CSE-221

Database Management Systems

Introduction to DBMS Concepts

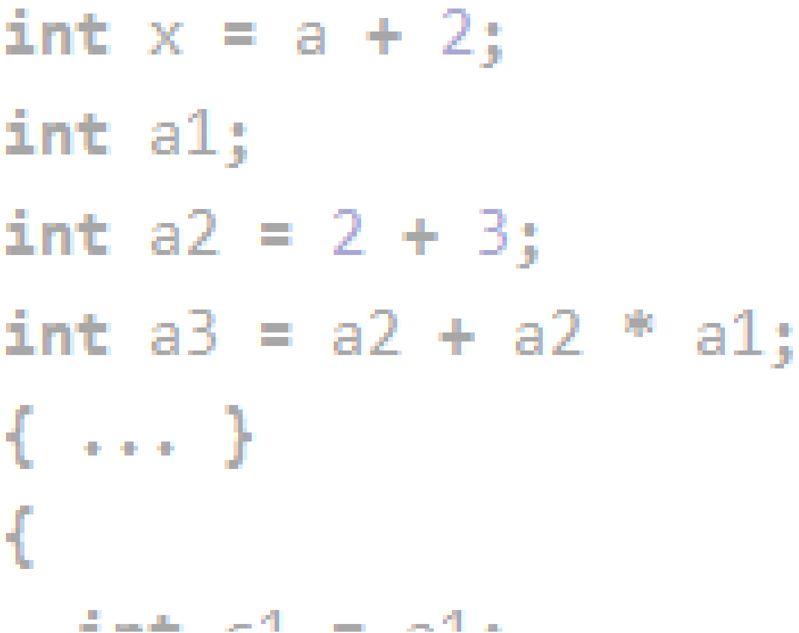
Reference:

Book Chapter 1.2,1.3

Md. Romizul Islam

romizul@cse.uiu.ac.bd

How do you store data in a Programming Language?



```
int x = a + 2;
int a1;
int a2 = 2 + 3;
int a3 = a2 + a2 * a1;
{ ... }
{
   int c1 = a1;
```

How do you store data in a Programming Language?

1. Variable
2. File
3. Database

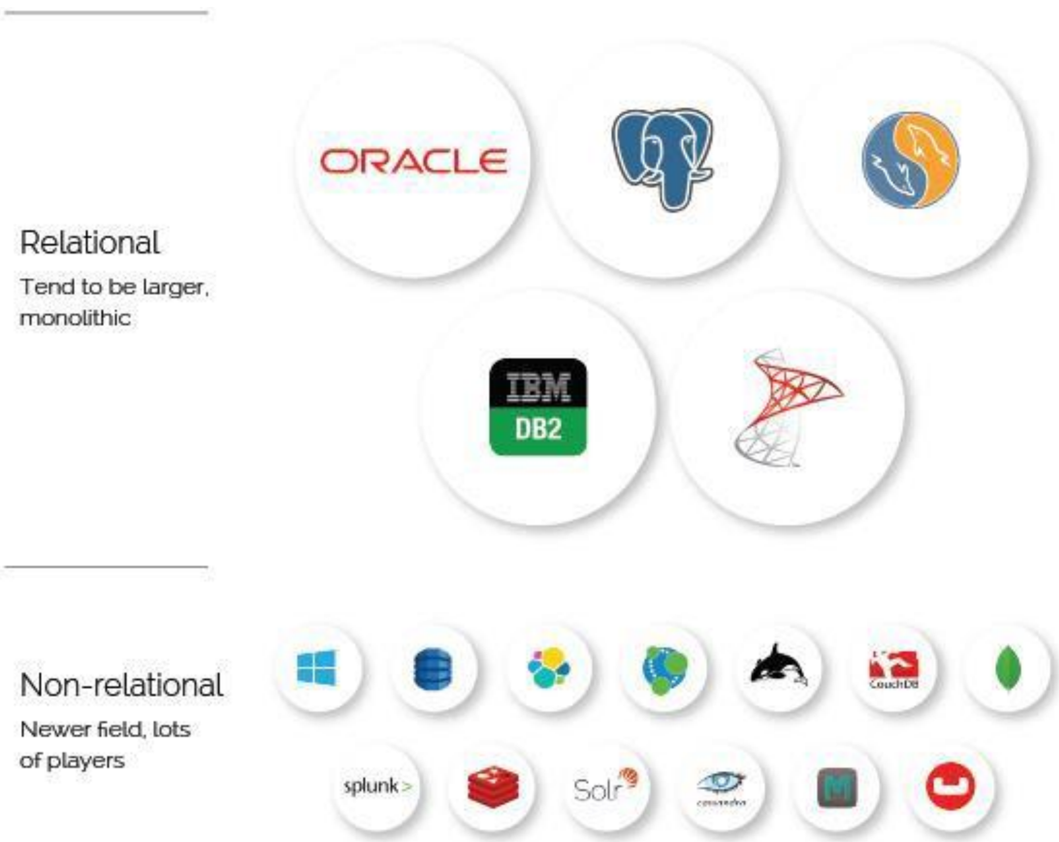
Why should we store in a Database?

A Piggy Bank

An Actual Bank

Database Management System (DBMS)

A software system that enables users to define, create, maintain and control access Information (stored in a database)



How a DBMS works

Hay DBMS, I need you to save a few students' **ID, name, and CT marks**. There will be 4 CTs.

