Database Basics

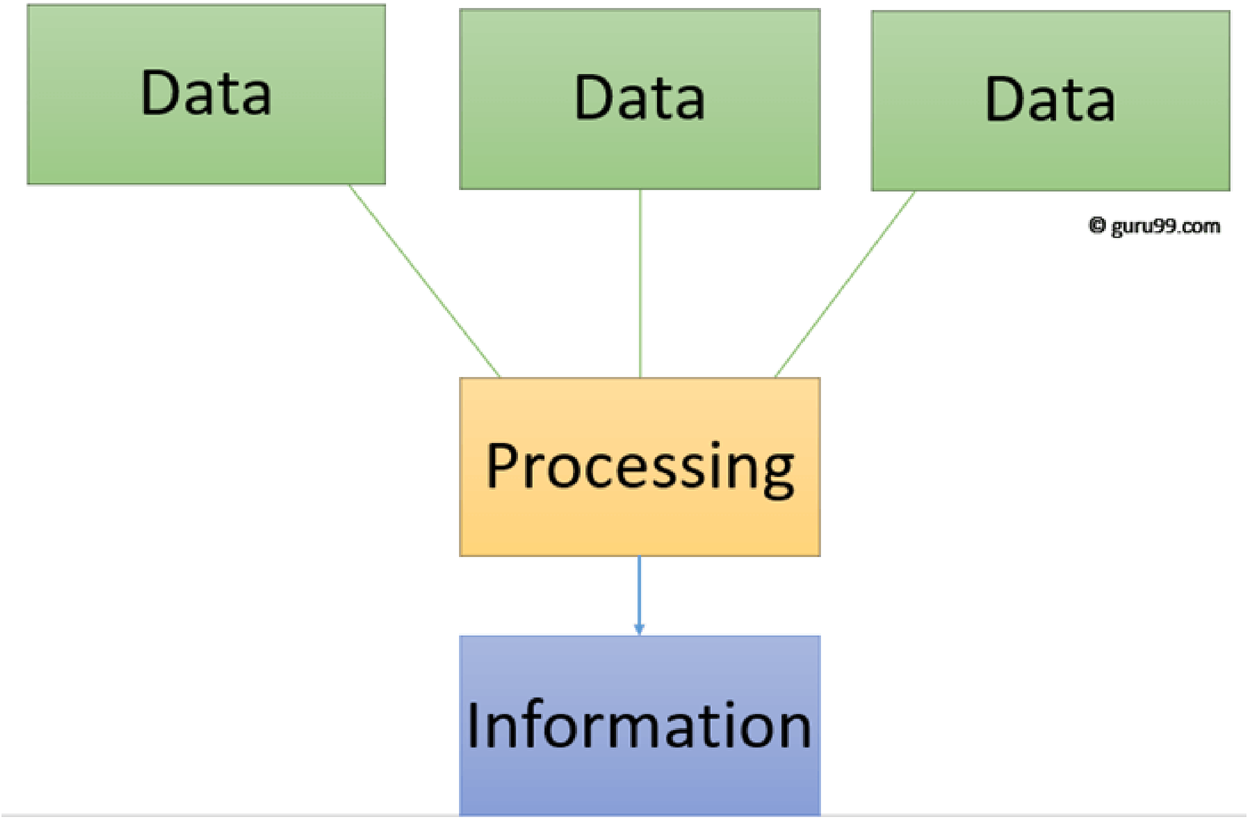
What is DATA?

- Data is raw fact or figures or entity.

- When activities in the organization takes place, the effect of these activities need to be recorded which is known as Data.

What is INFORMATION?

- Processed data is called information.
- The purpose of data processing is to generate the information required for carrying out the business activities.



Introduction to Databases

- A database is a structured collection of data that is organized and stored in a way that allows for efficient retrieval, management, and update of information.
- Databases are fundamental to modern computing and play a crucial role in storing and managing vast amounts of data for various applications and industries.

Database

- Database may be defined in simple terms as a collection of data

- A database is a collection of related data.

- The database can be of any size and of varying complexity.

- A database may be generated and maintained manually or it may be computerized.

Database Management System

- Database Management Systems (**DBMS**) are software systems used to store, retrieve, and run queries on data.

- *A DBMS serves as an interface between an end-user and a database, allowing users to create, read, update, and delete data in the database.*

- A DBMS is a collection of program that enables user to create and maintain a database.

Characteristics of DBMS

- Data Definition and Schema Management
- Data Manipulation
- Transaction Management
- Query Optimization
- Security and Access Control
- Security and Access Control
- Data Recovery and Backup
- Scalability
- Data Independence
- Query Language Support
- Backup and Recovery

DBMS Utilities

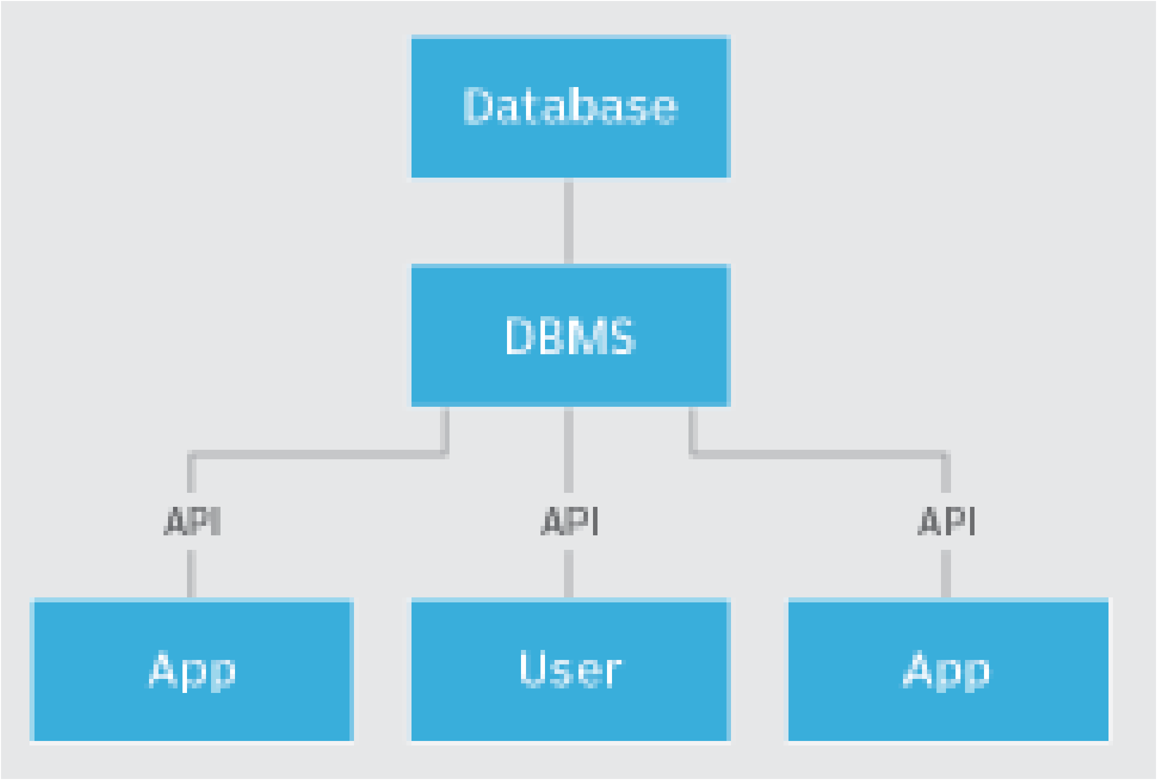
- A data loading utility:
  - Which allows easy loading of data from the external format without writing programs.
- A backup utility:
  - Which allows to make copies of the database periodically to help in cases of crashes and disasters.
- Recovery utility:
  - Which allows to reconstruct the correct state of database from the backup and history of transactions.

