



WhizAI Training Handbook

whiz.ai

Training Handbook

Revision Date: 23 September 2021

This documentation has been created for software version 1.0.45

It is also valid for subsequent software versions as long as no new document version is shipped with the product.

whiz.ai

Suite 105, 220 Davidson Ave, Somerset, NJ, USA 08873

Support:

For more information, visit <https://whiz.ai/contact-us/>

Copyright © 2021 whiz.ai All Rights Reserved.

Trademarks owned by whiz.ai

Disclaimer

No Warranties and Limitation of Liability

Every effort has been made to ensure that this document is an accurate representation of the functionality of WHIZAI platform. However, the development of the software is a continuous process for new features and change. So, small inconsistencies may occur. We would appreciate any feedback on this manual. Send comments via email to: assistant@whiz.ai

Table of Contents

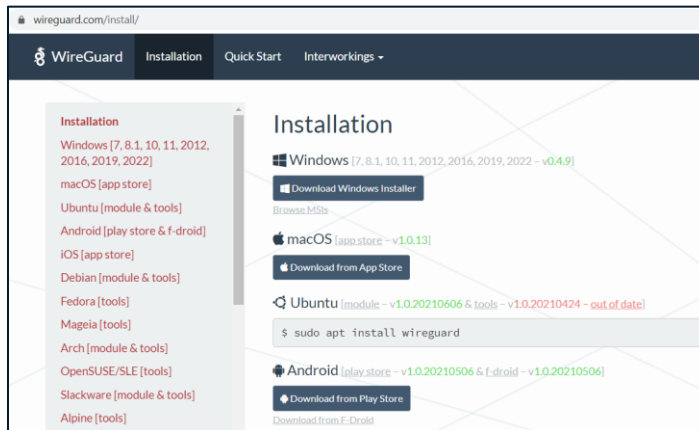
| | |
|---|----|
| Prerequisite softwares | 3 |
| Configure WhizAI Training VPN | 4 |
| Generate keys to access WhizAI Platform | 6 |
| Download and Setup Putty | 6 |
| Download and Setup Super Putty | 6 |
| Generating a new key Using PuTTYgen | 6 |
| Accessing WhizAI Platform | 9 |
| Configure SuperPutty and Access WhizAI Server | 9 |
| Configure WhizAI Application Database | 14 |
| Access WhizAI Airflow | 17 |
| Access Druid Database | 17 |
| Configure WinSCP | 18 |
| Accessing WhizAI Application | 20 |
| Configure Field Analytics Model | 22 |
| Update config.json file | 22 |
| Update environment. json file | 23 |
| Import Configurations from SQLs | 25 |
| Import Help responses | 25 |
| Create Data Model | 26 |
| Getting Started with Data Dictionary | 28 |
| Execute Airflow DAG | 31 |

Prerequisite softwares

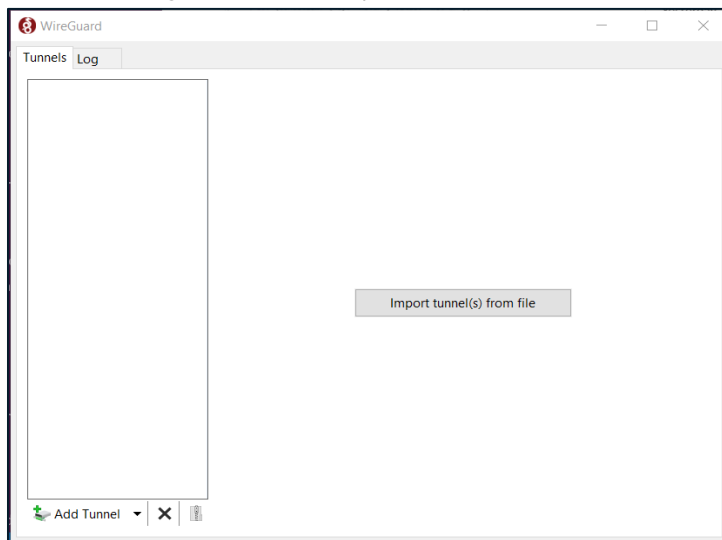
| Software | Purpose | Details |
|---------------------------------|----------------------------|---|
| DBeaver Universal Database Tool | Database Client | https://dbeaver.io/download/ |
| PuTTY | SSH Config Manager | PuTTY |
| SuperPuTTY | SSH Client | SuperPuTTY |
| PuTTYGen | Key Generator | PuTTY |
| Python/ Text IDE | Edit Python and Text Files | https://www.programiz.com/python-programming/ide |
| WinSCP | SFTP Client | https://winscp.net/eng/download.php |

Configuring WhizAI Training VPN

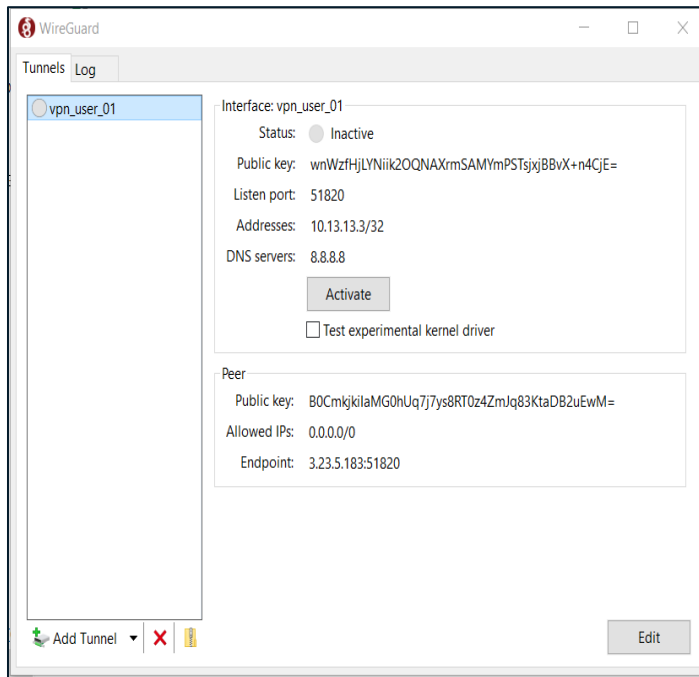
1. Download and Install WireGuard VPN Client : <https://www.wireguard.com/install/>



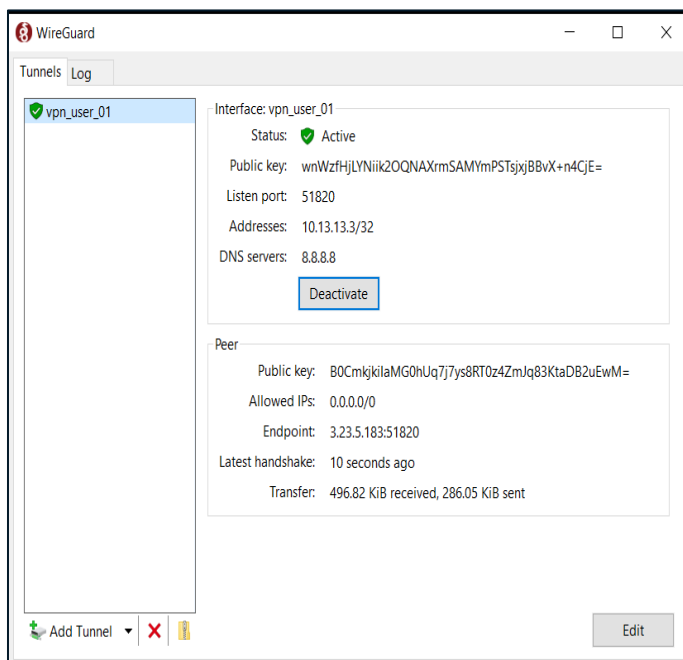
2. After Installing VPN Client , open client on workstation



3. Click Import Tunnel and select VPN Config file



4. Click Activate, VPN Status should be shown as Active



5. Navigate to `C:\Windows\System32\drivers\etc`
6. Open hosts file with Admin Privilege
7. Edit Host File to add following details as per shared server details

```
#ZS Training Server
<Server IP Address> zs-training-XX.whiz.ai
```

Generating keys to access WhizAI Platform

Download and Setup Putty

Follow the instructions to download and configure PuTTY on Windows workstation using following link. You can download putty.zip which consist of all required clients as single archive.

https://www.puttygen.com/download-putty#PuTTY_for_windows

Download and Setup Super Putty

Follow the instructions to download and configure SuperPuTTY on Windows workstation using following link. You can download zip archive or MSI installer.

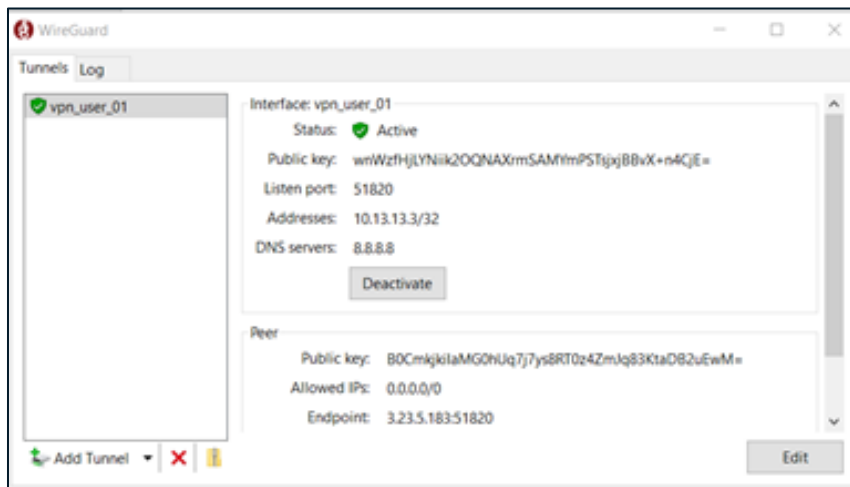
<https://www.puttygen.com/superputty>

Generating a new key Using PuTTYgen

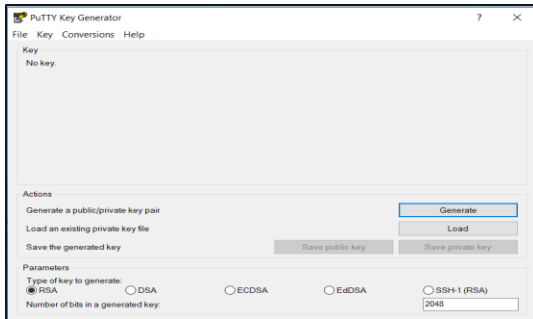


PuTTYgen is a key generator. It generates pairs of public and private keys to be used with PuTTY, PSCP, and Plink, as well as the PuTTY authentication agent, Pageant. PuTTYgen generates RSA and DSA keys.