DBMS

How a DBMS works

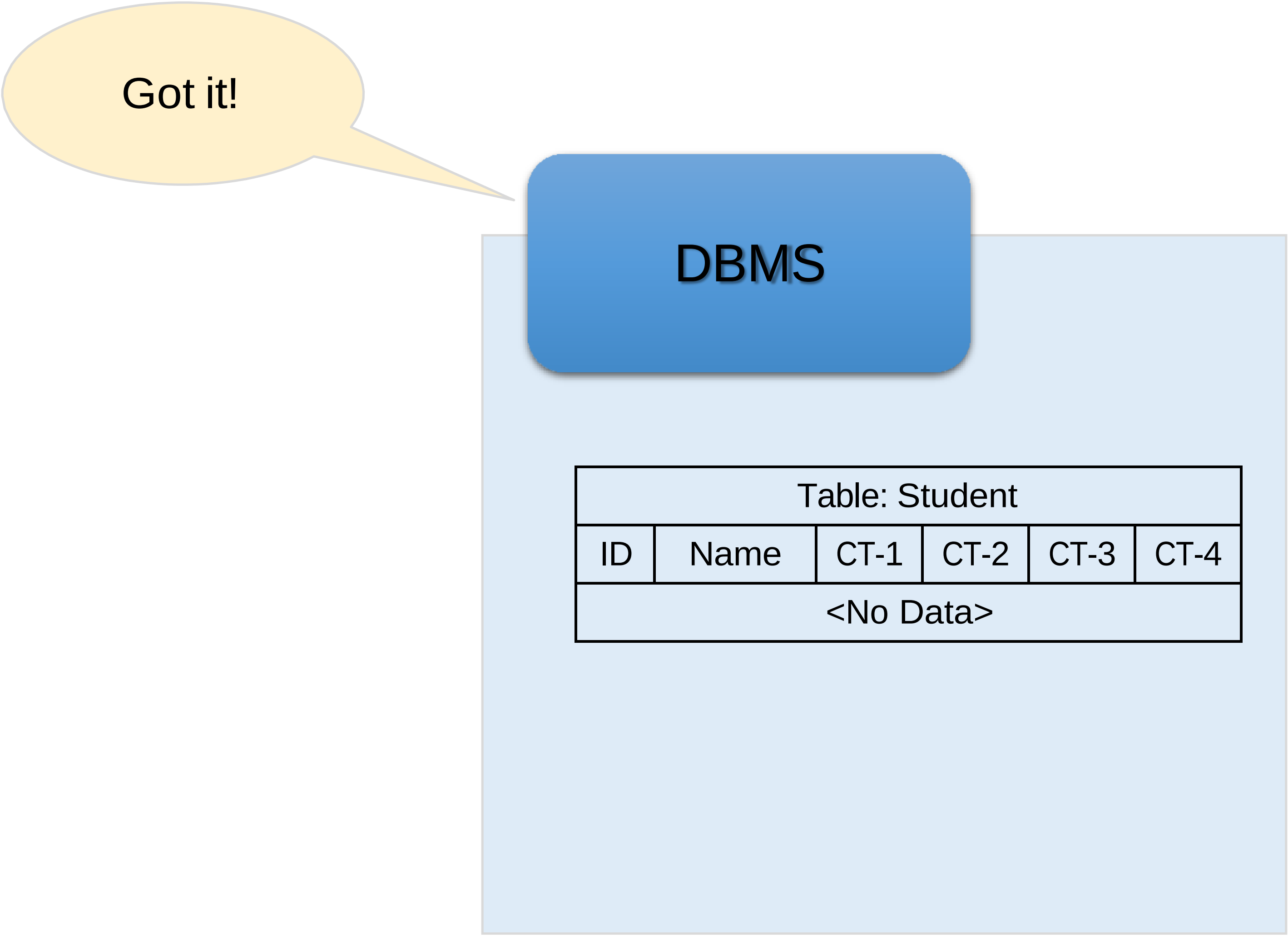


Table: Student					
ID	Name	CT-1	CT-2	CT-3	CT-4
<No Data>					

How a DBMS works

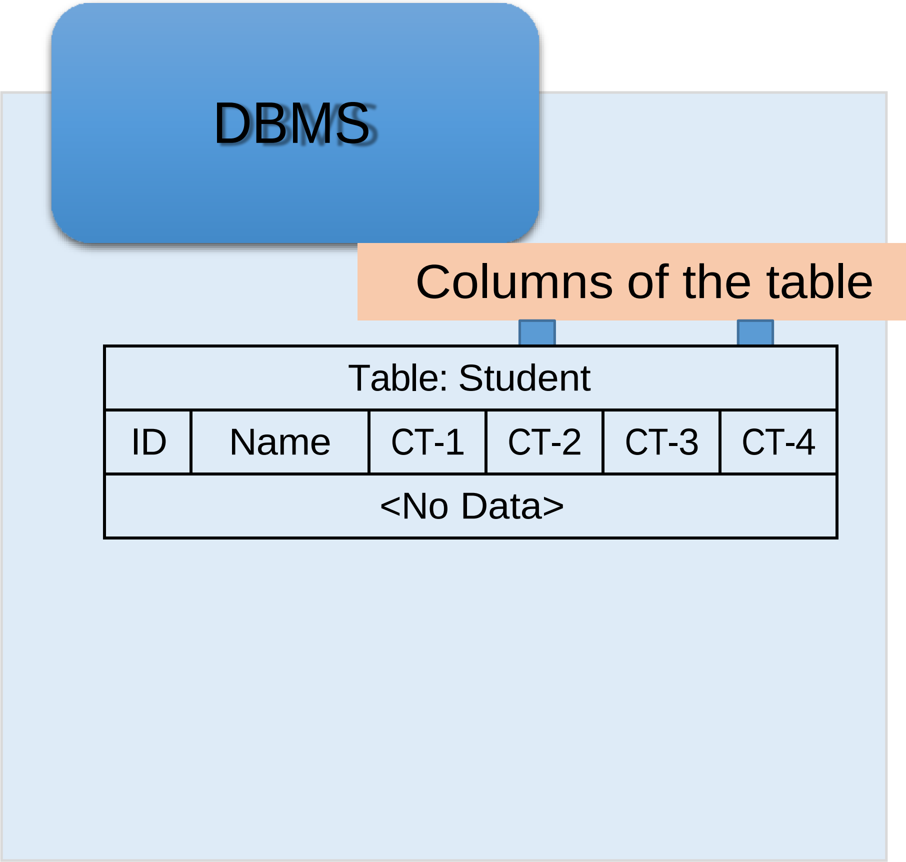


Table: Student

ID	Name	CT-1	CT-2	CT-3	CT-4
<No Data>					

How a DBMS works

Add some data to
it as follows:
(ID, Name, CT-1) =
(1, Mr. A, 13),
(2, Mr. B, 16)

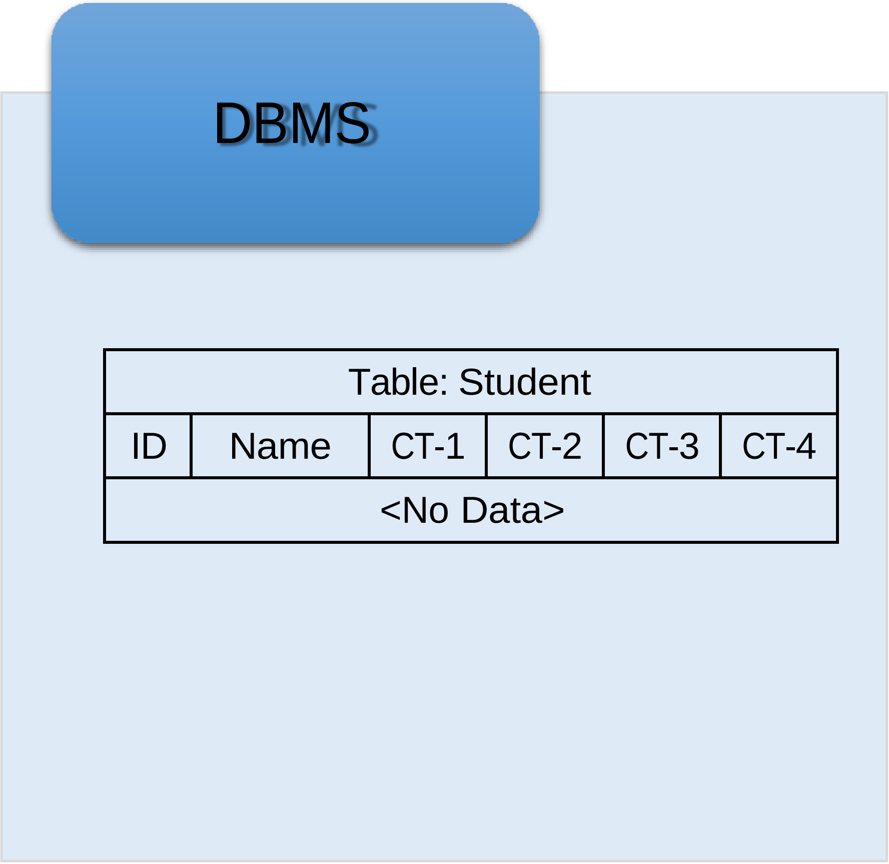
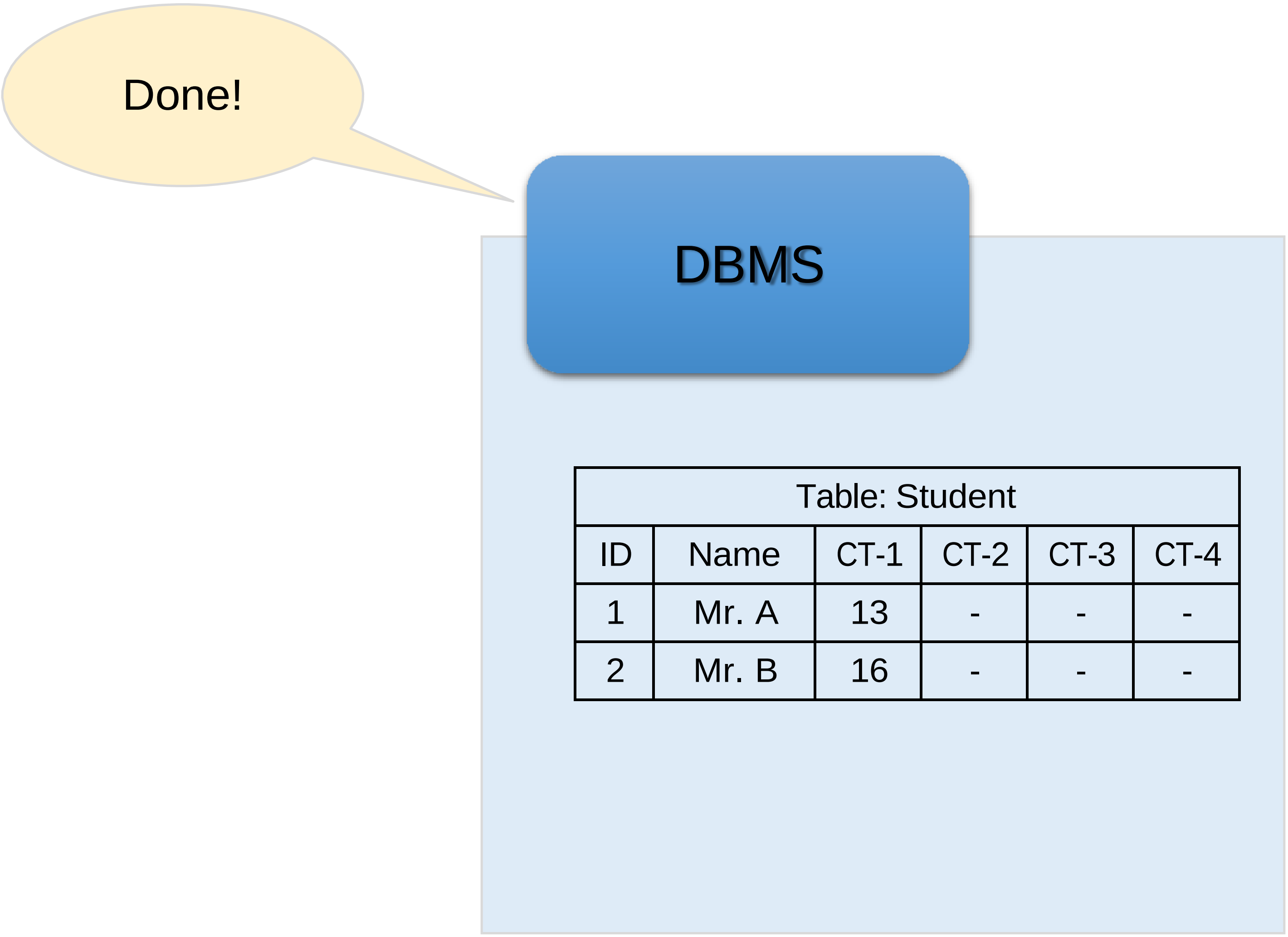


Table: Student					
ID	Name	CT-1	CT-2	CT-3	CT-4
<No Data>					

How a DBMS works



Done!

