

Serve data for analysis with Azure Synapse Analytics

Note:

You are not required to complete the processes, tasks, activities, or steps presented in this example. The various samples provided are for illustrative purposes only and it's likely that if you try this out you will encounter issues in your system.

The aim of all this work is to be able to serve up the data to be consumed by the various consumers within the business. This can include business users and data analysts creating reports and dashboards to Power BI, to enabling data scientist perform model training for predictive analytics as shown in the following steps.

1. Select the Develop hub.



Home



Data



Develop



Integrate



Monitor



Manage

2. Expand Notebooks, then select **1 Products Recommendation**.

Develop

Filter resources by name

- SQL scripts 4
- Notebooks 3**
 - 1 Products Recommendation**
 - 2 AutoML Number of Customer Visits...
 - 3 Campaign Analytics Data Prep
- Data flows 1
- Power BI 1

1 Products Recommendation

Product recommendations is a filtering system that predicts and shows the items that a user would likely purchase based on their purchase history.

Note:

This notebook is written in Scala, and there is interoperability between Scala and Python code.

Steps:

- 1) Data is ingested from Azure Synapse Data Warehouse using PySpark.
- 2) The model is trained using the PySpark ML-Lib recommendations module.
- 3) Product recommendations are generated for the user.

Connecting to Azure Synapse Data Warehouse

Connection to Azure Synapse Data Warehouse is initiated and the required data is ingested for processing. The warehouse is connected with a single line of code. Just point to actions in a table, click on a new notebook, and then click on "Load to DataFrame".

After providing the necessary details, the required data is loaded in the form of a Spark dataframe. One magical line of code converts a dataframe from Scala to Python!

Cell 3

```
[3] 1 %pyspark
      2 spark.read.csv('abfss://machine-learning@asadatalakecto.dfs.core.windows.net/customer_data.csv', header=True)
```

This Synapse Notebook provides customized product recommendations.

i) Data is ingested from CSV files using PySpark (**cell 4**).

Develop

Filter resources by name

- SQL scripts 8
- Notebooks 6**
 - 1 Products Recommendation**
 - 2 AutoML Number of Customer Visits...
 - 3 Campaign Analytics Data Prep
 - Activity 05 - Model Training
 - Lab 06 - Machine Learning
 - Lab 07 - Spark ML
- Data flows 1
- Power BI 1

1 Products Recommendation

After providing the necessary details, the required data is loaded in the form of a Spark dataframe. One magical line of code converts a dataframe from Scala to Python!

Cell 4

```
[4] 1 %pyspark
      2 customer_data = spark.read.load('abfss://machine-learning@asadatalakecto.dfs.core.windows.net/customer_data.csv', format='csv', header=True)
      3
      4 customer_data.show(10)
```

customer_id	product_id	rating	product_name
1402	29	5	Gray with white s...
51036	4	3	Brown Surfboard
33662	23	5	Crystal Wineglass
73162	3	5	Blue Surf Board
14164	17	5	Wood and Cork Coa...
36731	30	5	Brown Shoes
90545	14	1	Designer Coaster
36574	2	5	Retro surfboard
20246	26	3	Black Shoes
14262	14	1	Designer Coaster

The model is trained using the PySpark ML-Lib recommendations module (**cells 7 & 8**).

Develop

Filter resources by name

- SQL scripts 8
- Notebooks 6
 - 1 Products Recommendation
 - 2 AutoML Number of Customer Visit...
 - 3 Campaign Analytics Data Prep
 - Activity 05 - Model Training
 - Lab 06 - Machine Learning
 - Lab 07 - Spark ML
- Data flows 1
- Power BI 1

1 Products Recommendation

Attach to SparkPool01 Language PySpark (Python)

```

1 %%pyspark
2 product_recommendations = []
3
4 for key,value in trained_model.productFeatures().collect():
5     product_recommendations += calculate_similarities(key, value, 0.75)

