Fundamentos de bases de datos

Jose Emilio Labra Gayo

Universidad de Oviedo

Más información

Material del curso: https://github.com/cursosLabra/introBBDD

Sobre el profesor: https://labra.weso.es/

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

Importancia de las Bases de datos

- Bases de datos = corazón de la era digital
- Todo lo que usamos depende de Bases de Datos
  - Desde redes sociales, hospitales, sistemas financieros, …
  - Prácticamente todas las aplicaciones modernas se apoyan en Bases de Datos
- En mundo empresarial
  - Permiten la toma de decisiones
  - Análisis de grandes volúmenes de información
  - Esenciales para IA y aprendizaje automático
  - Aseguran consistencia, integridad y disponibilidad de la información
- Están en el núcleo de todos los servicios
  - Sin Bases de datos, no hay información
  - Sin información, no hay decisiones

Importancia y tecnología en negocios

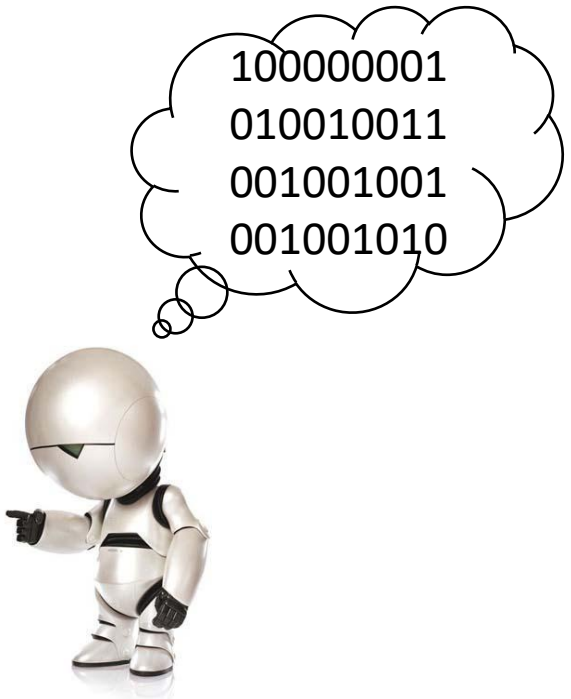
Función esencial de la informática = almacenar datos

- 2 problemas
  - ¿Cómo representar la información?
  - ¿Cómo almacenar esas representaciones?

"Los datos son algo precioso, y durarán más que los propios sistemas"
Tim Berners-Lee, Inventor de la Web

Personas vs Máquinas

Creatividad, imaginación
Imprevisibles (cometemos errores)
Nos cansamos ante tareas repetitivas
Comprensión basada en contexto



Programadas para ciertas tareas
Previsibles (sin errores*)
Tareas repetitivas sin problema
Dificultad para entender el contexto

*cuando están bien programadas

Representación de información

En ordenadores actuales, todo son 0s y 1s

Distintos tipos de información

- Valores booleanos: 0 (falso), 1 (true)
- Números: codificaciones de tamaño fijo ó variable
- Caracteres: Codificaciones como ASCII o Unicode (https://unicode.org/)
- Cadenas de texto
  - Secuencias de caracteres
- Registros y estructuras de datos
  - Valores compuestos. Ejemplo: coordenadas, tablas, árboles, grafos, ...
- Valores binarios
  - Secuencias de 0's y 1's

Tamaños de datos

Unidad	Abrev.	Tamaño	En bytes
Bit			
Byte	B	8 bits	
Kilobyte	KB	1024 B	≈ 1,000 bytes
Megabyte	MB	1024 KB	≈ 1,000,000 bytes
Gigabyte	GB	1024 MB	≈ 1,000,000,000 bytes
Terabyte	TB	1024 GB	≈ 1,000,000,000,000 bytes
Petabyte	PB	1024 TB	≈ 1,000,000,000,000,000 bytes
Exabyte	EB	1024 PB	≈ 1,000,000,000,000,000,000 bytes
Zettabyte	ZB	1024 EB	≈ 1,000,000,000,000,000,000,000 bytes
Yottabyte	YB	1024 ZB	≈ 1,000,000,000,000,000,000,000,000 bytes
Ronnabyte	RB	1024 YB	≈ 1,000,000,000,000,000,000,000,000,000 bytes

The Zettabyte era: https://en.wikipedia.org/wiki/Zettabyte_Era

Ficheros

Contienen información sobre cualquier cosa

Diferentes tipos

- Ficheros de texto
  - Legibles por seres humanos
  - Algunos pueden ser procesables automáticamente
  - Ej. Ficheros de configuración, programas informáticos, ficheros .csv, etc.
- Ficheros binarios
  - No legibles por humanos
  - Suelen estar optimizados para procesamiento automatizado
  - Ej. Imágenes (.jpg), vídeos (.mpg), Excel (.xls), Word (.doc), Empaquetados (.zip), ...

Sistema de ficheros

Sistema que asocia un nombre a un fichero

Normalmente en forma jerarquizada (árbol de directorios)

Cada ordenador puede almacenar información en sistema de ficheros

Pero...

- Información heterogénea y dispersa en ficheros
- Difícil realizar consultas
- Difícil procesamiento automatizado de la información
- Difícil gestionar accesos y seguridad
- . . .

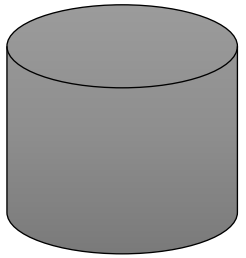
Bases de datos

Base de datos = colección de datos almacenados y accedidos de forma electrónica

- Bases de datos pequeñas pueden almacenarse en sistema de ficheros
- Pero…las grandes bases de datos se alojan en clústers de computadores o en la nube

Datos = clave del éxito de muchas empresas

- Los datos suelen perduran más que las propias aplicaciones



Sistema gestión de bases de datos (SGBD)

- Software que permite gestionar información de las bases de datos
  - Suele ser software complejo, realizado por grandes empresas
    - IBM, Oracle, Microsoft, ...
- Sistema de gestión de base de datos ≠ Base de datos
  - SGBD = software de gestión
    - Ejemplo: SQL Server, MySQL, SQLite, …
  - Base de datos = conjunto de datos específicos
    - Ejemplo: Datos de un banco concreto

¿Qué gestionan los SGBD?

