# Creating the Azure Databricks Lab Environment

**Create the Azure Databricks workspace using the Premium Pricing Tier.**

**Create the SQL Data Warehouse**

There are some pre-requisites for connecting Azure Databricks with SQL Data Warehouse that apply to the SQL Data Warehouse:

1. Create an in instance of SQL Data Warehouse and for the “select source”, choose Sample and then for Select sample choose AdventureWorksDW. A Gen1 size of DW100 is more than sufficient.
2. You need to [create a database master key](https://docs.microsoft.com/en-us/sql/relational-databases/security/encryption/create-a-database-master-key) for the Azure SQL Data Warehouse. For example:

-- Creates a database master key for the "databricks-sqldw" database.

-- The key is encrypted using the password.

USE [databricks-sqldw];

GO

CREATE MASTER KEY ENCRYPTION BY PASSWORD = '980AbctotheCloud427leet';

GO

1. You need to ensure that the [Firewall](https://docs.microsoft.com/en-us/azure/sql-database/sql-database-firewall-configure#manage-firewall-rules-using-the-azure-portal) on the Azure SQL Server that contains your SQL Data Warehouse is configured to allow Azure services to connect (e.g., Allow access to Azure services is set to On).

**Create the Blob Storage Account for use with SQL Data Warehouse**

Azure Storage blobs are used as the intermediary for the exchange of data between Azure Databricks and Azure SQL Data Warehouse. As a result of this, you will need:

1. To create a general purpose v1 Azure Storage account
2. Acquire the Account Name and Account Key for that Storage Account
3. Create a storage container dwtemp that will be used to store data used during the exchange (this must exists before you run an queries against SQL DW)

**Create the Cosmos DB**

1. Create a Cosmos DB account with the following properties:
   1. API: SQL
   2. Enable geo-redundancy: checked
2. Create a database called demos
3. Create a collection called documents (unlimited storage capacity, with 1000 RU’s is plenty, partition key “partitionId”)
4. Acquire the key to the Cosmos DB account

**Create an Event Hubs instance**

1. Create an Event Hub named databricks-demo-eventhub
   1. Use the Basic pricing tier
2. Once the namespace is created, create an Event Hub with the same name
   1. 2 partitions is sufficient
3. On the Event Hub create a policy named “both” that has Listen and Send permissions
4. Capture the Key for the Event Hub

**Upload the instructor and student notebooks**

Copy the contents of the Instructor folder to an Instructor folder in the Workspace \ Shared.

Copy the contents of the Student folder to an Student folder in the Workspace \ Shared.

**Create an unsecured cluster called "labs-standard" with:**

Cluster Type: Standard

Databricks Runtime Version: 4.0

Python Version: 3.0

Driver Type: Standard\_F4s

Worker Type: Standard\_F4s

Min Workers: 2

Max Workers: 4

Enable Autoscaling: Checked

Auto Termination: 480 minutes (8 hours)

Table Access Control: Enable table access control NOT checked

**Create a secured cluster called "labs-high-conc (secured)" with:**

Cluster Type: High-Concurrency

Databricks Runtime Version: 4.0

Python Version: 3.0

Driver Type: Standard\_F4s

Worker Type: Standard\_F4s

Min Workers: 2

Max Workers: 4

Enable Autoscaling: Checked

Auto Termination: 480 minutes (8 hours)

Table Access Control: Enable table access control checked

**Create an attach the required libraries to all clusters**

You will need to create libraries, using the [Create Library](https://docs.azuredatabricks.net/user-guide/libraries.html) interface in Azure Databricks, for the following and attach them to all clusters.

In the Shared workspace, create a new folder called libs and then create the following libraries in the libs folder:

*spark-deep-learning*

* Source: Maven Coordinate
* Select Search Spark Packages and Maven Central
* In the Search Packages dialog, search for `spark-deep-learning`, confirm the release is `1.0.0-spark2.3-s\_2.11`
* Choose +select
* Select Create library
* After the library is created, select attach automatically to all clusters

*tensorflow*

* Source: Upload Python Egg or PyPi
* PyPi Name: `tensorflow==1.6.0`
* Choose +select
* Select Install Library
* After the library is created, select attach automatically to all clusters

*keras*

* Source: Upload Python Egg or PyPi
* PyPi Name: `keras==2.1.5`
* Choose +select
* Select Install Library
* After the library is created, select attach automatically to all clusters

*azure-cosmosdb-spark*

* Source: Maven Coordinate
* Select Search Spark Packages and Maven Central
* In the Search Packages Packages dialog, change the dropdown to Maven Central then search for `azure-cosmosdb-spark`, confirm the artifact id is `azure-cosmosdb-spark\_2.2.0\_2.11` and the release is `1.0.0`
* Choose +select
* Select Install Library
* After the library is created, select attach automatically to all clusters

*azure-eventhubs-spark*

* Source: Maven Coordinate
* Select Search Spark Packages and Maven Central
* In the Search Packages dialog, select Maven Central and then search for `azure-eventhubs-spark`, confirm the artifact id is `azure-eventhubs-spark\_2.11` and the release is `2.3.1`
* Choose +select
* Select Install Library
* After the library is created, select attach automatically to all clusters

IMPORTANT: After installing these libraries, restart the clusters.

**Create the usedcars table**

Download the data for this table from: https://databricksdemostore.blob.core.windows.net/data/usedcars/UsedCars.csv

In the default database, add a table by uploading this file.

Select Create Table with UI

Select a Cluster to Preview the Table, choose any cluster and select Preview Table

In Specify Table Attributes, provide the table name of "usedcars"

Check the box next to First row is header

Select Create Table

**Create the usedcars\_clean table**

Download the data for this table from: https://databricksdemostore.blob.core.windows.net/data/usedcars/UsedCars\_Clean.csv

In the default database, add a table by uploading this file.

Select Create Table with UI

Select a Cluster to Preview the Table, choose any cluster and select Preview Table

In Specify Table Attributes, provide the table name of "usedcars\_clean"

Check the box next to First row is header

Select Create Table

**Create the usedcars\_secured table**

Download the data for this table from: https://databricksdemostore.blob.core.windows.net/data/usedcars/UsedCars\_Clean.csv

In the default database, add a table by uploading this file.

Select Create Table with UI

Select a Cluster to Preview the Table, choose any cluster and select Preview Table

In Specify Table Attributes, provide the table name of "usedcars\_secured"

Check the box next to First row is header

Select Create Table

**Create the user accounts that will access the Azure Databricks Workspace**

**Enable All Users to attach to all clusters**

For both clusters, set it so "all users" have "can attach to" permission.

See this for instructions: https://docs.azuredatabricks.net/administration-guide/admin-settings/cluster-acl.html#configure-individual-cluster-permissions

**Enable user access on a per user basis**

For *each* user added to AAD that will run the labs, the following needs to be done.

Add the user to the Workspace using the Admin Console. They should NOT be admin and they should NOT have Allow cluster creation.

Configure that user's permissions by running the following SQL commands within a notebook. Be sure to use ticks instead of quotes around the email address, and to provide the user's email address in each line:

GRANT CREATE ON DATABASE default to `databricks01@solliance.net`;

GRANT SELECT ON usedcars to `databricks01@solliance.net`;

GRANT SELECT ON usedcars\_clean to `databricks01@solliance.net`;

GRANT READ\_METADATA ON usedcars to `databricks01@solliance.net`;

GRANT READ\_METADATA ON usedcars\_clean to `databricks01@solliance.net`;

GRANT READ\_METADATA ON usedcars\_secured to `databricks01@solliance.net`;