PSCONF.EU

PowerShell Conference Europe

When Memory Fights Back

Jonas Sommer Nielsen

MALMÖ 25

Many thanks to our sponsors:

PSCONF.EU

PowerShell Conference Europe

When Memory Fights Back

Jonas Sommer Nielsen

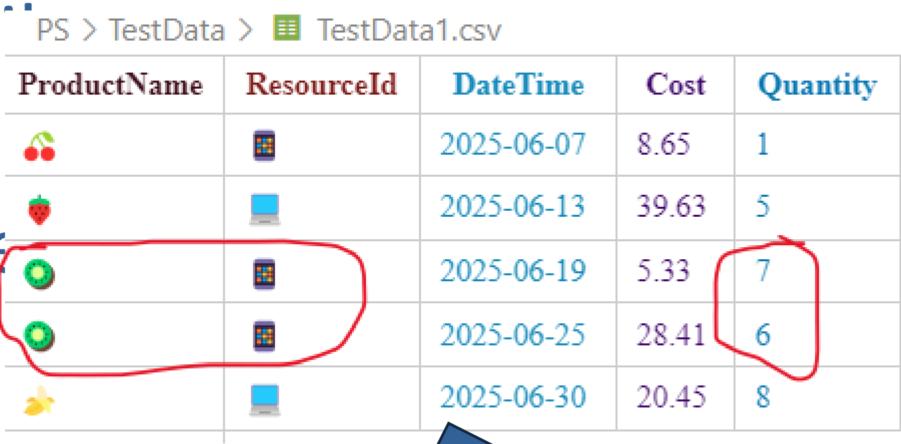
MALMÖ 25

Jonas Sommer Nielsen

🦋 @mrhvid.bsky.social 𝕏 🤷‍♂️

Task

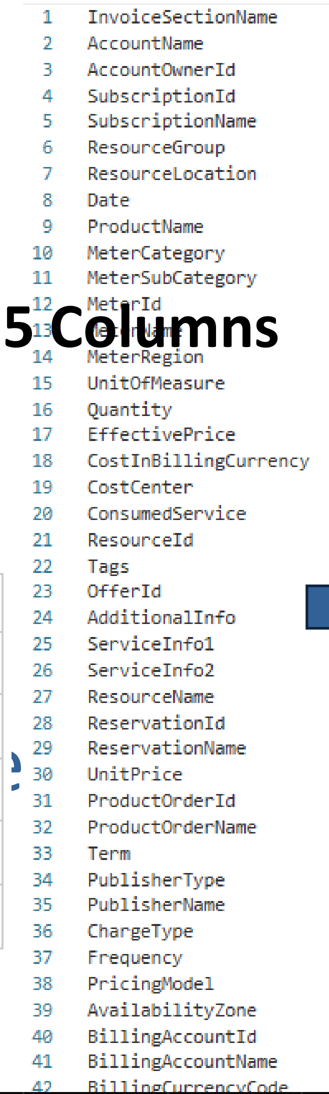
- Reduce size of Cost report
  - +3 GB .csv f
  - +3.3 million
  - Average line   mns
- Automate it



PS > TestData > TestData1.csv

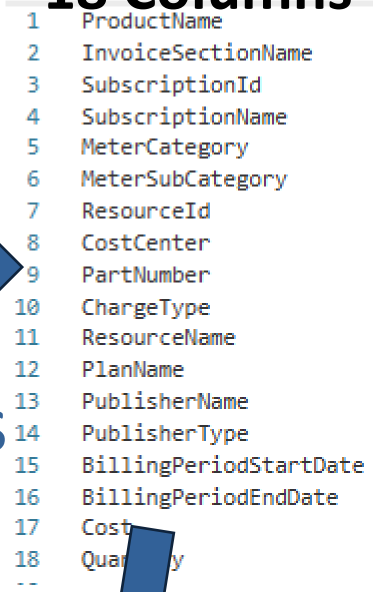
ProductName	ResourceId	DateTime	Cost	Quantity
🍒	📱	2025-06-07	8.65	1
🍓	💻	2025-06-13	39.63	5
🥝	📱	2025-06-19	5.33	7
🥝	📱	2025-06-25	28.41	6
🍌	💻	2025-06-30	20.45	8

55 Columns

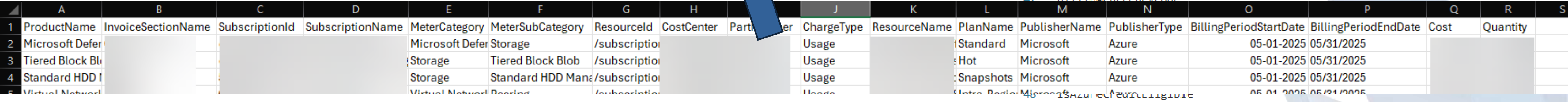


```
1   InvoiceSectionName
2   AccountName
3   AccountOwnerId
4   SubscriptionId
5   SubscriptionName
6   ResourceGroup
7   ResourceLocation
8   Date
9   ProductName
10  MeterCategory
11  MeterSubCategory
12  MeterId
13  MeterName
14  MeterRegion
15  UnitOfMeasure
16  Quantity
17  EffectivePrice
18  CostInBillingCurrency
19  CostCenter
20  ConsumedService
21  ResourceId
22  Tags
23  OfferId
24  AdditionalInfo
25  ServiceInfo1
26  ServiceInfo2
27  ResourceName
28  ReservationId
29  ReservationName
30  UnitPrice
31  ProductOrderId
32  ProductOrderName
33  Term
34  PublisherType
35  PublisherName
36  ChargeType
37  Frequency
38  PricingModel
39  AvailabilityZone
40  BillingAccountId
41  BillingAccountName
42  BillingCurrencyCode
48  IsAzureCreditEligible
49  PartNumber
50  PayGPrice
51  PlanName
52  ServiceFamily
53  CostAllocationRuleName
54  benefitId
55  benefitName
```

18 Columns



```
1   ProductName
2   InvoiceSectionName
3   SubscriptionId
4   SubscriptionName
5   MeterCategory
6   MeterSubCategory
7   ResourceId
8   CostCenter
9   PartNumber
10  ChargeType
11  ResourceName
12  PlanName
13  PublisherName
14  PublisherType
15  BillingPeriodStartDate
16  BillingPeriodEndDate
17  Cost
18  Quantity
```



