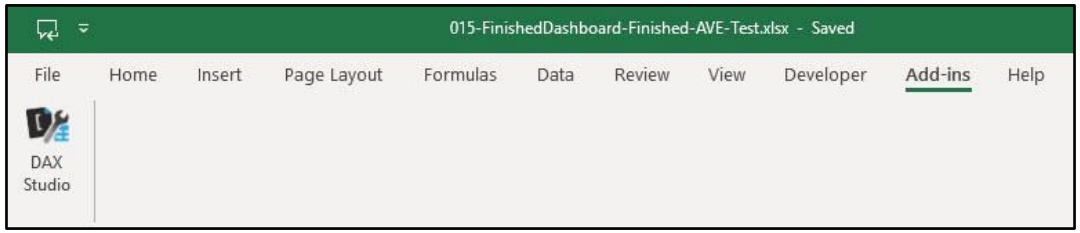
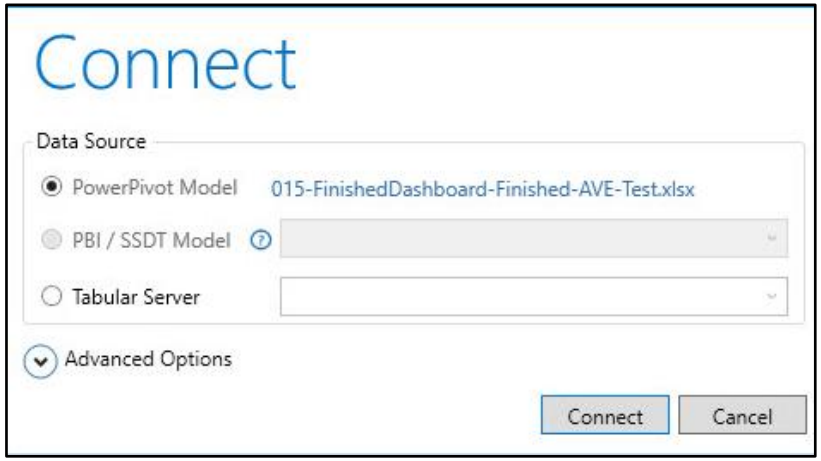
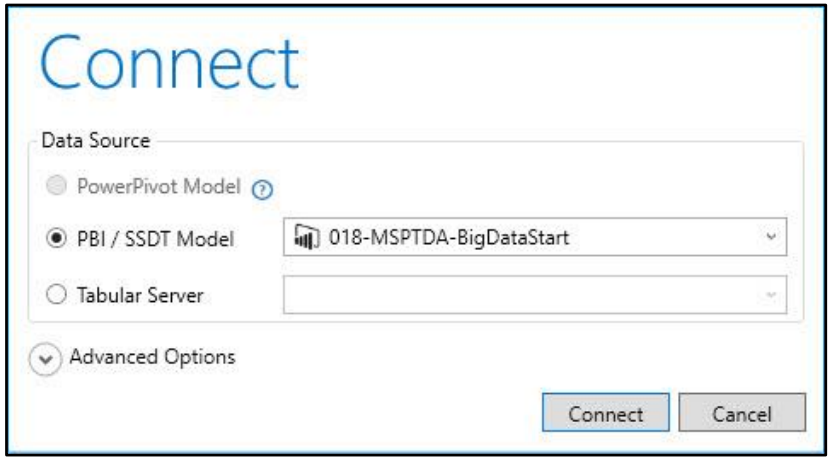
Microsoft Power Tools for Data Analysis #31
Faster DAX Averages: AVERAGEX or DISTINCTCOUNT?
Notes from Video:

Table of Contents

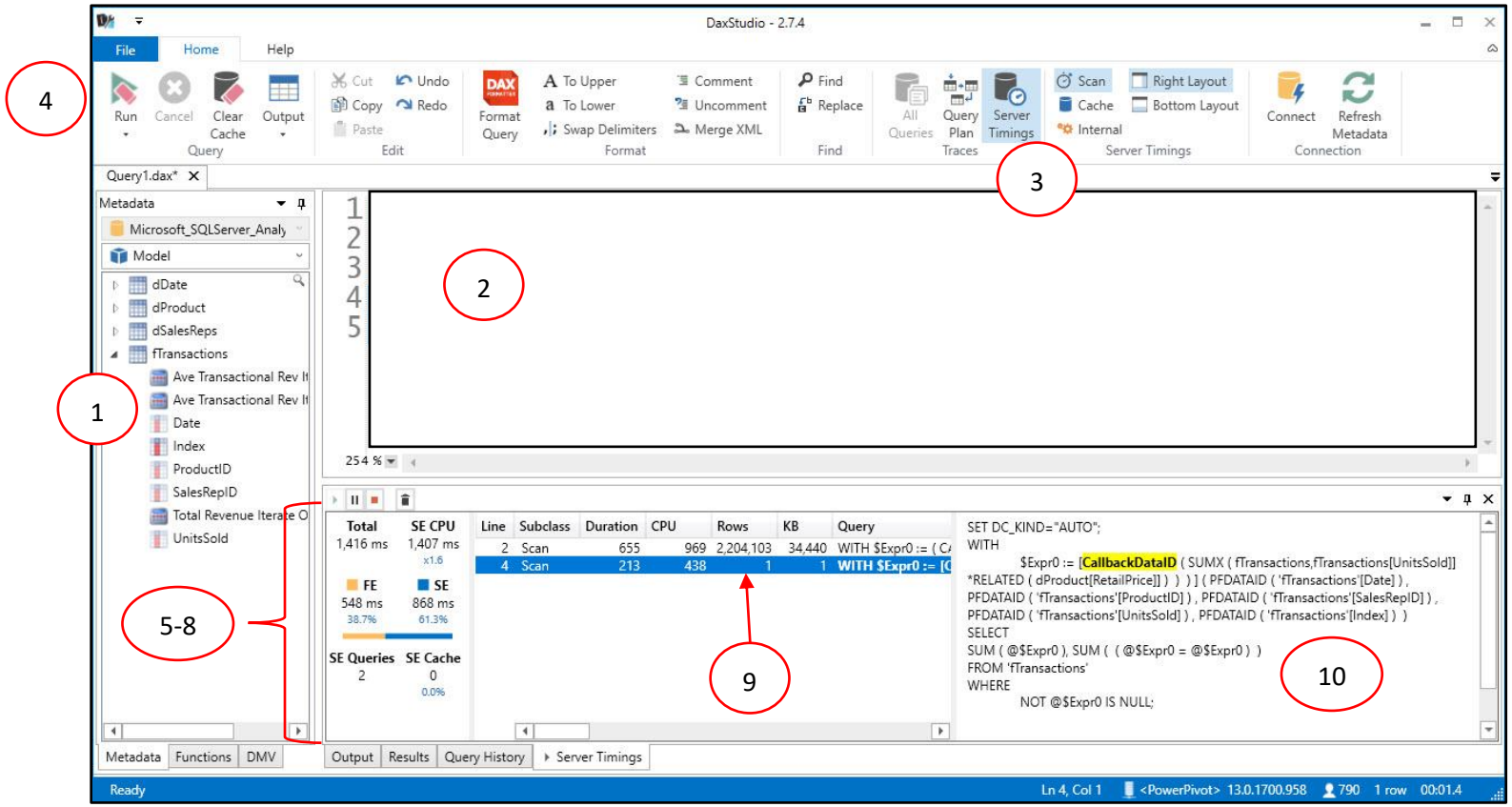
1) **Opening DAX Studio** :
    1. Search Google and download DAX Studio.
    2. In Excel, you can use Add-in button as seen here:
        i. 

