Esta clase va a ser

• grabada

Certificados oficialmente por

R/GA

CODERHOUSE

Clase 2. FUNDAMENTOS DE LA CIENCIA DE DATOS

Stack Tecnológico del Data Scientist

Certificados oficialmente por

R/GA

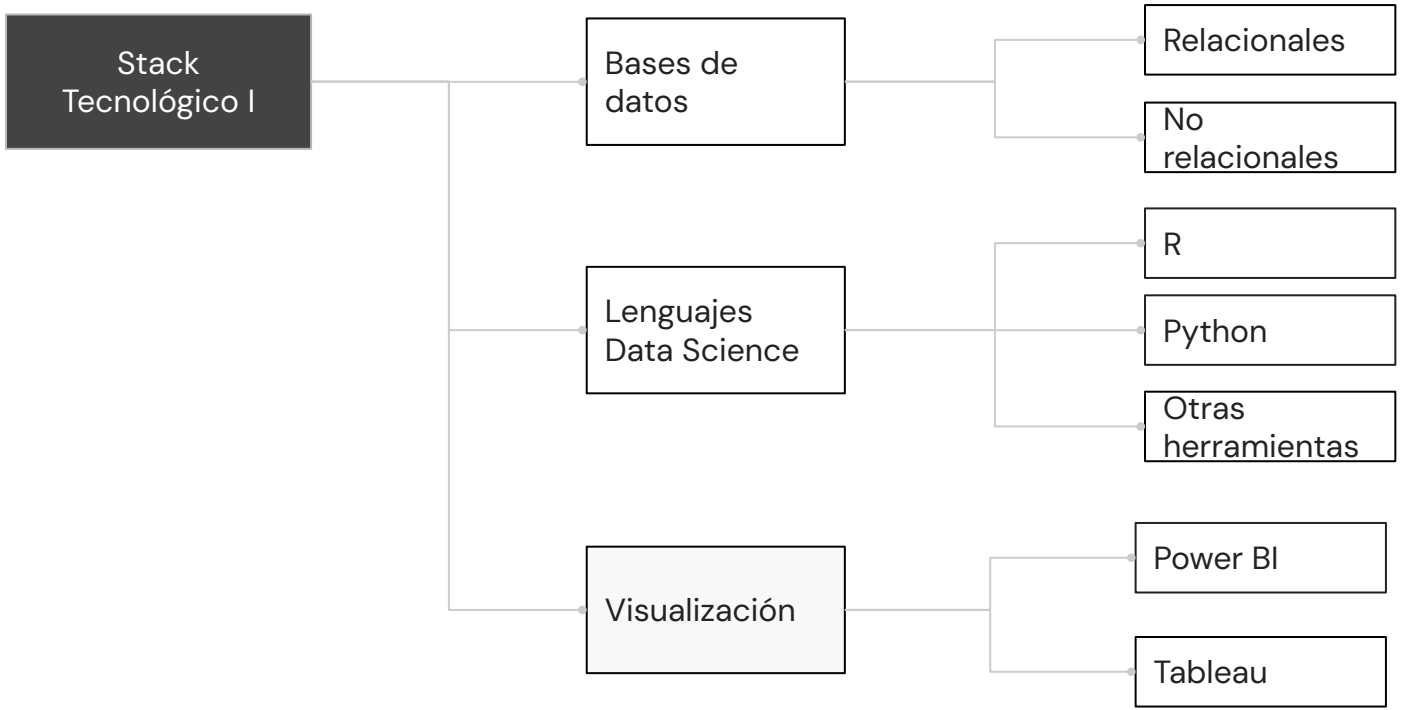
CODERHOUSE

Objetivos de la clase

- **Clasificar** las principales herramientas para un Científico de Datos y sus características.

CODERHOUSE

MAPA DE CONCEPTOS



¡Empecemos!

Un científico de datos es un profesional dedicado exclusivamente en analizar e interpretar grandes bases de datos. Para ello, **debe aprender a utilizar múltiples herramientas que estaremos clasificando a lo largo de esta sesión.**

CODERHOUSE

Bases de Datos

Introducción

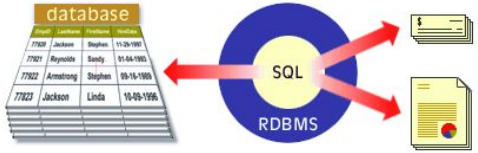
Una Base de Datos **es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso**.
En términos generales, podemos dividirlas en:

- ✓ **Bases de Datos Relacionales**
- ✓ **Base de Datos no Relacionales – No SQL**

Tecnologías de bases de datos relacionales

BD relacionales

Responden al **Modelo de Datos relacional propuesto por Edward Frank Codd en 1970**, tal cual como podemos observar en la siguiente imagen:



Primary key

Foreign key

Table Name: **EMP**

EMP NO	ENAME	JOB	DEPT NO
7839	KING	PRESIDENT	10
7698	BLAKE	MANAGER	30
7782	CLARK	MANAGER	10
7566	JONES	MANAGER	20

Primary key

Table Name: **DEPT**

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

CODERHOUSE

Microsoft SQL Server

Es un sistema de gestión de bases de datos relacionales (RDBMS) que admite una amplia variedad de aplicaciones de procesamiento de transacciones, inteligencia empresarial y análisis en entornos informáticos corporativos.

MySQL

Es un sistema de gestión de bases de datos relacional, desarrollado bajo una licencia dual: Licencia pública general / Licencia Comercial por Oracle Corporation. A su vez también, es una de las bases de datos más populares en general junto a Oracle y SQL Server.

