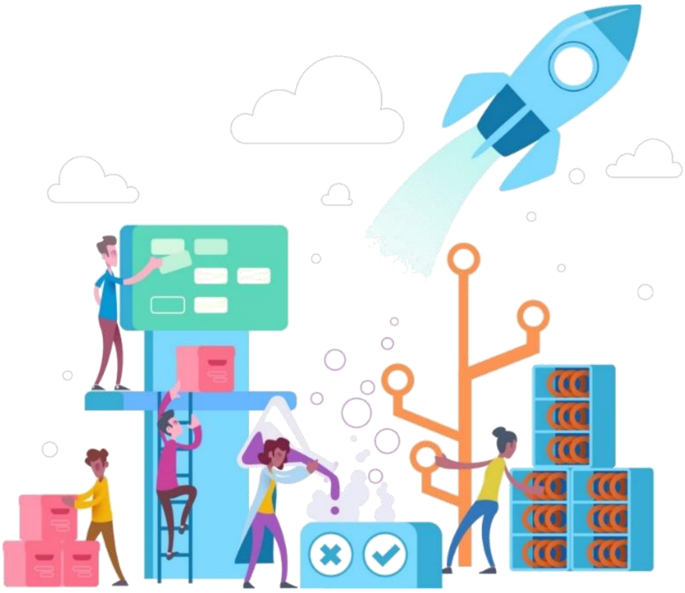
# MonkeyDWH™

## A monkey-proof DWH architecture (without a vendor lock)



Version 2020-11-12

## Table of contents

1 Introduction & context 3

2 What is MonkeyDWH™ 3

3 Benefits of MonkeyDWH™ 3

4 A monkey-proof architecture 4

5 Azure Resource Groups 5

6 Data Warehouse (DWH) objects 6

7 Data Mart (DM) objects 7

8 Azure Data Factory 8

9 Benchmark 10

10 Azure DevOps (CI&CD) with automatic unit testing 10

11 Tooling 11

12 License 12

13 Pricing 12

14 Planning 13

15 Questions & Answers 14

16 About Monkey Consultancy B.V. 15

17 Contact information 15

# 1 Introduction & context

With today's tight labor market, every hour spend on development has to count. It can easily takes a Data Engineer hours or even days to add additional source tables to a DWH with manual development.

The lack of experienced engineers within a company, or employees leaving a company, raise a risk on any DWH project. Let alone, keeping up with the fast pace of changes within the Azure data landscape, demands discipline and regular training.

# 2 What is MonkeyDWH™

The MonkeyDWH™ solution generates Azure SQL database and Azure Data Factory (ADF) objects that enables a company to quickly setup a Data Warehouse (DWH) on Azure in a matter of days.

With fully implemented Azure DevOps (CI&CD) pipelines, all new code will be build and deployed automatically. And with the automatic unit testing after each release, it's also able to automatically release new version to the next environment (optionally without manual approvals).

# 3 Benefits of MonkeyDWH™

### Future proof

By exclusively using native Azure Data Factory pipelines and activities, and thus without the use of an SSIS Integration Runtime (SSIS-IR), MonkeyDWH™ is easily adaptable to newly added ADF features or (code) changes in the future. MonkeyDWH™ uses Azure's own resource API's, which eliminates any custom PowerShell code or Runbooks to manage resources.

### No vendor lock

The MonkeyDWH™ is pure Microsoft-only, it generates SQL database- and ADF-objects without any encrypted code. It runs on your own Azure environment, no need for a hosted SaaS platform.

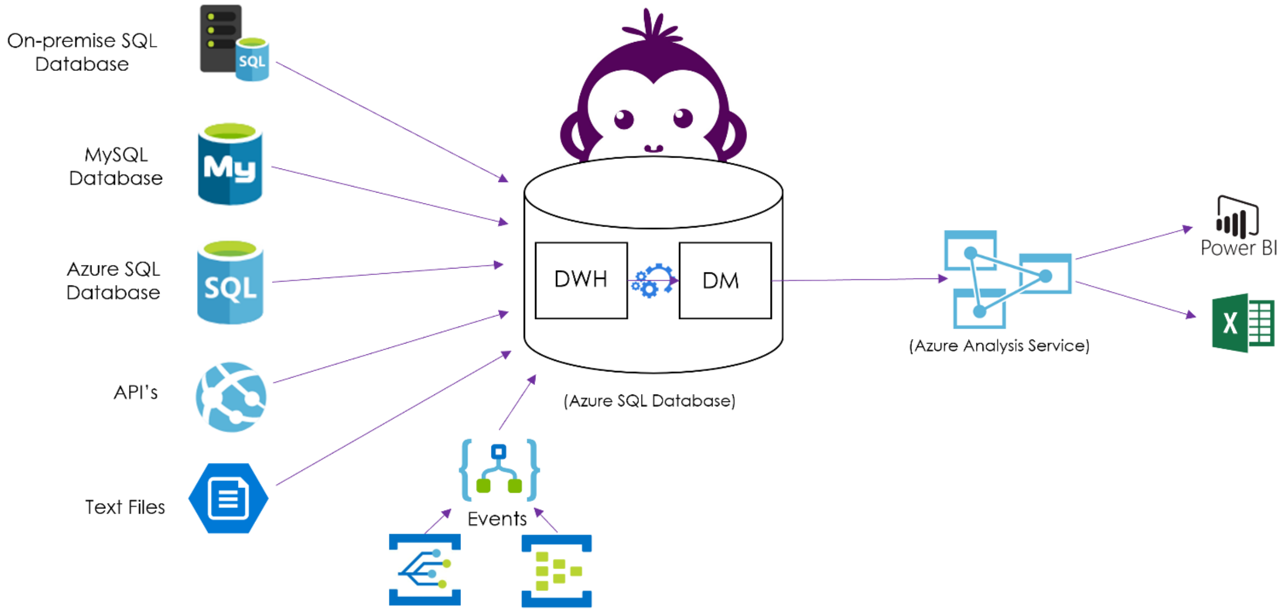
### Low runtime costs

Thanks to its architectural design, runtime costs are kept low due to the lack of expensive resources which are billed per month and can't be paused automatically. The DWH and Data Mart (DM) for example, use a single (Serverless) Azure SQL Database for which automatically pauses after an hour. Azure Data Factory (ETL) activities are billed per execution and can be grouped by executions per hour or per day, executing these pipelines only when needed. The costs are even lower thanks to the Azure SQL database instead of the more expensive Azure Synapse.

