# Report Configuration Approach

The report configuration not only needs to determine the report requirements, but also the logic of extracting data from the trading system. After extracting the data, define the business scope and fields of the data table, data conversion, and visual display of the data in the data platform. The following briefly introduces the method of report analysis.

## Report Analysis General Approach

Here is the general approach for Report analysis: First, you need to determine the report requirements with the customer, and determine whether the report requirements require data warehouse functions. Then, Edit the requirements document. and determine the number-drawing logic of the trading system and configure the appropriate number-drawing logic to meet the reporting requirements with the constraints of the trading system to ensure that the required data enters the data platform.

Second, create a raw topic into the data platform and determine how to deploy layers the data after entering the data platform, how to deploy layers means that different forms of data tables are placed at different levels and determine how different levels of data tables are converted and flowed so that the test data needs to be converted and flowed as expected.

Third, select a specific data table to generate a report.

There are three important parts in these parts. First, determine customer needs, second, data extraction logic, and third, data layering and transformation logic

## Report Configuration Prerequisites

As a data development and designer, the most important thing is to have a deep understanding of the data tables of the trading system, a deep understanding of the operation of the database, familiar with the basic SQL language.

As a report developer, you just need to be familiar with the operation of the data platform and understand the basic SQL language.

### Related environment configuration

1. Postman

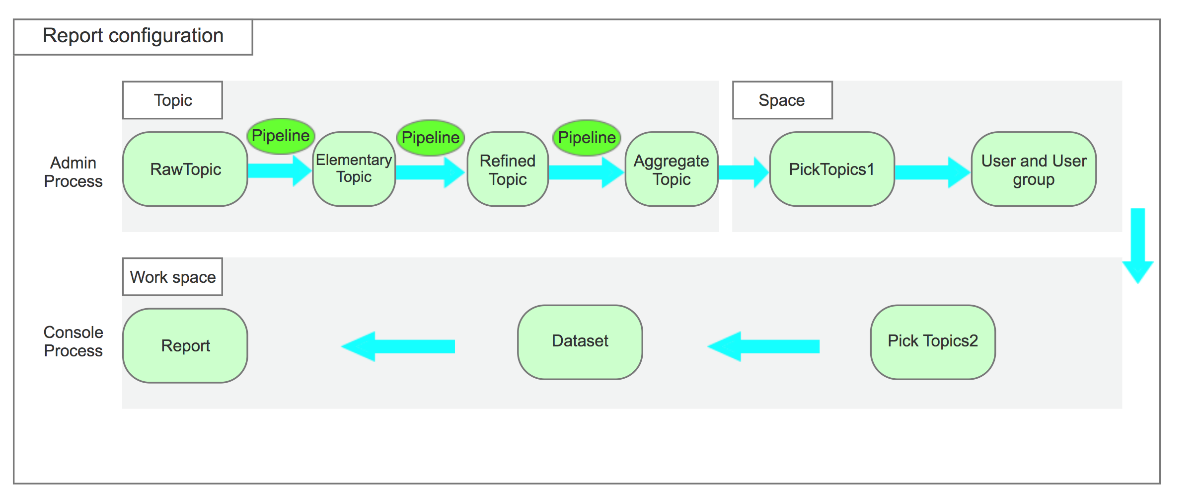
We can use Postman to test complex data. In order to push data, we must ensure that the user has permission. The authorization process is described in detail in the following section: [Input information](#_Input_information)

# Report Definition Process

Here is a general process to launch a new Report in terms of the system.

## Definition Process

Figure 2‑1 Report Definition Process Diagram



**Topic**

The Admin user has operation authority, and there are the following steps to generate Topic.

* Import Raw Topic table structure
* Create Elementary Topic table structure
* Create Refined Topic or Enumerates table structure
* Use Pipeline to connect multiple levels of Topic tables and configure transmission rules in them.

**Space**

The Admin user has the operation authority, and there are the following steps to create a Space.

* Add the required topics that have been created to the Space.
* Create a User and User group and add users to the Space. In the subsequent Workspace operations, only the added users have the operation authority.

**Work Space**

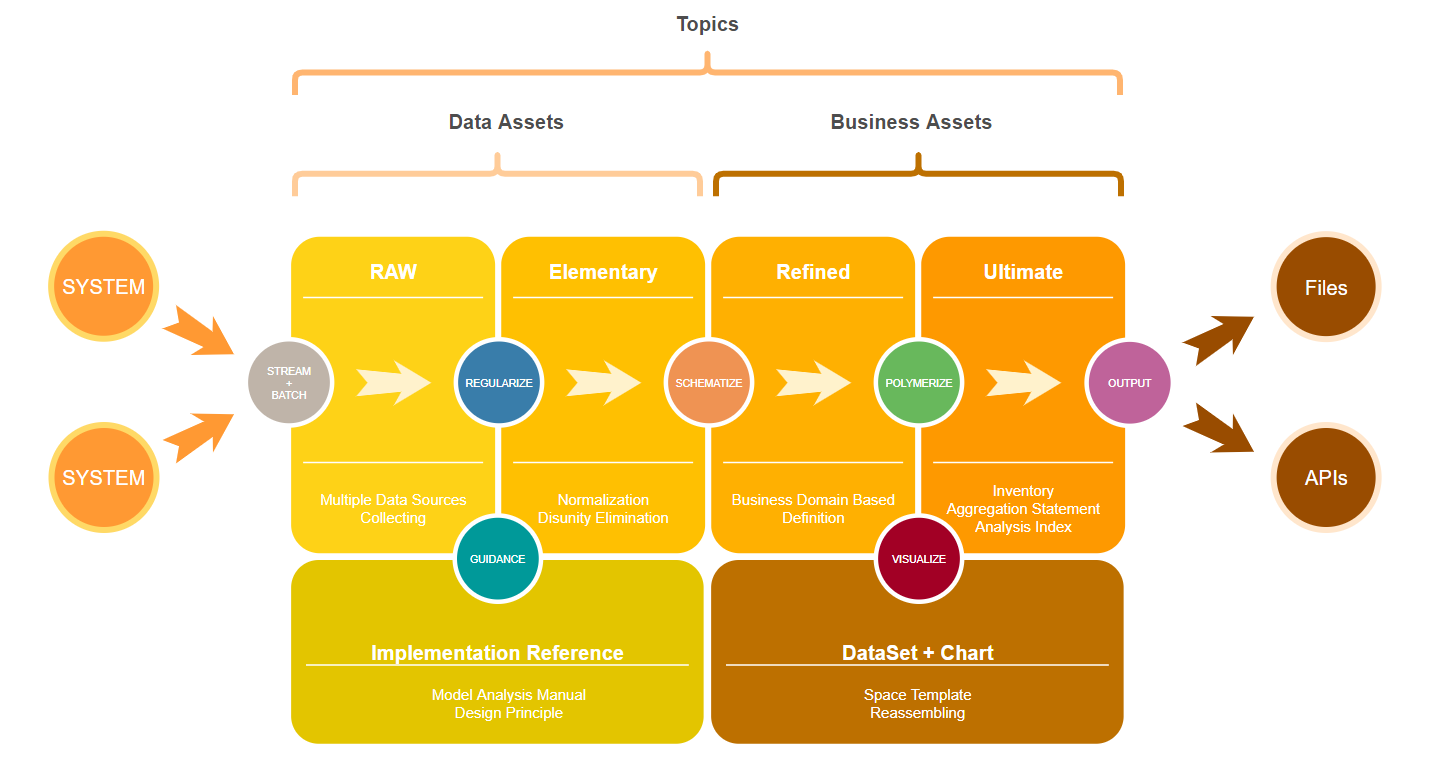
The console user has the operation authority, and the following steps are required to generate the report

* Select the required Topic, filter or join or group to generate a Dataset according to business requirements.
* Based on Dataset, use BI tools to generate Report.

## Topic layers Introduction

The core part is the Topic. The following is a detailed introduction of Topic layering. The actual application situation needs to be weighed with the data format and business requirements provided by the trading system.

Figure 2‑2 Topic layers



* Raw Topic:
* Mainly subject-oriented, integrated, current or near current, constantly changing data.
* Import metadata from the trading system to create the layer.
* Elementary Topic:
* Take the business process as the modeling drive, and build the most fine-grained detailed fact table based on the characteristics of each specific business process.
* Every elementary topic is mainly for different business topic of one module.
* It depends on the data that entered.
* Topic can be an overlay of historical data.
* Refined Topic
* Refined Topic is mainly to meet different business needs.
* It depends on the report requirements.
* Aggregate Topic:
* This layer is the data summary layer. Topic can be a complete set of multiple similar summary reports, or it can be a topic created for special business needs
* Dataset:
* Store personalized statistical indicator data
* Operate in the workspace and provide direct data for the Report.

# Report Configuration Tools

Report Configuration Tools contains the following functionalities:

* Topic
* Space
* Work Space

## Topic

Topic Definition contains the following sub-tasks:

* Import Topic
* Create Topic or Enumerates
* Pipeline

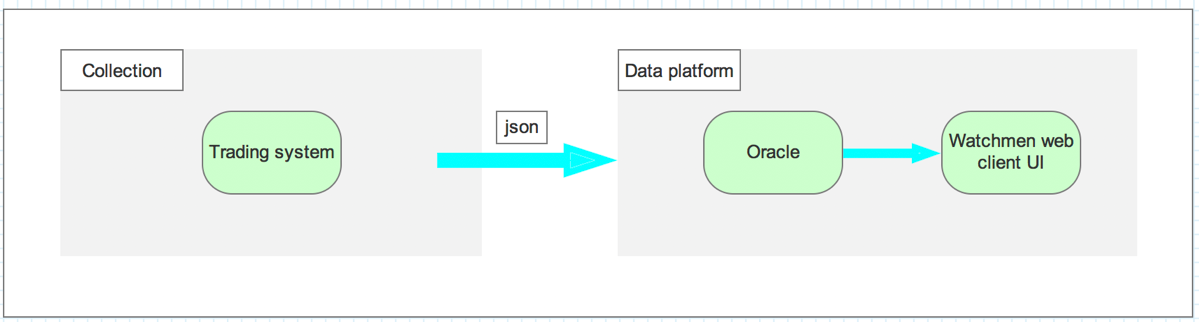
### Topic Panorama

Topic is a table in the database. The table in the database and the topic in Watchman are independent of each other. Then, after creating/deleting/renaming topic or deleting fields/adding fields on the watchman page, you need to perform topic deletion/renaming/adding operations in the database, too. The creation of Raw Topic relies on the validity and completeness of the data provided by the trading system and the validity of the data received by the platform background, such as the correct processing of the corresponding field types.

### Import Raw Topic

The following figure shows the process of creating a raw topic.

Figure 3‑1 Create raw topic processing

1. Trading system-generate the corresponding Json file according to the module

2. post to watchman interface-create raw topic.

NOTE

In the second step, the raw topic has been established. Because the Json file may not cover all the fields required by the module, that is, some fields may be empty in key-value, so the raw topic created may reduce a small number of fields. These Fields need to be manually added to Topic.

For actual projects, you need to consider the hierarchical structure of importing the module data. According to a given business, a data module to be invested in this module can be drawn up, such as importing finance into the Finance module as a subordinate level of the PA policy.

The possible problem is that the json data provided by the trading system is incomplete in the process of importing the data into the platform. The first is that the amount of data is reduced, and the second is that some fields are not entered. You can consider the collection part and the platform to locate the problem.

### Json format

#### Date Format

#### The following time format types are not supported

#### 08/10/2021T19:08:03.000

#### Array Format

#### The following array format types are not supported:

"channelParentEntityCodes": ["MyAIG","A&H","64\_AH"]

"channelParentEntityNames": ["MyAIG","A&H","HAADYAI"]

You can modify it to the following format:

channelEntityInfo":

{

"parentEntityLevel1Code": "MyAIG",

"parentEntityLevel1Name": "MyAIG",

"parentEntityLevel2Code": "A&H",

"parentEntityLevel2Name": "A&H",

"parentEntityLevel3Code": "64\_AH",

"parentEntityLevel3Name": "HAADYAI",

"parentEntityLevel4Code": null,

"parentEntityLevel4Name": null

}

### Create raw topic schema

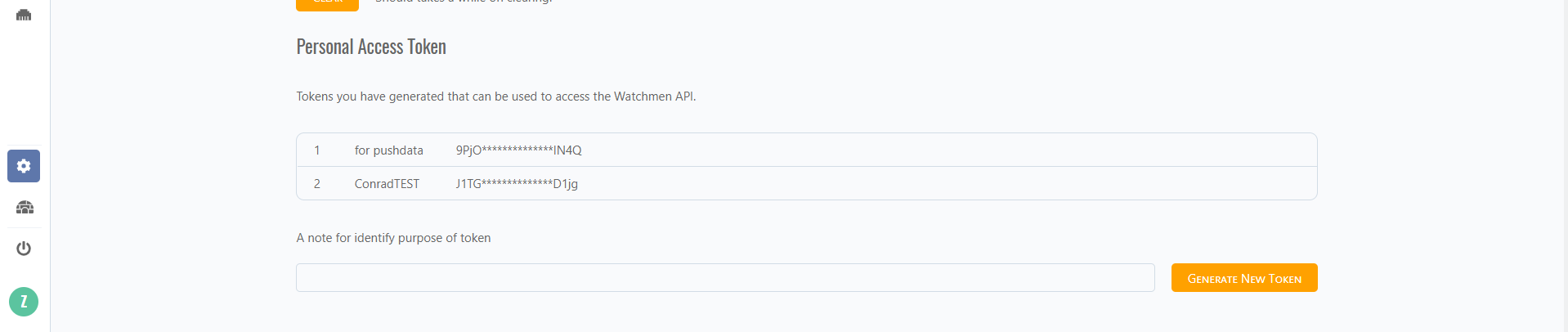
Before inputting data to the client, you need to determine the dataset’s name and factor format of the input data to ensure that standardizes data. Of course, you can also manually create raw topic on watchmen web client, but it takes more time, Therefore, using the postman tool to create raw topic schema is a good choice.

#### Input information

You can create a raw topic schema by using postman and posting Json format instance data. Here is the Create raw topic schema configuration page.

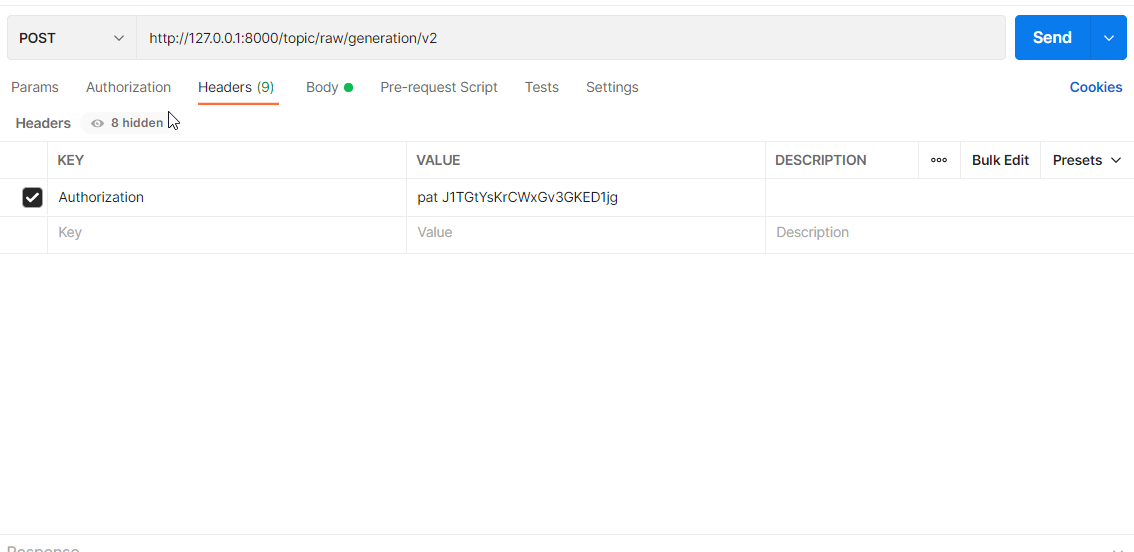
Before pushing data to create a raw topic, you must first log in to the watchman platform, click on the left directory bar and click on setting and input name that your token.

