A woman with brown hair, wearing a red top, is holding a yellow pear in her left hand and a green apple in her right hand. She has a confused or questioning expression on her face, looking slightly to the side. The background is plain white.

**Análisis de enfoques:
¿cómo comparar peras con manzanas?**

Bases de Datos No Relacionales

Instituto de Computación, FING, Udelar – 2023

CC-BY Lorena Etcheverry lorenae@fing.edu.uy

Preliminares

Una **Base de Datos (BD)** es una colección organizada de datos.

Los Sistemas de Gestión de Bases de Datos (DBMS) son **programas** que permiten el almacenamiento, modificación y extracción de datos de la BD.

Podemos clasificar y caracterizar a los DBMS según varios aspectos.

Algunos aspectos para clasificar sistemas de gestión de datos

Modelos de datos

Lenguajes de consulta e
interfases

Modelos de consistencia y
soporte a transacciones

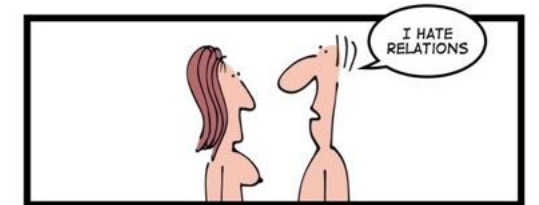
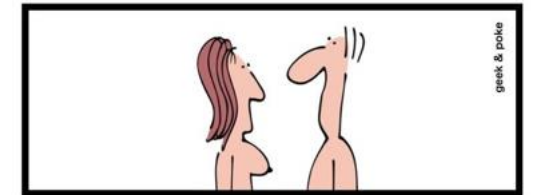
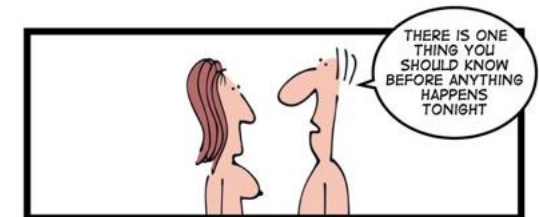
Arquitectura de datos y de
cómputo

Modelos de datos

Son lenguajes usados para especificar y manipular BDs y permiten expresar:

- **Estructuras:** Elementos de los problemas.
- **Restricciones:** Reglas que deben cumplir los datos para que la BD sea considerada válida.
- **Operaciones:** Insertar, borrar y consultar la BD.

The Hard Life of a NoSQL Coder



Part 1: The Outing

Modelos de datos (2)

Los modelos de datos pueden tener diferentes niveles de abstracción.

- **Modelos conceptuales:** Representan la realidad **independientemente** de la implementación de la BD
- **Modelos lógicos:** abstracción que implementa el DBMS
- **Modelos físicos:** estructuras de datos que implementa y usa el DBMS para almacenar los datos

