**Use file metadata in serverless SQL pool queries**

* Article
* 02/18/2022
* 2 minutes to read
* 7 contributors

Feedback

**In this article**

1. [Prerequisites](https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/query-specific-files#prerequisites)
2. [Functions](https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/query-specific-files#functions)
3. [Next steps](https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/query-specific-files#next-steps)

Serverless SQL pool can address multiple files and folders as described in the [Query folders and multiple files](https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/query-folders-multiple-csv-files) article. In this article, you learn how to use metadata information about file and folder names in the queries.

Sometimes, you may need to know which file or folder source correlates to a specific row in the result set.

You can use function filepath and filename to return file names and/or the path in the result set. Or you can use them to filter data based on the file name and/or folder path. These functions are described in the syntax section [filename function](https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/query-data-storage#filename-function) and [filepath function](https://learn.microsoft.com/en-us/azure/synapse-analytics/sql/query-data-storage" \l "filepath-function). In the following sections, you'll find short descriptions along samples.

**Prerequisites**

Your first step is to **create a database** with a datasource that references storage account. Then initialize the objects by executing [setup script](https://github.com/Azure-Samples/Synapse/blob/master/SQL/Samples/LdwSample/SampleDB.sql) on that database. This setup script will create the data sources, database scoped credentials, and external file formats that are used in these samples.

**Functions**

**Filename**

This function returns the file name that row originates from.

The following sample reads the NYC Yellow Taxi data files for the last three months of 2017 and returns the number of rides per file. The OPENROWSET part of the query specifies which files will be read.

SQLCopy

SELECT

nyc.filename() AS [filename]

,COUNT\_BIG(\*) AS [rows]

FROM

OPENROWSET(

BULK 'parquet/taxi/year=2017/month=9/\*.parquet',

DATA\_SOURCE = 'SqlOnDemandDemo',

FORMAT='PARQUET'

) nyc

GROUP BY nyc.filename();

The following example shows how *filename()* can be used in the WHERE clause to filter the files to be read. It accesses the entire folder in the OPENROWSET part of the query and filters files in the WHERE clause.

Your results will be the same as the prior example.

SQLCopy

SELECT

r.filename() AS [filename]

,COUNT\_BIG(\*) AS [rows]

FROM OPENROWSET(

BULK 'csv/taxi/yellow\_tripdata\_2017-\*.csv',

DATA\_SOURCE = 'SqlOnDemandDemo',

FORMAT = 'CSV',

PARSER\_VERSION = '2.0',

FIRSTROW = 2)

WITH (C1 varchar(200) ) AS [r]

WHERE

r.filename() IN ('yellow\_tripdata\_2017-10.csv', 'yellow\_tripdata\_2017-11.csv', 'yellow\_tripdata\_2017-12.csv')

GROUP BY

r.filename()

ORDER BY

[filename];

**Filepath**

The filepath function returns a full or partial path:

* When called without a parameter, it returns the full file path that the row originates from. When DATA\_SOURCE is used in OPENROWSET, it returns path relative to DATA\_SOURCE.
* When called with a parameter, it returns part of the path that matches the wildcard on the position specified in the parameter. For example, parameter value 1 would return part of the path that matches the first wildcard.

The following sample reads NYC Yellow Taxi data files for the last three months of 2017. It returns the number of rides per file path. The OPENROWSET part of the query specifies which files will be read.

SQLCopy

SELECT

r.filepath() AS filepath

,COUNT\_BIG(\*) AS [rows]

FROM OPENROWSET(

BULK 'csv/taxi/yellow\_tripdata\_2017-1\*.csv',

DATA\_SOURCE = 'SqlOnDemandDemo',

FORMAT = 'CSV',

PARSER\_VERSION = '2.0',

FIRSTROW = 2

)

WITH (

vendor\_id INT

) AS [r]

GROUP BY

r.filepath()

ORDER BY

filepath;

The following example shows how *filepath()* can be used in the WHERE clause to filter the files to be read.

You can use the wildcards in the OPENROWSET part of the query and filter the files in the WHERE clause. Your results will be the same as the prior example.

SQLCopy

SELECT

r.filepath() AS filepath

,r.filepath(1) AS [year]

,r.filepath(2) AS [month]

,COUNT\_BIG(\*) AS [rows]

FROM OPENROWSET(

BULK 'csv/taxi/yellow\_tripdata\_\*-\*.csv',

DATA\_SOURCE = 'SqlOnDemandDemo',

FORMAT = 'CSV',

PARSER\_VERSION = '2.0',

FIRSTROW = 2

)

WITH (

vendor\_id INT

) AS [r]

WHERE

r.filepath(1) IN ('2017')

AND r.filepath(2) IN ('10', '11', '12')

GROUP BY

r.filepath()

,r.filepath(1)

,r.filepath(2)

ORDER BY

filepath;