Data Science puede ayudar a las organizaciones

- Entender su ambiente
- Analizar problemas existentes
- Revelar oportunidades ocultas

Explora los datos de la empresa, buscando la mejor manera de utilizarlos para aportar valor a la empresa.

Qué datos necesitamos para poder resolver los problemas?

Y de dónde viene toda esa información?

Los científicos de datos pueden utilizar datos estructurados y no estructurados, de diferentes fuentes.

Usando diferentes modelos de datos para encontrar patrones, aunque a veces serán nuevos conocimientos, llevando a la empresa a otros enfoques.

El rol de un científico de datos es contar una historia con los datos, comunicando los resultados.

¿Qué es ciencia de datos?

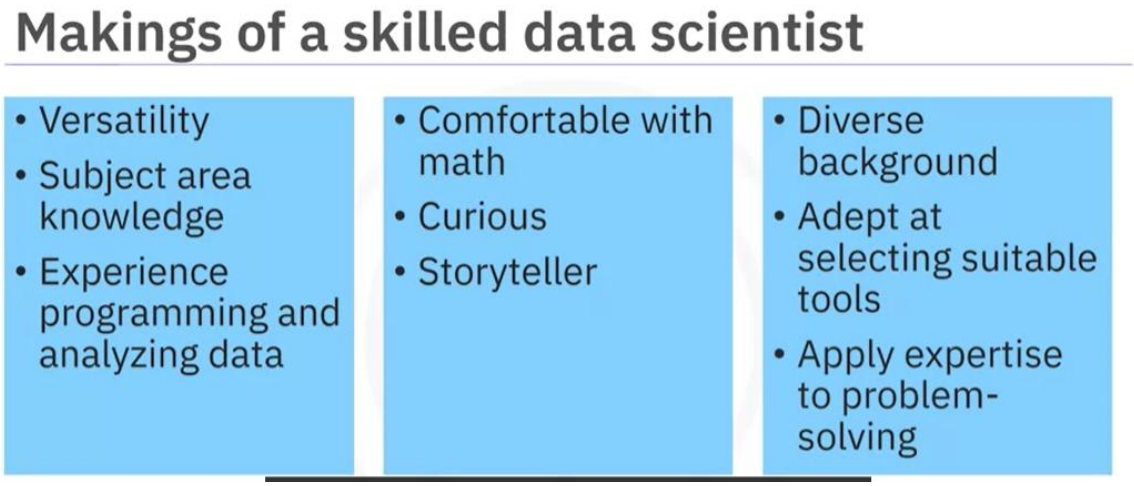
Es la ciencia encargada del estudio de los datos para entender el mundo, es el arte del descubrimiento de nuevas estadísticas y tendencias.

La Ciencia de los datos se centra en la comprensión de los datos, no en la predicción de los avances.

Los científicos de datos traducen los datos en historias para generar perspectivas.

El proceso de recopilar perspectivas para los datos se basa en:

1. Clarificar el problema.
2. Recolectar datos.
3. Analizar
4. Reconocer patrones.
5. Contar una historia con los datos.
6. Visualización.

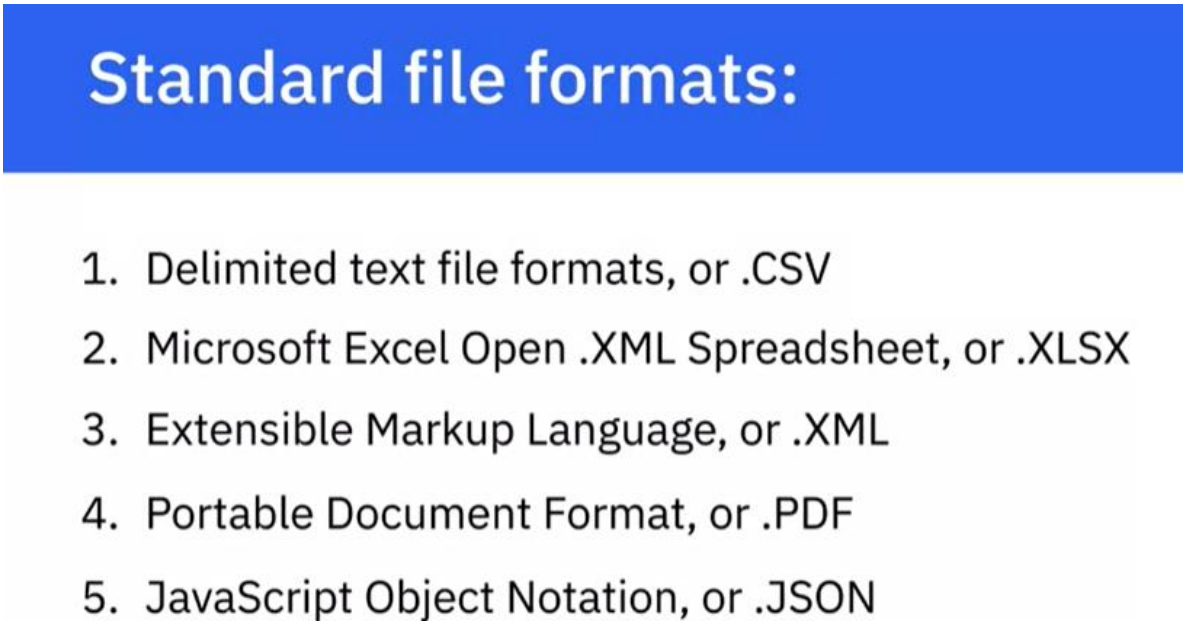


Pensar de manera lógica, recolectar data y analizar de manera metódica y cuidadosa los datos involucrados para así conseguir resultados exitosos para la empresa.

GLOSARIO DE DEFINICIONES CIENCIA DE DATOS

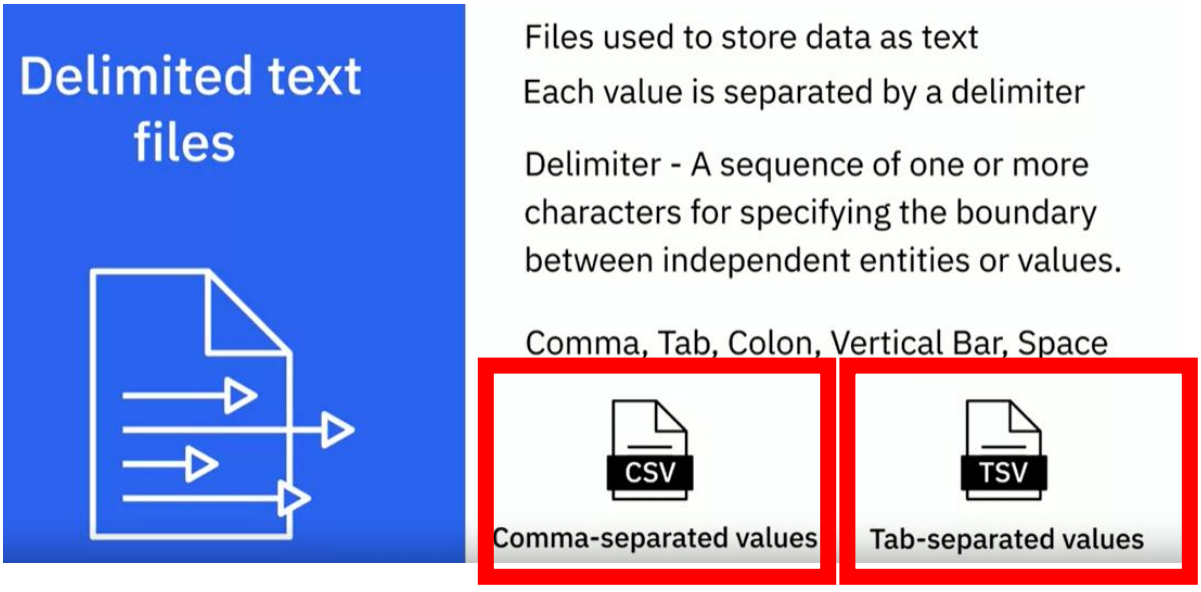
Términos	Definición
Algoritmos	Una instrucción paso a paso para resolver un problema o completar una tarea
Modelo	Una representación de las relaciones y patrones encontrados en los datos para hacer predicciones o analizar sistemas complejos, manteniendo los elementos esenciales necesarios para el análisis.
Outliers (Valores atípicos)	Cuando un punto de datos o varios puntos se encuentran significativamente fuera de la mayoría de los demás datos en un conjunto, lo que podría indicar anomalías, errores o fenómenos únicos que podrían afectar el análisis estadístico o la modelación.
Análisis Cuantitativo	Se utiliza un enfoque sistemático mediante análisis matemático y estadístico para interpretar datos numéricos.
Data estructurada	Los datos se organizan y se formatean en un esquema predecible, generalmente en tablas relacionadas con filas y columnas.
Data no estructurada	Datos desorganizados que carecen de un modelo de datos o una organización predefinida, lo que dificulta su análisis utilizando métodos tradicionales. Este tipo de datos a menudo incluye texto, imágenes, videos y otros contenidos que no encajan fácilmente en filas y columnas como los datos estructurados.

ENTENDIENDO LOS DIFERENTES TIPOS DE FORMATOS EN DOCUMENTOS

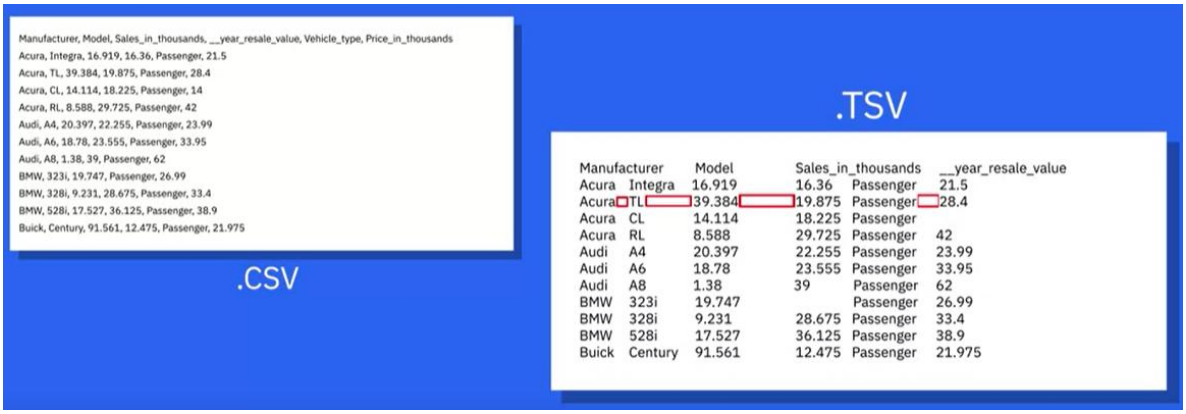


DELIMITED TEXT FILES

1.

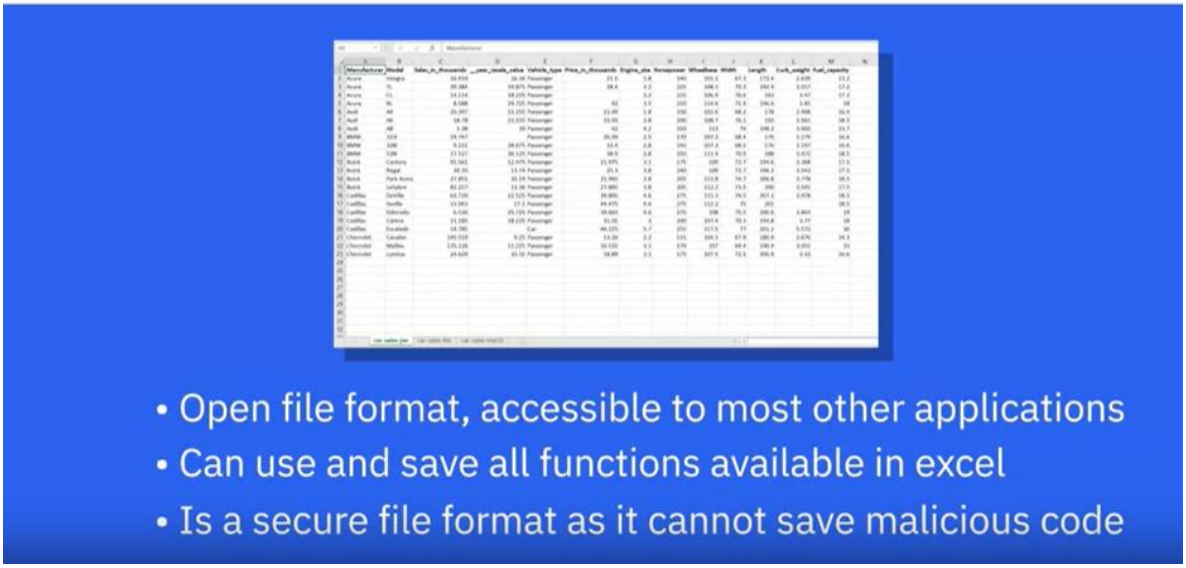


Ej.



2.

Microsoft Excel Open XML Spreadsheet, or .XLSX

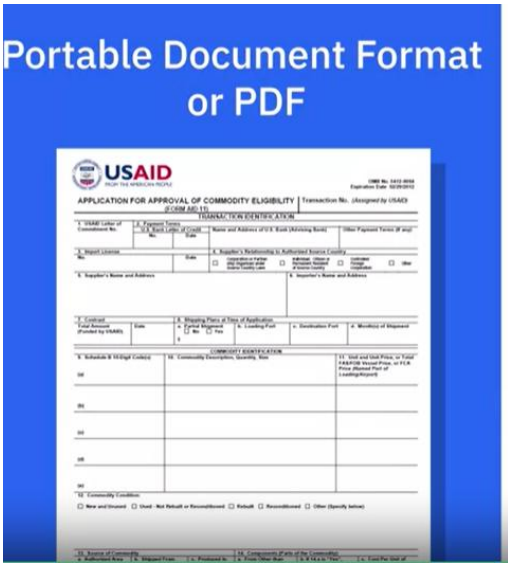


3.

Extensible Markup Language, or XML, is a markup language with set rules for encoding data.

- Readable by both humans and machines
- Self-descriptive language
- Similar to .HTML in some respects
- Does not use predefined tags like .HTML does
- Platform independent
- Programming language independent
- Makes it simpler to share data between systems

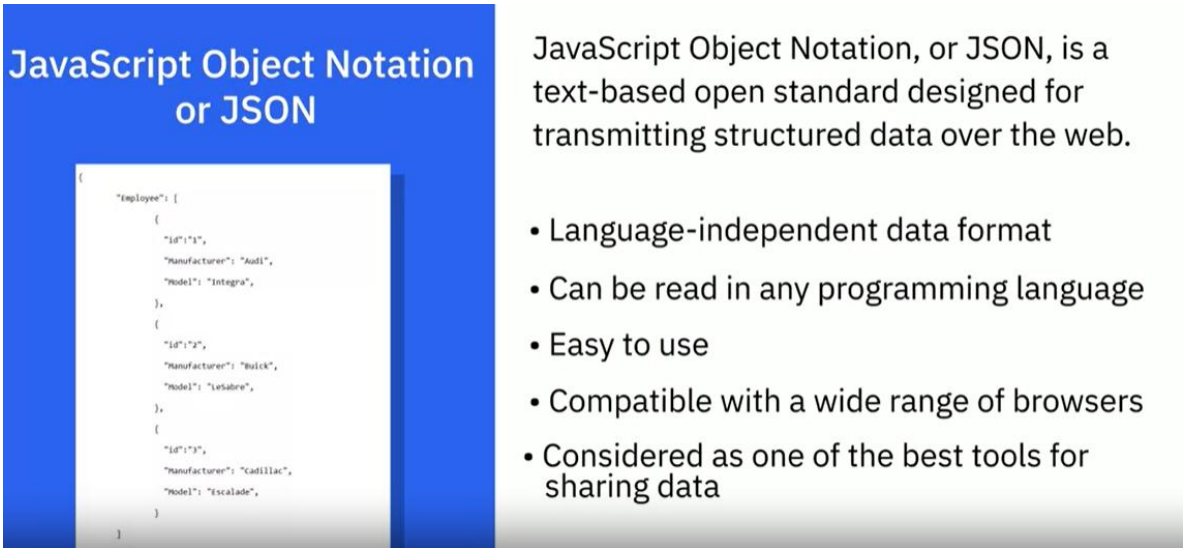
4.



Portable Document Format, or PDF,Is a file format developed by Adobe to present documents independent of application software, hardware, and operating systems.

- Can be viewed the same way on any device
- Is frequently used in legal and financial documents
- Can also be used to fill in data for forms

5.



Puede utilizarse con audio y vídeo.

Utilizar algoritmos complicados de aprendizaje automático **no** siempre garantiza obtener un mejor rendimiento. Ocasionalmente, un algoritmo sencillo como *k-próximo más* cercano puede producir un rendimiento satisfactorio comparable al obtenido utilizando un algoritmo complicado. Todo depende de los datos.

- (•) Cierto.
- ( ) Falso.

Correcto
En cualquier campo, y la ciencia de datos no es diferente, siempre se prefiere una solución sencilla a una complicada, sobre todo si el rendimiento es comparable.

Define a un científico de datos como alguien que puede procesar fácilmente un conjunto de datos de 50 millones de filas en un par de horas y que desconfía de los modelos (estadísticos). Distingue la ciencia de datos de la estadística. Sin embargo, enumera álgebra, cálculo y formación en probabilidad y estadística como antecedentes necesarios para comprender la ciencia de datos.

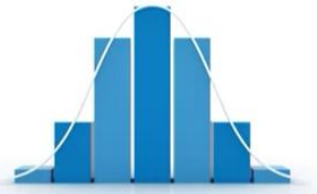
Stata, un software comúnmente utilizado por científicos de datos y estadísticos, anunció que ahora se pueden procesar entre 2 mil millones y 24.4 mil millones de filas utilizando sus soluciones de escritorio.

El otro ingrediente clave para un científico de datos exitoso es un rasgo de comportamiento: la curiosidad. Un científico de datos debe tener una mente muy curiosa, dispuesta a dedicar tiempo y esfuerzos significativos para explorar sus corazonadas.

Mathematical sciences

Algebra

Probability



Calculus

Statistics

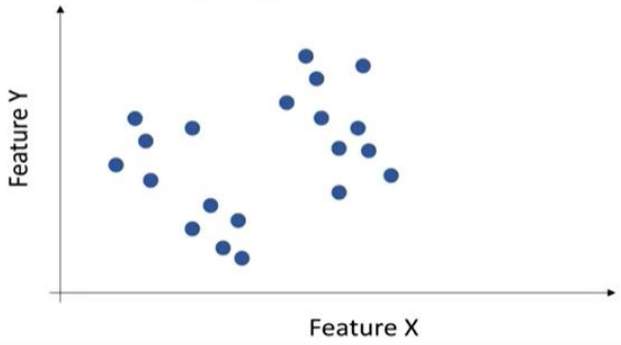
Models and algorithms

Regression

Relationship between two variables

Nearest neighbor

Machine learning algorithm



About the data

Data sources

- Video
- Audio
- Text

Data structure

- Structured
- Unstructured

Data formats

- Text files
- Spreadsheets
- XML
- PDFs
- JSON

Aquí tienes la traducción al español latino (México) de la imagen:

- **Hadoop**: Un framework de código abierto diseñado para almacenar y procesar grandes conjuntos de datos en clústeres de computadoras.

- **JavaScript Object Notation (JSON)**: Un formato de datos compatible con varios lenguajes de programación para que dos aplicaciones intercambien datos estructurados.

- **Jupyter notebooks**: Un entorno computacional que permite a los usuarios crear y compartir documentos que contienen código, ecuaciones, visualizaciones y texto explicativo. Ver cuadernos Python.

- **Nearest neighbor**: Un algoritmo de aprendizaje automático que predice una variable objetivo en función de su similitud con otros valores en el conjunto de datos.

- **Neural networks**: Un modelo computacional utilizado en el aprendizaje profundo que imita la estructura y funcionamiento de las vías neuronales del cerebro humano. Toma una entrada, la procesa usando aprendizaje previo y produce una salida.

- **Pandas**: Una biblioteca de Python de código abierto que proporciona herramientas para trabajar con datos estructurados, y que se utiliza con frecuencia para la manipulación y análisis de datos.

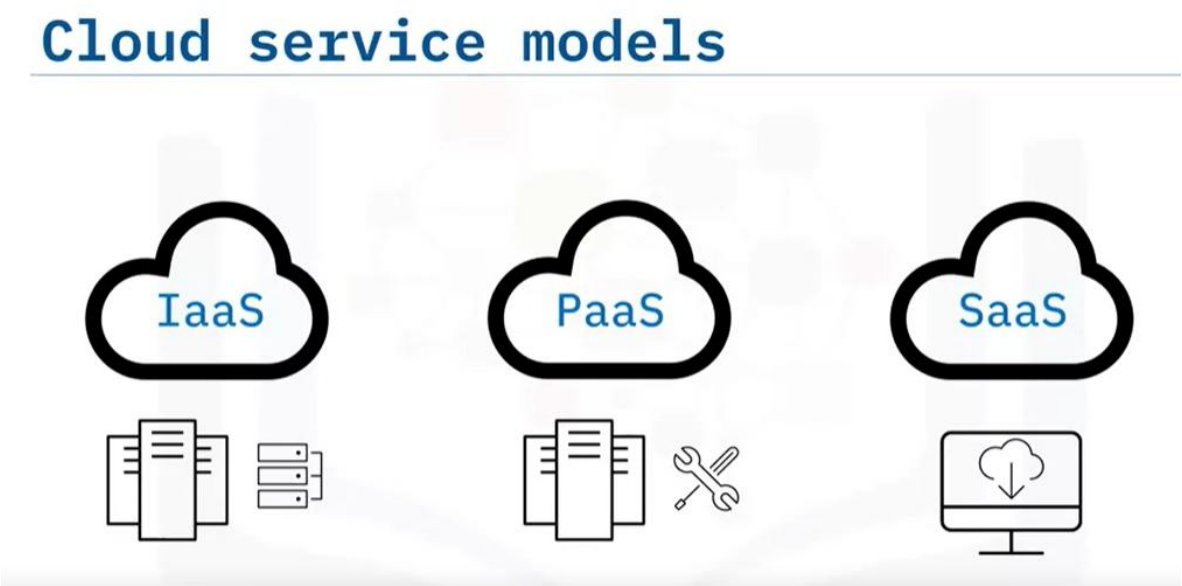
- **Python notebooks**: También conocidos como cuadernos "Jupyter", este entorno computacional permite a los usuarios crear y compartir documentos que contienen código, ecuaciones, visualizaciones y texto explicativo.

- **R**: Un lenguaje de programación de código abierto utilizado para la computación estadística, análisis de datos y visualización de datos.

- **Recommendation engine**: Un programa de computadora que analiza la entrada del usuario, como comportamientos o preferencias, y hace recomendaciones personalizadas en función de ese análisis.

- **Regression**: Un modelo estadístico que muestra una relación entre una o más variables predictoras con una variable de respuesta.

- Los nuevos científicos de datos deben tener curiosidad, criterio y capacidad de argumentación. Self learning and tinkering.
- Algunos términos clave relacionados con la ciencia de datos que ha aprendido en esta lección son: valores atípicos, modelo, algoritmos, JSON, XML. CSV, y regresión.

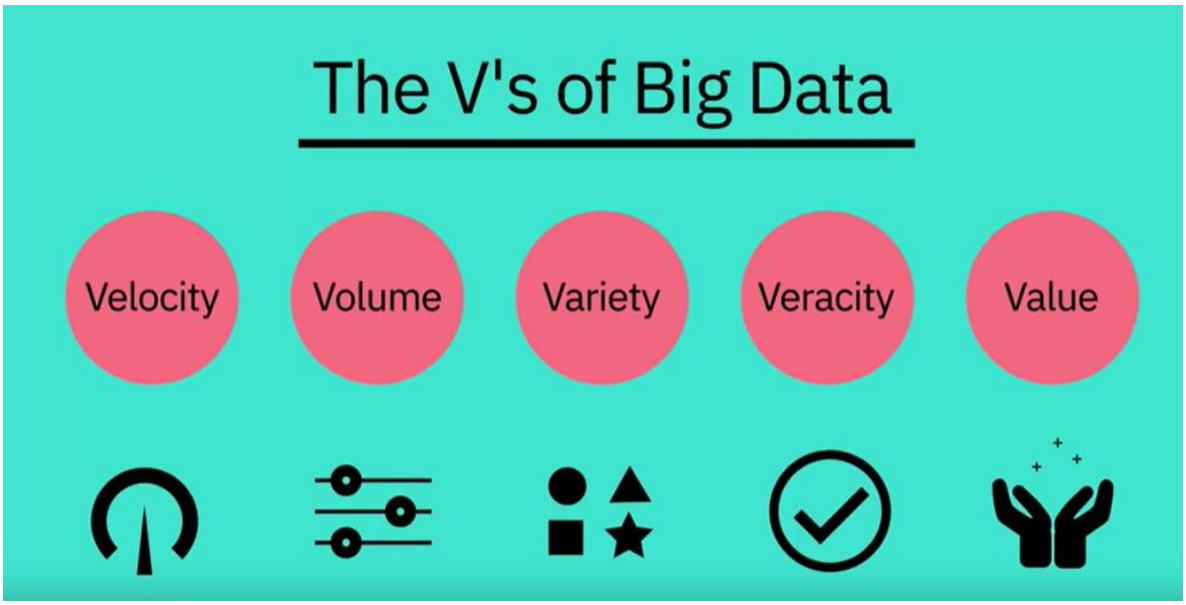


1. IaaS. Infrastructure as a Service. Se refiere a las estructuras físicas de los recursos computacionales, como servidores almacenaje, networking, centros de data sin la necesidad de administrarlos ni operarlos.
2. PaaS. Puedes acceder a la plataforma que incluye las herramientas del hardware y el software que normalmente se utilizan para desarrollar e implementar aplicaciones en el internet.
3. SaaS. Es un modelo de licencia y entrega de software en el que el software y las aplicaciones se alojan de manera centralizada y se licencian mediante su suscripción. Software on-demand.

Resumen

En este video, aprendiste que:

- La computación en la nube es la entrega de recursos de cómputo bajo demanda a través de Internet, pagados según el uso.
- La computación en la nube se compone de cinco características esenciales, tres modelos de implementación y tres modelos de servicio.
- Las cinco características esenciales de la computación en la nube son: autoservicio bajo demanda, acceso a la red amplio, agrupamiento de recursos, elasticidad rápida y servicio medido.
- Los tres modelos de implementación en la nube son: público, privado e híbrido.
- Los tres modelos de servicio en la nube se basan en tres capas en una pila de cómputo: IaaS, PaaS y SaaS.



Velocidad. Es la velocidad en la cual los datos se acumulan, en un proceso que nunca se detiene.

Volumen. Es la escala de los datos, o el incremento en la cantidad de los datos guardados.

Variedades. Es la diversidad de los datos (Datos estructurados y no estructurados) estos vienen de diferentes fuentes, ya sean de redes sociales, imágenes, videos y números, así igual como de máquinas, personas y procesadores. Los datos deberán ser categorizados, analizados y visualizados.

Veracidad. Es la calidad y origen de los datos, precisos, consistencia competencia, integridad y ambigüedad. Los puntos clave serían el costo y necesidad de ser rastreado. En pocas palabras es real o falso.

Valor. Es la necesidad de transformar los datos en algo de valor.

APACHE SPARK & HADOOP. ← HERRAMIENTAS PARA BID DATA

REPASAR MÓDULO 2. BIG DATA PROCESSING TOOLS. HADOOP, HDFS, HIVE & APACHE SPARK.

```python
def build_libraries_from_url(url, name_pos, lat_long_pos):
    import requests
    import json

    r = requests.get(url)
    myjson = json.loads(r.text, parse_constant='utf-8')
    myjson = myjson['data']

    libraries = []
    k = 1
    for location in myjson:
        uname = location[name_pos]
        try:
            latitude = float(location[lat_long_pos][1])
            longitude = float(location[lat_long_pos][2])
        except TypeError:
            latitude = longitude = None
        try:
            name = str(uname)
        except:
            name = "???"
        name = "P_%s_%d" % (name, k)
        if latitude and longitude:
            cp = NamedPoint(name, longitude, latitude)
            libraries.append(cp)
            k += 1
    return libraries
```

Aquí tienes una explicación en lenguaje humano de lo que hace el código en la imagen:

1. **Función Principal**: `build_libraries_from_url(url, name_pos, lat_long_pos)`

   Esta función toma tres argumentos:

   - `url`: una dirección web de donde se obtienen los datos.

   - `name_pos`: la posición dentro de los datos donde se encuentra el nombre de una ubicación.

   - `lat_long_pos`: la posición donde se encuentran las coordenadas de latitud y longitud.

2. **Solicitar Datos**:

- La función comienza realizando una solicitud HTTP para obtener datos desde la URL proporcionada utilizando `requests.get(url)`.

- Los datos obtenidos se decodifican desde formato JSON y se almacenan en `myjson`.

3. **Inicializar Variables**:

- Se crea una lista vacía llamada `libraries` donde se guardarán las ubicaciones procesadas.

- `k` se inicializa en 1; se usará como un contador.

4. **Procesar Datos**:

- La función recorre cada elemento en `myjson`, que representa una ubicación.

- Para cada ubicación:

  - Intenta obtener el nombre de la ubicación desde la posición especificada por `name_pos`.

  - Intenta extraer las coordenadas de latitud y longitud de las posiciones dadas por `lat_long_pos`. Si ocurre un error (por ejemplo, si las coordenadas no están presentes), se establece la latitud y longitud como `None`.

  - Si no se puede convertir el nombre a texto, se asigna un nombre genérico como `"???"` o `P_%s_%d` donde `%s` es el nombre original y `%d` es el contador `k`.

5. **Agregar Ubicación**:

- Si la ubicación tiene latitud y longitud válidas, se crea un objeto `NamedPoint` (que parece representar un punto con nombre y coordenadas).

- Este objeto se añade a la lista `libraries`.

- El contador `k` se incrementa.

6. **Devolver Resultado**:

- La función devuelve la lista `libraries`, que contiene todos los puntos procesados.

En resumen, esta función toma una URL, descarga un conjunto de datos JSON, extrae las ubicaciones con sus nombres y coordenadas, y las convierte en objetos que son almacenados en una lista para su uso posterior.

En programación, el contador k (o cualquier otra variable que elijas) se usa para llevar un registro de la cantidad de veces que algo ha ocurrido. Aquí está el concepto explicado de una manera sencilla:

1. **Inicio del Contador**: Empieza con un valor inicial, como 0 o 1. Este es el punto de partida de tu cuenta.
2. **Incremento del Contador**: Cada vez que algo sucede (por ejemplo, procesas un nuevo elemento en una lista), aumentas el valor del contador en 1. Esto se hace con la operación k += 1, que es una forma abreviada de decir k = k + 1.
3. **Uso del Contador**: El contador te ayuda a llevar un seguimiento de cuántos elementos has procesado o cuántas veces ha ocurrido un evento. Por ejemplo, si estás asignando nombres únicos a objetos, puedes usar el contador para numerarlos secuencialmente.
4. **Ejemplo Práctico**: Si tienes una lista de 5 elementos y usas un contador k para recorrer la lista, empezarías con k = 1 y lo incrementarías a medida que procesas cada elemento. Entonces, k tomaría los valores 1, 2, 3, 4 y 5 a medida que avanzas.

En resumen, k (o cualquier nombre de variable que elijas) es simplemente una herramienta para contar o numerar elementos de manera ordenada en tu programa.

¿Qué significa %d?

- **%d** es un **marcador de posición** que se utiliza para insertar un número entero en una cadena de texto.
- La letra **d** proviene de **"decimal"**, indicando que el valor insertado será un número entero en formato decimal.

name = "P_%s_%d" % (nombre, k)

Explicación paso a paso:

1. **Cadena de formato:** "P_%s_%d"
    - **%s**: marcador de posición para una **cadena de texto**.
    - **%d**: marcador de posición para un **número entero**.
2. **Valores a insertar:** (nombre, k)
    - **nombre**: una variable que contiene una cadena de texto.
    - **k**: una variable que contiene un número entero.
3. **Resultado:** Python reemplaza los marcadores de posición con los valores correspondientes en el orden en que aparecen.

nombre = "biblioteca"

```
k = 5

name = "P_%s_%d" % (nombre, k)

print(name)
```

salida

```
P_biblioteca_5
```

¿Qué sucedió aquí?

- **%s** fue reemplazado por "biblioteca".
- **%d** fue reemplazado por 5.
- La cadena final resultante es "P_biblioteca_5".

¿Por qué usar %d?

- **Formateo claro y conciso:** Permite construir cadenas de texto dinámicamente de una manera organizada y legible.
- **Evita concatenaciones complicadas:** En lugar de concatenar múltiples partes de una cadena con diferentes tipos de datos, el formateo con % simplifica el proceso.

Otros marcadores de posición comunes

- **%s**: para cadenas de texto.
- **%f**: para números de punto flotante (decimales).
- **%e**: para notación científica.

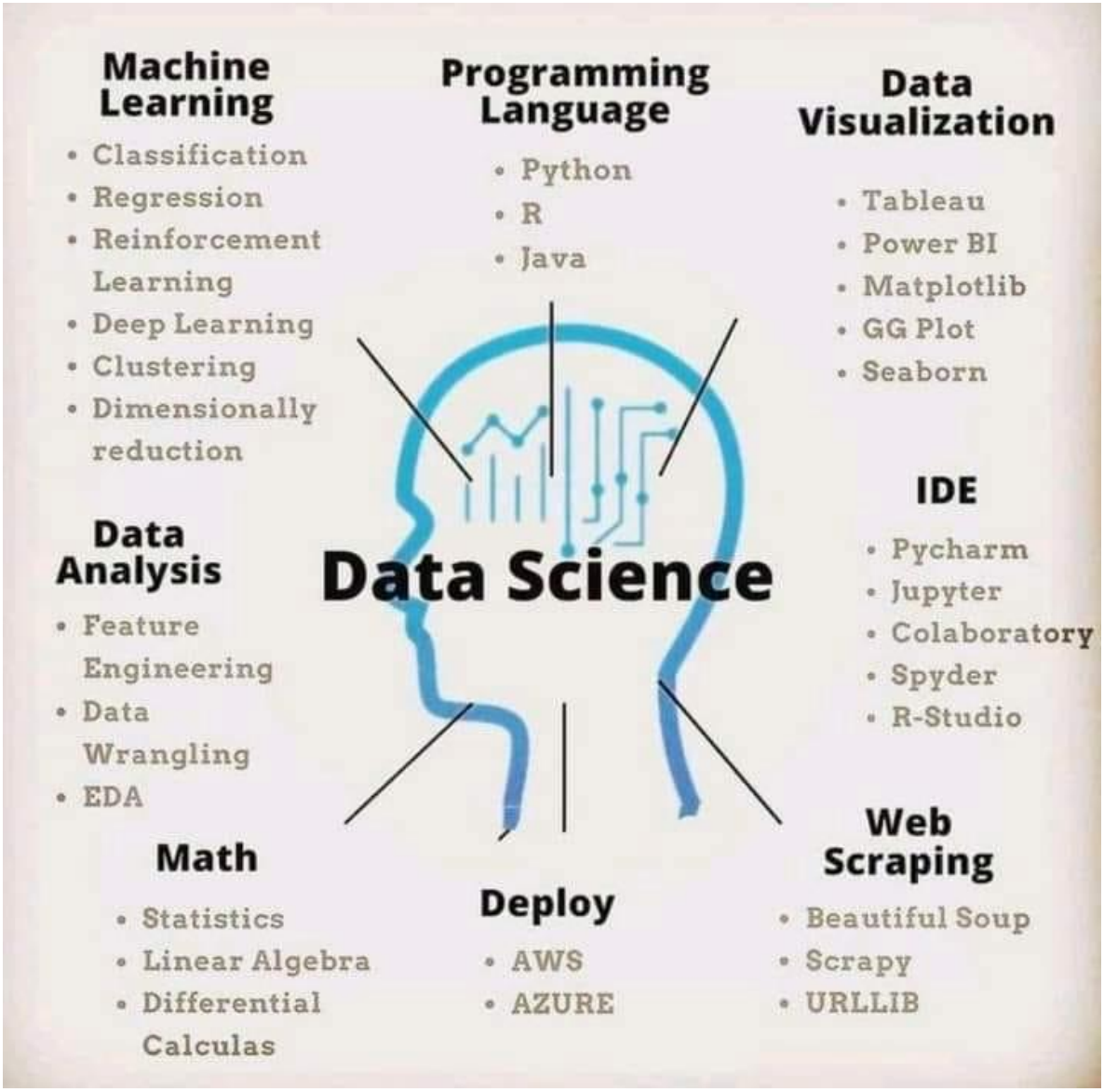
```
nombre = "estación"

temperatura = 23.5

contador = 10


mensaje = "La %s tiene una temperatura de %f grados y un conteo de %d personas." % (nombre, temperatura, contador)

print(mensaje)
```

salida

```
La estación tiene una temperatura de 23.500000 grados y un conteo de 10 personas.
```

Los análisis resumen los hallazgos en tablas y gráficos. El científico de datos debe entonces utilizar los conocimientos adquiridos para construir la narrativa que comunicará dichos hallazgos.

