KIRK DAHL
McKesson **Assignment 06** Deep Azure

Handed out: 11/15/2017 Due by 11:59 PM, midnight (CST) on Tuesday, 11/21/2017

You can do this assignment on your PC, Mac, Linux on Windows or Linux VM. Tell us which environment you are using. Implementation of this assignment will require some light reading of Azure literature on the Internet. If the assignment refers to CLI, you are welcome to use Power Shell or even one of programming languages if that is more convenient for you.

Problem 01. Create an Azure blob container and move the attached file `storage_table_demo.py` as a Blob into that container. Demonstrate that you can modify the file and upload it to the same container again. This time download the file back to your operating system and prove that you received back file with recent modifications. Delete the blob, container and the storage account. Do it all using Azure CLI.
(15%)

CREATE RESOURCE GROUP - RG-KIRK

```
kirks-mbp:~ el5vgxz$ az group create --name rg-kirk --location "West US"
{
```

CREATE STORAGE ACCOUNT (truncated output)

```
kirks-mbp:~ el5vgxz$ az storage account create --name sakirk -g rg-kirk -l "West US"
--sku Standard_LRS --encryption blob
{
  "accessTier": null,
  "creationTime": "2017-11-20T19:22:23.598668+00:00",
  "customDomain": null,
  "enableHttpsTrafficOnly": false,
  "encryption": {
    "keySource": "Microsoft.Storage",
    "keyVaultProperties": null,
    "services": {
      "blob": {
        "enabled": true,
        "lastEnabledTime": "2017-11-20T19:22:23.605669+00:00"
      },
      "file": {
        "enabled": true,
        "lastEnabledTime": "2017-11-20T19:22:23.605669+00:00"
      },
      "queue": null,
      "table": null
    }
  },
  "id": "/subscriptions/d5f65876-4dec-4a2e-afe1-b61e713a8612/resourceGroups/rg-
kirk/providers/Microsoft.Storage/storageAccounts/sakirk",
```

DISPLAY KEYS

```
kirks-mbp:~ el5vgxz$ az storage account keys list --account-name sakirk --resource-
group rg-kirk --output table
KeyName    Permissions    Value
---------  -------------  ----------------------------------------------------------
-----------------------------
key1       Full           9lsYtzs1kGOdSULPlUXgfh+V/A/rTVHa1wozF/tVyZzya1PsMNqAWddKxo
BKzCv5h819Z3fcGk4bcWRbu7Aogg==
key2       Full           m2mBLXbHBD3/CAADwpow6WNKlUPlZp41hjDnXghXFqVw+7C0EMIArUfPgZ
h6/gvEsHoIH6YZezsLmvnCVNt5/w==
```

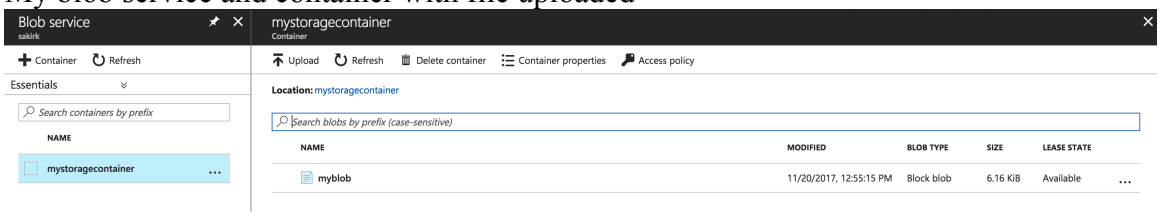
CREATE CONTAINER

```
kirks-mbp:~ el5vgxz$ az storage container create --name mystoragecontainer
{
  "created": true
}
```

UPLOAD FILE

```
kirks-mbp:Azure Training el5vgxz$ az storage blob upload --container-name
mystoragecontainer --name myblob --file ./storage_table_demo.py
Finished[#############################################################]  100.0000%
{
  "etag": "\"0x8D530509E10038C\"",
  "lastModified": "2017-11-20T19:55:15+00:00"
}
```

My blob service and container with file uploaded



MODIFY FILE ADDING LINE TO END – TAIL LAST 2 LINES – UPLOAD NEW FILE – MODIFICATION TIME ON FILE CHANGED

```
kirks-mbp:Azure Training el5vgxz$ echo "File modified by Kirk" >>
storage_table_demo.py
kirks-mbp:Azure Training el5vgxz$ tail -n2 storage_table_demo.py
    print('Error deleting resource group.')
File modified by Kirk
kirks-mbp:Azure Training el5vgxz$ az storage blob upload --container-name
mystoragecontainer --name myblob --file ./storage_table_demo.py
Finished[#############################################################]  100.0000%
{
  "etag": "\"0x8D530520B04AD87\"",
  "lastModified": "2017-11-20T20:05:27+00:00"
}
```

