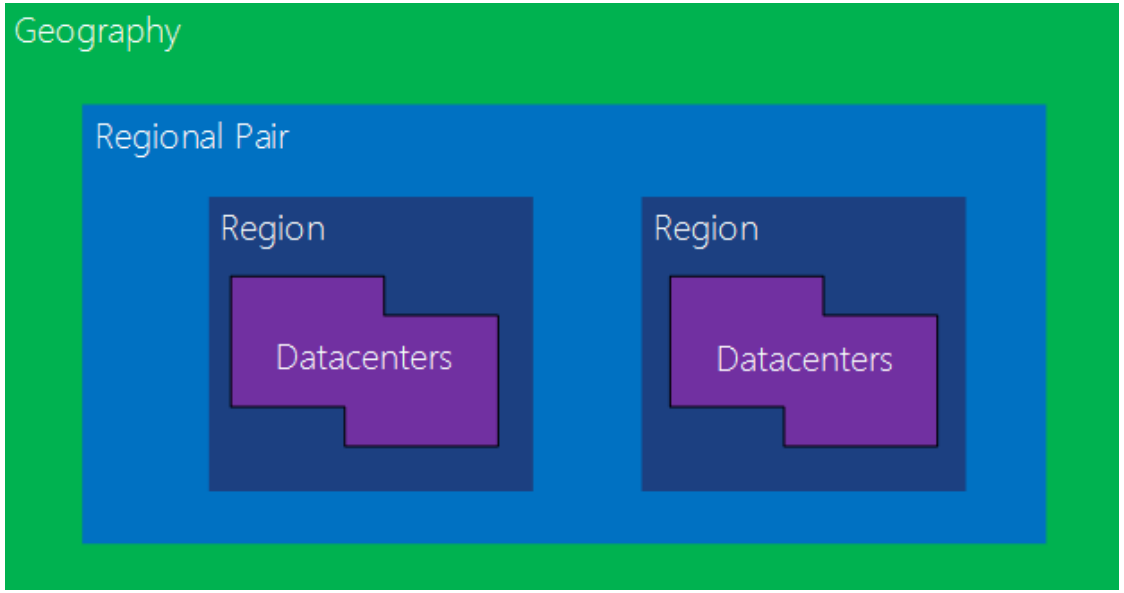
Azure Data Warehouse- Operations and Monitoring

Topics

- Operational
- Monitoring
- Alerts
- Auditing
- ADW Diagnostic Logs + Azure Monitor Log
- Specific Use Case Examples

Backup

- **Automatic Restore Points**
  - On by default
  - Snapshots taken every few hours
  - 8 Hour RPO
  - 7 day Retention (14 day on-demand)
- **User-defined Restore Points**
  - Triggered Manually
  - 7 day retention or 42 restore points (whichever comes first)
  - Good for snapshots before/after large data modifications
- **Geo-backups**
  - Once per day (24 hour RPO)
  - Taken to a paired data center