En consultoría y negocios, el entregable final puede presentarse en diversas formas. Puede ser un documento breve de menos de 1,500 palabras ilustrado con tablas y gráficos, o puede ser un documento exhaustivo que comprenda varios cientos de páginas.



1. Imagine que está trabajando en un proyecto de IA que implica la creación de nuevos contenidos como imágenes, música y lenguaje. ¿En qué tecnología de inteligencia artificial se centraría principalmente?

- ○ Aprendizaje automático
- ○ Redes neuronales
- ○ Procesamiento del lenguaje natural (PLN)
- ◉ IA generativa

Correcto
Correcto La IA generativa es el subconjunto de la IA que se concentra en la producción de nuevos datos y contenidos, como imágenes, música, lenguaje, etc.

2. ¿Qué diferencia al aprendizaje profundo de las redes neuronales tradicionales?

- ◉ Múltiples capas de redes neuronales
- ○ Mejora de la eficiencia computacional
- ○ Mayor integración de la computación en nube
- ○ Transformaciones lineales en el análisis de datos

Correcto
Correcto El Aprendizaje profundo se caracteriza por el uso de múltiples capas, lo que permite el reconocimiento de patrones complejos.

3. En el ámbito del aprendizaje automático, ¿qué aplicación significativa implica la tarea de predecir elementos de interés para los usuarios basándose en sus interacciones o comportamientos anteriores?

- ○ Identificación de transacciones fraudulentas en tiempo real
- ◉ Sistemas de recomendación para sugerencias de contenidos personalizados
- ○ Optimización de los algoritmos de clasificación de los motores de búsqueda
- ○ Analizar las tendencias de las ventas al por menor para la gestión del inventario

Correcto
Correcto Los sistemas de recomendación utilizan el aprendizaje automático para predecir y sugerir artículos de interés para los usuarios, como películas o productos.

¿Cuál es la finalidad última de la analítica en el contexto de la aportación de conocimientos y conclusiones?

- (●) Resumir los resultados en tablas y gráficos para su comunicación.
- ( ) Elaborar diapositivas PowerPoint con gráficos y tablas.
- ( ) Para crear informes exhaustivos con numerosas tablas y gráficos.
- ( ) Generar ensayos e informes en el mundo académico.

Correcto
Correcto El objetivo último de la analítica es resumir los resultados en tablas y gráficos para una comunicación eficaz de las ideas y conclusiones.

¿Cuál es la principal ventaja de utilizar clusters de big data?

- ( ) Simplifican el análisis de los datos al centralizarlos todos.
- ( ) Requieren menos servidores para manejar grandes conjuntos de datos.
- ( ) Reducen la necesidad de realizar análisis estadísticos.
- (●) Permiten una distribución sencilla y un procesamiento paralelo de los datos.

Correcto
Correcto El contenido destaca que los clústeres de macrodatos distribuyen los datos a muchos ordenadores para su procesamiento en paralelo.

BIG DATA = HADOOP & SPARK

La portada debe incluir el título del informe, nombres de los autores, sus afiliaciones y contactos, el nombre de la institución que lo publica (si corresponde) y la fecha de publicación. He visto numerosos informes sin la fecha de publicación, lo que hace imposible citarlos sin el año y mes de publicación. Además, desde un punto de vista empresarial, los autores deben facilitar que el lector se comunique con ellos. Tener los detalles de contacto en la portada facilita la tarea.

"Una tabla de contenido (ToC)" es como un mapa necesario para un viaje nunca realizado. Necesitas tener una idea del viaje antes de emprenderlo. Un mapa proporciona una representación visual del viaje real con detalles sobre los hitos que pasarás en tu camino. La ToC con los encabezados principales y las listas de tablas y figuras ofrece una vista previa de lo que se encontrará en el documento. Nunca dudes en incluir una ToC, especialmente si tu documento, excluyendo la portada, tabla de contenido y referencias, tiene cinco o más páginas de longitud.

Incluso para un documento corto, recomiendo un "resumen" o un **"resumen ejecutivo".** Nada es más poderoso que explicar la esencia de tus argumentos en tres párrafos o menos. Una "sección introductoria" siempre es útil para plantear el problema al lector que podría ser

nuevo en el tema y que podría necesitar ser introducido suavemente al tema antes de ser inmerso en detalles intrincados.

En la sección de "metodología", introduces los métodos de investigación y las fuentes de datos que utilizaste para el análisis. Si has recolectado nuevos datos, explica el ejercicio de recolección de datos en detalle. Te referirás a la revisión de la literatura para respaldar tu elección de variables, datos y métodos y cómo te ayudarán a responder tus preguntas de investigación.

La sección de resultados es donde presentas tus hallazgos empíricos. Comenzando con estadísticas descriptivas, y gráficos ilustrativos, pasarás a probar formalmente tu hipótesis. En caso de que necesites ejecutar modelos estadísticos, podrías recurrir a modelos de regresión.

Ten en cuenta que muchos informes en el sector empresarial presentan los resultados de una manera más digerible al omitir los detalles estadísticos y confiar en gráficos ilustrativos para resumir los resultados.

No todos los análisis devuelven una prueba concluyente. A veces, más frecuentemente de lo que me gustaría admitir, los resultados solo proporcionan una respuesta parcial a la pregunta y eso, también, con una larga lista de advertencias.

En la sección de "conclusión", generalizas tus hallazgos específicos y adoptas un enfoque de marketing para promover tus hallazgos de modo que el lector no quede atrapado en las advertencias que has mencionado voluntariamente anteriormente. También podrías identificar futuros posibles desarrollos en la investigación y aplicaciones que podrían resultar de tu investigación.

Adobe Spark
Un conjunto de herramientas de software que permiten a los usuarios crear y compartir contenido visual como gráficos, páginas web y videos.

Analytical skills
La capacidad de analizar información de manera sistemática, lógica y organizada.

Chief information officer (CIO)
Un ejecutivo empresarial responsable de los sistemas de tecnología de la información de una organización y de las iniciativas relacionadas con la tecnología.

Computational thinking
Descomponer problemas en partes más pequeñas y utilizar algoritmos, lógica y abstracción para desarrollar soluciones. A menudo utilizado, pero no limitado a, la informática.

Data clusters
Un grupo de puntos de datos similares y relacionados, distintos de otros clusters.

Executive summary
Generalmente ubicado al principio de un documento de investigación, esta sección resume las partes importantes del documento, incluidos los hallazgos clave.

High-performing computing (HPC) cluster
Una tecnología informática que utiliza un sistema de computadoras en red diseñado para resolver problemas complejos y computacionalmente intensivos en entornos tradicionales.

Mathematical computing
El uso de computadoras para calcular, simular y modelar problemas matemáticos.

Matrices
Plural de matrix, las matrices son un arreglo rectangular (tabular) de números a menudo utilizado en matemáticas, estadísticas e informática.

Stata
Un paquete de software utilizado para el análisis estadístico.

Statistical distributions
Una forma de describir la probabilidad de diferentes resultados basados en un conjunto de datos. La "curva de campana" es una distribución estadística común.

Structured Query Language (SQL)
Un lenguaje utilizado para gestionar datos en una base de datos relacional.

TCP/IP network
Una red que utiliza el protocolo TCP/IP para comunicarse entre dispositivos conectados en esa red. Internet utiliza TCP/IP.

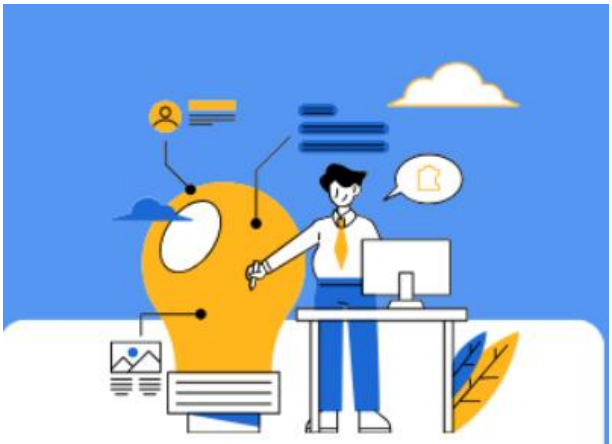
- Un informe claramente organizado y lógico debe comunicar al lector lo siguiente:
    - Qué ganan con la lectura del informe
    - Objetivos claramente definidos
    - La importancia de su contribución
    - El contexto apropiado, aportando suficientes antecedentes
    - Por qué este trabajo es práctico y útil
    - Conjeturar desarrollos futuros plausibles que puedan derivarse de su trabajo

Many Paths

Diverse educational and career backgrounds
Exposure to data challenges sparked interest
Data science is adaptable across professions

Tools & Techniques

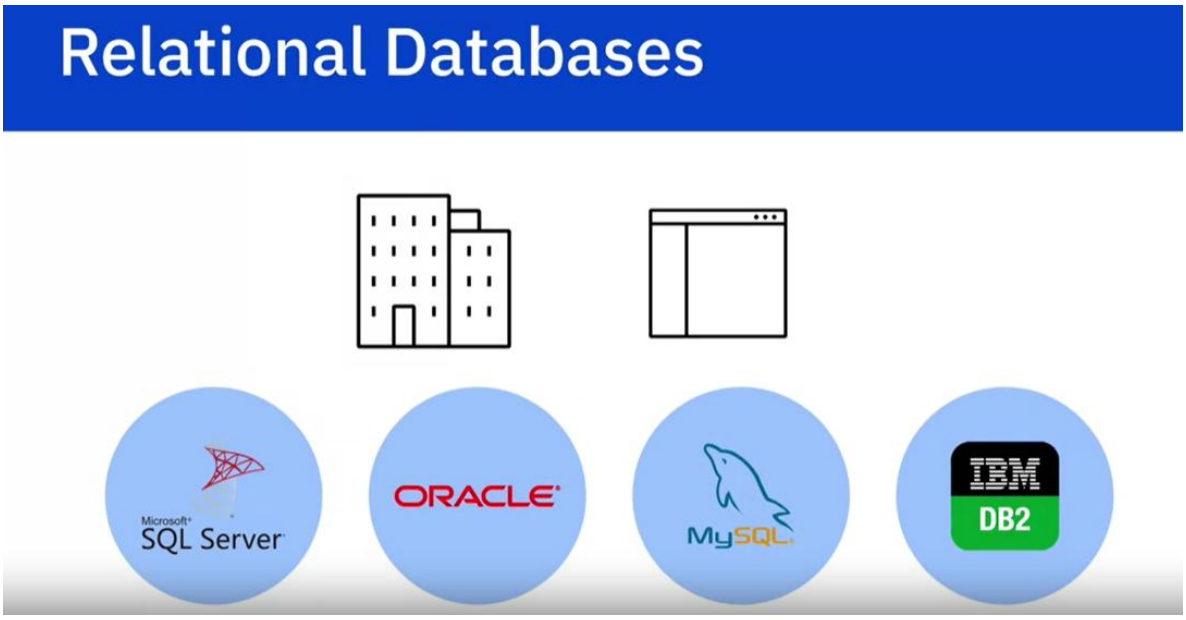
Programming with Python and R
Hadoop
Python libraries: NumPy, pandas, scikit-learn
Data visualization tools
Machine learning algorithms
Data preprocessing techniques



Foundational Skills

Statistical knowledge.
Mathematics, Calculus, Linear Algebra
Exploratory data analysis
Select, train, and test models
Communication and presentation skills

Solving
Identify Patterns
Utilize External Data Sources
Communication of Findings

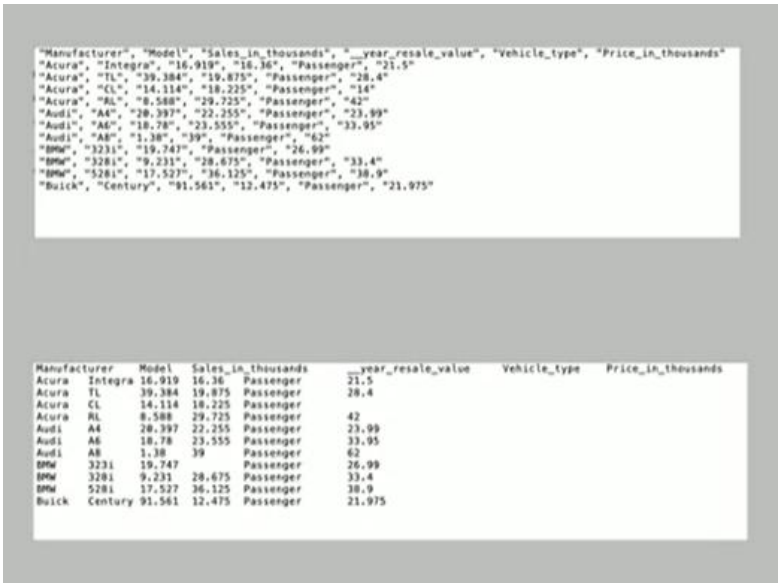

Relational Databases
Microsoft SQL Server
ORACLE
MySQL
IBM
DB2

Storage:


Relational Databases
Business activities
Customer transactions
Human resource activities
Workflows

Flat files

- Store data in plain text format
- Each line, or row, is one record
- Each value is separated by a delimiter
- All of the data in a flat file maps to a single table
- Most common flat file format is .CSV

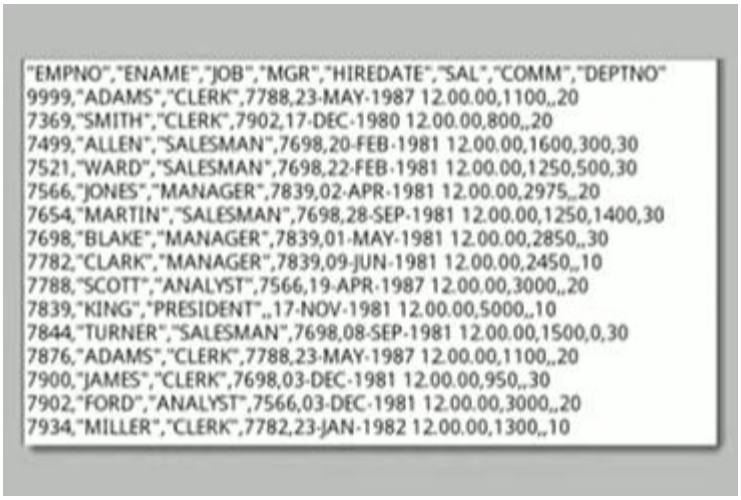


```
"Manufacturer", "Model", "Sales_in_thousands", "__year_resale_value", "Vehicle_type", "Price_in_thousands"
"Acura", "Integra", "16.919", "16.36", "Passenger", "21.5"
"Acura", "TL", "39.384", "19.875", "Passenger", "28.4"
"Acura", "CL", "14.114", "18.225", "Passenger", "14"
"Acura", "RL", "8.588", "29.725", "Passenger", "42"
"Audi", "A4", "20.397", "22.255", "Passenger", "23.99"
"Audi", "A6", "18.78", "23.555", "Passenger", "33.95"
"Audi", "A8", "1.38", "39", "Passenger", "62"
"BMW", "323i", "19.747", "Passenger", "26.99"
"BMW", "328i", "9.231", "28.675", "Passenger", "33.4"
"BMW", "528i", "17.527", "36.125", "Passenger", "38.9"
"Buick", "Century", "91.561", "12.475", "Passenger", "21.975"
```

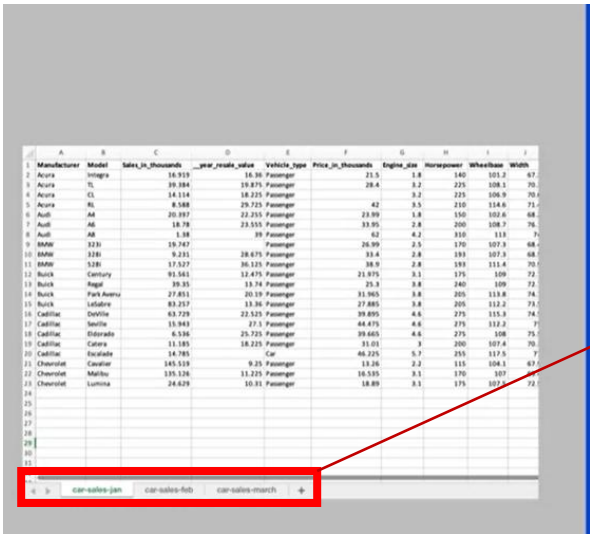
Manufacturer	Model	Sales_in_thousands	__year_resale_value	Vehicle_type	Price_in_thousands
Acura	Integra	16.919	16.36	Passenger	21.5
Acura	TL	39.384	19.875	Passenger	28.4
Acura	CL	14.114	18.225	Passenger	
Acura	RL	8.588	29.725	Passenger	42
Audi	A4	20.397	22.255	Passenger	23.99
Audi	A6	18.78	23.555	Passenger	33.95
Audi	A8	1.38	39	Passenger	62
BMW	323i	19.747		Passenger	26.99
BMW	328i	9.231	28.675	Passenger	33.4
BMW	528i	17.527	36.125	Passenger	38.9
Buick	Century	91.561	12.475	Passenger	21.975

Delimiter = coma, semi colon o espacio

Formato *.CSV= imagen de abajo



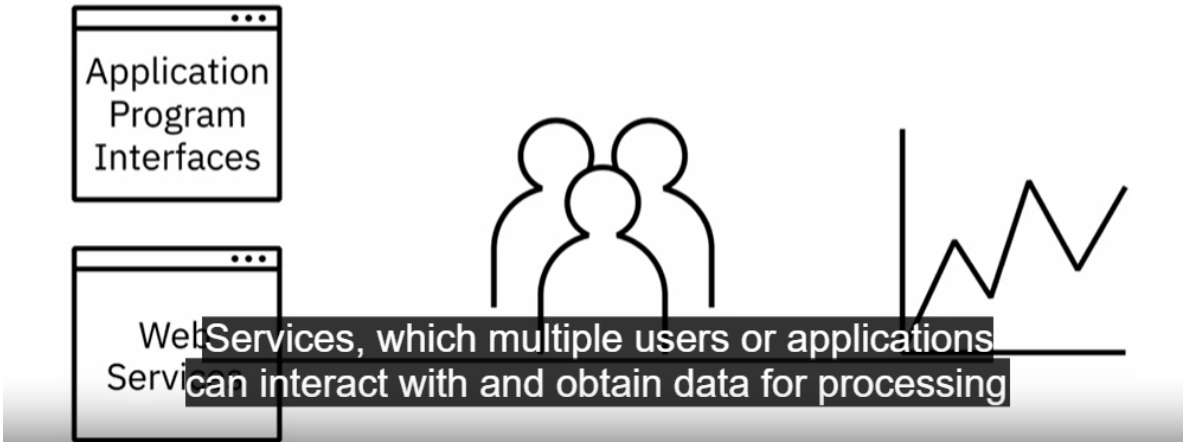
```
"EMPNO","ENAME","JOB","MGR","HIREDATE","SAL","COMM","DEPTNO"
9999,"ADAMS","CLERK",7788,23-MAY-1987 12.00.00,1100,,20
7369,"SMITH","CLERK",7902,17-DEC-1980 12.00.00,800,,20
7499,"ALLEN","SALESMAN",7698,20-FEB-1981 12.00.00,1600,300,30
7521,"WARD","SALESMAN",7698,22-FEB-1981 12.00.00,1250,500,30
7566,"JONES","MANAGER",7839,02-APR-1981 12.00.00,2975,,20
7654,"MARTIN","SALESMAN",7698,28-SEP-1981 12.00.00,1250,1400,30
7698,"BLAKE","MANAGER",7839,01-MAY-1981 12.00.00,2850,,30
7782,"CLARK","MANAGER",7839,09-JUN-1981 12.00.00,2450,,10
7788,"SCOTT","ANALYST",7566,19-APR-1987 12.00.00,3000,,20
7839,"KING","PRESIDENT",,17-NOV-1981 12.00.00,5000,,10
7844,"TURNER","SALESMAN",7698,08-SEP-1981 12.00.00,1500,0,30
7876,"ADAMS","CLERK",7788,23-MAY-1987 12.00.00,1100,,20
7900,"JAMES","CLERK",7698,03-DEC-1981 12.00.00,950,,30
7902,"FORD","ANALYST",7566,03-DEC-1981 12.00.00,3000,,20
7934,"MILLER","CLERK",7782,23-JAN-1982 12.00.00,1300,,10
```



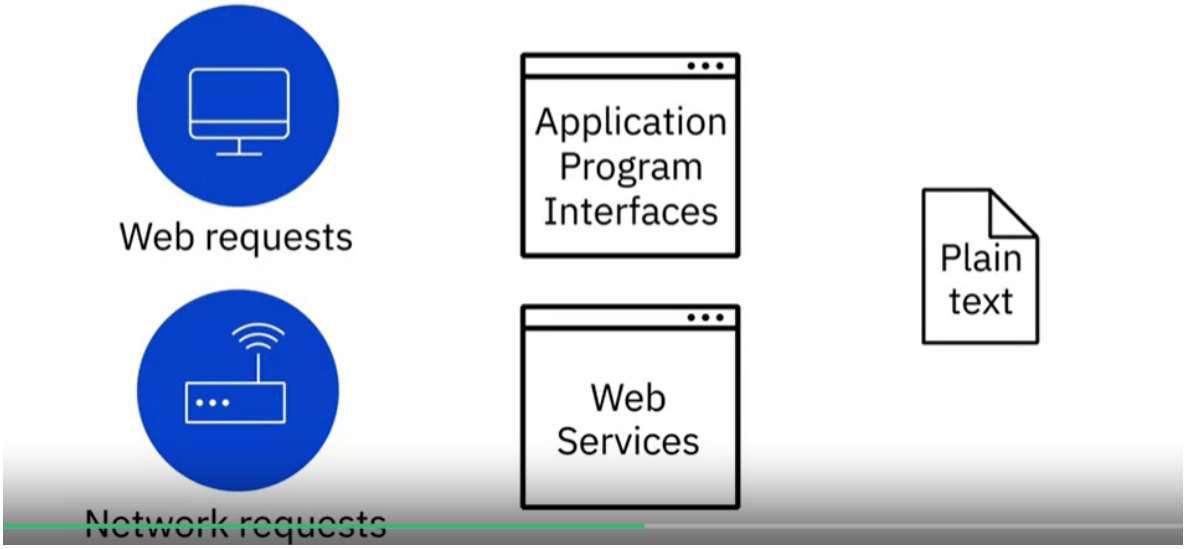
Spreadsheet files

- Special type of flat files
- Organize data in a tabular format
- Can contain multiple worksheets
- .XLS or .XLSX are common spreadsheet formats
- Other formats include Google Sheets, Apple Numbers, and LibreOffice Calc

APIs and Web Services



APIs and Web Services



Popular examples of APIs

Twitter and Facebook APIs
for customer sentiment analysis

Stock Market APIs
for trading and analysis

Data Lookup and Validation APIs
for cleaning and co-relating data

Web scraping

- Extract relevant data from unstructured sources
- Also known as Screen scraping, Web harvesting, and Web data extraction
- Downloads specific data based on defined parameters
- Can extract text, contact information, images, videos, product items, and more...

Web scraping

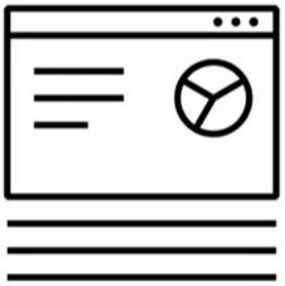
Popular uses:

- Providing price comparisons by collecting product details from retailer, manufacturers, and eCommerce websites
- Generating sales leads through public data sources
- Extracting data from posts and authors on various forums and communities
- Collecting training and testing datasets for machine learning models

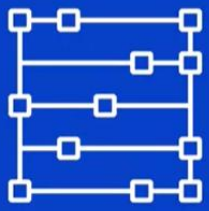
Web scraping

Popular web scraping tools:

- BeautifulSoup
- Scrapy
- Pandas
- Selenium



Data Streams and feeds



- Social media feeds for sentiment analysis
- Sensor data feeds for monitoring industrial or farming machinery
- Web click feeds for monitoring web performance and improving design
- Real-time flight events for rebooking and rescheduling

Popular technologies used to process data streams include:

kafka

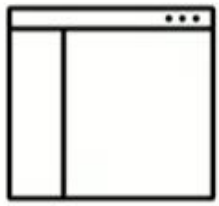
APACHE Spark™

APACHE STORM™

RSS (or Really Simple Syndication) feeds

Capturing updated data from online forums and news sites where data is refreshed on an ongoing basis.

Online forums



News sites

- Using SQL for:
  - Moving relational data from one place to another
  - Structuring data
  - Securing data

Working with multiple data sources, types, and formats

- You will need to work with data at rest, streaming data or data in motion
- You might not have the skills to work with different data sources to begin with, but you need to be able to learn as you go
- Log data is unstructured and you may need to write your own tools to parse the data

- The data had a lot of different characters in it which made it challenging to get a character that could be used as a delimiter
- We had to use different separators for different tables

Unstructured data

Heterogeneous

Comes from broad range of sources

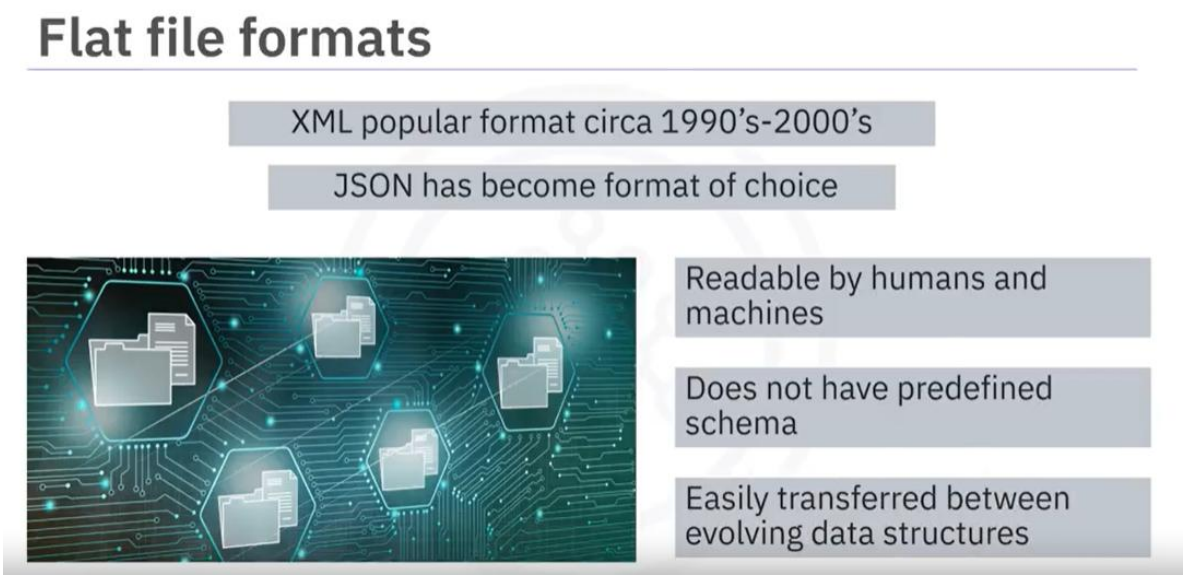
Variety of business intelligence and analytics applications

Analysis requires artificial intelligence to gain insights

Metadata

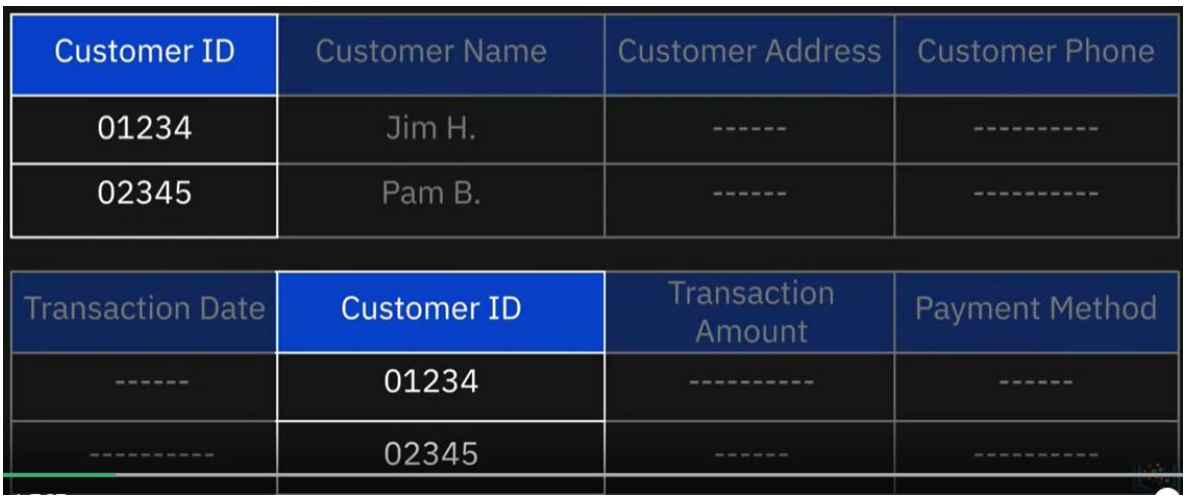
- Also needs managing
- Usually stored in a data catalog

A data catalog enhances data discovery, repeatability, governance, and access



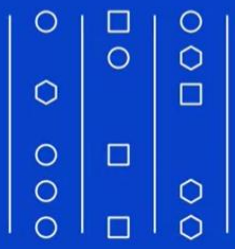
¿Qué es una base de relación?

Es una colección de datos organizados en estructura de tablas, donde cada tabla puede ser relacionada o vinculada, basados en datos que existan en común de cada una.



Incluso te permite entender las relaciones de todos los datos relacionados y así obtener nuevos insights para tomar mejores decisiones.

Examples of RDBMS



Relational Databases can be:

- Open-source with internal support
- Open-source with commercial support
- Commercial closed-source

IBM DB2 · Microsoft SQL Server · MySQL · ORACLE DATABASE · PostgreSQL

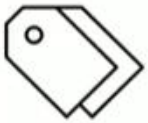
Cloud-Based Relational Databases, or Database-as-a-Service:

Amazon RDS · Google SQL · IBM DB2 on Cloud · ORACLE CLOUD · Azure SQL

- Ideal for the optimized storage, retrieval, and processing of data for large volumes of data
- Each table has a unique set of rows and columns
- Relationships can be defined between tables
- Fields can be restricted to specific data types and values
- Can retrieve millions of records in seconds using SQL for querying data
- Security architecture of relational databases provides greater access control and governance

¿En qué casos se usa el RDBMS?

Relational Databases are well suited for:



Online Transaction Processing (OLTP) application

Can support transaction-oriented tasks that run at high rates and

- Accommodate large number of users
- Manage small amounts of data
- Support frequent queries and fast response times

What is a NoSQL database?

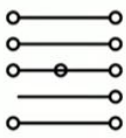
NoSQL (not only SQL) or Non SQL is a non-relational database design that provides flexible schemas for the storage and retrieval of data

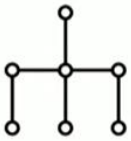
- Built for specific data models
- Has flexible schemas that allow programmers to create and manage modern applications
- Do not use a traditional row/column/table database design with fixed schemas
- Do not, typically, use the structured query language (or SQL) to query data

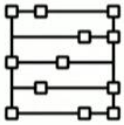
Key-Value Store

Not a great fit if you want to:

- Query data on specific data value
- Need relationships between data values
- Need multiple unique keys





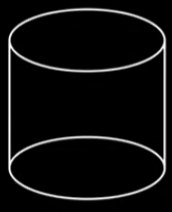


Extract, Transform, and Load Process

Extract

STAGING AREA

Transform and Load



Data Repository

For example,

- Standardizing date formats and units of measurement
- Removing duplicate data
- Filtering out data that is not required
- Enriching data
- Establishing key relationships across tables
- Applying business rules and data validations

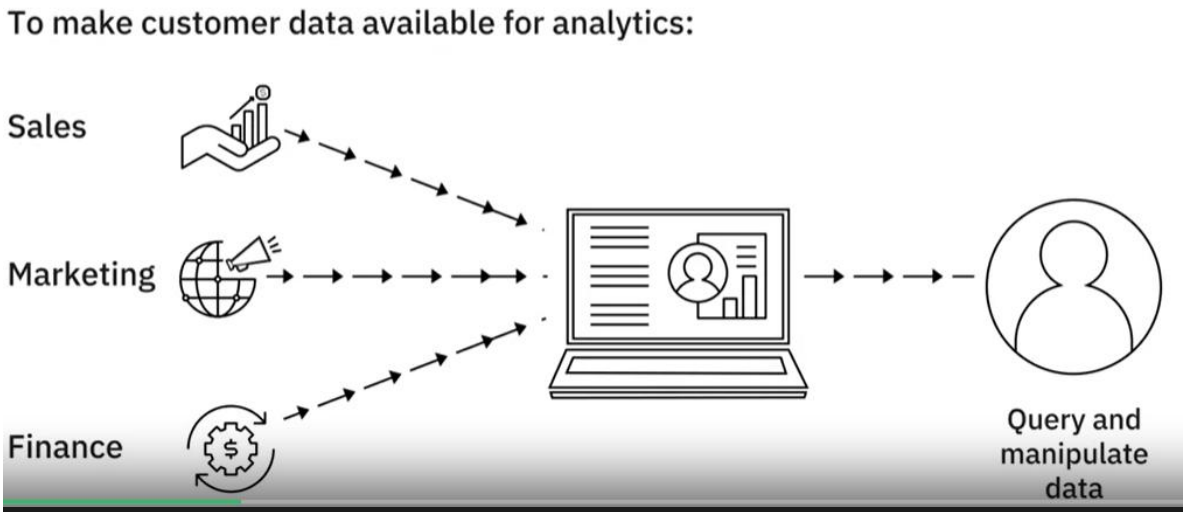
Data Integration includes:

Accessing, queueing, or extracting data from operational systems

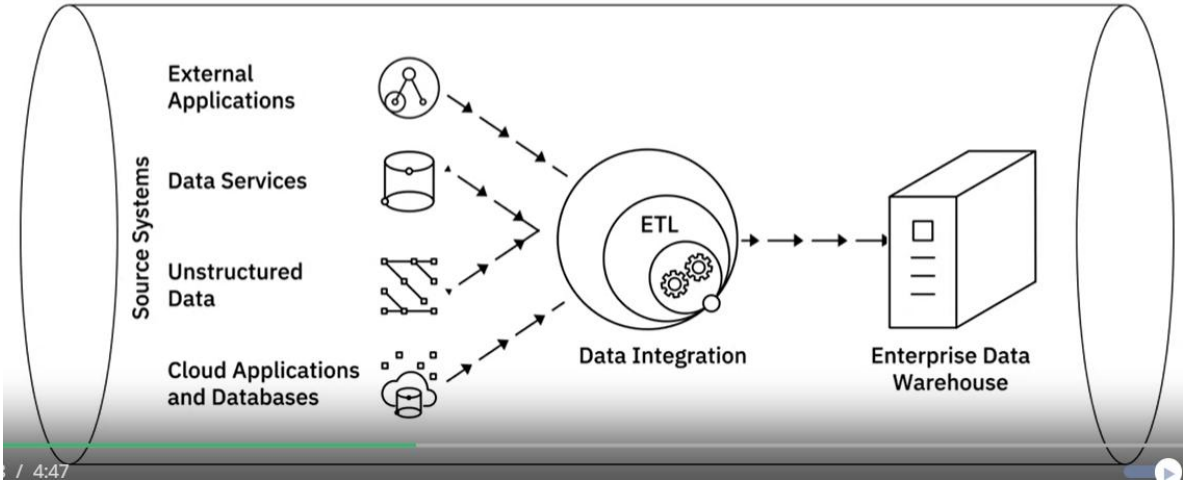
Transforming and merging extracted data either logically or physically

Data quality and governance

Delivering data through an integrated approach for analytics purposes



Data Integration Workflow



Data pipeline es el proceso completo desde la fuente hasta el destino.

