



# ESCUELA POLITÉCNICA NACIONAL

## ESCUELA DE FORMACIÓN DE TECNÓLOGOS



### BASES DE DATOS (TSDS)

ASIGNATURA:

Bases de Datos

PROFESOR:

Ing. Lorena Chulde

FECHA DE ENTREGA:

2025 - 04-08

PERÍODO ACADÉMICO:

2025-A

## TÍTULO

## PROYECTO FINAL



### Estudiantes

**Ganchala Castillo Eduardo**  
**Girón Cedeño María Paula**

## Sistema Integral de Gestión de Datos para EPNprende (PostgreSQL)



### Introducción

EPNprende es una aplicación de escritorio desarrollada para la Escuela Politécnica Nacional (EPN), cuyo objetivo es facilitar la gestión de productos y servicios de estudiantes y profesores emprendedores. A través de esta plataforma, los miembros de la comunidad politécnica pueden publicar, vender y comprar productos, interactuar entre sí y mantener un entorno controlado para sus actividades comerciales.

Para garantizar un almacenamiento seguro, estructurado y eficiente de toda la información, se implementa una base de datos relacional en PostgreSQL. Este proyecto desarrolla dicha base de datos, desde el diseño en 3FN hasta la implementación de funciones avanzadas como procedimientos, triggers, auditoría, respaldo y roles de seguridad.

### Objetivos

#### Objetivo General

Diseñar e implementar una base de datos relacional en PostgreSQL que soporte de forma segura, eficiente y escalable la gestión de usuarios, productos, servicios, ofertas, notificaciones y auditoría para la aplicación EPNprende.

#### Objetivos Específicos

- Modelar un esquema de datos en 3FN que incluya usuarios, perfiles, publicaciones, interacciones, ofertas y auditoría.
- Implementar procedimientos, funciones y triggers que automaticen procesos clave y garanticen la integridad de los datos.
- Definir roles de seguridad y aplicar cifrado para proteger datos sensibles.
- Validar el rendimiento de la base con índices, consultas optimizadas y pruebas de carga.
- Documentar respaldos, auditoría y mecanismos de protección contra SQL Injection.

## Alcance del Proyecto de Base de Datos

La base de datos abarca los módulos principales de la aplicación EPNprende: gestión de usuarios y perfiles, publicación de productos y servicios (con categorías, fotos, ofertas), interacciones (favoritos, comentarios), estadísticas (clicks y productos destacados) y un sistema de auditoría robusto. No incluye, en esta versión, pasarelas de pago ni integración con APIs externas.

## Requerimientos Técnicos del Proyecto

- Uso de restricciones NOT NULL, CHECK, DEFAULT, UNIQUE y claves foráneas con ON DELETE CASCADE/SET NULL según contexto.

```
CREATE DATABASE "EPNprende"
WITH
  OWNER = postgres
  ENCODING = 'UTF8'
  LC_COLLATE = 'Spanish_Spain.1252'
  LC_CTYPE = 'Spanish_Spain.1252'
  LOCALE_PROVIDER = 'libc'
  TABLESPACE = pg_default
  CONNECTION LIMIT = -1
  IS_TEMPLATE = False;

COMMENT ON DATABASE "EPNprende"
  IS 'Esta base de datos relacional en PostgreSQL que soporta de forma segura, eficiente y escalable la gestión de usuarios y perfiles, publicación de productos y servicios (con categorías, fotos, ofertas), interacciones (favoritos, comentarios), estadísticas (clicks y productos destacados) y un sistema de auditoría robusto. No incluye, en esta versión, pasarelas de pago ni integración con APIs externas.';

CREATE TYPE products_stock
AS ENUM ('available', 'unavailable');

CREATE TABLE categories(
  category_id SMALLSERIAL PRIMARY KEY,
  category_name VARCHAR(100) NOT NULL UNIQUE,
  category_description TEXT
);

CREATE TYPE users_rol
AS ENUM ('admin', 'user');

CREATE TABLE users(
  user_id SMALLSERIAL PRIMARY KEY,
  user_firebase VARCHAR(128) NOT NULL UNIQUE,
  user_rol USERS_ROL NOT NULL DEFAULT 'user'
);

CREATE TABLE profiles(
  profile_id SMALLSERIAL PRIMARY KEY,
  profile_name VARCHAR(255) NOT NULL,
  profile_description TEXT NOT NULL,
  user_id INT NOT NULL,
  CONSTRAINT fk_user_id FOREIGN KEY (user_id)
  REFERENCES users (user_id) ON DELETE CASCADE
);
```

## Creación de al menos 5 procedimientos (inserción validada, actualizaciones masivas, eliminaciones seguras, reportes).

Creación de los procedimientos almacenados en las tablas más críticas del sistema (users, products, profiles, offers y reports).

-- Tabla de usuarios: Procedimiento para insertar un usuario con validaciones.

```
573 RETURN;  
574 END IF;  
575  
576 -- Insertar el nuevo usuario  
577 INSERT INTO users(user_firebase, user_rol)  
578 VALUES (p_user_firebase, p_user_rol);  
579  
580 p_mensaje_resultado := 'Usuario registrado correctamente con ID: ' || currval('users_');  
581 END;  
582 $$;  
583  
584 -- Llamada básica  
585 CALL insertar_usuario(  
586     p_user_firebase := 'abc123xyz456',  
587     p_user_rol := 'user',  
588     p_mensaje_resultado := ''  
589 );  
590  
591  
592  
593  
594
```

Data Output		Messages	Notifications
	p_mensaje_resultado		
	text		
1	Usuario registrado correctamente con ID: 51		

-- Tabla de productos: Procedimiento para actualizar estado de productos.

```
617  
618 -- Cambiar estado a 'unavailable' para categoría 5  
619 CALL actualizar_estado_productos(  
620     p_categoria_id := 5,  
621     p_nuevo_estado := 'unavailable',  
622     p_productos_actualizados := 0,  
623     p_mensaje_resultado := ''  
624 );  
625  
626  
627  
628
```

Data Output		Messages	Notifications
	p_productos_actualizados		p_mensaje_resultado
	integer		text
1	1	Actualización completada. Productos modificados: 1	

-- Tabla perfiles: Procedimiento para eliminar perfil y sus dependencias (profiles)

```

653      -- Eliminar el perfil
654      DELETE FROM profiles WHERE profile_id = p_profile_id;
655
656      p_mensaje_resultado := 'Perfil eliminado correctamente. ' ||
657                          'Productos eliminados: ' || v_productos_eliminados || ', ' ||
658                          'Comentarios eliminados: ' || v_comentarios_eliminados;
659  END;
660  $$;
661
662  -- Llamada
663  CALL eliminar_perfil_seguro(10, '');
664
665

```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1

	p_mensaje_resultado
1	Error: El perfil especificado no existe

-- Procedimiento para actualización masiva de precios Tablas: (products + offers)

```

703      AND (p_categoria_id IS NULL OR p_categoria_id = p_categoria_id);
704
705      GET DIAGNOSTICS p_ofertas_actualizadas = ROW_COUNT;
706
707      p_mensaje_resultado := 'Actualización masiva completada. ' ||
708                          'Productos actualizados: ' || p_productos_actualizados || ', ' ||
709                          'Ofertas ajustadas: ' || p_ofertas_actualizadas;
710  END;
711  $$;
712  -- Aumento del 10% para todos los productos
713  CALL actualizar_precios_masivo(
714      p_porcentaje_aumento := 10,
715      p_categoria_id := NULL,
716      p_productos_actualizados := 0,
717      p_ofertas_actualizadas := 0,
718      p_mensaje_resultado := ''
719  );
720
721