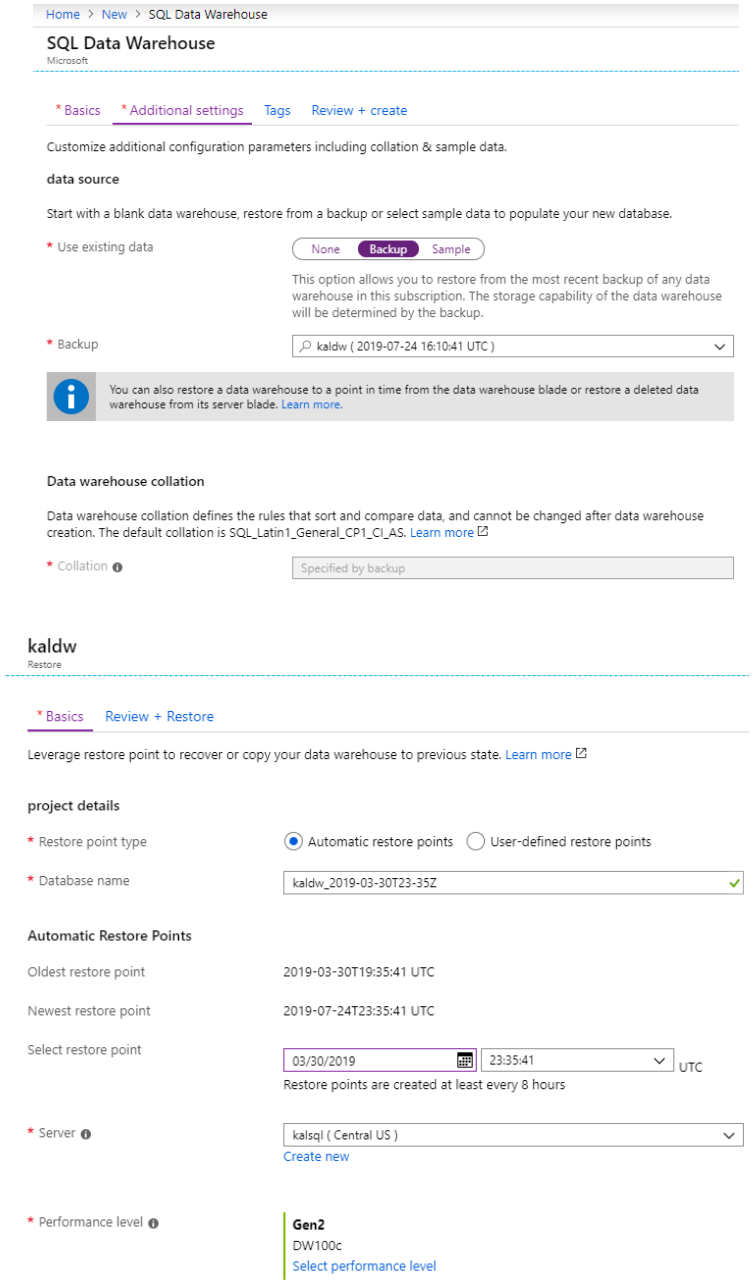
Restore

- Can be invoked from Portal, PowerShell, REST API
- Regional restores under 20 minutes (irrespective of the data size)
- Can be used to quickly create dev/test/reporting copies
- Allows cross-region, cross-RG restores
- Cross-sub restores require ‘move logical server’ operation
- Cannot overwrite an existing DW
- Can be used to recover a deleted DW

```
# Get the deleted database to restore
$DeletedDatabase = Get-AzureRmSqlDeletedDatabaseBackup -ResourceGroupName $ResourceGroupName -ServerName $ServerName

# Restore deleted database
$RestoredDatabase = Restore-AzureRmSqlDatabase –FromDeletedDatabaseBackup –DeletionDate $DeletedDatabase.DeletionDate

# Verify the status of restored database
$RestoredDatabase.status
```

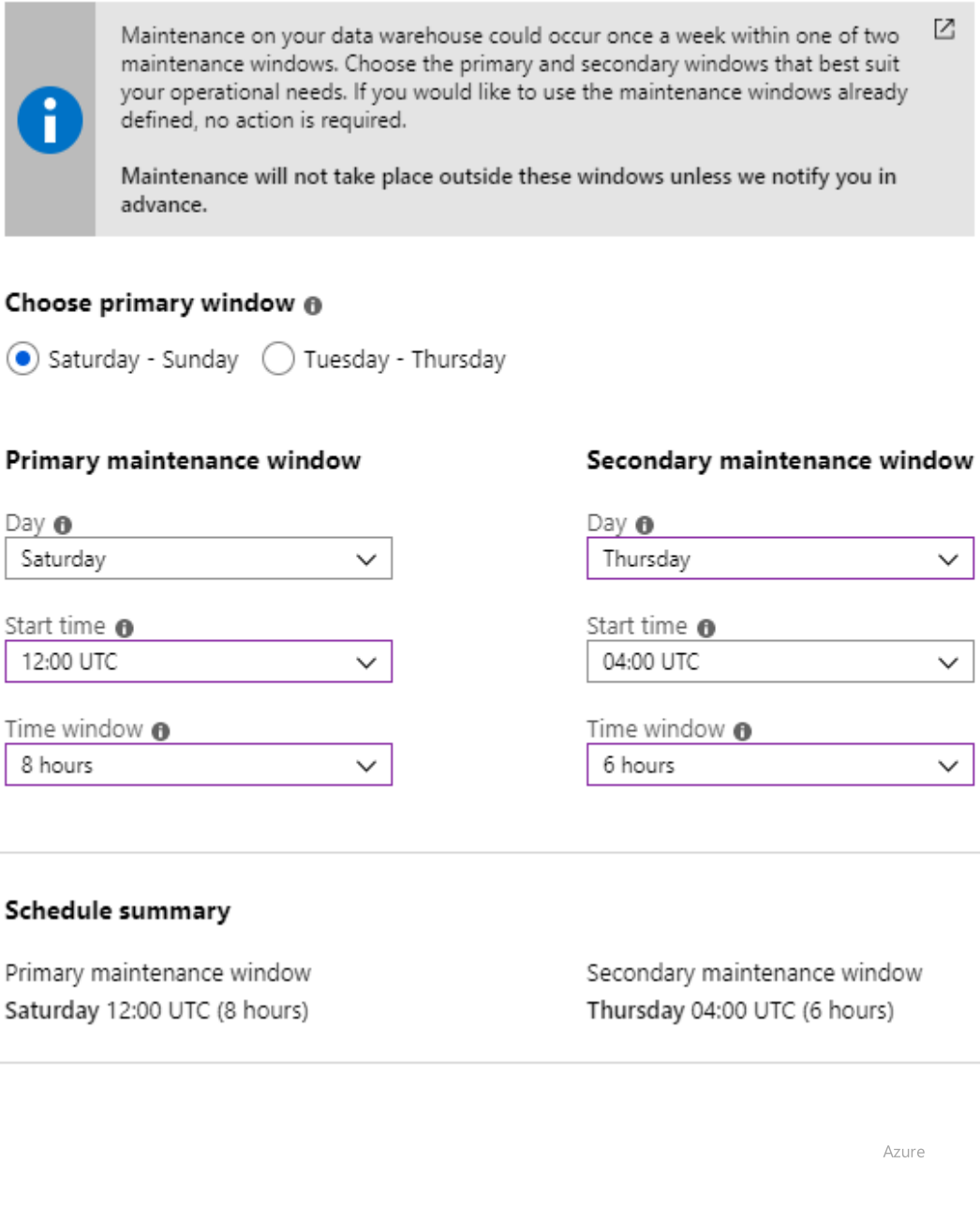


How to increase the retention period (>14 days)

- **Configure the 14-day retention**
- **Identify the oldest restore point**
  - Select top 1 * from sys.pdw_loader_backup_runs ORDER BY run_id
- **Restore the oldest restore point to a new DW and PAUSE**
- **Drop the oldest DW that is crossing the desired retention point**
- **Repeat the above steps daily**
- **Data for a paused DW is retained indefinitely**
- **No additional restore points will be created on a paused DW**

Maintenance Windows

- 3 to 8 Hour window to apply patches/upgrades/new features
- Primary and Secondary windows within 7-day period
- Applies to 500 DWUc or higher
- All active sessions will be cancelled/rolled back
- Notifications
  - 24-hour in advance
  - Cancellation (if applicable)
  - After completion
- Benefits
  - Ensure upgrades happen on your schedule.
  - Predictable planning for long-running jobs.
  - Stay informed of start and end of maintenance.



