Building a Data Warehouse Dimensional Model

using

Azure Synapse Analytics Serverless SQL Pool

Azure Synapse Analytics
Data Toboggan – Saturday 30th January 2021

Andy Cutler
Lightning Session

Andy Cutler

Independent Consultant & Contractor

Azure Data Platform & Power BI

www.datahai.co.uk

https://twitter.com/MrAndyCutler

https://www.linkedin.com/in/andycutler/

SCAN ME

Session Overview

- Question…
- What is the Synapse Serverless SQL Pool?
- What is the Dimensional Model?
- Initial Load Dimensions
- Initial Load Facts
- Incremental Loading
- Reading the Dimensional Data
- Considerations

DATA TOBOGGAN

Question...

Can I..?

Can I build a Data Warehouse using the Dimensional Modelling technique and use Azure Synapse Serverless SQL Pool as the processing engine?

Why do I want to do this?

- I would like to leverage my existing SQL skills
- I'm comfortable with Dimensional modelling theory
- I'd like to leverage the flexibility of the Data Lake
- I'm interested in exploring the Data Lakehouse concept

What would I usually do?

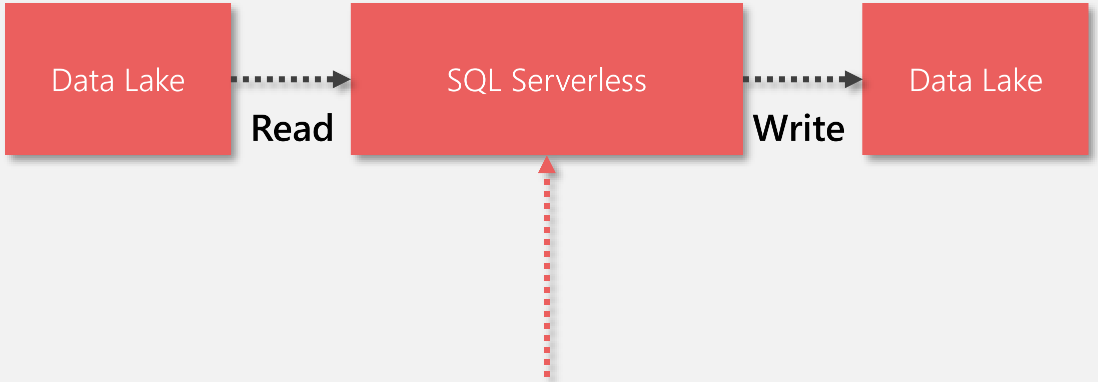
- SQL Server 20XX: On-Premises or Azure Virtual Machine (VM)
- Azure SQL Database: Feature-rich relational database service
- Synapse Analytics Dedicated SQL Pools (AKA SQL Data Warehouse)

What is Synapse Serverless SQL Pool?

Serverless SQL Pools is a SQL-based query service built into Azure Synapse Analytics that allows reading and writing CSV, Parquet and JSON data stored within Azure Data Lake Gen2.

Cost model is based on amount of data processed…..currently £3.727 per 1TB

Part of a suite of services within the overall Azure Synapse Service which also includes Dedicated SQL Pools, Pipelines (Data Factory) & Power BI.



No data is stored within the Serverless SQL Pool

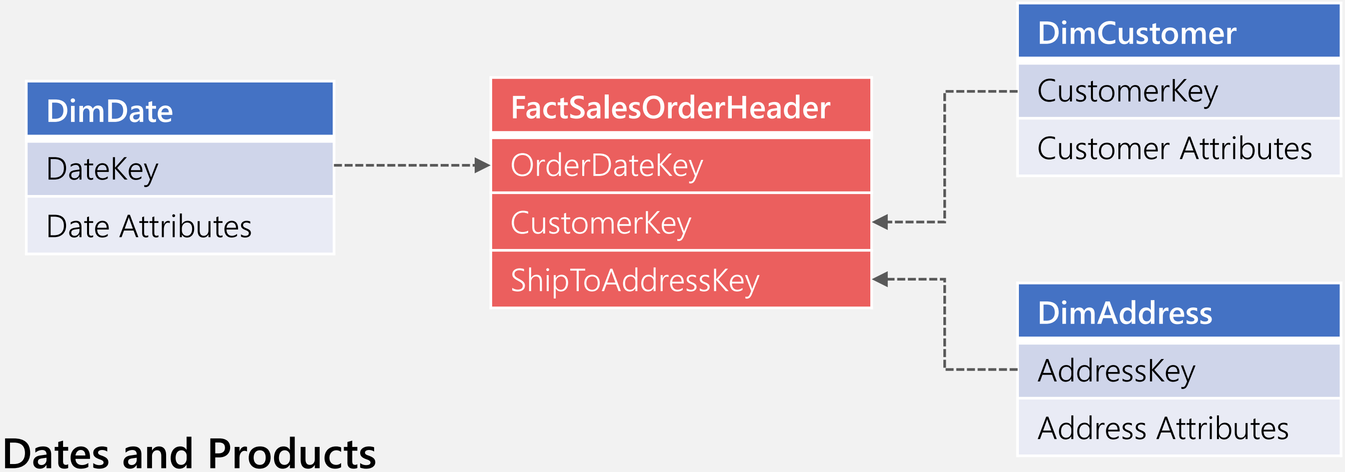
Serverless SQL Pools support familiar SQL objects:

- Create Databases to store objects
- SQL syntax to write data transformations
- Stored Procedures to encapsulate logic

What is the Dimensional Model?

