Dp 203 prep

Sunday, October 1, 2023

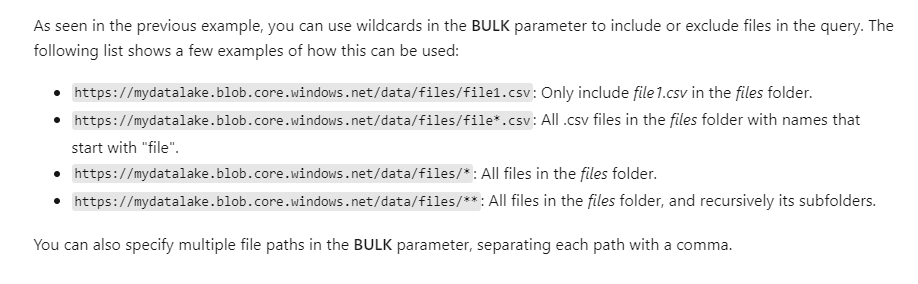
11:58 PM

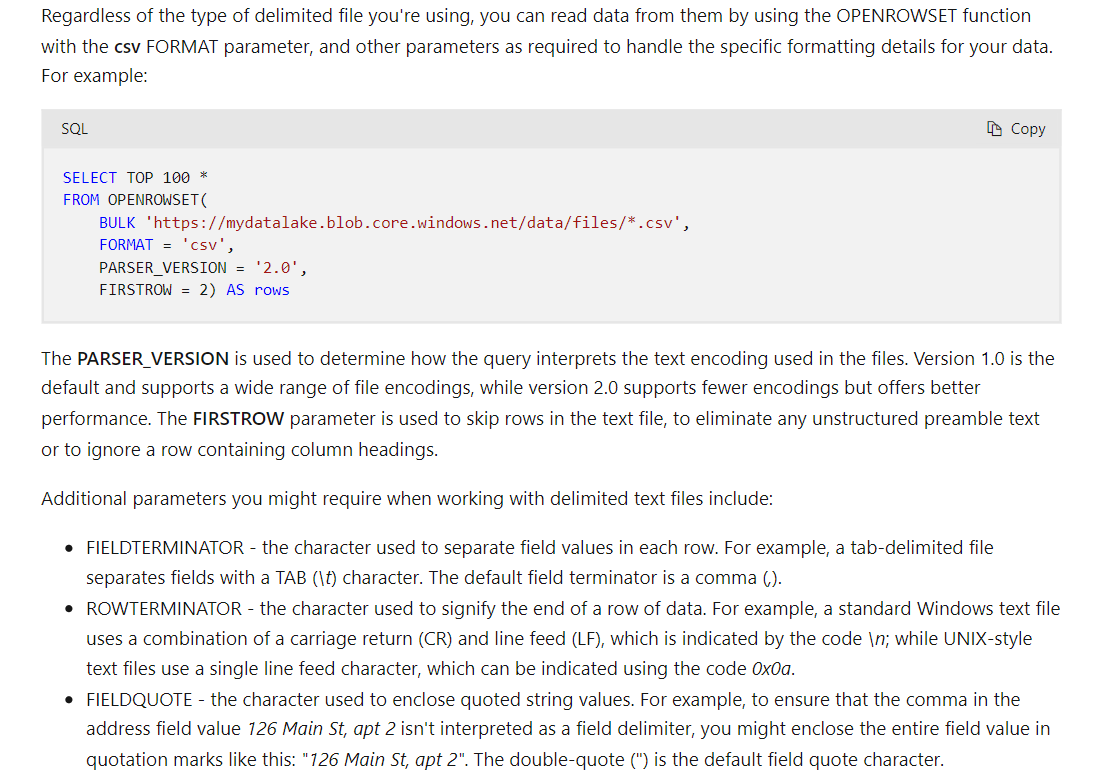
Introduction to Azure data lake

Big data processing and analytics 
gig data scenarios usually refer to analytical workloads that involve massive volumes of data in a of formats that 
needs to be processed at a fast velocity - the so-called fthree Azure Data Lake Storage Gen 2 provides a scalable and 
secure distributed data state on which big data services such as Azure Synapse Analytics, Azure Databricks, and Azure 
HDlnsight can apply data processing frameworks as Apache Spark Hive. and Hadoop. The distributed nature of the 
Storage and the enables tasks to be performed in parallel, resulting in high•performatke and 
scalability even when huge amounts of data, 

