**文档信息**

|  |  |  |  |
| --- | --- | --- | --- |
| 文档版本编号: | V1.0.0 | 文档版本日期: | 2023年1月18日 |
| 起草人: | 龚长华 | 起草日期: | 2023年1月18日 |

**版本记录**

| 版本编号 | 版本日期 | 创建/修改 | 说明 |
| --- | --- | --- | --- |
| V1.0.0 | 2023-01-18 | 龚长华 | 初稿 |
| V1.0.1 | 2023-01-30 | 龚长华 | DataSource/RuntimeBatchRequest  添加\_data\_dt变量 |
| V1.1.0 | 2023-01-31 | 龚长华 | 增加custom\_actions.py打通短信通路 |
| V1.2.0 | 2023-01-31 | 龚长华 | 调整7.4适配GE工作空间 |

# 文档目的

为数据开发人员、数据质量管理人员执行质量管理行为提供指导性说明。

# 背景

当前应用在生产环境的质量管理组件为Apache Griffin，其存在度量不丰富、与其他组件集成困难、社区不活跃等问题。经过调研，选用Great Expectations（以下简称GE）作为大数据平台的通用质量管理套件。

# 目标

1. 安装GE质量管理组件；
2. GE与Datahub治理中心集成；
3. GE与Airflow调度中心集成；
4. 使用GE完成Iceberg表度量，并将结果推送至Datahub。

# 范围

GE套件仅适用批处理环境（包括微批），不适用于实时处理环境。

# 安装说明

## 环境说明

|  |  |  |  |
| --- | --- | --- | --- |
| 环境 | 版本 | 部署路径 | 服务器IP |
| 研发 | 0.15.28 | /hdp\_data/gch/venv/great\_expectations | 88.88.16.189 |
| 生产 | 0.15.28 | /hdp\_data/gch/venv/great\_expectations | 10.50.253.6 |

表 1 - GE部署环境说明

## 前提条件

Python需要3.7~3.10版本。当前生产环境版本为3.9.14，可通过如下命令查看Python版本。

python --version

Python 3.9.14

## 安装步骤

1. 在研发环境/hdp\_data/gch/venv/目录下，创建Python虚拟环境

python -m venv great\_expectations

1. 激活虚拟环境，并安装GE

source /hdp\_data/gch/venv/great\_expectations/bin/activate

pip install great\_expectations

pip install pyspark==3.0.1

注：pyspark为通过spark连接iceberg数据源，版本号必须为3.0.1。

1. 验证安装

great\_expectations --version

great\_expectations, version 0.15.28

至此，GE安装成功。

1. 使用附录中configurator\_dev.py文件替换lib目录下configurator.py文件，路径：

/hdp\_data/gch/venv/great\_expectations/lib/python3.9/site-packages/great\_expectations/checkpoint/configurator.py

1. 将custom\_actions.py文件，放在路径下：

/hdp\_data/gch/venv/great\_expectations/great\_expectations/plugins

1. 将虚拟环境整体打包，复制到生产环境并解压，注意保持部署路径一致，否则导致虚拟环境不可用。

# 升级说明

1. 在[5.3](#_安装步骤)安装步骤基础上，升级GE。

source /hdp\_data/gch/venv/great\_expectations/bin/activate

pip install --upgrade great\_expectations==0.15.43

1. 升级后，注意比较configurator\_dev.py文件与

/hdp\_data/gch/venv/great\_expectations/lib/python3.9/site-packages/great\_expectations/checkpoint/configurator.py文件差异，无问题再行替换。

1. 将custom\_actions.py文件，放在路径下：

/hdp\_data/gch/venv/great\_expectations/great\_expectations/plugins

验证安装成功。

great\_expectations --version

此处应输出升级后的版本号，0.15.43.

1. 将虚拟环境整体打包，复制到生产环境并解压，注意保持部署路径一致，则导致虚拟环境不可用。

# 使用说明

## 初始化GE工作空间

为GE创建工作空间，GE在工作空间存储和读取项目所需的各项配置。在激活虚拟环境后，使用如下命令初始化GE环境。

great\_expectations init

执行完毕后，会生成对应的目录。

(great\_expectations) [root@namenode1 great\_expectations]# pwd

/hdp\_data/gch/venv/great\_expectations/great\_expectations

(great\_expectations) [root@namenode1 great\_expectations]# ls

checkpoints expectations great\_expectations.yml plugins profilers uncommitted

* great\_expectations.yml为顶层配置文件，目前尚未配置
* checkpoints为checkpoint目录，保存checkpoint配置
* expectations为expectation目录，保存度量标准
* plugins为插件目录，可扩展插件保存此处
* profilers为profiler目录，目前尚未使用到
* uncommitted为文档存储目录，存储经过渲染易读的度量报告，可下载到本地使用浏览器打开index.html浏览报告

## 生成度量依据（标准）

初始化GE工作空间后，借助Data Assistant结合已存数据生成度量标准。

import os

import pyspark

from great\_expectations.core import ExpectationSuite

from great\_expectations.core.batch import RuntimeBatchRequest

from great\_expectations.core.util import get\_or\_create\_spark\_application

from ruamel import yaml

import great\_expectations as ge

from great\_expectations.data\_context.data\_context import DataContext

os.environ["PYSPARK\_PYTHON"] = "/usr/local/bin/python3"

os.environ["PYSPARK\_DRIVER\_PYTHON"] = "/usr/local/bin/python3"

# Settings

# Table name to profile

table\_name = "hadoop\_catalog.sdm.s\_tft\_ups\_dc\_trip\_order"

data\_dt = "20230116" # 如果是非分区表，则置为空串

# Columns to profile

include\_column\_names = ["update\_time", "coupon\_use\_status", "pay\_type"]

# datahub\_schema and datahub\_dataset fields to emit to datahub

datahub\_schema = table\_name.split(".")[1]

datahub\_dataset = app\_name = table\_name.split(".")[2]

# If Spark type is decimal, raise errors when computing mean. In details, refer to https://github.com/great-expectations/great\_expectations/issues/6393

expectation\_suite\_name = table\_name.replace(".", "\_")

checkpoint\_name = table\_name.replace(".", "\_")

# Construct a Spark Dataframe

spark\_conf = {

"spark.sql.extensions": "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions",