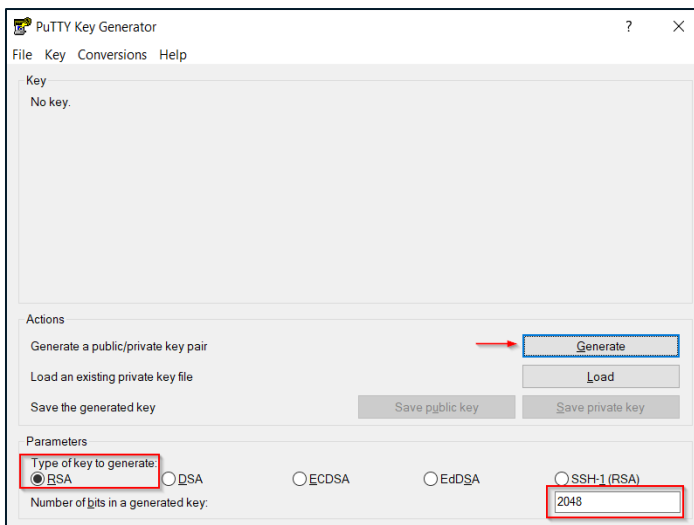
1. Ensure that VPN Connection is Activated



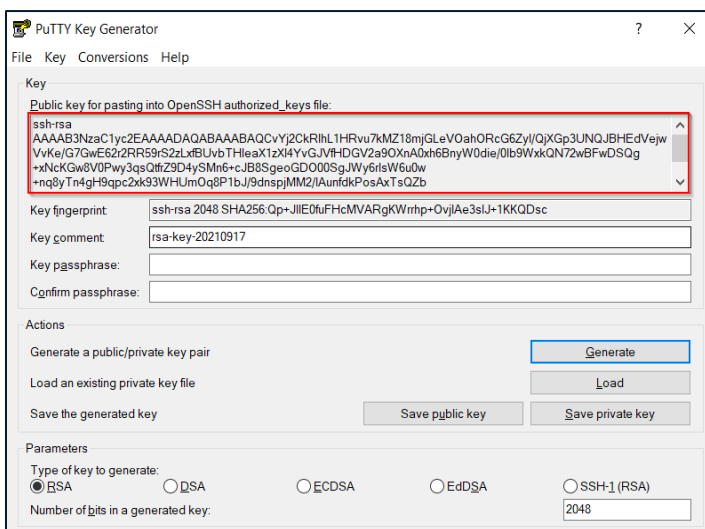
2. Launch PuTTYgen



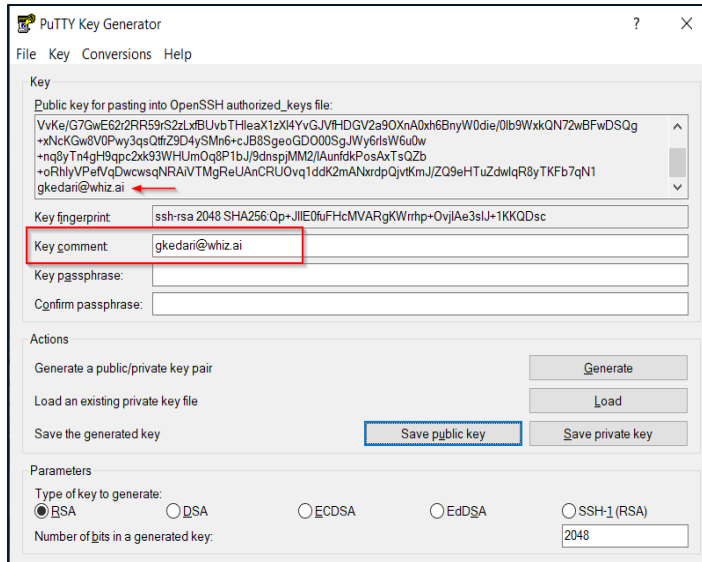
3. Select Type as RSA and Number of bits as 2048 , Click on Generate



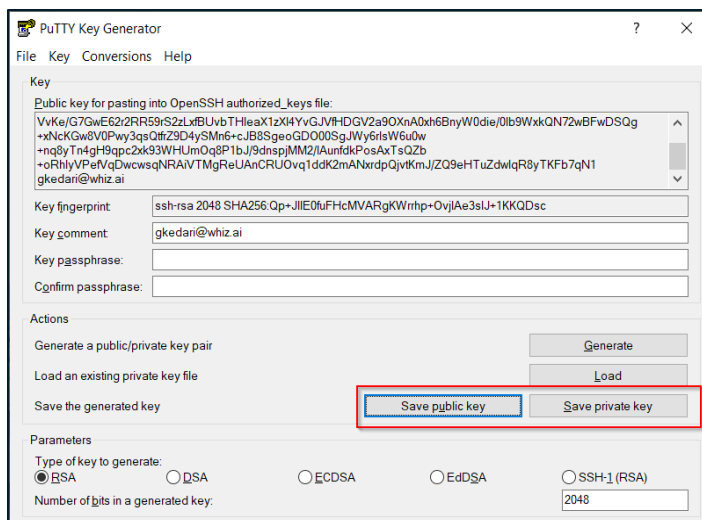
4. RSA Key will be created in Keys Box



5. Ensure that key start with ssh-rsa token
6. Update Key Comment as Email Address



7. Click on Save Public Key and Save Private Key , Save both files on filesystem to use later.

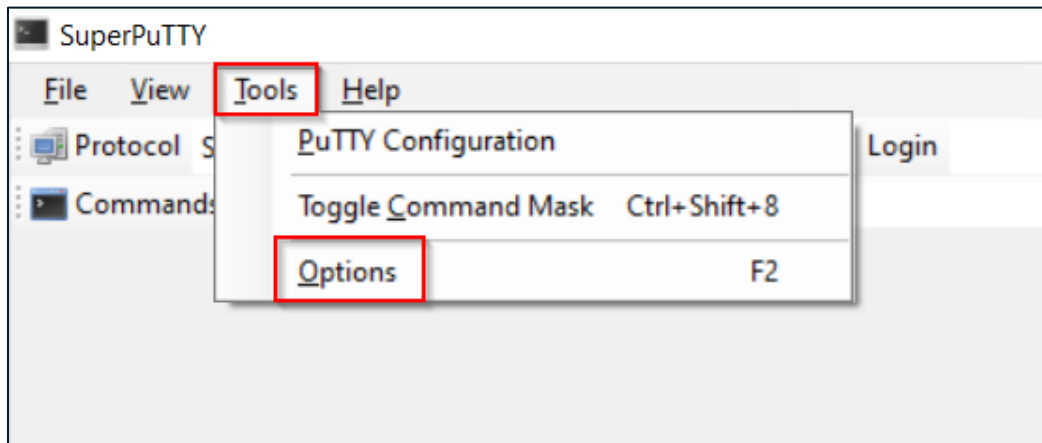


8. Share Public key with WhizAI team for enabling access on Training Server.

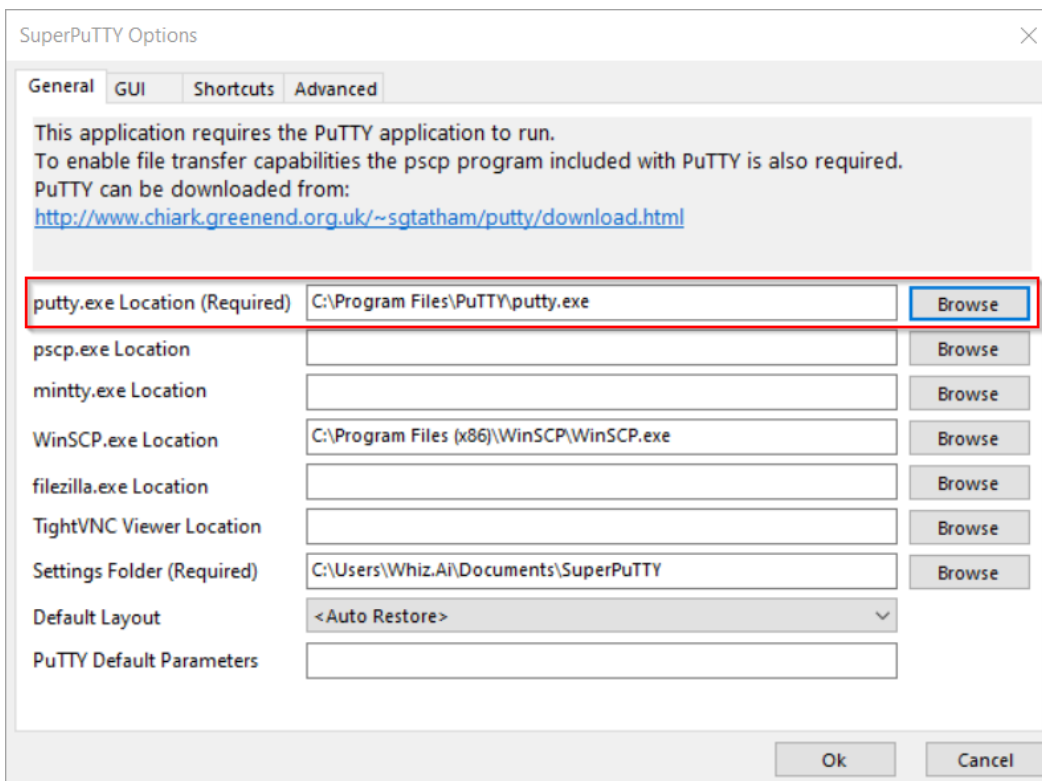
Accessing WhizAI Platform

Configure SuperPutty and Access WhizAI Server

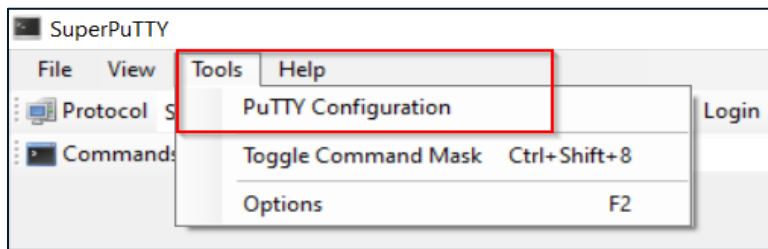
1. Launch SuperPutty and navigate to Tools > Options , Or Press F2 on SuperPutty Screen