Ejemplos en el caso relacional

Modelo conceptual: modelo Entidad-Relación

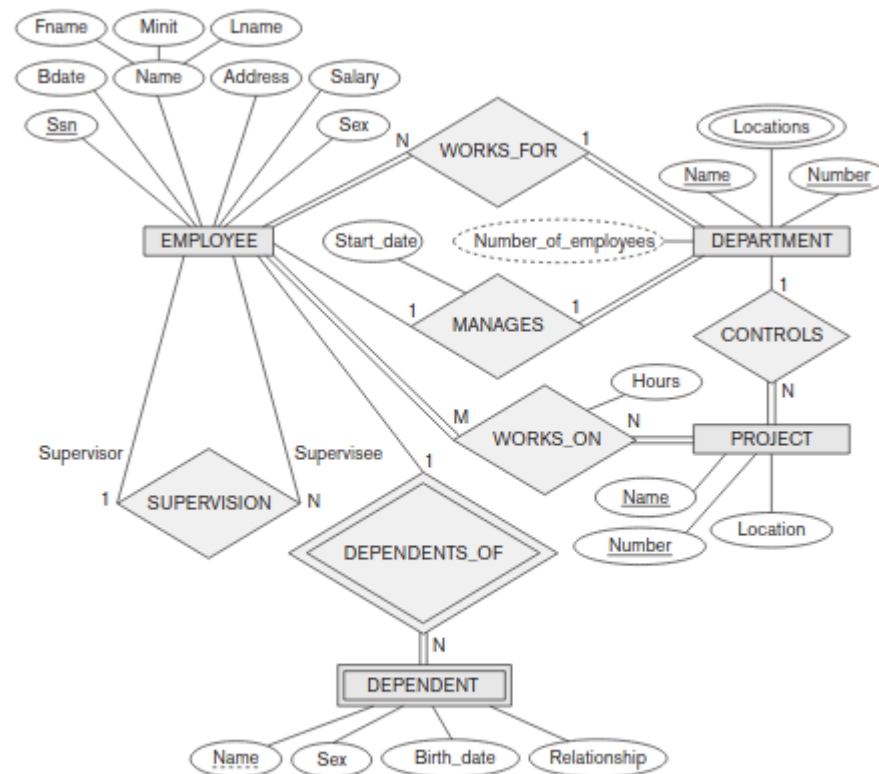


Figure 3.2

An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter and is summarized in Figure 3.14.

Ejemplos en el caso relacional

Modelo lógico: modelo relacional (tablas, restricciones, etc)

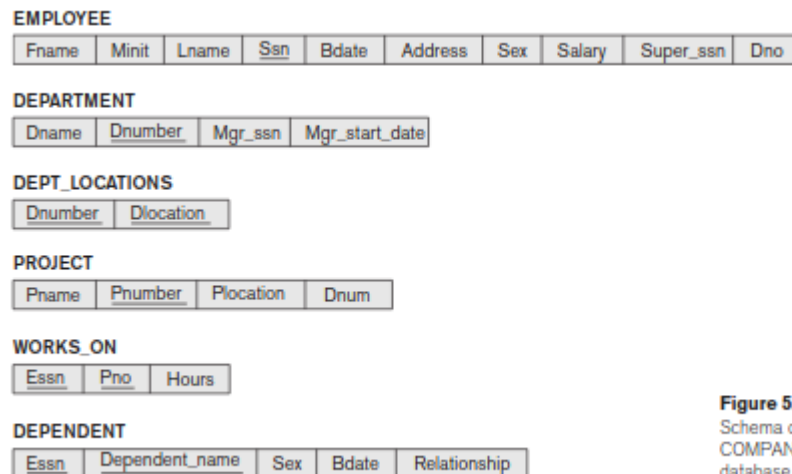


Figure 5.5
Schema diagram for the
COMPANY relational
database schema.