"spark.sql.catalog.hadoop\_catalog": "org.apache.iceberg.spark.SparkCatalog",

"spark.sql.catalog.hadoop\_catalog.type": "hadoop",

"spark.sql.debug.maxToStringFields": "100",

"spark.sql.catalog.hadoop\_catalog.warehouse": "hdfs:///hadoop\_catalog",

"spark.sql.sources.partitionOverwriteMode": "dynamic",

"spark.sql.adaptive.enabled": "true",

"spark.sql.adaptive.coalescePartitions.enabled": "true",

"spark.sql.adaptive.localShuffleReader.enabled": "true",

"spark.sql.adaptive.advisoryPartitionSizeInBytes": "128M",

"spark.sql.adaptive.coalescePartitions.minPartitionNum": "1",

"spark.sql.shuffle.partitions": "200",

"spark.jars": "hdfs:///spark\_tool/jars/iceberg-spark3-runtime-0.11.1.jar",

"spark.app.name": app\_name + "\_ge",

"spark.master": "spark://namenode1:7077,namenode2:7077",

"spark.cores.max": "1",

"spark.executor.memory": "2G",

"spark.executor.cores": "1"

}

spark\_session: pyspark.sql.session.SparkSession = (

get\_or\_create\_spark\_application(spark\_conf, True)

)

# RuntimeBatchRequest with batch\_data as Spark Dataframe

# noinspection SqlDialectInspection,SqlNoDataSourceInspection

sql = "select {} from {} where data\_dt = '{}'".format(",".join(include\_column\_names), table\_name,

data\_dt) if data\_dt else \

"select \* from {}".format(",".join(include\_column\_names), table\_name)

df: pyspark.sql.dataframe.DataFrame = spark\_session.sql(sql)

# Prerequisites: Init a ge context via great\_expectations init

context: DataContext = ge.get\_context()

yaml = yaml.YAML(typ="safe")

# Create a expectation suite by context

context.create\_expectation\_suite(

expectation\_suite\_name=expectation\_suite\_name,

overwrite\_existing=True

)

suite: ExpectationSuite = context.get\_expectation\_suite(

expectation\_suite\_name=expectation\_suite\_name

)

# Create a datasource

#########################################

# Ensure all datasource using force\_reuse\_spark\_context option which is true, otherwise error `'NoneType' object has no attribute 'setCallSite'` occurs,

# refer to https://github.com/great-expectations/great\_expectations/issues/4812

#########################################

datasource\_yaml = f"""

name: spark\_iceberg\_datasource

class\_name: Datasource

module\_name: great\_expectations.datasource

execution\_engine:

module\_name: great\_expectations.execution\_engine

class\_name: SparkDFExecutionEngine

force\_reuse\_spark\_context: true

data\_connectors:

my\_runtime\_data\_connector:

class\_name: RuntimeDataConnector

batch\_identifiers:

- datahub\_schema

- datahub\_dataset

- datahub\_datadt

"""

context.add\_datasource(\*\*yaml.load(datasource\_yaml))

# Datasource spark\_iceberg\_datasource has been added ago

runtime\_batch\_request = RuntimeBatchRequest(

datasource\_name="spark\_iceberg\_datasource",

data\_connector\_name="my\_runtime\_data\_connector",

data\_asset\_name=table\_name,

runtime\_parameters={"batch\_data": df},

batch\_identifiers={

"datahub\_schema": datahub\_schema,

"datahub\_dataset": datahub\_dataset,

"datahub\_datadt": data\_dt,

},

)

validator = context.get\_validator(

batch\_request=runtime\_batch\_request,

expectation\_suite\_name=expectation\_suite\_name

)

# Sample some rows

column\_names = [f'"{column\_name}"' for column\_name in validator.columns()]

print(f"Columns: {', '.join(column\_names)}.")

print(validator.head(n\_rows=5, fetch\_all=False))

# Create a expectation suite

expectation\_suite = context.create\_expectation\_suite(

expectation\_suite\_name=expectation\_suite\_name, overwrite\_existing=True

)

# Profile

data\_assistant\_result = context.assistants.onboarding.run(

batch\_request=runtime\_batch\_request,

include\_column\_names=include\_column\_names

)

validator.expectation\_suite = data\_assistant\_result.get\_expectation\_suite(

expectation\_suite\_name=expectation\_suite\_name

)

# Save the expectation suite

validator.save\_expectation\_suite(discard\_failed\_expectations=False)

checkpoint\_yaml\_config = f"""

name: {checkpoint\_name}

config\_version: 1.0

class\_name: SimpleCheckpoint

run\_name\_template: '%Y%m%d-%H%M%S-{datahub\_dataset}'

expectation\_suite\_name: {expectation\_suite\_name}

action\_list:

- name: datahub\_action

action:

module\_name: datahub.integrations.great\_expectations.action

class\_name: DataHubValidationAction

server\_url: http://10.50.253.9:8080

"""

context.add\_checkpoint(\*\*yaml.load(checkpoint\_yaml\_config))

上述脚本借助Data Assistant创建度量标准，并生成expectation和checkpoint。

标红部分为需要修改部分，生产环境中的运行路径：

/hdp\_data/gch/venv/great\_expectations/prepare/create\_expectation\_and\_checkpoint.py

注意：当前GE不支持类型为Decimal的数据类型，在配置include\_column\_names时需将Decimal类型字段剔除，

参考https://github.com/great-expectations/great\_expectations/issues/6393

按需配置后，执行create\_expectation\_and\_checkpoint.py脚本。

source bin/activate

cd prepare/

python create\_expectation\_and\_checkpoint.py

最后输出度量依据。

(great\_expectations) [root@namenode1 prepare]# python create\_expectation\_and\_checkpoint.py

23/01/17 15:36:05 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Setting default log level to "WARN".

To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).

23/01/17 15:36:06 WARN util.Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.

