05/10/2022

Datamine

1. Introduction

Datamine was born in the 2000s with the need to easily and quickly consult informations related to the structure of a SQL Server database and to be able to easily query the data of a table and linked tables.

Datamine was set up gradually, according to the needs expressed by questions of development and administration. The writing of Datamine being done on personal time was rather carried out in spurts, more than by large phases of implementation.

Datamine is of course not intended to replace Microsoft SQL Server Management Studio

The first versions of Datamine were written for SQL Server 2000. Today the minimum supported version is SQL Server 2008R2 and Datamine has been tested with all versions up to SQL Server 2019

2. Purposes

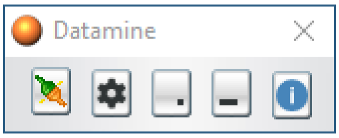
There are several purposes for writing Datamine:

- Display a more complete database tree than in SSMS, for example the following information is provided directly in the tree:
    - Number of rows in tables
    - Number of columns for tables, views, etc.
    - Rows size
    - Dependencies for each object
- Interface based on free windows
- Simplification of table data consultation: a table can be opened in a QBE window, filters can be added, data windows can be opened for linked tables with automatic refresh according to the line displayed in the first window
- Simplify the search in the structure information: it is for example possible to search for a column name to know in which tables, views, triggers, stored procedures, etc. is used this column
- Automate the search for anomalies by running a set of controls to find out what is wrong or should alert on a database (examples: unverified elements, indexes covered by other indexes)

- Allow easy consultation of all database objects with all their characteristics (examples: list of tables with the space occupied, list of indexes with their fragmentation rates, list of stored procedures with their source code, etc.)

3. Main window and icon bar

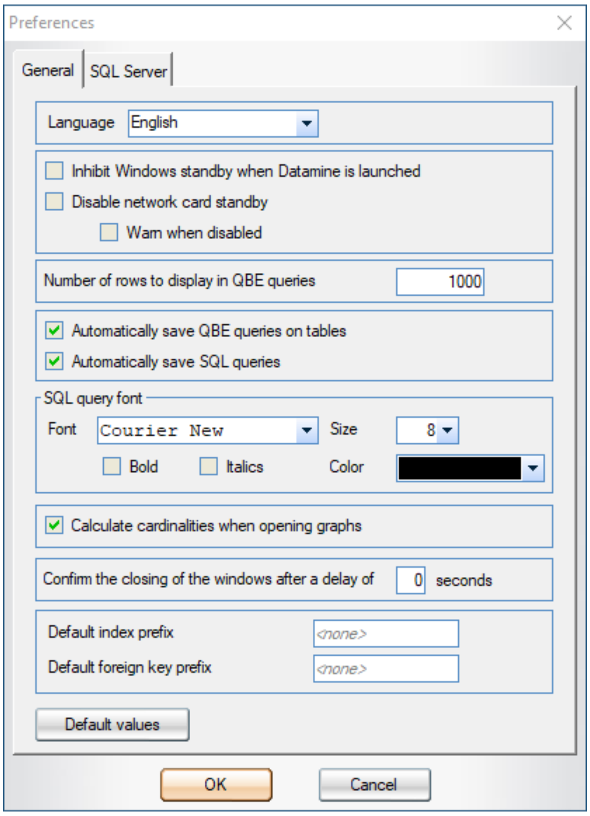
Datamine can be launched directly or with the option ('/Systray') which iconizes the application directly in the notification area near the time



Here is the meaning of the different buttons:

🔌	Allows you to manage database connections Allows you to connect
⚙	Managing Datamine settings
.	Iconizes the main window in the notification area A left click opens the window A right click opens a context menu
▁	Iconizes all open Datamine windows in the taskbar
i	Shows version, copyright and release notes

4. Parameters



Language: English

☐ Inhibit Windows standby when Datamine is launched

☐ Disable network card standby

☐ Warn when disabled

Number of rows to display in QBE queries: 1000

☑ Automatically save QBE queries on tables

☑ Automatically save SQL queries

SQL query font

Font: Courier New Size: 8

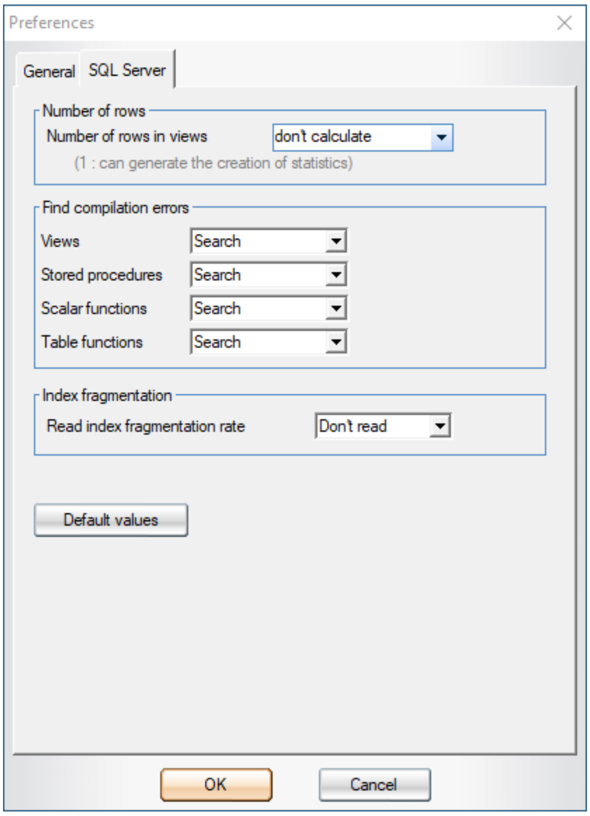
☐ Bold ☐ Italics Color: ■

☑ Calculate cardinalities when opening graphs

Confirm the closing of the windows after a delay of 0 seconds

Default index prefix: <none>

Default foreign key prefix: <none>



Number of rows

Number of rows in views: dont calculate

(1 : can generate the creation of statistics)

Find compilation errors

Views: Search

Stored procedures: Search

Scalar functions: Search

Table functions: Search

Index fragmentation

Read index fragmentation rate: Don't read

The language selection allows you to choose between French and English

Disabling the standby of network cards will only be effective if the user is an administrator of his machine

Calculating the number of rows in views can generate statistics due to the execution of a 'select count(*)' and is therefore not necessarily desired

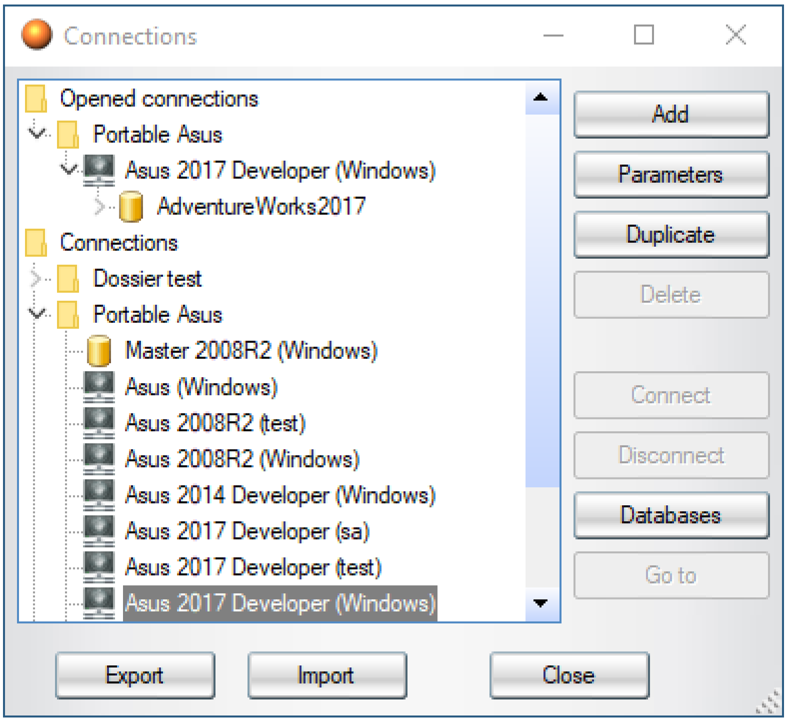
The number of rows in the tables is always calculated if the user has the 'view database state' right and uses sys.dm_db_partition_stats and there is no side effect

The search for compilation errors can significantly increase the loading of the structure of a database depending on the number of elements to be checked and it may be interesting to deactivate these checks

Reading the index fragmentation rate is not enabled by default, i.e. this calculation is not made when loading the database structure; but the fragmentation of the indexes can be read later, once connected to the database.