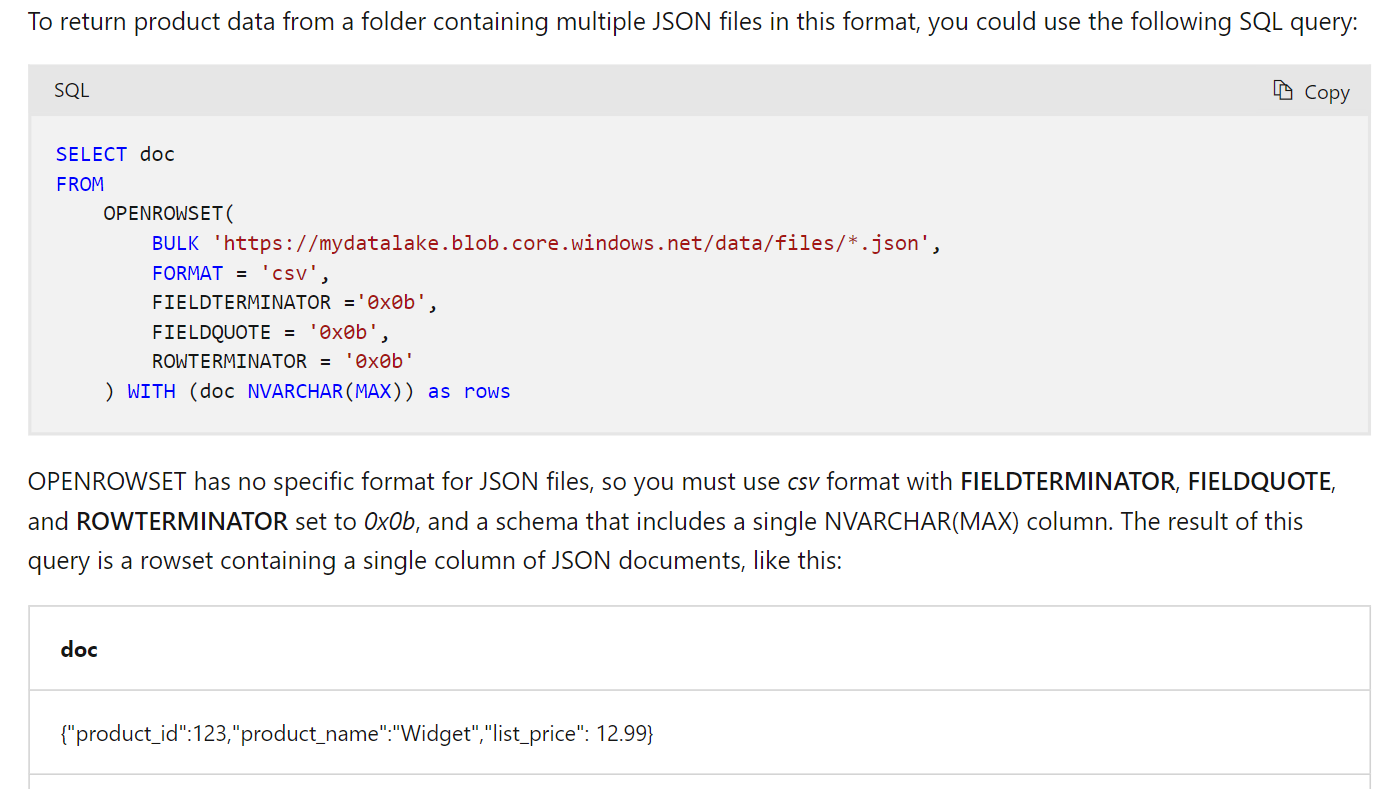
Querying and manipulating data with SQL 
Structured Query Language (SQL) is a ubiquitous language for querying and manipulating data, and is the foundation for 
relational databases, including the popular Microsoft SQL Server database platform. Azure Synapse Analytics supports 
SQL-based data querying and manipulation through two kinds of SQL pool that are based on the SQL Server relational 
database engine: 
A built-in serverless pool that is optimized for using relational SQL semantics to query file-based data in a data lake. 
• Custom dedicated SQL pools that host relational data warehouses. 
The Azure Synapse SQL system uses a distributed query processing model to parallelize SQL operations, resulting in a 
highly scalable solution for relational data processing. You can use the built-in serverless pool for cost-effective analysis 
and processing of file data in the data lake, and use dedicated SQL pools to create relational data warehouses for 
enterprise data modeling and reporting. 

