Database

Data هي أي حاجة بتتخزن أو أي حاجة ليها قيمة أو أي مجموعة من القيم ونخزن الداتا دي في الـ Database

*Databases* store data, do many operations on it and find relations between them.

الفرق بين الـ spreadsheet والـ database هو ان في spreadsheet بنخزن الداتا بشكل منفصل في اعمدة وصفوف بس للأسف لما نيجي ندور على حاجة معينة الموضوع بيكون صعب وبيكون شبه مستحيل لما الداتا تكون كبيرة جدا فبالتالي مش بنقدر نتعامل معاه بشكل كويس

أما DB بتسهل علينا اننا نوصل لجزء معين وكمان بتخزن عدد مرات الدخول وفيه خاصية إمكانية الوصول لو فيه بيانات معينة مينفعش تكون public وكمان لو فيه بيانات متكررة او لو فيه بيانات مرتبطة ببعض

**Relational database**

عبارة عن combining attributes of real word

**Entity**: is anything we store data about

**Attribute:** is things we store about this entity

يعني لو بنخزن بيانات عن شخص ف الشخص دا entity واسمه وعمره مثلا دي attributes

وبنخزن البيانات دي في جدول يتكون من صفوف وأعمدة

كل صف **row** عبارة عن all attribute values for specific entity

وكل عمود **column** عبارة عن all values for specific attribute type

Database management system DBMS

* used to store, retrieve, and run queries on data.
* Change the way of presented data. (View mechanism)

 there are three schemas:

1. conceptual schema: identify the highest-level relationships between the different entities.
2. Logical schema: describes the data as much detail as possible.
3. Physical schema: describe how data is to be represented and stored.

SQL programming language used to communicate to database

Used to

* Define the database structure 🡪DDL.
* manipulate the data within🡪 DML.

**Database Design**

Used to separate info over multiple tables rather than having one huge table.

**Data integrity**

the overall accuracy, completeness, and consistency of data.

يعني بنتأكد من البيانات وصحتها وعدم تكرارها وال relationships بينها

* entity integrity “unique entity”

عشان نتأكد من صحة بيانات الـ entity وعدم تكرارها بنخلي فيه حاجة unique زي ال primary key

* referential integrity

لو فيه relation بين جدولين مثلا فلازم يكون فيه حاجة بت reference على الجدول التاني وبنستخدم الـ foreign key

* domain integrity “the type and the range of what be stored.”

يعني مثلا لو فيه attribute اسمه phone number فأكيد لازم يكون أرقام مينفعش نكتب char مثلا ولو انا محدد انه لازم يكونوا عدد معين من الأرقام مينفعش اكتب اكتر من كدا

**atomic value**

لما بنخزن حاجة ف ال database بن حزنهم على هيئة atomic value زي مثلا لو عندنا attributeاسمه address ف ونقسمه لـ city, street and area code

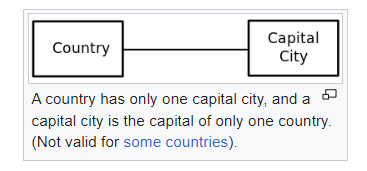
**Entity Relationship Modeling**

بستخدمه عشان أقدر احول الداتا design وشكل مفهوم أقدر اطلع منه relationships مع بعض

**Relationship**

it is an association between tables.

* **one to one**

one entity has a connection with one other entity ex. each country has one capital city

* **one to many**

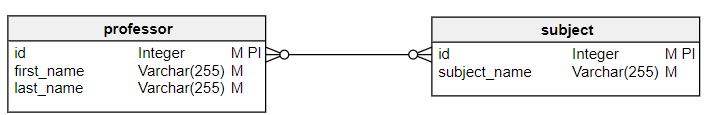
one entity can have a relationship with multiple other entities.

ex.one post has many comments, one person can make many comments.

* **many to many**

more than one entity has a relationship with more than one other.

ex. one or more profs teach many subjects and vice versa.



**parent table and child table**

***parent table*** has a primary key and does not inherit anything from the child table.

***child table*** has a foreign key and inherit from the parent table.

They’re linked in a way that's described by a *parent–child relationship*. It’s usually used to specify where one table’s value refers to the value in another table (usually a primary key of another table)

**introduction of keys**

keys are the way to identify each record separately and uniquely, i.e., no duplicates.

A key in DBMS is an attribute or a set of attributes that help to uniquely identify a tuple (or row) in a relation (or table). keys should not change and never be empty and must be unique. Keys are also used to establish relationships between the different tables and columns of a relational database. Individual values in a key are called key values.

**lookup table**

A lookup table or LUT maps keys to values because keys are unique, and no value appears more than once.

it’s known as foreign key constraints.

its benefits:

* integrity
* uniqueness
* less work for updating data.
* improve functionality.
* allows for added complexity.

**Super key**

Any numbers of columns that force every row to be unique.

**Candidate key**

is at least number of columns needed to force every row to be unique.

**Primary key**: uniquely identify the specific row and must contain UNIQUE values {num, string, any type}, and cannot contain NULL values. And never change.