/hdp\_data/gch/venv/great\_expectations/lib/python3.9/site-packages/pyspark/sql/context.py:75: DeprecationWarning: Deprecated in 3.0.0. Use SparkSession.builder.getOrCreate() instead.

warnings.warn(

Calculating Metrics: 50%|██████████████████████████████████████████████████████████████████▌ Calculating Metrics: 50%|██████████████████████████████████████████████████████████████████▌ Calculating Metrics: 100%|█████████████████████████████████████████████████████████████████████████████████Calculating Metrics: 100%|█████████████████████████████████████████████████████████████████████████████████Calculating Metrics: 100%|█████████████████████████████████████████████████████████████████████████████████Calculating Metrics: 100%|█████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████| 2/2 [00:00<00:00, 24.67it/s]

Columns: "update\_time", "coupon\_use\_status", "pay\_type".

Calculating Metrics: 100%|█████████████████████████████████████████████████████████████████████████████████]Calculating Metrics: 100%|█████████████████████████████████████████████████████████████████████████████████Calculating Metrics: 100%|█████████████████████████████████████████████████████████████████████████████████Calculating Metrics: 100%|█████████████████████████████████████████████████████████████████████████████████Calculating Metrics: 100%|█████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████| 1/1 [00:05<00:00, 5.88s/it]

update\_time coupon\_use\_status pay\_type

0 2023-01-16 19:15:09 1 TFT\_PAY

1 2023-01-16 18:55:40 1 CCB\_DCEP\_WALLET

2 2023-01-16 18:17:43 1 CCB\_DCEP\_WALLET

3 2023-01-16 18:55:59 1 CCB\_DCEP\_WALLET

4 2023-01-16 22:50:44 1 CCB\_DCEP\_WALLET

Generating Expectations: 0%| | 0/8 [00:00<?, ?it/s]

Profiling Dataset: 0%| | 0/1 [00:00<?, ?it/s]

Calculating Metrics: 50%|█████████████████████████████████████

.......

Generating Expectations: 100%|███████████████████████████████████████████████████████████████████████████████Generating Expectations: 100%|███████████████████████████████████████████████████████████████████████████████ ██████████████████████████████████████████████████| 8/8 [00:23<00:00, 2.93s/it]

一般来说，使用Data Assistant生成的度量标准，基本不符合使用要求，需优化或补充。

## 优化度量依据（标准）

### 优化前的度量标准

在优化度量标准前，必须了解GE的度量标准有哪些，官方链接：

<https://greatexpectations.io/expectations/>。进入expectations目录，修改度量标准。

(great\_expectations) [root@namenode1 expectations]# pwd

/hdp\_data/gch/venv/great\_expectations/great\_expectations/expectations

(great\_expectations) [root@namenode1 expectations]# ls

hadoop\_catalog\_sdm\_s\_tft\_tsm\_t\_trade.json hadoop\_catalog\_sdm\_s\_tft\_ups\_dc\_trip\_order.json

# hadoop\_catalog\_sdm\_s\_tft\_ups\_dc\_trip\_order.json

{

"data\_asset\_type": null,

"expectation\_suite\_name": "hadoop\_catalog\_sdm\_s\_tft\_ups\_dc\_trip\_order",

"expectations": [

{

"expectation\_type": "expect\_table\_row\_count\_to\_be\_between",

"kwargs": {

"max\_value": 17980,

"min\_value": 17980

}, # 期望数据集的最小记录数和最大记录数，需要调整

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {},

"metric\_dependencies": null,

"metric\_name": "table.row\_count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_table\_columns\_to\_match\_set",

"kwargs": {

"column\_set": [

"coupon\_use\_status",

"pay\_type",

"update\_time"

],

"exact\_match": null

},

"meta": {

"profiler\_details": {

"success\_ratio": 1.0

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_not\_be\_null",

"kwargs": {

"column": "update\_time"

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "update\_time"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.nonnull.unexpected\_count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_not\_be\_null",

"kwargs": {

"column": "coupon\_use\_status"

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.nonnull.unexpected\_count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_not\_be\_null",

"kwargs": {

"column": "pay\_type"

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.nonnull.unexpected\_count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_min\_to\_be\_between", # 该度量删除

"kwargs": {

"column": "update\_time",

"max\_value": "2023-01-16T00:00:40",

"min\_value": "2023-01-16T00:00:40",

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "update\_time"

},

"metric\_dependencies": null,

"metric\_name": "column.min",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_max\_to\_be\_between", # 该度量删除

"kwargs": {

"column": "update\_time",

"max\_value": "2023-01-16T23:56:43",

"min\_value": "2023-01-16T23:56:43",

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "update\_time"

},

"metric\_dependencies": null,

"metric\_name": "column.max",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_be\_between", # 该度量删除

"kwargs": {

"column": "update\_time",

"max\_value": "2023-01-16T23:56:43",

"min\_value": "2023-01-16T00:00:40",

"mostly": 1.0,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"column\_max\_values\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "update\_time"

},

"metric\_dependencies": null,

"metric\_name": "column.max",

"metric\_value\_kwargs": null

},

"num\_batches": 1

},

"column\_min\_values\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "update\_time"

},

"metric\_dependencies": null,

"metric\_name": "column.min",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

}

},

{

"expectation\_type": "expect\_column\_value\_lengths\_to\_be\_between",

"kwargs": {

"column": "coupon\_use\_status",

"max\_value": 1,

"min\_value": 1,

"mostly": 1.0,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"column\_max\_length\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.length.max",

"metric\_value\_kwargs": null

},

"num\_batches": 1

},

"column\_min\_length\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.length.min",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_match\_regex",

"kwargs": {

"column": "coupon\_use\_status",

"mostly": 1.0,

"regex": "(?:[A-Fa-f0-9]){0,4}(?: ?:? ?(?:[A-Fa-f0-9]){0,4}){0,7}"

},

"meta": {

"profiler\_details": {

"evaluated\_regexes": {

"(?:25[0-5]|2[0-4]\\d|[01]\\d{2}|\\d{1,2})(?:.(?:25[0-5]|2[0-4]\\d|[01]\\d{2}|\\d{1,2})){3}": 0.0,

"(?:[A-Fa-f0-9]){0,4}(?: ?:? ?(?:[A-Fa-f0-9]){0,4}){0,7}": 1.0,

"-?\\d+": 1.0,

"-?\\d+(?:\\.\\d\*)?": 1.0,

"<\\/?(?:p|a|b|img)(?: \\/)?>": 0.0,

"[A-Za-z0-9\\.,;:!?()\\\"'%\\-]+": 1.0,

"\\b[0-9a-fA-F]{8}\\b-[0-9a-fA-F]{4}-[0-5][0-9a-fA-F]{3}-[089ab][0-9a-fA-F]{3}-\\b[0-9a-fA-F]{12}\\b ": 0.0,

