**Entities:** Una entidad pues, pueden ser **generic** or **specific.** Each entity gets their own table

**Attributes:** Un atributo de la entidad, puede ser **generic** or **specific**

For example:

Entity = customer

Generic attributes = height, hair color, age

Specific attributes = 2m, grey, 29

**SQL** = Structured Query Language. Es un lenguaje estandar

**RDBMS**: **Relational** Database Management System. Fancy way of saying “database”

**Relations**: Mathematical way of describing tables

**Null:** Absence of a value

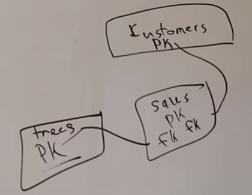
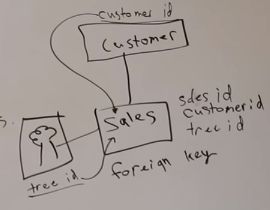
**Relationship**: Connection between two tables

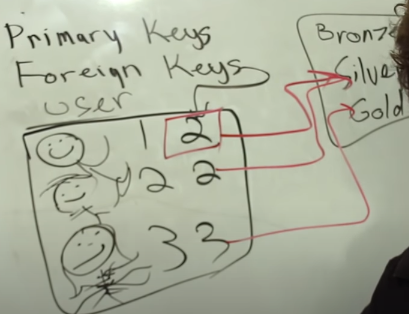
**Primary key**: How to keep every single row in a table unique. Every entity will have its own ID. Por ejemplo si en una tabla tiene el stock de árboles y tienes otra tabla de los arboles vendidos entonces cuando un árbol se vende el primary key es el que viaja con ese arbol vendido. PK gets indexed automatically. PK does not change. Por ejemplo si tienes dos Caleb entonces se diferencian porque esas dos rows van a tener diferentes keys. Every single table will have PK. An example would be employee number

**Natural key**: Has a real world meaning.

**Surrogate key**: Random database generated number to prevent duplicate values.

**Foreign Key:** Reference to a PK in another table.





**Forein key constraint**: Son las reglas para link entre FK y PK

**Constraint**: Things that prevent relationships between PK and FK to be broken, maintains integrity

**Indexes**: Helps find information faster through generic attributes. Por ejemplo los PK se indexan automaticamente con oracle. Tambien los PK se deben indexar

**Composite Index:** Cuando usas dos cosas como index, por ejemplo email + name

**Naming convention**: There are several naming conventions with DBs

**Data type:** Es el tipo de data y lo puedes asignar a la data.

**Atomic**: Concepto de granular la data lo más posible without losing its meaning

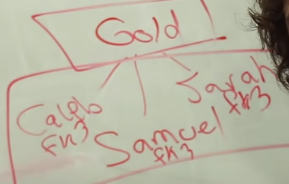
**Integrity**: Es la integridad de la DB y hay 3 tipos:

* **Entity**:Needs a PK
* **Referential**: Constraints de relaciones entre tablas para asegurar integridad. Por ejemplo “wow” comment made by NULL
* **Domain**: Give possible options for columns and tables. Prevent certain entries in columns. Por ejemplo todos los user\_names deben ser letras. Prevents que la gente ponga basura en tu tabla

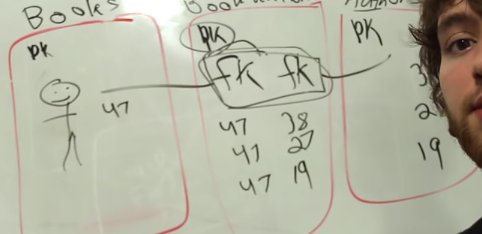
**Entity Relationship Diagram (ERD)**: Diagama que muestra las relaciones entre las tablas

**Relationships:**

* **One-to-one:** Unique attribute for entity
* **One-to-many**: For example one table of gym memberships with 3 tiers and another table of customers. Many customers can have one of those 3 tiers. Parent-child relationship:



* **Many-to-many**: Needs an **intermediary table** to associate different tables together. Basically it´s entity-to-entity with intermediary table. Books, books authors, authors



**Normalization:** Rules you can opt to follow to protect database integrity. Normal forms:

1. All columns need to describe something atomic. Data for each row is one thing
2. Get rid of partial dependencies with composite keys.
3. Evitar transative dependencies

**Functional Dependency**

An analysis of dependency operates on the attribute level, i.e. one or more attribute is determined by another attribute, it comes before the concept of keys. 'The role of a key is based on the concept of determination. *'Determination is the state in which knowing the value of one attribute makes it possible to determine the value of another.' Database Systems 12ed*

Functional dependency is when one or more attributes determine one or more attributes. For instance:

Social Security Number -> First Name, Last Name.

