BASES DE DATOS PARA IA/DATA SCIENCE

PABLO OCTAVIANO

AGENDA

- Presentación
- Programa
- Introducción a Bases de datos
- Tipos de Bases de datos
- Estructuras/Organización de Datos

Presentación

- Ing Pablo Octaviano
  - Data Scientist/Data Specialist en ICBC
  - Contacto:
    - Email: octaviano.pjavier@gmail.com
    - Tel: 3814662993

Programa

1. Introducción a bases de datos. Tipos de base de datos. Estructuras de datos.
2. Diagramas de entidad relación. Interpretación de los diagramas y construcción. Introducción lenguaje SQL
3. DER y normalización de datos. Construcción de queries.
4. Sistemas NoSQL. Tipos y utilización. Querying en NoSQL.
5. Implementaciones cloud. Principales vendors. Arquitecturas típicas y soluciones híbridas orientadas a ML.
6. Bases de datos de vectores, principales usos y casos prácticos.
7. Datalake, Datawarehouse, Mejores prácticas en SQL.
8. Presentación de Trabajo Final

Objetivo

1. Disponibilizar resultados de ML/AI a aplicaciones cliente.
2. Identificar/interpretar modelos de datos en soluciones de negocio
3. Definir arquitectura necesaria para integrarse con otras aplicaciones.
4. Crear modelos de datos y arquitecturas de mantenimiento escalables

Evaluación

El trabajo práctico final consta de:

- Selección de algún motor de bases de datos para hacer consultas
- Carga de datasets correspondientes
- Resolución de consultas


BREAK GLASS IN EMERGENCY
docker

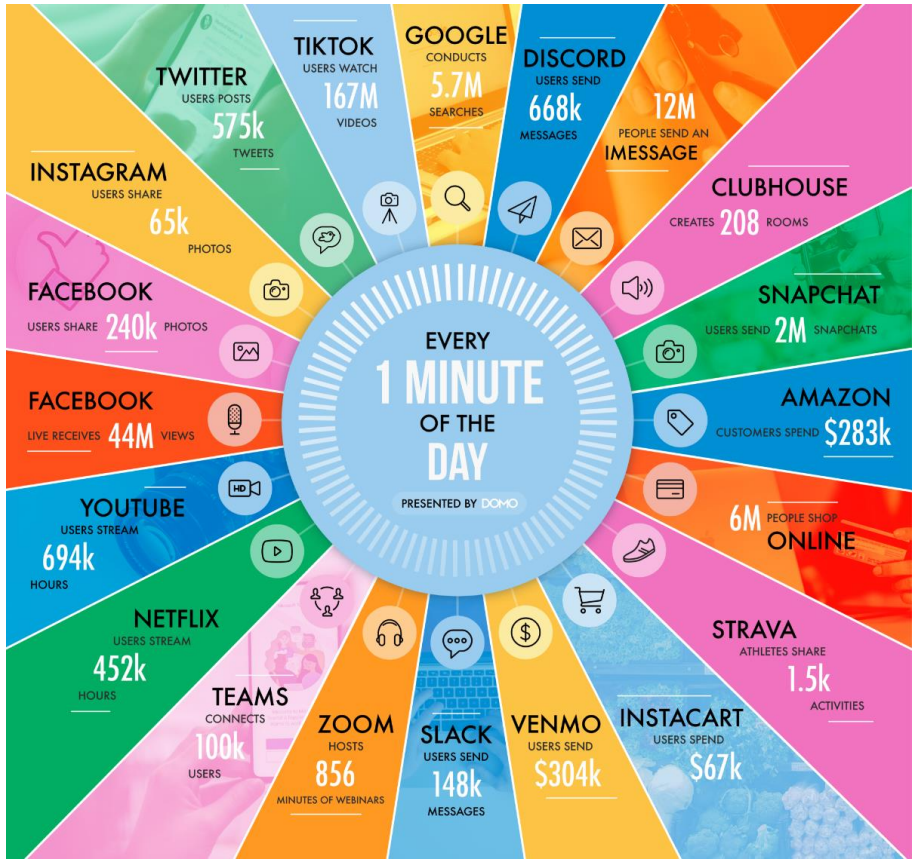
BASES DE DATOS

MOTIVACIÓN Y CONCEPTOS BÁSICOS

The 4 V's of big data

- Cuando analizamos un sistema de datos, en la actualidad podemos considerar que el concepto de *big data* está asociado con 4 conceptos claves:

- Volumen
- Variedad
- Velocidad
- Veracidad



La importancia de los datos y del almacenamiento

SE GENERAN DEL ORDEN DE 300 MILLONES DE TERABYTES POR DÍA

"Data is the New Oil."

Clive Humby | 2006
British data commercialization entrepreneur

¿Qué es una base de datos?

- Un conjunto de **datos almacenados** en un formato específico e interrelacionados por un contexto en común.

- Podemos pensar como una colección de datos que están:
  - *Lógicamente relacionados**
  - *Compartidos*
  - *Protegidos*
  - *Administrados*

Componentes de una DB

Informacion:	DBMS:	USERS:	HARDWARE:
• La información se obtiene de la base de datos, está integrada y compartida	• Es el Sistema de gestion de los datos (dbms por sus siglas en ingles). Es nuestro motor	• Quienes acceden ya sean personas o aplicaciones	• Donde montamos nuestro dbms

TIPOS DE BASES DE DATOS

Sql.

Modelo Logico/Relacional.

El modelo Lógico

- El modelo debería mantenerse indistinto del volumen de datos
- Los datos están organizados de acuerdo a lo que representa (Las tablas deben ser consistentes)
- Es genérico, el modelo es un template para la implementación física en *cualquier* rdbms

Normalización

- Es el proceso de reducción de complejidad de nuestra estructura de datos inicial a un modelo simple y *estable.*
- Es un proceso que involucra la remoción de *atributos*, *keys* y *relaciones* del modelo conceptual

Procesamiento transaccional.

Es la unidad de trabajo que tiene que ser ejecutada atómicamente y en un entorno aparentemente aislado. Tiene las siguientes características:

- **Logging:** Para asegurar durabilidad de las mismas
- **Control de concurrencia:** La ejecución de las transacciones tienen que (en apariencia) ser aisladas las unas de las otras.
- **Resolución de impasses (*Deadlocks*):** debemos considerar que las transacciones "pelean" por los recursos.

Principio de transacciones *ACID.*

Las transacciones ejecutadas en una RDBMS cumplen con las siguientes propiedades:

- **Atomicity:** Everything in a transaction succeeds lest it is rolled back.
- **Consistency:** A transaction cannot leave the database in an inconsistent state.
- **Isolation:** One transaction cannot interfere with another.
- **Durability:** A completed transaction persists, even after applications restart.

Tablas

Una tabla (también conocida como *entidad*) es una estructura bidimensional compuesta por ***tuplas*** (filas, registros) y ***atributos*** (campos, columnas o celdas).

Para crear/analizar las tablas que componen el MR, utilizamos *algebra relacional.*

EMPLOYEE NUMBER	MANAGER EMPLOYEE NUMBER	DEPARTMENT NUMBER	JOB CODE	LAST NAME	FIRST NAME	HIRE DATE	BIRTH DATE	SALARY AMOUNT
1006	1019	301	312101	Stein	John	861015	631015	3945000
1008	1019	301	312102	Kanieski	Carol	870201	680517	3925000
1005	0801	403	431100	Ryan	Loretta	861015	650910	4120000
1004	1003	401	412101	Johnson	Darlene	861015	560423	4630000
1007				Villegas	Arnando	870102	470131	5970000
1003	0801	401	411100	Trader	James	860731	570619	4785000

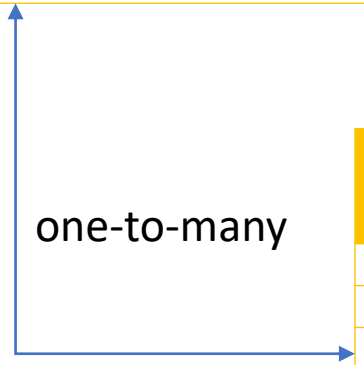
La tabla que observamos es una “proyección” conveniente del modelo relacional que planteamos

Relaciones en tablas.

