ONE - Workshop

DQ rules & DQ checks

Prepared for:	v15.4
Prepared by:	Ataccama
Dated:	November 2024

Contents of the Document

Introduction

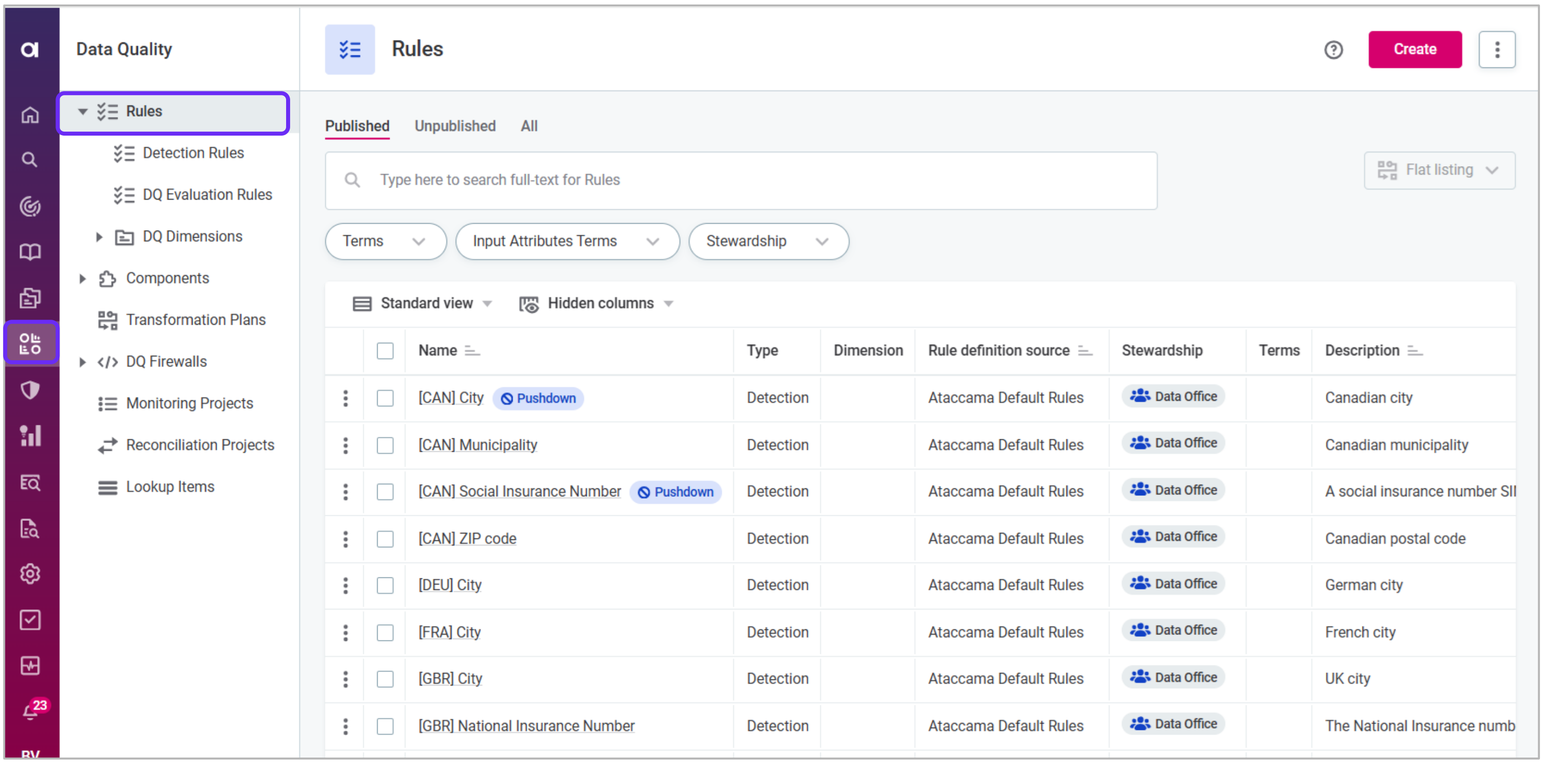
In this ONE workshop, you will focus on Data Quality Evaluation rules. You will create the rules in the ONE Web application, implement them, apply them to Catalog Items, and observe the results.

Tasks

Ataccama ONE already contains a wide range of DQ rules preconfigured. You can find them in the ***Data Quality*** section of the ONE Web Application. As already explained in the second workshop, we have the following two types of rules:

- **Detection Rules** that are used for detecting business terms in Catalog Items.
- **DQ Evaluation Rules** that are used to evaluate the quality of data.

In this workshop, we will focus on the second group and create two evaluation rules.



Name	Type	Dimension	Rule definition source	Stewardship	Terms	Description
[CAN] City Pushdown	Detection		Ataccama Default Rules	Data Office		Canadian city
[CAN] Municipality	Detection		Ataccama Default Rules	Data Office		Canadian municipality
[CAN] Social Insurance Number Pushdown	Detection		Ataccama Default Rules	Data Office		A social insurance number SII
[CAN] ZIP code	Detection		Ataccama Default Rules	Data Office		Canadian postal code
[DEU] City	Detection		Ataccama Default Rules	Data Office		German city
[FRA] City	Detection		Ataccama Default Rules	Data Office		French city
[GBR] City	Detection		Ataccama Default Rules	Data Office		UK city
[GBR] National Insurance Number	Detection		Ataccama Default Rules	Data Office		The National Insurance numb

1. Create a DQ Rule from a lookup

If you navigate to the *Knowledge Catalog* and review the catalog item products, look closer at a column called **productline**. What does the data look like?

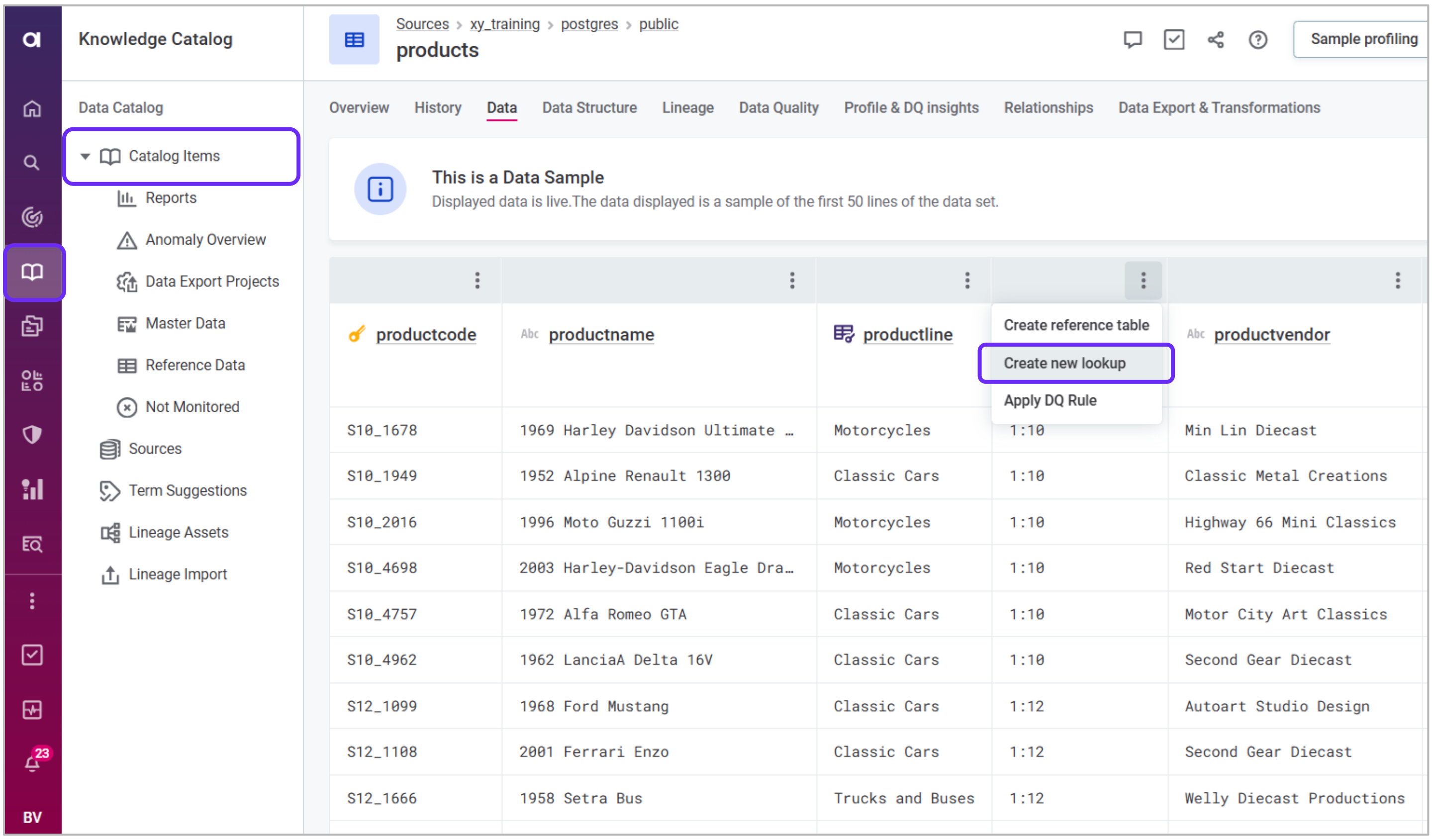
> *You can have a look at the attribute's samples on the **Data** tab or review the profiling results on the tab **Profile & DQ Insights**.*

The values of this attribute make it a perfect choice to create a lookup from. Furthermore, we can create and implement a DQ Rule and use this lookup as a reference in it. In this example we will then apply the rule to the productline column itself; if any changes happen to its data in the future, we can check whether the values exist in our lookup or not.

1a - Creating a Lookup

First, we will create the lookup itself. We can use the same data that is already stored in the **productline** column:

- Go to the **Data** tab of the **products** catalog item.
- In the header section, click on the three dots above the **productline** column name.
- Select the **Create new Lookup** option:

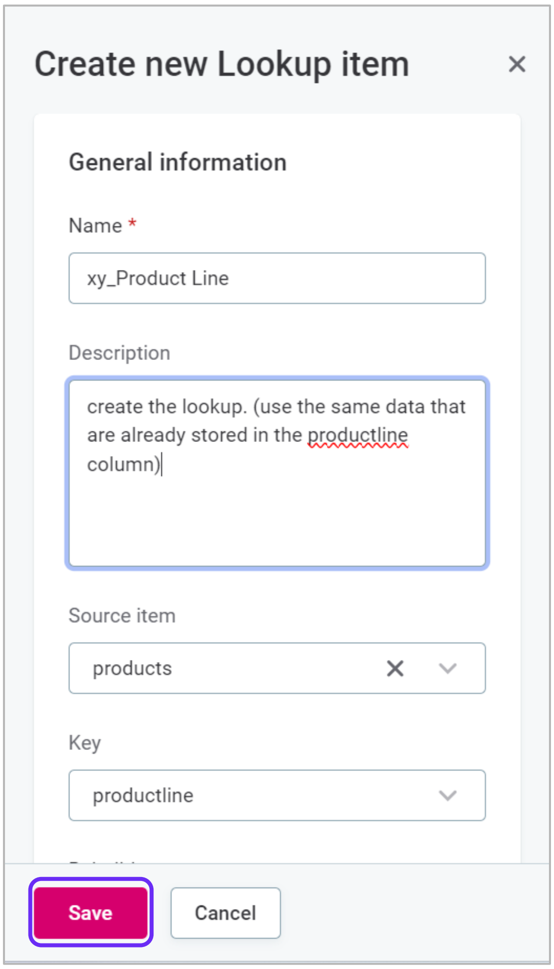


products

This is a Data Sample
Displayed data is live.The data displayed is a sample of the first 50 lines of the data set.

productcode	productname	productline		productvendor
S10_1678	1969 Harley Davidson Ultimate …	Motorcycles	1:10	Min Lin Diecast
S10_1949	1952 Alpine Renault 1300	Classic Cars	1:10	Classic Metal Creations
S10_2016	1996 Moto Guzzi 1100i	Motorcycles	1:10	Highway 66 Mini Classics
S10_4698	2003 Harley-Davidson Eagle Dra…	Motorcycles	1:10	Red Start Diecast
S10_4757	1972 Alfa Romeo GTA	Classic Cars	1:10	Motor City Art Classics
S10_4962	1962 LanciaA Delta 16V	Classic Cars	1:10	Second Gear Diecast
S12_1099	1968 Ford Mustang	Classic Cars	1:12	Autoart Studio Design
S12_1108	2001 Ferrari Enzo	Classic Cars	1:12	Second Gear Diecast
S12_1666	1958 Setra Bus	Trucks and Buses	1:12	Welly Diecast Productions

- Fill in the lookup **Name** (e.g. **<prefix>_Product Line'**) and optionally some **Description** if needed.
- Leave the remaining options unchanged and complete the creation by clicking on **Save**. You will need to **Publish** (click on the three dots beside the lookup's name), then click on the name **Product Line**.



