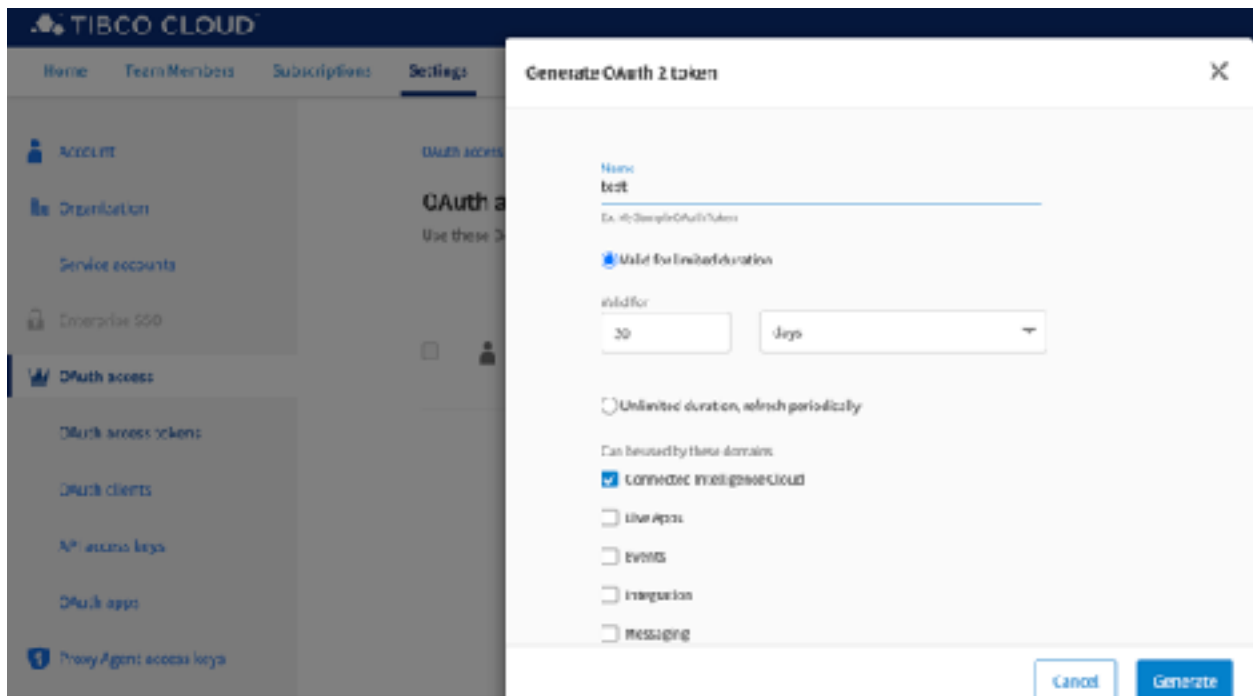


Setup	2
Tutorials	3
Tutorial 1 - Generate rule-driven synthetic data (Only single table is supported at present)	3
Tutorial 2 - Model Driven Single Table Data Generation	7
Tutorial 3 - Model Driven Multi-Tables Data Generation	9
Tutorial 4 - Model Driven Time Series Data Generation	12