فيه نوعين من الـ primary key

* *Surrogate key* also called a synthetic primary key, is generated when a new record is inserted into a table automatically by a database والنوع دا ملوش وجود في العالم الحقيقي وملوش دلالة على عكس النوع الثاني زي مثلا ترتيب وترقيم.
* *Natural key* is a type of unique key in a database formed of attributes that exist and are used in the external world outside the database. ex. SSN, ID

**Alternate key**

We didn't choose it, but it may be selected as a primary key.

**Foreign key**: an attribute that links another database table and refers to the primary key of another table.

دا بيعرفنا ايه الـ relationships اللي بين الجداول

**Foreign key constraints**

**-**keep everything consistent and protect data integrity and when update anything then the references would have to updated too (referential integrity).

**So,** we use these concepts:

**ON** **DELETE RESTRICT 🡪** when we delete the pk in parent, it’s going to throw an error and prevent it.

**ON UPDATE RESTRICT 🡪** when we update the pk in parent, it’s going to throw an error and prevent it.

**ON DELETE CASCADE 🡪** when we delete pk in parent then the same thing ‘ll happen to FK in the child

**ON UPDATEA CASCADE** **🡪** when we update pk in parent then the same thing ‘ll happen to FK in the child

**ON DELETE SET NULL 🡪** when we delete pk in parent then the child ‘ll set NULL

**ON UPDATEA SET NULL** **🡪** when we update pk in parent then then the child ‘ll set NULL

**Simple key**

It consists of one column and it’s most common with surrogate key.

**Composite key**: is a combination of two or more columns in a table that can be used to uniquely identify each row in the table.

This is most common with natural keys.

 زي first\_name and last\_name واستخدام تاني لما نضيف اتنين foreign keys مع بعض في جدول معين بيكونوا primary key للجدول دا

**Compound key**

The key that has multiple columns and they’ s all keys themselves.

بنشوفه لما تكون الـ relation M:N فبلاقي اتنين FK بيعودوا على نفس الحاجة ومرتبطين ببعض زي مثلا لما نقول ان مجموعة من الطلاب مسجلين ف مواد كتير والعكس المواد الكثير فيها طلاب كتير فبنحطهم ف جدول واحد

**Introduction to Entity Relationship Modeling**

**Cardinality**

It refers to the number of unique values in a relational table column relative to the total number of rows in the table.

* **One to one**
* **One to many**
* **Many to many**

**Modality**

It describes whether a relationship between two or more entities is even required or not.

**Normalization**

A process to correct things or data integrity problem like repeating data

There are 3 main forms to produce a good structure database:

**1NF** A relation is in *1st Normal Form* in DBMS (or 1NF) if it contains atomic values. It states that an attribute of a table cannot hold multiple values.

**2NF** A relation is said to be in *2nd Normal Form* in DBMS (or 2NF) when it is in the First Normal Form but has no non-prime attribute functionally dependent on any candidate key's proper subset in a relation.

**3NF** The first condition for the table to be in *Third Normal Form* is that the table should be in the Second Normal Form. The second condition is that there should be no transitive dependency for non-prime attributes, which indicates that non-prime attributes (which are not a part of the candidate key) should not depend on other non-prime attributes in a table. Therefore, a transitive dependency is a functional dependency in which A → C (A determines C) indirectly, because of A → B and B → C (where it is not the case that B → A).The third Normal Form ensures the reduction of data duplication. It is also used to achieve data integrity.

clustered index

A clustered index is used to define the order or to sort the table or arrange the data by alphabetical order just like a dictionary.

It is faster than a non-clustered index.

Primary Keys of the table by default are clustered index.

non-clustered index

A non-clustered index collects the data at one place and records at another place.

It is slower than the clustered index.

Composite key when used with unique constraints of the table act as non-clustered index.

Data types

Date

* DATE A date. Format: YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31'

Numeric

* 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).
* 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.

String

* 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.

**JOIN**

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

* SELF JOIN A self join is a regular join, but the table is joined with itself.
* INNER JION

selects records that have matching values in both tables.

Example between 2 tables:

SELECT Orders.OrderID, Customers.CustomerName

FROM Orders

INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;

Example JOIN Three Tables:

SELECT Orders.OrderID, Customers.CustomerName, Shippers.ShipperName

FROM Orders

INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID)

INNER JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID);

* OUTER JOIN

**Left** returns all records from the left table (table1), and the matching records from the right table (table2). The result is 0 records from the right side if there is no match.

**Right** returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side if there is no match.

**Full** returns all records when there is a match in left (table1) or right (table2) table records.

FULL OUTER JOIN and FULL JOIN are the same.

Aliases

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

Aliases are often used to make column names more readable.

An alias only exists for the duration of that query.

An alias is created with the AS keyword.

Aliases can be useful when:

* There are more than one table involved in a query
* Functions are used in the query
* Column names are big or not very readable
* Two or more columns are combined together

Syntax:

SELECT column\_name AS alias\_name  
FROM table\_name;

REFFRENCES

THIS BASED ON THIS VIDEO

* <https://youtu.be/ztHopE5Wnpc>

and these websites helped me too ^-^

* <https://www.simplilearn.com/tutorials/sql-tutorial>
* <https://www.ibm.com>
* <https://www.geeksforgeeks.org>
* <https://www.w3schools.com/sql>