The Dimensional model is a Data Warehouse modelling process that has existed for many years (30+?!!)

It allows the modelling of data as either a “measurement” or a “label” of a business process



DimDate
DateKey
Date Attributes

FactSalesOrderHeader
OrderDateKey
CustomerKey
ShipToAddressKey

DimCustomer
CustomerKey
Customer Attributes

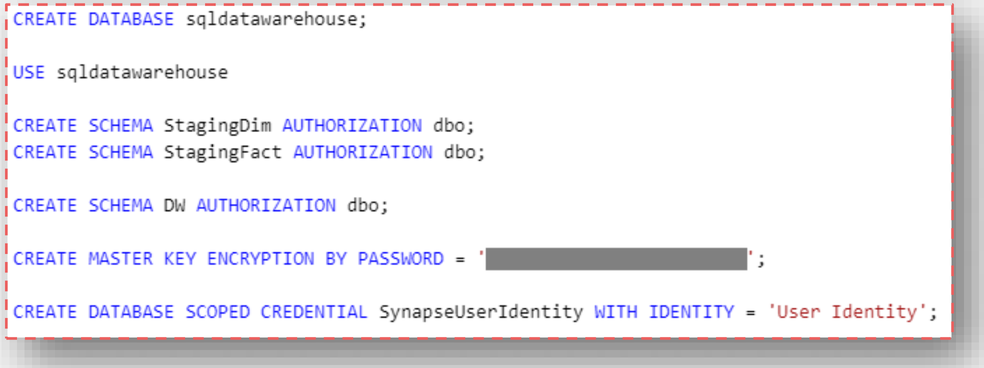
DimAddress
AddressKey
Address Attributes

- There are 2 basic types of data:
    - Dimensions: The business reference data. E.G. Dates and Products
    - Facts: Measurements of a business process E.G. Sales
- Also known as a Star Schema
- Well-known and popular Data Warehouse methodology

Setup the Azure Synapse Environment

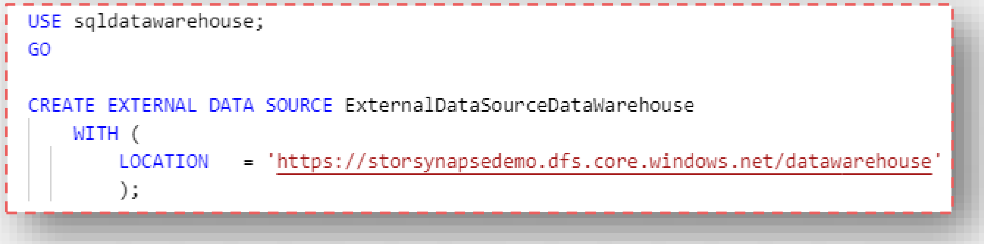
We'll setup the environment by creating a SQL Serverless database, schema, security, external data sources and file formats

1



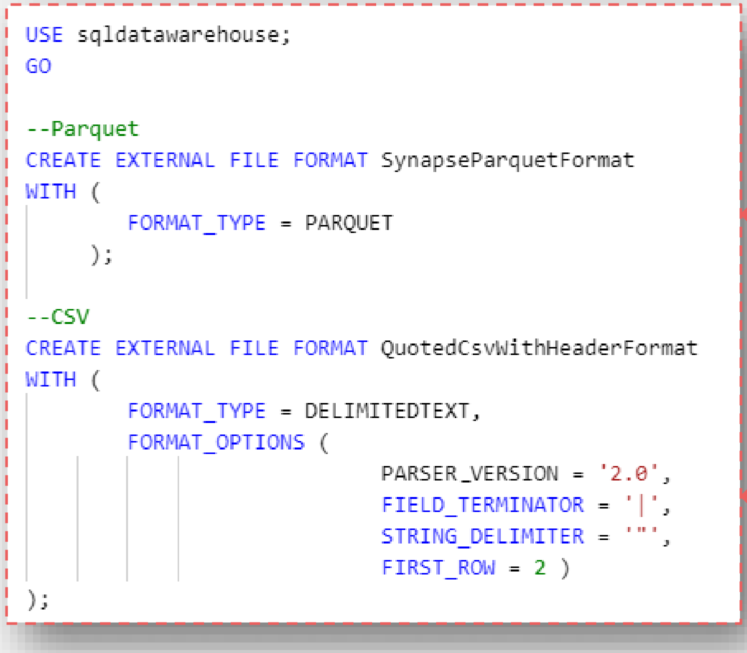
```sql
CREATE DATABASE sqldatawarehouse;

USE sqldatawarehouse

CREATE SCHEMA StagingDim AUTHORIZATION dbo;
CREATE SCHEMA StagingFact AUTHORIZATION dbo;

CREATE SCHEMA DW AUTHORIZATION dbo;

CREATE MASTER KEY ENCRYPTION BY PASSWORD = '[redacted]';

CREATE DATABASE SCOPED CREDENTIAL SynapseUserIdentity WITH IDENTITY = 'User Identity';
```