```

Data Output Messages Notifications

Showing rows: 1 to 1 Page No: 1 of 1

	p_productos_actualizados	p_ofertas_actualizadas	p_mensaje_resultado
1	48	48	Actualización masiva completada. Productos actualizados: 48, Ofertas ajustadas: 48

-- Tabla reports: Inserta un nuevo reporte, asegurando que el perfil que lo genera existe y que se reporta al menos un perfil o producto, cumpliendo con las restricciones de integridad definidas.

```

759         p_descripcion,
760         NOW(),
761         p_perfil_generador,
762         p_perfil_reportado,
763         p_producto_reportado
764     );
765
766     RAISE NOTICE 'Reporte insertado correctamente.';
767 END;
768 $$;
769 CALL insertar_reporte_simple('Comportamiento sospechoso', 1, 3, NULL);
770 CALL insertar_reporte_simple('Producto con imágenes inapropiadas', 2, NULL, 15);
771 select*from reports;

```

Data Output Messages Notifications

NOTICE: Reporte insertado correctamente.

CALL

Query returned successfully in 121 msec.

83	105	2025-07-30 18:38:19.495865	Producto con imágenes inapropiadas	2	[null]	15
84	106	2025-07-30 18:41:01.815316	Comportamiento sospechoso	1	3	[null]

- Implementación de 3 funciones (cálculos de métricas, descuentos, estados de usuario).

-- Función para calcular el descuento promedio de ofertas.

```

784 SELECT calcular_descuento_promedio() AS descuento_promedio;

```

Data Output Messages Notifications

Showing

	descuento_promedio numeric
1	-10.00

-- Función para determinar la Popularidad de un producto.

```

802
803 -- Determinar popularidad
804 IF v_clicks > v_promedio_clicks * 2 THEN
805     v_popularidad := 'Muy Popular';
806 ELSIF v_clicks > v_promedio_clicks THEN
807     v_popularidad := 'Popular';
808 ELSE
809     v_popularidad := 'Normal';
810 END IF;
811
812 RETURN v_popularidad;
813 END;
814 $$ LANGUAGE plpgsql;
815 SELECT product_name, determinar_popularidad(product_id) AS popularidad
816 FROM products;

```

Data Output Messages Notifications

Showing rows:

	product_name character varying (255)	popularidad character varying
1	Smartphone X1	Popular
2	Vestido de verano	Normal
3	Balón de fútbol	Popular

-- Función para Validar una nueva oferta.

```
826 -- Verificar si el producto existe y está disponible
827 SELECT EXISTS (
828     SELECT 1 FROM products
829     WHERE product_id = p_product_id
830     AND product_stock = 'available'
831 ) INTO v_producto_valido;
832
833 -- Verificar si ya tiene una oferta activa
834 SELECT EXISTS (
835     SELECT 1 FROM offers
836     WHERE product_id = p_product_id
837     AND valid_until > CURRENT_DATE
838 ) INTO v_oferta_activa;
839
840 -- Retornar TRUE solo si el producto es válido y no tiene oferta activa
841 RETURN v_producto_valido AND NOT v_oferta_activa;
842 END;
843 $$ LANGUAGE plpgsql;
844 SELECT validar_oferta_basica(5) AS es_oferta_valida;
```

es_oferta_valida
boolean
1 false

- Implementación de 3 triggers (auditoría de cambios, control de stock, notificaciones).

## Auditoría de cambios de precio en productos

```
30 SELECT * FROM products;
31 INSERT INTO products (product_id,product_name,product_description,product_price,product_stock,product_publication_date,profile_id,category_id)
32 VALUES (1000,'Cámara HD','Cámara de alta definición para fotografía profesional',100.00,'available','2025-08-04',1,3);
33
34 UPDATE products
35 SET product_price = 120.00
36 WHERE product_id = 1000;
37
38 SELECT * FROM audit_products;
39
```

audit_id	product_id	old_price	new_price	changed_at
[PK] integer	integer	numeric (10,2)	numeric (10,2)	timestamp without time zone
1	1000	100.00	120.00	2025-08-04 00:41:57.196271

Guarda un registro de cuándo y cómo cambió el precio, qué precio tenía antes y cuál es el nuevo. Permite saber quién, cuándo y cómo se modificaron los precios, lo cual es importante para control interno, auditorías o seguridad. Si algo cambia mal, puedes tener registro para revisar y corregir.

## Control de stock

```
51 DROP TRIGGER IF EXISTS trg_check_stock ON products;
52 CREATE TRIGGER trg_check_stock
53 BEFORE INSERT OR UPDATE ON products
54 FOR EACH ROW
55 EXECUTE FUNCTION check_product_stock();
56
57 INSERT INTO products (product_id,product_name,product_description,product_price,product_stock,product_publication_date,profile_id,category_id)
58 VALUES (1001,'P55 SLIM','Consola que acepta juegos digital y físicos',500.00,'unavailable','2025-08-04',35,36);
59
```

ERROR: Stock insuficiente  
CONTEXT: PL/pgSQL function check\_product\_stock() line 4 at RAISE

Si el valor no cumple la condición (en tu código original, si NEW.product\_stock != 'available'), entonces se genera un error mediante RAISE EXCEPTION 'Stock insuficiente'. Esto provoca que la operación de inserción o actualización **se cancele** y no se modifique la tabla. Si el valor sí cumple la condición, la función retorna NEW y la operación continúa normalmente.

## Notificaciones por nuevos comentarios

```

93 -- =====
94 -- USO
95 -- =====
96 SELECT * FROM comment_users;
97 INSERT INTO comment_users (comment_text,comment_date,comment_update,comment_rate,profile_id,product_id)
98 VALUES ('!Me encanta este producto!',NOW(),NOW(),5,1,1000 );
99
100 SELECT * FROM notification_log ORDER BY created_at DESC;|
101

```

Data Output

Messages

Notifications

</

Cada vez que alguien comenta un producto, automáticamente queda registrada una notificación sin que el usuario o aplicación tenga que preocuparse por hacer esto manualmente. Se puede consultar la tabla `notification_log` para mostrar a los usuarios notificaciones o para análisis internos.

## Definición de roles y privilegios (`admin_db`, `auditor_db`, `operador_db`, `usuario_final`).

El diseño de roles y privilegios en la base de datos EPNprendeDB se ha estructurado para garantizar seguridad, eficiencia y segregación de funciones. Se implementaron cuatro roles principales con distintos niveles de acceso, asignados a esquemas específicos para controlar el alcance de las operaciones que cada usuario puede realizar. Esta arquitectura sigue el principio de mínimo privilegio, otorgando solo los permisos estrictamente necesarios para cada función.

### - Roles y sus Funcionalidades

#### Rol Administrador (`admin_db`)

Este rol tiene privilegios completos sobre toda la base de datos, incluyendo todos los esquemas (`config`, `operaciones`, `auditoria` y `publico`). Puede crear, modificar y eliminar cualquier objeto, así como gestionar otros roles. Es el único con capacidad para alterar la estructura de la base de datos o realizar operaciones sensibles. Su configuración incluye una ruta de búsqueda optimizada que prioriza los esquemas administrativos.