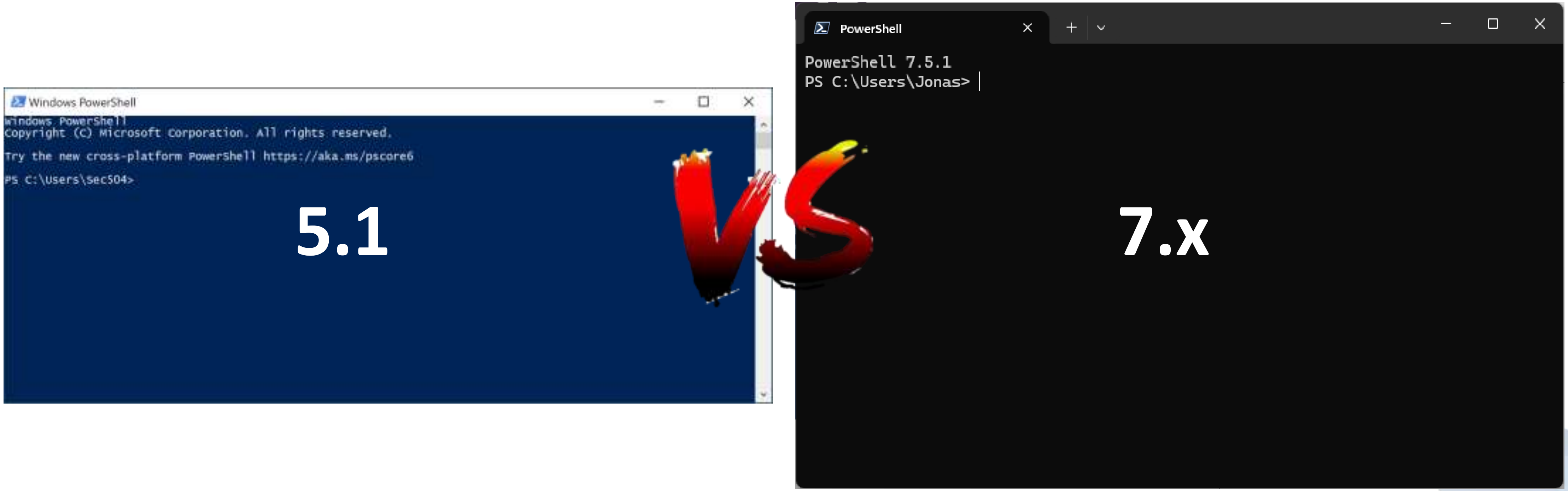
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	ProductName	InvoiceSectionName	SubscriptionId	SubscriptionName	MeterCategory	MeterSubCategory	ResourceId	CostCenter	Part er	ChargeType	ResourceName	PlanName	PublisherName	PublisherType	BillingPeriodStartDate	BillingPeriodEndDate	Cost	Quantity
2	Microsoft Defer	[redacted]	[redacted]	[redacted]	Microsoft Defer	Storage	/subscriptio	[redacted]	[redacted]	Usage	[redacted]	Standard	Microsoft	Azure	05-01-2025	05/31/2025	[redacted]	[redacted]
3	Tiered Block Bl	[redacted]	[redacted]	[redacted]	Storage	Tiered Block Blob	/subscriptio	[redacted]	[redacted]	Usage	[redacted]	Hot	Microsoft	Azure	05-01-2025	05/31/2025	[redacted]	[redacted]
4	Standard HDD I	[redacted]	[redacted]	[redacted]	Storage	Standard HDD Mana	/subscriptio	[redacted]	[redacted]	Usage	[redacted]	Snapshots	Microsoft	Azure	05-01-2025	05/31/2025	[redacted]	[redacted]
5	Virtual Networl	[redacted]	[redacted]	[redacted]	Virtual Networl	Peering	/subscriptio	[redacted]	[redacted]	Usage	[redacted]	Intra-Regio	Microsoft	Azure	05-01-2025	05/31/2025	[redacted]	[redacted]

@mrhvid.bsky.social

Demo

”Import-csv | **group-by** | export-csv”

