

THIS FILE IS COVERED BY MICROSOFT'S NDA. IT IS ONLY TO BE SHARED WITH AUTHORIZED CUSTOMERS WHO ARE PART OF THE Microsoft PURVIEW DQ PRIVATE PREVIEW PROGRAM.

Microsoft Purview Data Quality Documentation

1 Introduction

Welcome to Microsoft Purview's Data Quality (DQ) Private Preview, where Microsoft Purview is showing the very earliest versions of its Data Quality features. Features, capabilities, etc. can change frequently and things may break. Right now, support is "best effort" which means if something goes wrong then it will be diagnosed and fixed during office hours.

The most important thing to remember about the private preview is that when in doubt, if there are questions, concerns, bugs, anything you need to know about the private preview, please email purviewdqprsupport@microsoft.com.

To get going on our journey please review sections 3-6.

2 Change Log

Date	Description
4/7/2022	Update to new UX
4/20/2022	Clarify that we don't support CSV support right now
5/2/2022	Called out the FQN issue won't be fixed even by public preview because of how rare we believe it is
5/10/2022	Included a new section to the document: Enable Azure Trusted Services if data source is behind VNet Updated reference sections
5/17/2022	Updated section 8.2 including all supported VNet scenarios
5/20/2022	Swapped sections 5 and 6. Updated product name. Added history task to goals.

3 Glossary

Microsoft Purview – Microsoft's premier cloud hosted Software as a Service providing data governance.

Data Quality Rule – A check that determines if the data is in the form required.

Data Quality Scan – The application of rules to data in order to determine if the data is in its correct form. The output of a data quality scan is a global score, rule scores and row by row error information.

Data Quality Cleansing – Fixing issues identified by Data Quality scans.

Microsoft Purview DQ – A shorthand for Microsoft Purview Data Quality.

4 Known Limitations

Data Sovereignty Compliance – When Purview DQ runs a Data Quality scan it retrieves the data from wherever the data is located and moves it for analysis in the region where the Purview account resides. The results of this analysis produces data samples to enable quick previews as well as the full values from the source asset in a sample of rows with errors in order to enable detailed error reporting. This means that data is taken from the source asset during analysis and physically stored in Purview's data stores to power the preview and detailed error results features. The implications of this are that if, say, a Purview Account is in East US 2 and the data to be analyzed is in West Europe then when a DQ job is initiated from the East US 2 Purview Account, the data will be moved from West Europe to East US 2, analyzed and the results, including source data, will be stored in a storage account in East US 2. For customers for whom moving the data would produce potential compliance issues, the workaround during the private preview is to create Purview accounts in regions where the data needs to remain and to only process the data in that region. The current product roadmap is that at General Availability (GA), Purview DQ will be able to analyze and store this information in region specific stores. In other words, the plan is that once Purview DQ reaches GA if a Purview Account in East US 2 is told to run a DQ scan on data in West Europe then a data analysis cluster in West Europe would be used to process the data and the samples and detailed error information would be stored in a storage account in West Europe. Please contact purviewdqpripsupport@microsoft.com with any questions.

Losing History – As part of the development of the private preview at one or more points all history of past runs may be deleted. This should not result in the loss of any rule definitions.

CSV Limitations – We do not currently support getting the schema for CSV files and so can't run rules on CSV files.

Unique Rule Limitation – We currently only support at most two columns having unique rules assigned to them on the same asset.

SQL Limitations – Column types of `sql_variant` in Azure SQL Database cannot be scanned. Columns of type `sysname` and `uniqueidentifier` numeric are not supported in Azure SQL DW.

Delta Lake Limitations – Currently we only support delta lake files whose Purview fully qualified domain name ends in `{SparkPartitions}`.

Parquet Maps, Lists & Structs – Microsoft Purview DQ does not currently support specifying DQ rules on maps, lists and structs in Parquet data structures. Please let the Microsoft Purview team know if this is a problem so fixing it can be properly prioritized.

Case insensitive column names – Microsoft Purview DQ does not support case insensitive column names for the purposes of matching a rule to a column in an asset. So "foo" and "FOO" are different column names to Microsoft Purview DQ even if the underlying asset is case insensitive.

Column names with `\n` or `\r\n` – If an asset contains a column name that contains a new line or similar control character will break the rule definition format.

Column names with acute accent character – We currently do not support rules whose names contain acute accent characters.

Simple Resource Set Support – Right now if we are pointed at a resource set our behavior is NOT correct. What we will do is take the resource set's fully qualified name (for example, `https://something/more/{guid}/something/{n}-whatever.parquet`) and we will look for the first segment starting from the left that contains `{}` and we will then step one segment to the left of that and scan everything under that folder that ends in `.parquet`. In the example URL we would start from <https://something/more> (since directory `{guid}` is a pattern match and contains `{}`) and we would take every file under `/more` that ends in `.parquet` and try to scan it. Note that if the parquet files do not have the exact same schemas, then everything will break. Also note that if the folders contain any files that are not parquet files then the data quality scan may fail. We are also still working on getting clearer error messages when this happens. Therefore, each set of files that belong to the same resource set should be output into a single directory with no extraneous content. Please note that this issue does not apply to Delta Lake files where we use the transaction log rather than the resource set definition to determine which files are members of the asset.

Duplicate Fully Qualified Names – It is possible to create a situation in which two assets in Purview have different types but the same fully qualified name. Please note that this is not a common situation. Currently Data Quality indexes DQ information based just on the FQN and not the combination of data type and FQN. This issue should only apply to scans and at worst would create a situation in which only one of the assets with the matching FQN but different types would be scanned. We believe this is a very rare situation and so are not prioritizing fixing this even for Public Preview.

5 Private Preview Goals

To help make progress in the private preview we have outlined below some goals that those joining us in the private preview can use to help them navigate through the private preview:

Goal	Description
Setting up the team	Get at least two to three users who are assigned the DQ Curator role on collections with assets to be tested during the private preview. Get at least one user who has Data Source Admin and DQ Curator rights on the collection so that scheduled scans can be set up.
The local expert	Have at least one person in the organization go through the starter kit below so that they have familiarity with all the features offered by the DQ Private Preview.
Scan an asset	Pick at least one asset and get rules on at least 6 columns. Then manually run a scan on the asset.
Scheduled quality scans	Set up a daily scheduled quality scan run that includes the assets that are being experimented with.
Examine History	After running DQ scans on an asset several times, use the history feature to go back in time and see old runs, their rules and scores.

6 Supported DQ Regions

The private preview is currently intended to be used with dedicated test Purview accounts that are created in the following regions:

- Canada Central
- East US 2

- Japan East
- North Europe
- US South Central
- West Europe

7 Starter Kit

The purpose of the starter kit is to walk through an end-to-end scenario for Data Quality in Microsoft Purview. In this starter kit we will first load sample data, register and scan it, specify rules for it, run those rules manually, examine the results and then schedule a run of those rules on a regular basis.

7.1 Step 1 – Get Data Ready To Use

Our goal in this section are to:

1. Find or create an ADLS Gen2 storage location to use to hold the test files where we will run DQ on
2. Give Microsoft Purview permission to access the ADLS Gen2 storage location
3. Connect to the storage location where we have put the test files and copy them over to the storage location in step 1

7.1.1 Find or create an ADLS storage location to use to hold our test files

We need a location to hold our test files. If you already have an ADLS Gen2 account just use that.

Otherwise, we'll need to create one. If you do need to create one then please head over to

<https://portal.azure.com> and:

1. Click on "+ Create a resource"
2. In the search bar type "Storage account" and click on "Storage account" in the results
3. Click create
4. Pick the subscription you will use for your storage account and resource group, specify a storage account name (pick whatever region, perf or redundancy you would like, it really doesn't matter for this test) and click "Next: Advanced >"
5. In the section titled "Data Lake Storage Gen2" click on the box marked "Enable hierarchical namespace"
6. Now click "Review + create"
7. Take a look at the summary and click "Create"
8. Now wait until you get confirmation that the storage account has been created

Please remember the storage account name, we are going to need it.

7.1.2 Give Microsoft Purview permission to access the ADLS Gen2 storage location

If you haven't connected your ADLS Gen2 storage account to your Microsoft Purview DQ test account, now is a good time to do it. If you just created your storage account then you might still have the "Your deployment is complete" screen so you can hit "Go to resource". Otherwise go to

<https://portal.azure.com> and type in the name of your storage account and click on its entry in the results to navigate you to the storage account's Azure Portal screen. Then

1. Go to "Access Control (IAM)" for your storage account
2. Go to the "+ Add" button and choose "Add role assignment"

3. In the “Add role assignment” screen set role to “Storage Blob Data Reader” and click next.
4. Under Members make sure “User, group, or service principal” is selected and then click on “+ select members”. Type in your Purview account’s name, click on it in the drop down and then click select.
5. Click “Review + assign”
6. Click “Review + assign” a second time

7.1.3 Connect to the storage location where we have put the test files and copy them over to the storage location connected to Microsoft Purview

In this section we will use the Microsoft Azure Storage Explorer. This is a very useful utility for examining and working with Azure Storage. You can download the explorer application at [Azure Storage Explorer – cloud storage management | Microsoft Azure](#).

