

Homework Assignment 1 for CSCI-GA-2433 Database System

Yue Zhou

June 10, 2021

Abstract

In this homework assignment, we are discussing two different database systems, and how to migrate data from a CSV file in local repository to them.

1 Microsoft's Azure

1.1 Introduction

Microsoft Azure is a cloud computing service created by Microsoft for building, testing, deploying, and managing applications and services through Microsoft-managed data center. In summary, the types of services from Azure can be categorized as computer services: identity, mobile services, storage services, data management, messaging, media services, CDN, developer, management, Azure AI, Azure block-chain workbench, functions, and internet of things(IoT). [5] Under the category of storage services, there are four types of storage in Azure: file, blob, queue, table.[6] The focus from this report is blob storage type.

2 Blob container and data table

2.1 Addressing the Problems

In real life tasks, for example, we want to immigrate the local CSV(comma-separated values) files to the cloud storage. What should we do?

Let us start from a simple hypothesis and proposing few questions:

Assuming there is a CSV(comma-separated values) file on the local repository, What are the capacities of the cloud data storage(which is Microsoft's Azure blob container in this case)?

What are the internal logic and process during the migration of the CSV file from the local to the cloud?

What are the specific operations or commands for a data scientist to use as the

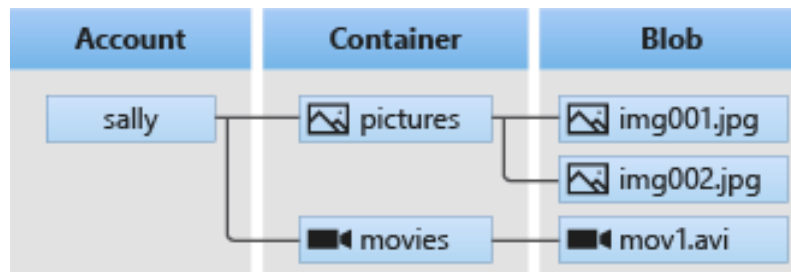
tutorial?

Please allow me answer these questions one by one in the following procedures.

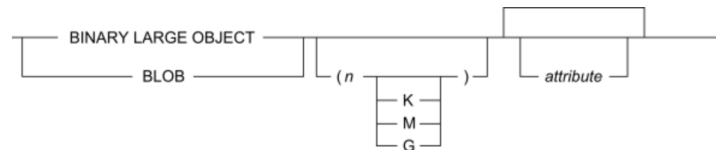
2.2 Capability of Blob Storage

The blob storage is a feature service in Azure to store unstructured data in Microsoft's cloud platform. The concrete format of blob storage is each "blob container" under the user's Microsoft Azure account.[2] To demonstrate the relation among the account, blob container, and blobs, we have:

- The storage account
- A blob container in the storage account
- Blobs in a blob container



From the example in the graph above, the stored files are picture and video files. Indeed, the speciality of Blob storage is to store large files such as pictures and videos, due to the data type of blob. "Blob" stands for "Binary Large Object" and is used for storing information in databases. Comparing to integer, floating point number, character, and string, blob is also a data type that can store binary data.[7] Moreover, "Binary Large Object(BLOB)" means it refers to an object. Unlike a string or character data type, the object data type only contains a pointer or reference to the value of the object. Due to this fact, BLOB is capable to hold data in large size, from documents to images to videos.[4] In addition, blob data type represents a large binary string of raw bytes. The blob(binary large object) column is the place to store binary objects, such as images, videos, and documents. To demonstrate this, the graph down below shows the inner structure of the blob data type. The number n represents the number of bytes allocated for the blob column. The number K, M, and G represent the corresponding kilobytes(KB), megabytes(MB), and gigabytes(GB).[8]



In summary, blob storage is designed for:

- Serving images or documents directly to a browser
- Storing files for distributed access
- Streaming video and audio
- Writing to log files
- Storing data for backup and restore, disaster recovery, and archiving
- Storing data for analysis by an on-premises or Azure-hosted service.

2.3 Breaking down the CSV file

The fact is that the CSV files contain information in different formats. Each row in CSV file is composed of variety of data types, such as strings, characters, images, and url links. Furthermore, there are multiple rows and columns in each CSV file. Each row can be considered as an individual(entity), and each column is the property(attribute) of each individual. For example, a row in CSV file can represent a person, including all the information of the person. Columns correspond to the person's information, such as name, ID number, date of birth, position, age and so on. In some cases, there does not only exist information in string or character data type, but as well as in the format of images, such as jpg format. Hence, it is necessary to come up with a method to store all different types of data in both the local CSV file and the cloud storage. The blob container and data table from Microsoft's Azure offer such a solution in cloud storage.