Mientras ETL es el proceso dentro de la integración de datos.

El procesamiento de transacciones en línea (OLTP) es un tipo de procesamiento de datos que permite ejecutar un gran número de transacciones en tiempo real por parte de muchas personas. Algunos ejemplos de aplicaciones OLTP son:

- Cajeros automáticos
- Aplicaciones de banca en línea
- Procesamiento de pagos con tarjeta de crédito
- Entrada de pedidos
- Reservas en línea
- Mantenimiento de registros
- Comercio electrónico
- Industria
- Sistema de gestión de supermercados

Las aplicaciones OLTP permiten el procesamiento rápido y preciso de datos, y tienen grandes beneficios, como: Agilidad en el procesamiento, Consistencia y precisión, Soporte para decisiones en tiempo real, Alto nivel de concurrencia, Integración con aplicaciones empresariales.

Los sistemas de bases de datos OLTP están diseñados para generar copias de seguridad en segundo plano de forma periódica, lo que permite garantizar la integridad de los datos transaccionales sin afectar su capacidad operativa.

Los SGBD actúan como una interfaz entre los usuarios y las aplicaciones, y permiten almacenar diferentes datos de varias formas, como datos textuales, numéricos, binarios y datos y tiempo. Además, los SGBD permiten definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de estos.

Algunos ejemplos de SGBD son Oracle, DB2, PostgreSQL, MySQL y MS SQL Server.

OLTP (procesamiento de transacciones en línea) y SGBD (sistema de gestión de bases de datos) **son dos tipos de sistemas de procesamiento de datos diferentes**:

OLTP

Se utiliza para administrar y procesar transacciones en tiempo real, como compras, fabricación, nómina, contabilidad, banca en línea, entrada de pedidos y envío de mensajes de texto. Los sistemas OLTP están optimizados para el procesamiento transaccional y las actualizaciones en tiempo real, y tienen un gran número de usuarios que realizan transacciones breves. Los sistemas OLTP utilizan un esquema normalizado y relacional, y se encargan principalmente de ingresar, almacenar y recuperar los datos.

SGBD

Permite el almacenamiento, manipulación y consulta de datos pertenecientes a una base de datos organizada en uno o varios ficheros. En el modelo más extendido (base de datos relacional), la base de datos consiste en un conjunto de tablas entre las que se establecen relaciones.

Types of NoSQL databases

Document-based: Documents grouped into collections

Key-value: Each data entry has a key to access it

Columnar: Data stored in columns rather than rows

Graph: Data stored in nodes with relationships and properties

Big Data storage

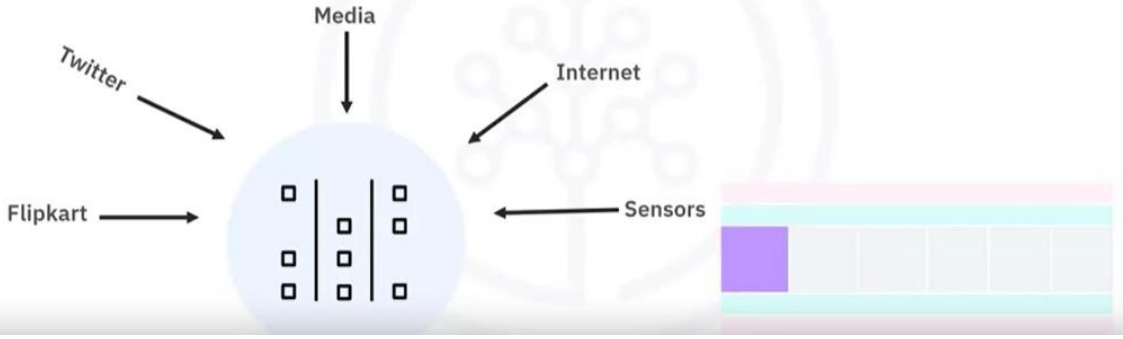
Data warehouse →	Multipurpose analysis ready for massive data stores
Data mart →	Sub-section of a data warehouse restricted access
Date lake →	Handles a combination of structured, semi-, and unstructured data

Data Science categories

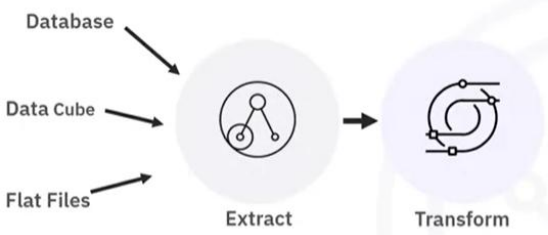
Data Management	Data Integration and Transformation	Data Visualization	Model Building	Model Deployment	Model Monitoring and Assessment

Data Management

- Collecting, persisting, and retrieving data securely, efficiently, and cost-effectively
- Data is collected from many sources



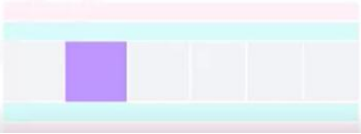
Data Integration and Transformation



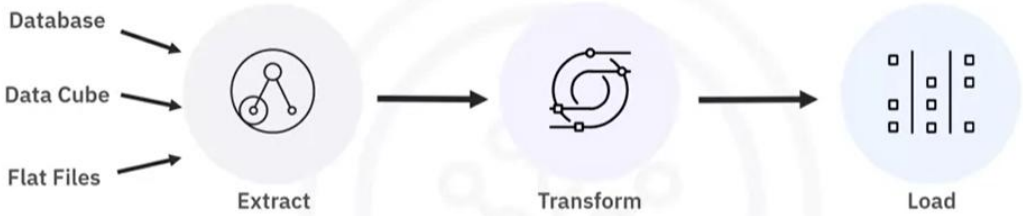
Index	Height	Weight
0	65.78	112.99
1	71.52	136.49
2	69.40	153.03
3	68.22	142.34
4	67.79	144.30
5	65.78	112.99

Index	Height	Weight
0	1.67	51.25
1	1.82	61.91
2	1.76	69.41
3	1.73	64.56
4	1.72	65.45
5	1.67	51.25

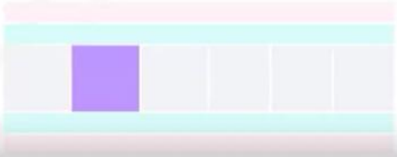
Data transformation is the process of transforming the values, structure and format of data.



Data Integration and Transformation



Transformed data is loaded back to the data warehouse.



Data Visualization

- Graphical representation of data and information
- In the form of charts, plots, maps, and animations
- Data visualization conveys data more effectively

Model Building

- Model building is a step where you train the data and analyze patterns using suitable machine learning algorithms
- Create machine learning models using IBM Watson machine learning

Data Asset Management

- Platform for organizing and managing the data
- Replication, backup, and access right management

Execution Environment

- An **execution environment** has libraries for code compiling and system resources to execute and verify code
- Cloud-based execution environments aren't tied to specific hardware or software
- IBM Watson Studio has tools for data preprocessing model training, and deployment

La compilación de datos es el proceso de recolectar, organizar y reunir información de diferentes fuentes para crear un conjunto coherente y utilizable de datos. Este proceso puede involucrar la recolección de datos de encuestas, registros, bases de datos, o cualquier otra fuente de información, y luego organizar esos datos de manera que sean fáciles de analizar o interpretar.

La compilación de datos es crucial en campos como la investigación científica, la estadística, y el análisis de mercado, ya que permite a los investigadores o analistas obtener una visión clara y completa del tema o problema que están estudiando.

Data integration and transformation

- Data integration and transformation in the classic data warehousing world is for ETL or ELT
- Also termed Data Refinery and Cleansing

Data visualization

Open-source tools:

- Supported by programming libraries where you need to use code
- Containing a user interface

Recap

In this video, you learned that:

- Data management tools are MySQL, PostgreSQL, MongoDB, Apache CouchDB, Apache Cassandra, Hadoop File System, Ceph, and elastic search
- Data integration and transformation tools are Apache AirFlow, KubeFlow, Apache Kafka, Apache Nifi, Apache SparkSQL, and NodeRED
- Data Visualization tools are Pixie Dust, Hue, Kibana, and Apache Superset
- Model deployment tools are Apache PredictionIO, Seldon, Kubernetes, Redhat OpenShift, Mleap, TensorFlow service, TensorFlow lite, and TensorFlow dot JS
- Model monitoring tools are ModelDB, Prometheus, IBM AI Fairness 360, IBM Adversarial Robustness 360 Toolbox, and IBM AI Explainability 360
- Code asset management tools are Git, GitHub, GitLab, and Bitbucket
- Data asset management tools are Apache Atlas, ODPi Egeria, and Kylo

- **IBM Watson Studio:** Diseñado como un entorno integrado, Watson Studio simplifica el desarrollo, la formación y el despliegue de modelos. Ofrece compatibilidad con varios lenguajes y marcos de trabajo, como Python, R y TensorFlow, además de funciones de colaboración, herramientas de preparación de datos y opciones de despliegue versátiles.

- IBM AutoAI**:** IBM AutoAI, una función notable integrada en Watson Studio, agiliza el proceso de construcción de modelos de aprendizaje automático. Mediante la exploración dinámica de varios algoritmos e hiperparámetros, pretende identificar el modelo óptimo para un conjunto de datos determinado.

- **IBM Watson OpenScale:** Como plataforma para supervisar y gestionar modelos de IA en producción, Watson OpenScale desempeña un papel fundamental a la hora de garantizar la equidad, la explicabilidad y la mitigación de sesgos de los modelos. Proporciona información sobre el rendimiento del modelo y su evolución a lo largo del tiempo, lo que facilita la toma de decisiones informadas.

- **IBM Watson Aprendizaje automático:** Watson Aprendizaje automático, disponible como servicio en la plataforma IBM Cloud, permite a los usuarios escalar su formación y despliegue de modelos de aprendizaje automático. Da soporte sin problemas a marcos populares como TensorFlow, PyTorch y Scikit-learn, y ofrece API para una integración perfecta con otras aplicaciones.

¿Qué herramienta comercial puede utilizarse para definir y ejecutar procesos de integración de datos al estilo de una hoja de cálculo?

- ○ Watson Studio Desktop
- ○ RStudio
- ○ Modelador SPSS
- ◉ Base de datos Oracle

Incorrecto
Incorrecto. Esta herramienta se utiliza para la gestión de datos.

¿Qué servicio ofrece un formato de estructura de datos de documentos?

- ◉ Refinería de datos
- ○ Db2
- ○ JSON
- ○ DataMeer

Incorrecto
Incorrecto. Data Refinery permite transformar grandes cantidades de datos en bruto en información consumible y de calidad en una interfaz de usuario similar a una hoja de cálculo.

- SQL's scope is limited to querying and managing data
- SQL was designed for managing data in relational databases
- SQL behaves like an interpreter between you and the database

Deep Learning Libraries in Python

3. Deep Learning Libraries

TensorFlow

TensorFlow
(Deep Learning: Production and Deployment)

PyTorch

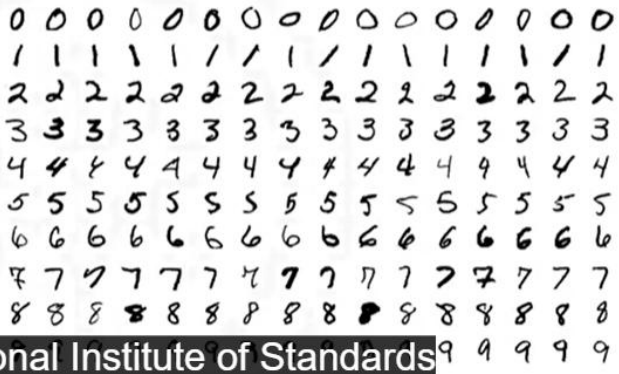
PyTorch
(Deep Learning: regression, classification)

What's a data set?

- Collection of data
- Data structures
  - Tabular data
  - Hierarchical data, network data
  - Raw files

Se utiliza para representar relaciones entre los datos

Trabaja con imágenes o audio.

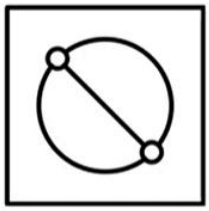


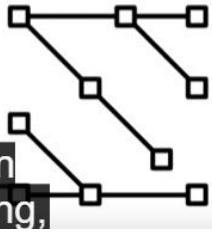
The Modified National Institute of Standards and Technology (MNIST) dataset is popular

Data ownership

- Private data
  - Confidential
  - Private or personal information
  - Commercially sensitive
- Open data
  - Publicly available
  - Companies
  - Scientific institutions
  - Government
  - Organizations
  - Companies

Data Science, Machine Learning, Artificial Intelligence





Open data has played a significant role in the growth of data science, machine learning,

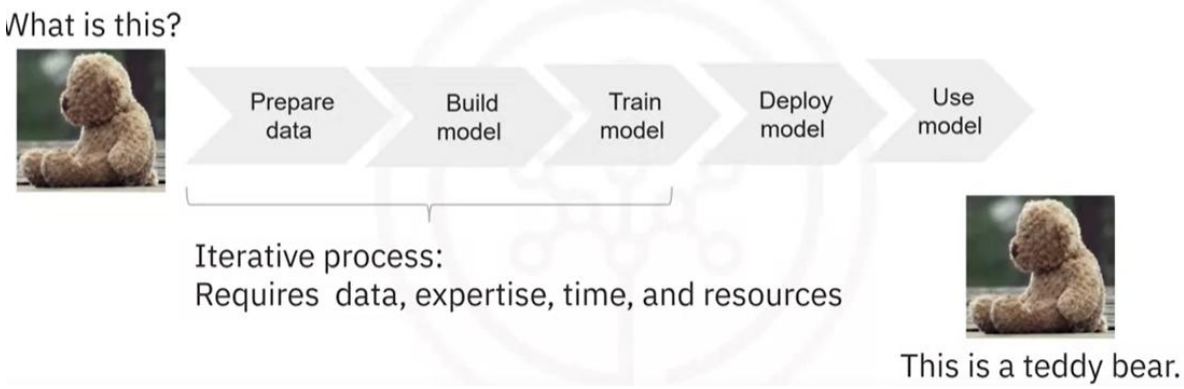
Where to find open data

- Open data portal list from around the world
  - http://datacatalogs.org/
- Governmental, intergovernmental and organization websites
  - http://data.un.org/ (United Nations)
  - https://www.data.gov/ (USA)
  - https://www.europeandataportal.eu/en/ (Europe)
- Kaggle
  - https://www.kaggle.com/data sets
- Google data set search
  - https://data setsearch.research.google.com/

Deep Learning

- Tries to loosely emulate show the human brain works
- Applications
  - Natural Language Processing
  - Image, audio, and video analysis
  - Time series forecasting
- Requires large datasets of labeled data and is compute intensive
- Requires special purpose hardware

Using models to solve a problem



- Python ofrece un variado ecosistema de librerías para la Ciencia de datos, que abarca la computación científica (Pandas, NumPy), la visualización (Matplotlib, Seaborn) y el Aprendizaje automático de alto nivel (Scikit-learn). Estas bibliotecas ofrecen herramientas para la manipulación de datos, operaciones matemáticas y desarrollo simplificado de modelos de aprendizaje automático.

- Las interfaces de programación de aplicaciones (API) facilitan la comunicación entre componentes de software. Las API REST, en concreto, facilitan la comunicación por Internet y el acceso a recursos como el almacenamiento. Los términos clave de las API son cliente (usuario o código que accede a ella), recurso (servicio o datos) y punto final (URL de la API).
- El Acuerdo de Licencia de Datos Comunitarios (CDLA) facilita la puesta en común de datos abiertos proporcionando condiciones de licencia claras para su distribución y uso, y el sitio **IBM Data Asset eXchange (DAX)** contiene conjuntos de datos abiertos de alta calidad.

4. ¿Qué ofrece el Data Asset eXchange (DAX)?

- ○ Acceso exclusivo a conjuntos de datos propiedad de IBM
- ○ Conjuntos de datos cerrados de alta calidad
- ◉ Una colección de conjuntos de datos abiertos
- ○ Conjuntos de datos de uso exclusivamente académico

Correcto
Correcto. Data Asset eXchange (DAX) proporciona una colección seleccionada de conjuntos de datos abiertos. Estos conjuntos de datos proceden tanto de IBM Research como de fuentes externas de confianza. Están disponibles bajo el Acuerdo de Licencia de Datos de la Comunidad (CDLA), fomentando el intercambio de datos y la colaboración.

6. ¿Cómo se distribuyen los microservicios del modelo MAX?

- ○ Como archivos ejecutables independientes
- ○ Como API en la nube
- ◉ Como imágenes Docker de código abierto
- ○ Como paquetes Python

Correcto
Correcto. Los microservicios que sirven al modelo MAX se construyen y distribuyen como imágenes Docker de código abierto.

https://www.coursera.org/learn/open-source-tools-for-data-science/ungradedWidget/FHXOD/additional-sources-of-datasets

2. ¿Cuál es la finalidad del núcleo en el cuaderno Jupyter?

- ○ Traduce el código a otro idioma
- ◉ Ejecuta el código
- ○ Exporta el código a un archivo diferente
- ○ Escribe el código

Correcto
Correcto Un núcleo de cuaderno es un motor computacional que ejecuta el código contenido en un archivo de cuaderno.

https://try.github.io/

Learn by reading

Git Handbook

Git, GitHub, DVCS, oh my! Learn all the lingo and the basics of Git.

Cheat Sheets

Keep these handy! Reference sheets covering Git commands, features, SVN migrations, and bash. Available in a multiple languages.

Learn by doing

Learn Git branching

Try Git commands right from your web browser. Featuring some of your soon-to-be favorites: branch, add, commit, merge, revert, cherry-pick, rebase!

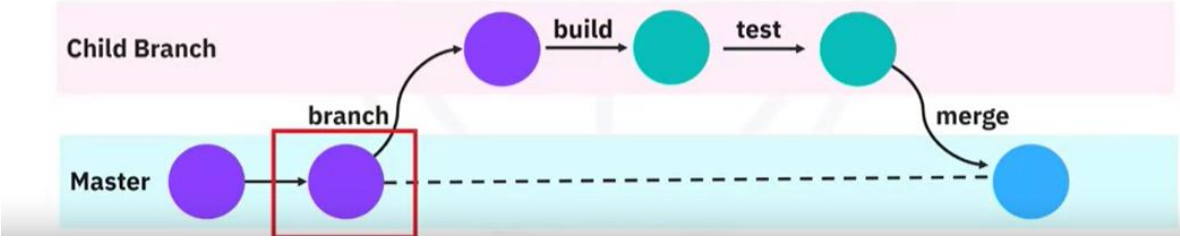
Visualizing Git

Look under the hood! Explore how Git commands affect the structure of a repository within your web browser with a free explore mode, and some constructed scenarios.

kily for us, GitHub has amazing resources available to help you get sta

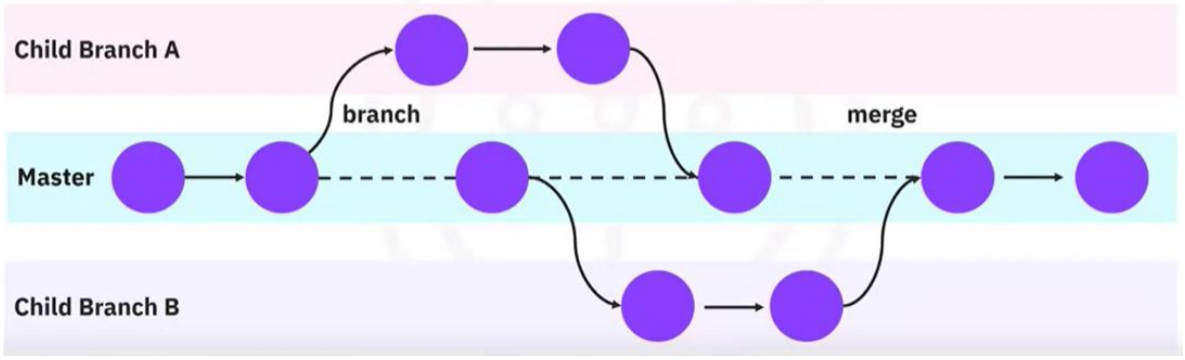
Why create a branch?

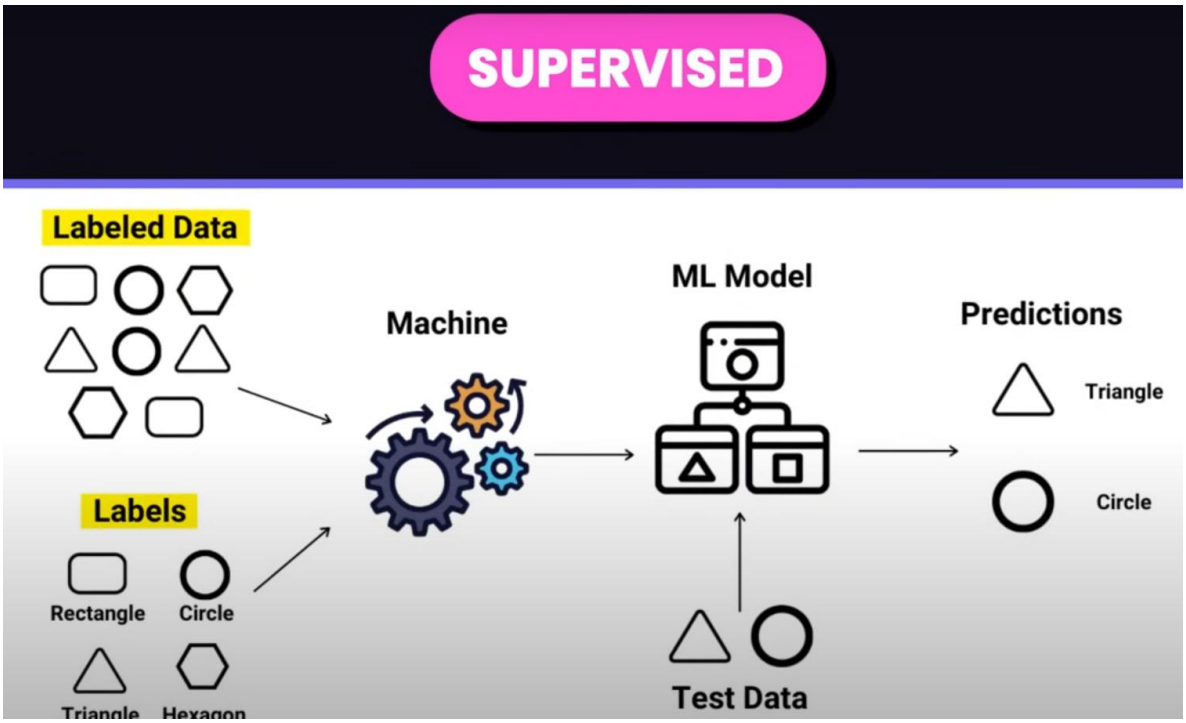
- Edits and changes are made in the child branch
- Tests are done to ensure quality before merging with the Master branch

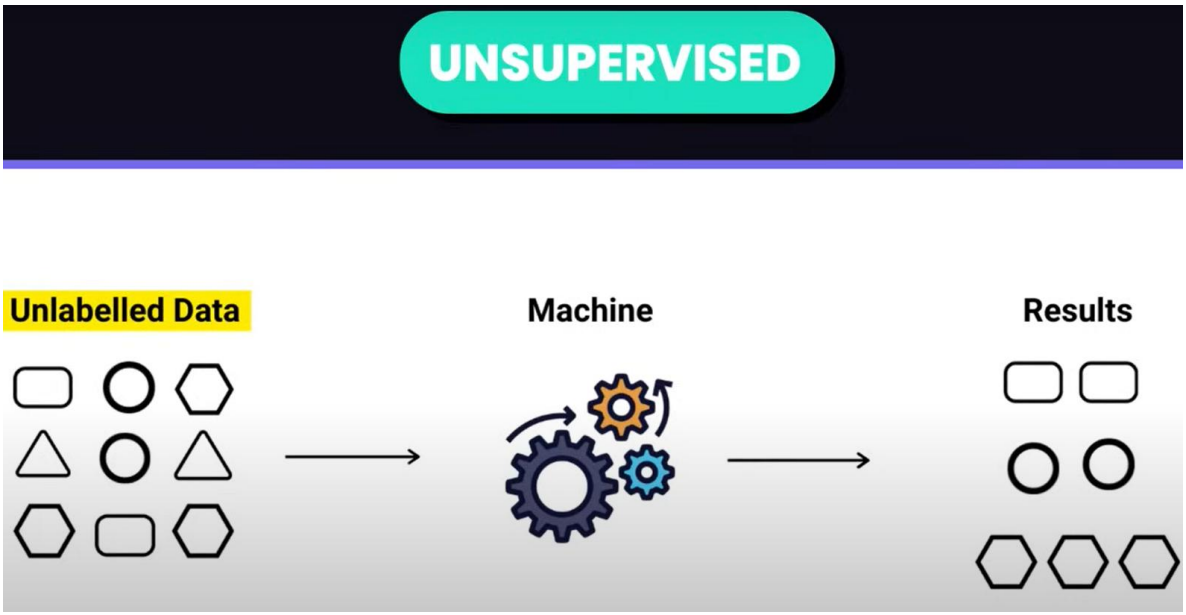


Merging multiple branches

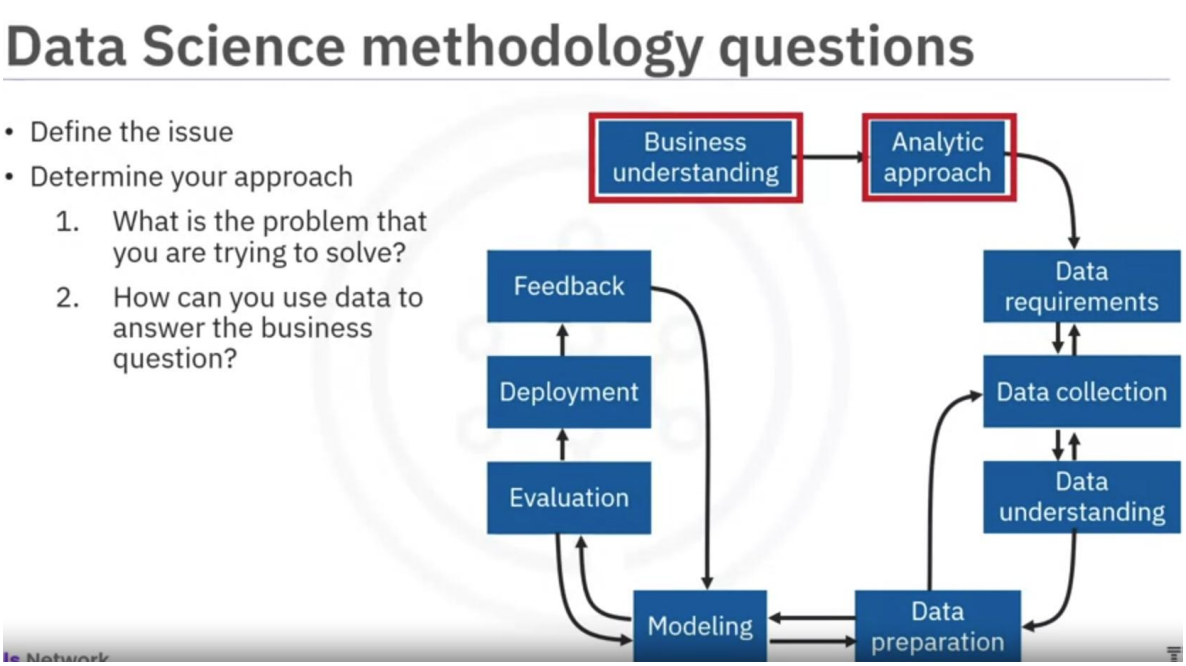
- Branches allow for simultaneous development and testing by multiple team members

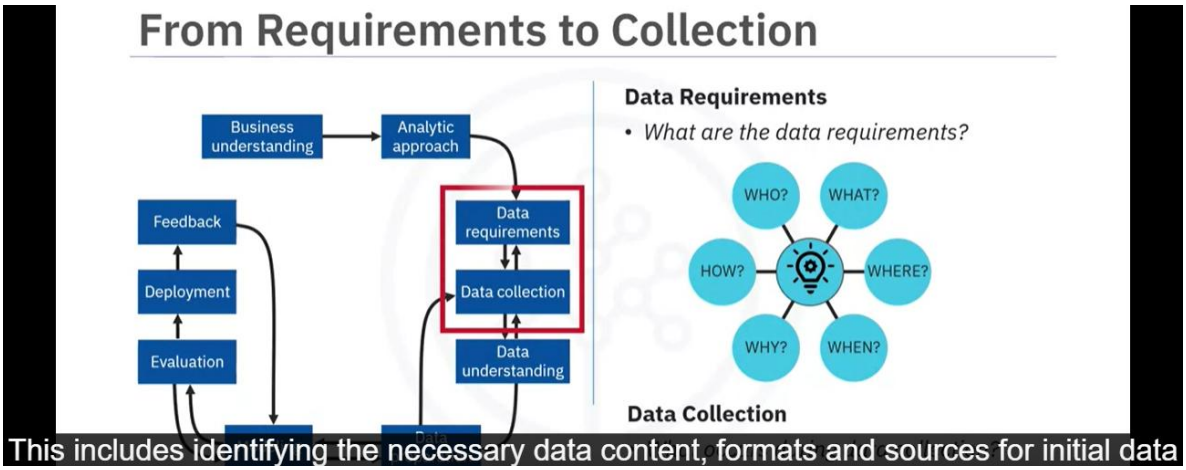






No hay etiquetas, a lo que el programa trata de entender el contexto para así arrojar un resultado.

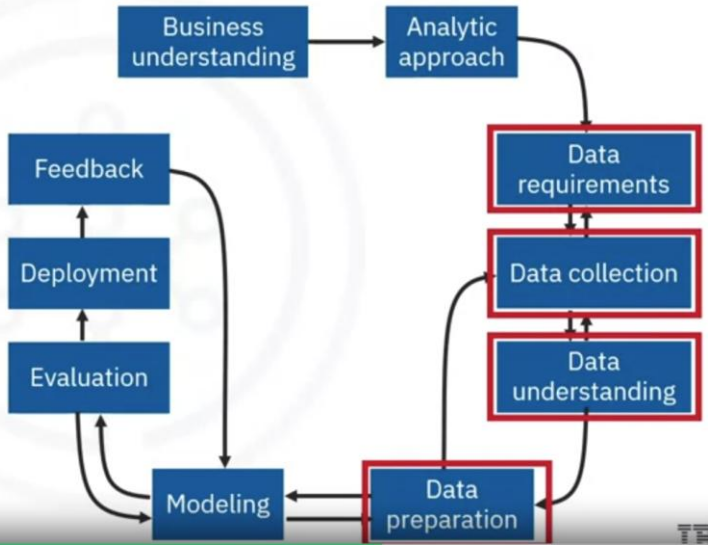




Data Science methodology questions

Next, get organized around the data

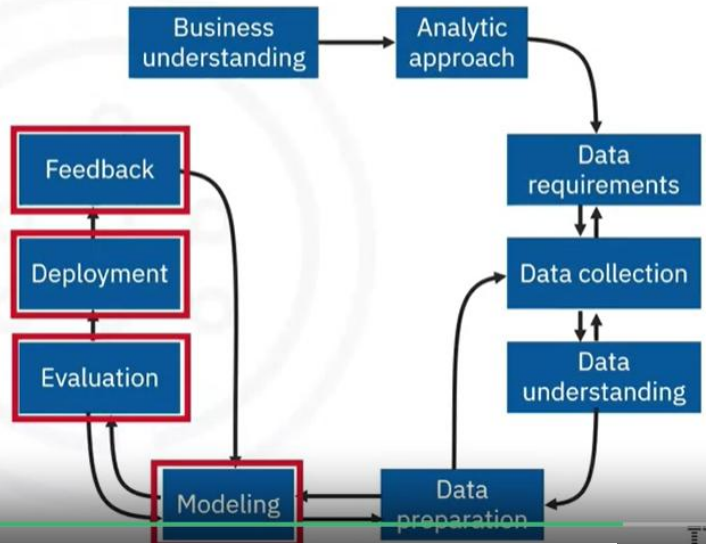
3. What data do you need to answer the question?
4. Where is the data sourced from, and how will you receive the data?
5. Does the data you collected represent the problem to be solved?
6. What additional work is required to manipulate and work with the data?