First, we need to load the location where we will store the test data. This is the storage location you gave Microsoft Purview read permissions to.

1. Open Azure Storage Explorer



2. Click on
3. Click on “Subscription”
4. Click on Azure and click next
5. Go through the sign in experience
6. This returns us to the explorer except now it’s on the “Account Management” pane where we select (e.g. click the box next to) the Azure subscription in our tenant that contains the storage account (e.g. the one we created above or otherwise identified)
7. Then click on the “Open Explorer” button which shows a list of connected subscriptions.
8. Click on the desired subscription (e.g. the one we selected in step 5) and then click on “Storage Accounts”
9. From the drop down select the storage account that you connected to Microsoft Purview
10. Click on the arrow by the storage account and in the drop down click on Blob Containers
11. If you don’t already have a container under Blob Containers you want to use then select it otherwise right click on the Blob Containers entry and select “Create Blob Container” and then give the new blob container a name.
12. A recommendation is that you right click on the container’s name in the explorer view and select “Add to quick access”. This will put the container into the Quick Access section.
13. Once you have selected your Blob Container then click “+ New Folder”, type in a name and click Ok.

So now we need to connect to the location of the test files.



1. Click on
2. Select “ADLS Gen2 container or directory”
3. Click on “Shared access signature URL (SAS)” and click next

4. Type in a display name of “Microsoft Purview DQ Test files” (or whatever you want, it really doesn’t matter) and copy the following URL into the section titled “Blob container or directory SAS URL”:
<https://yarongqualitypurview.blob.core.windows.net/yarongroot/DQTestFiles?sv=2020-02-10&st=2021-08-31T20%3A37%3A21Z&se=2025-01-01T21%3A37%3A00Z&sr=d&sp=rl&sig=cPhzNJQccDsrJPf0C8tf0x%2BISbDuAbMXe0KD70a6W7U%3D&sdd=1>
5. Click next
6. You should now see a summary screen, click Connect

If all went well you should now be looking at a tab that shows two directories. Something like:

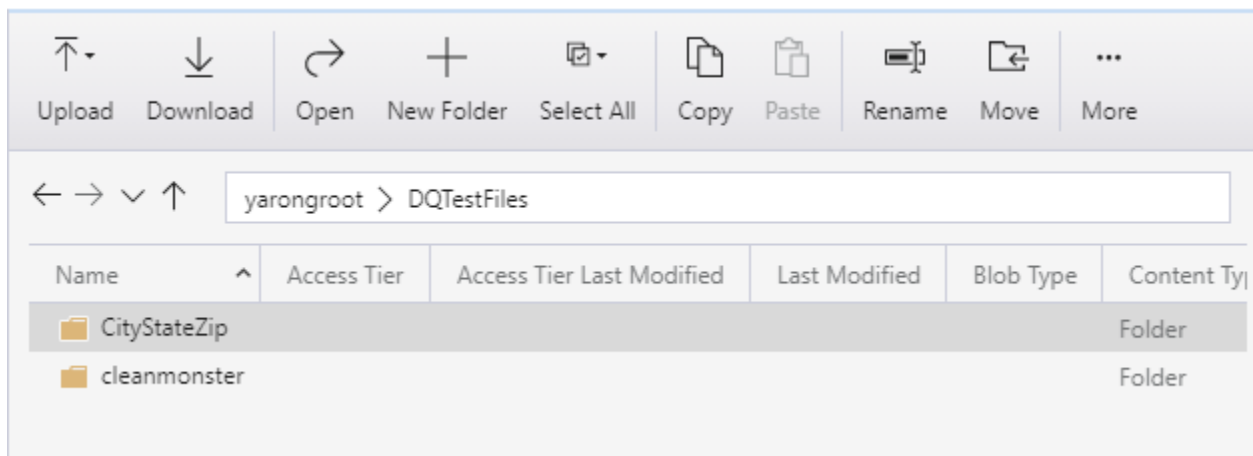


Figure 1 – Azure Explorer Data Files

These directories contain the test data for the starter kit.

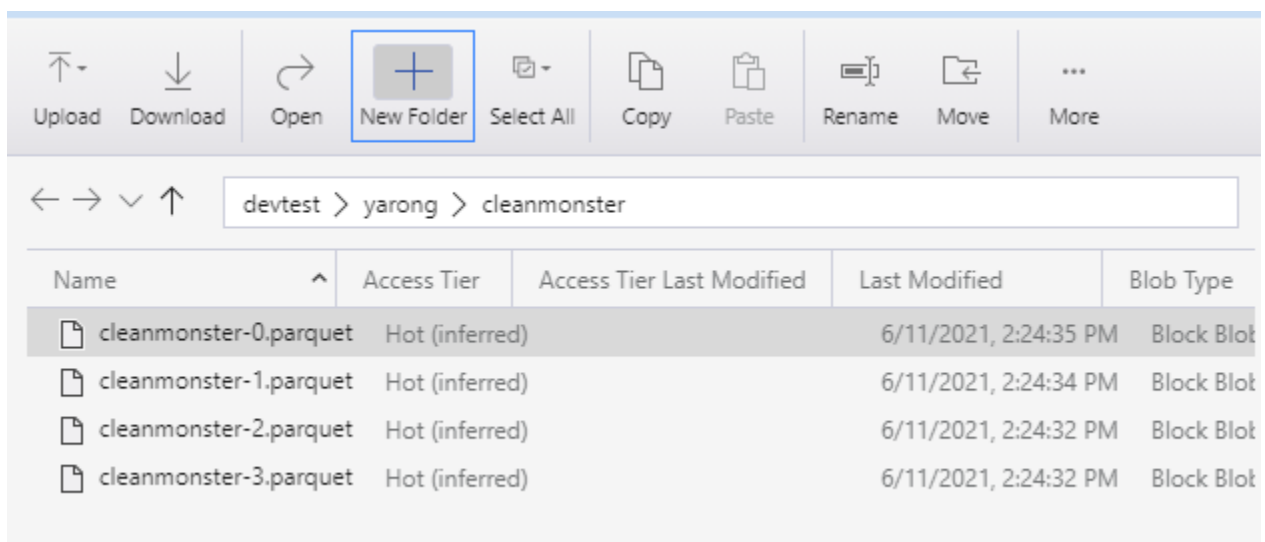


Figure 2 – Azure Explorer view of cleanmonster contents

Inside the cleanmonster directory are four files are taken from the CSV in [Perfect Dataset to get the hands dirty ! | Kaggle](#). These files are data taken from monster.com. All we did was just translate them to a parquet file format.

In the CityStateZip directory is a file called “citystatezip.snappy.parquet” that was generated from data provided at <https://simplemaps.com/data/us-zips>.

7. Now select the two directories and click the copy button
8. Ideally you will still have the tab left open where your ADLS Gen2 account that you connected to Microsoft Purview is, but if not then go to Quick Access (if you followed our hint above) and click on the container name there and navigate down to the folder you want to put the test files into. Please make sure there is nothing else in the folder you are going to use.
9. Click on Paste.

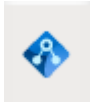
If all went well you should now see the two directories and their contents in your storage account.

7.2 Step 2 – Scan sample data

In order to scan data the user initiating the scan must be in the Data Source Admin role and the DQ Curator role on the collection that the data will be stored in. Please see section 8.2 for details.

If you already have a scan set up for the ADLS Gen2 account then just manually run it now to update the Microsoft Purview Account’s state. Otherwise follow these instructions:

1. Go to your Microsoft Purview account portal (the one that DQ is enabled on)



2. Go to Datamap (click on)
3. Click Sources and then Register
4. Select “Azure Data Lake Storage Gen2” and click “Continue”
5. From the Register Sources screen you can jump to “Storage account name” and just type in the storage account’s name, select it from the search list and click Register
6. Select the scan icon on the storage account in the data map view ()
7. Click Continue
8. It’s easiest to just scan everything but if necessary you can use the tree view to just click on the folder where the test data was copied then click Continue
9. The SYSTEM DEFAULT AdlsGen2 scan rule set should be fine so just click Continue
10. Set the scan trigger to once and click Continue
11. Click Save and Run
12. Click on “view details” on the card with the storage account you are scanning
13. You should now see your scan listed under “Recent scans” with its status. It probably shows Queued or In Progress. You will need to hit the “Refresh” button to get an update on its status. Keep refreshing until it says “Completed”

7.3 Step 3 – Find sample data, set up rules and run a data quality scan

In order to run a DQ scan the user must be in the DQ Curator role on the collection that contains the data. Please see section 8.2 for details.

1. Go to the “search assets” box at the top of the Microsoft Purview portal page and type in “cleanmonster”
2. In the results you will see a Resource set with a URL that ends in “cleanmonster-{N}.parquet”. Please click on that resource set’s name in the search results.
3. Click on the Data quality tab to set up rules.

Before we continue it’s a good idea to stop for a moment and look at the actual data. The data comes from Monster.com and represents job postings. This is just a sample from a much larger file.

All the jobs have the country “United States Of America” and the country_code US. Which is fine but we should at least validate that none of the values are null and that the values are set as expected.