2.4 Logic of Blob

As indicated in figure 1: Cloud, each CSV file in the local repository can be migrated to the cloud storage as a data table. Each jpg formatted image in my local repository can be migrated to the could storage as a blob. Each blob is stored in the blob container. After the local CSV file has been uploaded to the cloud storage, most of its information becomes a data table on the cloud. By saying "most", I am referring to all the information except the images and videos. Instead of storing the images in the data table, there are few steps of

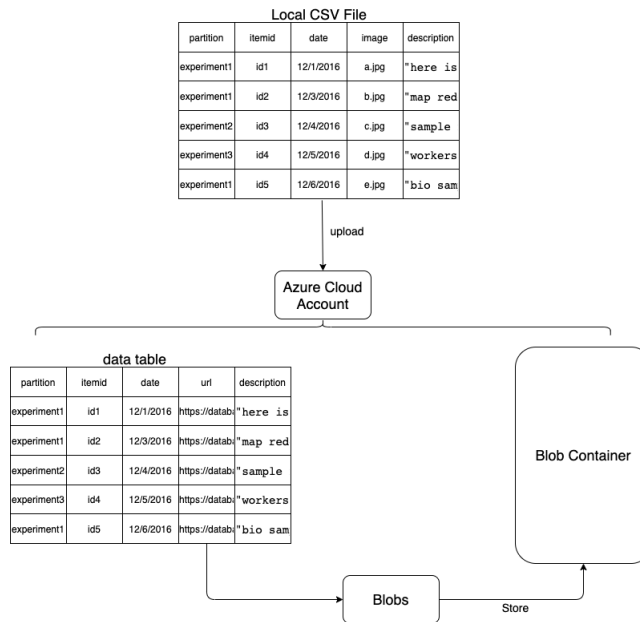
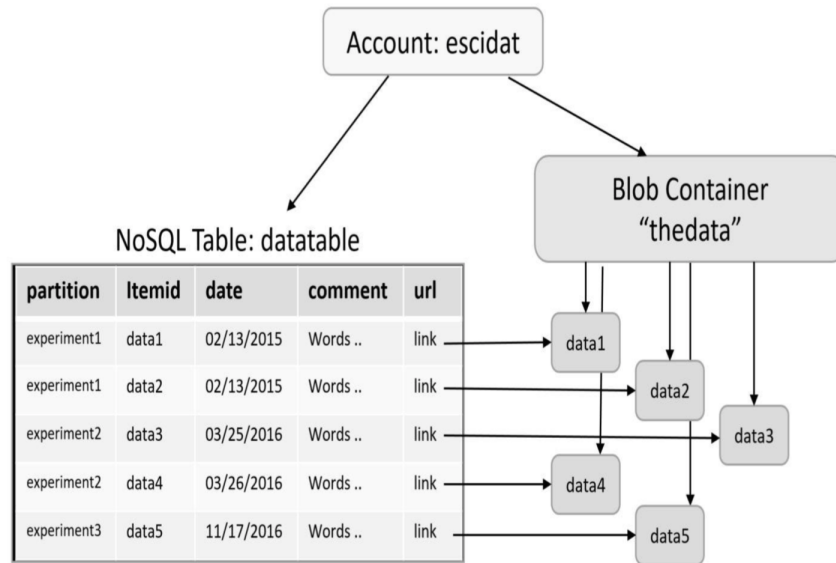


Figure 1: Cloud

operations. First, we insert url links into the data table, to indicate the location where each image is stored. Second, we convert each image into a blob. Third, we store each blob in the blob container. Hence, the original CSV file from the local repository is separated stored on the cloud as a data table and blobs in the blob container. The blobs and the data table is connected by url links.

Indeed, the graph above indicates the same ideal as it is in professor's assignment(shown below). It indicates my own interpretation and perspective.



2.5 Setups

Before we move onto the actual coding part, there are few steps to set up the environment for migrating data.

First, register an account in Microsoft's azure.

Second, establish a storage account, where I can find the storage key.

