



# Pro Power BI Architecture

Development, Deployment, Sharing, and  
Security for Microsoft Power BI Solutions

—  
*Second Edition*

—  
Reza Rad

Apress®

# **Pro Power BI Architecture**

Development, Deployment,  
Sharing, and Security for  
Microsoft Power BI Solutions

Second Edition



**Reza Rad**

**Apress®**

# ***Pro Power BI Architecture: Development, Deployment, Sharing, and Security for Microsoft Power BI Solutions***

Reza Rad  
Auckland, New Zealand

ISBN-13 (pbk): 978-1-4842-9537-3  
<https://doi.org/10.1007/978-1-4842-9538-0>

ISBN-13 (electronic): 978-1-4842-9538-0

Copyright © 2023 by Reza Rad

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress Media LLC: Welmoed Spahr  
Acquisitions Editor: Jonathan Gennick  
Development Editor: Laura Berendson  
Editorial Project Manager: Shaul Elson  
Copy Editor: Kezia Endsley

Cover image by wal\_172619 from Pixabay

Distributed to the book trade worldwide by Springer Science+Business Media New York, 1 New York Plaza, Suite 4600, New York, NY 10004-1562, USA. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail [orders-ny@springer-sbm.com](mailto:orders-ny@springer-sbm.com), or visit [www.springeronline.com](http://www.springeronline.com). Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail [booktranslations@springernature.com](mailto:booktranslations@springernature.com); for reprint, paperback, or audio rights, please e-mail [bookpermissions@springernature.com](mailto:bookpermissions@springernature.com).

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at <https://www.apress.com/bulk-sales>.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at [www.apress.com/](http://www.apress.com/). For more detailed information, please visit <http://www.apress.com/source-code>.

Printed on acid-free paper

*To Leila*

*Who showed me how strong a human being can be*

# Table of Contents

<b>About the Author .....</b>	<b>xxix</b>
<b>About the Technical Reviewers .....</b>	<b>xxxii</b>
<b>Acknowledgments .....</b>	<b>xxxiii</b>
<b>Introduction .....</b>	<b>xxxv</b>
<b>■ Part I: Getting Started.....</b>	<b>1</b>
<b>■ Chapter 1: Power BI Components.....</b>	<b>3</b>
What Is Power BI? .....	4
The Power BI Desktop .....	4
Power Query for Data Transformation .....	5
The Power BI Report.....	6
Analytical Engine (Tabular Engine) .....	7
Datasets .....	8
The Power BI Dashboard .....	8
Dataflows .....	9
Datamarts.....	10
The DAX Language .....	11
The Power BI Service for Hosting Reports .....	12
The Power BI Mobile App .....	13
The Power BI Report Server .....	14
Paginated Reports .....	15
The Power BI Report Builder .....	16

■ TABLE OF CONTENTS

Metrics and Scorecards .....	17
The On-Premises Gateway .....	18
Workspaces.....	18
Power BI App .....	19
Power BI Premium.....	20
Power BI Embedded .....	20
Developer API or REST API.....	20
Summary.....	21
<b>Chapter 2: Tools and Preparation .....</b>	<b>23</b>
Using the Power BI Desktop .....	23
The Power BI Desktop: Report Authoring Tool .....	24
The Power BI Desktop from the Microsoft Store .....	24
The Power BI Desktop as a Separate Download Link.....	26
Auto-Update from the Microsoft Store App .....	26
Flexibility on Installation: The Download Link.....	27
Development Work.....	27
Setting Up Power BI Accounts .....	27
Creating Power BI Accounts from Office 365 .....	28
Is Your Account Pro, Free, or PPU?.....	29
Installing Power BI Mobile App.....	30
Power BI Report Server .....	30
On-Premises Gateway .....	30
Microsoft SQL Server Database and Management Tools .....	30
The Power BI Helper.....	30
Downloading Dataset Files.....	30
Summary.....	30

<b>Part II: Development.....</b>	<b>31</b>
<b>Chapter 3: Import Data or Scheduled Refresh.....</b>	<b>33</b>
The Connection Type Is Not the Data Source Type .....	33
Import Data or Scheduled Refresh .....	33
A Closer Look at Import Data.....	33
Loading Data to the Model .....	35
How Power BI Stores Data in Memory .....	36
How xVelocity Compresses Data.....	36
Where Data Is Stored in Memory .....	38
How About Power BI Service? .....	38
Is There a Limitation on Size? .....	39
Combining Data Sources; Power Query Unleashed.....	40
DAX: A Powerful Analytical Expression Language .....	43
Publishing a Report.....	44
Gateway Configuration .....	44
Scheduled Refresh .....	46
Advantages and Disadvantages of Import Data .....	49
Advantages of Import Data .....	49
Disadvantages of Import Data .....	49
When Should You Use Import Data? .....	50
Summary.....	50
<b>Chapter 4: DirectQuery .....</b>	<b>51</b>
What Is DirectQuery?.....	51
Which Data Sources Support DirectQuery?.....	51
How to Use DirectQuery.....	52
No Data Tab in DirectQuery Mode.....	54
How DirectQuery Works .....	55
Performance.....	58
Performance Tuning on the Data Source .....	58

## ■ TABLE OF CONTENTS

Query Reduction .....	60
Maximum Connections Per Data Source .....	62
Adding Extra Data Sources: Composite Mode .....	64
Limited Power Query .....	65
Limited Modeling and DAX .....	66
No Refresh Needed .....	66
Large Scale Dataset .....	66
Summary .....	67
<b>■ Chapter 5: Live Connection .....</b>	<b>69</b>
What Is Live Connection? .....	69
Create a Report with Live Connection .....	69
Live Connection Behind the Scenes .....	72
Performance .....	73
Single Data Source .....	74
No Power Query Transformations .....	74
Modeling and Report-Level Measure .....	76
Publishing the Report and Gateway Configuration .....	78
Live Connection to Power BI Service .....	83
How Live Connection Differs from DirectQuery .....	84
Summary .....	84
<b>■ Chapter 6: Composite Model .....</b>	<b>87</b>
What Is Composite Model? .....	87
Why Use Composite Model? .....	88
How Does Composite Model Work? .....	90
Dual Storage Mode .....	93
Modeling with the Power BI Composite Model .....	95
Relationships in the Power BI Composite Model .....	95
Calculations and DAX .....	96
DirectQuery to Power BI Datasets .....	96
Summary .....	97

<b>■ Chapter 7: DirectQuery for Power BI Datasets and Analysis Services .....</b>	<b>99</b>
Power BI with Import Data .....	99
One Important Problem with Import Data.....	99
Power BI Using Live Connection.....	100
Big Limitation of Live Connection.....	100
DirectQuery for Power BI Datasets or Analysis Services Datasets.....	101
Why Is This Feature a Big Deal? .....	101
How It Works Under the Hood.....	102
Not from My Workspace .....	102
Creating a DirectQuery to Power BI Dataset.....	103
DirectQuery to Power BI Dataset vs Live Connection to Power BI Dataset .....	106
Multiple DirectQuery Connections Are Supported .....	107
Creating a Relationship Between Different Data Sources .....	107
Publish to Service.....	108
Lineage View .....	113
Sharing the Report.....	114
Building Access to the Source Power BI Datasets.....	114
Summary.....	116
<b>■ Chapter 8: Choosing the Right Connection Type: DirectQuery, Live, Import, or Composite Model .....</b>	<b>117</b>
Why Is This a Tough Decision? .....	117
What Is Import Data (Scheduled Refresh)? .....	118
Compression Engine of xVelocity.....	118
Important Pros and Cons of This Method.....	118
What Is DirectQuery?.....	119
Important Pros and Cons of This Method.....	119
What Is Live Connection? .....	121
Important Pros and Cons of This Method.....	121
Composite Model: The Best of Both Worlds.....	122
Differences Between Live Connection and DirectQuery.....	123
Relationship Configuration .....	123

## ■ TABLE OF CONTENTS

Report-Level Measures .....	124
No Power Query in Live Connection .....	124
<b>Pros and Cons of Each Method .....</b>	<b>124</b>
Import Data or Scheduled Refresh .....	124
DirectQuery.....	124
Live Connection .....	124
Which Method Is the Best and Fastest? .....	125
Which Method Is More Flexible?.....	125
Which Method Is More Scalable? .....	125
<b>Which Architecture Scenarios Are Best for Each Method?.....</b>	<b>125</b>
Import Data for Agility and Performance .....	125
Live Connection for an Enterprise Solution .....	126
Live Connection for Team Development .....	126
Direct Query for Non-Microsoft Sources .....	126
Summary.....	128
<b>■ Chapter 9: Dataflows .....</b>	<b>129</b>
<b>What Is a Dataflow? .....</b>	<b>129</b>
A Dataflow Is Service-Only (Cloud-Only) Object.....	130
A Dataflow Is Not Just for Power BI.....	130
<b>Where Do Dataflows Store Data? .....</b>	<b>131</b>
Standard vs. Analytical Dataflows .....	132
<b>Dataflows Are Powered by Power Query Online.....</b>	<b>133</b>
<b>Dataflows Can Be Used in Power BI, Excel, and Other Services .....</b>	<b>135</b>
<b>Example Dataflow Scenarios.....</b>	<b>136</b>
Using One Power Query Table in Multiple Power BI Reports .....	136
Different Data Sources with Different Refresh Schedules.....	138
Centralized Data Warehouse.....	139
<b>Getting Started with Dataflows in Power BI .....</b>	<b>140</b>
<b>Administrator's Control .....</b>	<b>141</b>
<b>Creating Your First Dataflow .....</b>	<b>142</b>
Sample Datasets.....	145

Power Query Editor Online.....	146
Setting Up a Gateway .....	147
Blank Query as the Data Source.....	148
Scheduled Refreshes.....	150
Get Data from Dataflow in the Power BI Desktop.....	151
Get Data from Dataflows Using Excel .....	154
<b>Summary.....</b>	<b>155</b>
<b>■ Chapter 10: Shared Datasets.....</b>	<b>157</b>
What Is the Dataset in Power BI? .....	157
What Is Included in the Dataset?.....	159
What Is a Shared Dataset? .....	159
How Do You Create a Shared Dataset?.....	162
Sharing Datasets Across Multiple Workspaces .....	162
External Dataset: How Does Shared Dataset Work Behind the Scenes? .....	163
Why Use Shared Datasets? .....	163
Shared Datasets in the Power BI Architecture.....	164
Endorsement: Certified and Promoted Datasets.....	165
Summary.....	168
<b>■ Chapter 11: Datamarts .....</b>	<b>169</b>
Power BI for the Citizen Data Analyst.....	169
Governance and Reusability.....	169
What are Datamarts in Power BI? .....	170
Who Are Datamarts for? .....	172
Power BI Datamart for the Citizen Data Analyst .....	173
Power BI Datamart for Enterprises .....	173
Power BI Datamart for Developers .....	173
What Are the Features of Datamarts that Empower Power BI Development?.....	173
Datamarts Complete the BI Ecosystem .....	173
One Place to Manage and Build, No Other Tools Needed.....	174
One License Is Enough .....	175

## ■ TABLE OF CONTENTS

The Vast Horizon and the Future of Datamarts .....	175
Governance.....	175
<b>Getting Started with Power BI Datamarts .....</b>	<b>175</b>
Premium Workspace.....	175
Creating a Datamart .....	177
Get Data: Power Query Online .....	179
The Datamart Editor.....	182
<b>Power BI Datamart Components .....</b>	<b>190</b>
Power BI Datamarts Under the Hood .....	190
<b>Summary.....</b>	<b>202</b>
<b>■ Chapter 12: The Multi-Layer Architecture .....</b>	<b>203</b>
Challenges of Using a Single PBIX File for Everything .....	203
Using Dataflows to Decouple the Data Preparation Layer.....	204
Shared Datasets for the Data Modeling Layer.....	206
Thin Power BI Reports, Paginated Reports, and the Analyze in Excel Option.....	206
Power BI Architecture with Dataflows, Shared Datasets, and Thin Reports.....	208
Benefits of a Multi-Layered Architecture.....	209
Enhancing the Architecture Using Power BI Datamarts.....	210
Layered Architecture for Dataflow .....	211
Chained Datasets.....	214
What About Enterprise Architecture?.....	215
Summary.....	216
<b>■ Chapter 13: Datamarts vs. Dataflows vs. Datasets .....</b>	<b>217</b>
What Is a Dataflow? .....	217
What Is a Dataset? .....	218
What Is a Datamart? .....	218
Differences Between Dataflows and Datasets .....	219
Dataflows vs. Datasets .....	224

What About Datamarts? .....	225
Will Dataflows Be Replaced by Datamarts?.....	226
Are Datasets Being Replaced by Datamarts? .....	227
Summary.....	228
<b>■ Chapter 14: Paginated Reports.....</b>	<b>229</b>
What Is a Paginated Report in Power BI?.....	229
Why Paginated Reports? .....	231
Differences Between Power BI Reports and Paginated Reports .....	233
Do You Need a Paginated Report?.....	234
Architecture Best Practice: One Dataset to Rule Them All .....	235
The Power BI Report Builder .....	235
Summary.....	240
<b>■ Chapter 15: Excel and Power BI Integration.....</b>	<b>241</b>
The Analyze in Excel Feature.....	241
The Analyze in Excel Feature from the Power BI Service .....	241
Implicit Measures Don't Work in Excel .....	244
Excel Is Connected Live to the Power BI Model in the Service.....	247
Getting Data from Power BI Dataset.....	248
Getting Data from a Power BI Dataflow .....	251
Why is Analyze in Excel Better than Using Export Data? .....	252
Publish to Power BI from Excel .....	254
Upload Your Workbook to Power BI .....	255
Export Workbook Data to Power BI .....	256
Import Excel into the Power BI Desktop.....	256
Summary.....	257
<b>■ Chapter 16: Real-Time Streaming Datasets in Power BI .....</b>	<b>259</b>
Real-time vs. Live or DirectQuery.....	259
Sample Scenario .....	260
Capturing Inputs with a Microsoft Form.....	260

## ■ TABLE OF CONTENTS

<b>The Power BI Streaming Dataset .....</b>	<b>261</b>
<b>Power Automate Pushes Data to the Power BI Dataset .....</b>	<b>265</b>
<b>The Power BI Real-Time Dashboard.....</b>	<b>268</b>
<b>Streaming Dataset Types.....</b>	<b>269</b>
Push Datasets.....	269
Streaming Datasets .....	269
Hybrid Datasets .....	269
<b>Streaming Services .....</b>	<b>270</b>
Power BI Streaming Dataset and REST API .....	270
Azure Stream Analytics.....	270
PubNub .....	270
<b>Creating Calculations on Real-Time Datasets .....</b>	<b>270</b>
Writing Calculations Using Q&A.....	270
Live Connection to a Streaming Dataset .....	274
Summary.....	278
<b>■ Chapter 17: Performance Tuning with Aggregations.....</b>	<b>279</b>
<b>How Does Aggregation Work? .....</b>	<b>280</b>
Aggregated Table .....	281
Layers of Aggregation.....	282
Step 1: Create the Aggregated Table .....	283
Dual Storage Mode; The Most Important Configuration for Aggregations! .....	293
Step 2: Power BI Aggregations .....	293
Step 3: Configure Aggregation Functions and Test Aggregations in Action .....	310
Multiple Layers of Aggregations .....	318
<b>Aggregation for Imported Data.....</b>	<b>328</b>
DAX Measures Instead of Manage Aggregation.....	331
<b>Automatic Aggregation.....</b>	<b>336</b>
Summary.....	337

<b>■ Chapter 18: Big Data with Incremental Refresh and Hybrid Tables.....</b>	<b>339</b>
What Is Incremental Refresh? .....	339
Hybrid Tables.....	339
Configuring Incremental Refresh .....	340
Setting Up Incremental Refresh .....	340
Parameters in Power Query.....	340
Filter Data Based on the Parameters.....	342
Power BI Desktop Incremental Refresh Setup.....	344
Hybrid Table Setup.....	347
Incremental Refresh for Multiple Tables.....	348
Publish to Service.....	349
What Do the Partitions Look Like? .....	350
Detecting Data Changes.....	354
Only Refresh When Period Is Complete .....	355
Incremental Refresh for Dataflows or Datamarts.....	356
Summary.....	357
<b>■ Chapter 19: Development Best Practices .....</b>	<b>359</b>
Reuse Tables Generated in Power Query .....	359
Reuse DAX Calculations .....	359
Use Power Query Functions for Reusable Data Transformation.....	360
Parametrize the Data Transformation Process Using Power Query Parameters .....	360
Categorize Power Query Using Power Query Groups (or Folders).....	360
Disable the Load of Temp Tables in Power Query.....	361
Only Load What You Need for Reporting: Filter the Data .....	361
Use Reference and Duplicate in Their Rightful Places .....	361
Multi-Layer Architecture Everywhere.....	361
Sync Slicers.....	362
Use a Theme File .....	362
Use a Background Image for Report Pages.....	362

## ■ TABLE OF CONTENTS

Incorporate Conditional Formatting Using DAX, Parameters, and Tables .....	362
Create a Measure Table.....	363
Create and Use Hierarchies.....	363
Set Auto Summarization at the Field Level .....	363
Consider Using a Custom Date Table.....	363
Design Mobile Reports .....	364
Use Aggregations for Big Tables.....	364
Use Incremental Refresh and Hybrid Tables When Possible .....	364
Design Star-Schema Models .....	365
Avoid Both-Directional Relationships .....	365
Clean Up Your Models.....	365
<b>■ Part III: Deployment and Collaboration.....</b>	<b>367</b>
<b>■ Chapter 20: The Power BI Service .....</b>	<b>369</b>
The Power BI Desktop: A Report Authoring Tool .....	369
Hosting Options for Power BI Reports .....	370
What Is the Power BI Service? .....	371
Publish Reports to the Service .....	372
Capacity-Based or User-Based.....	373
The Power BI Service .....	374
Template Apps .....	375
Apps.....	375
Workspaces .....	376
Dataflows.....	377
Shared Datasets .....	378
Datamarts .....	378
Deployment Pipelines.....	379
Dashboards.....	380
Metrics.....	380

Content Certification.....	381
Data Lineage.....	382
Summary.....	382
<b>■ Chapter 21: The Power BI Report Server.....</b>	<b>383</b>
What Is the Power BI Report Server? .....	383
Requirements for Setting It Up .....	384
Installing the Power BI Report Server .....	385
Configuring the Power BI Report Server .....	390
Database Set Up .....	392
Web URL Set Up.....	396
Installing the Power BI Desktop Report Server .....	402
Developing Reports with the Power BI Report Server.....	403
Publish Report to the Report Server .....	405
Managing Datasets on the Report Server .....	407
Schedule Refresh Requirement.....	409
Pros and Cons of the Report Server .....	412
No Gateway Is Needed.....	412
All Types of Connections Are Supported .....	413
The Power BI Report Server Is a Fully On-Premises Solution.....	413
The Power BI Service Features Are Not Available.....	413
Licensing the Report Server .....	413
Summary.....	414
<b>■ Chapter 22: Power BI Gateways .....</b>	<b>415</b>
What Is a Gateway?.....	415
Do You Always Need a Gateway?.....	415
Types of Gateways.....	415
Gateway Execution Flow.....	417
Important Points to Consider Before Installing a Gateway .....	418
How Many Gateways Are Required?.....	418
Installing Gateways .....	419

■ TABLE OF CONTENTS

<b>Adding Data Sources .....</b>	<b>424</b>
Set Up a Gateway Connection to the Dataset .....	428
<b>Users and Access Controls for Gateways .....</b>	<b>429</b>
Gateway Installers .....	429
Gateway Users.....	430
Data Source Users .....	431
Additional Settings for Gateway .....	432
<b>Troubleshooting.....</b>	<b>435</b>
<b>Summary.....</b>	<b>436</b>
<b>■ Chapter 23: Power BI Licensing .....</b>	<b>437</b>
<b>Two Types of Licensing.....</b>	<b>437</b>
The Power BI Desktop .....	438
Why Do I Need a License if the Power BI Desktop Is Free?.....	438
<b>User-Based Licensing.....</b>	<b>439</b>
Power BI Free .....	439
Power BI Pro .....	439
Power BI PPU: Premium Per User .....	441
<b>Capacity-Based Licensing.....</b>	<b>442</b>
Power BI Embedded .....	442
Power BI Premium.....	444
The Difference Between A SKUs and P/EM SKUs .....	446
<b>SQL Server Enterprise Edition Plus Software Assurance .....</b>	<b>447</b>
<b>Summary.....</b>	<b>449</b>
<b>■ Chapter 24: Power BI Admin Portal and Tenant Settings .....</b>	<b>451</b>
<b>Power BI Administrator .....</b>	<b>451</b>
<b>Tenant Settings .....</b>	<b>452</b>
Help and Support Settings.....	453
Workspace Settings.....	455
Information Protection .....	455
Export and Sharing Settings.....	456

Content Pack and App Settings .....	459
Integration Settings .....	460
Custom Visual Settings .....	462
Audit and Usage Settings .....	463
Developer Settings .....	464
Dataflow, Template Apps, and Datamarts .....	464
<b>Usage Metrics .....</b>	<b>465</b>
Embed Codes .....	465
Organization Visuals .....	466
Workspaces .....	468
Custom Branding .....	468
Summary .....	469
<b>■ Chapter 25: PowerShell Cmdlets for Power BI .....</b>	<b>471</b>
PowerShell .....	471
PowerShell Cmdlets for Power BI .....	471
Getting Started .....	472
Example: Export a List of All Workspaces in the Organization as a CSV File .....	478
Example: Deploy a Power BI Report from One Workspace to Another .....	479
Are PowerShell Cmdlets the Same as REST APIs in Power BI? .....	483
Summary .....	483
<b>■ Chapter 26: Power BI REST API .....</b>	<b>485</b>
The Power BI REST API .....	485
Getting Started .....	485
Calling the API from .NET .....	490
Embedding into a Custom Application .....	492
Power Automate .....	493
PowerShell to Call REST API .....	493
Summary .....	493

■ TABLE OF CONTENTS

<b>■ Chapter 27: Power BI Audit Log for the Tenant .....</b>	<b>495</b>
The Problem .....	495
The Audit Log .....	498
Using PowerShell to Extract the Audit Log .....	502
The Power BI Report for the Audit Log .....	505
Summary .....	512
<b>■ Chapter 28: XMLA Endpoint .....</b>	<b>513</b>
Behind the Scenes of a Power BI Dataset.....	513
SSAS Is More Than What You See .....	514
Can I Access the SSAS Model of My Power BI Dataset? .....	516
XMLA Endpoint .....	517
How Does an XMLA Connection Work?.....	518
The XMLA Endpoint URL .....	519
Admin Control under Capacity Configuration.....	519
Client Tools .....	520
Sample Connection.....	521
The Power BI Helper .....	522
Summary .....	522
<b>■ Chapter 29: Dashboard and Report Sharing .....</b>	<b>523</b>
Power BI Content Owner .....	523
How Does Dashboard Sharing Work?.....	523
Three Levels of Access.....	528
Manage Permissions .....	528
Advantages of Dashboard Sharing .....	532
Disadvantages of Dashboard Sharing .....	532
No Edit Access.....	533
Share Objects One at a Time .....	533
Licensing for a Large User Base.....	533
Summary .....	533

<b>■ Chapter 30: Power BI Workspaces .....</b>	<b>535</b>
What Are Power BI Workspaces? .....	535
Creating Workspaces.....	536
Adding Content to a Workspace .....	541
Four Levels of Workspace Access .....	543
Tenant Admin Control for Workspaces .....	544
Advantages of Workspaces .....	547
Disadvantages of Workspaces .....	547
Summary.....	548
<b>■ Chapter 31: Power BI Apps.....</b>	<b>549</b>
Power BI Apps .....	549
App Workspaces .....	549
Creating an App .....	551
Setup .....	552
Content .....	554
Audience.....	556
Getting Apps .....	559
Making Changes to an App.....	561
Isolated Environments for Developers and Users.....	562
What About Data Refresh?.....	564
Advantages of Power BI Apps .....	564
Cons of Power BI Apps .....	565
Summary.....	565
<b>■ Chapter 32: Publish to Web .....</b>	<b>567</b>
What Is Publish to Web?.....	567
Security Issues with Publish to Web .....	570
Removing Access .....	572

■ TABLE OF CONTENTS

<b>Central Monitoring for all Embed Codes.....</b>	<b>574</b>
Who Can Publish Reports on the Web?.....	574
<b>Differences with Other Sharing Methods .....</b>	<b>575</b>
Summary.....	578
<b>■ Chapter 33: Power BI Embedded .....</b>	<b>579</b>
<b>Using Power BI Embedded .....</b>	<b>579</b>
Myths about Power BI Embedded.....	580
<b>What Is Power BI Embedded? .....</b>	<b>580</b>
Embedded Playground.....	580
<b>Licensing Power BI Embedded.....</b>	<b>582</b>
Some SKUs Can Be Turned On and Off .....	583
Power BI Users or Custom Application Users .....	583
<b>Implementing Power BI Embedded .....</b>	<b>584</b>
Summary.....	585
<b>■ Chapter 34: Secure Embed .....</b>	<b>587</b>
<b>Why Another Embedding Method? .....</b>	<b>587</b>
Publish to Web.....	587
Power BI Embedded .....	588
Secure Embed: Good Features from Both Worlds!.....	589
<b>Using Secure Embed .....</b>	<b>589</b>
Who Can View the Report? .....	590
Licensing Needs .....	593
<b>Advantages of Secure Embed .....</b>	<b>593</b>
<b>Limitations.....</b>	<b>594</b>
<b>Scenarios for Using this Method .....</b>	<b>594</b>
Summary.....	595

<b>Chapter 35: Embed in SharePoint Online and Teams.....</b>	<b>597</b>
How to Use Embed in SharePoint Online.....	597
Sharing a SharePoint Page with an Embedded Power BI Report.....	604
Access to SharePoint.....	604
Power BI Permissions.....	605
Advantages and Disadvantages of Embedding in SharePoint Online.....	608
Advantages of Embed in SharePoint Online .....	608
Disadvantages of Embed in SharePoint Online.....	609
Power BI and Teams.....	609
Power BI and PowerPoint.....	610
Summary.....	611
<b>Chapter 36: Comparison of Sharing Methods for Power BI.....</b>	<b>613</b>
Types of Sharing Methods .....	613
Basic Sharing for Dashboards and Reports.....	614
Advantages of Dashboard Sharing .....	615
Disadvantages of Dashboard Sharing.....	615
Workspaces.....	615
Advantages of Workspaces.....	616
Disadvantages of Workspaces.....	617
Power BI Apps .....	617
Advantages of Power BI Apps.....	618
Cons of Power BI Apps.....	619
The Publish to Web Option .....	619
Security Issues with Publish to Web.....	620
Embed in SharePoint Online.....	620
Advantages of Embed in SharePoint Online .....	621
Disadvantages of Embed in SharePoint Online.....	622
Power BI Embedded .....	622
Pros and Cons of Power BI Embedded .....	622

■ TABLE OF CONTENTS

<b>Secure Embed .....</b>	<b>623</b>
Advantages of Secure Embed.....	623
Disadvantages and Limitations of Secure Embed .....	624
<b>Summary.....</b>	<b>624</b>
Cheat Sheet for Choosing a Method .....	624
<b>■ Chapter 37: Types of Power BI Users.....</b>	<b>625</b>
The Different Types of Power BI Users .....	625
Access Control .....	626
Training.....	627
The Layers and User Categories Are Different in Each Organization.....	628
Users Can Move Between Layers .....	629
Summary.....	629
<b>■ Chapter 38: Best Practices for Setting Up Workspace Roles.....</b>	<b>631</b>
The Modern Power BI Workspace .....	631
Roles in the Workspace .....	632
Viewer.....	632
Contributor.....	633
Member .....	634
Admin .....	637
Best Practices for Each Role .....	638
Viewer Role: Use Power BI Apps Instead .....	638
Contributor Role: Developers Only .....	639
Member Role: Deployment Group Only .....	639
Admin Role: Admin Group .....	640
Use Groups, Not Individual Accounts .....	640
Summary.....	640

<b>■ Chapter 39: Build Access Level .....</b>	<b>641</b>
Build Access Example .....	641
Are Build and Edit the Same? A Row-Level Security Example .....	643
Providing Build But Not Edit Access .....	645
Manage Permissions: Add Build Access .....	645
Create an App with Build Access .....	646
Who Should Have Build Access? .....	648
Licensing .....	648
Summary .....	649
<b>■ Chapter 40: How to Organize Workspaces in a Power BI Environment.....</b>	<b>651</b>
Considering a Power BI Workspace as a Single Development-Sharing Unit.....	651
Separating Audiences Using Power BI Apps .....	651
Separating Developers with Multiple Workspaces .....	652
Split the Load, Use the Capacity.....	654
Sharing Workspaces Among Multiple Developers .....	654
Layers of Shared Workspaces .....	655
Separating the Environments .....	656
Summary .....	660
<b>■ Chapter 41: Content Certification and Endorsement .....</b>	<b>661</b>
The Importance of Endorsement .....	661
Levels of Endorsement .....	661
What Kinds of Content Can Be Endorsed? .....	663
Who Can Endorse It? .....	663
Dataset Discoverability .....	665
An Example of an Endorsement and Content Certification .....	666
Summary .....	668

■ TABLE OF CONTENTS

<b>■ Chapter 42: Deployment Pipelines .....</b>	<b>669</b>
Why Multiple Environments?.....	669
How Many Environments Do You Need?.....	672
What Is a Deployment Pipeline?.....	673
How Does a Deployment Pipeline Work?.....	674
Creating Deployment Pipelines.....	674
Assigning a Workspace.....	675
Comparing Content.....	676
Deploy.....	678
Rules and Connections .....	680
Automate the Deployment.....	680
What If There Is No Premium License? .....	681
Summary.....	681
<b>■ Chapter 43: Data-Level Security.....</b>	<b>683</b>
Introduction .....	683
Row-Level Security .....	683
Static Row-Level Security .....	684
Dynamic Row-Level Security.....	693
Column-Level Security .....	703
Object-Level Security .....	704
Page-Level Security.....	704
Summary.....	704
<b>■ Chapter 44: Power BI Helper .....</b>	<b>705</b>
Documenting a Power BI File and Report.....	705
Download and Install Power BI Helper for Free .....	705
Open Power BI Helper as an External Tool.....	705
Connect to Model.....	706
Visualization Information .....	707
Documentation .....	708

What Is Included in the Documentation? .....	710
Configurations .....	710
Exporting the Data in a Power BI Table .....	712
Connect to Model.....	712
Export the Data.....	713
Any Size, Any Table Type.....	716
Consider Using Analyze in Excel.....	717
Reducing the Size of Power BI File .....	717
Steps to Reduce File Size .....	718
Documenting Power BI Tenant Objects .....	722
Service Tenant Settings in Power BI Helper .....	723
Scan Service Objects.....	730
Configuring the Documentation's Output.....	732
Exporting the Power BI Audit Log .....	735
What Is the Power BI Audit Log?.....	735
Difficulties Exporting the Audit Log .....	735
A Simple Way to Export the Audit Log Without Limitations.....	736
Power BI File Cleanup in a Few Steps.....	739
Visualization Information.....	740
Hiding Technical Fields .....	743
Summary.....	745
<b>Index.....</b>	<b>747</b>

## CHAPTER 16



# Real-Time Streaming Datasets in Power BI

Power BI datasets can use the Import Data, DirectQuery, or Live Connection connection types. However, there is also one specific type of dataset that is different, called a *streaming dataset*. A streaming dataset is used with a real-time dashboard and comes with various setups and configurations. This chapter talks about this type of dataset.

## Real-time vs. Live or DirectQuery

The first question I get most often when I explain the real-time dataset is, “What is the difference between real-time, Live, and DirectQuery?” The common belief is that a Live Connection is real-time. This is not true. One of the best ways of understanding the differences is as follows:

- In a Live Connection or DirectQuery connection report or dashboard, when you refresh the visual (which can be done either by refreshing the report itself or by clicking a visual or slicer that triggers a query from the data source), you get the most up-to-date data. The report or dashboard can also be scheduled to refresh automatically so that you have the most up-to-date data on scheduled interval.
- A real-time dashboard will update automatically as soon as the new data row appears in the dataset. This new data row can come from various methods, such as pushing through a REST API of Power BI, stream analytics, or other streaming services such as PubNub. A real-time dashboard does not need a scheduled refresh.

One big difference you can spot in these types of connections is that a real-time dataset doesn’t wait for a refresh, it pushes the data directly to the dashboard, and the dashboard shows that change immediately. In a Live or DirectQuery connection, you get the most up-to-date data whenever you ask for it, and that is by refreshing the report. (Although, even if you don’t refresh the report, perhaps you are looking at the data that was refreshed less than an hour ago.)

A real-time dataset is mainly used in scenarios where the change has to be visible as soon as it happens. Let’s say you have a temperature sensor in a room. You want to have a Power BI dashboard that shows the temperature changes in a line chart as soon as they change. A real-time streaming dataset is necessary here.

Live or DirectQuery connections are used when you want up-to-date data. If you do not have an IoT device that sends the data as soon as it appears, and your purpose is to see the most up-to-date data in your report and dashboard whenever you look at it, a Live or DirectQuery connection is best. (Of course, there are differences between the Live Connection and DirectQuery, which I explained in previous chapters.)

Many businesses will ask for a real-time dashboard, but they really need a Live or DirectQuery connection. Use the definitions and the differences described here as your reference when making this decision. As a Power BI architect, you should understand the requirements, read between the lines, and come up with the best solution for the business needs.

In the rest of this chapter, you learn how to set up and use a real-time streaming dataset in Power BI.

## Sample Scenario

For the real-time streaming scenario of this chapter, I explain this procedure and how to implement it:

- A Microsoft form is shared with users to get feedback on their Power BI roles and salaries.
- Power Automate is used to pass the feedback posted in the form (as soon as it arrives) to the Power BI streaming dataset.
- The Power BI streaming dataset receives the feedback and presents it in real-time on a dashboard.

This scenario might not be the most common, but it is one of the simplest to understand because it doesn't require much coding, and the components can be easily accessed without needing a special license for those tools and services.

## Capturing Inputs with a Microsoft Form

I used a Microsoft form to capture the user's input (see Figure 16-1).

**2021 Power BI Salary Survey**

This survey is to help you to understand your salary vs what others getting paid in your profession. Rules to fill the salary survey:  
Only fill it once per person  
No sensitive information is required (anonymous)  
No emails or company information is required  
The result will be visible through a Power BI report here:

<https://app.powerbi.com/view?r=eyJrIjoiNzE2L2ZlQzNzNCNzNzY0IyODJMTIBNzNyODVlZWdCNzkyNzY0DzrOGQ2LTNmMzUngzNzIwMzY4LWszZWEKQzg4NzS3S3mMzQzrC5%3D&embed=true&hostId=true&pageName=Blank>

\* Required

1. Your Salary (In local currency, annual, before taxes) \*

Enter your answer

2. Your Country \*

Enter your answer

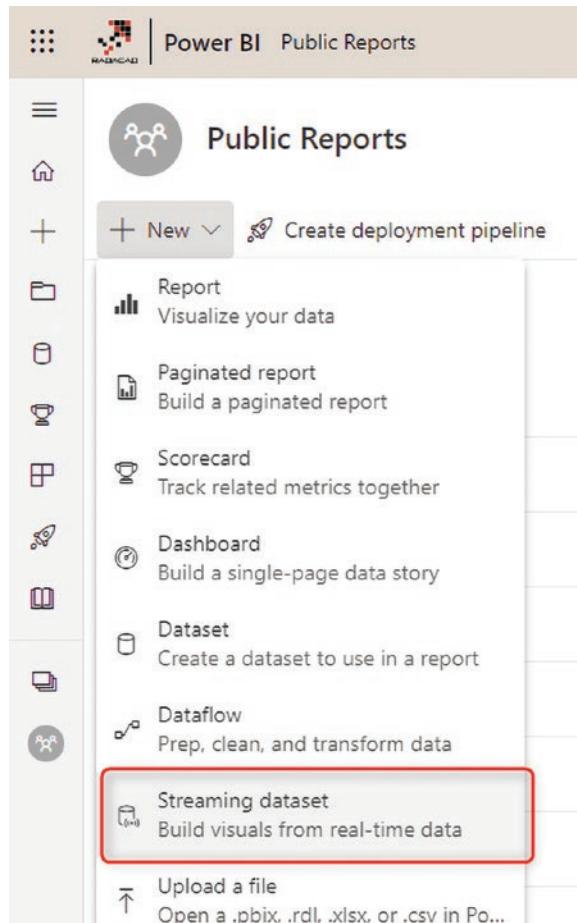
3. What's your role? \*

**Figure 16-1.** Microsoft form for a salary survey

The form shown in Figure 16-1 is used for this fictitious Power BI 2021 Salary Survey<sup>1</sup>. You can create a form like this in Microsoft Forms by going to [forms.office.com/](https://forms.office.com/).<sup>2</sup>

## The Power BI Streaming Dataset

To capture the form's data and send it to Power BI, you need to create a streaming dataset with the same set of fields. A streaming dataset cannot be created in the Power BI Desktop. You have to log in to the Power BI Service and create the streaming dataset from there, as shown in Figure 16-2.

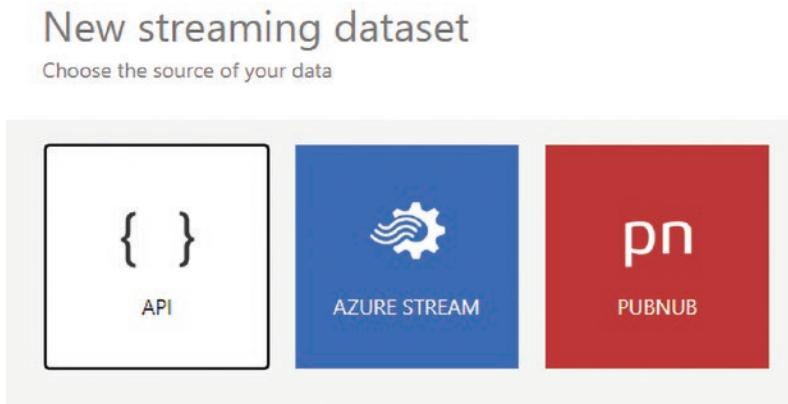


**Figure 16-2.** Creating a streaming dataset in Power BI

<sup>1</sup>[radacad.com/power-bi-salary-survey-and-report-2021](https://radacad.com/power-bi-salary-survey-and-report-2021)

<sup>2</sup>[forms.office.com/](https://forms.office.com/)

There are three ways to create a streaming dataset in Power BI, as illustrated in Figure 16-3.



**Figure 16-3.** Power BI streaming dataset options

It is important to understand how the Power BI streaming dataset works behind the scenes, which I explain later in this chapter. For now, consider these three methods as three different data streaming service providers—Azure Stream Analytics, PubNub, and Power BI Rest API. This example uses the Power BI Rest API.

In the next step, you need to name this streaming dataset and set up the fields and their data types; see Figure 16-4.

## Edit streaming dataset

Create a streaming dataset and integrate our API into your device or application to send data. [Learn more about the API.](#)

\* Required

Dataset name \*

Values from stream \*

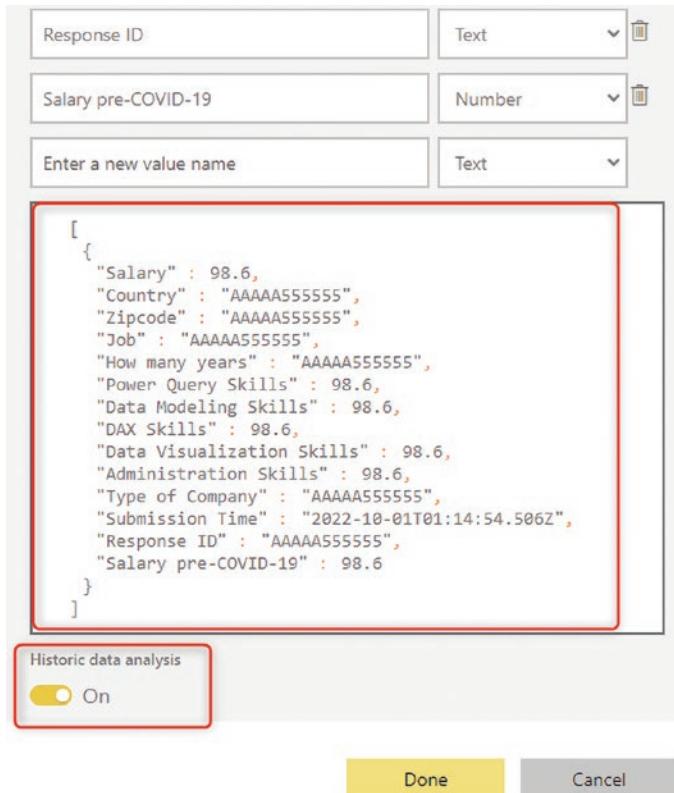
Salary	Number	✖
Country	Text	✖
Zipcode	Text	✖
Job	Text	✖
How many years	Text	✖
Power Query Skills	Number	✖
Data Modeling Skills	Number	✖
DAX Skills	Number	✖
Data Visualization Skills	Number	✖
Administration Skills	Number	✖

**Done** **Cancel**

**Figure 16-4.** Setting up the streaming dataset in Power BI

The data type for fields is a limited set of Text, Number, or Datetime. The streaming dataset is different from a normal Power BI dataset.

Once you create the fields, you can see a sample record generated underneath them in JSON format (see Figure 16-5 for an example). This is the format that the record data will be sent to the dataset. You can also see a Historical Data Analysis option. Turn this on. This means that data is not only streamed through this dataset but is also stored. I explain this option later.



**Figure 16-5.** Historical data analysis in a Power BI streaming dataset

When the dataset is created and ready, the Push API URL will appear. You can choose the format for sending data to it as a sample, as shown in Figure 16-6.

## API info on Salary Survey 2021

Use the API endpoint URL and one of the examples shown below to send data to your streaming dataset. For more information, [read our API documentation and integration guide](#).

The screenshot shows the 'Push URL' configuration for a Power BI dataset. A red box highlights the URL input field, which contains `https://api.powerbi.com/beta/` followed by a dropdown menu showing `/data`. Below this, there are three tabs: 'Raw' (selected), 'CURL', and 'PowerShell'. A large red box encloses the JSON data sample:

```
[{"Salary": 98.6, "Country": "AAAAA55555", "Zipcode": "AAAAA55555", "Job": "AAAAA55555", "How many years": "AAAAA55555", "Power Query Skills": 98.6, "Data Modeling Skills": 98.6, "DAX Skills": 98.6, "Data Visualization Skills": 98.6, "Administration Skills": 98.6, "Type of Company": "AAAAA55555", "Submission Time": "2022-10-01T01:19:58.513Z", "Response ID": "AAAAA55555", "Salary pre-COVID-19": 98.6}]
```

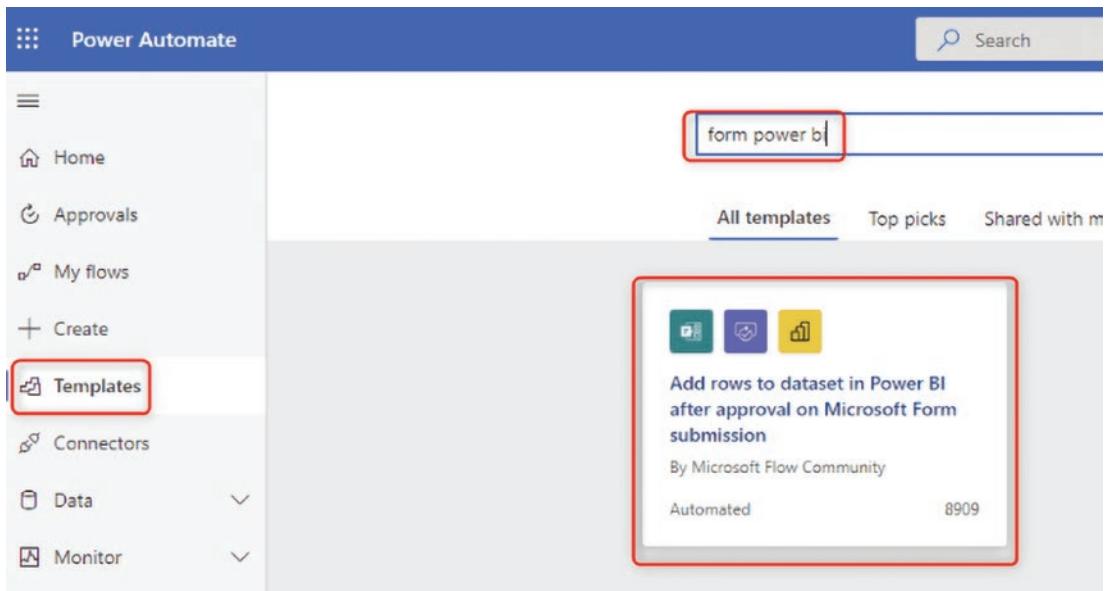
**Figure 16-6.** Pushing URL details for Power BI Dataset

The Push URL is the API URL that other applications should use to pass data rows (or push them) into this dataset. This example uses Power Automate to push data to this dataset.

## Power Automate Pushes Data to the Power BI Dataset

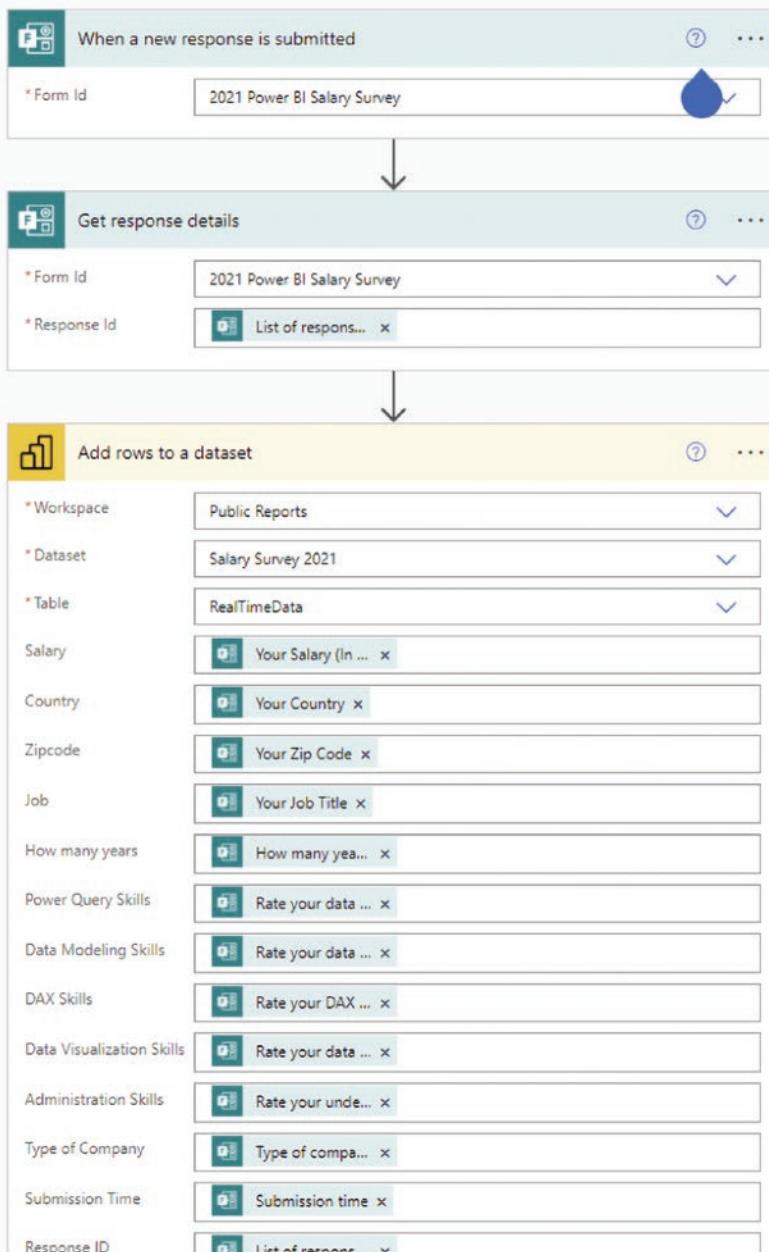
You can use many applications to push data into the streaming dataset, such as a C# application, PowerShell scripts, and so on. However, for simplicity, this example uses Power Automate, which you can access from [make.powerautomate.com/](https://make.powerautomate.com/). Power Automate is a service in the Power Platform toolset for If-Then-Else flow definitions. Here, you use it to set up a flow. If there is an entry in the form, then Power Automate will push that record to the streaming dataset in Power BI.

You can use the template in Power Automate (see Figure 16-7), which has the process you want. You just need to enter your credentials and environment details in it.



**Figure 16-7.** Using a template in Power Automate

The flow definition is simple. As shown in Figure 16-8, it has three steps: connect to the Microsoft form when a new response is submitted, read the response, and push it to the Power BI dataset.



**Figure 16-8.** Pushing data to the Power BI streaming dataset using Power Automate

As you can see, the Add Rows to Power BI Dataset component generates the data row and pushes it to the streaming dataset in Power BI. This flow will run automatically as soon as a Microsoft form entry is entered (this is set by the When a New Response Is Submitted trigger).

## The Power BI Real-Time Dashboard

The last step is creating a Power BI Dashboard to show the real-time changes. This dashboard is similar to the other dashboards in Power BI. The only difference is that it is sourced from the streaming dataset. The dashboard can be created from a report, or it can be created as a blank, and then you can add a tile connecting to a streaming data source, as shown in Figure 16-9. Then you can select the streaming dataset for it.



**Figure 16-9.** Dashboard connected to streaming dataset

There are just a few charts and visuals you can use for a streaming dataset tile, which are listed in Figure 16-10.

## Add a custom streaming data tile

Choose a streaming dataset > Visualization design



**Figure 16-10.** Visuals that can connect to streaming datasets

You can now test the solution. Go to your form, enter a result, and see if the result immediately appears in the Power BI dashboard (depending on what part of the result you chose to show in the dashboard).

## Streaming Dataset Types

One of the most important aspects of a streaming dataset is understanding the types of datasets. The streaming dataset can have three types—Push, Streaming, and Hybrid.

### Push Datasets

When this dataset is created, the Power BI service automatically creates a database to store the data. For this type of real-time dataset, you can create reports.

### Streaming Datasets

With this dataset, the data is only stored in a temporary cache, which would expire. This dataset has no underlying database. You cannot create reports from this dataset; you can only create a dashboard.

### Hybrid Datasets

This dataset is a combined version of Push and Streaming. With this dataset, there is a temporary cache for the data that is coming and going, and there is also a database to store the data. You can create this by enabling the Historic Data Analysis option when creating the streaming dataset. This is the type of dataset used in the scenario in this chapter.

Understanding these three types of datasets is important because, if you want to also store the streaming data for further report creation and data analysis, then a Hybrid dataset is what you need instead of a Streaming dataset.

## Streaming Services

You can use three services for a real-time dataset for Power BI:

- Power BI streaming dataset and REST API
- Azure Stream Analytics
- PubNub

### Power BI Streaming Dataset and REST API

The example in this chapter uses the Power BI streaming dataset with Power BI REST API. This uses only Power BI objects and services. You don't need any additional services or licenses. However, there are some limitations. For example, you can have up to 1 million rows of data pushed every hour with the Push dataset; the request size also has some limitations.

Using the Power BI REST API does not necessarily mean using Power Automate. You can call the API using any other application.<sup>3</sup>

### Azure Stream Analytics

Azure Stream Analytics is the data streaming service of Microsoft Azure. It can be used for Power BI and many other tools and services in the Microsoft toolset. You can use this service to capture data input from IoT devices, for example, and pass part of that data to Power BI and another part to Azure Machine Learning for data mining. I wrote an article about a sample solution with Azure Stream Analytics and Power BI at [radacad.com/stream-analytics-and-power-bi-join-forces-to-real-time-dashboard](http://radacad.com/stream-analytics-and-power-bi-join-forces-to-real-time-dashboard).

### PubNub

PubNub is a streaming service that is not only for Microsoft. It integrates with many tools and services. You can check out the website for the details of how this service works.

## Creating Calculations on Real-Time Datasets

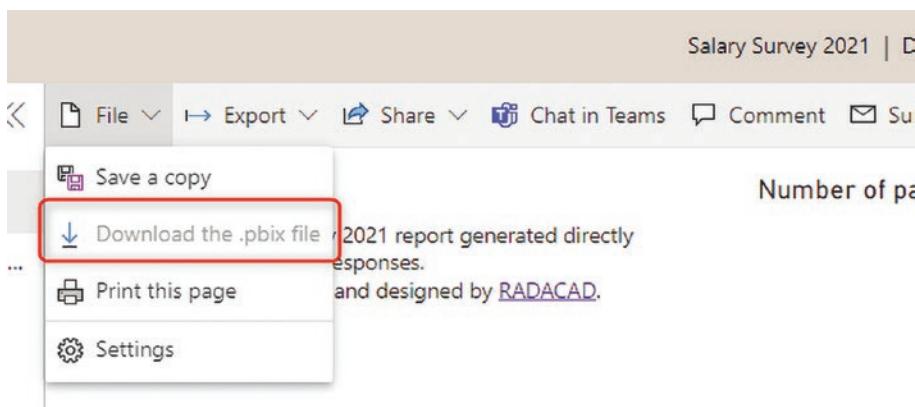
A streaming dataset is not like a normal Power BI dataset. In a normal Power BI dataset, you can use the Power BI Desktop to write DAX calculations and use Power Query to transform the data. A streaming dataset, however, is different. When you work with a streaming dataset, you are limited. There are some workarounds in writing calculations, which I explain in this section.

### Writing Calculations Using Q&A

The first workaround is to use Q&A to write simple calculations. The big development problem with the streaming dataset is that you don't have a PBIX file (see Figure 16-11). You cannot open this solution in the Power BI Desktop. And as a result, you cannot bring in other datasets, modify or edit the data using Power Query, or write calculations using DAX.

---

<sup>3</sup>Learn more about how to use a C# application to push data to a streaming dataset in Power BI: [radacad.com/monitor-real-time-data-with-power-bi-dashboards](http://radacad.com/monitor-real-time-data-with-power-bi-dashboards); [radacad.com/integrate-power-bi-into-your-application-part-6-real-time-streaming-and-push-data](http://radacad.com/integrate-power-bi-into-your-application-part-6-real-time-streaming-and-push-data)



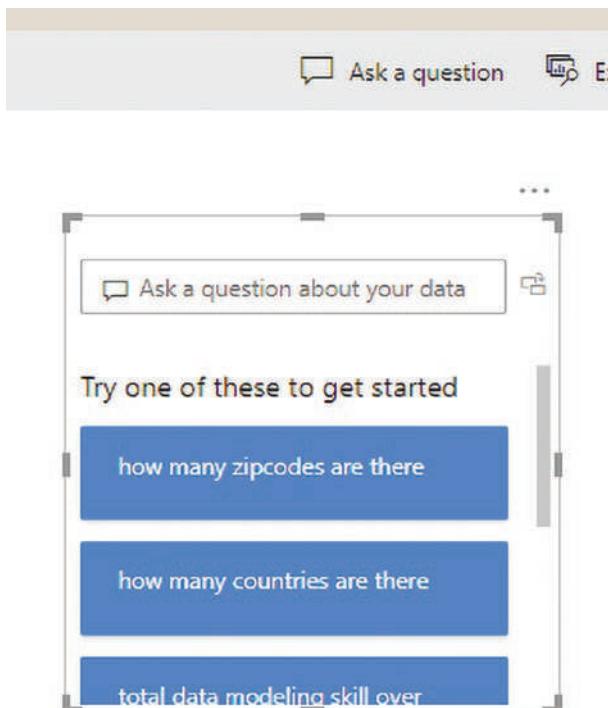
**Figure 16-11.** You cannot download a PBIX file of a streaming Power BI dataset

The download option is not even available from the dataset (see Figure 16-12).

A screenshot of the Power BI service interface. At the top, there are tabs for "All", "Content", and "Datasets + dataflows", with "Datasets + dataflows" being the active tab. Below this, a list of datasets is shown in a table format with columns "Name" and "Type". There are six datasets listed, all of which are "Dataset" type. The last dataset in the list is "Salary Survey 2021". A context menu is open for this dataset, listing options: "Create report", "Delete", "Manage permissions", "Quick insights", "Edit", and "API Info". The "Edit" option is highlighted with a red box.

**Figure 16-12.** You cannot download a Power BI streaming dataset or report

Although you cannot write calculations, you can use Q&A. Q&A in Power BI is a very useful way to analyze data and build visualizations by just asking questions.<sup>4</sup> Q&A gives you the existing field's list and understands some calculations, so you can use it to create calculations in a streaming dataset.



**Figure 16-13.** Using Q&A in the Power BI Service

As you can see in Figure 16-4, I can subtract the two fields—Salary and Salary pre-COVID-19—and add more criteria to my visual. This creates the calculation and uses it in that visual.

<sup>4</sup>Learn more about Q&A at [radacad.com/qa-visual-in-power-bi-desktop](http://radacad.com/qa-visual-in-power-bi-desktop)

The screenshot shows the Power BI Q&A interface. At the top, there is a search bar with the query: "salary - salary pre-COVID-19 by country where salary is greater than 0". Below the search bar, there are several options for visualizing the results: "table" (highlighted in yellow), "map", "matrix", and "card". A "Show more" link is also present. Below these options is a table with the following data:

Country	Salary
Switzerland	200,000.00
South Africa	180,000.00
US	102,500.00
Mexico	81,000.00
poland	56,000.00
Ireland	40,000.00
Slovakia	31,710.00
Romania	30,000.00

**Figure 16-14.** Running calculations in Q&A

The screenshot shows the Power BI ribbon. A tooltip message "Turn this Q&A result into a standard visual." is displayed above the ribbon tabs. Below the ribbon, there is a button labeled "is greater than 0 as table" with a red box drawn around it. To the right of this button is a small circular icon with a question mark inside.

**Figure 16-15.** Turning the Q&A result into a standard Power BI visual

When I convert that visual into a normal visual using the button highlighted in Figure 16-15, the standard visual will include the calculation, as shown in Figure 16-16.

The screenshot shows the Power BI Desktop interface. On the left, there is a table visual titled "Country" with a single column "Salary - Salary pre-COVID-19". The table lists various countries and their corresponding salaries. A context menu is open over the row for South Africa, showing options like "Add data", "Format", "Delete", and "Copy". To the right of the table is the "Filters" pane, which contains filters for "Country" (is All), "Salary" (is greater than 0), and "Salary - Salary pre-COVID-19" (is All). Below the filters is a section for "Values" with a dropdown set to "Country". The "Visualizations" pane is also visible on the right, showing icons for different types of charts.

**Figure 16-16.** Calculation in a standard Power BI visual

This will not of course support all types of calculations. I believe many complex scenarios are not supported. However, the ability to do simple calculations in a streaming dataset (that otherwise there was no other way to do) is great.

If the calculation that you are after is too complex for the Q&A, you should do the calculation before sending the data row to the streaming dataset.

## Live Connection to a Streaming Dataset

The second method is to create a report with a live connection to the Power BI dataset. This type of connection won't be real-time anymore, and it works when you have a Push or Hybrid dataset.

If the calculation you are after is a complex dynamic calculation, you can write DAX expressions for it. This can be done using a live connection to the streaming dataset and the Power BI Desktop, as shown in Figure 16-17.

The screenshot shows the Power BI Desktop ribbon. The "Home" tab is selected. In the "Get data" section of the ribbon, there are several options: "Cut", "Copy", "Format painter", "Clipboard", "Get data", "Excel", "Power BI datasets", "SQL Server", and "Ent". Below the ribbon, a "Common data sources" pane is open, listing "Excel", "Power BI datasets", and "Power BI dataflows". The "Power BI datasets" option is highlighted with a red box.

**Figure 16-17.** Getting data from the Power BI dataset

The streaming dataset can then be used as the source of the live connection, as shown in Figure 16-18.

Name	Endorsement	Owner	Workspace	Refreshed
2021 detailed Salary Survey		Reza Rad	Public Reports	3 days ago
Salary Survey 2021		Reza Rad	Public Reports	16 days ago

**Figure 16-18.** Choosing a streaming dataset as the source of the live connection

This will create a live connection to the streaming dataset, as shown in Figure 16-19.

Connected live to the Power BI dataset: Salary Survey 2021 in Public Reports [Make changes to this model](#)

**Figure 16-19.** Live connection created to the streaming dataset

When the live connection is created, you can write DAX measures using normal measures or quick measures, as highlighted in Figure 16-20.

```

1 List of Salary values =
2 VAR __DISTINCT_VALUES_COUNT = DISTINCTCOUNT('RealTimeData'[Salary])
3 VAR __MAX_VALUES_TO_SHOW = 3
4 RETURN
5 IF(
6   __DISTINCT_VALUES_COUNT > __MAX_VALUES_TO_SHOW,
7   CONCATENATE(
8     CONCATENATEX(
9       TOPN(
10      __MAX_VALUES_TO_SHOW,
11      VALUES('RealTimeData'[Salary]),
12      'RealTimeData'[Salary],
13      ASC
14    ),
15    'RealTimeData'[Salary],
16    ",",
17    'RealTimeData'[Salary],
18    DESC
19  ),
20  ", etc."
21 ),
22 CONCATENATEX(
23   VALUES('RealTimeData'[Salary]),
24   'RealTimeData'[Salary],
25   ",",
26   'RealTimeData'[Salary],
27   DESC
28 )
29 )

```

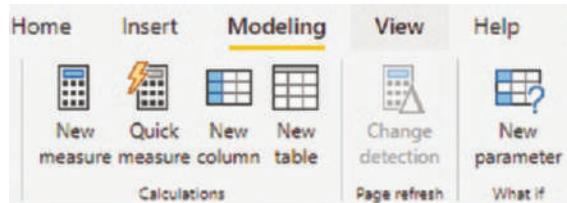
**Figure 16-20.** Writing DAX measures on a live connection to a Power BI streaming dataset

## Would it Be Real-Time?

One important consideration here is that the report generated this way won't be a real-time report or dashboard. This would be a live connection to the real-time dataset. That means any dashboard created on top of it would be refreshed with the frequency that's set at the dashboard level. Every time you browse the report, you will have the report from the data at that moment.

## DirectQuery to the Streaming Dataset: Even More Flexibility

The ability to write measures is great. However, sometimes, you want to create columns too, and maybe include another data table to combine with this data. Using DirectQuery to Power BI dataset, you can do this, as Figure 16-21 indicates.



**Figure 16-21.** Changes in the model using DirectQuery to the Power BI dataset

You can have a table as the DirectQuery connection to the streaming Power BI dataset, which enables you to import other data tables and have a more complete solution; see Figure 16-22.

Name	'RealTimeData'
Storage mode	DirectQuery
Data source type	SQL Server Analysis Services
Server	powerbi://api.powerbi.com/v1.0/myorg/public%20reports
Database	Salary Survey 2021

**Figure 16-22.** DirectQuery to a Power BI streaming dataset

## Summary

A real-time streaming dataset is different from a Live or DirectQuery connection. Using this dataset, you can see changes immediately after the data changes. The push data API or other streaming services can help you achieve this. The streaming dataset can also store the data in a database for further analysis. A streaming dataset is not like a normal Power BI dataset; it comes with limitations. It is important to know when to use a real-time streaming dataset versus Live and DirectQuery datasets.

## CHAPTER 17



# Performance Tuning with Aggregations

Power BI is not only a solution for small datasets—it also caters to big datasets. If the volume of data is huge, you can switch to DirectQuery mode. Because DirectQuery does not store a copy of the data in the memory of the machine that runs the Power BI model, Power BI will send queries back to the data source for every page rendered in the report.

However, as Figure 17-1 depicts, DirectQuery mode is slow. Consider a page with three, four, or five visuals. That page will send five queries to the data source with every change in the filtering context, such as changing the slicer, clicking a column chart to highlight part of it, or refreshing the page. Sending queries to the data source for every interaction makes DirectQuery mode very slow. Because of that, DirectQuery is not the best way to connect in Power BI. My suggestion is to use DirectQuery only when the other methods cannot be used.

**DirectQuery will send all queries to the data source =>  
DirectQuery is SLOW**



**Figure 17-1.** DirectQuery has a downside

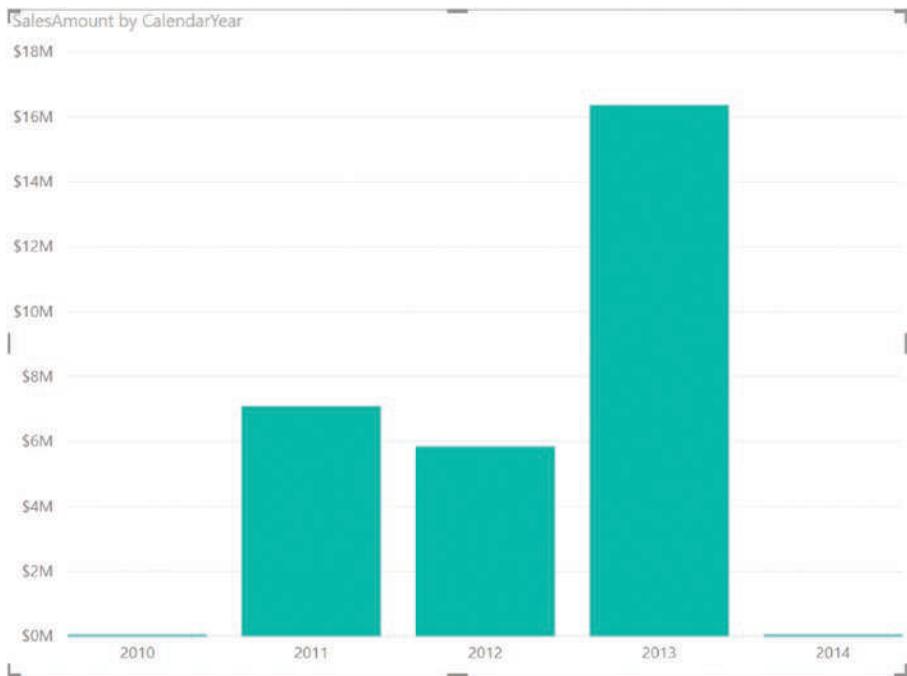
Earlier in this book, you read about another type of connection in Power BI, called composite mode. Composite mode allows part of the model to use DirectQuery (for large tables) and part of the model to use Import Data (for smaller tables). This way, you get better performance when you work with smaller tables because they query the in-memory structure of the data.

However, the table(s) that are part of the DirectQuery connection are still slow. Composite mode comes with a fantastic feature called *aggregations*. Aggregations speed up the DirectQuery sourced tables in a composite model. With aggregations, you can create layers of pre-aggregated values, which can be stored in memory and performed faster.

## How Does Aggregation Work?

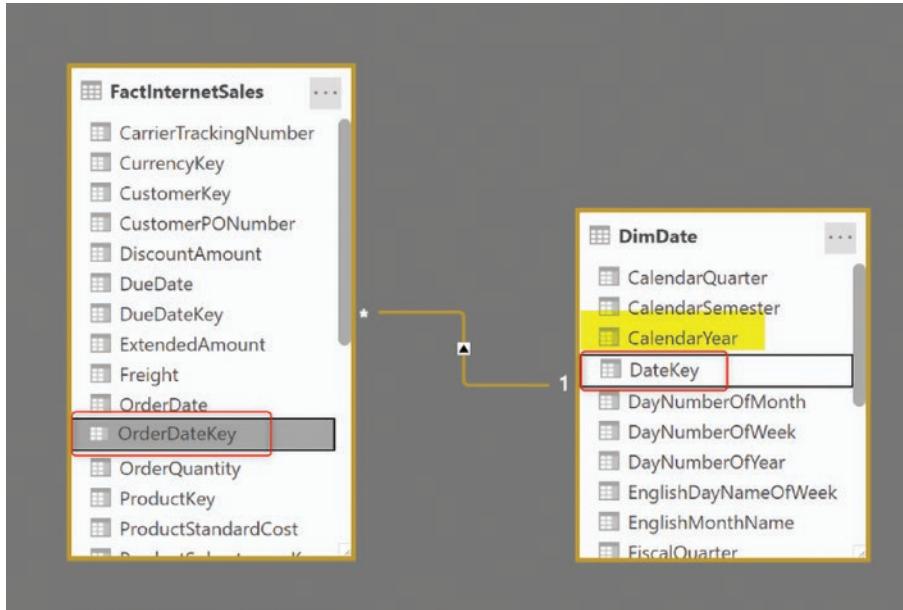
Imagine a fact table with 250 million rows. Such a fact table is big enough to be considered a good candidate for a DirectQuery connection. You don't want to load such a big table into memory, and the Power BI file size probably exceeds the 1GB limitation. Now, think about your reporting solution for a second. Do you always query this fact table at the finest or minimum granular level? Do you always look at single transaction in this table when you run a report on it?

The answer is likely no. Most of the time, you query the data by other fields or columns. For example, you might query the Sales value in the fact table by Year, as shown in Figure 17-2. Or you might query the fact table's values by the customer's education category. Or you might query the values in the fact table by each product. When you consider real-world scenarios, you are usually querying the fact table by aggregations of dimension tables.



**Figure 17-2.** Querying SalesAmount from the fact table by CalendarYear

Figure 17-3 is querying SalesAmount from the fact table by the CalendarYear from DimDate.



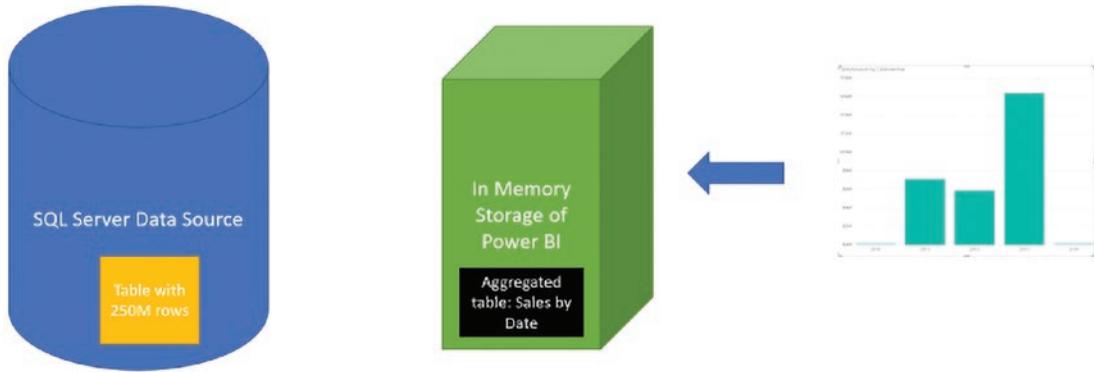
**Figure 17-3.** Creating relationships

## Aggregated Table

So, if you are querying only by CalendarYear, you can create an aggregated table. The aggregated table can pre-calculate the Sum of SalesAmount for every CalendarYear. In this example, you would have a table with five rows. One row for each year, from 2010 to 2014. The aggregated table is so small (five rows) that it can be easily imported into memory, and whenever you query that table, you'll get a super-fast result.

You might also query this by quarter. You may want to drill down into the month, week, or even every day. That is fine. Considering there are 365 days in each year, you have a table with  $5 \times 365$  rows. This would be a table including 1827 rows maximum (you may have one or two leap years in the period as well). The aggregated table size is still very tiny (fewer than 2000 rows) compared to a fact table with 250 million rows (see Figure 17-4). You can still import your aggregated table into memory. Such a table will cover all the data analysis you want every day.

Sales by Year will hit the aggregated table in the memory



**Figure 17-4.** Aggregated table

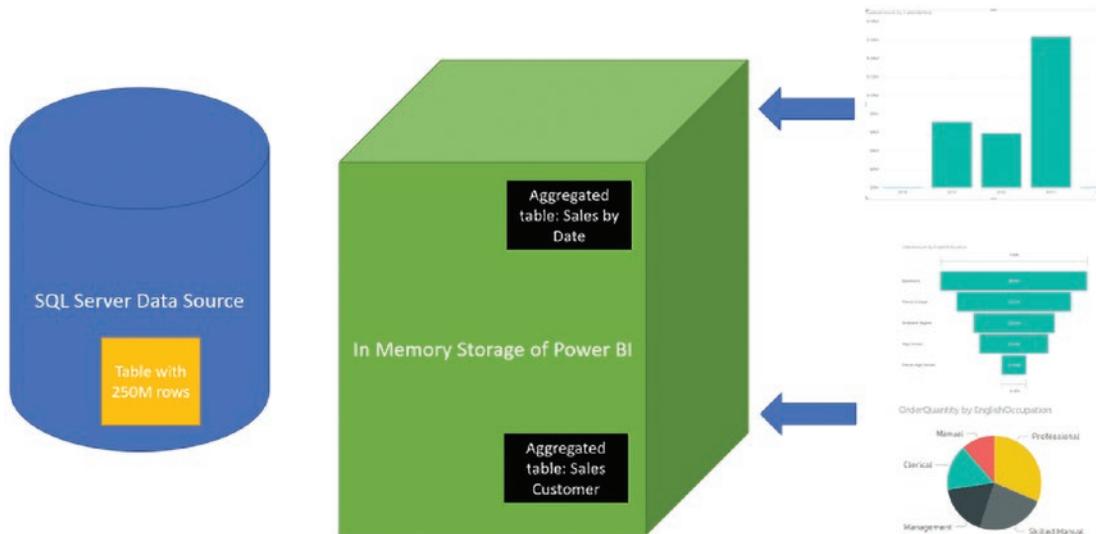
## Layers of Aggregation

You may need to create other aggregations by other dimensions too. Because dimensions are much smaller than the fact table, your aggregated tables will always be smaller than the fact table. These aggregated tables are your layers of aggregation in the model.

The golden rule for a composite model in Power BI is, do not use a lower-level table if there is an aggregation on top of it that can be used for the query.

If you are querying a DirectQuery table with 250 million rows, but you are only querying it by date, then Power BI acts differently. Power BI will not send a query to the data source of the fact table. Instead, it will query the aggregated table in memory, as shown in Figure 17-5, and you'll get a fast response. Power BI only switches to the table underneath if aggregated tables cannot answer the question.

Queries will be answered by Aggregated tables if exists



**Figure 17-5.** Querying aggregated tables

The overall process for using an aggregation is as follows:

- Creating an aggregation table
- Creating relationships necessary for the aggregation table and the DirectQuery table
- Setting the Storage mode of the aggregated and dimension tables
- Configuring the aggregation

The process is explained in detail in the following sections.

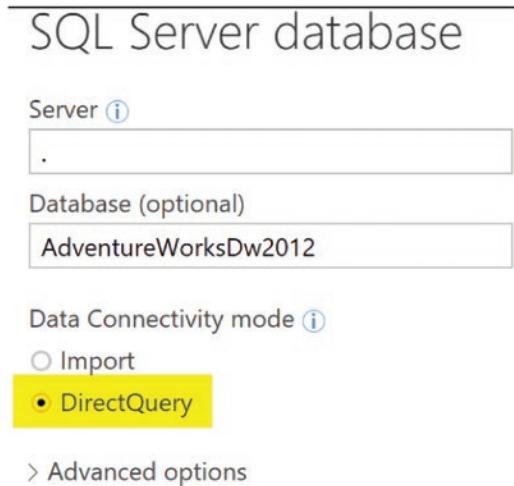
## Step 1: Create the Aggregated Table

If you want to follow the example scenario here, you need a SQL Server database named AdventureWorksDW. I made some changes to my dataset so it's bigger, so I can show you the functionality of aggregations. You can download the database link from here:

[radacad.com/files/adventureworksdw2012.bak](http://radacad.com/files/adventureworksdw2012.bak)

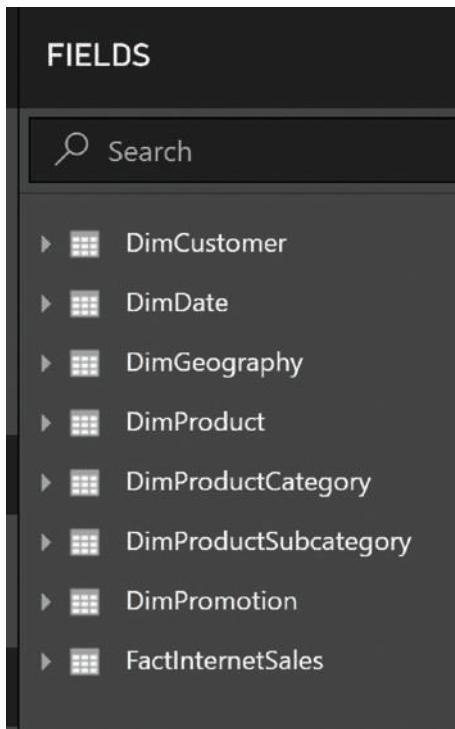
## Sample Model

This sample model analyzes the data in FactInternetSales (which is the big fact table). First create a Power BI report with a DirectQuery connection to SQL Server, as shown in Figure 17-6.



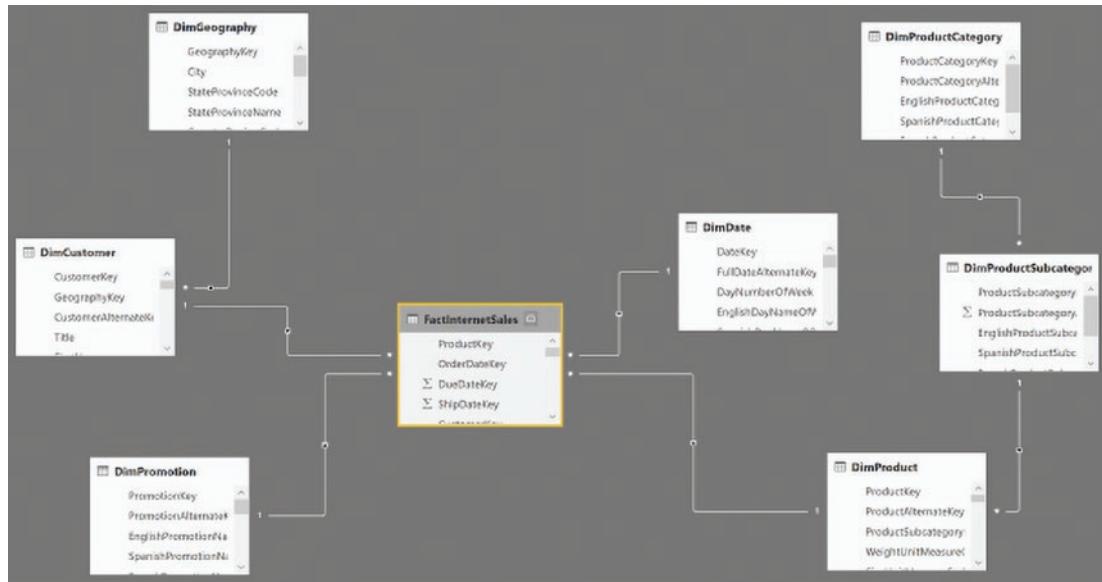
**Figure 17-6.** Creating a Power BI report with DirectQuery

With the DirectQuery option, select the FactInternetSales, DimCustomer, DimDate, DimProduct, DimProductCategory, DimProductSubCategory, DimPromotion, and DimGeography tables (the selections will be viewed, as shown in Figure 17-7).



**Figure 17-7.** Selecting tables through DirectQuery

Tables loaded into Power BI will have relationships. I limited the relationship between DimDate and the FactInternetSales table to one active relationship based on DateKey (in DimDate) and OrderDateKey (in FactInternetSales). The diagram of these relationships is shown in Figure 17-8.



**Figure 17-8.** Diagram of the table relationships

## What Is an Aggregated Table?

An aggregated table is a table in Power BI aggregated by one or more fields from the DirectQuery source table. In this case, the aggregated table is grouped by specific fields from the FactInternetSales table. You can use various methods to create an aggregated table. You can create an aggregated table with T-SQL statements from SQL Server, or you can create one in Power Query. Or you can even create one in DAX using Summarize or GroupBy or other aggregation functions. You can create an aggregated table in all other data transformation tools and query languages. Since this example does everything with Power BI, I create the aggregated table using Power Query.

---

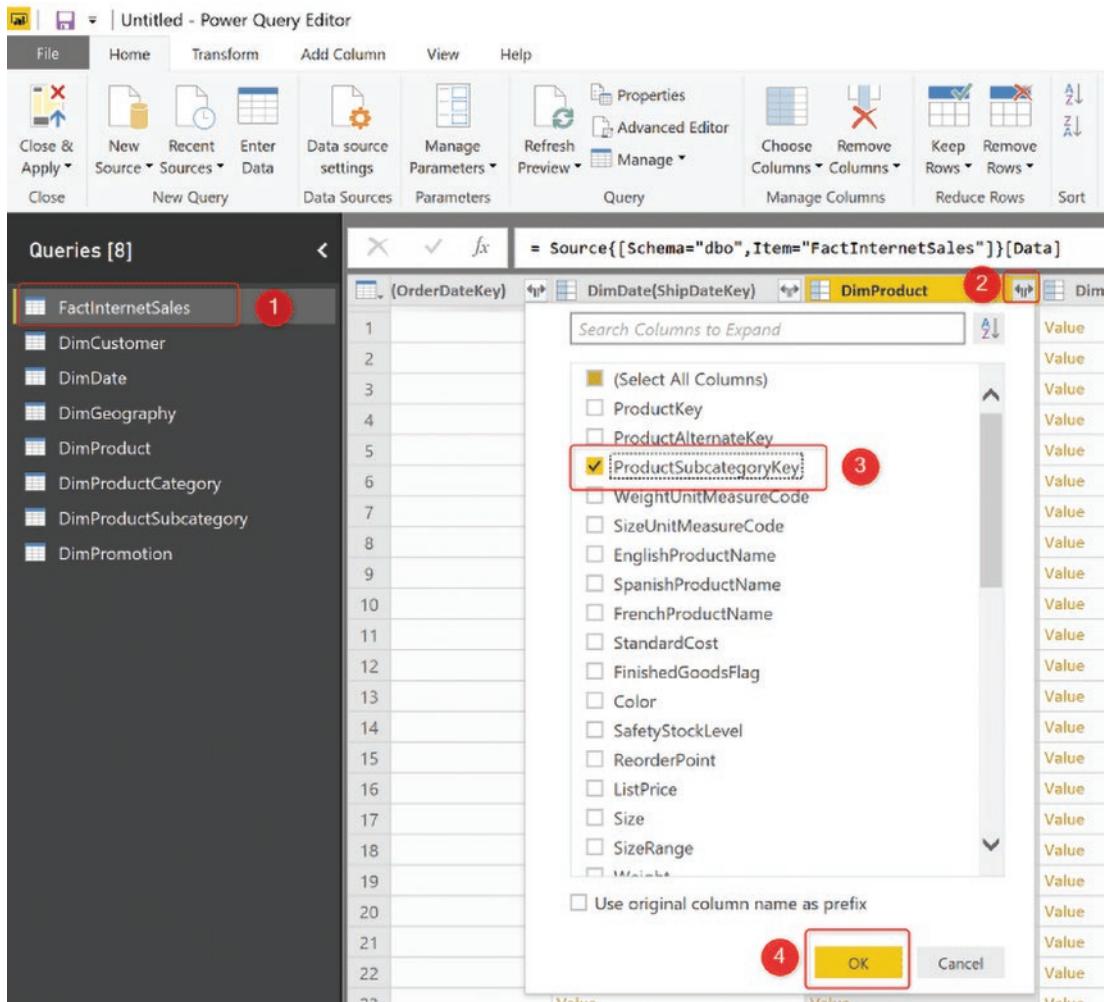
An aggregated table can be created in the data source with T-SQL queries, or in Power Query, or anywhere else that you can create a grouped table.

---

## Creating an Aggregated Table

Go to the Power Query Editor and select the FactInternetSales table. The aggregated table created in this example is going to include three fields—OrderDateKey, CustomerKey, and ProductSubCategoryKey. The first two fields exist in FactInternetSales. Using relationship columns, you can retrieve that last table.

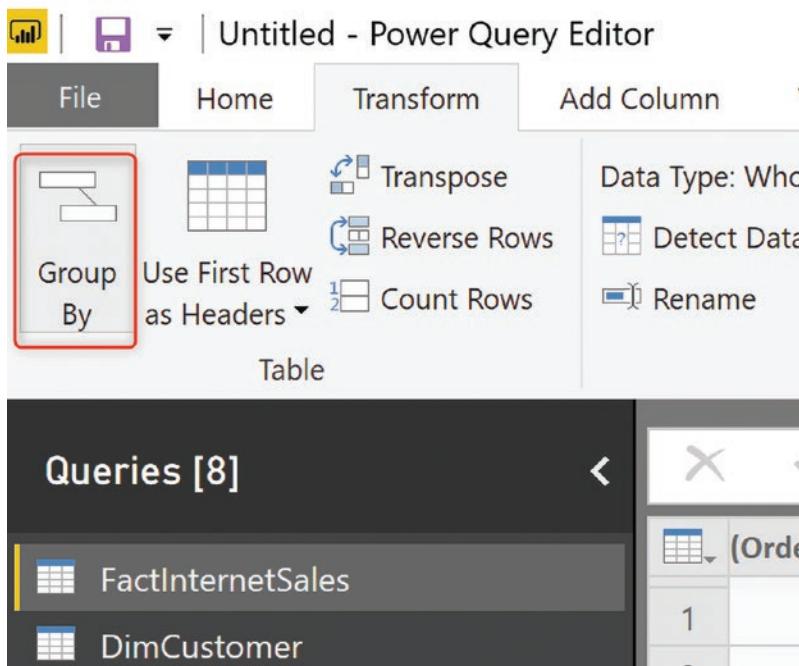
Scroll to the right in the FactInternetSales table columns to find Product, then click Expand. In the Expand options, simply select ProductSubCategoryKey. These steps are shown in Figure 17-9.