DBMS

Table: Student					
ID	Name	CT-1	CT-2	CT-3	CT-4
1	Mr. A	13	-	-	-
2	Mr. B	16	-	-	-

How a DBMS works

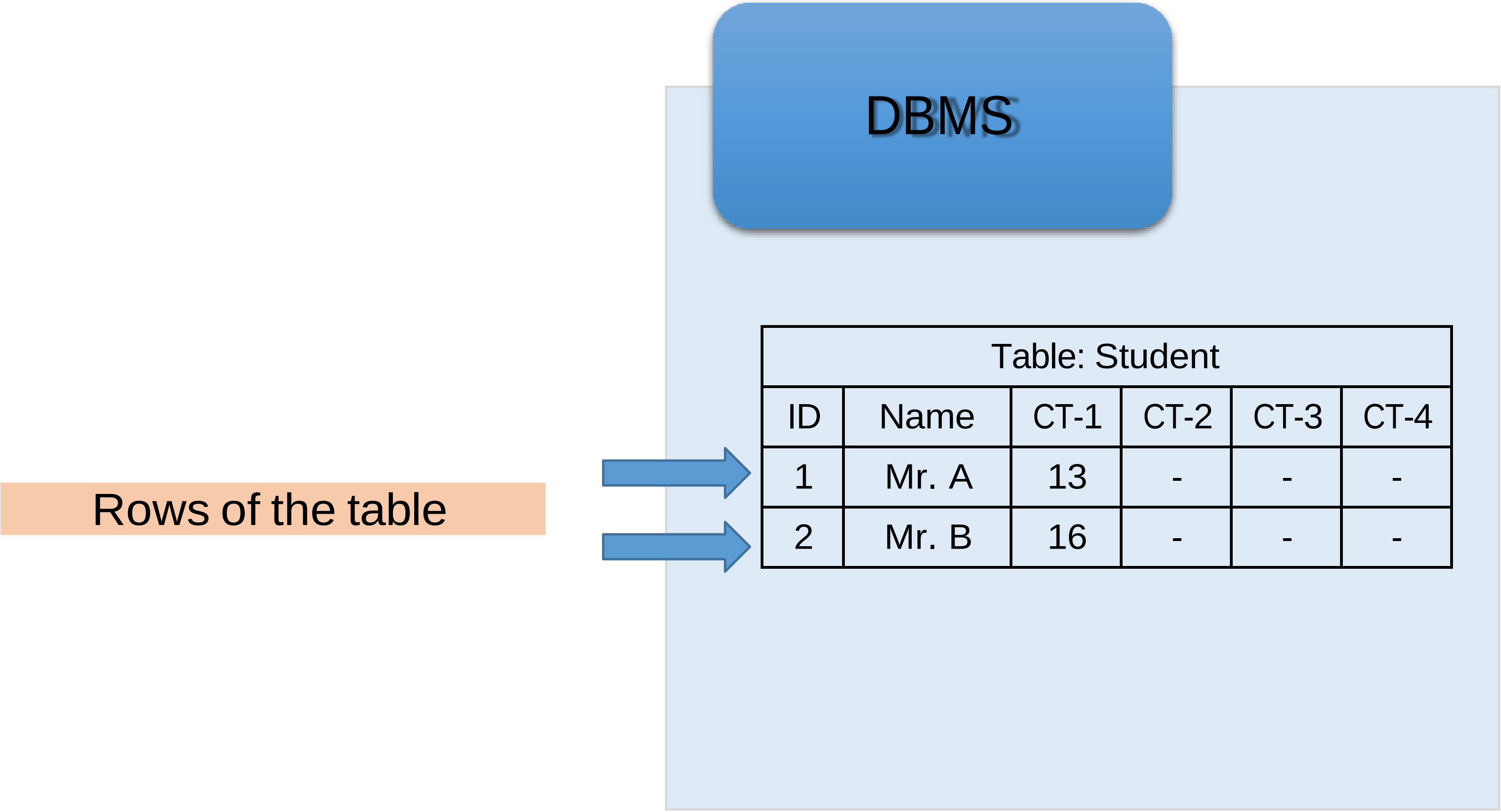
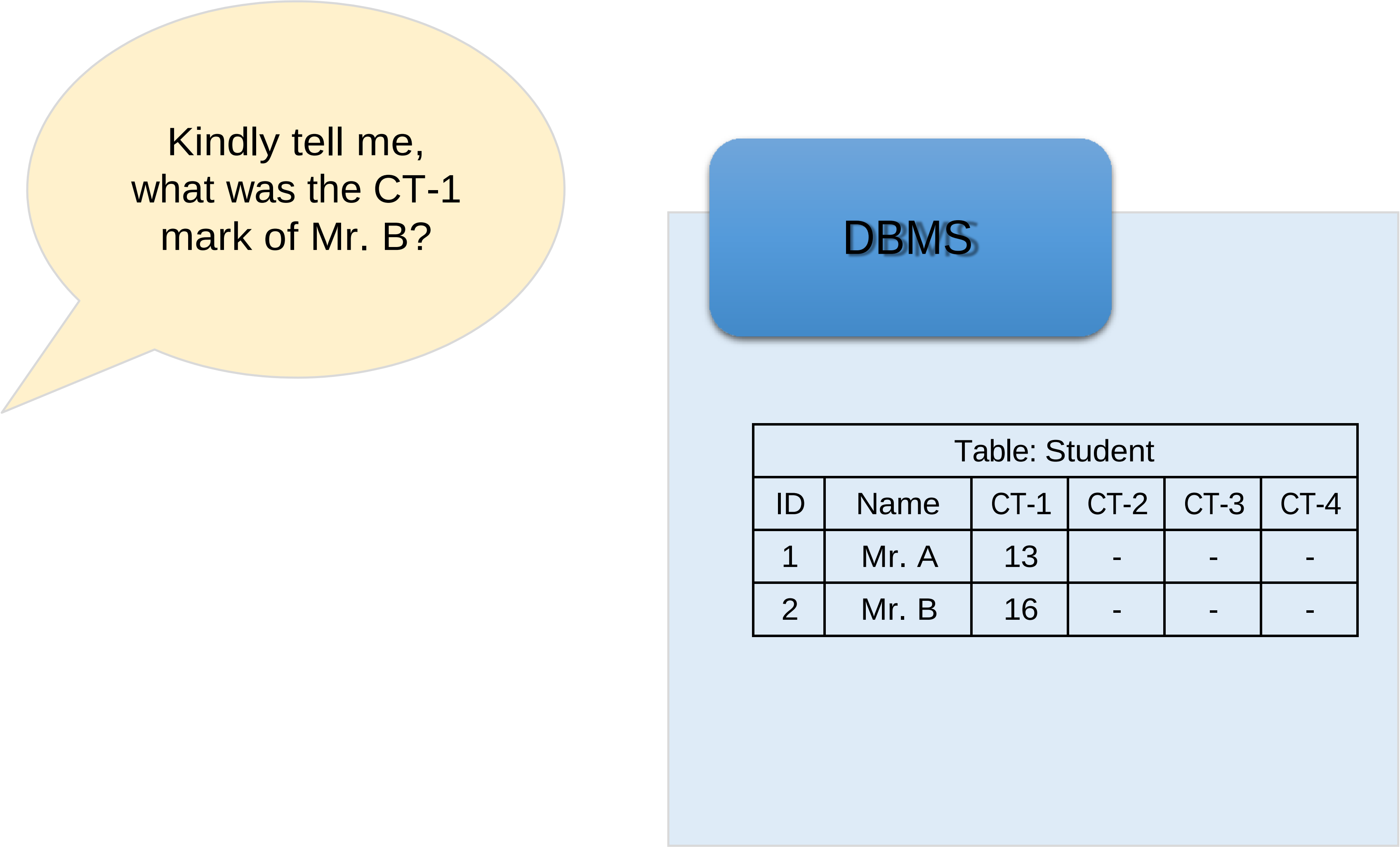


Table: Student

ID	Name	CT-1	CT-2	CT-3	CT-4
1	Mr. A	13	-	-	-
2	Mr. B	16	-	-	-

How a DBMS works



Kindly tell me, what was the CT-1 mark of Mr. B?

DBMS

Table: Student					
ID	Name	CT-1	CT-2	CT-3	CT-4
1	Mr. A	13	-	-	-
2	Mr. B	16	-	-	-

Structured Query Language

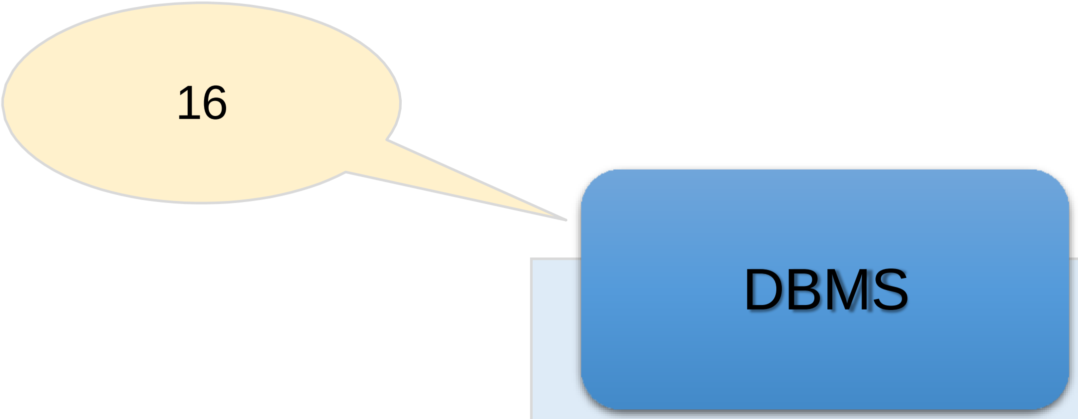


Table: Student					
ID	Name	CT-1	CT-2	CT-3	CT-4
1	Mr. A	13	-	-	-
2	Mr. B	16	-	-	-

Structured Query Language

- We can "**talk**" to a DBMS through a programming language called SQL (Structured Query Language)

- SQL is an *declarative* language (Unlike C/C++/Java which are *imperative* languages)

- In a declarative language, you tell the system what you want, and it understands how to do it.