```

```

1 %%pyspark
2
3 recommend_df = spark.createDataFrame(product_recommendations, ['ProductId', 'RecommendedProductId', '
4
5 result = recommend_df \
6     .join(product_info, recommend_df.ProductId == product_info.product_id, how='inner') \
7     .withColumnRenamed('product_name', 'ProductName') \
8     .select('ProductId', 'ProductName', 'RecommendedProductId', 'Similarity') \
9     .join(product_info, recommend_df.RecommendedProductId == product_info.product_id, how='inner') \
10    .withColumnRenamed('product_name', 'RecommendedProductName') \
11    .select('ProductId', 'ProductName', 'RecommendedProductId', 'RecommendedProductName', 'Similarity
12    .orderBy('ProductId')
13    result.show(100)

```

7	Red Surfboard	7	Red Surfboard	1.0
7	Red Surfboard	29	Gray with white s...	0.788370071801426
7	Red Surfboard	4	Brown SurfBoard	0.8351011661754266
8	Red Corkscrew	12	Blue Coaster	0.9567298352775614
8	Red Corkscrew	28	Black with red so...	0.7668994862051393
8	Red Corkscrew	8	Red Corkscrew	1.0
8	Red Corkscrew	32	Beach Shoes	0.759050145099239
9	Wine corkscrew	5	Orange SurfBoard	0.7668548454674974
9	Wine corkscrew	21	Cheese circle	0.8447028464987419
9	Wine corkscrew	28	Black with red so...	0.8323387361546503
9	Wine corkscrew	9	Wine corkscrew	1.0
10	Steel corkscrew	10	Steel corkscrew	1.0
10	Steel corkscrew	31	Blue Shoes	0.7542318899283764
10	Steel corkscrew	5	Orange SurfBoard	0.8455631051308501
10	Steel corkscrew	16	Turkish Lira	0.8185468131850703
10	Steel corkscrew	30	Brown Shoes	0.754519117277862
11	Black corkscrew	27	Pink Shoes	0.829772070801043
11	Black corkscrew	11	Black corkscrew	1.0
11	Black corkscrew	23	Crystal Wineglass	0.7634615492067875

```

1 result \
2     .repartition(1) \
3     .write.format('csv') \
4     .option("header", "true") \
5     .mode("overwrite") \
6     .save('abfss://machine-learning@asadatalakecto.dfs.core.windows.net/product-recommendations.csv')

