

Data modeling and SQL for Data Science

By Md. Meftaul Haque Mishu

Agenda

- What is Data Modeling?
- Types of Data Models
- Entity-Relationship Model
- SQL for Data Science
 - SQL Basics
 - SQL Joins
 - SQL Aggregations
 - SQL Subqueries
- Practice

What is Data Modeling?

An abstract model that **organizes elements of data** and **standardizes how they relate to one another** and to the properties of real-world entities.

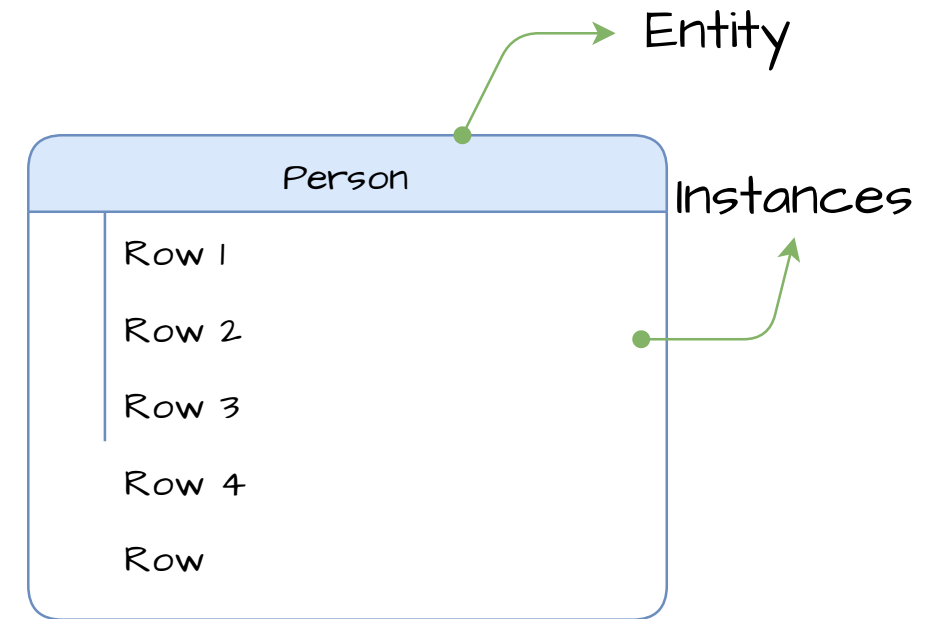
Components of Data Modeling

- **Entity:** A real-world object or concept.
- **Attribute:** A property or characteristic of an entity.
- **Relationship:** Describes how entities are related to one another.

Entity

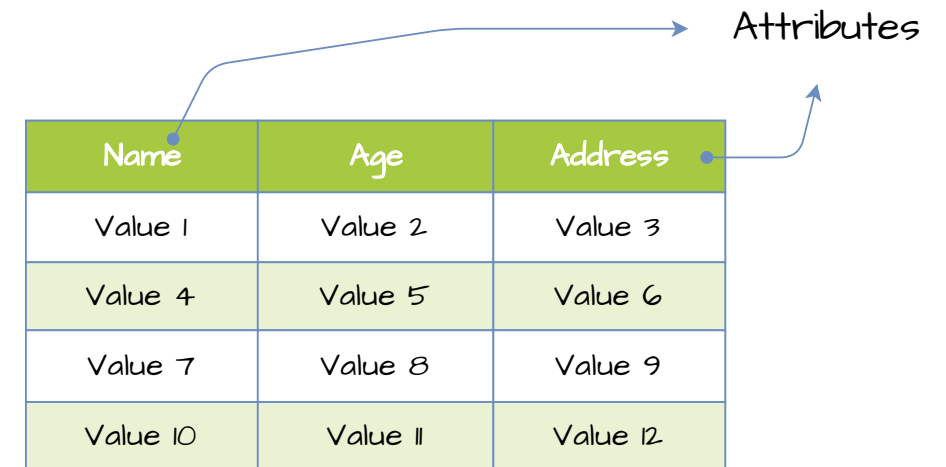
In data modeling, an entity is a **real-world object or concept** that can be uniquely identified and described, such **as a person, place, thing, event**, or concept.

An entity is typically represented by a table in a relational database, and each instance of the entity is represented by a row in the table.



Attribute

An attribute is a **characteristic or property of an entity**, such as a name, age or address. Attributes are represented by columns in the table that represents the entity.