### Maintenance updates and support

The solution will be kept in good shape, thanks to regular maintenance updates. And with the help from technical support, any concerns or issues regarding your DWH will be attended to.

# 4 A monkey-proof architecture

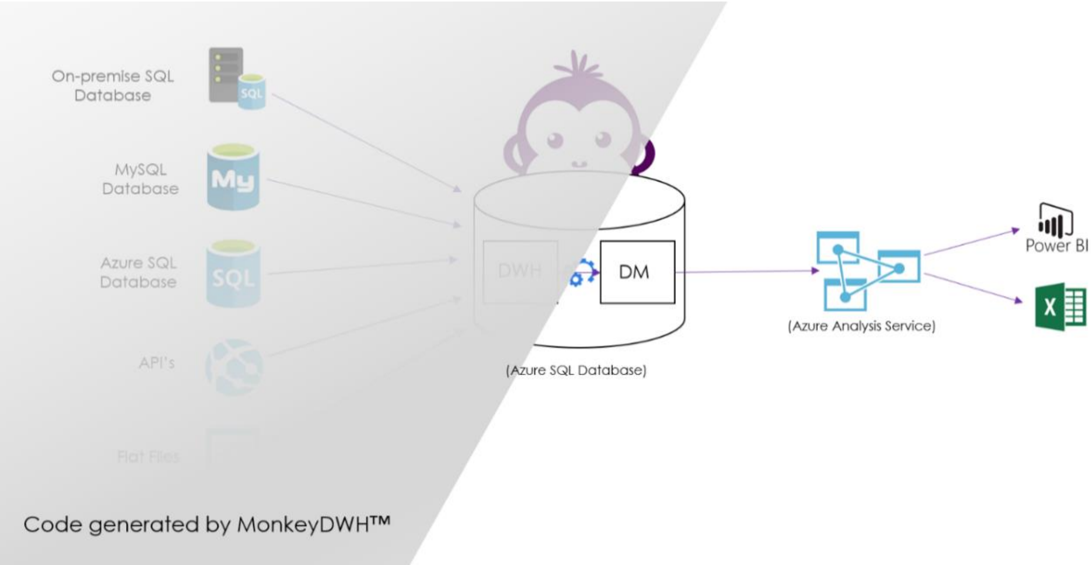


The overall architectural design is seen here, including all supported sources.

Data from different kind of sources is extracted, stored in staging-tables and new changes will be processed by the DWH. The Data Mart contains the dimensional model and will be populated using stored procedures inside the Azure SQL database. After that the Azure Analyses Services Tabular Model will be processed and made available for end-users.

End-users will use Excel PivotTables and Power BI dashboards to analyze the facts, dimensions and KPI's within the Tabular Model. Data Scientists are able to directly query the 'raw' data inside the DWH tables (which are identical to the source tables) or analyze prepared data inside the DM or Tabular Model.

The maximum compressed data volume for the DWH on Business Critical is 4,0TB, but with Hyperscale up to 100TB of storage can be allocated.



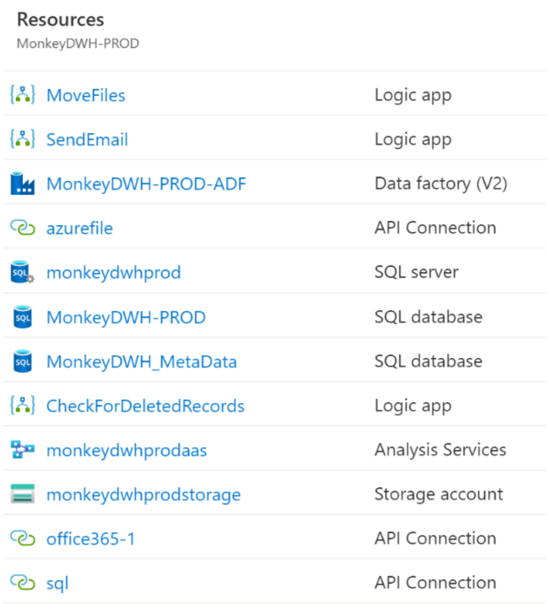
The MonkeyDWH™ generates all Azure SQL database- and Data Factory objects needed to extract data from different kind of sources and store data historically inside the DWH. It also generate database tables for the DM and the required ADF pipelines and datasets.

Due to this feature, a Data Engineer doesn't have to spend his/her precious development time on this 'prepping-phase'. He/she is able to fully focus on creating insights, designing an Analysis Service tabular model (also known as a 'cube') and creating Power BI dashboards for the business.

The DWH project would be delivering results (as in usable insights via Power BI dashboards) within a single month, a high ROI with low budgetable costs.

# 5 Azure Resource Groups

An example of an Azure Resource Group can be seen below. This is how the architecture will look like within the Azure Portal:



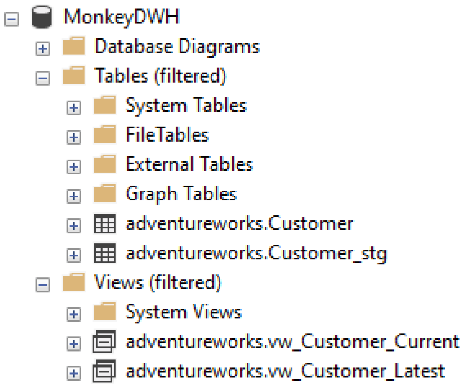
**Resources**
MonkeyDWH-PROD

| | |
|---|---|
| MoveFiles | Logic app |
| SendEmail | Logic app |
| MonkeyDWH-PROD-ADF | Data factory (V2) |
| azurefile | API Connection |
| monkeydwhprod | SQL server |
| MonkeyDWH-PROD | SQL database |
| MonkeyDWH_MetaData | SQL database |
| CheckForDeletedRecords | Logic app |
| monkeydwhprodaas | Analysis Services |
| monkeydwhprodstorage | Storage account |
| office365-1 | API Connection |
| sql | API Connection |

# 6 Data Warehouse (DWH) objects

A DWH can be seen as sort of a flight recorder, it searches for changes in the data and records it. Each change will be recorded with full history (inserts, updates and optionally also deletes).

