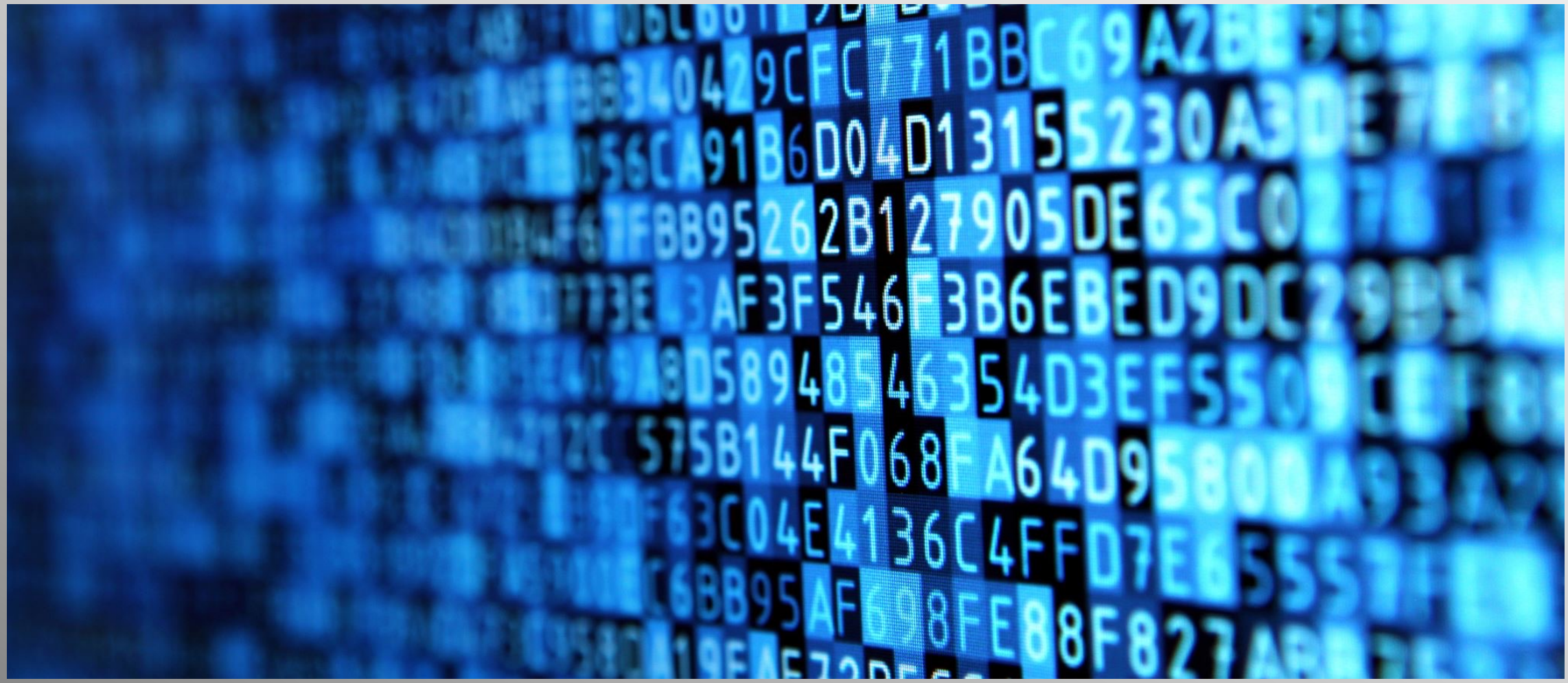
Let your Fabric work!

Load testing Microsoft Fabric

Reitse Eskens

Definitions

A large amount of data



Load testing

Reitse Eskens

Engineer | Architect

Axians Business Analytics

Reitse.Eskens@axians.com
/in/reitseeskens
https://sqlreitse.com
@2meterDBA

MVP Microsoft® Most Valuable Professional

axians Microsoft CERTIFIED TRAINER MCT

Coming from a SQL world

From definitions to Monitoring

Learning a new tool:
Fabric Capacity Metrics

About capacity units

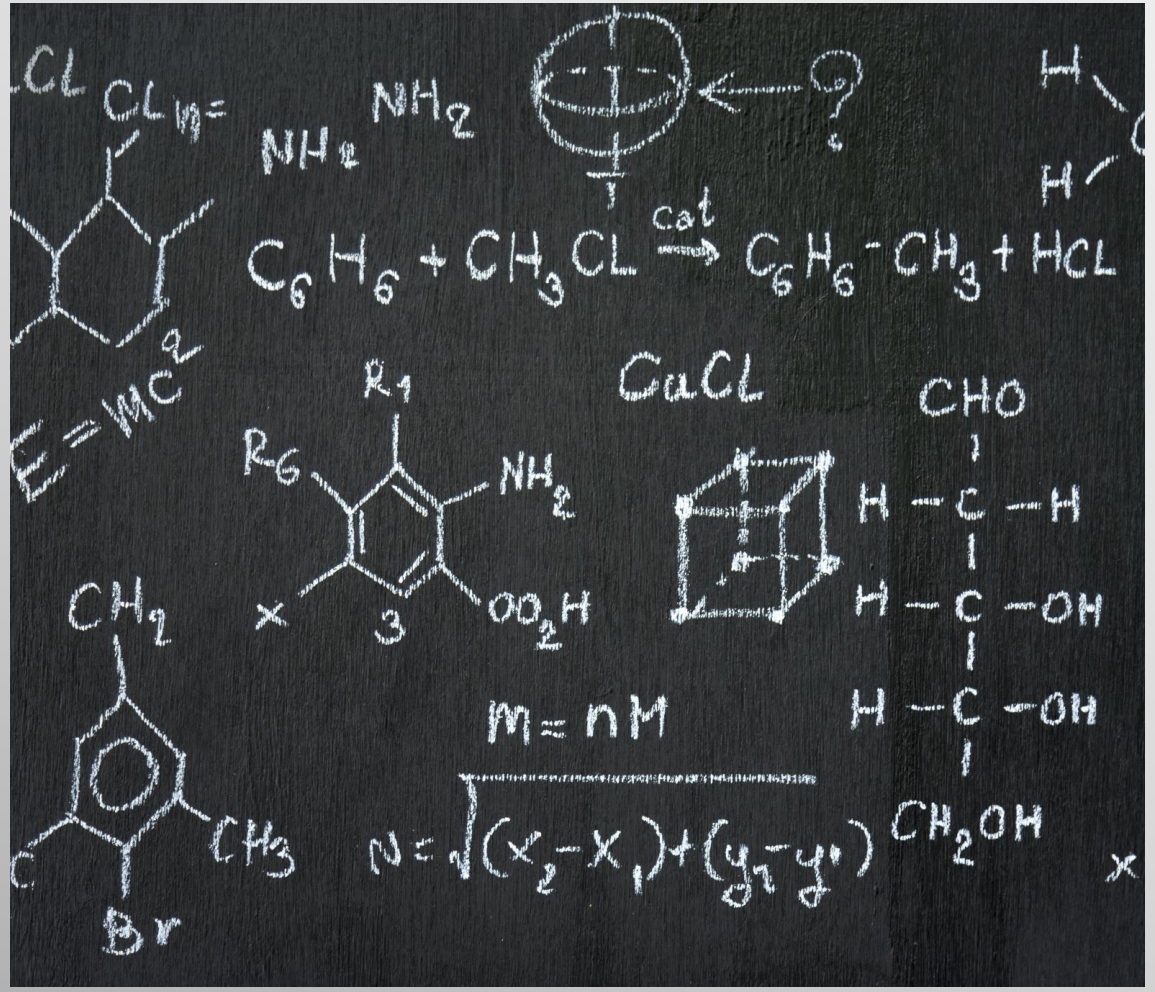
1 second of work = 1 capacity unit

F2 equals to 2 capacity units per second

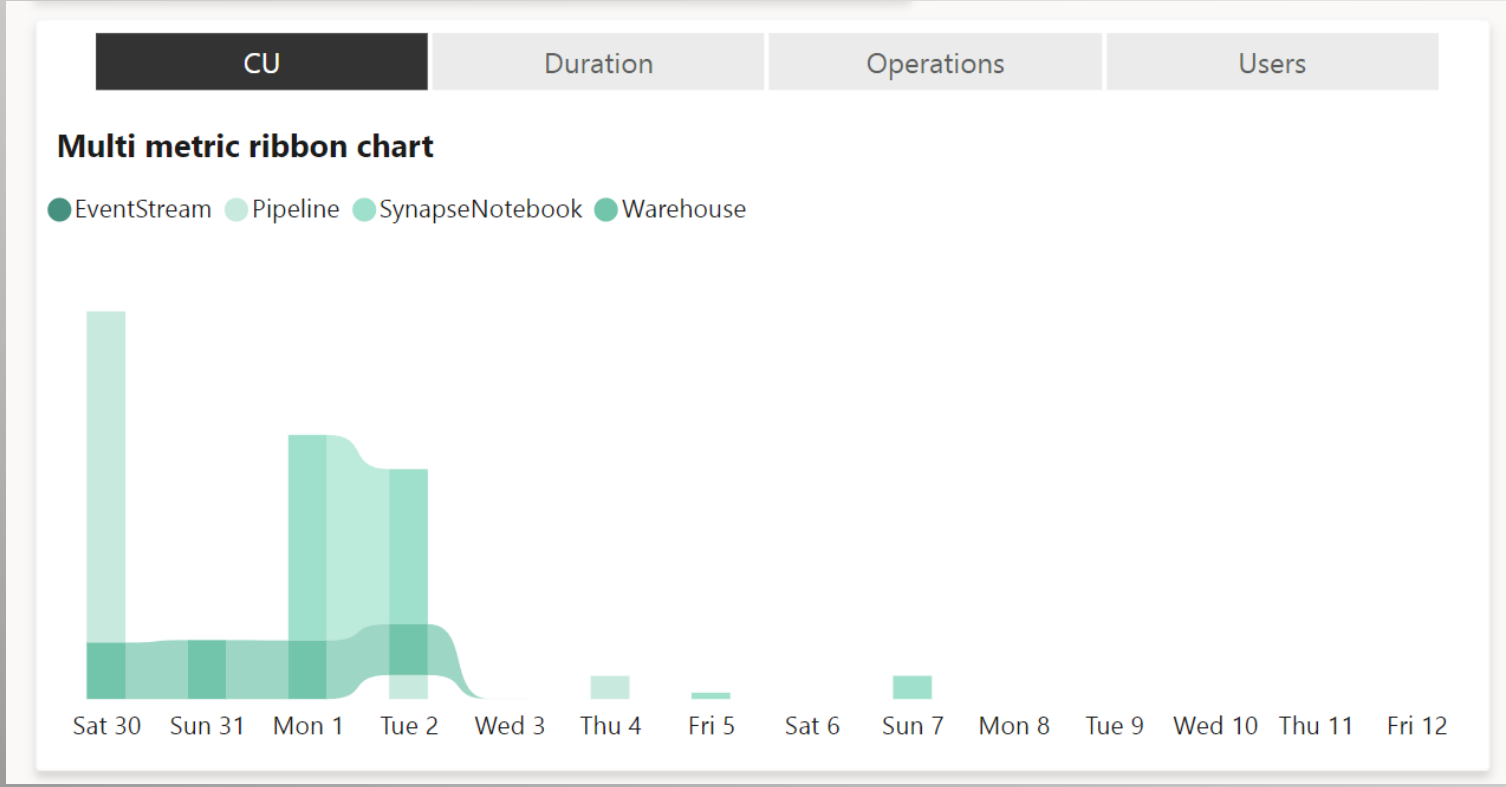
Capacity units are evaluated in blocks of 30 seconds