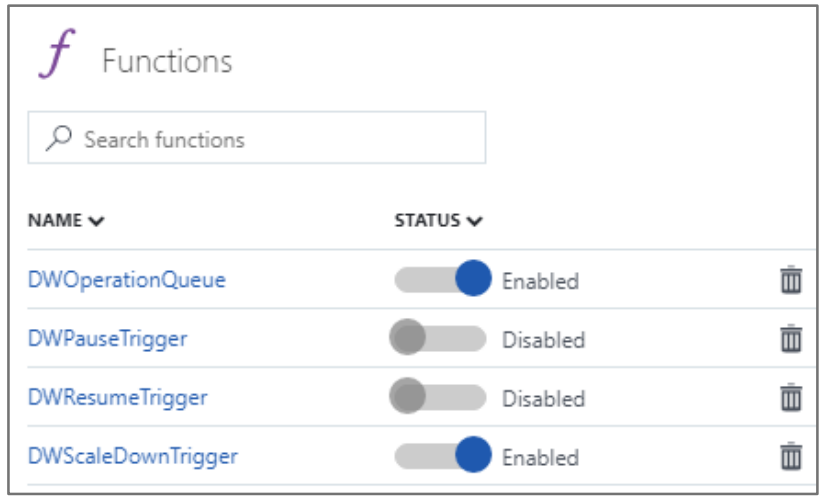
Automating compute management

Enable 'Auto-Scale' with Azure Functions/Logic Apps

Use Azure Functions to schedule SQL Data Warehouse triggers.

Scale-up, scale-down, pause, and resume automatically.

Combine with PowerShell, T-SQL, or REST API to prevent active query cancellation.

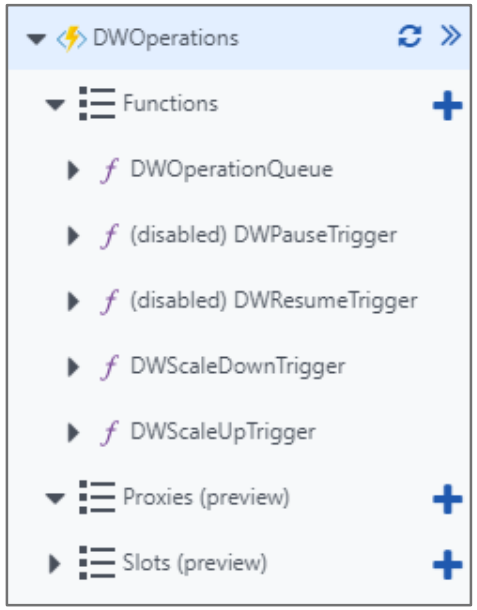


Benefits

Scale data warehouse to meet higher compute needs.

Minimize costs by scaling down or pausing compute.

Scale operations completed in minutes.

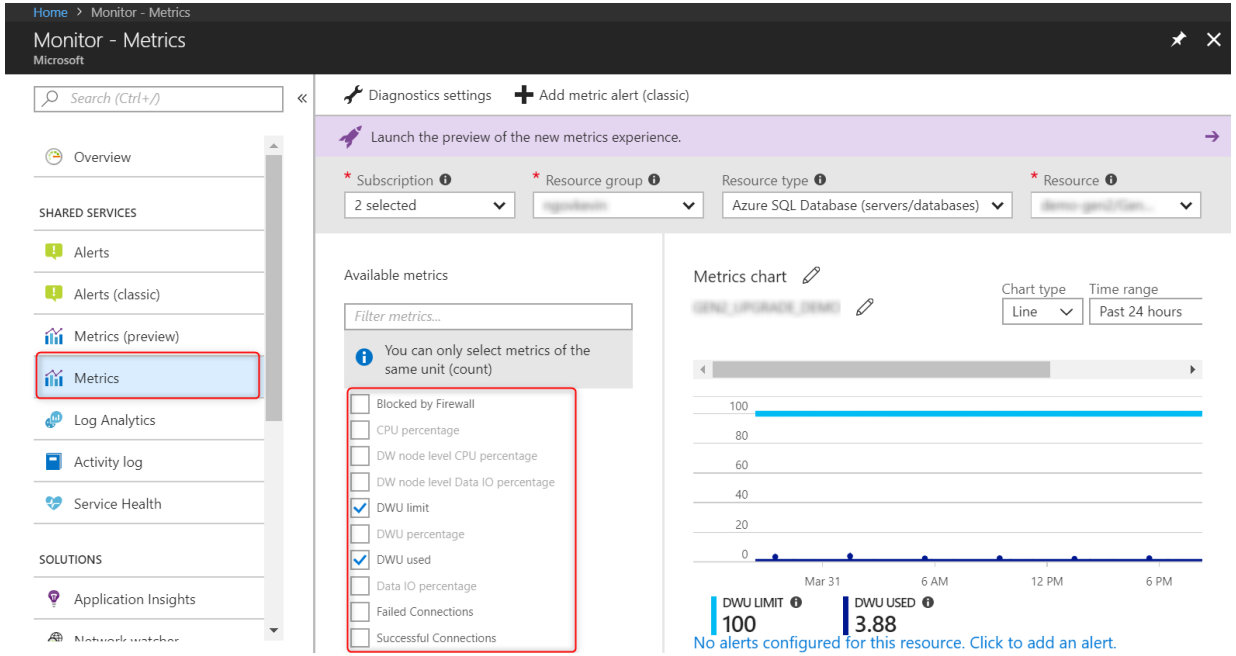


Azure Data Warehouse Monitoring Tools

- **Azure Portal**
  - Azure Monitor
  - Alerts
  - Log Analytics
  - Azure Advisors
- **SQL Server Management Studio**
  - DMVs
- **Visual Studio Data Tools**
- **Azure Data Studio**
  - Reports

Monitoring using the Portal

- **Overview**
  - Metrics surfaced through Azure Monitor
  - Tracks metrics related to utilization and security
  - Export to Log Analytics for further analysis
  - No cost for first 31 days of each GB of data
- **Benefits**
  - Holistic monitoring across Azure analytics platform
  - Customize charts and create enriched dashboards
  - Identify under or over-utilization
  - Make informed scaling decisions



Metrics

CPU Percentage

Data IO Percentage

Successful Connections

Failed Connections

Blocked Connections

DWU (Compute) Limit

DWU Percentage (Used)

Cache Hit Percentage

Cache Used Percentage

Local tempdb Percentage

Azure Monitor - Alerts

Overview

Set alerts on metrics or activity logs.