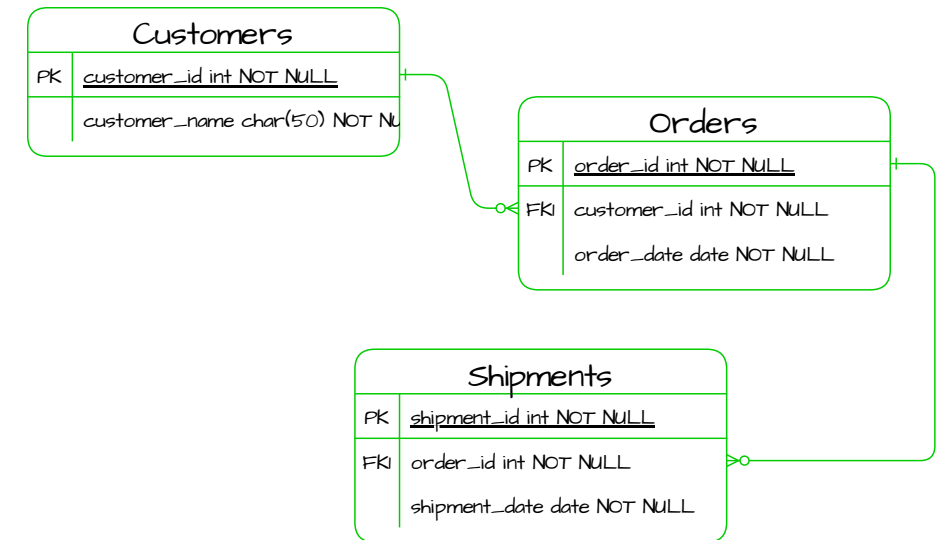
The diagram illustrates the concept of attributes using a table. The table has three columns: Name, Age, and Address. The first row contains 'Value 1', 'Value 2', and 'Value 3'. The second row contains 'Value 4', 'Value 5', and 'Value 6'. The third row contains 'Value 7', 'Value 8', and 'Value 9'. The fourth row contains 'Value 10', 'Value 11', and 'Value 12'. Arrows point from the column headers 'Name', 'Age', and 'Address' to the word 'Attributes' on the right, indicating that these columns represent attributes of an entity.

Name	Age	Address
Value 1	Value 2	Value 3
Value 4	Value 5	Value 6
Value 7	Value 8	Value 9
Value 10	Value 11	Value 12

Relationship

A relationship is a **connection or association between two or more entities**.

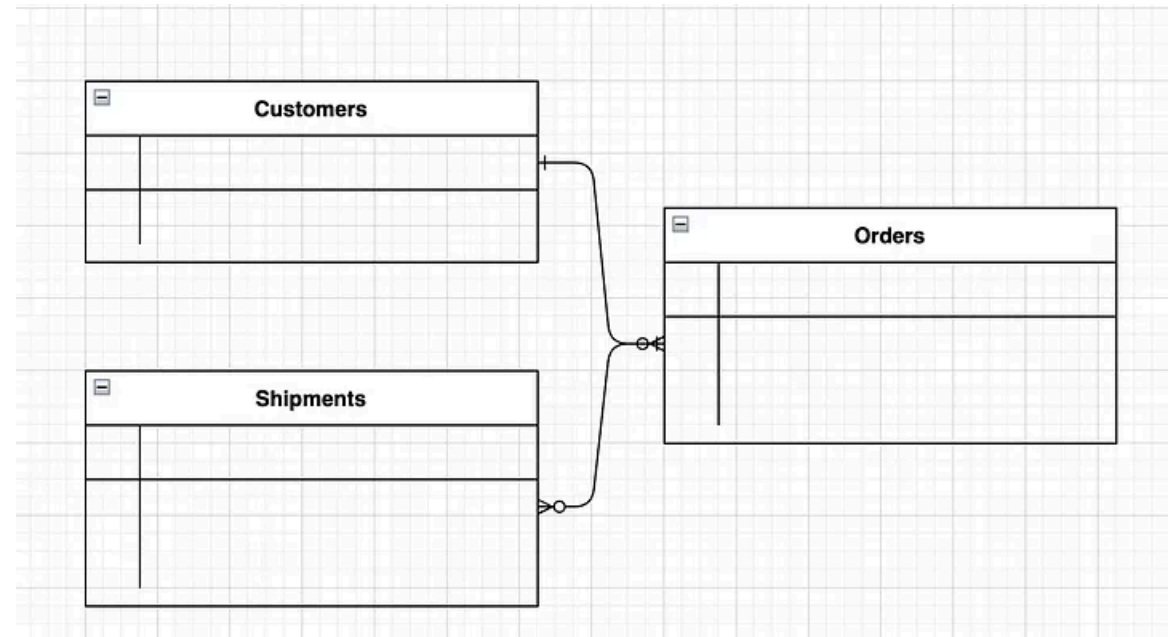
Relationships describe how entities interact or relate to each other, and are represented by lines or connectors between the entities in a data model. Relationships can be one-to-one, one-to-many, or many-to-many.