Figure 3‑3 Create raw topic schema: Postman configuration



Then click generate New token and record the token. You need to add the token to the header part of postman. The token is to control the permission to push data. After that, try to use the same token to deliver data.

Figure 3‑4 Create raw topic schema: Postman configuration



|  |  |
| --- | --- |
| Field | Value |
| Headers | **/** |
| Authorization | **pat** 【Your token from watchman data platform】 |

Figure 3‑5 Create raw topic schema: Postman configuration

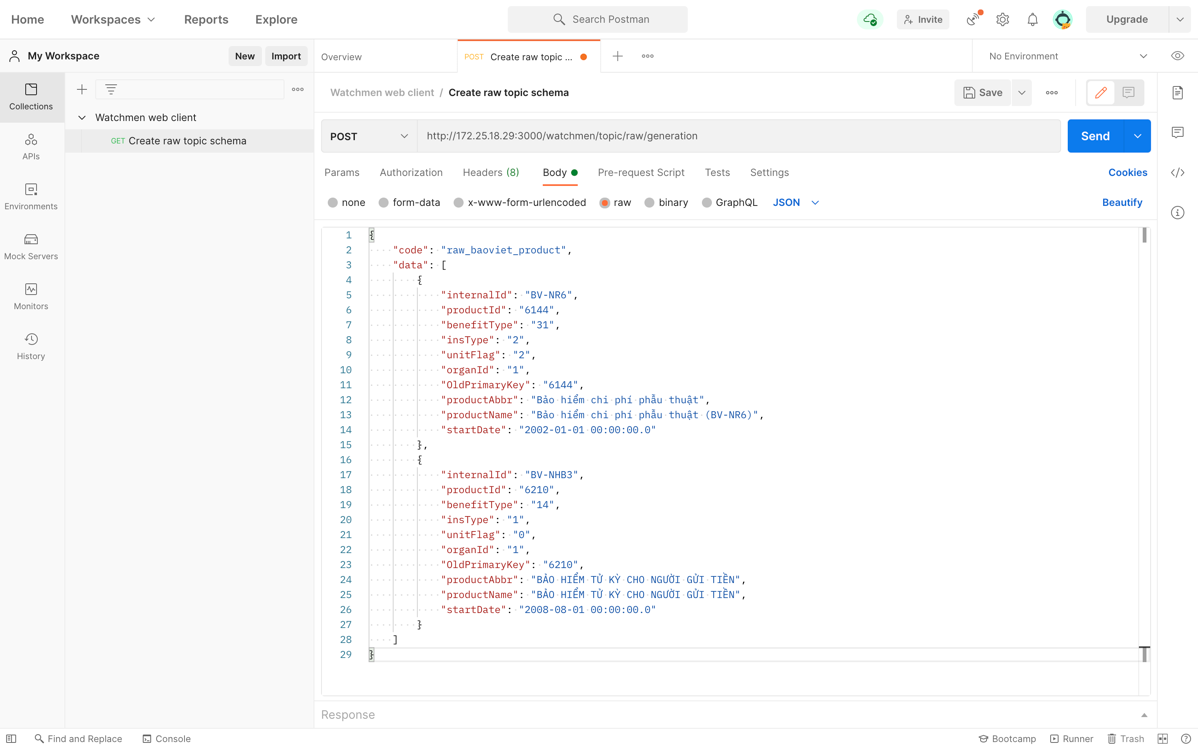


Table 3‑1 Create raw topic schema: Postman’s configuration

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Value | | |
| Method | Post | | |
| URL | http://172.25.18.\*\*:3001/watchmen/topic/raw/generation/v2  http://172.25.18.\*\*:3001/watchmen/topic/raw/generation/v3 | | |
| Body | raw | / | |
|  | / | Json | / |

NOTE

* The method and body cannot be changed. If You have other environments, you can change the URL, but " /watchmen/topic/raw/generation /v2 " cannot be changed. If you have your own doll program, you URL should be "http://IP:PORT/topic/raw/generation/v2".
* Note that in the source table of the trading system, useless fields should be filtered first, and then sent to the platform.
* v3 url is suitable for json that lacks a surrogate key in the nested array, which means that watchmen will automatically establish the relevant surrogate key on the watchman platform.In most cases, we use v2 to create raw topics. If you want to use V3, please discuss in detail.

Table 3‑2 Create raw topic schema: Json format instance data configuration

|  |  |  |
| --- | --- | --- |
| Key | Value | Description |
| code | raw\_baoviet\_product | The name of the topic to be created |
| data | [{},{},{}] | Input rules:   * The value **must** be in array format, that is, **the beginning and the end are ended with square brackets**. * In the array, each set of factors in curly braces represents an instance. * There is at least one instance in the array. * If there are at least two instances, the created raw topic schema is the format of the factor that belongs to the union of all instances. |

1. Create raw topic schema: Input

Method: post

URL: <http://172.25.18.29:3001/watchmen/topic/raw/generation>/v2

Body: raw, Json

{

"code": "raw\_baoviet\_product\_test",

"data": [

{

"internalId": "BV-NR6",

"benefitType": "31",

"insType": "2",

"unitFlag": "2",

},

{

"internalId": "BV-NHB3",

"productId": "6210",

"unitFlag": "0",

"organId": "1",

}

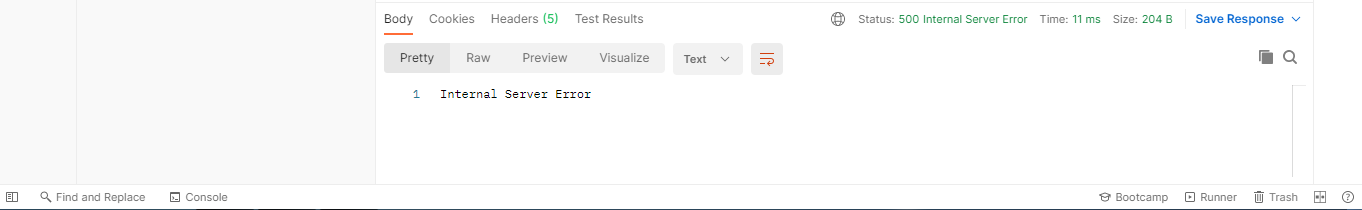
]

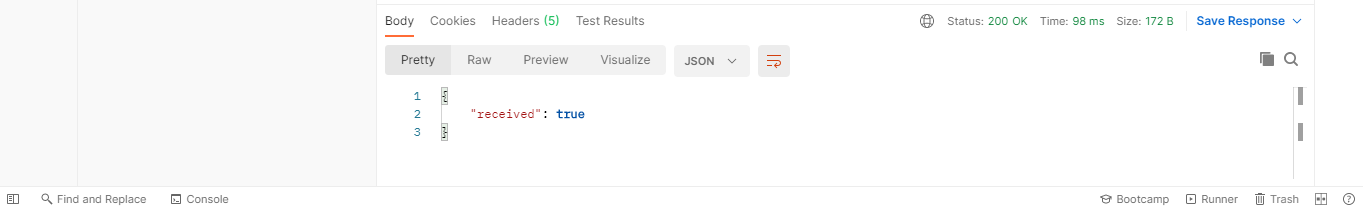
}

That means a raw topic named "raw\_baoviet\_product\_test" is created and It has six factors including "internalId","insType", "unitFlag","benefitType" "productid ","organId".

#### Output information

Here is the Common return form.

* It means acceptance failure.
* It means acceptance success.



### Input test data

After you create a raw topic, and also configure other topics and pipeline, you can use postman to test whether the entire process is complete.

#### Input information

Click【Add a request】and renamed request to Input test data. Here is the Input test data configuration page. Don't forget your token.

There is an **easy mistake** that the content format of post data is different from the content format of generation topic. The value of data for post data doesn’t have the **square brackets.**

Figure 3‑3 Input test data: Postman configuration

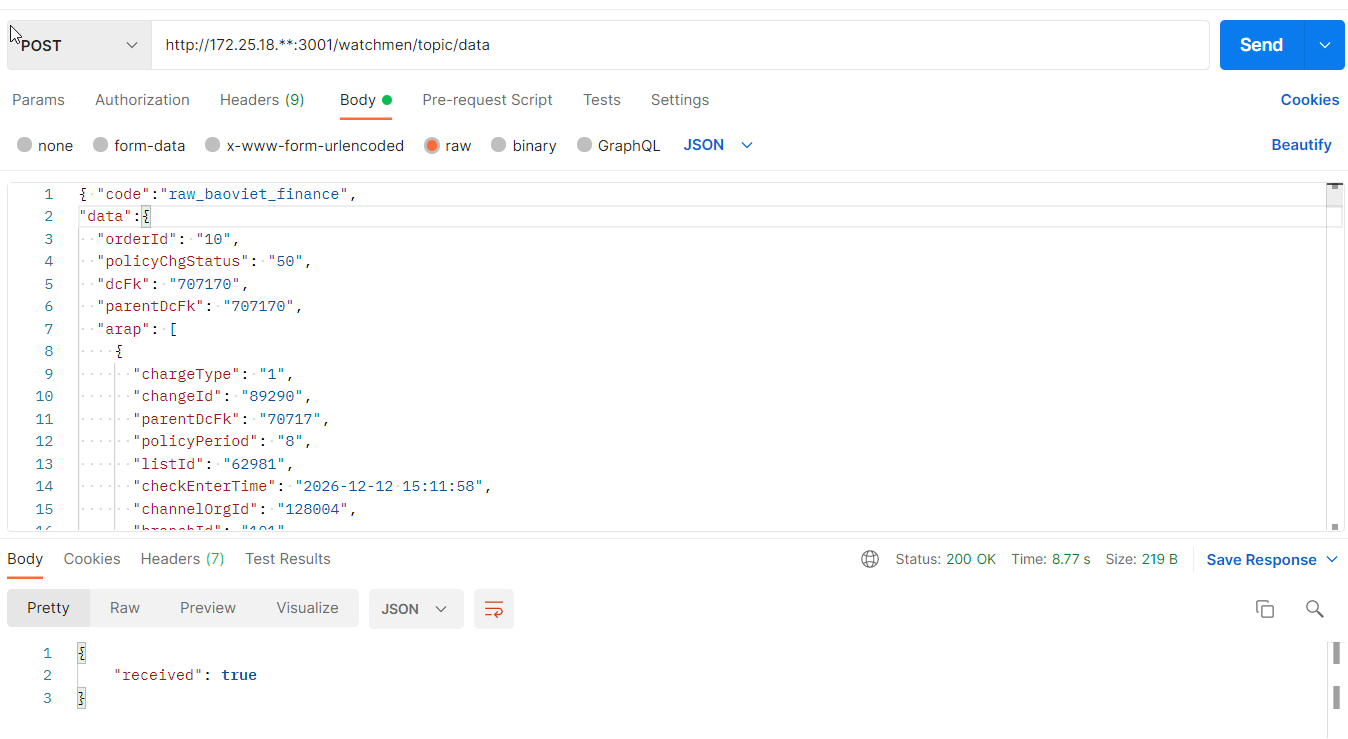


Table 3‑3 Input test data: Postman’s configuration

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Value | | |
| Method | Post | | |
| URL | http://172.25.18.29:3001/watchmen/topic/data  http://172.25.18.29:3001/watchmen/topic/data/v3 | | |
| Body | raw | / | |
|  | / | Json | / |

NOTE

The method and body cannot be changed. If You have other environments, you can change the URL, but ‘/watchmen/topic/data’ cannot be changed. If raw topic is created by v3, you should use v3 url to push data.

Table 2‑5 Input test data: Json format instance data configuration

|  |  |  |
| --- | --- | --- |
| Key | Value | Description |
| code | raw\_baoviet\_product | The name of the topic that receives the data |
| data | {} | Input rules:   * The content contained in curly braces represents an instance * Only **one instance** can be posted |

1. Input test data: Input

Method: post

URL: http://172.25.18.29:3001/watchmen/topic/data

Body: raw, Json

{

"code": "raw\_baoviet\_product\_test",

"data":

{

"internalId": "BV-NR6",

"benefitType": "31",

"insType": "2",

"unitFlag": "2",

}

}

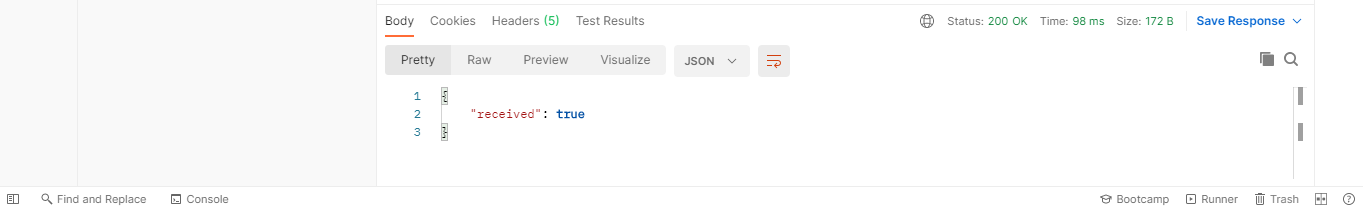
This means that you have input two test instances to the topic named "raw\_ baoviet\_product\_test", And the factor that is not included is a null value.

#### Output information

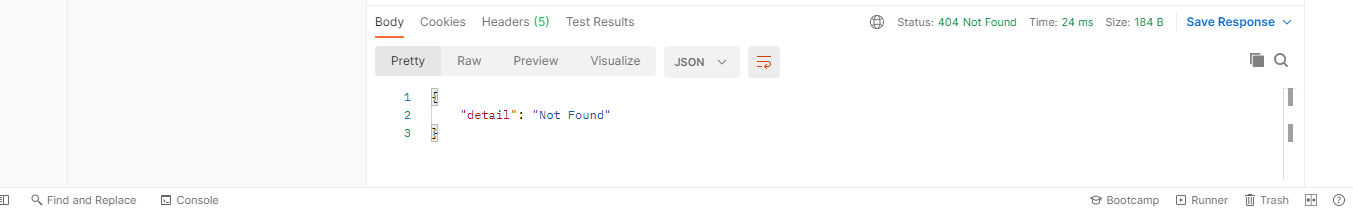
After posting the data to the watchmen web client, postman will return some information to let us judge whether the post is correct and where the problem is.

Here is the Common return form.