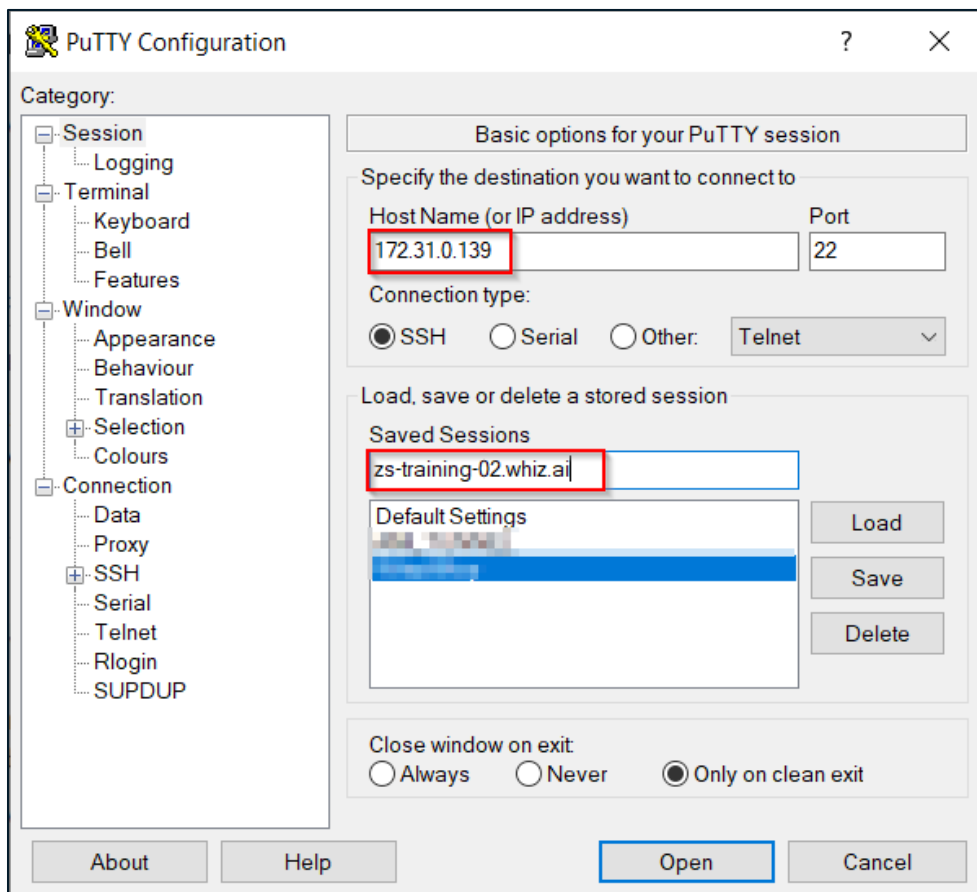
2. Select Puty Location and Click OK



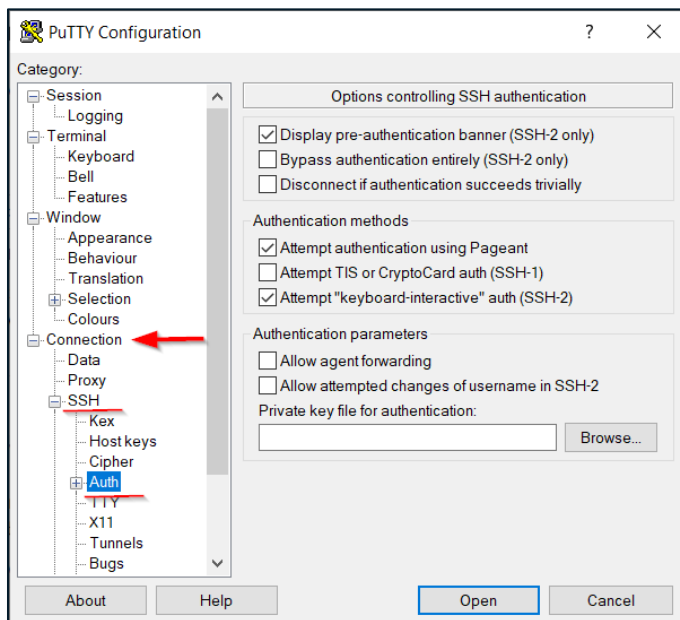
3. Navigate to Tools > Putty Configurations



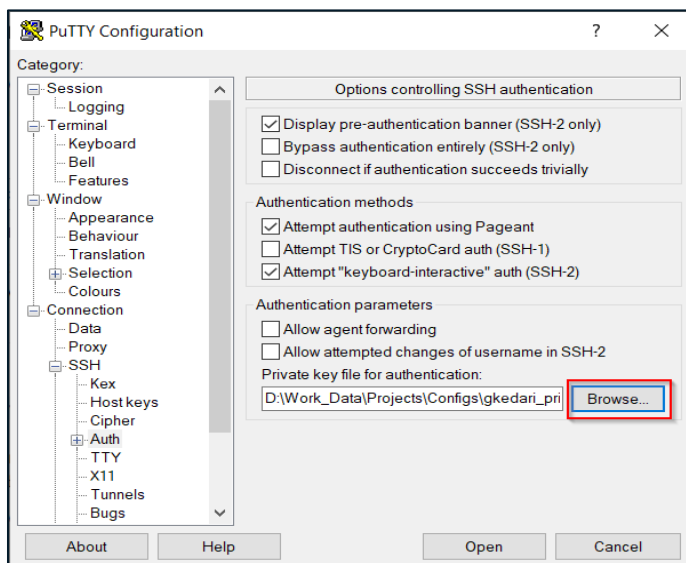
4. Add Host IP for Training server provided by WhizAI team , In Saved sessions text box, enter name as Host name for given server and Click Save



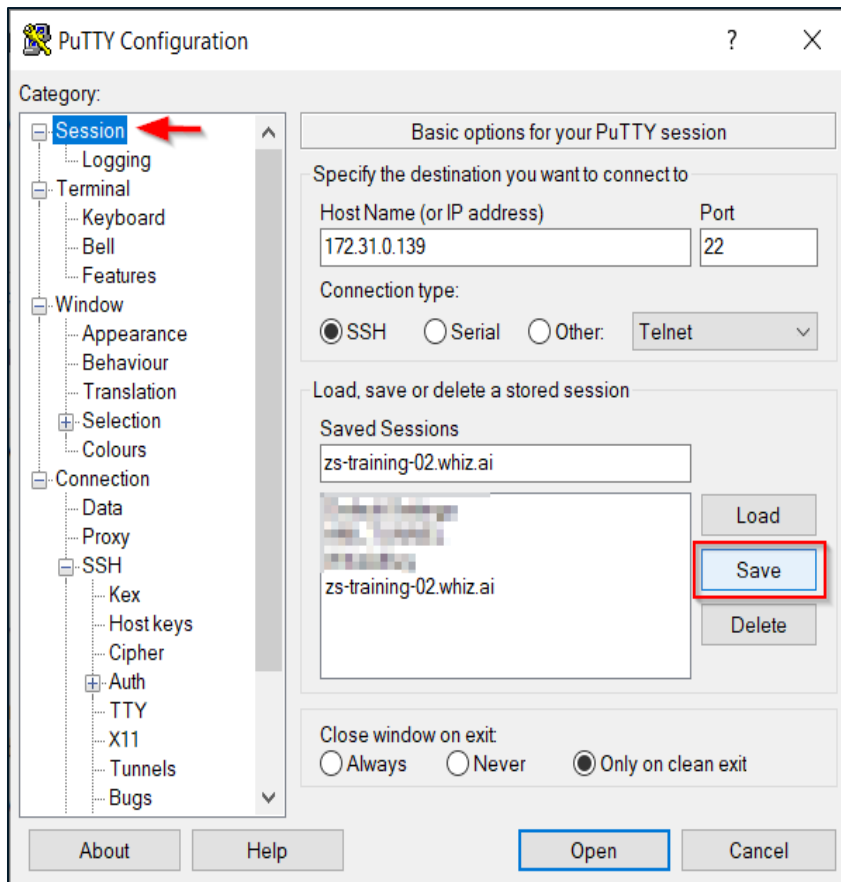
5. In Putty Configuration, navigate to Connection>SSH>Auth



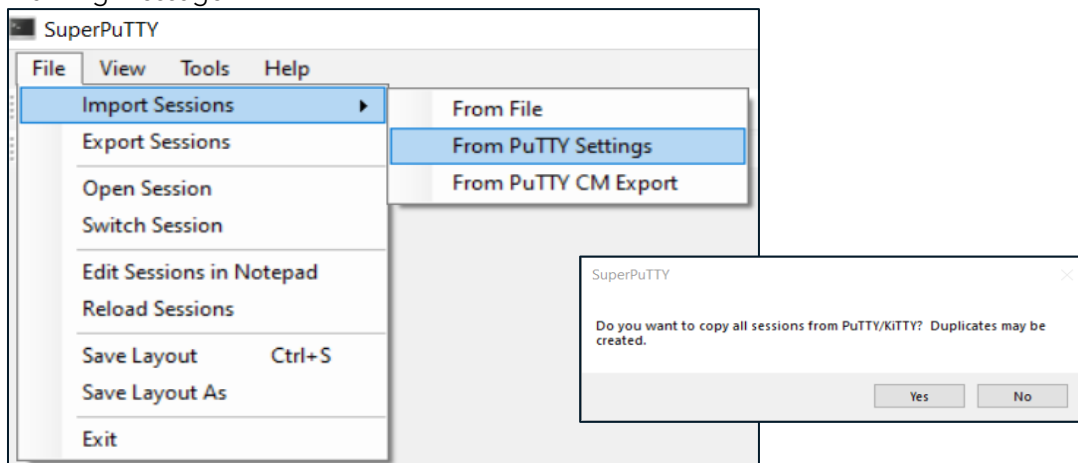
6. Select Private key generated in earlier Step



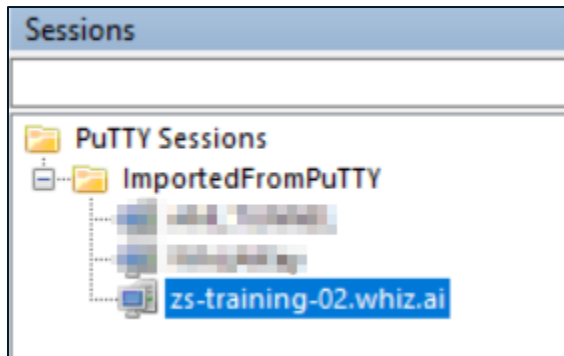
7. Navigate to Session and Click Save. Ensure that IP and Session Name is selected in Saved Session Textbox before Saving.