    3. In Excel, to Connect to the Excel File, use this dialog box:
        i. 

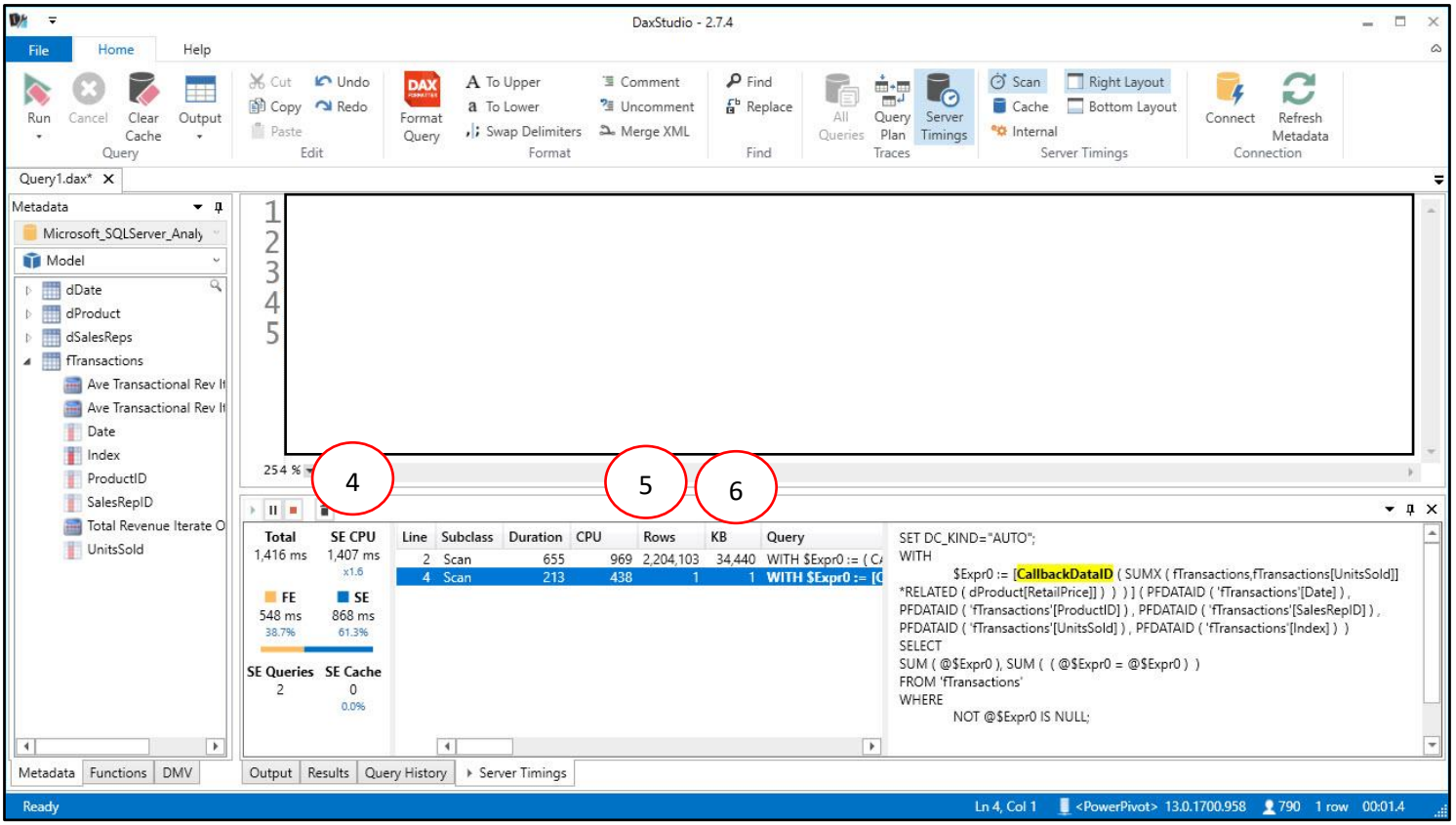
    4. In Power BI Desktop, to connect to a Power BI Desktop file, use the Start Menu or a DAX Studio Shortcut to open DAX Studio, then use the "PBI / SSDT Model" dialog button as seen here: **SSDT = SQL Server Data Tools
        i. 