Almacenamiento
- Pueden optimizar la representación interna de los datos

Usuarios y permisos
- Quién puede ver/modificar qué datos

Integridad de los datos

Accesos concurrentes

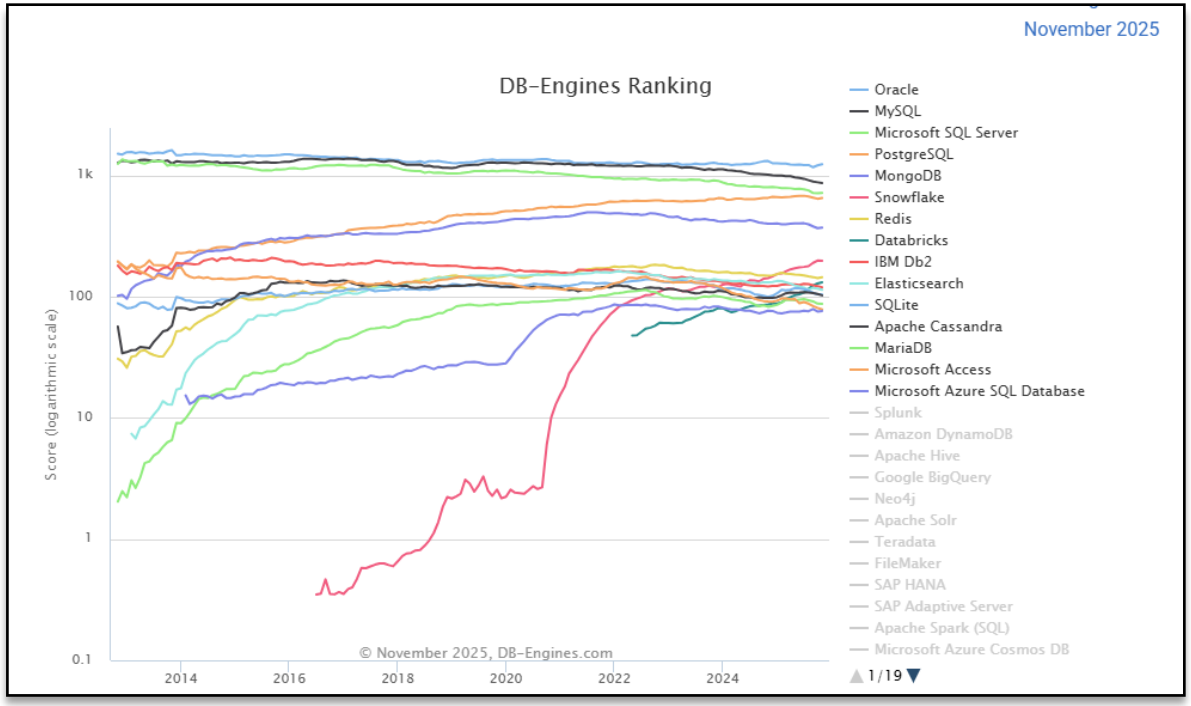
Transacciones y atomicidad

Recuperación de datos

Algunos SGBD

Múltiples SGBD:

- Oracle Database
- MySQL
- Microsoft SQL Server
- PostgresSQL
- SQLite
- . . .



Ránking: https://db-engines.com/en/ranking

Aplicaciones de bases de datos

Las bases de datos son una parte fundamental de cualquier empresa

Cada vez más, las empresas se convierten en empresas de datos

En cualquier dominio

- Banca, seguros, ventas, siderurgia, etc.
- Tecnológicas: eBay, Facebook, Google, Amazon, ...
- Algunos datos
  - El negocio de SGBD = 89 mil millones de $ en 2024
  - Se espera que alcance los 248 mil millones en 2034 [1]

[1] https://www.expertmarketresearch.com/reports/database-management-system-market

Actividades profesionales relacionadas

Gobierno de datos

- Ciclo de vida, seguridad, acceso, etc.

Ingeniería de datos

- Representación de datos, infraestructura, rendimiento, costes

Ciencia de datos

- Análisis de datos, extracción de información/conocimiento

Ingeniería del software

- Aplicaciones que trabajan con los datos, integración, explotación, etc.

Administración de base de datos

- Mantenimiento Bases de datos, seguridad, réplicas, *backups*, etc.

Actividades relacionadas

Migración de datos

- Cambiar los esquemas de las BBDD
- Cambiar el SGBD

Análisis y minería de datos

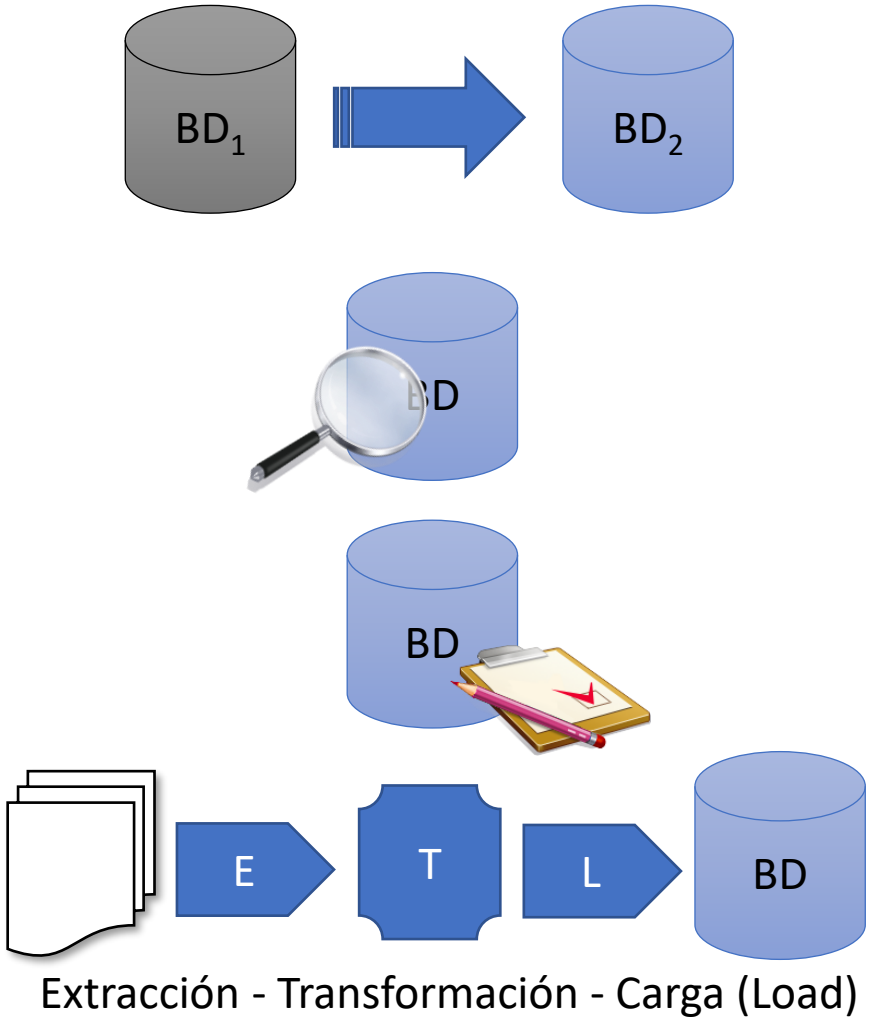
- Extraer información a partir de datos
- En ocasiones se extrae de los *logs* de las BBDD

Auditoría de datos

- Certificación

Ingestión y conservación de datos

- Procesos ETL (Extracción - Transformación - Carga)



Extracción - Transformación - Carga (Load)

¿Hojas de cálculo como alternativa a BBDD?

Hojas de cálculo

Modelo de tabla (celdas)

Valores de celdas pueden calcularse a partir de otras

Pensadas para acceso interactivo

- Proceso automatizado más engorroso

Tamaño/capacidad limitado

Control de usuarios limitado

Acceso concurrente limitado

No hay soporte para recuperación

Bases de datos

Varios modelos (tablas...)

