|  |  |
| --- | --- |
|  |  |
|  | Lab: SQL Data Warehouse – Introduction, Load, and Enrich |

Contents

Contents

[Introduction 3](#_Toc506839769)

[Take-away 3](#_Toc506839770)

[Prerequisites 3](#_Toc506839771)

[Class 4](#_Toc506839772)

[Section 1: Provision SQL Data Warehouse 4](#_Toc506839773)

[Section 2: Connect to SQL Data Warehouse 10](#_Toc506839774)

[Section 3: Create user and assign resource class 13](#_Toc506839775)

[Section 4: Load data from Azure Blob Storage into SQL Data Warehouse using PolyBase 13](#_Toc506839776)

[Section 5: Manage SQL DW - Scale/Pause/Resume 13](#_Toc506839777)

[Section 6: Enrich Analysis 13](#_Toc506839778)

[Section 7: Publish Data (Optional) 13](#_Toc506839779)

[Terms of use 13](#_Toc506839780)

|  |
| --- |
| Introduction |

This class introduces students to Azure SQL Data Warehouse. It helps student to understand the value proposition of SQL Data Warehouse over other data warehouse technologies. They should understand how to create SQL Data Warehouse, load data into it, how external tables function, how to export data, etc.

|  |
| --- |
| Take-away |

1. Provision SQL Data Warehouse.
2. Connect to SQL Data Warehouse.
3. Create User and Assign Resource Class.
4. Extract data from a *Data Lake* (Azure Storage Blob) and load into SQL Data Warehouse using PolyBase.
5. Scale SQL Data Warehouse.
6. Export Data from SQL Data Warehouse into Azure Storage Blob.

|  |
| --- |
| Prerequisites |

1. An Azure subscription. See [Get Azure free trial](https://azure.microsoft.com/en-gb/documentation/videos/get-azure-free-trial-for-testing-hadoop-in-hdinsight/).
2. Supported operating system to perform this class: Windows 10 / Windows Server 2016
3. Download and install [Visual Studio](https://www.visualstudio.com/downloads/).

|  |
| --- |
| Class |

## Section 1: Provision SQL Data Warehouse

|  |  |  |
| --- | --- | --- |
| Section | Steps | Screen Shot |
| 1. Access Azure Preview Portal | 1. Sign in to the [**Azure preview portal**](https://ms.portal.azure.com/). |  |
| 1. Create a new logical SQL Server in the Azure Portal | 1. Click **New,** search Logical Server and then hit Enter. (Fig. 1.1) 2. Click Create to open the new SQL Server blade. (Fig. 1.2) | **Fig. 1.1**    **Fig. 1.2** |