#### - Rol Auditor (`auditor_db`)

Diseñado exclusivamente para actividades de monitoreo, este rol tiene permisos de solo lectura sobre el esquema de auditoría. Puede consultar los registros de logs y tablas de trazabilidad, pero no tiene acceso a modificar datos ni a visualizar información en otros esquemas. Esta restricción asegura que las actividades de auditoría sean transparentes, pero no interferentes.



#### - **Rol Operador (operador\_db)**

Dirigido al personal operativo, este rol puede realizar operaciones básicas CRUD (Crear, Leer, Actualizar) en el esquema de operaciones, pero específicamente se le ha revocado el permiso DELETE para prevenir eliminaciones accidentales o malintencionadas. Se ha implementado un límite de conexiones simultáneas para evitar sobrecargas, y su ruta de búsqueda está configurada para priorizar el esquema operaciones.

#### - **Rol Usuario Final (usuario\_final)**

Con el acceso más restrictivo, este rol solo puede realizar consultas de lectura (SELECT) en el esquema público, donde residen los datos destinados a visualización general. No tiene permisos para acceder a esquemas sensibles ni para ejecutar operaciones de escritura. Al igual que el rol operador, tiene un límite de conexiones para gestionar recursos.

### **Configuraciones Clave en la base de datos**

#### - **Límite de Conexiones Simultáneas (CONNECTION LIMIT)**

Esta configuración restringe el número máximo de conexiones que cada rol puede establecer simultáneamente con la base de datos. Por ejemplo, el operador\_db tiene un límite de 20 conexiones, mientras que el usuario\_final está limitado a 10. Esto previene que un usuario agote los recursos del servidor mediante múltiples conexiones, ya sea por error o por actividades maliciosas. Cuando se alcanza el límite, cualquier intento adicional de conexión será rechazado automáticamente.

#### - **Ruta de Búsqueda (search\_path)**

La ruta de búsqueda define el orden en que PostgreSQL busca objetos cuando no se especifica explícitamente el esquema. Para el usuario\_final, por ejemplo, está configurada para buscar primero en el esquema público y luego en public. Esto permite que las consultas puedan omitir el nombre del esquema (ej: "SELECT \* FROM productos" en lugar de "SELECT \* FROM publico.productos"), mejorando la usabilidad. Además, al controlar esta ruta, se asegura que los usuarios accedan prioritariamente a los esquemas designados para su rol, reduciendo riesgos de acceso accidental a datos sensibles.

### **Estructura de Esquemas y su Relación con los Roles**

La base de datos organiza los objetos en esquemas especializados:

- **config:** Parámetros del sistema, accesible solo por admin\_db.
- **operaciones:** Datos transaccionales del negocio, accesible por operador\_db y admin\_db.
- **auditoria:** Registros de seguimiento, accesible por auditor\_db y admin\_db.
- **publico:** Datos de consulta pública, accesible por usuario\_final y admin\_db.

Esta segmentación permite un aislamiento lógico de los datos según su sensibilidad y uso. Adicionalmente, se aplicaron restricciones explícitas para evitar que roles no autorizados accedan a esquemas fuera de su alcance.

## Beneficios de la Implementación

### Seguridad

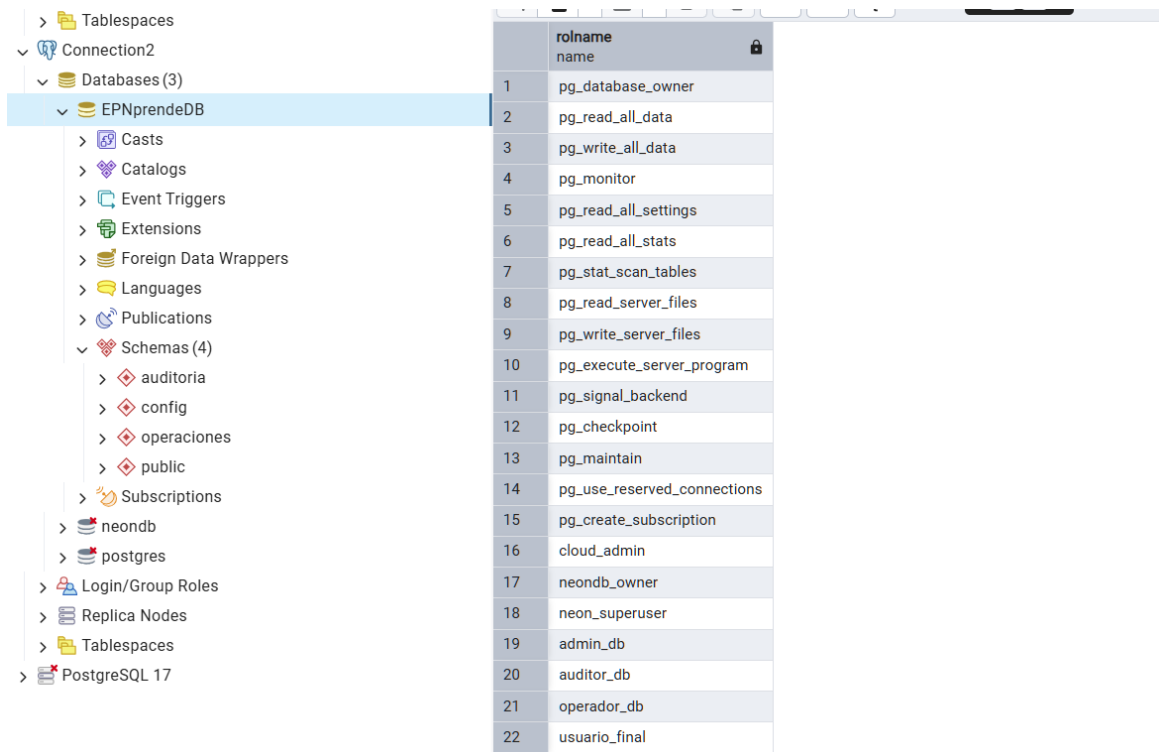
- **Segregación estricta** de funciones mediante roles especializados.
- **Acceso mínimo necesario**, reduciendo superficies de ataque.
- **Protección contra sobrecargas** mediante límites de conexión.

### Rendimiento

- **Optimización de consultas** mediante rutas de búsqueda personalizadas.
- **Distribución controlada** de recursos del servidor.

### Mantenibilidad

- **Organización clara** de objetos por esquemas.
- **Facilidad de auditoría** gracias a permisos.



	rolname name	
1	pg_database_owner	
2	pg_read_all_data	
3	pg_write_all_data	
4	pg_monitor	
5	pg_read_all_settings	
6	pg_read_all_stats	
7	pg_stat_scan_tables	
8	pg_read_server_files	
9	pg_write_server_files	
10	pg_execute_server_program	
11	pg_signal_backend	
12	pg_checkpoint	
13	pg_maintain	
14	pg_use_reserved_connections	
15	pg_create_subscription	
16	cloud_admin	
17	neondb_owner	
18	neon_superuser	
19	admin_db	
20	auditor_db	
21	operador_db	
22	usuario_final	

## **Credenciales especificadas por fines educativos**

1. Rol Administrador
  - i. Clave: EPNprende@Admin2025!
2. Rol Auditor
  - i. Clave: EPNprende@Auditor2025\*
3. Rol Operador
  - i. Clave: EPNprendeOperador@EPN2025%
4. Rol usuario
  - i. Clave: EPNprende@Usuario2025#

## **Uso de pgcrypto para cifrado de contraseñas y correos, con digest (SHA256) y cifrado simétrico.**

### **Proceso de Cifrado de IDs de Firebase en PostgreSQL**

PostgreSQL ofrece la extensión pgcrypto para operaciones criptográficas, permitiendo:

- Hash seguro (SHA-256) para contraseñas de los usuarios (id de los usuarios de firebase).