Third, in local terminal, install the tutorial written in python from dbganon/tutorial in docker, by:

- `docker pull dbganon/tutorial`
- `docker run -i -t -p 8888:8888 dbganon/tutorial`

Fourth, open the jupyter notebook in browser by `http://localhost:8888`. The password is "tutorial".

Fifth, select `azure.ipynb` to open.

Thereby, the tutorial jupyter notebook has been opened.

2.6 How to migrate data from local repository to the cloud

Continuing from the last section, in python jupyter notebook, we first open the CSV file.

```
with open('/datadir/experiments.csv', 'rb') as csvfile:
    csvf = csv.reader(csvfile, delimiter=',', quotechar='|')
```

Second, we read the CSV file row by row.

```
for item in csvf:
    print item
```

Third, we create each blob based on each picture in each row of the local CSV file.

```
block_blob_service.create_blob_from_path(
    'datacont', item[3],
    "/datadir/"+item[3]
)
```

Fourth, we create each url link based on our azure account and each picture in the local CSV file.

```
url = "https://"+account+".blob.core.windows.net/datacont/"+item[3]
```

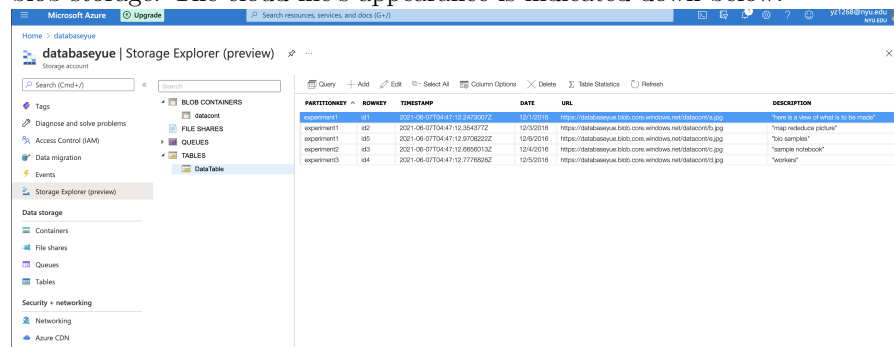
Fifth, we create the data table by all the rest information from the local CSV file and the url links that we have just generated.

```
metadata_item = {'PartitionKey': item[0], 'RowKey': item[1],
                 'description': item[4], 'date': item[2], 'url': url}
table_service.insert_entity('DataTable', metadata_item)
```

Hence, we have shown the functionality of each part of the code in the tutorial notebook. One last little thing to mention, in order to upload another csv files to another cloud account, we have to modify the local root location of CSV file and the cloud account information in order to upload.

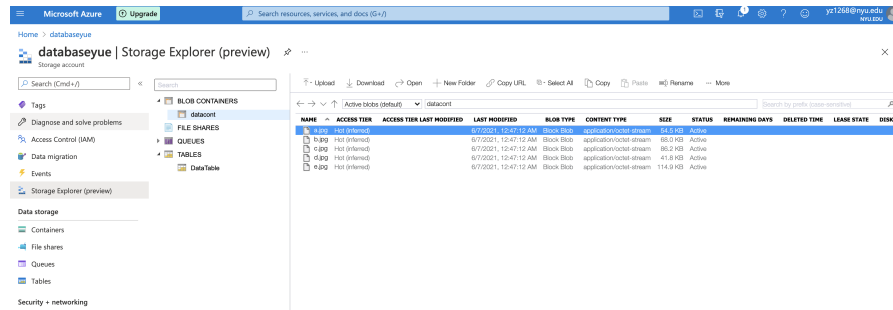
2.6.1 Result

As a result, the local CSV file has been successfully uploaded to Azure's cloud blob storage. The cloud file's appearance is indicated down below:



The screenshot shows the Microsoft Azure Storage Explorer interface. The left sidebar displays the navigation pane with 'Storage Explorer (preview)' selected. The main pane shows a table with the following columns: PARTITIONKEY, ROWKEY, TIMESTAMP, DATE, URL, and DESCRIPTION. The table contains five rows of data, each representing an experiment.