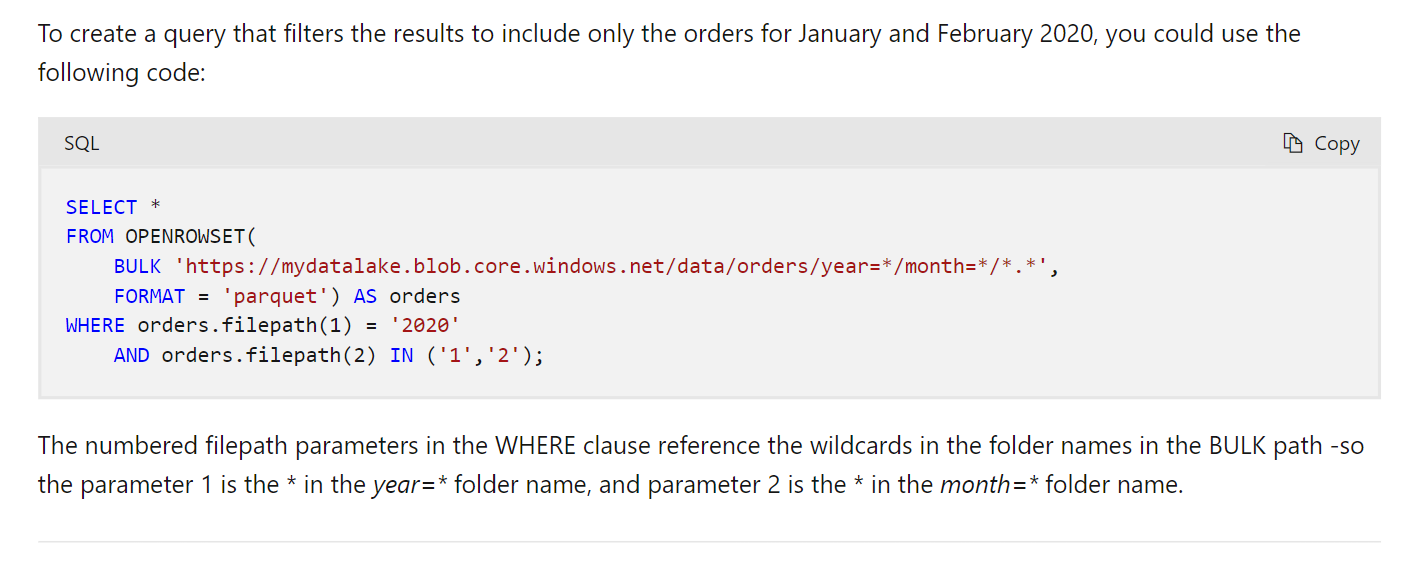
Query files using a serverless SQL pool 
10 minutæ 
You Can use a SQL pool to query data files in various common file formats. including: 
Delimited text, Such as comma-separated values (CSV) files. 
• JavaScript object notation (JSON) files. 
Parquet files. 
The basic Syntax for querying is the same for all Of these types Of file, and is built on the OPENRCAVSET SQL function; 
which generates a tabular rowset from data in one or more files. For example. the following query could be used to 
entract data from CSV files 
SELECT TOP lee • 
FRO" OPENROhSET( 
BULK 
The OPENROWSET function includes more parameters that determine factors such as: 
The schema of the resulting rowset 
Additional formatting options for delimited text 
Copy 











