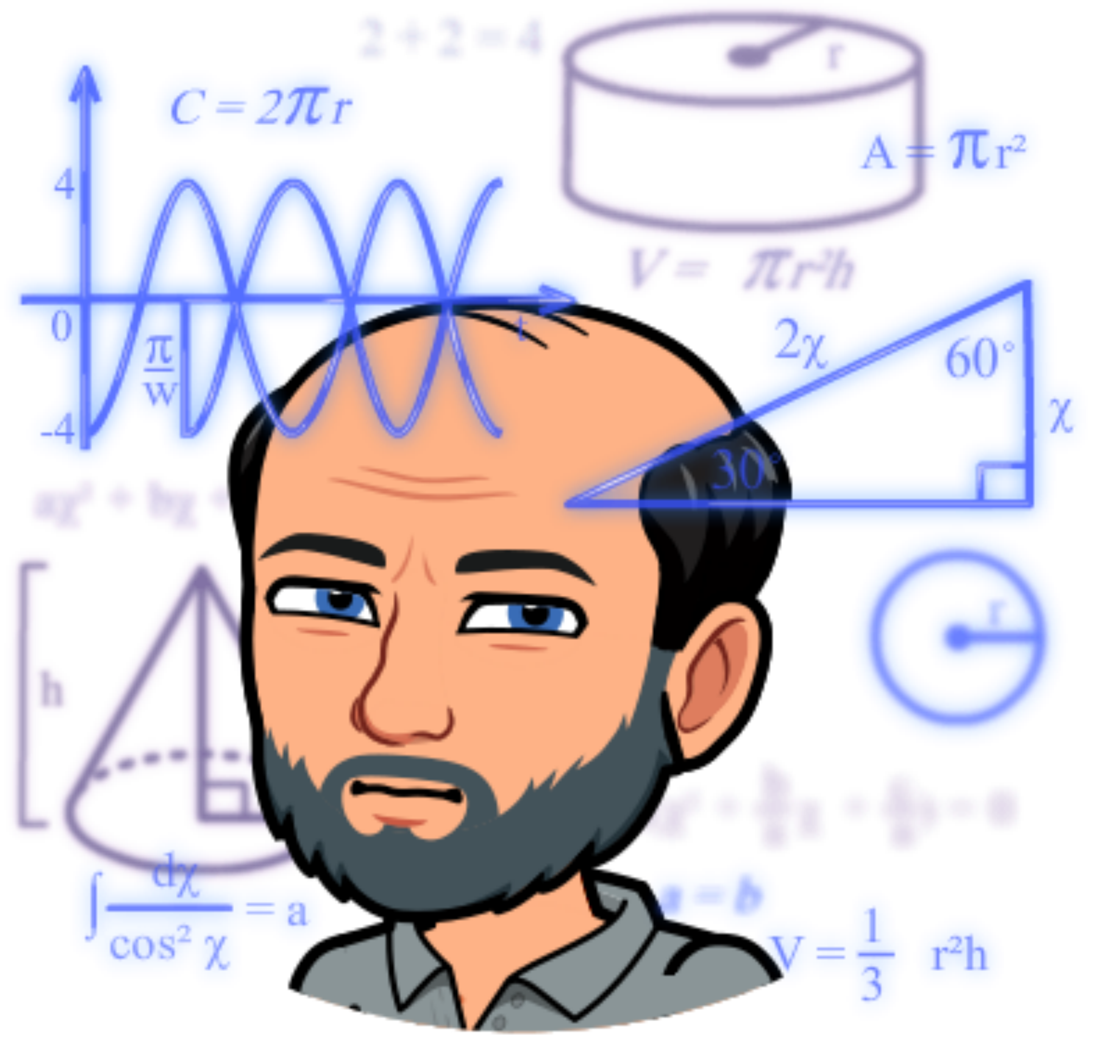


# From SQL to KQL via Azure Data Explorer

Thomas Hütter

PASS Microsoft Data Platform Community
DEUTSCHLAND e.V.

pmOne

SOLARWINDS

smart
casual
datadesign

Mainzer Datenfabrik

b.telligent
smart data. smart decisions.

