Database Version Control

Release 0.5.1

Ken Ho

Jun 26, 2022

CONTENTS:

DVC

Version control your database!

DVC to version control your database!

CHAPTER

ONE

INTRODUCTION

1.1 What is Version Control?

In its most general sense, Version Control means to track and manage the different versions of a document or set of documents. While it is mostly practised in the world of software engineering, its use cases also abound in other areas. For instance, book writers often benefit from version controlling their drafts. This enables them to explore different writing styles, themes and so on. Failed attempt in a certain direction can be safely scrapped while the writer jumps back to the version he is most comfortable in.

1.2 What is Software Version Control?

As mentioned, software engineering practices version control heavily. Unlike hardware, software has high malleability. This makes it necessary to keep track of the changes made to the software and to have the ability to jump back to the prior version when the latest version breaks. The current industry standard for Software Version Control is Git, a distributed vrelease-0.1.15ersion control system created by Linux Torvald. It has superseded its predecessors like SVN, which is a local version control system which stores a project's history in a single server.

1.3 What is Database Version Control?

Interestingly, while there is much discussion around Software Version Control, there seems to be a lack of discussion around database version control. Database, especially Relational Database Management System (RDBMS), does not merely store data. It also stores, among others, data about data (metadata, like DDL) and user access privileges (DCL), Given that a lot of applications are data-driven, which means the software's behaviour is affected by the data which it receives, version controlling your database should not come off as secondary to version controlling your software.

Examples of Existing Database Version Control Systems in the market

1.4 Existing Database Version Control Tools

There are a number of existing database version control tools.

1.4.1 Commercial

Examples which are commericial and written in java.

- flyway
- liquibase

1.4.2 Open Source

In the python world, we have the below open-source products.

- alembic
- yoyo-migration

1.5 Personal experience

I have experimented with both alembic and yoyo-migration, but found the following shortcomings:

1.5.1 alembic

The metadata is mostly stored as files in the repository, and NOT as an entry in the database.

It makes it difficult to `directly query` the database for its current version.

1.5.2 yoyo-migration

The SQL migration files must be prefixed with numbers left padded with zeros.

For instance, '0001__description.sql'. If we have more than 1000 SQL files, then we probably need to rename the old files (e.g. to '00001__descrition.sql'). However, given that file names must be immutable in order for the database version control system to work properly, it means that it always has a upper limit as to how many SQL migration files you can have.

1.5.3 Goal of a custom database version control tool

This makes me want to create my own version of database version control system (DVC).

My own database version control system in python: Design Requirements

My goal is to have a DVC satisfying the below requirements:

- It works only with the Postgresql database (Optionally, it should be extendable)
- It accepts only SQL files (It does not use ORM, like SQLalchemy)
- It retains rich metadata in the database, so that we can check the database version via SQL.
- The spirit is that the DVC should act like *git*, which stores all the git objects in a *.git* folder. Similarly, there should be a table in the database, which stores all the metadata of the DVC, such that a simple SQL can be used to query the database version.

CHAPTER

TWO

FEATURES

This page explains the features of the CLI.

2.1 Overview

2.1.1 Command Line Interface

- Showing CLI commands.

2.1.2 Rich Database Metadata

Note: More information about the metadata can be found under design/metadata

- **Showing the metadata tables.**
  - dvc.database_revision_history table which shows the revision SQL files applied.
  - dvc.database_version_history table which shows the database version which results from the revision SQL files applied.

2.1.3 Upgrade or Downgrade

- **Showing upgrade and downgrade command.**
  - Started from database version 1 (Shown via *dvc db current*)
  - Showed the Revision SQL files under the default folder *sample_revision_sql_files*.
  - Applied database upgrade via *dvc db upgrade*.
  - Showed the database version became 2.
  - Applied database downgrade via *dvc db downgrade*.
  - Showed the database version was back to 1.

2.1.4 Flexible Configuration Format

Configuration is read either from i. Configuration File (config.yaml) or ii. Environment Variable

Configuration File

Note: The configuration file template can be generated via *dvc cfg init*

- **Showing dvc tool reads configuration from a configuration file.**
  - Ran a postgres DB via docker *docker run -e POSTGRES_USER=test -e POSTGRES_PASSWORD=test -e POSTGRES_DB=test -p 5433:5432 postgres:latest*
  - Copied a *config.yaml* file with configurations which match the spun up postgres DB.
  - Pinged the DB with *dvc db ping*. Success!
- The config.yaml file looks as follows:

```
credentials:
  dbflavour: postgres
  dbname: 'test'
  host: 'localhost'
  password: 'test'
  port: 5433
  user: 'test'
database_revision_sql_files_folder: sample_revision_sql_files
logging_level: DEBUG
```

Environment Variable

Note: The names of the environment variables can be found in the docker compose file

- **Showing dvc tool reads configuration from environment variables**
  - Ran a postgres DB via docker *docker run -e POSTGRES_USER=test -e POSTGRES_PASSWORD=test -e POSTGRES_DB=test -p 5433:5432 postgres:latest*
  - Loaded the env variables to the shell.
  - Pinged the DB with *dvc db ping*. Success!
- The environment variables look as follows:

```
>>> printnev| grep DVC
DVC__DATABASE_REVISION_SQL_FILES_FOLDER=sample_revision_sql_files
DVC__USER=test
DVC__PASSWORD=test
DVC__HOST=postgres_db
DVC__PORT=5432
DVC__DBNAME=test
DVC__DBFLAVOUR=postgres
DVC__LOGGING_LEVEL: DEBUG
```

2.2 Demo Files

- This page shows the SQL files used for the demonstraion.

2.2.1 RV1

Upgrade

```
-- create schema
CREATE SCHEMA IF NOT EXISTS fundamentals;
-- create table
CREATE TABLE IF NOT EXISTS fundamentals.source
  (
     source_id INTEGER PRIMARY KEY,
     source VARCHAR(50) NOT NULL UNIQUE,
     created_at TIMESTAMP NOT NULL DEFAULT (now() at time zone 'utc'),
     CONSTRAINT source_ux UNIQUE (source)
  );
-- Initialise with pre-set source
INSERT INTO fundamentals.source (source_id, source)
VALUES (1, 'yahoo'), (2, 'investing_dot_com') on conflict do nothing;
-- create table
CREATE TABLE IF NOT EXISTS fundamentals.price
  (
     ticker VARCHAR(10) NOT NULL,
     date DATE NOT NULL,
     open   NUMERIC(50,10) NOT NULL CHECK (open >=0),
     high   NUMERIC(50,10) NOT NULL CHECK (high >=0),
     low   NUMERIC(50,10) NOT NULL CHECK (low >=0),
     close   NUMERIC(50,10) NOT NULL CHECK (close >=0),
     adj_close   NUMERIC(50,10) NOT NULL CHECK (adj_close >=0),
     volume   NUMERIC(50,10) NOT NULL CHECK (volume >=0),
     source_id  INTEGER,
     created_at TIMESTAMP NOT NULL DEFAULT (now() at time zone 'utc'),
     updated_at TIMESTAMP NOT NULL DEFAULT (now() at time zone 'utc'),
     PRIMARY KEY (ticker, date),
     CONSTRAINT fk_source FOREIGN KEY(source_id) REFERENCES fundamentals.source(source_
↪id) ON DELETE CASCADE
  );
```

Downgrade

```
drop schema fundamentals cascade;
```

2.2.2 RV2

Upgrade

```
create schema if not exists datetime;
create table if not exists datetime.special_date(
  date date primary key,
  is_hk_public_holiday boolean default False,
  created_at TIMESTAMP NOT NULL DEFAULT (now() at time zone 'utc'),
  updated_at TIMESTAMP NOT NULL DEFAULT (now() at time zone 'utc')
);
```

Downgrade

```
drop schema datetime cascade;
```

2.2.3 RV3

Upgrade

```
create or replace view datetime.vw_trading_days_since_2021 as with weekdays (weekday) as (
select
        dt
-- Set timezone of now() to UTC
from
        generate_series('2021-01-01':: date, now() at time zone 'utc', '1 day':: interval) dt
where
        extract(dow
from
        dt) not in (6, 0)
        ),
        hk_holidays (hk_holiday) as (
select
        date
from
        datetime.special_date sd
where
        is_hk_public_holiday = true
        ),
trading_days (trading_day) as (
select
        weekday
from
        weekdays
except
select
        hk_holiday
```

(continues on next page)

(continued from previous page)

```
from
        hk_holidays)
select
        trading_day
from
        trading_days
order by
        trading_days ;
```

Downgrade

```
drop view if exists datetime.vw_trading_days_since_2021;
```

2.3 Step-by-step Guide

- This page explains how the CLI can be used

2.3.1 Check version

```
/home/ken $ dvc version
WARNING:root:Cannot find logging_level. Using default INFO
0.4.0
```

2.3.2 Initialise configuration File

```
/home/ken $ dvc cfg init
WARNING:root:Cannot find logging_level. Using default INFO
INFO:root:Now generating default config file config.yaml
INFO:root:Reading config from file...
INFO:root:Generating database revision folder
```

2.3.3 Create Test Database

```
 /home/ken $ psql -U postgres
psql (14.3)
Type "help" for help.

postgres=# create database my_test_db;
CREATE DATABASE
postgres=# exit
```

2.3.4 Create Test Database

```
  /home/ken $ vim config.yamlA

  1 credentials:¬
  2 ··dbflavour:·postgres¬
  3 ··dbname:·'my_test_db'¬
  4 ··host:·'localhost'¬
  5 ··password:·xxxxx
  6 ··port:·5432¬
  7 ··user:·'postgres'¬
  8 database_revision_sql_files_folder:·sample_revision_sql_files¬
  9 logging_level:·INFO¬
~
```