PostgreSQL

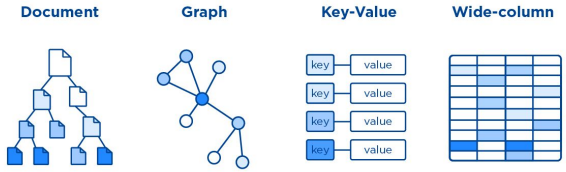
Es un sistema gestor de bases de datos relacionales, orientado a objetos, multiplataforma y open source. Está desarrollado desde 1996 por la comunidad de SGBD POSGRES.

Otras: Oracle Database, IMB DB2, Access, SQL Cloud.

Tecnologías de bases de datos no relacionales

BD no relacionales

Modelo propuesto por **Carlo Strozzi en 1998**, como una base de datos "relacional" de código abierto y liviana que no usa SQL, desarrollado en principio para datos web (no estructurados) y por la necesidad de un procesamiento más rápido.



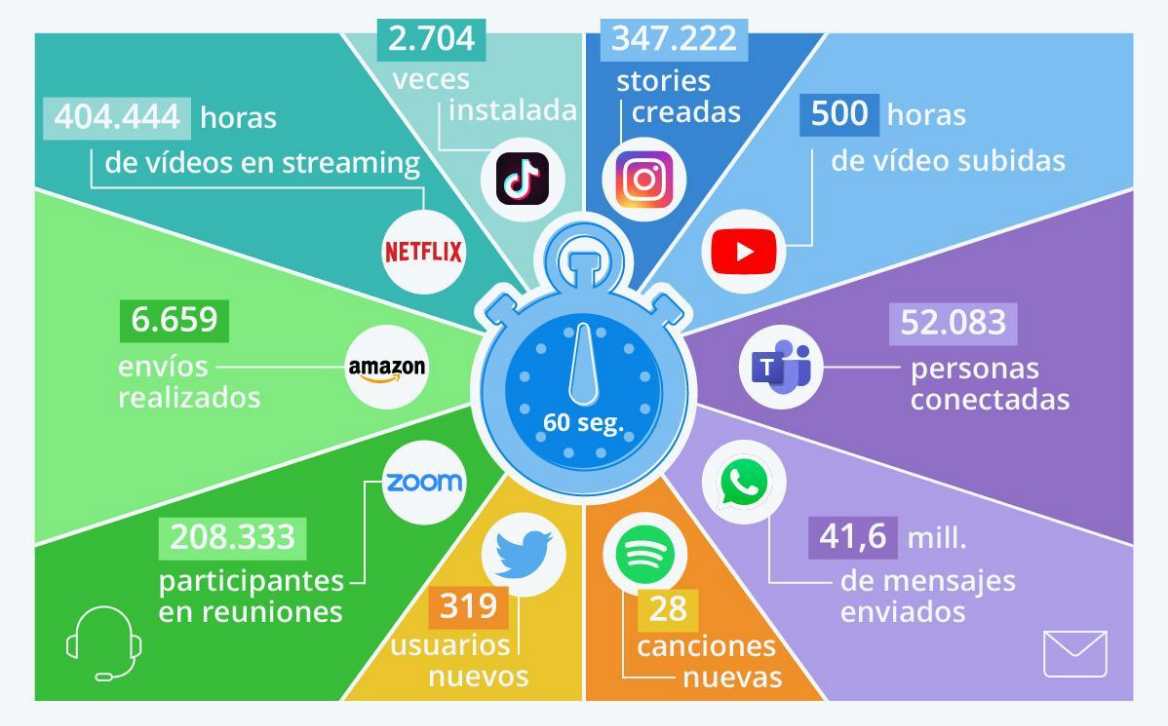
Key	Document
1001	{ "CustomerID": 99, "OrderItems": [{ "ProductID": 2010, "Quantity": 2, "Cost": 520 }, { "ProductID": 4365, "Quantity": 1, "Cost": 18 }], "OrderDate": "04/01/2017" }
1002	{ "CustomerID": 220, "OrderItems": [{ "ProductID": 1285, "Quantity": 1, "Cost": 120 }], "OrderDate": "05/08/2017" }

CODERHOUSE

NoSQL

Los datos masivos, reciben el nombre de **Big Data,** y el tipo de tecnología que ha surgido para tratar de poner solución a muchos de estos problemas se conoce como **NoSQL.** Los sistemas NoSQL no solo pueden manejar datos estructurados y no estructurados, sino que también **pueden procesar Big Data no estructurado rápidamente**

Internet en 1 minuto – Big Data



Fuente: statista.com

CODERHOUSE

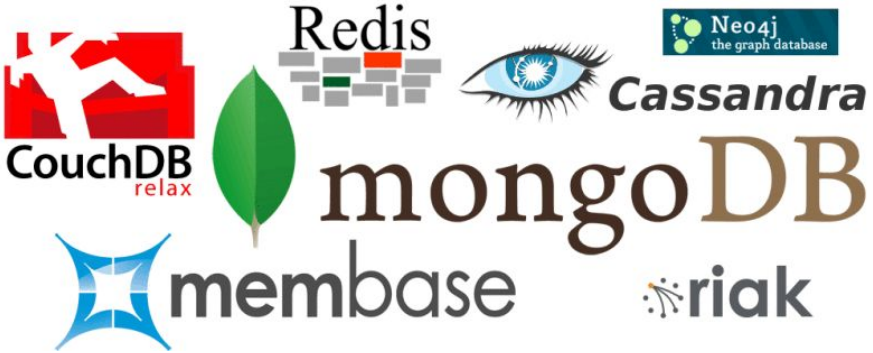
¿Por qué usar bases de datos NoSQL?

