Data is Fact or Figure is about anything (related to pensonal data, or online purchasing). Data gets collected and stored in databases from various sources for various reasons. The way data is related and presented enables people to form a better understanding of existing data.

Database(dB) is a form of electronic storage in which data is organized systematically, to make it more manageable, efficient and secure. Example bank can use a database to store their data of customers, bank accounts, and transactions. In a hospital to store the patient data , staff data, lab data and much more. A Database looks like data organized systematically looks like table or spreadsheet.

Systematically means all :

* data contains identifiable features or attributes (Person can be identified by age, or height)
* This data separated and stores much knowns as entities (represent Elements, contains rows and columns that store data relating to the specific elements). This entities could be physical representation (employee and customers) or conceptual representation (orders and invoice or quotation) . Entity then store data in a table like format against the attributes / features related to the element.

In a Relational database, this entities known by relation or tables

The attributes became Columns of the table. Each table rows represent instances of the entities.

Fields are head of columns (attributes), and existing information named record of the table. All this field and rows are working together to store information also known as entities. Every row and recore of customer table are instance of the customer entities.

A screenshot of a computer

Description automatically generated

* Customer ID – FirstName – LastName – Email : is a Field
* Customer ID or FirstName or LastName or Email is attributes.
* C1 – sarah hogan until C5 – Mish Taleb : is a Record.
* C1 – Sarah hogan is customer instances.
* Customer Table muss be relation with others table, with unique instance for example CustomerID related within 2 others table. CustomerID called Primary Key.

Different types of database :

1. Object oriented database – data stored in the form of the objects instead of table or relation (example online book store).
2. Graph databases – data stored in the form of nodes. Entities example customers, order and product represent as a node. The relationship between them are represented by edges.
3. Document database – data stored as JSON objects.

Where the databases saved, could be in dedicated machine on a premises organization, or could be in a cloud hosting. Cloud hosting become more popular because of can be access the data anywhere via internet, and low cost option.

**Record or instances must be uniquely identifiable : Primary key**

**Could be alo described as a a Primary Key field contains unique values that cannot be replicated elsewhere in the table. This avoids potential confusion between tables with similarities in data.** So within unique id your data can be determined which data belongs to.

A screenshot of a computer

Description automatically generated

* OrderID as a primary key in Order table
* Field CustomerID , with EXACT same data as customer id in a Customer table. So between this 2 tables, related within CustomerID field. **CustomerID named as ForeignKey. Foreign key is a field in one table that connect to the primary key field in the original table. The foreign key is used to connect tables, not to identify a specific record of data in a relational database.**in this case is a customer table.
* CustomerID in customer table as a primary key, but in a order table be a foreign key. (RELATIONS)

A screenshot of a table

Description automatically generated

Relational data example charts (a picture is worth a thousand words) :

1. **Bar Chart :** A bar chart is a graph **that presents categorical data with rectangular bars**, where the heights of the bars are proportional to the values that they represent.
2. **Bubble Chart :** A Bubble chart is another popular type of data chart. It shows **how different values compare to each** other in terms of bubble size. The smaller bubbles represent smaller values, and the larger bubbles represent larger values.
3. **Line Chart :** A line chart presents information as **a series of data points called “markers”** connected by straight line segments. Line charts are extremely popular and are widely used in most data analytics fields.Line chart are best used to identify **trends that help predict the future.**
4. **Pie Chart :** A pie chart is another type of data chart **that displays how various data make up a whole of 100 percent**. In this type of chart, each data point is allocated a "slice" of the pie according to its value. Pie charts are a simple way to show how various parts create a whole.
5. Heat maps, scatter plot, etc….so many : Some charts can serve multiple purposes, whereas others are much better at conveying specific types of

Relational databases have a limitations of when it’s storing data , mostly using structured data using tabular format . Traned nowadays a database requires more and more unstructured data. **NoSQL databases a types of a database that store data in a variety of different format. Uses in a social media platform , IoT , AI. Types of NoSQL databases : Document databases, key-value databases, and Graph databases.**

Big Data is Complex data that can increase exponentially with time. Complex from social media platform, online shopping sites, and others platform that can generate massive data. Within IoT more and more devices connect to internet, and generating more and more data.This how big data are created.