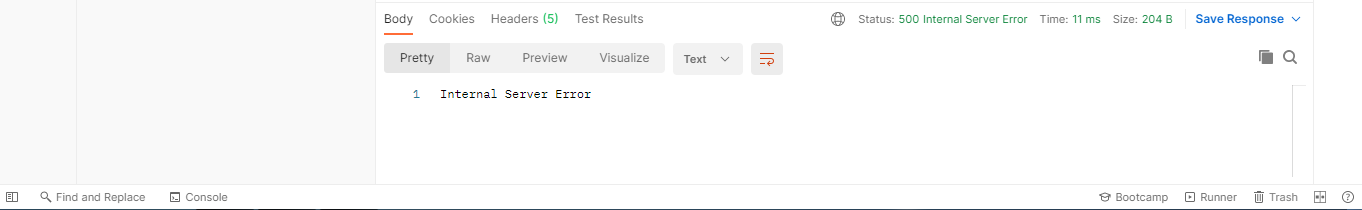
* It means acceptance success.



* It means acceptance failure and the reason is that the URL is incorrectly filled.

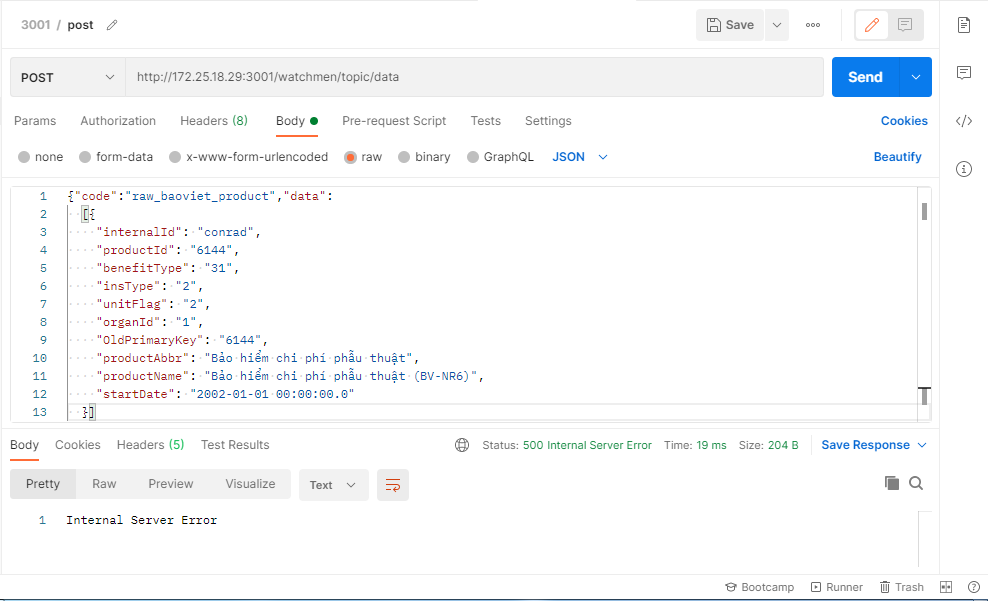


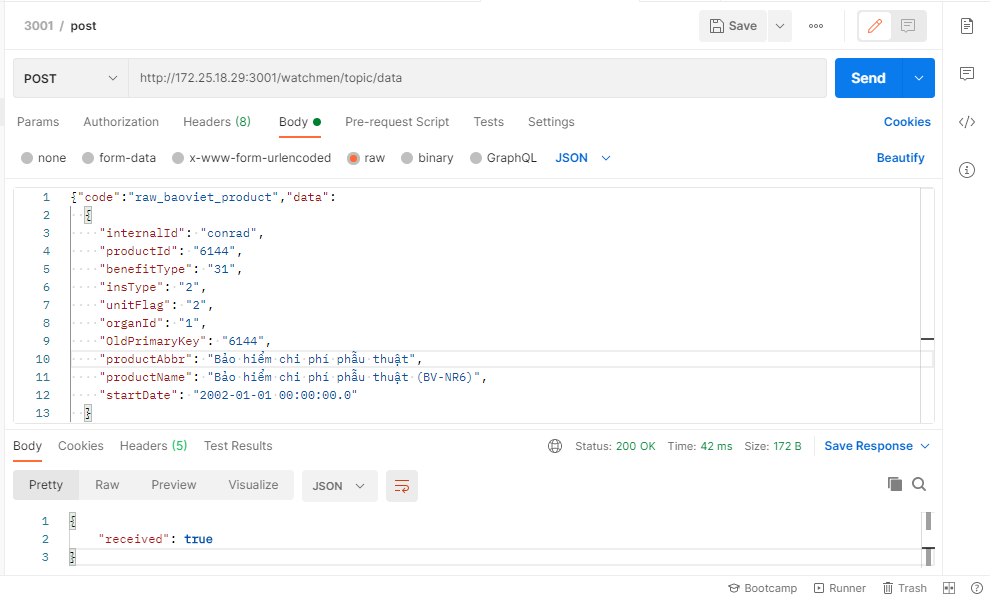
* It means acceptance failure and the reason is data format error or topic definition error or pipeline definition error.



1. Input test data: output

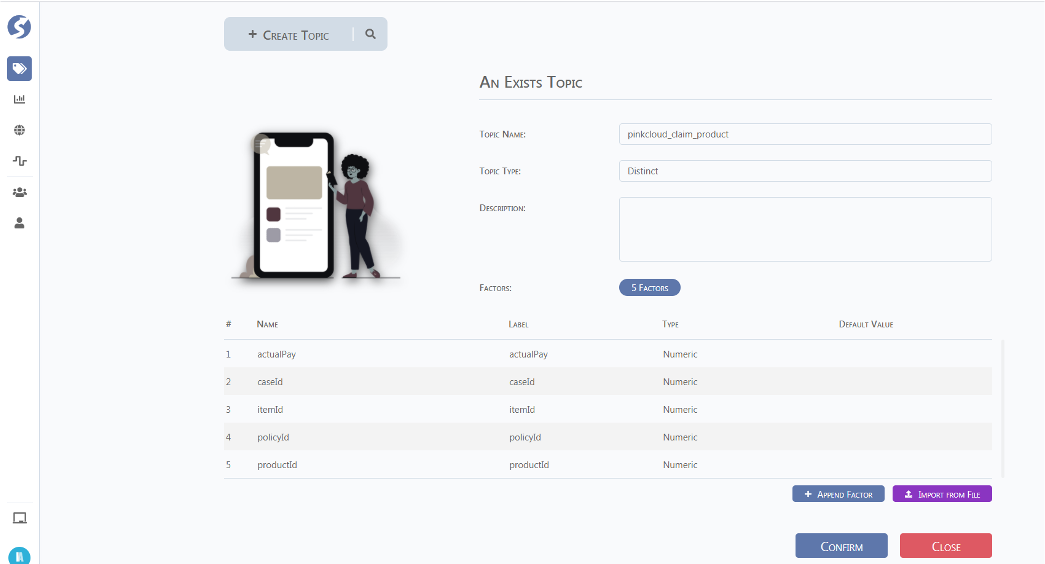
* If you add an instance in a pair of square brackets, an error will be reported



* The correct form is shown below

### Create Topic or Enumerates

#### Create Topic



Note

* The created Topic is the structure of the data table. If the data stream does not conform to the data table structure, you need to modify the Topic Factors structure or modify the input data stream structure.
* It is recommended to write topic names, field names, and attribute documents.
* Topic naming rules, follow the little hump method. After the topic is created, you need to click confirm to save it, otherwise the creation record cannot be kept.
* The purpose of Topic layering is to present the data structure in the clearest and most efficient way and to easily solve different business requirements. That is to answer the following question: Where is the scope of the included field of each topic? How do different topics connect to each other in the simplest way? Are there some fields that can be shared by different business fields to create a common topic?
* In addition to RawTopic, the definition of other topics and the structure between Topic and Topic are flexible. What needs to be improved is the business-based definition and structure specifications.
* If you have been created an aggregate topic , you must use **Insert or Merge** function to group factors and make sure that every group factor should be **Unique index**, Otherwise, there will be multiple records with the same group factors in the aggregate topic due to concurrency issues.
* Note that after creating a topic, you need to select the created topic in the pipeline group, otherwise you will not see the topic's visualization page in the pipeline group page.
* After you create some topics and fields (not including rawtopic), you should download the scrpits of the topic, and using scrpits to create the table corresponding to the topic in the database. **It is recommended that** **download scripts to configure tables in database after you have been configured all topics and pipelines**.

Table 3‑1 Field Descriptions on Create Topic Page

|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| 🔍 | / | Filter a specific field, if filter by name, the rule is n:name. |
| Topic name | / | The topic name must conform to the naming rules. |
| Topic kind | Business | Business Topic. Commonly used |
|  | System | Log monitor related topics |
| Topic Type | Distinct | The Topic has a unique key |
|  | Raw | Raw Topic |
| Aggregate | Aggregate Topic |
| Time | sequentially Topic |
| Ratio | Ratio Topic |
| Data Source | / | Select using data Source |
| Description | / | factor description |
| Factors | / | Number of fields |
| Append Factors | Name | Field name that displayed in the database |
|  | Label | factor name that displayed in watchman |
| Type | Type of factor, drop down selection |
| Default Value | default value |
| Index Group | same as the Index of database |
|  | Flatten column | To identify the pushed raw topic instance data, for example, after the policychangeid field is checked, you can see that the field is displayed in the raw topic in the database. |

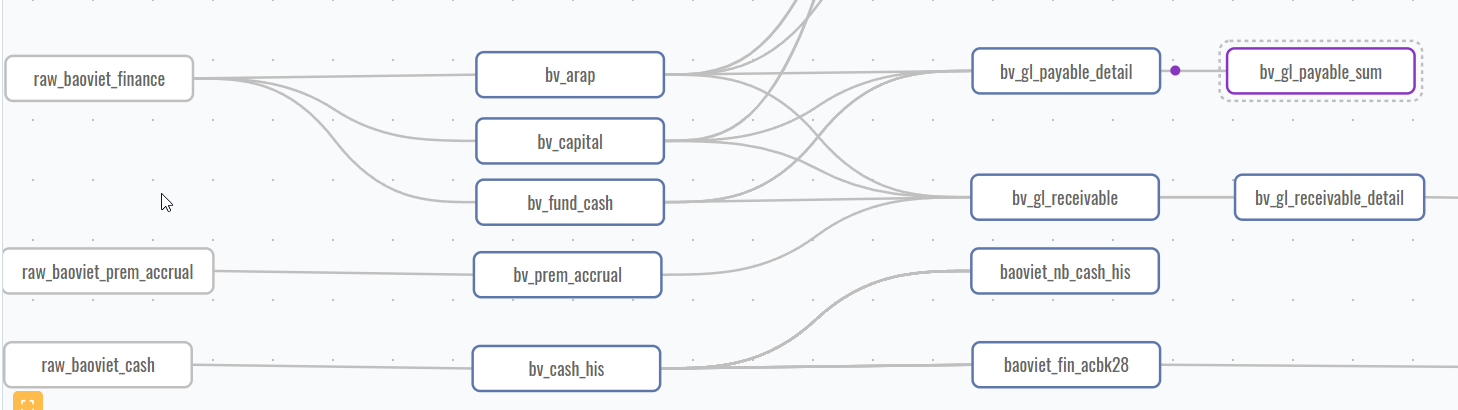
#### Create Enumerates

**This feature is temporarily not supported , you can use topic instead.**

## Pipeline

* Before using the Pipeline, the Topic table structure of the complete process must be established, and the Pipeline establishes the connection between the Topics.
* The design of Pipeline needs to be considered in the way of processing **a single record**

### Pipelines Panorama



The Pipeline Panorama page has the following features

Table 3‑2 Presentation form of topic

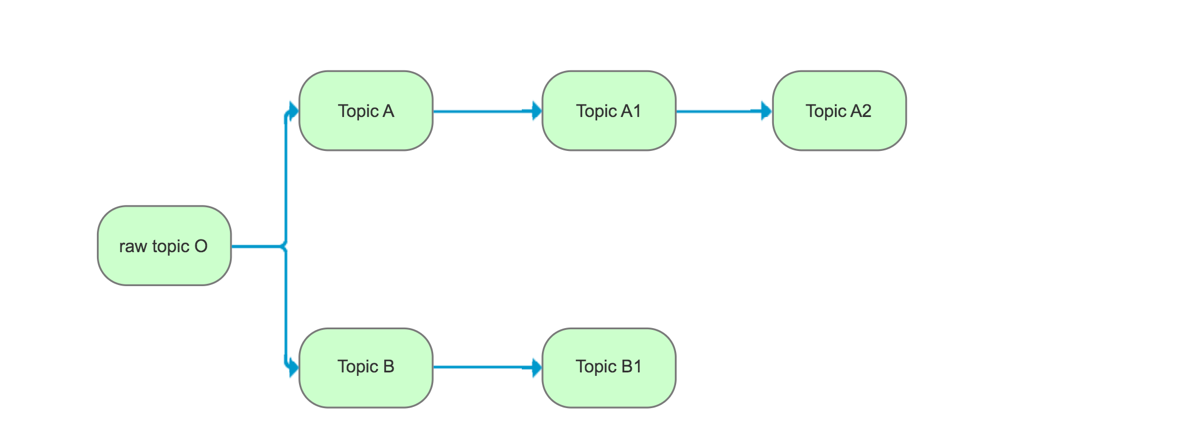
|  |  |  |
| --- | --- | --- |
| Displaying | Example | Description |
| Gray solid line frame |  | raw topic |
| Blue solid frame |  | Detailed topic |
| Purple solid line frame |  | Aggregate data in real time |

#### Link rules

Here we mainly introduce the connection rules of Topic and Topic, such as Topic time dependence.

* Topic time dependence

Problem Description:



Here, Pipeline OA/B is one pipeline. When pipeline A-A1 needs to write some factors from topic B, but the OB pipeline may not end, so the factors written in topic A1 may be empty.

Solution:

Determine whether there is such a dependency based on the correspondence between write/mapping and read. If there is such a dependency, wait until the OB pipeline runs to end, and then write the factors in Topic B.it means that OA and OB pipeline must be in a sequence with unit.

Rules:

* This situation is not supported when pipeline OB reads the factor of topic A2。
* The situation of different raw topics to read row : need to divide the data into different modules and push the data in different orders
* Subclass topic try to inherit the parent class or elder class, rather than directly inherit the parent class of the parent class
* Static data, that is, a data set that has not changed for a long time, should not be built in the detail or aggregation layer.

#### Pipeline page rules

Click raw topic/Create outgoing pipelines to enter the Pipeline definition page.

* The content of the outgoing pipelines and ingoing pipelines of the two topics connected to each other are exactly the same
* You can only create pipelines in the page outgoing pipelines.
* The pipeline naming rule A to B must be strictly named to facilitate testing and locating problems.

The following describes the basic structure of the pipeline.