5. Connecting to a database

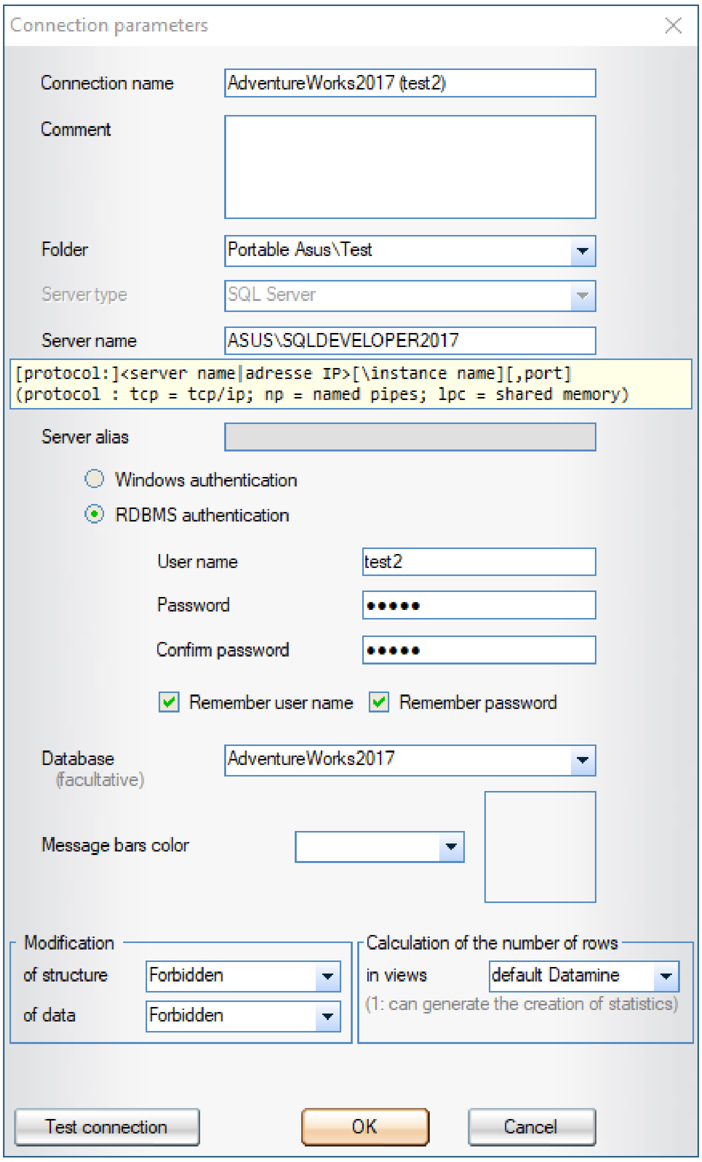
Here is an example of the contents of the connection management window



The 'Open connections' section displays the connections already open and allows you to switch to the window containing the database tree structure. If Datamine is iconized in the notification area, the contextual menu also allows you to switch directly to an open connection

The 'Connections' section displays all the configured connections. These can be sorted into folders. A connection is defined either at the level of a SQL Server or at the level of a database of a server. If it is a server, you will have to choose the database to reach when opening the connection

Here is the connection settings window:



Connection parameters

Connection name: AdventureWorks2017 (test2)

Comment:

Folder: Portable Asus\Test

Server type: SQL Server

Server name: ASUS\SQLDEVELOPER2017

```
[protocol:]<server name|adresse IP>[\instance name][,port]
(protocol : tcp = tcp/ip; np = named pipes; lpc = shared memory)
```

Server alias:

○ Windows authentication

● RDBMS authentication

User name: test2

Password: •••••

Confirm password: •••••

☑ Remember user name ☑ Remember password

Database (facultative): AdventureWorks2017

Message bars color:

Modification

of structure: Forbidden

of data: Forbidden

Calculation of the number of rows

in views: default Datamine

(1: can generate the creation of statistics)

The ‘Connection name’ is the name displayed in the connection management window. It also appears in all open windows related to this connection

The ‘Folder’ allows you to indicate in which folder to classify the connection. Each sub-folder level must be separated by a ‘\’

The ‘Server type’ must be SQL Server. Datamine was designed to be multi-DBMS, but adding a new system is not on the agenda

The ‘Server Name’ indicates which server to reach. The possible syntaxes are indicated in the help area

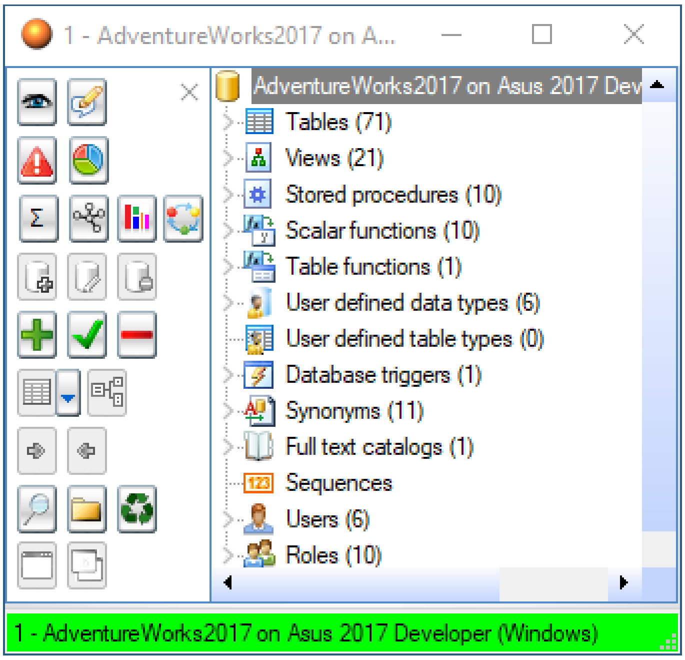
The ‘Server alias’ field is read-only. It contains the value returned by @@ServerName and is filled in if the value is different from ‘Server Name’

The name of the database is optional, if it is not filled in, it is a connection setting to a server that is made and the database to be reached must be chosen when opening a connection

The message bar color is optional; if it is specified, all the message bars of the windows linked to the connection will have this color. A classic use is to use the colors according to the type of environment (red = Production, orange = Recipe, green = Development)

