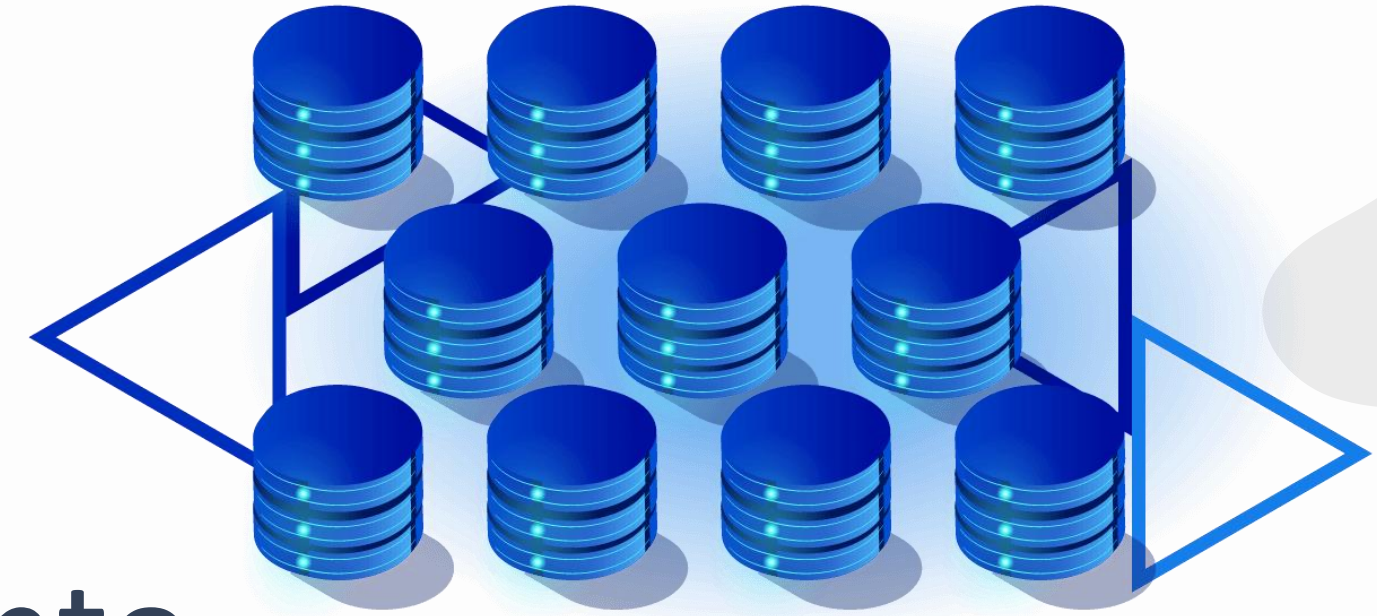
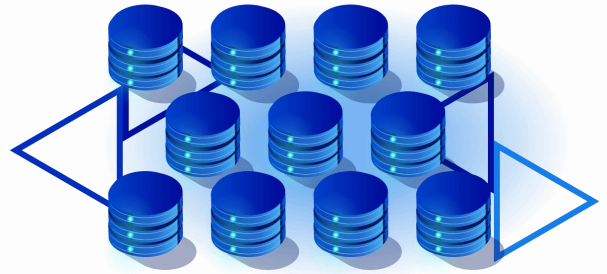
Tema 5. Recolección y almacenamiento de datos.



Gobernanza de datos y Big Data en los ayuntamientos del siglo XXI

FVMP
Federació Valenciana de Municipis i Províncies

Contenidos del tema



- Recolección de datos
  - Importancia de la recolección de datos.
  - Fuentes de datos
- Métodos de recolección de datos
  - Automatizados
  - Manuales
- Almacenamiento de datos
  - Conceptos básicos
  - Arquitecturas de almacenamiento.
- Tecnologías de almacenamiento
  - BD relacionales
  - BD NoSQL
  - DataWarehouses
  - DataLakes
- Buenas prácticas en almacenamiento de datos
  - Seguridad de los datos
  - Optimización y escalabilidad
- Resumen y preguntas.

Introducción a la recolección de datos

- La **recolección** de datos es el proceso sistemático de obtener, medir y analizar información de diversas fuentes con el fin de adquirir conocimientos precisos y útiles para la toma de decisiones. Este proceso abarca la identificación de las fuentes de datos, la selección de métodos de recolección adecuados, la captura efectiva de datos y la garantía de la calidad y precisión de los mismos.

- La recolección de datos es esencial para entender y responder a las dinámicas del mercado, mejorar procesos internos, personalizar la oferta de productos y servicios, y cumplir con los objetivos estratégicos de la organización. Se lleva a cabo utilizando diversas técnicas y herramientas, dependiendo del tipo de datos y las necesidades específicas del negocio.

Ejemplos de recolecciones exitosas

- **Venta al por menor:** Empresas como Amazon utilizan datos de compras, navegación y preferencias del cliente para **personalizar recomendaciones** y mejorar la experiencia del usuario.

- **Salud:** Instituciones médicas utilizan datos de pacientes para mejorar diagnósticos y tratamientos, optimizando la atención al paciente.

- **Transporte y Logística:** Compañías como UPS recolectan datos de rutas y entregas para optimizar tiempos y reducir costos.

¿Se os ocurre algún caso de recolección en vuestra organización?

