

# Troubleshoot Data Partitioning Bottlenecks



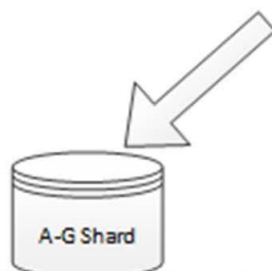


# Partitioning

- **Partitioning** – Process of physically dividing data into separate data store
- **Why Partitioning**
  - **Improve** – Scalability, Security and availability
  - **Reduce** – Contention
  - **Optimize** – Performance
- **Three strategies of partitioning**
  - Horizontal partitioning
  - Vertical partitioning
  - Functional partitioning

## Horizontal Partitioning

Key	Name	Description	Stock	Price	LastOrdered
ARC1	Arc welder	250 Amps	8	119.00	25-Nov-2013
BRK8	Bracket	250mm	46	5.66	18-Nov-2013
BRK9	Bracket	400mm	82	6.98	1-Jul-2013
HOS8	Hose	1/2"	27	27.50	18-Aug-2013
WGT4	Widget	Green	16	13.99	3-Feb-2013
WGT6	Widget	Purple	76	13.99	31-Mar-2013



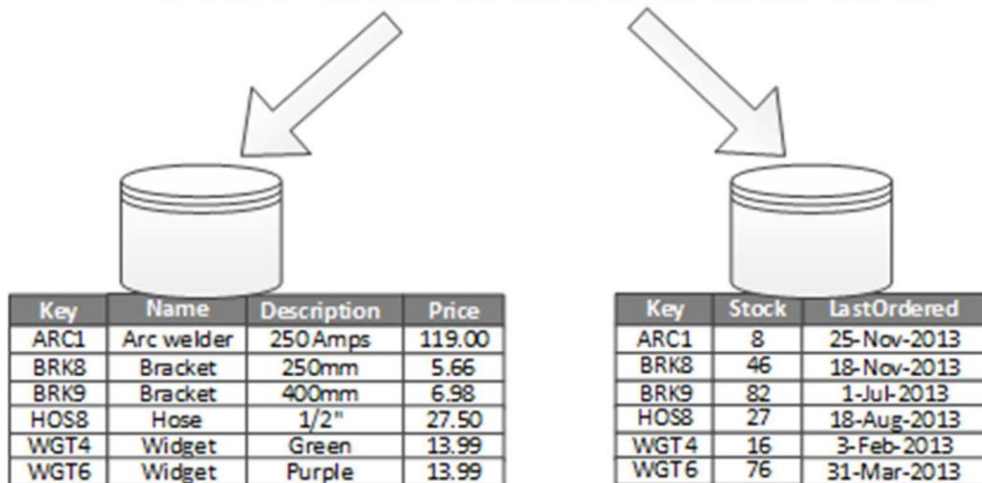
Key	Name	Description	Stock	Price	LastOrdered
ARC1	Arc welder	250 Amps	8	119.00	25-Nov-2013
BRK8	Bracket	250mm	46	5.66	18-Nov-2013
BRK9	Bracket	400mm	82	6.98	1-Jul-2013



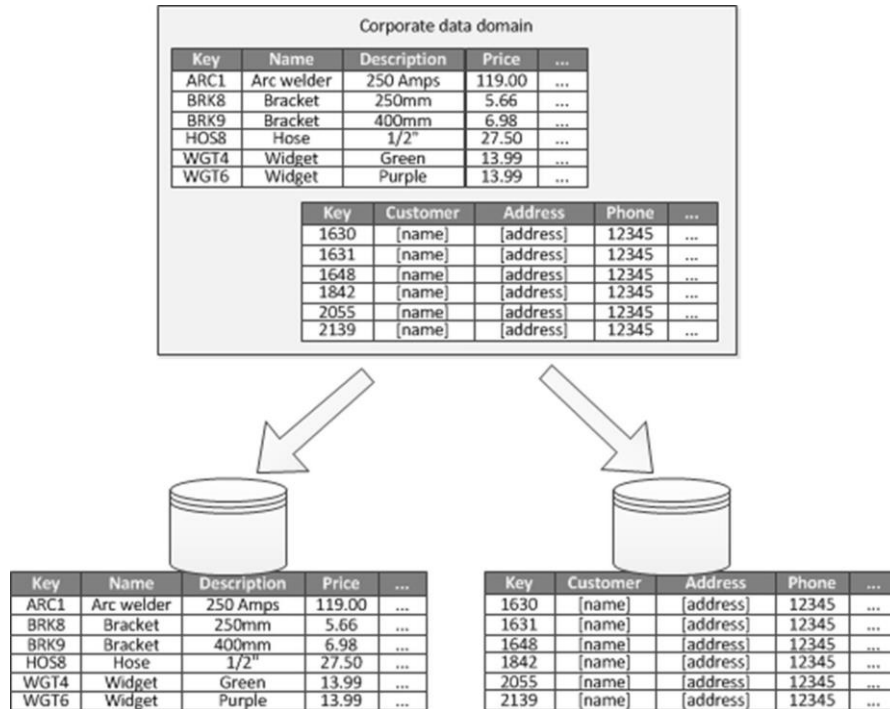
Key	Name	Description	Stock	Price	LastOrdered
HOS8	Hose	1/2"	27	27.50	18-Aug-2013
WGT4	Widget	Green	16	13.99	3-Feb-2013
WGT6	Widget	Purple	76	13.99	31-Mar-2013