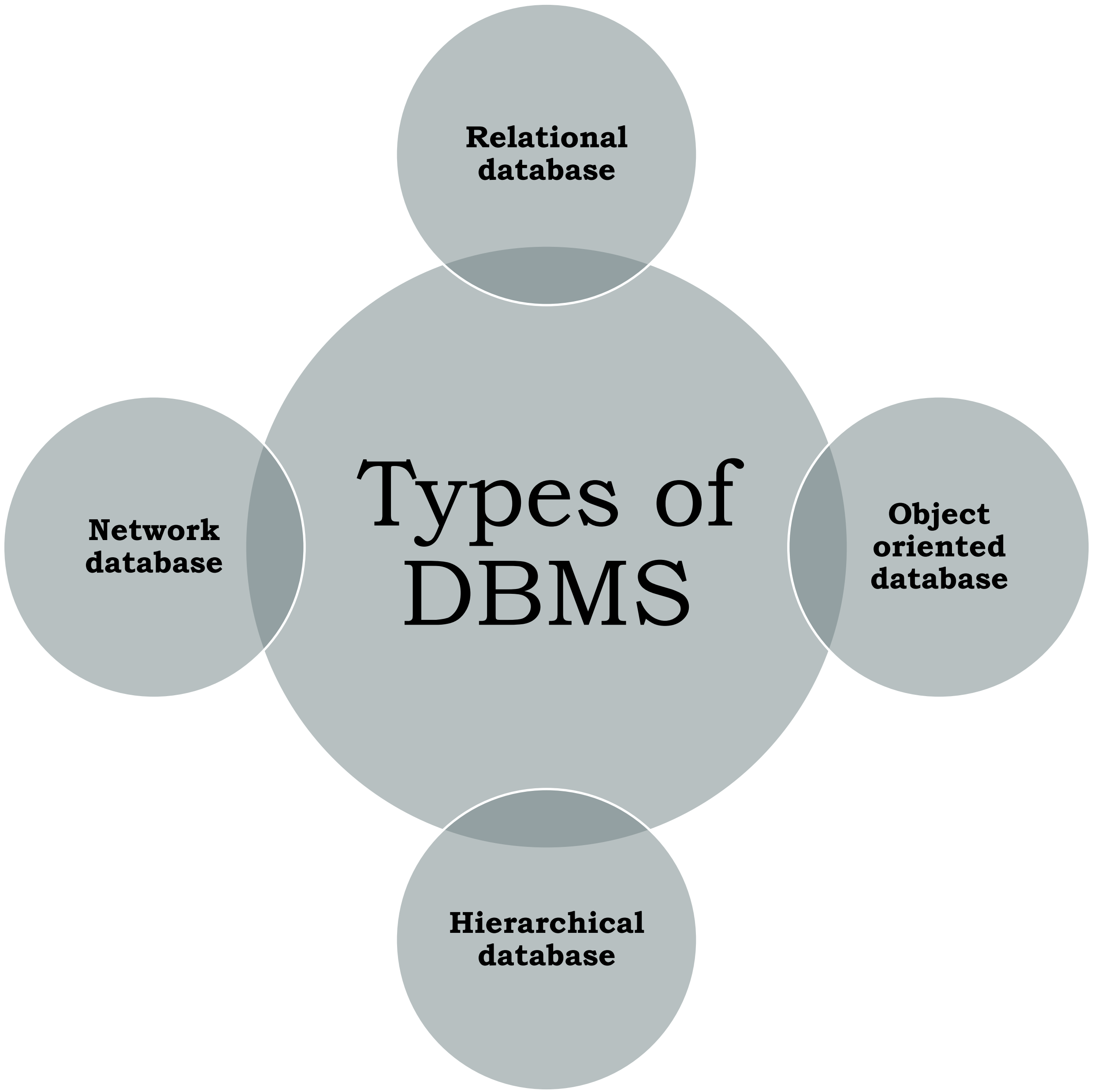
File system VS Database

File System	DBMS
File system is a collection of data. Any management with the file system, user has to write the procedures.	DBMS is a collection of data and user is not required to write the procedures for managing the database.
File system gives the details of the data representation and Storage of data.	DBMS provides an abstract view of data that hides the details.
In File system storing and retrieving of data cannot be done efficiently.	DBMS is efficient to use since there are wide varieties of sophisticated techniques to store and retrieve the data.
Concurrent access to the data in the file system has many problems.	DBMS takes care of Concurrent access using some form of locking.
File system doesn’t provide crash recovery mechanism.	DBMS has crash recovery mechanism, DBMS protects user from the effects of system failures.
Protecting a file under file system is very difficult.	DBMS has a good protection mechanism.

Advantages of DBMS

- Data independency
- Efficient data access
- Data integrity and security
- Data Administration
- Concurrent access and Crash recovery
- Reduced application development time





Types of DBMS

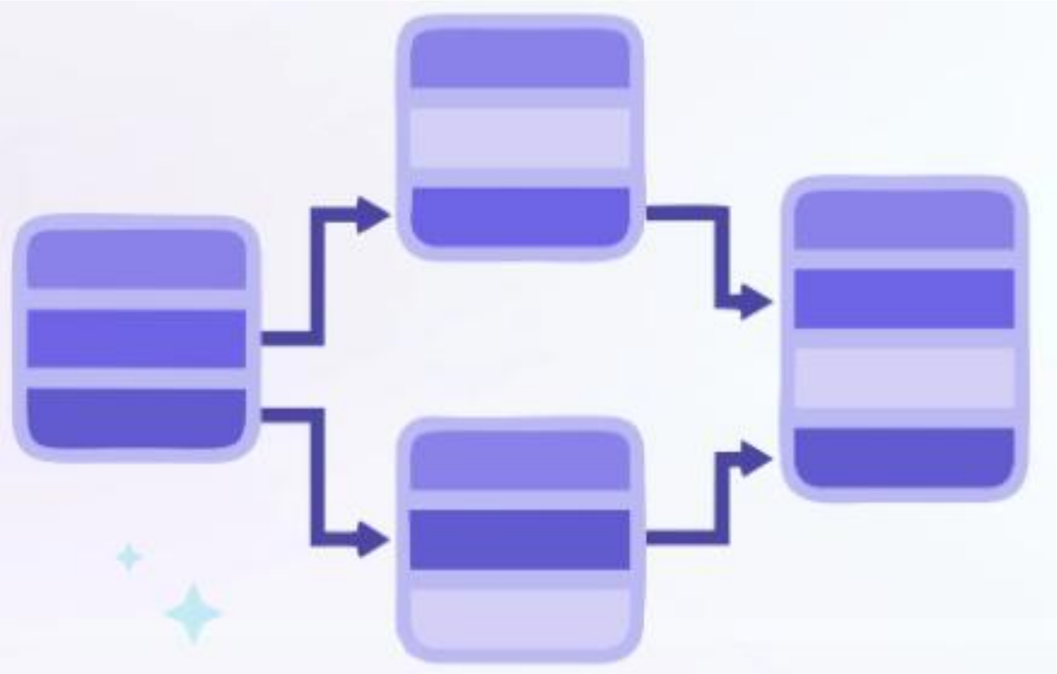
Relational database

Network database

Object oriented database

Hierarchical database

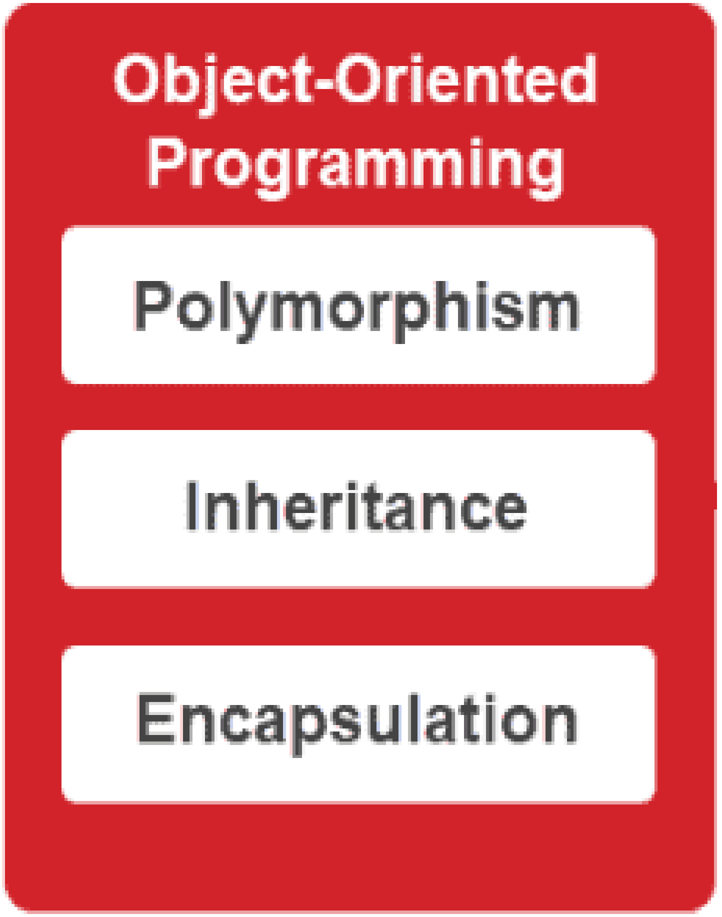
Relation Database



- It is based on SQL.
- A relational database management system.
- This is one of the most popular data models which is used in industries.
- Every table in a database has a key field which uniquely identifies each record.
- RDBMS is a system where data is organized in two-dimensional tables using rows and columns.

Object Oriented Database

- It is a combination of relational database concepts and object-oriented principles.
- OOPs principles are data encapsulation, inheritance, and polymorphism.
- It requires less code and is easy to maintain.
- For example – Object DB software.



Object-Oriented Programming

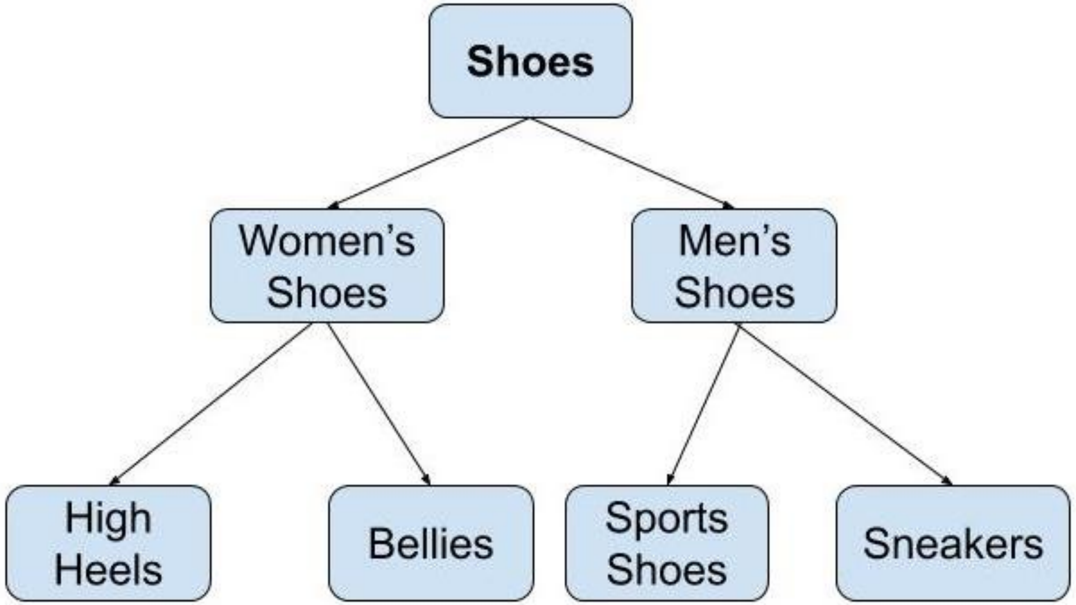
Polymorphism

Inheritance

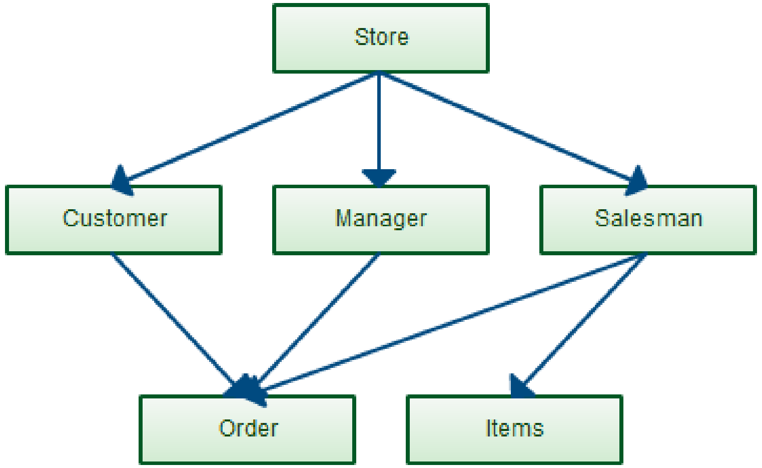
Encapsulation

Hierarchical Database

- It is used in industry on mainframe platforms.
- The hierarchy starts from the root node, connecting all the child nodes to the parent node.
- For example
  - IMS (IBM)
  - Windows registry (Microsoft).