```

3. Select the Develop hub.



Home



Data



Develop



Integrate

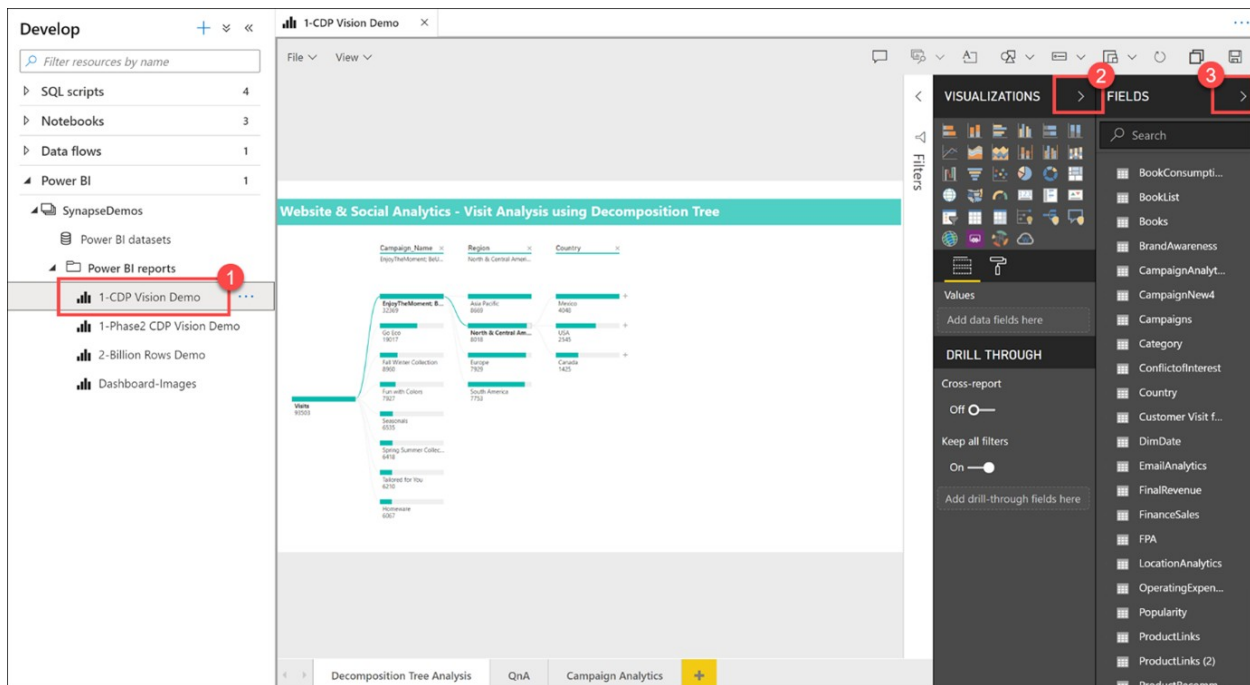


Monitor



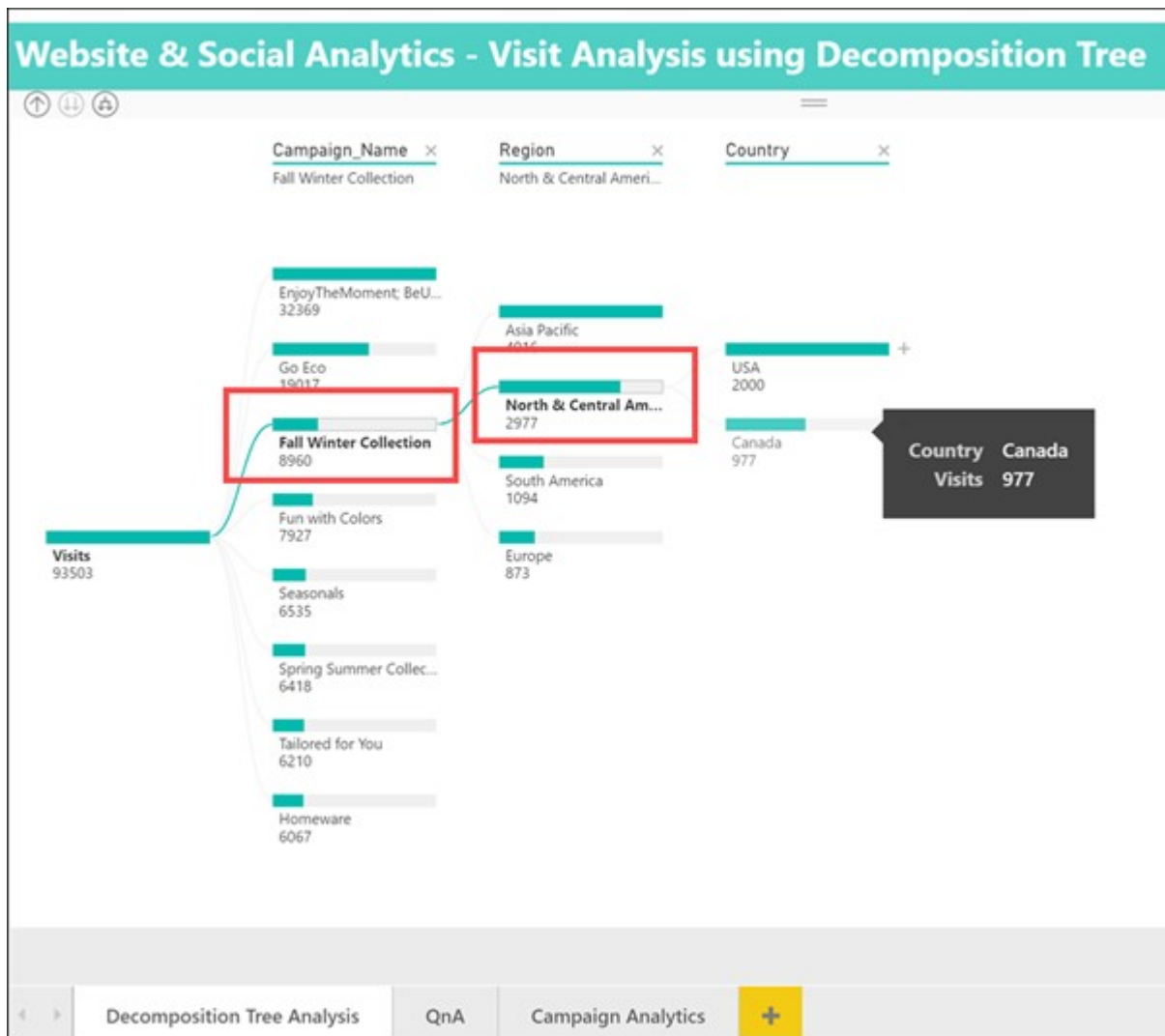
Manage

4. Expand Power BI, expand SynapseDemos, expand Power BI reports, then select **1-CDP Vision Demo (1)**. Select the arrows to collapse the **Visualizations** pane **(2)** and the **Fields** pane **(3)** to increase the report size.



