LICENCIATURA EN CIENCIAS DE LA COMPUTACIÓN

Teoría de Base de Datos

Prof. Sergio Salinas
(Año 2025)

GUÍA DE ACTIVIDADES

La materia de Teoría de Base de Datos constituye un pilar fundamental en la formación académica de los estudiantes de la Licenciatura en Ciencias de la Computación de la Facultad de Ingeniería de la Universidad Nacional de Cuyo. En este contexto, el desarrollo de competencias prácticas y teóricas en el manejo, diseño e implementación de bases de datos no solo es esencial para su formación profesional, sino que también representa una herramienta crítica en el procesamiento y gestión de la información en la era digital.

Esta guía de actividades ha sido diseñada para acompañar a los estudiantes a través del aprendizaje que abarca desde los conceptos fundamentales de las bases de datos hasta las técnicas más avanzadas de modelado, diseño, y optimización. El propósito es ofrecer un recurso integral que, mediante ejercicios prácticos, casos de estudio, y proyectos, facilite la comprensión profunda de los principios teóricos y su aplicación en escenarios reales.

A lo largo de esta guía, los estudiantes tendrán la oportunidad de enfrentarse a desafíos que simulen situaciones de la vida real, promoviendo así el desarrollo de habilidades analíticas, críticas, y creativas. Se promoverá el trabajo colaborativo, fomentando la discusión y el intercambio de ideas, aspectos fundamentales en la dinámica profesional del campo de la informática.

Esta guía de actividades está compuesta por cinco trabajos prácticos los cuales deben realizarse de la siguiente forma. El trabajo práctico número 1 se puede resolver en grupo de un máximo de 4 alumnos. El grupo deberá elaborar un documento con un resumen de los conceptos relacionados con bases de datos. El documento debe ser una guía rápida para acceder a una definición breve de cada concepto. Los trabajos prácticos 2 y 3 se deben resolver en forma individual ya que permitirá la configuración de herramientas para trabajar durante el cursado de la materia. En este caso de forma individual se deberá elaborar una guía de los pasos de instalación y cualquier otra información relevante para documentar el proceso de instalación y configuración de las herramientas utilizadas. Los trabajos prácticos 4, 5 y 6 deben resolverse en su totalidad cada uno de los casos presentados de forma individual. En cuanto a la documentación es posible crear scripts en SQL o crear un documento pdf con el contenido del código utilizado. Para cada trabajo práctico habrá un plazo de entrega máximo y deberá enviarse en el formato correspondiente mediante la plataforma moodle antes de la fecha límite establecida.

Trabajo Práctico 1

Investigación sobre conceptos relacionados con datos.

En este trabajo práctico es posible utilizar cualquier herramienta disponible para obtener información sobre el amplio universo de los datos en los tiempos actuales. Para ello se propone analizar los conceptos que se presentan a continuación y eventualmente exponerlos en clases. El resultado obtenido es posible ser traducido a código LaTeX, copiar y pegar el resultado en un documento en la plataforma Overleaf (www.overleaf.com). En clase se mostrará mediante ejemplos como utilizar estas herramientas. El documento generado formará parte de la entrega del trabajo práctico que deberá subirse a la plataforma Moodle (aula abierta).

1 Conceptos generales relacionados con el concepto "datos".

1. **Datos:** Representan hechos, observaciones o registros que pueden ser procesados, almacenados o transmitidos y que tienen algún significado implícito o explícito.
2. **Datos estructurados:** Son datos organizados en un formato específico y definido, como tablas en una base de datos relacional, que facilitan su almacenamiento, búsqueda y análisis.
3. **Datos no estructurados:** Son datos que no tienen una estructura predefinida y pueden incluir texto sin formato, imágenes, videos, archivos de audio, correos electrónicos, etc.
4. **Datos semiestructurados:** Son datos que no se ajustan a un modelo de datos formal, pero tienen una estructura parcialmente definida, como documentos JSON, XML o archivos CSV.
5. **Big Data:** Se refiere a conjuntos de datos extremadamente grandes y complejos que superan la capacidad de las herramientas de procesamiento de datos tradicionales para capturar, almacenar, gestionar y analizar de manera efectiva.
6. **Análisis de datos:** Es el proceso de examinar, limpiar, transformar y modelar datos para descubrir información útil, llegar a conclusiones y apoyar la toma de decisiones.
7. **Minería de datos:** Es el proceso de descubrir patrones, tendencias y conocimientos ocultos en grandes conjuntos de datos utilizando técnicas de estadísticas, aprendizaje automático y análisis predictivo.
8. **Almacén de datos:** Es una base de datos centralizada que se utiliza para almacenar datos de diferentes fuentes en un formato estructurado y optimizado para el análisis y la generación de informes.
9. **Inteligencia de negocios (BI):** Es el conjunto de tecnologías, herramientas y prácticas que permiten a las organizaciones recopilar, almacenar, analizar y visualizar datos para obtener información empresarial y apoyar la toma de decisiones.
10. **Aprendizaje automático:** Es un subcampo de la inteligencia artificial que se centra en el desarrollo de algoritmos y modelos que permiten a las computadoras aprender a partir de datos y realizar tareas sin ser programadas explícitamente.

11. **Visualización de datos:** Es el proceso de representar datos de manera gráfica y visualmente atractiva para facilitar la comprensión y el análisis de patrones, tendencias y relaciones en los datos.

12. **Privacidad de datos:** Se refiere a la protección de la información personal y confidencial de los individuos, así como al cumplimiento de regulaciones y leyes relacionadas con la recopilación, almacenamiento y procesamiento de datos.

13. **Gobierno de datos:** Es el conjunto de procesos, políticas, normas y controles que aseguran la calidad, integridad, confiabilidad y seguridad de los datos en toda una organización.

14. **Algoritmo:** Es un conjunto ordenado de operaciones o reglas bien definidas que permiten realizar un cálculo, resolver un problema o realizar una tarea específica.

15. **Modelo de datos:** Es una representación conceptual de la estructura y relaciones entre los datos en una base de datos o sistema de información.

16. **SQL (Structured Query Language):** Es un lenguaje de programación utilizado para gestionar y manipular bases de datos relacionales, que permite realizar consultas, actualizaciones, inserciones y eliminaciones de datos.

17. **NoSQL:** Es un término genérico que se refiere a una amplia gama de tecnologías de bases de datos que no se basan en el modelo relacional tradicional, como bases de datos de documentos, bases de datos de grafos y bases de datos de clave-valor.

18. **Blockchain:** Es una tecnología de registro distribuido que utiliza criptografía para garantizar la seguridad, integridad y transparencia de los datos almacenados en una red descentralizada.

19. **Metadatos:** Son datos que describen y proporcionan información sobre otros datos, como su estructura, significado, origen, calidad y contexto.

20. **Transacción:** Es una secuencia de operaciones que se ejecutan como una unidad coherente e indivisible en una base de datos, garantizando la integridad y consistencia de los datos.

21. **Normalización:** Es el proceso de diseñar una base de datos relacional de manera que evite la redundancia y la inconsistencia de datos mediante la eliminación de dependencias funcionales no deseadas.

22. **Desnormalización:** Es el proceso de optimizar el rendimiento de una base de datos relacional al permitir cierto grado de redundancia de datos en lugar de seguir estrictamente los principios de normalización.

23. **Índice:** Es una estructura de datos utilizada para mejorar la velocidad de recuperación de datos al proporcionar un acceso rápido a registros específicos en una base de datos.

24. **Repositorio de datos:** Es un sistema de almacenamiento centralizado que almacena y gestiona datos en un formato estándar y coherente para su uso y análisis en toda una organización.

25. **ETL (Extract, Transform, Load):** Es el proceso de extraer datos de múltiples fuentes, transformarlos en un formato adecuado y cargarlos en un almacén de datos para su análisis y generación de informes.

26. **Data Lake:** Es un repositorio de datos que almacena grandes volúmenes de datos en su formato original, sin necesidad de estructuración previa, lo que permite un acceso rápido y flexible a los datos para su análisis y procesamiento.

27. **Data Mart:** Es un subconjunto de un almacén de datos que se enfoca en un área específica de interés, como ventas, marketing o finanzas, y que se utiliza para facilitar el análisis y la generación de informes en esa área.

28. **Data Warehouse en la nube:** Es un almacén de datos alojado en la nube que proporciona escalabilidad, flexibilidad y bajo costo de infraestructura para almacenar y analizar grandes volúmenes de datos.

29. **Streaming de datos:** Es el proceso de transferir continuamente datos en tiempo real desde múltiples fuentes a un sistema de procesamiento de datos para su análisis y procesamiento inmediato.

30. **Segmentación de datos:** Es el proceso de dividir un conjunto de datos en subconjuntos más pequeños o segmentos con características similares, lo que facilita su análisis y tratamiento individualizado.

31. **Almacenamiento en la memoria:** Es una técnica de almacenamiento de datos que utiliza la memoria RAM en lugar de discos duros para almacenar y acceder a datos de manera más rápida y eficiente.

32. **Replicación de datos:** Es el proceso de copiar y distribuir datos desde una fuente a uno o más destinos para garantizar la disponibilidad, redundancia y tolerancia a fallos de los datos.

33. **Backup y recuperación de datos:** Son procesos y procedimientos diseñados para proteger y restaurar datos en caso de pérdida, corrupción o desastre, garantizando la disponibilidad y la integridad de los datos.

34. **Almacenamiento en la nube:** Es un modelo de almacenamiento de datos en el que los datos se almacenan en servidores remotos y se accede a ellos a través de Internet, lo que proporciona escalabilidad, flexibilidad y disponibilidad bajo demanda.

35. **Data Governance (Gobierno de Datos):** Es el conjunto de procesos, políticas, normas y controles que aseguran la calidad, integridad, confiabilidad y seguridad de los datos en toda una organización.

36. **Data Quality (Calidad de Datos):** Es el grado en que los datos son precisos, completos, consistentes y relevantes para su uso previsto, lo que garantiza la confiabilidad y la utilidad de los datos para la toma de decisiones.

37. **Data Privacy (Privacidad de Datos):** Se refiere a la protección de la información personal y confidencial de los individuos, así como al cumplimiento de regulaciones y leyes relacionadas con la recopilación, almacenamiento y procesamiento de datos.

38. **Data Security (Seguridad de Datos):** Es el conjunto de medidas y controles diseñados para proteger los datos contra accesos no autorizados, pérdida, robo, corrupción y daños, garantizando la confidencialidad, integridad y disponibilidad de los datos.

39. **Data Discovery (Descubrimiento de Datos):** Es el proceso de encontrar y explorar datos relevantes para un análisis específico, identificando fuentes de datos, entendiendo su estructura y calidad, y evaluando su idoneidad para su uso previsto.

40. **Data Integration (Integración de Datos):** Es el proceso de combinar datos de múltiples fuentes y formatos en una sola vista coherente y unificada, lo que facilita su análisis y uso en toda la organización.

41. **Data Science (Ciencia de Datos):** Es un campo interdisciplinario que utiliza métodos científicos, procesos, algoritmos y sistemas para extraer conocimientos y patrones útiles a partir de datos, lo que permite la toma de decisiones informadas y la generación de valor.

42. **Data Engineering (Ingeniería de Datos):** Es el conjunto de técnicas, herramientas y prácticas utilizadas para diseñar, construir, mantener y gestionar sistemas y arquitecturas de datos que soportan el procesamiento, almacenamiento y análisis de datos a gran escala.

43. **Data Analyst (Analista de Datos):** Es un profesional que utiliza técnicas analíticas y herramientas para explorar, interpretar y visualizar datos, identificar patrones y tendencias, y comunicar hallazgos y recomendaciones a las partes interesadas.

44. **Data Scientist (Científico de Datos):** Es un profesional con habilidades en matemáticas, estadísticas, programación y análisis de datos que utiliza técnicas avanzadas de análisis para extraer conocimientos y generar valor a partir de grandes volúmenes de datos.

45. **Data Architect (Arquitecto de Datos):** Es un profesional que diseña la estructura, arquitectura y modelos de datos de un sistema de información para garantizar la integridad, calidad y eficiencia de los datos, así como su alineación con los objetivos empresariales.

46. **Data Warehouse Architect (Arquitecto de Almacén de Datos):** Es un profesional especializado en el diseño, implementación y gestión de almacenes de datos y sistemas de Business Intelligence que integran datos de diferentes fuentes para su análisis y generación de informes.

47. **Data Engineer (Ingeniero de Datos):** Es un profesional que desarrolla y mantiene sistemas y arquitecturas de datos para la recopilación, almacenamiento, procesamiento y análisis de datos a gran escala, utilizando herramientas y plataformas de Big Data y Cloud Computing.