Así mismo, las tablas pueden contener relaciones con otras tablas que componen nuestro MR. Existen 3 tipos de relaciones:

- *one-to-one*
- *one-to-many*
- *many-to-many*

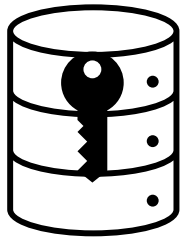
EMPLOYEE NUMBER	MANAGER EMPLOYEE NUMBER	DEPARTMENT NUMBER	JOB CODE	LAST NAME	FIRST NAME	HIRE DATE	BIRTH DATE	SALARY AMOUNT
1006	1019	301	312101	Stein	John	861015	631015	3945000
1008	1019	301	312102	Kanieski	Carol	870201	680517	3925000
1005	0801	403	431100	Ryan	Loretta	861015	650910	4120000
1004	1003	401	412101	Johnson	Darlene	861015	560423	4630000
1007		501		Villegas	Arnando	870102	470131	5970000
1003	0801	401	411100	Trader	James	860731	570619	4785000

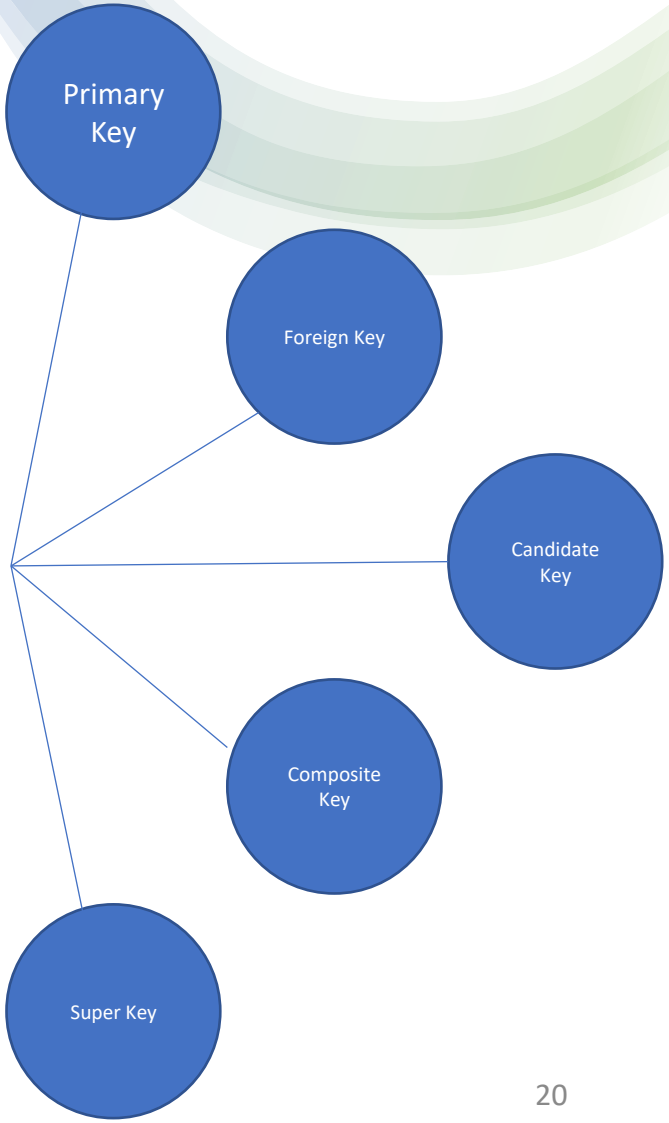


DEPARMENT NUMBER	DEPARTMENT NAME	BUDGET AMOUNT	MANAGER EN
501	marketing sales	80050000	1017
301	R&D	46560000	1019
302	product planning	22600000	1016
403	Education	93200000	1005
402	software support	30800000	1011
401	customer support	98230000	1003
201	technical operations	29380000	1025

CLAVES (KEYS).

- Dentro del mundo de RDBs vamos a encontrar que siempre se habla de distintos tipos de llaves (keys) que gobiernan nuestros datos.
- Estas llaves no son más que uno o más atributos que nos permiten identificar de manera univoca un dato en nuestra tabla.
- Además permiten establecer las relaciones del modelo.





CLAVES

- Clave primaria (PK)

- Es un requerimiento de una tabla tener una PK asignada para identificar cada una de ellas.
- Una PK es *única* por lo cual no pueden existir dos en una misma tabla.
- No pueden tener valores duplicados.
- Puede estar compuesta por una o más columnas
- **NO** puede ser nula
- Se consideran que son valores “no cambiantes”

- Clave Foranea (FK)

- FK son valores opcionales
- Puede haber más de una FK por tabla
- Se permiten duplicados y nulos
- Son valores que pueden cambiar.
- Una FK **tiene** que existir en otra tabla como una **PK.**

VENTAJAS DEL MODELO RELACIONAL

- Favorece el proceso de ***Normalización*** (por construcción), esto permite eliminar la redundancia de los datos
- Mediante el uso del lenguaje SQL podemos rápidamente generar “proyecciones” de nuestro MR para generar reportes/consultas
- Podemos crear varias relaciones no solo una
- Estas relaciones nos permite además evitar la duplicidad de registros.
- Garantiza la ***Integridad Referenciada***. Esto significa que si un dato está relacionado a otro, el modelo no va a permitir que se elimine.

Objetos de una RDB.

Tablas

Son los elementos que contienen los registros de mis datos

Vistas:

Son proyecciones predefinidas de las tablas existentes.

Usualmente las crean los equipos de DbA’s para optimizer el uso la base.

Triggers:

SQL Statements asociados a una tabla. Que automatizan procesos típicos cuando ciertos eventos ocurren

Stored Procedures:

Procedimientos generalizados/normalizados que se utilizan cross usuarios.

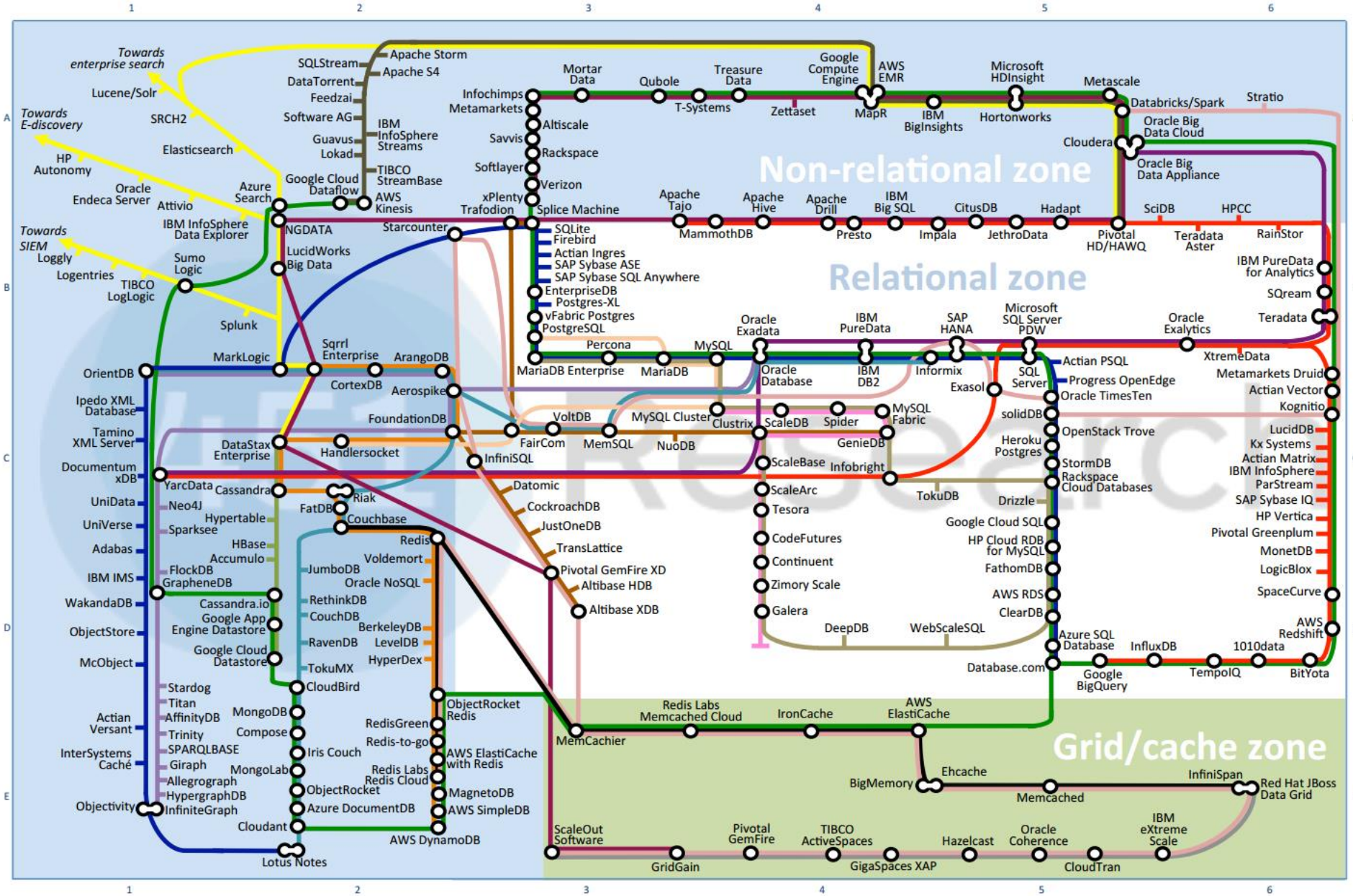
User Defined Functions

Programas/scripts que definimos para agilizar consultas.

UNIVERSO NOSQL.

UNIVERSO NOSQL.

