
Proyectos de computación aplicados a I.E.

Tarea # 2

David Antonio Rodas Alvarez

202010039

Fecha: Guatemala, 31 de julio del 2024

Tarea # 2*

David Antonio, Rodas Alvarez, 202010039^{1, **}

¹Escuela de Ingeniería Mecánica Eléctrica, Universidad de San Carlos, Guatemala.
(Dated: 3 de agosto de 2024)

I. RESUMEN

En la segunda tarea del curso, se llevó a cabo el procedimiento para crear y gestionar una base de datos utilizando PostgreSQL. Con el objetivo principal de desarrollar una base de datos. El proceso comenzó con la configuración y armado de la base de datos en PostgreSQL. Una vez establecida la base de datos, se utilizaron los lenguajes de programación Octave y Python para interactuar con ella. A través de estos lenguajes, se logró conectar con la base de datos y ejecutar consultas para extraer la información deseada. En particular, se diseñó una consulta que genera un listado compuesto por dos columnas: una para el nombre del alumno y otra para su número de registro académico.

II. MARCO TEÓRICO

Base de datos

Las bases de datos son colecciones estructuradas y sistemáticas de datos almacenados electrónicamente, que pueden incluir diversos tipos de información como texto, números, imágenes, videos y archivos. El software utilizado para gestionar estas bases de datos se denomina Sistema de Gestión de Bases de Datos (DBMS), y facilita el almacenamiento, recuperación y edición de los datos. En el contexto de sistemas informáticos, el término base de datos puede referirse tanto al DBMS como al sistema de base de datos y las aplicaciones asociadas.

Importancia de las Bases de Datos

Las bases de datos son fundamentales para las organizaciones debido a su capacidad para manejar grandes volúmenes de información, garantizar la integridad y seguridad de los datos, y permitir análisis avanzados. Son esenciales para las operaciones internas, el almacenamiento de interacciones con clientes y proveedores, y la gestión de información administrativa y especializada. Ejemplos incluyen sistemas de bibliotecas digitales, sistemas de reserva de viajes y sistemas de inventario.

1. **Escalabilidad:** Las bases de datos pueden manejar grandes cantidades de datos y escalar de manera eficiente.
2. **Integridad de los datos:** Incorporan reglas y condiciones para mantener la coherencia de los datos.
3. **Seguridad:** Ofrecen mecanismos para cumplir con requisitos de privacidad y conformidad.
4. **Análisis de datos:** Facilitan el análisis de tendencias, patrones y la realización de predicciones.

Tipos de Bases de Datos

Las bases de datos se pueden clasificar según su contenido, ámbito de aplicación y aspectos técnicos, como la estructura y el tipo de interfaz. Los modelos de bases de datos muestran la estructura lógica y definen las relaciones y reglas para el almacenamiento y manipulación de datos. A continuación, se describen los principales modelos de bases de datos:

1. Jerárquicas: Utilizan una estructura de árbol con relaciones entre registros principales y secundarios.
2. De red: Permiten múltiples relaciones entre registros principales y secundarios.
3. Relacionales: Organizan los datos en tablas y son populares por su flexibilidad y compatibilidad con hardware avanzado.
4. Orientadas a objetos: Tratan los datos como objetos, reflejando la programación orientada a objetos.
5. NoSQL: No utilizan relaciones tabulares y están diseñadas para arquitecturas distribuidas y escalabilidad horizontal.

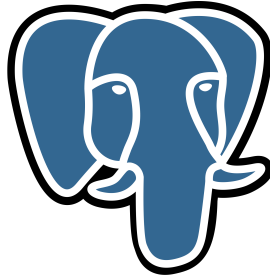
PostgreSQL

Tiene sus raíces en el proyecto POSTGRES, iniciado en 1986 en la Universidad de California en Berkeley bajo la dirección del profesor Michael Stonebraker. POSTGRES fue concebido como una mejora del sistema Ingres, también desarrollado en Berkeley, con la intención de añadir soporte para objetos más complejos y proporcionar mayor extensibilidad. El nombre POSTGRES es un acrónimo de *Post Ingres* que refleja su objetivo de ser un sucesor avanzado de Ingres. Este proyecto introdujo varios conceptos innovadores en la gestión de bases de datos, como la capacidad de definir tipos de datos personalizados y el uso de reglas para gestionar la lógica de las bases de datos.

* Escuela de Ingeniería Mecánica Eléctrica

** e-mail: 3711111370101@ingenieria.usac.edu.gt

Figura 1: Logo de PostgreSQL



Fuente: Elaboración Propia, 2024.

Evolución y mejoras:

Desde su concepción, POSTGRES ha evolucionado significativamente. En 1996, el proyecto fue renombrado a PostgreSQL para reflejar su soporte completo de SQL, el lenguaje estándar para la gestión de bases de datos. A lo largo de los años, PostgreSQL ha incorporado numerosas mejoras y nuevas características. Entre los hitos importantes se encuentran la introducción del soporte para transacciones ACID, recuperación ante fallos mediante WAL (Write-Ahead Logging), y la capacidad de manejar grandes volúmenes de datos y consultas complejas. La comunidad de desarrolladores ha jugado un papel crucial en su evolución, contribuyendo a hacer de PostgreSQL una de las bases de datos más robustas y versátiles disponibles hoy en día.