👉 Evitar la complejidad innecesaria

👉 Conseguir un alto rendimiento

👉Escalabilidad horizontal y hardware de bajo costo

👉 Transformar la famosa frase: “One size fit’s it all”



Ejemplo de consulta SQL

```
SELECT p.FirstName, p.LastName, a.City, cd.Detail
FROM Person p
JOIN ContactDetail cd ON cd.PersonId = p.Id
JOIN ContactDetailType cdt ON cdt.Id = cd.TypeId
JOIN Address a ON a.PersonId = p.Id
```

Ejemplo de consulta no SQL

```
{
        "Id": "1",
        "firstName": "Thomas",
        "lastName": "Andersen",
        "addresses":  [
              {
                    "line1": "100 Some Street",
                    "line2": "Unit 1",
                    "city": "Seattle",
                    "state": "WA",
                    "zip": "98012"
              }
        ],
        "contactDetails": [
              {"email": "thomas@anderson.com"},
              {"phone": "*1 555 555-5555", "extension": 5555},
        ]
}
```

Algunas bases NoSQL

MongoDB es una base de datos orientada a **documentos**. Esto quiere decir que en lugar de guardar los datos en registros, guarda los datos en documentos. Estos documentos son almacenados en **BSON**, que es una representación binaria de **JSON**.

Apache Cassandra se trata de un software NoSQL distribuido y basado en un modelo de almacenamiento de **«clave-valor»**, de código abierto que está escrita en Java. Permite grandes volúmenes de datos en forma distribuida.

Algunas bases NoSQL

Redis es un motor de base de datos en memoria, basado en el almacenamiento en tablas de hashes **(clave/valor)** pero que opcionalmente puede ser usada como una base de datos durable o persistente.

Neo4j es una base de datos open-source **orientada a grafos escrita en java**. Con este tipo de base de datos NO SQL puedo guardar información en formato de nodos y relacionales.

Otras: Hbase, CouchDB, No Sql Cloud

Ventajas y desventajas de bases SQL

Ventajas	Desventajas
Simplicidad del modelo: Muy simple, no requiere consultas complejas	**Mantenimiento:** dificil por acumulacion de datos en el tiempo
Fácil uso: usuarios pueden acceder/recuperar fácilmente la información requerida en segundos sin caer en la complejidad.	**Costo:** se generan costos fijos y variables por mantenimiento
Precision: bien definidas y organizadas, no duplicados.	**Almacenamiento físico:** requiere mucha memoria física.
Integridad de datos: brindan coherencia en todas las tablas.	**Poca escalabilidad:** los datos no son escalables en diferentes servidores de almacenamiento físico
Normalización: se divide la información en partes manejables para reducir el tamaño del almacenamiento	**Estructura compleja:** sólo puede almacenar datos en forma tabular, dificultando representación compleja.
Colaboración: muchos usuarios interactuando al tiempo	**Reducción de performance en tiempo:** mayor complejidad
Integridad y Seguridad: Sistemas medianamente confiables	**Menor tiempo de respuesta:** muchos datos poca eficiencia

Ventajas	Desventajas
Modelo flexible: puede almacenar y combinar cualquier tipo de datos, tanto estructurados como no estructurados	**Falta de estandarización:** No existe un estándar que defina reglas y roles de las bases de datos NoSQL.
Modelo de datos en evolución: permite actualizar dinámicamente el esquema para evolucionar con los requisitos cambiantes sin interrupciones.	**Algunos problemas de backup:** No está del todo desarrollado este ámbito en este tipo de bases de datos.
Fácil escalamiento: pueden escalar para adaptarse a cualquier tipo de crecimiento de datos manteniendo un bajo costo.	**Consistencia:** NoSQL prioriza la escalabilidad y el rendimiento, pero cuando se trata de la consistencia de los datos no es tan eficiente.
Alto performance: gran rendimiento, medido en términos de rendimiento y latencia (retraso entre la solicitud y la respuesta real).	**Difícil mantenimiento:** pueden llegar a ser costosos y requerir de personal especializado
Acceso libre: no requieren tarifas de licencia costosas y pueden ejecutarse en hardware económico, lo que hace que su implementación sea rentable.	**Poco nivel de madurez:** Son relativamente más nuevas que las bases relacionales por ende tienen todavía mucho por mejorar.

Para pensar

¿Qué casos de implementación en la industria conoces o has escuchado hablar, tanto de SQL como de NoSQL en empresas?

Escribe en el chat de Zoom

Break

¡10 minutos y volvemos!