2.3.5 Populate sample_database_revision_files_folder

```
 /home/ken $ ls ./sample_revision_sql_files
RV1__create_scm_fundamentals_and_tbls.downgrade.sql
RV1__create_scm_fundamentals_and_tbls.upgrade.sql
RV2__create_scm_datetime_and_tbls.downgrade.sql
RV2__create_scm_datetime_and_tbls.upgrade.sql
RV3__datetime__create_vw_trading_days_since_2021.upgrade.sql
RV3__datetime__drop_vw_trading_days_since_2021.downgrade.sql
```

2.3.6 Ping the database

```
 /home/ken $ dvc db ping
INFO:root:Reading config from file...
Database connection looks good!
Database: my_test_db
Host: localhost
```

2.3.7 Initialise the database

```
 /home/ken $ dvc db init
INFO:root:Reading config from file...
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Database init successful!
Database: my_test_db
Host: localhost
```

2.3.8 Check Current Database Version

```
 /home/ken $ dvc db current
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Database Current Version: V0
```

2.3.9 Do Dry-run for db upgrade

- Check the logs to console and ensure the SQL file to be applied is really the correct one.

```
 /home/ken $ dvc db upgrade --dry-run
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Current Database Version is V0
Next Upgrade Revision Version will be 1
INFO:root:Reading config from file...
INFO:root:Reading config from file...
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Below files will be applied:
[File Path: sample_revision_sql_files/RV1__create_scm_fundamentals_and_tbls.upgrade.sql]
INFO:root:Dry run is complete
Aborted!
```

2.3.10 Run DB Upgrade

- Upgrade the DB and check the db version afterwards

```
 /home/ken $ dvc db upgrade
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Current Database Version is V0
Next Upgrade Revision Version will be 1
INFO:root:Reading config from file...
INFO:root:Reading config from file...
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Below files will be applied:
[File Path: sample_revision_sql_files/RV1__create_scm_fundamentals_and_tbls.upgrade.sql]
Going to apply file File Path: sample_revision_sql_files/RV1__create_scm_fundamentals_
↪and_tbls.upgrade.sql .....
You sure you want to continue ? [y/N]: y
INFO:root:Now applying sample_revision_sql_files/RV1__create_scm_fundamentals_and_tbls.
↪upgrade.sql and marking to metadata table
INFO:root:Reading config from file...
INFO:root:Reading config from file...

 /home/ken $ dvc db current
INFO:root:Reading config from file...
```

(continues on next page)

(continued from previous page)

```
INFO:root:Reading config from file...
Database Current Version: V1
```

2.3.11 Use –head

- Check the SQL file(s) to be applied with the –head flag

```
 /home/ken $ dvc db upgrade --head
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Current Database Version is V1
Next Upgrade Revision Version will be 2
INFO:root:Reading config from file...
INFO:root:Reading config from file...
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Below files will be applied:
[File Path: sample_revision_sql_files/RV2__create_scm_datetime_and_tbls.upgrade.sql,␣
↪File Path: sample_revision_sql_files/RV3__datetime__create_vw_trading_days_since_2021.
↪upgrade.sql]
Going to apply file File Path: sample_revision_sql_files/RV2__create_scm_datetime_and_
↪tbls.upgrade.sql .....
You sure you want to continue ? [y/N]: y
INFO:root:Now applying sample_revision_sql_files/RV2__create_scm_datetime_and_tbls.
↪upgrade.sql and marking to metadata table
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Going to apply file File Path: sample_revision_sql_files/RV3__datetime__create_vw_
↪trading_days_since_2021.upgrade.sql .....
You sure you want to continue ? [y/N]: y
INFO:root:Now applying sample_revision_sql_files/RV3__datetime__create_vw_trading_days_
↪since_2021.upgrade.sql and marking to metadata table
INFO:root:Reading config from file...
INFO:root:Reading config from file...
```

2.3.12 Use –head

- Check the result in the database

```
 /home/ken $ psql -U postgres -d my_test_db
psql (14.3)
Type "help" for help.

my_test_db=# select * from information_schema.tables where table_schema not in (
↪'information_schema', 'pg_catalog');
 table_catalog | table_schema |         table_name         | table_type | self_
↪referencing_column_name | reference_generation | user_defined_type_cata
log | user_defined_type_schema | user_defined_type_name | is_insertable_into | is_typed␣
↪| commit_action
```

(continues on next page)

(continued from previous page)

```
---------------+--------------+----------------------------+------------+----------------
↪--------------+----------------------+-----------------------
----+--------------------------+------------------------+--------------------+----------
↪+---------------
 my_test_db    | dvc          | database_revision_history  | BASE TABLE |               ␣
↪              |                      |
    |                          |                          | YES                | NO       |
 my_test_db    | dvc          | database_version_history   | BASE TABLE |               ␣
↪              |                      |
    |                          |                          | YES                | NO       |
 my_test_db    | fundamentals | source                     | BASE TABLE |               ␣
↪              |                      |
    |                          |                          | YES                | NO       |
 my_test_db    | fundamentals | price                      | BASE TABLE |               ␣
↪              |                      |
    |                          |                          | YES                | NO       |
 my_test_db    | datetime     | special_date               | BASE TABLE |               ␣
↪              |                      |
    |                          |                          | YES                | NO       |
 my_test_db    | datetime     | vw_trading_days_since_2021 | VIEW       |               ␣
↪              |                      |
    |                          |                          | NO                 | NO       |
(6 rows)
```

2.3.13 Downgrade the database with –base and –no-confirm

- Check the result in the database

```
 /home/ken $ dvc db downgrade --base --no-confirm
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Current Database Version is V3
Next Downgrade Revision Version will be 3
INFO:root:Reading config from file...
INFO:root:Reading config from file...
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Below files will be applied:
[File Path: sample_revision_sql_files/RV3__datetime__drop_vw_trading_days_since_2021.
↪downgrade.sql, File Path: sample_revision_sql_files/RV2__create_scm_datetime_and_tbls.
↪downgrade.sql, File Path: sample_revision_sql_files/RV1__create_scm_fundamentals_and_
↪tbls.downgrade.sql]
Going to apply file File Path: sample_revision_sql_files/RV3__datetime__drop_vw_trading_
↪days_since_2021.downgrade.sql .....
INFO:root:Now applying sample_revision_sql_files/RV3__datetime__drop_vw_trading_days_
↪since_2021.downgrade.sql and marking to metadata table
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Going to apply file File Path: sample_revision_sql_files/RV2__create_scm_datetime_and_
↪tbls.downgrade.sql .....
```

(continues on next page)

(continued from previous page)

```
INFO:root:Now applying sample_revision_sql_files/RV2__create_scm_datetime_and_tbls.
↪downgrade.sql and marking to metadata table
INFO:root:Reading config from file...
INFO:root:Reading config from file...
Going to apply file File Path: sample_revision_sql_files/RV1__create_scm_fundamentals_
↪and_tbls.downgrade.sql .....
INFO:root:Now applying sample_revision_sql_files/RV1__create_scm_fundamentals_and_tbls.
↪downgrade.sql and marking to metadata table
INFO:root:Reading config from file...
INFO:root:Reading config from file...
```

2.3.14 Check Database tables again

- Check the result in the database

```
 /home/ken $ psql -U postgres -d my_test_db
psql (14.3)
Type "help" for help.

my_test_db=# select * from information_schema.tables where table_schema not in (
↪'information_schema', 'pg_catalog');
 table_catalog | table_schema |        table_name         | table_type | self_
↪referencing_column_name | reference_generation | user_defined_type_catal
og | user_defined_type_schema | user_defined_type_name | is_insertable_into | is_typed |␣
↪commit_action
---------------+--------------+---------------------------+------------+-----------------
↪-------------+----------------------+-------------------------
---+--------------------------+-------------------------+--------------------+----------+-
↪--------------
 my_test_db    | dvc          | database_revision_history | BASE TABLE |                ␣
↪              |                      |
   |                          |                           | YES                | NO       |
 my_test_db    | dvc          | database_version_history  | BASE TABLE |                ␣
↪              |                      |
   |                          |                           | YES                | NO       |
(2 rows)


my_test_db=# \pset format wrapped
Output format is wrapped.
my_test_db=# select * from dvc.database_revision_history;
 revision_id | executed_sql_file_folder |    executed_sql_file_name     | executed_sql_
↪file_content_hash | executed_sql_file_content | operation | revision_applied |      ␣
↪created_at
-------------+--------------------------+-------------------------------+----------------
↪----------------+---------------------------+-----------+------------------+-----------
↪--------------
           1 | sample_revision_sql_file.| RV1__create_scm_fundamentals_.|␣
↪0f76966869d3d4eb2d3511f1dcd6f6.| -- create schema         +| Upgrade   | RV1          ␣
↪   | 2022-06-25 09:56:53.161.
             |.s                        |.and_tbls.upgrade.sql          |.1d           ␣
↪               | CREATE SCHEMA IF NOT EXIS.|           |                  |503
```

(continued on next page)

