Deploy Power BI as Code.

Professionalizing your solution using Power BI Project Files and Git integration

March 2024

Platinum partners	creates.		In Summa	
Goud partners	Kimura	plainwater de kracht van heldere data		KASPAROV FINANCE & BI
Zilver partners	rockfeather	Dynamic People		GET RESPONSIVE
Brons partners	Hso	macaw	iqbs	VICTA BUSINESS INTELLIGENCE
	Quanto collective analytics	ilionx	valcon	VALID STAY AHEAD
Community partners	broadwick Data & development recruiters	THE DATA COOKS	Tabular Editor	Datamanzi
	Power BI Connector by DAVISTA	MINOVA	AZURRO FINANCE	DATA KINGDOM
	volda; INFORMATIESPECIALISTEN	DashData.	VisionBI Smart Data Experts	easydash

Paulien van Eijk

Data & Analytics Consultant
Macaw Netherlands

linkedin.com/in/Paulien-van-Eijk

PowerBIPrincess.com

FAVORITE STUFF:

Marc Lelijveld

Technical Evangelist | Solution Architect
Macaw Netherlands

Microsoft® MVP Most Valuable Professional

sessionize MOST ACTIVE SPEAKER 2022

sessionize MOST ACTIVE SPEAKER 2023

@MarcLelijveld

linkedin.com/in/MarcLelijveld

Data-Marc.com

FAVORITE STUFF:

After this session

Challenges	File formats	Git	Deployment
Understand challenges working with multiple developers on the same Power BI solutions	Understand which file formats Power BI supports and explain the differences and advantages of each	Understand how Git can help version your solutions, collaboration and branching of solutions	Deployment patterns using the new file format and / or Git integration

Report Development Cycle

Publishing your report online

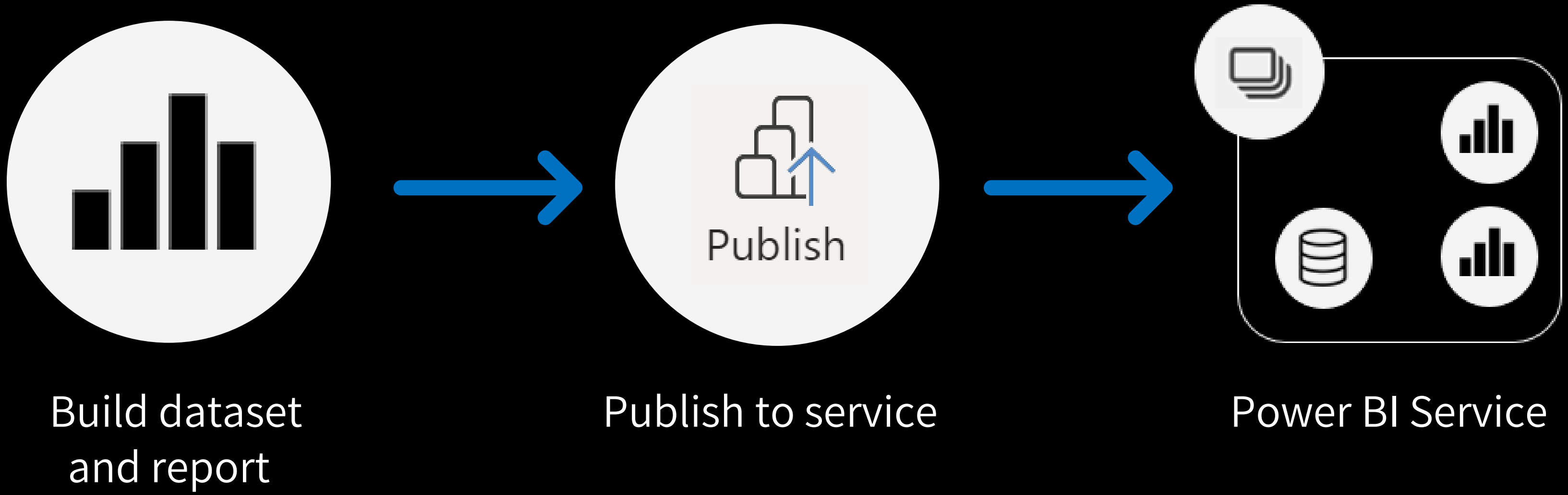
Gather

Clean

Model

Visualize

Publishing your report online



Who is using this development cycle as their way of work?

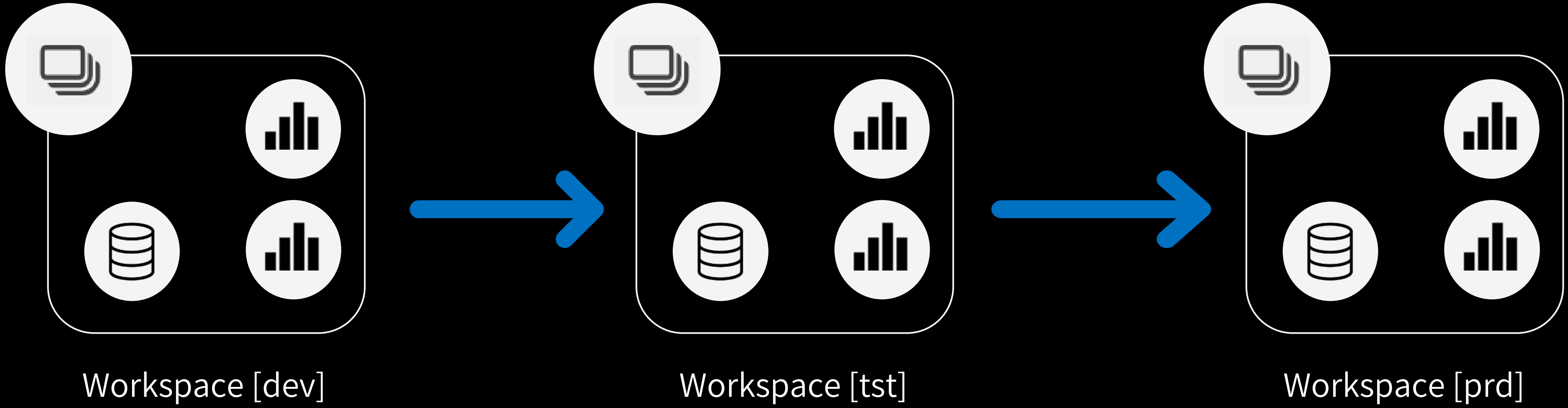
What are your experiences?

☹ Everybody?