Syntax to create external objects - <https://learn.microsoft.com/en-gb/training/modules/query-data-lake-using-azure-synapse-serverless-sql-pools/4-external-objects>

Yes, you are correct. In many cases, a single (star) is used as a wildcard to represent all 
files in a directory, just like The difference lies in how the wildcards are interpreted by the 
specific system or tool being used. 
(single star): Represents all files in the current directory 
Represents all files in the current directory and its subdirectories. 
Both and 9k*' are commonly used as wildcards in various systems to represent multiple 
files and the choice between them depends on the specific context and the system being 
used. In some systems both wildcards might behave equivalently, while in others, there 
might be a distinction in their behavior. It's always a good practice to refer to the system's 
documentation or specifications to understand the exact behavior of these wildcards in a 
given context. 

J Azure Synapse serverless SQL pools / Create a lake database in Azure Synapse Analytics / 
Unit 1 of 8 v 
Introduction 
1 minute 
Next > 
100 XP 
Data analysts and engineers often find themselves forced to choose between the flexibility of storing data files in a data 
lake, with the advantages Of a structured schema in a relational database. Lake databases in Azure Synapse Analytics 
provide a way to combine these two approaches and benefit from an explicit relational schema of tables, views, and 
relationships that is decoupled from file-based storage. 

**DOUBT**

<https://www.google.com/search?q=row+level+security+in+databricks&rlz=1C1GCEU_en-GBIN1000IN1000&oq=row+level+scurity&gs_lcrp=EgZjaHJvbWUqCQgIEAAYDRiABDIGCAAQRRg5MgkIARAAGA0YgAQyCQgCEAAYDRiABDIJCAMQABgNGIAEMgkIBBAAGA0YgAQyCQgFEAAYDRiABDIJCAYQABgNGIAEMgkIBxAAGA0YgAQyCQgIEAAYDRiABDIJCAkQABgNGIAE0gEIOTQ5OWowajeoAgCwAgA&sourceid=chrome&ie=UTF-8#ip=1> - DIFFICULT to DIGEST

Manage user permissions in Azure 
Synapse serverless SQL pools 
3 minutes 
100 XP 
TO secure data, Azure Storage implements an access control model that supports both Azure role-based access control 
(Azure RBAC) and access control lists (ACLs) like Portable Operating System Interface for Unix (POSIX) 
You can associate a security principal with an access level for files and directories. These associations are captured in an 
access control list (ACL). Each file and directory in your storage account has an access control list. When a security 
principal attempts an operation on a file or directory, an ACL check determines whether that security principal (user, 
group, service principal, or managed identity) has the correct permission level to perform the operation. 

Making conditional updates 
While you can make data modifications in a dataframe and then replace a Delta Lake table by overwriting it, a more 
common pattern in a database is to insert, update or delete rows in an existing table as discrete transactional operations. 
To make such modifications to a Delta Lake table, you can use the DeltaTable object in the Delta Lake API, which supports 
update, delete, and merge operations. For example, you could use the following code to update the price column for all 
rows with a category column value of "Accessories": 
Python 
from delta . tables import * 
from pyspark . sq1. functions import * 
# Create a deltaTab1e object 
deltaTabIe = DeltaTabIe.forpath(spark, 
# Update the table (reduce price of accessories by 
deltaTab1e . update( 
condition = "Category 'Accessories ' " , 
set = { "Price": "price * 0.9" 
[b Copy 