Reminder

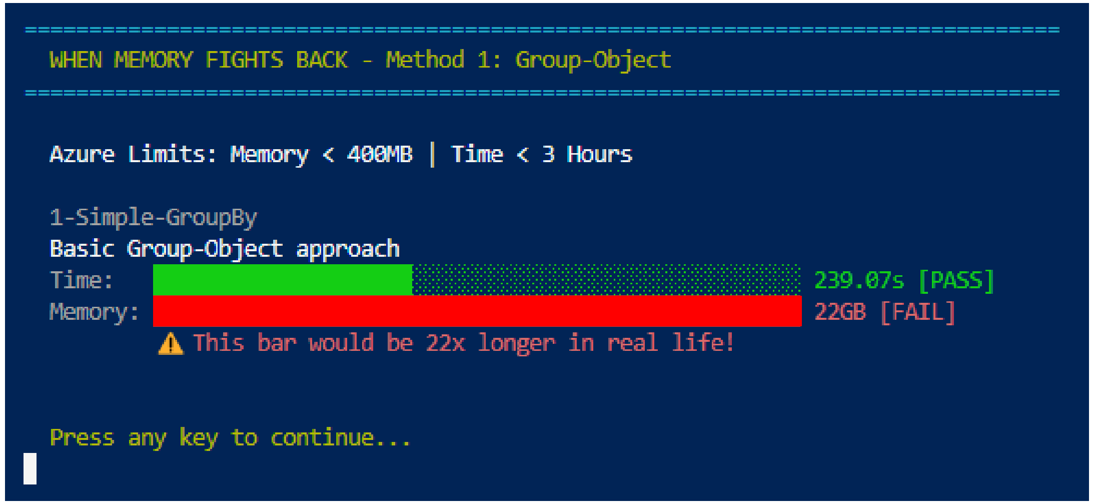


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Sec504>
```

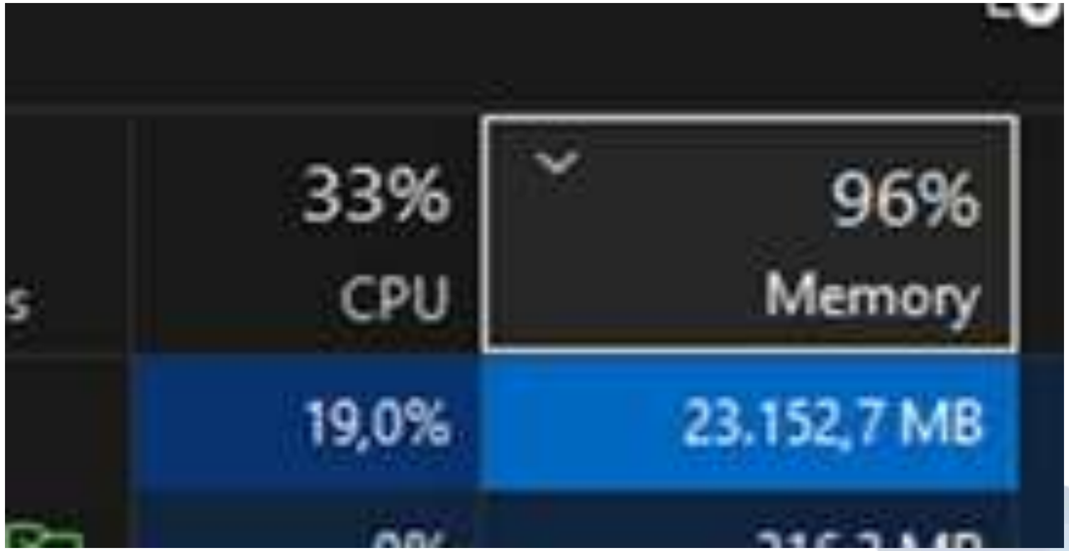
```
PowerShell 7.5.1
PS C:\Users\Jonas>
```

First try – locally

”Import-csv | group-by | export-csv”



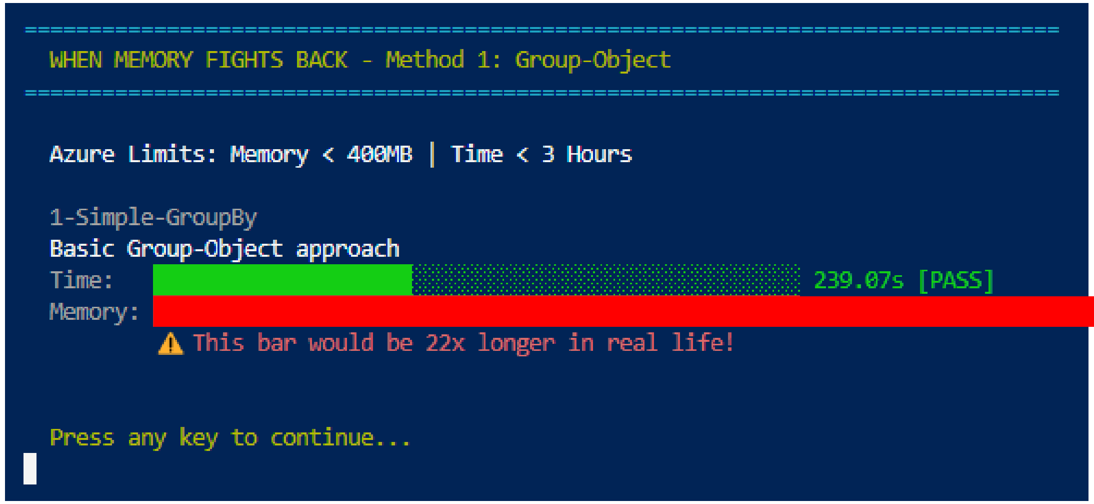
```
==================================================================================
  WHEN MEMORY FIGHTS BACK - Method 1: Group-Object
==================================================================================

  Azure Limits: Memory < 400MB | Time < 3 Hours

  1-Simple-GroupBy
  Basic Group-Object approach
  Time:    ████████████████████░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░ 239.07s [PASS]
  Memory:  ██████████████████████████████████████████████████ 22GB [FAIL]
           ⚠ This bar would be 22x longer in real life!


  Press any key to continue...
▌
```



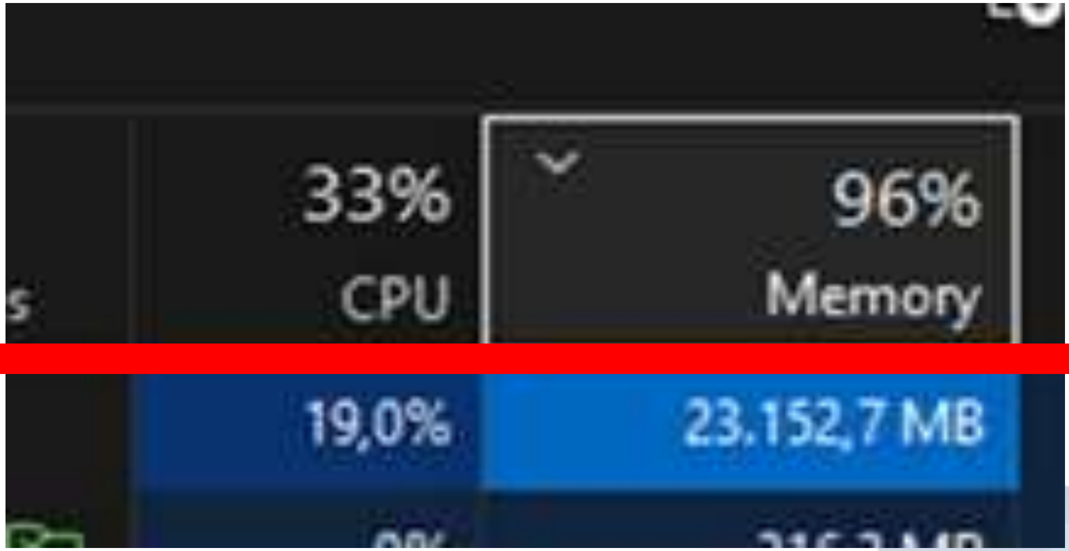
CPU	Memory
33%	96%
19,0%	23.152,7 MB

First try – locally

”Import-csv | group-by | export-csv”



```
==================================================================================
  WHEN MEMORY FIGHTS BACK - Method 1: Group-Object
==================================================================================

  Azure Limits: Memory < 400MB | Time < 3 Hours

  1-Simple-GroupBy
  Basic Group-Object approach
  Time:                                                          239.07s [PASS]
  Memory:
          ⚠ This bar would be 22x longer in real life!