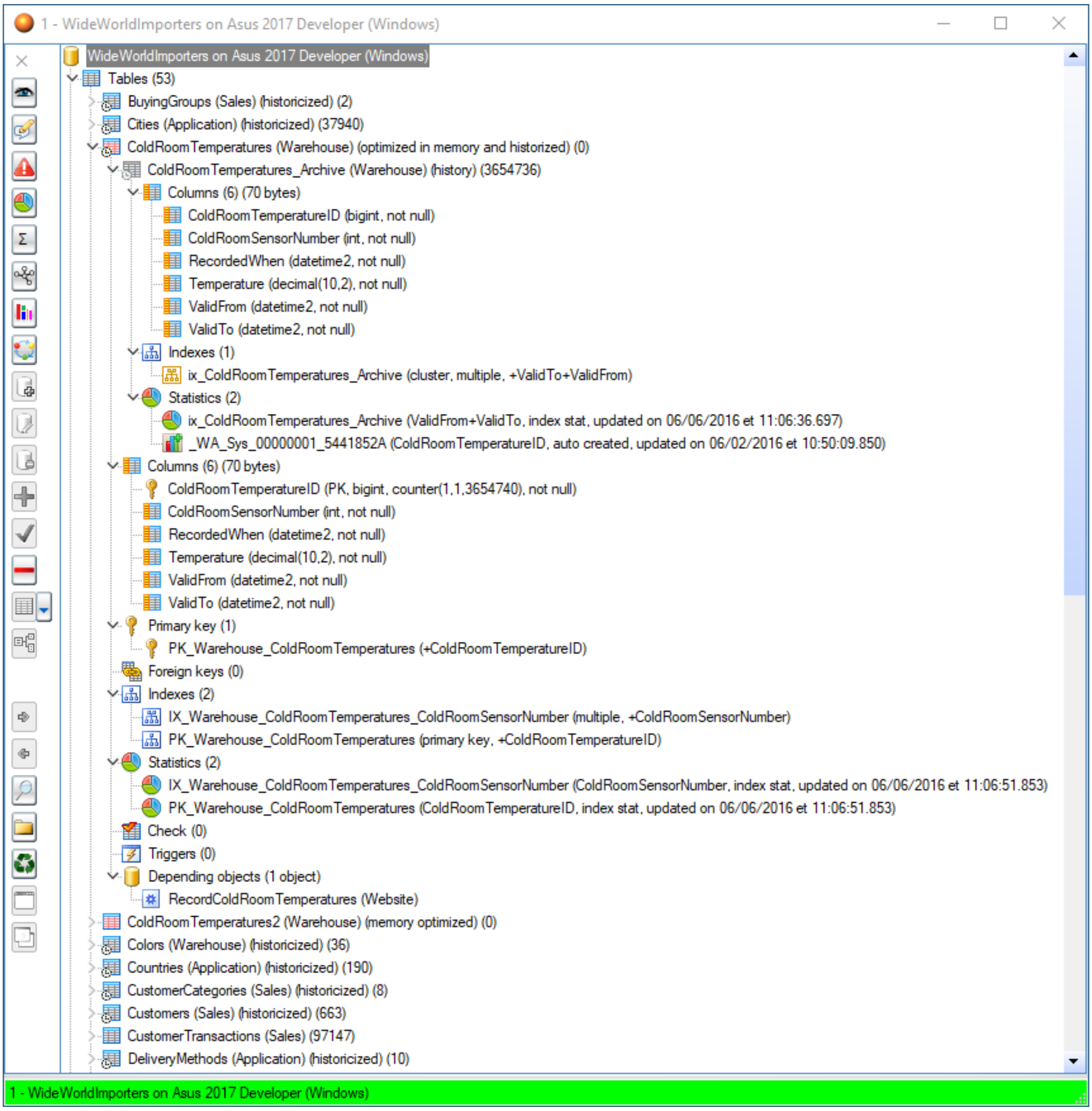
6. Main database window

Here is an example of the main database window


1 - AdventureWorks2017 on A...
AdventureWorks2017 on Asus 2017 Dev
Tables (71)
Views (21)
Stored procedures (10)
Scalar functions (10)
Table functions (1)
User defined data types (6)
User defined table types (0)
Database triggers (1)
Synonyms (11)
Full text catalogs (1)
Sequences
Users (6)
Roles (10)
1 - AdventureWorks2017 on Asus 2017 Developer (Windows)

On the left are all the usable commands and on the right the tree structure. The commands panel can be hidden by clicking on the closing cross at the top right of the panel

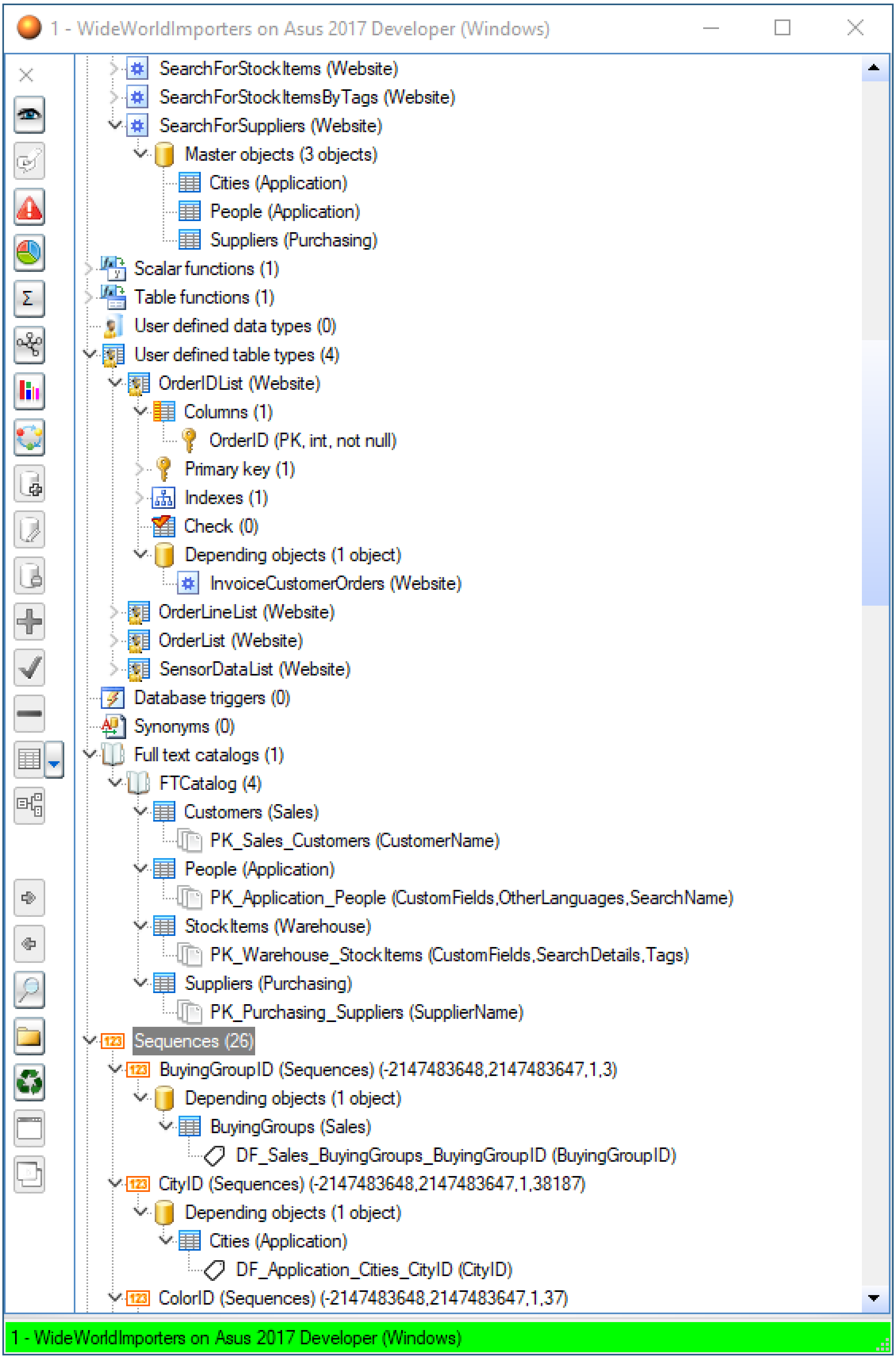
Here are several views of a database tree



1 - WideWorldImporters on Asus 2017 Developer (Windows)

```
WideWorldImporters on Asus 2017 Developer (Windows)
  Tables (53)
    BuyingGroups (Sales) (historicized) (2)
    Cities (Application) (historicized) (37940)
    ColdRoomTemperatures (Warehouse) (optimized in memory and historized) (0)
      ColdRoomTemperatures_Archive (Warehouse) (history) (3654736)
        Columns (6) (70 bytes)
          ColdRoomTemperatureID (bigint, not null)
          ColdRoomSensorNumber (int, not null)
          RecordedWhen (datetime2, not null)
          Temperature (decimal(10,2), not null)
          ValidFrom (datetime2, not null)
          ValidTo (datetime2, not null)
        Indexes (1)
          ix_ColdRoomTemperatures_Archive (cluster, multiple, +ValidTo+ValidFrom)
        Statistics (2)
          ix_ColdRoomTemperatures_Archive (ValidFrom+ValidTo, index stat, updated on 06/06/2016 et 11:06:36.697)
          _WA_Sys_00000001_5441852A (ColdRoomTemperatureID, auto created, updated on 06/02/2016 et 10:50:09.850)
      Columns (6) (70 bytes)
        ColdRoomTemperatureID (PK, bigint, counter(1,1,3654740), not null)
        ColdRoomSensorNumber (int, not null)
        RecordedWhen (datetime2, not null)
        Temperature (decimal(10,2), not null)
        ValidFrom (datetime2, not null)
        ValidTo (datetime2, not null)
      Primary key (1)
        PK_Warehouse_ColdRoomTemperatures (+ColdRoomTemperatureID)
      Foreign keys (0)
      Indexes (2)
        IX_Warehouse_ColdRoomTemperatures_ColdRoomSensorNumber (multiple, +ColdRoomSensorNumber)
        PK_Warehouse_ColdRoomTemperatures (primary key, +ColdRoomTemperatureID)
      Statistics (2)
        IX_Warehouse_ColdRoomTemperatures_ColdRoomSensorNumber (ColdRoomSensorNumber, index stat, updated on 06/06/2016 et 11:06:51.853)
        PK_Warehouse_ColdRoomTemperatures (ColdRoomTemperatureID, index stat, updated on 06/06/2016 et 11:06:51.853)
      Check (0)
      Triggers (0)
      Depending objects (1 object)
        RecordColdRoomTemperatures (Website)
    ColdRoomTemperatures2 (Warehouse) (memory optimized) (0)
    Colors (Warehouse) (historicized) (36)
    Countries (Application) (historicized) (190)
    CustomerCategories (Sales) (historicized) (8)
    Customers (Sales) (historicized) (663)
    CustomerTransactions (Sales) (97147)
    DeliveryMethods (Application) (historicized) (10)
```