Dimension tables 
Dimension tables describe business entities, such as products, people, places, and dates. Dimension tables contain 
columns for attributes of an entity. For example, a customer entity might have a first name, a last name, an email address, 
and a postal address (which might consist of a street address, a city, a postal code, and a country or region). In addition to 
attribute columns, a dimension table contains a unique key column that uniquely identifies each row in the table. In fact, 
it's common for a dimension table to include two key columns: 
• a surrogate key that is specific to the data warehouse and uniquely identifies each row in the dimension table in the 
data warehouse - usually an incrementing integer number. 
• An alternate key, often a natural or business key that is used to identify a specific instance of an entity in the 
transactional source system from which the entity record originated - such as a product code or a customer ID. 

**DOUBT**

Data integrity constraints 
Dedicated SQL pools in Synapse Analytics don't support foreign key and unique constraints as found in other relational 
database systems like SQL Server. This means that jobs used to load data must maintain uniqueness and referential 
integrity for keys, without relying on the table definitions in the database to do so. 
Q Tip 
For more information about constraints in Azure Synapse Analytics dedicated SQL pools, see Primary key, foreign 
key, and unique key using dedicated SQL pool in Azure Synapse Analytics. 

**DOUBT**

Creating a dedicated SQL pool 
To create a relational data warehouse in Azure Synapse Analytics, you must create a dedicated SQL Pool. The simplest way 
to do this in an existing Azure Synapse Analytics workspace is to use the Manage page in Azure Synapse Studio, as shown 
here: 

Distribution 
Azure Synapse Analytics dedicated SQL pools use a massively parallel processing (MPP) architecture, as opposed to the 
symmetric multiprocessing (SMP) architecture used in most OLTP database systems. In an MPP system, the data in a table 
is distributed for processing across a pool of nodes. Synapse Analytics supports the following kinds of distribution: 
• 
• 
Hash: A deterministic hash value is calculated for the specified column and used to assign the row to a compute 
node. 
Round-robin: Rows are distributed evenly across all compute nodes. 
Replicated: A copy of the table is stored on each compute node. 

The table type often determines which option to choose for distributing the table. 
Table 
type 
Dimension 
Fact 
Staging 
Recommended distribution option 
Use replicated distribution for smaller tables to avoid data shuffling when joining to distributed fact tables. If tables 
are too large to store on each compute node, use hash distribution. 
Use hash distribution with clustered columnstore index to distribute fact tables across compute nodes. 
Use round-robin distribution for staging tables to evenly distribute data across compute nodes. 

Creating dimension tables 
When you create a dimension table, ensure that the table definition includes surrogate and altemate keys as well as 
columns for the attributes of the dimension that you want to use to group aggregations. It's often easiest to use an 
IDENTITY column to auto-generate an incrementing surrogate key (otherwise you need to generate unique keys every 
time you load data). The following example shows a CREATE TABLE statement for a hypothetical DimCustomer dimension 
table. 
Copy 
CREATE TABLE 
INT IDENTITY NOT NULL, 
customerA1ternateKey Nutt, 
NVARCHAR(8a) NOT NULL, 
EmailAddress 
uvARCHAR(sa) NULL, 
phone NULL, 
StreetAddress 
uvARCHAR(1øe) , 
city MVARCHAR(2ø), 
postalcode WARCHAR(1ø), 
NVARCHAR(20) 
DISTRIBUTION = REPLICATE, 
CLUSTERED COLumusTORE 

In most cases, you should implement a data warehouse load process that performs tasks in the following order: 
1. Ingest the new data to be loaded into a data lake, applying pre-load cleansing or transformations as required. 
2. Load the data from files into staging tables in the relational data warehouse. 
3. Load the dimension tables from the dimension data in the staging tables, updating existing rows or inserting new 
rows and generating surrogate key values as necessary. 
4. Load the fact tables from the fact data in the staging tables, looking up the appropriate surrogate keys for related 
dimensions. 
5. Perform post-load optimization by updating indexes and table distribution statistics. 
After using the COPY statement to load data into staging tables, you can use a combination of INSERT , UPDATE, MERGE, and 
CREATE TABLE AS SELECT (C TAS) statements to load the staged data into dimension and fact tables. 