Things you might have encountered

- Collaboration is difficult
- Keeping track of changes is (almost) impossible
- Download report from service to get latest version
- Publishing a previous version

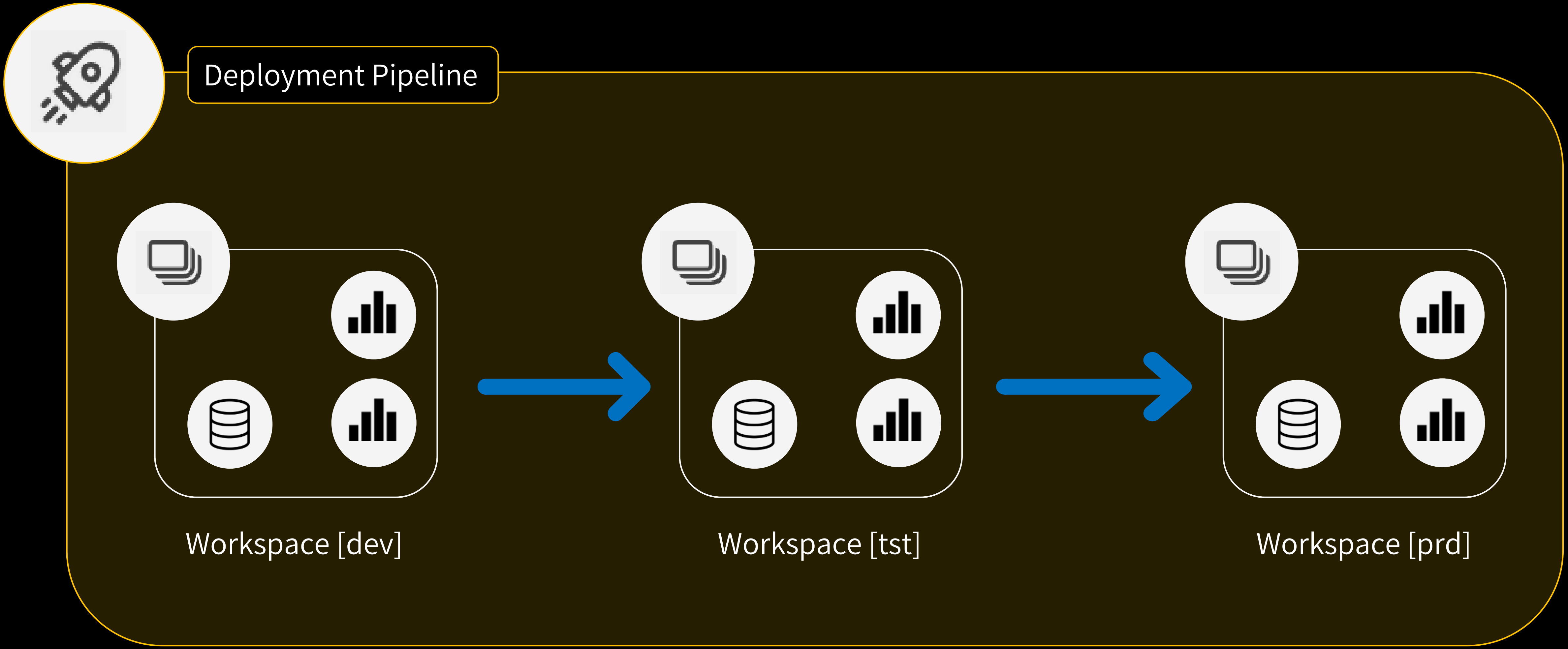
Working in stages (DTAP)



Things you might have encountered

- Accidently deploying to production instead of development.. whoops
- Forgot to change data source connection from dev to prod
- Overwriting data in production

Things got a bit better



Demo

What improved / can be avoided?

- Collaboration is difficult
- Keeping track of changes is (almost) impossible
- Download report from service to get latest version
- Publishing a previous version

When using DTAP:

- Accidently deploying to production instead of development.. whoops
- Forgot to change data source connection from dev to prod
- Overwriting data in production

New file format: .pbip

New file format: .pbip

- Power BI Project file
- Saving report and semantic model artifacts in separate plain text files in a clear folder structure
- Introduced in June 2023, but still in preview

Why should we care?

Enables capabilities, such as:

- Editable format: Easily make changes using code editors
- Source Control: Track version history, compare versions, revert to previous versions
- CI / CD: Quality controls (review, testing) before deployment to production

Demo

How do we enable the other benefits?

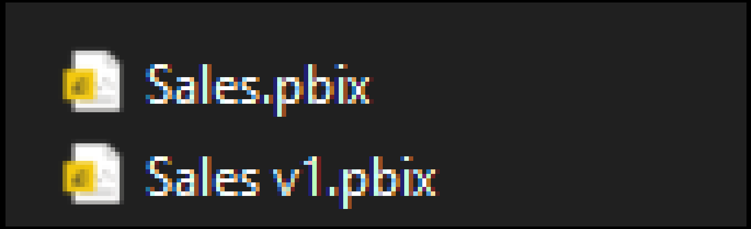
- Editable format: Easily make changes using code editors
- Source Control: Track version history, compare versions, revert to previous versions
- CI / CD: Quality controls (review, testing) before deployment to production

Source Control

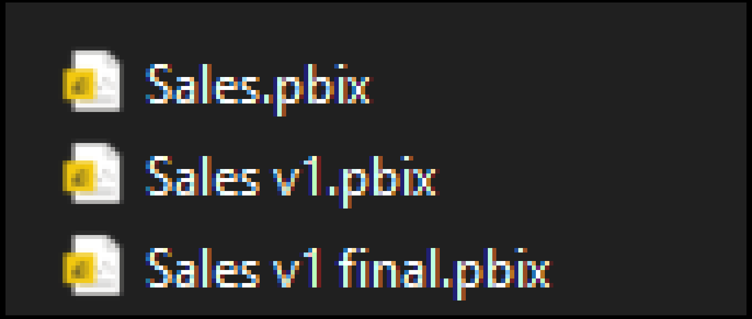
Is there a better way of versioning than..

Sales.pbix

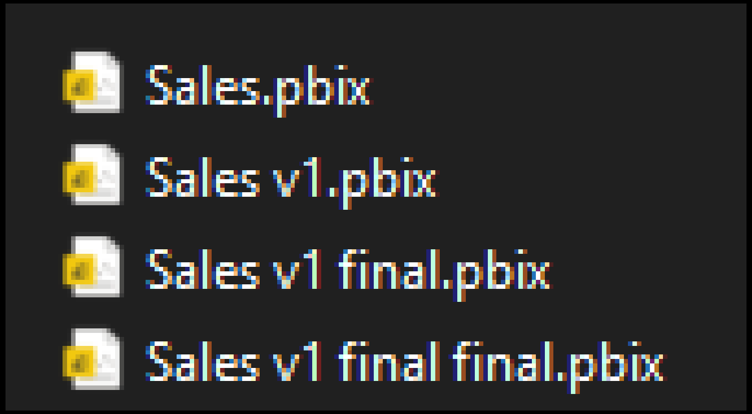
Is there a better way of versioning than..