1 - WideWorldImporters on Asus 2017 Developer (Windows)



1 - WideWorldImporters on Asus 2017 Developer (Windows)

```
    SearchForStockItems (Website)
    SearchForStockItemsByTags (Website)
    SearchForSuppliers (Website)
      Master objects (3 objects)
        Cities (Application)
        People (Application)
        Suppliers (Purchasing)
  Scalar functions (1)
  Table functions (1)
  User defined data types (0)
  User defined table types (4)
    OrderIDList (Website)
      Columns (1)
        OrderID (PK, int, not null)
      Primary key (1)
      Indexes (1)
      Check (0)
      Depending objects (1 object)
        InvoiceCustomerOrders (Website)
    OrderLineList (Website)
    OrderList (Website)
    SensorDataList (Website)
  Database triggers (0)
  Synonyms (0)
  Full text catalogs (1)
    FTCatalog (4)
      Customers (Sales)
        PK_Sales_Customers (CustomerName)
      People (Application)
        PK_Application_People (CustomFields,OtherLanguages,SearchName)
      StockItems (Warehouse)
        PK_Warehouse_StockItems (CustomFields,SearchDetails,Tags)
      Suppliers (Purchasing)
        PK_Purchasing_Suppliers (SupplierName)
  Sequences (26)
    BuyingGroupID (Sequences) (-2147483648,2147483647,1,3)
      Depending objects (1 object)
        BuyingGroups (Sales)
          DF_Sales_BuyingGroups_BuyingGroupID (BuyingGroupID)
    CityID (Sequences) (-2147483648,2147483647,1,38187)
      Depending objects (1 object)
        Cities (Application)
          DF_Application_Cities_CityID (CityID)
    ColorID (Sequences) (-2147483648,2147483647,1,37)
```

1 - WideWorldImporters on Asus 2017 Developer (Windows)

7. Commands of the main window of a database

👁	Database properties Used to display all the elements of the database structure with all their properties in the form of tables. See the specific paragraph below
✎	Allow to enter a description for the selected element in the database
⚠	Runs database controls. See the specific paragraph below
◔	Allow to delete statistics created automatically by SQL Server
Σ	Harmonizes the ordering of the columns so that they use that of the database
	Recompiles triggers, views, stored procedures, scalar functions and table functions then reloads all dependencies between database objects
📊	Retrieves fragmentation rate and number of pages for each index
⟳	Runs a check of all sources in the database
	Adds an object based on the selected node in the tree This function is currently only available for foreign keys
✎	Modifies an object according to the node selected in the tree structure This function is currently only available for foreign keys and indexes The taking into account of the indexes being only partial, the recording of the modifications is not possible
	Deletes an object based on the selected node in the tree This function is currently only available for tables, views, foreign keys, indexes and statistics
➕	Enables and controls disabled objects
✓	Controls not trusted objects
➖	Disables active objects
▦ ▾	Opens a QBE window on the selected table or view The arrow allows you to load a saved QBE query See the specific paragraph below
	Opens the relationship graph from the selected table See the specific paragraph below
→	Lets browse dependencies
←	Allows you to go back in the path of dependencies
🔍	Search function for an element in the database structure See the specific paragraph below
📁	Tree view by schema Return to standard display
♻	Reread the database structure
🗔	Displays the list of windows open and attached to the current connection Allows to select a specific window
	Bring windows of the current connection to the foreground

8. Database properties

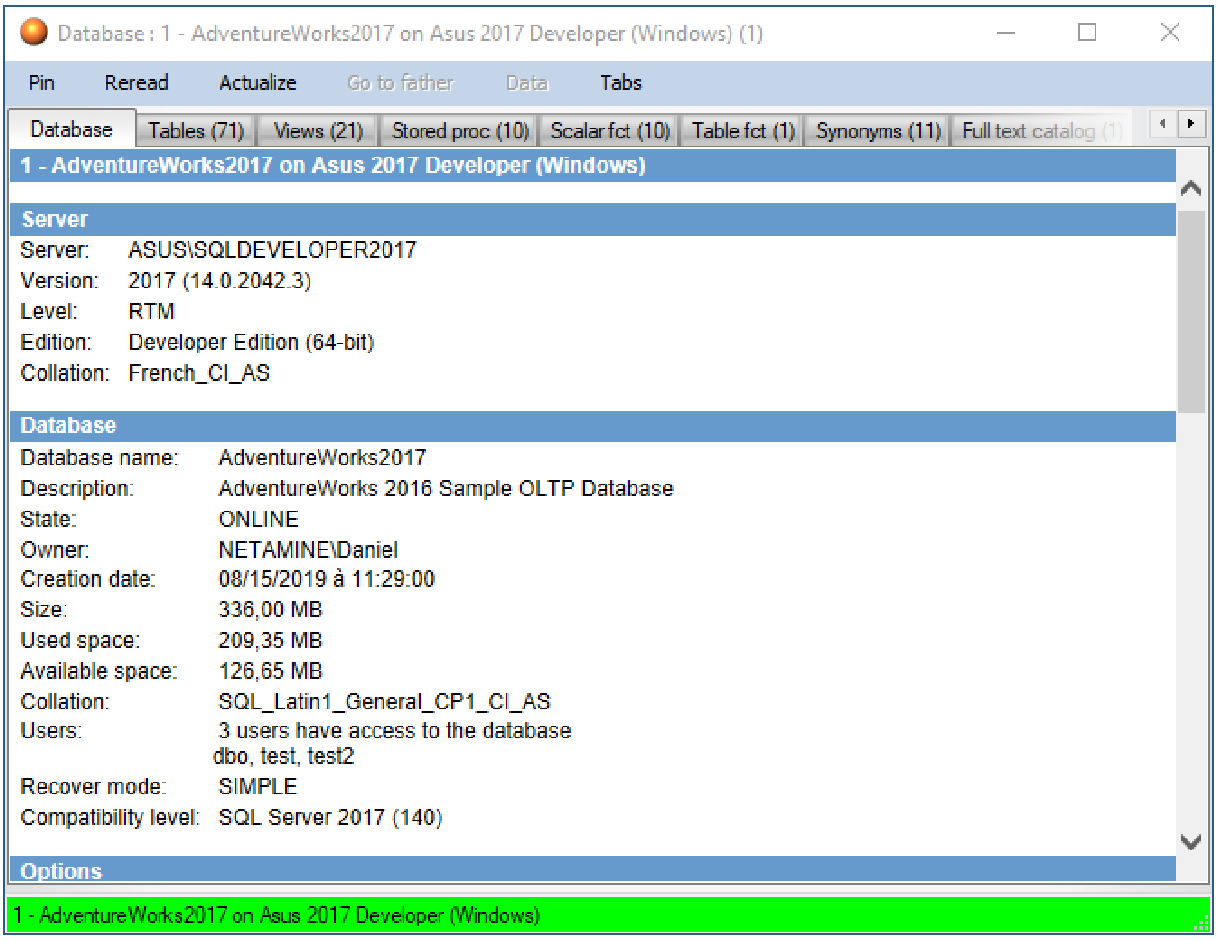
The database properties window allows you to display all the objects of the database structure with all their properties in the form of tables

There is a first tab with general database information (server version, database size, options, etc.) then a tab for each type of object with sub-tables when relevant (for example in the Tables tab, there are, among other things, Columns and Foreign Keys sub-tables)

Many functions are available in the displayed tables; for example, it is possible to search by clicking on the magnifying glass in the title of the columns, to make a filter by right-clicking on the same magnifying glass, to select the columns displayed by clicking on the icon above the elevator, to export the data in Excel, etc.