DEPARMENT NUMBER	DEPARTMENT NAME	BUDGET AMOUNT	MANAGER EN
501	marketing sales	80050000	1017
301	R&D	46560000	1019
302	product planning	22600000	1016
403	Education	93200000	1005
402	software support	30800000	1011
401	customer support	98230000	1003
201	technical operations	29380000	1025



451 Research

Data Platforms Map

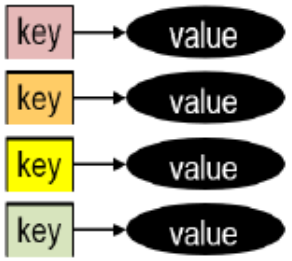
October 2014

Key:

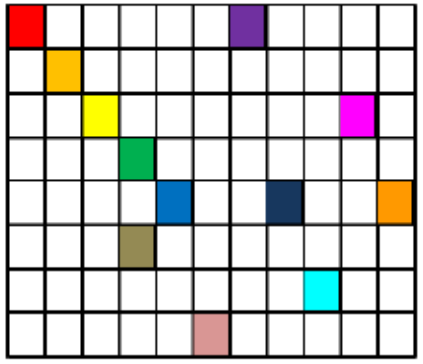
- General purpose
- Specialist analytic
- -as-a-Service
- BigTables
- Graph
- Document
- Key value stores
- Key value direct access
- Hadoop
- MySQL ecosystem
- Advanced clustering/sharding
- New SQL databases
- Data caching
- Data grid
- Search
- Appliances
- In-memory
- Stream processing

https://451research.com/dashboard/dpa

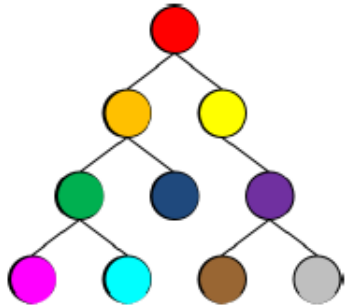
Tipos de bases NoSQL.



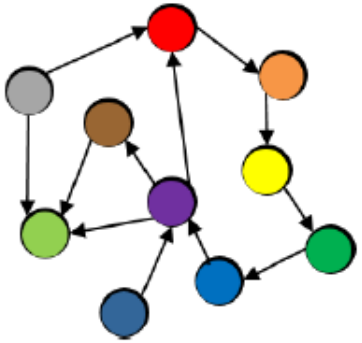
Key-Vlue



Column Family



Document



Graph Database

Sistemas híbridos (poliglotas)

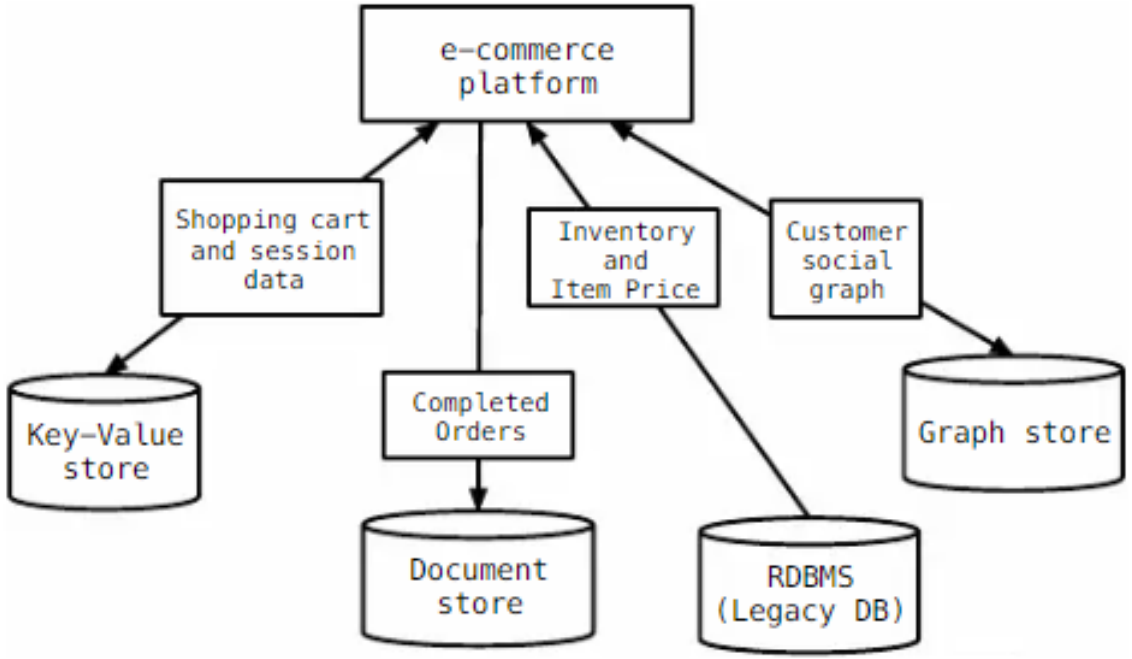


Figure 13.3 *Example implementation of polyglot persistence*

NoSQL Distilled

A Brief Guide to the Emerging World of Polyglot Persistence

Pramod J. Sadalage
Martin Fowler

371 systems in ranking, June 2021

Rank			DBMS	Database Model	Score		
Jun 2021	May 2021	Jun 2020			Jun 2021	May 2021	Jun 2020
1.	1.	1.	Oracle	Relational, Multi-model	1270.94	+1.00	-72.65
2.	2.	2.	MySQL	Relational, Multi-model	1227.86	-8.52	-50.03
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	991.07	-1.59	-76.24
4.	4.	4.	PostgreSQL	Relational, Multi-model	568.51	+9.26	+45.53
5.	5.	5.	MongoDB	Document, Multi-model	488.22	+7.20	+51.14
6.	6.	6.	IBM Db2	Relational, Multi-model	167.03	+0.37	+5.23
7.	7.	↑ 8.	Redis	Key-value, Multi-model	165.25	+3.08	+19.61
8.	8.	↓ 7.	Elasticsearch	Search engine, Multi-model	154.71	-0.65	+5.02
9.	9.	9.	SQLite				
10.	10.	↑ 11.	Microsoft Access				
11.	11.	↓ 10.	Cassandra				
12.	12.	12.	MariaDB				
13.	13.	13.	Splunk				
14.	14.	14.	Hive				
15.	15.	↑ 23.	Microsoft Azure SQL Database				
16.	16.	16.	Amazon DynamoDB				
17.	17.	↓ 15.	Teradata				

DB-Engines

DB-Engines is an initiative to collect and present information on database management systems (DBMS). In addition to established relational DBMS, systems and concepts of the growing NoSQL area are emphasized.

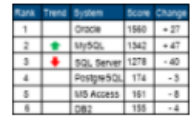
The DB-Engines Ranking is a list of DBMS ranked by their current popularity. The list is updated monthly.

THE GROWING DICHOTOMY

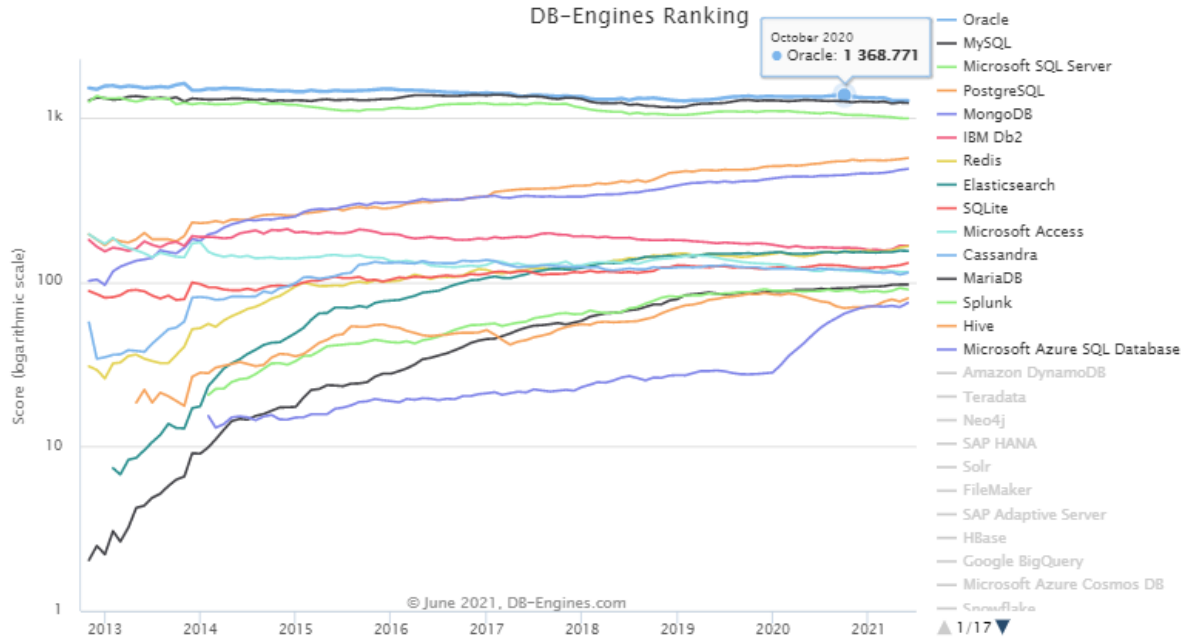
DB-Engines Ranking - Trend Popularity

The DB-Engines Ranking ranks database management systems according to their popularity.

