UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

ABD Project (Sistema de Asistencia)

Administración de Bases de Datos

Nombre del proyecto: ABD Project

Integrantes del equipo: Mata Guerra David - 325797

Docente: Melchor Leal

Fecha de Entrega: 3 de diciembre del 2025

Documentación General del Sistema de Asistencia.

1 Arquitectura del sistema gestor de base de datos

1.1 Características del DBMS utilizado

- Versión de PostgreSQL utilizada: 16.10 (instalada sobre Ubuntu 24.04 dentro de un contenedor Docker).
- Características relevantes del proyecto:
  - Motor relacional con soporte transaccional completo (ACID).
  - Soporte de claves primarias, foráneas, restricciones y vistas, utilizado para el modelado de la base de datos de asistencia.
  - Soporte de funciones y procedimientos almacenados en SQL, aprovechados en el archivo `procedimientos_asistencia.sql`.
  - Mecanismo de triggers, utilizado para automatizar lógica de negocio definida en `triggers_asistencia.sql`.
  - Mecanismo de índices configurables, documentados en `indices_asistencia.sql`, para optimizar consultas frecuentes.

1.2 Características del DBMS utilizado

- Versión de PostgreSQL utilizada: 16.10, ejecutándose sobre Ubuntu 24.04 dentro de un contenedor Docker.
- Se trata de un sistema gestor de bases de datos relacional, que permite trabajar con tablas relacionadas, vistas, índices y procedimientos almacenados.
- En este proyecto se aprovechan estas características para modelar y gestionar la base de datos de asistencia (tablas, índices, procedimientos, triggers y vistas definidos en los scripts de la carpeta `db-arquitecture/`).

1.3 Estructura de memoria y procesos de la instancia

- Explicación general de los procesos principales de PostgreSQL:
  - Un proceso maestro `postgres` que coordina la instancia del servidor.
  - Procesos auxiliares como *checkpointer*, *writer*, *wal writer* y *autovacuum*, encargados de escritura en disco, manejo del WAL y mantenimiento automático de tablas.
- Mención de los archivos/directorios importantes:
  - Directorio de datos: `/var/lib/postgresql/16/main` dentro del contenedor (configuración por defecto del paquete de Ubuntu).
  - Archivo de configuración principal: `postgresql.conf`, donde se definen parámetros como puerto, memoria compartida, autovacuum, etc.
  - Archivo de control de accesos: `pg_hba.conf`, donde se especifican los métodos de autenticación y orígenes permitidos.

1.4 Estructura de memoria y procesos de la instancia

- PostgreSQL se ejecuta como un servicio dentro del contenedor y mantiene varios procesos en segundo plano que se encargan de atender conexiones y guardar la información en disco.
- Los datos de la base se almacenan en un directorio interno de PostgreSQL (por defecto en `/var/lib/postgresql/16/main` dentro del contenedor).
- La configuración principal del servidor se realiza a través de los archivos `postgresql.conf` y `pg_hba.conf`, donde se ajustan parámetros como el puerto y las reglas de acceso.

1.5 Instalación y configuración básica

- Resumen del entorno donde corre la BD:
  - Sistema operativo base: Ubuntu 24.04 ejecutándose dentro de un contenedor Docker.
  - El contenedor se construye a partir del `Dockerfile` ubicado en `ubuntu-container/` y se orquesta mediante `docker-compose.yml`.
  - El servicio de PostgreSQL se instala mediante los paquetes `postgresql`, `postgresql-contrib` y `postgresql-client`.
- Variables de ambiente relevantes:
  - `DEBIAN_FRONTEND=noninteractive` para evitar prompts interactivos durante la instalación.
  - `TZ=UTC` para configurar la zona horaria del sistema dentro del contenedor.
- Configuraciones mínimas realizadas para que el sistema funcione:
  - Se modifica la contraseña del usuario `postgres` mediante: `ALTER USER postgres WITH PASSWORD 'postgres';`
  - El servicio PostgreSQL se expone en el puerto `5432` del contenedor y se mapea al puerto `5432` del host mediante Docker Compose.
  - El contenedor se mantiene en ejecución iniciando el servicio con `service postgresql start` y luego un comando de espera (`sleep infinity`) para permitir el acceso continuo durante las pruebas.