A frequent use is to position yourself on the Columns tab, which displays all the columns of the database and to use the search or filter functions to find out where a column is used

Here are some screenshots of database properties:



1 - AdventureWorks2017 on Asus 2017 Developer (Windows)

Server

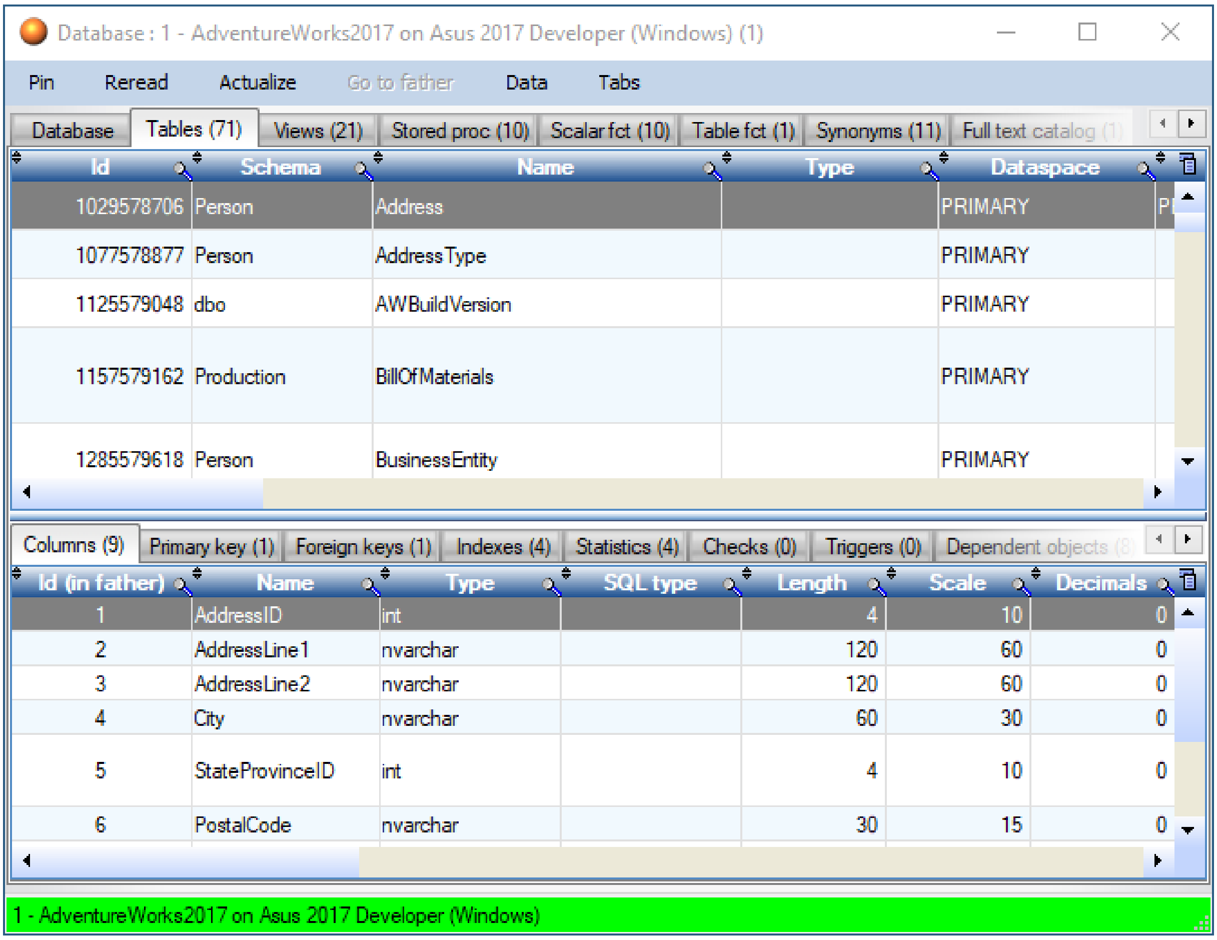
Server: ASUS\SQLDEVELOPER2017
Version: 2017 (14.0.2042.3)
Level: RTM
Edition: Developer Edition (64-bit)
Collation: French_CI_AS

Database

Database name: AdventureWorks2017
Description: AdventureWorks 2016 Sample OLTP Database
State: ONLINE
Owner: NETAMINE\Daniel
Creation date: 08/15/2019 à 11:29:00
Size: 336,00 MB
Used space: 209,35 MB
Available space: 126,65 MB
Collation: SQL_Latin1_General_CP1_CI_AS
Users: 3 users have access to the database
dbo, test, test2
Recover mode: SIMPLE
Compatibility level: SQL Server 2017 (140)

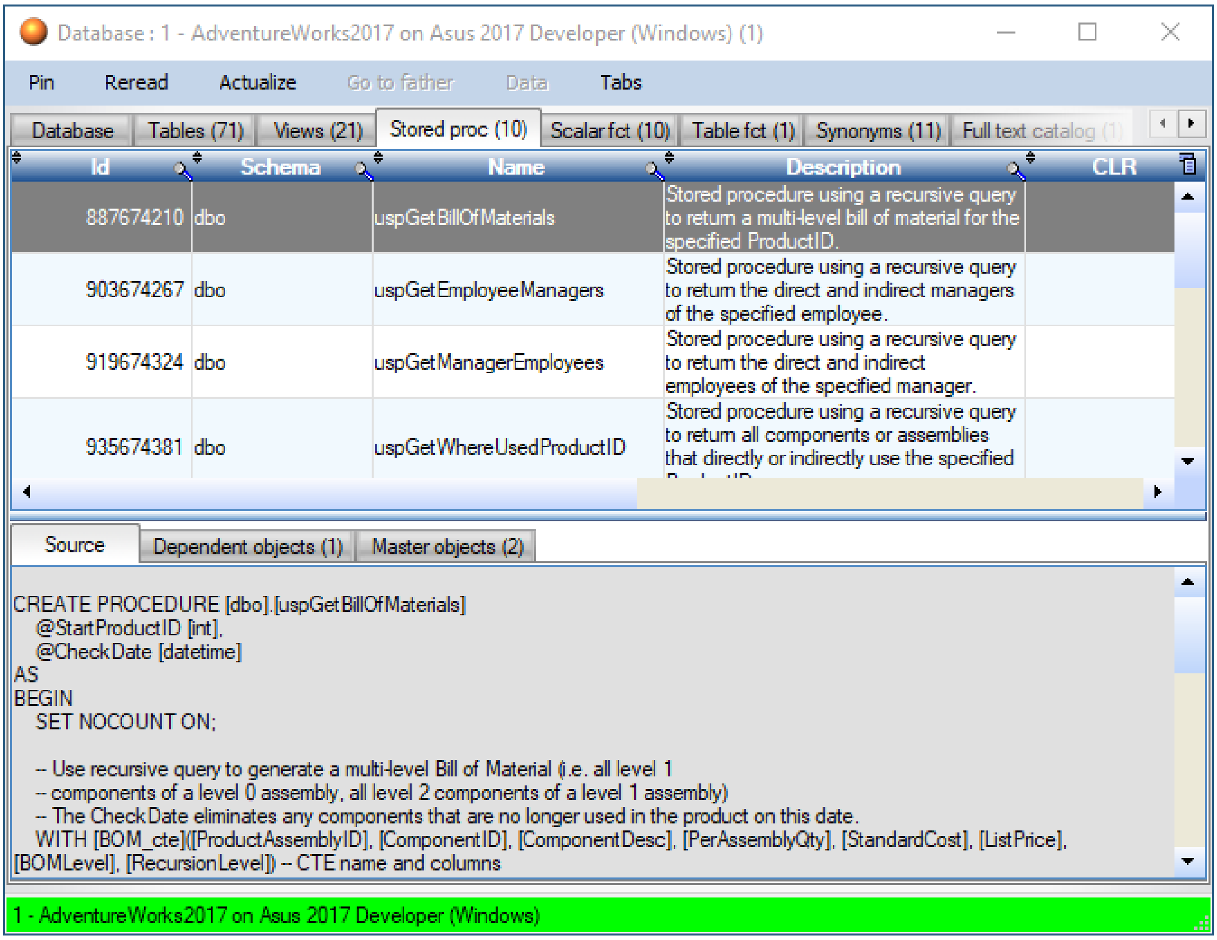
Options

1 - AdventureWorks2017 on Asus 2017 Developer (Windows)