  Press any key to continue...
```



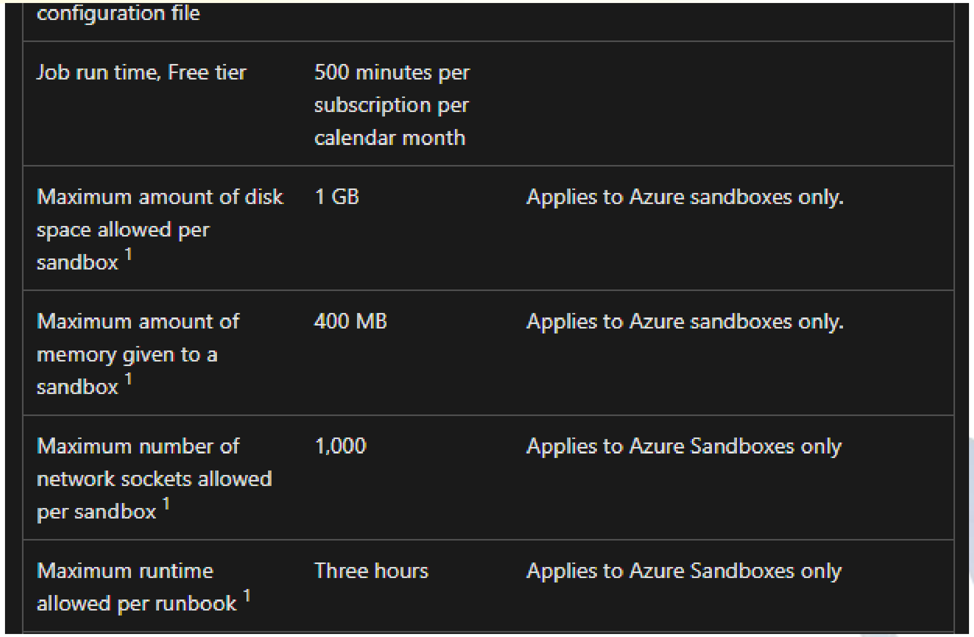
33% CPU	96% Memory
19,0%	23.152,7 MB

KEEP
CALM
AND
AUTOMATE

WHEN MEMORY

Fights Back

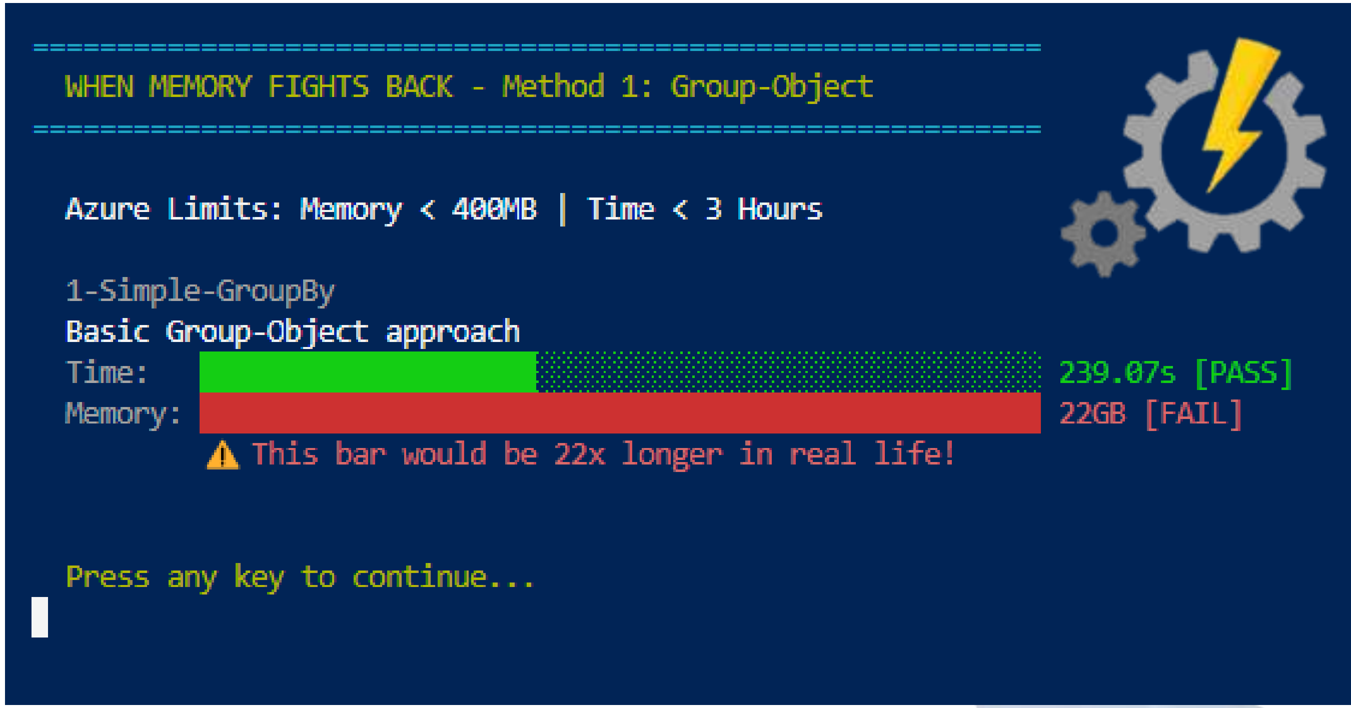
- 1 GB disk space
- 400 MB memory
- 3 Hours runtime



configuration file		
Job run time, Free tier	500 minutes per subscription per calendar month	
Maximum amount of disk space allowed per sandbox [1]	1 GB	Applies to Azure sandboxes only.
Maximum amount of memory given to a sandbox [1]	400 MB	Applies to Azure sandboxes only.