F2 = 2 * 30s = 60 capacity units

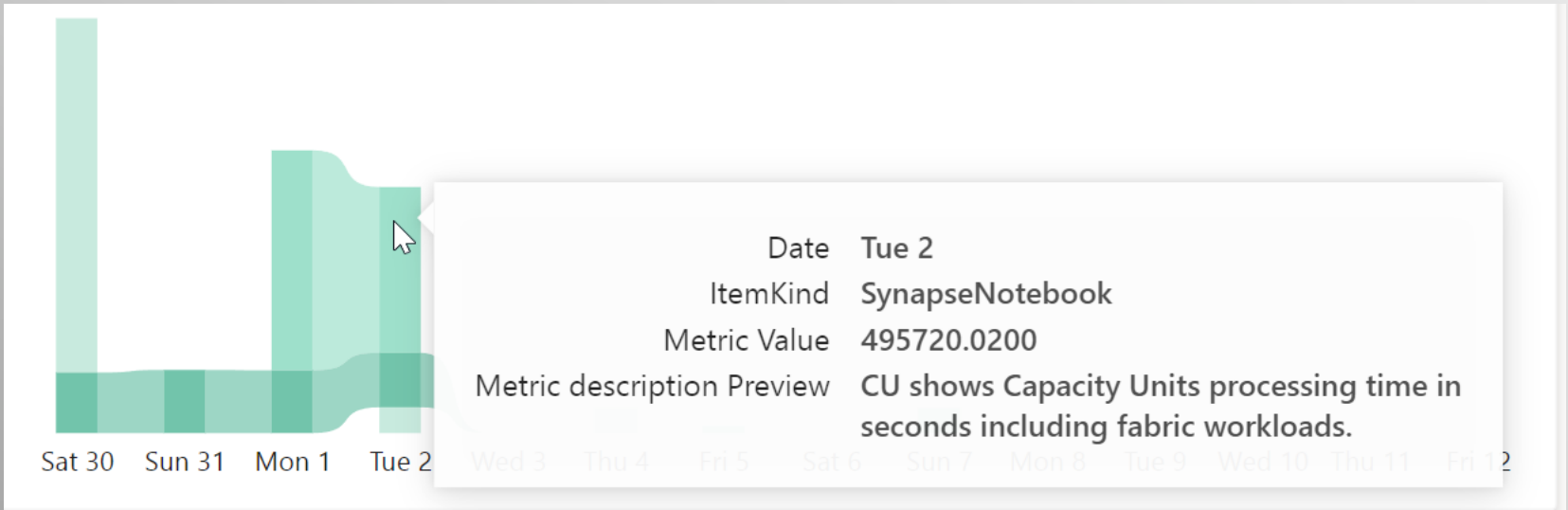
F64 = 64 * 30s = 1920 capacity units



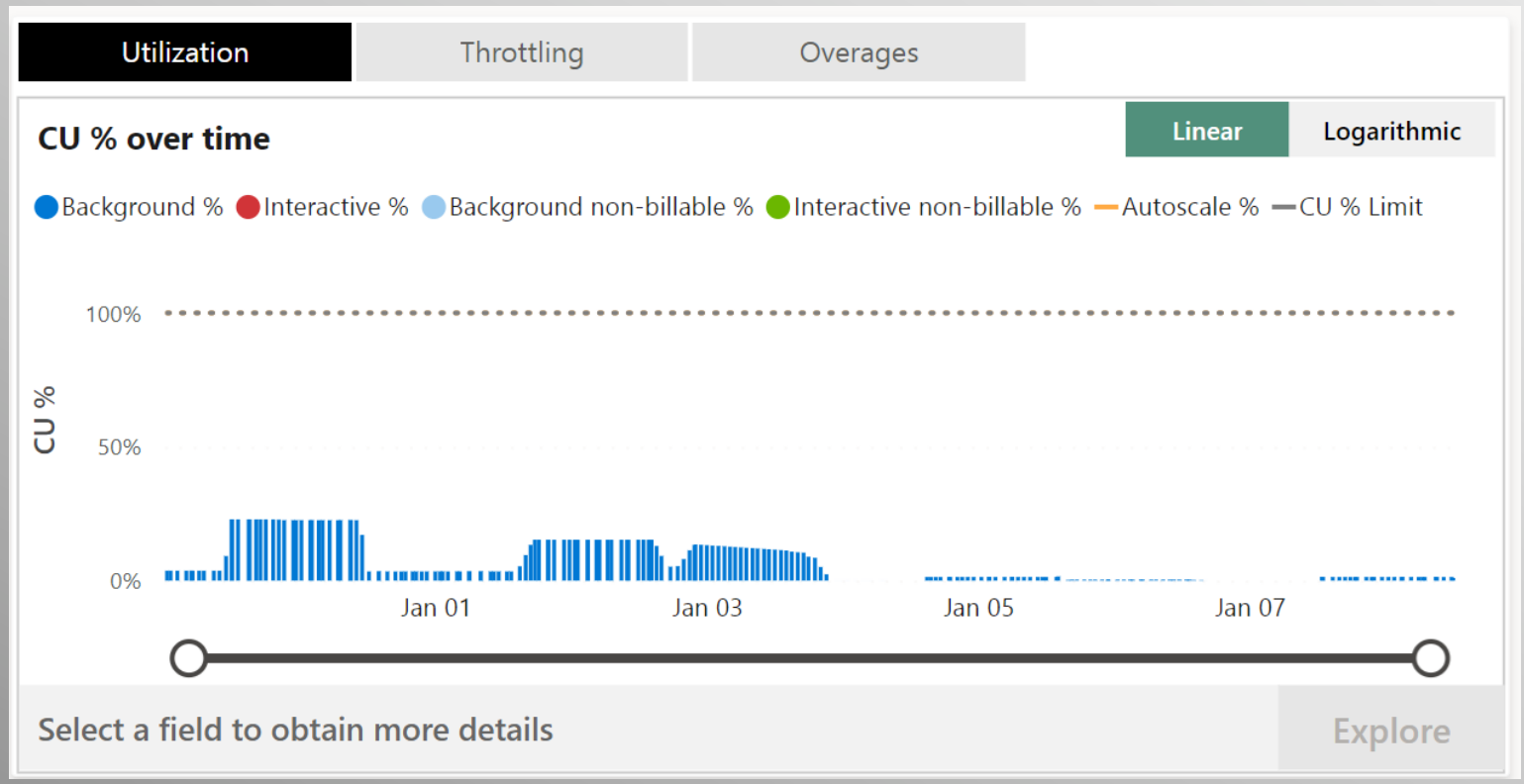
Fabric Capacity Metrics: Multi metric ribbon chart



Fabric Capacity Metrics:
Multi metric ribbon chart



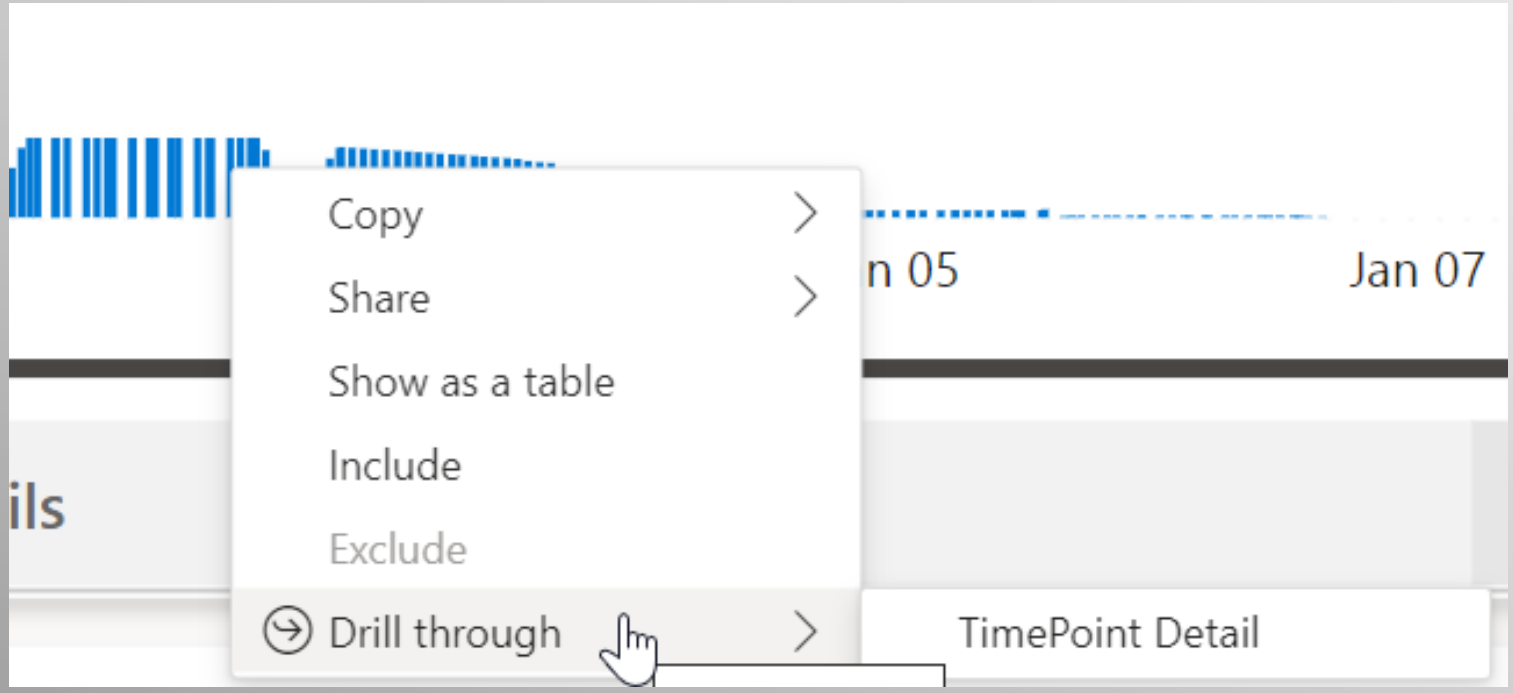
Fabric Capacity Metrics:
Capacity Utilization and Throttling



Fabric Capacity Metrics:
Capacity Utilization and Throttling

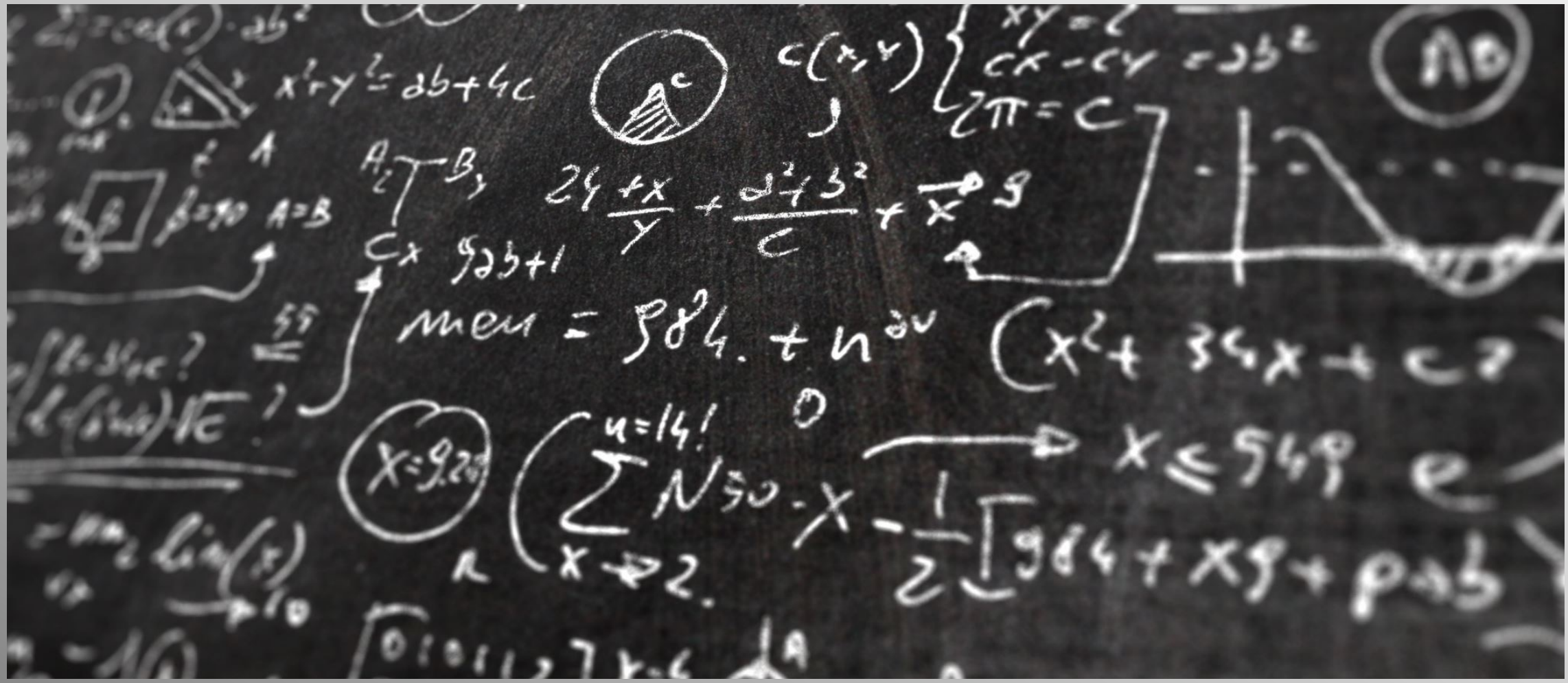
Policy	Consumption	Impact
Overage protection	Usage <= 10 minutes	Jobs can consume 10 minutes of future capacity use without throttling.
Interactive delay	10 minutes < usage <= 60 minutes	User requested interactive jobs are throttled.
Interactive rejection	60 minutes < usage <= 24 hours	User requested interactive jobs are rejected.
Background rejection	Usage > 24 hours	User scheduled background jobs are rejected and not executed.

Fabric Capacity Metrics: Drill through



- Large amounts of data is a matter of perspective
- Microsoft Fabric allows you to learn new skills
- Measuring Fabric capacity usage is one of them

From theory to data



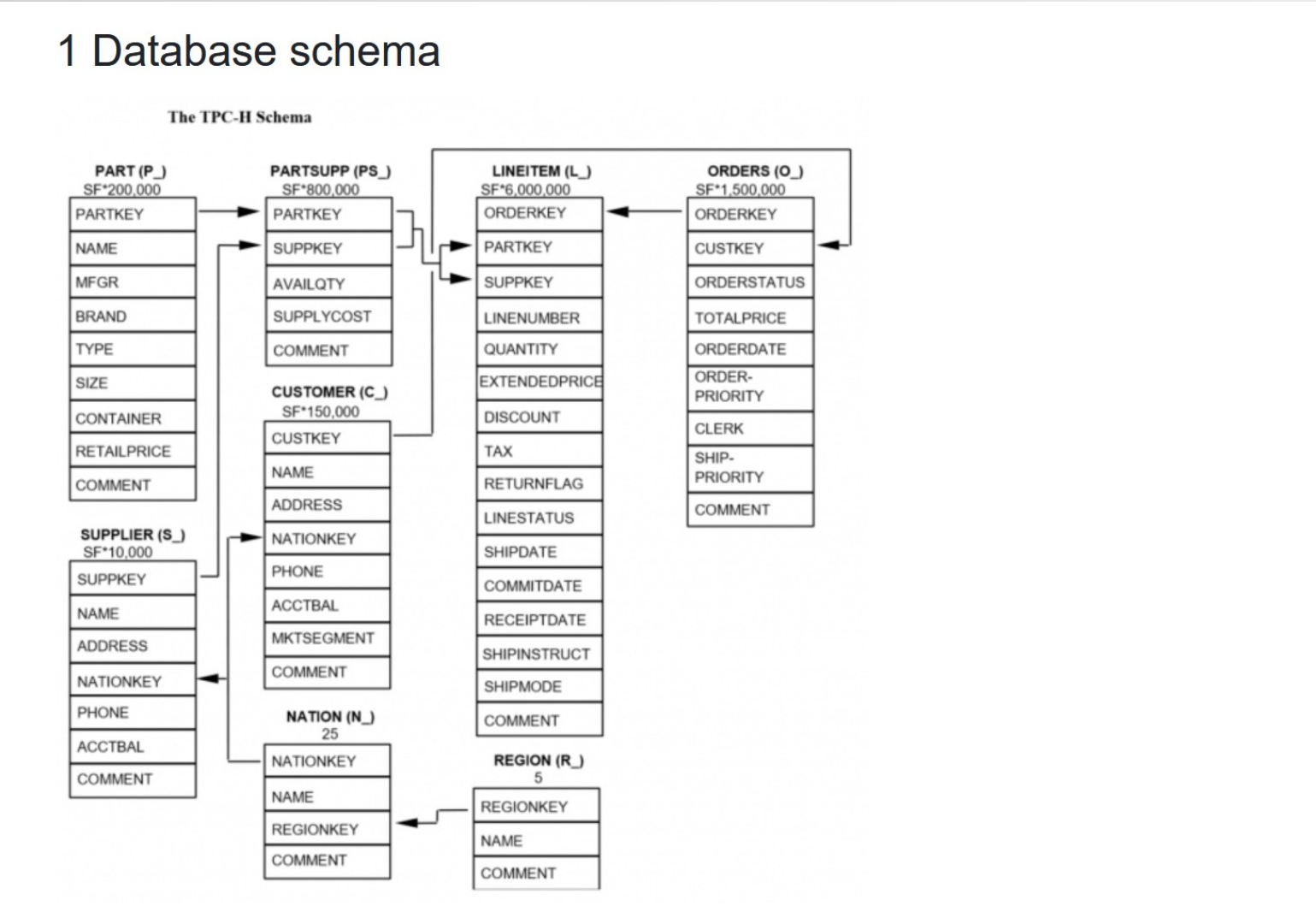
Preparing the data

- TPC-H
- .\dbgen.exe -s 10 -C 3 -v -S 1
  - -s 10 = 10 GB of data
  - -C 3 = three sets of tables
  - -v = verbose
  - -S 1 = first set of the three sets