Define metric thresholds.

Track individual or count of log events.

Send emails or call webhooks on triggers.

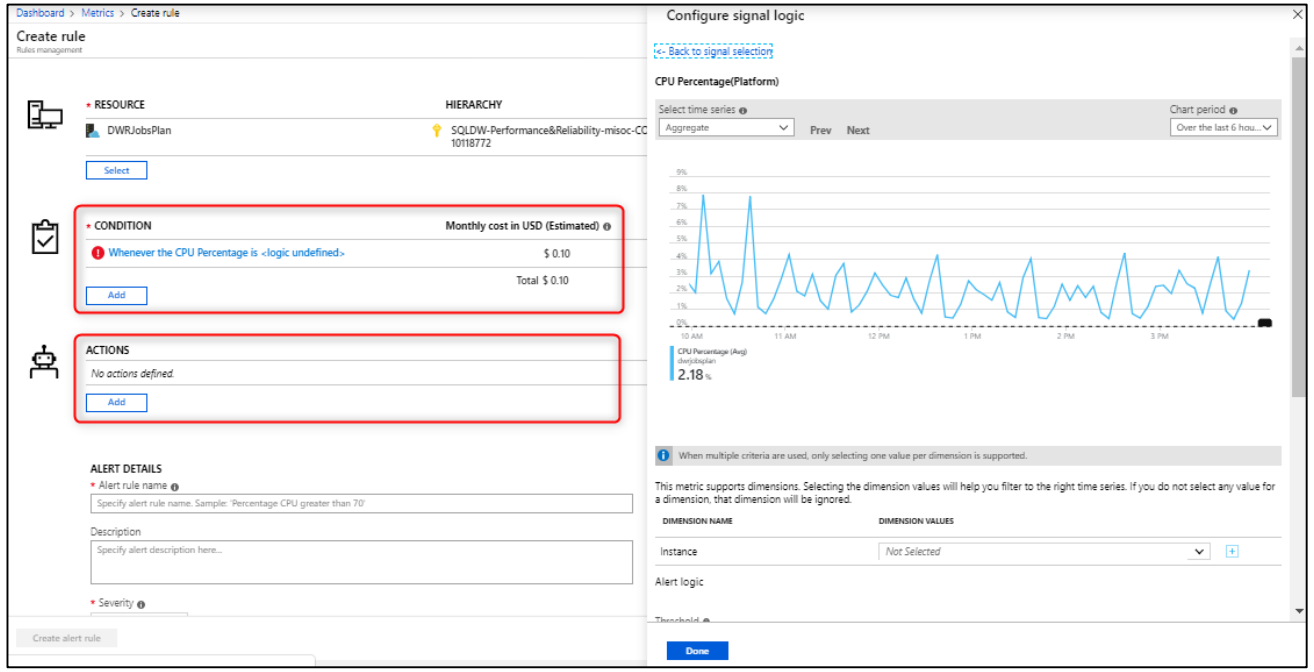
Manage with Azure portal, PowerShell, CLI, and REST APIs.

Benefits

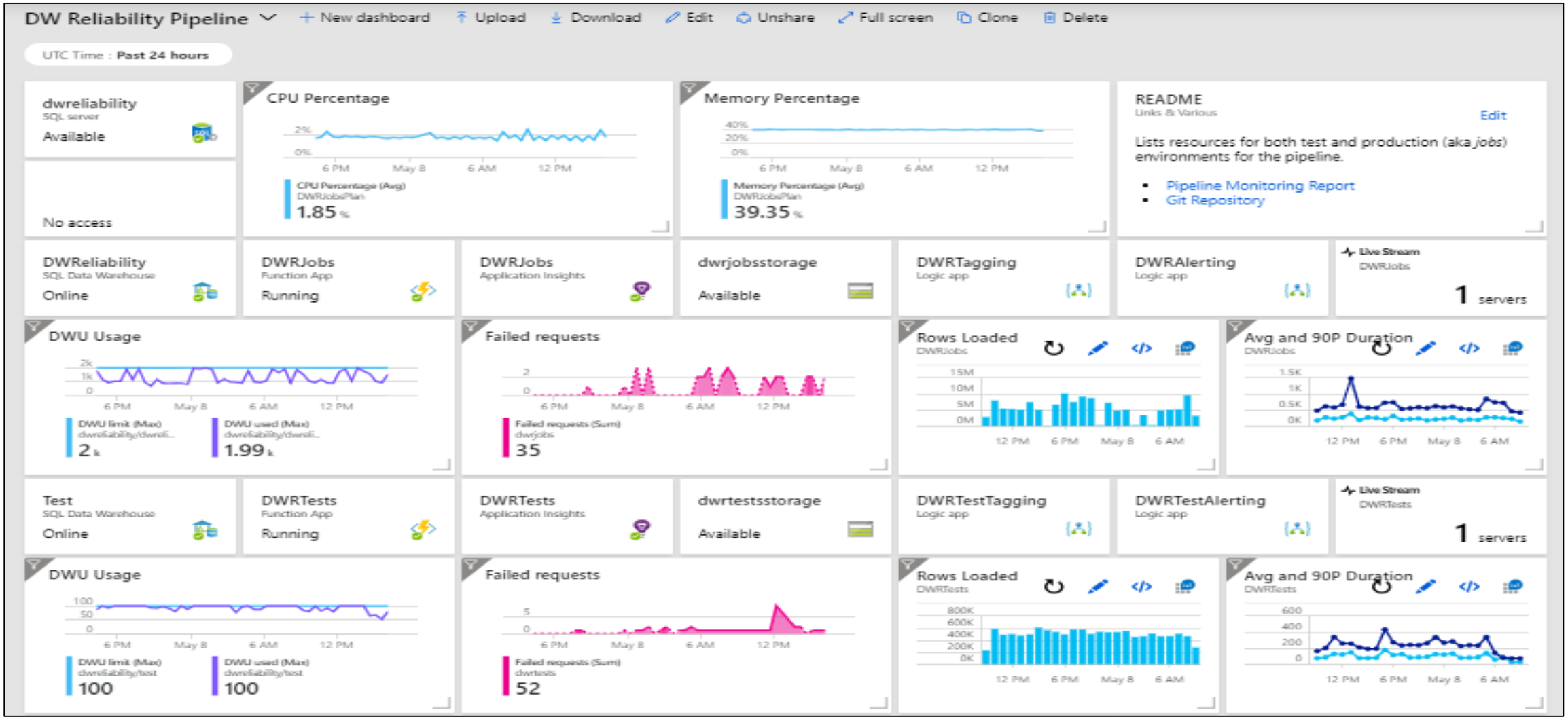
Automatically track data warehouse state.