There is a date_added column that is mostly empty so we won’t check it for nulls but we should validate that it contains dates.

The has_expired column is just no’s so we’ll ignore it for now.

The job_board always has the value “jobs.monster.com”. But we’ll play with that a little to show how we can validate it with a like pattern. More information on that below.

Job_description, job_title and job_Type are open text fields but they should have values so we’ll check them for nulls.

Location seems to have two main variants and a lot of exceptions. One variant is “city, state” and the other is “city, state zip”. So we’ll use a regular expression to check for both and flag any violators. There should always be a location so we’ll check for nulls.

Organization is another open field so we’ll just make sure it has a value and is of type string.

Page_url should never be null and its value should be unique. E.g. every job posting should have its own page_url. What’s interesting about page_url is that it contains the city, state and country of the job posting. So this is a good chance for us to use a custom rule. The structure isjob-[City]?[- more city]-[Two Letter State]-US-.... So we can use this to compare against the state in the location column.

Salary is clearly optional but it should contain a string so we’ll at least check for that.

Sector, like Salary, also looks optional but it should contain a string.

Last is uniq_id is a 32 digit ID. The easiest way to validate that is with a simple custom rule (o.k. Regular Expression could do it easily too but we already used that).

In this case our data set is reasonably small with only a few columns but most data sets have many columns and it can be really useful to identify which columns need attention. This is where the star feature comes in. The star feature lets one click on a star by each column that needs DQ attention. This doesn’t change any state for the column, it just puts it first in the default sort ordering to make it easier to find. So feel free to click on a few stars but keep in mind that the only column we won’t be creating a rule on is has_expired.

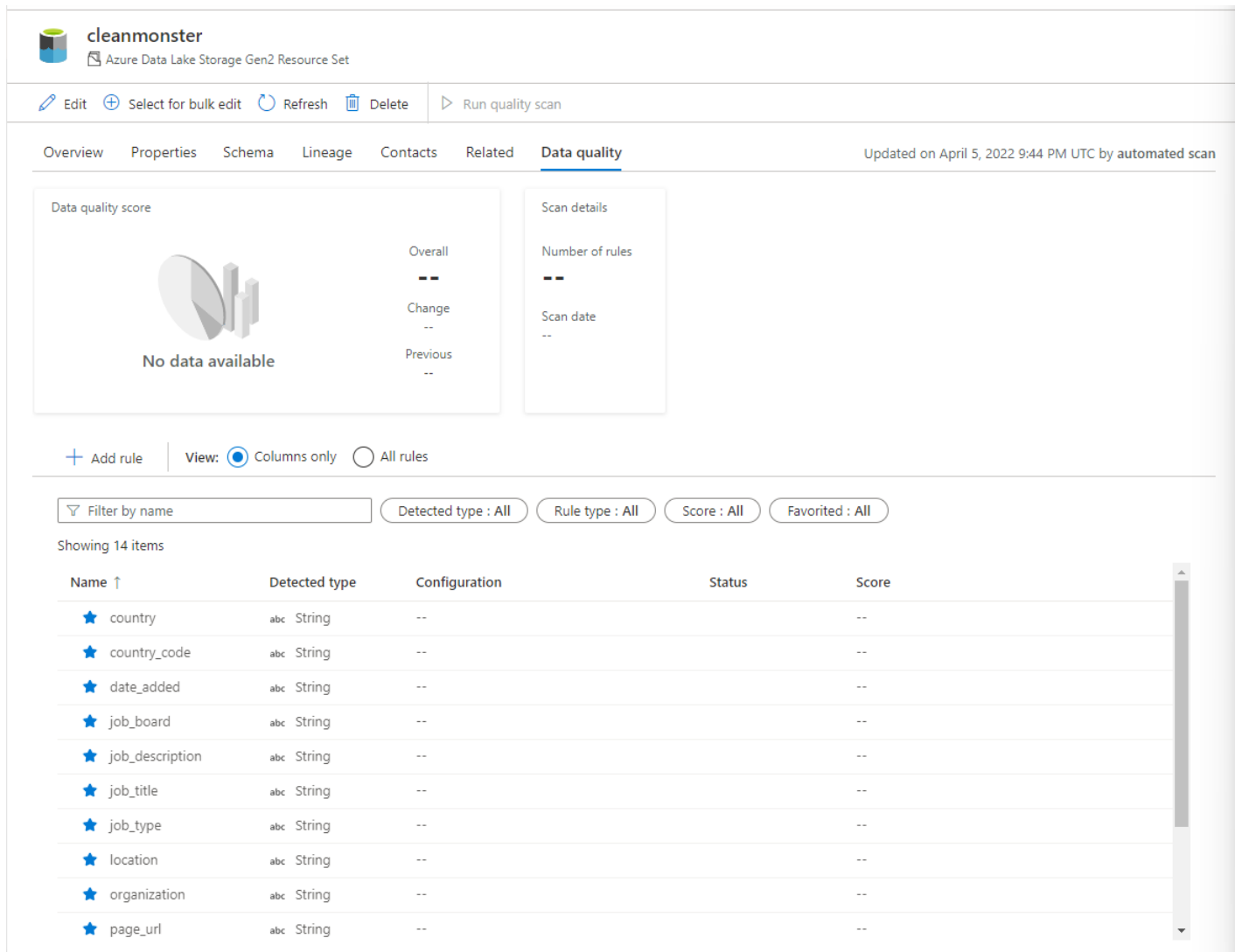



Figure 3 Example of stored columns

But now we can start to add in rules to validate the system's behavior. Let's start by checking that the data types for the columns have the values we are looking for:

1. Click on "+ Add rule"
2. Click on Data type match and then click on Next
3. Under columns click on all columns but has_expired to add them to the selected tab
4. All the columns but date_added are strings and that is the default so the only type we need to change is date_added, please change it to date.

Rule details



Data type match

Verifies that values in a column match their data type requirements.

Description

Columns *

Add all columns

uniq_id

Column name	Data type	Example	13 columns selected
country	abc string	e.g.,	
country_code	abc string	e.g.,	
date_added	date	e.g.,	
job_board	abc string	e.g.,	
job_description	abc string	e.g.,	
job_title	abc string	e.g.,	
job_type	abc string	e.g.,	
location	abc string	e.g.,	
organization	abc string	e.g.,	

Publish

Back

Cancel

Figure 4 - Applying data type matches to columns

5. Click on Publish

We now see an updated screen where each column but has_expired has a (1) by it indicating that it has one rule defined for it. But another way to see things is to click on view: All Rules. This will show that we have just one rule type we have used so far, Data type match, and we have applied it to 13 columns.