Preparing the data

- .\dbgen.exe -v -U200
  - -v = verbose
  - -U 200 = 200 sets of update and delete sets

Database schema

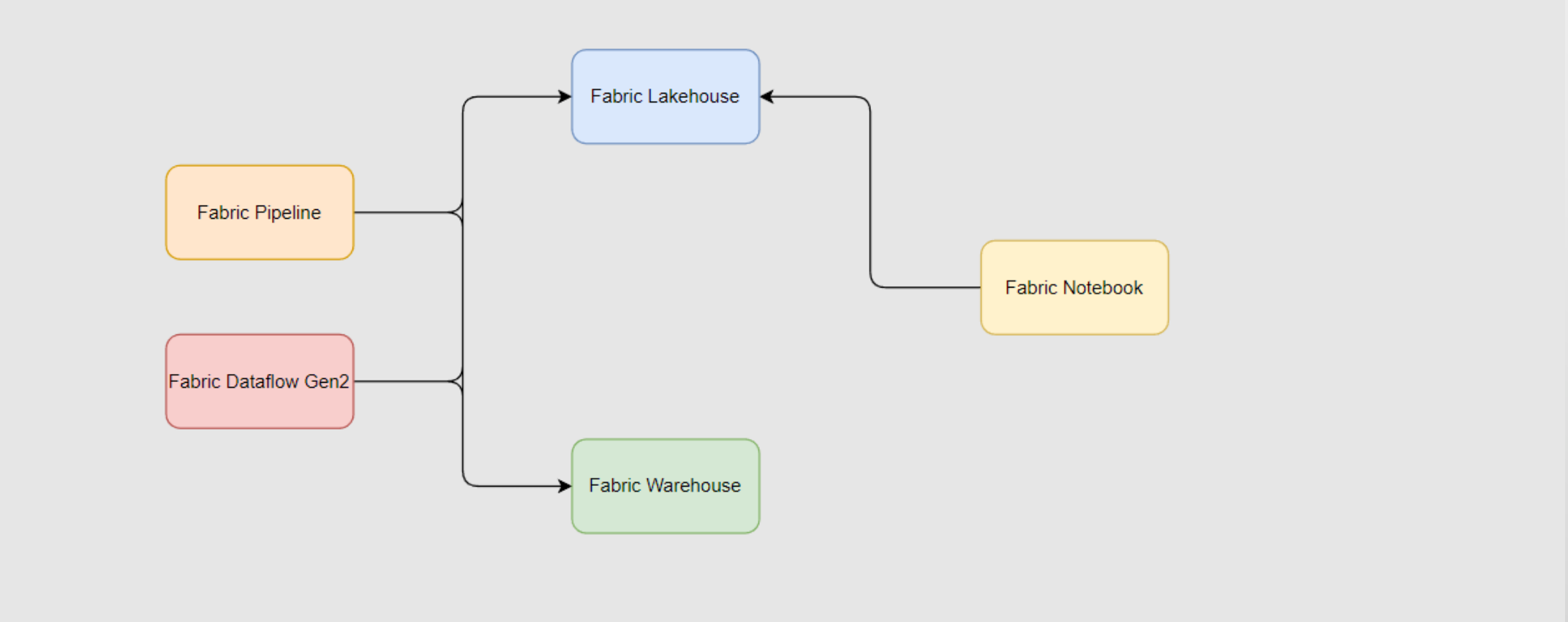


Files

Name	Access Tier	Access Tier Last Modified	Last Modified	Blob Type	Content Type	Size	Status
region.tbl.csv	Hot (inferred)		2-1-2024 15:45	Block Blob	text/plain	394 B	Active
nation.csv	Hot (inferred)		2-1-2024 15:45	Block Blob	text/plain	2,20 KiB	Active
supplier.tbl.1.csv	Hot (inferred)		2-1-2024 15:08	Block Blob	text/plain	137,20 MiB	Active
supplier.tbl.5.csv	Hot (inferred)		2-1-2024 16:13	Block Blob	text/plain	138,24 MiB	Active
supplier.tbl.2.csv	Hot (inferred)		2-1-2024 15:44	Block Blob	text/plain	138,25 MiB	Active
supplier.tbl.3.csv	Hot (inferred)		2-1-2024 16:02	Block Blob	text/plain	138,29 MiB	Active
supplier.tbl.4.csv	Hot (inferred)		2-1-2024 16:35	Block Blob	text/plain	138,29 MiB	Active
part.tbl.1.csv	Hot (inferred)		2-1-2024 16:26	Block Blob	text/plain	2,30 GiB	Active
part.tbl.5.csv	Hot (inferred)		2-1-2024 15:45	Block Blob	text/plain	2,31 GiB	Active
part.tbl.4.csv	Hot (inferred)		2-1-2024 16:15	Block Blob	text/plain	2,31 GiB	Active
part.tbl.2.csv	Hot (inferred)		2-1-2024 16:34	Block Blob	text/plain	2,31 GiB	Active
part.tbl.3.csv	Hot (inferred)		2-1-2024 15:08	Block Blob	text/plain	2,31 GiB	Active
customer.tbl.3.csv	Hot (inferred)		2-1-2024 16:38	Block Blob	text/plain	2,32 GiB	Active
customer.tbl.5.csv	Hot (inferred)		2-1-2024 15:33	Block Blob	text/plain	2,32 GiB	Active
customer.tbl.2.csv	Hot (inferred)		2-1-2024 16:19	Block Blob	text/plain	2,32 GiB	Active
customer.tbl.4.csv	Hot (inferred)		2-1-2024 16:35	Block Blob	text/plain	2,32 GiB	Active
partsupp.tbl.1.csv	Hot (inferred)		2-1-2024 15:50	Block Blob	text/plain	11,51 GiB	Active
partsupp.tbl.3.csv	Hot (inferred)		2-1-2024 15:33	Block Blob	text/plain	11,55 GiB	Active
partsupp.tbl.5.csv	Hot (inferred)		2-1-2024 15:44	Block Blob	text/plain	11,55 GiB	Active
partsupp.tbl.2.csv	Hot (inferred)		2-1-2024 16:15	Block Blob	text/plain	11,55 GiB	Active
partsupp.tbl.4.csv	Hot (inferred)		2-1-2024 16:39	Block Blob	text/plain	11,55 GiB	Active
orders.tbl.1.csv	Hot (inferred)		2-1-2024 15:59	Block Blob	text/plain	16,79 GiB	Active
orders.tbl.2.csv	Hot (inferred)		2-1-2024 16:37	Block Blob	text/plain	16,87 GiB	Active
orders.tbl.4.csv	Hot (inferred)		2-1-2024 16:34	Block Blob	text/plain	16,96 GiB	Active
orders.tbl.3	Hot (inferred)		2-1-2024 16:18	Block Blob	text/plain	16,96 GiB	Active
orders.tbl.5.csv	Hot (inferred)		2-1-2024 16:01	Block Blob	text/plain	16,96 GiB	Active
lineitem.tbl.1	Hot (inferred)		2-1-2024 15:43	Block Blob	text/plain	75,42 GiB	Active
lineitem.tbl.1.csv	Hot (inferred)		2-1-2024 15:31	Block Blob	text/plain	75,42 GiB	Active
lineitem.tbl.2.csv	Hot (inferred)		2-1-2024 16:34	Block Blob	text/plain	75,70 GiB	Active
lineitem.tbl.3.csv	Hot (inferred)		2-1-2024 15:56	Block Blob	text/plain	76,07 GiB	Active
lineitem.tbl.5.csv	Hot (inferred)		2-1-2024 15:21	Block Blob	text/plain	76,08 GiB	Active
lineitem.tbl.4.csv	Hot (inferred)		2-1-2024 16:13	Block Blob	text/plain	76,08 GiB	Active

- TPC-H is free and contains nonsense data
- A little quirky to get running
- Very flexible to get the size you want

Loading options



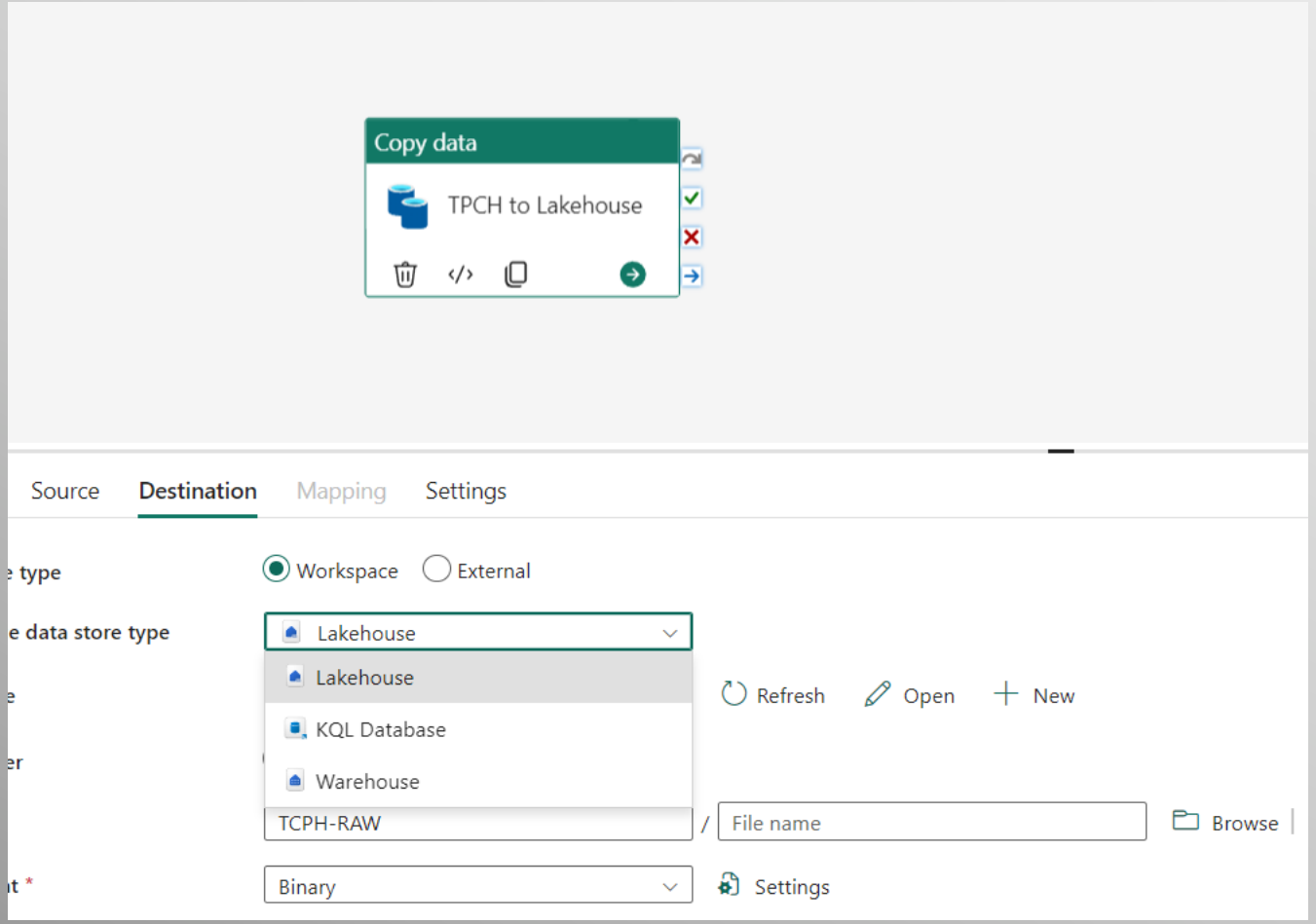
Loading…

- Pipeline
- Notebook
- Dataflow Gen2

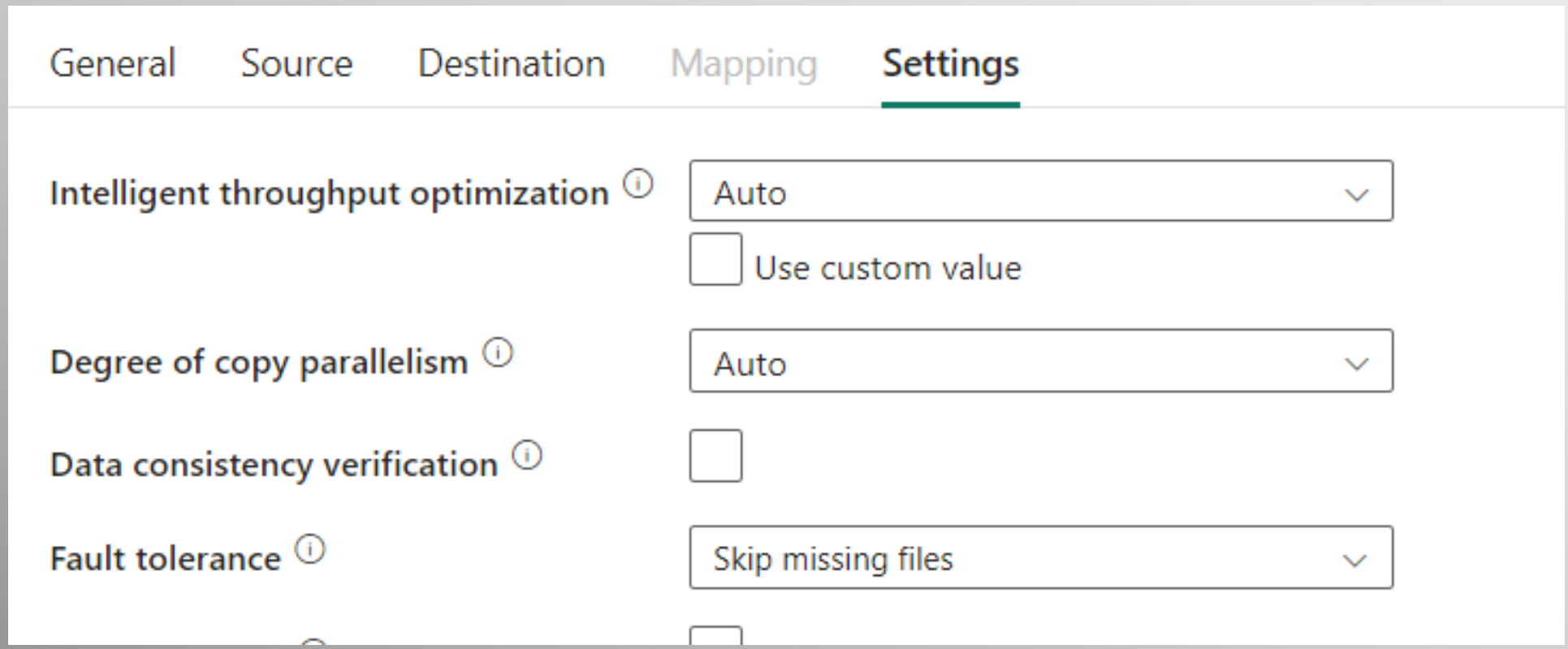
Pipeline

Easy going

Pipeline


Copy data
TPCH to Lakehouse
Source
Destination
Mapping
Settings
e type
Workspace
External
e data store type
Lakehouse
Lakehouse
KQL Database
Warehouse
Refresh
Open
New
er
TCPH-RAW
/
File name
Browse
t *
Binary
Settings