Customizable triggers for immediate action.

Integration with SQL Data Warehouse monitoring.



Azure Dashboards



Azure Monitor Logs

Overview

Analyze SQL Data Warehouse logs and metrics.

Stream directly to Event Hubs.

Customizable retention with Azure Storage.

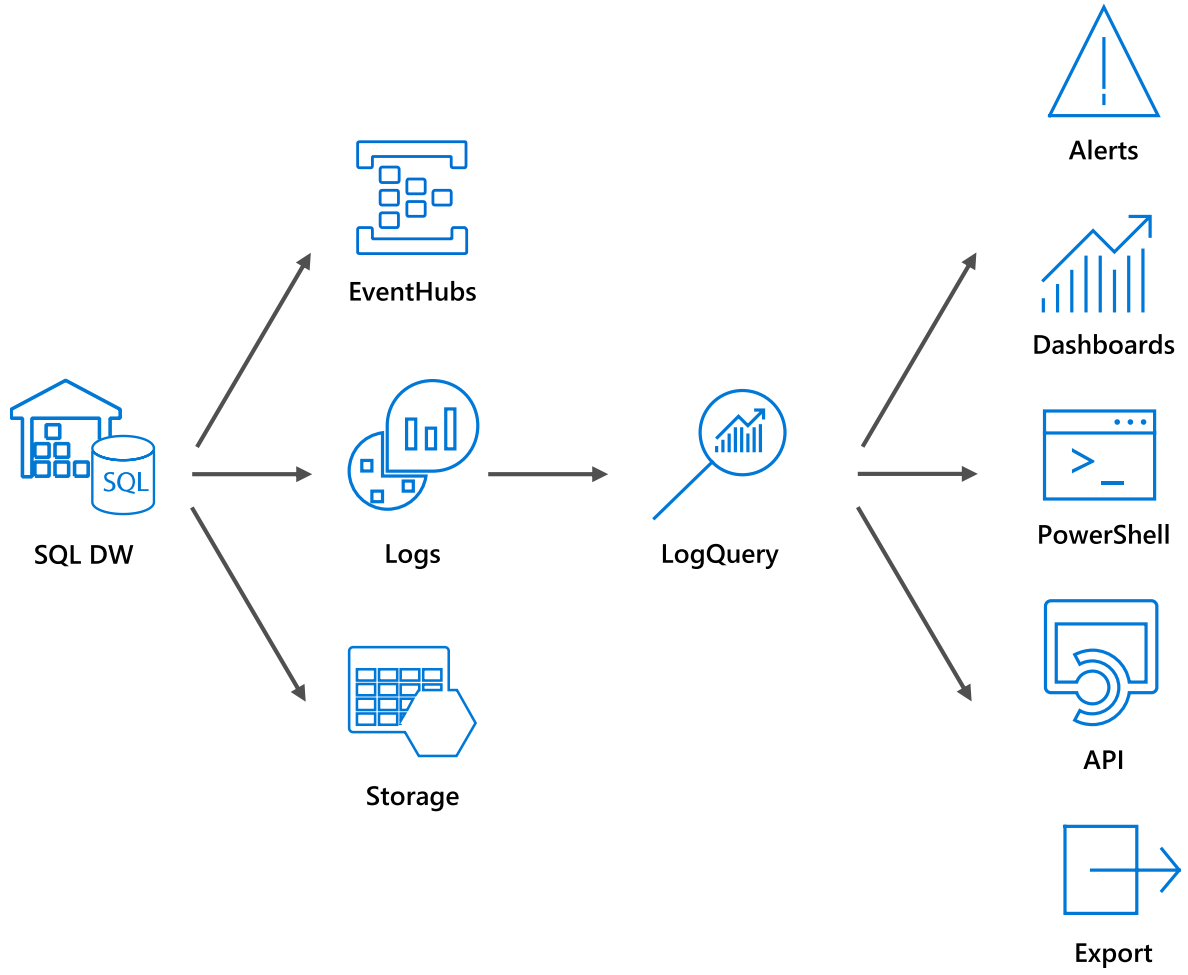
Operationalize logs and metrics for insights.

Benefits

Historical query troubleshooting.

Real-time monitoring and long-term.

Rich and customizable visualization options.