Id	Schema	Name	Type	Dataspace	
1029578706	Person	Address		PRIMARY	P
1077578877	Person	AddressType		PRIMARY	
1125579048	dbo	AWBuildVersion		PRIMARY	
1157579162	Production	BillOfMaterials		PRIMARY	
1285579618	Person	BusinessEntity		PRIMARY	

Id (in father)	Name	Type	SQL type	Length	Scale	Decimals
1	AddressID	int		4	10	0
2	AddressLine1	nvarchar		120	60	0
3	AddressLine2	nvarchar		120	60	0
4	City	nvarchar		60	30	0
5	StateProvinceID	int		4	10	0
6	PostalCode	nvarchar		30	15	0



Id	Schema	Name	Description	CLR
887674210	dbo	uspGetBillOfMaterials	Stored procedure using a recursive query to return a multi-level bill of material for the specified ProductID.	
903674267	dbo	uspGetEmployeeManagers	Stored procedure using a recursive query to return the direct and indirect managers of the specified employee.	
919674324	dbo	uspGetManagerEmployees	Stored procedure using a recursive query to return the direct and indirect employees of the specified manager.	
935674381	dbo	uspGetWhereUsedProductID	Stored procedure using a recursive query to return all components or assemblies that directly or indirectly use the specified	

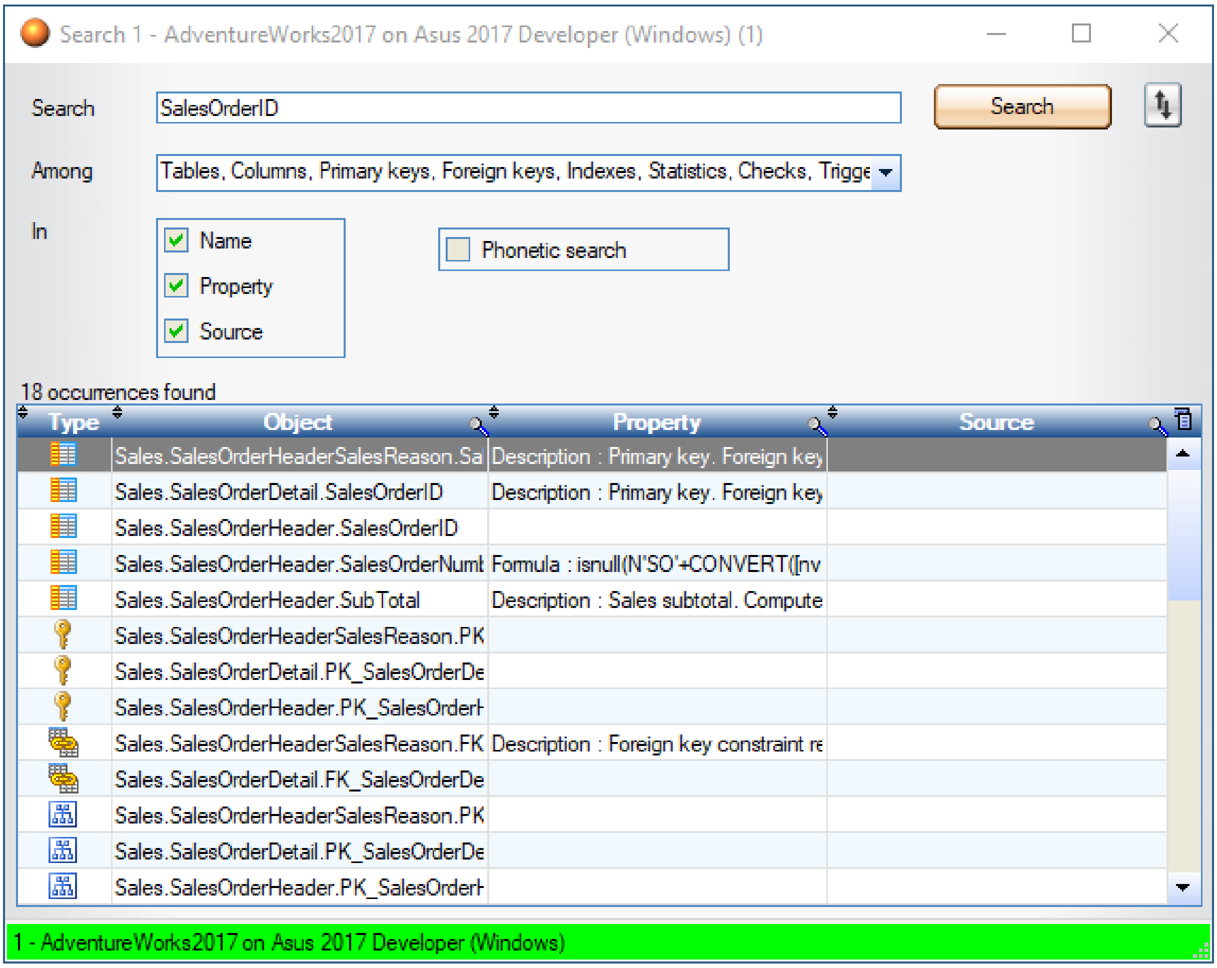
```
CREATE PROCEDURE [dbo].[uspGetBillOfMaterials]
    @StartProductID [int],
    @CheckDate [datetime]
AS
BEGIN
    SET NOCOUNT ON;

    -- Use recursive query to generate a multi-level Bill of Material (i.e. all level 1
    -- components of a level 0 assembly, all level 2 components of a level 1 assembly)
    -- The CheckDate eliminates any components that are no longer used in the product on this date.
    WITH [BOM_cte]([ProductAssemblyID], [ComponentID], [ComponentDesc], [PerAssemblyQty], [StandardCost], [ListPrice],
[BOMLevel], [RecursionLevel]) -- CTE name and columns
```

9. Search

The search function allows you to search for a term in one or more elements of the database, in the name, the properties (description, formula of a calculated column, etc.) and the source code

A double click on a line go to the element in the tree structure

Here is an example search:



Search 1 - AdventureWorks2017 on Asus 2017 Developer (Windows) (1)
Search

Search: SalesOrderID

Among: Tables, Columns, Primary keys, Foreign keys, Indexes, Statistics, Checks, Trigge

In: ☑ Name ☑ Property ☑ Source

□ Phonetic search

18 occurrences found

Type	Object	Property	Source
▦	Sales.SalesOrderHeaderSalesReason.Sa	Description : Primary key. Foreign key	
▦	Sales.SalesOrderDetail.SalesOrderID	Description : Primary key. Foreign key	
▦	Sales.SalesOrderHeader.SalesOrderID		
▦	Sales.SalesOrderHeader.SalesOrderNumt	Formula : isnull(N'SO'+CONVERT([nv	
▦	Sales.SalesOrderHeader.SubTotal	Description : Sales subtotal. Compute	
🔑	Sales.SalesOrderHeaderSalesReason.PK		
🔑	Sales.SalesOrderDetail.PK_SalesOrderDe		
🔑	Sales.SalesOrderHeader.PK_SalesOrderH		
🔗	Sales.SalesOrderHeaderSalesReason.FK	Description : Foreign key constraint re	
🔗	Sales.SalesOrderDetail.FK_SalesOrderDe		
⊞	Sales.SalesOrderHeaderSalesReason.PK		
⊞	Sales.SalesOrderDetail.PK_SalesOrderDe		
⊞	Sales.SalesOrderHeader.PK_SalesOrderH		

