

Arquitectura, schemas y objetos

<input checked="" type="checkbox"/> Checklist	<input type="checkbox"/>
# Día	2
☰ Estado	En progreso
📅 Fecha	@20 de diciembre de 2025
☰ Notas Técnicas	<p>EJERCICIO PRÁCTICO:</p> <p>1. Definir los siguientes conceptos:</p> <ul style="list-style-type: none">• Objeto vs Estructura: <p>Un objeto es cada entidad que existe dentro de un schema, como tablas, vistas, funciones, índices o secuencias.</p> <p>La estructura corresponde a la organización y jerarquía lógica de PostgreSQL, que incluye instancia, cluster, base de datos, schemas y objetos.</p> <ul style="list-style-type: none">• Schema public: Es el schema que PostgreSQL crea por defecto en cada base de datos. Si no se especifica un schema al crear objetos, estos se almacenan automáticamente en public, lo que puede generar desorden y problemas de seguridad en entornos productivos. <p>2. Ventajas de usar múltiples schemas:</p> <ul style="list-style-type: none">- Permiten segmentar y organizar la información de forma lógica.- Facilitan el control de acceso y la seguridad mediante permisos.- Permiten que múltiples equipos o aplicaciones trabajen sobre la misma base de datos sin conflictos.

- Mejoran el mantenimiento y la claridad de la estructura.

3. Riesgos de usar solo public:

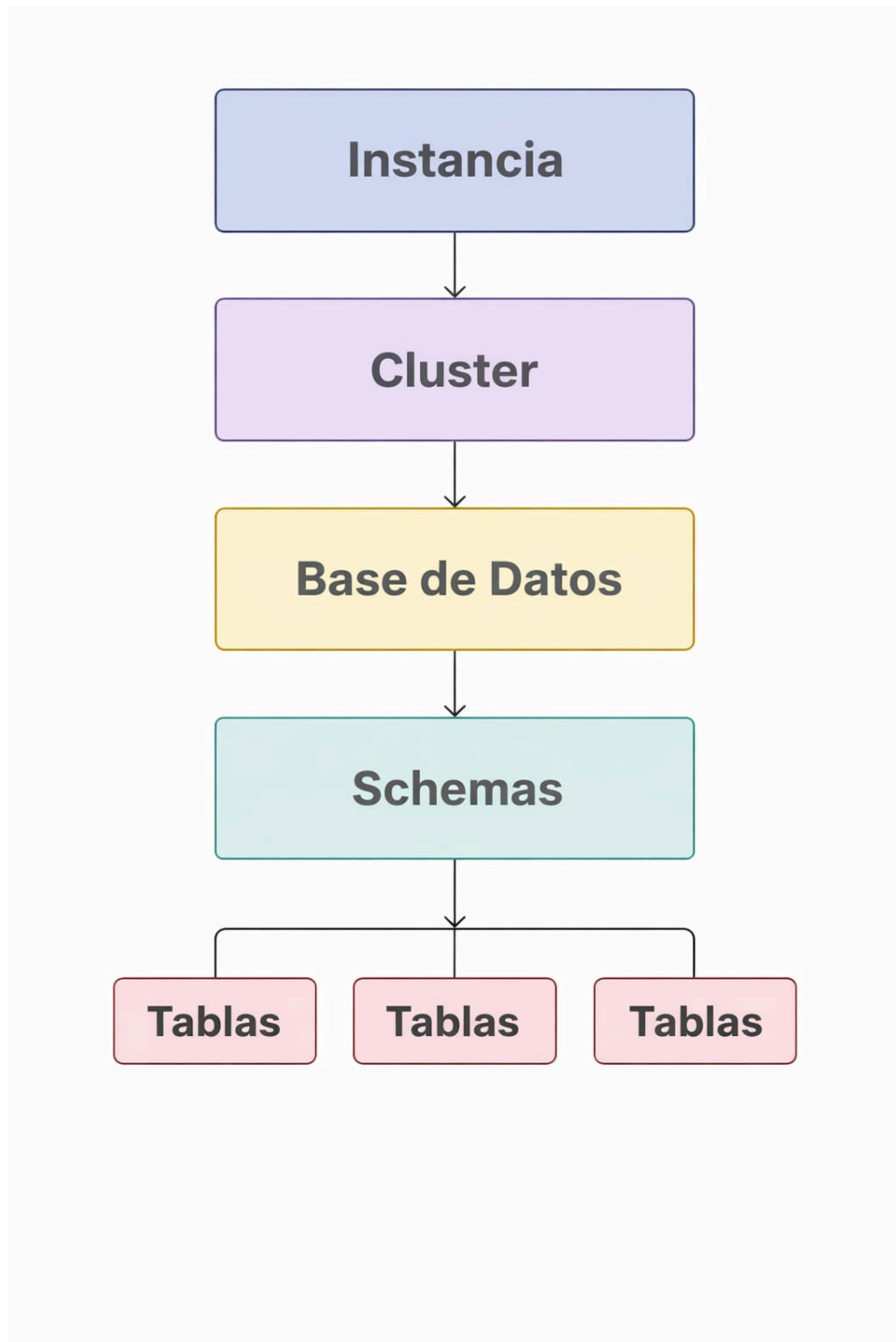
- Falta de organización en bases de datos grandes.
- Riesgo de modificar o eliminar objetos incorrectos.