Ejemplos en el caso relacional

Modelo físico: registros, árboles B, tablas de hash

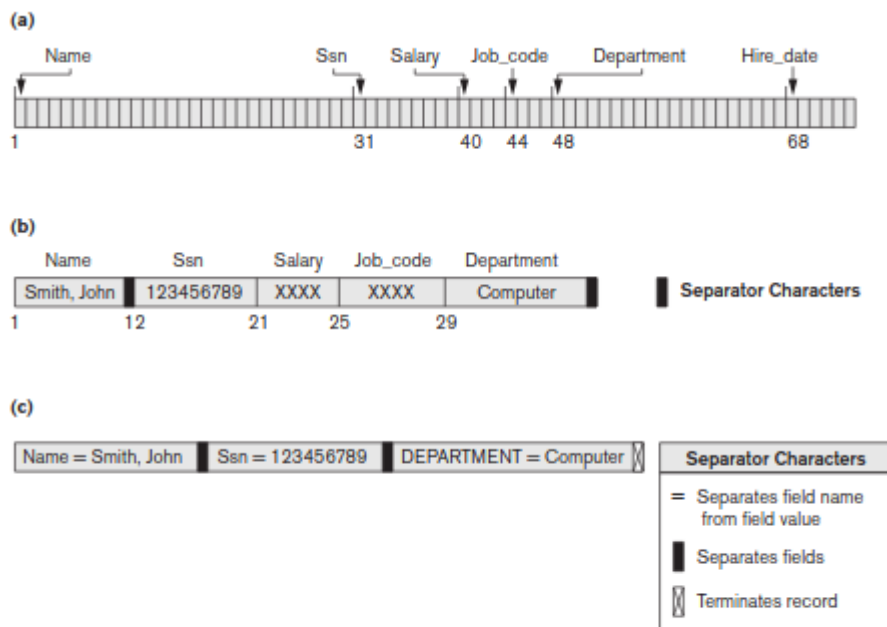


Figure 16.5
Three record storage formats. (a) A fixed-length record with six fields and size of 71 bytes. (b) A record with two variable-length fields and three fixed-length fields. (c) A variable-field record with three types of separator characters.

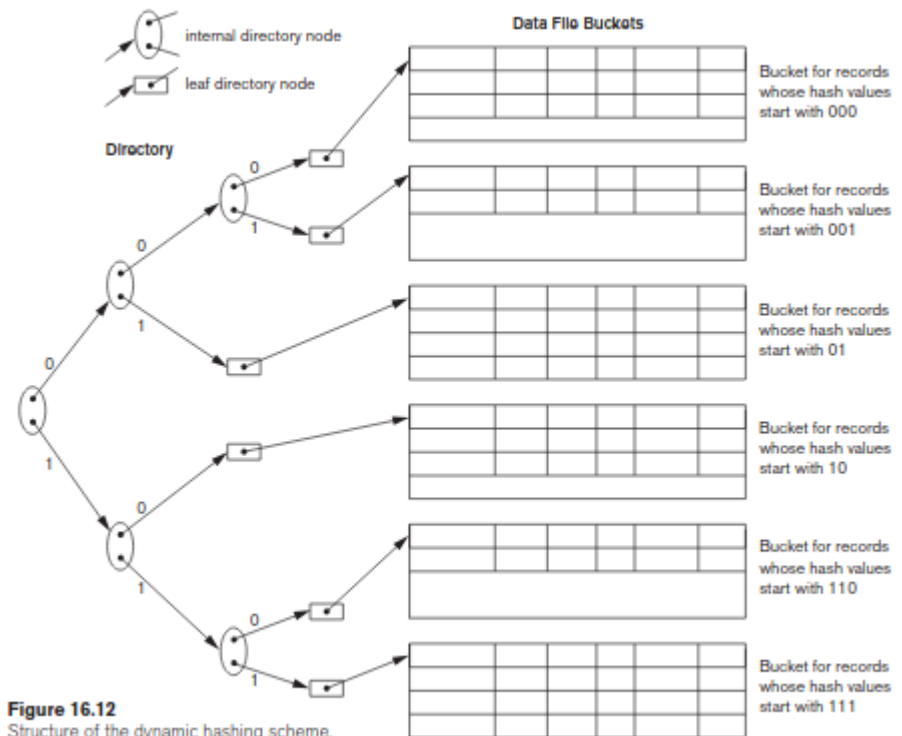


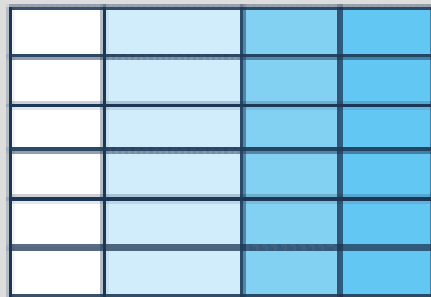
Figure 16.12
Structure of the dynamic hashing scheme.

¿y que pasa en el caso no relacional?