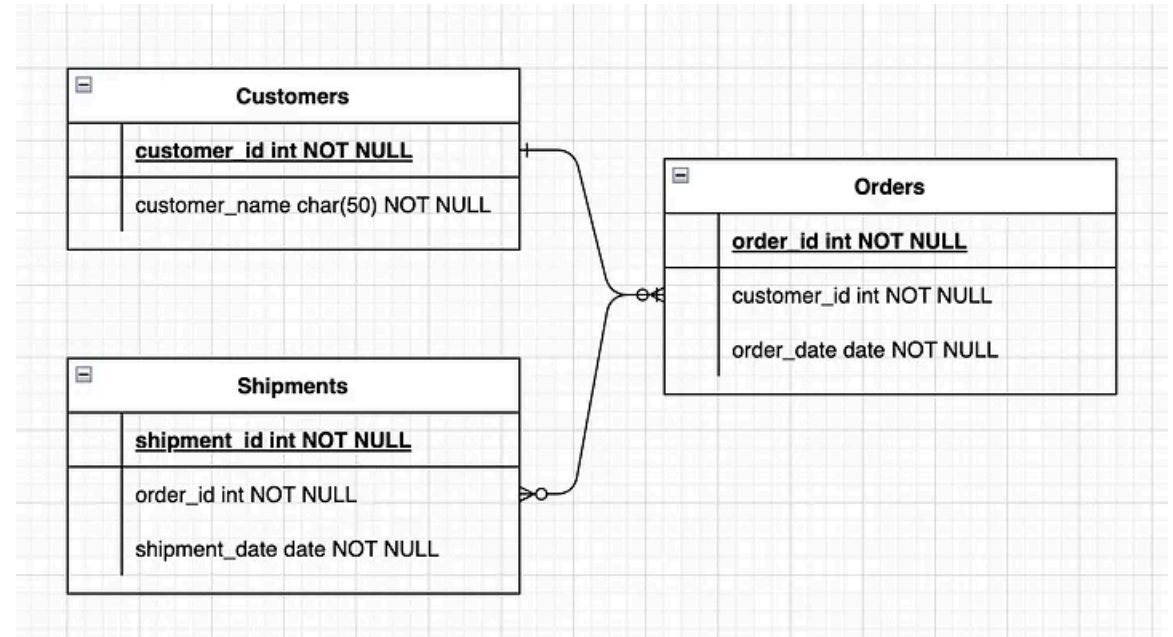
Three Levels of Data Modeling

1. Conceptual Data Model
2. Logical Data Model
3. Physical Data Model

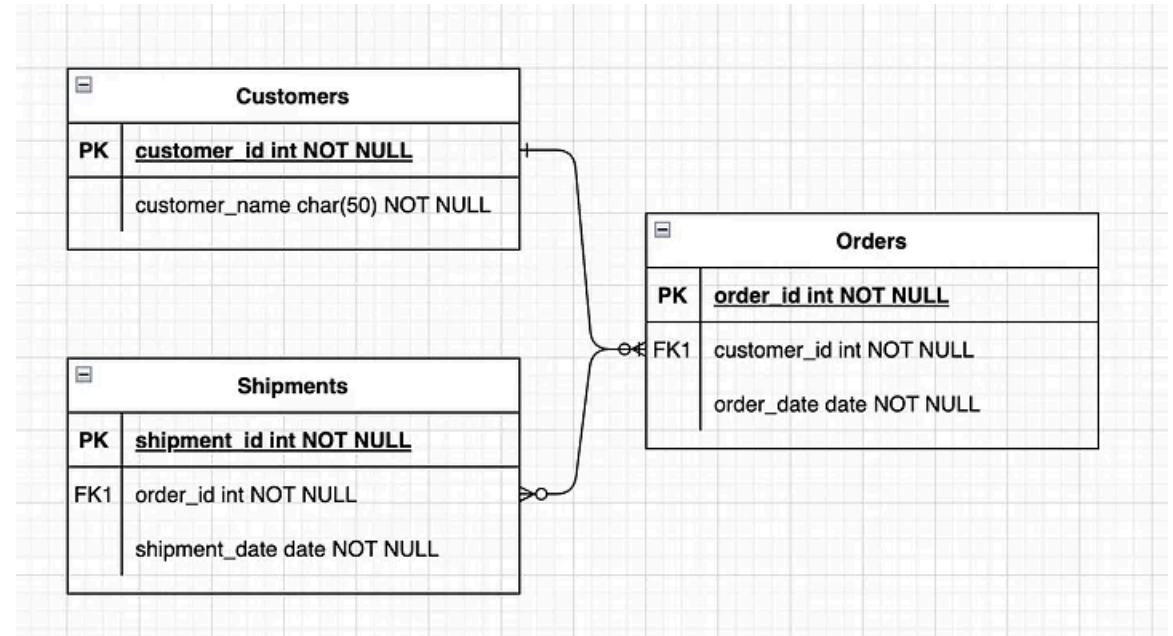
1. Conceptual Data Model: High-level, static business structures and concepts.