Copy 
CREATE TABLE dba. Dimproduct 
WITH 
DISTRIBUTION = REPLICATE, 
CLUSTERED COLUMNSTORE INDEX 
SELECT OVER(ORDER BY ProdiD) AS ProdKey, 
ProdID as ProdAItKey, 
Produ ctName, 
Produ ctCategory , 
Color, 
Size, 
Listprice, 
Disconti nued 
FROM dba. Stageproduct; 
O Note 
You cant use IDENTITY to generate a unique integer value for the surrogate key when using a CTAS statement, so 
this example uses the ROW_NUMBER function to generate an incrementing row number for each row in the results 
ordered by the ProductlD business key in the staged data. 

<https://learn.microsoft.com/en-gb/training/modules/load-optimize-data-into-relational-data-warehouse/3-load-dimension-tables> - CAN be asked once.

Manage workloads in Azure Synapse 
Analytics 
10 minutes 
100 XP 
Azure Synapse Analytics allows you to create, control and manage resource availability when workloads are competing. 
This allows you to manage the relative importance of each workload when waiting for available resources. 
To facilitate faster load times, you can create a workload classifier for the load user with the "importance" set to 
above_normal or High. Workload importance ensures that the load takes precedence over other waiting tasks of a lower 
importance rating. IJse this in conjunction with your own workload group definitions for workload isolation to manage 
minimum and maximum resource allocations during peak and quiet periods. 
Dedicated SQL pool workload management in Azure Synapse consists of three high-level concepts: 
Workload Classification 
Workload Importance 
Workload Isolation 
These capabilities give you more control over how your workload utilizes system resources. 

How Azure Synapse Analytics works with Azure Advis 
Azure Advisor recommendations are free, and the recommendations are based on telemetry data that is generated 
Azure Synapse Analytics. The telemetry data that is captured by Azure Synapse Analytics include 
Data Skew and replicated table information. 
Column statistics data 
• TempDB utilization data. 
Adaptive Cache. 
Azure Synapse 
Analytics 
Azure Advisor Recommendation 
Replicate 
tables 
Recommendation 
Telemetry 
Tempdb 
Adaptive 
Cache 

I don’t get this since ever: (security types/methods)

<https://learn.microsoft.com/en-us/training/modules/secure-data-warehouse-azure-synapse-analytics/2-understand-network-security-options>

Using PolyBase for SQL Data Warehouse loads 
SQL Data Warehouse supports many loading methods, including non-PolyBase options (BCP and SQLBulkCopy API), and 
PolyBase options CTAS/INSERT, PolyBase with SSIS, Azure Data Factory (ADF), and third party tools including Azure 
Databricks, Attunity eCloudbeamE1, Striim c? , Informatica El, and TalendrZ . 
PolyBase is by far the fastest and most scalable SQL Data Warehouse loading method to date, so we recommend it as 
your default loading mechanism. PolyBase is a scalable, query processing framework compatible with Transact-SQL that 
can be used to combine and bridge data across relational database management systems, Azure Blob Storage, Azure 
Data Lake Store and Hadoop database platform ecosystems (APS only). 
Note As a general rule, we recommend making PolyBase your first choice for loading data into SQL Data Warehouse 
unless you can't accommodate PolyBase-supported file formats. Currently PolyBase can load data from UTF-8 and UTF-16 
encoded delimited text files as well as the popular Hadoop file formats RC File, ORC, and Parquet (non-nested format). 
PolyBase can load data from gzip, zlib and Snappy compressed files. PolyBase currently does not support extended ASCII, 
fixed-file format, WinZip and semi-structured data such as Parquet (nested/hierarchical), JSON, and XML. A popular 
pattern to load semi-structured data is to use Azure Databricks or similarly HDl/Spark to load the data, flatten/transform 
to the supported format, then load into SQL DVM. 