Setup

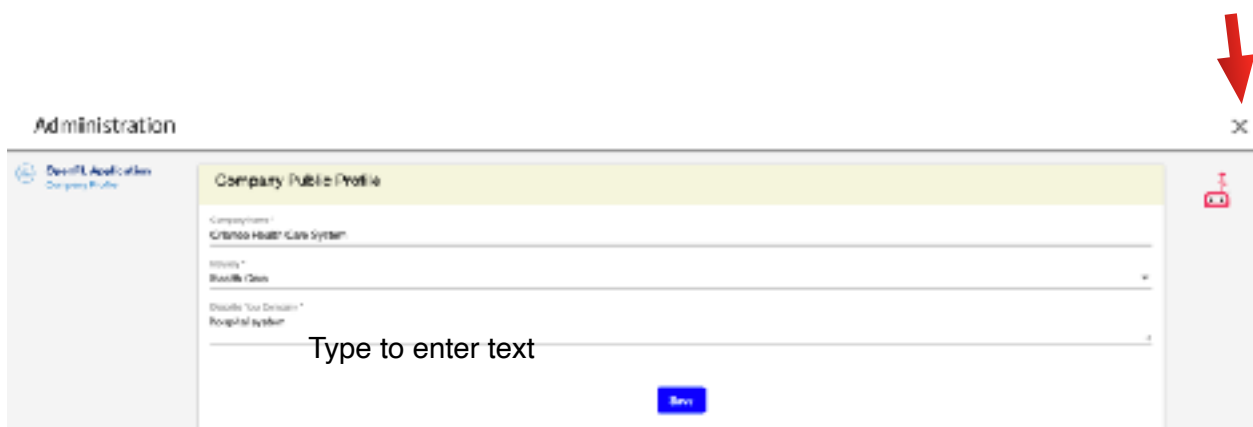
1. git clone <https://github.com/mwenyan/syntheticdata.git>
2. docker pull mweny88/project-synthetic-data-sample:1.0.0
3. docker pull mweny88/project-synthetic-data-model:1.0.0
4. Register at cloud.tibco.com if you don't have an account already
5. Generate an OAuth Access Token with access to "Connect Intelligence Cloud"



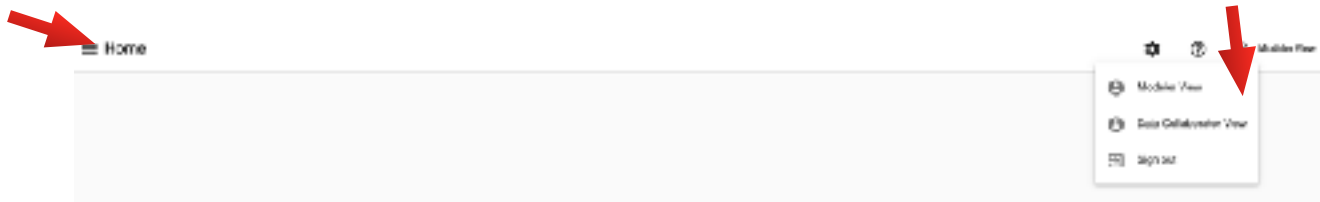
6. Go to OpenFL web console and login with your access token.

<https://n8ah4g4d0j.execute-api.us-east-2.amazonaws.com/login>

For the first time login, you will be directed to the profile page, enter your company info, and close by click on the X link



7. From the home page, you can select the view perspective, it is default to modeler view, for Synthetic Data Generation, select Data Collaborator View



8. Click on the Home menu button on the left corner to see list of available menus for Synthetic Data Generation

- **Define Dataset :**
 - Define the meta data about your dataset
- **View Datasets:**
 - View list of datasets defined
 - Create synthetic data generation tasks
 - Model driven
 - Rules driven
- **View Synthetic Data Results:**
 - View task status
 - Download task config json file
 - Use existing task as template to create new task
 - Create model-driven synthetic data generation task using trained models from a completed task

Tutorials

Tutorial 1 - Generate rule-driven synthetic data (Only single table is supported at present)

1 Define a profile single table dataset

1.1 Dataset General Info

A screenshot of the 'Define Data Set' form. The form has a progress bar at the top with four steps: 'Define Data Set' (active), 'Define Data Parameters', 'Define Data Constraints', and 'Review & Save'. The form contains three input fields: 'Dataset Identifier' with the value 'profile', 'Dataset Description' with the value 'a person's profile', and 'Select Dataset Type' with the value 'Single Table'.

1.2 Dataset Fields - Import variables from the sample file (sample_profile.csv) provided

Click on “Add New Variable” to manually add a new variable. Or you can import variables from a sample CSV File, it is expected to have a header row and at least one data row. Header name will become the variable name, and the data type is derived from sample data. Once it is imported, you can manually edit each variable to add new meta information, such as variable type, and if it is a primary key field.