DELETE FILE AND DOWNLOAD FILE S "NEWFILE"

```
kirks-mbp:Azure Training el5vgxz$ rm storage_table_demo.py
kirks-mbp:Azure Training el5vgxz$ ls stora*
ls: stora*: No such file or directory
kirks-mbp:Azure Training el5vgxz$ az storage blob download --container-name
mystoragecontainer --name myblob --file ./newfile
{
  "content": null,
  "metadata": {},
```

```
  "name": "myblob",
  "properties": {
```

TAIL NEWFILE SHOWING MODIFICATIONS

```
kirks-mbp:Azure Training el5vgxz$ tail -n3 newfile
else:
    print('Error deleting resource group.')
File modified by Kirk
```

DELETE EVERYTHING INDIVIDUALLY

```
kirks-mbp:Azure Training el5vgxz$ az storage blob delete --container-name
mystoragecontainer --name myblob
{
  "deleted": null
}
kirks-mbp:Azure Training el5vgxz$ az storage container delete --name
mystoragecontainer
{
  "deleted": true
}
kirks-mbp:Azure Training el5vgxz$ az storage account delete --resource-group rg-kirk
--name sakirk
Are you sure you want to perform this operation? (y/n): y
kirks-mbp:Azure Training el5vgxz$ az group delete --name rg-kirk
Are you sure you want to perform this operation? (y/n): y
kirks-mbp:Azure Training el5vgxz$
```

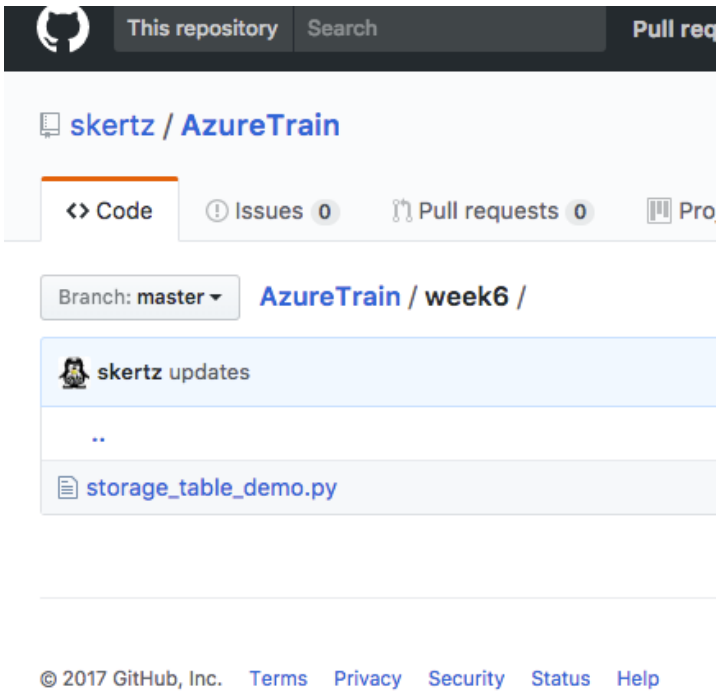
Problem 02. Examine attached file table `storage_table_demo.py`. This is the code we discussed in class which creates and populates Azure Table structure called "`itemstable`". Examine the code carefully. It contains practically all the tools you need to deal with Azure Tables. Replace prefix "`zdj`" of all object names with a short string unique for you. Rather than populate table with Pizzas, populate one of its partitions with cars, as if you are a car dealership. Cars are characterized by make, model, year, color and price. Populate yet another partition with coffee shop inventory. Coffee is characterized by the brand, flavor, size of the cup and price per cup. Place your modified file in your GitHub repository. Open Portal's Cloud Shell and transfer file from your repository into your home directory of the Cloud Shell. Cloud Shell does have `git` installed. Run your Python program using command: `python2.7 storage_table_demo.py`. Capture the output. Notice that program waits for your commands. Before you delete your table `itemstable`, open Azure Storage Explorer and capture the content of your table. Could you think of another way of transferring files to your Cloud Shell home directory?
(20%)