(continued from previous page)

```
        |                          |                              |              ␣
↪           |.TS fundamentals;         +|                  |            |
        |                          |                              |              ␣
↪           | -- create table          +|                  |            |
        |                          |                              |              ␣
↪           | CREATE TABLE IF NOT EXIST.|                  |            |
        |                          |                              |              ␣
↪           |.S fundamentals.source    +|                  |            |
        |                          |                              |              ␣
↪           |   (                      +|                  |            |
        |                          |                              |              ␣
↪           |      source_id INTEGER PR.|                  |            |
        |                          |                              |              ␣
↪           |.IMARY KEY,               +|                  |            |
        |                          |                              |              ␣
↪           |      source VARCHAR(50) N.|                  |            |
        |                          |                              |              ␣
↪           |.OT NULL UNIQUE,          +|                  |            |
        |                          |                              |              ␣
↪           |      created_at TIMESTAMP.|                  |            |
        |                          |                              |              ␣
↪           |. NOT NULL DEFAULT (now() .|                  |            |
        |                          |                              |              ␣
↪           |.at time zone 'utc'),     +|                  |            |
        |                          |                              |              ␣
↪           |      CONSTRAINT source_ux.|                  |            |
        |                          |                              |              ␣
↪           |. UNIQUE (source)         +|                  |            |
        |                          |                              |              ␣
↪           |   );                     +|                  |            |
        |                          |                              |              ␣
↪           | -- Initialise with pre-se.|                  |            |
        |                          |                              |              ␣
↪           |.t source                 +|                  |            |
        |                          |                              |              ␣
↪           | INSERT INTO fundamentals..|                  |            |
        |                          |                              |              ␣
↪           |.source (source_id, source.|                  |            |
        |                          |                              |              ␣
↪           |.)                        +|                  |            |
        |                          |                              |              ␣
↪           | VALUES (1, 'yahoo'), (2, .|                  |            |
        |                          |                              |              ␣
↪           |.'investing_dot_com') on c.|                  |            |
        |                          |                              |              ␣
↪           |.onflict do nothing;      +|                  |            |
        |                          |                              |              ␣
↪           | -- create table          +|                  |            |
        |                          |                              |              ␣
↪           | CREATE TABLE IF NOT EXIST.|                  |            |
        |                          |                              |              ␣
↪           |.S fundamentals.price     +|                  |            |
```

(continues on next page)

(continued from previous page)

```
             |                          |                               |               ␣
↪                 |   (                      +|           |                  |
             |                          |                               |               ␣
↪                 |      ticker VARCHAR(10) N.|           |                  |
             |                          |                               |               ␣
↪                 |.OT NULL,                  +|           |                  |
             |                          |                               |               ␣
↪                 |      date DATE NOT NULL, +|           |                  |
             |                          |                               |               ␣
↪                 |      open   NUMERIC(50,10.|           |                  |
             |                          |                               |               ␣
↪                 |.) NOT NULL CHECK (open >=.|           |                  |
             |                          |                               |               ␣
↪                 |.0),                      +|           |                  |
             |                          |                               |               ␣
↪                 |      high   NUMERIC(50,10.|           |                  |
             |                          |                               |               ␣
↪                 |.) NOT NULL CHECK (high >=.|           |                  |
             |                          |                               |               ␣
↪                 |.0),                      +|           |                  |
             |                          |                               |               ␣
↪                 |      low   NUMERIC(50,10).|           |                  |
             |                          |                               |               ␣
↪                 |. NOT NULL CHECK (low >=0).|           |                  |
             |                          |                               |               ␣
↪                 |.,                        +|           |                  |
             |                          |                               |               ␣
↪                 |      close   NUMERIC(50,1.|           |                  |
             |                          |                               |               ␣
↪                 |.0) NOT NULL CHECK (close .|           |                  |
             |                          |                               |               ␣
↪                 |.>=0),                    +|           |                  |
             |                          |                               |               ␣
↪                 |      adj_close   NUMERIC(.|           |                  |
             |                          |                               |               ␣
↪                 |.50,10) NOT NULL CHECK (ad.|           |                  |
             |                          |                               |               ␣
↪                 |.j_close >=0),            +|           |                  |
             |                          |                               |               ␣
↪                 |      volume   NUMERIC(50,.|           |                  |
             |                          |                               |               ␣
↪                 |.10) NOT NULL CHECK (volum.|           |                  |
             |                          |                               |               ␣
↪                 |.e >=0),                  +|           |                  |
             |                          |                               |               ␣
↪                 |      source_id  INTEGER, +|           |                  |
             |                          |                               |               ␣
↪                 |      created_at TIMESTAMP.|           |                  |
             |                          |                               |               ␣
↪                 |. NOT NULL DEFAULT (now() .|           |                  |
             |                          |                               |               ␣
↪                 |.at time zone 'utc'),     +|           |                  |
```

(continues on next page)

(continued from previous page)

```
             |                              |                                    |                 ␣
↪                |      updated_at TIMESTAMP.|           |                    |
             |                              |                                    |                 ␣
↪                |. NOT NULL DEFAULT (now() .|           |                    |
             |                              |                                    |                 ␣
↪                |.at time zone 'utc'),     +|           |                    |
             |                              |                                    |                 ␣
↪                |      PRIMARY KEY (ticker,.|           |                    |
             |                              |                                    |                 ␣
↪                |. date),                  +|           |                    |
             |                              |                                    |                 ␣
↪                |      CONSTRAINT fk_source.|           |                    |
             |                              |                                    |                 ␣
↪                |. FOREIGN KEY(source_id) R.|           |                    |
             |                              |                                    |                 ␣
↪                |.EFERENCES fundamentals.so.|           |                    |
             |                              |                                    |                 ␣
↪                |.urce(source_id) ON DELETE.|           |                    |
             |                              |                                    |                 ␣
↪                |. CASCADE                 +|           |                    |
             |                              |                                    |                 ␣
↪                |   );                     +|           |                    |
             |                              |                                    |                 ␣
↪                |                               |           |                    |
          2 | sample_revision_sql_file.| RV2__create_scm_datetime_and_.|␣
↪993c55319cdeec01775b868b94b7d6.| create schema if not exis.| Upgrade   | RV2          ␣
↪   | 2022-06-25 09:59:01.019.
             |.s                            |.tbls.upgrade.sql              |.2e              ␣
↪                |.ts datetime;             +|           |                    |.371
             |                              |                                    |                 ␣
↪                | create table if not exist.|           |                    |
             |                              |                                    |                 ␣
↪                |.s datetime.special_date( +|           |                    |
             |                              |                                    |                 ␣
↪                |   date date primary key, +|           |                    |
             |                              |                                    |                 ␣
↪                |   is_hk_public_holiday bo.|           |                    |
             |                              |                                    |                 ␣
↪                |.olean default False,     +|           |                    |
             |                              |                                    |                 ␣
↪                |   created_at TIMESTAMP NO.|           |                    |
             |                              |                                    |                 ␣
↪                |.T NULL DEFAULT (now() at .|           |                    |
             |                              |                                    |                 ␣
↪                |.time zone 'utc'),        +|           |                    |
             |                              |                                    |                 ␣
↪                |   updated_at TIMESTAMP NO.|           |                    |
             |                              |                                    |                 ␣
↪                |.T NULL DEFAULT (now() at .|           |                    |
             |                              |                                    |                 ␣
↪                |.time zone 'utc')         +|           |                    |
             |                              |                                    |                 ␣
↪                | );                            |           |                    |
```

(continues on next page)

(continued from previous page)

```
          3 | sample_revision_sql_file.| RV3__datetime__create_vw_trad.|␣
↪ffe591a58e951e7bbe87b0c06b7d8c.| create or replace view da.| Upgrade   | RV3          ␣
↪   | 2022-06-25 09:59:01.832.
            |.s                        |.ing_days_since_2021.upgrade.s.|.fa          ␣
↪               |.tetime.vw_trading_days_si.|           |                  |.872
            |                          |.ql                              |          ␣
↪               |.nce_2021 as with weekdays.|           |                  |
            |                          |                                 |          ␣
↪               |. (weekday) as (          +|           |                  |
            |                          |                                 |          ␣
↪               | select                   +|           |                  |
            |                          |                                 |          ␣
↪               |         dt               +|           |                  |
            |                          |                                 |          ␣
↪               | -- Set timezone of now() .|           |                  |
            |                          |                                 |          ␣
↪               |.to UTC                   +|           |                  |
            |                          |                                 |          ␣
↪               | from                     +|           |                  |
            |                          |                                 |          ␣
↪               |         generate_series('.|           |                  |
            |                          |                                 |          ␣
↪               |.2021-01-01':: date, now().|           |                  |
            |                          |                                 |          ␣
↪               |. at time zone 'utc', '1 d.|           |                  |
            |                          |                                 |          ␣
↪               |.ay':: interval) dt       +|           |                  |
            |                          |                                 |          ␣
↪               | where                    +|           |                  |
            |                          |                                 |          ␣
↪               |         extract(dow      +|           |                  |
            |                          |                                 |          ␣
↪               | from                     +|           |                  |
            |                          |                                 |          ␣
↪               |         dt) not in (6, 0)+|           |                  |
            |                          |                                 |          ␣
↪               |         ),               +|           |                  |
            |                          |                                 |          ␣
↪               |         hk_holidays (hk_h.|           |                  |
            |                          |                                 |          ␣
↪               |.oliday) as (             +|           |                  |
            |                          |                                 |          ␣
↪               | select                   +|           |                  |
            |                          |                                 |          ␣
↪               |         date             +|           |                  |
            |                          |                                 |          ␣
↪               | from                     +|           |                  |
            |                          |                                 |          ␣
↪               |         datetime.special_.|           |                  |
            |                          |                                 |          ␣
↪               |.date sd                  +|           |                  |
            |                          |                                 |          ␣
↪               | where                    +|           |                  |
```

(continues on next page)

