AY2023/24 Trimester 1

Dr. Tong Rong

INF2003
Database Systems

Instructors

Asst/P Tong Rong
Module lead
tong.rong@singaporetech.edu.sg

Asst/P Zhang Wei
Co- Module lead
Wei.Zhang@singaporetech.edu.sg

Ms. Cherine Ng Siew Eng
Lab instructor – Dover
Cherine.Ng@Singaporetech.edu.sg

Mr. Chan Nai Tiong
Lab instructor – NYP
naitiong.chan@singaporetech.edu.sg

Ms. Yang Changjun
Lab instructor – NYP
changjun.yang@singaporetech.edu.sg

Mr. Jonathan Lim Yu Kuang
Lab instructor – NYP, Dover
IF_Jonathan.Lim@singaporetech.edu.sg

Schedule – first half

	Thursday 9:00-11:00 Lecture: Zoom	Thursday/Friday
Week 1	Introduction about DB and relational DB	Lab1 (Dover Thursday 4-6 PM /NYP week 2 Friday 4-6PM)
Week 2	Relational algebra and SQL	Lab2 (Dover Thursday 4-6 PM /NYP Friday 9-11 AM)
Week 3	SQL	Lab3 (Dover Thursday 4-6 PM /NYP Friday 9-11 AM)
Week 4	ER model	Lab4 (Dover Thursday 4-6 PM /NYP Friday 9-11 AM)
Week 5	DB refinement	Tutorial (online, Friday 4-6 PM)
Week 6	No lecture, Quiz 1 Oct 6 Friday 4-6 PM (all contents in week 1-5)	DV-AP-SR3K, NYP-LT1A,NYP-LT2B, NYP-LT3B,NYP-LT4A, NYP-SR5D,NYP-SR6G, NYP-SR6H

Schedule – labs

Time	Venue	Group	Instructor
Thursday 4-6 PM	Dover, DV-AP-SR1L	P1,P2	Cherine Ng Siew Eng, Jonathan Lim Yu Kuang
Friday 9-11 AM	NYP-SR4A	P3	Chan Nai Tiong
	NYP-SR4G	P4	Yang Changjun
	NYP-LT4A	P5,P6	Jonathan Lim Yu Kuang

Lab work to be submitted by the following Monday night 11:30PM when the lab is conducted

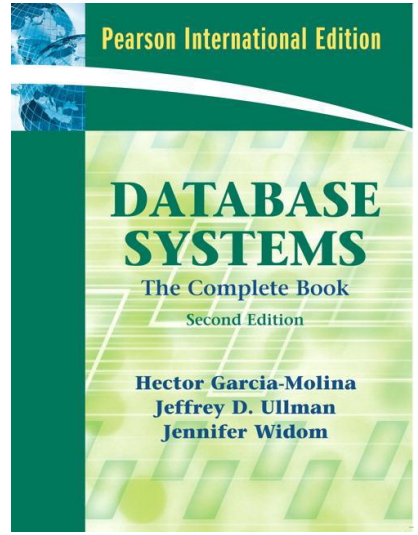
Lab1 (NYP, P3-P6) rescheduled to week 2 Friday (Sep 8) 4-6 PM

- **deadline same as lab2**
- **rooms:**
  - **P3: SR5C**
  - **P4: SR5D**
  - **P5 & P6: SR6G**

Assessments schedule

Assessment	Description	Weights
Lab	Participation, submission: 8 labs (first half 4, second half 4) 2.5% each	20%
Quiz 1	Week 6, contents in week 1-5	25%
Quiz 2	Week 14, contents in week 8-12	25%
Project (group)	Proposal, presentation, report, code, peer assessment	30%

Textbook



- Garcia-Molina, H., Widom, J. and Ullman, J., *Database Systems: the Complete Handbook, 2nd Edition,* Prentice Hall, 2008.

Recommended Resources

- Avi Silberschatz, Henry F. Korth and S. Sudarshan, *Database System Concepts*, *McGraw-Hill, 2019.*
- Ramakrishnan, R., Gehrke, J. *, Database Management Systems, 3rd Edition, McGraw-Hill, 2002.*
- Online: Coursera, Udacity

Learning outcome – first half

- Describe the differences between a flat file and a relational database
- Understand relational algebra and use it to perform database manipulation
- Model data and processes using an ER diagram and derive the relational schema
- Design and implement a relational database from a relational schema, including multiple tables and queries
- Use SQL queries to perform CRUD (create, read, update, and delete) operations information from a relational database
- Understand database optimization criteria and apply in database implementation