48. **Data Steward (Guardián de Datos):** Es un profesional responsable de definir y mantener políticas, normas y procesos relacionados con la gestión, calidad, privacidad y seguridad de los datos en una organización.

49. **Data Governance Committee (Comité de Gobierno de Datos):** Es un grupo de personas dentro de una organización que se encarga de establecer y supervisar políticas, normas y procesos relacionados con la gestión, calidad, privacidad y seguridad de los datos.

50. **Data Lake Architect (Arquitecto de Data Lake):** Es un profesional especializado en el diseño, implementación y gestión de repositorios de datos en formato sin procesar que almacenan grandes volúmenes de datos de diversas fuentes para su análisis y procesamiento posterior.

2 Formatos de Intercambio de Datos

1. JSON (JavaScript Object Notation)
2. XML (eXtensible Markup Language)
3. CSV (Comma-Separated Values)
4. YAML (YAML Ain't Markup Language)
5. Parquet
6. Avro
7. ORC (Optimized Row Columnar)
8. Protocol Buffers (Protobuf)
9. Apache Arrow
10. BSON (Binary JSON)
11. HDF5 (Hierarchical Data Format version 5)
12. Excel (XLSX, XLS)
13. JSON Lines (JSONL)
14. MessagePack
15. TOML (Tom's Obvious, Minimal Language)
16. INI (Initialization)
17. EDI (Electronic Data Interchange)
18. PDF (Portable Document Format)
19. HTML (Hypertext Markup Language)
20. RTF (Rich Text Format)
21. SOAP (Simple Object Access Protocol)
22. GraphQL

23. Apache Thrift
24. RDBMS (Relational Database Management System) dump files (e.g., SQL, MySQL dump)
25. GeoJSON (Geospatial JSON)
26. KML (Keyhole Markup Language)
27. GeoTIFF (Georeferenced Tagged Image File Format)
28. GML (Geography Markup Language)
29. GeoRSS (Geographically Encoded Objects for RSS Feeds)
30. WKT (Well-Known Text)
31. WKB (Well-Known Binary)
32. GPX (GPS Exchange Format)
33. DICOM (Digital Imaging and Communications in Medicine)
34. FITS (Flexible Image Transport System)
35. NetCDF (Network Common Data Form)
36. ASN.1 (Abstract Syntax Notation One)
37. FIX (Financial Information eXchange)
38. SWIFT (Society for Worldwide Interbank Financial Telecommunication)
39. OData (Open Data Protocol)
40. OPC UA (Open Platform Communications Unified Architecture)
41. MQTT (Message Queuing Telemetry Transport)
42. STIX (Structured Threat Information eXpression)
43. OpenAPI (formerly known as Swagger)
44. RIF (Rule Interchange Format)
45. XLIFF (XML Localization Interchange File Format)
46. XBRL (eXtensible Business Reporting Language)
47. DITA (Darwin Information Typing Architecture)
48. SAML (Security Assertion Markup Language)
49. Sitemaps (XML)
50. CAP (Common Alerting Protocol)

3 Protocolos de Comunicación para Datos

1. HTTP (Protocolo de Transferencia de Hipertexto)
2. HTTPS (Protocolo de Transferencia de Hipertexto Seguro)
3. FTP (Protocolo de Transferencia de Archivos)
4. FTPS (Protocolo de Transferencia de Archivos Seguro)
5. SFTP (Protocolo de Transferencia de Archivos a través de SSH)
6. SCP (Protocolo de Copia Segura)
7. SMTP (Protocolo Simple de Transferencia de Correo)
8. IMAP (Protocolo de Acceso a Mensajes de Internet)
9. POP3 (Protocolo de Oficina de Correos versión 3)
10. LDAP (Protocolo Ligero de Acceso a Directorios)
11. DNS (Sistema de Nombres de Dominio)
12. DHCP (Protocolo de Configuración Dinámica de Host)
13. SNMP (Protocolo Simple de Administración de Red)
14. Telnet
15. SSH (Shell Seguro)
16. RDP (Protocolo de Escritorio Remoto)
17. VNC (Computación de Red Virtual)
18. WebSocket
19. MQTT (Protocolo de Telemetría de Cola de Mensajes)
20. AMQP (Protocolo Avanzado de Cola de Mensajes)
21. CoAP (Protocolo de Aplicación Restringida)
22. DDS (Servicio de Distribución de Datos)
23. RTMP (Protocolo de Mensajes en Tiempo Real)
24. RTSP (Protocolo de Streaming en Tiempo Real)
25. SIP (Protocolo de Inicio de Sesión)
26. XMPP (Protocolo Extensible de Mensajes y Presencia)
27. WebRTC (Comunicación en Tiempo Real en la Web)

28. SMB (Bloque de Mensajes del Servidor)
29. NFS (Sistema de Archivos de Red)
30. AFP (Protocolo de Archivo de Apple)
31. Bonjour (Redes de Configuración Cero)
32. NTP (Protocolo de Tiempo de Red)
33. PPTP (Protocolo de Túnel de Punto a Punto)
34. L2TP (Protocolo de Túnel de Capa 2)
35. IPsec (Seguridad de Protocolo de Internet)
36. BGP (Protocolo de Gateway de Frontera)
37. OSPF (Primer Camino más Corto Abierto)
38. EIGRP (Protocolo de Enrutamiento de Puerta de Enlace Interior Mejorado)
39. ICMP (Protocolo de Mensaje de Control de Internet)
40. IGMP (Protocolo de Gestión de Grupo de Internet)
41. ARP (Protocolo de Resolución de Direcciones)
42. DHCPv6 (Protocolo de Configuración Dinámica de Host para IPv6)
43. PPP (Protocolo de Punto a Punto)
44. PPPoE (Protocolo de Punto a Punto sobre Ethernet)
45. MPLS (Conmutación de Etiquetas Multiprotocolo)
46. LDP (Protocolo de Distribución de Etiquetas)
47. RSVP (Protocolo de Reserva de Recursos)
48. IKE (Intercambio de Claves de Internet)
49. IS-IS (Sistema Intermedio a Sistema Intermedio)
50. PPTP (Protocolo de Túnel de Punto a Punto)

4 Rol de las Bases de Datos Relacionales en el Mundo de los Datos

En el mundo de los datos, las bases de datos relacionales juegan un papel fundamental en la gestión, almacenamiento y manipulación de grandes cantidades de información. Aquí hay varios roles importantes que desempeñan las bases de datos relacionales:

1. **Almacenamiento de Datos:** Las bases de datos relacionales proporcionan un entorno estructurado y organizado para almacenar datos de manera eficiente. Utilizan tablas con filas y columnas para almacenar información en un formato fácilmente accesible.

2. **Gestión de Datos:** Las bases de datos relacionales permiten gestionar los datos de manera centralizada, lo que facilita su acceso, actualización y mantenimiento. Los sistemas de gestión de bases de datos relacionales (SGBD) proporcionan herramientas y funciones para administrar bases de datos, como la creación de tablas, la gestión de usuarios y la optimización del rendimiento.

3. **Integridad de Datos:** Las bases de datos relacionales garantizan la integridad de los datos mediante la aplicación de reglas y restricciones, como las claves primarias y foráneas, que garantizan la consistencia y la coherencia de los datos almacenados.

4. **Consulta y Análisis de Datos:** Una de las principales fortalezas de las bases de datos relacionales es su capacidad para realizar consultas complejas y análisis de datos. Los usuarios pueden utilizar lenguajes de consulta, como SQL (Structured Query Language), para extraer información específica de la base de datos y realizar análisis estadísticos, agregaciones y filtrados.

5. **Seguridad de Datos:** Las bases de datos relacionales proporcionan funciones de seguridad para proteger los datos sensibles de accesos no autorizados. Esto incluye la gestión de usuarios y permisos, la encriptación de datos y el registro de auditoría para realizar un seguimiento de las actividades de acceso y modificación de datos.

6. **Escalabilidad y Confiabilidad:** Las bases de datos relacionales están diseñadas para ser escalables y confiables, lo que significa que pueden gestionar grandes volúmenes de datos y mantener un alto nivel de disponibilidad y rendimiento incluso en entornos de producción críticos.

En resumen, las bases de datos relacionales son una piedra angular en el mundo de los datos, proporcionando una infraestructura robusta y flexible para gestionar y aprovechar el valor de la información en una amplia gama de aplicaciones y sectores industriales.

5 Herramientas del Ecosistema para Gestionar Datos

1. Bases de Datos Relacionales:

   - PostgreSQL
   - MySQL
   - Oracle Database
   - Microsoft SQL Server
   - SQLite
   - MariaDB
   - IBM Db2

2. Bases de Datos NoSQL:

- MongoDB
- Cassandra
- Redis
- Couchbase
- Amazon DynamoDB
- Apache HBase
- Firebase

3. Almacenamiento en la Nube:
   - Amazon S3
   - Google Cloud Storage
   - Microsoft Azure Blob Storage
   - IBM Cloud Object Storage
   - Dropbox
   - Box
   - Backblaze B2
4. Plataformas de Big Data:
   - Apache Hadoop
   - Apache Spark
   - Apache Flink
   - Apache HBase
   - Apache Kafka
   - Apache Hive
   - Apache Cassandra
5. Frameworks de Procesamiento de Datos:
   - Apache Airflow
   - Apache NiFi
   - Apache Beam
   - Apache Storm
   - Apache Samza
   - Apache Flume
   - Spring Batch

6 Herramientas de ETL (Extract, Transform, Load)

- Talend
- Informatica PowerCenter
- Apache NiFi
- Pentaho Data Integration
- Apache Spark
- Microsoft SQL Server Integration Services (SSIS)
- AWS Glue

7 Herramientas de Visualización de Datos

- Tableau
- Power BI
- QlikView
- Looker
- Google Data Studio
- Plotly
- Matplotlib

8 Lenguajes de Programación

- Python
- R
- SQL
- Java
- Scala
- JavaScript
- Julia

9 Frameworks y Bibliotecas de Aprendizaje Automático

- TensorFlow
- PyTorch
- Scikit-learn
- Keras
- XGBoost
- LightGBM
- Spark MLlib

10 Herramientas de Monitorización y Gestión de Datos

- Prometheus
- Grafana
- Nagios
- Splunk
- Datadog
- ELK Stack (Elasticsearch, Logstash, Kibana)
- New Relic

11 Herramientas de Colaboración y Gestión de Proyectos

- Jira
- Trello
- Asana
- Slack
- Microsoft Teams
- Basecamp
- GitHub

12 Herramientas de Seguridad de Datos

- HashiCorp Vault
- CyberArk
- Symantec Data Loss Prevention
- McAfee Data Protection
- IBM Guardium
- Varonis Data Security Platform
- Protegrity

13 Herramientas Tecnológicas Utilizadas en Big Data

1. **Apache Hadoop:** Un framework de software de código abierto para el almacenamiento distribuido y el procesamiento de grandes conjuntos de datos en clústeres de computadoras.
2. **Apache Spark:** Un framework de computación en memoria que proporciona una API unificada para el procesamiento de datos en batch, streaming y machine learning.
3. **Apache Kafka:** Una plataforma de transmisión de datos de código abierto que se utiliza para la ingestión y el procesamiento de flujos de datos en tiempo real.

4. **Apache Flink:** Un sistema de procesamiento de datos distribuido de código abierto que proporciona capacidades de procesamiento de datos en tiempo real y por lotes.

5. **Apache Hive:** Una infraestructura de almacenamiento y procesamiento de datos de código abierto construida sobre Hadoop que permite realizar consultas SQL en grandes conjuntos de datos almacenados en HDFS.

6. **Apache HBase:** Una base de datos NoSQL distribuida y escalable diseñada para proporcionar acceso aleatorio y en tiempo real a grandes volúmenes de datos no estructurados.

7. **Apache Cassandra:** Una base de datos distribuida NoSQL diseñada para manejar grandes volúmenes de datos en clústeres de servidores.

8. **MongoDB:** Una base de datos NoSQL de documento orientado a documentos que se utiliza para el almacenamiento de datos semi-estructurados y no estructurados.

9. **Amazon S3:** Un servicio de almacenamiento en la nube de objetos escalable y altamente disponible proporcionado por Amazon Web Services.

10. **Google BigQuery:** Un servicio de almacén de datos en la nube de Google Cloud Platform que permite realizar consultas SQL en grandes conjuntos de datos almacenados en la nube.

11. **Microsoft Azure HDInsight:** Un servicio de análisis de big data en la nube de Microsoft Azure que proporciona implementaciones administradas de Apache Hadoop, Spark, HBase y otros frameworks de big data.

12. **Cloudera CDH:** Una distribución de software de código abierto de Apache Hadoop y otros proyectos relacionados con el ecosistema de big data.

