Overview

Pyrseas provides utilities to maintain a PostgreSQL database schema. Its purpose is to enhance and follow through on the concepts of the Andromeda Project.

Whereas Andromeda expects the database designer or developer to provide a single YAML specification file of the database to be created, Pyrseas allows the development database to be created using the familiar SQL CREATE statements. The developer can then run the *dbtoyaml* utility to generate the YAML specification from the database. The spec can then be stored in any desired version control (VCS) repository. Similarly, she can add columns or modify tables or other objects using SQL ALTER statements and regenerate the YAML spec with dbtoyaml.

When ready to create or upgrade a test or production database, the *yamltodb* utility can be used with the YAML spec as input, to generate a script of SQL CREATE or ALTER statements to modify the database so that it matches the input spec.

A third tool, *dbaugment*, can be used to add custom attributes and supporting objects to a given schema. For example, an *updated* column can be added to various tables, together with trigger functions to ensure the columns are automatically modified as changes are made.

Use Cases

The following sections discuss the main scenarios where Pyrseas tools may be helpful.

Version Control

The case for implementing a tool to facilitate version control over SQL databases was made in a couple of blog posts: Version Control, Part 1: Pre-SQL and Version Control, Part 2: SQL Databases. In summary, SQL data definition commands are generally incompatible with traditional version control approaches which usually require comparisons (diffs) between revisions of source files.

A refinement of the approach described in the aforementioned blog posts will be of interest to users with many objects in their database schemas, i.e., many tables, views, functions, and other more complex objects. Instead of storing a complete database specification in a single YAML file, by using the *–multiple-files* option to **dbtoyaml**, the specification can be broken down into files

corresponding, generally, to a single database object. This allows a VCS **diff** facility to easily highlight database changes. Please refer to the dbtoyaml - Database to YAML and yamltodb - YAML to Database utilities for further details.

The Pyrseas version control tools are not designed to be the ultimate SQL database version control solution. Instead, they are aimed at assisting two or more developers or DBAs in sharing changes to the underlying database as they implement a database application. The sharing can occur through a distributed or centralized VCS. The Pyrseas tools may even be used by a single DBA in conjunction with a distributed VCS to quickly explore alternative designs. The tools can also help to share changes with a conventional QA team, but may require additional controls for final releases and production installations.

Supplementary Actions

In many instances, a database schema needs to be supplemented by rarely-modified data kept in certain tables, e.g., a codes-descriptions table. The data import and export features, controlled by *datacopy* configuration parameter (see Configuration Items for details) facilitates this need.

In other cases, DBAs may want to standardize certain additional table columns or processing. For example, they may want to capture the user and time of modification of a certain set of tables using a common procedure. The dbaugment - Augment a database utility was introduced to support these needs.

Generating SQL by determining what changed between one schema version and another is sometimes not sufficient. Although the change may be as simple as adding a column to a table and adding a referential constraint to the new column, if the tables already have data, it may not be possible to run the SQL generated by **yamltodb**. This requires manual intervention by the DBAs or developers. The project would like to assist with these types of changes. The blog post The Future of Pyrseas: Part 1 is a first step in discussing this requirement.

Naming

The project name comes from Python, the programming language, and Perseas [1], the Greek mythological hero who rescued Andromeda from a sea monster [2]. It is hoped that Pyrseas will rescue the Andromeda project <grin>. You can pronounce Pyrseas like the hero.

Footnotes

[1] The common English name for Perseas is Perseus and the Ancient Greek name is Perseos. However, in modern Greek Περσέας is the more common spelling for the mythical hero.

[2] He is better known for having killed Medusa.