Database Intro

Motivations to Learn DBMS

- Data is everywhere, explosively increasing.
  - Organizations: bank, university, airlines, telecom etc
  - At Least 1200 PB (1PB=1000TB) of data are stored and shared in digital media daily (Facebook, Google, Amazon, Microsoft, etc.)* *https://earthweb.com/how-much-data-is-created-every-day/
- Managing data is the key and foundation to almost any application in the world.
- DBMS technology encompasses broad areas in CS :OS, languages, AI, multimedia, logic, …
- Four people have won Turing Awards in this area:

Charles Bachman, 1973
IDS and CODASYL

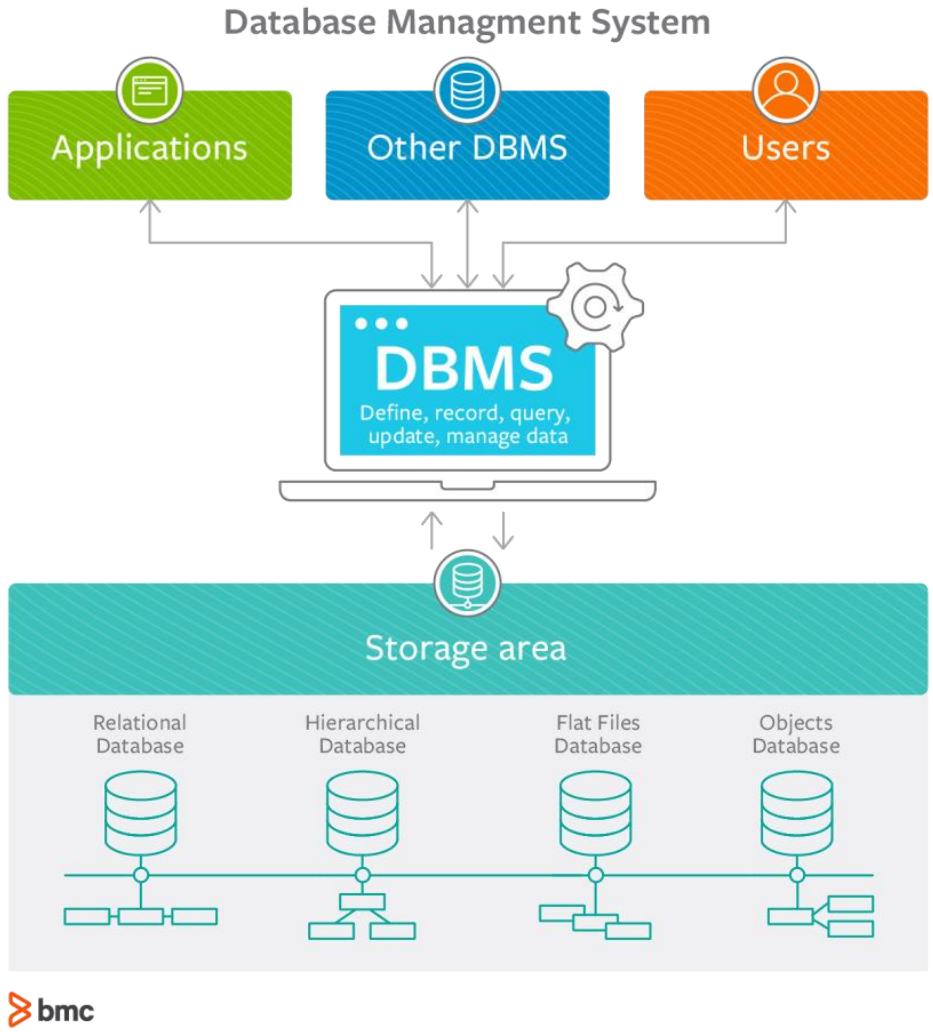
Ted Codd, 1981
Relational model

Jim Gray, 1998
Transaction processing

Michael Stonebraker, 2014
INGRES and Postgres

Database

- Database (**DB**)
  - An organized collection of information.
- Database Management System (**DBMS**)
  - A software package designed to store and manage databases.
  - Provides data processing environment that is both convenient and efficient to use.
  - Address all complications in data management



https://s7280.pcdn.co/wp-content/uploads/2018/08/database-management-system-810x898.png

Why DBMS: advantages over flat file

- Data independence and efficient access
  - DB independent to the application: logical design, physical storage
  - Efficient access: easy retrieval, large file index, query optimization
- Reduced application development time
  - DB system provide query facilities
- Data integrity and security
  - Integrity constraints: validate data update request
  - DBMS provides access control
- Uniform data administration
  - Maintenance and data admin: fine tuning data presentation, backups etc
- Concurrent access and crash recovery

DBMS – Advantages

Scenarios:

1: Given a student ID, how do I find the student, and the courses he/she is taking?