13. **Hortonworks Data Platform:** Una plataforma de big data de código abierto basada en Apache Hadoop y otros proyectos de código abierto.

14. **Tableau:** Una herramienta de visualización de datos que permite crear dashboards interactivos y paneles de control para analizar y compartir información.

15. **Power BI:** Una herramienta de business intelligence de Microsoft que permite crear informes interactivos y paneles de control para visualizar y analizar datos.

16. **QlikView y Qlik Sense:** Herramientas de business intelligence que permiten la creación de aplicaciones de análisis de datos interactivas y personalizadas.

17. **TensorFlow:** Una biblioteca de código abierto para aprendizaje automático desarrollada por Google que se utiliza para construir y entrenar modelos de aprendizaje automático.

18. **PyTorch:** Una biblioteca de aprendizaje automático de código abierto desarrollada por Facebook que se utiliza para la creación y entrenamiento de modelos de aprendizaje profundo.

19. **Scikit-learn:** Una biblioteca de aprendizaje automático de código abierto para Python que proporciona una amplia gama de algoritmos de aprendizaje automático y herramientas para su uso.

20. **Keras:** Una biblioteca de aprendizaje profundo de alto nivel que se ejecuta sobre TensorFlow y que se utiliza para la construcción y entrenamiento de modelos de redes neuronales.

21. **Docker:** Una plataforma de código abierto para la creación, implementación y ejecución de aplicaciones en contenedores.

22. **Kubernetes:** Un sistema de orquestación de contenedores de código abierto que se utiliza para automatizar el despliegue, la escalabilidad y la gestión de aplicaciones en contenedores.

23. **Apache NiFi:** Una plataforma de automatización de flujo de datos que se utiliza para automatizar el movimiento, procesamiento y gestión de datos en tiempo real.

24. **Elasticsearch:** Un motor de búsqueda y análisis distribuido de código abierto que se utiliza para indexar, buscar y analizar grandes volúmenes de datos en tiempo real.

25. **Logstash:** Una herramienta de procesamiento de datos de código abierto que se utiliza para recopilar, transformar y enviar datos a Elasticsearch u otros sistemas de almacenamiento.

26. **Kibana:** Una herramienta de visualización de datos de código abierto que se utiliza para crear y compartir paneles de control interactivos para analizar datos almacenados en Elasticsearch.

27. **Prometheus:** Un sistema de monitoreo y alerta de código abierto que se utiliza para registrar y analizar métricas de sistemas y servicios en tiempo real.

28. **Grafana:** Una herramienta de visualización de métricas de código abierto que se utiliza para crear paneles de control y gráficos para visualizar y analizar datos de diferentes fuentes.

29. **Nagios:** Una herramienta de monitoreo de código abierto que se utiliza para monitorear sistemas, redes y servicios y para enviar alertas en caso de problemas.

30. **Splunk:** Una plataforma de análisis de datos que se utiliza para buscar, monitorear y analizar grandes volúmenes de datos generados por aplicaciones, sistemas y dispositivos.

31. **Datadog:** Una plataforma de supervisión y análisis de datos en la nube que se utiliza para recopilar, visualizar y analizar métricas y registros de aplicaciones y sistemas.

32. **New Relic:** Una plataforma de monitoreo de aplicaciones y servicios en la nube que se utiliza para supervisar y analizar el rendimiento de las aplicaciones y servicios en tiempo real.

33. **GitHub:** Una plataforma de desarrollo de software de código abierto que se utiliza para alojar, revisar y colaborar en proyectos de software utilizando el control de versiones Git.

34. **Jira:** Una herramienta de gestión de proyectos y seguimiento de problemas que se utiliza para planificar, seguir y administrar proyectos de desarrollo de software y equipos de trabajo.

35. **Trello:** Una herramienta de gestión de proyectos basada en tableros que se utiliza para organizar tareas, proyectos y equipos de trabajo de manera visual y colaborativa.

36. **Asana:** Una herramienta de gestión de proyectos y colaboración en equipo que se utiliza para organizar tareas, asignar responsabilidades y realizar un seguimiento del progreso del trabajo.

37. **Slack:** Una plataforma de mensajería en equipo y colaboración empresarial que se utiliza para comunicarse y colaborar con equipos de trabajo en tiempo real.

38. **Microsoft Teams:** Una plataforma de colaboración en equipo de Microsoft que se utiliza para chatear, realizar videollamadas, compartir archivos y colaborar en proyectos de equipo.

39. **Basecamp:** Una herramienta de gestión de proyectos y colaboración en equipo que se utiliza para organizar tareas, compartir archivos y comunicarse con equipos de trabajo.

40. **HashiCorp Vault:** Una herramienta de gestión de secretos y protección de datos que se utiliza para almacenar, asegurar y controlar el acceso a secretos, como contraseñas y claves de API.

41. **CyberArk:** Una plataforma de gestión de privilegios y protección de identidades que se utiliza para proteger y gestionar el acceso a cuentas privilegiadas y secretos en entornos de TI.

42. **Symantec Data Loss Prevention:** Una solución de seguridad de datos que se utiliza para proteger y prevenir la pérdida de datos confidenciales mediante la monitorización y el control de su uso y movimiento.

43. **McAfee Data Protection:** Una suite de soluciones de seguridad de datos que se utiliza para proteger los datos confidenciales mediante el cifrado, la prevención de la pérdida de datos y la gestión de accesos.

44. **IBM Guardium:** Una solución de seguridad de datos que se utiliza para proteger y controlar el acceso a bases de datos y almacenes de datos mediante la monitorización de la actividad y la detección de amenazas.

45. **Varonis Data Security Platform:** Una plataforma de seguridad de datos que se utiliza para proteger y gestionar el acceso a datos confidenciales mediante la monitorización, la auditoría y el análisis del comportamiento del usuario.

46. **Protegrity:** Una solución de seguridad de datos que se utiliza para proteger y enmascarar datos confidenciales mediante el cifrado, el enmascaramiento y la tokenización.

14 Principales Conceptos de Small Data

1. **Volumen de datos pequeño:** A diferencia del Big Data, Small Data se refiere a conjuntos de datos que son relativamente pequeños en tamaño y pueden ser gestionados y analizados fácilmente utilizando herramientas y técnicas convencionales.

2. **Datos estructurados y bien definidos:** A menudo, los datos en entornos de Small Data son estructurados y bien definidos, lo que significa que están organizados en formatos simples como tablas, hojas de cálculo o archivos de texto delimitados.

3. **Fácilmente accesibles:** Los datos en entornos de Small Data suelen ser fácilmente accesibles y pueden residir en sistemas locales o en la nube. No requieren infraestructuras de almacenamiento y procesamiento especializadas como en el caso del Big Data.

4. **Análisis basado en casos de uso específicos:** El análisis de Small Data se centra en casos de uso específicos y en la obtención de información relevante para resolver problemas concretos o tomar decisiones operativas.

5. **Enfoque en la calidad de los datos:** Dado que los conjuntos de datos son más pequeños, hay una mayor atención a la calidad de los datos, incluyendo la precisión, la integridad y la consistencia de la información.

6. **Rápida generación de insights:** Debido al tamaño y la naturaleza de los datos, el análisis de Small Data tiende a producir insights de manera más rápida y eficiente en comparación con el análisis de Big Data.

7. **Personalización y segmentación:** Los datos en entornos de Small Data suelen utilizarse para personalizar y segmentar la experiencia del cliente, mejorar la precisión de las recomendaciones y optimizar las estrategias de marketing.

8. **Integración con datos externos:** A menudo, los datos en entornos de Small Data se integran con fuentes de datos externas, como datos de redes sociales, datos demográficos o datos de transacciones, para enriquecer el análisis y obtener una visión más completa.

Estos son algunos de los principales conceptos relacionados con Small Data, que se caracteriza por su enfoque en conjuntos de datos más pequeños y gestionables para obtener insights y tomar decisiones informadas.

15 Principales Conceptos sobre Dark Data

El Dark Data se refiere a los conjuntos de datos que son recopilados, procesados y almacenados por organizaciones, pero que no se utilizan ni se analizan de manera activa para generar valor o informaciones relevantes. A continuación, se describen los principales conceptos relacionados con Dark Data:

1. **Datos no estructurados:** El Dark Data a menudo consiste en datos no estructurados, como correos electrónicos, documentos, imágenes, videos, registros de llamadas, mensajes de texto y datos de redes sociales, que son difíciles de analizar con herramientas tradicionales de bases de datos.

2. **Recopilación involuntaria:** A menudo, el Dark Data se recopila de manera involuntaria como subproducto de las operaciones comerciales regulares, sin un propósito específico en mente. Esto puede incluir registros de transacciones antiguas, datos de sensores, registros de servidor, entre otros.

3. **Costo de almacenamiento:** El almacenamiento de grandes volúmenes de Dark Data puede representar un costo significativo para las organizaciones, ya que requiere infraestructura de almacenamiento y gestión de datos adicional, sin proporcionar ningún valor perceptible.

4. **Desafíos de privacidad y cumplimiento:** El Dark Data puede contener información sensible o confidencial, lo que plantea desafíos adicionales en términos de privacidad y cumplimiento

normativo. Las organizaciones deben asegurarse de cumplir con las regulaciones de protección de datos al almacenar y procesar este tipo de información.

5. **Oportunidades desaprovechadas:** A pesar de su aparente falta de utilidad, el Dark Data puede contener información valiosa que podría utilizarse para obtener perspectivas significativas, identificar tendencias, mejorar la toma de decisiones y obtener una ventaja competitiva en el mercado.

6. **Necesidad de estrategias de gestión de datos:** Para aprovechar el potencial del Dark Data, las organizaciones deben implementar estrategias de gestión de datos efectivas que incluyan la identificación, clasificación, almacenamiento, análisis y protección de estos conjuntos de datos.

7. **Tecnologías emergentes:** El avance de tecnologías como el análisis avanzado de datos, el aprendizaje automático y la inteligencia artificial está ayudando a las organizaciones a descubrir y aprovechar el valor oculto en el Dark Data, permitiendo la identificación de patrones y correlaciones que antes pasaban desapercibidos.

En resumen, el Dark Data representa un desafío y una oportunidad para las organizaciones, ya que requiere una gestión adecuada para mitigar los riesgos asociados y aprovechar su potencial para generar valor y obtener insights significativos.

16 Herramientas de Data Lake

1. **Amazon S3 (Simple Storage Service):** Ofrece almacenamiento escalable y duradero en la nube para datos estructurados y no estructurados.

2. **Azure Data Lake Storage:** Proporciona un almacenamiento escalable y seguro para grandes volúmenes de datos en la nube de Microsoft Azure.

3. **Google Cloud Storage:** Ofrece almacenamiento en la nube escalable y flexible para datos estructurados y no estructurados en la plataforma de Google Cloud.

4. **Apache Hadoop:** Es un framework de software de código abierto que permite el procesamiento distribuido de grandes conjuntos de datos en clústeres de computadoras.

5. **Apache Spark:** Es un motor de procesamiento de datos de código abierto que proporciona una interfaz unificada para el procesamiento de datos en memoria y en disco.

6. **Apache Hive:** Es una infraestructura de almacenamiento de datos y un sistema de consulta construido sobre Apache Hadoop para proporcionar análisis de datos escalables y de alto rendimiento.

7. **Apache HBase:** Es una base de datos NoSQL distribuida y escalable construida sobre Hadoop que proporciona acceso aleatorio y lectura/escritura en tiempo real a los datos.

8. **Cloudera Data Platform (CDP):** Es una plataforma de gestión de datos empresariales que integra varias herramientas de código abierto y propietarias para la gestión y el análisis de datos en entornos de Data Lake.

9. **Databricks Delta Lake:** Es una capa de gestión de datos sobre Apache Spark que proporciona transacciones ACID y un control de versiones escalable para los datos en el lago de datos.

10. **AWS Glue:** Es un servicio de ETL (Extract, Transform, Load) totalmente administrado que facilita la preparación y carga de datos en entornos de Data Lake en AWS.

11. **Azure Data Lake Analytics:** Ofrece un servicio de análisis de big data en la nube que permite ejecutar consultas de análisis complejas sobre grandes volúmenes de datos almacenados en Azure Data Lake Storage.

12. **Google Cloud Dataproc:** Es un servicio de Google Cloud Platform que permite ejecutar clústeres de Apache Hadoop y Spark de forma rápida y sencilla en la nube.

Estas son solo algunas de las herramientas comunes utilizadas en entornos de Data Lake. La elección de herramientas dependerá de los requisitos específicos del proyecto, la infraestructura existente y las preferencias del equipo.

