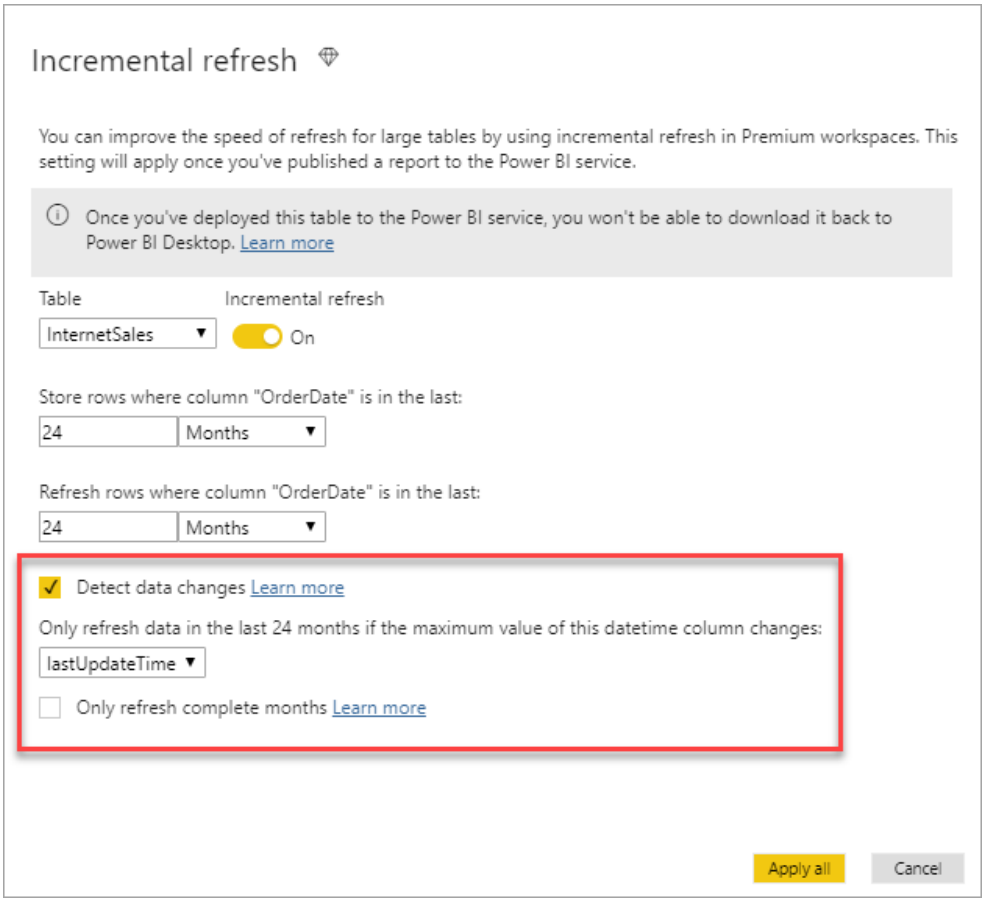
7 things to know about Detect Data Changes with Incremental Refresh