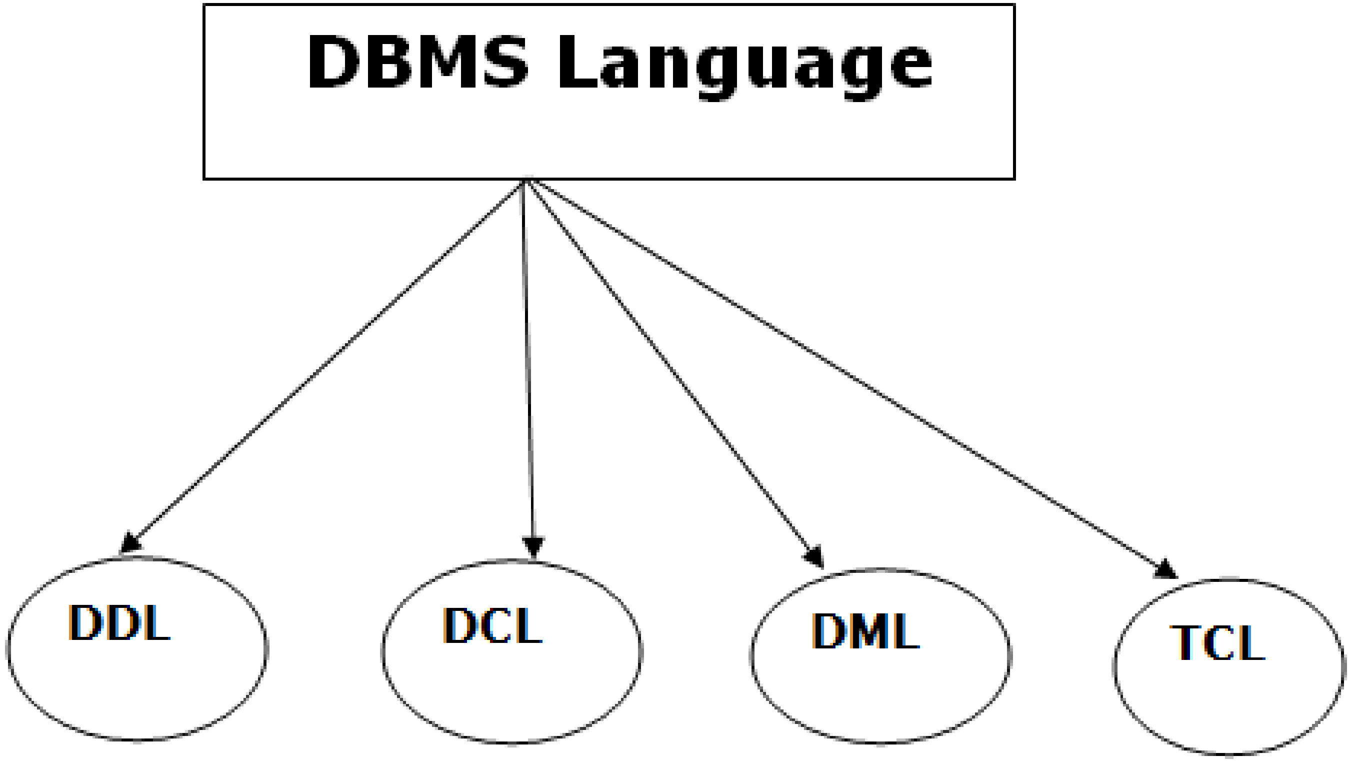


Network database



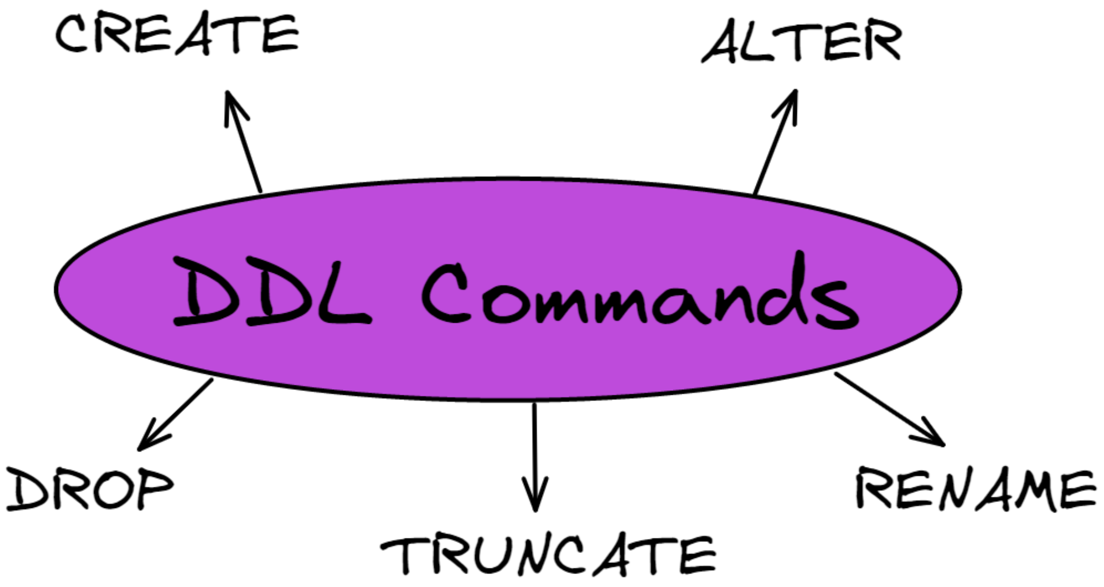
- A Network database management system.
- This maintain one to one relationship (1: 1) or many to many relationship (N: N).
- It is based on a network data model, which allows each record to be related to multiple primary records and multiple secondary records.

DBMS Language



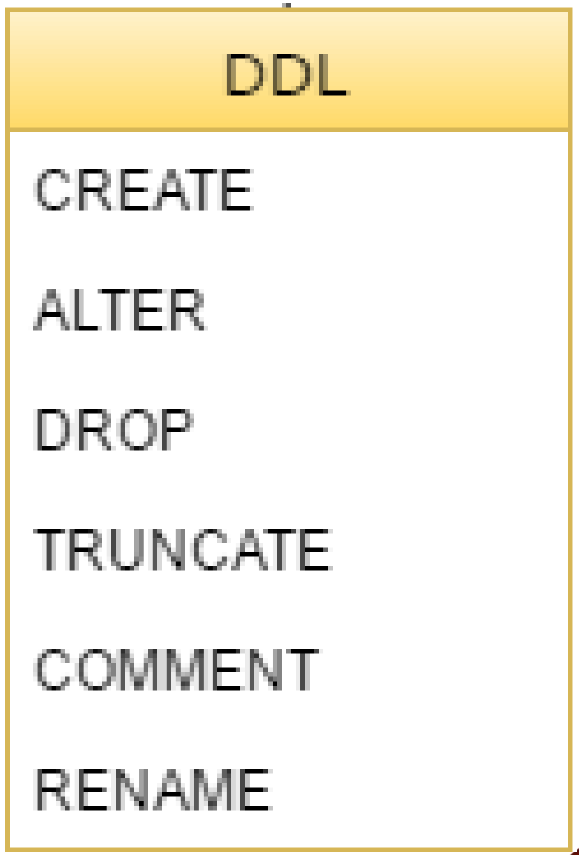
Data Definition Language (DDL)

- It is used to define database structure or pattern.
- It is used to create schema, tables, indexes, constraints, etc. in the database.
- Using the DDL statements, you can create the skeleton of the database.



Data Definition Language (DDL)

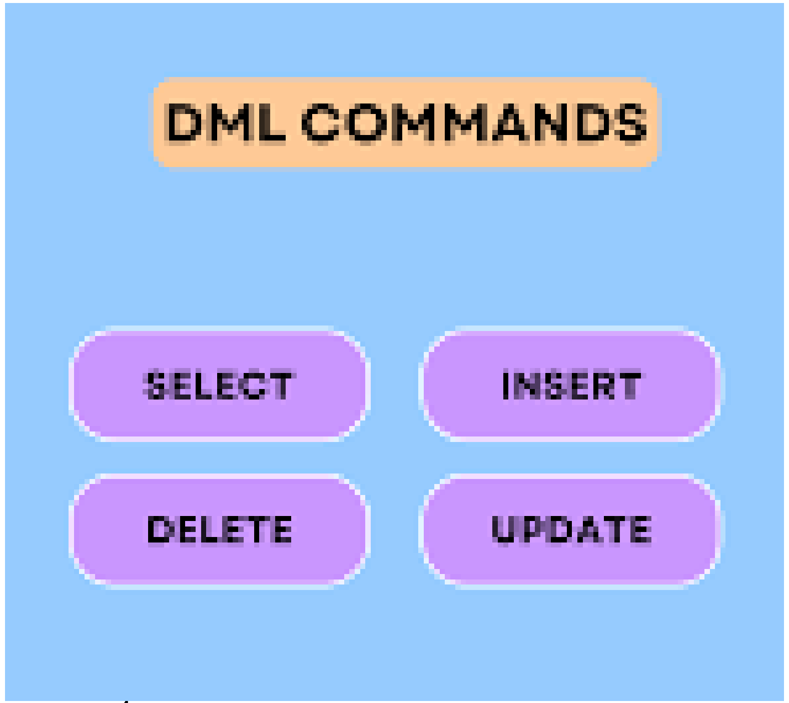
- Create: It is used to create objects in the database.
- Alter: It is used to alter the structure of the database.
- Drop: It is used to delete objects from the database.
- Truncate: It is used to remove all records from a table.
- Rename: It is used to rename an object.
- Comment: It is used to comment on the data dictionary.