Create new Lookup item

General information

Name *

xy_Product Line

Description

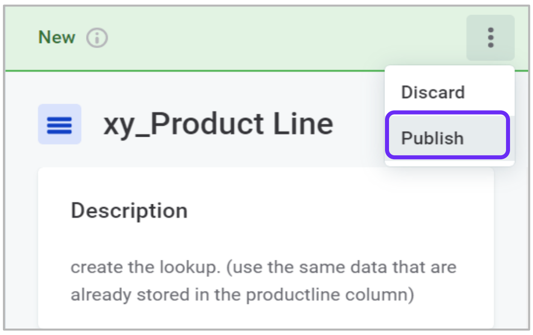
create the lookup. (use the same data that are already stored in the productline column)

Source item

products

Key

productline

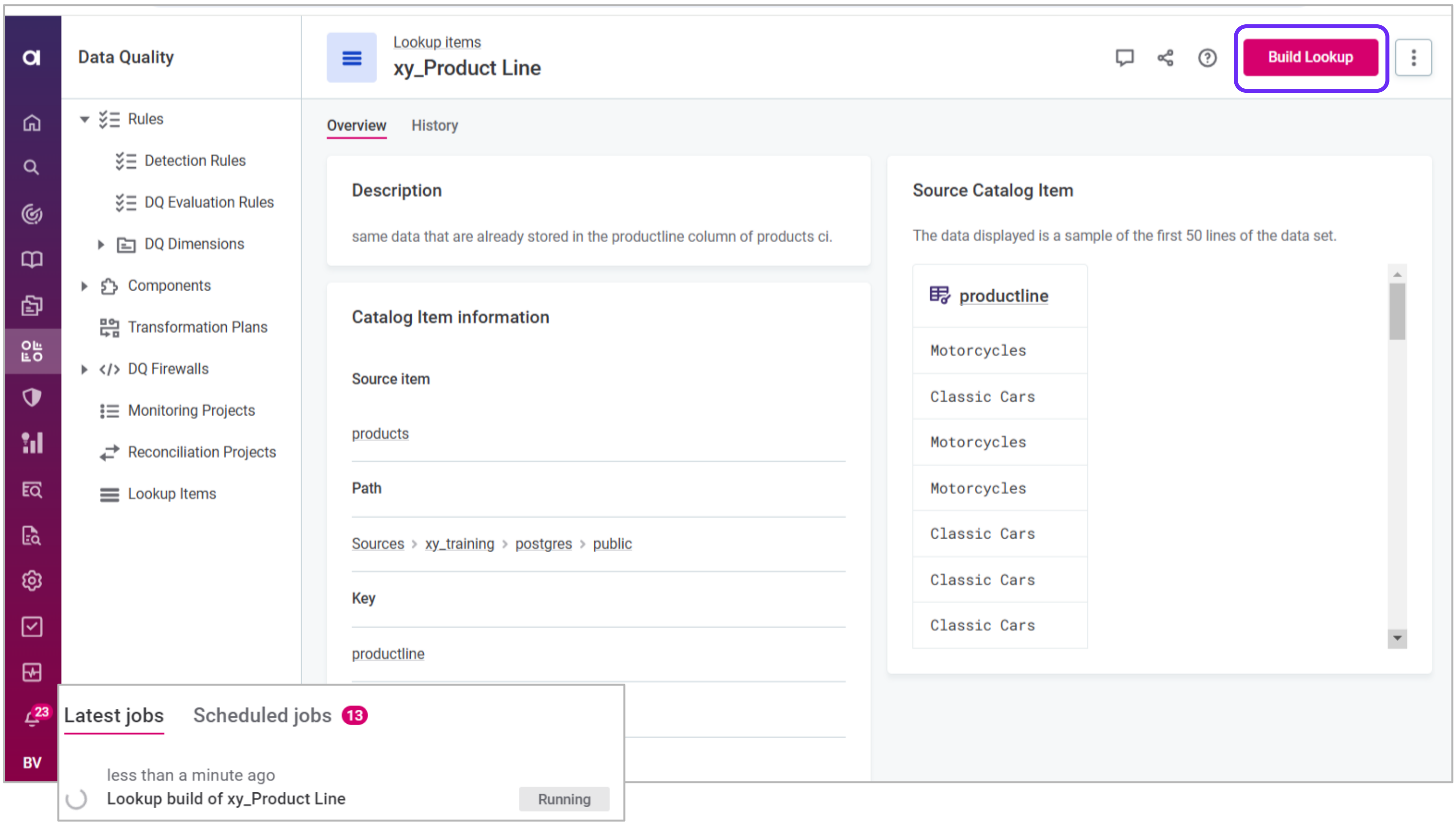


xy_Product Line

Description

create the lookup. (use the same data that are already stored in the productline column)

- Now you are in the lookup's **Overview** screen; click the **Build Lookup** in the upper right corner to finish the creation of the new lookup item:



xy_Product Line

Description

same data that are already stored in the productline column of products ci.

Catalog Item information

Source item

products

Path

Sources › xy_training › postgres › public

Key

productline

Source Catalog Item

The data displayed is a sample of the first 50 lines of the data set.

productline
Motorcycles
Classic Cars
Motorcycles
Motorcycles
Classic Cars
Classic Cars
Classic Cars

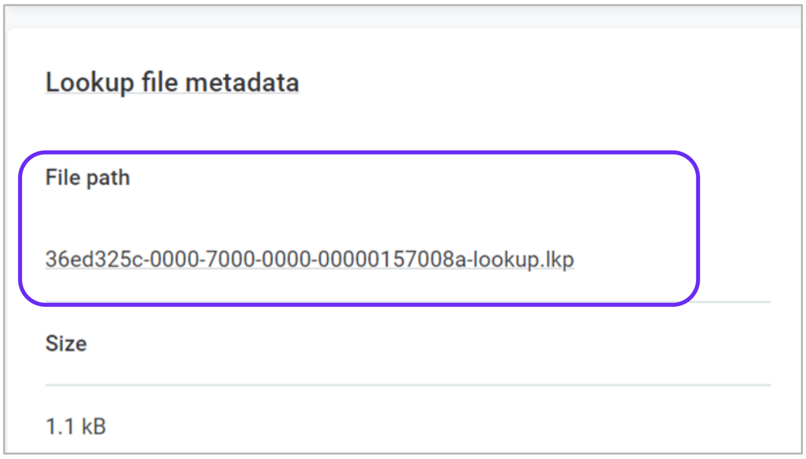
Latest jobs   Scheduled jobs 13

less than a minute ago
Lookup build of xy_Product Line   Running

NOTE

*You can check the process of the lookup building in the **Processing Center** section. You will receive a notification once it's complete and the lookup becomes available.*

Once finished, you can see some new information in the **Lookup metadata** section of your new lookup. Here you can also download the lookup file (**.lkp** extension) directly. If done so, the lookup can then be used in the ONE Desktop plans and components.



Lookup file metadata

File path

36ed325c-0000-7000-0000-00000157008a-lookup.lkp

Size

1.1 kB

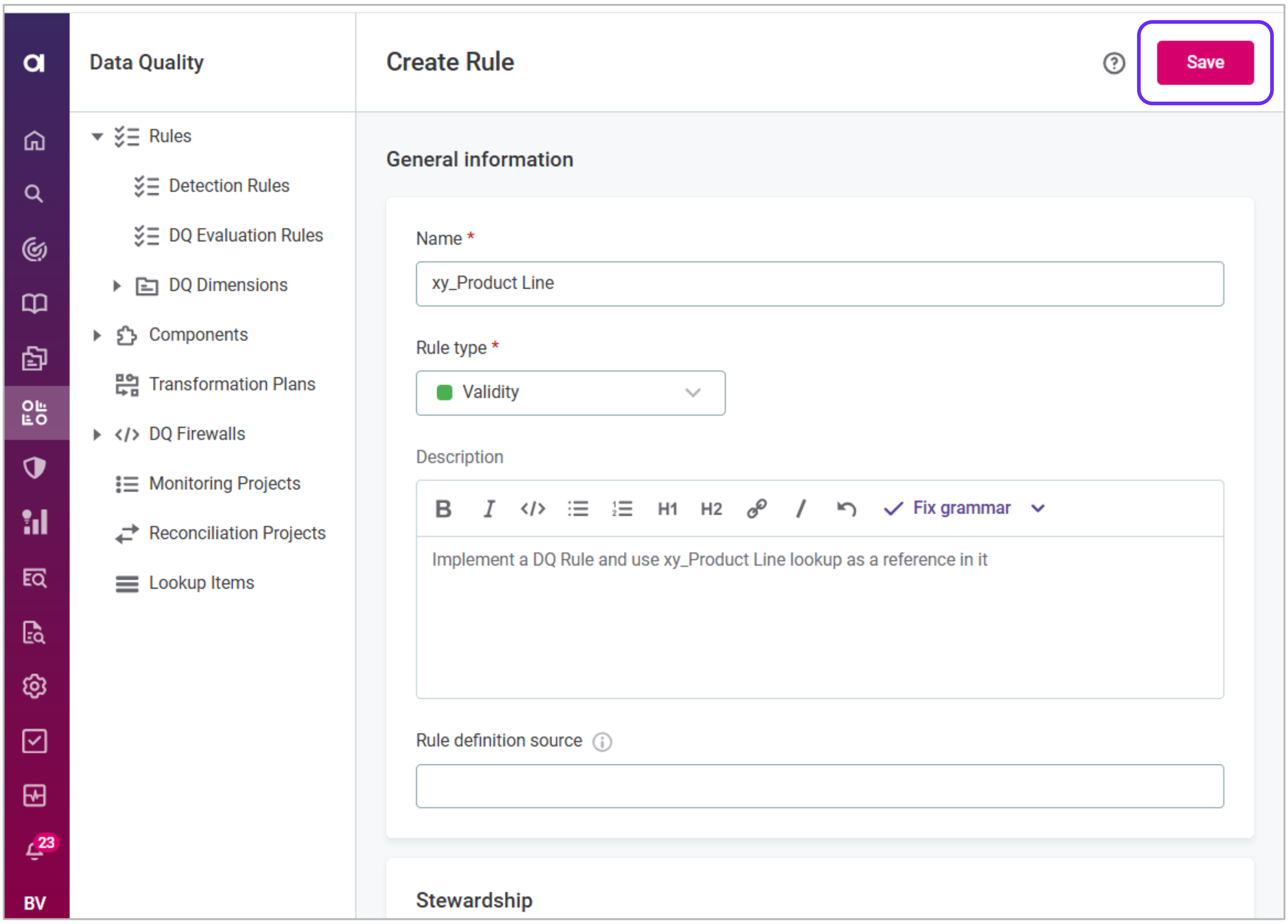
HINT

*You can search through available lookups in the **Data Quality** section **- Lookup Items.***

1b – DQ Rule Definition

Now that you have the lookup ready, you can create the data quality rule.

