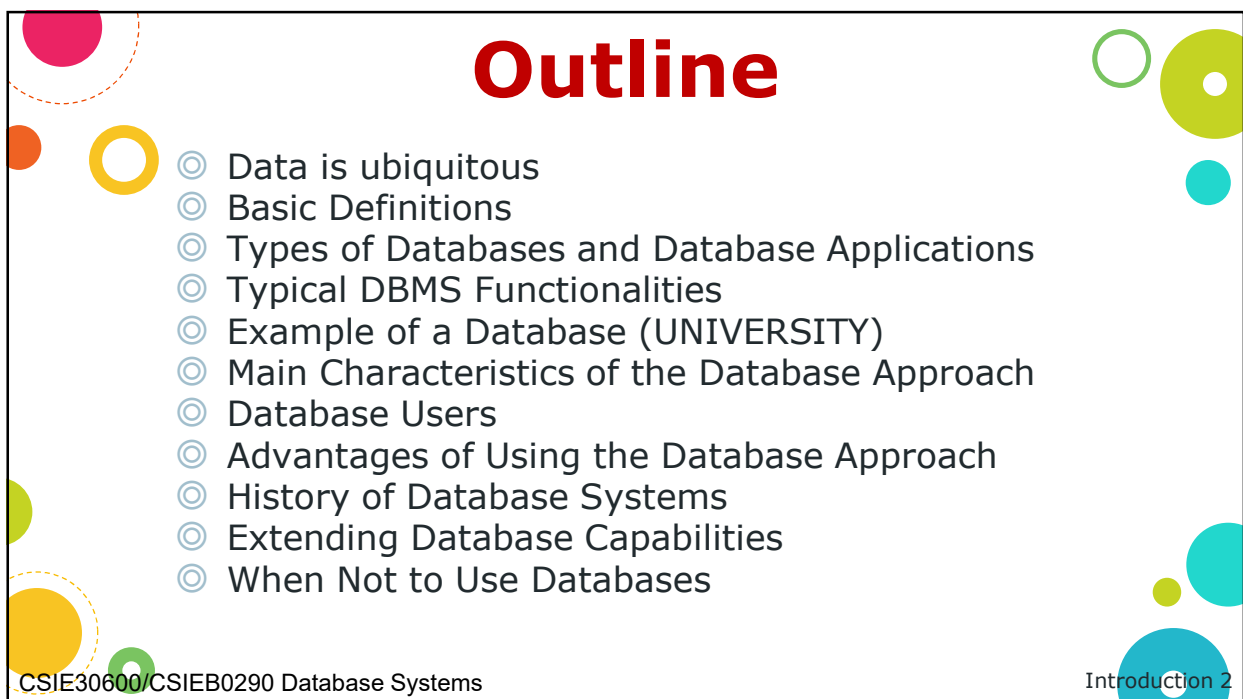




CSIE30600/CSIEB0290
Database Systems
**Lecture 1:
Introduction**

The slide features a decorative background with various colored circles (cyan, green, yellow, orange, pink) and a dashed line. In the bottom right corner, there is an icon of three blue database cylinders.



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

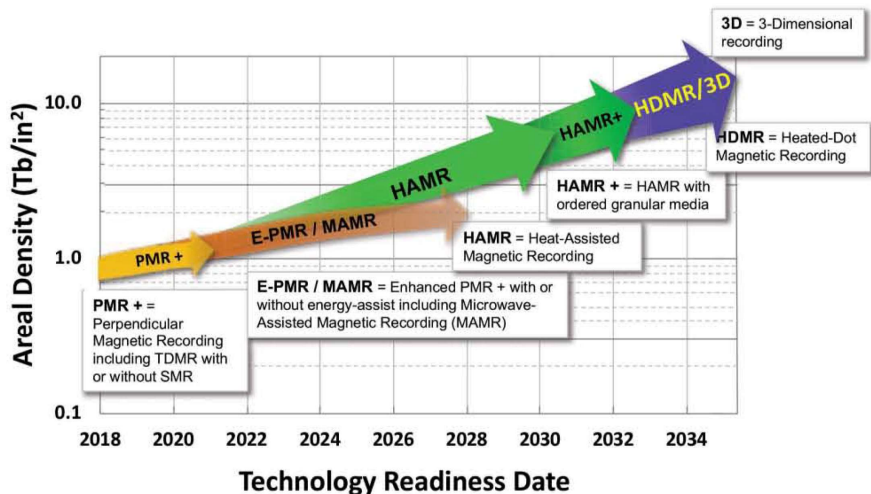
CSIE30600/CSIEB0290 Database Systems Introduction 2

The slide features a decorative background with various colored circles (pink, orange, yellow, green, cyan) and a dashed line.