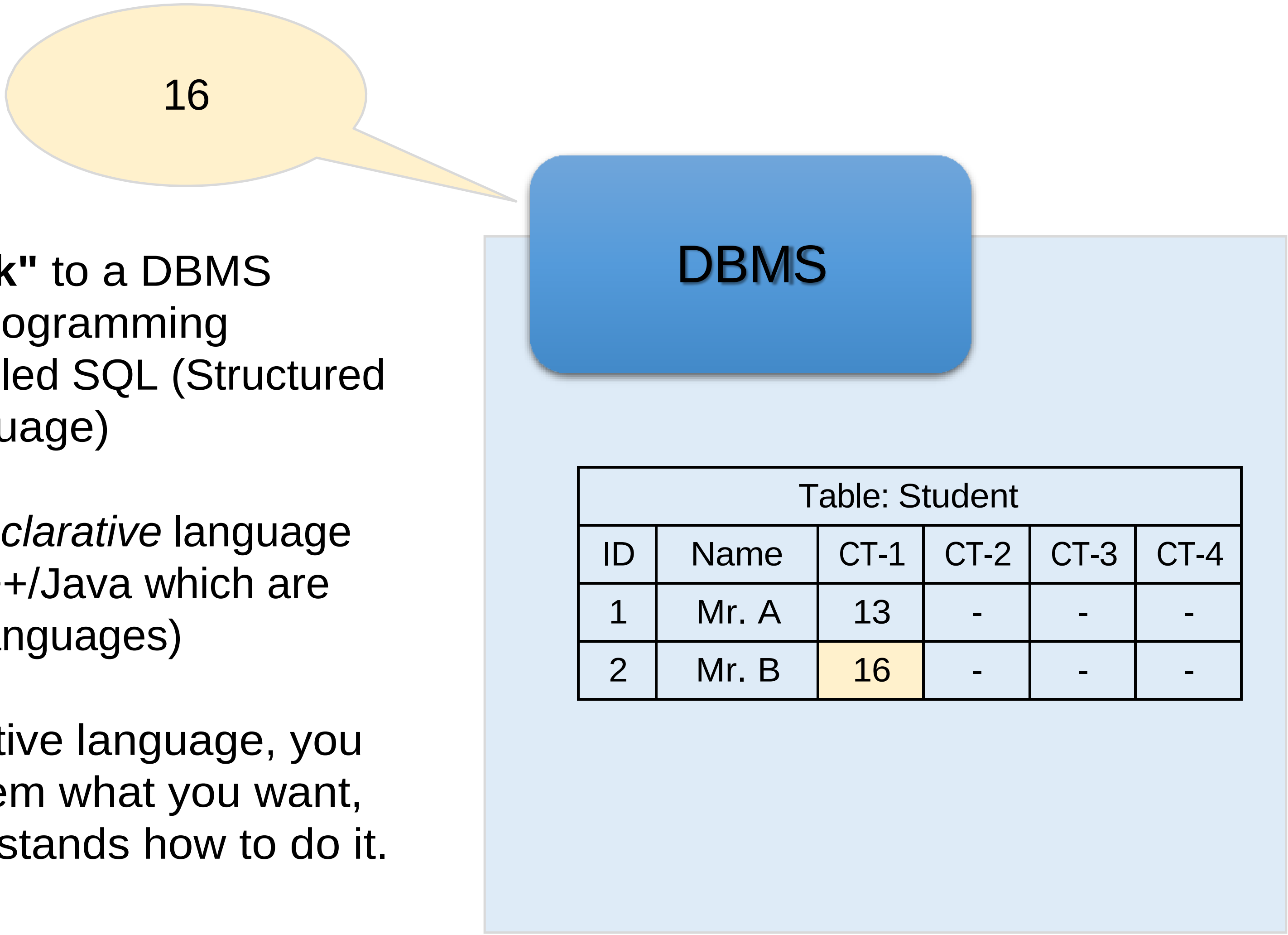
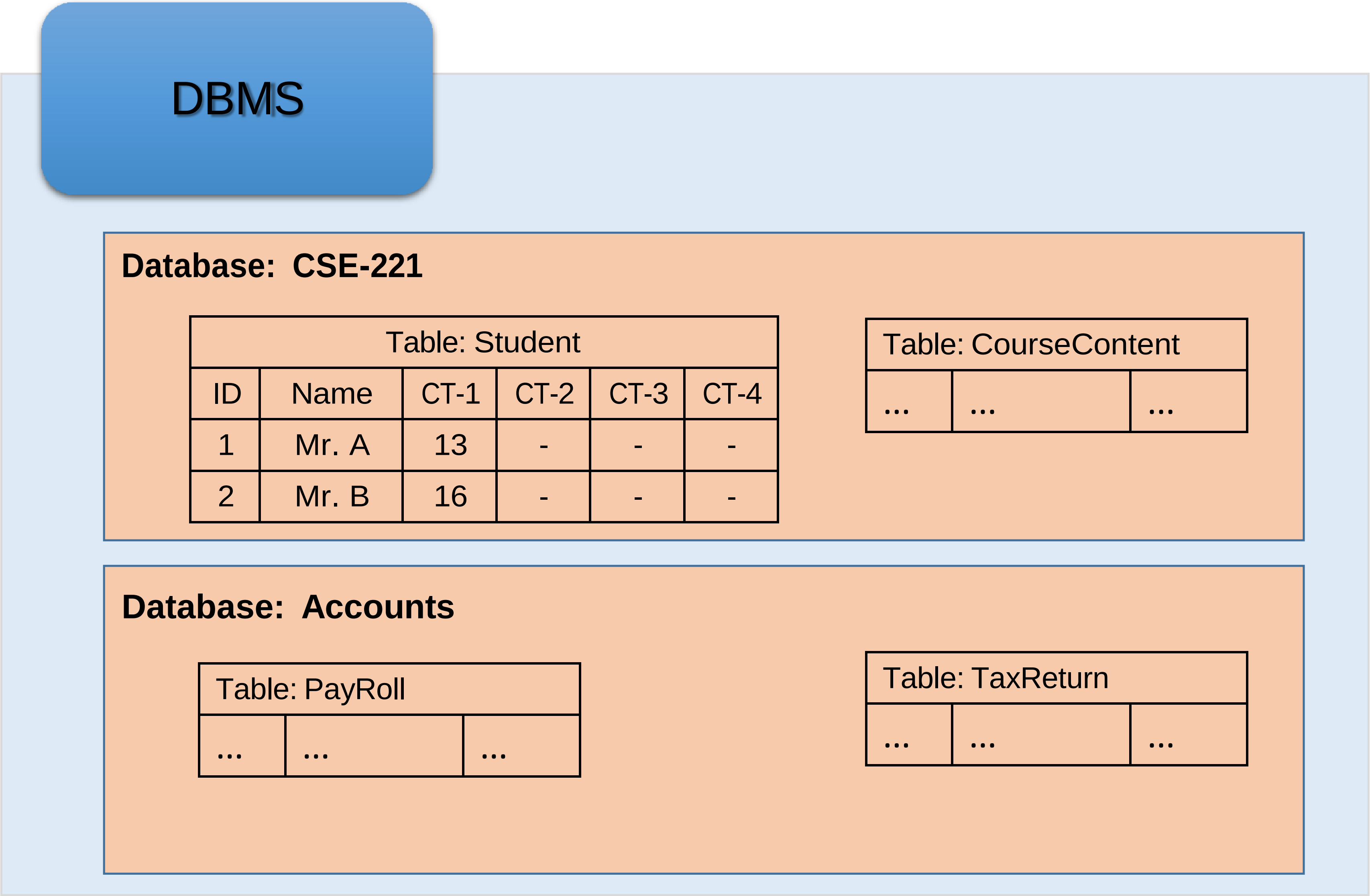


Table: Student

ID	Name	CT-1	CT-2	CT-3	CT-4
1	Mr. A	13	-	-	-
2	Mr. B	16	-	-	-

A DBMS can hold multiple Databases



DBMS

Database: CSE-221

Table: Student

ID	Name	CT-1	CT-2	CT-3	CT-4
1	Mr. A	13	-	-	-
2	Mr. B	16	-	-	-

Table: CourseContent

...	...	...

Database: Accounts

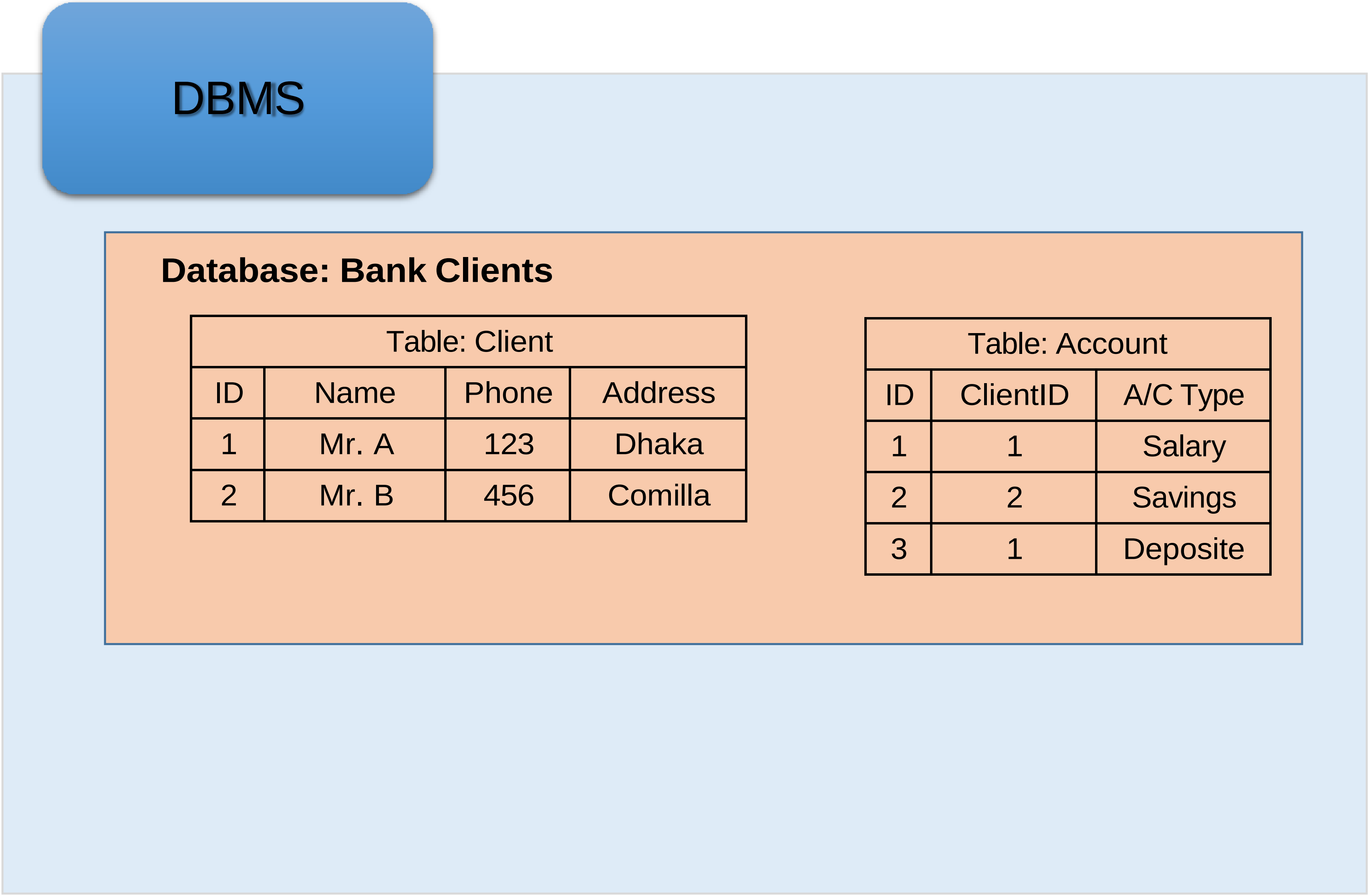
Table: PayRoll

...	...	...