1 - AdventureWorks2017 on Asus 2017 Developer (Windows)

10. Graphs

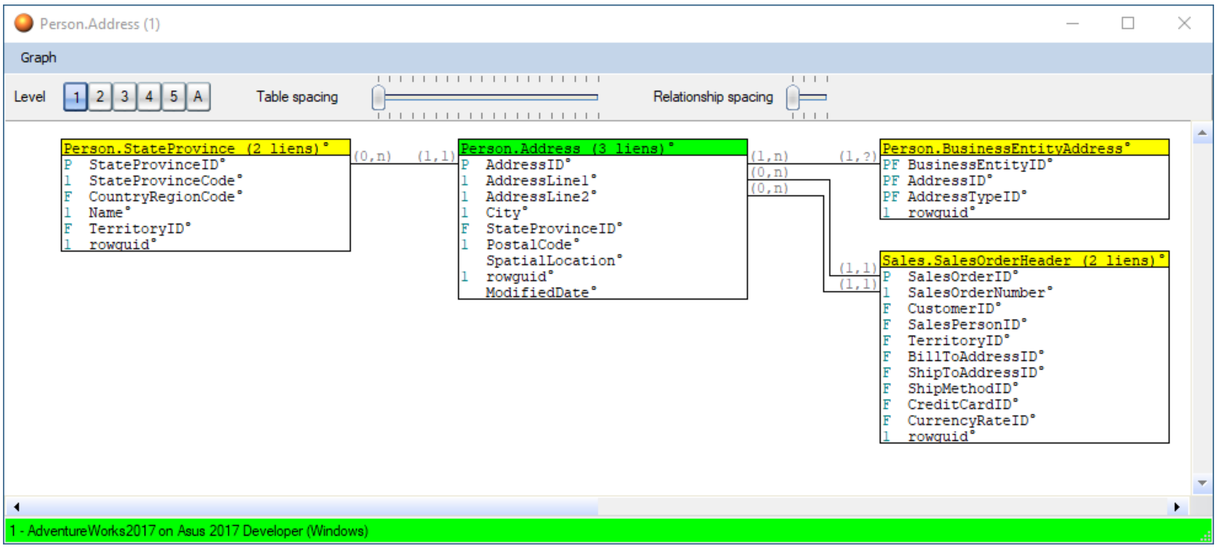
The graph display function allows you to open the graph of the relations of the selected table. The main table is placed in the center, the master tables are on the left and the dependent tables on the right

The goal is not to make a complete schema of the database, but to understand the relationships from a given table

A lot of informations are available via tooltips and context menus

The cardinalities are calculated when the graphs are opened or on demand, depending on the settings of Datamine. They are those found in the database.

Here is an example of a graph:



```
Person.StateProvince (2 liens)°
P  StateProvinceID°
1  StateProvinceCode°
F  CountryRegionCode°
1  Name°
F  TerritoryID°
1  rowguid°
```

```
Person.Address (3 liens)°
P  AddressID°
1  AddressLine1°
1  AddressLine2°
1  City°
F  StateProvinceID°
1  PostalCode°
   SpatialLocation°
1  rowguid°
   ModifiedDate°
```

```
Person.BusinessEntityAddress°
PF BusinessEntityID°
PF AddressID°
PF AddressTypeID°
1  rowguid°
```

```
Sales.SalesOrderHeader (2 liens)°
P  SalesOrderID°
1  SalesOrderNumber°
F  CustomerID°
F  SalesPersonID°
F  TerritoryID°
F  BillToAddressID°
F  ShipToAddressID°
F  ShipMethodID°
F  CreditCardID°
F  CurrencyRateID°
1  rowguid°
```

1 - AdventureWorks2017 on Asus 2017 Developer (Windows)

11. QBE

The QBE function allows you to consult the contents of a table or a view of the database

The functions available for a table are more complete than for a view, since it is possible to use dependencies via foreign keys. Some features are under development

The number of rows displayed by default depends on the settings of Datamine (1000 by default). It is not recommended to use the QBE function to display a large number of rows, it is preferable to use an SQL query

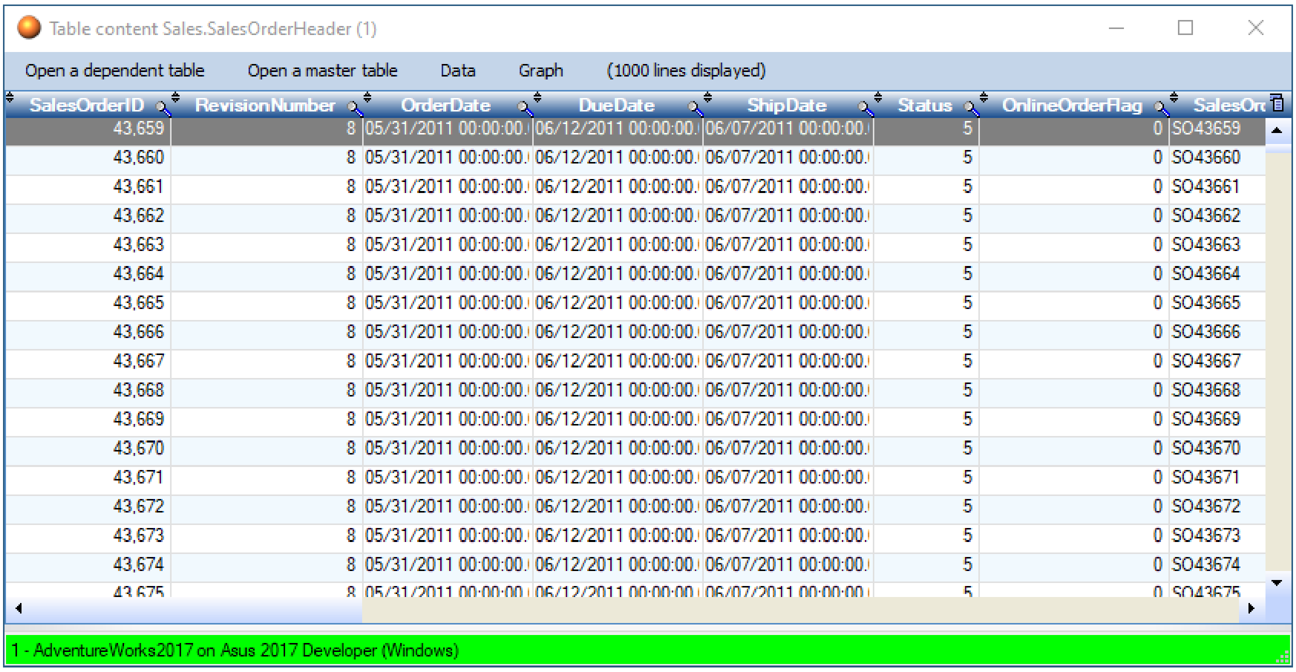
The Data menu contains a whole set of options to manage the data to be displayed. For example, the QBE option allows you to enter conditions for each column, it is also possible to calculate the number of rows in the dependent tables, etc.