|  |  |  |
| --- | --- | --- |
| 1. Provide SQL Server Details. | 1. New SQL Server blade will get opened. (Fig. 1.3) 2. Enter desired name to server in **Server Name** input box. (Fig. 1.3)   A green check mark appears beside the cluster name if it is available.   1. Enter appropriate username in **Server admin login** input box. (Fig. 1.4) 2. Enter appropriate password for **Server admin login** in **Password** input box. (Fig. 1.5) 3. Confirm the entered password by entering it again in **Confirm Password** input box. (Fig. 1.5)   *\*Note: Note Username and Password of SQL server which will be used in further steps to connect to SQL Server.* | **Fig. 1.3**    **Fig. 1.4**    **Fig. 1.5** |
|  | 1. Selectappropriate subscription in **Subscription** choice box, which you want to use for creating all Azure Resources for this lab. (Fig. 1.6) 2. In the **Resource group** text box, Select **Create new** and then, in the **Resource group** text box, provide a valid name for the new resource group. A green check mark indicates that you have provided a valid name. (Fig. 1.7) 3. In the **Location** text box, select desired location for data center – such as “West US”. (Fig. 1.8) 4. Click **Create** button present at the bottom of blade. This step may take several minutes to complete. | **Fig. 1.6**    **Fig. 1.7**    **Fig. 1.8** |
| 1. Create Server-level firewall rule in the Azure Portal. | 1. On the SQL Server blade, under **Settings**, click **Firewall** to open the Firewall blade for the SQL Server. (Fig. 1.9) 2. Review the client IP address displayed and validate that this is your IP address on the Internet using a browser of your choice (ask "what is my IP address). Occasionally they do not match for a various reason. 3. Assuming that the IP addresses match, click **Add client IP** on the toolbar. (Fig. 1.11). If these IP addresses do not match, add the IP address that SSMS/Visual Studio will provide you when you attempt to connect in later steps. 4. Click **Save** on the toolbar to save this server-level firewall rule and then click **OK**. (Fig. 1.12)   This step allows your IP address to access SQL server and all databases created under this SQL Server. | **Fig. 1.9**    **Fig. 1.11**    **Fig. 1.12** |
| 1. Create a SQL Data Warehouse. | 1. Click **New** > **Databases** > **SQL Data Warehouse**. (Fig. 1.13) 2. Provide following values for properties   **Database Name:** Enter appropriate name for SQL Data Warehouse. (Fig. 1.14)  **Subscription:** Select same subscription you used for creating SQL Server in earlier steps. (Fig. 1.15)  **Resource Group:** Check Use existing radio button and select Resource Group you created while provisioning SQL Server. (Fig. 1.16)  **Select Source:** Select **Blank Database.** (Fig. 1.17)  **Server:** Click this field to open SQL servers list provisioned in your selected subscription and select SQL Servercreated in earlier steps. (Fig. 1.18)  **Performance Level:** Set performance level to 200 DWU. (Fig. 1.20).  **Collation:** Keep default value **SQL\_Latin1\_General\_CP1\_CI\_AS**. (Fig. 1.19)   1. Press **Create** button at the bottom of blade to provision new SQL Data Warehouse. This process should take a few minutes. | **Fig. 1.13**    **Fig. 1.14**    **Fig. 1.15**    **Fig. 1.16**    **Fig. 1.17** |
|  |  | **Fig. 1.18**    **Fig. 1.19**    **Fig. 1.20** |

## Section 2: Connect to SQL Data Warehouse

This section guides how to make connection to new SQL Data Warehouse using **server admin login** in Visual Studio 2015.

Use following connection properties to make connection:

**SQL Server:** SQL server created for this lab

**Authentication:** SQL Server Authentication

**Username**: Server Admin

**Database:** SQL Server Data Warehouse created for this lab

|  |  |  |
| --- | --- | --- |
| Section | Steps | Screen Shot |
| 1. Get connection information and Authentication using Visual Studio | 1. To connect to your data warehouse, you need to connect through the logical SQL server you created earlier. 2. Visit Azure Portal and Select your data warehouse from Resource Group created for this lab. 3. Find the full name of the logical SQL server. (Fig. 2.21) 4. Open **Visual Studio 2013 or 2015 or SSMS**. 5. Open SQL Server Object Explorer. To do this, select **View** > **SQL Server Object Explorer**. (Fig. 2.22) 6. Click the **Add SQL Server** icon. (Fig. 2.22) 7. Fill in the fields in the Connect to Server window. (Fig. 2.23) 8. Enter the **server name** previously identified in step 3. (Fig. 2.24) 9. Select **SQL Server Authentication.** (Fig. 2.25) 10. Enter user name and password provided while provisioning SQL Server in earlier steps of this lab. In step 3 of section 3. (Fig. 2.26) 11. Select the SQL Data Warehouse created under the SQL Server for this lab. (Fig. 2.27) 12. Click **Connect** at the bottom of window to connect to SQL Data Warehouse. (Fig. 2.27)   Note – in this step, you may find that your IP address is not whitelisted. Please add the IP provided by Visual Studio/SSMS to your firewall settings in Section 1. | **Fig. 2.21**    **Fig. 2.22**    **Fig. 2.23**    **Fig. 2.24**    **Fig. 2.25**    **Fig. 2.26**    **Fig. 2.27** |
| 1. Verify Connected SQL Data warehouse | 1. After successful connection to SQL Data Warehouse new connection node will be get added to SQL Server list of SQL Server Object Explorer. (Fig. 2.28) 2. Expand newly added connection node and find your SQL Data Warehouse inside **Databases** node. | **Fig. 2.28**    **Fig. 2.29** |