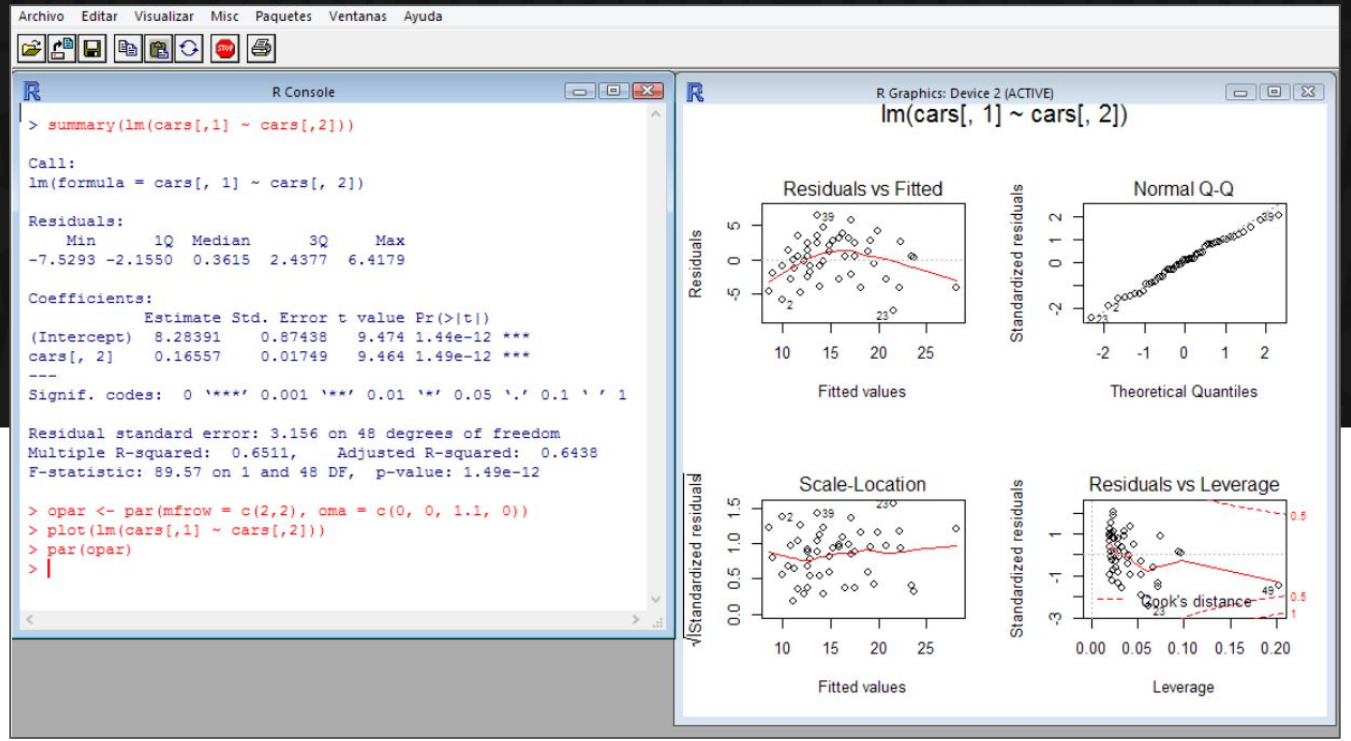
Lenguajes
Data Science

R

R

Entorno y lenguaje de programación con un enfoque al análisis estadístico. Se trata de uno de los lenguajes de programación más utilizados en investigación científica. **R proporciona una amplia variedad de técnicas estadísticas** (modelos lineales y no lineales, pruebas estadísticas clásicas, análisis de series temporales, clasificación, agrupamiento, etc), generación de gráficos y es altamente extensible.

¿Cómo se visualiza R?



CODERHOUSE

¿Un poco rústico verdad?

La realidad, es que la interfaz gráfica de R, no es realmente muy atractiva e intuitiva 😏
Es por ello, que **para nuestras clases prácticas trabajaremos con RStudio, el IDE de R**.

Entorno de desarrollo integrado

¿Qué es un IDE?

Un entorno de desarrollo integrado o entorno de desarrollo interactivo, en inglés Integrated Development Environment, **es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software.**
En este caso en particular RStudio, nos brindará una interfaz mucho más cómoda y amigable para trabajar con R.

Beneficios de RStudio

Las más importantes a mencionar son:

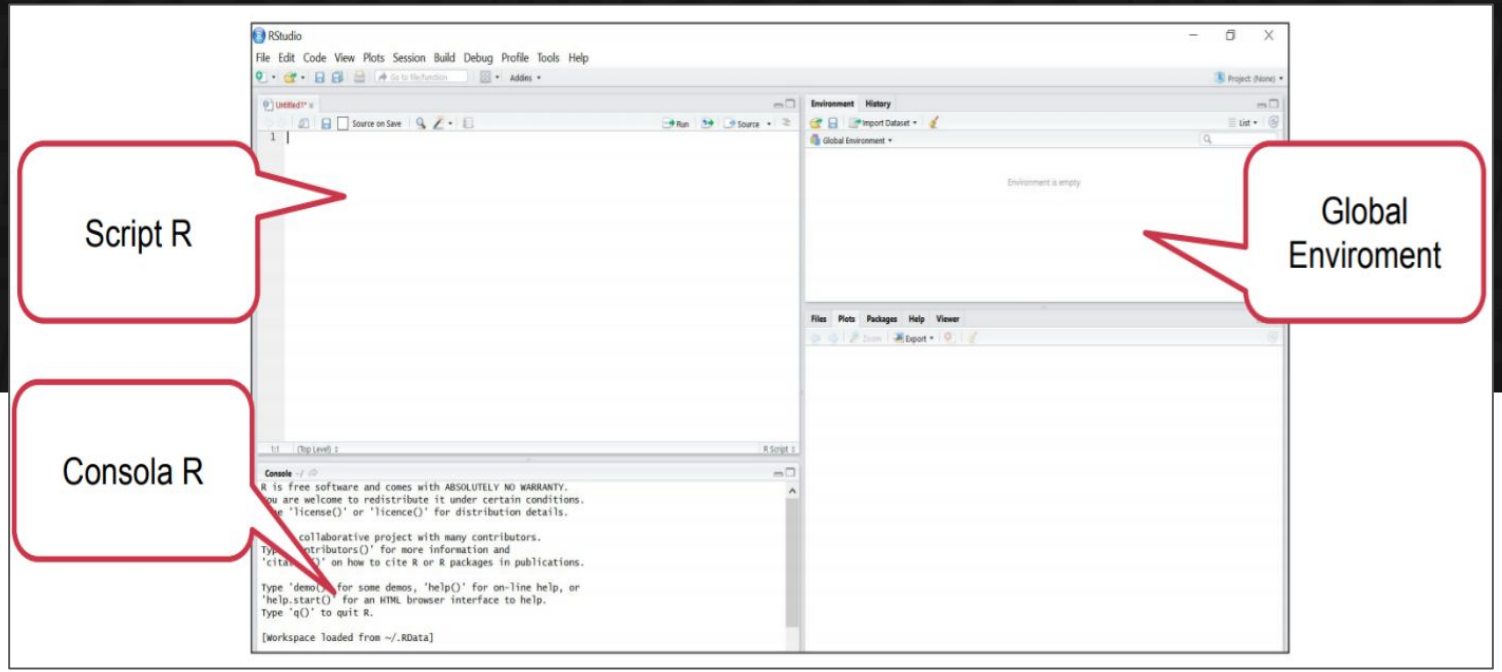
👉 Autocompletado.

