**TITLE:** COMPARISION BETWEEN TWO POSTGRESQL DATABASES USING PYTHON

**Purpose of the Article:**

The purpose of this article is to provide comparing schema structures, tables, views, column attributes ,functions, procedures, triggers, sequence and datatypes across two PostgreSQL databases using Python It provides easy-to-follow steps and example code to help developers and administrators ensure database consistency for migrations, version control, and deployments.

**Intended Audience:** Database Administrators, Developers ,Data Engineers, Technical Auditors

**Tools and Technology:** Python, Psycopg2, PostgreSQL **,**SQL

**Keywords**

* PostgreSQL schema comparison
* Database structure validation
* Python and psycopg2 database script
* Schema consistency verification
* Database administration

**Step-by-Step process**

1. **Import Necessary Libraries:**

Identify and import required Python libraries such as psycopg2, yaml, and pandas for database connectivity, YAML parsing, and data manipulation.

* import yaml
* import psycopg2
* import pandas as pd

**2. Create Config File (YAML):**

* Store details (e.g., host, dB name, user, password) in a YAML file.
* Read the YAML file and store the data in a variable as a dictionary.
* A screenshot of a computer

  Description automatically generated

A close-up of a computer screen

Description automatically generated

**3**. **Database Connectivity using psycopg2:**

The “connect\_to\_db” function connects to two databases using the given parameters and manages connection errors. If both connections succeed, it moves on to the comparison; otherwise, it shows an error message explaining the issue.

A screenshot of a computer

Description automatically generated

**4. Fetch common Schemas between two DB's and Compare tables and views:**

Retrieve schemas present in both databases and identify common ones for comparison. For each common schema, fetch the list of tables and views, compare their presence, and highlight differences.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer code

Description automatically generated

**5. Fetch common columns & compare data type and size between two databases:**

To compare schemas between two databases, first, extract column details such as names, data types, and sizes for all tables in each database. Next, identify common columns shared by both databases for each table. Then, validate the data types of these common columns to ensure consistency, and verify their size specifications, such as string length or numeric precision and scale, to confirm compatibility. This process helps identify any discrepancies in the schema structure between the databases.

A screenshot of a computer program

Description automatically generated

**6. Compare Constraints between two databases:**

Extract constraints (e.g., primary keys, foreign keys, unique constraints) for each common table and compare definitions across databases.

**7. Compare common table triggers between two databases:**

Identify triggers associated with common tables and compare their names, events, and definitions.

**A screenshot of a computer program

Description automatically generated**

**8. Compare Functions and Procedures between two databases**

Now, functions and procedures in two databases, identifying any missing routines or discrepancies in their definitions. It separates the functions from procedures, checks for differences, and outputs the comparison results for each schema. A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

9. **Compare Sequence between two databases:**

Fetch sequences in both databases, compare attributes (e.g., start, increment, max value), and identify discrepancies.

**A screenshot of a computer program

Description automatically generated**

10. **Compare data types between two databases**:

Conduct a high-level comparison of data types supported by both databases, focusing on naming conventions, sizes, and compatibility.

A screenshot of a computer

Description automatically generated

**11.Convert results to DataFrame and export to Excel:**

The results are compiled into a DataFrame and exported as an Excel file with a dynamically generated, timestamped filename in the format db\_comparison\_{dbname1}\_{dbname2}\_{timestamp}.xlsx. The file is saved in a designated directory with a path constructed dynamically. The process concludes by confirming the file's location through a printed message.A computer code with text

Description automatically generated with medium confidence

**Output in excel file:**

