Databases : Python such as SQLite3 MySQL MongoDB

# DATABASE: A database is information that is set up for easy access, management and updating.

# DBMS : a computerized data-keeping system

Types:   
Relational database management systems

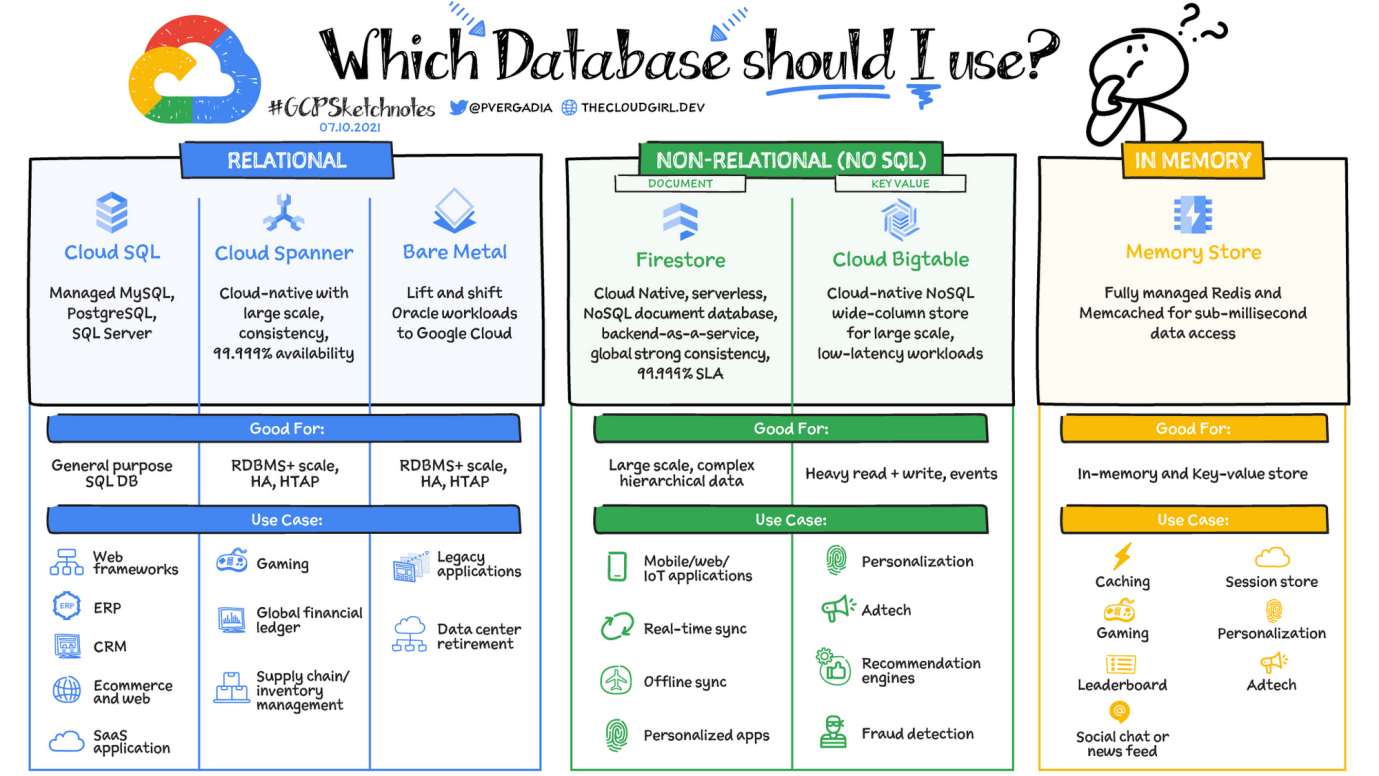
?hierarchical database systems

?network database systems

?object-oriented database systems

**Difference between Database and DBMS:**

| S. No. | Category | Database | DBMS |
| --- | --- | --- | --- |
| 1. | **Definition** | A database is a collection of connected information about people, locations, or things. | A database management system (DBMS) is a collection of programs that allow you to create, manage, and operate a database. |
| 2. | **Storage** | Besides computers, databases can even be maintained in physical ledgers, books, or papers. | In a database management system (DBMS), all the records are maintained only on a computer. |
| 3. | **Data Retrieval** | The retrieval of information from the databases can be done manually, through queries, or by using programs (C, C++, Java, etc.). | We can retrieve the data from the database management system through queries written in SQL. |
| 4. | **Speed** | As databases can be handled manually or via computers, when SQL is not used to retrieve information, it can be very slow. | As a computer system is involved in a database management system, the retrieval of information is very quick. |
| 5. | **Access** | The databases are not designed for a large number of people who can access data at the same time, rather it is designed for a very small number of people (preferably few people) who access data at different times. | The database management system is designed for a large number of people who can access the data at the same time. |
| 6. | **Data** | Data is stored in databases. | A database management system (DBMS) manages and manipulates data. |
| 7. | **Data Manipulation** | In the case of the databases, very less information can be modified at a time. | In the database management system (DBMS), a lot of information can be changed at one time (as it can have many users using it at the same time). |
| 8. | **Backup and Recovery** | The databases do not ensure that the data will be available after failure arises. | The database management system (DBMS) ensures that the data will always be available even after system failures. |



RDBMS: From E.F.Codd’s A relational model of data for large shared data banks, RDBMS came into existence.

?Rules of E F Codd:  
Rule 0 − Foundation rule

Any relational database management system that is propounded to be RDBMS or advocated to be a RDBMS should be able to manage the stored data in its entirety through its relational capabilities.

Rule 1 − Rule of Information

Relational Databases should store the data in the form of relations. Tables are relations in Relational Database Management Systems. Be it any user defined data or meta-data, it is important to store the value as an entity in the table cells.

Rule 2 − Rule of Guaranteed Access