1.6 Instalación y configuración básica

- La base de datos se ejecuta en un contenedor Docker basado en Ubuntu 24.04, definido en la carpeta `ubuntu-container/` mediante un `Dockerfile` y un archivo `docker-compose.yml`.
- Durante la construcción de la imagen se instalan los paquetes de PostgreSQL y se configura la contraseña del usuario administrador `postgres` con el valor `postgres`.
- El puerto `5432` del contenedor se publica en el mismo puerto del equipo host, lo que permite conectar desde herramientas externas usando la dirección `localhost:5432`.

1.7 Clientes e IDEs de acceso

- Herramientas utilizadas:
  - `psql`, el cliente de línea de comandos de PostgreSQL, instalado dentro del contenedor y utilizado para ejecutar los scripts SQL del proyecto (inicialización, índices, procedimientos, triggers, vistas, seguridad, etc.).
  - Opcionalmente, un IDE gráfico como *pgAdmin* o *DBeaver* desde el host, conectándose a `localhost:5432` con el usuario `postgres` y contraseña `postgres`.
- Captura de pantalla de la conexión a la BD de asistencia:
  - Imagen

2 Diseñado de la base de datos

2.1 Modelo Entidad-Relación

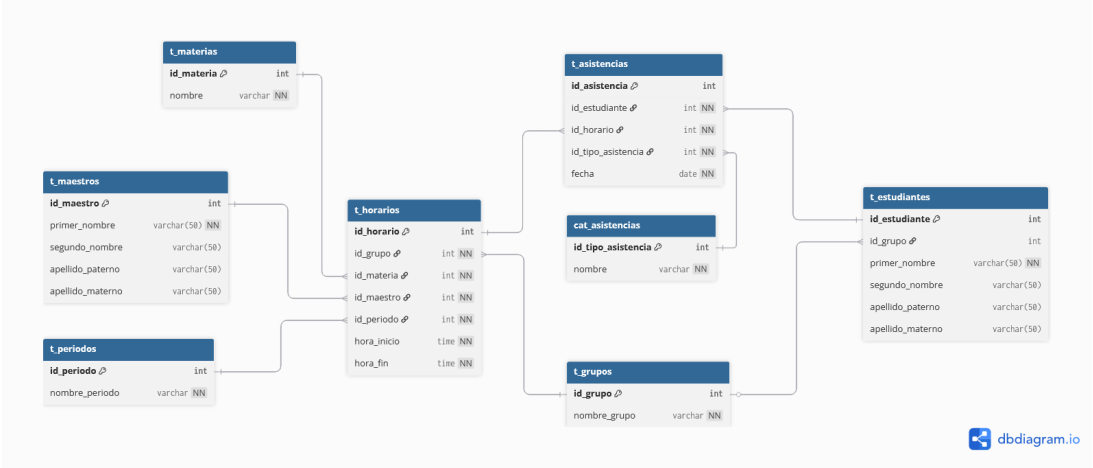


Figura 1: Diagrama Entidad-Relación de la base de datos de asistencia

2.2 Modelo Físico de la base de datos

Explicación lógica de las tablas

- t_estudiantes: almacena la información básica de los estudiantes, incluyendo su grupo asignado.
- t_maestros: almacena la información básica de los maestros.
- t_materias: contiene las materias disponibles en la institución.
- t_grupos: define los grupos o clases a los que pertenecen los estudiantes.
- t_periodos: registra los periodos académicos.
- t_horario: relaciona grupos, materias, maestros y periodos con horarios específicos.
- cat_asistencias: catálogo de tipos de asistencia (asistencia, retardo, falta, justificada, etc.).

- t_asistencias: registra las asistencias de los estudiantes por horario y fecha.