- Navigate to the **Data Quality** section on the left panel and click on **create button** in the top-right corner.
- Name the Rule '**<prefix>_Product Line**'.
- Choose the Rule type **DQ Evaluation Rules > Validity** option.
- Confirm your changes by clicking the **Save** button.



Create Rule

General information

Name *: xy_Product Line

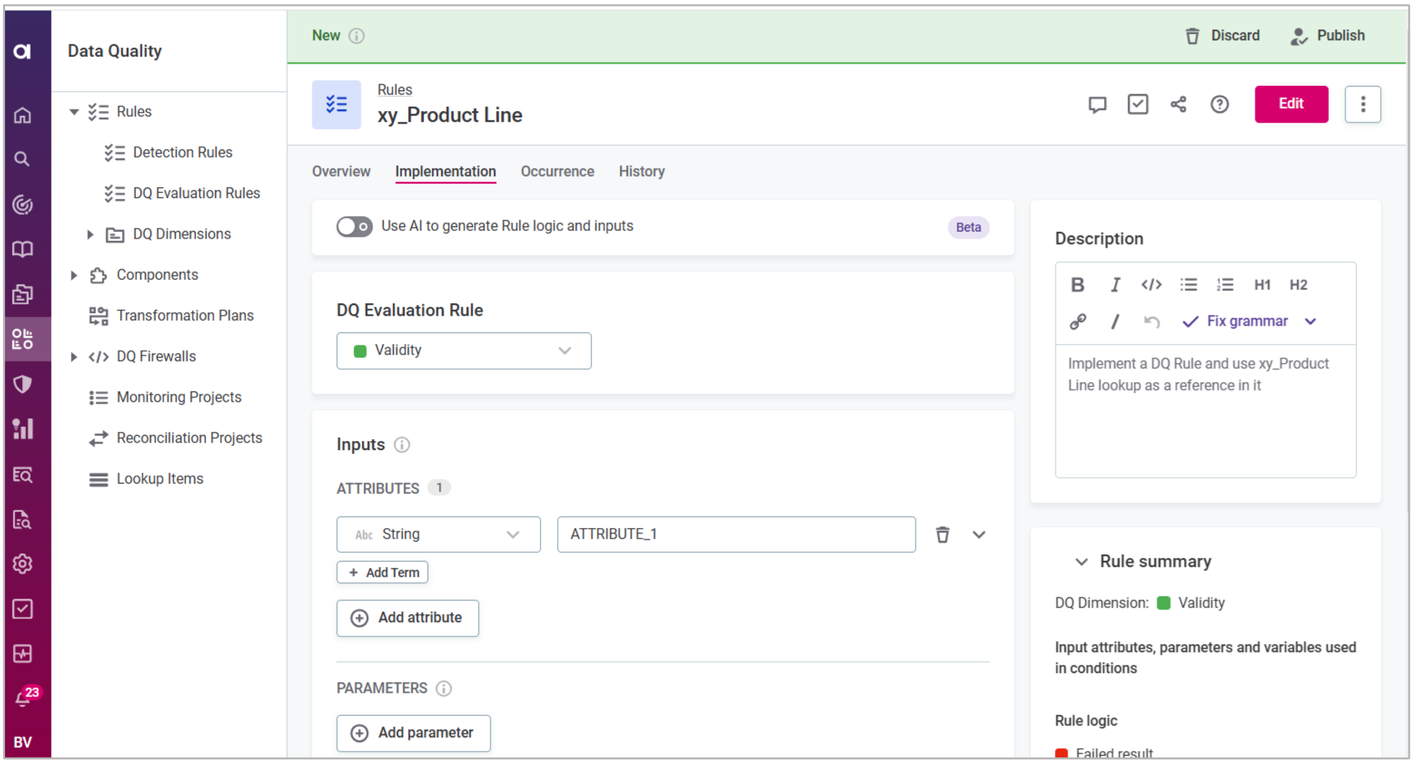
Rule type *: 🟩 Validity

Description: Implement a DQ Rule and use xy_Product Line lookup as a reference in it

Rule definition source:

Stewardship

The screen will change to the **implementation** tab and now includes options for developing the logic.



Rules

xy_Product Line

Use AI to generate Rule logic and inputs

DQ Evaluation Rule

🟩 Validity

Inputs

ATTRIBUTES 1

Abc String | ATTRIBUTE_1

PARAMETERS

Description

Implement a DQ Rule and use xy_Product Line lookup as a reference in it

Rule summary

DQ Dimension: 🟩 Validity

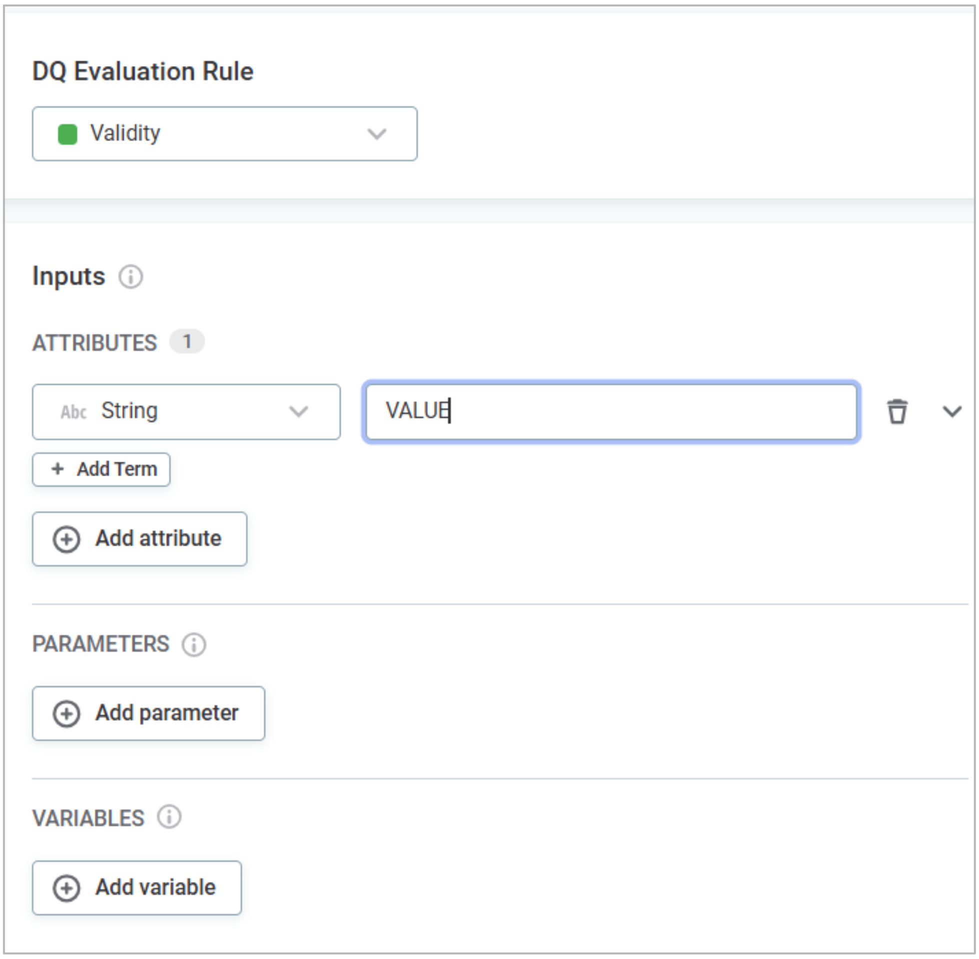
Input attributes, parameters and variables used in conditions

Rule logic

🟥 Failed result

Let’s start with renaming the attribute to a more meaningful one:

- In the **Input Attributes** section, replace the original name with a new value = **‘VALUE’**



DQ Evaluation Rule

🟩 Validity

Inputs

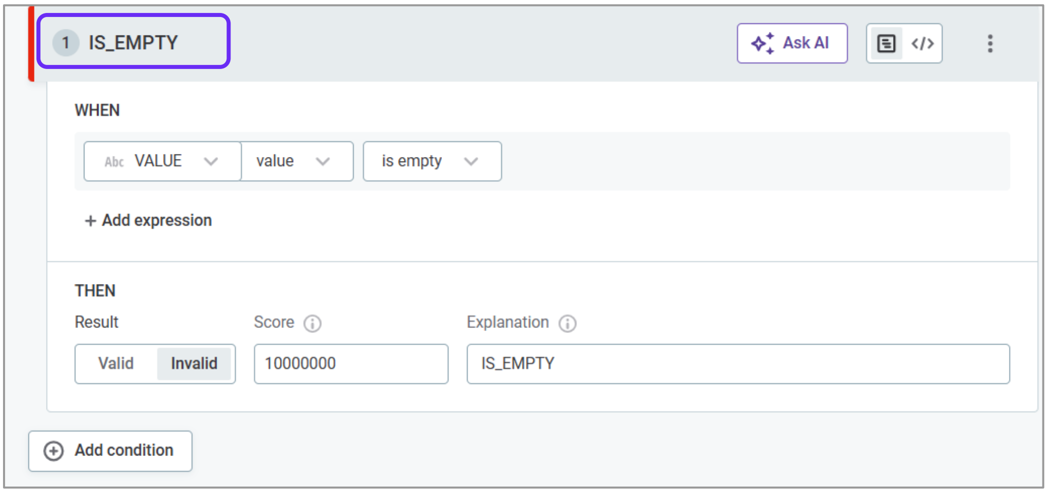
ATTRIBUTES 1

Abc String | VALUE

PARAMETERS

VARIABLES

- Moving on to the next **Rules Logic** section, notice that there is already one existing condition created by default ('IS_EMPTY') – this will consider any empty value as INVALID. Set its **Score** to **10000000.**



1 IS_EMPTY

WHEN

VALUE | value | is empty

+ Add expression

THEN

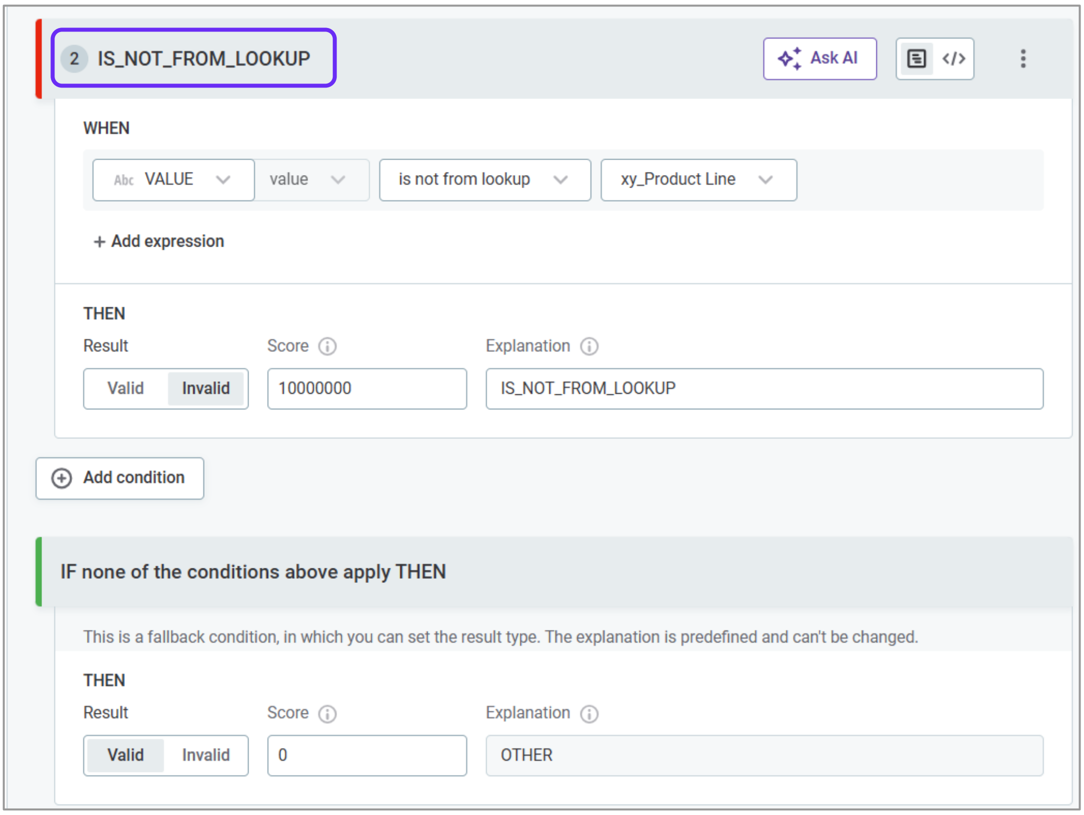
Result	Score	Explanation
Valid [✓ Invalid]	10000000	IS_EMPTY

- Create the second condition by clicking on **Add condition** to check if the value exists in your newly created lookup item **<prefix>_Product Line.**

  Explanation: **NOT_IN_LOOKUP**

  Set the score to **1000000**.

