Base de Datos

Elaborado por:
Gary Briceño

Contenido:

1. Persistencia de datos
2. Bases de Datos Relacionales vs NoSQL
3. Que es SQLite?
4. Que es un ORM:
    a. SQLAlquemy
    b. SQLModel
5. Uso de PyMongo
6. Boto3 y OpenSearch

Persistencia de Datos

- Es la opción de preservar la información de un objeto de forma permanente.
- También se refiere a la opción de poder recuperar la información del mismo y poder utilizarlo.
- En el caso de la persistencia de objetos, la información que persiste es el valor de los atributos.

Referencia:

- https://es.wikipedia.org/wiki/Persistencia_(inform%C3%A1tica)

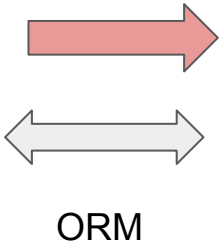
Objeto - Relacional

Python objects

```
class Person:
  first_name = "John"
  last_name = "Connor"
  phone_number = "+16105551234"
```

```
class Person:
  first_name = "Matt"
  last_name = "Makai"
  phone_number = "+12025555689"
```

```
class Person:
  first_name = "Sarah"
  last_name = "Smith"
  phone_number = "+19735554512"
```



ORM

Base de Datos **Relacional**: Oracle, MySQL, Postgree

ID	FIRST_NAME	LAST_NAME	PHONE
1	John	Connor	+16105551234
2	Matt	Makai	+12025555689
3	Sarah	Smith	+19735554512
...	...	...	...

- Uso de SQL para interactuar con la BD.

Tipos de Repositorios

- RDBMS: Relational Database Management System
  - Oracle
  - Postgre
  - MySQL
  - Aurora
- NoSQL
  - MongoDB
  - Cassandra
  - ElasticSearch
  - DynamoDB
  - Redis

NoSQL

- Almacena y devuelve datos en diferentes formas:
  - Almacenamiento de Llave - Valor
  - Base de datos de grafos
  - Almacenamiento de documentos
- Permite la extracción de datos sin consultar a otras tablas (joins)
- Permite almacenar la data y consultarla sin la creación de un modelo entidad-relación.
- Soporta una escalabilidad lineal: a más procesadores, más potencia.
- NoSQL: Not Only SQL

Tipo de NoSQL	Uso	Ejemplos
Llave-Valor	- Almacenamiento de imágenes - Cache de objetos - Sistemas diseñados para escalar	- Memcache - Redis - DynamoDB
BD de Columna de Familias: almacena los datos en columnas en lugar de filas.	- Resultado de un web crawler - Problemas de BigData	- Apache Cassandra - Apache HBase - Hypertable
BD de Grafos: formada por nodos y aristas, las relaciones entre nodos las conforman las aristas.	- Redes Sociales - Detección de fraude	- Neo4j - AllegroGraph - InfiniteGraph
BD de Documentos: almacena los datos en forma de documentos, que es un archivo JSON.	- Búsqueda de documentos - Data con alta variabilidad - Manejador de contenido	- MongoDB - OpenSearch

Bases Relacionales vs No Relacionales (NoSQL)

RDBMS pros:

- El manejo de transacciones ACID, a nivel de BD, hace el desarrollo más sencillo.

- La seguridad a nivel de columnas y filas, asegura que usuarios no autorizados realicen consultas o cambios.

- La mayoría de código SQL es portátil.

- La integridad referencial ayuda a la calidad de los datos.

- La mayoría esta relacionado con SQL.

RDBMS cons:

- El mapeo objeto-relacional puede ser complejo.

- El modelo entidad-relación debe completarse antes del inicio de las pruebas, hace lento el desarrollo.

- La RDBMS no escalan cuando se requieren joins.

- Se puede fragmentar en muchos servidores, pero requiere un código de aplicación.

- La búsqueda de textos requiere herramientas de terceros.

- Puede resultar difícil almacenar datos de alta variabilidad en tablas.

Bases Relacionales vs No Relacionales (NoSQL)

NoSQL pros:

- La carga de datos puede completarse antes de completar el modelado.

- La arquitectura modular permite el intercambio de componentes.