2.2.1. init_asistencia.sql

El script init_asistencia.sql contiene las instrucciones SQL para crear las tablas mencionadas anteriormente, junto con sus claves primarias, foráneas y restricciones necesarias para mantener la integridad referencial. Además, incluye las relaciones entre las tablas que reflejan el modelo entidad-relación diseñado para la base de datos de asistencia. Y finalmente población de la bd con un mínimo de

- 10 estudiantes
- 5 maestros
- 5 materias
- 3 grupos
- 1 periodo
- 1 semana de registros de asistencia

```
root@55d8b1e32762:/work# sudo -u postgres psql -d asistencia_db -f /work/init_asistencia.sql
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
INSERT 0 1
INSERT 0 3
INSERT 0 4
INSERT 0 5
INSERT 0 10
INSERT 0 4
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 10
INSERT 0 10
INSERT 0 10
INSERT 0 10
INSERT 0 10
root@55d8b1e32762:/work#
```

Figura 2: Tablas de la base de datos de asistencia

2.3 Optimización de la base de datos

Definicion de indices

- Índice para búsqueda rápida de asistencia por estudiante (igualdad).
  Se usa un índice **HASH** sobre t_asistencias(id_estudiante) para optimizar consultas de igualdad como: `SELECT * FROM t_asistencias WHERE id_estudiante = ?`. Los índices hash son ideales para lookups exactos.
- Índice para consultas por fecha (rangos temporales).
  Se usa un índice **BRIN** sobre t_asistencias(fecha) para acelerar rangos por día/semana/mes en tablas que crecen cronológicamente. BRIN es liviano y eficiente con datos correlacionados por bloque.
- Índice para consultas por grupo y materia (filtrado y joins vía horario).
  Se usa un índice **BTREE compuesto** sobre t_horarios(id_grupo, id_materia) para filtrar rápidamente combinaciones de grupo+materia y mejorar los joins posteriores con t_asistencias por id_horario.

indices_asistencia.sql

El script indices_asistencia.sql contiene las instrucciones SQL para crear los índices definidos anteriormente en la base de datos de asistencia.

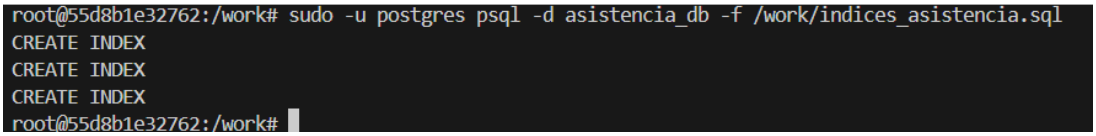


Figura 3: Creacion de indices

Explicación de las decisiones tomadas en los índices:

- **¿Por qué se eligió cada índice?**
  HASH para igualdad en id_estudiante; BRIN para rangos en fecha con crecimiento cronológico; BTREE compuesto para combinaciones frecuentes id_grupo+id_materia.
- **¿Qué tipo de consultas se benefician?**
  HASH: `WHERE id_estudiante = ?`. BRIN: `WHERE fecha BETWEEN ? AND ?` o por día. BTREE compuesto: `WHERE id_grupo = ? AND id_materia = ?` y joins hacia t_asistencias por id_horario.
- **Impacto en inserciones/actualizaciones:**
  Los índices son primordiales en una buena base de datos, además el tipo de indice compuesto aunque no lo hayamos visto a profundidad en la clase, el maestro nos mencionó de su utilidad e importancia.

2.4 Estructuras avanzadas de la base de datos

Vistas

En el archivo vistas_asistencia.sql se definen las siguientes vistas para facilitar consultas.

```
root@55d8b1e32762:/work# sudo -u postgres psql -d asistencia_db -f /work/vistas_asistencia.sql
CREATE VIEW
CREATE VIEW
CREATE VIEW
CREATE VIEW
```

Figura 4: Creacion de vistas

Procedimientos almacenados

En el archivo procedimientos_asistencia.sql se definen los procedimientos almacenados para encapsular las reglas de negocio.

```
root@55d8b1e32762:/work# sudo -u postgres psql -d asistencia_db -f /work/procedimientos_asistencia.sql
CREATE PROCEDURE
```

Figura 5: Creacion de procedimientos almacenados

Triggers

```
root@55d8b1e32762:/work# sudo -u postgres psql -d asistencia_db -f /work/triggers_asistencia.sql
CREATE TABLE
CREATE FUNCTION
CREATE TRIGGER
CREATE TABLE
CREATE FUNCTION
CREATE TRIGGER
```

Figura 6: Creacion de triggers

Se adjunta imagen de las respectivas tablas de logs.

	id_log [PK] integer	accion character varying (20)	usuario character varying (50)	fecha timestamp without time zone
1	1	INSERT	postgres	2025-12-02 18:22:16.645513