"\\d+": 1.0,

"\\s+$": 0.0,

"^\\s+": 0.0,

"https?:\\/\\/(?:www\\.)?[-a-zA-Z0-9@:%.\_\\+~#=]{2,255}\\.[a-z]{2,6}\\b(?:[-a-zA-Z0-9@:%\_\\+.~#()?&//=]\*)": 0.0

},

"success\_ratio": 1.0

}

}

},

{

"expectation\_type": "expect\_column\_value\_lengths\_to\_be\_between",

"kwargs": {

"column": "pay\_type",

"max\_value": 17,

"min\_value": 7,

"mostly": 1.0,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"column\_max\_length\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.length.max",

"metric\_value\_kwargs": null

},

"num\_batches": 1

},

"column\_min\_length\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.length.min",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_match\_regex",

"kwargs": {

"column": "pay\_type",

"mostly": 1.0,

"regex": "(?:[A-Fa-f0-9]){0,4}(?: ?:? ?(?:[A-Fa-f0-9]){0,4}){0,7}"

},

"meta": {

"profiler\_details": {

"evaluated\_regexes": {

"(?:25[0-5]|2[0-4]\\d|[01]\\d{2}|\\d{1,2})(?:.(?:25[0-5]|2[0-4]\\d|[01]\\d{2}|\\d{1,2})){3}": 0.0,

"(?:[A-Fa-f0-9]){0,4}(?: ?:? ?(?:[A-Fa-f0-9]){0,4}){0,7}": 1.0,

"-?\\d+": 0.0,

"-?\\d+(?:\\.\\d\*)?": 0.0,

"<\\/?(?:p|a|b|img)(?: \\/)?>": 0.0,

"[A-Za-z0-9\\.,;:!?()\\\"'%\\-]+": 1.0,

"\\b[0-9a-fA-F]{8}\\b-[0-9a-fA-F]{4}-[0-5][0-9a-fA-F]{3}-[089ab][0-9a-fA-F]{3}-\\b[0-9a-fA-F]{12}\\b ": 0.0,

"\\d+": 0.0,

"\\s+$": 0.0,

"^\\s+": 0.0,

"https?:\\/\\/(?:www\\.)?[-a-zA-Z0-9@:%.\_\\+~#=]{2,255}\\.[a-z]{2,6}\\b(?:[-a-zA-Z0-9@:%\_\\+.~#()?&//=]\*)": 0.0

},

"success\_ratio": 1.0

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_be\_in\_set",

"kwargs": {

"column": "coupon\_use\_status",

"mostly": 1.0,

"value\_set": [

"0",

"1"

]

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column.distinct\_values",

"metric\_value\_kwargs": null

},

"num\_batches": 1,

"parse\_strings\_as\_datetimes": false

}

}

},

{

"expectation\_type": "expect\_column\_unique\_value\_count\_to\_be\_between",

"kwargs": {

"column": "coupon\_use\_status",

"max\_value": 2,

"min\_value": 2,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column.distinct\_values.count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_proportion\_of\_unique\_values\_to\_be\_between",# 该度量删除

"kwargs": {

"column": "coupon\_use\_status",

"max\_value": 0.00011123470522803115,

"min\_value": 0.00011123470522803115,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column.unique\_proportion",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_be\_in\_set",

"kwargs": {

"column": "pay\_type",

"mostly": 1.0,

"value\_set": [

"CMB\_SUB\_WALLET",

"TFT\_WALLET",

"ALI\_FAST",

"BOC\_DCEP\_WALLET",

"CCB\_SUB\_WALLET",

"ICBC\_SUB\_WALLET",

"WX\_FAST",

"BOC\_SUB\_WALLET",

"BCM\_SUB\_WALLET",

"CCB\_DCEP\_WALLET",

"WEBANK\_SUB\_WALLET",

"UNION\_PAY",

"TFT\_PAY"

]

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column.distinct\_values",

"metric\_value\_kwargs": null

},

"num\_batches": 1,

"parse\_strings\_as\_datetimes": false

}

}

},

{

"expectation\_type": "expect\_column\_unique\_value\_count\_to\_be\_between",

"kwargs": {

"column": "pay\_type",

"max\_value": 13,

"min\_value": 13,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column.distinct\_values.count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_proportion\_of\_unique\_values\_to\_be\_between",#该度量删除

"kwargs": {

"column": "pay\_type",

"max\_value": 0.0007230255839822024,

"min\_value": 0.0007230255839822024,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column.unique\_proportion",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

}

],

"ge\_cloud\_id": null,

"meta": {

"citations": [

{

"citation\_date": "2023-01-17T07:36:41.063745Z",

"comment": "Created by effective Rule-Based Profiler of OnboardingDataAssistant with the configuration included.\n"

}

],

"great\_expectations\_version": "0.15.28"

}

}

### 优化后的度量标准

cat hadoop\_catalog\_sdm\_s\_tft\_ups\_dc\_trip\_order.json

{

"data\_asset\_type": null,

"expectation\_suite\_name": "hadoop\_catalog\_sdm\_s\_tft\_ups\_dc\_trip\_order",

"expectations": [

{

"expectation\_type": "expect\_table\_row\_count\_to\_be\_between",

"kwargs": {

"max\_value": 50000,

"min\_value": 1000

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {},

"metric\_dependencies": null,

"metric\_name": "table.row\_count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_table\_columns\_to\_match\_set",

"kwargs": {

"column\_set": [

"coupon\_use\_status",

"pay\_type",

"update\_time"

],

"exact\_match": null

},

"meta": {

"profiler\_details": {

"success\_ratio": 1.0

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_not\_be\_null",

"kwargs": {

"column": "update\_time"

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "update\_time"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.nonnull.unexpected\_count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_not\_be\_null",

"kwargs": {

"column": "coupon\_use\_status"

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.nonnull.unexpected\_count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_not\_be\_null",

"kwargs": {

"column": "pay\_type"

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.nonnull.unexpected\_count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_value\_lengths\_to\_be\_between",