## Section 3: Create user and assign resource class

To add SQL Server Authenticated user to SQL Data warehouse we need to do following things:

1. Create new server login
2. Add user to master database using this new server login
3. Add user to SQL Data Warehouse using this new server login

|  |  |  |
| --- | --- | --- |
| Section | Steps | Screen Shot |
| 1. Add user in master database. | 1. Connect to **master** database, present under SQL Server created in earlier steps of this lab by following steps mentioned in Section 2: Connect to SQL Data Warehouse.   Use following properties while making connection to master database if not already connected: (Fig. 3.1)  **SQL Server:** SQL server created for this lab  **Authentication:** SQL Server Authentication  **Username**: Server Admin  **Database:** master   1. Right click on newly added SQL Server connection node and select New Query. (Fig. 3.2) 2. Check if **master** database selected for query window. (Fig. 3.3) 3. Create a new server login by **executing** below statement in query window. (Fig. 3.4)  |  | | --- | | CREATE LOGIN LabUserLogin WITH PASSWORD = 'Str0ng\_password'; |  1. Create user in **master** database using the above login by **executing** this query. (Fig. 3.5)  |  | | --- | | CREATE USER LabUser FOR LOGIN LabUserLogin; | | **Fig. 3.1**    **Fig. 3.2**    **Fig. 3.3**    **Fig. 3.4**    **Fig. 3.5** |
| 1. Add user to SQL Data Warehouse | 1. Connect to **SQL Data Warehouse**, created in earlier steps of this lab by following steps mentioned in Section 2: Connect to SQL Data Warehouse.   Use following properties while making connection to master database: (Fig. 3.6)  **SQL Server:** SQL server created for this lab  **Authentication:** SQL Server Authentication  **Username:** Server Admin  **Database:** SQL Data Warehouse created for  this lab   1. Right click on newly added SQL Server connection node and select New Query. (Fig. 3.2) 2. Check if the **SQL Data Warehouse** is selected in the query window. (Fig. 3.7) 3. To create user in **SQL Data Warehouse** using login created in earlier steps, execute below statement.  |  | | --- | | CREATE USER LabUser FOR LOGIN LabUserLogin; |  1. Execute following statement to allow LabUser to read data from SQL Data Warehouse  |  | | --- | | GRANT CONTROL ON DATABASE::[<database name>] TO LabUser | | **Fig. 3.6**    **Fig. 3.7** |
| 1. Assign Resource Class. | Resource Class manages resource allocation for query execution in SQL Data Warehouse. Resource Class helps to control memory allocation, CPU cycles and Concurrency Slots given to query.  Click [here](https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-develop-concurrency) to learn more about resource class.  To assign resource class to user added in SQL Data Warehouse, execute below statements in same query window used for previous step.   |  | | --- | | EXEC sp\_addrolemember 'staticrc20', 'LabUser'; |   Above statement assigns ‘**staticrc20’** resource class to the newly added user. This resource class grants the user’s queries 200MB per distribution, or 12GB system-wide. | **Fig. 3.8** |

## Section 4: Load data from Azure Blob Storage into SQL Data Warehouse using PolyBase

PolyBase is a technology that accesses data outside of the database via the **t-sql** language. In Azure SQL Data Warehouse, you can import/export/query data from Azure Blob Storage and Azure Data Lake Store. Follow steps in this section to learn load data from Azure Blob Storage into SQL Data Warehouse using Polybase.