HMS
analytical software

ORAYLIS
turn your data into value

TIMEXTENDER

Microsoft

ITGAIN

SOLISYON
www.solisyon.de

GDS
BUSINESS INTELLIGENCE
GmbH

# From SQL to KQL via Azure Data Explorer

Thomas Hütter, Diplom-Betriebswirt

- Developer for ERP apps, SQL scripts and BI stuff
- Worked at consultancies, ISVs, end user companies
- 1995: SQL Server, 2001: Nav/BC, 2014: R, 2020: Power*
- Speaker at SQL / data / dev events around Europe

@DerFredo https://twitter.com/DerFredo

de.linkedin.com/in/derfredo

https://techhub.social/@DerFredo

## Agenda

- Where we come from: SQL - Structured Query Language
- Starring: the `SELECT` statement
- Where we are going: KQL - Kusto Query Language
- Helping us along the way: ADX - Azure Data Explorer
- Hands-on examples: KQL in action
- Next level: visualizations, dashboards, time series
- Round-up, resources, learning

## Where we come from: SQL - Structured Query Language

- The Structured Query Language (SQL for short) is a set-based declarative, domain-specific, cross-platform 4GL language that was first released by IBM in 1974.
- Based upon relational algebra and tuple calculus (☞ Edgar F. Codd's relational model of data).
- It is mainly used for managing relational data in an RDBMS.
- Standardized by ISO and ANSI, latest version ISO/IEC 9075:2023 or SQL:2023, but all the vendors roll their own set of additions, extensions, omissions to the standard.
- The concept of NULL values enforces 3-valued logic.
- Whitespace is generally ignored, allowing to format for good readability.

## Where we come from: SQL - Structured Query Language

Lately, SQL has been divided into 4 sublanguages:

- DCL - Data Control Language

  `GRANT, DENY, REVOKE`

- DDL - Data Definition Language

  `CREATE, DROP, ALTER, TRUNCATE`

- DML - Data Manipulating Language

  `INSERT, UPDATE, DELETE`

- DQL - Data Query Language

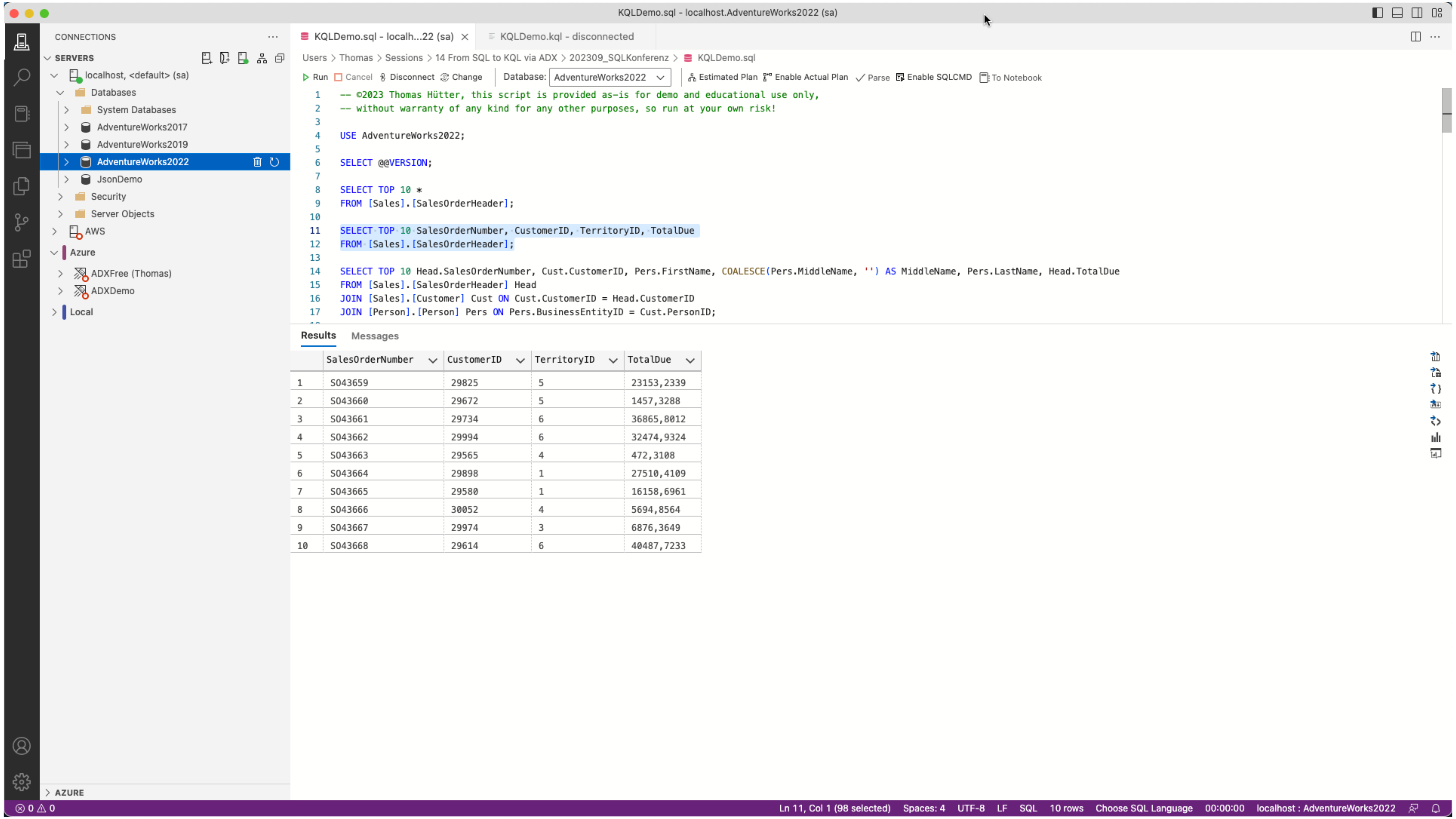
  `SELECT`

## Starring: the `SELECT` statement

The `SELECT` statement and it’s clauses:

- `5. SELECT (8. TOP <n> | 6. DISTINCT) <columns>, <aggregates>`
- `1. FROM <table1>`
  `(<jointype> JOIN <table2> ON <join_condition> …)`
- `2. WHERE <constraint>`
- `3. GROUP BY <column_list>`
- `4. HAVING <constraint_on_group>`
- `7. ORDER BY <column_list> (ASC | DESC)`
- `8. LIMIT <n> / OFFSET <m>`

# Starring: the `SELECT` statement



```sql
-- ©2023 Thomas Hütter, this script is provided as-is for demo and educational use only,
-- without warranty of any kind for any other purposes, so run at your own risk!

USE AdventureWorks2022;

SELECT @@VERSION;

SELECT TOP 10 *
FROM [Sales].[SalesOrderHeader];

SELECT TOP 10 SalesOrderNumber, CustomerID, TerritoryID, TotalDue
FROM [Sales].[SalesOrderHeader];

SELECT TOP 10 Head.SalesOrderNumber, Cust.CustomerID, Pers.FirstName, COALESCE(Pers.MiddleName, '') AS MiddleName, Pers.LastName, Head.TotalDue
FROM [Sales].[SalesOrderHeader] Head
JOIN [Sales].[Customer] Cust ON Cust.CustomerID = Head.CustomerID
JOIN [Person].[Person] Pers ON Pers.BusinessEntityID = Cust.PersonID;
```

| | SalesOrderNumber | CustomerID | TerritoryID | TotalDue |
|---|---|---|---|---|
| 1 | SO43659 | 29825 | 5 | 23153,2339 |
| 2 | SO43660 | 29672 | 5 | 1457,3288 |
| 3 | SO43661 | 29734 | 6 | 36865,8012 |
| 4 | SO43662 | 29994 | 6 | 32474,9324 |
| 5 | SO43663 | 29565 | 4 | 472,3108 |
| 6 | SO43664 | 29898 | 1 | 27510,4109 |
| 7 | SO43665 | 29580 | 1 | 16158,6961 |
| 8 | SO43666 | 30052 | 4 | 5694,8564 |
| 9 | SO43667 | 29974 | 3 | 6876,3649 |
| 10 | SO43668 | 29614 | 6 | 40487,7233 |

## Where we are going: KQL - Kusto Query Language

- Kusto* Query Language (KQL) is
  „… the query language used by Azure Data Explorer, Real-Time Analytics in Microsoft Fabric, Log Analytics in Azure Monitor, Microsoft Sentinel, Microsoft 365 Defender, Azure Resource Graph, and Resource Manager, among other Microsoft products. Querying data using KQL lets you gain insights about your IT, business, and security from large data sets you collect, in near real-time.“
- A Kusto query is
  a read-only request to the query engine to process data, stated in plain-text, which will be evaluated top-down and can consist of one or more statements, separated by a semicolon (;).
- A query statement can be
  a tabular expression, a „let“ statement or a „set“ statement; statements are separated by a pipe (|).
- A management / control command
  is used to retrieve/modify metadata or objects, ingest or export data etc…;
  always starts with a period (.); result may be piped as input to a query.