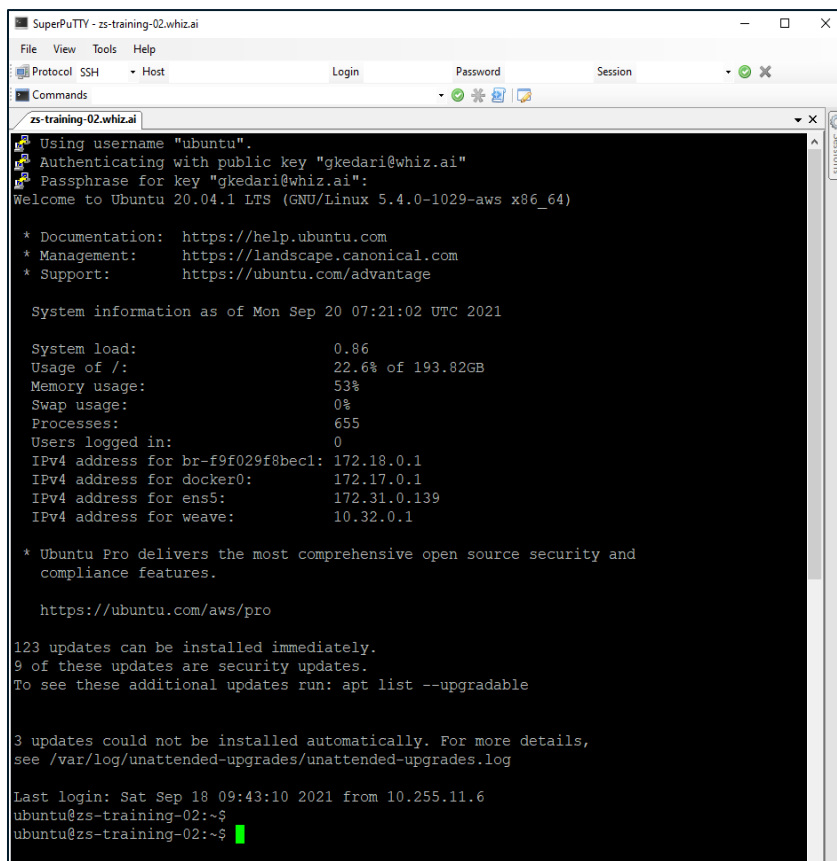
8. Close Putty Configuration
9. In Superputty , navigate to File > Import Sessions > From Putty Settings , Select Yes for Duplicate Warning message .



10. On Sessions Tab , notice that Newly added Session is imported Under *ImportedFromPuTTY* folder



11. Double Click on a session to launch SSH Client for this server.



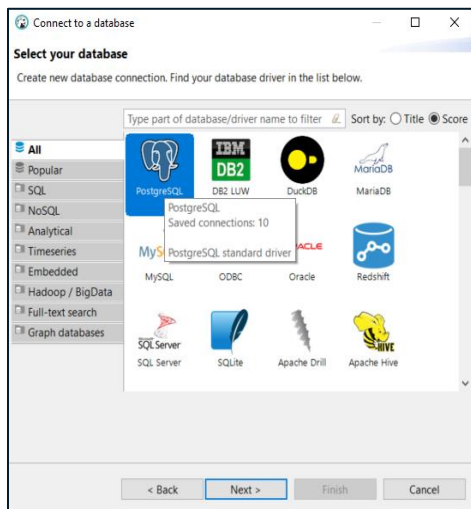
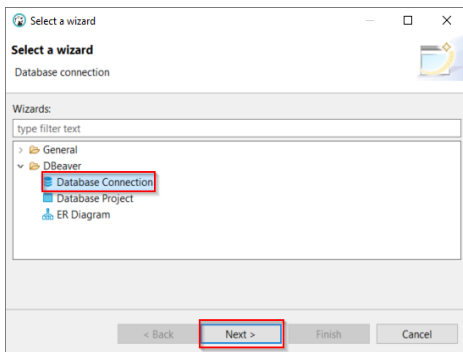
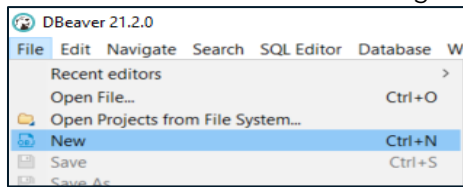
12. Navigate to `/app/k8s/druid-data/airflow/dags`

```
cd /app/k8s/druid-data/airflow/dags/
```

```
ubuntu@zs-training-02:/app/k8s/druid-data/airflow/dags$ cd /app/k8s/druid-data/airflow/dags/
ubuntu@zs-training-02:/app/k8s/druid-data/airflow/dags$ pwd
/app/k8s/druid-data/airflow/dags
ubuntu@zs-training-02:/app/k8s/druid-data/airflow/dags$
```

Configure WhizAI Application Database

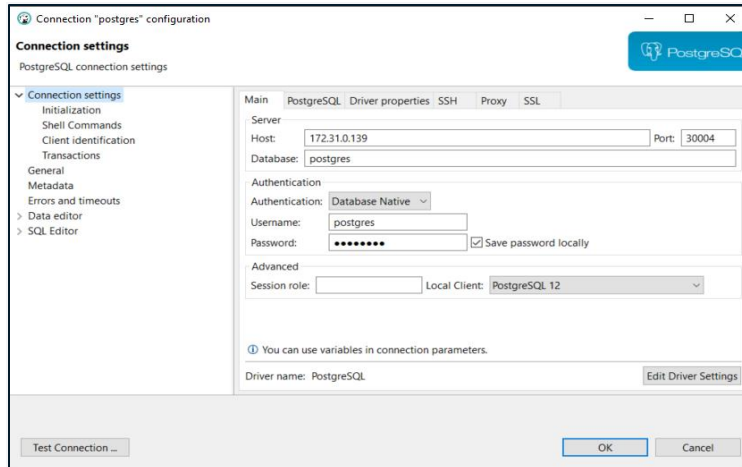
1. Create Database Connection using DBeaver



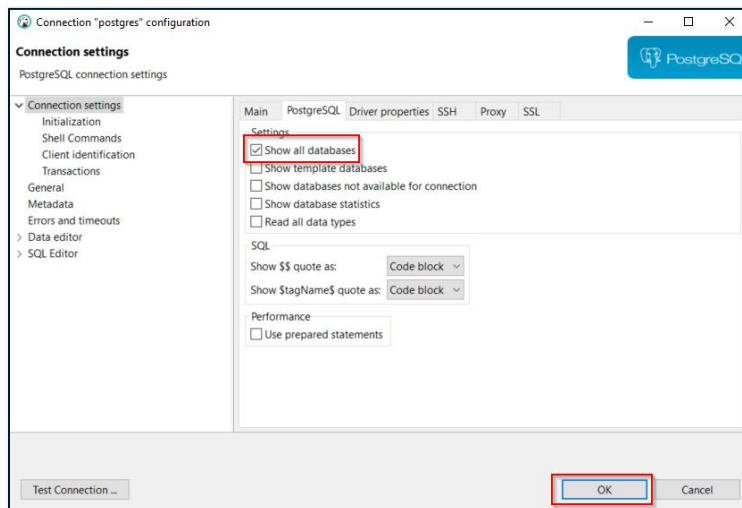
2. Configure Server Details :

Sever :

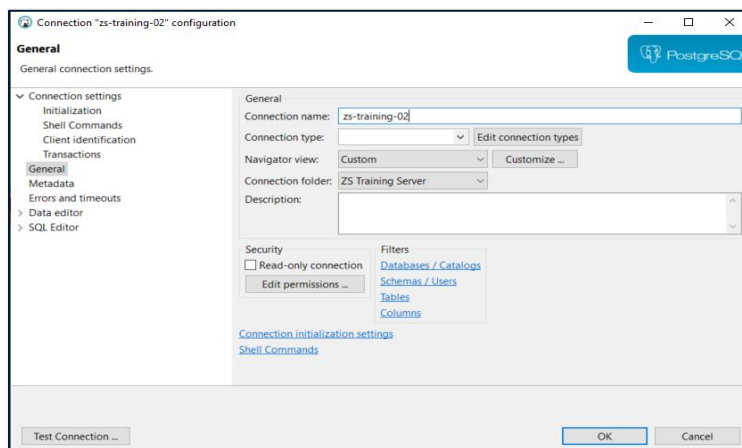
| | |
|----------|-------------|
| Host | : <HOST IP> |
| Port | : 30004 |
| User | : postgres |
| Password | : whizuser |



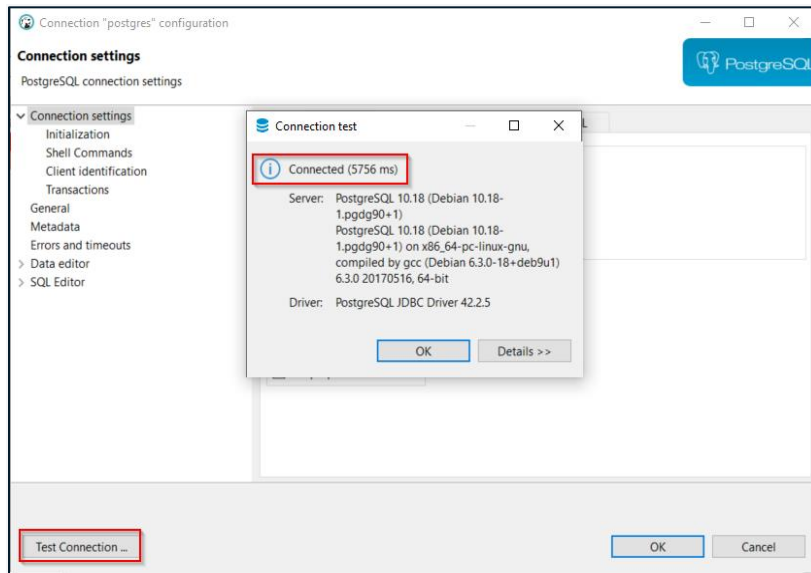
3. Select Show all databases option for Connection Settings