Pipeline


General
Source
Destination
Mapping
Settings
Intelligent throughput optimization
Auto
Use custom value
Degree of copy parallelism
Auto
Data consistency verification
Fault tolerance
Skip missing files

Comparison

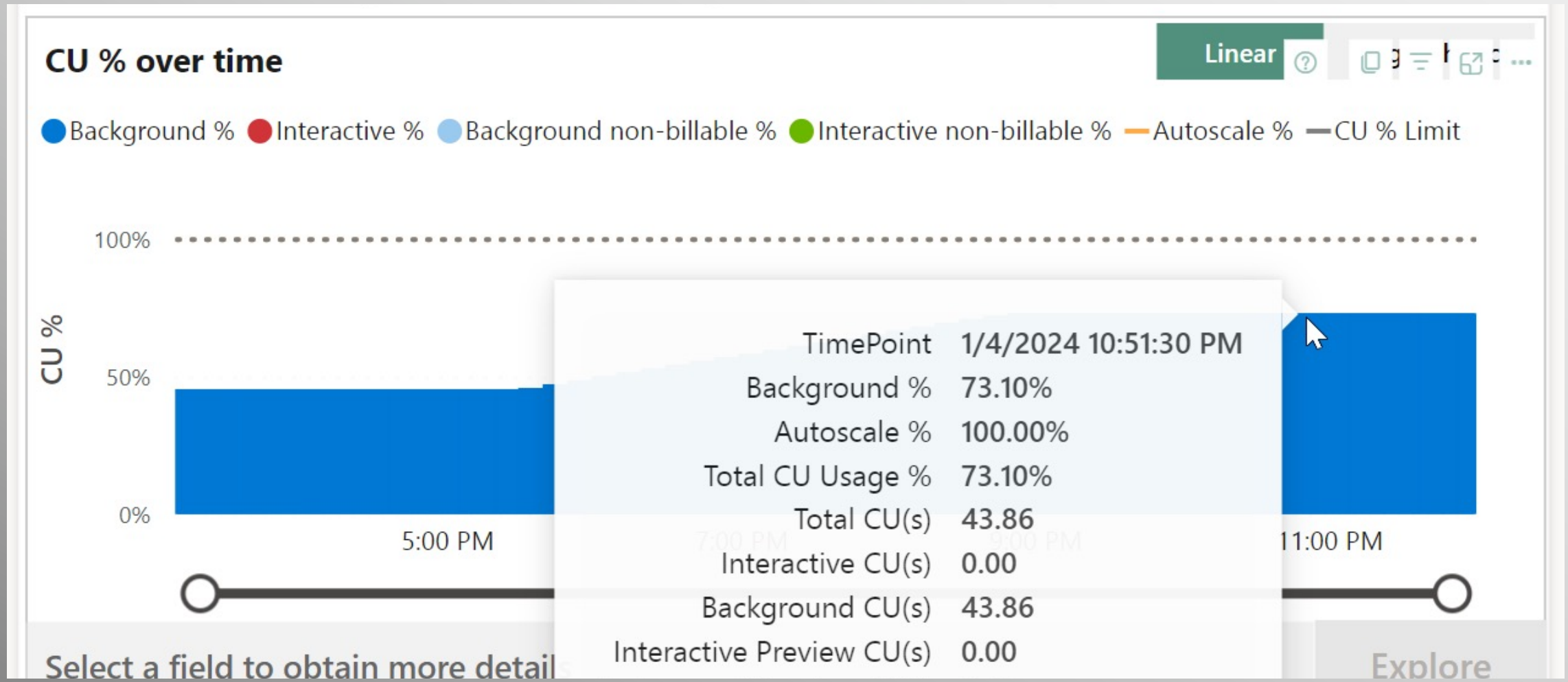
Pipeline F64

Throughput	335.235 MB/s
Total duration	00:33:14
Duration breakdown	
Start time	1/2/2024, 4:54:14 PM
Optimized throughput	Balanced
Used parallel copies	638

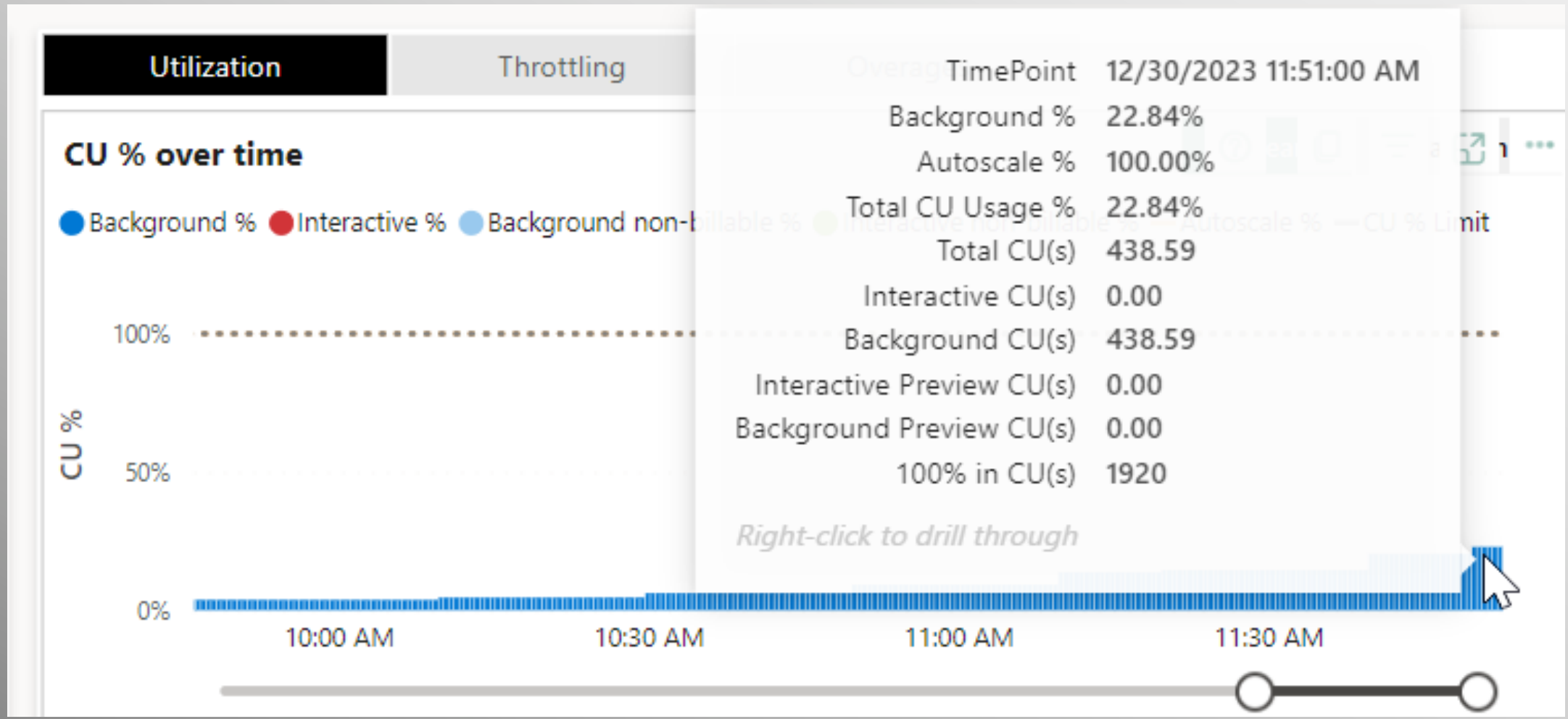
Pipeline F2

Throughput	341.603 MB/s
Total duration	00:32:39
Duration breakdown	
Start time	1/4/2024, 2:29:49 PM
Optimized throughput	Balanced
Used parallel copies	638

Pipeline F2


CU % over time
Linear
Background %
Interactive %
Background non-billable %
Interactive non-billable %
Autoscale %
CU % Limit
100%
50%
0%
CU %
5:00 PM
11:00 PM
TimePoint 1/4/2024 10:51:30 PM
Background % 73.10%
Autoscale % 100.00%
Total CU Usage % 73.10%
Total CU(s) 43.86
Interactive CU(s) 0.00
Background CU(s) 43.86
Interactive Preview CU(s) 0.00
Select a field to obtain more detail
Explore

Pipeline F64



Pipeline F64

- **Timepoint CU(s)** - This metric is the total CU(s) divided by 2,880, and is used to determine how much CU(s) the background operation contributes to this timepoint.

	Status	User	Duration (s)	Total CU (s)	Timepoint CU (s)	Throttling (s)	% of Base capacity	Billing type
.	Success	Power BI Service	920	318,240.0000	110.5000	0	5.76%	Billable
.	Success	Power BI Service	938	243,000.0000	84.3750	0	4.39%	Billable
.	Success	Power BI Service	459	145,080.0000	50.3750	0	2.62%	Billable
.	Success	Power BI Service	701	120,960.0000	42.0000	0	2.19%	Billable
.	Success	Power BI Service	1,087	88,560.0000	30.7500	0	1.60%	Billable
.	Success	Power BI Service	218	50,040.0000	17.3750	0	0.90%	Billable
.	Success	Power BI Service	179	46,800.0000	16.2500	0	0.85%	Billable
.	Success	Power BI Service	188	46,440.0000	16.1250	0	0.84%	Billable

(24 * 60 *60) / 2880 = 30 (seconds)

Pipeline

F2 uses 73% capacity

F64 uses 2,2% capacity

What else is going on?

Notebook

Let’s code!

Notebook (CSV to Parquet)

```
folderPath = "Files/TCPH-RAW/files/customer.tbl.*.csv"
table_name = "raw_customers"

schema = StructType([
    StructField("C_CUSTKEY", IntegerType()),
    StructField("C_NAME", StringType()),
    StructField("C_ADDRESS", StringType()),
    StructField("C_NATIONKEY", IntegerType()),
    StructField("C_PHONE", StringType()),
    StructField("C_ACCTBAL", DoubleType()),
    StructField("C_MKTSEGMENT", StringType()),
    StructField("C_COMMENT", StringType()),
    StructField("C_Empty", StringType())
])

df = spark.read.csv(
    folderPath,
    header=False,
    mode="DROPMALFORMED",
    sep="|",
    schema=schema
)
df.write.mode("overwrite").format("delta").save("Tables/" + table_name)
```

Notebook (CSV to Parquet) F2

✓ 2 min 29 sec -Command executed in 2 min 28 sec 902 ms by Admin on 11:15:34 AM, 1/05/24

Tasks	Duration	Processed	Data read	Data written
2/2 succeeded	198 ms	19 rows	12.12 KB	0 B
76/76 succeeded	1 min 4 sec 806 ms	120,000,000 rows	0 B	6.3 GB
5/5 succeeded	1 min 17 sec 26 ms	120,000,000 rows	6.3 GB	4.68 GB

Notebook (CSV to Parquet) F64

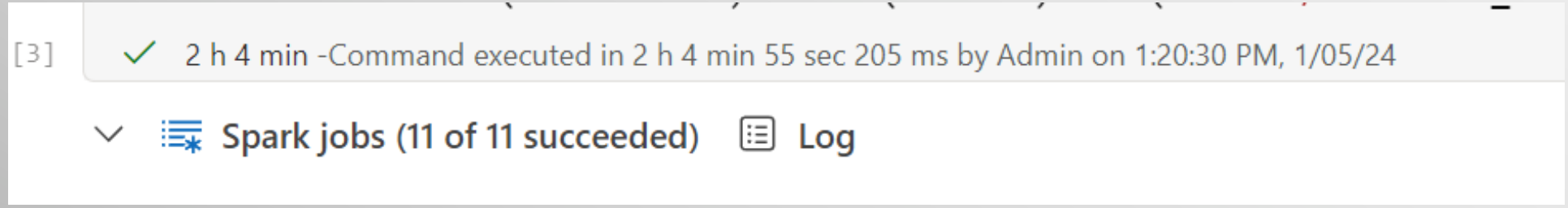
✓ 1 min 13 sec -Command executed in 1 min 13 sec 264 ms by Admin on 1:46:41 PM, 1/05/24

Tasks	Duration	Processed	Data read	Data written
1/1 succeeded	186 ms	16 rows	5.96 KB	5.25 KB
50/50 succeeded	466 ms	58 rows	5.25 KB	4.37 KB
1/1 succeeded	47 ms	50 rows	4.37 KB	0 B
76/76 succeeded	33 sec 632 ms	120,000,000 rows	0 B	6.3 GB
5/5 succeeded	35 sec 49 ms	120,000,000 rows	6.3 GB	4.68 GB

Notebook (CSV to Parquet)

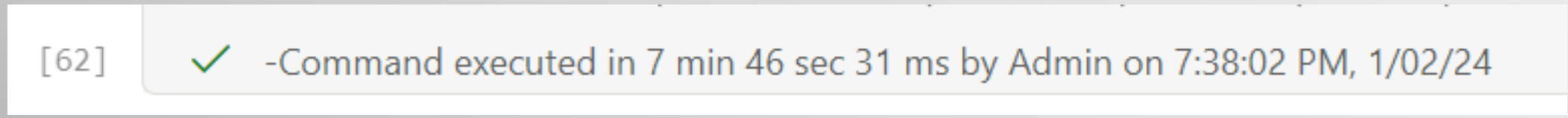
```
folderPath = "Files/TCPH-RAW/files/lineitem.tbl.*.csv"
table_name = "raw_lineitems"

schema = StructType([
    StructField("L_ORDERKEY", IntegerType()),
    StructField("L_PARTKEY", IntegerType()),
    StructField("L_SUPPKEY", IntegerType()),
    StructField("L_LINENUMBER", IntegerType()),
    StructField("L_QUANTITY", IntegerType()),
    StructField("L_EXTENDEDPRICE", DoubleType()),
    StructField("L_DISCOUNT", DoubleType()),
    StructField("L_TAX", DoubleType()),
    StructField("L_RETURNFLAG", StringType()),
    StructField("L_LINESTATUS", StringType()),
    StructField("L_SHIPDATE", StringType()),
    StructField("L_COMMITDATE", StringType()),
    StructField("L_RECEIPTDATE", StringType()),
    StructField("L_SHIPINSTRUCT", StringType()),
    StructField("L_SHIPMODE", StringType()),
    StructField("L_COMMENT", StringType()),
    StructField("L_Empty", StringType())
])

df = spark.read.csv(
    folderPath,
    header=False,
    mode="DROPMALFORMED",
    sep="|",
    schema=schema
)
df.write.mode("overwrite").format("delta").save("Tables/" + table_name)
```