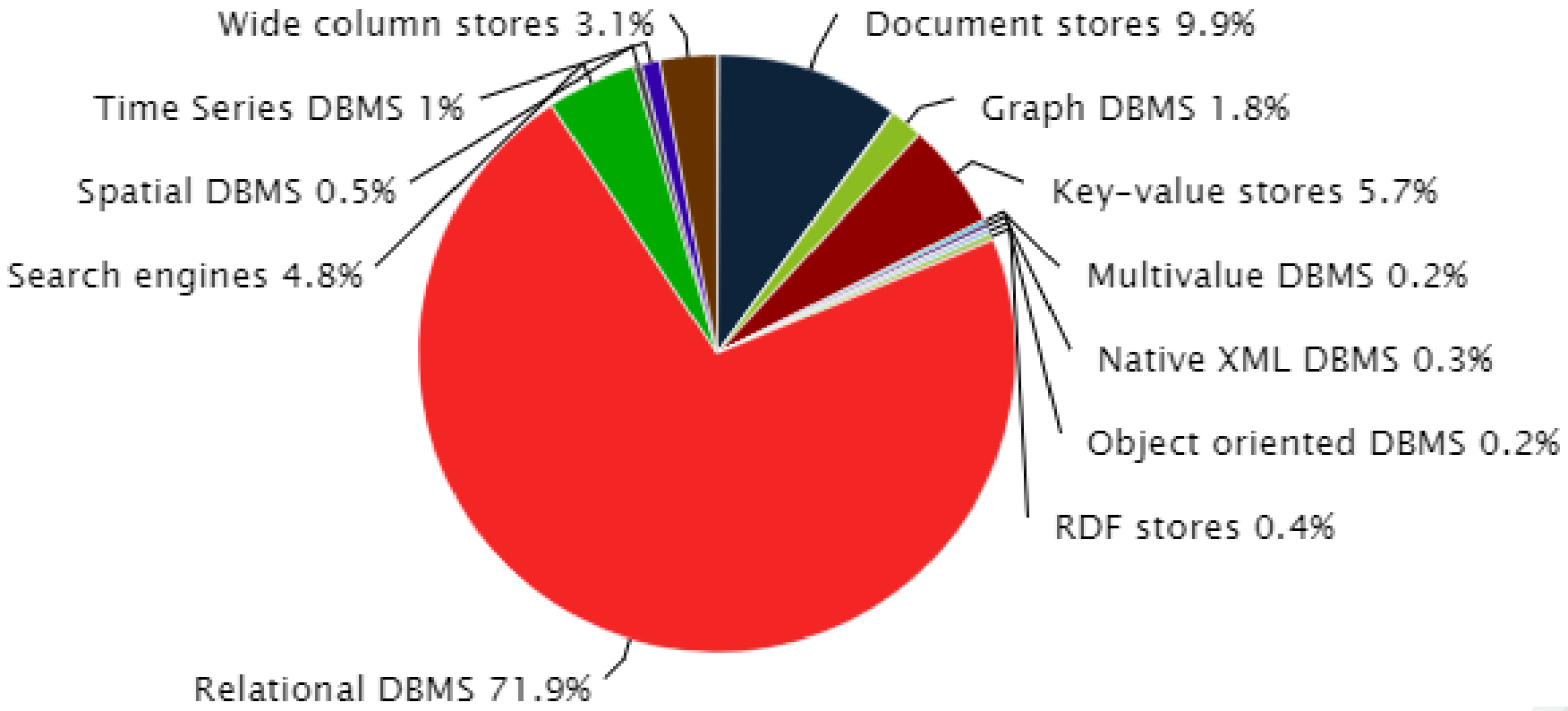
Read more about the method of calculating the scores.



ranking table
June 2021



Ranking scores per category in percent, April 2022



Open source vs Commercial

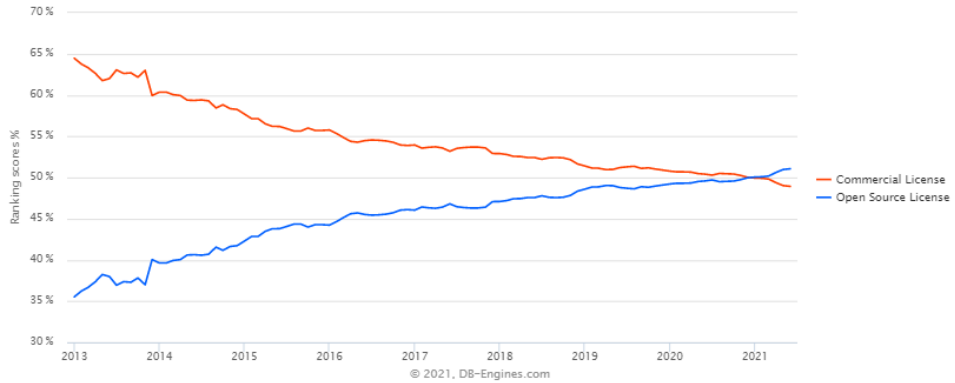
The top 5 commercial systems, June 2021

Rank	System	Score	Overall Rank
1.	Oracle	1271	1.
2.	Microsoft SQL Server	991	3.
3.	IBM Db2	167	6.
4.	Microsoft Access	115	10.
5.	Splunk	90	13.

The top 5 open source systems, June 2021

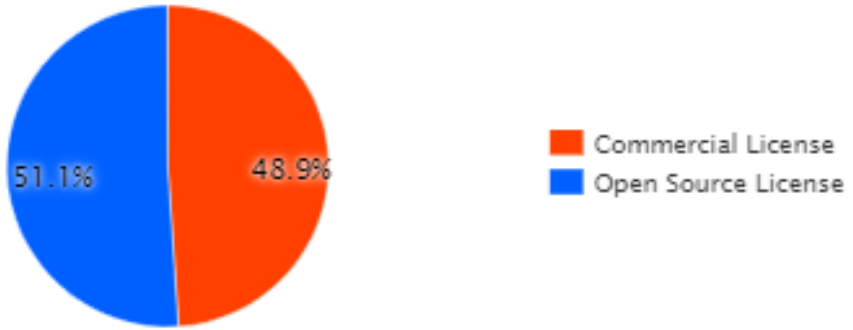
Rank	System	Score	Overall Rank
1.	MySQL	1228	2.
2.	PostgreSQL	569	4.
3.	MongoDB	488	5.
4.	Redis	165	7.
5.	Elasticsearch	155	8.

Popularity trend



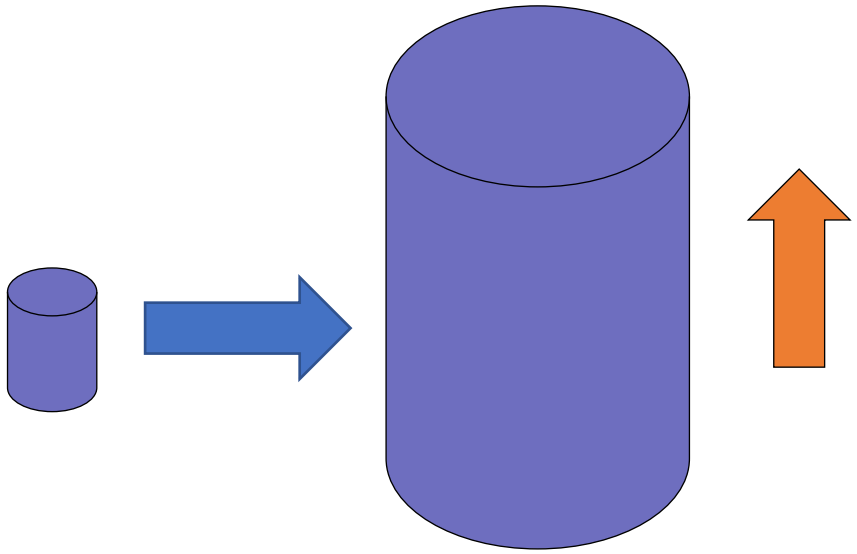
The above chart shows the historical trend of the popularity of open source and commercial database management systems.

Popularity scores, June 2021

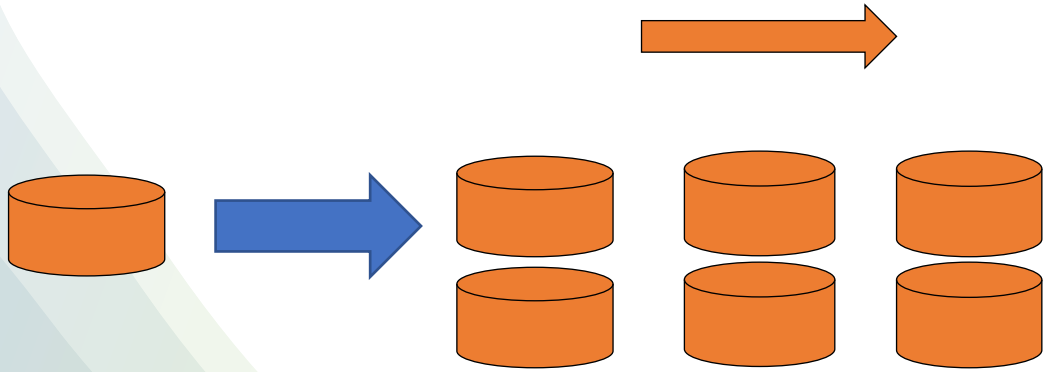


SCALING HORIZONTALLY

RELATIONAL DATABASES IS A CHALLENGING TASK, IF NOT IMPOSSIBLE.