Azure Monitor Logs

Configure ADW for Azure Monitor Logging

- Portal pane of your Data Warehouse - Monitoring
  - Diagnostic Settings → Configure Azure Monitor Logs

DMVs currently supported:

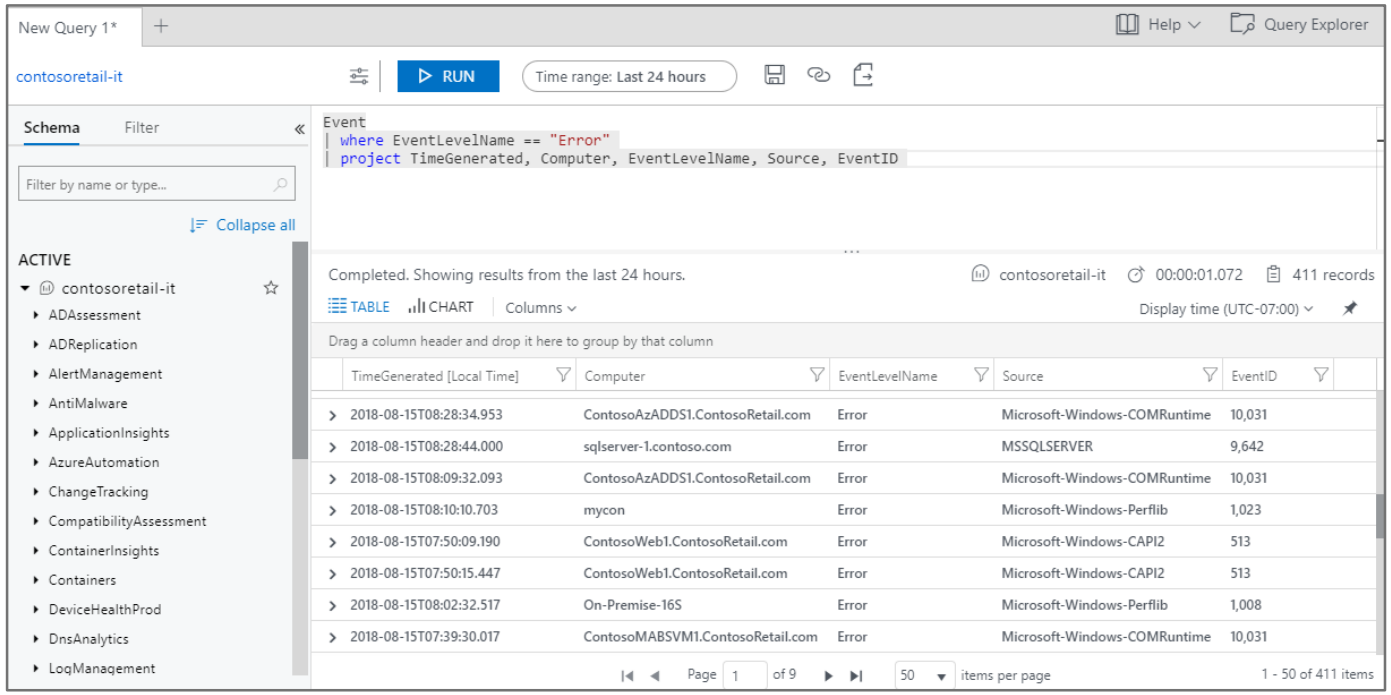
[sys.dm_pdw_exec_requests]

[sys.dm_pdw_request_steps]

[sys.dm_pdw_dms_workers]

[sys.dm_pdw_waits]

[sys.dm_pdw_sql_requests]



Azure Data Warehouse – Azure Monitor Logs

Review logs

- Log Analytics → Portal – Monitor – Logs

  Analyze logs using log queries

  Save queries for reuse

  Create log alerts

  Pin query results to a dashboard

Log analytics – example

Log Analytics is based on Azure Data Explorer - uses Kusto Query Language (KQL)

Sample query to determine the most active resource classes

```
AzureDiagnostics
| where Category contains "ExecRequests"
| where Status_s == "Completed"
| summarize totalQueries = dcount(RequestId_s) by ResourceClass_s
| render barchart

//Count of all queued queries
AzureDiagnostics
| where Category contains "waits"
| where Type_s == "UserConcurrencyResourceType"
| summarize totalQueuedQueries = dcount(RequestId_s)

//Chart for top requests most impacted by data movement operations
AzureDiagnostics
| where Category == "RequestSteps"
| where OperationType_s in ("ShuffleMoveOperation", "BroadcastMoveOperation",
"PartitionMoveOperation", "RoundRobinMoveOperation", "SingleSourceRoundRobinMoveOperation",
"MoveOperation", "TrimMoveOperation")
| where Status_s == "Complete"
| project RequestId_s, duration=datetime_diff('millisecond',EndTime_t, StartTime_t)
| order by duration desc
| take 10
| render barchart
```

Demo – Portal Monitor Features

Azure Advisor recommendations

Overview

Azure Advisor provides automatic SQL DW recommendations.

Detect and suggest common performance improvements.

Generated every 24 hours.

Integrates with all Azure Advisor recommendations.

Benefits

Improve SQL Data Warehouse performance.

Make informed scaling decisions.

Maximize cost-efficiency of current scale.