Una de las características más destacadas de PostgreSQL es su extensibilidad. Los usuarios pueden añadir nuevos tipos de datos, definir funciones y operadores personalizados, y crear índices específicos para sus necesidades. Esta capacidad de personalización permite adaptar PostgreSQL a una amplia variedad de aplicaciones y casos de uso específicos, desde bases de datos científicas y financieras hasta aplicaciones web de alto rendimiento. Además, los desarrolladores pueden escribir extensiones en múltiples lenguajes de programación, incluyendo PL/pgSQL, Python, y C, lo que amplía aún más las posibilidades de personalización y optimización.

III. CÓDIGO

A. Octave

```
pkg load database;
conn = pq_connect (setdbopts ("dbname", "postgres", "host",
    "localhost", "port", "5432", "user", "postgres",
    "password", "Lobodefuego01"));
N = pq_exec_params (conn, "select * from T202010039;")
```

B. Python

```
import psycopg2

try:
    connection = psycopg2.connect(
        host='localhost',
        user='postgres',
        password='Lobodefuego01',
        database='postgres'
    )

    print("Conexion exitosa.")
    cursor = connection.cursor()
    cursor.execute("SELECT version()")
    row = cursor.fetchone()
    cursor.execute("SELECT * from T202010039")
    rows = cursor.fetchall()
    for row in rows:
        print(row)

finally:
    connection.close() # Se cerro la conexion a la BD.
    print("La conexion ha finalizado.")
```

C. Link de GitHub

<https://github.com/dvd-r16/proyectos>

IV. IMÁGENES

A. PostgreSQL

Figura 2: Creación de Tabla

```
postgres=# CREATE TABLE T202010039 (
postgres(# Nombre VARCHAR(20),
postgres(# Carnet INT NOT NULL
postgres(# );
CREATE TABLE
```

Fuente: Elaboración Propia, 2024.

Figura 3: Observación de Tabla (Vacía)

```
postgres=# SELECT * FROM T202010039;
 nombre | carnet
-----+-----
(0 rows)

postgres=#
```

Fuente: Elaboración Propia, 2024.

Figura 4: Anexo de un usuario a la tabla

```
postgres=# INSERT INTO T202010039 (Nombre, Carnet) VALUES ('David Rodas', 202010039);
INSERT 0 1
```

Fuente: Elaboración Propia, 2024.

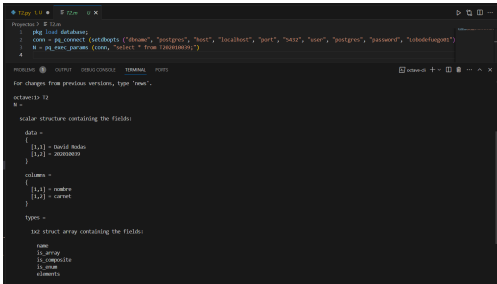
Figura 5: Observación de Tabla (Con Dato)

```
postgres=# SELECT * FROM T202010039;
 nombre | carnet 
-----+-----
 David Rodas | 202010039
(1 row)
```

Fuente: Elaboración Propia, 2024.

B. Octave

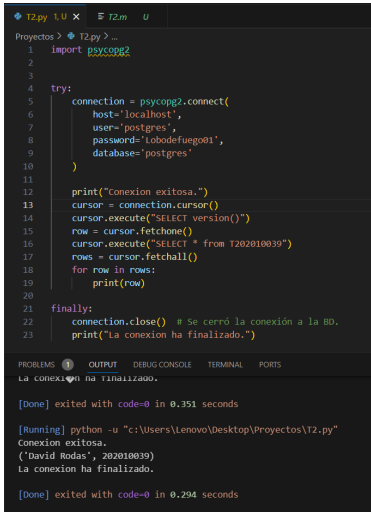
Figura 6: Simulación de programa

The image shows a screenshot of the Octave environment. The top part displays the code used to connect to a PostgreSQL database using the 'libpq' library. The code includes setting the host to 'localhost', user to 'postgres', password to 'Lobodefuego01', and database to 'postgres'. The bottom part shows the output of the code, indicating a successful connection and displaying the result of a SQL query: a scalar structure containing the fields 'data' and 'columns'.

Fuente: Elaboración Propia, 2024.

C. Python

Figura 7: Simulación de programa

The image shows a screenshot of a Python script being executed in a terminal. The script uses the 'psycopg2' library to connect to a PostgreSQL database. It sets the host to 'localhost', user to 'postgres', password to 'Lobodefuego01', and database to 'postgres'. The script then prints the connection status, executes a SQL query to select all data from the 'T202010039' table, and prints the results. The output shows a successful connection and the data from the table: ('David Rodas', 202010039). The script also shows the connection being closed and the final status message.

Fuente: Elaboración Propia, 2024.

V. CONCLUSIONES

1. Se logro demostrar la capacidad de establecer una conexión entre PostgreSQL y los lenguajes de programación, Python y Octave, con el uso de la bibliotecas/paquetes. Esto permite la ejecución de consultas SQL desde un entorno de programación, facilitando la interacción con la base de datos de manera programática. Esta integración es crucial para automatizar procesos de manejo de datos en aplicaciones de ingeniería eléctrica y otras disciplinas.
2. Dependiendo del lenguaje de programación, la cantidad de líneas de código será diferente. En este caso, Octave hizo uso de 5 líneas de programación, mientras que Python de 19 líneas de código (Ignorando las líneas vacías) para realizar una conexión exitosa con PostgreSQL.
3. Se logró entender los conceptos clave de las bases de datos, al igual el diseño y la estructuración de tablas, la importancia de las consultas SQL y el manejo de conexiones a bases de datos.
