

Read data from the transactional store

Note In this reading you can see the steps involved in the process of reading data from the transactional store.

There may be scenarios where you may need to read data from the Azure Cosmos DB transactional store, for example, where you cannot tolerate the near-real-time latency constraints of the analytical store to surface specific information, or where schema inference rules have made data unavailable in the analytical store.

When connecting to the transactional store, any queries you run will consume Azure Cosmos DB request units and could impact workloads running against the transactional store. For most analytical type queries from Spark, the analytical store is going to be your best choice, from both a performance and cost perspective.

Now that you are familiar with reading from the analytical store into a DataFrame; reading from the transactional store will be simple, as it requires only a single change to the query. For the transactional store, the format parameter is specified as **cosmos.oltp** instead of the cosmos.olap used to access the analytical store.

1. Paste the code below into a **new cell (A)**, and click the **run cell** button.

```
dfCustomer = spark.read\  
    .format("cosmos.oltp")\  
    .option("spark.synapse.linkedService", "AdventureWorksSQL")\  
    .option("spark.cosmos.container", "Customer")\  
    .load()  
  
display(dfCustomer.limit(10))
```

1
2
3
4
5
6
7

Microsoft Azure | Synapse Analytics | synapseinkadventureworks

Synapse live | Validate all | Publish all

Cosmos DB Notebook

Cell | Run all | Publish | Attach to: adventurespark | Language: PySpark (Python) | Preview Features

Ready

```

1 dfCustomer = spark.read\
2   .format("cosmos.oltp")\
3   .option("spark.synapse.linkedService", "AdventureWorksSQL")\
4   .option("spark.cosmos.container", "Customer")\
5   .load()
6
7 display(dfCustomer.limit(10))

```

Command executed in 4s 73ms

Job execution Succeeded Spark 2 executors 16 cores

View: table | chart

_attachments	_etag	_rid	_self	_ts	address	creationDate	emailAddress	firstName	id
attachments/	"56000a83-0000-...	mP07Aluc-AgBA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2011-09-07T00:0...	samantha6@adv...	Samantha	A8D0C3B8-50CC...
attachments/	"56000b83-0000-...	mP07Aluc-AgCA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2013-01-06T00:0...	samuel21@adve...	Samuel	A7C84443-A5AD...
attachments/	"56000c83-0000-...	mP07Aluc-AgDA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2014-01-19T00:0...	edward67@adve...	Edward	A695E5F8-DE3A...
attachments/	"56000d83-0000-...	mP07Aluc-AgEA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2014-01-23T00:0...	henry4@adventu...	Henry	A59053FD-3D9A...
attachments/	"56000e83-0000-...	mP07Aluc-AgFA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2013-11-22T00:0...	jesse43@advent...	Jesse	A481FE7A-28AD...
attachments/	"56000f83-0000-...	mP07Aluc-AgGA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2012-02-10T00:0...	robert82@adven...	Robert	A3803AB7-9527...
attachments/	"56001083-0000-...	mP07Aluc-AgHA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2013-10-23T00:0...	jillian5@adventu...	Jillian	A236A6BC-4E02...
attachments/	"56001183-0000-...	mP07Aluc-AgIA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2014-05-03T00:0...	tina11@adventur...	Tina	A102AE79-95B8...
attachments/	"56001283-0000-...	mP07Aluc-AgJAA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2013-07-11T00:0...	kari7@adventure...	Kari	9F7FA5B2-54FB...
attachments/	"56001383-0000-...	mP07Aluc-AgKA...	dfs/mP07AA==f/...	1607057561	▶ "[addressLine1]:"	2013-11-24T00:0...	dana17@advent...	Dana	9E35C07B-D841...

Connecting to the transactional store

A similar approach applies to creating a Spark SQL table to access the transactional store, the USING clause is changed to specify cosmos.oltp as the source.

2. Paste the code below into a **new cell (B)**, and click the **run cell** button.

1
2
3
4
5

```

%%sql
CREATE TABLE CustomersOLTP USING cosmos.oltp OPTIONS (
    spark.synapse.linkedService 'AdventureWorksSQL',
    spark.cosmos.container 'Customer'
)

```

To validate the table is working as expected:

3. Paste the code below into a **new cell (C)**, and click the **run cell** button.

1
2
3

```

%%sql
SELECT * FROM CustomersOLTP
LIMIT 10

```

You can see that the result is the content of the Azure Cosmos DB transactional store.

Microsoft Azure

Synapse Analytics

synapselinkadventureworks

Search

Synapse live

Validate all

Publish all

Cosmos DB Notebook

Cell

Run all

Publish

Attach to adventurespark

Language PySpark (Python)

Preview Features

Ready

1 %sql

2 CREATE TABLE Customers0LTP USING cosmos.oltp OPTIONS (

3 spark.synapse.LinkedService 'AdventureWorksSQL',

4 spark.cosmos.container 'Customer'

5)

Command executed in 6s 101ms

Job execution Succeeded Spark 2 executors 16 cores

View in monitoring

Open Spark UI

View

Table

Chart

No Data Available!

+

141

1 %sql

2 SELECT * FROM Customers0LTP

3 LIMIT 10

Command executed in 2s 31ms

Job execution Succeeded Spark 2 executors 16 cores

View in monitoring

Open Spark UI

View

Table

Chart

_attachments	_etag	_rid	_self	_ts	address
attachments/	"8d008c14-0000-0800-0000-5fcb...	mP07Aluc-AgBAAAAAAAAA==	dbb/mP07AA==/colls/mP07Aluc...	1607143166	▶ {"schema":{"name":"addressLine1
attachments/	"56000b83-0000-0800-0000-5fc...	mP07Aluc-AgCAAAAAAAAAA==	dbb/mP07AA==/colls/mP07Aluc...	1607057561	▶ {"schema":{"name":"addressLine1
attachments/	"56000c83-0000-0800-0000-5fc9...	mP07Aluc-AgDAAAAAAAAA==	dbb/mP07AA==/colls/mP07Aluc...	1607057561	▶ {"schema":{"name":"addressLine1