Data vs 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 all kinds of data at real-time
- Clever **processing methods** (AI & machine learning)—allows us to process data more intelligently

HDD Density Growth



(<https://ieeexplore.ieee.org/document/9918580>, over 200TB HDDs by mid-2030's)

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

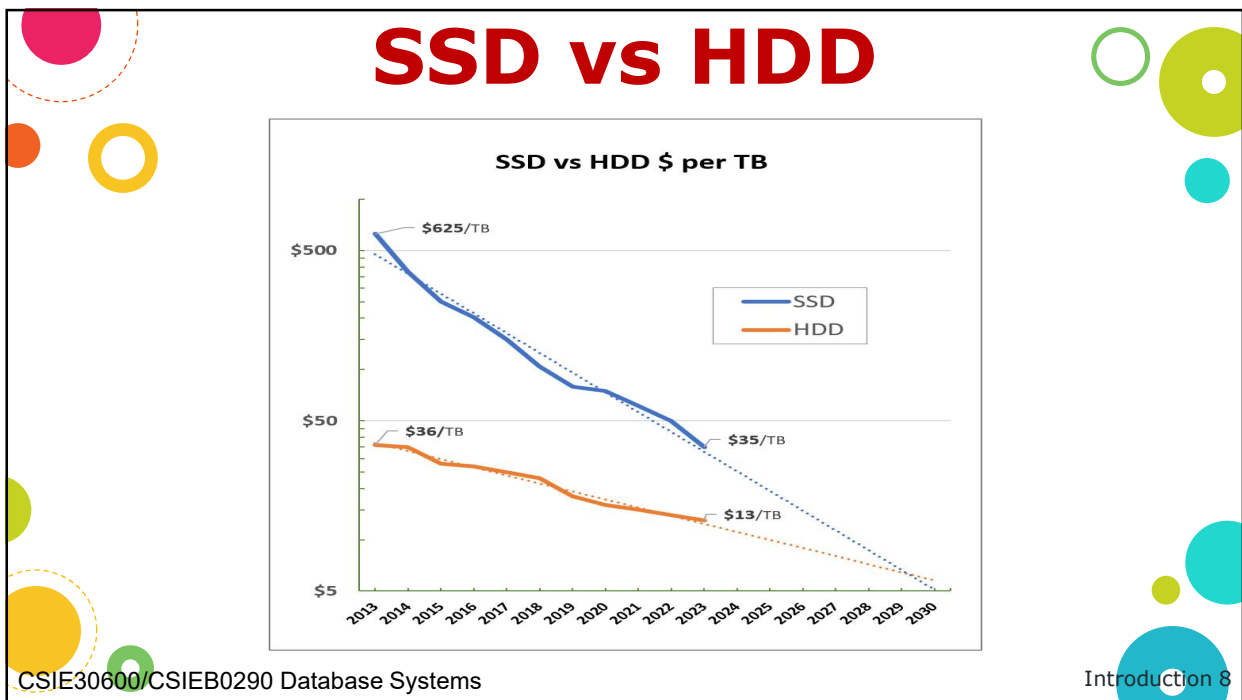
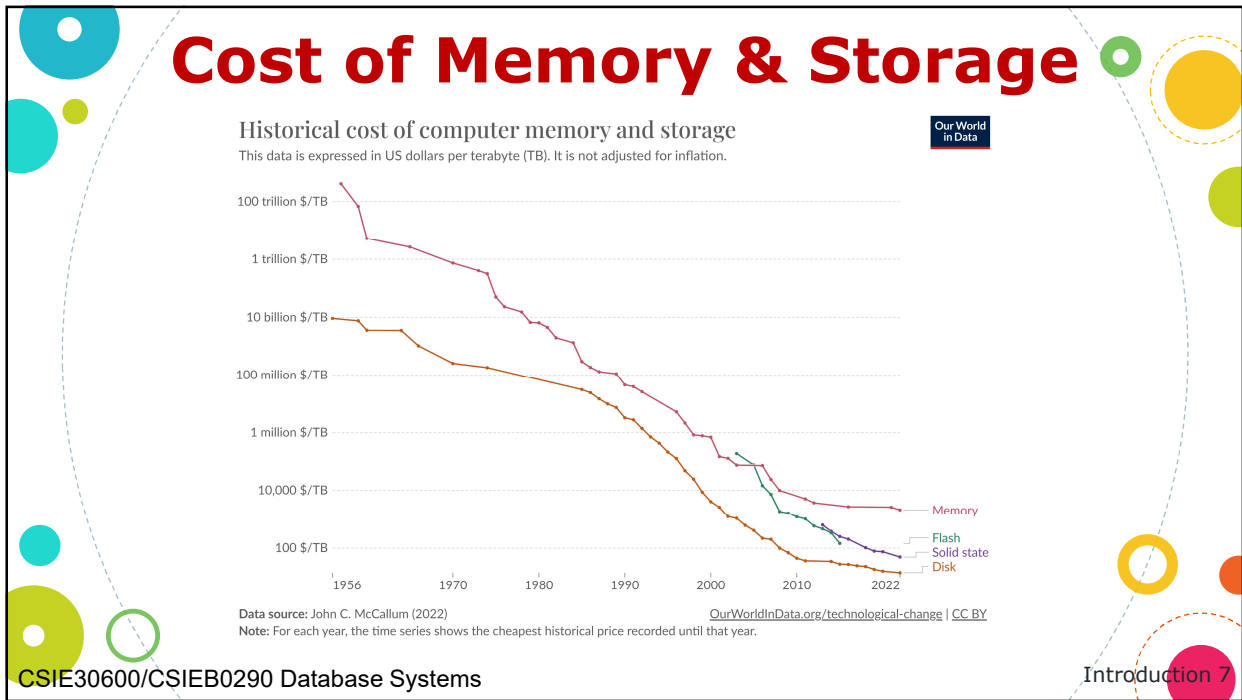
CSIE30600/CSIEB0290 Database Systems Introduction 5

