



Power BI

Actualización Incremental



Ing. Alejandro Cabanchik
Consultor BI

<https://www.smartbi.com.ar/>



Alejandro Cabanchik



Alejandro.Cabanchik@smartbi.com.ar



@ale_cabanchik



t.me/ale_cabanchik



t.me/powerbiargentina

<https://www.smartbi.com.ar/>



smart BI

Microsoft
Partner

Silver Data Analytics
Silver Data Platform

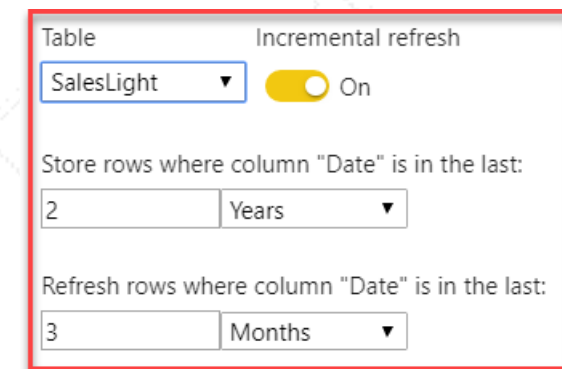
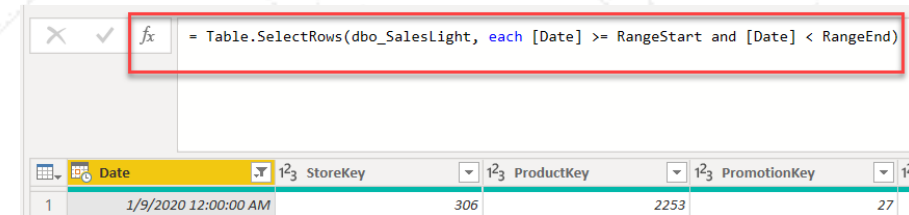
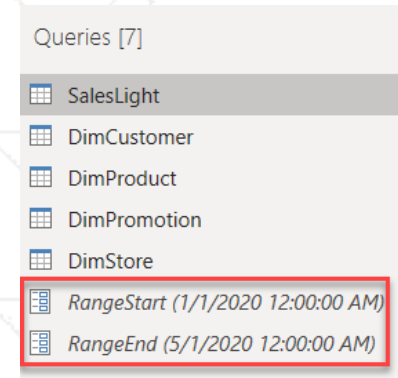


Actualización incremental

- “Capacidad de actualizar tablas por rangos de fecha”
 - Basado en Particiones
 - En Modo Importado
 - “Query folding”
- Ayuda a refrescar grandes conjuntos de datos
 - Menor tiempo procesamiento
 - Menor consumo de recursos
- Licencia Pro
 - Inicialmente en Power BI Premium
 - Desde Febrero 2020 en Pro
 - Power BI ideas

Configuración

- En Power BI Desktop
 - Definimos
 - Filtros
 - Políticas (cuanta historia y últimos periodos)
- En Power BI Services
 - Creación de particiones
 - Ejecución de políticas de actualización
- Parámetros RangeStart – RangeEnd
 - Tipo Datetime
 - Filtro en Power Query
 - \geq RangeStart
 - $<$ RangeEnd
 - Cada partición define sus valores límites



- ☐ Detect data changes [Learn more](#)
- ☐ Only refresh complete months [Learn more](#)

Advertencias

- ¡¡¡Guardar copia!!!
 - No podemos descargar nuevamente
- ¡¡¡Verificar para evitar Datos duplicados!!!
 - Utilizar Fecha estable
 - ¿Fecha de transacción?
 - ¡¡¡Si utilizamos fecha de modificación podemos duplicar registros!!!
 - RangeEnd
 - Es un límite externo
 - Inicio del siguiente rango
 - WHERE ColumnaFecha >= RangeStart AND ColumnaFecha < RangeEnd