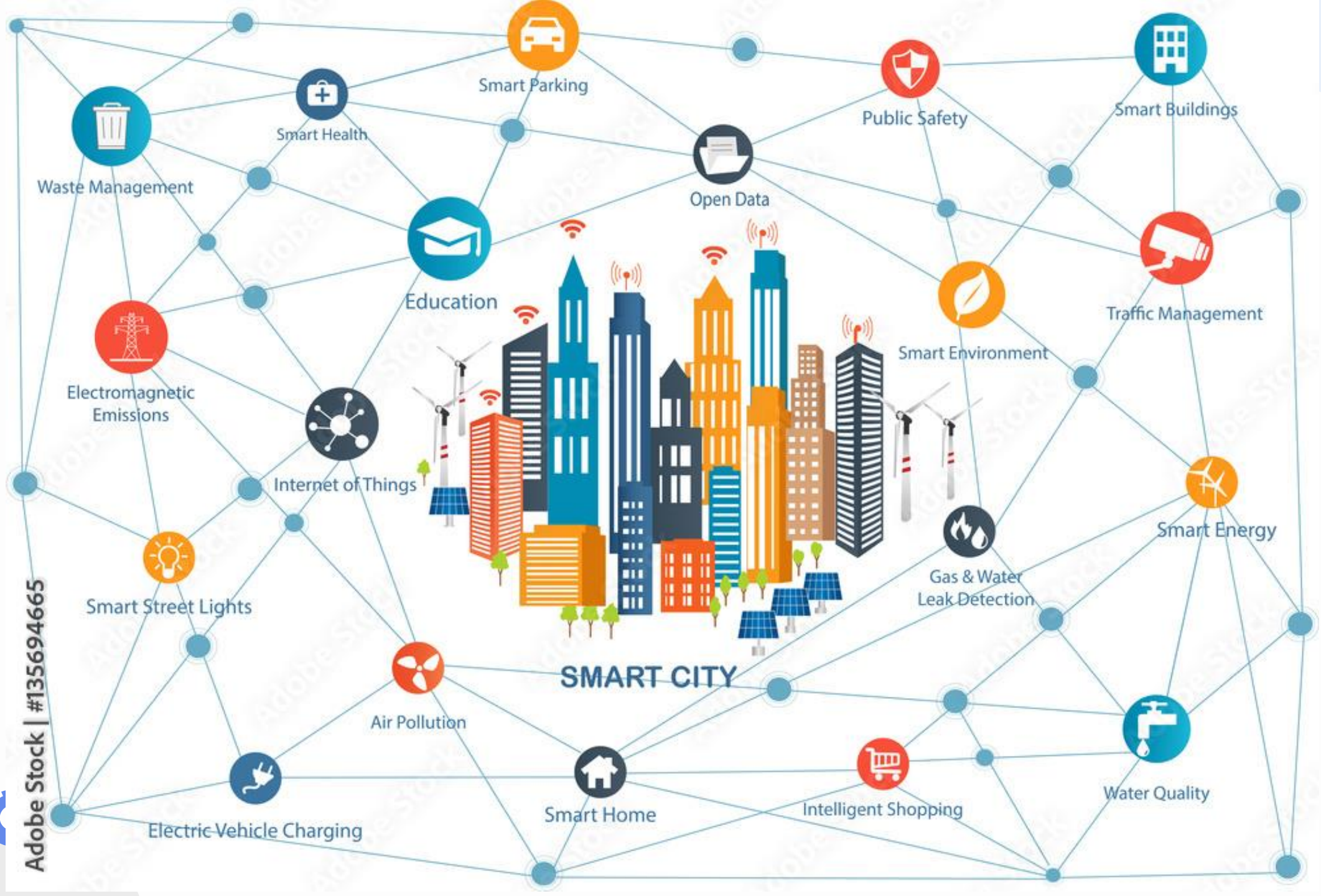
Fuentes de datos

- **Estructurados, semi estructurados y no estructurados.**
- Fuentes internas
  - Bases de datos.
  - Sistemas de información
    - Web corporativa
    - Contabilidad
    - Sistema recursos humanos
    - Seguimiento de expedientes
    - GIS
    - Smart City
    - Gestión tributaria
    - ...
- Fuentes externas
  - Redes sociales
  - Datos públicos (datos abiertos)
  - El BIG DATA
  - Datos contratados

Métodos de recolección de datos (1)

- **Métodos automatizados:** La recolección automatizada de datos implica el uso de tecnologías y sistemas que capturan datos sin intervención humana directa. Estos métodos son eficaces para manejar grandes volúmenes de datos y asegurar la consistencia y precisión en la captura de información.
  - Sensores y dispositivos IoT.
    - Herramientas que capturan datos del entorno físico y los transmiten a sistemas centralizados para su análisis.
    - Ejemplos: sensores de temperatura, humedad, movimiento, presión, tráfico...
    - Aplicaciones prácticas:
      - Industria: Monitoreo de maquinaria y condiciones de operación para mantenimiento predictivo.
      - Agricultura: Seguimiento de condiciones del suelo y clima para optimizar el riego y fertilización.
      - Salud: Dispositivos portátiles que monitorean signos vitales de pacientes en tiempo real.
    - Beneficios: En tiempo real, evita errores humanos y se manejan muchos datos.


IoT devices
e-healthcare
Satellite
Smart building
Cellular Tower
IoT devices
Gateway
Smart school
RSU
Cellular Tower
Cellular Tower
RSU
Smart Transportation
e-healthcare


Smart Parking
Smart Health
Waste Management
Public Safety
Smart Buildings
Open Data
Education
Traffic Management
Smart Environment
Electromagnetic Emissions
Internet of Things
Smart Energy
Smart Street Lights
Gas & Water Leak Detection
SMART CITY
Air Pollution
Water Quality
Smart Home
Intelligent Shopping
Electric Vehicle Charging
Adobe Stock | #135694665