The use of pointers to access data logically is strictly forbidden. Every data entity which is atomic in nature should be accessed logically by using a right combination of the name of table, primary key represented by a specific row value and column name represented by attribute value.

Rule 3 − Rule of Systematic Null Value Support

Null values are completely supported in relational databases. They should be uniformly considered as ‘missing information’. Null values are independent of any data type. They should not be mistaken for blanks or zeroes or empty strings. Null values can also be interpreted as ‘inapplicable data’ or ‘unknown information.’

Rule 4 − Rule of Active and online relational Catalog

In the Database Management Systems lexicon, ‘metadata’ is the data about the database or the data about the data. The active online catalog that stores the metadata is called ‘Data dictionary’. The so called data dictionary is accessible only by authored users who have the required privileges and the query languages used for accessing the database should be used for accessing the data of data dictionary.

Rule 5 − Rule of Comprehensive Data Sub-language

A single robust language should be able to define integrity constraints, views, data manipulations, transactions and authorizations. If the database allows access to the aforementioned ones, it is violating this rule.

Rule 6 − Rule of Updating Views

Views should reflect the updates of their respective base tables and vice versa. A view is a logical table which shows restricted data. Views generally make the data readable but not modifiable. Views help in data abstraction.

Rule 7 − Rule of Set level insertion, update and deletion

A single operation should be sufficient to retrieve, insert, update and delete the data.

Rule 8 − Rule of Physical Data Independence

Batch and end user operations are logically separated from physical storage and respective access methods.

Rule 9 − Rule of Logical Data Independence

Batch and end users can change the database schema without having to recreate it or recreate the applications built upon it.

Rule 10 − Rule of Integrity Independence

Integrity constraints should be available and stored as metadata in data dictionary and not in the application programs.

Rule 11 − Rule of Distribution Independence

The Data Manipulation Language of the relational system should not be concerned about the physical data storage and no alterations should be required if the physical data is centralized or distributed.

Rule 12 − Rule of Non Subversion

Any row should obey the security and integrity constraints imposed. No special privileges are applicable.

# SQL – Structured Query Language: Providing Queries to a Database in a structured format called SQL

The Language Used in an RDBMS, but not Co-dependent.

**QUERY** - a question, especially one expressing doubt or requesting information.

?**SCHEMA** - Schema is the overall description of the database. The basic structure of how the data will be stored in the database is called schema. Schema is of three types: Logical Schema, Physical Schema and view Schema.