👉 Reconocimiento de sintaxis de programación.

👉 Depurador de errores.

👉 Manual de usuarios y ayuda en línea.

¿Cómo se visualiza RStudio?



Link de descarga: https://rstudio.com/products/rstudio/download/

CODERHOUSE

Python

Python

Lenguaje de programación poderoso y fácil de aprender. Puede ser clasificado como un lenguaje interpretado (ejecuta las instrucciones a medida que las va leyendo) y de alto nivel. Python fue creado a finales de los años 80, por un programador holandés llamado Guido van Rossum, quien sigue siendo aún hoy el líder del desarrollo del lenguaje.

En este curso, aprenderemos a utilizarlo.

PYTHON

CODERHOUSE

R vs Python

Características	R	Python
Alcance	(Principalmente) Análisis de datos y modelado estadístico	Diferentes propósitos: desarrollo de aplicaciones web, ciencia de datos
Usuarios	Estadísticos, Investigadores, Analistas y Científicos de Datos.	Desarrolladores, Ingenieros y Científicos de Datos.
Flexibilidad	Librerías disponibles fáciles de usar.	Fácil para construir nuevos algoritmos desde el inicio.
Paquetes esenciales	Tydyverse, caret, ggplot2	Numpy, pandas, scipy, scikitlearn
Herramientas de visualización	Ggplot2, plotly, ggmap	Matplotlib, ploty, seaborn

Visualización de datos

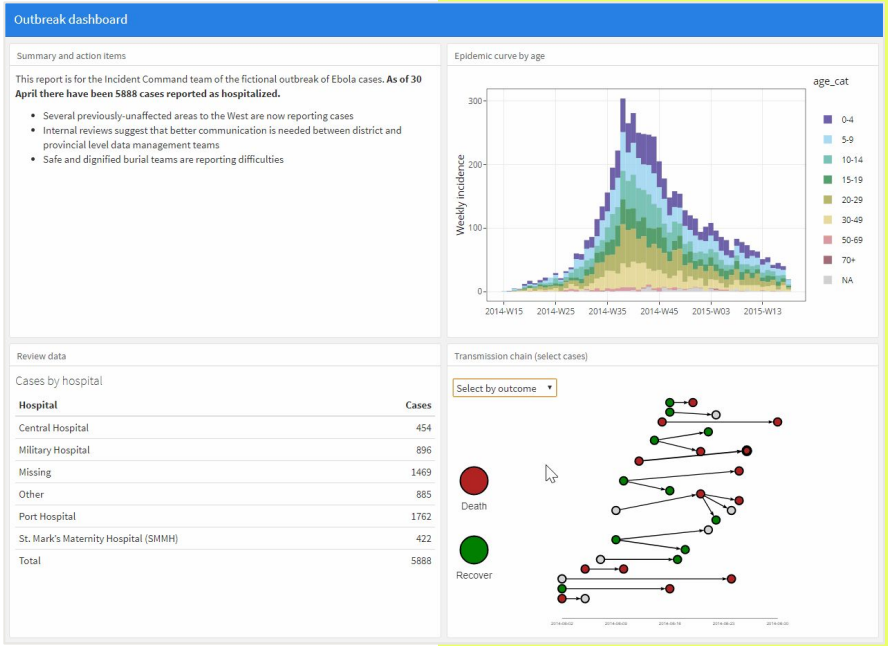
Herramientas de visualización de datos

- Dash
- Shiny
- Power BI
- Pentaho
- Tableau




Dash

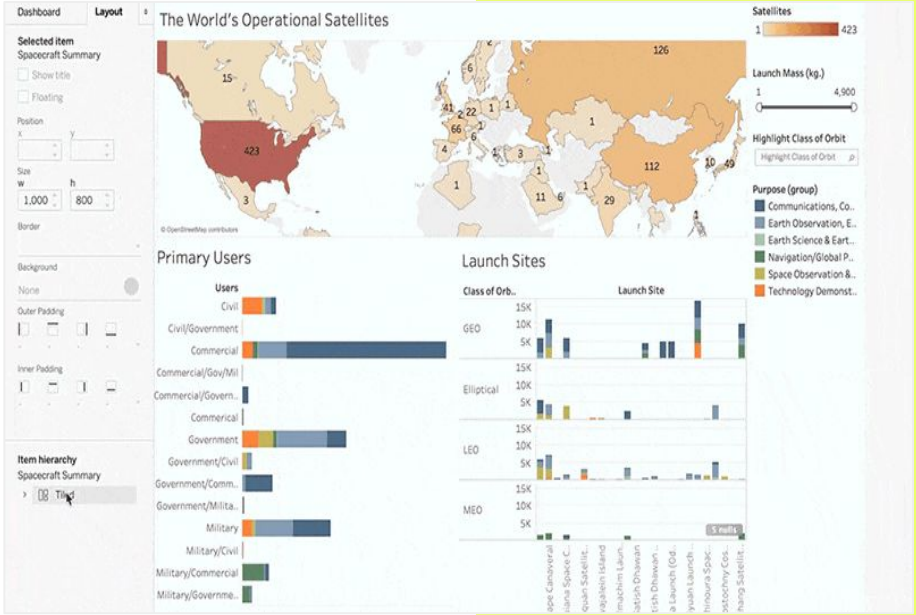
Framework de Python creado por plotly para crear aplicaciones web interactivas. Código de Flask, Plotly.js y React.jsy sin tener que aprender HTML, CSS y Javascript. Dash es de **código abierto** y la creación de la aplicación utilizando un framework se ve en el navegador web.