- For other cases, the result will be **VALID**.



2 IS_NOT_FROM_LOOKUP

WHEN

VALUE | value | is not from lookup | xy_Product Line

+ Add expression

THEN

Result	Score	Explanation
Valid [✓ Invalid]	10000000	IS_NOT_FROM_LOOKUP

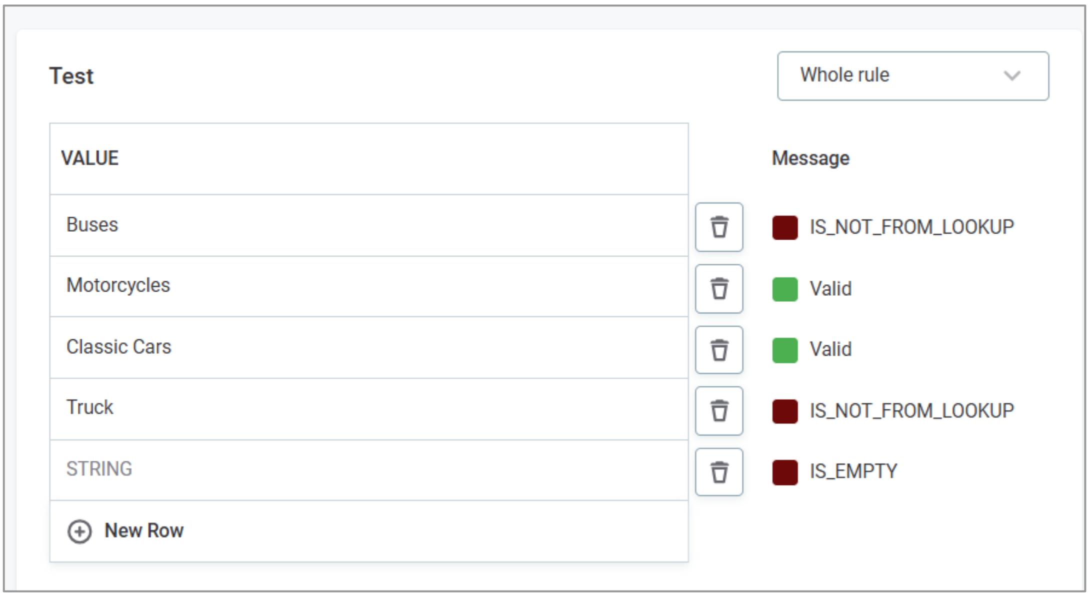
IF none of the conditions above apply THEN

This is a fallback condition, in which you can set the result type. The explanation is predefined and can't be changed.

THEN

Result	Score	Explanation
[✓ Valid] Invalid	0	OTHER

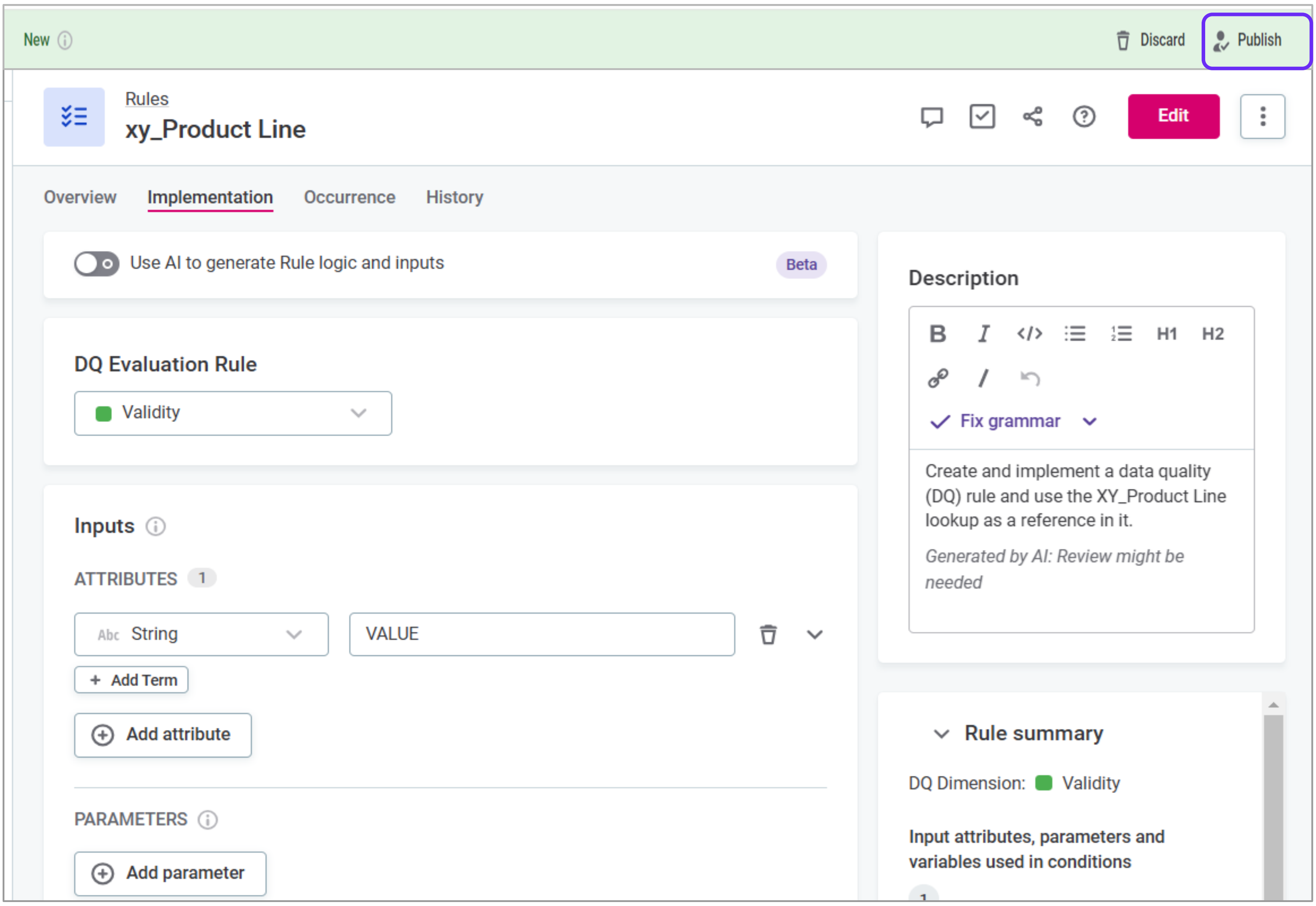
- You can use the **Test Rule** button to verify the functionality of the rule.


Test

Whole rule

VALUE	Message
Buses	🟥 IS_NOT_FROM_LOOKUP
Motorcycles	🟩 Valid
Classic Cars	🟩 Valid
Truck	🟥 IS_NOT_FROM_LOOKUP
STRING	🟥 IS_EMPTY

- Once you are satisfied with everything, you can **Publish** the rule.



xy_Product Line

Use AI to generate Rule logic and inputs

DQ Evaluation Rule

🟩 Validity

Inputs

ATTRIBUTES 1

String | VALUE

PARAMETERS

Description

Create and implement a data quality (DQ) rule and use the XY_Product Line lookup as a reference in it.

Generated by AI: Review might be needed

Rule summary

DQ Dimension: 🟩 Validity

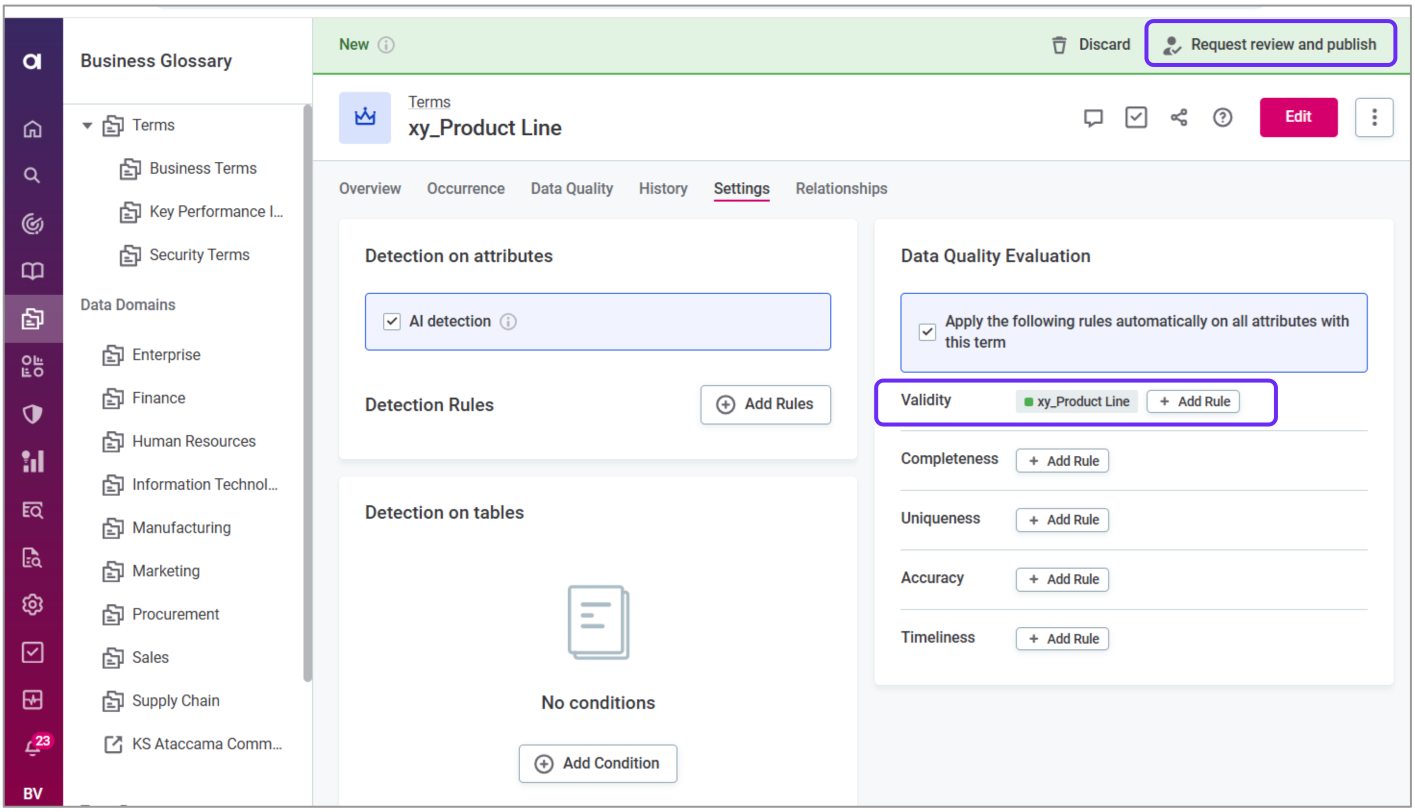
Input attributes, parameters and variables used in conditions

1c – Implementing the DQ Rule

Now that we have the rule created, there are three options on how to use it. We can either assign it to a term and evaluate the data using it after the term is detected on an attribute, create a **Data Quality Monitoring Project,** or directly assign it to a Catalog Item's attribute.