17 Principales Herramientas de Data Warehouse

1. **Snowflake:** Es una plataforma de Data Warehouse en la nube que proporciona un entorno escalable y seguro para almacenar y analizar datos estructurados y semiestructurados.

2. **Amazon Redshift:** Es un servicio de Data Warehouse completamente administrado que se ejecuta en la nube de Amazon Web Services (AWS) y permite analizar grandes conjuntos de datos con herramientas de análisis estándar.

3. **Google BigQuery:** Es un almacén de datos completamente administrado y altamente escalable en la nube de Google Cloud Platform que permite realizar consultas rápidas sobre grandes conjuntos de datos utilizando SQL estándar.

4. **Microsoft Azure Synapse Analytics (anteriormente conocido como Azure SQL Data Warehouse):** Es un servicio de análisis de datos que combina almacén de datos y análisis de big data en una única plataforma unificada en la nube de Microsoft Azure.

5. **Teradata:** Es una plataforma de Data Warehouse escalable y de alto rendimiento que permite a las organizaciones almacenar, procesar y analizar grandes volúmenes de datos empresariales.

6. **IBM Db2 Warehouse:** Es un almacén de datos en la nube de IBM que proporciona un entorno seguro y escalable para almacenar y analizar datos estructurados y no estructurados.

7. **Oracle Autonomous Data Warehouse:** Es un servicio de Data Warehouse en la nube de Oracle que utiliza inteligencia artificial y aprendizaje automático para proporcionar un rendimiento óptimo, seguridad y disponibilidad automática.

8. **Vertica:** Es una plataforma de análisis de datos escalable y de alto rendimiento que permite realizar análisis en tiempo real sobre grandes volúmenes de datos.

Estas son algunas de las principales herramientas de Data Warehouse disponibles en el mercado, cada una con sus propias características, capacidades y ventajas competitivas. La elección de una herramienta específica dependerá de los requisitos y objetivos del proyecto, así como de la infraestructura tecnológica existente en la organización.

18 Bases de Datos No Relacionales

1. **MongoDB:** Una base de datos de documentos orientada a documentos JSON, que es escalable y flexible, adecuada para una variedad de casos de uso, como aplicaciones web, móviles e IoT.
2. **Cassandra:** Una base de datos distribuida y altamente escalable diseñada para manejar grandes volúmenes de datos distribuidos a través de múltiples nodos y centros de datos.
3. **Redis:** Una base de datos de estructura de datos en memoria que es rápida y eficiente para el almacenamiento y recuperación de datos en caché, así como para la implementación de colas y otros patrones de mensajería.
4. **Amazon DynamoDB:** Un servicio de base de datos de clave-valor y documento completamente administrado que ofrece rendimiento y escalabilidad para aplicaciones que requieren acceso rápido y predecible a los datos.
5. **Apache CouchDB:** Una base de datos de documentos distribuida y sin esquema que utiliza el formato JSON para almacenar datos y proporciona replicación bidireccional para garantizar la disponibilidad y la resistencia a fallos.
6. **Neo4j:** Una base de datos de grafos que utiliza estructuras de grafo para representar y almacenar datos, lo que permite consultas y análisis de datos altamente relacionados y complejos.
7. **Elasticsearch:** Una base de datos de búsqueda y análisis distribuida que está diseñada para indexar, buscar y analizar grandes volúmenes de datos no estructurados, como logs, documentos y datos geoespaciales.
8. **HBase:** Una base de datos de columnas distribuida que se ejecuta en la infraestructura de Hadoop y proporciona un almacenamiento escalable y de alta disponibilidad para datos estructurados y semiestructurados.
9. **Riak:** Una base de datos de clave-valor distribuida que está diseñada para proporcionar alta disponibilidad, escalabilidad y tolerancia a fallos para aplicaciones web y móviles.
10. **Firebase:** Una plataforma de desarrollo de aplicaciones móviles y web de Google que incluye una base de datos en tiempo real que sincroniza automáticamente los datos entre clientes y servidores en tiempo real.

Esta lista no es exhaustiva y hay muchas otras bases de datos no relacionales disponibles, cada una con sus propias características y casos de uso específicos. La elección de una base de datos no relacional dependerá de los requisitos y objetivos del proyecto, así como de las preferencias tecnológicas y de infraestructura de la organización.

19 Herramientas para el Gobierno de Datos

1. **Collibra:** Plataforma de gobierno de datos que ayuda a las organizaciones a gestionar, encontrar y comprender sus datos.
2. **Informatica Axon:** Solución de gobierno de datos que proporciona un catálogo de datos empresariales, gestión de metadatos y capacidades de colaboración.
3. **Alation:** Plataforma de datos que ofrece descubrimiento de datos, colaboración, gobernanza y automatización para equipos de datos.
4. **IBM InfoSphere Information Governance Catalog:** Herramienta de IBM que proporciona capacidades de descubrimiento, catalogación y gobierno de datos.
5. **SAP Data Intelligence:** Plataforma de SAP que ofrece gobierno de datos, integración de datos y capacidades de inteligencia artificial.
6. **Datumize:** Solución de gobierno de datos que ayuda a las organizaciones a entender, catalogar y gestionar sus datos de forma efectiva.
7. **Dataiku:** Plataforma de ciencia de datos que incluye funcionalidades de gobierno de datos, colaboración y automatización de procesos.
8. **Datawatch Monarch:** Herramienta de preparación de datos que incluye funcionalidades de perfilado y catalogación para el gobierno de datos.
9. **erwin Data Intelligence (erwin DI):** Plataforma de inteligencia de datos que proporciona descubrimiento, catalogación y gobierno de datos.
10. **Talend Data Catalog:** Solución de Talend que ofrece catalogación de datos, gobernanza y colaboración para equipos de datos.

Estas herramientas están diseñadas para ayudar a las organizaciones a establecer políticas, procesos y controles para gestionar eficazmente sus datos y garantizar su calidad, seguridad y cumplimiento normativo.

20 Empresas Líderes en Datos

1. **Amazon Web Services (AWS):** Ofrece una amplia gama de servicios de almacenamiento, procesamiento y análisis de datos en la nube.
2. **Microsoft Azure:** Plataforma en la nube de Microsoft que proporciona servicios para el almacenamiento, procesamiento y análisis de datos, así como herramientas de inteligencia artificial y aprendizaje automático.
3. **Google Cloud Platform (GCP):** Ofrece servicios de almacenamiento, procesamiento y análisis de datos en la nube, así como herramientas de inteligencia artificial y aprendizaje automático.

4. **IBM:** Ofrece soluciones de software y servicios en la nube para el almacenamiento, procesamiento, análisis y gestión de datos, así como herramientas de inteligencia artificial y aprendizaje automático.

5. **Oracle:** Proporciona una amplia gama de soluciones de base de datos y servicios en la nube para el almacenamiento, procesamiento y análisis de datos, así como herramientas de inteligencia artificial y aprendizaje automático.

6. **SAP:** Ofrece soluciones de software empresarial que incluyen sistemas de gestión de datos, análisis de datos y aplicaciones de inteligencia empresarial.

7. **Salesforce:** Proporciona una plataforma en la nube para la gestión de relaciones con los clientes (CRM) que incluye capacidades de análisis de datos y herramientas de inteligencia artificial.

8. **Snowflake:** Ofrece una plataforma de almacenamiento de datos en la nube que permite el almacenamiento y análisis de grandes volúmenes de datos de manera eficiente y escalable.

9. **Cloudera:** Proporciona soluciones de software basadas en Hadoop para el almacenamiento, procesamiento y análisis de datos a escala empresarial.

10. **Databricks:** Ofrece una plataforma unificada de análisis de datos basada en Apache Spark que permite el procesamiento y análisis de grandes volúmenes de datos de manera eficiente.

Estas empresas son reconocidas por su experiencia y liderazgo en el ámbito de los datos, proporcionando soluciones y servicios innovadores para ayudar a las organizaciones a gestionar y aprovechar el valor de sus datos.

21 Trabajos Populares sobre Datos

1. **Científico de Datos (Data Scientist):** Se encarga de analizar grandes volúmenes de datos para obtener información útil y tomar decisiones estratégicas basadas en datos.

2. **Ingeniero de Datos (Data Engineer):** Diseña, construye y mantiene sistemas de procesamiento de datos para almacenar y gestionar grandes volúmenes de datos de manera eficiente.

3. **Analista de Datos (Data Analyst):** Analiza datos para identificar tendencias, patrones y relaciones que puedan ayudar a las organizaciones a tomar decisiones informadas.

4. **Arquitecto de Datos (Data Architect):** Diseña y gestiona la arquitectura de datos de una organización, incluyendo bases de datos, almacenes de datos y sistemas de integración de datos.

5. **Analista de Business Intelligence (BI Analyst):** Utiliza herramientas de business intelligence para analizar datos y crear informes y paneles de control que ayuden a las organizaciones a entender su rendimiento y tomar decisiones estratégicas.

6. **Ingeniero de Machine Learning (Machine Learning Engineer):** Desarrolla y mantiene modelos de aprendizaje automático para resolver problemas empresariales y mejorar procesos.

7. **Analista de Investigación de Mercado (Market Research Analyst):** Recopila, analiza y presenta datos sobre el mercado y los clientes para ayudar a las organizaciones a comprender las tendencias del mercado y tomar decisiones de marketing.

8. **Analista de Seguridad de Datos (Data Security Analyst):** Protege la seguridad y privacidad de los datos de una organización mediante la implementación de medidas de seguridad y la monitorización de posibles amenazas.

9. **Ingeniero DevOps:** Colabora con equipos de desarrollo y operaciones para automatizar y gestionar la infraestructura de TI, incluyendo sistemas de almacenamiento y procesamiento de datos.

10. **Científico de Datos de la Salud (Health Data Scientist):** Utiliza datos de salud para identificar patrones, predecir resultados y mejorar la atención médica y los resultados de los pacientes.

Estos son solo algunos ejemplos de trabajos populares en el campo de los datos, y la demanda de profesionales en este ámbito sigue creciendo a medida que las organizaciones buscan aprovechar el poder de sus datos para obtener ventajas competitivas.

22 Lenguajes de Programación de Uso en Datos en Orden de Importancia

1. **Python:** Es ampliamente considerado como el lenguaje de programación más importante en el ámbito de los datos debido a su facilidad de uso, su amplia variedad de bibliotecas y su popularidad en la comunidad de ciencia de datos.

2. **SQL (Structured Query Language):** Aunque no es un lenguaje de programación en el sentido tradicional, SQL es fundamental para trabajar con bases de datos relacionales y realizar consultas para extraer y manipular datos.

3. **R:** Es especialmente popular en la academia y la investigación, así como en el análisis estadístico y la visualización de datos. A menudo se utiliza en conjunción con Python para tareas específicas.

4. **Java:** Aunque no es tan comúnmente asociado con el análisis de datos como Python o R, Java sigue siendo relevante en el ámbito de los datos, especialmente en el desarrollo de aplicaciones empresariales y en la implementación de sistemas de gestión de datos.

5. **Scala:** Es un lenguaje de programación que se ejecuta en la máquina virtual de Java (JVM) y es especialmente popular en el procesamiento de datos distribuidos con Apache Spark.

6. **JavaScript:** Además de su uso en el desarrollo web, JavaScript también se utiliza en el ámbito de los datos, especialmente en la visualización interactiva de datos en aplicaciones web.

7. **Julia:** Es conocido por su velocidad de ejecución y su facilidad de uso en cálculos numéricos y científicos. Aunque menos utilizado que Python o R, está ganando popularidad en el ámbito de la ciencia de datos y el cómputo técnico.

Esta lista no es exhaustiva y la importancia de cada lenguaje puede variar según el contexto específico y las necesidades del proyecto.