(continued from previous page)

```
             |                              |                                  |                 ␣
↪                |         is_hk_public_holi.|           |                      |
             |                              |                                  |                 ␣
↪                |.day = true                   +|           |                      |
             |                              |                                  |                 ␣
↪                |         ),                   +|           |                      |
             |                              |                                  |                 ␣
↪                | trading_days (trading_day.|           |                      |
             |                              |                                  |                 ␣
↪                |.) as (                       +|           |                      |
             |                              |                                  |                 ␣
↪                | select                       +|           |                      |
             |                              |                                  |                 ␣
↪                |         weekday              +|           |                      |
             |                              |                                  |                 ␣
↪                | from                         +|           |                      |
             |                              |                                  |                 ␣
↪                |         weekdays             +|           |                      |
             |                              |                                  |                 ␣
↪                | except                       +|           |                      |
             |                              |                                  |                 ␣
↪                | select                       +|           |                      |
             |                              |                                  |                 ␣
↪                |         hk_holiday           +|           |                      |
             |                              |                                  |                 ␣
↪                | from                         +|           |                      |
             |                              |                                  |                 ␣
↪                |         hk_holidays)         +|           |                      |
             |                              |                                  |                 ␣
↪                | select                       +|           |                      |
             |                              |                                  |                 ␣
↪                |         trading_day          +|           |                      |
             |                              |                                  |                 ␣
↪                | from                         +|           |                      |
             |                              |                                  |                 ␣
↪                |         trading_days         +|           |                      |
             |                              |                                  |                 ␣
↪                | order by                     +|           |                      |
             |                              |                                  |                 ␣
↪                |         trading_days ;       |           |                      |
          4 | sample_revision_sql_file.| RV3__datetime__drop_vw_tradin.|␣
↪b4fd8e1c50f3f17eab0ad8f05d2d50.| drop view if exists datet.| Downgrade | RV3           ␣
↪    | 2022-06-25 10:09:41.442.
             |.s                            |.g_days_since_2021.downgrade.s.|.b1             ␣
↪                |.ime.vw_trading_days_since.|           |                      |.939
             |                              |.ql                               |                 ␣
↪                |._2021;                       |           |                      |
          5 | sample_revision_sql_file.| RV2__create_scm_datetime_and_.|␣
↪d040183e1231b5e4e82f0c7e4133d0.| drop schema datetime casc.| Downgrade | RV2           ␣
↪    | 2022-06-25 10:09:41.471.
             |.s                            |.tbls.downgrade.sql               |.c6             ␣
↪                |.ade;                         |           |                      |.216
```

(continues on next page)

(continued from previous page)

```
          6 | sample_revision_sql_file.| RV1__create_scm_fundamentals_.|␣
↪f28acbf63e26ece3c0324a9cd74dd2.| drop schema fundamentals .| Downgrade | RV1          ␣
↪   | 2022-06-25 10:09:41.493.
            |.s                       |.and_tbls.downgrade.sql        |.87           ␣
↪               |.cascade;                  |           |                 |.714
```

CHAPTER

THREE

QUICKSTART

3.1 Executable

Note: Executables are available on Github Releases

- The releases page provides executables on the below three Operating Systems.
  - Mac (Latest)
  - Linux Ubunutu (Latest)
  - Windows (Latest)

if you happen to use one of the OSes listed above, you can download the executable directly and use it without installing python!

3.2 PyPI Library

Note: The commandline tool is uploaded to PyPI

Run the below to see it in action.

```
# Install the library from PyPi
pip install database-version-control

# To get more instructions of the commandline tool, run the below in the terminal
dvc --help
```

See detailed deployment of the commandline tool

3.3 Docker Image

Note: The tool is containerised and is distributed on Dockerhub

Check out the *docker-compose.yml* file in the github repository to run a demo!

Run the below to see it in action.

```
# Clone the repo and checkout release branch
git clone -b release git@github.com:kenho811/Python_Database_Version_Control.git

# cd to the repository
cd Python_Database_Version_Control/docker_compose_demo

# Fnd the docker-compose.yml and run
docker compose up

# Using psql as client, access the postgres DB and see the result
(URL: postgres://test:test@localhost:5433/test)
PGPASSWORD=test psql -U test -d test -h localhost -p 5433

# Check out docker-compose.yml file for usage as a microservice
```

See detailed deployment of the Docker Image

CHAPTER

FOUR

DEPLOYMENT

This page explains how the tool can be deployed.

4.1 Docker Image

This page explains how to use the tool packaged in a Docker Image.

4.1.1 Docker compose

- *Docker Compose* allows spinning up several containers together as a single service.
- Configurations are stored in a *docker-compose.yaml* file.
- For illustration, please refer to the docker-compose.yml file.

4.1.2 Kubernetes

- TBC

4.2 CLI

This page explains how to use the tool packaged as commandline tool (CLI).

It is easier to use a configuration file than environment variables.

A typical workflow is as follows:

```
# Generate configuration file
dvc cfg init

# Update the generated config.yaml

# Test connection to the database with the updated config.yaml
dvc db ping

# Create a folder to store all the SQL files to be applied to the database
# Remember to use the same folder name as specified in the config.yaml file
```

(continues on next page)

(continued from previous page)

```
# Make sure the SQL files follow the naming conventions.

# Run upgrade script
dvc db upgrade

# (Optional) To revert the migration, create a downgrade script and run it
dvc db downgrade

# (Optional) Check the current database version
dvc db current
```

CHAPTER

FIVE

DESIGN

This page explains the design of the tool

5.1 Overview

This page explains gives a graphical overview of the tool.

5.1.1 DVC CLI commands and subcommands

This section explains the client-facing side of the tool. The library is exposed via the commandline *dvc*.

5.1.2 Data Structures

Core Structure

This section explains the core of the tool. It shows the interaction between the classes (marked in *yellow*). Greyed out items are features yet to be implemented.

DatabaseVersion and DatabaseRevisionFile

- Below is a series of graphs illustrating the relationship between
  - Current Database Version (DatabaseVersion)
  - Target Database Version (DatabaseVersion)
  - One or more Database Revision Files (DatabaseRevisionFiles)

5.2 Configuration

- **Two ways to pass configurations to DVC tool.**
  - Via Environment Variables.
  - Via Configuration FIle.
- Note that `Configuration File` has precedence OVER `Environment Variables`. In other words, if a configuration file is detected by DVC tool, the environment variables will be ignored.

