dbtvault Contributor Quick Setup

Prerequisites

- Python >= 3.9.6
- Pipenv

Install dependencies

Run `pipenv install --dev`

Setting up your development environment

1. Getting the code

For external contributors

In general, we follow the "fork-and-pull" Git workflow

- Fork the repository to your own GitHub account
- Clone the project to your machine
- Create a branch (from `develop`) locally with a succinct but descriptive name.
- If it's a new feature, prefix the branch name with `feat/`
- Commit changes to the branch
- Push changes to your fork
- Open a PR in our repository

2. Create environment files from templates (EXTERNAL ONLY)

In the `<project_root>/env/templates` folder you will find two files: `db.tpl.env` and `profiles.tpl.yml`.

profiles_external.tpl.yml

This file is a template which can be used as a drop-in replacement for your dbt `profiles.yml` file, or added to an existing `profiles.yml`.

Inside your python environment, run: `inv init-external -p <platform>`

Platform may be one of:

- `snowflake`
- `bigquery`
- `sqlserver`
- `databricks`

This will create a file: `env/profiles.yml`, which will look something like this:

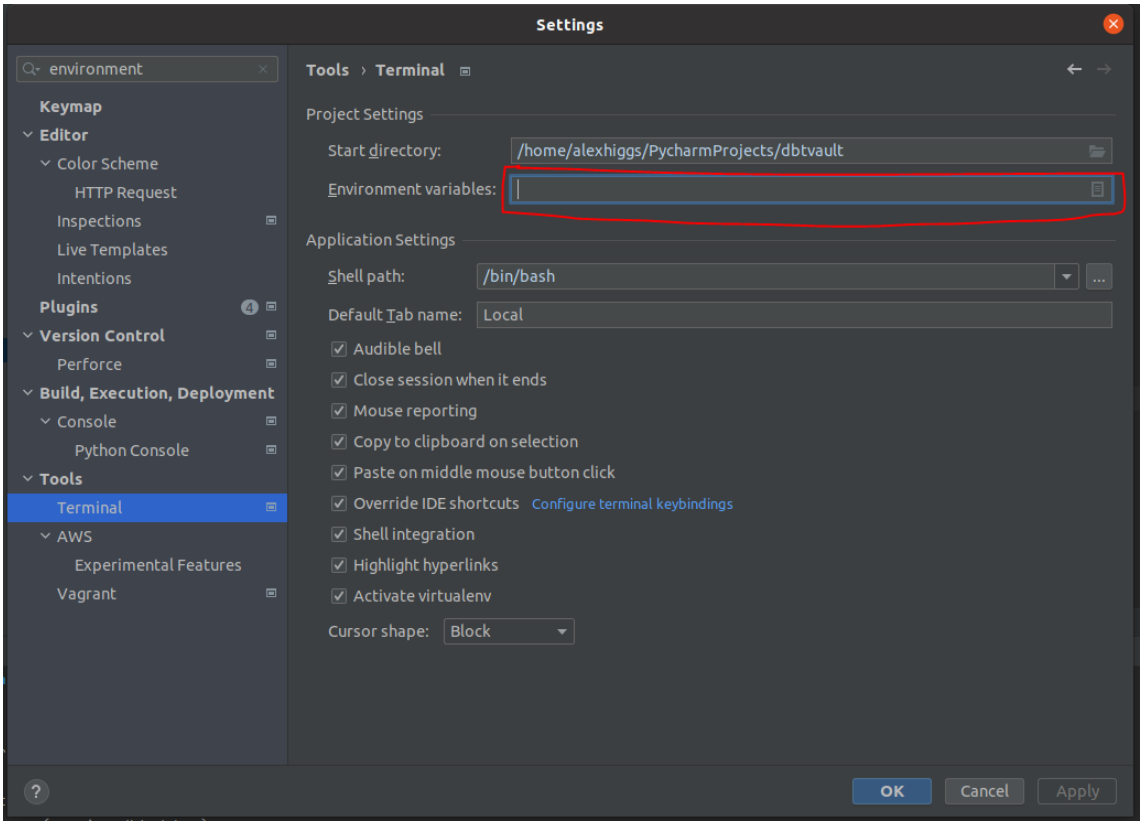
```
dbtvault:
  outputs:
    databricks:
      type: spark
      method: odbc
      driver: /opt/simba/spark/lib/64/libsparkodbc_sb64.so
      schema: "{{ env_var('DATABRICKS_SCHEMA') }}"
      host: "{{ env_var('DATABRICKS_HOST') }}"
      port: "{{ env_var('DATABRICKS_PORT', 443) | as_number }}"
      token: "{{ env_var('DATABRICKS_TOKEN') }}"
      endpoint: "{{ env_var('DATABRICKS_ENDPOINT') }}"

      threads: 4

  target: databricks
```

You may now set up your environment variables, any way you wish. Some examples are below.