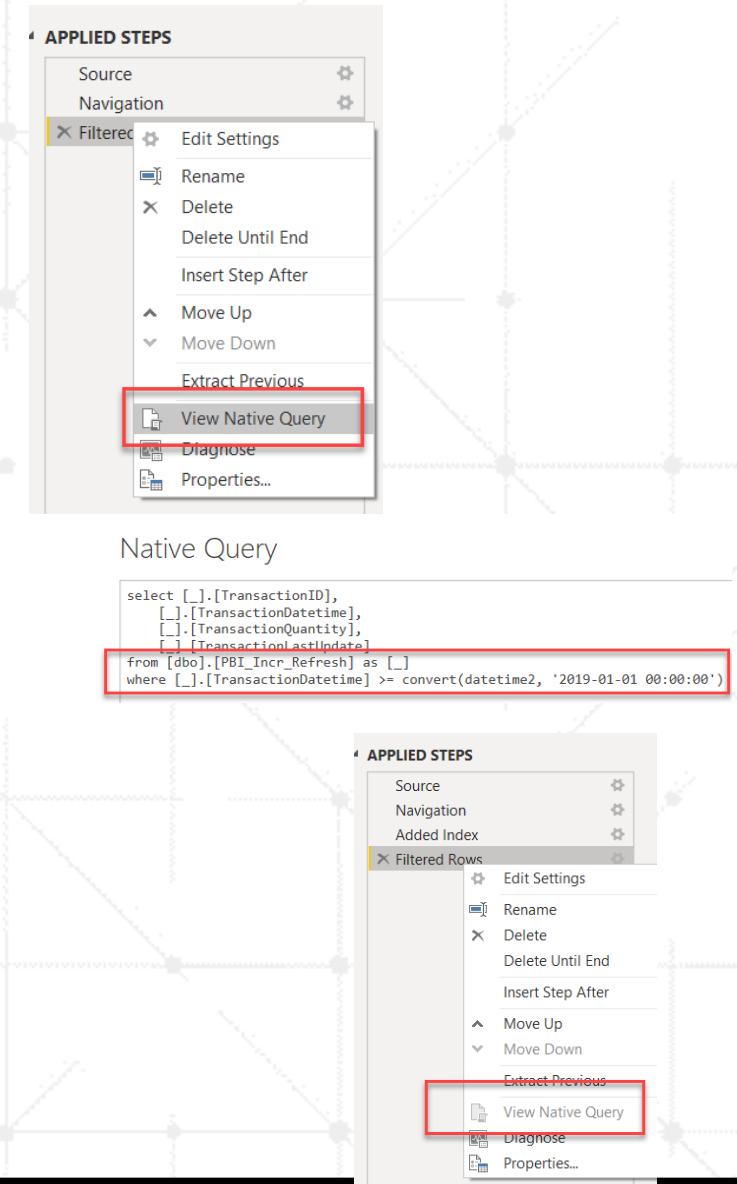
Transaction...	TransactionDatetime	Transaction...	Transactio
1	2019-01-01 00:00:00.000	1.0000	2019-01-0
2	2019-02-01 00:00:00.000	1.0000	2019-02-0
3	2019-03-01 00:00:00.000	1.0000	2019-03-0
4	2019-04-01 00:00:00.000	1.0000	2019-04-0
5	2019-05-01 00:00:00.000	1.0000	2019-05-0
6	2019-06-01 00:00:00.000	1.0000	2019-06-0
7	2019-07-01 00:00:00.000	1.0000	2019-07-0
8	2019-08-01 00:00:00.000	1.0000	2019-08-0
9	2019-09-01 00:00:00.000	1.0000	2019-09-0
10	2019-10-01 00:00:00.000	1.0000	2019-10-0
11	2019-11-01 00:00:00.000	1.0000	2019-11-0
12	2019-12-01 00:00:00.000	1.0000	2019-12-0
13	2020-01-01 00:00:00.000	1.0000	2020-01-0
14	2020-02-01 00:00:00.000	1.0000	2020-02-0
15	2020-03-01 00:00:00.000	1.0000	2020-03-0
16	2020-04-01 00:00:00.000	1.0000	2020-04-0