Cifrado de IDs de Firebase en la Tabla users: Esta acción será realizada en el esquema de auditoría ya que esta sección, está diseñada para manejar datos sensibles y operaciones críticas. Por defecto solo es accesible por admin\_db y auditor\_db (según tu configuración previa en la sección de roles y privilegios).

### **1. Objetivo del Script**

Implementar un sistema seguro de almacenamiento para los IDs de Firebase mediante cifrado AES, eliminando los datos sensibles en texto plano y asegurando su accesibilidad solo para roles autorizados.

## **2. Actividades Realizadas**

### **A. Preparación del Entorno**

#### **Creación del esquema operaciones**

- Se estableció con permisos exclusivos para admin\_db y acceso limitado para operador\_db.

#### **Activación de la extensión pgcrypto**

- Requerida para funciones de cifrado avanzado (pgp\_sym\_encrypt/decrypt).

### **Configuración de permisos en el esquema auditoria**

- Se restringió el acceso a PUBLIC y se otorgó acceso exclusivo a admin\_db.

## **B. Modificación de la Estructura de Datos**

### **Adición de columnas para cifrado**

- user\_firebase\_cifrado (BYTEA): Almacena los IDs cifrados con AES.
- user\_firebase\_iv (BYTEA): Reservada para IV (no utilizada finalmente).

### **Función de cifrado auditoria.cifrar\_firebase\_id**

- Recibe texto plano y una clave, devuelve el contenido cifrado.
- Configurada como SECURITY DEFINER para ejecutarse con privilegios de administrador.

## **C. Proceso de Cifrado**

### **Actualización masiva de datos en la tabla**

```

34
35 -- Ejecutar como admin_db la encriptación
36 v UPDATE users
37 SET user_firebase_cifrado = auditoria.cifrar_firebase_id(user_firebase, 'tu_clave_secreta_AES');
38 select*from users;
39

```

Todos los IDs se cifraron usando una clave simétrica.

### **Verificación de resultados**

- Consulta SELECT \* FROM users para validar la transformación.

Data Output			
Messages			
Notifications			
	user_id [PK] smallint	user_role users_role	user_firebase bytea
1	1	admin	[binary data]
2	2	user	[binary data]
3	3	user	[binary data]
4	4	user	[binary data]
5	5	user	[binary data]
6	6	user	[binary data]
7	7	user	[binary data]
8	8	user	[binary data]
9	9	user	[binary data]
10	10	user	[binary data]
11	11	user	[binary data]
12	12	user	[binary data]
13	13	user	[binary data]
14	14	user	[binary data]

## D. Gestión de Seguridad Post-Cifrado

### Backup de datos originales

- Se creó la tabla auditoria.backup\_firebase\_ids para preservar los IDs en texto plano (con fines de auditoría/reversión).

### Eliminación de datos sensibles

- Se eliminó la restricción UNIQUE en user\_firebase.
- Se eliminó la columna user\_firebase (texto plano).
- La columna cifrada se renombró a user\_firebase para mantener compatibilidad.

### Eliminación del IV no utilizado

- Se verificó que user\_firebase\_iv estaba vacía (NULL) y se eliminó.

## Flujo del Proceso



### Medidas de Seguridad Implementadas

- **Cifrado AES-256:** A través de `pgp_sym_encrypt`.
- **Backup en esquema restringido:** Solo accesible por `admin_db`.
- **Eliminación de datos redundantes:** Minimiza superficies de ataque.

### Notas Clave

- **El IV no se usó:** PostgreSQL lo gestiona internamente en `pgp_sym_encrypt`.
- **El esquema auditorio** centraliza todas las operaciones sensibles.
- **El rol operador\_db** solo tiene acceso a datos cifrados, no a funciones de descifrado.

### Esquemas en la Base de Datos

#### 1. Esquema operaciones

- **Propósito:** Contiene todas las tablas y datos transaccionales del negocio (ej: información de usuarios, productos, ventas).
- **Acceso:**
  - `neondb_owner`: Permisos completos (CREAR/MODIFICAR/ELIMINAR).

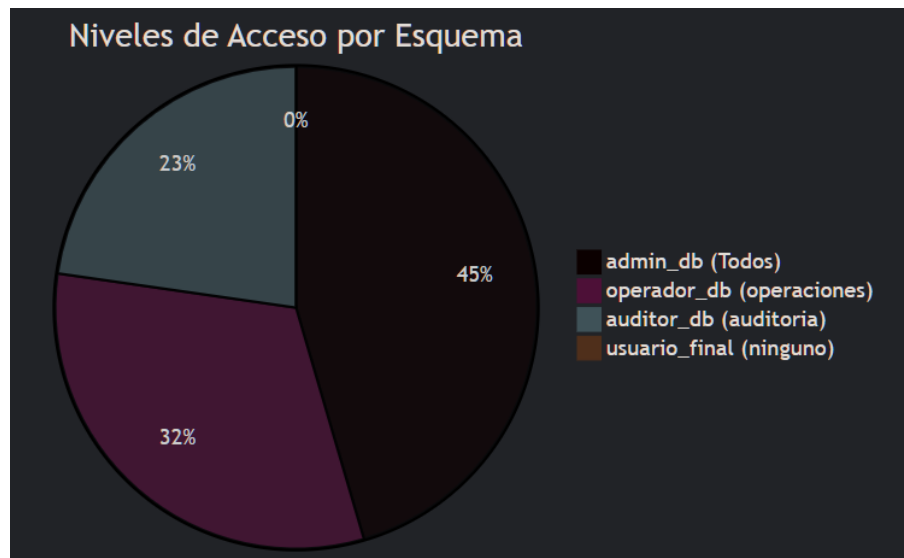
- operador\_db: Solo USAGE + permisos CRUD limitados (sin DELETE).

## 2. Esquema auditoria

- **Propósito:** Almacena registros de seguridad y cambios críticos:
  - Logs de cifrado (backup\_firebase\_ids).
  - Funciones de encriptación (cifrar\_firebase\_id).
- **Acceso:**
  - Solo admin\_db y auditor\_db (este último con permisos de solo lectura).

## 3. Esquema public (por defecto)

- **Propósito:**
  - Objetos compartidos o temporales.
  - Acceso restringido, solo permitido a al administrador y el auditor.