- El escalado lineal se lleva a cabo al agregar nuevos nodos al cluster.

- Las funciones de búsqueda integrada producen resultados de búsqueda de alta calidad.

- No hay necesidad de una capa de mapeo objeto-relacional.

- Es fácil almacenar datos de alta variabilidad.

NoSQL cons:

- Las transacciones ACID solo se pueden realizar a nivel de documento, otras transacciones deben ser realizadas a nivel de aplicación.

- Los almacenes de documento no proporcionan una seguridad detallada a nivel de elemento.

- Los sistemas NoSQL son nuevos para muchos.

- El almacén de datos tiene su propio lenguaje patentado, que no ayuda a la portabilidad.

- El almacén de datos no funcionará con las herramientas de reportes como OLAP.

Que es SQLite?

- Base de datos, que permite tener una base de datos embebida.
- Implementa el estándar SQL92, con lo cual permite consultas SQL.
- La base de datos se encuentra en un solo archivo.
- Puede funcionar en memoria.
- Cuenta con librerías de acceso para muchos lenguajes de programación.

Que es un ORM?

Definicion:

- Permite mapear estructuras de una base de datos relacional, sobre una estructura con objetos, a fin de acelerar el desarrollo de aplicaciones.
- Encapsula la complejidad de la BD y el uso de SQL para nosotros.

Examples:

- SQLAlchemy
- Pony
- Java: Hibernate, TopLink

Referencia:

- https://www2.deloitte.com/es/es/pages/technology/articles/que-es-orm.html

SQLAlchemy

- Librería de Python que facilita el acceso a una base de datos relacional.
- Independiente del motor de base de datos a utilizar.
- Se puede utilizar con consultas de SQL Nativo.
- La principal ventaja es que es un ORM: mapea tablas a clases Python.
- Implementa patrones de diseño y abstrae tareas como la creación del pool de conexiones.
    - Para el pool de conexiones, se utiliza el patrón Singleton.

Referencia:

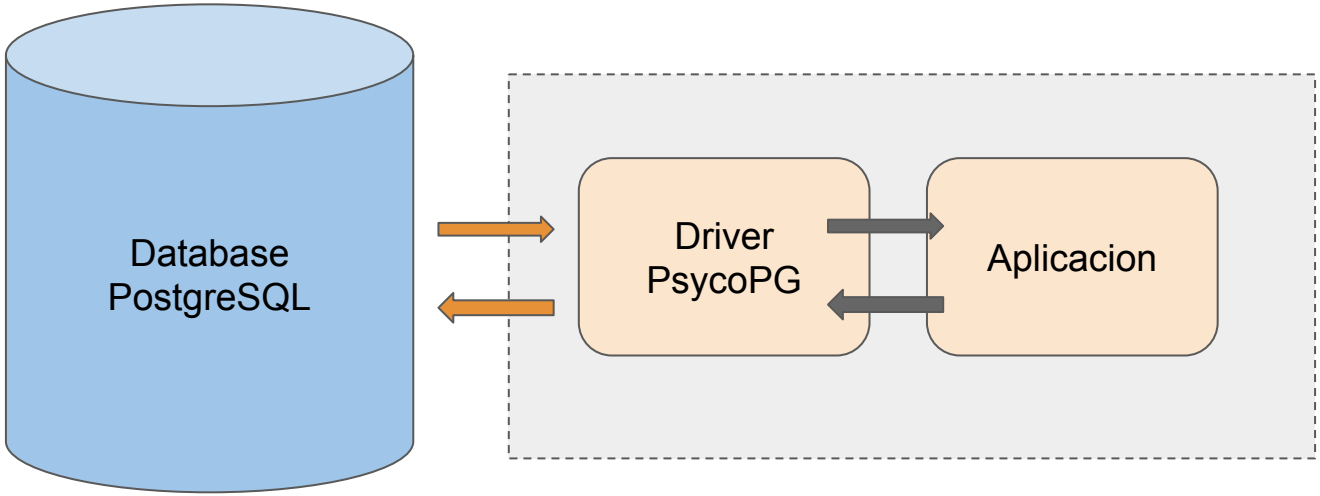
- https://j2logo.com/python/sqlalchemy-tutorial-de-python-sqlalchemy-guia-de-inicio/#sqlalchemy-quees
- https://www.clubdetecnologia.net/blog/2019/sqlalchemy-python-y-las-bases-de-datos/