Figura 7: Tabla de log para t_maestros

	id_log [PK] integer	accion character varying (20)	usuario character varying (50)	fecha timestamp without time zone
1	1	INSERT	postgres	2025-12-02 18:19:58.589721

Figura 8: Tabla de log para t_asistencias

Índices

Se hace referencia a el apartado de indices en la seccion de optimizacion

3 Modelos de seguridad en bases de datos

3.1 Seguridad a nivel de usuarios y roles

Modelo de seguridad

- Roles mínimos:
  - rol_profesor: puede consultar sus grupos y registrar asistencias.
  - rol_control_escolar: puede consultar reportes globales, modificar estudiantes/grupos.
  - rol_consulta: solo lectura de vistas de reporte.
- Usuarios de ejemplo:
  - u_profesor_demo (asignado a rol_profesor).
  - u_control_escolar_demo (asignado a rol_control_escolar).
- Script seguridad_asistencia.sql
  - CREATE ROLE / CREATE USER o CREATE ROLE ... LOGIN.
  - GRANT y REVOKE a nivel base de datos, esquema, tablas y vistas. o Comentarios que expliquen qué puede y qué no puede hacer cada rol/usuario.

Implementación y pruebas

Se implementaron los roles y usuarios mencionados en el script seguridad_asistencia. Los roles definen permisos mínimos necesarios para cada tipo de usuario, siguiendo el principio de privilegios mínimos.
Pruebas de sanidad de cada usuario/rol:

```
PS C:\Projects\abd-project> docker exec -it 55d8b1e32762 bash -lc "psql -h localhost -U u_control_escolar_demo -d asistencia_db"
Password for user u_control_escolar_demo:
psql (16.10 (Ubuntu 16.10-0ubuntu0.24.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

asistencia_db=> SELECT * FROM vw_asistencia_estudiante_detalle
```