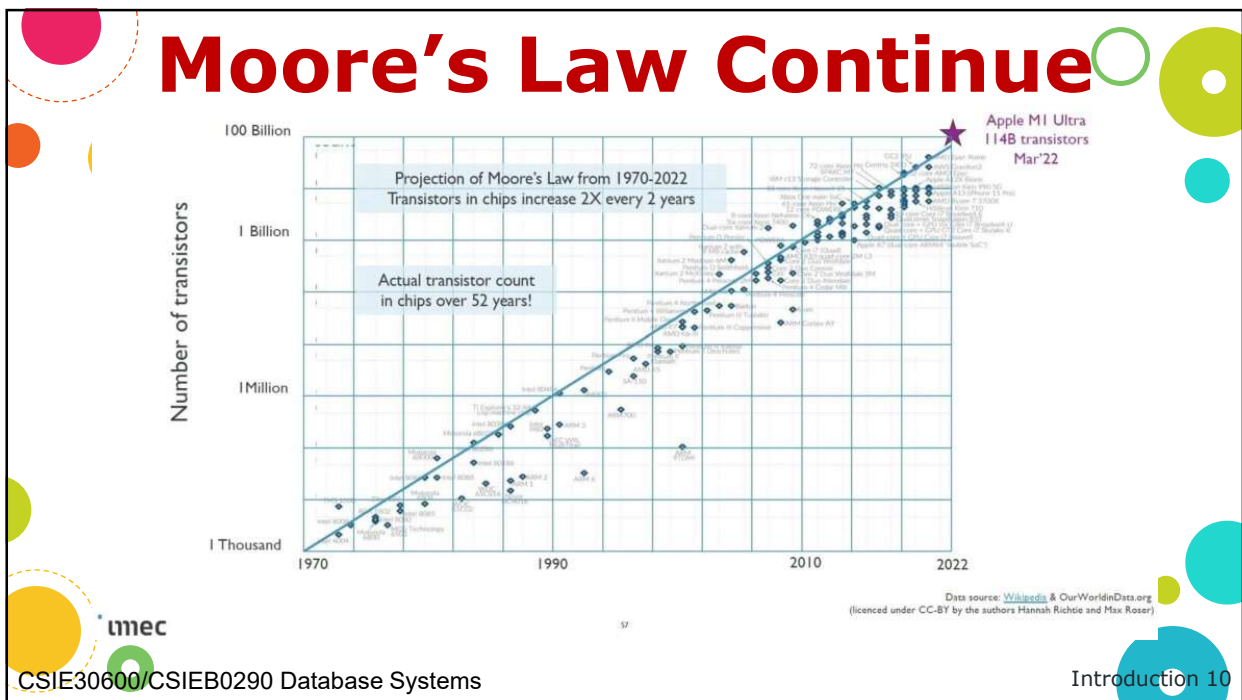
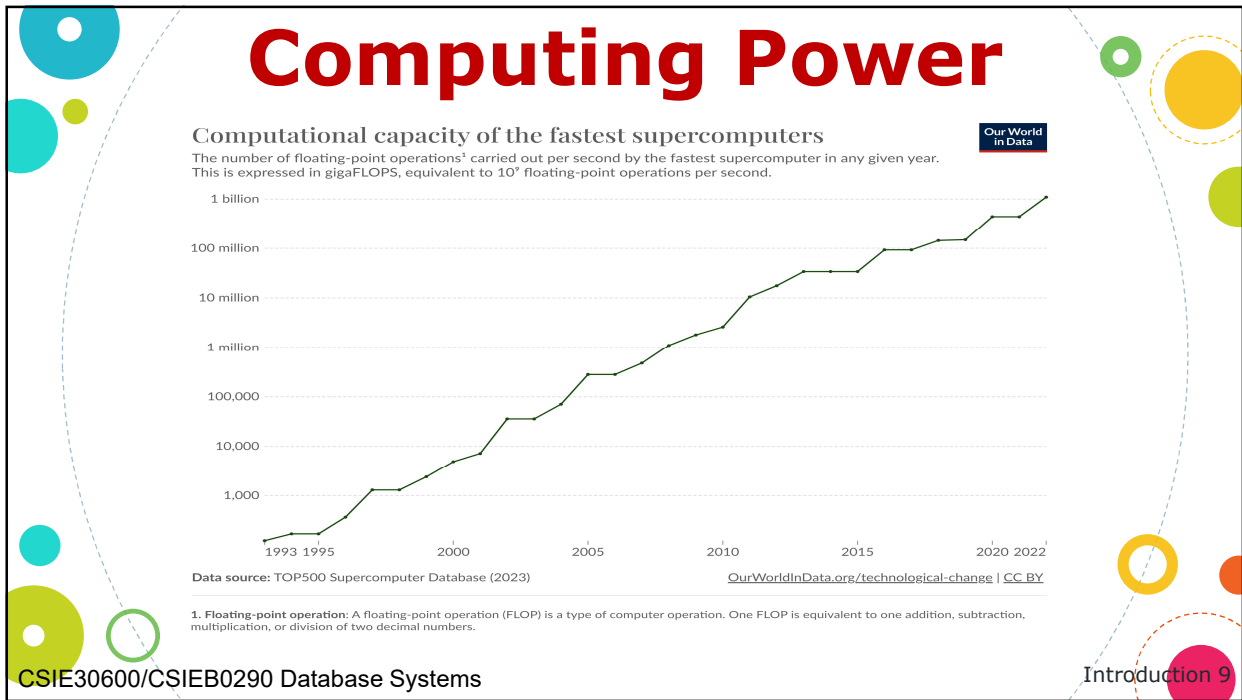
HDD Price per TB