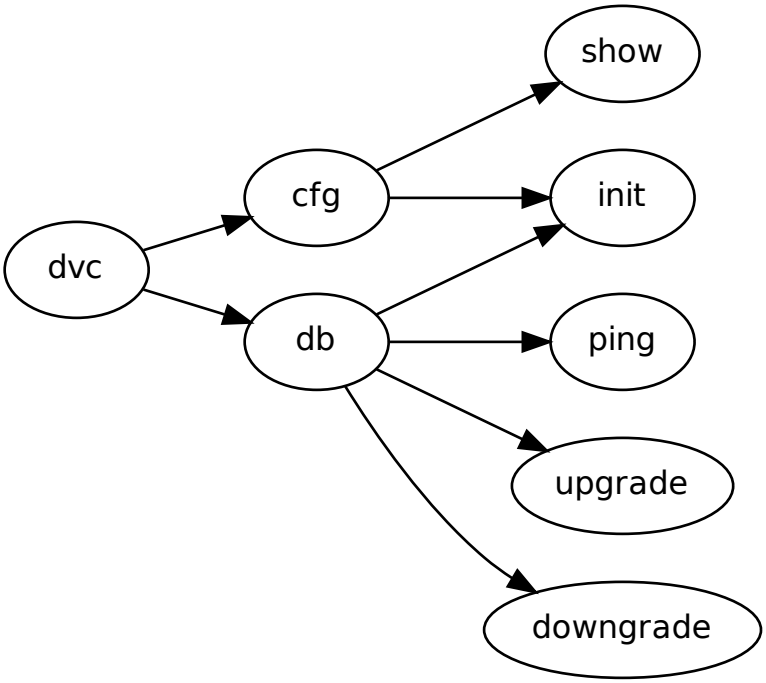


Fig. 1: DVC CLI Commands and Subcommands

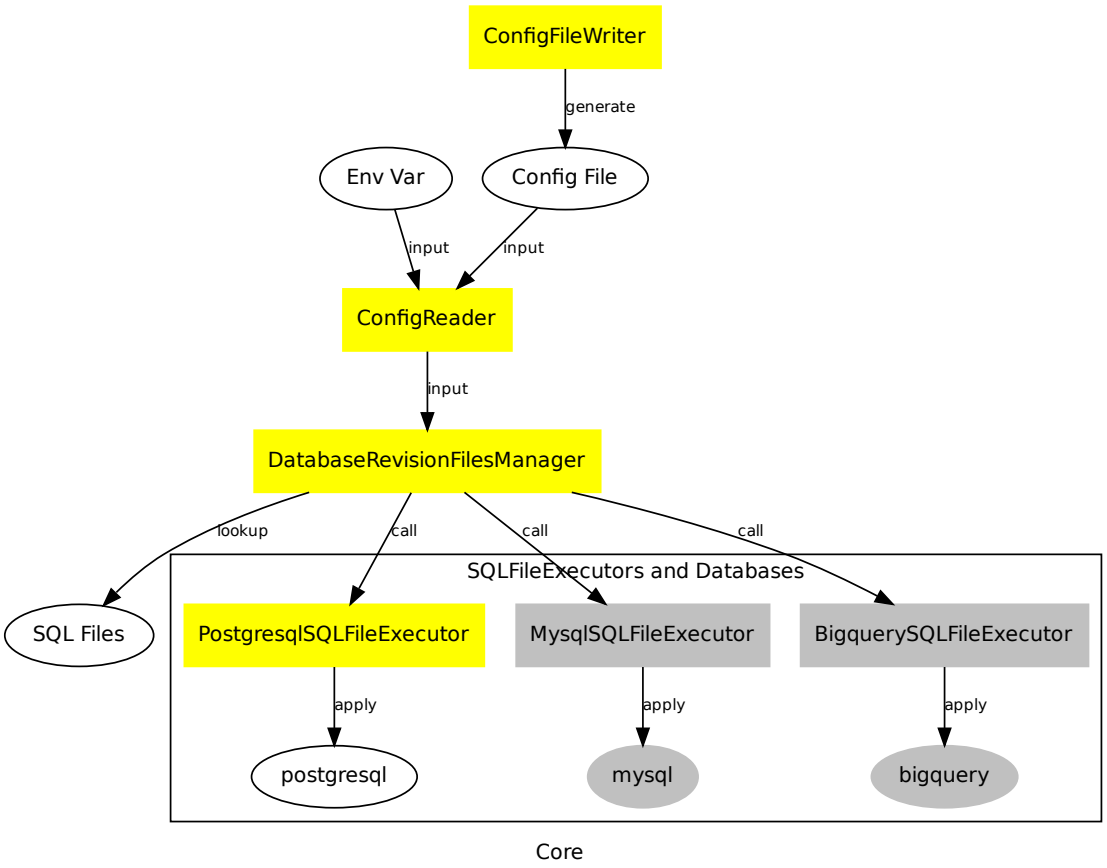


Fig. 2: DVC Core structure

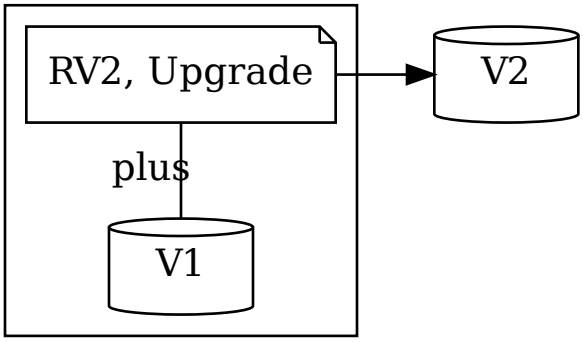


Fig. 3: V1 + RV2 (Upgrade) = V2

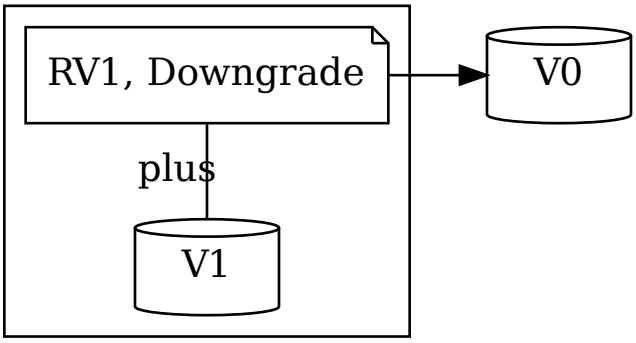


Fig. 4: V1 + RV1 (Downgrade) = V0

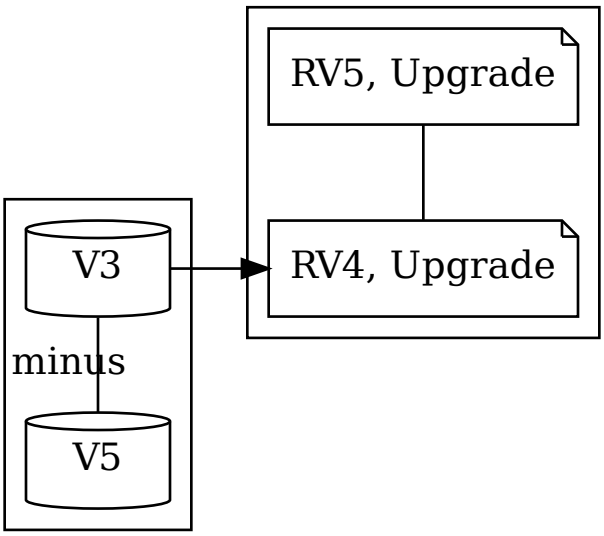


Fig. 5: V5 - V3 = [RV4 Upgrade + RV5 Upgrade]

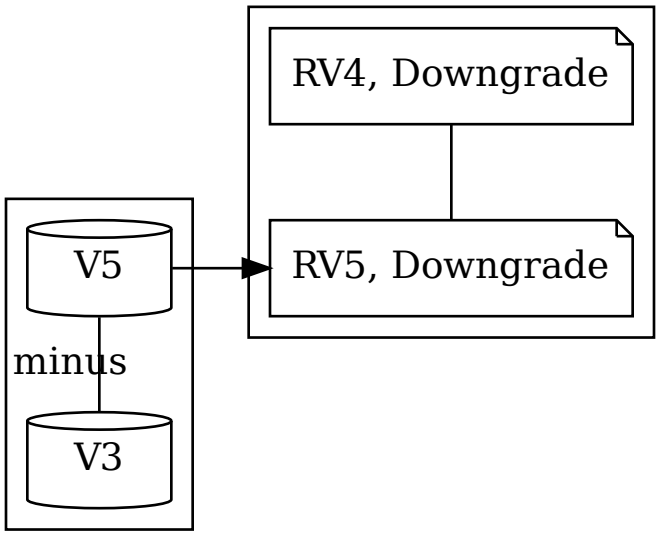


Fig. 6: V3 - V5 = [RV5 Downgrade + RV4 Downgrade]

5.3 Revision SQL files Namaing Convention

The tool discovers and applies SQL files to the database for version control.

These files need to follow a certain naming conventions.

- All SQL files are considered *revision files*
- They must follow the pattern *RV[0-9]*__.*.(upgrade|downgrade).sql*. In words, it means
  - They start with the prefix *RV*
  - After *RV*, it follows an arbitrary revision number (e.g. RV1, RV2, RV3 etc. etc.)
  - After *RV(arbitrary_revision_number)*, it follows double underscores and an arbitrary number of characters. Everything after __ describes what the SQL file does.
  - After *RV(arbitrary_revision_number)__(description)*, it follows a dot and the character group of either *upgrade* or *downgrade*. When applied, an upgrade revision file will move the database version upward by 1, while a downgrade revision file will move the database version downward by 1.
  - After *RV(arbitrary_revision_number)__(description).(upgrade/downgrade)*, it follows a dot and the character group of *sql* .
  - Overall, *RV(arbitrary_revision_number)__(description).(upgrade/downgrade).sql*
- Example SQL revision files
  - RV1__create_scm_company_secrets_and_tbl_earnings.upgrade.sql
  - RV1__delete_scm_company_secrets_cascade.downgrade.sql
  - RV2__alter_scm_company_secrets_tbl_earnings_updated_at_add_index.upgrade.sql
  - RV2__alter_scm_company_secrets_tbl_earnings_updated_at_remove_index.downgrade.sql

5.4 Database Metadata

Just like git which stores all the metadata in a dot git folder (.git), the tool also stores metadata in the database where SQL Revision Files are applied.

- Schema dvc will be created
  - **Table dvc.database_revision_history will be created.**
    - History of revision SQL files applied.
  - **Table dvc.database_version_history will be created.**
    - History of database versions which result from revision SQL files applied.

CHAPTER

SIX

CONTRIBUTION

This page explains how to contribute to the codebase.

6.1 Development and Release Workflow

This page explains the development/release workflow.

It follows the model of Release Branching Strategy

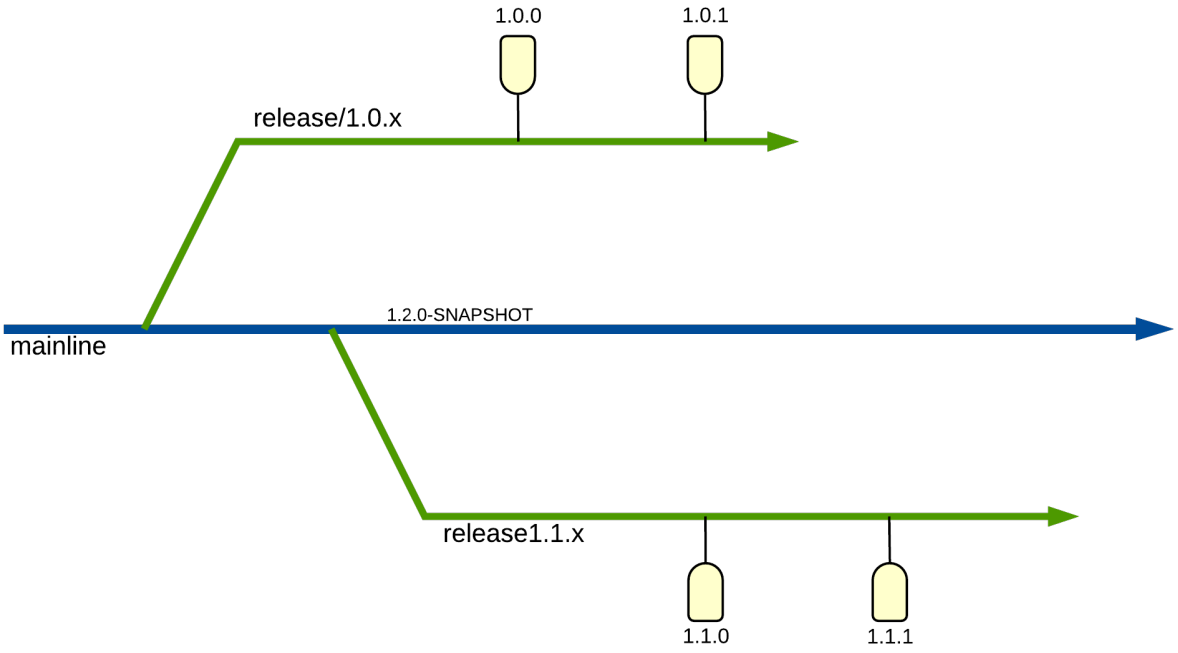


Fig. 1: Sourced from http://releaseflow.org/

6.1.1 Development - Add new features

In General,

- Create a feature branch from master. Call it *feature/{theme}*
- Keep making changes to the feature branch until the feature is ready. Make sure it passes all automated tests.
- Merge it to master branch after it's complete.

```
# Git clone the repo and checkout master
git clone -b master git@github.com:kenho811/Python_Database_Version_Control.git

# create a feature branch from the master branch
git checkout -b feature/{theme}

# Pip install dependencies
pip install with `pip install ".[dev]"`

# Development

# Write unit + integration tests

# Run pytest
pytest

# Generate Documentation locally. ISLOCAL=1 removes local dependencies.
cd docs
ISLOCAL=1 make clean html

# Open PR against master
```