Example DWH objects for ‘Customer’ can be seen here:

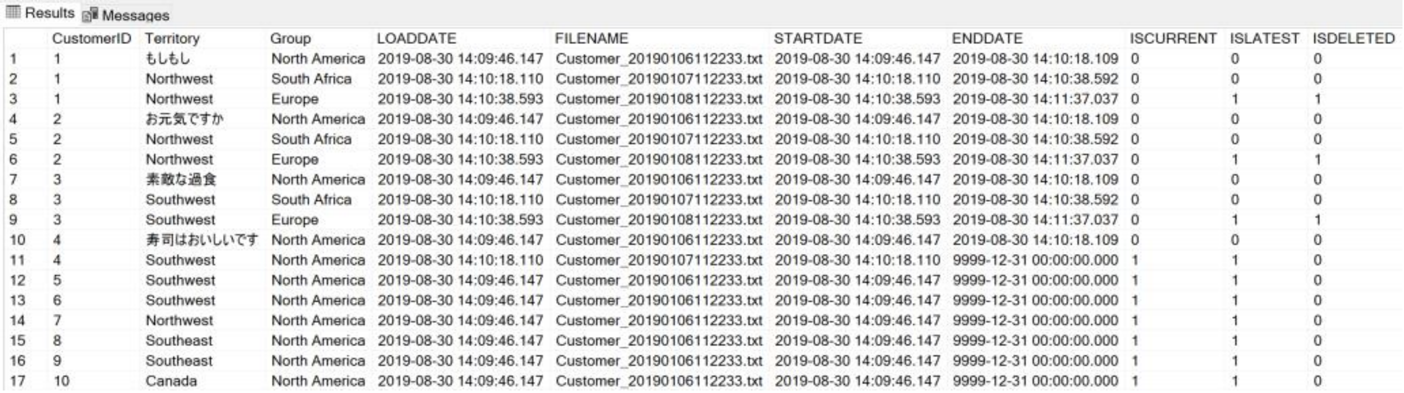


- MonkeyDWH
  - Database Diagrams
  - Tables (filtered)
    - System Tables
    - FileTables
    - External Tables
    - Graph Tables
    - adventureworks.Customer
    - adventureworks.Customer_stg
  - Views (filtered)
    - System Views
    - adventureworks.vw_Customer_Current
    - adventureworks.vw_Customer_Latest

The database contains a lot of database objects per source table (such as ‘customer’):

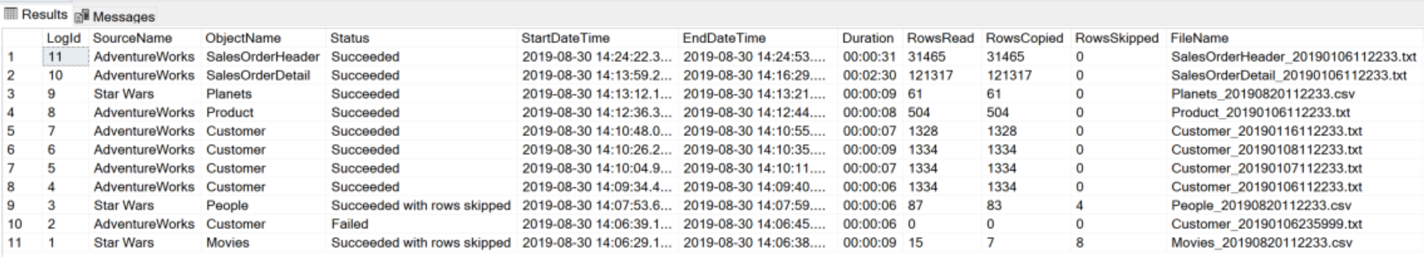
- Two tables for each source table
- Two views to (quickly) analyze <u>todays</u> current- or latest -state of the data
- Several logging objects

The generated views provide quick insights on all historical changes. They also include start- and end-dates, and booleans for IsCurrent, IsLatest and IsDeleted as shown below:



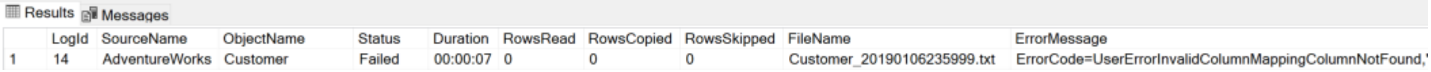
| | CustomerID | Territory | Group | LOADDATE | FILENAME | STARTDATE | ENDDATE | ISCURRENT | ISLATEST | ISDELETED |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | もしもし | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 2019-08-30 14:10:18.109 | 0 | 0 | 0 |
| 2 | 1 | Northwest | South Africa | 2019-08-30 14:10:18.110 | Customer_20190107112233.txt | 2019-08-30 14:10:18.110 | 2019-08-30 14:10:38.592 | 0 | 0 | 0 |
| 3 | 1 | Northwest | Europe | 2019-08-30 14:10:38.593 | Customer_20190108112233.txt | 2019-08-30 14:10:38.593 | 2019-08-30 14:11:37.037 | 0 | 1 | 1 |
| 4 | 2 | お元気ですか | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 2019-08-30 14:10:18.109 | 0 | 0 | 0 |
| 5 | 2 | Northwest | South Africa | 2019-08-30 14:10:18.110 | Customer_20190107112233.txt | 2019-08-30 14:10:18.110 | 2019-08-30 14:10:38.592 | 0 | 0 | 0 |
| 6 | 2 | Northwest | Europe | 2019-08-30 14:10:38.593 | Customer_20190108112233.txt | 2019-08-30 14:10:38.593 | 2019-08-30 14:11:37.037 | 0 | 1 | 1 |
| 7 | 3 | 素敵な過食 | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 2019-08-30 14:10:18.109 | 0 | 0 | 0 |
| 8 | 3 | Southwest | South Africa | 2019-08-30 14:10:18.110 | Customer_20190107112233.txt | 2019-08-30 14:10:18.110 | 2019-08-30 14:10:38.592 | 0 | 0 | 0 |
| 9 | 3 | Southwest | Europe | 2019-08-30 14:10:38.593 | Customer_20190108112233.txt | 2019-08-30 14:10:38.593 | 2019-08-30 14:11:37.037 | 0 | 1 | 1 |
| 10 | 4 | 寿司はおいしいです | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 2019-08-30 14:10:18.109 | 0 | 0 | 0 |
| 11 | 4 | Southwest | North America | 2019-08-30 14:10:18.110 | Customer_20190107112233.txt | 2019-08-30 14:10:18.110 | 9999-12-31 00:00:00.000 | 1 | 1 | 0 |
| 12 | 5 | Southwest | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 9999-12-31 00:00:00.000 | 1 | 1 | 0 |
| 13 | 6 | Southwest | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 9999-12-31 00:00:00.000 | 1 | 1 | 0 |
| 14 | 7 | Northwest | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 9999-12-31 00:00:00.000 | 1 | 1 | 0 |
| 15 | 8 | Southeast | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 9999-12-31 00:00:00.000 | 1 | 1 | 0 |
| 16 | 9 | Southeast | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 9999-12-31 00:00:00.000 | 1 | 1 | 0 |
| 17 | 10 | Canada | North America | 2019-08-30 14:09:46.147 | Customer_20190106112233.txt | 2019-08-30 14:09:46.147 | 9999-12-31 00:00:00.000 | 1 | 1 | 0 |