|  |  |  |
| --- | --- | --- |
| Section | Steps | Screen Shot |
| 1. Make connection to SQL Data Warehouse | 1. Connect to SQL Data Warehouse using Visual Studio by following steps mentioned in **Section 2: Connect to SQL Data Warehouse**.   Use following properties to make connection: (Fig. 11.1.1)  **SQL Server:** SQL server created for this lab  **Authentication:** SQL Server Authentication  **Username**: LabUserLogin  **Password:** Str0ng\_password  **Database:** SQL Data Warehouse created for this lab (Fig. 4.1.1)   1. Right click on newly added connection node in **SQL Server** list of **SQL Server Object Explorer** and select **New Query** menu. (Fig. 4.1.2) 2. Ensure **SQL Data Warehouse** created for this lab is selected for this query window. (Fig. 4.1.3) | **Fig. 4.1.1**    **Fig. 4.1.2**    **Fig. 4.1.3** |
| 1. Create external data source | 1. Execute following statements in query window to create external data source which defines connection details to the **nyc** container of a public Azure Storage account.  |  | | --- | | CREATE EXTERNAL DATA SOURCE LabNycStorage  WITH  (  TYPE = Hadoop  , LOCATION = 'wasbs://nyc@saoa.blob.core.windows.net/'  );  GO |  * TYPE = [ HADOOP | SHARD\_MAP\_MANAGER | RDBMS | BLOB\_STORAGE]   Specifies the data source type. Use HADOOP when the external data source is Hadoop or Azure Storage blob for Hadoop.   * LOCATION = <location\_path>   For Azure blob storage with Hadoop, specifies the URI for connecting to Azure blob storage.   * LOCATION = 'wasb[s]://container@account\_name.blob.core.windows.net' * Note that when using secured storage (best practice), you will have to create a scoped credential and provide a key for your storage account. | **Fig. 4.2.1** |
| 1. Define external file format | 1. Definition of external file format helps Data Warehouse to understand format of external file to be loaded. It defines field terminator, string delimiter and date field format. These properties helps to capture fields inside file. 2. Clear query window and execute following statements in it to define external file format.  |  | | --- | | CREATE EXTERNAL FILE FORMAT compressedcsv  WITH  ( FORMAT\_TYPE = DELIMITEDTEXT  , FORMAT\_OPTIONS ( FIELD\_TERMINATOR = '|'  , STRING\_DELIMITER = ''  , DATE\_FORMAT = ''  )  , DATA\_COMPRESSION = 'org.apache.hadoop.io.compress.GzipCodec'  );  GO |  * **FIELD\_TERMINATOR** = field\_terminator   Applies only to delimited text files. This specifies one or more characters that mark the end of each field (column) in the text-delimited file. External file used for this lab has comma (,) as text delimiter   * **STRING\_DELIMITER** = string\_delimiter   Specifies the field terminator for data of type string in the text-delimited file.   * **DATE\_FORMAT** = datetime\_format   Specifies a custom format for all date and time data that might appear in a delimited text file.  When the DATE\_FORMAT is empty or an empty string, PolyBase will use several default formats.   * **DATA\_COMPRESSION** = data\_compression\_method   Specifies the method with which data was compressed. If this value is not specified, PolyBase will assume the data is uncompressed. This option is necessary for loading compressed data formats. | **Fig. 4.3.1** |
| 1. Create schema for external tables | 1. Clear query window and execute following statements to create new schema for external table.  |  | | --- | | CREATE SCHEMA [asb];  GO | | **Fig. 4.4.1** |
| 1. Create external table | External tables refers to data from external data source. Data is not stored within SQL Data Warehouse.  Clear query window and execute following statements to define external table.   |  | | --- | | CREATE EXTERNAL TABLE [asb].[Trip]  (  [DateID] [int] NOT NULL,  [MedallionID] [int] NOT NULL,  [HackneyLicenseID] [int] NOT NULL,  [PickupTimeID] [int] NOT NULL,  [DropoffTimeID] [int] NOT NULL,  [PickupGeographyID] [int] NULL,  [DropoffGeographyID] [int] NULL,  [PickupLatitude] [float] NULL,  [PickupLongitude] [float] NULL,  [PickupLatLong] [varchar](50) NULL,  [DropoffLatitude] [float] NULL,  [DropoffLongitude] [float] NULL,  [DropoffLatLong] [varchar](50) NULL,  [PassengerCount] [int] NULL,  [TripDurationSeconds] [int] NULL,  [TripDistanceMiles] [float] NULL,  [PaymentType] [varchar](50) NULL,  [FareAmount] [money] NULL,  [SurchargeAmount] [money] NULL,  [TaxAmount] [money] NULL,  [TipAmount] [money] NULL,  [TollsAmount] [money] NULL,  [TotalAmount] [money] NULL  )  WITH  (  LOCATION='/MillionTrips',  DATA\_SOURCE = [LabNycStorage],  FILE\_FORMAT = compressedcsv,  REJECT\_TYPE = VALUE,  REJECT\_VALUE = 0  )  GO |  * CREATE EXTERNAL TABLE allows one or more column definitions. * The column definitions, including the data types and number of columns must match the data in the external files. If there is a mismatch, the file rows will be rejected when querying the actual data. This data type strictness ensures cleanliness of data and ensures query-ability but requires more care with ETL. * **LOCATION** = 'folder\_or\_filepath'   Specifies the folder or the file path and file name for the actual data in Hadoop or Azure blob storage. The location starts from the root folder; the root folder is the data location specified in the external data source.  In above statements, the location properties are defined at the folder level. This means that queries involving these tables will draw all data from this root level downward before processing.   * **DATA\_SOURCE** = external\_data\_source\_name   Specifies the name of the external data source that contains the location of the external data. This location is either a Hadoop or Azure blob storage.  Here we are referring **LabNycStorage** external data store defined in earlier steps.  This external data store points to a container inside an Azure Blob Storage account.   * **FILE\_FORMAT** = external\_file\_format\_name   Specifies the name of the external file format object that stores the file type and compression method for the external data.  In above statement FILE\_FORMAT is set to object **TextFileFormat** created in earlier steps.   * **Reject Options**   You can specify reject parameters that determine how PolyBase will handle dirty records it retrieves from the external data source. A data record is considered ‘dirty’ if it actual data types or the number of columns do not match the column definitions of the external table.   * **REJECT\_TYPE** = value | percentage   Clarifies whether the REJECT\_VALUE option is specified as a literal value or a percentage.   * **REJECT\_VALUE**   If REJECT\_TYPE is value, then the PolyBase query will fail when the number of rejected rows exceeds the reject\_value.  If REJECT\_TYPE is percentage, then a PolyBase query will fail when the percentage of failed rows exceeds reject\_value. The percentage of failed rows is calculated at intervals.  In the above statement, REJECT\_VALUE is set to 1 to avoid headers in external text file. |  |
| 1. Evaluate performance of external table | SQL Data Warehouse can query data located in Azure Storage Blob and Azure Data Lake Storage. When you query data from external data sources, SQL Data Warehouse will import the entirety of the data source temporarily for query processing.  This query does an aggregate over an external data source. Note the time to completion.   |  | | --- | | SELECT SUM (TotalAmount) AS TotalRevenueByMedallion  , MedallionId  FROM asb .Trip  GROUP BY MedallionId | |  |
| 1. Create final internal table | Clear query window and execute following statement to get data from external table.   |  | | --- | | CREATE TABLE [dbo].[Trip]  (  [TripId] [bigint] IDENTITY(1,1) NOT NULL,  [DateID] [int] NOT NULL,  [MedallionID] [int] NOT NULL,  [HackneyLicenseID] [int] NOT NULL,  [PickupTimeID] [int] NOT NULL,  [DropoffTimeID] [int] NOT NULL,  [PickupGeographyID] [int] NULL,  [DropoffGeographyID] [int] NULL,  [PickupLatitude] [float] NULL,  [PickupLongitude] [float] NULL,  [PickupLatLong] [varchar](50) NULL,  [DropoffLatitude] [float] NULL,  [DropoffLongitude] [float] NULL,  [DropoffLatLong] [varchar](50) NULL,  [PassengerCount] [int] NULL,  [TripDurationSeconds] [int] NULL,  [TripDistanceMiles] [float] NULL,  [PaymentType] [varchar](50) NULL,  [FareAmount] [money] NULL,  [SurchargeAmount] [money] NULL,  [TaxAmount] [money] NULL,  [TipAmount] [money] NULL,  [TollsAmount] [money] NULL,  [TotalAmount] [money] NULL  )  WITH  (  CLUSTERED COLUMNSTORE INDEX  , DISTRIBUTION=HASH([MedallionId])  ) | |  |
| 1. Create schema for staging tables | Clear query window and execute following statements to create new schema for tables.   |  | | --- | | CREATE SCHEMA [stg];  GO | |  |
| 1. Move data into staging | Create table in SQL Data Warehouse using CREATE TABLE AS SELECT (CTAS) feature. It is a fully parallelized operation that creates a new table based on the output of a SELECT statement. The speed at which SQL Data Warehoue can load data is dependent on the number of compute nodes available. With DW200, you receive approximately two times the load performance as a DW100.  In this lab CTAS feature is used to create table in SQL Data Warehouse based on external table defined in earlier step. It not only creates tables but also loads data in table obtained in select statement.  Clear query window and execute following statements to create table in SQL Data Warehouse:   |  | | --- | | CREATE TABLE stg.Trip  WITH  (  HEAP  , DISTRIBUTION=HASH([MedallionId])  )  AS  SELECT \*  FROM [asb].[Trip]  OPTION (LABEL = 'CTAS : Load [stg].[Trip]'); |  * **DISTRIBUTION** HASH ( distribution\_column\_name ) | ROUND\_ROBIN | REPLICATE   The CTAS statement requires a distribution option.  Above table is created using HASH distribution method on column MedallionId   * **LABEL**   This option labels the query which helps to identify query easily. |  |
| 1. Insert data into final table | |  | | --- | | INSERT INTO dbo.Trip  SELECT \*  FROM stg.Trip  OPTION (LABEL = 'Insert into final table : Load [dbo].[Trip]'); |   By loading data into a HEAP staging table, we optimize the speed of data ingestion into. We then load our prepped data into our final production table which takes advantage of a clustered columnstore index and an identity column.  This method is primarily used when optimizing for data ingestion only. If you have complex transformations that proceed this ingestion step, columnstore indexing may provide a better foundation for aggregates and joins. |  |
| 1. Query Final Data | |  | | --- | | SELECT SUM(TotalAmount) AS TotalRevenueByMedallion  , MedallionId  FROM dbo.Trip  GROUP BY MedallionId |   Note the performance difference between this aggregate query when the table data is materialized within SQL Data Warehouse contrasted with step 6. |  |