- We need: **efficient** data access.

2: I have multiple people making changes to the data, how do I ensure that the data is in a consistent state? How to solve the problem of concurrent access?

- We need: data **integrity** and **concurrent** access.

3: How do we enforce different users to perform different operations on subsets of data?

- We need: **security** / access control.

Do we need DBMS everywhere?

DBMS – Data Abstraction

View level (external) **user, developer**

user performs queries and operations, often in a software.

Logical level (conceptual) **developer, DB admin**

communication protocol, define the abstracted structure: how the data pieces relate to one another.

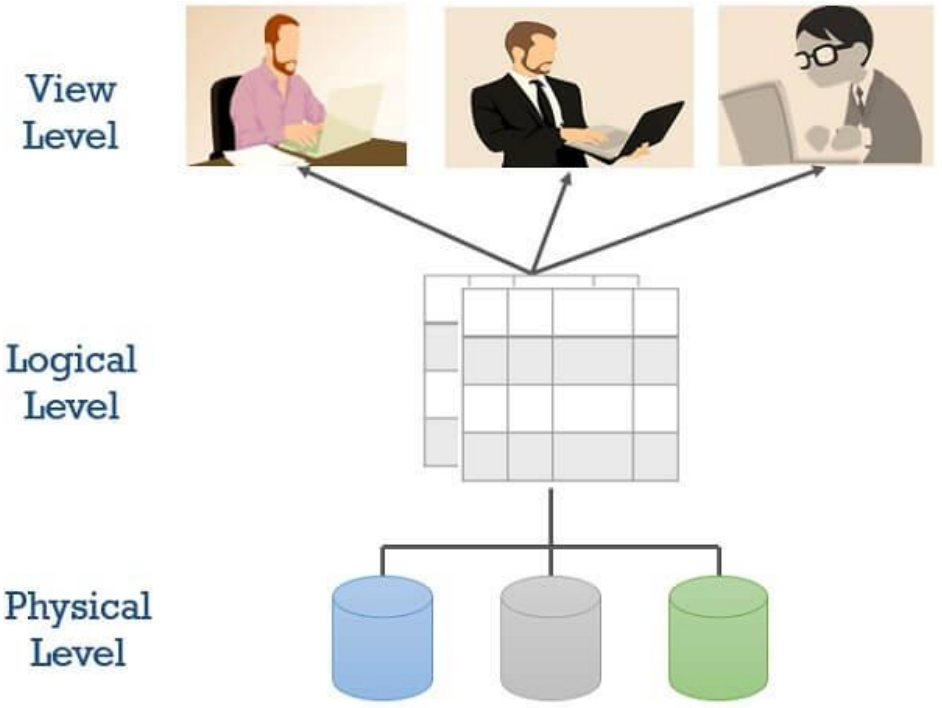
Physical level (internal) **DB admin**

how data stored in the disk, memory, cache, etc.

Data is relatively isolated in different levels

⬇

Data independence.



Three-Schema Architecture

https://binaryterms.com/view-of-data.html

DBMS – Data Independence

Physical independence	Change the physical characteristics without affecting the conceptual level • Modify the physical schema • Data is stored on disk or magnetic Tapes • Changes to compression techniques or hashing algorithms.
Logical independence	Change the logic structure of the data without affecting the other layers of the database • Add/modify/delete a new attribute, entity or relationship is possible without a rewrite of existing application programs • Merging two records into one • Breaking an existing record into two or more records

Which one is more difficult?

Data Independence example - A simple university database

View level

```
class student {
    char name[30];
    int id;
    …
};
```

Logical level

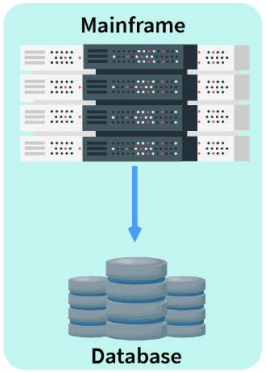
- Students(Sid, Name, Age, Gpa)
- Module(Mid, Mname, Credit)
- Enroll (Sid, Mid, Grade)