2. **Logical Data Model:** Entity types, data attributes, and relationships between them.



3. **Physical Data Model:** Represents the actual design of a database or how will it be stored in a file system.

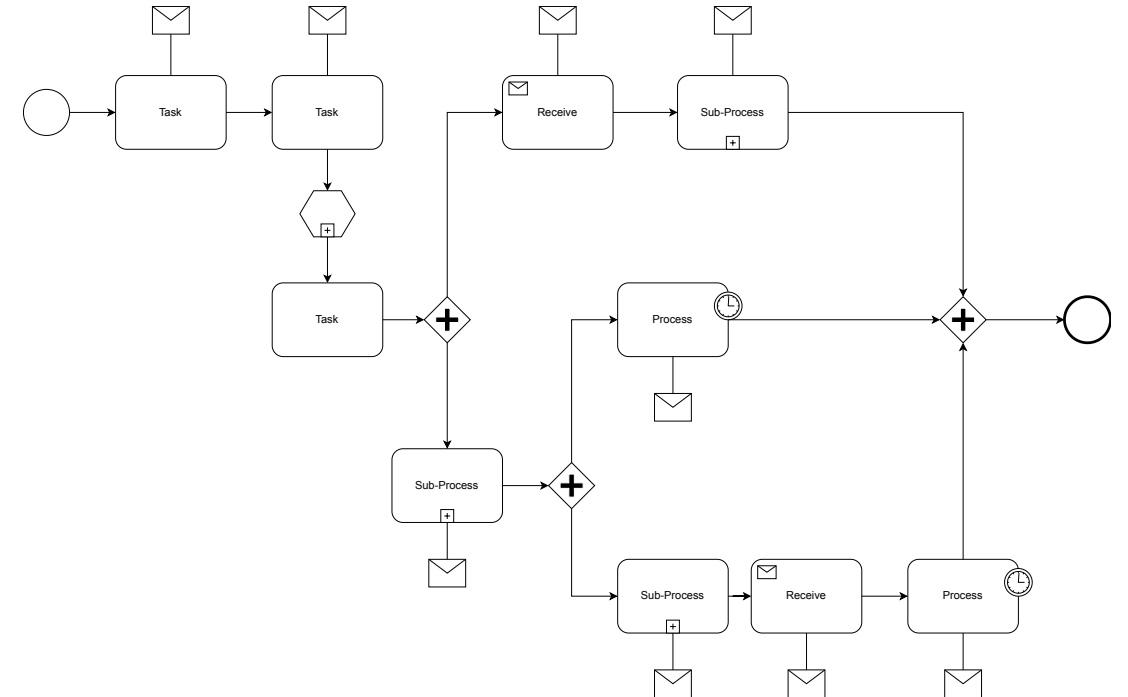


Data Modeling Notations

- Entity-Relationship Diagram (ERD)
- Unified Modeling Language (UML)
- Data Flow Diagram (DFD)
- Business Process Model and Notation (BPMN)

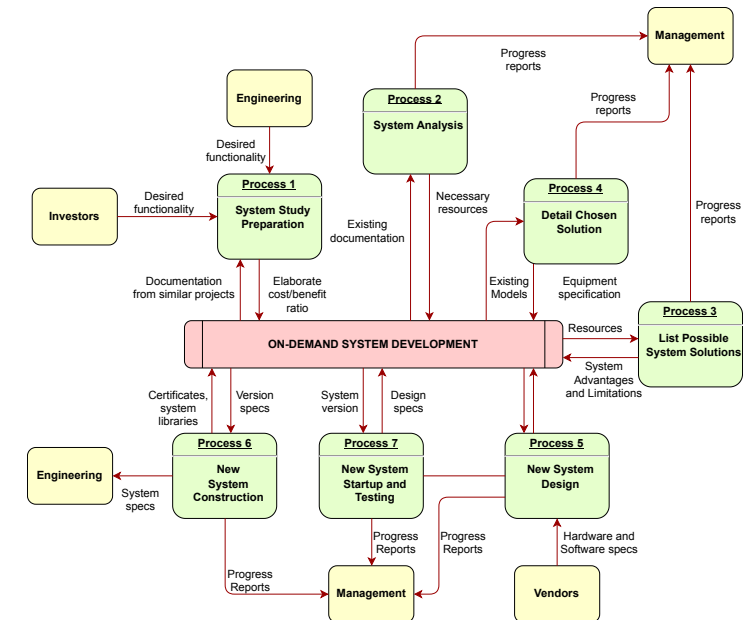
BPMN

BPMN is a notation used to model business processes. BPMN diagrams consist of activities, gateways, events, and flows, and are commonly used in business process management.