- **ONE OF THE MAIN PROBLEMS THAT A NOSQL DATABASES SOLVES IS SCALE, AMONG OTHERS.**



No to SQL? Anti-Database Movement Gains Steam

The meet-up in San Francisco last month had a whiff of revolution about it, like a latter-day techie version of the American Patriots planning the Boston Tea Party.

By Eric Lai

Computerworld | JUL 1, 2009 8:00 AM PT

The meet-up in San Francisco last month had a whiff of revolution about it, like a latter-day techie version of the American Patriots planning the Boston Tea Party.

The inaugural get-together of the burgeoning NoSQL community crammed 150 attendees into a meeting room at CBS Interactive.

Like the Patriots, who rebelled against Britain's heavy taxes, NoSQLers came to share how they had overthrown the tyranny of slow, expensive relational databases in favor of more efficient and cheaper ways of managing data.

MORE LIKE THIS ::

Java in the Cloud: Google, Aptana, and Stax

Open Source CRM and ERP: Bending the Back Office

Review: Google Bigtable scales with ease

.NoSQL originally started off as a simple combination of two words—No and SQL—clearly and completely visible in the new term. No acronym. What it literally means is, "I do not want to use SQL". To elaborate, "I want to access database without using any SQL syntax"

"…people are continually interpreting nosql to be *anti-RDBMS*, it's the only rational conclusion when the only thing some of these projects share in common is that they are *not relational databases*."

Hitos

Bigtable: A Distributed Storage System for Structured Data

Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach
har Chandra, Andrew Fikes, Robert E. Gruber
wilsonh,kerr,m3b,tushar,fikes,gruber}@google.com
Google, Inc.

achieved scalability and high performance, but Bigtable provides a different interface than such systems. Bigtable does not support a full relational data model; instead, it provides clients with a simple data model that supports dynamic control over data layout and format, and allows clients to reason about the locality properties of the data represented in the underlying storage. Data is indexed using row and column names that can be arbitrary strings. Bigtable also treats data as uninterpreted strings, although clients often serialize various forms of structured and semi-structured data into these strings. Clients can control the locality of their data through careful choices in their schemas. Finally, Bigtable schema parameters let clients dynamically control whether to serve

emory or from disk.

lescribes the data model in more detail, and ovides an overview of the client API. Sec-

for managing
a very large
f commodity
a in Bigtable,
d Google Fi-
rent demands
rom URLs to
requirements
data serving).
successfully
tion for all of
cribe the sim-

Dynamo: Amazon's Highly Available Key-value Store

Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati,
Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall
and Werner Vogels

Amazon.com

ABSTRACT

Reliability at massive scale is one of the biggest challenges we face at Amazon.com, one of the largest e-commerce operations in the world; even the slightest outage has significant financial consequences and impacts customer trust. The Amazon.com platform, which provides services for many web sites worldwide, is implemented on top of an infrastructure of tens of thousands of servers and network components located in many datacenters around the world. At this scale, small and large components fail continuously and the way persistent state is managed in the face of these failures drives the reliability and scalability of the software systems.

This paper presents the design
highly available key-value stora
core services use to provide
achieve this level of availabilit
under certain failure scenarios.
versioning and application-assis
that provides a novel interface f

One of the lessons our organization has learned from operating Amazon's platform is that the reliability and scalability of a system is dependent on how its application state is managed. Amazon uses a highly decentralized, loosely coupled, service oriented architecture consisting of hundreds of services. In this environment there is a particular need for storage technologies that are always available. For example, customers should be able to view and add items to their shopping cart even if disks are failing, network routes are flapping, or data centers are being destroyed by tornados. Therefore, the service responsible for managing shopping carts requires that it can always write to and read from its data store, and that its data needs to be available

A Big Data Modeling Methodology for Apache Cassandra

Artem Chebotko
DataStax Inc.
Email: achebotko@datastax.com

Andrey Kashlev
Wayne State University
Email: andrey.kashlev@wayne.edu

Shiyong Lu
Wayne State University
Email: shiyong@wayne.edu

***Abstract*—Apache Cassandra is a leading distributed database of choice when it comes to big data management with zero downtime, linear scalability, and seamless multiple data center deployment. With increasingly wider adoption of Cassandra**

because of the expressivity of the Structured Query Language (SQL) that readily supports relational joins, nested queries, data aggregation, and numerous other features that help to

Bigtable, 2006

Column family
indexed by a row key, column key, and a timestamp

Amazon, Dynamo, 2007

Key value

Facebook release Cassandra, 2008

Column family

NoSQL

- Key-Value
- Column Family
- Documentos
- Grafos

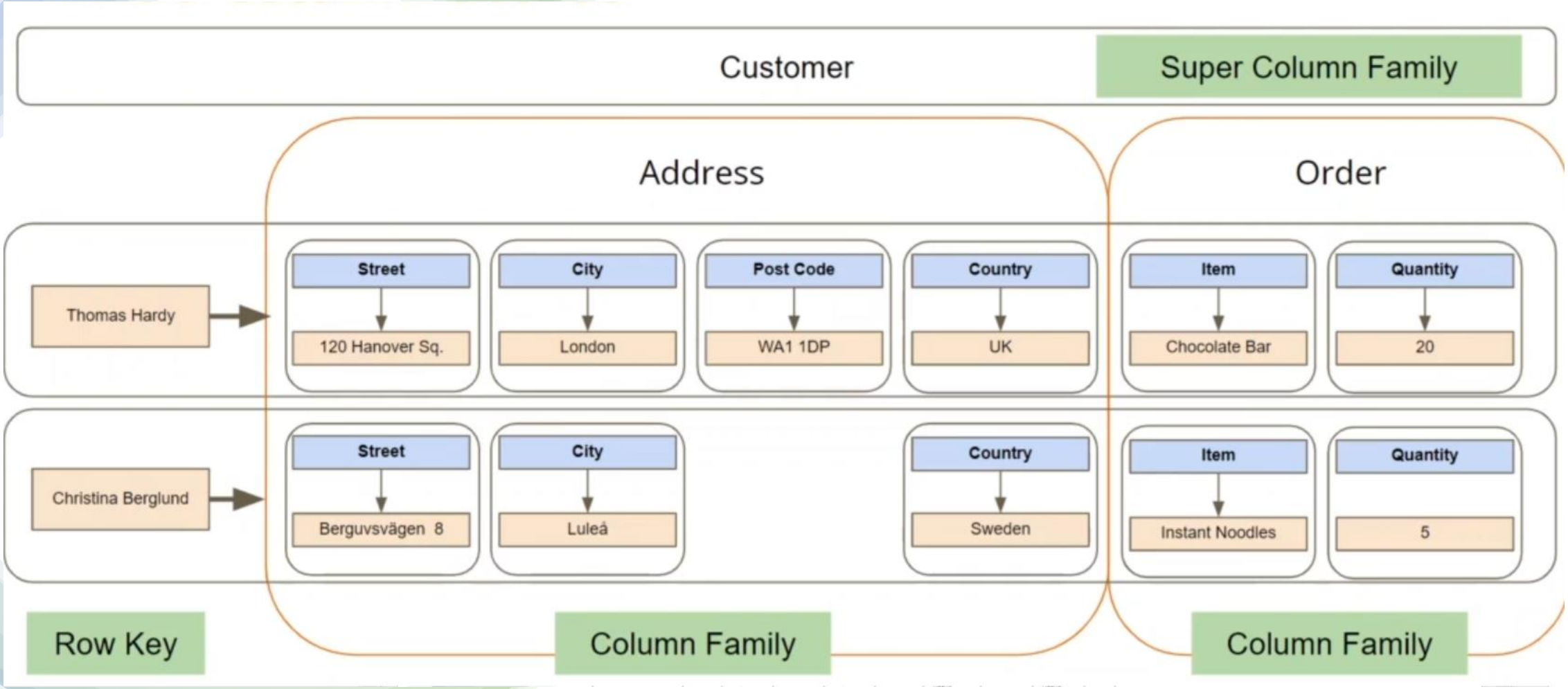
NoSQL

- Key-Value
  - Carrito ecommerce
  - Guardar sesiones
  - Respuestas de API en cache
  - Recomendaciones de productos
- Column Fam
- Documentos
- Grafos

key	value
123	Name,Address,Email
456-321-211	John Turner
223	Name,Address,Email, Payments

NoSQL

- Key-Value
- Column Family
  - Analítica real time
  - Series temporales
  - IoT
- Documentos
- Grafos