1. Use a different date field

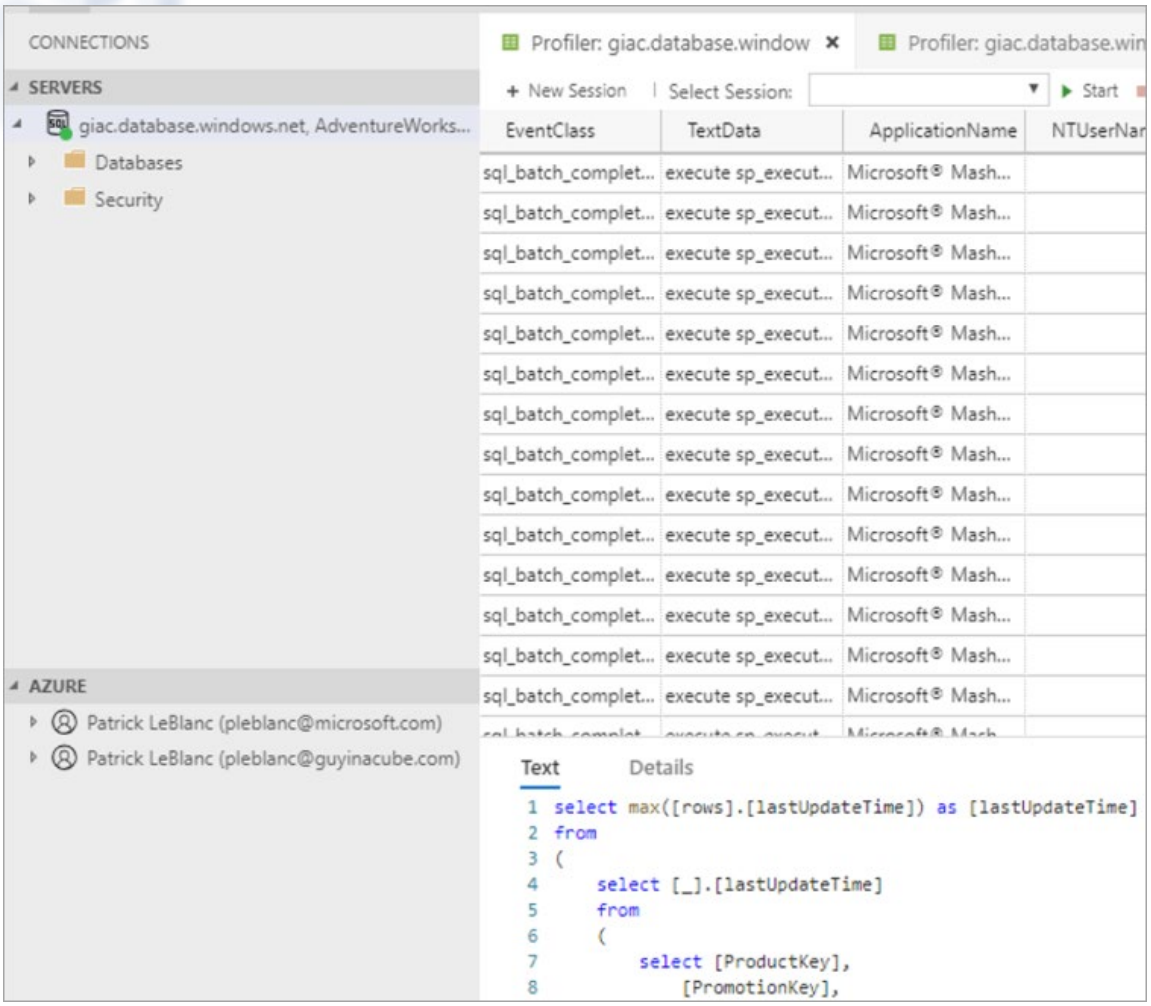
When configuring the Detect data changes option, use a different datetime column than the one use to specify the ranges. This is typically an audit column that is updated when a row is inserted or updated.



2. Use a monitoring tool

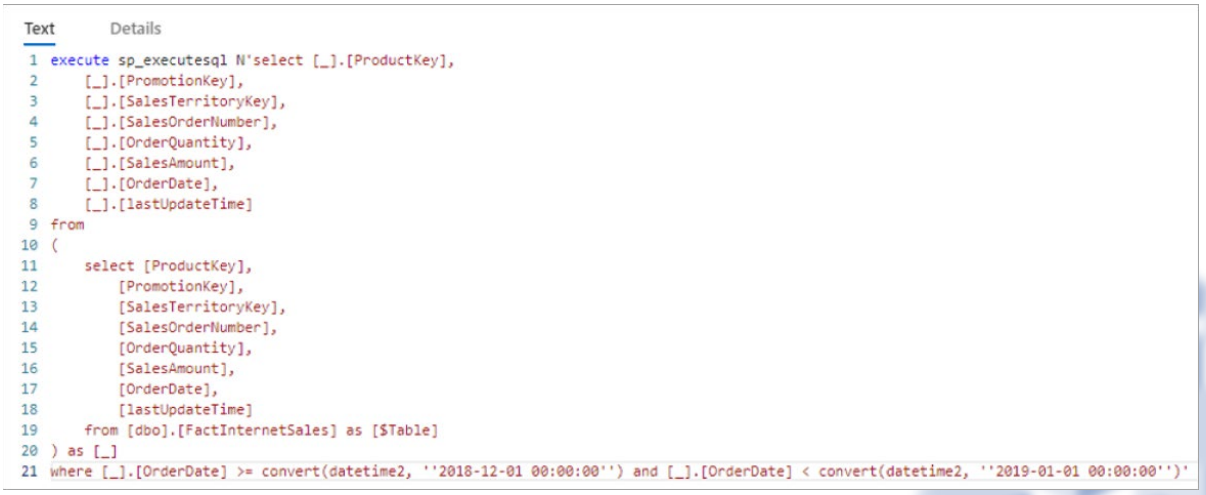
You can use a monitoring tool such as SQL Server Profiler or Azure Data Studio to view refresh activity against your SQL Server. Or, use the tool that works with your data source.

During the initial (first) refresh, when using incremental refresh, all time periods (partitions) will be refreshed.



3. Knowing which time period will be updated

When the audit column is updated during an insert or update, only the corresponding time period (partition) is refreshed in the model.



4. Refresh a specific time period

This relates to the previous item. You can refresh a specific time period (partition) by updating the audit column of any row within that time period. Make sure the audit column is later than the last refresh time!

5. Make sure the requests are optimal

As a best practice, run the captured queries against your backend relational database to ensure that the execution is optimal. For SQL Server, you are looking at the execution plan.