Price per TB	Price	Size	Drive
\$14.00	\$279.99	20TB	Seagate Exos X20 ST20000NM007D 20TB 7200 RPM 256MB Cache 3.5" Internal Hard Drive
\$14.50	\$144.99	10TB	Toshiba X300 10TB Performance & Gaming Internal Hard Drive 7200 RPM SATA 6Gb/s 256MB Cache 3.5 inch - HDWR11AXZSTA (RETAIL PACKAGE)
\$15.00	\$119.99	8TB	TOSHIBA X300 HDWR480XZSTA 8TB 7200 RPM 256MB Cache SATA 6.0Gb/s 3.5" Desktop Internal Hard Drive Retail Packaging
\$15.37	\$245.99	16TB	Toshiba 16TB Enterprise HDD SATA 6.0Gb/s 512e 7200 RPM 512MB Cache 3.5" Internal Hard Drive MG08ACA16TE
\$15.71	\$219.99	14TB	Seagate Exos X18 ST14000NM000J 14TB 7200 RPM 256MB Cache SATA 6.0Gb/s 3.5" Hard Drives
\$16.07	\$224.99	14TB	TOSHIBA X300 HDWR31EXZSTA 14TB 7200 RPM 512MB Cache SATA 6.0Gb/s 3.5" Desktop Internal Hard Drive Retail Packaging
\$16.25	\$129.99	8TB	WD Blue WD80EAZZ 8TB 5640 RPM 128MB Cache SATA 6.0Gb/s 3.5" Internal Hard Drive
\$16.50	\$197.99	12TB	TOSHIBA X300 HDWR21CXZSTA 12TB 7200 RPM 256MB Cache SATA 6.0Gb/s 3.5" Internal Hard Drive
\$16.66	\$199.97	12TB	TOSHIBA X300 Pro HDWR51CXZSTB 12TB 7200 RPM 512MB Cache SATA 6.0Gb/s 3.5" Internal Hard Drive
\$16.67	\$299.99	18TB	Seagate 18TB Exos X18 7200 RPM SATA 6Gb/s 256MB Cache 3.5-Inch Enterprise Hard Drive HDD (ST18000NM000J)
\$16.87	\$269.99	16TB	TOSHIBA X300 HDWR31GXZSTA 16TB 7200 RPM 512MB Cache SATA 6.0Gb/s 3.5" Desktop Internal Hard Drive Retail Packaging
\$16.87	\$269.99	16TB	TOSHIBA N300 HDWG31GXZSTA 16TB 7200 RPM 512MB Cache SATA 6.0Gb/s 3.5" Internal Hard Drive
\$16.87	\$269.99	16TB	Seagate Exos 16TB Enterprise HDD X16 SATA 6Gb/s 512e/4Kn 7200 RPM 256MB Cache 3.5" Internal Hard Drive ST16000NM001G
\$16.87	\$269.99	16TB	Seagate Exos 16TB Enterprise HDD X18 SATA 6Gb/s 512e/4Kn 7200 RPM 256MB Cache 3.5" Internal Hard Drive (ST16000NM000J)
\$17.12	\$136.99	8TB	Seagate BarraCuda NE-ST8000DM004 8TB 5400 RPM 256MB Cache SATA 6.0Gb/s 3.5" Internal Hard Drive Bare Drive
\$17.14	\$239.96	14TB	TOSHIBA X300 Pro HDWR51EXZSTB 14TB 7200 RPM 512MB Cache SATA 6.0Gb/s 3.5" Internal Hard Drive