23 Áreas de Investigación sobre Datos

1. Aprendizaje Automático (Machine Learning)
2. Minería de Datos (Data Mining)
3. Análisis de Redes Sociales (Social Network Analysis)
4. Procesamiento del Lenguaje Natural (Natural Language Processing - NLP)
5. Visualización de Datos (Data Visualization)
6. Optimización de Decisiones (Decision Optimization)
7. Análisis Predictivo y Prescriptivo (Predictive and Prescriptive Analytics)
8. Análisis de Imágenes y Visión por Computadora (Image Analysis and Computer Vision)
9. Bioinformática y Ciencias de la Salud (Bioinformatics and Health Sciences)
10. Ciencias Sociales y Humanidades Digitales (Social Sciences and Digital Humanities)
11. Análisis de Texto (Text Analytics)
12. Minería de Texto (Text Mining)
13. Análisis de Sentimientos (Sentiment Analysis)
14. Reconocimiento de Patrones (Pattern Recognition)
15. Análisis de Secuencias Temporales (Time Series Analysis)
16. Análisis de Grafos (Graph Analysis)
17. Procesamiento de Señales (Signal Processing)
18. Análisis de Datos Espaciales (Spatial Data Analysis)
19. Análisis de Datos Genómicos (Genomic Data Analysis)
20. Análisis de Datos Biomédicos (Biomedical Data Analysis)
21. Análisis de Datos Ambientales (Environmental Data Analysis)
22. Análisis de Datos Financieros (Financial Data Analysis)
23. Análisis de Datos de Mercado (Market Data Analysis)
24. Análisis de Datos de Consumidores (Consumer Data Analysis)
25. Análisis de Datos de Recursos Humanos (Human Resources Data Analysis)

26. Análisis de Datos de Ingeniería (Engineering Data Analysis)
27. Análisis de Datos de Energía (Energy Data Analysis)
28. Análisis de Datos de Transporte (Transportation Data Analysis)
29. Análisis de Datos de Telecomunicaciones (Telecommunications Data Analysis)
30. Análisis de Datos de Medios (Media Data Analysis)

Estas áreas abarcan una amplia gama de disciplinas y aplicaciones, y continúan siendo objeto de investigación y desarrollo en la comunidad científica y académica.

Trabajo Práctico 2

Instalación y configuración del software a utilizar en la cátedra.

El trabajo práctico tiene como objetivos en primer lugar instalar el software que se utilizará en la cátedra, realizar prácticas de configuración del motor de base de datos y documentar resultados. El software a instalar es el siguiente:

- PostgreSQL: el motor de base de datos.
- PgAdmin: una plataforma de administración y desarrollo de bases de datos para PostgreSQL.
- DBeaver: una plataforma de administración y desarrollo de bases de datos para PostgreSQL. Es otra alternativa en caso de que existan inconvenientes de instalación de PgAdmin.
- Netbeans: un IDE de desarrollo para Java.

Para realizar las actividades del trabajo práctico se utilizará chatGPT. Se enumerarán actividades de configuración del motor de la base de datos y se provee el prompt que se deberá ingresar en chatGPT. Al resultado obtenido se deberá traducir a LaTeX, copiar y pegar en un documento en Overleaf. De esta manera, será posible generar un archivo pdf con la documentación que luego deberá ser entregada por medio de la plataforma Moodle en formato pdf. En clases se asistirá a cada una de estas actividades para realizarlas con el acompañamiento del docente. A continuación se enumeran las actividades propuestas:

Actividades de instalación:

1. Instalar el motor de base de datos Postgresql disponibles en el sitio Postgresql. Las instrucciones de instalación y configuración se encuentran disponible en Postgresql Docs.
2. Instalar la aplicación pgAdmin PgAdmin o DBeaver para gestionar la base de datos disponible en: DBeaver.
3. Instalar el IDE Netbeans para el lenguaje de programación Java disponible en: Netbeans.

Probar los siguientes comandos de línea de PostgreSQL.

1. **pg_ctl**: Controla el inicio y la parada del servidor PostgreSQL.
2. **psql**: Cliente de línea de comandos de PostgreSQL para ejecutar consultas y comandos SQL.
3. **pg_dump**: Realiza copias de seguridad de bases de datos PostgreSQL.
4. **pg_restore**: Restaura bases de datos PostgreSQL desde archivos de respaldo creados con pg_dump.
5. **createdb**: Crea una nueva base de datos PostgreSQL.
6. **dropdb**: Elimina una base de datos PostgreSQL.
7. **createuser**: Crea un nuevo usuario en PostgreSQL.

8. **dropuser**: Elimina un usuario de PostgreSQL.
9. **pg_upgrade**: Actualiza una versión anterior de PostgreSQL a una nueva versión.
10. **vacuumdb**: Analiza y optimiza bases de datos PostgreSQL para recuperar espacio en disco y mejorar el rendimiento.
11. **reindexdb**: Reconstruye los índices de una base de datos PostgreSQL.
12. **initdb**: Crea un nuevo clúster de base de datos PostgreSQL.
13. **pg_basebackup**: Realiza una copia de seguridad de una instancia PostgreSQL en caliente.
14. **pg_config**: Muestra la configuración de la instalación de PostgreSQL.
15. **pg_lsclusters**: Lista los clústeres de PostgreSQL instalados en el sistema.
16. **pg_resetwal**: Restablece los archivos de registro de transacciones de WAL.
17. **pg_rewind**: Retrocede un clúster PostgreSQL a un punto específico en el tiempo.
18. **pg_ctlcluster**: Controla el clúster de PostgreSQL (Linux).
19. **pg_createcluster**: Crea un nuevo clúster de PostgreSQL (Linux).
20. **pg_dropcluster**: Elimina un clúster de PostgreSQL (Linux).

Actividades de configuración del motor de base de datos: A continuación se enumeran un conjunto de tareas de configuración de la base de datos y sus respectivos prompts para ser utilizados en chatGPT y obtener instrucciones de como llevar a cabo cada tarea.

1. **Instalación de PostgreSQL:** explicar detalladamente los pasos de instalación de PostgreSQL en linux.
2. **Configuración del archivo postgresql.conf:** explicar detalladamente los parámetros que pueden modificarse en el archivo postgresql.conf.
3. **Configuración del archivo pg_hba.conf:** explicar detalladamente los parámetros que pueden modificarse en el archivo pg_hba.conf.
4. **Creación de roles de usuario y asignación de permisos:** mostrar diversos ejemplos detallados de creación de roles de usuario y asignación de permisos en postgresql.
5. **Configuración de autenticación SSL/TLS:** explicar detalladamente los parámetros que pueden modificarse para la autenticación SSL/TLS.
6. **Configuración de parámetros de rendimiento:** explicar detalladamente el archivo y los parámetros que pueden modificarse sobre el rendimiento de postgresql.
7. **Configuración de respaldo y recuperación:** describir el nombre del archivo de configuración de respaldo y recuperación de postgresql. Mostrar todos los parámetros de configuración de este archivo.

8. **Configuración de la replicación maestro-esclavo:** explicar detalladamente en que consiste la configuración de la replicación maestro-esclavo en postgresql. Explicar detalladamente en que consiste la configuración de la replicación maestro-esclavo en postgresql, como se llama el archivo de configuración y mostrar los parámetros del mismo.

9. **Configuración de la replicación síncrona:** explicar detalladamente en que consiste la configuración de la replicación síncrona en postgresql, el archivo de configuración y detallar los parámetros de configuración.

10. **Configuración de particionamiento de tablas:** explicar en que consiste la configuración de particionamiento de tablas en postgresql, cual es el archivo y parámetros de configuración.

11. **Configuración de índices y estadísticas:** explicar en que consiste la configuración de índices y estadísticas en postgresql, cual es el archivo de configuración y parámetros de configuración. Explicar detalladamente como se configuran los índices y estadísticas en postgresql.

12. **Configuración de políticas de retención de datos:** explicar detalladamente en que consiste la configuración de políticas de retención de datos en postgresql.

13. **Configuración de horarios de mantenimiento:** explicar detalladamente en que consiste la configuración de horarios de mantenimiento en postgresql. Mostrar ejemplos.

14. **Configuración de políticas de seguridad:** explicar en que consiste la configuración de políticas de seguridad, como se realiza y mostar ejemplos.

15. **Configuración de conexiones de aplicación y pool de conexiones:** explicar en que consiste en postgresql la configuración de conexiones de aplicación y pool de conexiones. Además, describir el nombre del archivo de configuración, parámetros de configuración y ejemplos.

16. **Configuración de logs y monitoreo:** explicar detalladamente como se realiza la configuración de logs y monitoreo en postgresql.

17. **Configuración de alertas y notificaciones:** explicar detalladamente como se realiza la configuración de alertas y notificaciones en postgresql.

18. **Configuración de extensiones adicionales:** explicar detalladamente como se realiza la configuración de extensiones adicionales de postgresql. Describir detalladamente todas las extensiones adicionales de postgresql.

19. **Configuración de políticas de almacenamiento:** describir detalladamente la configuración de políticas de almacenamiento en postgresql. describir ejemplos detallos de configuración de políticas de almacenamiento en postgresql.

20. **Configuración de parámetros de transacciones:** explicar en qué consiste la configuración de parámetros de transacciones en postgresql. Explicar como se configuran los parámetros de transacciones en postgresql.

21. **Configuración de parámetros de planificación de consultas:** explicar en qué consiste la configuración de parámetros de planificación de consultas en postgresql. Mostrar ejemplos detallados de configuración de planificación de consultas en postgresql.

22. **Configuración de almacenamiento en caché de consultas:** explicar en qué consiste la configuración de almacenamiento en caché de consultas en postgresql. Mostrar ejemplos detallados configuración de almacenamiento en caché de consultas en postgresql.

23. **Configuración de almacenamiento de blobs y archivos grandes:** explicar en que consiste la configuración de almacenamiento de blobs y archivos grandes en postgresql. Mostrar ejemplos detallados de configuración de almacenamiento de blobs y archivos grandes en postgresql.

24. **Configuración de parámetros de bloqueo:** explicar en que consiste la configuración de parámetros de bloqueo en postgresql. Mostrar ejemplos detallados de la configuración de parámetros de bloqueo en postgresql.

25. **Configuración de parámetros de compresión:** explicar en que consiste la configuración de parámetros de compresión en postgresql. Mostrar ejemplos detallados de la configuración de parámetros de compresión en postgresql.

26. **Configuración de parámetros de planificación de respaldo:** explicar en que consiste la configuración de parámetros de planificación de respaldo en postgresql. Mostrar ejemplos detallados de parámetros de planificación de respaldo en postgresql.

27. **Configuración de parámetros de auditoría:** explicar en que consiste la configuración de parámetros de auditoría en postgresql. Mostrar ejemplos detallados de parámetros de configuración de parámetros de auditoría en postgresql.

28. **Configuración de políticas de encriptación:** explicar en que consiste la configuración de políticas de encriptación en postgresql. Mostrar ejemplos detallados de parámetros de configuración de políticas de encriptación en postgresql.

29. **Configuración de políticas de archivado de datos:** explicar en que consiste la configuración de políticas de archivado de datos en postgresql. Mostrar ejemplos detallados de parámetros de configuración de políticas de políticas de archivado de datos en postgresql.

30. **Configuración de parámetros de gestión de recursos:** explicar en que consiste la configuración de parámetros de gestión de recursos en postgresql. Mostrar ejemplos detallados de parámetros de parámetros de gestión de recursos en postgresql.

Trabajo Práctico 3

Configuración de acceso a una base de datos.

1. Crea un nuevo rol de usuario llamado "analista" con la contraseña "analista123" y permisos para realizar consultas en todas las tablas de la base de datos.
2. Concede al rol "analista" permisos de escritura en la tabla "ventas" de la base de datos.
3. Revoca el permiso de actualización en la tabla "clientes" del rol "analista".
4. Elimina el rol de usuario "analista" de la base de datos.
5. Crea un nuevo rol de superusuario llamado "admin_db" con la contraseña "admin123".
6. Limita el acceso del rol "admin_db" a la base de datos solo desde la dirección IP 192.168.1.100.
7. Cambia la contraseña del rol "admin_db" a "nuevacontraseña123".
8. Crea un nuevo rol de solo lectura llamado "lector_datos" con la contraseña "lectura123" y permisos para realizar consultas en todas las tablas de la base de datos.
9. Revoca el permiso de eliminación en la tabla "empleados" del rol "lector_datos".
10. Elimina el rol de usuario "lector_datos" de la base de datos.
11. Crea un nuevo rol de escritura llamado "escritor_datos" con la contraseña "escritura123" y permisos para realizar inserciones y actualizaciones en todas las tablas de la base de datos.
12. Revoca el permiso de actualización en la tabla "productos" del rol "escritor_datos".
13. Elimina el rol de usuario "escritor_datos" de la base de datos.
14. Crea un nuevo rol de solo lectura para una tabla específica llamado "lector_ventas" con la contraseña "lectura123" y permisos para realizar consultas en la tabla "ventas" de la base de datos.
15. Revoca todos los permisos en la tabla "clientes" del rol "lector_ventas".
16. Elimina el rol de usuario "lector_ventas" de la base de datos.
17. Crea un nuevo rol de solo lectura para una función específica llamado "lector_funciones" con la contraseña "lectura123".
18. Concede al rol "lector_funciones" permisos para ejecutar la función "calcular_ingresos_mensuales".
19. Revoca todos los permisos en la función "calcular_ganancias_anuales" del rol "lector_funciones".
20. Elimina el rol de usuario "lector_funciones" de la base de datos.
21. Crea un nuevo rol de solo lectura llamado "consulta_ventas" con la contraseña "ventas123" y permisos para realizar consultas en las tablas "ventas" y "productos".