Datos estáticos (manipulación mediante programas)

Pensadas para acceso *batch* o mediante software

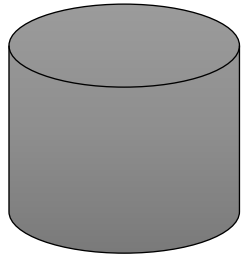
- Proceso interactivo más engorroso

Almacenamiento grandes cantidades de datos

Control de usuarios formar parte de SGBD

Soporte para concurrencia y transacciones

Soporte para recuperación (backups) y similares



Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

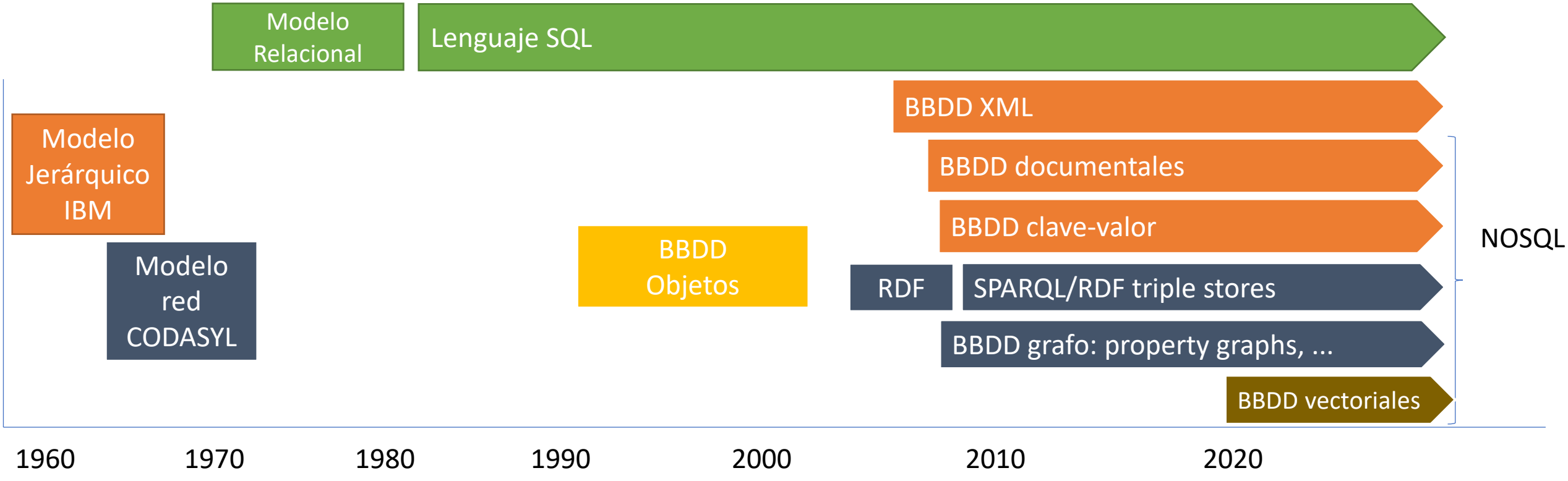
Evolución de bases de datos

Los primeros modelos eran jerárquicos y en red

Modelo relacional: Desde mediados 70

SQL no es el único modelo: modelos NOSQL

Reciente popularidad de bases de datos vectoriales



Clasificación tradicional en 2 tipos

Datos operacionales

- Utilizados para facilitar la operación de la empresa
- Ejemplos: datos de usuarios, inventario, transacciones, etc.
- Son necesarios para que la empresa pueda funcionar
- Operaciones habituales: Altas, Bajas, Modificaciones
  - A veces se utiliza el término CRUD (Create Retrieve Update Delete)
- Se asocia con OLTP (*OnLine Transaction Processing*) y SGBD

Datos analíticos

- Utilizados por científicos de datos y analistas de negocios
- Objetivo: predicciones, tendencias, inteligencia de negocio, ...
- No suele ser transaccional ni relacional
- Los datos no son críticos para el funcionamiento de día a día
- Se asocia con OLAP (*OnLine Analytical Processing*) y *data warehouses*

Clasificación de Bases de datos según modelo

Relacionales (SQL)

- Modelo basado en tablas, claves primarias y secundarias
- Ofrecen soporte para lenguaje SQL
- Ejemplos: MySQL, SQL Server, SQLite, Oracle, …

NoSQL

- Ideales para datos no estructurados en tablas
- Pueden escalar horizontalmente (clústeres de servidores)
- Varios modelos: Documentales, clave-valor, grafos, etc.
- Ejemplos: MongoDB, CouchDB, Neo4J, …

Vectoriales

- Almacenan vectores numéricos (representaciones texto, imágenes, vídeos,…)
- Aplicaciones para Búsqueda semántica, similaridad, IA generativa
- Aplicaciones con embeddings
- Ejemplo: Pinecone, Qdrant, Milvus, …

Clasificación según dónde están los datos

On Premises:

- Instaladas en servidores propios o de la organización
- La empresa controla infraestructura, seguridad, copias de respaldo
- Requieren mantenimiento técnico y hardware
- Ejemplos: Oracle, SQL Server, MySQL, PostgreSQL, SQLite (en dispositivo)

En la nube:

- Se accede por internet, proveedor gestiona infraestructura y mantenimiento
- Pago por uso (Database as a Service)
- Escalabilidad rápida y flexible
- Ejemplos: Amazon, Google Cloud, Microsoft Azure

Algunos datos sobre cuotas de mercado

Mercado global estimado en unos 78 mil millones de dólares en 2024

- BBDD relacionales ocupan 52% del mercado
- BBDD NoSQL muestran la mayor tasa de crecimiento (15-16%)
- BBDD como servicio pasó de 23 mil millones en 2023 y se espera que alcance los 104 mil millones en 2032

Fuente: https://www.emergenresearch.com/industry-report/dbms-market

Niveles y vistas de las bases de datos

Describe una parte de la base de datos
Simplificar interacción con el sistema
Objetivo: Usuario de bases de datos

Qué datos se almacenan y cómo se relacionan
Independencia de detalles físicos
Objetivo: Administradores de bases de datos

Describe cómo se almacenan los datos
Estructuras de datos complejas de bajo nivel
Objetivo: Desarrolladores de SGBD

Lenguajes de bases de datos

- Clasificación tradicional
  - Lenguaje de definición de datos (DDL)
    - Utilizado para definir la estructura o el esquema de la base de datos
    - Ejemplo. Los nombres de las columnas, los tipos de datos, etc.
  - Lenguaje de manipulación de datos (DML)
    - Consulta
    - Insertar/borrar/actualizar datos
- En bases de datos relacionales, SQL soporta las 2 modalidades

Bases de datos y Big data

Big data

- Datos que no pueden procesarse por medios tradicionales
- 3Vs: Volumen, Velocidad, Variedad

Logs de datos

- Registran la actividad que se realiza en la Base de datos
- Tradicionalmente, a los *logs* se les daba poca importancia
- Pero cada vez son más importantes para datos analíticos

Data warehousing

- "*Lagos*" de datos (*data lakes)*
- Bases de datos distribuidas

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

SQL

SQL - Structured Query Language

- Creado en IBM inicialmente durante años 70
- En 1986 fue estandarizado por ANSI
- Años 80-90 se popularizaron las implementaciones: Oracle, DB2, SQL Server, MySQL, …

Dispone de instrucciones para:

- Lenguaje definición datos:
  - Ejemplo de instrucciones: `CREATE TABLE`, `ALTER TABLE`, `DROP TABLE`, …
- Lenguaje manipulación datos
  - Inserción (`INSERT`), modificación (`UPDATE`), borrado (`DELETE`)
  - Consulta: `SELECT`

SQL ha alcanzado una gran popularidad

- Puede considerarse la *lingua franca* de *BBDD* relacionales

SQLite

En el curso utilizaremos SQLite para los ejercicios

Razones: es muy ligero y fácil de instalar

- Se puede invocar desde Python y Jupyter notebooks fácilmente

Descarga del software: https://sqlite.org/

Existen varias herramientas:

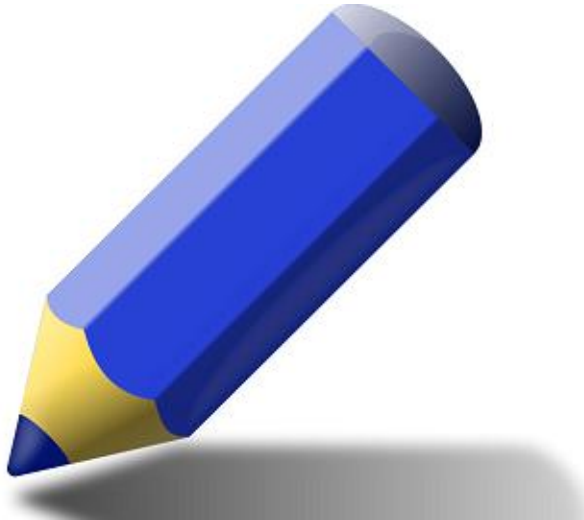
- DB Browser for SQLite: https://sqlitebrowser.org/
- SQLite Studio: https://sqlitestudio.pl/

Ejercicio:

Descargar SQLite localmente y crear una base de datos

Arrancar Jupyter notebook y ejecutar desde Google Colab

https://github.com/cursosLabra/introBBDD/



Sintaxis básica de SQL

SQL intenta alcanzar un compromiso entre:

- Lenguaje legible y natural para el ser humano
- Lenguaje no ambiguo y procesable por las máquinas

Sentencias tipo: SELECT, INSERT, etc.

- Palabras clave no distinguen minúsculas/mayúsculas
- Cadenas de texto entre comillas
- Cada instrucción finaliza en punto y coma (;)
- Trabaja con conjuntos de datos (no registro a registro)

Ejemplo de comando:

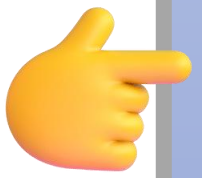
```
SELECT Apellidos, Nombre
FROM   Alumnos
WHERE Nombre = 'Ana’ AND Nota >= 9;
```

Curiosidad:
Un antecesor de SQL, era SEQUEL:
Structure English Query Language

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

Diseño conceptual de bases de datos relacionales



- Diagramas Entidad-Relación y modelo relacional
- SQL como lenguaje de definición de datos (DDL)
  - Creación de tablas
- SQL como lenguaje de manipulación de datos (DML)
  - Inserción, modificación y borrado de datos
- SQL como lenguaje de consulta (DQL)
  - Selección de valores

Diagramas entidad-relación

Propuestos por Peter P. Chen en 1970

Entidad:

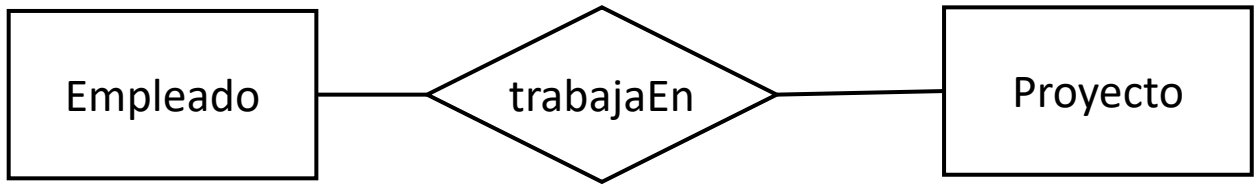
- Cualquier objeto o concepto
- Se representan por cajas

Relación:

- Correspondencia o asociación entre entidades
- Se representan con un rombo
- Grado: número de entidades que relacionan

Peter Chen

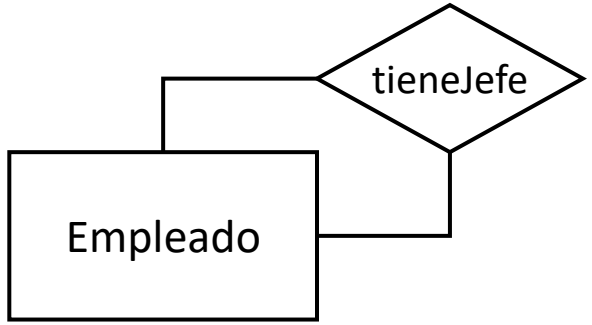
Fuente: https://www.csc.lsu.edu/~chen/



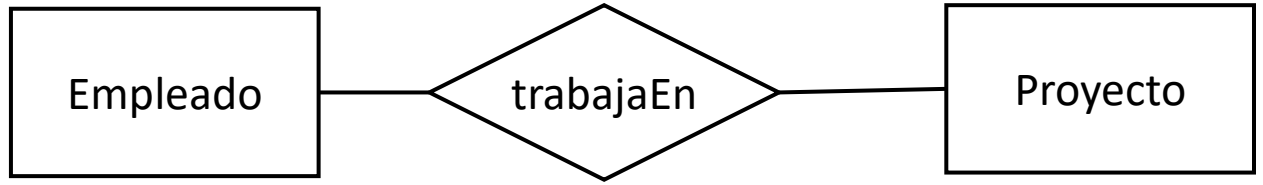
Grados de una relación

Grado = número de entidades que intervienen en una relación

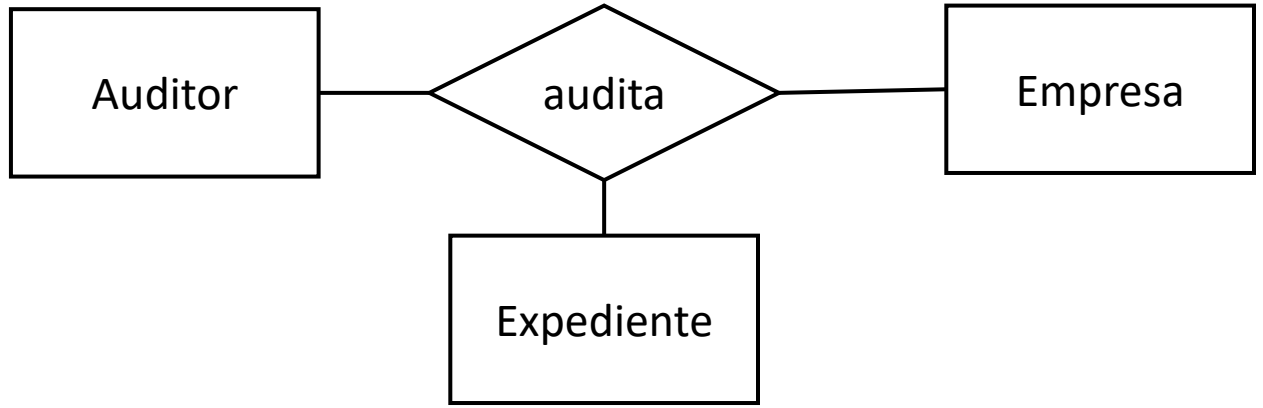
Relación reflexiva: grado 1



Relación binaria: grado 2



Relación ternaria: grado 3



Cardinalidad

Anotar el número de elementos mínimo y máximo que intervienen

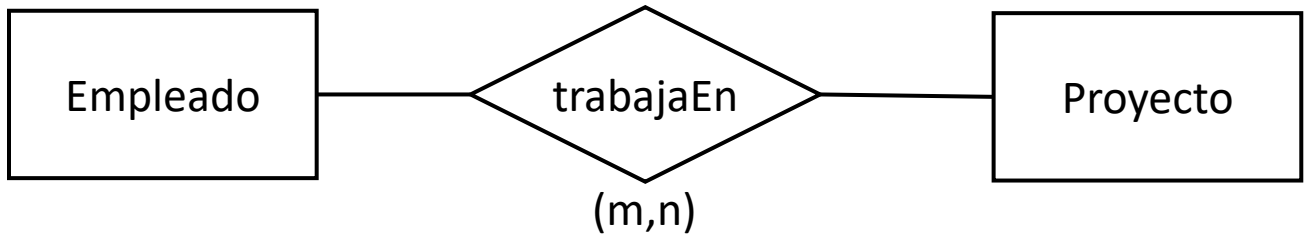
(1,1) = exactamente 1

(0,1) = opcional

(0,n) = 0 ó más

(1,n) = 1 ó más (al menos 1)