Recommendation Data Sources

Data distribution and replicated tables

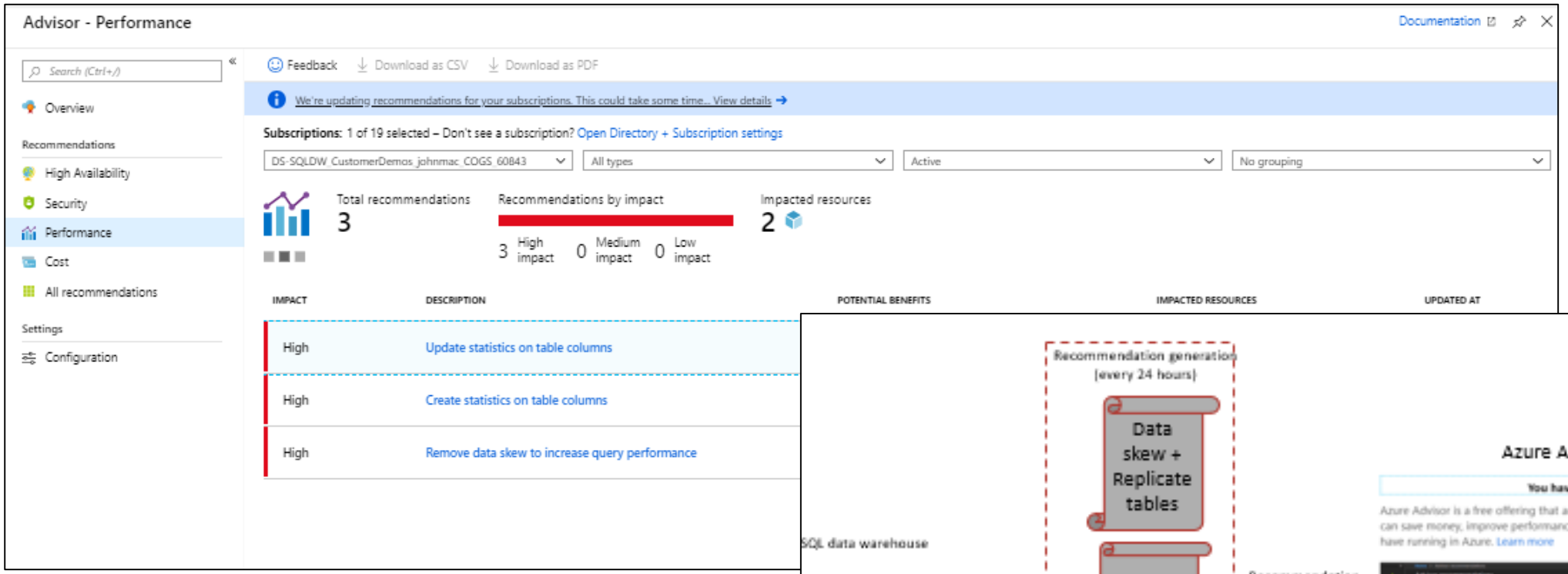
Column statistics

Tempdb usage

Adaptive cache hits

Create or update table statistics

Azure Data Warehouse – Azure Advisors



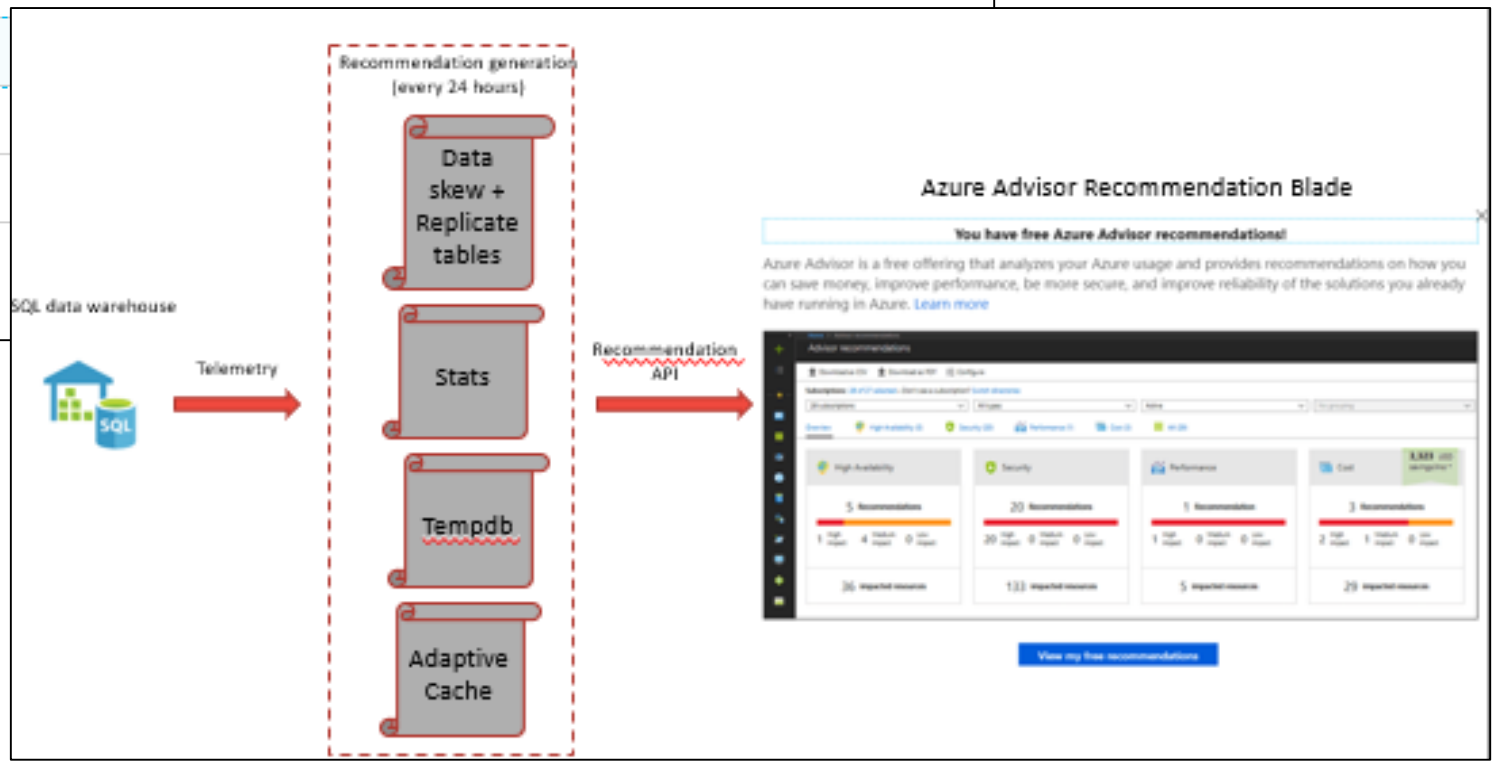
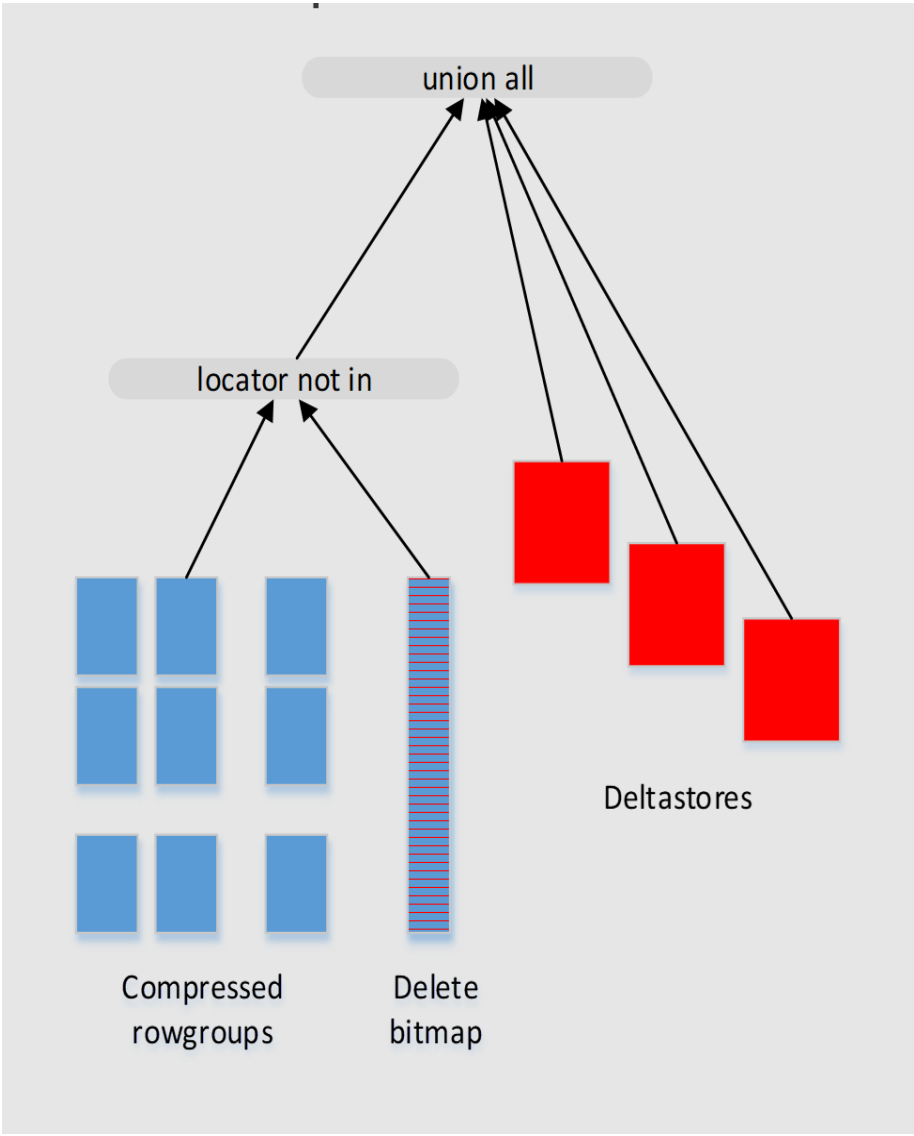