## Section 5: Manage SQL DW - Scale/Pause/Resume

|  |  |  |
| --- | --- | --- |
| Section | Steps | Screen Shot |
| 1. Scale SQL DW in Azure portal. | 1. Go to your Data Warehouse in Azure Portal. 2. Under **COMMON TASKS** section click **Scale** tab. (Fig. 4.1) 3. In the Scale blade, move the slider left or right to change the DWU setting. (Fig. 4.2). Change you DW to DW100. 4. Click **Save** to change Performance Units. A confirmation message appears. Click **yes** to confirm. (Fig. 4.3). | **Fig. 5.1**    **Fig. 5.2**    **Fig. 5.3** |

## Section 6: Enrich Analysis

|  |  |  |
| --- | --- | --- |
| Section | Steps | Screen Shot |
| 1. Create External File Format for parquet types. | Ensure that your scaling operation has completed before attempting to execute this query. The portal will inform you of when a scaling operation is complete.  In this step, we are creating an external file format definition for Parquet files. This is a columnar storage format widely used by big data technologies.   |  | | --- | | CREATE EXTERNAL FILE FORMAT parquetfile  WITH  (  FORMAT\_TYPE = PARQUET,  DATA\_COMPRESSION = 'org.apache.hadoop.io.compress.SnappyCodec'  ) |     You should notice that your connection was closed by the scaling operation. This is expected as scaling operations are offline. Retrying this query will cause SSMS/Visual Studio to reconnect.  Note – the error will look something like: (provider: TCP Provider, error: 0 - An existing connection was forcibly closed by the remote host.) |  |
| 1. Create External Table | External table data need not always be stored permanently in SQL Data Warehouse. SQL Data Warehouse can utilize data sets produced by other big data technologies to enrich analysis when necessary.  This weather data is stored in blogs in a Parquet format which is why we defined the external file format from earlier.   |  | | --- | | CREATE EXTERNAL TABLE [asb].[Weather]  (  [DateID] [int] NOT NULL,  [GeographyID] [int] NOT NULL,  [PrecipitationInches] [float] NOT NULL,  [AvgTemperatureFahrenheit] [float] NOT NULL  )  WITH  (  LOCATION='/Weather/',  DATA\_SOURCE = [LabNycStorage],  FILE\_FORMAT = parquetfile,  REJECT\_TYPE = VALUE,  REJECT\_VALUE = 0  )  GO |   Note – this table is a small dimension table. This form of querying is not recomended for large files as SQL Data Warehouse today will import the entire file before query processing. |  |
| 1. Query data stored in SQL Data Warehouse with external Weather data. | |  | | --- | | SELECT SUM SUM(wt.PrecipitationInches) AS TotalInchesPrecipitation  , tr.PickupGeographyId  FROM [dbo].[Trip] tr  JOIN [asb].[Weather] wt  ON tr.[.[PickupGeographyID] PickupGeographyID] = wt.GeographyID  GROUP BY tr.PickupGeographyId | |  |