CODERHOUSE

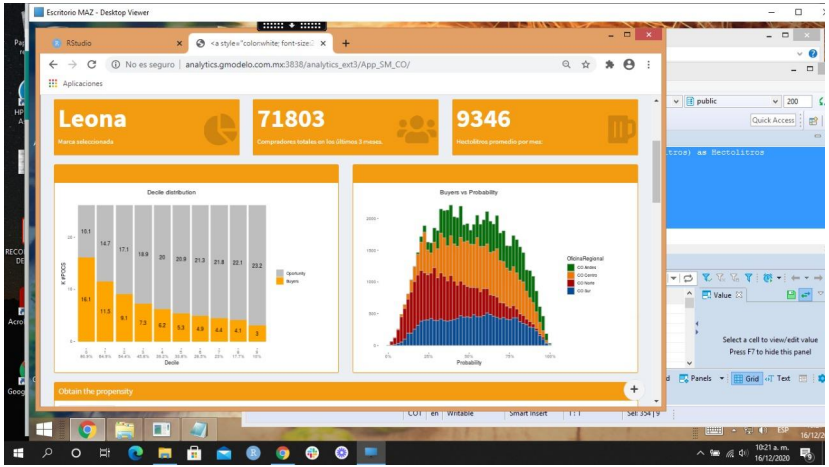
Shiny

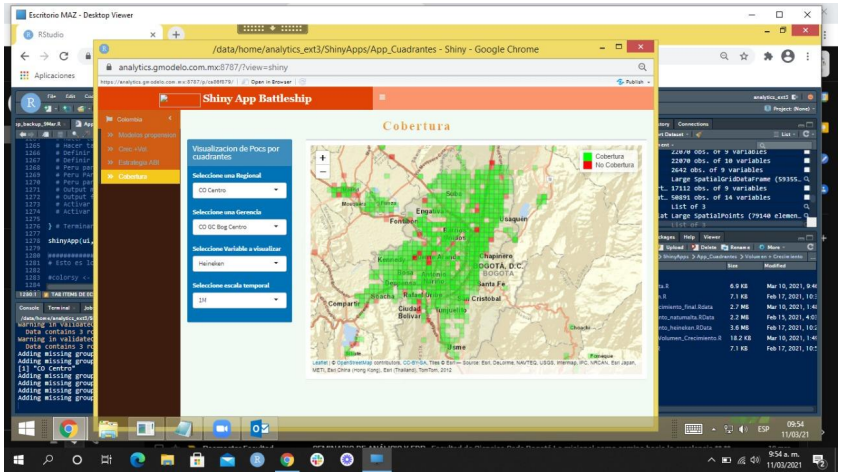
Shiny es un paquete de R que facilita la creación de aplicaciones web interactivas desde R. Puede alojar aplicaciones independientes en una página web o incrustarlas en documentos de R Markdown o crear paneles. Permite el uso de temas CSS, widgets html y acciones de JavaScript.