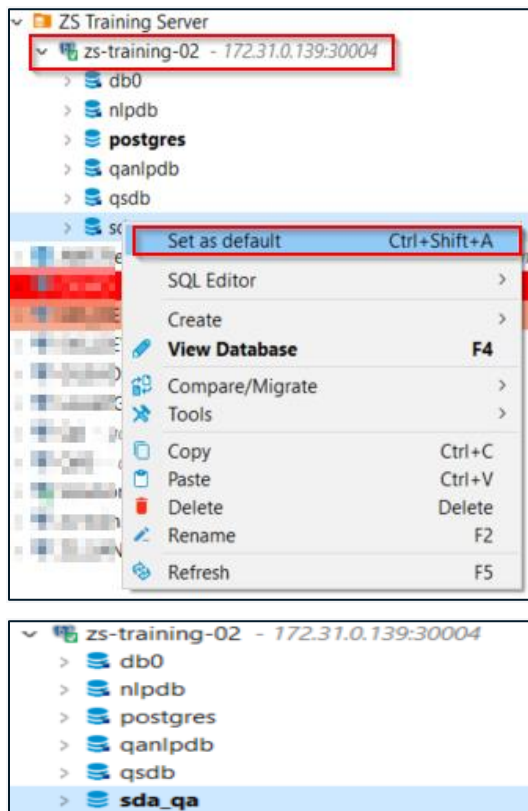
4. Select Show all databases option for Connection Settings