However, by definition of functional dependency:

(SSN, First Name) -> Last Name

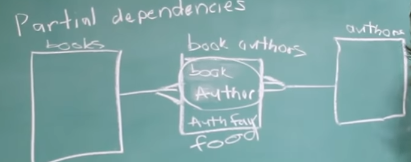
This is also a valid functional dependency. The **determinants** (The attribute that which determines another attribution) are called **super key**.

**Full Functional Dependency**

Thus, as a subset of functional dependency, there is the concept of **full functional dependency**, where the bare minimal determinant is considered. We refer those bare minimal determinants collectively as one **candidate key** (weird linguistic quirk in my opinion, like the concept of vector).

**Partial Functional Dependency**

However, sometimes one of the attributes in the candidate key is sufficient to determine another attribute(s), BUT not all, in a relation (a table with no rows). That, is when you have a partial functional dependency within a relation. Por ejmplo:



Author food partially depends on the author but not the books so it´s a partial dependency.

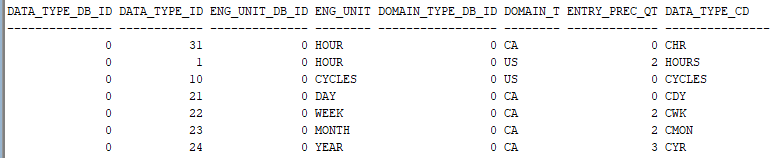
**Composite (Compound) Key:** Two keys combined. Por ejemplo combinacion de dos PK en un intermediary table

**Transative Dependency:** A colum that depends on a column that depends on a column , etc.

**Useful Entities:**

* **INV\_CURR\_USAGE:** Sacas horas current
  + TSN\_QT: Current TSN
  + TSO\_QT. Current TSO
* **INV\_AC\_REG**: Sacas toda la identificacion del avion:
  + LINE\_NO\_OEM : Line number
  + VAR\_NO\_OEM: variable number
  + AC\_REG\_CD : matricula
* **INV\_INV:**
  + ASSMBL\_CD: Assy code
  + MANUFACT\_DT: Manufacture date
  + SERIAL\_NO\_OEM: S/N
  + INV\_COND\_CD
* **EQP\_PART\_NO**
  + PART\_NO\_OEM: P/N
  + BOM\_PART\_CD: El config slot
* **EVT\_EVENT:**
  + EVENT\_STATUS\_CD: Status del evento
  + EVENT\_DT: Complete date
  + HIST\_BOOL: Te dice si el evento ya se completo
  + SCHED\_START\_DT: Scheduled time to start event in local time
* **TASK\_SCHED\_RULE:**
  + DEF\_INTERVAL\_QT: Intervalo in “data type units” (investigar que significa)
* **TASK\_TASK:**
  + TASK\_CD: El task code pues
* **EQP\_ASSMBL\_BOM:**
  + ASSMBL\_BOM\_CD: config slot

**Concatenation:** SELECT eqp\_assmbl.assmbl\_cd||' ('||eqp\_assmbl.assmbl\_name||')' FROM eqp\_assmbl



**SELECT**  sdsdsds **FROM** sdsdasdas **INNOR JOIN** sdsdsdsd **ON**  sadssad = sdadsasds **WHERE** eqp\_assmbl.assmbl\_db\_id = 100001

SELECT SERIAL\_NO\_OEM, TSN\_QT FROM INV\_INV INNER JOIN INV\_CURR\_USAGE ON INV\_INV.INV\_NO\_ID = INV\_CURR\_USAGE.INV\_NO\_ID WHERE PART\_NO\_ID = '105739'

**Para sacar el usage de un avion especifico:**

SELECT AC\_REG\_CD,TSN\_QT FROM INV\_AC\_REG INNER JOIN INV\_CURR\_USAGE ON INV\_AC\_REG.INV\_NO\_ID = INV\_CURR\_USAGE.INV\_NO\_ID WHERE AC\_REG\_CD = 'HP-1827CMP' AND DATA\_TYPE\_ID = 10