**?ER DIAGRAMS** – An Entity Relationship Model Diagram is used to conceptualize a database, it consists of Entities, Attributes and Relationships

**TABLE:** Tables are database objects that contain the data in a database. In tables, data is logically organized in a row-and-column format similar to a spreadsheet. Each **row** represents a unique record or tuples, and each **column** represents a field or attribute in the record

**CELL:** a part of a table where a row and column intersect. A cell is designed to hold a specified portion of the data within a record.

**?NORMALIZATION:** Normalization organizes the columns and tables of a database to ensure that database integrity constraints properly execute their dependencies. It is a systematic technique of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update, and Deletion anomalies.

1NF

2NF

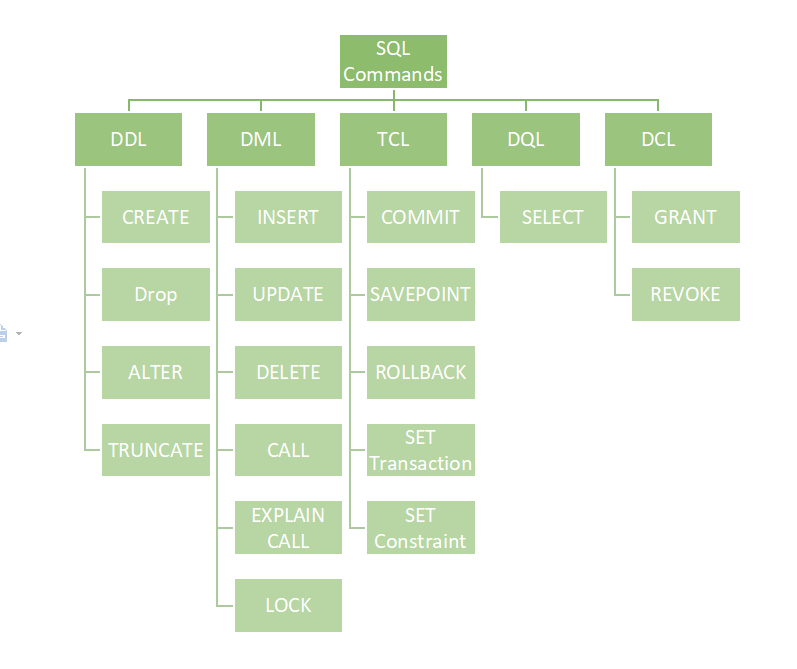
3NF

BOYCE CODD NORMAL FORM … 4NF

**SQL STATEMENT**: A SQL statement is a set of instruction that consists of identifiers, parameters, variables, names, data types, and SQL reserved words that compiles successfully.

**SQL EXPRESSION:** SQL expression is a combination of one or more values, operators and SQL functions that results in to a value.

**SQL COMMANDS:** The instructions used to communicate with a database to perform tasks, functions, and queries with data.



**?DDL**( **DATA DEFINITION LANGUAGE** ):Create, alter, and drop schema objects. Grant and revoke privileges and roles. Analyse information on a table, index, or cluster. Establish auditing options. Add comments to the data dictionary.

**?CREATE**

**?ALIAS AS**

**DESCRIBE table\_name;** Shows Table Schema.

**DATA TYPES(Memory Representation): Mandatory Information** that must be given upon creation of a table, for the attribute or field.  
some data types have arguments that can be passed inside them as parameters.

There are categorized by strings, numeric and date-time,

For Strings, some common data types are CHAR(n), VARCHAR(n)/VARCHAR2(n), TEXT – in Mysql, etc

For Numerics, some common data types are INT(n), FLOAT(n), etc

For Date-Time, some common data types are DATE, YEAR, DATETIME(fsp)

**QUALITY OF LIFE RULES:**

* It is always better to use lowercase, numbers and underscores when writing or naming, since SQL non case-sensitive, it will be better if the column names don’t use uppercase, as names are case-sensitive. So avoid Uppercase, Spaces and Special Case.
* Column names can be descriptive and with a good readability.
* Names should never use reserve words or keywords (around 204).
* Use Plurals for Table names.
* For Primary keys, stick to: id int primary key, Whenever possible,  
  and for Foreign keys , refer table name and put id: emp\_id.
* UTC for dates