5. Click OK and Test Connection

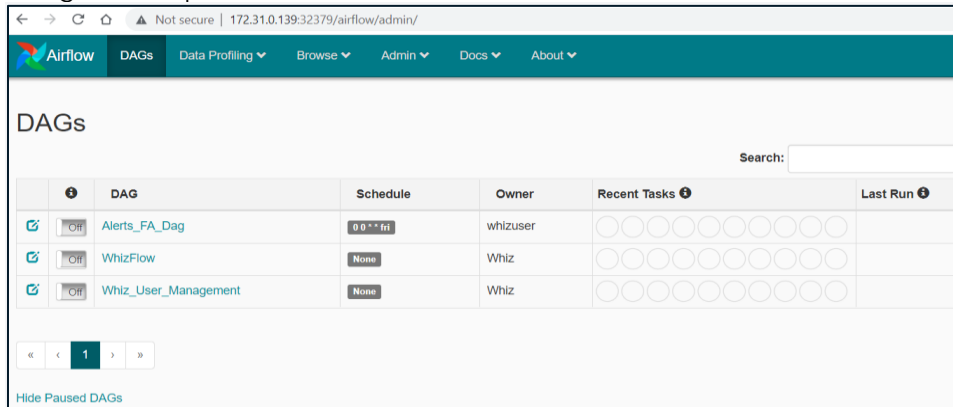


6. Set sda_qa database as default database



Accessing WhizAI Airflow

1. Navigate to `http://<Hostname>:32379/airflow/admin/`

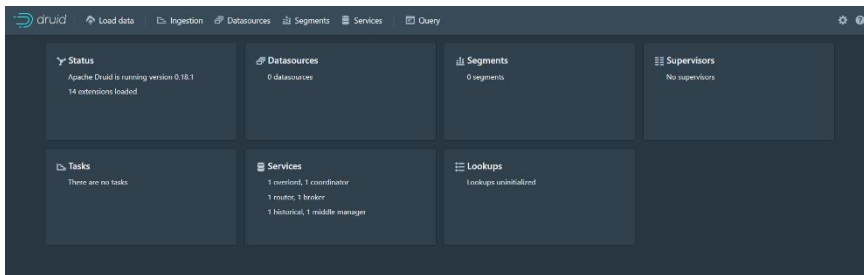


2. Ensure that you can view 3 DAGs configured

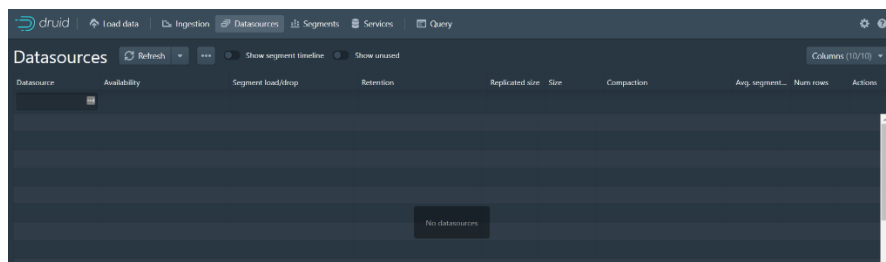
Accessing Druid Database

1. Connect to Druid Co-Ordinator URL

`http://<ServerIP>:32669/unified-console.html`

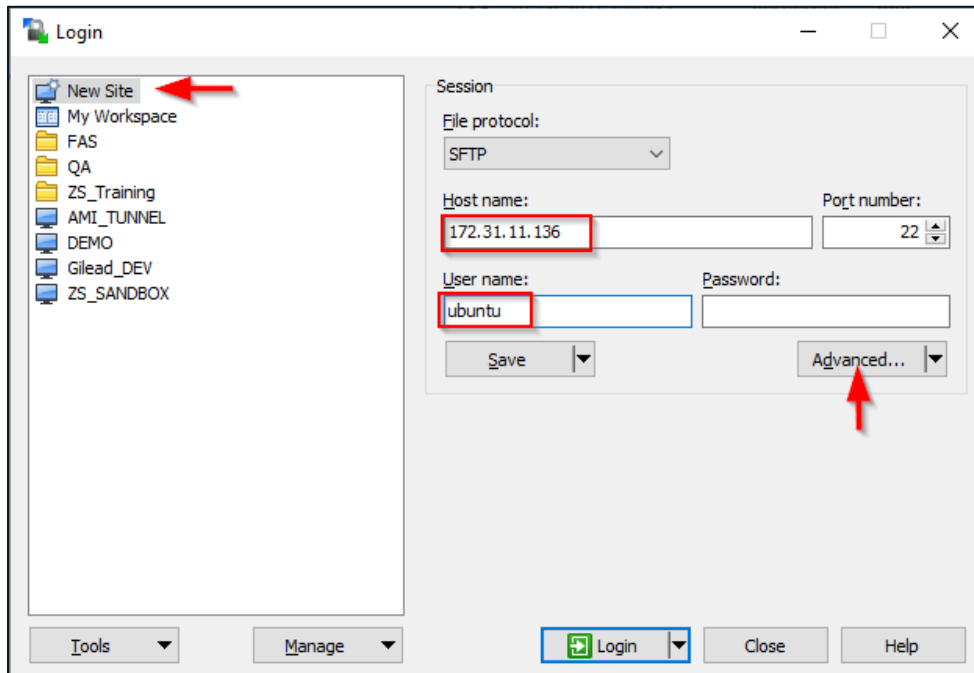


2. Navigate to Datasources Page

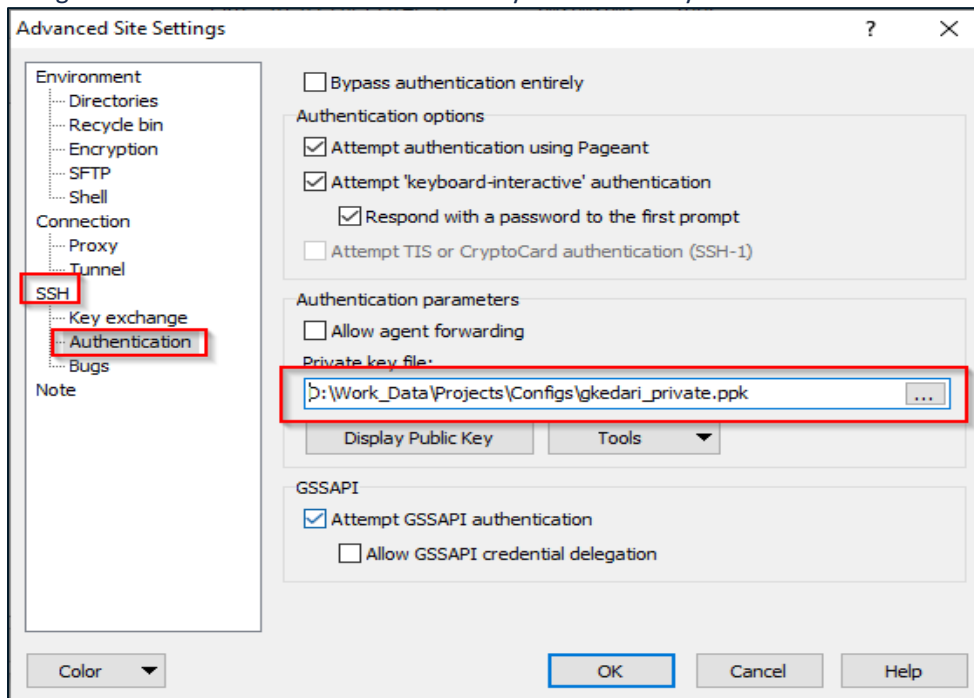


Configuring WinSCP

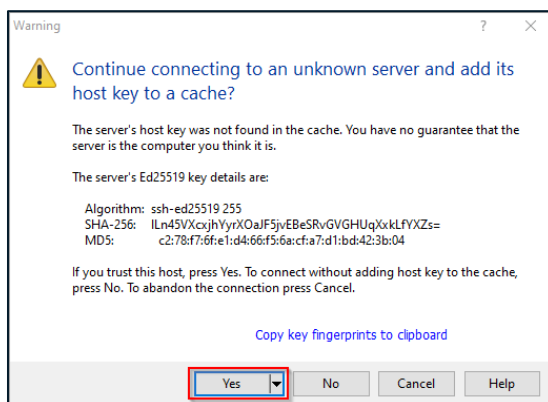
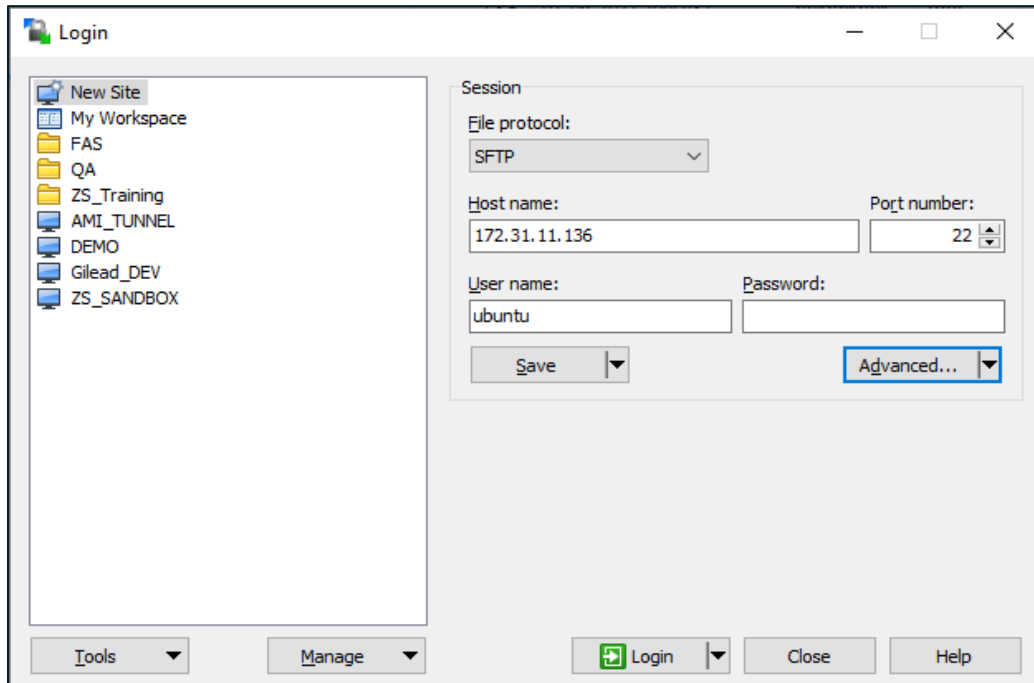
1. Open WinSCP and click on New Site
2. Add Hostname as IP of training server and user as ubuntu , Click Advance



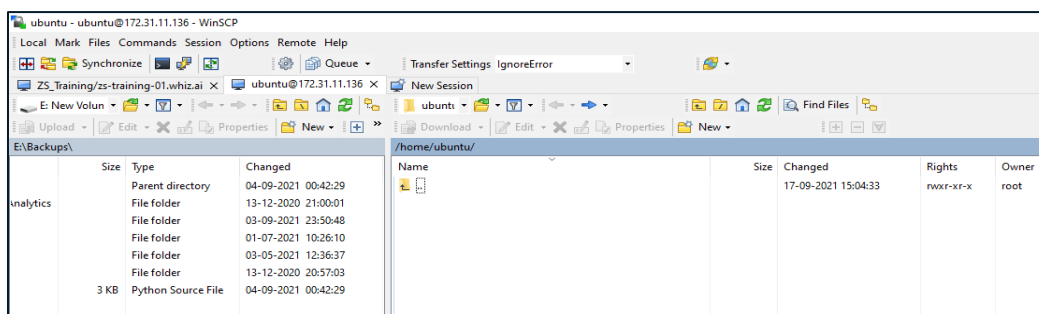
3. Navigate to SSH > Authentication and select your Private Key



4. Click Login

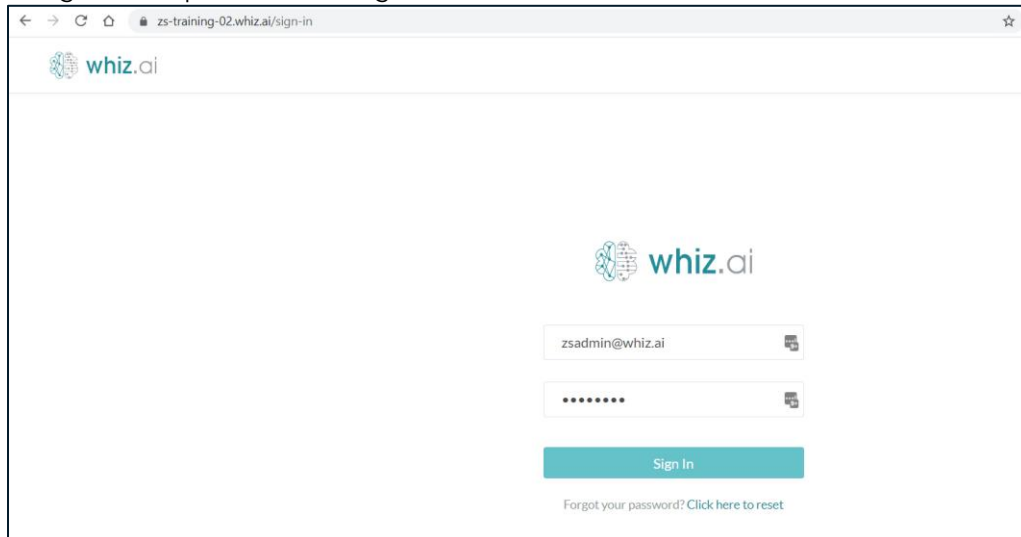


5. Navigate to /app/k8s/druid-data/airflow/dags/whizpy



Accessing WhizAI Application

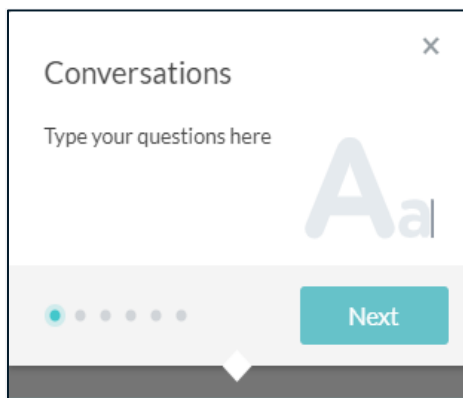
1. Navigate to `https://<zs-training-XX.whiz.ai>/`



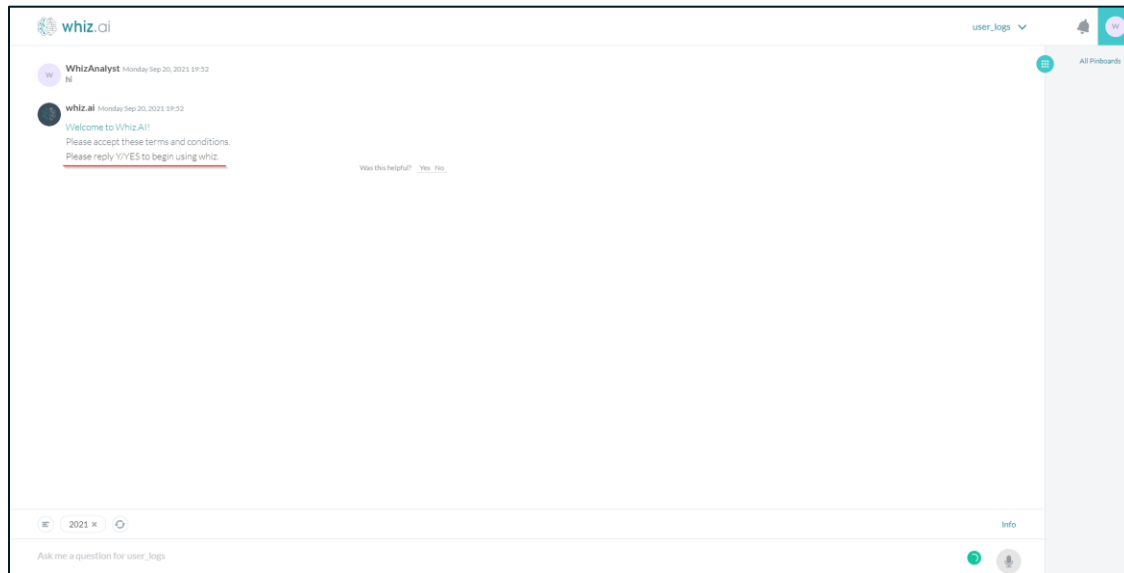
Login Details:

USER : zsadmin@whiz.ai
PASSWORD : Whiz@123

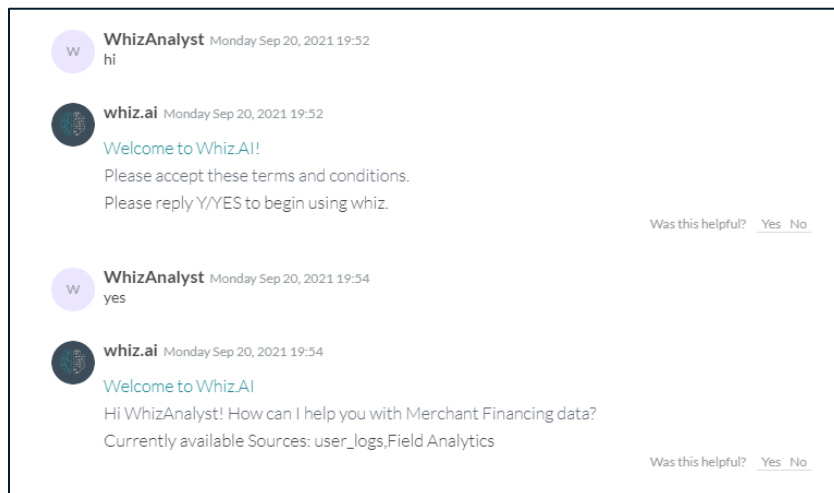
2. Complete onboarding flow



3. On Explorer Type 'Hi' , this will return response with term and conditions.



4. WhizAI will show 'Welcome to WhizAI' message



Configuring Field Analytics Model

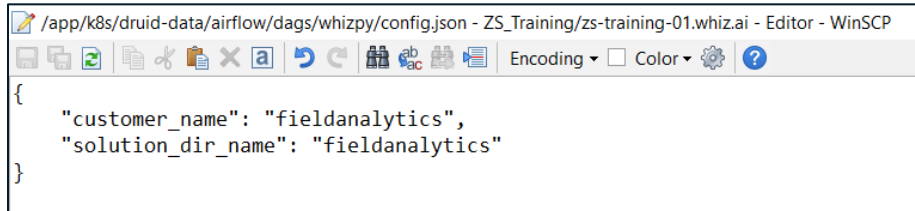
Updating config.json file

1. Using WinSCP , navigate to /app/k8s/druid-data/airflow/dags/whizpy

| /app/k8s/druid-data/airflow/dags/whizpy/ | | | | | |
|--|------|---------------------|-----------|--------|--|
| Name | Size | Changed | Rights | Owner | |
| .. | | 01-09-2021 20:55:28 | rw-rw-rw- | root | |
| test | | 01-09-2021 20:42:37 | rw-rw-rw- | root | |
| framework | | 01-09-2021 20:57:47 | rw-rw-rw- | root | |
| documentation | | 01-09-2021 20:42:37 | rw-rw-rw- | root | |
| TODO | 1 KB | 01-09-2021 20:42:37 | rw-rw-rw- | root | |
| setup.py | 1 KB | 01-09-2021 20:42:37 | rw-rw-rw- | root | |
| README.md | 2 KB | 01-09-2021 20:42:37 | rw-rw-rw- | root | |
| config.json_backup | 1 KB | 01-09-2021 20:55:33 | rw-rw-rw- | ubuntu | |
| config.json | 1 KB | 01-09-2021 20:57:47 | rw-rw-rw- | ubuntu | |