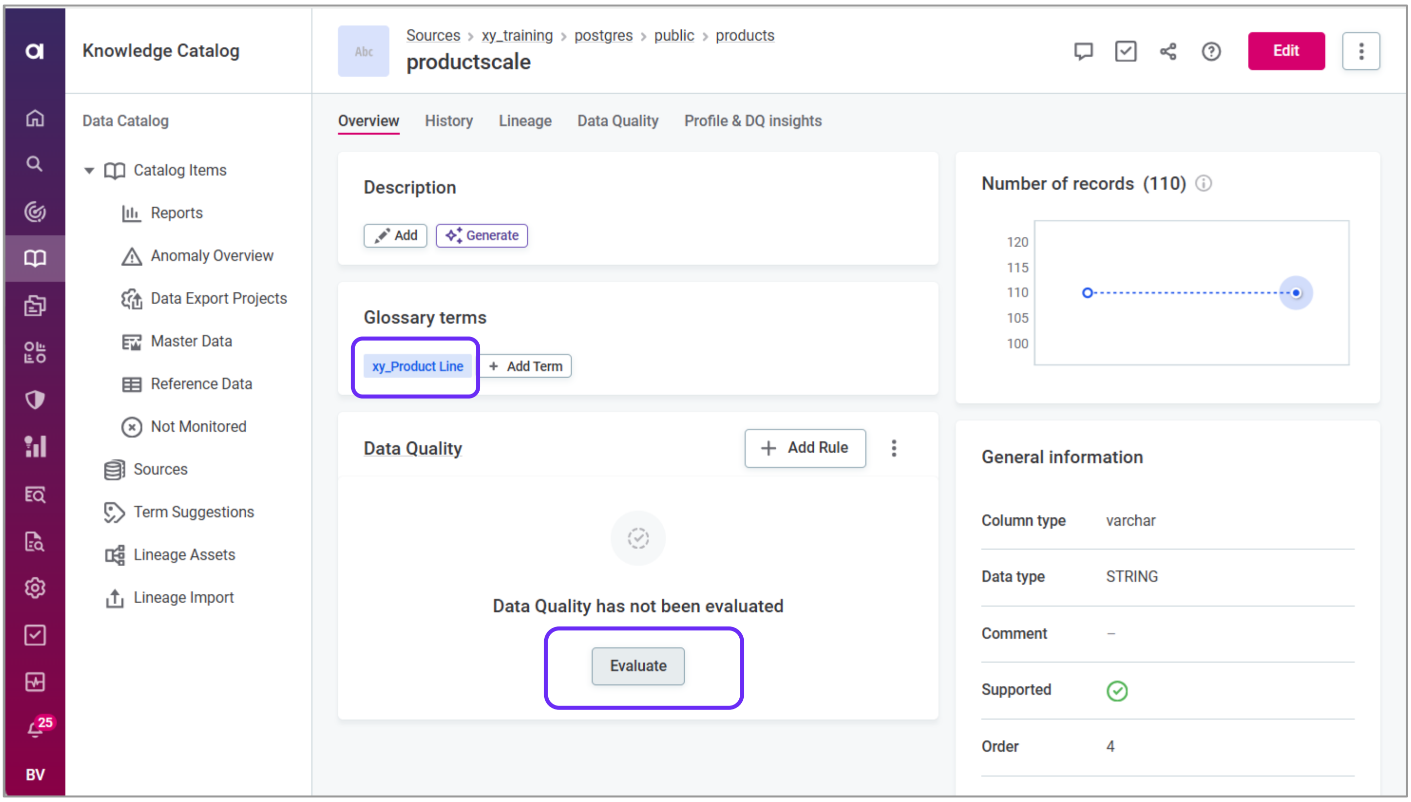
In this exercise, we will try the first option.

- Create a new term called **<prefix>_Product Line**.
- On the **Settings** tab, add the new rule as a **Validity** rule.
- When done, **Review and Publish** the changes.



Now let's test the rules! Since we don't have any detection rule assigned to the new term yet, we have to assign the term manually at least once on our own. This will also help the **AI** machine learning engine to start detecting the Term in similar data values:

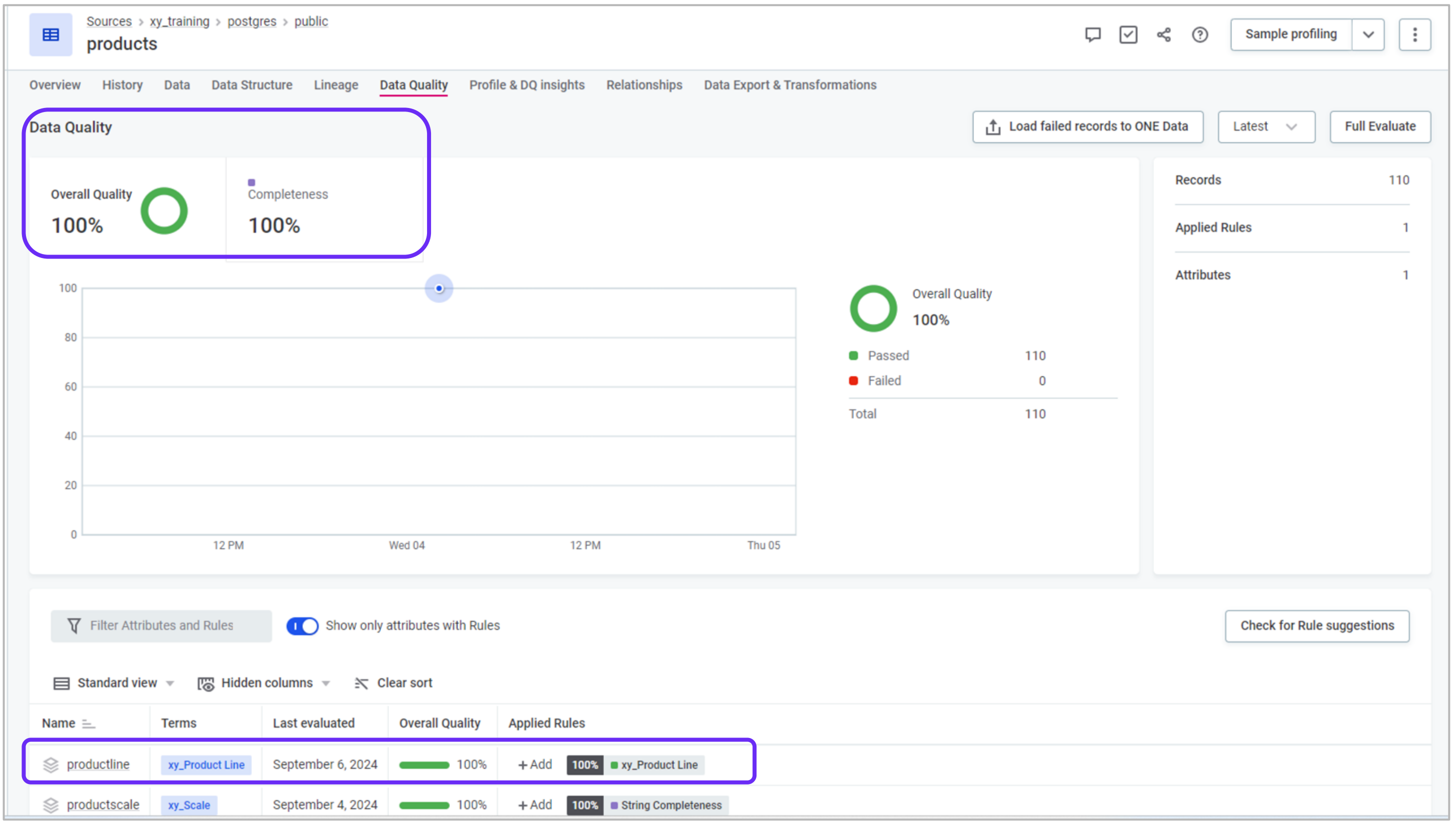
- Go to the catalog item **products** and open a detail window of the attribute **productline**.
- Add the new term **<prefix>_Product Line** in the **Glossary Terms** section via the **'+ Add Term'** button.
- To see the results of the DQ rule, run the DQ evaluation using the button **Evaluate**:



Column type	varchar
Data type	STRING
Comment	–
Supported	✓
Order	4

If there is any existing catalog item with an attribute containing values like the **productline** attribute, the **AI** will start suggesting the term for it. Each time a similar pattern is detected, you can observe it in the Glossary Terms section.

By **approving** ✓ or **rejecting** × the proposal suggestions you help the AI to learn how to recognize the term better.



Passed	110
Failed	0
Total	110

Records	110
Applied Rules	1
Attributes	1

Name	Terms	Last evaluated	Overall Quality	Applied Rules
productline	xy_Product Line	September 6, 2024	100%	+ Add 100% xy_Product Line
productscale	xy_Scale	September 4, 2024	100%	+ Add 100% String Completeness

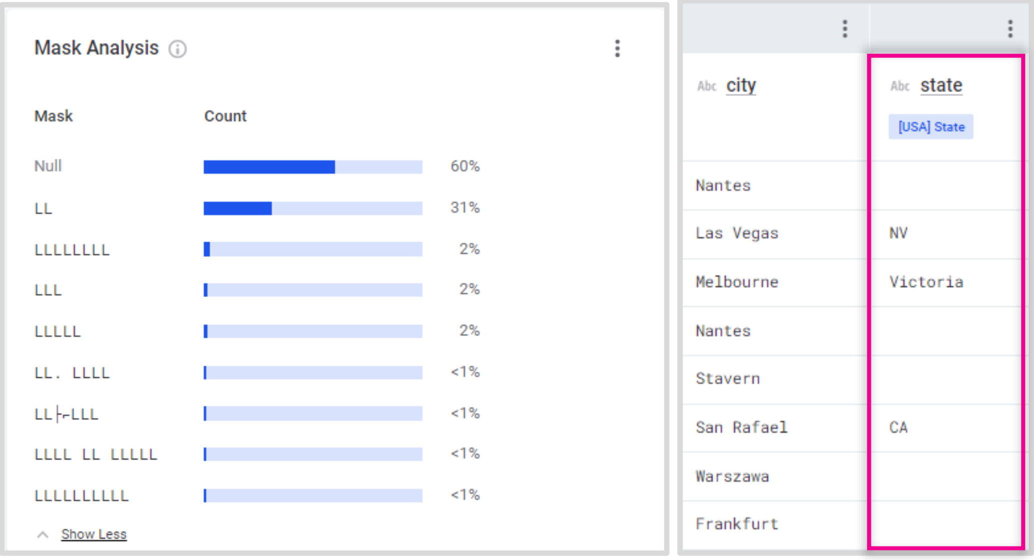
2. Create a DQ Rule to detect a pattern

In the second example of DQ evaluation rules, we will try to detect a specific pattern within the data values. We simply want to check if the values here are following that defined pattern or not.

For this to be demonstrated, navigate to the ***Knowledge Catalog***, review the catalog item **customers,** and look closer at the results of the column **state.** Observe the value variants.

