Index the Data Lake storage with Hyperspace

When loading data from Azure Data Lake Gen 2, searching in the data in one of the most resource consuming operations. Hyperspace introduces the ability for Apache Spark users to create indexes on their datasets, such as CSV, JSON, and Parquet, and use them for potential query and workload acceleration.

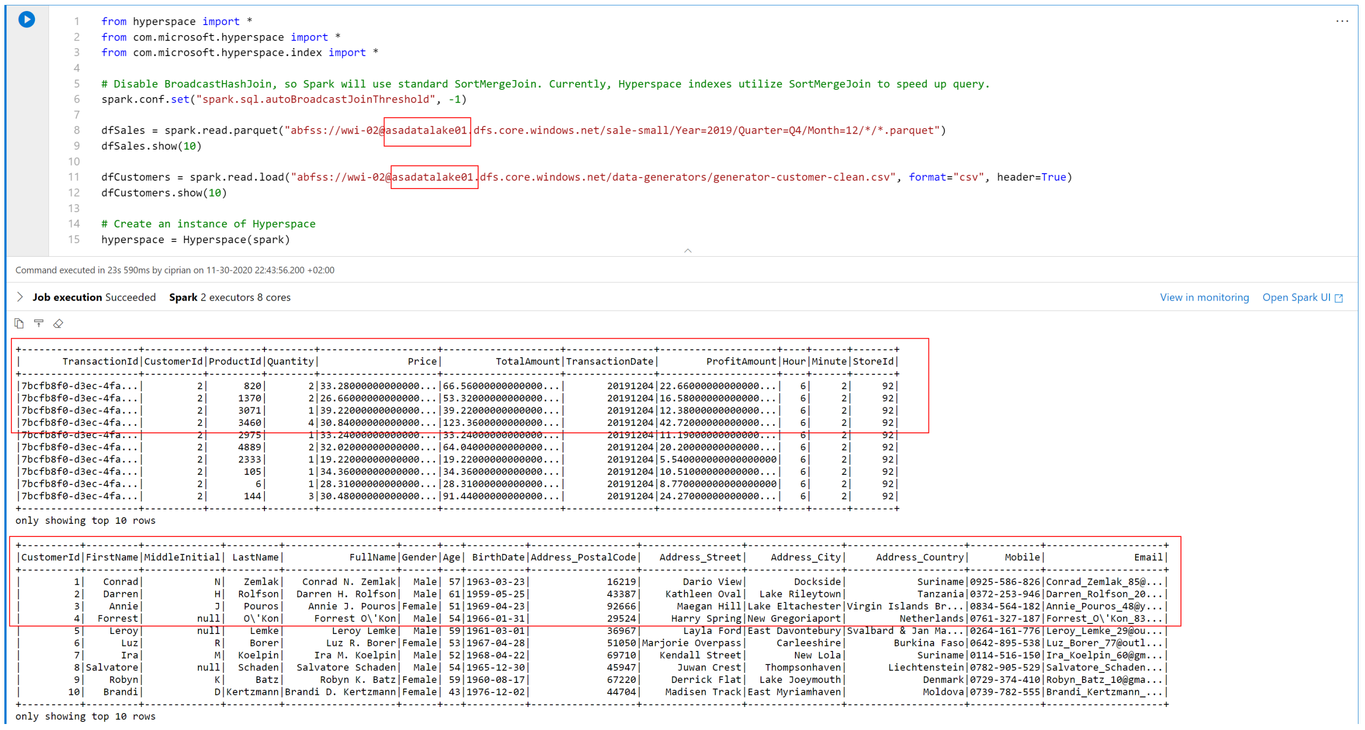
Hyperspace lets you create indexes on records scanned from persisted data files. After they're successfully created, an entry that corresponds to the index is added to the Hyperspace's metadata. This metadata is later used by Apache Spark's optimizer during query processing to find and use proper indexes. If the underlying data changes, you can refresh an existing index to capture that.

Also, Hyperspace allows users to compare their original plan versus the updated index-dependent plan before running their query.

Add a new cell to your notebook with the following code (remember to replace `<unique_suffix>` with the value you specified during the Synapse Analytics workspace deployment):

```
from hyperspace import *
from com.microsoft.hyperspace import *
from com.microsoft.hyperspace.index import *

# Disable BroadcastHashJoin, so Spark will use standard SortMergeJoin. Currently,
Hyperspace indexes utilize SortMergeJoin to speed up query.
spark.conf.set("spark.sql.autoBroadcastJoinThreshold", -1)

dfSales = spark.read.parquet("abfss://wwi-
02@asagadatalake<unique_suffix>.dfs.core.windows.net/sale-
small/Year=2019/Quarter=Q4/Month=12/*/*.parquet")
dfSales.show(10)

dfCustomers = spark.read.load("abfss://wwi-
02@asagadatalake<unique_suffix>.dfs.core.windows.net/data-generators/generator-
customer-clean.csv", format="csv", header=True)
dfCustomers.show(10)

# Create an instance of Hyperspace
hyperspace = Hyperspace(spark)
```