2. Open config.json in text editor ,In config.json set the following parameters

```
"customer_name": "fieldanalytics",  
"solution_dir_name": "fieldanalytics"
```



```
{  
  "customer_name": "fieldanalytics",  
  "solution_dir_name": "fieldanalytics"  
}
```

Updating environment.json file

1. Using WinSCP , Navigate to /app/k8s/druid-data/airflow/dags/whizpy/framework

| /app/k8s/druid-data/airflow/dags/whizpy/framework/ | | | | |
|--|-------|---------------------|-----------|--------|
| Name | Size | Changed | Rights | Owner |
| .. | | 01-09-2021 20:57:47 | rw-rw-rw- | root |
| solution | | 01-09-2021 20:42:37 | rw-rw-rw- | root |
| framework | | 01-09-2021 20:45:47 | rw-rw-rw- | root |
| fieldanalytics | | 01-09-2021 21:02:53 | rw-rw-rw- | ubuntu |
| docker | | 01-09-2021 20:42:37 | rw-rw-rw- | root |
| __pycache__ | | 01-09-2021 20:45:53 | rw-rw-rw- | 50000 |
| user_management_dag.py | 5 KB | 01-09-2021 20:42:37 | rw-rw-rw- | root |
| framework_dag_definition.py | 11 KB | 01-09-2021 20:42:37 | rw-rw-rw- | root |
| environment.json_backup | 12 KB | 01-09-2021 20:51:28 | rw-rw-rw- | root |
| environment.json | 17 KB | 02-09-2021 16:57:20 | rw-rw-rw- | ubuntu |

2. Open environment.json in text editor and update Destination Datasource details

```
{
  "customer_name": "fieldanalytics",
  "destination_specs": {
    "type": "druid",
    "host": "XXXXXX",
    "port": "30023",
    "database": "druid",
    "username": "druid",
    "password": "FoolishPassword",
    "overwrite": true,
    "backup_directory": "backups/destination",
    "incremental_load": false,
  }
}
```

3. For each datasource listed ensure that enable load is set as true

List of datasource :

1. sales
2. call_plan
3. marketing
4. speakerprogram
5. salesgoal
6. employee_roaster
7. field_time_territory
8. customer_alignment
9. fct_calls
10. sales_nbrx
11. sob_switch

```
"datasources": [
  {
    "dataSource": "sales",
    "enable_load": true,
    "dateFormat": "%Y-%m-%d",
    "ioConfig": {
```

4. Configure NLP service details

Update Host and Port to assigned sev

```
"nlp": {
  "enable_autoimport": false,
  "overwrite": true,
  "ue_url": "http://XXXXX:31670/updateEntities",
  "ner_url": "http://XXXXX:31670/ner",
  "entity_url": "http://XXXXX:31670/entity",
  "entity_hierarchies": "http://XXXXX:31670/nlp/entity-hierarchies",
  "ue_models": [
    "common",
    "sales"
  ],
  "ue_root_dir": "/app/models/",
  "ue_clear_flag": true,
  "ue_multi_lang": false,
  "ue_update_cache": true,
  "entity_replacement_flag": true,
  "entity_replacement": "http://XXXXX:31670/replacements/importV200?clear=True",
  "entity_replacement_file": "/opt/airflow/dags/whizpy/framework/fieldanalytics/inputs/nlp/nlp_replacements.json",
  "generate_system_synonyms": true,
  "languages": []
},
```

5. App Layer configurations

Update Host and Port to assigned sever in configuration URL and Update application database configurations to update Host and Port details

```
"app_layer": {
  "export_configuration": false,
  "enable_autoimport": false,
  "url": "http://XXXXX:30006/api/auth/sign_in",
  "username": "admin@whiz.ai",
  "password": "12345678",
  "datasource_code": "sales",
  "datasources_url": "http://XXXXX:30006/api/dataSources",
  "refreshpins_url": "http://XXXXX:30006/api/auth/sign_in",
  "entities": "http://XXXXX:30006/api/entities",
  "replace_suggested_queries": true,
  "backup_directory": "backups/suggested_queries",
  "app_layer_connections": {
    "host": "XXXXX",
    "database": "sda_ga",
    "user": "postgres",
    "password": "whizuser",
    "port": 30004
  }
},
```

6. Users section

This can be left as is, we are not enabling User Import in this training

7. Error recovery

Update Email to receive email on Pipeline Execution failure

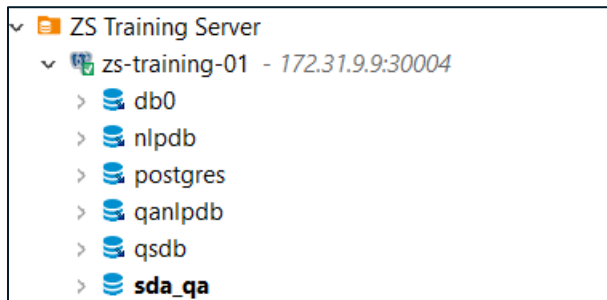


SMTP server is not configured on training servers hence you will not receive any emails on failures.

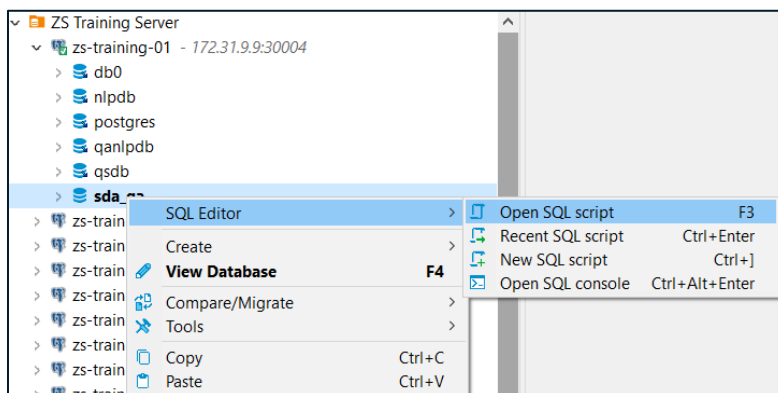
Importing Configurations from SQLs

Importing Help responses

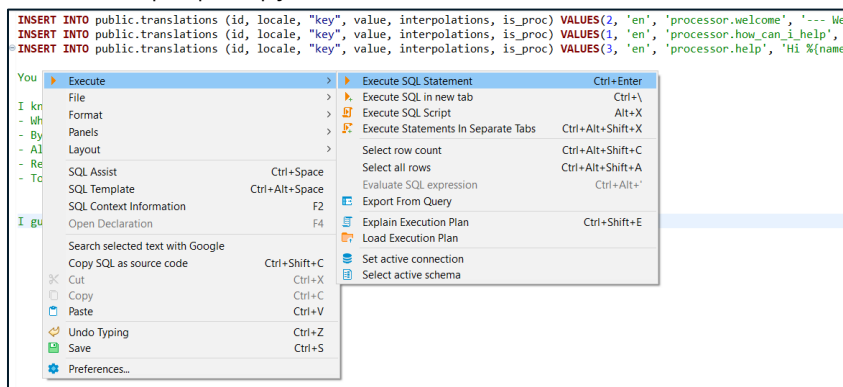
1. Login to Postgresql Database using Dbeaver



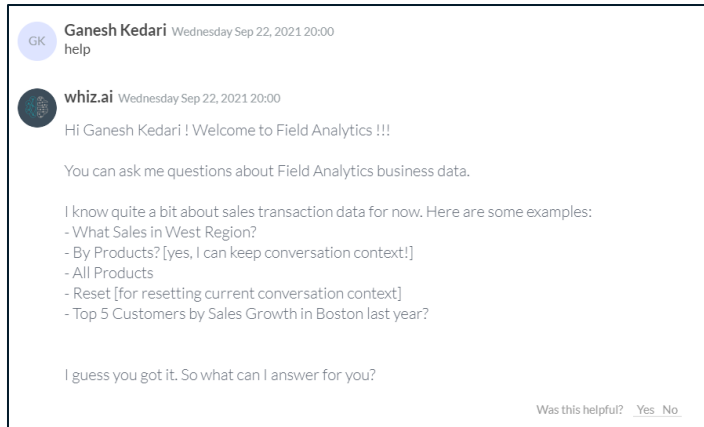
2. Launch SQL Editor



3. In WinSCP, Navigate to fieldanalytics/configurations/RailsConfigs folder and open Model_Setup.sql. Copy SQL to DBever and execute

