WHAT IS DATABASE?

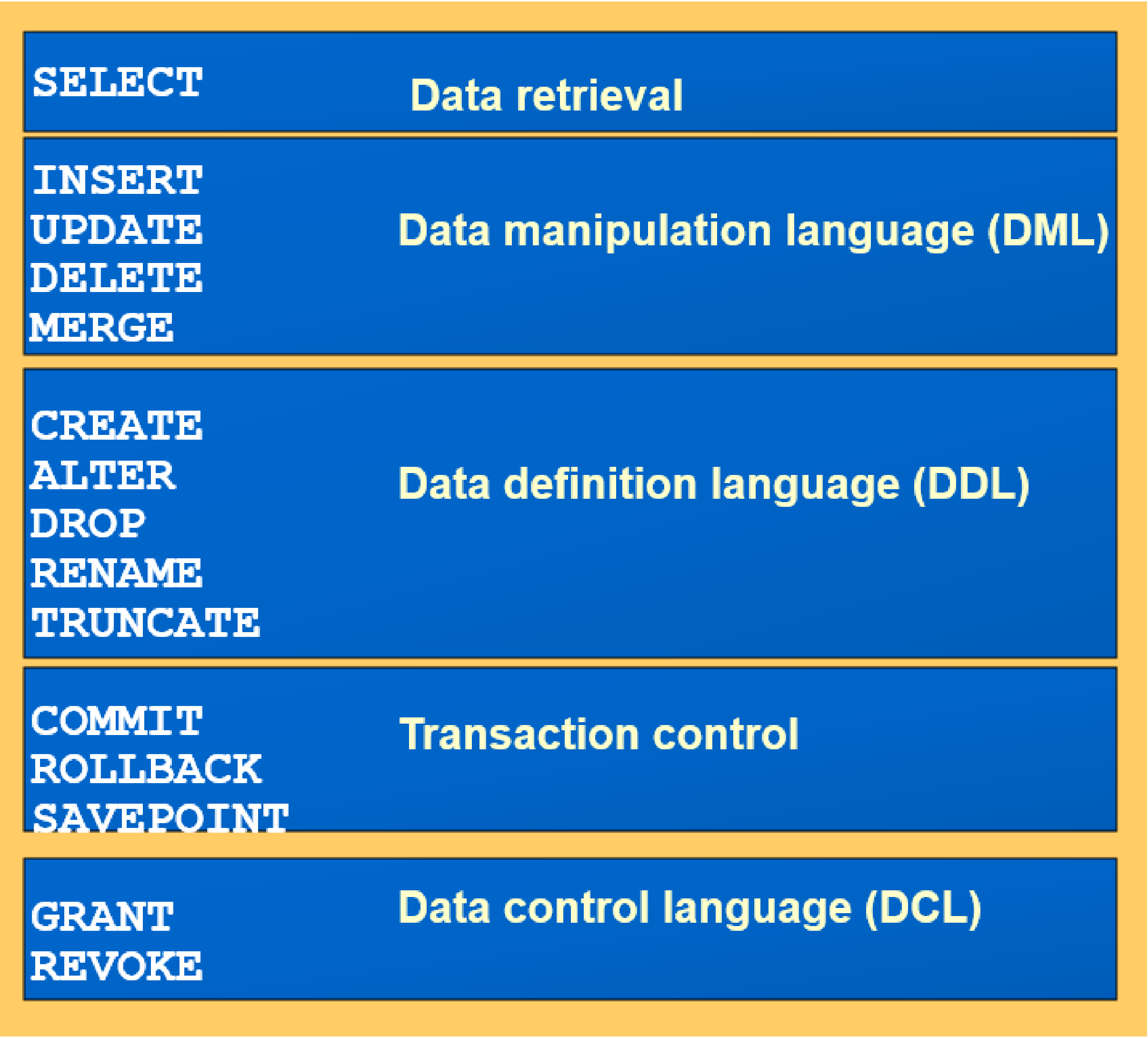
Database allows us to store data.

WHAT IS DBMS?

DBMS stands for Database Management System. DBMS facilitates not only storing of data but also provides a means to access the stored data.
DBMS is a set of programs that defines constraints,data types. Construct dealing with storage options and manipulate dealing with insert,update,delete.

SQL(Structured Query Language)

1. Commands not case sensitive.
2. Data not case sensitive.
3. String,Date Data Types should be enclosed in single or double quotes.
4. Accepted as Standard Language(ISO,ANSI) for Relational database
5. Non Procedural Language (what should be done/not how it should be done)
6. Interface used to access data from database



SELECT	Data retrieval

INSERT
UPDATE
DELETE
MERGE
	Data manipulation language (DML)

CREATE ALTER DROP RENAME TRUNCATE	Data definition language (DDL)
COMMIT ROLLBACK SAVEPOINT	Transaction control
GRANT REVOKE	Data control language (DCL)

Database Models

1. **Hierarchical :** Hierarchical structures were widely used in the early mainframe database management systems, such as the Information Management System (IMS) by IBM. This structure allows one 1:N relationship between two types of data.
2. **Network :** A network model in a database refers to a type of data model that represents the relationships between data entities using a graph structure. In this model, entities are represented as nodes and the relationships between them are represented as edges. This model is useful for representing complex relationships and is often used in applications such as social networks, recommendation systems, and fraud detection.