Figura 9: Pruebas de seguridad rol de control escolar

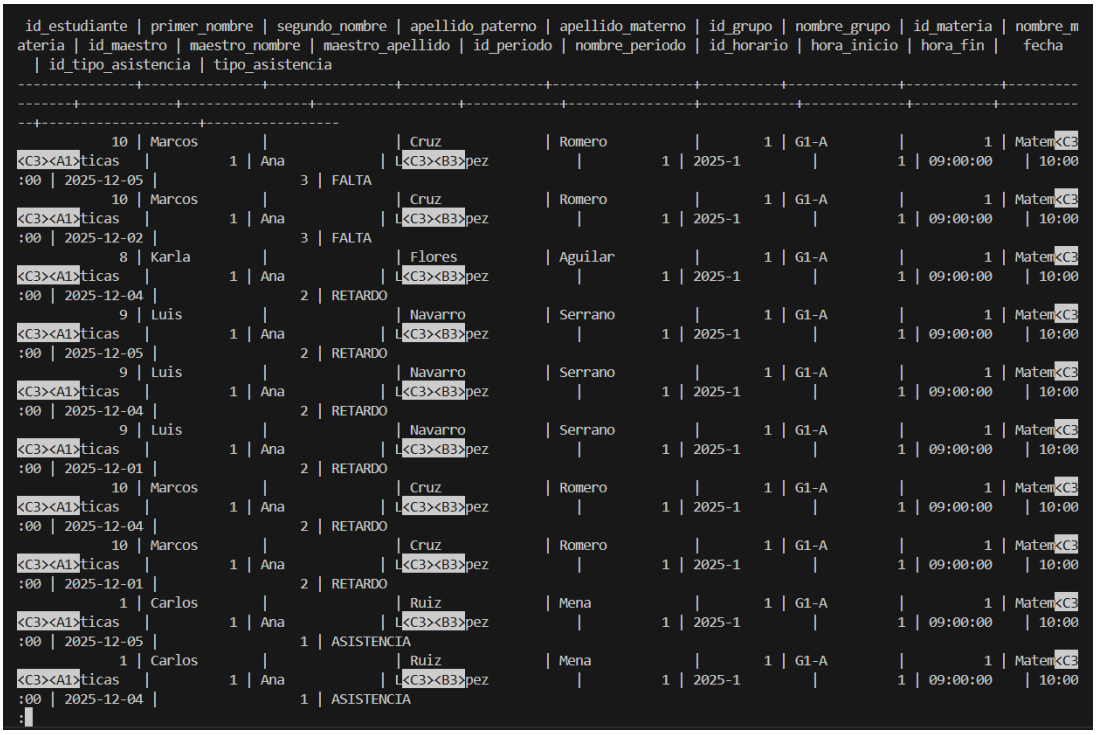


Figura 10: Pruebas de seguridad rol de control escolar

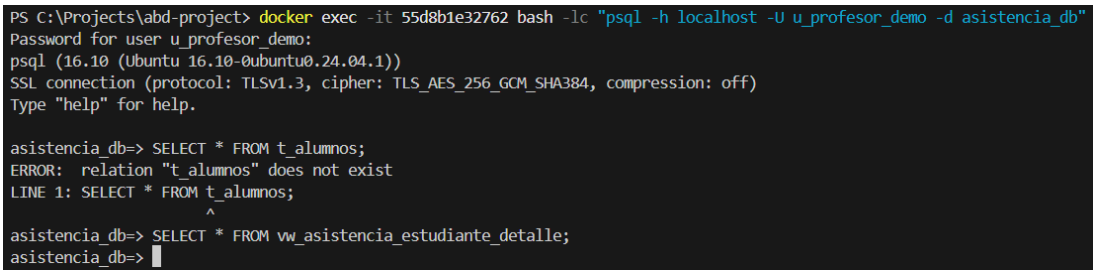
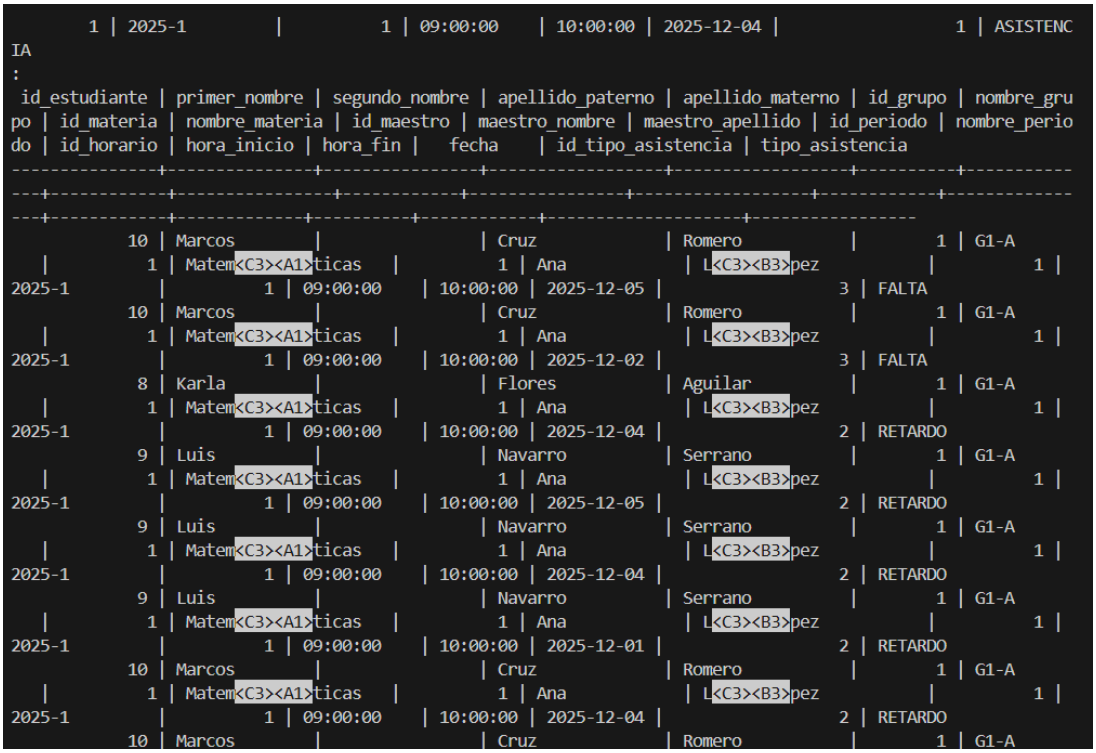