Copying data into storage 
The general load process begins with migrating your data into Azure Blob Storage. Depending on your network's 
capabilities, reliability, and utilization, you can use AZCOPYe• to upload your source data files to Azure Storage Blobs with 
an upload rate from 80 MB/second to 120 MB/second. 
Then, in SQL Data Warehouse, you configure your credentials that will be used to access Azure Blob Storage: 
CREATE DATABASE SCOPED CREDENTIAL myid_credential WITH IDENTITY = imyid', 
Next you define the external Azure Blob Storage data source with the previously created credential: 
CREATE EXTE 
RCE data_ltb WITH (TYPE = HADOOP, LOCATION 
CREDENTIAL; m 
And for the source data, define the file format and external table definition: 
CREATE EXTERNAL FILE FORMAT pipedelimited 
WITH (FORMAT_TYPE = DELIMITEDTEXT, 

**Visit this link for above images :** [**https://learn.microsoft.com/en-us/archive/blogs/sqlcat/azure-sql-data-warehouse-loading-patterns-and-strategies**](https://learn.microsoft.com/en-us/archive/blogs/sqlcat/azure-sql-data-warehouse-loading-patterns-and-strategies)

**azure sql server vs synapse dedicated sql pool -** [**https://learn.microsoft.com/en-us/answers/questions/976202/azure-sql-server-vs-synapse-dedicated-sql-pool**](https://learn.microsoft.com/en-us/answers/questions/976202/azure-sql-server-vs-synapse-dedicated-sql-pool)

**Ingesting data into Azure Synapse Analytics using PolyBase -** [**https://www.sqlshack.com/ingesting-data-into-azure-synapse-analytics-using-polybase/**](https://www.sqlshack.com/ingesting-data-into-azure-synapse-analytics-using-polybase/)

**LEARN : Partition Switching, RANGE RIGHT FOR VALUES, PARTITION SCHEME,**

**LRS,GRS,ZRS,RA-GRS,RA-ZRS -** [**https://cloudbuild.co.uk/tag/lrs-vs-grs-vs-ragrs-vs-zrs-vs-gzrs-vs-ragzrs/**](https://cloudbuild.co.uk/tag/lrs-vs-grs-vs-ragrs-vs-zrs-vs-gzrs-vs-ragzrs/)

In the following image example, if , a PolyBase query will return rows from mydata.txt and 
mydata2.txt . It won't return mydata3.txt because it's in a subfolder of a hidden folder. And it won't return _hidden .txt 
because it's a hidden file. 
mydata.bt 
mydata2 at 
*uddentolder,' 
ata3,Ot 
Unlike Hadoop external tables, native external tables don't return subfolders unless you specify at the end of path. In 
this example, if LOCATION- /webdata/• , a serverless SQL pool query, will return rows from mydata.txt. It won't return 
mydata2.txt and mydata3_txt because they're located in a subfolder. Hadoop tables will return all files within any sub- 
folder. 
Both Hadoop and native external tables will skip the files with the names that begin with an underline C) or a period (.). 

HOTSPOT - 
You are building an Azure Stream Analytics job to identify how much time a user spends interacting with a feature on a webpage. 
The job receives events based on user actions on the webpage. Each row of data represents an event. Each event has a type of either 'start' or 'end. 
You need to calculate the duration between start and end events. 
How should you complete the query? TO answer, select the appropriate options in the answer area. 
NOTE: Each correct selection is worth one point. 
Hot Area: 
Answer Area 
SELECT 
feature, 
DATEADD ( 
DATEDIFF ( 
DATE PART ( 
e con d. 
OVER BY 
ISFIRST 
LAST 
TopoNE 
Time) du r at i On 
FRON input TIMESTAMP BY Time 
*HERE 
E V ent 
feature LIMIT 

**NICE QUERY**