22. Concede al rol "consulta_ventas" permisos de lectura en las tablas "clientes" y "empleados".
23. Revoca todos los permisos en la tabla "empleados" del rol "consulta_ventas".
24. Elimina el rol de usuario "consulta_ventas" de la base de datos.
25. Crea un nuevo rol de escritura llamado "escritura_productos" con la contraseña "productos123" y permisos para realizar inserciones y actualizaciones en la tabla "productos".
26. Concede al rol "escritura_productos" permisos de lectura en las tablas "ventas" y "empleados".
27. Revoca todos los permisos en la tabla "ventas" del rol "escritura_productos".
28. Elimina el rol de usuario "escritura_productos" de la base de datos.
29. Crea un nuevo rol de superusuario llamado "admin_super" con la contraseña "super123".
30. Limita el acceso del rol "admin_super" a la base de datos solo desde la dirección IP 192.168.1.200.

Trabajo Práctico 4

SQL - Lenguaje de Definición de Datos.

En este trabajo práctico se describen ejemplos de bases de datos presentando las entidades y relaciones forman la estructura básica de una base de datos.

1. Analizar cada caso e identificar errores si existieran, mejorar el modelo en caso de ser necesario.
2. De acuerdo a la estructura que se presenta en cada caso implementar la correspondiente base de datos incluyendo las restricciones de claves.
3. Documentar el ejercicio en un documento en Overleaf incluyendo las instrucciones SQL que se utilizaron al realizar el ejercicio.

Comandos para la gestión de una base de datos.

A continuación se describen los comandos para gestionar una base de datos. Aplicar cada comando a los casos de estudio que se presentan en este trabajo práctico.

1. **CREATE DATABASE**: Este comando se utiliza para crear una nueva base de datos en PostgreSQL. Al ejecutar este comando, se especifica el nombre de la base de datos que se desea crear.
2. **DROP DATABASE**: El comando DROP DATABASE se utiliza para eliminar una base de datos existente de PostgreSQL. Al ejecutar este comando, se elimina la base de datos especificada, incluyendo todas las tablas, índices, vistas y otros objetos relacionados.
3. **ALTER DATABASE**: Este comando permite realizar modificaciones en una base de datos existente. Puedes cambiar el propietario de la base de datos, entre otras configuraciones.
4. **RENAME DATABASE**: Con este comando, puedes cambiar el nombre de una base de datos existente en PostgreSQL.
5. **CREATE TABLE**: Utilizado para crear una nueva tabla en la base de datos. Al ejecutar este comando, se especifica el nombre de la tabla y las columnas que contendrá, junto con los tipos de datos y restricciones.
6. **ALTER TABLE**: Se utiliza para modificar la estructura de una tabla existente. Puedes agregar, eliminar o modificar columnas, así como también aplicar restricciones adicionales.
7. **DROP TABLE**: Con este comando, puedes eliminar una tabla existente de la base de datos, junto con todos sus datos y metadatos asociados.
8. **TRUNCATE TABLE**: Utilizado para eliminar todos los registros de una tabla, pero conservando la estructura de la tabla misma.
9. **INSERT INTO**: Se utiliza para agregar nuevos registros a una tabla existente en la base de datos.

10. **SELECT**: Este comando es esencial para recuperar datos de una o varias tablas en la base de datos. Puedes especificar las columnas que deseas recuperar y aplicar condiciones para filtrar los resultados.

11. **UPDATE**: Utilizado para modificar registros existentes en una tabla. Puedes cambiar los valores de una o varias columnas de acuerdo a una condición específica.

12. **DELETE**: Este comando se utiliza para eliminar registros de una tabla en función de una condición especificada.

13. **CREATE INDEX**: Se utiliza para crear un índice en una o varias columnas de una tabla. Los índices se utilizan para mejorar el rendimiento de las consultas SELECT al permitir búsquedas más rápidas.

14. **DROP INDEX**: Con este comando, puedes eliminar un índice existente en la base de datos.

Caso 1: Gestión Universitaria

A continuación se describen las entidades con sus respectivos atributos y descripción.

Estudiante:

- Atributos: Nombre, Apellido, Fecha de nacimiento, Dirección, Teléfono, Email, ID de estudiante.
- Descripción: Representa a los estudiantes matriculados en la universidad.

Profesor:

- Atributos: Nombre, Apellido, Especialidad, Teléfono, Email, ID de profesor.
- Descripción: Representa a los profesores que imparten clases en la universidad.

Curso:

- Atributos: Nombre del curso, Créditos, Horas semanales, Departamento, ID de curso.
- Descripción: Representa los cursos ofrecidos por la universidad.

Departamento:

- Atributos: Nombre del departamento, Ubicación, Teléfono, ID de departamento.
- Descripción: Representa los departamentos académicos de la universidad.

Matrícula

- Atributos: ID de estudiante, ID de curso, Año académico, Semestre, ID de matrícula.
- Descripción: Relaciona a los estudiantes con los cursos en los que están matriculados.

Calificación:

- Atributos: ID de estudiante, ID de curso, Calificación, ID de calificación.
- Descripción: Almacena las calificaciones de los estudiantes en los cursos.

Horario:

- Atributos: ID de curso, Día de la semana, Hora de inicio, Hora de fin, ID de horario.
- Descripción: Define los horarios de clase para cada curso.

Programa Académico:

- Atributos: Nombre del programa, Duración, Tipo, ID de programa.
- Descripción: Representa los programas académicos ofrecidos por la universidad, como pregrado, posgrado, etc.

Asignatura:

- Atributos: Nombre de la asignatura, Descripción, Créditos, ID de programa, ID de asignatura.
- Descripción: Representa las asignaturas que forman parte de los programas académicos.

Aula:

- Atributos: Número de aula, Capacidad, Edificio, ID de aula.
- Descripción: Representa las aulas disponibles para las clases.

A continuación se describen las relaciones entre entidades.

Relaciones:

- Estudiante-Matrícula: relación muchos a muchos entre Estudiante y Curso a través de Matrícula. Un estudiante puede estar matriculado en varios cursos y un curso puede tener varios estudiantes matriculados.
- Profesor-Curso: relación uno a muchos entre Profesor y Curso. Un profesor puede enseñar varios cursos, pero un curso solo tiene un profesor.
- Curso-Matrícula: relación uno a muchos entre Curso y Matrícula. Un curso puede tener varias matrículas, pero una matrícula pertenece solo a un curso.
- Curso-Calificación: Relación uno a muchos entre Curso y Calificación. Un curso puede tener varias calificaciones, pero una calificación pertenece solo a un curso.
- Curso-Horario: relación uno a muchos entre Curso y Horario. Un curso puede tener varios horarios, pero un horario pertenece solo a un curso.

- Asignatura-Programa Académico: relación uno a muchos entre Asignatura y Programa Académico. Una asignatura puede pertenecer a un solo programa académico, pero un programa académico puede tener varias asignaturas.

Caso 2: Agencia de Remises

A continuación se describen las entidades con sus respectivos atributos y descripción.

Remisero

- **Atributos:**
  - Nombre: El nombre del remisero.
  - Apellido: El apellido del remisero.
  - DNI: El número de documento de identidad del remisero.
  - Teléfono: El número de teléfono del remisero.
  - Dirección: La dirección del remisero.
  - Licencia de conducir: El número de licencia de conducir del remisero.
  - Vehículo asignado: El vehículo asignado al remisero.
- **Descripción:** Esta entidad representa a los remiseros que trabajan para la empresa.

Vehículo

- **Atributos:**
  - Marca: La marca del vehículo.
  - Modelo: El modelo del vehículo.
  - Año: El año de fabricación del vehículo.
  - Patente: La patente del vehículo.
  - Color: El color del vehículo.
  - Tipo: El tipo de vehículo (sedán, utilitario, etc.).
- **Descripción:** Esta entidad representa a los vehículos utilizados por los remiseros para prestar el servicio.

Cliente

- **Atributos:**
  - Nombre: El nombre del cliente.
  - Apellido: El apellido del cliente.
  - DNI: El número de documento de identidad del cliente.
  - Teléfono: El número de teléfono del cliente.
  - Dirección: La dirección del cliente.
- **Descripción:** Esta entidad representa a los clientes que solicitan el servicio de remis.

Viaje

- **Atributos:**
  - Fecha y hora de inicio: La fecha y hora en que comenzó el viaje.
  - Fecha y hora de fin: La fecha y hora en que finalizó el viaje.
  - Origen: El lugar de origen del viaje.
  - Destino: El lugar de destino del viaje.
  - Tarifa: La tarifa cobrada por el viaje.
- **Descripción:** Esta entidad representa los viajes realizados por los remiseros.

A continuación se describen las relaciones entre entidades.

Remisero - Vehículo

- **Descripción:** Relación uno a uno entre Remisero y Vehículo. Cada remisero tiene asignado un vehículo y cada vehículo está asignado a un remisero.

Cliente - Viaje

- **Descripción:** Relación uno a muchos entre Cliente y Viaje. Un cliente puede solicitar varios viajes, pero un viaje está asociado a un solo cliente.

Remisero - Viaje

- **Descripción:** Relación uno a muchos entre Remisero y Viaje. Un remisero puede realizar varios viajes, pero un viaje es realizado por un solo remisero.

Caso 3: Agencia de venta de autos

A continuación se describen las entidades con sus respectivos atributos y descripción.

Entidades

1. **Auto**

   [label=–,leftmargin=1cm]

   - **Atributos:**
     - **ID_Auto**: Identificador único del auto.
     - **Marca**: Marca del auto.
     - **Modelo**: Modelo del auto.
     - **Año**: Año de fabricación del auto.
     - **Precio**: Precio de venta del auto.
     - **Color**: Color del auto.
   - **Descripción:** Representa los autos disponibles en la agencia para la venta.

2. **Cliente**

   - **Atributos:**
     - **ID_Cliente**: Identificador único del cliente.
     - **Nombre**: Nombre del cliente.
     - **Apellido**: Apellido del cliente.
     - **Dirección**: Dirección del cliente.
     - **Teléfono**: Número de teléfono del cliente.
   - **Descripción:** Representa a los clientes interesados en comprar autos en la agencia.

3. **Empleado**

   - **Atributos:**
     - **ID_Empleado**: Identificador único del empleado.
     - **Nombre**: Nombre del empleado.
     - **Apellido**: Apellido del empleado.
     - **Cargo**: Cargo del empleado en la agencia.
   - **Descripción:** Empleados que trabajan en la agencia de venta de autos.

4. **Venta**

   - **Atributos:**
     - **ID_Venta**: Identificador único de la venta.
     - **ID_Auto**: Identificador del auto vendido.
     - **ID_Cliente**: Identificador del cliente que realizó la compra.
     - **ID_Empleado**: Identificador del empleado que realizó la venta.

  - **Fecha**: Fecha de la venta.
  - **Total**: Total de la venta.
- **Descripción:** Registra las ventas realizadas en la agencia, incluyendo el auto vendido, el cliente que lo adquirió, el empleado que realizó la venta y el monto total.

A continuación se describen las relaciones entre entidades.

- **Relación Auto-Cliente (Interés de Compra):** Un cliente puede estar interesado en comprar uno o varios autos de la agencia. Esta relación se establece cuando un cliente muestra interés en un auto específico.
- **Relación Venta-Auto:** Un auto puede ser vendido en una o varias ventas realizadas en la agencia. Esta relación registra qué autos fueron vendidos en cada transacción.
- **Relación Venta-Cliente:** Un cliente puede realizar una o varias compras de autos en la agencia. Esta relación registra qué clientes realizaron cada venta.
- **Relación Venta-Empleado:** Un empleado puede ser responsable de realizar una o varias ventas en la agencia. Esta relación registra qué empleados realizaron cada venta.

Comandos de Privilegios en PostgreSQL

Para alguna de las bases de datos creadas anteriormente aplicar cada uno de los comandos. Documentar cada uno de los ejercicios.

Privilegios de Base de Datos:

- **GRANT:** Concede privilegios sobre una base de datos a un rol.
- **REVOKE:** Revoca privilegios sobre una base de datos a un rol.