## Section 7: Publish Data (Optional)

|  |  |  |
| --- | --- | --- |
| Section | Steps | Screen Shot |
| 1. Access Azure Preview Portal | 1. Sign in to the [**Azure preview portal**](https://ms.portal.azure.com/). |  |
| 1. Create a new storage account | 1. Click **New** > **Storage** > **Storage account – blob, file, table, queue**. (Fig. 7.1) 2. Provide following values for properties   **Name:** Enter appropriate name for SQL Data Warehouse. You will need this name for step 5. (Fig. 1.14)  **Deployment Model:** Resource Manager**.** (Fig. 7.2)  **Account Kind:** StorageV2. (Fig. 7.2)  **Performance:** Standard. (Fig. 7.2)  **Replication:** LRS. (Fig. 7.2)  **Access tier:** Hot. (Fig. 7.2)  **Secure Transfer required:** Enabled. (Fig. 7.2)  **Subscription:** Select same subscription you used for creating SQL Server in earlier steps.  **Resource Group:** Check Use existing radio button and select Resource Group you created while provisioning SQL Server.   1. Press **Create** button at the bottom of blade to provision new storage account. | **Figure 7.1**    **Figure 7.2** |
| 1. Create a storage container | 1. Go to your Storage Account in Azure portal. 2. Select **Blobs** under **Services** (Fig 7.1) 3. Click **+ Container.** 4. Provide the following values:   **Name**: Create a name for your container. In this example, name it *export*.  **Public access level:** Private (no anonymous access). (Fig 7.3)   1. Select **OK** to create your container. | **Fig 7.1**    **Fig 7.2** |
| 1. Gather the storage account access key | 1. Navigate to your Storage Account in Azure portal. 2. Navigate to **Settings > Access Keys** (Fig 7.3) 3. Copy **key1 > Key.** Copy this value to Notepad or a similar location temporarily. | **Fig 7.3**    **Fig 7.4** |
| 1. Create a connection to your private storage account. | 1. Connect to SQL Data Warehouse using Visual Studio by following steps mentioned in **Section 2: Connect to SQL Data Warehouse**.   Use following properties to make connection: (Fig. 11.1.1)  **SQL Server:** SQL server created for this lab  **Authentication:** SQL Server Authentication  **Username**: LabUserLogin  **Password:** Str0ng\_password  **Database:** SQL Data Warehouse created for this lab.   1. Execute following statements in query window to create setup credentials to your new private storage account. 2. Create a master key and database scoped credential to encrypt and store secrets. You will need the storage account key information from step 4. Fig (7.5.1)  |  | | --- | | CREATE MASTER KEY;  GO  CREATE DATABASE SCOPED CREDENTIAL dsc  WITH  IDENTITY = 'labuser'  , SECRET = '<storage account key from step 4>'  GO |  1. Create External Data Source pointing to your newly created storage account. You will need the storage account name from Step 2.  |  | | --- | | CREATE EXTERNAL DATA SOURCE DataLakeData  WITH (  TYPE = HADOOP,  LOCATION = 'wasbs://export@<storageaccountname>.blob.core.windows.net/',  CREDENTIAL = dsc  ); | | **Fig 7.5.1**    **Fig 7.5.2** |
| 1. Export your analysis | 1. Execute the following query to export your result set out to the private storage account as a parquetfile (Fig 7.6.1):  |  | | --- | | CREATE EXTERNAL TABLE MyAnalysis  WITH  (  LOCATION = 'myanalysis'  , DATA\_SOURCE = DataLakeData  , FILE\_FORMAT = parquetfile  )  AS  SELECT SUM SUM(wt.PrecipitationInches) AS TotalInchesPrecipitation  , tr.PickupGeographyId  FROM [dbo [dbo].[Trip] tr  JOIN [asb [asb].[Weather] wt  ON tr.[.[PickupGeographyID] PickupGeographyID] = wt.GeographyID  GROUP BY tr.PickupGeographyId |  1. Navigate to your Storage Account in Azure portal. 2. Select Blobs (Fig 7.6.2) 3. Select export > myanalysis > myanalysis 4. Observe the exported parquet files. Users of Parquet compatible technologies should now be able to read this data by providing them access to this folder. | **Fig 7.6.1**    **Fig 7.6.2**    **Fig 7.6.3** |