(https://edwardbetts.com/price_per_tb/, Last updated: 07 November 2023)

CSIE30600/CSIEB0290 Database Systems Introduction 6





Moor's Law at Intel

Intel internal analysis of Intel products.
Future projections based on products still in design.
Future transistor counts are projections and are inherently uncertain.

Aspiring to
1 Trillion
transistors in 2030

- ✓ RibbonFET
- ✓ PowerVia
- ✓ High NA
- ✓ 2.5D/3D packaging

intel.

CSIE30600/CSIEB0290 Database Systems Introduction 11

Network Bandwidth Growth

Nielsen's law: A high-end user's connection speed grows by **50% per year**.

Internet Connectivity (Bits Per Second)

1983 1988 1993 1998 2003 2008 2013 2018 2023

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
- ⊙ ...

New Types of Data

- ⊙ **Social Networks** (FB, IG, Twitter, Linked-In, ...) capturing a lot of information about **people** and **relationships**-posts, tweets, photos, videos, ...
- ⊙ **Search Engines** - Google, Bing, Yahoo : collect their own repository of web pages.
- ⊙ **Cloud services** – huge amount of data now resides on the cloud.
- ⊙ **Internet of Things (IoT)** generate real-time sensing and streaming data.
- ⊙ All of the above constitutes **new types** of **data**.

Data from IoT Sensors

Traffic Monitor
 Strain Gauge
 Airborne Imaging Device
 Health Monitor
 Industrial Process Monitor
 Webcam
 Satellite-borne Imaging device
 Environmental Monitor
 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 15

Effective Management Required

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

CSIE30600/CSIEB0290 Database Systems Introduction 16

Basic Concepts

- ⦿ **Data**: Known facts (recordable) with 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** and 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

417 systems in ranking, February 2024

Rank			DBMS	Database Model	Score		
Feb 2024	Jan 2024	Feb 2023			Feb 2024	Jan 2024	Feb 2023
1.	1.	1.	Oracle +	Relational, Multi-model f	1241.45	-6.05	-6.08
2.	2.	2.	MySQL +	Relational, Multi-model f	1106.67	-16.79	-88.78
3.	3.	3.	Microsoft SQL Server +	Relational, Multi-model f	853.57	-23.03	-75.52
4.	4.	4.	PostgreSQL +	Relational, Multi-model f	629.41	-19.55	+12.90
5.	5.	5.	MongoDB +	Document, Multi-model f	420.36	+2.88	-32.41
6.	6.	6.	Redis +	Key-value, Multi-model f	160.71	+1.33	-13.12
7.	7.	↑ 8.	Elasticsearch	Search engine, Multi-model f	135.74	-0.33	-2.86
8.	8.	↓ 7.	IBM Db2	Relational, Multi-model f	132.23	-0.18	-10.74
9.	9.	↑ 12.	Snowflake +	Relational	127.45	+1.53	+11.80
10.	↑ 11.	↓ 9.	SQLite +	Relational	117.28	+2.08	-15.38
11.	↓ 10.	↓ 10.	Microsoft Access	Relational	113.17	-4.50	-17.86
12.	12.	↓ 11.	Cassandra +	Wide column, Multi-model f	109.27	-1.77	-6.95
13.	13.	13.	MariaDB +	Relational, Multi-model f	97.23	-2.00	+0.42
14.	14.	14.	Splunk	Search engine	91.65	-1.07	+4.57
15.	↑ 16.	15.	Amazon DynamoDB +	Multi-model f	82.90	+1.96	+3.21
16.	↓ 15.	16.	Microsoft Azure SQL Database	Relational, Multi-model f	79.56	-1.51	+0.81
17.	17.	↑ 19.	Databricks +	Multi-model f	76.91	-3.62	+16.58
18.	18.	↓ 17.	Hive	Relational	65.81	-1.15	-6.31
19.	19.	↑ 22.	Google BigQuery +	Relational	63.63	+0.15	+11.17
20.	20.	↓ 18.	Teradata	Relational, Multi-model f	51.24	-1.94	-11.79

CSIE30600/CSIEB0290 Database Systems https://db-engines.com/en/ranking Introduction 19