**?CONSTRAINTS:**

**?NOT NULL**

**?UNIQUE**

**?PRIMARY KEY**

**?FOREIGN KEY**

**?CHECK**

**?DEFAULT**

?Column Level Constraint

?Table level Constraint

**INDEX: CREATE INDEX**

**?DROP**

**?ALTER**

**?TRUNCATE**

**?COMMENT**

**?KEYS:**

**Primary keys**

**?Candidate keys**

**Foreign keys**

**?Composite keys**

**?Unique keys**

**?Alternate keys**

?DML (DATA MANIPULATION LANGUAGE): commands permitting users to manipulate data in a database. This manipulation involves inserting data into database tables, retrieving existing data, deleting data from existing tables and modifying existing data.

INSERT INTO \_table\_ VALUES (data)

UPDATE \_table\_ SET \_att\_=\_data\_ WHERE \_cond\_

DELETE \_record\_ FROM \_table\_ WHERE \_cond\_

IN order to work with any existing data, we have to understand how to fetch the existing data and understand about the DQL Command, SELECT and the Clauses.

?DQL (DATA QUERY LANGUAGE): The commands of SQL that are used to retrieve data from the database are collectively called DQL. So all Select statements come under DQL. The purpose of DQL Command is to get some schema relation based on the query passed to it.

**SET LINES 200;  
SET PAGES 200;**

**SELECT column\_names FROM table\_name :**  Select fetches and returns the data from the table DEPENDING UPON THE CLAUSES.

?The data returned is stored in a result table, called the result-set.

**Splat Operator/Asterisk/ Star - SELECT \* -** Select everything, and -tablename dot-notation asterisk- table\_name.\*

**?ALIAS AS:** Alternate name

**?DISTINCT – REMOVE DUPLICATES**

**SELECT & CLAUSES**: Below are the clauses of the Select command, listed in order of the syntax.

**FROM**- Suffixed by a table name, Gets the raw data from the schema entity.

The concept of getting raw data from multiple tables called ‘join’ is implemented here.

**WHERE**- The SQL WHERE clause is used to specify a condition while fetching the data from a single table or by joining with multiple tables.

**OPERATORS:** a reserved word or a character that is used to query our database in a SQL expression.

Arithmetic Operators: + Addition, - Subtraction, \* Multiplication, / Division, % Modulo.

Comparison/Relational Operator: (returns UNKOWN for one or more NULL)  
= Equal to, != Not Equal to, > Greater than, < Lesser than, >= Greater than or equal to, <= Lesser than or equal to.

Concatenation Operator: ||

Union Operator: UNION

Logical Operators: AND, OR, NOT.

Special Operators: IN, NOT IN, BETWEEN, NOT BETWEEN, IS, IS NOT, LIKE, NOT LIKE, ANY, ALL, SOME, EXISTS, NOT EXISTS.  
The Special Operator LIKE goes along with % percentages and \_ underscores, % can hold in multiple data, whereas \_ holds a single character.

**Functions: Builtin and Defined, Multi Row Functions (MRF) and Single Row functions (SRF)**

**MRF(AGGREGATE FUNCTIONS):** COUNT(), MAX(), MIN(), SUM(), AVERAGE()

**GROUP BY**- (used along with mrf or the column\_name specified)

**HAVING**-

**SRF(SINGLE ROW FUNCTIONS):** MOD(), ROUND(), TRUNC(), LENGTH(), UPPER(), LOWER(), INITCAP(), REVERSE(), SUBSTR(), INSTR(), REPLACE(), TO\_CHAR(), TO\_DATA(), NVL() ….. etc

**ORDER BY**-

**ORDER OF EXECUTION OF CLAUSES:**

**From-** choose & join tables to get base data **where-** Filter base data  
**group by-** Aggregate base data  
**having-** filter aggregate  
**select-** return/about to return final data  
**order by-** sort final data  
**limit-** limits the returned data to row count

Tcl

Joins

Co-related subqueries

Basic for python with built-in SQLite3

Relational database management  
doesn’t require server and can be embedded

**Connect  
Establish a connection object in order to establish a connection to a database**

Connect to a database: creates if it doesn’t exist, connects if it exists using connect method

Create connection object:   
connection\_object=sqlite3.connect(‘datab\_name.db’)

We need a cursor, interact objects to database  
cursor\_name=connection\_object.cursor()