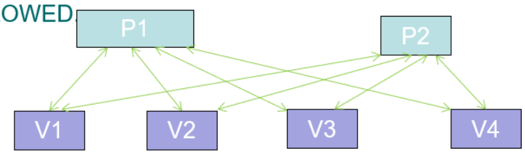
•A SYSTEM THAT EMPLOYS A SERIES OF SOFTWARE POINTERS FROM ONE DATA ITEM TO ANOTHER.

•A *POINTER* IS A CODE THAT INDICATES A LOCATION IN A FILE.

•UNLIKE HIERARCHICAL STRUCTURES, NETWORK STRUCTURES ARE NOT RESTRICTED TO PATHS UP AND DOWN.

•TEND TO HAVE LESS REDUNDANT DATA STORAGE.

•MULTIPLE PARENTS ARE ALLOWED.



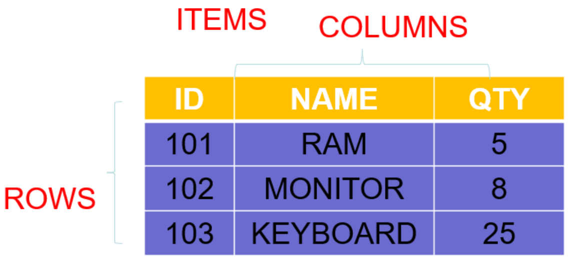
DISADVANTAGES:

DIFFICULT TO MODIFY

MORE EXTENSIVE LINKAGE INFORMATION MUST BE STORED.

3. **Relational :** The relational model consists of the following:

- Collection of objects or relations
- Set of operators to act on the relations
- Data integrity for accuracy and consistency


ITEMS
COLUMNS
ROWS

ID	NAME	QTY
101	RAM	5
102	MONITOR	8
103	KEYBOARD	25

- Table is a two dimensional matrix consisting of rows and columns. In order to work with tables one should be knowing the structure of the table.

Getting Started

CHECKING AVAILABLE DATABASES

After getting Mysql prompt, to begin with check the databases available using the command

```
mysql> show databases;
+--------------------+
| Database |
+--------------------+
| information_schema |
| mysql |
----------------------
```

CREATE DATABASE

To create a database, we use the CREATE DATABASE statement as shown in the following syntax:

CREATE DATABASE ***databasename;***

eg: Create Database Test;

USE CREATED DATABASE

A DBMS can manage multiple databases on one computer. Therefore, we need to select the database that we want to use. Write the following SQL statement for using the database:

```
mysql> use Test;
```

SEE TABLES AVAILABLE

Having selected the database to use, next we need to check tables available. Command used is SHOW TABLES

```
mysql> show tables;
```

WHAT IS A TABLE?

Table is a two dimensional matrix consisting of rows and columns. In order to work with tables one should be knowing the structure of the table. Command that can be used is

```
mysql> show columns from dept;
mysql> desc dept;
+-------------+---------------+------+-------+----------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------------+---------------+------+-------+----------+-------+
| DEPTNO | int | NO | PRI | NULL | |
| DNAME | varchar(14) | YES | | NULL | |
| LOC | varchar(13) | YES | | NULL | |
+-------------+----------------+------+-------+----------+-------+
```

CREATE TABLE COMMAND

```
--Showing CREATE TABLE command used for `users`
SHOW CREATE TABLE `users`;
```

DATABASE METADATA

To see the default character set and collation for a given database, use these statements:

```
SELECT @@character_set_database, @@collation_database;
```

Alternatively, to display the values without changing the default database:

```
SELECT DEFAULT_CHARACTER_SET_NAME, DEFAULT_COLLATION_NAME
FROM INFORMATION_SCHEMA.SCHEMATA WHERE SCHEMA_NAME = 'db_name';
```

A character set in a database is a set of characters that are used to store and represent data in the database. It is used to define the encoding of text data stored in the database and determines how the characters are stored and displayed. Examples of character sets include UTF-8, UTF-16, and ISO-8859-1.

In a database, **collation** refers to the set of rules used to compare and sort characters in a character set. This includes rules for letter case, accent marks, and other diacritical marks, as well as the specific language or locale used. Collation is important for ensuring that data is sorted and compared correctly.