DB Engines Ranking Trend

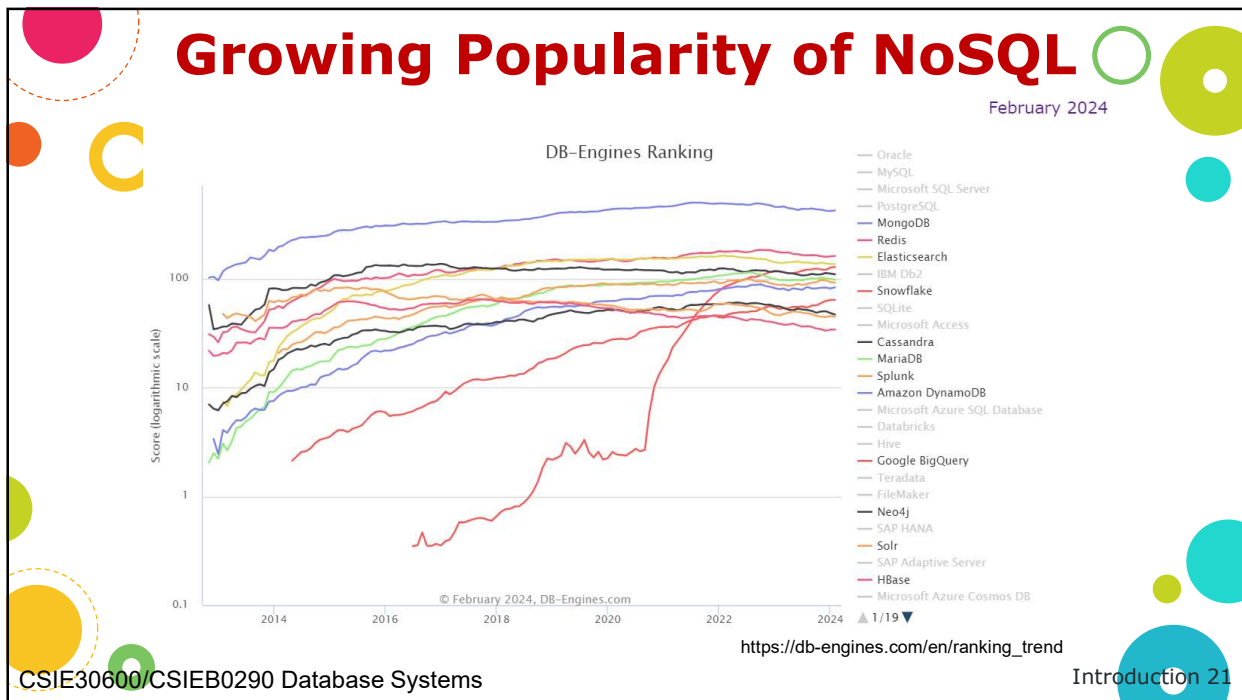
February 2024

DB-Engines Ranking

Score (logarithmic scale)

© February 2024, DB-Engines.com

https://db-engines.com/en/ranking_trend Introduction 20



Main Goals of DB Course

- ⦿ To learn **why**, **when**, and **where** DBs 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
 - ⦿ AI & ML

CSIE30600/CSIEB0290 Database Systems Introduction 22

Types of DBs and Applications

- ◎ **Traditional Applications:** Numeric and Textual DBs
- ◎ 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, 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