As you can see, we can create, edit, and view Power BI reports from within Synapse Studio! As a business analyst, data engineer, or developer, you no longer need to open another browser window, sign in to Power BI, and toggle back and forth between environments.

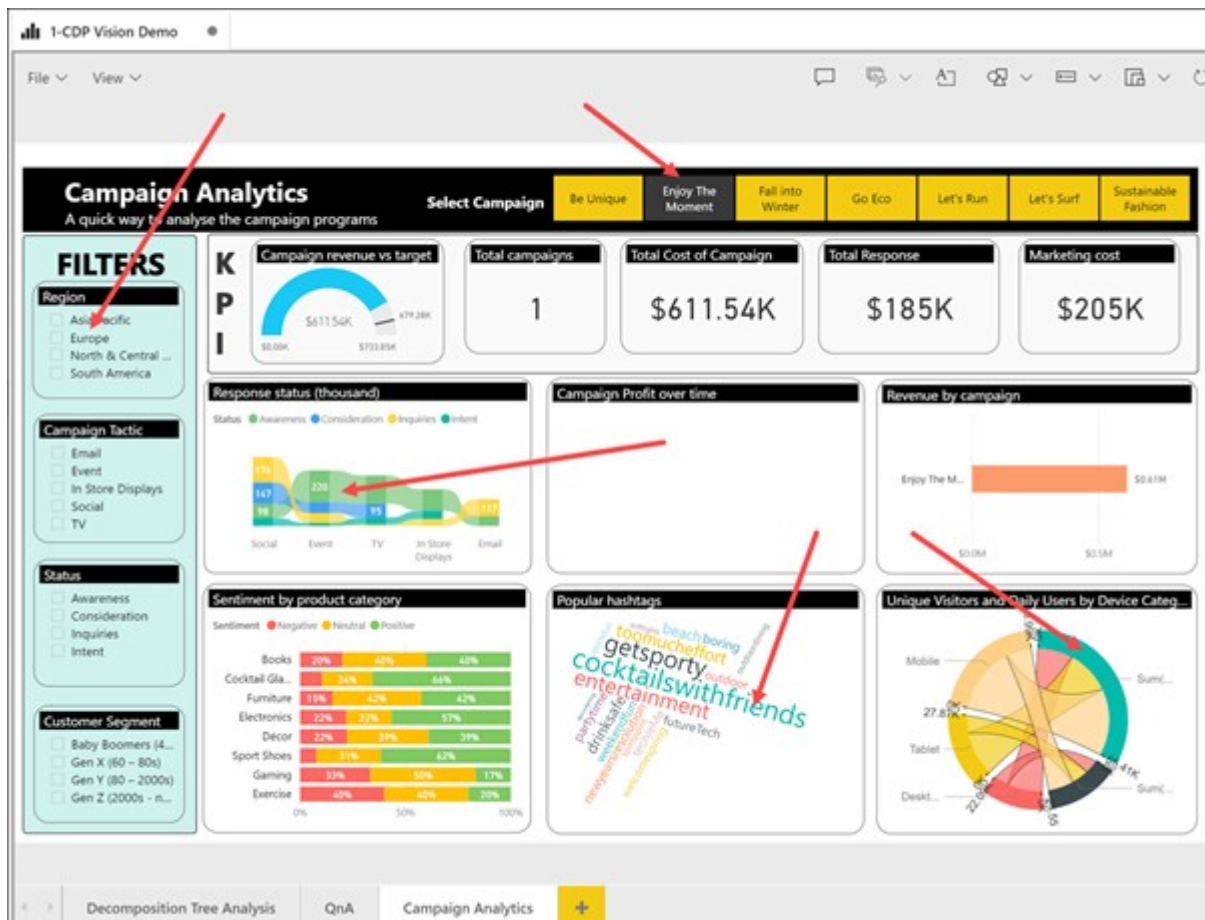
5. Select a **Campaign Name** and **Region** within the **Decomposition Tree Analysis** tab to explore the data. If you hover over an item, you will see a tool tip.