For rule-driven synthetic data generation, variable name, data type, variable type and primary key fields are required.

Define Data Set | Define Data Properties | Define Data Constraints | Review & Save

List of Variables

From Sample CSV File

Add New Variable

id

firstName

lastName

address

Variable Name (letters, numbers, and _ only)

Variable Description

Select Data Type

Is Machine Learning Classifier Field

Primary Key?

Field Transformer

id

Select Sample File

Choose File | sample_profile.csv | File

Variable Name	Inferred Data Type
id	string
firstName	string
lastName	string
address	string
city	string
state	string
zipcode	integer
birthDate	datetime
gender	string
email	string

Done Cancel

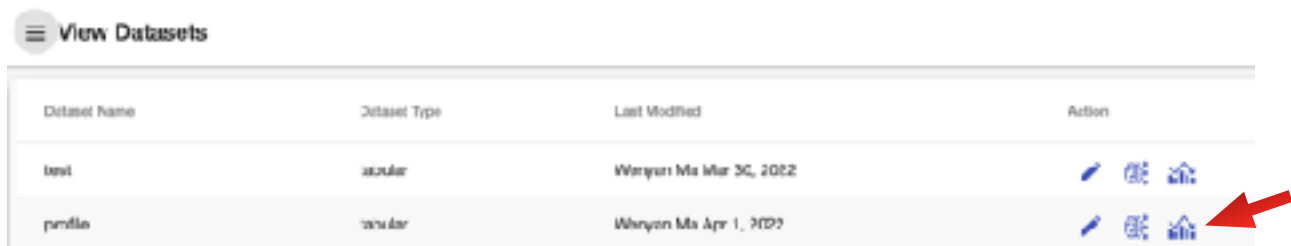
1.3 After import, update following fields







- id - check primary key field
- gender - change variable type to categorical
- zipcode - change data type to “text”, and variable type to “label”