DDL
CREATE
ALTER
DROP
TRUNCATE
COMMENT
RENAME

Data Manipulation Language (DML)

- It is used for **accessing and manipulating** data in a database.
- Here are some tasks that come under DML:
  - **Select**: It is used to retrieve data from a database.
  - **Insert**: It is used to insert data into a table.
  - **Update**: It is used to update existing data within a table.
  - **Delete**: It is used to delete all records from a table.
  - **Merge**: It performs UPSERT operation, i.e., insert or update operations.



Data Control Language (DCL)

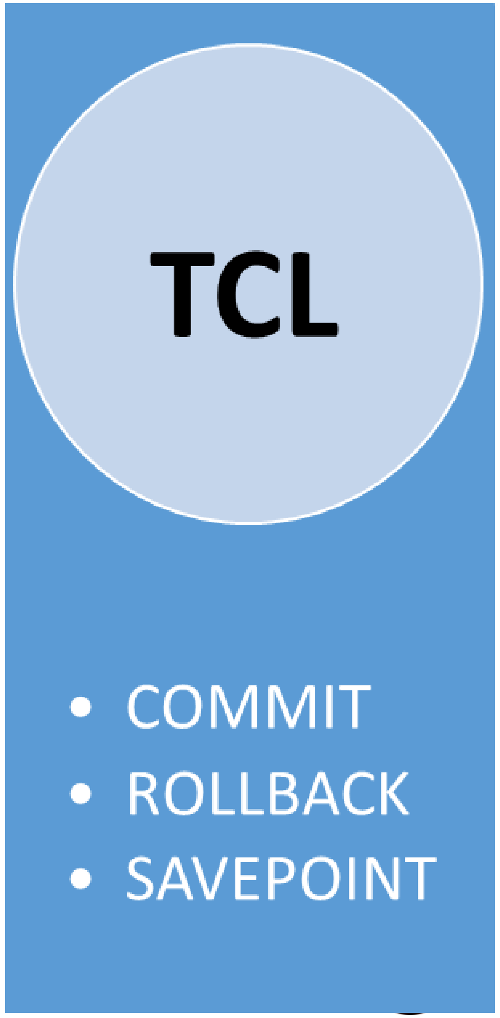
- DCL stands for Data Control Language.
- It is used to retrieve the stored or saved data.
- The DCL execution is transactional.
- It also has rollback parameters.
- Here are some tasks that come under DCL:
  - **Grant**: It is used to give user access privileges to a database.
  - **Revoke**: It is used to take back permissions from the user.

DCL

- GRANT
- REVOKE

Transaction Control Language (TCL)

- TCL is used to run the changes made by the DML statement.
- TCL can be grouped into a logical transaction.
- Here are some tasks that come under TCL:
  - **Commit**: It is used to save the transaction on the database.
  - **Rollback**: It is used to restore the database to original since the last Commit.



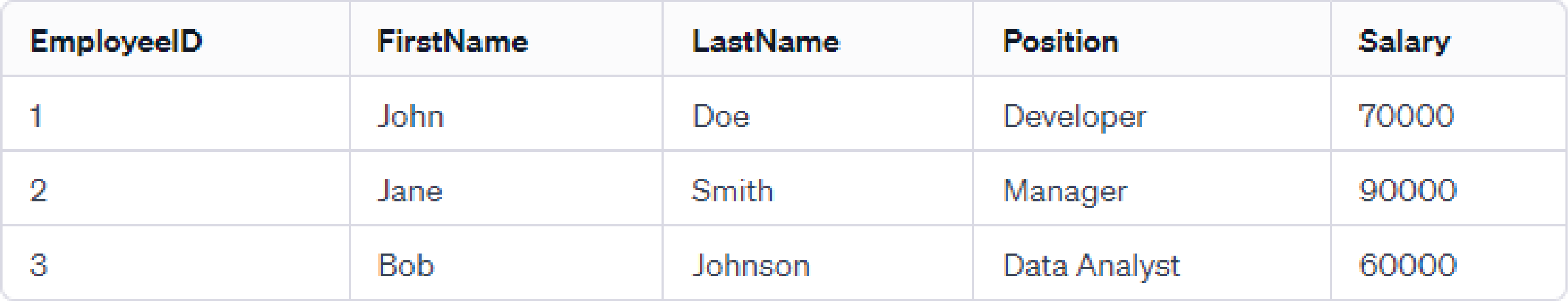
TCL

- COMMIT
- ROLLBACK
- SAVEPOINT

Key Concepts

These are fundamental components that help organize and structure data

- Tables
- Rows
- Columns



EmployeeID	FirstName	LastName	Position	Salary
1	John	Doe	Developer	70000
2	Jane	Smith	Manager	90000
3	Bob	Johnson	Data Analyst	60000

Key Concepts - Tables

- A table is a structured representation of data in a relational database.
- It is often compared to a spreadsheet, where data is organized into rows and columns.
- Tables are used to store information about a specific entity or concept, such as customers, products, or orders.

Key Concepts - Rows

- Also known as records or tuples, rows represent individual entries in a table.
- Each row contains data related to a specific instance of the entity represented by the table.

Key Concepts - Columns

- Columns, also referred to as fields or attributes, represent the different properties or attributes of the data stored in a table.
- Each column in a table has a specific data type (such as text, number, date) and holds a particular kind of information.

Terminologies used in DBMS

- Primary key
- Foreign key
- Index
- Query
- Normalization

- Transaction
- ACID Property
- Schema
- View

Introduction to Structured Query Language (SQL)

- SQL is a specialized programming language designed for managing and manipulating relational databases.
- It serves as the standard language for interacting with relational database management systems (RDBMS), allowing users to create, modify, and query databases.
- SQL provides a set of powerful and flexible commands for handling data, maintaining database integrity, and retrieving information efficiently.

Key aspects of SQL

- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Query Language (DQL)
- Data Control Language (DCL)

- Data Integrity
- Transactions
- Views

Data Definition Language (DDL)

- DDL defines & manage the structure of the database.
- DDL commands include:
  - **CREATE**: Used to create database objects such as tables, indexes, or views.
  - **ALTER**: Modifies the structure of existing database objects.
  - **DROP**: Deletes database objects like tables or indexes.

Data Manipulation Language (DML)

- DML statements are used to manipulate the data stored in the database.
  - **SELECT**: Retrieves data from one or more tables.
  - **INSERT**: Adds new records to a table.
  - **UPDATE**: Modifies existing records in a table.
  - **DELETE**: Removes records from a table.

Data Query Language (DQL)

- DQL is a subset of SQL used exclusively for querying and retrieving data from the database.
- The primary DQL command is **SELECT**.

Data Control Language (DCL)

DCL statements are used to control access to data within the database.

- **GRANT**: Provides specific privileges to database users.
- **REVOKE**: Removes specific privileges from database users.

Data Integrity

- SQL allows the definition of constraints to maintain data integrity.
  - **PRIMARY KEY**: Ensures the uniqueness of a column's values in a table.
  - **FOREIGN KEY**: Establishes a link between two tables based on a column.
  - **CHECK**: Defines conditions that must be satisfied for data to be entered into a table.
  - **UNIQUE**: Ensures that all values in a column are unique.