10	2019-10-01 00:00:00.000	1.0000	2019-10-01 00:00:00.000
11	2019-11-01 00:00:00.000	1.0000	2019-11-01 00:00:00.000
12	2020-04-01 00:00:00.000	1.0000	2019-12-01 00:00:00.000
13	2020-01-01 00:00:00.000	1.0000	2020-01-01 00:00:00.000

TransactionID	TransactionDatetime	TransactionQuant
1	1/1/2019 12:00:00 AM	1.
2	2/1/2019 12:00:00 AM	1.
3	3/1/2019 12:00:00 AM	1.
4	4/1/2019 12:00:00 AM	1.
5	5/1/2019 12:00:00 AM	1.
6	6/1/2019 12:00:00 AM	1.
7	7/1/2019 12:00:00 AM	1.
8	8/1/2019 12:00:00 AM	1.
9	9/1/2019 12:00:00 AM	1.
10	10/1/2019 12:00:00 AM	1.
11	11/1/2019 12:00:00 AM	1.
12	12/1/2019 12:00:00 AM	1.
12	4/1/2020 12:00:00 AM	1.
13	1/1/2020 12:00:00 AM	1.
14	2/1/2020 12:00:00 AM	1.

Query folding

- Es importante empujar el filtro a la fuente de datos
- Si la fuente no soporta Query folding
 - Flat files, Blob, Web, dataflow
 - El filtro se produce localmente
 - Viajan todos los datos y se filtran a posterior
 - No mejorarían tiempos
 - Puede agotar recursos de procesamiento
- Power BI Desktop verifica existencia de Query folding
 - En algunos tipos de orígenes debemos verificar con un Trace



Campo No Datetime

- Castear en Power Query

- Date

- CampoFecha \geq Datetime.Date(RangeStart) And
CampoFecha $<$ Datetime.Date(RangeEnd)

```
fx = Table.SelectRows(dbo_PBI_Incr_Refresh, each [TransactionDate] >= DateTime.Date(RangeStart) and [TransactionDate] < DateTime.Date(RangeEnd))
```

- Numérico

- Date.Year(RangeStart) * 10000 + Date.Month(RangeStart)
* 100 + Date.Day(RangeStart)

Queries [4]

- PBI_Incr_Refresh
- RangeStart (1/1/2019 12:00:00 AM)
- RangeEnd
- fx fnDateTok

Enter Parameter

x

```
fx = (x as datetime) => Date.Year(x)*10000 + Date.Month(x)*100 + Date.Day(x)
```

```
= Table.SelectRows(dbo_PBI_Incr_Refresh, each [TransactionDateKey] >= fnDateToKey RangeStart) and [TransactionDateKey] < fnDateToKey(RangeEnd))
```

Hoy

- La fecha actual es determinada en Power BI Services
- Utiliza el Time Zone definido en la planificación del conjunto de datos
 - En actualizaciones manuales y automáticas

Scheduled refresh

Keep your data up to date

☒ On

Refresh frequency Daily

Time zone (UTC-03:00) Montevideo

Detectar cambios de datos

- Permite reducir las particiones a procesar
 - Genera consultas al origen para determinar el procesamiento
- Debe ser un campo fecha diferente
 - Fecha de Modificación
- Requiere que la columna este en el modelo
- Podemos reducir tamaño por
 - Guardar el máximo valor (Power Query)
 - Modificar precisión del campo
 - Definir un custom query para detectar cambios (Premium)

Table: PBI_Incr_Refresh

Incremental refresh: On

Store rows where column "TransactionDatetime" is in the last: 1 Years

Refresh rows where column "TransactionDatetime" is in the last: 2 Months

☒ Detect data changes [Learn more](#)

Only refresh data in the last 2 months if the maximum value of this column is less than or equal to the value in the refresh range

TransactionLastUpdate

☐ Only refresh complete months [Learn more](#)

Actualizar solo periodos completos

- Permite evitar incorporar periodos en curso
 - Excluye en periodo vigente

Table: PBI_Incr_Refresh

Incremental refresh: ☒ On

Store rows where column "TransactionDatetime" is in the last: 1 Years

Refresh rows where column "TransactionDatetime" is in the last: 2 Months

☐ Detect data changes [Learn more](#)