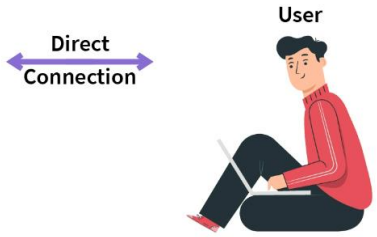
Physical level

- Relations stored as unordered files
- Index on student ID

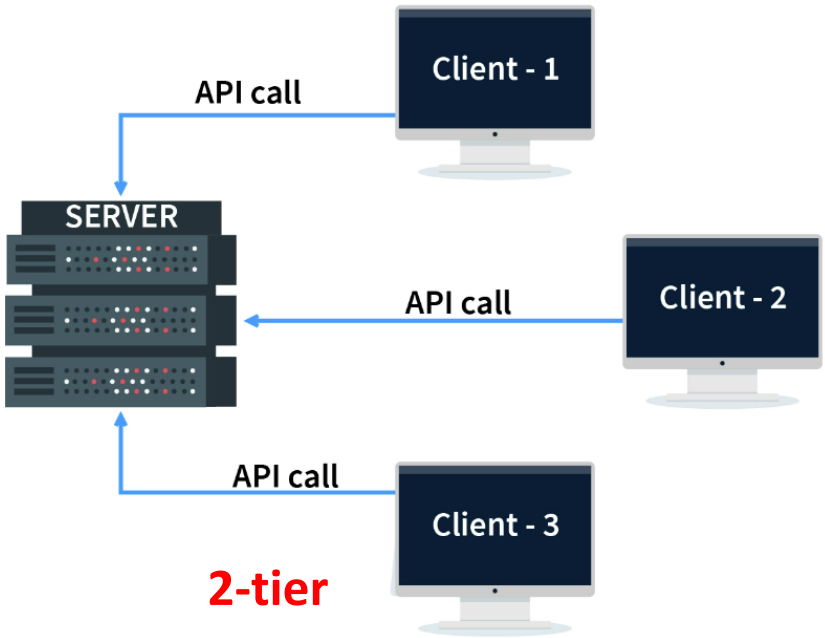
Data on disk

DBMS architecture

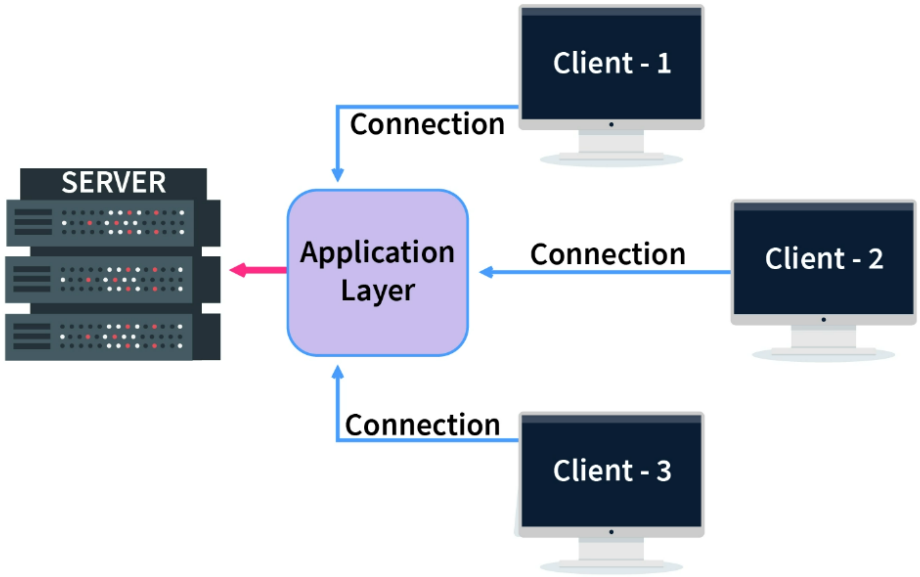




1-tier



2-tier

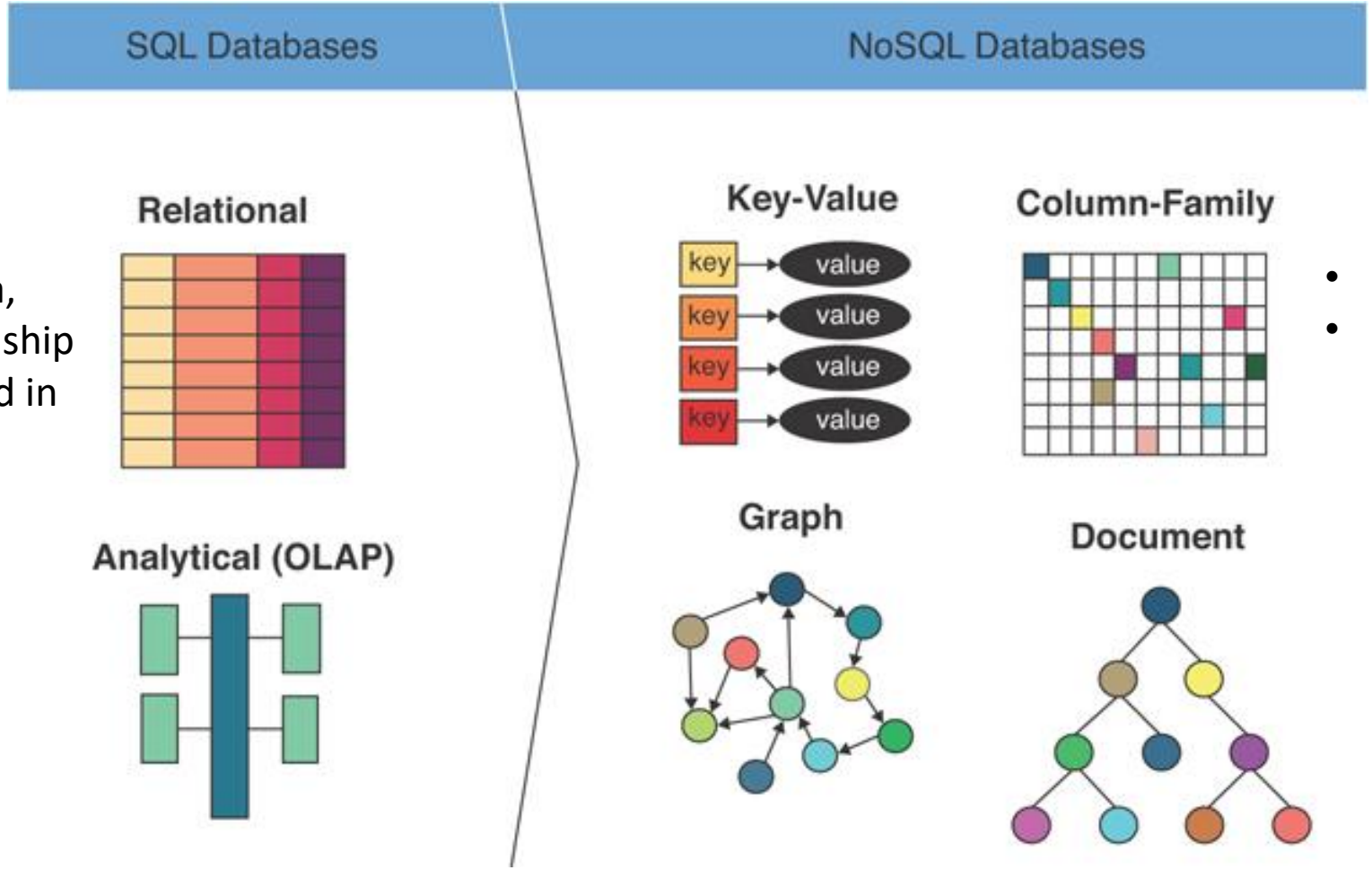


3-tier

https://www.scaler.com/topics/dbms/dbms-architecture/

Types of DBMS

- Structured data, explicit relationship
- Commonly used in enterprises



- Unstructured data
- Used in big data and real-time web applications

https://ptgmedia.pearsoncmg.com/images/chap4_9780135853290/elementLinks/04fig04_alt.jpg