6.1.2 Release - Make App available for deployment

In General,

- Do a code-cut from the master branch.
- Name the new branch *release/{major.minor}*
- Create a tag based on the new branch *release{major.minor.patch}*

```
# Review and merge PR into master branch

# Update local master branch
git checkout master
git pull

# Update local master branch

# Update the tool's version under src/dvc/version.py
# See: https://github.com/kenho811/Python_Database_Version_Control/blob/master/src/dvc/
↪version.py#L1
# Example: __version__ = "{major.minor.patch}"

```

(continues on next page)

(continued from previous page)

```
# Cut a release branch with the same updated version number
git checkout -b release/{major.minor}
git push --set-upstream origin release/{new_version_num}

# Create a new release tag
git tag release/{major.minor.patch}
git push --tags
```

6.1.3 Hotfix - Fix deployment bugs

In General,

- Say the release with release tag *release/0.5.0* has bugs.
- Create a new hotfix branch from the branch *release/0.5*
- After the fix is in place, merge it against *release/0.5*
- Create a new release tag based on the latest commit at *release/0.5*. Call it *release/0.5.1*.

```
# Checkout the release branch which the problematic release tag belongs to
git checkout release/0.5

# Create hotfix branch
git branch -b hotfix/against_0.5.0

# Once done, open a PR and merge back to release/0.5

# At release/0.5 branch, update the __version__
# See: https://github.com/kenho811/Python_Database_Version_Control/blob/master/src/dvc/
↪version.py#L1
# Example: __version__ = "0.5.1"

# Create a new release tag
git tag release/0.5.1
git push --tags
```

- The CI pipeline specified here will manage the rest.

6.2 Continuous Integration

The Codebase is on github.

CI is currently done via github action. It is integrated to the following destinations:

- Dockerhub
- PyPI
- ReadtheDocs (Latest)

The table below shows the details:

Table 1: CI convention for branches

Branch	Performs Tests?	Artifacts	DockerHub Tag	Push DockerHub Readme?	PyPI Version	Readthedocs Version
master	Yes	Pytest report.	latest	Yes	N/A	latest
feature/**	Yes	N/A	N/A	No	N/A	N/A
release/{major.minor}	Yes	N/A	N/A	No	N/A	N/A

Table 2: CI convention for tags

Tag	Performs Tests?	Artifacts	DockerHub Tag	Push DockerHub Readme?	PyPI Version	Readthedocs Version
release/{major.minor.patch}	Yes	Pytest artifacts + executables in github releases.	release-{major.minor.patch}	No	{major.minor.patch}	release-{major.minor.patch}

CHAPTER

SEVEN

SOURCE DOCUMENTATION

7.1 Src Package

7.1.1 dvc package

Subpackages

dvc.app package

Subpackages

dvc.app.cli package

Subpackages

dvc.app.cli.commands package

Subpackages

dvc.app.cli.commands.database package

Submodules

dvc.app.cli.commands.database.backend module

class dvc.app.cli.commands.database.backend.**DatabaseInteractor**(*config_file_path_str: str*)

Bases: object

Exposes API to interact with Various Database flavours

MAPPING = {SupportedDatabaseFlavour.Postgres: <function DatabaseInteractor.<lambda>>}

property conn

Returns

property database_revision_files_manager: *dvc.core.config.DatabaseRevisionFilesManager*

Returns DatabaseRevisionFilesManager

execute_single_sql_file(*database_revision_file:* dvc.core.struct.DatabaseRevisionFile, *mark_only: bool = False*) → None
Execute DatabaseRevisionFile to the Database and optionally mark it as metadata

Parameters

- **database_revision_file** – Database Revision File to apply to the Database
- **mark_only** – whether or not to mark the SQL file as being done as metadata, without actually executing the SQL file

Returns

get_target_database_revision_files(*steps: int*, *pointer: Optional[*dvc.core.config.DatabaseRevisionFilesManager.Pointer*] = None*) → List[*dvc.core.struct.DatabaseRevisionFile*]
Helper to get target database revision files Check number of returned revision files must be same as steps specified

Parameters **steps** – Specify how many steps ahead/ backwards.. When None, it goes to the very end in either direction

Returns List of DatabaseRevisionFiles, if any

property latest_database_version: *dvc.core.struct.DatabaseVersion*

Returns latest Database Version

ping() → None
Ping the database connection

Returns

property sql_file_executor

Module contents

database subcommand

Submodules

dvc.app.cli.commands.config module

config subcommand

Module contents

Submodules

dvc.app.cli.main module

Starting point of the CLI

Module contents

Module contents

dvc.core package

Subpackages

dvc.core.database package

Subpackages

dvc.core.database.bigquery package

Module contents

class dvc.core.database.bigquery.**BigQuerySQLFileExecutor**(*db_conn: psycopg2.extensions.connection*)

Bases: *dvc.core.database.SQLFileExecutorTemplate*

dvc.core.database.mysql package

Module contents

class dvc.core.database.mysql.**MySQLSQLFileExecutor**(*db_conn: psycopg2.extensions.connection*)

Bases: *dvc.core.database.SQLFileExecutorTemplate*

dvc.core.database.postgres package

Module contents

class dvc.core.database.postgres.**PostgresSQLFileExecutor**(*db_conn: psycopg2.extensions.connection*)

Bases: *dvc.core.database.SQLFileExecutorTemplate*

FILE_HASHER = <dvc.core.hash.FileHasher object>

METADATA_SQL_FOLDER_PATH = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/python-database-version-control/checkouts/feature-doc/src/dvc/core/database/postgres')

execute_database_revision(*database_revision_file:* dvc.core.struct.DatabaseRevisionFile)

Execute database revision and write to database version control tables :param database_revision_file: :return:

get_latest_database_version() → *dvc.core.struct.DatabaseVersion*

Get the latest database version :return:

set_up_database_revision_control_tables()

Create all database revision control schema and tables :return:

Module contents

class dvc.core.database.**SQLFileExecutorTemplate**(*db_conn: psycopg2.extensions.connection*)
Bases: abc.ABC

Abstract Base Class for all SQLFileExecutors for different datbaases

abstract execute_database_revision(*database_revision:* dvc.core.struct.DatabaseRevisionFile)

abstract get_latest_database_version()

abstract set_up_database_revision_control_tables()

class dvc.core.database.**SupportedDatabaseFlavour**(*value*)
Bases: enum.Enum

List of database flavours supported in the programme

Postgres = 'postgres'

Submodules

dvc.core.config module

class dvc.core.config.**ConfigDefault**
Bases: object

KEY__DATABASE_REVISION_SQL_FILES_FOLDER = 'DVC__DATABASE_REVISION_SQL_FILES_FOLDER'

KEY__DBFLAVOUR = 'DVC__DBFLAVOUR'

KEY__DBNAME = 'DVC__DBNAME'

KEY__HOST = 'DVC__HOST'

KEY__LOGGING_LEVEL = 'DVC__LOGGING_LEVEL'

KEY__PASSWORD = 'DVC__PASSWORD'

KEY__PORT = 'DVC__PORT'

KEY__USER = 'DVC__USER'

VAL__DATABASE_REVISION_SQL_FILES_FOLDER = 'sample_revision_sql_files'

VAL__DBFLAVOUR = 'postgres'

VAL__DBNAME = ''

VAL__FILE_PATH: pathlib.Path = PosixPath('config.yaml')

VAL__FilE_NAME: str = 'config.yaml'

VAL__HOST = ''

VAL__LOGGING_LEVEL: str = 'INFO'

VAL__PASSWORD = ''

VAL__PORT = 5432

VAL__USER = ''

classmethod get_config_dict(*database_revision_sql_files_folder: str*, *user: str*, *password: str*, *host: str*, *port: int*, *dbname: str*, *dbflavour: str*, *logging_level: int*, *as_file=False*)

Parameters

- **database_revision_sql_files_folder** –
- **user** –
- **password** –
- **host** –
- **port** –
- **dbname** –
- **dbflavour** –
- **logging_level** – Assumed to be integer value
- **as_file** – whether to dump the dict as file.

