Taller I: Definición de Datos en SQL

Objetivos

a) Conocer las principales funcionalidades del entorno de trabajo *pgAdmin IV*.
b) Crear, documentar y guardar scripts SQL.
c) Familiarizarse con los siguientes comandos del lenguaje SQL:

```
CREATE DATABASE                 | DROP DATABASE
CREATE TABLE                    | DROP TABLE
INSERT INTO ... VALUES          | DELETE FROM ...
COPY
```

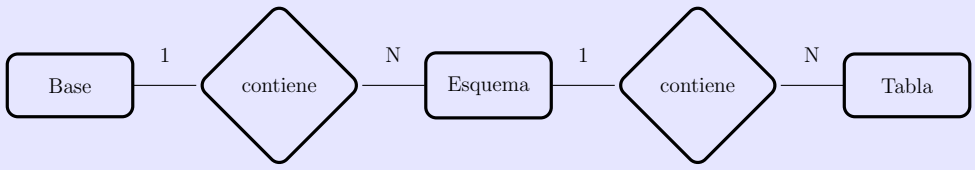
1. *(Instalación)* Instale el servidor de bases de datos *PostgreSQL* y el administrador *pgAdmin*. Bajo Ubuntu, basta con instalar los paquetes `postgresql` y `pgadmin4` con la herramienta `apt-get`.

2. *(Establecimiento de una conexión)*. Ingrese al administrador *pgAdmin*. Tómese unos minutos para explorar sus íconos principales y menúes. Cree una conexión al servidor de *PostgreSQL* local a su computadora, que escucha por defecto en el puerto `5432` de `localhost`.

3. *(Creación de una nueva base)* Utilice el comando *New Database...* para crear una nueva base de nombre `Transporte`. Una vez creada, navegue por la estructura *Transporte* → *Schemas* → *Public* → *Tables*.

Nota

Una base en *Postgres* está conformada por un conjunto de `esquemas (schemas)`, y un esquema está formado por un conjunto de `tablas` (aunque un esquema también contiene otros objetos, como funciones, vistas, tipos de dato, etc.).



La diferencia entre bases y esquemas es que una conexión a un servidor de *PostgreSQL* se realiza a una base específica, aunque puede trabajar con más de un esquema de dicha base. Los esquemas son una separación lógica de las tablas, mientras que tablas en bases distintas no pueden verse entre sí.
Cuando instalamos *PostgreSQL*, automáticamente se configura una base `postgres` con un esquema `public`.

4. *(Creación de una tabla)* Reconozca el ícono que abre el editor de SQL. Escriba un script con consultas de tipo `CREATE TABLE` en lenguaje SQL a los efectos de crear las siguientes dos tablas:

   - `paradas(cod_parada, longitud, latitud, tipo_parada, calle, altura, entre1, entre2)`
   - `colectivos_por_parada(cod_parada, num_colectivo)`

   Nota: Considere que la longitud y latitud son números de punto flotante, que los nombres de calles y tipos de parada son cadenas de longitud variable de 50 caracteres como máximo, y que el resto de los atributos son enteros positivos. Considere también que el código de parada, el número de colectivo, la latitud y la longitud no pueden tener valor nulo.

   Sugerencias

   Escriba sus consultas SQL en varias líneas y utilizando tabulaciones, de manera que sean más legibles. El Anexo incluye una consulta de creación de tabla a modo de ejemplo. Puede introducir comentarios al final de las líneas con los caracteres '`--`' para documentar su script.

   Guarde el script en un archivo con extensión `.sql`. Visualice una de las tablas creadas en el explorador de objetos. Observe las opciones que ofrece el menú contextual, en particular las opciones dentro de *New Object*, *Delete/Drop*, *Scripts* y *View Data*.

5. *(Eliminación de tablas)* Mejore el script agregándole al inicio del archivo una consulta `DROP` para eliminar las tablas. Ejecute el script.

6. *(Inserción manual de datos)* Abra un nuevo script utilizando la funcionalidad *Scripts* → *INSERT Script* y complételo para agregar una fila de datos a la tabla `paradas`. Guarde el script en un archivo, ejecútelo, y utilice la funcionalidad *View Data* para ver la tabla con los datos insertados.

   Nota

   En SQL los *strings* se delimitan con comillas simples (''').

7. *(Eliminación de datos)* Abra un nuevo script utilizando la funcionalidad *Scripts* → *DELETE Script* y complételo para eliminar la fila insertada. Guarde el script en un archivo, ejecútelo, y utilice la funcionalidad *View Data* para verificar que la tabla esté ahora vacía.

8. *(Carga de datos desde archivos)* El comando `COPY` de *PostgreSQL* es un comando *no estándar* que permite cargar una tabla desde un archivo *.csv* y viceversa. Utilice el comando `COPY` para cargar en cada una de las tablas los datos de los archivos provistos en el Campus para este taller: `paradas.csv` y `colectivosPorParada.csv`. Luego utilice la funcionalidad *View Data* para examinar las tablas.

9. *(SQL dump y exportación de datos)* Exporte cada una de las tablas creadas a un archivo `.csv` con el comando `COPY`. Exporte luego toda la base de datos a un *SQL dump* utilizando el comando `pg_dump` de *PostgreSQL*, y observe la estructura del archivo que se generó.
   Nota: Un *SQL dump* es un script con consultas SQL que permite permite reconstruir la base de datos desde cero. Sirve entre otras cosas como backup de la misma.

Anexo

Para el script de creación de tablas puede guiarse por el siguiente ejemplo base:

```
DROP TABLE IF EXISTS mi_tabla;
CREATE TABLE mi_tabla (
    c1      VARCHAR(10)       NOT NULL,
    c2      INT               NOT NULL,
    c3      NUMERIC(15,12)    NOT NULL,
    c4      INT,
    c5      ...               ...,
    ...     ...               ...
);
```

Y el siguiente es un uso básico del comando `COPY`:

```
COPY mi_tabla
FROM path_archivo
DELIMITER ';'
CSV HEADER         --para indicar que saltee la primera línea
ENCODING 'LATIN1';
```

Le sugerimos también leer las siguientes entradas de la documentación de *PostgreSQL*, y en particular los ejemplos que las mismas incluyen:

- https://www.postgresql.org/docs/current/static/datatype.html (DATATYPE-TABLE)
- https://www.postgresql.org/docs/current/static/sql-createtable.html (CREATE TABLE)
- https://www.postgresql.org/docs/current/static/sql-insert.html (INSERT INTO)
- https://www.postgresql.org/docs/current/static/sql-copy.html (COPY)

Los datos de paradas de colectivos fueron extraídos y adaptados de:

- https://data.buenosaires.gob.ar/dataset/colectivos-paradas