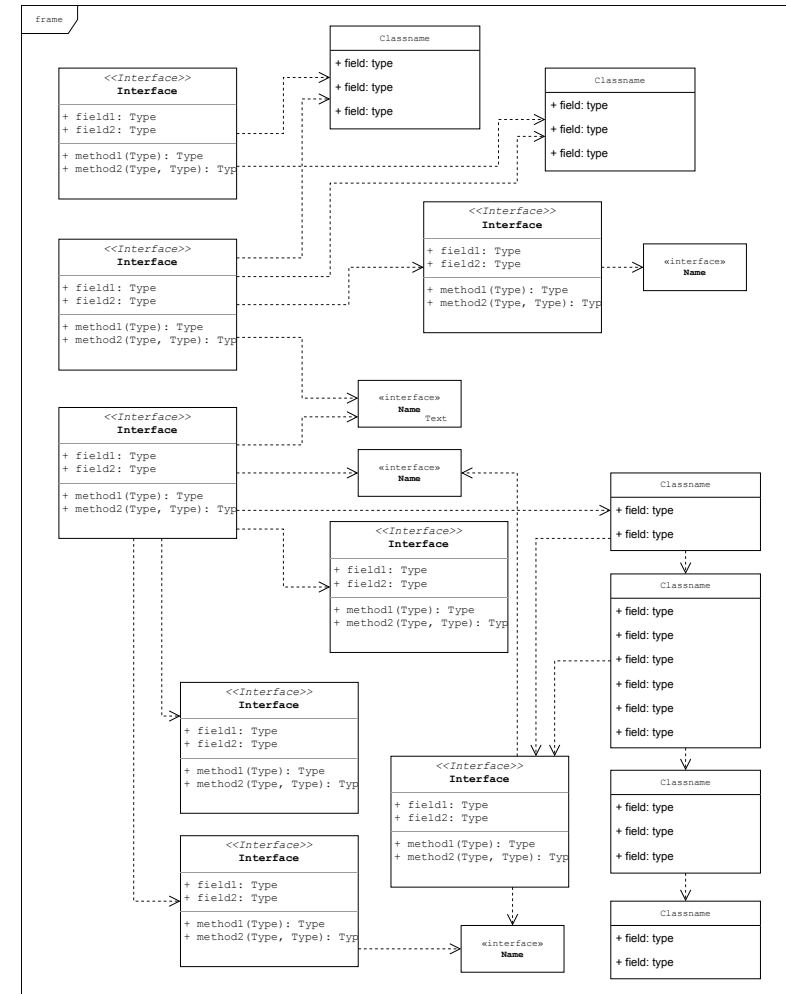
DFD

DFDs are used to model the flow of data within a system or process. DFDs consist of processes, data stores, and data flows, and are commonly used in system analysis and design.



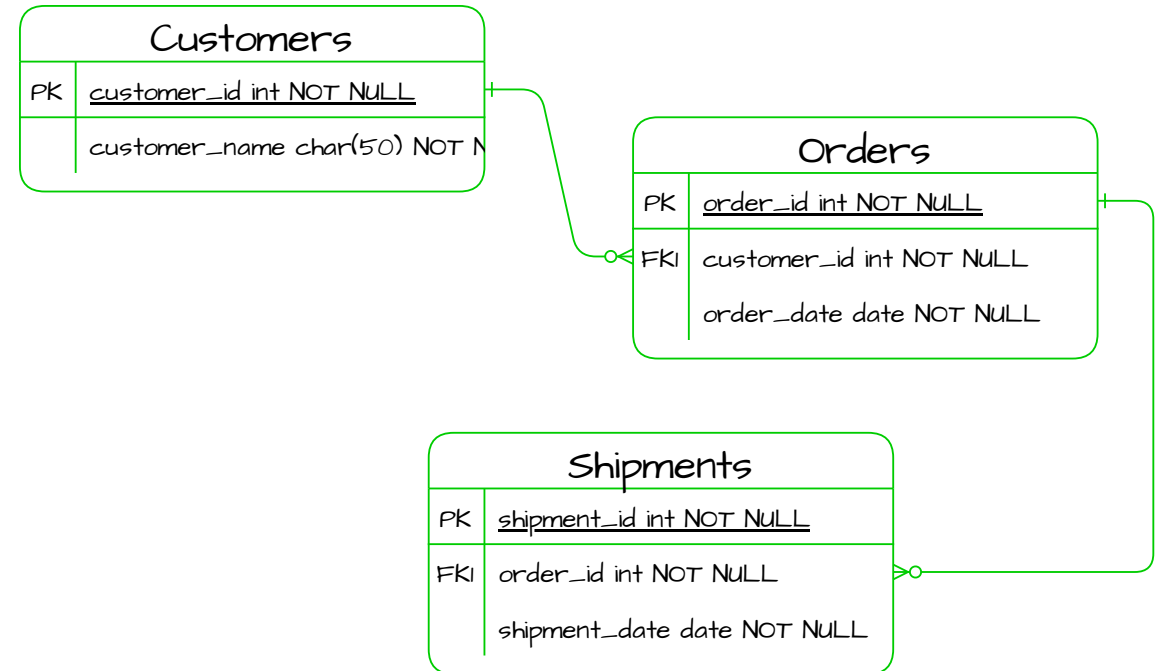
UML

UML is a general-purpose modeling language that can be used to model a wide range of systems, including software systems and business processes. UML diagrams can represent classes, objects, use cases, activities, and more.