SENSOR NETWORKS DATABASES AND DATA MANAGEMENT

SMART VISUALIZATION

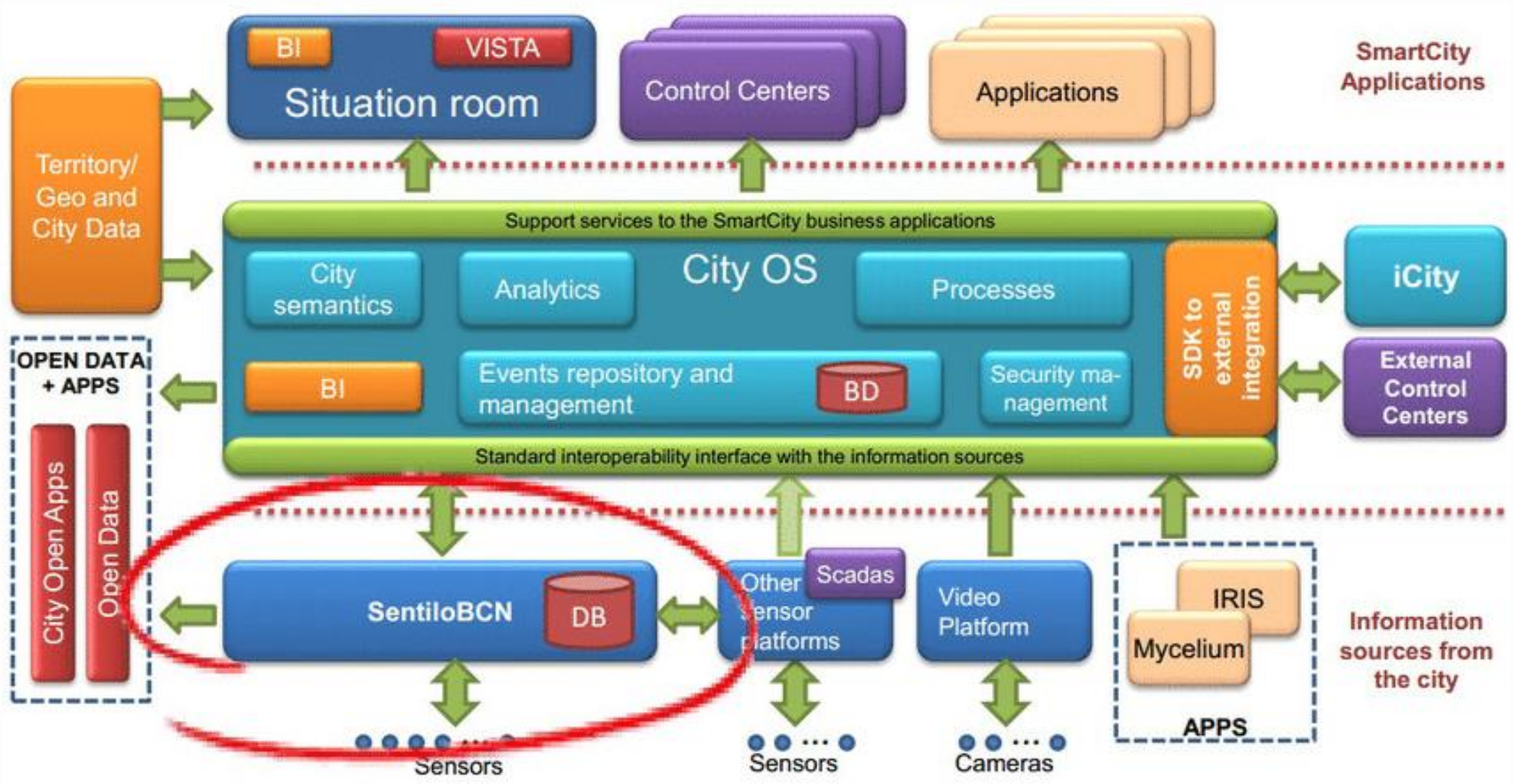
SMART CITY DATABASE

QUALITY AND VERACITY

CITY ANALYTICS

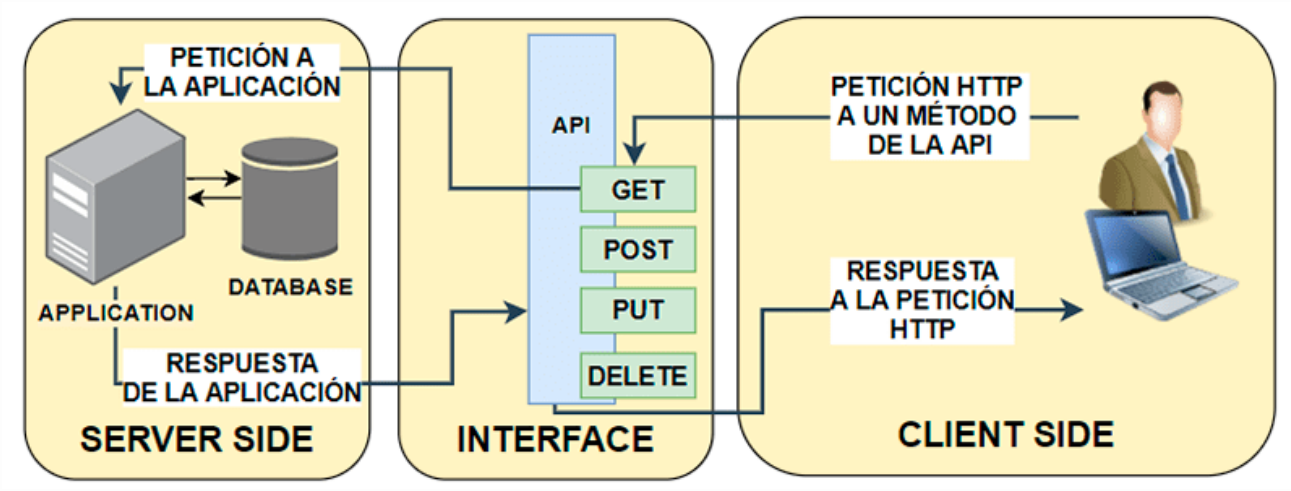
GDPR

Arquitectura Smart City (Ejemplo)



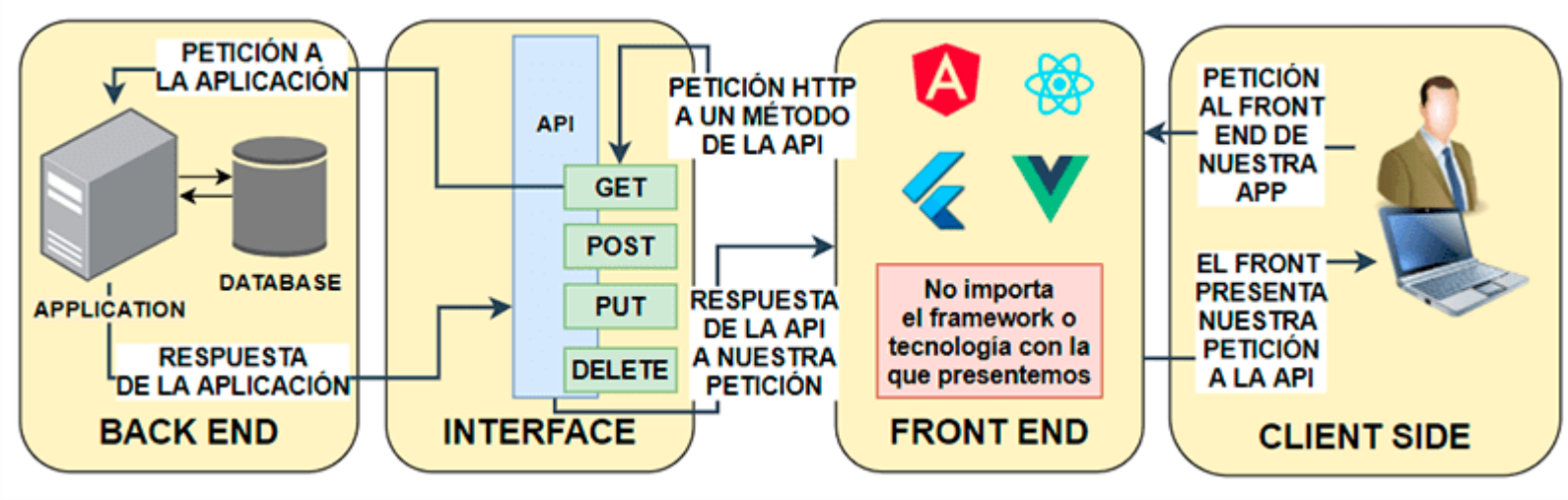
Métodos de recolección de datos (2)

- APIs (Interfaz de Programación de Aplicaciones)
  - Las APIs son conjuntos de reglas que permiten a diferentes aplicaciones comunicarse entre sí y compartir datos.
  - Facilitan la integración de sistemas y el acceso a datos de manera programática.
  - Aplicaciones prácticas:
    - La integración entre la contabilidad y la GT
    - La integración entre la contabilidad y el gestor de expedientes.
    - La recolección de datos de los dispositivos IOT
  - Ventajas
    - Interoperabilidad entre diferentes sistemas.
    - Automatización de flujos de trabajo y procesos.
    - Acceso a datos externos en tiempo real.



API sin FrontEnd (Directa)

API con FrontEnd (Indirecta)



Métodos de recolección de datos (3)

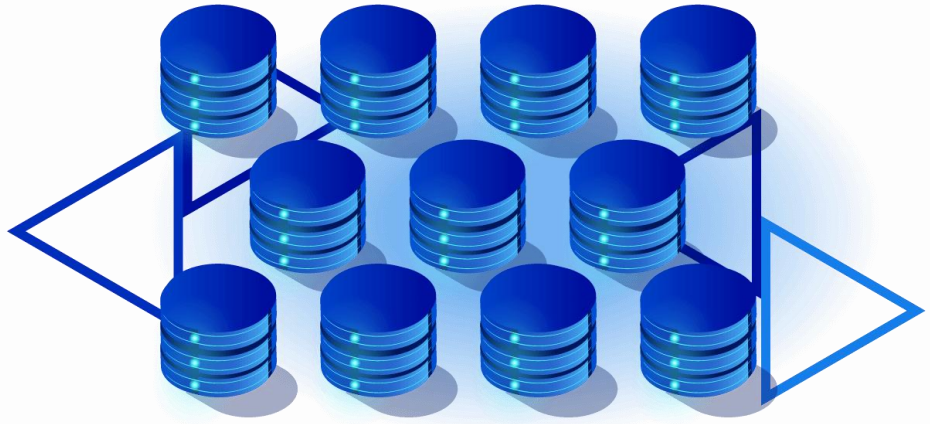
- **Web Scraping (Big data)**
  - Web scraping es la técnica de extraer datos de sitios web mediante el uso de bots o programas automatizados.
  - Implica el análisis del contenido de las páginas web y la recolección de información relevante.
  - Ejemplos
    - Investigación de Mercado: Extracción de opiniones y reseñas de productos de plataformas de comercio electrónico.
    - Periodismo de Datos: Obtención de datos públicos para reportajes e investigaciones.
  - Beneficios
    - Acceso a datos que no están disponibles a través de APIs.
    - Posibilidad de recopilar información masiva de forma rápida y eficiente.
    - Flexibilidad para obtener datos de diversas fuentes y formatos.