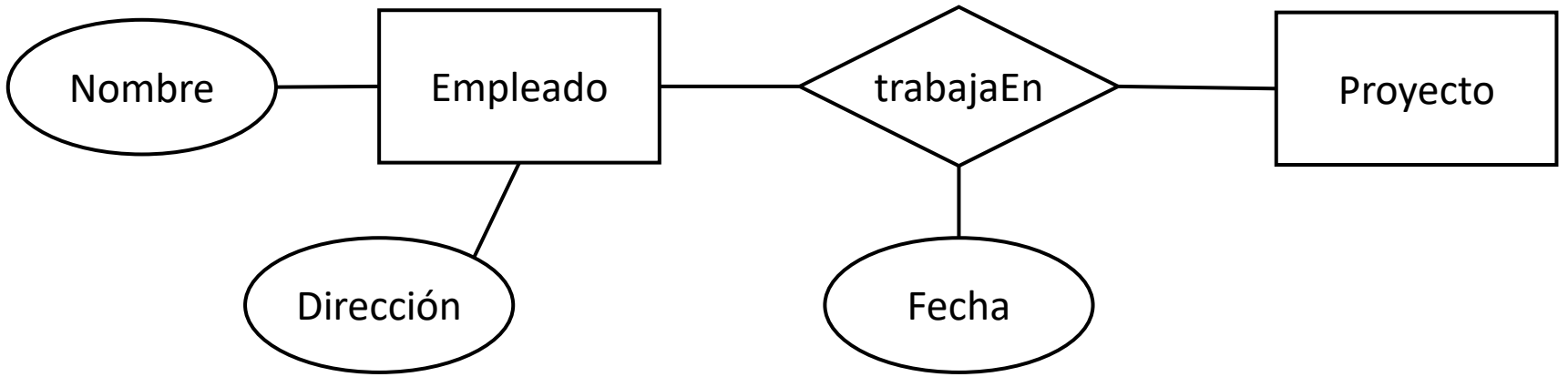
(m,n) = muchos a muchos



Atributos

Atributo = característica o propiedad de una entidad o relación

Pueden representarse mediante elipses



Dominios

Un dominio representa conjunto de valores permitidos para un atributo

Por ejemplo:

- Un número entero, una fecha, una cadena de texto, un valor de una lista, ...

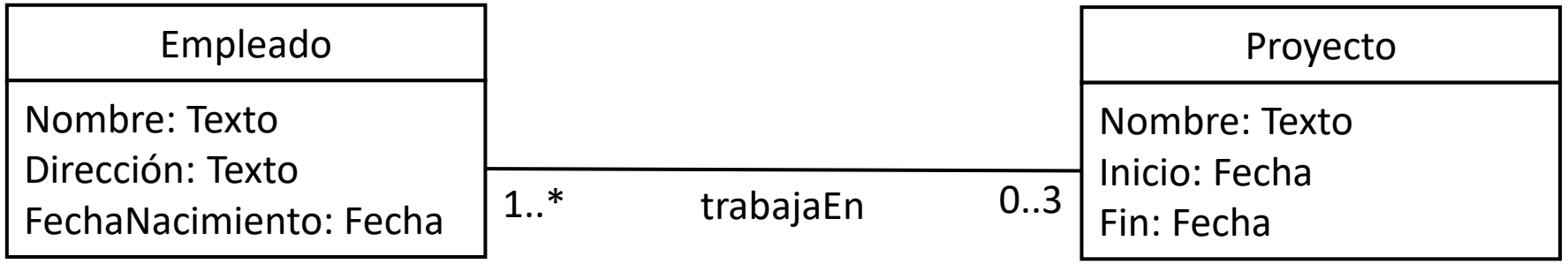
Dominio atómico: cuando es un único valor

- Ejemplo:
  - El dni de una persona es un único valor
  - El número de teléfono podría ser atómico si cada persona tiene un único número
    - No siempre se cumple en la práctica

Notación UML para entidades

UML = Unified Modelling Language

- Lenguaje de modelado utilizado para diseñar sistemas
- Varios tipos de diagramas
- Los diagramas de clases pueden utilizarse para representar relaciones