[8:28 PM] Jean Bertholin Fabrega (CM)

recomendacion... si quieres saber si te la cagaste o no en un join (por ejemplo, que te haya faltado un key o algo) cuenta cuantos resultados tenias antes del join y cuantos tenias despues

​

[8:29 PM] Jean Bertholin Fabrega (CM)

no tienes que poner a correr todo el query para obtener ese resultado

​

[8:29 PM] Jean Bertholin Fabrega (CM)

una vez ejecutas, dale click derecho sobre los resultados y hay una opcion que dice count rows

usualmente cuando te duplican resultados asi es que le hace falta algun key

**INFO DE AVION**

**SELECT INV\_AC\_REG.AC\_REG\_CD, INV\_INV.MANUFACT\_DT, INV\_AC\_REG.LINE\_NO\_OEM, INV\_AC\_REG.VAR\_NO\_OEM, INV\_INV.SERIAL\_NO\_OEM,**

**INV\_CURR\_USAGE.TSN\_QT, EQP\_PART\_NO.PART\_NO\_OEM AS AC\_MODEL**

**FROM INV\_AC\_REG**

**INNER JOIN INV\_CURR\_USAGE ON**

**INV\_AC\_REG.INV\_NO\_ID = INV\_CURR\_USAGE.INV\_NO\_ID**

**INNER JOIN INV\_INV ON**

**INV\_AC\_REG.INV\_NO\_ID = INV\_INV.INV\_NO\_ID**

**INNER JOIN EQP\_PART\_NO ON**

**INV\_INV.PART\_NO\_ID = EQP\_PART\_NO.PART\_NO\_ID**

**WHERE AC\_REG\_CD = 'HP-1825CMP'**

**PARTES INSTALDAS:**

SELECT

INV\_AC\_REG.AC\_REG\_CD,

EQP\_PART\_NO.PART\_NO\_OEM,

EQP\_PART\_NO.PART\_NO\_SDESC,

EQP\_MANUFACT.MANUFACT\_NAME,

EQP\_BOM\_PART.BOM\_PART\_CD

FROM INV\_INV II

INNER JOIN INV\_INV II\_AC

ON ii\_ac.inv\_no\_db\_id = ii.h\_inv\_no\_db\_id

AND ii\_ac.inv\_no\_id = ii.h\_inv\_no\_id

AND ii\_ac.inv\_cond\_cd NOT IN 'ARCHIVE'

AND ii\_ac.authority\_id IS NULL

INNER JOIN INV\_AC\_REG

ON inv\_ac\_reg.inv\_no\_db\_id = ii\_ac.inv\_no\_db\_id

AND inv\_ac\_reg.inv\_no\_id = ii\_ac.inv\_no\_id

INNER JOIN EQP\_PART\_NO ON

ii.part\_no\_id = eqp\_part\_no.part\_no\_id

INNER JOIN EQP\_MANUFACT ON

EQP\_PART\_NO.MANUFACT\_CD = EQP\_MANUFACT.MANUFACT\_CD

INNER JOIN EQP\_BOM\_PART ON

ii.BOM\_PART\_DB\_ID = EQP\_BOM\_PART.BOM\_PART\_DB\_ID AND

ii.BOM\_PART\_ID = EQP\_BOM\_PART.BOM\_PART\_ID

WHERE ii.inv\_class\_cd IN 'TRK'

AND ii.inv\_cond\_cd IN 'INSRV'

AND INV\_AC\_REG.AC\_REG\_CD = 'HP-1839CMP'

AND EQP\_PART\_NO.PART\_NO\_SDESC NOT LIKE '%SOFTWARE%'