FILE MODIFIED FOR CAR AND COFFEE

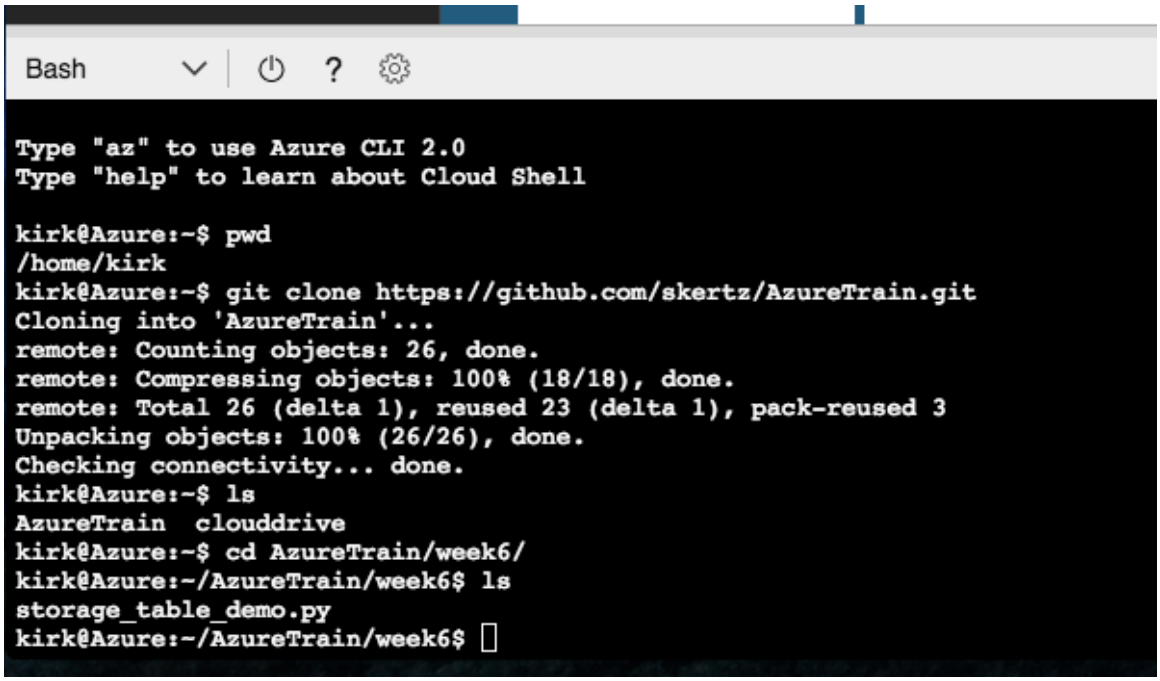
```
# A partition key tracks how like-minded entries in the Table are created and
queried.
# A row key is a unique ID for each entity in the partition
# These two properties are used as a primary key to index the Table. This makes
queries much quicker.

car = Entity()
car.PartitionKey = 'Inventory'
car.RowKey = '001'
```

```
car.year = '2017'
car.make = 'Tesla'
car.model = 'Model S'
car.color = 'Black'
car.price = 72000
table_service.insert_entity('itemstable', car)
print('Created entry for Kirks car')

car.PartitionKey = 'Inventory'
car.RowKey = '002'
car.year = '2003'
car.make = 'BMW'
car.model = '330 ci'
car.color = 'Black'
car.price = 5000
table_service.insert_entity('itemstable', car)
print('Created entry for Matts car')



coffee = Entity()
coffee.PartitionKey = 'coffehouse'
coffee.RowKey = '002'
coffee.brand = 'Folders'
coffee.flavor = 'regular'
coffee.size = 'large'
coffee.cost = 1.99
table_service.insert_entity('itemstable', coffee)
print('Created entry for a large regualre Folders...\n')
time.sleep(1)

coffee = Entity()
coffee.PartitionKey = 'coffehouse'
coffee.RowKey = '001'
coffee.brand = 'Maxwell'
coffee.flavor = 'decaf'
coffee.size = 'small'
coffee.cost = .98
table_service.insert_entity('itemstable', coffee)
print('Created entry for a small decaf Maxwell...\n')
time.sleep(1)
```

FILE UPLOADED TO GITHUB



SETUP CLOUD SHELL AND GIT CLONE MY REPO

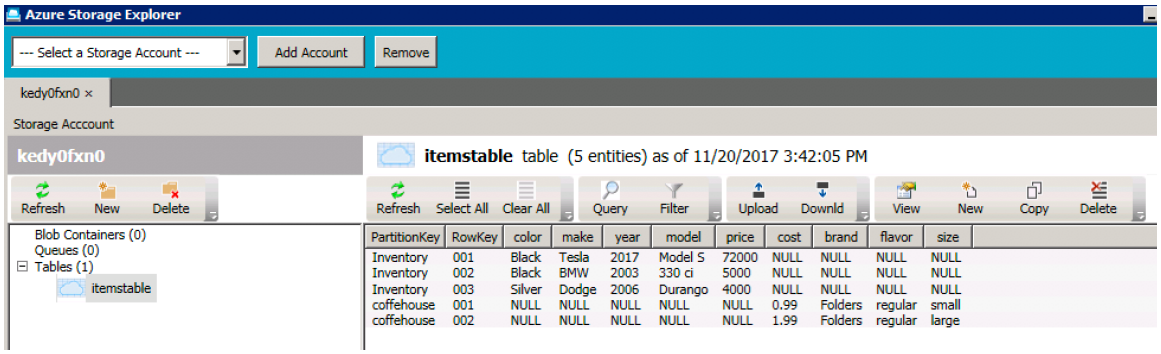


EXECUTE PYTHON SCRIPT

```
kirk@Azure:~/AzureTrain/week6$ python2.7 storage_table_demo.py
Resource group: kedqrp9q1 created successfully.
```

```
Storage account: ked8lqwpj created successfully.

Let's create an Azure Storage Table to store some data.
Press Enter to continue...
Storage Table: itemstable created successfully.

Now let's add some entries to our Table.
Remember, Azure Storage Tables is a NoSQL datastore, so this is
similar to adding records to a database.
Press Enter to continue...
Created entry for Kirks car
Created entry for Matts car
Created entry for Renees car
Created entry for a small regular Folders...

Created entry for a large regualre Folders...

Created entry for a small decaf Maxwell...

With some data in our Azure Storage Table, we can query the data.
Let's see what the car inventory looks like.
Press Enter to continue...

This is a basic example of how Azure Storage Tables behave like a
database.
To keep things tidy, let's clean up the Azure Storage resources
we created.
Press Enter to continue...
Storage table: itemstable deleted successfully.
Resource group: kedqrp9q1 deleted successfully.
```