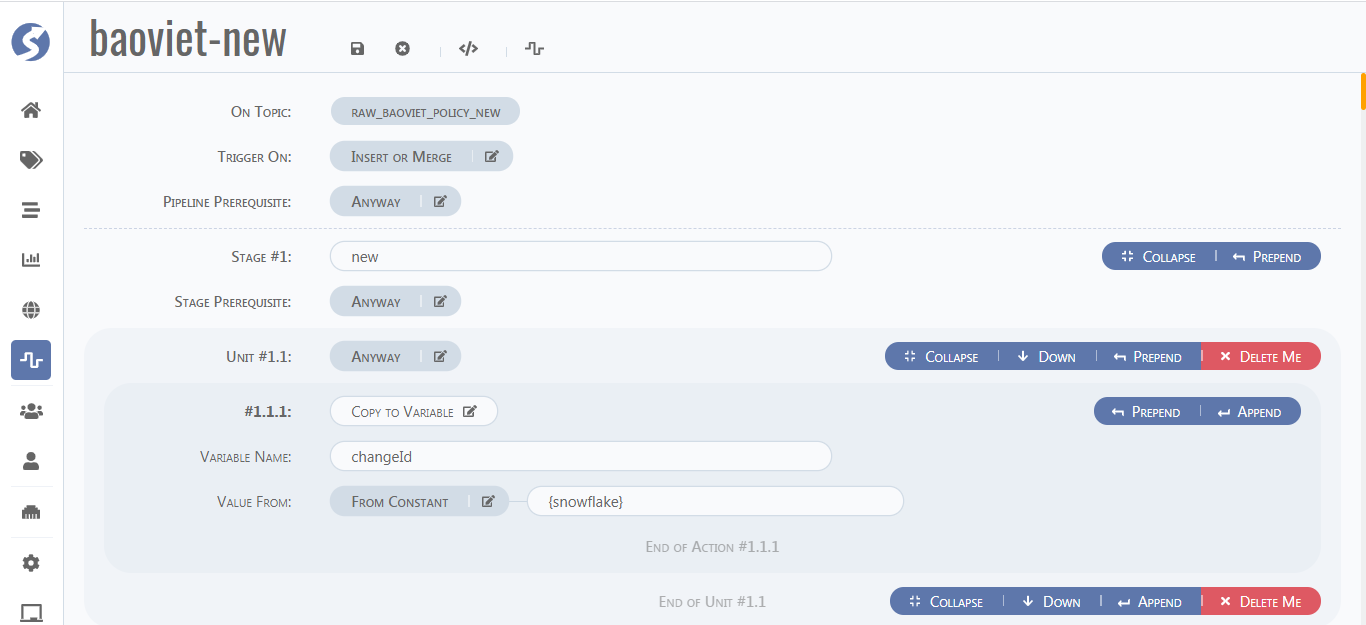


Table 3‑3 Pages roles

|  |  |
| --- | --- |
| Field | Description |
| Collapse | Put away the Stage or Unit |
| Up | Move up one position at the same level |
| Down | Move down one position at the same level |
| Prepend | Add a section of the same level to the head of this section |
| Delete me | delete section |
| Append | Add a part of the same level at the end of this part |

For the page in the pipeline, it is divided into three parts: stage, unit, and action. One stage has multiple units, and one unit has multiple actions. The criteria for distinguishing stages mainly depend on business requirements and requirements for code simplicity and legibility.

It is recommended to be divided into two types of stages:

* stage one，Read factors in non-parent topics and write repetitive functions for subsequent reference.
* stage two，Perform specific operations on records.

Before the start of each stage or unit or action, you can add prerequisite or filter conditions. Conditional functions are introduced in detail in the prerequisites section. The action part is to perform specific operations on the data that meets the requirements of this part, and the operation functions are introduced in the subsequent part.

### Header Function

The header section of the interface after entering the pipeline is as shown in the figure below.

Try to put **a small number** of filtering conditions in the header section, because the conditions in this part have two characteristics: first, it is for the whole world, and second, it does not support the addition of judging existence and reading factor as conditions in this section.

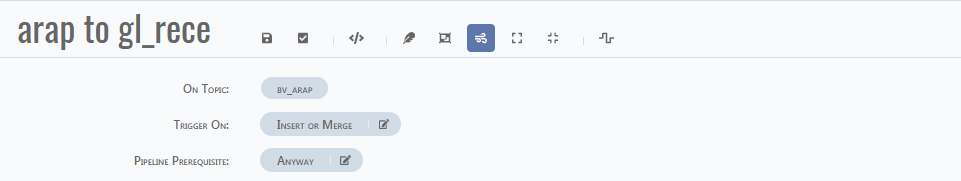


Table 3‑4 Header Function

|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| Save pipeline | / | Save the pipeline, but it does not take effect. |
| Enable pipeline | / | **Save the pipeline and take effect** |
| View in DSL | / | View pipeline content in DSL language |
| Focus on Unit | / | View pipeline content by unit |
| Focus on stage | / | View pipeline content by stage |
| Free walk | expand all | Expand the content of all organizations |
|  | collapse all to units | Collapse all organization content |
| back to Catalog |  | Return to the main pipeline page |
| On topic | / | Topic that is acted on |
| Trigger on | Insert or Merge | The trigger action for the acted topic is insert or merge. In other words, if the action in the original topic connected by the pipeline is inert, the pipeline will be triggered |
|  | Insert | The trigger action for the acted topic is insert. |
|  | Merge | Update |
|  | Delete | delete |
| Pipeline prerequisites | Anyway | default situation |
|  | And | The joining conditions have a screening effect on the overall situation and satisfy |
|  | Or | The joining conditions have a screening effect on the overall situation, and only one of them can be satisfied |

### Prerequisites Function

* Prerequisites Function can be added in the pipeline header section, stage, and unit. Specifically, the location and content of prerequisites can be determined according to the size of the data operation range in the subsequent parts.
* Note that the stage condition only applies to the defined stage, and does not constitute a constraint on other stages

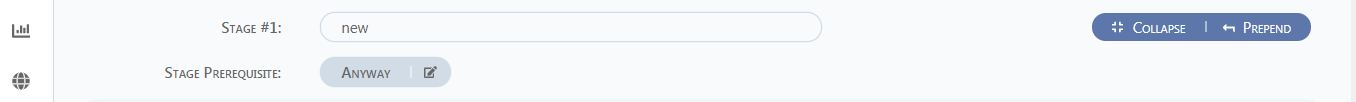


Table 3‑5 Prerequisites Function

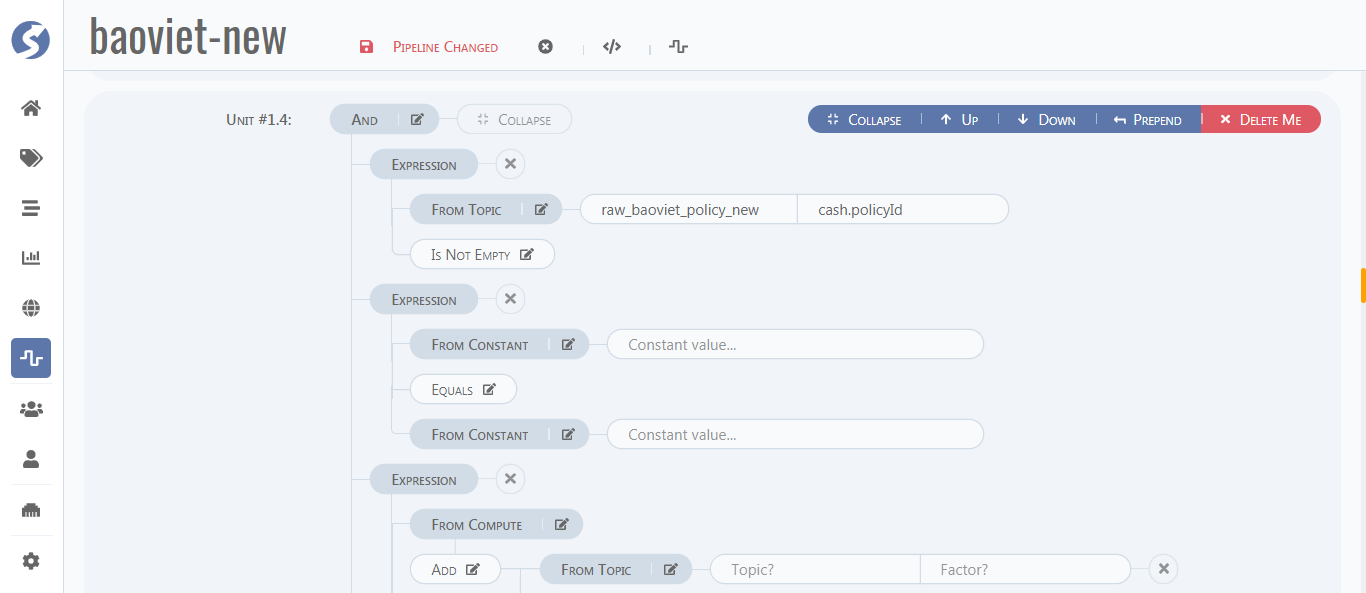
|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| STAGE #1/2/3 | / | / |
| STAGE prerequisite | Anyway | default |
|  | And | The joining conditions have a screening effect on the Stage, and satisfy at the same time |
|  | Or | The joining conditions filter the Stage and satisfy one of them |

##### And/or

The **And** function means that the relationship between the expressions it contains must be established at the same time.

The **Or** function means that only one relationship between expressions contained in it is established.

The **And** function and the **or** function can be nested infinitely, but it is recommended to configure the filter conditions in the simplest form, that is, to merge similar items to reduce the amount of reading.



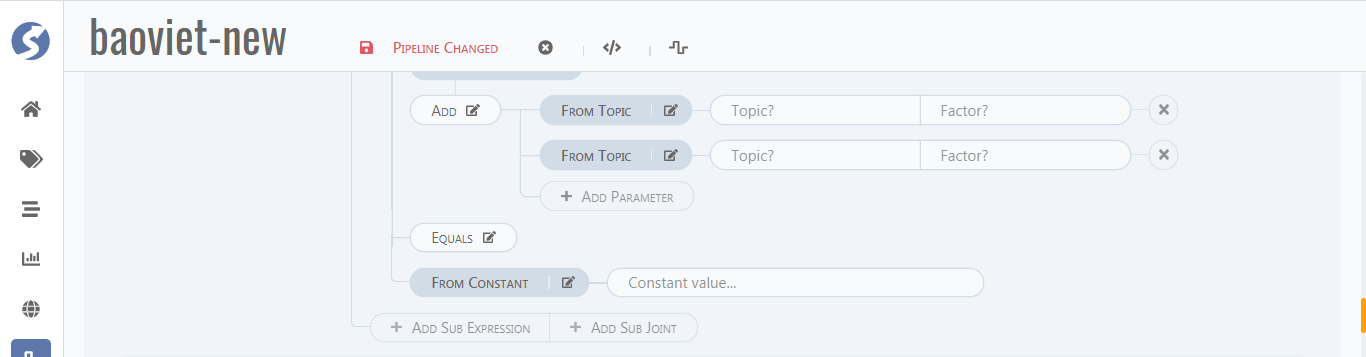


Table 3‑6 And/or funciton

|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| Add sub expression | / | Add one expression |
| Add sub joint | / | Add multiple expressions |

##### From topic /constant

You can use the expression 'From constant 1 = From constant 2 'as a general comment rule.

Table 3‑7 From topic/constant

|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| From topic/constant | Is empty/is not empty | Determine whether the field is empty |
|  | Equals/not Equals | / |
|  | Less than/less than or equals | Only supports numeric fields |
|  | Greater than/Greater than or equals | Only supports numeric fields |
|  | In /not in | Is it in a certain set |

##### From compute

Table 3‑8 From compute

|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| From compute | Add/Subtract/multiply/divide | Only supports numeric fields |
|  | Modulus | Only supports numeric fields |
|  | Year of /half year of/quarter of/ month of/ week of Year/  Week of Month/ day of month/Day of week | Only supports date fields |
|  | Case then | **Anyway** is the condition of the case, and then is the subsequent rows, and every two rows are one case, and the data of the case can be operated when one is satisfied. |

### Action Function

* Each stage can contain multiple units, and each unit corresponds to multiple action functions.
* Action function is the main processing part of data.

The following parts are all operation functions.

Table 3‑9 Action Function

|  |  |
| --- | --- |
| Action Function | Description |
| Loop | Loop action operation, you can split and flatten the array factors |
| Alarm | / |
| copy to Variable | Copy the value of the field and save it as a variable that can be referenced. |
| Exists | Determine whether a variable exists, the return value is true or false |
| Read Factor | Read topic factors of non-parent and non-subclasses, which are global variables in the pipeline |
| Read row | Read a single line and save it as a variable |
| Read Factors | Read a factor in the form of a single list |
| Read rows | Read multiple rows and save them as variables |
| Write Factor | Write a factor that has been read |
| Insert or merge row | Insert or merger row by some conditions |
| Insert row | Insert a row to target topic |
| Merge row | merge row by some conditions |

#### Loop function

The loop function only supports the use from the raw topic to the first level topic. we usually use the action **copy to variable** to make the list factor to be a variable to loop.

Table 3‑10 Loop Function

|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| Loop variable name | / | If the variable is a list, all actions related to the variable will be looped under the unit; if it is a character, it will loop only once. |

* Common situation, Factor has no list

Input a row，select the Factor , loop just once

* Common situation, Factor has lists.

Input a row, select the listed Factor, loop length of list, perform action.

* Read a listed Factor.

Select the Factor, loop length of list, perform actions.

* Read a no-listed Factor.

Loop just once.

* Read a Row that one of factor is a lists

Not support.

* Read a Row that has no lists.

Loop length of list, perform actions.

If the variable is filled in, but the variable is empty, then all actions will be automatically skipped.

#### Alarm

Provide an alert when the value is over/under the limit.

#### Read Factor/Row

Read row: Read and save a whole line. Read Factor: Read and save a column

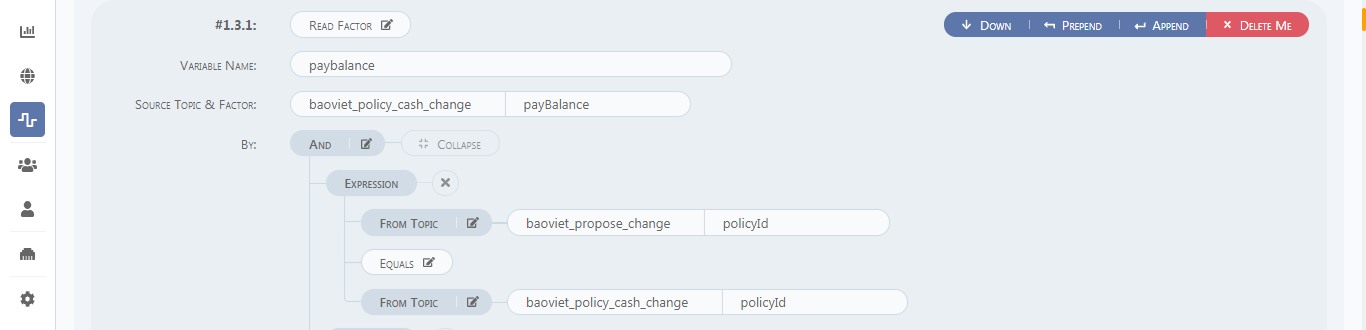


Table 3‑11 Read Factor/Row

|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| Variable Name | / | Save the source Factor as a new variable, which is a global variable. Write Factor can refer to the variable. |
| aggregate |  |  |
| Source Topic&Factor | / | Read the source of Factor |
| By | / | Add matching conditions. You can see [Condition function](#_Condition_function) for notes. |
| And | same as Unit section configuration | / |

#### Insert or Merge row

* Insert Row and Merge row configuration rules are similar to Insert or Merge
* Insert or merge row is to insert a row of data including the specified factors.
* If you need to update the data in the table in real time, you can choose Insert or Merge row. If you don’t need to update the data in the table and just insert data, you can choose Insert row.