## Registro de intentos fallidos o sospechosos (simulado).

Se implementa una tabla específica para almacenar los registros de intentos fallidos o sospechosos.

```

8 GRANT USAGE ON SCHEMA auditoria TO admin_db;
9
10 GRANT USAGE ON SCHEMA auditoria TO auditor_db;
11
12 GRANT ALL ON SCHEMA auditoria TO neondb_owner;
13
14 -- Registro de intentos fallidos o sospechosos (simulado).
15 -- Crear tabla para registrar intentos fallidos
16 CREATE TABLE auditoria.intentos_fallidos (
17     id SERIAL PRIMARY KEY,
18     usuario TEXT,           -- Usuario que intentó acceder
19     ip INET,               -- Dirección IP del intento
20     fecha TIMESTAMPTZ DEFAULT CURRENT_TIMESTAMP,
21     accion TEXT,           -- 'LOGIN_FAILED', 'SQL_INJECTION_ATTEMPT'
22     detalles TEXT          -- Detalles del error o payload sospechoso
23 );
24
25 GRANT INSERT ON auditoria.intentos_fallidos TO admin_db;
26 GRANT SELECT ON auditoria.intentos_fallidos TO auditor_db;
27
37
38 -- Registro de intentos fallidos o sospechosos (simulado).
39 -- Ver todos los registros
40 SELECT * FROM auditoria.intentos_fallidos
41 ORDER BY fecha DESC;
42
43 -- Filtrar por tipo de acción [ej: intentos de SQL Injection]
44 SELECT * FROM auditoria.intentos_fallidos
45 WHERE accion = 'SQL_INJECTION_ATTEMPT';

```

Data Output Messages Notifications

Showing rows: 1 to 2 Page No: 1 of

	id [PK] integer	usuario text	ip inet	fecha timestamp with time zone	accion text	detalles text
1	2	hacker	10.0.0.5	2025-08-04 04:27:58.350861+00	SQL_INJECTION_ATTEMPT	Payload: ' OR '1'='1

Total rows: 2 Query complete 00:00:06.238

## Revisión del historial de roles asignados y auditoría de privilegios activos.

```

13
14 -- Revisión del historial de roles asignados y auditoría de privilegios activos.
15 -- Crear tabla de historial de roles
16 CREATE TABLE auditoria.historial_roles (
17     id SERIAL PRIMARY KEY,
18     rol_afectado TEXT NOT NULL,           -- Rol modificado
19     accion TEXT NOT NULL CHECK (accion IN ('CREACION', 'MODIFICACION', 'ELIMINACION', 'ASIGNACION_PRIVILEGIO')),
20     privilegios TEXT,                   -- Detalle de privilegios asignados/revocados
21     ejecutado_por TEXT NOT NULL,       -- Rol que realizó el cambio
22     fecha TIMESTAMPTZ DEFAULT CURRENT_TIMESTAMP,
23     detalles TEXT
24 );
25
26 GRANT INSERT, SELECT ON auditoria.historial_roles TO admin_db;
27 GRANT SELECT ON auditoria.historial_roles TO auditor_db;

```

## Configuración SSL/TLS para PostgreSQL en Windows

- Creación de índices simples y compuestos para optimizar consultas.



```

4  -- Índice simple
5  CREATE INDEX IF NOT EXISTS idx_product_name ON products(product_name);
6  -- =====
7  -- USO
8  -- =====
9  SELECT * FROM products
10 WHERE product_name ILIKE '%Cámara%';
11

```

	product_id [PK] integer	product_name character varying (255)	product_description text	product_price numeric (10,2)	product_stock products_stock	product_publication_date timestamp without time zone	profile_id integer	category_id integer
1	458	Cámara 360	Cámara para grabación panorámica	199.99	available	2025-07-31 21:12:00	458	22
2	468	Cámara de Vigilancia	Cámara con visión nocturna	89.99	available	2025-07-31 21:12:00	468	44
3	22	Cámara réflex	Cámara DSLR 24MP con lente 18-55mm	549.99	available	2025-07-31 16:37:31.982614	21	22
4	488	Cámara Compacta	Cámara portátil de 20MP	199.99	available	2025-07-31 21:12:00	488	22
5	498	Cámara de Viaje	Cámara ligera para aventuras	149.99	available	2025-07-31 21:12:00	498	22
6	408	Cámara de Acción 4K	Cámara resistente para deportes	149.99	available	2025-07-31 21:12:00	408	22
7	418	Cámara Compacta	Cámara portátil de 20MP	199.99	available	2025-07-31 21:12:00	418	22

```

12 -- 0:
13 SELECT * FROM products
14 ORDER BY product_name;
15

```

	product_id [PK] integer	product_name character varying (255)	product_description text	product_price numeric (10,2)	product_stock products_stock	product_publication_date timestamp without time zone	profile_id integer	category_id integer
1	147	Acete Corporal	Acete hidratante de almendras	14.99	available	2025-07-31 21:06:00	147	7
2	11	Acete de motor	Acete sintético 5W-30 1 litro	14.29	available	2025-07-31 16:37:31.982614	23	11
3	123	Armario Organizador	Armario de tela para ropa	59.99	available	2025-07-31 21:06:00	123	33
4	139	Armónica	Armónica diatónica en do	14.99	available	2025-07-31 21:06:00	139	26
5	61	Auriculares Bluetooth	Auriculares inalámbricos con cancelación de ruido	89.99	available	2025-07-31 21:06:00	61	36
6	371	Auriculares Gaming	Auriculares con micrófono para juegos	79.99	available	2025-07-31 21:06:00	371	38
7	221	Auriculares Gaming	Auriculares con micrófono para juegos	79.99	available	2025-07-31 21:06:00	221	38

Este índice mejora el rendimiento de las consultas que **buscan, filtran o ordenan por el nombre del producto**.

```

23 -- Búsqueda por producto:
24 SELECT * FROM comment_users
25 WHERE product_id = 10;

```

	comment_id [PK] integer	comment_text text	comment_date timestamp without time zone	comment_update timestamp without time zone	comment_rate smallint	profile_id integer	product_id integer
1	13	La guitarra tiene un sonido increíble	2025-07-31 16:37:31.982614	2025-07-31 16:37:31.982614	5	14	10

```

27 -- Búsqueda combinada:
28 SELECT * FROM comment_users
29 WHERE product_id = 1000 AND profile_id = 1;

```

	comment_id [PK] integer	comment_text text	comment_date timestamp without time zone	comment_update timestamp without time zone	comment_rate smallint	profile_id integer	product_id integer
1	55	¡Me encanta este producto!	2025-08-04 01:03:21.448931	2025-08-04 01:03:21.448931	5	1	1000

Este índice mejora consultas que **filtran comentarios por producto** y opcionalmente por **usuario (perfil)**.

```

30 -- Índice compuesto en ofertas
31 select*from offers;
32 CREATE INDEX IF NOT EXISTS idx_offer_product_dates
33 ON offers(product_id, valid_from, valid_until);
34 -- =====
35 -- USO
36 -- =====
37 SELECT * FROM offers
38 WHERE product_id = 50 AND valid_from >= CURRENT_DATE;
39
40

```

	offer_id integer	old_price numeric (10,2)	new_price numeric (10,2)	valid_from timestamp without time zone	valid_until timestamp without time zone	offer_description text	product_id integer
1	50	32.99	36.29	2027-02-01 00:00:00	2027-02-28 00:00:00	Promoción de jardinería	50

Este índice está diseñado para **acelerar las búsquedas de ofertas válidas** por producto en un **rango de fechas**.