Privilegios de Tablas:

- **GRANT SELECT:** Concede el privilegio de lectura sobre una tabla a un rol.
- **GRANT INSERT:** Concede el privilegio de inserción de filas en una tabla a un rol.
- **GRANT UPDATE:** Concede el privilegio de actualización de filas en una tabla a un rol.
- **GRANT DELETE:** Concede el privilegio de eliminación de filas en una tabla a un rol.
- **REVOKE SELECT:** Revoca el privilegio de lectura sobre una tabla a un rol.
- **REVOKE INSERT:** Revoca el privilegio de inserción de filas en una tabla a un rol.
- **REVOKE UPDATE:** Revoca el privilegio de actualización de filas en una tabla a un rol.
- **REVOKE DELETE:** Revoca el privilegio de eliminación de filas en una tabla a un rol.
- **GRANT ALL:** Concede todos los privilegios sobre una tabla a un rol.
- **REVOKE ALL:** Revoca todos los privilegios sobre una tabla a un rol.

Privilegios de Columnas:

- **GRANT SELECT (columna):** Concede el privilegio de lectura sobre una columna específica de una tabla a un rol.
- **GRANT INSERT (columna):** Concede el privilegio de inserción de valores en una columna específica de una tabla a un rol.
- **GRANT UPDATE (columna):** Concede el privilegio de actualización de valores en una columna específica de una tabla a un rol.
- **REVOKE SELECT (columna):** Revoca el privilegio de lectura sobre una columna específica de una tabla a un rol.
- **REVOKE INSERT (columna):** Revoca el privilegio de inserción de valores en una columna específica de una tabla a un rol.
- **REVOKE UPDATE (columna):** Revoca el privilegio de actualización de valores en una columna específica de una tabla a un rol.
- **GRANT ALL (columna):** Concede todos los privilegios sobre una columna específica de una tabla a un rol.
- **REVOKE ALL (columna):** Revoca todos los privilegios sobre una columna específica de una tabla a un rol.

Privilegios de Base de Datos:

- `GRANT CREATE`: Concede el privilegio de crear objetos en la base de datos.
- `REVOKE CREATE`: Revoca el privilegio de crear objetos en la base de datos.
- `GRANT TEMPORARY`: Concede el privilegio de crear tablas temporales en la base de datos.
- `REVOKE TEMPORARY`: Revoca el privilegio de crear tablas temporales en la base de datos.

Privilegios de Esquema:

- `GRANT USAGE`: Concede el privilegio de usar un esquema en la base de datos.
- `REVOKE USAGE`: Revoca el privilegio de usar un esquema en la base de datos.
- `GRANT CREATE ON SCHEMA`: Concede el privilegio de crear objetos en un esquema.
- `REVOKE CREATE ON SCHEMA`: Revoca el privilegio de crear objetos en un esquema.

Privilegios de Secuencia:

- `GRANT USAGE ON SEQUENCE`: Concede el privilegio de usar una secuencia.
- `REVOKE USAGE ON SEQUENCE`: Revoca el privilegio de usar una secuencia.
- `GRANT SELECT ON SEQUENCE`: Concede el privilegio de seleccionar valores de una secuencia.
- `REVOKE SELECT ON SEQUENCE`: Revoca el privilegio de seleccionar valores de una secuencia.

Privilegios de Función:

- `GRANT EXECUTE`: Concede el privilegio de ejecutar una función.
- `REVOKE EXECUTE`: Revoca el privilegio de ejecutar una función.

Privilegios de Tipos de Datos:

- `GRANT USAGE ON TYPE`: Concede el privilegio de usar un tipo de dato definido.
- `REVOKE USAGE ON TYPE`: Revoca el privilegio de usar un tipo de dato definido.

Privilegios de Triggers:

- `GRANT TRIGGER`: Concede el privilegio de crear un trigger en una tabla.
- `REVOKE TRIGGER`: Revoca el privilegio de crear un trigger en una tabla.

Trabajo Práctico 5

SQL - Lenguaje de Manipulación de Datos.

En este trabajo práctico se utilizará una base de datos modelo con datos que se puede acceder a través del siguiente link: dvd rental. Investigar formas de importar la base de datos, aplicar alguna de las formas identificadas y a continuación ejecutar las instrucciones que se encuentran el este trabajo práctico. Documentar las instrucciones utilizadas como parte de la entrega que deberá realizar de acuerdo a las condiciones establecidas en la cátedra.

Instrucciones SQL INSERT soportadas por PostgreSQL

PostgreSQL admite varias formas de la instrucción SQL INSERT para insertar datos en una tabla. A continuación se presenta una descripción de cada una de ellas:

1. **INSERT INTO ... VALUES**: Esta es la forma más común de la instrucción INSERT, donde se especifican los valores a insertar directamente en la tabla en el mismo orden que las columnas.

```
INSERT INTO nombre_tabla (columna1, columna2, ...)
VALUES (valor1, valor2, ...);
```

2. **INSERT INTO ... SELECT**: Permite insertar datos en una tabla a partir de los resultados de una consulta SELECT.

```
INSERT INTO nombre_tabla (columna1, columna2, ...)
SELECT valor1, valor2, ...
FROM otra_tabla
WHERE condición;
```

3. **INSERT INTO ... DEFAULT VALUES**: Inserta un nuevo registro con los valores predeterminados o nulos para todas las columnas.

```
INSERT INTO nombre_tabla DEFAULT VALUES;
```

4. **INSERT INTO ... ON CONFLICT**: PostgreSQL admite la cláusula ON CONFLICT para manejar conflictos de restricción de clave única. Puedes especificar acciones como UPDATE o IGNORE en caso de conflicto.

```
INSERT INTO nombre_tabla (columna1, columna2, ...)
VALUES (valor1, valor2, ...)
ON CONFLICT (columna_conflicto)
DO UPDATE SET columna1 = EXCLUDED.columna1, columna2 = EXCLUDED.columna2;
```

5. **INSERT INTO ... RETURNING**: Esta forma de INSERT permite recuperar los valores insertados o calcular valores derivados después de la inserción.

```
INSERT INTO nombre_tabla (columna1, columna2, ...)
VALUES (valor1, valor2, ...)
RETURNING columna_generada;
```

Instrucciones SQL UPDATE soportadas por PostgreSQL

PostgreSQL admite varias formas de la instrucción SQL UPDATE para modificar datos en una tabla. A continuación se presenta una descripción de cada una de ellas:

1. **UPDATE con SET**: Esta es la forma más común de la instrucción UPDATE, donde se especifican las columnas que se van a modificar y los nuevos valores para esas columnas.

```
UPDATE nombre_tabla
SET columna1 = nuevo_valor1, columna2 = nuevo_valor2, ...
WHERE condición;
```

2. **UPDATE con FROM**: Permite actualizar una tabla utilizando los resultados de una consulta SELECT en otra tabla. Se puede utilizar para actualizar columnas basadas en valores de otra tabla.

```
UPDATE tabla_destino
SET columna1 = tabla_fuente.valor1, columna2 = tabla_fuente.valor2, ...
FROM tabla_fuente
WHERE tabla_destino.columna = tabla_fuente.columna;
```

3. **UPDATE con USING**: Similar a la opción con FROM, pero se puede utilizar para hacer referencia a múltiples tablas en la consulta SELECT de origen.

```
UPDATE tabla_destino
SET columna1 = tabla_fuente.valor1, columna2 = tabla_fuente.valor2, ...
FROM tabla_fuente1, tabla_fuente2
WHERE tabla_destino.columna = tabla_fuente1.columna
AND tabla_destino.columna2 = tabla_fuente2.columna2;
```

4. **UPDATE con RETURNING**: Permite recuperar los valores actualizados o calcular valores derivados después de la actualización.

```
UPDATE nombre_tabla
SET columna1 = nuevo_valor1, columna2 = nuevo_valor2, ...
WHERE condición
RETURNING columna_actualizada;
```

5. **UPDATE con JOINS**: Permite actualizar una tabla utilizando unir múltiples tablas en la cláusula FROM, lo que puede ser útil para realizar actualizaciones basadas en relaciones complejas entre tablas.

```
UPDATE tabla_destino
SET columna1 = nuevo_valor1, columna2 = nuevo_valor2, ...
FROM tabla_fuente
WHERE tabla_destino.columna = tabla_fuente.columna;
```

Instrucciones SQL DELETE soportadas por PostgreSQL

PostgreSQL admite varias formas de la instrucción SQL DELETE para eliminar datos de una tabla. A continuación se presenta una descripción de cada una de ellas:

1. **DELETE con WHERE**: Esta es la forma más común de la instrucción DELETE, donde se especifica una condición para eliminar solo los registros que cumplan con esa condición.

```
DELETE FROM nombre_tabla
WHERE condición;
```

2. **DELETE sin WHERE**: Si no se especifica una condición, se eliminarán todos los registros de la tabla.

```
DELETE FROM nombre_tabla;
```

3. **DELETE con USING**: Permite eliminar registros de una tabla basados en los resultados de una consulta SELECT en otra tabla.

```
DELETE FROM tabla_destino
USING tabla_fuente
WHERE tabla_destino.columna = tabla_fuente.columna
AND condición;
```

4. **DELETE con RETURNING**: Permite recuperar los valores eliminados o calcular valores derivados después de la eliminación.

```
DELETE FROM nombre_tabla
WHERE condición
RETURNING columna_eliminada;
```

Instrucción SQL SELECT en PostgreSQL

PostgreSQL admite varias formas de la instrucción SQL SELECT para recuperar datos de una o varias tablas que se presentan a continuación.

1. **Selección Básica**:

   ```
   SELECT * FROM nombre_tabla;
   ```

   Permite seleccionar todas las columnas de una tabla.

2. **Selección con Columnas Específicas**:

   ```
   SELECT columna1, columna2 FROM nombre_tabla;
   ```

   Permite seleccionar columnas específicas de una tabla en lugar de todas las columnas.

3. **Selección con Alias de Columnas**:

   ```
   SELECT columna1 AS alias1, columna2 AS alias2 FROM nombre_tabla;
   ```

   Permite asignar alias a las columnas seleccionadas.

4. **Selección con WHERE**:

   ```
   SELECT * FROM nombre_tabla WHERE condición;
   ```

   Permite filtrar los resultados de la consulta utilizando una condición.

5. **Selección con ORDER BY**:

   ```
   SELECT * FROM nombre_tabla ORDER BY columna ASC/DESC;
   ```

   Permite ordenar los resultados de la consulta en función de una o más columnas.

6. **Selección con LIMIT**:

```
SELECT * FROM nombre_tabla LIMIT cantidad;
```

Limita el número de filas devueltas por la consulta.

7. **Selección con OFFSET**:

```
SELECT * FROM nombre_tabla OFFSET cantidad;
```

Permite omitir un número específico de filas del resultado de la consulta.

8. **Selección con DISTINCT**:

```
SELECT DISTINCT columna FROM nombre_tabla;
```

Elimina filas duplicadas del resultado de la consulta.

9. **Selección con Funciones de Agregación**:

```
SELECT COUNT(columna) FROM nombre_tabla;
```

Permite realizar operaciones de agregación como COUNT, SUM, AVG, MIN, MAX, etc.

10. **Selección con JOIN**:

```
SELECT * FROM tabla1 INNER JOIN tabla2 ON tabla1.columna = tabla2.columna;
```

Combina datos de dos o más tablas en base a una condición de unión.

11. **Selección con GROUP BY**:

```
SELECT columna, COUNT(*) FROM nombre_tabla GROUP BY columna;
```

Agrupa filas que tienen los mismos valores en una o más columnas.

12. **Selección con HAVING**:

```
SELECT columna, COUNT(*) FROM nombre_tabla GROUP BY columna
HAVING COUNT(*) > valor;
```

Permite filtrar los resultados de la consulta después de haber sido agrupados por la cláusula GROUP BY.

13. **Selección con Subconsultas (Subqueries)**:

```
SELECT * FROM nombre_tabla WHERE columna IN (SELECT columna FROM otra_tabla);
```

Permite incluir una consulta dentro de otra consulta.

14. **Operadores de Conjuntos (Set Operators)**:

- **UNION**:

```
SELECT columna FROM tabla1
UNION
SELECT columna FROM tabla2;
```

Combina los resultados de dos consultas y elimina duplicados.

- **UNION ALL**:

```
SELECT columna FROM tabla1
UNION ALL
SELECT columna FROM tabla2;
```

Combina los resultados de dos consultas sin eliminar duplicados.

- **INTERSECT**:

```
SELECT columna FROM tabla1
INTERSECT
SELECT columna FROM tabla2;
```

Devuelve los registros presentes en ambos resultados.

- **EXCEPT (MINUS)**:

```
SELECT columna FROM tabla1
EXCEPT
SELECT columna FROM tabla2;
```

Devuelve los registros de la primera consulta que no están presentes en la segunda.

Trabajo Práctico 6

Programación en una base de datos.

STORED PROCEDURES

A continuación se presentan tres ejemplos de stored procedures que pueden utilizarse de referencia para resolver los ejercicios del trabajo práctico.