**Figure 17-9.** Setting up Power Query

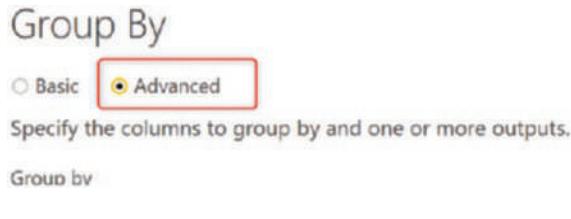
Now that you have ProductSubCategoryKey in the table, you can apply the Group By. But if you use Group By in the existing query, you will change it to the aggregated table. You do, however, need to keep the existing table intact. So you'll create a REFERENCE from the existing FactInternetSales table.<sup>1</sup> Name it Sales Agg and then you'll run the Group By on Sales Agg. In the Transformation tab, click Group By, as shown in Figure 17-10.

<sup>1</sup>To learn more about reference and how it differs from duplicate, visit [radacad.com/reference-vs-duplicate-in-power-bi-power-query-back-to-basics](http://radacad.com/reference-vs-duplicate-in-power-bi-power-query-back-to-basics)



**Figure 17-10.** Initiating Group By

When the Group By dialog box pops up, choose Advanced, as shown in Figure 17-11.



**Figure 17-11.** Selecting an advanced Group By

Choose OrderDateKey, CustomerKey, and ProductSubcategoryKey in the Group By fields, as Figure 17-12 shows.

## Group by

OrderDateKey  ...

CustomerKey

ProductSubcategoryKey

**Figure 17-12.** Group By field selection

Then add four aggregations, as shown in Figure 17-13.

## Group By

Basic  Advanced

Specify the columns to group by and one or more outputs.

### Group by

OrderDateKey

CustomerKey

ProductSubcategoryKey

New column name	Operation	Column
SalesAmount_Sum	Sum	SalesAmount
UnitPrice_Sum	Sum	UnitPrice
UnitPrice_Count	Count Rows	
FactInternetSales_Count	Count Rows	

**Figure 17-13.** Adding aggregations

Let's call this table the Sales Agg table. Figure 17-14 shows the Sales Agg table.

	OrderDateKey	CustomerKey	ProductSubcategoryKey	1.2 SalesAmount_Sum	1.2 UnitSales_Sum	1.2 UnitSales_Count	1.2 FactInternetSales_Count
1	20130624	25357	39	8.99	8.99	1	1
2	20130715	16247	39	8.99	8.99	1	1
3	20140114	26581	31	34.99	34.99	1	1
4	20130106	27666	2	2570.27	2570.27	1	1
5	20130729	39724	37	34.99	34.99	1	1
6	20130718	27809	2	1120.49	1120.49	1	1
7	20130317	35488	28	14.98	34.98	2	2
8	20130326	24810	32	34.99	34.99	1	1
9	20130209	1275					
10							
11	20130723	2785					
12	20130811	27779					
13	20130518	21246	37	52.99	52.99	2	2
14	20130203	26257	37	33.00	33.00	2	2
15	20131209	23388	37	34.99	34.99	2	2
16	20131211	27833	2	539.99	539.99	1	1
17	20131213	31272	37	32.99	32.99	1	1
18	20130510	16553	1	259.09	259.09	1	1
19	20130511	13836	39	8.99	8.99	1	1
20	20130704	22730	31	34.99	34.99	1	1
21	20130729	33880	2	269.09	269.09	1	1
22	20130819	24230	37	8.99	4.99	1	1
23	20130927	17898	37	52.99	34.99	1	1

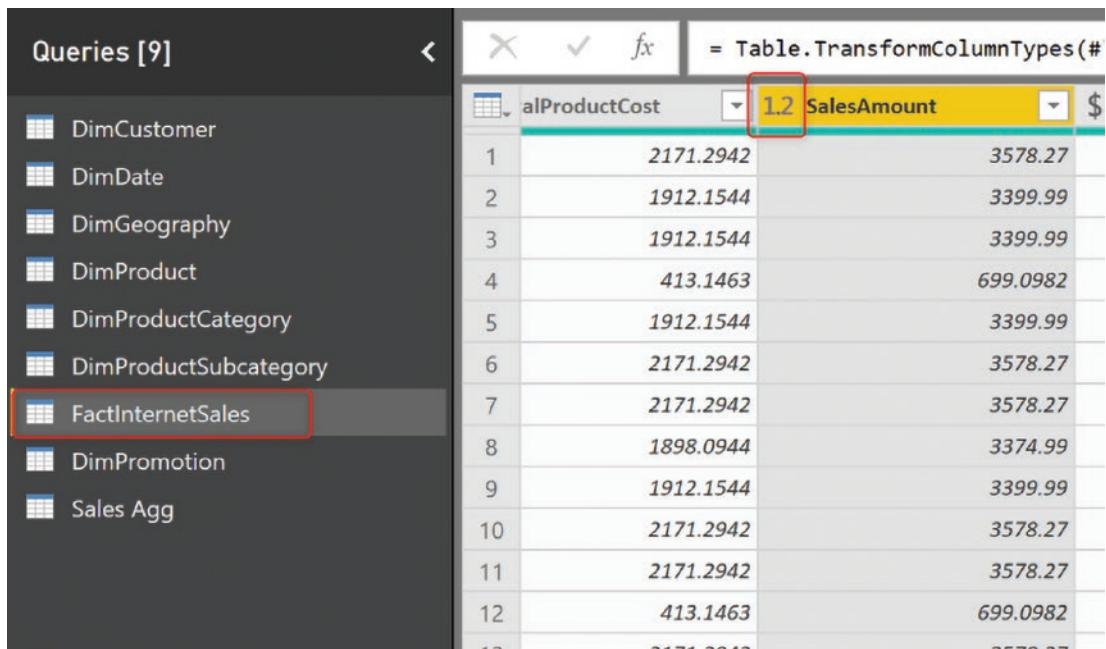
Figure 17-14. Sample aggregated table

## Key Consideration for Upcoming Steps

Aggregation columns in the aggregated table should follow specific rules.

### Rule #1: Exact Match for Data Types of Aggregations by Sum, Min, Max, or Average

Columns that you apply an operation on, such as Sum, Average, Min, or Max, should have exactly the same data type as the original source column after the aggregation. If they don't, change the data types to have the same. For example, the SalesAmount field in FactInternetSales must have a Decimal data type, as shown in Figure 17-15.

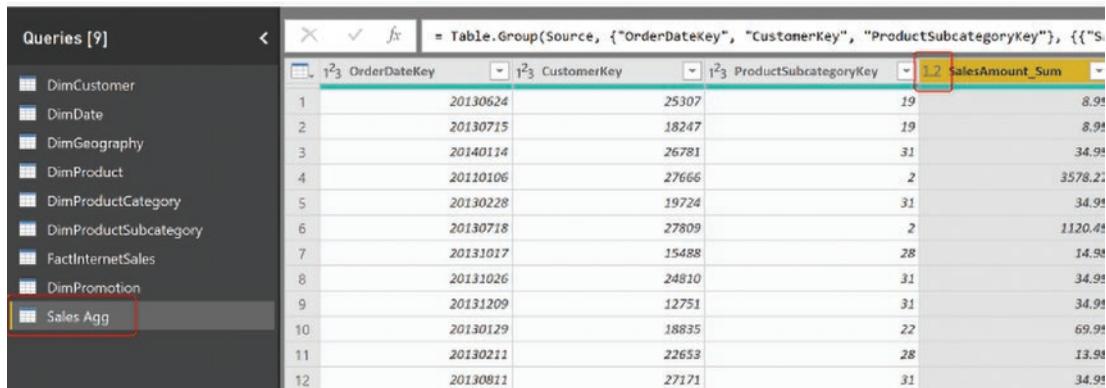


The screenshot shows the Power BI Data Editor interface. On the left, the 'Queries [9]' pane lists various tables: DimCustomer, DimDate, DimGeography, DimProduct, DimProductCategory, DimProductSubcategory, FactInternetSales, DimPromotion, and Sales Agg. The 'FactInternetSales' table is currently selected and highlighted with a red border. On the right, the data preview pane displays a portion of the FactInternetSales table. The 'SalesAmount' column is highlighted with a yellow background and has a data type indicator '1.2' above it, confirming it is a decimal type.

	alProductCost	SalesAmount
1	2171.2942	3578.27
2	1912.1544	3399.99
3	1912.1544	3399.99
4	413.1463	699.0982
5	1912.1544	3399.99
6	2171.2942	3578.27
7	2171.2942	3578.27
8	1898.0944	3374.99
9	1912.1544	3399.99
10	2171.2942	3578.27
11	2171.2942	3578.27
12	413.1463	699.0982

**Figure 17-15.** Confirming that the columns use the decimal data type

The SalesAmount\_Sum column in the Sales Agg table should also have the same data type. In this case, it's Decimal, as shown in Figure 17-16.



The screenshot shows the Power BI Data Editor interface. On the left, the 'Queries [9]' pane lists various tables: DimCustomer, DimDate, DimGeography, DimProduct, DimProductCategory, DimProductSubcategory, FactInternetSales, DimPromotion, and Sales Agg. The 'Sales Agg' table is currently selected and highlighted with a red border. On the right, the data preview pane displays a portion of the Sales Agg table. The 'SalesAmount\_Sum' column is highlighted with a yellow background and has a data type indicator '1.2' above it, confirming it is a decimal type.

	OrderDateKey	CustomerKey	ProductSubcategoryKey	SalesAmount_Sum
1	20130624	25307	19	8.95
2	20130715	18247	19	8.95
3	20140114	26781	31	34.95
4	20110106	27666	2	3578.27
5	20130228	19724	31	34.95
6	20130718	27809	2	1120.45
7	20131017	15488	28	14.95
8	20131026	24810	31	34.95
9	20131209	12751	31	34.95
10	20130129	18835	22	69.95
11	20130211	22653	28	13.95
12	20130811	27171	31	34.95

**Figure 17-16.** The data type of SalesAmount\_Sum is decimal

Note that having the Decimal data type is not important to this rule. It is important that both data types match exactly. This process, in this example, should be completed for the SalesAmount\_Sum and UnitPrice\_Sum columns.

#### Rule #2: The Whole Number Data Type Is Mandatory for Aggregations by Count

Any aggregations that use Count as their aggregation function should have the data type of the whole number or, let's say, Integer. The sample table has two columns with the Count function, as shown in Figure 17-17.

UnitPrice_Count	Count Rows	
FactInternetSales_Count	Count Rows	

**Figure 17-17.** Sample columns with the Count function

Make sure that these two columns have a Whole Number data type after the Group By transformation, as shown in Figure 17-18.

UnitPrice_Count	FactInternetSales_Count
8.99	1
8.99	1
4.99	1
8.27	1
4.99	1
0.49	1
4.98	2
4.99	1
4.99	1
9.99	1
3.98	2

**QUERY SETTINGS**
  
**PROPERTIES**  
 Name: Sales Agg  
[All Properties](#)
  
**APPLIED STEPS**  
 Source  
 Grouped Rows  
X Changed Type

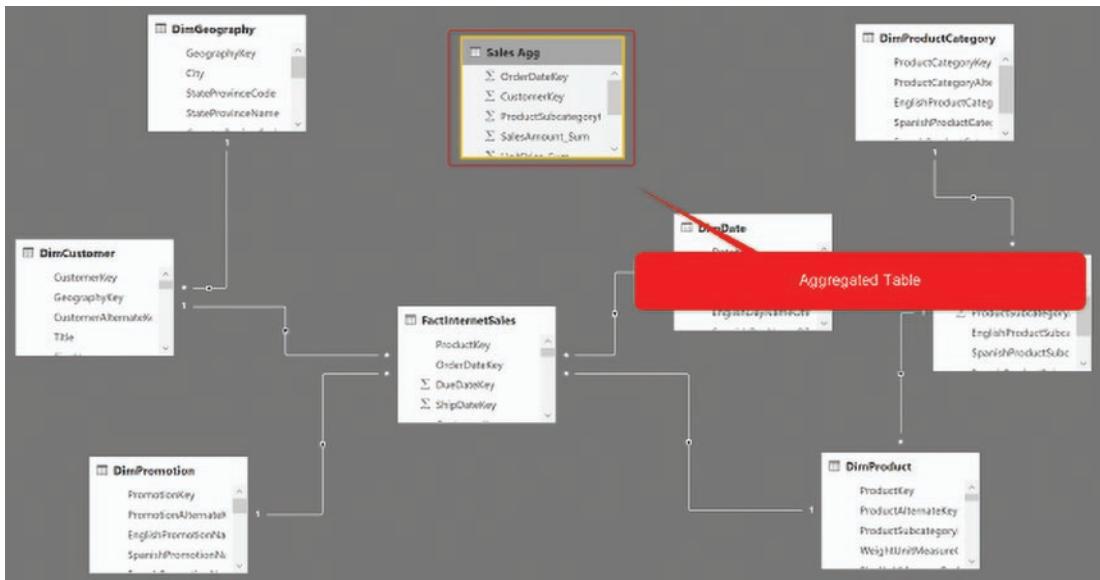
**Figure 17-18.** Column using whole number data types

These two rules are important in the next steps of configuring an aggregation in Power BI.

## An Aggregated Table Is an Import Table

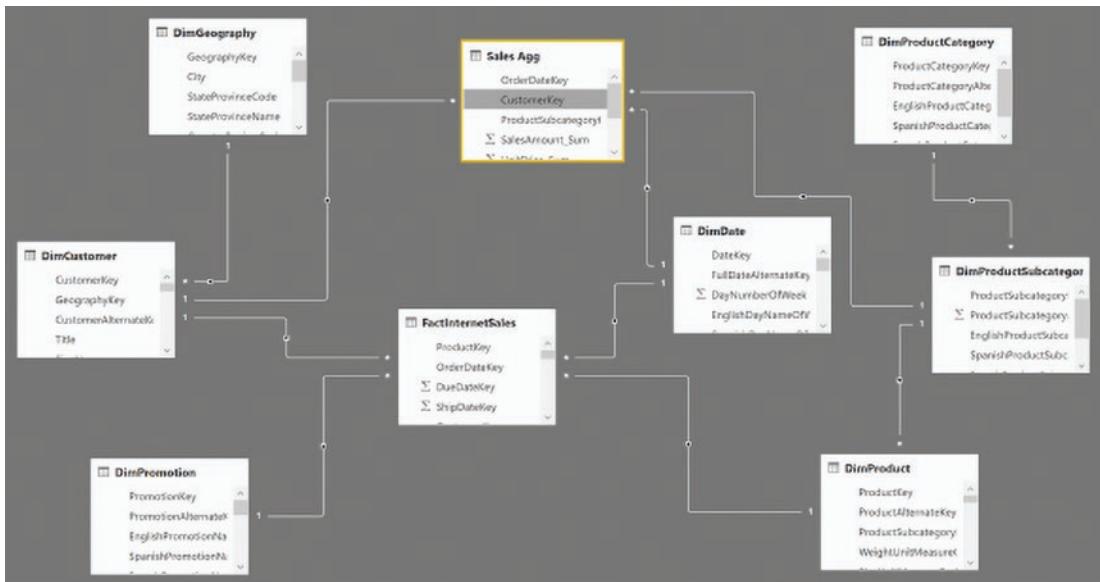
The aggregated table you created is called Sales Agg. Because this table is much smaller than the FactInternetSales table, it can be stored in memory. This way, you get the best query response time when you query something at the aggregated level.

Your data model in Power BI should now show the aggregated table, as shown in Figure 17-19.



**Figure 17-19.** Completed data model with the aggregated table

Congratulations, you built an aggregated table! Now you can create relationships between this table and the three dimension tables: DimCustomer, DimDate, and DimProductSubcategory. Figure 17-20 displays the full relationship diagram.



**Figure 17-20.** Diagram of relationships between the aggregate table and the dimension tables

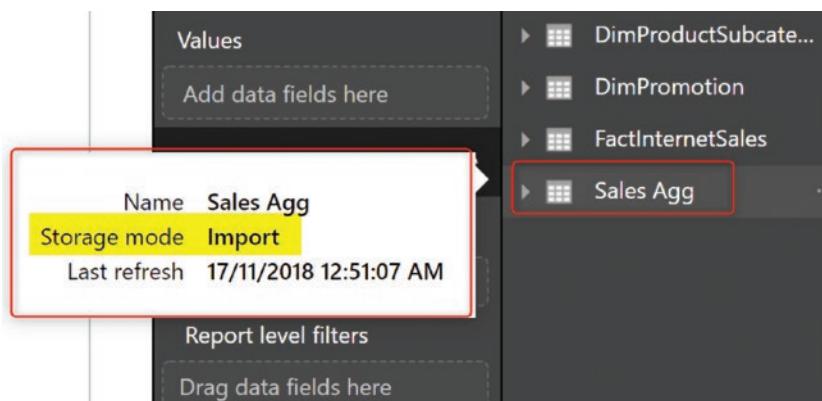
## Dual Storage Mode; The Most Important Configuration for Aggregations!

### Step 2: Power BI Aggregations

An aggregated table can be a layer on the DirectQuery source table. This table needs to have a proper relationship set up with other tables and also the proper storage mode configuration. Configuring the storage mode of the composite model is a critical part of setting up the aggregation. The storage mode configuration of Import Data and DirectQuery is self-explanatory, but what about Dual Storage mode? The next sections explain in detail what Dual mode is. Let's dig in.

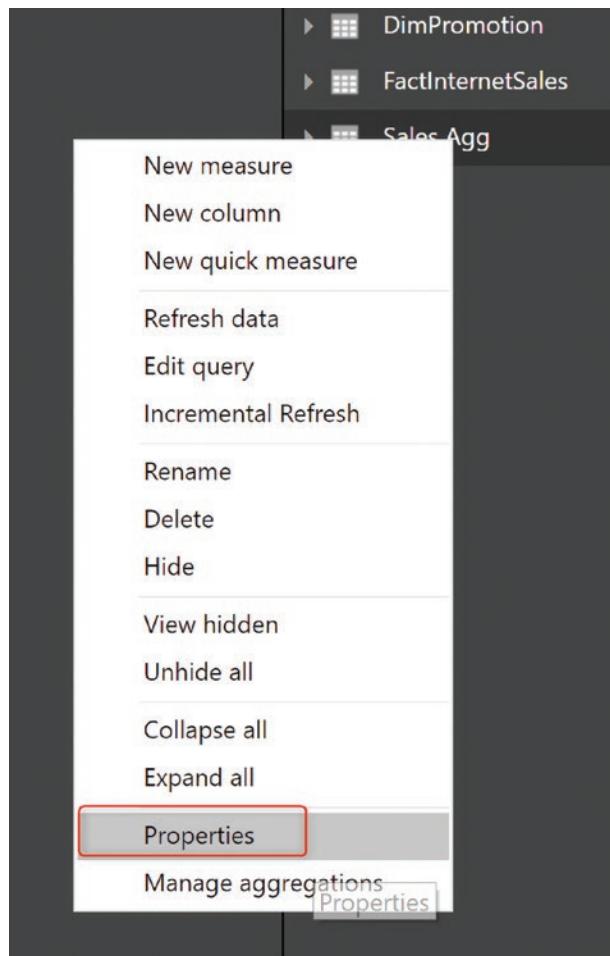
### What Is Storage Mode?

Storage mode in Power BI tables determines where the data of that table is stored and how queries are sent to the data source. You can determine the storage mode of a table by hovering the mouse on it in the Power BI Desktop in the fields section, as shown in Figure 17-21.



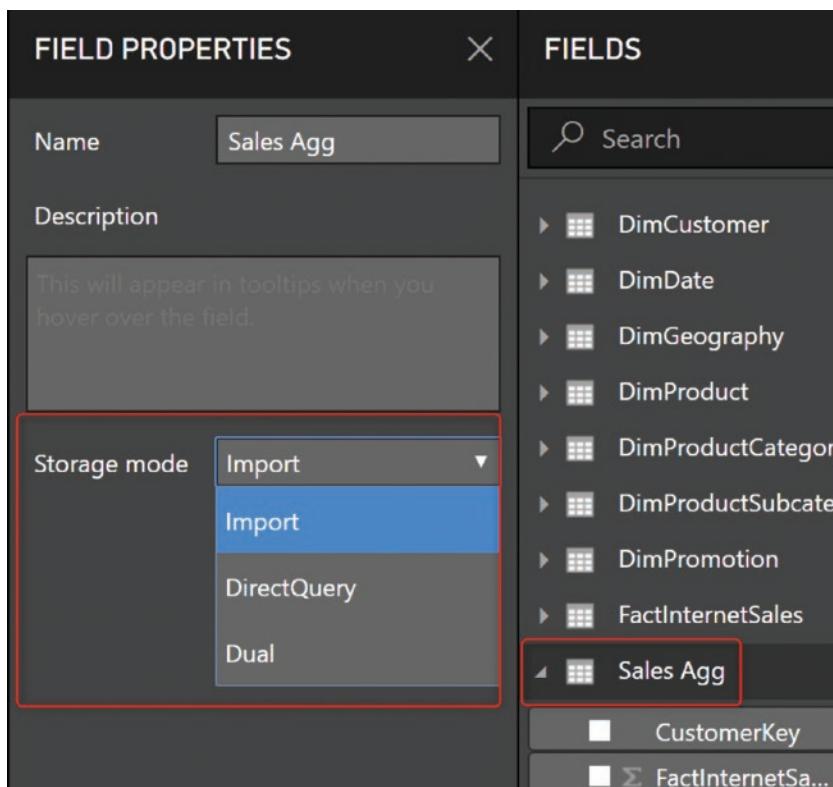
**Figure 17-21.** One way to confirm the table storage mode

You can also determine the storage mode by right-clicking the table and selecting properties, as shown in Figure 17-22.



**Figure 17-22.** Alternate method of confirming the storage mode, step one

In the Properties pane, you can see the Storage Mode drop-down list (see Figure 17-23).



**Figure 17-23.** Alternate method of confirming the storage mode, step two

There are three storage modes:

- Import
- DirectQuery
- Dual

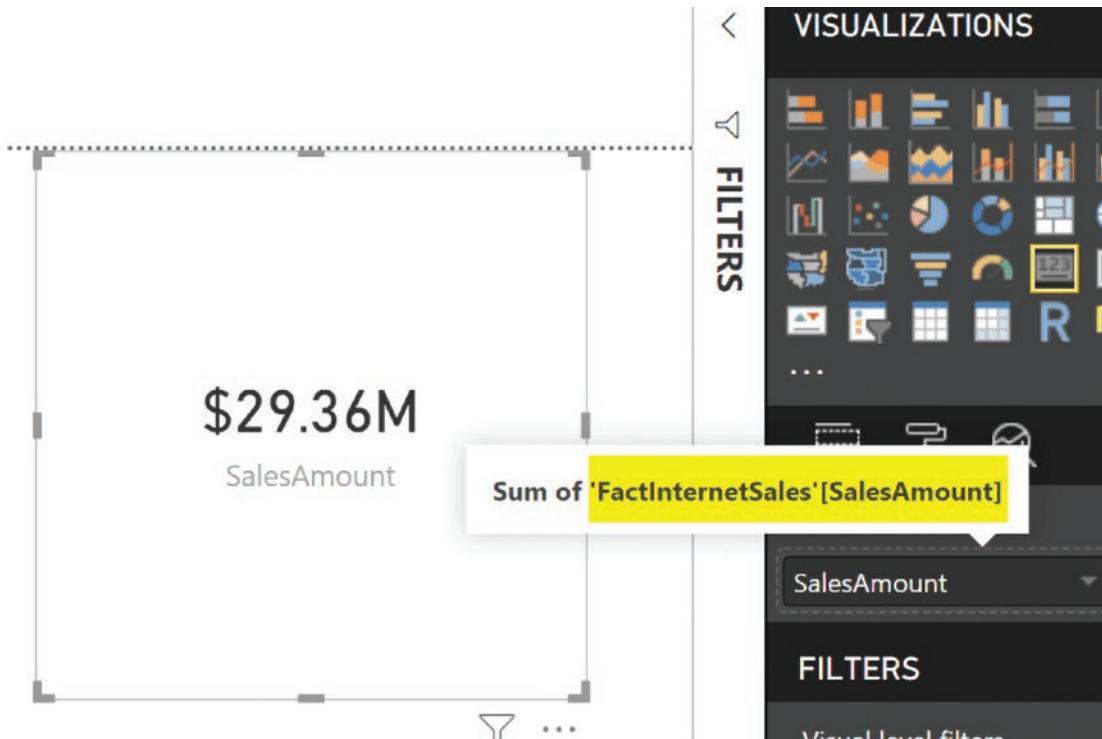
## Import Data

Import and DirectQuery are the obvious options in this list. For example, If a table's storage mode is Import Data, then it means the data of that table will be stored in the in-memory storage of the Power BI server (the machine that runs the Power BI engine). Every query to the data will query the in-memory structure, not the data source.

Suppose you have a table sourced from SQL Server that uses the Import Data storage mode. Then a copy of that data will be stored in the memory engine of Power BI. Whenever you refresh a visualization in the Power BI report, it will query the in-memory structure rather than sending the query to the SQL Server data source.

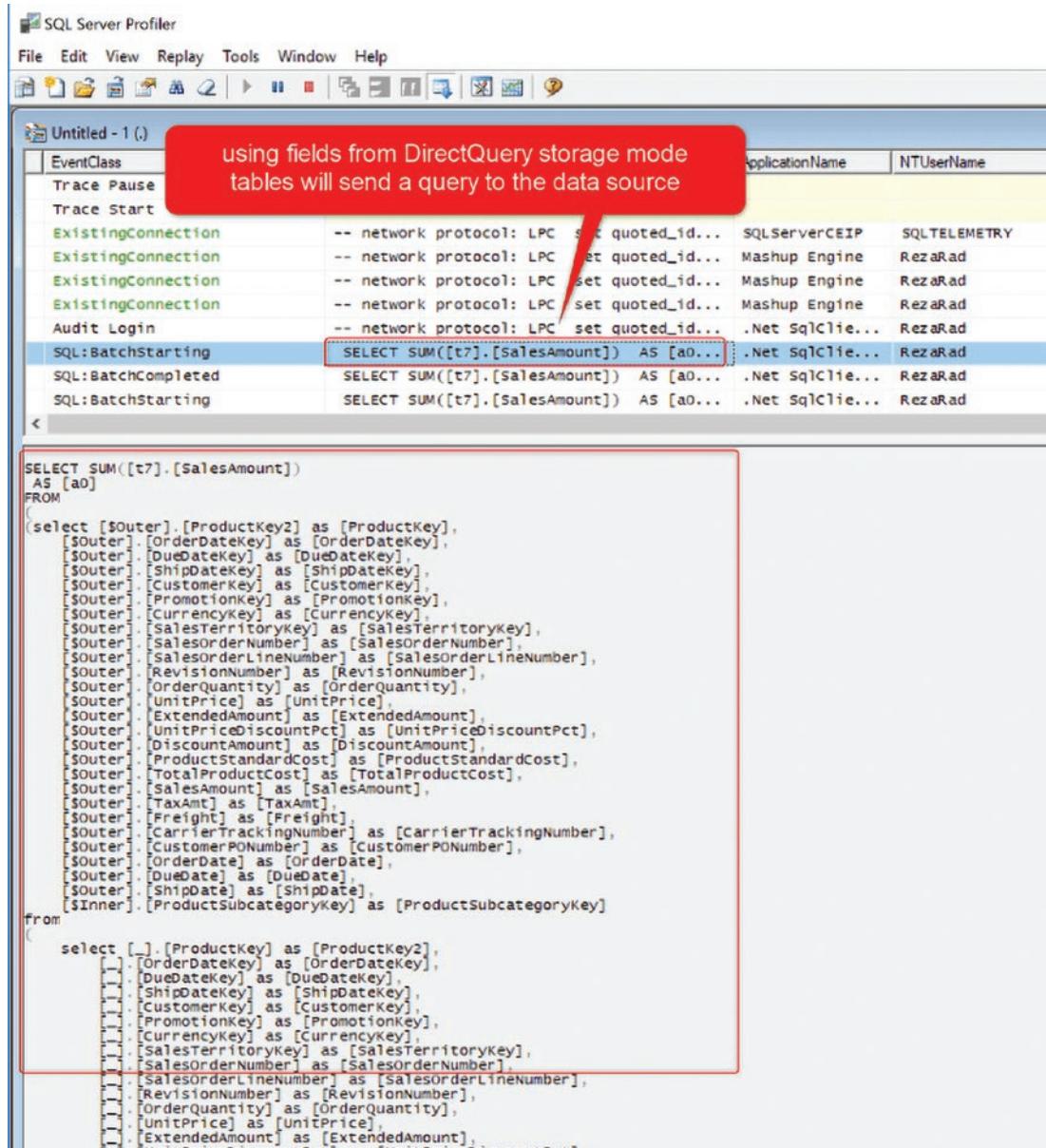
## DirectQuery

Tables with the DirectQuery storage mode keep the data in the data source. FactInternetSales stores the data in the SQL Server data source in the example dataset. If you have a visualization from a table with this storage mode, Power BI will send a T-SQL query to the data source and get the results. For example, Figure 17-24's visualization in the Power BI report is just a card visual of the SalesAmount field in FactInternetSales.



**Figure 17-24.** The SalesAmount field comes from the FactInternetSales table

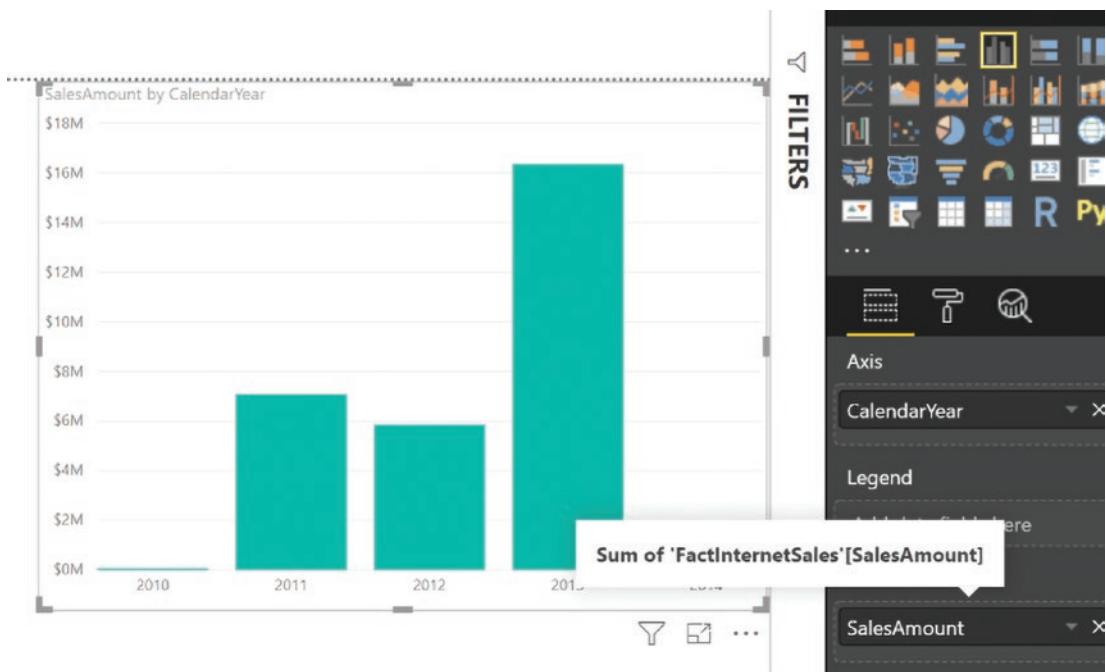
Because this table uses the DirectQuery storage mode, if you run SQL Profiler simultaneously (to capture queries sent to the data source), you'll get a query captured to the SQL Server database, as shown in Figure 17-25.



**Figure 17-25.** View of the query sent to the data source

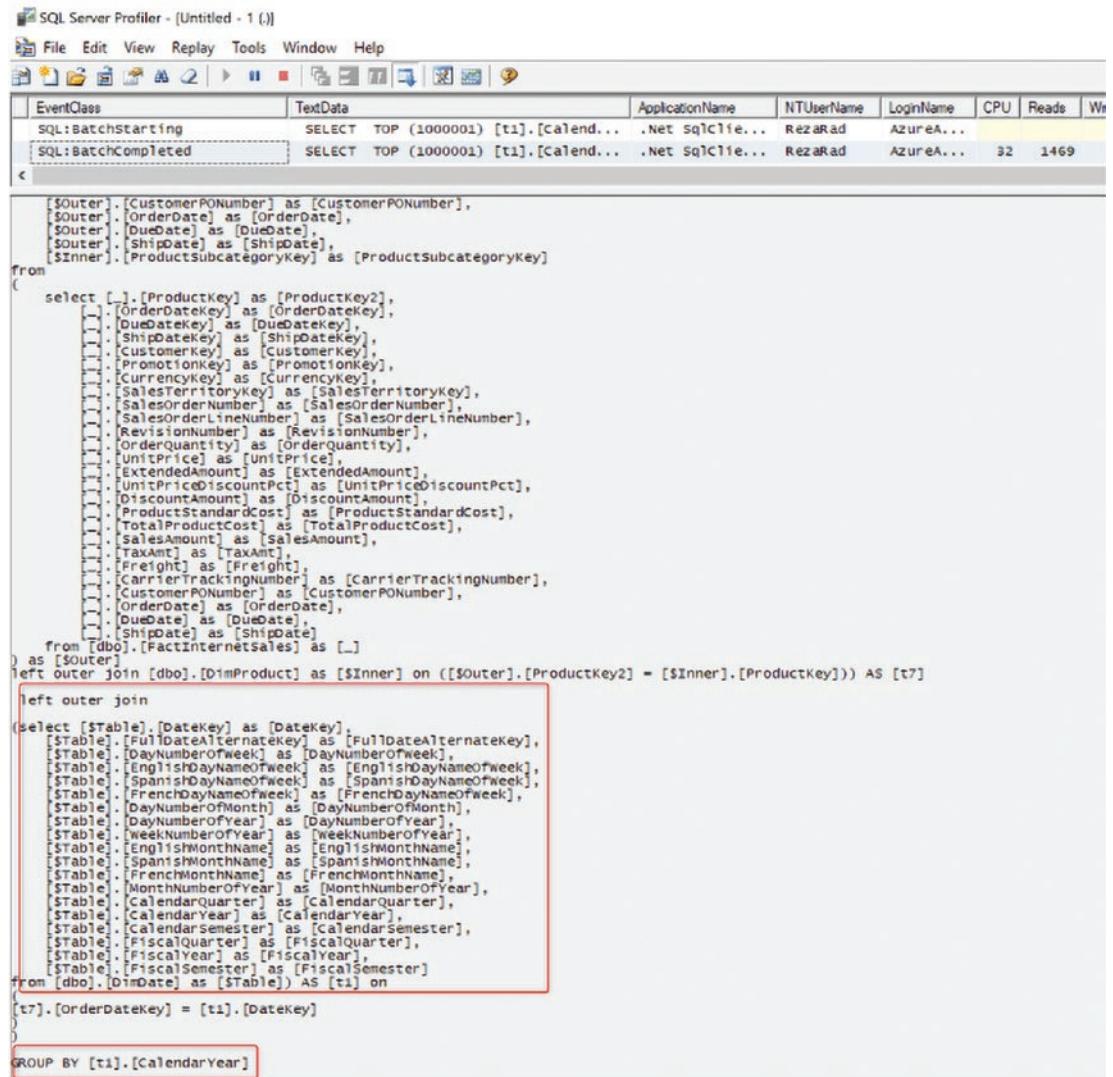
## Query from Multiple DirectQuery Tables

If you have multiple tables with DirectQuery storage mode, the result will be a query sent to the data source for that combination. Figure 17-26 includes a column chart by SalesAmount from FactInternetSales (DirectQuery) and CalendarYear from DimDate (DirectQuery).



**Figure 17-26.** Sample column chart by SalesAmount from FactInternetSales (DirectQuery) and CalendarYear from DimDate (DirectQuery)

Figure 17-27 reveals the query that is sent.



The screenshot shows the SQL Server Profiler interface with a single event class entry. The event is 'BatchStarting' and the text data is a complex T-SQL query. The query starts by selecting from the 'FactInternetSales' table, which includes joins to 'DimProduct' and 'DimDate'. It then performs a left outer join to 'DimDate' again, filtering by 'OrderDateKey'. Finally, it groups by 'CalendarYear'. A red box highlights the section of the query where it joins 'DimDate' again and filters by 'OrderDateKey'.

```

SQL Server Profiler - [Untitled - 1 ()]
File Edit View Replay Tools Window Help
EventClass TextData ApplicationName NTUserName LoginName CPU Reads Writes
SQL:BatchStarting SELECT TOP (1000001) [ti].[calend... .Net SqlClient... RezaRad AzureA...
SQL:BatchCompleted SELECT TOP (1000001) [ti].[calend... .Net SqlClient... RezaRad AzureA... 32 1469

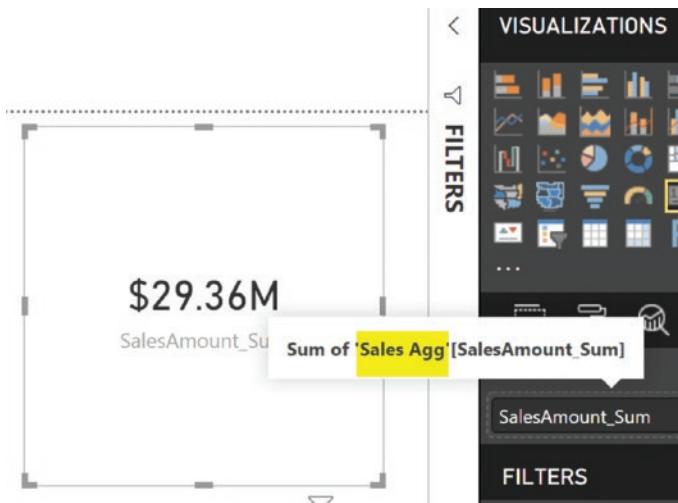

[$outer].[CustomerPONumber] as [CustomerPONumber],
[$outer].[OrderDate] as [OrderDate],
[$outer].[DueDate] as [DueDate],
[$outer].[ShipDate] as [ShipDate],
[$inner].[ProductSubcategoryKey] as [ProductSubcategoryKey]
from
(
    select [...].[ProductKey] as [ProductKey2],
    [...].[OrderDateKey] as [OrderDateKey],
    [...].[DueDateKey] as [DueDateKey],
    [...].[ShipDateKey] as [ShipDateKey],
    [...].[CustomerKey] as [CustomerKey],
    [...].[PromotionKey] as [PromotionKey],
    [...].[CurrencyKey] as [CurrencyKey],
    [...].[SalesTerritoryKey] as [SalesTerritoryKey],
    [...].[SalesOrderNumber] as [SalesOrderNumber],
    [...].[SalesOrderLineNumber] as [SalesOrderLineNumber],
    [...].[RevisionNumber] as [RevisionNumber],
    [...].[OrderQuantity] as [OrderQuantity],
    [...].[UnitPrice] as [UnitPrice],
    [...].[ExtendedAmount] as [ExtendedAmount],
    [...].[UnitPriceDiscountPct] as [UnitPriceDiscountPct],
    [...].[DiscountAmount] as [DiscountAmount],
    [...].[ProductStandardCost] as [ProductStandardCost],
    [...].[TotalProductCost] as [TotalProductCost],
    [...].[SalesAmount] as [SalesAmount],
    [...].[TaxAmt] as [TaxAmt],
    [...].[Freight] as [Freight],
    [...].[CarrierTrackingNumber] as [CarrierTrackingNumber],
    [...].[CustomerPONumber] as [CustomerPONumber],
    [...].[OrderDate] as [OrderDate],
    [...].[DueDate] as [DueDate],
    [...].[ShipDate] as [ShipDate]
    from [dbo].[FactInternetSales] as [...]
) as [$outer]
left outer join [dbo].[DimProduct] as [$inner] on ([$outer].[ProductKey2] = [$inner].[ProductKey])) AS [t7]
left outer join
(
select [...].[DateKey] as [DateKey],
    [...].[FullDateAlternateKey] as [FullDateAlternateKey],
    [...].[DayNumberOfWeek] as [DayNumberOfWeek],
    [...].[EnglishDayNameOfWeek] as [EnglishDayNameOfWeek],
    [...].[SpanishDayNameOfWeek] as [SpanishDayNameOfWeek],
    [...].[FrenchDayNameOfWeek] as [FrenchDayNameOfWeek],
    [...].[DayNumberOfMonth] as [DayNumberOfMonth],
    [...].[DayNumberOfYear] as [DayNumberOfYear],
    [...].[WeekNumberOfYear] as [WeekNumberOfYear],
    [...].[EnglishMonthName] as [EnglishMonthName],
    [...].[SpanishMonthName] as [SpanishMonthName],
    [...].[FrenchMonthName] as [FrenchMonthName],
    [...].[MonthNumberOfYear] as [MonthNumberOfYear],
    [...].[CalendarQuarter] as [CalendarQuarter],
    [...].[CalendarYear] as [CalendarYear],
    [...].[CalendarSemester] as [CalendarSemester],
    [...].[FiscalQuarter] as [FiscalQuarter],
    [...].[FiscalYear] as [FiscalYear],
    [...].[FiscalSemester] as [FiscalSemester]
from [dbo].[DimDate] as [...]) AS [ti] on
[t7].[OrderDateKey] = [ti].[DateKey]
)
GROUP BY [ti].[CalendarYear]


```

*Figure 17-27. Query sent to the data source*

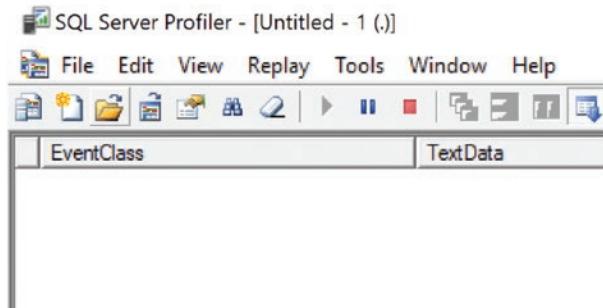
## Caution: Combining DirectQuery and Import

So far, with these examples, everything happens as expected. Nothing strange happens. However, combining fields from DirectQuery sourced tables with Import storage mode tables would be strange. As an example, the Sales Agg table is an Import Data table. The purpose of creating this table is to query faster from the in-memory structure. If you create a visual that contains something from the Sales Agg table, as shown in Figure 17-28, no query will be sent to the data source. It all happens in memory, as expected.



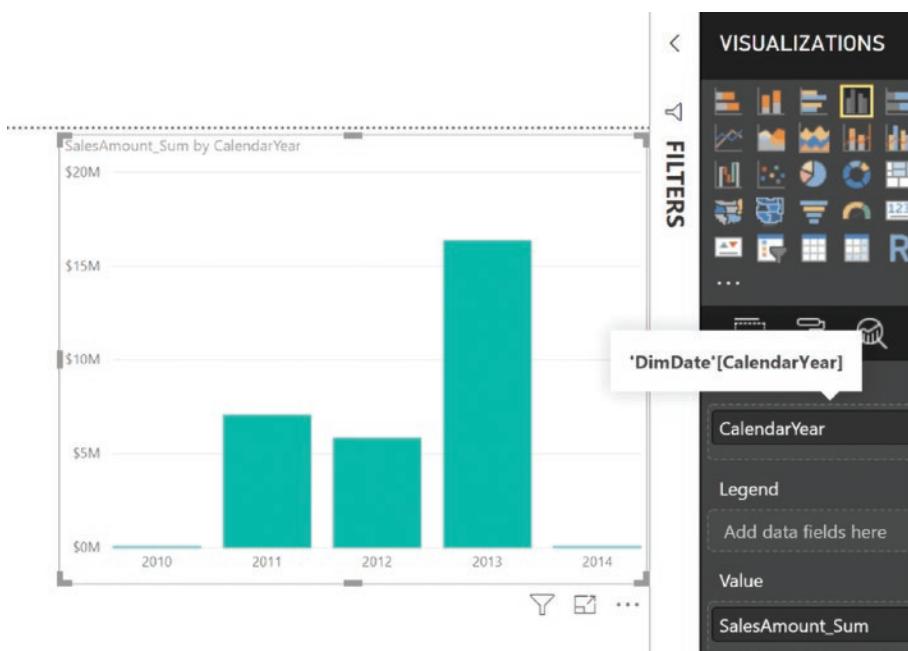
**Figure 17-28.** The Sales Agg table visual will not trigger a query

As Figure 17-29 shows, SQL Profiler doesn't catch any queries sent to the data source.



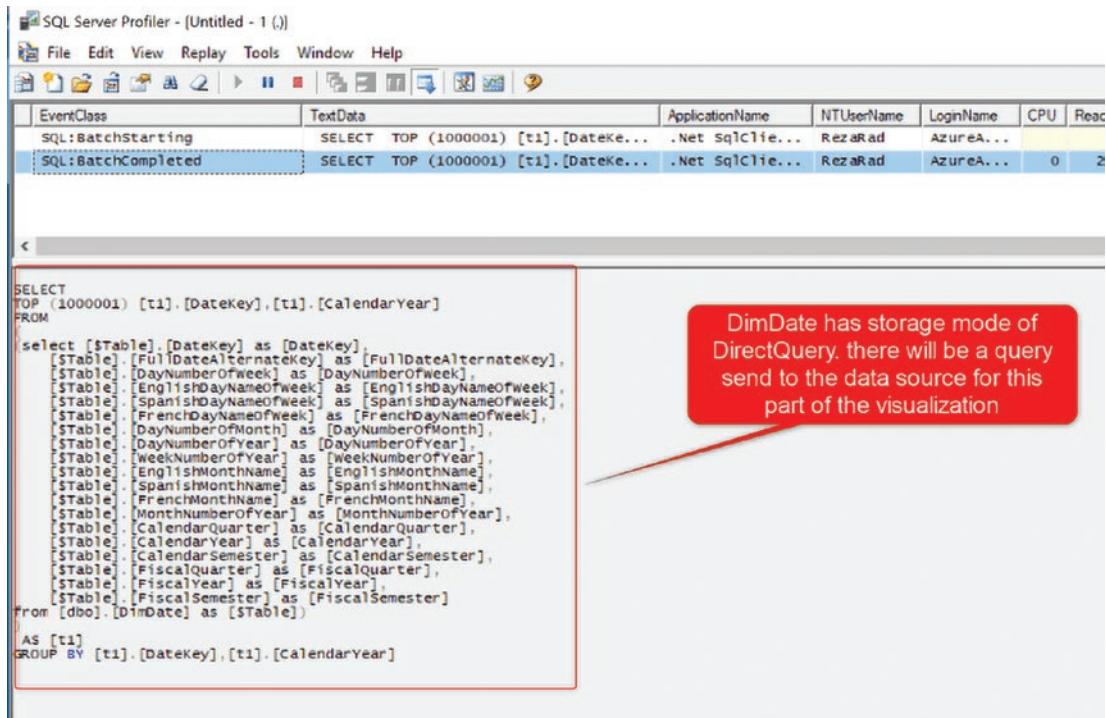
**Figure 17-29.** No queries are tracked

However, the behavior is different if, as shown in Figure 17-30, you have a visual that gets data from the Sales Agg table and a DirectQuery table such as DimDate.



**Figure 17-30.** Visual of the Sales Agg table and a DirectQuery table

This time, you see a T-SQL query tracked in SQL Profiler, querying the DimDate in the SQL Server database (see Figure 17-31).



**Figure 17-31.** T-SQL query tracked in SQL Profiler

This is not what you would expect to see. The purpose of the Sales Agg table is to speed up the process from DirectQuery mode, but you are still querying the DimDate from the database. So, what is the solution? Do you change the storage mode of DimDate to Import Data? If you do that, what happens to the connection between DimDate and FactInternetSales? You want that connection to work as DirectQuery, of course.

Now that you learned about the challenge, it is a good time to talk about the third storage mode, Dual.

## Dual Storage Mode

The Dual storage mode is built to cover a scenario like the one just discusses. With Dual storage mode, one table can act either as DirectQuery or Import Data, respective to the relationship to other tables. Dual storage mode is the secret sauce of composite mode and aggregation in Power BI. Let's see how this mode works.

If you change the storage mode of DimDate to Import Data, you get the issue of sending the query to the data source when you are querying FactInternetSales. If you change the storage mode of DimDate to DirectQuery, then even for Sales Agg, you are sending the query to the data source. The solution is to change the storage mode of the common dimension tables to Dual.

When you set the storage mode of a table such as DimDate to Dual, you get a warning that this process may take some time (see Figure 17-32). By changing the storage mode to Dual, you get a copy of that table in memory.

## Storage mode

X

Setting the storage mode to Dual has the following implications.

Please consider them carefully before proceeding.

This operation will refresh tables set to Dual, which may take time depending on factors such as data volume.

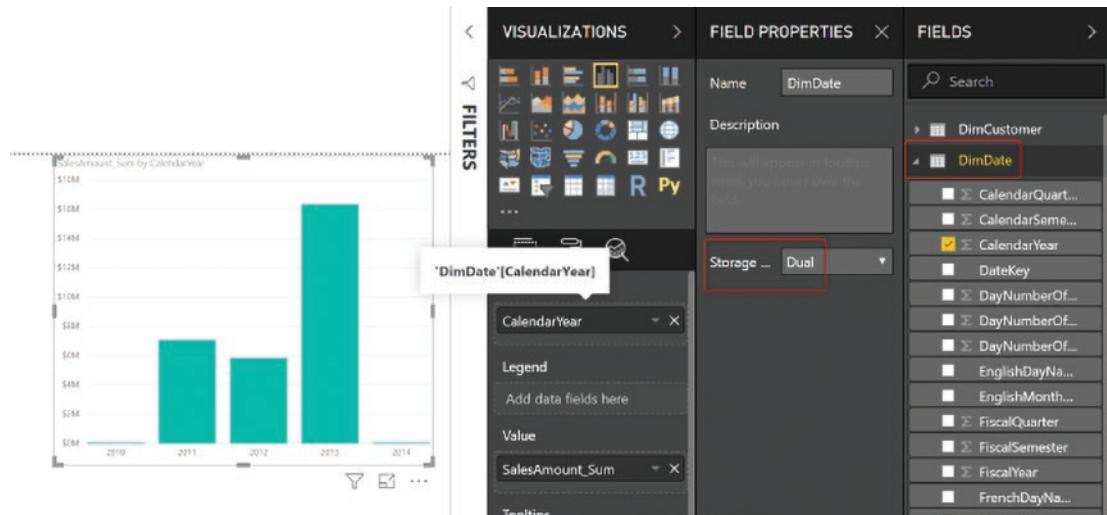
[Learn more about setting storage mode](#)



**Figure 17-32.** Setting storage to Dual triggers a warning

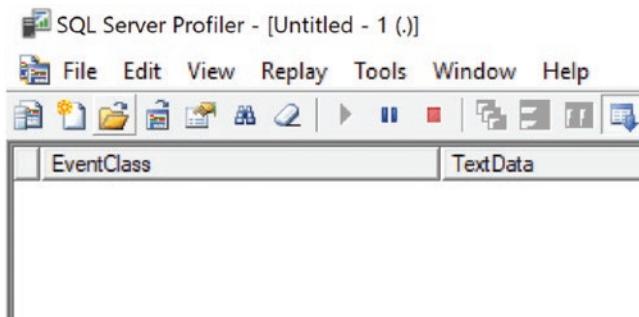
The copy of that table in memory acts like an Import table. There will be another version of that table that works through DirectQuery, though. It looks like you have two identical tables, one for Import Data and another for DirectQuery.

What would be the impact of having two tables on reporting? The same visual you had before in the previous example is now working through the in-memory engine, as shown in Figure 17-33.



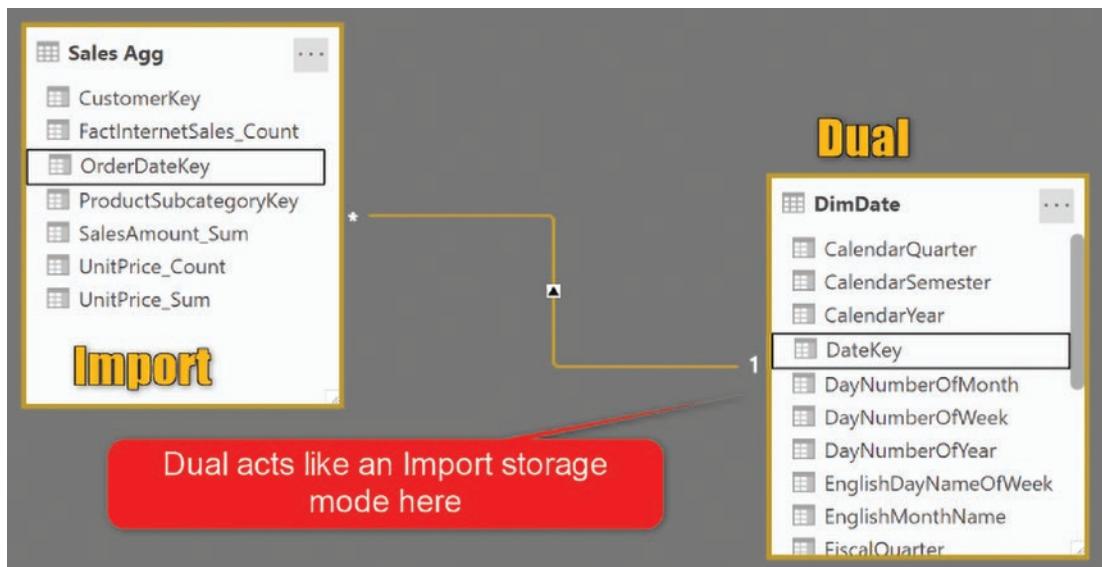
**Figure 17-33.** Dual storage mode for the DimDate table

As you can see in Figure 17-34, no queries are sent to the data source for this table.



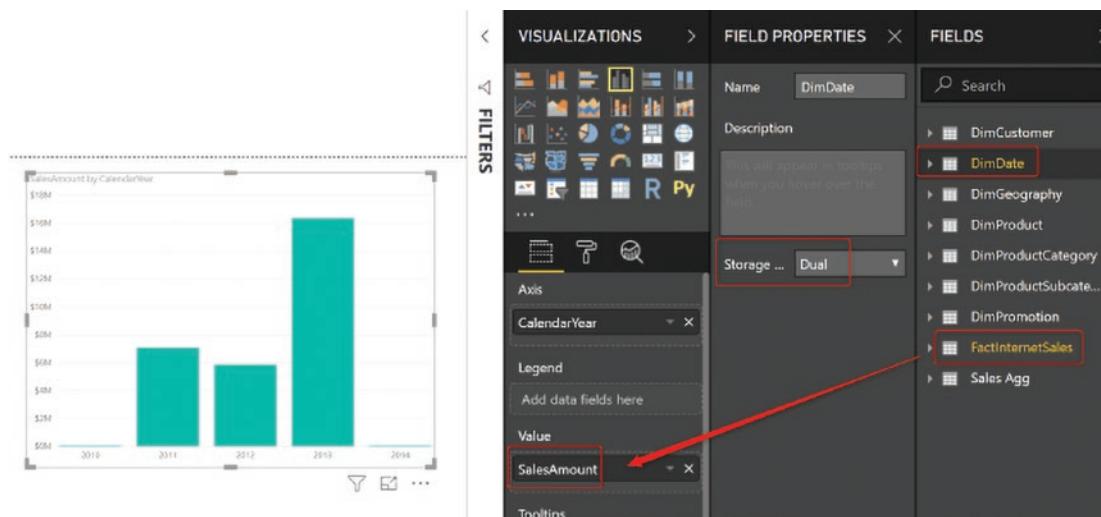
**Figure 17-34.** No queries are tracked

Because this table is connected to an Import table (Sales Agg), the Dual storage mode acts like an Import table storage (see Figure 17-35). Everything is queried from memory.



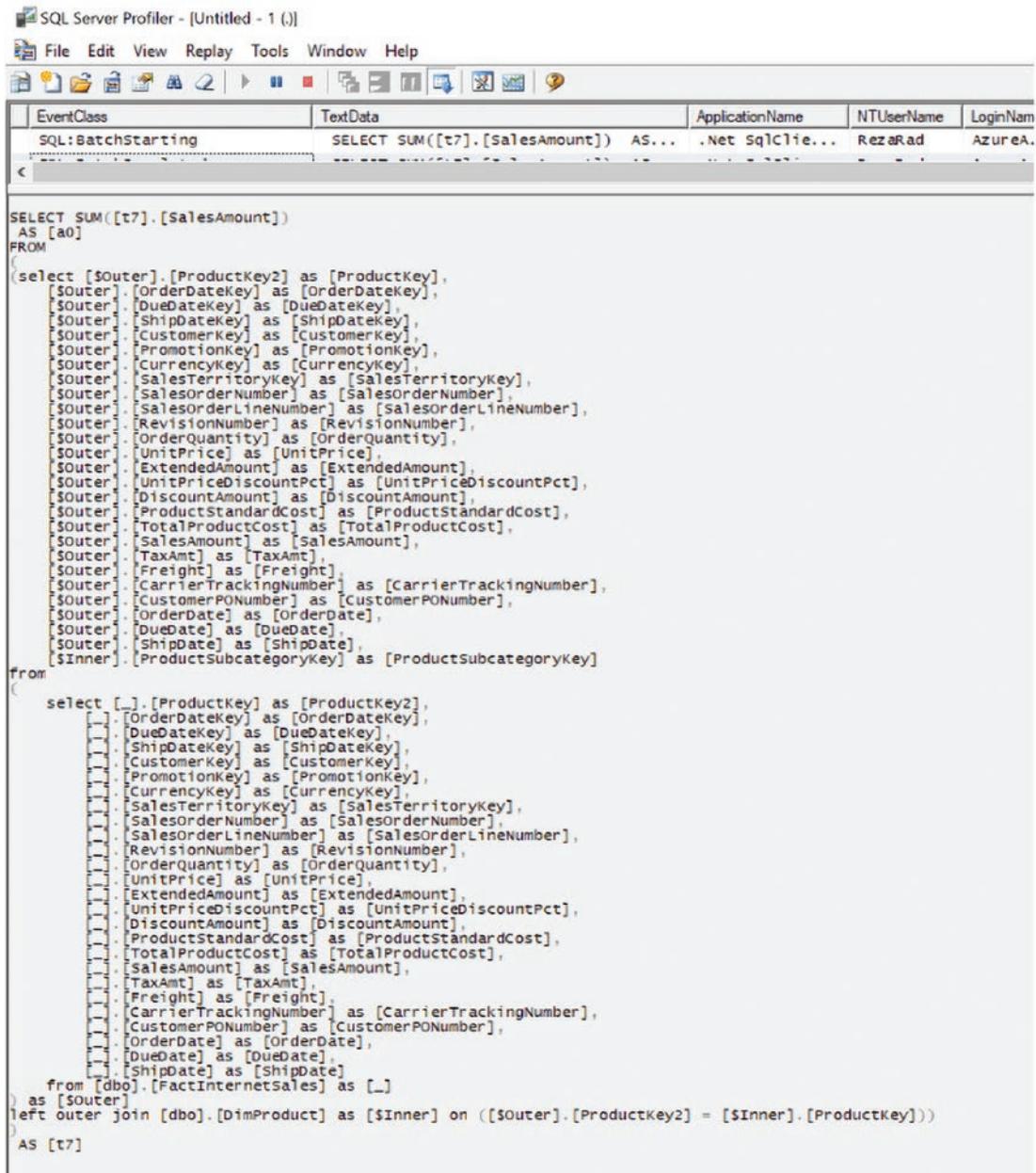
**Figure 17-35.** Dual storage behaves like an import table storage

Say you have the Dual storage mode table, used in a visualizing alongside a DirectQuery table, similar to Figure 17-36.



**Figure 17-36.** Visualization of data from a table using Dual storage mode

This visualization will send a query to the data source, as you can see in the SQL Profiler track featured in Figure 17-37.



The screenshot shows the SQL Server Profiler interface with a single tracked event. The event class is 'SQL:BatchStarting' and the text data is a complex T-SQL query. The application name is '.Net SqlClient-DataAccess Library', the NT user name is 'RezaRad', and the login name is 'AzureAd'. The query itself is a multi-step aggregation and join operation on the FactInternetSales table.

```

SQL:BatchStarting
SELECT SUM([t7].[SalesAmount]) AS... .Net SqlClient-DataAccess Library RezaRad AzureAd
SELECT SUM([t7].[SalesAmount])
AS [a0]
FROM
(
select [$outer].[ProductKey2] as [ProductKey],
[$outer].[OrderDateKey] as [OrderDateKey],
[$outer].[DueDateKey] as [DueDateKey],
[$outer].[ShipDateKey] as [ShipDateKey],
[$outer].[CustomerKey] as [CustomerKey],
[$outer].[PromotionKey] as [PromotionKey],
[$outer].[CurrencyKey] as [CurrencyKey],
[SalesTerritoryKey] as [SalesTerritoryKey],
[$outer].[SalesorderNumber] as [SalesorderNumber],
[$outer].[SalesorderLineNumber] as [SalesorderLineNumber],
[$outer].[RevisionNumber] as [RevisionNumber],
[$outer].[OrderQuantity] as [OrderQuantity],
[$outer].[UnitPrice] as [UnitPrice],
[ExtendedAmount] as [ExtendedAmount],
[UnitPriceDiscountPct] as [UnitPriceDiscountPct],
[DiscountAmount] as [DiscountAmount],
[ProductstandardCost] as [ProductstandardCost],
[TotalProductCost] as [TotalProductCost],
[SalesAmount] as [SalesAmount],
[TaxAmt] as [TaxAmt],
[Freight] as [Freight],
[CarrierTrackingNumber] as [CarrierTrackingNumber],
[CustomerPONumber] as [CustomerPONumber],
[OrderDate] as [OrderDate],
[DueDate] as [DueDate],
[$outer].[ShipDate] as [ShipDate],
[$inner].[ProductSubcategoryKey] as [ProductSubcategoryKey]
from
(
    select [...].[ProductKey] as [ProductKey2],
    [...].[OrderDateKey] as [OrderDateKey],
    [...].[DueDateKey] as [DueDateKey],
    [...].[ShipDateKey] as [ShipDateKey],
    [...].[CustomerKey] as [CustomerKey],
    [...].[PromotionKey] as [PromotionKey],
    [...].[CurrencyKey] as [CurrencyKey],
    [SalesTerritoryKey] as [SalesTerritoryKey],
    [SalesorderNumber] as [SalesorderNumber],
    [SalesorderLineNumber] as [SalesorderLineNumber],
    [RevisionNumber] as [RevisionNumber],
    [OrderQuantity] as [OrderQuantity],
    [UnitPrice] as [UnitPrice],
    [ExtendedAmount] as [ExtendedAmount],
    [UnitPriceDiscountPct] as [UnitPriceDiscountPct],
    [DiscountAmount] as [DiscountAmount],
    [ProductstandardCost] as [ProductstandardCost],
    [TotalProductCost] as [TotalProductCost],
    [SalesAmount] as [SalesAmount],
    [TaxAmt] as [TaxAmt],
    [Freight] as [Freight],
    [CarrierTrackingNumber] as [CarrierTrackingNumber],
    [CustomerPONumber] as [CustomerPONumber],
    [OrderDate] as [OrderDate],
    [DueDate] as [DueDate],
    [...].[ShipDate] as [ShipDate]
    from [dbo].[FactInternetSales] as [...]
) as [$outer]
left outer join [dbo].[DimProduct] as [$inner] on ([$outer].[ProductKey2] = [$inner].[ProductKey]))
AS [t7]

```

**Figure 17-37.** SQL Profiler tracks queries

In this scenario, the Dual storage mode acts like a DirectQuery because it comes with a connection to a DirectQuery sourced table, as Figure 17-38 demonstrates.

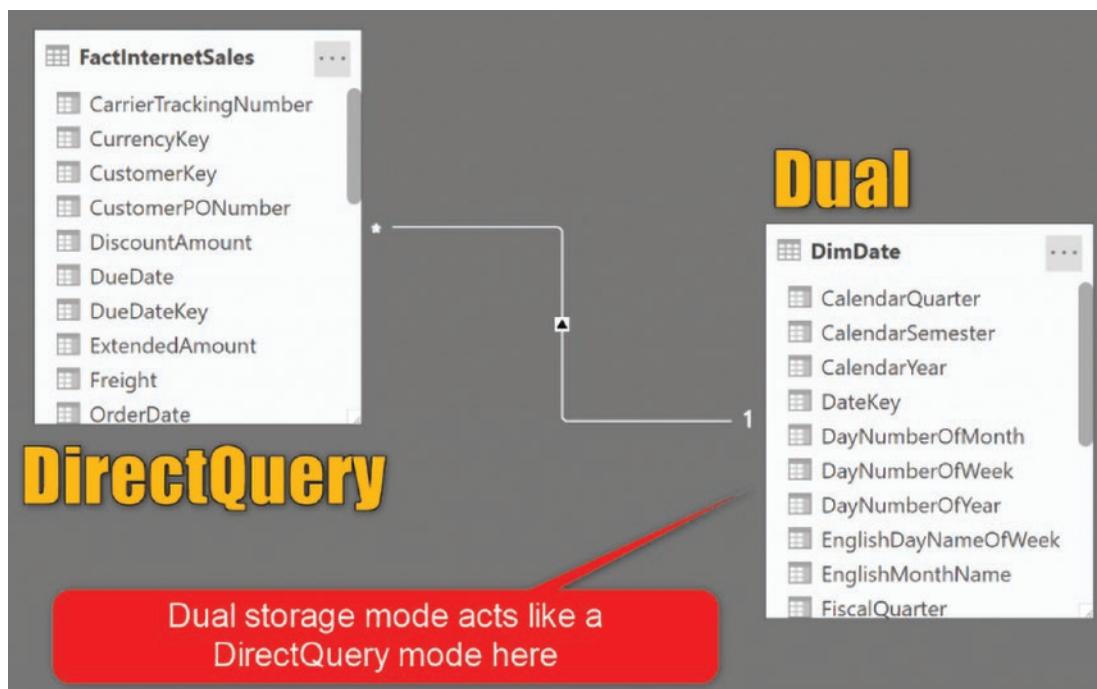


Figure 17-38. Dual mode behaves like DirectQuery mode

Now you can see why this mode is called Dual. Sometimes it acts like Import and sometimes like DirectQuery, depending on the table that is combined through the visualization. See Figure 17-39.



**Figure 17-39.** Dual mode can behave in two ways

If you don't care about the details, and you just want to know how to get the aggregations working, the answer is easy. All tables connected to both Sales Agg (the aggregation table, which is Import), and FactInternetSales (the big fact table, which is DirectQuery) should be set to the Dual storage mode. This way, they can act both ways, depending on the situation.

---

All tables connected to Import table (aggregation) and DirectQuery table (big fact table) should be set to Dual storage mode.

---

In this example, Dual storage mode is used with DimDate, DimProductSubcategory, and DimCustomer. However, when you apply Dual storage mode to DimCustomer, a message pops up saying that DimGeography will also be set as Storage mode (see Figure 17-40). That is correct; other tables that are related to this through a many-to-one relationship need to be set to Dual too.

## Storage mode

X

Setting the storage mode to Dual has the following implications.  
Please consider them carefully before proceeding.

This operation will refresh tables set to Dual, which may take time depending on factors such as data volume.

Weak relationships may be introduced by this change.

The number of weak relationships can be reduced by setting the following tables to Dual.

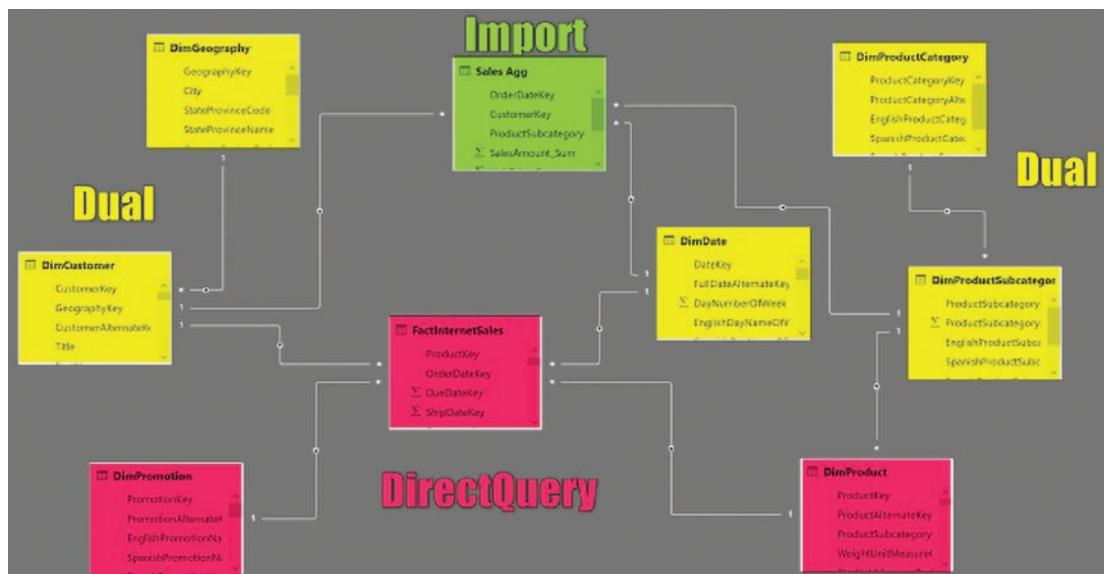
- DimGeography
- Set affected tables to dual

[Learn more about setting storage mode](#)

OK Cancel

**Figure 17-40.** Storage mode alert

The same process happens with DimProductSubcategory and DimProductCategory. In the sample diagram and model, all tables selected in Figure 17-41 are set to Dual storage mode.



**Figure 17-41.** Tables set to Dual storage mode

## Step 3: Configure Aggregation Functions and Test Aggregations in Action

The aggregated table is created and has proper relationships and storage modes. However, Power BI is unaware of this aggregation (remember you can create the aggregation outside of Power BI with the Group By clause in T-SQL). You have to let Power BI know about this aggregation. That is why you need to set up and configure it.

Right-click the Sales Agg table and select Manage Aggregations, as shown in Figure 17-42.

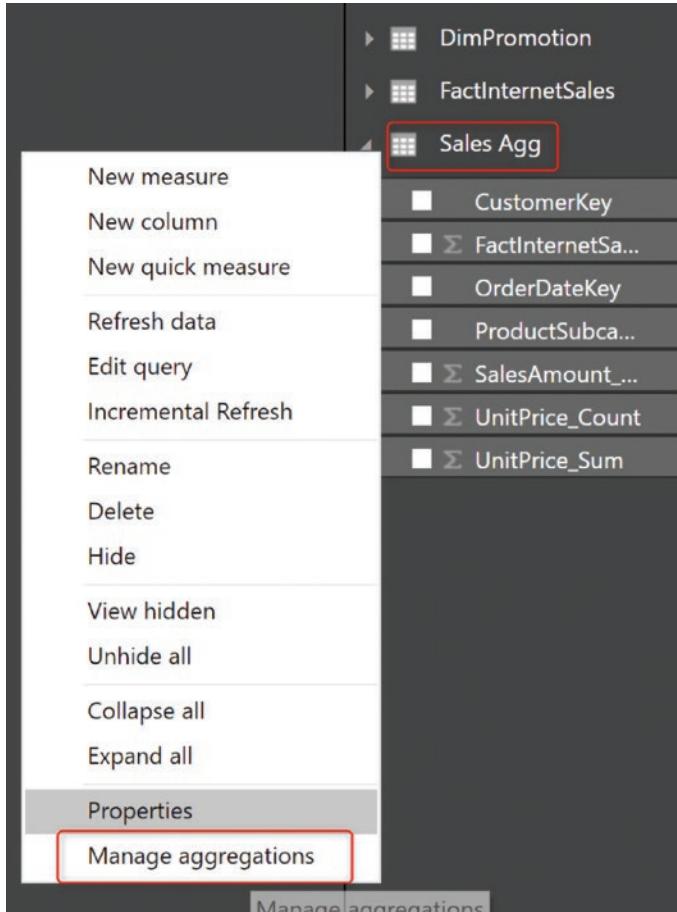
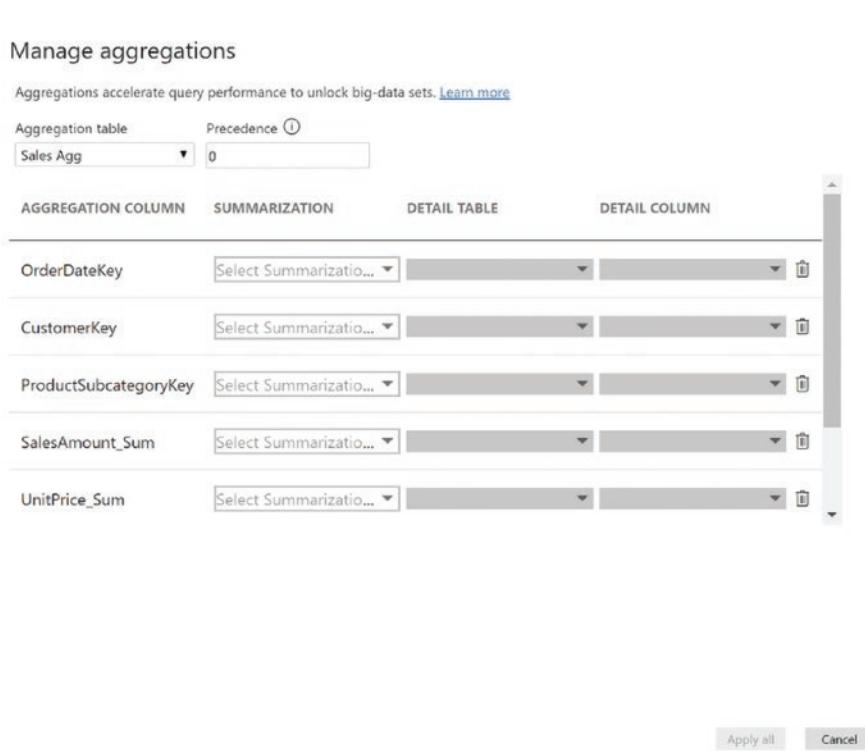


Figure 17-42. Managing aggregations

In the Manage Aggregations window, you can choose aggregation functions and fields that the function will be applied to (see Figure 17-43).



**Figure 17-43.** Selecting aggregations

This aggregation configuration should match the aggregation configuration on the aggregated table. What does that mean? When creating the aggregated table, you created a Group By with the setup shown in Figure 17-44.

## Group By

Basic     Advanced

Specify the columns to group by and one or more outputs.

Group by																				
OrderDateKey																				
CustomerKey																				
ProductSubcategoryKey																				
<input type="button" value="Add grouping"/>																				
<table border="1"> <thead> <tr> <th>New column name</th> <th>Operation</th> <th>Column</th> </tr> </thead> <tbody> <tr> <td>SalesAmount_Sum</td> <td>Sum</td> <td>SalesAmount</td> </tr> <tr> <td>UnitPrice_Sum</td> <td>Sum</td> <td>UnitPrice</td> </tr> <tr> <td>UnitPrice_Count</td> <td>Count Rows</td> <td></td> </tr> <tr> <td>FactInternetSales_Count</td> <td>Count Rows</td> <td></td> </tr> <tr> <td colspan="3"> <input type="button" value="Add aggregation"/> </td> </tr> </tbody> </table>			New column name	Operation	Column	SalesAmount_Sum	Sum	SalesAmount	UnitPrice_Sum	Sum	UnitPrice	UnitPrice_Count	Count Rows		FactInternetSales_Count	Count Rows		<input type="button" value="Add aggregation"/>		
New column name	Operation	Column																		
SalesAmount_Sum	Sum	SalesAmount																		
UnitPrice_Sum	Sum	UnitPrice																		
UnitPrice_Count	Count Rows																			
FactInternetSales_Count	Count Rows																			
<input type="button" value="Add aggregation"/>																				

**Figure 17-44.** Aggregation setup

As you can see in Figure 17-44, you have three fields that are the Group By fields—OrderDateKey, CustomerKey, and ProductSubcategoryKey. These fields should be marked as Group By on their respective related field in the FactInternetSales table (the original fact table). See the highlighted portion of Figure 17-45.

## Manage aggregations

Aggregations accelerate query performance to unlock big-data sets. [Learn more](#)

The screenshot shows the 'Manage aggregations' interface. At the top, there's a dropdown for 'Aggregation table' set to 'Sales Agg' and a 'Precedence' field with value '0'. Below this is a table with four columns: 'AGGREGATION COLUMN', 'SUMMARIZATION', 'DETAIL TABLE', and 'DETAIL COLUMN'. The first three columns have dropdown menus. The 'DETAIL COLUMN' column has a red border around its dropdown menu, which is also highlighted in yellow. The table rows are:

AGGREGATION COLUMN	SUMMARIZATION	DETAIL TABLE	DETAIL COLUMN
OrderDateKey	GroupBy	FactInternetSales	OrderDateKey
CustomerKey	GroupBy	FactInternetSales	CustomerKey
ProductSubcategoryKey	GroupBy	FactInternetSales	ProductSubcategory...

**Figure 17-45.** Setting Group By fields

Then you should set up aggregation functions according to the original grouping configuration. For SalesAmount\_Sum and UnitPrice\_Sum, the aggregation should be set as Sum to their respective fields in the FactInternetSales table, as shown in Figure 17-46.

## Manage aggregations

Aggregations accelerate query performance to unlock big-data sets. [Learn more](#)

The screenshot shows the 'Manage aggregations' interface. At the top, there's a dropdown for 'Aggregation table' set to 'Sales Agg' and a 'Precedence' field with value '0'. Below this is a table with four columns: 'CustomerKey', 'ProductSubcategoryKey', and two rows for aggregation functions. The 'CustomerKey' and 'ProductSubcategoryKey' rows have red borders around their dropdown menus. The table rows are:

CustomerKey	GroupBy	FactInternetSales	CustomerKey
ProductSubcategoryKey	GroupBy	FactInternetSales	ProductSubcategory...
SalesAmount_Sum	Sum	FactInternetSales	SalesAmount
UnitPrice_Sum	Sum	FactInternetSales	UnitPrice

**Figure 17-46.** Setting aggregation as Sum

### Rule #1: The Detail column must be the same data type of the grouped column to use the Sum function

At the time of creating the aggregated table, I mentioned that if you are using SUM as aggregation, the data type of the column after aggregation (SalesAmount\_Sum) should be the same data type as the original column (SalesAmount).

If you don't follow this rule, the column will be grayed out in the list of columns, and you cannot select it. Change the data type first, then come back to set it.

For the other two columns—`UnitPrice_Count` and `FactInternetSales_Count`—the aggregation is `Count` and the table is `FactInternetSales`, as Figure 17-47 demonstrates:

UnitPrice_Count	Count	FactInternetSales	UnitPrice	
FactInternetSales_Count	Count	FactInternetSales	Select value...	

**Figure 17-47.** Aggregation is set to Count

#### Rule #2: The data type of the column must be Whole Number (Integer) if you used the Count function

If you created any aggregated result using `Count`, that column's data type should be a whole number. Otherwise, the `Count` function in the manage aggregations will be grayed out. After configuring everything, your Manage Aggregations window should look like Figure 17-48.

### Manage aggregations

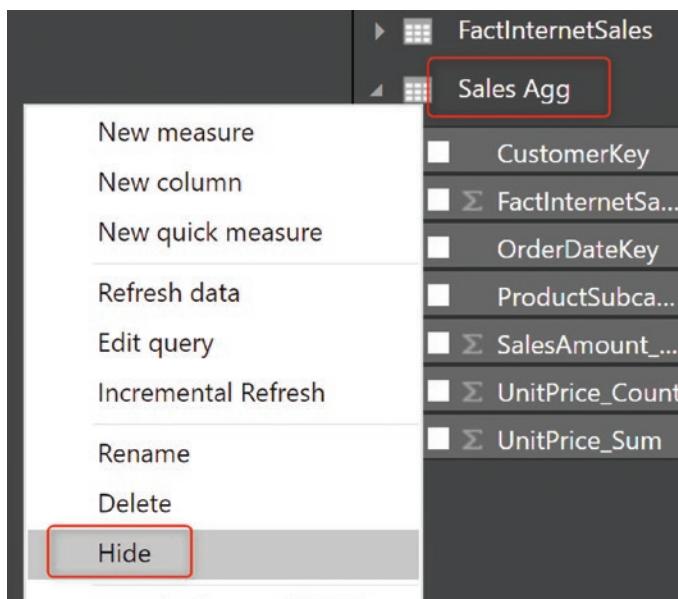
Aggregations accelerate query performance to unlock big-data sets. [Learn more](#)

AGGREGATION COLUMN	SUMMARIZATION	DETAIL TABLE	DETAIL COLUMN
OrderDateKey	GroupBy	FactInternetSales	OrderDateKey
CustomerKey	GroupBy	FactInternetSales	CustomerKey
ProductSubcategoryKey	GroupBy	FactInternetSales	ProductSubcategory...
SalesAmount_Sum	Sum	FactInternetSales	SalesAmount
UnitPrice_Sum	Sum	FactInternetSales	UnitPrice

**Figure 17-48.** The Manage Aggregations window, set up correctly

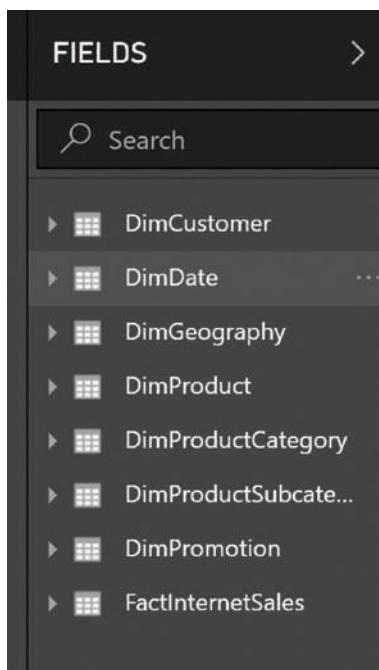
### Hiding the Aggregated Table

The final step is to hide the aggregated table! You created the aggregated table for Power BI to automatically switch from the DirectQuery big fact table to the small Import Data aggregated table. However, this is just behind the scenes. The user should know nothing about this process. As shown in Figure 17-49, hide the `Sales Agg` table to make this experience seamless from the user's point of view. Power BI will automatically do this in the recent version of the Power BI Desktop.



**Figure 17-49.** Hiding the aggregated table from the users

The user will see nothing about this table. The list of tables in the fields section contains only one fact table, called FactInternetSales. See Figure 17-50.



**Figure 17-50.** Aggregated table successfully hidden from view

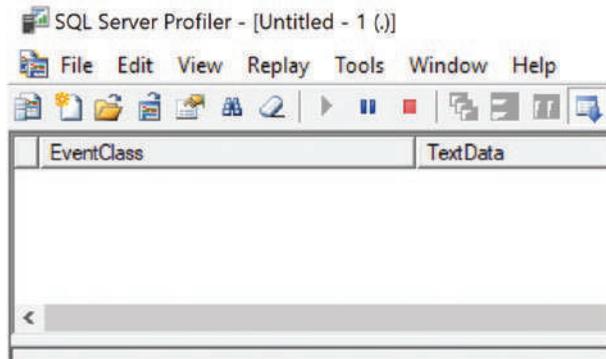
### Testing the Result

Now is the time to test how the aggregation works in action. The report in Figure 17-51 has all kinds of visualizations from FactInternetSales and SalesAmount sliced and diced by education and occupation (from DimCustomer), CalendarYear (from DimDate), ProductSubcategory, and ProductCategory.



**Figure 17-51.** Report with visualizations fetched from an aggregated table

I have SQL Profiler running all the time, and Figure 17-52 shows the result.



**Figure 17-52.** No query is sent to the DirectQuery source

This is how the aggregated table works behind the scenes. Since all of these visualizations are slicing and dicing the data by fields in the aggregated table (CustomerKey, OrderDateKey, and ProductSubcategoryKey), the result will always be automatically fetched from the aggregated table. Even though the Sales Agg table is not used in these visualizations.

If, however, I have a visualization that uses a different field that's not in the aggregated table, it is time to query from the data source. Figure 17-53 looks at every promotion.

SalesAmount	EnglishPromotionName
\$27,307,607.0825	No Discount
\$30,992.91	Touring-1000 Promotion
\$14,847	Touring-3000 Promotion
\$2,005,230.2282	Volume Discount 11 to 14
<b>\$29,358,677.2207</b>	

**Figure 17-53.** Visualization of a field that is not in aggregation

SQL Profiler tracks the query, as shown in Figure 17-54.