Run the new cell. It will load the two DataFrames with data from the data lake and initalize Hyperspace.



```python
from hyperspace import *
from com.microsoft.hyperspace import *
from com.microsoft.hyperspace.index import *

# Disable BroadcastHashJoin, so Spark will use standard SortMergeJoin. Currently, Hyperspace indexes utilize SortMergeJoin to speed up query.
spark.conf.set("spark.sql.autoBroadcastJoinThreshold", -1)

dfSales = spark.read.parquet("abfss://wwi-02@asadatalake01.dfs.core.windows.net/sale-small/Year=2019/Quarter=Q4/Month=12/*/*.parquet")
dfSales.show(10)

dfCustomers = spark.read.load("abfss://wwi-02@asadatalake01.dfs.core.windows.net/data-generators/generator-customer-clean.csv", format="csv", header=True)
dfCustomers.show(10)

# Create an instance of Hyperspace
hyperspace = Hyperspace(spark)
```

Command executed in 23s 590ms by ciprian on 11-30-2020 22:43:56.200 +02:00

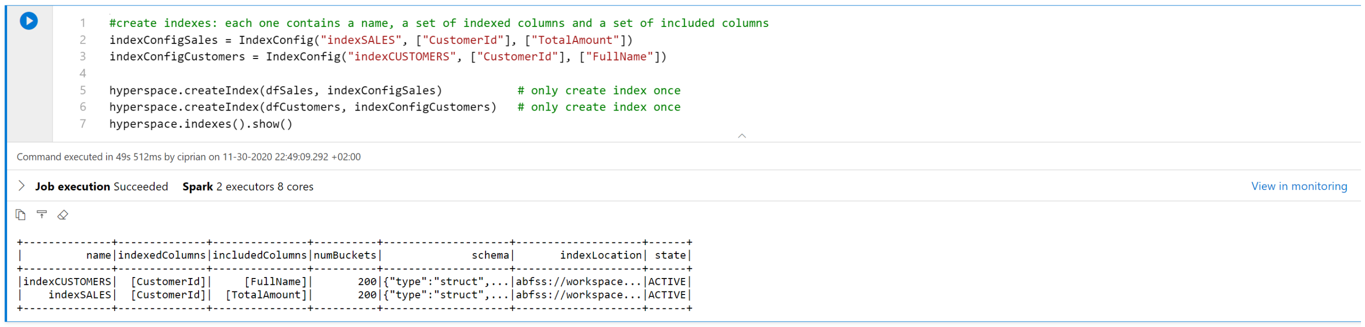
Job execution Succeeded **Spark** 2 executors 8 cores View in monitoring Open Spark UI

```
+--------------------+----------+---------+--------+--------------------+--------------------+---------------+--------------------+----+------+-------+
|       TransactionId|CustomerId|ProductId|Quantity|               Price|         TotalAmount|TransactionDate|        ProfitAmount|Hour|Minute|StoreId|
+--------------------+----------+---------+--------+--------------------+--------------------+---------------+--------------------+----+------+-------+
|7bcfb8f0-d3ec-4fa...|         2|      820|       2|33.28000000000000...|66.56000000000000...|       20191204|22.66000000000000...|   6|     2|     92|
|7bcfb8f0-d3ec-4fa...|         2|     1370|       2|26.66000000000000...|53.32000000000000...|       20191204|16.58000000000000...|   6|     2|     92|
|7bcfb8f0-d3ec-4fa...|         2|     3071|       1|39.22000000000000...|39.22000000000000...|       20191204|12.38000000000000...|   6|     2|     92|
|7bcfb8f0-d3ec-4fa...|         2|     3460|       4|30.84000000000000...|123.3600000000000...|       20191204|42.72000000000000...|   6|     2|     92|
|7bcfb8f0-d3ec-4fa...|         2|     2975|       1|33.24000000000000...|33.24000000000000...|       20191204|11.19000000000000...|   6|     2|     92|
|7bcfb8f0-d3ec-4fa...|         2|     4889|       2|32.02000000000000...|64.04000000000000...|       20191204|20.20000000000000...|   6|     2|     92|
|7bcfb8f0-d3ec-4fa...|         2|     2333|       1|19.22000000000000...|19.22000000000000...|       20191204|5.540000000000000000|   6|     2|     92|
|7bcfb8f0-d3ec-4fa...|         2|      105|       1|34.36000000000000...|34.36000000000000...|       20191204|10.51000000000000...|   6|     2|     92|
|7bcfb8f0-d3ec-4fa...|         2|        6|       1|28.31000000000000...|28.31000000000000...|       20191204|8.770000000000000000|   6|     2|     92|
|7bcfb8f0-d3ec-4fa...|         2|      144|       3|30.48000000000000...|91.44000000000000...|       20191204|24.27000000000000...|   6|     2|     92|