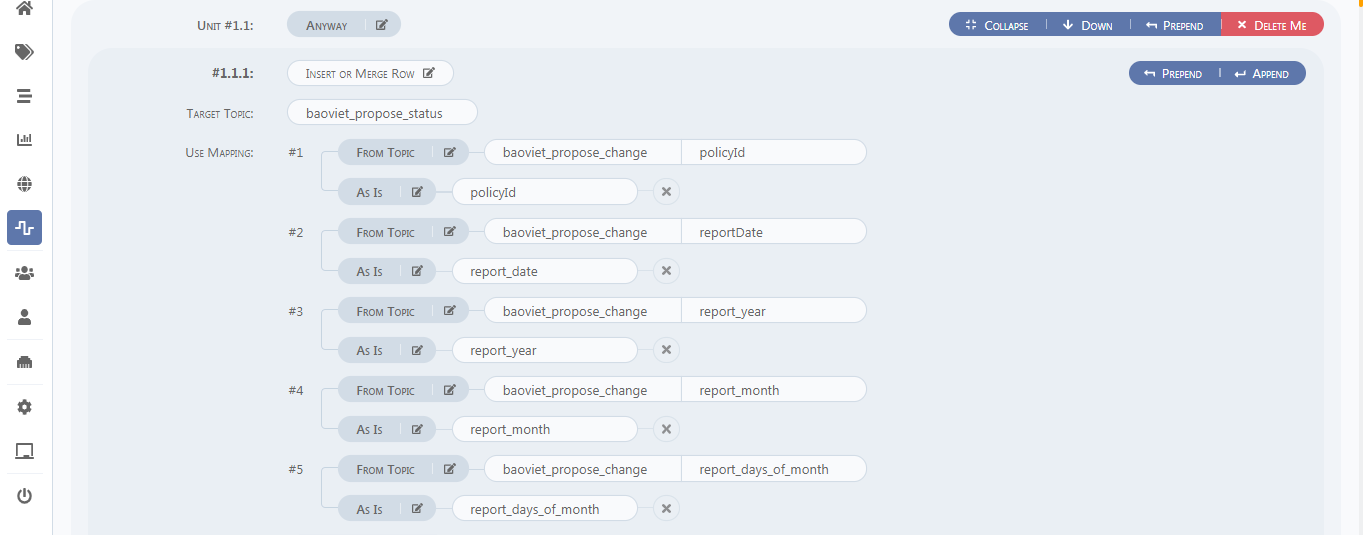


Table 3‑12 Insert or Merge row

|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| Target Topic | / | The name of the target topic to be written |
| Use Mapping | / | Map Topic fields or constants to Target Topic factors |
| From Topic/constant | / | / |
|  | As is | The name of the target topic's factor |
|  | Sum/ Avg/count | / |
|  | MAX/Min/Median | / |
| From Compute | / |  |
|  | same as And / or | / |

Note

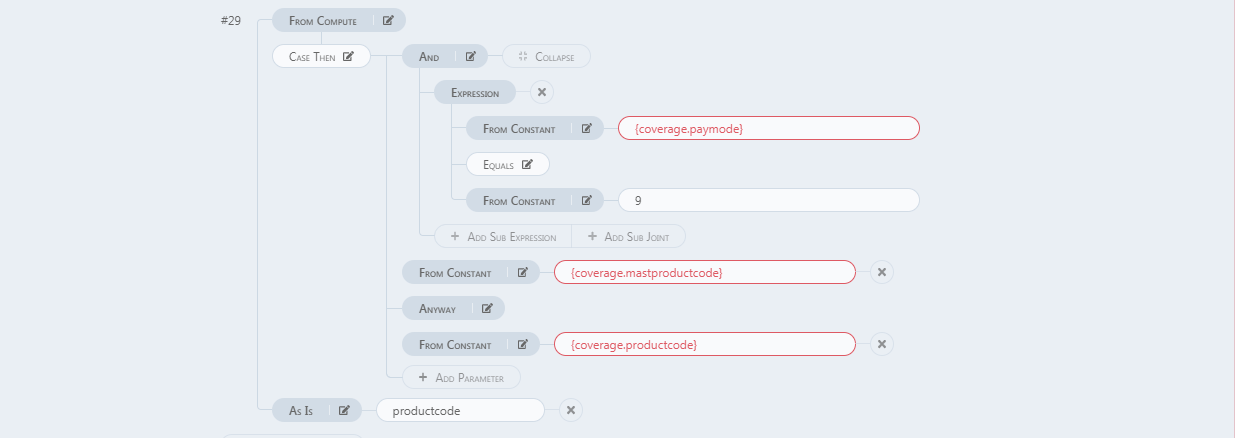
* For Insert or merge row，you should add **all** factors you want in action parts, it means that every insert or merge is **for one record**.
* For merge row ，you should add factors that you want to update . it means that every merge is just the **part of one record**.

1. Read Row and case then

Use the Read Row function in stage 1.unit 1.1 to extract the coverage field in baoviet\_pa\_coverage based on the itemid in baoviet\_pa\_coverage and bv\_gl\_receivable.finally,store it as a variable named coverage.



In stage2, the purpose is to map the fields of the source topic to some fields in the target topic under certain conditions. The CASE THEN function will be used here, that is, a value will be mapped to the target field under certain conditions. The following specific example shows that if the paymode field of the coverage table is equal to 9, then the mastproductcode in the coverage table is mapped to the productcode in the target topic table, otherwise the productcode in the coverage table is mapped to the productcode in the target topic table.



#### Write Factor/Row

Write row: Write a line. Write Factor: Write a column

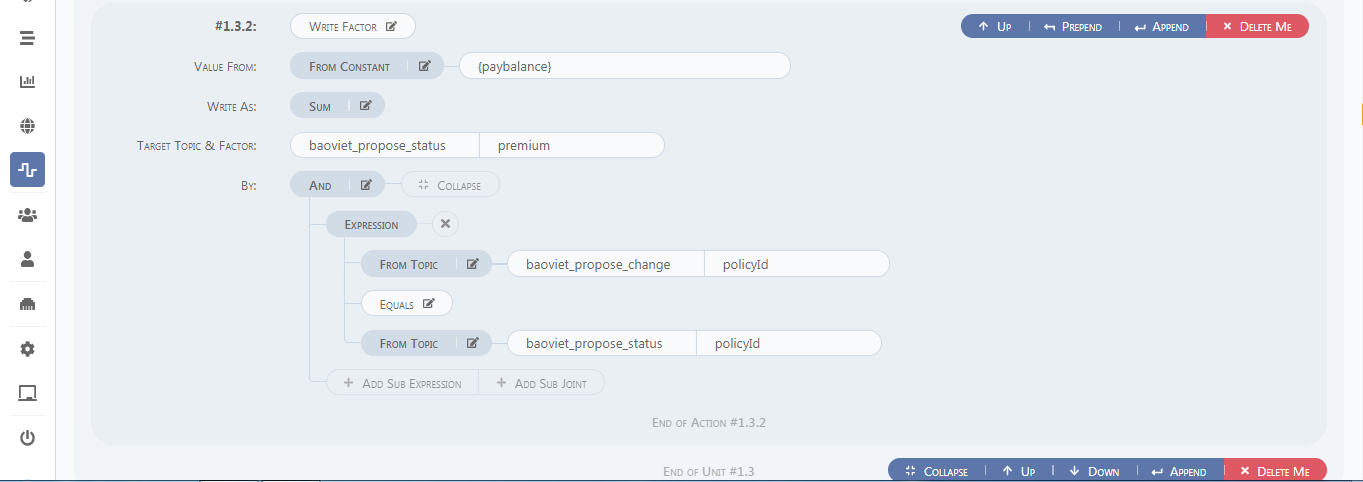


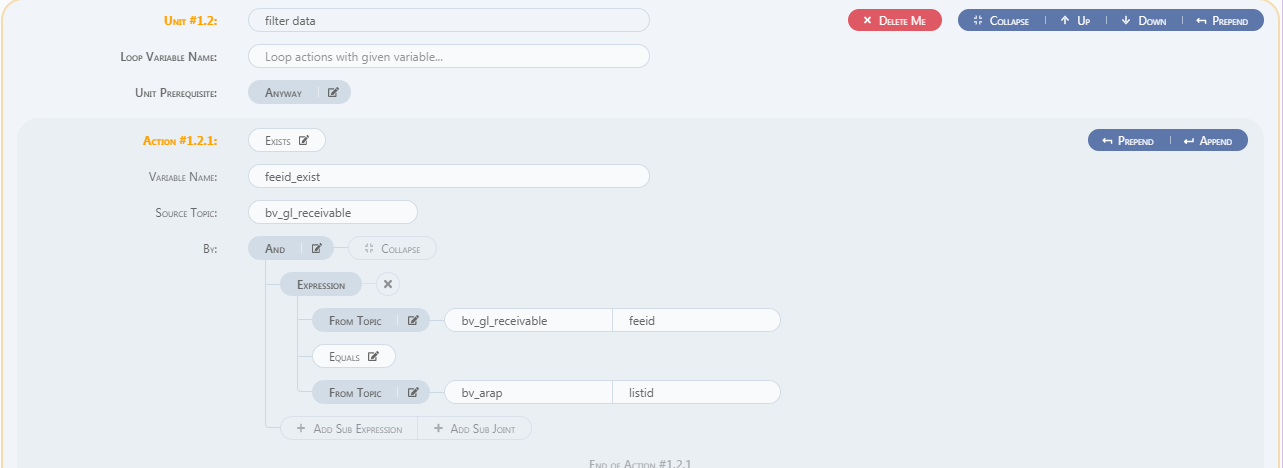
Table 3‑13 Write Factor/Row

|  |  |  |
| --- | --- | --- |
| Field | SonField | Description |
| Value From | From constant | Constant value or variable saved by Read Factor |
|  | From topic | Factor of a specific topic |
| Write as | As is | as the target factor |
|  | Sum/ Avg/count | Sum/Avg/count a certain field in the record each time |
|  | Max/Min/Median | Max/Min/Median a certain field in the record each time |
| Target Topic & Factor | / | / |
| By | same as And /or | / |

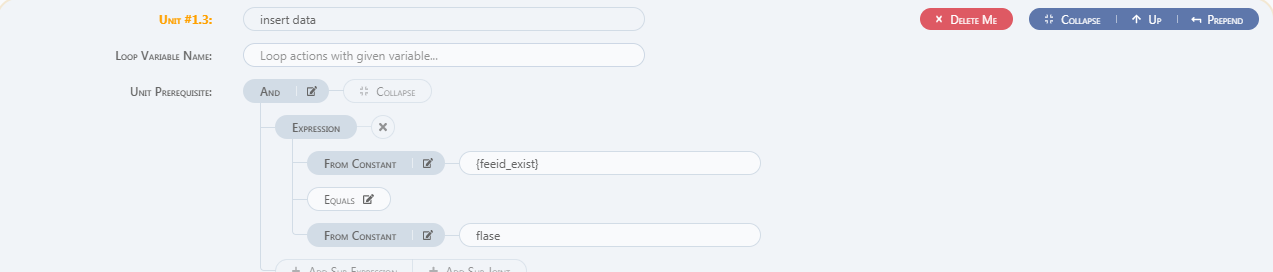
#### Exists

Determine whether the variable of the source topic exists, that is, the value of the corresponding variable generated by the action is True or False, and the variable needs to be quoted later.

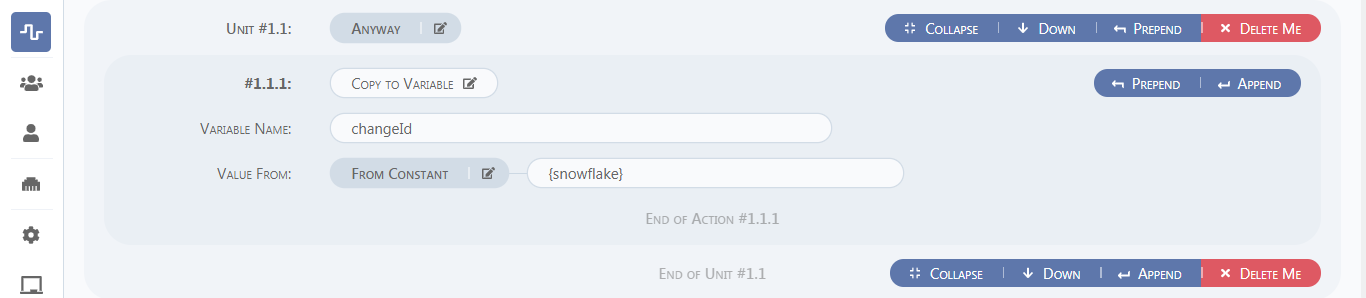
* Configure Exist variables



* Quote Exist variables



#### Copy to Variable



* Copy a field or array or value to a new variable and display it by value, as shown in the figure, the sequence value changeId is generated

#### Special function

{&nextSeq} function generates IDs sorted in sequence.

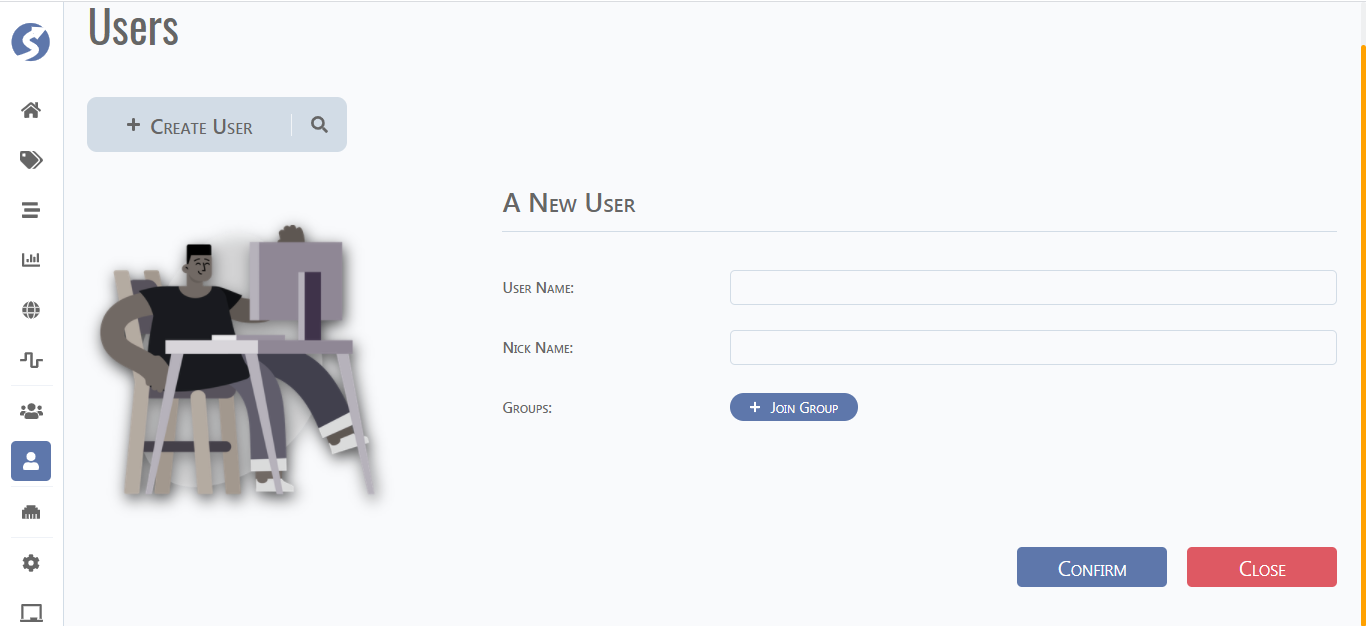
#### Condition function

There are two filtering functions in the pipeline. The first is **Prerequisites Function**, which means that the data is being calculated in the memory and has not yet entered the storage medium(database). The second is the **by** function in **Insert or merge** or **read factor**. After the by function, a filter condition is added. The query engine queries or reads data from the storage medium(database) according to the condition. Therefore, you need to pay attention to the placement of the condition when adding the filter condition.

