BASE DE DATOS Y DBMS

*"Una **base de datos** es una colección de datos organizados relevantes a un dominio que son administrados y consultados mediante un sistema de administración de base de datos (DBMS)."*

*" Un **Database Management System (DBMS)** es un sistema que permite la gestión y consulta de base de datos."*

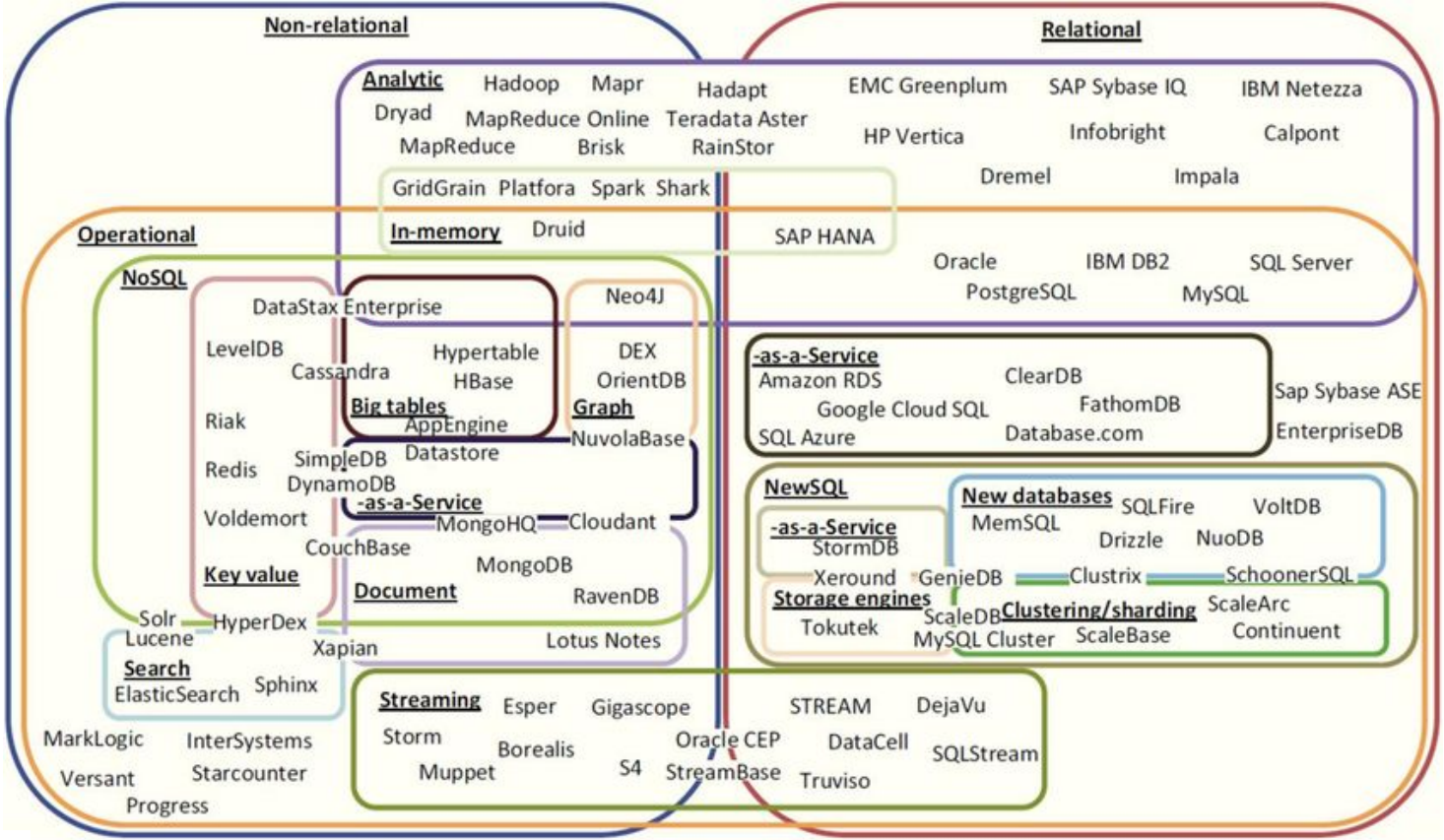
- Gestion:
  - Definición de esquema.
  - Creación de datos.
  - Creación de relaciones, actualización de datos, seguridad, etc.
- Consulta:
  - Responder preguntas sobre los datos.
  - Realizar análisis con los datos.