Table: TaxReturn

...	...	...

Tables in a DB are usually related/connected



DBMS

Database: Bank Clients

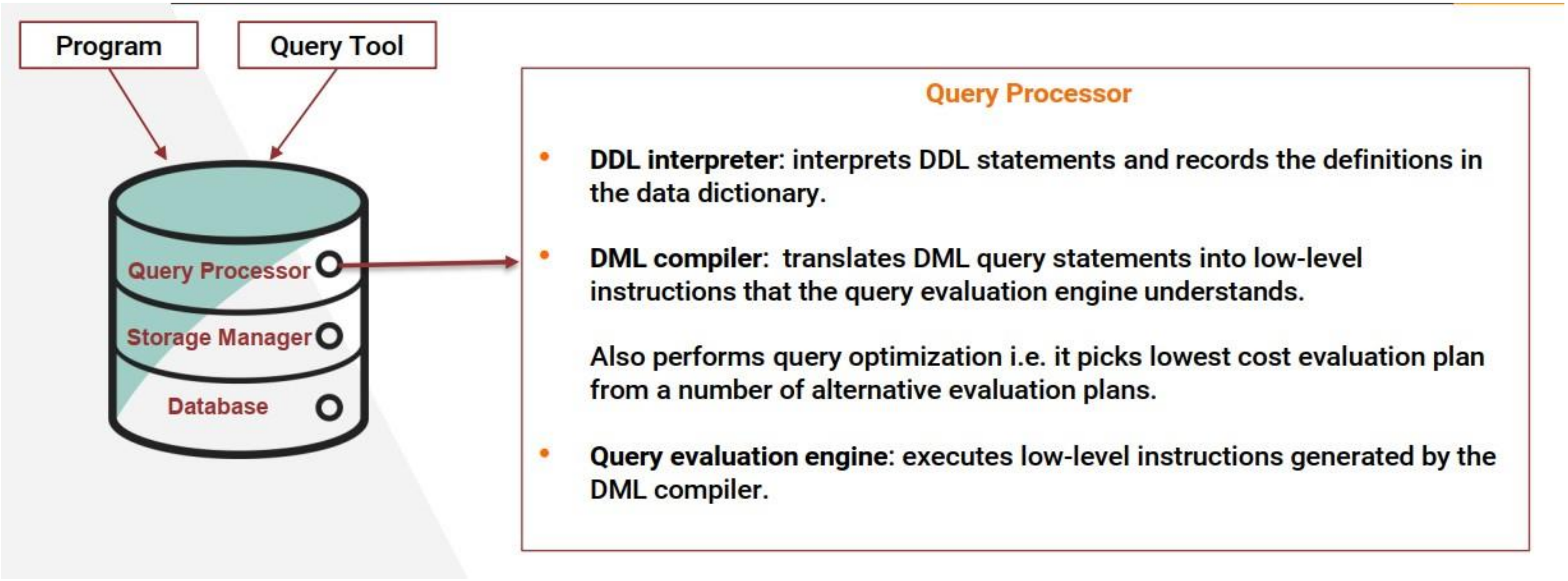
Table: Client

ID	Name	Phone	Address
1	Mr. A	123	Dhaka
2	Mr. B	456	Comilla

Table: Account

ID	ClientID	A/C Type
1	1	Salary
2	2	Savings
3	1	Deposite

Components of a DBMS

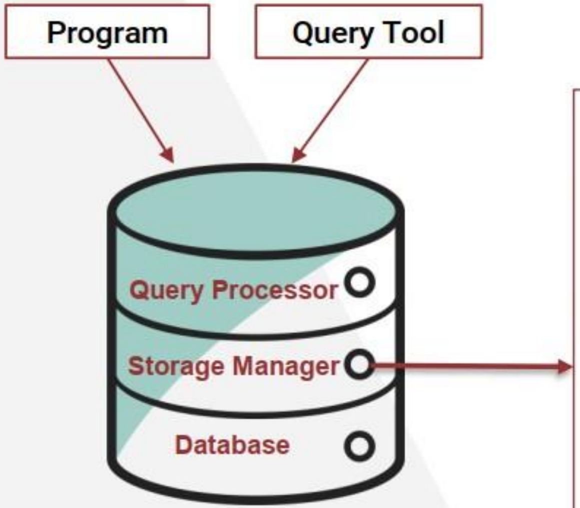


Query Processor

- **DDL interpreter**: interprets DDL statements and records the definitions in the data dictionary.
- **DML compiler**: translates DML query statements into low-level instructions that the query evaluation engine understands.

  Also performs query optimization i.e. it picks lowest cost evaluation plan from a number of alternative evaluation plans.
- **Query evaluation engine**: executes low-level instructions generated by the DML compiler.

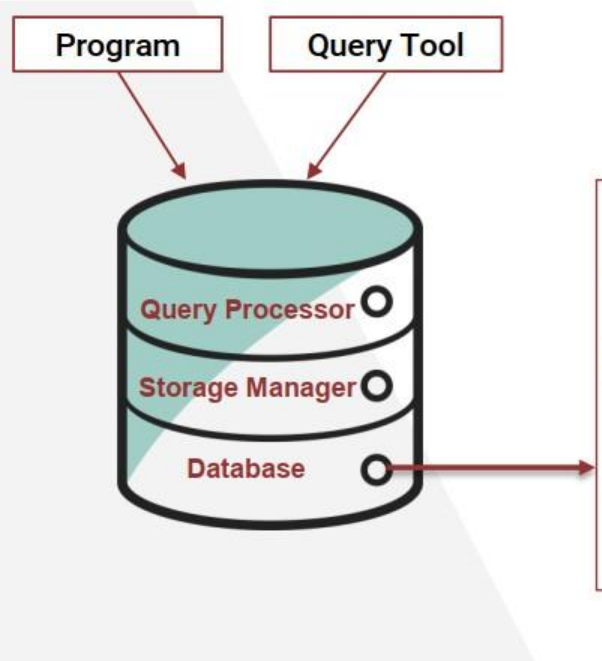
Components of a DBMS



Storage Manager

- **Authorization and integrity manager** tests for the satisfaction of integrity constraints and checks the authority of users to access data.
- **Transaction manager** ensures that the database remains in a consistent (correct) state despite system failures, and that concurrent transaction executions proceed without conflicting.
- **File manager** manages the allocation of space on disk storage and the data structures used to represent information stored on disk.
- **Buffer manager** is responsible for fetching data from disk storage into main memory, and deciding what data to cache in main memory.