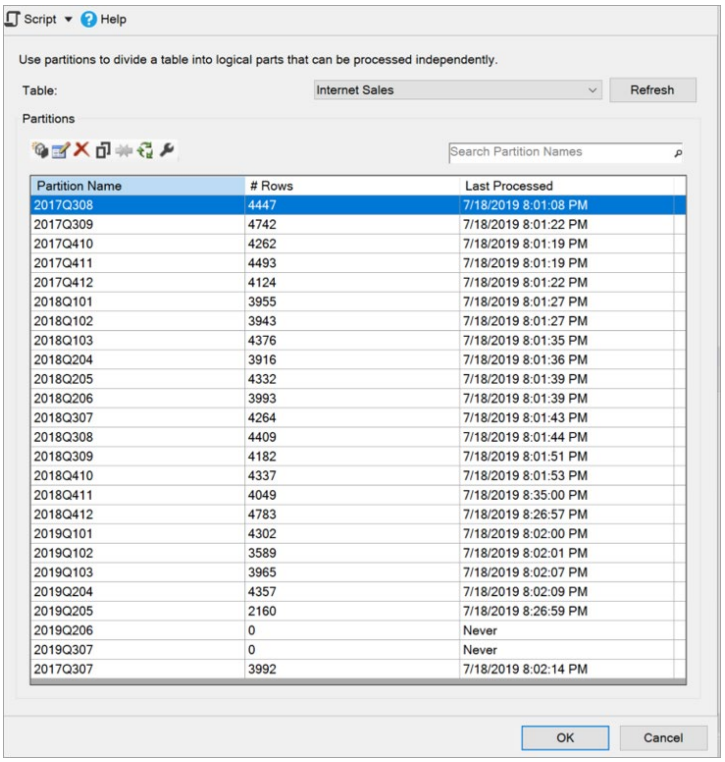
Adding indexes to the source can dramatically reduce the amount of time spent refreshing the data. The below SQL statement will create an index that can be used to optimize queries run to capture the maximum update time for each partition.

```
CREATE INDEX IX_FactInternetSales_OrderDate_Include_LastUpdateTime
ON dbo.FactInternetSales (OrderDate)
INCLUDE (lastupdatetime)
```

6. View partitions in the model

You can view the partition and when they were last updated by connecting to the XMLA endpoint for your Power BI App Workspace that is backed by Power BI Premium capacity.

- Use **SQL Server Management studio** to connect to the XMLA Endpoint using Analysis Services
- Expand the folder labeled **Database**.
- Expand the folder labeled **Tables**.
- Right-click on the table that is configured for Incremental Refresh
- Select **Partitions** from the context menu



Partition Name	# Rows	Last Processed
2017Q308	4447	7/18/2019 8:01:08 PM
2017Q309	4742	7/18/2019 8:01:22 PM
2017Q410	4262	7/18/2019 8:01:19 PM
2017Q411	4493	7/18/2019 8:01:19 PM
2017Q412	4124	7/18/2019 8:01:22 PM
2018Q101	3955	7/18/2019 8:01:27 PM
2018Q102	3943	7/18/2019 8:01:27 PM
2018Q103	4376	7/18/2019 8:01:35 PM
2018Q204	3916	7/18/2019 8:01:36 PM
2018Q205	4332	7/18/2019 8:01:39 PM
2018Q206	3993	7/18/2019 8:01:39 PM
2018Q307	4264	7/18/2019 8:01:43 PM
2018Q308	4409	7/18/2019 8:01:44 PM
2018Q309	4182	7/18/2019 8:01:51 PM
2018Q410	4337	7/18/2019 8:01:53 PM
2018Q411	4049	7/18/2019 8:35:00 PM
2018Q412	4783	7/18/2019 8:26:57 PM
2019Q101	4302	7/18/2019 8:02:00 PM
2019Q102	3589	7/18/2019 8:02:01 PM
2019Q103	3965	7/18/2019 8:02:07 PM
2019Q204	4357	7/18/2019 8:02:09 PM
2019Q205	2160	7/18/2019 8:26:59 PM
2019Q206	0	Never
2019Q307	0	Never
2017Q307	3992	7/18/2019 8:02:14 PM

7. Be careful with deletes!

Simply deleting a row in the time period (partition) will not cause a refresh of that partition. Consider using soft deletes instead.

- Add a column that will be used to do a soft delete of the column. Ensure that the default value of the column is 0 or False. For example, you can add a column named *isDeleted* as a bit column. Instead of deleting a row, you set the value of the column to true or 1.
- In addition to updating the *isDeleted* column, update the value of the audit column used in the Incremental Refresh configuration. The following query is a sample pattern that can be used to implement the soft delete.

```
UPDATE top(1) FactInternetSales

SET
       lastUpdateTime = GETDATE(),
       isDeleted = 1
```

- In the **Power Query Editor**, filter all the rows where the value for the isDeleted column is true or 1. This will remove those rows from the model during a refresh.