ERD

ER notation is used to create data models that represent entities (objects) and the relationships between them. ER diagrams consist of entities, attributes, and relationships, and are commonly used in database design. ER diagrams are the most known and used notations in the data domain.



Few Popular Data Modeling Tools

- [Microsoft Visio](#)
- [MySQL Workbench](#)
- [Draw.io](#)
- [Lucidchart](#)
- [Oracle SQL Developer Data Modeler](#)

DATABASES

What is a Database?

A database is an organized collection of data, generally stored and accessed electronically from a computer system.

Types of Databases

- **Relational Database:** Data is stored in tables and relationships are established using primary and foreign keys.
- **NoSQL Database:** Data is stored in a non-tabular format, such as key-value pairs, document stores, or graph databases.

Relational Database

- MySQL
- PostgreSQL
- SQLite
- Oracle

NoSQL Database

- MongoDB
- Cassandra
- Couchbase
- Redis

Properties of Database (ACID)

- **Atomicity:** All the operations in a transaction are treated as a single unit of work.
- **Consistency:** The database remains in a consistent state before and after the transaction.
- **Isolation:** Transactions are executed in isolation from each other.
- **Durability:** Once a transaction is committed, the changes made by the transaction are permanent.

Design a database

1. Determine the purpose of the database
2. Find and organize the information required
3. Divide the information into tables
4. Turn information items into columns
5. Specify primary keys
6. Set up the table relationships
7. Refine the design
8. Apply the **normalization rules**

Design a database for a bank

- **Purpose:** To store customer, account and transaction information for a bank.
- **Information:** Customer name, account number, account balance, transaction amount, transaction date.
- **Tables:** Customers, Accounts, Transactions

- **Columns:**
 - Customers: Customer Name, Customer Address
 - Accounts: Account Number, Customer Name, Account Balance
 - Transactions: Transaction ID, Account Number, Transaction Amount, Txn Date
- **Primary Keys:**
 - Customers: Customer Name
 - Accounts: Account Number
 - Transactions: Transaction ID
- **Table Relationships:**
 - Customers and Accounts: One-to-many relationship
 - Accounts and Transactions: One-to-many relationship

Normalization

Normalization is the process of organizing the columns and tables of a relational database to **minimize data redundancy and dependency**. Normalization usually involves dividing a database into two or more tables and defining relationships between the tables.

Employee Name	Address	Skills
John Doe	123 Main St.	SQL, Python
Jane Smith	456 Elm St.	Python, R

1NF - First Normal Form

Employee Name	Address	Skill
John Doe	123 Main St.	SQL
John Doe	123 Main St.	Python
Jane Smith	456 Elm St.	Python
Jane Smith	456 Elm St.	R

2NF - Second Normal Form

ID	Employee Name	Account Address
1	John Doe	123 Main St.
2	Jane Smith	456 Elm St.

Employee ID	Skill
1	SQL
1	Python
2	R

3NF - Third Normal Form

ID	Employee Name	Account Address
1	John Doe	123 Main St.
2	Jane Smith	456 Elm St.

ID	Skill
1	SQL
1	Python
2	R

Employee ID	Skill ID
1	1
1	2
2	2

Summary

- **Data modeling** is the process of organizing data elements and how they relate to one another.
- **Database** is an organized collection of data.
- **Relational databases** store data in tables and use primary and foreign keys to establish relationships between tables.
- **Normalization** is the process of organizing a database to minimize data redundancy and dependency.
- **ACID** properties ensure that database transactions are processed reliably.
- **Designing a database** involves determining the purpose of the database, organizing the information, dividing the information into tables, and applying normalization rules.

SQL (Structured Query Language)

SQL

- SQL is a standard language for storing, manipulating, and retrieving data in databases.
- SQL is used to query, insert, update, and delete data, and to create and modify database structures.