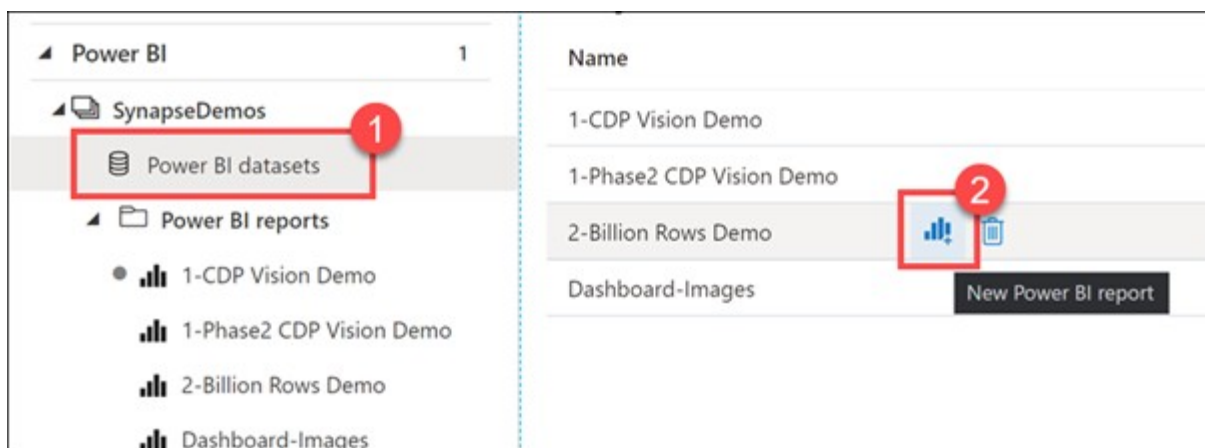
6. Select the **Campaign Analytics** tab at the bottom of the report.



The Campaign Analytics report combines data from the various data sources we looked at today to create a compelling visualization of valuable data within an interactive interface. You can select various filters, campaigns, and chart values to filter the results. Select an item to for the second time to deselect it.