Here are some examples of collation values that can be used in MySQL:

- utf8_general_ci: A case-insensitive, accent-insensitive collation for UTF-8 encoded data
- utf8mb4_unicode_ci: A case-insensitive, accent-insensitive collation for UTF-8 encoded data
- latin1_swedish_ci: A case-insensitive, accent-insensitive collation for ISO-8859-1 encoded data
- latin1_general_cs: A case-sensitive, accent-sensitive collation for ISO-8859-1 encoded data
- utf8_unicode_520_ci: A case-insensitive, accent-insensitive collation for UTF-8 encoded data

DATA TYPES IN PYTHON

String Data Types

- **CHAR(size)** A FIXED length string (can contain letters, numbers, and special characters). The size parameter specifies the column length in characters - can be from 0 to 255. Default is 1
- **VARCHAR(size)** A VARIABLE length string (can contain letters, numbers, and special characters). The size parameter specifies the maximum column length in characters - can be from 0 to 65535
- **BINARY(size)** Equal to CHAR(), but stores binary byte strings. The size parameter specifies the column length in bytes. Default is 1
- **VARBINARY(size)** Equal to VARCHAR(), but stores binary byte strings. The size parameter specifies the maximum column length in bytes.
- **TINYBLOB** For BLOBs (Binary Large OBjects). Max length: 255 bytes
- **TINYTEXT** Holds a string with a maximum length of 255 characters
- **TEXT(size)** Holds a string with a maximum length of 65,535 bytes
- **BLOB(size)** For BLOBs (Binary Large OBjects). Holds up to 65,535 bytes of data
- **MEDIUMTEXT** Holds a string with a maximum length of 16,777,215 characters
- **MEDIUMBLOB** For BLOBs (Binary Large OBjects). Holds up to 16,777,215 bytes of data
- **LONGTEXT** Holds a string with a maximum length of 4,294,967,295 characters
- **LONGBLOB** For BLOBs (Binary Large OBjects). Holds up to 4,294,967,295 bytes of data
- **ENUM(val1, val2, val3, ...)** A string object that can have only one value, chosen from a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted. The values are sorted in the order you enter them
- **SET(val1, val2, val3, ...)** A string object that can have 0 or more values, chosen from a list of possible values. You can list up to 64 values in a SET list

Numeric Data Types

- **BIT(size)** A bit-value type. The number of bits per value is specified in size. The size parameter can hold a value from 1 to 64. The default value for size is 1.

- **TINYINT(size)** A very small integer. Signed range is from -128 to 127. Unsigned range is from 0 to 255. The size parameter specifies the maximum display width (which is 255)
- **BOOL** Zero is considered as false, nonzero values are considered as true.
- **BOOLEAN** Equal to BOOL
- **SMALLINT(size)** A small integer. Signed range is from -32768 to 32767. Unsigned range is from 0 to 65535. The size parameter specifies the maximum display width (which is 255)
- **MEDIUMINT(size)** A medium integer. Signed range is from -8388608 to 8388607. Unsigned range is from 0 to 16777215. The size parameter specifies the maximum display width (which is 255)
- **INT(size)** A medium integer. Signed range is from -2147483648 to 2147483647. Unsigned range is from 0 to 4294967295. The size parameter specifies the maximum display width (which is 255)
- **INTEGER(size)** Equal to INT(size)
- **BIGINT(size)** A large integer. Signed range is from -9223372036854775808 to 9223372036854775807. Unsigned range is from 0 to 18446744073709551615. The size parameter specifies the maximum display width (which is 255)
- **FLOAT(size, d)** A floating point number. The total number of digits is specified in size. The number of digits after the decimal point is specified in the d parameter. This syntax is deprecated in MySQL 8.0.17, and it will be removed in future MySQL versions
- **FLOAT(p)** A floating point number. MySQL uses the p value to determine whether to use FLOAT or DOUBLE for the resulting data type. If p is from 0 to 24, the data type becomes FLOAT(). If p is from 25 to 53, the data type becomes DOUBLE()
- **DOUBLE(size, d)** A normal-size floating point number. The total number of digits is specified in size. The number of digits after the decimal point is specified in the d parameter
- **DOUBLE PRECISION(size, d)**
- **DECIMAL(size, d)** An exact fixed-point number. The total number of digits is specified in size. The number of digits after the decimal point is specified in the d parameter. The maximum number for size is 65. The maximum number for d is 30. The default value for size is 10. The default value for d is 0.
- **DEC(size, d)** Equal to DECIMAL(size,d)

Date and Time Data Types

- **DATE** A date. Format: YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31'
- **DATETIME(fsp)**A date and time combination. Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'. Adding DEFAULT and ON UPDATE in the column definition to get automatic initialization and updating to the current date and time
- **TIMESTAMP(fsp)** A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC. Automatic initialization and updating to the current date and time can be specified using DEFAULT CURRENT_TIMESTAMP and ON UPDATE CURRENT_TIMESTAMP in the column definition
- **TIME(fsp)** A time. Format: hh:mm:ss. The supported range is from '-838:59:59' to '838:59:59'
- **YEAR** A year in four-digit format. Values allowed in four-digit format: 1901 to 2155, and 0000.
- MySQL 8.0 does not support year in two-digit format.