UT1. Comparativa de Servidores de Bases de Datos

El objetivo de esta práctica es investigar y comparar tres sistemas gestores de bases de datos ("SGBD"). Para ampliar nuestra perspectiva frente al imperante SQL, cada uno de nuestros tres SGBD, siendo todos *open source*, adopta un modelo de base de datos diferente (o incluso varios). Destacaremos sus características, diferencias, fortalezas, flaquezas y casos de uso.

1. Selección de Servidores

- **PostgreSQL**: en desarrollo desde 1986 y escrito en su mayoría en el lenguaje de programación C, se ha consolidado como el SGBD más popular en 2023.
- **Redis**: base de datos en memoria no-relacional (NoSQL) donde la información se almacena formando parejas clave-valor. Creada en 2009 y escrita en C y notablemente en Tcl.
- **SurrealDB**: SGBD multi-modelo iniciado en 2016 (Go), *open source* desde 2021 y reescrito en Rust. En 2022 se lanzó una versión beta pública y en Septiembre de 2023 se ha lanzado la versión 1.0.0.

2. Investigación y Documentación

> **ORM**: software que aporta compatibilidad entre estructuras de datos SQL y otras propias de lenguajes de de programación orientada a objetos. Ejemplos: Prisma, SQLAlchemy, Django, Laravel...

> **ACID**: conjunto de propiedades que garantizan que las transacciones de una base de datos se procesan de manera fiable. Atomicidad, Consistencia, Aislamiento (Isolation) y Durabilidad.

> **NoSQL**: aquel SGBD que adopta estructuras de datos alternativos a las tablas relacionales para almacenar la información. Diferentes modelos: documento, grafo, parejas clave-valor...

2.1. PostgreSQL

- PostgreSQL sigue una arquitectura cliente-servidor: un motor central gestiona la base de datos y acepta conexiones, y las aplicaciones clientes se conectan al servidor para realizar acciones.
- Su modelo de datos es **objeto-relacional**, organizando la información en tablas con filas y columnas mediante un SQL altamente extensible que admite tipos de datos personalizados, funciones y procedimientos almacenados con polimorfismo e *inheritance*.
- Se destacan extensiones como PostGIS (datos geo-espaciales), CITUS (*sharding*) o pg_embedding (AI).
- El rendimiento y la capacidad de carga de PostgreSQL son eficientes gracias a su optimizador de consultas y su gestión de índices para acelerar búsquedas.
- PostgreSQL es escalable y admite replicación tanto síncrona como asíncrona. Es posible configurar clústeres y réplicas para distribuir la carga y aumentar la disponibilidad.
- La seguridad se impone a varios niveles: protección interna de archivos críticos, conexiones basadas en sockets UNIX y no TCP/IP, control de acceso basado en roles, permisos, grupos, dirección IP...
- Para el control de concurrencia y transacciones, PostgreSQL incorpora técnicas como la multi-versión y la *serializable snapshot isolation*, permitiendo la ejecución simultánea de varias transacciones.
- PostgreSQL dispone de diversas herramientas de administración, como pgAdmin y psql, de modelaje como pgModeler, y es compatible con herramientas de respaldo, restauración y contenerización.
- PostgreSQL cuenta con un catálogo de módulos que integran Java, PHP, Python y otros lenguajes de programación. Además, constantemente se desarrollan nuevos ORM que fortalecen el ecosistema.

> Ejemplo de objeto pgSQL como tipo de dato:

```
-- Creación de un objeto como tipo de dato
CREATE TYPE Persona AS (
    nombre TEXT,
    fecha_nacimiento DATE,
    bio JSON,
    habilidades hstore            -- hstore: pareja clave-valor
);

-- Aplicación del objeto a una tabla
CREATE TABLE estudiantes (
    id SERIAL PRIMARY KEY,
    informacion Persona
);

-- Añadir información a un objeto
INSERT INTO estudiantes (informacion)
VALUES (
    ROW(
        'Pablo',
        '1995-03-10',
        '{"estado": "ando programando"}'::json, -- casting: 'foo::bar'
        'bash=>avanzado,sql=>principiante'::hstore
    )
);
```

2.2 Redis

- Redis es una base de datos **en memoria** de alto rendimiento muy utilizada como **caché** entre un SGBD relacional en expansión y las aplicaciones clientes (generalmente aplicaciones web).
- Su arquitectura cliente-servidor puede implantarse con un único nodo o mediante complejos *clusters*.
- Su modelo de datos es sencillo y flexible, ya que permite almacenar numerosas estructuras de datos como **parejas clave-valor**, donde la clave identifica aquella estructura registrada como valor.
- El rendimiento de Redis es excepcional ya que mantiene todos los datos en la memoria RAM (aunque permite persistencia). Puede manejar una alta capacidad de carga, tanto lecturas como escrituras.
- Redis admite la replicación maestro-esclavo (principal-réplica) para garantizar escalabilidad y alta disponibilidad, permitiendo la partición de datos en múltiples instancias.
- La seguridad de Redis ha venido siendo confesadamente elemental, necesitando una barrera externa (e.g. aplicación web) que gestione la autenticación y operaciones del usuario.
- Para su administración, Redis cuenta con la interfaz de comandos RedisCLI y la interfaz gráfica RedisInsight. De forma comercial se ofrece Redis Entreprise Cloud (DBaaS), que viene incoporando numerosos módulos especializados para el tratamiento de distintas estructuras de datos.
- Existen bibliotecas y clientes para una amplia gama de lenguajes de programación, incluyendo Python, JavaScript, Java y más, facilitando su integración con diversas aplicaciones y frameworks.
- Redis es ideal para casos de uso que requieren alta velocidad y baja latencia, como almacenamiento en caché, colas de mensajes, contadores en tiempo real y sesiones web.