USE AZURE STORAGE EXPLORER



Problem 03. Using Azure CLI create a File Share. Move two or free files to that file share. Create two or three directories in that file share. Subsequently, mount that file share as a shared drive on your Windows or MacOS machine. Demonstrate that you can place files from your Windows or MacOS into one of the directories on the shared drive and that they will be visible in Azure Portal.
(20%)

CREATE RESOURCE GROUP

```
kirks-mbp:week6 el5vgxz$ az group create --name rg-kirk --location "West US"
```

CREATE STORAGE ACCOUNT (kind = file)

```
kirks-mbp:week6 el5vgxz$ az storage account create --kind Storage --resource-group
rg-kirk --name sakirk --sku Standard_LRS
{
  "accessTier": null,
  "creationTime": "2017-11-20T21:54:30.949268+00:00",
  "customDomain": null,
```

DISPLAY CONNECTION STRING

```
kirks-mbp:week6 el5vgxz$ az storage account show-connection-string -n sakirk -g rg-
kirk --query 'connectionString' -o tsv
DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;AccountName=sakirk;Ac
countKey=FbiiQ+9LqkkO6Kgc+qfbWBn0MEEpiCIB6/uym7y+xWSQJm27Xt6IhDYqQI7wfuGCvE5yU9BvRI/
UFE83UlQ4vQ==
```

CREATE FILE SHARE

```
kirks-mbp:week6 el5vgxz$ az storage share create --name kirkshare --account-name
sakirk
{
  "created": true
}
```

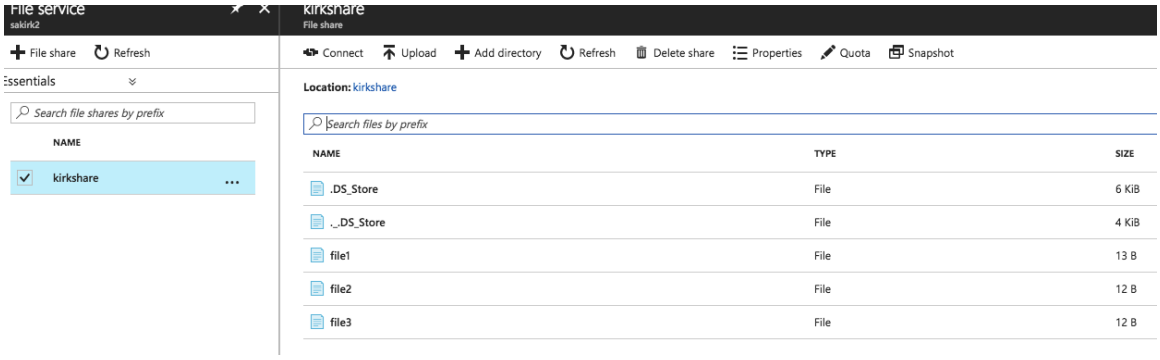
MOUNT SHARE (showing with df command)

```
kirks-mbp:week6 el5vgxz$ mount -t smbfs -o -f=0777,-d=0777
//sakirk2.file.core.windows.net/kirkshare azuremount

kirks-mbp:week6 el5vgxz$ df | grep azu
//sakirk2@sakirk2.file.core.windows.net/kirkshare 4294967296       128
4294967168     1%       2           134217724    0%   /Users/el5vgxz/Desktop/code/Az
ureTrain/week6/azuremount
```