Traditional databases system could deal with data using tables, record and relationships. BIG Data is the new challenges.

Big data is :

* combination of structured, semi structured and unstructured data collecting from many sources
* Adds more power of data because can address more complex business problems than tradional data can handle
* Big data can provides uniques insights to help improve decision making

Traditional database means store data, but Organizations needs a larger data to support business. **Within Business Intelligence technologies ,organization can analyze the data and extract valuable information to help them to make informed decisions.**

**CRUD Operations in SQL : Create, Read, Update, Delete .** It’s a common task if you are working with database.

* **Create** Table => **Insert** the data into this tables => **Modify** the data if something changing

**Database Management System (DbMS) : change SQL instruction into a form understood by the database. Takes responsibility for transforming SQL instruction into form that understood by underlying database.**

SQL Subsets :

1. DDL (Data definition Language) – Create, Alter, Drop

DDL **Create** Command used to **create** storage objects in a database, like tables. DDL also is how to find your data in database. DDL **Alter** command used **to modify** the structure of a table object in a database. DDL **Drop** command to **remove an exiting object** from a database.

2. DML (Data manipulation Language) – Insert, Update, Delete

DML **Insert** command **to insert records of data** into a database table. DML **Update** is used **to edit data that already exits** in a database table. DML **Delete** command is **to delete or remove one or more rows of data** from a table.

3. DQL (Data Query Language) – SELECT

DQL is To Read a data in a database. Use **Select to retrieve data** from one or multiple tables based on filter criteria.

4, DCL (Data Control Language)

DCL is to control access to database. Use **Revoke access limited to user.**

Advantages of SQL : (SQL is interface between relational database and users)

1. **Simple/**User friendly – requires very little coding skill to use

2. **Standar** language that can be used with all relational databases.

3. **Portable** language that can be used on any hardware and any operating system platform.

4. **Efficient** Data processing – SQL processes large amounts of data quickly and efficiently.

CREATE DATABASE database\_name; ie. Database COLLEGE #then create the tables.

CREATE TABLE table\_name; ie. Table STUDENT #then insert the data

**Add data to a table**

INSERT INTO table\_name (column\_one, column\_two, column\_three…) VALUES (value1, value2, value3, …)

(INSERT INTO Student (column\_ID, first\_name, last\_name, date\_of\_birth) VALUES (value1, value2, value3,values4);

A table with numbers and text

Description automatically generated A table with a number of students

Description automatically generated with medium confidence

1. UPDATE Data in a table (STUDENT)

UPDATE Student SET date\_of\_birth = ‘2000-10-12’ WHERE ID = 02; #WHERE is condition

2. DELETE data from a table (STUDENT)

DELETE FROM Student WHERE ID = 03; #WHERE is condition

3. Query data within a table

SELECT first\_name, last\_name FROM Student WHERE ID = 01;

1. Data Definition Language (DDL) - provides commands for defining, deleting and modifying tables in a databas

* CREATE Command Purpose: To create the database or tables inside the database

CREATE TABLE table\_name (column\_name1 datatype(size), column\_name2 datatype(size), column\_name3 datatype(size));

CREATE TABLE table\_name (column\_name1 datatype(size), column\_name2 datatype(size), column\_name3 datatype(size));

* DROP Command Purpose: To delete a database or a table inside the database.

DROP TABLE table\_name;

* ALTER Command Purpose: To change the structure of the tables in the database such as changing the name of a table, adding a primary key to a table, or adding or deleting a column in a table.

ALTER TABLE table\_name ADD (column\_name datatype(size));

ALTER TABLE table\_name ADD primary key (column\_name);

* TRUNCATE Command Purpose: To remove all records from a table, which will empty the table but not delete the table itself.

TRUNCATE TABLE table\_name;

* COMMENT Command Purpose: To add comments to explain or document SQL statements by using double dash (--) at the start of the line. Any text after the double dash will not be executed as part of the SQL statement. These comments are not there to build the database. They are only for your own use.

--Retrieve all data from a table

SELECT \* FROM table\_name;

2. Data QUERY Language (DQL) - provide the ability to query and retrieve data from the database

* SELECT Command Purpose: To retrieve data from tables in the database.

SELECT \* FROM table\_name;

3. Data MANIPULATION Language (DML) - provide the ability to query, delete and update data in the database

* INSERT Command Purpose: To add records of data into an existing table.