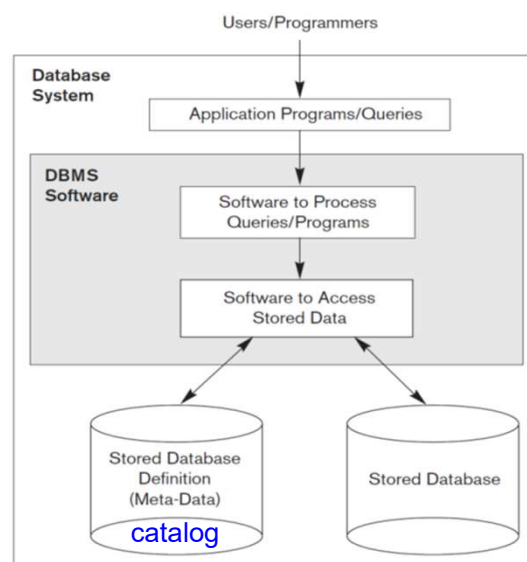
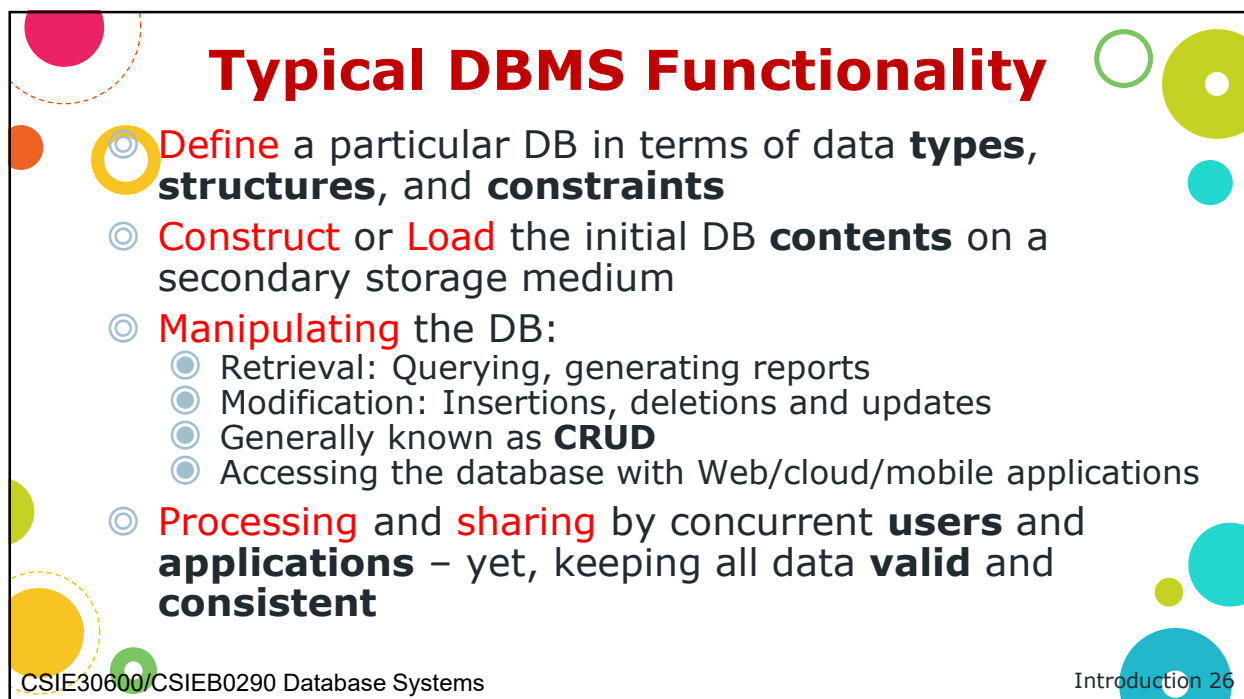
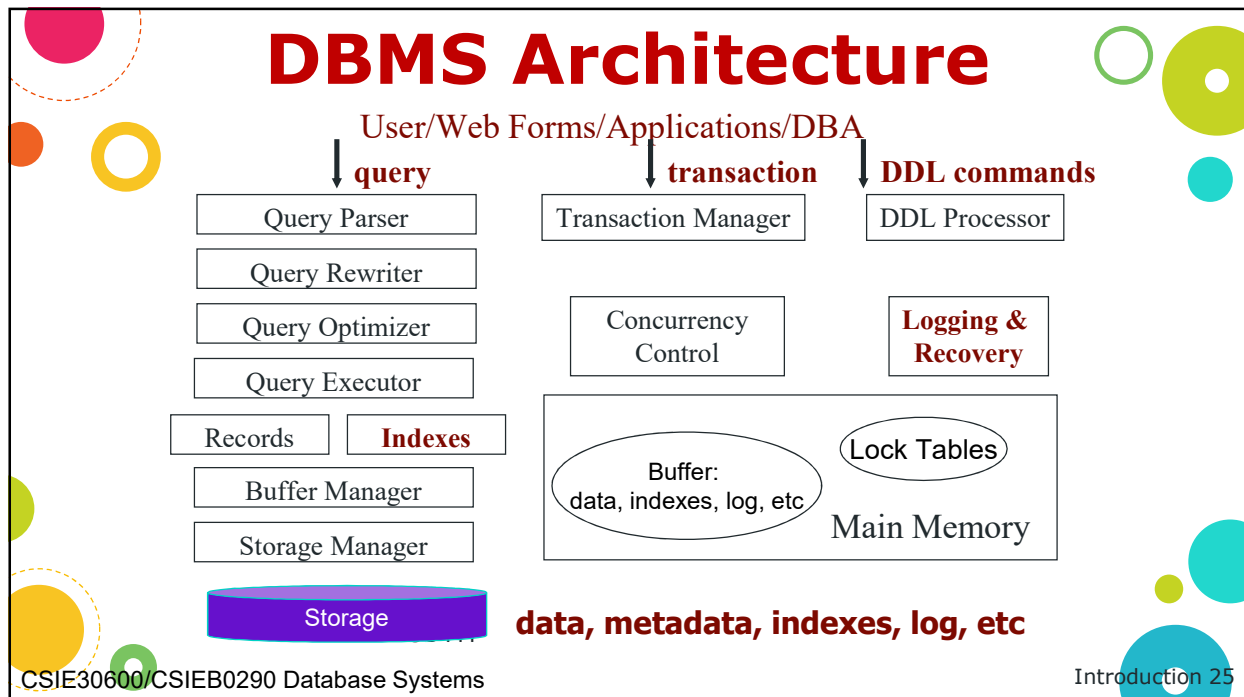


Figure 1.1
A simplified database system environment.