The other advantage of the QBE module is that it is possible to open other QBE windows on linked tables (directly or indirectly) and the data remains synchronized, i.e. when a row is selected in main window, all linked windows are updated. This feature makes it very easy to browse a result set with its dependencies

For example, it is possible to open the following QBE windows:

- The table of order headers on which filters can be applied via the QBE menu
- The table dependent on the detail of the orders, which will be updated according to the order selected in the first table
- The customer master table which will also be updated according to the selected order header

Here is an example of QBE windows:



SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status	OnlineOrderFlag	SalesOr
43,659	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43659
43,660	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43660
43,661	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43661
43,662	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43662
43,663	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43663
43,664	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43664
43,665	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43665
43,666	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43666
43,667	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43667
43,668	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43668
43,669	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43669
43,670	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43670
43,671	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43671
43,672	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43672
43,673	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43673
43,674	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43674
43,675	8	05/31/2011 00:00:00.	06/12/2011 00:00:00.	06/07/2011 00:00:00.	5	0	SO43675

1 - AdventureWorks2017 on Asus 2017 Developer (Windows)

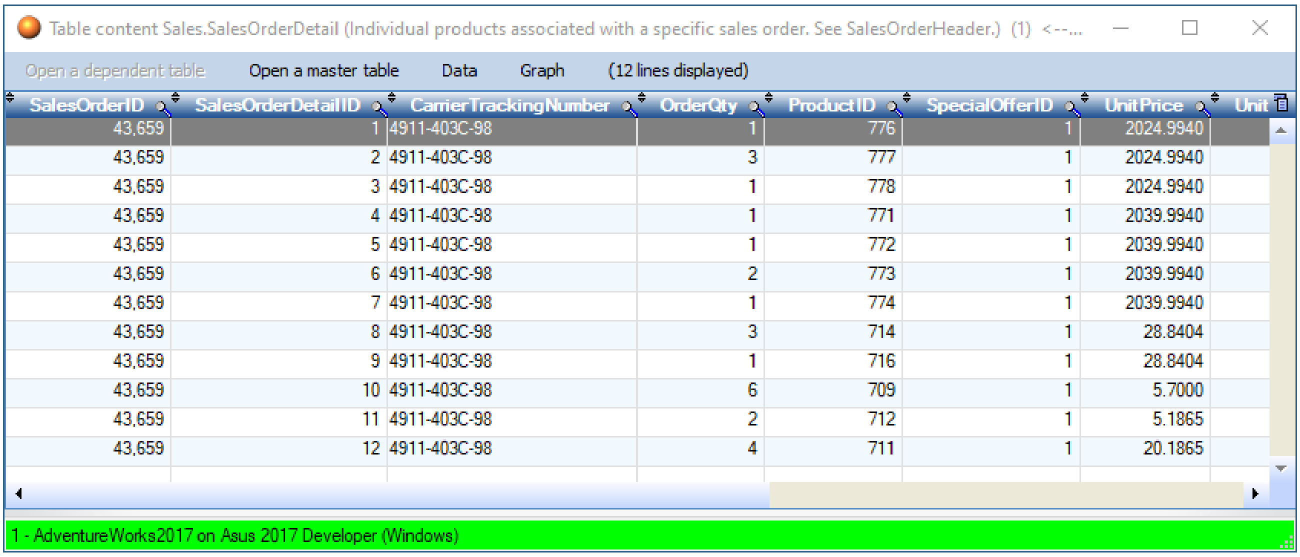


Table content Sales.SalesOrderDetail (Individual products associated with a specific sales order. See SalesOrderHeader.) (1) <--...
Open a dependent table
Open a master table
Data
Graph
(12 lines displayed)

SalesOrderID	SalesOrderDetailID	CarrierTrackingNumber	OrderQty	ProductID	SpecialOfferID	UnitPrice	Unit
43,659	1	4911-403C-98	1	776	1	2024.9940	
43,659	2	4911-403C-98	3	777	1	2024.9940	
43,659	3	4911-403C-98	1	778	1	2024.9940	
43,659	4	4911-403C-98	1	771	1	2039.9940	
43,659	5	4911-403C-98	1	772	1	2039.9940	
43,659	6	4911-403C-98	2	773	1	2039.9940	
43,659	7	4911-403C-98	1	774	1	2039.9940	
43,659	8	4911-403C-98	3	714	1	28.8404	
43,659	9	4911-403C-98	1	716	1	28.8404	
43,659	10	4911-403C-98	6	709	1	5.7000	
43,659	11	4911-403C-98	2	712	1	5.1865	
43,659	12	4911-403C-98	4	711	1	20.1865	

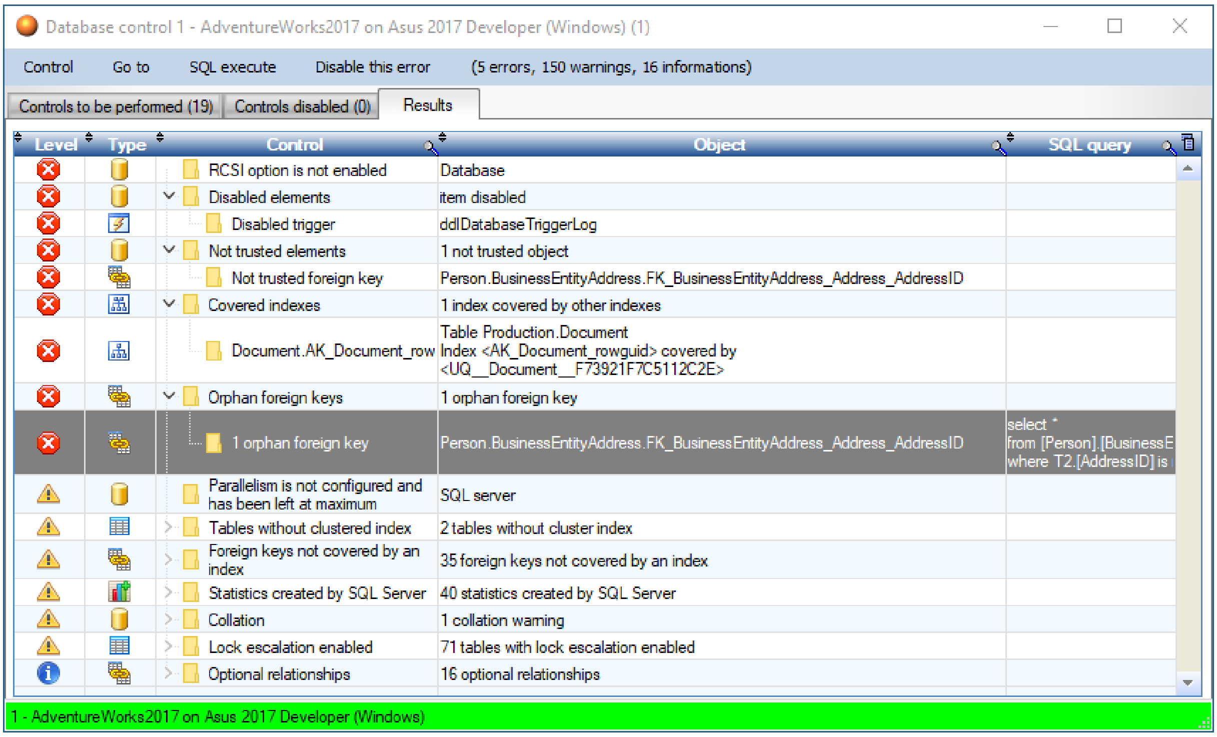
1 - AdventureWorks2017 on Asus 2017 Developer (Windows)

12. Control of a database

Database control function allows a set of controls to be run automatically

It is possible to choose the controls to be carried out and to deactivate some of them (the control globally or a specific anomaly)

Here is an example of the result of a database control:



Database control 1 - AdventureWorks2017 on Asus 2017 Developer (Windows) (1)

(5 errors, 150 warnings, 16 informations)

Level	Type	Control	Object	SQL query
⊗		RCSI option is not enabled	Database	
⊗		Disabled elements	item disabled	
⊗		Disabled trigger	ddlDatabaseTriggerLog	
⊗		Not trusted elements	1 not trusted object	
⊗		Not trusted foreign key	Person.BusinessEntityAddress.FK_BusinessEntityAddress_Address_AddressID	
⊗		Covered indexes	1 index covered by other indexes	
⊗		Document.AK_Document_row	Table Production.Document Index <AK_Document_rowguid> covered by <UQ__Document__F73921F7C5112C2E>	
⊗		Orphan foreign keys	1 orphan foreign key	
⊗		1 orphan foreign key	Person.BusinessEntityAddress.FK_BusinessEntityAddress_Address_AddressID	select * from [Person].[BusinessE where T2.[AddressID] is
⚠		Parallelism is not configured and has been left at maximum	SQL server	
⚠		Tables without clustered index	2 tables without cluster index	
⚠		Foreign keys not covered by an index	35 foreign keys not covered by an index	
⚠		Statistics created by SQL Server	40 statistics created by SQL Server	
⚠		Collation	1 collation warning	
⚠		Lock escalation enabled	71 tables with lock escalation enabled	
ℹ		Optional relationships	16 optional relationships	

1 - AdventureWorks2017 on Asus 2017 Developer (Windows)