2) **Dax Studio for Timing Formulas** :

1. Tables, Columns, Calculated Columns and Measures from Data Model
2. Editor pane of DAX Studio (White area) contains the code of your Query. We have to use EVALUATE Command to materialize table. We use the ROW DAX Function with a column name in the first argument and the Measure in the second argument so that we can materialize a one-column and one-row table with our Measure Result.
3. "Server Timing" button in the Traces group will allow us to time the Measure in milliseconds.
4. Click Run and Clear Cache button, to run query and time Measure.
5. FE = Formula Engine. FE can only use one processor to calculate the formula.
6. SE = Storage Engine. SE can use multiple processors to calculate the formula. In general, the more that the SE does the work, the faster the formula will calculate.
7. Total Time (in milliseconds)= SE Time + FE Time.
8. SE CPU = total time SE uses with multiple processors.
9. Rows tells how many rows where in the table that had to be materialized internally during the Measure Calculation process.
10. xmSQL code is the language that is used by the VertiPaq Engine (Columnar Database and Formula Calculation Engine)



3) **More notes on DAX Studio for timing formulas** :



1. Times are in Millisecond ➔ 1000 ms = 1 second.
2. Any time below 16 milliseconds is “noise”. “16 milliseconds is the granularity of the clock”
3. 20, 40 milliseconds not much time…
4. **SE CPU Time**
    1. Time in milliseconds that describes how much CPU has been consumed
    2. Usually time spent by CPU consuming the SE result
    3. This can be bigger than Total because it is using multiple cores (threads)
    4. SE CPU Time = SE Time * Number of Threads (they call then “Cores”)
        i. “Parallelism”
        ii. Each Core or Thread works on one segment (1 Million in Excel and Power BI Desktop and 8 million in SSAS)
5. **Rows**
    1. Number Rows Materialized from SE Query
    2. Row column in DAX Studio may not yield an accurate number : it is just an estimate. You might have to look in Query Plan to see actual number of rows that are materialized.
6. **KB**
    1. Size of materializing

4) **CallbackDataID** :

1. **CallbackDataID** means that the xmSQL language cannot convert the DAX Code in the Formula into something that the SE can read.
2. **CallbackDataID** means the code can be read by the FE but not by the SE
3. This means that there will have to be a back and forth or a callback to the FE, many times, and this can slow down calculations.

5) **Some of what to look for when considering the efficiency of a DAX Formula** :

1. Total Time
2. 2 Parts of Time: FE & SE
3. How Many Queries
4. # Rows Materialized internally in DAX Formula
5. xmSQL Code: CallbackDataID?

6) **Formulas and Timings we saw in Video** :

1. Formulas we saw in Video :

```
EVALUATE
ROW("Ave Country Sales AVEX",AVERAGEX(dCountry,[Total Sales]))

EVALUATE
ROW("Ave Country Sales DC FFK",DIVIDE([Total Sales],DISTINCTCOUNT(fTransactions[CountryCode])))

EVALUATE
ROW("Ave Daily Sales AVEX",AVERAGEX(dDate,[Total Sales]))

EVALUATE
ROW("Ave Daily Sales DC FFK",DIVIDE([Total Sales],DISTINCTCOUNT(fTransactions[Date])))

EVALUATE
ROW("Ave Monthly Sales AVEX",AVERAGEX(VALUES(dDate[Month-Year]),[Total Sales]))

EVALUATE
ROW("Ave Monthly Sales DC DAC",DIVIDE([Total Sales],DISTINCTCOUNT(dDate[Month-Year])))
```

2. Timings using DAX Studio :

```
EVALUATE
ROW("Ave Country Sales AVEX",AVERAGEX(dCountry,[Total Sales]))
```

218 %

Total	SE CPU
12 ms	31 ms x2.8
FE	SE
1 ms 8.3%	11 ms 91.7%
SE Queries	SE Cache
2	0 0.0%

Line	Subclass	Duration	CPU	Rows	KB	Query
2	Scan	10	31	129	3	SELECT 'dCountry'[CountryCode]
4	Scan	1	0	1	1	WITH $Expr0 := [CallbackData

```
SET DC_KIND="AUTO";
WITH
        $Expr0 := [CallbackDataID ( SUM ( fTransactions[Net Sales]] ) ) ] ( PFDATAID ( 'dCountry'[CountryCode] ) )
SELECT
SUM ( @$Expr0 ), SUM (  ( @$Expr0 = @$Expr0 ) )
FROM 'dCountry'
WHERE
        NOT @$Expr0 IS NULL;

'Estimated size ( volume, marshalling bytes ) : 1, 24'
```

```
EVALUATE
ROW("Ave Country Sales DC FFK",DIVIDE([Total Sales],DISTINCTCOUNT(fTransactions[CountryCode])))
```