Maximum number of network sockets allowed per sandbox [1]	1,000	Applies to Azure Sandboxes only
Maximum runtime allowed per runbook [1]	Three hours	Applies to Azure Sandboxes only

https://learn.microsoft.com/en-us/azure/automation/automation-subscription-limits-faq

GROUPER vs runbook requirements



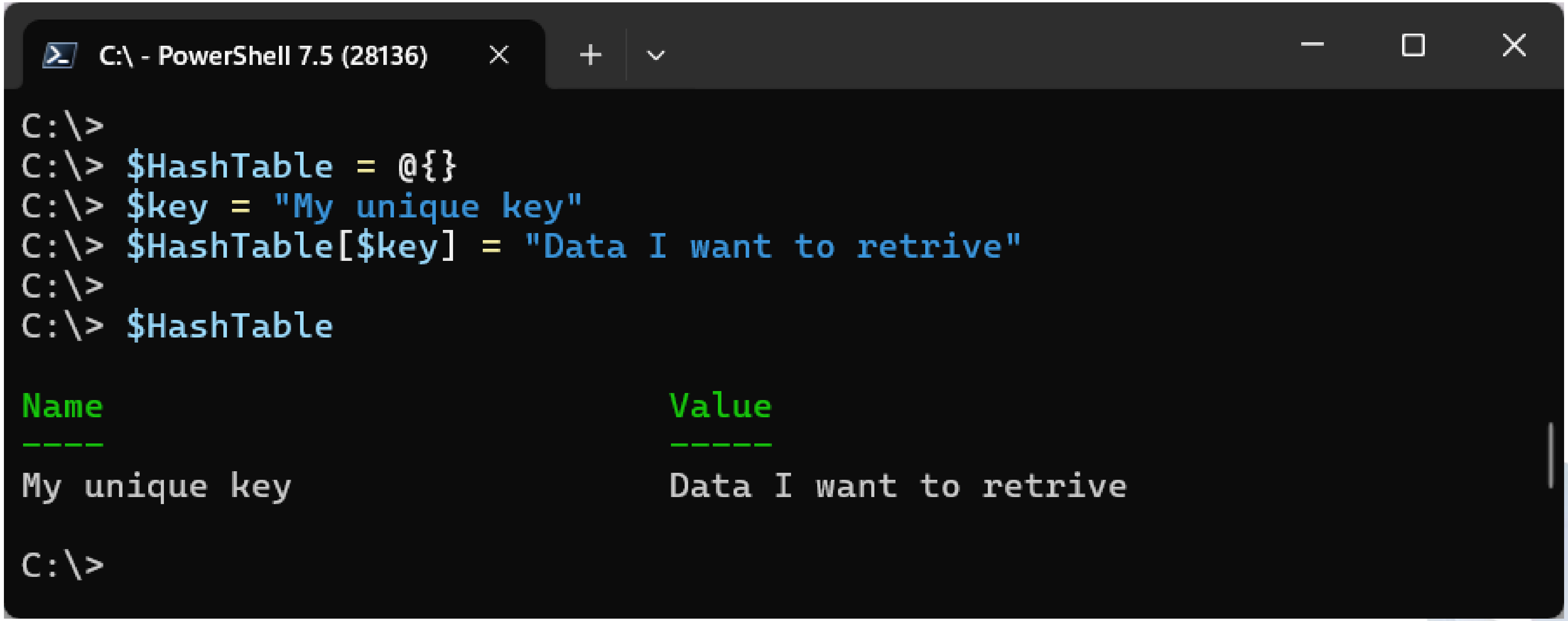
```
===========================================================
  WHEN MEMORY FIGHTS BACK - Method 1: Group-Object
===========================================================