INSERT INTO table\_name (column1, column2, column3) VALUES (value1, value2, value3);

* UPDATE Command Purpose: To modify or update data contained within a table in the database.

UPDATE table\_name SET column1 = value1, column2 = value2 WHERE condition;

* DELETE Command Purpose: To delete data from a table in the database.

DELETE FROM table\_name WHERE condition;

4. Data Control Language (DCL) - rights and permissions of users of a database system

GRANT Command to provide the user of the database with the privileges required to allow users to access and manipulate the database.

REVOKE Command to remove permissions from any user.

5. Transaction Control Language (TCL) - are used to manage transactions in the database

COMMIT Command to save all the work you have already done in the database.

ROLLBACK Command to restore a database to the last committed state.

A screenshot of a computer

Description automatically generated

TABLE are made by Rows and Columns which hold data. And table stored in a database. A database hold multiple tables. This table knows as relations (related one each other).

* **Table also known as entity known also as object**. Table is where the data is stored. It is responsible for storing data in the database. A database table also consists of rows and columns. A table contains all the fields, attributes and records for a type of entity.
* **Attributes which are details about the table or entity.** In other words, attributes describe the table.
* **Columns known as field**. Each column field has unique name and data type. Columns run vertically. They are like the definition of each field. Each field contains a different attribute.
* **Rows knows as Record known also a tuple.** Record are combinations of columns or field that contain data. A table row or a record is also known as a tuple.
* **Data type defines what type of value a table column can hold, and tell (guideline) SQL what data type to expect in each column.** ie. String or Character, Numeric, Date and Time, Binary, and miscellaneous data types.
* **Domain is setup of legal values that can be assign to an attribute.** **Basically this means making short a values of field can hold a well defined.**
* Each table or relation in a database has its own schema. **Schema simply means the structure.**
* In a table, there is a field or column that is known as a key which can uniquely identify a particular tuple (row) in a relation (table). **This key is specifically known as a primary key.** In some cases, the primary key can comprise more than one column or field, This is also known as a composite primary key. For example the EMP\_ID and DEPT\_ID columns together can make a record unique.
* A screenshot of a computer

  Description automatically generated
* Every table in a database should abide by rules or constraints. These are known as integrity constraints. There are three main integrity constraints:
  1. Key constraints

The key constraint specifies that there should be a column, or columns, in a table that can be used to fetch data for any row. This key attribute or primary key should never be NULL or the same for two different rows of data. This key attribute or primary key should never be NULL

* 1. Domain constraints

Domain constraints refer to the rules defined for the values that can be stored for a certain column. For instance, you cannot store the home address of a student in the first name column. Similarly, a contact number cannot exceed ten digits.

* 1. Referential integrity constraints

When a table is related to another table via a foreign key column, then the referenced column value must exist in the other table.

A diagram of a company

Description automatically generated with medium confidence

**Logical database structure**

**The logical structure of a database is represented using a diagram known as the Entity Relationship Diagram (ERD).** It is a visual representation of how the database will be implemented into tables during physical database design, using a Database Management System (DBMS) like MySQL or Oracle, for example.

A part of the logical database structure is how relationships are established between entities. These relationships are established between the instances of the entities. Accordingly, there can be three ways in which entity instances can be related to each other:

* One-to-one relationships
* One-to-many relationships
* Many-to-many relationships

This is also known as cardinality of relationships. The logical database structure which is represented using an ERD also depicts these relationships. Here’s an example of an ERD that has all these elements.

A diagram of a function

Description automatically generated with medium confidence

**Physical database structure**

In the physical database structure, where entities are implemented as tables, **the relationships are established using a field known as a foreign key.** A foreign key is a field in one table that refers to a common field in another table (usually the primary key).

Let’s take the example of a database that contains two tables: student and department. The student table has a primary key of “Stud\_id”, which is also present in the Department table as a foreign key. Therefore, the two tables are related to each other via the “Stud\_id” field.

A screenshot of a computer

Description automatically generated

A table with text on it

Description automatically generated

Each table has relevant columns, each column represents attributes of the table entities. Each table has Team Name column.