Notebook (CSV to Parquet) F2



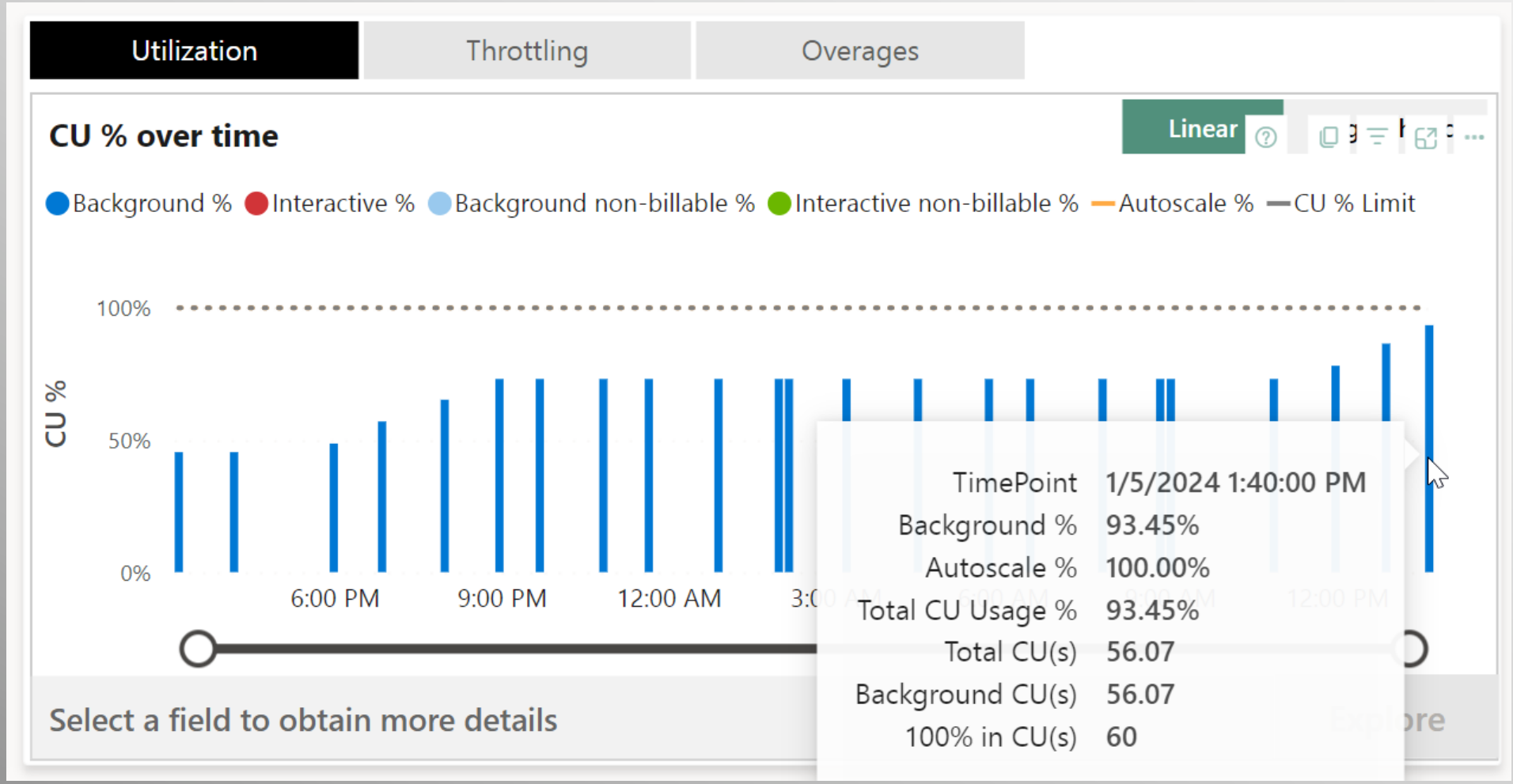
3036/3044 succeeded, 8 failed	1 h 17 min 29 sec 991 ms	4,294,965,612 rows	6 GB	174.67 GB
136/136 succeeded	47 min 19 sec 41 ms	4,294,965,612 rows	174.67 GB	77.44 GB

Notebook (CSV to Parquet) F64

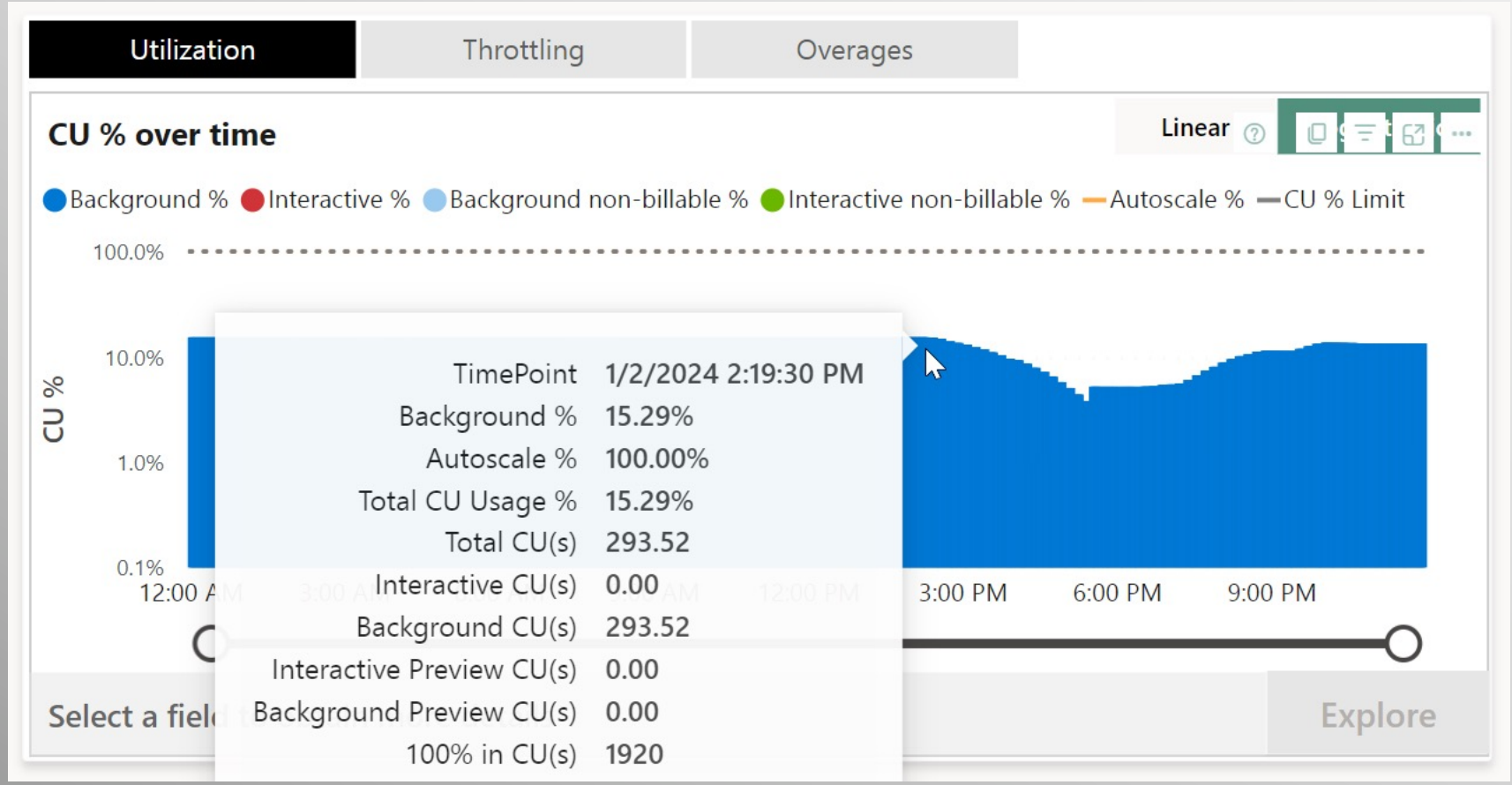


3036/3044 succeeded, 8 failed	1 h 17 min 29 sec 991 ms	4,294,965,612 rows	6 GB	174.67 GB
136/136 succeeded	47 min 19 sec 41 ms	4,294,965,612 rows	174.67 GB	77.44 GB