2



```sql
USE sqldatawarehouse;
GO

CREATE EXTERNAL DATA SOURCE ExternalDataSourceDataWarehouse
    WITH (
        LOCATION   = 'https://storsynapsedemo.dfs.core.windows.net/datawarehouse'
        );
```

3



```sql
USE sqldatawarehouse;
GO

--Parquet
CREATE EXTERNAL FILE FORMAT SynapseParquetFormat
WITH (
        FORMAT_TYPE = PARQUET
    );

--CSV
CREATE EXTERNAL FILE FORMAT QuotedCsvWithHeaderFormat
WITH (
        FORMAT_TYPE = DELIMITEDTEXT,
        FORMAT_OPTIONS (
                            PARSER_VERSION = '2.0',
                            FIELD_TERMINATOR = '|',
                            STRING_DELIMITER = '"',
                            FIRST_ROW = 2 )
);
```

Parquet is a columnar file format which stores the data, schema and statistics

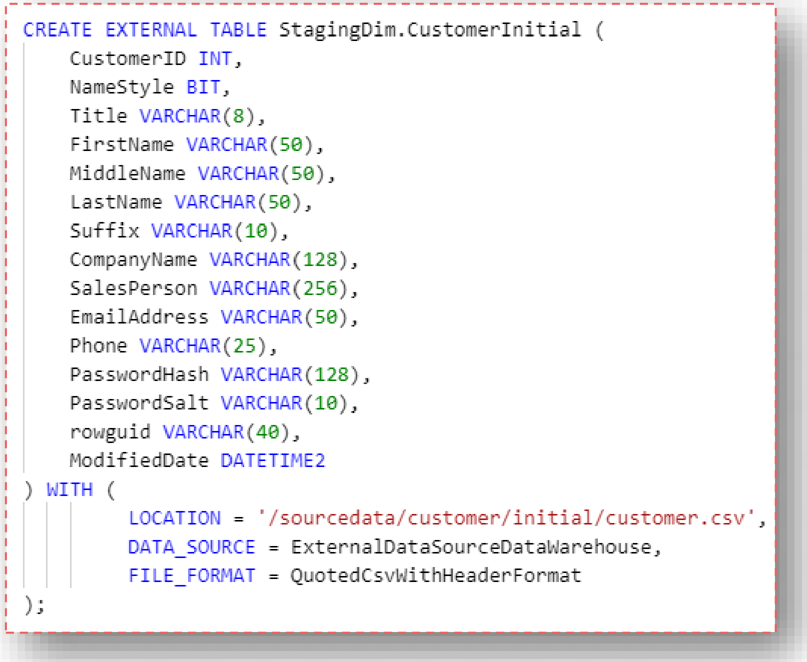
CSV format to match the source data

Create External Tables

Creating External Tables that point to the source CSV initial load location.

For each table used for a Dimension

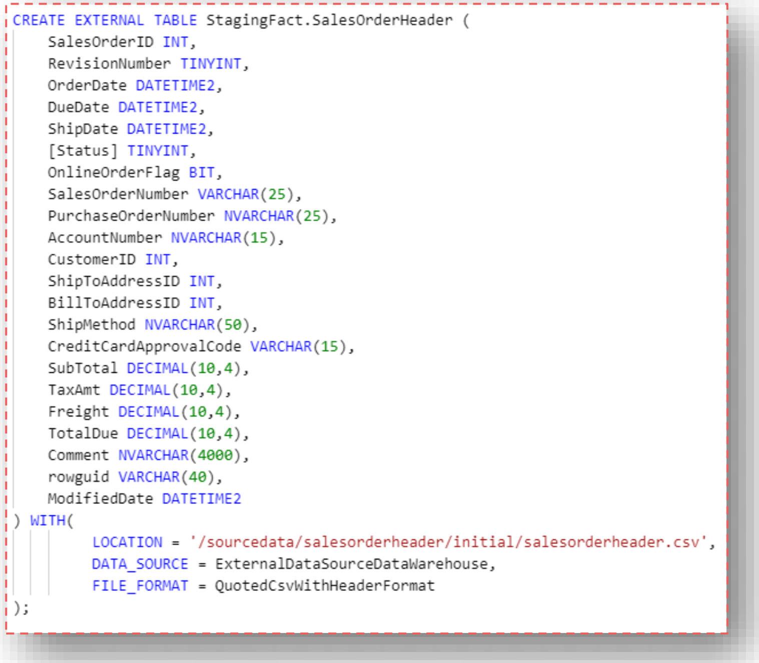
Create External table in StagingDim Schema pointing to source CSV file in “initial” folder.



```
CREATE EXTERNAL TABLE StagingDim.CustomerInitial (
    CustomerID INT,
    NameStyle BIT,
    Title VARCHAR(8),
    FirstName VARCHAR(50),
    MiddleName VARCHAR(50),
    LastName VARCHAR(50),
    Suffix VARCHAR(10),
    CompanyName VARCHAR(128),
    SalesPerson VARCHAR(256),
    EmailAddress VARCHAR(50),
    Phone VARCHAR(25),
    PasswordHash VARCHAR(128),
    PasswordSalt VARCHAR(10),
    rowguid VARCHAR(40),
    ModifiedDate DATETIME2
) WITH (
         LOCATION = '/sourcedata/customer/initial/customer.csv',
         DATA_SOURCE = ExternalDataSourceDataWarehouse,
         FILE_FORMAT = QuotedCsvWithHeaderFormat
);
```