☒ Only refresh complete months [Learn more](#)

Power BI Premium

- XMLA endpoint
 - Actualizar desde Management Studio (SSMS)
 - Procesar particiones puntuales
 - Modificar el comportamiento de la actualización
 - TMSL + TOM
 - applyRefreshPolicy → False → Procesamiento full
 - effectiveDate → modificar fecha de hoy
 - Query custom para detector cambios
 - TMSL + TOM
 - Permitiría evitar tener la columna "Fecha modificación" en el modelo
 - pollingExpression → expresión M que se aplica a cada partición
- Desplegar solo Metadata
 - Power BI Desktop siempre actualiza modelo completo, forzando procesamiento full

Use partitions to divide a table into logical parts that can be processed independently.

Table: SalesLight

Partitions

Partition Name	# Rows	Last Processed
2018	3615870	4/14/2020 10:06:43 AM
2019	4579586	4/14/2020 10:06:48 AM
2020Q101	408394	4/14/2020 10:06:08 AM
2020Q102	407737	4/14/2020 10:05:42 AM
2020Q103	469698	4/14/2020 10:06:08 AM
2020Q204	213740	4/14/2020 10:06:07 AM

KMLAQuery1.xml...ctor@pepsico.com)

```
{
  "refresh": {
    "type": "full",
    "objects": [
      {
        "database": "PBI Inc - 2 - 12MM - Incr",
        "table": "SalesLight",
        "partition": "2018"
      }
    ]
  },
  "createOrReplace": {
    "object": {
      "database": "PBI Inc - 2 - 12MM - Incr",
      "table": "SalesLight",
      "partition": "2020Q102"
    },
    "partition": {
      "name": "2020Q102",
      "mode": "import",
      "source": {
        "type": "policyRange",
        "start": "2020-02-01T00:00:00",
        "end": "2020-03-01T00:00:00",
        "granularity": "month"
      }
    }
  }
}
```

Power BI dataflow

- Incremental en Premium
- No podemos definir nuestras condiciones de Where
- Dataflow no es una fuente que soporte Query folding

The screenshot displays the 'Query settings' pane for a dataflow named 'SalesLight'. The 'Entity type' is set to 'Custom'. The 'Applied steps' list includes 'Source' and 'Navigation'. The 'Incremental refresh' settings are configured as follows:

- On** (toggle)
- Choose a DateTime field to filter by:** Date
- Store rows from the past:** 2 Years
- Refresh rows from the past:** 2 Months
- ☐ Detect data changes [Learn more](#)
- ☐ Only refresh data if the maximum value in this field changes [Learn more](#)
- ☐ Only refresh complete months [Learn more](#)

A red box highlights the DAX formula in the 'Where' clause: `Table.SelectRows(Navigation, each DateTime.From([Date]) >= RangeStart and DateTime.From([Date]) < RangeEnd)`. Below the settings, a table of data is shown with columns: StoreKey, ProductKey, PromotionKey, CustomerKey, SalesQuantity, SalesAmount, and UpdateDate. The table contains 6 rows of data.

	StoreKey	ProductKey	PromotionKey	CustomerKey	SalesQuantity	SalesAmount	UpdateDate
2:00:00 ...	306	782	10	333	1	10.36	
2:00:00 ...	306	782	10	334	1	10.36	
2:00:00 ...	306	782	10	335	1	10.36	
2:00:00 ...	306	782	10	336	1	10.36	
2:00:00 ...	306	782	10	337	1	10.36	
2:00:00 ...	306	1701	10	333	1	3.98	



¡GRACIAS!

¿Alguna pregunta?

Si no tienes ninguna ahora, puedes comunicarte luego:



Alejandro.Cabanchik@smartbi.com.ar



@ale_cabanchik



t.me/ale_cabanchik



t.me/powerbiargentina

<https://www.smartbi.com.ar/>