Coffing Data Warehousing®

Azure Architecture Class Outline

Introduction to the Azure SQL Data Warehouse

Introduction to the Family of SQL Server Products

Introduction to the Family Continued

Microsoft Azure SQL Data Warehouse

Symmetric Multi-Processing (SMP)

What is Parallel Processing?

The Basics of a Single Computer

Data in Memory is fast as Lightning

Parallel Processing of Data

A Table has Columns and Rows

The Azure SQL Data Warehouse has Linear Scalability

The Architecture of the Azure SQL Data Warehouse

Nexus is now available on the Microsoft Azure Cloud

The MPP Engine is the Optimizer

The Azure SQL Data Warehouse System

The Azure SQL Data Warehouse System is Scalable

The Control Node

The Data Rack

The Landing Zone

The Backup Node

Software as a Service (SaaS) and the Elastic Database

Azure Data Lake

Azure Disaster Recovery

Security and Compliance

How to Get an EXPLAIN Plan

The Azure SQL Data Warehouse Table Structures

The 5 Concepts of Azure SQL Data Warehouse Tables

Tables are Either Distributed by Hash or Replicated (1 of 5)

Table Rows are Either Sorted or Unsorted (2 of 5)

Tables are Stored in Either Row or Columnar Format (3 of 5)

Tables can be Partitioned (4 of 5)

There are Permanent, Temporary and External Tables (5 of 5)

Creating a Table with a Distribution Key

Creating a Table that is replicated

Distributed by Hash vs. Replication

The Concept is all about the Joins

Creation of a Hash Distributed Table with a Clustered Index

A Clustered Index Sorts the Data Stored on Disk

Each Node Has 8 Distributions

How Hashed Tables are Stored among a Single Node

Hashed Tables Will Be Distributed Among All Distributions

Creation of a Replicated Table

How Replicated Tables are Stored among a Single Node

Replicated Table will be duplicated among Each Node

Distributed by Replication

How Hashed and Replicated Tables Work Together

Tables are stored as Row-based or Column-based

Creation of a Columnar Table that is hashed

How Hashed Columnar Tables are Stored on a Single Node

How Hashed Columnar Tables are Stored on All Distributions

Comparing Normal Table vs. Columnar Tables

Columnar can move just One Segment to Memory

Segments on Distributions are aligned to rebuild a Row

Why Columnar?

Columnar Tables Store Each Column in Separate Pages

Visualize the Data – Rows vs. Columns

Creation of a Columnar Table that is replicated

Creating a Partitioned Table per Month

A Visual of One Year of Data with Range per Month

Another Create Example of a Partitioned Table

Creating a Partitioned Table per Month That is a Columnstore

Visual of Row Partitioning and Columnar Storage

CREATE TABLE AS (CTAS) Example

Creating a Temporary Table

Facts about Tables

Hashing and Data Distribution

Distribution Keys Hashed on Unique Values Spread Evenly

Distribution Keys with Non-Unique Values Spread Unevenly

Best Practices for Choosing a Distribution Key

The Hash Map determines which Distribution owns the Row

The Hash Map determines which Node will own the Row

A Review of the Hashing Process

Non-Unique Distribution Keys have Skewed Data

The Technical Details

Every Node has the Exact Same Tables

Hashed Tables are spread across All Distributions

The Table Header and the Data Rows are Stored Separately

A Distribution Stores the Rows of a Table inside a Data Block

To Read a Data Block a Node Moves the Block into Memory

A Full Table Scan Means All Nodes Must Read All Rows

Rows are organized inside a Page

Moving Data Blocks is Like Checking in Luggage

As Row-Based Tables Get Bigger, the Page Splits

Data Pages are Processed One at a Time per Unit

Creating a Table that is a Heap

Heap Page

Extents

Creating a Table that has a Clustered Index

Clustered Index Page

The Row Offset Array is the Guidance System for Every Row

The Row Offset Array Provides Two Search Options (1 of 2)

The Row Offset Array Provides Two Search Options (2 of 2)

The Row Offset Array Helps with Inserts

B-Trees

The Building of a B-Tree for a Clustered Index (1 of 3)

The Building of a B-Tree for a Clustered Index (2 of 3)

The Building of a B-Tree for a Clustered Index (3 of 3)

- When Do I Create a Clustered Index?
- When Do I Create a Non Clustered Index?
- B-Tree for Non Clustered Index on a Clustered Table (1 of 2)
- B-Tree for Non Clustered Index on a Clustered Table (2 of 2)
- Adding a Non Clustered Index to A Heap
- B-Tree for Non Clustered Index on a Heap Table (1 of 2)
- B-Tree for Non Clustered Index on a Heap Table (2 of 2)
- Max Levels on the Azure SQL Data Warehouse
- Azure SQL Data Warehouse Data Types
- Character Data Types for SQL Server
- Numeric Data Types for SQL Server
- Date and Time Data Types for SQL Server
- Additional Data Types for SQL Server

CREATE Statistics

- CREATE Statistics Syntax
- CREATE Statistics on a Percentage of a Table
- CREATE Statistics on a Sample by Using the System Default
- CREATE Statistics on a Multi-Column Join Key
- What to Column(s) to CREATE Statistics On
- CREATE Statistics Using a WHERE Clause
- Updating All Statistics on a Table

Updating Only Certain Statistics on a Table

Dropping Statistics on Certain Statistics on a Table

Showing the Statistics

DBCC SHOW_STATISTICS

DBCC SHOW_STATISTICS WITH HISTOGRAM