1.4 Click Next and Next to “Review & Save” tab, save the dataset

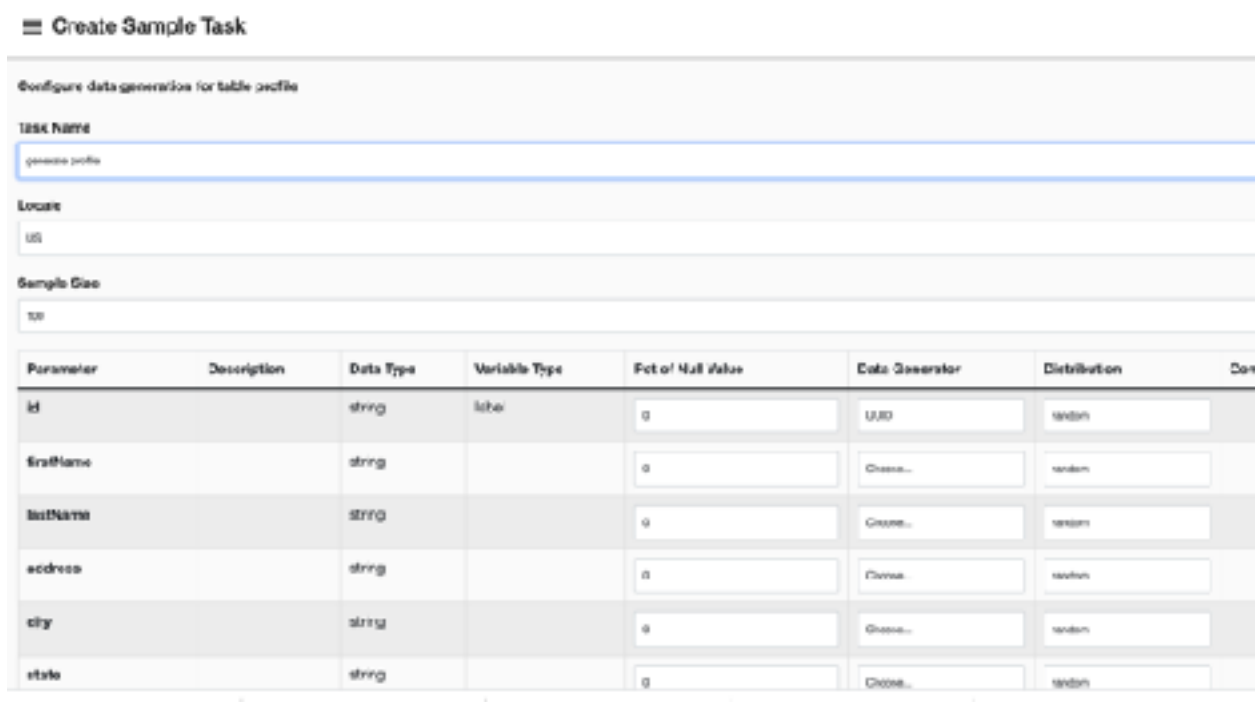
2 Create Synthetic Data Generation Task

2.1 Go to Home -> View Datasets and click on the “graph” icon



Dataset Name	Dataset Type	Last Modified	Action
test	categorical	Wenyan Ma Mar 30, 2022	  
profile	categorical	Wenyan Ma Apr 1, 2022	  

2.2 Configure task



Create Sample Task

Configure data generation for table profile

Task Name: generate profile

Location: US

Sample Size: 100

Parameter	Description	Data Type	Variable Type	Pct of Null Value	Data Generator	Distribution	Con
id		string	label	0	UUID	random	
firstName		string		0	Choose...	random	
lastName		string		0	Choose...	random	
address		string		0	Choose...	random	
city		string		0	Choose...	random	
state		string		0	Choose...	random	

update generator and distribution for each field, and configure rules if necessary

- id - default to UUID generator, keep as is
- firstName - use “First Name” generator

- lastName - use "Last Name" generator
- address - use "Street Address" generator
- city - use "City" generator
- state - use "State" generator
- zipcode - use "Postal Code" generator
- birthDate - use "Birth Date" generator
 - use random distribution
 - Configure min and max age - e.g. 18 and 70
- gener - use "Choose from a list" generator
 - for categorical values, you can configure a list of codes to use
 - Add "M", "F" codes
- email - use "Email" generator
- phone - use "Phone" generator
- employed - use "Boolean" generator
- income - use "Random Float" generator
 - use "decision rules" distribution
 - select dependent field "yearsEmployed"
 - Add rule 1
 - Operator: >= && <
 - Lower bound: 1
 - Upper bound: 10
 - Distribution: normal
 - Configure mean/std/lower/upper/decimals: 35000/1/25000/50000/2
 - Add rule 2
 - Operator: >= && <
 - Lower bound: 10
 - Upper bound: 20
 - Distribution: manual
 - Configure manual rules
 - 50000 - 100000, pct: 100, decimals: 2
 - Add rule 3
 - Operator: >=
 - Value: 20
 - Distribution: manual
 - Configure manual rules
 - 100000 - 125000, pct: 50, decimals: 2
 - 125000 - 150000, pct: 25, decimals: 2
 - 150000 - 500000, pct: 25, decimals: 2
- yearEmployed - use "Random Integer" generator
 - Change to "normal" distribution
 - configure mean/std/min/max values: e.g. 5/1/1/40

2.3 Save or Export

- Save for later use
- Export the task json file

2.4 Run the task

Task can be run locally or automatically. To run automatically, an agent (opendata-agent) needs to run on your local machine, for this tutorial, we will run it manually.