*: Why „Kusto“? Think of Jacques Cousteau diving the oceans to find hidden treasures 😉

## Where we are going: KQL - Kusto Query Language

- A tabular expression
  is usually composed of a tabular data source (often a table reference or a table literal), tabular data operators such as filters or projections, optional rendering operators; each step connected by a pipe (|), expecting tabular input and providing tabular output (except graphic renderers).

- A rendering statement
  is used to render the result into a graphical output - generate a visualization
  ( usually the last statement in a query ).

- A „let“ statement
  defines a named variable that can hold a scalar or tabular value, a function or a virtual table.

- A „set“ statement
  sets an option value, valid for the scope of the query.

## Helping us along the way: ADX - Azure Data Explorer

- Azure Data Explorer (ADX ) is
  „… a fully managed, high-performance, big data analytics platform that makes it easy to analyze high volumes of data in near real time. The Azure Data Explorer toolbox gives you an end-to-end solution for data ingestion, query, visualization, and management.“

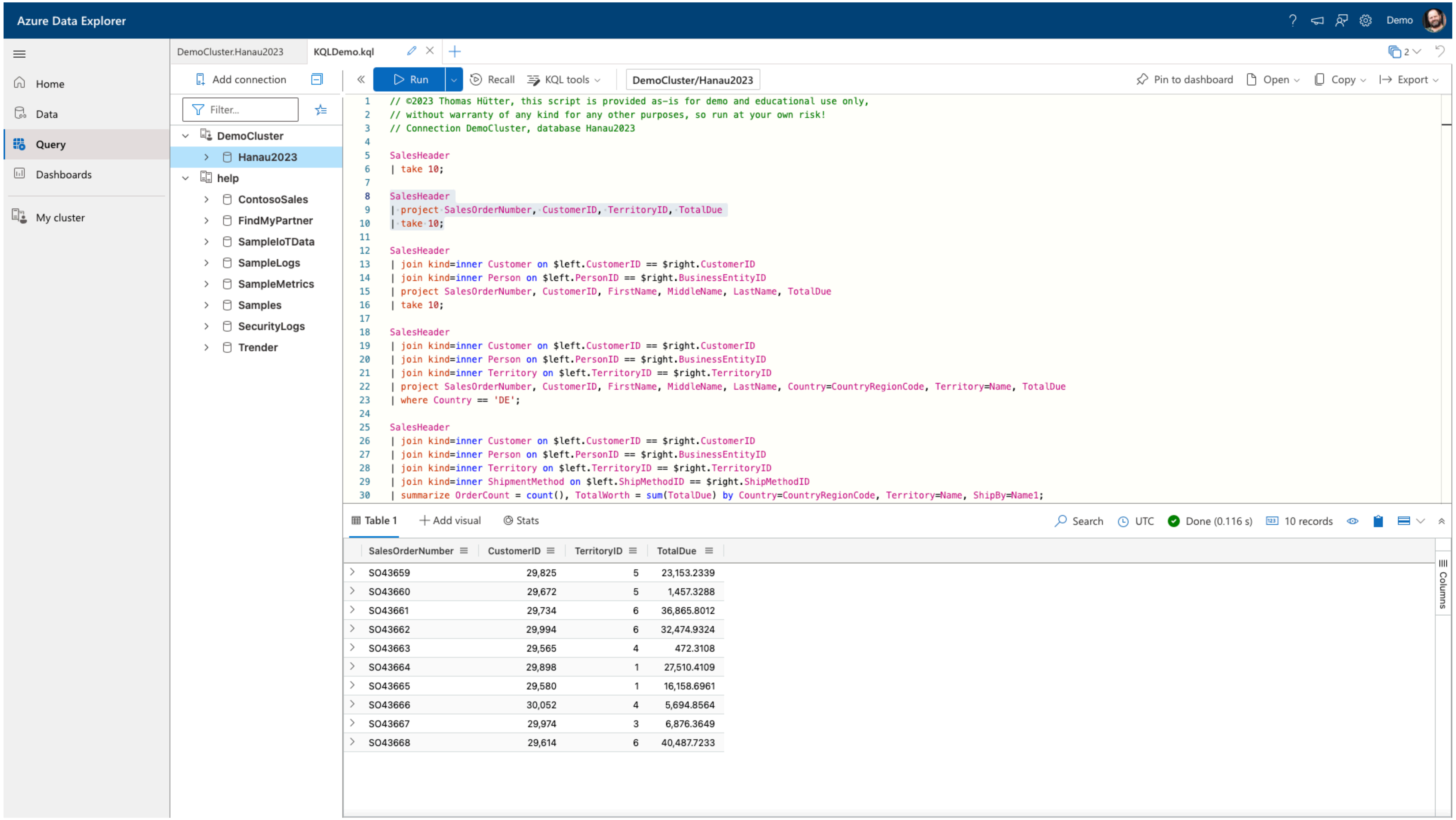
- ADX allows for advanced analytics, including versatile data visualization, using a user-friendly (i.e. easy to read, understand and learn) query language.