# Vertical Partitioning

Key	Name	Description	Stock	Price	LastOrdered
ARC1	Arc welder	250 Amps	8	119.00	25-Nov-2013
BRK8	Bracket	250mm	46	5.66	18-Nov-2013
BRK9	Bracket	400mm	82	6.98	1-Jul-2013
HOS8	Hose	1/2"	27	27.50	18-Aug-2013
WGT4	Widget	Green	16	13.99	3-Feb-2013
WGT6	Widget	Purple	76	13.99	31-Mar-2013



# Functional Partitioning



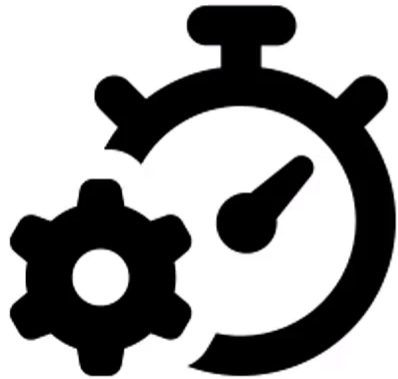


# Partitioning

## Design Considerations

- Balance data distribution
- Balance workload distribution
- Minimize cross-partition data access or joins operations.
- Replicate static reference data
- Prefer eventual consistency
- Replicate partitions

# Data Lake Optimization Techniques



- **Data Ingestion considerations**
  - Storage hardware
  - High speed internal network
  - Fast network connection b/w on-premises and cloud
- **Parallel read/write**
  - e.g. Data Factory parallel copies settings
- **Structure your data set**
  - File size vs number of files
    - File size b/w 256 MB to 100 GB
  - Folder and file structure
    - e.g. Dataset\YYYY\MM\DD\datafile\_YYYY\_MM\_DD\_HH\_MM.tsv
- **Same region**
- **Batch Data**

## Stream Analytics – Optimization



# Stream Analytics – Optimization



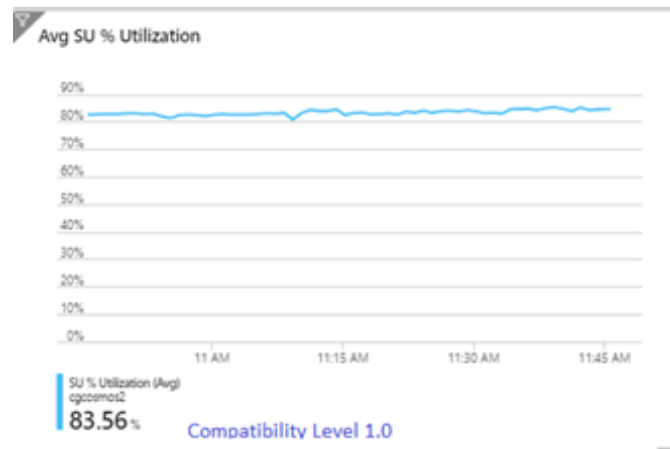
## Three main component

- Input
- output
- Data processing Query

# Stream Analytics – Optimization

## Streaming Units (SUs)

- Processing power (CPU and Memory) allocated to your stream analytics job.
- Azure Stream Analytics jobs perform all processing in memory
- If SU% utilization is low and input events get backlogged
- Microsoft recommends setting an alert on 80% SU Utilization metric to prevent resource exhaustion
- The best practice is to start with 6 SUs for queries that don't use PARTITION BY
- Complex query logic could have high SU% utilization even when it is not continuously receiving input events.



# Stream Analytics – Optimization

## Parallelization

- Partitioning helps to divide data in subsets.
- This would be based on partition key.
- If the data in the Event Hub has a partition key defined, then it is highly recommended to define the partition key in the input of Stream Analytics Job.
- Input are already partitioned, output needs to be partitioned
- Embarrassingly parallel jobs
  - An embarrassingly parallel job is the most scalable scenario in Azure Stream Analytics.
  - It connects one partition of the input to one instance of the query to one partition of the output.
  - The number of input partitions must equal the number of output partitions.

SQL

```
SELECT *  
INTO output  
FROM input  
PARTITION BY DeviceID  
INTO 10
```

# Stream Analytics – Optimization

## Steps in Query

- You can have multiple step in a query.
- You can start with 6 SUs for queries that don't use PARTITION BY
- You can also add 6 streaming units for each partition in a partitioned step.
- Example:
  - Let's say your input stream is partitioned by value of 10, and you only have one step in query

SQL

```
SELECT *  
INTO output  
FROM input  
PARTITION BY DeviceID  
INTO 10
```



## Synapse Analytics Optimization

- **Maintain Statistics**
  - Automatically detect and create statistics on columns
  - `AUTO_CREATE_STATISTICS`
  - Update statistics of more relevant columns like date (or columns used in joins, where and group by clause)
- **PolyBase**
  - ADF or BCP can be used for small load
  - PolyBase is best choice for large volume of data
  - MPP architecture
  - CTAS or INSERT INTO
- **Hash distribution large tables**
  - Default is Round Robin distribution
  - Small tables joins – Round Robin is fine
  - Big tables joins – use Hash Distribution



## Synapse Analytics Optimization

- **Do not over partition**
  - Too many partition can slow down query
  - Partition should have more than 1 million rows
  - 60 partition by default
  - So if you manually create 100 partition, behind the scene it is  $100 \times 60 = 6000$  partitions.
- **Use the smallest possible column size**
  - Important for char or varchar type columns
  - Use varchar instead nvarchar
- **Scaling**
  - Before you perform a heavy data loading or transformation operation
  - During peak business hours
- **Pausing and resuming compute**
  - Storage and compute are separate
  - Transaction cancel