"kwargs": {

"column": "coupon\_use\_status",

"max\_value": 1,

"min\_value": 1,

"mostly": 1.0,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"column\_max\_length\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.length.max",

"metric\_value\_kwargs": null

},

"num\_batches": 1

},

"column\_min\_length\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.length.min",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_match\_regex",

"kwargs": {

"column": "coupon\_use\_status",

"mostly": 1.0,

"regex": "(?:[A-Fa-f0-9]){0,4}(?: ?:? ?(?:[A-Fa-f0-9]){0,4}){0,7}"

},

"meta": {

"profiler\_details": {

"evaluated\_regexes": {

"(?:25[0-5]|2[0-4]\\d|[01]\\d{2}|\\d{1,2})(?:.(?:25[0-5]|2[0-4]\\d|[01]\\d{2}|\\d{1,2})){3}": 0.0,

"(?:[A-Fa-f0-9]){0,4}(?: ?:? ?(?:[A-Fa-f0-9]){0,4}){0,7}": 1.0,

"-?\\d+": 1.0,

"-?\\d+(?:\\.\\d\*)?": 1.0,

"<\\/?(?:p|a|b|img)(?: \\/)?>": 0.0,

"[A-Za-z0-9\\.,;:!?()\\\"'%\\-]+": 1.0,

"\\b[0-9a-fA-F]{8}\\b-[0-9a-fA-F]{4}-[0-5][0-9a-fA-F]{3}-[089ab][0-9a-fA-F]{3}-\\b[0-9a-fA-F]{12}\\b ": 0.0,

"\\d+": 1.0,

"\\s+$": 0.0,

"^\\s+": 0.0,

"https?:\\/\\/(?:www\\.)?[-a-zA-Z0-9@:%.\_\\+~#=]{2,255}\\.[a-z]{2,6}\\b(?:[-a-zA-Z0-9@:%\_\\+.~#()?&//=]\*)": 0.0

},

"success\_ratio": 1.0

}

}

},

{

"expectation\_type": "expect\_column\_value\_lengths\_to\_be\_between",

"kwargs": {

"column": "pay\_type",

"max\_value": 17,

"min\_value": 7,

"mostly": 1.0,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"column\_max\_length\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.length.max",

"metric\_value\_kwargs": null

},

"num\_batches": 1

},

"column\_min\_length\_range\_estimator": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column\_values.length.min",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_match\_regex",

"kwargs": {

"column": "pay\_type",

"mostly": 1.0,

"regex": "(?:[A-Fa-f0-9]){0,4}(?: ?:? ?(?:[A-Fa-f0-9]){0,4}){0,7}"

},

"meta": {

"profiler\_details": {

"evaluated\_regexes": {

"(?:25[0-5]|2[0-4]\\d|[01]\\d{2}|\\d{1,2})(?:.(?:25[0-5]|2[0-4]\\d|[01]\\d{2}|\\d{1,2})){3}": 0.0,

"(?:[A-Fa-f0-9]){0,4}(?: ?:? ?(?:[A-Fa-f0-9]){0,4}){0,7}": 1.0,

"-?\\d+": 0.0,

"-?\\d+(?:\\.\\d\*)?": 0.0,

"<\\/?(?:p|a|b|img)(?: \\/)?>": 0.0,

"[A-Za-z0-9\\.,;:!?()\\\"'%\\-]+": 1.0,

"\\b[0-9a-fA-F]{8}\\b-[0-9a-fA-F]{4}-[0-5][0-9a-fA-F]{3}-[089ab][0-9a-fA-F]{3}-\\b[0-9a-fA-F]{12}\\b ": 0.0,

"\\d+": 0.0,

"\\s+$": 0.0,

"^\\s+": 0.0,

"https?:\\/\\/(?:www\\.)?[-a-zA-Z0-9@:%.\_\\+~#=]{2,255}\\.[a-z]{2,6}\\b(?:[-a-zA-Z0-9@:%\_\\+.~#()?&//=]\*)": 0.0

},

"success\_ratio": 1.0

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_be\_in\_set",

"kwargs": {

"column": "coupon\_use\_status",

"mostly": 1.0,

"value\_set": [

"0",

"1"

]

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column.distinct\_values",

"metric\_value\_kwargs": null

},

"num\_batches": 1,

"parse\_strings\_as\_datetimes": false

}

}

},

{

"expectation\_type": "expect\_column\_unique\_value\_count\_to\_be\_between",

"kwargs": {

"column": "coupon\_use\_status",

"max\_value": 2,

"min\_value": 2,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "coupon\_use\_status"

},

"metric\_dependencies": null,

"metric\_name": "column.distinct\_values.count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

},

{

"expectation\_type": "expect\_column\_values\_to\_be\_in\_set",

"kwargs": {

"column": "pay\_type",

"mostly": 1.0,

"value\_set": [

"CMB\_SUB\_WALLET",

"TFT\_WALLET",

"ALI\_FAST",

"BOC\_DCEP\_WALLET",

"CCB\_SUB\_WALLET",

"ICBC\_SUB\_WALLET",

"WX\_FAST",

"BOC\_SUB\_WALLET",

"BCM\_SUB\_WALLET",

"CCB\_DCEP\_WALLET",

"WEBANK\_SUB\_WALLET",

"UNION\_PAY",

"TFT\_PAY"

]

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column.distinct\_values",

"metric\_value\_kwargs": null

},

"num\_batches": 1,

"parse\_strings\_as\_datetimes": false

}

}

},

{

"expectation\_type": "expect\_column\_unique\_value\_count\_to\_be\_between",

"kwargs": {

"column": "pay\_type",

"max\_value": 13,

"min\_value": 13,

"strict\_max": false,

"strict\_min": false

},

"meta": {

"profiler\_details": {

"metric\_configuration": {

"domain\_kwargs": {

"column": "pay\_type"

},

"metric\_dependencies": null,

"metric\_name": "column.distinct\_values.count",

"metric\_value\_kwargs": null

},

"num\_batches": 1

}

}

}

],

"ge\_cloud\_id": null,

"meta": {

"citations": [

{

"citation\_date": "2023-01-17T07:36:41.063745Z",

"comment": "Created by effective Rule-Based Profiler of OnboardingDataAssistant with the configuration included.\n"

}

],

"great\_expectations\_version": "0.15.28"