- **Evidencia de respaldo y restauración (en caliente y frío).**

## Caliente

```
PS C:\> pg_dump --host=ep-old-surf-acgsonfj-pooler.sa-east-1.aws.neon.tech `
>> --port=5432 `
>> --username=neondb_owner `
>> --dbname=EPNprendeDB `
>> --format=custom `
>> --file=backup_EPNprende.dump
>>
Contraseña:
PS C:\> _
```

backup\_EPNprende.dump

3/8/2025 22:23

Archivo DUMP

138 KB

```
PS C:\> pg_restore --host=ep-old-surf-acgsonfj-pooler.sa-east-1.aws.neon.tech `
>> --port=5432 `
>> --username=neondb_owner `
>> --dbname=EPNprendeRestaurada `
>> --verbose `
>> --clean `
>> backup_EPNprende.dump
>>
pg_restore: conectando a la base de datos para reestablecimiento
Contraseña:
pg_restore: eliminando FK CONSTRAINT log_auditoria log_auditoria_usuario_id_fkey
pg_restore: durante PROCESAMIENTO DE TABLA DE CONTENIDOS:
pg_restore: en entrada de la tabla de contenidos 3523; 2606 65546 FK CONSTRAINT log_auditoria log_auditoria_usuario_id_f
key neondb_owner
pg_restore: error: could not execute query: ERROR: relation "public.log_auditoria" does not exist
La orden era: ALTER TABLE ONLY public.log_auditoria DROP CONSTRAINT log_auditoria_usuario_id_fkey;
pg_restore: eliminando FK CONSTRAINT profiles fk_user_id
pg_restore: error: could not execute query: ERROR: relation "public.profiles" does not exist
La orden era: ALTER TABLE ONLY public.profiles DROP CONSTRAINT fk_user_id;
```

- **Simulación y mitigación de ataques de SQL Injection mediante consultas preparadas y validaciones.**

```
1  -- =====
2  -- CONSULTA PREPARADA (SQL Injection Mitigación)
3  -- =====
4  select * from profiles;
5  -- Consultar por id del usuario
6  SELECT * FROM profiles WHERE user_id = 40;
7  -- Inyección
8  SELECT * FROM profiles WHERE user_id = 0 OR '1'='1';
9
10 -- Consulta segura usando PREPARE y EXECUTE
11 v PREPARE get_profile_by_user_id(INT) AS
12     SELECT * FROM profiles WHERE user_id = $1;
13
14 -- Ejecutar la consulta con un valor real
15 EXECUTE get_profile_by_user_id(40);
```

Data Output

Messages

Notifications

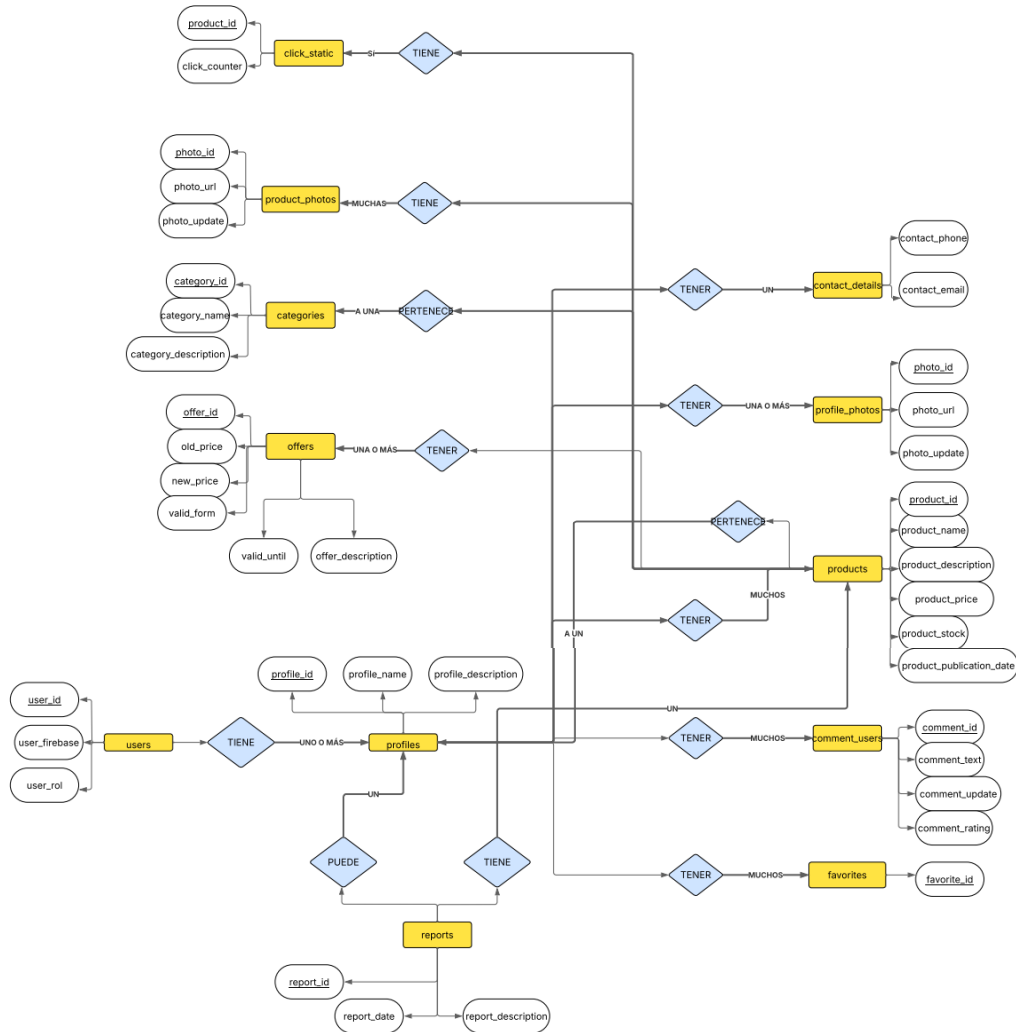
SQL

	profile_id [PK] smallint	profile_name character varying (255)	profile_description text	user_id integer
1	40	Diana Ramos	Software y licencias originales	40

Los valores se tratan como datos, no como código, valida los tipos de los parámetros, una vez preparada, la consulta puede ejecutarse muchas veces sin volver a analizar el plan.

## Modelo Conceptual y Lógico

El modelo de datos se basa en un DER con relaciones 1:N y N:M, cubriendo usuarios, perfiles, productos, categorías, ofertas, favoritos, comentarios, estadísticas y auditoría. El modelo relacional asegura integridad referencial completa, con claves foráneas en todas las relaciones.





- Triggers para auditoría y automatización de procesos.
- Roles y privilegios definidos para control de acceso.

## Validación y Pruebas

Se realizaron pruebas para validar:

- Rendimiento de consultas antes y después de aplicar índices.

## Monitoreo y Rendimiento

- Consulta de tamaño de tablas, índices, uso de disco.

```
18
19 -- Tamaño por tabla (incluyendo índices y otras transacciones en el sistema)
20 SELECT
21     table_name,
22     pg_size_pretty(pg_total_relation_size(quote_ident(table_name))) AS tamaño
23 FROM information_schema.tables
24 WHERE table_schema = 'public'
25 ORDER BY pg_total_relation_size(quote_ident(table_name)) DESC;
```

Data Output Messages Notifications

Showing rows: 1 to 12

	table_name name	tamaño text
1	users	176 kB
2	products	144 kB
3	profiles	104 kB
4	offers	64 kB
5	favorites	56 kB
6	click_statistics	56 kB
7	categories	48 kB
8	reports	32 kB
9	comment us...	32 kB

### ¿Qué hace esta consulta?

#### Propósito:

Lista todas las tablas del esquema operaciones junto con su tamaño en formato legible (KB, MB, GB).