CODERHOUSE

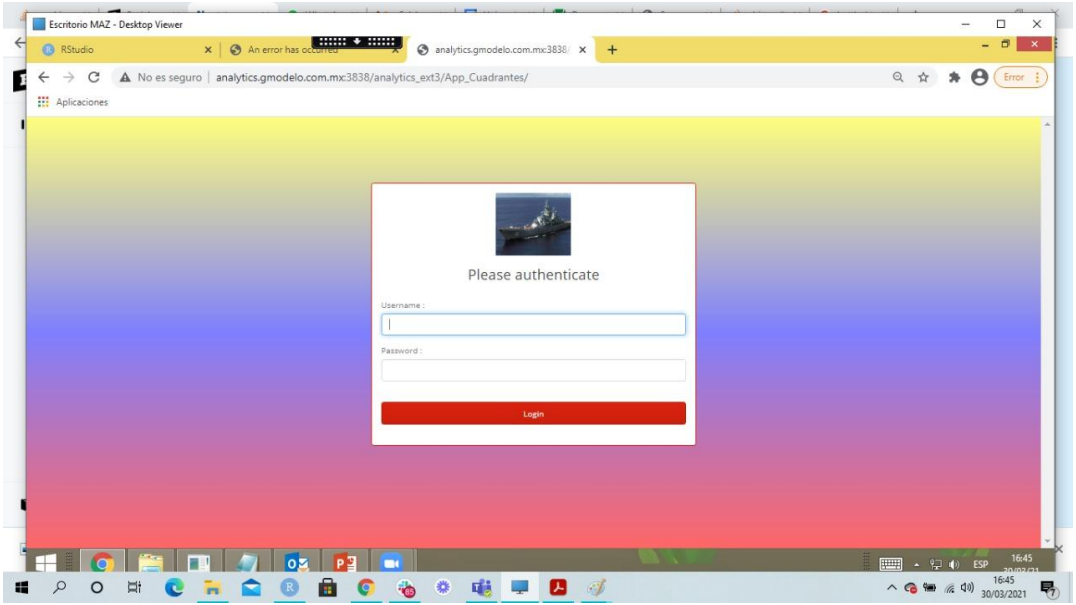
Ejemplos reales





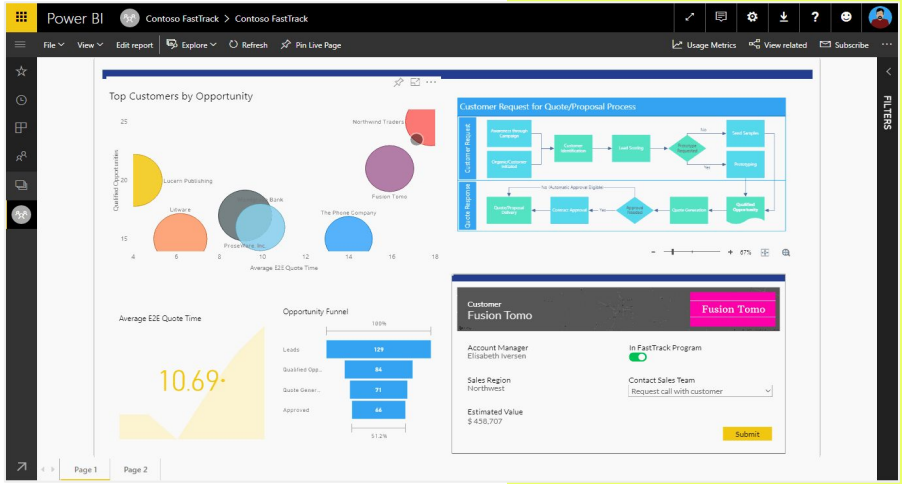
CODERHOUSE

Ejemplos reales



Power BI

Conjunto de herramientas y servicios de business intelligence,
que permite conectarse a diferentes orígenes de datos, para ser analizados, visualizarlos y compartirlos con toda la organización y clientes. Se compone de varias aplicaciones y servicios (Versión desktop, mobile y el servidor).

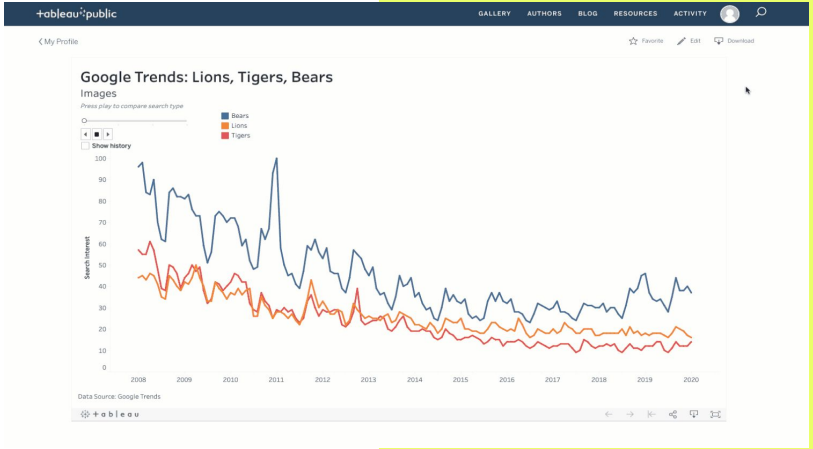


CODERHOUSE

Ejemplo

Tableau

Herramienta de visualización de datos potente también utilizada en el área de la Inteligencia de negocios. La esencia de Tableau es simple y a la vez muy relevante: ayudar a las personas y empresas a ver y comprender todos sus datos.



CODERHOUSE

Tableau

Tableau funciona a través de 3 medios principales:

- ✓ Escritorio (Tableau Desktop)
- ✓ Servidor (Tableau Server)
- ✓ En línea (Tableau Online)

Además integra otras herramientas adicionales para proporcionar una experiencia más completa a los usuarios:

- ✓ Tableau Mobile.
- ✓ Tableau Public.
- ✓ Tableau Prep.

Git vs Github

Diferencias Git vs Github

Git	Github
Instalado localmente	Hosteado en la nube
Lanzado en 2005 y mantenido por The Linux Foundation	Lanzado en 2008 y mantenido por Microsoft
Se enfoca en control de versiones y código compartido	Se enfoca en un hosting de código centralizado
Se ejecuta en la terminal	Administrado en la web
Provee una interface desktop llamada Git Gui	Provee una interfaz desktop llamada Github Desktop
Pocas configuraciones externas disponibles	Marketplace diverso para integración de herramientas
Licencia Open Source	Versión gratuita y también paga

Ejemplo en vivo

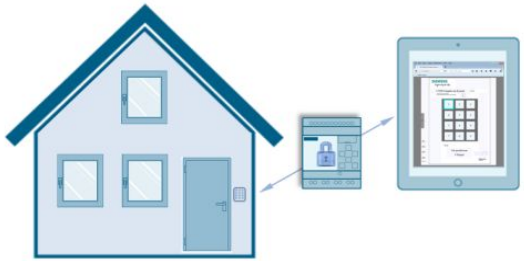
Vamos a ver como crear una cuenta en Github
y el paso a paso para subir allí información.

Duración: **5 minutos**

CODERHOUSE

Sistemas in-house

Servidor In-House



- ✓ Un servidor In-house es aquel que almacena y mantiene en el espacio físico en un entorno local.
- ✓ Por lo tanto, no hay necesidad de una conexión a Internet para acceder a sus datos.
- ✓ Permite el control físico sobre su copia de seguridad en todo momento.
- ✓ Pueden ser viables para pequeñas empresas

Ventajas y desventajas In-House

Ventajas	Desventajas
Seguridad. Manejo de datos sensibles sin terceros	Requiere de grandes inversiones iniciales para infraestructura y hardware
Sin tantos gastos fijos (pago a terceros por servicio)	Gastos variables por consumo y espacio además de mantenimiento
Posibilidad de autogestión dentro de la compañía.	Tiene una carga máxima (capacidad de procesamiento)
No requiere de conexión a Internet (no es limitante velocidad de conexión)	Requiere de personal capacitado para resolver problemas
Hay control sobre la infraestructura (control de acuerdo a las necesidades)	Pierde flexibilidad (acceso remoto) además de ser susceptible a perdida de datos (daños estructura)

Fuente: Kharb L. et al. (2018. A Comprehensive Study of Security in Cloud Computing)

Resumen: Ventajas in-house

Customización: Con nuestra propia infraestructura de redes y de datos **podemos obtener una mayor customización** de recursos por ejemplo en base a los requerimientos de la organización.