Procedimiento para obtener el nombre completo de un usuario según su ID

Este procedimiento toma el ID de un usuario como entrada y devuelve su nombre completo concatenando el nombre y el apellido desde la tabla `usuarios`.

```
CREATE OR REPLACE PROCEDURE obtener_nombre_completo(
    IN p_usuario_id INTEGER,
    OUT p_nombre_completo VARCHAR
)
LANGUAGE plpgsql
AS $$
BEGIN
    SELECT CONCAT(nombre, ' ', apellido) INTO p_nombre_completo
    FROM usuarios
    WHERE id = p_usuario_id;
END;
$$;
```

Para llamar a este procedimiento, se debe ejecutar las siguientes instrucciones:

```
CALL obtener_nombre_completo(1, nombre_completo);
SELECT nombre_completo;
```

Procedimiento para actualizar la edad de un usuario

Este procedimiento toma el ID de un usuario y su nueva edad como entrada, y actualiza la edad del usuario en la tabla `usuarios`.

```
CREATE OR REPLACE PROCEDURE actualizar_edad_usuario(
    IN p_usuario_id INTEGER,
    IN p_nueva_edad INTEGER
)
LANGUAGE plpgsql
AS $$
BEGIN
```

```
    UPDATE usuarios
    SET edad = p_nueva_edad
    WHERE id = p_usuario_id;
    COMMIT;
END;
$$;
```

Para llamar a este procedimiento, puedes hacer lo siguiente:

```
CALL actualizar_edad_usuario(1, 35);
```

Procedimiento para eliminar un usuario y sus registros relacionados

Este procedimiento toma el ID de un usuario como entrada y elimina al usuario de la tabla `usuarios`, así como también elimina todos sus registros relacionados en otras tablas.

```
CREATE OR REPLACE PROCEDURE eliminar_usuario(
    IN p_usuario_id INTEGER
)
LANGUAGE plpgsql
AS $$
BEGIN
    DELETE FROM registros WHERE usuario_id = p_usuario_id;
    DELETE FROM comentarios WHERE usuario_id = p_usuario_id;
    DELETE FROM usuarios WHERE id = p_usuario_id;
    COMMIT;
END;
$$;
```

Para llamar a este procedimiento, puedes hacer lo siguiente:

```
CALL eliminar_usuario(1);
```

Ejercicios sobre Stored Procedures

A continuación se presentan los enunciados de los ejercicios a resolver y documentar como parte de la entrega de documentación del trabajo práctico.

Ejercicio 1: Procedimiento de Actualización

Crea un procedimiento almacenado llamado **actualizar_nombre_producto** que acepte dos parámetros de entrada: el ID del producto y el nuevo nombre que se le desea asignar. Este procedimiento debe actualizar el nombre del producto en la tabla **productos** según el ID proporcionado. Luego, llama al procedimiento para actualizar el nombre del producto con ID 1001 a "Nuevo Nombre".

Ejercicio 2: Procedimiento de Eliminación

Escribe un procedimiento almacenado llamado **eliminar_pedidos_antiguos** que no reciba parámetros de entrada. Este procedimiento debe eliminar todos los pedidos de la tabla **pedidos** que tengan una fecha de creación anterior al 1 de enero de 2022. Luego, ejecuta el procedimiento para eliminar los pedidos antiguos.

Ejercicio 3: Procedimiento de Consulta

Crea un procedimiento almacenado llamado **obtener_total_pedidos_cliente** que acepte el ID de un cliente como parámetro de entrada y devuelva el total de pedidos realizados por ese cliente. El procedimiento debe consultar la tabla **pedidos** y sumar el monto total de todos los pedidos asociados al cliente proporcionado. Luego, llama al procedimiento para obtener el total de pedidos del cliente con ID 2001.

Funciones

A continuación se presentan tres ejemplos de funciones que pueden utilizarse de referencia para resolver los ejercicios del trabajo práctico.

Función para Calcular el Área de un Círculo

Esta función toma el radio de un círculo como entrada y devuelve el área del círculo calculada usando la fórmula πr^2.

```
CREATE OR REPLACE FUNCTION calcular_area_circulo(radio FLOAT)
RETURNS FLOAT AS
$$
DECLARE
    area FLOAT;
BEGIN
    area := pi() * radio * radio;
    RETURN area;
END;
$$
LANGUAGE plpgsql;
```

Función para Concatenar Dos Cadenas de Texto

Esta función toma dos cadenas de texto como entrada y devuelve una cadena que contiene la concatenación de ambas cadenas, separadas por un espacio.

```
CREATE OR REPLACE FUNCTION concatenar_texto(texto1 TEXT, texto2 TEXT)
RETURNS TEXT AS
$$
```

```
DECLARE
    resultado TEXT;
BEGIN
    resultado := texto1 || ' ' || texto2;
    RETURN resultado;
END;
$$
LANGUAGE plpgsql;
```

Función para Calcular el Promedio de una Lista de Números

Esta función toma una lista de números como entrada y devuelve el promedio de esos números.

```
CREATE OR REPLACE FUNCTION calcular_promedio(numeros INT[])
RETURNS FLOAT AS
$$
DECLARE
    suma FLOAT := 0;
    cantidad INT := 0;
    promedio FLOAT;
BEGIN
    FOREACH num IN ARRAY numeros
    LOOP
        suma := suma + num;
        cantidad := cantidad + 1;
    END LOOP;

    IF cantidad > 0 THEN
        promedio := suma / cantidad;
    ELSE
        promedio := 0;
    END IF;

    RETURN promedio;
END;
$$
LANGUAGE plpgsql;
```

Ejercicios sobre Funciones

A continuación se presentan los enunciados de los ejercicios a resolver y documentar como parte de la entrega de documentación del trabajo práctico.

23.1 Ejercicio 1: Función para Calcular el Descuento

Crea una función llamada **calcular_descuento** que acepte dos parámetros de entrada: el precio original de un producto y el porcentaje de descuento a aplicar. La función debe calcular el monto del descuento y devolver el precio final del producto después de aplicar el descuento. Luego, llama a la función con un precio original de $100 y un descuento del 20%.

Ejercicio 2: Función para Convertir Temperatura

Escribe una función llamada **convertir_temperatura** que convierta la temperatura de grados Celsius a grados Fahrenheit. La función debe aceptar un parámetro de entrada que represente la temperatura en grados Celsius y devolver la temperatura equivalente en grados Fahrenheit. Luego, llama a la función con una temperatura de 25 grados Celsius.

Ejercicio 3: Función para Validar Correo Electrónico

Crea una función llamada **validar_correo** que verifique si una dirección de correo electrónico es válida. La función debe aceptar un parámetro de entrada que represente la dirección de correo electrónico y devolver verdadero si la dirección es válida y falso si no lo es. Luego, llama a la función con diferentes direcciones de correo electrónico para probar su funcionamiento.

Triggers

A continuación se presentan tres ejemplos de triggers que pueden utilizarse de referencia para resolver los ejercicios del trabajo práctico.

Trigger para Auditar Cambios en una Tabla

Este trigger registra cada vez que se inserta, actualiza o elimina un registro en una tabla de auditoría. Guarda información sobre el usuario que realizó la acción, la fecha y hora de la acción, el tipo de operación (inserción, actualización o eliminación) y los valores antiguos y nuevos afectados por la operación.

```
CREATE OR REPLACE FUNCTION auditar_tabla()
RETURNS TRIGGER AS
$$
BEGIN
    IF TG_OP = 'INSERT' THEN
        INSERT INTO auditoria (accion, usuario, fecha, nuevo_valor)
        VALUES ('INSERT', current_user, current_timestamp, NEW.*);
    ELSIF TG_OP = 'UPDATE' THEN
        INSERT INTO auditoria (accion, usuario, fecha, antiguo_valor, nuevo_valor)
        VALUES ('UPDATE', current_user, current_timestamp, OLD.*, NEW.*);
```

```
    ELSIF TG_OP = 'DELETE' THEN
        INSERT INTO auditoria (accion, usuario, fecha, antiguo_valor)
        VALUES ('DELETE', current_user, current_timestamp, OLD.*);
    END IF;
    RETURN NULL;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER trigger_auditar_tabla
AFTER INSERT OR UPDATE OR DELETE ON tabla
FOR EACH ROW
EXECUTE FUNCTION auditar_tabla();
```

Trigger para Validar Datos antes de la Inserción

Este trigger asegura que no se inserten registros en una tabla si la suma de dos columnas supera un cierto límite.

```
CREATE OR REPLACE FUNCTION validar_suma()
RETURNS TRIGGER AS
$$
BEGIN
    IF NEW.columna1 + NEW.columna2 > 100 THEN
        RAISE EXCEPTION 'La suma de columna1 y columna2 no puede ser mayor que 100';
    END IF;
    RETURN NEW;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER trigger_validar_suma
BEFORE INSERT ON tabla
FOR EACH ROW
EXECUTE FUNCTION validar_suma();
```

Trigger para Actualizar Datos Relacionados

Este trigger actualiza automáticamente la fecha de última modificación de un registro en otra tabla cada vez que se actualiza un registro en la tabla principal.

```
CREATE OR REPLACE FUNCTION actualizar_fecha_modificacion()
RETURNS TRIGGER AS
$$
BEGIN
```

```
    UPDATE tabla_relacionada
    SET fecha_modificacion = current_timestamp
    WHERE id_tabla_relacionada = NEW.id_tabla_relacionada;
    RETURN NEW;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER trigger_actualizar_fecha_modificacion
AFTER UPDATE ON tabla_principal
FOR EACH ROW
EXECUTE FUNCTION actualizar_fecha_modificacion();
```

Ejercicios sobre Triggers

A continuación se presentan los enunciados de los ejercicios a resolver y documentar como parte de la entrega de documentación del trabajo práctico.

Ejercicio 1: Trigger para Registrar Cambios en una Tabla

Crea un trigger que registre automáticamente cada vez que se inserte, actualice o elimine un registro en una tabla específica. El registro debe incluir detalles como el tipo de operación realizada (inserción, actualización o eliminación), el usuario que realizó la operación, la fecha y hora de la operación, y los valores antiguos y nuevos afectados por la operación.

Ejercicio 2: Trigger para Validar Datos antes de la Inserción

Escribe un trigger que valide los datos antes de insertar un nuevo registro en una tabla. Por ejemplo, puedes crear un trigger que verifique que el valor de una columna no sea nulo o que cumpla con ciertos criterios de formato. Si los datos no cumplen con los criterios especificados, el trigger debe evitar la inserción del registro y generar un mensaje de error.

Ejercicio 3: Trigger para Mantener la Integridad Referencial

Crea un trigger que mantenga la integridad referencial entre dos tablas relacionadas. Por ejemplo, supongamos que tienes una tabla de pedidos y una tabla de productos, donde cada pedido está asociado a un producto. Puedes crear un trigger que evite la eliminación de un producto si hay pedidos asociados a ese producto. El trigger debe verificar si existen registros relacionados en la tabla de pedidos antes de permitir la eliminación del producto.

Eventos

A continuación se presentan tres ejemplos de eventos que pueden utilizarse de referencia para resolver los ejercicios del trabajo práctico.

Evento de Eliminación Masiva

Supongamos que deseas realizar una operación de eliminación masiva en una tabla, pero deseas tener un registro de los registros que se eliminan. Puedes simular este evento mediante el uso de un trigger:

```
CREATE OR REPLACE FUNCTION registro_eliminacion_masiva()
RETURNS TRIGGER AS
$$
BEGIN
    INSERT INTO registros_eliminados (id, fecha, datos)
    SELECT id, current_timestamp, OLD.*
    FROM old_table
    WHERE condition;

    RETURN OLD;
END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER trigger_eliminacion_masiva
AFTER DELETE ON old_table
FOR EACH ROW
EXECUTE FUNCTION registro_eliminacion_masiva();
```

Evento de Importación de Datos Externos

Supongamos que deseas ejecutar ciertas operaciones en la base de datos cada vez que se importan datos externos a una tabla específica. Puedes manejar este evento utilizando un script externo que ejecuta instrucciones SQL después de completar la importación de datos.

Evento de Copia de Seguridad Programada

Supongamos que deseas crear una copia de seguridad programada de tu base de datos en ciertos momentos del día. Puedes manejar este evento programando un script o una tarea cron que ejecute instrucciones SQL para crear la copia de seguridad en el momento deseado.

Estos son ejemplos de cómo puedes manejar eventos en PostgreSQL utilizando triggers o programación externa para observar ciertos cambios o situaciones y realizar acciones en respuesta a ellos.

Ejercicio sobre Eventos

Para el caso de eventos investigar sobre la posibilidad de utilizar eventos en algunas de las bases de datos implementadas en trabajos prácticos anteriores y documentar los resultados.