- Run the task manually
 - Create a working directory
 - Copy the task ison file to this folder
 - `docker run --rm -v <replace with your working dir>:/output mweny88/project-synthetic-data-sample:1.0.0 ./run.sh <replace your task_id uuid value>`
 - Sample data will be created in your working directory under task_id sub-directory

Tutorial 2 - Model Driven Single Table Data Generation

1 Define a diabetes single table dataset

1.1 Dataset General Info (see Tutorial 1)

1.2 Dataset Fields - Import variables from the sample file (diabetes.csv) provided

1.3 After import, modify following fields

- Outcome - check “Is Machine Learning Classifier Field”

1.4 click Next to “Define Data Constraints” Step

- Select diabetes table
- Select “Value Constraints” tab
- Add two rounding constraints, click “Apply Changes for diabetes” button

Define Data Set | Define Data Properties | Define Data Constraints | Review & Save

Select Table: diabetes

Unique Constraints | Value Constraints | Formula Columns

Specify the value constraints of a numeric or datetime column.

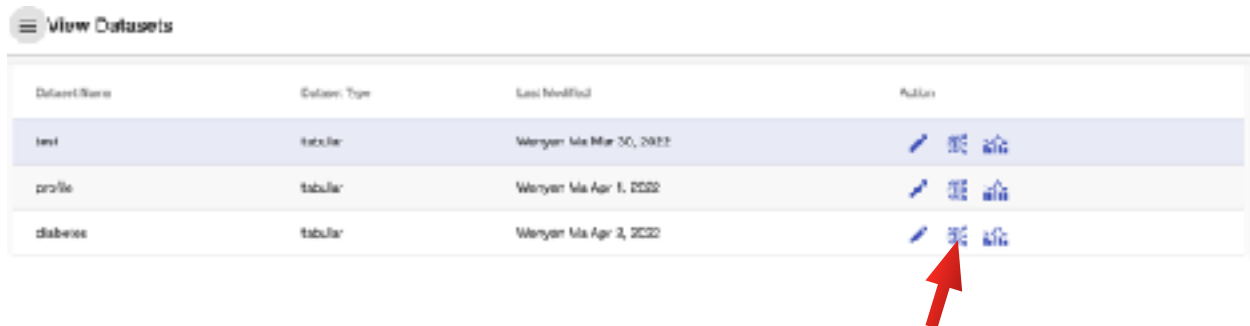
Action	Constraint Name	Column	Constraint Type	Lower # of Decimals	High
	AGE ROUNDING	AGE	rounding	1	
	Blood Pressure Rounding	BloodPressureFunction	rounding	2	










[Apply Changes for diabetes](#)