}

}

## 在Airflow中执行GE

至此，GE的expectation和Checkpoint准备完毕，接下来在Airflow中执行GE质量度量任务。其实，GE官方提供了Airflow的Operator，但是。

这里，采用PythonOperator执行GE作业，以s\_tft\_tsm\_t\_trade\_ge.py为例。

#!/usr/bin/env python

# -\*- coding:utf-8 -\*-

# author: Changhua Gong

import pendulum

from airflow.decorators import dag

from airflow.operators.python import PythonOperator

from airflow.utils.trigger\_rule import TriggerRule

# noinspection PyPep8Naming

import tft.util.common\_util as CommonUtil

from tft.util.ge\_util import run\_ge\_checkpoint, ge\_env

ge\_env()

table\_name\_full\_path = "hadoop\_catalog.sdm.s\_tft\_tsm\_t\_trade"

data\_dt = CommonUtil.data\_dt # If no data\_dt column exists, set to empty string ""

col\_list = ["order\_state", "trade\_channel"]

table\_name = table\_name\_full\_path.split(".")[2]

@dag(

catchup=False,

dag\_id=f"{table\_name}\_ge",

schedule="32 16 \* \* \*",

start\_date=pendulum.datetime(2022, 10, 28, tz="Asia/Shanghai"),

max\_active\_runs=1,

tags=['great\_expectation', f'{table\_name}']

)

# If Obtain spark session here, spark session would be created by airflow when scanning dags code

# So, dataframe reference must be in a single piece code .

def run\_this():

# Queue only for operator level

PythonOperator(python\_callable=run\_ge\_checkpoint,

task\_id=f"{table\_name}\_ge\_task",

op\_kwargs={"table\_name": table\_name\_full\_path, "data\_dt": data\_dt,

"col\_list": col\_list},

trigger\_rule=TriggerRule.ALL\_SUCCESS,

retries=0,

queue="ge")

run\_this()

红色部分，请与[7.2](#_生成度量依据（标准）)和[7.3](#_优化度量依据（标准）)中度量字段保持一致，将上述DAG中部署在生产环境并启用。GE作业触发并且成功完成后，则将度量结果投递至Datahub中。

## 在Datahub查看度量结果

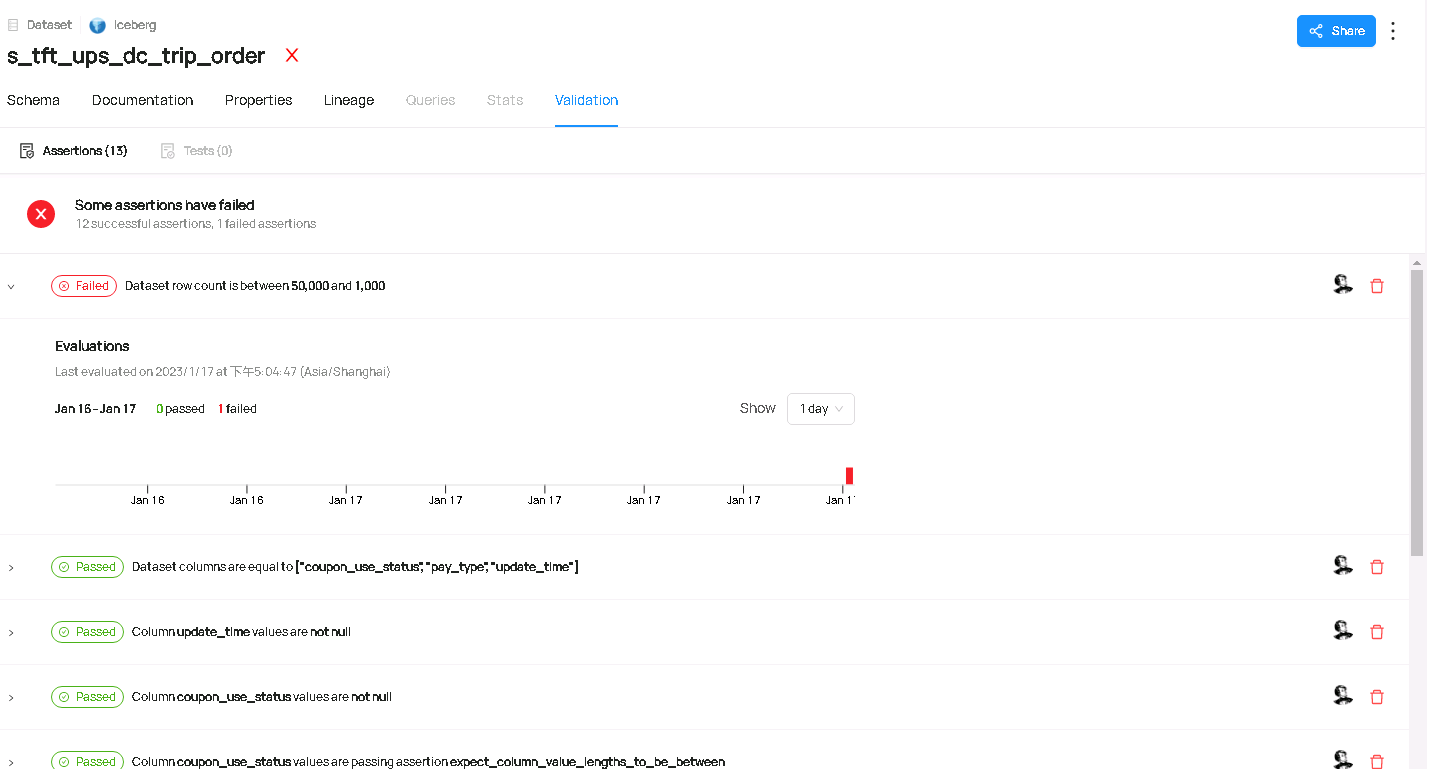


图 1 - 未通过的度量结果

图中红叉表示最近一次度量并未通过，只要一个度量规则未通过，则判定为未通过。假设成功的度量结果再次推送至Datahub中，则以最新的度量结果为准，图中红叉会变更为绿勾，例如：