PARTITIONKEY	ROWKEY	TIMESTAMP	DATE	URL	DESCRIPTION
experiment1	g1	2021-06-07T04:47:12.247300Z	12/13/2016	https://databaseyue.blob.core.windows.net/datacont/g1.jpg	"There is a view of what is to be made"
experiment1	g2	2021-06-07T04:47:12.354372Z	12/3/2016	https://databaseyue.blob.core.windows.net/datacont/g2.jpg	"map-reduce picture"
experiment1	g3	2021-06-07T04:47:12.376522Z	12/5/2016	https://databaseyue.blob.core.windows.net/datacont/g3.jpg	"On sample"
experiment2	g4	2021-06-07T04:47:12.665015Z	12/4/2016	https://databaseyue.blob.core.windows.net/datacont/g4.jpg	"sample notebook"
experiment3	g4	2021-06-07T04:47:12.777888Z	12/5/2016	https://databaseyue.blob.core.windows.net/datacont/g4.jpg	"secretary"



The first graph above is data table, and the second graph above is blob container(with blobs inside).

3 XML

3.1 Definition of XML

XML stands for extensible markup language. XML is one type of languages to write and modify hypertext documents which are the method of representing the content and formatting the web pages. According to Elmasri, the definition of XML is “a new language -namely, XML(Extensible Markup Language) -has emerged as the standard for structuring and exchanging data over the Web in text files.” [1] Comparing to the previous languages,such as HTML,XML is designed for carrying data; the advantage of XML is providing structure and meaning of data in the Web pages instead of exclusively providing the format in display. On the other hand, HTML is designed for displaying data, which is how data looks.

3.2 The advantage and features of XML

Since XML stores data in a plain text format, XML makes the data compatible, transferable, and accessible.[3] Hence, XML simplifies data in such ways:

- XML simplifies data sharing
- XML simplifies data transport
- XML simplifies data availability
- XML simplifies platform changes

3.3 The structure of XML

From the logic perspective of the XML structure, XML follows the tree structure. There exists a root. Under the root, there are children. Each children can have sub-children.

```

<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>

```

From the syntax perspective of the XML structure, the essential components in a XML document are character, processor and application, markup and content, tag, element, attribute and XML declaration.[9]

The basic component to form a XML document is a string of characters. XML document allows almost every legal Unicode character to appear.

The processor's duty is to parse the content in a XML file and pass the parsed structured information to an application.

From the perspective of the markup, the language in XML can be divided into markup and content. The markup has two formats. It either begins with < and end with >, or begins with & and end with ;.

The tag consists of markup beginning with < and ending with >. There are three types of tags, start-tag as < *section* >, end-tag as < /*section* >, and empty-element tag as < *line – break*/ >.

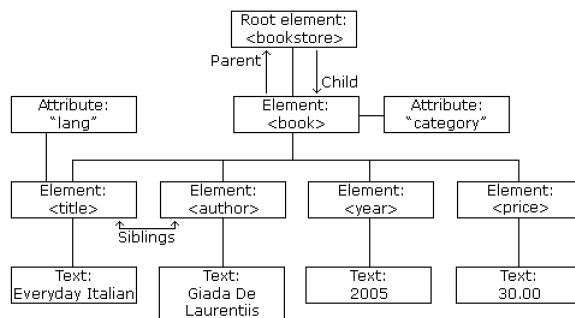
An element is the document content in between a start-tag and an end-tag. Sometimes, an element is solely an empty-element tag.

The attribute is a name-value pair existing inside of a start-tag or an empty-element tag. For example, the start-tag < *imgsrc* = "*madonna.jpg*" *alt* = "*Madonna*" / > has two attributes: src and alt. The value of src is *madonna.jpg* and the value of alt is *Madonna*.

The XML declaration is in the format of <?...? >. It is common to see such a declaration as the first line in a XML file. XML declaration plays the role of description.

Hence, we have introduced all the basic components in the XML language.

Let us start with an example, in the left picture down below, the logic in a tree structure has been indicated. Its corresponding concrete XML content is shown as following.



```

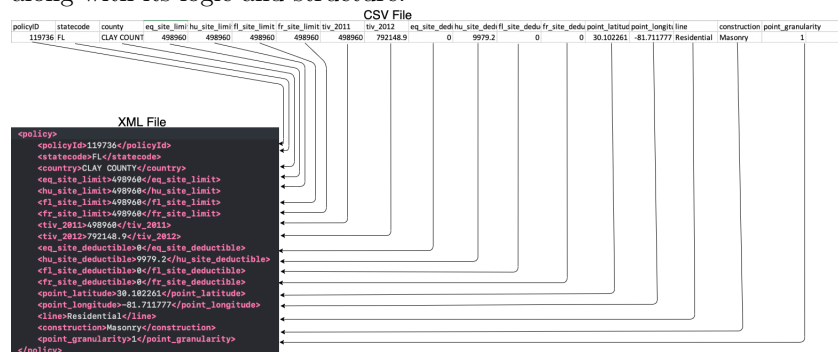
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>

```

In this specific example, the bookstore is the root element. This root element has three children: cooking, children, and web. Each child also has four attributes: title, author, year, price.[3] From the XML language on the right side, it contains the specific information of each attribute in each entity. Each piece of specific information is lying between a start-tag and an end-tag.