Métodos de recolección de datos (4)

- **Software de tracking y registro.**
  - Estos programas monitorean y registran actividades y eventos en sistemas informáticos y aplicaciones.
  - Incluyen herramientas de analítica web, software de gestión de logs y sistemas de monitoreo de red.
  - Aplicaciones prácticas
    - Analítica Web: Google Analytics, que rastrea el comportamiento del usuario en sitios web.
    - Seguridad Informática: Sistemas de gestión de eventos e información de seguridad (SIEM) que registran y analizan eventos de seguridad.
    - Desarrollo de Software: Herramientas de integración continua y despliegue continuo (CI/CD) que monitorean y registran el ciclo de vida del desarrollo de software.
  - Ventajas
    - Obtención de datos precisos sobre el comportamiento y rendimiento del sistema.
    - Identificación de problemas y oportunidades de optimización.
    - Automatización de la recolección y análisis de grandes volúmenes de datos.

Almacenamiento de datos

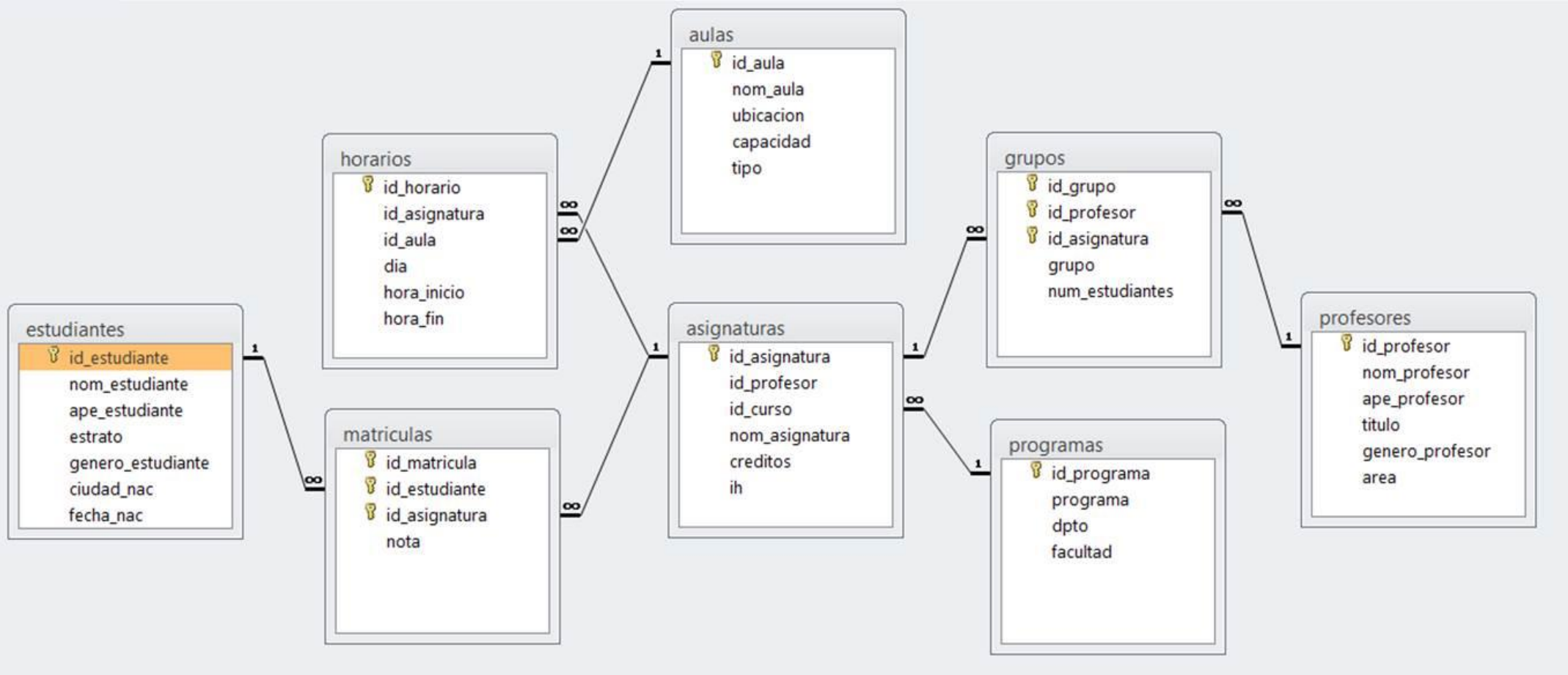
- El almacenamiento de datos es el proceso de guardar y gestionar información digital en un sistema que permita su acceso y uso eficiente. Este proceso es fundamental para la gobernanza de datos y big data, ya que asegura que los datos recolectados estén disponibles, seguros y organizados para su análisis posterior.



Bases de Datos Relacionales vs. NoSQL

- Bases de datos relacionales:
  - **Definición**: Sistemas que organizan los datos en tablas con filas y columnas. Utilizan SQL (Structured Query Language) para gestionar y consultar datos.
  - **Características**: Estructura fija y predefinida, integridad referencial, ideal para transacciones y aplicaciones empresariales.
  - **Ejemplos**: MySQL, PostgreSQL, Oracle.
  - **Ventajas**: Alta consistencia, soporte para transacciones ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad).
  - **Desventajas**: Escalabilidad limitada en comparación con algunas bases de datos NoSQL, menos flexibilidad para datos no estructurados.

