

# Cloud Computing for Science

## Part 1. Managing Data in the Cloud

Dennis Gannon

# Motivating Examples

- BIG objects
  - Climate scientists with big simulation output files in NetCDF.
    - Assume 10 TB in big objects
- Many small CVS files
  - Environmental Engineers with 1 Million records of observations each in CSV format
    - May be 100 TB total
- Streams
  - Scientists with a distributed collection of several thousand instruments.
    - Each generates a stream of records that must be collected and analyzed every few hours or continuously

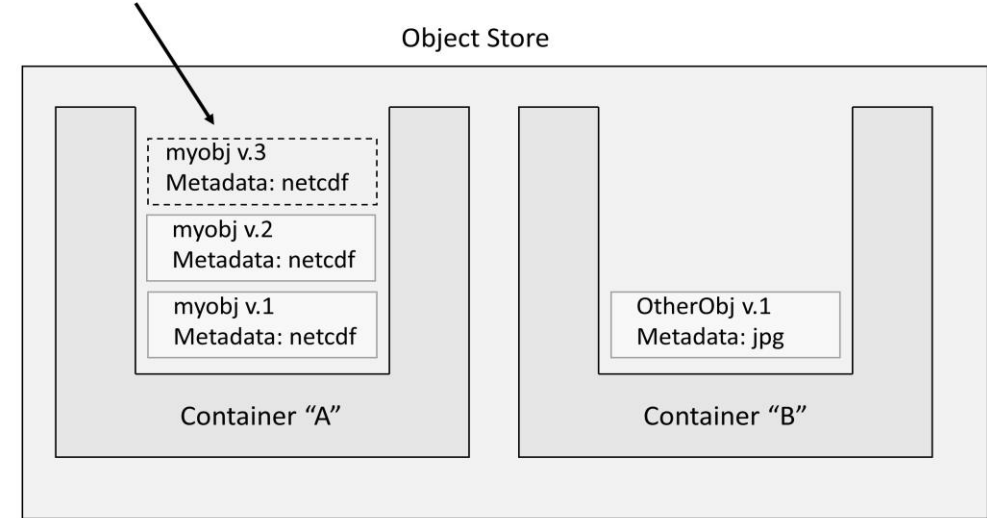
# Types of Cloud Data Storage Systems

- Basic Blob Object store
  - Buckets of immutable objects
  - Highly scalable & reliable
- Databases
  - SQL Style relational databases
  - NoSQL storage
  - Data warehouses
- Attached File stores
- Graph databases
- Streaming systems

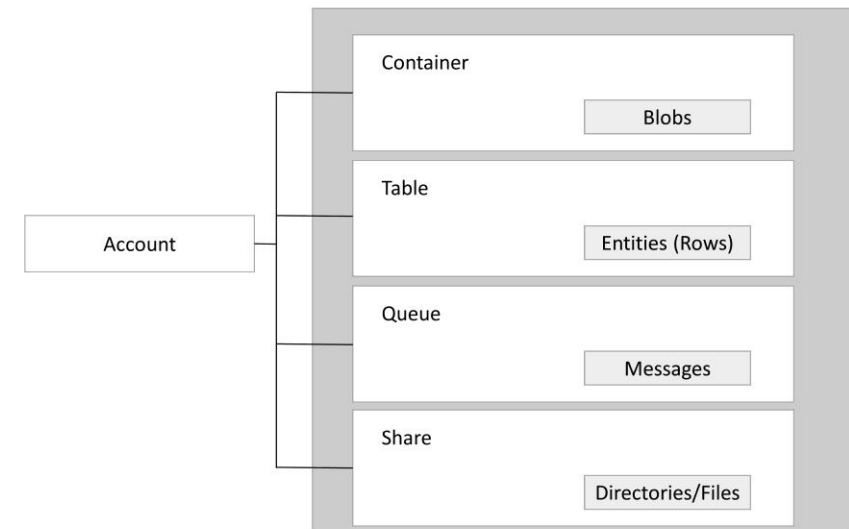
# Object stores

- Amazon AWS
  - S3- buckets of immutable objects
    - Organized in a 2-level folder system
    - Each object has associated metadata
- Microsoft Azure
  - Storage accounts contain blob storage along with tables, queues and file shares.
  - Blob containers are similar to S3
- Google Cloud Storage
  - Different models based on availability and cost
- OpenStack does not have a standard but semi-standards exist and various ones are used in various deployment

```
PutObject(myobj, Container='A', metadata = 'NetCDF')
```