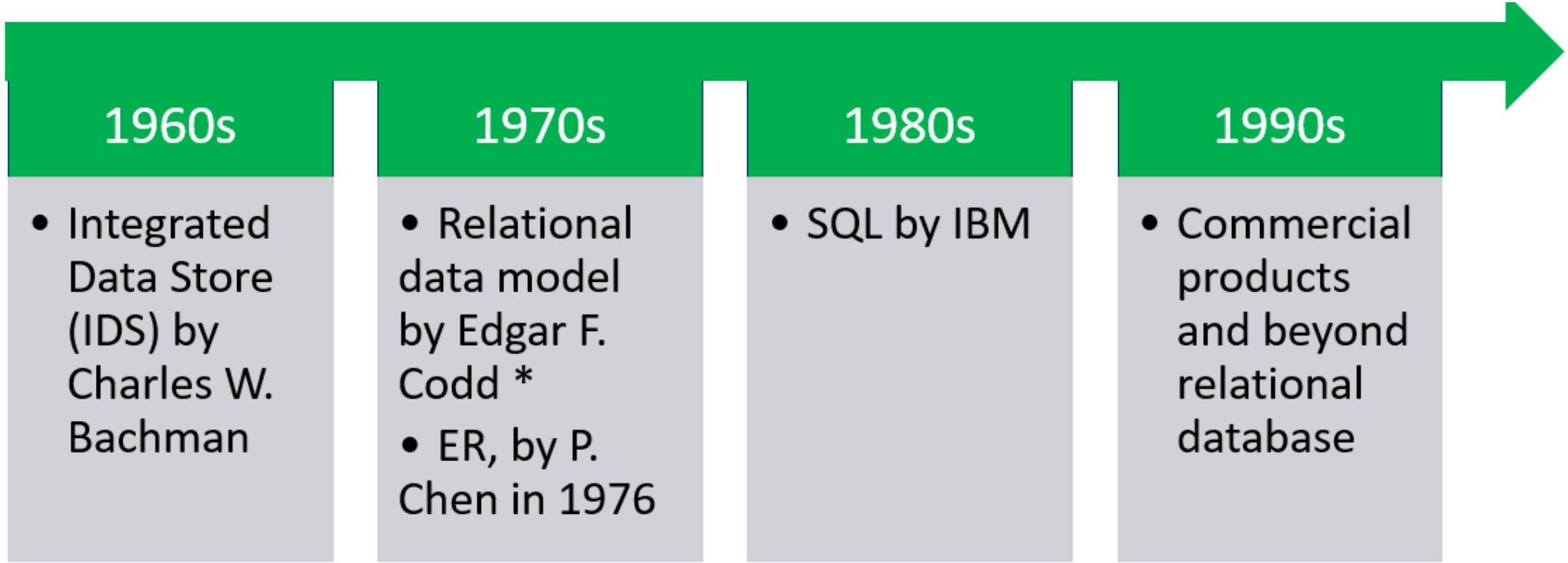
Popular database

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

414 systems in ranking, April 2023

Rank			DBMS	Database Model	Score		
Apr 2023	Mar 2023	Apr 2022			Apr 2023	Mar 2023	Apr 2022
1.	1.	1.	Oracle	Relational, Multi-model	1228.28	-33.01	-26.54
2.	2.	2.	MySQL	Relational, Multi-model	1157.78	-25.00	-46.38
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	918.52	-3.49	-19.94
4.	4.	4.	PostgreSQL	Relational, Multi-model	608.41	-5.41	-6.05
5.	5.	5.	MongoDB	Document, Multi-model	441.90	-16.89	-41.48
6.	6.	6.	Redis	Key-value, Multi-model	173.55	+1.10	-4.05
7.	7.	↑ 8.	IBM Db2	Relational, Multi-model	145.49	+2.57	-14.97
8.	8.	↓ 7.	Elasticsearch	Search engine, Multi-model	141.08	+2.01	-19.76
9.	9.	↑ 10.	SQLite	Relational	134.54	+0.72	+1.75
10.	10.	↓ 9.	Microsoft Access	Relational	131.37	-0.69	-11.41
11.	↑ 12.	11.	Cassandra	Wide column	111.81	-1.98	-10.19
12.	↓ 11.	↑ 14.	Snowflake	Relational	111.12	-3.27	+21.68
13.	13.	↓ 12.	MariaDB	Relational, Multi-model	95.93	-0.90	-14.38
14.	14.	↓ 13.	Splunk	Search engine	85.44	-2.54	-9.81
15.	↑ 16.	15.	Microsoft Azure SQL Database	Relational, Multi-model	79.06	+1.62	-6.72
16.	↓ 15.	16.	Amazon DynamoDB	Multi-model	77.45	-3.32	-5.46
17.	17.	17.	Hive	Relational	71.65	+0.74	-9.77

Relational Database –brief history



* Codd's 12 rules: A DB that follows the rule is referred to as a real relational DB management system (RDBMS).

Relational DB standard in transaction: ACID

Atomicity	• The entire transaction takes place at once or not at all.
Consistency	• Database must be consistent before and after transaction
Isolation	• Multiple transactions occur independently without interference
Durability	• Once a transaction is committed, considered permanent, even when a system failure

https://en.wikipedia.org/wiki/ACID

By Jim Gray

Database design

Stages of Database Design

S1
- Requirement Analysis

S2
- Conceptual Database Design

S3
- Logical Database Design

S4
- Refine the Schemas

S5
- Physical Database Design

S6
- Application and Security Design

Database Design stage 1 - requirement analysis

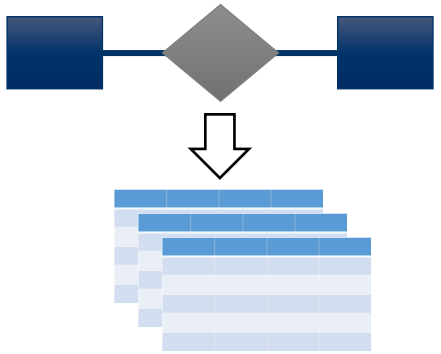
- Bidirectional:
  - Our **clients tell** what they want. Normally not, or even far from complete.
  - We **developers** shall **ask and clarify**: what are the needs of our customers?
- What **data** to be stored? Numerical or text?
- What kind of **applications** to use the data? Online shopping or banking?
- What **operation** needs to be performed? Read intensive? On many tables?