**EVENTOS DE C-CK (NEXT DUE):**

SELECT

TASK\_TASK.TASK\_ID,

TASK\_TASK.TASK\_CD,

INV\_AC\_REG.AC\_REG\_CD,

SCHED\_STASK.BARCODE\_SDESC,

EVT\_SCHED\_DEAD.SCHED\_DEAD\_DT

FROM TASK\_TASK

INNER JOIN SCHED\_STASK ON

TASK\_TASK.TASK\_DB\_ID = SCHED\_STASK.TASK\_DB\_ID AND

TASK\_TASK.TASK\_ID = SCHED\_STASK.TASK\_ID

INNER JOIN INV\_AC\_REG ON

SCHED\_STASK.MAIN\_INV\_NO\_ID = INV\_AC\_REG.INV\_NO\_ID AND

SCHED\_STASK.MAIN\_INV\_NO\_DB\_ID = INV\_AC\_REG.INV\_NO\_DB\_ID

INNER JOIN EVT\_SCHED\_DEAD ON

sched\_stask.sched\_db\_id = EVT\_SCHED\_DEAD.event\_db\_id AND

sched\_stask.sched\_id = EVT\_SCHED\_DEAD.event\_id

INNER JOIN EVT\_EVENT ON

EVT\_SCHED\_DEAD.EVENT\_ID = EVT\_EVENT.EVENT\_ID AND

EVT\_SCHED\_DEAD.EVENT\_DB\_ID = EVT\_EVENT.EVENT\_DB\_ID

WHERE TASK\_TASK.TASK\_CD = 'C-CK-1 - 737-NG' AND

EVT\_EVENT.EVENT\_STATUS\_CD = 'ACTV' AND

EVT\_SCHED\_DEAD.USAGE\_REM\_QT <= 1095 AND

INV\_AC\_REG.AC\_REG\_CD = 'HP-1375CMP'

**LAST C-CK**:

SELECT

TASK\_TASK.TASK\_ID,

TASK\_TASK.TASK\_CD,

INV\_AC\_REG.AC\_REG\_CD,

SCHED\_STASK.BARCODE\_SDESC,

EVT\_SCHED\_DEAD.SCHED\_DEAD\_DT

FROM TASK\_TASK

INNER JOIN SCHED\_STASK ON

TASK\_TASK.TASK\_DB\_ID = SCHED\_STASK.TASK\_DB\_ID AND

TASK\_TASK.TASK\_ID = SCHED\_STASK.TASK\_ID

INNER JOIN INV\_AC\_REG ON

SCHED\_STASK.MAIN\_INV\_NO\_ID = INV\_AC\_REG.INV\_NO\_ID AND

SCHED\_STASK.MAIN\_INV\_NO\_DB\_ID = INV\_AC\_REG.INV\_NO\_DB\_ID

INNER JOIN EVT\_SCHED\_DEAD ON

sched\_stask.sched\_db\_id = EVT\_SCHED\_DEAD.event\_db\_id AND

sched\_stask.sched\_id = EVT\_SCHED\_DEAD.event\_id

INNER JOIN EVT\_EVENT ON

EVT\_SCHED\_DEAD.EVENT\_ID = EVT\_EVENT.EVENT\_ID AND

EVT\_SCHED\_DEAD.EVENT\_DB\_ID = EVT\_EVENT.EVENT\_DB\_ID

WHERE TASK\_TASK.TASK\_CD = 'C-CK-1 - 737-NG' AND

EVT\_EVENT.EVENT\_STATUS\_CD = 'COMPLETE' AND

INV\_AC\_REG.AC\_REG\_CD = 'HP-1724CMP' AND

to\_date(CURRENT\_DATE) - to\_date(EVT\_SCHED\_DEAD.SCHED\_DEAD\_DT) <=1095 AND

EVT\_SCHED\_DEAD.DATA\_TYPE\_ID = 21

**SHAREPOINT ACCESS**

import requests

from requests.auth import HTTPBasicAuth

USERNAME = 'ggalina'

PASSWORD = 'Copa2020a'

SITE = 'https://copaairlines.sharepoint.com/teams/VP\_Tecnica/Dir\_Ing\_y\_Soporte/Ingenieria/default.aspx'

headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.97 Safari/537.36'}

response = requests.get(SITE, auth=HTTPBasicAuth(USERNAME, PASSWORD),headers=headers)

print(response.status\_code)

END POINT URL PARA SACAR ITEMS DE UNA LISTA

<https://copaairlines.sharepoint.com/teams/VP_Tecnica/Dir_Ing_y_Soporte/Ingenieria/_api/web/lists/GetByTitle('AD%20CONTROL')/items>

<https://docs.microsoft.com/en-us/sharepoint/dev/sp-add-ins/get-to-know-the-sharepoint-rest-service?tabs=http> (DOC PARA SP API)