COPY 3 TEST FILES TO SHARE/MOUNT

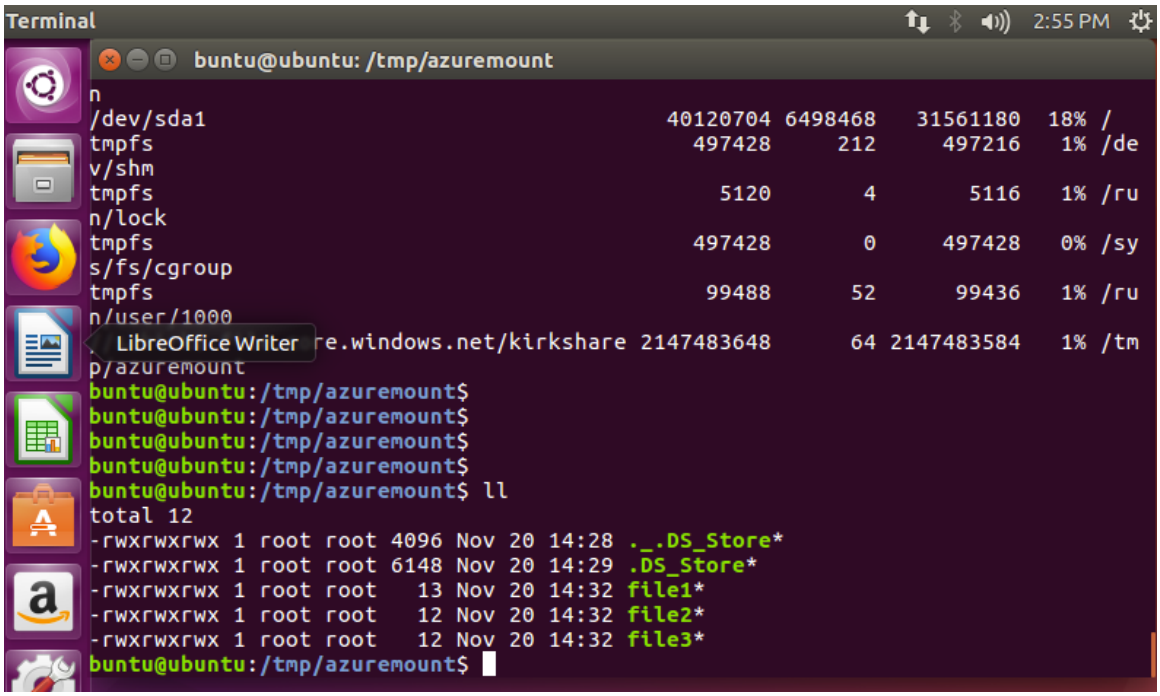
```
kirks-mbp:week6 el5vgxz$ ls
azuremount           file1                file2                file3
       storage_table_demo.py
kirks-mbp:week6 el5vgxz$ cp file* azuremount/
kirks-mbp:week6 el5vgxz$
```

SHOWING FILES EXIST ON SHARE

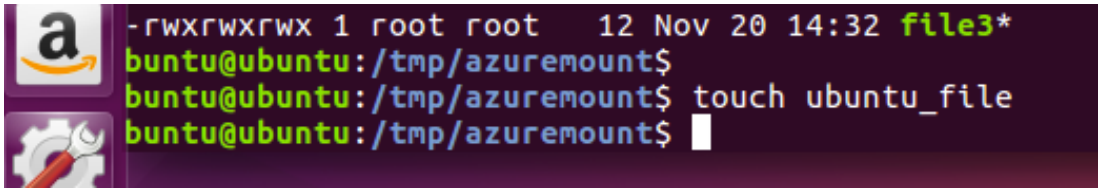


Problem 04. Mount the file share you created in problem 3 as a shared drive on your Linux VM, either a CentOS or Ubuntu. This might require some Internet search. You need to find out how to open port 445 on your VM. Demonstrate that you can navigate to the shared folder, create a new directory using Linux `mkdir` command and copy a file from your Linux OS into that directory. Verify that the directory and the file are visible in Azure Portal. Could you find Azure CLI or Power Shell command to list the content of your Azure File Share
(20%)

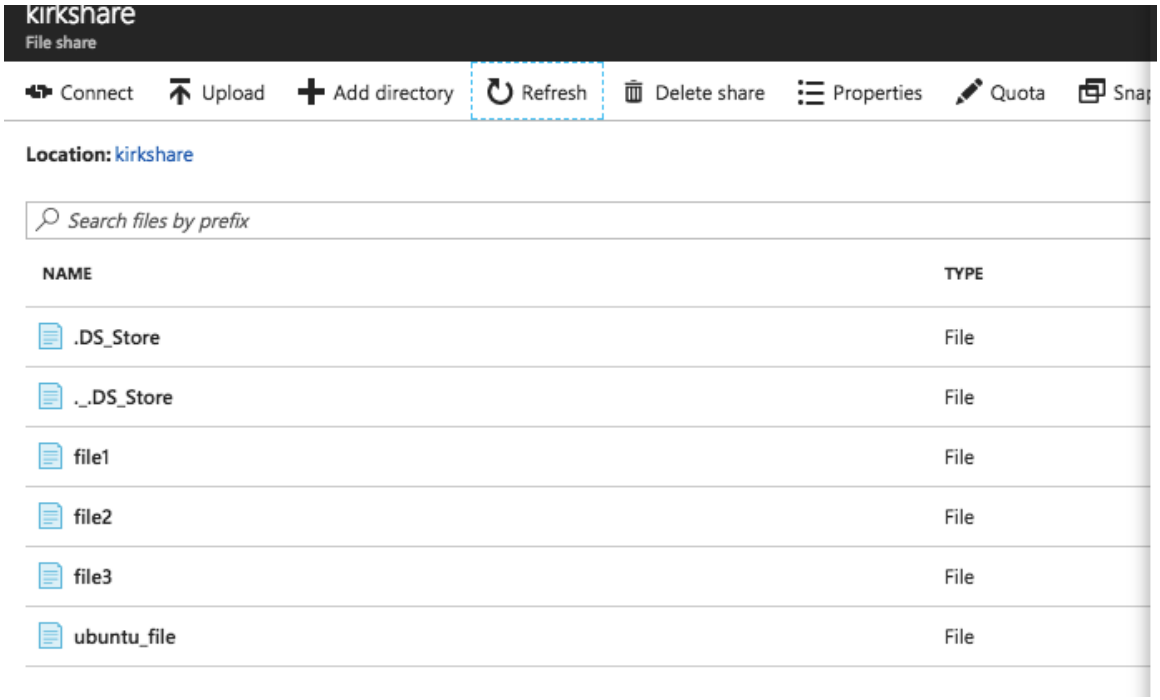
MOUNTED SHARE IN UBUNTU – SEE DF COMMAND AND LS –L of existing files