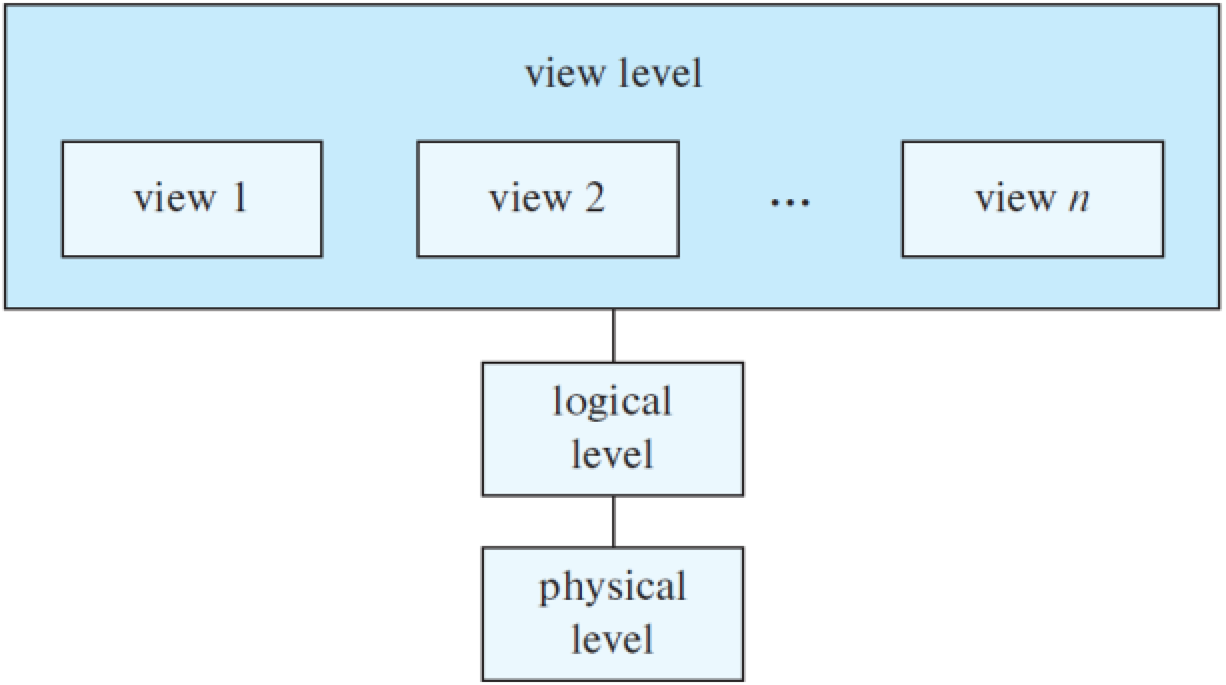
Components of a DBMS



Database

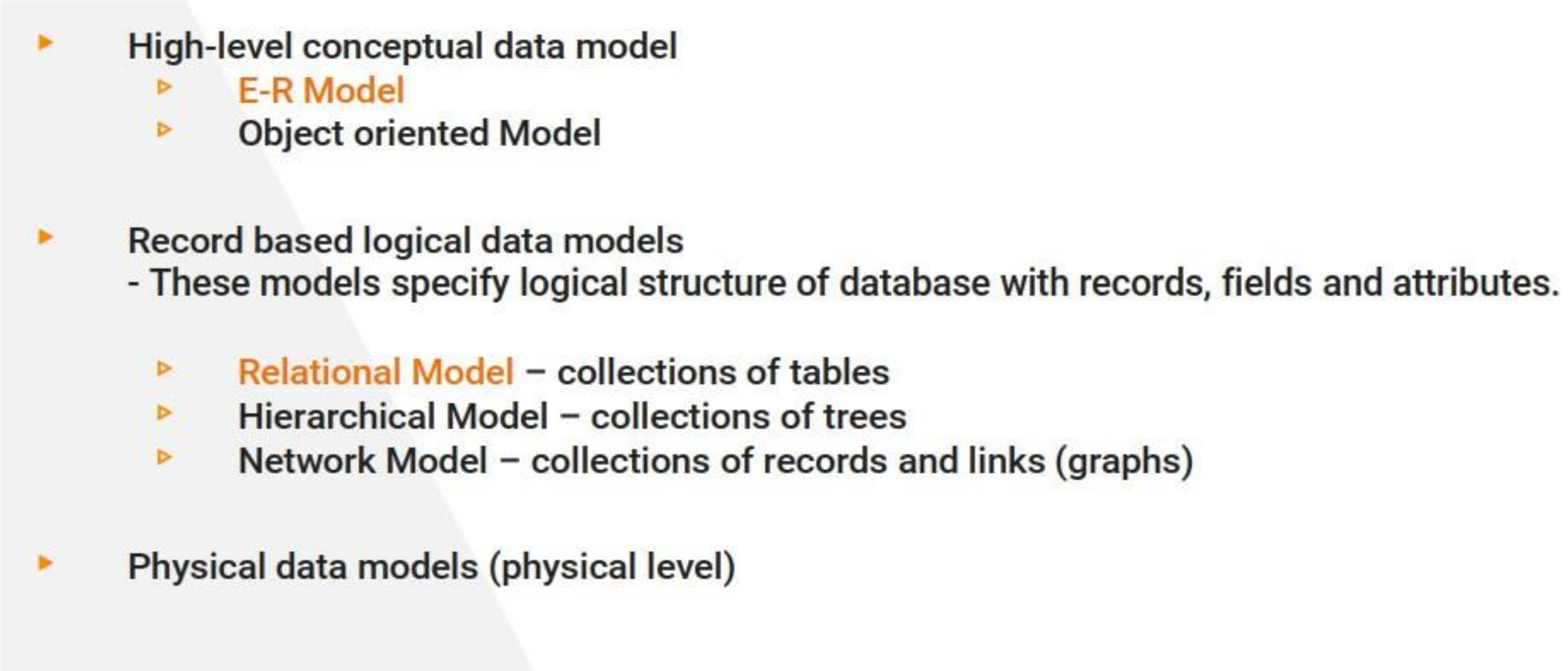
- **Data files**, which store the database itself.
- **Data dictionary**, which stores metadata about the structure of the database, in particular the schema of the database.
- **Indices**, which provide fast access to data items. A database index provides pointers to those data items that hold a particular value.

Data Abstraction



- A major purpose of a database system is to provide users with an abstract view of the data. That is, the system hides certain details of how the data are stored and maintained.
- To simplify users' interactions with the system, developers hide the complexity from users through several levels of data abstraction.

Data Models



- High-level conceptual data model
  - E-R Model
  - Object oriented Model

- Record based logical data models
  - These models specify logical structure of database with records, fields and attributes.

  - Relational Model – collections of tables
  - Hierarchical Model – collections of trees
  - Network Model – collections of records and links (graphs)

- Physical data models (physical level)

Benefits of DBMS over Plain File Storage

In your notebook, try to write five benefits of DBMS over a plain file storage.

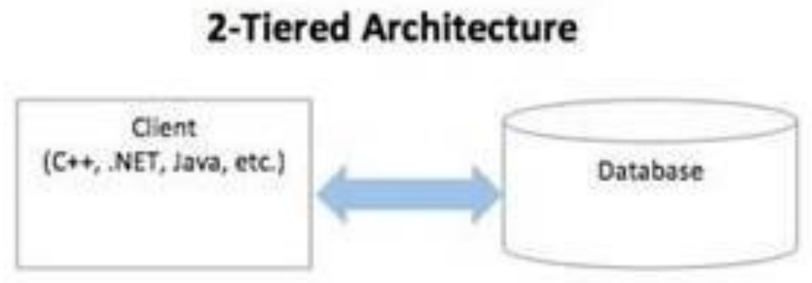
Benefits of DBMS over Plain File Storage

A typical DBMS System tries to solve the following issues-

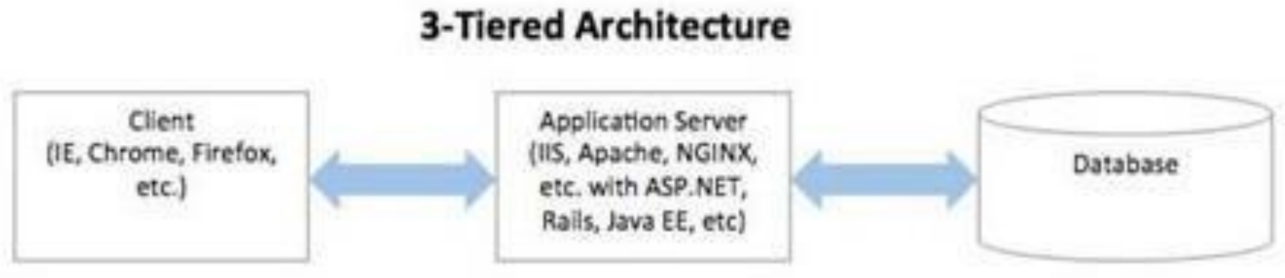
1. Data redundancy and inconsistency
2. Difficulty in accessing filtered data
3. Data isolation (Different data formats)
4. Integrity problems (Consistency constraints)
5. Atomicity problems
    - Concurrent-access anomalies
    - Security problems

Tier Architectures

The client in a Two-tier architecture application has the code written for saving data in the database. It handles both the Presentation layer (application interface) and Application layer (logical operations), while the server system handles the database layer.



2-Tiered Architecture



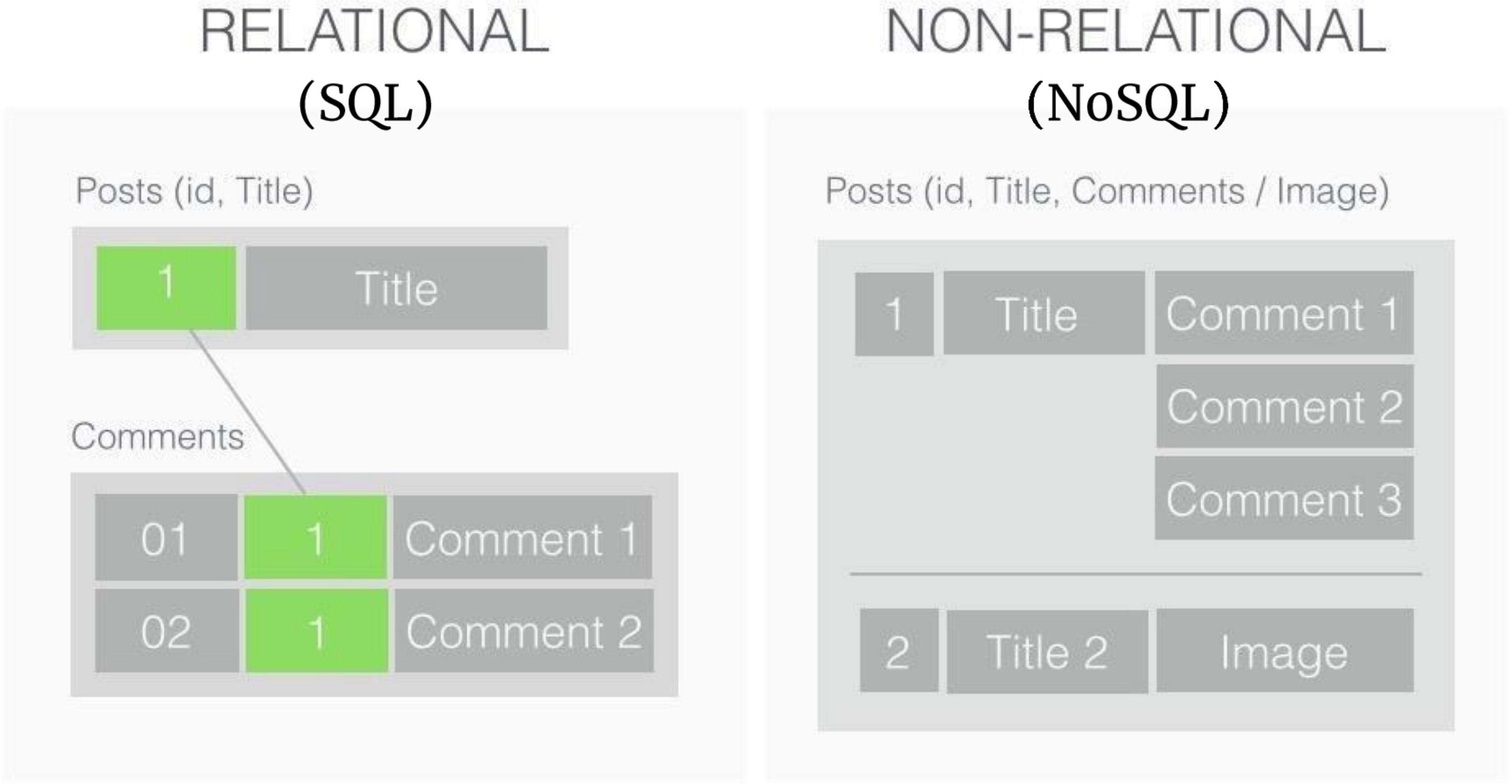
3-Tiered Architecture

Client
(IE, Chrome, Firefox, etc.)
Application Server
(IIS, Apache, NGINX, etc. with ASP.NET, Rails, Java EE, etc)
Database