Within the DWH database, two logging tables are available. One for recording all ADF Copy activities while reading and copying data into the staging-tables. Invalid records are skipped by default, but are still recorded in this logging-table (see ‘*RowsSkipped*’):



| | LogId | SourceName | ObjectName | Status | StartDateTime | EndDateTime | Duration | RowsRead | RowsCopied | RowsSkipped | FileName |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 11 | AdventureWorks | SalesOrderHeader | Succeeded | 2019-08-30 14:24:22.3... | 2019-08-30 14:24:53.... | 00:00:31 | 31465 | 31465 | 0 | SalesOrderHeader_20190106112233.txt |
| 2 | 10 | AdventureWorks | SalesOrderDetail | Succeeded | 2019-08-30 14:13:59.2... | 2019-08-30 14:16:29.... | 00:02:30 | 121317 | 121317 | 0 | SalesOrderDetail_20190106112233.txt |
| 3 | 9 | Star Wars | Planets | Succeeded | 2019-08-30 14:13:12.1... | 2019-08-30 14:13:21.... | 00:00:09 | 61 | 61 | 0 | Planets_20190820112233.csv |
| 4 | 8 | AdventureWorks | Product | Succeeded | 2019-08-30 14:12:36.3... | 2019-08-30 14:12:44.... | 00:00:08 | 504 | 504 | 0 | Product_20190106112233.txt |
| 5 | 7 | AdventureWorks | Customer | Succeeded | 2019-08-30 14:10:48.0... | 2019-08-30 14:10:55.... | 00:00:07 | 1328 | 1328 | 0 | Customer_20190116112233.txt |
| 6 | 6 | AdventureWorks | Customer | Succeeded | 2019-08-30 14:10:26.2... | 2019-08-30 14:10:35.... | 00:00:09 | 1334 | 1334 | 0 | Customer_20190108112233.txt |
| 7 | 5 | AdventureWorks | Customer | Succeeded | 2019-08-30 14:10:04.9... | 2019-08-30 14:10:11.... | 00:00:07 | 1334 | 1334 | 0 | Customer_20190107112233.txt |
| 8 | 4 | AdventureWorks | Customer | Succeeded | 2019-08-30 14:09:34.4... | 2019-08-30 14:09:40.... | 00:00:06 | 1334 | 1334 | 0 | Customer_20190106112233.txt |
| 9 | 3 | Star Wars | People | Succeeded with rows skipped | 2019-08-30 14:07:53.6... | 2019-08-30 14:07:59.... | 00:00:06 | 87 | 83 | 4 | People_20190820112233.csv |
| 10 | 2 | AdventureWorks | Customer | Failed | 2019-08-30 14:06:39.1... | 2019-08-30 14:06:45.... | 00:00:06 | 0 | 0 | 0 | Customer_20190106235999.txt |
| 11 | 1 | Star Wars | Movies | Succeeded with rows skipped | 2019-08-30 14:06:29.1... | 2019-08-30 14:06:38.... | 00:00:09 | 15 | 7 | 8 | Movies_20190820112233.csv |

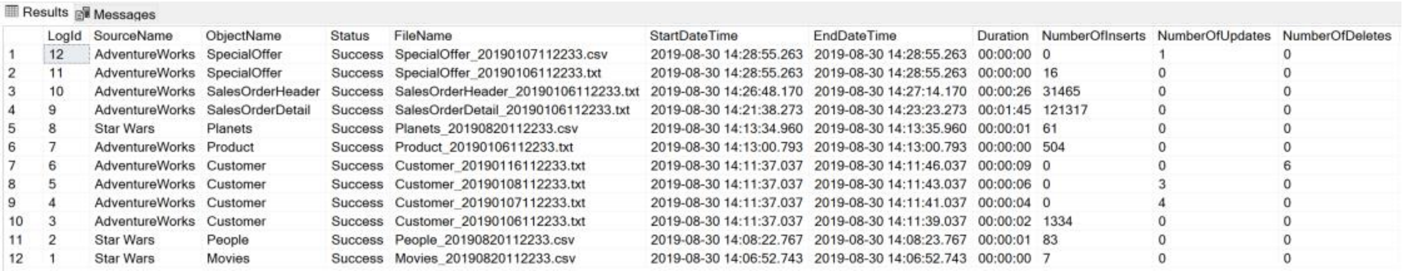
Failed ADF Copy activities are also recorded properly:



Results | Messages

|   | LogId | SourceName | ObjectName | Status | Duration | RowsRead | RowsCopied | RowsSkipped | FileName | ErrorMessage |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 14 | AdventureWorks | Customer | Failed | 00:00:07 | 0 | 0 | 0 | Customer_20190106235999.txt | ErrorCode=UserErrorInvalidColumnMappingColumnNotFound,' |

The other logging-table records the output from each 'processing stored procedure'. Each insert, update and delete is properly recorded:

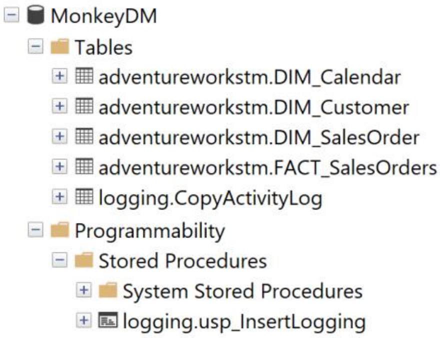


Results | Messages

|   | LogId | SourceName | ObjectName | Status | FileName | StartDateTime | EndDateTime | Duration | NumberOfInserts | NumberOfUpdates | NumberOfDeletes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 12 | AdventureWorks | SpecialOffer | Success | SpecialOffer_20190107112233.csv | 2019-08-30 14:28:55.263 | 2019-08-30 14:28:55.263 | 00:00:00 | 0 | 1 | 0 |
| 2 | 11 | AdventureWorks | SpecialOffer | Success | SpecialOffer_20190106112233.txt | 2019-08-30 14:28:55.263 | 2019-08-30 14:28:55.263 | 00:00:00 | 16 | 0 | 0 |
| 3 | 10 | AdventureWorks | SalesOrderHeader | Success | SalesOrderHeader_20190106112233.txt | 2019-08-30 14:26:48.170 | 2019-08-30 14:27:14.170 | 00:00:26 | 31465 | 0 | 0 |
| 4 | 9 | AdventureWorks | SalesOrderDetail | Success | SalesOrderDetail_20190106112233.txt | 2019-08-30 14:21:38.273 | 2019-08-30 14:23:23.273 | 00:01:45 | 121317 | 0 | 0 |
| 5 | 8 | Star Wars | Planets | Success | Planets_20190820112233.csv | 2019-08-30 14:13:34.960 | 2019-08-30 14:13:35.960 | 00:00:01 | 61 | 0 | 0 |
| 6 | 7 | AdventureWorks | Product | Success | Product_20190106112233.txt | 2019-08-30 14:13:00.793 | 2019-08-30 14:13:00.793 | 00:00:00 | 504 | 0 | 0 |
| 7 | 6 | AdventureWorks | Customer | Success | Customer_20190116112233.txt | 2019-08-30 14:11:37.037 | 2019-08-30 14:11:46.037 | 00:00:09 | 0 | 0 | 6 |
| 8 | 5 | AdventureWorks | Customer | Success | Customer_20190108112233.txt | 2019-08-30 14:11:37.037 | 2019-08-30 14:11:43.037 | 00:00:06 | 0 | 3 | 0 |
| 9 | 4 | AdventureWorks | Customer | Success | Customer_20190107112233.txt | 2019-08-30 14:11:37.037 | 2019-08-30 14:11:41.037 | 00:00:04 | 0 | 4 | 0 |
| 10 | 3 | AdventureWorks | Customer | Success | Customer_20190106112233.txt | 2019-08-30 14:11:37.037 | 2019-08-30 14:11:39.037 | 00:00:02 | 1334 | 0 | 0 |
| 11 | 2 | Star Wars | People | Success | People_20190820112233.csv | 2019-08-30 14:08:22.767 | 2019-08-30 14:08:23.767 | 00:00:01 | 83 | 0 | 0 |
| 12 | 1 | Star Wars | Movies | Success | Movies_20190820112233.csv | 2019-08-30 14:06:52.743 | 2019-08-30 14:06:52.743 | 00:00:00 | 7 | 0 | 0 |

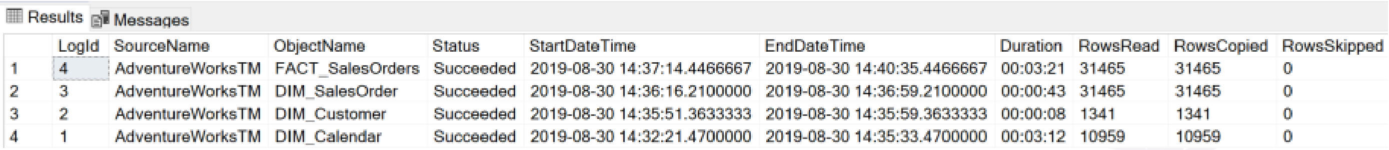
# 7 Data Mart (DM) objects

The DM objects will look like this:



```
MonkeyDM
  Tables
    adventureworkstm.DIM_Calendar
    adventureworkstm.DIM_Customer
    adventureworkstm.DIM_SalesOrder
    adventureworkstm.FACT_SalesOrders
    logging.CopyActivityLog
  Programmability
    Stored Procedures
      System Stored Procedures
      logging.usp_InsertLogging
```

The logging-table inside the DM database records all ADF Copy activities while populating the Data Mart. Invalid records are skipped, but recorded in this logging-table (see '*RowsSkipped*'):



Results | Messages

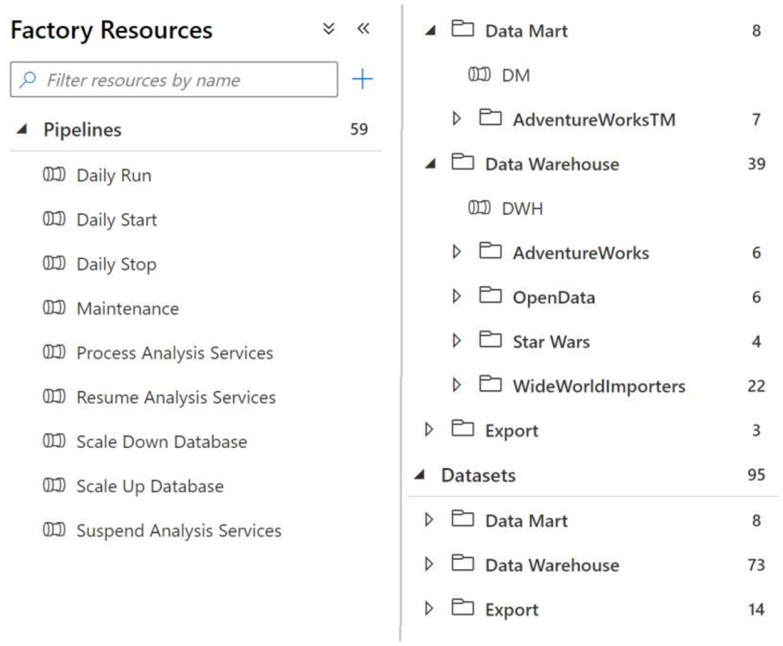
|   | LogId | SourceName | ObjectName | Status | StartDateTime | EndDateTime | Duration | RowsRead | RowsCopied | RowsSkipped |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 4 | AdventureWorksTM | FACT_SalesOrders | Succeeded | 2019-08-30 14:37:14.4466667 | 2019-08-30 14:40:35.4466667 | 00:03:21 | 31465 | 31465 | 0 |
| 2 | 3 | AdventureWorksTM | DIM_SalesOrder | Succeeded | 2019-08-30 14:36:16.2100000 | 2019-08-30 14:36:59.2100000 | 00:00:43 | 31465 | 31465 | 0 |
| 3 | 2 | AdventureWorksTM | DIM_Customer | Succeeded | 2019-08-30 14:35:51.3633333 | 2019-08-30 14:35:59.3633333 | 00:00:08 | 1341 | 1341 | 0 |
| 4 | 1 | AdventureWorksTM | DIM_Calendar | Succeeded | 2019-08-30 14:32:21.4700000 | 2019-08-30 14:35:33.4700000 | 00:03:12 | 10959 | 10959 | 0 |

# 8 Azure Data Factory

Azure Data Factory (ADF) is responsible for copying and processing all data in the Azure architecture.