CREATE FILE FROM UBUNTU WITH "TOUCH"



FILE (ubuntu_file) IS AVAILABLE ON PORTAL



AZ CLI TO DISPLAY FILES IN SHARE

```
kirks-mbp:week6 el5vgxz$ az storage file list --share-name kirkshare --account-name
sakirk2 | grep name
    "name": ".DS_Store",
    "name": "._.DS_Store",
    "name": "file1",
    "name": "file2",
    "name": "file3",
    "name": "ubuntu_file",
```

Problem 05. Follow the slides from Lab 06 on creation of a data factory for the Visual Studio (C#) or for Python. You are expected to follow one or the other approach. Create all resources noted in the slides. For the data create a file called `astroplayers.txt` with 5 players from the World Series Houston Astros Roster (first and last names) or use names from your favorite team. Create a container called `playerscontainer`. Upload `astroplayers.txt` to a folder in your container called `myteam`. After you run your code check that all of your 5 favorite players are in your SQL database table (Visual Studio) or your container's output folder (Python) using the Azure portal.
(20%)

INPUT PARMS

AZ Subscription ID: d5f65876-4dec-4a2e-afe1-b61e713a8612
Storage Name: saskertz234564
Storage
Key: mEw3N5/jQ0cPEi9scSQ+y35AUxKUt/lS2tlpERA1QWO9G+hm+KtOsj/K+qOz/PDkfku+JoPfYN7fUAi91W2NSA==
Container & Folder: myblob/input

Data File Name: employee.txt
Application ID: 4e2e5043-f67b-4bd7-849a-dad4fe1d53ab
Secret Key: Mykdappkey: 3wzRDpdnn3HGZN0vOIGKQ3zIHzbPGFFfVGLInNHzviE=
Tenant ID: cd28ef90-210d-4c10-b290-de42a4d6adfe

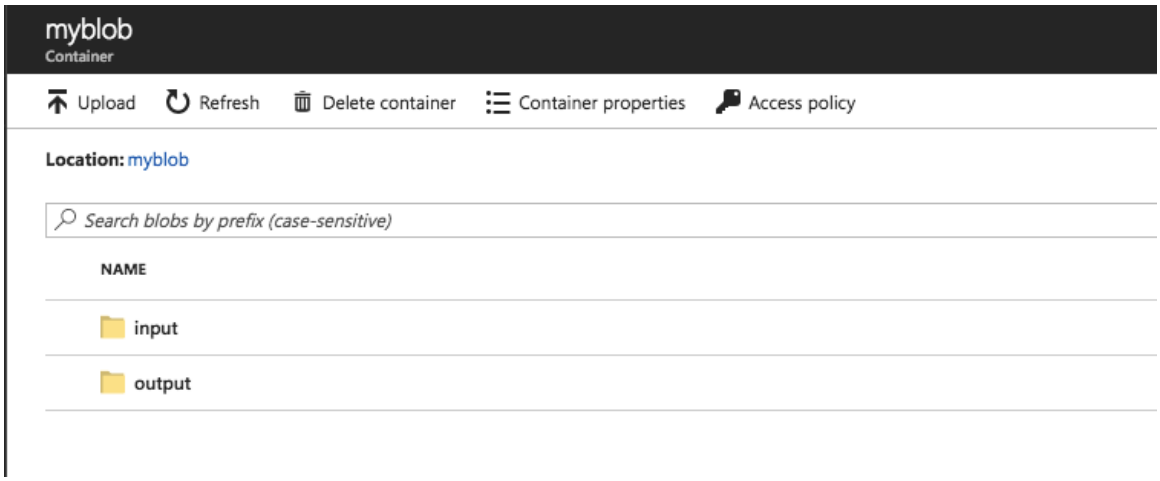
OUTPUT

jupyter Untitled Last Checkpoint: a minute ago (autosaved)