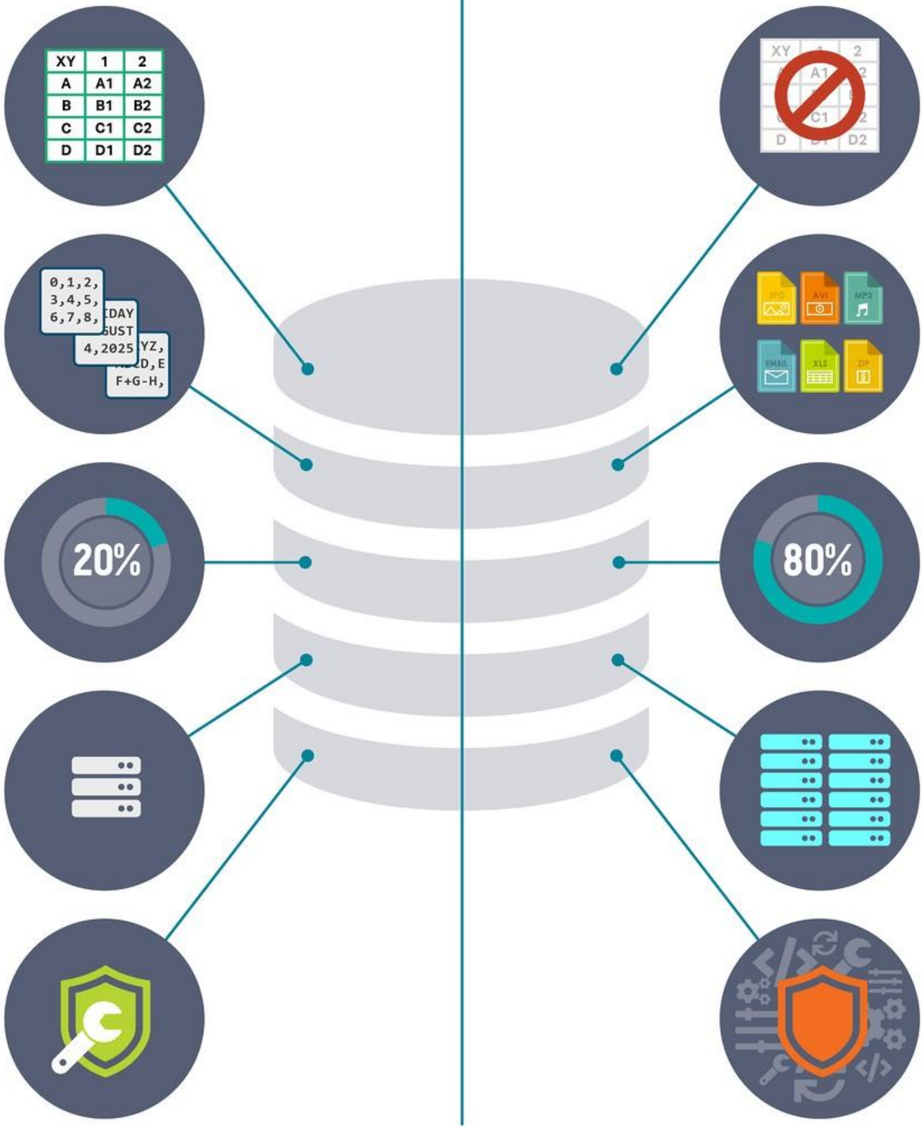
The Three-tier Architecture is divided into 3 parts:

1. Presentation Layer (Client tier)
2. Application Layer (Business tier)
3. Database Layer (Data tier)

Two Types of DBMS



Structured Data VS Unstructured Data



Structured Data

Can be displayed in rows, columns and relational databases

Numbers, dates and strings

Estimated 20% of enterprise data *(Gartner)*

Requires less storage

Easier to manage and protect with legacy solutions

Unstructured Data

Cannot be displayed in rows, columns and relational databases

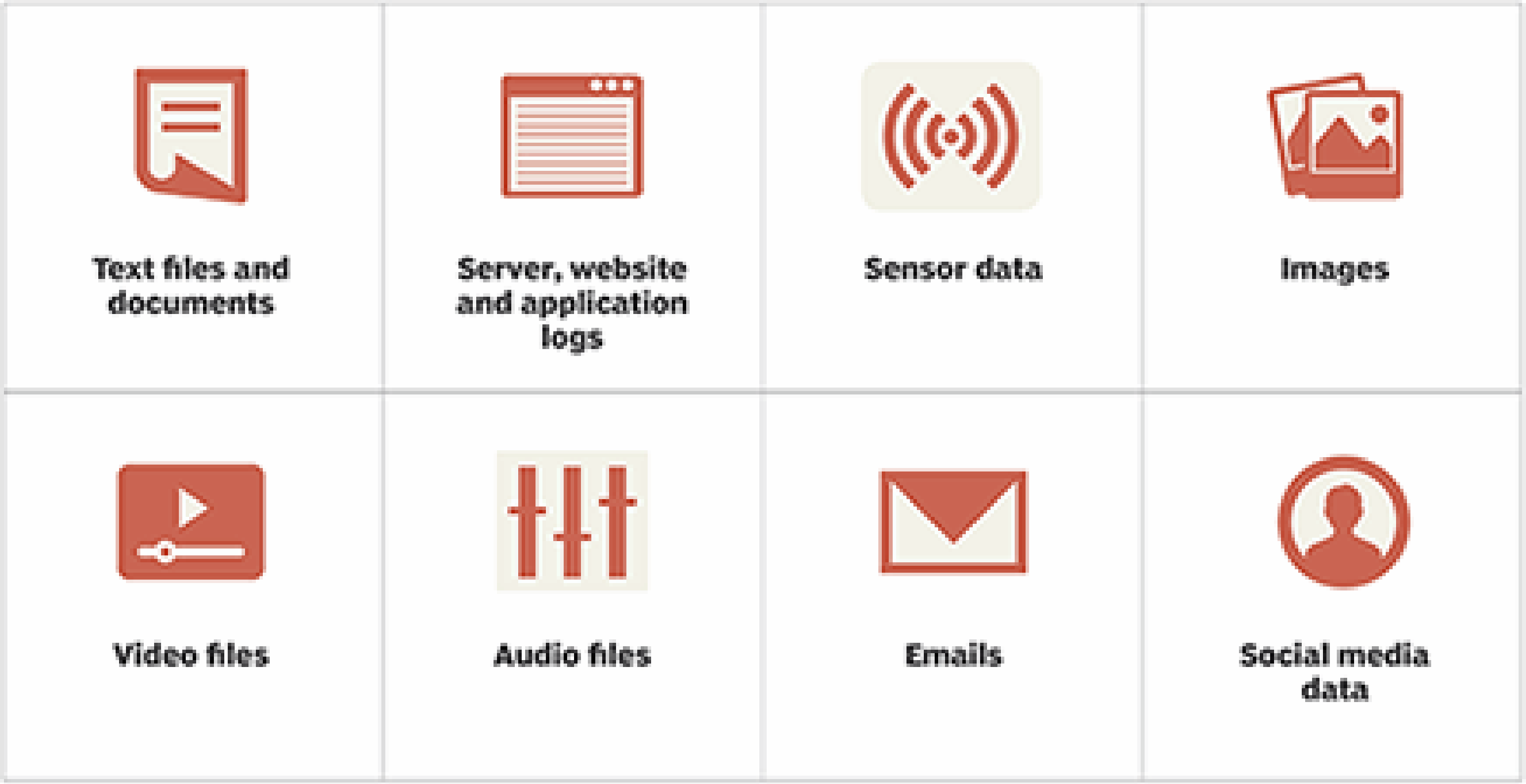
Images, audio, video, word processing files, e-mails, spreadsheets

Estimated 80% of enterprise data *(Gartner)*

Requires more storage

More difficult to manage and protect with legacy solutions

Unstructured data types



Text files and documents	Server, website and application logs	Sensor data	Images
Video files	Audio files	Emails	Social media data

Relational DBMS

Relational databases is that RDBMS schemas (tables) rigidly define how all data inserted into the database must be typed and composed.

Examples:

- **Oracle**: Oracle Database (commonly referred to as Oracle RDBMS or simply as Oracle) is a multi-model database management system produced and marketed by Oracle Corporation.
- **MySQL**: MySQL is an open-source relational database management system (RDBMS) based on Structured Query Language (SQL). MySQL runs on virtually all platforms, including Linux, UNIX, and Windows.
- **Microsoft SQL Server**: Microsoft SQL Server is an RDBMS that supports a wide variety of transaction processing, business intelligence, and analytics applications in corporate IT environments.
- **PostgreSQL**: PostgreSQL, often simply Postgres, is an object-relational database management system (ORDBMS) with an emphasis on extensibility and standards compliance.
- **DB2**: DB2 is an RDBMS designed to store, analyze, and retrieve data efficiently.