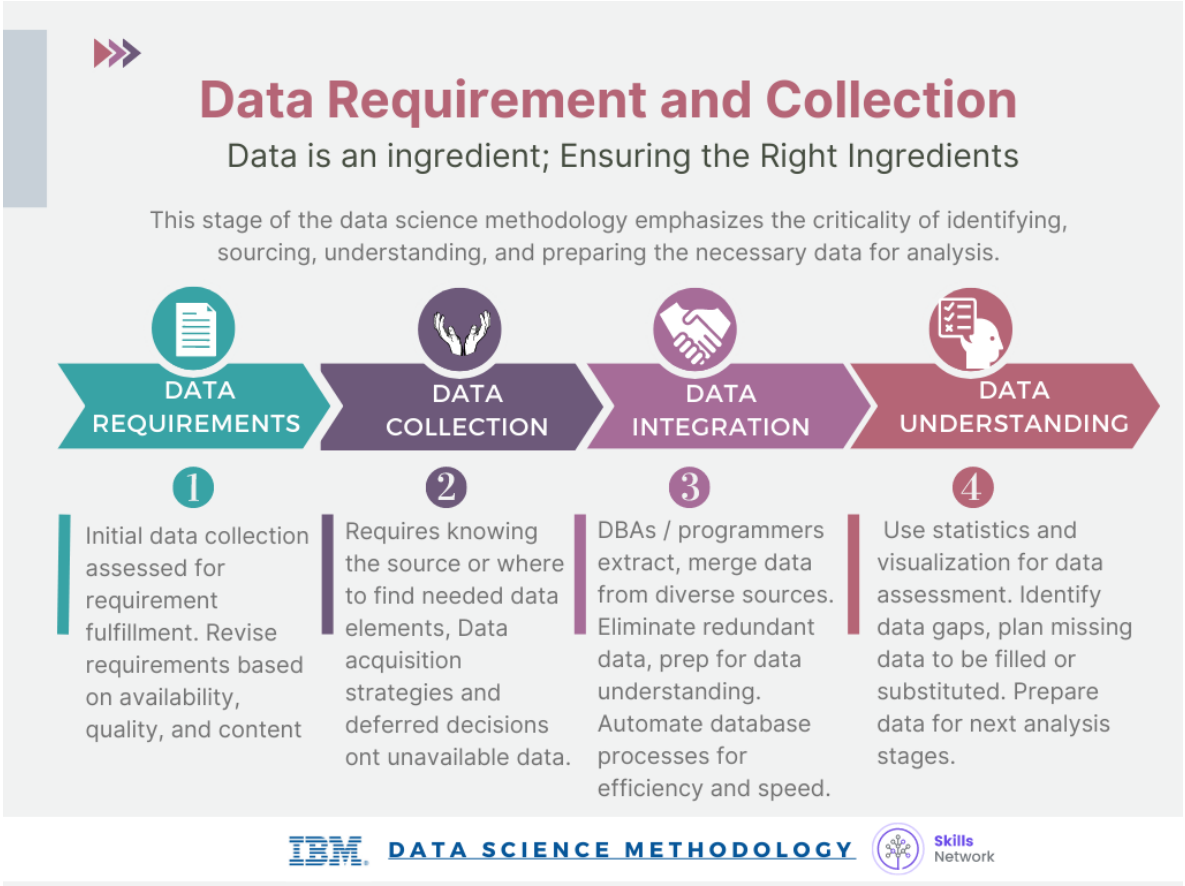
Data Science methodology questions

Validate your approach and the final design of the data

7. When you apply data visualizations, do you see answers that address the business problem?
8. Does the data model answer the initial business question, or must you adjust the data?
9. Can you put the model into practice?
10. Can you get constructive feedback from the data and the stakeholder to answer the business question?



1. Predict CHF readmission outcome (Y or N) for each patient
2. Predict the readmission risk for each patient
3. Understand explicitly what combination of events led to the predicted outcome for each patient
4. Easy to understand and apply to new patients to predict their readmission risk

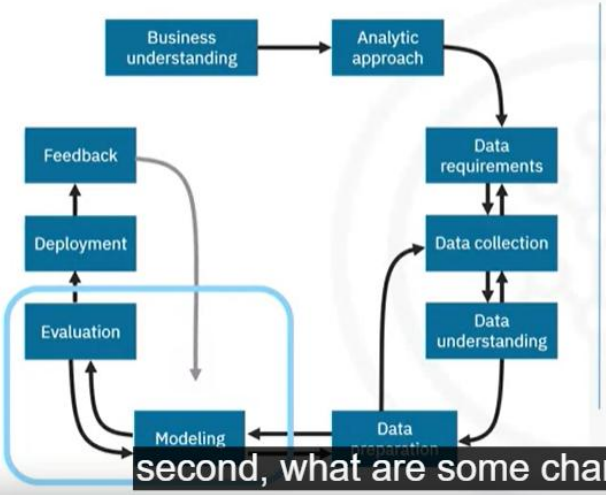


Examples of data cleansing

	A	B	C	D	E
1	**Name**	**Date**	**Age**	**Location**	**Country**
2	John Doe	2012 02 20	32	ON	CAN
3	May Lag	2013 02 33	2	ON	CA
4	Henry Oon	30-Sep-12	35	Ontario	CANADA
5	Kelly, Tom	2015 02 20	65	ON	CA
6	John Kell	2016 02 20		AB	CA
7	Henry Oon	30-Sep-12	35	Ontario	CANADA
8					

Invalid Values
Missing Data
Remove Duplicates
Formatting

From modeling to evaluation



Modeling

- *In what way can the data be visualized to get to the answer that is required?*

Evaluation

- *Does the model used really answer the initial question or does it need to be adjusted?*

second, what are some characteristics of this process?

Modeling to Evaluation

Unveil the Modeling Process and Assess Performance

Modeling and evaluation play a pivotal role in shaping analytical outcomes and refining problem-solving strategies

DATA MODELING	TRAINING SET AND CALIBRATIN	MODEL EVALUATION	DIAGNOSTIC MEASURES
1	2	3	4
Aims to determine the characteristics and purpose of the modeling process. Models can be descriptive/predictive. They are based on statistical or machine learning approaches.	Training set serves as a gauge to calibrate the model. Experiment with algorithms for variable selection is done. Success depends on problem understanding and analytical approach.	Model evaluation is iterative, done alongside model building. It occurs before deployment, assessing model quality and alignment with the initial request.	Diagnostic measures ensure intended model functionality. Significance testing interprets data. ROC curve aids optimal classification selection by quantifying performance.

Case study: Using the ROC curve

Diagnostic tool for classification model evaluation

- Classification model performance
- True-Positive Rate vs False-Positive Rate
- Optimal model at maximum separation

From Deployment to Feedback

Real-world Deployment, feedback and Redeployment

Maximizing Data Science Impact through Stakeholder Engagement and Iterative Refinement

1 STAKEHOLDER ENGAGEMENT	2 DEPLOYMENT AND FEEDBACK	3 ITERATIVE PROCESS	4 IMPROVEMENT AND REDEPLOYMENT
Making the answer relevant and useful involve engaging stakeholders. Stakeholders include solution owners, marketing, IT admins, developers. Diverse specialties ensure model's applicability.	Deploy the evaluated model with data scientist's confidence for real-time functionality testing. User feedback is critical for refining, evaluating, and enhancing model performance.	The cyclical Methodology refines each stage. Feedback spurs continuous learning and enhancement. Refinement relies on post-implementation data and knowledge. Further improvements can arise from feedback.	Integrate feedback insights to refine model and interventions. Redeploy refined model and actions, sustaining feedback. Loop ensures ongoing improvement and impact assessment, especially. Include ethical considerations.

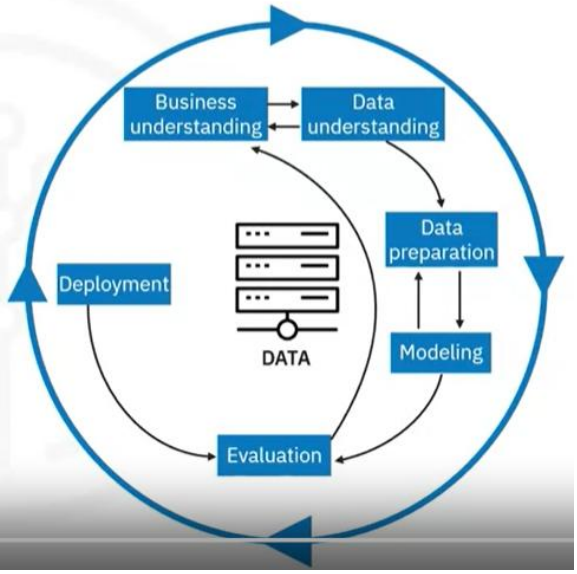
What is CRISP-DM?

- An acronym for Cross-Industry Standard Process for Data Mining
- A structured approach to guide data-driven decision-making

CRISP-DM as a data methodology

The CRISP-DM model includes:

- Data mining stages
- Data mining stage descriptions
- Explanations of the relationships between tasks and stages



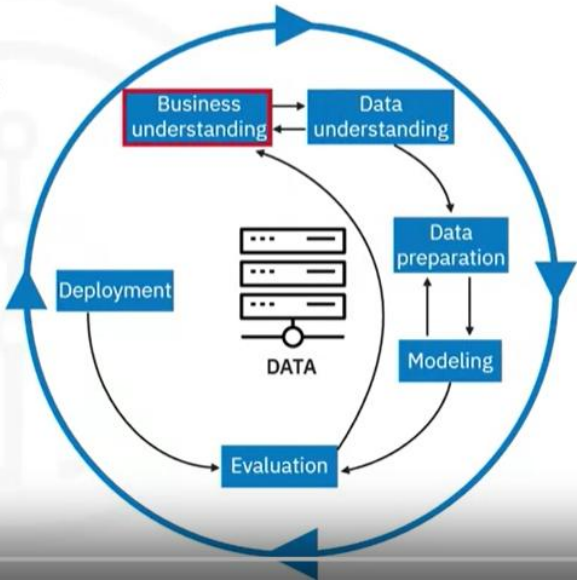
CRISP-DM: A high-level process model

Provides high-level insights into the data mining life cycle

The Business Understanding stage

The Business Understanding stage:

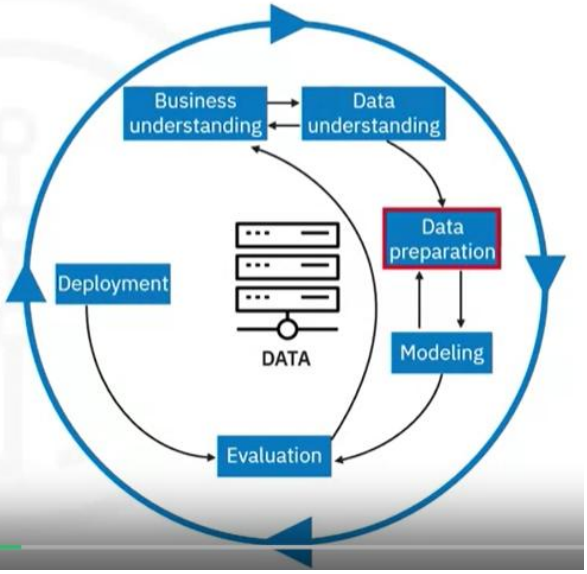
- Sets and outlines the project's data analysis intentions and goals
- Requires communication and clarity to overcome stakeholders' differing objectives, biases, and information modalities
- Is necessary to avoid wasted time and resources



The Data Preparation stage

Data scientists perform the following tasks:

- Transform data
- Determine if more data is needed
- Address questionable missing and ambiguous data values



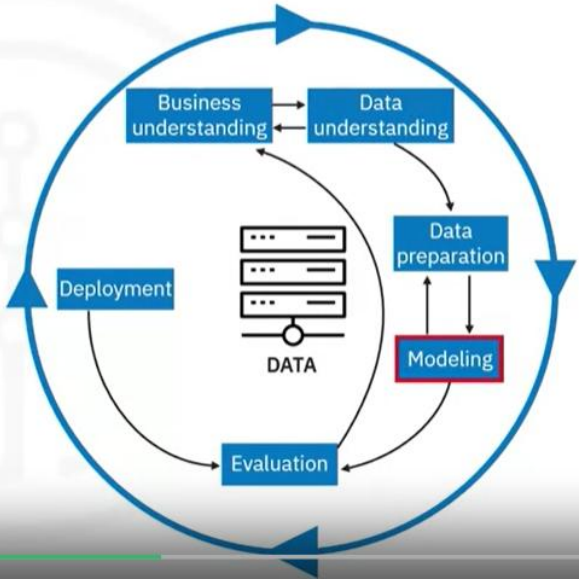
The Modeling stage

Data mining:

- Reveals patterns and structure within the data
- Provides knowledge and insights that address the stated business problem and goals

Data scientists perform the following tasks:

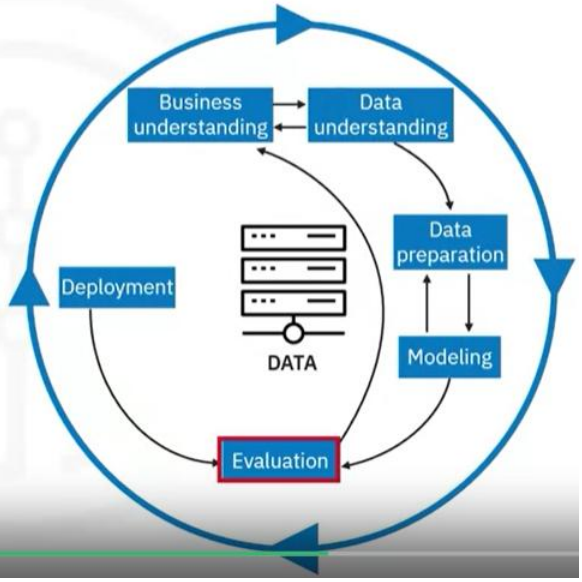
- Select data models
- Adjust the models



The Evaluation stage

Data scientists perform the following tasks:

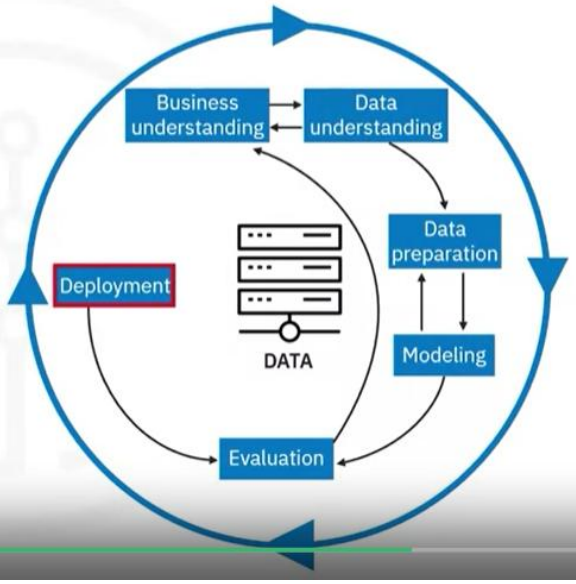
- Test the selected module
- Assess the model's effectiveness
- Results determine the model's efficacy



The Deployment stage

Data scientists and stakeholders perform the following tasks:

- Use the data model on new data outside of the data set
- Analyze the results to determine the need for new variables, a new data set, or a new model



Python is great for a huge range of tasks:


Data Analysis


Web Scraping

Big Data

Finance


Computer Vision


Natural Language

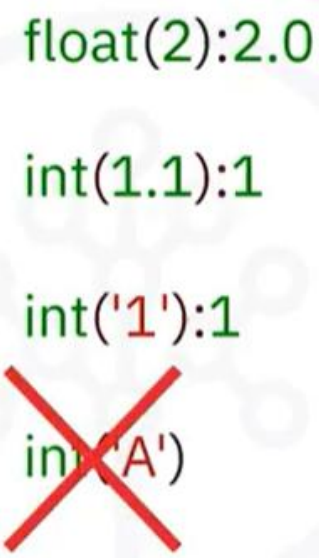

Machine Learning


Deep Learning

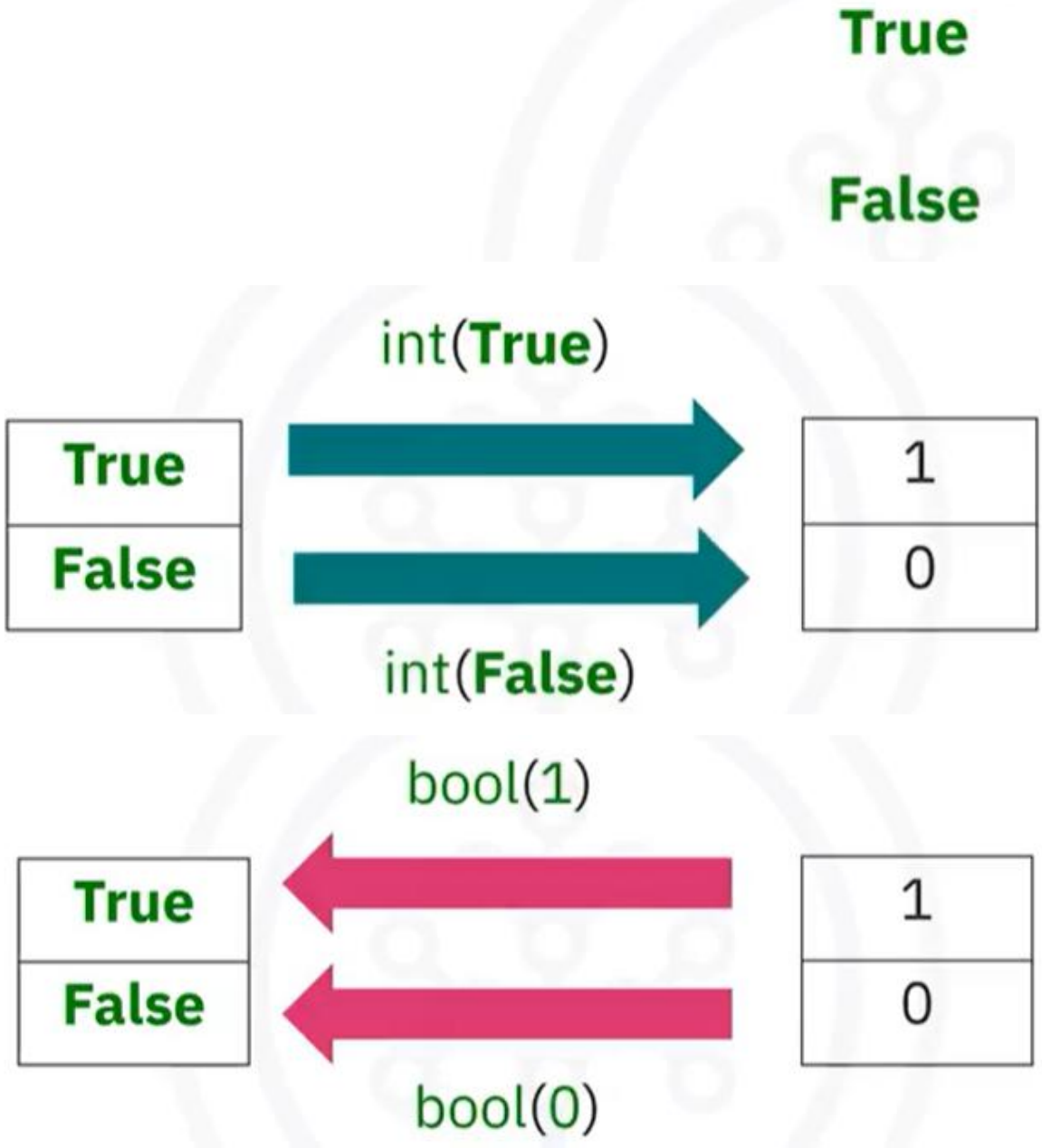
The type command

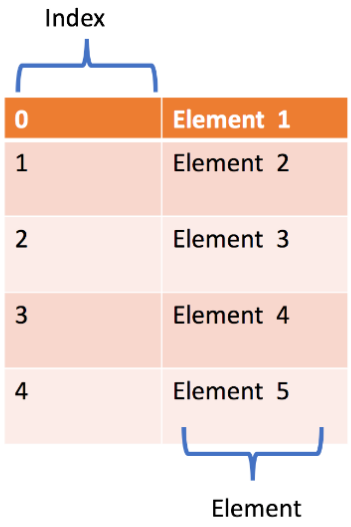
type(11)	int
type(21.213)	float
type("Hello Python 101")	str

Changing expression type



Data types: Boolean





[Element 1 , Element 2 , Element 3 , Element 4, Element 5]

Index 0 1 2 3 4

```
L =["Michael Jackson", 10.1,1982]
```

-3	0	"Michael Jackson"
-2	1	10.1
-1	2	1982

L[-3]: "Michael Jackson"

L[-2]: 10.1

L[-1]: 1982

```
L =["Michael Jackson", 10.1,1982, "MJ", 1]
```

0	1	2	3	4

```
# Use extend to add elements to list

L = [ "Michael Jackson", 10.2]
L.extend(['pop', 10])
L
```

```
['Michael Jackson', 10.2, 'pop', 10]
```

Another similar method is `append`. If we apply `append` instead of `extend`, we add one element to the list:

```
# Use append to add elements to list

L = [ "Michael Jackson", 10.2]
L.append(['pop', 10])
L
```

```
['Michael Jackson', 10.2, ['pop', 10]]
```

If we append the list `['a','b']` we have one new element consisting of a nested list:

```
# Use append to add elements to list

L.append(['a','b'])
L
```

```
['Michael Jackson',
 10.2,
 'pop',
 10,
 ['a', 'b'],
 ['a', 'b'],
 ['a', 'b'],
 ['a', 'b'],
 ['a', 'b'],
 ['a', 'b']]
```

We can also delete an element of a list using the `del` command:

```
# Delete the element based on the index

print('Before change:', A)
del(A[0])
print('After change:', A)
```

```
Before change: ['hard rock', 10, 1.2]
After change: [10, 1.2]
```

```
# Split the string, default is by space

'hard rock'.split()
```

```
['hard', 'rock']
```

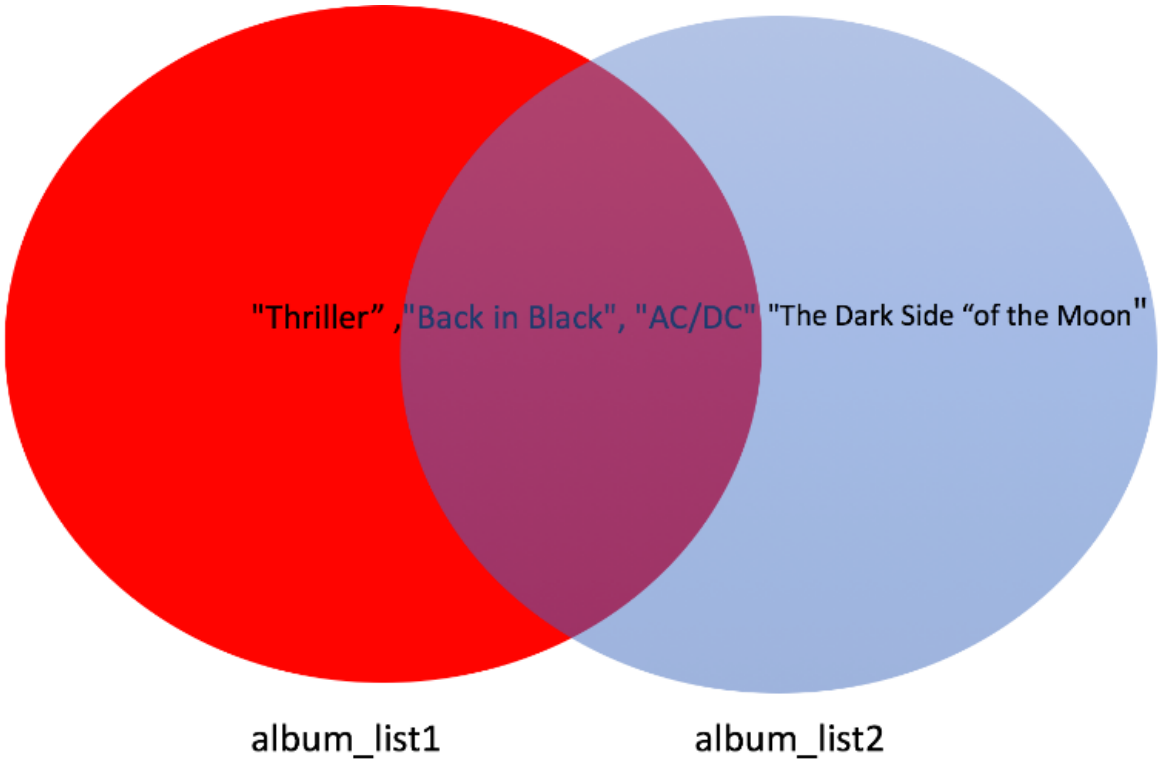
Split the string by comma

'A,B,C,D'.split(',')

['A', 'B', 'C', 'D']

```
album_set2.difference(album_set1)
```

```
{'The Dark Side of the Moon'}
```



You can find the intersect of two sets as follow using `&`:

```
# Find the intersections

intersection = album_set1 & album_set2
intersection
```

```
{'AC/DC', 'Back in Black'}
```

```
# Remove the element from set

A.remove("NSYNC")
A
```

```
{'AC/DC', 'Back in Black', 'Thriller'}
```

We can verify if an element is in the set using the `in` command:

```
# Verify if the element is in the set

"AC/DC" in A
```

```
True
```

If we add the same element twice, nothing will happen as there can be no duplicates in a set:

```
# Try to add duplicate element to the set

A.add("NSYNC")
A
```

```
{'AC/DC', 'Back in Black', 'NSYNC', 'Thriller'}
```

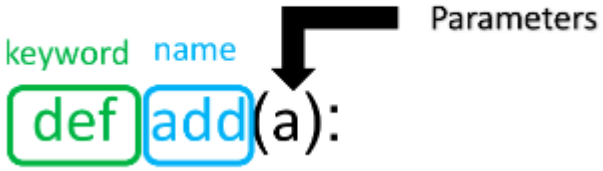
When there are multiple letters, the first letter takes precedence in ordering:

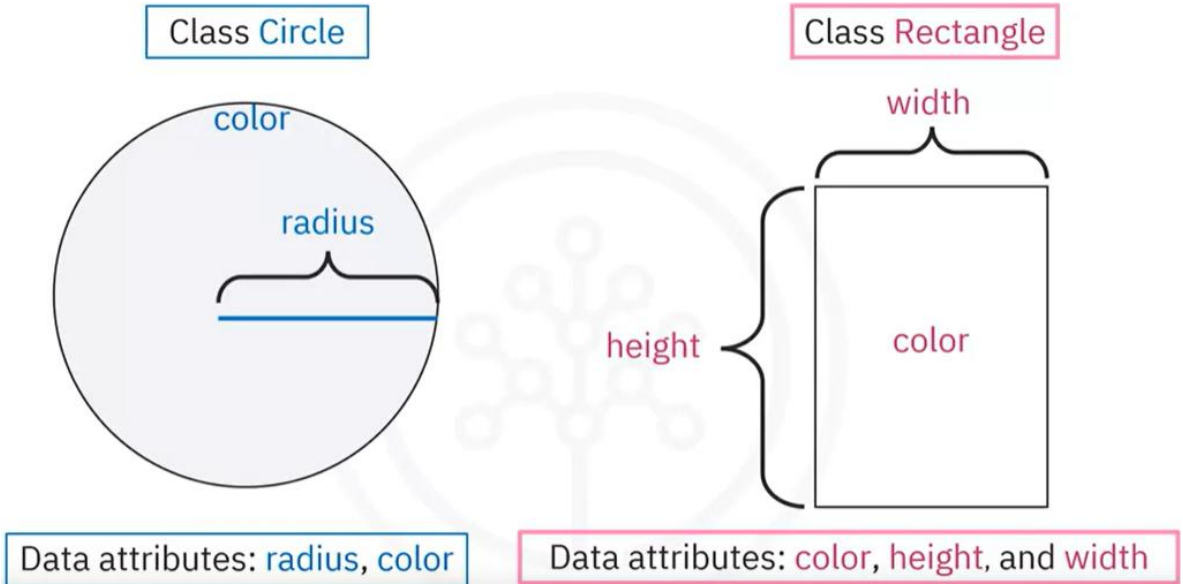
```
# Compare characters

'BA' > 'AB'
```

```
True
```

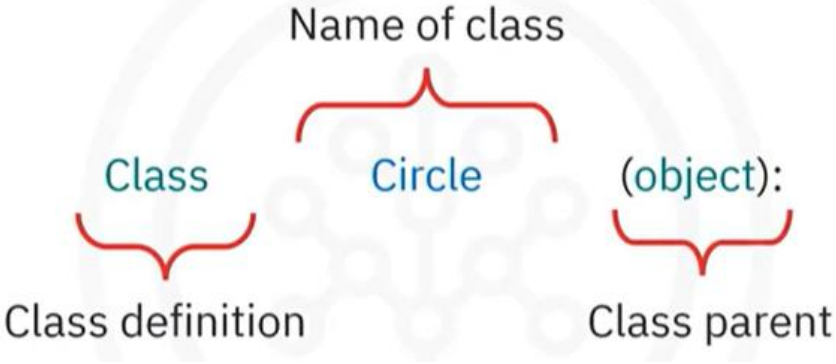
- equal: **==**
- not equal: **!=**
- greater than: **>**
- less than: **<**
- greater than or equal to: **>=**
- less than or equal to: **<=**

The elif statement, short for else if, allows us to check additional conditions if the condition statements before it are **False**. If the condition for the elif statement is **True**, the alternate expressions will be run.

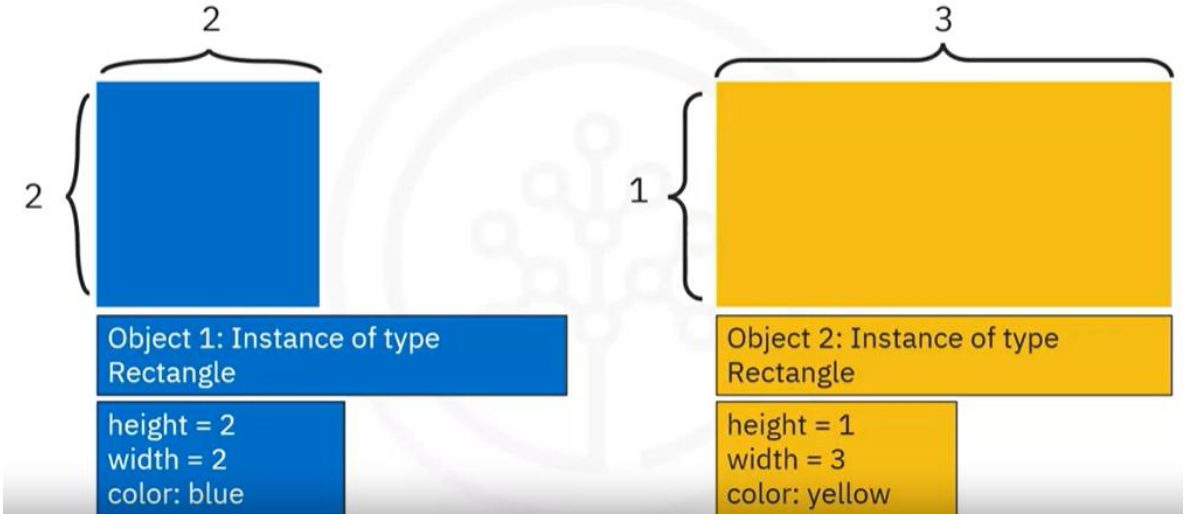




Create a class: Circle



Attributes and objects




```
class Circle(object):
```

Define your class

```
def __init__(self, radius , color):
    self.radius = radius
    self.color = color
```

Data attributes used to initialize each instance of the class

Special method or constructor used to initialize some data attributes

Parameters

```
def __init__(self, radius , color):
```

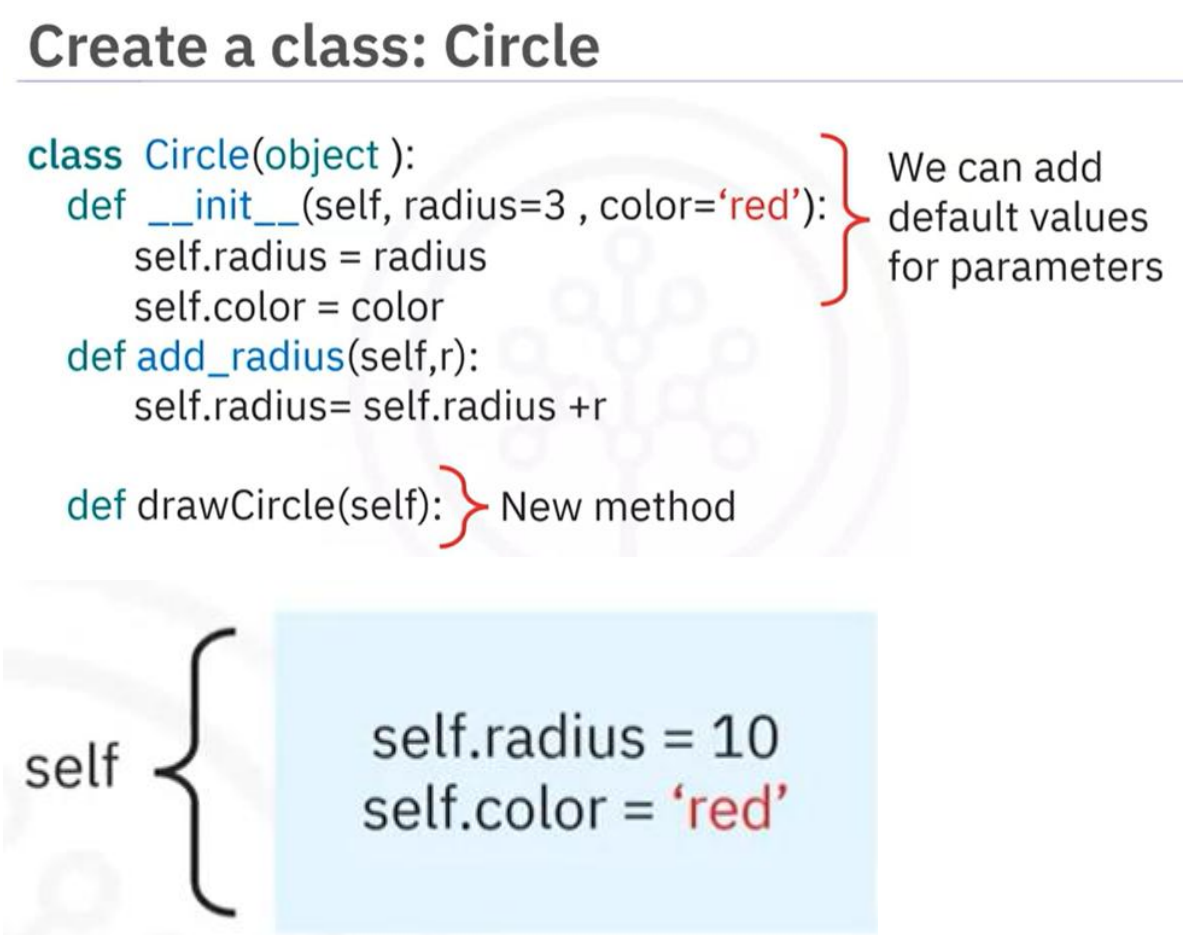
Self parameter

```
self.radius = radius
self.color = color
```

Class declaration (class ClassName)

- The `class` keyword is used to declare a class in Python.
- `ClassName` is the name of the class, typically following CamelCase naming conventions.

```
1  class ClassName:
```



```python
# Create a class Circle

class Circle(object):

    # Constructor
    def __init__(self, radius=3, color='blue'):
        self.radius = radius
        self.color = color

    # Method
    def add_radius(self, r):
        self.radius = self.radius + r
        return(self.radius)

    # Method
    def drawCircle(self):
        plt.gca().add_patch(plt.Circle((0, 0), radius=self.radius, fc=self.color))
        plt.axis('scaled')
        plt.show()
```

```python
with open("Example1.txt","r") as file1:

    file_stuff=file1.read()

    print(file_stuff)

print(file1.closed)
print(file_stuff)
```

Mode	Syntax	Description
‘r’	`'r'`	Read mode. Opens an existing file for reading. Raises an error if the file doesn't exist.
‘w’	`'w'`	Write mode. Creates a new file for writing. Overwrites the file if it already exists.
‘a’	`'a'`	Append mode. Opens a file for appending data. Creates the file if it doesn't exist.
‘x’	`'x'`	Exclusive creation mode. Creates a new file for writing but raises an error if the file already exists.
‘rb’	`'rb'`	Read binary mode. Opens an existing binary file for reading.
‘wb’	`'wb'`	Write binary mode. Creates a new binary file for writing.
‘ab’	`'ab'`	Append binary mode. Opens a binary file for appending data.
‘xb’	`'xb'`	Exclusive binary creation mode. Creates a new binary file for writing but raises an error if it already exists.
‘rt’	`'rt'`	Read text mode. Opens an existing text file for reading. (Default for text files)
‘wt’	`'wt'`	Write text mode. Creates a new text file for writing. (Default for text files)
‘at’	`'at'`	Append text mode. Opens a text file for appending data. (Default for text files)
‘xt’	`'xt'`	Exclusive text creation mode. Creates a new text file for writing but raises an error if it already exists.
‘r+’	`'r+'`	Read and write mode. Opens an existing file for both reading and writing.
‘w+’	`'w+'`	Write and read mode. Creates a new file for reading and writing. Overwrites the file if it already exists.
‘a+’	`'a+'`	Append and read mode. Opens a file for both appending and reading. Creates the file if it doesn't exist.
‘x+’	`'x+'`	Exclusive creation and read/write mode. Creates a new file for reading and writing but raises an error if it already exists.