Conocimiento: El conocimiento de los sistemas y datos reside únicamente in-house. Esto **puede ser preferible para compañías con requerimientos muy específicos de seguridad**.

Sin costos mensuales: Usualmente, los planes de deployment on-premise se estructuran en planes anuales o multi-anuales, **potencialmente eliminando la necesidad de costos mensuales.**

Resumen: Desventajas in-house

Elasticidad: **Falta de elasticidad para el uso de más recursos** en caso de ser necesario dado un pico de demanda.

Costos de Infraestructura: Los costos de infraestructura para montar un **datacenter propio son mucho más superiores que si adquirimos una licencia** con algún proveedor de servicios Cloud.

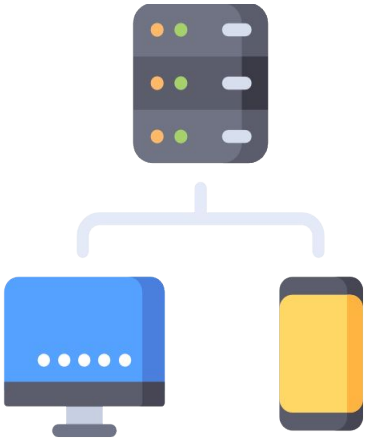
Personal altamente capacitado: Si quisiéramos tener nuestra infraestructura In – House **deberemos contar con personal altamente capacitado en temas de Seguridad e Infraestructura informática.**

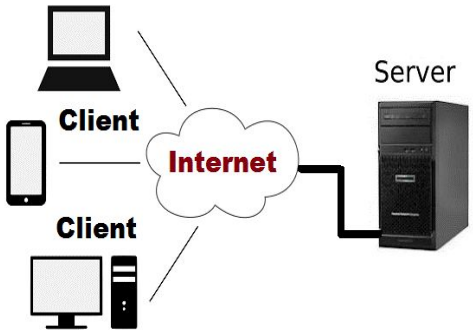
Cloud Computing

Cómputo en la nube

Cloud Computing es una tendencia en alza especialmente dentro del mundo de la Analítica de Datos y como Data Scientist resulta realmente muy importante que podamos entender qué es el **Cloud, sus características y cuáles son los diferentes** servicios que podemos utilizar en la nube para complementar nuestro Stack Tecnológico como Científicos de Datos.

Arquitectura Cliente – Servidor





Claves de la arquitectura
→ Cliente
→ Servidor

CODERHOUSE

¿Qué es el Cloud Computing?

Nueva tecnología que nos permite **acceder mediante un sistema remoto, al software, al procesamiento de datos y al almacenamiento de archivos.**

Tipos disponibles

Nube pública **Nube privada**

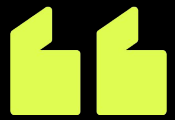
CODERHOUSE

Beneficios de Cloud Computing en Big Data

Existen muchas ventajas, mencionaremos las más relevantes:

- ✔ Abaratamiento de costos
- ✔ Inmediatez
- ✔ Capacidad de proceso
- ✔ Concurrencia
- ✔ Seguridad

CODERHOUSE



“El Cloud Computing, permite la aceleración y gestión de procesos computacionales, haciéndolos más eficientes”

- Observatorio Nacional de Telecomunicaciones y de la SI

Ventajas y desventajas Cloud

Ventajas	Desventajas
Pocos costos operacionales (proveedor se encarga de mantenimiento, equipos y red de servidores)	Sistemas de seguridad que tienen politicas y terminos a veces desconocidos
Hardware y software de punta (CPU, GPU), IA, ML, DL	Esquemas de costos complejos basados en consumo y servicios ocupados
Acceso rápido por conexión con alto Gbps y muchos puntos de intercambio	Existen costos grandes cuando se transfiere la información de una organización a otra
Permite el acceso remoto y conectividad rápida	En algunos casos el soporte al cliente no es óptimo
Infraestructura más segura y sistemas de backup regulares	Requiere de conexiones estables e internet constante para acceso remoto

Fuente: Kharb L. et al. (2018. A Comprehensive Study of Security in Cloud Computing)

¿Preguntas?

CLASE N°2

Glosario

Esquema relacional: desarrollado por Frank Codd en los años 70, es el fundamento de las bases de datos relacionales con prioridad a la consistencia y disponibilidad de los datos

Esquema no relacional : desarrollado por Carlo Strozzi en el 98, es el fundamento de las bases de datos no relacionales con prioridad a la disponibilidad y tolerancia a la partición de datos.

Lenguajes de data science : son aquellos que nos permiten generar estructuras de código para realizar algoritmos, limpiar, estructurar datos (e.g R, Python, Julia, Java, C++)

Herramientas de visualización: son todas aquellas herramientas que permiten generar visualizaciones en entornos gráficos para la presentación de resultados (e.g PowerBI, Shiny, Dash, Pentaho, Tableau)

Muchas gracias.

Resumen
de la clase hoy

- ✓ Bases de Datos Relacionales y No Relacionales
- ✓ Lenguajes de Data Science
- ✓ Visualización de Datos
- ✓ Herramientas Complementarias.

Opina y valora esta clase

#DemocratizandoLaEducación