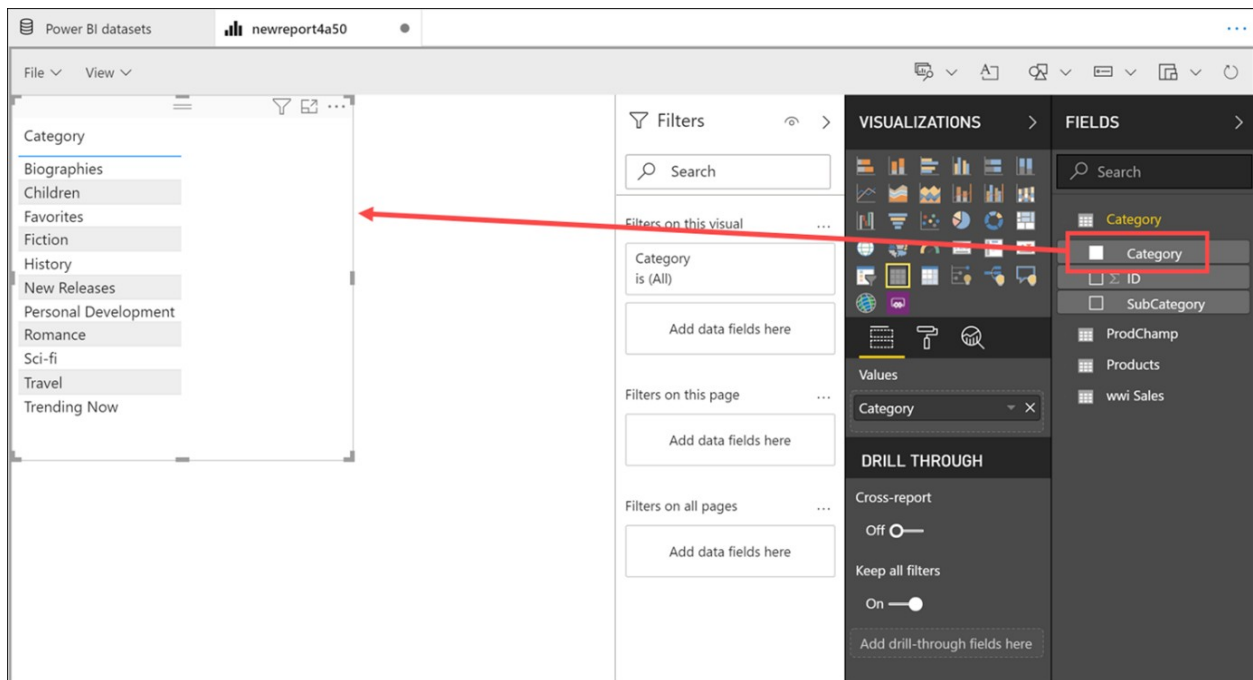


7. Select **Power BI datasets (1)** in the left-hand menu, hover over the **2-Billion Rows Demo** dataset and select the **New Power BI report** icon (2).

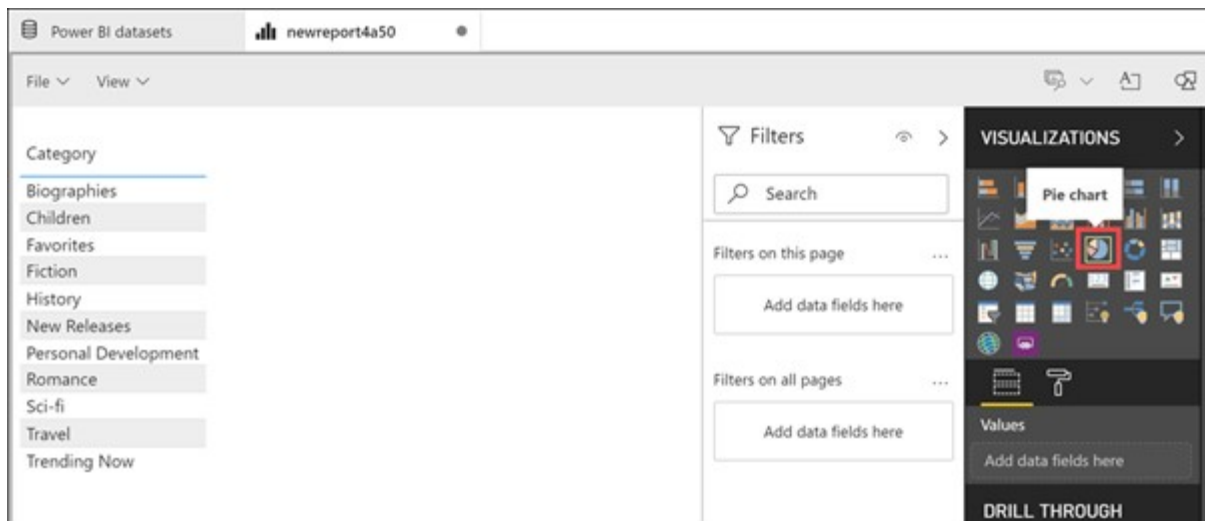


Here is how we can create a brand new Power BI report from a dataset that is part of the linked Power BI workspace, from within Synapse Studio.