Over 80+ different types of so called ‘connectors’ are natively supported by ADF, but in most cases, only a handful are used. MonkeyDWH™ supports CSV-files, JSON-files, API’s, Azure SQL databases, on-premise SQL databases and MySQL databases. Other types of sources can be added on request.

In the screenshot below, pipelines for DWH are generated for three different sample databases, namely PerformanceTest, Star Wars and WideWorldImporters. Pipelines for the AdventureWorks flat files (both CSV and TXT combined) are also included. The Data Mart (DM) contains a Fact and three Dimensions, all based upon four stored procedures as a source:



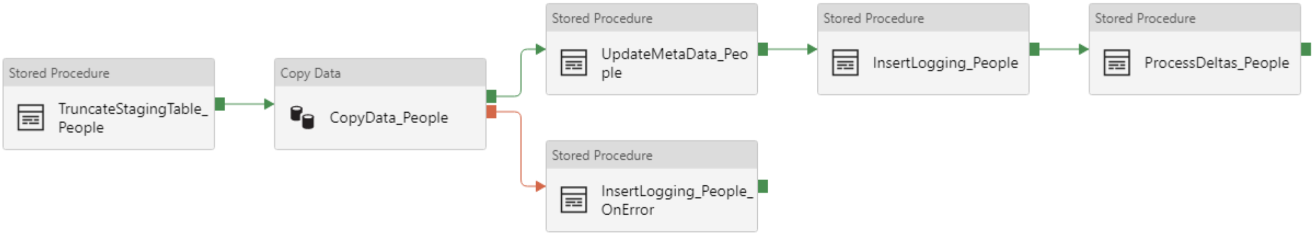
Factory Resources

Filter resources by name

- Pipelines 59
  - Daily Run
  - Daily Start
  - Daily Stop
  - Maintenance
  - Process Analysis Services
  - Resume Analysis Services
  - Scale Down Database
  - Scale Up Database
  - Suspend Analysis Services
  - Data Mart 8
    - DM
    - AdventureWorksTM 7
  - Data Warehouse 39
    - DWH
    - AdventureWorks 6
    - OpenData 6
    - Star Wars 4
    - WideWorldImporters 22
  - Export 3
- Datasets 95
  - Data Mart 8
  - Data Warehouse 73
  - Export 14

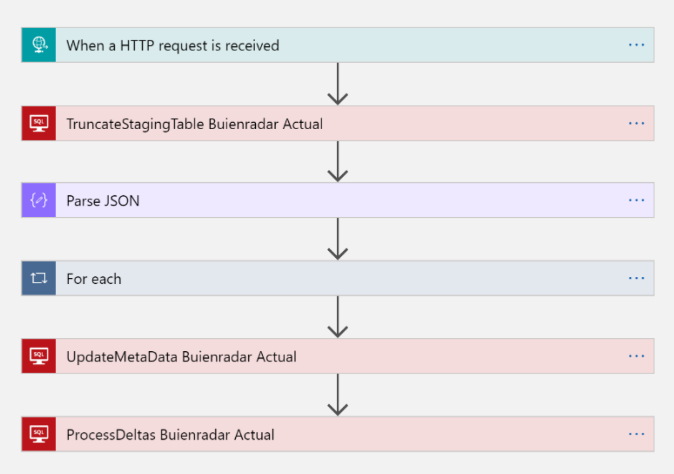
## Database as a source

In comparison, the pipeline below shows the required activities to pull data out of a source database and process it properly in the DWH:



## Event as a source

Event-driven architectures are becoming more and more mainstream, and processing such events within the Azure landscape, is also support by the MonkeyDWH™ solution. Events send by for example an Event Hub, Webhook or Event Grid Topic, will be handled by customized Logic Apps:



When a HTTP request is received

TruncateStagingTable Buienradar Actual

Parse JSON

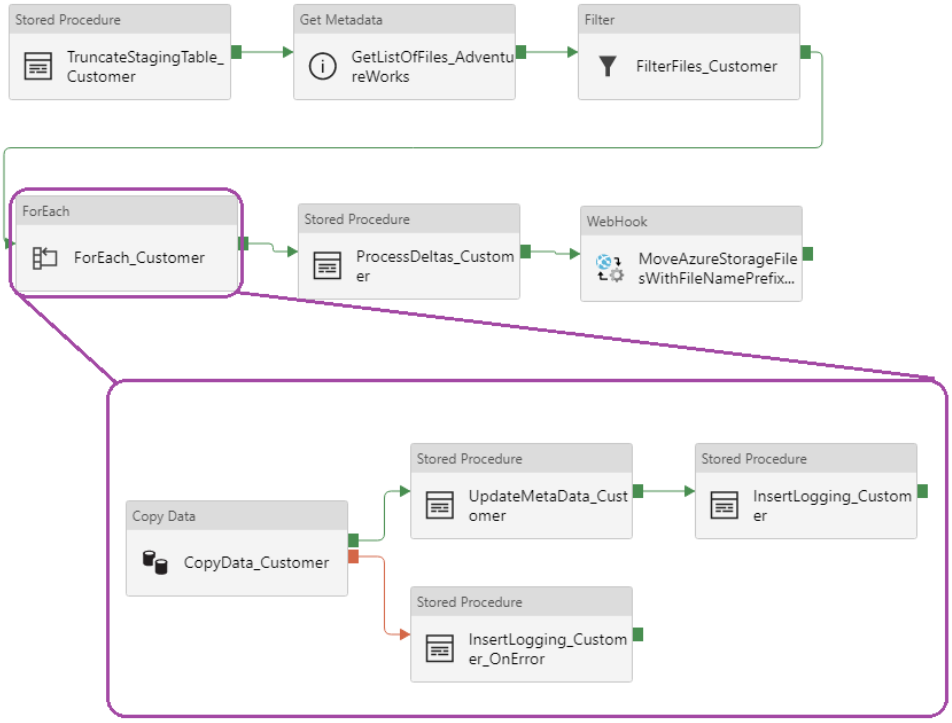
For each

UpdateMetaData Buienradar Actual

ProcessDeltas Buienradar Actual

## Flat File as a source

An inside view of an ADF pipeline can be seen below. This pipeline consists of only native ADF activities and processes flat files:



## 9 Benchmark

The DWH has four processing methods (configurable per source table), namely:

- Bulk Import
- Inserts Only
- Inserts & Updates
- Inserts, Updates & Deletes

During this benchmark, a total of 44 million records were ingested from an Azure SQL database on a different SQL Server and Resource Group. Eight ADF pipelines ran in parallel, copying the data from to source to the sink tables and processing the data depending on the different methods. It took no more than 28 minutes and 55 seconds:

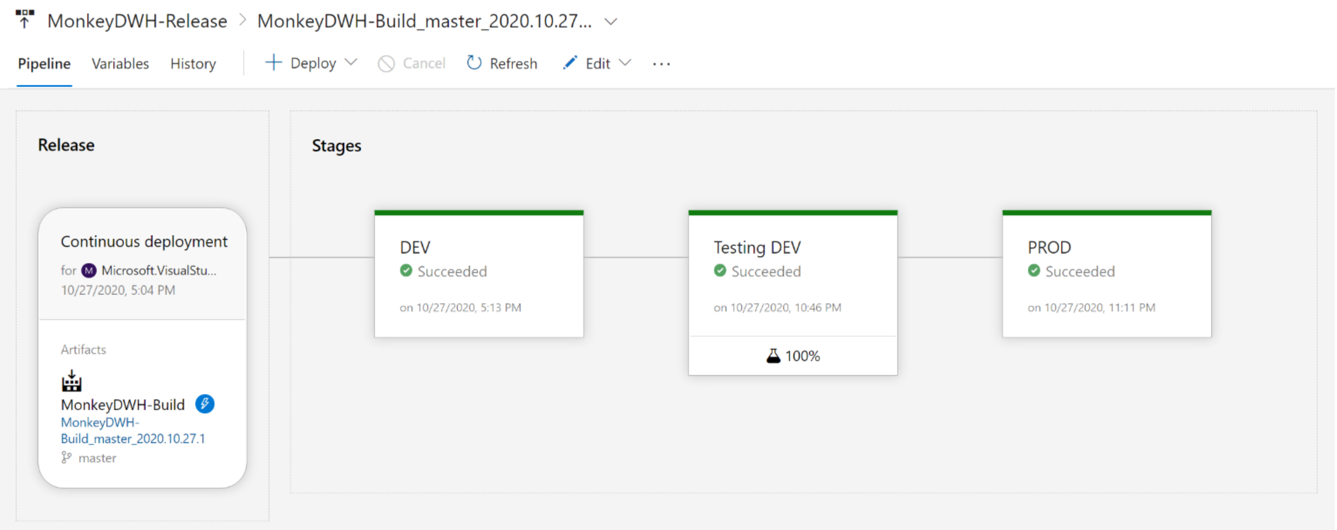
| | 1 million records | | 10 million records | |
|---|---|---|---|---|
| | Processing time | Total duration | Processing time | Total duration |
| Bulk Import | 01:02 | 02:10 | 14:31 | 26:21 |
| Inserts Only | 00:59 | 02:15 | 12:33 | 24:25 |
| Inserts & Updates | 01:00 | 02:09 | 12:39 | 23:34 |
| Inserts, Updates & Deletes | 01:11 | 02:21 | 16:31 | 28:55 |

(Running in parallel on Gen5 - *General Purpose - GP_S_Gen5_4)*

## 10 Azure DevOps (CI&CD) with automatic unit testing

The MonkeyDWH™ will be configured using Azure DevOps with Continuous Integration, Continuous Delivery or Continuous Deployment (CI/CD). Thanks to our automatic unit testing, we're able to implement Continuous Deployment to a Production-environment as seen below.

Azure DevOps Release Pipeline, including a stage for unit tests on the Development-environment:



Continuous deployment
for Microsoft.VisualStu...
10/27/2020, 5:04 PM

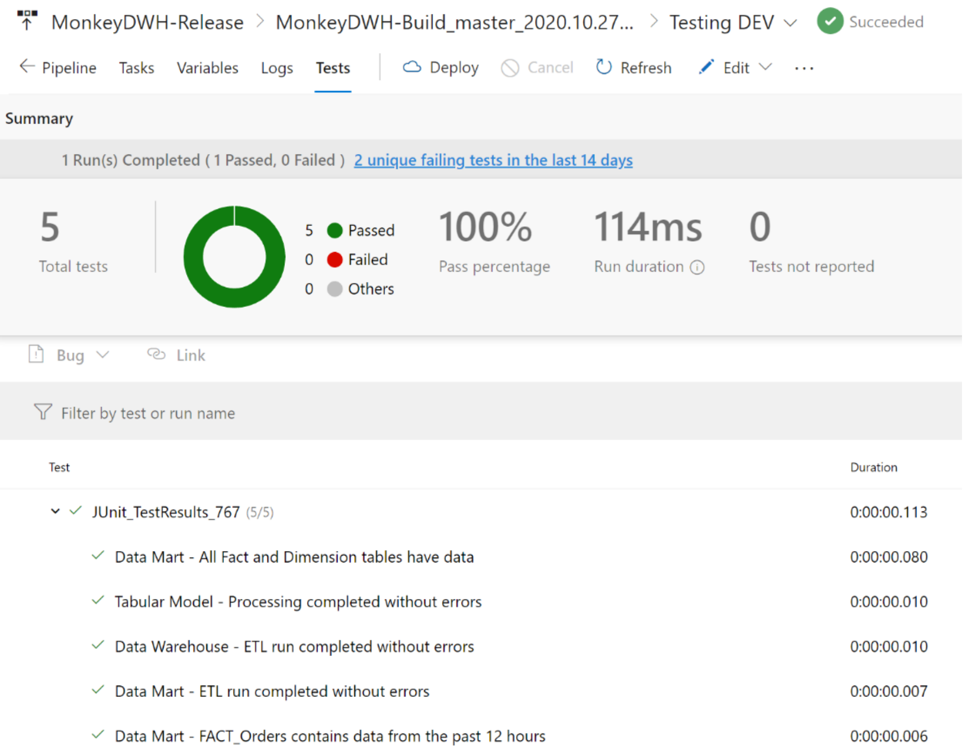
Artifacts
MonkeyDWH-Build
MonkeyDWH-Build_master_2020.10.27.1
master

DEV
Succeeded
on 10/27/2020, 5:13 PM

Testing DEV
Succeeded
on 10/27/2020, 10:46 PM
100%

PROD
Succeeded
on 10/27/2020, 11:11 PM

The results of the unit tests:



MonkeyDWH-Release > MonkeyDWH-Build_master_2020.10.27... > Testing DEV ✓ Succeeded

Pipeline Tasks Variables Logs **Tests** Deploy Cancel Refresh Edit ···

Summary

1 Run(s) Completed ( 1 Passed, 0 Failed ) 2 unique failing tests in the last 14 days

5 Total tests

5 ● Passed
0 ● Failed
0 ● Others

100% Pass percentage

114ms Run duration

0 Tests not reported

Bug Link

Filter by test or run name

| Test | Duration |
|---|---|
| ✓ JUnit_TestResults_767 (5/5) | 0:00:00.113 |
| ✓ Data Mart - All Fact and Dimension tables have data | 0:00:00.080 |
| ✓ Tabular Model - Processing completed without errors | 0:00:00.010 |
| ✓ Data Warehouse - ETL run completed without errors | 0:00:00.010 |
| ✓ Data Mart - ETL run completed without errors | 0:00:00.007 |
| ✓ Data Mart - FACT_Orders contains data from the past 12 hours | 0:00:00.006 |

## 11 Tooling

A solution which is future-proof and cloud native, means creating a solution which solely uses T-SQL coding in combination with meta-data, to generate all database- and ADF-objects.