Database Design stage 2 – conceptual design

- Given user requirements, what **tools** to model the requirements before creating the relevant tables in a DBMS?
  - Create one relation for all modules and students? Or create multiple?
- Popular option: Entity Relationship (**ER**) **Diagram**.
  - Requirement analysis information -> develop a high-level **description** of the data.
  - Understand what are the **constraints** that need to be modeled.

Database Design stage 3 – logical design

- ER -> relational database schemas.
  - i.e., what are the columns and what are the rows.
- Database schema: blueprint or architecture of data
  - All important or relevant data
  - Consistent formatting for all data entries
  - Unique keys for all entries and database objects
  - Each column in a table has a name and data type
- ER model is most relevant to the first three stages.

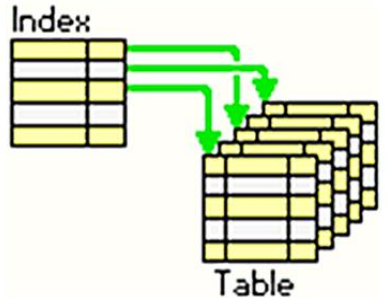


Database Design stage 4 – schema refinement

- Analyze the relations -> identify potential problem -> refine it.
- Schema refinement (normalization): the process to reduce data redundancy and improve data integrity.
  - Redundancy: repetition of same data or duplicate copies of same data
  - Anomalies: the problems occurred after poorly planned
- Most popular technique: decomposition
  - decomposing tables to eliminate data redundancy

Database Design stage 5 – physical design

- The process of transforming a logical data model into a physical model
- Physical design is optimized for data-access, performance requirements and other constraints of the target environment, i.e. hardware and software.
  - Consider typical workloads
  - Build indexes on tables and cluster some tables.



https://www.javatpoint.com/types-of-databases

Database Design stage 6 – application & security design

Security design protecting a database from

- unauthorized access
- malicious destruction
- any accidental loss or misuse.

- Authorization: identify what is accessible or inaccessible.
- Access control: enforce access rules, i.e., guest and admin.
- Backup and recovery: recover the database from failure or damage
- Encryption: protect highly sensitive data

https://www.netsolutions.com/insights/the-ultimate-guide-to-database-se

Relational data model

Relational Data Model

- Relational database, a collection of relations, each has a unique name
- Relation, a table with rows and columns
  - Each row in the table specifies a relationship between the values in that row
  - Simple data representation.
  - Easy query, expressing what you want, i.e., which column which row.
- Relations are unordered:
  - Allows optimized storage

Note: SQL ≠ relational data model

Relational Data Model (cont.)

- Relation **STUDENT**
  - Table STUDENT has 3 rows, or tuples.
  - Each row has 3 columns, or attributes.
- Each tuple specifies the relationship of the attributes
  - Fixed size, with the same number of elements.
- Each attribute has a domain
  - Specifies the set of valid values for the attribute
  - Attribute values are (normally) required to be **atomic**;

  that is, indivisible, e.g.,
    - sid is the set of 1 char + 7 digits

STUDENT

sid	sname	saddress
S2022001	John Snow	Blk 123 Ang Mo Kio
S2022002	Arya Stark	Blk 45 Sengkang
S2022003	Sansa Stark	Blk 108 Bukit Batok

Relational Schema

- Relation schema: every relation has a schema
  - Specifies the type information for relations, i.e., attribute name and domain.
  - Different relations can share the same schema, i.e., year 1 year 2 students.
- A relation schema includes:
  - a set of attributes
  - the domain of each attribute
- Notation: ***r(R)*** - the schema of ***r*** is ***R***.
  - STUDENT(sid, sname, saddress)
    - ***r***: STUDENT
    - ***R***: (sid, sname, saddress)
  - Domains are not stated explicitly in this notation!

STUDENT

sid	sname	saddress
S2022001	John Snow	Blk 123 Ang Mo Kio
S2022002	Arya Stark	Blk 45 Sengkang
S2022003	Sansa Stark	Blk 108 Bukit Batok

Domain - NULL

- The special value **NULL** is a member of every domain. Indicated that the value is unknown or unspecified
- A placeholder to denote values that are missing or do not know
- The null value causes complications in the definition of many operations
  - Operations with **NULLs** are quite troublesome
    - *NULL>100? NULL *100 =?*
    - avoid using **NULL** if possible.
  - In DBMS, we can specify **default values**, i.e.,
    - if not specified, record 0.
    - cannot find a page? give error 404.

https://blog.sqlauthority.com/wp-content/uploads/2007/06/null-500x259.png

Avoid Duplications – Keys

- A database contains many records, how to fetch a particular one?
- Keys are used to distinguish individual tuples
- Definition: a certain minimal subset of relation attributes uniquely identifies a tuple.
  - Keys define the meaning of the relation.
  - All relations have a key, some may have multiple keys
  - Unique: two distinct tuples cannot have same values in all key attributes.