Figura 11: Pruebas de seguridad rol de profesor



```
      1 | 2025-1        |         1 | 09:00:00    | 10:00:00 | 2025-12-04 |                1 | ASISTENC
IA
:
 id_estudiante | primer_nombre | segundo_nombre | apellido_paterno | apellido_materno | id_grupo | nombre_gru
po | id_materia | nombre_materia | id_maestro | maestro_nombre | maestro_apellido | id_periodo | nombre_perio
do | id_horario | hora_inicio | hora_fin |   fecha    | id_tipo_asistencia | tipo_asistencia
--------------+---------------+----------------+------------------+------------------+----------+----------
---+------------+----------------+------------+----------------+------------------+------------+------------
---+------------+-------------+----------+------------+--------------------+----------------
          10 | Marcos        |                | Cruz             | Romero           |        1 | G1-A
   |         1 | Matem<C3><A1>ticas   |         1 | Ana            | L<C3><B3>pez          |          1 |
2025-1        |         1 | 09:00:00    | 10:00:00 | 2025-12-05 |                  3 | FALTA
          10 | Marcos        |                | Cruz             | Romero           |        1 | G1-A
   |         1 | Matem<C3><A1>ticas   |         1 | Ana            | L<C3><B3>pez          |          1 |
2025-1        |         1 | 09:00:00    | 10:00:00 | 2025-12-02 |                  3 | FALTA
           8 | Karla         |                | Flores           | Aguilar          |        1 | G1-A
   |         1 | Matem<C3><A1>ticas   |         1 | Ana            | L<C3><B3>pez          |          1 |
2025-1        |         1 | 09:00:00    | 10:00:00 | 2025-12-04 |                  2 | RETARDO
           9 | Luis          |                | Navarro          | Serrano          |        1 | G1-A
   |         1 | Matem<C3><A1>ticas   |         1 | Ana            | L<C3><B3>pez          |          1 |
2025-1        |         1 | 09:00:00    | 10:00:00 | 2025-12-05 |                  2 | RETARDO
           9 | Luis          |                | Navarro          | Serrano          |        1 | G1-A
   |         1 | Matem<C3><A1>ticas   |         1 | Ana            | L<C3><B3>pez          |          1 |
2025-1        |         1 | 09:00:00    | 10:00:00 | 2025-12-04 |                  2 | RETARDO
           9 | Luis          |                | Navarro          | Serrano          |        1 | G1-A
   |         1 | Matem<C3><A1>ticas   |         1 | Ana            | L<C3><B3>pez          |          1 |
2025-1        |         1 | 09:00:00    | 10:00:00 | 2025-12-01 |                  2 | RETARDO
          10 | Marcos        |                | Cruz             | Romero           |        1 | G1-A
   |         1 | Matem<C3><A1>ticas   |         1 | Ana            | L<C3><B3>pez          |          1 |
2025-1        |         1 | 09:00:00    | 10:00:00 | 2025-12-04 |                  2 | RETARDO
          10 | Marcos        |                | Cruz             | Romero           |        1 | G1-A
```

Figura 12: Pruebas de seguridad rol de profesor

3.2 Backups y recuperación con cron

- Script de respaldo lógico automático en bash: ver `ubuntu-container/work/backup_asistencia.s` Utiliza `pg_dump -F c`, nombra el archivo con fecha/hora (`asistencia_db_YYYYMMDD_HHMM.backu` guarda en `/var/backups/postgres` y elimina respaldos mayores a 7 días con `find ... -mtime +7 -delete`.
- Cronjob para ejecutar el respaldo diario: ver `ubuntu-container/work/cron_asistencia.txt`. La línea de crontab programada es `45 19 * * * root /work/backup_asistencia.sh` (diario a las 19:45).