Ejemplo de esquema de base de datos relacional

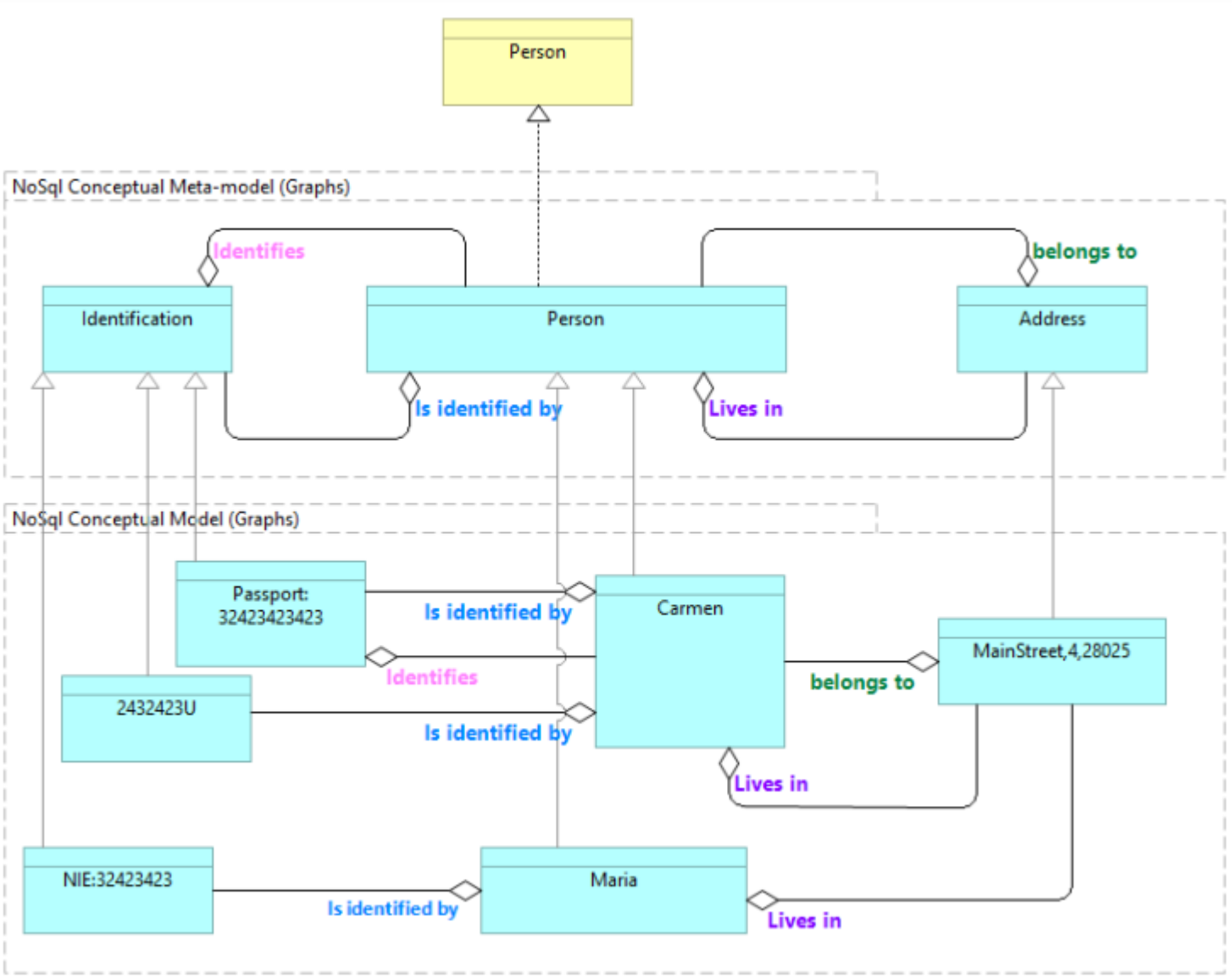


*Select * from estudiantes where id_estudiante not in (select id_estudiante from matriculas)*
¿Qué devolvería?

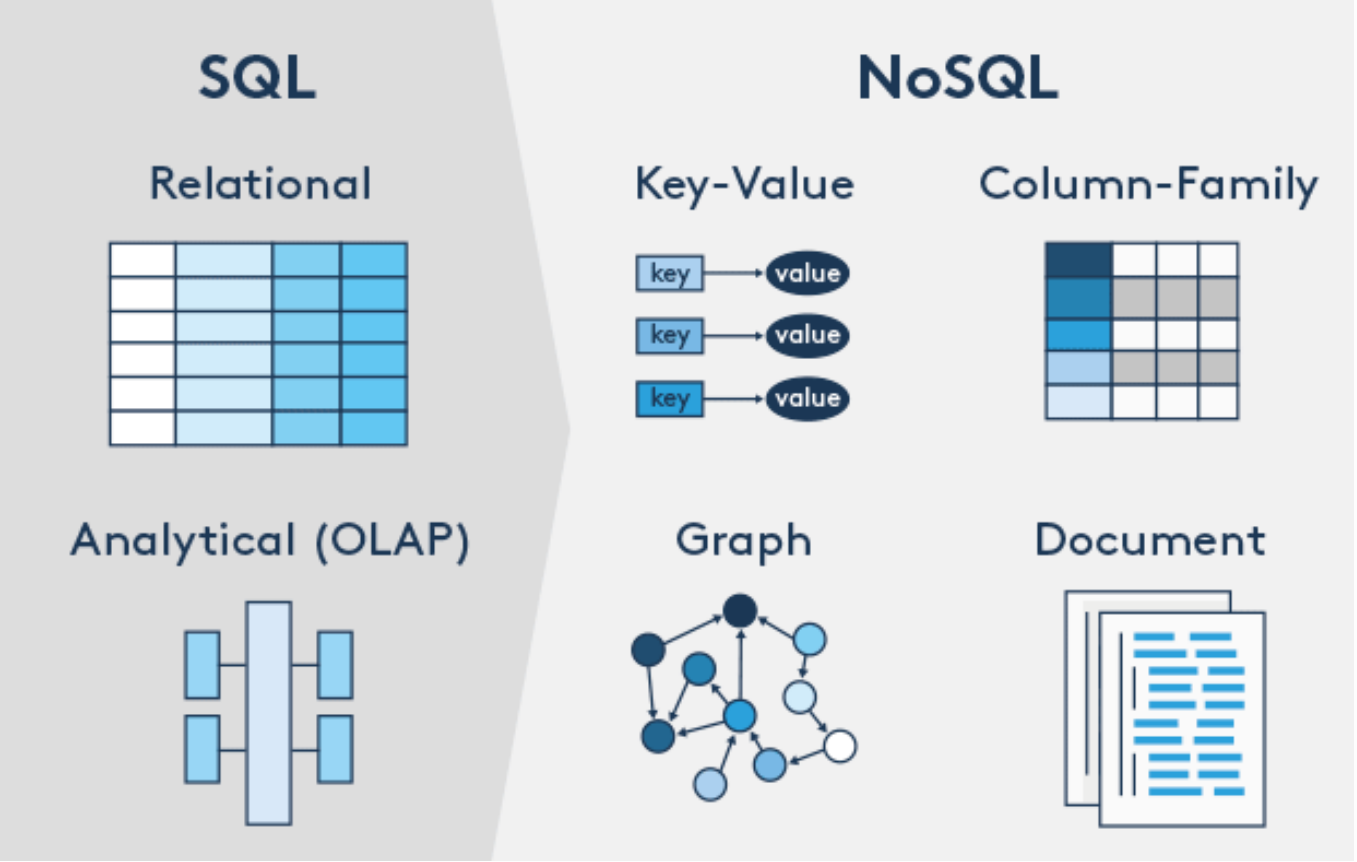
Bases de Datos Relacionales vs. NoSQL

- **Bases de datos NoSQL**:
  - **Definición:** Sistemas diseñados para almacenar y gestionar grandes volúmenes de datos no estructurados o semi-estructurados. No utilizan un esquema fijo y pueden ser orientados a documentos, clave-valor, columnares o gráficos.
  - **Características:** Escalabilidad horizontal, flexibilidad en el esquema, mejor rendimiento para ciertos tipos de consultas.
  - **Ejemplos:** MongoDB, Cassandra, HBase.
  - **Ventajas:** Escalabilidad masiva, flexibilidad, adecuada para big data y aplicaciones en tiempo real.
  - **Desventajas:** Menor consistencia en algunas configuraciones, falta de soporte para transacciones complejas.