**Figure 17-54.** Query tracked in SQL Profiler

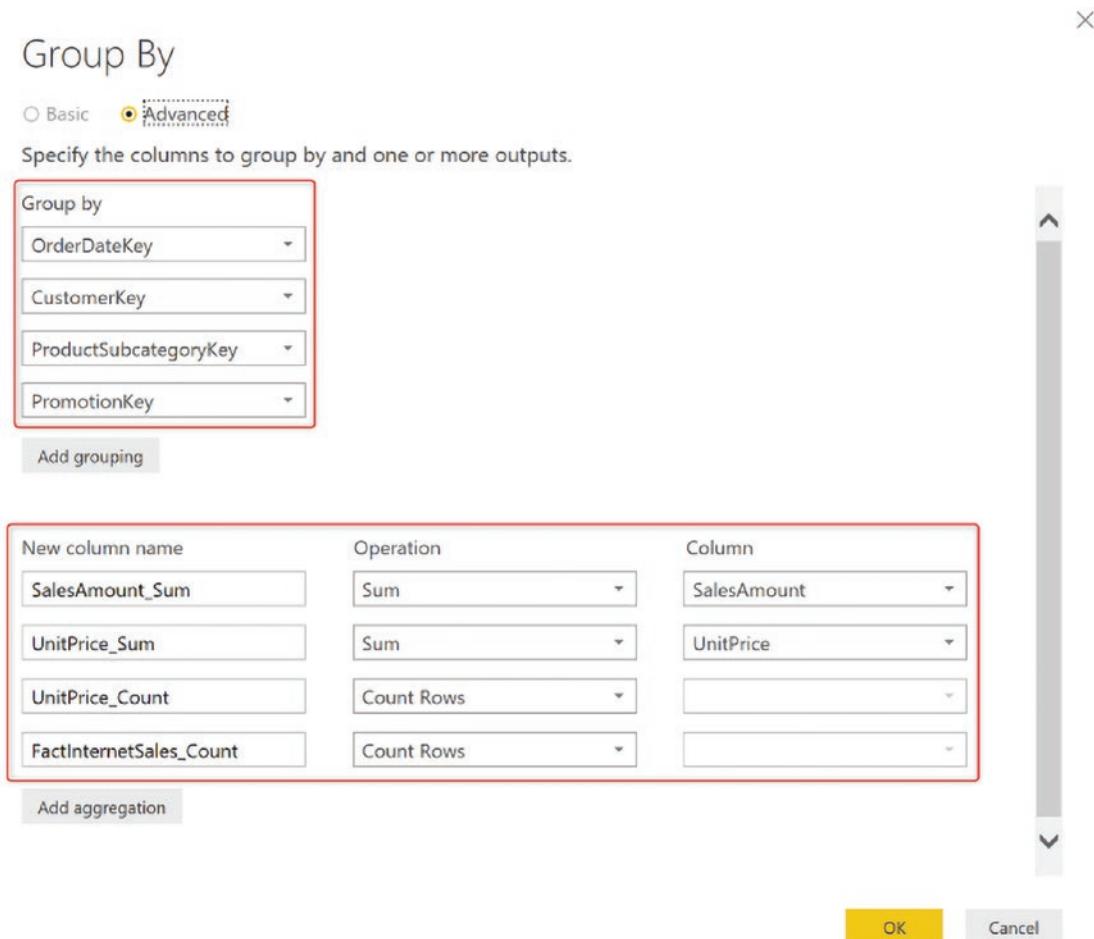
So here you go; you have an aggregation that speeds up the performance of the DirectQuery sourced table. Nothing stops you from creating multiple levels of aggregation. It is recommended to do so, especially in scenarios where a different combination of fields will be used by different users or visuals.

## Multiple Layers of Aggregations

Aggregations speed up the model. However, the aggregated table is not just one table; it can be multiple layers of aggregations—aggregation by date, aggregation by date and product, aggregation by date, product, and customer. Multiple layers ensure that you always have the best performance result possible, and you only query the DirectQuery data source for the most atomic requests. Let's see how this process is possible and helpful.

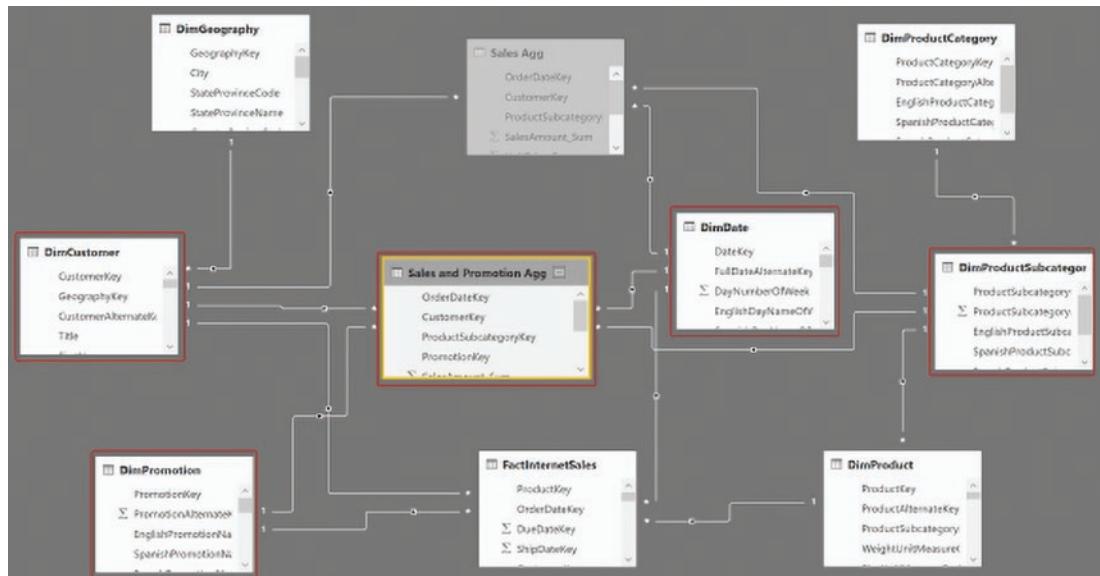
## Second Aggregation Layer

The second aggregation layer added to this includes promotion options. Figure 17-55 shows the Group By settings for this second aggregation table in Power Query.



**Figure 17-55.** Sample group settings for another aggregation table

You have to load the table into Power BI and create relationships to DimDate, DimCustomer, DimProductSubcategory, and DimPromotion. See Figure 17-56.



**Figure 17-56.** Sample relationship diagram

Set the storage mode of DimPromotion to Dual, as shown in Figure 17-57. It is important to use Dual storage mode. With this setup, Power BI for aggregation-based analysis uses the in-memory copy of DimPromotion, and for the atomic transaction levels, it uses the DirectQuery version of it.

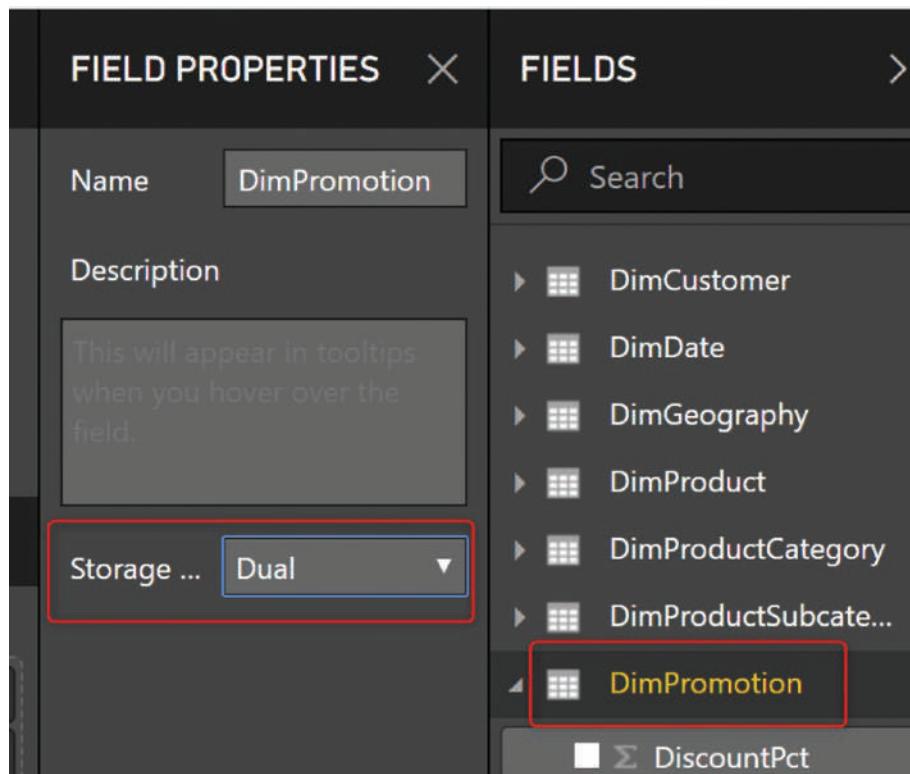


Figure 17-57. Setting Dual storage mode

After setting up relationships and storage modes, you need to manage the aggregations on the Sales and Promotion Agg tables (this is the second aggregation table). See Figure 17-58.

The screenshot shows the 'Manage aggregations' dialog box. At the top, there's a header with a close button (X) and a sub-header 'Manage aggregations'. Below that, a note says 'Aggregations accelerate query performance to unlock big-data sets. [Learn more](#)'. The main area is a table with four columns:

AGGREGATION COLUMN	SUMMARIZATION	DETAIL TABLE	DETAIL COLUMN
OrderDateKey	GroupBy	FactInternetSales	OrderDateKey
CustomerKey	GroupBy	FactInternetSales	CustomerKey
ProductSubcategoryKey	GroupBy	FactInternetSales	ProductSubcategory...
PromotionKey	GroupBy	FactInternetSales	PromotionKey
SalesAmount_Sum	Sum	FactInternetSales	SalesAmount

At the bottom right of the dialog are two buttons: 'Apply all' (yellow background) and 'Cancel'.

**Figure 17-58.** Managing the aggregations

Everything in Figure 17-58 is similar to the aggregation setup you saw for the Sales Agg table. For this example, the only addition is the Group By action on PromotionKey.

## Precedence Setup

When you have multiple layers of aggregation, you must set their precedence. The aggregated table you want to take the highest priority should have the highest precedence. In this case, Sales Agg can be 1, and Sales and Promotion Agg can be 0.

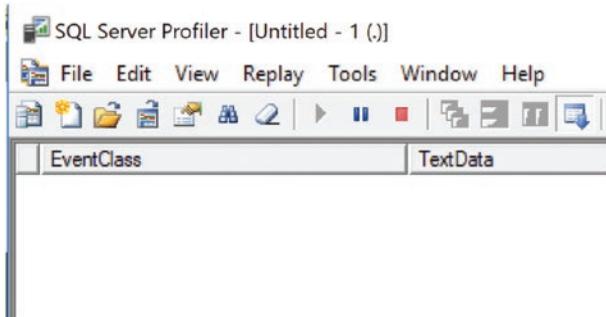
### Testing the result

With this setup, you can have visualizations that use promotions, Customer, Date, and ProductSubcategory and are still sourced from the aggregation. Figure 17-59 offers one example.

FirstName	LastName	EnglishPromotionName	SalesAmount	FullDateAlternateKey	EnglishProductSubcategoryName	EnglishProduct	Y	Z	?
Aaron	Adams	No Discount	\$34.99	Sunday, 28 April 2013	Helmets	Accessories			
Aaron	Adams	No Discount	\$53.99	Sunday, 28 April 2013	Jerseys	Clothing			
Aaron	Adams	No Discount	\$28.98	Sunday, 28 April 2013	Tires and Tubes	Accessories			
Aaron	Alexander	No Discount	\$69.99	Friday, 13 December 2013	Shorts	Clothing			
Aaron	Allen	No Discount	\$3,399.99	Friday, 2 December 2011	Mountain Bikes	Bikes			
Aaron	Baker	No Discount	\$49.99	Monday, 9 September 2013	Jerseys	Clothing			
Aaron	Baker	No Discount	\$1,700.99	Monday, 9 September 2013	Road Bikes	Bikes			
Aaron	Bryant	No Discount	\$8.99	Wednesday, 24 April 2013	Caps	Clothing			
Aaron	Bryant	No Discount	\$49.99	Wednesday, 24 April 2013	Jerseys	Clothing			
Aaron	Bryant	No Discount	\$34.99	Friday, 26 July 2013	Helmets	Accessories			
Aaron	Bryant	No Discount	\$39.99	Friday, 26 July 2013	Tires and Tubes	Accessories			
Aaron	Butler	No Discount	\$14.98	Thursday, 26 December 2013	Bottles and Cages	Accessories			
Aaron	Campbell	No Discount	\$34.99	Tuesday, 10 September 2013	Helmets	Accessories			
Aaron	Campbell	No Discount	\$1,120.49	Tuesday, 10 September 2013	Road Bikes	Bikes			
Aaron	Carter	No Discount	\$34.99	Sunday, 22 December 2013	Helmets	Accessories			
Aaron	Carter	No Discount	\$4.99	Sunday, 22 December 2013	Tires and Tubes	Accessories			
Aaron	Chen	No Discount	\$34.99	Saturday, 13 July 2013	Helmets	Accessories			
Aaron	Chen	No Discount	\$4.99	Saturday, 13 July 2013	Tires and Tubes	Accessories			
Aaron	Coleman	No Discount	\$21.98	Thursday, 5 December 2013	Fenders	Accessories			
Aaron	Coleman	No Discount	\$34.99	Thursday, 5 December 2013	Helmets	Accessories			
Aaron	Coleman	No Discount	\$4.99	Thursday, 5 December 2013	Tires and Tubes	Accessories			
Aaron	Collins	No Discount	\$3,578.27	Tuesday, 25 January 2011	Road Bikes	Bikes			
Aaron	Collins	No Discount	\$34.99	Saturday, 16 November 2013	Helmets	Accessories			
Aaron	Collins	No Discount	\$49.99	Saturday, 16 November 2013	Jerseys	Clothing			
Total			\$29,358,677.2207						

**Figure 17-59.** Sourced from the aggregation

This kind of visualization will not send a query to the DirectQuery source, as you can see in Figure 17-60.

**Figure 17-60.** Query not sent to the source

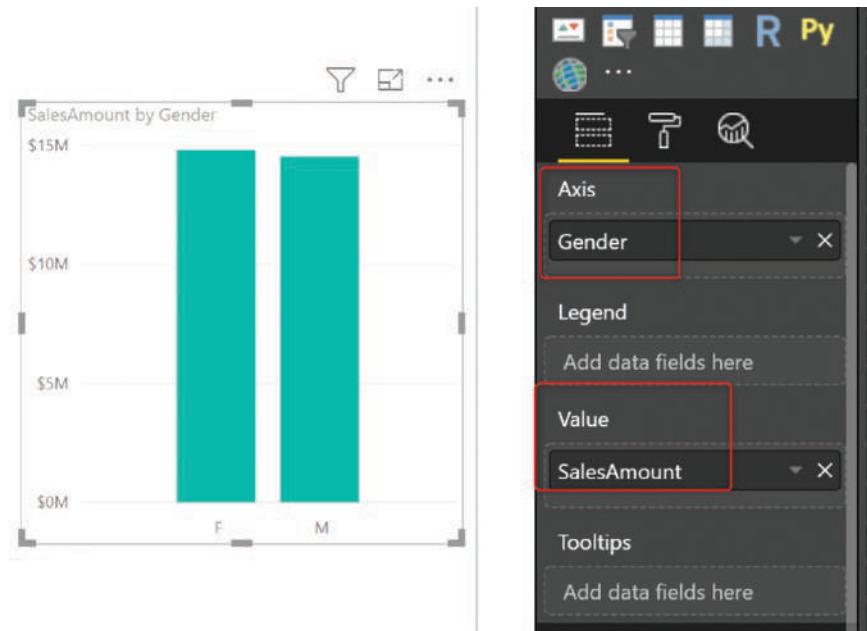
## How about the Precedence of Execution?

When you have multiple aggregation layers, determining which one runs first is important. As an example, let's assume some numbers. These are not real numbers; these just help you understand the scenario. The big fact table (FactInternetSales) uses DirectQuery mode and has 250 million rows. And the Sales Agg table has only 10,000 rows. But the Sales and Promotion Agg table has 1,000,000 rows. In such a scenario, when you are querying something that can be answered with the table with 10K rows (Sales Agg),

you have to use that, because it would be much faster than the table with 1M rows (Sales and Promotion Agg). Therefore, you need to set up the precedence of execution. Smaller aggregation tables should be the source of analysis first.

## Query Hits the First Aggregated Table in Memory

Figure 17-61 shows the Power BI page using Gender (from DimCustomer) and SalesAmount (from FactInternetSales).



**Figure 17-61.** Sample visual

The visual in Figure 17-61 will query the aggregated table behind the scenes, not the FactInternetSales table. In this case, it will query the Sales Agg table. Figure 17-62 shows the behind-the-scenes query sent to the VertiPaq engine (result generated from SQL Profiler).

Query Begin	s = DATAQUERY	2018-12-11 09:16:25...	2018-12-1
VertiPaq SE Query Begin	0 - VertiP...	2018-12-11 09:16:25...	2018-12-1
VertiPaq SE Query Begin	10 - Inter...	2018-12-11 09:16:25...	2018-12-1
VertiPaq SE Query End	10 - Inter...	2018-12-11 09:16:25...	2018-12-1
VertiPaq SE Query End	0 - VertiP...	2018-12-11 09:16:25...	2018-12-1
Query End			

```
< 
SET DC_KIND="DENSE";
SELECT
SUM([Sales Agg (2460)].[SalesAmount Sum (2563)]) AS [$Measure0], COUNT()
FROM [Sales Agg (2460)];
```

Figure 17-62. Querying the aggregated table

In this example, the query hits the first aggregated table in memory, as Figure 17-63 clearly illustrates.

### Query hits the first aggregated table in the memory

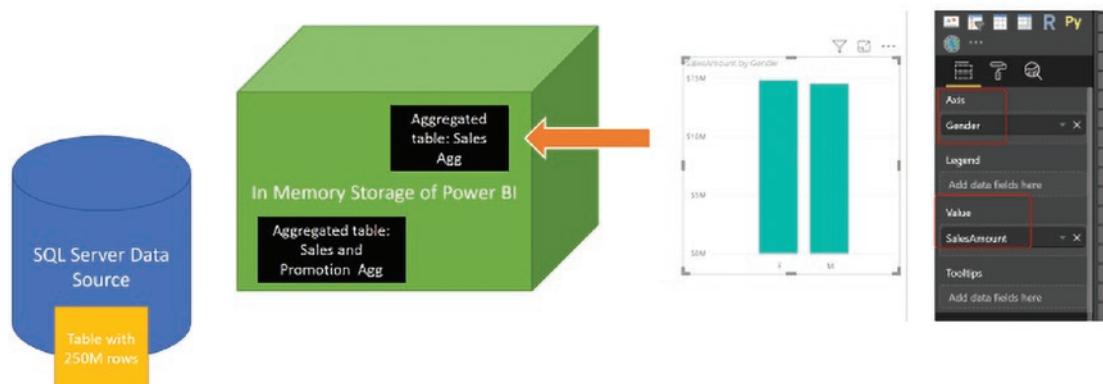
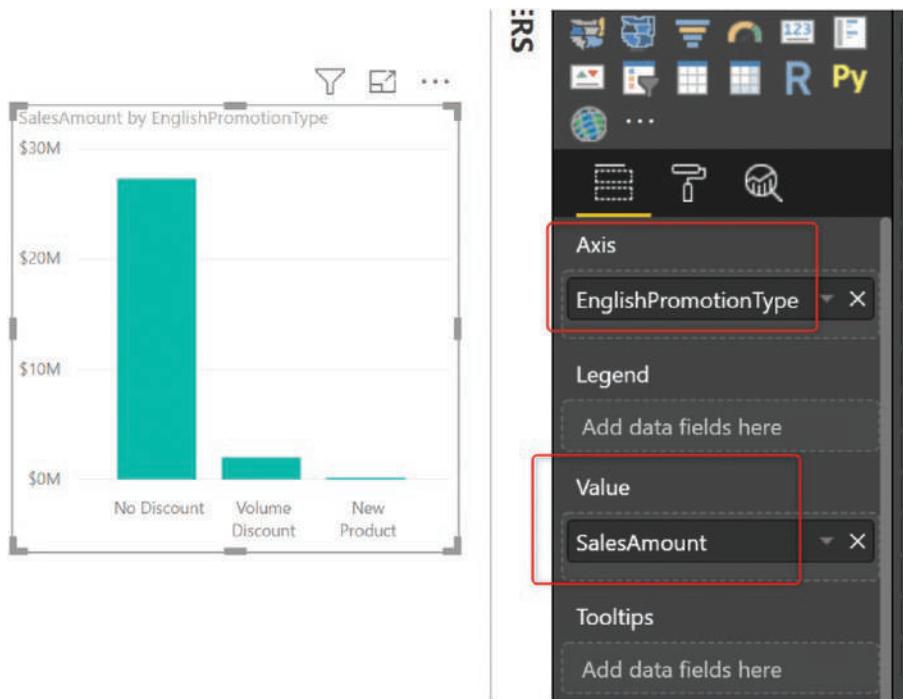


Figure 17-63. Query path

### Query Hits the Second Aggregated Table in Memory

Figure 17-64 offers another visual that uses EnglishPromotionType (from DimPromotion) and SalesAmount (from FactInternetSales).



**Figure 17-64.** Another sample visualization

The result this time cannot be fetched from the Sales Agg table (because DimPromotion is not there as the Group By function), so it will be queried from the second aggregated table—Sales and Promotion Agg—as highlighted in Figure 17-65.

```

Command: DAX
        12 - DAX    2018-12-11 09:21:04... 2018-12-11 09:21:04... 12 6818f1ec-... RezaRad AzureAD
Query Begin
3 - DAXQuery 2018-12-11 09:21:04... 2018-12-11 09:21:04... 12 6818f1ec-... RezaRad AzureAD
VertiPaq SE Query Begin
0 - VertiP... 2018-12-11 09:21:04... 2018-12-11 09:21:04... 12 6818f1ec-... RezaRad AzureAD
VertiPaq SE Query Begin
10 - Inter... 2018-12-11 09:21:04... 2018-12-11 09:21:04... 12 6818f1ec-... RezaRad AzureAD
VertiPaq SE Query End
10 - Inter... 2018-12-11 09:21:04... 2018-12-11 09:21:04... 12 6818f1ec-... RezaRad AzureAD
VertiPaq SE Query End
0 - VertiP... 2018-12-11 09:21:04... 2018-12-11 09:21:04... 12 6818f1ec-... RezaRad AzureAD
Query End

<                                     >

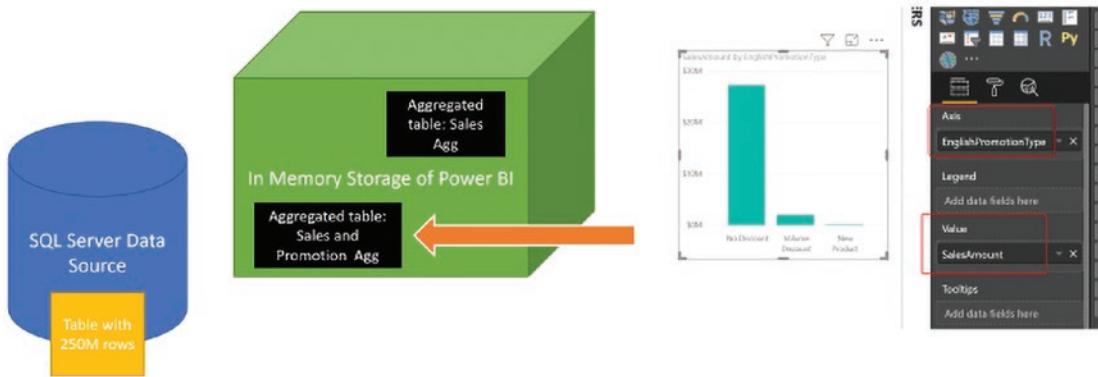
SET DC_KIND='DENSE';
SELECT
[DimPromotion (1353)].{EnglishPromotionType (1362)} AS [DimPromotion (1353)SEnglishPromotionType (1362)],
SUM([Sales and Promotion Agg (5275)].[SalesAmount Sum ($282)]) AS [$measured], COUNT()
FROM [Sales and Promotion Agg (5275)]
LEFT OUTER JOIN [DimPromotion (1353)] ON [Sales and Promotion Agg (5275)].[PromotionKey (5282)]=[DimPromotion (1353)].[PromotionKey (1356)];

```

**Figure 17-65.** Querying the second aggregated table

In this instance, the query hits the second aggregated table in memory, as illustrated in Figure 17-66.

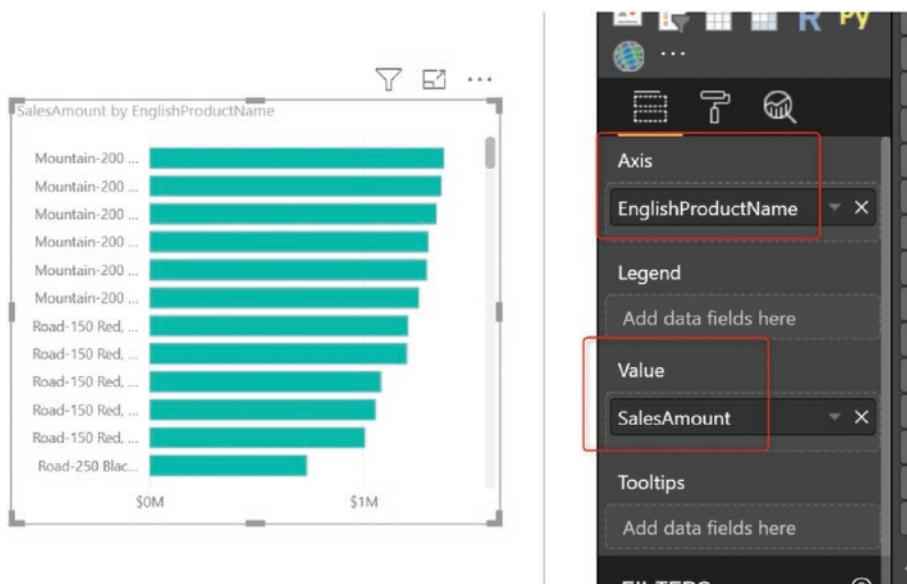
## Query hits the second aggregated table in the memory



**Figure 17-66.** Query path to the second aggregated table

## Query Hits the DirectQuery Table in the Source

Figure 17-67 uses EnglishProductName (from DimProduct) and SalesAmount (from FactInternetSales).



**Figure 17-67.** A final sample visualization

The result this time cannot be fetched from any of the aggregated tables, so it comes directly from the DirectQuery source table—FactInternetSales—as shown in Figure 17-68.

```
EventClass: Trace Start
SQL:BatchStarting SELECT TOP (1000000) [ts].[English...].Net SqlClient-... Rezitated AzureA... 4296 54 2018
SQL:BatchCompleted SELECT TOP (1000000) [ts].[English...].Net SqlClient-... Rezitated AzureA... 46 1961 0 66 4296 54 2018


[Outer].[RevisionNumber] as [RevisionNumber],
[Outer].[OrderQuantity] as [OrderQuantity],
[Outer].[UnitPrice] as [UnitPrice],
[Outer].[ExtendedAmount] as [ExtendedAmount],
[Outer].[UnitPriceDiscountPct] as [UnitPriceDiscountPct],
[Outer].[DiscountAmount] as [DiscountAmount],
[Outer].[ProductStandardCost] as [ProductStandardCost],
[Outer].[TotalProductCost] as [TotalProductCost],
[Outer].[SalesAmount] as [SalesAmount],
[Outer].[TaxAmt] as [TaxAmt],
[Outer].[Freight] as [Freight],
[Outer].[CarrierTrackingNumber] as [CarrierTrackingNumber],
[Outer].[CustomerNumber] as [CustomerNumber],
[Outer].[OrderDate] as [OrderDate],
[Outer].[DueDate] as [DueDate],
[Outer].[ShipDate] as [ShipDate],
[Outer].[ProductSubcategoryKey] as [ProductSubcategoryKey]
from
select [...] .[Productkey] as [Productkey],
[orderdatekey] as [Orderdatekey],
[shipdatekey] as [Shipdatekey],
[customerkey] as [Customerkey],
[productkey] as [Productkey],
[currencykey] as [Currencykey],
[salesterritorykey] as [Salesterritorykey],
[salesterritoryname] as [Salesterritoryname],
[saleorderlinenumber] as [SaleorderlineNumber],
[revisionnumber] as [RevisionNumber],
[orderquantity] as [OrderQuantity],
[unitprice] as [UnitPrice],
[extendedamount] as [ExtendedAmount],
[unitpricediscountrct] as [UnitPriceDiscountPct],
[productstandardcost] as [ProductStandardCost],
[totalproductcost] as [TotalProductCost],
[salesamount] as [SalesAmount],
[taxamt] as [TaxAmt],
[freight] as [Freight],
[carriertrackingnumber] as [CarrierTrackingNumber],
[customerponumber] as [CustomerPONumber],
[orderdate] as [OrderDate],
[duedate] as [DueDate],
[shipdate] as [ShipDate]
From [dbo].[FactInternetSales] as [...]
as [Outer]
left outer join [dbo].[DimProduct] as [Sinner] on ([Outer].[Productkey] = [Sinner].[Productkey]) AS [t?]

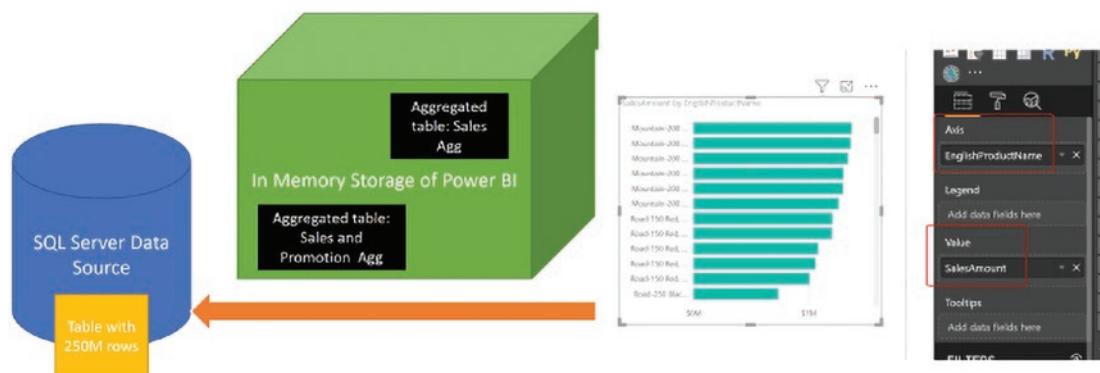
left outer join
Select [*Table3].[Productkey] as [Productkey],
[*Table3].[ProductAlternateKey] as [ProductAlternateKey],
[*Table3].[Name] as [Name],
[*Table3].[Color] as [Color],
[*Table3].[Size] as [Size],
[*Table3].[Weight] as [Weight],
[*Table3].[RetailPrice] as [RetailPrice],
[*Table3].[ListPrice] as [ListPrice],
[*Table3].[StandardCost] as [StandardCost],
[*Table3].[LastEditedBy] as [LastEditedBy],
[*Table3].[ModifiedDate] as [ModifiedDate]
From [dbo].[DimProduct] AS [Table3]

```

**Figure 17-68.** From the DirectQuery source table

In the last example, the query hits the DirectQuery table in the data source; see Figure 17-69.

Query hits the DirectQuery table in the data source



**Figure 17-69.** Query path toward data source

Power BI is switching nicely between layers of aggregated tables without you noticing it. All these operations are happening behind the scenes. The user will feel that one table (`FactInternetSales`) is serving all queries, and the query response time will be super-fast (with the help of aggregations).

Aggregation can be implemented on multiple levels to speed up the performance. If a specific combination of dimension fields is used in user visualizations, that combination is a good candidate for aggregation. Depending on the size of the aggregation table, precedence should be followed.

## Aggregation for Imported Data

Aggregation is not only for DirectQuery tables. Sometimes a very large table that is imported can be slow. Aggregated tables on imported tables can also improve performance. For situations like that, you need to use DAX measures to switch between the main table and the aggregated table.

One of the questions I normally get after explaining aggregations in Power BI is, does the aggregation work only for composite mode and in scenarios to speed up the DirectQuery? Or does it work in Import mode as well? This section addresses this question and shows how you can speed up your model with aggregation.

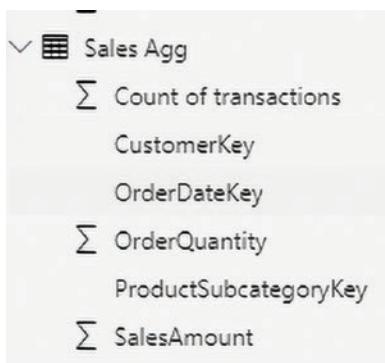
### The Aggregated Table

Creating an aggregated table can be done anywhere—in Power Query, in the backend data source, and using DAX calculated tables. The fact table includes all the columns shown in Figure 17-70.

FactInternetSales
CarrierTrackingNumber
Σ CurrencyKey
CustomerKey
CustomerPONumber
Σ DiscountAmount
📅 DueDate
Σ DueDateKey
Σ ExtendedAmount
Σ Freight
📅 OrderDate
OrderDateKey
Σ OrderQuantity
ProductKey
Σ ProductStandardCost
PromotionKey
Σ RevisionNumber
Σ SalesAmount
Σ SalesOrderLineNumber
SalesOrderNumber
Σ SalesTerritoryKey
📅 ShipDate
Σ ShipDateKey
Σ TaxAmt
Σ TotalProductCost
Σ UnitPrice
Σ UnitPriceDiscountPct

**Figure 17-70.** Sample fact table

In this case, the aggregated table is the GROUPED-BY version of that table (which are grouped by key columns and aggregations), as shown in Figure 17-71.



**Figure 17-71.** The resulting aggregated table

#### No Manage Aggregations in Imported Mode

When you create aggregation on an imported table versus DirectQuery table, one of the differences is the Manage Aggregation option, previewed in Figure 17-72. This feature, which is also called aggregation awareness, works only if the main table is DirectQuery.

The screenshot shows the 'Manage aggregations' dialog box. At the top, it says 'Manage aggregations' and 'Aggregations accelerate query performance to unlock big-data sets. [Learn more](#)'. Below this is a section for 'Aggregation table' (set to 'Sales Agg') and 'Precedence' (set to '0'). The main area is a table with four columns: 'AGGREGATION COLUMN', 'SUMMARIZATION', 'DETAIL TABLE', and 'DETAIL COLUMN'.

AGGREGATION COLUMN	SUMMARIZATION	DETAIL TABLE	DETAIL COLUMN
Count of transactions	Count	DimGeography	
CustomerKey	Select Summarizatio...	DimProduct	
OrderDateKey	Select Summarizatio...	DimProductCategory	
OrderQuantity	Select Summarizatio...	DimProductSubcategory	
ProductSubcategoryKey	Select Summarizatio...	DimPromotion	
		FactInternetSales	

A note at the bottom left says 'This table will be hidden if aggregations are set because agg...'. At the bottom right are 'Apply all' and 'Cancel' buttons.

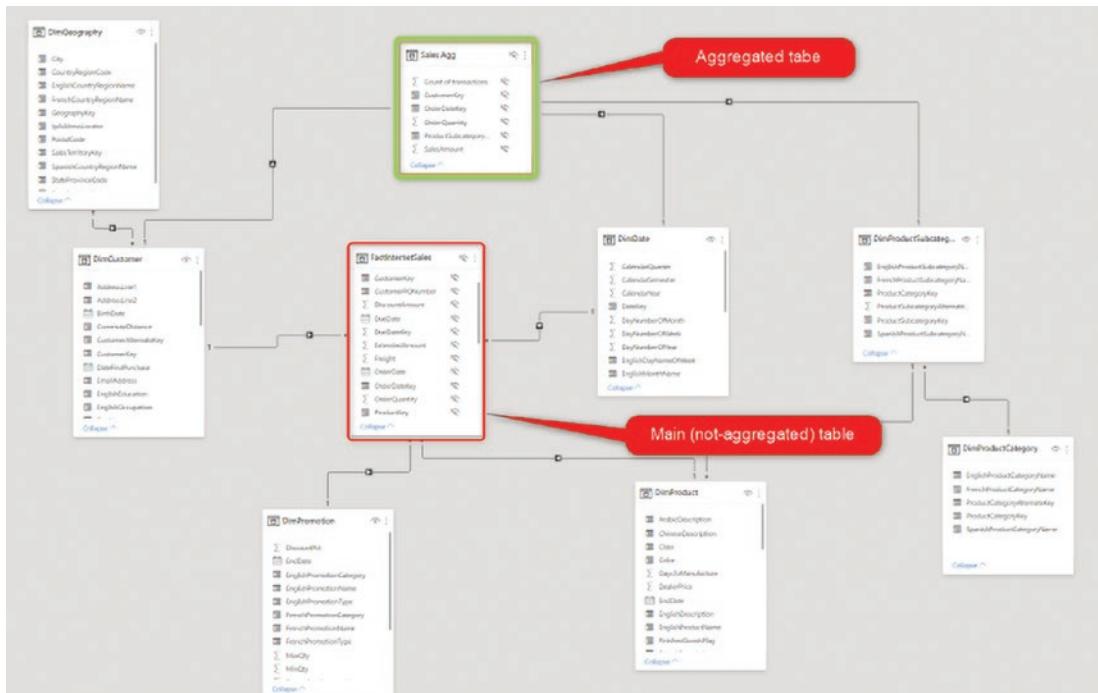
**Figure 17-72.** The Manage Aggregations window

This doesn't mean that you cannot use aggregations in the Import Data model. It just means that this feature (which is Power BI's ability to find the aggregated table when the main table is queried) doesn't work, and you have to do it in another way.

## DAX Measures Instead of Manage Aggregation

You don't have the aggregation awareness of Power BI in all imported models. However, you can use DAX measures to do almost anything. You can use a simple IF statement with another function that checks what tables are filtered, to switch between the aggregated and the main table.

To understand how it works, first look at the model in Figure 17-73



**Figure 17-73.** Sample model of tables

In this model, the Sales Agg is an aggregated table, which includes the grouped by data dimDate, dimCustomer, and dimProductSubCategory. This means that if you use anything from these tables, as well as DimGeography and DimProductCategory, you can get what you want from the Sales Agg table.

However, if you slice and dice the data by DimPromotion or DimProduct, then you would need FactInternetSales; see Figure 17-74.



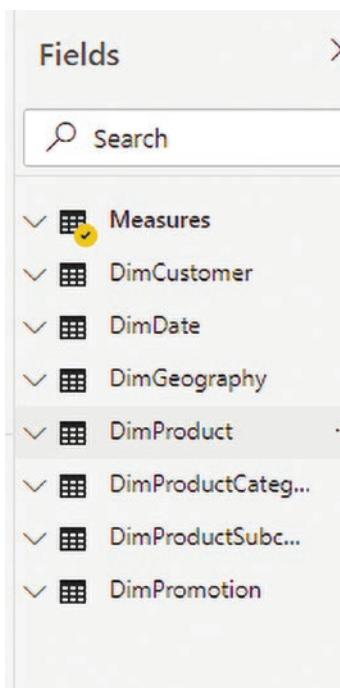
**Figure 17-74.** Sample model with sliced data

This can be done using an IF statement and a filter-checking function in DAX;

```
Sales = IF(  
ISCROSSFILTERED(DimPromotion[PromotionKey]) || ISCROSSFILTERED(DimProduct[ProductKey]),  
SUM(FactInternetSales[SalesAmount]), // main table  
SUM('Sales Agg'[SalesAmount]) // aggregated table  
)
```

This expression uses the main (non-aggregated) table if any of the fields from DimPromotion or DimProduct are used in a visualization. (ISCROSSFILTERED will check if any combination of fields from that table filters the output.) The expression will use the aggregated table otherwise.

After these steps, the two main and aggregated tables can be hidden from the report view (see Figure 17-75).



**Figure 17-75.** Table selection

### Testing the Aggregation

To check if you are fetching data from an aggregated table or non-aggregated table, you can create another measure as follows:

```
Sales from which table = IF(
ISCROSSFILTERED(DimPromotion[PromotionKey]) || ISCROSSFILTERED(DimProduct[ProductKey]),
"FactInternetSales",
"Sales Agg"
)
```

This measure can now be used in a tooltip, and Figure 17-76 shows how it works.

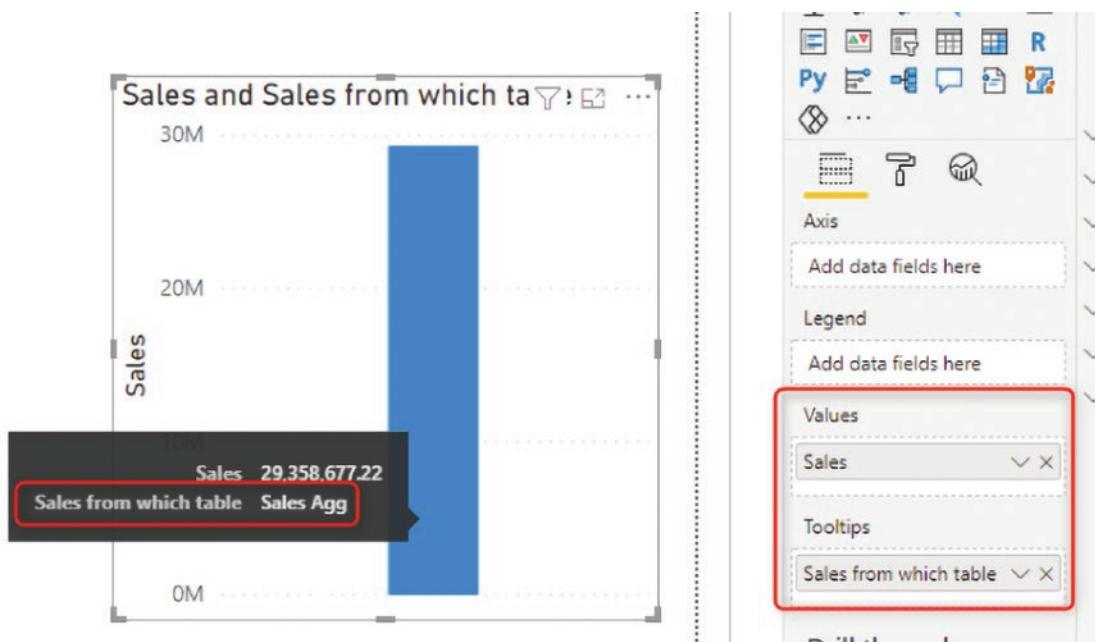
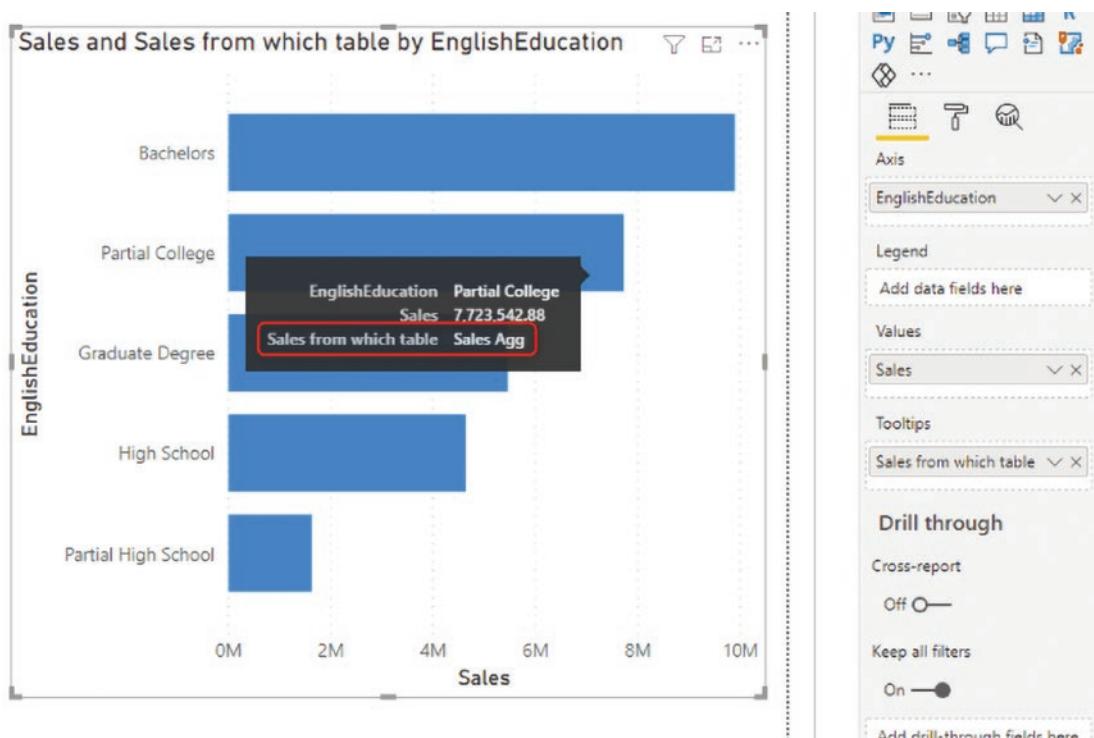


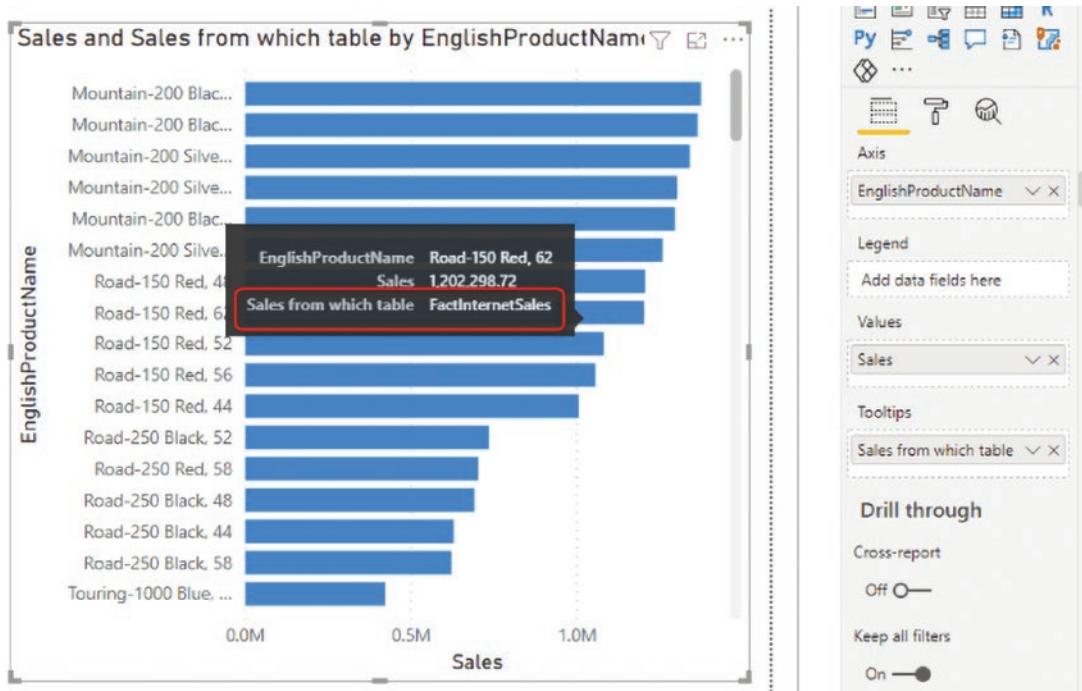
Figure 17-76. Testing using tooltips

If you slice and dice data by anything other than DimProduct and DimPromotion, the calculation result will come from Sales Agg, as shown in Figure 17-77.



**Figure 17-77.** Data results from the aggregated table

If you filter data by any of the DimProduct or DimPromotion fields, you will get the data from a non-aggregated table, as shown in Figure 17-78.



**Figure 17-78.** Data results from a non-aggregated table

#### Multiple Layers of Aggregations Are Also Possible

Using the DAX measure approach, multiple layers of aggregation are also possible. All you need to do is add more conditions to your IF expression. You can also use SWITCH and get the result from the relevant aggregated table.

In addition to good data modeling, proper DAX calculations, and having calculations in their rightful places, aggregation is another way to speed up the performance of your model. If you are dealing with large tables, aggregations help reduce the number of rows to process for the calculation.

## Automatic Aggregation

Creating aggregations in Power BI highly depends on using columns and tables in the report. Performance tuning advice is to start creating aggregations on those columns mostly used in visuals. The good news is that Power BI can automate this process for you. It is called *automatic aggregation*.

Automatic aggregation is the process in which Power BI checks the usage logs of the columns and tables in the Power BI dataset and creates automatic aggregation tables based on that. This requires minimum effort from you as a developer. It will all be handled very simply by Power BI. You can enable this from the dataset settings feature, as demonstrated in Figure 17-79.

◀ Scheduled refresh and performance optimization

To improve the performance of exploring reports, enable scheduled refresh and automatic aggregations and estimate how caching can improve query response times.

**Automatic aggregations training** ⓘ

To speed up exploring reports, Power BI can generate aggregations tables to cache some of the data for queries—your reports will run faster, and visuals with cached data will load more quickly. [Learn more](#)



**Figure 17-79.** Enabling automatic aggregation

There is a setting for how much query coverage you want to get with the automatic aggregation, as Figure 17-80 shows.

**Query coverage** ⓘ

Adjust the percentage of queries that will use aggregated caches.



ⓘ Increasing the percentage of queries that use aggregations increases refresh time and data latency. More aggregations generate more queries on the source system. More queries increase the probability of query timeouts, system overload and costs.

**Figure 17-80.** Setting up query coverage for automatic aggregation

## Summary

Aggregation is a game-changer in the performance and speed of Power BI solutions when the data source table is huge. With the help of aggregations, you can have layers of pre-calculations stored in memory and ready to respond to queries from users. The DirectQuery data source will be used for the atomic transaction queries. Aggregation is not just for DirectQuery models; it can also improve the performance of Import Data models.

## CHAPTER 18



# Big Data with Incremental Refresh and Hybrid Tables

The default configuration for the Power BI dataset is to wipe out the entire data and reload it again on each refresh. This can be a long process if you have a big dataset. Hybrid tables in Power BI keep part of the data in DirectQuery, and the rest of the data is imported for data freshness and performance. This chapter explains how to set up Incremental Refresh in Power BI. You also learn about hybrid tables. Incremental Refresh is not just used in Power BI datasets but also in dataflows and datamarts. In this chapter, you learn how to load only part of the changed data instead of loading the entire data each time.

## What Is Incremental Refresh?

When you load data from the source into the destination (Power BI), there are two methods: Full Load and Incremental Refresh. Full Load fetches the entire dataset each time and wipes out the previous data. When I say the entire dataset, I mean after all the Power Query transformations because there might be some filtering in Power Query. When I talk about the entire data, I am referring to whatever data loads into the Power BI dataset.

If the dataset is small, or the refresh process is quick, then using Full Load is not a problem. The problem happens when your dataset is big, or the refresh process takes too long. Imagine you have a large dataset that includes 20 years of data. Data from 20 years ago likely won't change, or even the data from 5 years ago, and sometimes even a year ago. Why reprocess data that never changes? Why reload data that isn't being updated? Incremental Refresh is the process of loading only the part of the data that could change and adding it to the previous dataset, which is no longer changing.

Incremental Load splits the table into partitions. The number of partitions is based on the settings applied at the time of Incremental Refresh. For example, if you want to refresh only last year's data, a yearly partition will likely be created for every year, and the one for the current year will be refreshed on a scheduled basis.

## Hybrid Tables

If you need data freshness (near real-time) on a big table, there is an option for you. You can design your table to keep both your DirectQuery and Import Data in a single table. The DirectQuery part ensures near real-time data, and the imported part ensures the best performance in Power BI.

Hybrid tables are partitioned, so their most recent partition is a DirectQuery from the data source, and their historical data is imported into other partitions. This is different from dual storage mode. This is a table whereby part of it is imported, and part of it is DirectQuery. Hybrid tables can only be applied on a table that is configured for Incremental Refresh.

# Configuring Incremental Refresh

## Table with Date Field(s)

To set up Incremental Refresh, you must have one or more tables with a date field. The date field is used to implement a partial refresh of the data. For example, let's say you have a FactSales table. You want to load all sales made earlier than a year ago just once, but everything from a year ago to now, you want to load regularly. You need to have a date field in your table. Often, this field will be called something like CreatedDate, ModifiedDate, OrderDate, PublishDate, and so on.

## A data source that supports query folding

*Query folding* is when the Power Query transformations are translated to the data source language (such as T-SQL when querying from SQL Server). Although you can implement Incremental Refresh on any data source, even if it does not support query folding, it would be pointless to do so. The main point of Incremental Refresh is that Power BI reads the data that has changed rather than reading the entire data from the source. With a data source that supports query folding, that is possible because Power BI can query only the recent part of the data. However, if your data source doesn't support query folding (for example, it is a CSV file), then Power BI will read the entire set of data anyway.

## Licensing Requirement

You don't need a Premium or PPU license to use Incremental Refresh. You can even set it up using a Power BI Pro license. However, hybrid tables require a Power BI Premium or PPU capacity.

## Limitations: Things to Know Beforehand

One important limitation to consider is that after setting up Incremental Refresh, you can no longer download the PBIX file from the service. That's because the data is now partitioned (see Figure 18-1). This also makes sense because the data size is likely too large for downloading.

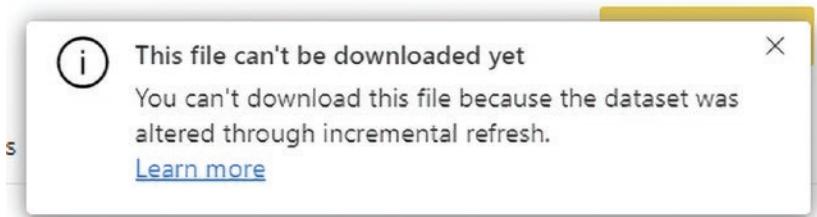


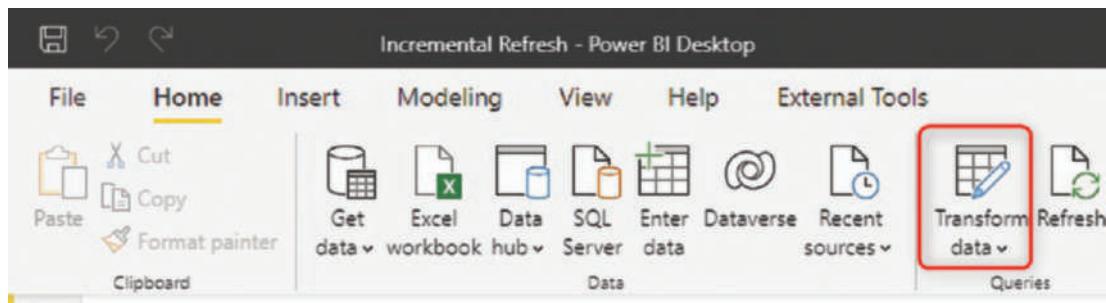
Figure 18-1. PBIX download limitation

# Setting Up Incremental Refresh

To set up Incremental Refresh, you go through some steps in the Power BI Desktop and then in the Power BI Service. Let's look at these steps one by one.

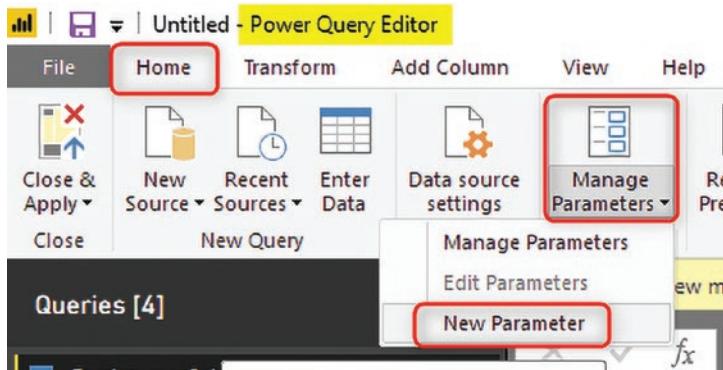
## Parameters in Power Query

You need to use Power Query parameters to set up an Incremental Refresh in Power BI. You need to create two parameters with the reserved names of RangeStart and RangeEnd. (Note that Power Query is a case-sensitive language.) Go to Transform Data in your Power BI Desktop solution, as shown in Figure 18-2.



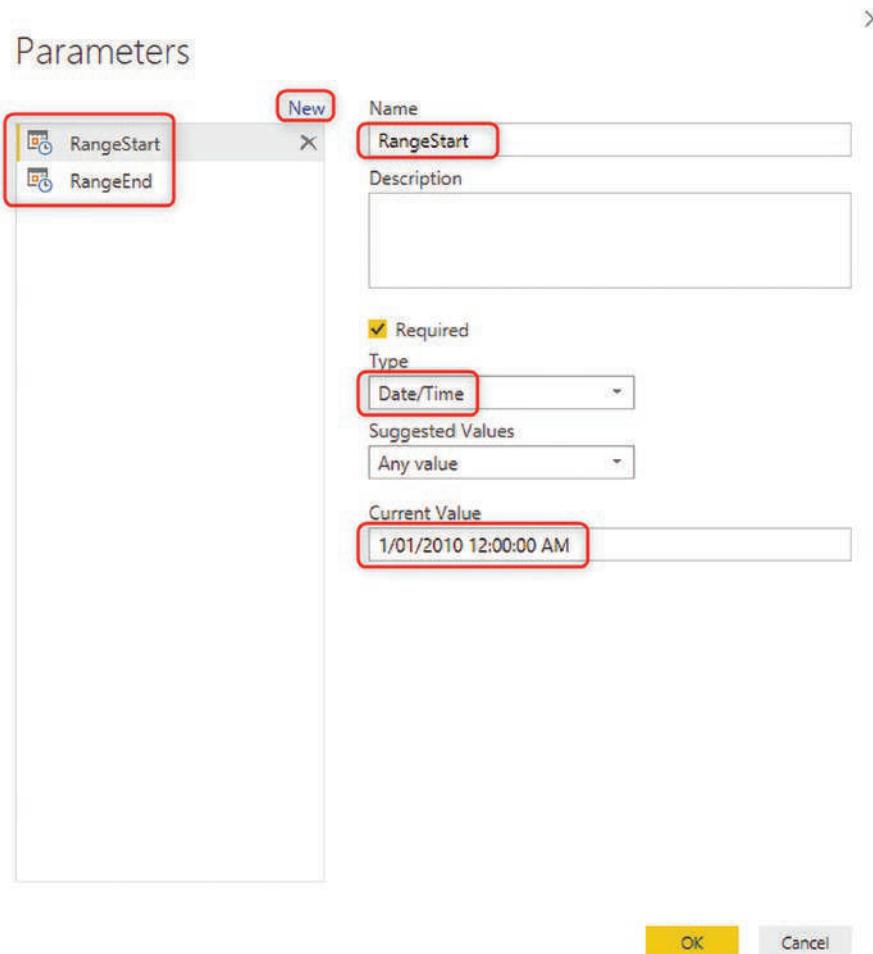
**Figure 18-2.** Setting up parameters (choose Transform Data)

Click New Parameter, as shown in Figure 18-3.



**Figure 18-3.** Setting up parameters (choose New Parameter)

Then, as shown in Figure 18-4, create two parameters with the DateTime data type, called RangeStart and RangeEnd, and set a default value for each. The default values can be anything.



**Figure 18-4.** Setting up parameters (choose RangeStart and RangeEnd)

## Filter Data Based on the Parameters

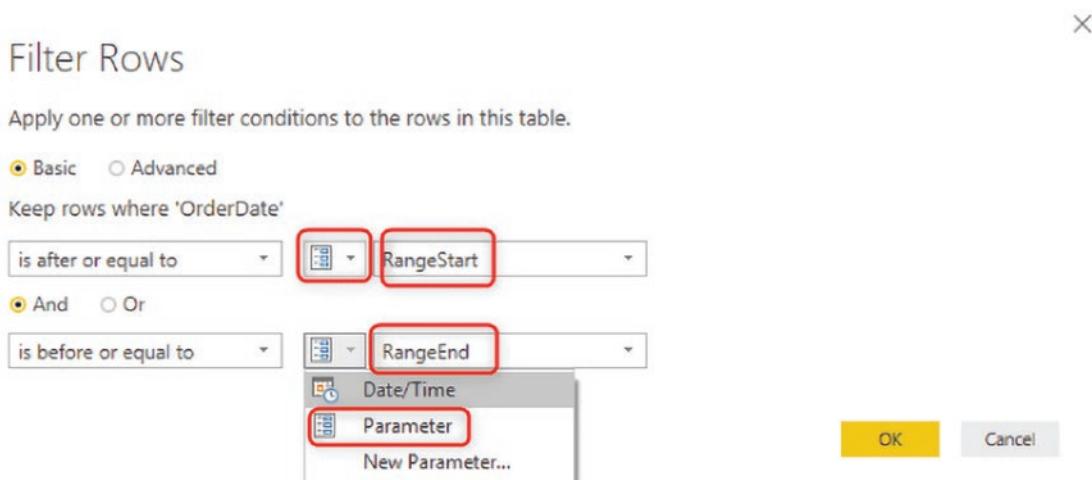
After creating the two parameters, you need to filter the data of the date field based on these two parameters. For example, in the FactInternetSales table, I can filter OrderDate using column filtering, as Figure 18-5 demonstrates.

A screenshot of the Power BI Desktop interface showing a context menu for filtering data. The menu is triggered by a right-click on a column header. The 'Date/Time Filters' option is highlighted with a red box. The 'Between...' option under the 'Date/Time Filters' section is also highlighted with a red box. The menu includes other options like Sort Ascending, Sort Descending, Clear Sort, Clear Filter, Remove Empty, Equals..., Before..., After..., In the Next..., and In the Previous... . The background shows a table with columns gNumber, CustomerPONumber, OrderDate, DueDate, and Ship.

gNumber	CustomerPONumber	OrderDate	DueDate	Ship
ABC	123	13/07/2005 12:00:00 AM	8	
		13/07/2005 12:00:00 AM	8	
		13/07/2005 12:00:00 AM	8	
		13/07/2005 12:00:00 AM	8	
		13/07/2005 12:00:00 AM	8	
		14/07/2005 12:00:00 AM	9	

**Figure 18-5.** Filtering data

You can use a Between filter and the RangeStart and RangeEnd parameters, as shown in Figure 18-6.

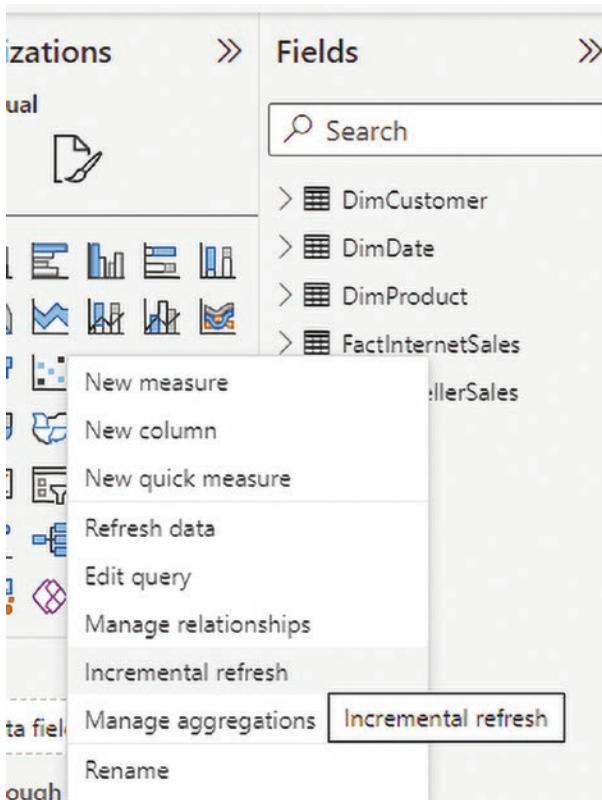


**Figure 18-6.** Filtering options

After this action, your data in the table will be filtered based on the default values you set for the RangeStart and RangeEnd parameters. However, don't worry about that for now. These two parameters will be overwritten with the configuration you make in the Incremental Refresh setting of the Power BI Desktop.

## Power BI Desktop Incremental Refresh Setup

The final step in the Power BI Desktop is to close and apply the Power Query Editor window and determine the Incremental Refresh setting for the table. Right-click the table in the Power BI Desktop and select Incremental Refresh, as shown in Figure 18-7.



**Figure 18-7.** Selecting Incremental Refresh

In the Incremental Refresh and Real-Time Data window, start by selecting the table. If the table doesn't include the RangeStart and RangeEnd parameters used in the filter criteria, you can't configure it. For example, in Figure 18-8, DimCustomer doesn't give me the option to do the configuration.

## Incremental refresh and real-time data

 Before you can set up incremental refresh on this table, you need to set up parameters. [Learn more](#)

Refresh large tables faster with incremental refresh. Plus, get the latest data in real time with DirectQuery (Premium only). [Learn more](#)

-  These settings will apply when you publish the dataset to the Power BI service. Once you do that, you won't be able to download it back to Power BI Desktop. [Learn more](#)

### 1. Select table

DimCustomer

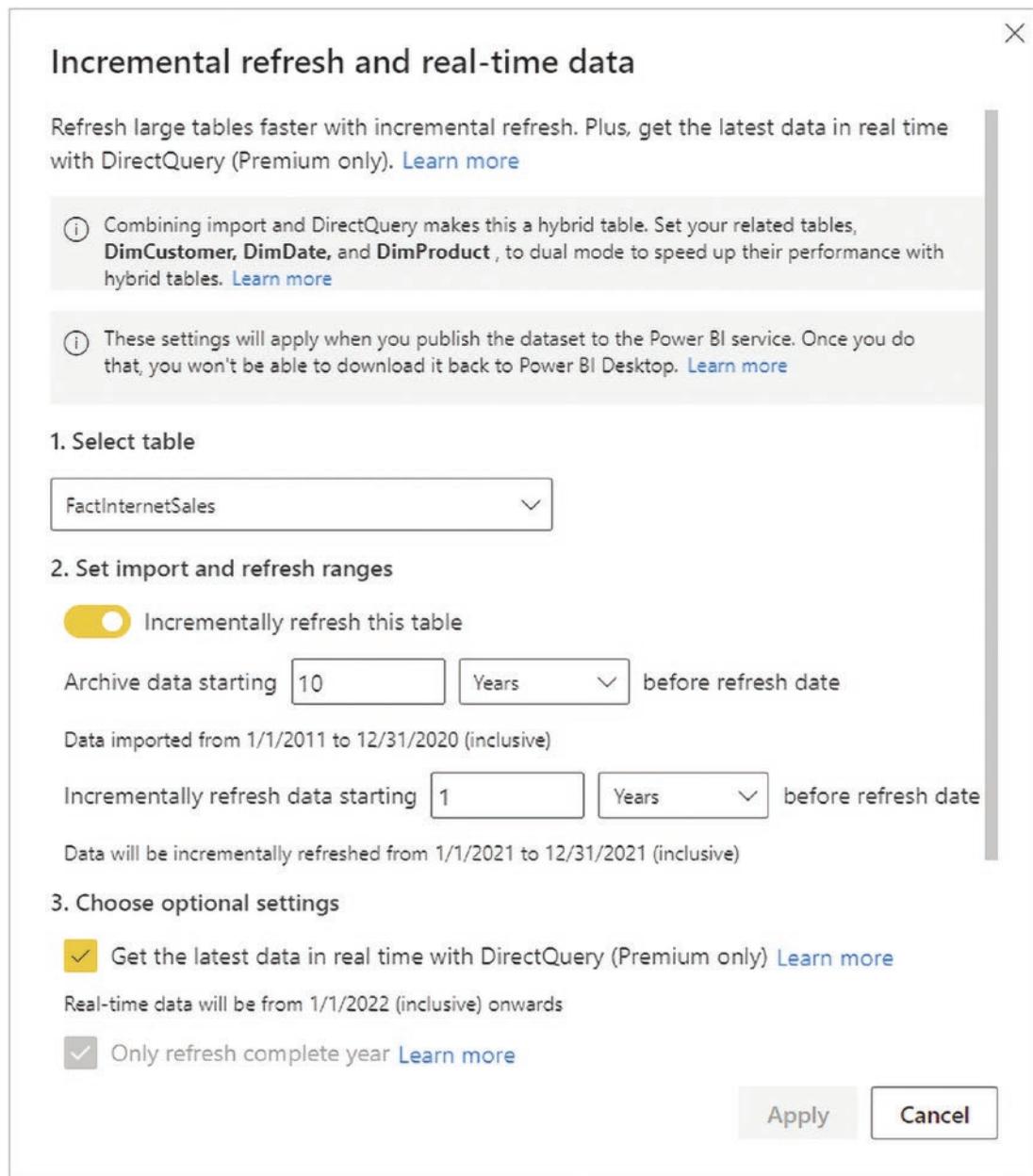
### 2. Set import and refresh ranges

Incrementally refresh this table

**Figure 18-8.** Parameters needed to set up in Incremental Refresh

However, I can configure FactInternetSales because I did filter the OrderDate field of this table based on the parameters. There are two things to notice at this step: If your data source supports query folding, then Incremental Refresh works best. If not, it is not recommended to use it. Large volumes of data can come from a relational data store system, and query folding lets you load only the subset of the data that you need.

Another thing to note is the Get the Latest Data in Real-Time with DirectQuery option, which requires Premium or PPU licensing. And finally, remember that you cannot download a PBIX file from the service when Incremental Refresh is set up.

**Figure 18-9.** Incremental Refresh settings

Configuring Incremental Refresh is easy. You set up which rows to store (load only once and store them) and which rows to refresh (reload every time); see Figure 18-10.

**1. Select table**

FactInternetSales

**2. Set import and refresh ranges** Incrementally refresh this tableArchive data starting  Years  before refresh date

Data imported from 1/1/2011 to 12/31/2020 (inclusive)

Incrementally refresh data starting  Years  before refresh dateData will be incrementally refreshed from 1/1/2021 to 12/31, **3. Choose optional settings** Get the latest data in real time with DirectQuery

Real-time data will be from 1/1/2022 (inclusive) onwards

 Only refresh complete year [Learn more](#)

Days

Months

[Learn more](#)

Quarters

Years

**Figure 18-10.** Selecting refresh specifics

## Hybrid Table Setup

You have one more configuration if you want to set up your table as a hybrid table. This requires a Premium or PPU license. Select the Get the Latest Data in Real-Time with DirectQuery option, as shown in Figure 18-11. Your dataset must have been published into a Premium or PPU workspace. Once you set this configuration, you can see the period of real-time data and a diagram of the timeline of your setup.

## 2. Set import and refresh ranges

Incrementally refresh this table

Archive data starting  Years  before refresh date  
 Data imported from 1/1/2011 to 12/31/2020 (inclusive)

Incrementally refresh data starting  Years  before refresh date  
 Data will be incrementally refreshed from 1/1/2021 to 12/31/2021 (inclusive)

## 3. Choose optional settings

Get the latest data in real time with DirectQuery (Premium only) [Learn more](#)

Real-time data will be from 1/1/2022 (inclusive) onwards

Only refresh complete year [Learn more](#)

Detect data changes [Learn more](#)

## 4. Review and apply

Archived	Incremental Refresh	Real time
10 years before refresh date	1 year before refresh date	Refresh date

**Figure 18-11.** Hybrid table settings

After enabling this on your dataset, it can only be published to a Premium or PPU workspace.

## Incremental Refresh for Multiple Tables

You can set up the Incremental Refresh for multiple tables. You don't need more parameters; the two existing RangeStart and RangeEnd parameters are enough. You just need to set the filter in any other tables you want to refresh incrementally. See Figure 18-12.

The screenshot shows the Power Query Editor interface with the 'FactResellerSales' query selected. The table has columns: SalesAmount, Taxamt, Freight, CarrierTrackingNumber, CustomerPONumber, OrderDate, and DueDate. A date-time filter is applied to the OrderDate column, set to 'Between...' from 1/07/2005 12:00:00 AM to 1/07/2005 12:00:00 AM.

**Figure 18-12.** Filter tables using the same DateTime parameters

You then set up the configuration in the Power BI Desktop for each table, as demonstrated in Figure 18-13.

### 1. Select table

FactResellerSales

### 2. Set import and refresh ranges

Incrementally refresh this table

Archive data starting  Months  before refresh date

Data imported from 4/1/2022 to 8/31/2022 (inclusive)

Incrementally refresh data starting  Months  before refresh date

Data will be incrementally refreshed from 9/1/2022 to 9/30/2022 (inclusive)

**Figure 18-13.** Configuring Incremental Refresh for each table

Note that although you are using the same RangeStart and RangeEnd parameters, the Store and Refresh configurations for the Incremental Refresh can be different for each table.

## Publish to Service

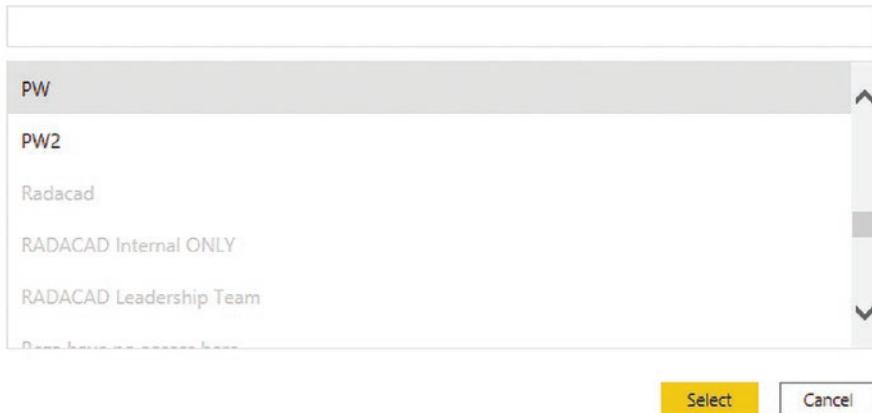
After configuring everything, you can publish the Power BI file to the service. Note that if you use hybrid tables, you can only publish to Premium workspaces, as indicated in Figure 18-14.

X

## Publish to Power BI

ⓘ At least one of your tables has a combination of incremental refresh and real-time data, which can only be published to Premium workspaces. [Learn more](#)

Select a destination



**Figure 18-14.** Hybrid tables require a Premium license

## What Do the Partitions Look Like?

Now that you have set up Incremental Refresh and the hybrid table settings, you can check out the partitions in the Power BI dataset. The method I used to show this works only if you have a Premium or PPU workspace. You can use the XMLA endpoint to connect to the dataset using the SQL Server Management Studio (SSMS) and see the partitions on a table.

As Figure 18-15 shows, you begin by opening the dataset settings in the Power BI Service.

The screenshot shows the Power BI service interface with the 'Datasets + dataflows' tab selected. A dataset named 'Incremental Refresh' is selected, and its context menu is open, with the 'Settings' option highlighted.

Name	Type	Owner
Datamart1	Dataset	PW
Incremental Refresh	Dataset	PW

**Figure 18-15.** Beginning the process of accessing the model and partitions from SSMS

In the Dataset settings, expand Server Settings and copy the connection string (if you don't see this section, your workspace may not be a Premium or PPU workspace). See Figure 18-16.

The screenshot shows the 'Connection string' settings page in Power BI service. The connection string 'powerbi://api.powerbi.com/v1.0/' is displayed in a text input field, and a yellow 'Copy' button is visible below it.

**Figure 18-16.** Copying the connection string

Open SSMS, create a connection to Analysis Services, and paste the connection string as the server name. Then set authentication to Azure Active Directory–Universal with MFA. Finally, use your Power BI email as the username. After authentication, you can see your dataset in the SSMS. Expand the tables, as shown in Figure 18-17.

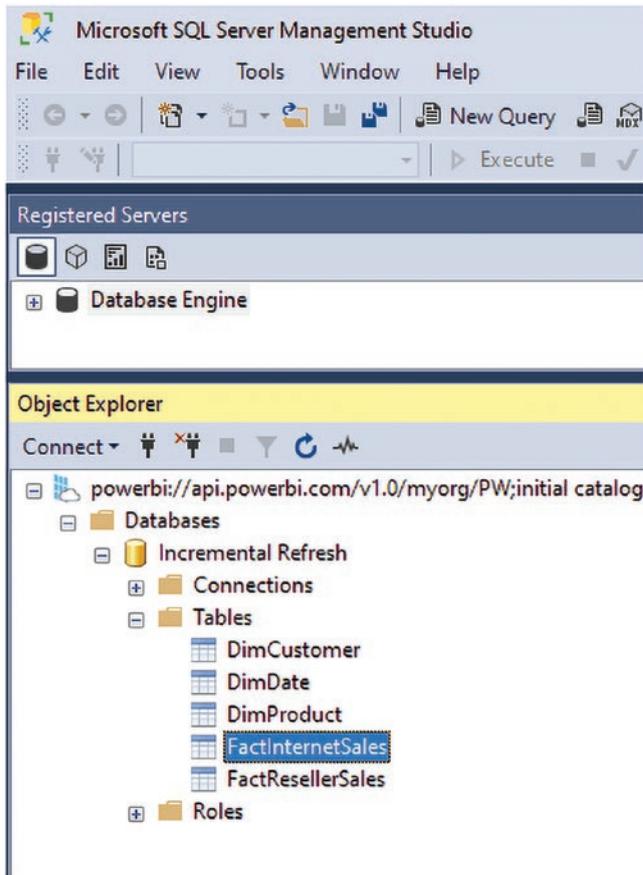
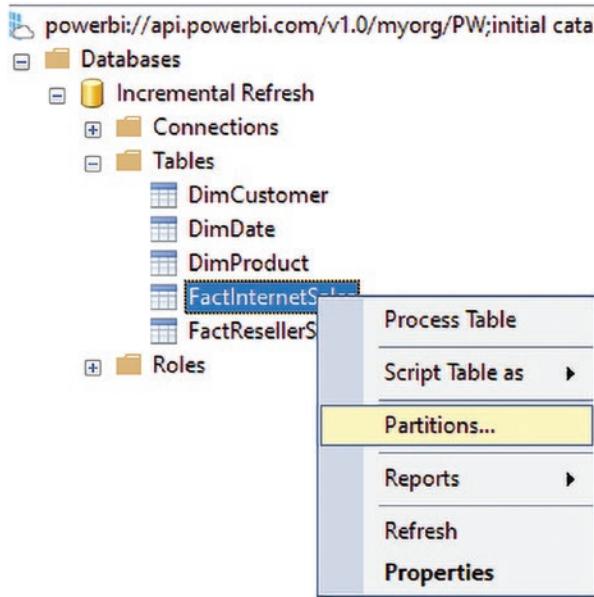


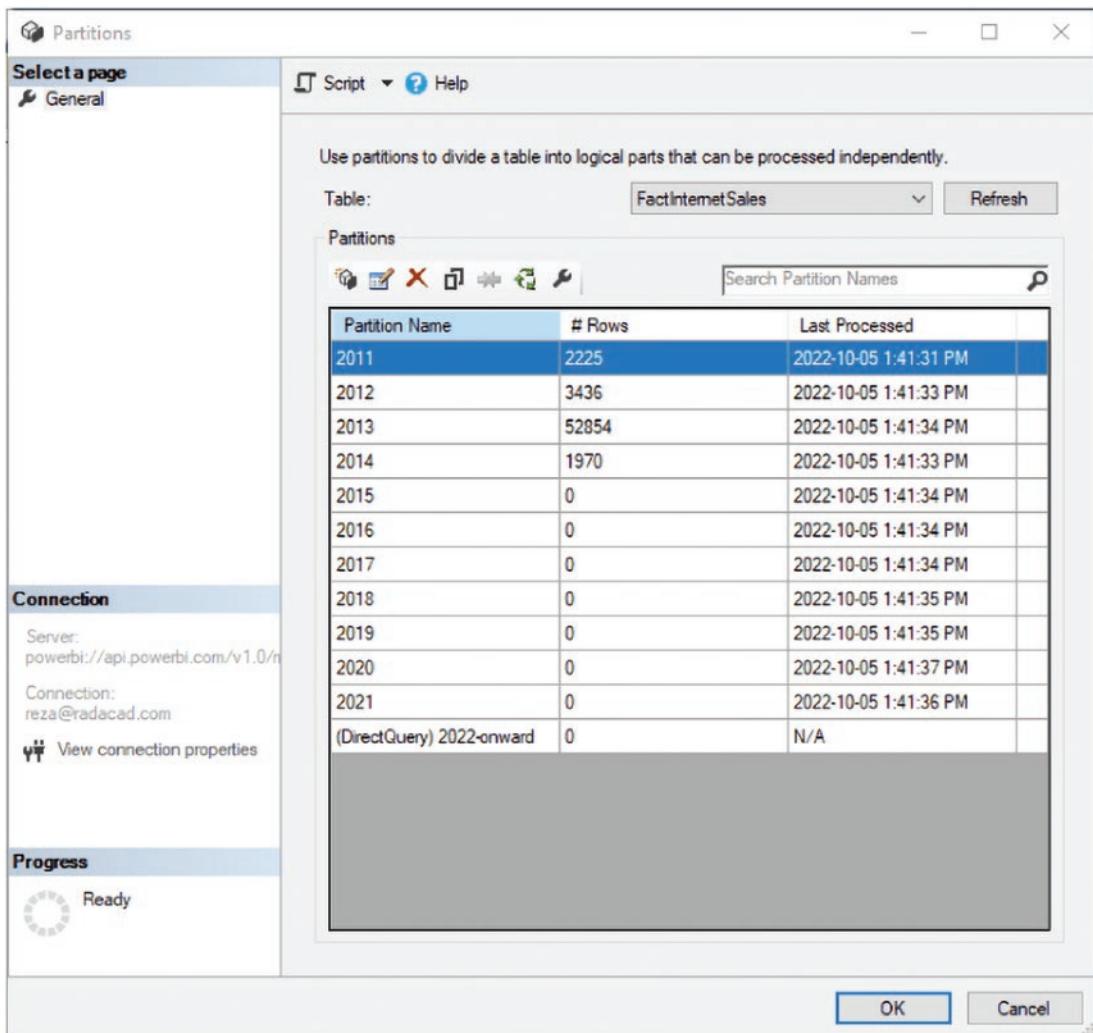
Figure 18-17. The model is visible in SQL Server Management Studio

Right-click a table (such as FactInternetSales) and select Partitions (see Figure 18-18).



**Figure 18-18.** Opening partitions

You will then be able to see the partitions, which may appear as shown in Figure 18-19.

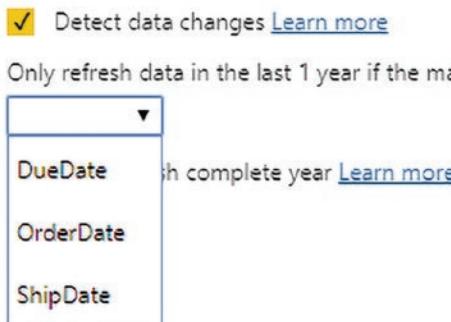


**Figure 18-19.** View of the partitions

If your table has a single partition, perhaps it doesn't have Incremental Refresh set up on it. If your table has hybrid table settings, the last partition will be a DirectQuery partition (which is what you see in Figure 18-19).

## Detecting Data Changes

Incremental Refresh makes processing much faster, because it reduces the number of rows being loaded into the dataset. However, there is still one better way to do that. Suppose you have a `ModifiedDate` (or `UpdatedDate`) column in your table. In that case, the Incremental Refresh process can monitor that field and only get rows whose date/time is after the last date/time in that field in the previous refresh. To enable this process, you enable Detect Data Changes and then choose the `ModifiedDate` or `UpdatedDate` column from the table, as shown in Figure 18-20. Notice that this is different from `OrderDate` and `TransactionDate`. Not all tables have a suitable field like this.



**Figure 18-20.** Enabling the Detect Data Changes option

## Only Refresh When Period Is Complete

Depending on the period you selected when you set up the Incremental Refresh, you can choose to just refresh when the period is complete. For example, in the FactInternetSales field shown in Figure 18-21, I set the refresh period to Year. Then I can set the option to Only Refresh the Complete Period. Even if it was in February 2019, it would only refresh the data up to December 2018 (because that is the last date that I have a complete year of data for).

### 2. Set import and refresh ranges

Incrementally refresh this table

Archive data starting  Years  before refresh date

Data imported from **1/1/2011 to 12/31/2020 (inclusive)**

Incrementally refresh data starting  Years  before refresh date

Data will be incrementally refreshed from **1/1/2021 to 12/31/2021 (inclusive)**

### 3. Choose optional settings

- Get the latest data in real time with DirectQuery (Premium only) [Learn more](#)
- Only refresh complete year [Learn more](#)
- Detect data changes [Learn more](#)

**Figure 18-21.** Setting the refresh period

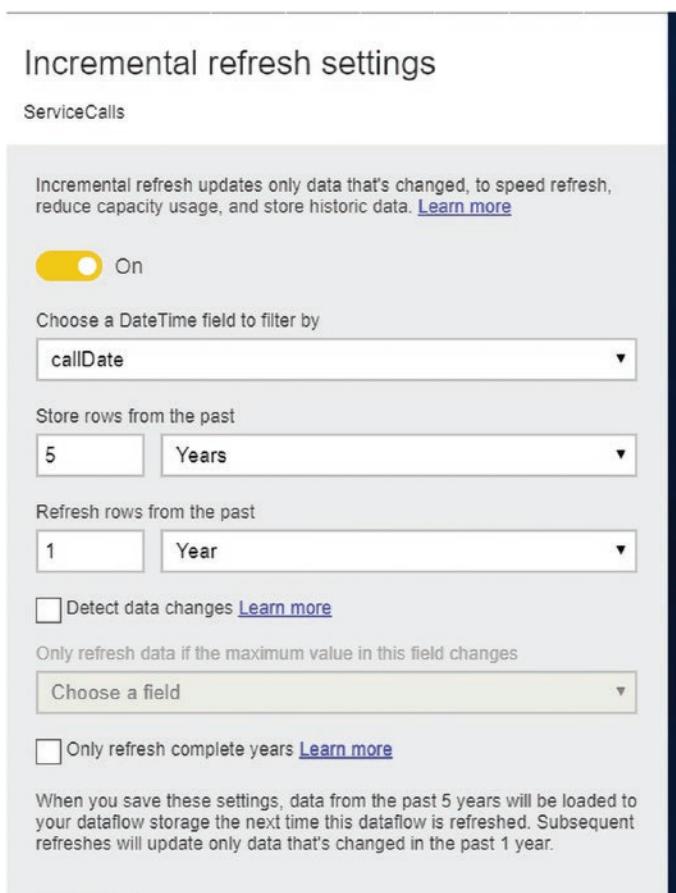
## Incremental Refresh for Dataflows or Datamarts

Recall that dataflows are ETL processes in the cloud for the Power BI Service. You can set up the Incremental Refresh for dataflows, and it is even easier to do. You don't need to create the RangeStart and RangeEnd parameters. You just go to the Incremental dataflow Refresh setting directly, as shown in Figure 18-22.