## Space

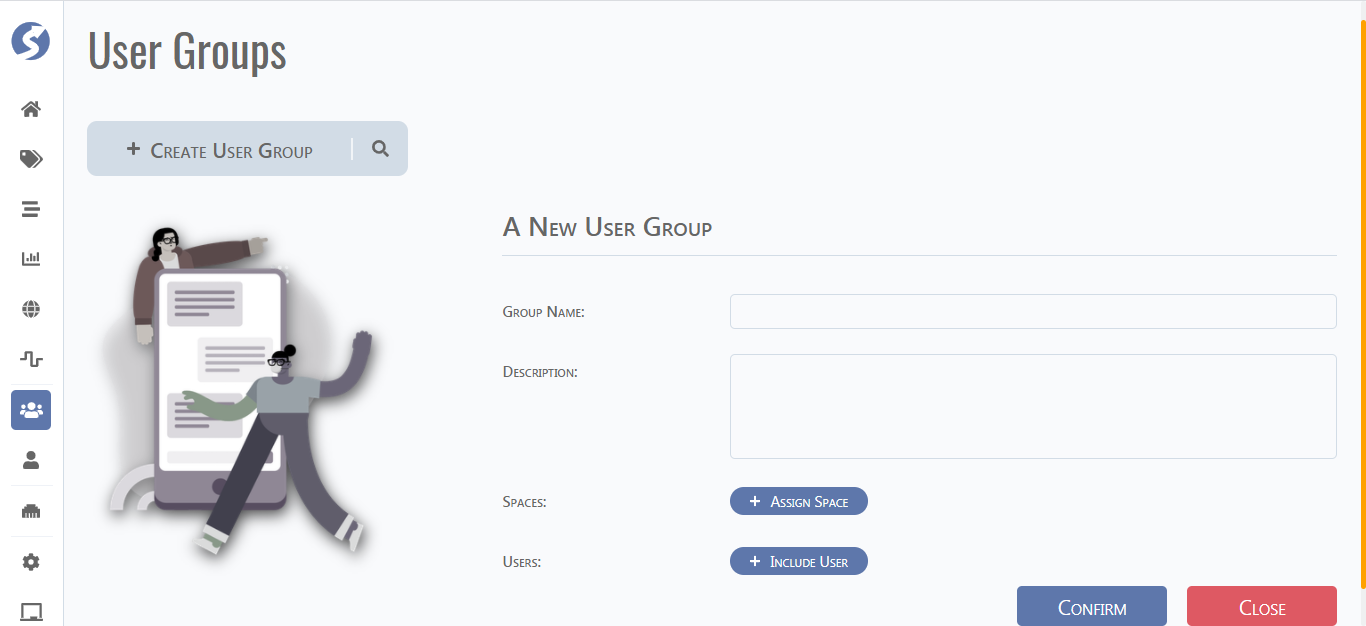
Space is mainly to create a business module, select specific tables suitable for business use, and give specific users access rights. the console user only can use the space defined by admin.

1. Create user



|  |  |
| --- | --- |
| Field | Description |
| Join Group | Add the newly created User to the Group. |

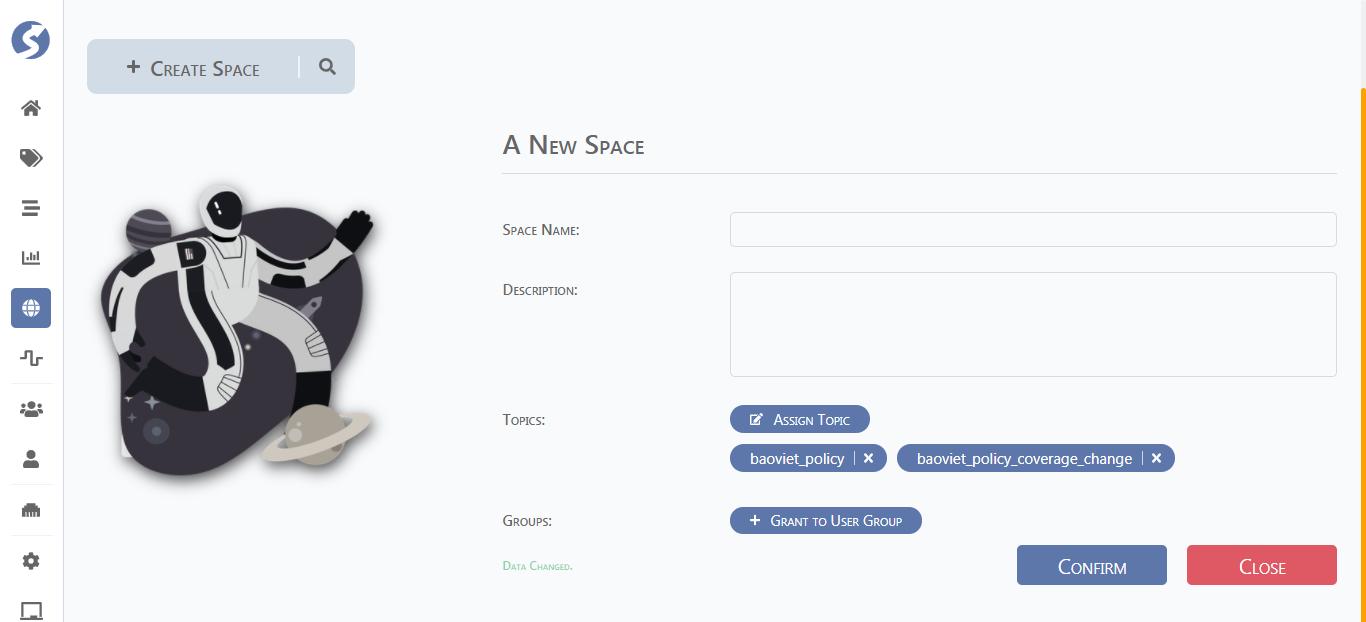
1. Create group



|  |  |
| --- | --- |
| Field | Description |
| Include user | Add newly created User |
| Assign Space | Assign the Group to the created space |

1. Create space

In this step you need to select the topic you need.



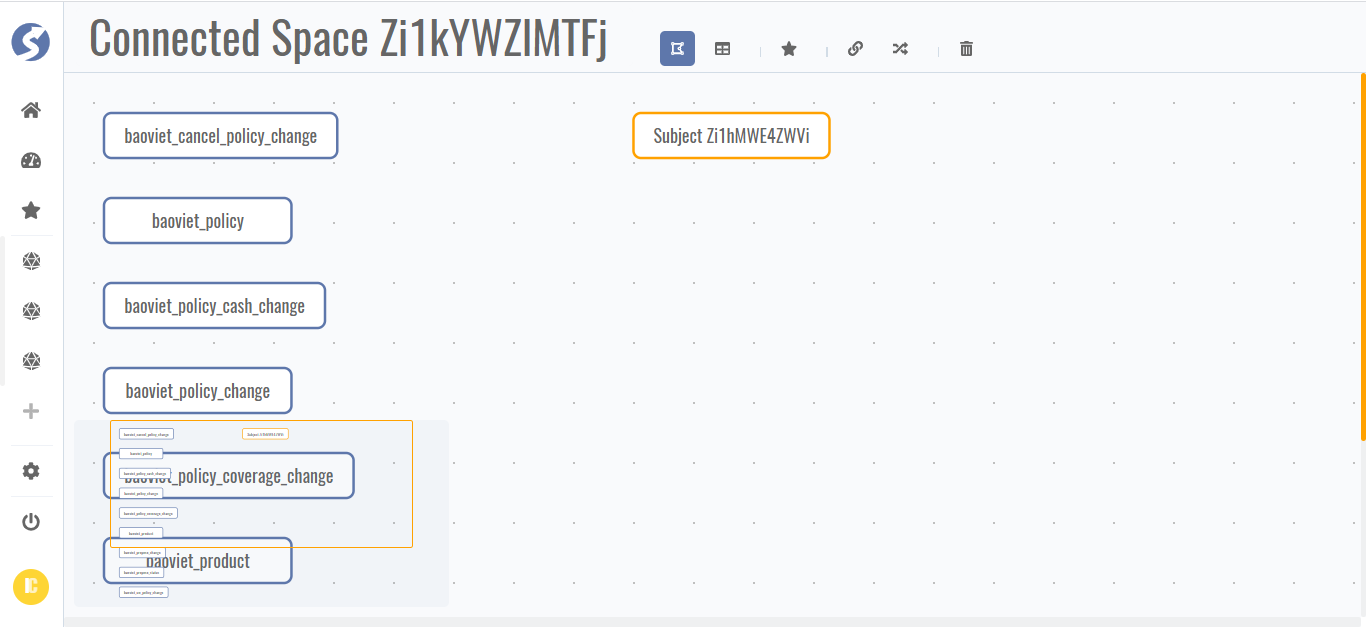
|  |  |
| --- | --- |
| Field | Description |
| Assign Topic | Add configured Topic |
| Grant to user Group | Add established Group |

## Workspace

Use the Console account to log into the workspace to perform follow-up operations on the established Space. **Note that** only the user account in the user groups can log in to the console. and then Click on connect space on the home of the console. workspace is mainly for final data filtering in the face of customers.

### Dataset

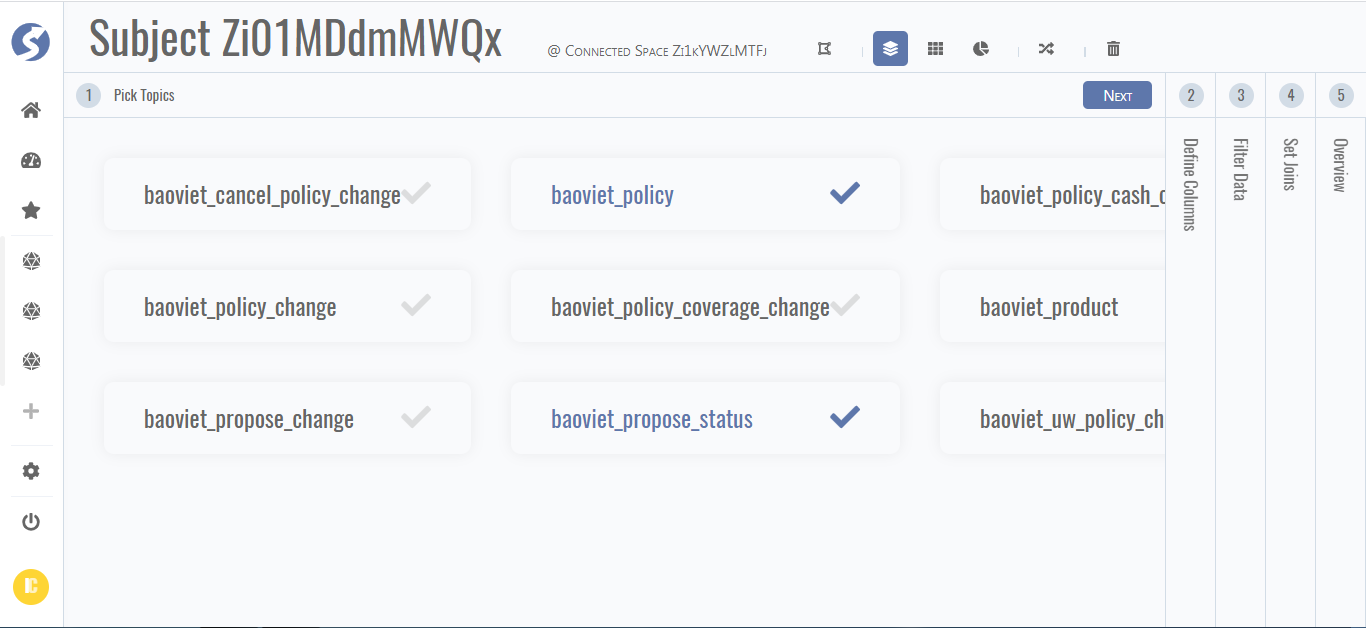
Click on add subject and show topics that set in the space section.