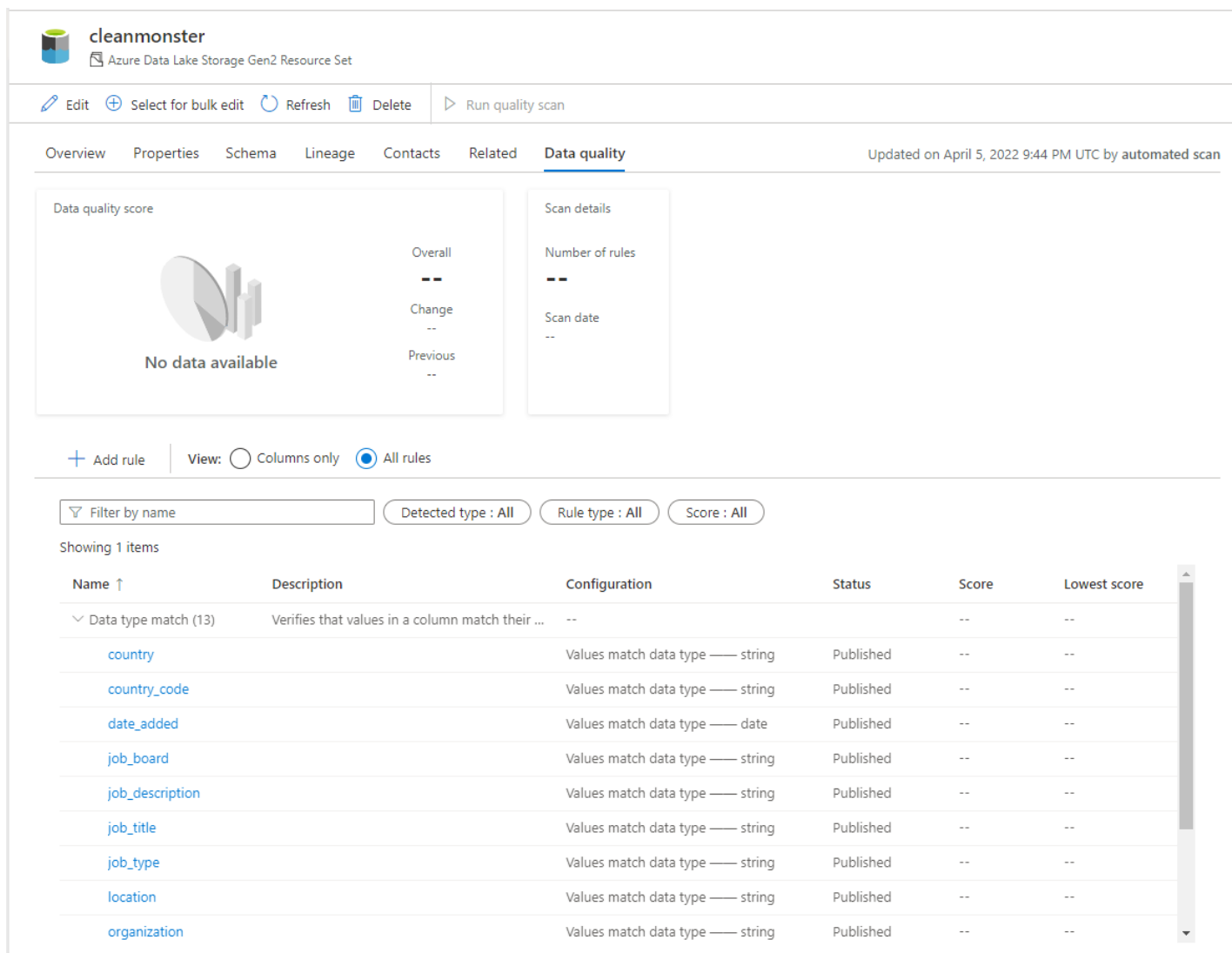



Figure 5 - All rules view of Data type match rules

Certain columns should always have values. So let's add in a check that these columns are not empty.

1. Click on "+ Add rule"
2. Click on Empty/blank fields and click Next
3. Click on the following Columns: country, country_code, job_board, page_url, uniq_id
4. Click on Publish

Rule details

**Empty/blank fields**
Looks for blank and empty fields in a column where there should be values.

Description

Columns *

5 selected

☒ country
☒ country_code
☒ job_board
☒ page_url
☒ uniq_id
☐ date_added
☐ has_expired
☐ job_description
☐ job_title

Publish

Back

Cancel

Figure 6 - Selected columns for Empty/blank fields rule

There are also two columns who values should always be unique, so let's put in rules for that:

1. Click on "+Add rule"
2. Click on Unique values and click on Next
3. From the columns drop down select page_url and uniq_id
4. Click on Publish

Rule details

Unique values
Confirms that values in a column are unique.

Description

Columns *

2 selected

- ☒ page_url
- ☒ uniq_id
- ☐ country
- ☐ country_code
- ☐ date_added
- ☐ has_expired
- ☐ job_board
- ☐ job_description
- ☐ job_title


Publish | **Back** **Cancel**

Figure 7 - Selected unique columns

The next step is to make sure that the country column contains the right value. The sample that we picked only has jobs posted in the United States of America. So let's validate that.

1. Click on "+ Add rule"
2. Click on "Format match" and click on Next
3. From Columns pick country
4. From Validation type pick Enumeration
5. In Validation criteria type in: United States of America
6. Click on Publish

Rule details



Format match
Confirms that values in a column match a specific format or other criteria.

Description

Columns *

country

Validation type *

Enumeration

Validation criteria * ⓘ

United States of America

Values will be checked if they are in the list:
United States of America

Publish

Back


Cancel

Figure 8 - Format match for country column

Another type of format match is a Like pattern. This is a simpler version of regular expression where the pattern is treated as just a string but with two wild card characters, `_` matches any single character and `%` matches zero or more characters.

1. Click on “+Add rule”
2. Click on Format match and next
3. From Columns select `job_board`
4. From Validation type select Like pattern
5. In Validation criteria enter: `_%._%.com`. This is a simple pattern that just says that the value should be a domain name.
6. Click on Publish

Rule details



Format match

Confirms that values in a column match a specific format or other criteria.

Description

Columns *

job_board

Validation type *

Like pattern

Validation criteria *

_%.%.com

Publish

Back


Cancel

Figure 9 - Format match on job_board

One last Format match example just to illustrate all our capabilities. This one shows a regular expression that we can use to validate that the location column contains either a city, state or a city, state zip.

1. Click on "+ Add rule"
2. Click on Format Match and Next
3. From Columns select location
4. From Validation type select Regular Expression
5. In Validation criteria put in: `.*, [A-Z]{2}([0-9]{5})?`
6. Click on Publish

Rule details

**Format match**Published

Confirms that values in a column match a specific format or other criteria.

Description

Confirms that values in a column match a specific format or other criteria.

Columns *

location

Validation type *

Regular expression

Validation criteria *

.*, [A-Z]{2}([0-9]{5})?



Publish |  Cancel

Figure 10 - Format match for location column using a regular expression

The location column contains city and state or sometimes city and state and zip for where the job is listed. We want to validate if those values are correct so we can use the Table Lookup rule to let us validate the values in this table against a remote table that contains a list of all the city, state and city, state, zip combinations in the US.

1. Click on "+ Add rule"
2. Click on Table lookup and click Next
3. Click on location from the Columns drop down
4. Search in Asset for "citystatezip.snappy.parquet" and select when found
5. For Column under Comparison table please select "Data".
6. Click on Publish

Rule details

 **Table lookup** Published

Confirms that a value in one table can be found in a specific column of another table.

Description

Confirms that a value in one table can be found in a specific column of another table.

Columns *

location

Comparison table

Asset *

citystatezip.snappy.parquet

Column *

abc Data (string)

Publish


Cancel

Figure 11 - Table lookup rule example

Now let's look at some custom rules. These enable us to use expressions to specify our validation logic. In this case we will just check that the uniq_id column's values are all 32 characters long.

1. Click on "+ Add rule"
2. Click on Custom and Next
3. Give the rule a name such as: Uniq_ID Length Validation
4. In expression paste: `length(uniq_id)==32`
5. Click on Publish

Rule details



Custom

Create a custom rule with the visual expression builder.

Rule details

Rule name *

Uniq_ID Length Validation

Description

Row expression *

1

length(uniq_id)==32

☐ Use filter expression

+

-

*

/

||

&&

!

^

==

===

<==>

!=

>

<

>=

<=

[]

Expression elements

All

Functions

Columns

Expression values

Filter by keyword

123

abs(123 number)

123

acos(123 number)

ANY

add(ANY any,ANY any)

addDays(date_or_timestamp,ANY days_to_add)

addMonths(date_or_timestamp,ANY months_to_add,abc string)

Publish

Back

Cancel

Figure 12 - Simple custom rule example

Next let's do a slightly more sophisticated custom rule. It turns out that the `page_url` contains the name of the city that the job was posted in, if the job has a city. So what we want to do is first look for rows that have a location value. If they don't have a location value we want to ignore them, we can use the filter expression to let us ignore rows that don't meet our requirements. Then we will put in a row expression that will compare the city in the location to the city in the `page_url` and make sure they are the same.


1. Click on "+ Add rule"
2. Click on Custom and Next
3. Put in a rule name like "City Validation"
4. Click on the box "Use filter expression"
5. In the Filter expression box paste in: `isNull(location)==false()`

In the Row expression box paste in: `upper(regexExtract(page_url, '.*-([a-zA-Z]{2})-(us|US)-.*',1)) ==`

`upper(regexExtract(location, `.*, ([A-Z]{2})([0-9]{5})?`,1))`

6. Click Publish

Rule details

**Custom**
Create a custom rule with the visual expression builder.

Rule details

Rule name *

City Validation

Description

Row expression *

☒ Use filter expression

```
1 upper(regexExtract(page_url, '.*-([a-zA-Z]{2})-(us|US)-.*',1)) ==
2 upper(regexExtract(location, `.*, ([A-Z]{2})([0-9]{5})?`,1))
```

Filter expression

```
1 isNull(location) == false()
```

Expression elements

All

Expression values

Filter by keyword


Publish Back Cancel

Figure 13 - Example of more complex custom rule

There are two more rule types to explore. The next one is a duplicate rows rule. This lets us see if the combination of values from two or more columns are unique when combined together.

1. Click "+ Add rule"
2. Click on "Duplicate rows" and Next
3. From the Columns drop down select page_url and uniq_id
4. Click on Publish

Rule details



Duplicate rows

Checks for duplicate rows with the same values across two or more columns.

Description

Columns *

2 selected

☒ page_url

☒ uniq_id

☐ country

☐ country_code

☐ date_added

☐ has_expired

☐ job_board

☐ job_description

☐ job_title

Publish

Back


Cancel

Figure 14 - Duplicate rows rule example

Last, but not least, we have the Freshness rule. This lets us validate that the asset was last updated within a certain period of time.

1. Click on “+ Add rule”
2. Click on Freshness and Next
3. Set Last updated within “1” and set Unit to days
4. Click Publish

Rule details



Freshness

Confirms that all values are up to date.

Description

Last updated ⓘ

Last modified date

Timeframe *

Custom time frame

Last updated within

Amount

1

Unit

Days

of data quality scan start time.

Publish

Back

Cancel

Figure 15 - Freshness rule example

Now we are ready to start running the data quality scan. To do so just click on the “run quality scan” button. Once the scan is triggered a “Scan in progress” warning will be presented and a “Stop scan in progress” button will be available to terminate the scan early if desired. The screen will automatically refresh with state so when the scan is done a notification will be shown and the “Scan in progress” warning will disappear.