ENTITY NAME	ENTITY TYPE	ACTIONS
ServiceCalls	Custom	
ServiceCallsAggregated	Custom	
Daily ServiceCalls	Custom	
Accounts	Custom	
Date	Custom	

**Figure 18-22.** Setting up Incremental Refresh for dataflows is relatively simple

You can see in Figure 18-23 that the same Incremental Refresh settings are available for dataflows.



**Figure 18-23.** Incremental Refresh settings available for dataflows

A similar Incremental Refresh setting can be configured for datamarts.

## Summary

Setting up Incremental Refresh in Power BI allows you to load only part of the data regularly and store the unchanged data. This process makes your refresh time much faster. However, there are some requirements for it. You must have a date field in your table. It is recommended only when the data source supports query folding. You can do this configuration on a Power BI dataset or in Power BI dataflows and datamarts. If you do this on a dataset, after publishing the dataset, you cannot download the PBIX file.

Hybrid tables can be an addition to your Incremental Refresh setup. They are helpful in accessing the most up-to-date data using the DirectQuery partition while the historical data is stored as Import Data in other partitions. From the Power BI point of view, these are all part of a single table.

## CHAPTER 19



# Development Best Practices

Regardless of your job function (developer, consultant, or architect), following certain practices with Power BI ensures a good quality solution. This chapter explains some of those best practices, including why they are helpful and links for how to use them. These tips are related to developing a Power BI solution, not deploying, publishing, or sharing the solution.

## Reuse Tables Generated in Power Query

If you are using Power Query to connect to the data source and transform the data, then it is likely that you will create a table that you may need in another file in the future. If you use Power Query inside the Power BI Desktop for transformation, then reusing the table in other PBIX files will be a challenge. Your only choice would be to copy and paste the codes into the new file. This would create another issue—redundancy of the Power Query code.

The proper way to connect to the data source and reuse data in other Power BI files is to use dataflows. Dataflows run on the cloud, and the data is stored in a destination such as Azure Data Lake Storage or a dataverse. If a table is generated in the Power Query of the dataflow, it can be used easily in multiple Power BI files.

The wrong way to reuse tables: copying and pasting Power Query tables between PBIX files.

The right way to reuse tables: creating Power Query tables in dataflow and getting data from them in PBIX files.

## Reuse DAX Calculations

DAX is the language for writing calculations in Power BI. You can use DAX to write calculations such as year-over-year changes and percentages, percentage of the total, or rank of customers by their yearly revenue. Writing calculations in DAX takes time, and you may likely need to reuse a calculation in multiple reports.

Creating copies of the PBIX file every time you need to reuse a calculation is not ideal. The better approach is to create a shared dataset with the DAX calculations and then create thin reports with live connections to that shared Power BI dataset. Using a shared dataset ensures that all the reports use the same DAX calculations. If you need to make a change, it is only necessary in the shared dataset. Maintaining a solution like this is much easier.

The wrong way to reuse DAX calculations: copying and pasting them between PBIX files.

The right way to reuse DAX calculations: creating a shared Power BI dataset and connecting to it live as a Power BI thin report.

## Use Power Query Functions for Reusable Data Transformation

There are likely a specific set of data transformation steps for multiple data sources. For example, the Sales data has the same structure for all the branches of a company. Each Sales field is in a different database (or in a different Excel or CSV file). You can develop a set of data transformations and reuse them inside Power Query using custom functions.

Power Query custom functions are reusable pieces of Power Query transformation steps. A function can have input parameters (or it might not have any parameters if it is a generator function) and can generate output. The output of the custom function can then be used in other queries.

Custom functions reduce the need for rewriting a piece of code or transformation. As a result, maintaining that code is easier. If you need to make a change, you only have to do that inside the Power Query custom function. You can reuse the custom function in multiple places inside the Power BI file.

If you use a custom connector, you can include the custom function in it so that you can use that custom function even in multiple PBIX files.

The wrong way: copying and pasting the Power Query transformation between tables.

The right way: creating a Power Query custom function and reusing it in multiple places.

## Parametrize the Data Transformation Process Using Power Query Parameters

Your data source might change from one server to another, or the folder that Power BI gets the Excel files from moves to a shared folder on a server, or the email address that Power BI gets data from Exchange might change. In any of these cases, changing the data source from the previous value to the new value is a hassle. You would have to open the Power BI file in the Power BI Desktop, change the data source settings, or even go into Power Query Editor to make such changes.

If you use Power Query parameters where you think the value might change, you can change the parameter value, even outside of Power Query. If your file is already published to the service, then under the Dataset Settings, there is a place where you can change the values of parameters before the next refresh of the Power BI dataset.

Using Power Query parameters enables you to parameterize the values in the data transformation. These can be the data source names, paths, server names, table and column names, and many other values that are likely to change.

The wrong way: using values (such as server names, folder paths, and other values that are likely to change) directly in Power Query.

The right way: creating Power Query parameters for values likely to change and using parameters in Power Query.

## Categorize Power Query Using Power Query Groups (or Folders)

Categorize Power Query objects (tables, parameters, functions, lists, records, and so on) in folders. In Power Query, these folders are called *groups*. You can create as many groups as you want. The method by which you create the group is dependent on you. Some prefer creating groups per data source (such as all the tables from SQL Server data source under one folder), and others prefer creating groups per table functions (such as all product tables from all data sources under one folder). Some generic groups, such as final tables (the group that includes all enable-load queries) or temp tables (the group that includes all disable-load queries) are also common.

Be sure to determine your standard for defining groups and use them to categorize your Power Query object structure.

## Disable the Load of Temp Tables in Power Query

Power BI loads everything into the memory. If you have a table in Power Query that you are not using directly, but are using it to load data into another table, consider disabling its load. For example, suppose you are building a product table by merging a product category, subcategory, and product details. In that case, you can disable those three tables' loads, especially if the data that you need will be available in the product table at the end.

Disabling the load of a table won't load it into the memory of the Power BI dataset. The table will still be part of the refresh process in Power Query. Disabling the load of temp tables leads to a significant performance improvement in the Power BI model. Adding unnecessary tables to Power BI not only affects performance, but it also makes the model complicated and confusing (you may select the Product ID from a table that is not the primary product table).

## Only Load What You Need for Reporting: Filter the Data

If you are loading 20 years of sales transactions just in the hopes that someday, someone will use them in visualization, rethink your approach. It might be better to filter this data for the last few years that you need (such as the last three years) and then create a Power Query parameter to change it if required.

Reducing the amount of data loaded into Power BI leads to a significant performance improvement. Just load the data that you need.

The same rule also applies to tables with hundreds of columns. If you need five columns from a table, don't import the entire table. This is a common mistake when getting data from CRM or ERP systems, where the tables are wide. You can use Power Query to load more columns or tables in the future if needed.

## Use Reference and Duplicate in Their Rightful Places

There are two ways to copy a table in Power Query—Duplicate and Reference. Duplicate creates an identical copy of the table without a link to the original table. You can change this new table without any concerns about the main query. Reference keeps the transformation in the primary table, and the new table is a link to the original table. This is used when you want to apply a set of transformations on a currently existing query and still follow the transformations of the primary query. Choosing between Reference and Duplicate is a crucial choice.

## Multi-Layer Architecture Everywhere

A best practice when reusing components is to design them in layers. When you have a component in a layer, you can replace it easily, change it easily without needing heavy maintenance, and reuse it easily. Dataflows used for shared Power Query tables or shared datasets for Power BI thin reports are examples of multi-layer architecture.

However, the multi-layer architecture is not just about using dataflows and datasets. Anywhere in your architecture, think of methods that make reusability easier. For example, if the data source is likely to change from SQL Server or Oracle, it might make the change easier if you create a data staging layer using dataflows. The staging layer can then be easily changed without much change needed in the transformations or other layers.

Datasets can be in layers themselves. If you get data from a dataset and apply some changes to it, you've created a chained dataset. This way, you are creating layers of Power BI datasets.

## Sync Slicers

If you are using the same slicer across multiple pages, it makes sense to sync slicers. Syncing slicers means that if you change the values of the slicer on one page, you don't have to redo that change on another page. The slicers will be synced.

Syncing slicers can also help when creating a slicer page. This is particularly helpful if the number of fields you want to define a slicer on is too many to be part of a report page. In this case, you can create a single page with all the slicers, then sync it with the other pages and use buttons and bookmarks to go back and forth to the slicer page.

## Use a Theme File

Many organizations use company color codes . You may also want to set some predefined styles for font sizes, colors, borders, and other visual effects in your report. It is not good to set these for each page and visual one by one. Maintaining a solution like that wouldn't be easy, and if someone else wants to follow your standards, they would have a hard time doing so.

You can instead create a Power BI theme for your visual standards. A Power BI theme can include colors, fonts, and other generic visual properties you want to synchronize across the pages. The theme can then be stored as a JSON file and be reused for other PBIX files. All you need to do to reuse it is to browse for the theme and apply it.

If you create a theme for a team of developers, it is a good idea to keep the file in a shared folder so that everyone can access and reuse it.

## Use a Background Image for Report Pages

In addition to the theme file, it is common to use background images for report pages. You can design a background image with the border, color, header, and footer you want and design sections for visual settings. Then this background image can easily be used in report pages to give them a professional yet consistent look. The report user won't be distracted when moving from one page to another with a different look. Your entire report will look like an application, with a constant look and feel.

## Incorporate Conditional Formatting Using DAX, Parameters, and Tables

Visualization is more than just beautiful charts and graphs; it is the art of conveying the right message to the users. The right message can sometimes be passed by color-coding values of a visual like a table or matrix visual. In Power BI, you can use conditional formatting to make some values more visible than others by color coding them.

Conditional formatting can be done by hard-coding values directly into the visual configuration (which is not recommended), or it can be done using DAX measures, parameter tables, and what-if parameters. The latter is a better approach for conditional formatting, because if you do the conditional formatting based on a DAX measure, you can use the same DAX measure in other pages and visuals and have a consistent look across your reports. The color codes can be stored in tables for more straightforward configuration and maintenance too.

## Create a Measure Table

Measures can get lost inside a Power BI data model that has many tables. It can sometimes be hard to determine which table you created measures in. If you cannot remember the measure name, you will have to expand each table and scan the measure names gradually. This can be avoided by using a measure table.

A measure table is a blank table (which can be created anywhere—Power Query, data source, DAX, and so on) with no data. The valuable objects in this table are measures. Once you create this table, you can move all the measures in it. This way, you can find the measures all in one place.

You can also use the display folders in addition to the Measure table. Display folders give you folders and subfolders for better categorization.

## Create and Use Hierarchies

Suppose you have a product category, subcategory, and product name. You may want to combine these three fields in multiple visuals. Perhaps in some of them, you want to have the ability to drill down in these three levels. Instead of going to each product and dragging and dropping these three fields one by one, you can create a hierarchy of these three fields and reuse that hierarchy in other visuals.

Although you can achieve the same outcome by dragging and dropping the three fields separately, creating the hierarchy will give you consistency across your report and will make it easier to use the fields in other visuals. If you want to use two levels of hierarchy in one visual, you can easily adjust them at the visual level. Hierarchies are part of the data model and they unify field use.

## Set Auto Summarization at the Field Level

Numeric fields are automatically aggregated in Power BI unless they are used in a relationship or some other conditions are applied to them. This is very helpful for fact fields, such as Sales and Budget fields. However, If you use a Date table with the Year column numeric, or if you have a parameter table with the Age column on it, you won't want the Year or the Age columns to be automatically summarized when using it in Power BI reports.

It is best to set the auto summarization (or auto aggregation) on each field, based on that field at the model level. That way, when you use the field in visuals, they follow the right summarization. This is a simple configuration to apply, but it helps in future reporting and visualization.

## Consider Using a Custom Date Table

Although Power BI comes with a default Date table, for many advanced use cases, you will need extensions of a Date table. If you want to slice and dice data by weekdays and weekends, run holiday data analysis, or investigate many other scenarios, they require a custom Date table.

You can create the custom Date table using Power Query, DAX, or even the data source. If you use Power Query to create a custom Date table, it would be better to do it in a dataflow so that you can reuse it in other files. Once you create the custom Date table, make sure to mark it as a Date table so that the Time Intelligence calculations work correctly. Also disable the Auto-Date/Time option in Power BI.

## Design Mobile Reports

Power BI reports are designed for interactivity. The new era of reporting and dashboarding does not rely on offline reporting methods, such as printed papers. Nowadays, using technologies such as Power BI, report users can access live, interactive reports and dashboards from their mobile devices everywhere. This not only reduces the need for printing, but it also helps users contribute to the reports and send feedback to the developers.

Although Power BI reports are mobile-friendly by default, in order to view them in their best mode on a mobile device, the report developer has to design the mobile view of the report and dashboard. This is a simple yet effective process. Report pages can have different configurations and settings for desktop and mobile views (such as different font sizes and layouts).

Creating mobile-friendly views for your reports and dashboards is a big step toward the adoption of Power BI in your organization.

## Use Aggregations for Big Tables

If you are working with large tables and the reports are slow because of the enormous amount of rows in tables, you need to consider using aggregations. Aggregations can be done on top of large tables (DirectQuery or Import Data tables). Aggregation is a grouped version of the original table based on just a few columns. This grouped table (aggregated table) is a layer between the visualization and the main table. Because the aggregated table is smaller, querying data from it is faster. If the visualization values can be fetched from the aggregated table, then the aggregated table will be used. Otherwise, the main table will be used. This way, calculations and visualizations are faster. You can have multiple layers of aggregations to support different visualizations.

Defining aggregations on big tables in Power BI is a crucial performance-tuning step in Power BI modeling. Aggregations can also be automatically generated based on field use. The auto-aggregation feature can be enabled as a Power BI Premium functionality.

## Use Incremental Refresh and Hybrid Tables When Possible

Loading all the data of a table might take a long time, especially if the table has many rows. Instead of reloading the entire data every time, you can reload only the data that has changed, and the historical data can be loaded once. This process is called Incremental Refresh. Using Incremental Refresh, you can load historical data (let's say for the past ten years) only once and then reload the last period (let's say the last year) every time the dataset refreshes. This process makes the dataset refresh time much faster. This is mainly helpful with the data sources that support query folding because, in that case, instead of reading the many years of data, you only read and transfer one year of data each time.

Hybrid tables can also accompany Incremental Refresh. Hybrid tables store part of their data in real-time using DirectQuery to the source table (usually the most recent period of data) and the remaining data is imported. Hybrid tables are suitable for scenarios where data freshness is also needed in a table with many rows and where Incremental Refresh is already set up.

## Design Star-Schema Models

Designing the structure of tables and their relationships is one of the primary tasks of every BI system. A common best practice in designing the table structure is called Star-Schema. In this type of design, tables are categorized into two types—Dimension and Fact tables. Fact tables include the numeric and additive measures and the dimensions, including the descriptive fields that slice and dice the data of the fact table. The relationship between Dimension and Fact tables is one-to-many from the Dimension table to Fact table. There can be multiple dimensions per Fact table, and there can be multiple Fact tables inside the model. The design of Dimension and Fact tables is dependent on the reporting requirements.

## Avoid Both-Directional Relationships

A vital performance consideration in a Power BI dataset is avoiding both-directional relationships. The direction of a relationship in Power BI is how the filter propagates between the tables. Sometimes, to get the values of a field from another field in another table, you may feel the need to change the direction of the relationship. However, changing a relationship to both-directional comes at the cost of performance, as well as ambiguity and confusion in the data model.

A star-schema-designed model usually doesn't need both-directional relationships. However, on rare occasions, if required, it can be evaluated separately, and the solution can be designed without a both-directional relationship. Using DAX functions such as *CrossFilter* inside a measure can also be a remedy for scenarios where the both-directional relationship seems your only recourse. Using the measure instead of changing the direction of the relationship can result in better performance, because that measure might not be used in all the report pages, so the performance reduction may not occur all the time.

## Clean Up Your Models

It is a good idea to review your models over time. In the review, you may find fields that are not used anymore and measures and calculations that are created for test purposes and can be removed. Cleaning up your model is a task that can be done over time. Fortunately, many community tools can help you with this process. I use Power BI Helper, which is a free tool for this purpose; it has many other helpful Power BI development features.

# CHAPTER 20

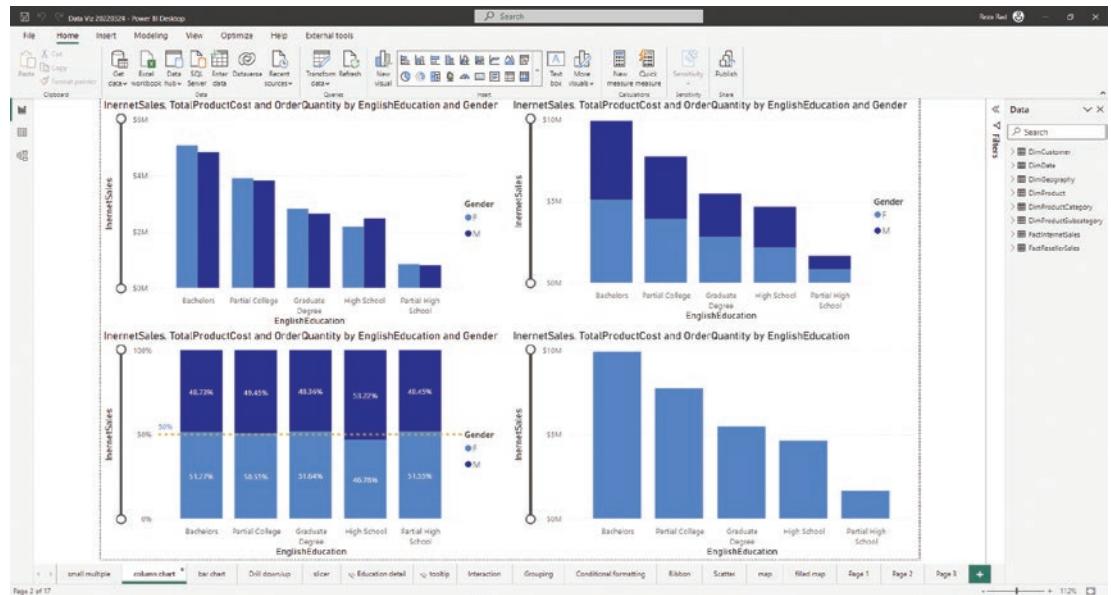


# The Power BI Service

The Power BI toolset comes in many shapes and forms. There is a Power BI Desktop application, a Power BI mobile app, the Power BI Report Server, and the Power BI Service (along with some other applications and components). This chapter explains the Power BI Desktop, Power BI reports, the Power BI Service, and more.

## The Power BI Desktop: A Report Authoring Tool

The report authoring experience in Power BI is usually performed in a desktop application called the Power BI Desktop. Note that Microsoft is working hard to create a similar experience of it as a web version. You can install the Power BI Desktop on your machine for free. You don't pay a cent to use it. You don't even need an account. All you need to do is download and install it. Figure 20-1 offers a look at the application in action.



**Figure 20-1.** The Power BI Desktop is for authoring and developing reports

The Power BI Desktop has everything you need to create reports. However, the next step after creating the report is sharing it with others, even if that means just simply showing the results to others. You might also want to make your reports available on other devices, such as your mobile device.

Power BI files created in the Power BI Desktop application can be saved as \*.PBIX files and then reopened using the Power BI Desktop application. However, you should not use that method to share the Power BI report with users. That would require them to install the Power BI Desktop on their machines, and there are some problems with that:

- The Power BI Desktop is a report authoring tool. Even if the users can install it, they would have too much power. They could edit and change the report. Then they would come to you to fix their problem.
- Users are meant to use the report everywhere—on their mobile devices, tablets, and so on. The Power BI Desktop cannot be installed on all of these tools.

## Hosting Options for Power BI Reports

To share a Power BI report correctly, you must host it somewhere. Then the hosted report can be viewed by users using a web browser or the Power BI mobile application. This is the right way to share reports and would not lead to any of the problems mentioned in the last section.

There are two hosting options for Power BI reports—cloud-based hosting (called the Power BI Service or website) and on-premises hosting (called Power BI Report Server). See Figure 20-2.



**Figure 20-2.** The Power BI Service is a cloud-based app that is accessible from all devices

## What Is the Power BI Service?

The Power BI Service (or, as some call it, the Power BI website) is the cloud-based hosting environment for Power BI reports, datasets, dashboards, and more. This cloud-hosting environment is provided by Microsoft and is part of the Microsoft cloud service offering (in Azure and Microsoft 365). You can see the portal in Figure 20-3.

The screenshot shows the Power BI Service portal. On the left is a navigation sidebar with options like Home, Create, Browse, Data hub, Metrics, Apps, Deployment pipelines, Learn, Workspaces, and My workspace. The main area has a greeting "Good afternoon, Reza" and a subtitle "Find and share actionable insights to make data-driven decisions". Below this is a "Recommended" section with five cards: "You recently open this" (user icon), "Reza Rad opened this" (bar chart icon), "Popular in your org" (bar chart icon), "You recently open this" (document icon), and "Reza Rad opened this" (bar chart icon). At the bottom is a table titled "Recent" showing the following data:

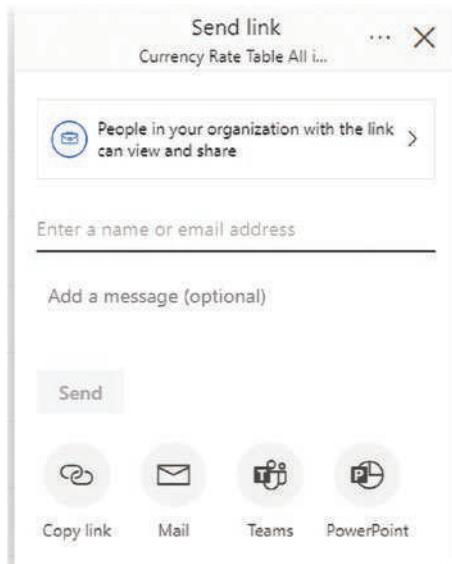
Name	Type	Opened	Location	Endorsement	Sensitivity
[Icon] [Name]	Workspace	2d ago	My workspace	—	—
[Icon] [Name]	Dashboard	4m ago	—	—	—
[Icon] [Name]	Report	4m ago	My workspace	—	—
[Icon] [Name]	Workspace	4m ago	My workspace	—	—
[Icon] [Name]	App	2 months ago	Apps	—	—
[Icon] [Name]	Registered Report	2 months ago	In	—	—

**Figure 20-3.** The Power BI Service portal

In the Power BI Service, organizations are separated using tenants. Tenants can be managed by using the Azure Active Directory portal or the Microsoft 365 portal. Under tenants, there are users. These users are Azure Active Directory users.

You can access the Power BI service at [app.powerbi.com/](http://app.powerbi.com/).

In addition to users and authentications, there is the concept of workspaces in the Power BI Service. A *workspace* is like a shared folder between a team of users. This can be a place to share some of the Power BI content. There are many ways to share Power BI content that is published to the Power BI service. Figure 20-4 shows just one of those sharing options.

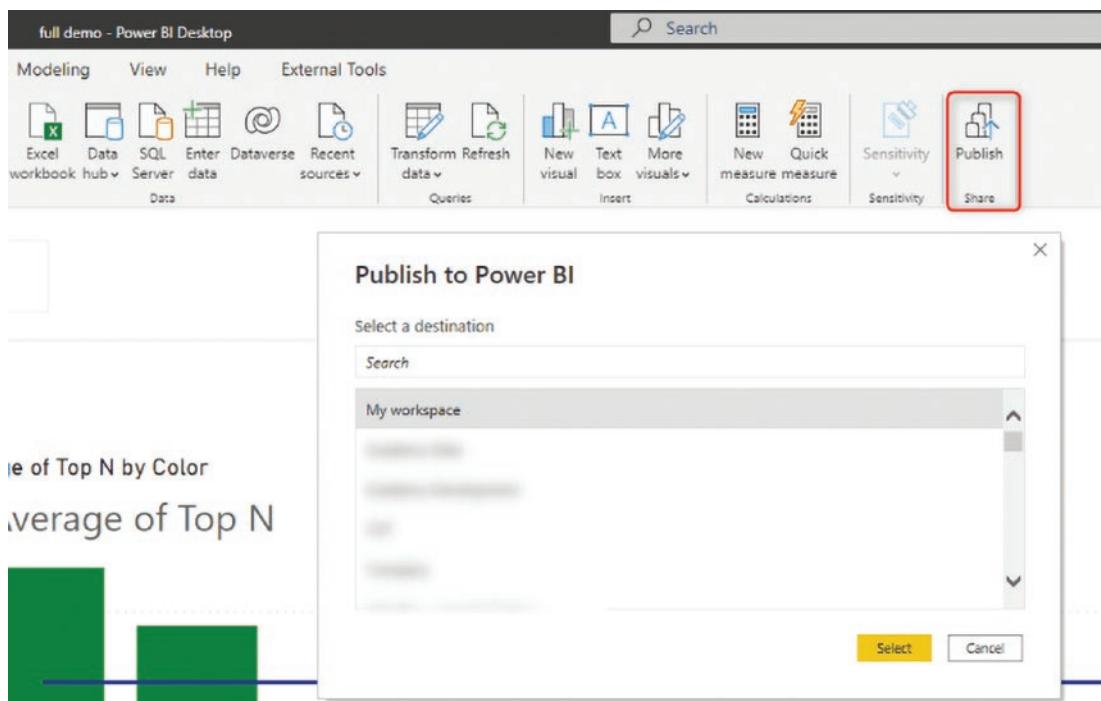


**Figure 20-4.** Sharing content in the Power BI Service

The Power BI Service allows security to be applied at the workspace or object level. There are tenant configurations and settings that can be applied to govern the entire organization's use of Power BI objects.

## Publish Reports to the Service

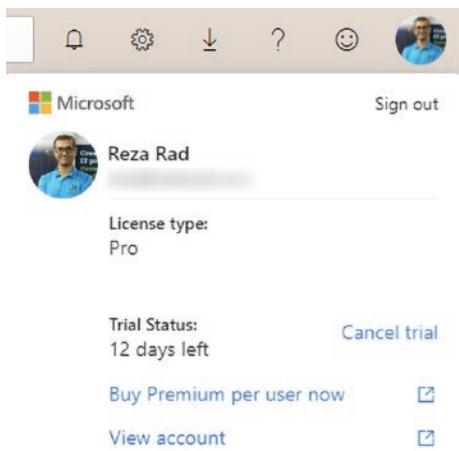
Reports created in the Power BI Desktop can easily be published to the Power BI Service, as shown in Figure 20-5. You can even create objects directly in the Power BI Service, such as dashboards as landing pages for multiple reports.



**Figure 20-5.** Publishing to the Power BI Service

## Capacity-Based or User-Based

To use the Power BI Service, you must have a Power BI license (see Figure 20-6). There are two types of licenses for Power BI—capacity-based and user-based licensing. Capacity-based licensing is usually better for organizations with hundreds of users or more, and user-based licensing is good for small- to medium-sized businesses. Although, for some of the operations, in addition to the capacity-based licensing, you would also need to get user-based licensing.



**Figure 20-6.** User profile and Power BI licensing

## The Power BI Service

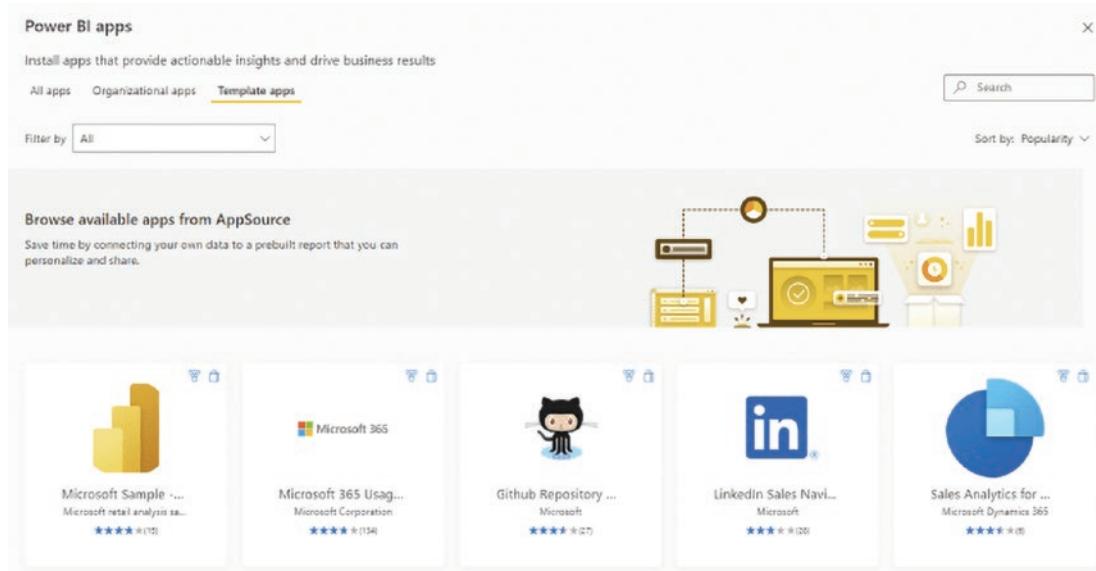
The Power BI Service is not just a place to host reports. It does much more than that. Many of the enterprise features of Power BI were developed for the Power BI Service. Among these are components such as:

- Dataflows
- Shared datasets
- Datamarts
- Metrics
- Data lineage
- Content certification
- Sensitivity labels
- Deployment pipelines
- Workspaces
- Apps

These components and features are Power BI Service-only features. They enable you to better govern Power BI adoption and provide a better architecture for the Power BI implementation. However, to use some of these, you may need premium licensing. In the following sections, some of the most important items are explained.

## Template Apps

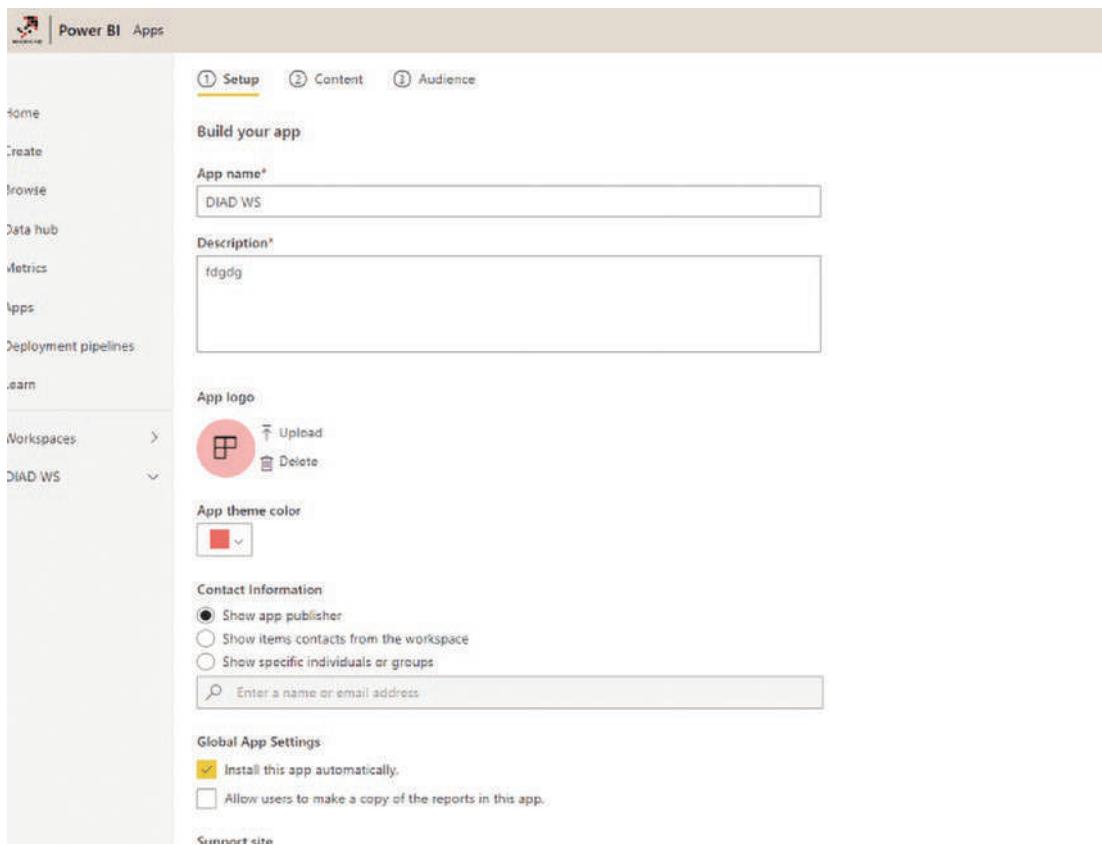
One of the more useful features available in the Power BI Service are template apps. Template apps are prebuilt Power BI reports and data models. Third parties have created these prebuilt reports. You may need to purchase some of them, but many are free. These templates, some of which are shown in Figure 20-7, enable you to connect to your data source without needing to create the report and get the analysis ready.



**Figure 20-7.** Power BI template apps

## Apps

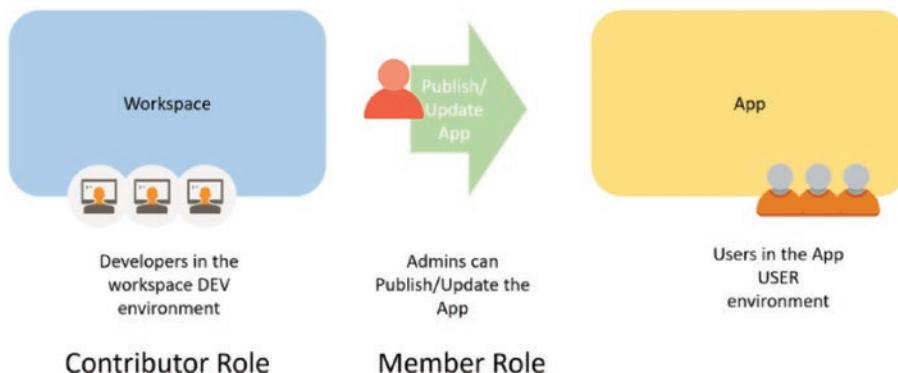
Delivering the Power BI content to the user can be organized in packages called apps. Each app can contain multiple Power BI reports and other content. As shown in Figure 20-8, the app can be designed with a background color and theme, and you can set the landing page and audience when creating the app.



**Figure 20-8.** A Power BI app

## Workspaces

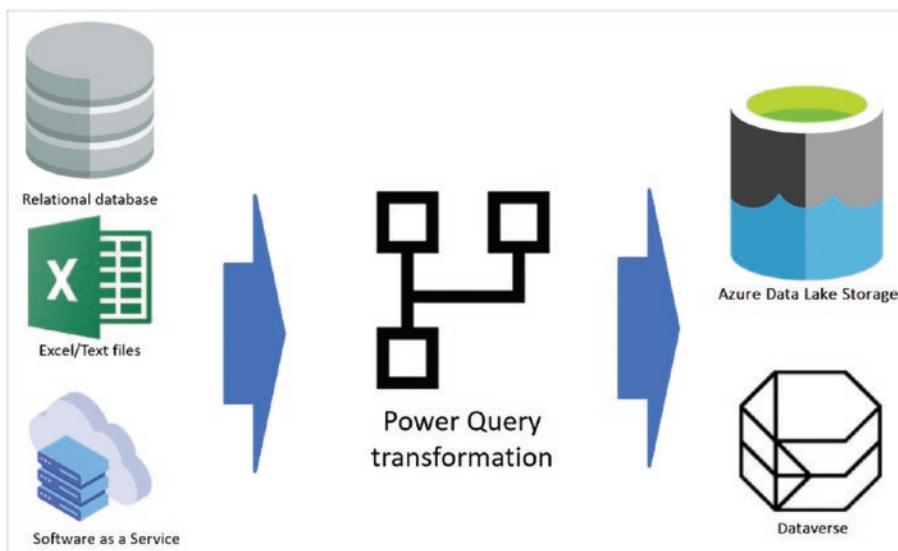
Power BI workspaces are collaborative environments for teams using the Power BI Service. This is where teams can share Power BI content, review and edit each other's work, and deploy the changes to other environments. Each member of a team can have different levels of access in these shared environments, depending on their roles. See Figure 20-9.



**Figure 20-9.** Power BI workspace

## Dataflows

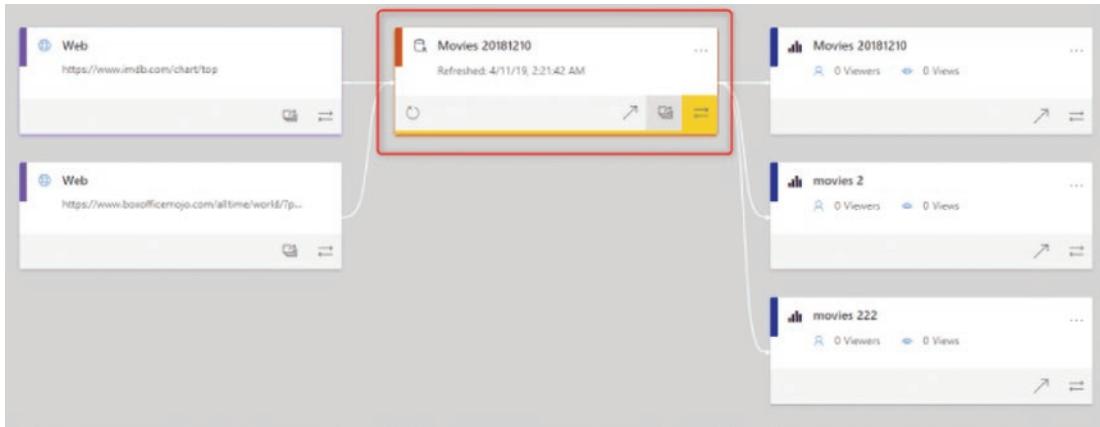
Dataflows are the cloud-based ETL operations that use the Power Query engine. Instead of copying the data transformation between multiple Power BI files, you can use dataflows for shared tables as a central place for data transformation and to store the data. Power BI files can get data from the dataflow. Dataflows play an important role in a multi-layered Power BI architecture, as illustrated in Figure 20-10.



**Figure 20-10.** Power BI dataflow

## Shared Datasets

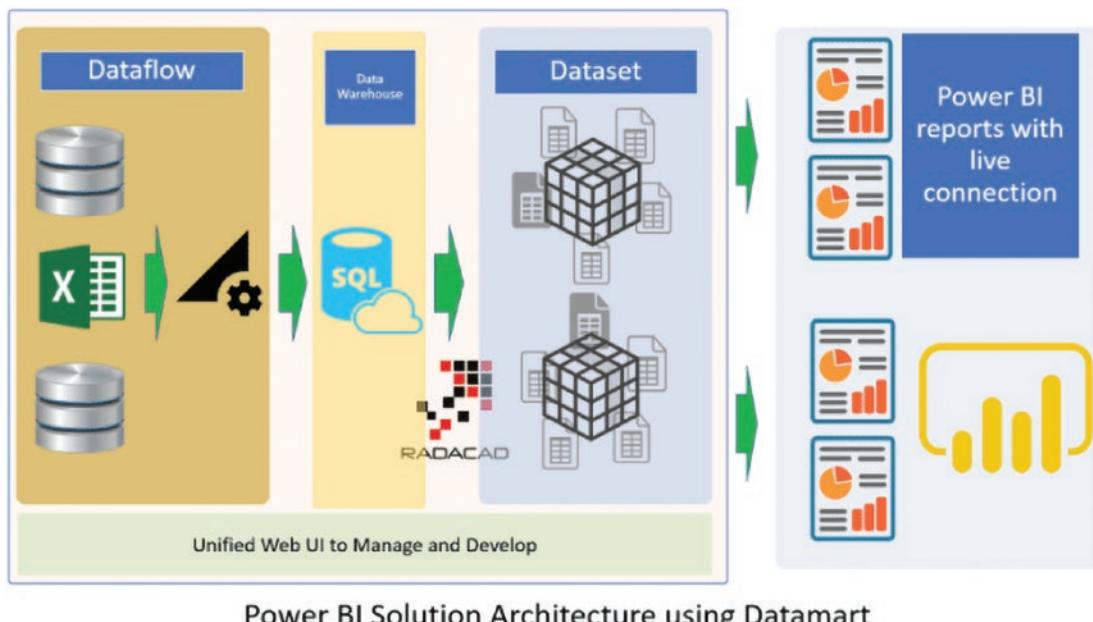
Once a dataset is published to the Power BI Service, it can be reused to create a new thin report. A thin report is a report without a dataset, or in other words, a report that connects live to an existing Power BI dataset or other tabular data model. When you use a shared dataset, you can reuse the calculations and modeling done in that dataset in multiple reports. Shared datasets are another important component of a multi-layered Power BI architecture, as illustrated in Figure 20-11.



**Figure 20-11.** Power BI shared datasets

## Datamarts

Datamarts make modeling Power BI easier to manage by bringing all the configurations into the Power BI Service with one unified editor to create the dataflows and datasets together, as shown in Figure 20-12. Internally, the data is stored in an Azure SQL database, which comes as part of your Power BI licensing. You do not need to license the database separately. Datamarts are going to be the next-generation way to build Power BI data models.

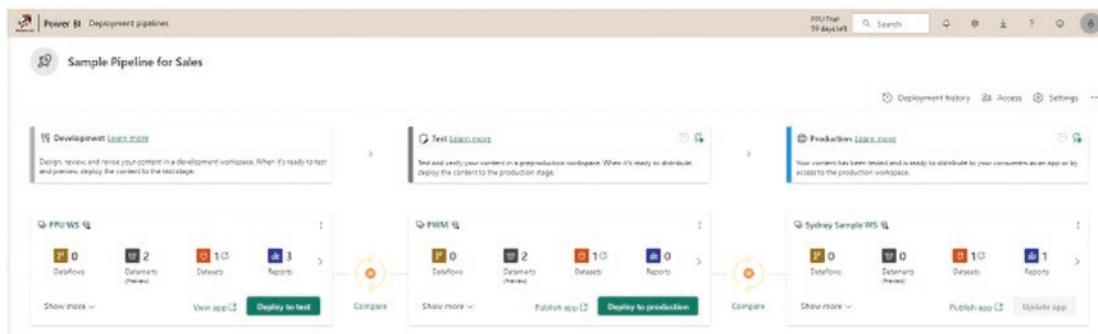


**Power BI Solution Architecture using Datamart**

**Figure 20-12.** A Power BI datamart in the Power BI multi-layered architecture

## Deployment Pipelines

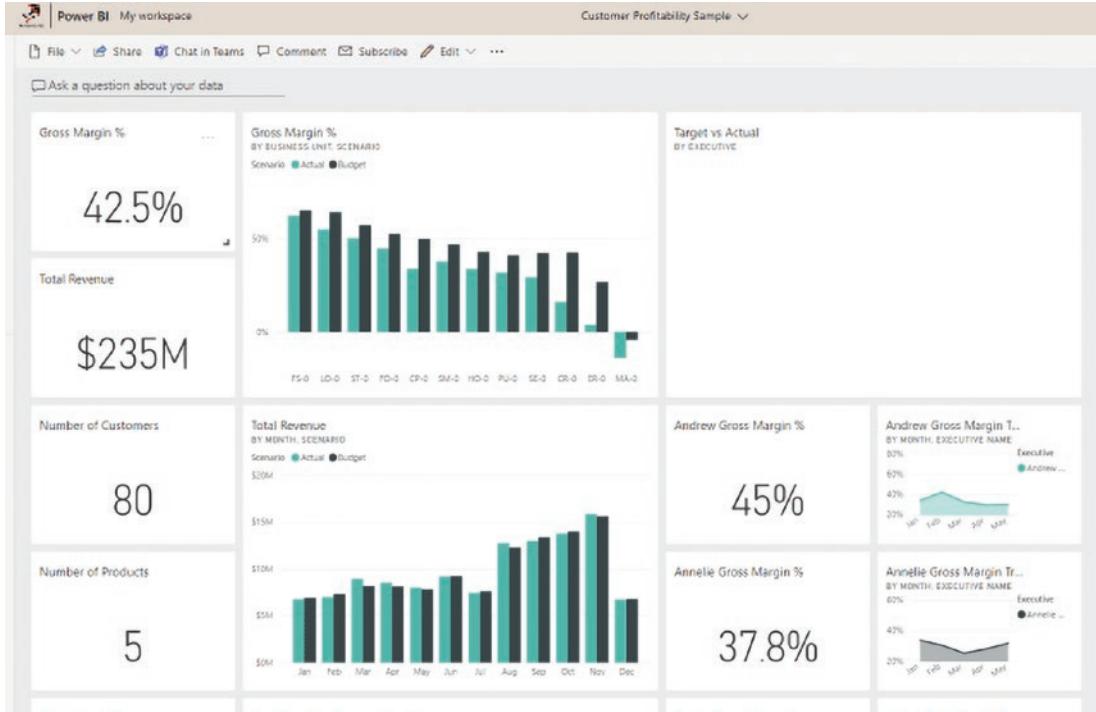
To ensure that users aren't impacted by the changes done by developers immediately, it is essential to separate the development environment from the user environment. In some organizations, there are often three environments with names like Dev, Test, and Production. When you have multiple items in your Dev environment, pushing the changes into the next environment (Test or Production) can be managed easily using deployment pipelines. Deployment pipelines consider your Dev, Test, and Production workspaces and compare the changes between these environments. This process makes deployment easier to control. See Figure 20-13.



**Figure 20-13.** Deployment pipelines

## Dashboards

The Power BI Service offers another layer of visualization called the dashboard. Dashboards can be the landing pages for multiple reports or they can be used for real-time visualizations (see Figure 20-14).

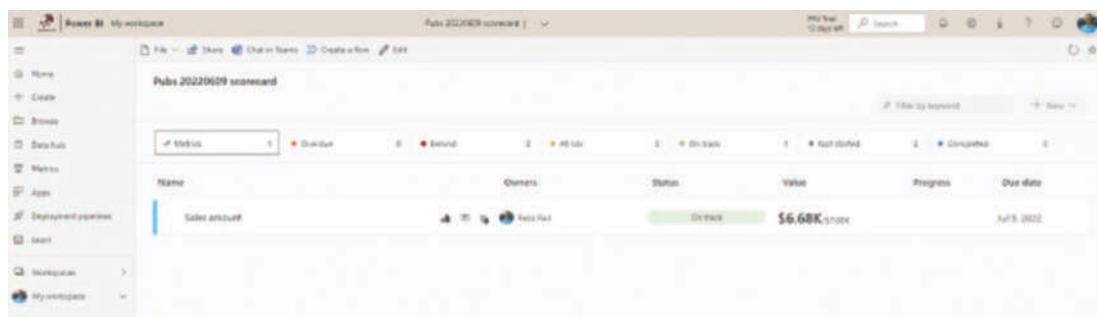


**Figure 20-14.** A Power BI dashboard

Using dashboards is optional; however, they come with additional options such as data alerts and auto-refresh of tiles.

## Metrics

Power BI has a comprehensive component for showing KPIs (Key Performance Indicators) and scorecards. This component was previously called Power BI Goals; now it's called Power BI Metrics. A sample metrics landing page is shown in Figure 20-15. Using Power BI Metrics, you can view all aspects of a KPI, such as its values, targets, trends, statuses, and rules that define whether the value is on track or behind. The metric can use a Power BI shared dataset as its source.



**Figure 20-15.** Power BI Metrics

## Content Certification

The Power BI Service also offers some data governance settings and configurations. One of these is called *content certification*. Power BI objects such as datasets, dataflows, reports, and apps can be promoted for others to use, and even certified by a certain group in the organization. As shown in Figure 20-16, the Power BI tenant administrator can determine which members can certify the content. This helps in the adoption and usage of content and helps other users find reliable content.

### Certification

*Enabled for the entire organization*

Allow users in this org to certify datasets, dataflows, reports, and apps.

Note: When a user certifies an item, their contact details will be visible along with the certification badge.



Specify URL for documentation page

Enter URL

Apply to:

The entire organization

Specific security groups

Except specific security groups

Apply

Cancel

**Figure 20-16.** Power BI content certification

## Data Lineage

Because there are many data-related components available in the Power BI Service, it's easy to be confused about where the data is coming from—which dataset and dataflow fetches that data and from which data source. The Power BI Service offers a Data Lineage view, shown in Figure 20-17, that can be used to see these details. The Data Lineage view is still a work in progress and I believe it will be enhanced in the future, as some cases aren't yet supported.

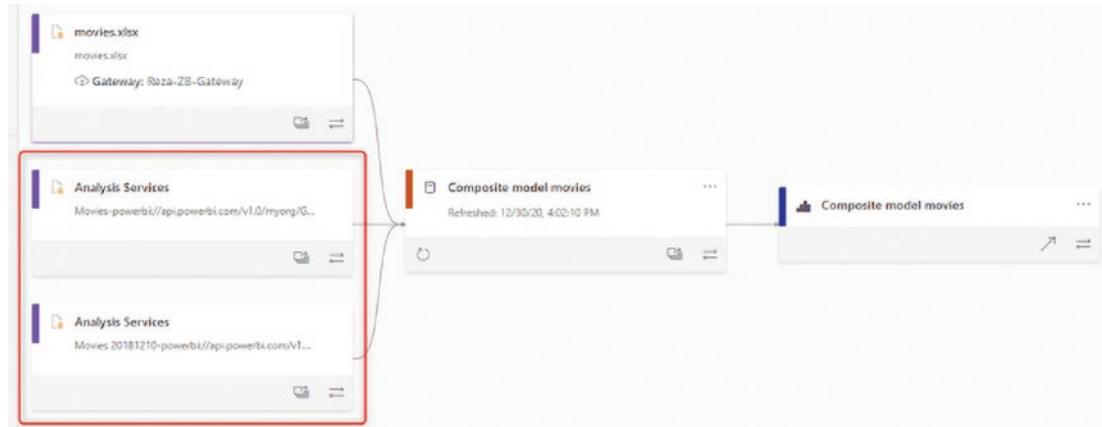


Figure 20-17. Power BI Data Lineage view

## Summary

The Power BI Service is one of two hosting options for Power BI content. The Power BI Service hosts the Power BI content in the cloud and is used in the majority of cases, instead of the Power BI Report Server (which is the on-premises hosting solution for Power BI content). The Power BI Service comes with all the features and configurations needed for a hosting platform (user access, tenant-level configurations, workspaces, categorizing the content, sharing, and more). It also comes with extra useful features and components. These extra components play an important role in Power BI adoption and implementation. Some of these components are dataflows, shared datasets, datamarts, metrics, dashboards, deployment pipelines, content certification, workspaces, and apps. These extra components and the regular updates to the Power BI Service make it an important part of Power BI adoption in many organizations.

## CHAPTER 21



# The Power BI Report Server

Power BI is not only a cloud-based reporting technology. Due to the demand to have data and reporting solutions on-premises, Power BI also has the option to be deployed on-premises. Power BI on-premises hosting is handled via the Power BI Report Server. This chapter covers using Power BI in a fully on-premises solution with the Power BI Report Server.

This chapter will teach you everything you need to know about the on-premises world of Power BI. You learn how to install the Power BI Report Server, learn all the requirements and configurations for the Power BI Report Server to work correctly, and learn about all the pros and cons of this solution. At the end of this chapter, you will be able to decide if Power BI on-premises is the right choice for you, and if it is, you will be able to set a Power BI on-premises solution up and run it.

## What Is the Power BI Report Server?

Power BI reports can be hosted in two environments—cloud-based and on-premises. The cloud-based Power BI hosting is called the Power BI Service. The on-premises option is called the Power BI Report Server.

The Power BI Report Server is a specific edition of SQL Server Reporting Services that can host Power BI reports. To run Power BI Report Server, you don't need an SQL Server installation disk; the Report Server comes with its own setup files. You can download the setup files, which is explained in the next section. The Power BI Report Server can host Power BI Reports and Reporting Services (SSRS) Reports, also called *paginated reports*.

With the Power BI Report Server, there will be an instance of the Power BI Desktop installation. The Power BI Desktop edition that comes with the Report Server is used to create Power BI reports. Otherwise, reports cannot be hosted on the report server. The good news is that the Power BI Desktop Report Server edition is regularly updated, and its experience will be very similar to the Power BI Desktop.

## Requirements for Setting It Up

You need to download the latest edition of the Power BI Report Server from this link, as shown in Figure 21-1:

<https://powerbi.microsoft.com/en-us/report-server/>

You will have two installation items—the Power BI Report Server and the Power BI Desktop Report Server edition (which comes in 32-bit and 64-bit versions).

### Microsoft Power BI Report Server- September 2022

The screenshot shows the Microsoft Power BI Report Server download page. At the top, there is a note: "Important! Selecting a language below will dynamically change the complete page content to that language." Below this is a "Select Language:" dropdown menu set to "English" and a large red "Download" button. A dashed-line box highlights the "Details" section, which contains a note about multiple files and their sizes:

File Name:	Date Published:
PBIDesktopRS.msi	10/5/2022
PBIDesktopRS_x64.msi	305.2 MB
PowerBIReportServer.exe	344.4 MB
	401.9 MB

**Figure 21-1.** Downloading the Power BI Report Server

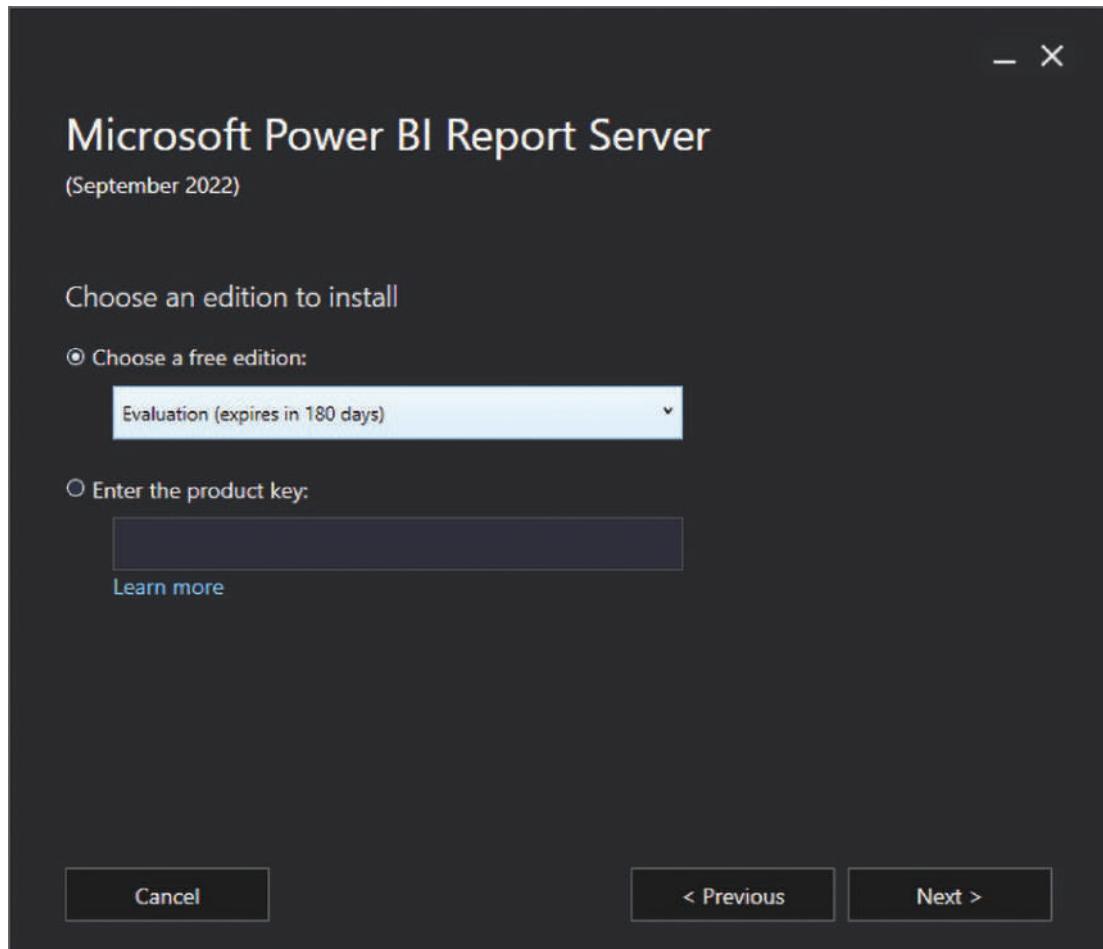
## Installing the Power BI Report Server

Setting up the Power BI Report Server is simple. You just run the setup file and continue the instructions, as shown in Figure 21-2.



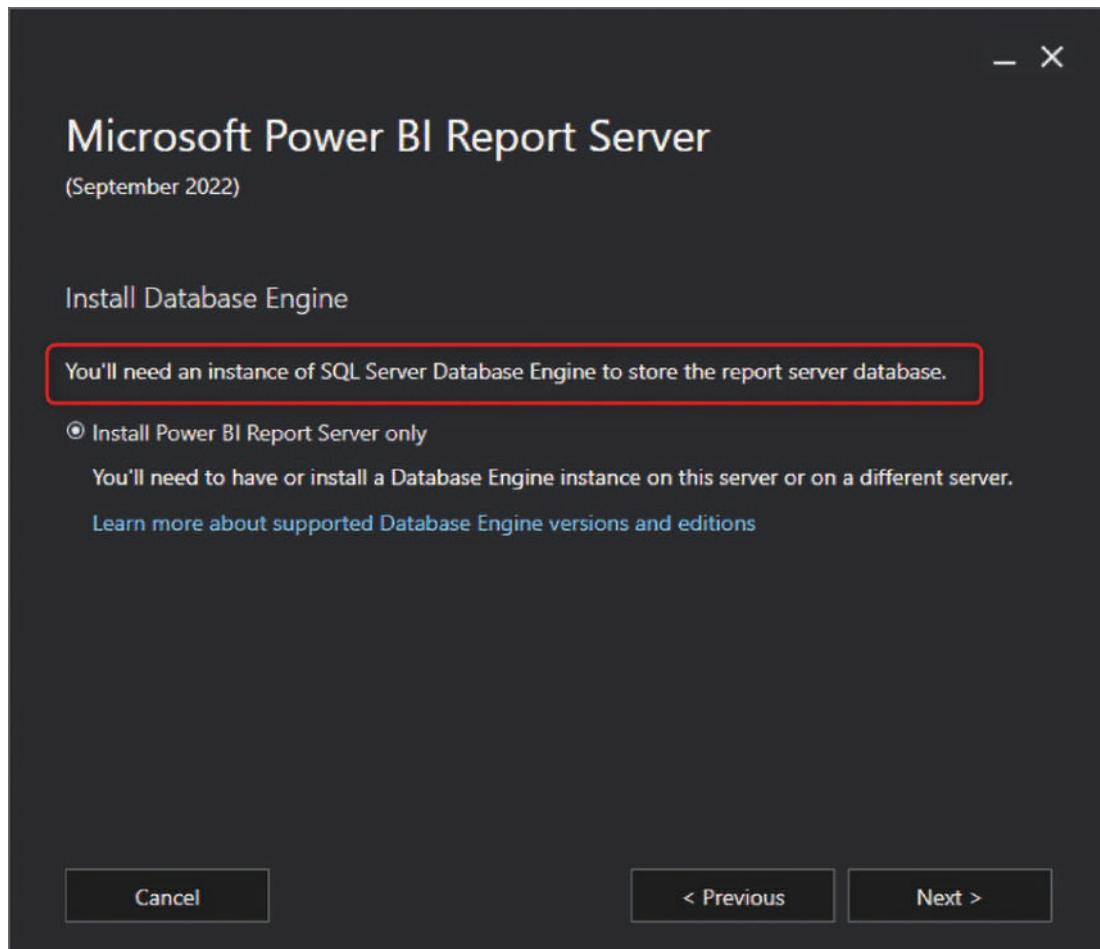
**Figure 21-2.** The Power BI Report Server installation process

As shown in Figure 21-3, you can choose the evaluation edition (valid for six months) or get the licensed version (licensing of the Power BI Report Server is explained later in this chapter).



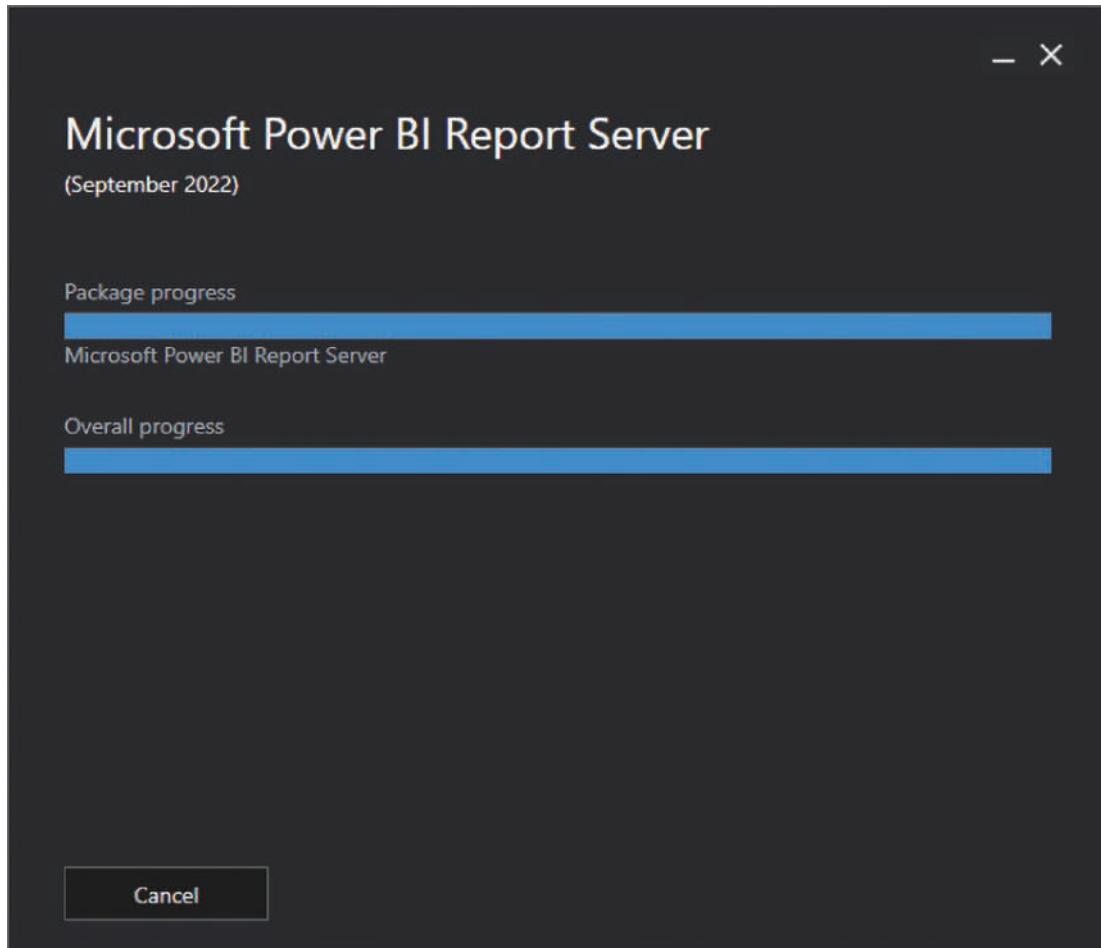
**Figure 21-3.** Selecting the edition of the Power BI Report Server to install

Earlier in this chapter, I mentioned that you don't need to install SQL Server to get the Power BI Report Server. However, the SQL Server database engine is needed for the report server to run, as shown in Figure 21-4. If you don't have SQL Server, don't worry; this setup process will install the database engine automatically.



**Figure 21-4.** The Power BI Report Server requires an instance of the SQL Server database engine

The remaining installation steps are straight-forward (see Figure 21-5).



**Figure 21-5.** The installation process of the Power BI Report Server

After completing the setup, you can open the configuration section by clicking Configure Report Server, as highlighted in Figure 21-6. The instruction asks you to restart and then go to the Report Server. Both options are fine. If you decide to restart your server, then after the restart, choose Start ▶ Programs ▶ Report Server Configuration Manager.

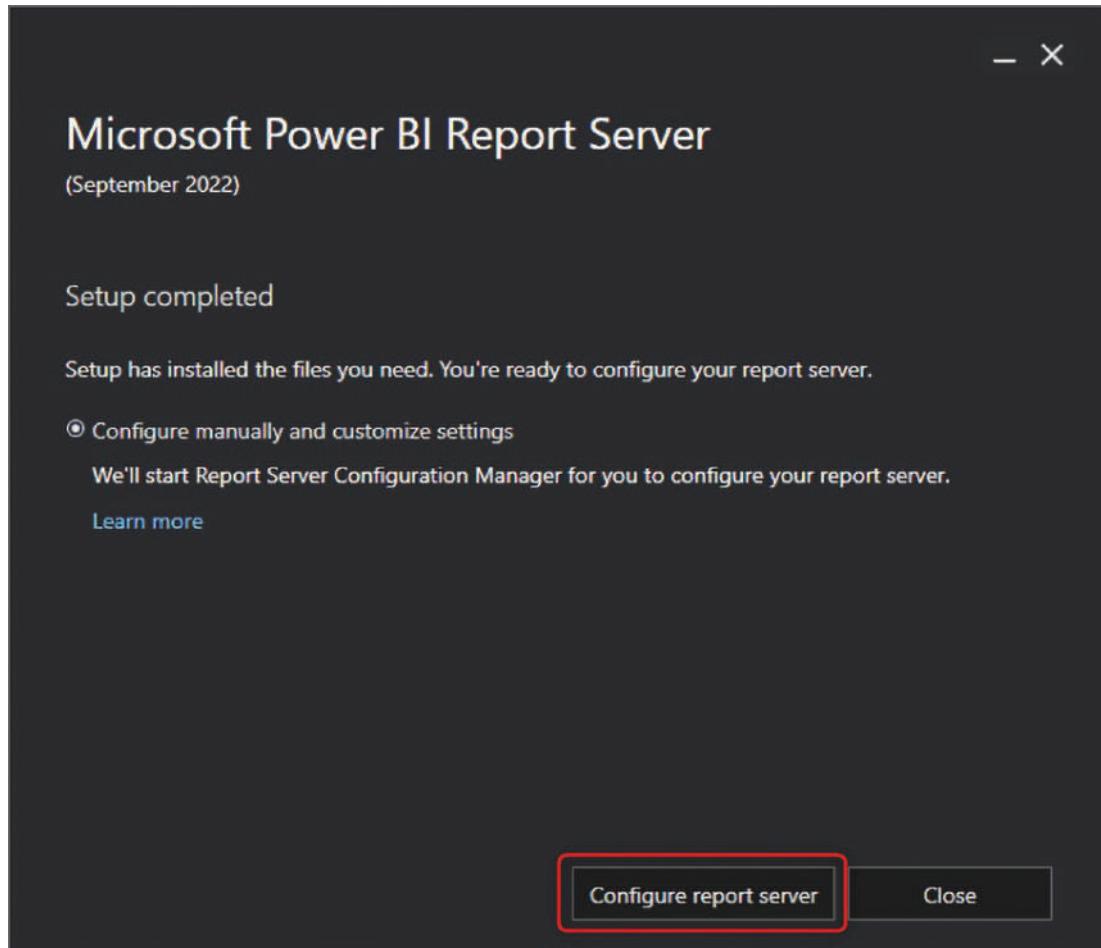
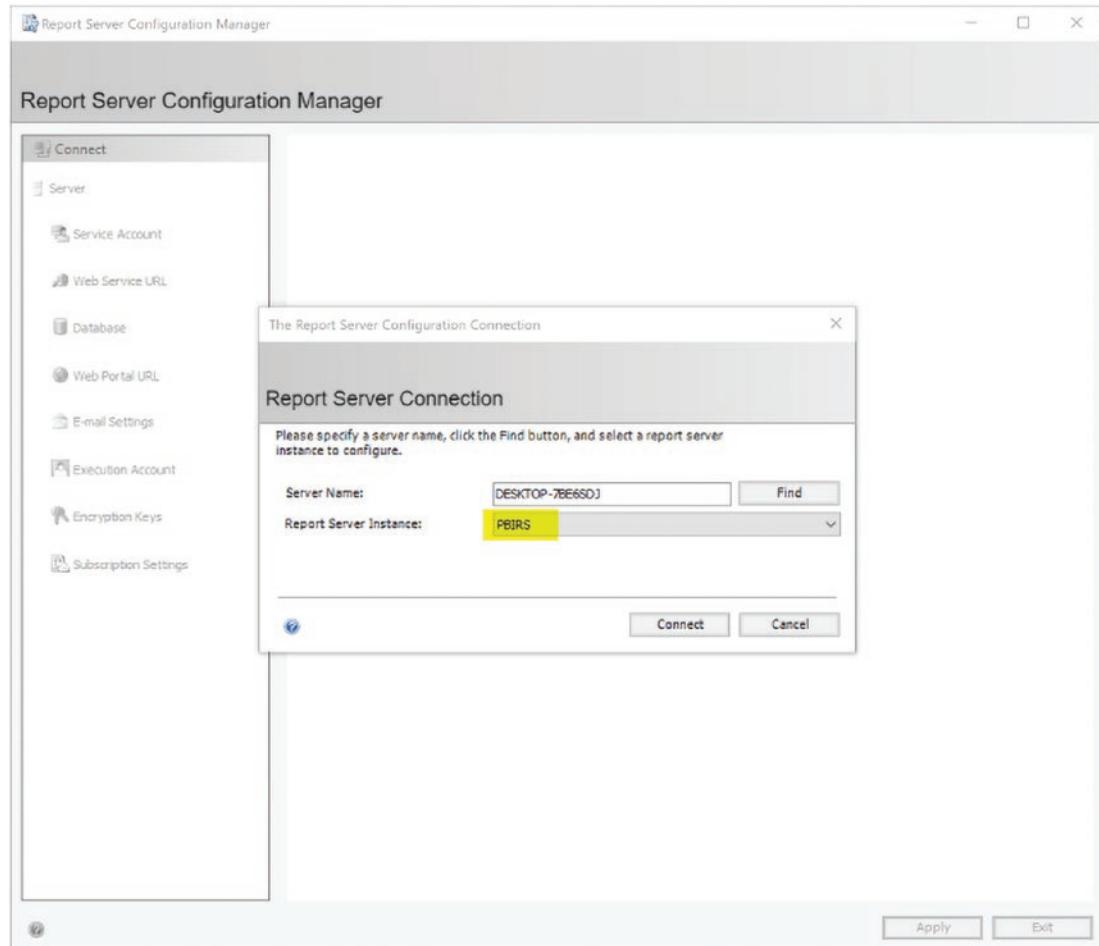


Figure 21-6. Configuring Report Server

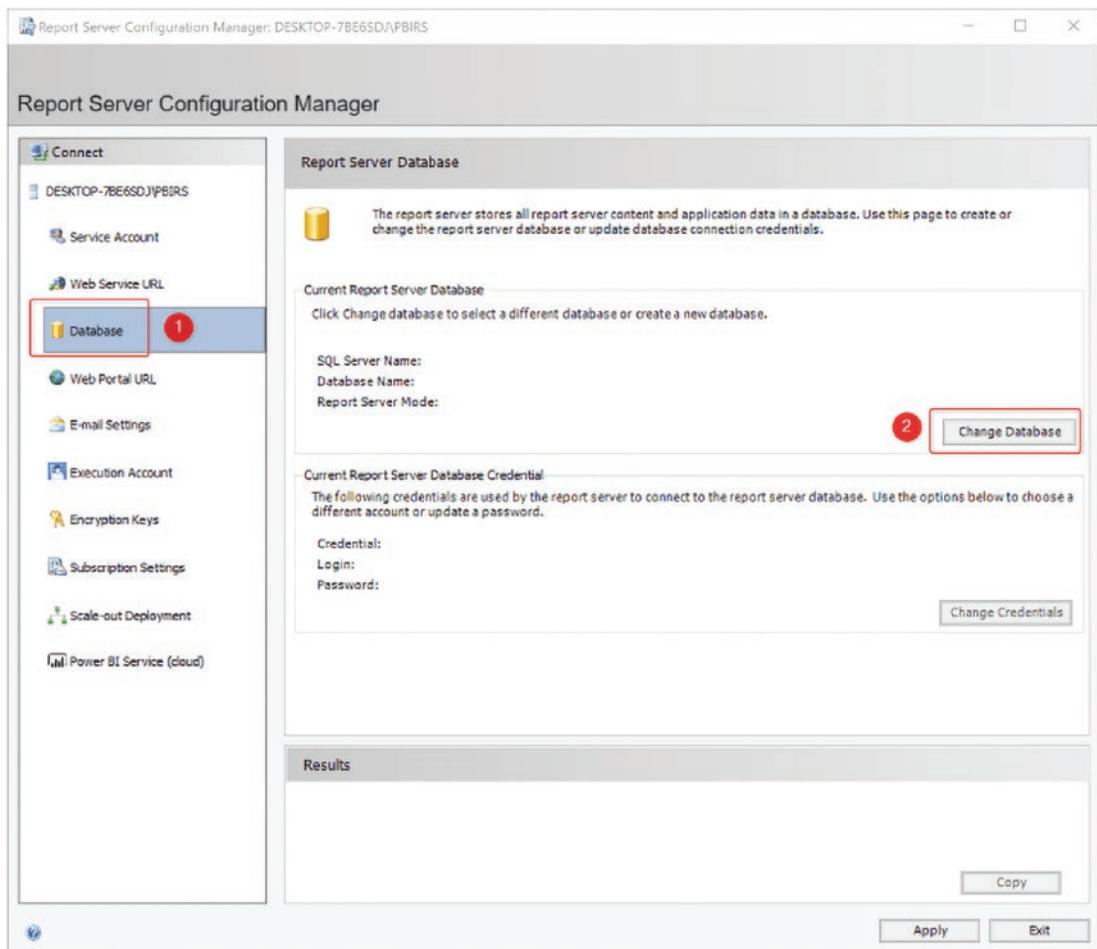
## Configuring the Power BI Report Server

To configure the Report Server, you need to connect to the server you just installed. As Figure 21-7 shows, the instance name of this server is PBIRS.



**Figure 21-7.** Connecting to the Report Server

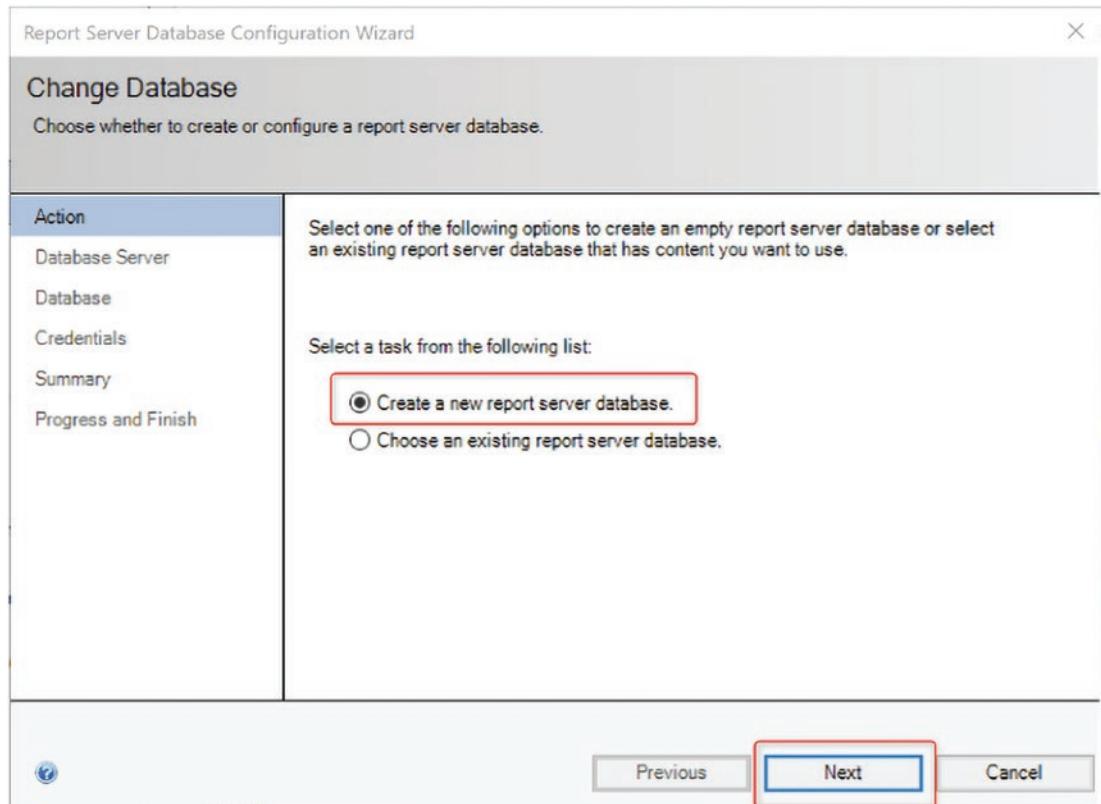
After connecting to the server, it is time to configure it. The first step is to configure databases. To configure the database, click Database on the left side and then choose Change Database, as indicated in Figure 21-8.



**Figure 21-8.** Setting up the database for the Report Server

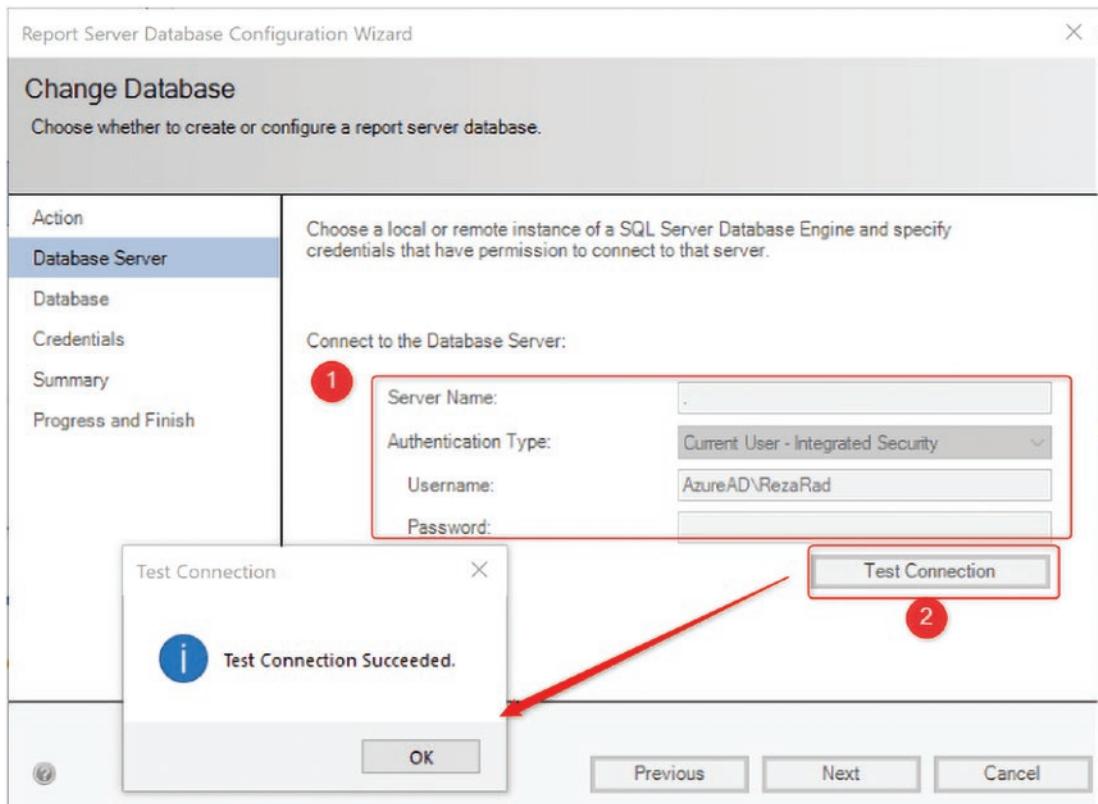
## Database Set Up

You can create databases for the Report Server using the Change Database wizard. As shown in Figure 21-9, select the option to Create a New Report Server Database and then click Next.



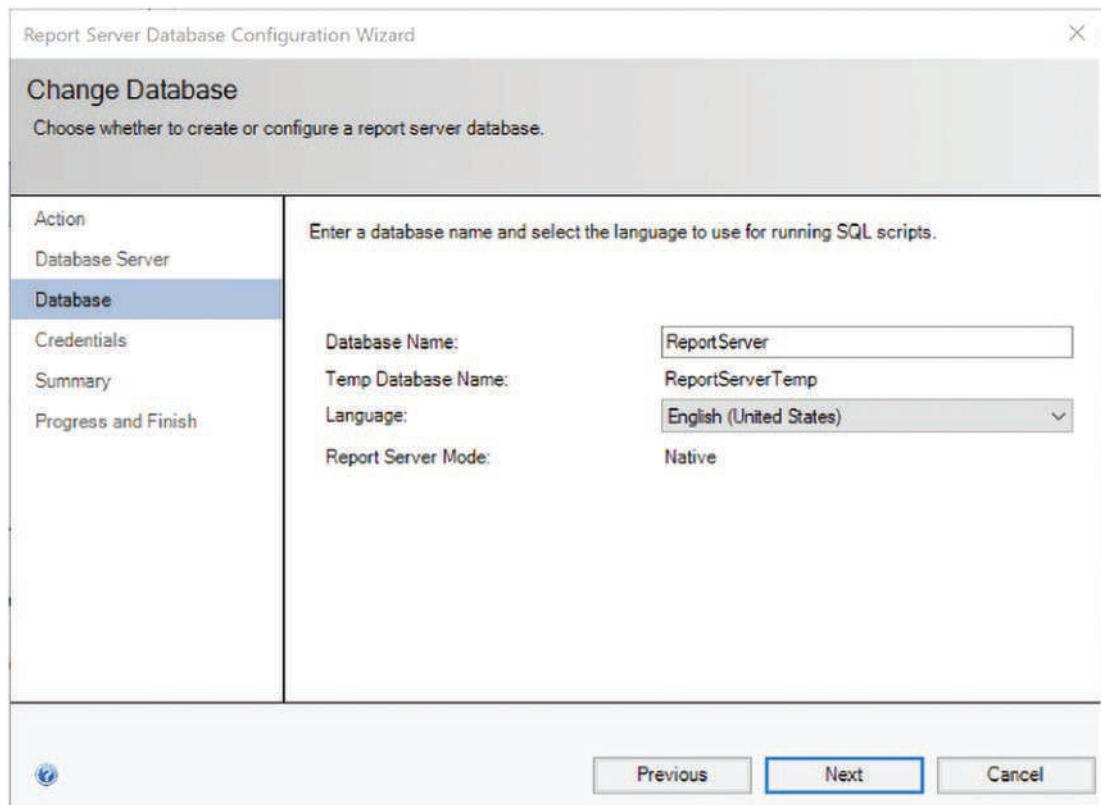
**Figure 21-9.** Creating the database for the Report Server

In the next step, you need to connect to the SQL Server database instance where the Report Server databases will be created. If you have only one local instance of the SQL Server database, you can connect to it with a single dot (a single dot indicates the local database server). If you have multiple instances, you should enter the database server and the username and password. You can also test the connection afterwards to make sure everything is correct. Figure 21-10 shows this alternate process.



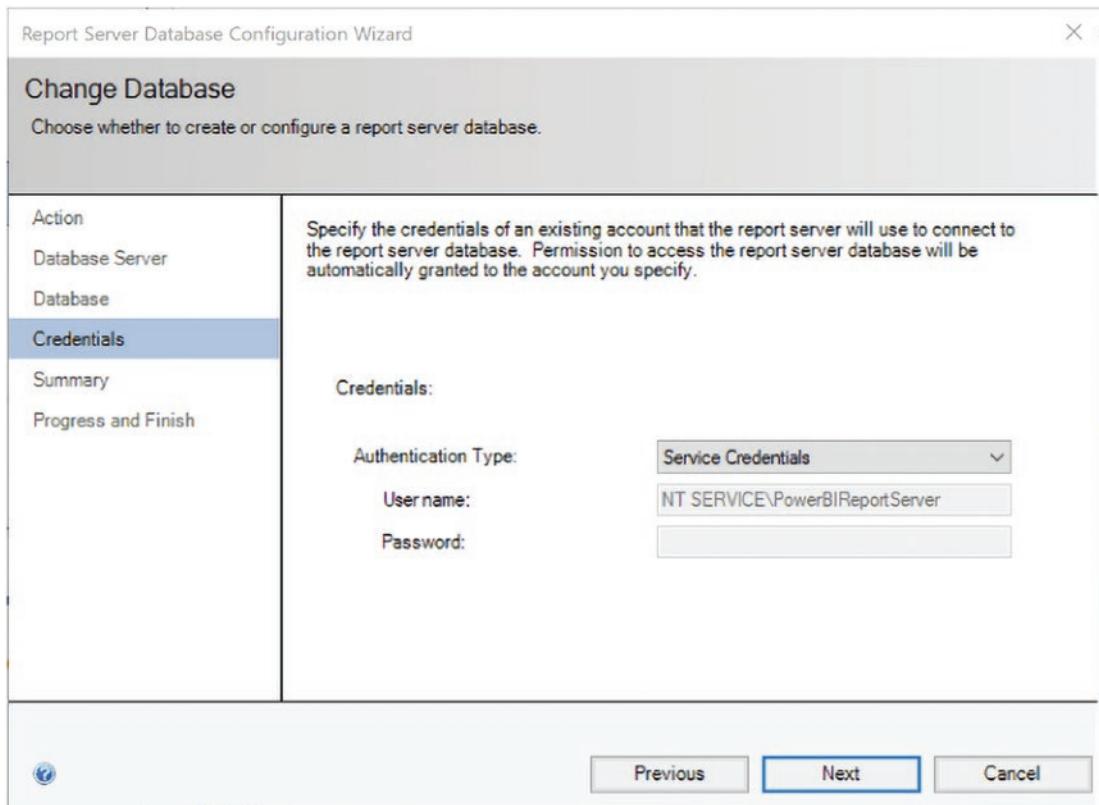
**Figure 21-10.** The database is successfully created and tested

In the next step, choose the database name (the default is ReportServer) and continue, as shown in Figure 21-11.