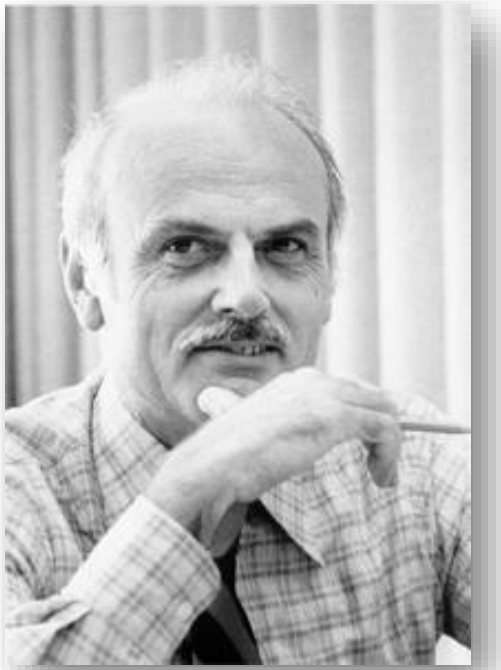


Modelo relacional

Propuesto por Edgar F. Codd a principios de 1970

Se basa en tablas que definen relaciones

Define un álgebra relacional

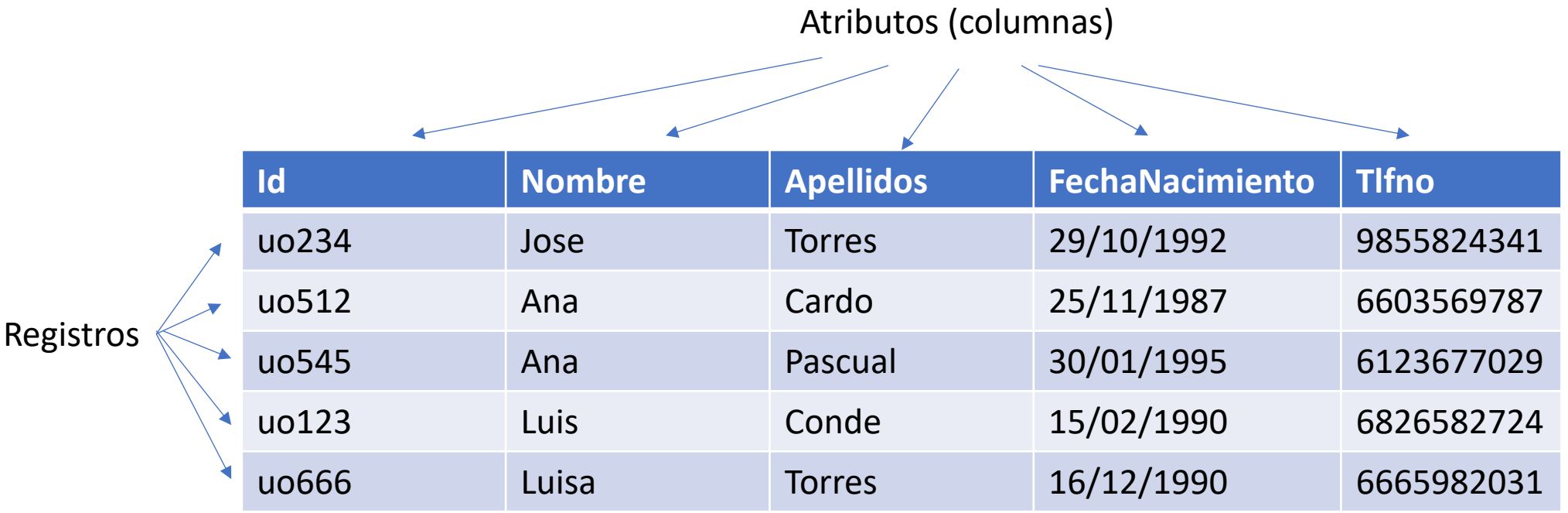


Edgar F. Codd

Fuente: Wikipedia

Tablas

Una tabla representa una relación



Id	Nombre	Apellidos	FechaNacimiento	Tlfno
uo234	Jose	Torres	29/10/1992	9855824341
uo512	Ana	Cardo	25/11/1987	6603569787
uo545	Ana	Pascual	30/01/1995	6123677029
uo123	Luis	Conde	15/02/1990	6826582724
uo666	Luisa	Torres	16/12/1990	6665982031

(*) Nota: Los datos personales de la tabla son ficticios. Cualquier coincidencia con la realidad es mera casualidad

Tipos de datos

Los valores internos de las celdas pueden ser:

Nombre	Descripción	Ejemplo
Text	Cadena de texto	'Juan'
Integer	Número entero sin decimales	365
Real	Número con decimales	3.1415
Blob (Binary Large Object)	Valor binario	
Null	Valor nulo	Null

Otras bases de datos manejan otros tipos como:

Boolean, Date, Time, DateTime, VarChar, ...

SQLite los convierte a los valores primitivos

Más información: Datatypes In SQLite

Clave primaria

Valor (o conjunto de valores) que permite identificar un registro

- Puede ser un valor único, ej. Id
- O combinado, ej. Nombre + apellidos

Id	Nombre	Apellidos	FechaNacimiento	Tlfno
uo234	Jose	Torres	29/10/1992	9855824341
uo512	Ana	Cardo	25/11/1987	6603569787
uo545	Ana	Pascual	30/01/1995	6123677029
uo123	Luis	Conde	15/02/1990	6826582724
uo666	Luisa	Torres	16/12/1990	6665982031

Clave externa o foránea

Se utilizan para hacer referencia a claves de otras tablas

AlumnoId	Nota
uo666	7.8
uo123	3
uo545	10

Id	Nombre	Apellidos	FechaNacimiento	Tlfno
uo234	Jose	Torres	29/10/1992	9855824341
uo512	Ana	Cardo	25/11/1987	6603569787
uo545	Ana	Pascual	30/01/1995	6123677029
uo123	Luis	Conde	15/02/1990	6826582724
uo666	Luisa	Torres	16/12/1990	6665982031

Restricciones

Restricciones de integridad

Cardinalidad:

- Ejemplos:
  - Una persona solamente puede tener un DNI
  - No puede haber más de una persona con el mismo DNI
  - Un empleado puede tener entre 1 y 3 proyectos asociados
  - . . .

Diseño de bases de datos relacionales

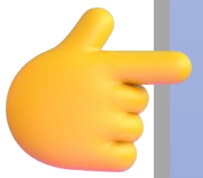
Definir esquemas de Bases de datos

Concepto de normalización:

- Normalización: Dividir en varias tablas para evitar duplicidades
- Denormalización: Juntar en una tabla para mejorar rendimiento

Solución de compromiso: Mantenimiento vs rendimiento

Diseño conceptual de bases de datos relacionales



Diagramas Entidad-Relación y modelo relacional

SQL como lenguaje de definición de datos (DDL)

- Creación de tablas

SQL como lenguaje de manipulación de datos (DML)

- Inserción, modificación y borrado de datos

SQL como lenguaje de consulta (DQL)

- Selección de valores

Creación de tablas

CREATE permite crear tablas

```
CREATE TABLE NombreTabla (col1 Tipo1,...colN TipoN)
```

Ejemplo:

```
CREATE TABLE alumnos (
    Id                TEXT,
    Nombre            TEXT,
    Apellidos         TEXT,
    FechaNacimiento Date,
    Tlfno             TEXT
)
```

Id	Nombre	Apellidos	FechaNacimiento	Tlfno

NOTA: SQLLite añade automáticamente una columna denominada ROWID

Restricciones en tablas

Es posible añadir restricciones en los valores de las tablas

```
CREATE TABLE Contactos (
    id integer primary key,
    nombre text not null collate nocase,
    email  text not null unique,
    tlfno  text not null default 'UNKNOWN',
    edad   integer check (edad >= 0 and edad <= 120),
    unique (nombre, tlfno)
);
```

Definiciones de claves

```sql
CREATE TABLE Alumnos (
    Id INTEGER PRIMARY KEY,
    Nombre TEXT NOT NULL,
    Apellidos TEXT NOT NULL,
    FechaNacimiento DATE NOT NULL
);
```

```sql
CREATE TABLE Cursos (
    Id INTEGER PRIMARY KEY,
    Nombre TEXT NOT NULL,
    Descripcion TEXT
);
```

```sql
CREATE TABLE Notas (
    Id INTEGER PRIMARY KEY,
    AlumnoId INTEGER,
    CursoId INTEGER,
    Nota REAL CHECK(Nota >= 0 AND Nota <= 10),
    FOREIGN KEY (AlumnoId) REFERENCES Alumnos(Id),
    FOREIGN KEY (CursoId) REFERENCES Cursos(Id)
);
```

Modificación de tablas (ALTER)

Permite modificar una tabla

Cambiar de nombre:

```
ALTER TABLE alumnos     RENAME TO Estudiantes;
```

Añadir/renombrar/borrar columnas

```
ALTER TABLE Estudiantes ADD COLUMN Email Text;
ALTER TABLE Estudiantes RENAME COLUMN Email TO Correo;
ALTER TABLE Estudiantes DROP COLUMN Correo;
```

Eliminación de tablas (DROP)

Borrar una tabla

Ejemplo:

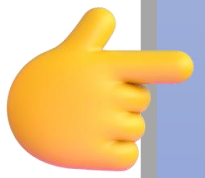
```
DROP TABLE Estudiantes;
```

Diseño conceptual de bases de datos relacionales

Diagramas Entidad-Relación y modelo relacional

SQL como lenguaje de definición de datos (DDL)

- Creación de tablas



SQL como lenguaje de manipulación de datos (DML)