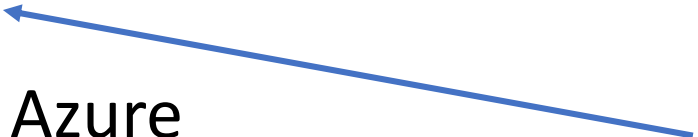


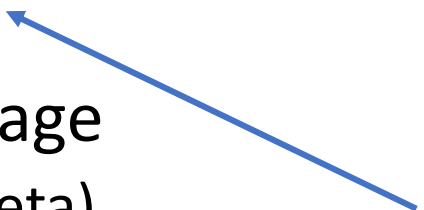
## Azure storage account structure



# Relational Databases

- Amazon AWS
  - Relational Data Services (RDS)
  - Aurora
- Microsoft Azure
  - Azure SQL
    - 3 service tiers
    - Premium tier up to 1TB
    - Up to 30000 concurrent sessions
  - Azure Data Lake
- Google Cloud Storage
  - Cloud Spanner (beta)
    - Full relational
    - Strongly consistent
    - Scalable to thousands of servers

- 
- Aurora is distributed
    - Scalable from 2 vCPUs to 32 vCPUs
    - Data up to 64TB
    - MySQL compatible
    - Fully geo-replicated

- 
- Data Lake is a platform
    - Structured & unstructured data
    - Scalable to petabytes
    - Distributed and designed to support analytics

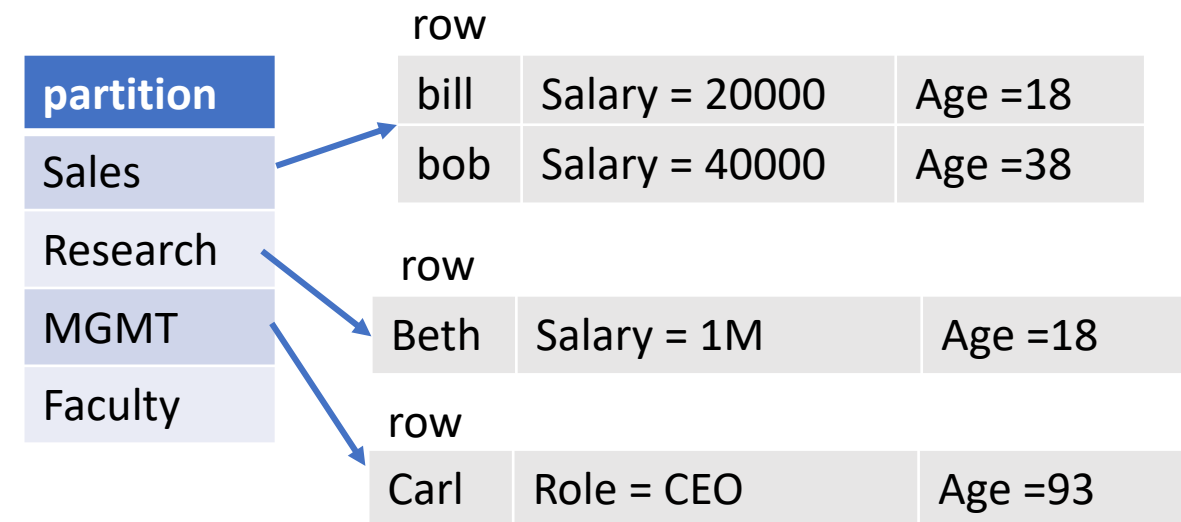
# NoSQL Storage Systems

- Main Concepts
  - Designed for massive scale
    - Distributed over many storage nodes
  - Support some SQL operations (not joins)
  - May be only eventually consistent
  - Different types
    - Key-value
    - Column oriented
    - Document style
    - Graph