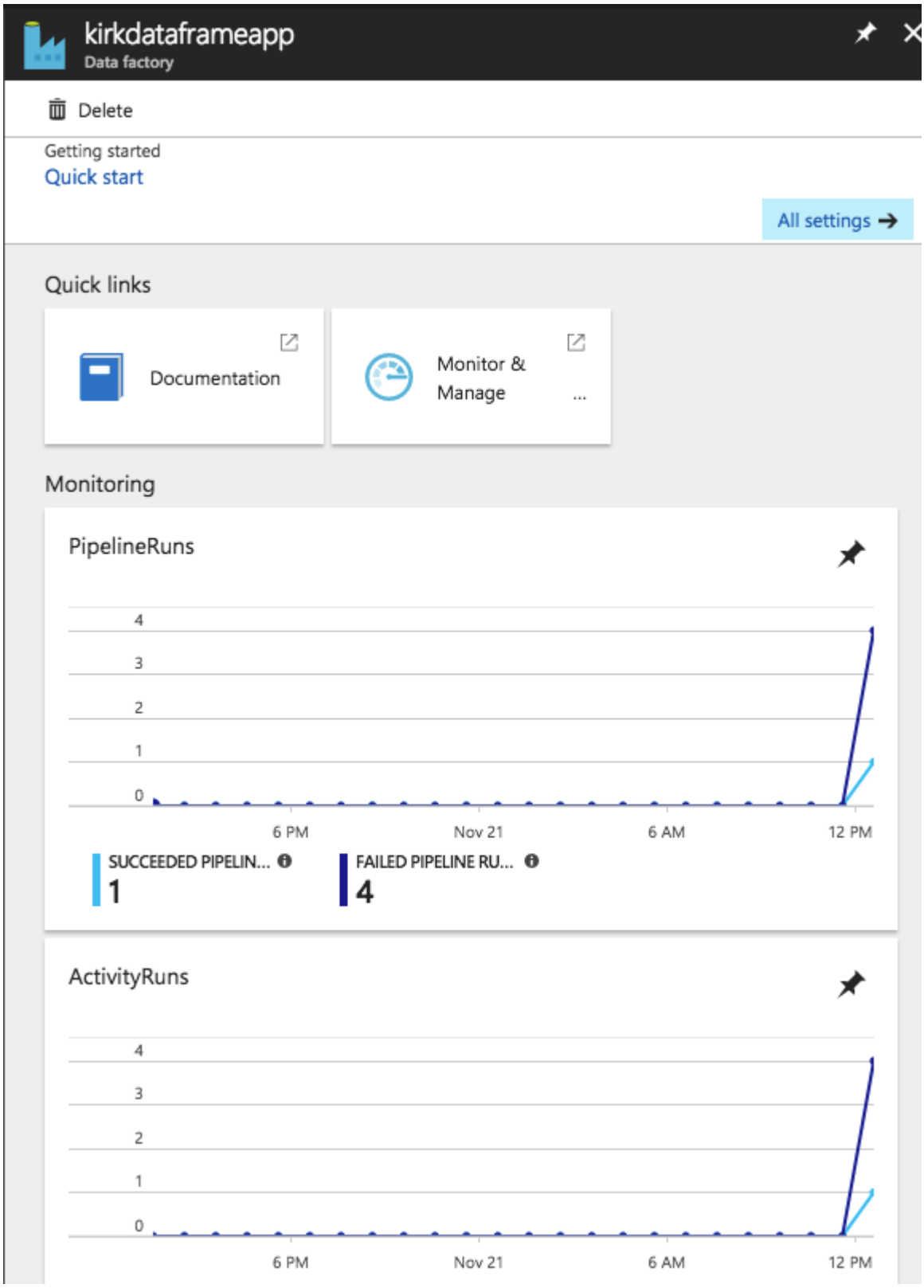
File Edit View Insert Cell Kernel Widgets Help Trusted Python 2

```
        Name: kirkdataframeapp
        Id: /subscriptions/d5f65876-4dec-4a2e-afe1-b61e713a8612/resourceGroups/rg-kirkeast/providers/Microsoft.DataFa
ctory/factories/kirkdataframeapp
        Location: eastus
        Tags: {}


        Name: storageLinkedService
        Id: /subscriptions/d5f65876-4dec-4a2e-afe1-b61e713a8612/resourceGroups/rg-kirkeast/providers/Microsoft.DataFa
ctory/factories/kirkdataframeapp/linkedservices/storageLinkedService



        Name: ds_in
        Id: /subscriptions/d5f65876-4dec-4a2e-afe1-b61e713a8612/resourceGroups/rg-kirkeast/providers/Microsoft.DataFa
ctory/factories/kirkdataframeapp/datasets/ds_in



        Name: ds_out
        Id: /subscriptions/d5f65876-4dec-4a2e-afe1-b61e713a8612/resourceGroups/rg-kirkeast/providers/Microsoft.DataFa
ctory/factories/kirkdataframeapp/datasets/ds_out



        Name: copyPipeline
        Id: /subscriptions/d5f65876-4dec-4a2e-afe1-b61e713a8612/resourceGroups/rg-kirkeast/providers/Microsoft.DataFa
ctory/factories/kirkdataframeapp/pipelines/copyPipeline


        Pipeline run status: Succeeded

        Activity run details

        Activity run status: Succeeded
        Number of bytes read: 58
        Number of bytes written: 58
        Copy duration: 4
```