Here's an overview of all tools:

About these tools:

- SQL Server is used locally for hosting the model database (for DWH and DM)
- Visual Studio will contain both database projects and also a project for the Analysis Services Tabular Model. This in combination with Git for code versioning
- Azure DevOps is used for building and releasing new code automatically (CI&CD)

# 12 License

The MonkeyDWH comes with a single license, namely:

MonkeyDWH™ perpetual license

This perpetual license for the MonkeyDWH™ allows the licensee to use the MonkeyDWH™ framework, coding, meta-data database and all the generated Azure Data Factory and SQL Database objects within their Azure and Azure DevOps environment. Technical updates are included for minor changes in the coding and/or new features.

# 13 Pricing

About the amount of peanuts…

License:

| | |
|---|---|
| - MonkeyDWH™ perpetual license | € 24.995,- (ca. € 521,- per month over 4 years) |
| - Azure DevOps CI&CD + automatic unit testing | € 4.995,- |

Monthly costs:

| | |
|---|---|
| - Azure running costs (PROD & DEV) | Starting from € 400,- up to € 1.400,- per month* |

Additional consultancy services:

- Support subscriptions
- Azure & DevOps training courses and workshops

* This estimation is based on two environments (DEV & PROD) with 5 GB to 50 GB of compressed data in the DWH, a daily load between 1 GB and 5 GB and an Azure Analysis Service (PROD) scaled at Standard S0.

# 14 Planning

Within a week, the new Azure & DevOps architecture is fully configured with sample data(bases):

Week 1:
- Setting up and configuring Azure
- Implementing Azure DevOps Build- and Release-pipelines for CI&CD
- Adding the first source to the MonkeyDWH™

Week 2:
- Adding additional sources to the MonkeyDWH™

Week 3:
- Creating the first Facts and Dimensions for the Dimensional Model (DataMart) and cube

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Initial setup of Azure & DevOps | ▓ | ▓ | ▓ | ░ | ░ | | | | | | | | | | |
| Adding Source #1 | | | | ▓ | ▓ | ░ | | | | | | | | | |
| Adding Source #2 | | | | | | ▓ | ▓ | ░ | | | | | | | |
| Adding Source #3 | | | | | | | | ▓ | ▓ | ░ | | | | | |
| Adding Source #4 | | | | | | | | | | ▓ | ▓ | ░ | | | |
| Creating the first Fact's and Dim's | | | | | | | | | | | ▓ | ▓ | ▓ | ░ | ░ |
| Training internal employees | | | ░ | ░ | ░ | ░ | ░ | ░ | ░ | ░ | ░ | ░ | ░ | ░ | ░ |

Knowledge and Skill Requirements:
Minimum set of skills to operate: common BI/DWH knowledge
(e.g. DWH, SCD, Dimensional modelling, Business Keys, T-SQL and Hashes)

# 15 Questions & Answers

Q) Why use a SQL database resource instead of Azure Synapse on Azure?

A) Most of our customers have far less than 100GB of compressed data. An Azure SQL database is ideal for these type of volumes, when looking at supported database features and low runtime costs. Azure Synapse Analytics (also known as an Azure SQL DWH) is more practical when dealing with at least 2,0TB of compressed data[1].

Q) When our compressed data volume exceeds 2,0TB, are we able to switch to Synapse Analytics instead?

A) Yes, but the current version of the MonkeyDWH™ solution doesn't natively support Azure Synapse Analytics. Different objects (especially indexes, partitions and queries) will need to be generated.
An important fact is that the maximum compressed data volume for the DWH on Business Critical is 4,0TB, but with Hyperscale up to 100TB of storage can be allocated.

Q) Are we able to export data from the DWH to flat files?

A) Yes, the MonkeyDWH™ solution has an export functionality, which extracts data from various database source types (e.g. Azure SQL-, On-premise SQL- and MySQL- databases) and export this data to flat files with different text formats. This functionality can be configured to export data from the DWH to an Azure File Storage (SMB) or Data Lake.

Q) Is it possible to include a Data Lake in the architecture?

A) Yes, with use of the export functionality (mentioned above), you're able to export data directly to a Data Lake. Importing data from Azure Data Lake Storage is possible with minimal custom development using Azure Data Factory Data Flows for example.

Q) Does the MonkeyDWH™ solution support the generation of a Data Vault 2.0 model?

A) No, not by default, but this would be possible with custom development.

Q) Where are all business- and transformation-rules located within the DWH?

A) The Data Factory pipelines execute stored procedures, which extract 'raw' data from the DWH tables and apply business- and transformation-rules towards a dimensional model.

Q) Is it possible to add an additional source type?

A) Yes, this is very much possible with minimal custom development.

[1] For more info, please visit: https://stackify.com/azure-sql-database-vs-warehouse/
and https://www.blue-granite.com/blog/is-azure-sql-data-warehouse-a-good-fit-updated/

# 16 About Monkey Consultancy B.V.

http://www.monkeyconsultancy.nl/

Monkey Consultancy is specialized in designing, generating and managing Data Warehouse (DWH) architectures on Azure. Especially combining these new architectures with Azure DevOps for being able to automatically build and release new versions of code. All Azure environments, combined with fully operational Azure DevOps (CI&CD) processes will be handed over to new customers in just a few weeks.

Over the past few years, we've shared our passion via the Monkey Update, a monthly newsletters about SQL Server, Microsoft BI, Azure, Azure DevOps and Power BI. Currently having 295 opt-in subscribers, a proud achievement.

With over 13 years of experience within the Microsoft BI competence, we've done business with:

# 17 Contact information

Questions or interested in a live demo?
Please feel free to contact:

Clint Huijbers
info@monkeydwh.com