Once the scan is over one should see a screen that looks something like:

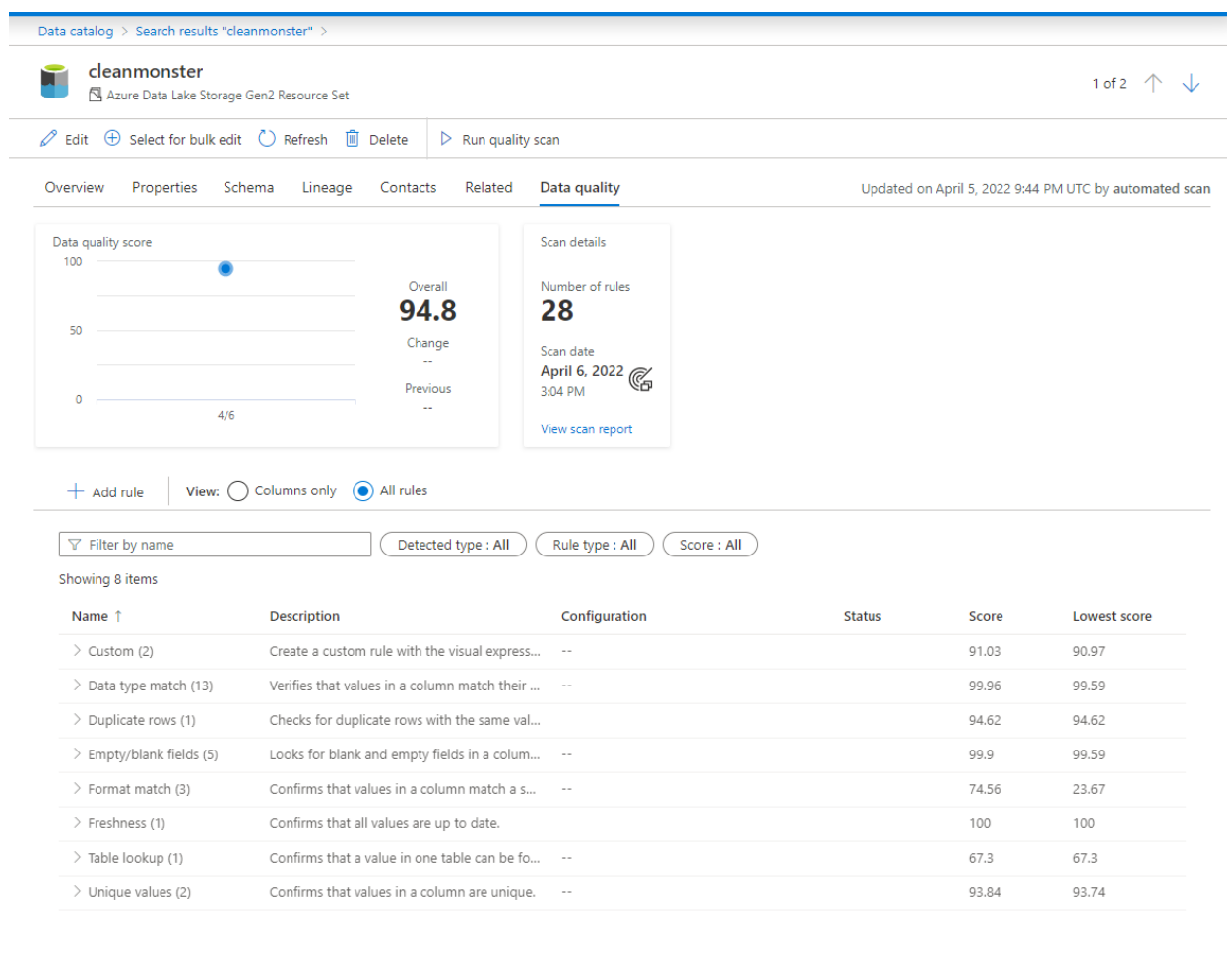


Figure 16 - Sample output of Data Quality run

Note that your exact score may be different due to a variety of factors.

At this point one can explore. Right now the screen is set to “All rules” so we can see all the rule types. Right away the Table lookup rule type jumps out with a low score of 67.3 but we also see an issue with Format match which has an average score of 74.56 but a lowest score of 23.67.

If we click on “Format match” we can see that it is location that has the lowest score. We can even hover over the (i) icon by the score and see that 16786 rows failed the test. So clearly we have a problem. We can also click on Table lookup and see that location failed 7191 rows. So there is a real issue here.

And, no surprise, if we change the view to “Columns only” we see that the lowest scoring column is location.

The fastest way to understand what went wrong is to look at the detailed error results. To see those we click on “View scan report” and then the “Error report” button. This will download a CSV file listing the first 500 error rows. An error row is any row in the source asset that failed at least one rule.

The columns from A to N are the original data taken from the source asset. The remaining columns identify particular rules and indicate if they passed, failed or were skipped for reasons such as null/empty being allowed or a filter expression in a custom rule.

So if we look for the column titled “location – Table lookup” we can see a large number of failures. In scrolling down the “fail” value and looking to the location column we are expecting values of the form “city, ST zip” but we see many values that are random strings. The reason for this is that the original monster data was taken from a CSV file but at the time this demo was first created we didn’t have the CSV support we now have, so the CSV file was translated to a Parquet file. Well it turns out that the CSV to Parquet translator didn’t properly handle some of the escape sequences in the original file and that caused data corruption. We hadn’t caught this at the time.

Another interesting lesson from the location - Table lookup column is that proper looking values like “Des Moines, IA 50301” failed (note that your error file may or may not contain this particular row since we just sample 500 error rows). Which is odd because 50301 is Des Moines, IA. What we didn’t realize when we grabbed the data set we use as our reference data is that the data set is only a sample of all the city, state and zip codes in the US. It’s not the complete set. So by looking at the error details we could actually see this and understand that we have DQ issues both in the data (e.g. the Monster.com job posting) and in our reference data (e.g. the reference list of city, state and zip). By engaging with the detailed errors we were able to not just find DQ issues with the data we were evaluating, but also the data we were depending on.


7.4 Step 4 – Scheduling regular data quality scans

Above when we clicked on the “Run quality scan” button that ran a single quality scan, exactly one time. But if this were a production scenario it’s likely that our source data is being constantly updated and so we want to make sure we are regularly monitoring its data quality in order to detect any issues. To enable us to manage regularly updating quality scans we can use the Data Quality scan capability.

In order to run this section one must be both a Data Quality Curator and a Data Source Admin on the collection that the scan is to be configured on. Please see section 8.2 for details on how to add someone into these roles.

This allows us set up regular scans on a predefined schedule. So let’s set up a scan for our test data.



1. Click on the datamap icon ()
2. Click on the Data quality tab
3. Click on Schedule quality scan button
4. In the Schedule quality scan pop up provide a name like “Test scan”
5. For Data source select the ADLS Gen2 account, for credential we can use the default MSI since we used MSI to set up this example, now please click on continue
6. If the test connection works then we will be taken to the Scope tab where we can see all folders in the ADLS Gen2 account. Feel free to either select the individual file or the entire data lake. The scan will only apply to assets that have rules on them so selecting the entire ADLS Gen2 account will only apply to the test asset if that is the only asset with rules. Click Continue.

7. We should set the scan to recurring since that is a good test of the system. We can set the Recurrence to days and to run at 1 AM UTC with no end date. Please see below and then click Continue.

[Overview](#) [Scope](#) [Schedule](#) [Review](#)

Set a assessment trigger to run the assessment at specific dates and times. If once, the assessment will start after set up is completed. If recurring, the assessment will start at a date and time you choose.

☒ Recurring ☐ Once

Time zone * ⓘ

(UTC) Coordinated Universal Time

Recurrence *

Every 1 Day(s)

Schedule scan time (UTC)

1:00:00 AM

Start recurrence at (UTC) *

2022-02-07 5:41:00 PM

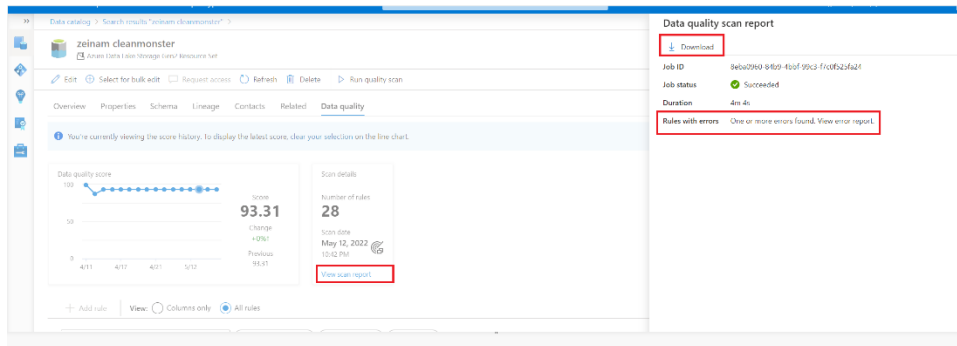
☐ Specify recurrence end date (UTC)

8. We can then, on the review pane, either click Save or even better is Save and run (click the drop down to see that option) since that tests everything immediately.