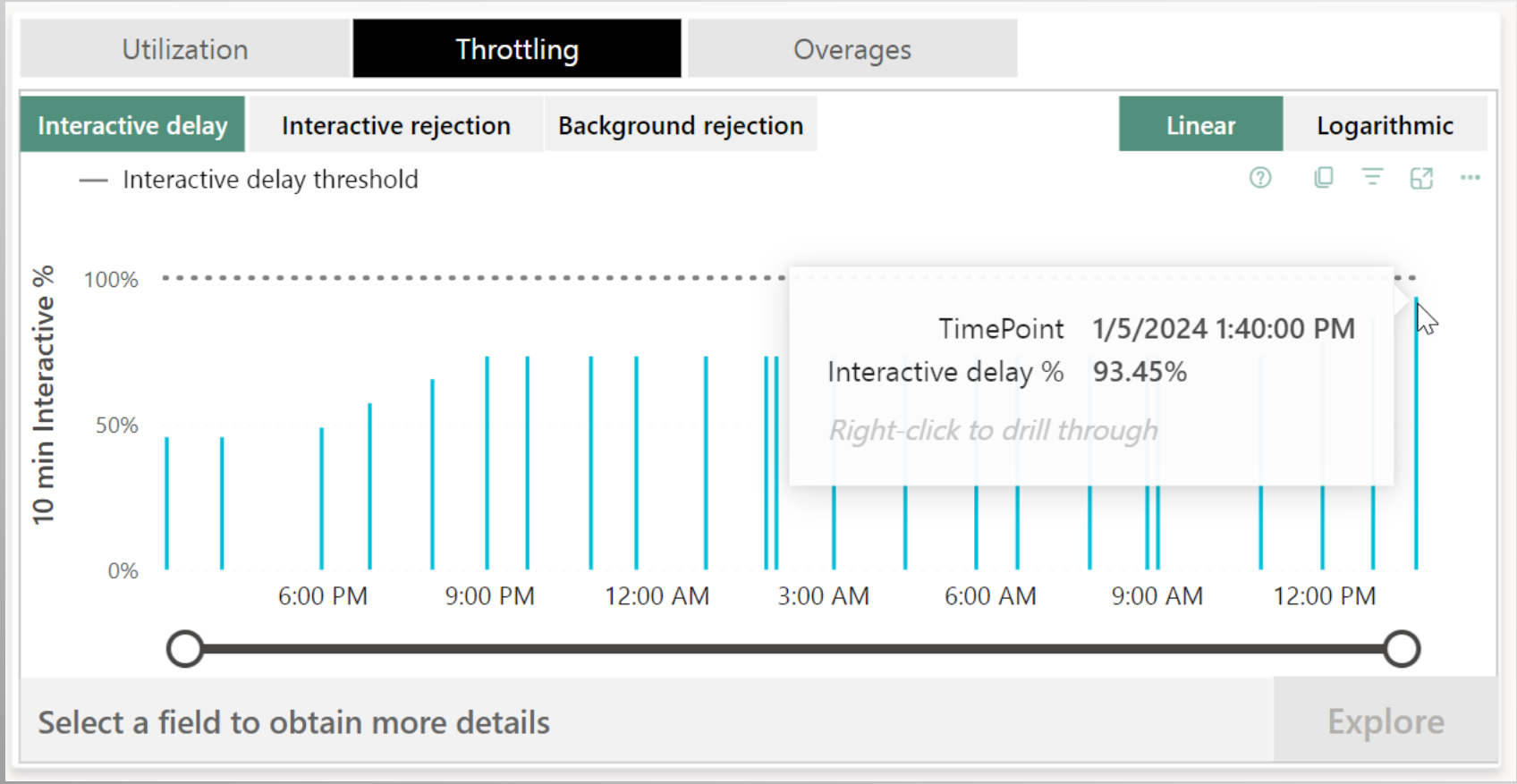
Notebook (CSV to Parquet) Metrics



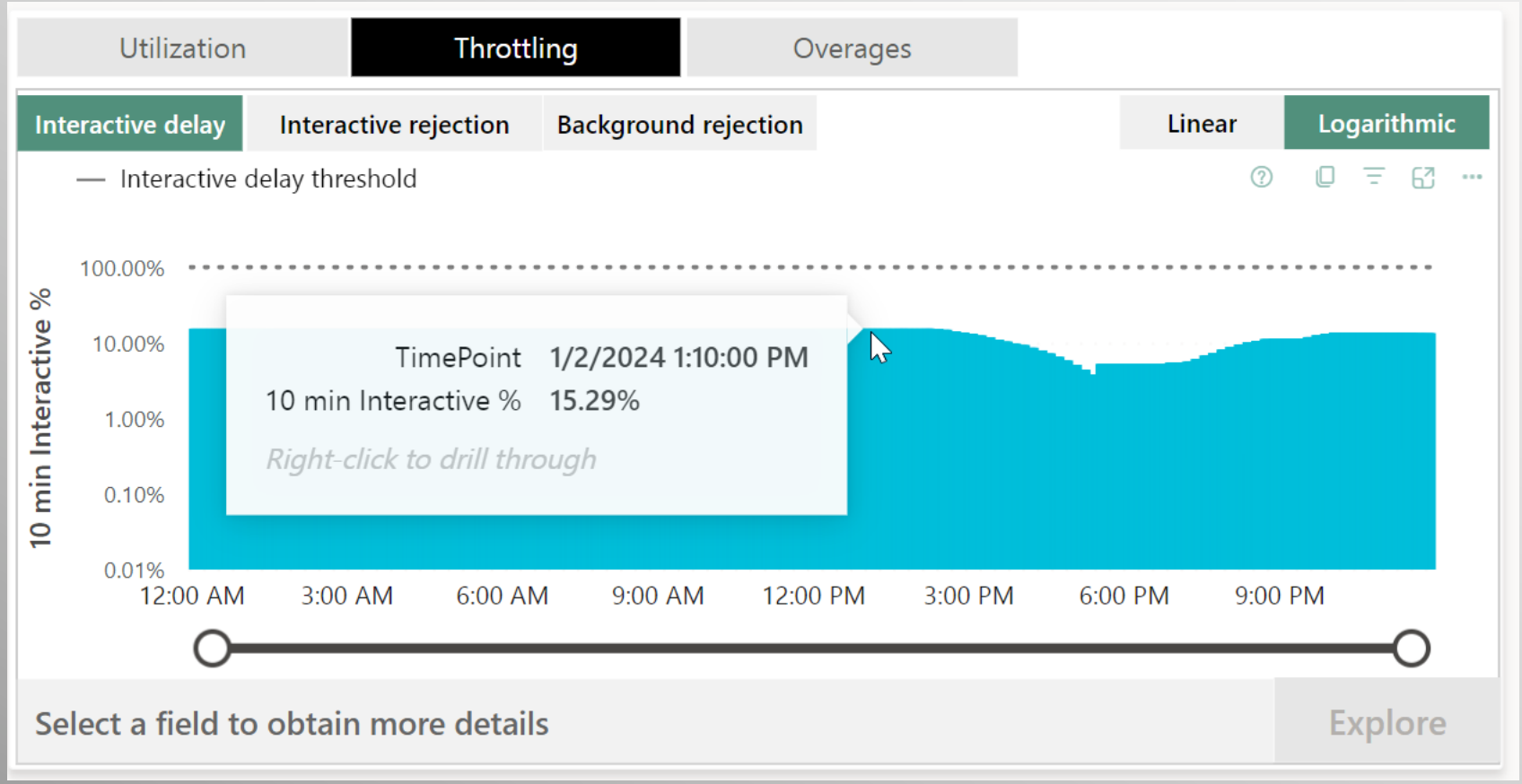
Notebook (CSV to Parquet) Metrics



Notebook (CSV to Parquet) Metrics



Notebook (CSV to Parquet) Metrics



Create a table with some extra colums F2

```python
from pyspark.sql.functions import col, year, month, quarter

table_name = 'fact_orders_2cu'

df = spark.sql("SELECT * FROM Demo_Lakehouse_2CU.raw_orders")
df = df.withColumn('Year', year(col("o_orderdate")))
df = df.withColumn('Quarter', quarter(col("o_orderdate")))
df = df.withColumn('Month', month(col("o_orderdate")))

df.write.mode("overwrite").format("delta").partitionBy("Year","Quarter").save("Tables/" + table_name)
```

✓ 18 min 7 sec -Command executed in 18 min 7 sec 97 ms by Admin on 2:56:34 PM, 1/05/24

s	773,741,822 rows	16.28 GB	23.73 GB
s	773,741,822 rows	23.73 GB	15.85 GB

Create a table with some extra colums F64

```python
from pyspark.sql.functions import col, year, month, quarter

table_name = 'fact_orders'

df = spark.sql("SELECT * FROM FabricLakehouse.raw_orders")
df = df.withColumn('Year', year(col("o_orderdate")))
df = df.withColumn('Quarter', quarter(col("o_orderdate")))
df = df.withColumn('Month', month(col("o_orderdate")))

df.write.mode("overwrite").format("delta").partitionBy("Year","Quarter").save("Tables/" + table_name)
```

✓ 8 min 21 sec -Command executed in 8 min 20 sec 680 ms by Admin on 12:00:16 PM, 1/07/24

s	773,741,822 rows	16.28 GB	23.73 GB
s	773,741,822 rows	23.73 GB	15.85 GB

Join two huge tables F2

```
1   from pyspark.sql.functions import col, year, month, quarter
2
3   table_name = 'fact_orderlines'
4
5   dforder = spark.sql("SELECT * FROM Demo_Lakehouse_2CU.raw_orders")
6   dforder = dforder.withColumn('Year', year(col("o_orderdate")))
7   dforder = dforder.withColumn('Quarter', quarter(col("o_orderdate")))
8   dforder = dforder.withColumn('Month', month(col("o_orderdate")))
9
10  dfline = spark.sql("SELECT * FROM Demo_Lakehouse_2CU.raw_lineitems")
11
12  dforderline = dforder.join(dfline,dforder.O_ORDERKEY == dfline.L_ORDERKEY, how='Inner')
13
14  dforderline.write.mode("overwrite").format("delta").partitionBy("Year","Quarter").save("Tables/" + table_name)
```

Loading...

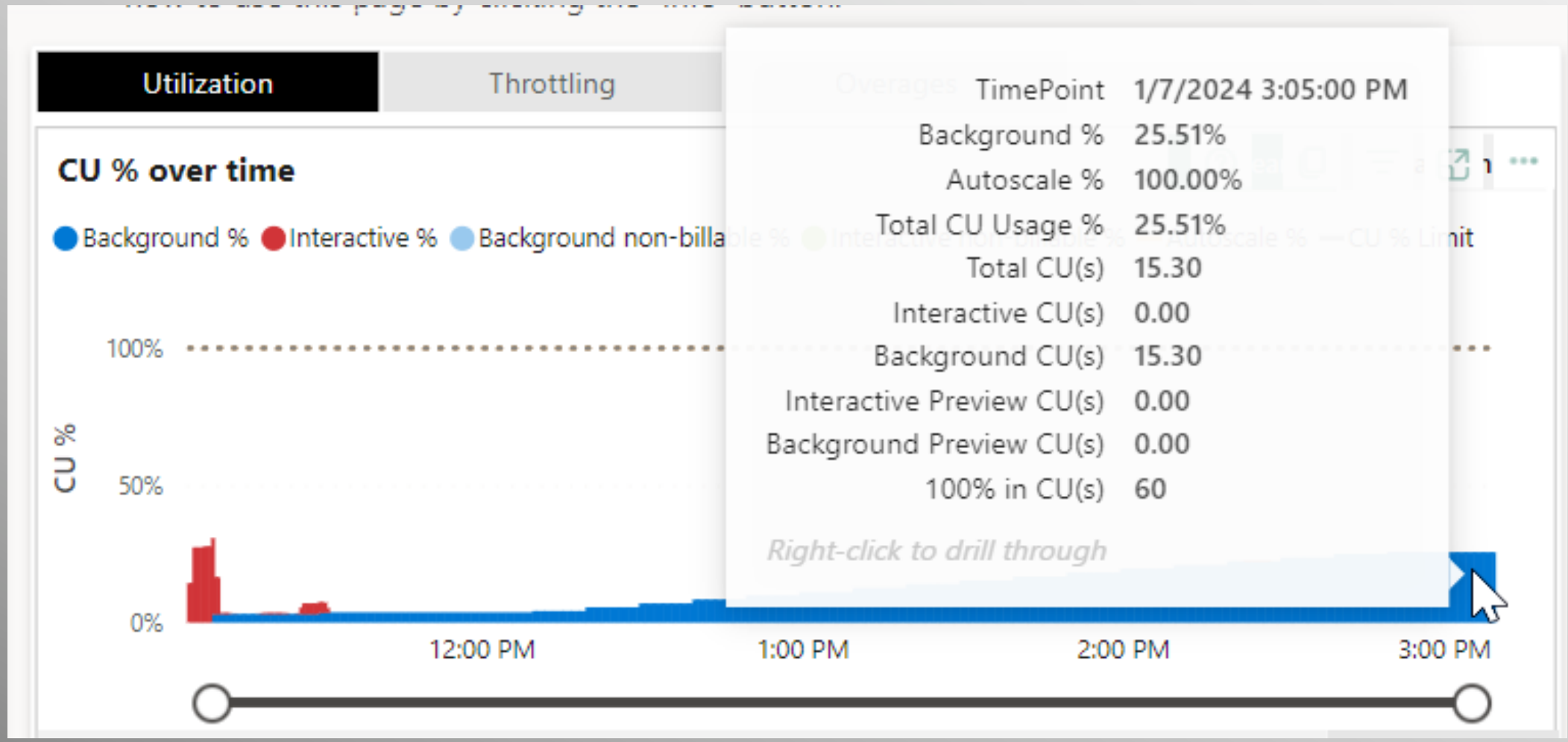
✓ 2 h 16 min -Command executed in 2 h 16 min 23 sec 908 ms by Admin on 2:22:06 PM, 1/07/24

Join two huge tables F64

```
1   from pyspark.sql.functions import col, year, month, quarter
2
3   table_name = 'fact_orderlines'
4
5   dforder = spark.sql("SELECT * FROM FabricLakehouse.raw_orders")
6   dforder = dforder.withColumn('Year', year(col("o_orderdate")))
7   dforder = dforder.withColumn('Quarter', quarter(col("o_orderdate")))
8   dforder = dforder.withColumn('Month', month(col("o_orderdate")))
9
10  dfline = spark.sql("SELECT * FROM FabricLakehouse.raw_lineitems")
11
12  dforderline = dforder.join(dfline,dforder.O_ORDERKEY == dfline.L_ORDERKEY, how='Inner')
13
14  dforderline.write.mode("overwrite").format("delta").partitionBy("Year","Quarter").save("Tables/" + table_name)
```

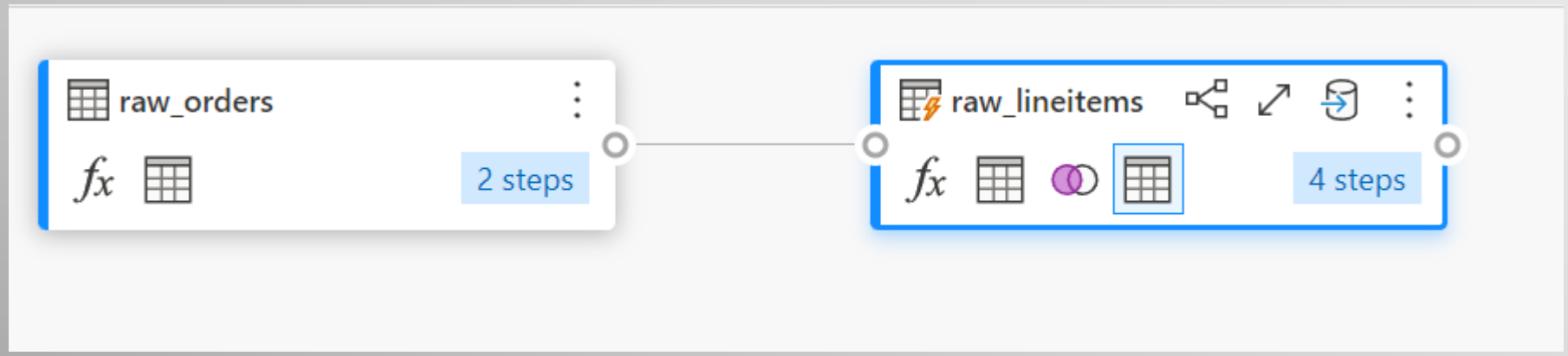
✓ -Apache Spark session ready in 2 min 39 sec 346 ms. Command executed in 14 min 51 sec 480 ms by Admin on 9:33:17 PM, 1/02/24

Join CU requests



Pipeline Gen2

Let’s ….


raw_orders
2 steps
raw_lineitems
4 steps


Applied steps
Source
Navigation 1
Merged queries
Navigation

DataSource.Error: Microsoft SQL: The SQL pool is warming up. Please try again.

Details

Reason = DataSource.Error

DataSourceKind = Lakehouse

DataSourcePath = Lakehouse

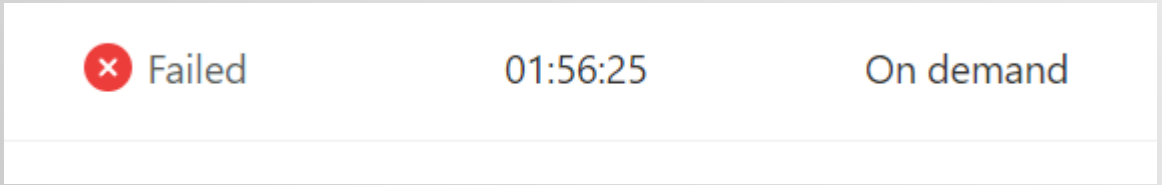
Message = The SQL pool is warming up. Please try again.