Sales.pbix
Sales v1.pbix

Is there a better way of versioning than..

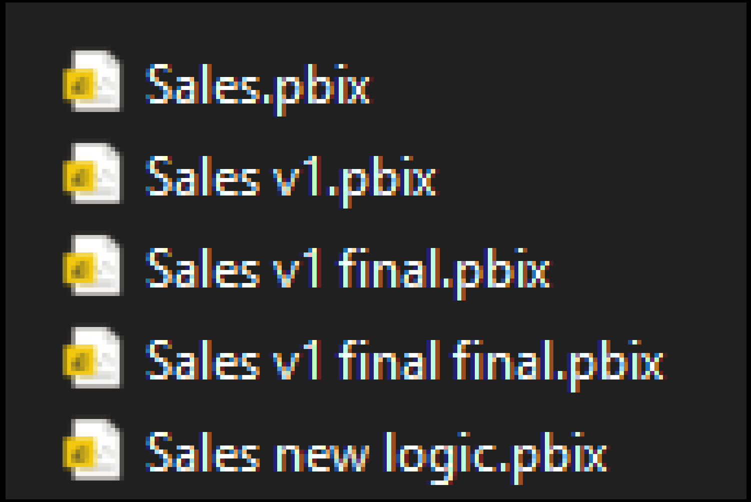

Sales.pbix
Sales v1.pbix
Sales v1 final.pbix

Is there a better way of versioning than..



Sales.pbix
Sales v1.pbix
Sales v1 final.pbix
Sales v1 final final.pbix

Is there a better way of versioning than..



Sales.pbix
Sales v1.pbix
Sales v1 final.pbix
Sales v1 final final.pbix
Sales new logic.pbix

Options

- SharePoint
- OneDrive

But only track the binairy file as a whole. So, we don’t know;

- When we deleted that one table?
- When we introduced that issue in our measure…
- Etcetera.

But we are talking about ‘professionalizing’ – so let’s take it to the next level…

GIT ALL THE WAY!!

Who has used Git before?

Git, what is it?

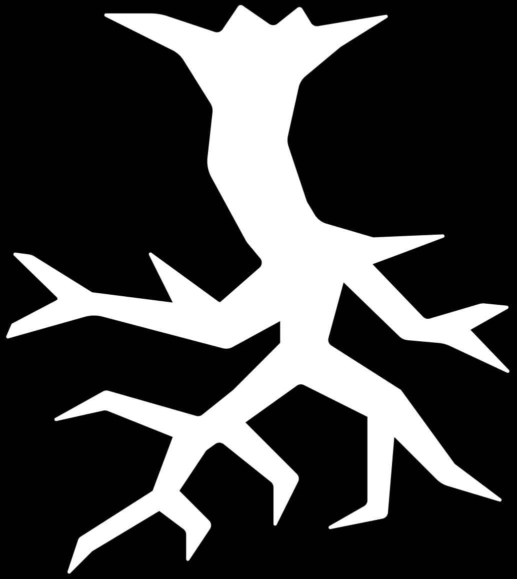
Git is a version control system to **track and manage changes**

It provides functionalities for:

- Version control
- Collaboration
- Tracking changes
- Compare versions

But how?

Branching

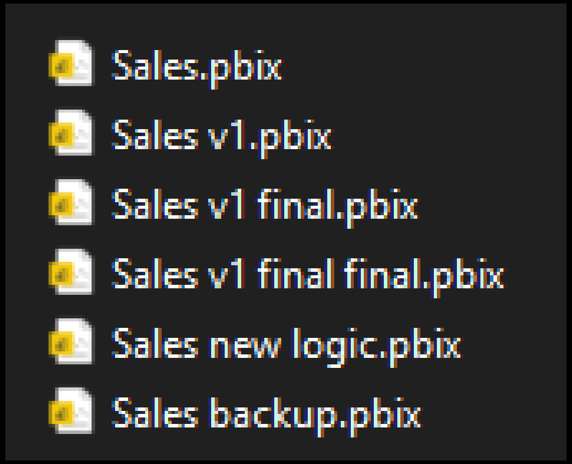


Merging

Branching – General concept

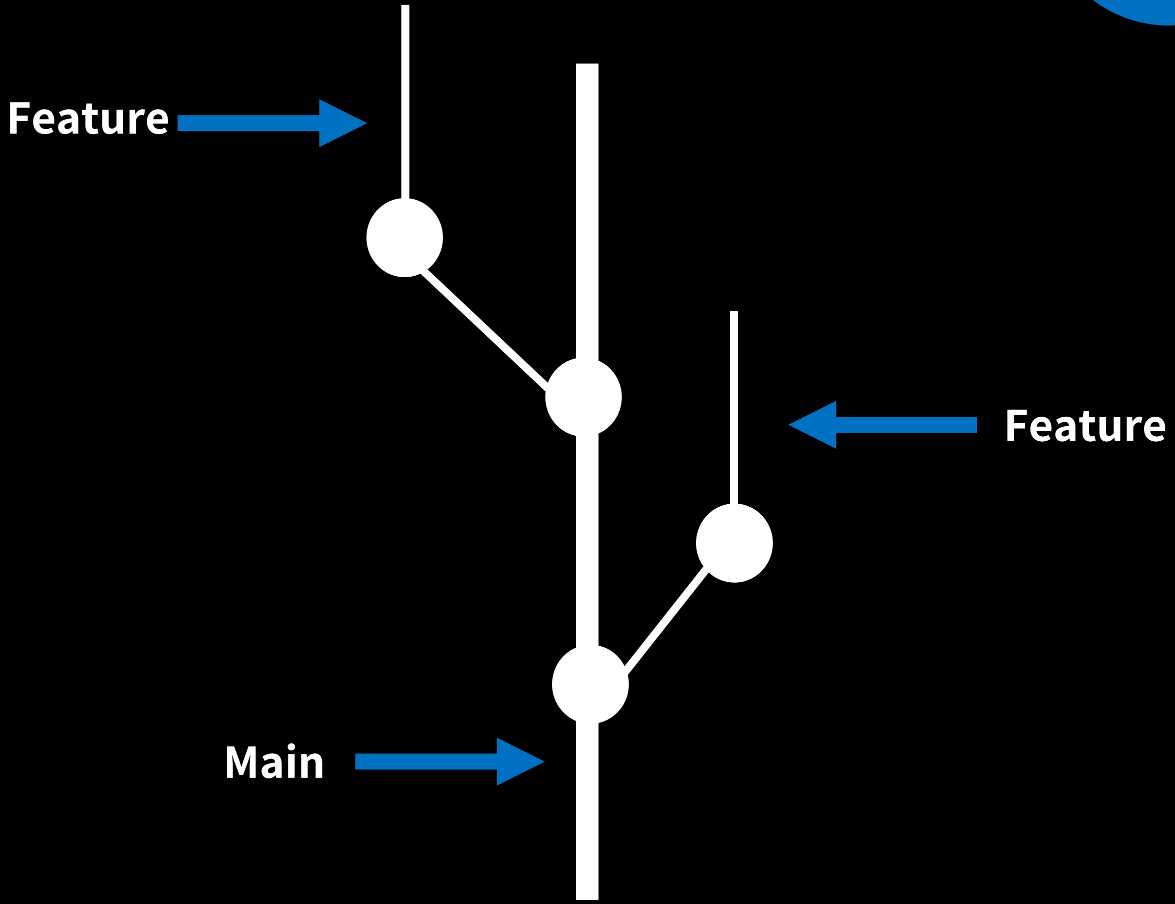
- Isolate development workflow
- Safely create new feature / fix bug
- Copy of code, without modifying "production",
- Test before saving to "production"