+--------------------+----------+---------+--------+--------------------+--------------------+---------------+--------------------+----+------+-------+
only showing top 10 rows

+----------+---------+-------------+---------+-------------------+------+---+----------+------------------+-----------------+----------------+--------------------+------------+--------------------+
|CustomerId|FirstName|MiddleInitial| LastName|           FullName|Gender|Age| BirthDate|Address_PostalCode|   Address_Street|    Address_City|     Address_Country|      Mobile|               Email|
+----------+---------+-------------+---------+-------------------+------+---+----------+------------------+-----------------+----------------+--------------------+------------+--------------------+
|         1|   Conrad|            N|   Zemlak|   Conrad N. Zemlak|  Male| 57|1963-03-23|             16219|       Dario View|        Dockside|            Suriname|0925-586-826|Conrad_Zemlak_85@...|
|         2|   Darren|            H|  Rolfson|  Darren H. Rolfson|  Male| 61|1959-05-25|             43387|    Kathleen Oval|  Lake Rileytown|            Tanzania|0372-253-946|Darren_Rolfson_20...|
|         3|    Annie|            J|   Pouros|    Annie J. Pouros|Female| 51|1969-04-23|             92666|      Maegan Hill|Lake Eltachester|Virgin Islands Br...|0834-564-182|Annie_Pouros_48@y...|
|         4|  Forrest|         null|   O\'Kon|     Forrest O\'Kon|  Male| 54|1966-01-31|             29524|     Harry Spring|New Gregoriaport|         Netherlands|0761-327-187|Forrest_O\'Kon_83...|
|         5|    Leroy|         null|    Lemke|        Leroy Lemke|  Male| 59|1961-03-01|             36967|       Layla Ford|East Davontebury|Svalbard & Jan Ma...|0264-161-776|Leroy_Lemke_29@ou...|
|         6|      Luz|            R|    Borer|       Luz R. Borer|Female| 53|1967-04-28|             51050|Marjorie Overpass|     Carleeshire|        Burkina Faso|0642-895-538|Luz_Borer_77@outl...|
|         7|      Ira|            M|  Koelpin|     Ira M. Koelpin|  Male| 52|1968-04-22|             69710|   Kendall Street|        New Lola|            Suriname|0114-516-150|Ira_Koelpin_60@gm...|
|         8|Salvatore|         null|  Schaden|  Salvatore Schaden|  Male| 54|1965-12-30|             45947|      Juwan Crest|   Thompsonhaven|       Liechtenstein|0782-905-529|Salvatore_Schaden...|
|         9|    Robyn|            K|     Batz|      Robyn K. Batz|Female| 59|1960-08-17|             67220|     Derrick Flat|  Lake Joeymouth|             Denmark|0729-374-410|Robyn_Batz_10@gma...|
|        10|   Brandi|            D|Kertzmann|Brandi D. Kertzmann|Female| 43|1976-12-02|             44704|    Madisen Track|East Myriamhaven|             Moldova|0739-782-555|Brandi_Kertzmann_...|
+----------+---------+-------------+---------+-------------------+------+---+----------+------------------+-----------------+----------------+--------------------+------------+--------------------+
only showing top 10 rows
```