Returns

class dvc.core.config.**ConfigFileWriter**(*config_file_path: Union[pathlib.Path, str] = PosixPath('config.yaml')*)

Bases: object

Read Config Files (in different formats) to Python Dictionary

write_to_yaml() → None

class dvc.core.config.**ConfigReader**(*config_file_path: Union[pathlib.Path, str] = PosixPath('config.yaml')*)

Bases: object

Read Config (in different formats) to Python Dictionary

Precedence in descending order 1. Config File 2. Environment Variable

class dvc.core.config.**DatabaseConnectionFactory**(*config_reader:* dvc.core.config.ConfigReader)

Bases: object

Return connections for various databases

MAPPING = {SupportedDatabaseFlavour.Postgres: 'self.pgconn'}

property conn: psycopg2.extensions.connection

Return the expected connection object for different database flavours :return:

property pgconn: psycopg2.extensions.connection

Return Postgres Database Connection

Returns

validate_requested_database_flavour() → *dvc.core.database.SupportedDatabaseFlavour*

Validate if requested database flavour is supported

Returns

class dvc.core.config.**DatabaseRevisionFilesManager**(*config_file_reader:* dvc.core.config.ConfigReader)

Bases: object

Manager all Database Revision Files

class Pointer
Bases: object

Head: ALl the way to the latest

BASE = 'base'

HEAD = 'head'

create_database_revision_files_folder() → None
Safely create the database revision files folder.

get_target_database_revision_files_by_pointer(*current_database_version:* dvc.core.struct.DatabaseVersion, *candidate_database_revision_files:* *List[*dvc.core.struct.DatabaseRevisionFile*]*, *pointer:* dvc.core.config.DatabaseRevisionFilesManager.Pointer) → List[*dvc.core.struct.DatabaseRevisionFile*]
Given current database version and pointer, filter for target database revision files in the folder

Parameters

- **current_database_version** –
- **candidate_database_revision_files** –

Returns

get_target_database_revision_files_by_steps(*current_database_version:* dvc.core.struct.DatabaseVersion, *steps: int*, *candidate_database_revision_files:* *List[*dvc.core.struct.DatabaseRevisionFile*]*) → List[*dvc.core.struct.DatabaseRevisionFile*]
Given current database version and number of steps, filter for target database revision files in the folder.

Returns

dvc.core.exception module

exception dvc.core.exception.DatabaseConnectionFailureException
Bases: Exception

Exception raised when connection to the database fails

exception dvc.core.exception.EnvironmentVariableNotSetException(*missing_env_var: str*)
Bases: Exception

Exception raised when required environment variables are not found

exception dvc.core.exception.InvalidDatabaseRevisionFilesException(*status:* dvc.core.exception.InvalidDatabaseRevisionFile *config_file_path:* *Optional[pathlib.Path]*, *database_revision_file_paths:* *List[pathlib.Path]*)

Bases: Exception

Exception Raised when something is wrong with the DatabaseRevisionFiles

class Status(*value*)
Bases: enum.Enum

List of all reasons

FEWER_REVISION_SQL_FILES_FOUND_THAN_REQUIRED_STEPS_SPECIFIED = 103

MORE_REVISION_SQL_FILES_FOUND_THAN_REQUIRED_STEPS_SPECIFIED = 102

NONCONSECUTIVE_REVISION_SQL_FILES_FOR_HEAD_OR_BASE_POINTER = 104

NON_CONFORMANT_REVISION_FILE_NAME_EXISTS = 101

exception dvc.core.exception.InvalidDatabaseVersionException(*database_version: str*)
Bases: Exception

Exception raised when format of Database Version is wrong

exception dvc.core.exception.OperationNotAccountedForException(*operation_type=<enum 'Operation'>*)
Bases: Exception

Exception raised when operation is requested but is not yet developed

exception dvc.core.exception.RequestedDatabaseFlavourNotSupportedException(*requested_database_flavour: str*)
Bases: Exception

Exception raised when requested database flavour is not supported

dvc.core.file module

dvc.core.file.validate_file_exist(*file_path: pathlib.Path*) → None
Throw FileNotFileError if a given file does not exist

dvc.core.hash module

class dvc.core.hash.FileHasher
Bases: object

Hash content of any given file

md5(*file_path: pathlib.Path*) → str
Extract content from a file and hash its output

Parameters file_path – Pathlib Path

Returns string

dvc.core.logger module

class dvc.core.logger.**SetRootLoggingLevel**(*func*)
Bases: object

Used as a decorator to set the root logging level

set_logging_level(*logging_level: int*)

dvc.core.regex module

dvc.core.regex.**get_matched_files_in_folder_by_regex**(*folder_path: pathlib.Path*, *file_name_regex: str*) → List[pathlib.Path]
Loop recursively for all files in a given folder. Return those files whose name satisfy the regex.

Parameters

- **folder_path** – Path pointing tothe folder with files
- **file_name_regex** – regex used to filter for files with the desired file name

Returns

dvc.core.struct module

class dvc.core.struct.**DatabaseRevisionFile**(*file_path: pathlib.Path*)
Bases: object

Raise error when File Path does not conform to standard

STANDARD_RV_FILE_FORMAT_REGEX = '^RV[0-9]+__.*\\.(upgrade|downgrade)\\.sql$'

property description: str
Get the file description

Returns

property ending: str
Get the file ending

Returns

classmethod get_dummy_revision_file(*revision: str*, *operation_type:* dvc.core.struct.Operation) → *dvc.core.struct.DatabaseRevisionFile*
Return a dummy revision file

Parameters

- **revision** –
- **operation_type** –

Returns

property operation_type: *dvc.core.struct.Operation*
Get the operation type

Returns

property revision_number: int
Get the revision number

Returns

class dvc.core.struct.**DatabaseVersion**(*version: str*, *created_at: Optional[datetime.datetime] = None*)

Bases: object

STANDARD_DATABASE_VERSION_FORMAT_REGEX = '^V[0-9]+$'

property created_at

property next_downgrade_database_revision_file: *dvc.core.struct.DatabaseRevisionFile*

Get the database revision file for downgrade :return:

property next_upgrade_database_revision_file: *dvc.core.struct.DatabaseRevisionFile*

Get the database revision file for upgrade :return:

property version

property version_number: int

class dvc.core.struct.**Operation**(*value*)

Bases: enum.Enum

Database Operations Allowed

Downgrade = 'downgrade'

Upgrade = 'upgrade'

Module contents

Submodules

dvc.version module

Module contents

7.2 Tests Package

7.2.1 tests package

Subpackages

tests.assets package

Module contents

tests.test_dvc package

Subpackages

tests.test_dvc.test_app package

Subpackages

tests.test_dvc.test_app.test_cli package

Subpackages

tests.test_dvc.test_app.test_cli.test_commands package

Subpackages

tests.test_dvc.test_app.test_cli.test_commands.test_database package

Submodules

tests.test_dvc.test_app.test_cli.test_commands.test_database.test_backend module

Module contents

Submodules

tests.test_dvc.test_app.test_cli.test_commands.test_config module

tests.test_dvc.test_app.test_cli.test_commands.test_main module

tests.test_dvc.test_app.test_cli.test_commands.test_main.**test__version__only_contain_semver**()
Test *dvc version* command output version in SemVer format

tests.test_dvc.test_app.test_cli.test_commands.test_sql module

Module contents

Module contents

Module contents

tests.test_dvc.test_core package

Submodules

tests.test_dvc.test_core.test_config module

class tests.test_dvc.test_core.test_config.**TestConfigFileWriter**
Bases: object

pytestmark = [Mark(name='unit', args=(), kwargs={})]

test__write_dummy_user_configuration(*dummy_user_configuration_with_supported_db_flavour*, *dummy_absent_config_file_path*)
GIVEN a non-existing config file path WHEN ConfigFileReader.user_config is called THEN check dummy user configuration is returned

class tests.test_dvc.test_core.test_config.**TestConfigReader**
Bases: object

pytestmark = [Mark(name='unit', args=(), kwargs={})]

test__when_both_config_file_and_env_var_and_absent__raise_environment_variables_not_set_exception(*d*
GIVEN a dummy config file with dummy user configuration, WHEN ConfigFileReader.user_config is called THEN check dummy user configuration is returned

test__when_config_file_is_absent_but_env_var_present__return_expected_user_config_from_env_var(*dum*
dum
mon
key-
patc
GIVEN a dummy config file with dummy user configuration, WHEN ConfigFileReader.user_config is called THEN check dummy user configuration is returned

test__when_config_file_is_persent__return_expected_user_config_from_config_file(*dummy_user_configurat*
*dummy_existing_config_*
GIVEN a dummy config file with dummy user configuration, WHEN ConfigFileReader.user_config is called THEN check dummy user configuration is returned

class tests.test_dvc.test_core.test_config.**TestDatabaseConnectionFactory**
Bases: object

pytestmark = [Mark(name='unit', args=(), kwargs={})]

test__pass_user_credentials_to_connect_as_kwargs(*dummy_config_file_reader_with_supported_db_flavour*, *dummy_pgconn*)
(Currently test postgres specifically) GIVEN patched psycopg2.connect WHEN DatabaseConnectionFactory.conn is called THEN check psycopg2.connect is called once and with expected args

test__raise_requested_database_not_supported_exception(*dummy_config_file_reader_with_unsupported_db_flavou*
GIVEN a fake database flavour WHEN DatabaseConnectionFactory.validate_requested_database_flavour is called THEN assert RequestedDatabaseFlavourNotSupportedException is raised

class tests.test_dvc.test_core.test_config.**TestDatabaseRevisionFilesManager**
Bases: object

pytestmark = [Mark(name='unit', args=(), kwargs={})]

test__get_target_database_revision_files_by_pointer(*dummy_config_file_reader_with_supported_db_flavour*, *current_database_version*, *candidate_database_revision_files*, *steps*, *expected_database_revision_files*, *expected_exception*)

tests.test_dvc.test_core.test_logger module

class tests.test_dvc.test_core.test_logger.**TestSetRootLoggingLevel**
Bases: object