Relational table

Name	Job	Salary	age
Bill	Sales	\$20000	18
Beth	Research	\$1m	35
Carl	CEO	0	93
Jill	Prof	\$100000	24

NoSQL Key-value system

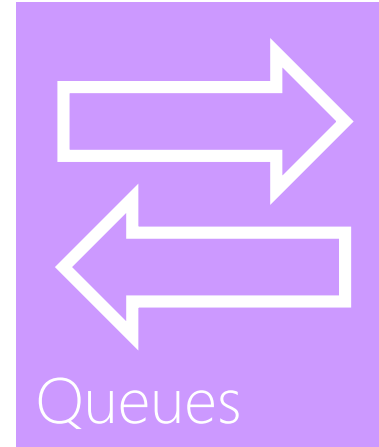


Key-value NoSQL examples *similar* to  
Amazon AWS DynamoDB  
Azure Tables  
Google BigTable and Cloud DataStore

# Azure Storage



Storage for any type of data, analogous to files in a file system, with individual blobs storing up to 4.75 TB of data



Reliable messaging for workflow processing and for communication between applications or application components

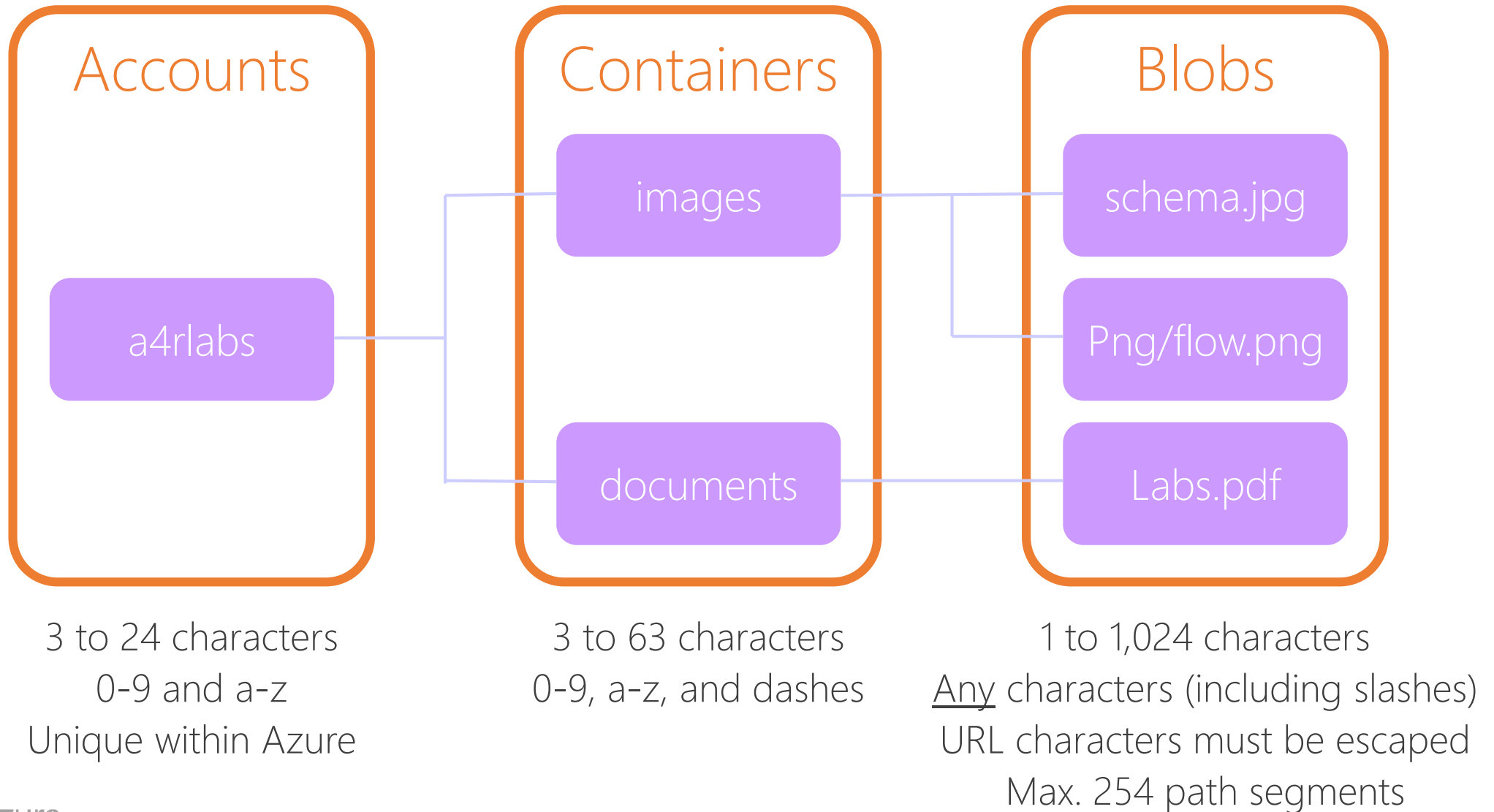


NoSQL storage of semi-structured data for rapid development and fast access to large quantities of data



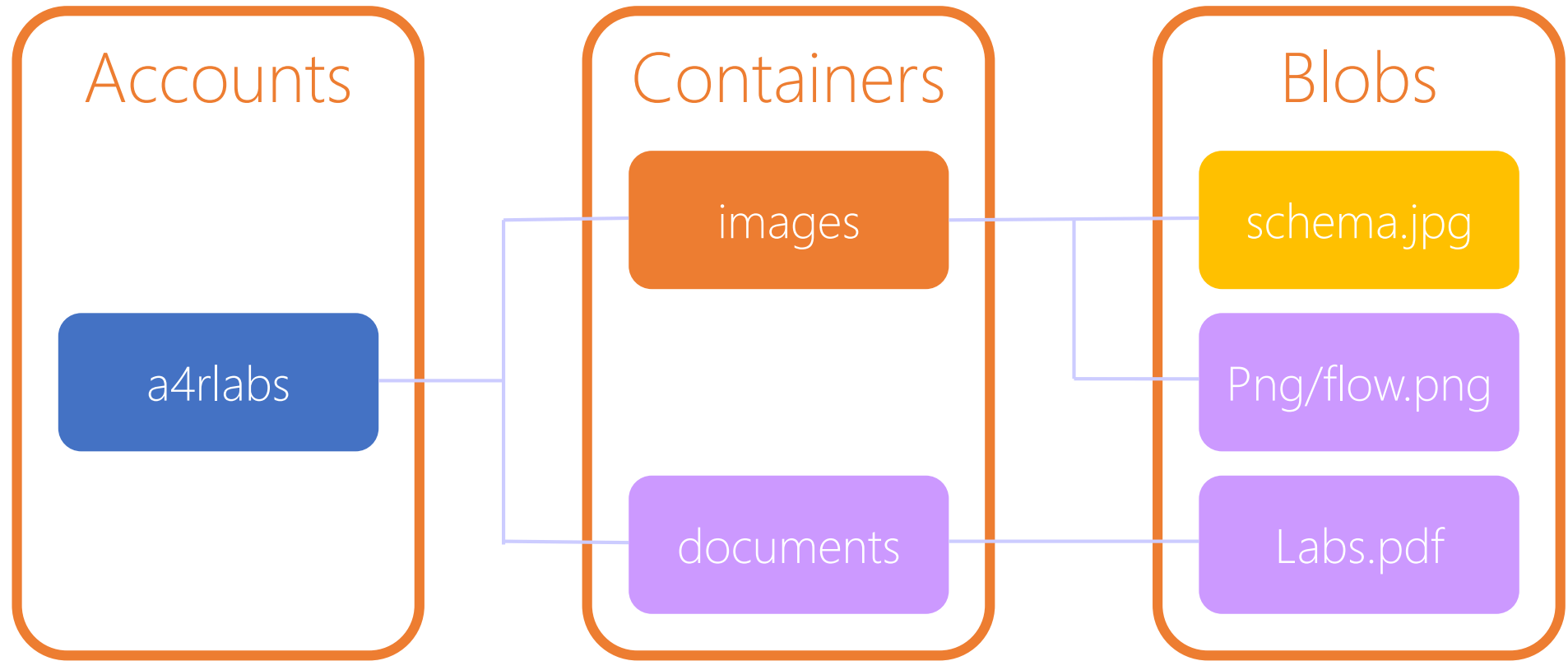
File sharing using Server Message Block (SMB) protocol