- Without the need for:



Sales.pbix
Sales v1.pbix
Sales v1 final.pbix
Sales v1 final final.pbix
Sales new logic.pbix
Sales backup.pbix

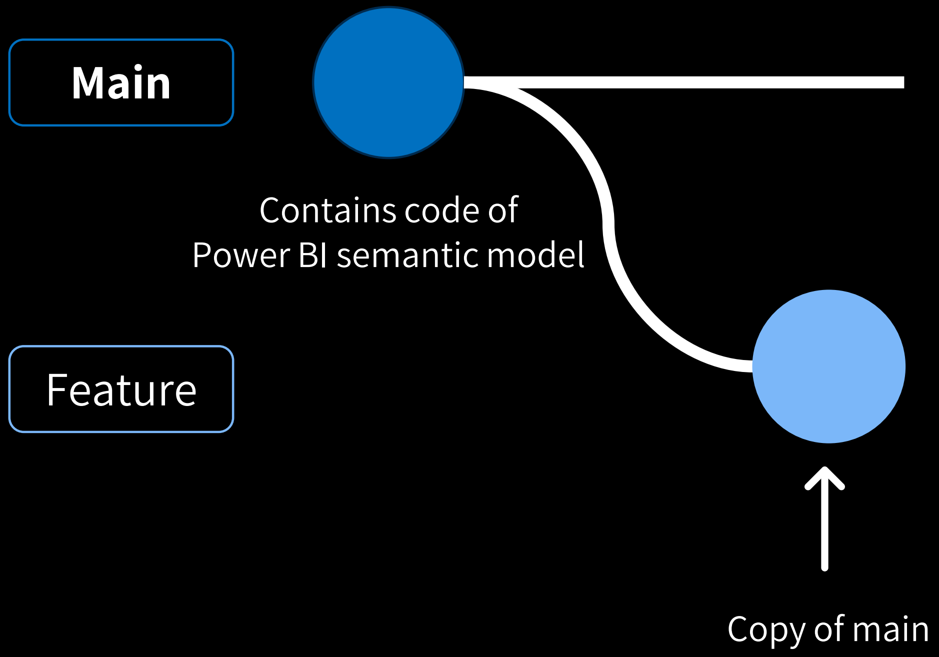
But how?