HINT

*You can have a look at the attribute's samples on the **Data** tab or review the profiling results on the tab **Profile & DQ Insights**.*



Mask Analysis

Mask	Count
Null	60%
LL	31%
LLLLLLLL	2%
LLL	2%
LLLLL	2%
LL. LLLL	<1%
LL├-LLL	<1%
LLLL LL LLLLL	<1%
LLLLLLLLLL	<1%

Show Less

Abc city	Abc state [USA] State
Nantes	
Las Vegas	NV
Melbourne	Victoria
Nantes	
Stavern	
San Rafael	CA
Warszawa	
Frankfurt	

2a – Rule Definition

Some values here follow the pattern of exactly two capital letters to represent a state (e.g. **CA**'), while others are either empty or have a different format. We want to create a rule to confirm the existence of the default pattern which is the two capital letters only. Values not satisfying this condition will result in INVALID based on this new rule.

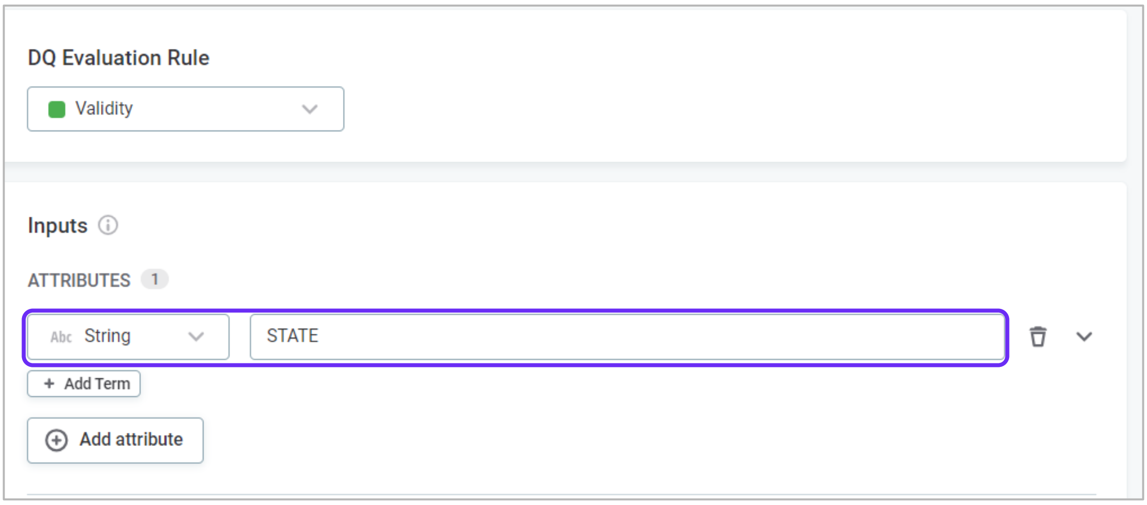
› Go ahead and create the new rule with the following properties:

- *Rule Name* **<prefix>_State code**
- *Rule Description* '**Validates a two-capital letter pattern**'

TASK

All by yourself now! You should now be familiar with creating a new DQ Rule by now. If in doubt, refer to the **ONE Workshop – Glossary** for instructions and more information

- Set the Rule Logic as **DQ Evaluation**, then choose **Validity** as dimension.
- In the **Implementation** tab, set the Input attribute as '**STATE'** and click on the '**Add Condition**' button to add a new line of logic definition to the default **'IS_EMPTY'** one:



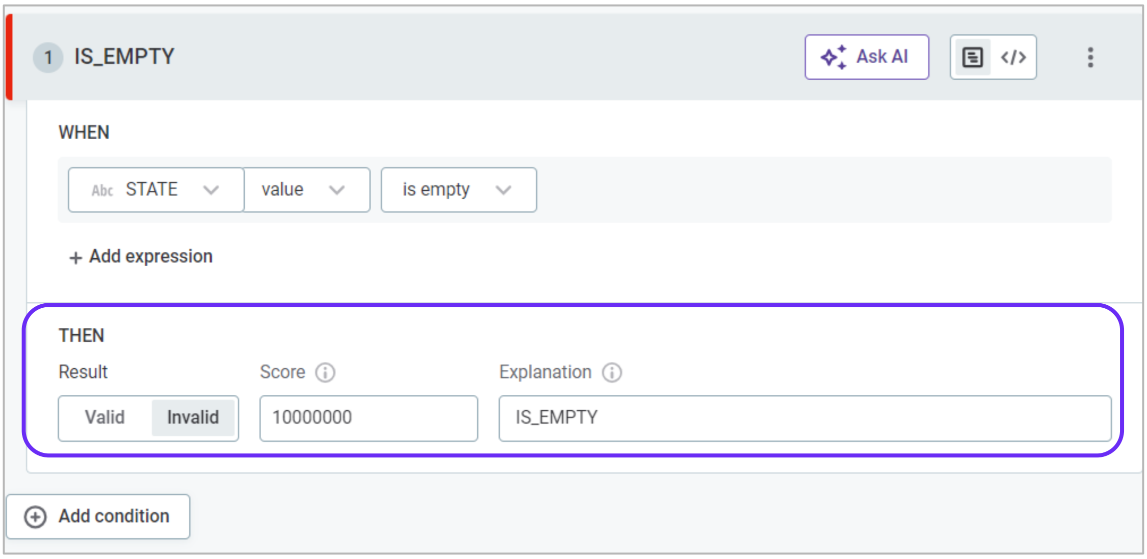
DQ Evaluation Rule

🟩 Validity

Inputs

ATTRIBUTES 1

String | STATE



1 IS_EMPTY

WHEN

STATE | value | is empty

THEN

Result: Valid / Invalid

Score: 10000000

Explanation: IS_EMPTY

In the new custom definitions of logic, we want to apply two different checks. Each check will have its own line of definition, explanation code and score value to assign. If none of the defined conditions are satisfied, we will consider it **VALID**.

Check A – **Value is too long** – the value's length is longer than expected 2 characters.

*This will assign a score of **40 000** and have the '**EXCEEDS_LENGTH**' explanation code.*

Check B – **Value doesn't fit the pattern** – there are not exactly two capital letters in the value.

*This will assign a score of **15 000** and have the '**PATTERN_MISMATCH**' explanation code.*

TASK

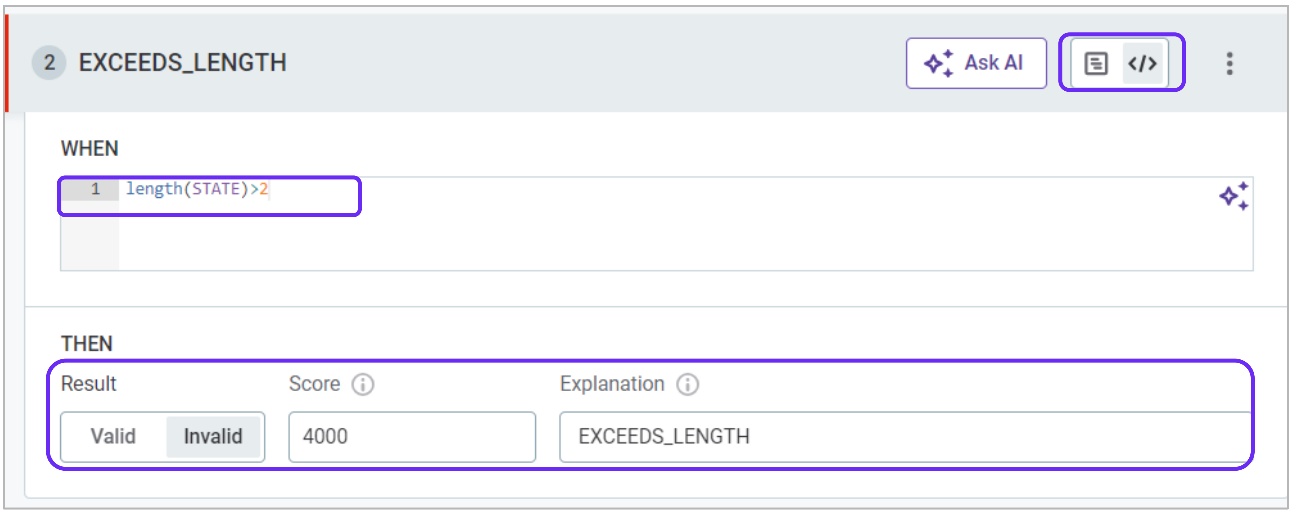
Try it yourself! You should already know how to construct the definitions from previous exercises. Continue only if you want to follow instructions,

Check A – Value is too long

This check will validate the total length of the incoming value and fails if it is longer than 2 characters.

Firstly, we want to use the expression language rather than the default condition builder.

- Switch to the '**Advanced Expression**' mode.
- Populate the **WHEN** field with the following expression: **length(STATE)>2.**
- Fill in the **Score** (40000) and **Explanation Code** ("EXCEEDS_LENGTH") values.
- Make sure that a value matching this logic will make the **Result** INVALID.



2 EXCEEDS_LENGTH

WHEN

```
length(STATE)>2
```

THEN

Result	Score	Explanation
Valid **Invalid**	4000	EXCEEDS_LENGTH

Now let's repeat the process and create the second definition:

Check B – Value doesn't fit the pattern

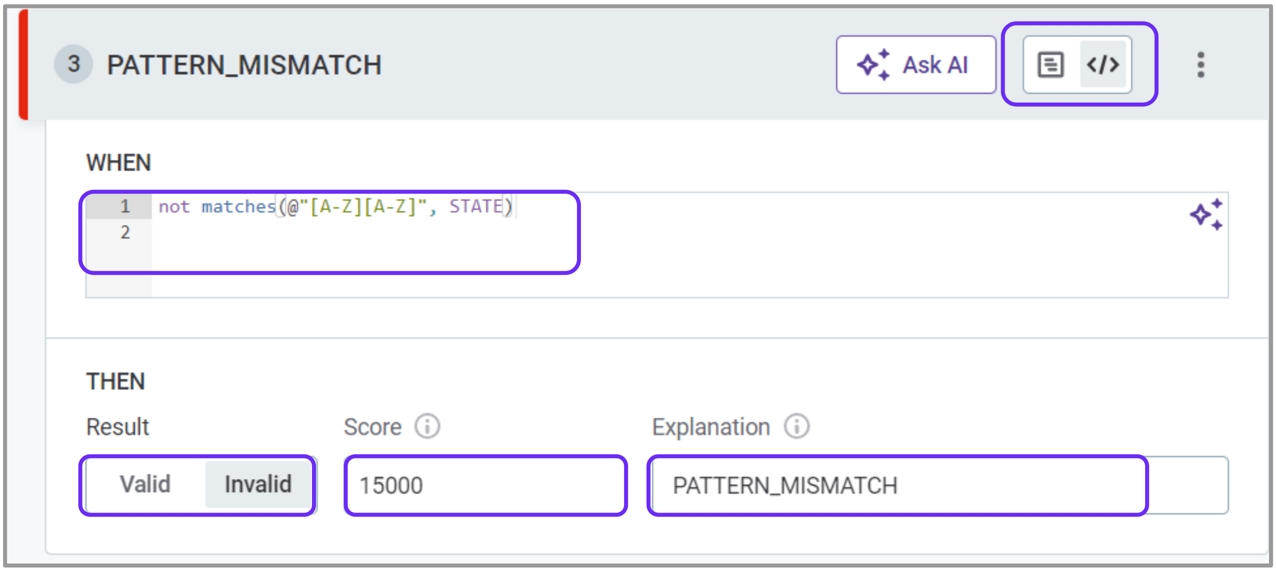
This check will use Regular expressions to look for an exact array of characters (capital letters) and exactly two of them:

- Click the '**Add Condition**' button to add another line of logic definition.