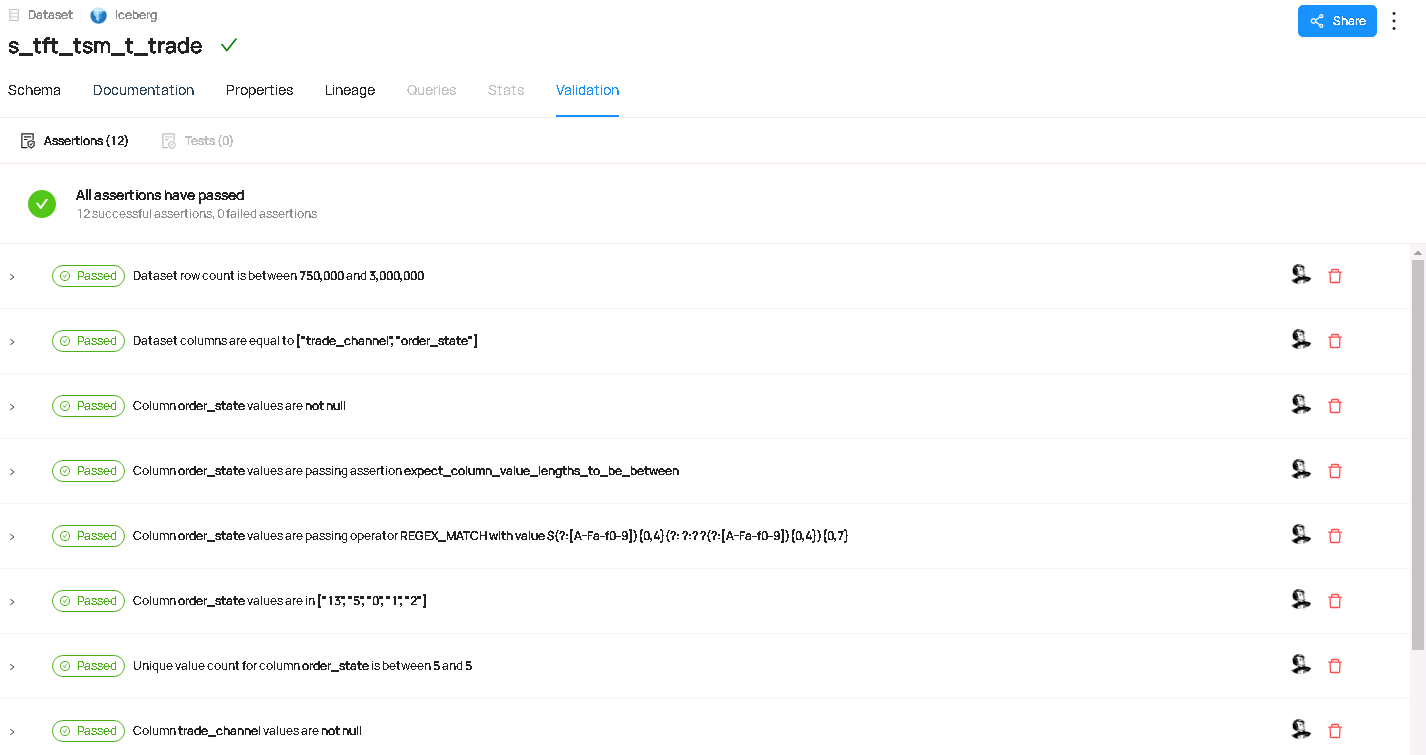


图 2 - 通过的度量标准

### 修订度量标准

当度量标准正常被执行，由于某些原因需调整度量标准后重新上线。这时，需执行以下两步：

1. 执行如[7.3](#_优化度量依据（标准）)中的步骤，改善GE工作空间expectations目录下度量标准；
2. 将Datahub中Validation Tab下的所有度量结果删除，让GE重新推送结果生成。