7.5 Step 5 – Viewing the results of previous Data Quality scans

Being able to view the history of already executed Data Quality scans is a critical feature in order to understand and analyze the trend of the Data Quality score over multiple scans.

The trend of the Data Quality Global Score is the key indicator which enables understanding of the change in the Data Quality score over time. Each scan run generates the Data Quality scan report as well.



8 Reference Documentation

8.1 Supported Asset types and authentication methods

- ADLS Gen2
 - File Types: The recommended file type is **Delta Lake**. But we also have support for Parquet and CSV.
- Azure SQL and Azure SQL Managed Instance

Currently Purview DQ can only authenticate itself to a data source in order to run a quality scan using a MSI.

8.2 Enable Azure Trusted Services if a data source is behind a VNet/Private Endpoint

This section applies if you are planning to scan an Azure SQL DB or ADLS Gen 2 account that is in a VNet, secured behind a private endpoint.

8.2.1 How to enable Azure Trusted Services for an Azure SQL Database

Perform the following steps on each Azure SQL Server resource that you are planning to onboard with data Quality inside Microsoft Purview:

1. Use a tool such as **SQL Management Studio** to grant Purview MSI with **db_datareader** access in each database on the Azure SQL Server.

```
CREATE USER "<puview_account_name>" FROM EXTERNAL PROVIDER;
EXEC sp_addrolemember 'db_datareader', "<puview_account_name>"
```

2. From the Azure portal, navigate to the Azure SQL Server resource, select **networking**.
3. Under Public access, enable **selected networks** and then turn on “**Allow Azure Services and resources to access this server**”.

For more information, see [Azure SQL Database and Azure Synapse Analytics network access controls](#).

8.2.2 How to enable Azure Trusted Services for ADLS Gen 2

Perform the following steps on each Azure Storage account resource (containing ADLS Gen 2) that you are planning to onboard with Data Quality inside Microsoft Purview:

1. From the Azure portal navigate to the storage account, resource group or subscription, select **Access Control (IAM)**. Add Purview MSI to the **Storage Blob Data Reader** role if it has not been added already.
2. Select Azure Storage, navigate to **Networking**. Under Firewalls and virtual networks, select **Public network access: "Enabled from selected virtual networks and IP addresses"**.
3. In the same blade select any of the following options, depending on your network and security requirements:
 - **Option 1:** Enable **"Allow Azure services on the trusted services list to access this storage account"**.
 - **Option 2:** Under **Resource instances**, under **Resource type** select **Microsoft.Purview/accounts** and under Instance name select your Microsoft Purview account name.

Note: Optionally, you can add any Azure VNets under **Virtual networks**.

For more information, see [Grant access to trusted Azure services](#).

8.3 Data Quality Authorization Roles

Data Quality provides functionality through two interfaces, one is the asset details page, the other is through the Data map Management experience.

For a user to see the Data quality tab on an asset detail page they must be in the DQ Curator role for the container that asset is in.

For a user to see the Data quality tab on the Data map screen they need to be both in the DQ Curator role as well as the Data Source Admin role for the container that contains the asset they want to schedule a quality scan for.

The mechanism to assign people to roles, who can do the assignment, etc. is all explained in [Understand access and permissions in the Microsoft Purview Data Map - Microsoft Purview | Microsoft Docs](#).

8.4 Data quality tab on the asset details page

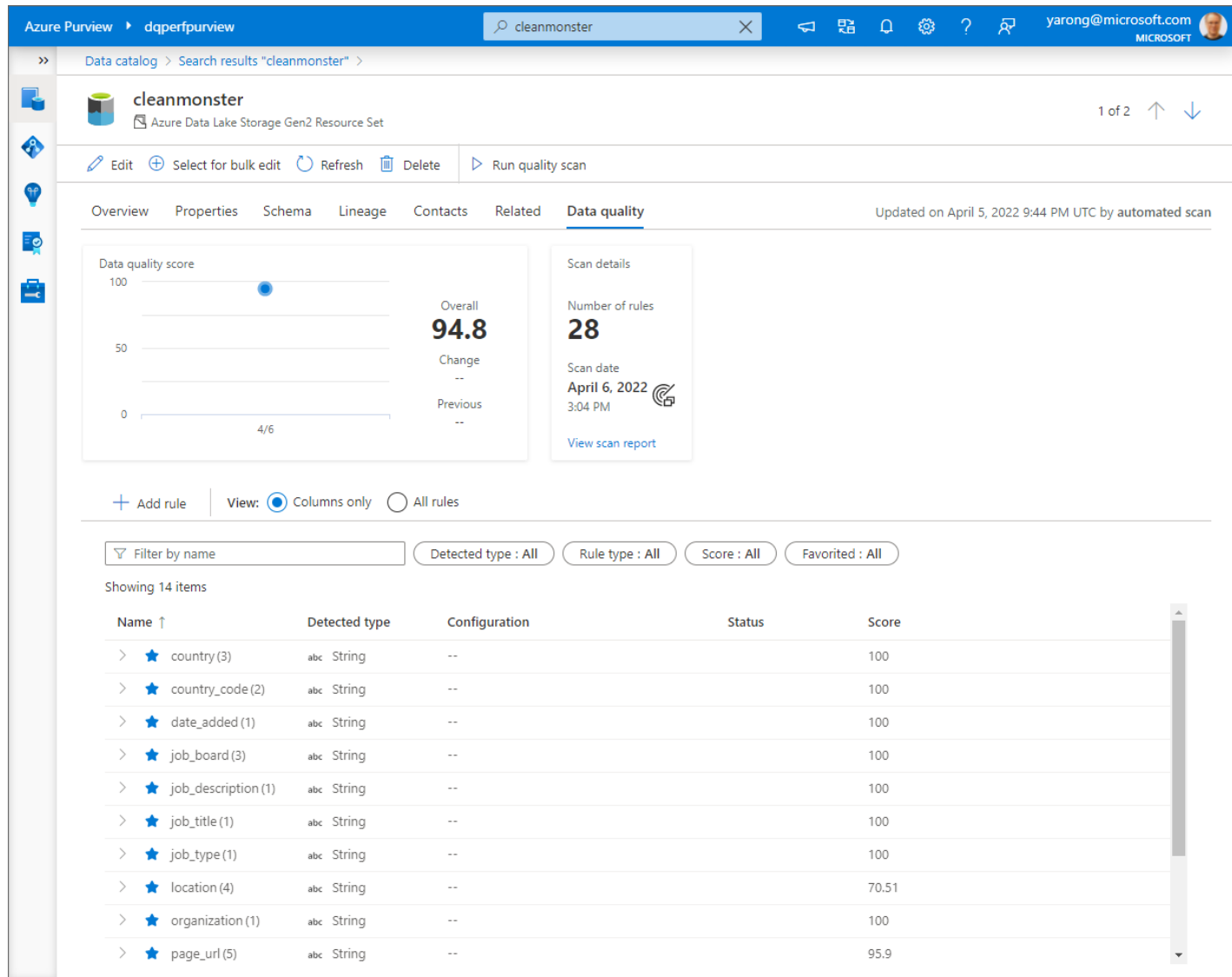


Figure 17 – Example of a data quality tab from an asset details page

The purpose of this page is to define the rules that specify the data's expected behavior. After a data quality scan is run, which is triggered by the "Run quality scan" button, each rule will have a score and that score will be combined to produce a global score. The details of the scoring calculations are given in section 8.8.

This screen provides basic context information including the global score (if any), the status of the last successful run and if the last run was a failure information about that failure. It also shows a graph of any historical runs to show how the global data quality score has changed over time.

To add a new rule one clicks on the "+ Add rule" button, selects the desired rule type and then defines the rule. More details on how this works for each rule type will be provided in section 8.5.

If there is at least one rule defined then the “Run quality scan” button will activate and one can run an analysis on all of the data against the rules.

Below that is information about the rules that have been defined to specify the expected behavior of the asset. Details on what those rules are and how to use them are given in section 8.5.

This screen provides two distinct ways to view rules, by “Columns only” and by “All rules”.

8.4.1 Columns only

In the previous screen rules are organized by “Columns only”. This provides a list of columns.

Each column has a detected type which is the type taken from the schema that was detected by Purview Data map scan.

If the column has rules defined on it and if a quality scan has been run with those rules then the column will have a score. The purpose of the score is to help quickly identify columns that may have quality issues that need attention. The column score is the average of the scores of the rules defined on that column.

If a column has at least one rule defined on it, then it will have a “>” character by it that can be clicked to open up the list of rules defined on the column.

Each rule will have a name, a summary of its configuration, a status (discussed in detail in section 8.7) and if the rule has been run as part of a quality scan, then it will also have a score. The score represents how many rows passed the rule divided by how many rows the rule was applied to. Note that in some cases rows will be ignored by a rule. For example, if a column doesn’t have the “Empty/blank fields” check defined on it then the column is implicitly allowed to have empty or blank fields. In which case rules like “Format match” would ignore those empty/blank rows. Next to the score will be an (i) icon that if hovered on will list how many rows passed this rule, how many failed the rule and how many were skipped.