- DDL (Data Definition Language)
 - CREATE, ALTER, DROP
- DML (Data Manipulation Language)
 - SELECT, INSERT, UPDATE, DELETE
- DCL (Data Control Language)
 - GRANT, REVOKE
- TCL (Transaction Control Language)
 - COMMIT, ROLLBACK

SQL Syntax

- SQL keywords are case-insensitive.
- SQL statements end with a semicolon.
- SQL comments start with `--` and end at the end of the line.

```
-- This is a comment  
SELECT * FROM customers;
```

Most Important SQL Commands

- **CREATE DATABASE:** Create a new database
- **ALTER DATABASE:** Modify a database
- **DROP DATABASE:** Delete a database

- **CREATE TABLE:** Create a new table
- **ALTER TABLE:** Modify a table
- **DROP TABLE:** Delete a table

- **INSERT INTO:** Insert new data into a database
- **SELECT:** Extract data from a database
- **UPDATE:** Update data in a database
- **DELETE:** Delete data from a database
- **CREATE INDEX:** Create an index (search key)
- **DROP INDEX:** Delete an index

SELECT Statement

```
SELECT column1, column2, ...FROM table_name;
```

SELECT DISTINCT Statement

DISTINCT keyword is used to return only distinct (different) values.

```
SELECT DISTINCT column1, column2, ...FROM table_name;
```


WHERE Clause

The WHERE clause is used to filter records.

```
SELECT column1, column2, ...  
FROM table_name  
WHERE condition;
```

Order By

The ORDER BY keyword is used to sort the result-set in ascending or descending order.

```
SELECT column1, column2, ...  
FROM table_name  
ORDER BY column1, column2, ... ASC|DESC;
```

And, Or, Not (Where Clause)

The AND, OR, and NOT operators are used to filter records based on more than one condition.

```
SELECT column1, column2, ...  
FROM table_name  
WHERE condition1 AND/OR condition2;
```

BETWEEN Operator

The BETWEEN operator selects values within a given range.

```
SELECT column1, column2, ...  
FROM table_name  
WHERE column_name BETWEEN value1 AND value2;
```

Aggregate Functions

- **COUNT()**: Returns the number of rows that matches a specified criteria.
- **SUM()**: Returns the total sum of a numeric column.
- **AVG()**: Returns the average value of a numeric column.
- **MIN()**: Returns the minimum value of a column.
- **MAX()**: Returns the maximum value of a column.

LIKE Operator and Wildcards

The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

- `%` : The percent sign represents zero, one, or multiple characters.
- `_` : The underscore represents a single character.

IN Operator

The IN operator allows you to specify multiple values in a WHERE clause.

```
SELECT column1, column2, ...  
FROM table_name  
WHERE column_name IN | NOT IN (value1, value2, ...);
```

Aliases

SQL aliases are used to give a table, or a column in a table, a temporary name.

```
SELECT column_name AS alias_name  
FROM table_name;
```

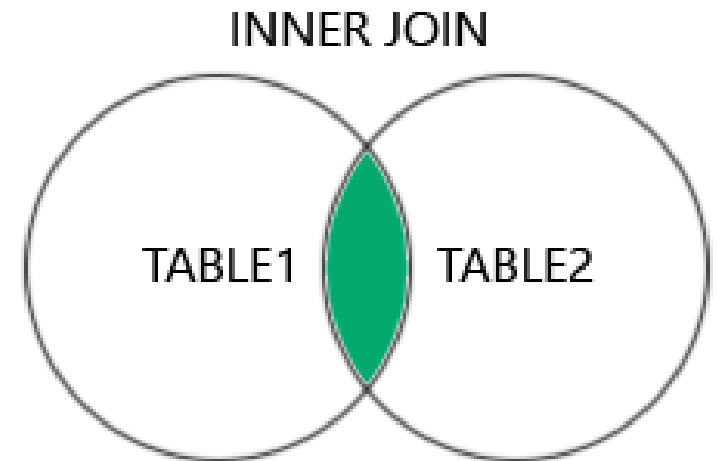
```
SELECT * FROM table_name AS alias_name;
```


SQL Joins

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

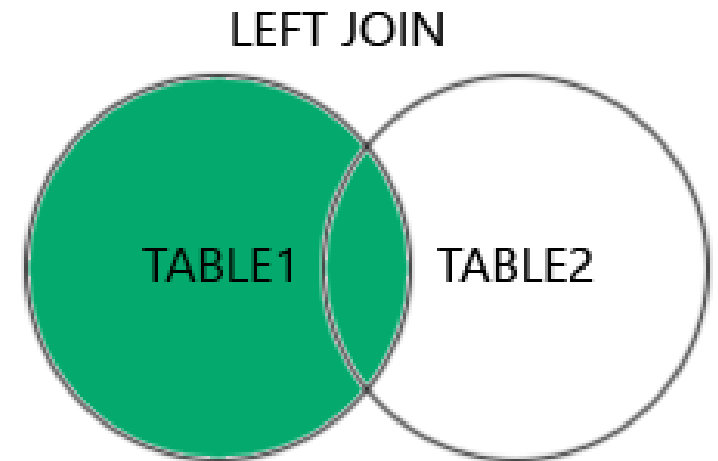
INNER JOIN: Returns records that have matching values in both tables.

```
select column_name(s)
from table1
inner join table2 on table1.column_name = table2.column_name;
```