- Probably is „The most powerful Azure service you've never heard of“ (Patrick LeBlanc)

- Can be tried without costs by using the ADX „free cluster“ option.

- Alternatives: Kusto.Explorer (Windows app), Kusto extension for Azure Data Studio

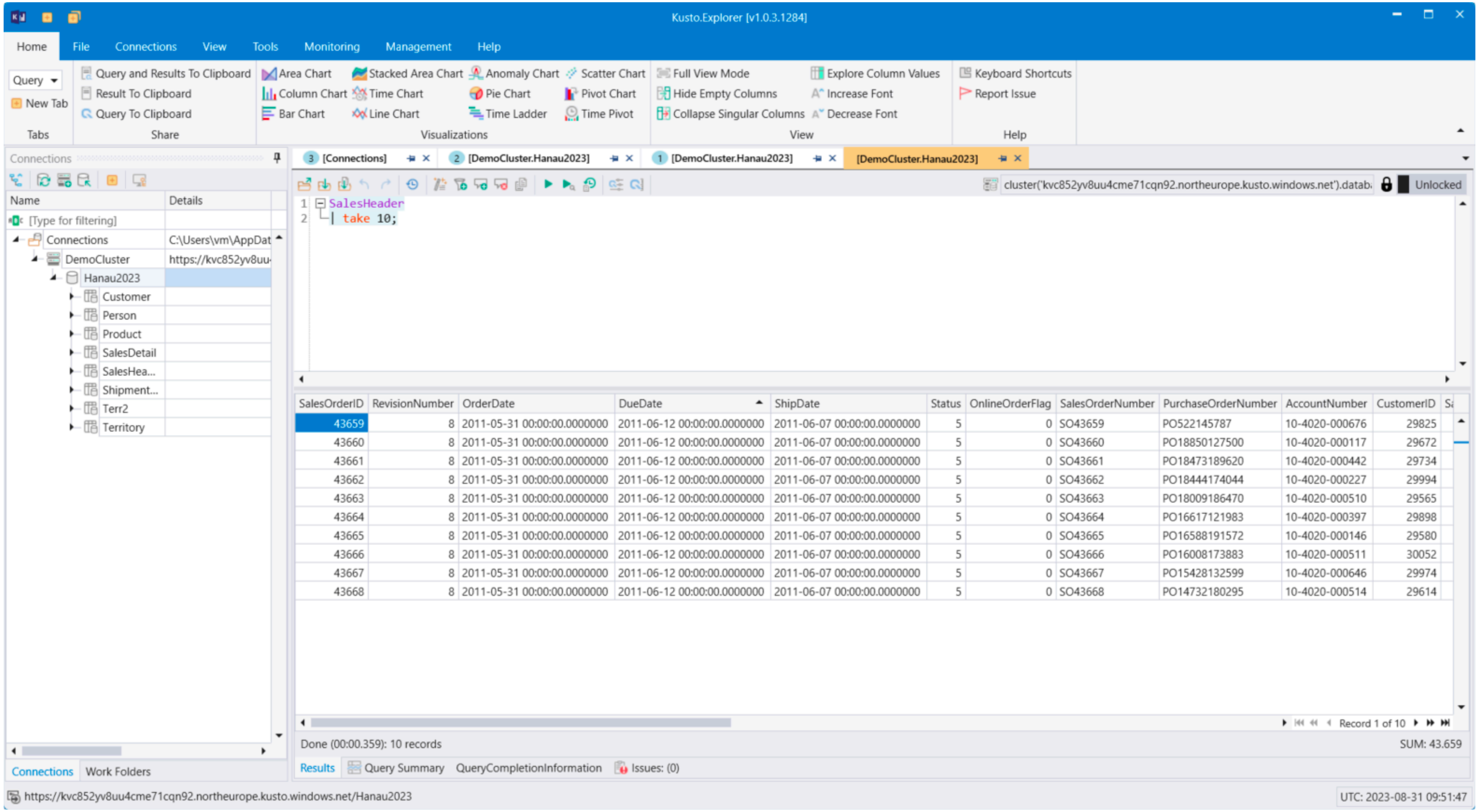
# Helping us along the way: ADX - Azure Data Explorer



```
// ©2023 Thomas Hütter, this script is provided as-is for demo and educational use only,
// without warranty of any kind for any other purposes, so run at your own risk!
// Connection DemoCluster, database Hanau2023

SalesHeader
| take 10;

SalesHeader
| project SalesOrderNumber, CustomerID, TerritoryID, TotalDue
| take 10;

SalesHeader
| join kind=inner Customer on $left.CustomerID == $right.CustomerID
| join kind=inner Person on $left.PersonID == $right.BusinessEntityID
| project SalesOrderNumber, CustomerID, FirstName, MiddleName, LastName, TotalDue
| take 10;

SalesHeader
| join kind=inner Customer on $left.CustomerID == $right.CustomerID
| join kind=inner Person on $left.PersonID == $right.BusinessEntityID
| join kind=inner Territory on $left.TerritoryID == $right.TerritoryID
| project SalesOrderNumber, CustomerID, FirstName, MiddleName, LastName, Country=CountryRegionCode, Territory=Name, TotalDue
| where Country == 'DE';

SalesHeader
| join kind=inner Customer on $left.CustomerID == $right.CustomerID
| join kind=inner Person on $left.PersonID == $right.BusinessEntityID
| join kind=inner Territory on $left.TerritoryID == $right.TerritoryID
| join kind=inner ShipmentMethod on $left.ShipMethodID == $right.ShipMethodID
| summarize OrderCount = count(), TotalWorth = sum(TotalDue) by Country=CountryRegionCode, Territory=Name, ShipBy=Name1;
```