MODELO DE DATOS

*"Un **modelo de datos** es una colección de herramientas conceptuales para describir los datos, las relaciones entre ellos, su semántica y las restricciones de consistencia."*

- Define la estructura lógica de la base de datos.
- Impacta en la forma en que los datos son almacenados y manipulados.
- Existen diferentes modelos de datos:
  - **Modelo Relacional (SQL)**
  - **Modelos No relacionales (NoSQL):**
    - Modelo de Objetos (Realm DB)
    - Modelo de Documentos (MongoDB)
    - Modelos de Grafos (Neo4J)
    - Modelos de Llave-Valor (Redis)
    - Modelos Columnares (SciDB)

CLASIFICACIÓN DE BASE DE DATOS



CLASIFICACIÓN DE BASE DE DATOS

BASE DE DATOS RELACIONALES (I)

Las bases de datos relacionales (RDBMS) son una implementación del modelo relacional introducido en los 70s por Edgar Codd en *A Relational Model of Data for Large Shared Data Banks*

- La estructura básica es la **tabla** (relación).
- La tabla define **columnas** (atributos) y tiene **filas de datos** (tuplas).
- Las columnas tienen un cierto **tipo de datos** (dominio).
- Se pueden **relacionar** una o más tablas.
- Usan **SQL** como lenguaje para manipular y consultar datos.

$Columna_1$	$Columna_2$	$Columna_n$
$Valor_{11}$	$Valor_{12}$	$Valor_{1n}$
...	...	...
$Valor_{m1}$	$Valor_{m2}$	$Valor_{mn}$

“Diseño” de Bases de Datos

Base de Datos 2024

Diseño de Base de Datos - Para que?

Una base datos bien diseñada debe:

- Eliminar la redundancia de datos
  - Los datos no deben almacenarse en más de un lugar.
  - No malgastar almacenamiento.
  - Evitar inconsistencias.
- Asegurar la integridad de los datos.
  - Claves primarias correctas
  - Evitar inconsistencias.
- Facilitar la aplicación de las reglas de negocio.

Cómo lo hacemos?

1. Análisis de Requerimientos.
2. Creación de Tablas
3. Definición de relaciones entre tablas
4. Normalizacion
5. Repetir

Análisis de Requerimientos

- Definir el objetivo de la base de datos.
- Para que se va a usar?
- Cual es el dominio de aplicación?
- Cuales son las entidades principales del dominio?
- Qué datos definen a las entidades?
- Que tipo de consultas/reportes debemos responder? (Acordate juan de SQL y NoSQL respecto a cómo diseñar basado en consultas)
- Por lo general, la base de datos se diseña junto con la aplicación.

Creación de Tablas

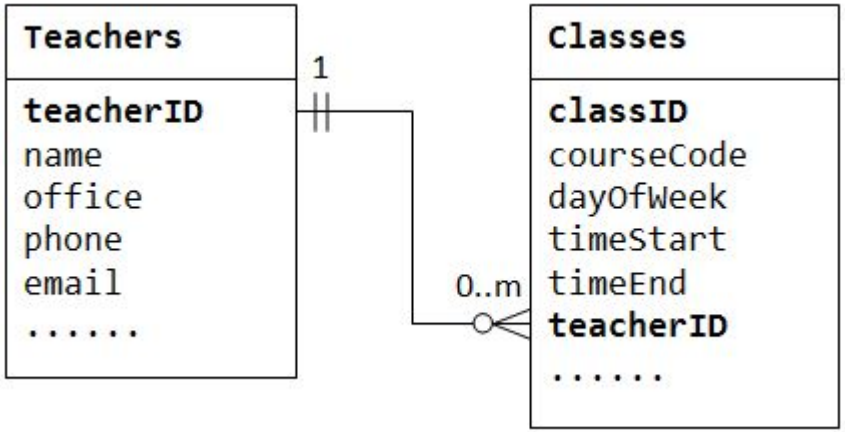
- Transformar en tablas los conceptos del dominio.
- Cuales son los tipos de datos?
- Especificar las claves primarias de las tablas.

Una clave primaria debe ser:

- Unica y No nula
- Debe ser simple e intuitiva.
- Debe ser inmutable.
- Usualmente son de tipo numérico y autoincremental.
- Preferir claves con la menor cantidad de columnas posible.