218 %

Total	SE CPU
4 ms	0 ms x0.0
FE	SE
3 ms 75.0%	1 ms 25.0%
SE Queries	SE Cache
1	0 0.0%

Line	Subclass	Duration	CPU	Rows	KB	Query
2	Scan	1	0	1	1	SELECT SUM ('fTransactions'[Net

```
SET DC_KIND="AUTO";
SELECT
SUM ( 'fTransactions'[Net Sales] )
FROM 'fTransactions';

'Estimated size ( volume, marshalling bytes ) : 1, 16'
```

```
EVALUATE
ROW("Ave Daily Sales AVEX",AVERAGEX(dDate,[Total Sales]))
```

218 %

Total	SE CPU
17 ms	47 ms
	x3.1
FE	SE
2 ms	15 ms
11.8%	88.2%
SE Queries	SE Cache
2	0
	0.0%

Line	Subclass	Duration	CPU	Rows	KB	Query
2	Scan	14	47	1,464	23	SELECT 'dCalendar'[Date], SUM (
4	Scan	1	0	1	1	WITH $Expr0 := [CallbackData

```
SET DC_KIND="AUTO";
WITH
        $Expr0 := [CallbackDataID ( SUM ( fTransactions[Net Sales]] ) ) ] ( PFDATAID ( 'dCalendar'[Date] ) )
SELECT
SUM ( @$Expr0 ), SUM (  ( @$Expr0 = @$Expr0 ) )
FROM 'dCalendar'
WHERE
        NOT @$Expr0 IS NULL;


'Estimated size ( volume, marshalling bytes ) : 1, 24'
```

```
EVALUATE
ROW("Ave Daily Sales DC FFK",DIVIDE([Total Sales],DISTINCTCOUNT(fTransactions[Date])))
```

218 %

Total	SE CPU
5 ms	0 ms
	x0.0
FE	SE
3 ms	2 ms
60.0%	40.0%
SE Queries	SE Cache
1	0
	0.0%

Line	Subclass	Duration	CPU	Rows	KB	Query
2	Scan	2	0	1	1	SELECT SUM ('fTransactions'[Net

```
SET DC_KIND="AUTO";
SELECT
SUM ( 'fTransactions'[Net Sales] )
FROM 'fTransactions';


'Estimated size ( volume, marshalling bytes ) : 1, 16'
```

```
EVALUATE
ROW("Ave Monthly Sales AVEX",AVERAGEX(VALUES(dDate[Month-Year]),[Total Sales]))
```

218 %

Total	SE CPU
13 ms	63 ms x6.3
FE	SE
3 ms 23.1%	10 ms 76.9%
SE Queries	SE Cache
1	0 0.0%

Line	Subclass	Duration	CPU	Rows	KB	Query
2	Scan	10	63	51	1	SELECT 'dCalendar'[Column], SU

```
SET DC_KIND="AUTO";
SELECT
'dCalendar'[Column],
SUM ( 'fTransactions'[Net Sales] )
FROM 'fTransactions'
        LEFT OUTER JOIN 'dCalendar' ON 'fTransactions'[Date]='dCalendar'[Date];


'Estimated size ( volume, marshalling bytes ) : 51, 816'
```

```
EVALUATE
ROW("Ave Monthly Sales DC DAC",DIVIDE([Total Sales],DISTINCTCOUNT(dDate[Month-Year])))
```

218 %

Total	SE CPU
4 ms	0 ms x0.0
FE	SE
1 ms 25.0%	3 ms 75.0%
SE Queries	SE Cache
1	0 0.0%

Line	Subclass	Duration	CPU	Rows	KB	Query
2	Scan	3	0	1	1	SELECT SUM ('fTransactions'[Ne

```
SET DC_KIND="AUTO";
SELECT
SUM ( 'fTransactions'[Net Sales] )
FROM 'fTransactions';


'Estimated size ( volume, marshalling bytes ) : 1, 16'
```