# Blob Storage





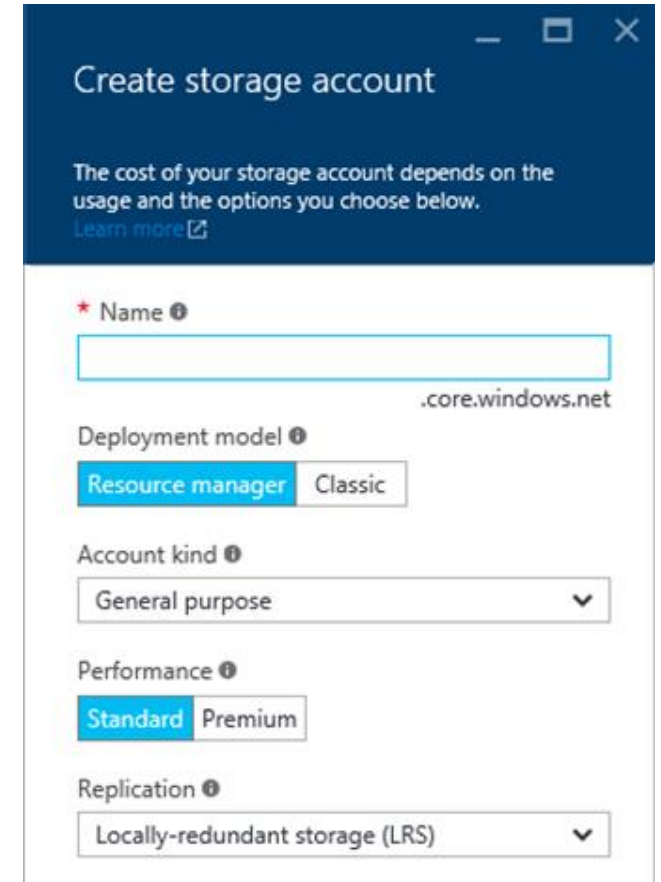
# Blob URLs



<https://a4rlabs.blob.core.windows.net/images/schema.jpg>

# Storage Accounts

- Up to 500 TB of data per account
- Maximum of 200 storage accounts per subscription
- Two types of accounts
  - "General purpose" and "Blob storage"
- Four types of replication
- Support optional 256-bit AES encryption for "data at rest"



The screenshot shows the 'Create storage account' form in the Microsoft Azure portal. The form is titled 'Create storage account' and includes a sub-header: 'The cost of your storage account depends on the usage and the options you choose below. [Learn more](#)'. The form contains several fields and options:

- Name:** A text input field with a red asterisk and an information icon. The text '.core.windows.net' is visible to the right of the input field.
- Deployment model:** Two buttons: 'Resource manager' (selected) and 'Classic'.
- Account kind:** A dropdown menu with 'General purpose' selected.
- Performance:** Two buttons: 'Standard' (selected) and 'Premium'.
- Replication:** A dropdown menu with 'Locally-redundant storage (LRS)' selected.







# Access Keys

- Access to storage by non-account-owners relies on access keys
- Keys should be "rolled" periodically for security
- Keys can be used to generate shared-access signatures (SAS) for secure and restricted access

Use access keys to authenticate your applications when making requests to this Azure storage account. Store your access keys securely - for example, using Azure Key Vault - and don't share them. We recommend regenerating your access keys regularly. You are provided two access keys so that you can maintain connections using one key while regenerating the other.