  Azure Limits: Memory < 400MB | Time < 3 Hours

  1-Simple-GroupBy
  Basic Group-Object approach
  Time:    ██████████░░░░░░░░░░░░░░░░  239.07s [PASS]
  Memory:  ██████████████████████████  22GB [FAIL]
           ⚠ This bar would be 22x longer in real life!


  Press any key to continue...
█
```

CONTINUE? 7

MY STRENGTH IS MUCH
GREATER THAN YOURS.

Level 2. Hash table

Import-csv | “hash table” | export-csv



```
C:\>
C:\> $HashTable = @{}
C:\> $key = "My unique key"
C:\> $HashTable[$key] = "Data I want to retrive"
C:\>
C:\> $HashTable

Name                           Value
----                           -----
My unique key                  Data I want to retrive

C:\>
```

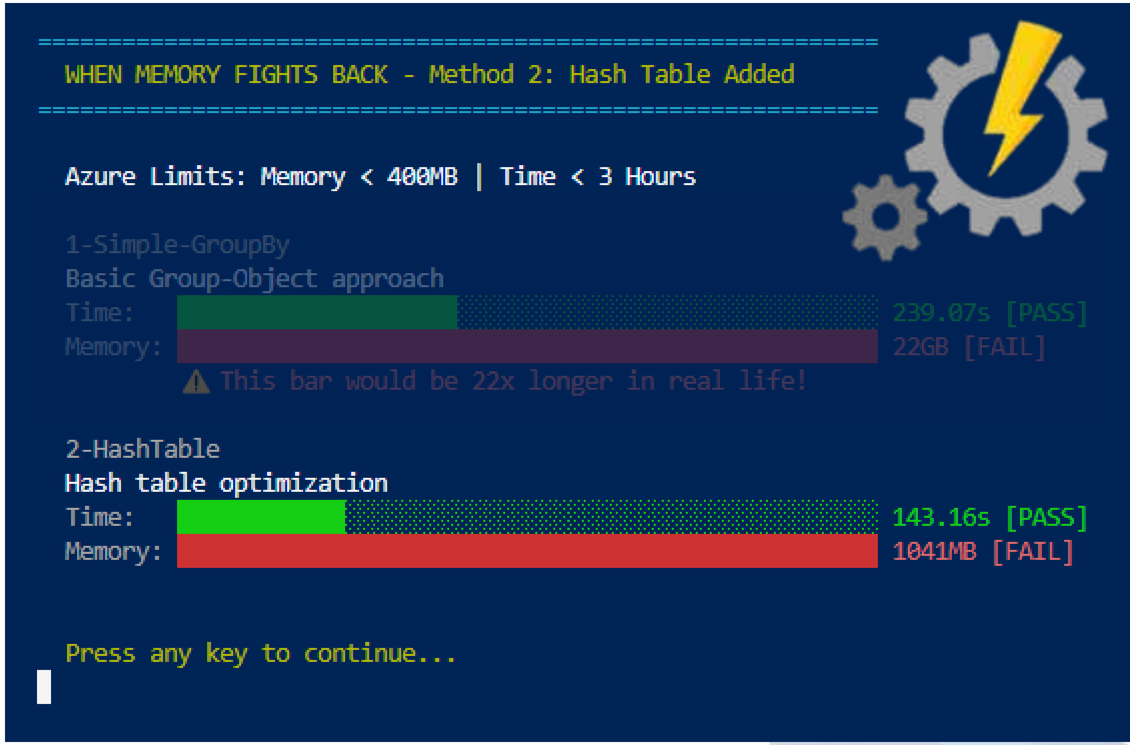
@mrhvid.bsky.social

Demos

”Import-csv | **“hash table”** | export-csv”

@mrhvid.bsky.social

HASHER vs runbook requirements

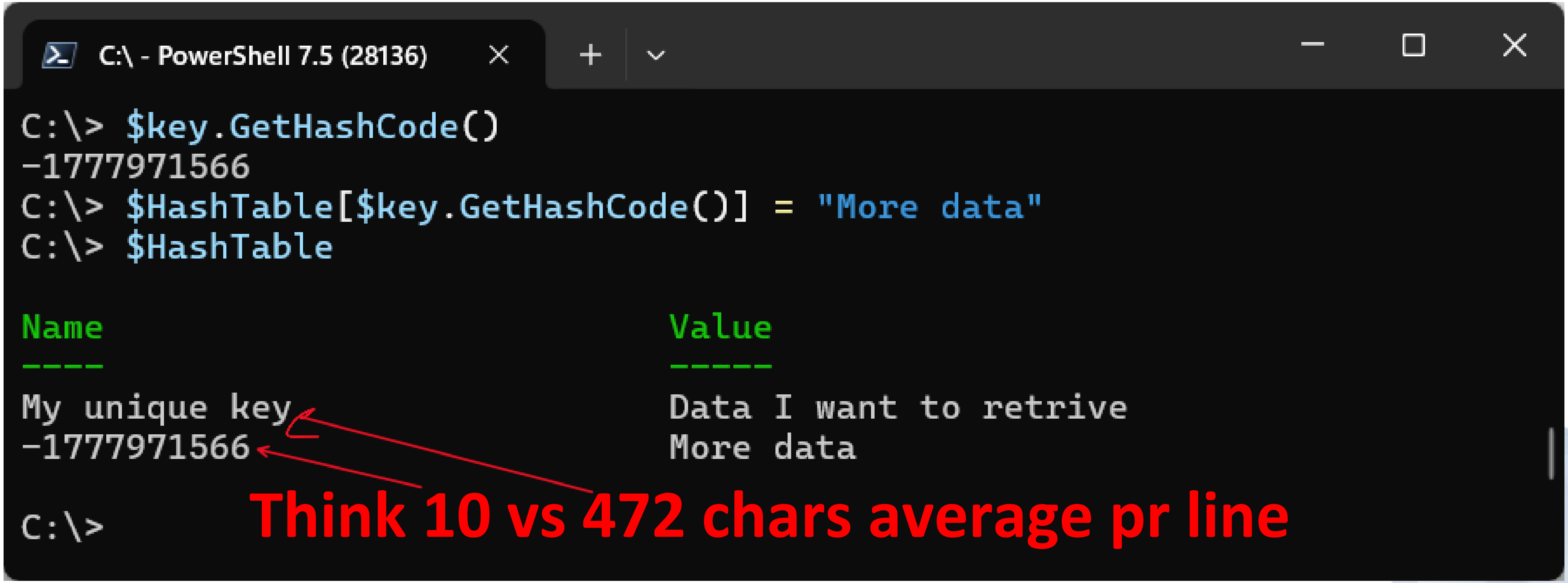


```
=============================================================================
  WHEN MEMORY FIGHTS BACK - Method 2: Hash Table Added
=============================================================================

  Azure Limits: Memory < 400MB | Time < 3 Hours

  1-Simple-GroupBy
  Basic Group-Object approach
  Time:                                                  239.07s [PASS]
  Memory:                                                22GB [FAIL]
          ⚠ This bar would be 22x longer in real life!

  2-HashTable
  Hash table optimization
  Time:                                                  143.16s [PASS]
  Memory:                                                1041MB [FAIL]