- Inserción, modificación y borrado de datos

SQL como lenguaje de consulta (DQL)

- Selección de valores

Inserción de registros

INSERT permite añadir registros a una tabla

```
INSERT INTO nombreTabla (nombreCol [, ...])
VALUES (nuevoValor [, ...]);
```

Ejemplo:

```
INSERT INTO alumnos (Id, Nombre, Apellidos, FechaNacimiento)
VALUES ("uo234", "Jose", "Torres",  "29/10/1992"),
       ("uo512", "Ana",  "Cardo",   "25/11/1987"),
       ("uo545", "Ana",  "Pascual", "30/01/1995");
```

Borrado de registros

DELETE permite borrar registros de una tabla

```
DELETE FROM table_name WHERE expression
```

Ejemplo:

```
DELETE FROM alumnos WHERE FechaNacimiento = "29/10/1993";
```

Actualización de registros

UPDATE permite actualizar valores de una tabla

```
UPDATE nombreTabla SET nombreCol=nuevoValor [, ...]
WHERE  expresión
```

Ejemplo:

```
UPDATE alumnos SET FechaNacimiento = "29/10/1993"
WHERE  Id = "uo234";
```

Diseño conceptual de bases de datos relacionales

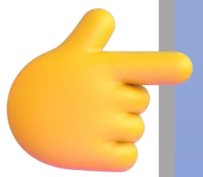
Diagramas Entidad-Relación y modelo relacional

SQL como lenguaje de definición de datos (DDL)

- Creación de tablas

SQL como lenguaje de manipulación de datos (DML)

- Inserción, modificación y borrado de datos



SQL como lenguaje de consulta (DQL)

- Selección de valores

Consultas

SELECT permite realizar consultas a la Base de datos

- Es un comando con una gran expresividad
- Permite combinar datos entre diferentes tablas

Formato general:

```
SELECT [DISTINCT] cabecera
FROM tablas
WHERE expresión
GROUP BY expresión
HAVING expresión
ORDER BY expresión
LIMIT valor OFFSET valor
```

Ejemplo

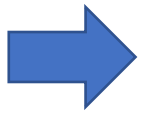
Ejemplo:

```
SELECT Apellidos, Nombre
FROM Alumnos
WHERE Nombre = 'Ana';
```

La cabecera puede contener:
- Una lista de columnas
- * para seleccionar todas las columnas
- Expresiones

A cada columna se puede asociar un alias mediante AS
Se puede usar SELECT DISTINCT para eliminar duplicados

Id	Nombre	Apellidos	FechaNacimiento
uo234	Jose	Torres	29/10/1992
uo512	Ana	Cardo	25/11/1987
uo545	Ana	Pascual	30/01/1995
uo123	Luis	Conde	15/02/1990
uo666	Luisa	Torres	16/12/1990



Apellidos	Nombre
Cardo	Ana
Pascual	Ana

ORDER BY

Permite clasificar las filas del resultado

Formato:

```
ORDER BY expresión [ASC|DESC] [,...]
```

La expresión suele ser una columna, pero puede ser más compleja

ASC/DESC indica orden ascendente/descentente (ASC por defecto)

LIMIT y OFFSET

Permiten extraer subconjuntos específicos de filas

LIMIT define el número máximo de filas a extraer

OFFSET define el número de filas a saltar antes de extraer la primera

```
LIMIT 10           -- devuelve las 10 primeras filas
LIMIT 10 OFFSET 3 -- devuelve las filas 4 a 13
LIMIT 3 OFFSET 20 -- devuelve las filas 21 a 23
```

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

Proyección

En la cabecera se indica qué columnas se muestran en resultado

La cabecera puede contener:

- Una lista de columnas
- * para seleccionar todas las columnas
- Expresiones
  - A cada columna se puede asociar un alias mediante AS

Se puede usar SELECT DISTINCT para eliminar duplicados

```
SELECT Apellidos, Nombre
FROM   Alumnos
```

```
SELECT *
FROM   Alumnos
```

```
SELECT FechaNacimiento AS Nacimiento,
       Nombre || ' ' || Apellidos AS NombreCompleto
FROM   Alumnos
```

```
SELECT DISTINCT Nombre
FROM  alumnos
```

Selección (WHERE)

Define una condición que permite filtrar registros en tabla de resultados

Se proporciona una expresión que se evalúa para cada fila

Las filas cuyo resultado sea falso o NULL son descartadas

Expresiones que pueden incluirse:

- Expresiones lógicas encadenadas mediante AND, OR, NOT
- Expresiones de comparación: <, >, >=, >=, =, !=
- Operadores aritméticos: +, - *, /, %
- Otras operaciones: ISNULL, NOT NULL, IS DISTINCT FROM, CASE, IN, NOT IN, ...
- Funciones: https://www.sqlite.org/lang_corefunc.html

Selección (WHERE)

Ejemplos:

```
SELECT Nombre, Apellidos, FechaNacimiento
 FROM  alumnos
 WHERE FechaNacimiento > '2000-01-01' OR FechaNacimiento < '1990-31-12'
```

```
SELECT Nombre, Apellidos, FechaNacimiento
 FROM  alumnos
 WHERE length(Apellidos) > 6
```

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

FROM

Permite identificar las tablas desde las que se obtendrán los valores

Puede contener:

- Un nombre de tabla
- Un Alias para el nombre de la tabla
- Varias tablas mediante JOIN
  - 3 tipos de JOIN: Interno, Externo

Explicando los joins con piezas Lego:

https://www.youtube.com/shorts/RiCsbGKKymU

CROSS JOIN (unión cruzada/producto cartesiano)

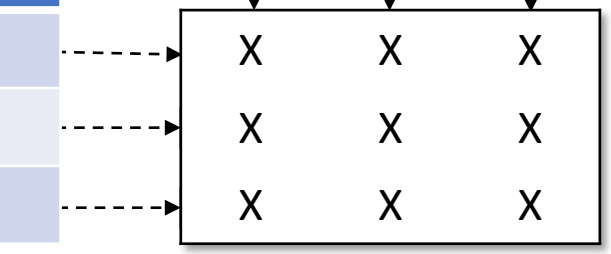
Nombres

Id	Nombre
uo234	Jose
uo512	Ana
uo545	Luis

Valores

Id	Nota
uo234	7.8
uo545	10
uo666	3

```
SELECT *
FROM Nombres CROSS JOIN Valores;
```



Id	Nombre	Id	Nota
uo234	Jose	uo234	7.8
uo512	Ana	uo234	7.8
uo545	Luis	uo234	7.8
uo234	Jose	uo545	10
uo512	Ana	uo545	10
uo545	Luis	uo545	10
uo234	Jose	uo666	3
uo512	Ana	uo666	3
uo545	Luis	uo666	3