Modelo NO SQL

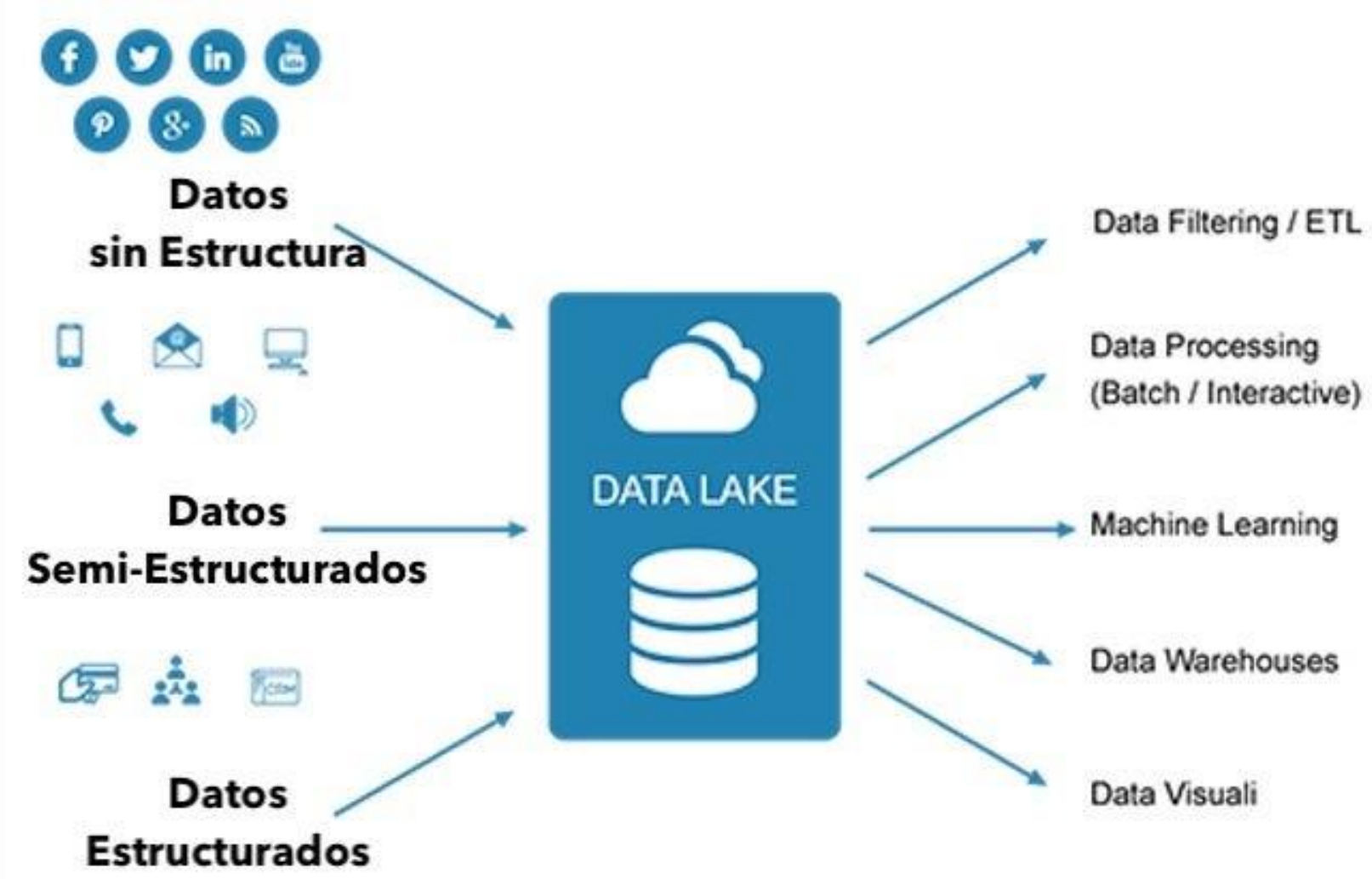


Resúmen


SQL
NoSQL
Relational
Key-Value
Column-Family
key
value
key
value
key
value
Analytical (OLAP)
Graph
Document

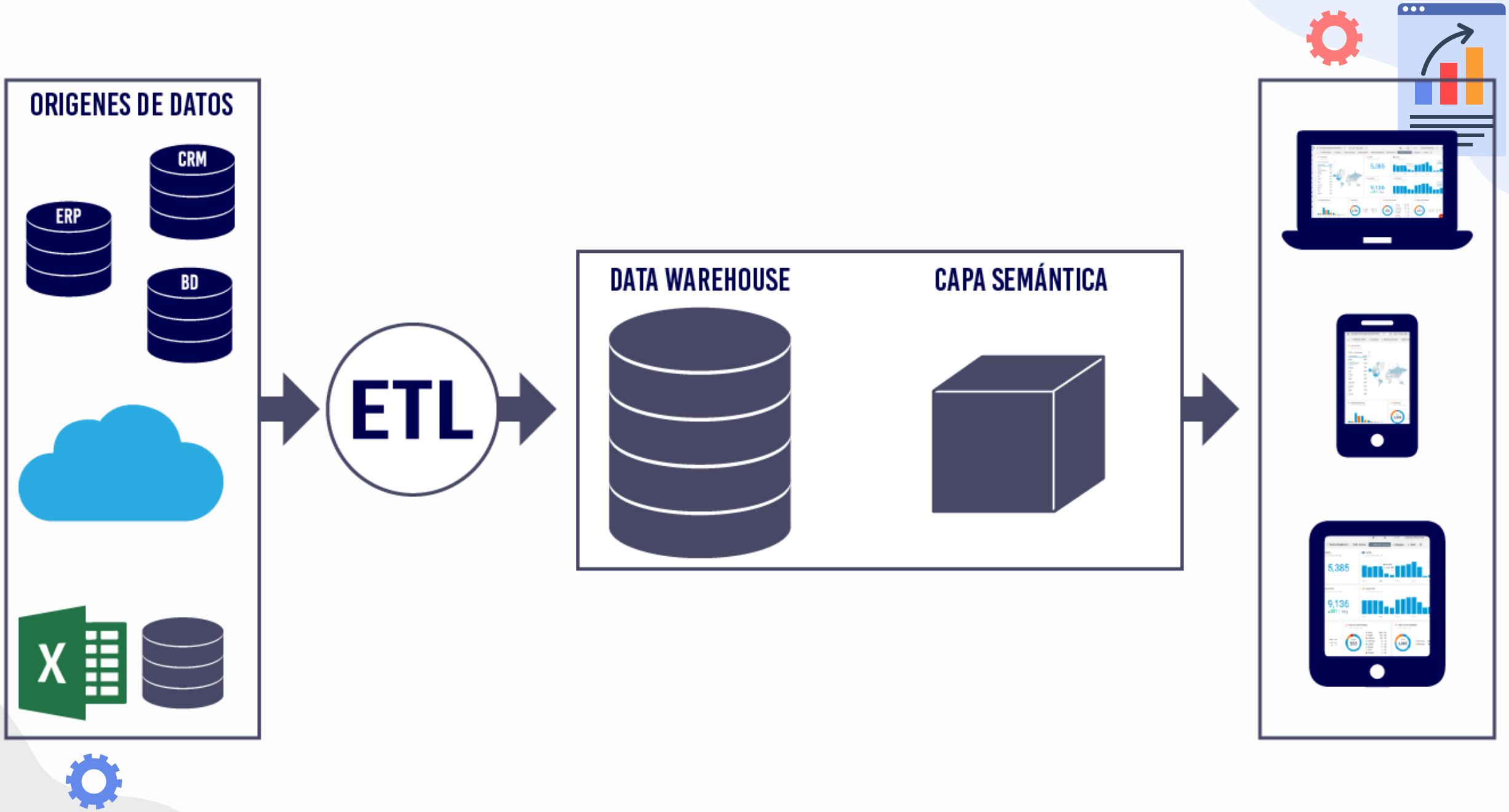
Datalakes y Datawarehouse

- **Datalake**
  - Definición: Repositorios centralizados que permiten almacenar grandes volúmenes de datos en su forma nativa o sin procesar. Son ideales para datos no estructurados y semi-estructurados.
  - Características: Alta capacidad de almacenamiento, flexibilidad para distintos tipos de datos, diseño para análisis masivo y procesamiento en paralelo.
  - Ejemplos: Apache Hadoop, Amazon S3.
  - Ventajas: Almacenamiento económico, capacidad de manejar datos variados, ideal para análisis avanzados y machine learning.
  - **Desventajas: Requiere una gobernanza estricta para evitar la creación de "data swamps" (lagos de datos desorganizados).**


Datos
sin Estructura
Datos
Semi-Estructurados
Datos
Estructurados
DATA LAKE
Data Filtering / ETL
Data Processing
(Batch / Interactive)
Machine Learning
Data Warehouses
Data Visuali

Datalakes y Datawarehouses

- Datawarehouse (DW)
  - **Definición**: Sistemas optimizados para el almacenamiento y consulta de datos estructurados y transaccionales. Están diseñados para informes y análisis de datos históricos.
  - **Características**: Estructura de datos organizada, soporte para consultas complejas, optimización para lectura y agregación de datos.
  - **Ejemplos**: Amazon Redshift, Google BigQuery, Snowflake.
  - **Ventajas**: Alto rendimiento para consultas complejas, integración con herramientas de BI (Business Intelligence), consistencia de datos.
  - **Desventajas**: Costos elevados, menos flexibilidad para datos no estructurados.