Transactions

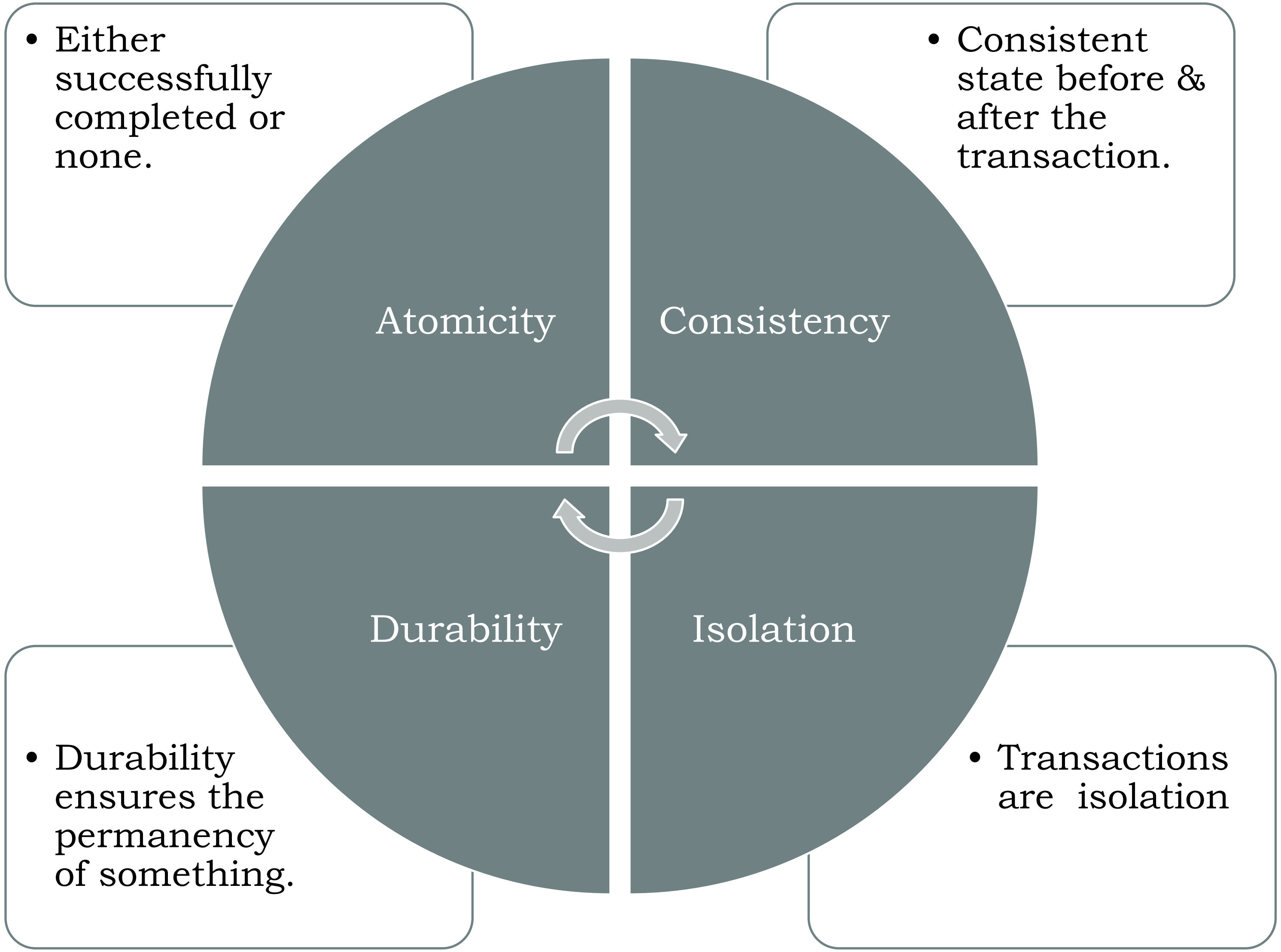
- SQL supports transaction management to ensure the consistency and reliability of database operations.
- The **ACID** properties:
  - Atomicity,
  - Consistency,
  - Isolation,
  - Durability

  define the characteristics of a transaction.

Views

- SQL allows the creation of virtual tables known as views.
- Views are based on the result of a SELECT query and provide a way to simplify complex queries or restrict access to certain columns.

ACID Properties



Atomicity

- Either successfully completed or none.

Consistency

- Consistent state before & after the transaction.

Isolation

- Transactions are isolation

Durability

- Durability ensures the permanency of something.

SQL

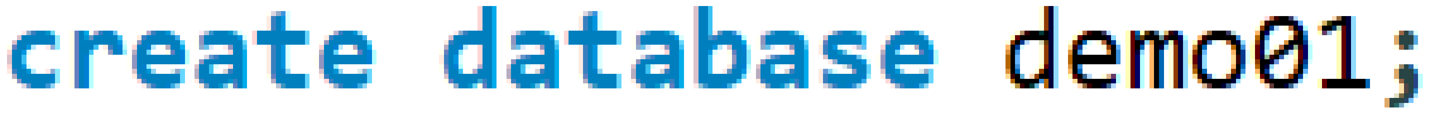
- It’s a language.
- Used for interacting with Relational DBMS (RDBMS).
- Performs C.R.U.D operation.
  - C = Create
  - R = Read/Retrieve
  - U = Update
  - D = Delete

- Also performs admin tasks:
  - Security
  - User management
  - Import/Export data
  - Backup

Create Database

Syntax to create a DB:

- CREATE DATABASE <DB-NAME>;



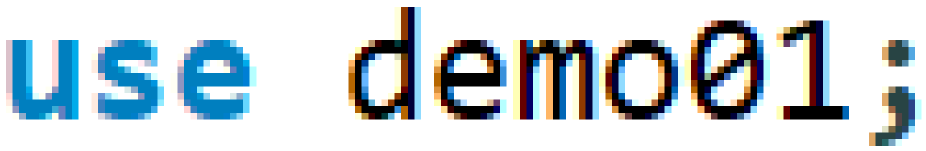
```
create database demo01;
```

Selecting DB

- In simple terms, the use statement selects a specific database and then performs operations on it using the inbuilt commands of SQL.

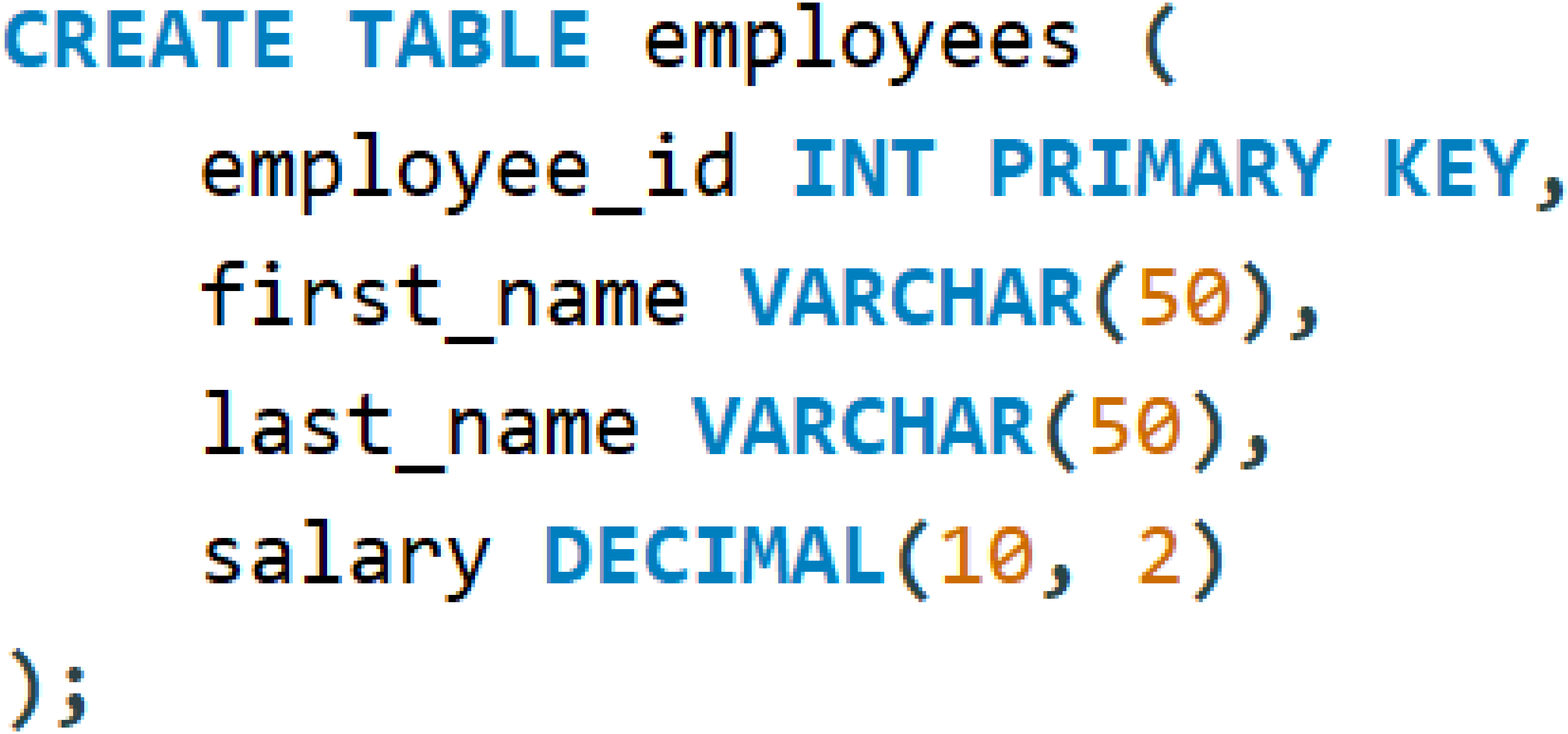
Syntax:

- USE DATABASE;



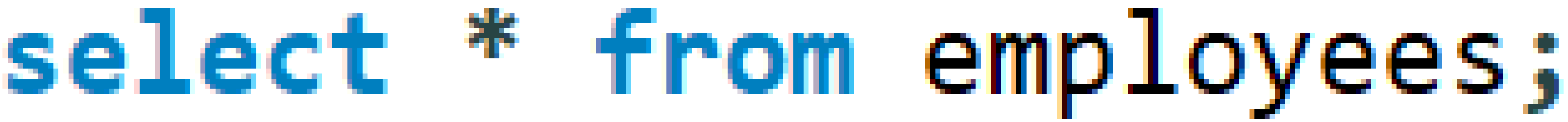
```
use demo01;
```

Creating Table



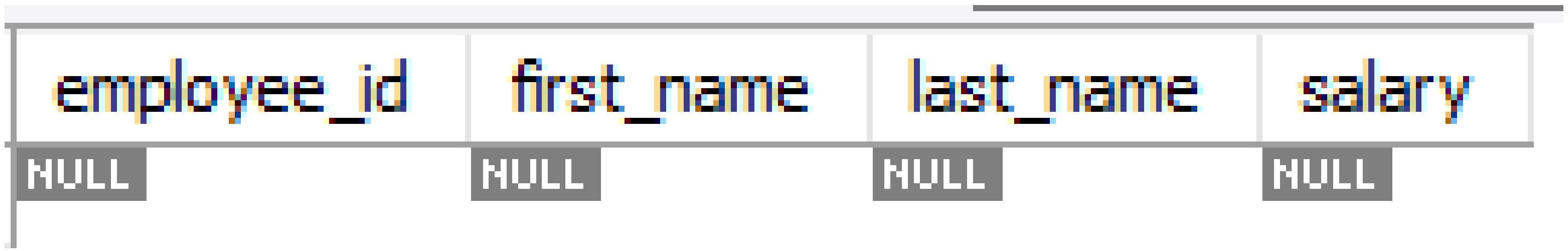
```sql
CREATE TABLE employees (
    employee_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    salary DECIMAL(10, 2)
);
```