  Press any key to continue...
```

@mrhvid.bsky.social

Hash the key + temp.csv

Import-Csv | “hash table” + temp.csv
Import-Csv temp.csv | Add Q & C from hash table | Export-Csv



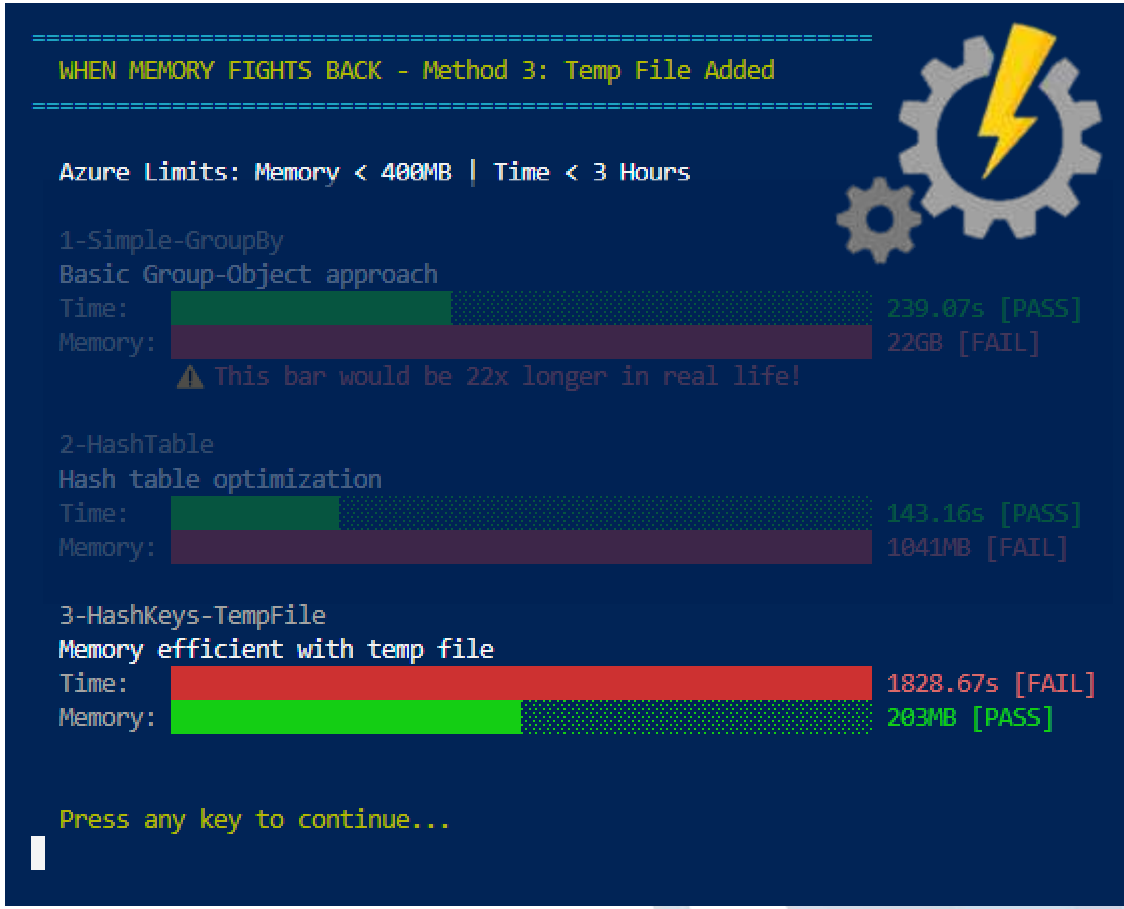
```
C:\> $key.GetHashCode()
-1777971566
C:\> $HashTable[$key.GetHashCode()] = "More data"
C:\> $HashTable

Name                           Value
----                           -----
My unique key                  Data I want to retrive
-1777971566                    More data

C:\>
```

Think 10 vs 472 chars average pr line

Demo

” Import-csv | **hash Keys + “hash table” + temp.csv** |… | export-csv””

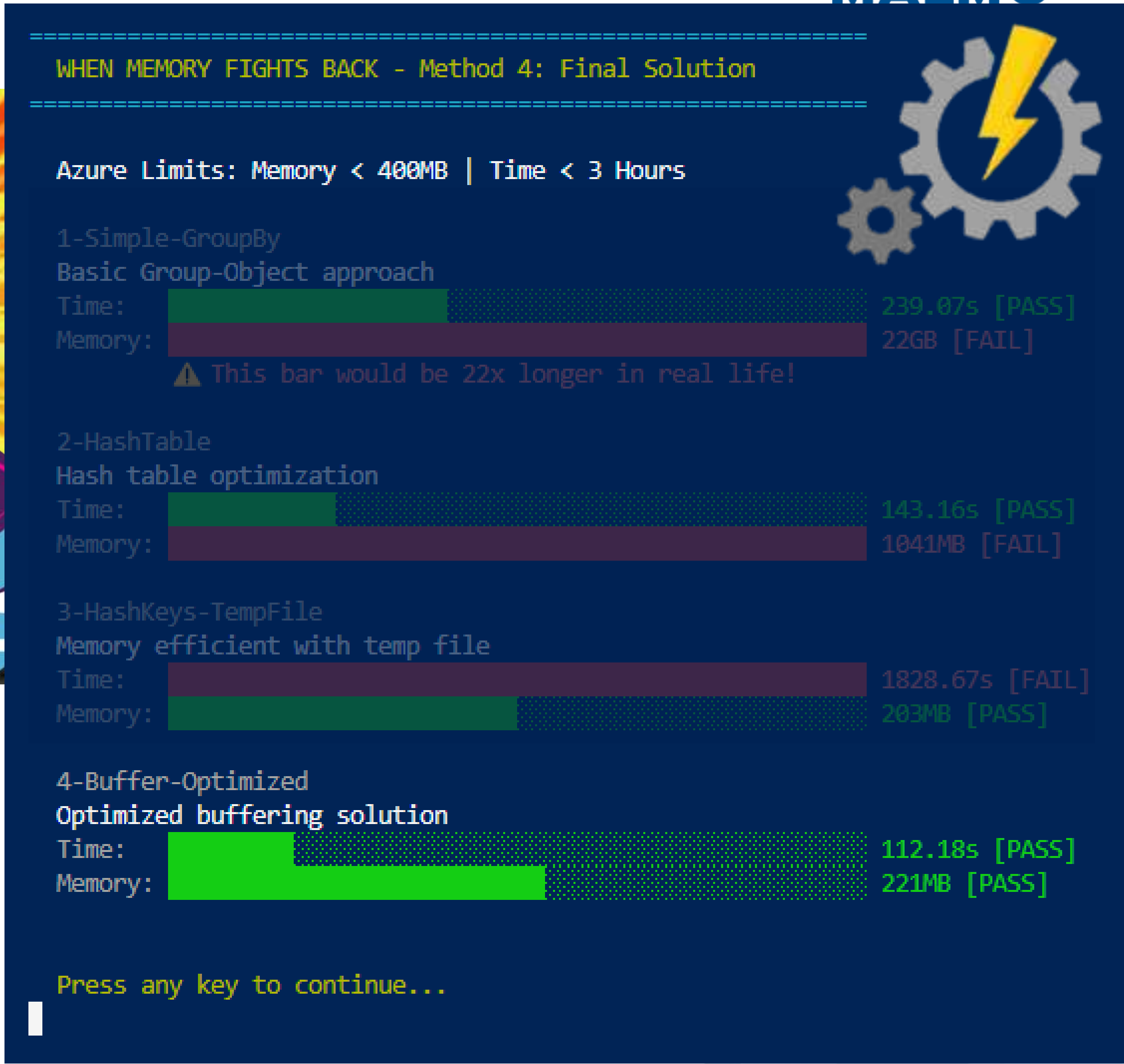
TEMPER vs runbook requirements



```
============================================================
  WHEN MEMORY FIGHTS BACK - Method 3: Temp File Added
============================================================

  Azure Limits: Memory < 400MB | Time < 3 Hours

  1-Simple-GroupBy
  Basic Group-Object approach
  Time:    ████████████████████░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░ 239.07s [PASS]
  Memory:  ██████████████████████████████████████████████████ 22GB [FAIL]
           ⚠ This bar would be 22x longer in real life!