| SalesOrderNumber | CustomerID | TerritoryID | TotalDue |
|---|---|---|---|
| SO43659 | 29,825 | 5 | 23,153.2339 |
| SO43660 | 29,672 | 5 | 1,457.3288 |
| SO43661 | 29,734 | 6 | 36,865.8012 |
| SO43662 | 29,994 | 6 | 32,474.9324 |
| SO43663 | 29,565 | 4 | 472.3108 |
| SO43664 | 29,898 | 1 | 27,510.4109 |
| SO43665 | 29,580 | 1 | 16,158.6961 |
| SO43666 | 30,052 | 4 | 5,694.8564 |
| SO43667 | 29,974 | 3 | 6,876.3649 |
| SO43668 | 29,614 | 6 | 40,487.7233 |

# ADX - Alternative 1: Kusto.Explorer



cluster('kvc852yv8uu4cme71cqn92.northeurope.kusto.windows.net').datab

```
SalesHeader
| take 10;
```

| SalesOrderID | RevisionNumber | OrderDate | DueDate | ShipDate | Status | OnlineOrderFlag | SalesOrderNumber | PurchaseOrderNumber | AccountNumber | CustomerID | Sa |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 43659 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43659 | PO522145787 | 10-4020-000676 | 29825 | |
| 43660 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43660 | PO18850127500 | 10-4020-000117 | 29672 | |
| 43661 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43661 | PO18473189620 | 10-4020-000442 | 29734 | |
| 43662 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43662 | PO18444174044 | 10-4020-000227 | 29994 | |
| 43663 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43663 | PO18009186470 | 10-4020-000510 | 29565 | |
| 43664 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43664 | PO16617121983 | 10-4020-000397 | 29898 | |
| 43665 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43665 | PO16588191572 | 10-4020-000146 | 29580 | |
| 43666 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43666 | PO16008173883 | 10-4020-000511 | 30052 | |
| 43667 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43667 | PO15428132599 | 10-4020-000646 | 29974 | |
| 43668 | 8 | 2011-05-31 00:00:00.0000000 | 2011-06-12 00:00:00.0000000 | 2011-06-07 00:00:00.0000000 | 5 | 0 | SO43668 | PO14732180295 | 10-4020-000514 | 29614 | |