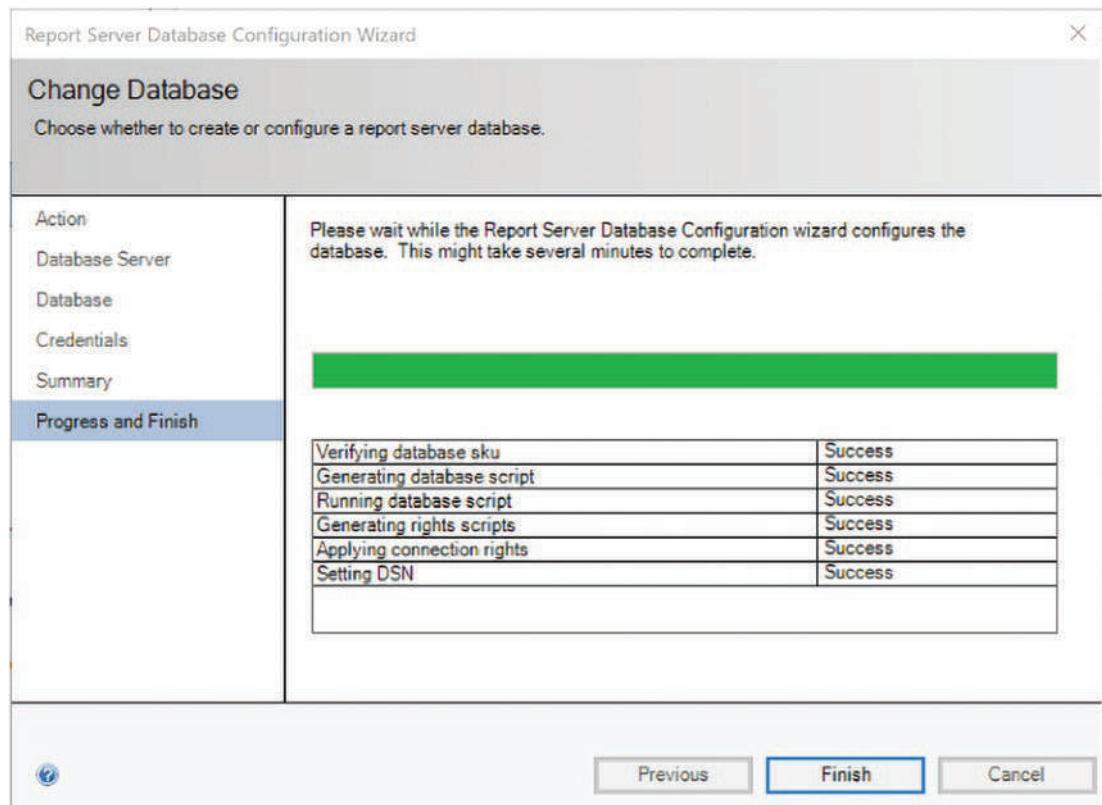
**Figure 21-11.** Creating a temp database

There will always be a second database called the Temp database; you don't need to configure anything about it. Continue with the wizard. In the next step, set the credentials and continue, as indicated in Figure 21-12.



**Figure 21-12.** Credentials to connect to the database server

After confirming things in the Summary step, the setup will continue and finish. Figure 21-13 shows that the database has been successfully set up.



**Figure 21-13.** Database setup has completed successfully

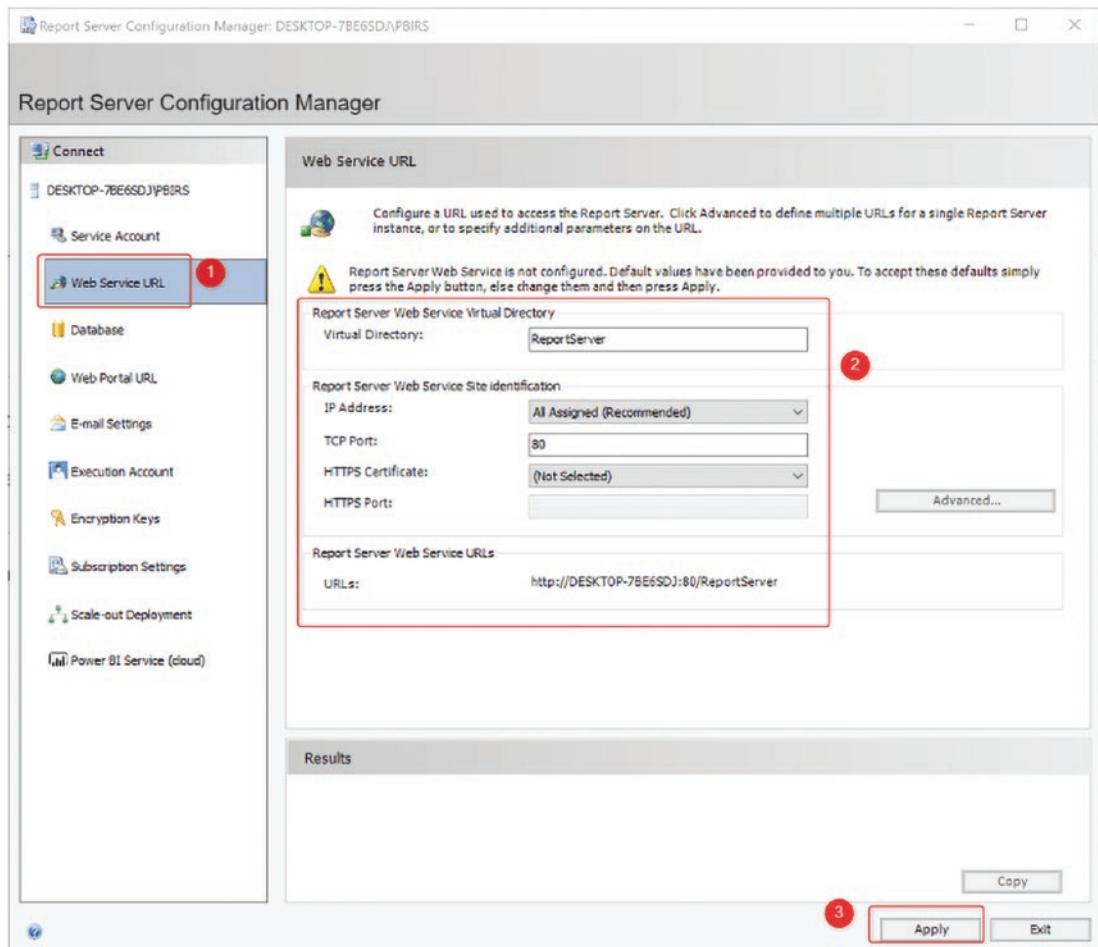
Click Finish. The database setup is done now; the next step is to set up the URLs.

## Web URL Set Up

The Report Server needs web portals, and you need to set up the URLs for it, for the web service, and for the web portal.

## Web Service Set Up

To create the web service, click the Web Service URL on the left side, as shown in Figure 21-14.

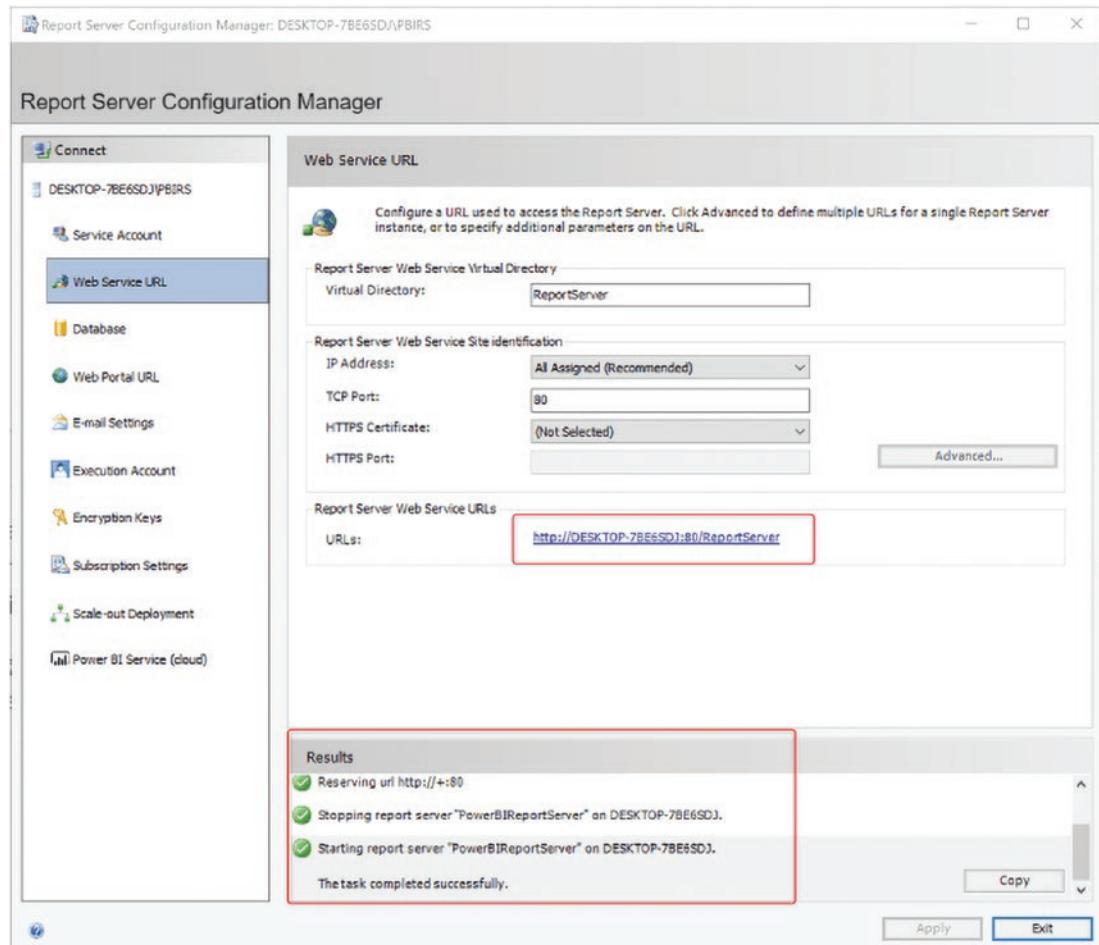


**Figure 21-14.** Getting started with Report Server configuration

You can set up configurations such as the address of the server, the port that this web service will run, and other configurations. If you choose a basic setup, you don't need to change anything here; simply click **Apply**.

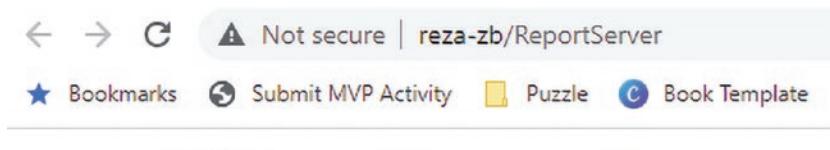
Changing the configuration is only required when you want to set it up on a different port or server with a specific configuration. If you want to do that, it is best to consult with your web admin.

After successfully completing this step, you should see messages and an URL that you can click to open the Report Server's web service, as shown in Figure 21-15.



**Figure 21-15.** Successful Report Service setup

If you click the URL, you should see that the web service's page up and running without any issues or errors. It will look similar to Figure 21-16.

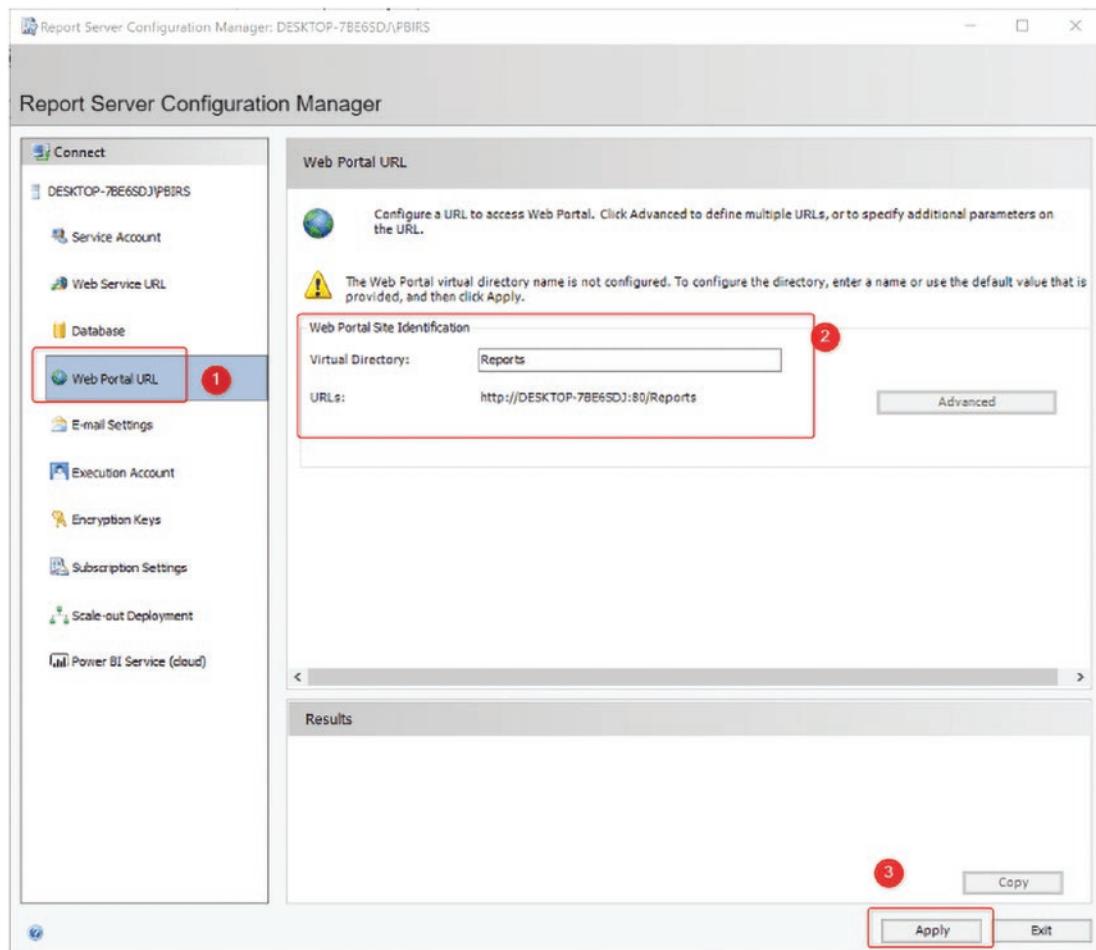


**Figure 21-16.** The Power BI web service page preview

On the Report Server's page, you won't see anything except the version of the Report Server and its name. Later, when you upload Power BI files, you'll be able to see the content.

## Web Portal Set Up

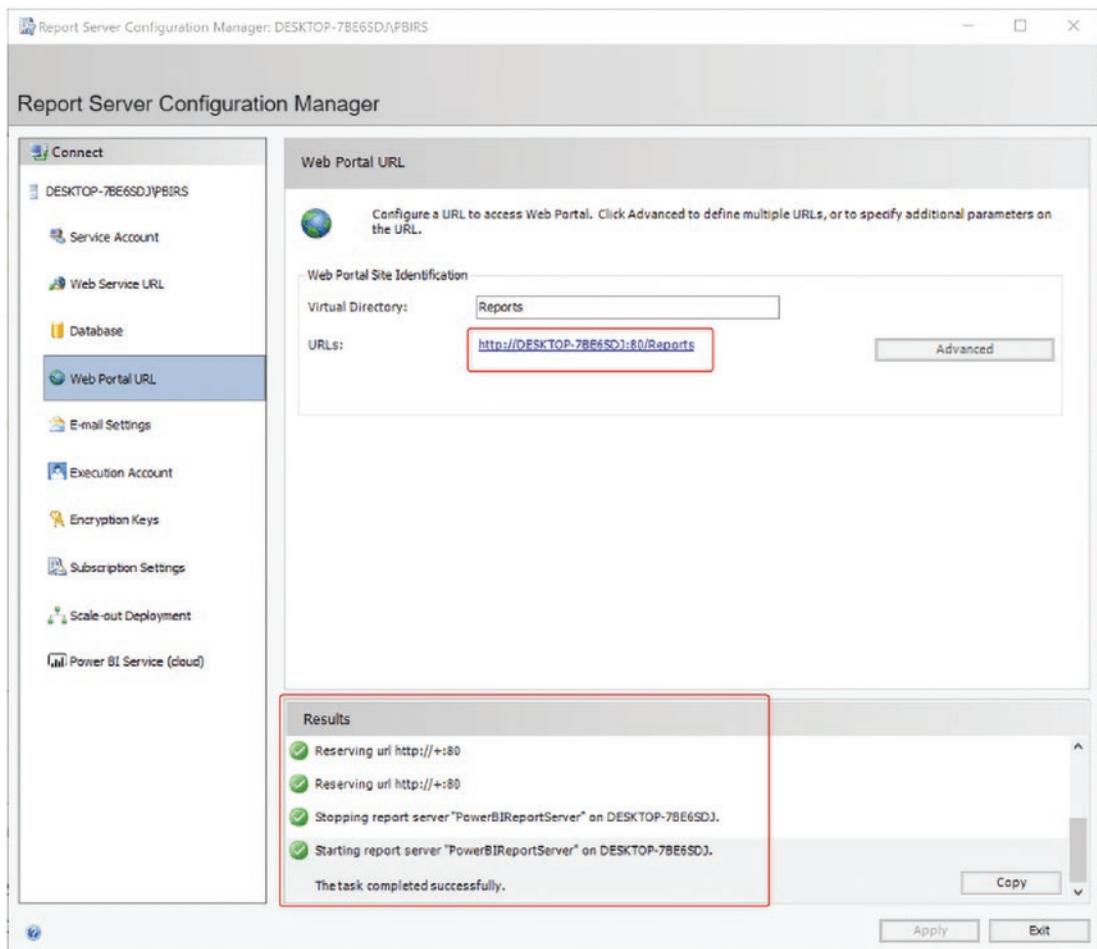
To set up the web portal, click the Web Portal URL on the left side. You can do some configuration if you want for the service, and then click Apply, as outlined in Figure 21-17.



**Figure 21-17.** Initiating the web portal setup process

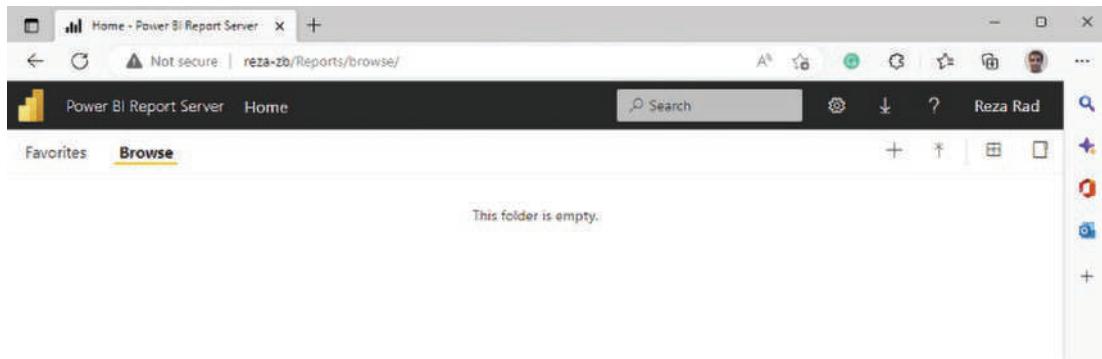
Similar to the web server setup process, if you want to do the advanced setup, consult with your web admin.

If the setup finishes successfully, you see the success message, and you can click the web portal URL, highlighted in Figure 21-18, to open it in a browser window.



**Figure 21-18.** Successful web portal setup

The web portal should show you the environment of the Power BI Report Server's admin view. As you can tell from Figure 21-19, there is no content on the server yet; later in this chapter, you'll learn how to add content.

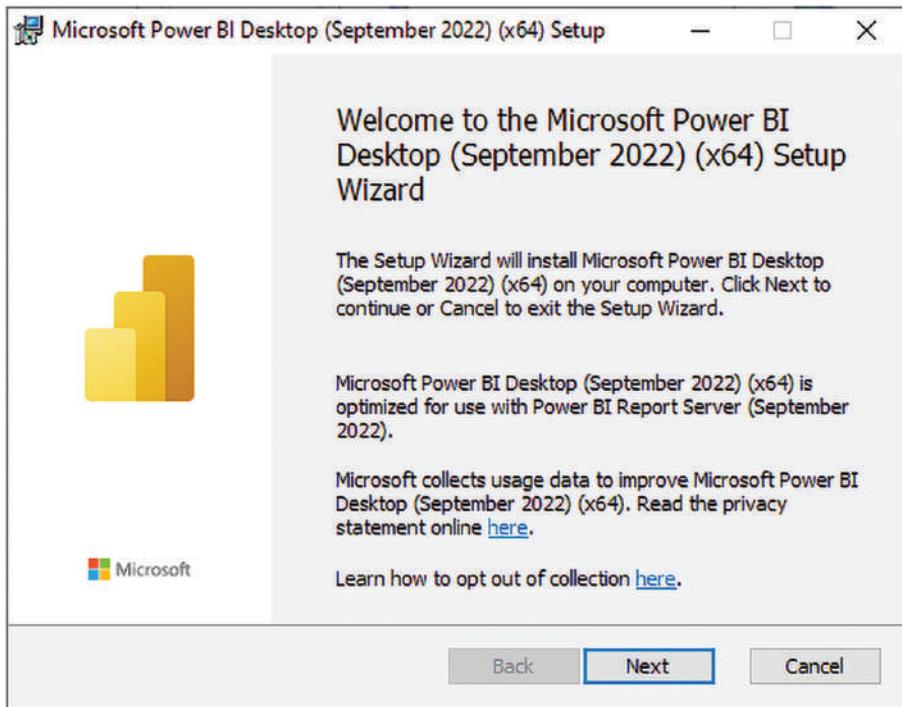


**Figure 21-19.** The Power BI Report Server web portal preview

Installing and configuring the Power BI Report Server is now finished. You can close the Report Server Configuration Manager.

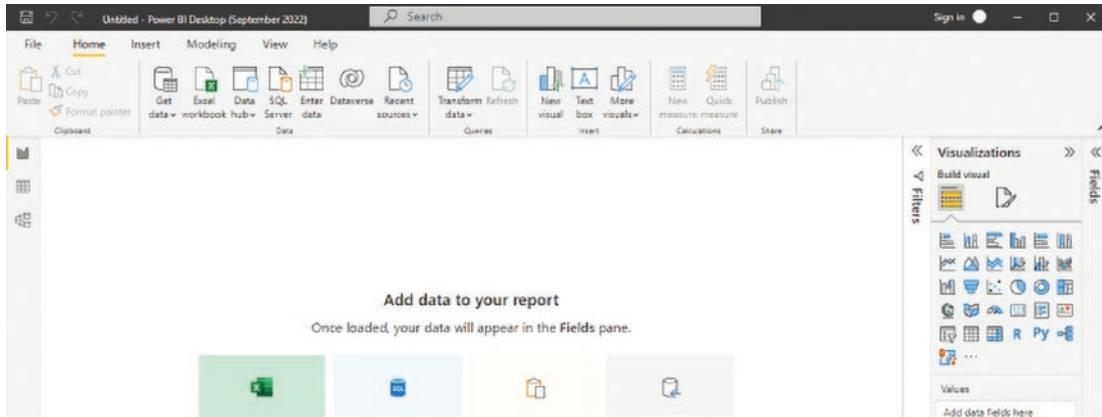
## Installing the Power BI Desktop Report Server

Power BI reports that you can host in the Report Server need to be developed with a specific edition of the Power BI Desktop called the Power BI Desktop Report Server. You get this edition of the Power BI Desktop from the same link you download the Report Server from. You'll begin the installation process as shown in Figure 21-20.



**Figure 21-20.** Installation of the Power BI Desktop Report Server

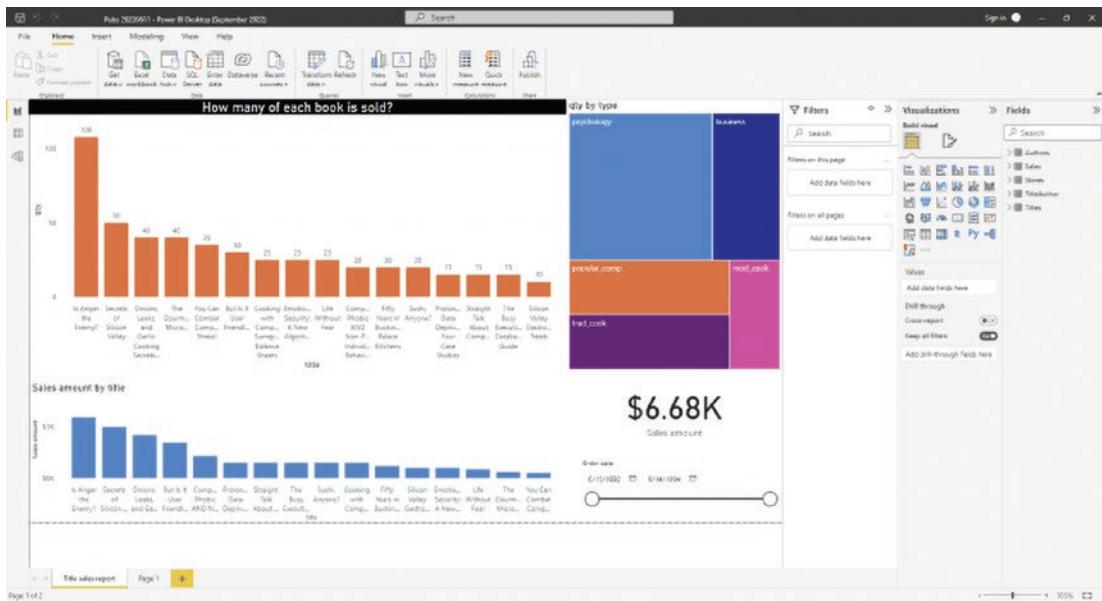
After a successful installation, you can open the Power BI Desktop Report Server. The Power BI Desktop Report Server is similar to the normal Power BI Desktop, with a slight difference, as shown in Figure 21-21.



**Figure 21-21.** The Power BI Desktop Report Server

# Developing Reports with the Power BI Report Server

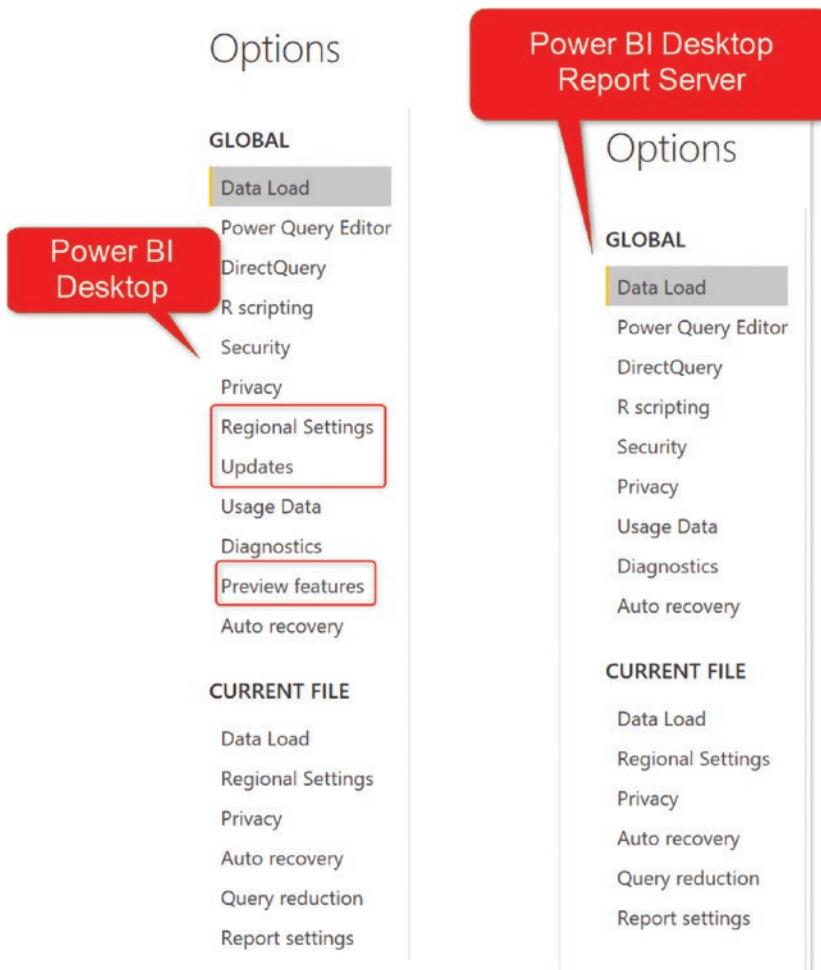
You can start creating a report in the Power BI Desktop Report Server, similar to how you do it in the normal Power BI Desktop. The report development experience in these two editions is very similar. You can even open a report developed with a normal Power BI Desktop in the Power BI Desktop Report Server. A sample report is shown in Figure 21-22.



**Figure 21-22.** The Power BI Desktop Report Server

The Power BI Desktop Report Server is slightly behind the Power BI Desktop. When features are added to the Power BI Desktop, it takes a few months before they are implemented in the Power BI Desktop Report Server (Power BI Desktop updates every month, but the Power BI Desktop Report Server updates every four months). One of the features you will lose in this edition is the Power BI Desktop preview feature. Because the Report Server cannot run the preview features, these items are unavailable here.

You can run the normal Power BI Desktop and the Power BI Desktop Report Server simultaneously on your system. Figure 21-23 illustrates how they differ.



**Figure 21-23.** The Power BI Desktop versus the Power BI Desktop Report Server

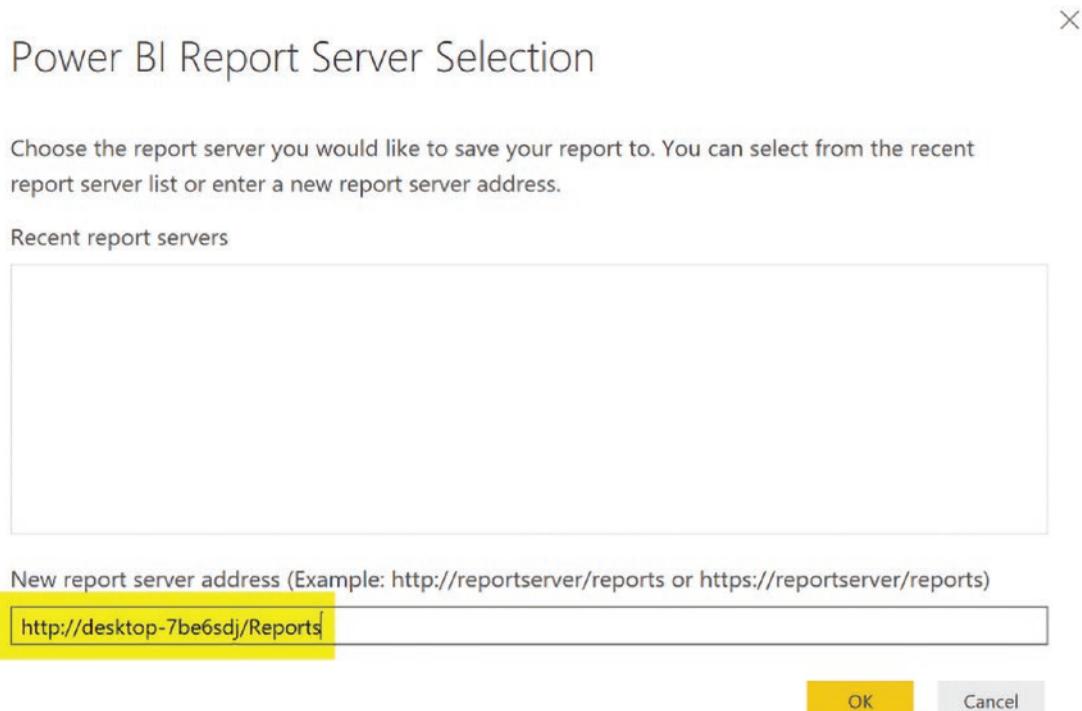
## Publish Report to the Report Server

There are two ways to publish the Power BI report to the report server. One way is from the Power BI Desktop Report server edition. First, you need to set up the URL to your report server. Go to the File menu and select Power BI Report Server from Open Report, as shown in the Figure 21-24.



**Figure 21-24.** Opening a report from the Power BI Report Server

In this window, you can connect to a report server. Enter the web portal URL from the step of configuring the report server (see Figure 21-25).



**Figure 21-25.** Publishing reports to a report server

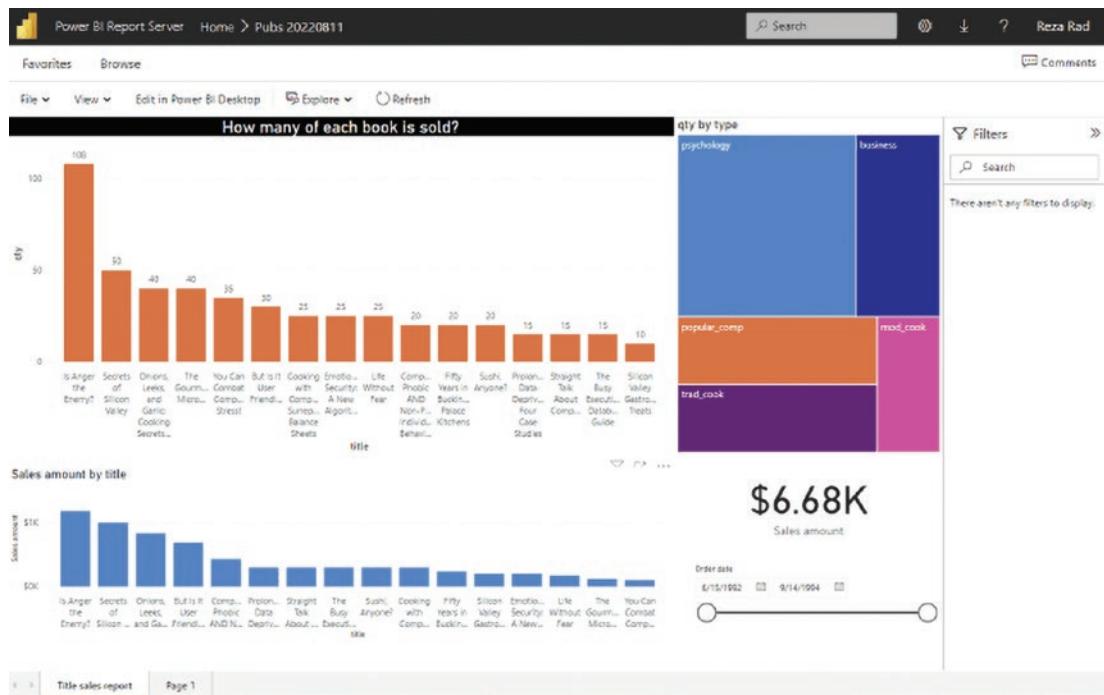
Like the method mentioned in Figure 21-25, you can go to Save As, select Power BI Report Server as your destination, and save your report there.

After successful deployment, you will see a message with a link to the report, previewed in Figure 21-26.



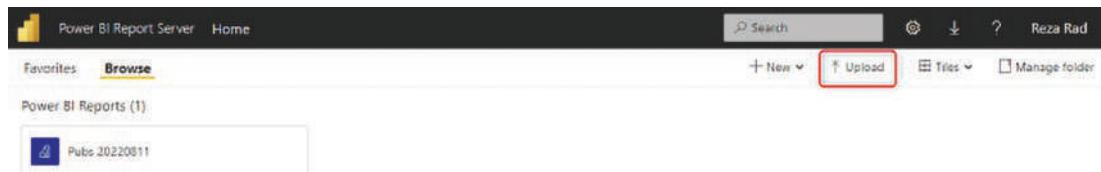
**Figure 21-26.** The report was successfully published to the Report Server

The report hosted on the Power BI Report Server is a fully interactive report like a Power BI report hosted in the service. Figure 21-27 shows a sample report.



**Figure 21-27.** A Power BI report hosted in the Power BI Report Server

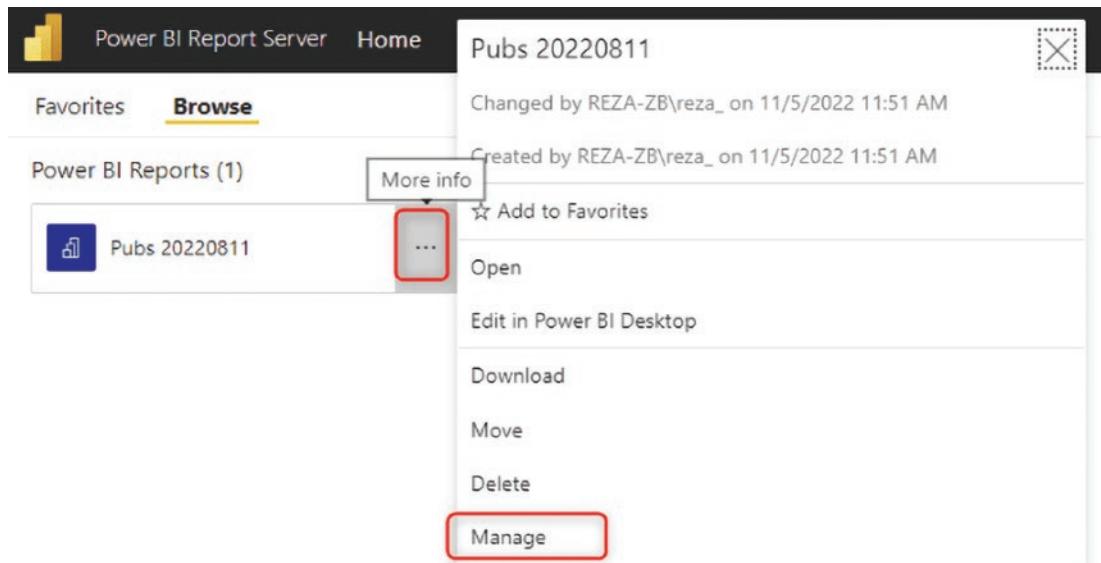
The second way to publish a Power BI report to the Report Server is to use the Upload item in the Power BI Report Server; you can use the Upload option from the web portal, as shown in Figure 21-28.



**Figure 21-28.** Upload a report using the Report Server's web portal

## Managing Datasets on the Report Server

A Power BI report published to the Report Server can be configured to refresh. To do this, open the Report Server web portal, and as shown in Figure 21-29, click More Info for the Power BI report.



**Figure 21-29.** Configuring the Power BI report on the Power BI Report Server

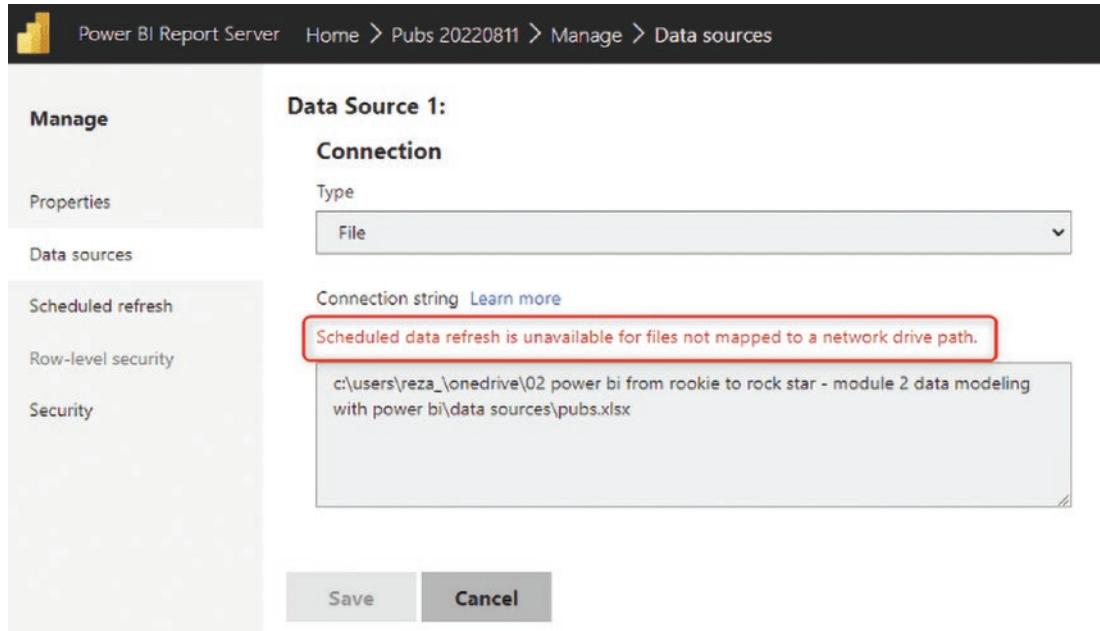
In the Manage tab of a report, you can configure the data source configuration, the connection to the data source, and schedule a refresh if required (see Figure 21-30).

The screenshot shows the 'Properties' dialog for a report named 'Pubs 20220811'. The top navigation bar includes 'Edit in Power BI Desktop', 'Download', 'Replace', 'Move', and 'Delete' buttons. On the left, a sidebar lists 'Manage', 'Properties', 'Data sources', 'Scheduled refresh', 'Row-level security', and 'Security'. The main area displays 'Modified by' (REZA-ZB\reza\_ on 11/5/2022 11:51 AM), 'Created by' (REZA-ZB\reza\_ on 11/5/2022 11:51 AM), 'Size' (143 KB), and a 'Properties' section. In the 'Properties' section, the 'Name\*' field is filled with 'Pubs 20220811'. Below it is a 'Description' field with a large empty text area. A 'Hide this item' toggle switch is shown as off. At the bottom are 'Apply' and 'Cancel' buttons.

**Figure 21-30.** Managing data source configuration

## Schedule Refresh Requirement

If your report is sourced from a file, as shown in the example in Figure 21-31, you may have some requirements to fulfill before you can schedule it for a refresh. You would need to source the file from a network path.



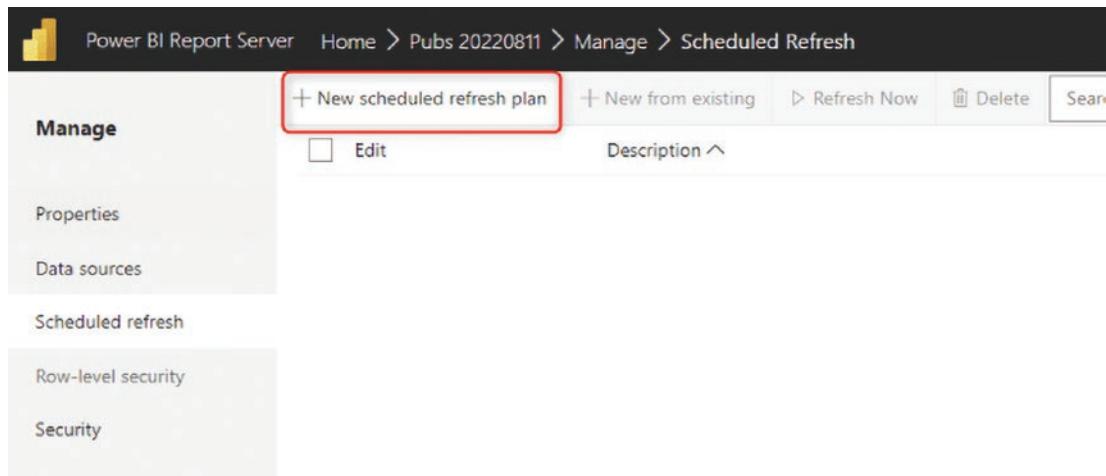
**Figure 21-31.** Use a network drive path for the file addresses

If you use a network-shared path to access the source file, you can set up the connection to the file, as shown in Figure 21-32.

The screenshot shows the 'Data sources' page in the Power BI Report Server. The left sidebar has 'Manage' selected. The main area is titled 'Data Source 1:' and shows the 'Connection' tab. The 'Type' dropdown is set to 'File'. The 'Connection string' field contains '\\reza-zb\ netpath\pubs.xlsx'. Under the 'Credentials' tab, 'Windows Authentication' is selected for 'Authentication Type'. The 'User name\*' field is filled with 'gateway' and the 'Password\*' field contains '\*\*\*\*\*'. A 'Test connection' button is shown with a success message: 'Connected successfully'. At the bottom are 'Save' and 'Cancel' buttons.

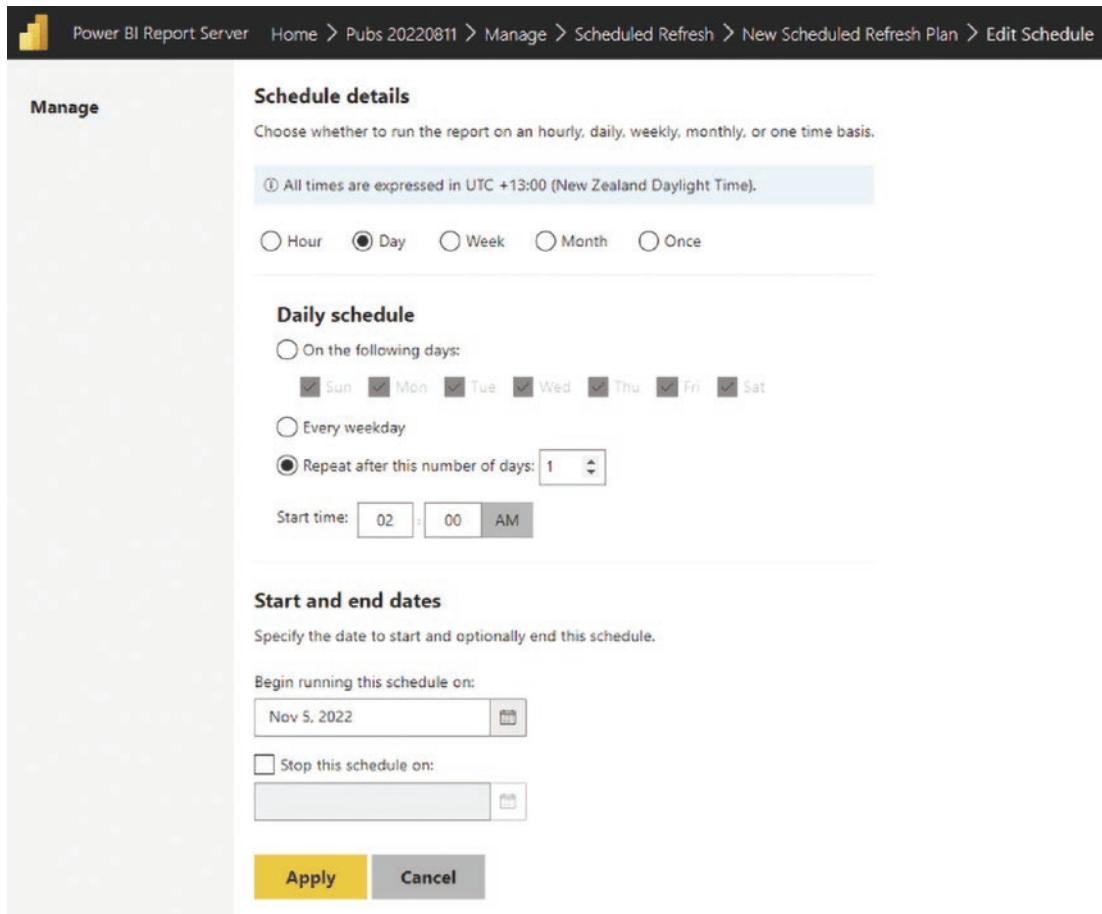
**Figure 21-32.** Setting up the data source credentials in the Power BI Report Server

Make sure to click Save after this step. Otherwise, you won't be able to schedule the refresh process. Then you can click the Scheduled Refresh and create a scheduled refresh plan. See Figure 21-33.



**Figure 21-33.** Create a scheduled refresh plan for the Power BI Report

The Scheduled Refresh feature of the Report Server has many more options than the Power BI Service; you can schedule hourly, daily, weekly, monthly, or any custom period. You can choose the start and end date and many other configurations. Figure 21-34 shows the available options.



**Figure 21-34.** Setting the schedule details for the refresh plan

For a scheduled refresh plan to be successful, the SQL Server Agent service must be up and running. In the Scheduled Refresh section, you will also see a list of configurations and their status.

## Pros and Cons of the Report Server

As you saw, the Report Server environment is similar in some ways (data source settings, scheduled refresh, security, and sharing) to the Power BI Service. The Power BI Report Server comes with advantages and disadvantages.

### No Gateway Is Needed

You read that right; you do not need a gateway with the Power BI Report Server. A gateway is only for all connections from the Power BI Service. Gateways are responsible for connecting the dataset from the Power BI Service to the data source on-premises. With the Power BI Report Server, everything is hosted on-premises. You do not need to install or configure a gateway.

## All Types of Connections Are Supported

With the very early releases of the Power BI Report Server, you could only create a live connection to SQL Server Analysis Services from Power BI reports. Now, you can use any connection (except the composite model). The example you saw earlier in this chapter used Import Data and scheduled the report to refresh. You can also use the DirectQuery connection or Live Connection to the Power BI Report Server.

## The Power BI Report Server Is a Fully On-Premises Solution

I wrote the first draft of this chapter while flying on a 17-hour flight, my first leg from New Zealand to the UK, with no Internet connection. All of the examples and screenshots were taken without an Internet connection. The Power BI Report Server is a fully on-premises solution. You will not publish your reports to the Power BI website, and you will not need a cloud-based technology.

The Power BI Report Server is an on-premises technology choice for companies not yet ready to move to cloud-based technologies.

## The Power BI Service Features Are Not Available

The Power BI Report Server has many great features. However, it also has some drawbacks. One of the main drawbacks of the Power BI Report Server is its isolation from the Power BI Service. You won't get great features of the Power BI website on the Report Server. The website has features such as usage metrics of the report, Power BI apps, Q&A and quick insights, and many others that are not available in the Report Server.

Here is a list of some of the main features that are not included in the Power BI Report Server:

- Analyze in Excel
- Composite model
- Dashboards
- Workspaces
- Apps
- Template apps
- Shared datasets
- Q&A
- Cross-report drill through
- Preview features
- Metrics
- Dataflows
- Datamarts

## Licensing the Report Server

The Power BI Report Server comes in two types of licensing—Power BI Premium and SQL Server Enterprise License with Software Assurance.

## Summary

The Power BI Report Server is an on-premises reporting technology. With the Power BI Report Server, you can use interactive Power BI reports on on-premises servers. This type of technology is based on SQL Server Reporting Services technology. You will need to set up the Power BI Report Server alongside a specific edition of Power BI Desktop.

There are some pros and cons of the Report Server. With the Power BI Report Server, you can host reports fully on-premises without needing a Power BI website. You do not need a gateway, and all types of connections (Scheduled Refresh, DirectQuery, and Live Connection) are supported (except the composite model). However, the Power BI Report Server doesn't have all the features and functionalities available in the Power BI Service.

The Power BI Report Server needs specific licensing from Power BI Premium or SQL Server Enterprise license with software assurance.

## CHAPTER 22



# Power BI Gateways

Power BI is a data analysis tool that connects to many data sources. Suppose the data source for Power BI is located on-premises. In that case, the connection from the cloud-based Power BI Service to the on-premises data source should be created with an application called a *Gateway*. This chapter explains what a Gateway is, the types of the Gateways, their differences, installing a Gateway, and scheduling a dataset with that Gateway.

## What Is a Gateway?

Power BI Gateways connect Power BI cloud-based data analysis technology and the data source on-premises. A Gateway is an application that can be installed on any server in the local domain. The Gateway is responsible for creating the connection and passing data through.

## Do You Always Need a Gateway?

You don't need a Gateway in all scenarios, only when the data source is on-premises. For online or cloud-based data sources, no Gateway is required. For example, if you are getting data from CRM online, you don't need a Gateway. However, if you are getting data from a SQL Server database located on your local domain server, you need a Gateway. For Azure SQL DB, you don't need a Gateway. However, a SQL Server database on an Azure Virtual Machine is considered on-premises and needs a Gateway.

## Types of Gateways

Power BI Gateways come in two modes—personal mode and standard mode. The difference between these two is not the paid or licensing plan. Both Gateways are free to use. The difference is the way that you want to use the Gateway. Personal mode is mainly used for one-person use, not for a team. On-premises standard Gateway, on the other hand, is a good choice when you want to work in a collaborative environment. Let's look at their differences in detail. In Table 22-1, the on-premises mode is the standard mode of a Power BI on-premises Gateway.

**Table 22-1.** Personal versus Standard On-Premises Modes

	<b>Personal Mode</b>	<b>On-premises Mode</b>
<b>Target Persona</b>	Business Analyst who wants to set up and use gateway to run his/her reports	BI Admins to set up the gateway for their companies Multiple BI developers to use the gateway for their reports
<b>Usage</b>	Analyst	BI Admin Developer
<b>Connection Type</b>	Import Data or Scheduled Refresh	Import Data or Schedule Refresh Live Connection DirectQuery
<b>Management</b>	Per user data source management	Central data source management
<b>Monitoring/Control</b>	No monitoring/Control	Central Monitoring and Control
<b>Services Supported</b>	Power BI only	Power BI PowerApps Microsoft Flow Azure Logic Apps

## Personal Mode

When you install a Gateway in personal mode, you can use it yourself only. You can connect it to local data sources such as SQL Server, Excel, and other data sources. However, the Gateway installed in personal mode only supports one type of connection—Import Data or Scheduled Refresh. This Gateway is only used for Power BI; you cannot use it for other applications.

Because this Gateway is personal, you cannot use it in a team development scenario. Multiple developers cannot leverage this Gateway. You can create reports and connect them to this Gateway and share it with multiple users. However, only one developer can use the Gateway. That is why it is called personal mode.

Installing the personal mode and configuring it is easier than the on-premises Gateway. When you install the Gateway in personal mode, you don't have the configuration option to set data sources. There is no place to configure it after installation. This Gateway uses the same credentials as the person who installed it. This Gateway mode is meant to be used for business analysts with the least hassle to get their reports published and refreshed.

This type of Gateway is usually for one business analyst who wants to publish Power BI reports and schedule them to refresh and share them with users. There are not many configuration options, it's easy to set up, and its single developer features make it a good option for such scenarios.

## On-Premises Standard Mode

Power BI's on-premises Gateway recommends the standard mode. This mode of installation supports a multi-developer environment. Multiple developers can use the Gateway. This Gateway type is built for team development; you can have a Gateway administrator. There is a central configuration section for Gateways to add data sources and control them.

On-premises Gateways support not only Power BI but also PowerApps, Azure Logic Apps, and Microsoft Flow, which are other Microsoft cloud-based technologies.

On-premises Gateways also support all types of connections from Power BI. The Import Data or Scheduled Refresh is supported, as well as DirectQuery and Live Connection.

This type of Gateway is for enterprise usage of Power BI or where Power BI needs to be used alongside other applications such as PowerApps. Multiple developers can work with the same Gateway if the administrator authorizes them to do so. More centralized control and monitoring exist for this type of Gateway.

## Gateway Execution Flow

Figure 22-1 shows the architecture of this Gateway.

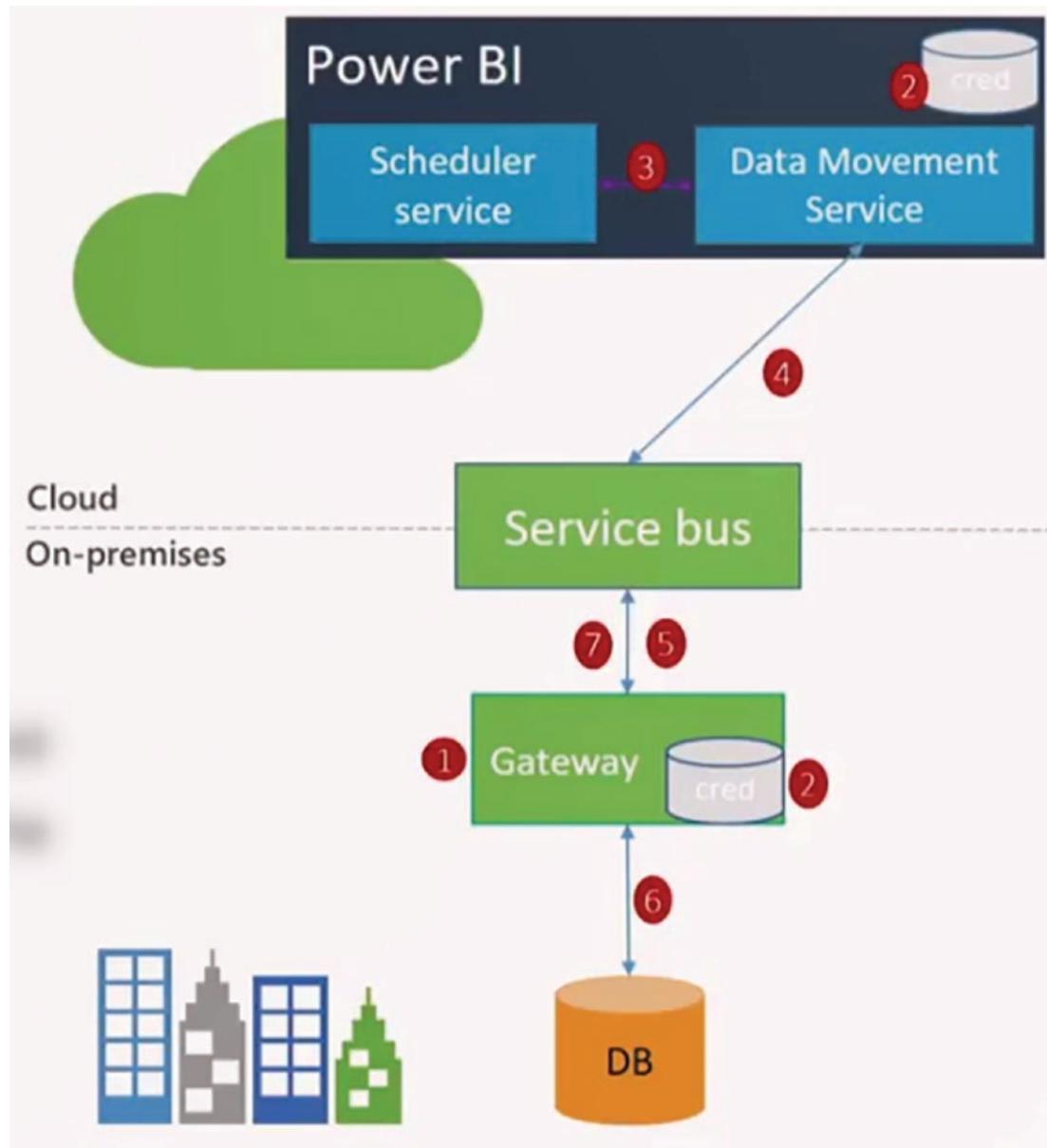


Figure 22-1. The flow of execution for the Power BI Gateway

Gateways are installed on a machine in the on-premises domain. During this installation, credentials are stored in local and Power BI Services.

Credentials entered for the data source in Power BI are encrypted and then stored in the cloud. Only the Gateway can decrypt the credentials.

The Power BI Service kicks off a dataset refresh; this happens through a scheduler service in Power BI.

Data movement service analyzes the query and pushes it to the appropriate service bus instance. There is a queue of requests on the service bus. The Gateway pulls the bus for pending requests. The Gateway gets the query and executes it on the data source. After getting the result, the Gateway pushes that back to Power BI.

When the Gateway pulls the bus to check if there are any pending requests, the bus cannot trigger the Gateway. The reason for this architecture is security. If the bus can trigger the Gateway, the inbound security ports need to be open, which is not a good practice for security. So, you can say that the Gateway connection is very secure because it only uses outbound ports.

## Important Points to Consider Before Installing a Gateway

A Gateway can be installed on any machine in the on-premises domain. However, it is not recommended to be installed on the domain controller. Here are the requirements for a Gateway installation.

Minimum requirements:

- .NET Framework 4.7.2 (Gateway release December 2020 and earlier)
- .NET Framework 4.8 (Gateway release February 2021 and later)
- A 64-bit version of Windows 8 or a 64-bit version of Windows Server 2012 R2 with current TLS 1.2 and cipher suites
- 4GB disk space for performance monitoring logs (in the default configuration)

Recommendations:

- An 8-core CPU
- 8GB of memory
- A 64-bit version of Windows Server 2012 R2 or later
- Solid-state drive (SSD) storage for spooling

## How Many Gateways Are Required?

One Gateway should be enough for many situations. However, sometimes you get more benefits from having more Gateways. As an example, if you have a Gateway that is used for scheduled data refresh, and the same Gateway is used for a Live Connection, then you get slow performance for the Live Connection if there is a scheduled data refresh in process at that time. So, in this scenario, you might consider having one Gateway for your Live Connection and another for a scheduled refresh.

The Gateway can be installed only on 64-bit Windows operating systems.

I recommend choosing the version of the Gateway you need on that machine carefully. If it is a server, I highly recommend installing an on-premises standard Gateway rather than a personal one. The Gateway machine should always be up and running to handle data refresh queries.

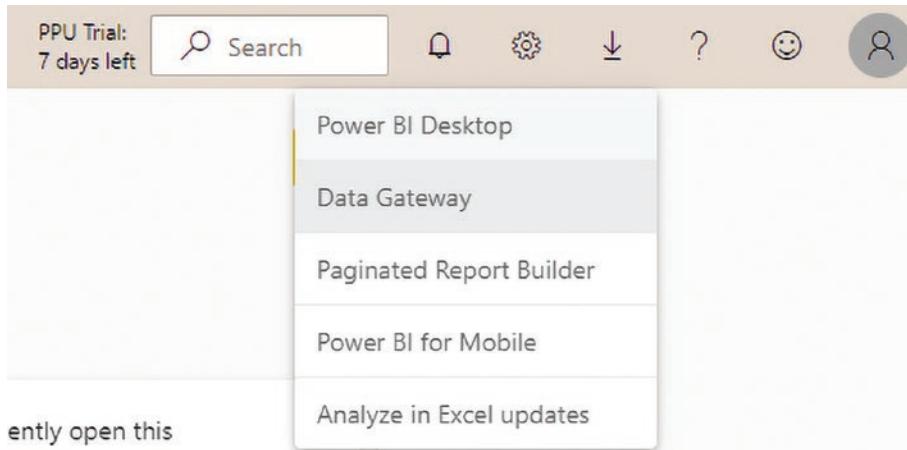
Do not install the Gateway on a machine that is connected through a wireless network. Gateways will perform more slowly on a wireless network. Ports that need to be open for the Gateway are all outbound ports: TCP 443 (default), 5671, 5672, and 9350-9354. The Gateway does not require inbound ports.

## Installing Gateways

You can download the Power BI Gateway from this link:

[powerbi.microsoft.com/en-us/gateway/](http://powerbi.microsoft.com/en-us/gateway/)

Or you can find the link when you log in to the Power BI Service. Under Download, choose Data Gateway, as shown in Figure 22-2.



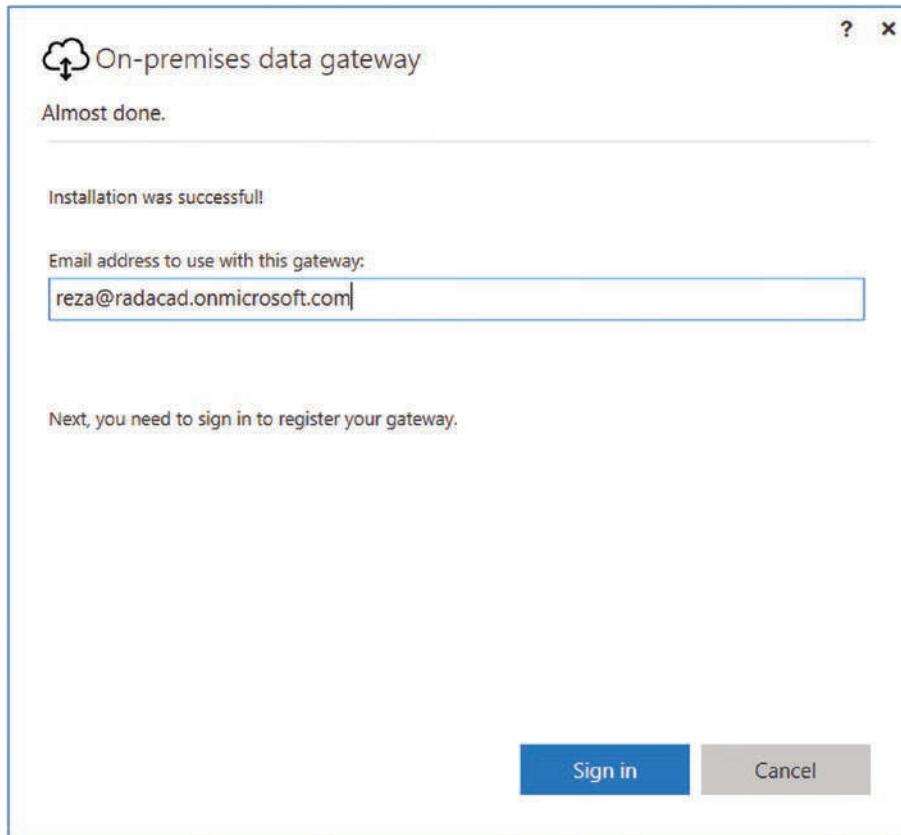
**Figure 22-2.** Downloading the Power BI Gateway

After downloading and running the installation file, you will see the options you can configure.

For this example, you are going to install the on-premises recommended Gateway option because it supports Live Connection and DirectQuery.

After choosing the Gateway type, the installer downloads the files required for installation, and then you can continue the installation. You need to choose a folder to install the Gateway. In this example, you'll use the default. The installation process is simple. After installation, you need to register your Gateway.

To register your Gateway, you need to sign in using your Power BI email account, as shown in Figure 22-3.



**Figure 22-3.** Signing in with a Power BI account

You can then register a new Gateway or migrate or restore an existing Gateway. Select Register a New Gateway and continue. You need to enter two important pieces of information:

- **Gateway name:** A name that can remind you where this Gateway is installed. For example, Reza-Vaio-Gateway.
- **Recovery key:** This is a very important key required for recovering the Gateway later. If you want to uninstall it and install it again, or if you want to move the Gateway from one machine to another without the hassle of changing all the connections, then keep the Gateway name and recovery key in a safe place.

You can also add the Gateway to an existing Gateway cluster. This option is for having high availability through Gateways. For this example, leave that unchecked, as shown in Figure 22-4.

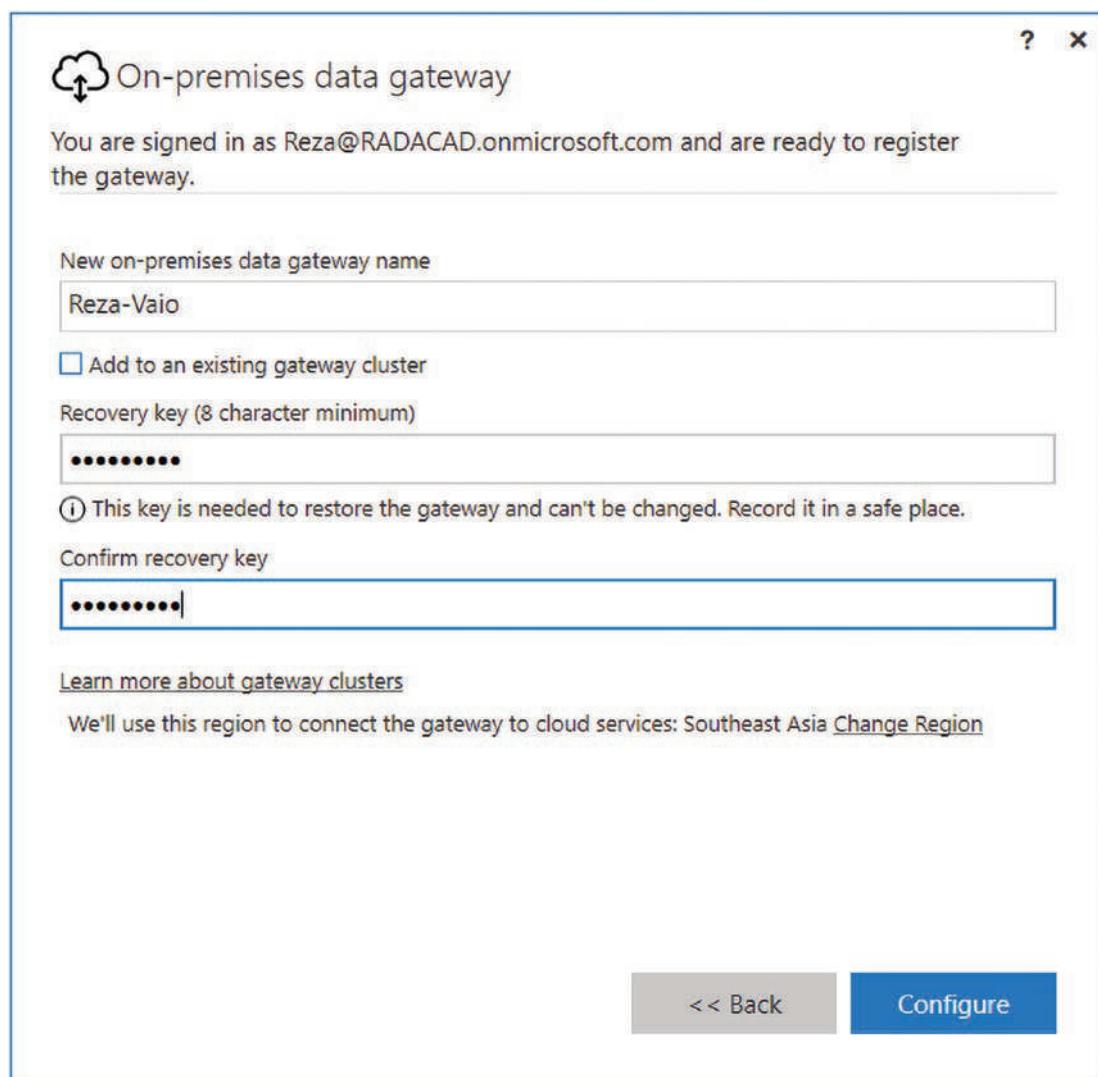
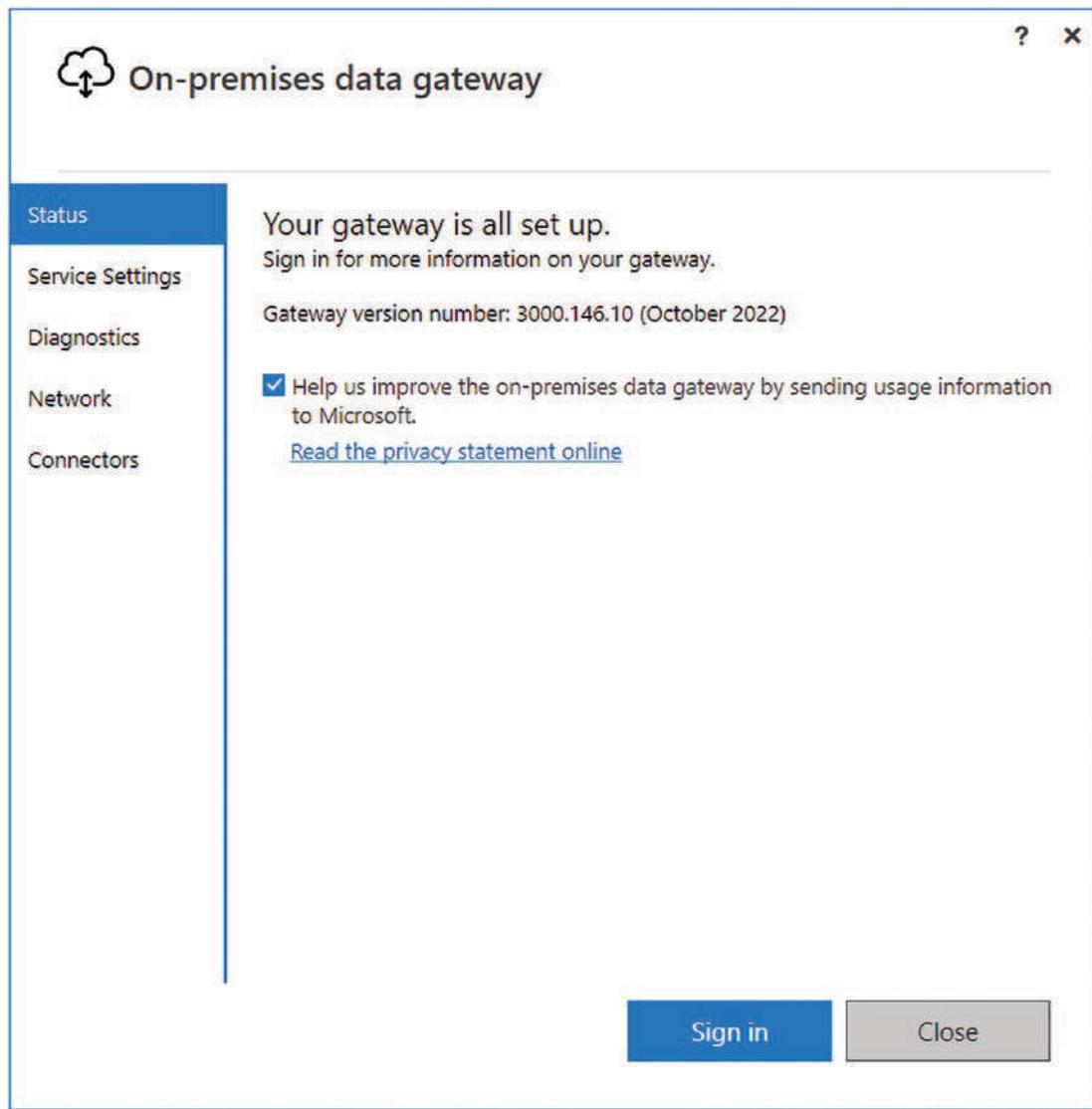


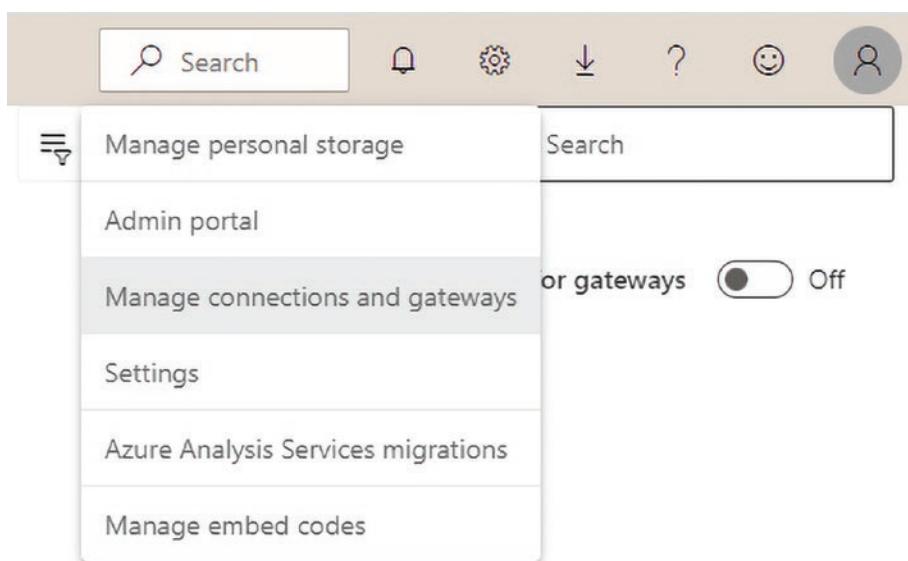
Figure 22-4. Registering a new Gateway

After successfully registering your Gateway, you should see a message that says that the Gateway is online and ready to go, as shown in Figure 22-5.



**Figure 22-5.** Power BI on-premises Gateway status

Now you can see the Gateway in the Power BI service under your account as well. In the Power BI Service, click the Setting icon, and then choose Manage Gateways, as shown in Figure 22-6.



**Figure 22-6.** Managing Gateways from the Power BI Service

You should see all the Gateways set up under your account. In Figure 22-7, you can see that I have a few Gateways in my account.

The screenshot shows the 'Data (preview)' section of the Power BI Service. On the left is a sidebar with icons for workspace, help, data sources, and other settings. The main area has tabs for 'Data sources' (selected) and 'On-premises data gateways' (highlighted with a red box). Below the tabs is a note about data-gateways. A table lists three gateways:

Name	Contact Info	Users	Status	Gateways
Raza Surface	Raza@...	Leila, Raza, Raza	OK	1
Raza-28-Gateway	Raza@...	Raza, Raza	OK	1
Webserver	Raza@...	Raza	OK	1

**Figure 22-7.** Managing Gateways in the Power BI Service

If you are the tenant administrator, you can turn the Tenant Administration for Gateways option on and see and manage all the Gateways under your organization's tenant, even if you are not the direct administrator of that Gateway. This is demonstrated in Figure 22-8.

The screenshot shows the Power BI desktop interface with the title bar "Power BI - My workspace". In the top right, there are search, refresh, and help icons. Below the title bar, there are links for "Manage gateway installers" and "Get help". A dropdown menu "On-premises data gateway" is open, showing "Selected administration for gateways" with a switch set to "On". The main area is titled "Data (preview)" with tabs for "Data sources", "On-premises data gateway", and "Virtual network data gateway". A note below the tabs states: "The data gateway acts as a bridge, providing quick and secure data transfer between on-premises data and Power BI, Microsoft Flow, Logic Apps, and PowerApps. [Learn more](#)." A table lists various gateways with columns: Name, Contact info, Users, Status, and Gateways.

Name	Contact info	Users	Status	Gateways
Em-Laptop	[REDACTED]	User1	OK	1
DavidBadasd	[REDACTED]	David	OK	1
Gateway	[REDACTED]	User1	OK	1
Mari	[REDACTED]	student2	OK	1
Office Gateway (StudentIT)	[REDACTED]		OK	1
Raza Surface	[REDACTED]	Leila, Raza, Raza	OK	1
Raza-S9 Gateway	[REDACTED]	Raza	OK	1
Raza-ZS Gateway	[REDACTED]	Raza, Raza	OK	1
SOHO-Power-01	[REDACTED]	User1234	OK	1
Webserver	[REDACTED]	Raza	OK	1

**Figure 22-8.** Seeing all the Power BI Gateways under the tenant

## Adding Data Sources

The Gateway itself is just for creating the connection from the cloud to the local domain. In order for your datasets to refresh through this Gateway, you need to add the data sources. Data sources are connections to every on-premises database, file, folder, and so on, that is used in Power BI as a connection.

To add data sources to the Gateway, you first need to check the Power BI file and see what data sources have been used. One easy way of finding that out is to open the \*.pbix file in the Power BI Desktop.

After opening the file, choose **Edit Queries > Data Source Settings** (see Figure 22-9).

The screenshot shows the Power BI Desktop ribbon with the "Home" tab selected. The "Data" tab is active in the ribbon. On the far right, a context menu is open with several options: "Transform Refresh data", "Transform data", "Data source settings" (which is highlighted with a red box), "Edit parameters", and "Edit variables". A tooltip "Manage settings for your data sources." is visible near the "Data source settings" option.

**Figure 22-9.** Data source settings in the Power BI Desktop

In the Data Source Settings area, you will see all the data sources used in the current file. Click every data source, click Change Source, and then copy the path for the file, as demonstrated in Figure 22-10.

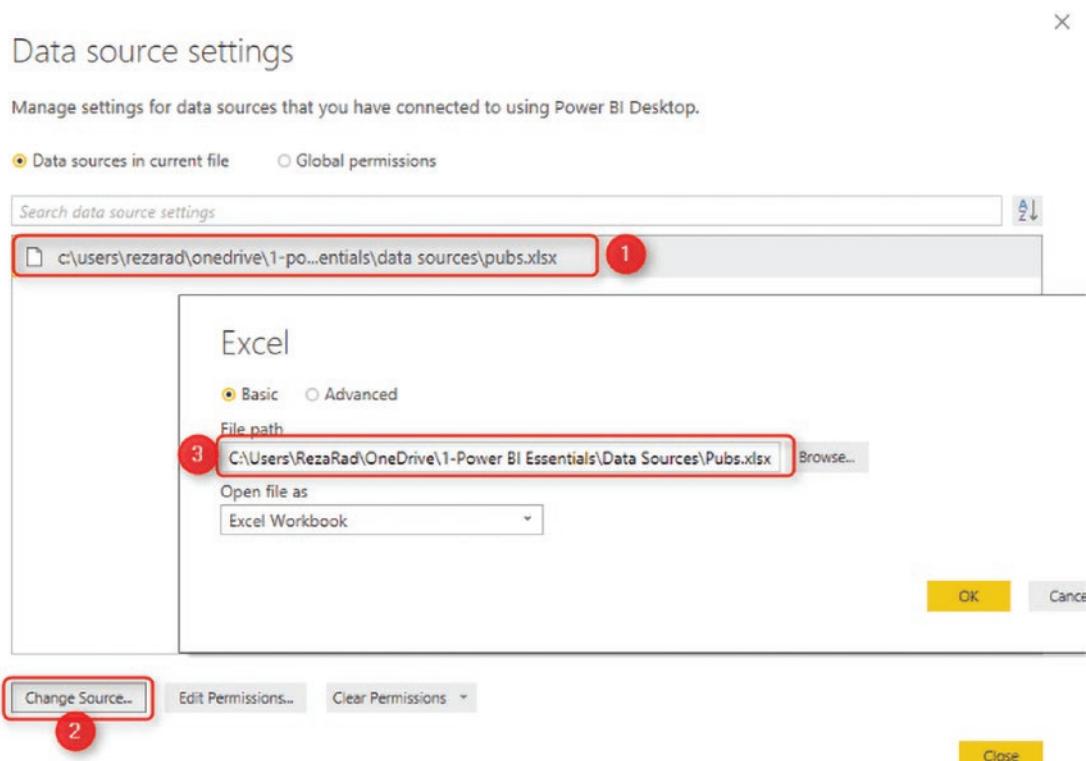
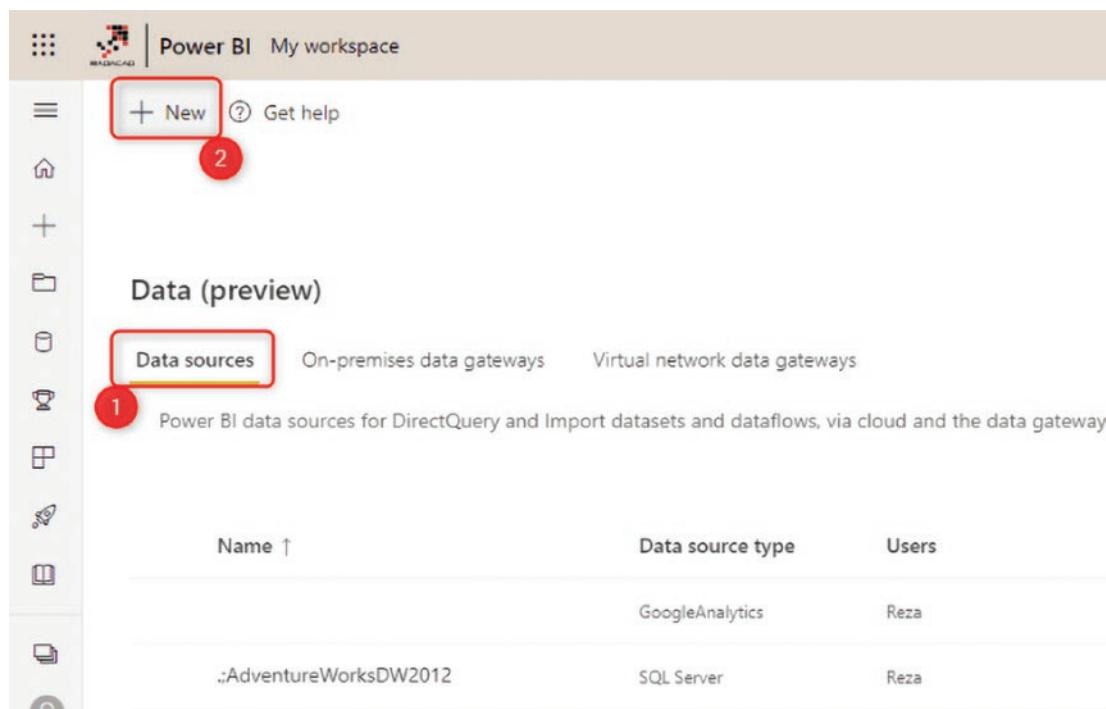


Figure 22-10. Exploring data source settings

As demonstrated in Figure 22-11, you can go to the Power BI Service, Manage Gateway section, select the Gateway you installed previously, and then click Add Data Source.



**Figure 22-11.** Adding a data source

In Add Data Source tab, you need to set some options. The name of the data source is only important for remembering it later. The first important option is Data Source Type. In this example, I choose File because my source is an Excel file. However, this can be a SQL Server database or any other data source.

After choosing the data source type, you need to enter other configurations for that source. I used the file, so I need to specify the full path of the file. This path should be the path of the file from the machine where the Gateway is installed. If the file is in a shared folder path, then that path should be accessible from the machine where the Gateway is installed. This should also be the same path used in the data source configuration of the Power BI Desktop.

You need to enter a username and password to access the data source as well. In this case, because I used a file, the username and password should be the local username and password that have access to that data source from the machine where the Gateway is installed. The username should always have a domain name leading it (domain\username), as shown in Figure 22-12.

New data source X

Gateway cluster name \*

Data source name \*

Data source type \*

Full path \*

**Authentication** ⓘ

Authentication method \*

Windows username \*

Windows password \*

Skip test connection

**General**

Privacy level \*

Create Close

**Figure 22-12.** Setting up data sources properly

If everything is set up correctly, you should see a message that says that the connection was successful.  
*Important note: If you have multiple sources, you must go through this process for every source.*

## Set Up a Gateway Connection to the Dataset

After adding the required data sources, you can create the connection through the Gateway. You should select the data source to configure the Power BI Service. You can then set the dataset in the Power BI Service.

Click Gateway Connection. If your Gateway has all the data sources needed for this dataset, you will see it under Use a Data Gateway, and you can select it, then click Apply, as highlighted in Figure 22-13.

The screenshot shows the 'Datasets' tab selected in the navigation bar. A dataset named 'Settings for Pubs 20220811' is open. In the 'Gateway connection' section (marked with a red circle 1), there is a note about using an On-premises or VNet data gateway. Below this, a toggle switch is set to 'On'. The 'Data sources included in this dataset' section (marked with a red circle 2) lists a single data source: 'File("path": "c:\\users\\reza\_\\\\onedrive\\\\02 power bi from ro okie to rock star - module 2 data modeling with power bi \\data sources\\\\pubs.xlsx")'. This source is mapped to 'pubs file 2022'. At the bottom, the 'Apply' button is highlighted with a red circle 3, while the 'Discard' button is shown in grey.

**Figure 22-13.** Mapping the Gateway's data source to the dataset

You have now configured your dataset to refresh through this Gateway. You can test it by manually refreshing your dataset. To manually refresh your dataset, find it in your workspace (see Figure 22-14).

Name	Type	Owner	Refreshed	Next refresh	Endorsement	Sensitivity
Pubs 20220611	Report	Reza Rad	8/11/22, 3:27:17 PM	—	—	—
Pubs 20220611	Dataset	Reza Rad	8/11/22, 3:27:17 PM	N/A	—	—

**Figure 22-14.** Testing the refresh of the Power BI dataset

After finding your dataset, click Refresh. The last refresh time should update with no error if everything is set up correctly. Congratulations! You have set up the Gateway for your dataset.

## Users and Access Controls for Gateways

There are multiple levels of controls to user access when it comes to Power BI Gateways.

### Gateway Installers

By default, anyone in your organization can install a Gateway (they need a Power BI account). However, as the tenant administrator, you can control this by clicking Manage Gateway Installers, as shown in Figure 22-15.

Name ↑	Contact info	Users	Status	Gateways
Bo_Laptop	labuser1@radacad.com	labuser1	OK	1
DavidRadacad	david@radacad.com	David	OK	1

**Figure 22-15.** Managing the Gateway installers in the organization

In Figure 22-16, you can see that when you turn this option on, you can choose the group or people who can install Gateways.

The screenshot shows the 'Manage gateway installers' settings page. At the top, there is a heading 'Manage gateway installers' and a close button (X). Below the heading, a descriptive text states: 'Manage who can install gateway in your organization. This does not impact gateway administration capabilities.' followed by a link 'Learn more.'. A section titled 'Restrict users in your organization from installing gateways' contains a toggle switch that is set to 'On'. Below this, a section titled 'Users who can install gateways' shows a search bar with the placeholder 'Enter a name or email address'. A list box displays a single user entry: 'Leila Etaati' with an 'X' icon to its right. At the bottom of this section is a yellow 'Add' button. Finally, a section titled 'Current gateway installers' shows a user entry for 'Reza Rad' with the email 'reza@radacad.com', a blue circular profile picture with 'RR' initials, and a close ('X') icon to its right.

**Figure 22-16.** Controlling installing capabilities

This is good for organizations with many Power BI users, and the control of the installation of the Gateway is better to be governed.

## Gateway Users

Each Gateway can have three access types for the users, as shown in Figure 22-17.

The screenshot shows the 'Manage users' interface for a Power BI Gateway. At the top, there's a search bar labeled 'Enter a name or email address'. Below it, a message says 'Share this data gateway with others in your organization' and 'You currently have Admin permissions for this gateway. You can add, remove, and modify users.' A close button 'X' is in the top right corner.

**Shared with:**

- Reza Rad (Admin) - Shared with this user.
- Reza Rad (Admin) - Shared with this user.

**User Roles:**

- Connection Creator: Allows the user to create data sources and connections on the gateway.
- Connection Creator with resharing: Allows the user to create data sources and connection on the gateway and reshare gateway access.
- Admin: Allows the user to create data sources and connections on the gateway, manage gateway access, configurations, credentials and updates.

**Figure 22-17.** User access to the Power BI Gateway

## Gateway Connection Creator

This type of user can create data sources under the Gateway and use them for a connection to the datasets and dataflows.

## Gateway Connection Creator with Resharing

In addition to creating data sources and using them in connections, this user can also reshare access to the Gateway.

## Gateway Admin

This user has full control of the Gateway. In addition to adding and removing data sources, this user can manage access to the Gateway, control the settings, and remove the Gateway.

## Data Source Users

In addition to giving access at the Gateway level, you can give users access at the data source level. This is more granular access and is helpful when a user only needs permission or access to a few data sources and not the entire Gateway. The data source users can also have three types of access, as shown in Figure 22-18.

The screenshot shows the 'Manage users' interface in Power BI Gateway. At the top, it says 'Users who can use this data source in published Power BI reports. [Learn more.](#)' and 'You currently have Admin permissions for the associated gateway. You can add, remove, and modify users.' Below this, there's a search bar labeled 'Enter a name or email address' and a list titled 'Shared with' containing 'Reza Rad User'. To the right, there are three radio button options for roles:

- User**: Allows the user to use the data source.
- User with resharing**: Allows the user to use the data source and reshare with others.
- Owner**: Allows the user to use the data source, manage data source configurations and credentials.

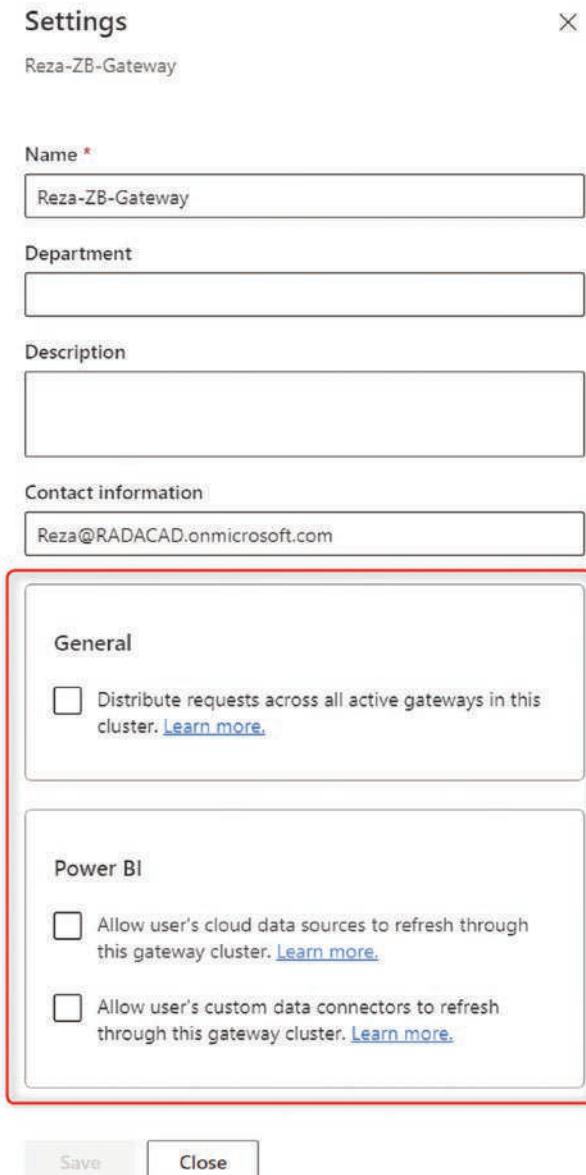
**Figure 22-18.** Data source users in a Power BI Gateway

- **User:** This is just a pure user of the data source. This user cannot change the data source defined under the Gateway but can use it to connect to a Power BI dataset or dataflow.
- **User with resharing:** In addition to being the user, this user can also reshare this data source with other users.
- **Owner:** This user has full control of the data source itself but not of other data sources or the Gateway.

You must understand the differences between data source users and Gateway users and correctly assign access to users based on their use cases. Too much access can sometimes be challenging for a user not trained to use it with caution.

## Additional Settings for Gateway

There are a few other important additional settings for the Gateway, shown in Figure 22-19. An explanation for each setting follows.



**Figure 22-19.** Additional settings for Power BI Gateways

## Distribute Requests Across All Active Gateways in This Cluster

This sets load balancing on the Gateways. This is very helpful if the Gateway is under many parallel requests. When you set up a Gateway cluster (a group of Gateway installations bundled together to serve as one Gateway), you can enable this functionality. To learn more about load balancing, visit [learn.microsoft.com/en-nz/data-integration/gateway/service-gateway-high-availability-clusters#load-balance-across-gateways-in-a-cluster](https://learn.microsoft.com/en-nz/data-integration/gateway/service-gateway-high-availability-clusters#load-balance-across-gateways-in-a-cluster).

## Allow User's Cloud Data Sources to Refresh Through This Gateway Cluster

When you need to combine multiple data sources in a single Power Query table (when one of the data sources is on-premises and another is cloud-based), enabling this option will give you that ability. Otherwise, you may need to create two separate queries and combine them.<sup>1</sup>

## Allow User's Custom Data Connectors to Refresh Through This Gateway Cluster

You can build your own Power Query custom connector and use it through a Gateway. You need to set up the custom connector settings in the installed Gateway, shown in Figure 22-20, and in the service, you need to enable this option.

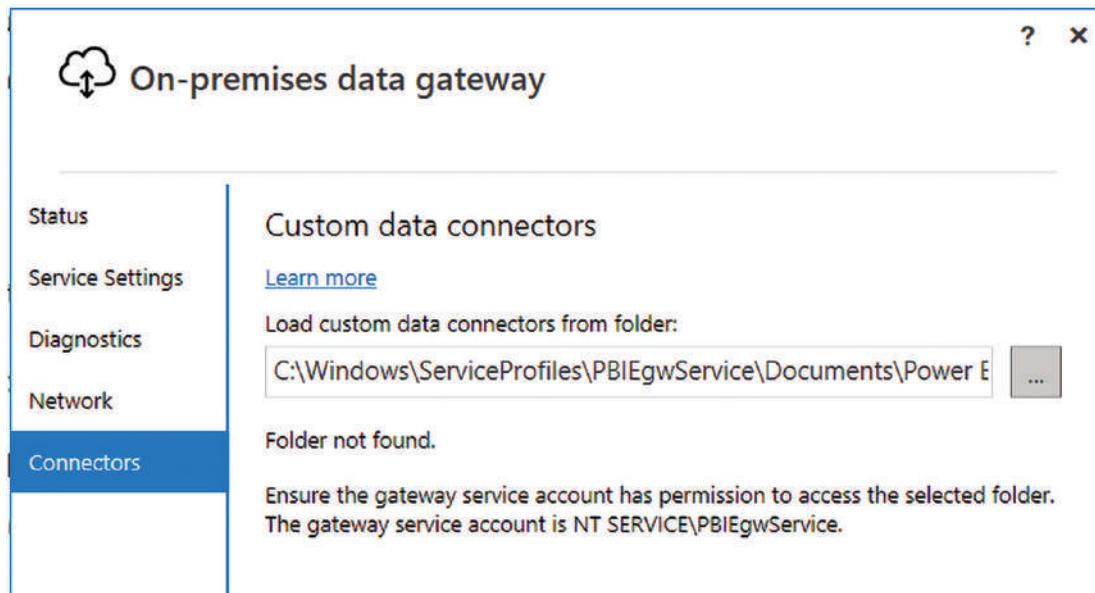


Figure 22-20. Using a custom connector in a Power BI Gateway

<sup>1</sup> Learn more about it at [learn.microsoft.com/en-nz/power-bi/connect-data/service-gateway-mashup-on-premises-cloud](https://learn.microsoft.com/en-nz/power-bi/connect-data/service-gateway-mashup-on-premises-cloud)

## Troubleshooting

There are a few scenarios in which you may face an issue when setting up the Gateway. This section explains them:

- You cannot see the Gateway when you go to your dataset setting, most probably because you did not add all the data sources needed for that dataset. Go to the Power BI Desktop and check whether you have added all the data sources.<sup>2</sup>
- Gateway has a logging system that can be helpful when an issue comes up. You can enable additional logging and access the Gateway logs from the on-premises data Gateway installed in the application. This process is shown in Figure 22-21.<sup>3</sup>

---

<sup>2</sup>Analysis Services Live Connection with Gateway requires more configuration, which I explain in this article: [radacad.com/step-by-step-walk-through-on-premises-live-sql-server-connection-with-power-bi-enterprise-gateway](http://radacad.com/step-by-step-walk-through-on-premises-live-sql-server-connection-with-power-bi-enterprise-gateway)

<sup>3</sup>There are a few known issues with the on-premises Gateway, which you can read more about here: [docs.microsoft.com/en-us/power-bi/service-gateway-onprem-tshoot](https://docs.microsoft.com/en-us/power-bi/service-gateway-onprem-tshoot) as well as a few known issues with the personal Gateway, which you can read more about at [docs.microsoft.com/en-us/power-bi/personal-gateway](https://docs.microsoft.com/en-us/power-bi/personal-gateway)

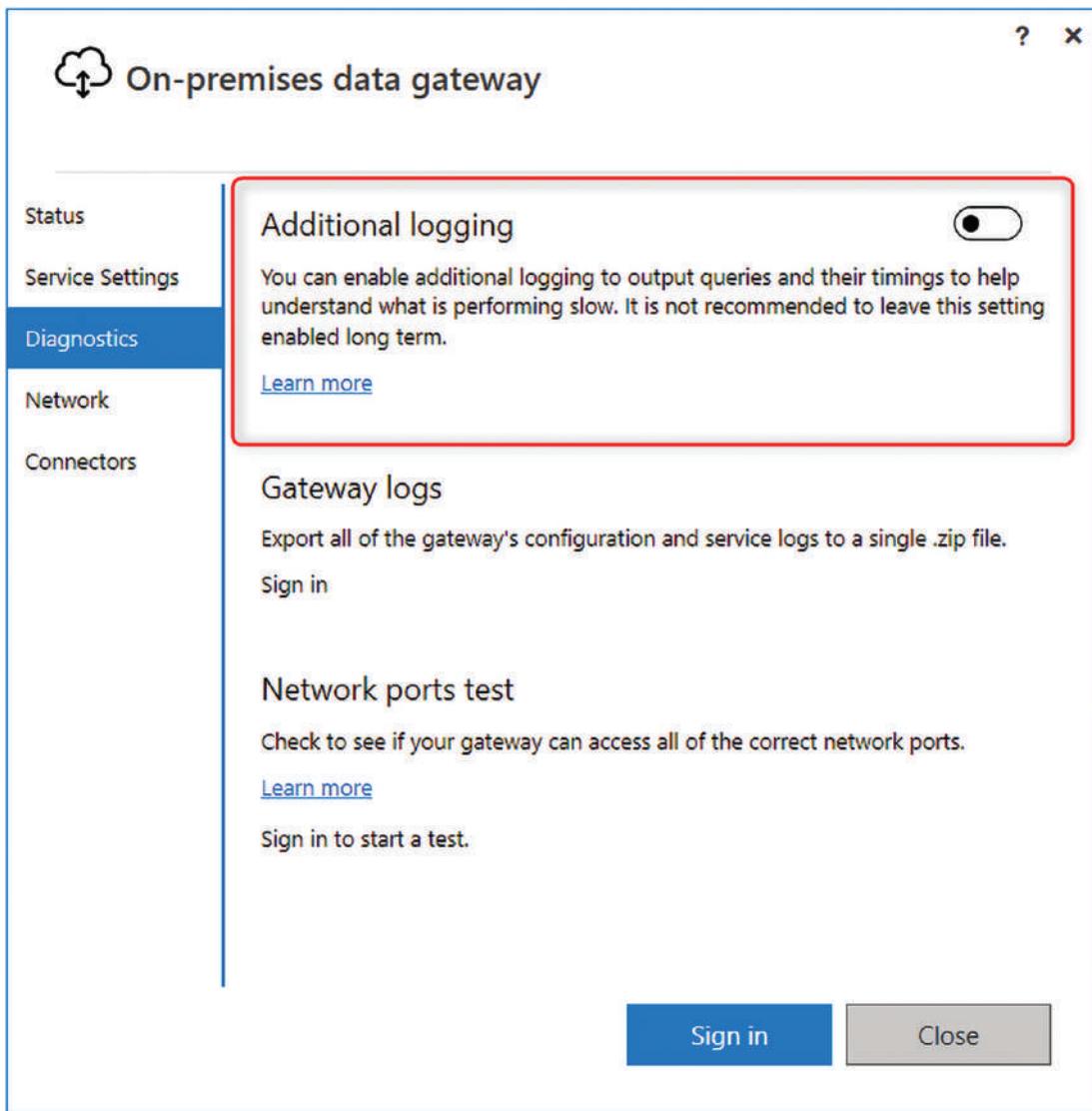


Figure 22-21. Enabling additional logging in a Power BI Gateway

## Summary

In this chapter, you've learned about Gateways. Gateways connect the Power BI cloud-based dataset and the data source on-premises. You learned that Gateways are only required for on-premises connections. There are two modes to install a Gateway—personal and recommended (on-premises). You learned that the on-premises recommended Gateway can serve more than one developer at a time and can be used for Power BI, PowerApps, and a few other applications. It also supports multiple connection types.

The chapter also explained the installation and configuration of the Gateway and connected one Power BI dataset to it. The key to using a Gateway is to add all the required data sources under it and then map them to the dataset.

## CHAPTER 23



# Power BI Licensing

There are many ways to license Power BI. Users often do not understand which features are included in which licensing plans. In this chapter, you learn about all the different licensing plans for Power BI, the scenarios in which to use the licensing, and scenarios in which you may need to change your chosen licensing plan. This chapter is intended to help you to find the most cost-effective licensing plan for your requirements. You learn about the following licensing options:

- Power BI Free
- Power BI Pro
- Power BI Premium Per User
- Power BI Embedded
- Power BI Premium
- Power BI Report Server Only licensing

## Two Types of Licensing

Power BI licensing can be separated into two main categories:

- User-based licensing
- Capacity-based licensing

To understand this, let's look at the Power BI Service setup. Power BI Service is a cloud-based hosting solution for Power BI objects. This cloud-based hosting solution offers two options for Power BI users—they can either use a shared capacity of resources (CPU, memory, and all the shared resources of the cloud computing for the Power BI service in the cloud), or they can use capacity that is dedicated just for themselves and not shared with anyone else. When you choose user-based licensing, you are using a shared capacity. Capacity-based licensing allows you to use a capacity dedicated to yourself. These two types of licensing also have some interactions. For example, reports hosted in a dedicated capacity can be published by a user-based license and consumed by a user-based license. You learn more about this later in this chapter.

It is essential to say that these two types of licensing are mainly designed for the Power BI Service. If you intend to use Power BI as a full on-premises option (using Power BI Report Server), there is a separate licensing option, which you learn about later in this chapter.

All the prices mentioned in this chapter are based on USD.

## The Power BI Desktop

Note that the Power BI Desktop, which is the tool for authoring and creating the data model, the report, and the PBIX files, is free. You do not even need an account to use it. It is free to use for anyone worldwide; no signup is needed; you can download and install it.

There is a log-in/sign-in option when you open the Power BI Desktop, but you can skip that, and you can still use the Power BI Desktop. A few of the features, such as browsing the gallery of custom visuals, might not be available if you are not signed in, but you can still use the main capabilities of the Power BI Desktop, which involve creating Power BI models and reports.

## Why Do I Need a License if the Power BI Desktop Is Free?

The Power BI Desktop creates reports, it does not consume them. The Power BI Desktop is a developer tool. Although you can install it on the end user's machine and give them the PBIX file to view it, the Power BI Desktop is not designed to be an end user tool. Here are some reasons why:

- The Power BI Desktop cannot be installed on all devices (phones or some tablets, for example). This is important because many end users want to use mobile devices to consume reports.
- The Power BI Desktop empowers users to edit and change datasets and reports. The end user might unknowingly remove something from the report or change something that causes it to stop working correctly. Ideally, an end user tool should not allow users to change reports.
- If you use the Power BI Desktop to build the model, your model isn't following the best practices of the multi-layered architecture, and you are creating silos of reports in your organization. This is likely to cause a big failure in the Power BI adoption across your organization.
- There isn't a way to add security around the PBIX file shared outside of your organization.
- There isn't a way to track and monitor the usage of the Power BI reports.
- You will end up with many versions of the PBIX files floating around, and users will likely be confused about which version is correct.
- And many other issues.

Unfortunately, in some organizations, the Power BI Desktop is still used for sharing and consuming reports. This must be stopped; this list explains just a few of the issues involved with this procedure. So, how should you share reports with other users?

To share the reports with other users, you must deliver them from a hosting solution to which you can add governance, security, and architecture. You need a hosting solution that can give you the options to share a read-only version of a report if needed and also give you the option to push changes and updates through a developer channel. Power BI offers two hosting options—the Power BI Service and the Power BI Report Server. To use either of these two, you need a Power BI license.

Now that you understand why licensing is needed in the world of Power BI, let's talk about the details.

# User-Based Licensing

User-based licensing comes in three flavors:

- Power BI Free
- Power BI Pro
- Power BI Premium Per User

These three options provide a different range of features. Power BI's user-based licensing options are under Office 365 user management. That means that the Office 365 administrator of an organization can assign licenses to the users.

## Power BI Free

Certain features in Power BI are free to use (apart from the Power BI Desktop); however, you need a free Power BI license. The Power BI administrator or the Office 365 administrator can assign free licenses. You can also sign up for a free Power BI account if your organization allows you to do so.

A free Power BI account will give you access to the Power BI Service. You can publish and host your reports in the service. You can see the reports using a web browser or the Power BI mobile app. However, you cannot share the report with others.

If you have a free Power BI account and want to see a report that's been shared with you, depending on how the report is shared and where it is hosted, you might be able to see it using your Power BI Free account. I explain that ambiguity more later in this chapter.

Note that there is a free way to share Power BI content without needing Power BI user accounts, called Publish to Web. This method is not secure and is not recommended for confidential data. This chapter covers the licensing required to share confidential reports and data throughout an organization.

## Power BI Pro

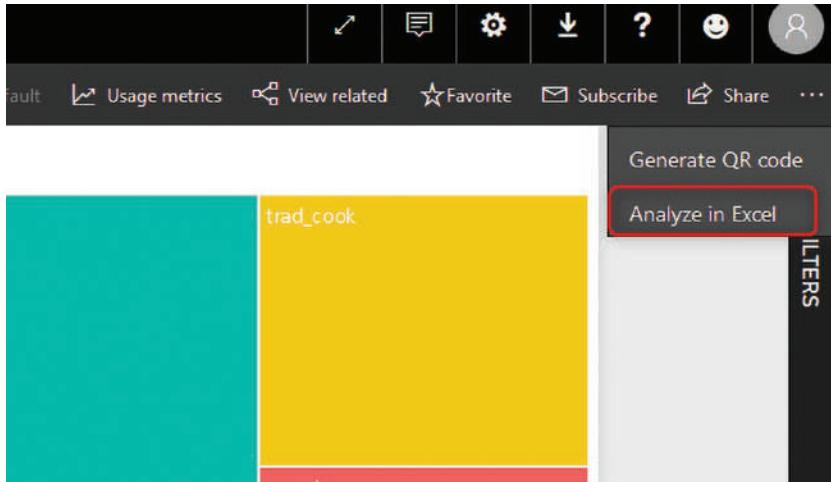
Power BI Pro is the per-user subscription for Power BI. At the time of writing this chapter, it costs \$9.99 USD per user per month. With Power BI Pro, you can get everything that a free account has, plus many other Power BI Service features, as well as other methods of sharing reports. A Pro license is designed for developers.

## Sharing

With Power BI Pro, you can use all sharing methods except Power BI Embedded (which comes with a different licensing option). You can use Simple Sharing, Workspaces, Power BI Apps, and Embed in SharePoint Online. If you have a Power BI Pro license, you can share the reports internally or externally (if you're allowed to by the Power BI Administrator). However, if you share a report with someone else, they must also have a paid Power BI license to view the reports. If you have a Pro license and share a report with a free user, they cannot see it unless they upgrade to the Pro user or Premium per user. (There is an exception to this, which is when the report is hosted in a dedicated premium capacity, in which the free users can consume the Power BI content through apps, and I explore this option later in this chapter.)

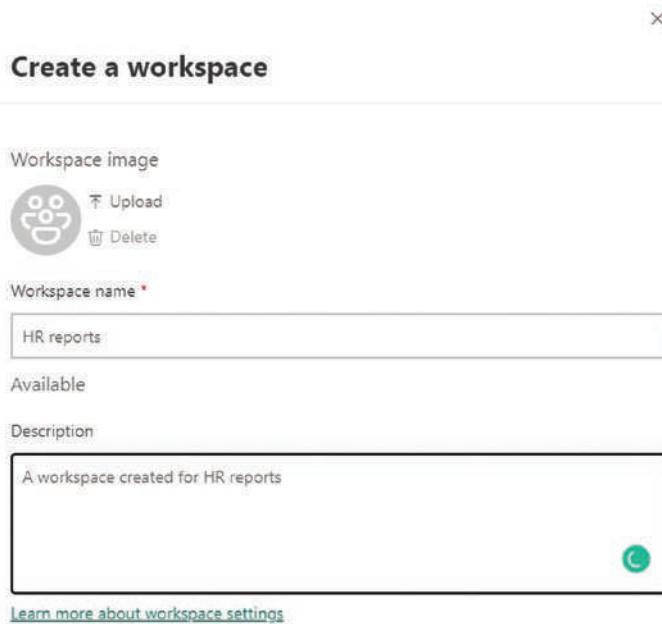
## Developer Functions in the Service

With Power BI Pro, you get some integration features of Power BI, such as Analyze in Excel, highlighted in Figure 23-1. If a member of your developer team wants to connect to the Power BI dataset using Excel and analyze it, they can do so.



**Figure 23-1.** Power BI Pro gives you all the authoring features (such as Analyze in Excel) and sharing options

With a Pro license, you can create workspaces, as shown in Figure 23-2, or be part of a workspace. A Power BI workspace is like a development environment. You can collaborate with other developers in workspaces.



Advanced 

**Figure 23-2.** The Power BI Pro license enables you to use workspaces

A Pro license also includes actions so you can maintain your Power BI objects, such as the model size of 1GB in a capacity of 10GB, the use of dataflows, paginated reports, shared datasets, monitoring, and some governance around the objects.

## Power BI Pro Licenses Are Designed for Developers

A Pro license enables you to use most of the features that a Power BI developer needs. You may, however, need a higher-level license to use some of the advanced features of Power BI. The Premium Per User license (in short, PPU) is designed for advanced features of Power BI at a user-based licensing option. At the time of writing this chapter, the cost for a PPU license is \$20 USD per user per month.

## Power BI PPU: Premium Per User

Although the Power BI Pro license gives you most of the features you need for Power BI development in an organization, some features are still not included.

- **Bigger model size:** If you use the Power BI Pro, your model size cannot exceed 1GB. This is a compressed size of the PBIX file; however, if you are dealing with a lot of data, that size might not fit your purpose. With a PPU license, you can have a model size of 100GB.

- **More refreshes:** Sometimes, the eight refreshes of the Power BI dataset offered by the Power BI Pro license is not enough; you may need to refresh it more frequently. The PPU license gives you up to 48 refresh times a day, plus unlimited API refreshes.
- **AI functions:** Some AI functionalities in Power BI work with cognitive services in Azure. To use some of those features, you need a PPU license. A Power BI Pro license doesn't cover those features.
- **Datamarts and advanced dataflow features:** The most recent announcements about Power BI datamarts reveal a new way of building Power BI solutions, and this is limited only to PPU and Premium licenses. On the other hand, if you want to use some of the advanced features of the Power BI dataflows; such as Computed Entity and Enhanced Compute Engine, you need a PPU license (or Premium).
- **Deployment pipelines:** You can create an application lifecycle management system in Power BI by assigning workspaces to the development, test, and production environments. You can have deployment pipelines managing the changes from one environment to another. This feature is limited to PPU and Premium.

PPU offers many more features than Pro. Some of these features make the maintenance of the model in the Power BI service more accessible (deployment pipelines), some of them offer a better architecture (advanced dataflow and datamarts), and some of them offer extra features (such as AI functionalities). Microsoft introduced PPU for companies with fewer users who do not need dedicated capacity but still want to use premium functionalities. I think the extra features that PPU provides are so comprehensive that I recommend it to organizations.<sup>1</sup>

## Capacity-Based Licensing

The next category of licensing in Power BI is the capacity-based licensing option. It is offered in two ways:

- Power BI Embedded
- Power BI Premium

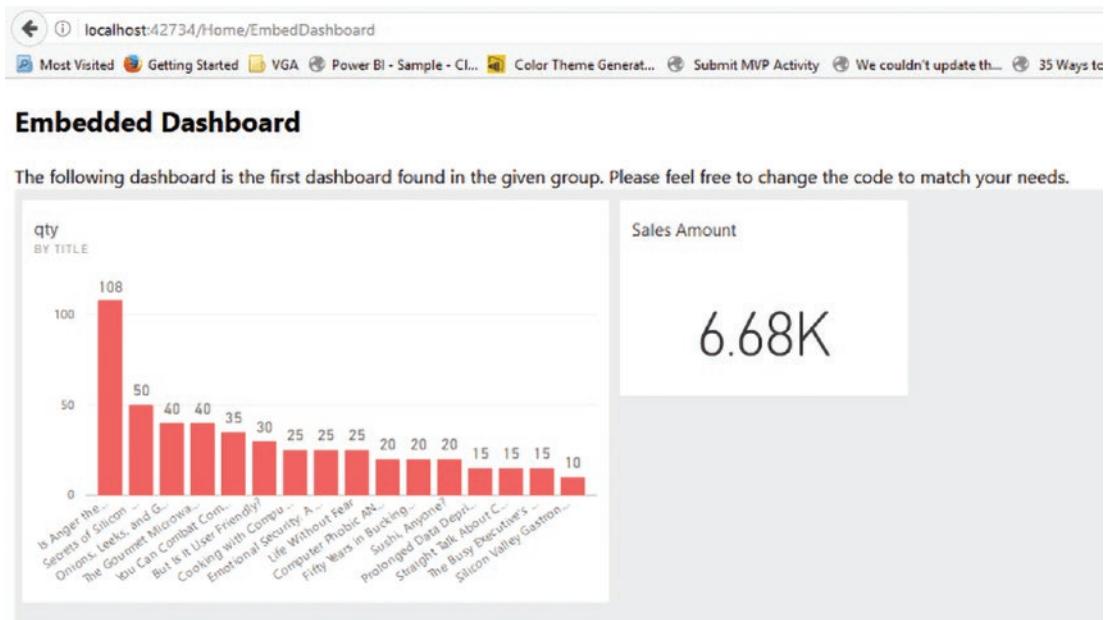
Each of these options has different SKUs, pricing levels, and tiers. The main thing is that the capacity-based licensing options cannot be used alone; you need to combine them with a user-based license. For example, you need a Power BI Pro license for the developer hosting the Power BI reports in a workspace with a Power BI Premium capacity.

### Power BI Embedded

If you ever want to embed Power BI content into a custom application and use a custom application's user management, Power BI Embedded is the licensing plan for you (see Figure 23-3). This licensing plan is based on page renders.

---

<sup>1</sup> Explaining PPU and its use cases would take an entire chapter unto itself. I encourage you to read my article about Power BI PPU at [radacad.com/what-is-premium-per-user-license-for-power-bi-and-what-is-it-good-for](http://radacad.com/what-is-premium-per-user-license-for-power-bi-and-what-is-it-good-for)



**Figure 23-3.** Power BI Embedded licensing is calculated based on page renders

Every refresh on the page that has Power BI content in it is a page render; if you select a slicer, that causes another page render. If you click a column in a column chart and that causes interactivity of other charts, then that is another page render.

With Power BI Embedded, you can reserve buckets of page renders per peak hour. Table 23-1 lists the costs at the time of writing this chapter.

**Table 23-1.** Power BI Embedded Tiers and Pricing In USD<sup>2</sup>

Node Type	Virtual Cores	Memory	Frontend/Backend Cores	Peak Renders per Hour	Price
A1	1	3GB RAM	0.5 / 0.5 <sup>1</sup>	1 – 300	\$1.0081/hour
A2	2	5GB RAM	1 / 1 <sup>1</sup>	301 – 600	\$2.0081/hour
A3	4	10GB RAM	2 / 2	601 – 1,200	\$4.0242/hour
A4	8	25GB RAM	4 / 4	1,201 – 2,400	\$8.0565/hour
A5	16	50GB RAM	8 / 8	2,401 – 4,800	\$16.13/hour
A6	32	100GB RAM	16 / 16	4,801 – 9,600	\$32.26/hour

The pricing in Table 23-1 may scare you, and you may immediately think of not going through the embedded path. However, there are some scenarios in which Power BI Embedded can be a more cost-effective option than Pro. Here is an example:

<sup>2</sup>[azure.microsoft.com/en-us/pricing/details/power-bi-embedded/](https://azure.microsoft.com/en-us/pricing/details/power-bi-embedded/)

Assume that you have 100 users for your Power BI solution. Your users are not connecting simultaneously to use Power BI reports. You may have maximum page renders of 300 per hour if you use Embedded. In such a case, Embedded for that scenario would cost you about \$700 USD per month, whereas the Power BI Pro for 100 users would be \$1000 USD per month. This means saving \$3,600 USD per year. Power BI Embedded can be more cost-effective than Pro.

An important note to consider when you think about Embedded is the hidden cost of a web developer. Power BI Embedded brings Power BI content embedded into your custom application, and who is going to do that? A web developer. If later on through the path, you want to change the way that users are working with the application, who is going to make that change? A web developer.

Power BI Embedded gives you the ability to embed Power BI content into a custom application and share it based on a custom user management through that application.

## Power BI Premium

Power BI Pro is expensive when you have a large user base, and Embedded needs constant maintenance by a web developer. If you have a large user base (say 10,000 users), Power BI Premium is the best licensing option for you. Power BI Premium is designed for large user base scenarios where the data size is huge.

Power BI Premium is not priced per user, it is per node. In Power BI Premium, you pay for nodes, which have dedicated capacity and resources. Table 23-2 shows the existing nodes and their details at the time of writing this chapter.

**Table 23-2.** Power BI Premium Licensing Tiers

Capacity		Dataset					Dataflow	Export API
Capacity SKUs	V-cores	Max memory (GB) <sup>1, 2, 3</sup>	DirectQuery/Live connection (per second) <sup>1, 2</sup>	Max memory per query (GB) <sup>1, 2</sup>	Model refresh parallelism <sup>2</sup>	Dataflow parallel tasks <sup>5</sup>	Max concurrent pages <sup>6</sup>	
EM1/A1	1	3	3.75	1	5	4	20	
EM2/A2	2	5	7.5	2	10	8	25	
EM3/A3	4	10	15	2	20	16	35	
P1/A4	8	25	30	6	40	32	55	
P2/A5	16	50	60	6	80	64	95	
P3/A6	32	100	120	10	160	64	175	
P4/A7 <sup>4</sup>	64	200	240	10	320	64	200	
P5/A8 <sup>4</sup>	128	400	480	10	640	64	200	

Pricing starts at P1 nodes, which cost \$5K USD per month, P2 is twice that price, and P3 is twice P2. As you see, the licenses are in two categories of EM (lightweight for embedding purposes), and P SKUs (what premium nodes are normally called). If you use Premium licenses, you can also use Embedded features.

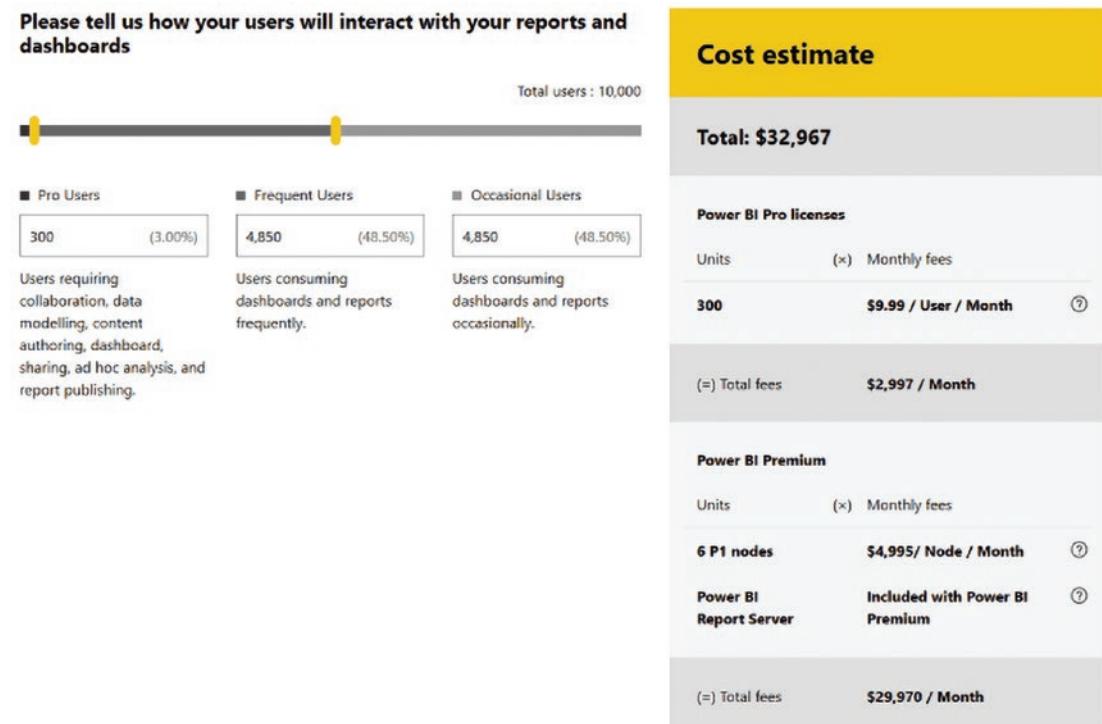
Power BI Premium includes all the features that PPU offers, plus some other exciting options. The most important of them is the ability to share content with free users.

## Sharing Content with Free Users

This is perhaps the most significant advantage of the Premium license over PPU. Suppose you have Power BI content hosted in a workspace. If your workspace is a Premium workspace (connected to a Premium capacity), you can create an app on the workspace and share the workspace's content through that app with the end users, even if they don't have a paid Power BI license. They can have a Power BI Free account. In other words, a Power BI report hosted in a Premium-capacity workspace can be shared with thousands of free Power BI account holders.<sup>3</sup>

### Premium Calculation Scenario

It can be difficult to determine how many nodes you need, or how big the nodes need to be, when creating your Power BI solution. Figure 23-4 offers an example calculation for 10,000 users.



**Figure 23-4.** Sample Pro BI costs

If you compare the total costs of \$33K per month with \$100K per month (\$100K per month if all 10,000 users purchase Power BI Pro), you can see how Power BI Premium can be more cost-effective with a larger user base scenario. The idea behind creating the Power BI Premium licensing is that users who are only reading a report should not pay Pro pricing.

<sup>3</sup>I explain this in more detail in my article at [radacad.com/the-read-only-license-for-power-bi](http://radacad.com/the-read-only-license-for-power-bi)

## Extra Features in Premium

Power BI Premium also provides some extra features. Some of these features have been released, and some are still in progress and in the roadmap:

- Dedicated Power BI resources
- Massive dataset storage and no user quotas: 100TB storage
- More frequent dataset refreshes: 48 times a day rather than 8 times a day
- Power BI Report Server licensing to use Power BI Report Server on-premises
- Support for larger datasets (up to 400GB)
- Datamarts
- AI functionalities
- The ability to share organizational apps with free users
- Autoscale

Autoscale enables you to add vCores to your capacity as the need arises (this feature requires the Power BI Premium per capacity Gen2).

---

Power BI Premium licensing is designed for large user base scenarios. This licensing tier gives you many extra features as well as incremental load.

---

To read more about Power BI Premium, read my blog post at [radacad.com/power-bi-premium-is-it-for-you-or-not](http://radacad.com/power-bi-premium-is-it-for-you-or-not).

## The Difference Between A SKUs and P/EM SKUs

You perhaps noticed that there are similarities in A, EM, and P SKUs over the pricing tables for Premium and Embedded and wondered about the differences between them. You can embed Power BI reports into web pages in more than one way. One method requires users to log in using their organizational accounts (such as Secure Embed), another method doesn't need the users to have Power BI accounts at all, and the application can manage the users itself. To use organizational accounts, you need to use EM or P licenses. However, for the application to have its own user management, EM, P, or A licenses can be used. A licenses are part of the Azure offering, whereas the EM or P licenses are part of Office's offering. An A license is billed hourly, whereas the EM and P licenses are on monthly billing periods.

Table 23-3 features several scenarios in which to use different SKUs.

**Table 23-3.** Scenarios to Use Each SKU for Power BI Licensing<sup>4</sup>

Scenario	Azure	Office
	(A SKU)	(P and EM SKUs)
Embed for your customers (app owns data)	✓	✓
Embed for your organization (user owns data)	✗	✓
Microsoft 365 apps (formerly known as Office 365 apps)	✗	✓
• Embed in Teams • Embed in SharePoint		
Secure URL embedding (embed from Power BI service)	✗	✓

Table 23-4 lists some capacity considerations for each SKU.

**Table 23-4.** Capacity Considerations for Each SKU<sup>5</sup>

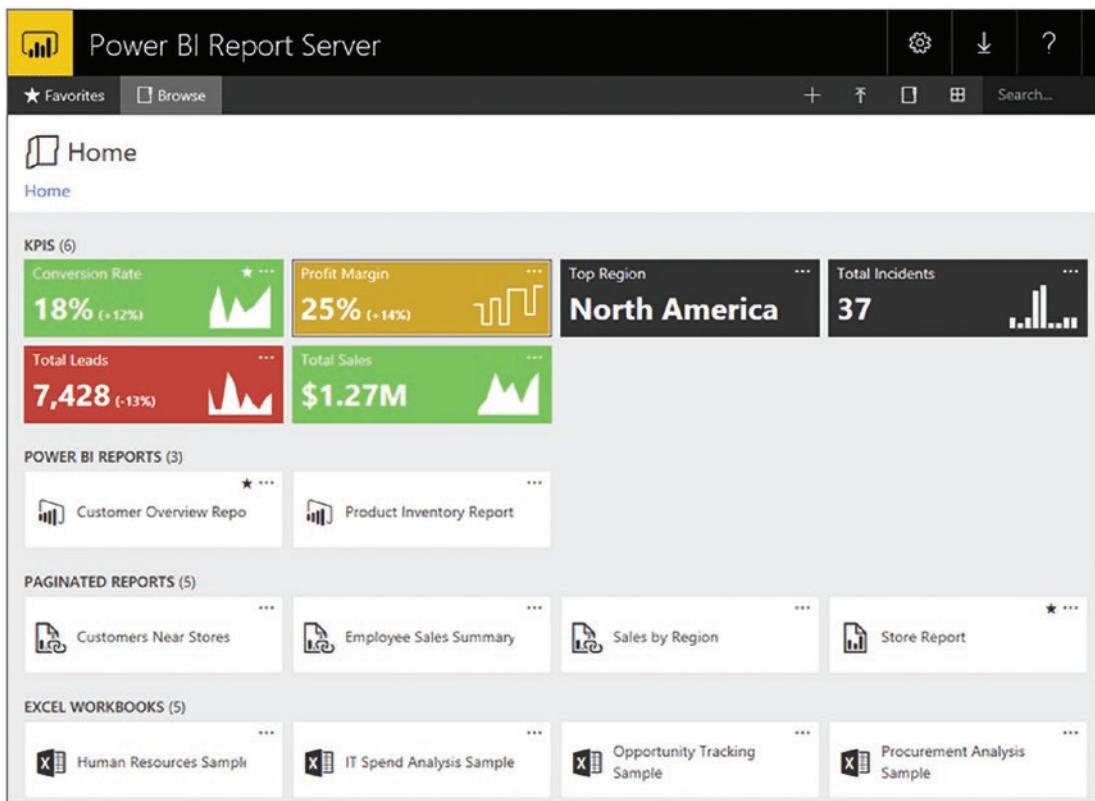
Payment and usage	Power BI Embedded	Power BI Premium	Power BI Premium
Offer	Azure	Office	Office
SKU	A	EM	P
Billing	Hourly	Monthly	Monthly
Commitment	None	Yearly	Monthly or yearly
Usage	Azure resources can be:	Embed in apps, and in Microsoft applications	Embed in apps, and in Power BI service
	• Scaled up or down • Paused and resumed		

## SQL Server Enterprise Edition Plus Software Assurance

The combination of SQL Server Enterprise Edition and software assurance provides you with licensing for Power BI Report Server (see Figure 23-5). The Power BI Report Server is the on-premises hosting solution for Power BI. Contact your Microsoft contact for these products.

<sup>4</sup>[learn.microsoft.com/en-us/power-bi/developer/embedded/embedded-capacity](https://learn.microsoft.com/en-us/power-bi/developer/embedded/embedded-capacity)

<sup>5</sup>[learn.microsoft.com/en-us/power-bi/developer/embedded/embedded-capacity](https://learn.microsoft.com/en-us/power-bi/developer/embedded/embedded-capacity)



**Figure 23-5.** The Power BI Report Server can be available as part of SQL Server Enterprise licensing

---

If you already have SQL Server Enterprise Edition licensing in your organization, and you intend to use Power BI only on-premises using the Power BI Report Server, buying Software Assurance is the more cost-effective option.

---

## Summary

You learned about six different licensing plans for Power BI in this chapter. You learned what features are included in each plan and in which situations they are most cost-effective. Table 23-5 summarizes the features in each licensing plan.

**Table 23-5.** Power BI Licensing Comparison Table

	Power BI Free	Power BI Pro	Power BI Premium Per user	Power BI Embedded (A SKU)	Power BI Premium Per capacity (EM and P SKU)	SQL Server Enterprise with Software Assurance
Power BI Desktop	Yes	Yes	Yes	Yes	Yes	
Power BI Mobile app	Yes	Yes	Yes		Yes	
Dataflow		Yes	Yes	Yes	Yes	
Incremental Refresh		Yes	Yes	Yes	Yes	
Deployment Pipelines			Yes	Yes	Yes	
XMLA endpoint read/write			Yes	Yes	Yes	
Datamart				Yes	Yes	
Advanced AI (text analytics, image detection...)				Yes	Yes	
Secure Embed					Yes	
Autoscale					Yes	
Consume content without paid user license					Yes	
Embed in custom application					Yes	
Paginated Reports	Yes	Yes	Yes	Yes	Yes	
Report Server					Yes	Yes
maximum model size	1 GB	1 GB	100 GB	400 GB	400 GB	As per server spec
Refresh rate		8/day	48/day	48/day	48/day	unlimited
Maximum storage	10 GB / user	10 GB / user	100 TB	100 TB	100 TB	As per server spec

Last but not least, if you want to set up a free environment in which you can try Power BI Service features, you can use the Power BI Sandbox.<sup>6</sup>

---

<sup>6</sup>This sits beyond the scope of this book. Learn more about it at <https://radacad.com/power-bi-sandbox-an-environment-to-learn-power-bi-service-for-free>

## CHAPTER 24



# Power BI Admin Portal and Tenant Settings

In the world of Power BI, there are some configurations in the Desktop tool and in the service. One of the more critical configurations is the Tenant Settings of the Power BI administrator panel. Tenant Settings include a list of highly important configurations across your Power BI tenant. If you don't configure the settings properly, it might result in leaking data, authorizing people who should not be authorized to see reports, and many other catastrophic scenarios. In this chapter, you learn about the configurations available in Tenant Settings and the recommended options for each.

## Power BI Administrator

By default, the Office 365 administrator is the Power BI administrator. However, you can add a specific Power BI administrator by selecting the Power BI administrator role from the Office 365 Portal. Here are the details of assigning a Power BI administrator role to a user.

Log in to [portal.office.com](https://portal.office.com) with an Office 365 administrator account. Go to the Admin panel. Find the user in the list of active users and select them from the list, as shown in Figure 24-1.

A screenshot of the Microsoft 365 Admin Center interface. On the left, the navigation menu shows 'Active users' selected (marked with a red circle 1). The main area is titled 'Active users' and lists several users: Reza Domiali, Reza PPU, Reza Rad, Reza Rad (selected and highlighted with a red box 2), and reza\_raad. To the right, a sidebar titled 'Manage admin roles' shows a list of roles with 'Reza Rad selected'. The 'Power BI Administrator' role is checked (marked with a red circle 3) and highlighted with a red box. Other roles listed include 'Kacala Administrator', 'Knowledge Administrator', 'Knowledge Manager', 'Network Administrator', 'Office Apps Administrator', 'Power Platform Administrator', 'Search Administrator', 'Search Editor', 'SharePoint Administrator', 'Styler for Business Administrator', and 'Teams Administrator'.

Role	Description
Kacala Administrator	
Knowledge Administrator	
Knowledge Manager	
Network Administrator	
Office Apps Administrator	
<b>Power BI Administrator</b>	(Selected)
Power Platform Administrator	
Search Administrator	
Search Editor	
SharePoint Administrator	
Styler for Business Administrator	
Teams Administrator	

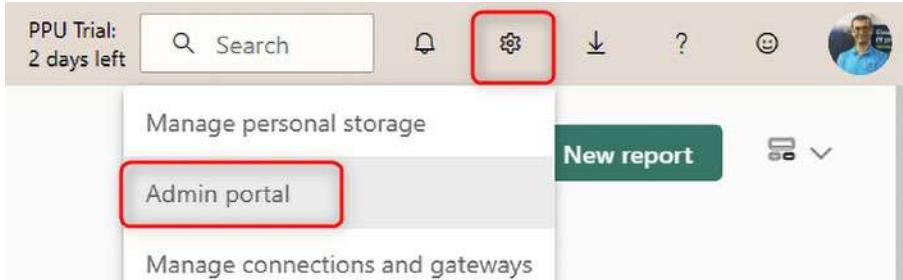
Figure 24-1. Assigning the Power BI administrator role

Next, click Manage Roles. Under Admin Center Access, expand the Show All by category. Then, from the list, select Power BI Administrator.

You can also use PowerShell scripts to assign the Power BI administrator role to a user.

## Tenant Settings

The Power BI administrator can access Tenant Settings from the Power BI Service. To do that, you need to click the Setting icon and choose Admin Portal, as shown in Figure 24-2.



**Figure 24-2.** Navigating to Power BI Admin Portal

From the Admin Portal, you can click Tenant Settings, as depicted in Figure 24-3 (note that you can access this page only if you are a Power BI administrator). If you are not a Power BI administrator, under Admin Portal, you will perhaps just see the capacity settings.

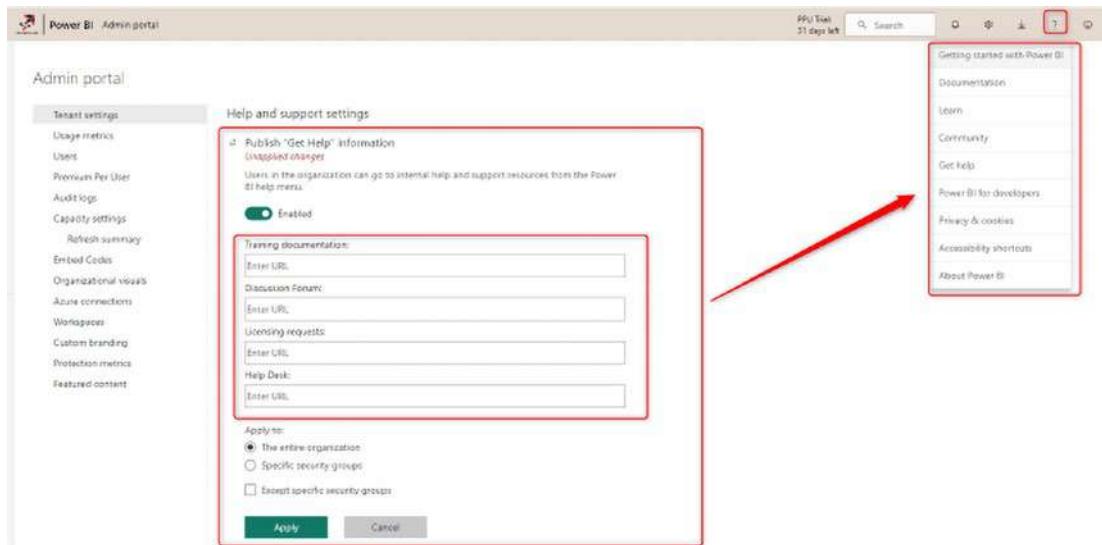
The screenshot shows the Power BI Admin portal interface. At the top, there's a navigation bar with icons for Home, Admin, and Help. Below it, the main header says "Power BI Admin portal". On the left, there's a sidebar with various icons and a list of settings: Usage metrics, Users, Premium Per User, Audit logs, Capacity settings (with "Refresh summary" under it), Embed Codes, Organizational visuals, Azure connections, Workspaces, Custom branding, Protection metrics, and Featured content. The "Tenant settings" option is highlighted with a red box. To the right, under "Help and support settings", there are four items: "Publish 'Get Help' information" (disabled), "Receive email notifications for service outages or incidents" (disabled), "Allow users to try Power BI paid features" (enabled), and "Show a custom message before publishing reports" (disabled). Below that, under "Workspace settings", there are two items: "Create workspaces (new workspace experience)" (enabled) and "Use datasets across workspaces" (enabled).

**Figure 24-3.** Admin Portal of Power BI for a Power BI Administrator

Tenant Settings are categorized into groups. Since the first public release of Power BI until now, many features have been added to the Power BI, and with new additions come new configurations. Over time, the Tenant Settings configuration list has gradually expanded to a long list of options. The following sections explain some of the most important aspects of these configurations.

## Help and Support Settings

You can set links to be used for help and support throughout your organization. When a user in your tenant logs in to the Power BI Service and click the Help icon, they can be redirected to a specific URL or page you want them to see for documentation or FAQs. To set these up, use the Get Help information setup, as shown in Figure 24-4.



**Figure 24-4.** Setting up the Get Help links in the Power BI Service

The settings you apply here are then be used in the Get Help links that are available in the Power BI Service. You can set the settings for the entire organization or for only a specific group of people (this type of categorization exists in most of the configurations of the Tenant Settings).

As shown in Figure 24-5, there are other options that mostly affect your support or development team by allowing them to try some of the new paid features or get a message when they want to publish a report.

## Help and support settings

- ▷ Publish "Get Help" information  
*Disabled for the entire organization*
- ▷ Receive email notifications for service outages or incidents  
*Disabled for the entire organization*
- ▷ Allow users to try Power BI paid features  
*Enabled for the entire organization*
- ▷ Show a custom message before publishing reports  
*Disabled for the entire organization*

**Figure 24-5.** Additional help and support settings on the Power BI Tenant Settings page

## Workspace Settings

Workspaces are the heart of collaboration in the Power BI Service environment. It is better to control who can create the workspace (see Figure 24-6). This is one of the options that I recommend only be enabled for the Power BI development group in your organization. However, using datasets across workspaces is a very good option for the entire organization. This will enable users to use a shared dataset and avoid creating silos of Power BI objects.

### Workspace settings

- ▷ Create workspaces (new workspace experience)  
*Enabled for the entire organization*
- ▷ Use datasets across workspaces  
*Enabled for the entire organization*
- ▷ Block scheduled upgrade of empty workspaces  
*Enabled for the entire organization*
- ▷ Block notifications for scheduled workspace upgrades  
*Disabled for the entire organization*

**Figure 24-6.** Workspace settings on the Power BI Tenant Settings page

## Information Protection

You can enable sensitivity labels in Power BI. This feature uses Microsoft Purview Information Protection, which has a separate licensing structure. Figure 24-7 offers a preview of the Information Protection options. It requires some prerequisite setup,<sup>1</sup> and then you can use the option provided in this section.

<sup>1</sup>[learn.microsoft.com/en-us/power-bi/enterprise/service-security-enable-data-sensitivity-labels](https://learn.microsoft.com/en-us/power-bi/enterprise/service-security-enable-data-sensitivity-labels)

## Information protection

- ▷ Allow users to apply sensitivity labels for content  
*Disabled for the entire organization*
- ▷ Require users to apply sensitivity labels to Power BI content (preview)  
*Disabled for the entire organization*
- ▷ Apply sensitivity labels from data sources to their data in Power BI  
*Disabled for the entire organization*
- ▷ Automatically apply sensitivity labels to downstream content  
*Disabled for the entire organization*
- ▷ Allow workspace admins to override automatically applied sensitivity labels  
*Disabled for the entire organization*
- ▷ Restrict content with protected labels from being shared via link with everyone in your organization  
*Disabled for the entire organization*

**Figure 24-7.** Information protection settings on the Power BI Tenant Settings page

## Export and Sharing Settings

This is a large section of settings; it includes configurations regarding sharing across your tenant (except sharing with apps or other methods), the ability to export or copy and paste, and more.

## External Users

There are a few options to allow external users to access Power BI, and even to enable them to manage and edit your Power BI content. Figure 24-8 offers a look at these options. Set these as you need.

### Export and sharing settings

- ▷ Allow Azure Active Directory guest users to access Power BI  
*Enabled for the entire organization*
- ▷ Invite external users to your organization  
*Enabled for the entire organization*
- ▷ Allow Azure Active Directory guest users to edit and manage content in the organization  
*Disabled for the entire organization*
- ▷ Show Azure Active Directory guests in lists of suggested people  
*Enabled for the entire organization*
- ▷ Publish to web ⓘ

**Figure 24-8.** External user configurations on the Power BI Tenant Settings page

## Publish to Web

This configuration, shown in Figure 24-9, is often a good enough reason to access the Tenant Settings page. By default, everyone in your organization can publish their reports on the web! If you don't know about Publish to Web, it's covered in later chapter in this book. Publish to Web makes the Power BI content publicly available. This is a very dangerous option to always have on. You must make sure that you either disable this option or allow a very restricted group of people to use it.

### ↳ Publish to web ⓘ

*Enabled for the entire organization*

People in your org can publish public reports on the web. Publicly published reports don't require authentication to view them.

Go to [Embed Codes](#) in the admin portal to review and manage public embed codes. If any of the codes contain private or confidential content remove them.

Review embed codes regularly to make sure no confidential information is live on the web. [Learn more about Publish to web](#)

 Enabled

Choose how embed codes work

- Only allow existing codes
- Allow existing and new codes

Apply to:

- The entire organization
- Specific security groups
- Except specific security groups

 Apply

 Cancel

**Figure 24-9.** Publish to Web settings for the Power BI tenant

## Export, Copy, Print, and Integrate Data

There are options for enabling and disabling exporting functionalities such as Export to Excel or to CSV, as well as the ability to download reports, create a live connection to a dataset, export reports as different formats and print.

Some of these configuration options are shown in Figure 24-10, but there are more in the list, such as allowing DirectQuery to connect to Power BI dataset to create a chained dataset, to enable the integration with Microsoft Teams and PowerPoint.

- ▷ Copy and paste visuals  
*Enabled for the entire organization*
- ▷ Export to Excel  
*Enabled for the entire organization*
- ▷ Export to .csv  
*Enabled for the entire organization*
- ▷ Download reports  
*Enabled for the entire organization*
- ▷ Allow live connections  
*Enabled for the entire organization*
- ▷ Export reports as PowerPoint presentations or PDF documents  
*Enabled for the entire organization*
- ▷ Export reports as MHTML documents  
*Enabled for the entire organization*
- ▷ Export reports as Word documents  
*Enabled for the entire organization*
- ▷ Export reports as XML documents  
*Enabled for the entire organization*
- ▷ Export reports as image files  
*Disabled for the entire organization*
- ▷ Print dashboards and reports  
*Enabled for the entire organization*

**Figure 24-10.** Export data settings on the Power BI Tenant Settings page

## Certifications

Your Power BI contents can be marked as certified, which shows that they have been through a process of testing and quality checks. It is important that only a certain group within the organization be able to certify Power BI objects. I explain this later in the chapter that covers certification and endorsements.

In addition to this setting, there are some important discovery options, shown in Figure 24-11, including Make Promoted Content and Certified Content Discoverable.

### Discovery settings

- ▷ Make promoted content discoverable  
*Enabled for the entire organization*
- ▷ Make certified content discoverable  
*Enabled for the entire organization*
- ▷ Discover content  
*Enabled for the entire organization*

**Figure 24-11.** Certification Discovery settings in Power BI

## Content Pack and App Settings

Content Pack and Power BI Apps are ways to share Power BI reports.<sup>2</sup> The Content Pack method is almost obsolete, and the Power BI App method is a prevalent way of sharing. In this section, you can configure this method of sharing, as shown in Figure 24-12.

### Content pack and app settings

- ▷ Publish content packs and apps to the entire organization  
*Enabled for the entire organization*
- ▷ Create template organizational content packs and apps  
*Enabled for the entire organization*
- ▷ Push apps to end users  
*Enabled for the entire organization*

**Figure 24-12.** App settings in the Power BI Tenant Settings page

---

<sup>2</sup>To learn more about Content Pack, visit [radacad.com/content-pack-sharing-self-service-and-governance-together](http://radacad.com/content-pack-sharing-self-service-and-governance-together)

## Push Apps to End Users

Shown in Figure 24-13, this setting is one of the very new options added to the Tenant Settings. Without this option selected, Power BI apps won't automatically be pushed to the end user's apps section. When you turn this option on, any apps created will be automatically pushed to your users. You don't need to get the app from each user's profile individually. This is a very good option to be turned on. However, you might want to limit it to a specific group in the organization.

### 4 Push apps to end users

*Enabled for the entire organization*

Users can share apps directly with end users without requiring installation from AppSource.



Enabled

Apply to:

The entire organization

Specific security groups

Except specific security groups

Apply

Cancel

**Figure 24-13.** Push apps to end users

## Integration Settings

Configurations in the Integration Settings area, featured in Figure 24-14, are related to integrating Power BI with other technologies, such as ArcGIS, XMLA endpoint, Azure Maps, Snowflake, Redshift, Google BigQuery, and so on.

## Integration settings

- ▷ Allow XMLA endpoints and Analyze in Excel with on-premises datasets  
*Enabled for the entire organization*
- ▷ Dataset Execute Queries REST API  
*Enabled for the entire organization*
- ▷ Use ArcGIS Maps for Power BI  
*Enabled for the entire organization*
- ▷ Use global search for Power BI  
*Enabled for the entire organization*
- ▷ Use Azure Maps visual  
*Disabled for the entire organization*
- ▷ Map and filled map visuals  
*Enabled for the entire organization*
- ▷ Integration with SharePoint and Microsoft Lists  
*Enabled for the entire organization*
- ▷ Dremio SSO  
*Disabled for the entire organization*
- ▷ Snowflake SSO  
*Disabled for the entire organization*
- ▷ Redshift SSO  
*Disabled for the entire organization*
- ▷ Google BigQuery SSO  
*Disabled for the entire organization*
- ▷ Azure AD Single Sign-On (SSO) for Gateway  
*Disabled for the entire organization*
- ▷ Power Platform Solutions Integration (Preview)  
*Enabled for the entire organization*

**Figure 24-14.** Integration settings on the Power BI Tenant Settings page

Some of these options are very helpful and essential to use, such as the XMLA endpoint. This gives you the ability to connect to the Power BI dataset using third-party tools and build a better data model in Power BI. Some of them are specific to visuals, such as Map and Filled Map. Because some of the map data is processed in the servers, it requires the tenant admin to agree with that term before allowing it. The same is true for sign-on abilities with services such as Snowflake, Redshift, and Google BigQuery.

## Custom Visual Settings

Custom visuals, as shown in Figure 24-15, are great add-ins. Custom visuals are built by third parties. Not all custom visuals are supported well. Also, some of the custom visuals are paid. As the Power BI administrator, you may want to restrict your tenant's usage of custom visuals by changing this option. Or you may only want to allow certified custom visuals (which have been through a validation process).

You can also control the usage of R and Python visuals. If you enable this function, note that using R and Python visual also requires a Power BI personal gateway. See Figure 24-15.

### Power BI visuals

- ▷ Allow visuals created using the Power BI SDK  
*Enabled for the entire organization*
  - ▷ Add and use certified visuals only (block uncertified)  
*Disabled for the entire organization*
  - ▷ Allow downloads from custom visuals  
*Disabled for the entire organization*
- 

### R and Python visuals settings

- ▷ Interact with and share R and Python visuals  
*Enabled for the entire organization*

**Figure 24-15.** Custom visual settings in Power BI

There are many visuals for R, and Leila Etaati has also written a book about R and Power BI.<sup>3</sup>

---

<sup>3</sup>[radacad.com/online-book-analytics-with-power-bi-and-r](http://radacad.com/online-book-analytics-with-power-bi-and-r)

## Audit and Usage Settings

The Power BI usage metrics work with audit logs and give you a detailed analysis of the usage and consumption of your Power BI content. Options in this section are related to the audit log and usage metrics. You can enable the usage for content creators (which is recommended to be on), and enable per-user data in the usage metrics (which helps to identify user activities).

## Create Audit Logs for Internal Activity Auditing and Compliance

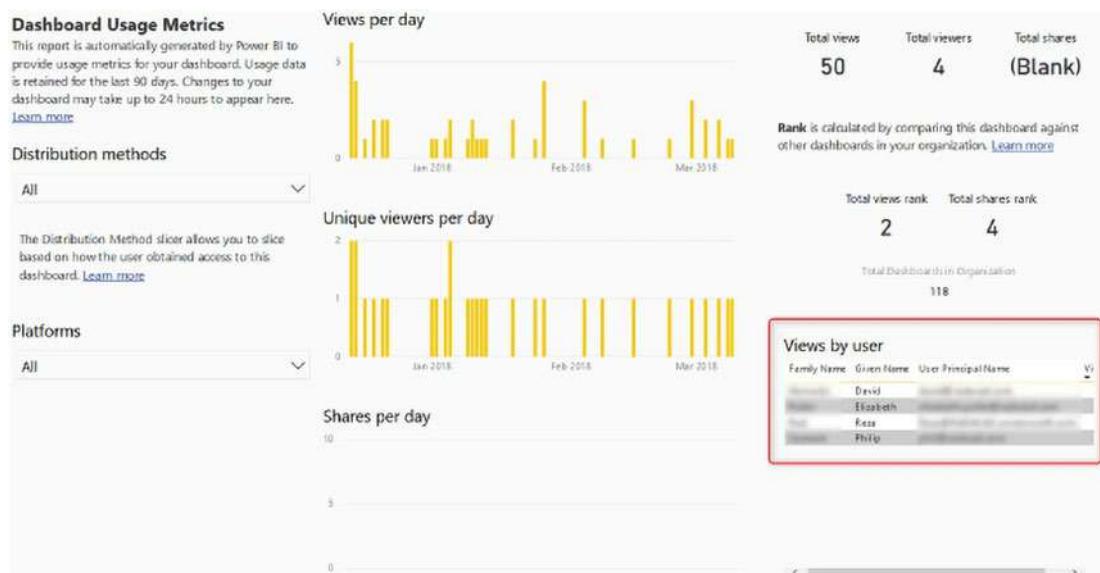
The audit logs are created by this option, and users can use it for monitoring and data analysis.

# Usage Metrics for Content Creators

Content creators are people who create Power BI reports, dashboards, and datasets. By default, content creators have access to the usage metrics report monitoring the usage of their content. You can change the option if you want to turn off this feature.

# Per-User Data in Usage Metrics for Content Creators

Usage metrics data can also include the per-user individual metrics. Per-user data is useful, especially if you want to monitor usage by users to see if they are using the content shared with them. Figure 24-16 shows an example of a usage metrics report with per-user data.



**Figure 24-16.** Usage metrics with user data

## Developer Settings

As a developer, you can leverage Power BI Embedded API and embed reports or dashboards into an application. This is a powerful feature, and it can be turned on or off from here. To learn more about Power BI Embedded, read the chapter related to that in this book. See Figure 24-17 for a look at the developer and admin API settings.

### Developer settings

- ▷ Embed content in apps  
*Enabled for the entire organization*
  - ▷ Allow service principals to use Power BI APIs  
*Disabled for the entire organization*
  - ▷ Allow service principals to create and use profiles  
*Disabled for the entire organization*
  - ▷ Block ResourceKey Authentication  
*Disabled for the entire organization*
- 

### Admin API settings

- ▷ Allow service principals to use read-only admin APIs  
*Disabled for the entire organization*
- ▷ Enhance admin APIs responses with detailed metadata  
*Disabled for the entire organization*
- ▷ Enhance admin APIs responses with DAX and mashup expressions  
*Disabled for the entire organization*

**Figure 24-17.** Power BI developer and API settings on the Tenant Settings page

There are also other developer configurations that you can set to use APIs. Power BI has a good list of admin APIs, which helps third-party applications (or even PowerShell scripts) interact with Power BI objects in the service and automate part of the Power BI administration process.

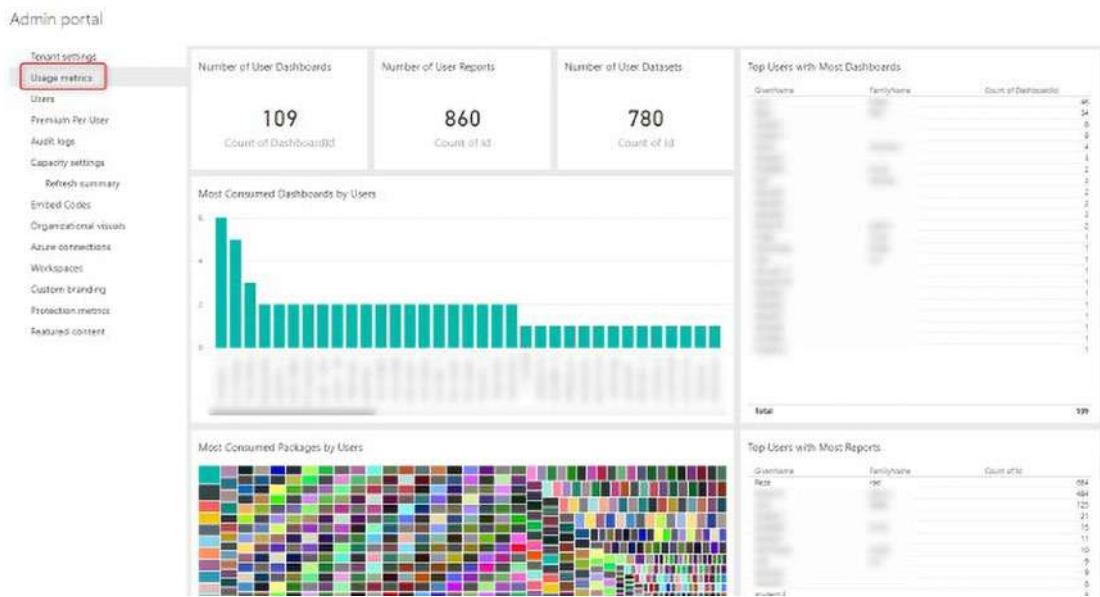
These options can enhance the way that Power BI is used throughout the organization.

## Dataflow, Template Apps, and Datamarts

There are many configurations that enable or disable the use and creation of objects such as dataflows, datamarts, metrics, template apps, and many of the new and preview features in Power BI.

## Usage Metrics

The Admin Portal also has other sections that are helpful for a Power BI administrator. One monitoring part of the Admin Portal is the usage metrics of all Power BI content across the tenant. This can be accessed from the Admin Portal by clicking Usage Metrics, as shown in Figure 24-18.



**Figure 24-18.** Usage metrics report for the Power BI administrator

The usage metrics here are different from the usage metrics for each dashboard and report. The usage metrics for the Power BI administrator provide an overall monitoring view of which users use most of the content, and which dashboards or reports have been used most often. How many dashboards, reports, or data sets exist in the tenant? This will reveal information about workspaces and more. The usage metrics reporting however, is not flexible and you cannot customize it.

## Embed Codes

If you enabled the Publish to Web feature of Power BI in the tenant settings, you need to monitor the content that users publish using this option. The Embed Codes section of the Admin Portal, featured in Figure 24-19, tells you which reports are published to the web by which user. You can see the report or delete the embed code (deleting the embed code will not delete the report, it will only unpublish it from the web link).

**Admin portal**

**Embed Codes**

View embed codes that have been created by your organization. To change users' ability to use publish-to-web, see [Tenant settings](#).

C Refresh E Export View on web Delete

Report name	Workspace name	Published by	Status
My workspace	Lelia Etter	Active	
View on web	Lelia Etter	Active	
Delete	Lelia Etter	Active	
My workspace	Rosa Rad	Active	
My workspace	Lelia Etter	Active	
My workspace	Lelia Etter	Active	
My workspace	Lelia Etter	Active	

**Figure 24-19.** Seeing all the embed codes across the Power BI tenant

## Organization Visuals

You can add a Power BI visual designed by your organization, as shown in Figure 24-20. You need the \*.pbviz file to upload here.

**Admin portal**

**Organization visuals**

Add new custom visuals for your organization or manage them. [Learn more](#)

NAME	PUBLISHED BY
Add a custom visual	

**Figure 24-20.** Adding a custom Power BI visual

Then you can enter the details of the visual, as shown in Figure 24-21.

## Edit custom visual

Last updated: Mar 13, 2018

Choose a .pbviz file \* \*Required

 Browse

Name your custom visual \*

Icon \*



Upload an image or company logo

This icon will be seen on the custom visual store.  
Image max size should be 65 KB, 1:1 aspect ratio,  
JPG or PNG format.

[Use default](#)

Description

This is a R custom visual for Power BI users. This chart able to shows at the same time 4 to 5 variables.

ApplyCancel

**Figure 24-21.** Finalizing the custom visuals

These visuals will then be available in the Power BI Desktop from the My Organization tab, as shown in Figure 24-22.

## Power BI Visuals

MARKETPLACE | MY ORGANIZATION



**Figure 24-22.** Custom visuals can be found on the My Organization tab

## Workspaces

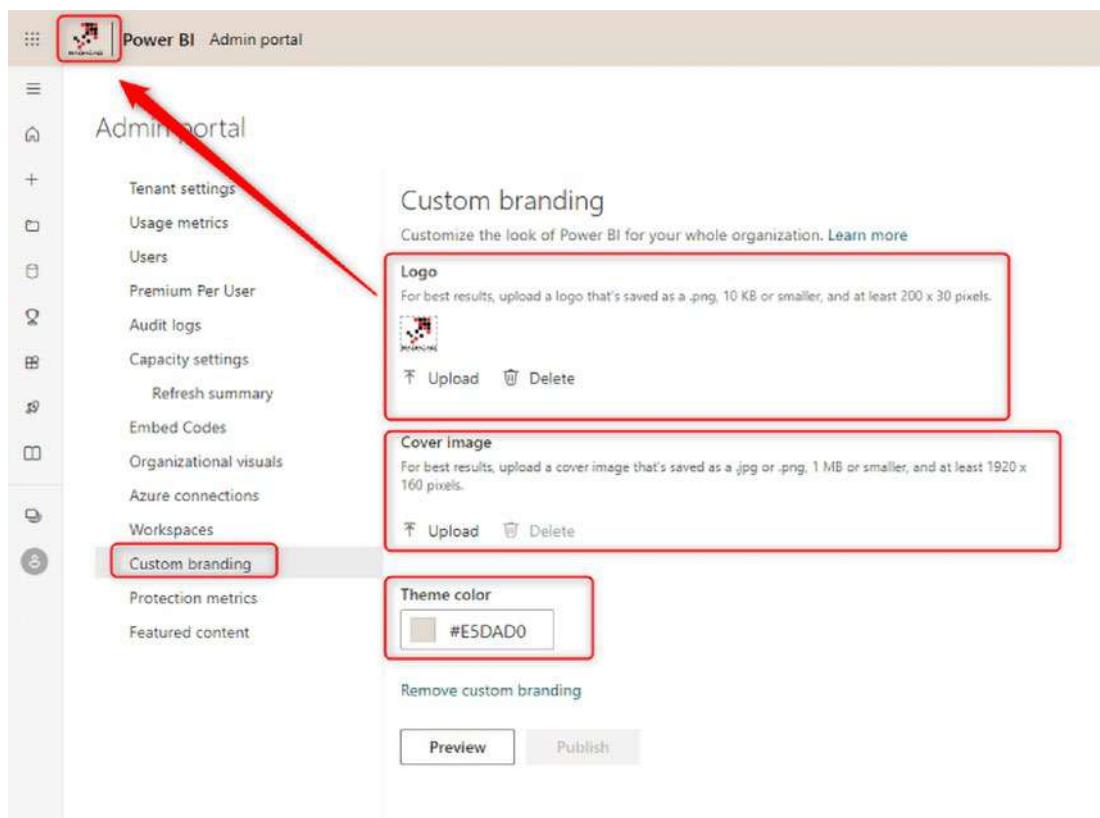
Figure 24-23 shows that, as the tenant admin, you can see the full list of Power BI workspaces if they are defined under a Premium capacity.

Name	Description	Type	State	Capacity name	Capacity SKU Tier	Upgrade status
Blank workspace		Workspace	Active			
BIAD WS		Workspace	Active			
Diversity		Workspace	Active			
the new v2 sample		Workspace	Active			
data flow example Sydney		Workspace	Active			
workspace v2 df		Workspace	Active			

**Figure 24-23.** The full list of Power BI workspaces in the Power BI Admin Portal

## Custom Branding

If you want to add your organization's logo and colors to the Power BI Service, you can easily do so, as shown in Figure 24-24.



**Figure 24-24.** Adding custom branding the Power BI Service

## Summary

Power BI administrators have access to specific parts of the Admin Portal, such as Tenant Settings, Manage Embed Codes, Organization Visuals, Usage Metrics, and more. The Tenant Settings area has many configurations and you need to be careful when using these options. For example, Publish to Web and the integration settings are important options to consider. The Admin Portal and Tenant Settings are updated in each version of the Power BI Service. There were some options that I did not explain in this chapter, as their use is either obvious or is related to a different topic (for example, Azure connections are related to creating external dataflow in Power BI that stores the data in your own Azure Data Lake storage subscription).

## CHAPTER 25



# PowerShell Cmdlets for Power BI

Power BI has a set of PowerShell Cmdlets that help automate part of the operations with Power BI. However, PowerShell is not a familiar technology. In the world of Power BI, we are used to working with graphical options and settings provided in the tools and the service. However, using commands provided for Power BI in a command/scripting tool such as PowerShell can be an excellent asset to a Power BI administrator, architect, and developer. In this chapter, you learn about the PowerShell Cmdlets for Power BI.

## PowerShell

*PowerShell is a task automation and configuration management program from Microsoft, consisting of a command-line shell and the associated scripting language.<sup>1</sup>*

PowerShell has become a common tool for administrators over the past few years. As an administrator, there are many libraries that you can access using PowerShell modules. PowerShell's scripting and command line experience are not as complicated as learning a programming language (such as C#.NET). This results in an easy-to-use tool that provides great power in configuration and task automation. Learning PowerShell is a topic that is outside of this chapter.<sup>2</sup>

## PowerShell Cmdlets for Power BI

PowerShell includes modules that provide access to certain functionalities. Consider these modules like libraries. Each module gives you access to certain objects and configurations in the Power BI Service.

For example, one module gives you access to the list of workspaces in the Power BI Service, and another helps you with the reports inside the workspace. You can use a module to capture activities through the Power BI tenant in your organization. You can use a combination of these modules to extract a report from one workspace and then export it to another.

Table 25-1 shows the list of the modules.<sup>3</sup>

<sup>1</sup>[en.wikipedia.org/wiki/PowerShell](https://en.wikipedia.org/wiki/PowerShell)

<sup>2</sup>The PowerShell documentation is a good place to start: [learn.microsoft.com/en-us/powershell/](https://learn.microsoft.com/en-us/powershell/)

<sup>3</sup>[learn.microsoft.com/en-us/powershell/power-bi/overview?view=powerbi-ps](https://learn.microsoft.com/en-us/powershell/power-bi/overview?view=powerbi-ps)

**Table 25-1.** PowerShell Modules for Power BI

Description	Module Name
Rollup module for Power BI Cmdlets	MicrosoftPowerBIMgmt
Admin module for Power BI Cmdlets	MicrosoftPowerBIMgmt.Admin
Capacities module for Power BI Cmdlets	MicrosoftPowerBIMgmt.Capacities
Data module for Power BI Cmdlets	MicrosoftPowerBIMgmt.Data
Profile module for Power BI Cmdlets	MicrosoftPowerBIMgmt.Profile
Reports module for Power BI	MicrosoftPowerBIMgmt.Reports
Workspaces module for Power BI	MicrosoftPowerBIMgmt.Workspaces

If you install the first module, it includes all the other modules. You can also choose to install modules as you need. To use the modules in Table 25-1, you must meet the following requirements:

- Windows PowerShell v3.0 and up with .NET 4.7.1 or above
- PowerShell Core (v6) and up on any OS platform supported by PowerShell Core

## Getting Started

Let's go through a sample use of these modules. To start, open Windows PowerShell ISE (this is a more user-friendly version of Windows PowerShell). If you don't have this application, use the normal Windows PowerShell. It is better if you start this application as an administrator, as shown in Figure 25-1.

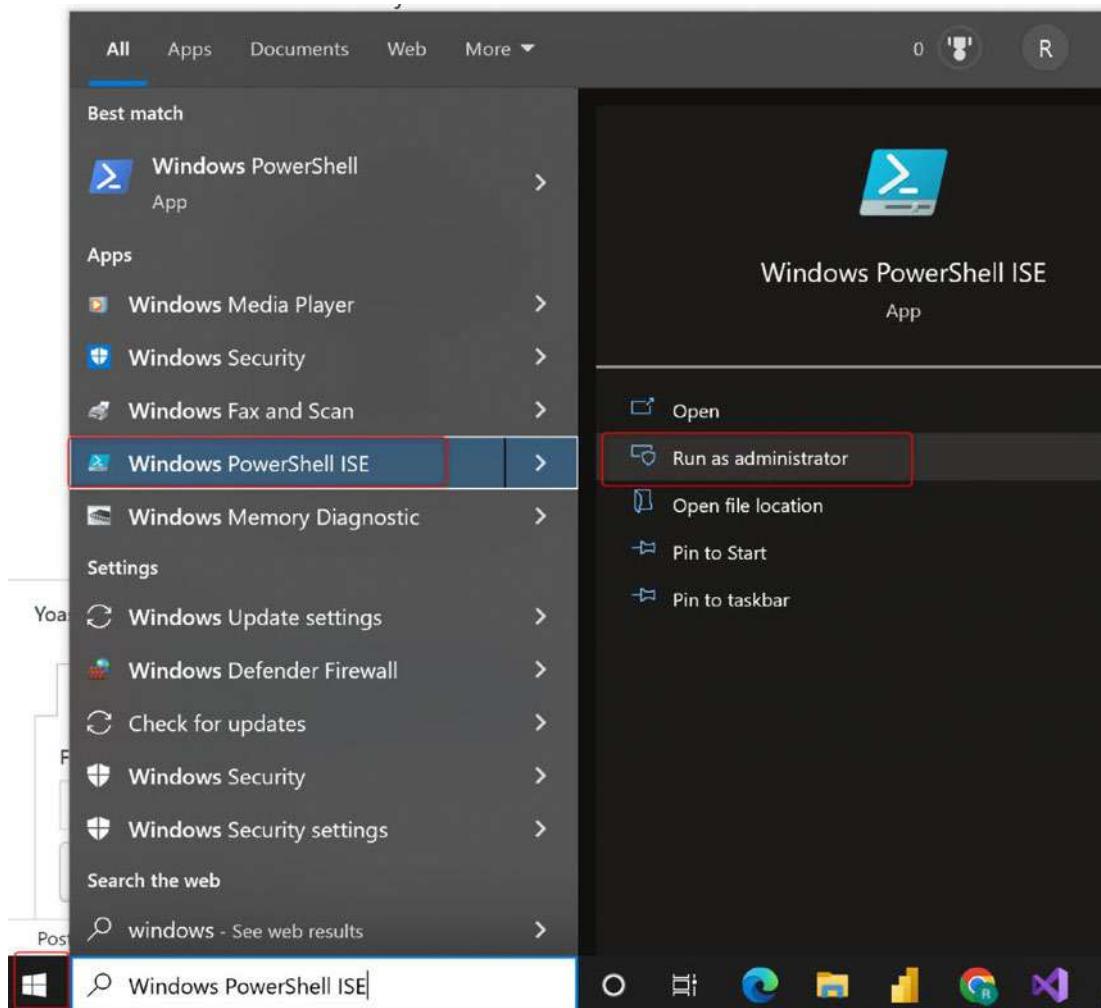
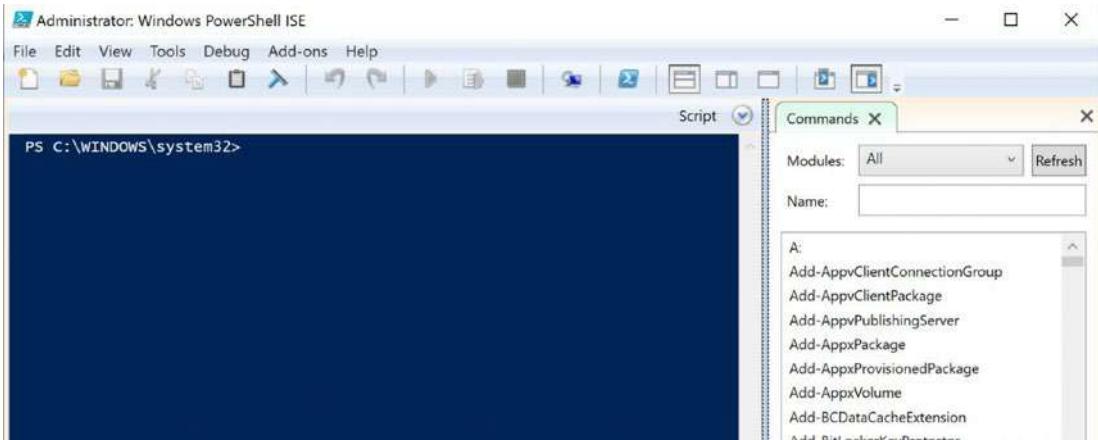


Figure 25-1. Opening Windows PowerShell ISE as administrator

This will open the application (see Figure 25-2), in which you can either type the commands (also called Cmdlets) or use the list on the right side and select from the Cmdlets. (Note that this list is only available in ISE, the basic Windows PowerShell doesn't come with it.)

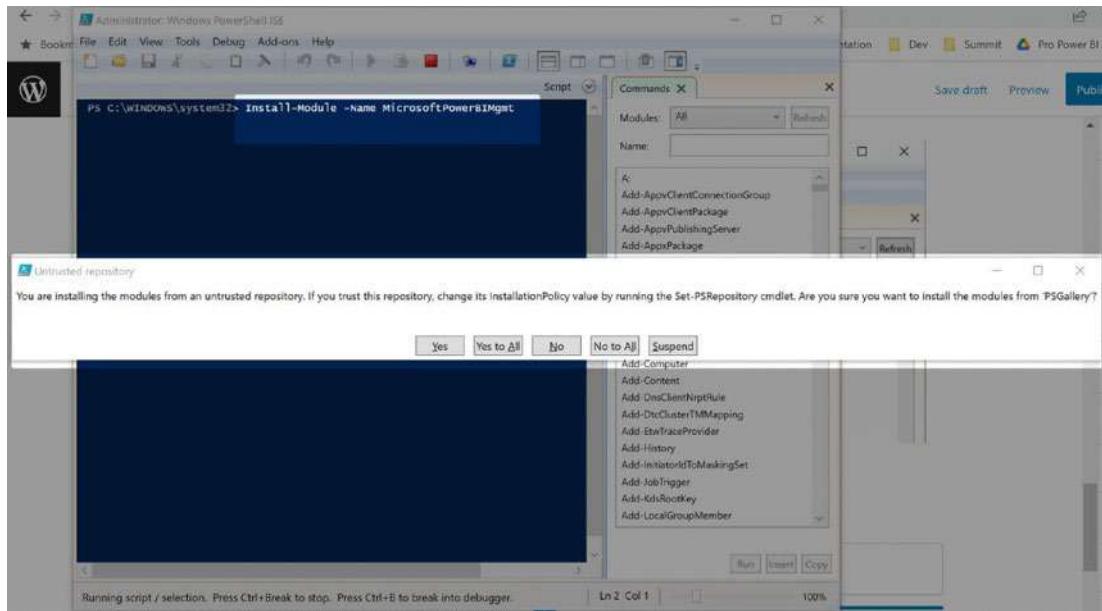


**Figure 25-2.** Windows PowerShell ISE

To get the modules, you can use the `Install-Module` Cmdlet as follows:

```
Install-Module -Name MicrosoftPowerBIMgmt
```

You can then confirm the installation of the module, as shown in Figure 25-3.



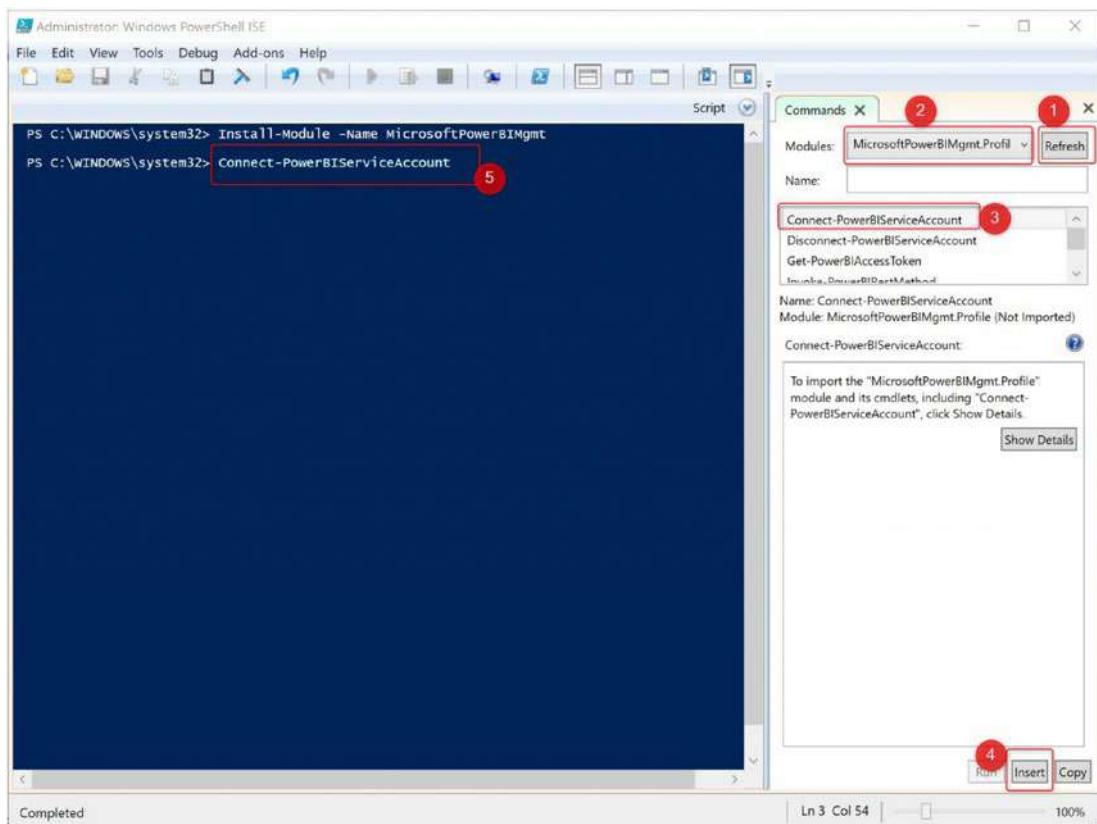
**Figure 25-3.** Installing Power BI modules for PowerShell

This step will install the module. If you already have the module installed and want to update it to the most recent version, use the `Update-Module` Cmdlets with the same parameters.

To use the Power BI Cmdlets, you first need to log in to Power BI using the Connect-PowerBIServiceAccount Cmdlet as follows:

#### Connect-PowerBIServiceAccount

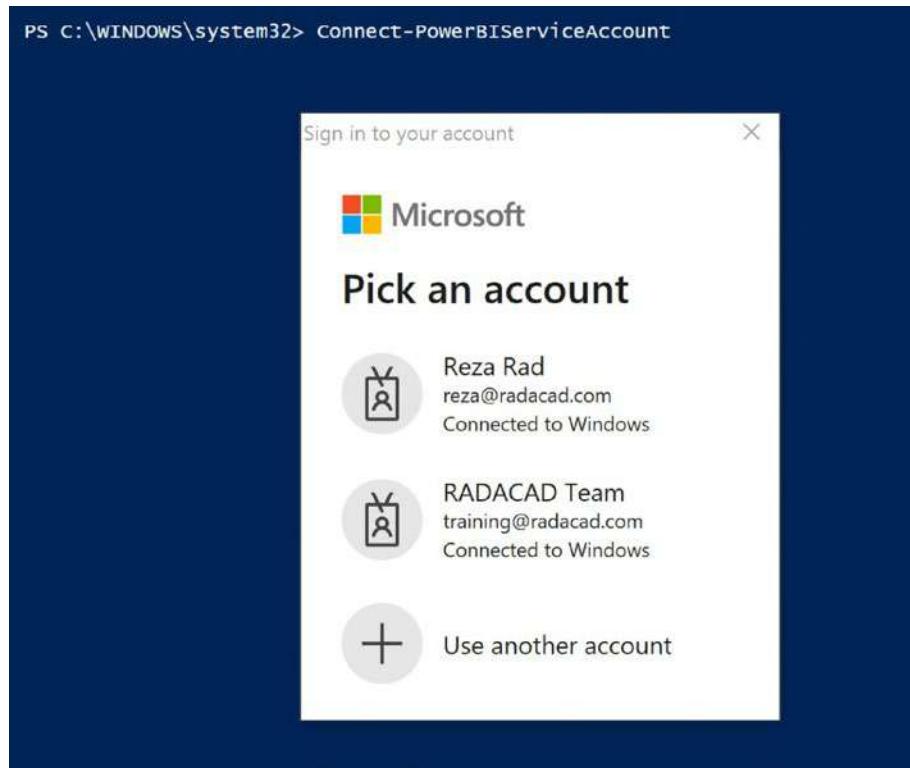
Or, if you prefer to use the list of modules in the PowerShell ISE, refresh the modules list, select the Power BI Profile module, and then select the Cmdlet. You can either run the command from here or insert it, which will insert in the command line. Figure 25-4 outlines these steps.



**Figure 25-4.** Using the list of Cmdlets in PowerShell ISE

If you choose a Cmdlet that requires parameters, the UI will also have places to enter those details.

Use the sign-in box to use your account for the Power BI (see Figure 25-5). If your account is a Power BI administrator, it might give you tenant results in your command execution. Otherwise, you can still use your non-admin Power BI account to use some of the commands.



**Figure 25-5.** Log in using the Power BI account

If the login is successful, you will see your tenant ID and username, as shown in Figure 25-6.

```
PS C:\WINDOWS\system32> Connect-PowerBIServiceAccount

Environment : Public
TenantId    : ccc80f8d6-3c35-4da4-b0f8-cbdeadb00467e
UserName    : reza@radacad.com

PS C:\WINDOWS\system32>
```

**Figure 25-6.** The output of the Connect-PowerBIServiceAccount Cmdlet

Now you can test some of the Cmdlets and see the results. For example, the following Cmdlet gives you a list of all the workspaces (these are the workspaces that you have access to, not all the workspaces in the organization):

```
Get-PowerBIWorkspace -All
```

The output is a list of workspaces, their names with IDs, and other information, as shown in Figure 25-7.

```

Id : 63678ecc-f862-4fb4-a463-a1e72badb8fa
Name : Community
Type : Workspace
IsReadOnly : False
IsOrphaned : False
IsOnDedicatedCapacity : False
CapacityId :

Id : d8aeefb1-1052-476b-9d0e-dcddde69b623
Name : PW
Type : Workspace
IsReadOnly : False
IsOrphaned : False
IsOnDedicatedCapacity : True
CapacityId : 6C533637-8A43-4B09-88CF-67D689A04CBB

Id : d3847ea9-b624-48c1-b025-37b1bff5ede1
Name : PW2
Type : Workspace
IsReadOnly : False
IsOrphaned : False
IsOnDedicatedCapacity : True
CapacityId : 6C533637-8A43-4B09-88CF-67D689A04CBB

Id : 48e4d0ce-84e1-472c-a306-a23abe4cf5ad
Name : PPU WS
Type : Workspace
IsReadOnly : False
IsOrphaned : False
IsOnDedicatedCapacity : True
CapacityId : 6C533637-8A43-4B09-88CF-67D689A04CBB

Id : 36f1f927-13c8-4c1c-a4aa-d0b2bdd7ecca
Name : WS DIS
Type : Workspace
IsReadOnly : False
IsOrphaned : False
IsOnDedicatedCapacity : True
CapacityId : 6C533637-8A43-4B09-88CF-67D689A04CBB

PS C:\WINDOWS\system32>

```

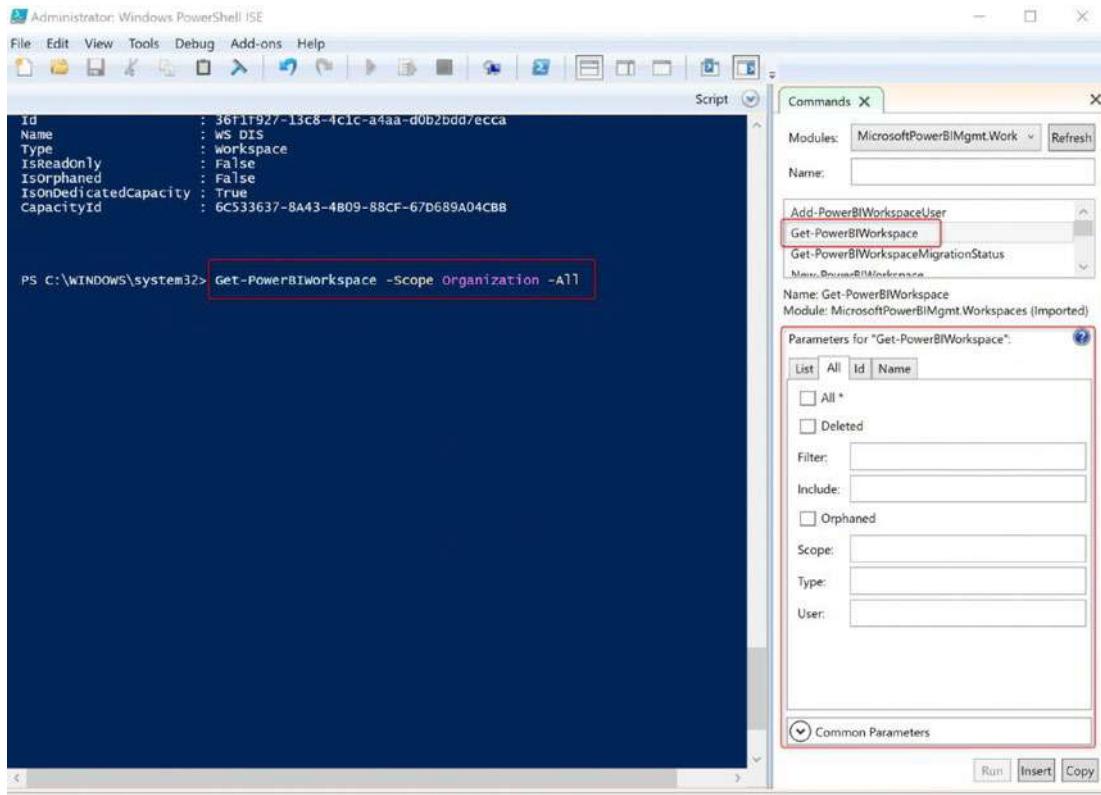
**Figure 25-7.** Getting a list of all workspaces for a Power BI user

## Example: Export a List of All Workspaces in the Organization as a CSV File

If your account is a Power BI Administrator account, you can use a parameter for scope and set that to Organization. This will give you all the workspaces in your tenant (even if you don't have access to those workspaces and other users in your organization created them):

```
Get-PowerBIWorkspace -Scope Organization -All
```

As mentioned, you can also select this Cmdlet in the list on the right side and see all the other parameters (see Figure 25-8).



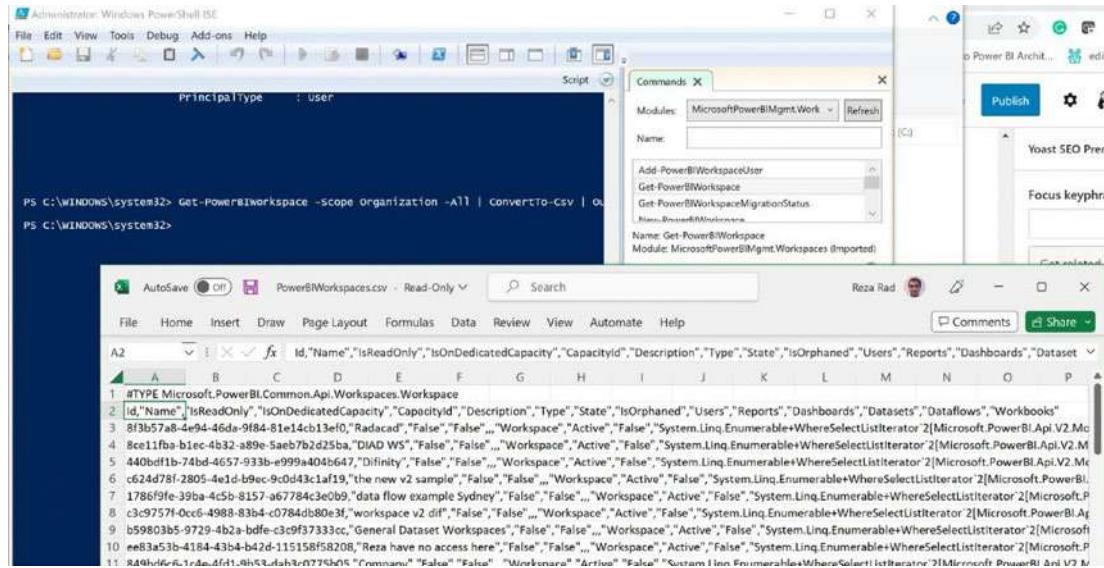
**Figure 25-8.** Getting all the Power BI workspaces in the organization

This command might take some time to run if you have many workspaces in your organization.

Now comes the power of PowerShell itself. This output is great, but what if you could export the output as a file, which you could use for other applications and purposes? Many modules and Cmdlets in PowerShell can help you with general tasks like that. The following example uses the `ConvertTo-Csv` Cmdlet with the `Out-File` parameter, which specifies the path of the CSV file.

```
Get-PowerBIWorkspace -Scope Organization -All | ConvertTo-Csv | Out-File c:\PowerBIWorkspaces.csv
```

This command gives you a CSV file export of all workspace details, as demonstrated in Figure 25-9.



**Figure 25-9.** Exporting all the Power BI workspaces to a CSV file

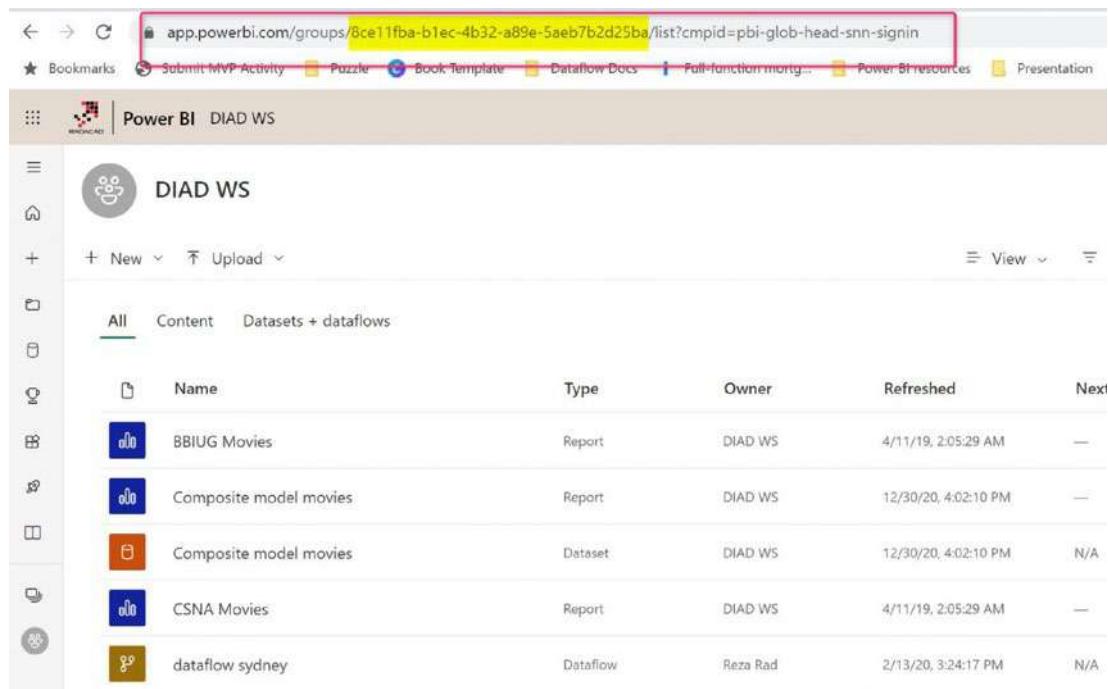
Isn't this helpful? Now imagine if you could schedule these simple few lines of script to run every night from a server and store a list of workspaces. Now you have an application that stores the history of workspace changes throughout time in your organization. It is needless to say that you can generate the filename using the date and timestamp. So, if you have a history of all workspaces throughout the entire organization at any point in time, you can schedule the script. The next example shows how to schedule a PowerShell script using Task Scheduler.

## Example: Deploy a Power BI Report from One Workspace to Another

PowerShell Cmdlets for Power BI can also create deployment pipeline solutions. This, then, can be a good replacement for the Deployment Pipeline feature, which is a Premium function in Power BI (of course, building something as comprehensive as that might take a while using the PowerShell Cmdlets).

I am going to show you part of this as an example. Let's say you want to export a single report from one workspace to another. You can use a combination of Cmdlets for this purpose, including Get-PowerBIReport and Export-PowerBIReport.

You need to know the workspace ID to get a Power BI report from a workspace. The workspace ID is a unique identifier. As Figure 25-10 shows, this ID can be recognized in the page URL when you are inside the workspace.



The screenshot shows a Power BI workspace named 'DIAD WS'. The URL in the browser bar is [app.powerbi.com/groups/bce11fba-b1ec-4b32-a89e-5aeb7b2d25ba/list?cmplid=pb1-glob-head-snn-signin](https://app.powerbi.com/groups/bce11fba-b1ec-4b32-a89e-5aeb7b2d25ba/list?cmplid=pb1-glob-head-snn-signin). The workspace interface includes a sidebar with icons for New, Upload, Content, Datasets + dataflows, and a list of items:

	Name	Type	Owner	Refreshed	Next
	BBIUG Movies	Report	DIAD WS	4/11/19, 2:05:29 AM	—
	Composite model movies	Report	DIAD WS	12/30/20, 4:02:10 PM	—
	Composite model movies	Dataset	DIAD WS	12/30/20, 4:02:10 PM	N/A
	CSNA Movies	Report	DIAD WS	4/11/19, 2:05:29 AM	—
	dataflow sydney	Dataflow	Reza Rad	2/13/20, 3:24:17 PM	N/A

**Figure 25-10.** Getting the workspace ID

The same workspace ID is also returned as the result of PowerShell Cmdlets such as Get-PowerBIWorkspace, as shown in Figure 25-11.

<b>Id</b>	: 6a6aa149-1ed8-446c-a235-41a3d1f5ca83
<b>Name</b>	: DPS WS
<b>Type</b>	: Workspace
<b>IsReadOnly</b>	: False
<b>IsOrphaned</b>	: False
<b>IsOnDedicatedCapacity</b>	: False
<b>CapacityId</b>	:

**Figure 25-11.** Getting the workspace ID from PowerShell Cmdlets

This workspace ID can now be used as a parameter for the Get-PowerBIReport Cmdlet to give you the list of all the reports in the workspace.

```
Get-PowerBIReport -WorkspaceId <id of your workspace>
```

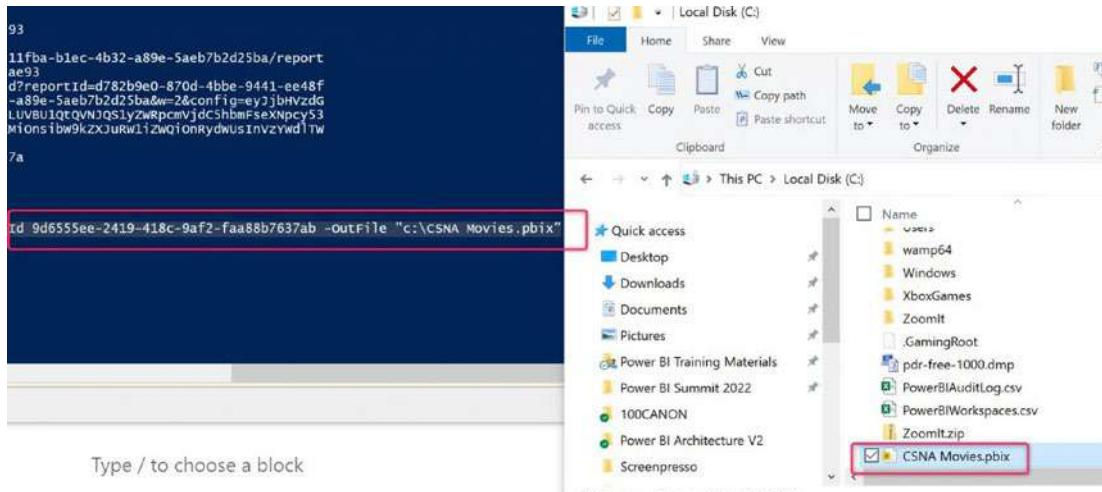
Figure 25-12 shows the output.

**Figure 25-12.** Getting a list of Power BI reports in a workspace

To export the PBIX file of one of these reports, you can use the report ID and the output filename as follows:

```
Export-PowerBIReport -Id <report Id> -OutFile "c:\CSNA Movies.pbix"
```

This downloads the PBIX file for that report into the local path (see Figure 25-13).



**Figure 25-13.** The PBIX file was exported using the PowerShell Cmdlet

Now you can publish this PBIX file to another workspace. You need the workspace ID of the destination workspace, and you can use the following Cmdlet:

```
New-PowerBIReport -Path "C:\CSNA Movies.pbix" -WorkspaceId <workspace id>
```

The report is now copied into this destination workspace, as shown in Figure 25-14.

The screenshot shows the Power BI service interface. At the top, there is a navigation bar with icons for 'PW' (Power BI), 'Development', and other options. Below the navigation bar, there are buttons for '+ New', 'Upload', 'View deployment pipeline', 'View', 'Filters', 'Settings', 'Access', and 'Search'.

The main area displays a table with columns: 'All', 'Name', 'Type', 'Owner', 'Refreshed', 'Next refresh', and 'Endorser'. A red box highlights the row for 'CSNA Movies', which is a 'Report' type owned by 'PW' and was last refreshed on 4/11/19 at 2:05:29 AM. The 'Next refresh' column shows a blank entry.

To the right of the table, there is a 'Module: MicrosoftPowerBIManagementReports (Imported)' section with a 'Parameters for "New-PowerBIReport"' dialog. This dialog contains fields for 'WorkspaceId' (set to 'd8aeefb1-1052-476b-90de-dcdde69b623'), 'Path' (set to 'C:\CSNA Movies.pbix'), 'ConflictAction', 'Name', 'Timeout', and 'WorkspaceId' (set to 'd8aeefb1-1052-476b-90de-dcdde69b623'). There is also a 'Common Parameters' section.

**Figure 25-14.** Publishing the Power BI report into a workspace

This action is different from the Copy-PowerBIReport Cmdlet. The difference between these two is that the Copy Cmdlet creates a live connection to the dataset in the source workspace, whereas the method just explained creates a copy of the dataset.

This method can be expanded into looping through all the reports in the source workspace and exporting them into the destination workspace, like a full deployment pipeline from the Development environment into the Test or Production environment. However, explaining such a process is outside of this chapter's scope.

## Are PowerShell Cmdlets the Same as REST APIs in Power BI?

The answer to this question is yes and no. These Cmdlets are designed on the basis of the REST API functions. However, far more capabilities are available in the REST APIs than in the PowerShell Cmdlets. PowerShell Cmdlets are kind of a layer on top of REST APIs. There is not a Cmdlet for every REST API; there are functions that you can only do through REST API.

One big benefit of PowerShell Cmdlets compared to REST API, however, is that to use the PowerShell Cmdlets, you don't need to be a C# or VB.NET developer. PowerShell scripts are much simpler to use and learn. An administrator can use them with a little practice.

However, if you need more extensive usage of APIs, REST APIs through a custom application is the way to go. If you have seen the capabilities in the Power BI Helper,<sup>4</sup> which helps with documentation in the service<sup>5</sup> and exports reports from one workspace to another,<sup>6</sup> those are implemented using the .NET REST API.

## Summary

In a nutshell, Power BI has a set of modules and Cmdlets for PowerShell. These are simple to use and, combined with the generic Cmdlets in Power BI, gives you capabilities such as the scenarios mentioned in this chapter (creating deployment scripts or generating documentation on a time-based schedule). Learning PowerShell is not as complex as learning a programming language. This, combined with the ability to connect to Power BI Service objects from PowerShell, enables Power BI administrators and developers to leverage these functionalities and automate some scenarios. The Cmdlets and modules are still a work in progress; however, they are always behind the REST API functionality. If you need more extensive capabilities through programming, REST API is a better option.

---

<sup>4</sup> Learn more: <https://radacad.com/power-bi-helper>

<sup>5</sup> Learn more: <https://radacad.com/document-the-power-bi-tenant-objects-workspace-reports-datasets-dataflows-with-no-code>

<sup>6</sup> Learn more: <https://radacad.com/copy-workspace-publish-to-multiple-workspaces-download-all-pbix-files-and-much-more-in-power-bi-helper-april-edition>

## CHAPTER 26



# Power BI REST API

You can interact with Power BI objects in the Power BI Service through a set of APIs called the Power BI REST API. The Power BI REST API can help automate tasks, build tools that work with Power BI, configure Power BI outside of the platform, and embed Power BI into third-party applications. This chapter describes the REST API and explains why it is useful with a few examples.

## The Power BI REST API

A REST API is a type of web service that can be used with or without parameters and provide responses about a specific area. In the case of Power BI, the Power BI REST API is a set of web services that can work with the Power BI objects in the Power BI Service. There are many Power BI use cases for the REST API. Some examples follow:

- Refreshing the dataset automatically when it fails
- Refreshing the dataset immediately after successfully finishing the refresh of all related dataflows
- Creating a deployment pipeline
- Embedding a Power BI report or dashboard into a web application
- Automating documentation of Power BI Service objects
- Sending data rows to a streaming dataset in the Power BI Service
- Extracting the use of Power BI objects by the users
- And many other scenarios

Power BI REST APIs can be used in programming (.NET or other languages), PowerShell, and other tools. Most of the use cases for the APIs are through a programming language. This chapter doesn't go into detail about the program in which you invoke the REST API. Instead, this chapter focuses on where to get the information.

## Getting Started

To start working with REST API for Power BI, you can go to [learn.microsoft.com/en-us/rest/api/power-bi/](https://learn.microsoft.com/en-us/rest/api/power-bi/). This is the link to the documentation for the REST API. The REST API operation is categorized into groups to find the function you are looking for easily, as shown in Figure 26-1.

Operation group	Description
Admin	Operations for working with administrative tasks.
Apps	Operations for working with Apps.
Available Features	Operations that return available features.
Capacities	Operations for working with capacities.
Dashboards	Operations for working with dashboards.
Dataflow Storage Accounts	Operations for working with dataflow storage accounts.
Dataflows	Operations for working with dataflows.
Datasets	Operations for working with datasets.
Embed Token	Operations for working with embed tokens.
Gateways	Operations for working with gateways.
Groups	Operations for working with groups.
Imports	Operations for working with imports.
Pipelines	Operations for working with deployment pipelines.
Push Datasets	Operations for working with push datasets.
Reports	Operations for working with reports.
Template Apps	Operations for working with Template Apps.
Users	Operations for working with users.

**Figure 26-1.** Power BI REST API operation groups

This list is updated regularly, and you can expect more functions and operations to be available through the REST API in the future. Once you click an operation group, you will see all the functions under that group, as shown in Figure 26-2.

Clone Report	Clones the specified report from My workspace.
Clone Report In Group	Clones the specified report from the specified workspace.
Delete Report	Deletes the specified report from My workspace.
Delete Report In Group	Deletes the specified report from the specified workspace.
Export Report	Exports the specified report from My workspace to a Power BI .pbix or .rdl file.
Export Report In Group	Exports the specified report from the specified workspace to a Power BI .pbix or .rdl file.
Export To File	Exports the specified report from My workspace to the requested file format.
Export To File In Group	Exports the specified report from the specified workspace to the requested file format.
Get Datasources	Returns a list of data sources for the specified paginated report (RDL) from My workspace.
Get Datasources In Group	Returns a list of data sources for the specified paginated report (RDL) from the specified workspace.
Get Export To File Status	Returns the current status of the Export to File job for the specified report from My workspace.
Get Export To File Status In Group	Returns the current status of the Export to File In Group job for the specified report from the specified workspace.
Get File Of Export To File	Returns the file from the Export to File job for the specified report from My workspace.
Get File Of Export To File In Group	Returns the file from the Export to File In Group job for the specified report from the specified workspace.
Get Page	Returns the specified page within the specified report from My workspace.
Get Page In Group	Returns the specified page within the specified report from the specified workspace.
Get Pages	Returns a list of pages within the specified report from My workspace.
Get Pages In Group	Returns a list of pages within the specified report from the specified workspace.
Get Report	Returns the specified report from My workspace.
Get Report In Group	Returns the specified report from the specified workspace.
Get Reports	Returns a list of reports from My workspace.
Get Reports In Group	Returns a list of reports from the specified workspace.
Rebind Report	Rebinds the specified report from My workspace to the specified dataset.
Rebind Report In Group	Rebinds the specified report from the specified workspace to the specified dataset.
Take Over In Group	Transfers ownership of the data sources for the specified paginated report (RDL) to the current authorized user.
Update Datasources	Updates the data sources of the specified paginated report (RDL) from My workspace.
Update Datasources In Group	Updates the data sources of the specified paginated report (RDL) from the specified workspace.
Update Report Content	Updates the content of the specified report from My workspace with the content of a specified source report.
Update Report Content In Group	Updates the content of the specified report from the specified workspace with the content of a specified source report.

**Figure 26-2.** The REST API functions for Power BI reports

The list in Figure 26-2 is for the REST API functions for Power BI reports. If you compare this list to the Power BI Cmdlets for PowerShell reports, you can see how extensive the REST API list is. This shows you immediately that the REST API is more powerful than the PowerShell Cmdlets for Power BI (although the PowerShell Cmdlets are easier for non-programmers to use).

A helpful feature in the Microsoft documentation for these REST APIs is that you can try a REST API function from the web page see (Figure 26-3). This way, you can call the function in your environment using your Power BI account and see if it is what you are looking for.

**Figure 26-3.** Trying a REST API function through a browser

For example, if you run the Get Refresh History API for a sample dataset, you can get the response through the Try window in the browser (see Figure 26-4).

Run ▶

Response Code: 200

Headers

HTTP Copy

```
cache-control: no-store, must-revalidate, no-cache
content-length: 1188
content-type: application/json; odata.metadata=minimal
pragma: no-cache
requestid: 9009b528-98d2-46bc-b0de-fb7cca90d829
```

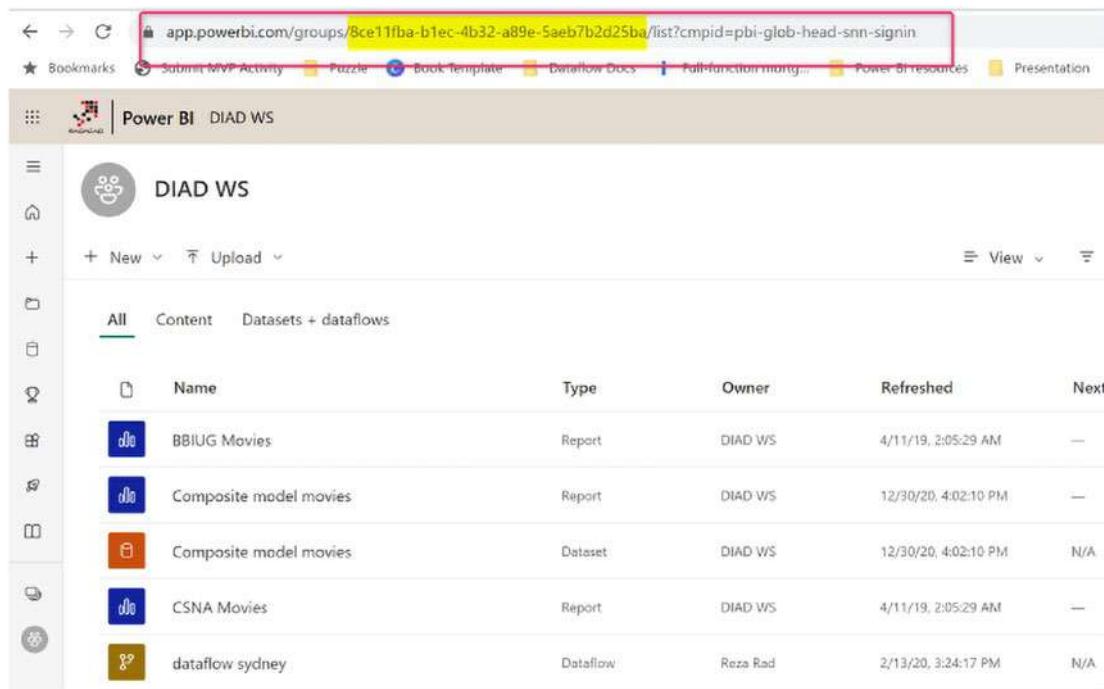
Body

JSON Copy

```
{
  "@odata.context": "http://wabi-south-east-asia-redirect.analysis.windows.net/v1.0/myorg/$metadata#refreshes",
  "value": [
    {
      "requestId": "0dc570b5-eb8c-4c6b-bfbb-50e3d85f2e69",
      "id": 2127983696,
      "refreshType": "Scheduled",
      "startTime": "2023-01-17T17:01:06.873Z",
      "endTime": "2023-01-17T17:11:43.583Z",
      "status": "Completed"
    },
    {
      "requestId": "8d8daf54-6726-47c3-a71c-bdfb73623897",
      "id": 2126933294,
      "refreshType": "Scheduled",
      "startTime": "2023-01-16T17:01:17.643Z",
      "endTime": "2023-01-16T17:09:38.647Z",
      "status": "Completed"
    },
    {
      "requestId": "c2c164ca-2fd3-40a5-a993-ed6cbdd562a2",
      "id": 2126933295,
      "refreshType": "Scheduled",
      "startTime": "2023-01-16T17:01:17.643Z",
      "endTime": "2023-01-16T17:09:38.647Z",
      "status": "Completed"
    }
  ]
}
```

**Figure 26-4.** The REST API sample results in the web browser

Some APIs require the ID of the objects in the Power BI service. These IDs are the unique identifier codes that can be fetched through the URL of the object in the Power BI Service (see Figure 26-5) or through the response of some other API functions.



The screenshot shows a web browser window with the URL `app.powerbi.com/groups/8ce11fb-a-b1ec-4b32-a89e-5aeb7b2d25ba/list?cmplid=pb1-glob-head-snn-signin` highlighted in yellow. The browser interface includes a back button, forward button, and search bar. Below the address bar, there are links for Bookmarks, Submit MVP Activity, Puzzle, Book template, Dataflow Docs, Full function library, Power BI resources, and Presentation. The main content area is titled "Power BI - DIAD WS". On the left is a sidebar with various icons. The main content area displays a table with five rows of data:

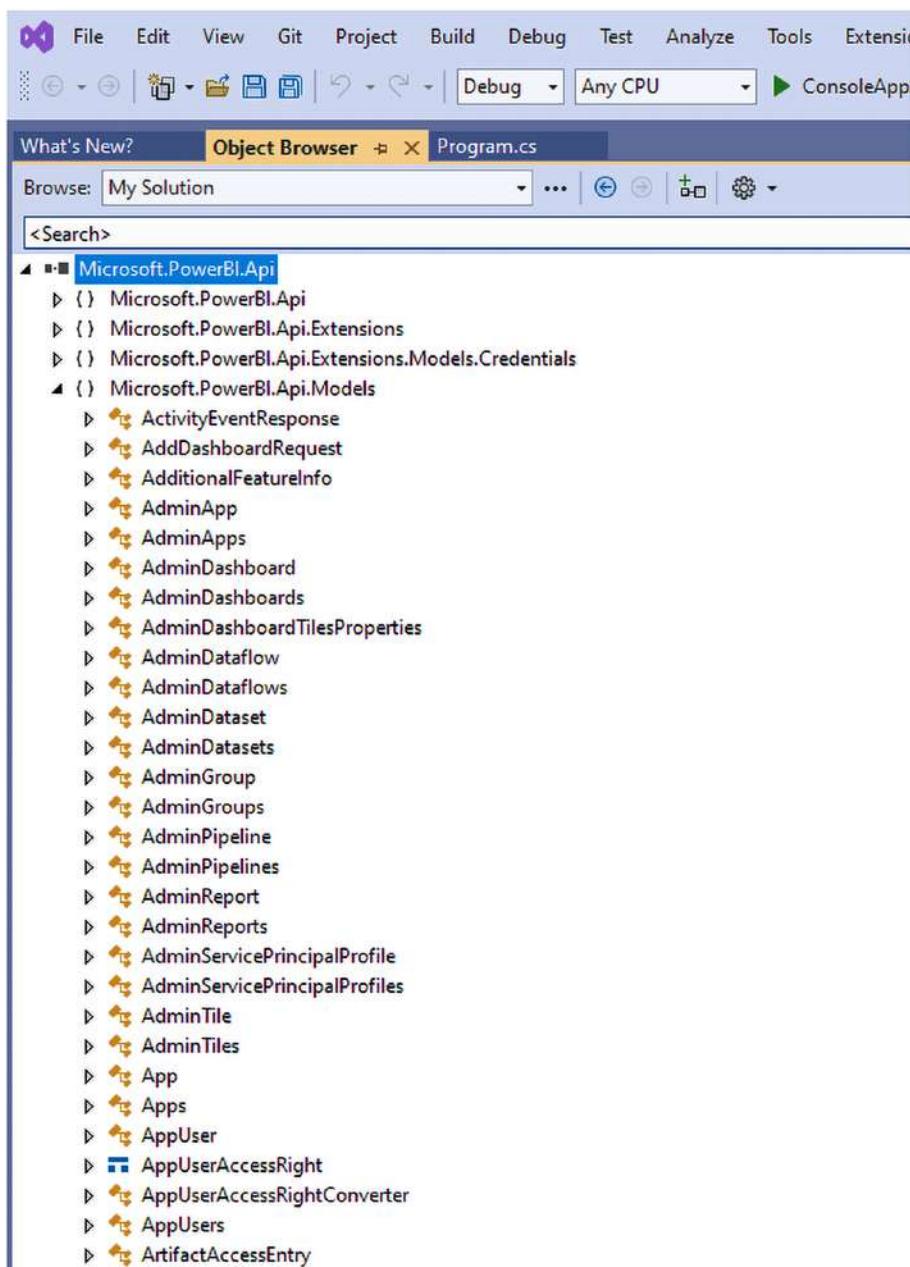
	Name	Type	Owner	Refreshed	Next
	BBIUG Movies	Report	DIAD WS	4/11/19, 2:05:29 AM	—
	Composite model movies	Report	DIAD WS	12/30/20, 4:02:10 PM	—
	Composite model movies	Dataset	DIAD WS	12/30/20, 4:02:10 PM	N/A
	CSNA Movies	Report	DIAD WS	4/11/19, 2:05:29 AM	—
	dataflow sydney	Dataflow	Reza Rad	2/13/20, 3:24:17 PM	N/A

**Figure 26-5.** Fetching a workspace ID from the Power BI Service URL

## Calling the API from .NET

As mentioned, most REST API use is from a custom application or program. C# and VB.NET are the two most commonly used languages in the Microsoft world for the REST API of Power BI. To use the REST API through these programming languages, you can download (or install) the .NET SDK for Power BI (which is updated regularly).

This Power BI NuGet will give the programmer the list of functions to use, as shown in Figure 26-6.



**Figure 26-6.** The Power BI functions and classes in the Power BI SDK for .NET

An example of such an application is the Power BI Helper, which uses the functions in the .NET SDK of Power BI to get the Power BI service documentation.

To get started with the .NET SDK of Power BI, you must register an application for Power BI in your organization's tenant active directory. Then you must complete the authentication process. If you are

interested in learning more about it and seeing some samples of working with the .NET SDK of Power BI, read this article series:

- Part 1: Register the Application ([radacad.com/integrate-power-bi-into-your-application-part-1-register-your-app](http://radacad.com/integrate-power-bi-into-your-application-part-1-register-your-app))
- Part 2: Authentication ([radacad.com/integrate-power-bi-into-your-application-part-2-authenticate](http://radacad.com/integrate-power-bi-into-your-application-part-2-authenticate))
- Part 3: Embed Content ([radacad.com/integrate-power-bi-into-your-application-part-3-embed-content](http://radacad.com/integrate-power-bi-into-your-application-part-3-embed-content))
- Part 4: Refresh the Dataset ([radacad.com/integrate-power-bi-into-your-application-part-4-refresh-data-set](http://radacad.com/integrate-power-bi-into-your-application-part-4-refresh-data-set))
- Part 5: Manage the Data Source ([radacad.com/integrate-power-bi-into-your-application-part-5-data-source-management](http://radacad.com/integrate-power-bi-into-your-application-part-5-data-source-management))
- Part 6: Send Data Rows to Real-Time Streaming Datasets ([radacad.com/integrate-power-bi-into-your-application-part-6-real-time-streaming-and-push-data](http://radacad.com/integrate-power-bi-into-your-application-part-6-real-time-streaming-and-push-data))

## Embedding into a Custom Application

A common use case for the REST API of Power BI is embedding the report or dashboard into a custom web application. The Power BI documentation has two good resources for that:

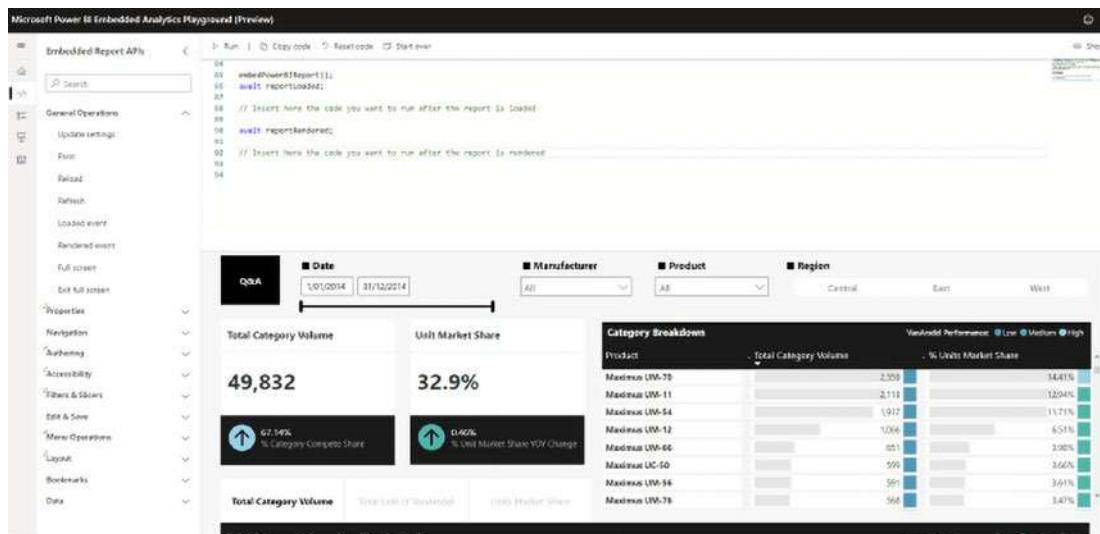
- Power BI Embedded analytics documentation—See [learn.microsoft.com/en-us/power-bi/developer/embedded/](https://learn.microsoft.com/en-us/power-bi/developer/embedded/)
- Power BI Playground—See [playground.powerbi.com/en-us/](https://playground.powerbi.com/en-us/)

The first link contains all the documentation you need for Power BI Embedded. The Power BI Playground (see Figure 26-7) is an interesting place to run embedded code as a test in the browser.



**Figure 26-7.** The Power BI Embedded Analytics Playground

The playground environment gives you a sandbox to see how code results perform in the application, get sample codes, and even run the code for your tenant, as shown in Figure 26-8.



**Figure 26-8.** Testing codes in the Power BI Embedded Playground

## Power Automate

Some of the most common REST API functions for Power BI are streamlined through Power Automate. Power Automate is a service for designing flow and automation. You can, for example, use the Power BI functions in Power Automate to do tasks like the following:

- Export Power BI paginated reports using Power Automate ([learn.microsoft.com/en-us/power-bi/collaborate-share/service-automate-paginated-integration](https://learn.microsoft.com/en-us/power-bi/collaborate-share/service-automate-paginated-integration))
- Export and email reports using Power Automate ([learn.microsoft.com/en-us/power-bi/collaborate-share/service-automate-power-bi-report-export](https://learn.microsoft.com/en-us/power-bi/collaborate-share/service-automate-power-bi-report-export))

Power Automate includes a user-friendly UI; you don't need to be a programmer to work with the Power BI functions. An example of using Power Automate functions for Power BI is using Power Automate to trigger when a Microsoft Form is submitted and send the result to a real-time streaming Power BI dataset.

## PowerShell to Call REST API

Another way to call the REST API is through a PowerShell command called `Invoke_RestMethod`. Check out the documentation of this function to learn how to use it at [learn.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/invoke-restmethod?view=powershell-7.3](https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/invoke-restmethod?view=powershell-7.3).

## Summary

Power BI REST APIs are web services that can work with the Power BI Service objects (such as workspaces, datasets, dataflows, reports, and more). Using the REST API, a programmer can use a custom application to automate configurations for a Power BI Service solution and extend the solution to another level. Things that are not possible through the normal GUI of the Power BI service might be possible through the REST API.

## CHAPTER 27



# Power BI Audit Log for the Tenant

The Power BI dashboard and reports come with a usage metric, so you can see how users used this content. There is another report for usage metrics across the entire tenant, which you can see if you have access to the Power BI Administrator account under Admin Panel in the Power BI Service. However, what if you want to create your own detailed usage metrics report across the entire tenant? If you want to see across all workspaces in the tenant, you need a different approach. This information is not easily accessible in the Power BI Service. This chapter explains how to extract the Audit Log from Office 365, export it into text files, and create a Power BI report from it. Or in other words, how to create a custom usage metrics report across the tenant.

## The Problem

Let's first consider the issue. Each report and dashboard has a usage metrics report, which shows where and when and by whom the content was used. You can usually find it at the top of the dashboard or report in the Power BI Service, as shown in Figure 27-1.

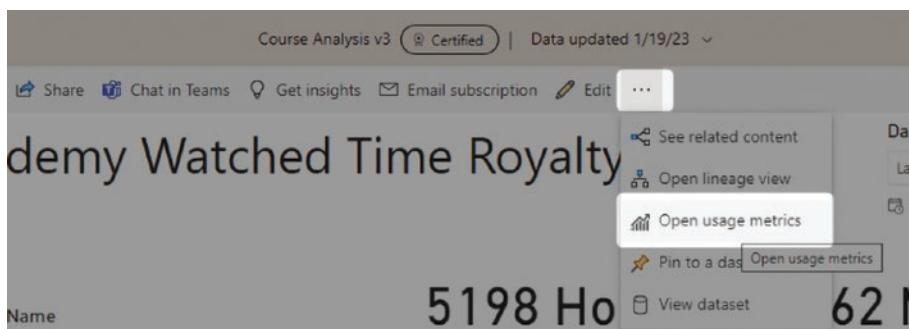
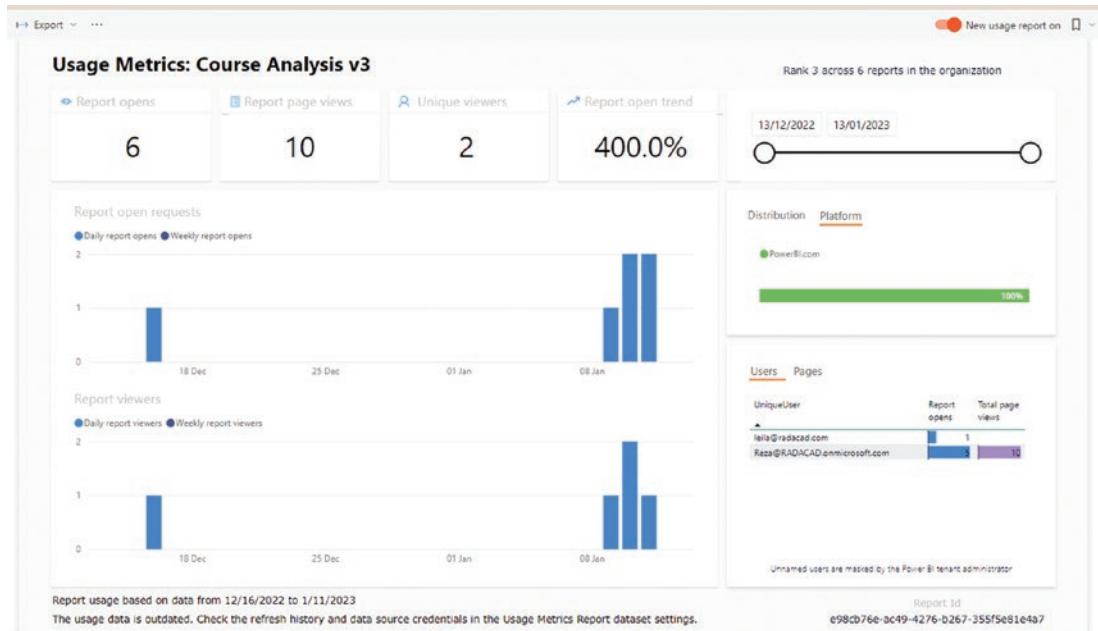


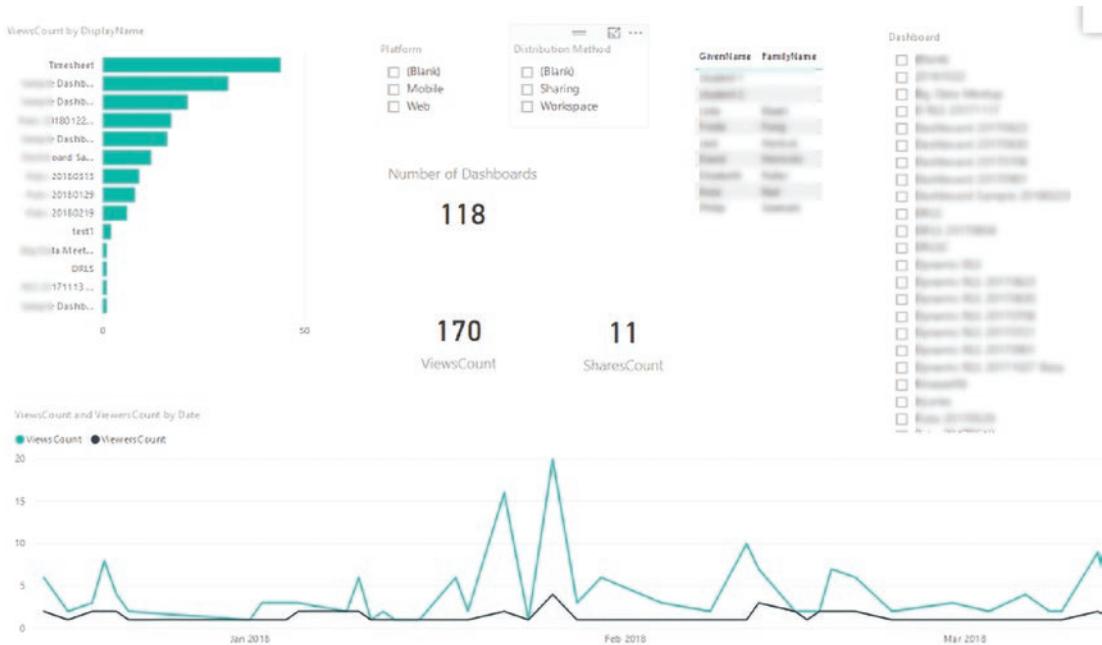
Figure 27-1. Viewing usage metrics

Figure 27-2 shows an example of a usage metrics report.



**Figure 27-2.** Usage metrics for a report in the Power BI Service

I previously wrote an article and explained how you can customize this report and create your version of this report using Save As, and then edit the report. Figure 27-3 shows a customized usage metrics report that I created.



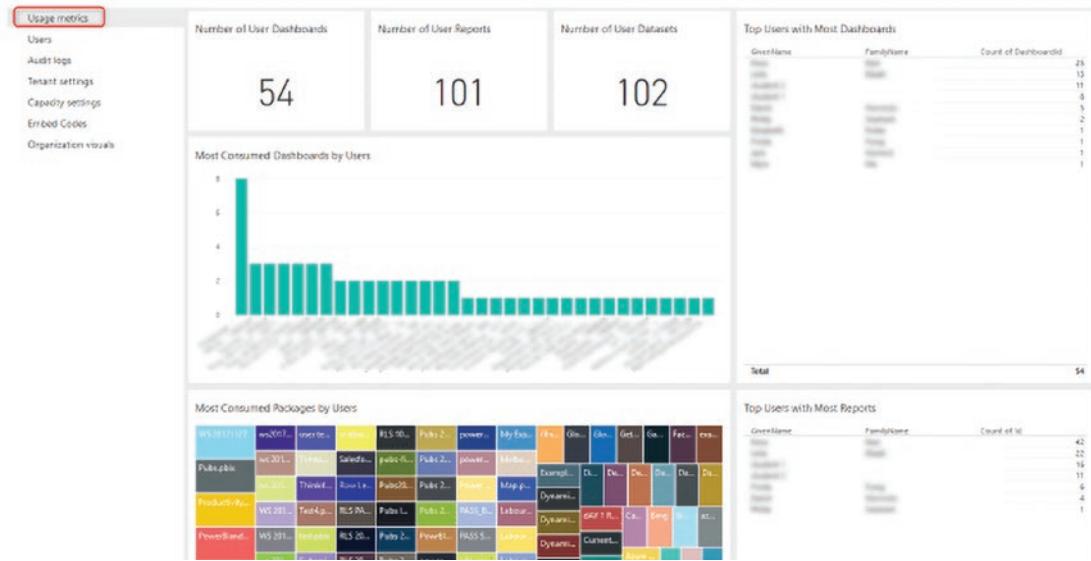
**Figure 27-3.** A customized usage metrics report

However, there is still a big challenge. This report will show me the usage metrics only in the current workspace, not in other workspaces. I would have to produce this report in each workspace separately!

Usage metrics of reports and the dashboard, even after customizing, show only the current workspace's data, not all workspace data.

On the other hand, if you have access to the Power BI Administrator account, under Admin Portal, you can find the usage metrics of all reports and dashboards across the tenant (see Figure 27-4).

## Admin portal



**Figure 27-4.** Viewing usage metrics via the Admin Portal

The challenge with this report is that you cannot customize it.

The usage metrics in the Admin Portal give you the metrics related to all content in the Power BI tenant, but that report is customizable and lots of details are missing.

Considering these two points, you are looking for a way to get a usage metrics report that gives you information about all the activities in the Power BI tenant. Let's consider how this is possible.

## The Audit Log

The Power BI Service leverages the Office 365 logging system. The Power BI activities are already logged into Office 365; you just need to find them. If you have an account with sufficient privilege to the Audit Log, you can go to Admin Portal, and under Audit Log, access the Audit Log through the Microsoft 365 Admin Center. See Figure 27-5.

The screenshot shows the Power BI Admin portal interface. On the left, there's a sidebar with navigation links: Tenant settings, Usage metrics, Users, Premium Per User, Audit logs (which is highlighted), Capacity settings, Refresh summary, and Embed Codes. The main content area has a heading "Admin portal" and a sub-section titled "Audit logs are managed in the Microsoft 365 Admin Center". It includes a link "Go to Microsoft 365 Admin Center". Below this, there's a note about auditing being available in certain regions while it's in preview, with a link to learn more.

**Figure 27-5.** Link to Audit Log from the Power BI Admin Portal

This will open the Audit Log search in the Office Portal (see Figure 27-6).

The screenshot shows the "Audit log search" dialog in the Office Portal. The left sidebar lists various security categories: Home, Alerts, Permissions, Classifications, Data loss prevention, Data governance, Supervision, Threat management, Mail flow, Data privacy, Search & investigation, and Reports. The main search interface includes a "Search" bar with a "Clear" button, a "Results" table header with columns for Date, IP address, User, Activity, and Item, and a "Run a search to view results" button. The search form allows filtering by activities (Show results for all activities), start date (2019-02-19, 00:00), end date (2019-02-27, 00:00), users (Show results for all users), file/folder/URL (Add all or part of a file name, folder name, or URL), and a "Search" button.

**Figure 27-6.** The Audit Log search dialog

Now you can search for the activities by selecting them in the Activities drop-down list, as shown in Figure 27-7.

CHAPTER 27 ■ POWER BI AUDIT LOG FOR THE TENANT

Results			
Date ▾	IP address	User	Activity
<a href="#">Show results for all activities</a> ▾			
<a href="#">Clear all to show results for all activities</a>			
<input type="text" value="power b"/>			
<b>Power BI activities</b>			
Viewed Power BI dashboard	Created Power BI dashboard	Edited Power BI dashboard	
Deleted Power BI dashboard	Shared Power BI dashboard	Printed Power BI dashboard	
Viewed Power BI tile	Exported Power BI tile data	Viewed Power BI report	
Deleted Power BI report	Printed Power BI report page	Downloaded Power BI report	
Published Power BI report to web	Exported Power BI report visual data	Created Power BI report	
Edited Power BI report	Created Power BI dataset	Deleted Power BI dataset	
Created Power BI group	Deleted Power BI group	Added Power BI group members	
Removed Power BI group members	Created organizational Power BI content pack	Created Power BI app	
Installed Power BI app	Updated Power BI app	Updated organization's Power BI settings	
Started Power BI trial	Started Power BI extended trial	Analyzed Power BI dataset	

**Figure 27-7.** Selecting activities via the Activities drop-down list

You can also add other criteria, such as your search's start and end date and users to search. Figure 27-8 shows some example output.

## Audit log search

Need to find out if a user deleted a document or changed someone's password? Search the Office 365 audit logs to find out what the users and admins in your organization have been doing. You'll be able to find activity related to email, groups, documents, permissions, direct mail services, and much more. Learn more about searching the audit log.

**Figure 27-8.** Output based on example search criteria

This output can then be exported as a source of a Power BI report. However, you cannot export all Power BI activities of all users. You will get an error like the one shown in Figure 27-9.

## Audit log search

Need to find out if a user deleted a document or if permissions, directory services, and much more. Le

Search Clear

Please reduce the number of selected activities or users

Activities

Viewed Power BI dashboard, ... (88) ▾

Start date

2019-01-01 Calendar 00:00 ▾

End date

2019-02-27 Calendar 00:00 ▾

Users

Show results for all users

File, folder, or site i

Add all or part of a file name, folder name, or URL.

Search

Figure 27-9. Error message generated when trying to export all activity by all Power BI users

Exporting the log data manually, and partially each time, is not practical, so you need another way to extract the Audit Log information.

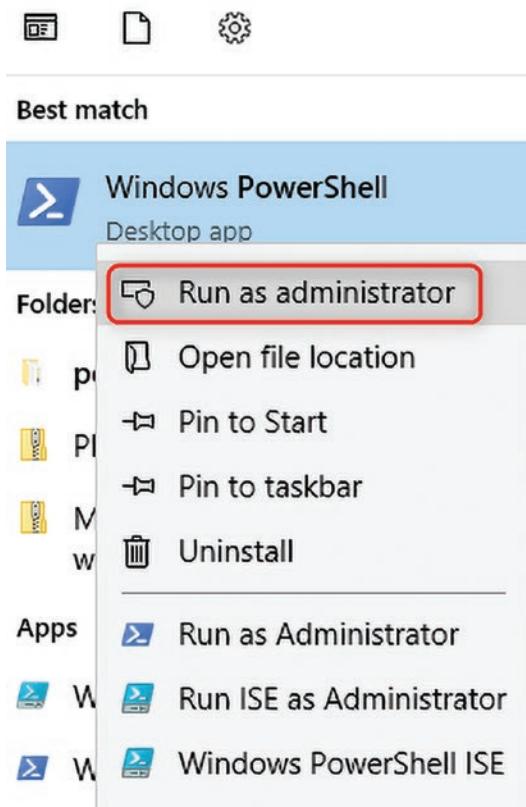
## Using PowerShell to Extract the Audit Log

There is more than one way to extract the Audit Log. Here are some methods:

- Using Office 365 Audit Log PowerShell Cmdlets (used in this chapter)
- Using Power BI Cmdlets for PowerShell
- Using the REST API for Power BI
- Using a third-party tool, such as Power BI Helper

These methods can all be used to export the Audit Log. The method I explain here uses the Office 365 Audit Log Cmdlets for PowerShell. This method can also extract logs about all other parts of Office 365.

PowerShell is an expression-based tool that automates some of the work for admins. PowerShell can be used to access the Office 365 Audit Log. First of all, you need to open PowerShell as an administrator (see Figure 27-10).



**Figure 27-10.** Running PowerShell as the administrator

Then start the following script (needed only once for a machine):

```
Install-Package ExchangeOnlineManagement
Import-Module ExchangeOnlineManagement
```

After installing and importing the module, you can then log in to your Power BI account using the following command (see Figure 27-11):

```
Connect-ExchangeOnline -UserPrincipalName <Power BI email account>
```

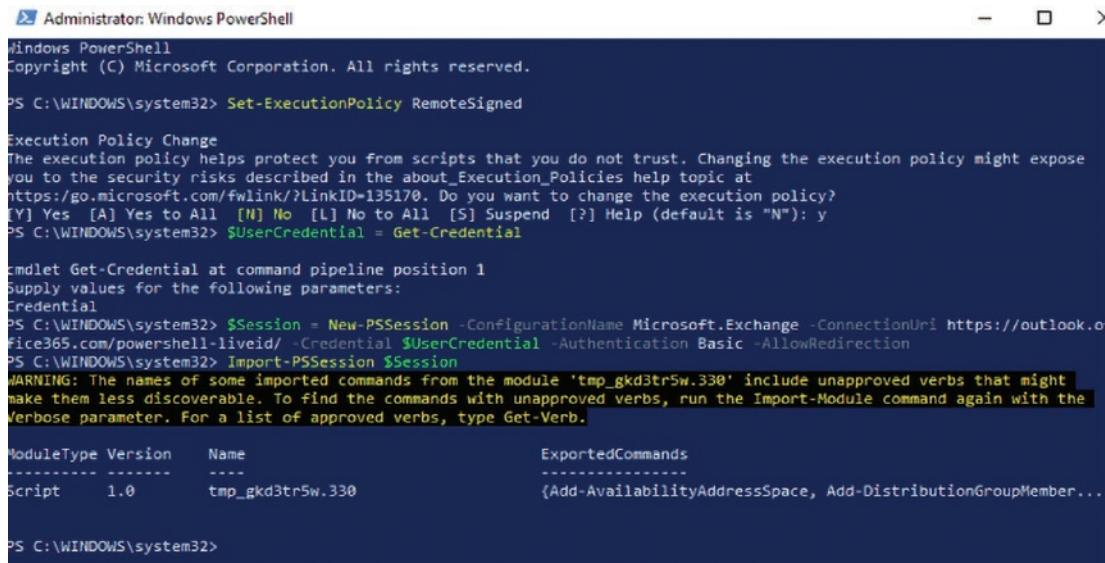


**Figure 27-11.** Log in to the Office 365 account using the PowerShell Cmdlet

To access the entire Audit Log across the tenant, you need access to the admin account. This is the same account you use in the previous script.

After successfully logging in, you can start accessing the Audit Log. To do that, you need another package, called ExchangePowerShell. Install and Import (see Figure 27-12) uses the following script (needed only once per machine):

```
Install-Package ExchangePowerShell
Import-Module ExchangePowerShell
```



**Figure 27-12.** The Install and Import package

As the final step, you can read the Audit Log using the `Search-UnifiedAuditLog` function:

```
Search-UnifiedAuditLog -StartDate (Get-Date).AddDays(-90) -EndDate (Get-Date) -RecordType PowerBIAudit -ResultSize 10
```

This will give you only ten rows of the Audit Log (I have limited the result size to ten), as shown in Figure 27-13.

```
C:\> Administrator: Windows PowerShell
PS C:\WINDOWS\system32> Search-UnifiedAuditLog -StartDate (Get-Date).AddDays(-90) -EndDate (Get-Date) -RecordType PowerBIAudit -ResultSize 10

UnspacedId : 7379e9e2a-543a-45bf-a22d-56a1e001f58f
RecordType : PowerBIAudit
CreationDate : 25/02/2019 4:09:37 AM
UserIds : elizabeth.puller@radacad.com
Operations : ViewDashboard
AuditData : {"Id": "21bb0b2b-6474-49a4-88fe-7b342c4056f6", "RecordType": 20, "CreationTime": "2019-02-25T04:09:37", "Operation": "ViewDashboard", "OrganizationId": "cc80f8d6-3c35-4da6-b6f8-cead0e0467e", "UserType": 0, "UserKey": "10837FFEA6615A3", "Workload": "PowerBI", "UserId": "elizabeth.puller@radacad.com", "ClientIP": "101.98.104.205", "UserAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.109 Safari/537.36", "Activity": "ViewDashboard", "ItemName": "Timesheet", "WorkspaceName": "My Workspace", "DashboardName": "Dashboard1Name", "Timesheet": "Timesheet", "WorkspaceId": "My Workspace", "Object": "Timesheet", "DashboardId": "f15a8e28-94b8-4eab-9cad-f0bb071a5dd", "Datasets": [{"DatasetId": "13f005f2-1070-43d9-acd5-161b7ae0e9df", "DatasetName": "Timesheet"}], "IsSuccess": true, "RequestId": "9bb880ac-4062-41b8-a5b8-9e8e17a40987", "ActivityId": "73c9c77-celd-4093-8502-87b6e1d1d1d1"}
ResultIndex : 1
ResultCount : 1370
Identity : 21bb0b2b-6474-49a4-88fe-7b342c4056f6
IsValid : True
ObjectState : Unchanged

UnspacedId : 7379e9e2a-543a-45bf-a22d-56a1e001f58f
RecordType : PowerBIAudit
CreationDate : 25/02/2019 4:09:37 AM
UserIds : elizabeth.puller@radacad.com
Operations : ViewDashboard
AuditData : {"Id": "b21b1cf9-0af9-498c-a994-0de5419c9e13", "RecordType": 20, "CreationTime": "2019-02-25T04:09:37", "Operation": "ViewDashboard", "OrganizationId": "cc80f8d6-3c35-4da6-b6f8-cead0e0467e", "UserType": 0, "UserKey": "10837FFEA6615A3", "Workload": "PowerBI", "UserId": "elizabeth.puller@radacad.com", "ClientIP": "101.98.104.205", "UserAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.109 Safari/537.36", "Activity": "ViewDashboard", "ItemName": "Timesheet", "WorkspaceName": "My Workspace", "DashboardName": "Timesheet", "Timesheet": "Timesheet", "WorkspaceId": "My Workspace", "Object": "Timesheet", "DashboardId": "f15a8e28-94b8-4eab-9cad-f0bb071a5dd", "Datasets": [{"DatasetId": "13f005f2-1070-43d9-acd5-161b7ae0e9df", "DatasetName": "Timesheet"}], "IsSuccess": true, "RequestId": "b39920d-cb59-7172-2a23-682330ca961f", "ActivityId": "73c9c77-celd-4093-8502-87b6e1d1d1d1"}
ResultIndex : 2
ResultCount : 1370
Identity : b21b1cf9-0af9-498c-a994-0de5419c9e13
IsValid : True
ObjectState : Unchanged

UnspacedId : 7379e9e2a-543a-45bf-a22d-56a1e001f58f
RecordType : PowerBIAudit
CreationDate : 25/02/2019 3:20:02 AM
UserIds : elizabeth.puller@radacad.com
Operations : ViewReport
AuditData : {"Id": "11b8f10f-71f2-485a-adcc-5e75428591b6", "RecordType": 20, "CreationTime": "2019-02-25T03:20:02", "Operation": "ViewReport", "OrganizationId": "cc80f8d6-3c35-4da6-b6f8-cead0e0467e", "UserType": 0, "UserKey": "10837FFEA6615A3", "Workload": "PowerBI", "UserId": "elizabeth.puller@radacad.com", "ClientIP": "101.98.104.205", "UserAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.109 Safari/537.36", "Activity": "ViewReport", "ItemName": "Timesheet", "WorkspaceName": "My Workspace", "DatasetName": "Timesheet", "ReportName": "Report1", "ReportId": "13f005f2-1070-43d9-acd5-161b7ae0e9df", "Object": "Timesheet", "DatasetId": "13f005f2-1070-43d9-acd5-161b7ae0e9df", "ReportId": "11324f27-197a-4cb1-8c79-5e62c7eb1#84", "IsSuccess": true, "ReportType": "PowerBIReport", "RequestId": "3b0b1b7e-55f8-c8b1-46c9-6e04ff309a6a", "ActivityId": "8066d447-c02d-86f1-51ac-c34077dfb9f0"}, "ResultIndex : 3
```

**Figure 27-13.** Reading an Audit Log using the `Search-UnifiedAuditLog` function

As you can see in Figure 27-13, you will have a list of log activities with a field named `AuditData`, which includes anything about the activity, including the object name and ID, the operation, the time and date of the operation, users who performed the operation, and the result of the operation.

The next step is to format this data and export it to a CSV file:

```
Search-UnifiedAuditLog -StartDate (Get-Date).AddDays(-90) -EndDate (Get-Date) -RecordType PowerBIAudit -ResultSize 5000 | ConvertTo-Csv | Out-File c:\PowerBAuditLog.csv
```

This will export the output of 5,000 log entries in the period of the last three months to CSV format, as illustrated in Figure 27-14.

```
#TYPE Deserialized.Microsoft.Exchange.Data.ApplicationLogic.UnifiedAuditLogEvent
"PSComputerName","RunspaceId","PSShowComputerName","RecordType","CreationDate","UserIds","Operations","AuditData",
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","26/02/2019 12:08:30 AM","Re
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","25/02/2019 4:09:37 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","25/02/2019 4:09:37 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","25/02/2019 3:20:02 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","25/02/2019 3:09:24 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","25/02/2019 1:08:07 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","25/02/2019 12:02:45 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","22/02/2019 4:08:22 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","22/02/2019 3:41:36 AM","Re
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","22/02/2019 3:07:06 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","22/02/2019 1:05:52 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","22/02/2019 12:07:32 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","21/02/2019 11:03:53 PM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","21/02/2019 10:08:38 PM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","21/02/2019 9:04:35 PM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","21/02/2019 8:05:22 PM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","21/02/2019 4:44:53 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","21/02/2019 4:05:13 AM","el
"outlook.office365.com","737f9e2a-543a-45bf-a22d-56ale001f58f","False","PowerBAudit","21/02/2019 3:03:41 AM","el
```

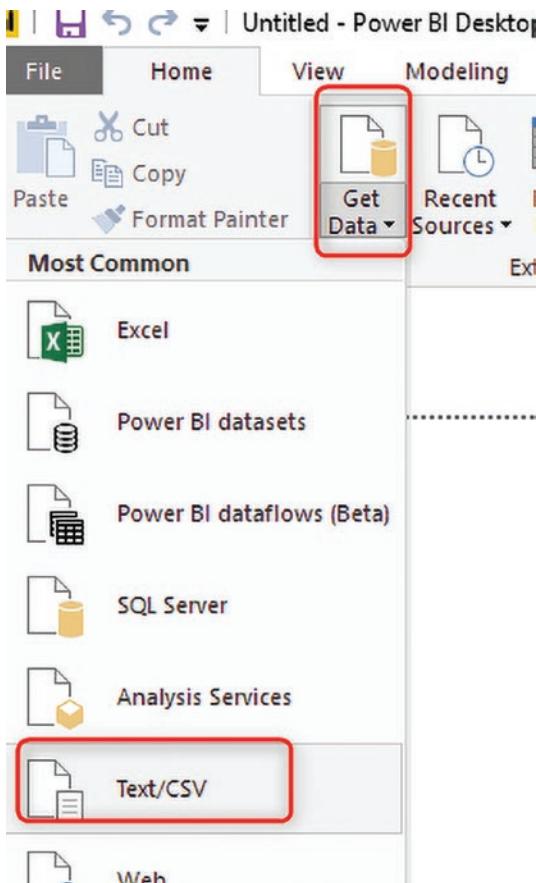
**Figure 27-14.** Formatting data and exporting it to a CSV file

Here are a few tips about the export process in Figure 27-14:

- The default time period that Office 365 keeps the Audit Log is 90 days. You can change it if you want.
- The export only supports up to 5,000 Audit Log transactions. You need to split it into multiple exports if you have more than that.

## The Power BI Report for the Audit Log

Now that you have a CSV file of all the Audit Logs, you can use that as a data source in Power BI. Open a new Power BI file and get the data from the CSV file (see Figure 27-15).



**Figure 27-15.** Getting data from an existing .csv file

After selecting the source file, click Transform Data to go to the Power Query Editor window (see Figure 27-16).

The screenshot shows the Power Query Editor interface with a table containing 20 rows of data. The columns are labeled: RowIndex, PSCreatedOn, PSShowComputerName, RecordType, and CreationDate. The data includes various computer names and creation dates, such as 'outlook.office365.com' on 26/02/2019 at 12:08:20 AM.

RowIndex	PSCreatedOn	PSShowComputerName	RecordType	CreationDate
1	#TYPE! Deserialized.Microsoft.Exchange.Data.ApplicationLogic.Unified...			
2	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	26/02/2019 12:08:20 AM
3	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	25/02/2019 4:09:31 AM
4	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	25/02/2019 4:09:37 AM
5	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	25/02/2019 3:20:02 AM
6	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	25/02/2019 3:20:04 AM
7	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	25/02/2019 3:20:24 AM
8	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	25/02/2019 1:08:07 AM
9	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	25/02/2019 12:02:45 AM
10	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	22/02/2019 4:08:22 AM
11	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	22/02/2019 3:41:36 AM
12	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	22/02/2019 3:07:06 AM
13	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	22/02/2019 1:05:52 AM
14	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	22/02/2019 12:07:32 AM
15	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	21/02/2019 11:03:55 PM
16	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	21/02/2019 10:58:38 PM
17	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	21/02/2019 9:04:35 PM
18	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	21/02/2019 8:05:22 PM
19	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	21/02/2019 4:44:53 AM
20	outlook.office365.com	737f9e2a-543a-45bf-a22d-56a1e001150f	PowerBIAudit	21/02/2019 4:05:13 AM

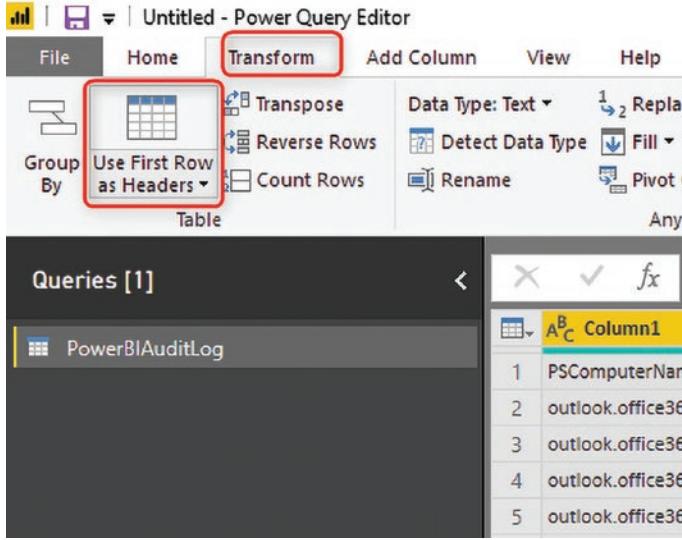
Figure 27-16. The Power Query Editor window

Remove the top row, as it doesn't contain any useful information (see Figure 27-17).

The screenshot shows the Power Query Editor ribbon with the 'Remove Rows' dropdown menu open. The 'Remove Top Rows' option is highlighted with a red box. A tooltip below the menu explains: 'Remove the top N rows from this table.' Below the ribbon, there are several other cleanup options: 'Remove Duplicates', 'Remove Blank Rows', and 'Remove Errors'.

Figure 27-17. Removing the top row of the .csv file

In the next step, go to the Transform tab and choose Use First Row as Headers (see Figure 27-18).



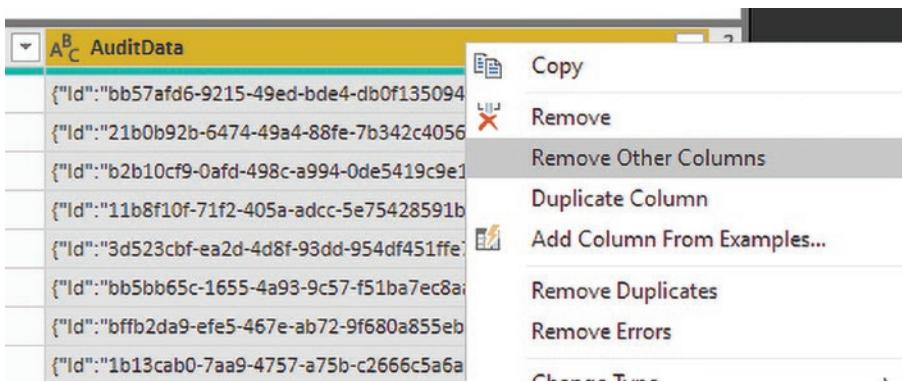
**Figure 27-18.** Selecting Use First Row as Headers from the Transform tab

You will see the log output shown in Figure 27-19.

PComputerName	PSessionId	PSessionComputerName	PRecordType	CreationDate	Userids	Operations	AuditData
outlook.office365.com	7379fe2b-543a-42d6-a22d-5d61e001f5f1		FALSE PowerBIAudit	26/02/2024 22:08:20 AM	Reca@RADACAD.onmicrosoft.co...	ViewUsageMetrics	["id": "b607faef-9215...
outlook.office365.com	7379fe2b-543a-42d6-a22d-5d61e001f5f1		FALSE PowerBIAudit	25/02/2024 0:09:37 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "22d09202-8471...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	25/02/2024 0:09:37 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "6103cf9-093...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	25/02/2024 3:20:02 AM	elizabeth.pullan@radac...	ViewReports	["id": "1118f06e-7101...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	25/02/2024 3:09:24 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "246232cb-ea25...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	25/02/2024 1:08:07 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "bb30085c-1605...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	25/02/2024 22:02:08 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "6ff26d9-ef7e...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	22/02/2024 9:08:22 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "1131030-7819...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	22/02/2024 9:41:16 AM	Reca@RADACAD.onmicrosoft.co...	ViewUsageMetrics	["id": "a356e072-e03c...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	22/02/2024 9:07:06 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "0989ed-3fb4...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	22/02/2024 10:05:52 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "7315c40-8c85...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	22/02/2024 12:07:32 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "7315c40-8c85...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	22/02/2024 11:03:53 PM	elizabeth.pullan@radac...	ViewDashboards	["id": "4938874-e746...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	22/02/2024 10:08:38 PM	elizabeth.pullan@radac...	ViewDashboards	["id": "f802526-e453...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	21/02/2024 9:04:35 PM	elizabeth.pullan@radac...	ViewDashboards	["id": "8315e760-2602...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	21/02/2024 8:05:22 PM	elizabeth.pullan@radac...	ViewDashboards	["id": "e72579f0-2447...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	21/02/2024 4:44:53 AM	elizabeth.pullan@radac...	ViewReport	["id": "03181091-93ec...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	21/02/2024 9:05:33 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "9439f9b-249...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	21/02/2024 9:03:41 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "2726e47-3c10...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	21/02/2024 9:06:24 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "901c475-3cd3...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	21/02/2024 22:06:58 AM	elizabeth.pullan@radac...	ViewDashboards	["id": "b71c4ef-5680...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	20/02/2024 11:01:17 PM	elizabeth.pullan@radac...	ViewDashboards	["id": "5f1354e-b810...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	20/02/2024 9:37:12 PM	elizabeth.pullan@radac...	ViewReport	["id": "cfc61e1f6598...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	19/02/2024 9:04:56 AM	reca@radac...	AnalyzeByExternalApplication	["id": "7115929-225...
outlook.office365.com	7379fe2b-543a-450f-a22d-5d61e001f5f1		FALSE PowerBIAudit	10/02/2024 2:00:29 AM	Reca@RADACAD.onmicrosoft.co...	AnalyzeByExternalApplication	["id": "3250000-144...

**Figure 27-19.** The Power BI log output

As you can see, there is some generic information, and the main details are all in the AuditData field. The AuditData field is formatted as JSON. Let's remove all the other fields and only keep the AuditData (see Figure 27-20).



**Figure 27-20.** Removing all fields except for AuditData

Because the field is in JSON format, you can choose the Parse JSON option, as illustrated in Figure 27-21.

A screenshot of the Power Query Editor. The "AuditData" table is selected. The "Transform" tab is highlighted (Step 2). The "Parse" dropdown is open, showing options for XML and JSON (Step 3). The "JSON" option is highlighted (Step 4). A tooltip for the JSON option states: "Parse each cell value in the selected columns as a JSON document."

**Figure 27-21.** Using the Parse JSON option

JSON data will be as a record in every cell, and you can expand it to underlying columns, as depicted in Figure 27-22.

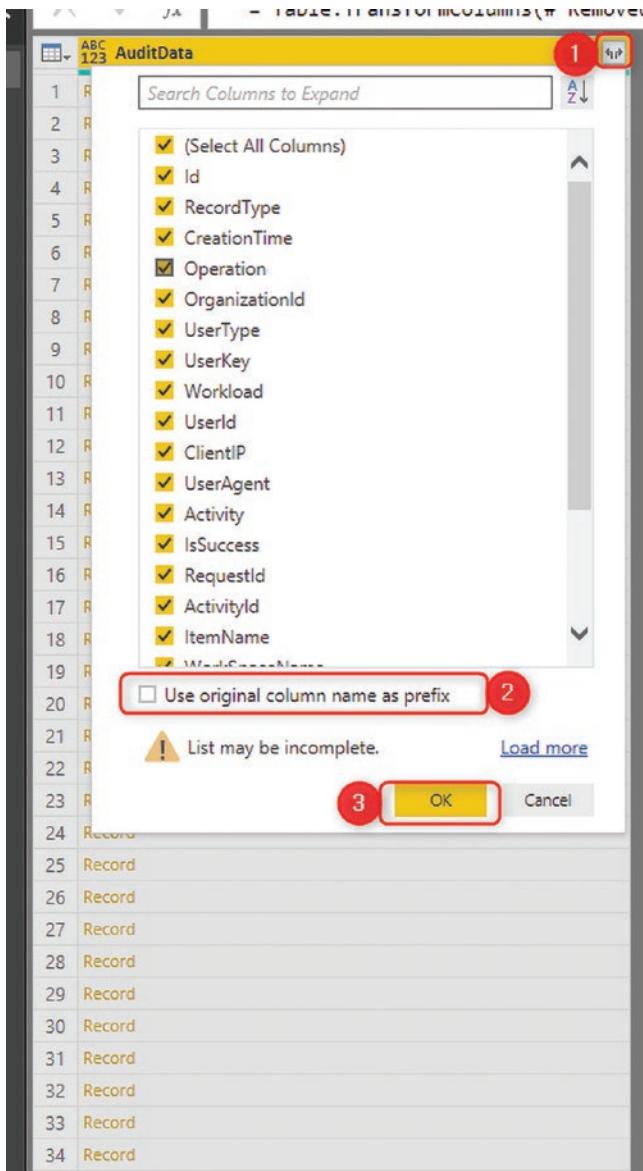


Figure 27-22. Expanding JSON data to underlying cells

Figure 27-23 shows all the Audit Log data.

ID	Ref	RecordType	CreatedTime	Operation	OrganizationId	UserType	UserKey	Workload
1	80578f96-9215-49e0-03a4-cc0f13509487	2018-02-26T08:08:30		ViewUsageMetrics	cc80986f-3c35-4d04-0098-cbe98900467e	2	10037FFE8C0B0193	PowerBI
2	21e0692b-6474-4984-88f6-7b342c056cf6	2018-02-25T08:08:37		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
3	b2b10ef8-0ad4-498c-a694-dae54126e413	2018-02-25T08:08:37		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
4	11bd820f-716-405b-adc5-5675420591bc	2018-02-25T08:20:02		ViewReport	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
5	5d5252cf-ea2d-448f-9264-054-04f107	2018-02-25T08:20:24		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
6	b55b655e-1655-4a31-9c57-f157a7ed0a1	2018-02-25T08:20:08:07		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
7	bfb39a8c-e163-447e-872-09820a55ebe	2018-02-25T08:20:08:45		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
8	1b11c0ff-7a0d-475f-a75b-2066c5e0aae	2018-02-27T04:08:22		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
9	a50e0e72-eb3d-4209-9943-0eabca5d8aee	2018-02-27T04:13:38		ViewUsageMetrics	cc80986f-3c35-4d04-0098-cbe98900467e	2	10037FFEB8D0B0393	PowerBI
10	c9990c0f-3f84-49e0-9955-383466525732	2018-02-27T05:07:06		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
11	44511156-3414-410e-a700-997a775e5743	2018-02-27T01:05:52		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
12	7a83cc40-9c18-489c-a87b-4814f1c19399	2018-02-27T00:07:32		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
13	49c24636-7184-48ef-8ec7-0f564ee76053	2018-02-21T28:03:53		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
14	f5d2529e-a413-4c11-af54-7be68fa9695	2018-02-21T22:08:38		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
15	8311e785-2820-4104-9242-ae90207af3	2018-02-21T21:04:35		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
16	e72d7ff0-2447-4107-8621-40712859e2b	2018-02-21T20:05:22		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
17	11a81d91-59e0-4e9c-9fcb-e0173a2d11b	2018-02-21T04:44:53		ViewReport	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
18	94e8ff8d-2494-4f3a-8056-5a7d4424969b	2018-02-21T04:05:13		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
19	2772e47-ca16-455f-93a8-c7389c729c9	2018-02-21T05:05:41		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
20	905c5d73-3c33-4184-9267-a0f154548512	2018-02-21T01:06:34		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
21	67f754ff-5683-4d56-98b1-76d730ff7ca	2018-02-21T00:06:38		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
22	5f3542e-8000-4448-8015-90729262968	2018-02-20T23:09:17		ViewDashboard	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
23	cfe46df-8594-4f4e-bee5-b892c6f71a3	2018-02-20T21:37:12		ViewReport	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEA6615AA3	PowerBI
24	7c33992b-4205-49a0-8233-7f55750e9c	2018-02-19T02:04:56		AnalyzeByExternalApplication	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEBFC78579	PowerBI
25	ndc4931c-data-4ea7-0484-54fc-20097e7	2018-02-18T02:00:39		AnalyzeByExternalApplication	cc80986f-3c35-4d04-0098-cbe98900467e	0	10037FFEB8D0B0193	PowerBI

Figure 27-23. Displaying all the Audit Log data

The remaining steps involve visualizing this data. Figure 27-24 shows some sample visualizations I created. This report shows all users and their operations on certain items (report, dataset, dashboard, and workspaces).

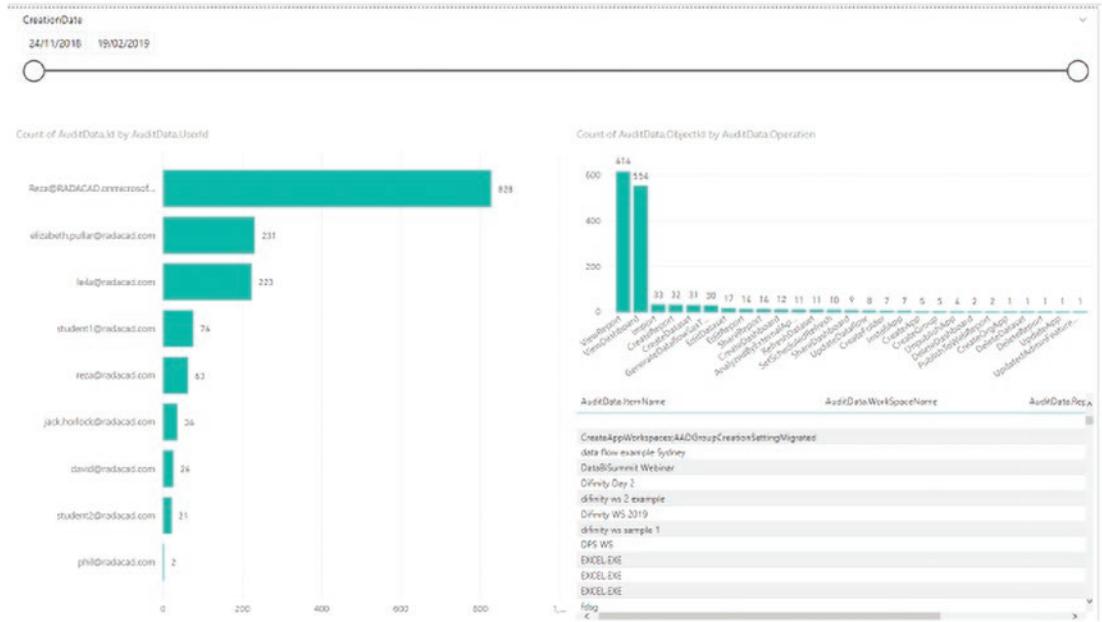
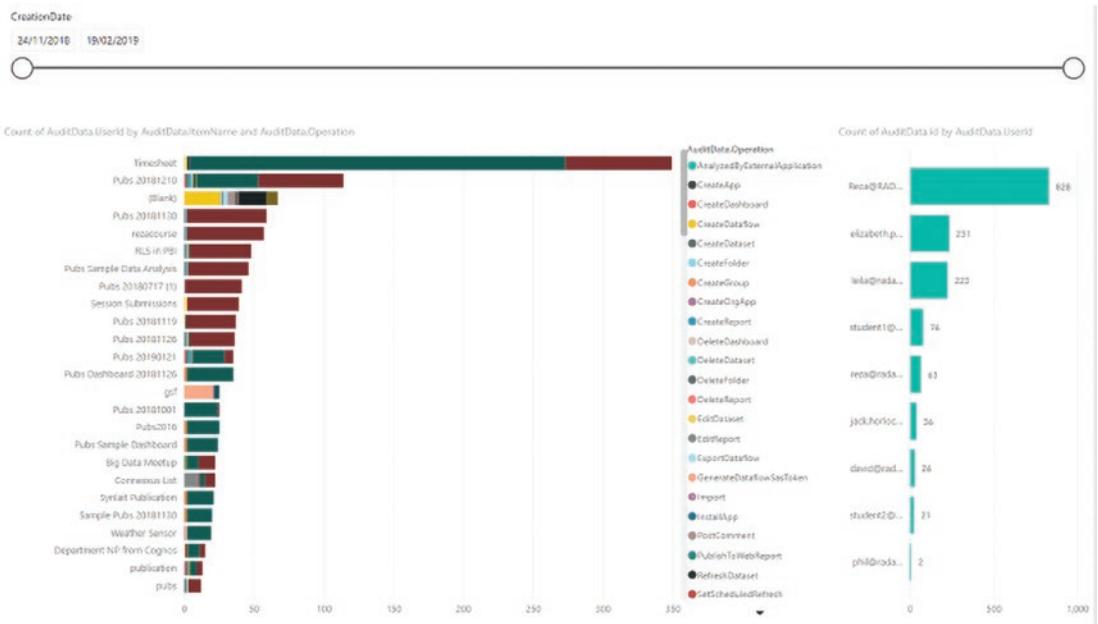


Figure 27-24. Sample data visualization showing all users and their activities

Figure 27-25 shows all the objects and the operations on them with the list of users.



**Figure 27-25.** Sample data visualization showing all objects and operations performed on them

## Summary

Yes, you can create your own custom audit and usage metrics reports across the entire tenant from all workspaces. You can leverage PowerShell to export the Audit Log into CSV and then use that as the source for the Power BI report. This process can be automated using SSIS, Azure Data Factory, or Task Scheduler. This information is also available through Power BI Helper. Also, PowerShell can be useful for getting much other information from the service.

## CHAPTER 28



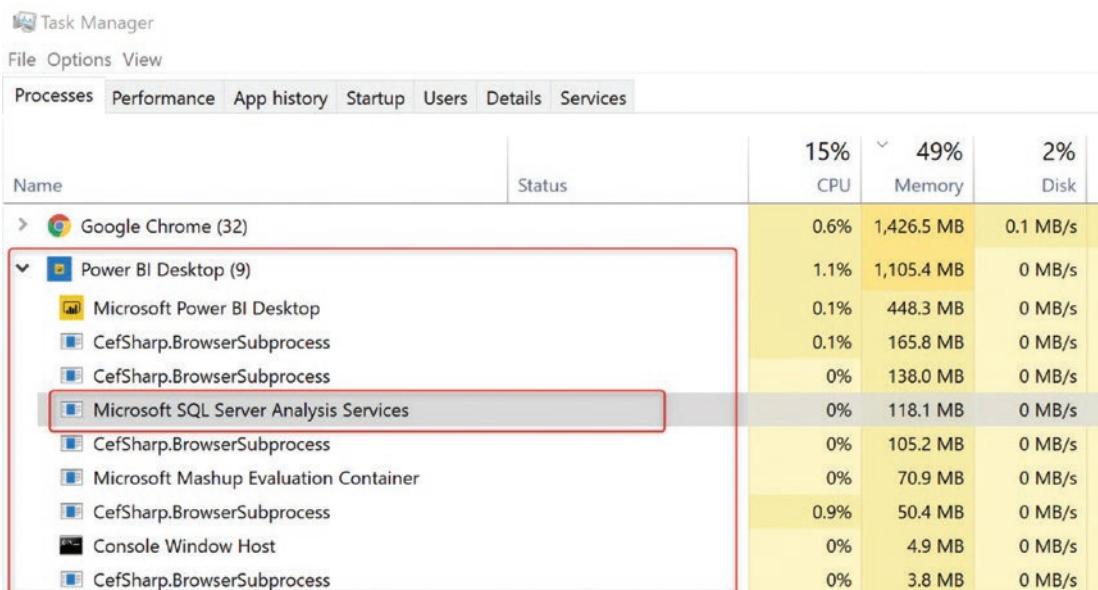
# XMLA Endpoint

The XMLA Endpoint is still too technical for many Power BI report developers. Many people come to me asking what exactly an XMLA endpoint is and what it's used for. This chapter explains the XMLA Endpoint in detail. Let's dig in.

## Behind the Scenes of a Power BI Dataset

To understand what an XMLA Endpoint is and what it does for you, you need to understand the backstage of a Power BI dataset. By backstage, I mean what is behind the Power BI report that you see.

A Power BI report is a visualization element connected to an in-memory dataset behind the scenes. (I am talking about the most common method of using Power BI, which is Import Data.) The in-memory dataset behind the scenes has all the data loaded into the memory, with all calculations, relationships, and connections to the data source. When you open a \*.PBIX file, behind the scenes, there are two elements—a report (the visualization part) and a dataset (the data model). This separation is visible in the Power BI Desktop app resource details, accessed from the Task Manager (see Figure 28-1).

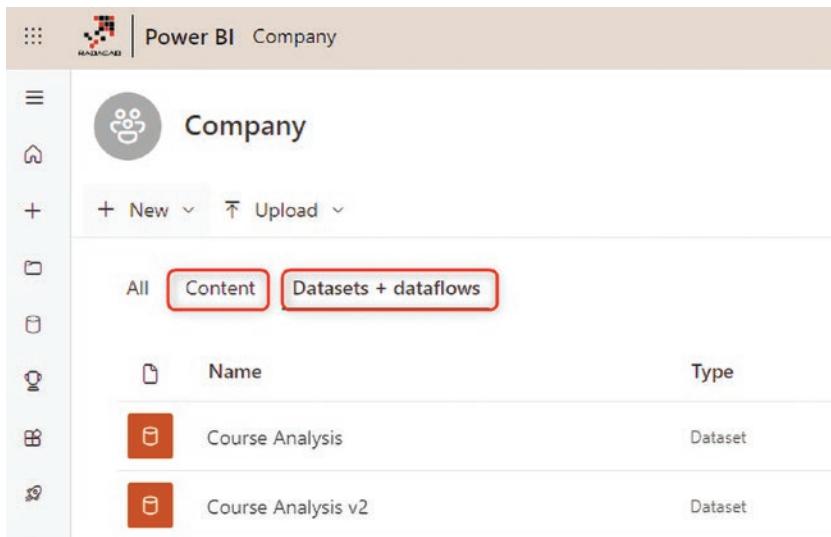


**Figure 28-1.** Viewing resource details in the Task Manager

As you can see in Figure 28-1, a Microsoft SQL Server Analysis Services task is running under the Power BI Desktop list. A Power BI report stores its data in memory, which is managed by the Microsoft SQL Server Analysis Services Engine. (Let's call it by its shorter, more familiar, name—SSAS.) Even if you run the Power BI Desktop on a machine that doesn't have SSAS installed, you will see this service because the Power BI Desktop automatically installs a version of SSAS.

A Power BI dataset is hosted through a SQL Server Analysis Services Engine.

When you publish your \*.pbix file to the Power BI Service, you will also see this separation of the data model (the dataset) and the visualization (the report), as shown in Figure 28-2.



The screenshot shows the Power BI Service interface. At the top, there is a navigation bar with icons for Home, Reports, Datasets, and Dataflows. The main area is titled "Company". Below the title, there are buttons for "+ New" and "Upload". There are two tabs: "Content" (which is selected and highlighted with a red box) and "Datasets + dataflows". The "Content" tab displays a list of datasets:

	Name	Type
	Course Analysis	Dataset
	Course Analysis v2	Dataset

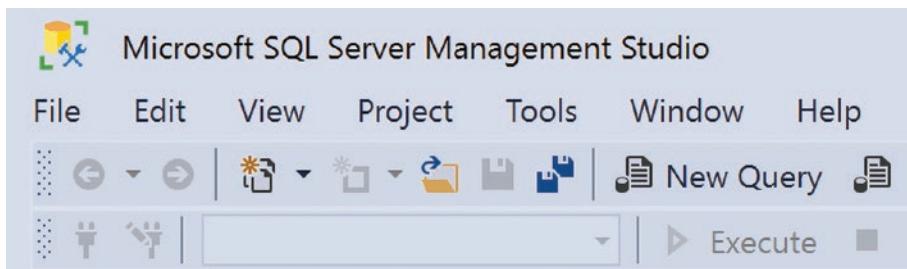
**Figure 28-2.** The separation of the dataset and the report (content) in the Power BI Service

When you host your report (or publish it) in the Power BI Service, the dataset will be managed by a version of SSAS installed on a cloud machine that you don't see (or, better to say, a version of Azure Analysis Services).

## SSAS Is More Than What You See

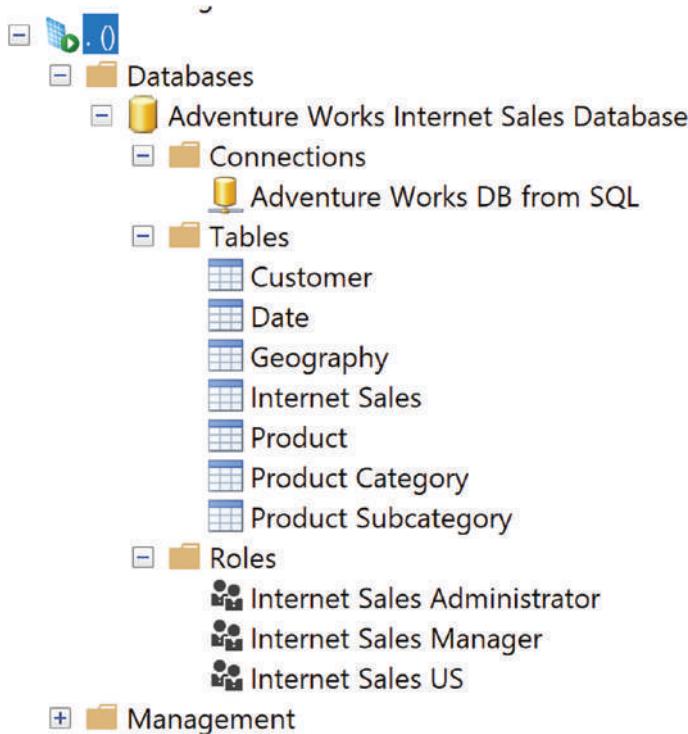
Now that you know a Power BI dataset is an SSAS model behind the scenes, the next question is, what is the point? Or what is the benefit of it? To understand that, let's look at SSAS more closely.

SSAS is a modeling engine in Microsoft SQL Server. It's more than 20 years old and is a mature technology. SSAS is a server-side modeling technology. The model is hosted on a server, and the client tools can work with it. Through many years, the two most common client tools to work with SSAS are SQL Server Data Tools (SSDT) and SQL Server Management Studio (SSMS), the latter of which is shown in Figure 28-3.



**Figure 28-3.** The SQL Server Management Studio UI

These tools not only build the model in SSAS but also manage it. When it comes to management, that includes monitoring, controlling, backing up, and restoring the model, and many other features. Figure 28-4 shows a view of an SSAS database (model) from SSMS.



**Figure 28-4.** Viewing an SSAS database (model) in SQL Server Management Studio

SSAS models can be monitored using queries and commands called Dynamic Management Views (DMV). For example, the command in Figure 28-5 shows all the users querying or working with this SSAS model.

```
Select * from $System.discover_sessions
```

SESSION_ID	SESSION_TYPE	SESSION_USER_NAME	SESSION_DURATION	SESSION_START_DATE	SESSION_END_DATE	SESSION_STATUS	SESSION_ERROR_CODE	
E9AC0...	798	AzureAD\RezaRad	Advent...	22	6/04/20...	187863	6/04/20...	6/04/20... 0
EB9134...	827	AzureAD\RezaRad	Advent...	3	6/04/20...	48806	6/04/20...	6/04/20... 15
29C2DF...	831	AzureAD\RezaRad	Advent...	3	6/04/20...	48389	6/04/20...	6/04/20... 0

**Figure 28-5.** Executing a command that shows all users querying or working with the SSAS model

---

SQL Server Analysis Services is a server-side technology that gives you a lot of details about the model. It can be monitored through client tools using scripts and commands such as Dynamic Management Views.

---

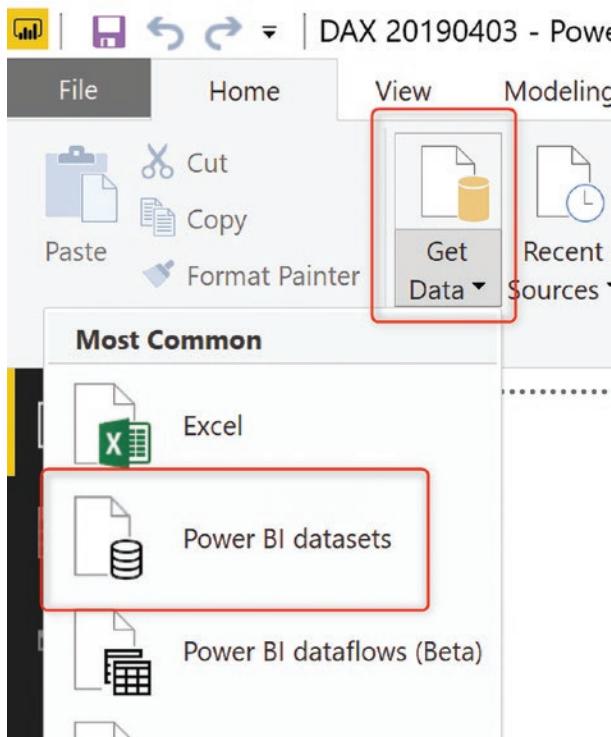
## Can I Access the SSAS Model of My Power BI Dataset?

You learned so far that your Power BI dataset is an SSAS model behind the scenes. You also learned that you can control and monitor SSAS models using client tools. However, when the Power BI dataset is hosted in the Power BI Service, how can you connect to that SSAS model? How can you control, manage, or monitor it? Or, even more importantly, why is it important to do it? See Figure 28-6.

---

If you can connect to the SSAS model of your Power BI dataset, you can connect to the data model directly. You can see how many users are using it. You can see what processes take longer and which ones are slow. You can leverage all those monitoring features to build a better model moving forward. There are two common ways to connect to the Power BI dataset hosted in the service—the Power BI Desktop (using Get Data from the Power BI dataset) and Excel (using Analyze in Excel).

---



**Figure 28-6.** Connecting to a Power BI dataset hosted in a service

Both of these tools are reporting tools. You can (somehow) run monitoring queries from these tools, but these are not built for that purpose. You will have better control if you can access this through other client tools. The good news is that now you can! That's where XMLA Endpoint comes in to play.

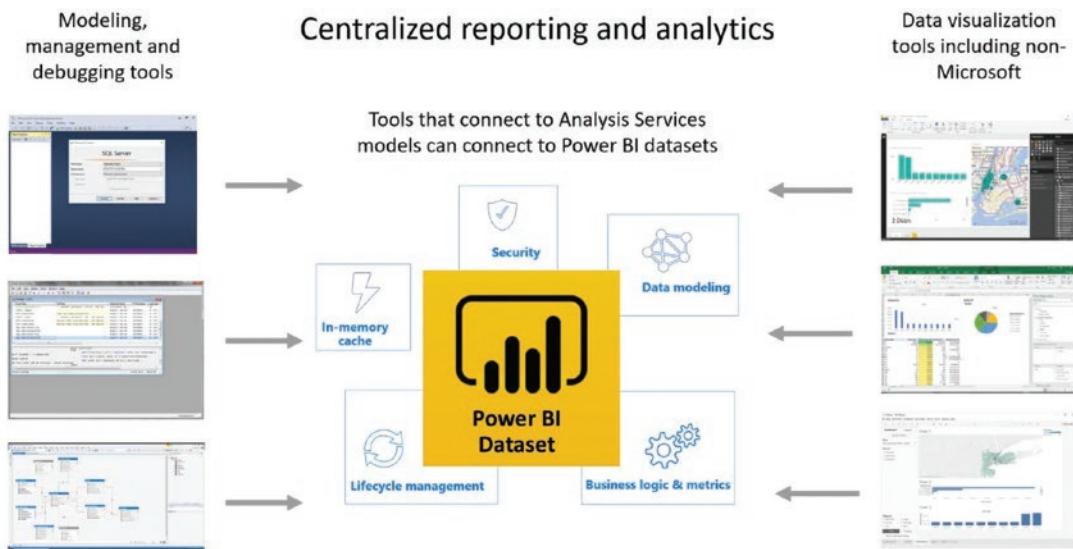
## XMLA Endpoint

XMLA stands for Extensible Markup Language for Analysis. Now is a good time to explain what an XMLA Endpoint is. An XMLA Endpoint creates a connectivity channel for other tools and services (which can be third-party tools) to the SSAS model. An XMLA Endpoint is available for SSAS models hosted through SQL Server and has been used for a long time. You can use a tool such as SSMS and connect to the local SSAS engine. Without an XMLA Endpoint, access to the SSAS model will be very limited and probably unsupported.

An XMLA Endpoint creates a connectivity channel for other tools and services (which can be third-party tools) to the SSAS model.

The good news is that XMLA Endpoints are available for Power BI datasets hosted in the Power BI Service. It means you can use any client tools that support XMLA connectivity to connect to Power BI datasets. When I say any, I mean it. In addition to SSMS or SSDT, or Microsoft SQL Server client tools, you can use third-party tools, such as DAX Studio and Power BI Helper, as well as Tableau! Yes, you read that right. You can use Tableau to connect to a Power BI dataset hosted in the Power BI Service and then have your visualization in Tableau.

Figure 28-7 says it all. All of these tools can be used to connect to a Power BI dataset hosted in the service.



**Figure 28-7.** The various tools that can be used to connect to a Power BI dataset hosted in the service

In other words, the Power BI dataset is not just for Power BI tools; it can be the source for any other tools that have XMLA connectivity support—Tableau, SSMS, Power BI Helper, and more.

## How Does an XMLA Connection Work?

There are two types of connections for an XMLA Endpoint—Read and Read/Write. The XMLA functionality is limited to Premium licensing. The Read connection enables you to read data from the dataset, which can be useful for monitoring and querying. The Read/Write connection enables you to change the dataset, such as adding calculation groups or object-level security to the Power BI dataset.

## The XMLA Endpoint URL

If you go to a Premium capacity allocated workspace, under the Premium Capacity, you will see the XMLA Endpoint connection URL (see Figure 28-8).

The screenshot shows the 'Settings' page in the Power BI Admin Portal. The 'Premium' tab is selected. Under 'License mode', 'Premium per user' is chosen. Under 'Default storage format', 'Small dataset storage format' is selected. A red box highlights the 'Workspace Connection' section, which displays the URL 'powerbi://api.powerbi.com/v1.0/myorg'. A green 'Copy' button is also highlighted with a red box.

**Figure 28-8.** The XMLA Endpoint connection URL

The format is as follows:

```
powerbi://api.powerbi.com/v1.0/myorg/<workspace name>
```

The workspace name can have spaces and is case sensitive. For example:

```
powerbi://api.powerbi.com/v1.0/myorg/Reza SAMPLE workspace
```

## Admin Control under Capacity Configuration

Your capacity admin should enable you to use an XMLA Endpoint. This is possible from the Admin Portal by choosing Capacity Setting, selecting the capacity, and then choosing Workloads. If you are using Premium Per User licensing, there is a place to set this under the Admin Portal from Premium Per User (see Figure 28-9).

The screenshot shows the Power BI Admin portal interface. On the left, there's a sidebar with various icons and a list of navigation items. The main area is titled "Admin portal" and contains sections for "Premium Per User" settings. One section, "XMLA Endpoint", is highlighted with a red box and shows a dropdown menu with four options: "Read Only", "Off", "Read Only" (which is selected), and "Read Write".

Power BI Admin portal

Admin portal

Tenant settings

Usage metrics

Users

Premium Per User

Audit logs

Capacity settings

Refresh summary

Embed Codes

Organizational visuals

Azure connections

Workspaces

Custom branding

Protection metrics

Featured content

Premium Per User

Auto Refresh

Automatic page refresh

On

Minimum refresh interval

5 Minutes

Change detection measure

On

Minimum execution interval

30 Seconds

Dataset workload settings

XMLA Endpoint

Read Only

Off

Read Only

Read Write

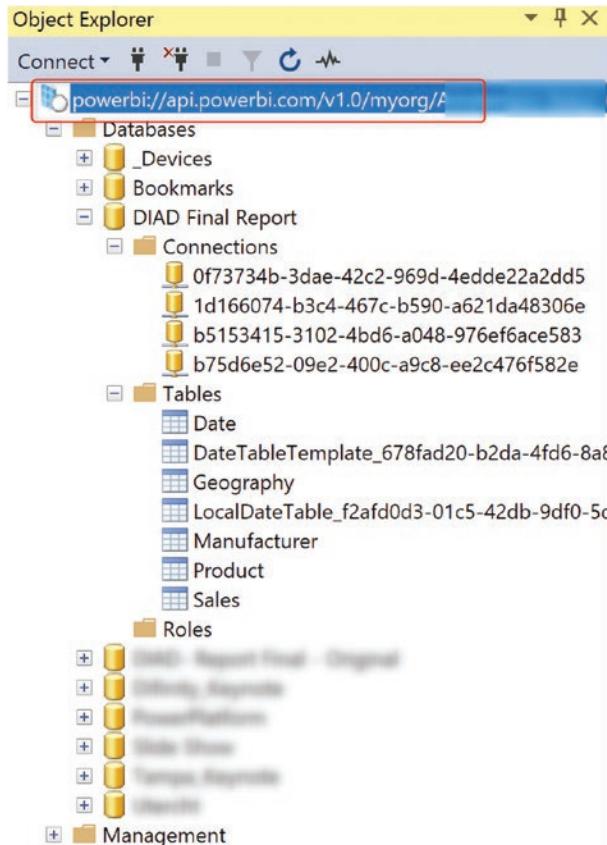
Figure 28-9. The XMLA Endpoint connection type in the Admin Portal

## Client Tools

You should have a client tool for this type of connection. You can use a tool such as SSMS. However, you need SSMS 18.0 RC1 or above, which can be downloaded from [docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-2017#ssms-180-rc1](https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-2017#ssms-180-rc1). The client tools must have installed the libraries listed at [docs.microsoft.com/en-us/azure/analysis-services/analysis-services-data-providers](https://docs.microsoft.com/en-us/azure/analysis-services/analysis-services-data-providers) to be able to connect.

## Sample Connection

Figure 28-10 shows a sample of the connection created using an XMLA Endpoint through SSMS.



**Figure 28-10.** Example connection created using an XMLA Endpoint through SSMS

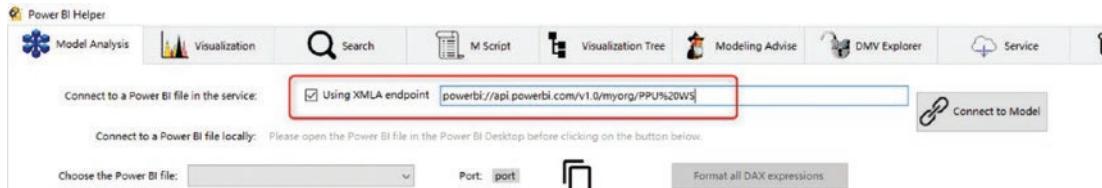
As an example, Figure 28-11 shows a query that lists how many users are using this dataset with DMVs.

Select * from \$System.discover_sessions												
SESSION_ID	SESSION_TYPE	SESSION_USER_NAME	SESSION_STATUS	SESSION_DURATION	SESSION_CREATED_DATE	SESSION_MODIFIED_DATE	SESSION_LAST_ACTIVITY_DATE	SESSION_LAST_ACTIVITY_TIME	SESSION_LAST_ACTIVITY_IP	SESSION_LAST_ACTIVITY_BROWSER	SESSION_LAST_ACTIVITY_OS	SESSION_LAST_ACTIVITY_DEVICE
E9AC0...	798	8	AzureAD\RezaRad	Advent...	22		6/04/20...	187863	6/04/20...	6/04/20...	0	
EB9134...	827	11	AzureAD\RezaRad	Advent...	3		6/04/20...	48806	6/04/20...	6/04/20...	15	
29C2DF...	831	12	AzureAD\RezaRad	Advent...	3		6/04/20...	48389	6/04/20...	6/04/20...	0	

**Figure 28-11.** Example query showing how many individuals are using this dataset via DMVs

## The Power BI Helper

The Power BI Helper can also connect to an XMLA Endpoint, as shown in Figure 28-12.



**Figure 28-12.** The Power BI Helper connecting using an XMLA Endpoint

It can also give you documentation and information about the datasets in this workspace, as shown in Figure 28-13.

The screenshot shows the Power BI Helper interface with the 'Model Analysis' tab selected. It displays three main sections: Tables, Columns, and Measures. The Tables section shows a list of tables with columns for Name, Description, Storage Mode, Source, and Is Hidden. The Columns section shows a list of columns with filters and export options. The Measures section shows a list of measures with descriptions and checkboxes for All and Description. At the bottom, there are 'fx Expression:' and 'fx Measure Expression:' input fields.

Name	Description	Storage Mode	Source	Is Hidden
Customers		Import	let \$source = Sq...	
Employees		Import	let \$source = Sq...	
Order_Details		Import	let \$source = Sq...	
Orders		Import	let \$source = Sq...	
Shippers		Import	let \$source = Sq...	

Name	Table Name	State	Data Category
RowNumber-2642079B-1795-4f74...	Customers	Ready	Int
CustomerID	Customers	Ready	Str
CompanyName	Customers	Ready	Str
ContactName	Customers	Ready	Str
ContactTitle	Customers	Ready	Str
Address	Customers	Ready	Str
City	Customers	Ready	Str

Name	State
Freight YTD	Ready

**Figure 28-13.** Using the Power BI Helper to view documentation and information about datasets in the workspace

## Summary

Using XMLA Endpoints, you can use client tools to control, manage, and monitor Power BI datasets in the service. Also, a Power BI dataset can be used as the data model for other visualization tools, such as Tableau. XMLA Endpoints can be read-only for querying and logging purposes or read-write for making changes to the model.

## CHAPTER 29



# Dashboard and Report Sharing

Power BI provides multiple ways to share content with users. Each sharing method has its pros and cons and can be used for specific scenarios. Some sharing methods can be used together to build a framework for sharing. This chapter discusses the most basic way to share Power BI content. This method is called dashboard (or report) sharing. Dashboard sharing is the easiest way to share; however, it may not always be the best way. In this chapter, you'll learn how this method works, its pros and cons, and the scenarios for using it.

## Power BI Content Owner

Before going through dashboard sharing, you need to understand how content security works in Power BI. When you publish a \*.pbix report to the Power BI Service, especially when you publish it under My Workspace, no one else can see or access your report. You can decide with whom you want to share this report.

All Power BI content (reports, dashboards, and datasets) has an owner; the content owner is the person who created and published that content to Power BI. The owner has full access to the content. The owner can share that content with others as needed.

## How Does Dashboard Sharing Work?

Dashboard or report sharing, as the name implies, is based on a dashboard. You can only share a dashboard or report using this method, not a dataflow or a datamart. Consider that you have a dashboard like the one shown in Figure 29-1, and you want to share it. Note the share link in the top-right corner of the dashboard.

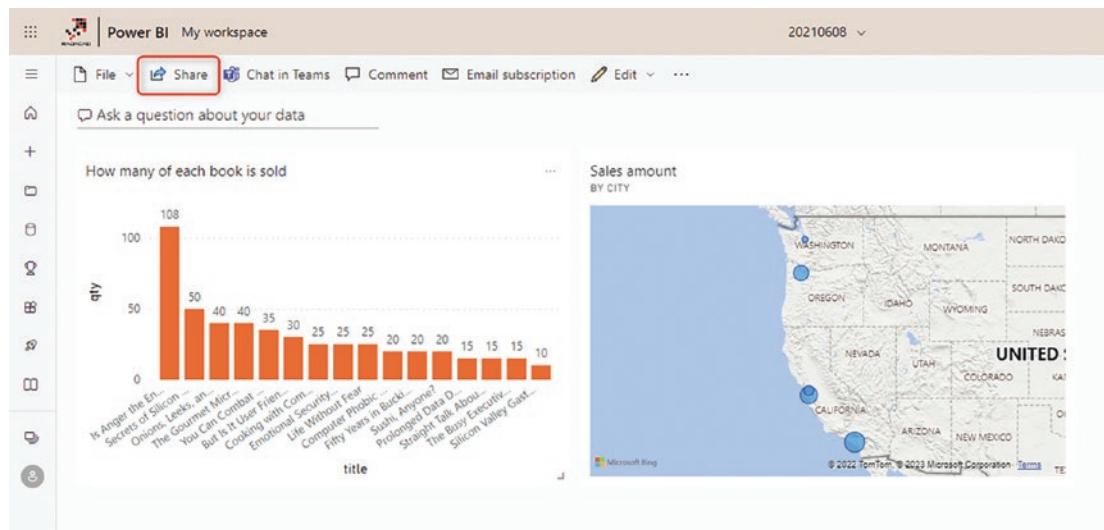
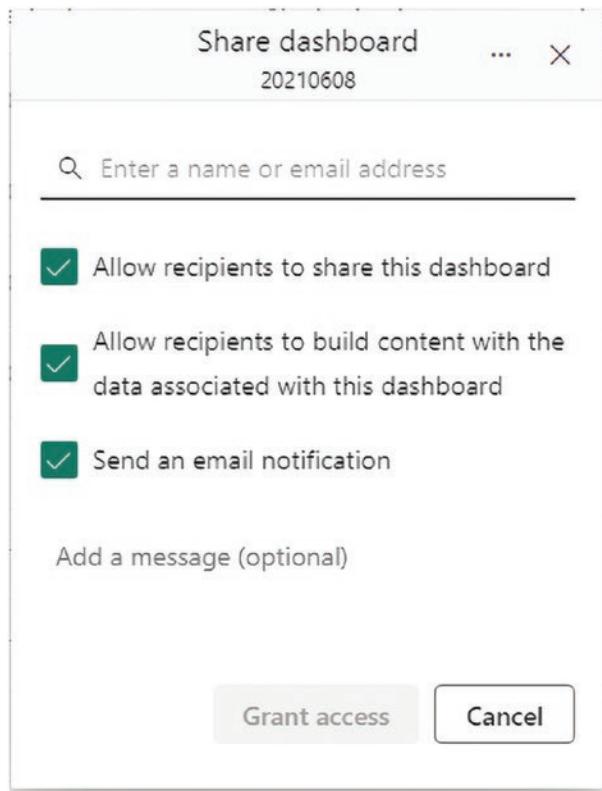


Figure 29-1. Sharing from a dashboard page in Power BI

Dashboard sharing has a few options you can set and is very simple to configure. You need to add the email address of the people with whom you want to share the report. You can also write messages to them. If you are sharing a report or a dashboard, you might get two slightly different options. If you share a dashboard, you will get an option like the one shown in Figure 29-2.

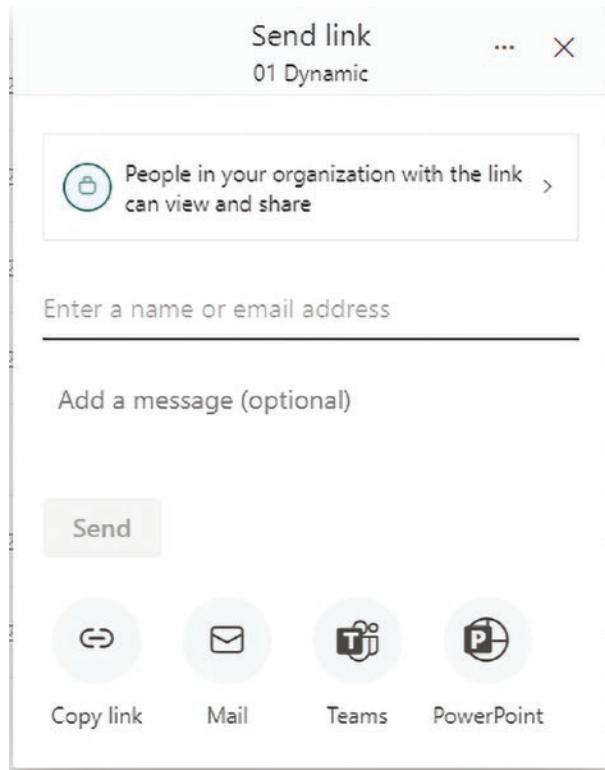


**Figure 29-2.** Sharing dashboard in Power BI

There are a few options to set:

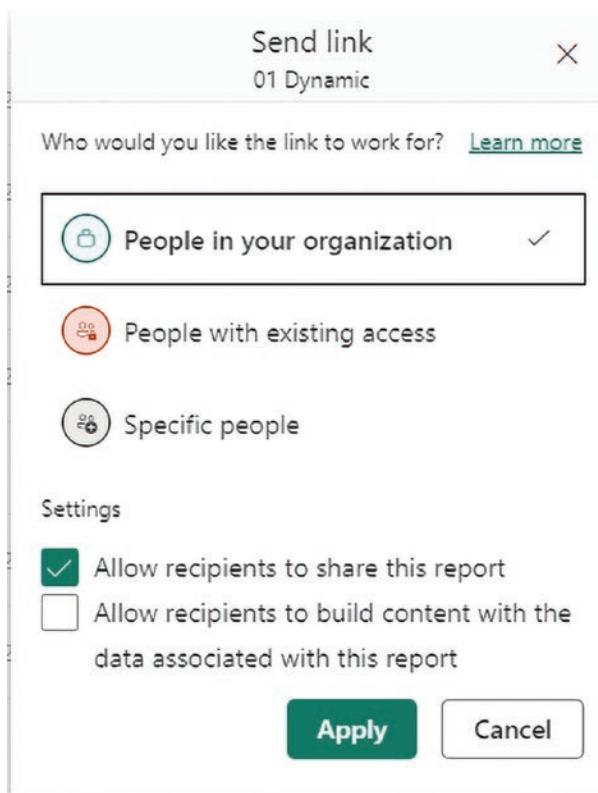
- Allow recipients to share your dashboard
- Allow recipients to build content with the data associated with this dashboard
- Send email notifications to recipients

And at the top, you can enter the email of the people you share this dashboard with. For report sharing, the options are also similar, with a small difference (see Figure 29-3).



**Figure 29-3.** Report sharing in Power BI

By default, as you can see in Figure 29-3, the option called People in Your Organization with the Link Can View and Share is selected. This can be modified, but by clicking it, you have the options shown in Figure 29-4.



**Figure 29-4.** Changing the settings of report sharing in Power BI

You can choose whom to share the report with (the entire organization or specific people), and you can choose the access level of the audience to view, or plus reshare, or even plus build. There are also options to share in multiple ways—Copy link, Mail, Teams, and PowerPoint.

After configuring this, click the Share button. The recipient will immediately have access to the report. If you select Send Email Notification to Recipients, they will receive an email. Otherwise, they get a notification in Power BI. When users log in to the service (or in the mobile app for Power BI), they will find this dashboard or report in the Shared with Me section, which is in the Browse menu (see Figure 29-5).

Name	Type	Shared	Owner
[Redacted]	Report	8/13/19, 8:36:58 AM	sample DGroup WS
[Redacted]	Report	1/2/20, 4:59:02 PM	Public Reports
[Redacted]	Report	1/20/20, 6:38:51 AM	General Dataset Wor...
[Redacted]	Report	1/11/21, 1:35:59 PM	Public Reports

**Figure 29-5.** The Shared with Me section in the Power BI Service

The recipient can also access the report or dashboard through the link. An important point here is that those with whom the report is shared need a Power BI Pro or PPU account to see the content (this is one of the limitations of this method of sharing).

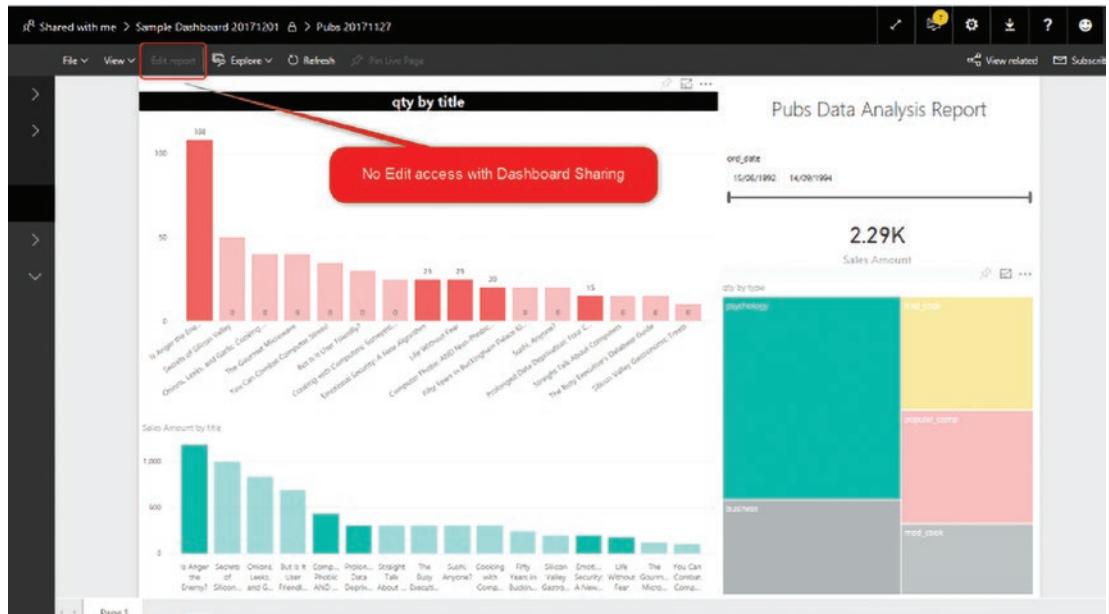
## Three Levels of Access

With dashboard (or report) sharing, users have three levels of access—Read, Read and Reshare, and Build. If you give them access without selecting the Allow Recipients to Share Your Dashboard option, this access is Read. If you choose the option mentioned previously, the access is Read and Reshare. To provide Build access, you need to select Allow Recipients to Build Content with the Data Associated with this Dashboard.

Build access provides the ability to connect live to the dataset. This live connection can be done through the Power BI Desktop, through a new report inside the Power BI service, or using Analyze in Excel. The live connection to a dataset in Power BI is a great step toward a multi-layer architecture for Power BI development.

## Manage Permissions

Another way to set access is through Manage Permissions in the dashboard, report, or dataset. If you share a dashboard, by default, the report and the dataset will also be shared as read-only for users. Users can click the dashboard and go to the report; they can interact with the report quickly. However, they cannot edit the report (see Figure 29-6). Access to edit reports cannot be provided through this method.



**Figure 29-6.** Restricting users from editing a report

To manage permissions on every item (dashboards, reports, and datasets) individually, go to Manage Permissions in the More options of the object, as depicted in Figure 29-7.

The screenshot shows the Power BI service interface with a list of datasets. A context menu is open over the first dataset, showing options like 'Analyze in Excel', 'Delete', 'Quick insights', 'Save a copy', 'Settings', 'View usage metrics report', 'View lineage', 'Create paginated report', and 'Manage permissions'. The 'Manage permissions' option is highlighted with a red box.

Name	Type	Access
00 Static	Report	R
00 Static		R
01 Dynamic		R
01 Dynamic		R
20210608		R
20210608		R
20210608		R
20221115		R

**Figure 29-7.** Managing permissions on a Power BI report, dashboard, or dataset

Manage Permissions will show you a detailed access list regarding the dashboard, reports, and datasets. You will see related reports and datasets on the left side of the Manage Permissions section (see Figure 29-8). You can click the report.

The screenshot shows the 'Manage permissions' section for a dataset named '20210608'. On the left, there's a sidebar with 'Related content' showing 'Dashboards' (20210608) and 'Workbooks' (20210608). The main area shows a table of access links:

Links	Who has Access	Permissions	Creator	Email Address
<a href="https://app.powerbi.com/links/27edf5471fd1c030fb5e...">Links</a>	People in your organization	Read, Write	Reza Rad	Reza@RACACAO.onmicrosoft.com
<a href="https://app.powerbi.com/links/1f8eef1d1cbe4e05e6b...">Links</a>		Read	Reza Rad	Reza@RACACAO.onmicrosoft.com

**Figure 29-8.** Related content in the Manage Permissions setting

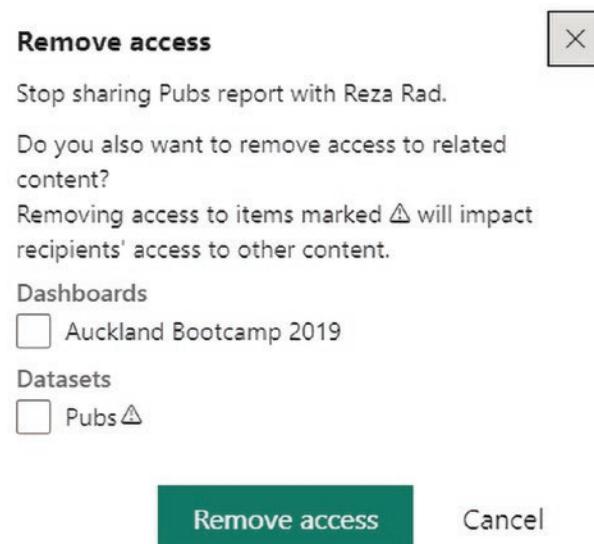
Access can be seen through links or direct access (directly authorized through Manage Permissions). You will see the permission specified for that object by clicking a report or dataset. And you can change it. For example, the `reza@radacad.com` user has access as Read to the report in Figure 29-9 (because I shared this dashboard with him, the report sharing happened automatically after that). You can remove that access by clicking more options.

The screenshot shows a table of links with columns: Links, Who has Access, Permissions, Creator, and Email Address. Two rows are visible. A tooltip is overlaid on the second row's 'More' button, containing options: Copy link, Delete, Manage Access, and See shared views. The 'Manage Access' button is highlighted with a red box.

Links	Who has Access	Permissions	Creator	Email Address
<a href="https://app.powerbi.com/reports/27zHRSwsA7cd+cc010d6...">https://app.powerbi.com/reports/27zHRSwsA7cd+cc010d6...</a>	People in your organization	Read, Refresh	Reza Rad	Reza@RADACAD.onmicrosoft.com
<a href="https://app.powerbi.com/reports/1TratoHuhL7ctidzcc0fEd6...">https://app.powerbi.com/reports/1TratoHuhL7ctidzcc0fEd6...</a>	⋮	Read	Reza Rad	Reza@RADACAD.onmicrosoft.com

**Figure 29-9.** Changing access levels through Manage Permissions

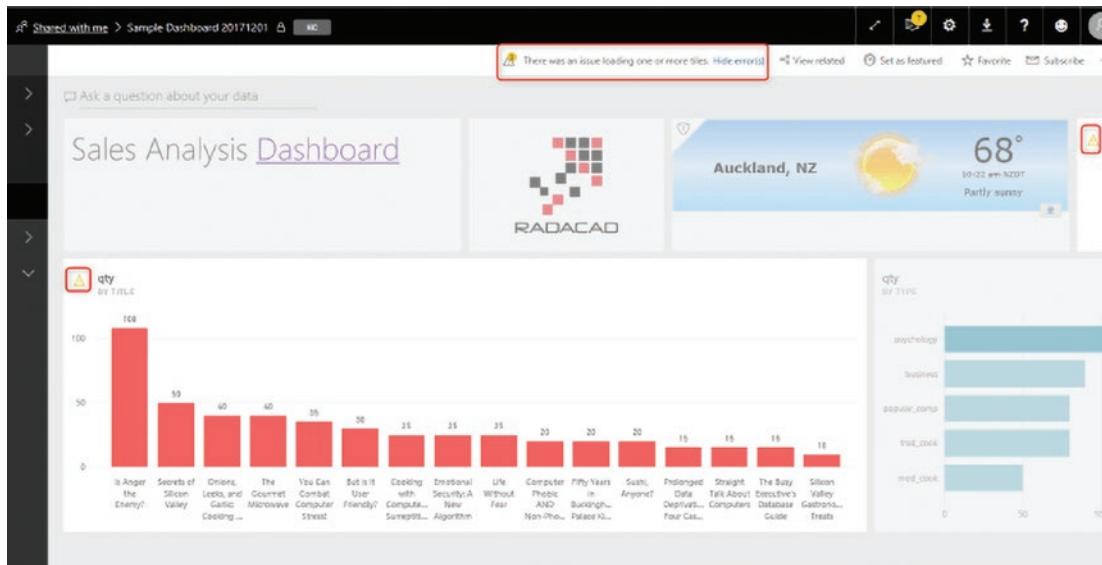
You will see the Remove Access window, which asks whether you want to remove access to some of the related content (see Figure 29-10).



**Figure 29-10.** Removing access to the related items in Power BI

If you remove access to other items, you should be careful because that item might be used in multiple other objects. For example, if you remove access to the dataset, that dataset might be used in multiple reports.

If you shared a dashboard with a user but removed access to the report or dataset, the user will see the error message for tiles coming from that report when logged in and accessing the dashboard (see Figure 29-11). Users cannot drill into the report because they don't have access.



**Figure 29-11.** A dashboard shared with a user, but with access to the report removed

Dashboard or report sharing, like many other Power BI methods, is a paid feature. The account sharing the content should be a Power BI Pro account (Or PPU), and people using the shared content should be part of the paid account Power BI Pro account or PPU. Free users cannot leverage content shared with this method

## Advantages of Dashboard Sharing

Dashboard (or report) sharing is the most basic way to share content in Power BI. This method is quick and easy to set up. The ability to share information very quickly makes this method the most common way to share for testing.

---

If you created Power BI content and want to share it with others easily for testing purposes, dashboard sharing is a good option.

---

## Disadvantages of Dashboard Sharing

Dashboard sharing is simple; however, it has many drawbacks, which make it hard to use in production. I do not recommend using this method to share Power BI content with users in a production environment because of the reasons explained in the next sections.

## No Edit Access

With dashboard sharing, you cannot specify edit access. For end users, you never want to give edit access; however, if you are working with a team of developers and want to provide them with access to edit the content, you cannot do that with dashboard sharing. You have to use other sharing methods, which are explained in the next few chapters.

## Share Objects One at a Time

You can only share one dashboard at a time. What if you wanted to share hundreds of dashboards? You must go to each dashboard and share items individually. Sharing every dashboard would add a lot of maintenance overhead to your work. The best method is to have the contents under a group and share it all with others at once.

## Licensing for a Large User Base

If you have thousands of users, this method is expensive. A Power BI Premium capacity can be set at the workspace level, and then free Power BI users can consume the content using Power BI apps, which is a more cost-effective way to share content.

## Summary

Dashboard or report sharing is straightforward. It has three access levels—Read, Read and Reshare, and Build. You can use this method efficiently for test scenarios. When you want to share a dashboard with a user for testing, dashboard sharing is a good option.

Dashboard sharing, however, has some disadvantages. There is no edit access. Also, if you want to share multiple items, you have to go to each dashboard and share them individually. Because of these two significant limitations, dashboard sharing is never used in the development or production environment of Power BI implementation. Other methods, which I explain in the next few chapters, address these limitations.

## CHAPTER 30



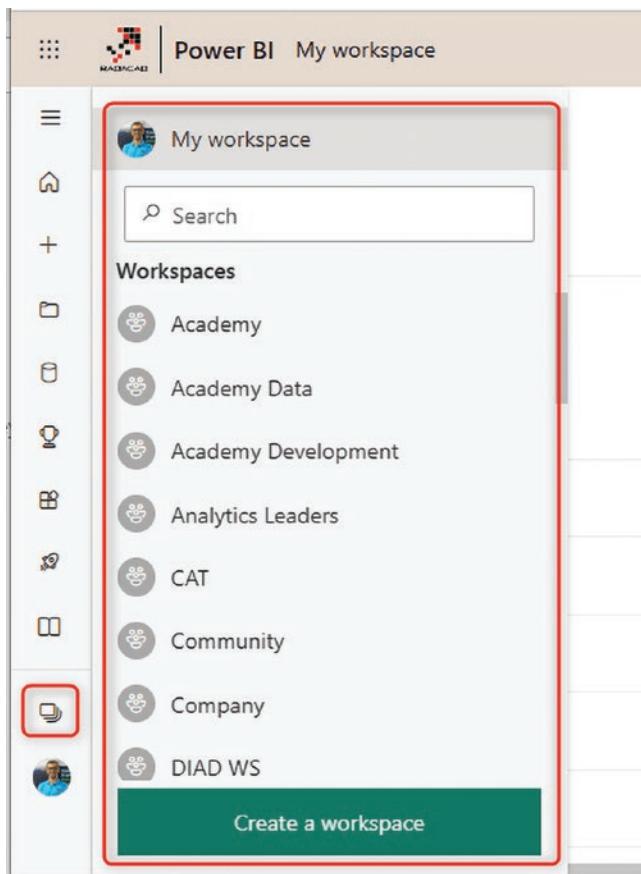
# Power BI Workspaces

Workspaces provide another way of sharing Power BI content with other people. The benefit of this sharing approach is that you can share content with a group of people and create a collaborative development environment, whereby everyone can access the content. This chapter explains how to share workspaces, the limitations and advantages of doing so, and how this method is different from dashboard sharing. By the end of this chapter, you will understand which scenarios are suitable for this method of sharing.

## What Are Power BI Workspaces?

Workspaces are shared environments. You can have multiple Power BI content in a workspace. One workspace can have hundreds of dashboards, reports, and datasets. Certain objects in the Power BI ecosystem can only exist within the context of a workspace, such as dataflows and datamarts. You can add people (or Power BI accounts) to the workspace and allow them to edit or read the content. In other words, workspaces are like shared folders containing the Power BI content in your organization.

One account may be part of multiple workspaces (see Figure 30-1) or various accounts and have access to one workspace. Everyone has a workspace named My Workspace. This is similar to the My Documents folder on your machine. My Workspace should never be used to share content with others, except for testing, because it is your personal workspace.



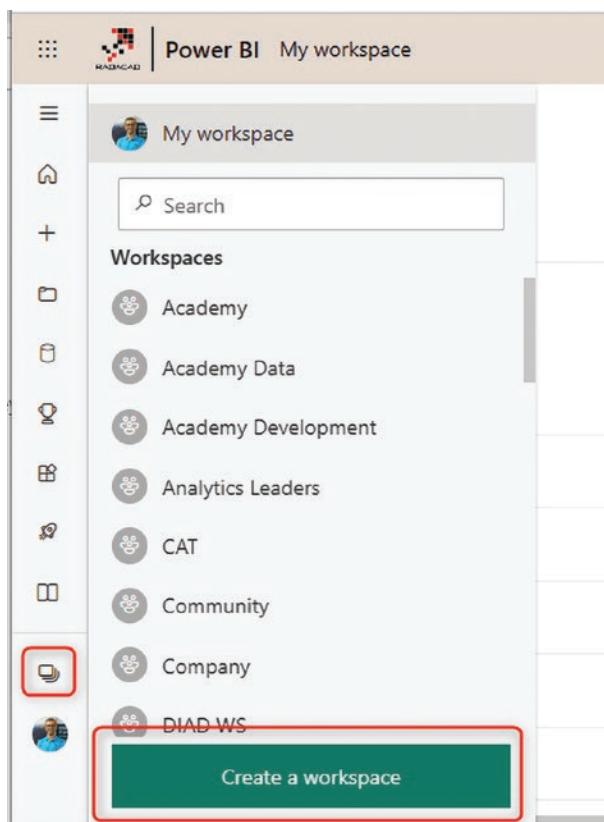
**Figure 30-1.** Workspaces in the Power BI Service

If you want to share content with others, your starting point is to create another folder, which in Power BI terminology, is called a workspace. Workspaces are also called App Workspaces and Organizational Workspaces, because you can create an app based on them and they are part of the organization's tenant.

Workspaces are best used as collaborative environments to share content between people on a team. Let's now look at how you can use workspaces.

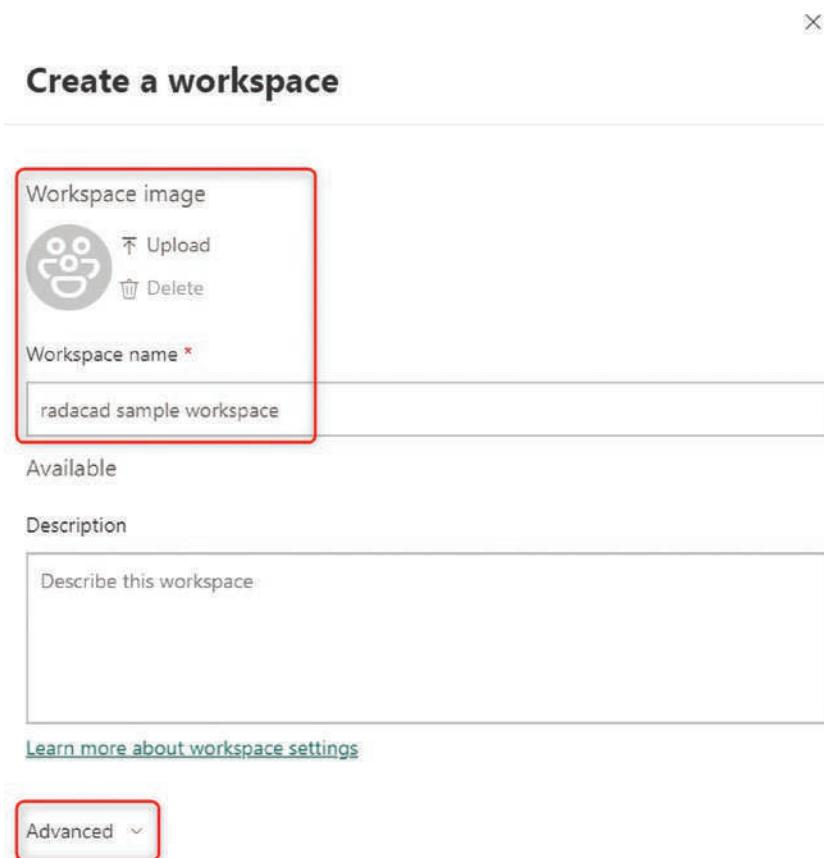
## Creating Workspaces

Creating workspaces is easy. You do so from the Power BI Service. Log in to the service and click Workspaces, as shown in Figure 30-2.



**Figure 30-2.** Creating a Power BI workspace

If you are already part of one or more workspaces, you'll see them in the list of workspaces. Click Create a Workspace. When you create a new workspace, you need to assign it a name (see Figure 30-3). The name of the workspace will be the name that others see when they join this workspace.



**Figure 30-3.** Setting up general settings for a Power BI workspace

The basic settings for the workspace include a name, a description, and an image. You are the workspace administrator (because you created it). You can use the Advanced section to set more detailed workspace settings, such as capacity settings.

## Create a workspace

[Learn more about workspace settings](#)

Advanced ^

Contact list

- Workspace admins
- Specific users and groups

Enter users and groups

Workspace OneDrive

(Optional)

License mode ①

- Pro
- Premium per user
- Premium per capacity
- Embedded ②

Develop a template app

Template apps are developed for sharing outside your organization.

A template app workspace will be created for developing and releasing the app. [Learn more](#)

Security settings

Allow contributors to update the app for this workspace

**Save**

**Cancel**

**Figure 30-4.** Advanced settings for a Power BI workspace

The Advanced settings include a contact list for the workspace. The workspace security settings will be set up later; this is just a contact list.

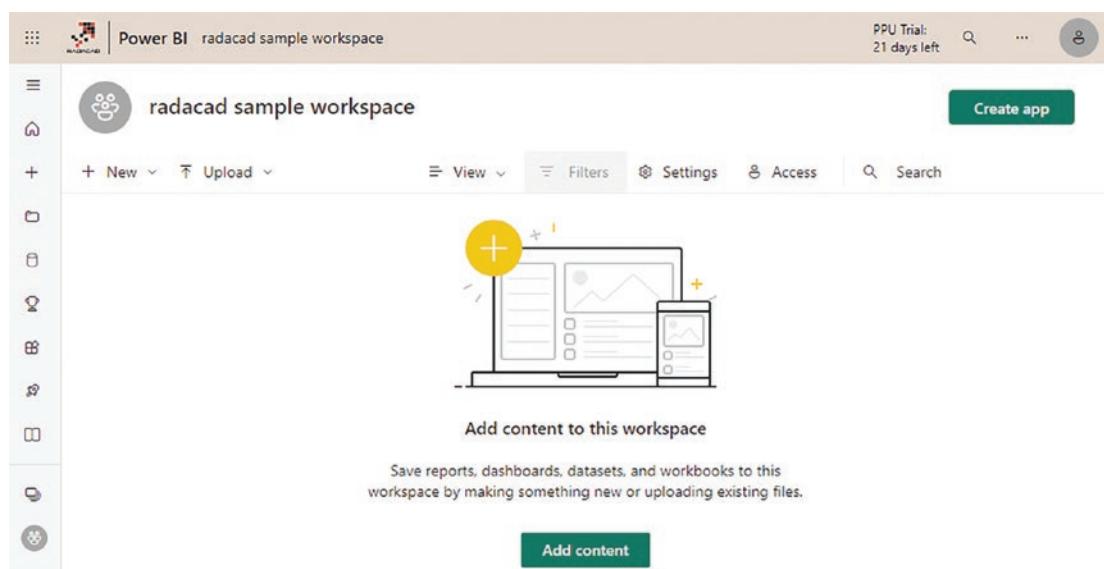
Another critical aspect of setting up a workspace is assigning it to a Premium workspace. If your organization purchased a Premium capacity or uses a Premium Per User (PPU) license, you can set up the Premium configuration in this section. Assigning a workspace to the Premium capacity enables you to use all the Premium functionalities (such as datamarts, AI functions inside the dataflow, computed entities, and so on). See Figure 30-5.



**Figure 30-5.** Premium capacity settings for a Power BI workspace

If you want to use a OneDrive as a shared folder so the workspace users can share content (not just Power BI content, but any file), fill in the Workspace OneDrive text area, as shown in Figure 30-4.

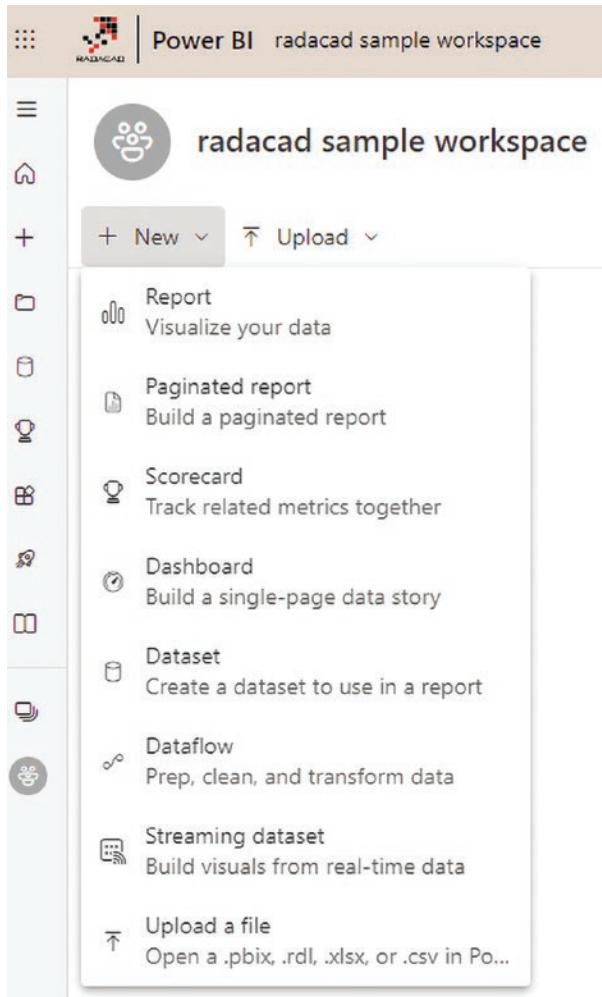
After creating the workspace, you will be automatically navigated from My Workspace to your new workspace. However, as you can see in Figure 30-6, the workspace doesn't have any content yet.



**Figure 30-6.** A new workspace is created in the Power BI Service

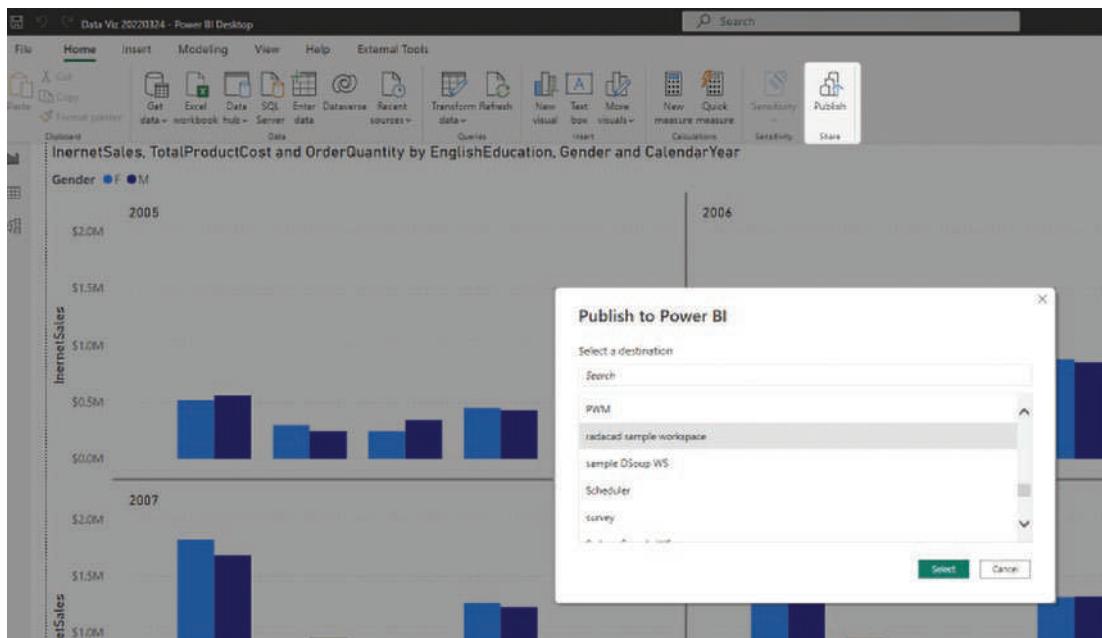
## Adding Content to a Workspace

There are many Power BI objects that exist inside a workspace (see Figure 30-7). Some of those can be created directly from the service, such as reports, dataflows, dashboards, datasets, streaming datasets, datamarts, and paginated reports. Others can be published through other applications, such as the Power BI Desktop.



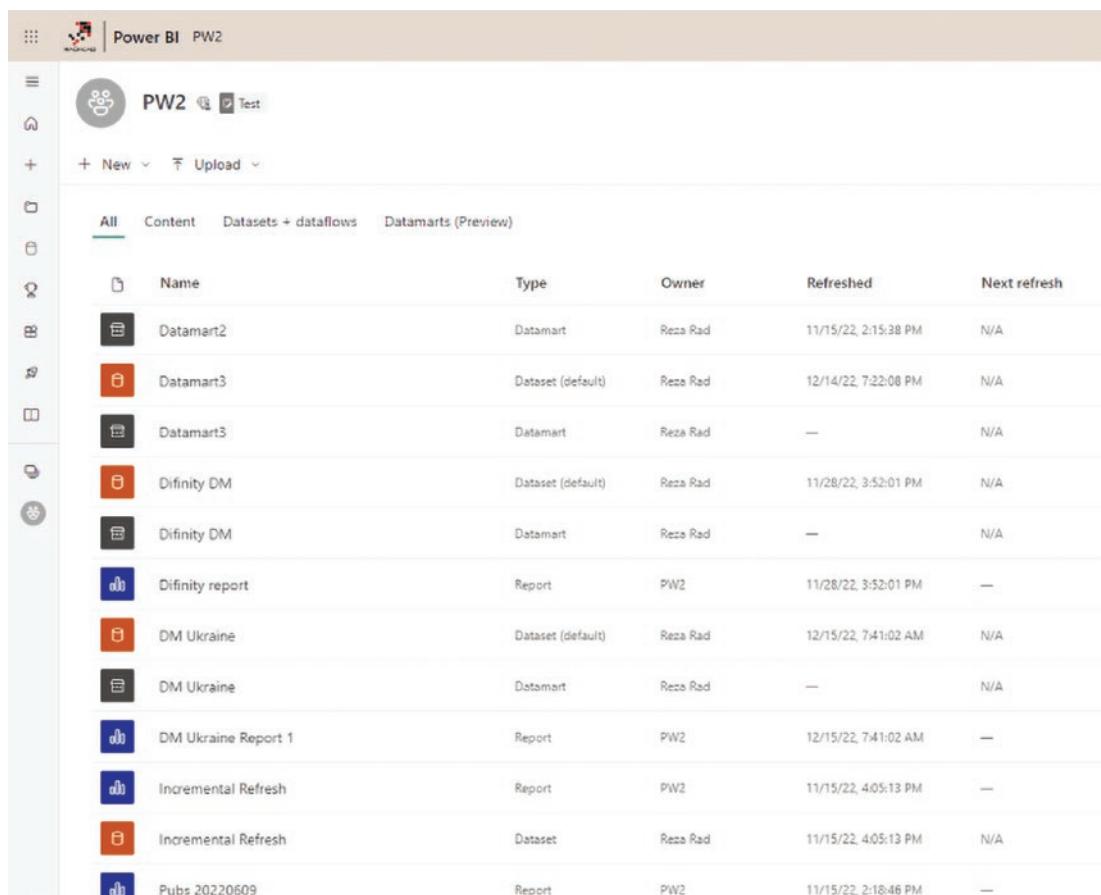
**Figure 30-7.** Creating Power BI objects within the workspace

When you open a Power BI file in the Power BI Desktop, you can click Publish, and if you are part of a workspace, you will see a popup window asking which workspace you want to publish the report to it. As an example, you could select the [radacad sample workspace](#) created here (see Figure 30-8).



**Figure 30-8.** Publish a Power BI file from the Power BI Desktop into a workspace

After publishing the content to the workspace, you and anyone else who is part of that workspace will see that content. Figure 30-9 shows an example of content in a workspace.



The screenshot shows the Power BI workspace content interface. At the top, there's a navigation bar with icons for Home, Power BI, and PW2, followed by a search bar and a 'Test' button. Below the navigation is a toolbar with '+ New' and 'Upload' dropdowns. A sidebar on the left contains various icons for navigation. The main area has tabs for 'All', 'Content', 'Datasets + dataflows', and 'Datamarts (Preview)'. The 'All' tab is selected, displaying a table of items:

	Name	Type	Owner	Refreshed	Next refresh
	Datamart2	Datamart	Reza Rad	11/15/22, 2:15:38 PM	N/A
	Datamart3	Dataset (default)	Reza Rad	12/14/22, 7:22:08 PM	N/A
	Datamart5	Datamart	Reza Rad	—	N/A
	Difinity DM	Dataset (default)	Reza Rad	11/28/22, 3:52:01 PM	N/A
	Difinity DM	Datamart	Reza Rad	—	N/A
	Difinity report	Report	PW2	11/28/22, 3:52:01 PM	—
	DM Ukraine	Dataset (default)	Reza Rad	12/15/22, 7:41:02 AM	N/A
	DM Ukraine	Datamart	Reza Rad	—	N/A
	DM Ukraine Report 1	Report	PW2	12/15/22, 7:41:02 AM	—
	Incremental Refresh	Report	PW2	11/15/22, 4:05:13 PM	—
	Incremental Refresh	Dataset	Reza Rad	11/15/22, 4:05:13 PM	N/A
	Pubs 20220609	Report	PW2	11/15/22, 2:18:46 PM	—

**Figure 30-9.** Power BI workspace content

Let's now talk about the access levels for the workspace users.

## Four Levels of Workspace Access

With workspaces, you can provide four levels of access. Three provide edit access and one is read-only:

- **Viewer:** Can view the content in a read-only mode.
- **Contributor:** Can create, edit, publish, and delete content in the workspace. A contributor cannot modify users or publish an app.
- **Member:** Can publish and update an app. Can share or allow others to reshare items. Can add members or others with lower permissions. Members can also do everything that contributors can do.
- **Administrator:** Can delete and update the workspace. Can add and remove members (or even other admins). Administrators can also do everything that members can do.

## Tenant Admin Control for Workspaces

The Power BI tenant administrator can control some settings for an organizational workspace through the Tenant Settings area. In the Tenant Settings, the administrator can control who is authorized to create a workspace and whether users can use a dataset across workspaces (see Figure 30-10).

The screenshot shows the Power BI Admin portal interface. On the left is a sidebar with various icons and a list of settings: Usage metrics, Users, Premium Per User, Audit logs, Capacity settings, Refresh summary, Embed Codes, Organizational visuals, Azure connections, Workspaces, Custom branding, and Protection metrics. The 'Workspaces' item is highlighted with a red box. To its right is a main content area titled 'Admin portal'. Below it is a section titled 'Tenant settings' with a red box around it. This section contains links to 'Usage metrics', 'Users', 'Premium Per User', 'Audit logs', 'Capacity settings', 'Refresh summary', 'Embed Codes', 'Organizational visuals', 'Azure connections', 'Workspaces', 'Custom branding', and 'Protection metrics'. To the right of the 'Tenant settings' box is another section titled 'Workspace settings' enclosed in a red box. It lists five configuration items:

- ▷ Create workspaces (new workspace experience)  
*Enabled for the entire organization*
- ▷ Use datasets across workspaces  
*Enabled for the entire organization*
- ▷ Block scheduled upgrade of empty workspaces  
*Enabled for the entire organization*
- ▷ Block notifications for scheduled workspace upgrades  
*Disabled for the entire organization*
- ▷ Users can reassign personal workspaces  
*Enabled for the entire organization*

**Figure 30-10.** Workspace settings for the Power BI tenant administrator

The administrator can also control whether users can assign a personal workspace to a Premium capacity. This last option also allows the tenant administrator to set a location for personal workspaces throughout the organization, as shown in Figure 30-11.

The screenshot shows the Power BI Admin portal interface. On the left, a sidebar lists various administrative settings like Tenant settings, Usage metrics, and Capacity settings (which is currently selected). The main area displays capacity details for 'Power BI Premium > O1'. It shows a 'Size | EM2' section with 'Base v-cores' set to '2' and an 'Autoscale | Off' section with 'Additional v-cores in use' set to '0'. Below these, there's a note about autoscale and a 'Manage Autoscale' button. A 'Capacity usage report' link is also present.

On the right, a modal window titled 'Assign workspaces' is open. It contains an 'Apply to:' section with three options: 'Workspaces by users', 'Specific workspaces', and 'The entire organization's My Workspaces' (which is selected and highlighted with a red box). Below this, a note explains that selecting 'The entire organization's My Workspaces' allows users to reassign individual workspaces to premium capacity. At the bottom of the modal are 'Apply' and 'Cancel' buttons.

**Figure 30-11.** Setting the location for *My Workspace* throughout the entire organization

Source: [powerbi.microsoft.com/en-us/blog/announcing-my-workspace-governance-improvement-public-preview/](https://powerbi.microsoft.com/en-us/blog/announcing-my-workspace-governance-improvement-public-preview/)

Another important option for the tenant administrator is connecting to personal workspaces throughout the organization and gaining access to them. This is particularly helpful if the employee the workspace belongs to has left the company and the Power BI content has been left in their personal account without anyone else having access. This is illustrated in Figure 30-12.

The screenshot shows the Power BI Admin portal interface. On the left, there's a sidebar with various navigation options like Tenant settings, Usage metrics, Users, Premium Per User, Audit logs, Capacity settings, Embed Codes, Organizational visuals, Azure connections, and Workspaces (which is highlighted with a red box). The main content area is titled 'Workspaces' and contains a sub-instruction: 'View personal and group workspaces that exist in your organization. To change users' ability to create workspaces'. Below this is a table with columns for 'Name' and 'Description'. The table lists four workspaces: 'PersonalWorkspace labuser17', 'PersonalWorkspace labuser29', 'PersonalWorkspace labuser26', and 'PersonalWorkspace Jon Song'. A context menu is open over the first workspace, showing options: 'Details', 'Get Access' (which is highlighted with a red box), and 'Capacity'. Another 'Get Access' button is also highlighted with a red box.

**Figure 30-12.** Gaining access to personal workspaces throughout the Power BI tenant

Administrators can also restore deleted workspaces, as shown in Figure 30-13.

This screenshot shows the 'Details' view for a workspace in the Power BI Admin portal. At the top, it says 'My workspace'. Below that is a blurred workspace preview. On the right, there's a 'Details' section with a 'Restore' button, which is highlighted with a red box. There's also a 'Restore' button inside a larger red box.

**Figure 30-13.** Restoring a Power BI workspace

You can restore deleted workspaces only within 90 days after they were deleted. During this period, the restore can be done through the Admin Portal.

# Advantages of Workspaces

## **Sharing multiple Contents with the team**

You may have shared a dashboard with a couple of your colleagues in your organization, but after a few weeks, you need a new dashboard, and you share that dashboard with them. A couple of months later, another team member asks for access to a Power BI dataset to create a report and share it with others. Power BI workspaces enable you to share content (dashboards, reports, and datasets) with all group members. You don't need to share each dashboard with each user, because groups make it easy for you.

## **Sharing all types of objects**

The dashboard only allows sharing the Power BI report, dashboard, and dataset. However, sharing through the workspace allows all the content to be shared. This includes but is not limited to dataflows, datasets, datamarts, dashboards, reports, metrics, paginated reports, and more.

## **Multiple workspaces**

It is hectic when you are part of multiple teams, and each team has its own dashboards, reports, and datasets. Your Shared with Me section in Power BI could have hundreds of items, and finding content might become a problem. Power BI workspaces create a separate environment for all members of the group. You can easily switch between workspaces in Power BI.

## **Isolated user/group administration**

When you share content with an individual in the organization, and that person leaves the company or is replaced by someone from another team, you have to remove sharing from a previous user account and assign it to the new user account. The best practice is to share content with groups. Workspace members can easily be managed by the administrator. Power BI workspaces can be shared with Office 365 groups. Once you use a group in Power BI, it is only an admin's task to add/remove members.

## **Best developer environment**

You need an environment to share multiple Power BI content for a team of developers. Everyone needs to have edit access to the content provided by the team. A Power BI workspace is the perfect solution for a development environment. You can create a workspace as a development environment and then share it with other developer team members with edit access. Then you all have access to the same content in your development workspace.

A Power BI workspace is a perfect solution for a development environment.

# Disadvantages of Workspaces

Workspaces do include some drawbacks you should consider.

## **Not suitable for end users**

Workspaces are not suitable for sharing content with end users. You can give users of the workspace read-only access to the content. However, this is only half of the requirement. In an end-user sharing environment, one of the primary requirements is to have the development and user environment separated.

Assume that you created a workspace and shared it with the end users. If you suddenly make changes in the workspace while they are using it, their view of the workspace breaks and changes.

With one workspace, your development and user environment are the same.

You cannot use one workspace and share it between developers and users. Creating multiple workspaces also leads to another challenge. To overcome this challenge, you can use apps on top of the workspaces to share content with end users.

### Complications of the workspace structure

Setting up a good workspace structure is a challenge. A workspace structure should cover the development, user needs, and deployment structure. This is more of a caution than a limitation. Use workspaces with care and make sure you have a good setup.

#### Power BI Pro or PPU

Creating Power BI workspaces, or even being part of them (even just to view them), requires a Pro or PPU license and is not part of a Power BI free user account. However, it is possible to create an app for the workspace from the Premium capacity and assign free Power BI users to it. This limitation is one of the main reasons this is not the most cost-effective option for sharing content with end users. See Figure 30-14.

### Upgrade to Power BI Pro



This feature is only available to users with a Power BI Pro license. When you upgrade, you get access to collaborate with others and distribute content. Upgrade today or try it free for 60 days. [Learn more](#)

[Try Pro for free](#) [Upgrade account](#) [Cancel](#)

**Figure 30-14.** The Power BI Pro option

## Summary

Power BI workspaces are a great way to share multiple Power BI content with users. If you have hundreds of dashboards, reports, and datasets, you can easily share them through a workspace with others.

Workspaces provide edit access and read-only access. Because of that, workspaces are a great way to create a collaborative development environment. Multiple developers can access the same content in a workspace with edit access.

Workspaces are better used in a development environment, not for an end-user environment. The main reason is that having one workspace for the dev or user environment makes it hard to develop. If a developer makes a change, the end user will be affected immediately. Managing multiple workspaces is also not an easy job.