For each table used in a Fact

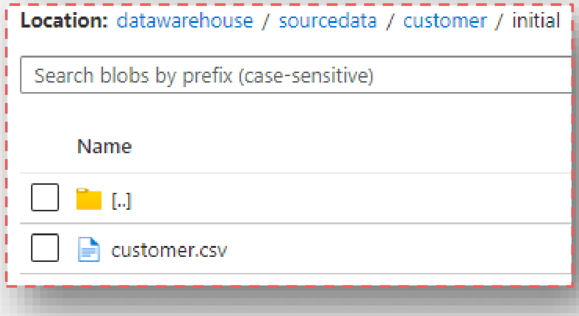
Create External table in StagingFact Schema pointing to source CSV file in “initial” folder.



```
CREATE EXTERNAL TABLE StagingFact.SalesOrderHeader (
    SalesOrderID INT,
    RevisionNumber TINYINT,
    OrderDate DATETIME2,
    DueDate DATETIME2,
    ShipDate DATETIME2,
    [Status] TINYINT,
    OnlineOrderFlag BIT,
    SalesOrderNumber VARCHAR(25),
    PurchaseOrderNumber NVARCHAR(25),
    AccountNumber NVARCHAR(15),
    CustomerID INT,
    ShipToAddressID INT,
    BillToAddressID INT,
    ShipMethod NVARCHAR(50),
    CreditCardApprovalCode VARCHAR(15),
    SubTotal DECIMAL(10,4),
    TaxAmt DECIMAL(10,4),
    Freight DECIMAL(10,4),
    TotalDue DECIMAL(10,4),
    Comment NVARCHAR(4000),
    rowguid VARCHAR(40),
    ModifiedDate DATETIME2
) WITH(
         LOCATION = '/sourcedata/salesorderheader/initial/salesorderheader.csv',
         DATA_SOURCE = ExternalDataSourceDataWarehouse,
         FILE_FORMAT = QuotedCsvWithHeaderFormat
);
```

Initial Dimension Loading

Extracting source data from the CSV file and loading to the Data Lake

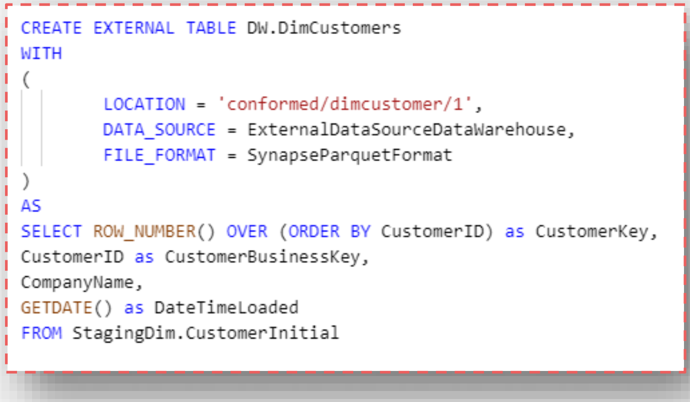


Location: datawarehouse / sourcedata / customer / initial

Search blobs by prefix (case-sensitive)