  2-HashTable
  Hash table optimization
  Time:    ████████████░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░░ 143.16s [PASS]
  Memory:  ██████████████████████████████████████████████████ 1041MB [FAIL]

  3-HashKeys-TempFile
  Memory efficient with temp file
  Time:    ██████████████████████████████████████████████████ 1828.67s [FAIL]
  Memory:  █████████████████████████░░░░░░░░░░░░░░░░░░░░░░░░░ 203MB [PASS]


  Press any key to continue...
█
```

Demo

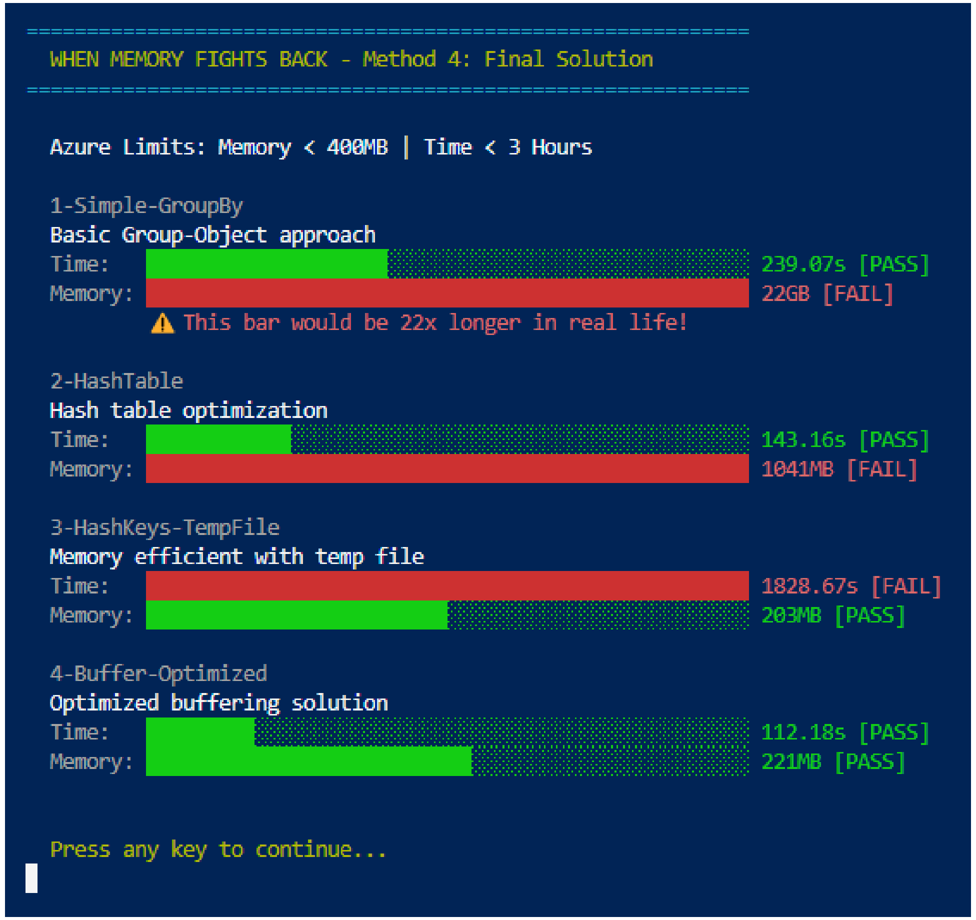
”Same + **buffer**”



```
=========================================================================
  WHEN MEMORY FIGHTS BACK - Method 4: Final Solution
=========================================================================

  Azure Limits: Memory < 400MB | Time < 3 Hours

  1-Simple-GroupBy
  Basic Group-Object approach
  Time:                                                       239.07s [PASS]
  Memory:                                                     22GB [FAIL]
          ⚠ This bar would be 22x longer in real life!

  2-HashTable
  Hash table optimization
  Time:                                                       143.16s [PASS]
  Memory:                                                     1041MB [FAIL]

  3-HashKeys-TempFile
  Memory efficient with temp file
  Time:                                                       1828.67s [FAIL]
  Memory:                                                     203MB [PASS]

  4-Buffer-Optimized
  Optimized buffering solution
  Time:                                                       112.18s [PASS]
  Memory:                                                     221MB [PASS]


  Press any key to continue...
```

@mrhvid.bsky.social 🤷

The Journey: From 22GB to 221MB



```
=================================================================
  WHEN MEMORY FIGHTS BACK - Method 4: Final Solution
=================================================================

  Azure Limits: Memory < 400MB | Time < 3 Hours

  1-Simple-GroupBy
  Basic Group-Object approach
  Time:                                                        239.07s [PASS]
  Memory:                                                      22GB [FAIL]
          ⚠ This bar would be 22x longer in real life!

  2-HashTable
  Hash table optimization
  Time:                                                        143.16s [PASS]
  Memory:                                                      1041MB [FAIL]

  3-HashKeys-TempFile
  Memory efficient with temp file
  Time:                                                        1828.67s [FAIL]
  Memory:                                                      203MB [PASS]

  4-Buffer-Optimized
  Optimized buffering solution
  Time:                                                        112.18s [PASS]
  Memory:                                                      221MB [PASS]


  Press any key to continue...
```

- 99% memory reduction
- Problem solving process
  - Taskmanager
  - Measure-command
  - Profiler module
- Often no magic needed
- Know the basics

Q&A

15 minutes

🦋 @mrhvid.bsky.social 𝕏 🤷