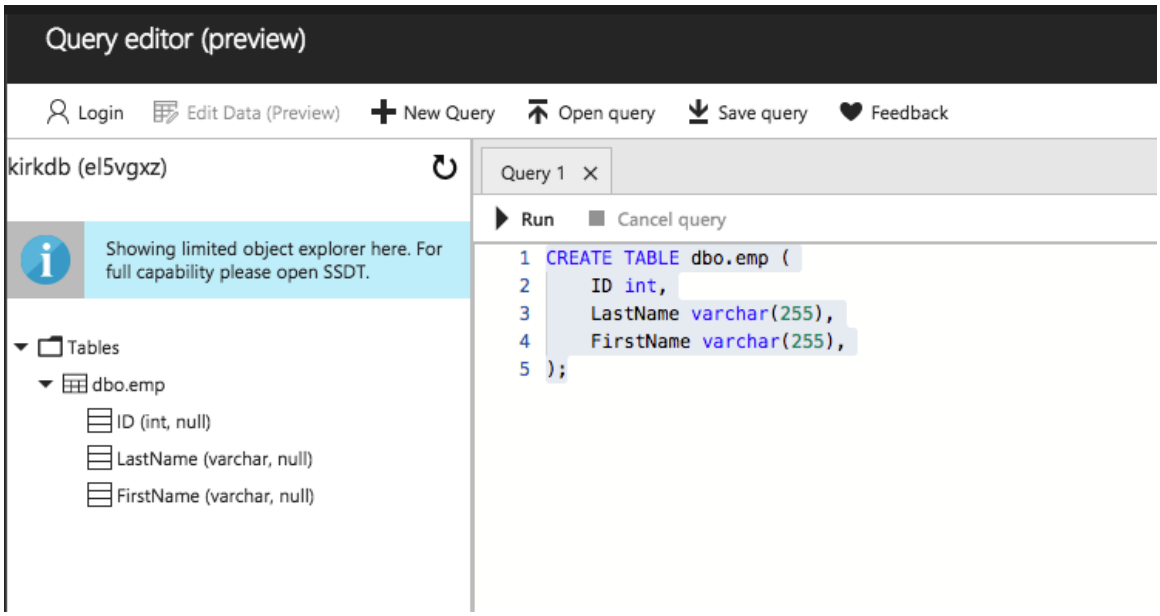
I NOW HAVE OUTPUT FOLDER



SHOWING MY PIPELINE RUNES


kirkdataframeapp
Data factory
Delete
Getting started
Quick start
All settings
Quick links
Documentation
Monitor & Manage
Monitoring
PipelineRuns
4
3
2
1
0
6 PM
Nov 21
6 AM
12 PM
SUCCEEDED PIPELIN...
1
FAILED PIPELINE RU...
4
ActivityRuns
4
3
2
1
0
6 PM
Nov 21
6 AM
12 PM

CREATED DB



Problem 06. Remove all resource groups created in this assignment using Azure CLI. Please use CLI to show that you have no resources left.
(5%)

LIST RESOURCE GROUPS AND DELETE

```
kirks-mbp:week6 el5vgxz$  az group list
[
  {
    "id": "/subscriptions/d5f65876-4dec-4a2e-afe1-b61e713a8612/resourceGroups/cloud-
shell-storage-eastus",
    "location": "eastus",
    "managedBy": null,
    "name": "cloud-shell-storage-eastus",
    "properties": {
      "provisioningState": "Succeeded"
    },
    "tags": null
  },
  {
    "id": "/subscriptions/d5f65876-4dec-4a2e-afe1-b61e713a8612/resourceGroups/rg-
kirkeast",
    "location": "eastus",
    "managedBy": null,
    "name": "rg-kirkeast",
    "properties": {
      "provisioningState": "Succeeded"
    },
    "tags": null
  }
]
kirks-mbp:week6 el5vgxz$
kirks-mbp:week6 el5vgxz$ az group delete --name rg-kirkeast
Are you sure you want to perform this operation? (y/n): y
kirks-mbp:week6 el5vgxz$ az group delete --name cloud-shell-storage-eastus
```

```
Are you sure you want to perform this operation? (y/n): y
kirks-mbp:week6 el5vgxz$
```

SUBMISSION INSTRUCTIONS:

Your main submission should be a MS Word or PDF document containing descriptions of your action while configuring Azure services. If your MS Word document is larger than 1 MB, save it as a MINIMIZED PDF. Please be merciful and capture small JPGs. Describe the purpose of every action and the significance of the results. Start with the text of this homework assignment as the template. Please add any other files that you might have used or generated. Please write your solution as if you are writing a tutorial for your colleagues. Please make your text readable. Make sure that your fonts, especially in captured images are not unreadable. Please do not provide ZIP or RAR or any other archives. Canvas cannot open those archives and they turn into a nuisance for us.