Add another new cell to your notebook with the following code:

```python
#create indexes: each one contains a name, a set of indexed columns and a set of
included columns
indexConfigSales = IndexConfig("indexSALES", ["CustomerId"], ["TotalAmount"])
indexConfigCustomers = IndexConfig("indexCUSTOMERS", ["CustomerId"], ["FullName"])

hyperspace.createIndex(dfSales, indexConfigSales)             # only create index
once
hyperspace.createIndex(dfCustomers, indexConfigCustomers)   # only create index
once
hyperspace.indexes().show()
```

Run the new cell. It will create two indexes and display their structure.



```python
#create indexes: each one contains a name, a set of indexed columns and a set of included columns
indexConfigSales = IndexConfig("indexSALES", ["CustomerId"], ["TotalAmount"])
indexConfigCustomers = IndexConfig("indexCUSTOMERS", ["CustomerId"], ["FullName"])

hyperspace.createIndex(dfSales, indexConfigSales)           # only create index once
hyperspace.createIndex(dfCustomers, indexConfigCustomers)   # only create index once
hyperspace.indexes().show()
```

Command executed in 49s 512ms by ciprian on 11-30-2020 22:49:09.292 +02:00

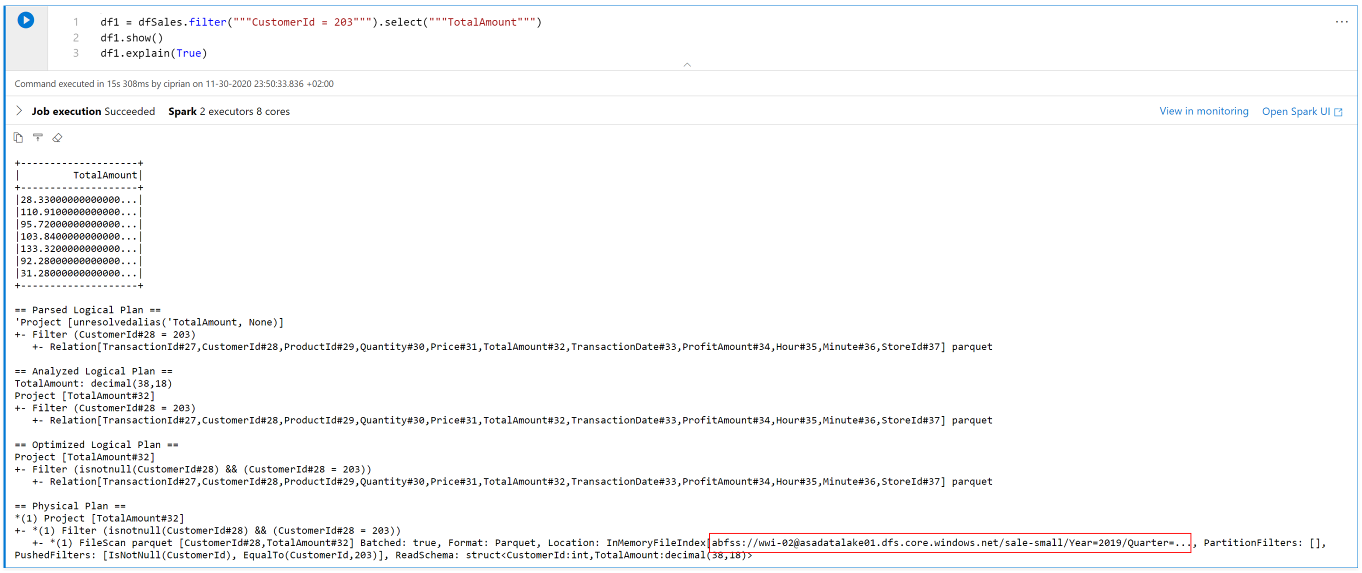
Job execution Succeeded **Spark** 2 executors 8 cores View in monitoring

```
+--------------+--------------+---------------+----------+--------------------+--------------------+------+
|          name|indexedColumns|includedColumns|numBuckets|              schema|       indexLocation| state|
+--------------+--------------+---------------+----------+--------------------+--------------------+------+
|indexCUSTOMERS|  [CustomerId]|     [FullName]|       200|{"type":"struct",...|abfss://workspace...|ACTIVE|
|    indexSALES|  [CustomerId]|  [TotalAmount]|       200|{"type":"struct",...|abfss://workspace...|ACTIVE|
+--------------+--------------+---------------+----------+--------------------+--------------------+------+
```

Add another new cell to your notebook with the following code:

```python
df1 = dfSales.filter("""CustomerId = 203""").select("""TotalAmount""")
df1.show()
df1.explain(True)
```

Run the new cell. The output will show that the physical execution plan is not taking into account any of the indexes (performs a file scan on the original data file).



```
df1 = dfSales.filter("""CustomerId = 203""").select("""TotalAmount""")
df1.show()
df1.explain(True)
```