In Python, you can copy the contents of one file to another by reading from the source file and writing to the destination file. Here's an example code snippet that demonstrates this:

```python
# Open the source file for reading
with open('source.txt', 'r') as source_file:
    # Open the destination file for writing
    with open('destination.txt', 'w') as destination_file:
        # Read lines from the source file and copy them to the destination file
        for line in source_file:
            destination_file.write(line)
    # Destination file is automatically closed when the 'with' block exits
# Source file is automatically closed when the 'with' block exits
```

```python
import pandas as pd

# Read the CSV file into a DataFrame
df = pd.read_csv('your_file.csv')
```

```python
import pandas as pd

# Creating a DataFrame from a dictionary
data = {'Name': ['Alice', 'Bob', 'Charlie', 'David'],
        'Age': [25, 30, 35, 28],
        'City': ['New York', 'San Francisco', 'Los Angeles', 'Chicago']}

df = pd.DataFrame(data)

print(df)

```

Column Selection:

You can select a single column from a DataFrame by specifying the column name within double brackets. Multiple columns can be selected in a similar manner, creating a new DataFrame.

```python
print(df['Name'])  # Access the 'Name' column
```

Accessing Rows:

You can access rows by their index using .iloc[] or by label using .loc[].

```python
print(df.iloc[2])   # Access the third row by position
print(df.loc[1])    # Access the second row by label
```

DataFrames provide numerous attributes and methods for data manipulation and analysis, including:

- **shape**: Returns the dimensions (number of rows and columns) of the DataFrame.
- **info()**: Provides a summary of the DataFrame, including data types and non-null counts.
- **describe()**: Generates summary statistics for numerical columns.
- **head(), tail()**: Displays the first or last n rows of the DataFrame.
- **mean(), sum(), min(), max()**: Calculate summary statistics for columns.
- **sort_values()**: Sort the DataFrame by one or more columns.
- **groupby()**: Group data based on specific columns for aggregation.
- **fillna(), drop(), rename()**: Handle missing values, drop columns, or rename columns.
- **apply()**: Apply a function to each element, row, or column of the DataFrame.

Slicing:

You can slice DataFrames to select specific rows and columns.

```
print(df[['Name', 'Age']])  # Select specific columns
print(df[1:3])              # Select specific rows
```

Finding Unique Elements:

Use the unique method to determine the unique elements in a column of a Data

```
unique_dates = df['Age'].unique()
```

Conditional Filtering:

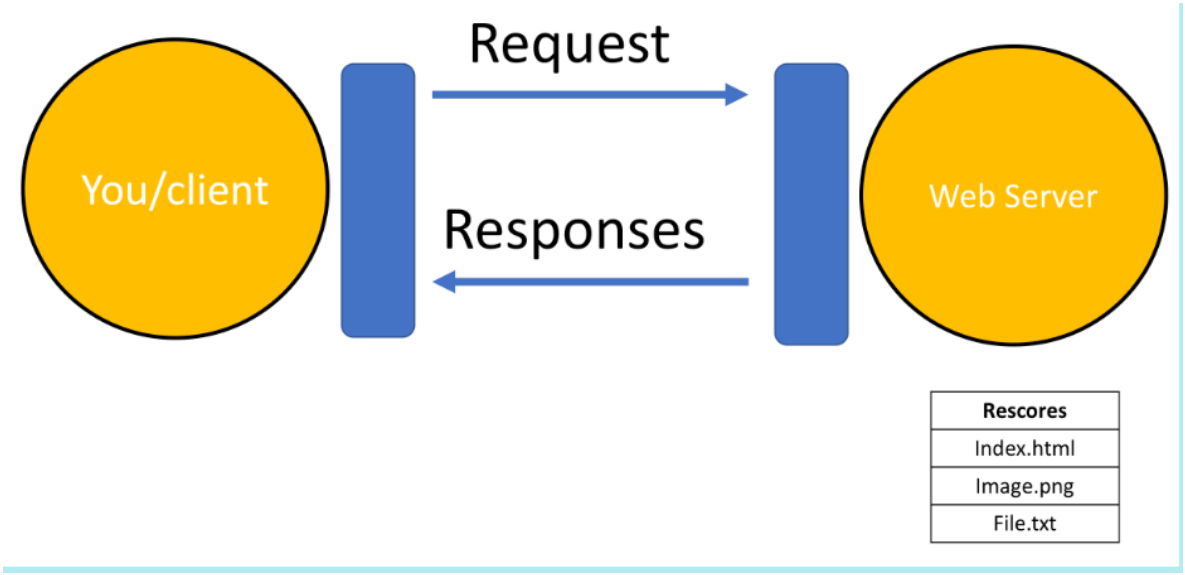
You can filter data in a DataFrame based on conditions using inequality operators.
For instance, you can filter albums released after a certain year.

```
high_above_102 = df[df['Age'] > 25]
```

Saving DataFrames:

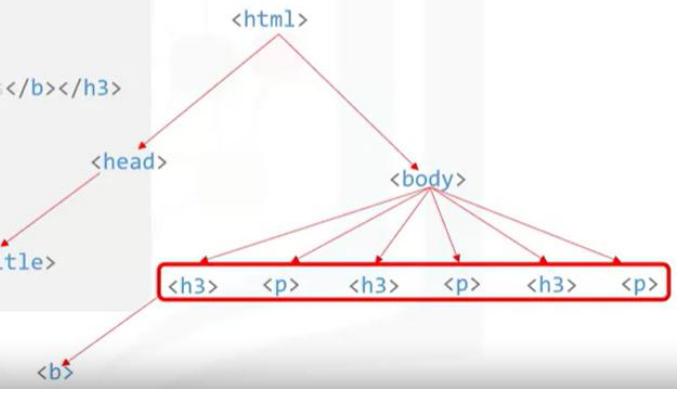
To save a DataFrame to a CSV file, use the to_csv method and specify the filename with a ".csv" extension.Pandas provides other functions for saving DataFrames in different formats.

```
df.to_csv('trading_data.csv', index=False)
```

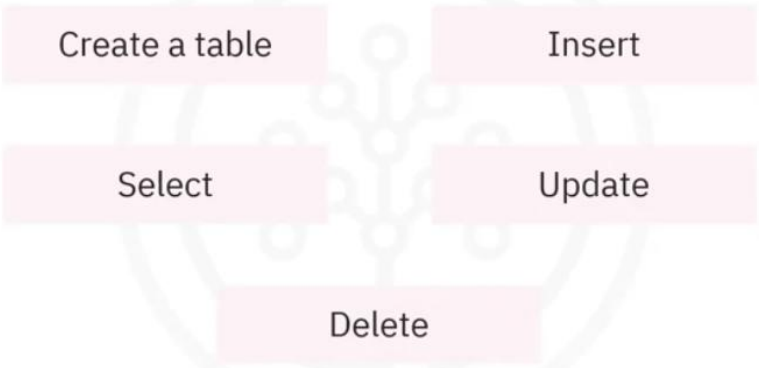


Document Tree

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
  </head>
  <body>
    <h3>  <b id='boldest'> Lebron James</b></h3>
    <p> Salary: $ 92,000,000 </p>
    <h3> Stephen Curry</h3>
    <p> Salary: $85,000, 000 </p>
    <h3> Kevin Durant </h3>
    <p> Salary: $73,200, 000</p>
   </body>
</html>
```



Basic SQL Commands

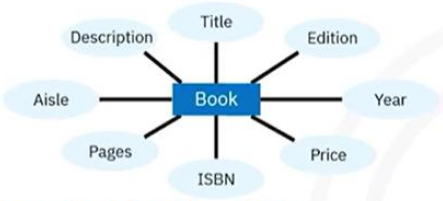


Retrieving rows from a table

- After creating a table and inserting data into the table, we want to see the data
- SELECT statement
  - A Data Manipulation Language (DML) statement used to read and modify data

```
Select statement: Query
Result from the query:  Result set/table

Select * from <tablename>
```

Using the SELECT Statement



Title	Edition	Year	Price	ISBN	Pages	Aisle	Description
Database Fundamentals	1	2010	24.99	978-0-9800628 3-1-1	300	DB-A02	Teaches you the fundamentals of databases
Getting started with DB2 Express-C	1	2010	24.99	978-0-9866628 3-5-1	280	DB-A01	Teaches you the essentials of DB2 using DB2 Express-C

Example: select * from Book

```
db2 => select * from Book
```

Book_ID	Title	Edition	Year	Price	ISBN	Pages	Aisle	Description
B1	Getting started with DB2 Express-C	1	2010	24.99	978-0-98666283-5-1	280	DB-A01	Teaches you the essentials of DB2 using DB2 Express-C
B2	Database Fundamentals	1	2010	24.99	978-0-98006283-1-1	300	DB-A02	Teaches you the fundamentals of databases
B3	Getting started with DB2 App Dev	1	2011	35.99	978-0-98086283-4-1	345	DB-A03	Teaches you the essentials of developing applications for DB2.
B4	Getting started with WAS CE	1	2010	49.99	978-0-98946283-3-1	458	DB-A04	Teaches you the essentials of WebSphere Application Server
4 record(s) selected.								

Retrieving a subset of the columns

- You can retrieve just the columns you want
  - SELECT <column 1>, <column 2> from Book

```
db2 => select book_id, title from Book
```

Book_ID	Title
B1	Getting started with DB2 Express-C
B2	Database Fundamentals
B3	Getting started with DB2 App Dev
B4	Getting started with WAS CE
4 record(s) selected.	

SELECT Title, Director, Writer FROM FilmLocations;

Restricting the Result Set: WHERE Clause

- Restricts the result set
- Always requires a Predicate:
  - Evaluates to: True, False, or Unknown
  - Used in the search condition of the Where clause

```
select book_id, title from Book
       WHERE predicate
```

```
db2 => select book_id, title from Book
       WHERE book_id='B1'
```

Book_ID	Title
B1	Getting started with DB2 Express-C
1 record(s) selected	

SELECT Title, ReleaseYear, Locations FROM FilmLocations WHERE ReleaseYear>=2001;

1. **SELECT Title, ProductionCompany, Locations, ReleaseYear FROM FilmLocations WHERE Writer<>"James Cameron";**

Equal to	=
Greater than	>
Lesser than	<
Greater than or equal to	>=
Less than or equal to	<=
Not equal to	<>

1. The **general syntax** of a SELECT statement retrieves the data under the listed columns from Table_1. The code is:

```
1  SELECT COLUMN1, COLUMN2, ... FROM TABLE_1 ;
```

COUNT

Example:

```
select COUNT(COUNTRY) from MEDALS
       where COUNTRY='CANADA'
```

Result:

```
             1
--------------
            29
```

Database: SanFranciscoFilmLocations

```
SELECT COUNT(Locations) FROM FilmLocations WHERE Writer="James Cameron";
```

Submit query

Results

All commands ran successfully

SELECT COUNT(Locations) FROM FilmLocations WHERE Writer="James Cameron"

COUNT(Locations)
48

Retrieve the number of rows having a release year older than 1950 from the "FilmLocations" table.

```
SELECT Count(*) FROM FilmLocations WHERE ReleaseYear<1950;
```

Submit query

Results

All commands ran successfully

SELECT Count(*) FROM FilmLocations WHERE ReleaseYear<1950

Count(*)
62

DISTINCT

2. DISTINCT

Removes duplicate values from a result set.

DISTINCT

List of unique countries that received GOLD medals:

```
select DISTINCT COUNTRY from MEDALS
        where MEDALTYPE = 'GOLD'
```

DISTINCT

1. Retrieve the names of all unique films released in the 21st century and onwards, along with their release years.

```
SELECT DISTINCT Title, ReleaseYear FROM FilmLocations WHERE ReleaseYear>=2001;
```

Database: SanFranciscoFilmLocations

```
1 SELECT COUNT(DISTINCT Distributor) FROM FilmLocations WHERE Actor1="Clint Eastwood";
```

Tip: Autocomplete with Ctrl+Enter or Cmd+Enter

Submit query

Results

All commands ran successfully

Support

SELECT COUNT(DISTINCT Distributor) FROM FilmLocations WHERE Actor1="Clint Eastwood"

COUNT(DISTINCT Distributor)
3

Powered by Datasette

LIMIT

3. LIMIT

Restricts the number of rows retrieved from the database.

LIMIT

Retrieve just the first 10 rows in a table:

```
select * from tablename LIMIT 10
```

LIMIT

Retrieve 5 rows in the MEDALS table for a particular year:

```
select * from MEDALS
        where YEAR = 2018 LIMIT 5
```

Result:

COUNTRY	GOLD	SILVER	BRONZE	TOTAL	YEAR
Norway	14	14	11	39	2018
Germany	14	10	7	31	2018
Canada	11	8	10	29	2018
United States	9	8	6	23	2018
Netherlands	8	6	6	20	2018

SELECT DISTINCT Title FROM FilmLocations WHERE ReleaseYear=2015 LIMIT 10;

Database: SanFranciscoFilmLocations

```
1 SELECT DISTINCT Title FROM FilmLocations WHERE ReleaseYear=2015 LIMIT 3 OFFSET 5;
```

Tip: Autocomplete with Ctrl+Enter or Cmd+Enter

Submit query

Results

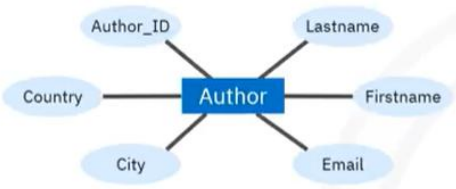
All commands ran successfully

Support

SELECT DISTINCT Title FROM FilmLocations WHERE ReleaseYear=2015 LIMIT 3 OFFSET 5

Title
I Am Michael
Steve Jobs
Quitters

Inserting multiple rows



Author_ID	Lastname	Firstname	Email	City	Country
A1	Chong	Raul	rfc@ibm.com	Toronto	CA
A2	Ahuja	Rav	ra@ibm.com	Toronto	CA
A3	Hakes	Ian	ih@ibm.com	Toronto	Ca
A4	Sharma	Neeraj	ns@ibm.com	Chennai	IN
A5	Perniu	Liviu	lp@ibm.com	Transylvania	RO

```
INSERT INTO AUTHOR
       (AUTHOR_ID, LASTNAME, FIRSTNAME, EMAIL, CITY, COUNTRY)
VALUES
       ('A1', 'Chong', 'Raul', 'rfc@ibm.com', 'Toronto', 'CA'),
       ('A2', 'Ahuja', 'Rav', 'ra@ibm.com', 'Toronto', 'CA')
```

When using the UPDATE statement, if you do not specify the WHERE clause, all the rows in the table are updated.

- (•) True

INSERT INTO Instructor(ins_id, lastname, firstname, city, country)

VALUES(5, 'Doe', 'John', 'Sydney', 'AU'), (6, 'Doe', 'Jane', 'Dhaka', 'BD');

Luego tienes que Volver a poner el código SELECT * FROM Instructor; para que aparezca la tabla actualizada.

Using the DELETE statement

Author_Id	LastName	FirstName	Email	City	Country
A1	Chong	Raul	rfc@ibm.com	Toronto	CA
A2	Ahuja	Rav	ra@ibm.com	Toronto	CA
A3	Hakes	Ian	ih@ibm.com	Toronto	CA
A4	Sharma	Neeraj	ns@ibm.com	Chennai	IN
A5	Perniu	Liviu	lp@ibm.com	Transylvania	RO

```
DELETE FROM AUTHOR
   WHERE AUTHOR_ID IN ('A2', 'A3')
```

Author_Id	LastName	FirstName	Email	City	Country
A1	Chong	Raul	rfc@ibm.com	Toronto	CA
A4	Sharma	Neeraj	ns@ibm.com	Chennai	IN
A5	Perniu	Liviu	lp@ibm.com	Transylvania	RO

If no WHERE clause is used, all rows will be removed

```
1  UPDATE Instructor
2  SET city='Toronto'
3  WHERE firstname="Sandip";
```

Copy the solution code above by clicking on the little copy bu
the codeblock below and paste it to the textbox of the Datase
query.

Copy the code below by clicking on the little copy button on t
codeblock below and paste it to the textbox of the Datasette t
query.

```
1  SELECT * FROM Instructor;
```

Your output resultset should look like the image below:

Results

All commands ran successfully

SELECT * FROM Instructor

ins_id	lastname	firstname	city	country
1	Ahuja	Rav	Toronto	CA
2	Chong	Raul	Toronto	CA
3	Vasudevan	Hima	Chicago	US
4	Saha	Sandip	Toronto	CA
5	Doe	John	Sydney	AU
6	Doe	Jane	Dhaka	BD
7	Cangiano	Antonio	Vancouver	CA
8	Ryan	Steve	Barlby	GB

ALTER TABLE

ALTER TABLE statements can be used to add or remove columns from a table, to modify the data type of columns, to add or remove keys, and to add or remove constraints. The syntax of the ALTER TABLE statement is:

ADD COLUMN syntax

```
ALTER TABLE table_name
ADD column_name data_type;
```

A variation of the syntax for adding column is:

```
ALTER TABLE table_name
ADD COLUMN column_name data_type;
```

For example, to add a **telephone_number** column to the **author** table in the **library** database, the statement will be written as:

```
ALTER TABLE author
ADD telephone_number BIGINT;
```

Here, BIGINT is a data type for Big Integer.
After adding the entries to the new column, a sample output is shown below.

author_id	lastname	firstname	email	city	country	telephone_number
1001	Thomas	John	johnt@...	New York	USA	5551111
1002	James	Alice	alicej@...	Seattle	USA	5551112
1003	Wells	Steve	stevew:@...	Montreal	Canada	5552222
1004	Kumar	Santosh	kumars@...	London	UK	5553333

Modify column data type

```
ALTER TABLE table_name
MODIFY column_name data_type;
```

Sometimes, the data presented may be in a different format than required. In such a case, we need to modify the data_type of the column. For example, using a **numeric** data type for **telephone_number** means you cannot include **parentheses**, **plus signs**, or **dashes as part of the number**. For such entries, the appropriate choice of data_type is CHAR.

To modify the data type, the statement will be written as:

```
ALTER TABLE author
MODIFY telephone_number CHAR(20);
```

author_id	lastname	firstname	email	city	country	telephone_number
1001	Thomas	John	johnt@...	New York	USA	555-1111
1002	James	Alice	alicej@...	Seattle	USA	555-1112
1003	Wells	Steve	stevew@...	Montreal	Canada	555-2222
1004	Kumar	Santosh	kumars@...	London	UK	555-3333

TRUNCATE Table

TRUNCATE TABLE statements are used to delete all of the rows in a table. The syntax of the statement is:

```
TRUNCATE TABLE table_name;
```

So, to truncate the "author" table, the statement will be written as:

```
TRUNCATE TABLE author;
```

author_id	lastname	firstname	email	city	country

Note: *The TRUNCATE statement will delete the rows and not the table.*

CREATE TABLE statement

In the previous video, we saw the general syntax to create a table:

```
CREATE TABLE TableName (
   COLUMN1 datatype,
   COLUMN2 datatype,
   COLUMN3 datatype,
   ...
);
```

Consider the following examples:

1. Create a TEST table with two columns - ID of type integer and NAME of type varchar. For this, we use the following SQL statement.

```
CREATE TABLE TEST (
   ID int,
   NAME varchar(30)
);
```

2. Create a COUNTRY table with an integer ID column, a two-letter country code column, and a variable length country name column. For this, we may use the following SQL statement.

```
CREATE TABLE COUNTRY (
   ID int,
   CCODE char(2),
   Name varchar(60)
);
```

3. In the example above, make ID a primary key. Then, the statement will be modified as shown below.

```
CREATE TABLE COUNTRY (
   ID int NOT NULL,
   CCODE char(2),
   Name varchar(60)
   PRIMARY KEY (ID)
);
```

In the above example, the ID column has the **NOT NULL** constraint added after the datatype, meaning that it cannot contain a NULL or an empty value. This is added since the database does not allow Primary Keys to have NULL values.

DROP TABLE

If the table you are trying to create already exists in the database, you will get an error indicating **table XXX.YYY already exists**. To circumvent this error, create a table with a different name or first DROP the existing table. It is common to issue a DROP before doing a CREATE in test and development scenarios.

The syntax to drop a table is:

```
1   DROP TABLE TableName;
```

For example, consider that you wish to drop the contents of the table COUNTRY if a table exists in the dataset with the same name. In such a case, the code for the last example becomes

```
1  DROP TABLE COUNTRY;
2  CREATE TABLE COUNTRY (
3     ID int NOT NULL,
4     CCODE char(2),
5     Name varchar(60)
6     PRIMARY KEY (ID)
7  );
```

WARNING: Before dropping a table, ensure it doesn't contain important data that can't be recovered easily.

Note that if the table does not exist and you try to drop it, you will see an error like **XXX.YYY is an undefined name**. You can ignore this error if the subsequent CREATE statement is executed successfully.

- **Create views**
  Views are virtual tables that allow you to query data from multiple tables as if they were a single table. You can use SQL scripts to create views that simplify complex queries and make it easier to work with your data.

- **Create stored procedures**
  Stored procedures are precompiled SQL statements that can be executed on demand. You can use SQL scripts to create stored procedures that encapsulate complex business logic and make it easier to manage your database.

- **Create triggers**
  Triggers are special types of stored procedures that are automatically executed in response to certain events, such as an insert, update, or delete operation. You can use SQL scripts to create triggers that enforce business rules and maintain data integrity.

The steps below explain loading data into the tables you created in Task 2.

1. Download the 5 CSV files below to your local machine.
    - Patients.csv
    - MedicalHistory.csv
    - MedicalProcedures.csv
    - MedicalDepartments.csv
    - MedicalLocations.csv

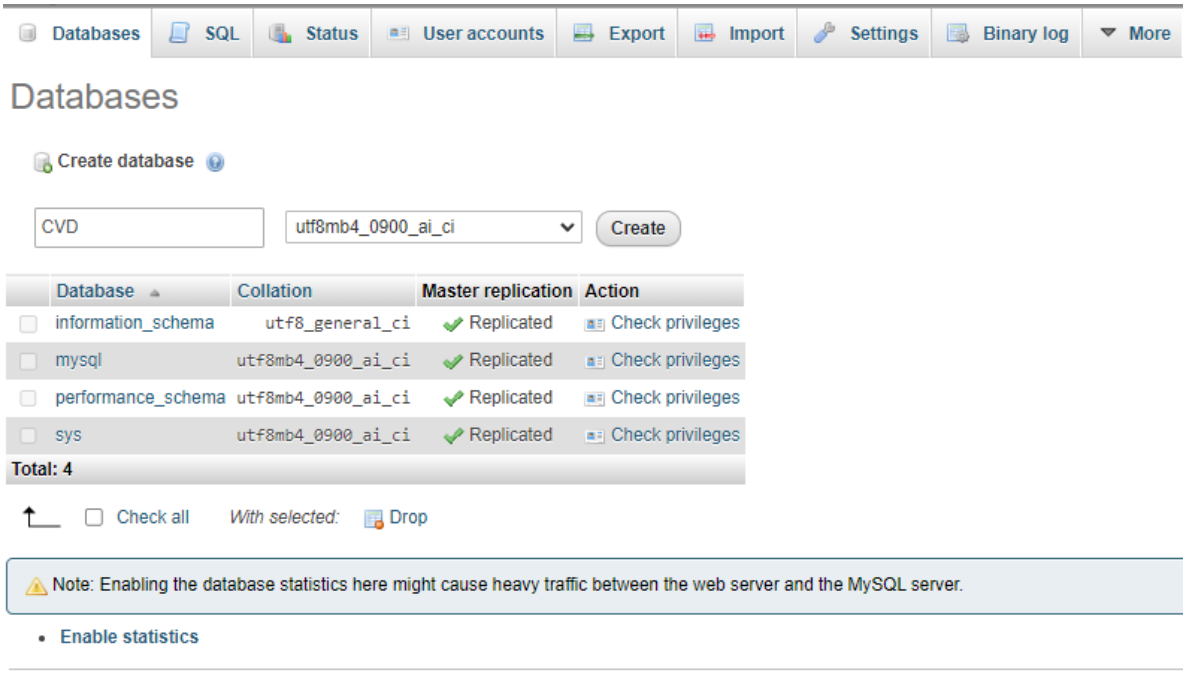
The steps to load a CSV to a table are as follows.

- Select the table.
- Click the Import tab.
- Browse to the location of the CSV file and click 'Go' to load the CSV file.

The images below share how to load the CSV data to the `PATIENTS` table.

Once the table is loaded, you will get a message that the records are inserted successfully.

Further, you can click on browse and view the table's data.


Databases
SQL
Status
User accounts
Export
Import
Settings
Binary log
More
Databases
Create database
CVD
utf8mb4_0900_ai_ci
Create
Database
Collation
Master replication
Action
information_schema
utf8_general_ci
Replicated
Check privileges
mysql
utf8mb4_0900_ai_ci
Replicated
Check privileges
performance_schema
utf8mb4_0900_ai_ci
Replicated
Check privileges
sys
utf8mb4_0900_ai_ci
Replicated
Check privileges
Total: 4
Check all
With selected:
Drop
Note: Enabling the database statistics here might cause heavy traffic between the web server and the MySQL server.
Enable statistics

`ALTER TABLE` statement is used to add the columns to a table.

`ALTER TABLE DROP COLUMN` statement is used to remove columns from a table.	MySQL/DB2: `ALTER TABLE employee DROP COLUMN mobile ;`
MySQL: `ALTER TABLE MODIFY` MODIFY clause is used with the ALTER TABLE statement to modify the data type of columns. Db2: `ALTER TABLE ALTER COLUMN` statement is used to modify the data type of columns.	MySQL: `ALTER TABLE employee MODIFY mobile CHAR(20);` DB2: `ALTER TABLE employee ALTER COLUMN mobile SET DATA TYPE CHAR(20);`

MySQL: `ALTER TABLE CHANGE COLUMN` CHANGE COLUMN clause is used to rename the columns in a table. DB2: `ALTER TABLE RENAME COLUMN` statement is used to rename the columns in a table.	MySQL: `ALTER TABLE employee CHANGE COLUMN first_name name VARCHAR(255);` DB2: `ALTER TABLE employee RENAME COLUMN first_name TO name;`
MySQL: `TRUNCATE TABLE` statement is used to delete all of the rows in a table. Db2: The `IMMEDIATE` specifies to process the statement immediately and that it cannot be undone.	MySQL: `TRUNCATE TABLE employee;` DB2: `TRUNCATE TABLE employee IMMEDIATE ;`
Use the `DROP TABLE` statement to delete a table from a database. If you delete a table that contains data, by default the data will be deleted alongside the table.	MySQL/DB2: `DROP TABLE employee ;`

DDL

Significa Data Definition Language, o Lenguaje de Definición de Datos. Se utiliza para crear, modificar y eliminar estructuras de bases de datos, como tablas, índices, vistas y esquemas.

DML

Significa Data Manipulation Language, o Lenguaje de Manipulación de Datos. Se utiliza para insertar, actualizar y eliminar datos dentro de una base de datos.

DQL

Significa Data Query Language, o Lenguaje de Consulta de Datos. Permite obtener datos de la base de datos e imponerles un orden. Incluye la sentencia SELECT, que permite extraer los datos de la base de datos para realizar operaciones con ellos.

Retrieving rows - using a String Pattern

```
db2 => select firstname from Author
         WHERE firstname like 'R%'

Firstname
---------------------------------------------------
Raul
Rav
    2 record(s) selected.
```

Retrieving rows - using a Range

```
db2 => select title, pages from Book
          WHERE pages >= 290 AND pages <= 300

Title                                   Pages
--------------------------------------------------------------------
Database Fundamentals                   300
Getting started with DB2 App Dev        298
    2 record(s) selected.
```

```
db2 => select title, pages from Book
          WHERE pages >= 290 AND pages <= 300
```

Title	Pages
Database Fundamentals	300
Getting started with DB2 App Dev	298

2 record(s) selected.

```
db2 => select title, pages from Book
          WHERE pages between 290 and 300
```

Title	Pages
Database Fundamentals	300
Getting started with DB2 App Dev	298

2 record(s) selected.

ORDER BY clause – Descending order

```
db2 => select title from Book
          ORDER BY title
```

Title
Database Fundamentals
Getting started with DB2 App Dev
Getting started with DB2 Express-C
Getting started with WAS CE

4 record(s) selected.

Ascending order by default

```
db2 => select title from Book
          ORDER BY title DESC
```

Title
Getting started with WAS CE
Getting started with DB2 Express-C
Getting started with App Dev
Database Fundamentals

4 record(s) selected.

Descending order with DESC keyword

En SQL, cuando utilizas la cláusula `ORDER BY` sin especificar el tipo de orden, el orden predeterminado es **ascendente**. Esto significa que, al escribir:

sql Copiar código

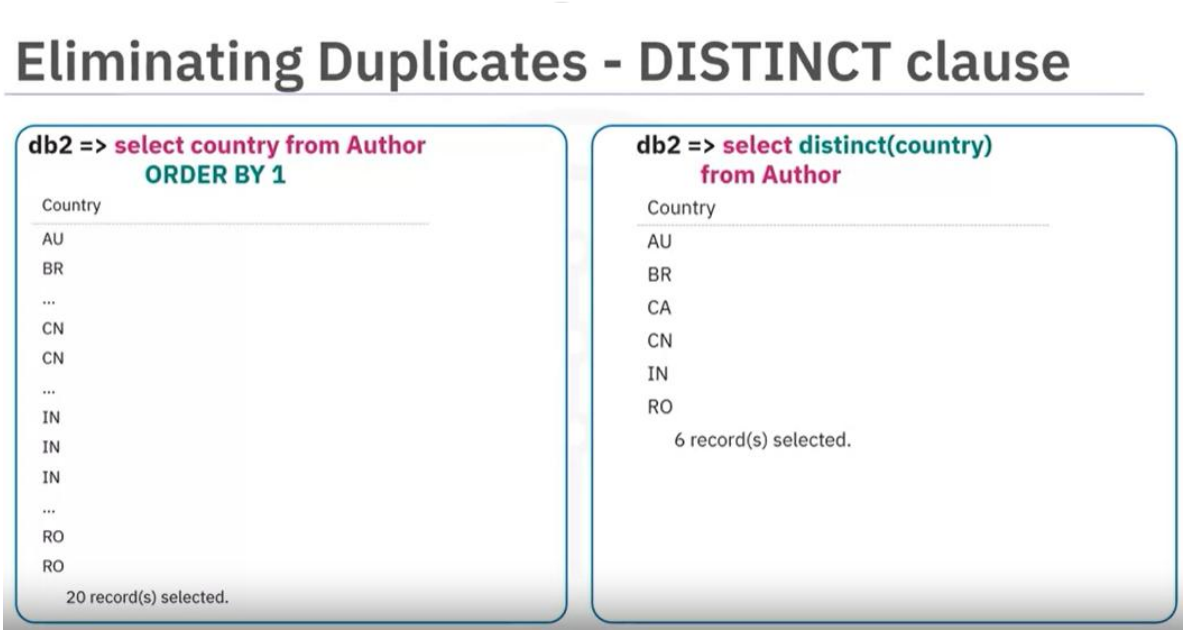
```sql
SELECT *
FROM EMPLOYEES
ORDER BY B_DATE;
```

El resultado se ordenará de manera **ascendente** (de la fecha más antigua a la más reciente) por la columna `B_DATE`, ya que no se ha especificado de manera explícita si debe ser ascendente (`ASC`) o descendente (`DESC`).

Si deseas que el orden sea descendente, deberías especificarlo de manera explícita:

sql Copiar código

```sql
SELECT *
FROM EMPLOYEES
ORDER BY B_DATE DESC;
```



GROUP BY clause

```
db2 => select country from Author
       ORDER BY 1

Country
AU
BR
...
CN
CN
...
IN
IN
IN
...
RO
RO
  20 record(s) selected.
```

```
db2 => select country, count(country)
       from Author GROUP BY country

Country 2
AU      1
BR      1
CA      3
CN      6
IN      6
RO      3
  6 record(s) selected.
```

```
db2 => select country, count(country)
       as Count from Author group by country

Country Count
AU      1
BR      1
CA      3
CN      6
IN      6
RO      3
  6 record(s) selected.
```

```
db2 => select country, count(country)
       as Count from Author
       group by country
       having count(country) > 4
```

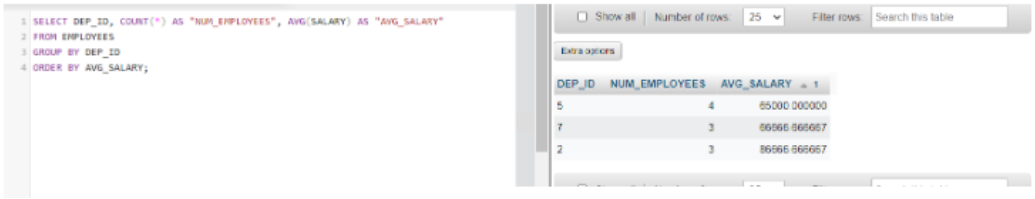
Country	Count
CN	6
IN	6

6 record(s) selected.

```sql
SELECT DEP_ID, COUNT(*) AS "NUM_EMPLOYEES", AVG(SALARY) AS "AVG_SALARY"
FROM EMPLOYEES
GROUP BY DEP_ID;
```

```
SELECT DEP_ID, COUNT(*) AS "NUM_EMPLOYEES", AVG(SALARY) AS "AVG_SALARY"
FROM EMPLOYEES
GROUP BY DEP_ID
ORDER BY AVG_SALARY;
```

The output of the query should look like:



In case you need to filter a grouped response, you have to use the HAVING clause. In the previous example, if we wish to limit the result to departments with fewer than 4 employees, We will have to use HAVING after the GROUP BY, and use the count() function in the HAVING clause instead of the column label.