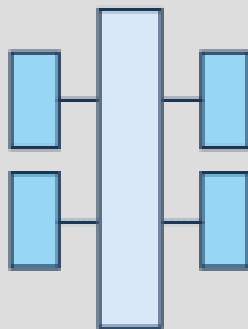
- Poco foco en el modelado conceptual 😬
- Muchos modelos lógicos, pero no siempre bien especificados: (ej el modelo **documental**, el modelo de **property graphs** para bases de datos de grafos)
- Gran diversidad en las implementaciones (modelos físicos)

SQL

Relational

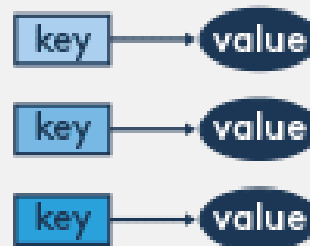


Analytical (OLAP)

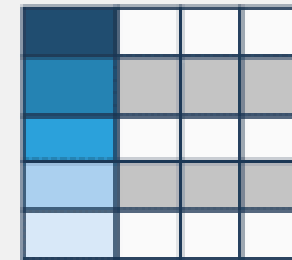


NoSQL

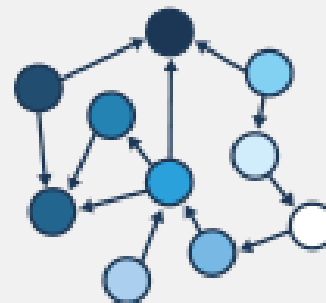
Key-Value



Column-Family



Graph



Document



Lenguajes de consulta e interfaces

En los RDBMS

- SQL es un estándar desde 1982
- Cada fabricante de RDBMS implementa variantes, pero las diferencias son sutiles.
- Casi todos los lenguajes de programación tienen bibliotecas y abstracciones (ej: JDBC en Java)

Query 0. Retrieve the birth date and address of the employee(s) whose name is 'John B. Smith'.

```
Q0:  SELECT  Bdate, Address
      FROM    EMPLOYEE
      WHERE   Fname = 'John' AND Minit = 'B' AND Lname = 'Smith';
```