When you regenerate your access keys, you must update any Azure resources and applications that access this storage account to use the new keys. This action will not interrupt access to disks from your virtual machines. [Learn more](#)

Storage account name

NAME	KEY			
key1	EzUn/JGjJrWii7m+qiPHcvPFGV34ZI6vkwRe+QgOt1E8YGtuTkV			
key2	WTH0W5PHIQLhKGwCC98klhDVcthZH/9vOjwD6B5WHvelsz			

# Get the container for the demo

- Make sure you have docker installed on your machine
- Download Docker for your pc or mac
  - <https://docs.docker.com/engine/installation/>
- Then do

```
docker run -i -t -p 8888:8888 dbgannon/tutorial
```
- This will take a while
- When it is up go to <https://localhost:8888>
  - You will need to add security exceptions in the browser. It is safe.
  - Password is “tutorial”
  - Open azure.ipynb in Jupyter
- Or, if using a different jupyter, download <https://SciengCloud.github.io/azure.ipynb>
- Using the azure portal create a storage account and have the key ready

# Azure Solution – now create table and blob container

```
import csv
import sys
import azure.storage
from azure.storage.table import TableService, Entity
from azure.storage.blob import BlockBlobService
from azure.storage.blob import PublicAccess

block_blob_service = BlockBlobService(account_name='tutorial',
                                       account_key='biglongaccesskey')

block_blob_service.create_container('datacont',
                                   public_access=PublicAccess.Container)

table_service = TableService(account_name='tutorial',
                              account_key='samebiglongkey')

if table_service.create_table('DataTable'):
    print "table created"
else:
    print "table already there"
```

# Azure Solution

- Assume data objects are stored in files in /home/me/ and there is a CSV file “thedata” with rows
  - Experiment name, item id, date, filename, comment string

```
with open('/datadir/experiments.csv', 'rb') as csvfile:
    csvf = csv.reader(csvfile, delimiter=',', quotechar='|')
    for item in csvf:
        print item
        block_blob_service.create_blob_from_path(
            'datacont', item[3],
            "/datadir/"+item[3]
        )
    url = "https://"+account+".blob.core.windows.net/datacont/"+item[3]
    metadata_item = {'PartitionKey': item[0], 'RowKey': item[1],
                     'description': item[4], 'date': item[2], 'url':url}
    table_service.insert_entity('DataTable', metadata_item)
```

# Use Azure Storage Explorer to inspect the table

tutorial x

Storage Account

tutorial

**DataTable** table (5 entities) as of 3/6/2017 4:50:33 PM

Refresh New Delete

Refresh Select All Clear All Query Filter Upload Download View New Copy Delete

Blob Containers (0)  
Queues (0)  
Tables (1)  
DataTable

PartitionKey	RowKey	date	url	description
experiment1	id1	12/1/2016	<a href="https://tutorial.blob.core.windows.net/datacont/a.jpg">https://tutorial.blob.core.windows.net/datacont/a.jpg</a>	"here is a view of what is to be"
experiment1	id2	12/3/2016	<a href="https://tutorial.blob.core.windows.net/datacont/b.jpg">https://tutorial.blob.core.windows.net/datacont/b.jpg</a>	"map reduce picture"
experiment1	id5	12/6/2016	<a href="https://tutorial.blob.core.windows.net/datacont/e.jpg">https://tutorial.blob.core.windows.net/datacont/e.jpg</a>	"bio samples"
experiment2	id3	12/4/2016	<a href="https://tutorial.blob.core.windows.net/datacont/c.jpg">https://tutorial.blob.core.windows.net/datacont/c.jpg</a>	"sample notebook"
experiment3	id4	12/5/2016	<a href="https://tutorial.blob.core.windows.net/datacont/d.jpg">https://tutorial.blob.core.windows.net/datacont/d.jpg</a>	"workers"

# Time for Exercise 1.

- Look for the file Exercise1.pdf and go through it. It will introduce you to the azure portal and the simple task of creating a storage account and uploading a file and viewing it with the azure storage explorer.
- If you are done with that and you are familiar with Docker, Jupyter and Python then move on to Exercise1b.pdf. That will show you how you can manage storage of blobs and tables directly from a python program without using the portal. If you don't have docker installed it may take more time than you have right now.



Next VMs and Containers