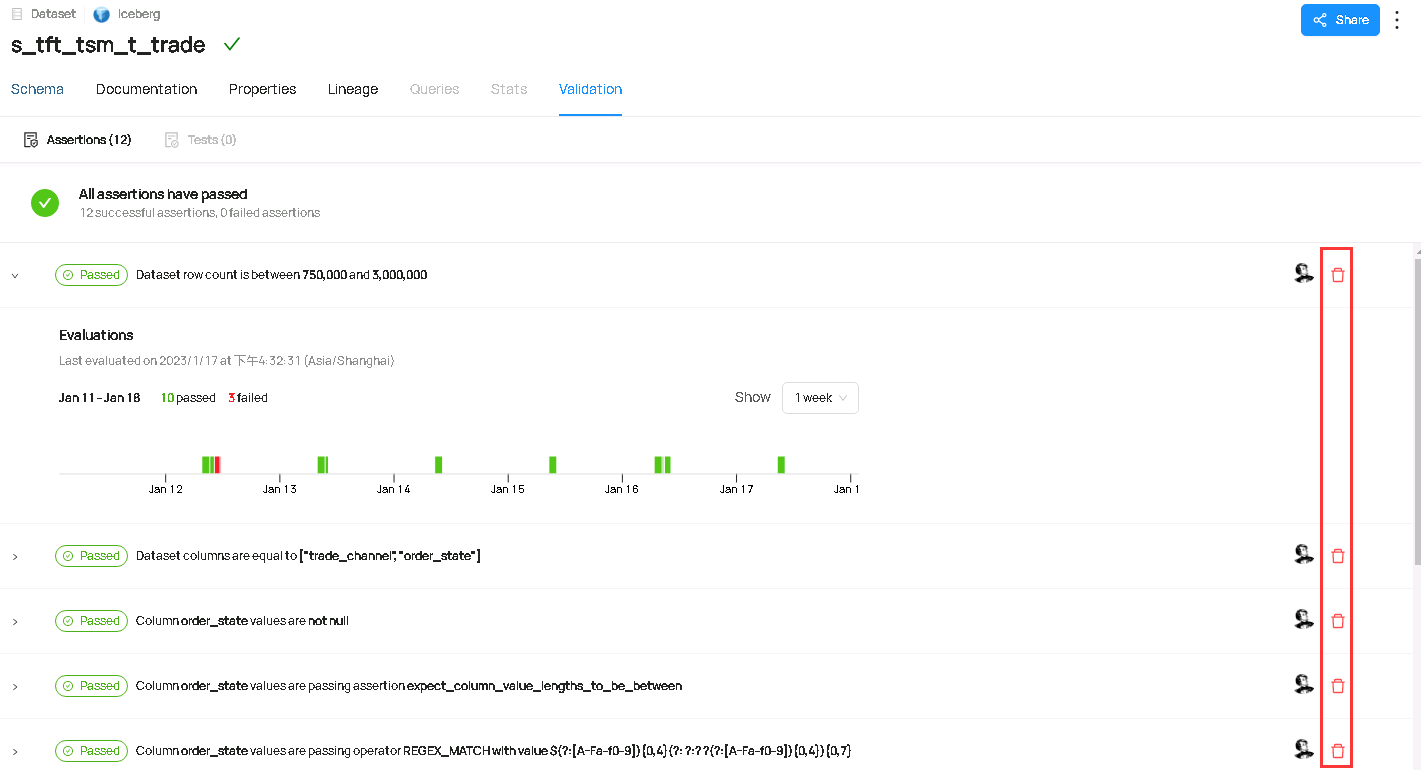


图 3 - 删除度量结果

注：如果不执行删除，重新推送的度量结果将错乱覆盖，具体原因尚未调查。

## 设置异常度量结果短信告警

将附录中custom\_actions.py上传至plugins目录，如果安装过程中已上传则忽略，生产环境路径为

/hdp\_data/gch/venv/great\_expectations/great\_expectations/plugins

[root@namenode1 plugins]# pwd

/hdp\_data/gch/venv/great\_expectations/great\_expectations/plugins

[root@namenode1 plugins]# ls

custom\_actions.py

### 在生成度量依据时设置

在[7.2](#_生成度量依据（标准）)生成度量依据中，直接设置短信告警的action。

checkpoint\_yaml\_config = f"""

name: {checkpoint\_name}

config\_version: 1.0

class\_name: SimpleCheckpoint

run\_name\_template: '%Y%m%d-%H%M%S-{datahub\_dataset}'

expectation\_suite\_name: {expectation\_suite\_name}

action\_list:

- name: datahub\_action

action:

module\_name: datahub.integrations.great\_expectations.action

class\_name: DataHubValidationAction

server\_url: http://10.50.253.9:8080

"""

找到如上类似代码段，添加短信告警action。

checkpoint\_yaml\_config = f"""

name: {checkpoint\_name}

config\_version: 1.0

class\_name: SimpleCheckpoint

run\_name\_template: '%Y%m%d-%H%M%S-{datahub\_dataset}'

expectation\_suite\_name: {expectation\_suite\_name}

action\_list:

- name: datahub\_action

action:

module\_name: datahub.integrations.great\_expectations.action

class\_name: DataHubValidationAction

server\_url: <http://10.50.253.9:8080>

- name: sms\_action

action:

module\_name: custom\_actions

class\_name: SmsNotificationAction

sms\_webhook: http://10.50.253.7:18888/sms/push

app\_name: great\_expectations

phones:

- 18011449543

- 15888888888

notify\_on: failure

"""

红色部分为添加代码，这样在制作checkpoint时直接配置。

配置说明：

1. Module\_name为自定义插件的模块名，对应custom\_actions.py，如果模块名发生变更或新增模块，则需调整；
2. Class\_name为上述模块中实现ValidationAction类的子类，如果类名发生变更，则需调整；
3. Sms\_webhook为短信服务地址，如果短信服务地址发生变更，则需调整；
4. App\_name为短信内容“大数据应用服务”名，一般来说不需要调整；
5. Phones为送达手机列表，一般来说需要调整；
6. Notify\_on定制发送内容，可选值为"all", "failure", "success"，"all"表示无论成功失败都发送短信，"failure"表示只有失败时才发送短信，"success"表示只有成功时才发送短信，成功失败说明：所有度量均成功则为成功，只要由一个度量失败则为失败，该配置根据需要更改。

发送短信内容示例：

【天府通】大数据应用服务[great\_expectations],运行情况[sdm.s\_tft\_tsm\_t\_trade(20230129) - success],请知晓

### 修改checkpoint配置

在checkpoint中配置短信通知的action，这里以sdm.s\_tft\_tsm\_t\_trade表为例。

[root@namenode1 checkpoints]# pwd

/hdp\_data/gch/venv/great\_expectations/great\_expectations/checkpoints

[root@namenode1 checkpoints]# more hadoop\_catalog\_sdm\_s\_tft\_tsm\_t\_trade.yml

name: hadoop\_catalog\_sdm\_s\_tft\_tsm\_t\_trade

config\_version: 1.0

template\_name:

module\_name: great\_expectations.checkpoint

class\_name: Checkpoint

run\_name\_template: '%Y%m%d-%H%M%S-s\_tft\_tsm\_t\_trade'

expectation\_suite\_name: hadoop\_catalog\_sdm\_s\_tft\_tsm\_t\_trade

batch\_request: {}

action\_list:

- name: store\_validation\_result

action:

class\_name: StoreValidationResultAction

- name: store\_evaluation\_params

action:

class\_name: StoreEvaluationParametersAction

- name: update\_data\_docs

action:

class\_name: UpdateDataDocsAction

site\_names: []

- name: datahub\_action

action:

module\_name: datahub.integrations.great\_expectations.action

class\_name: DataHubValidationAction

server\_url: http://10.50.253.9:8080

- name: sms\_action

action:

module\_name: custom\_actions

class\_name: SmsNotificationAction

sms\_webhook: http://10.50.253.7:18888/sms/push

app\_name: great\_expectations

phones:

- 18011449543

- 15888888888

notify\_on: failure

evaluation\_parameters: {}

runtime\_configuration: {}

validations: []

profilers: []

ge\_cloud\_id:

expectation\_suite\_ge\_cloud\_id:

# 总结

Great Expectations套件的引入，将质量管理工作从石器时代推进至半自动化时代。采用插件式设计架构，使其能够丰富适配各项数据源、度量引擎、Action，支持可扩展式接口开发。Great Expectations套件与Airflow、Datahub集成，使资源得以有效整合，数据治理工作一站化。

# 待办（TODO）

1. 或采取ExternalPythonOperator执行GE作业，以达到执行环境解耦；
2. 适配StarRocks数据源，执行GE作业。

# 附录



Great Expectations官方网站：https://docs.greatexpectations.io/docs/