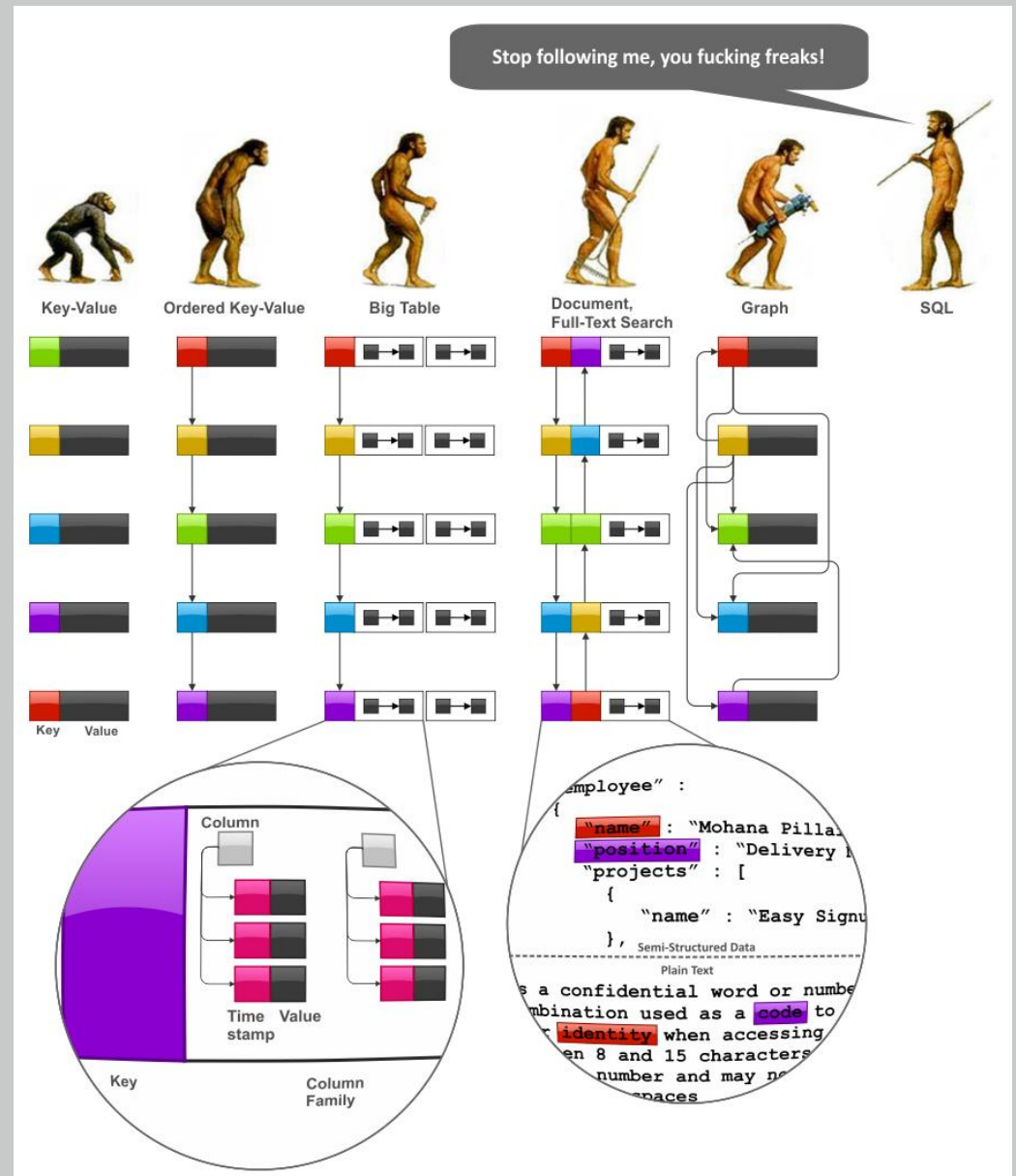
En los sistemas no relacionales hay mucha **diversidad**:

APIs y lenguajes de consulta

falta de estándares

Muchos brindan posibilidad de ejecutar consultas SQL-like

LA PORTABILIDAD ENTRE PRODUCTOS ES MUY DIFICIL



Consistencia y soporte a transacciones

Consistencia y transacciones en RDBMS

La noción de consistencia estricta implica que toda instancia de la BD satisface una serie de **restricciones** (ej: restricciones de clave primaria, clave foránea, etc)

Las operaciones sobre la BD garantizan que van de un estado consistente a otro estado consistente.

Esto se implementa con **transacciones** (soporte a acceso **concurrente**)

¿qué pasa en los sistemas noSQL?

Se define una **nueva noción de consistencia**.

Teorema CAP

Diferentes enfoques plantean estrategias diferentes:

- Consistencia estricta (típicamente via locks)
- *Eventual consistency* [1]
- Consistencia ajustable (ej: Dynamo DB)
 - En algunos casos puede requerir de mecanismos de resolución de conflictos

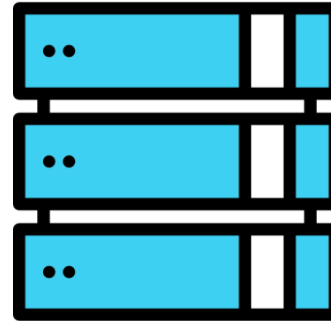
[1] Vogels, W. (2009). [Eventually consistent](#). Communications of the ACM, 52(1), 40-44.