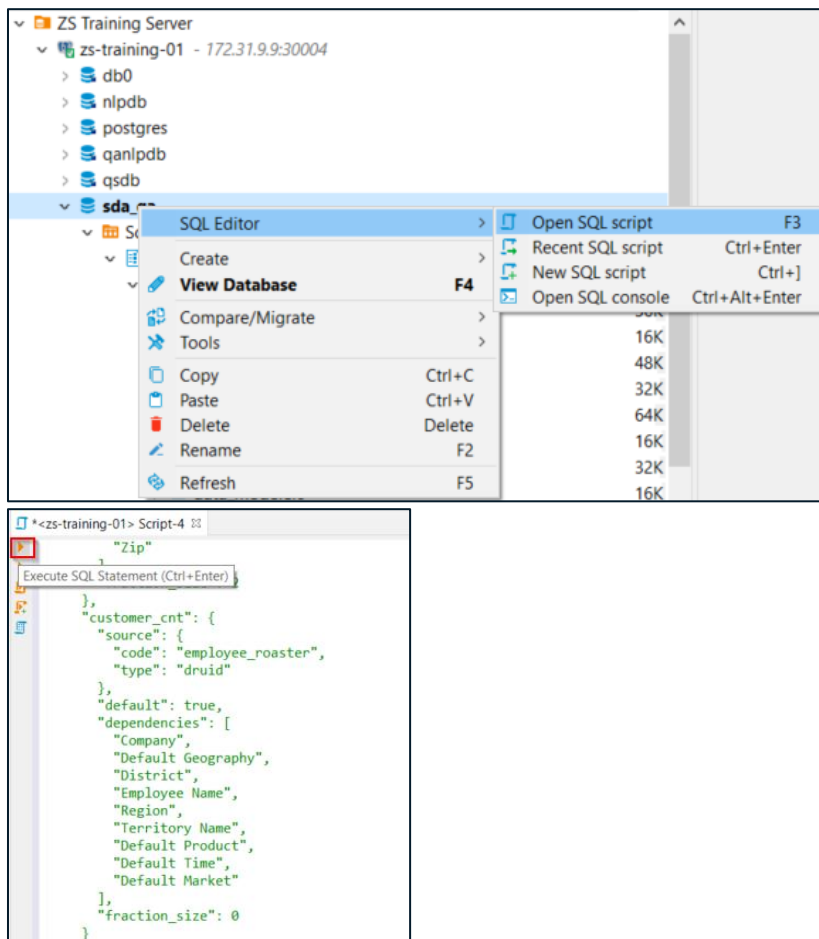


4. Navigate to WhizAI and Enter query 'Help', Configured message will be shown as response to Help command.

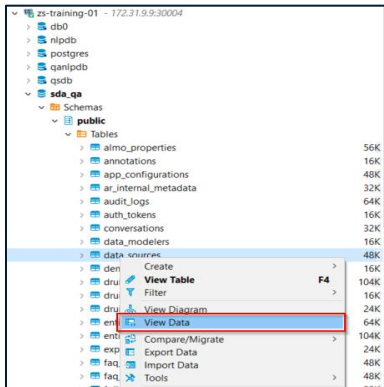


Creating Data Model

1. In WinSCP, navigate to fieldanalytics/configurations/RailsConfigs and open create_field_analytics_datasource.sql. Copy SQL to DBever and execute



2. Navigate to zs-training-01 > sda_qa > public.data_sources Table, right click on table name and select View Data

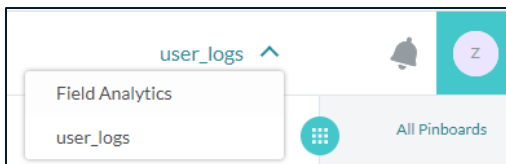


3. Notice that new datasource Sales is created in Database table

The screenshot shows the 'data_sources' table in the 'public@sdq_qa' schema. The table has columns: id, code, name, properties, created_at, updated_at, and source_type. The 'sales' table is highlighted in the list.

| id | code | name | properties | created_at | updated_at | source_type |
|----|-----------|-----------|--|-------------------------|-------------------------|-------------|
| 1 | user_logs | user_logs | [NULL] | 2021-08-27 08:38:48.975 | 2021-08-27 08:38:48.975 | druid |
| 2 | 100 | sales | Field Analytics { "metrics": { "PDOT": { "source": { "code": "sales", "2020-01-10 00:00:00.000 | 2021-08-16 10:09:41.693 | druid | |

4. Login to WhizAI and you can notice that new datasource is added as Field Analytics



Getting Started with Data Dictionary

1. Navigate to /app/k8s/druid-data/airflow/dags/whizpy/framework/fieldanalytics/configurations
2. Open DataDictionary.xlsx in Microsoft Excel

Data Dictionary Fields

1. ID :

Unique Row Identifier (Start index > 10000)

2. CLASS :

Metadata/Metrics

3. Enabled :

TRUE/FALSE. Must be TRUE if this row representing a dimension is to be used for further processing.

4. CODE :

Field name to be used as Code , this should match name from destination druid database column

5. NAME :

Name to be shown in UI

6. DESC :

Description values to be shown as Tooltip on info panel

7. HIERARCHY_CLASS :

HIERARCHY_CLASS indicates the name of the NLP csv file that will contain members of all those dimensions which have the same HIERARCHY_CLASS value. It is mandatory to be non-empty for rows that have LEVEL column set as 'Level'.

8. LEVEL :

Metadata fields : Dimensions, Level, Attribute

Dimension is a plain dimension of data that will be loaded and processed. Level and Attribute are more specialized kinds of Dimensions.

By setting a value 'Level', we want this dimension to be a part of the CSV with the name of the 'HIERARCHY_CLASS' for this row. Note that if the dimension is a Level, then HIERARCHY_CLASS cannot be empty.

Attribute value here means that this dimension is an attribute of the Parent dimension and will be included in the Attribute.csv with the value indicated in the Parent column for this row. Note that this row will still result in a distinct NLP CSV if required by the 'NLPGeneration' column.

Metrics fields :

- a. Base : Default metrics from source data
- b. Calc : Calculation Metrics configured in Rails DB
- c. API : API metrics used from Query Service

9. Datasource :

Source Database Name.

10. DestinationDatasource

Destination Database Name - Druid DB.

Typically we might have one dimension possibly a part of multiple data sources in Druid. We should not repeat the dimension in multiple lines. Instead, we should add the multiple data source names in comma separated format here, without spaces.

11. Lookup

TRUE / Blank

12. NLPGeneration

- a. BOTH : When this dimension should be included in the NLP CSVs as well as metadata/metrics.
- b. ONLY_METADATA : When this dimension should not be included in NLP CSVs but in metadata. Note that if level set as attributes, Attribute.csv will be generated but not the nlp column csv.
- c. SPECIAL_HANDLING : Similar to BOTH but additionally, this dimension may be part of a parent-child relationship with some other dimension or the NLP column type may be keyvalue (example: Region ID considered as a parent to Region Name then Region Name is marked for Special Handling and with column type as keyvalue).
- d. NONE : When this dimension should not be in NLP nor in metadata. This is typically the case when the data CSVs or customer data source has this column and the data will be imported to Druid but it is not relevant otherwise.

13. DateColumn

DateColumn / Date / Blank

- a. DateColumn: Druid datasource must have exactly one time dimension. DateColumn indicates that this dimension is that time dimension for all applicable druid datasources.
- b. Date: This dimension is not a time dimension but it is still of type date instead of a plain string.
- c. Blank: Either a plain string dimension or a metric.

14. NLPColumnType

- a. generic : No special properties or handling needed.
- b. keyvalue: This has a parent-child relationship with some other column. Example, Region Name may be marked keyvalue when there is a Region ID column that is kept as its parent.
- c. tag_process: Use this typically for multi-valued columns. You can specify a delimiter used in the data storage side (like Druid) and exported in CSVs for multi-valued data.
- d. acronym: For acronyms of specific member codes/names.

15. AcronymGroup :

This should contain comma separated list of dimension codes (D1, D2...) that are acronyms of this dimension (D0). The NLP column type of D0, D1, D2... should all be acronym. However, this AcronymGroup will only be populated for dimension D0 and be empty for D1, D2

16. Parent :

Parent Dimension Name. This name is used to indicate parent-child relationships between dimensions.

17. RelationDatasources :

A comma separated list of datasources to consider while generating relations CSV. This must be a subset of the destination datasources provided for this dimension. This dimension is considered as the single child dimension for the relation.

18. RelationParents :

A comma separated list of parent dimensions to this child dimension. Note that this information is different from the Parent column described previously. This column is only intended for creating the Relations CSV. The parent dimensions specified here should each, have their NLP csv created.

19. Exclusions :

Comma separated list of member codes from this dimension row that should not be part of the NLP CSV. Usually for removing #NA values

20. Translate :

Set TRUE/FALSE for enabling translation of NAME and all Synonym columns of the NLP CSV generated for this dimension row. See more in environment.json on setting the list of languages to translate into with English as starting point. Empty value implies no translation.

21. GenerateUserSynonyms :

If marked TRUE, corresponding sheet named after the CODE of this dimension and within <solution_dir>/inputs/nlp/UserDefinedSynonyms.xlsx will be read and user defined synonyms will be included in NLP CSV files after matching the member codes.

22. GenerateSystemSynonyms :

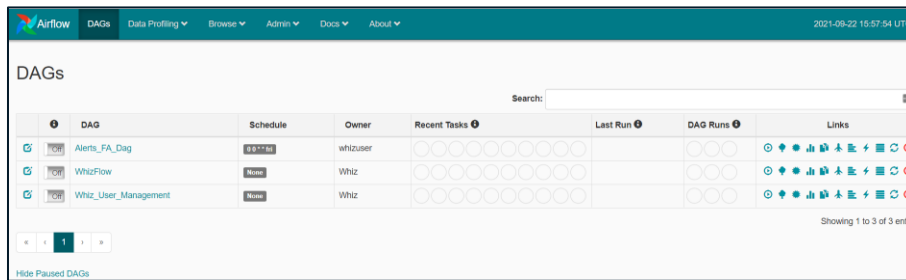
If marked TRUE, framework will generate some system synonyms for each member of the dimension listed in the NL CSV file. Note that the mapping sheet within <solution_dir>/inputs/nlp/UserDefinedSynonyms.xlsx will still be read even if user defined synonyms are not required for this dimension.

23. SYN1 - SYN14 :

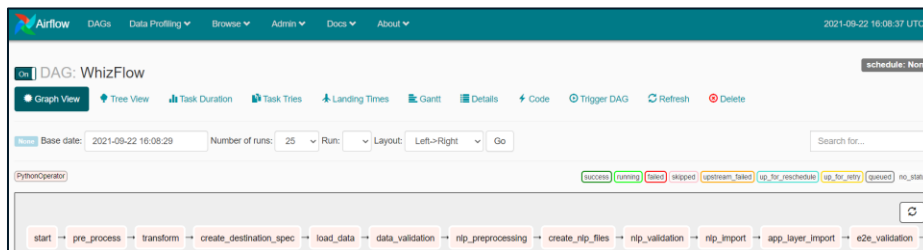
Synonyms

Executing Airflow DAG

1. Navigate to `http://<Server IP>: <PORT>/airflow/admin/`



2. Review WhizFlow DAG



3. Enable WhizFlow DAG

| | | DAG | Schedule | Owner |
|-------------------------------------|-----|----------------------|-----------|----------|
| <input type="checkbox"/> | Off | Alerts_FA_Dag | 0 0 * * * | whizuser |
| <input checked="" type="checkbox"/> | On | WhizFlow | None | Whiz |
| <input type="checkbox"/> | Off | Whiz_User_Management | None | Whiz |

4. Trigger WhizFlow DAG