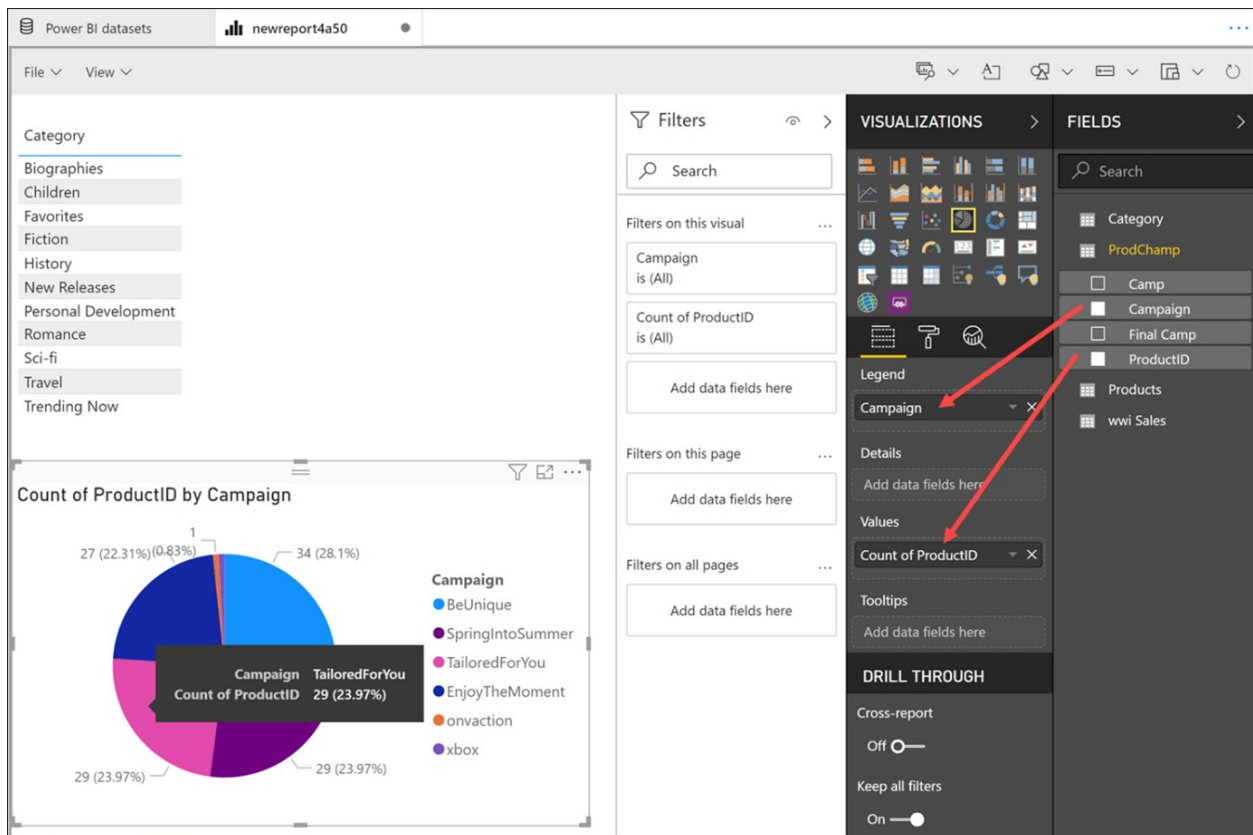
8. Expand the Category table, then **drag-and-drop** the **Category** field on to the report canvas. This creates a new Table visualization that shows the categories.



9. Select a blank area on the report canvas to deselect the table, then select the **Pie chart** visualization.



10. Expand the ProdChamp table. Drag **Campaign** onto the **Legend** field, then drag **ProductID** onto the **Values** field. Resize the pie chart and hover over the pie slices to see the tool tips.



We have very quickly created a new Power BI report, using data stored within our Synapse Analytics workspace, without ever leaving the studio. As you can see, the goal, and one of the primary strengths of Azure Synapse Analytics, is to help you build a modern data warehouse and have access to all of your data in one place