Mayor exposición de datos sensibles. Confusión en entornos con múltiples aplicaciones o equipos.

4. Qué es search_path y por qué puede causar errores:

El **search_path** define el orden en el que PostgreSQL busca y ubica la información cuando no se le informa el Schema que se desea trabajar, este search_path se puede modificar en cada sesión, pero es una práctica que se debe hacer con sumo cuidado, pues si no se tiene claro en que schema se está trabajando puede generar errores en las consultas, y alterar información en los schemas que no desean cambiar.

Jerarquía de la Arquitectura:



search_path:

Define el orden de **Schemas** donde PostgreSQL busca objetos cuando no se especifica el schema. Permite simplificar consultas, pero un mal uso puede provocar acceso a objetos incorrectos.

28	
29	<code>show search_path;</code>
30	

Data Output	Messages	Notifications
<div> <div>≡+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>▼</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> </div>		
	<div>search_path</div> <div>text</div> <div>🔒</div>	
1	app	

El **search_path** se puede modificar, pero este cambio solo durará la sesión en la que se está trabajando.

39	<code>set search_path to public;</code>
40	
41	<code>show search_path;</code>
42	

Data Output	Messages	Notifications
<div> <div>≡+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>▼</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> <div>SQL</div> </div>		
	<div>search_path</div> <div>text</div> <div>🔒</div>	
1	public	

Por medio de **schema_name** puedo conocer y saber cuales son los schemas que hay dentro de la Base de Datos que estoy trabajando

```
SELECT schema_name
FROM information_schema.schemata;
```

```

43 SELECT schema_name
44 FROM information_schema.schemata;
45

```

Data Output
Messages
Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

SQL

	schema_name name	🔒
1	public	
2	app	
3	information_sche...	
4	pg_catalog	
5	pg_toast	

PostgreSQL incluye schemas del sistema como **public**, **information_schema** y **pg_catalog**. Estos permiten organizar objetos y consultar metadatos de la base de datos sin acceder directamente a tablas internas.

- **Schema Public:**

Es el schema por defecto

Lugar donde se crean las tablas si no se especifica el schema

ejemplo:

1. Se especifica el Schema, la tabla se creará en el Schema App:

```

create table app.datos(
tipo TEXT,
colonia TEXT,
habitaciones INTEGER,
area NUMERIC,
valor NUMERIC
)

```

2. No se especifica el Schema, por defecto la tabla se creará en el Schema Public:

```
create table datos(
tipo TEXT,
colonia TEXT,
habitaciones INTEGER,
area NUMERIC,
valor NUMERIC
)
```

- **Schema information_schema:**

Vista estándar SQL

Describe la estructura de la base de datos

Portable entre motores (Postgres, MySQL, etc.)

- **Schema pg_catalog:**

Catálogo interno del motor

Muy específico de PostgreSQL

NO modificar directamente

⚠ Diferencias clave (resumen claro)

Schema	Para qué sirve
public	Objetos por defecto
information_schema	Metadatos estándar
pg_catalog	Metadatos internos Postgres

Checklist:

- ✓ Estudio del tema
- ✓ Ejecución de scripts
- ✓ Evidencia guardada

- ✓ ~~Notas técnicas escritas~~

Tareas del día:

- ✓ ~~Revisar la arquitectura lógica de PostgreSQL~~
- ✓ ~~Identificar la jerarquía: Instance → Cluster → Database → Schema~~
- ✓ ~~Explorar schemas existentes (`public` , `information_schema` , `pg_catalog`)~~
- ✓ ~~Crear schemas personalizados (`admin` , `ventas`)~~
- ✓ ~~Crear tablas dentro de schemas específicos~~
- ✓ ~~Explorar y modificar el `search_path`~~
- ✓ ~~Consultar objetos usando nombre calificado (`schema.objeto`)~~
- ✓ ~~Eliminar tablas y schemas de forma controlada~~
- ✓ ~~Guardar scripts y notas en el repositorio GitHub~~

Recursos de estudio:

- https://youtu.be/TAr47SP1dYw?si=VO_CLpp4Ntnnt9tv