Command executed in 15s 308ms by ciprian on 11-30-2020 23:50:33.836 +02:00

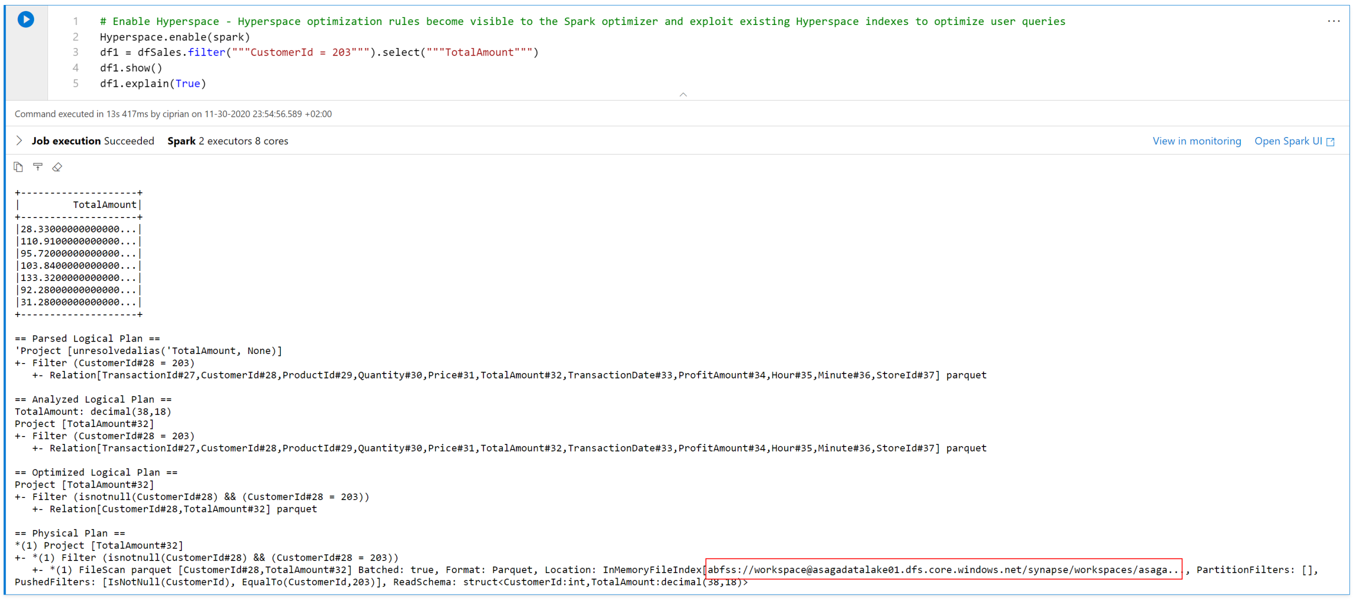
Job execution Succeeded   Spark 2 executors 8 cores   View in monitoring   Open Spark UI

```
+--------------------+
|         TotalAmount|
+--------------------+
|28.33000000000000...|
|110.9100000000000...|
|95.72000000000000...|
|103.8400000000000...|
|133.3200000000000...|
|92.28000000000000...|
|31.28000000000000...|
+--------------------+

== Parsed Logical Plan ==
'Project [unresolvedalias('TotalAmount, None)]
+- Filter (CustomerId#28 = 203)
   +- Relation[TransactionId#27,CustomerId#28,ProductId#29,Quantity#30,Price#31,TotalAmount#32,TransactionDate#33,ProfitAmount#34,Hour#35,Minute#36,StoreId#37] parquet

== Analyzed Logical Plan ==
TotalAmount: decimal(38,18)
Project [TotalAmount#32]
+- Filter (CustomerId#28 = 203)
   +- Relation[TransactionId#27,CustomerId#28,ProductId#29,Quantity#30,Price#31,TotalAmount#32,TransactionDate#33,ProfitAmount#34,Hour#35,Minute#36,StoreId#37] parquet

== Optimized Logical Plan ==
Project [TotalAmount#32]
+- Filter (isnotnull(CustomerId#28) && (CustomerId#28 = 203))
   +- Relation[TransactionId#27,CustomerId#28,ProductId#29,Quantity#30,Price#31,TotalAmount#32,TransactionDate#33,ProfitAmount#34,Hour#35,Minute#36,StoreId#37] parquet

== Physical Plan ==
*(1) Project [TotalAmount#32]
+- *(1) Filter (isnotnull(CustomerId#28) && (CustomerId#28 = 203))
   +- *(1) FileScan parquet [CustomerId#28,TotalAmount#32] Batched: true, Format: Parquet, Location: InMemoryFileIndex[abfss://wwi-02@asadatalake01.dfs.core.windows.net/sale-small/Year=2019/Quarter=..., PartitionFilters: [],
PushedFilters: [IsNotNull(CustomerId), EqualTo(CustomerId,203)], ReadSchema: struct<CustomerId:int,TotalAmount:decimal(38,18)>
```

Now add another new cell to your notebook with the following code (notice the extra line at the beginning used to enable Hyperspace optimization in the Spark engine):

```
# Enable Hyperspace - Hyperspace optimization rules become visible to the Spark
optimizer and exploit existing Hyperspace indexes to optimize user queries
Hyperspace.enable(spark)
df1 = dfSales.filter("""CustomerId = 203""").select("""TotalAmount""")
df1.show()
df1.explain(True)
```

Run the new cell. The output will show that the physical execution plan is now using the index instead of the orginal data file.



```
# Enable Hyperspace - Hyperspace optimization rules become visible to the Spark optimizer and exploit existing Hyperspace indexes to optimize user queries
Hyperspace.enable(spark)
df1 = dfSales.filter("""CustomerId = 203""").select("""TotalAmount""")
df1.show()
df1.explain(True)
```