|  |
| --- |
| Terms of use |

© 2014 Microsoft Corporation. All rights reserved.

By using this Hands-on Lab, you agree to the following terms:

The technology/functionality described in this Hands-on Lab is provided by Microsoft Corporation in a “sandbox” testing environment for purposes of obtaining your feedback and to provide you with a learning experience. You may only use the Hands-on Lab to evaluate such technology features and functionality and provide feedback to Microsoft.  You may not use it for any other purpose. You may not modify, copy, distribute, transmit, display, perform, reproduce, publish, license, create derivative works from, transfer, or sell this Hands-on Lab or any portion thereof.

COPYING OR REPRODUCTION OF THE HANDS-ON LAB (OR ANY PORTION OF IT) TO ANY OTHER SERVER OR LOCATION FOR FURTHER REPRODUCTION OR REDISTRIBUTION IS EXPRESSLY PROHIBITED.

THIS HANDS-ONLAB PROVIDES CERTAIN SOFTWARE TECHNOLOGY/PRODUCT FEATURES AND FUNCTIONALITY, INCLUDING POTENTIAL NEW FEATURES AND CONCEPTS, IN A SIMULATED ENVIRONMENT WITHOUT COMPLEX SET-UP OR INSTALLATION FOR THE PURPOSE DESCRIBED ABOVE.  THE TECHNOLOGY/CONCEPTS REPRESENTED IN THIS HANDS-ON LAB MAY NOT REPRESENT FULL FEATURE FUNCTIONALITY AND MAY NOT WORK THE WAY A FINAL VERSION MAY WORK.  WE ALSO MAY NOT RELEASE A FINAL VERSION OF SUCH FEATURES OR CONCEPTS.  YOUR EXPERIENCE WITH USING SUCH FEATURES AND FUNCITONALITY IN A PHYSICAL ENVIRONMENT MAY ALSO BE DIFFERENT.

**FEEDBACK**.  If you give feedback about the technology features, functionality and/or concepts described in this Hands-on Lab to Microsoft, you give to Microsoft, without charge, the right to use, share and commercialize your feedback in any way and for any purpose.  You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a Microsoft software or service that includes the feedback.  You will not give feedback that is subject to a license that requires Microsoft to license its software or documentation to third parties because we include your feedback in them.  These rights survive this agreement.

MICROSOFT CORPORATION HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS WITH REGARD TO THE HANDS-ON LAB , INCLUDING ALL WARRANTIES AND CONDITIONS OF MERCHANTABILITY, WHETHER EXPRESS, IMPLIED OR STATUTORY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT.  MICROSOFT DOES NOT MAKE ANY ASSURANCES OR REPRESENTATIONS WITH REGARD TO THE ACCURACY OF THE RESULTS, OUTPUT THAT DERIVES FROM USE OF THE VIRTUAL LAB, OR SUITABILITY OF THE INFORMATION CONTAINED IN THE VIRTUAL LAB FOR ANY PURPOSE.