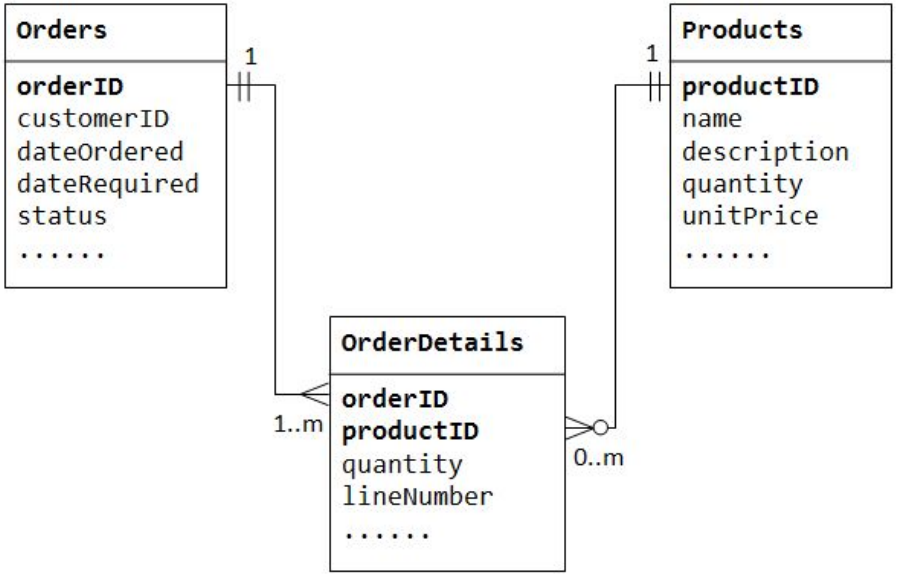
Relaciones entre tablas: uno-a-muchos

- No se pueden representar con una sola tabla.
- Existen una tabla padre (Uno) y una tabla hija (Muchos).
- En la tabla hija tenemos como clave foránea la clave primaria de la tabla padre.
- Para cada valor en la tabla padre pueden haber cero, una o más filas en la tabla hija.
- Para cada valor en la tabla hija solo existe una fila en la tabla padre.



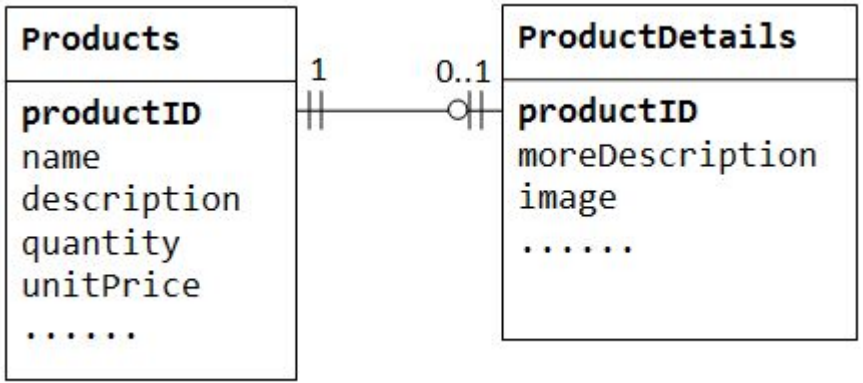
Relaciones entre tablas: muchos-a-muchos

- Para soportar relaciones muchos-a-muchos necesitamos introducir una tercer tabla: la **tabla de asociación**.
- Se modela como dos relaciones uno-a-muchos entre las tablas padres y la tabla de asociación.



Relaciones entre tablas: uno-a-uno

- Se suelen utilizar para representar información complementaria
- Útiles para partir una tabla “grande” en tablas más pequeñas.



Normalizacion

- **Primera Forma Normal(1NF):** El dominio de las columnas debe ser atómico.
  - Evitar listas de valores en una columna. Usar uno-a-muchos.
  - Si usan columnas JSON no son 1NF.
- **Segunda Forma Normal (2NF):** 1NF + toda columna que no forma parte de la clave primaria depende de todas las columnas de la clave primaria.
  - Todos las columnas están definidas por la clave primaria.
- **Tercera Forma Normal(3NF):** 2NF + toda columna que no forma parte de la clave primaria depende solamente de la clave primaria.
  - Evitar columnas derivadas

Consejos

- El diseño de una base de datos es más un arte que una ciencia.
- Hay mucha decisiones que tomar.
- La mejor decisión siempre depende del contexto.
- Es más fácil saber lo que NO hay hacer que lo SI hay que hacer.