1.5 Click “Next” to review step, and Save

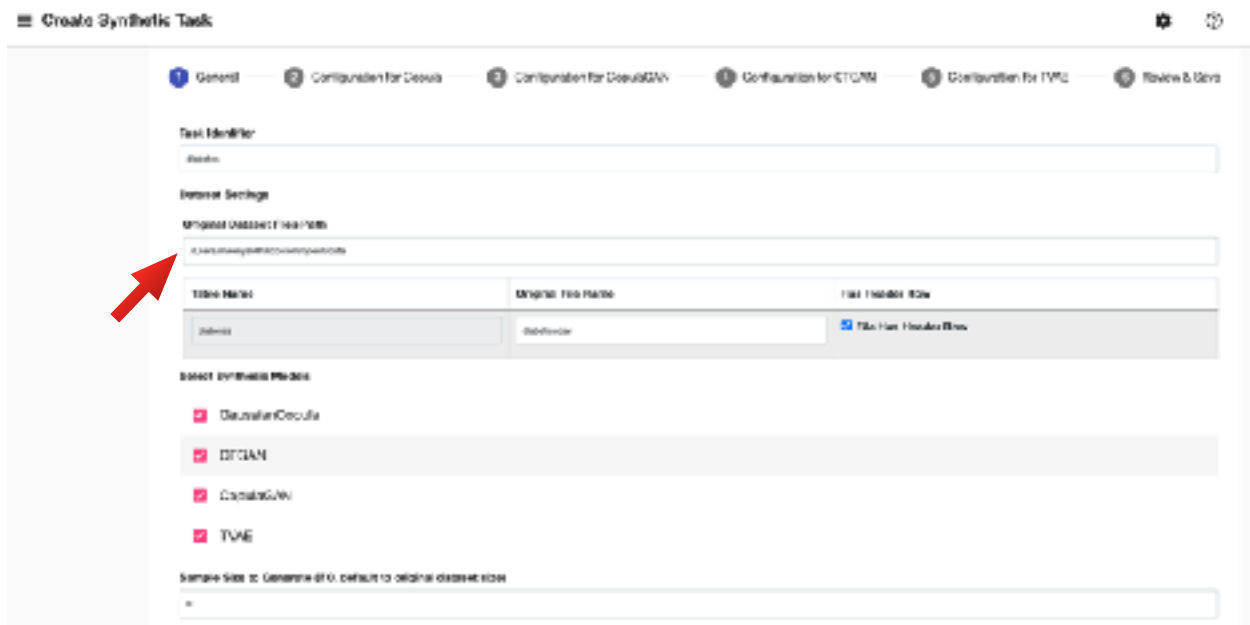
2 Create Synthetic Data Generation Task

2.1 Go to Home -> View Datasets and click on the “brain” icon



Dataset Name	Dataset Type	Last Modified	Actions
test	tabular	Wengyan Ma Mar 30, 2022	  
profile	tabular	Wengyan Ma Apr 1, 2022	  
diabetes	tabular	Wengyan Ma Apr 2, 2022	  

2.2 Create Synthetic Task



Create Synthetic Task

1 General 2 Configuration for Deepa 3 Configuration for DeepaGAN 4 Configuration for CTGAN 5 Configuration for TVM 6 Review & Send

Task Identifier

diabetes

Dataset Settings

Original Dataset Files Path

/mnt/manydata/workingpath000

Table Name **Original File Name** **File Name Suffix**

diabetes	diabetes.csv	
----------	--------------	--

Select Synthetic Models

☒ DataGenCecilia

☒ DTCAN

☒ DeepaGAN

☒ TVM

Sample Size to Generate (0, default to original dataset size)

10

- Update “Original Dataset Files Path” to the directory where diabetes.csv is located
- Click Next -> Next -> Next -> Next to Review Step and “Submit Task”

2.3 Run the task

for this tutorial, we will run it using the agent so we can view the result in the console.

- Create a working directory
- Create config.properties file with following entries

region = US

oauthToken = your oath token

`slimage` = mweny88/project-synthetic-data-sample:1.0.0
`mlimage` = mweny88/project-synthetic-data-model:1.0.0
`workDir` = your working dir

- Start up the agent
./start-agent.sh

the agent will pull all tasks with “SUBMITTED” status, and run each sequentially, the results will be available in your working directory under <task_id> subdirectory. You can also view the metrics and histograms in the web console



Tutorial 3 - Model Driven Multi-Tables Data Generation

In this tutorial, we will learn and generate from 3 mobile usage tables: Users, Sessions and Transaction. One user can have multiple sessions, and within each session, there can be multiple transactions.

1 Define a diabetes single table dataset

1.1 Dataset General Info, select “Relation Tables” as Dataset Type

Define Original Dataset

1 Define Table Set 2 Define Data Properties 3 Define Table Relationships 4 Define Data Constraints 5 Review & Generate

Dataset Identifier

diabetes

Dataset Description

mobile usage data

Select Dataset Type

Relation Tables

Next

1.2 Dataset Tables and Fields

Add 3 tables: **users**, **sessions** and **transactions**

The image shows two screenshots of the 'Define Original Dataset' interface. The top screenshot shows the 'Select or Add New Table' dropdown menu with 'New Table' selected. The bottom screenshot shows the 'Table Name' input field with 'new_table' entered, and the 'Add' and 'Save Table' buttons highlighted with red arrows.