#### Funciones clave:

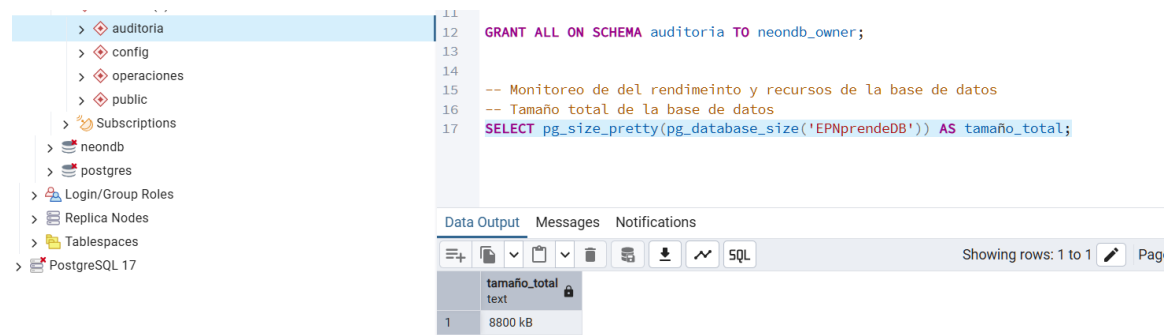
- `pg_total_relation_size()`: Calcula el tamaño total de una tabla (incluyendo índices y TOAST).
- `pg_size_pretty()`: Convierte el tamaño en bytes a formato humano (ej: 125 MB).
- `quote_ident()`: Previene inyección SQL al escapar nombres de tablas.

#### Filtro:

`WHERE table_schema = 'operaciones'` → Solo tablas de este esquema.

- Consulta del tamaño total de la base de datos.

Se muestra que el tamaño de la base de datos en el momento de la ejecución de la prueba es de 8000 Kb.



The screenshot shows a PostgreSQL query editor with a left sidebar containing a tree view of database objects. The main editor displays a SQL query with line numbers 11 through 17. The query includes a GRANT statement and a SELECT statement that uses the pg\_size\_pretty function to report the database size. Below the query editor, the 'Data Output' tab is active, showing a single row of results in a table with two columns: 'tamaño\_total' (text) and '8800 kB'.

```

11
12 GRANT ALL ON SCHEMA auditoria TO neondb_owner;
13
14
15 -- Monitoreo de del rendimeinto y recursos de la base de datos
16 -- Tamaño total de la base de datos
17 SELECT pg_size_pretty(pg_database_size('EPNprendeDB')) AS tamaño_total;

```

tamaño_total
8800 kB

- Control de crecimiento de registros por semana.

**Objetivo:** Identificar tablas con crecimiento acelerado.

**Implementación:**

**Crear tabla histórica:** Tabla diseñada para **almacenar versiones anteriores de los datos** de otra tabla principal, con el objetivo de mantener un **registro completo de cambios a lo largo del tiempo**.

**Programar tarea semanal con pgAgent en las tablas claves del sistema:**

```

34 -- Programar tarea semanal con pgAgent en las tablas claves del sistema
35 INSERT INTO auditoria.crecimiento_semanal (tabla, registros)
36 SELECT 'users', COUNT(*) FROM public.users;
37
38 INSERT INTO auditoria.crecimiento_semanal (tabla, registros)
39 SELECT 'offer_id', COUNT(*) FROM public.offers;
40
41 INSERT INTO auditoria.crecimiento_semanal (tabla, registros)
42 SELECT 'product_name', COUNT(*) FROM public.products;
43
44 INSERT INTO auditoria.crecimiento_semanal (tabla, registros)
45 SELECT 'category_id', COUNT(*) FROM public.categories;
46
47 INSERT INTO auditoria.crecimiento_semanal (tabla, registros)
48 SELECT 'report_id', COUNT(*) FROM public.reports;
49

```

**Visualizar tendencias de crecimiento en las tablas a monitorear:**

```

50
51 -- Vizualizar las tendencias
52 v SELECT tabla, fecha, registros
53 FROM auditoria.crecimiento_semanal
54 ORDER BY fecha DESC;
55
56

```

	tabla	fecha	registros
	text	date	bigint
1	users	2025-08-03	508
2	offer_id	2025-08-03	49
3	product_name	2025-08-03	499
4	category_id	2025-08-03	48
5	report_id	2025-08-03	48
6	category_id	2025-08-03	48

- Registro del uso de funciones, procedimientos, recursos.

**Objetivo:** Identificar cuellos de botella.

**Consultas claves**

**Funciones más usadas:**

**Recursos de CPU/RAM**

```

72 v SELECT
73     query,
74     calls,
75     total_exec_time AS total_cpu_ms,
76     mean_exec_time AS avg_cpu_ms,
77     rows,
78     100.0 * shared_blks_hit / NULLIF(shared_blks_hit + shared_blks_read, 0) AS cache_hit_ratio
79 FROM pg_stat_statements
80 ORDER BY total_exec_time DESC
81 LIMIT 10;

```

	query
	text
1	SELECT sum(pg_database_size(datname)) AS total
2	SELECT
3	-- We export stats for 10 non-system databases. Without this limit it is too
4	CREATE EXTENSION IF NOT EXISTS pg_stat_statements
5	SELECT state, to_char(state_change, \$1) AS state_change
6	-- NOTE: This is the 'internal' / 'machine-readable' version. This outputs the
7	select count(*) from pg_stat_replication where application_name != \$1
8	SELECT
9	select count(*) from pg_stat_activity where backend_type = \$1

Estos resultados muestran las 10 consultas SQL que más tiempo total de CPU han consumido en tu base de datos PostgreSQL, ordenadas de mayor a menor consumo.

Qué significa cada columna:

- **query**: El texto de la consulta SQL (algunas aparecen truncadas)
- **calls**: Cuántas veces se ha ejecutado esa consulta
- **total\_cpu\_ms**: Tiempo total de CPU consumido en milisegundos
- **avg\_cpu\_ms**: Tiempo promedio de CPU por ejecución en milisegundos
- **rows**: Cantidad total de filas procesadas
- **cache\_hit\_ratio**: Porcentaje de aciertos en caché (cuánto se leyó de memoria vs disco)

Consultas destacadas:

1. La primera consulta calcula el tamaño total de todas las bases de datos
2. Hay varias consultas de monitoreo del sistema (estado de réplicas, actividad)
3. Se menciona la extensión `pg_stat_statements` que es la que provee estos datos
4. Algunas consultas usan parámetros (\$1) en lugar de valores literales

Qué te indica esto:

- Las consultas de monitoreo aparecen porque probablemente se ejecutan con frecuencia
- No se ven consultas de aplicación problemáticas en este listado (solo aparecen consultas del sistema)
- El `cache_hit_ratio` se diría que se necesita más memoria para buffers (pero no se ven los valores específicos)

### **Estadísticas de las funciones ejecutadas:**

El resultado de la prueba arroja de en el momento no se han llamado a ninguna de las funciones, por lo tanto, el valor de las llamadas son null.