3.4 Migrating data from CSV files to XML files

When the data in the CSV file is migrated to XML file, each attribute in each entity from the CSV file becomes each attribute between the start-tag and end-tag on each line of code in the XML file. All the attributes from a single entity is also separated from other entities by a start-tag and end-tag. This example below shows how a single entity is migrated from the CSV file to the XML file, along with its logic and structure.



For the coding part in python, we follow the procedures:
First, open the CSV file:

```
f = open('/Users/zhousq/Documents/Semesters/2021_Summer/Database/Homeworks/HW1/2nd_example/sample.csv')
csv_f = csv.reader(f)
```

Second, create an empty list and store each row from CSV file to it:

```
data = []
for row in csv_f:
    data.append(row)
f.close()
```

Third, create a function in which record each element from the list we just created and put each element from the list into XML:

```
def convert_row(row):
    return """<policy>
    <policyId>%s</policyId>
    <statecode>%s</statecode>
    <country>%s</country>
    <eq_site_limit>%s</eq_site_limit>
    <hu_site_limit>%s</hu_site_limit>
    <fl_site_limit>%s</fl_site_limit>
    <fr_site_limit>%s</fr_site_limit>
    <tiv_2011>%s</tiv_2011>
    <tiv_2012>%s</tiv_2012>
    <eq_site_deductible>%s</eq_site_deductible>
    <hu_site_deductible>%s</hu_site_deductible>
    <fl_site_deductible>%s</fl_site_deductible>
    <fr_site_deductible>%s</fr_site_deductible>
    <point_latitude>%s</point_latitude>
    <point_longitude>%s</point_longitude>
    <line>%s</line>
    <construction>%s</construction>
    <point_granularity>%s</point_granularity>
</policy>""" % (row[0], row[1], row[2], row[3], row[4],
```

Fourth, store the XML to a local repository:

```
f = open('/Users/zhousq/Documents/Semesters/2021_Summer/Database/Homeworks/HW1/2nd_example/sample.csv')
csv_f = csv.reader(f)
```

Hence, we have migrated the data from a CSV file to a XML file. All the codes are included in the assignment zip file.

4 Conclusion

We have experienced the entire processes of migrating a CSV file to two different database systems: Microsoft's azure BLOB storage and XML. BLOB storage has its unique advantage in cloud storage and storing data which has large sizes. XML has its advantage in readability, compatibility and extendibility. What database to use? It depends on the goal of one's task and what the person wants to achieve.

References

- [1] Ramez Elmasri and Sham Navathe. *Fundamentals of Database Systems*. <https://www.auhd.site/upfiles/elibrary/Aza12020-01-22-12-28-11-76901.pdf>. 1989.

- [2] *Introduction to Azure Blob storage*. <https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction>. 2021.
- [3] *Introduction to XML*. https://www.w3schools.com/xml/xml_what_is.asp.
- [4] *SQL Data Types: BINARY LARGE OBJECT*. <https://study.com/academy/lesson/sql-data-types-binary-large-object.html>.
- [5] Amitabh Srivastava. *Introducing Windows Azure*. <https://web.archive.org/web/20100514093158/http://blogs.msdn.com/windowsazure/archive/2008/10/27/introducing-windows-azure.aspx>. 2010.
- [6] Minette Steynberg. *Different Azure Storage types (File, Blob, Queue and Table)*. <https://www.sqlshack.com/different-azure-storage-types-file-blob-queue-table/>. 2017.
- [7] techterms. <https://techterms.com/definition/blob>.
- [8] *teradata*. https://docs.teradata.com/r/~_sY_PYVxZzTnqKq45UXkQ/AH61TE0i89qTUht0Xa2_8g.
- [9] *XML*. <https://en.wikipedia.org/wiki/XML>.