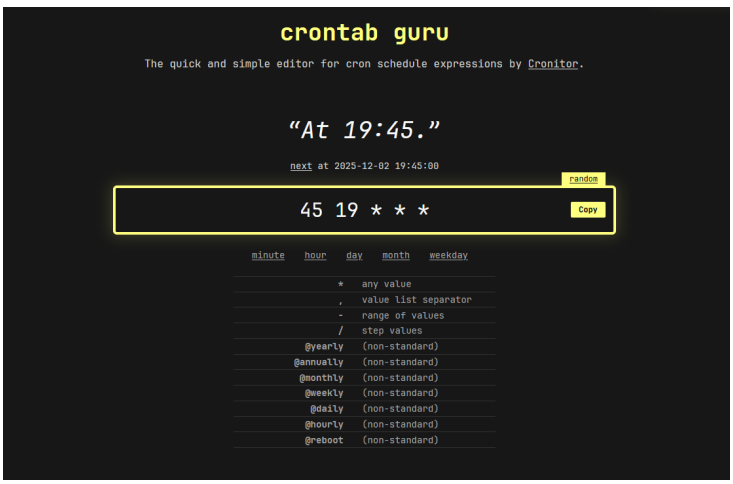


Figura 13: Screenshot de crontab guru para la configuración del cronjob

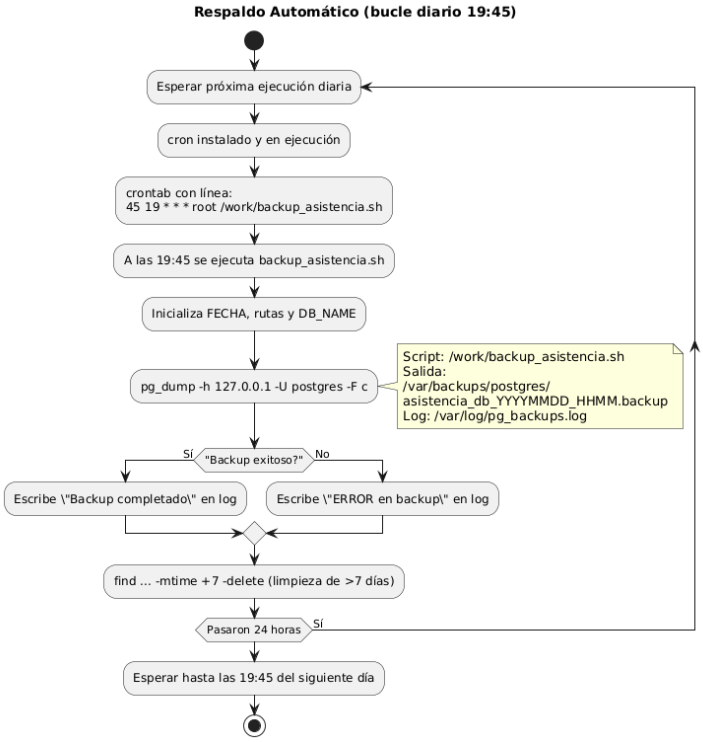


Figura 14: Diagrama de flujo del proceso de respaldo automático

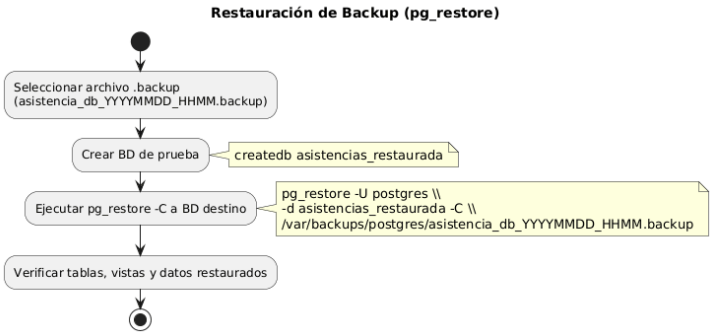


Figura 15: Diagrama de flujo del proceso de restauración desde respaldo

4 Conclusiones

Como conclusión, se logró implentar de manera exitosa un sistema de gestión de asistencia utilizando PostgreSQL como sistema gestor de bases de datos. A lo largo del desarrollo del proyecto, se aplicaron diversos conceptos fundamentales de bases de datos, tales como el diseño relacional, la normalización, la creación de índices para optimizar consultas, y la implementación de procedimientos almacenados y triggers para automatizar tareas y mantener la integridad de los datos. Sin embargo también hubo problemas, ya que no se pudo comprobar que la restauración de los respaldos funcionara correctamente, debido a que el contenedor de PostgreSQL no tenía acceso a los archivos de respaldo generados por el script de backup. Pienso que se debio a alguna configuracion del volumen de ubuntu.