ErrorCode = -2146232060

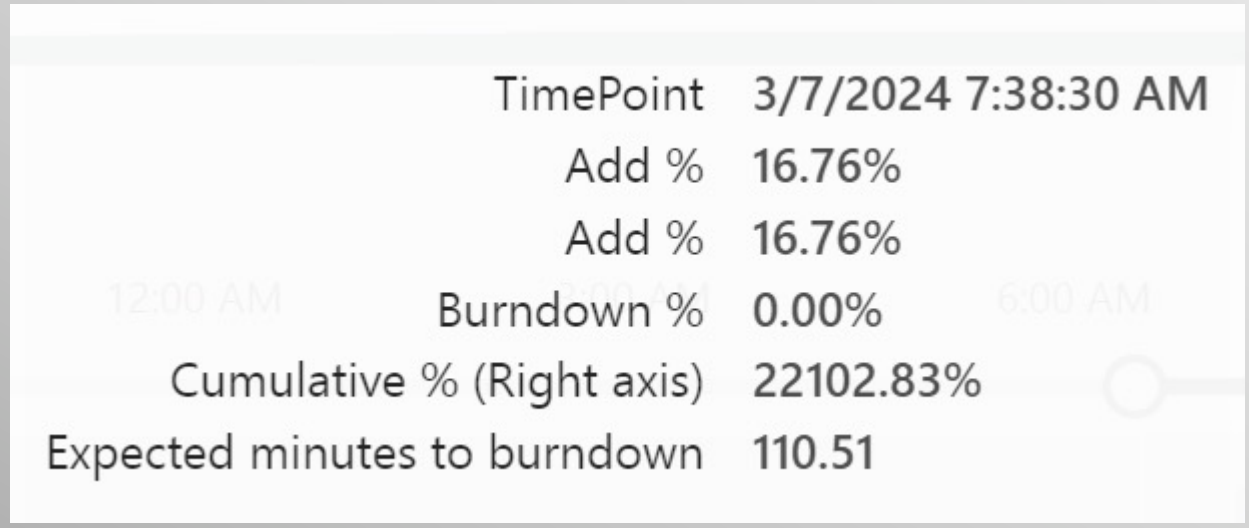
Number = 42109

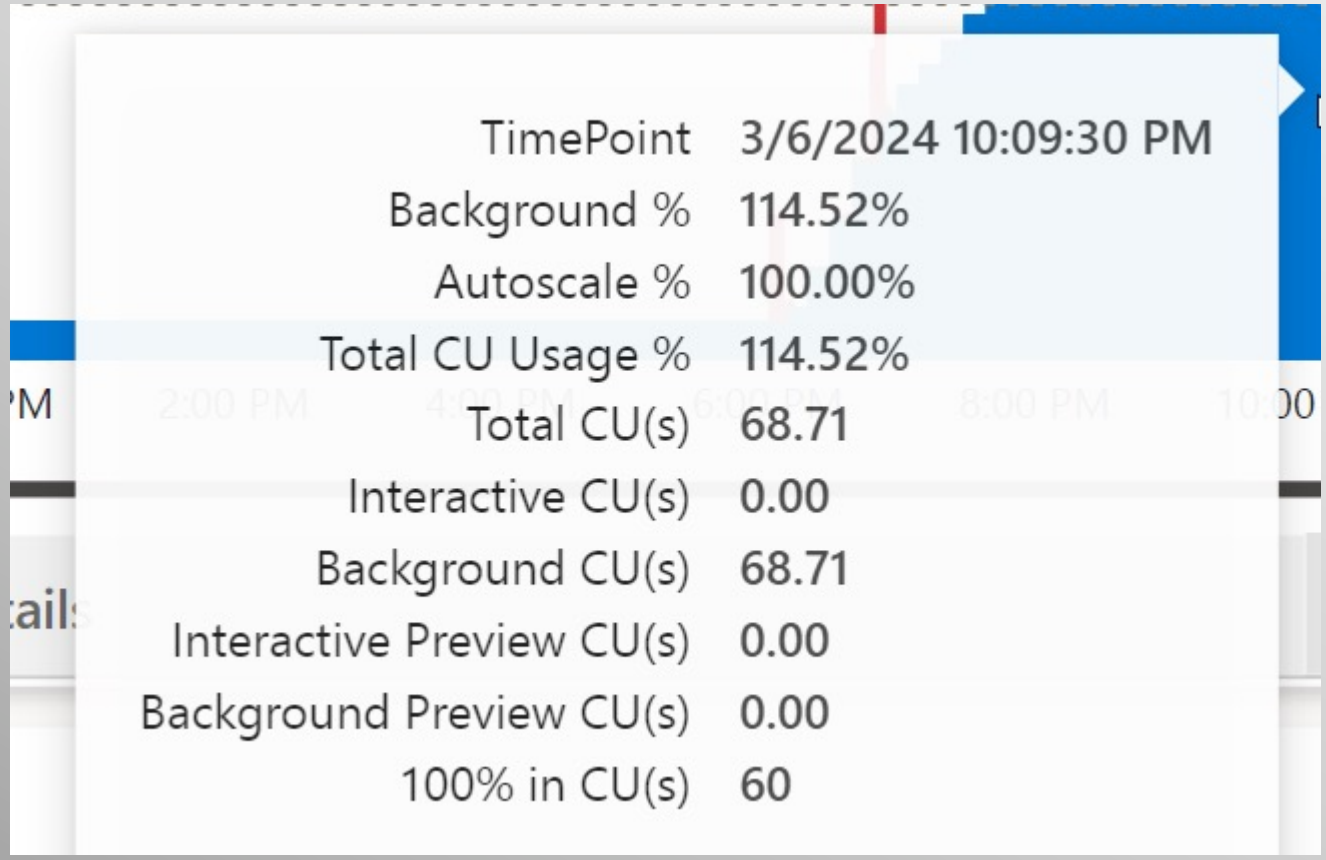
Class = 20

State = 1



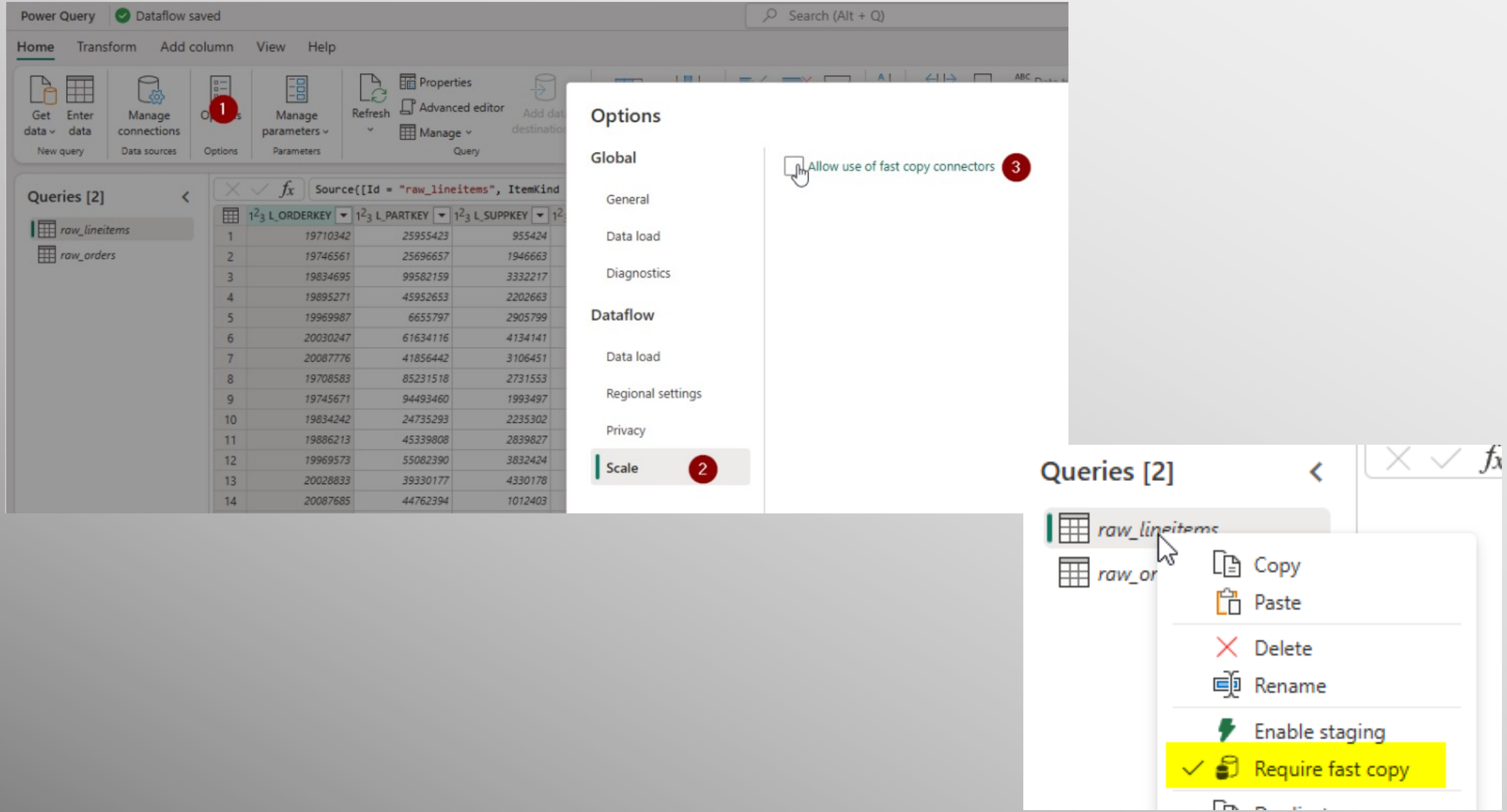
An external error occurred while refreshing the dataflow: Unable to refresh the dataflow because your organization’s Fabric compute capacity has exceeded its limits. Try again later. Learn more at https://aka.ms/capacitymessages


TimePoint 3/7/2024 7:38:30 AM
Add % 16.76%
Add % 16.76%
Burndown % 0.00%
Cumulative % (Right axis) 22102.83%
Expected minutes to burndown 110.51


TimePoint 3/6/2024 10:09:30 PM
Background % 114.52%
Autoscale % 100.00%
Total CU Usage % 114.52%
Total CU(s) 68.71
Interactive CU(s) 0.00
Background CU(s) 68.71
Interactive Preview CU(s) 0.00
Background Preview CU(s) 0.00
100% in CU(s) 60

Pipeline Gen2 Turbo button

Let’s!


Power Query
Dataflow saved
Search (Alt + Q)
Home
Transform
Add column
View
Help
Get data
Enter data
Manage connections
Options
Manage parameters
Refresh
Properties
Advanced editor
Manage
New query
Data sources
Options
Parameters
Query
1
Queries [2]
raw_lineitems
raw_orders
Source{[Id = "raw_lineitems", ItemKind
L_ORDERKEY
L_PARTKEY
L_SUPPKEY
1 19710342 25955423 955424
2 19746561 25696657 1946663
3 19834695 99582159 3332217
4 19895271 45952653 2202663
5 19969987 6655797 2905799
6 20030247 61634116 4134141
7 20087776 41856442 3106451
8 19708583 85231518 2731553
9 19745671 94493460 1993497
10 19834242 24735293 2235302
11 19886213 45339808 2839827
12 19969573 55082390 3832424
13 20028833 39330177 4330178
14 20087685 44762394 1012403
Options
Global
General
Data load
Diagnostics
Dataflow
Data load
Regional settings
Privacy
Scale
2
Allow use of fast copy connectors
3

Queries [2]
raw_lineitems
raw_or
Copy
Paste
Delete
Rename
Enable staging
Require fast copy

Comparison

Without Fast Copy

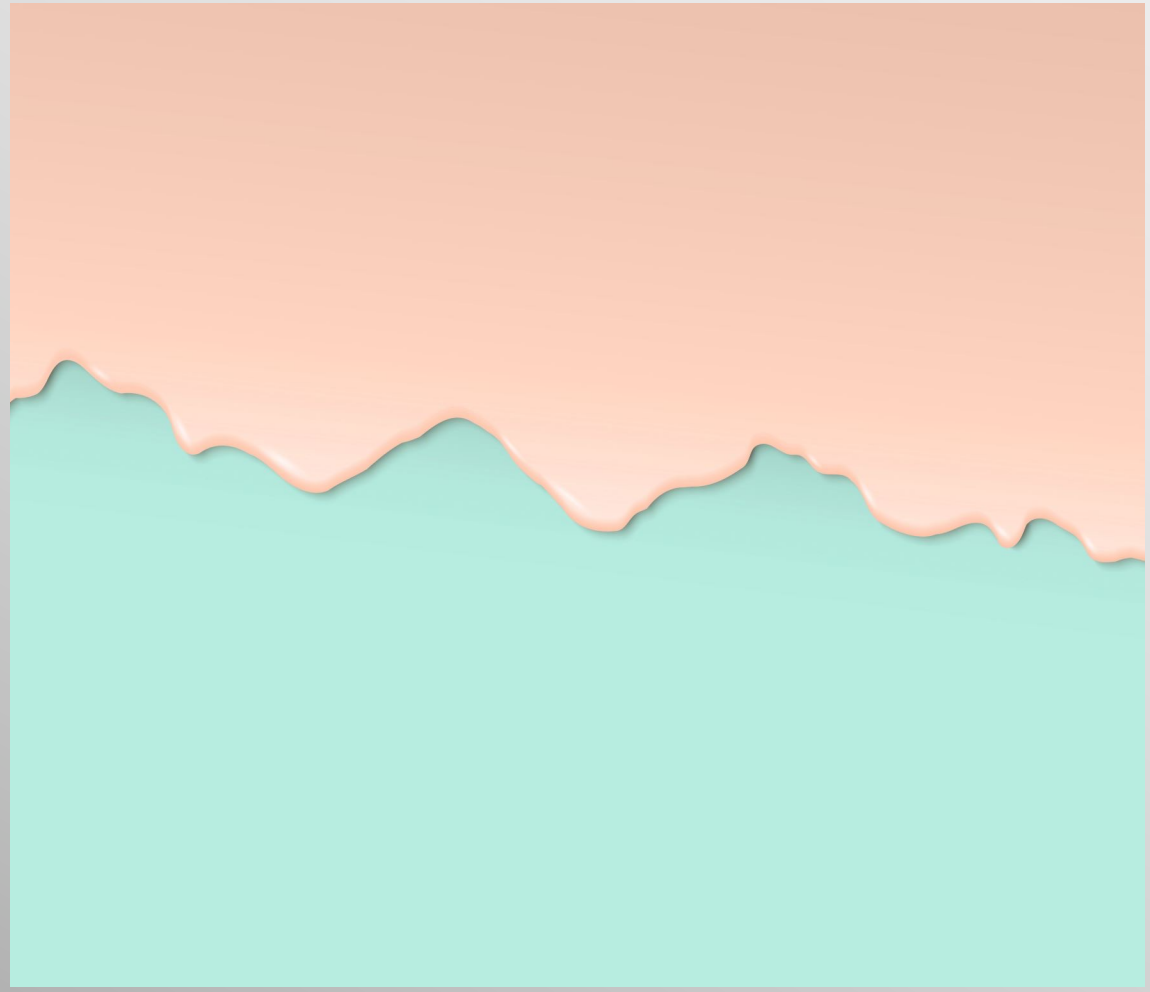
- Ingestion fails
- F2 and F64 fail
- 394 CU’s
- Joining tables fails