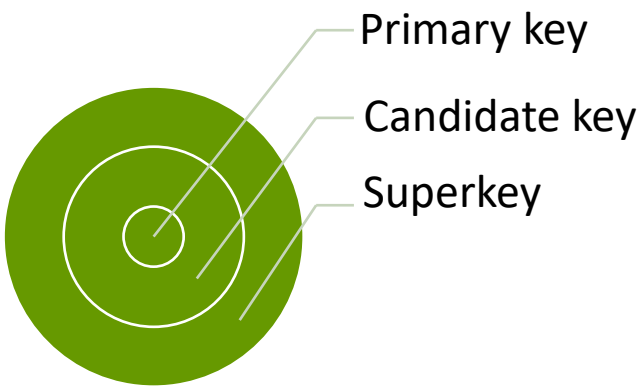
Avoid Duplications – Keys (cont.)

- **Superkey**: a set of attributes within a table that can uniquely identify each record

  sid / sid, sname /sphone/sid, address/ … How about saddress?

  - Not all superkeys are equally useful…
- **Candidate key**: the minimal set of fields which can uniquely identify each record
  - There can be more than one candidate key. sid / sphone
- **Primary key**: a candidate key that is most appropriate to become the main key: sid
  - How to indicate Primary key? -> underline it.

    STUDENT (sid, sname, saddress, sphone)



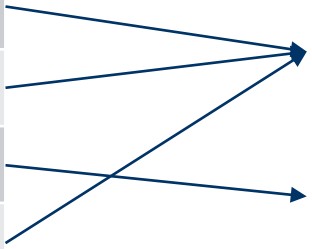
sid	sname	saddress	sphone
S2022001	John Snow	Blk 123 Ang Mo Kio	91234567
S2022002	Arya Stark	Blk 45 Sengkang	91234568
S2022003	Sansa Stark	Blk 108 Bukit Batok	91234569

Foreign Keys

- To link the data among relations/tables
  - Foreign key is the primary key of the other relations.
  - Referencing relation and referenced relation, Like a logical pointer.
- Example:
  - sid is a foreign key referring to STUDENT:

Enrolled: Referencing relation

sid	cid	grade
S2022001	ICT1002	C
S2022001	ICT1009	B
S2022003	ICT2103	A
S2022001	ICT2103	B



STUDENT (Referenced relation)

sid	sname	saddress
S2022001	John Snow	Blk 123 Ang Mo Kio
S2022002	Arya Stark	Blk 45 Sengkang
S2022003	Sansa Stark	Blk 108 Bukit Batok

Recap

- Introduction
  - General information
  - DBMS basics
- Stages of Database Design
  - Six stages
  - Often many rounds
- Relational data model
  - Concepts
  - Keys

Lab 1

- Google colab: tutorial uploaded in Xsite
- Python-based Database
  - Objectives: to learn the basic database operations via Python and SQLAlchemy.
  - SQLAlchemy allows us to bypass the SQL statements and perform the database operations in Python
  - Colab/Jupyter notebook

About Project

Identify an application

- an application that involves two types of databases: relational and non-relational

Identify datasets

- Real-world data: download, crawl etc
- Synthesized data if required

Application and DB design

- Both Relational db and NoSQL db
- Relatioanl db: ER diagram, NoSQL: schema and data model

Implementation

- Basic functions: CRUD
- Highlight DB design, complex DB related functions for the application

Analysis and Reflection

- Performance analysis
- Problems and improvements

About Project

- Group project, 5-6 students/group (251 students), self enrolment, by Sep 11, 11:30PM

Deliverable	**Due and Submission (strictly no extension)**
Project proposal (2 pages) Brief description of the application Data, system architecture, software requirements, implementation plan, team member task assignment	Week 5 Oct 1 11:30 PM XSite group Dropbox: Single file within 2 pages
Project final report (template provided)	Week 13 Nov 26, 11:30 PM ? XSite group dropbox, a .zip file contains: final report: *.pdf Presentation slides: *.ppt Presentation video: *.mp4 Source code: zip file
Project presentation slides	
Video presentation (10 mins)	
Source code with user manual	
Peer review	