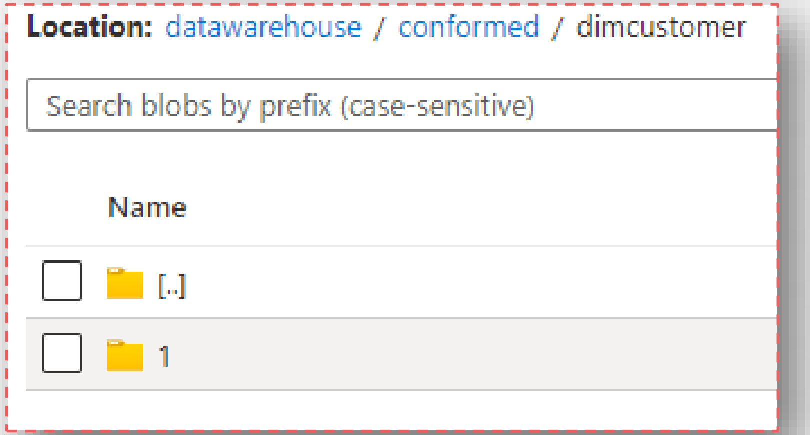
	Name
□	[..]
□	customer.csv

SELECT



```
CREATE EXTERNAL TABLE DW.DimCustomers
WITH
(
        LOCATION = 'conformed/dimcustomer/1',
        DATA_SOURCE = ExternalDataSourceDataWarehouse,
        FILE_FORMAT = SynapseParquetFormat
)
AS
SELECT ROW_NUMBER() OVER (ORDER BY CustomerID) as CustomerKey,
CustomerID as CustomerBusinessKey,
CompanyName,
GETDATE() as DateTimeLoaded
FROM StagingDim.CustomerInitial
```

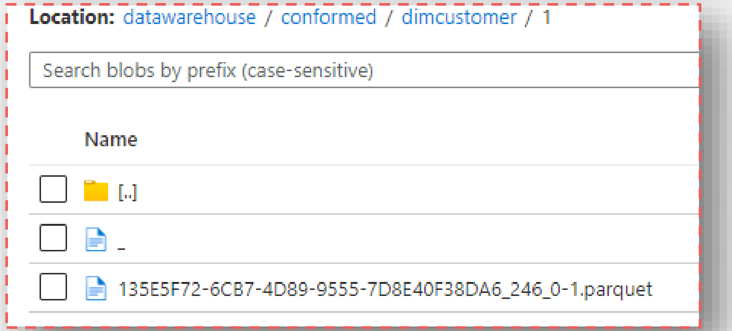
CREATE



Location: datawarehouse / conformed / dimcustomer

Search blobs by prefix (case-sensitive)

	Name
□	[..]
□	1



Location: datawarehouse / conformed / dimcustomer / 1

Search blobs by prefix (case-sensitive)

	Name
□	[..]
□	_
□	135E5F72-6CB7-4D89-9555-7D8E40F38DA6_246_0-1.parquet

We now use the CREATE TABLE AS SELECT (CETAS) syntax to select the data from the source CSV and write the transformed data into the Data Lake as a Parquet file

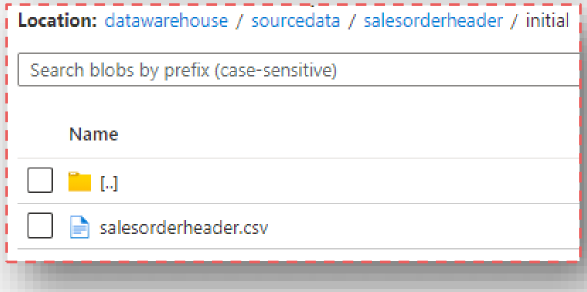
The initial load contains all of the current customer rows from the source database.

We can use a ROW_NUMBER() function to generate a Surrogate key for each dimension which will be used in the Fact table.

The Dimension data is loaded into a sequence number folder structure.

Initial Fact Loading

Extracting source data from the CSV file and loading to the Data Lake



Location: datawarehouse / sourcedata / salesorderheader / initial

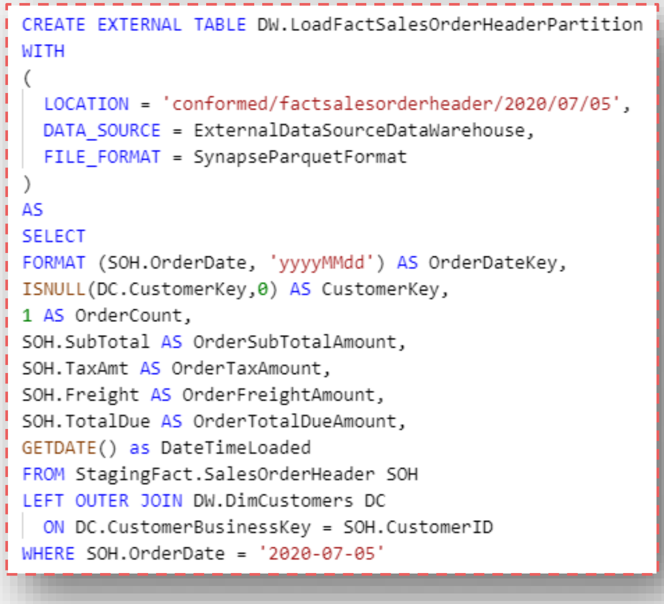
Search blobs by prefix (case-sensitive)

Name

□ [..]