Viewing table



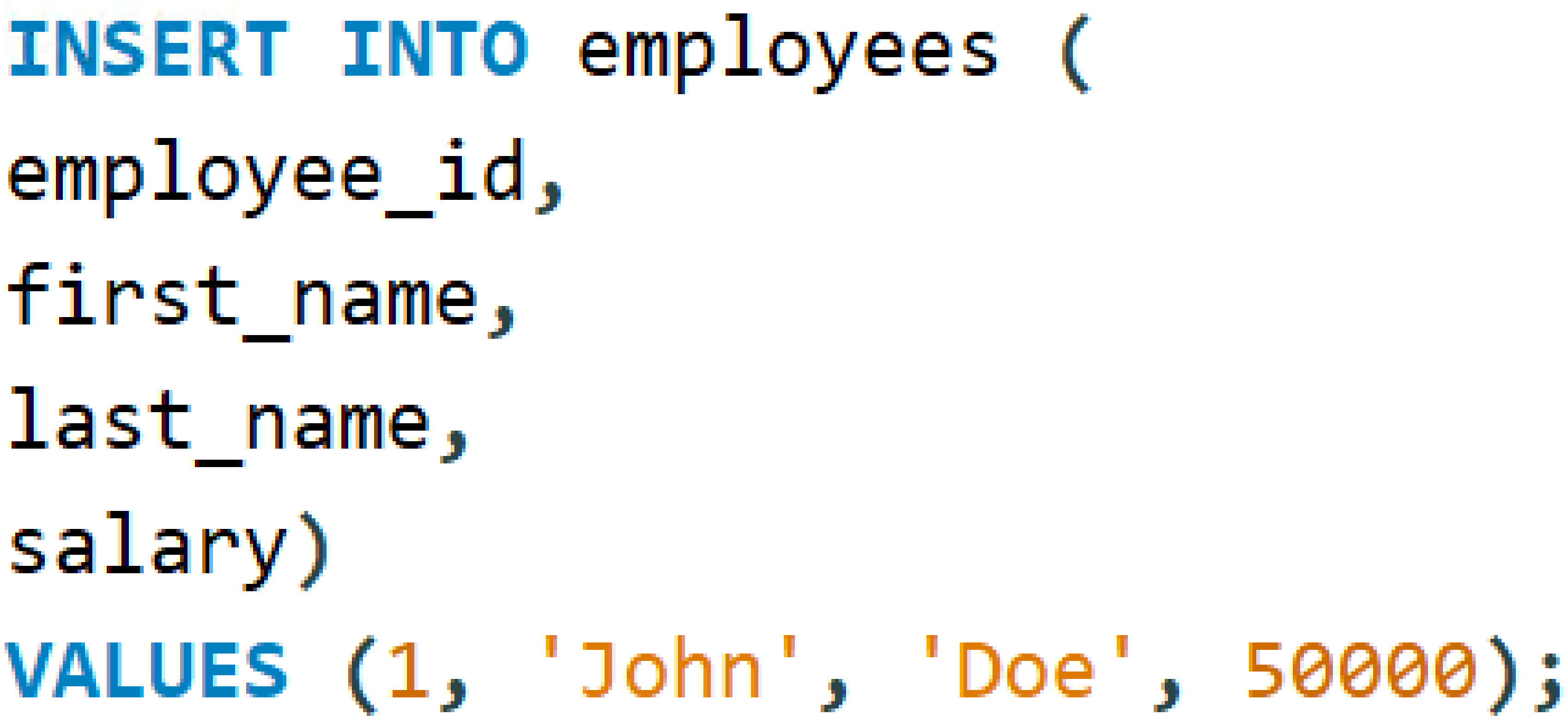
```
select * from employees;
```

Output:



employee_id	first_name	last_name	salary
NULL	NULL	NULL	NULL

Inserting 1st row in table

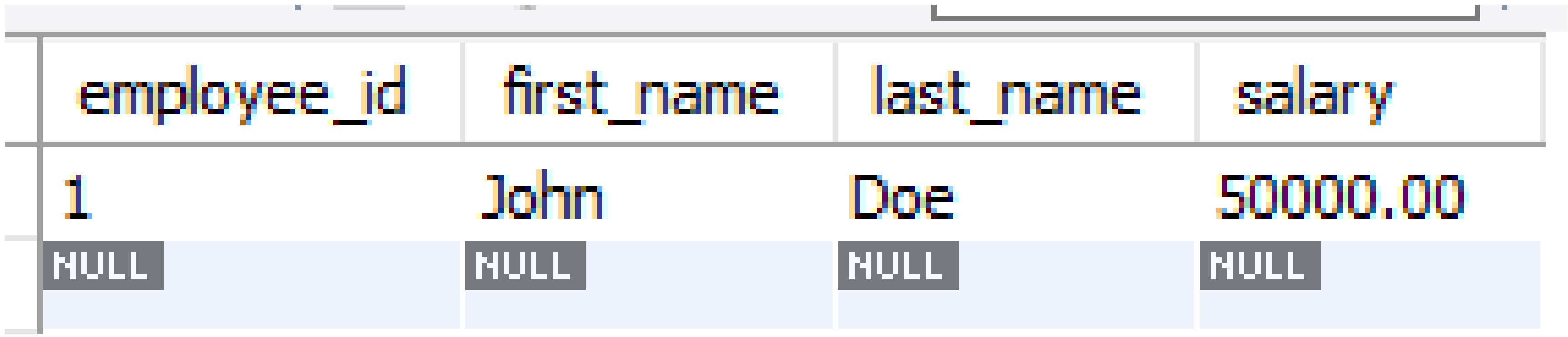


```sql
INSERT INTO employees (
employee_id,
first_name,
last_name,
salary)
VALUES (1, 'John', 'Doe', 50000);
```

Verifying:

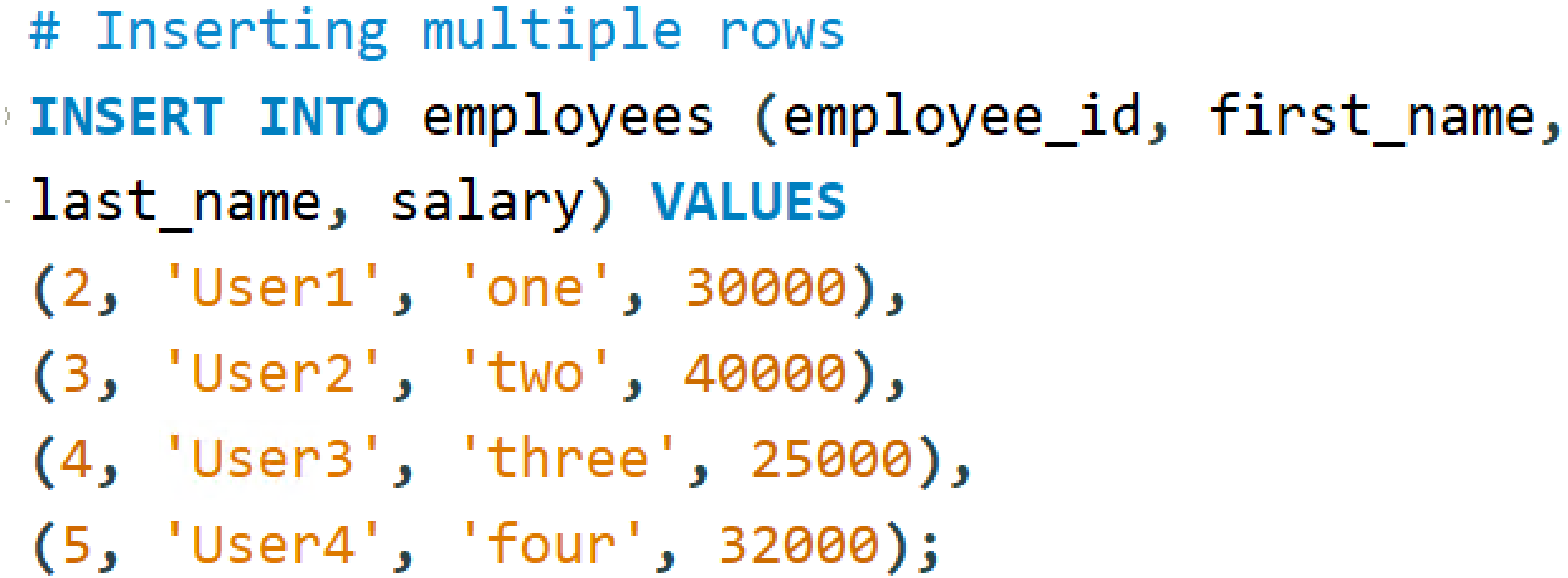
```
select * from employees;
```

Output:



employee_id	first_name	last_name	salary
1	John	Doe	50000.00
NULL	NULL	NULL	NULL

Inserting multiple values

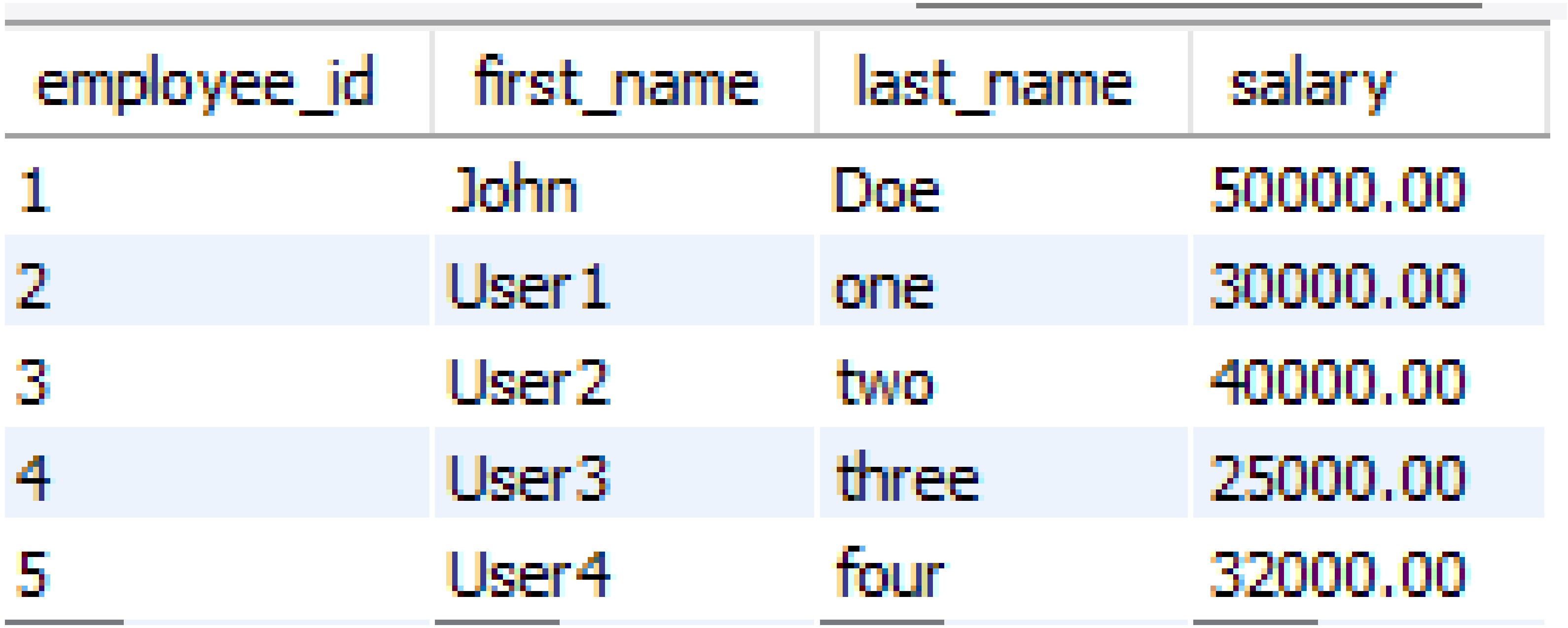


```
# Inserting multiple rows
INSERT INTO employees (employee_id, first_name,
last_name, salary) VALUES
(2, 'User1', 'one', 30000),
(3, 'User2', 'two', 40000),
(4, 'User3', 'three', 25000),
(5, 'User4', 'four', 32000);
```

Verifying:

```
select * from employees;
```

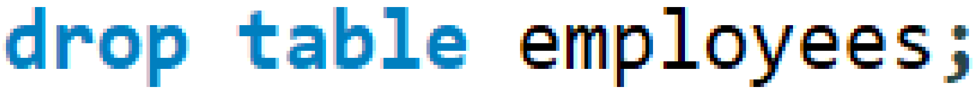
Output:



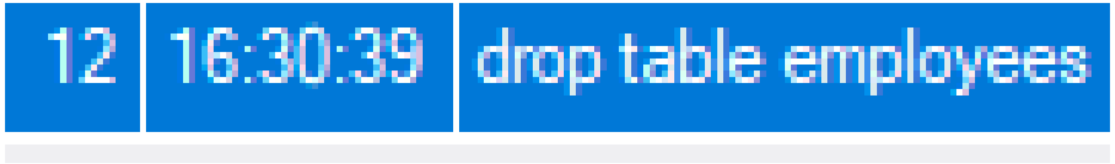
employee_id	first_name	last_name	salary
1	John	Doe	50000.00
2	User1	one	30000.00
3	User2	two	40000.00
4	User3	three	25000.00
5	User4	four	32000.00

Dropping

- Table

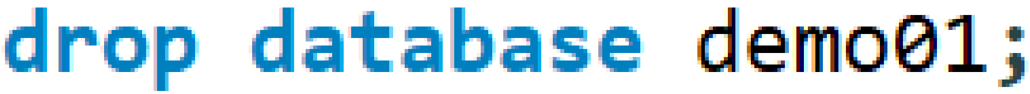


```
drop table employees;
```

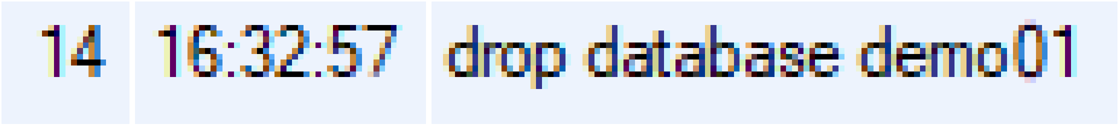


```
12  16:30:39  drop table employees
```

- Database



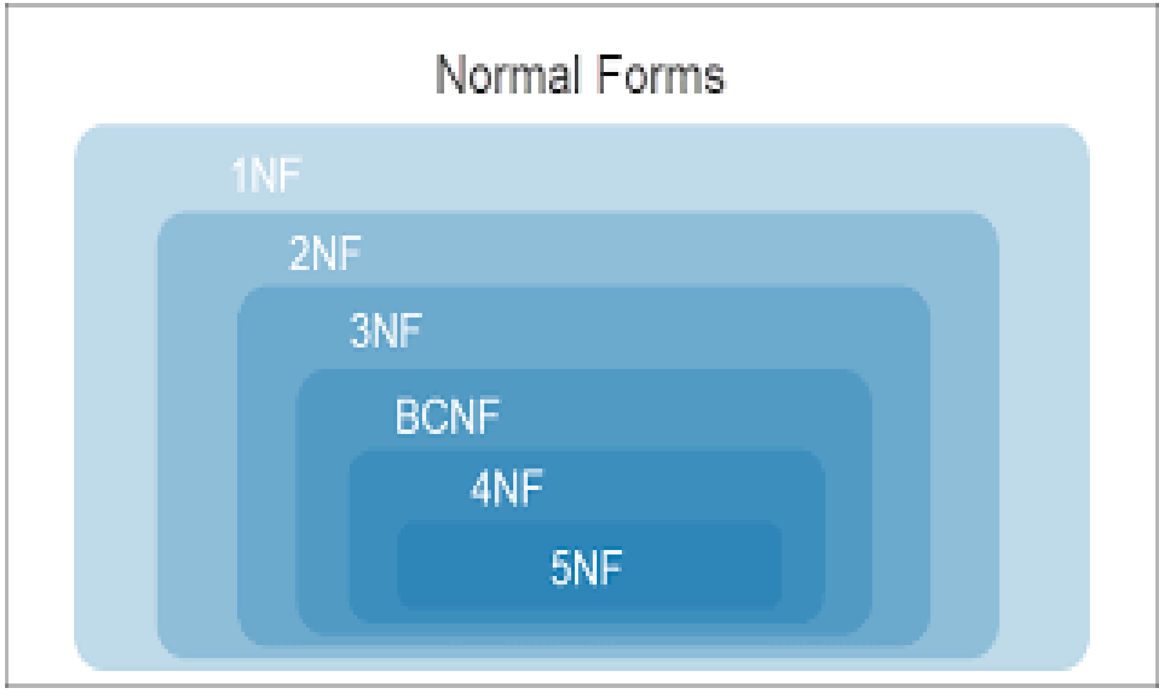
```
drop database demo01;
```



```
14  16:32:57  drop database demo01
```

Normalization

- Normalization is the process of minimizing redundancy from a relation or set of relations.

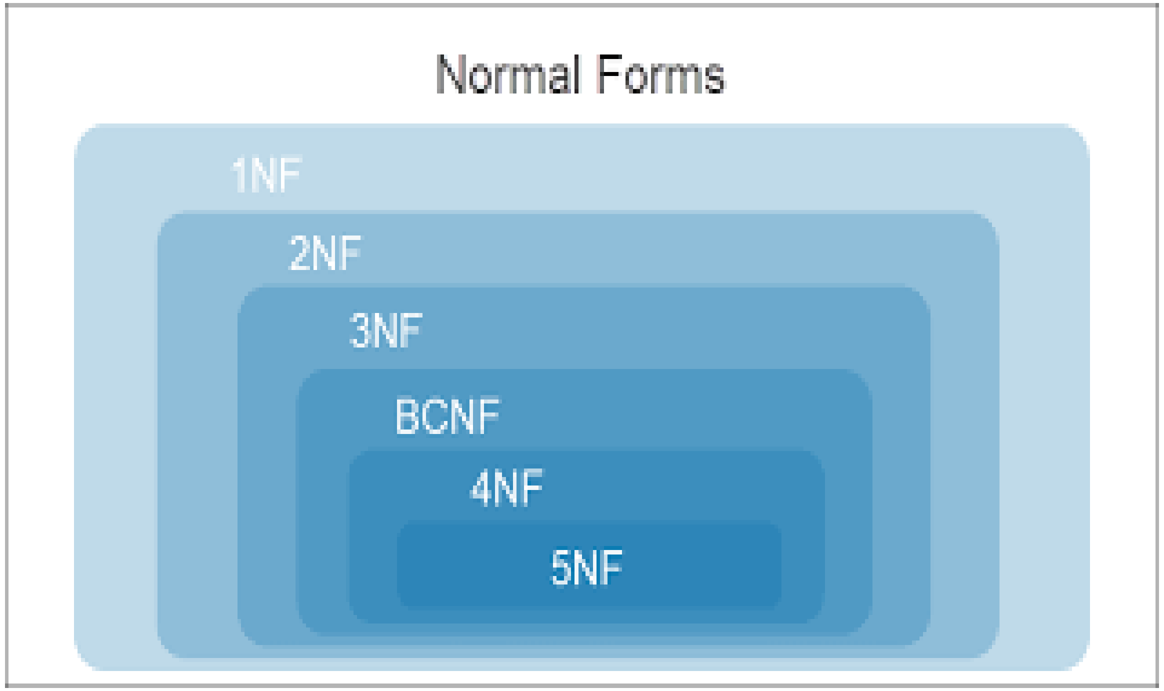


- Redundancy in relation may cause insertion, deletion, and update anomalies. So, it helps to minimize the redundancy in relations.