```
SELECT DEP_ID, COUNT(*) AS "NUM_EMPLOYEES", AVG(SALARY) AS "AVG_SALARY"
FROM EMPLOYEES
GROUP BY DEP_ID
HAVING count(*) < 4
ORDER BY AVG_SALARY;
```

- Puede utilizar la cláusula WHERE para refinar los resultados de su consulta.
- La condición de búsqueda de la cláusula WHERE utiliza un predicado para refinar la búsqueda.
- Puede utilizar el carácter comodín (%) como sustituto de caracteres desconocidos en un patrón.
- Puede utilizar BETWEEN ... AND ... para especificar un rango de números.
- Puede ordenar los resultados de la consulta en orden ascendente o descendente, utilizando la cláusula ORDER BY para especificar la columna sobre la que ordenar.
- Puede agrupar los resultados de la consulta utilizando la cláusula GROUP BY.

Column Alias

Example 2: Explicitly name the output column SUM_OF_COST:

```
select SUM(COST) as SUM_OF_COST
       from PETRESCUE
```

Example 2: Results:

```
SUM_OF_COST
1718.24
```

MIN, MAX

MIN: Return the MINIMUM value
MAX: Return the MAXIMUM value

Example 3A. Get the maximum QUANTITY of any ANIMAL:

```
select MAX(QUANTITY) from PETRESCUE
```

Example 3B. Results:

```
1
24
```

Example 3B. Get the minimum value of ID column for Dogs:

```
select MIN(ID) from PETRESCUE where ANIMAL =  'Dog'
```

Mathematical operations can be performed between columns.

Example 5. Calculate the average COST per ‘Dog’:

```
select AVG(COST / QUANTITY) from PETRESCUE
where ANIMAL =  'Dog'
```

Example 5. Results:

```
1
127.270000000000000000000000
```

Example 10: Use the DISTINCT() function to get unique values :

```
select DISTINCT(UCASE(ANIMAL)) from PETRESCUE
```

Example 10: Results:

```
1
CAT
DOG
GOLDFISH
HAMSTER
PARROT
```

In case the question was to round the value to 2 decimal places, the query would change to:

```
1  SELECT ROUND(COST, 2) FROM PETRESCUE;
```

For this query, we will use the function `DAY(COLUMN_NAME)`. The output of this query will be only the `DAY` part of the date in the column. The query for this question can be written as:

```
1  SELECT DAY(RESCUEDATE) FROM PETRESCUE;
```

In case the query was asking for `MONTH` of rescue, the query would change to:

```
1  SELECT MONTH(RESCUEDATE) FROM PETRESCUE;
```

In case the query was asking for `YEAR` of rescue, the query would change to:

```
1  SELECT YEAR(RESCUEDATE) FROM PETRESCUE;
```

sql Copiar código

```
SELECT DATE_SUB(RESCUEDATE, INTERVAL 3 DAY)
FROM PETRESCUE;
```

This query will return a result set where each `RESCUEDATE` has been reduced by 3 days.

For example, if a `RESCUEDATE` is `2024-10-05`, the result would show `2024-10-02`.

For this query, we will use the function `DATEDIFF(Date_1, Date_2)`. This function calculates the difference between the two given dates and gives the output in number of days. For the given question, the query would be:

```
SELECT DATEDIFF(CURRENT_DATE, RESCUEDATE) FROM PETRESCUE
```

To present the output in a YYYY-MM-DD format, another function `FROM_DAYS(number_of_days)` can be used. This function takes a number of days and returns the required formatted output. The query above would thus be modified to

```
SELECT FROM_DAYS(DATEDIFF(CURRENT_DATE, RESCUEDATE)) FROM PETRESCUE
```

Say you are asked to retrieve all employee records whose salary is lower than the average salary. You might use the following query to do this.

```
SELECT *
FROM EMPLOYEES
WHERE salary < AVG(salary);
```

However, this query will generate an error stating, "Illegal use of group function." Here, the group function is `AVG` and cannot be used directly in the condition since it has not been retrieved from the data. Therefore, the condition will use a sub-query to retrieve the average salary information to compare the existing salary. The modified query would become:

```
SELECT *
FROM EMPLOYEES
WHERE SALARY < (SELECT AVG(SALARY) FROM EMPLOYEES);
```

Now, consider executing a query that retrieves all employee records with EMP_ID, SALARY, and maximum salary as MAX_SALARY in every row. For this, the maximum salary must be queried and used as one of the columns. This can be done using the query below.

```
SELECT EMP_ID, SALARY, (SELECT MAX(SALARY) FROM EMPLOYEES) AS MAX_SALARY
FROM EMPLOYEES;
```

Now, consider that you wish to extract the first and last names of the oldest employee. Since the oldest employee will be the one with the smallest date of birth, the query can be written as:

```
SELECT F_NAME, L_NAME
FROM EMPLOYEES
WHERE B_DATE = (SELECT MIN(B_DATE) FROM EMPLOYEES);
```

You may also use sub-queries to create derived tables, which can then be used to query specific information. Say you want to know the average salary of the top 5 earners in the company. You will first have to extract a table of the top five salaries as a table. From that table, you can query the average value of the salary. The query can be written as follows.

```
SELECT AVG(SALARY)
FROM (SELECT SALARY
      FROM EMPLOYEES
      ORDER BY SALARY DESC
      LIMIT 5) AS SALARY_TABLE;
```

Note that it is necessary to give an alias to any derived tables.

1. Retrieve only the EMPLOYEES records corresponding to jobs in the JOBS table.

For such a question, you can implement the sub-query in the WHERE clause, such that the overlapping column of JOD ID can identify the required entries.

```
SELECT * FROM EMPLOYEES WHERE JOB_ID IN (SELECT JOB_IDENT FROM JOBS);
```

The expected output would look as shown below.

+ Options

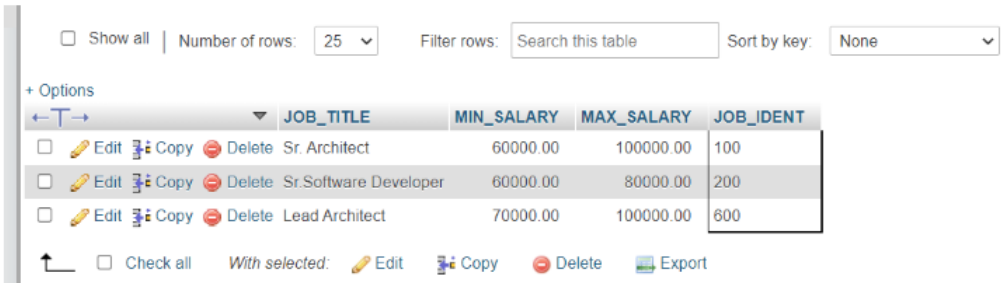
	EMP_ID	F_NAME	L_NAME	SSN	B_DATE	SEX	ADDRESS	JOB_ID	SALARY	MANAGER_ID	DEP_ID
Edit Copy Delete	E1001	John	Thomas	123456	1976-09-01	M	5631 Rice, OakPark,IL	100	100000.00	30001	2
Edit Copy Delete	E1002	Alice	James	123457	1972-07-31	F	980 Berry ln, Elgin,IL	200	80000.00	30002	5
Edit Copy Delete	E1003	Steve	Wells	123458	1980-10-08	M	291 Springs, Gary,IL	300	50000.00	30002	5
Edit Copy Delete	E1004	Santosh	Kumar	123459	1985-07-20	M	511 Aurora Av, Aurora,IL	400	60000.00	30004	5
Edit Copy Delete	E1005	Ahmed	Hussain	123410	1981-04-01	M	216 Oak Tree, Geneva,IL	500	70000.00	30001	2
Edit Copy Delete	E1006	Nancy	Allen	123411	1978-06-02	F	111 Green Pl, Elgin,IL	600	90000.00	30001	2
Edit Copy Delete	E1007	Mary	Thomas	123412	1975-05-05	F	100 Rose Pl, Gary,IL	650	65000.00	30003	7
Edit Copy Delete	E1008	Bharath	Gupta	123413	1985-06-05	M	145 Berry Ln, Naperville,IL	660	65000.00	30003	7
Edit Copy Delete	E1009	Andrea	Jones	123414	1990-09-07	F	120 Fall Creek, Gary,IL	234	70000.00	30003	7
Edit Copy Delete	E1010	Ann	Jacob	123415	1982-03-30	F	111 Britany Springs,Elgin,IL	220	70000.00	30004	5

2. Retrieve JOB information for employees earning over $70,000.

For this example, retrieve the details from the JOBS table, which has common IDs with those available in the EMPLOYEES table, provided the salary in the EMPLOYEES table is greater than $70,000. You can write the query as:

```
SELECT JOB_TITLE, MIN_SALARY, MAX_SALARY, JOB_IDENT
FROM JOBS
WHERE JOB_IDENT IN (select JOB_ID from EMPLOYEES where SALARY > 70000 );
```

The expected output would look as shown below.



Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

+ Options

	JOB_TITLE	MIN_SALARY	MAX_SALARY	JOB_IDENT
Edit Copy Delete	Sr. Architect	60000.00	100000.00	100
Edit Copy Delete	Sr.Software Developer	60000.00	80000.00	200
Edit Copy Delete	Lead Architect	70000.00	100000.00	600

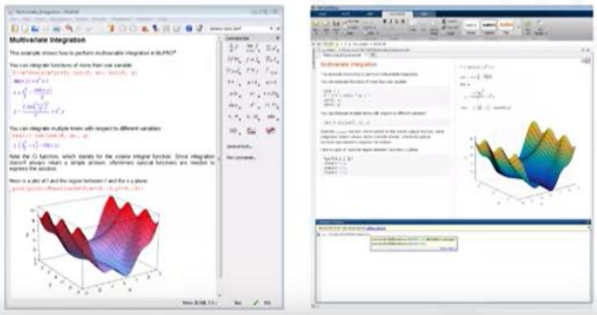
Check all | With selected: Edit Copy Delete Export

APIs used by popular SQL-based DBMS systems

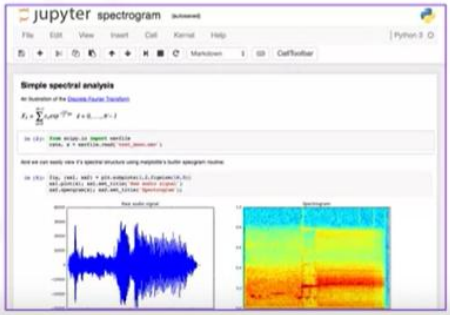
Application or Database	SQL API
MySQL	MySQL C API
PostgreSQL	psycopg2
IBM DB2	ibm_db
SQL Server	dblib API
Database access for Microsoft Windows OS	ODBC
Oracle	OCI
Java	JDBC

Notebooks allows creating and sharing documents containing live codes, equation, visualizations, and explanatory text.

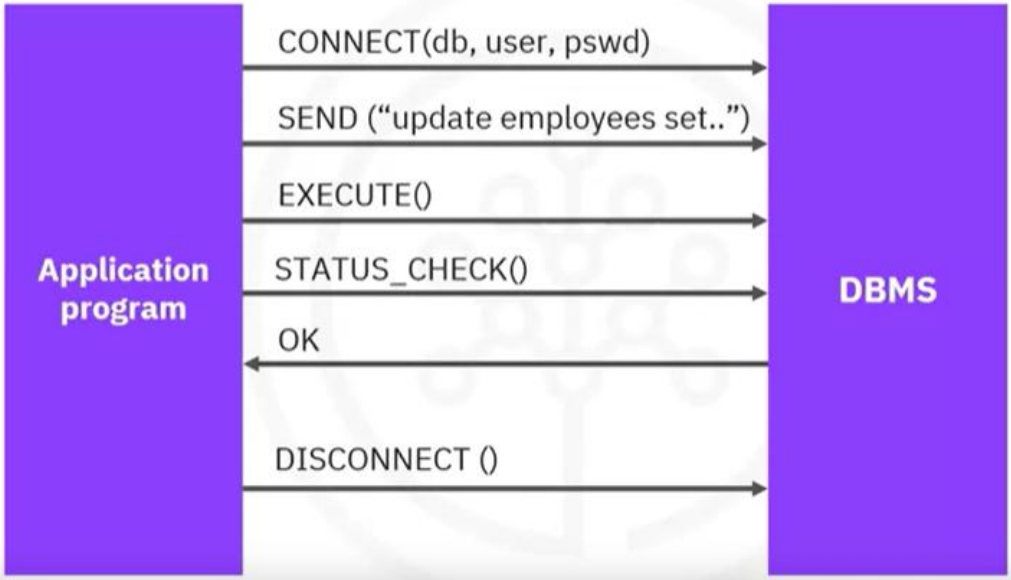
Matlab notebook



Jupyter notebook



What is a SQL API?



What are Connection methods?

- .cursor()
- .commit()
- .rollback()
- .close()

Connection Objects

- Database connections
- Manage transactions

Cursor Objects

- Database Queries
- Scroll through result set
- Retrieve results

What are cursor methods?

- .callproc()
- .execute()
- .executemany()
- .fetchone()
- .fetchmany()
- .fetchall()
- .nextset()
- .arraysize()
- .close()

Writing code using DB-API

```
from dbmodule import connect
#Create connection object
Connection =
connect('databasename',
'username', 'pswd')

#Create a cursor object
Cursor=connection.cursor()
```

```
#Run Queries
Cursor.execute('select * from
mytable')
Results=cursor.fetchall()

#Free resources
Cursor.close()
Connection.close()
```

```
statement = '''SELECT * FROM INSTRUCTOR'''
cursor_obj.execute(statement)

print("All the data")
output_all = cursor_obj.fetchall()
for row_all in output_all:
  print(row_all)
```

Types of Cell Magics

Commands that are prefixed with a single % character and operate on a single line of input.

Commands that are prefixed with two %% characters and operate on multiple lines of input.

Using Line Magic Statements

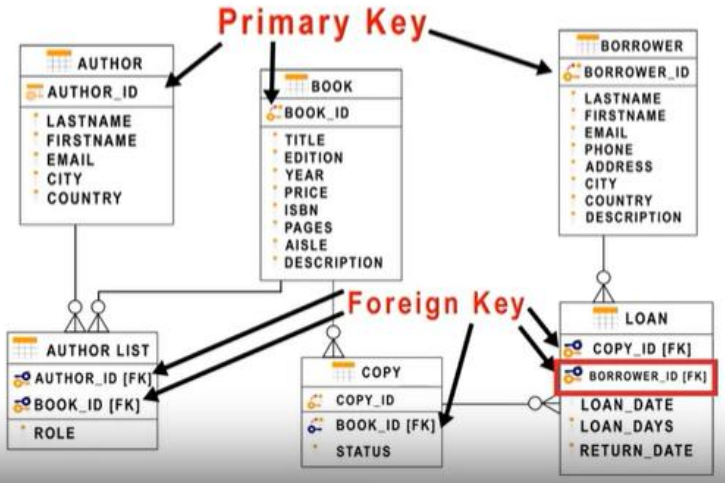
Line Magics	Uses
%pwd	prints the current working directory
%ls	lists all files in the current directory
%history	shows the command history
%reset	resets the namespace by removing all names defined by the user
%who	lists all variables in the namespace
%whos	provides more detailed information about all variables in the namespace
%matplotlib inline	makes matplotlib plots appear within the notebook
%timeit	times the execution of a single statement
%lsmagic	lists all available line magics

In this video, you learned that:

- A transaction is a unit of work which usually consists of multiple SQL statements
- In an ACID transaction all the SQL statements must complete successfully, or none at all
- ACID stands for Atomic, Consistent, Isolated, Durable
- BEGIN, COMMIT, ROLLBACK
- Can be called from languages like C, R and Python

Relational model ER diagram

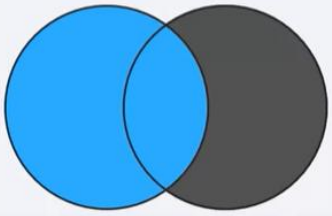
- Primary key: uniquely identifies each row in a table
- Foreign key: refers to a primary key of another table



Outer joins

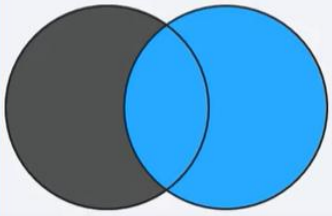
Left Outer Join

All rows from the left table to any matching rows from the right table



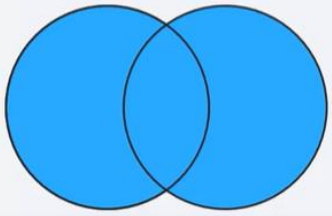
Right Outer Join

All rows from the right table to any matching rows from the left table

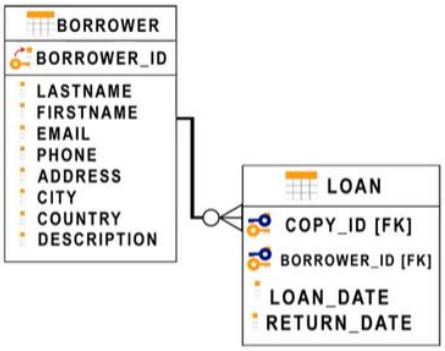


Full Outer Join

All rows from both tables



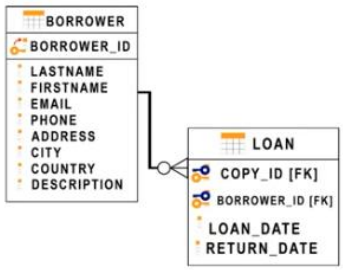
LEFT JOIN operator




```
SELECT B.BORROWER_ID, B.LASTNAME, B.COUNTRY,
       L.BORROWER_ID, L.LOAN_DATE
FROM BORROWER B LEFT JOIN LOAN L
    ON B.BORROWER_ID = L.BORROWER_ID
```

BORROWER_ID	LASTNAME	COUNTRY	BORROWER_ID	LOAN_DATE
D1	SMITH	CA	D1	11/24/2010
D2	SANDLER	CA	D2	11/24/2010
D3	SOMMERS	CA	D3	11/24/2010
D4	ARDEN	CA	D4	11/24/2010
D5	XIE	CA	D5	11/24/2010
D6	PETERS	CA	NULL	NULL
D7	LI	CA	NULL	NULL
D8	WONG	CA	NULL	NULL

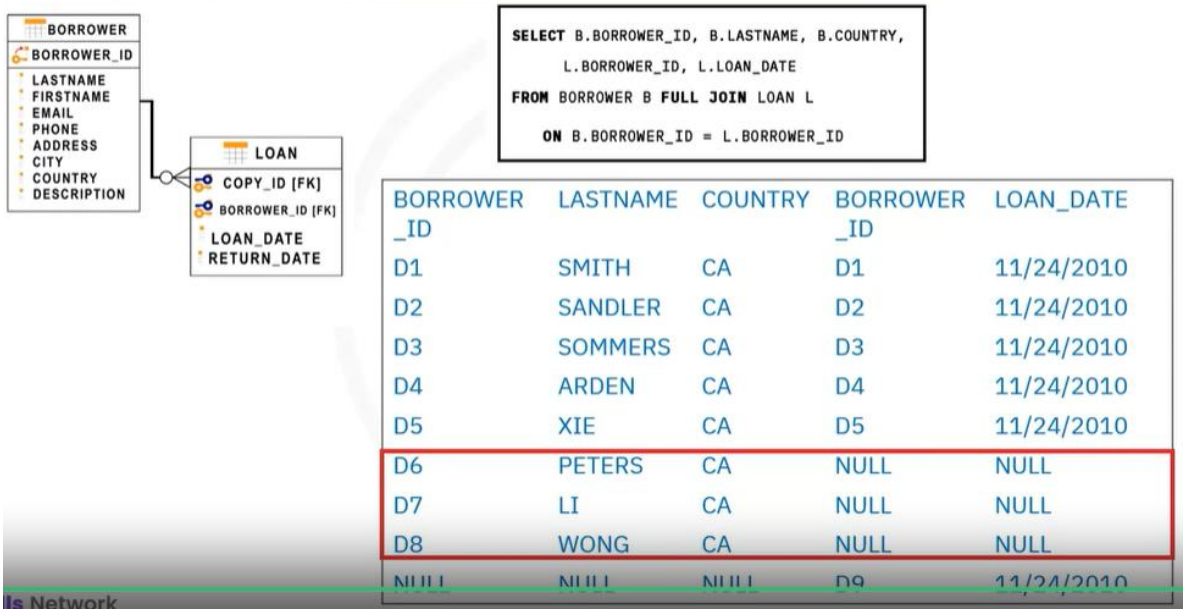
RIGHT JOIN operator




```
SELECT B. BORROWER_ID, B.LASTNAME, B.COUNTRY,
       L. BORROWER_ID, L.LOAN_DATE
FROM BORROWER B RIGHT JOIN LOAN L
    ON B.BORROWER_ID = L.BORROWER_ID
```

BORROWER_ID	LASTNAME	COUNTRY	BORROWER_ID	LOAN_DATE
D1	SMITH	CA	D1	11/24/2010
D2	SANDLER	CA	D2	11/24/2010
D3	SOMMERS	CA	D3	11/24/2010
D4	ARDEN	CA	D4	11/24/2010
D5	XIE	CA	D5	11/24/2010
NULL	NULL	NULL	D9	11/24/2010

FULL JOIN operator



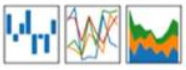
Summary

In this video, you learned that:

- Left outer joins return all rows from the left table, and all the rows from the right table that have a match
- Right outer joins return all rows from the right table, and all the rows from the left that have a match
- Full outer joins return all rows from both tables and all the rows from both tables that don't have a match

Scientifics computing libraries in Python

1. Scientifics Computing Libraries



Pandas

(Data structures & tools)

NumPy

(Arrays & matrices)

SciPy

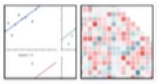
(Integrals, solving differential equations, optimization)

Visualization libraries in Python

2. Visualization Libraries

Matplotlib

(plots & graphs, most popular)



Seaborn

(plots: heat maps, time series, violin plots)

Algorithmic libraries in Python

3. Algorithmic Libraries

Scikit-learn

(Machine Learning : Regression, classification, and so on)

Statsmodels

(Explore data, estimate statistical models, and perform statistical tests)

Exporting to different formats in Python

Data Format	Read	Save
csv	pd.read_csv()	df.to_csv()
json	pd.read_json()	df.to_json()
Excel	pd.read_excel()	df.to_excel()
sql	pd.read_sql()	df.to_sql()

How to deal with missing data?

Check with the data collection source

Drop the missing values

- drop the variable
- drop the data entry

Replace the missing values

- replace it with an average (of similar datapoints)
- replace it by frequency
- replace it based on other functions

Leave it as missing data

Methods of normalizing data

Several approaches for normalization:

① $x_{new} = \frac{x_{old}}{x_{max}}$

Simple Feature scaling

② $x_{new} = \frac{x_{old} - x_{min}}{x_{max} - x_{min}}$

Min-Max

③ $x_{new} = \frac{x_{old} - \mu}{\sigma}$

Z-score

Simple Feature Scaling in Python

With Pandas:

length	width	height
168.8	64.1	48.8
168.8	64.1	48.8
180.0	65.5	52.4
...	...	...

length	width	height
0.81	64.1	48.8
0.81	64.1	48.8
0.87	65.5	52.4
...	...	...

```
df["length"] = df["length"]/df["length"].max()
```

Min-max in Python

With Pandas:

length	width	height
168.8	64.1	48.8
168.8	64.1	48.8
180.0	65.5	52.4
...	...	...

length	width	height
0.41	64.1	48.8
0.41	64.1	48.8
0.58	65.5	52.4
...	...	...

```
df["length"] = (df["length"]-df["length"].min())/
               (df["length"].max()-df["length"].min())
```

Z-score in Python

With Pandas:

length	width	height
168.8	64.1	48.8
168.8	64.1	48.8
180.0	65.5	52.4
...	...	...

length	width	height
-0.034	64.1	48.8
-0.034	64.1	48.8
0.039	65.5	52.4
...	...	...

```
df["length"] = (df["length"]-df["length"].mean())/df["length"].std()
```

Binning in Python Pandas

price
13495
16500
18920
41315
5151
6295
...

price	price-binned
13495	Low
16500	Low
18920	Medium
41315	High
5151	Low
6295	Low
...	...

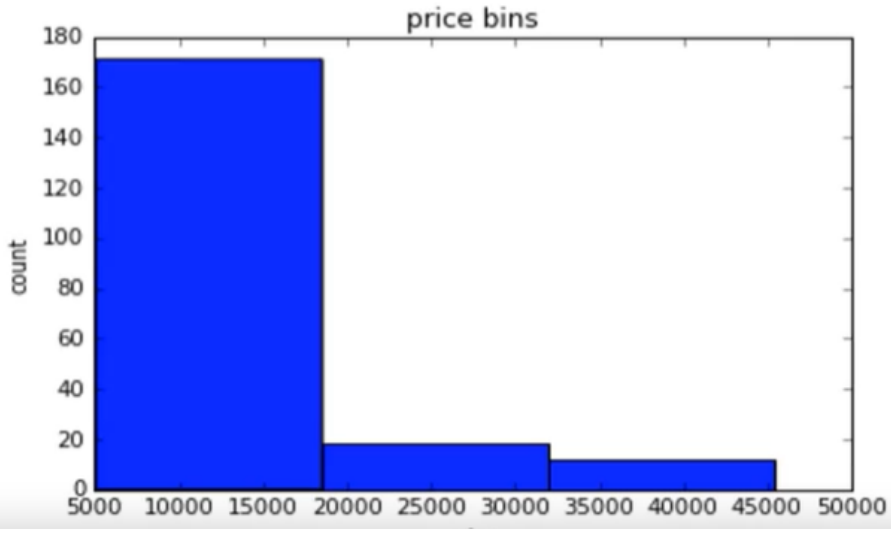
```
bins = np.linspace(min(df["price"]), max(df["price"]), 4)
```

```
group_names = [“Low”, “Medium”, “High”]
```

```
df[“price-binned”] = pd.cut(df[“price”], bins, labels=group_names, include_lowest=True )
```

Visualizing binned data

- Histograms



Dummy variables in Python pandas

- Use pandas.get_dummies() method.
- Convert categorical variables to dummy variables (0 or 1)

fuel
gas
diesel
gas
gas

gas	diesel
1	0
0	1
1	0
1	0

```
pd.get_dummies(df['fuel'])
```

Descriptive Statistics: Describe()

- Summarize statistics using pandas **describe()** method

```
df.describe()
```

	Unnamed: 0	symboling	normalized-losses	wheel-base	length	width	height	curb-weight	engine-size	bore	stroke
count	201.000000	201.000000	164.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000	201.000000
mean	100.000000	0.840796	122.000000	98.797015	174.200995	65.889055	53.766667	2555.666667	126.875622	3.319154	3.256766
std	58.167861	1.254802	35.442168	6.066366	12.322175	2.101471	2.447822	517.296727	41.546834	0.280130	0.316049
min	0.000000	-2.000000	65.000000	86.600000	141.100000	60.300000	47.800000	1488.000000	61.000000	2.540000	2.070000
25%	50.000000	0.000000	NaN	94.500000	166.800000	64.100000	52.000000	2169.000000	98.000000	3.150000	3.110000
50%	100.000000	1.000000	NaN	97.000000	173.200000	65.500000	54.100000	2414.000000	120.000000	3.310000	3.290000
75%	150.000000	2.000000	NaN	102.400000	183.500000	66.600000	55.500000	2926.000000	141.000000	3.580000	3.410000
max	200.000000	3.000000	256.000000	120.900000	208.100000	72.000000	59.800000	4066.000000	326.000000	3.940000	4.170000

Descriptive Statistics: Value_Counts()

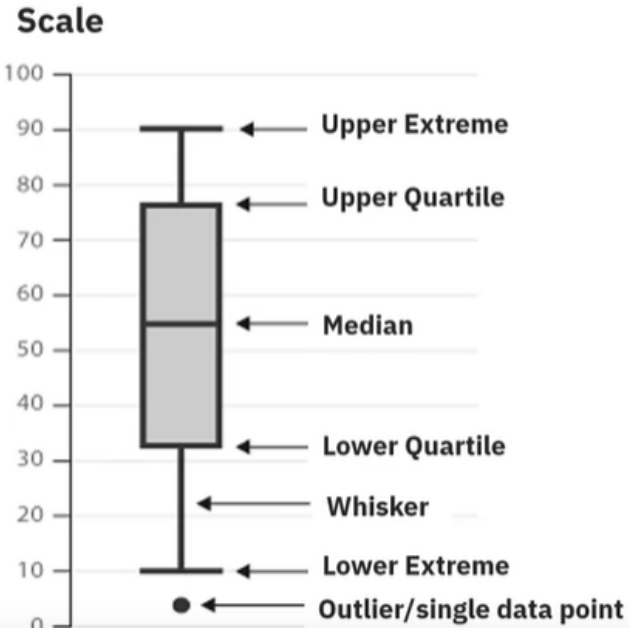
- Summarize the categorical data is by using the **value_counts()** method

```
drive_wheels_counts=df["drive-wheels"].value_counts()
```

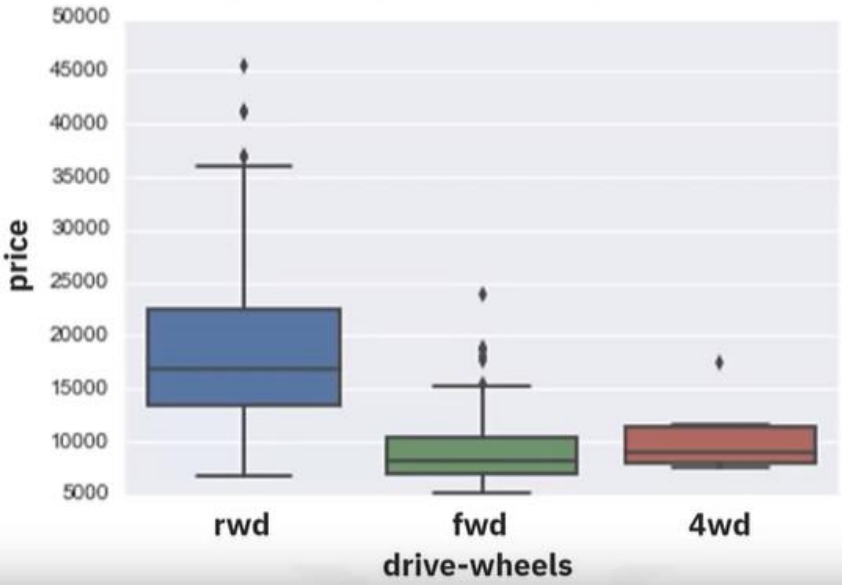
```
drive_wheels_counts.rename(colunms={'drive-wheels':'value_counts' inplace=True)
drive_wheels_counts.index.name= 'drive-wheels'
```

	value_counts
drive-wheels	
fwd	118
rwd	75
4wd	8

Descriptive Statistics: Box Plots




```
sns.boxplot(x= "drive-wheels", y= "price",data=df)
```



Descriptive Statistics: Scatter Plot

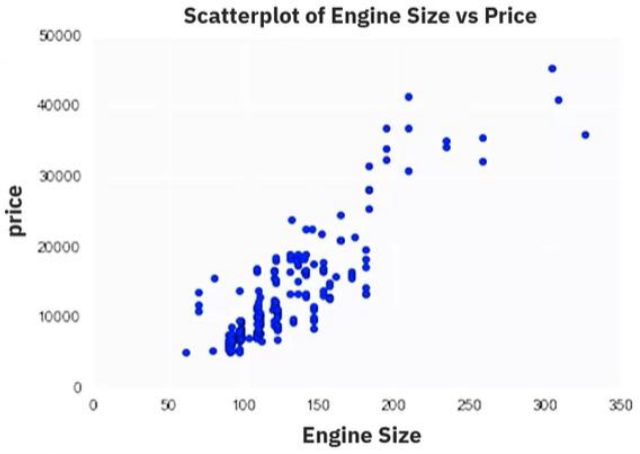
- Each observation represented as a point

- Scatter plots show the relationship between two variables:

    1. Predictor/independent variables on x-axis
    2. Target/dependent variables on y-axis

Scatterplot: Example

```
y=df["price"]
x=df["engine-size"]
plt.scatter(x,y)

plt.title("Scatterplot of Engine Size vs Price")
plt.xlabel("Engine Size")
plt.ylabel("Price")
```



Grouping data

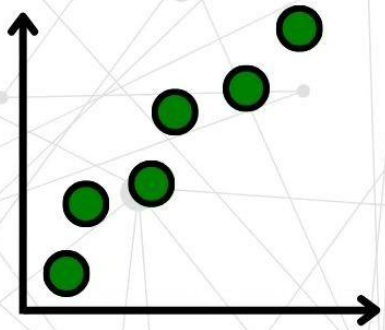
Question:

Is there any relationship between the different types of "drive system" and the "price" of the vehicles?

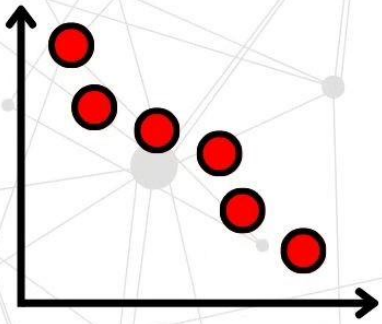
- Use Panda **dataframe.groupby()** method:
    - Can be applied to categorical variables
    - Group data into categories
    - Single or multiple variables

TYPES OF CORRELATION

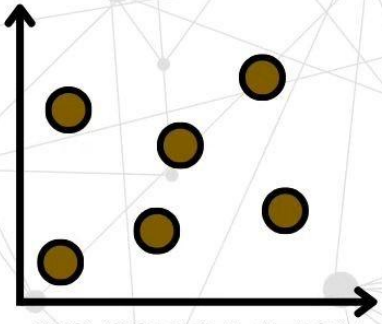
CORRELATION IS A STATISTICAL TECHNIQUE THAT CAN SHOW WHETHER AND HOW STRONGLY PAIRS OF VARIABLES ARE RELATED



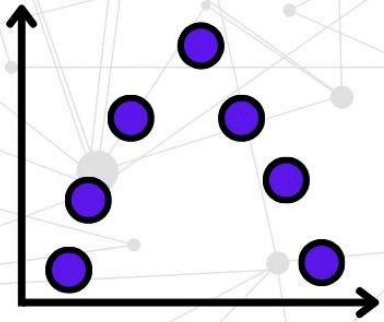
POSITIVE LINEAR CORRELATION



NEGATIVE LINEAR CORRELATION



NO CORRELATION



NON-LINEAR CORRELATION

groupby(): Example

```
df_test = df[['drive-wheels', 'body-style', 'price']]
df_grp = df_test.groupby(['drive-wheels', 'body-style'], as_index=False).mean()
df_grp
```

	drive-wheels	body-style	price
0	4wd	hatchback	7603.000000
1	4wd	sedan	12647.333333
2	4wd	wagon	9095.750000
3	fwd	convertible	11595.000000
4	fwd	hardtop	8249.000000
5	fwd	hatchback	8396.387755
6	fwd	sedan	9811.800000
7	fwd	wagon	9997.333333
8	rwd	convertible	23949.600000
9	rwd	hardtop	24202.714286
10	rwd	hatchback	14337.777778
11	rwd	sedan	21711.833333
12	rwd	wagon	16994.222222

Pandas method: Pivot()

- One variable is displayed along the columns, and the other variable is displayed along the rows

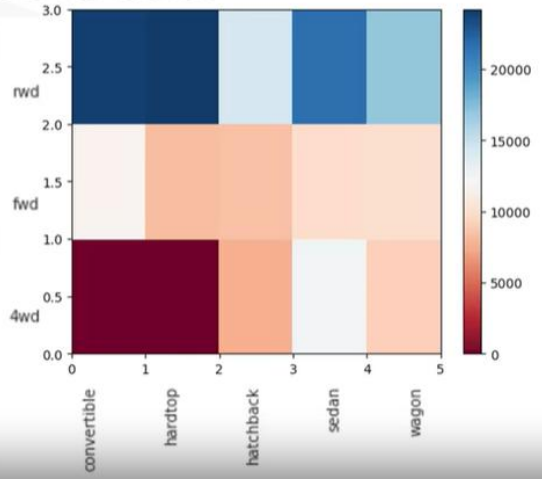
```
df_pivot = df_grp.pivot(index='drive-wheels', columns='body-style')
```

	price				
body-style	convertible	hardtop	hatchback	sedan	wagon
drive-wheels					
4wd	0.0	0.000000	7603.000000	12647.333333	9095.750000
fwd	11595.000000	8429.000000	8396.387755	9811.8000000	9997.333333
rwd	23949.600000	24202.714286	14337.777778	21711.833333	16994.222222

Heatmap

- Plot target variable against multiple variables