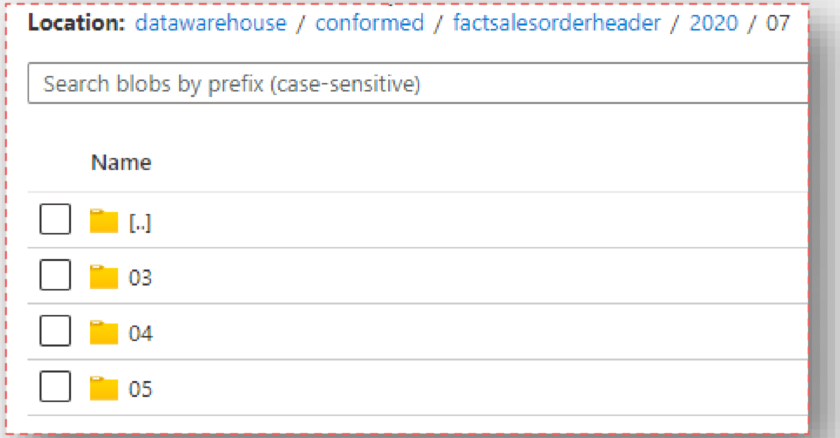
□ salesorderheader.csv

SELECT



```sql
CREATE EXTERNAL TABLE DW.LoadFactSalesOrderHeaderPartition
WITH
(
  LOCATION = 'conformed/factsalesorderheader/2020/07/05',
  DATA_SOURCE = ExternalDataSourceDataWarehouse,
  FILE_FORMAT = SynapseParquetFormat
)
AS
SELECT
FORMAT (SOH.OrderDate, 'yyyyMMdd') AS OrderDateKey,
ISNULL(DC.CustomerKey,0) AS CustomerKey,
1 AS OrderCount,
SOH.SubTotal AS OrderSubTotalAmount,
SOH.TaxAmt AS OrderTaxAmount,
SOH.Freight AS OrderFreightAmount,
SOH.TotalDue AS OrderTotalDueAmount,
GETDATE() as DateTimeLoaded
FROM StagingFact.SalesOrderHeader SOH
LEFT OUTER JOIN DW.DimCustomers DC
  ON DC.CustomerBusinessKey = SOH.CustomerID
WHERE SOH.OrderDate = '2020-07-05'
```

CREATE



Location: datawarehouse / conformed / factsalesorderheader / 2020 / 07

Search blobs by prefix (case-sensitive)

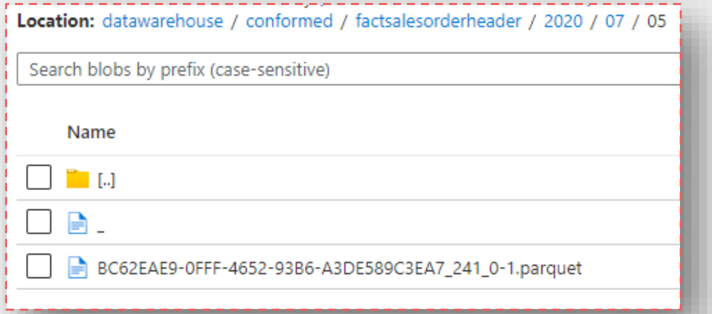
Name

□ [..]

□ 03

□ 04

□ 05



Location: datawarehouse / conformed / factsalesorderheader / 2020 / 07 / 05

Search blobs by prefix (case-sensitive)

Name

□ [..]

□ _

□ BC62EAE9-0FFF-4652-93B6-A3DE589C3EA7_241_0-1.parquet

As with the Dimension load, the CREATE TABLE AS SELECT (CETAS) syntax is used to select the data from the source CSV and write the transformed data into the Data Lake as a Parquet file

The initial load contains 3 days of Sales Data in a single CSV file.

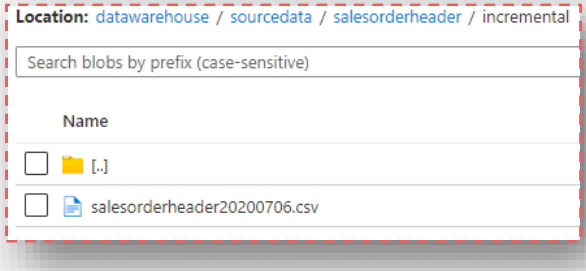
To extract each day, the whole CSV will need to be read by SQL Serverless.

Use CETAS statement to load the data to the Data Lake in a Year-Month-Day "partitioned" structure.