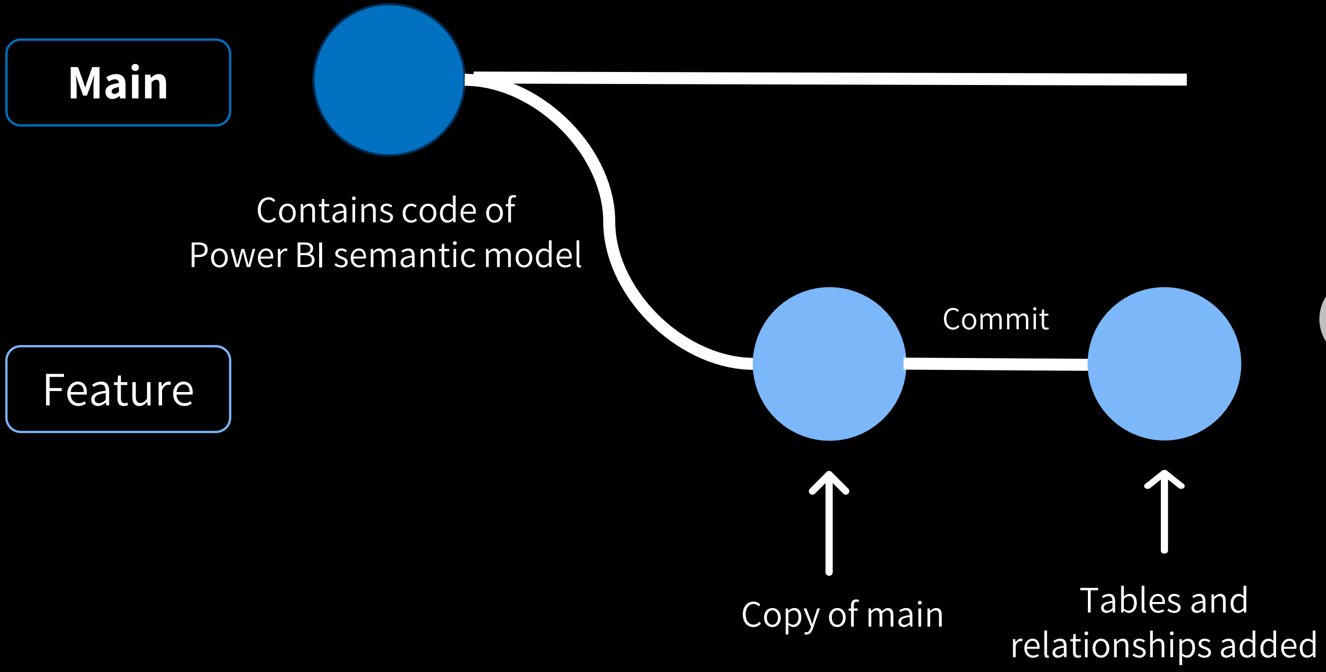
Branching

Contains code of Power BI semantic model

Branching

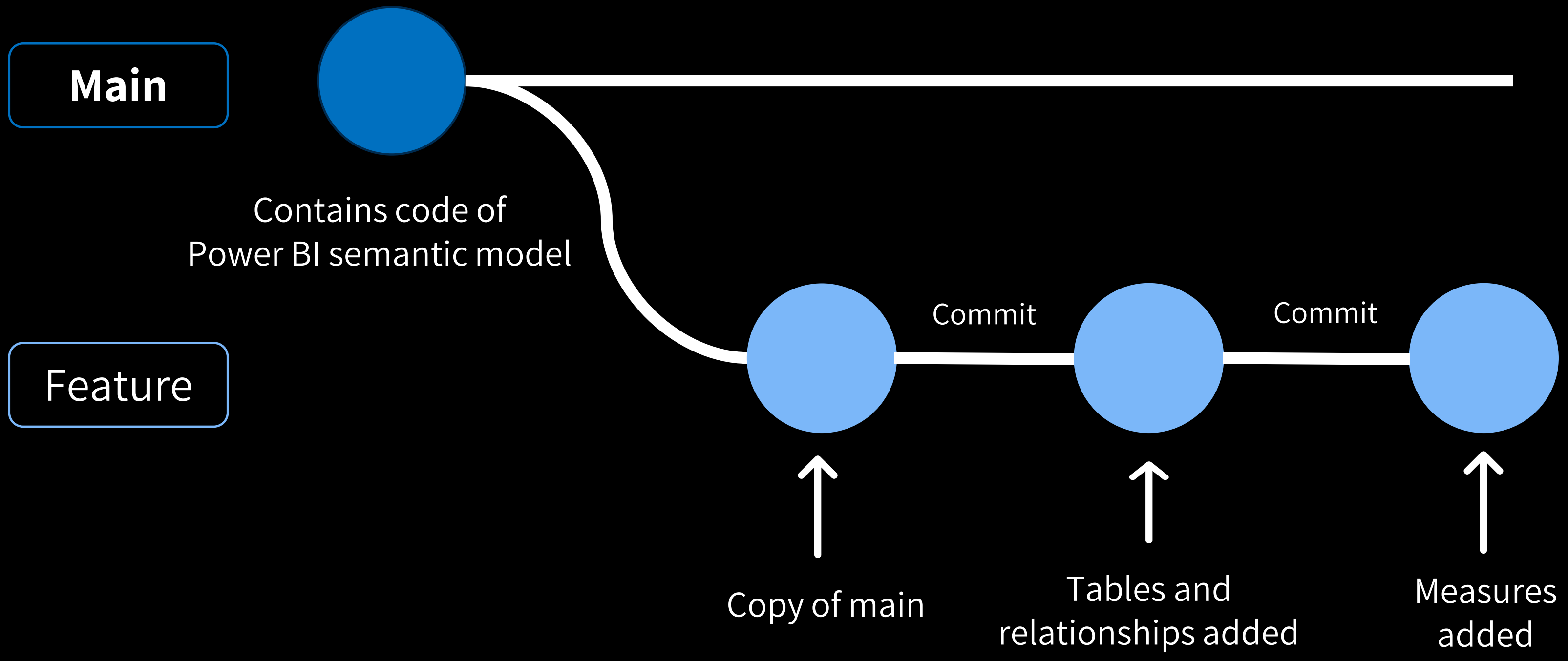

Main
Contains code of
Power BI semantic model
Feature
Copy of main

Branching



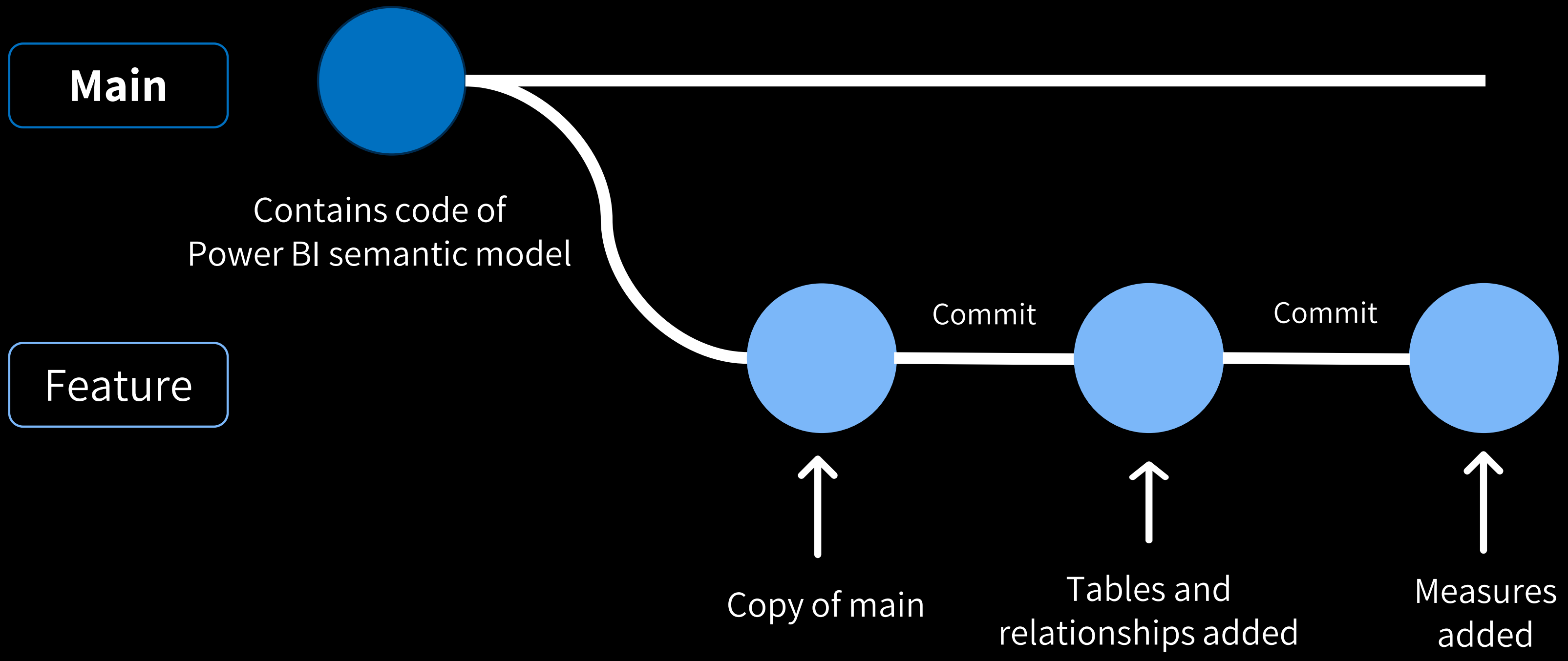
Commit: Creates snapshot of the repository. Committed snapshots are like clicking the "save" button in Power BI Desktop