Arquitectura de datos y de cómputo

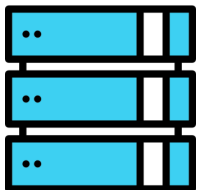
Estrategias para escalar



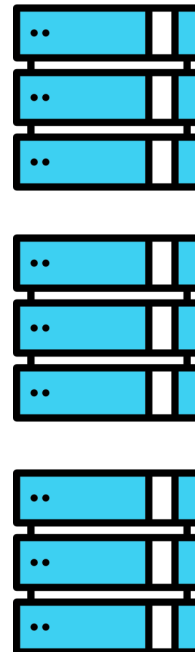
scale up



La misma solución en una máquina más grande



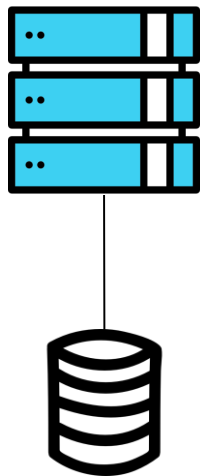
scale out



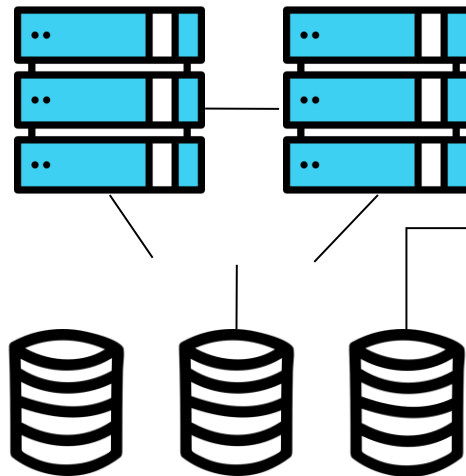
Una nueva solución que **distribuye** el cómputo, los datos, o ambos en varias máquinas

Scale out en RDBMs

Arquitectura *share everything*

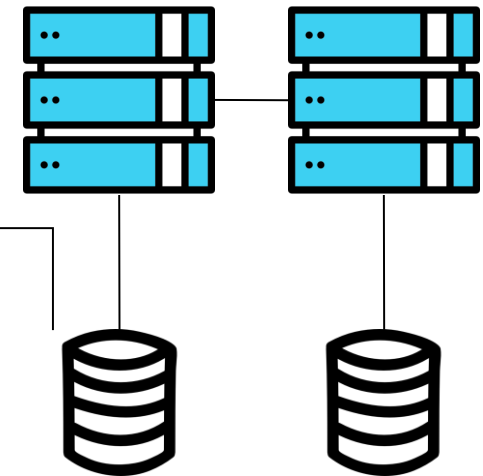


Arquitectura *share disk*



EJ: Oracle RAC

Arquitectura *share nothing*



EJ: sharded MySQL

Algunos problemas difíciles de gestionar

- *Shared-disk:*

- Gestión de los caches de cada nodo (*cache coherency*)

- *Shared-nothing:*

- ACID sobre varios nodos (2PC)
- Balance de carga (datos y cómputo)
- La caída de un nodo implica pérdida de datos (salvo q tenga redundancia)

¿qué pasa en los sistemas noSQL?

- Disponibilidad vs Consistencia
 - Teorema CAP
- Uso de *commodity-hardware*
 - Decenas de miles de nodos!
- Necesidad de resiliencia:
 - Los nodos fallan, el sistema debe recuperarse

¿qué pasa en los sistemas noSQL?

Hay al menos tres familias de estrategias para el particionamiento de datos

- *Sharding* por clave (y variantes) (ej: MongoDB)
- Un master que sabe donde colocar los datos (ej: Hadoop, HDFS y Hbase)
- Distribución via funciones de hash (ej: Amazon Dynamo)

Resumiendo

- Decidir cual es la estrategia adecuada para cada caso no es sencillo!!!
- Hay varios factores que inciden:
 - ¿qué modelo de datos se ajusta a cada problema?
 - Formas de interacción sobre los mismos, portabilidad, capacidades de los equipos de desarrollo
 - Características de los datos (volumen, naturaleza, volatilidad, etc)
 - Arquitecturas y formas de despliegue (on-premises, on-cloud, etc.)
 - \$\$\$