The "Date" (highlighted) can be parameterised using Dynamic SQL

Incremental Loading

We can incrementally load data by using the “partition” folder structure



Location: datawarehouse / sourcedata / salesorderheader / incremental

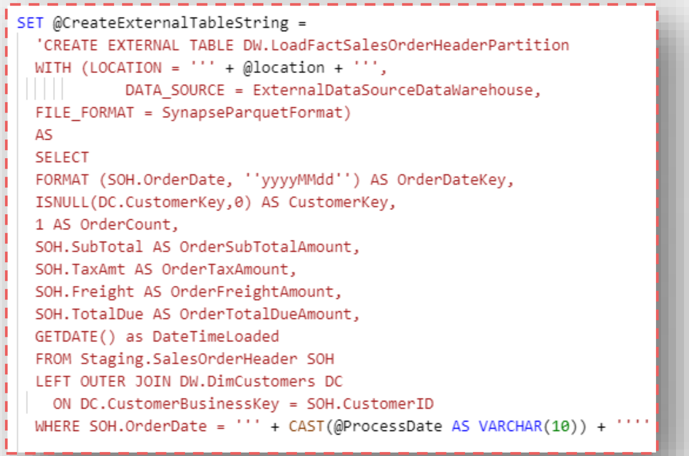
Search blobs by prefix (case-sensitive)

Name

□ [..]

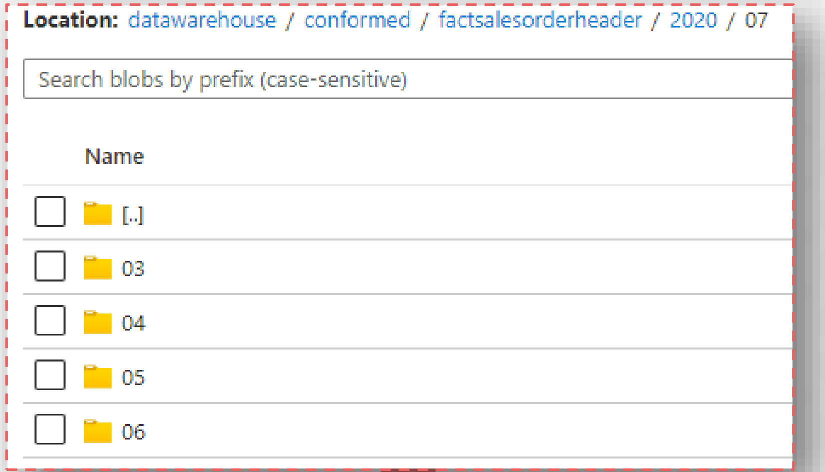
□ salesorderheader20200706.csv

SELECT



```
SET @CreateExternalTableString =
 'CREATE EXTERNAL TABLE DW.LoadFactSalesOrderHeaderPartition
 WITH (LOCATION = ''' + @location + ''',
           DATA_SOURCE = ExternalDataSourceDataWarehouse,
 FILE_FORMAT = SynapseParquetFormat)
 AS
 SELECT
 FORMAT (SOH.OrderDate, ''yyyyMMdd'') AS OrderDateKey,
 ISNULL(DC.CustomerKey,0) AS CustomerKey,
 1 AS OrderCount,
 SOH.SubTotal AS OrderSubTotalAmount,
 SOH.TaxAmt AS OrderTaxAmount,
 SOH.Freight AS OrderFreightAmount,
 SOH.TotalDue AS OrderTotalDueAmount,
 GETDATE() as DateTimeLoaded
 FROM Staging.SalesOrderHeader SOH
 LEFT OUTER JOIN DW.DimCustomers DC
   ON DC.CustomerBusinessKey = SOH.CustomerID
 WHERE SOH.OrderDate = ''' + CAST(@ProcessDate AS VARCHAR(10)) + ''''
```

CREATE



Location: datawarehouse / conformed / factsalesorderheader / 2020 / 07

Search blobs by prefix (case-sensitive)

Name

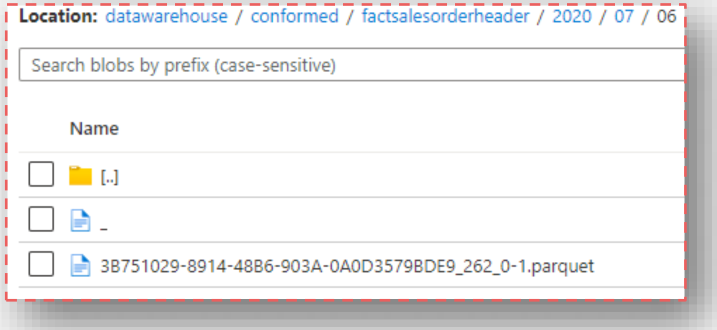
□ [..]

□ 03

□ 04

□ 05

□ 06



Location: datawarehouse / conformed / factsalesorderheader / 2020 / 07 / 06

Search blobs by prefix (case-sensitive)

Name

□ [..]