Branching


Main
Feature
Contains code of Power BI semantic model
Commit
Commit
Copy of main
Tables and relationships added
Measures added

Demo

Branching

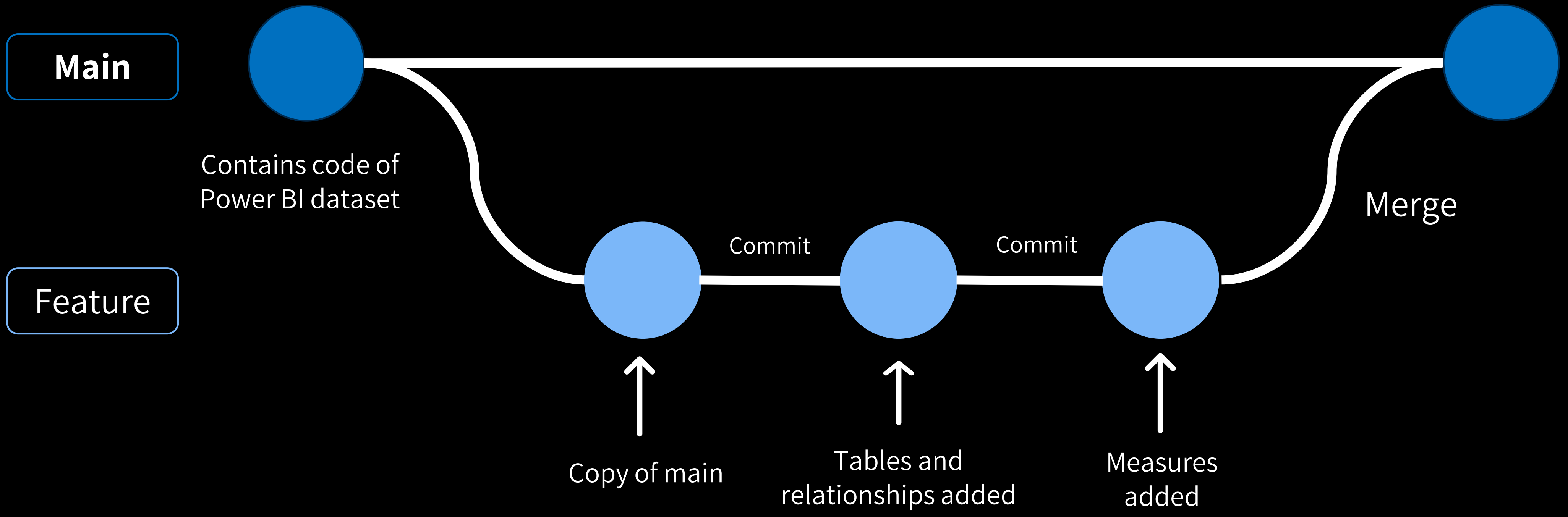

Main
Feature
Contains code of Power BI semantic model
Commit
Commit
Copy of main
Tables and relationships added
Measures added

Done developing ☺
Now what?

Merging

- Take the main branch and the feature branch and create one single source of truth.

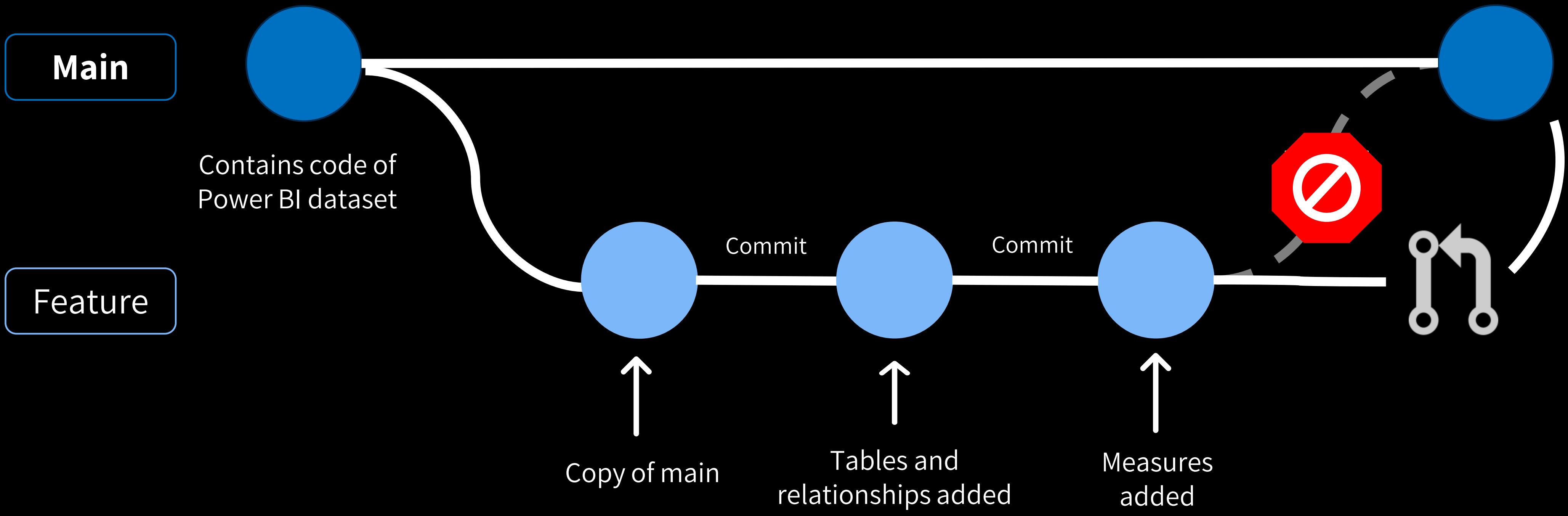
Merging


Main
Feature
Contains code of Power BI dataset
Commit
Commit
Merge
Copy of main
Tables and relationships added
Measures added

Considering main is our production environment...

Do we bring our changes directly to production?