A table with text on it

Description automatically generatedMultiple values attributes should be avoided in the relational database (for example in Subjects has more 1 attributes). a simple attribute type is designed to hold one single value.

* Staff ID will be **Key attribute** : **used to uniquely identify an individual record of data.**
* Staff ID & Contact number will be a **Candidate Key attribute** : any attribute that contains a unique value in each row of the table.
* Combination of Staff name and staff title will be a **Composite key attribute** : A key composed of two or more attributes to form a nuqie value in each new row.
* Staff id is A **Primary key** : A selected candidate key. (contains unique values)
* Contact number can be a **Alternate key** : a candidate key not selected as the primary key.
* **A Foreign key** : An attribute that referenes a unique key in another table. Reference as a primary key used in another table.

QUESTION :

1. Each row of the table will have a record of information that refers to a specific staff.
2. The primary key includes unique values in each row and is used to identify each record of a table
3. What is the minimum number of tables that must be present in a relational database? 2 tables : Not quite. More tables can be considered if needed, however, you can have a database with only one table.
4. Which of the following keys can you select as the primary key in a relational database?
   1. A candidate key has unique values in each row of the table
   2. The Composite key is composed of multiple columns to form a unique value in each row of the table, which makes it suitable to be chosen as a primary key.
   3. Alternate key has unique values in each row of the table, which makes it suitable to be chosen as a primary key.
   4. The foreign key could have repeated values in different rows. Therefore, it can’t be used to uniquely identify each record in a table.
5. Relational databases are the most popular type of databases because of their simplicity, robustness and scalability.
6. Tables are a very good technique to organize data as it provides a simple and clear view of data in the database.
7. In a bookshop database, the complete information about one specific book is referred to as a **RECORD**
8. SQL Popular because :
   1. SQL can be used with different relational database management systems.
   2. SQL requires less coding skill in comparison with other programming languages.
   3. SQL can be used with any laptop or computer that has MySQL installed.
9. The UPDATE command is used to modify or update data contained within a table in the database.
10. following database management systems uses the SQL language?
    1. MySQL database management system uses SQL.
    2. PostgreSQL database management system uses SQL.
    3. Oracle database management system uses SQL.
11. What is the importance of **a candidate key in a database**? A candidate key is a column that contains unique value in **each row of the table**, which makes it suitable to be chosen as a primary key.
12. CREATE TABLE Student : This is the right way to create a student table in SQL.
13. SQL is the standard language for data manipulation and SQL is the standard language for relational database management.

OTHERS MATERIAL CHAPTER 1 .

1. <https://support.microsoft.com/en-us/office/database-design-basics-eb2159cf-1e30-401a-8084-bd4f9c9ca1f5>
2. https://www.ibm.com/docs/en/control-desk/7.6.0?topic=design-relational-database-structure

.----------------------------------------------------------------------------------------------------------------------------------------------------------

SQL are made up of descriptive words and are easy to learn or easy to interpreted. **SQL is a non-procedural language** that cannot write complete applications with it (only interact and communicate with data), and relatively simple but powerful.

SQL are used in 3 Things :

1. Read/Retrieve data : Data are often stored in a database.
2. Write Data : Add data to a table
3. Update Data : insert new data

Jobs who uses SQL :

|  |  |  |
| --- | --- | --- |
| Backend developer | Data Scientist | System Admin |
| ETL Developer | Data Analyst | Database Admin (DBA) |
| QA Engineer | Data Architect | System Engineer |

RDBMS example :

|  |  |  |
| --- | --- | --- |
| Microsoft SQL Server | PostgreSQL | SQLite |
| MySQL | IBM DB2 Oracle | Apache Open office Base |

Thinking before coding is important. And Important to understand how the data in a dabase relates to one another.

**THINK BEFORE YOU CODE. What is the problem you are trying to solve? -speedup!!**

* What is data that you need to get & How the data is related each other (data Interact)
* Understand the business process or subject matter (problem you trying to solve)
* Know the business rules
* Understand how your data is organized and structured in the table

Database : A Container (set of files) to store organized data ; a set of related information. (ie. Filing cabinet system)

Tables : A structured list of data or specific type (tables in filing cabinet system are a binder contains the files)

A Table are made by Column and Row.

Column : A single field in a table – all tables are made up of one or more columns

Row : A record in a table.