- Normal forms are used to eliminate or reduce redundancy in database tables.

Normalization

- First Normal Form (1NF)
- Second Normal Form (2NF)
- Third Normal Form (3NF)
- Boyce-Codd Normal Form (BCNF)
- Fourth Normal Form (4NF)
- Fifth Normal Form (5NF)



Normalization of DBMS – 1NF

- This is the most basic level of normalization.
- In 1NF, each table cell should contain only a single value, and each column should have a unique name.
- The first normal form helps to eliminate duplicate data and simplify queries.

Normalization of DBMS – 2NF

- 2NF eliminates redundant data by requiring that each non-key attribute be dependent on the primary key.

- This means that each column should be directly related to the primary key, and not to other columns.

Normalization of DBMS – 3NF

- 3NF builds on 2NF by requiring that all non-key attributes are independent of each other.
- This means that each column should be directly related to the primary key, and not to any other columns in the same table.

Boyce-Codd Normal Form (BCNF):

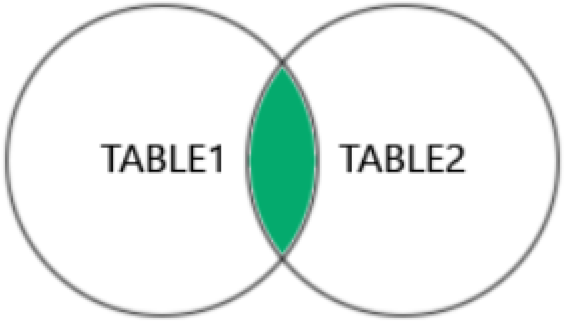
- BCNF is a stricter form of 3NF that ensures that each determinant in a table is a candidate key.
- In other words, BCNF ensures that each non-key attribute is dependent only on the candidate key.

SQL JOIN

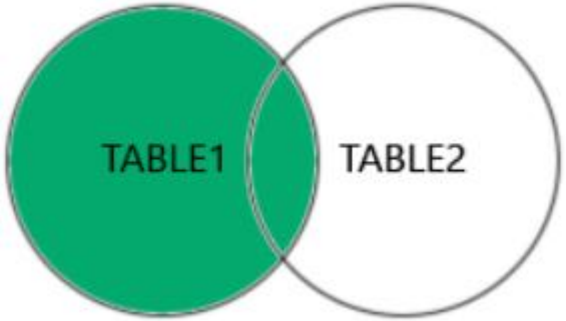
- SQL Join statement is used to combine data or rows from two or more tables based on a common field between them.
- Here are the different types of the JOINs in SQL:
  - **(INNER) JOIN**: Returns records that have matching values in both tables
  - **LEFT (OUTER) JOIN**: Returns all records from the left table, and the matched records from the right table
  - **RIGHT (OUTER) JOIN**: Returns all records from the right table, and the matched records from the left table
  - **FULL (OUTER) JOIN**: Returns all records when there is a match in either left or right table

SQL JOIN

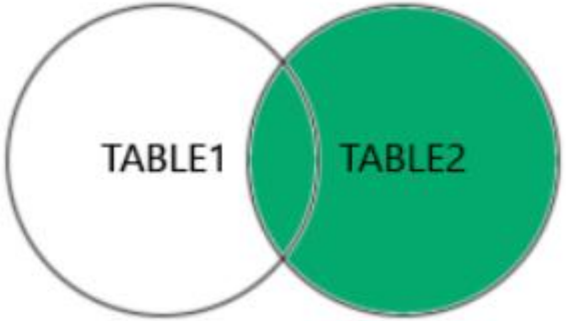
INNER JOIN



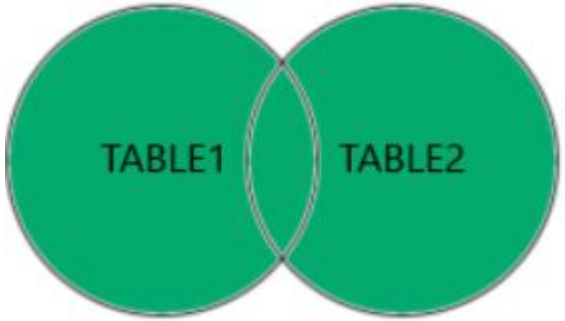
LEFT JOIN



RIGHT JOIN

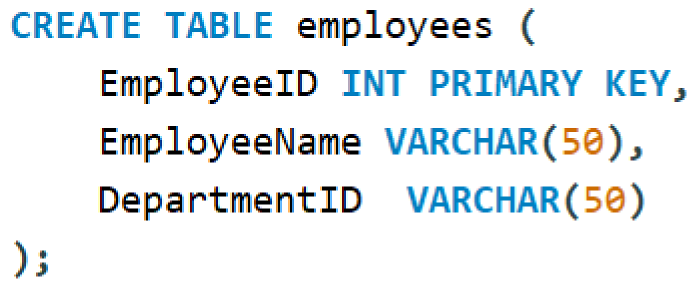


FULL OUTER JOIN

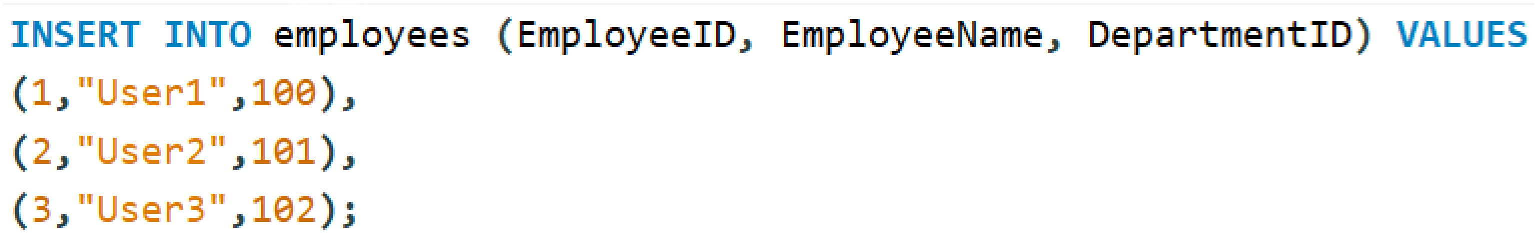


Inner Joins

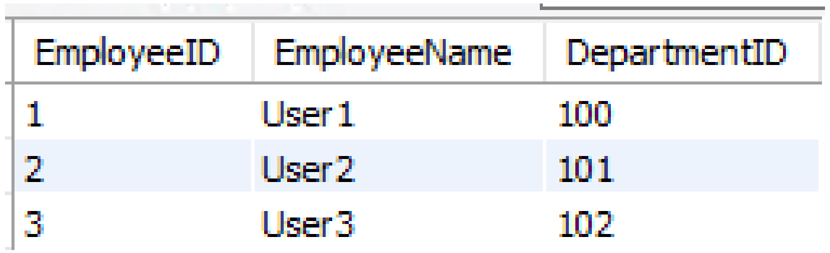
Table 1:



```
CREATE TABLE employees (
    EmployeeID INT PRIMARY KEY,
    EmployeeName VARCHAR(50),
    DepartmentID  VARCHAR(50)
);
```



```
INSERT INTO employees (EmployeeID, EmployeeName, DepartmentID) VALUES
(1,"User1",100),
(2,"User2",101),
(3,"User3",102);
```



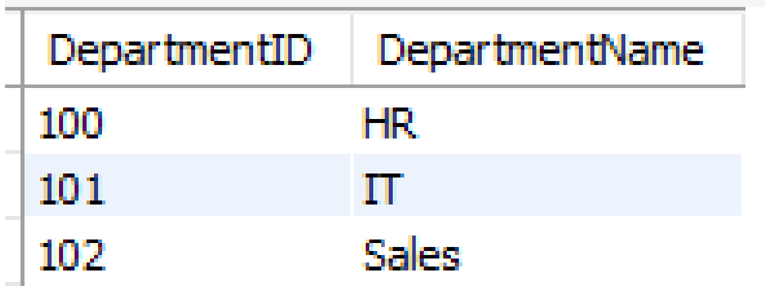
EmployeeID	EmployeeName	DepartmentID
1	User1	100
2	User2	101
3	User3	102

Inner Joins

Table 2:

```
CREATE TABLE departments (
    DepartmentID  VARCHAR(50),
    DepartmentName  VARCHAR(50)
);
```

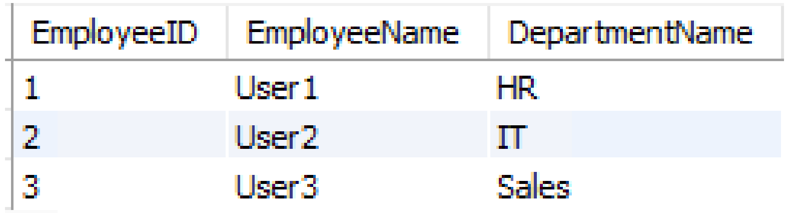
```
INSERT INTO departments (DepartmentID, DepartmentName) VALUES
(100,"HR"),
(101,"IT"),
(102,"Sales");
```



DepartmentID	DepartmentName
100	HR
101	IT
102	Sales

Inner Joins

```sql
SELECT Employees.EmployeeID, Employees.EmployeeName, Departments.DepartmentName
FROM Employees
INNER JOIN Departments ON Employees.DepartmentID = Departments.DepartmentID;
```



EmployeeID	EmployeeName	DepartmentName
1	User1	HR
2	User2	IT
3	User3	Sales

That's all Folks!