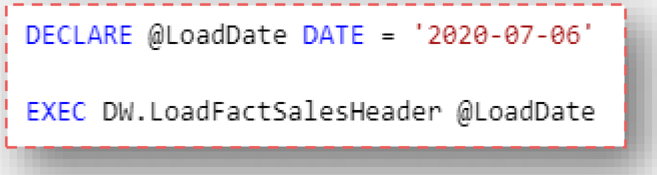
□ _

□ 3B751029-8914-48B6-903A-0A0D3579BDE9_262_0-1.parquet

We are treating the CETAS statement as a “staging” process to write the data to the Data Lake.

We can drop the External Table and the data will persist in the Data Lake.

The process is contained within a Stored Procedure in the Serverless SQL Pool database.



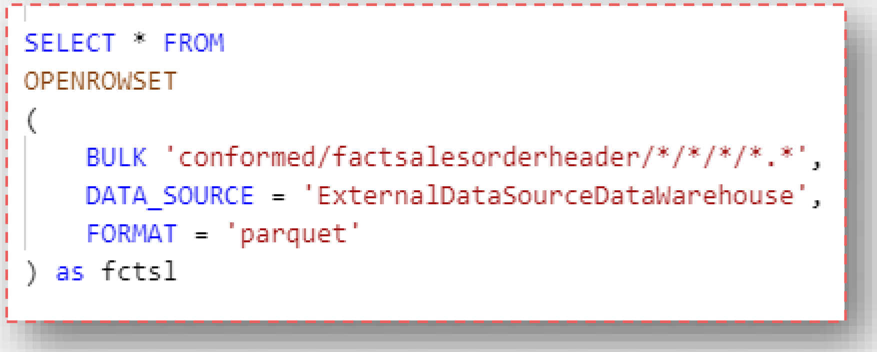
```
DECLARE @LoadDate DATE = '2020-07-06'

EXEC DW.LoadFactSalesHeader @LoadDate
```

The location to write the Parquet data to is parameterised along with the filter to SELECT the appropriate data.

Reading the Dimensional Data

We can SELECT data from the “partitions” that were created when running the CETAS process.

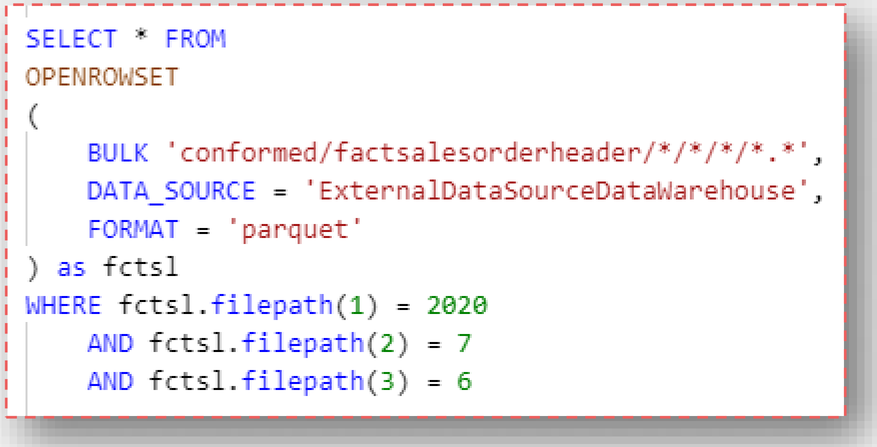
Selecting all the data available



```
SELECT * FROM
OPENROWSET
(
    BULK 'conformed/factsalesorderheader/*/*/*/*.*',
    DATA_SOURCE = 'ExternalDataSourceDataWarehouse',
    FORMAT = 'parquet'
) as fctsl
```

Using wildcards in the location to recursively select
Data from all the sub-folders

Selecting a specific “partition” of data



```
SELECT * FROM
OPENROWSET
(
    BULK 'conformed/factsalesorderheader/*/*/*/*.*',
    DATA_SOURCE = 'ExternalDataSourceDataWarehouse',
    FORMAT = 'parquet'
) as fctsl
WHERE fctsl.filepath(1) = 2020
    AND fctsl.filepath(2) = 7
    AND fctsl.filepath(3) = 6
```

Using the “filepath” function to select a specific folder of data

Considerations

- The cost model for both Reading and Writing data is based on the amount of data processed, not time or processing power.
- No caching of data retrieved, the same query touching the same data will incur costs.
- Currently data is immutable, it cannot be UPDATEd so your processes must take this into consideration

References & Further Reading

Synapse Analytics SQL Serverless

- https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/on-demand-workspace-overview
- https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/best-practices-sql-on-demand
- https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/on-demand-workspace-overview
- https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/data-processed
- https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/overview-features
- https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/develop-tables-statistics
- https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/create-use-views
- https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/create-external-table-as-select

Data Lakehouse

- https://databricks.com/blog/2020/01/30/what-is-a-data-lakehouse.html

Dimensional Modelling

- https://www.kimballgroup.com/data-warehouse-business-intelligence-resources/kimball-techniques/dimensional-modeling-techniques/

Parquet File Format
- https://parquet.apache.org/documentation/latest/

SCAN ME