Pull Request



Main
Feature
Contains code of Power BI dataset
Commit
Commit
Copy of main
Tables and relationships added
Measures added

Pull Request (PR)

Protect your main branch by defining branch policies. Nobody can directly commit to Main or approve their own work in a pull request.

With a PR, we realise:

- Validation of work
- 4-eye principle (or more)
- Test code as part of PR

Demo

Wasn’t this already possible?

- Yes! Branching and merging was already possible before.
- But, it was in an unreadable file format, called .pbix.
- Dataset and reports were not seperated → one big file
- Dataset contained data
  - Except if you seperately upload a model.bim / xmla
- PBIX files are often too large (volume wise) which required Large File System (LFS) to be enabled on the repository → LFS = anti-pattern

🍺 or 🍷 —— Scenarios

Choices…

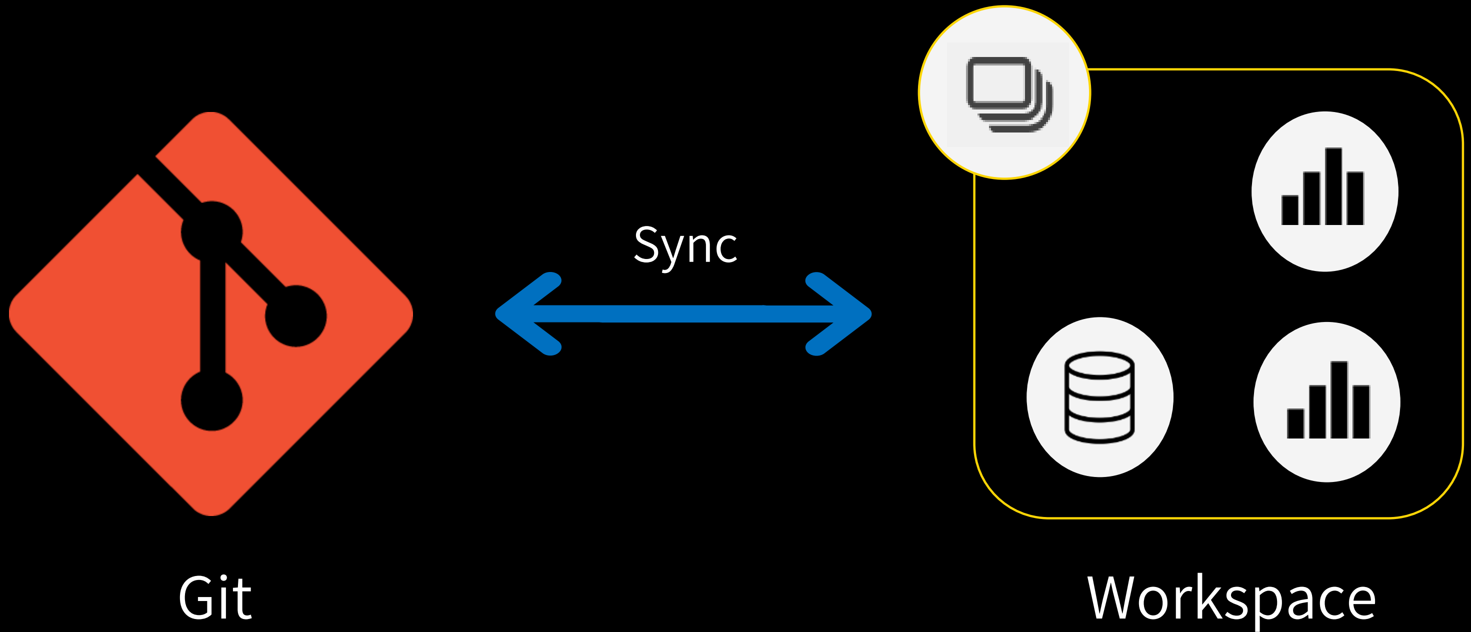
Solely versioning in Git

Git as your source control and versioning system – either locally or in the cloud

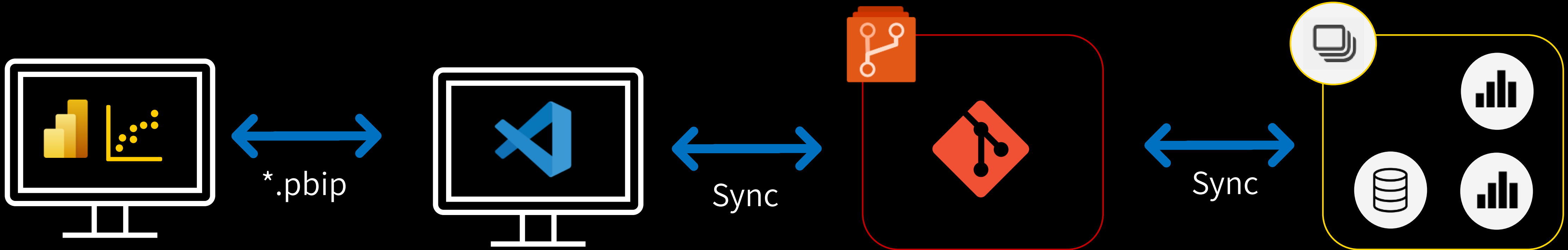
Connect Git to Power BI

Git Integration **in Power BI service** with Azure DevOps as our source control and versioning system