ORIGENES DE DATOS
CRM
ERP
BD
X
ETL
DATA WAREHOUSE
CAPA SEMÁNTICA
5,385
9,136

Arquitecturas de almacenamiento (1)

- **On premise**: Infraestructura de almacenamiento instalada y gestionada dentro de las instalaciones físicas de la organización.
  - Ventajas:
    - Control total sobre la infraestructura y los datos.
    - Mayor seguridad física.
    - Cumplimiento con requisitos específicos de regulación y privacidad.
  - Desventajas:
    - Costos elevados de implementación y mantenimiento.
    - Escalabilidad limitada sin inversiones adicionales significativas.
    - Necesidad de personal técnico especializado.

Arquitecturas de almacenamiento (2)

- **Cloud**: Infraestructura de almacenamiento proporcionada y gestionada por proveedores de servicios en la nube. Los datos se almacenan en servidores remotos accesibles a través de internet.
  - Ventajas:
    - Escalabilidad y flexibilidad prácticamente ilimitadas.
    - Reducción de costos iniciales y pago por uso.
    - Actualizaciones y mantenimiento gestionados por el proveedor.
  - Desventajas:
    - Dependencia del proveedor y posibles problemas de latencia.
    - Consideraciones de seguridad y privacidad de datos.
    - Costos recurrentes a largo plazo.

Arquitecturas de almacenamiento (3)

- **Híbrido y Multi-Cloud**: Combinación de infraestructuras on-premise y en la nube, o uso de múltiples proveedores de nube para diferentes necesidades de almacenamiento.
  - Ventajas:
    - Flexibilidad para elegir la mejor opción para cada tipo de datos y aplicación.
    - Redundancia y mayor resiliencia ante fallos.
    - Optimización de costos y rendimiento.
  - Desventajas:
    - Complejidad en la gestión y la integración de diferentes sistemas.
    - Necesidad de políticas claras de gobernanza de datos.

Tecnologías de almacenamiento (Ejemplos de mercado)

- En esta sección se explorarán las diversas tecnologías de almacenamiento que se utilizan para gestionar grandes volúmenes de datos en el contexto de big data. Cada tecnología tiene sus propias características, ventajas y desventajas, y su elección depende de las necesidades específicas de la organización y del tipo de datos que se manejen.

Bases de datos relacionales

- **MySQL**
  - MySQL es un sistema de gestión de bases de datos relacional (RDBMS) de **código abierto** que utiliza SQL (Structured Query Language) como lenguaje de consulta.
  - Características:
    - Alta fiabilidad y rendimiento.
    - Compatible con muchas aplicaciones y lenguajes de programación.
    - Escalabilidad moderada para cargas de trabajo pequeñas a medianas.
  - Casos de Uso:
    - Aplicaciones web, e-commerce, sistemas de gestión de contenido.
  - Ventajas:
    - Fácil de usar y administrar.
    - Gran comunidad de soporte y abundante documentación.
  - Desventajas:
    - Escalabilidad limitada en comparación con otras bases de datos.

Bases de datos relacionales

- **PostgreSQL**
  - PostgreSQL es un RDBMS de **código abierto** conocido por su robustez y conformidad con los estándares SQL.
  - Características:
    - Soporte para tipos de datos avanzados y funciones complejas.
    - Alto rendimiento y fiabilidad.
    - Extensibilidad a través de funciones y operadores personalizados.
  - Casos de Uso:
    - Aplicaciones empresariales, análisis de datos, sistemas geoespaciales.
  - Ventajas:
    - Muy versátil y potente.
    - Buena escalabilidad y rendimiento en operaciones complejas.
  - Desventajas:
    - Curva de aprendizaje más pronunciada que otros RDBMS.

PostgreSQL

Bases de datos relacionales

- **Oracle**
  - Oracle Database es un RDBMS desarrollado por Oracle Corporation, conocido por su capacidad para manejar grandes volúmenes de datos y aplicaciones empresariales complejas
  - Características:
    - Alta disponibilidad y recuperación ante desastres.
    - Funcionalidades avanzadas de seguridad y administración.
    - Soporte para transacciones ACID.
      - Conjunto de propiedades (**atomicidad , consistencia , aislamiento y durabilidad**) que garantizan la integridad de los datos en las transacciones de bases de datos.
  - Casos de Uso:
    - Aplicaciones críticas de negocios, procesamiento de transacciones en línea (OLTP), data warehousing.
  - Ventajas:
    - Altamente escalable y confiable.
    - Amplia gama de funcionalidades y soporte.
  - Desventajas:
    - Costos elevados de licencias y mantenimiento.

Bases de datos NoSQL

- **MongoDB**
  - MongoDB es una base de datos NoSQL orientada a documentos que almacena datos en formato BSON (Binary JSON).
  - Características:
    - Alta flexibilidad y escalabilidad horizontal.
    - Soporte para consultas ad-hoc y indexación.
    - Diseño sin esquema (schema-less), lo que permite cambios rápidos en la estructura de datos.
  - Casos de Uso:
    - Aplicaciones web, big data, Internet de las Cosas (IoT).
  - Ventajas:
    - Ideal para datos no estructurados y semi-estructurados.
    - Facilidad de escalado y ajuste a necesidades cambiantes.
  - Desventajas:
    - Menos adecuada para aplicaciones que requieren **transacciones complejas y consistencia estricta**.

Bases de datos NoSQL

- **Casandra**
  - Apache Cassandra es una base de datos distribuida NoSQL diseñada para manejar grandes cantidades de datos a través de muchos servidores sin un punto único de fallo.
  - Características:
    - Alta disponibilidad y escalabilidad horizontal.
    - Modelo de datos basado en columnas.
    - Tolerancia a fallos y replicación de datos.
  - Casos de Uso:
    - Servicios en la nube, aplicaciones de redes sociales, análisis en tiempo real.
  - Ventajas:
    - Excelente rendimiento en entornos distribuidos.
    - Escalabilidad prácticamente ilimitada.
  - Desventajas:
    - Complejidad en la configuración y gestión.

Bases de datos NoSQL

- **HBase**
  - Apache HBase es una base de datos distribuida, NoSQL, de código abierto, que se ejecuta sobre el sistema de archivos distribuido Hadoop (HDFS).
  - Características:
    - Modelo de datos basado en tablas de columnas.
    - Alta escalabilidad y capacidad de manejar grandes volúmenes de datos.
    - Integración con el ecosistema Hadoop para procesamiento distribuido.
  - Casos de Uso:
    - Análisis de big data, almacenamiento de datos históricos, aplicaciones en tiempo real.
  - Ventajas:
    - Escalabilidad horizontal y capacidad de gestionar datos masivos.
    - Integración estrecha con Hadoop y sus herramientas.
  - Desventajas:
    - Curva de aprendizaje pronunciada y alta complejidad operativa.