```
plt.pcolor(df_pivot, cmap='RdBu')
plt.colorbar()
plt.show()
```



Pearson Correlation

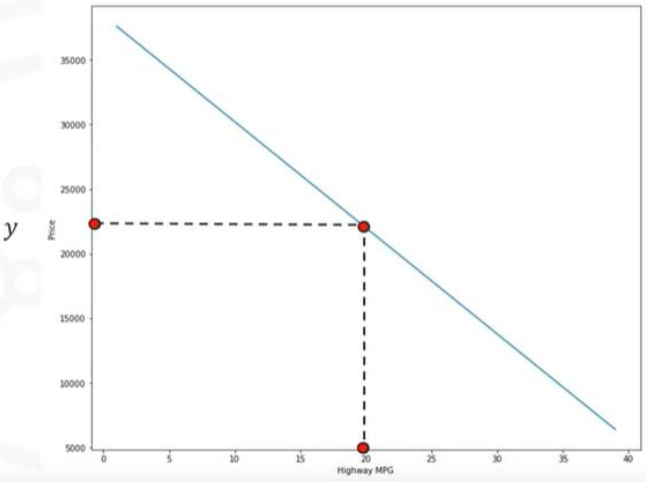
- Measure the strength of the correlation between two features
  - Correlation coefficient
  - P-value
- Correlation coefficient
  - Close to +1: Large Positive relationship
  - Close to -1: Large Negative relationship
  - Close to 0: No relationship
- Strong correlation
  - Correlation coefficient close to 1 or -1
  - P value less than 0.001
- P-value
  - P-value < 0.001 Strong certainty in the result
  - P-value < 0.05 Moderate certainty in the result
  - P-value < 0.1 Weak certainty in the result
  - P-value > 0. 1 No certainty in the result

Simple linear regression: Prediction

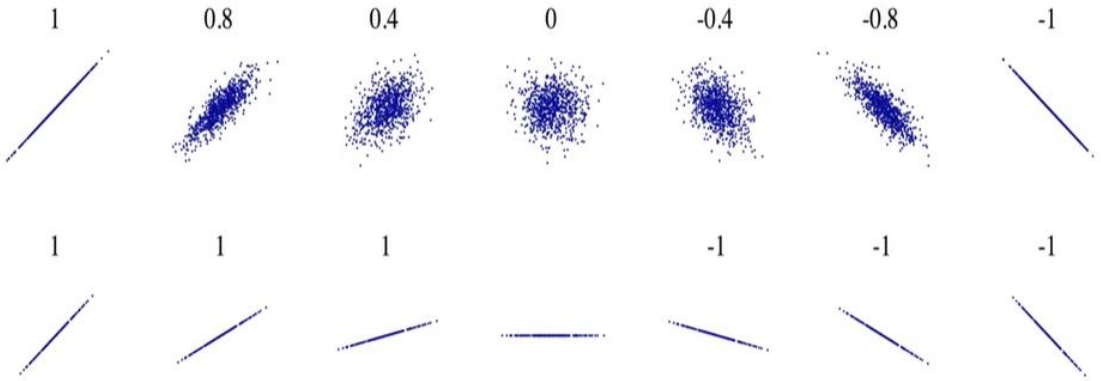
$$y = 38423 - 821x$$

$$= 38423 - 821\ (20)$$

$$= 22\ 003$$



Pearson Correlation

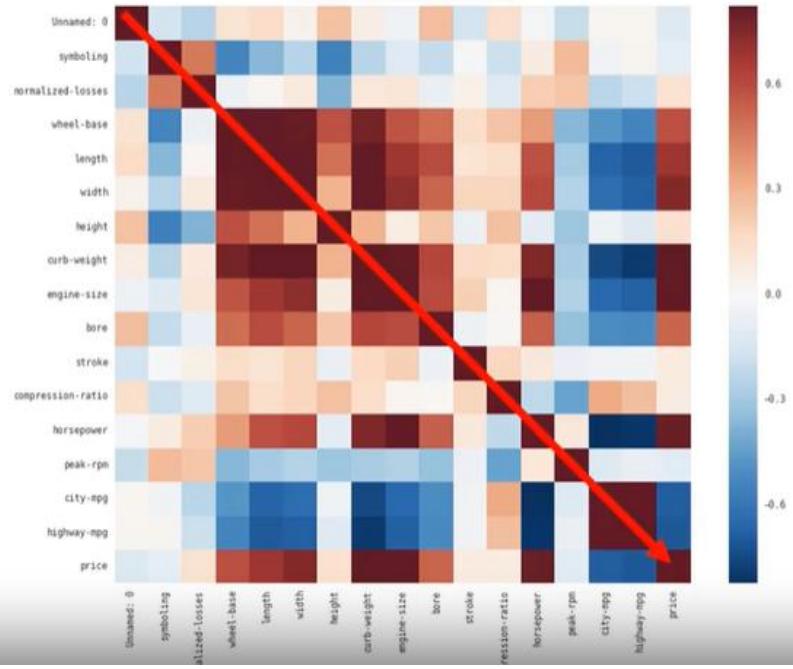


Pearson Correlation- Example

```
pearson_coef, p_value = stats.pearsonr(df['horsepower'], df[ 'price'])
```

- Pearson correlation: 0.81
- P-value : 9.35 e-48

Correlation- Heatmap

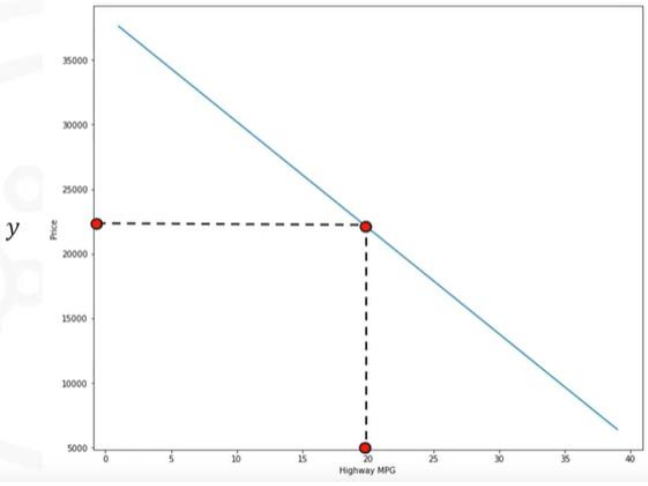


Simple linear regression: Prediction

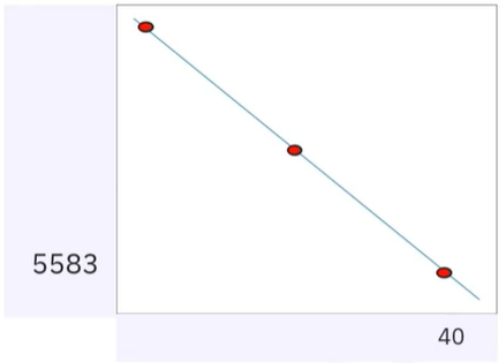
$$y = 38423 - 821x$$

$$= 38423 - 821\ (20)$$

$$= 22\ 003$$



Simple linear regression: Fit



$$X = \begin{bmatrix} 0 \\ 20 \end{bmatrix} \quad Y = \begin{bmatrix} 38423 \\ 22003 \end{bmatrix}$$

Fitting a simple linear model estimator

- X :Predictor variable
- Y:Target variable

1. Import linear_model from scikit-learn

```
from sklearn.linear_model import LinearRegression
```

1. Create a Linear Regression Object using the constructor:

```
lm=LinearRegression()
```

Fitting a simple linear model

- We define the predictor variable and target variable

```
X = df[['highway-mpg']]
Y = df['price']
```

- Then use lm.fit (X, Y) to fit the model , i.e find the parameters b_0 and b_1

```
lm.fit(X, Y)
```

- We can obtain a prediction

```
Yhat=lm.predict(X)
```

Yhat	X
2	5
:	
3	4

SLR - Estimated linear model

- We can view the intercept (b_0): `lm.intercept_`
  `38423.305858`
- We can also view the slope (b_1): `lm.coef_`
  `-821.73337832`
- The Relationship between Price and Highway MPG is given by:
- **Price = 38423.31 - 821.73 * highway-mpg**

$$\hat{Y} = b_0 + b_1 x$$

Distribution plots

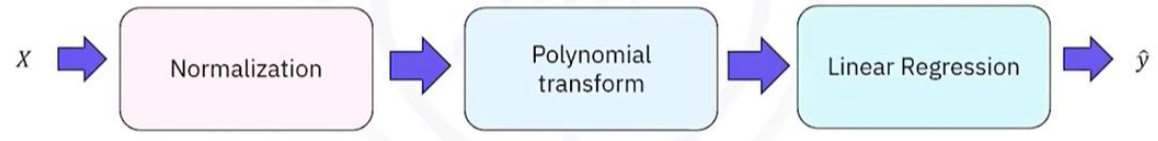
```
import seaborn as sns
```

```
ax1 = sns.distplot(df['price'], hist=False, color="r", label="Actual Value")

sns.distplot(Yhat, hist=False, color="b", label="Fitted Values" , ax=ax1)
```

Pipeline constructor

- We can train the pipeline object

```
Pipe.fit(df[['horsepower', 'curb-weight', 'engine-size', 'highway-mpg']],y)

yhat=Pipe.predict(X[['horsepower', 'curb-weight', 'engine-size', 'highway-mpg']])
```



Polynomial Regression

- Quadratic – 2nd order

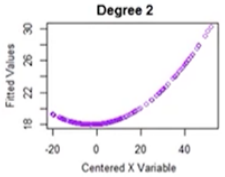
$$\hat{Y} = b_0 + b_1 x_1 + b_2 (x_1)^2$$

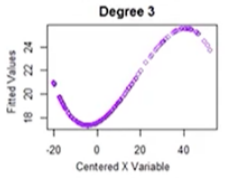
- Cubic – 3rd order

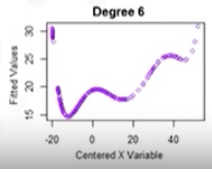
$$\hat{Y} = b_0 + b_1 x_1 + b_2 (x_1)^2 + b_3 (x_1)^3$$

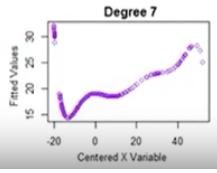
- Higher order

$$\hat{Y} = b_0 + b_1 x_1 + b_2 (x_1)^2 + b_3 (x_1)^3 + ..$$









Polynomial Regression

1. Calculate Polynomial of 3rd order

```
f=np.polyfit(x,y,3)

p=np.poly1d(f)
```

2. We can print out the model

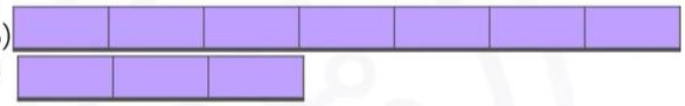
```
print (p)
```

$$-1.557(x_1)^3 + 204.8(x_1)^2 + 8965\,x_1 + 1.37\text{x}10^5$$

Training/Testing sets

Data:

- Split dataset into:
  - training set (70%)
  - testing set (30%)



- Build and train the model with a training set
- Use testing set to assess the performance of a predictive model
- When we have completed testing our model we should use all the data to train the model to get the best performance

Function training_test_split()

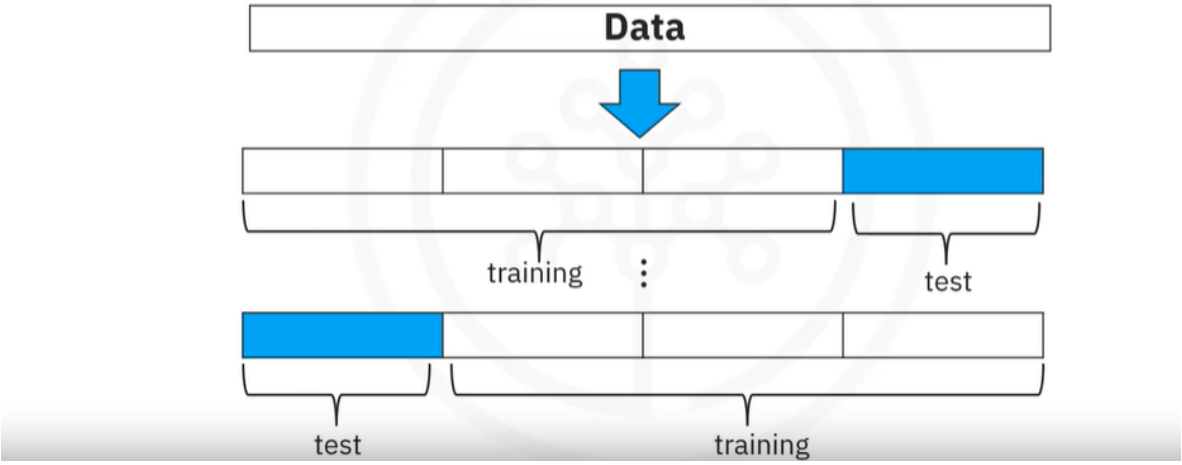
- Split data into random train and test subsets

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x_data, y_data, test_size=0.3, random_state=0)
```

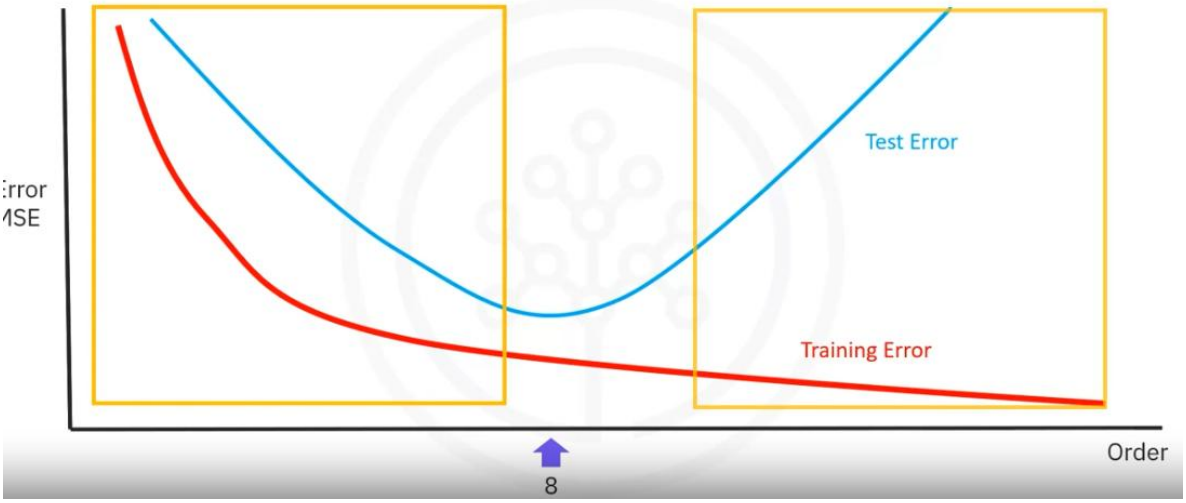
- **x_data**: features or independent variables
- **y_data**: dataset target: df['price']
- **x_train, y_train**: parts of available data as training set
- **x_test, y_test**: parts of available data as testing set
- **test_size**: percentage of the data for testing (here 30%)
- **random_state**: number generator used for random sampling

Cross validation

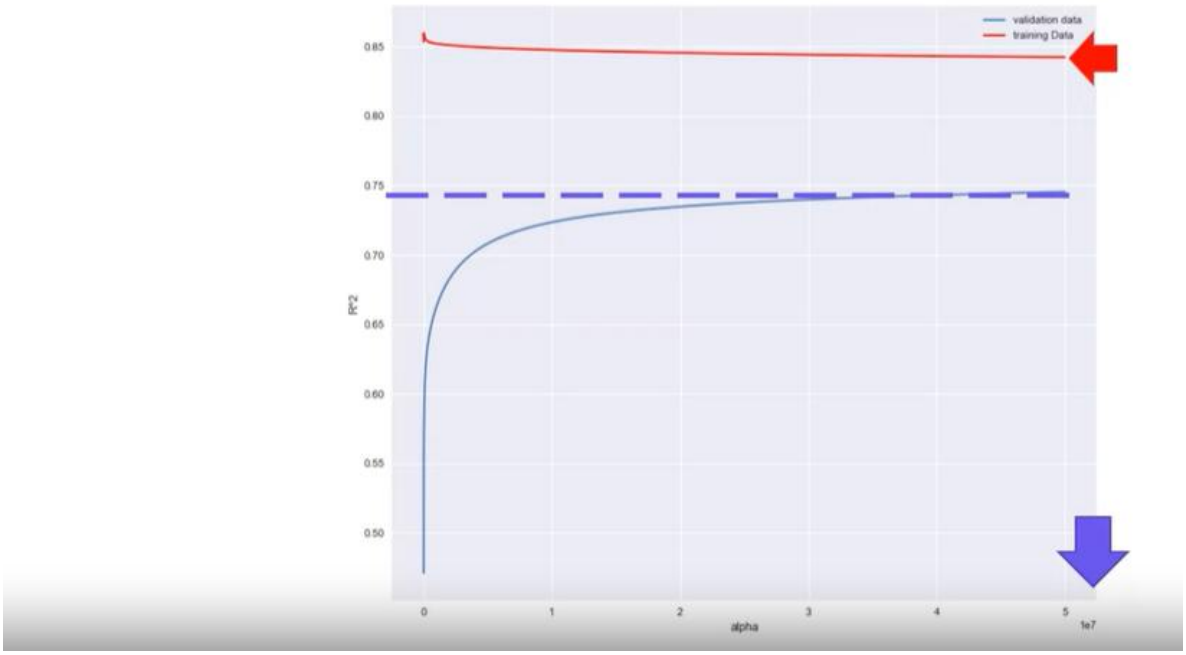
- Most common out-of-sample evaluation metrics
- More effective use of data (each observation is used for both training and testing)



Model Selection



Ridge Regression



Ridge Regression

$$\hat{y} = 1 + 2\,x - 3\,x^2 - 2x^3 - 12\,x^4 - 40x^5 + 80x^6 + 71x^7 - 141x^8 - 38x^9 + 75x^{10}$$

Alpha	x	x^2	x^3	x^4	x^5	x^6	x^7	x^8	x^9	x^{10}
0	2	-3	-2	-12	-40	80	71	-141	-38	75
0.001	2	-3	-7	5	4	-6	4	-4	4	6
0.01	1	-2	-5	-0.04	0.15	-1	1	-0.5	0.3	1
1	0.5	-1	-1	-0.614	0.70	-0.38	-0. 56	-0.21	-0.5	-0.1
10	0	-0.5	-0.3	-0.37	-0.30	-0.30	-0.22	-0.22	-0.22	-0.17

Grid Search

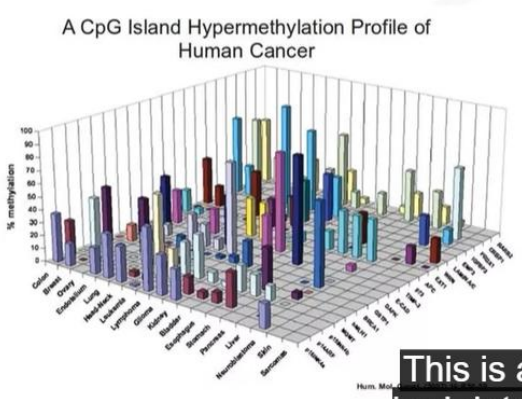
```
from sklearn.linear_model import Ridge
from sklearn.model_selection import GridSearchCV

parameters1= [{'alpha': [0.001,0.1,1, 10, 100, 1000,10000,100000,100000]}]

RR=Ridge()

Grid1 = GridSearchCV(RR, parameters1,cv=4)

Grid1.fit(x_data[['horsepower', 'curb-weight', 'engine-size', 'highway-
mpg']],y_data)

Grid1.best_estimator_

scores  = Grid1.cv_results_
```

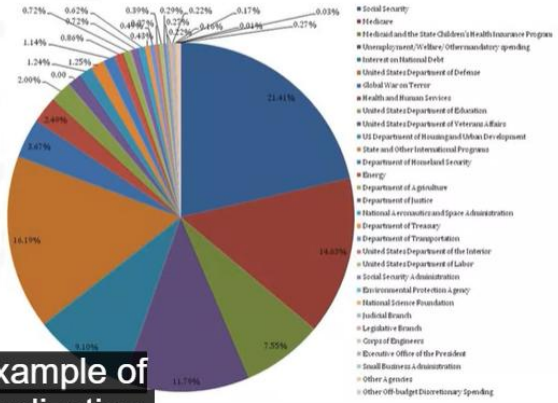
Grid Search

```
for param,mean_val, mean_test in zip(scores['params'], scores['mean_test_score'],
scores['mean_train_score']):

print(param, "R^2 on test data:", mean_val,"R^2 on train data:", mean_test)
```

```
{'alpha': 0.001, 'normalize': True} R^2 on tesst data: 0.66605547293 R^2 on train data: 0.814001968709
{'alpha': 0.001, 'normalize': False} R^2 on tesst data: 0.665488366584 R^2 on train data: 0.814002698797
{'alpha': 0.1, 'normalize': True} R^2 on tesst data: 0.694175625356 R^2 on train data: 0.810546768311
{'alpha': 0.1, 'normalize': False} R^2 on tesst data: 0.665488937796 R^2 on train data: 0.814002698794
{'alpha': 1, 'normalize': True} R^2 on tesst data: 0.690486934584 R^2 on train data: 0.749104440368
{'alpha': 1, 'normalize': False} R^2 on tesst data: 0.665494127178 R^2 on train data: 0.814002698472
{'alpha': 10, 'normalize': True} R^2 on tesst data: 0.321376875232 R^2 on train data: 0.341856042902
{'alpha': 10, 'normalize': False} R^2 on tesst data: 0.665545680812 R^2 on train data: 0.8140026666
{'alpha': 100, 'normalize': True} R^2 on tesst data: 0.0170551710263 R^2 on train data: 0.0496044796826
{'alpha': 100, 'normalize': False} R^2 on tesst data: 0.666029359996 R^2 on train data: 0.813999791851
{'alpha': 1000, 'normalize': True} R^2 on tesst data: -0.0301961745066 R^2 on train data: 0.005184451599
{'alpha': 1000, 'normalize': False} R^2 on tesst data: 0.668968215369 R^2 on train data: 0.813870488264
{'alpha': 10000, 'normalize': True} R^2 on tesst data: -0.0351687400461 R^2 on train data: 0.000520784757979
{'alpha': 10000, 'normalize': False} R^2 on tesst data: 0.673346359342 R^2 on train data: 0.812583743226
{'alpha': 100000, 'normalize': True} R^2 on tesst data: -0.0356685844558 R^2 on train data: 5.2101975528e-05
{'alpha': 100000, 'normalize': False} R^2 on tesst data: 0.657818838432 R^2 on train data: 0.789541446486
{'alpha': 100000, 'normalize': True} R^2 on tesst data: -0.0356685844558 R^2 on train data: 5.2101975528e-05
{'alpha': 100000, 'normalize': False} R^2 on tesst data: 0.657818838432 R^2 on train data: 0.789541446486
```

Best practices (bad example)



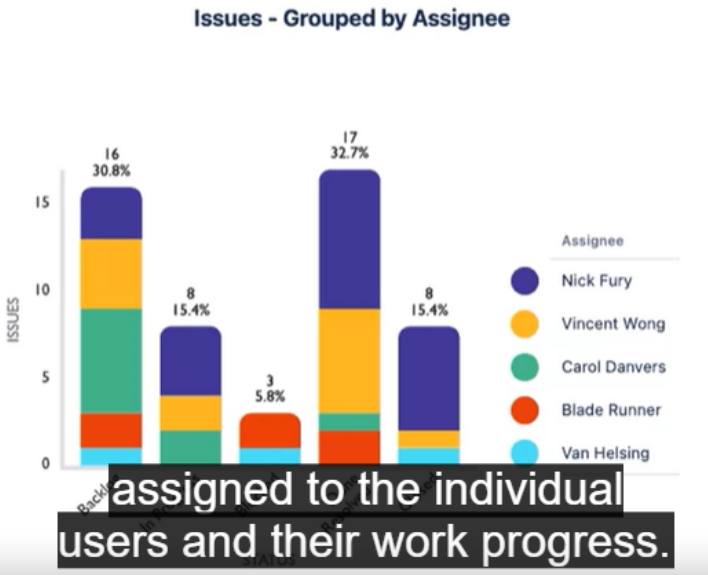


This is an example of bad data visualization.

SOURCE: https://www.livestories.com/blog/five-ways-to-fail-data-visualization

SOURCE: https://en.wikipedia.org/wiki/2007_United_States_federal_budget#.

Best practices (good example)



assigned to the individual users and their work progress.

SOURCE: https://www.oldstreetsolutions.com/good-and-bad-data-visualization

DARKHORSE ANALYTICS

When creating a visual, always remember:

- Less is more effective
- Less is more attractive
- Less is more impactful

Popular plot libraries

Plot libraries in Python:

- Matplotlib
- Pandas
- Seaborn
- Folium
- Plotly
- PyWaffle

Recap

In this video, you learned that:

- Matplotlib is a plotting library that offers a wide range of plotting capabilities.
- Pandas is a plotting library that provides Integrated plotting functionalities for data analysis.
- Seaborn is a specialized library for statistical visualizations, offering attractive default aesthetics and color palettes.
- Folium is a Python library that allows you to create interactive and customizable maps.
- Plotly is an interactive and dynamic library for data visualization that supports a wide range of plot types and interactive features.
- PyWaffle enables you to visualize proportional representation using squares or rectangles.

Backend layer

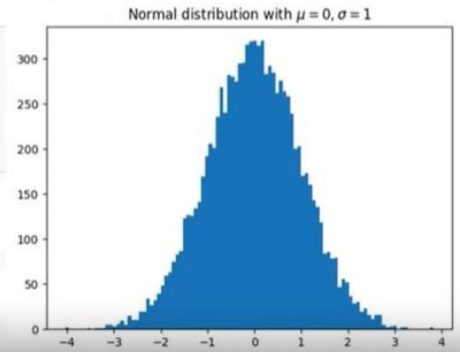
Has three built-in abstract interface classes:

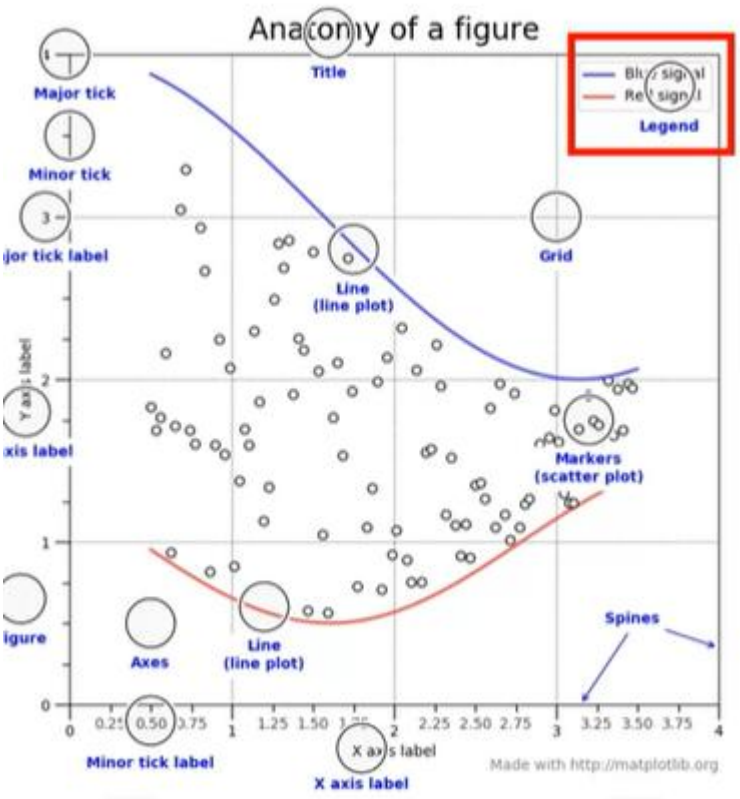
1. FigureCanvas: **matplotlib.backend_bases.FigureCanvas**
   - Encompasses the area onto which the figure is drawn
2. Renderer: **matplotlib.backend_bases.Renderer**
   - Knows how to draw on the FigureCanvas
3. Event: **matplotlib.backend_bases.Event**
   - Handles user inputs such as keyboard strokes and mouse clicks

Artist layer

- Comprised of one main object – **Artist**:
  - Knows how to use the Renderer to draw on the canvas.
- Title, lines, tick labels, and images, all correspond to individual **Artist** instances.
- Two types of **Artist** objects:
  1. **Primitive**: Line2D, Rectangle, Circle, and Text.
  2. **Composite**: Axis, Tick, Axes, and Figure.
- Each *composite* artist may contain other *composite* artists as well as *primitive* artists.

```
import matplotlib.pyplot as plt
import numpy as np

x = np.random.randn(10000)
plt.hist(x, 100)
plt.title(r'Normal distribution with $\mu=0, \sigma=1$')
plt.savefig('matplotlib_histogram.png')
plt.show()
```

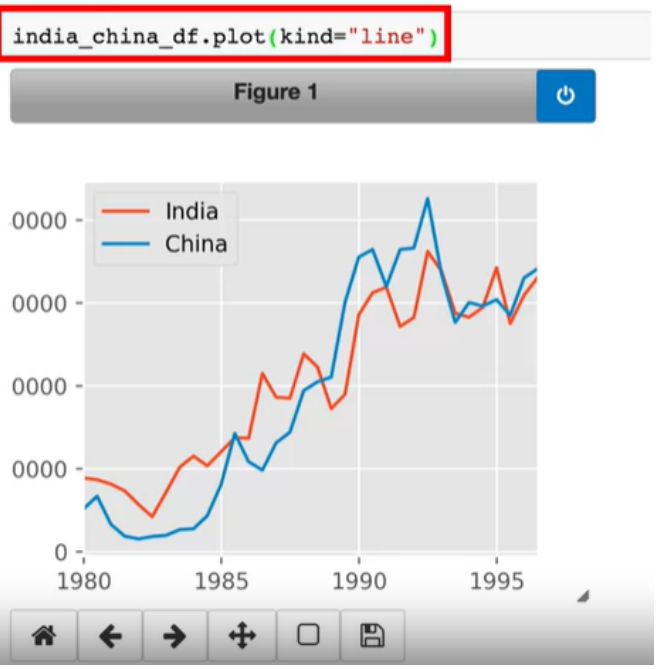




india_china_df

	India	China
1980	8880	5123
1981	8670	6682
1982	8147	3308
1983	7338	1863
1984	5704	1527

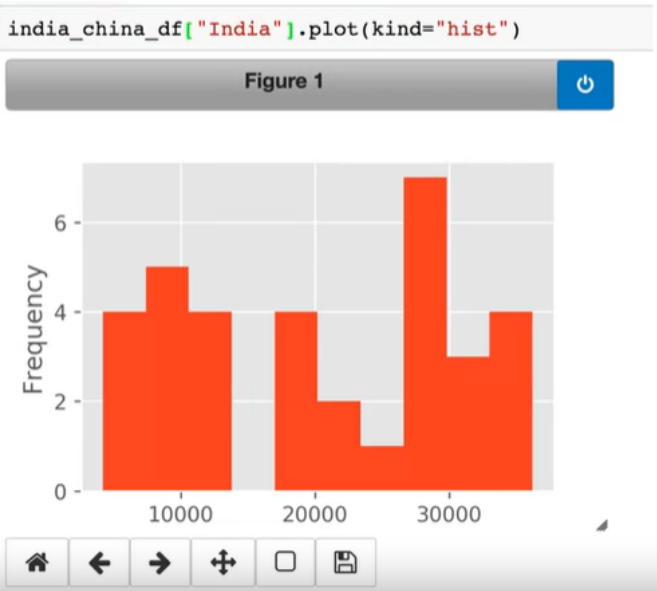
```
india_china_df.plot(kind="line")
```



india_china_df

	India	China
1980	8880	5123
1981	8670	6682
1982	8147	3308
1983	7338	1863
1984	5704	1527

```
india_china_df["India"].plot(kind="hist")
```



Generating area plots

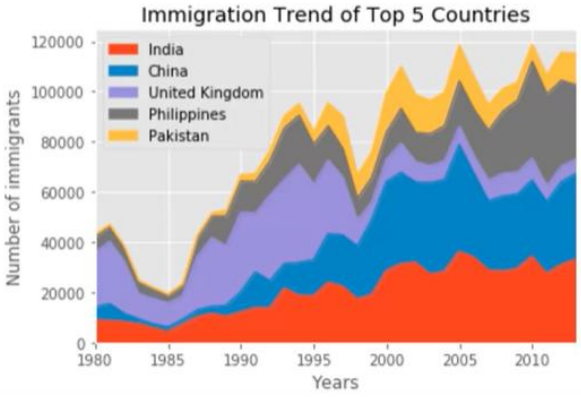
```
years = list(map(str, range(1980, 2014)))

df_canada.sort_values(['Total'], ascending = False, axis = 0, inplace = True)

df_top5 = df_canada.head()
df_top5 = df_top5[years].transpose()
```

Country	India	China	United Kingdom of Great Britain and Northern Ireland	Philippines	Pakistan
1980	8880	5123	22045	6051	978
1981	8670	6682	24796	5921	972
1982	8147	3308	20620	5249	1201
1983	7338	1863	10015	4562	900
1984	5704	1527	10170	3801	668

Area plots

```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

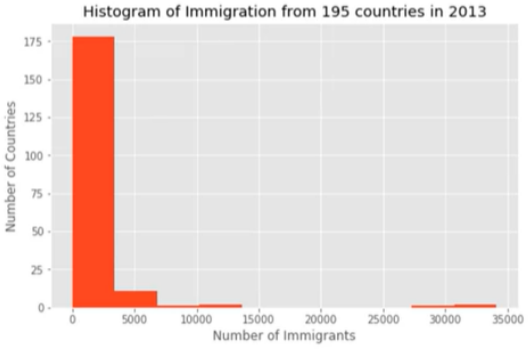
```
df_top5.plot(kind='area')

plt.title('Immigration trend of top 5 countries')
plt.ylabel('Number of immigrants')
plt.xlabel('Years')

plt.show()
```



Generating a histogram

```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

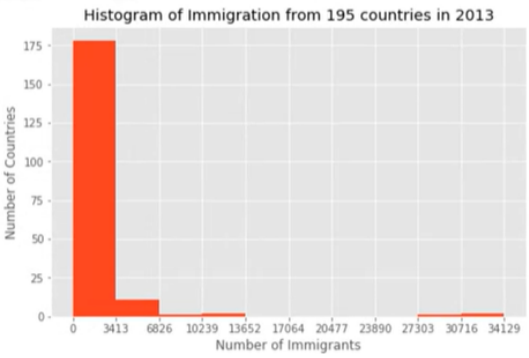
```
df_canada['2013'].plot(kind='hist')

plt.title('Histogram of Immigration from 195 countries in 2013')
plt.ylabel('Number of Countries')
plt.xlabel('Number of Immigrants')

plt.show()
```



Generating a histogram

```
import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np
```

```
count, bin_edges = np.histogram(df_canada['2013'])

df_canada['2013'].plot(kind='hist', xticks = bin_edges)

plt.title('Histogram of Immigration from 195 countries in 2013')
plt.ylabel('Number of Countries')
plt.xlabel('Number of Immigrants')

plt.show()
```



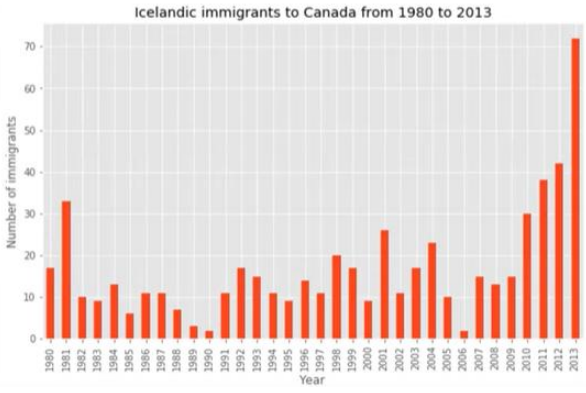
- You can use the Numpy library to create bins for the histogram representation.