import from users.csv, sessions.csv and transactions.csv to create variables for each table respectively. modify each table as follows, **Remember, click on “Save Table” button when you are done with each table**

users table:

- user_id : check “Is Primary Key”
- country: change variable type to “categorical”
- gender: change variable type to “categorical”

sessions table:

- session_id: check primary key
- device: change variable type to categorical
- os - change variable type to categorical

transactions table:

- transaction_id: check primary key
- timestamp: update Datetime Format to %Y-%m-%dT%H:%M
- amount: change data type to float

1.3 Define Table Relationships

1.3.1 users - sessions one to many relationship

1.3.2 sessions - transaction one to many relationship

Define Original Dataset

Define Data Set Define Data Properties Define Table Relationship Define Data Constraints

List of Relationships

-- Add New Relationship --

Select Table

transactions

Select Parent Table

users

Select Foreign Key Field

user_id

Add

Define Original Dataset

Define Data Set Define Data Properties Define Table Relationship Define Data Constraints Review & Save

List of Relationships

-- Add New Relationship --

users -> sessions
PK: user_id

transactions -> sessions
FK: transaction_id

Select Table

transactions

Select Parent Table

sessions

Select Foreign Key Field

session_id

Update Remove

1.4 Review and save the dataset

2 Create Synthetic Data Generation Task

2.1 Go to Home -> View Datasets and click on the “brain” icon

2.2 Specify dataset file path and file names

Create Synthetic Task

General Review & Save

Task Identifier

task

Dataset Settings

Original Dataset File Path

C:\Users\user\Documents\dataset

Table Name	Original File Name	File Header Row
users	users.csv	<input checked="" type="checkbox"/> File Header Row
sessions	sessions.csv	<input checked="" type="checkbox"/> File Header Row
transactions	transactions.csv	<input checked="" type="checkbox"/> File Header Row

Select Synthesizer Model

☒ HMM (GaussianCopula)

Sample Size to Generate (0 is default to original dataset size)

1

2.3 Review and Submit task

3. Run the task

See tutorial 2

Tutorial 4 - Model Driven Time Series Data Generation

In this tutorial, we will learn and generate time series data

1 Define a diabetes single table dataset

1.1 Dataset General Info, select “Time Series” as Dataset Type

Define Original Dataset

Define Data Set | Define Data Properties | Review & Save

Dataset Name: diabetes

Dataset Description: Diabetes Data

Select Dataset Type: Time Series

Next

1.2 Define data fields, import from stocks.csv file

- Symbol: change variable type to “categorical”, and select “Entity Column”
- Date: select “Sequence Index Column”
- Open: select “Data Column”
- Close: select “Data Column”
- Volume: select “Data Column”
- MarketCap: select “Context Column”
- Sector: select “Context Column”
- Industry: select “Context Column”

1.3 Review and Save

2. Create Synthetic Task

2.1 Go to Home -> View Datasets and click on the “brain” icon

2.2 Specify dataset file path and file names

Create Synthetic Task

1 General 2 Configuration for Timeseries 3 Review & Save

Task Identifier

Dataset Catalogs

Original Dataset Files Path

Table Name Original File Name New Identifier Name

entity dataset dataset [File as Header Row](#)

Select Synthetic Models

☒ FRR (GaussianCopula)

[Back](#)

2.3 Configure parameters for time series task

You can either define regex for Entity Column - Symbol, or Specify specific list of values for Symbol and its associated context columns, we will use regex in this tutorial

2.3.1 check “Use Regular Expression to Generate Entity Column”

2.3.2 use [A-Z]{2,4} as the regex

2.3.3 generate 1 records for each sequence

1 General 2 Configuration for Timeseries 3 Review & Save

☒ Use Regular Expression to Generate Entity Column

Regular Expressions

Entity Column	Regular Expression
Symbol	[A-Z]{2,4}

No of Records to Generate for Each Sequence (Optional, sampled from original data)

1

[Back](#) [Next](#)

2.3.4 Review and Submit task

3. Run the task

See tutorial 2