Again, we want to use the expression language rather than the default condition builder:

- Switch to the 'Advanced Expression' mode.
- Populate the **WHEN** field with the following expression: **not matches(@"[A-Z][A-Z]", STATE).**
- Fill in the **Score** (15 000) and **Explanation Code** (**PATTERN_MISMATCH**) values.

- Make sure that a value matching this logic will make the **Result** INVALID.



3 PATTERN_MISMATCH

WHEN

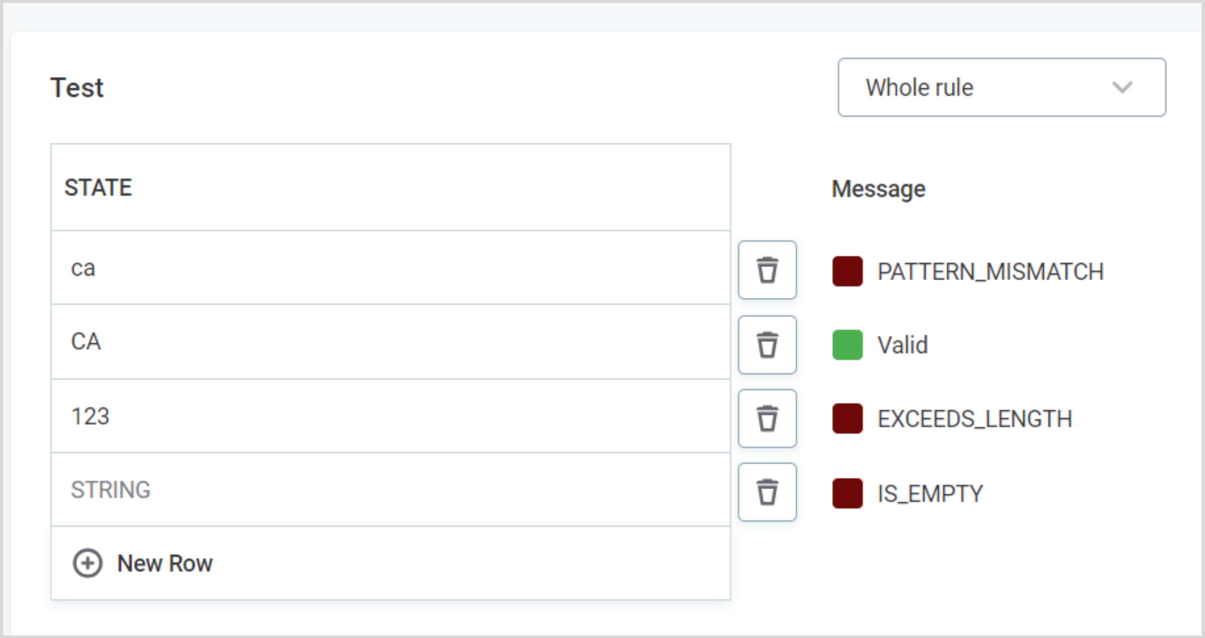
```
not matches(@"[A-Z][A-Z]", STATE)
```

THEN

Result	Score	Explanation
Valid [✓ Invalid]	15000	PATTERN_MISMATCH

When the definition is complete, it’s time to make sure it works as expected:

- Test the rule by clicking on the **Test Rule** button at the top of the section.
- Try writing a few data samples to make sure you write the expression correctly.



Test

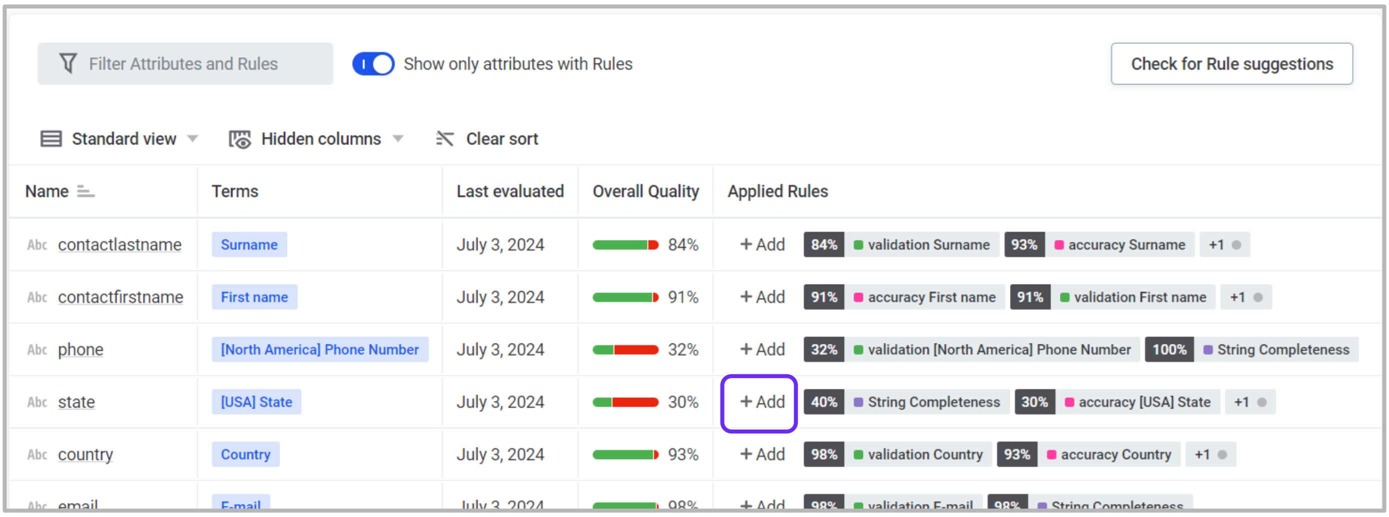
Whole rule

STATE	Message
ca	🟥 PATTERN_MISMATCH
CA	🟩 Valid
123	🟥 EXCEEDS_LENGTH
STRING	🟥 IS_EMPTY

2b - Add the Evaluation rule to an attribute

To see how this rule works, we will link it with the attribute **state.**

- Navigate to the **customers** Catalog Item and switch to its **Data Quality** tab.
- In the Rules column, select (**plus**) Add DQ Checks for the attribute **state.**

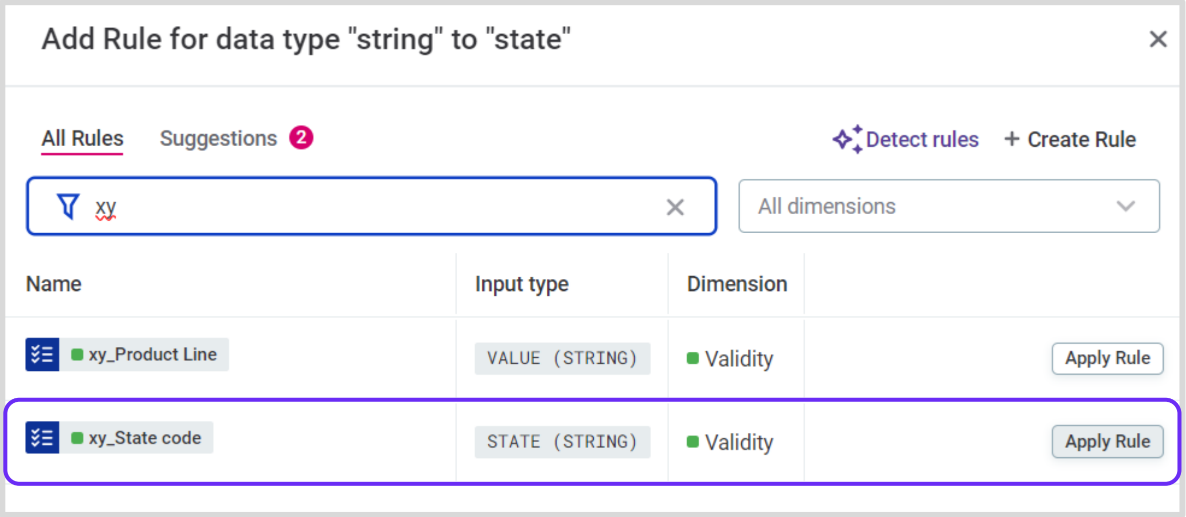


Filter Attributes and Rules | Show only attributes with Rules | Check for Rule suggestions

Standard view | Hidden columns | Clear sort

Name	Terms	Last evaluated	Overall Quality	Applied Rules
Abc contactlastname	Surname	July 3, 2024	84%	+ Add 84% validation Surname 93% accuracy Surname +1
Abc contactfirstname	First name	July 3, 2024	91%	+ Add 91% accuracy First name 91% validation First name +1
Abc phone	[North America] Phone Number	July 3, 2024	32%	+ Add 32% validation [North America] Phone Number 100% String Completeness
Abc state	[USA] State	July 3, 2024	30%	+ Add 40% String Completeness 30% accuracy [USA] State +1
Abc country	Country	July 3, 2024	93%	+ Add 98% validation Country 93% accuracy Country +1
Abc email	E-mail	July 3, 2024	98%	+ Add 98% validation E-mail 98% String Completeness

- Choose a **<prefix>_State code** rule from the options available.



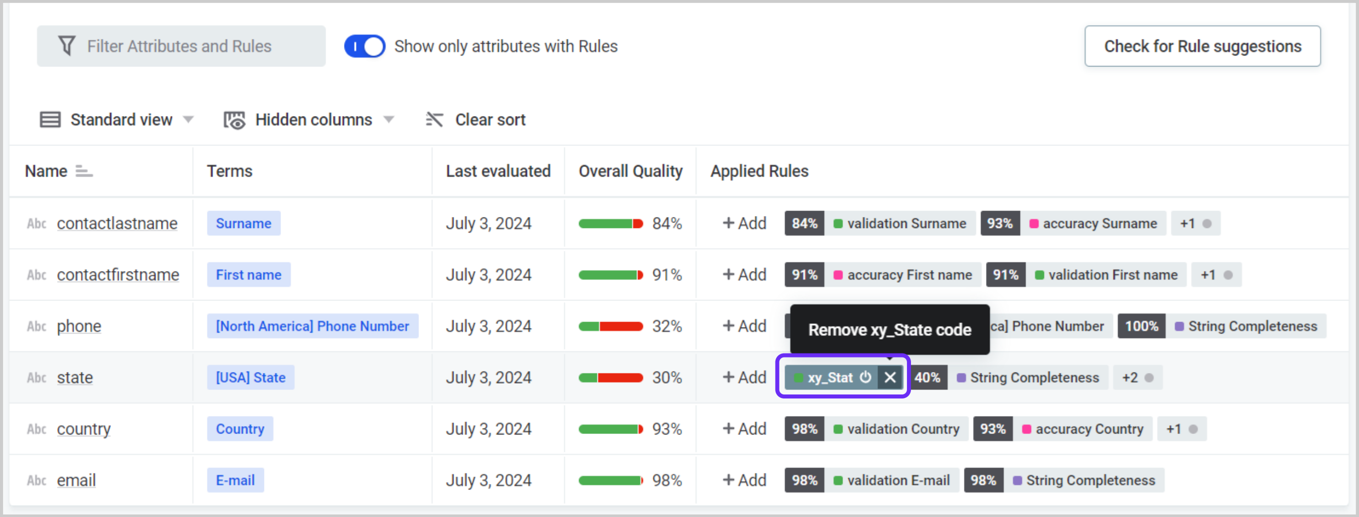
Add Rule for data type "string" to "state"

All Rules | Suggestions 2 | Detect rules | + Create Rule

xy | All dimensions

Name	Input type	Dimension	
xy_Product Line	VALUE (STRING)	Validity	Apply Rule
xy_State code	STATE (STRING)	Validity	Apply Rule