Individual rules can be edited by either clicking on their name or hovering the mouse to the right of their name causing a pencil and trash can icon to appear. The pencil icon can be used to edit and the trash icon to delete the rule. Note that it’s possible for the same rule to appear twice if it applies to more than one column. For example, both the Duplicate rows and Custom rules can apply to multiple columns and each instance of such a rule will appear under all the columns it applies to.

Column names have a star by them that can be clicked on. This feature is used to identify columns that are intended to have data quality rules defined on them or are otherwise important. This is especially useful on tables that have hundreds of columns. One can use the “Filter by name” feature to find relevant columns and then star them. Starred columns will be put on the top of the default sort order. Please note however that the stars are just for ease of access and do not imply anything about the rules on that column. It is perfectly o.k., for example, to have a column with rules that is not starred and columns with no rules that are starred. The point of the star feature is just to promote to the top of the list those columns that should be interacted with first.

The columns view can be filtered using the “Filter by name” feature so that only relevant columns are shown. Alternatively there are built-in filters:

- Detected type – This will filter out all columns that do not have the selected types from the schema.
- Rule type – This will filter out all columns that do not have rules defined on them of the selected type.
- Score – This will filter out all columns that do not have scores in the selected range or “—” for columns that don’t have no score at all.
- Favorited – This will filter out all columns that do not have the selected favorite state, e.g., show all favorites columns or only columns that are not favorited.

Because this view only shows rules that are associated with columns, certain rules, like Freshness, that are not column oriented will not be visible in this view.

8.4.2 All rules

The All rules view shows all the rules on the asset organized by rule type. Unlike the Columns only view, this view shows all rules of all types defined on the asset.

The only rule types that will be listed in this view are rule types that have at least one instance already defined on the asset. Each rule type will have a short description describing what it does. One can click on the “>” character to display all instances of the listed rule type.

For rules like Data type match or Format match that apply to a specific column the list of rules will begin with the name of the column the rule is defined on. For rules that can only have one instance, like Duplicate rows or Freshness, the rule’s name will be used as its title. And for rules like Custom that require a user specified name, that name will be displayed.

After the name will be a short description of the rule’s configuration, followed by a status (discussed in detail in section 8.7) and if the rule has been run as part of a quality scan then it will also have a score. For row specific rules like Duplicate rows or Table lookup, the score represents how many rows passed the rule divided by how many rows the rule was applied to. Note that in some cases rows will be ignored by a rule. For example, if a column doesn’t have the “Empty/blank fields” check defined on it then the column is implicitly allowed to have empty or blank fields. In which case rules like “Format match” would ignore those empty/blank rows. Next to the score will be an (i) icon that if hovered on will list how many rows passed this rule, how many failed the rule and how many were skipped.

Some rules, like Freshness, are not row oriented and their score is calculated differently, details are described in section 8.5.

Individual rules can be edited by either clicking on their name or hovering the mouse to the right of their name causing a pencil and trash can icon to appear. The pencil icon can be used to edit and the trash icon to delete the rule.

8.5 Rules

To create a new rule one clicks on “+ Add rule”. This will bring up a screen with a list of all supported rule types. Note that certain rules have restrictions on them, for example, there can only be a single Freshness or Duplicate Rows rule. Rules that cannot have more instances of them created will still be listed but will be greyed out.

8.5.1 Custom rule

The screenshot shows the 'Rule details' page in the Azure Purview interface. The top navigation bar includes 'Azure Purview', 'dqperfurview', a search bar, and user information 'yarong@microsoft.com'. The left sidebar contains navigation icons. The main content area is titled 'Rule details' and features a 'Custom' rule type with the description 'Create a custom rule with the visual expression builder.' Below this, there are input fields for 'Rule name *', 'Description', and 'Row expression *'. A checkbox labeled 'Use filter expression' is located to the right of the 'Row expression *' field. The 'Row expression *' field contains the number '1'. Below the input fields is a red error message: 'Enter a row expression.' Below the error message is a toolbar with various operators: +, -, *, /, ||, &&, !, ^, ==, ===, <==>, !=, >, <, >=, <=, and []. Below the toolbar are two panels: 'Expression elements' on the left with tabs for 'All', 'Functions', and 'Columns'; and 'Expression values' on the right with a 'Filter by keyword' search bar and a list of functions including 'abs', 'acos', 'add', 'addDays', and 'addMonths'.

Figure 18 - Sample custom rule screen

The custom rule enables specifying rules that try to validate rows based on one or more values in that row. The custom rule has two parts.

The first part is the filter expression which is optional and is activated by clicking the check box by “Use filter expression”. This is an expression that returns a Boolean value. The filter expression will be applied to a row and if it returns true then that row will be considered for the rule. If the filter expression returns false for that row then it means that row will be ignored for the purposes of this rule. The default behavior of the filter expression is to pass all rows so if no filter expression is specified, and one is not required, then all rows will be considered.

The second part is the row expression. This is a Boolean expression applied to each row that gets approved by the filter expression. If this expression returns true then the row passes, if false, then it marked as not passing.


Both the filter expression and row expression are defined using the Azure Data Factory expression language as introduced [here](#) with the language defined [here](#). Note, however that not all the functions defined for the generic ADF expression language are available. The full list of available functions are in

the Functions list available in the expression dialog. The following functions defined in [here](#) are not supported: isDelete, isError, isIgnore, isInsert, isMatch, isUpdate, isUpsert, partitionId, cached lookup and Window functions.

Note that any regular expressions included in custom rules must be surrounded by `` characters and any special characters must be escaped. The regular expression language is based on Java and works as given [here](#). [This page](#) identifies the characters that need to be escaped.

8.5.2 Data type match rule

Rule details

**Data type match**
Verifies that values in a column match their data type requirements.

Description

Columns * [Add all columns](#)

date_added

Column name	Data type	3 columns selected
country	<div>abc string</div>	<div></div>
country_code	<div>abc string</div>	<div></div>
date_added	<div>date</div>	<div></div>

Publish

Back

Cancel

Figure 19 – Example of Data type match rule

The Data type match rule specifies what data type the associated column is expected to contain. Because the rule engine has to run across many different data sources it can't use native types like BIGINT or VARCHAR. Instead it has its own typing system that it translates the native types into. This rule

tells the quality scan engine which of its built in types the native type should be translated to. The data type system is taken from the Azure Data Flow type system used in Azure Data Factory.

During a quality scan all the native types will be tested against the data type match type and if it isn't possible to translate the native type into the data type match type that row will be treated as being in error.

8.5.3 Duplicate rows

Rule details

Duplicate rows
Checks for duplicate rows with the same values across two or more columns.

Description

Columns *

2 selected

- ☒ page_url
- ☒ uniq_id
- ☐ country
- ☐ country_code
- ☐ date_added
- ☐ has_expired
- ☐ job_board
- ☐ job_description
- ☐ job_title

Publish | **Back** | **Cancel**


Figure 20 - Duplicate rows check

This rule checks if the combination of the values in the column are unique for every row in the table. In the previous figure the expectation is that that the concatenation of “page_url” and “uniq_id” will produce a value that is unique for all the rows in the table.

Each asset can have zero or one instances of this rule.

8.5.4 Empty/blank fields

Rule details



Empty/blank fields
Looks for blank and empty fields in a column where there should be values.

Description

Columns *
5 selected

☒ country

☒ country_code

☒ job_board

☒ page_url


☒ uniq_id

☐ date_added

☐ has_expired

☐ job_description

☐ job_title

Publish 

Back


Cancel

Figure 21 - Nulls not allowed

This rule asserts that the identified columns should not contain any null values and in the specific case of strings, no empty or whitespace only values either. During a quality scan any value in this column that are not null will be treated as correct. This rule will impact other rules such as the Unique values or Format match rules. If this rule is not defined on a column then those rules when run on that column will automatically ignore any null values. If this rule is defined on a column then those rules will examine null/empty values on that column and consider them for score purposes.

8.5.5 Format match

Rule details

**Format match**
Confirms that values in a column match a specific format or other criteria.

Description

Columns *

job_type

Validation type *

Enumeration

Validation criteria * ⓘ

Full Time, Part Time, Per Diem

Values will be checked if they are in the list:

Full Time
Part Time
Per Diem

Publish

Back

Cancel

Figure 22 – Example of format match rule


This rule checks if all the values in the column are valid. If the Empty/blank fields rule is not defined on a column then null/empty values will be ignored for the purposes of this rule.

This rule can validate each value in the column using three different approaches:

- **Enumeration** – This is a comma separated list of values. If the value being evaluated can't be compared to one of the listed values then it fails the check. Note that commas can be escaped by using \ and \ can be escaped by using \\. So "a \, b, c" contains two values the first is "a, b" and the second is "c".
- **Like Pattern** - See [Expression functions in the mapping data flow - Azure Data Factory | Microsoft Docs](#) for details
- **Regular Expression** – See [Expression functions in the mapping data flow - Azure Data Factory | Microsoft Docs](#). Note that no quotes are needed around the expression.