Test Set RootLoggingLevel as a decorator

test__when_config_file_and_env_var_are_absent__set_to_default_logging_level(*any_func*, *dummy_user_configuration_v*
dummy_absent_config_file_pa
GIVEN config file is absent and no env var is set WHEN SetRootLoggingLevel is called THEN use default logging level

Parameters

- **any_func** –
- **dummy_user_configuration_with_supported_db_flavour** –
- **dummy_absent_config_file_path** –

Returns

test__when_config_file_is_absent_but_env_var_is_present__set_to_user_defined_logging_level(*any_func*, *dummy_u*, *dummy_a*)

GIVEN config file is absent but env vars are set WHEN SetRootLoggingLevel is called THEN use the env vars' logging levels

Parameters

- **any_func** –
- **dummy_user_configuration_with_supported_db_flavour** –
- **dummy_absent_config_file_path_with_env_var** –

Returns

test__when_config_file_is_present_but_env_var_is_absent__set_to_config_file_logging_level(*any_func*, *dummy_use*, *dummy_exi*)

GIVEN config file is present and no env vars are set WHEN SetRootLoggingLevel is called THEN use the config file' logging level

Parameters

- **any_func** –
- **dummy_user_configuration_with_supported_db_flavour** –
- **dummy_existing_config_file_path** –

Returns

tests.test_dvc.test_core.test_logger.**any_func**() → Callable

Return a function which accepts any args and kwargs, but does nothing :return:

tests.test_dvc.test_core.test_regex module

class tests.test_dvc.test_core.test_regex.**TestGetMatchedFilesInFolderByRegex**

Bases: object

pytestmark = [Mark(name='unit', args=(), kwargs={})]

test__get_matched_files_in_folder_by_regex__assert_number_sql_files(*dummy_regex_files_folder_with_correct*, *file_name_regex*, *expected_num_matched_files_paths*)

GIVEN a dummy folder with dummy files WHEN get_matched_files_in_folder_by_regex is called with a certain regex THEN the returned paths should match the regex

tests.test_dvc.test_core.test_struct module

class tests.test_dvc.test_core.test_struct.**TestDatabaseRevisionFile**
Bases: object

test_database_revision_files_comparison(*file_1*, *file_2*, *predicate*, *expected*)

test_valid_database_revision_files(*sql_file_name: str*, *expectation*)

class tests.test_dvc.test_core.test_struct.**TestDatabaseVersion**
Bases: object

test_valid_dummy_database_revision_files_with_order(*target_database_version:* dvc.core.struct.DatabaseVersion, *current_database_version:* dvc.core.struct.DatabaseVersion, *expected_dummy_database_revision_files: List[*dvc.core.struct.DatabaseRevisionFile*]*)

Module contents

Module contents

Submodules

tests.conftest module

Module contents

7.3 Test Report

- See latest test report: here

CHAPTER

EIGHT

CHANGE LOG

8.1 0.5.1

- Removed bugs which throw errors when config.yaml is absent without fallback on environment variables.
- Fixed wrong regex for testing SemVer pattern.

8.2 0.5.0

- Added –dry-run to both *dvc db upgrade* and *dvc db downgrade*. When set to True, the CLI will stop before the execution of SQL files
- **For DatabaseRevisionFilesManager class:**
  - Added tests for i. getting files by pointer and ii. getting files by steps methods
  - Made both methods use the same exception handling raise_for_status() method
- Added option to change logging level in config file and env variables
- Added demo files and step-by-step guide under in the feature pages
- Added *dvc cfg show* to show some configurations

8.3 0.4.0

- Added dynamic generation of src package and tests package documentation in docs with sphinx-apidoc.
- Added –head to *dvc db upgrade* and –base *dvc db downgrade* respectively

8.4 0.3.3

- Added CI testing for binaries generated for 3 Oses. For Windows, Ubuntu Linux and Mac, run the below in the CI pipeline:
  - Build binary
  - Run postgres server. Run all DVC commands against it.
  - Push to Github release

8.5 0.3.2

- Included ./setup.cfg to Dockerfile. Fixed missing .sql files in Docker Image to Dockerhub

8.6 0.3.1

- Moved *package_data* to setup.cfg. Fixed missing .sql files in PyPI.

8.7 0.3.0

- Added –steps and –confirm flags to *dvc db upgrade* and *dvc db downgrade*
- **Added dunder methods for the below classes:**
  - __le__, __gr__, __eq__ for DatabaseRevisionFile
  - __add__, __sub__, __eq__ for DatabaseVersion
- **Codified the below relationship with dunder methods:**
  - *DatabaseVersion - DatabaseVersion = [DatabaseRevisionFiles]*
  - *DatabaseVersion + DatabaseRevisionFile = DatabaseVersion*
- Removed *dvc sql generate*, as that is Files System related.
- Added diagram to illustrate DatabaseVersion and DatabaseRevisionFile

8.8 0.2.1

- **Refactored Github workflows. Separated the below components from Github Workflows**
  - Running pytest
  - Building and pushing python library to PyPI
  - Building and Pushing artifacts to Readthedocs
  - Building and pushing to Dockerhub
  - Building and pushing Linux, Mac and Windows binaries to Github Releases

8.9 0.2.0

- Included help text and documentation URL in the CLI.
- Followed SemVer more closely. Bumped minor version with added feature and bump patch for bug fixes. Switched to using tag (not branch) for releases.

8.10 0.1.16

- Created binaries for windows, mac and linux with pyinstaller. added sql files to the binaries.
- Removed confirmation for both upgrade and downgrade command (i.e. *dvc db upgrade* and *dvc db downgrade*)

8.11 0.1.15

- Update Dockerfile. Changed instruction `CMD` to `ENTRYPOINT` for `dvc` command.
- Added pages to documentation using sphinx
- Created graphs using graphviz and dots
- **Added Github Action workflows to automate the below:**
  - generating pytest report artifacts
  - pushing both artifacts and .rst files to Readtheedocs for building documentation

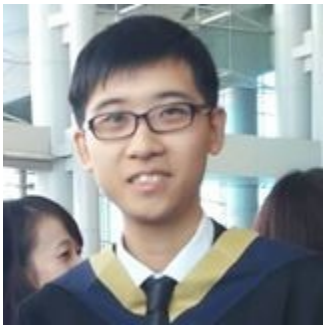
8.12 0.1.14

- Created ConfigReader, ConfigFileWriter and other objects to hold states.
- Added the option to pass configuration as environment variables.
- Added unit tests and integrations tests (for postgres)
- Created Dockerfile to containerise the tool
- Created docker-compose.yml file for demonstration purposes. Created demo_assets to be attached as volume to docker-compose containers.
- **Added Github Action workflows to automate the below:**
  - Pushing Docker image (with different tags) and readme to Dockerhub.
  - Pushing the tool to PYPI

CHAPTER

NINE

AUTHOR



9.1 About Me

- Language enthusiast
- Fan of automation via technology

9.2 Contact

Contact me via:

- linkedin

CHAPTER

TEN

SUMMARY

Compute (Application) and Storage (Database) are decoupled.

When you make changes to your application code, you should also mke changes to your database. In other words, you probably want to version control both your application code and your database. Without version controlling both, any changes in either side can cause incompatibility issues and break the entire service as a whole.

Use DVC now to version control your database!

CHAPTER

ELEVEN

DESCRIPTION

Database Version Control (DVC) is a CLI utility which version controls your database in the following ways:

- Generate metadata table(s) in your database;
- For each SQL script applied, update the metadata table(s);
- Exposes the metadata via CLI commands.

CHAPTER

TWELVE

BENEFITS

- Rich metadata is available in the database. The database can be directly queried with SQL for both historical and current database versions.
- Only plain SQL files are accepted. No extra abstraction layer as is generally available in ORM.

CHAPTER

THIRTEEN

MORE INFORMATION

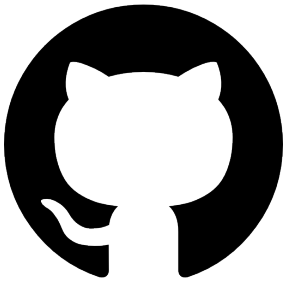


Fig. 1: Code on GitHub

Fig. 2: Docker Image on Dockerhub

Fig. 3: Documentation on Readthedocs (latest)

Fig. 4: Demo on Youtube

CHAPTER

FOURTEEN

SUPPORTED DATABASES

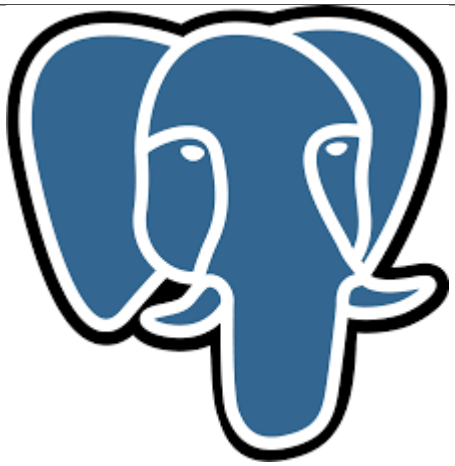


Fig. 1: Postgresql

PYTHON MODULE INDEX

d

t

INDEX

N

O

P

R

S

T

U

V

W