Command executed in 13s 417ms by ciprian on 11-30-2020 23:54:56.589 +02:00

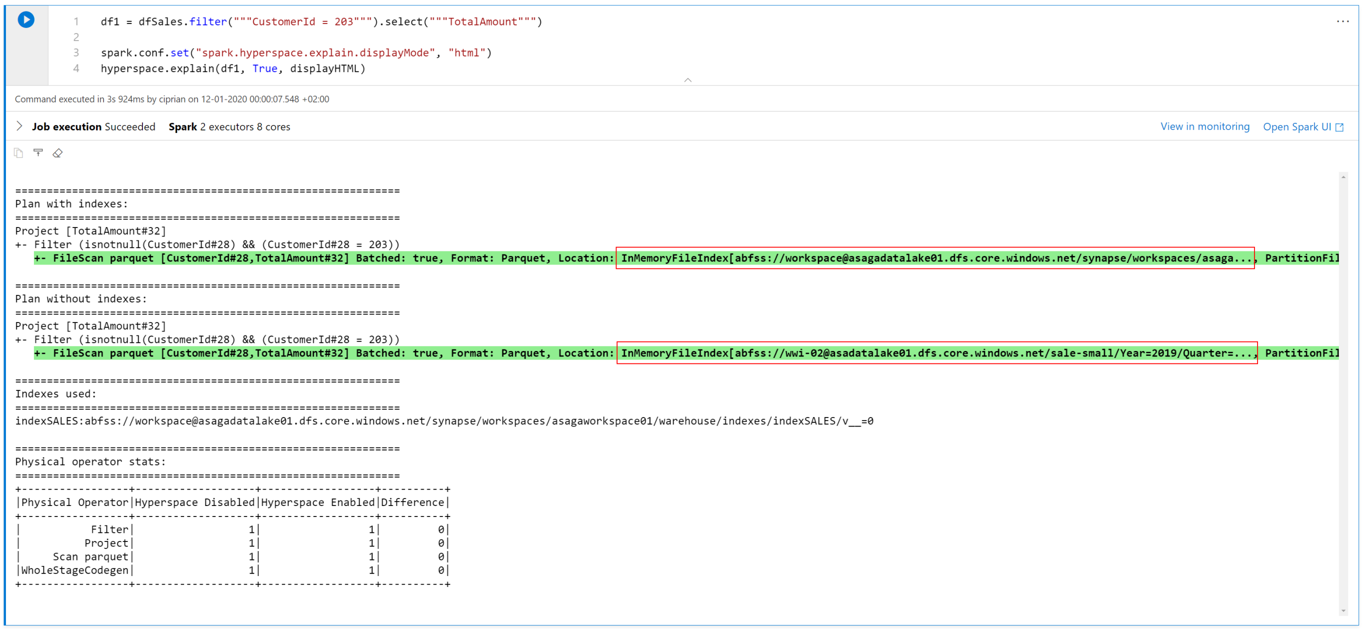
Job execution Succeeded   Spark 2 executors 8 cores   View in monitoring   Open Spark UI

```
+--------------------+
|         TotalAmount|
+--------------------+
|28.33000000000000...|
|110.9100000000000...|
|95.72000000000000...|
|103.8400000000000...|
|133.3200000000000...|
|92.28000000000000...|
|31.28000000000000...|
+--------------------+

== Parsed Logical Plan ==
'Project [unresolvedalias('TotalAmount, None)]
+- Filter (CustomerId#28 = 203)
   +- Relation[TransactionId#27,CustomerId#28,ProductId#29,Quantity#30,Price#31,TotalAmount#32,TransactionDate#33,ProfitAmount#34,Hour#35,Minute#36,StoreId#37] parquet

== Analyzed Logical Plan ==
TotalAmount: decimal(38,18)
Project [TotalAmount#32]
+- Filter (CustomerId#28 = 203)
   +- Relation[TransactionId#27,CustomerId#28,ProductId#29,Quantity#30,Price#31,TotalAmount#32,TransactionDate#33,ProfitAmount#34,Hour#35,Minute#36,StoreId#37] parquet

== Optimized Logical Plan ==
Project [TotalAmount#32]
+- Filter (isnotnull(CustomerId#28) && (CustomerId#28 = 203))
   +- Relation[CustomerId#28,TotalAmount#32] parquet

== Physical Plan ==
*(1) Project [TotalAmount#32]
+- *(1) Filter (isnotnull(CustomerId#28) && (CustomerId#28 = 203))
   +- *(1) FileScan parquet [CustomerId#28,TotalAmount#32] Batched: true, Format: Parquet, Location: InMemoryFileIndex[abfss://workspace@asagadatalake01.dfs.core.windows.net/synapse/workspaces/asaga..., PartitionFilters: [],
PushedFilters: [IsNotNull(CustomerId), EqualTo(CustomerId,203)], ReadSchema: struct<CustomerId:int,TotalAmount:decimal(38,18)>
```

Hyperspace provides an Explain API that allows you to compare the execution plans without indexes vs. with indexes. Add a new cell with the following code:

```
df1 = dfSales.filter("""CustomerId = 203""").select("""TotalAmount""")

spark.conf.set("spark.hyperspace.explain.displayMode", "html")
hyperspace.explain(df1, True, displayHTML)
```

Run the new cell. The output shows a comparison `Plan with indexes` vs. `Plan without indexes`. Observe how, in the first case the index file is used while in the second case the original data file is used.



```
df1 = dfSales.filter("""CustomerId = 203""").select("""TotalAmount""")

spark.conf.set("spark.hyperspace.explain.displayMode", "html")
hyperspace.explain(df1, True, displayHTML)
```