Demostración de RedisCLI

```
# Creación y lectura de una clave y su valor -- tipo de dato: string
redislabs.com:17025> set bike:1 Deimos
OK
redislabs.com:17025> get bike:1
"Deimos"

# Creación y lectura de una clave y su valor -- tipo de dato: hash
redislabs.com:17025> hset bike:1 model Deimos brand Ergonom type 'Enduro
bikes' price 4972
(integer) 4
redislabs.com:17025> hget bike:1 model
"Deimos"
redislabs.com:17025> hget bike:1 price
"4972"
```

2.3 SurrealDB

- La arquitectura de SurrealDB puede ser distribuida o de nodo-único e implantarse en memoria, en un navegador web, en disco o en la **nube**. Se trata de un SGBD muy escalable, *cloud-native* y serverless.
- SurrealDB es **multimodelo** (relacional, documento y grafo) y *schemaless* por defecto, lo que permite almacenar información "desestructurada" (aunque SurrealQL permite definiciones *schemafull*). Se sustituyen los JOINs y FOREIGN KEYS por *record links* y otros elementos de grafo.
- El rendimiento de SurrealDB está próximo al de otros SGBD. Se esperan *benchmarks* oficiales.
- SurrealDB implementa RBAC para la autenticación de usuarios y múltiples mecanismos de seguridad.
- Para su administración existen una interfaz CLI y numerosas herramientas para su implantación a través de la nube y su integración con otros lenguajes de programación, APIs, WebSockets y más.
- Producto *open source* grauito, su licencia solo coarta su comercialización como DBaaS.
- Este SGBD es muy joven y muy ambicioso. Parece alineado con las tendencias de *software* actuales pero necesita madurar para que podamos destacar usos de caso sobresalientes.

Ejemplos de comandos SurrealQL para el modelado de datos

```
-- Con CREATE creamos un registro:uniqueID, con SET le damos contenido
CREATE human:jeff SET
  nickname="Jeff", age=99, sex=true;

-- Creación de tablas schemafull
DEFINE TABLE human SCHEMAFULL;
DEFINE FIELD age ON TABLE human TYPE int;

-- Demo de lo que sustituye a las PKs y FKs ("Fetch from related table")
CREATE human:jeff SET
  Name= 'Jeff',
  friends = [human:chad, human:todd];
SELECT friends.name FROM human:jeff;
```

3. Comparativa

> Tablas, gráficos u otros formatos visuales: destacar las diferencias y similitudes en cada área

	PostgreSQL	Redis	SurrealDB
Paradigma	SQL	NoSQL	NewSQL
Relacional	Sí	No	Sí
Schema-	full	less	less + full
ACID	Sí	No	Sí
Serverless	¿Sí?	Sí	Sí

4. Conclusiones y Recomendaciones

En 1970 no solo se publicaba el Modelo Relacional de E. Codd, sino también el Protocolo de Control de Red para ARPANET. Aún no existía Internet. Hoy, 53 años después, la industria del software navega tendencias como la computación en la nube, la contenerización, los microservicios, el CI/CD, la integración de IA, el *Internet of Things*... Tecnologías y prácticas de lo más reciente e innovador que siguen valiéndose del Modelo Relacional como base para todo tipo de productos y servicios. Los SGBD relacionales siguen siendo fundamentales por su eficacia para el mantenimiento de la integridad de la información, su capacidad de representar relaciones complejas, un lenguaje de programación (SQL) maduro... Por esto, bases de datos como MySQL, **PostgreSQL** y Oracle Database prevalecen y es preciso saber trabajar con ellas.

El Modelo Relacional implementa las propiedades ACID desde 1981, lo que ha resuelto innumerables problemas para el almacenamiento y la gestión de información. Sin embargo la innovación no se detiene, con lo que surgen nuevos desafíos para los que el Modelo Relacional no tiene respuesta. Es por esto que se desarrollan nuevas bases de datos no solo relacionales (*Not only SQL*) que priorizan el rendimiento y la escalabilidad, en general adoptando nuevas estrategias y estructuras de datos más flexibles que las del Modelo Relacional. MongoDB almacena la información en formato JSON, usado por prácticamente todas las aplicaciones web. **Redis** aporta velocidad en cualquier infraestructura web. Apache Cassandra adopta una arquitectura *peer-to-peer* y pone el acento en la escalabilidad y la alta disponibilidad.

En definitiva, mientras las grandes empresas de software lanzan sus propias bases de datos relacionales y no relacionales en la nube, siguen surgiendo nuevos proyectos como **SurrealDB** que apuestan por integrar de forma nativa las tecnologías y convenciones (APIs, GraphQL, React, SQL, wasm...) que dan forma al internet que conocemos. Como resultado, las bases de datos SQL, NoSQL y NewSQL siguen en desarrollo, evolucionando y mejorando, a menudo implementando características comunes e implantándose en conjunto en el *stack* tecnológico de los incontables servicios digitales que pueblan la red informática mundial, ofreciendo innovadoras soluciones para los desafíos del mañana.