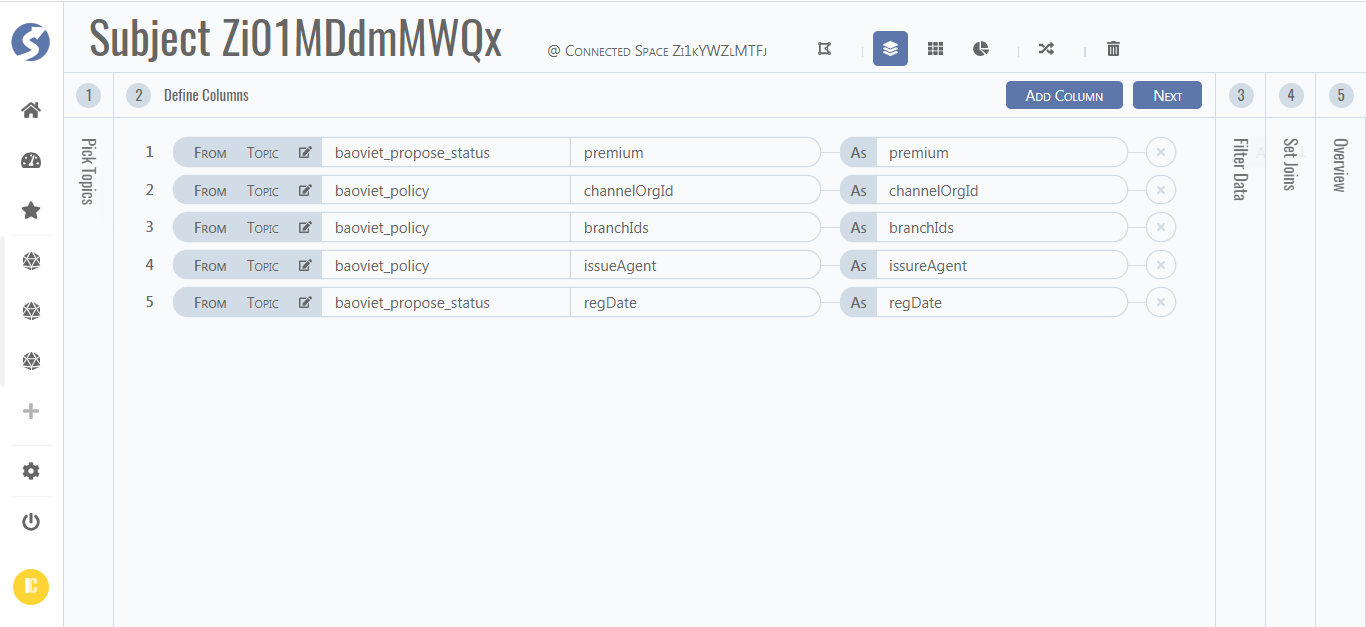
|  |  |
| --- | --- |
| Field | Description |
| Add subject | Configure dataset |

The following are the steps to configure the Dataset:

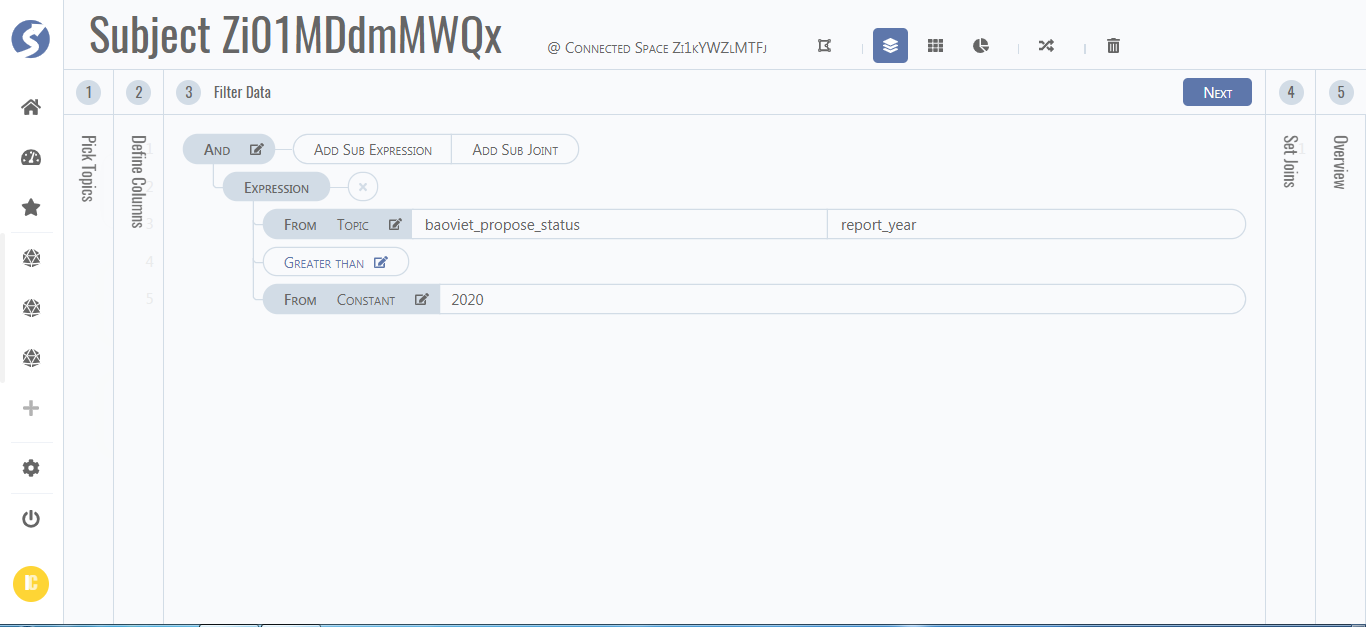
1. Pick Topics



1. Define columns



1. Filter Data



1. Join dataset



1. Overview



Now you can click 1 to see datasets or click 2 to create report.

### Report

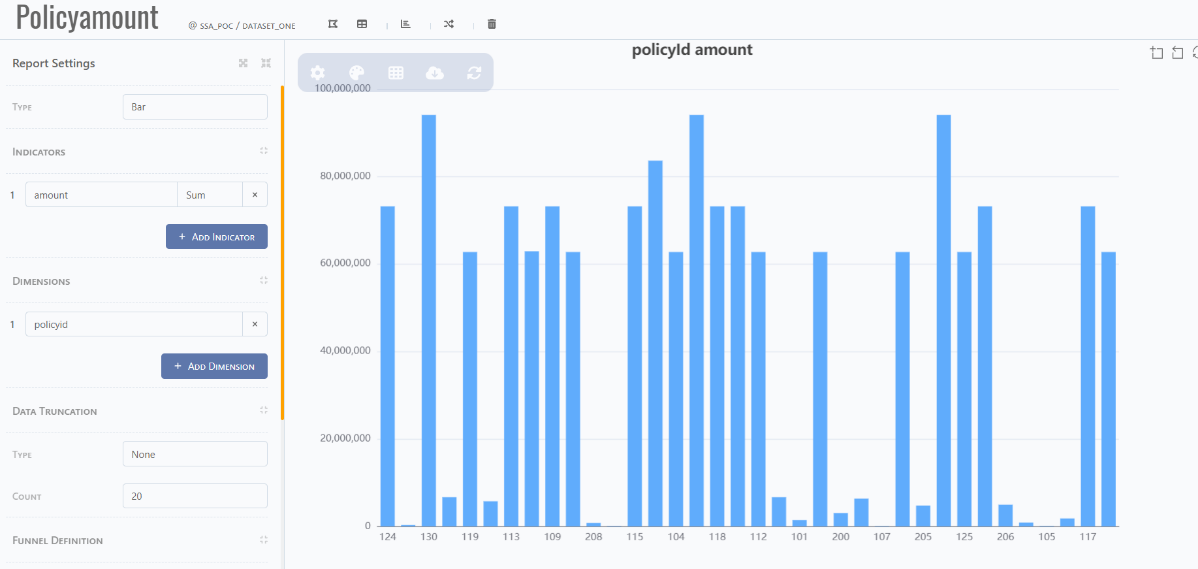


Table 4‑1 Report description

|  |  |
| --- | --- |
| Field | Description |
| Type | Single value, Bar, line, pie, scatter. |
| Indicators | Similar to Y axis |
| Dimensions | Similar to X axis |
| Data Truncation | / |
| Funnel definition | / |

## Testing process

Testing during the data development cycle: In the data development cycle, the **Simulator** should be the main test for pipeline and topic, due to various considerations:

1. Decouple from the database, do not need to rely on the database to complete the test work.

2. For pipeline and topic testing during the development period, data logic should be the main focus, and it needs to be "short and fast"

3. Control the scope of testing and complete the pipeline testing in a modular manner, instead of testing end to end.

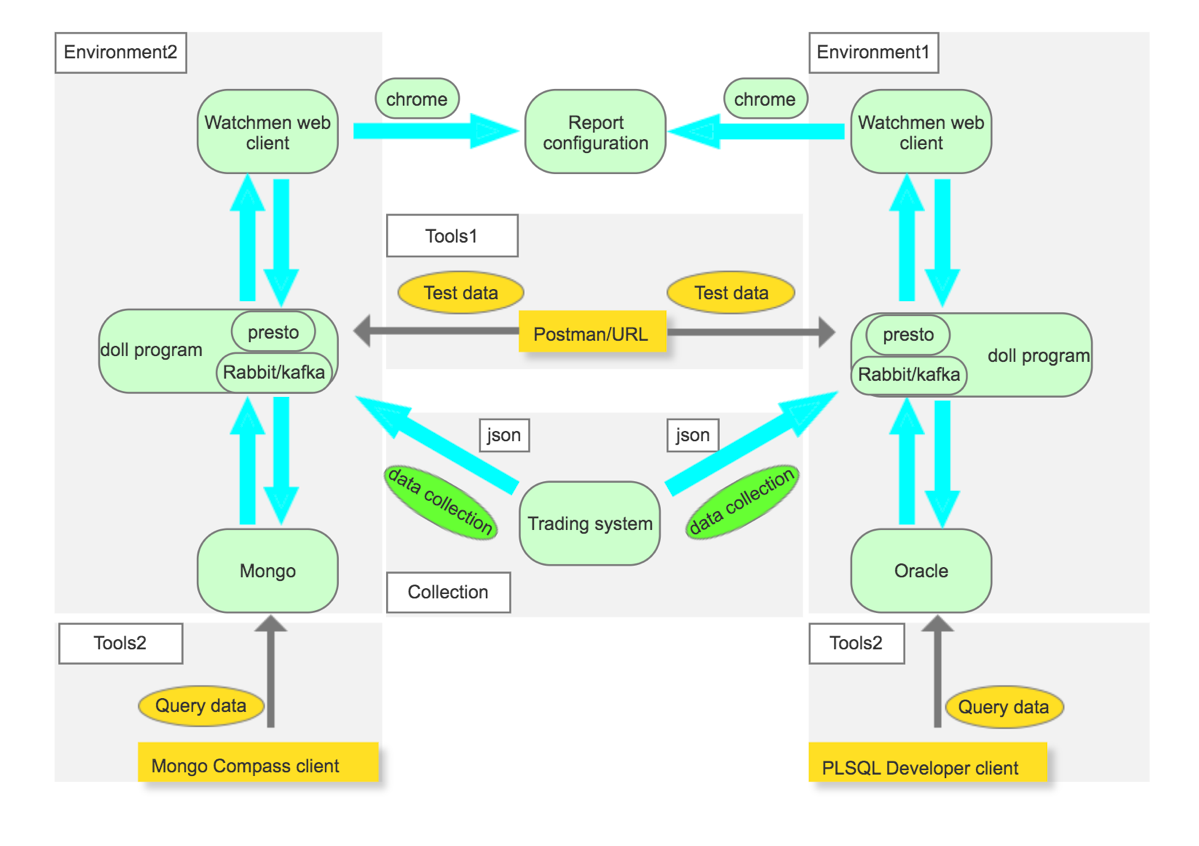
4. When the development cycle goes to the entire E2E test, it is necessary to introduce technical personnel (requires release and deployment) to assist, but it is still led by business personnel

5. At present, if the E2E test is to be completed independently, SQL technical background is required

In the figure below, the process of testing the correctness of the data is mainly described. There are three ways to test data:

* Use Postman to post the json file provided by the collection module staff to the watchman interface, and verify the correctness of the data in the corresponding database client.
* Use the URL provided by the collection module staff for posting, and the correctness of the data can be verified in the corresponding database client.
* Use the Simulator function inside watchman to post the json file provided by the collection module staff and verify it inside watchman.

Figure 3‑5 Report Testing process



Tools2 in the figure above is a common database management system, such as Oracle, mongo db. tools1 is the software postman for pushing data.

Collection is the process of extracting data from trading systems or other data sources. we can also use the Rabbit or kafka MQ to implement the function of collection.

Presto is a module that determines whether to install or not according to needs. Presto can implement interactive data query functions in the console module of data plat.

Of course, the method of using postman is still too cumbersome. It is recommended to use the Simulator function embedded in watchman. Before using it, you need to prepare test data, and then test directly in watchman.

## Data patch

Production operation and maintenance DP(data patch):

Production operation and maintenance need to be handled according to scenarios:

1. The method and content of DP need to be determined by detailed human judgment analysis

2. For scenarios where data processing is interrupted, subsequent pipeline operations can be completed through the platform interface

3. For scenarios with data errors, the topic can be updated and modified separately through the platform interface (the pipeline will not be triggered). It is also feasible to write SQL updates directly in this scenario, depending on the project team personnel.

4. In the case of a large amount of DP data, it must be processed in batches. The processing logic needs to be placed outside the data platform and designed separately, but it is still updated through the platform interface. It can be understood that there is a DP client to connect to the interface of the data platform to complete the DP (similar to Collector), and the data platform maintains the stream processing mode.

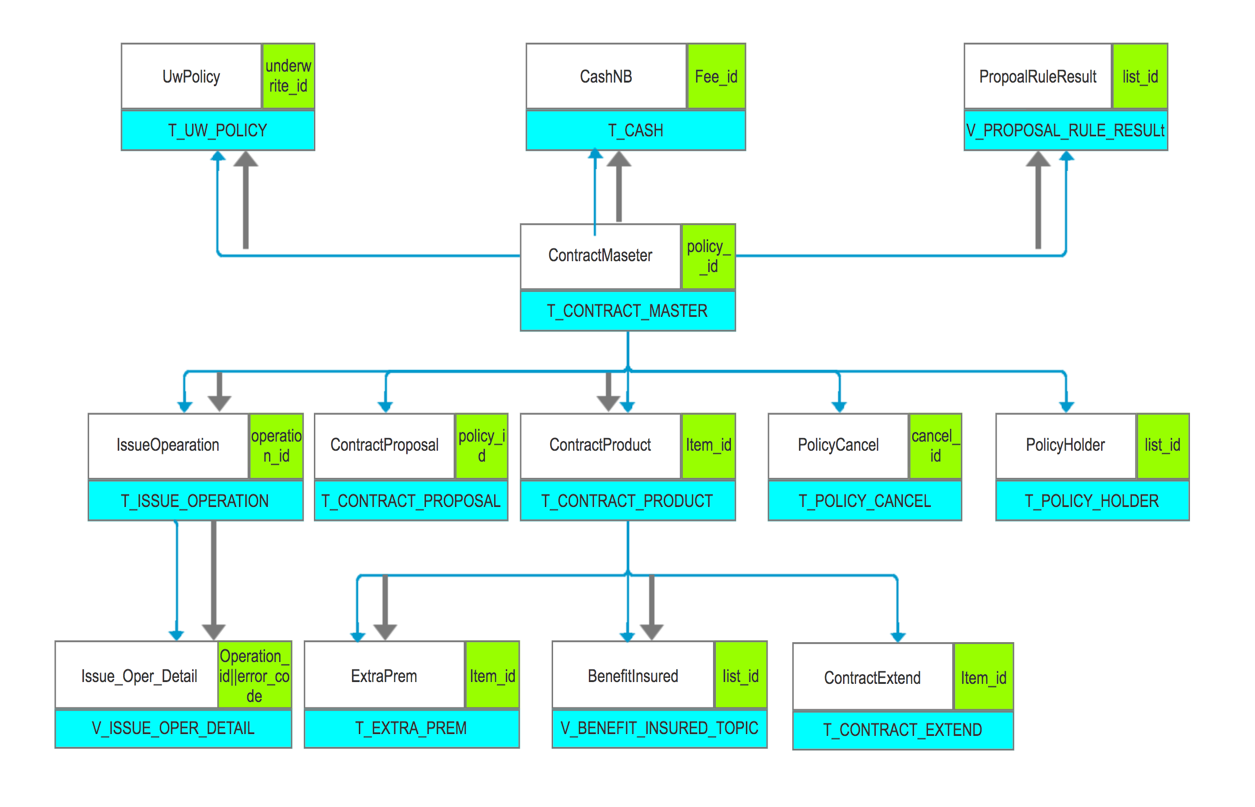
5. It is necessary to add conventional resources and processes to ensure the data quality of the data platform and cannot be left unattended.