Relational DBMS

Advantage

- Relational databases are well-documented and mature technologies, and RDBMSs are sold and maintained by a number of established corporations.
- SQL standards are well-defined and commonly accepted.
- A large pool of qualified developers have experience with SQL and RDBMS.
- All RDBMS are ACID-compliant, meaning they satisfy the requirements of Atomicity, Consistency, Isolation, and Durability.

Disadvantage

- RDBMSs don't work well — or at all — with unstructured or semi-structured data due to schema and type constraints. This makes them ill-suited for large analytics or IoT event loads.
- The tables in your relational database will not necessarily map one-to-one with an object or class representing the same data.
- When migrating one RDBMS to another, schemas and types must generally be identical between source and destination tables for migration to work (schema constraint). For many of the same reasons, extremely complex datasets or those containing variable-length records are generally difficult to handle with an RDBMS schema.

NoSQL/Non-Relational DBMS

NoSQL databases can be schema-agnostic, allowing unstructured and semi-structured data to be stored and manipulated. Following are some common types of NoSQL DBMS.

- **Key-value stores**, such as Redis and Amazon DynamoDB, are extremely simple database management systems that store only key-value pairs and provide basic functionality for retrieving the value associated with a known key.
- **Wide column stores**, such as Cassandra, Scylla, and HBase, are schema-agnostic systems that enable users to store data in column families or tables, a single row of which can be thought of as a record — a multi-dimensional key-value store. These solutions are designed with the goal of scaling well enough to manage petabytes of data across as many as thousands of commodity servers in a massive, distributed system.
- **Document stores**, including MongoDB and Couchbase, are schema-free systems that store data in the form of JSON documents. Document stores are similar to key-value or wide column stores, but the document name is the key and the contents of the document, whatever they are, are the value.
- **Graph databases**, such as Neo4J and Datastax Enterprise Graph, represent data as a network of related nodes or objects in order to facilitate data visualizations and graph analytics.
- **Search engines**: Elasticsearch, Splunk, and Solr, store data using schema-free JSON documents.

NoSQL/Non-Relational DBMS

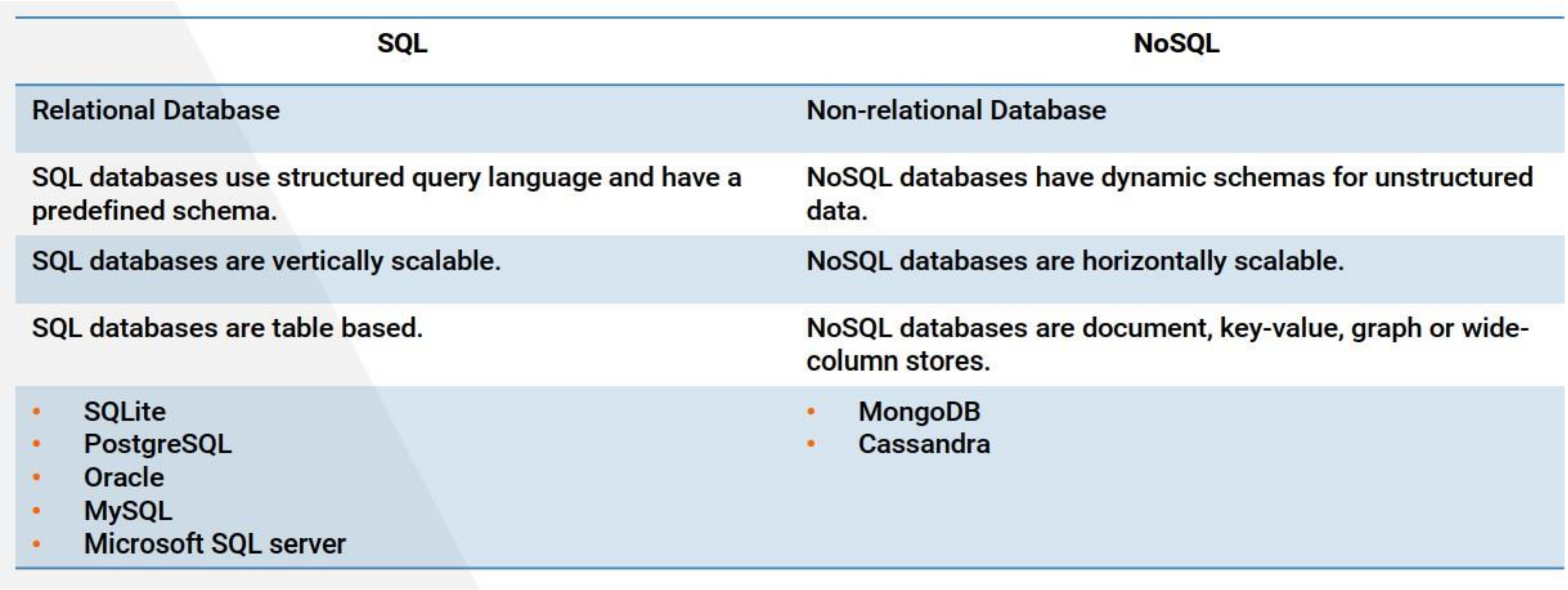
Advantage

- Schema-free data models are more flexible and easier to administer.
- NoSQL databases are generally more horizontally scalable and fault-tolerant.
- Data can easily be distributed across different nodes. To improve availability and/or partition tolerance, you can choose that data on some nodes be eventually consistent.

Disadvantage

- NoSQL databases are generally less widely adopted and mature than RDBMS solutions, so specific expertise is often required.
- There is a range of formats and constraints specific to each database type.

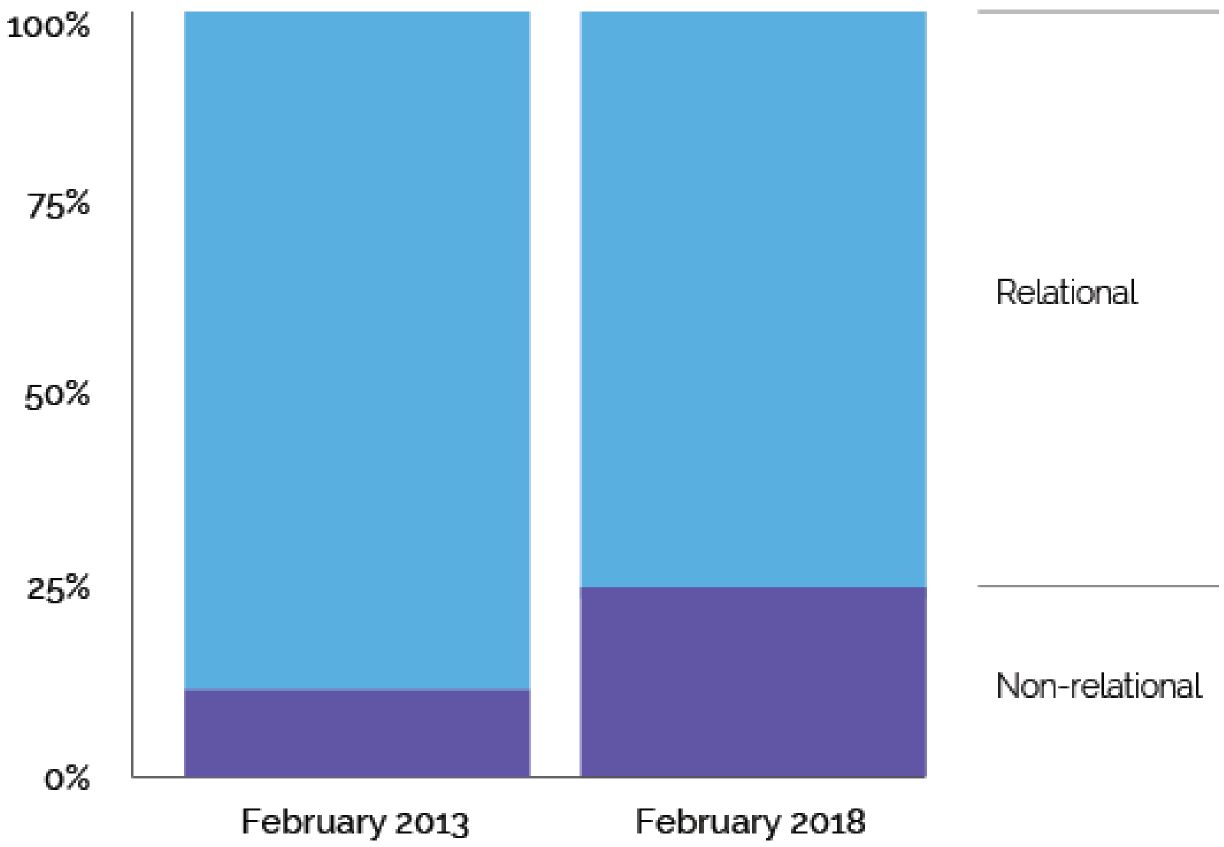
SQL vs NoSQL - Summary



SQL	NoSQL
Relational Database	Non-relational Database
SQL databases use structured query language and have a predefined schema.	NoSQL databases have dynamic schemas for unstructured data.
SQL databases are vertically scalable.	NoSQL databases are horizontally scalable.
SQL databases are table based.	NoSQL databases are document, key-value, graph or wide-column stores.
• SQLite • PostgreSQL • Oracle • MySQL • Microsoft SQL server	• MongoDB • Cassandra

Two Types of DBMS

Popularity (percentage) Relational Databases vs. Non-Relational Databases



Source: https://db-engines.com/en/ranking_trend

Task

Mention two examples of-

- Two-tire Architecture
- Three-tire Architecture
- Unstructured Data

Reference

1. https://www.slideshare.net/imamhossain75054/dbms-1-introduction-to-dbms-250584928
2. https://medium.com/@gacheruevans0/2-tier-vs-3-tier-architecture-26db56fe7e9c
3. https://dzone.com/articles/the-types-of-modern-databases
4. https://lawtomated.com/structured-data-vs-unstructured-data-what-are-they-and-why-care/