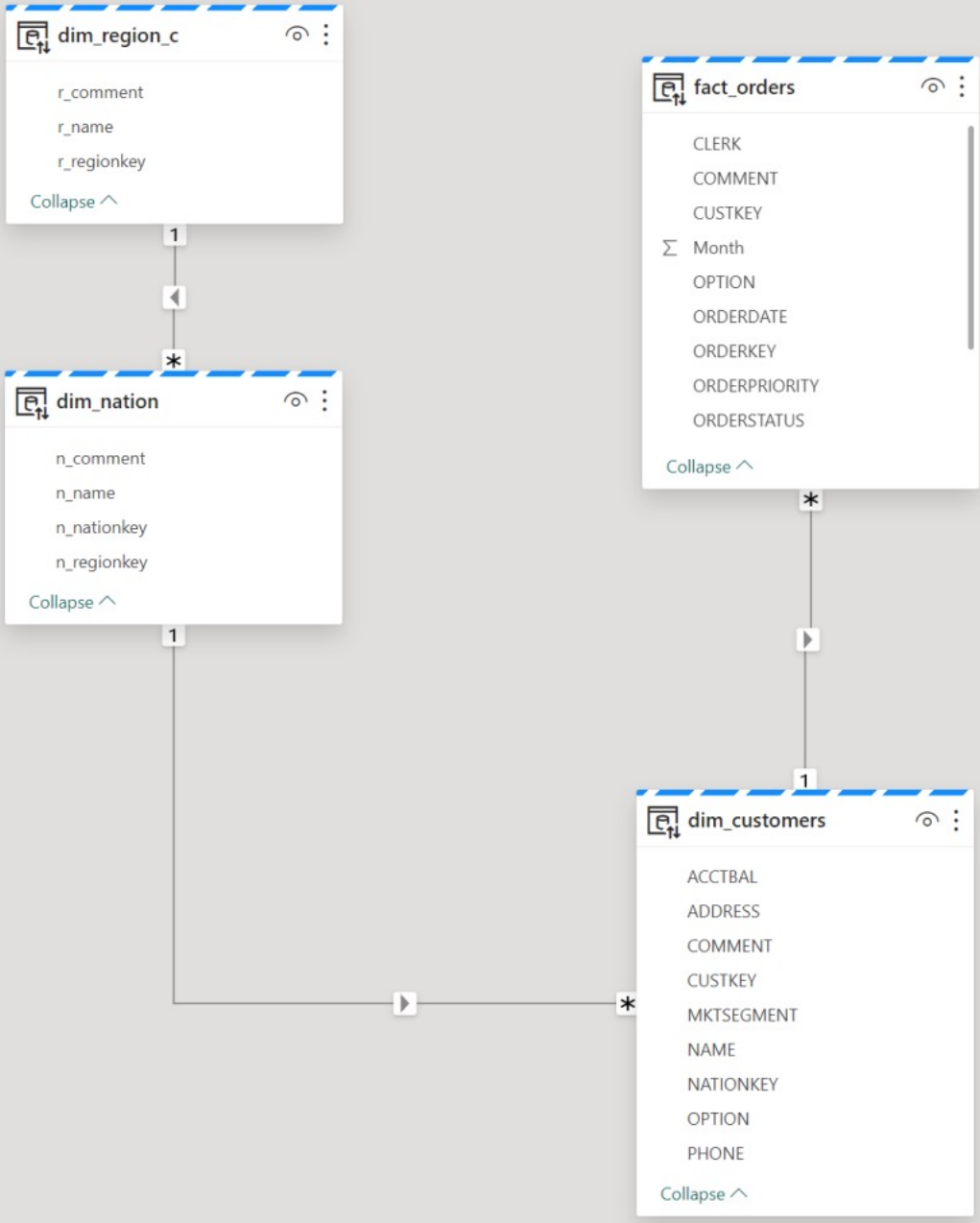
Fast Copy

- Ingestion succeeds
- F64 succeeds
- 78 CU’s
- Joining tables fails

PowerBI

Are you serious?




dim_region_c
r_comment
r_name
r_regionkey
Collapse
1
*
dim_nation
n_comment
n_name
n_nationkey
n_regionkey
Collapse
1
fact_orders
CLERK
COMMENT
CUSTKEY
Month
OPTION
ORDERDATE
ORDERKEY
ORDERPRIORITY
ORDERSTATUS
Collapse
*
1
dim_customers
ACCTBAL
ADDRESS
COMMENT
CUSTKEY
MKTSEGMENT
NAME
NATIONKEY
OPTION
PHONE
Collapse
*

Couldn't load the data for this visual

The resultset of a query to external data source has exceeded the maximum allowed size of '1000000' rows.

Please try again later or contact support. If you contact support, please provide these details.

See details

Contact support

Close

Unable to load model due to reaching capacity limits See details

Unable to load model due to reaching capacity limits

Unable to open this report because your organization's compute capacity has exceeded its limits. Try again later.

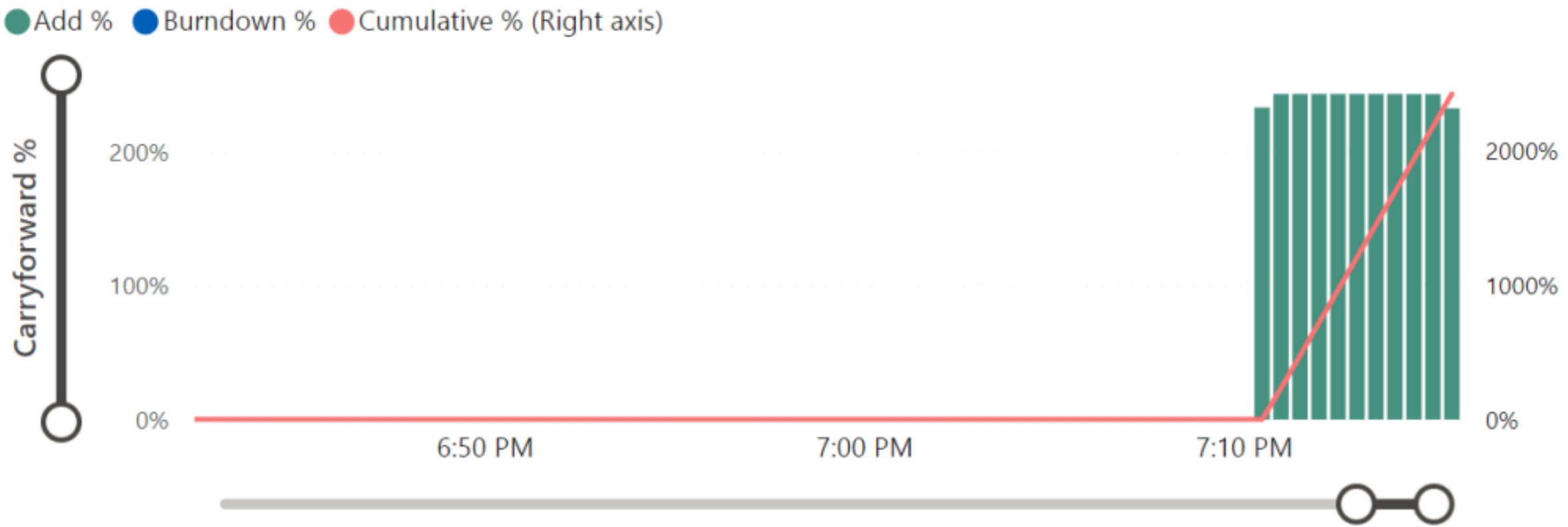
Please try again later or contact support. If you contact support, please provide these details.

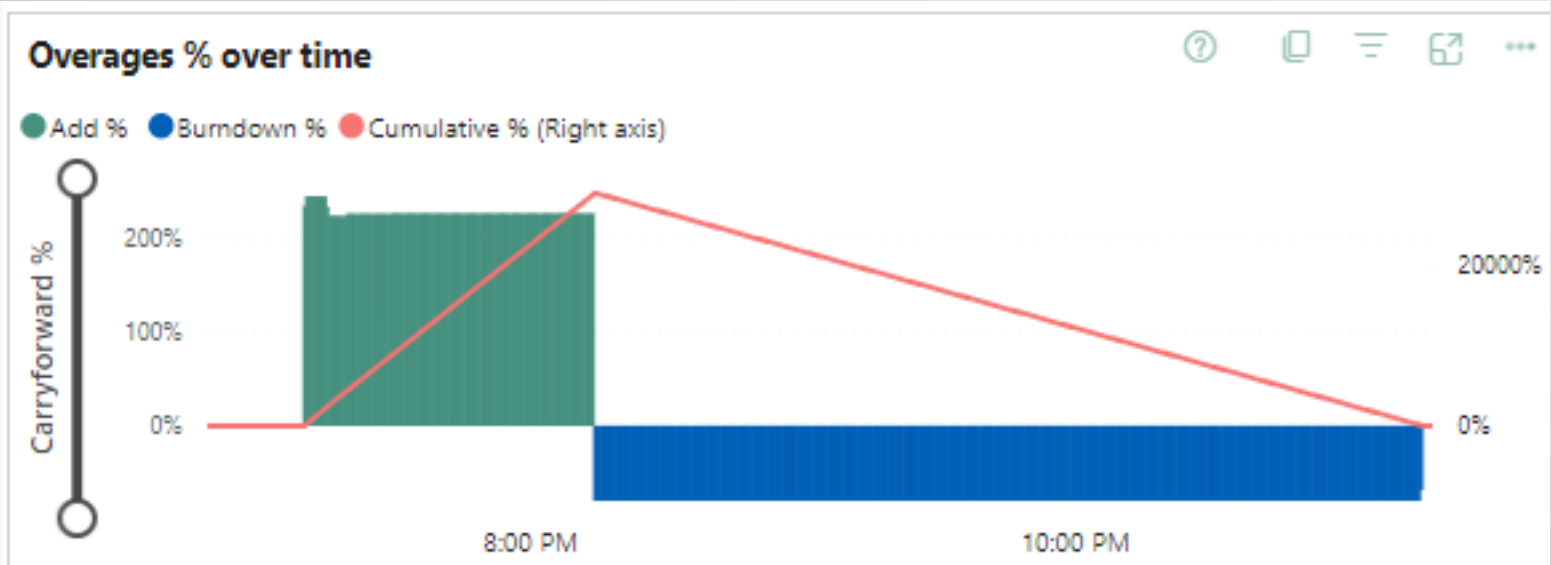
See details

Learn more about solving this issue

Close

Overages % over time




Overages % over time
Add %
Burndown %
Cumulative % (Right axis)
Carryforward %
200%
100%
0%
20000%
0%
8:00 PM
10:00 PM

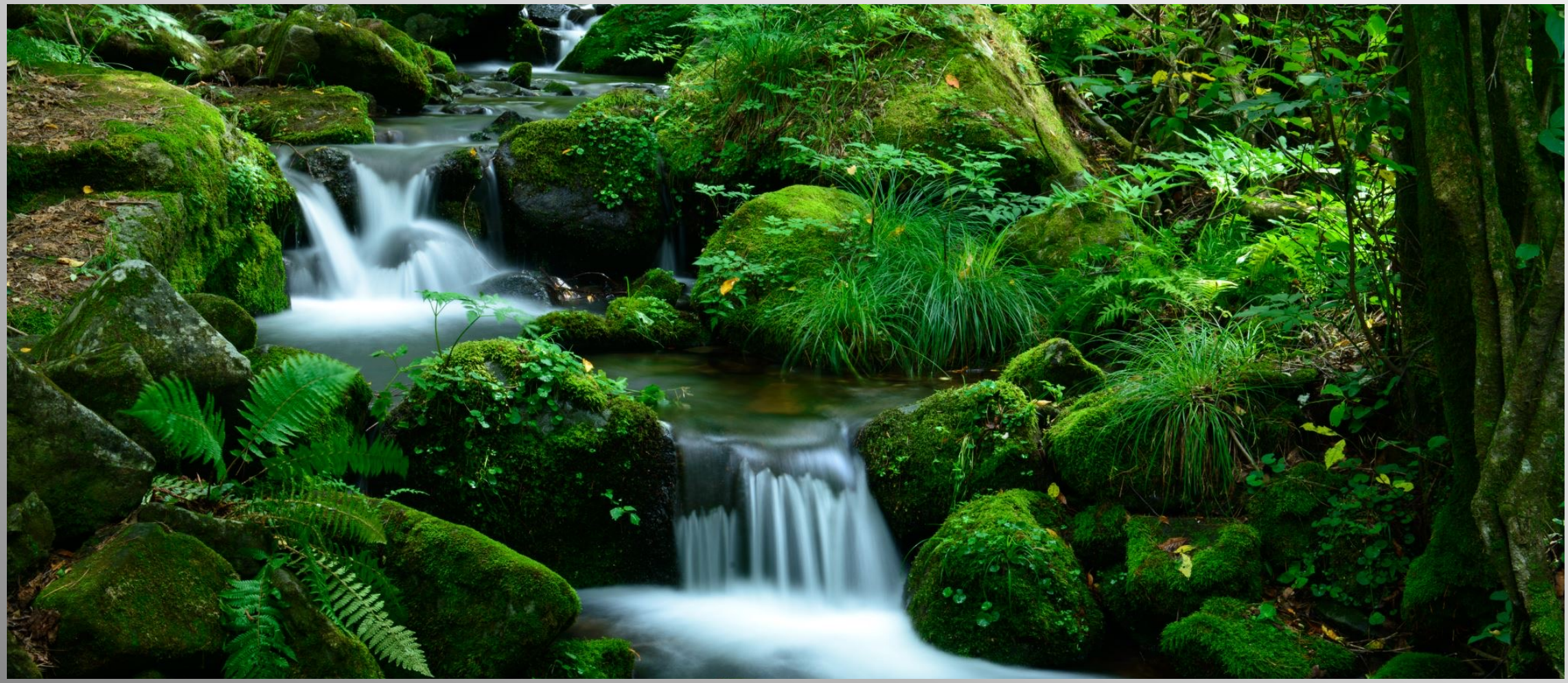
Faster burndown

Can I have some more CU’s please?

TimePoint	2/28/2024 9:43:30 PM
Burndown %	-42.48%
Add %	0.00%
Burndown %	-42.48%
Cumulative % (Right axis)	1406.49%
Expected minutes to burndown	7.03

TimePoint	2/28/2024 9:44:00 PM
Burndown %	-85.62%
Add %	0.00%
Burndown %	-85.62%
Cumulative % (Right axis)	341.00%
Expected minutes to burndown	1.71

Summaries



Notebook Metrics overview

TableName	Rows	F2	F64	Factor	CU(s)
Pipeline Ingestion	n/a	32 minutes 39 seconds	33 minutes 14 seconds	1x	43
Customers	120 million	2 minutes 29 seconds	1 minute 13 seconds	2x	56
LineItems	4.2 billion	2 hours, 4 minutes 55 seconds	7 minutes 46 seconds	17x	82
Fact_orders	773 million	18 minutes	8 minutes	2.25x	26
Order_lines	4.2 billion	2 hours 16 minutes	14 minutes 51 seconds	9x	15
Sum		**~5 hours**	**~65 minutes**	**5x**	**222**

Fabric licenses

SKU	Capacity Units (CU)	Power BI SKU	Power BI v-cores	Spark Cores
F2	2	-	0.25	4
F4	4	-	0.5	8
F8	8	EM/A1	1	16
F16	16	EM2/A2	2	32
F32	32	EM3/A3	4	64
F64	64	P1/A4	8	128

Results

- Install the Microsoft Fabric capacity metrics app - Microsoft Fabric | Microsoft Learn
- Test your workload as the CU(s) are like Sql Server Query costs
- Compare numbers and review duration and CU(s)

Resources

- TPC-H
  - https://www.tpc.org/tpch/
  - https://github.com/dimitri/tpch-citus/blob/master/schema/tpch-schema.sql
  - https://docs.deistercloud.com/content/Databases.30/TPCH%20Benchmark.90/index.xml?embedded=true
  - https://support.hpe.com/hpesc/public/docDisplay?docId=sf000078704en_us&docLocale=en_US

Resources

- Fabric Capacity App
  - https://gennakerdata.com/2023/12/12/capacity-metrics-app-analysis-english-version/
  - https://learn.microsoft.com/en-us/fabric/enterprise/metrics-app-install?tabs=1st
  - https://blog.fabric.microsoft.com/en-us/blog/capacity-platform-updates-for-pause-resume-and-capacity-metrics-for-copilot-and-vnet-gateways/

Thank you for attending!

Any questions?