8.5.6 Freshness

Rule details

**Freshness**
Confirms that all values are up to date.

Description

Last updated ⓘ
Last modified date

Timeframe *
Custom time frame

Last updated within

Amount

1

Unit

Hours

of data quality scan start time.

Publish

Back

Cancel

Figure 23 - Example of Freshness rule

The purpose of the Freshness rule is to determine if the asset has been updated within the expected time. Microsoft Purview currently supports checking Freshness by looking at last modified dates. But as the existence of the drop down (with only one choice at the moment) hints, Microsoft Purview intends to support other ways of detecting Freshness in the future.

For right now Microsoft Purview Freshness only support files in ADLS Gen2 of type Parquet, so if an asset consists of exactly one file then Microsoft Purview will check its last modified date and use that to calculate Freshness. If the resource is a resource set then Microsoft Purview will look at all the partition files in the resource set and pick the partition with the newest date and check it's last modified date.

The Timeframe drop down supports some pre-defined Freshness checks such as "Last 2 days". This means that the asset's last modified date is expected to be less than or equal to 2 days from the time of the quality scan. Another option is Custom time frame that enables one to specify that the asset was updated less than or equal to a certain amount (e.g. 1, 2, 3) of Units (Hours, Days, Weeks, Month) from the time of the quality scan.

An asset can have zero or one instances of the Freshness rule. Note that the score for the Freshness rule is either 100 (it passed) or 0 (it failed).

8.5.7 Table lookup

Rule details

Table lookup

Confirms that a value in one table can be found in a specific column of another table.

Published

Description

Confirms that a value in one table can be found in a specific column of another table.

Columns *

location

Comparison table

Asset *

citystatezip.snappy.parquet

Column *

abc Data (string)

Publish

Cancel

Figure 24 - Table Lookup

The Table Lookup rule will examine each value in the column the rule is defined on and compare it to a reference table. So in the example above the current table has a column called “location” that contains cities, states and zip codes in the form “city, state zip”. In this example there is another table, called citystatezip.snappy.parquet, which is a Parquet file, that contains all the legal combinations of cities, states and zip codes supported in the United States. The goal is to compare all the locations in the current column against that reference list to make sure that only legal combinations are being used.

To do this we first type in the “citystatezip.snappy.parquet”’s name into the search assets dialog. We then click on the desired asset and we then choose from the drop down the column we want to compare against.

Note that if a remote table contains a column whose type isn’t compatible with the current column then the column’s name will still be shown in the drop down but it will have a clear error indication to explain why it cannot be selected.

8.5.8 Unique values

Rule details

Unique values
Confirms that values in a column are unique.

Description

Columns *
2 selected

- ☒ page_url
- ☒ uniq_id
- ☐ country
- ☐ country_code
- ☐ date_added
- ☐ has_expired
- ☐ job_board
- ☐ job_description
- ☐ job_title

Publish | **Back** | **Cancel**

Figure 25 - Unique Values

The unique values rule states that all the values in this column must be unique. All values that are unique “pass” and those that do not are treated as failing. If the Empty/blank fields rule is not defined on the column then null/empty values will be ignored for the purposes of this rule.

Currently at most two columns can have the unique values rule defined on them.

8.6 View scan report

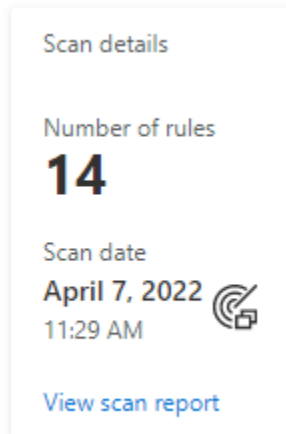


Figure 26 - Example of scan results description

When a scan has completed it will display a card like the above that shows how many rules were run in the last scan, when that scan ran and have a link to “View scan report”.

Data quality scan report

[Error report](#)

Job ID: 95202a8a-939a-42a8-b233-79aa06253a0a

Job status: ✔ Succeeded

Duration: 2m 7s

Rules with errors
One or more errors found. View error report.

View: ☒ Columns only ☐ All rules

Showing 14 items

Name ↑	Detected type	Configuration	Status	Score
> country(1)	abc String	--		100
> country_code(1)	abc String	--		100
> date_added(1)	abc String	--		100
has_expired	abc String	--		--
> job_board(1)	abc String	--		100
> job_description(1)	abc String	--		100
> job_title(1)	abc String	--		100
> job_type(1)	abc String	--		100
> location(2)	abc String	--		83.65
> organization(1)	abc String	--		100
> page_url(1)	abc String	--		100
> salary(1)	abc String	--		100
> sector(1)	abc String	--		100
> uniq_id(1)	abc String	--		100

OK

Figure 27 – Scan report example

The view scan report will give the Job ID, if the scan succeeded or failed and how long it took. If there were any errors they will be listed in the “Rules with errors” section. There is also the ability to see all the rules defined at the time of the scan using either the “Columns only” or the “All rules” views.

There is also a link titled “Error report”. This is only available on scans that succeeded and if clicked will cause a CSV file to be downloaded. That CSV file will contain the first 500 rows that had at least one rule fail on them.

The structure of the CSV file is that the first columns will reproduce all the columns from the source table. Following those columns will be one column per rule that applies to a row (some rules, like Freshness, do not apply to rows).

Rules that apply to a single column will be titled [Column Name] – [Rule Name], so for example, “salary – Data type match”.

Since there can be only one duplicate rows rule for an entire asset that rule is titled “Duplicate rows”, same for Freshness and for Custom rules (which can apply to multiple columns) they will be titled [Custom Rule Name] – Custom.

Each of the rule columns will contain pass, fail, null filtered or custom filtered. Pass/fail indicates that the rule passed or failed on that row. Null filtered means that row was ignored by the rule because nulls were allowed. Custom filtered means that the filter expression for the custom rule ignored that row.

8.7 Draft versus Testing versus Published

Rules in Microsoft Purview Data Quality can be in three states:

- Published – This is the default state. This means that when a data quality scan is run this rule will be executed against the data and its score will be used to calculate the global score.
- Testing – This can be selected from the drop down when saving a rule. It specifies that when a data quality scan is run this rule should be executed. But the rule is in a testing state so its score should not be used in calculating the global score.
- Draft – This can also be selected from the drop down when saving a rule. It specifies that the rule is not ready for execution yet. So, it is saved but it will not be run on data quality scans and hence will have no score.

8.8 Understanding scores

The goal of data quality rules is to provide a description of the state of the data. In particular, to help understand how far away the data is from the ideal state described by the rules. Each rule, when it runs, produces a score that describes how close the data is to its desired state. Most rules are very straight forward in terms of score, they take the total number of passing rows, divide by the total number of rows and that is their score.

However as described in previous sections it is possible for rows to be filtered out. For example, let's imagine there is a column that does not have "Empty/blank fields" rule defined on it. In that case, nulls are allowed. So certain rules, like the unique values rule, will filter out null values in that case. So if the asset has 10,000 rows in a table but 3,000 were null and 500 were not unique then the score would be $((10000 - 3000 - 500) / (10000 - 3000)) * 100 = 93$ (the score is multiplied by 100 to make the scores more readable). So the math ignores the null rows for the purposes of evaluating the data and determining a score.

In the case of custom rules there is a similar capability but in this case the filter is not on nulls but rather the filter expression.

Some rules, like the Freshness rule, are either pass or fail. So their scores will be either 0 or 100.

Microsoft Purview can then give a sense for the state of each column by generating a column score. This score is the average of the scores of the rules on that column.

Microsoft Purview also generates a global score. This is the average of all the production rule defined on the asset.

The global score is intended to be the official definition of the state of the asset.

As described in the previous section, rules that are in testing state will not have their scores contribute to the global score. And rules in a draft state will not be run at all during quality scans and so won't have scores.

8.9 Scheduling quality scans

Assets can be scheduled to have quality scans run on a regular schedule. A quality scan works very much like a Data Map Scan. The first step is to click on “Schedule quality scan” to create a new scan.

The schedule quality scan screen will start with Overview which asks for a mandatory name for the scan and the identification of the data source that is to be scanned. Once the data source is identified then a credential will be asked for, right now the system only supports MSI. The Test connection button can be used to test if Purview can access the asset.

After pressing Continue one is taken to the Scope tab where it’s possible to select specific tables, folders, files, etc. or just select everything. When the quality scan is run Purview will examine all of the folders or schemas or databases that are in scope and see which ones contain files/tables/etc. that have rules defined on them. So even if one selects an entire data lake with millions of files only those with quality rules defined on them will be evaluated.

The next tab is the schedule tab where the quality scan can be run just once or can be run on a regular schedule.

Then comes the review tab to review the configuration and decide if the quality scan configuration should be saved or saved and run.

The Data quality page contains two tabs, a Data quality scans tab and a Monitoring tab.

The Data quality scans tab contains a list of all defined quality scan configurations and one can choose to edit, run or delete them.

The monitoring tab lists all quality scans that have or are being run with options to filter in order to find particular scan runs. Running quality scans can be canceled from this tab. Completed scans will list which assets were scanned for quality purposes.