PyCharm



You may set up session-scoped (i.e. Only when PyCharm is open) environment variables in PyCharm, as per the above screenshot.

1. Navigate to the following: Settings > Tools > Terminal > Environment Variables
2. Paste one of the following into the text box:
    - snowflake
      `SNOWFLAKE_DB_ACCOUNT=;SNOWFLAKE_DB_USER=;SNOWFLAKE_DB_PW=;SNOWFLAKE_DB_ROLE=;SNOWFLAKE_DB_DATABASE=;SNOWFLAKE_DB_WH=;SNOWFL`
    - sqlserver
      `SQLSERVER_DB_SERVER=;SQLSERVER_DB_PORT=;SQLSERVER_DB_DATABASE=;SQLSERVER_DB_SCHEMA=;SQLSERVER_DB_USER=;SQLSERVER_DB_PW=;`
    - bigquery `GCP_PROJECT_ID=;GCP_DATASET=;`
    - databricks `DATABRICKS_SCHEMA=;DATABRICKS_HOST=;DATABRICKS_PORT=;DATABRICKS_TOKEN=;DATABRICKS_ENDPOINT=;`
3. Update the values for each variable by clicking the 
4. Start a new PyCharm terminal and check that the environment variables have been set, e.g. on Linux: `printenv`

Read the dbt docs to learn more about `profiles.yml`

3. Run the setup command

Without 1Password integration

Recommended for External Contributors

Inside the virtual environment, run `inv setup -p <platform> -d`

e.g. `inv setup -p snowflake -d`

With 1Password integration

Recommended for Datavault Employees

- First, sign in to 1Password, following internal guides.
- Inside the virtual environment, run `inv setup -p <platform>`

e.g. `inv setup -p snowflake`

Options:

`-p, --platform` Sets the development platform environment which the dbtvault test harness will be executed under.

`platform` must be one of:

- snowflake
- bigquery
- sqlserver

`-d, --disable-op` Disables 1Password CLI integration, which is used by Datavault Developers internally for secrets management.

A successful run should produce something similar to the following output:

```
(dbtvault) INFO: Defaults set.
(dbtvault) INFO: Project: test
(dbtvault) INFO: Platform: snowflake
(dbtvault) INFO: Platform set to 'snowflake'
(dbtvault) INFO: Project set to 'test'
(dbtvault) INFO: Checking dbt connection... (running dbt debug)
(dbtvault) INFO: Project 'test' is available at: '.../dbtvault/test/dbtvault_test'
Running with dbt=0.20.0
dbt version: 0.20.0
python version: 3.9.0
python path: ~/venvs/dbtvault/bin/python
os info: Linux-5.4.0-81-generic-x86_64-with-glibc2.31
Using profiles.yml file at ~/.dbt/profiles.yml
Using dbt_project.yml file at .../dbtvault/test/dbtvault_test/dbt_project.yml

Configuration:
  profiles.yml file [OK found and valid]
  dbt_project.yml file [OK found and valid]

Required dependencies:
 - git [OK found]

Connection:
  account: <redacted>
  user: <redacted>
  database: <redacted>
  schema: <redacted>
  warehouse: <redacted>
  role: <redacted>
  client_session_keep_alive: False
  Connection test: OK connection ok

(dbtvault) INFO: Installing dbtvault-dev in test project...
(dbtvault) INFO: Project 'test' is available at: '.../dbtvault/test/dbtvault_test'
Running with dbt=0.20.0
Installing ../../dbtvault-dev
  Installed from <local @ ../../dbtvault-dev>
Installing dbt-labs/dbt_utils@0.7.0
  Installed from version 0.7.0
(dbtvault) INFO: Setup complete!
```