Datawarehouses (DW)

- **Amazon Redshift**
  - Amazon Redshift es un servicio de almacenamiento de datos en la nube completamente gestionado y escalable ofrecido por Amazon Web Services (AWS).
  - Características:
    - Alto rendimiento para consultas complejas.
    - Soporte para carga masiva de datos.
    - Integración con el ecosistema de AWS.
  - Casos de Uso:
    - Análisis de datos empresariales, informes, BI (Business Intelligence).
  - Ventajas:
    - Facilidad de uso y administración.
    - Escalabilidad y rendimiento robusto.
  - Desventajas:
    - Costos variables según el uso y almacenamiento.

amazon REDSHIFT

Datawarehouses (DW)

- **Google BigQuery**
  - Google BigQuery es un almacén de datos en la nube completamente gestionado y altamente escalable, ofrecido por Google Cloud Platform.
  - Características:
    - Consultas SQL rápidas y en tiempo real.
    - Alta capacidad de procesamiento y escalabilidad.
    - Modelo de pago por consulta.
  - Casos de Uso:
    - Análisis de big data, BI, informes ad-hoc.
  - Ventajas:
    - Rápido y eficiente para análisis masivos.
    - Sin necesidad de gestionar la infraestructura.
  - Desventajas:
    - Costos pueden acumularse con el uso intensivo.

Datawarehouses (DW)

- **Snowflake**
  - Snowflake es una plataforma de almacenamiento de datos en la nube que ofrece capacidades de data warehousing, análisis y data lakes.
  - Características:
    - Separación de almacenamiento y computación para escalabilidad independiente.
    - Soporte para datos estructurados y semi-estructurados.
    - Alta disponibilidad y recuperación ante desastres.
  - Casos de Uso:
    - BI, análisis de big data, almacenamiento unificado de datos.
  - Ventajas:
    - Rendimiento escalable y flexible.
    - Modelo de pago por uso.
  - Desventajas:
    - Dependencia del entorno cloud y potenciales costos adicionales.

Datalakes

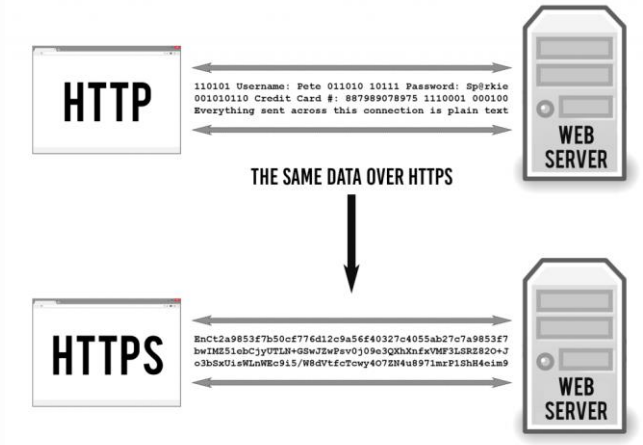
- **Apache Hadoop**
  - Apache Hadoop es un framework de software de código abierto que permite el almacenamiento y procesamiento distribuido de grandes volúmenes de datos utilizando hardware común.
  - Características:
    - Alta capacidad de almacenamiento y procesamiento distribuido.
    - Componentes principales: HDFS (Hadoop Distributed File System) y MapReduce.
    - Extensible con herramientas como Hive, Pig y HBase.
  - Casos de Uso:
    - Procesamiento de big data, almacenamiento masivo de datos sin procesar, análisis avanzado.
  - Ventajas:
    - Manejo eficiente de grandes volúmenes de datos.
    - Comunidad activa y ecosistema en crecimiento.
  - Desventajas:
    - Complejidad en la configuración y gestión.

Datalakes

- **Amazon S3**
  - Amazon Simple Storage Service (S3) es un servicio de almacenamiento de objetos escalable y de alta durabilidad ofrecido por Amazon Web Services (AWS).
  - Características:
    - Almacenamiento de objetos con acceso a través de API.
    - Alta durabilidad (99.999999999% de durabilidad) y disponibilidad.
    - Integración con otros servicios de AWS.
  - Casos de Uso:
    - Data lakes, almacenamiento de backups, almacenamiento de contenido estático.
  - Ventajas:
    - Escalabilidad y durabilidad extremas.
    - Fácil integración con otras herramientas de análisis y procesamiento.
  - Desventajas:
    - Costos pueden acumularse con el uso intensivo y almacenamiento a largo plazo.

Buenas prácticas

- **Encriptación en tránsito**
  - Descripción:
    - Protege los datos mientras se trasladan entre el cliente y el servidor, o entre diferentes sistemas y servicios.
  - Prácticas:
    - Utilizar protocolos seguros como HTTPS (TLS/SSL) para proteger los datos transmitidos a través de la web.
    - Implementar encriptación de datos en las APIs y servicios web para asegurar que la información no sea interceptada durante la transmisión.
    - Configurar redes privadas virtuales (VPN) para proteger la comunicación entre diferentes partes de la infraestructura.



Buenas prácticas

- **Encriptación en reposo:**
  - Descripción:
    - Protege los datos almacenados para que no puedan ser accedidos sin la clave de encriptación, incluso si el almacenamiento físico es comprometido.
  - Prácticas:
    - Utilizar algoritmos de encriptación fuertes (como AES-256) para encriptar los datos en discos y bases de datos.
    - Implementar cifrado a nivel de sistema de archivos o disco completo en servidores y dispositivos de almacenamiento.
    - Asegurar que las claves de encriptación sean gestionadas de manera segura, utilizando soluciones de gestión de claves (KMS) y políticas de rotación de claves.

Buenas prácticas

- **Control de acceso:**
  - Descripción:
    - Define quién puede acceder a los datos y qué acciones pueden realizar.
  - Prácticas:
    - Implementar control de acceso basado en roles (RBAC) para asignar permisos específicos según el rol del usuario en la organización.
    - Utilizar listas de control de acceso (ACLs) para definir permisos detallados sobre archivos y recursos específicos.
    - Realizar auditorías y revisiones periódicas de las políticas de acceso para asegurar su cumplimiento y adecuación.

Buenas prácticas – Optimización y escalabilidad

- **Indexación**:
  - Descripción:
    - Mejora el rendimiento de las consultas al crear estructuras de datos adicionales que permiten un acceso rápido a los registros.
  - Prácticas:
    - Crear índices en las columnas que se utilizan frecuentemente en las consultas para acelerar el tiempo de respuesta.
    - Utilizar índices compuestos para mejorar el rendimiento de consultas complejas que involucren múltiples columnas.
    - Mantener y actualizar regularmente los índices para asegurar su efectividad y evitar la fragmentación.

Buenas prácticas – Optimización y escalabilidad

- **Particionamiento**:
  - Descripción:
    - Divide grandes conjuntos de datos en partes más pequeñas y manejables para mejorar el rendimiento y la gestión.
  - Prácticas:
    - Implementar particionamiento horizontal (sharding) para distribuir los datos entre diferentes nodos o servidores, mejorando la escalabilidad horizontal.
    - Utilizar particionamiento vertical para separar columnas específicas en diferentes tablas, optimizando el acceso a datos que se utilizan con mayor frecuencia.
    - Configurar particiones basadas en rangos, listas o hashes según el patrón de acceso a los datos y los requisitos de la aplicación.

Preguntas, dudas

Comming up next…

Tema 6: **Calidad de datos y limpieza** (Vicente Castelló) (Dia 22 (hoy) y 24 mayo)

Tema 7. **Análisis de datos y visualización.** (Vicente Castelló) (Dia 24 y 28 mayo)