Generating a bar chart

```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

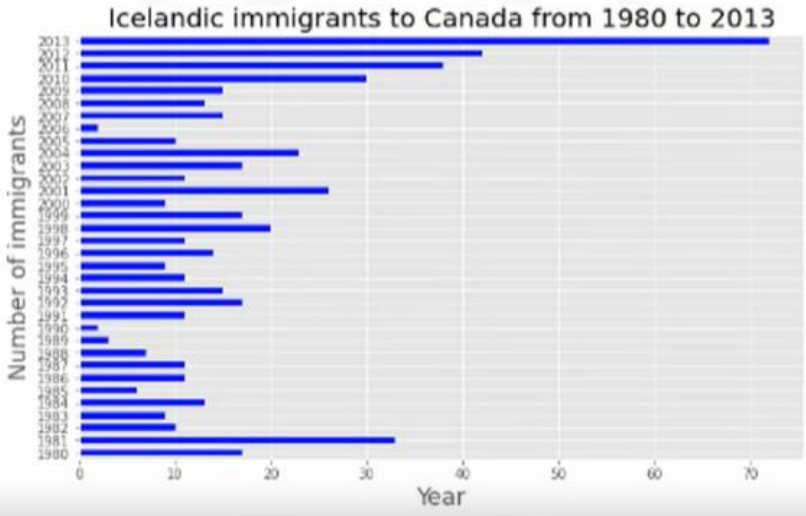
```
Years = list (map(str, range (1980, 2014)))
df_iceland = df_canada.loc[ 'Iceland',
years]
```

```
df_iceland.plot(kind='bar')

plt.title('Icelandic immigrants to
Canada from 1980 to 2013')
plt.xlabel('Year')
plt.ylabel('Number of immigrants')

plt.show()
```



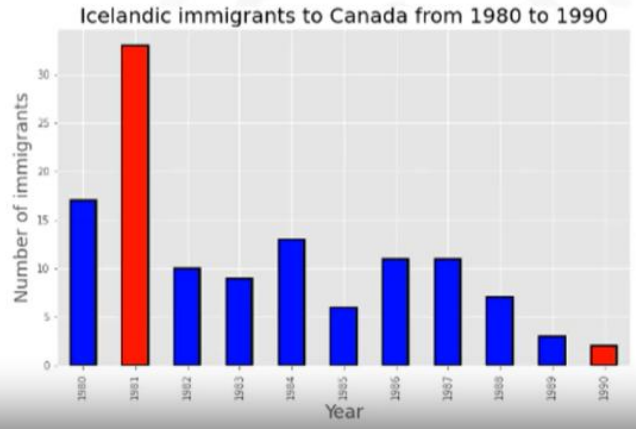
Horizontal bar chart

```
df_iceland.plot(kind='barh', color ='red')
```



Highlighted bar and edge color

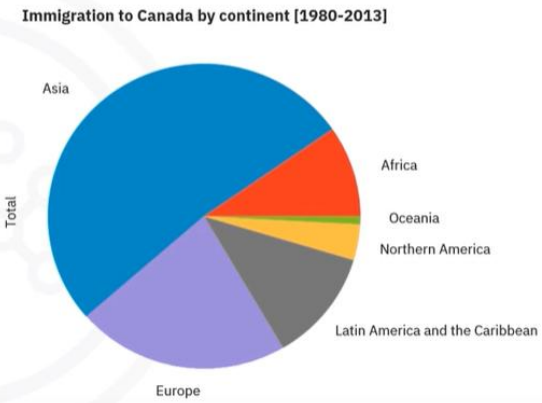
```
c=['blue','red',"blue','blue','blue','blue','blue','blue','blue','blue','red']
df_Iceland.plot(kind = 'bar', color = c, edgecolor = 'black')
```



Pie chart

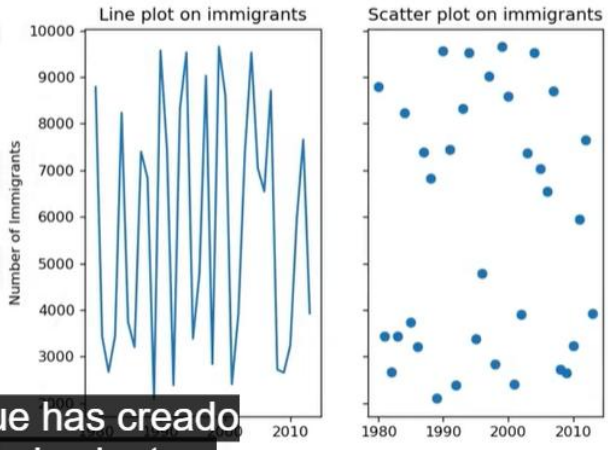
```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

```
df_continents['Total'].plot(kind='pie')

plt.title('Immigration to Canada by
Continent [1980-2013]')

plt.show()
```



Multiple plots and sub-plotting

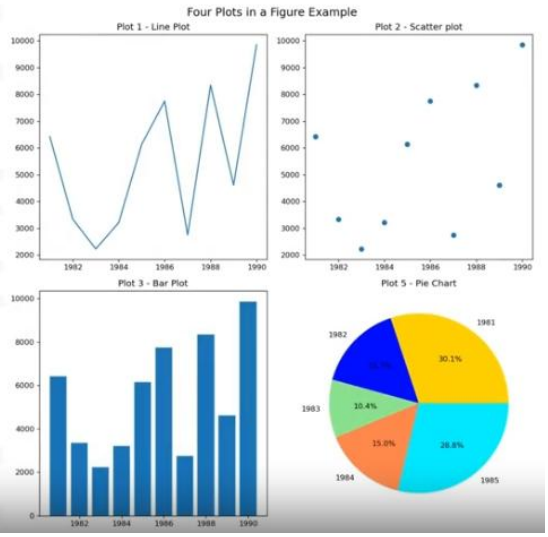
```
# Create a figure with two axes in a row
fig, axs = plt.subplots(1, 2, sharey=True)

#Plotting in first axes - the left one
axs[0].plot(years, immigrants)
axs[0].set_title("Line plot on immigrants")

#Plotting in second axes - the right one
axs[1].scatter(years, immigrants)
axs[1].set_title("Scatter plot on immigrants")
```



Y mira que has creado dos parcelas juntas.

Multiple plots and sub-plotting

```
# Add the second subplot (top-right)
ax2 = fig.add_subplot(2, 2, 2)
ax2.scatter(years, immigrants)
ax2.set_title('Plot 2 - Scatter plot')

# Add the third subplot (bottom-left)
ax3 = fig.add_subplot(2, 2, 3)
ax3.bar(years, immigrants)
ax3.set_title('Plot 3 - Bar Plot')

# Add the fourth subplot (bottom-right)
ax4 = fig.add_subplot(2, 2, 4)
ax4.pie(immigrants[0:5], labels=years[0:5],
        colors = ['gold','blue','lightgreen','coral','cyan'],
        autopct='%1.1f%%')
ax4.set_aspect('equal')
ax4.set_title('Plot 5 - Pie Chart')
```



Data storytelling and data visualization

- Data storytelling
  - Is the art of storytelling
  - Creates a narrative around the data
- Data visualization
  - Is an important aspect of data storytelling
  - Creates engaging visuals

Article link:

https://www.forbes.com/sites/brentdykes/2016/03/31/data-storytelling-the-essential-data-science-skill-everyone-needs/?sh=5aca60cd52ad

What is Folium?

- Folium is a powerful data visualization library in Python that was built primarily to help people visualize geospatial data.
- With Folium, you can create a map of any location in the world using latitude and longitude values. You can also create a map and superimpose markers and clusters on top of the map for interesting visualizations.
- You can also create maps of different styles, such as street-level maps, stamen maps, and a couple of others.

```
#import library
import folium
```

Creating a map of Canada

```
# define the world map centered around
# Canada with a low zoom level
world_map = folium.Map(
    location=[56.130, -106.35],
    zoom_start=4
)

# display world map
world_map
```

Map styles: Stamen Toner

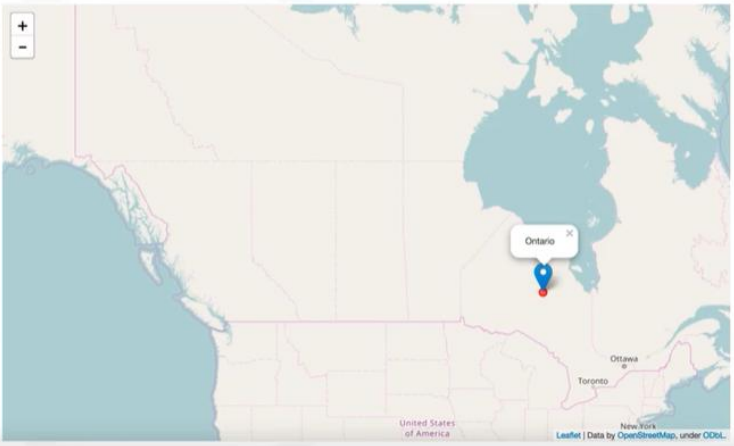
```
# create a Stamen Toner map of
# the world centered around Canada
world_map = folium.Map(
    location=[56.130, -106.35],
    zoom_start=4,
    tiles='Stamen Toner'
)

# display map
world_map
```

Map styles: Stamen Terrain

```
# create a Stamen Toner map of
# the world centered around Canada
world_map = folium.Map(
    location=[56.130, -106.35],
    zoom_start=4,
    tiles='Stamen Terrain'
)

# display map
world_map
```

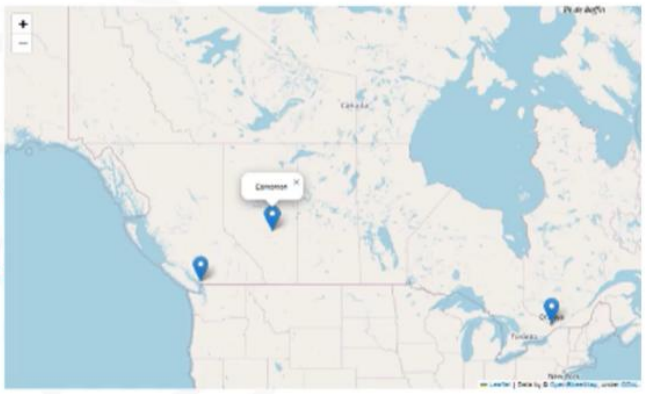
Label the marker

```
# generate map of Canada
canada_map = folium.Map(
    location=[56.130, -106.35],
    zoom_start=4
)

## add a red marker to Ontario

# create a feature group
ontario = folium.map.FeatureGroup()

# style the feature group
ontario.add_child(
    folium.features.CircleMarker(
    [51.25, -85.32], radius = 5,
    color = "red", fill_color = "Red"
    )
)

# add the feature group to the map
canada_map.add_child(ontario)

# label the marker
folium.Marker([51.25, -85.32],
    popup='Ontario').add_to(canada_map)

# display map
canada_map
```



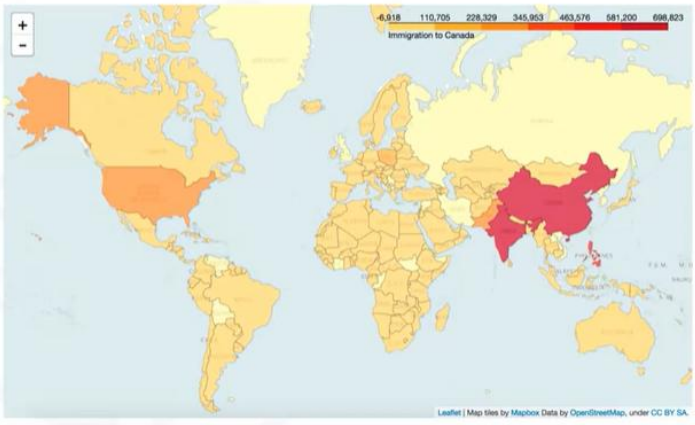
Multiple markers

```
# Define a list of locations and their corresponding popups
locations = [
    {"location": [45.4215, -75.6989], "popup": "Ottawa"},
    {"location": [53.5461, -113.4938], "popup": "Edmonton"},
    {"location": [49.2827, -123.1207], "popup": "Vancouver"},
    # Add more locations and their popups here
    ]

# Add markers for each location in the list
for loc in locations:
    folium.Marker(location=loc["location"],
                  popup=loc["popup"]).add_to(map)

# Display the map with the markers
map
```



```
#import MarkerCluster
from folium.plugins import MarkerCluster

# Create a MarkerCluster object
marker_cluster = MarkerCluster().add_to(map)

# Add markers for each location in the list to the MarkerCluster
for loc in locations:
    folium.Marker(location=loc["location"],
                  popup=loc["popup"]).add_to(marker_cluster)
```

```
# create a plain world map
world_map = folium.Map(
    zoom_start=2,
    tiles='Mapbox Bright'
)

## geojson file
world_geo = r'world_countries.json'

# generate choropleth map using the total
# population of each country to Canada from
# 1980 to 2013
world_map.choropleth(
    geo_path=world_geo,
    data=df_canada,
    columns=['Country', 'Total'],
    key_on='feature.properties.name',
    fill_color='YlOrRd',
    legend_name='Immigration to Canada'
)

# display map
world_map
```



Supervised vs unsupervised learning

Supervised Learning

- **Classification:** Classifies labeled data
- **Regression:** Predicts trends using previous labeled data
- Has more evaluation methods than unsupervised learning
- Controlled environment

Unsupervised Learning

- **Clustering:** Finds patterns and groupings from unlabeled data
- Has fewer evaluation methods than supervised learning
- Less controlled environment

Regression algorithms

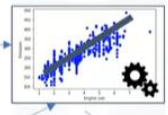
- Ordinal regression
- Poisson regression
- Fast forest quantile regression
- Linear, Polynomial, Lasso, Stepwise, Ridge regression
- Bayesian linear regression
- Neural network regression
- Decision forest regression
- Boosted decision tree regression
- KNN (K-nearest neighbors)

Train/Test split evaluation approach

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	212

Train

Test



	Prediction
6	234
7	256
8	267
9	210

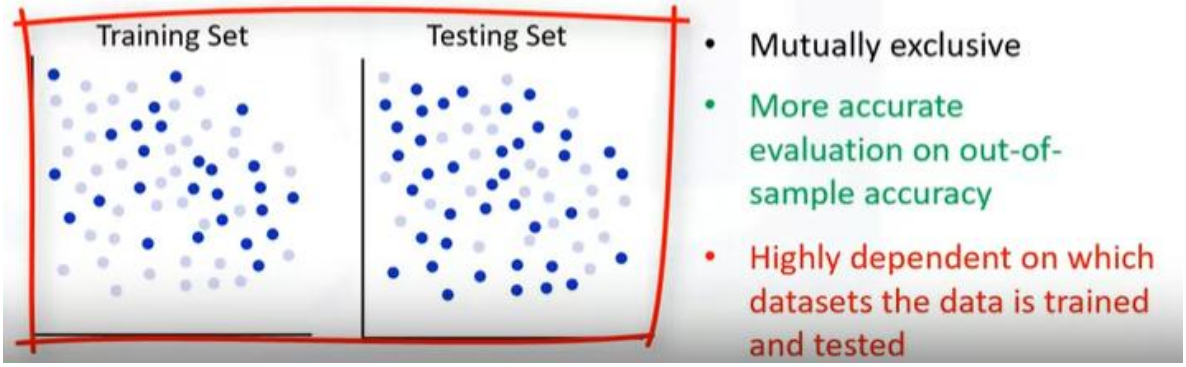
Actual values

Predicted values

Train/Test split evaluation approach

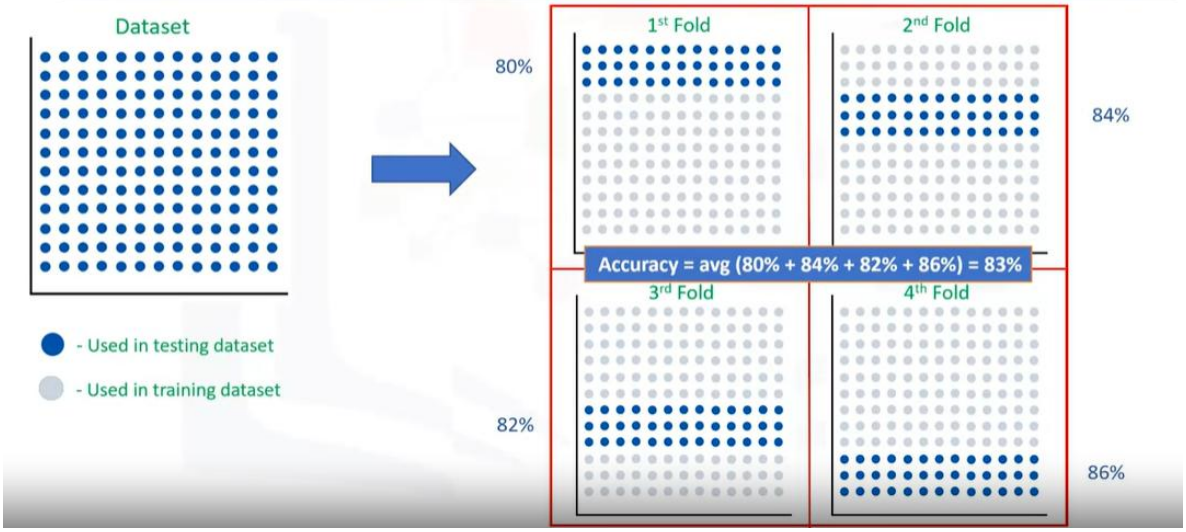
- Test-set is a portion of the train-set
- High "training accuracy"
- Low "out-of-sample accuracy"

- Mutually exclusive
- More accurate evaluation on out-of-sample accuracy

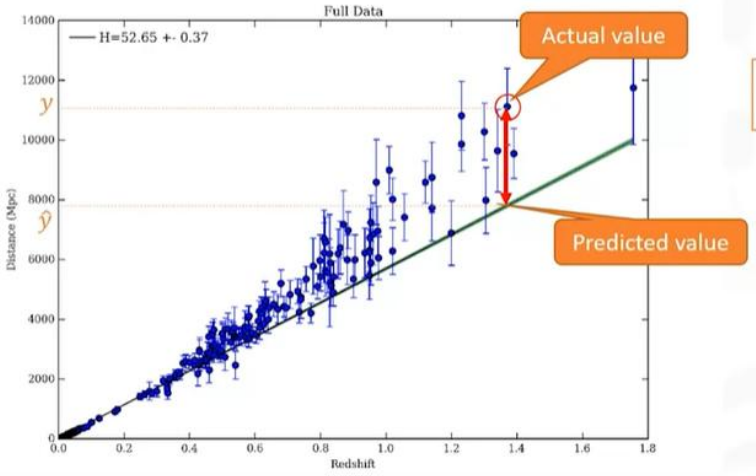


- Mutually exclusive
- More accurate evaluation on out-of-sample accuracy
- Highly dependent on which datasets the data is trained and tested

How to use K-fold cross-validation?



What is an error of the model?



Error: measure of how far the data is from the fitted regression line.

$$MAE = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

$$RAE = \frac{\sum_{j=1}^{n}|y_j - \hat{y}_j|}{\sum_{j=1}^{n}|y_j - \bar{y}|}$$

$$RSE = \frac{\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}{\sum_{j=1}^{n}(y_j - \bar{y})^2}$$

$$R^2 = 1 - RSE$$

RAE = Σ|actual - predicted| / Σ|actual - mean|

It's important to note the distinction between Relative Absolute Error (RAE) and Residual Sum of Squares (RSS):

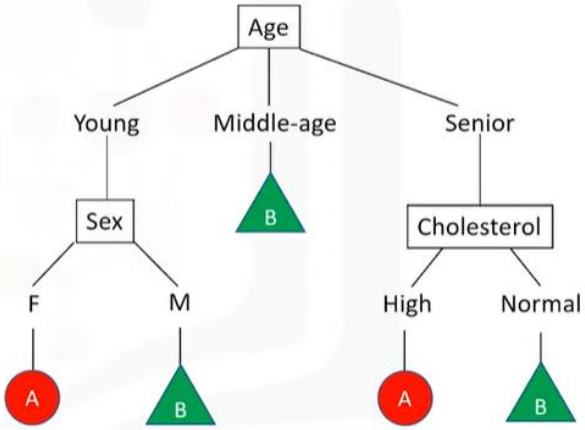
1. Relative Absolute Error (RAE): Measures the average absolute difference between actual and predicted values relative to the average absolute difference between actual values and their mean.
2. Residual Sum of Squares (RSS): Calculates the sum of the squared differences between actual and predicted values.

The formula for RSS:

RSS = Σ(actual - predicted)^2

Decision tree learning algorithm

1. Choose an attribute from your dataset.
2. Calculate the significance of attribute in splitting of data.
3. Split data based on the value of the best attribute.
4. Go to step 1.

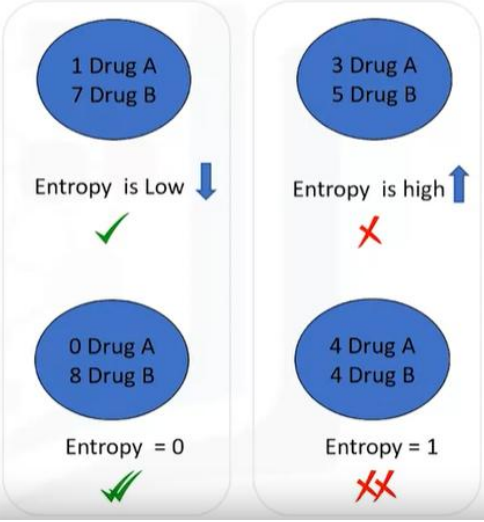


Entropy

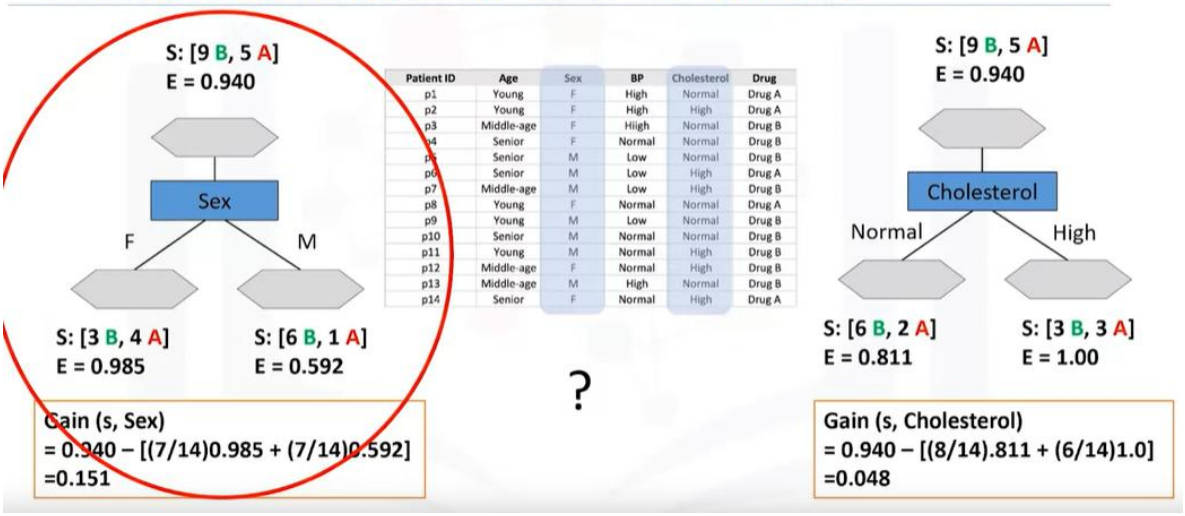
- Measure of randomness or uncertainty

$$\text{Entropy} = - p(A)\log_2(p(A)) - p(B)\log_2(p(B))$$

The lower the Entropy, the less uniform the distribution, the purer the node.



Which attribute is the best?

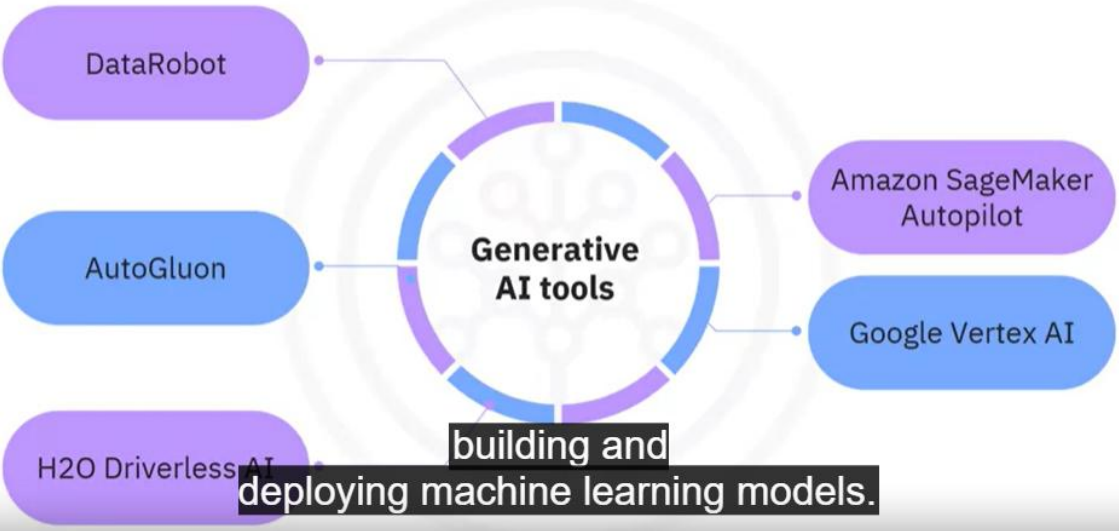


Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	Hiigh	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A

SVM applications

- Image recognition
- Text category assignment
- Detecting spam
- Sentiment analysis
- Gene Expression Classification
- Regression, outlier detection and clustering

Generative AI tools



Questions to ask: history

Ask questions about the interviewee's history, such as:

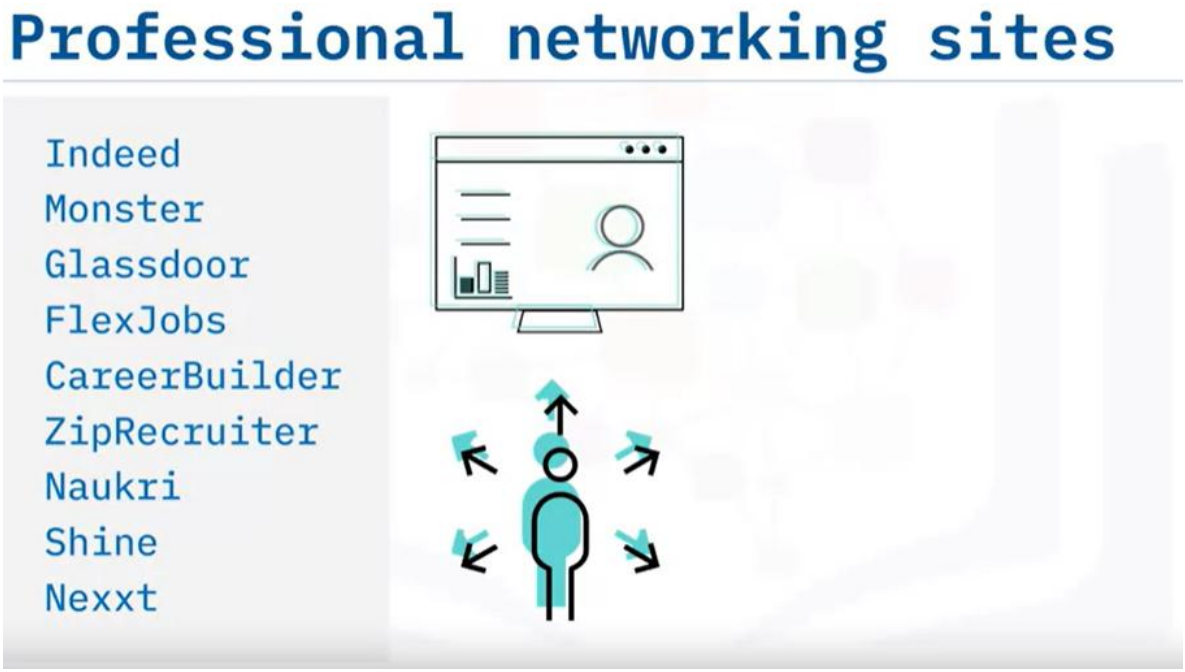
- How did you become interested in the data scientist role?
- How did you prepare to enter the field?
- How did you begin your career?

Questions to ask: current role

- What is a typical workday like for you?
- What do you like most about your work?
- What is one of your favorite projects that you have worked on?
- What is the most difficult challenge you have dealt with?
- What would you like to accomplish next in your career?

Questions to ask: advice

- What related fields or career paths would you recommend I also look into?
- How do you stay current and learn new technology?
- What current issues and trends in the field should I be aware of?
- What are the most effective strategies for seeking a position in this field?



Aquí tienes una lista de preguntas estratégicas que puedes hacer al entrevistador para evaluar si la empresa y el rol son adecuados para ti. Estas preguntas demuestran interés y te ayudan a obtener información clave:

Sobre el rol

1. **¿Cuáles son las responsabilidades principales de este puesto?**
2. **¿Cómo se mide el éxito en este rol?**
3. **¿Qué desafíos enfrentan actualmente en el equipo o el departamento que podría ayudar a resolver en este puesto?**
4. **¿Cómo se ve un día típico en este trabajo?**
5. **¿Qué habilidades considera esenciales para tener éxito en este rol?**

Sobre el equipo y la cultura

6. **¿Cómo describiría la dinámica del equipo y la cultura de la empresa?**
7. **¿Con quién trabajaré más estrechamente?**
8. **¿Cómo se fomenta el desarrollo profesional y el aprendizaje dentro del equipo?**
9. **¿Cómo maneja la empresa el equilibrio entre el trabajo y la vida personal?**
10. **¿Puede compartir un ejemplo de cómo la empresa apoya a los empleados en tiempos de dificultad o desafíos?**

Sobre la compañía y sus objetivos

11. **¿Cuáles son los objetivos a corto y largo plazo de la empresa?**

12. **¿Qué cambios recientes o futuros importantes están planeados para la empresa o este departamento?**
13. **¿Cómo describiría el enfoque de la empresa hacia la innovación y la mejora continua?**
14. **¿Qué desafíos enfrenta actualmente la empresa y cómo se están abordando?**
15. **¿Cuáles son los valores fundamentales de la compañía y cómo se reflejan en el día a día?**

Sobre oportunidades y crecimiento

16. **¿Cuáles son las oportunidades de crecimiento profesional dentro de la empresa?**
17. **¿Ofrecen programas de formación o certificaciones para los empleados?**
18. **¿Cómo apoya la empresa a los empleados que desean desarrollar nuevas habilidades?**
19. **¿Es común que las personas dentro de la empresa asciendan a roles más altos?**

Sobre el proceso y siguientes pasos

20. **¿Cómo es el proceso de evaluación de desempeño en la empresa?**
21. **¿Qué puede compartir sobre los próximos pasos en el proceso de selección?**
22. **¿Hay algo más que pueda proporcionarles o aclarar sobre mi experiencia para ayudar en su decisión?**

Estas preguntas te ayudan a entender mejor la cultura, las expectativas y las oportunidades del puesto, además de mostrar que estás comprometido y pensando estratégicamente sobre tu futuro en la empresa.

What are the most important ways to prepare for a job interview?

- Research the company
- Look at company and SEC reports
- Not understanding the company shows lack of preparation
- Ask thoughtful questions
- Practice with mock interviews

very well is mock interviews because they've helped me be less scripted.

Tameika Askew
Data Science Product Manager, U.S. Navy

What are the most important ways to prepare for a job interview?

- Preparation takes time
- Build your experience and knowledge
- Know and meet the specific job requirements

addressing these specific requirements and make sure that you are like up to date on those.

Romeo Kienzler
CTO and Chief Data Scientist, IBM

Can you give examples of what a candidate might see in technical screens (such as code challenges or take-home problems)?

1. Study LeetCode and Kaggle
2. Take IBM courses based on SQL and Python
3. Study the integration of coding into machine learning algorithms

cos para la pantalla técnica. Yo diría que, independientemente de la persona qu
el puesto,

Principal Data Scientist, Skill-Up Technologies

The STAR method

Situation: The project or challenge you faced

Task: Your responsibility in the situation

Action: The steps or procedure you followed to solve the problem

Result: What happened after you acted

Behavioral questions

- What would you do if you were asked to perform a task and weren't sure how to complete it?
- Tell me about a time when you went above and beyond your expected tasks while working on a project
- How would you overcome the challenge of working with a difficult coworker on a team project?

Questions to avoid

- How many holidays do you offer?
- Do you have long working hours?
- Do you have another open position?
- What does the company do?

Avoid questions that make it look as though you are uninterested, careless, or ill-informed

Don’t negotiate salary now

Can you give examples of behavioral questions that a candidate might encounter?

- Interviewer is looking for red flags
- Come prepared with a set of stories
- Describe a conflict or recent project
- Rehearse the behavioral question stories
- Be authentic and confident

Can you give examples of behavioral questions that a candidate might encounter?

- There are classic questions that most interviewers ask
- Questions about problems or conflicts
- Do research on behavioral questions and suggested answers
- Be authentic and truthful

Antonio Cangiano

What kind of questions should a candidate ask during an interview? What questions are not appropriate?

- What does success look like in the first six months?
- Why did the previous person leave?
- Don't ask about sick days and PTO
- Those questions give a bad impression in the first interview

What kind of questions should a candidate ask during an interview? What questions are not appropriate?

- Have some prepared questions
- Ask questions related to the job and the interviewer's experience
- Ask questions about the team
- Ask about a typical workday
- Don't ask personal or confrontational questions
- Don't focus on benefits, vacation, and sick leave

Antonio Cangiano
Software Development Manager, IBM

What is the best way to negotiate a job offer?

1. Wait for the employer to make an offer, and then make your counteroffer
2. Don't offer a salary figure at the outset
3. Do not undervalue yourself
4. Research the salary range for the job position
5. Request an offer higher than the minimum you would accept