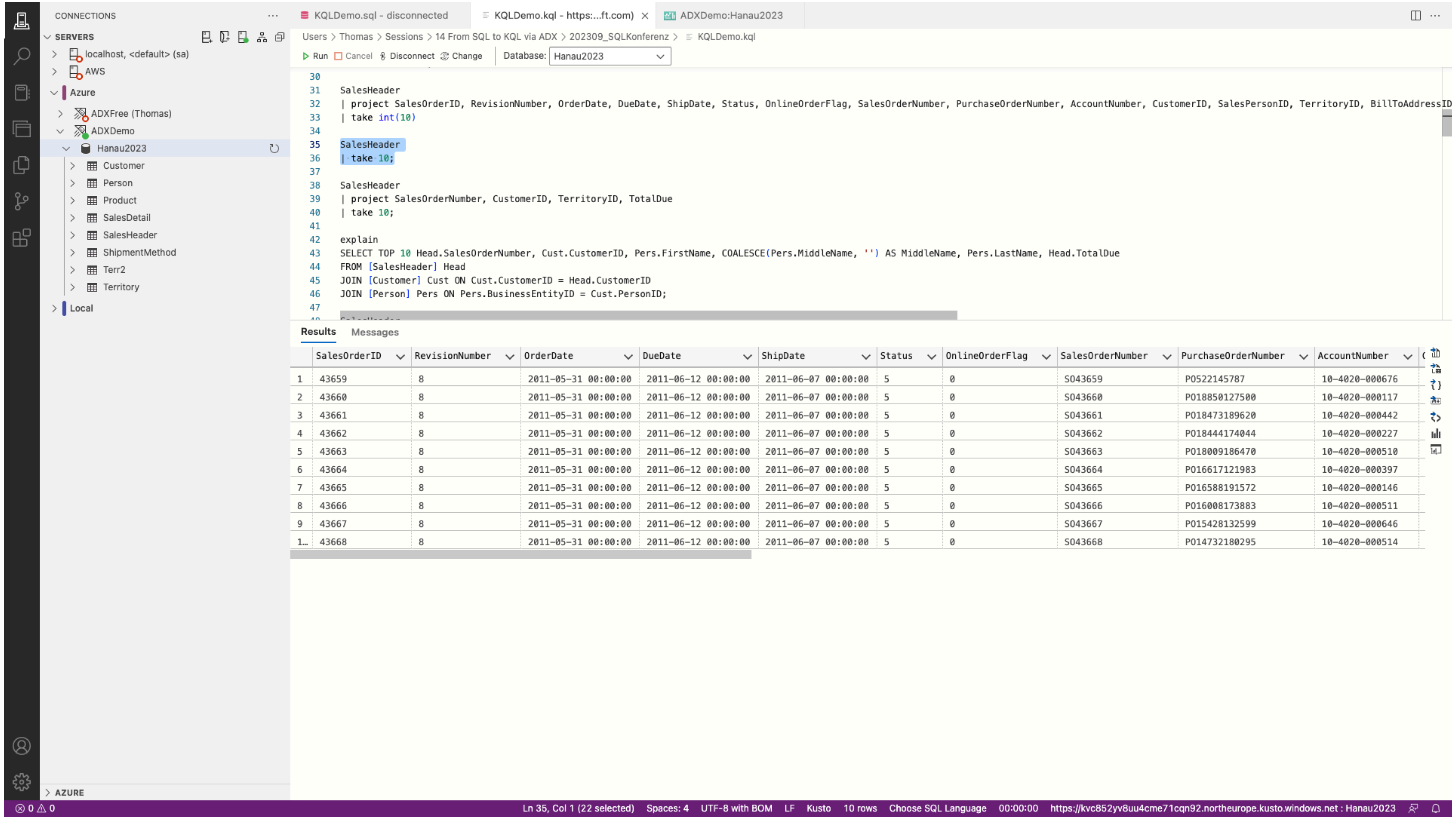
Done (00:00.359): 10 records

SUM: 43.659

https://kvc852yv8uu4cme71cqn92.northeurope.kusto.windows.net/Hanau2023

UTC: 2023-08-31 09:51:47

# ADX - Alternative 2: Kusto extension for Azure Data Studio



CONNECTIONS

SERVERS
- localhost, <default> (sa)
- AWS
- Azure
  - ADXFree (Thomas)
  - ADXDemo
    - Hanau2023
      - Customer
      - Person
      - Product
      - SalesDetail
      - SalesHeader
      - ShipmentMethod
      - Terr2
      - Territory
- Local

AZURE

KQLDemo.sql - disconnected | KQLDemo.kql - https:...ft.com) | ADXDemo:Hanau2023

Users > Thomas > Sessions > 14 From SQL to KQL via ADX > 202309_SQLKonferenz > KQLDemo.kql