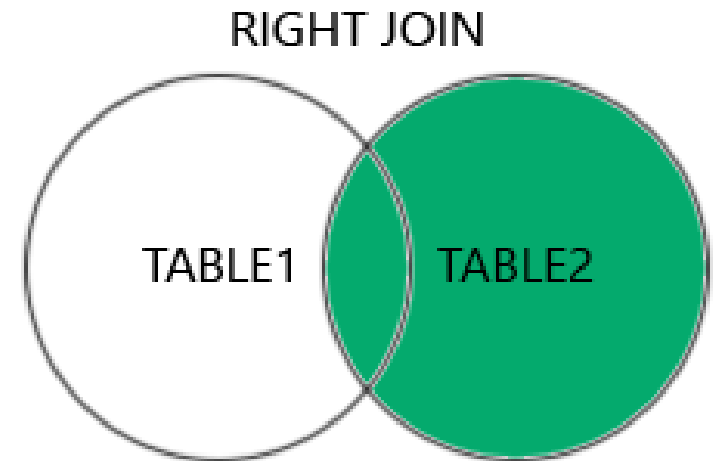
LEFT JOIN: Returns all records from the left table, and the matched records from the right table.

```
select column_name(s)
from table1
left join table2 on table1.column_name = table2.column_name;
```



RIGHT JOIN: Returns all records from the right table, and the matched records from the left table.

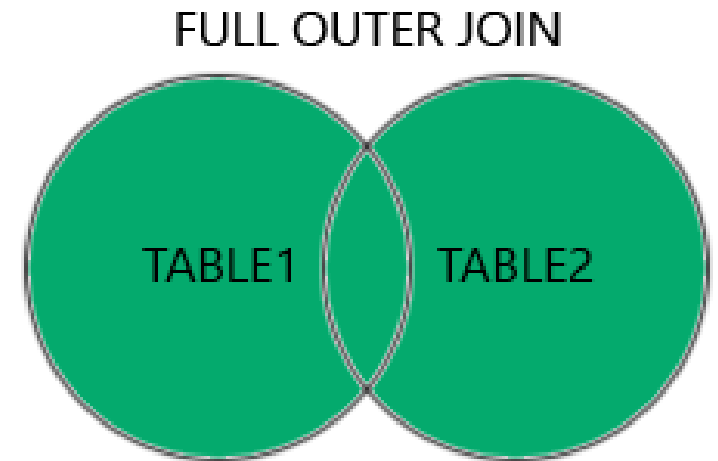
```
select column_name(s)
from table1
right join table2 on table1.column_name = table2.column_name;
```



FULL JOIN / FULL OUTER JOIN: Returns all records when there is a match in either left or right table.

```
select column_name(s)
from table1
full join table2 on table1.column_name = table2.column_name;
```

It may produce very large result sets!



UNION Operator

The UNION operator is used to combine the result-set of two or more SELECT statements.

```
SELECT column_name(s) FROM table1  
UNION  
SELECT column_name(s) FROM table2;
```

- Each SELECT statement within UNION must have the same number of columns.
- The columns must also have similar data types.
- The columns in each SELECT statement must also be in the same order.

GROUP BY Clause

The GROUP BY clause is used to group the result-set by one or more columns.

Often used with aggregate functions (COUNT, MAX, MIN, SUM, AVG) to group the result-set by one or more columns.

```
SELECT column_name(s) FROM table_name WHERE condition  
GROUP BY column_name(s)  
ORDER BY column_name(s);
```

HAVING Clause

The HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions.

```
SELECT column_name(s) FROM table_name WHERE condition  
GROUP BY column_name(s)  
HAVING condition;
```


DDL Commands

- **CREATE:** Create a new table, view, or database.
- **ALTER:** Modify an existing database object, such as a table.
- **DROP:** Delete an existing database object, such as a table.

CREATE TABLE

```
CREATE TABLE table_name (  
    column1 datatype,  
    column2 datatype,  
    column3 datatype,  
    ....  
);
```

ALTER TABLE

```
ALTER TABLE table_name  
ADD column_name datatype;
```

DROP TABLE

```
DROP TABLE table_name;
```

Data Types

- **INT**: A whole number.
- **VARCHAR(size)**: A variable-length string with a maximum size.
- **DATE**: A date in the format YYYY-MM-DD.
- **FLOAT**: A floating-point number.
- **BOOLEAN**: A true or false value.

Summary

- DDL commands are used to create, modify, and delete database objects.
- DML commands are used to retrieve, insert, update, and delete data.
- SQL queries can be used to filter, sort, and group data.
- SQL joins are used to combine rows from two or more tables based on a related column between them.

Thank You

Happy Learning 🚀 !!

meftaulhaque@gmail.com