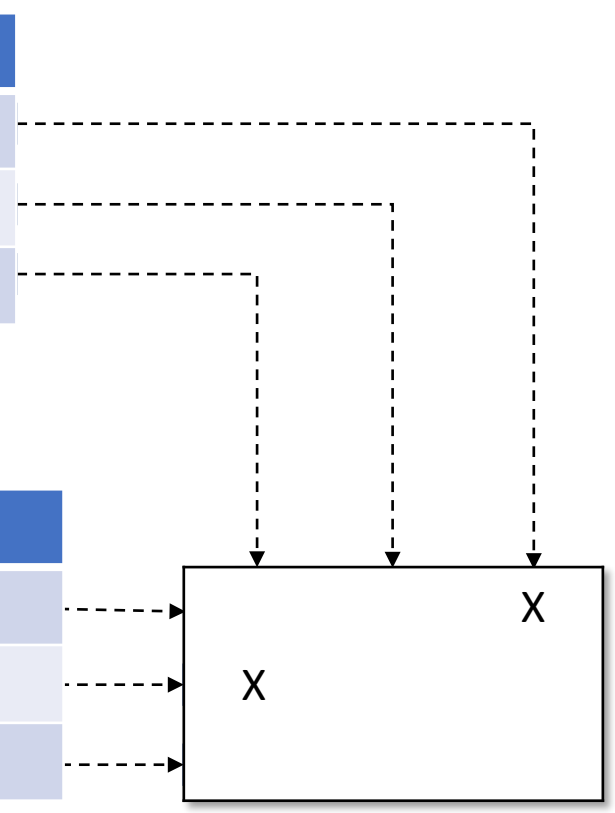
INNER JOIN (Unión natural)

Nombres

Id	Nombre
uo234	Jose
uo512	Ana
uo545	Luis

Valores

Id	Nota
uo234	7.8
uo545	10
uo666	3




```
SELECT Nombres.Id, Nombre, Nota
FROM Nombres INNER JOIN Valores
ON Nombres.Id = Valores.Id ;
```

Id	Nombre	Nota
uo234	Jose	7.8
uo545	Luis	10

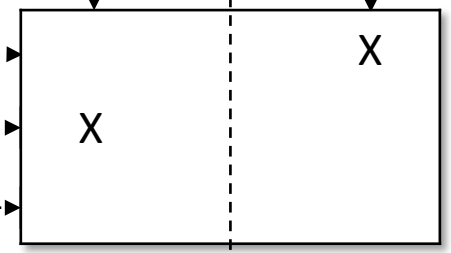
OUTER JOIN (Unión externa)

Nombres

Id	Nombre
uo234	Jose
uo512	Ana
uo545	Luis

Valores

Id	Nota
uo234	7.8
uo545	10
uo666	3




```
SELECT Nombres.Id, Nombre, Nota
FROM Nombres LEFT OUTER JOIN Valores
ON Nombres.Id = Valores.Id ;
```

Id	Nombre	Nota
uo234	Jose	7.8
uo545	Luis	10
uo512	Ana	Null

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

GROUP BY <expresión>

Permite agrupar filas

Objetivo: contar filas, sumar valores, calcular valor medio, mínimo, etc.

Nombre	Nota	Curso
Jose	7.8	Lógica
Eva	9	Álgebra
Luis	10	Lógica
Ana	4	Álgebra
Luis	7	Álgebra
Jose	6	Álgebra

```
SELECT curso, count(*) AS Num
FROM Notas
GROUP BY Curso;
```

Curso	Num
Lógica	2
Álgebra	4

GROUP BY <expresión>

Otras funciones de agregación: AVG, MIN, MAX,...

Nombre	Nota	Curso
Jose	7.8	Lógica
Eva	9	Álgebra
Luis	10	Lógica
Ana	4	Álgebra
Luis	7	Álgebra
Jose	6	Álgebra

```
SELECT
 curso,
 avg(Nota) AS Media,
 min(Nota) AS Mín,
 max(Nota) AS Máx
FROM Notas
GROUP BY Curso;
```

Curso	Media	Mín	Máx
Lógica	8.9	7.8	10.0
Álgebra	6.5	4.0	9.0

HAVING <Expresión>

Se utiliza para filtrar resultados después de un GROUP BY

Es parecido a WHERE

Diferencia:

HAVING aparece después de GROUP BY

HAVING puede hacer referencia a variables de la cabecera que usan agregados

Nombre	Nota	Curso
Jose	7.8	Lógica
Eva	9	Álgebra
Luis	10	Lógica
Ana	4	Álgebra
Luis	7	Álgebra
Jose	6	Álgebra

```
SELECT curso, avg(Nota) as Media
FROM Notas
GROUP BY Curso
Having Media > 8;
```

Curso	Media
Lógica	8.9

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

Subconsultas

Se pueden incluir consultas SELECT dentro de consultas SELECT

Puede ser útil para utilizar valores auxiliares

```
SELECT N.Id, N.Nota,
    (SELECT AVG(N2.Nota) FROM Valores N2) AS Nota_Media,
    N.Nota - (SELECT AVG(N3.Nota) FROM Valores N3) AS Desv
FROM Valores N
```

Valores

Id	Nota
uo234	7.8
uo545	10
uo666	3

Id	Nota	Nota_Media	Desv
uo234	7.8	6.93	0.86
uo545	10	6.93	3.06
uo512	3	6.93	-3.93

Versión más eficiente

```
SELECT N.Id, N.Nota,
       Stats.Nota_Media,
       N.Nota - Stats.Nota_Media AS Desv
FROM Valores N, (SELECT AVG(Nota) AS Nota_Media FROM Valores) Stats
```

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

Transacciones

Transacción = bloque de operaciones que se ejecutan como una unidad atómica

Es decir: O se ejecutan todas, o no se ejecuta ninguna

Propiedades ACID

- Atomicidad: todas las operaciones se aplican, o ninguna
- **C**onsistencia: la base datos pasa de un estado válido a un estado válido
- A**i**slamiento: cambios intermedios no afectan a otras operaciones concurrentes
- **D**urabilidad: los cambios confirmados se mantienen incluso si hay un fallo de energía

Comandos de transacciones

BEGIN [Transaction]

COMMIT

ROLLBACK

```
CREATE TABLE alumnos (nombre TEXT,nota REAL);
INSERT INTO alumnos VALUES ('Luis', 6);

BEGIN TRANSACTION;
INSERT INTO alumnos VALUES ('Juan', 7.5);
INSERT INTO alumnos VALUES ('Mara', 12);
ROLLBACK;
```

Nombre	Nota
Luis	6

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

Optimización de consultas y rendimiento

Normalización de bases de datos

Creación de índices

Planes de ejecución

- Comando EXPLAIN

Reconstrucción de la base de datos

- VACUUM

Actualización de estadísticas

Contenidos

- Introducción a las bases de datos
  - Importancia y tecnología en negocios
  - Tipos de bases de datos y sistemas de gestión de bases de datos
  - Introducción a SQL y su sintaxis básica
- Lenguaje SQL
  - Diseño conceptual de bases de datos relacionales
  - Selección y proyección
  - Uniones
  - Funciones agregadas y agrupamiento de datos
  - Subconsultas
  - Transacciones
  - Optimización de consultas y técnicas de rendimiento
  - Manejo de errores y excepciones

Manejo de errores y excepciones

Violaciones de restricciones

Uso de disparadores (*triggers*)

Tabla con log de errores mediante triggers

Copias de seguridad

Más allá de SQL

Otros modelos de datos

Bases de datos de grafo

RDF

SPARQL y RDF Triple Stores

Wikidata

RDF y Web Semántica

RDF surgió en 1998

Marco para describir recursos

Se basa en tripletas: Sujeto - Predicado - Objeto

Objetivo: Web Semántica

URIs para predicados

SPARQL

Lenguaje de consultas para RDF

Similar a SQL, pero para RDF

Recomendación W3C

Wikidata

Grafo de conocimiento que da soporte a Wikipedia

Capturar conocimiento de la humanidad de forma estructurada

Fin de la presentación