Table Skew

- **Uneven distribution of rows across 60 distributions**
- **Can be due to poor distribution column choice**
- **Also can be due to change in data patterns**
- **Up to 10% variation is OK**
- **Can cause severe performance impact due to**
  - Data Movement
  - Resource Contention
- **Can be detected using**
  - DBCC PDW_SHOWSPACEUSED
  - sys.dm_pdw_nodes_db_partition_stats
  - dbo.vTableSizes

CCI Health Check

- **Check for**
  - Too many open rowgroups
  - <100K rows/Rowgroup (~1million rows is ideal)
  - Trim Reason
- **How to avoid/fix:**
  - Avoid frequent deletes/updates
  - Avoid singleton inserts
  - Avoid too many partitions
  - Reconsider on tables < 100 million rows
  - Use highest possible RC for loads
  - Rebuild/Reorganize periodically



Indexes/Statistics

- **Statistics**
  - Useful for optimized query plans
  - Enable automatic creation (SET AUTO_CREATE_STATISTICS ON)
  - Create multi-column stats if needed
  - Update periodically (UPDATE STATISTICS)
  - Monitor using
    - STATS_DATE
    - DBCC SHOW_STATISTICS
- **Clustered/Non-Clustered INDEXES**
  - Monitor Fragmentation (sys.dm_db_index_physical_stats)
  - Rebuild if >30% Fragmentation
  - Reorganize if >5% and <=30%

Automatic statistics management

Overview

Statistics are automatically created and maintained in SQL DW. Incoming queries are analyzed and individual column statistics are generated on the columns that improve cardinality estimates to enhance query performance.

Statistics are automatically updated as data modifications occur in underlying tables. By default, these updates are synchronous but can be configured to be asynchronous.

Statistics are considered out of date when:

- There was a data change on an empty table
- The number of rows in the table at time of statistics creation was 500 or less, and more than 500 rows have been updated
- The number of rows in the table at time of statistics creation was more than 500, and more than 500 + 20% of rows have been updated

```
-- Turn on/off auto-create statistics settings
ALTER DATABASE {database_name}
SET AUTO_CREATE_STATISTICS { ON | OFF }

-- Turn on/off auto-update statistics settings
ALTER DATABASE {database_name}
SET AUTO_UPDATE_STATISTICS { ON | OFF }

-- Configure synchronous/asynchronous update
ALTER DATABASE {database_name}
SET AUTO_UPDATE_STATISTICS_ASYNC { ON | OFF }

-- Check statistics settings for a database
SELECT   is_auto_create_stats_on,
         is_auto_update_stats_on,
         is_auto_update_stats_async_on
FROM     sys.databases
```