Customer
Super Column Family
Address
Order
Thomas Hardy
Street
120 Hanover Sq.
City
London
Post Code
WA1 1DP
Country
UK
Item
Chocolate Bar
Quantity
20
Christina Berglund
Street
Berguvsvägen 8
City
Luleå
Country
Sweden
Item
Instant Noodles
Quantity
5
Row Key
Column Family
Column Family

NoSQL

- Key-Value
- Column Family
- Documentos
  - Catálogos
  - Aplicaciones web/ecommerce
  - IoT
  - Analitica realtime
- Grafos

Nc rce

Relational

id	first_name	last_name	age
1	John	Turner	35
2	Mary	Smith	28
3	Adam	Nicolson	37
4	Susan	Ford	21

order_id	user_id	order_date
83	1	2020-01-10
91	1	2019-07-01

supplier_id	user_id	supplier
25	1	supplierA
26	1	supplierE

prd_id	user_id	product	price
30	1	Grill basket	15.50
53	1	Chocolate bar	9.30

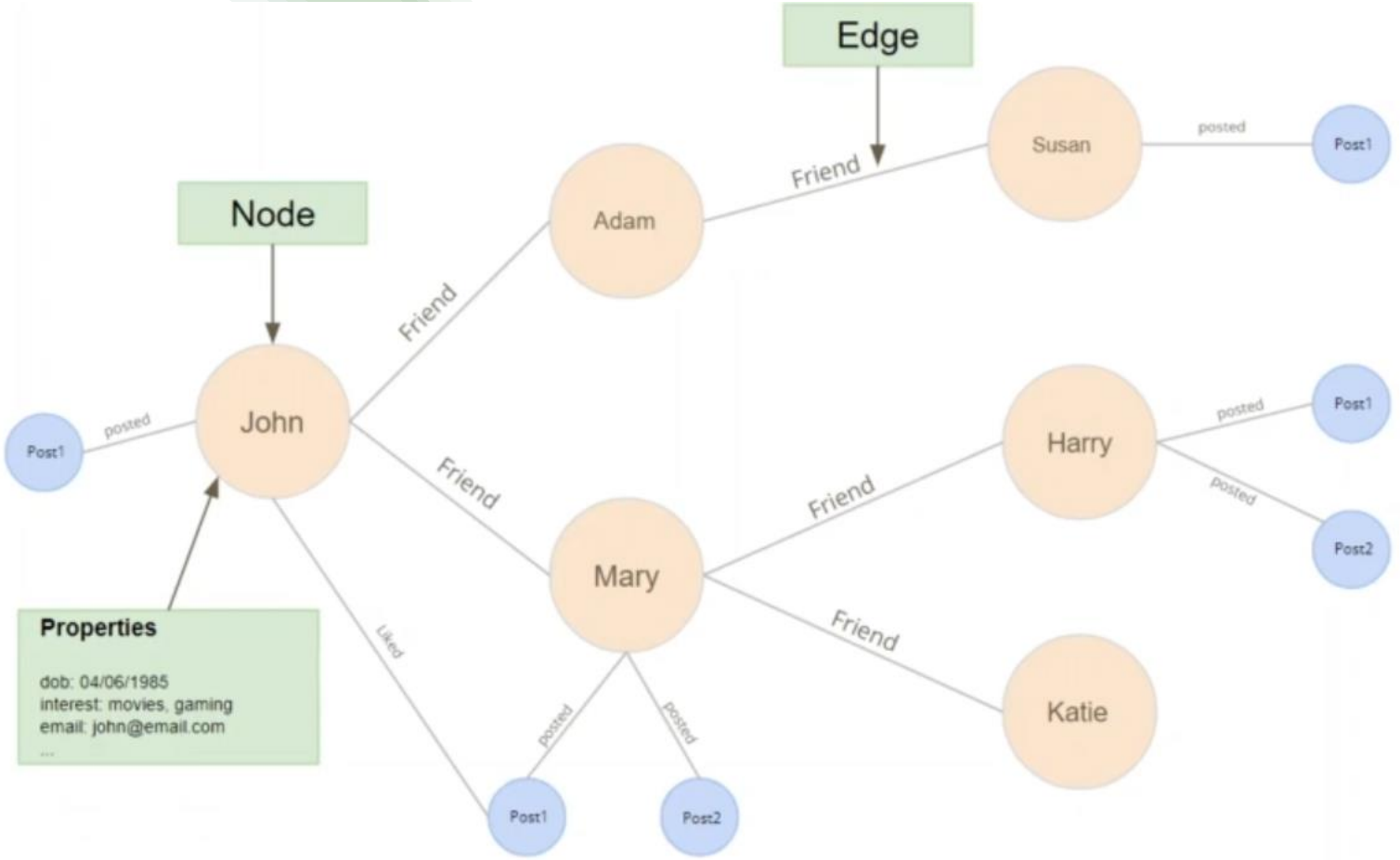
JSON Document


```
{
  "id": "1",
  "first_name": "John",
  "last_name": "Turner",
  "age": "35",
  "order": [
    {
      "supplier": "supplierA",
      "order_date": "2020-01-10",
      "product": "grill basket",
      "price": "15.50",
      "id":"83"
    },
    {
      "supplier": "supplierE",
      "order_date": "2019-07-10",
      "product": "chocolate bar",
      "price": "9.30",
      "id":"91"
    }
  ]
}
```

NoSQL

- Key-Value
- Column Family
- Documentos
- Grafos
  - Recomendaciones
  - Fraudes
  - NLP
  - Logistica
  - Redes sociales

NoSQL


Edge
Node
Adam
Susan
Friend
posted
Post1
Friend
John
Post1
posted
Friend
Mary
Harry
posted
Post1
posted
Post2
Friend
Friend
Katie
Properties
dob: 04/06/1985
interest: movies, gaming
email: john@email.com
...
Liked
posted
posted
Post1
Post2

CONSISTENCIAS

Modelos de consistencia

Determina la visibilidad y el orden aparente de las actualizaciones.

Es básicamente un contrato entre los procesos y el almacén de datos. Este contrato dice que, si los procesos aceptan obedecer ciertas reglas, el almacén promete funcionar correctamente.

Consistencia

- Consistencia fuerte

todas las operaciones de lectura deben devolver datos de la última operación de escritura completada.

- Consistencia eventual

las operaciones de lectura verán, conforme pasa el tiempo, las escrituras. En un estado de equilibrio, el estado devolvería el último valor escrito.

A medida que $t \rightarrow \infty$ los clientes verán las escrituras

Consistencia no estricta

Read Your Own Writes (RYOW) consistency

El cliente lees sus actualizaciones inmediatamente después de que hayan sido completadas, independientemente si escribe en un server y lee de otro. Las actualizaciones de otros clientes no tienen por qué ser visibles instantáneamente.

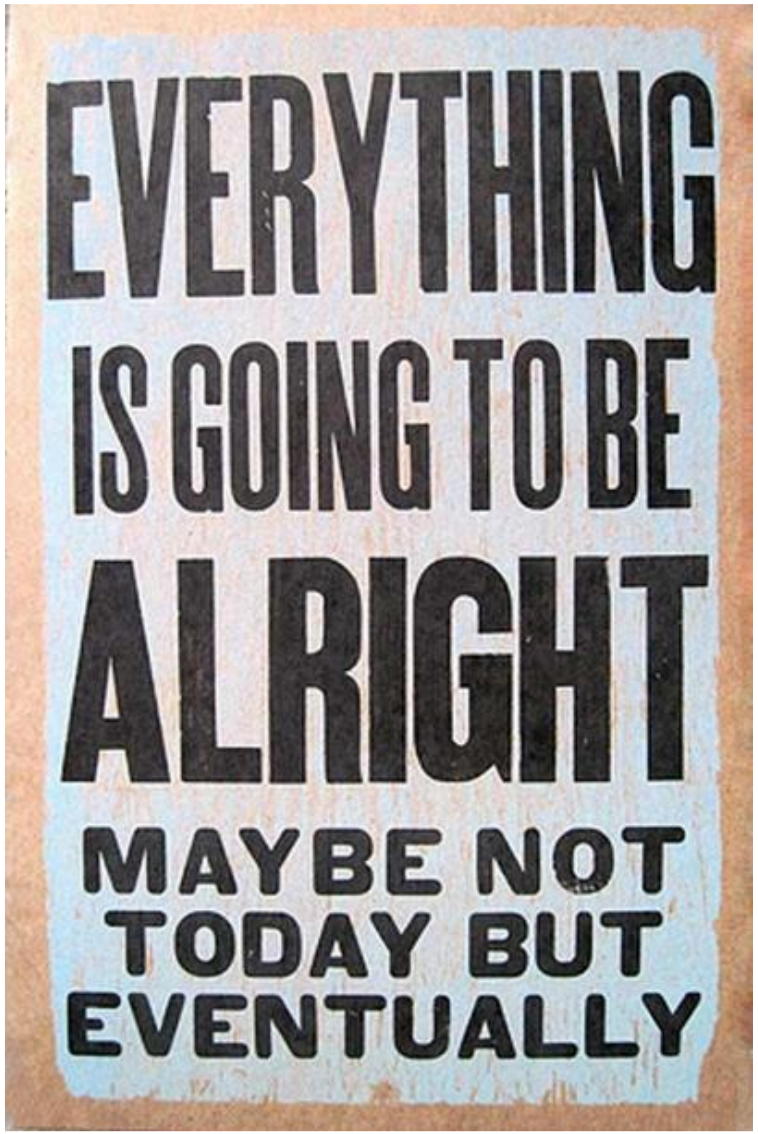
Consistencia no estricta

Consistencia de sesión.