**Data Modeling** is what we used to organized information from multiple tables and how they related into each others together. Data model represent business process or to understand a business process.

**Data Models** : Tables are represented and organized in a database. It’s different between Model and Data model. Model term used in a data scientist. From 1960 as a Hierarchical Data models – **Relational Data Model** – Entity Data models – And Now popular **a NoSQL** as a results of BIG DATA movement.

**Not Only SQL (NoSQL)** : A mechanism for storage and retrieval of **unstructured data** modeled by means other than tabular relations in relational databases.

**RELATIONAL vs TRANSACTIONAL Model**

|  |  |
| --- | --- |
| Relational Model | Transactional Model |
| Database design that choose relationship between the different table , and use to optimize the query data . Make it easy and intuitive access to data. | More operational database – insurance claims within a healthcare database. (may not be stored in a query) |

Data Model Building Blocks for Relational Model :

1. Entity : Person, place thing or event which are distinguishable, unique and distinct.
2. Attribute :A characteristic of an entity. Ie. Entity My self, and attribute is MALE.
3. Relationship : Describes association among different entities.
   1. One to one : Manager to store
   2. One to many : Customer to invoices (many)
   3. Many to many : (many)student to (many) classes

To understand this relationship between the tables better is often used with **ER Diagrams**. Than ER Model is composed of entity types and specifies relationships that can exist between instances of those entity types. ER Diagrams show relationships business process by represented visually between the tables with show the link (Primary keys).

How to join those tables together are 2 things :

Primary Key : A Column (or set of columns) **whose value uniquely identify** every row in a table

Foreign Key : One or more columns that **can be used together to identify** a single row in another table.

ER Diagram Notation :

1. Chen Notation :
   * + - 1. One to M for one to many relationships
         2. M to N for many to many relationships
         3. One to One for one to one relationship.

A black background with white text

Description automatically generated A screen shot of a computer

Description automatically generated A black background with white text

Description automatically generated

1. Crow’s Foot Notation

A black background with white text and green rectangles

Description automatically generated A diagram of a relationship

Description automatically generated A black and white text

Description automatically generated with medium confidence

1. UML Class Diagram Notation

A screen shot of a computer

Description automatically generated A screen shot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

**THINKING BEFORE YOU START (Using ER Diagram)**

A screenshot of a computer

Description automatically generated

**SELECT STATEMENT** need to specify two pieces of information. **SELECT What** you want columns and **FROM where** table you want to select it from.

* SELECT Column FROM table\_name ; (select what columns and from where table)
* Use comma if we want to add multiple columns names (SELECT Column1, Column2 FROM table\_name ;)
* **Use asterix wildcard** character if we want to request all columns (SELECT \* FROM table\_name; )
* **Use LIMIT** to make some limitations numbers of records / data if your database is Large.

**CREATING NEW TABLE** using CREATE TABLE table\_name ;

We must put all the columns, with the data types and key for relational. Ie. Table shoes contains

* Id with char(10) as datatypes and as a primary key of course can not be NULL.
* Brand with char(10) as datatypes and can not be NULL (NOT NULL)
* Color with char(10) as datatypes and can not be NULL (NOT NULL)
* Price with decimal(8,2) 8 digit with 2 kommas as datatypes and can not be NULL (NOT NULL)
* Description with char(10) as datatypes and can be NULL (NULL)

After define the table, time to adding the data into the table. We need A INSERT Command. INSERT INTO table\_name VALUES –followed by the **colums in order** as we mentioned before ; SAME Statement but with different results are shown :

Left side is not recommended, & right side is recommended because it written in details from columns orders and values.

Right side are safer because more have controls and where the data is going and into which column.

A screenshot of a computer program

Description automatically generated

CREATE TEMPORARY TABLES : Why??

1. Faster than creating a real table
2. Useful for complex queries using subsets and joins.
3. Will be deleted when current session is terminated

CREATE TEMPORARY TABLE table\_name AS ( SELECT What FROM Where WHERE Conditions) ;

A diagram of a table

Description automatically generated

COMMENT to SQL for mute the expression of code, and help us to remember what you are doing and why.