Run | Cancel | Disconnect | Change | Database: Hanau2023

```
SalesHeader
| project SalesOrderID, RevisionNumber, OrderDate, DueDate, ShipDate, Status, OnlineOrderFlag, SalesOrderNumber, PurchaseOrderNumber, AccountNumber, CustomerID, SalesPersonID, TerritoryID, BillToAddressID
| take int(10)

SalesHeader
| take 10;

SalesHeader
| project SalesOrderNumber, CustomerID, TerritoryID, TotalDue
| take 10;

explain
SELECT TOP 10 Head.SalesOrderNumber, Cust.CustomerID, Pers.FirstName, COALESCE(Pers.MiddleName, '') AS MiddleName, Pers.LastName, Head.TotalDue
FROM [SalesHeader] Head
JOIN [Customer] Cust ON Cust.CustomerID = Head.CustomerID
JOIN [Person] Pers ON Pers.BusinessEntityID = Cust.PersonID;
```

Results | Messages

| | SalesOrderID | RevisionNumber | OrderDate | DueDate | ShipDate | Status | OnlineOrderFlag | SalesOrderNumber | PurchaseOrderNumber | AccountNumber |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 43659 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43659 | PO522145787 | 10-4020-000676 |
| 2 | 43660 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43660 | PO18850127500 | 10-4020-000117 |
| 3 | 43661 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43661 | PO18473189620 | 10-4020-000442 |
| 4 | 43662 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43662 | PO18444174044 | 10-4020-000227 |
| 5 | 43663 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43663 | PO18009186470 | 10-4020-000510 |
| 6 | 43664 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43664 | PO16617121983 | 10-4020-000397 |
| 7 | 43665 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43665 | PO16588191572 | 10-4020-000146 |
| 8 | 43666 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43666 | PO16008173883 | 10-4020-000511 |
| 9 | 43667 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43667 | PO15428132599 | 10-4020-000646 |
| 1… | 43668 | 8 | 2011-05-31 00:00:00 | 2011-06-12 00:00:00 | 2011-06-07 00:00:00 | 5 | 0 | SO43668 | PO14732180295 | 10-4020-000514 |

⊗ 0 ⚠ 0 Ln 35, Col 1 (22 selected) Spaces: 4 UTF-8 with BOM LF Kusto 10 rows Choose SQL Language 00:00:00 https://kvc852yv8uu4cme71cqn92.northeurope.kusto.windows.net : Hanau2023

## Hands-on examples: KQL in action

- Understands a limited subset of T-SQL
- „Explain“ function translates SQL to KQL ( also limited 😕 )
- Observe special behavior:

  - default sort order: descending
  - default equality check: case-sensitive ( case-insensitive with =~ )
  - `union` allows for differing table schemas
  - `search` works on *all* columns in a table ( default case-*insensitive* )
  - `ago` allows for dynamic time intervals
  - …

# Next level: visualizations, dashboards, time series

- Visualizations? Yes, easy, just add a `render` clause at the end of your query 😉.
- Several types of visualization possible, including maps ( geospatial clustering ).
- Then „Pin to dashboard“ does exactly that.
- Dashboards can be filtered, even dynamically.
- Dashboards can be exported and shared with others in your org ( tenant ).
- Built-in time series analysis capabilities
- Graphical analysis, anomaly detection, decomposition, forecasting

## Resources:

- KQL overview: https://learn.microsoft.com/en-us/azure/data-explorer/kusto/query/
- SQL to Kusto cheat sheet: https://learn.microsoft.com/en-us/azure/data-explorer/kusto/query/sqlcheatsheet
- Azure Data Explorer: https://learn.microsoft.com/en-us/azure/data-explorer/data-explorer-overview
- ADX free cluster: https://dataexplorer.azure.com/freecluster
- Visualization overview: https://learn.microsoft.com/en-us/azure/data-explorer/viz-overview, dashboards: https://learn.microsoft.com/en-us/azure/data-explorer/azure-data-explorer-dashboards
- Microsoft learning paths: https://learn.microsoft.com/en-us/training/paths/kusto-query-language/, https://learn.microsoft.com/en-us/training/paths/data-analysis-data-explorer-kusto-query-language/, https://learn.microsoft.com/en-us/training/paths/analyze-monitoring-data-with-kql/
- ADX-in-a-day: https://github.com/Azure/ADX-in-a-Day

# From SQL to KQL via Azure Data Explorer

- 𝕏 @DerFredo https://twitter.com/DerFredo
- de.linkedin.com/in/derfredo
- https://techhub.social/@DerFredo

This file and the demo scripts can be found at:
https://bit.ly/DerFredoSQLK23

Feedback?

Yes, please!

https://sqlkonferenzeval.azurewebsites.net/34047284969258

Thank you very much for your attention.

Vielen Dank für Eure Aufmerksamkeit.