Un poco más débil que RYOW. Provee ésta consistencia solo si el cliente está dentro de la misma sesión. Usualmente sobre el mismo server.

Consistencia no estricta

Consistencia eventual



TEOREMA CAP

Consistency → todos ven los mismos datos al mismo tiempo

Availability → si se puede comunicar con un nodo en el cluster, entonces se pueden leer y escribir datos

Partition tolerance → el cluster puede sobrevivir a roturas de comunicación que lo dividan en particiones que no pueden comunicarse entre ellas

Es imposible para cualquier sistema de computación distribuida proveer las tres propiedades simultáneamente.

Las 8 Falacias de la computación distribuída

- La red es confiable
- La latencia es cero; la latencia no es problema
- El ancho de banda es infinito
- La red es segura
- La topología no cambia
- Hay uno y solo un administrador
- El coste de transporte es cero
- La red es homogenea

Teorema CAP

Las 8 Falacias de la Computación Distribuida

Consistency

Availability

Partition tolerance

Teorema CAP

Las 8 Falacias de la Computación Distribuida

¡Hay que tolerar particiones!

CP - Consistency/Partition Tolerance

AP - Availability/Partition Tolerance

Ejemplo

Las transacciones bancarias son inconsistentes, particularmente para ATMs, que están diseñados para tener un comportamiento en modo normal y otro en modo partición. En modo partición la *Availability* es elegida por sobre la *Consistencia*

Históricamente la industria financiera ha sido inconsistente a causa de la imperfección de las comunicaciones.

ATMs realizan operaciones conmutativas de manera que el orden en el cuales se aplican no importa.

- Si hay una partición el cajero puede seguir funcionando; luego al volver a recuperar la partición se envían las operaciones y el saldo final sigue siendo correcto.
- Lo que se hace es delimitar y administrar el riesgo. Los ATMs son rentables aún a costa de alguna pérdida.

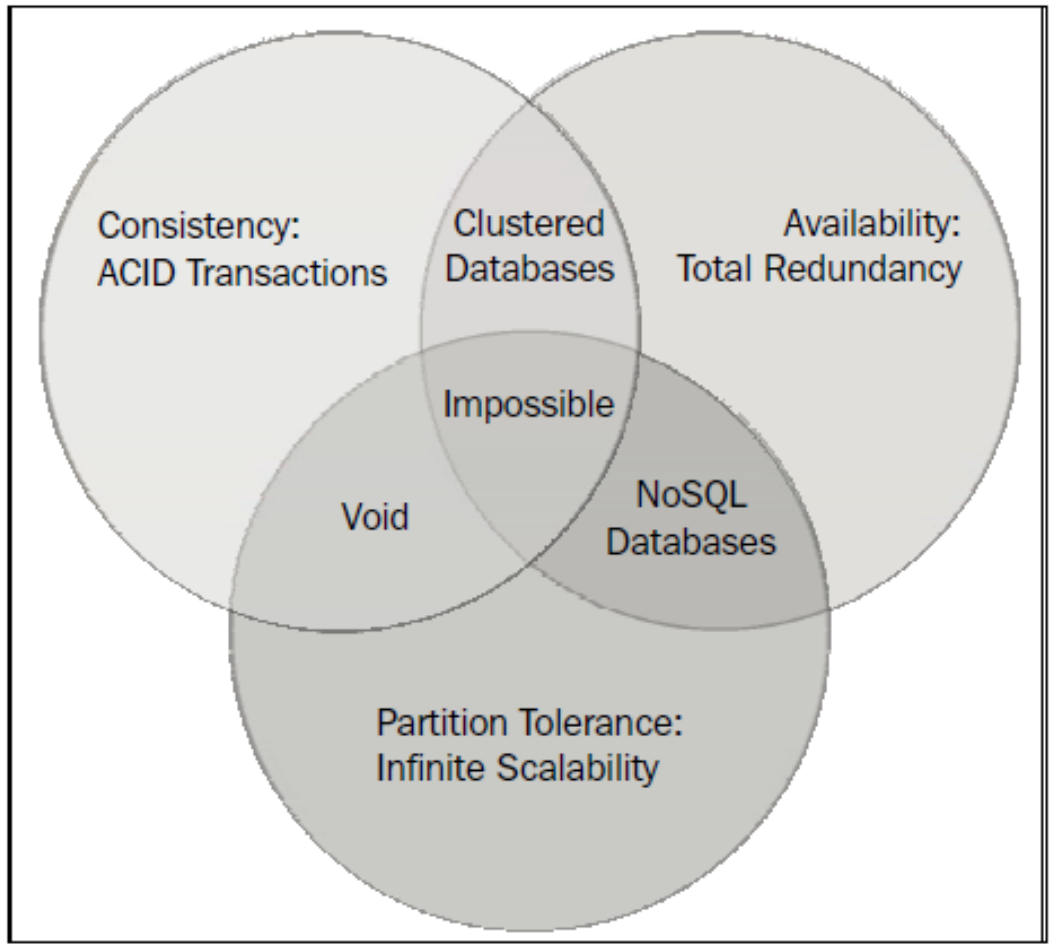
BASE, coined by Eric Brewer:

- **Basic availability**: Each request is guaranteed a response—successful or failed execution.

- **Soft state**: The state of the system may change over time, at times without any input (for eventual consistency).

- **Eventual consistency**: The database may be momentarily inconsistent but will be consistent eventually.

Relationship between CAP, ACID, and NoSQL



Feature	NoSQL Databases	Relational Databases
Performance	High	Low
Reliability	Poor	Good
Availability	Good	Good
Consistency	Poor	Good
Data Storage	Optimized for huge data	Medium sized to large
Scalability	High	High (but more expensive)

Replacing Datacenter Oracle with Global Apache Cassandra on AWS

ROOPA TANGIRALA
Engineering Manager
Netflix

1997

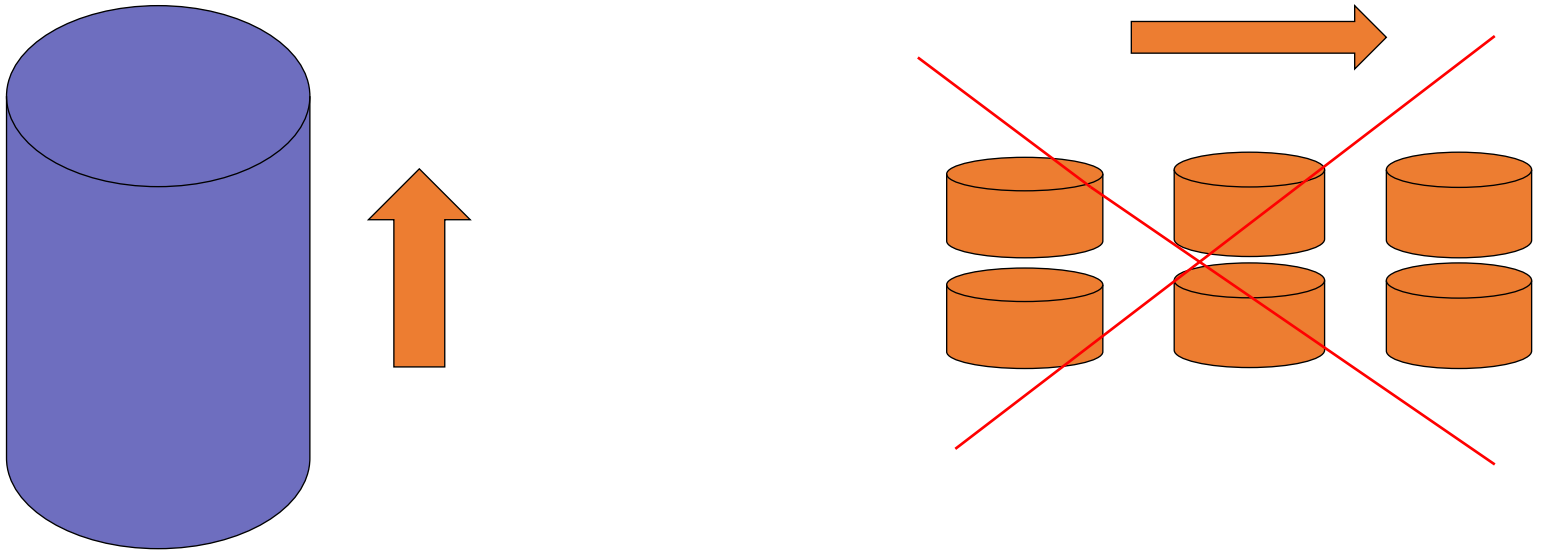
Netflix was originally nothing more than an online version of a traditional video rental store. Customers could visit the Netflix website and choose which items they wished to rent. Each rental cost $4, along with a $2 shipping fee.