- Select **Assign Rule**



Filter Attributes and Rules

Show only attributes with Rules

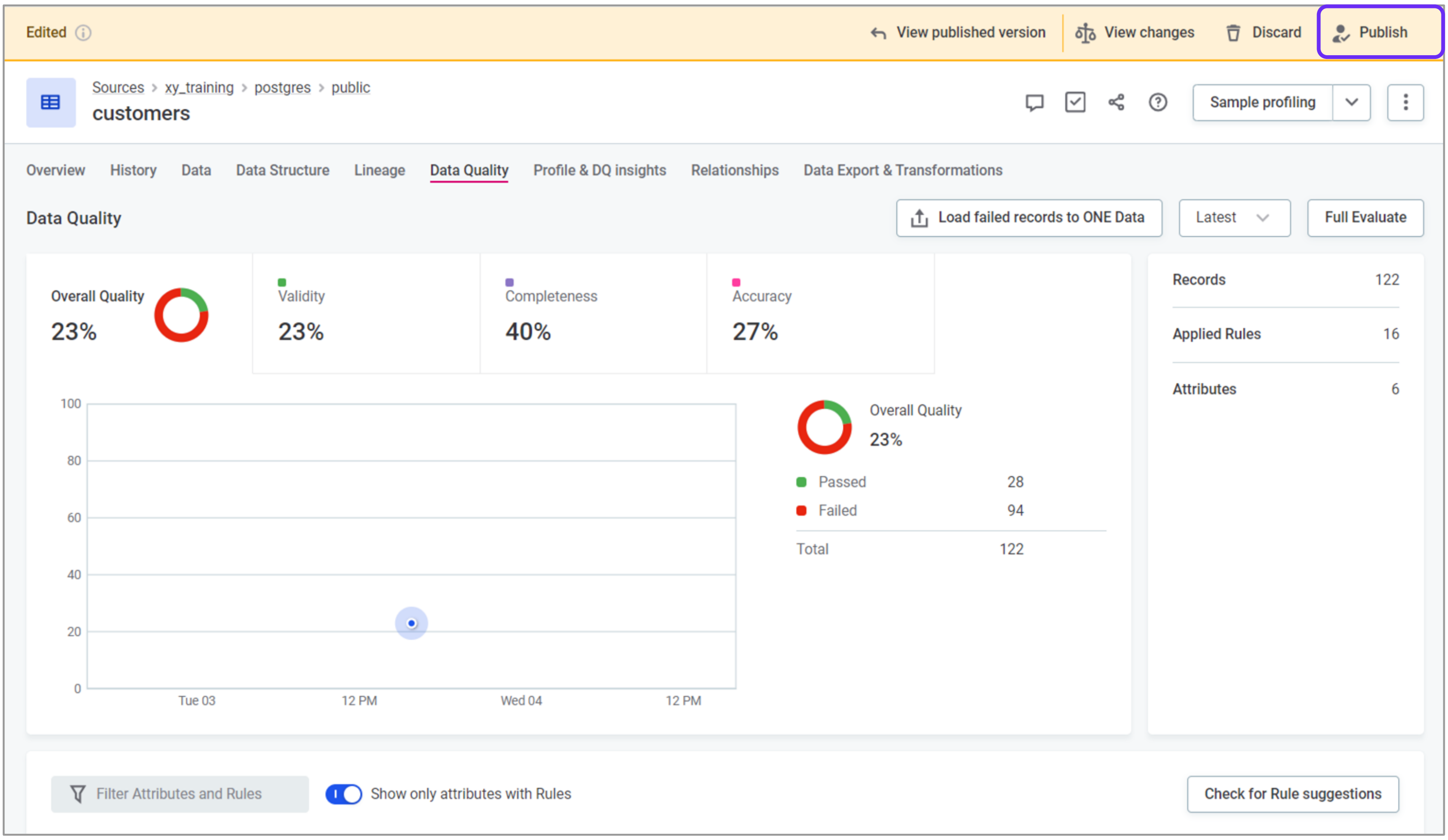
Check for Rule suggestions

Standard view · Hidden columns · Clear sort

Name	Terms	Last evaluated	Overall Quality	Applied Rules
Abc contactlastname	Surname	July 3, 2024	84%	+ Add 84% validation Surname 93% accuracy Surname +1
Abc contactfirstname	First name	July 3, 2024	91%	+ Add 91% accuracy First name 91% validation First name +1
Abc phone	[North America] Phone Number	July 3, 2024	32%	+ Add ca] Phone Number 100% String Completeness
Abc state	[USA] State	July 3, 2024	30%	+ Add xy_Stat ✕ 40% String Completeness +2
Abc country	Country	July 3, 2024	93%	+ Add 98% validation Country 93% accuracy Country +1
Abc email	E-mail	July 3, 2024	98%	+ Add 98% validation E-mail 98% String Completeness

Remove xy_State code

- **Publish** the changes.



Edited ⓘ

View published version · View changes · Discard · Publish

Sources › xy_training › postgres › public

customers

Sample profiling

Overview · History · Data · Data Structure · Lineage · Data Quality · Profile & DQ insights · Relationships · Data Export & Transformations

Data Quality

Load failed records to ONE Data · Latest · Full Evaluate

Overall Quality	Validity	Completeness	Accuracy
23%	23%	40%	27%

Records	122
Applied Rules	16
Attributes	6

Overall Quality 23%

🟩 Passed	28
🟥 Failed	94
Total	122

Filter Attributes and Rules

Show only attributes with Rules

Check for Rule suggestions

2c - Run the Evaluation

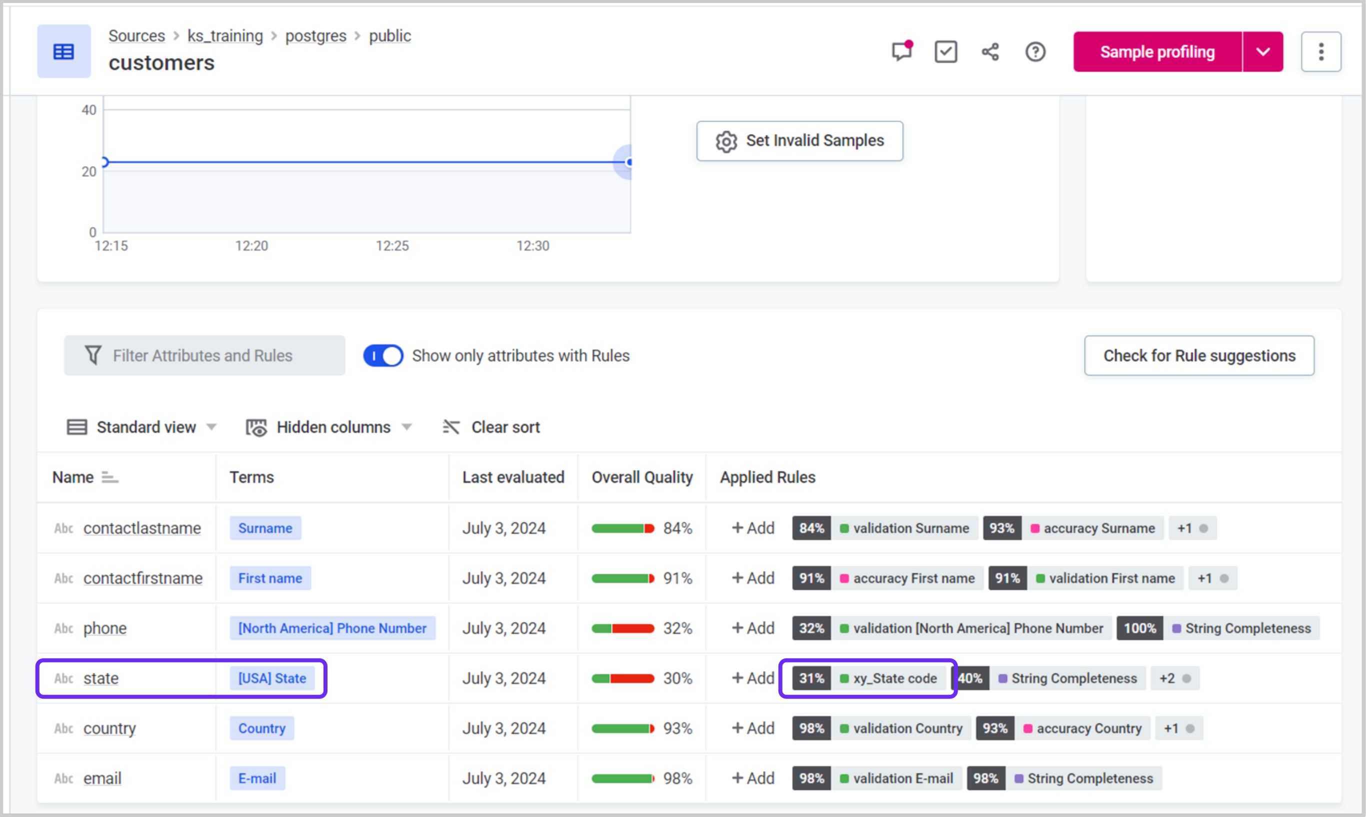
Now it’s time to see if the new rule works during the evaluation process.

- Navigate to the **customers** Catalog Item and switch to its **Data Quality** tab.

If you have already run any DQ evaluations on this table before, there should be a summary of the results. If not, it’s fine too because we will run the evaluation now anyway.

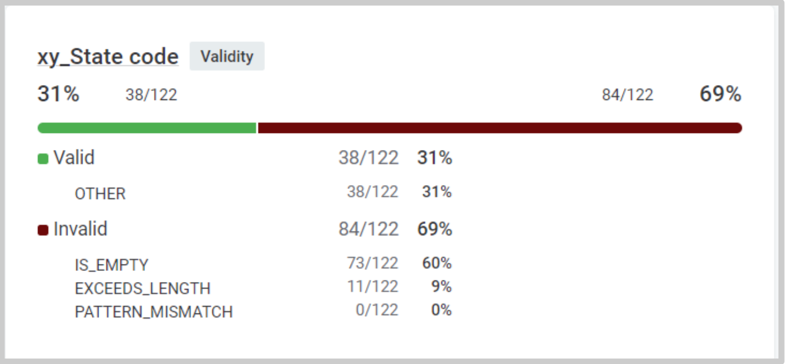
- Press the **FULL Evaluate** button in the top right corner to start it:

Your new rule will be applied to attribute(s) with the assigned term and the aggregated validation results will be soon available in the overview tab of the Catalog Item:


Sources > ks_training > postgres > public
customers
Sample profiling
40
20
0
12:15
12:20
12:25
12:30
Set Invalid Samples
Filter Attributes and Rules
Show only attributes with Rules
Check for Rule suggestions
Standard view
Hidden columns
Clear sort

Name	Terms	Last evaluated	Overall Quality	Applied Rules
Abc contactlastname	Surname	July 3, 2024	84%	+ Add 84% validation Surname 93% accuracy Surname +1
Abc contactfirstname	First name	July 3, 2024	91%	+ Add 91% accuracy First name 91% validation First name +1
Abc phone	[North America] Phone Number	July 3, 2024	32%	+ Add 32% validation [North America] Phone Number 100% String Completeness
Abc state	[USA] State	July 3, 2024	30%	+ Add 31% xy_State code 40% String Completeness +2
Abc country	Country	July 3, 2024	93%	+ Add 98% validation Country 93% accuracy Country +1
Abc email	E-mail	July 3, 2024	98%	+ Add 98% validation E-mail 98% String Completeness

- Click it to display details and observe the results of your new rule with totals and explanations:



xy_State code Validity

31% 38/122 84/122 69%

🟩 Valid	38/122	31%
OTHER	38/122	31%
🟥 Invalid	84/122	69%
IS_EMPTY	73/122	60%
EXCEEDS_LENGTH	11/122	9%
PATTERN_MISMATCH	0/122	0%

Conclusion

We have come to the end of this workshop!
We have created two DQ Evaluation (validation) rules using a lookup item and constructed logical conditions. If applicable, in further exercises you will also learn how to use ONE Desktop to create even more complex validation rules.