# Baoviet project

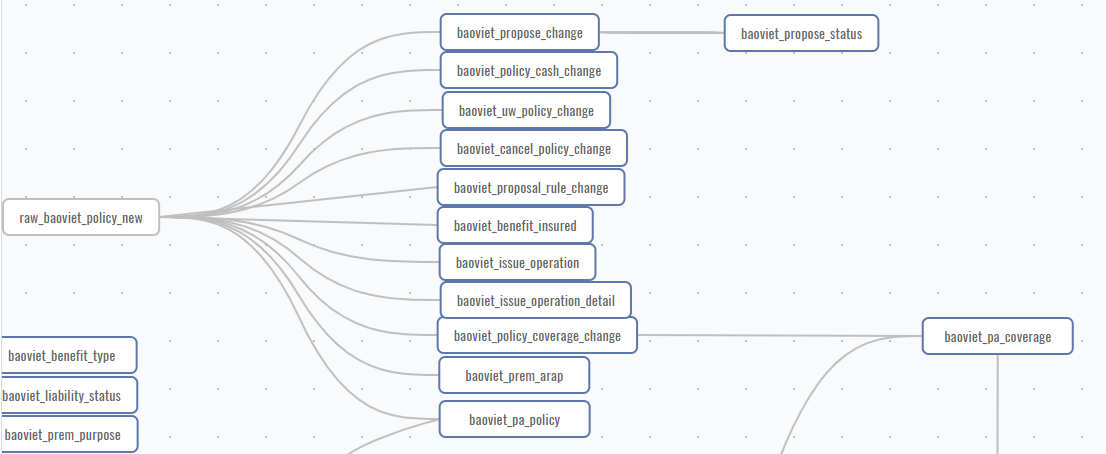
## Baoviet Topic Introduction

Here we mainly introduce the form of the table (**Topic**) structure from the trading system to the data platform in the baoviet project.

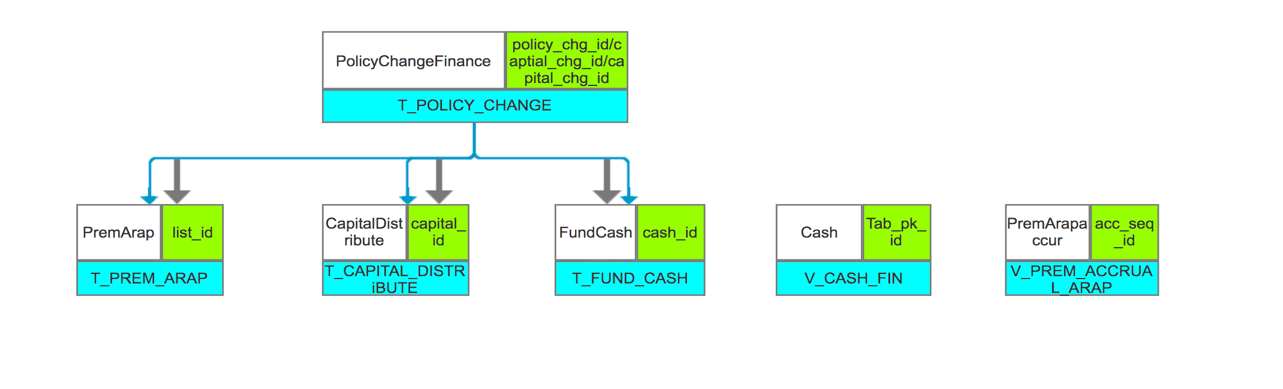
### Proposal module

The following figure shows the structure when data enters the raw topic, that is, the data table structure that the collector pushes into the raw topic. Cyan square is a table in the trading system database, the white square is the table name in the input json, and the green is the primary key of the table. The black arrow indicates one-to-many.

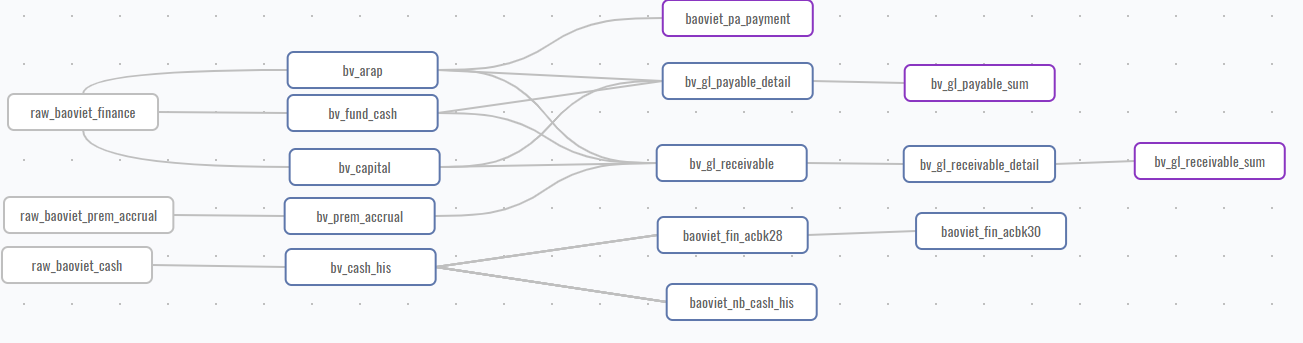
The following table shows the topics displayed on watchman. You can find that the levels of each table are at the same level. For example, the topic of baoviet \_propose \_status is the second-level topic for the business. the table of baoviet \_ propose \_status is designed for different states of proposal insurance policy.



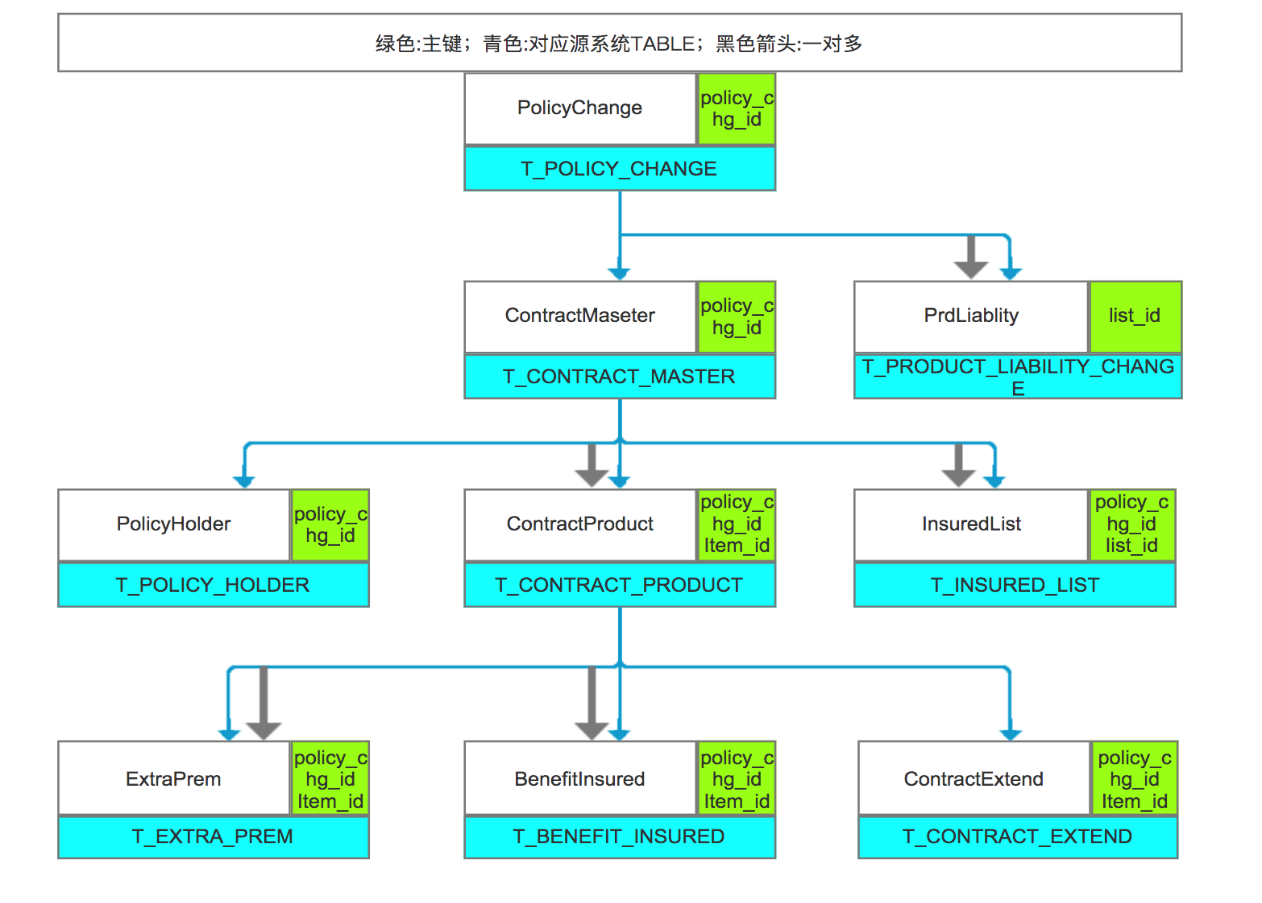
### Finance module



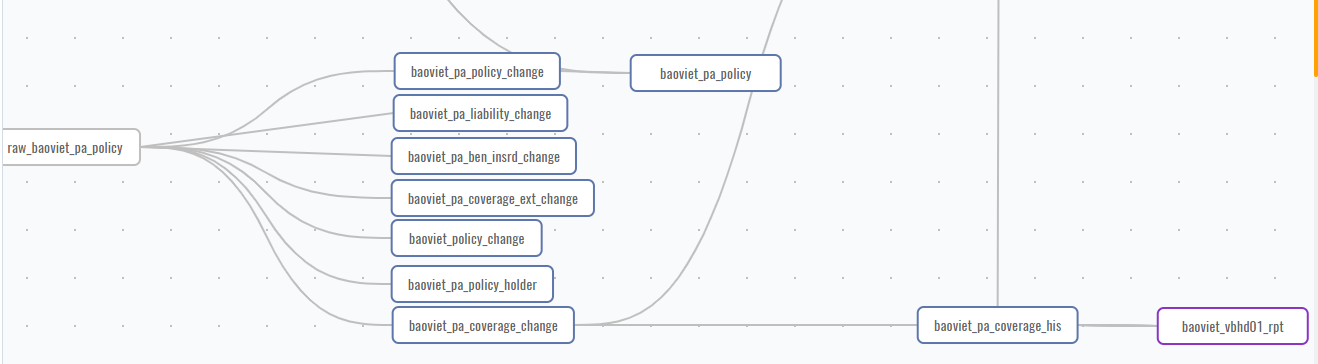
The main feature of this module is that there is a many-to-many situation between the second-level topic and the third-level topic.



### Policy change module



The biggest feature of the PA module is that some of its topics will intersect with the proposal and finance modules.

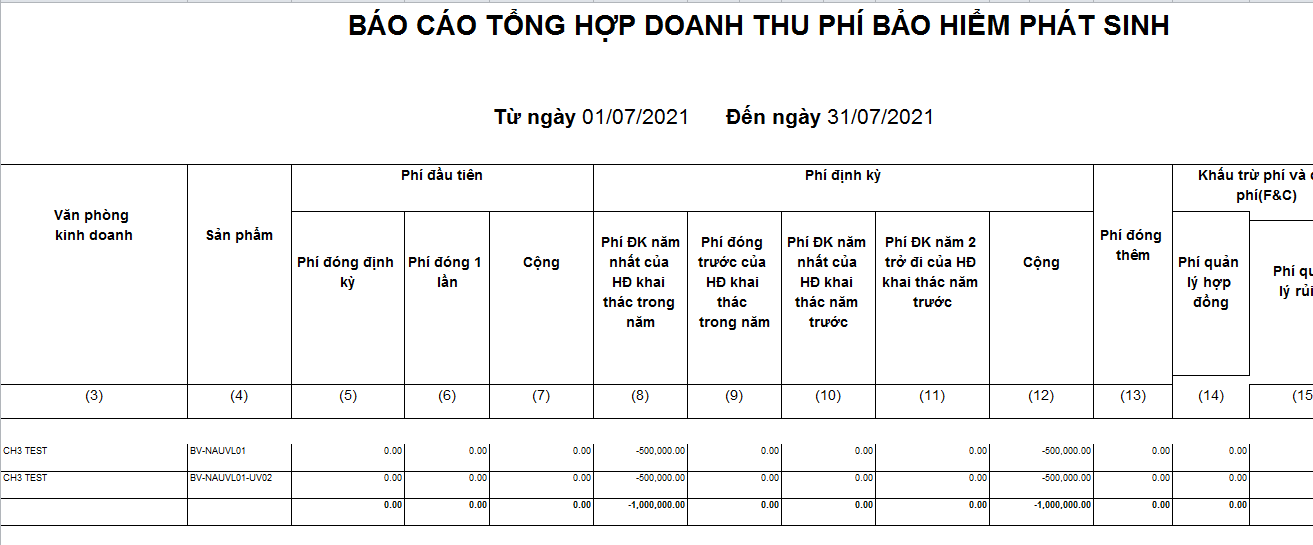


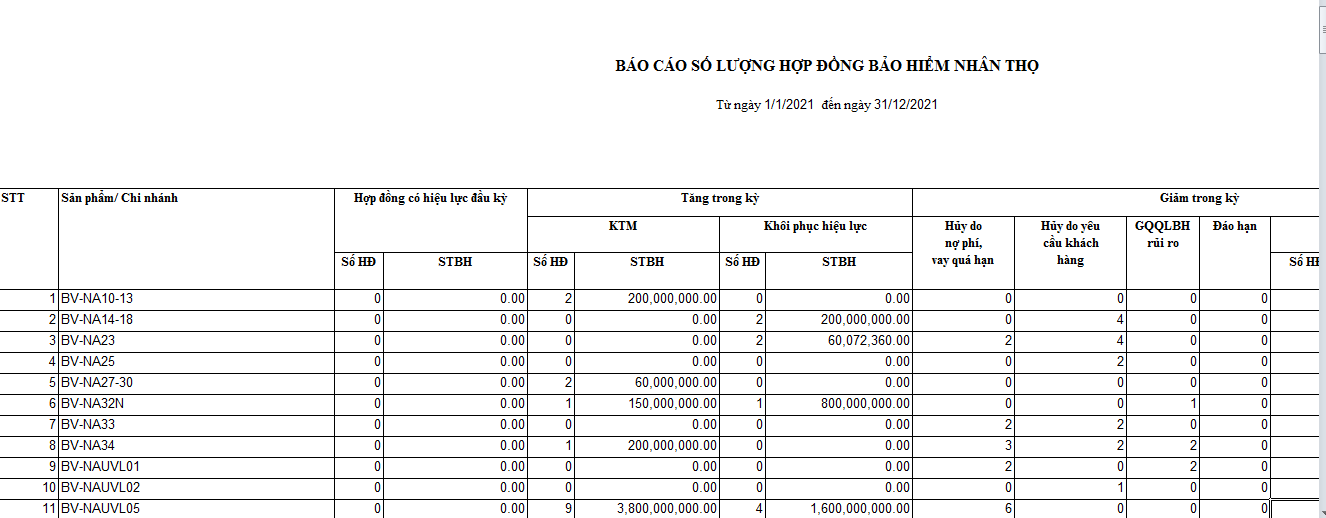
## Baoviet finance module Introduction

**To be supplemented.**

## Baoviet Report Introduction

Since baoviet currently uses crystal report, the following briefly show the generated table. These two tables come from the bv\_gl\_receivable\_sum and bv\_gl\_payable\_sum of the finance module.





If you have any questions, please email me:

conrad.wang@ebaotech.com