2007 Netflix shipped its billionth DVD!

BACKEND

- RDBMS not flexible enough to handle modern day applications, which are a mixture of structured, semi-structured and unstructured data. WHILE Oracle handles structured data well it lacks dynamic datamodel which is important for high traffic modern applications.

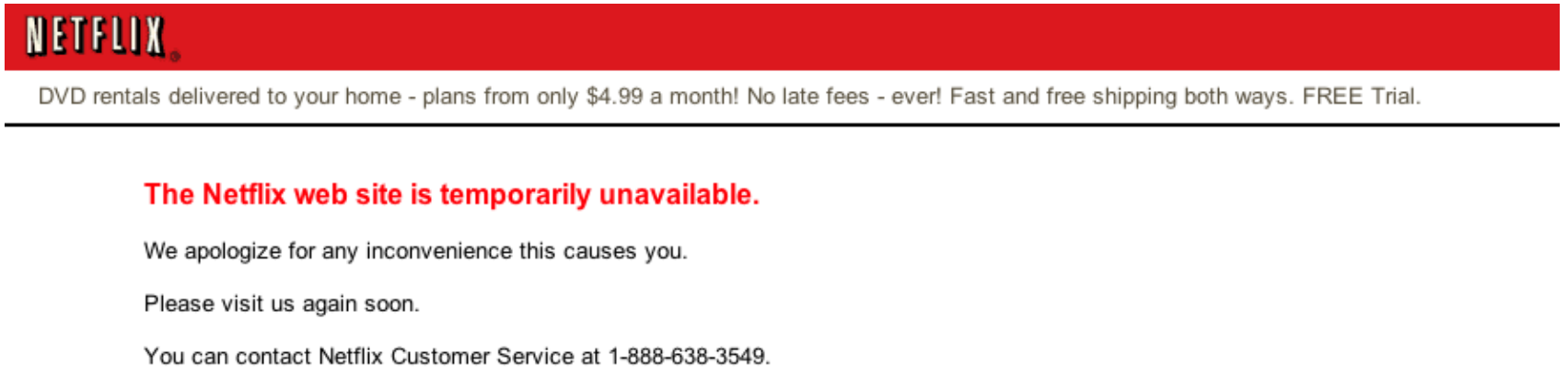
NO HORIZONTAL SCALING



Oracle scales up , master slave design limits both its scalability and performance for online applications, the failure of Oracle to add more capacity online in an elastic fashion without impacting the applications is a major drawback.

EVERY TWO WEEKS!!

- Website went down every two weeks to include changes to PLSQL procedures, packages, schema changes and other backend database changes. Was ok for DVD since the shipping could happen after the site was back up but not acceptable for streaming service.



2008

- This was the ugly outage and changed the way they think about infrastructure , reliability and availability.

Home / News & Analysis / **Netflix Outage Blamed on Hardware**

Netflix Outage Blamed on Hardware

BY CHLOE ALBANESIUS AUGUST 25, 2008 10:03AM EST 0 COMMENTS

Faulty hardware is to blame for a glitch that took Netflix shipping systems offline for several days earlier this month, according to a blog post from the company.

0 SHARES

Faulty hardware is to blame for a glitch that took Netflix shipping systems offline for several days earlier this month, according to a blog post from the company.

"With some great forensic help from our vendors, root cause was identified as a key faulty hardware component," according to Mike Osier, head of IT operations at Netflix. "It definitively caused the problem yet reported no detectable errors."

Problems started on Monday, August 11 when Netflix monitors flagged a database corruption in its shipping system. "Over the course of the day, we began experiencing similar problems in peripheral databases until our shipping system went down," Osier wrote.

MOVE TO CLOUD

REQUIREMENTS

- HIGHLY AVAILABLE
- MULTI DATACENTER SUPPORT
- PREDICTABLE PERFORMANCE AT SCALE

CHALLENGES

SECURITY

DENORMALIZE

DENORMALIZE

DENORMALIZE

DENORMALIZE

DENORMALIZE

DATALAKE DATAWAREHOUSE Y DATAMART

QUE ES UN DATAWAREHOUSE?

- CONTIENE INFORMACIÓN DE BASES DE DATOS TRANSACCIONALES
- SUMA INFORMACIÓN EXTERNA PARA VISTAS 360
- ORIENTADO A BIG DATA
- DATA ESTRUCTURADA DEL TIPO RELACIONAL
- ACID COMPLIANCE ES MENOS ESTRICTO
- AYUDA A CONSEGUIR INSIGHTS DE NEGOCIO EN BI

Google BigQuery

CARACTERÍSTICAS

- SUBJECT ORIENTED:

  SE ORIENTA A SOLUCIONAR PROBLEMAS DE NEGOCIO CON DATOS

- INTEGRADO:

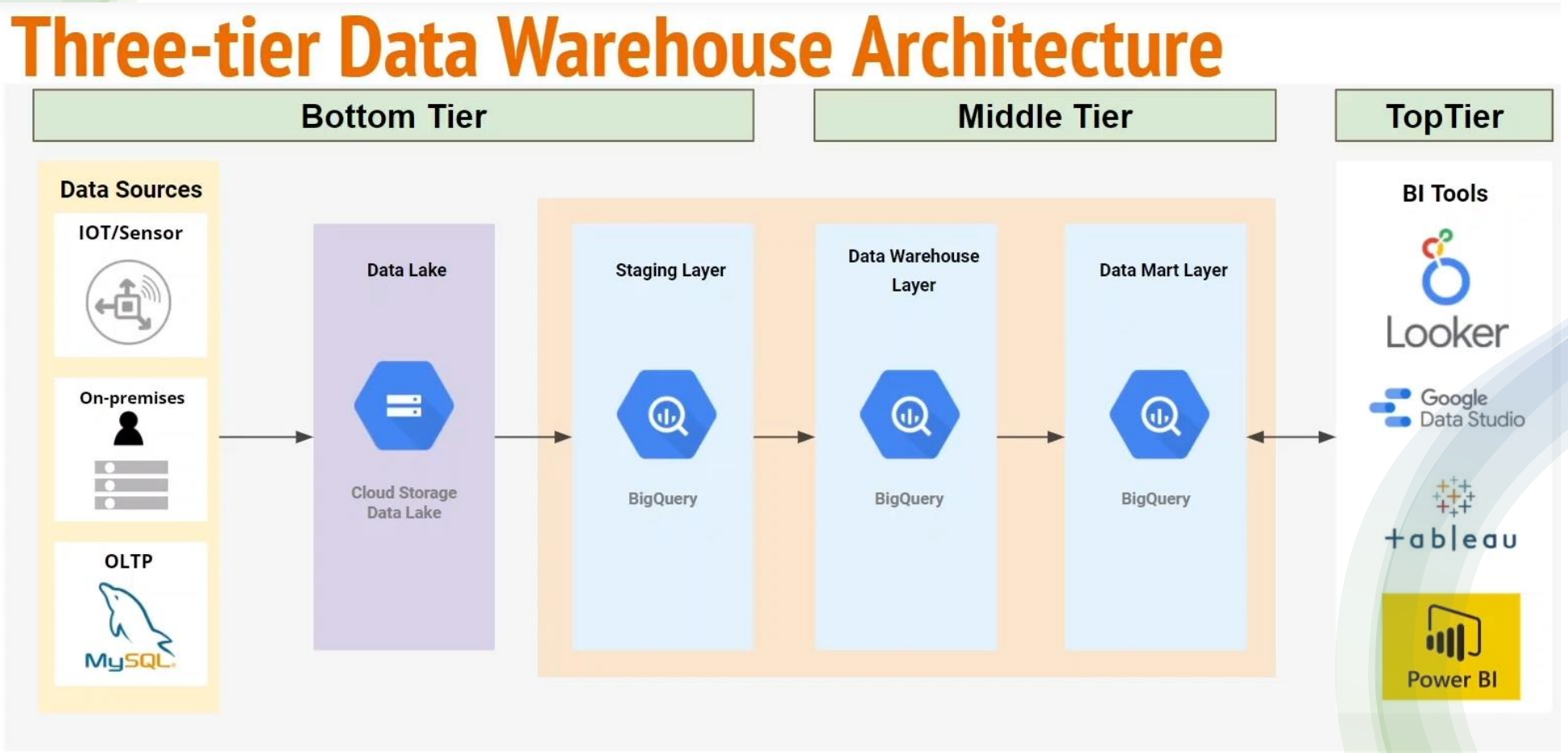
  RECIBE INFORMACIÓN DE DIFERENTES LUGARES CON EL FIN DE INTEGRARLOS

- VARIANTE EN EL TIEMPO

  SE GUARDA INFORMACIÓN CON CIERTA FRECUENCIA (DIARIO, SEMANAL) Y SE GUARDA LA INFORMACIÓN HISTÓRICA

- NO ES VOLATIL:
  - NUEVOS DATOS ENTRAN SIN QUE VIEJOS DATOS SE BORREN

COMO SE VE UN DW?



DUDAS?