Comments with – (double dash) and also can be section section /\* *(comments)* \*/

Read SQL BASICS :

1. <https://aws.amazon.com/what-is/sql/>
2. <https://www.ntchosting.com/encyclopedia/databases/structured-query-language/>
3. <https://www.w3resource.com/sqlite/>

READ Data Modeling and ER Diagrams :

1. <https://en.wikipedia.org/wiki/Data_modeling>
2. <http://business-analysis-excellence.com/what-is-data-modeling/>
3. <https://agiledata.org/essays/dataModeling101.html>
4. <https://vertabelo.com/blog/data-warehouse-modeling-star-schema-vs-snowflake-schema/>
5. <https://www.youtube.com/watch?v=c0_9Y8QAstg>

READ Comparing SQL vs NoSQL :

1. <https://www.mongodb.com/resources/basics/databases/nosql-explained/nosql-vs-sql>
2. <https://www.coursera.org/articles/nosql-vs-sql>
3. <https://www.integrate.io/blog/the-sql-vs-nosql-difference/#:~:text=SQL%20databases%20are%20vertically%20scalable,data%20like%20documents%20or%20JSON>.

NoSQL : NoSQL databases are scalable horizontally, meaning that they use multiple nodes in a cluster to handle increased workloads. This allows to simply scale them by supplementing clusters with additional servers. NoSQL databases are suitable for structured, semi-structured, and unstructured data. As a result, NoSQL databases don't follow a rigid schema but instead have more flexible structures to accommodate their data-types. Furthermore, instead of using SQL to query the database, NoSQL databases use varying query languages.

NoSQL non-relational databases work well with unstructured data and typically possess the following properties:

* NoSQL is schema-less (no fixed data model).
* NoSQL databases have a dynamic schema for unstructured data, making integrating data in certain types of applications easier and faster.
* NoSQL uses non-tabular data models, which can be document-oriented, key-value, or graph-based. The most common NoSQL databases include MongoDB, Cassandra, HBase, Redis, Neo4j, and CouchDB.

**NoSQL and SQL** databases have many similarities for example to supporting data storage and queries, they both also allow one to retrieve, update, and delete stored data.

**NoSQL and SQL** databases have some differences :

1. Relational

* SQL databases are relational
* NoSQL databases are non-relational . Means They don't contain any rows, tables, or keys. This type of database utilizes a storage model based on the type of data it stores.

1. Structure

* SQL databases are table based
* NoSQL databases can be document-oriented, key-value pairs, or graph structures. In a NoSQL database, a document can contain key value pairs, which can then be ordered and nested.

1. Scalability

* SQL databases **scale vertically**, usually on a single server.
* NoSQL databases **offer horizontal scalability,** meaning that more servers simply need to be added to increase their data load. This means that NoSQL databases are better for modern cloud-based infrastructures,

1. Language

* SQL databases use SQL (Structured Query Language). SQL has a fixed-defined schema,
* NoSQL databases use dynamic schema example JSON (JavaScript Object Notation), XML, YAML, or binary schema, facilitating for unstructured data. NoSQL databases are more flexible.

1. Support
   * SQL popular language, well supported
   * NoSQL has varying levels of support in various database systems.
2. Data Structure

* SQL databases are table-based
* NoSQL databases are document, key-value, graph, or wide-column stores

1. Use Cases

* SQL databases are table-based
* NoSQL databases are document, key-value, graph, or wide-column stores

|  |  |  |  |
| --- | --- | --- | --- |
| Pros of SQL | Cons of SQL | Pros of NoSQL | Cons of NoSQL |
| * SQL is widely understood and supported; most developers know it well. * SQL is extremely useful for simple aggregations over large datasets, such as calculating averages. * SQL is extremely useful for setting up simple ETL jobs, especially if the input and output formats are relational databases. * SQL is well-documented and easy to learn. | * The performance of SQL can be poor on substantial data sets because it requires multiple passes over the data to complete many operations (especially joins). * Debugging SQL can be complicated because it doesn't provide informative error messages. * The syntax of SQL tends to be verbose compared with programming languages like [Python or R](https://www.coursera.org/articles/python-or-r-for-data-analysis), which makes it harder to write complex transformations as scripts or functions. | * Flexible schema * Usable on distributed infrastructure platforms * Low-cost infrastructure * High availability and throughput | * Less mature technology and difficult to manage * Limited query capabilities * Data inconsistency and poor performance in some complex scenarios |