Command executed in 3s 924ms by ciprian on 12-01-2020 00:00:07.548 +02:00

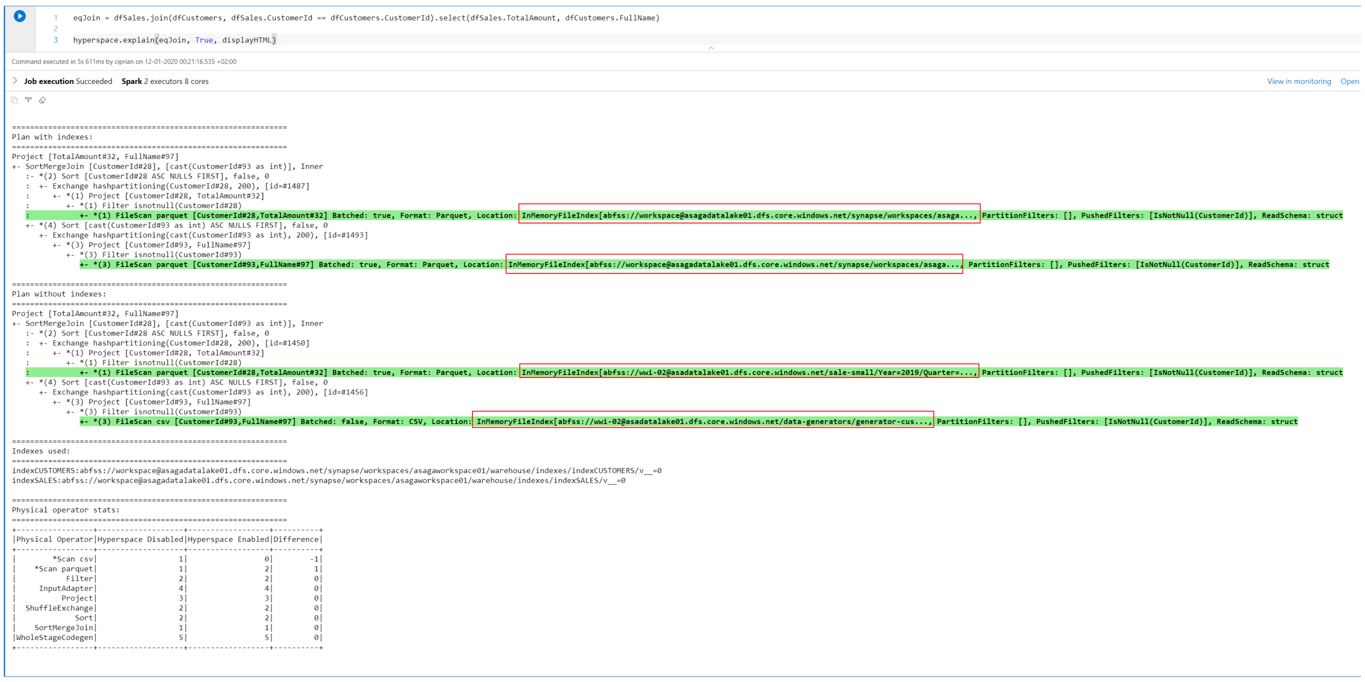
Job execution Succeeded **Spark** 2 executors 8 cores

```
=============================================================
Plan with indexes:
=============================================================
Project [TotalAmount#32]
+- Filter (isnotnull(CustomerId#28) && (CustomerId#28 = 203))
   +- FileScan parquet [CustomerId#28,TotalAmount#32] Batched: true, Format: Parquet, Location: InMemoryFileIndex[abfss://workspace@asagadatalake01.dfs.core.windows.net/synapse/workspaces/asaga..., PartitionFil

=============================================================
Plan without indexes:
=============================================================
Project [TotalAmount#32]
+- Filter (isnotnull(CustomerId#28) && (CustomerId#28 = 203))
   +- FileScan parquet [CustomerId#28,TotalAmount#32] Batched: true, Format: Parquet, Location: InMemoryFileIndex[abfss://wwi-02@asadatalake01.dfs.core.windows.net/sale-small/Year=2019/Quarter=..., PartitionFil

=============================================================
Indexes used:
=============================================================
indexSALES:abfss://workspace@asagadatalake01.dfs.core.windows.net/synapse/workspaces/asagaworkspace01/warehouse/indexes/indexSALES/v__=0

=============================================================
Physical operator stats:
=============================================================
+-----------------+-------------------+------------------+----------+
|Physical Operator|Hyperspace Disabled|Hyperspace Enabled|Difference|
+-----------------+-------------------+------------------+----------+
|           Filter|                  1|                 1|         0|
|          Project|                  1|                 1|         0|
|     Scan parquet|                  1|                 1|         0|
|WholeStageCodegen|                  1|                 1|         0|
+-----------------+-------------------+------------------+----------+
```

Let's investigate now a more complex case, involving a join operation. Add a new cell with the following code:

```
eqJoin = dfSales.join(dfCustomers, dfSales.CustomerId ==
dfCustomers.CustomerId).select(dfSales.TotalAmount, dfCustomers.FullName)

hyperspace.explain(eqJoin, True, displayHTML)
```

Run the new cell. The output shows again a comparison `Plan with indexes` vs. `Plan without indexes`, where indexes are used in the first case and the original data files in the second.



```
eqJoin = dfSales.join(dfCustomers, dfSales.CustomerId == dfCustomers.CustomerId).select(dfSales.TotalAmount, dfCustomers.FullName)

hyperspace.explain(eqJoin, True, displayHTML)
```