Data Output						Messages	Notifications
	funcid oid	schemaname name	funcname name	calls bigint	total_time_ms double precision	Showing	
17	24669	public	citext_le	[null]	[null]		
18	24670	public	citext_gt	[null]	[null]		
19	24671	public	citext_ge	[null]	[null]		
20	24678	public	citext_cmp	[null]	[null]		
21	24679	public	citext_hash	[null]	[null]		
22	24872	public	insertar_usuario	[null]	[null]		
23	24873	public	actualizar_estado_productos	[null]	[null]		
24	24874	public	eliminar_perfil_seguro	[null]	[null]		
25	24875	public	actualizar_precios_masivo	[null]	[null]		
26	24692	public	citext_smaller	[null]	[null]		
27	24693	public	citext_larger	[null]	[null]		
28	40960	public	insertar_reporte	[null]	[null]		
29	24877	public	calcular_descuento_promedio	[null]	[null]		
30	24878	public	determinar_popularidad	[null]	[null]		
31	24699	public	texticregexne	[null]	[null]		
32	24708	public	texticlike	[null]	[null]		
33	24709	public	texticnlike	[null]	[null]		
34	24710	public	texticregexeq	[null]	[null]		
35	24711	public	texticregexne	[null]	[null]		
36	24720	public	regex_match	[null]	[null]		
37	24721	public	regex_match	[null]	[null]		
38	24722	public	regex_matches	[null]	[null]		

## Simulación de Perfiles Profesionales

### Simulación de roles (Oficial de Seguridad)

Las actividades para realizar de acuerdo con el rol propuesto son:

#### La auditoría de roles

El resultado de la consulta muestra la tabla de los usuarios dentro del sistema, con sus respectivos permisos asignados a las diferentes tablas de la base de datos.

```

auditoria de roles:
SELECT grantee, privilege_type, table_name
FROM information_schema.role_table_grants;

```

	grantee name	privilege_type character varying	table_name name		grantee name	privilege_type character varying	table_name name
1	neondb_owner	INSERT	comment_users	25	neondb_owner	UPDATE	profile_photos
2	neondb_owner	SELECT	comment_users	26	neondb_owner	DELETE	profile_photos
3	neondb_owner	UPDATE	comment_users	27	neondb_owner	TRUNCATE	profile_photos
4	neondb_owner	DELETE	comment_users	28	neondb_owner	REFERENCES	profile_photos
5	neondb_owner	TRUNCATE	comment_users	29	neondb_owner	TRIGGER	profile_photos
6	neondb_owner	REFERENCES	comment_users	30	auditor_db	SELECT	profile_photos
7	neondb_owner	TRIGGER	comment_users	31	operador_db	INSERT	profile_photos
8	auditor_db	SELECT	comment_users	32	operador_db	SELECT	profile_photos
9	operador_db	INSERT	comment_users	33	operador_db	UPDATE	profile_photos
10	operador_db	SELECT	comment_users	34	neondb_owner	INSERT	product_photos
11	operador_db	UPDATE	comment_users	35	neondb_owner	SELECT	product_photos
12	neondb_owner	INSERT	users	36	neondb_owner	UPDATE	product_photos
13	neondb_owner	SELECT	users	37	neondb_owner	DELETE	product_photos
14	neondb_owner	UPDATE	users	38	neondb_owner	TRUNCATE	product_photos
15	neondb_owner	DELETE	users	39	neondb_owner	REFERENCES	product_photos
16	neondb_owner	TRUNCATE	users	40	neondb_owner	TRIGGER	product_photos
17	neondb_owner	REFERENCES	users	41	auditor_db	SELECT	product_photos
18	neondb_owner	TRIGGER	users	42	operador_db	INSERT	product_photos
19	auditor_db	SELECT	users	43	operador_db	SELECT	product_photos
20	operador_db	INSERT	users	44	operador_db	UPDATE	product_photos
21	operador_db	SELECT	users	45	neondb_owner	INSERT	reports
22	operador_db	UPDATE	users	46	neondb_owner	SELECT	reports
Total rows: 164				Query complete 00:00:03.792			

Rotación de Credenciales de los roles de usuario:

### Actualizar contraseñas de roles importantes en el sistema

```

19 -- Rotación de Credenciales de los roles de usuario
20 ALTER ROLE operador_db WITH PASSWORD 'EPNprendeAuditor@2025';

```

Data Output Messages Notifications

ALTER ROLE

Query returned successfully in 5 secs 729 msec.

### Revocar conexiones activas dentro de sistemas por los usuarios permitidos

```

22 -- REVOCAR CONEXIONES ACTIVAS DENTRO DE SISTEMAS
23 SELECT pg_terminate_backend(pid)
24 FROM pg_stat_activity
25 WHERE username = 'operador_db';

```

Data Output Messages Notifications

pg\_terminate\_backend  
boolean

Como usuario de seguridad se debe llevar un seguimiento de las actividades realizadas dentro del sistema por ello se registran las actividades en tablas de auditoría.

```
-- Creamos tabla para llevar el registro de la actividad realizada
CREATE TABLE IF NOT EXISTS auditoria.cambios_credenciales (
  id SERIAL PRIMARY KEY,
  rol TEXT NOT NULL,
  tipo_cambio TEXT NOT NULL CHECK (tipo_cambio IN ('ROTACION_CONTRASEÑA', 'CAMBIO_PERMISOS', 'CREACION_ROL')),
  ejecutado_por TEXT NOT NULL,
  fecha_cambio TIMESTAMPTZ DEFAULT CURRENT_TIMESTAMP,
  detalles TEXT,
  ip_origen INET
);
-- Asignar permisos (solo para roles autorizados)
GRANT INSERT, SELECT ON auditoria.cambios_credenciales TO admin_db;
GRANT SELECT ON auditoria.cambios_credenciales TO auditor_db;

-- Verificar el registro de la actividad
-- Consultar los últimos cambios
SELECT * FROM auditoria.cambios_credenciales
ORDER BY fecha_cambio DESC
LIMIT 5;
```

Data Output Messages Notifications								
Showing rows: 1 to 1 Page No: 1								
	id [PK] integer	rol text	tipo_cambio text	ejecutado_por text	fecha_cambio timestamp with time zone	detalles text	ip_origen inet	
1	1	operador_db	ROTACION_CONTRASEÑA	admin_seguridad	2025-08-03 07:07:51.668572+00	[null]	[null]	

### Simulación de roles (Usuario final)

Las actividades para realizar de acuerdo con el rol propuesto son:

**Consultas en el sistema sobre tablas específicas:**

```
51
52 -- Usuario Final con acceso controlado
53 -- Vistas permitidas:
54 CREATE VIEW view_product AS
55 SELECT product_name, product_description, product_price FROM products
56 WHERE product_stock = 'available';
57
58 GRANT SELECT ON view_product TO usuario_final;
59
60 -- Consulta típica en el sistema
61 SELECT * FROM view_product;
62
```

Data Output Messages Notifications			
Showing rows			
	product_name character varying (255)	product_description text	product_price numeric (10,2)
1	Auriculares Inalámbricos	Auriculares con cancelación de ruido	89.99
2	Chaqueta de Cuero	Chaqueta clásica para hombre	99.99
3	Sofá Cama	Sofá convertible en cama doble	399.99
4	Red de Voleibol	Red portátil para voleibol	29.99
5	Libro de Misterio	Novela de suspense intrigante	13.99
6	Mascarilla de Colágeno	Mascarilla rejuvenecedora	9.99
7	Cámara 360	Cámara para grabación panorámica	199.99

## Conclusiones y Recomendaciones

La base de datos diseñada para EPNprende en PostgreSQL cumple con los requerimientos técnicos exigidos: está normalizada, es segura, mantiene integridad referencial, soporta automatización mediante procedimientos y triggers, y fue validada en rendimiento y seguridad. Se recomienda a futuro:

- Integrar métricas avanzadas con herramientas de análisis (Power BI).
- Implementar un sistema de copias de seguridad automatizado.
- Mejora en el proceso y definición de la estructura del sistema de la base datos.
- Definir requerimientos iniciales para el sistema en el que se va a usar la base de datos.
- Hacer un análisis previo antes de realizar pruebas de rendimiento y seguridad.
- Implementar herramientas digitales y de automatización para monitoreo de recursos y rendimiento en la base datos.

## Anexos

Enlace del repositorio GitHub: [https://github.com/MariaGiron-code/ProyectoIntegrador\\_BasesDatos.git](https://github.com/MariaGiron-code/ProyectoIntegrador_BasesDatos.git)