implement specification, testing, debugging, maintaining softwares. Also known as **software developers** or **software engineers**.
- **Business analysts:** People who can analyze vast amounts of business data and real-time data (“Big Data”) for better decision making, planning, advertising, marketing etc.

## Users behind the Scene



- **DB designers** – design the database systems for end users
- **DBMS designers** – design database management systems and tools for building databases
- **Tool designers** – Design and implement tools that facilitate building of applications and allow using database effectively (eg. modeling and designing databases, performance monitoring, prototyping, test data generation, user interface creation, simulation etc.)
- **Operators** and **maintenance personnel**.

## Advantages of Using the Database Approach



- **Controlling redundancy** in data storage and in development and maintenance efforts.
  - Sharing of data among multiple users.
- **Restricting unauthorized access** to data.
- Providing **persistent storage** for program objects
  - In object-oriented DBMS
- Providing **storage structures** (e.g. **indexes**) efficient data access
- Provide **optimization of queries** for efficient processing

## Advantages of Using the Database Approach (cont.)



- Providing **backup** and **recovery** services.
- Providing **multiple interfaces** to different classes of users.
- Representing **complex relationships among data**.
- Enforcing **integrity constraints** on the database.
- Drawing **inferences and actions** from the stored data using deductive and active rules
- **Finding** actionable **insights** from large data sets
- ...

## Additional Implications



- Potential for **enforcing standards**:
  - This is very crucial for the success of database applications in large organizations. **Standards** refer to data item names, display formats, screens, report structures, meta-data (description of data), Web page layouts, etc.
- **Reduced application development time**:
  - Incremental time to add each new application is reduced.

## Additional Implications (cont.)



- **Flexibility** to change data structures:
  - Database structure may evolve as new requirements are defined.
- **Availability** of current information:
  - Extremely important for on-line transaction systems such as airline, hotel, car reservations.
- **Economies of scale**:
  - Wasteful overlap of resources and personnel can be avoided by consolidating data and applications across departments.

## History of DB Systems



- **Pre-1960s**
  - File processing systems
  - Redundancy and inconsistency between files
  - Incompatibility between access programs
  - Data isolation
  - Concurrent access anomalies
  - Security and integrity problems

## DB History (cont.)



- **The '60s**
  - **Carles Bachman** designed the 1st DBMS Integrated Data Store (and received the 1st Turing Award in 1973)
  - **Three-level architecture** (more about this in next lecture)
  - CODASYL, DBTG, and the **network model**
  - **Hierarchical model** and the IMS system

## DB History (cont.)



### • The '70s

- **Edgar Codd** (1970): The **Relational model** (Codd won the 1981 Turing Award)
- Provide a sound theoretical base.
- 1975, 1st ACM SIGMOD international conference
- 1975, 1st VLDB international conference
- **Peter Chen 陳品山** (1976): The **Entity-relationship model**
- **System R** (IBM), **INGRES** (UC-Berkely), **System 2000** (UT-Austin)
- **SQL, QUEL**

## DB History (cont.)



### • The '80s

- **Commercial** relational DBMS (DB2, ORACLE, SYBASE, INFORMIX, ...)
- DBMS on **PC's** (DBASE, PARADOX, ...)
- **Transaction management** (James Gray won the 1999 Turing Award)
- **Standards** (SQL standardized in the late 1980s)



## DB History (cont.)



- **The '90s**

- **New applications** (Web, CAD/CAM, CASE, office automation, science and engineering, VLSI, ...)
- Demand for **new DBMS technologies**
- Object-oriented DBs, Parallel/Distributed DBs, Active/Deductive DBs, Multimedia DBs, Mobile DBs, Temporal/Real-time DBs, Spatial DBs (such as GIS), ...
- The emergence of **ERP** (Enterprise Resource Planning) and **MRP** (Material Requirements Planning) packages
- **Data Warehousing** and **data mining**
- DBMS in the **Internet/Web** and **E-commerce** applications

## DB History (cont.)



- **The 2000s and beyond**

- **XML, XQuery** and the **Semantic Web**
- **Data Stream Management Systems (DSMS)**
  - Sensor databases, IoT data management
  - Network traffic analysis
  - RFID data management
  - ...
- **Mobile Data Management (MDM)**
- **Cloud Databases**
- **AIDB, MLDB** (DB + AI&ML)

## New Trends



- First decade of the 21st century has seen **tremendous growth** in **user generated data** and **automatically collected data** from applications and search engines.
- **Social Media** platforms such as Facebook, IG and Twitter are generating millions of transactions a day and businesses are interested to tap into this data to “understand” the users.
- **Cloud Storage** and **Backup** is making unlimited amount of storage available to users and applications

## Big Data, NoSQL/NewSQL, Distributed SQL



- New data storage, management and analysis technology was necessary to deal with the huge volume of data in petabytes a day ( $10^{15}$  bytes or 1000 terabytes) in some applications – “**Big Data**”.
- **Hadoop** and **MapReduce** approach to distributed data as well as the **Google File System** have given rise to Big Data technologies. Further enhancements are taking place in the form of **Spark** based technology.
- **NoSQL/NewSQL/Distributed SQL** systems have been designed for rapid search and retrieval from documents, processing of huge graphs, and other forms of data with flexible models of transaction processing across the globe.
- **True value of data** can now be revealed !!

## When NOT to Use a DBMS



- Main **inhibitors (costs)** of using a DBMS:
  - High initial investment and possible need for additional hardware. (Better with cloud)
  - The generality that a DBMS provides for defining and processing data
  - Overhead for providing security, concurrency control, recovery, and integrity functions.
- When a DBMS may be **unnecessary**:
  - If the database and applications are simple, well defined, and not expected to change.
  - If access to data by multiple users is not required.

## When NOT to use a DBMS



- When a DBMS may be **infeasible**:
  - E.g.: In embedded systems where a general purpose DBMS may not fit in available storage
- When **no** DBMS may **suffice**:
  - If there are hard real-time requirements that may not be met because of DBMS overhead
  - If the database system is not able to handle the complexity of data because of modeling limitations
  - If the database users need special operations not supported by the DBMS