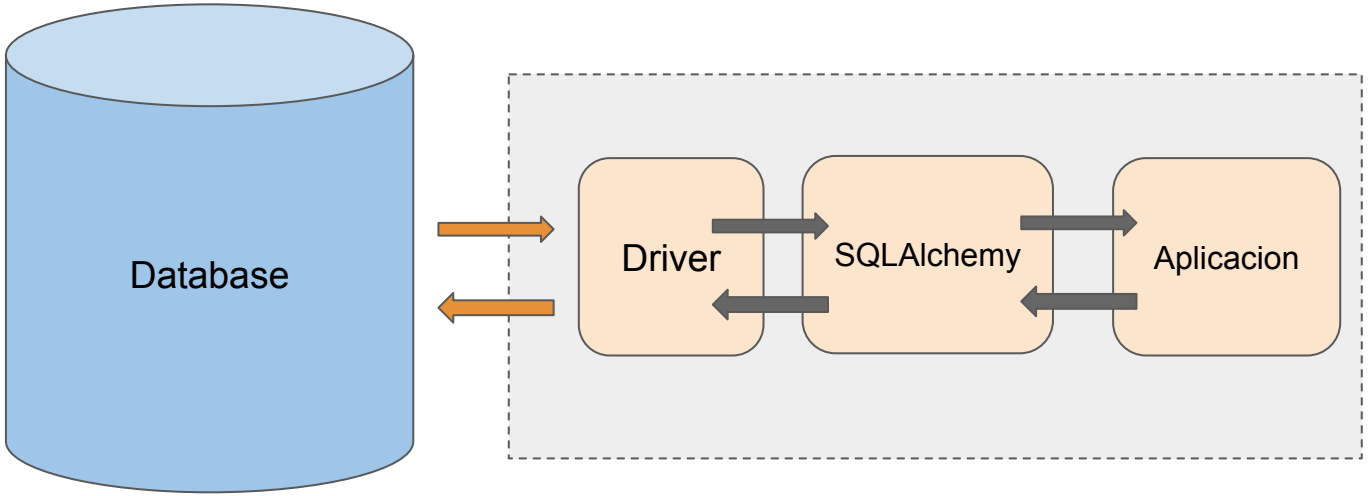
Psycopg

- Es un driver para PostgreSQL.
- Esta implementado en la especificación PythonDB API 2.0.

Referencia:

- https://www.psycopg.org/


Database
PostgreSQL
Driver
PsycoPG
Aplicacion


Database
Driver
SQLAlchemy
Aplicacion

SQLAlchemy y Psycopg2

- SQLAlchemy es el ORM.
- Psycopg2 es el driver a la base de datos.
- SQLAlchemy DEPENDE de Psycopg2 u otros drivers.

```
WARNING: Discarding https://files.pythonhosted.org/packages/19/79/35c7596bab4456f3610c12ec542a94d51c6781ced587d1d85127210b879b/psycopg2-2.0.10.tar.gz#sha256=e40cc04b43849085725076ae134bfef9e3b087f6dd7c964aeeb930e2f0bc14ab (from https://pypi.org/simple/psycopg2/). Command errored out with exit status 1: python setup.py egg_info Check the logs for full command output.
ERROR: Could not find a version that satisfies the requirement psycopg2 (from versions: 2.0.10, 2.0.11, 2.0.12, 2.0.13, 2.0.14, 2.2.0, 2.2.1, 2.2.2, 2.3.0, 2.3.1, 2.3.2, 2.4, 2.4.1, 2.4.2, 2.4.3, 2.4.4, 2.4.5, 2.4.6, 2.5, 2.5.1, 2.5.2, 2.5.3, 2.5.4, 2.5.5, 2.6, 2.6.1, 2.6.2, 2.7, 2.7.1, 2.7.2, 2.7.3, 2.7.3.1, 2.7.3.2, 2.7.4, 2.7.5, 2.7.6, 2.7.6.1, 2.7.7, 2.8, 2.8.1, 2.8.2, 2.8.3, 2.8.4, 2.8.5, 2.8.6, 2.9, 2.9.1, 2.9.2, 2.9.3)
ERROR: No matching distribution found for psycopg2
WARNING: You are using pip version 21.3.1; however, version 22.1.2 is available.
You should consider upgrading via the '/Users/gary/Developer/Python/fail-py/venv/bin/python -m pip install --upgrade pip' command.
```

```
### config.py ###

# Scheme: "postgres+psycopg2://<USERNAME>:<PASSWORD>@<IP_ADDRESS>:<PORT>/<DATAE

DATABASE_URI = 'postgres+psycopg2://postgres:password@localhost:5432/books'
```

```
Traceback (most recent call last):
  File "/Users/gary/Developer/Python/AWSWomen-Bootcamp2/ch02_DB/postgress_db/working_with_postgres.py", line 6, in <module>
    db = create_engine(db_string)
  File "<string>", line 2, in create_engine
  File "/Users/gary/Developer/Python/AWSWomen-Bootcamp2/venv/lib/python3.7/site-packages/sqlalchemy/util/deprecations.py", line 309, in warned
    return fn(*args, **kwargs)
  File "/Users/gary/Developer/Python/AWSWomen-Bootcamp2/venv/lib/python3.7/site-packages/sqlalchemy/engine/create.py", line 534, in create_engine
    entrypoint = u._get_entrypoint()
  File "/Users/gary/Developer/Python/AWSWomen-Bootcamp2/venv/lib/python3.7/site-packages/sqlalchemy/engine/url.py", line 661, in _get_entrypoint
    cls = registry.load(name)
  File "/Users/gary/Developer/Python/AWSWomen-Bootcamp2/venv/lib/python3.7/site-packages/sqlalchemy/util/langhelpers.py", line 344, in load
    "Can't load plugin: %s:%s" % (self.group, name)
sqlalchemy.exc.NoSuchModuleError: Can't load plugin: sqlalchemy.dialects:postgres
```

```
# ls
bin  boot  dev  docker-entrypoint-initdb.d  etc  home  lib  media  mnt  opt  proc  root  run  sbin  srv
# psql -U postgres
psql (14.3 (Debian 14.3-1.pgdg110+1))
Type "help" for help.

postgres=# \du
                                   List of roles
 Role name |                         Attributes                         | Member of
-----------+------------------------------------------------------------+-----------
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS | {}

postgres=# create database test
postgres-# \list
                                 List of databases
   Name    |  Owner   | Encoding |  Collate   |   Ctype    |   Access privileges
-----------+----------+----------+------------+------------+-----------------------
 postgres  | postgres | UTF8     | en_US.utf8 | en_US.utf8 |
 template0 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres          +
           |          |          |            |            | postgres=CTc/postgres
 template1 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres          +
           |          |          |            |            | postgres=CTc/postgres
(3 rows)

postgres-# create database test;
ERROR:  syntax error at or near "create"
LINE 2: create database test;
        ^
postgres=# CREATE DATABASE test;
CREATE DATABASE
postgres=# \list
                                 List of databases
   Name    |  Owner   | Encoding |  Collate   |   Ctype    |   Access privileges
-----------+----------+----------+------------+------------+-----------------------
 postgres  | postgres | UTF8     | en_US.utf8 | en_US.utf8 |
 template0 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres          +
           |          |          |            |            | postgres=CTc/postgres
 template1 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres          +
           |          |          |            |            | postgres=CTc/postgres
 test      | postgres | UTF8     | en_US.utf8 | en_US.utf8 |
(4 rows)

postgres=#
```

PyMongo

- Es la librería de Python para trabajar con MongoDB
- MongoDB es una base de datos orientada a documentos.

Referencia:

- https://pymongo.readthedocs.io/en/stable/tutorial.html
- https://www.mongodb.com/

Redis

- Es una base de datos NoSQL.
- Base de datos Llave-Valor.
- Se utiliza como memoria caché y message broker.

Referencia:

- https://en.wikipedia.org/wiki/Redis

Referencias: