PBIMCR

Mastering PBI Lifecycle:
From Data Connection to Git Integration

Agenda

1. Introduction to Power BI Lifecycle
2. Data Connection and Data Modeling
3. Deployment Steps for Power BI Reports
4. Enterprise Deployment Techniques
5. Integration of Power BI with Git
6. Collaborative Development and Version Control
7. Project Management for Large-Scale Power BI Initiatives

Marc Lelijveld

Technical Evangelist | Solution Architect
Macaw Netherlands

@MarcLelijveld

linkedin.com/in/MarcLelijveld

Data-Marc.com

FAVORITE STUFF:

Christian Todte

Power BI Consultant
Macaw Germany

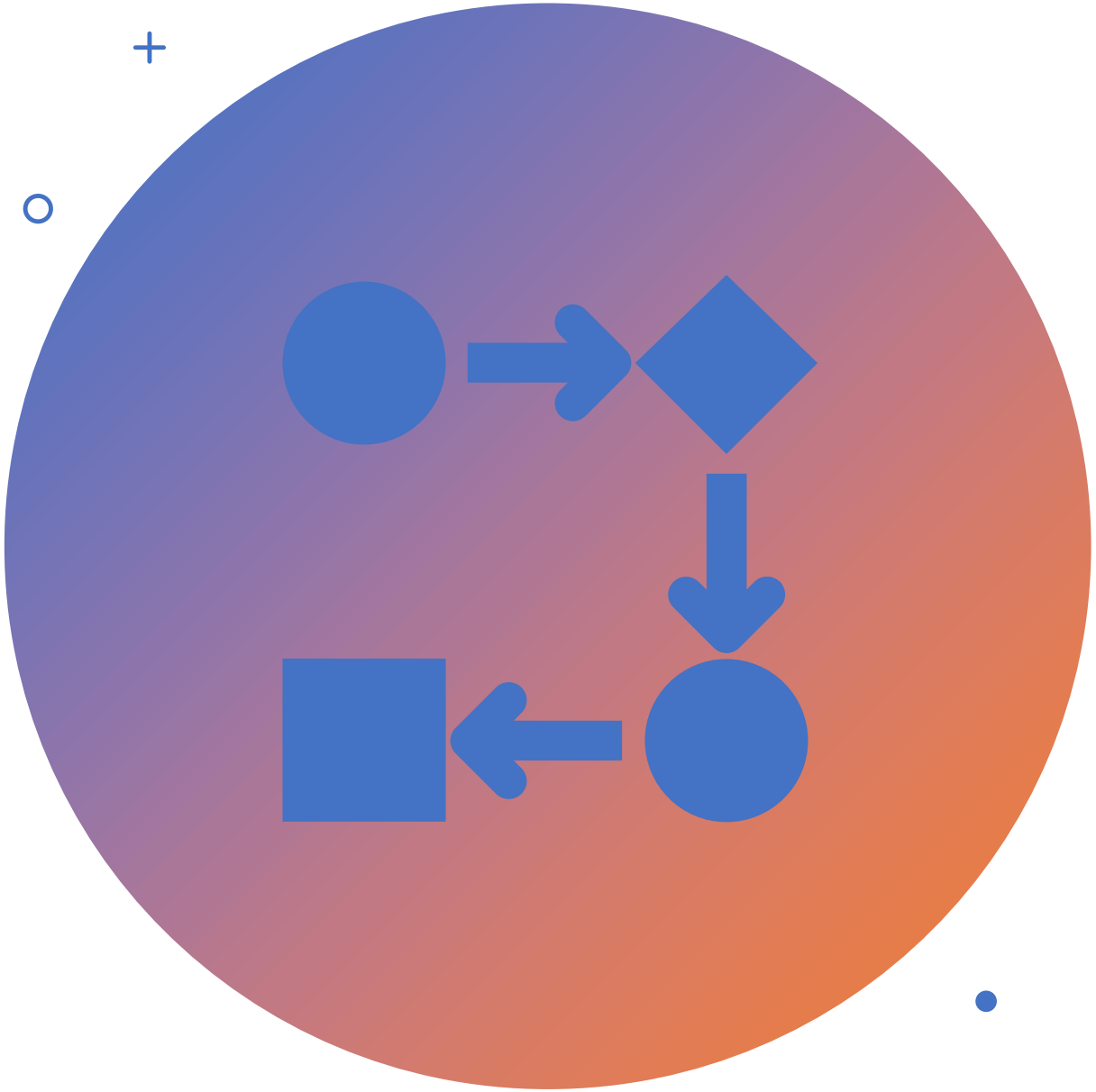
Microsoft Certified: Power BI Data Analyst Associate

Microsoft Certified: Azure Enterprise Data Analyst Associate

linkedin.com/in/ChristianTodte

FAVORITE STUFF:

Introduction to Power BI Lifecycle



Introduction to Power BI Lifecycle

The Power BI Lifecycle is an approach to developing, deploying, and managing Power BI solutions. It encompasses several key stages that ensure the effectiveness and efficiency of the entire process.

Power BI Lifecycle Stages

1. Connect: Establish connections to databases, Excel files, online services, etc.

2. Transform: Clean and shape data using Power Query Editor.

3. Model: Design relationships and calculations in the data model.

4. Visualize: Create visualizations using a variety of charts and graphs.

5. Deploy: Share reports on the Power BI Service or SharePoint.

6. Manage: Monitor usage, refresh schedules, and security.

Data Connection and Data Modeling

Data Connection and Data Modeling

Effective data connection and modeling are foundational for creating meaningful and insightful Power BI solutions. This involves connecting to diverse data sources and transforming raw data into a structured and usable format.

Key Aspects:

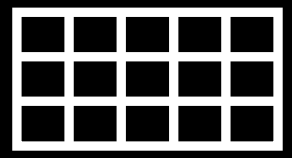
- Data Connection: Establish connections to databases, Excel files, web services, etc.

- Data Transformation: Use Power Query to clean, shape, and transform raw data.

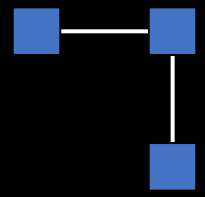
- Data Modeling: Design relationships, hierarchies, and calculations for meaningful analysis.

Major types of data models

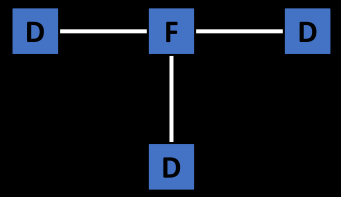
Dimensional



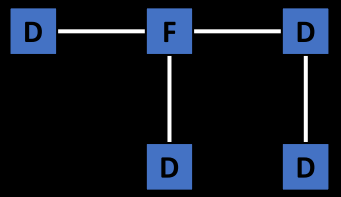
Flat



Normalized (n^{th} normal form)



Star



Snowflake

Flat or Denormalized Schema

- All attributes for model exist in a single table
- Highly inefficient
- Model has extra copies of data > slow performance
- Size of a flat table can blow up quickly as data model becomes complex

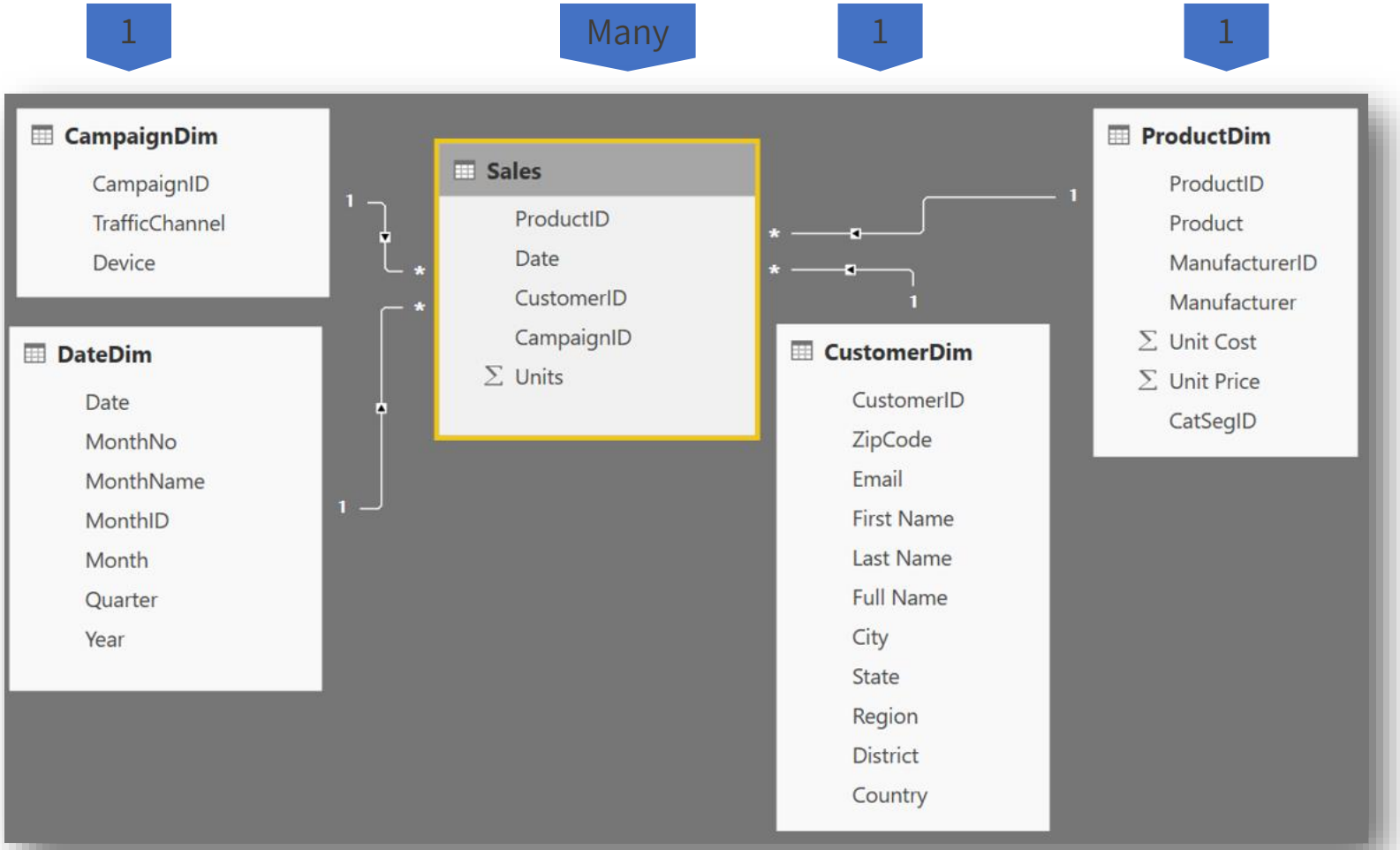
	ProductID	Product	Date	CustomerID	Email	Last Name	First Name	Full Name	CampaignID
1	676	Maximus UC-41	9/25/2011	70283	Farrah.Kent@xyza.com	Kent	Farrah	Farrah Kent	22
2	585	Maximus UC-50	3/24/2014	70283	Farrah.Kent@xyza.com	Kent	Farrah	Farrah Kent	15
3	585	Maximus UC-50	11/30/2014	138334	Martha.Mcclain@xyza.…	Mcclain	Martha	Martha Mcclain	8
4	585	Maximus UC-50	6/21/2015	27193	Hedda.Mcintosh@xyza.…	Mcintosh	Hedda	Hedda Mcintosh	22
5	585	Maximus UC-50	1/6/2013	238970	Lunea.Walker@xyza.com	Walker	Lunea	Lunea Walker	21
6	585	Maximus UC-50	3/22/2013	182241	Upton.Page@xyza.com	Page	Upton	Upton Page	17
7	449	Maximus UM-54	9/25/2011	195385	Drake.Wells@xyza.com	Wells	Drake	Drake Wells	22
8	449	Maximus UM-54	9/30/2014	168009	Wallace.Bender@xyza.…	Bender	Wallace	Wallace Bender	17
9	449	Maximus UM-54	8/12/2014	110391	Astra.Erickson@xyza.…	Erickson	Astra	Astra Erickson	20
10	449	Maximus UM-54	4/16/2014	49327	Echo.Bradley@xyza.com	Bradley	Echo	Echo Bradley	7
11	449	Maximus UM-54	2/28/2013	65952	Yoko.Gross@xyza.com	Gross	Yoko	Yoko Gross	17
12	449	Maximus UM-54	6/6/2013	97	Yoshi.Grant@xyza.com	Grant	Yoshi	Yoshi Grant	10
13	449	Maximus UM-54	5/14/2013	56757	Brian.Carrillo@xyza.…	Carrillo	Brian	Brian Carrillo	10
14	449	Maximus UM-54	4/9/2015	248715	Mark.Hewitt@xyza.com	Hewitt	Mark	Mark Hewitt	19
15	449	Maximus UM-54	4/28/2013	248715	Mark.Hewitt@xyza.com	Hewitt	Mark	Mark Hewitt	8
16	449	Maximus UM-54	3/28/2014	240831	Oscar.Avila@xyza.com	Avila	Oscar	Oscar Avila	18
17	449	Maximus UM-54	2/26/2014	201004	Duncan.Mcintosh@xyza…	Mcintosh	Duncan	Duncan Mcintosh	19
18	615	Maximus UC-80	5/14/2012	212645	Jacob.Santiago@xyza.…	Santiago	Jacob	Jacob Santiago	22
19	615	Maximus UC-80	5/14/2012	70666	Hilary.Collier@xyza.…	Collier	Hilary	Hilary Collier	22
20	615	Maximus UC-80	5/14/2012	114459	Chester.Mitchell@xyz…	Mitchell	Chester	Chester Mitche…	22
21	615	Maximus UC-80	5/14/2012	221670	Sage.Yang@xyza.com	Yang	Sage	Sage Yang	22
22	615	Maximus UC-80	6/3/2012	168009	Wallace.Bender@xyza.…	Bender	Wallace	Wallace Bender	22
23	615	Maximus UC-80	6/3/2012	154439	Iliana.Dunlap@xyza.c…	Dunlap	Iliana	Iliana Dunlap	22
24	615	Maximus UC-80	6/4/2012	191391	Joelle.Lee@xyza.com	Lee	Joelle	Joelle Lee	22

Normalized

- Reduce or eliminate data redundancy
- There are multiple types of normal forms, but we will not look into these as these are not suitable for Power BI

Star Schema

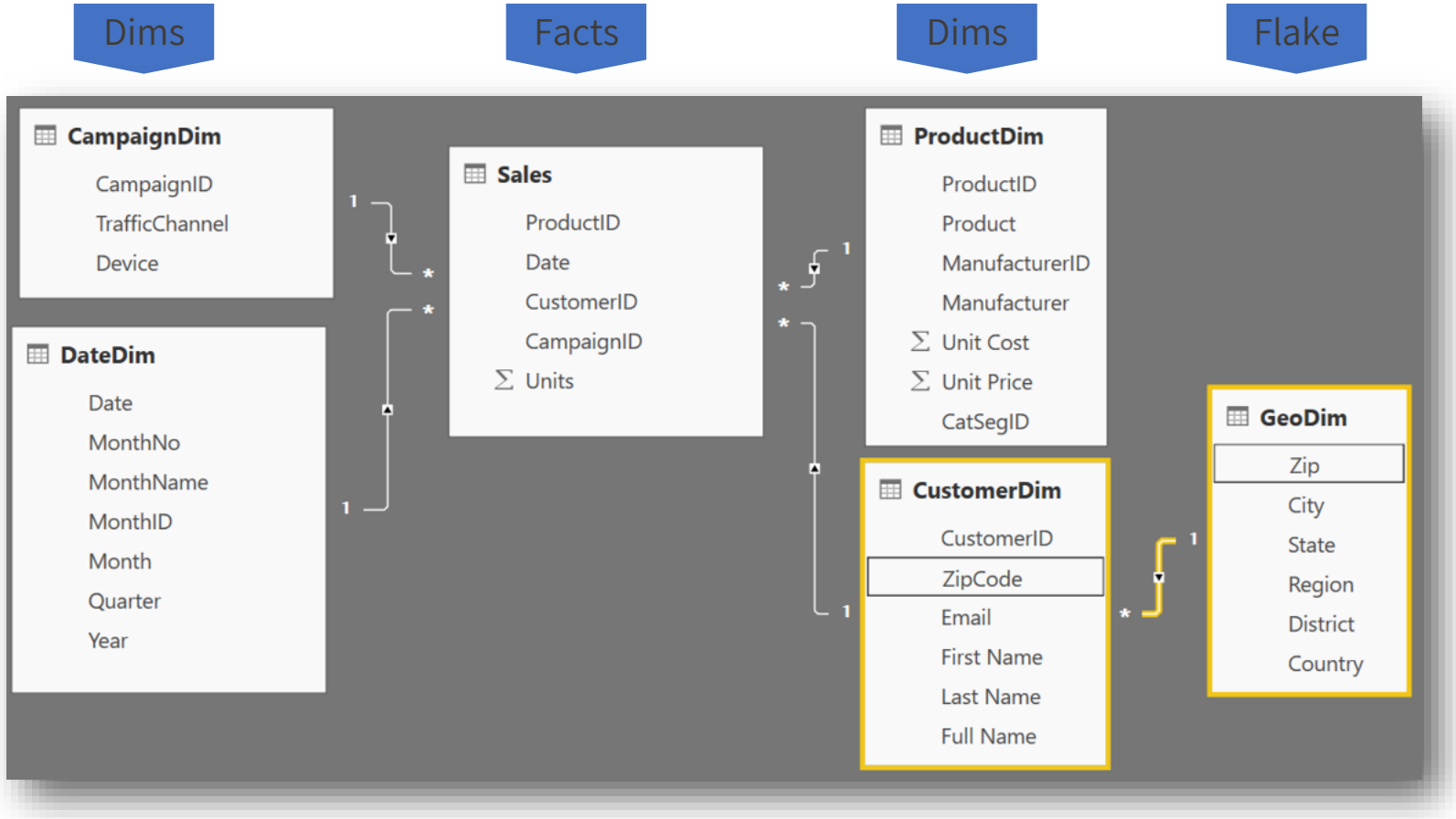
- **Fact** table in the middle
- Surrounded by **Dims**
- Looks like a '**Star**'
- Fact table is the "Many" side of the (one to many) relationship



macaw

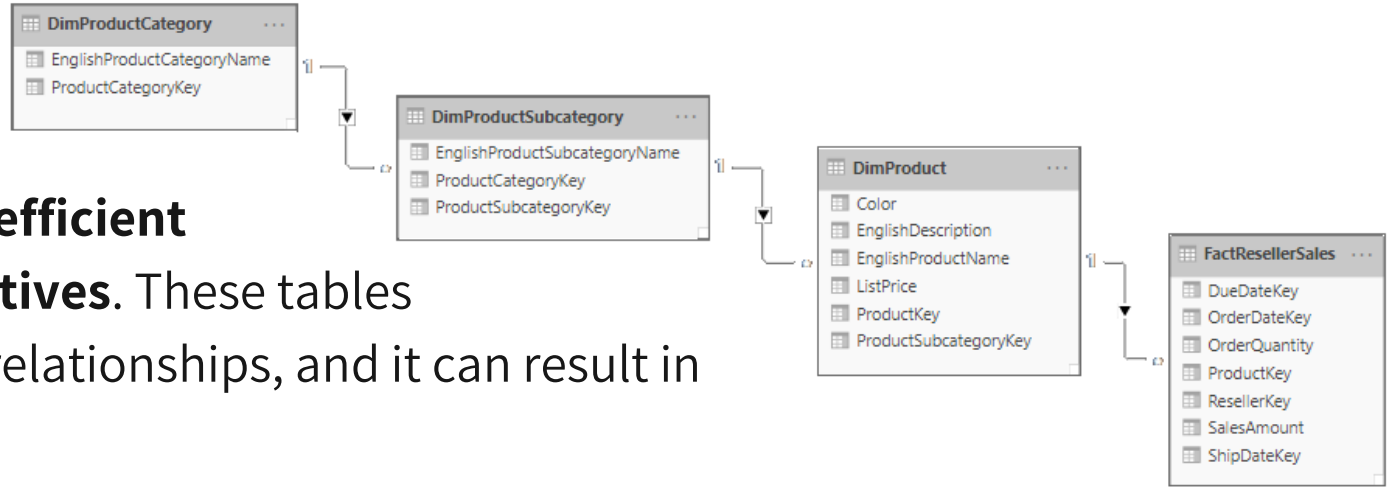
Snowflake Schema

- Center is a Star schema
- **Fact** table in middle
- Surrounded by **Dims**
- Dims “snowflake” off of other Dims
- If you have many, it looks like a ‘Snowflake’
- Dim or Fact tables can be the “Many” side of the relationship



macaw

Snowflake dimensions



Power BI loads more tables which is **less efficient from storage and performance perspectives**. These tables must include columns to support model relationships, and it can result in larger model size.

Longer relationship **filter propagation** chains will need to be traversed, which will likely be **less efficient** than filters applied to a single table.

The **Fields pane** presents more model tables to report authors, which can result in a **less intuitive experience**, especially when snowflake dimension tables contain just one or two columns.

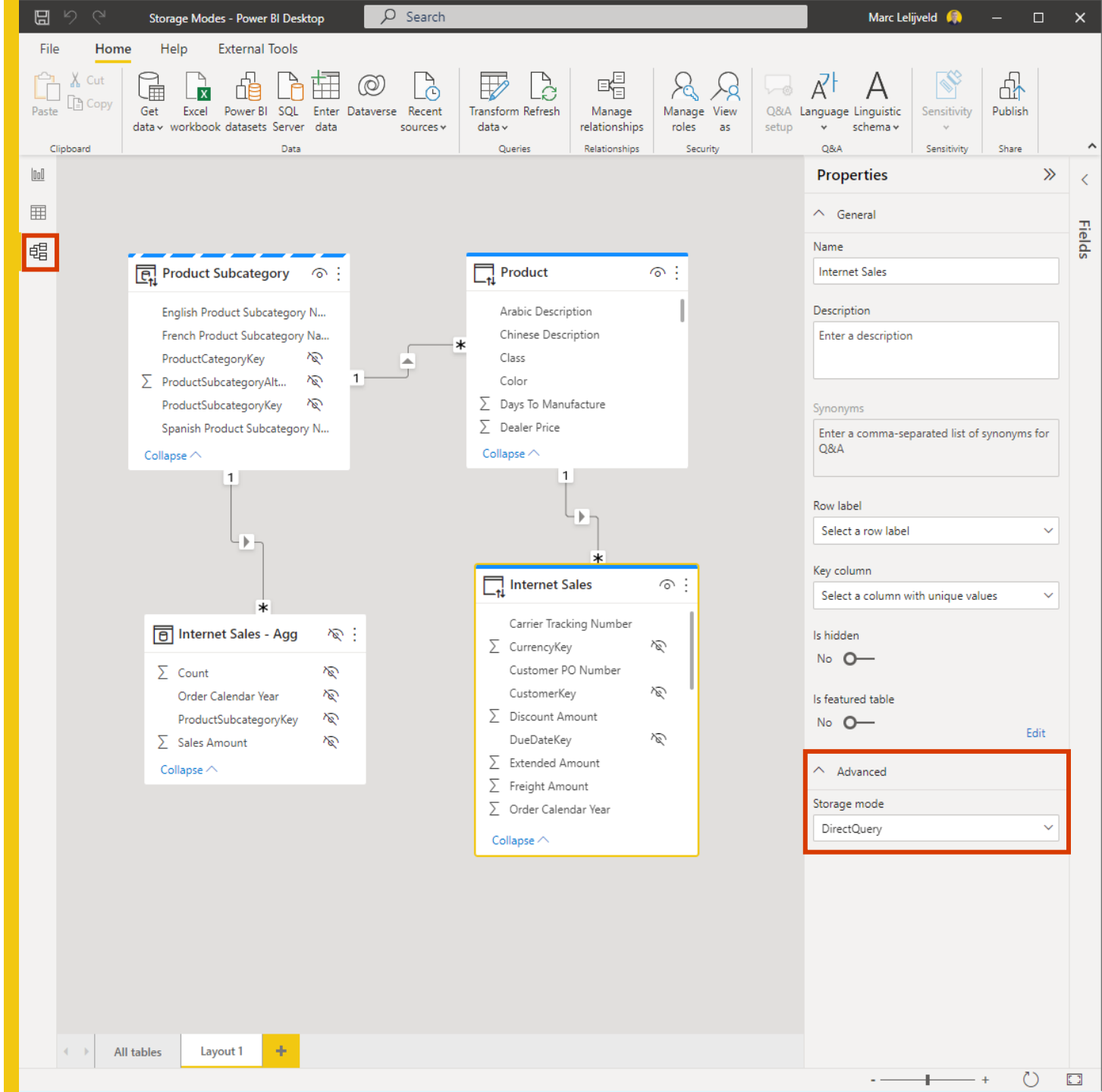
It's **not possible to create a hierarchy** that spans the tables.

macaw

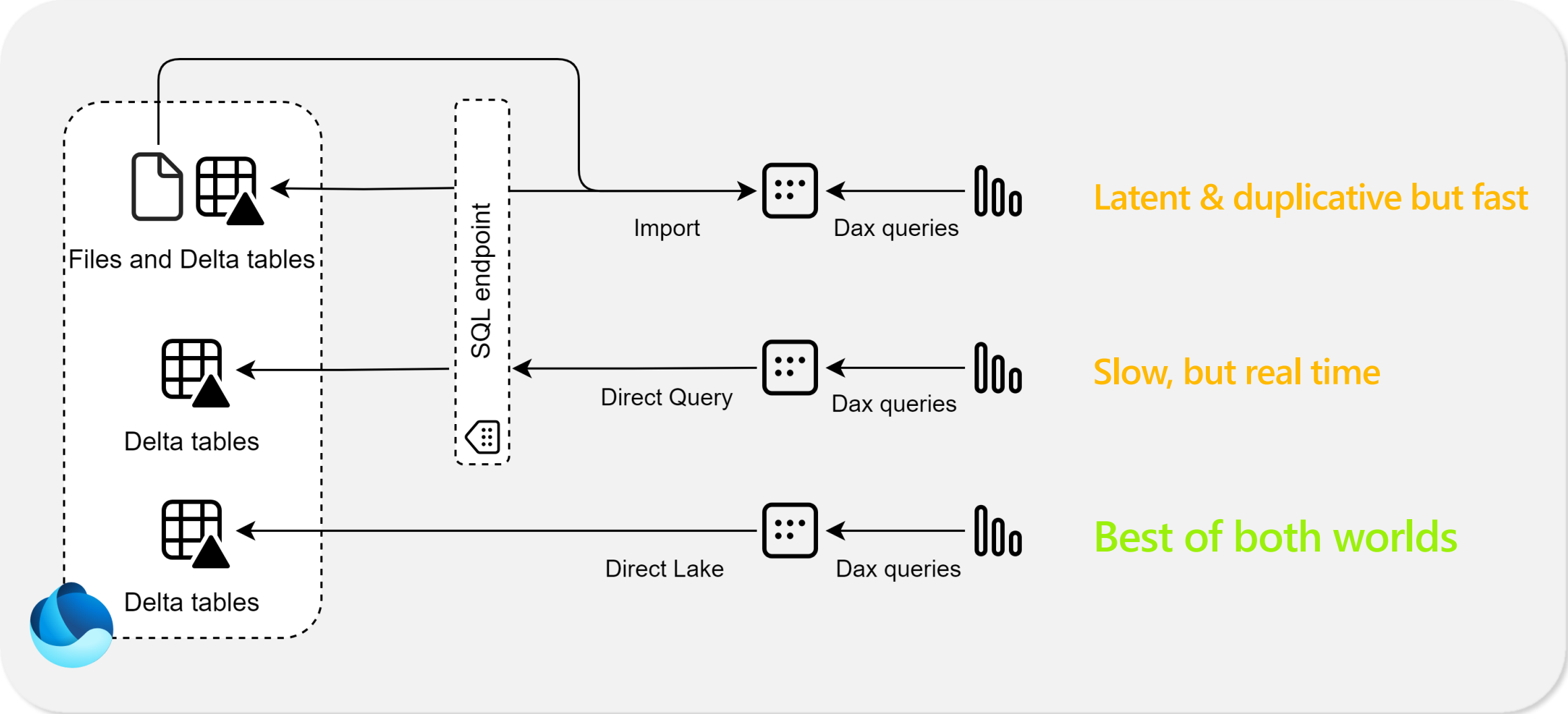
Different types of storage modes

Three familiar storage modes

- **Import** – data cached in the model
- **DirectQuery** – queries are submitted to the back-end data source
- **Dual** – can act in both above storage modes, depending on query context

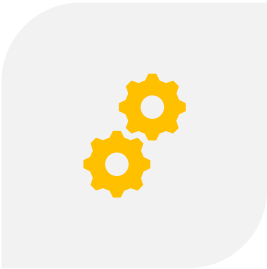


Direct Lake

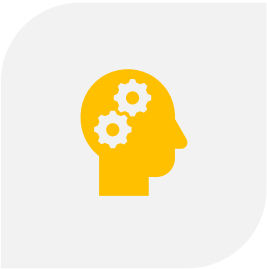

Files and Delta tables
Delta tables
Delta tables
SQL endpoint
Import
Direct Query
Direct Lake
Dax queries
Dax queries
Dax queries
Latent & duplicative but fast
Slow, but real time
Best of both worlds

Why is it important to have a Good Data model?

IMPROVES UNDERSTANDABILITY OF THE DATA



INCREASES PERFORMANCE OF DEPENDENT PROCESSES AND SYSTEMS



INCREASES RESILIENCE TO CHANGE

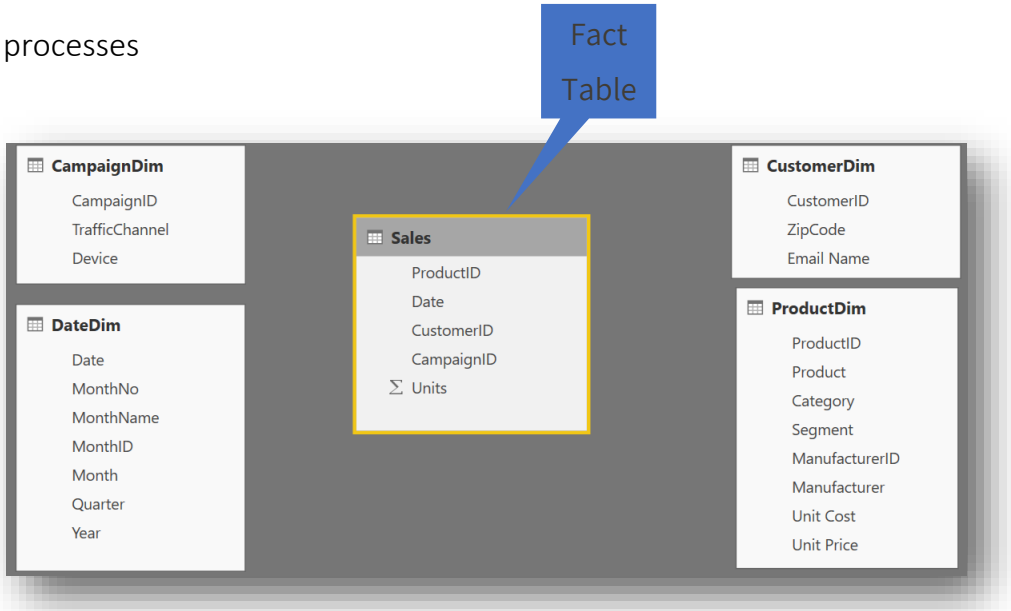
SPEEDS UP REPORT DEVELOPMENT

Components of a data model – Fact Table

Fact Table

- Contains numerical information about business processes
- These numbers can be **aggregated**
  Examples:
  - *Transactions*
  - *Sales Revenue*
  - *Units*
  - *Cost*

- Measures are usually **sliceable**
  Examples:
  - *By Month*
  - *By Customer*
  - *By Product*



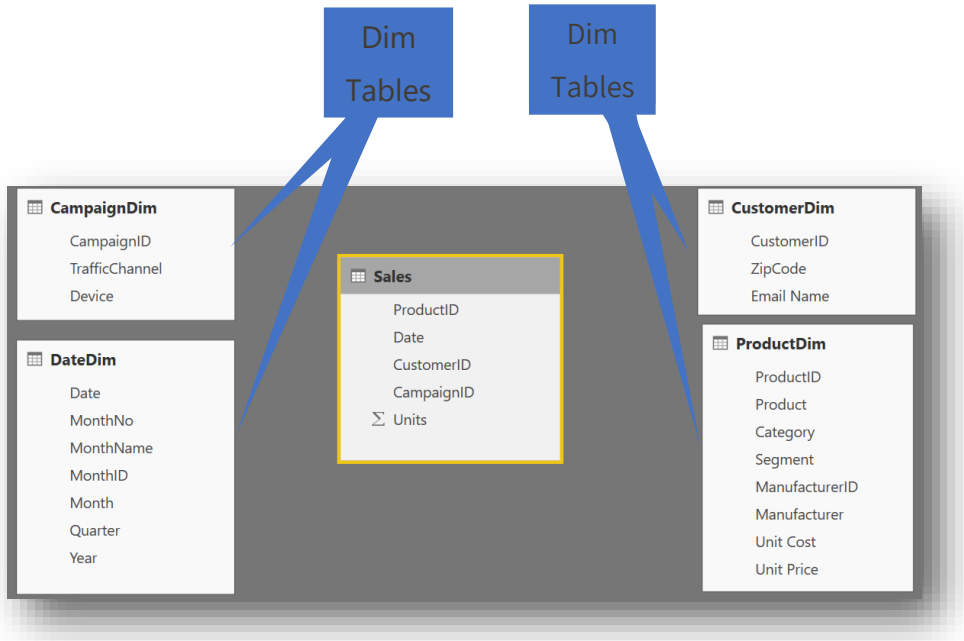
Components of a data model – Dim Table

Dim Table

- A dimension table (usually called dim) contains descriptive attributes that define how a fact should roll up.

Examples:

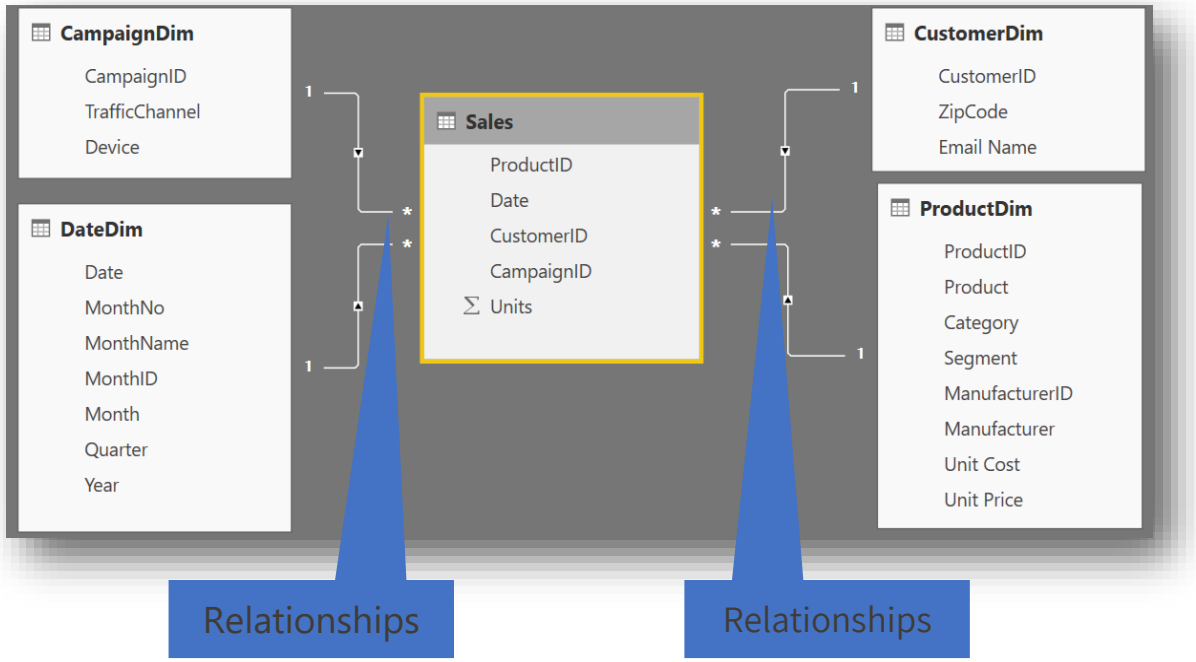
- *By Month*
- *By Customer*
- *By Country*



Components of a data model - Relationships

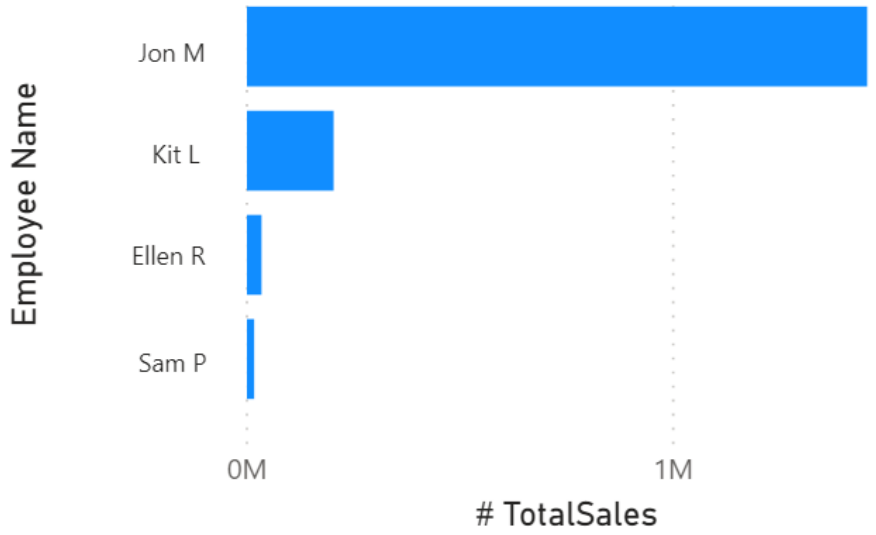
- **Relationships**
  - Connection between tables (usually fact & dim tables) using columns from each table.

- 3 kinds of Relationships
  - *1 to Many*
  - *1 to 1*
  - *Many to Many (with a bridge table)*
-



Facts & Dimensions with relation

- **Fact** and **Dimension** with proper **relation** allow for a highly performant, sustainable and correctly working visualization



Data modeling best practices in short

- Starschema all the things!
- Avoid bi-directional or many-to-many relationships
- Avoid limited relationships
- Implement role-playing dimensions rather than duplication
- Minimize redundant measure using calculation groups
- Avoid ambigiuous data models
- … etcetera

Deployment Steps for Power BI Reports

Deployment Steps for Power BI Reports

- Publish Reports: Use Power BI Desktop or Power BI Service to publish reports.

- Configure Data Sources: Ensure data sources are configured for accurate data refresh.

- Set Refresh Schedules: Schedule data refresh to keep reports up to date.

- Manage Security: Configure security settings to control access to reports.

Best Practices for Power BI Report Deployment

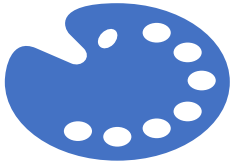
- Testing: Conduct thorough testing before deploying reports to production.
- Documentation: Maintain documentation for deployment processes and configurations.
- Backup Strategies: Implement backup strategies to safeguard report data.

Demo

Impactful Visualizations

- Clarity: Ensure clarity in visualizations to avoid misinterpretation.

- Relevance: Choose visuals that are relevant to the data and message.



- Interactivity: Use interactive elements for a dynamic user experience.

Best Practices in Power BI Visualizations

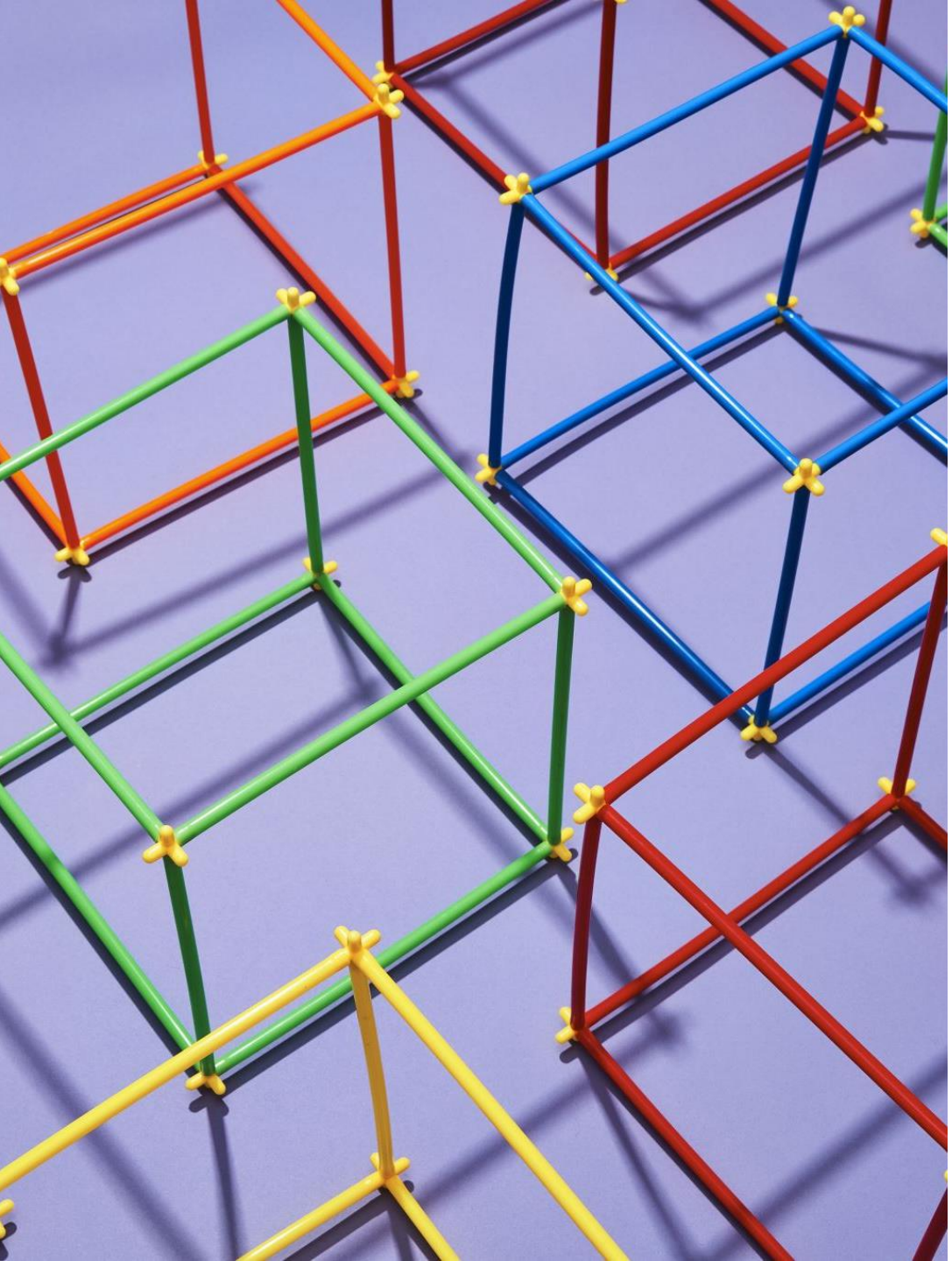
- Consistent Colors: Use a consistent color palette for better visual coherence.

- Proper Axis Scaling: Ensure accurate representation of data by scaling axes appropriately.

- Data Labels: Include data labels to provide additional context.

- Storytelling: Use Power BI features to tell a compelling data story.

Enterprise Deployment Techniques



Enterprise Deployment Techniques

Deploying Power BI solutions at an enterprise scale requires careful consideration of scalability, security, and performance optimization. This ensures the effective distribution and accessibility of insights across the organization.

- Scalability: Ensure the solution can handle a growing user base and data volume.

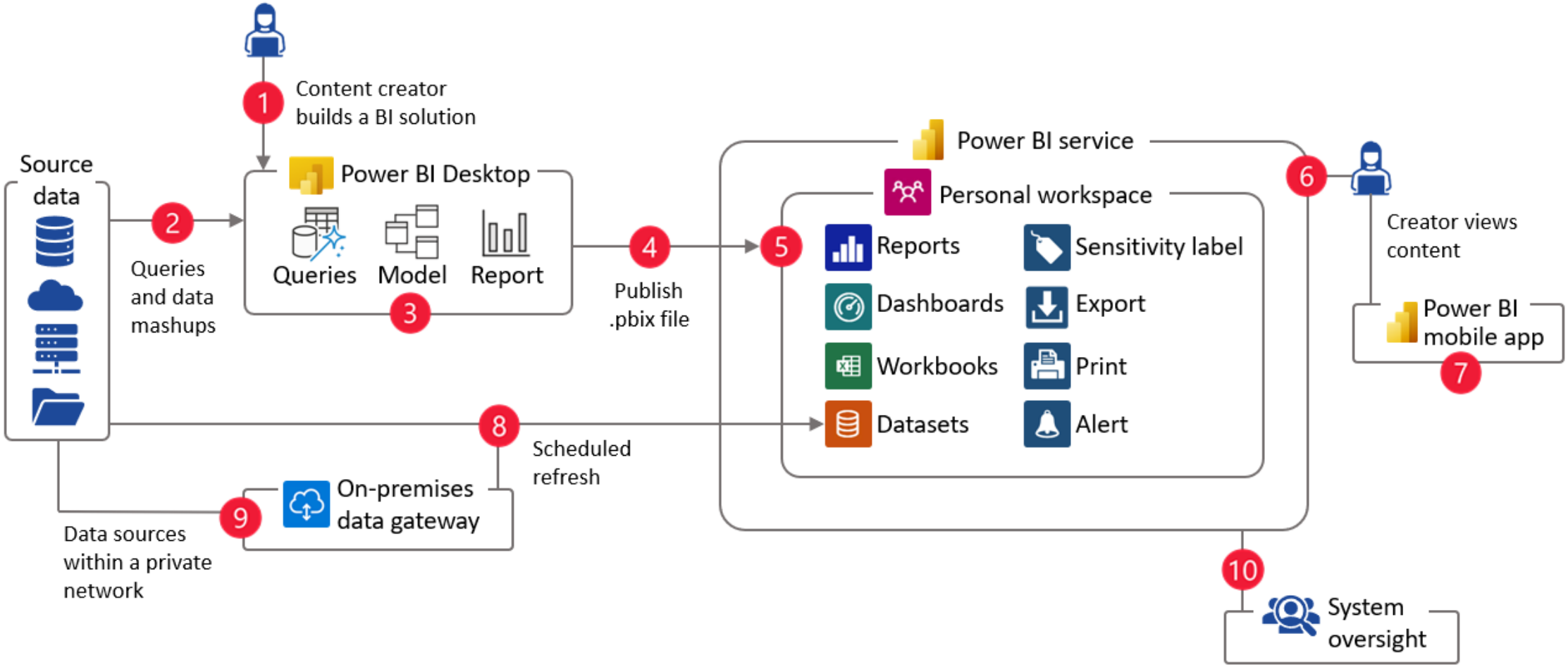
- Security: Implement robust security measures to protect sensitive data.

- Optimization: Optimize performance for efficient data retrieval and visualization.

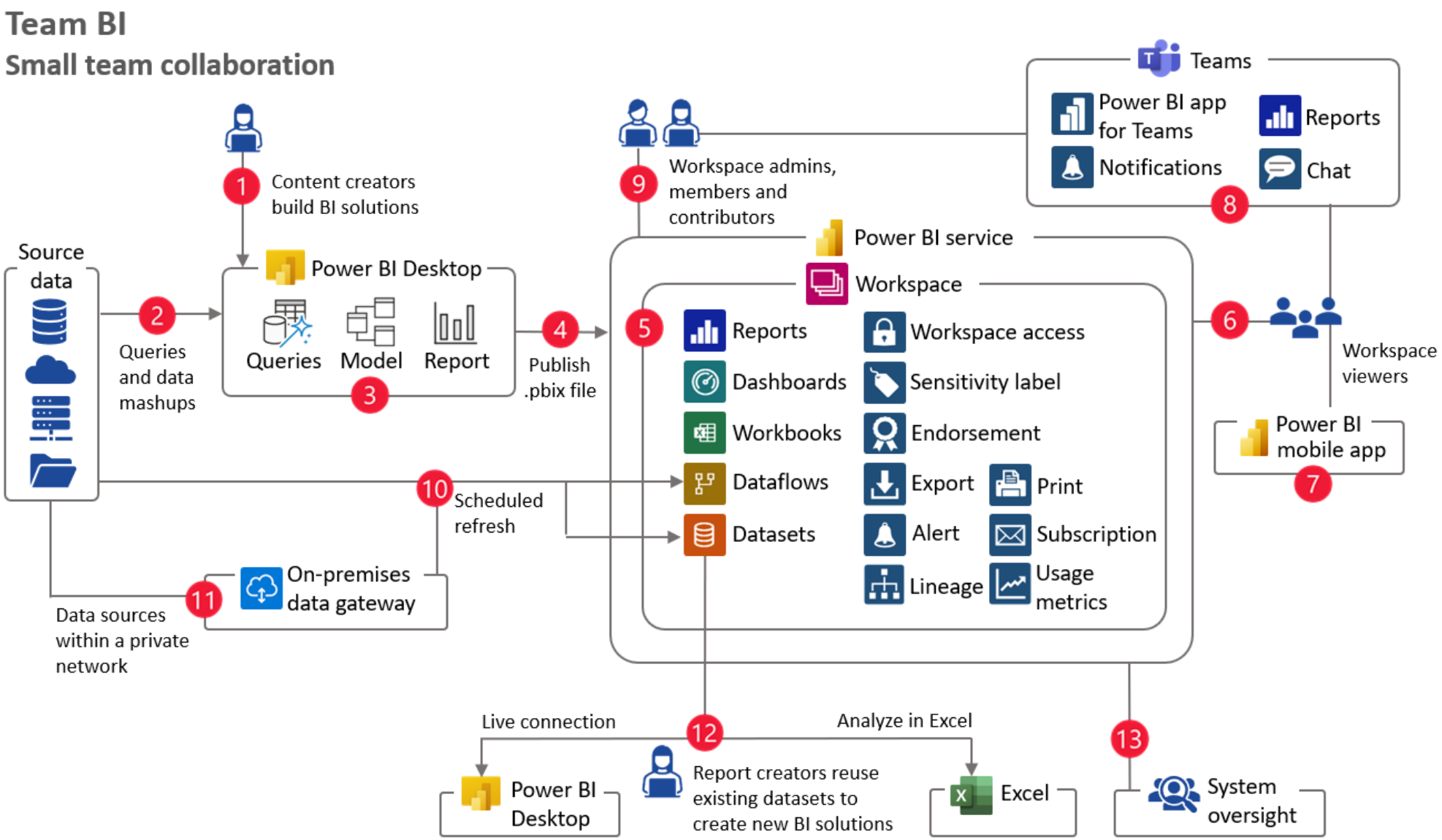
Power BI Architecture – Personal BI

Personal BI

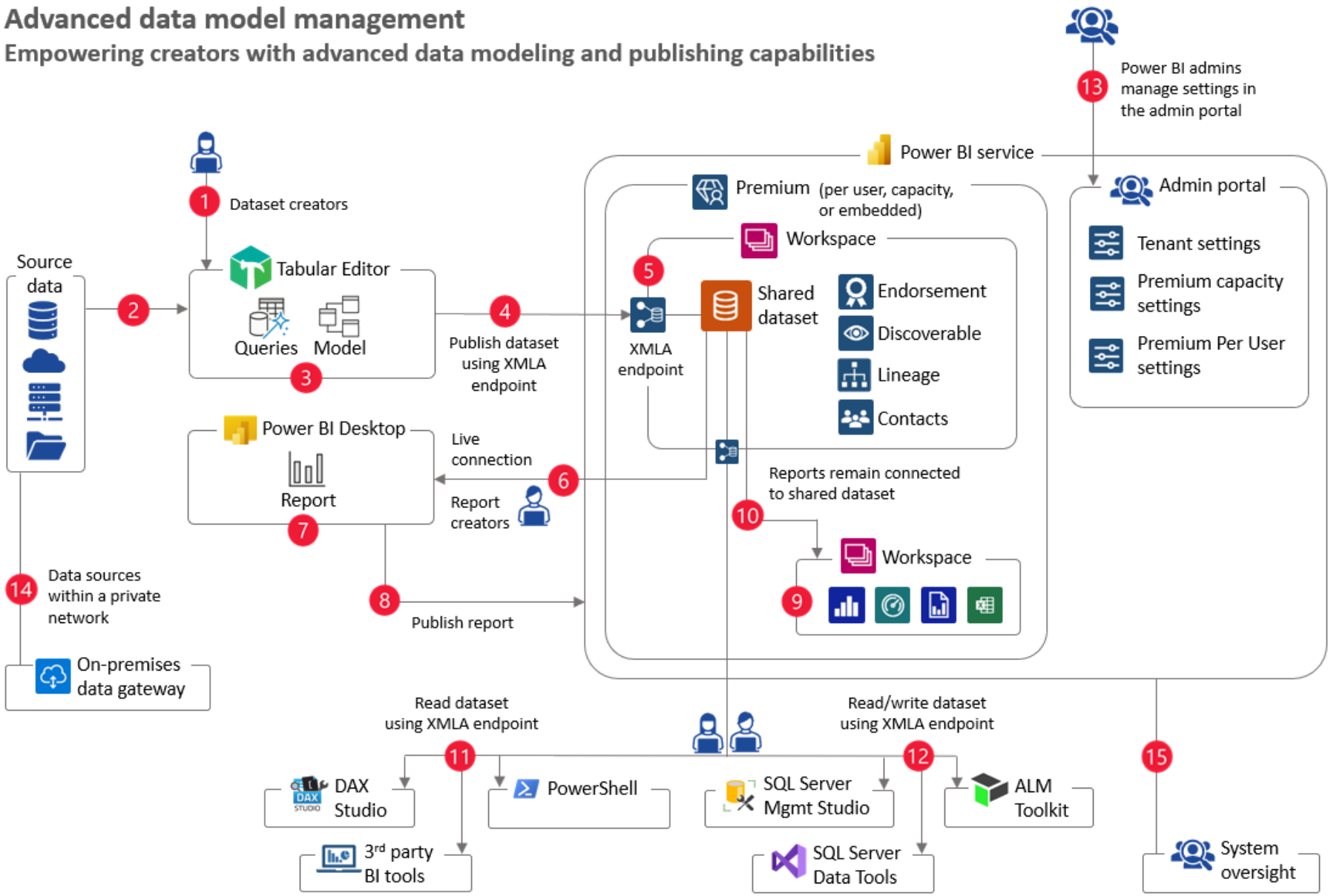
Private analytics for an individual



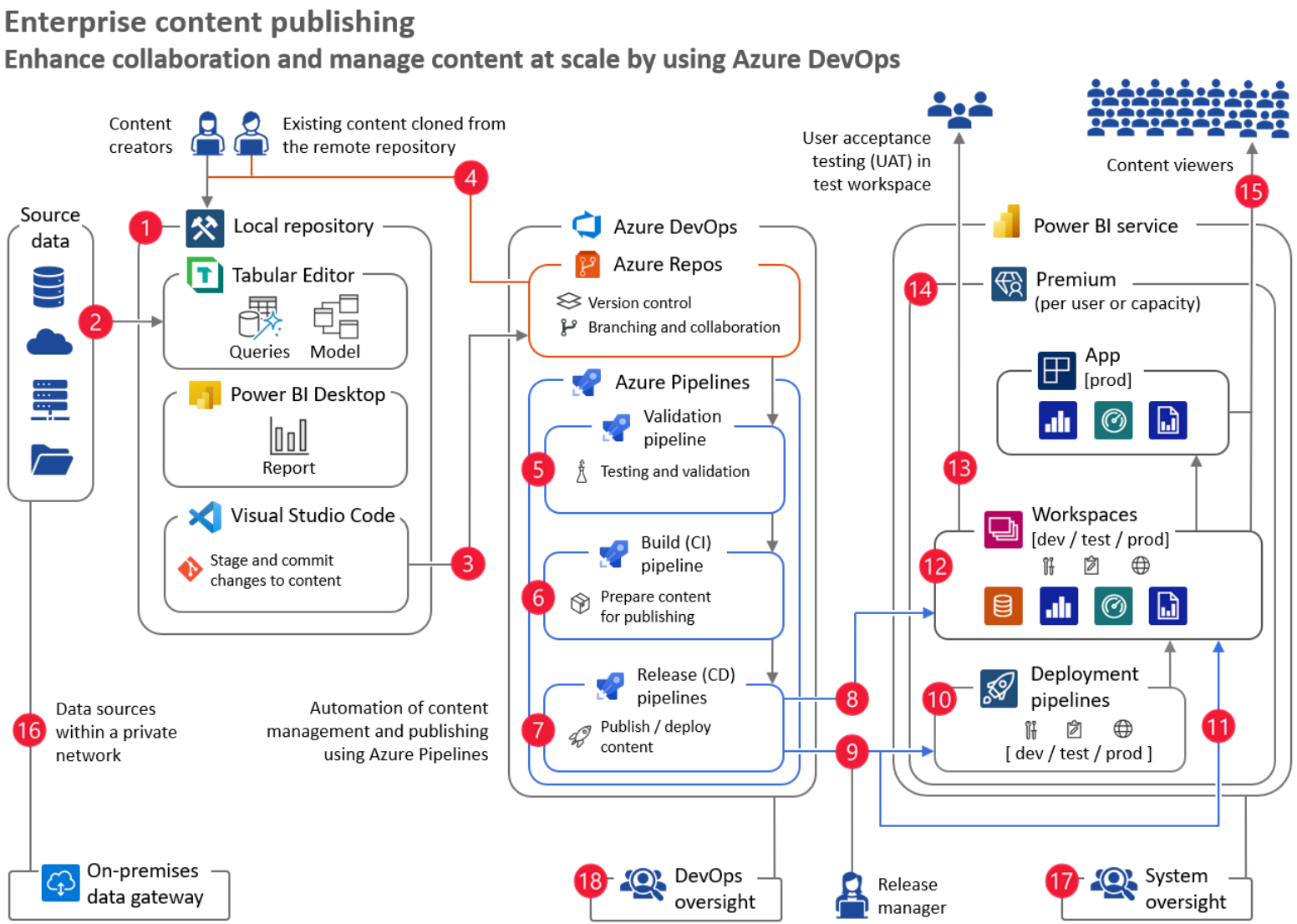
Power BI Architecture – Team BI



Enterprise Architecture – Advanced Data Model Management

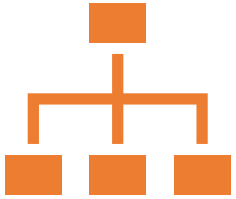


Enterprise Architecture – Content Management mit DevOps



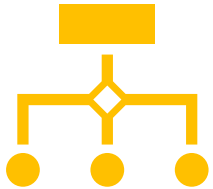
Integration of Power BI with Git

Integration of Power BI with Git



Version Control: Track changes and collaborate on Power BI projects with Git.

Collaborative Development: Enable multiple team members to work on the same project concurrently.



History Tracking: Maintain a history of changes for auditing and rollback purposes.

Getting Started with GIT Integration

Enabling GIT Integration requires the following tools*, which we will describe in the coming slides:

1. Git: https://Git-scm.com/
2. Visual Studio Code: https://code.visualstudio.com
3. Azure DevOps: https://dev.azure.com/
4. Power BI Desktop: https://powerbi.microsoft.com/en-us/desktop/

* There are multiple other options/providers available. The tools chosen are a personal preference.

Git

Git: Git itself is the version control system that you will use to track and manage changes in your Power BI project. Git provides the core functionality for version control, branching, merging, and collaboration. Git works best when it is used with code and does not handle large files well.

Visual Studio Code

Visual Studio Code

Git Client: You need Git client software installed on your computer to interact with Git repositories. There are several popular Git clients available, such as Git command-line interface (CLI), GitKraken, Sourcetree, and Visual Studio Code.

Visual Studio Code (VS Code) is a free, open-source code editor developed by Microsoft. Key features of Visual Studio Code include:

1. Code Editing
2. Git Integration

Azure DevOps

Azure DevOps is a **set of development tools** and services provided by Microsoft that aims to enable efficient software development and delivery. It provides a comprehensive **platform to plan, develop, test, deliver, and monitor applications.** Azure DevOps encompasses a range of features and services, including

1. Azure Boards
2. **Azure Repos**
3. Azure Pipelines
4. Azure Test Plans
5. Azure Artifacts

Power BI

Power BI Desktop is the desktop application that is part of the Microsoft Power BI suite. It is a Windows-based tool used for creating, editing, and publishing interactive reports and visualizations. Power BI Desktop offers advanced data modeling, transformation, and visualization capabilities, allowing users to build complex and insightful reports.

Git Integration Steps

1

Create a Git Repository: Set up a Git repository for your Power BI project.

2

Connect Power BI to Git: Use the Power BI Desktop to connect to your Git repository.

3

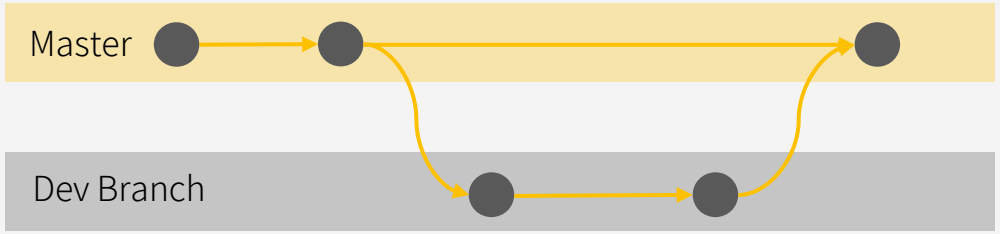
Publish to Git: Publish your Power BI project to the Git repository for version control.

4

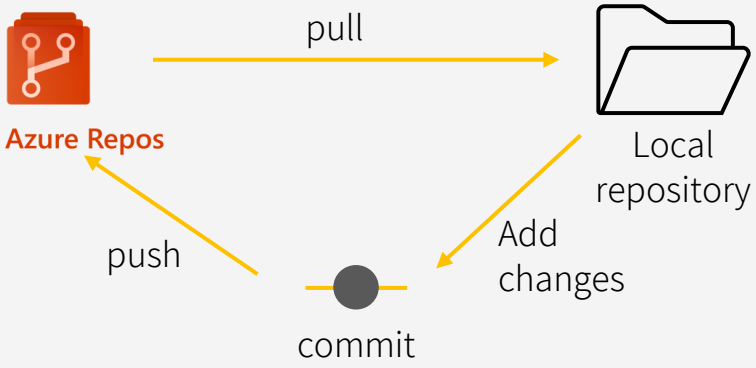
Collaborate: Multiple team members can now collaborate on the same project using Git features.

Important concepts for working Git

Branching



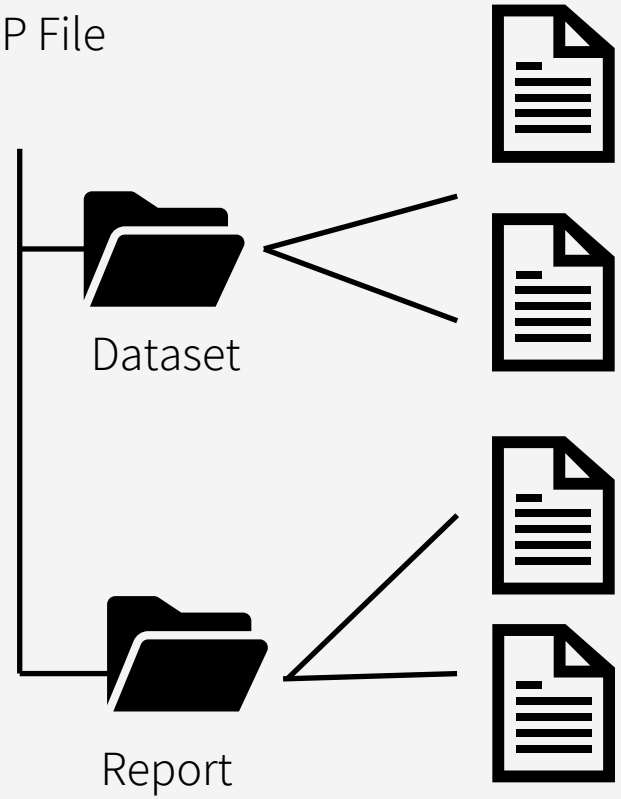
Push/pull and commit



What is a PBIP file

PBIX File

PBIP File

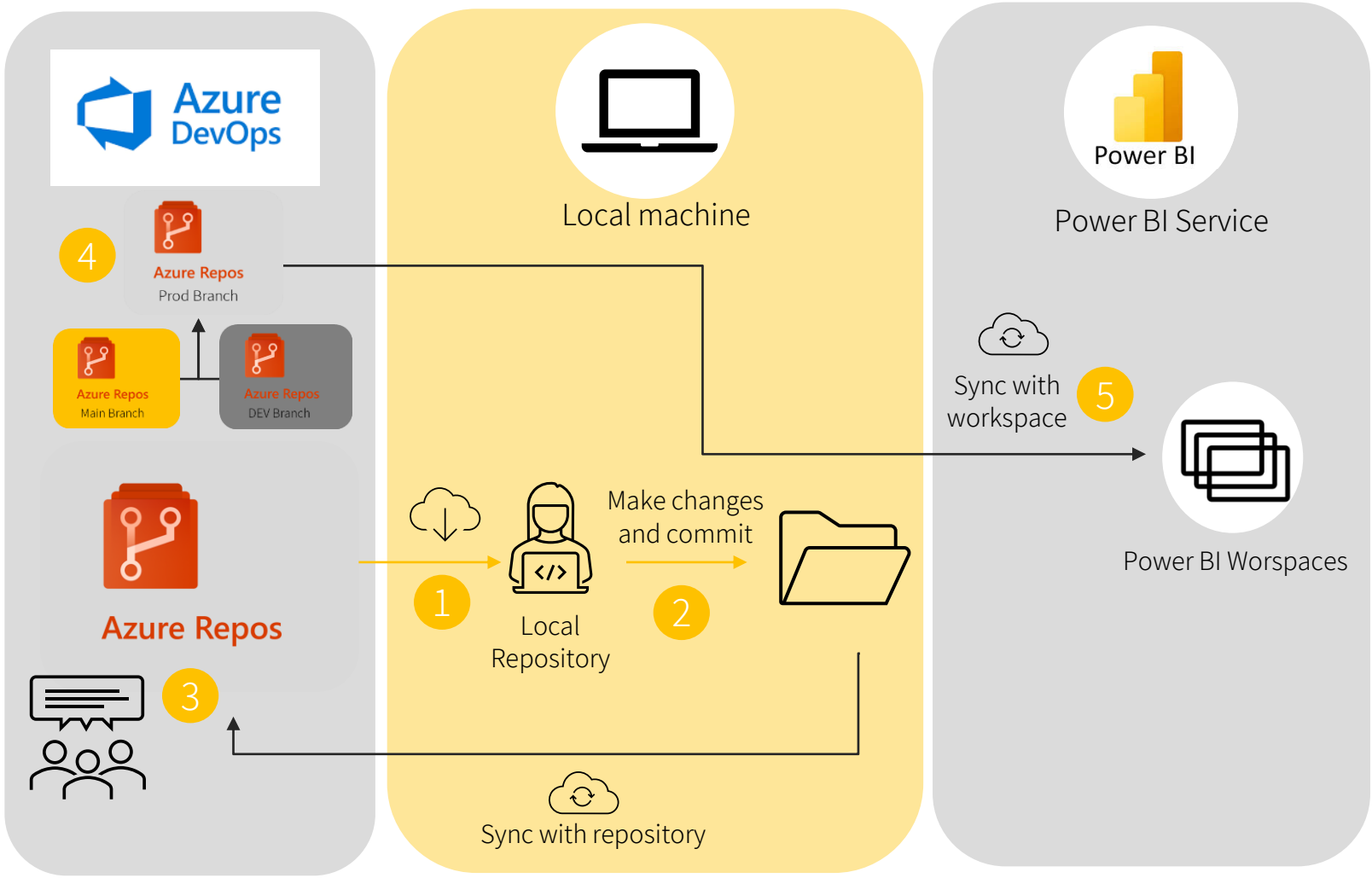


Let‘s have a look!

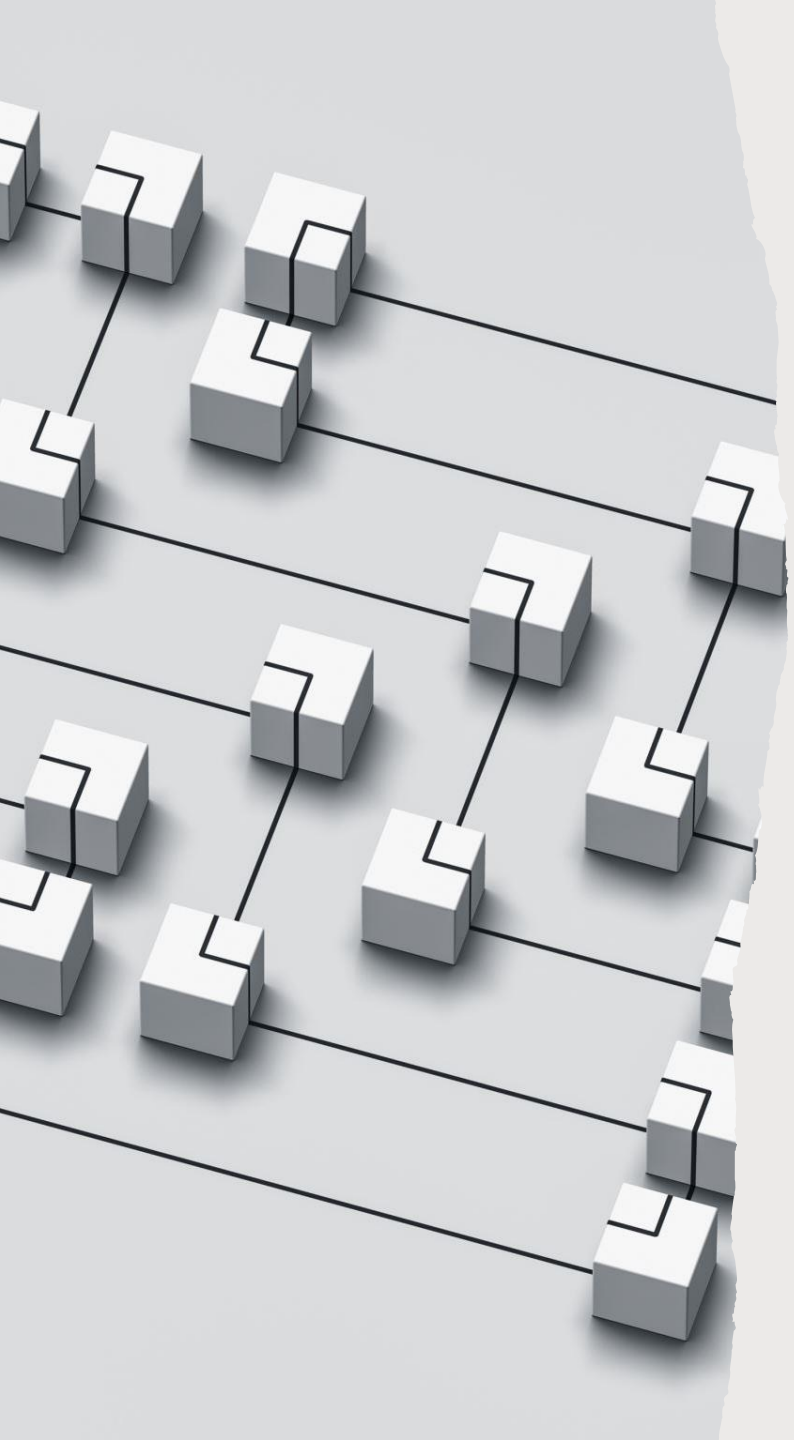
General Workflow

When using Git there are multiple options or workflows how to collaborate and build your project. A widely adopted and recommended approach is the "feature branch" or "git flow" workflow. This workflow involves the following steps

1. Create/clone a branch
2. Work on the branch
3. Review and merge
4. Continuous Integration
5. Continuous Delivery



Demo



Collaborative Development and Version Control

- Branching Strategies: Define clear branching strategies for parallel development.
- Code Reviews: Implement code reviews to ensure quality and consistency.
- Merging Process: Establish a smooth process for merging changes into the main branch.

Best Practices for Collaborative Development

- Clear Communication: Maintain open communication channels among team members.

- Documentation: Document development processes, guidelines, and decisions.

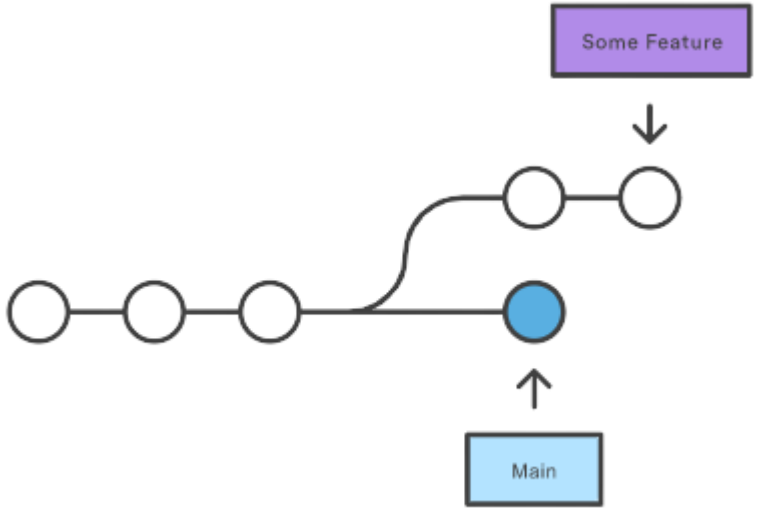
- Training: Provide training to ensure team members are familiar with Azure DevOps.

Branching

Create Branch.

- Branches are used to create separate, isolated lines of development that can be worked on independently.
- Common practice is to give your branch the same name as the feature you are working on

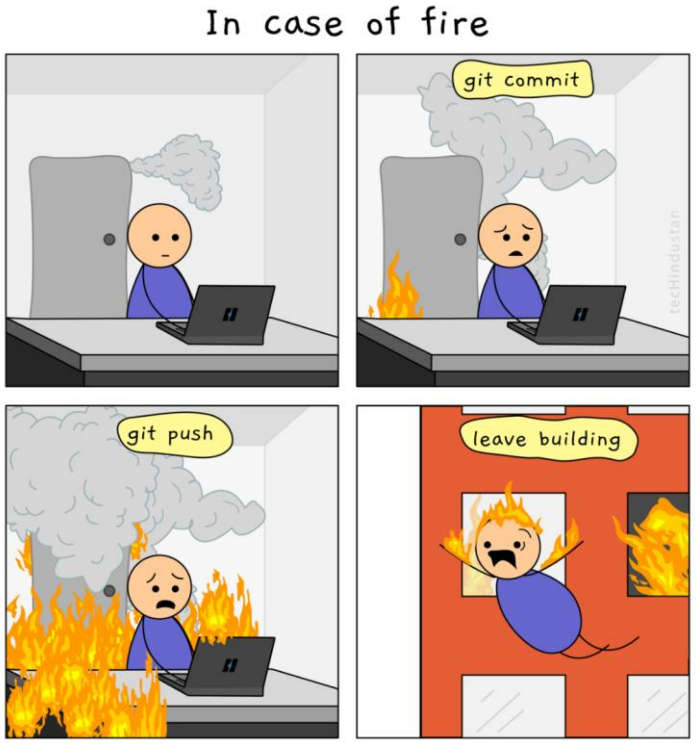
Note: Branches are an important tool for managing complex projects with multiple developers or features.



Push.

Push

- When performing a *Push*, Git uploads local commits to a remote repository



Pull Request.

Pull Request
Request

- A pull request is a feature of Git that allows developers to propose changes to a codebase and request that they be reviewed and merged into the main codebase.

Note: Good pull requests can help ensure the stability and maintainability of a codebase and make it easier for other developers to contribute to the project.

Merge.

Merge

- Merging changes in Git is the process of combining the changes from one branch into another.

Notes:

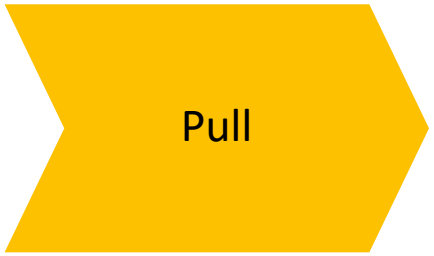
- *Merging changes is an important part of collaborating with others on a Git project, as it allows you to combine changes from multiple branches into a stable, maintainable codebase.*
- *It's important to be careful when merging changes to avoid introducing bugs or conflicts into the codebase.*

Merge Variations.

Merge

- Rebasing
- Squash commit

Pull.



- Pulling changes in Git is the process of updating your local repository with changes from a remote repository.

Demo

Project Management for Large-Scale Power BI Initiatives

Project Management for Large-Scale Power BI Initiatives

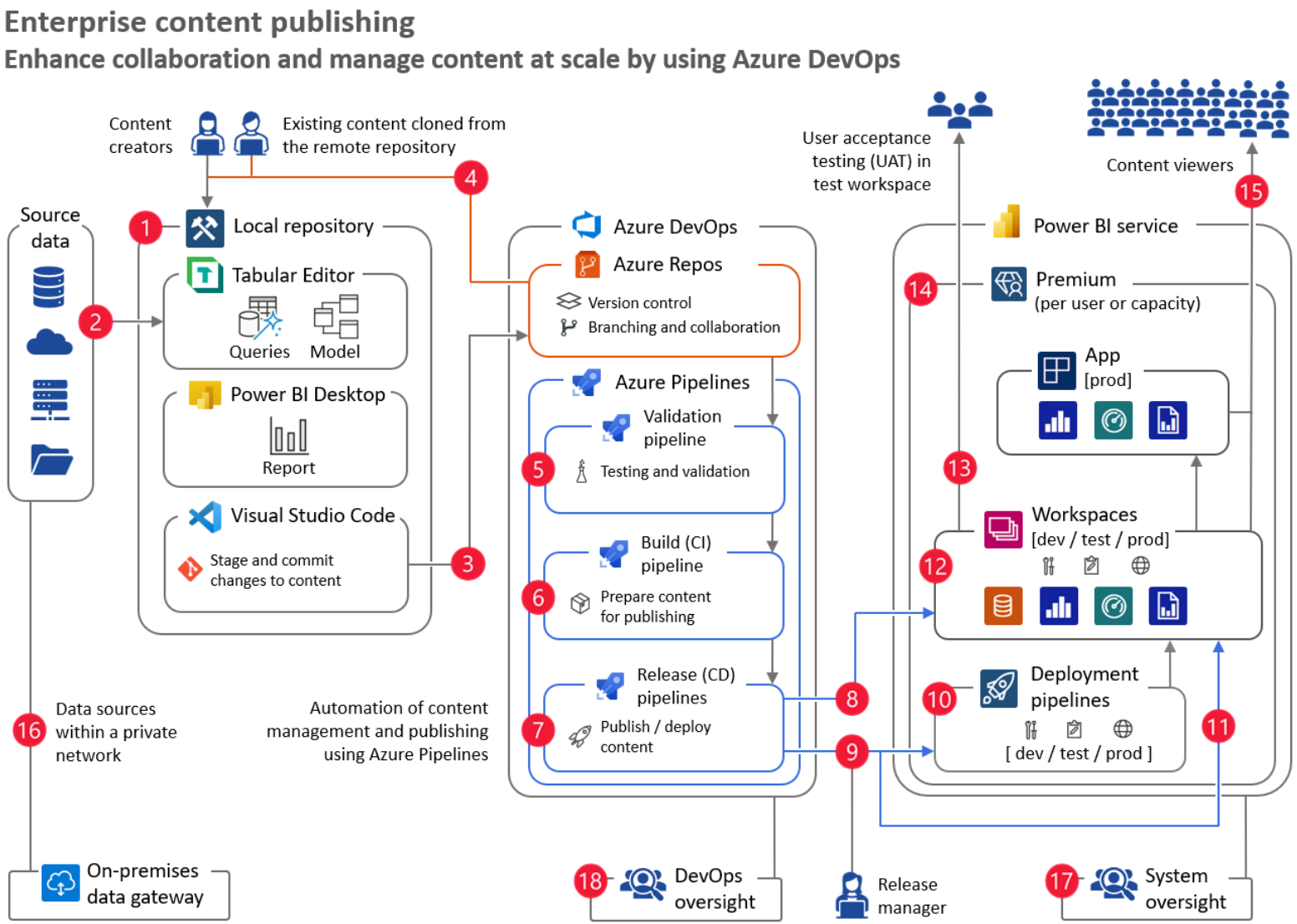
- SCOPE DEFINITION: CLEARLY DEFINE THE SCOPE AND OBJECTIVES OF THE POWER BI INITIATIVE.

- RESOURCE ALLOCATION: ALLOCATE RESOURCES EFFICIENTLY TO MEET PROJECT REQUIREMENTS.

- TIMELINE MANAGEMENT: DEVELOP AND ADHERE TO A REALISTIC PROJECT TIMELINE.

- STAKEHOLDER COMMUNICATION: MAINTAIN OPEN COMMUNICATION WITH STAKEHOLDERS THROUGHOUT THE PROJECT.

Enterprise Architektur – Content Management mit DevOps



Wrap up

- Create a playbook for your Power BI projects and **plan your stages**
- Spend time on **developing a good data model**
- Plan for the future and **make your solutions scalable**
- Git integration on workspace level **versions your work**
- PBIP allows us to save Power BI files ready to **automate and deploy reports as code**
- Collaboration in a team to work on the same artifacts becomes easier with **proper project management**

Q&A