Git Integration with Power BI

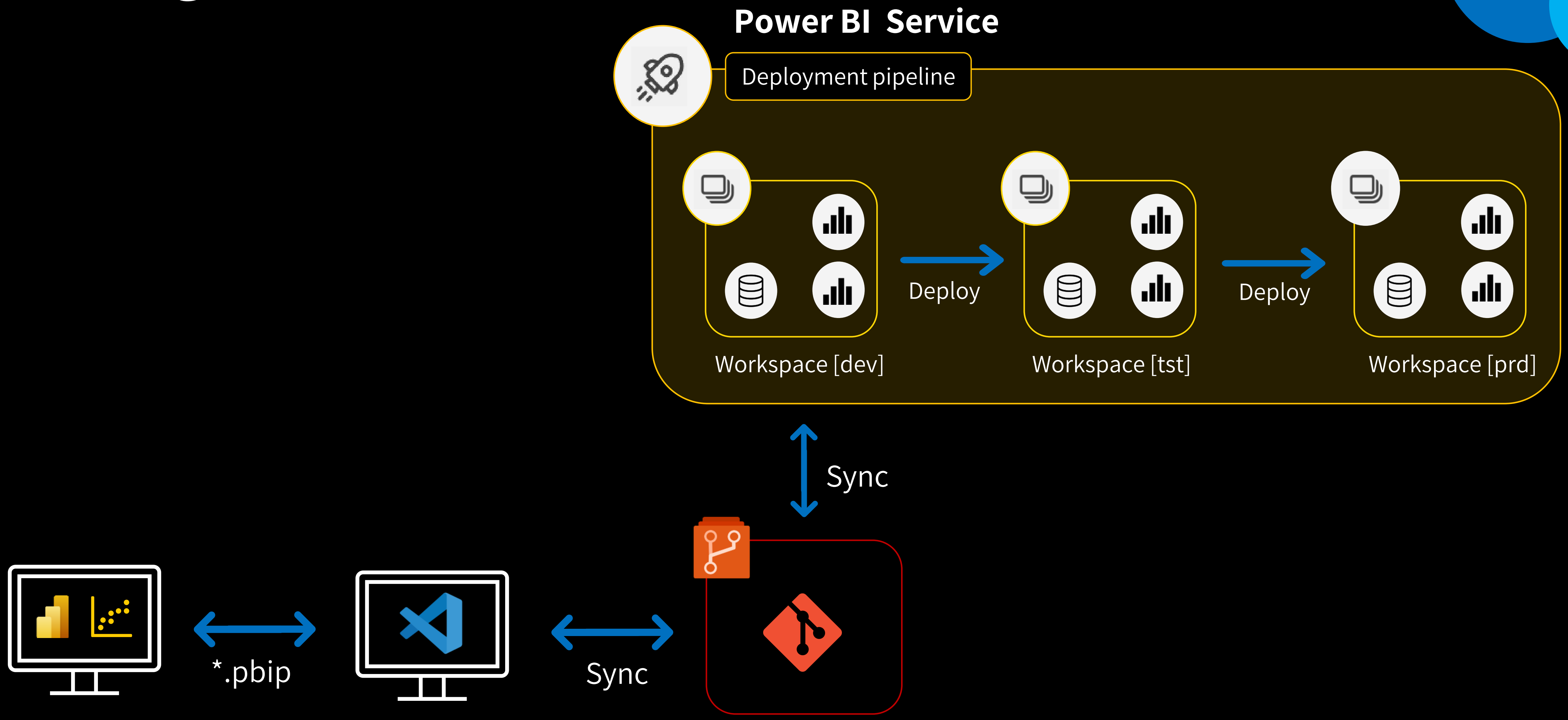


All together

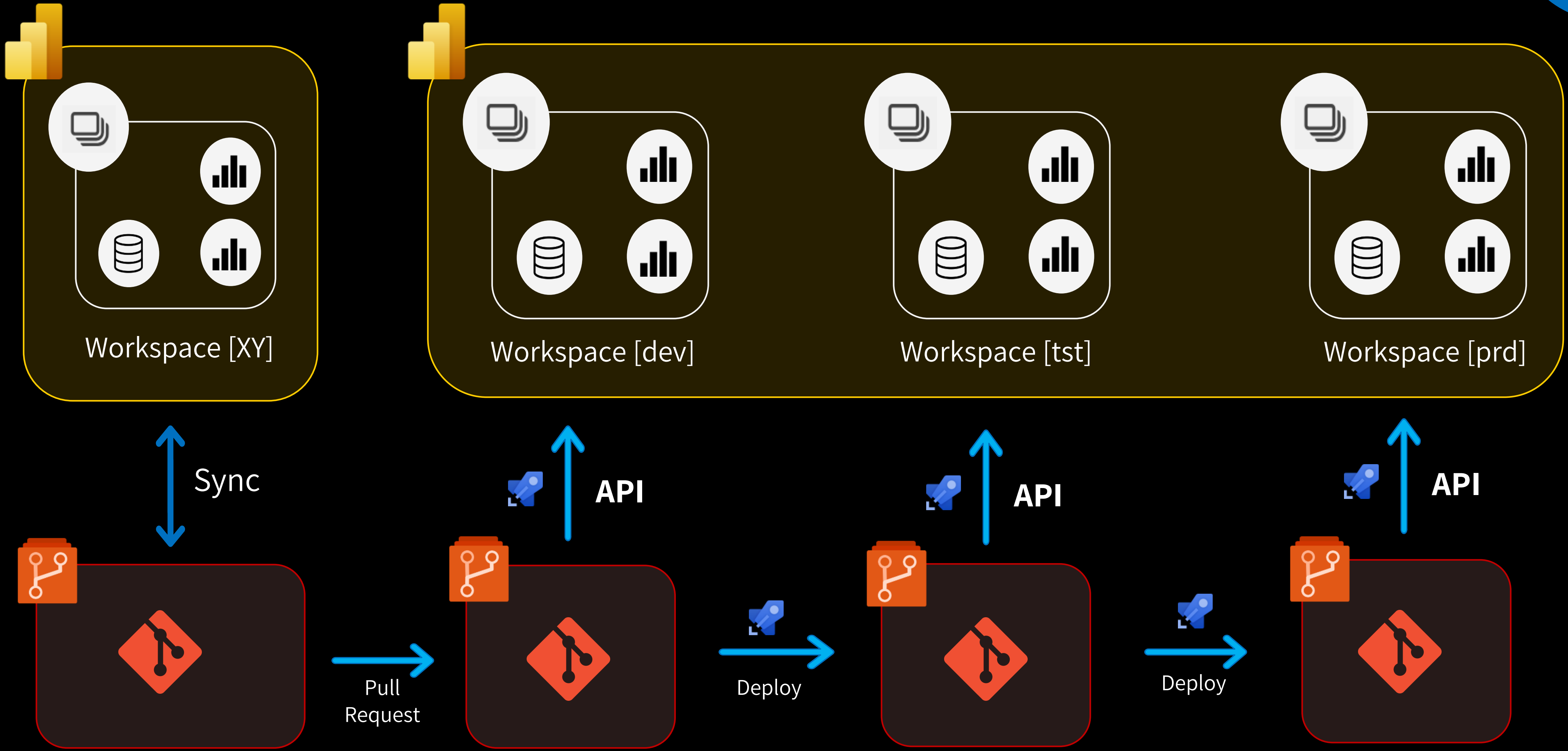


Demo

All together



All Together – Orchestration via Azure Devops



Demo

Combine scenarios

- It’s not as black or white as the solutions presented
- There are many ways to deviate from this design to make this way of work suitable for your organization.

Wrap up

With git integration, we get a **real developer experience** for Power BI solutions

Git allows to check in changes based on code and **track changes**

Choose the **best scenario** for your situation

Although it is not perfect (yet?) it has some **great potential** going forward

Setup your own playground/test environment to **get familiar** with the concepts

Q&A

Session evaluation

Event evaluation