FOREIGN

SQL: Table – rows and columns

///execute is a method used to query or statement  
///Query is in uppercase for readability and is case-Insensitive  
///always mention the data type of the data to be given inside the cell, String is TEXT, VARCHAR(len) in SQL

cursor\_name.execute(“CREATE TABLE table\_name (colum\_name1 TEXT, column\_name2 TEXT, column\_name3 INTEGER)”)

/// a 3 column table

connection\_name.commit() ///saves

connection\_name.close()

#$#$#$#$

cursor\_name.execute(“CREATE TABLE IF NOT EXISTS table\_name (colum\_name1 TEXT, column\_name2 TEXT, column\_name3 INTEGER)”)

/// IF NOT EXISTS – used when error return : table already exists , occurs if program runs more than once

#$#$#$#$#$#$

/// insert 3 rows of data inside the empty table

cursor\_name.execute(“INSERT INTO table\_name VALUES (‘column1data1’,’column2data1’,column3data1), (‘column1data2’,’column2data2’,column3data2), (‘column1data3’,’column2data3’,column3data3)”)

/// single quotes is used to define strings since the queries are passed within double quotes and viceversa

#$#$#$#$#$#$#

////Primary key or unique id for each data and is Auto incrementing

cursor\_name.execute(“CREATE TABLE table\_name (id\_name INTEGER PRIMARY KEY, colum\_name1 TEXT, column\_name2 TEXT, column\_name3 INTEGER)”)

cursor\_name.execute(“CREATE TABLE sub\_table (sub\_col\_id INTEGER PRIMARY KEY, id\_of\_another\_table INTEGER, sub\_table\_data TEXT, FOREIGN KEY (id\_of\_another\_table) REFERENCES table\_name(id\_name))”)

////// Reference Primary key of other table inside this new table

///cursor\_name.execute(“INSERT INTO table\_name VALUES (1,’marc’,’specter’,002)”)

///but since primary key is auto incrementing we can just define the other three columns by name

cursor\_name.execute(“INSERT INTO table\_name (colum\_name1, column\_name2 , column\_name3) VALUES (’marc’,’specter’,002)”)

#$#$#$#$#$#

cursor\_name.execute(“UPDATE table\_name SET column\_name1 = ‘jack’ where column\_name1= ‘marc’ “)

//// Update information in table

#$#$#$#$#$#$

///To read the data rows with data\_to\_match in column 2

cursor\_name.execute(“SELECT \* FROM table\_name WHERE column2=’data’ “)

/// \* every column , WHERE – condition

/// cursor\_name.execute(“SELECT column\_name1 FROM table\_name “)

/// for specific columns

/// to get them in specific order  
/// cursor\_name.execute(“SELECT \* FROM table\_name ORDER BY column\_name2 DESC “)

///DESC – descending order, ASC – ascending order

/// cursor\_name.execute(“SELECT \* FROM table\_name ORDER BY column\_name2 DESC LIMIT 1“)

/// returns only 1 data by limiting

rows\_variable= cursor\_name.fetchall()

/// fetchall() method in sql

print(rows\_variable)

#$#$#$#$#$

Combine OOP and databases

DATACLASSES:

Data classes ; data focussed class ; database class

Functionality : classifier classes, regression classes, etc

Data clas

#$#$#$#$#$

//// GROUP BY

cursor\_name.execute(“SELECT column\_name1 FROM table\_name GROUP BY column\_name1 “)

#$#$#$#$#$

“SELECT AVG(data) --- returns average of the numerical data

#$#$#$#$#$

Leftover queries

#$#$#$#$#$

Database operations with different data types  
#$#$#$##$

Data types and formats allowed by sqlite3

#$#$#$#$

DDL and DML

#$#$#$#$

Python syntax for sqlite3  
#$#$#$#$

cursor.executemany()

#$#$#$#$

Values(?,?,?)  
tuple with data – multiple tuples in a sequence

#$#$#$#$

CLOUD COMPUTING:

Serverless Computing/ Cloud computing- servers in cloud :

Aws lambda, google cloud functions

Os network patch – cloud, user – only code,

Serverless monotlithivc

iot and notifications or emails.

Firebase cloud functions , nodejs