Command executed in 5s 611ms by ciprian on 12-01-2020 00:21:16.535 +02:00

Job execution Succeeded Spark 2 executors 8 cores View in monitoring Open

```
=============================================================
Plan with indexes:
=============================================================
Project [TotalAmount#32, FullName#97]
+- SortMergeJoin [CustomerId#28], [cast(CustomerId#93 as int)], Inner
   :- *(2) Sort [CustomerId#28 ASC NULLS FIRST], false, 0
   :  +- Exchange hashpartitioning(CustomerId#28, 200), [id=#1487]
   :     +- *(1) Project [CustomerId#28, TotalAmount#32]
   :        +- *(1) Filter isnotnull(CustomerId#28)
   :           +- *(1) FileScan parquet [CustomerId#28,TotalAmount#32] Batched: true, Format: Parquet, Location: InMemoryFileIndex[abfss://workspace@asagadatalake01.dfs.core.windows.net/synapse/workspaces/asaga..., PartitionFilters: [], PushedFilters: [IsNotNull(CustomerId)], ReadSchema: struct
   +- *(4) Sort [cast(CustomerId#93 as int) ASC NULLS FIRST], false, 0
      +- Exchange hashpartitioning(cast(CustomerId#93 as int), 200), [id=#1493]
         +- *(3) Project [CustomerId#93, FullName#97]
            +- *(3) Filter isnotnull(CustomerId#93)
               +- *(3) FileScan parquet [CustomerId#93,FullName#97] Batched: true, Format: Parquet, Location: InMemoryFileIndex[abfss://workspace@asagadatalake01.dfs.core.windows.net/synapse/workspaces/asaga..., PartitionFilters: [], PushedFilters: [IsNotNull(CustomerId)], ReadSchema: struct

=============================================================
Plan without indexes:
=============================================================
Project [TotalAmount#32, FullName#97]
+- SortMergeJoin [CustomerId#28], [cast(CustomerId#93 as int)], Inner
   :- *(2) Sort [CustomerId#28 ASC NULLS FIRST], false, 0
   :  +- Exchange hashpartitioning(CustomerId#28, 200), [id=#1450]
   :     +- *(1) Project [CustomerId#28, TotalAmount#32]
   :        +- *(1) Filter isnotnull(CustomerId#28)
   :           +- *(1) FileScan parquet [CustomerId#28,TotalAmount#32] Batched: true, Format: Parquet, Location: InMemoryFileIndex[abfss://wwi-02@asadatalake01.dfs.core.windows.net/sale-small/Year=2019/Quarter=..., PartitionFilters: [], PushedFilters: [IsNotNull(CustomerId)], ReadSchema: struct
   +- *(4) Sort [cast(CustomerId#93 as int) ASC NULLS FIRST], false, 0
      +- Exchange hashpartitioning(cast(CustomerId#93 as int), 200), [id=#1456]
         +- *(3) Project [CustomerId#93, FullName#97]
            +- *(3) Filter isnotnull(CustomerId#93)
               +- *(3) FileScan csv [CustomerId#93,FullName#97] Batched: false, Format: CSV, Location: InMemoryFileIndex[abfss://wwi-02@asadatalake01.dfs.core.windows.net/data-generators/generator-cus..., PartitionFilters: [], PushedFilters: [IsNotNull(CustomerId)], ReadSchema: struct

=============================================================
Indexes used:
=============================================================
indexCUSTOMERS:abfss://workspace@asagadatalake01.dfs.core.windows.net/synapse/workspaces/asagaworkspace01/warehouse/indexes/indexCUSTOMERS/v__=0
indexSALES:abfss://workspace@asagadatalake01.dfs.core.windows.net/synapse/workspaces/asagaworkspace01/warehouse/indexes/indexSALES/v__=0

=============================================================
Physical operator stats:
=============================================================
+----------------+-------------------+------------------+----------+
|Physical Operator|Hyperspace Disabled|Hyperspace Enabled|Difference|
+----------------+-------------------+------------------+----------+
|       *Scan csv|                  1|                 0|        -1|
|   *Scan parquet|                  1|                 2|         1|
|          Filter|                  2|                 2|         0|
|    InputAdapter|                  4|                 4|         0|
|         Project|                  3|                 3|         0|
| ShuffleExchange|                  2|                 2|         0|
|            Sort|                  2|                 2|         0|
|   SortMergeJoin|                  1|                 1|         0|
|WholeStageCodegen|                  5|                 5|         0|
+----------------+-------------------+------------------+----------+
```

In case you want to deactivate Hyperspace and cleanup the indexes, you can run the following code:

```
# Disable Hyperspace - Hyperspace rules no longer apply during query optimization. Disabling Hyperspace has no impact on created indexes because they remain intact
Hyperspace.disable(spark)

hyperspace.deleteIndex("indexSALES")
hyperspace.vacuumIndex("indexSALES")
hyperspace.deleteIndex("indexCUSTOMERS")
hyperspace.vacuumIndex("indexCUSTOMERS")
```