Typical DBMS Functionality

- Additional 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 applications
 - Called database, software, and system maintenance

Application Activities

- Applications interact with a DB 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:** orders, customized recommendations
- ⦿ **Manufacturing:** production, inventory, supply chain, ...
- ⦿ **Human resources:** employee records, salaries, tax, ...
- ⦿ **Internet & Web:** search, data management, ...
- ⦿ **Social media:** content management, relationships, ...
- ⦿ DBs are **closely related** to **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 systems and locations
 - ⦿ **Integrity** problems
 - Integrity constraints (e.g. account balance > 0) become “buried” in 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 DB in an inconsistent state
 - Example: Transfer of funds from one account to another
 - ⊙ **Concurrent** access by multiple users
 - Concurrent accessed needed for performance
 - Uncontrolled accesses can lead to inconsistencies
 - Example: reading and updating at the same time
 - ⊙ **Security problems**
 - Hard to provide user access to some, but not all, data
- ⊙ DB systems offer solutions to **ALL** the problems above.

DB Example

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



DB Example

- ⦿ 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: Entities and relationships are typically expressed in a **conceptual data model**, such as the **Entity-Relationship** model (to be discussed later in class)

CSIE30600/CSIEB0290 Database Systems
Introduction 33

DB Example

STUDENT

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

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

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

Figure 1.2
A database that stores student and course information.

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

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

CSIE30600/CSIEB0290 Database Systems
Introduction 34

Main Characteristics

- ① **Self-describing:** A DBMS **catalog (meta-data)** stores the description of the database. (next slide)
- ① **Program-data Independence:** Allows changing storage structures w/o changing DBMS access programs.
- ① **Data abstraction:** **Data models** hide storage details and present the users with a **conceptual view** of the DB.
- ① **Multiple views:** Each user may see a different view of the DB.
- ① **Data sharing:** among multiple users
- ① **Transactions, concurrent access, recovery, OLTP**

A Simplified DB Catalog

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.

Figure 1.3

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

DB Users

- Users may be divided into
 - Those who use and control the DB content, and those who design, develop and maintain DB 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).

DB Users

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

DB Administrator (DBA)

- ⊙ Coordinates **all** the **activities** of the DB system; must have a good understanding of the enterprise's information **resources** and **needs**.
- ⊙ DBA'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

End-users

- ⊙ Actors on the scene (continued)
 - ⊙ **End-users**: use the data for queries, reports and some of them update the 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 DB.
 - ⊙ 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 DB.
- **Stand-alone:**
 - Mostly maintain personal DBs using ready-to-use packaged applications.
 - An example is a tax program user that creates its own internal DB.
 - 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 software. Also known as **software developers** or **software engineers**.
- **Business analysts:** People who can analyze vast amounts of business data and real-time data for better decision making, planning, advertising, marketing etc.

Users behind the Scene

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

Advantages of Using DBs

- ◎ **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**) for efficient data access
- ◎ Providing **optimization of queries** for efficient processing

Advantages of Using DBs(cont.)

- ⦿ Providing **backup** and **recovery** services.
- ⦿ Providing **multiple interfaces** to different users.
- ⦿ Representing **complex relationships among data**.
- ⦿ Enforcing **integrity constraints** on the DB.
- ⦿ 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 DB 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:
 - ⦿ DB 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
 - ⦿ Hard to share

DB History (cont.)

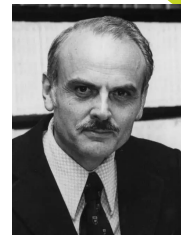
The '60s

- **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 F. 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/Fog/Edge Databases
- AIDB, MLDB (DB + AI&ML)

Extending DB Capabilities

- New functionalities are added to DBMSs in new areas:
 - Scientific Applications –Physics, Chemistry, Biology -Genetics
 - Earth and Atmospheric Sciences and Astronomy
 - XML (eXtensible Markup Language), JSON
 - Image Storage and Management
 - Audio and Video Data Management
 - Data Warehousing and Data Mining –a very major area for future development using new technologies
 - Spatial Data Management and Location Based Services
 - Time Series and Historical Data Management
- The above gives rise to new research and development in DB systems.

New Trends

- ◎ First decade of the 21st century has seen **tremendous growth** in **user generated data** and **automatically collected data** from APPs, search engines, sensors...
- ◎ **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
- ◎ **IoT** provides real-time sensing and streaming data
- ◎ **AI&ML** need to process huge amount of data

Big Data, NoSQL, NewSQL, Distributed SQL

- ◎ New data storage, management and analysis technologies were necessary for huge volume of data in PB a day (10^{15} bytes or 1000 TBs) – “**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 by **Spark**.
- ◎ **NoSQL/NewSQL/Distributed SQL** systems for rapid search and retrieval, processing 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

