Introduction to Postgres: Takeaways

Syntax

- Connecting to a database using psycopg2 :

```
import psycopg2
conn = psycopg2.connect("dbname=database_name user=username")
```

- Creating a table:

```
CREATE TABLE tableName(
   column1 dataType1 PRIMARY KEY,
   column2 dataType2,
   column3 dataType3,
   ...
);
```

- Inserting values using psycopg2:

```
import psycopg2
conn = psycopg2.connect("dbname=database_name user=username")
cur = conn.cursor()
cur.execute("INSERT INTO users VALUES (%s, %s, %s, %s);", (10, 'hello@dataquest.io', 'Some
Name', '123 Fake St.'))
conn.commit()
```

- Loading in a file using psycopg2:

```
conn = psycopg2.connect('dbname=database_name user=username')
cur = conn.cursor()
 # sample_file.csv has a header row.
with open('sample_file.csv', 'r') as f:
    # Skip the header row.
    next(f)
    reader = csv.reader(f)
    for row in reader:
        cur.execute("INSERT INTO users VALUES (%s, %s, %s, %s);", row)
```

- Returning the first result of a query:

```
cur.execute(query_string)
cur.fetchone()
```

- Returning all results of a query:

```
cur.execute(query_string)
cur.fetchall()
```

Workflow

- Connect to a database using the psycopg2.connect() function.

- Obtain a cursor object using the connection.cursor() method.
- Execute SQL queries using the cursor.execute() method.
- Commit your changes using the connection.commit() method.
- When you are done, close the connection using the connection.close() method.

Concepts

- Data engineers need to build a data pipeline that connects all the pieces of the data ecosystem together, and they need to keep it running.
- The parts of a data pipeline are the following:
    - Collecting
    - Short-term storage
    - Processing
    - Long-term storage
    - Presenting
- Relational databases are the most common storage used for web content, large business storage, and for data platforms.
- Postgres (or PostgreSQL) is one of the biggest open-source relational databases.
- Postgres is one of the best options for data analysts.
- Postgres is a more robust engine that is implemented as a server. Postgres can also handle multiple connections and can implement more advanced querying features.
- `psycopg2` is an open-source library that implements the Postgres protocol to connect to our Postgres server.
- SQL transactions prevent loss of data by ensuring all queries in a transaction block are executed at the same time. If any transactions fail, then the whole group fails, and no changes are made to the database.
- A new transaction will automatically be created when we open a connection in `psycopg2` .
- When a commit is called, the Postgres engine will run all the queries at once. Not calling a commit or rollback will cause the transaction to stay in a pending state, and the changes will not be made.
- Parametrized queries should use the `cursor.execute()` method and not Python string formatting.

Resources

- Comparison of Relational Databases
- Psycopg2 documentation
- PostgreSQL documentation
- Passing parameters to SQL queries