FINAL PROJECT: “ADVENTURE WORKS DASHBOARD AND EXPLORATORY DATA ANALYSIS”

**Dmitriy Kirichenko**

**Ignacio Beneyto**

**Luis Llobell**

Data Science Degree

**Project Big Data ll**

Professor: Alexander PERDIGUERO O´LEARY

Index

[Index 1](#_Toc1620809177)

[About the project 3](#_Toc1564800909)

[Problems 4](#_Toc2094950526)

[Creation of working environment 5](#_Toc262576518)

[Git repository creation 5](#_Toc614971670)

[Git repository access 6](#_Toc1965981297)

[Data extraction 7](#_Toc246948626)

[Finding the source data 8](#_Toc1793033650)

[Restoring the backup version 8](#_Toc1260341568)

[DB first look 9](#_Toc791504930)

[Extracting the table’s relationships 10](#_Toc93924976)

[Desired format 11](#_Toc862528515)

Table of contents

# **No table of figures entries found.**

# About the project

The project was assigned to an entire group with the goal of achieving certain set results. We would need to achieve an advanced level of quality, work ethics, advanced theoretical and practical knowledge applications.

The path of making this project should look as follows:

1. Creation of suitable environment for parallel work as a team.
2. Data extraction from the corresponding sources.
3. Data wrangling.
4. Exloratory Data Analysis.
5. KPI’s and questions assignment.
6. Creation of Data Warehouse.
7. Data streaming.
8. Predictive Model Training.
9. Creation of user-friendly Power Bi dashboard.
10. Final documentation commitment.

# Problems

When using GitHub to collaborate on a project that involves connecting from three different computers, performing Exploratory Data Analysis (EDA), running SQL queries, and working in Power BI, several challenges and potential issues could arise:

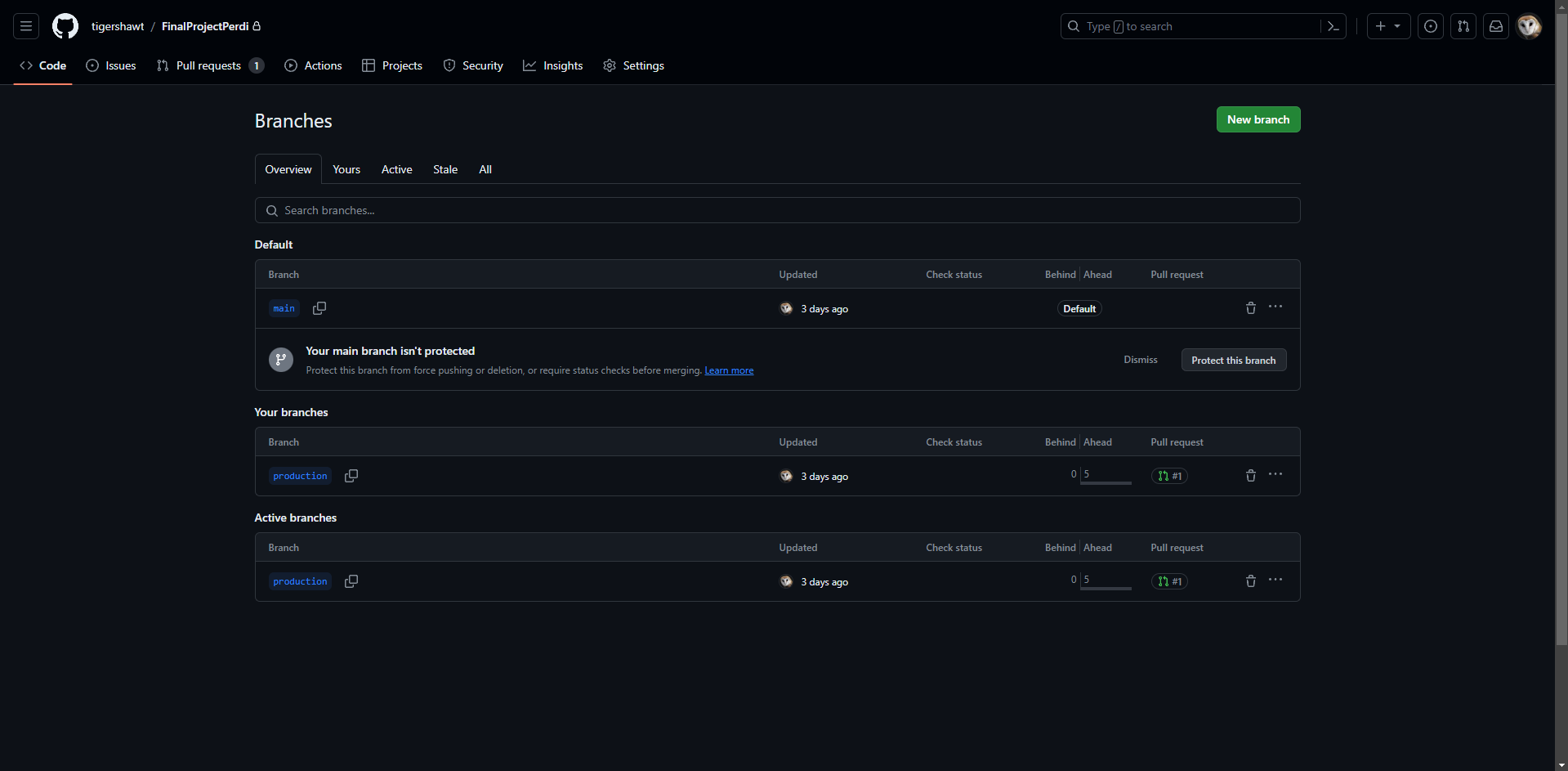
* Version synchronization: It is essential to ensure that all team members are working with the same version of code and files. Version conflicts can arise if changes are not synchronized correctly.
* Merge conflicts: When multiple people are working simultaneously on various parts of the project, conflicts may occur when merging changes. This may require manual resolution and communication between team members.
* Concurrent access to shared resources: If multiple people try to access and modify the same files simultaneously, locking and conflict issues can arise. It is important to establish protocols to avoid such situations.
* Data security and privacy: When working with sensitive data, such as SQL queries that could contain sensitive information, it is vital to ensure data security and privacy. Appropriate security and encryption measures must be implemented.
* Performance and scalability: Depending on the size of the project and the amount of data involved, system performance and scalability can become issues. It is important to monitor and optimize system performance, as necessary.
* Tool and version compatibility: Ensuring that all team members are using compatible versions of necessary tools, such as SQL and Power BI, can avoid incompatibility issues.
* Communication and coordination: Effective communication between team members is key to ensuring everyone is aware of changes, issues, and pending tasks. Clear communication channels must be established, and a log of activities must be kept up to date.
* Training and technical knowledge: Some team members may not be familiar with all the tools and technologies used in the project. Providing training and technical support can help mitigate this problem.

# Creation of working environment

## Git repository creation

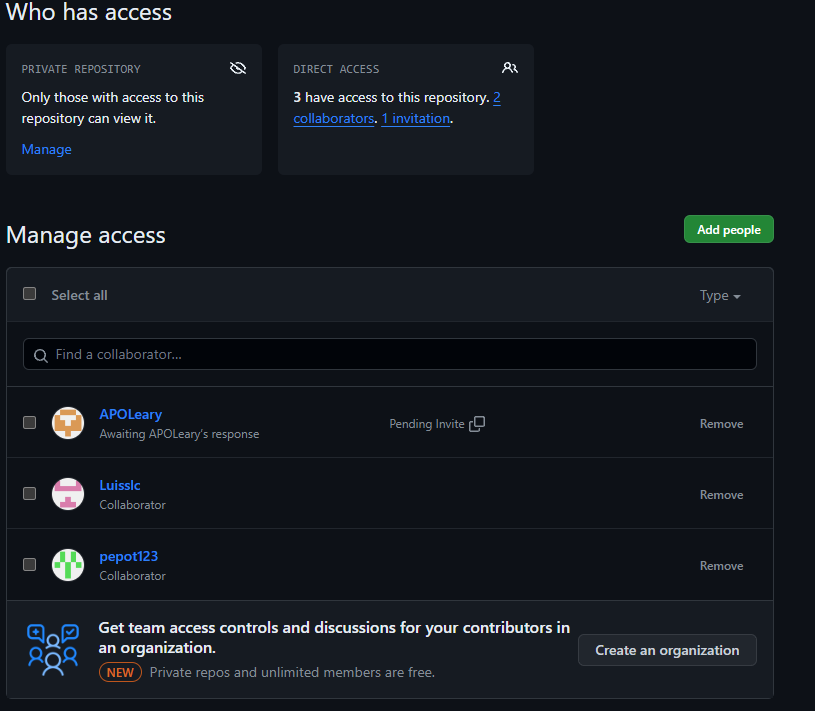
First, it is vital to create sharable working space with logs and version controls to be able to do work simultaneously and test it. For this reason, we have chosen GitHub (<https://github.com/>) since it is the most advanced and reliable resource available on the internet for free. The material part of this project is not sponsored at any means, so we must work with null budget.

We have created the repository called “FinalProjectPerdi” where every piece of work from members of the team would come together and be version controlled in order if someone would commit any mistakes. For this reason, we have created additional to a main brunch called “production” where all the changes would be revised before merging the changed or added files into the main branch.



## Git repository access

We have made the repository the closed one, and only team members with permissions would have access to it. All the members have the same rights. They need to identify themselves from a local instance to proceed with the git functionality in our repository (Later will be explained in detail).



Repository roles:

Dmitriy Kirichenko - tigershawt – owner

Ignacio Beneyto – pepot123 – collaborator

Luis Llobell – Luisslc – collaborator

Alexander PERDIGUERO O´LEARY - APOLeary – collaborator

# Data extraction

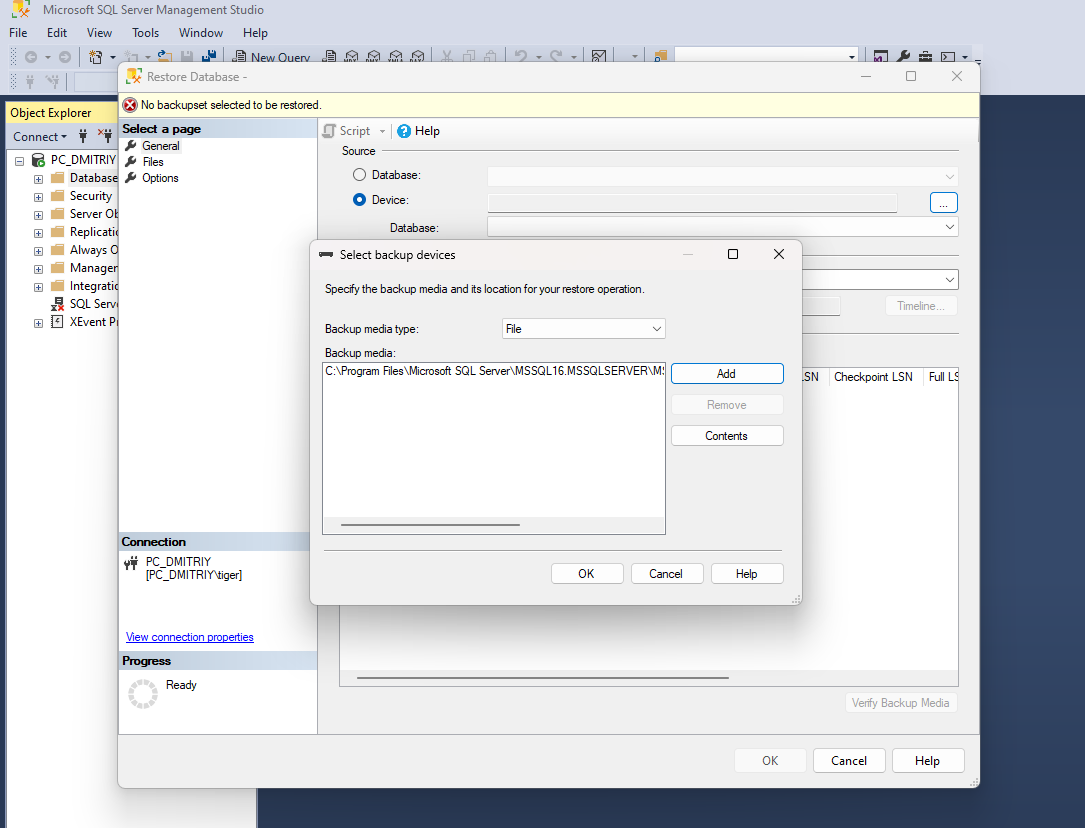
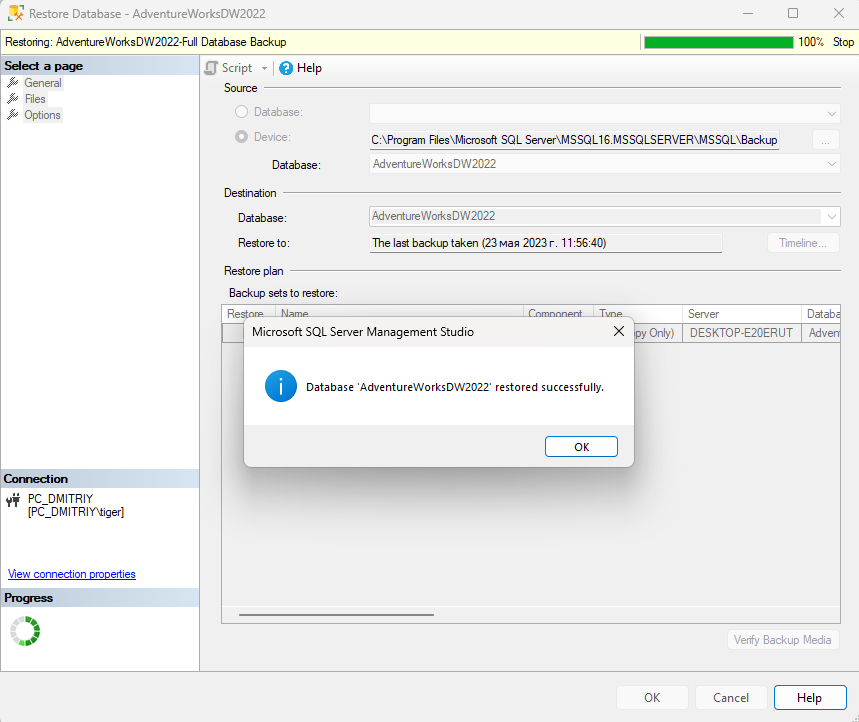
## Finding the source data

Finding a source data was a trivial task. First search in a google would give the needed result. Adventure Works database is available as a backup file in many places, we have downloaded it through GitHub. The version we have chosen would be the latest since we hope it would be the most saturated with data insights. The file was “AdventureWorks2022.bak” of (OLTP) full database backup (<https://github.com/Microsoft/sql-server-samples/releases/tag/adventureworks>).

## Restoring the backup version

We have followed the guide from Microsoft official website (<https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms#download-backup-files>) that suggests using “Microsoft SQL Server” to restore the backup version. We have chosen the local SQL environment (https://www.microsoft.com/es-es/sql-server/sql-server-downloads) to do not use any cloud solutions which might be costly. This software is needed to launch local SQL server to proceed with the backup.

To complete the backup, we have installed SSMS to be able to finally restore the DB, manipulate with the data, and clearly see the internal structure for better understanding. All these steps are described in the same guide from the closest link above.

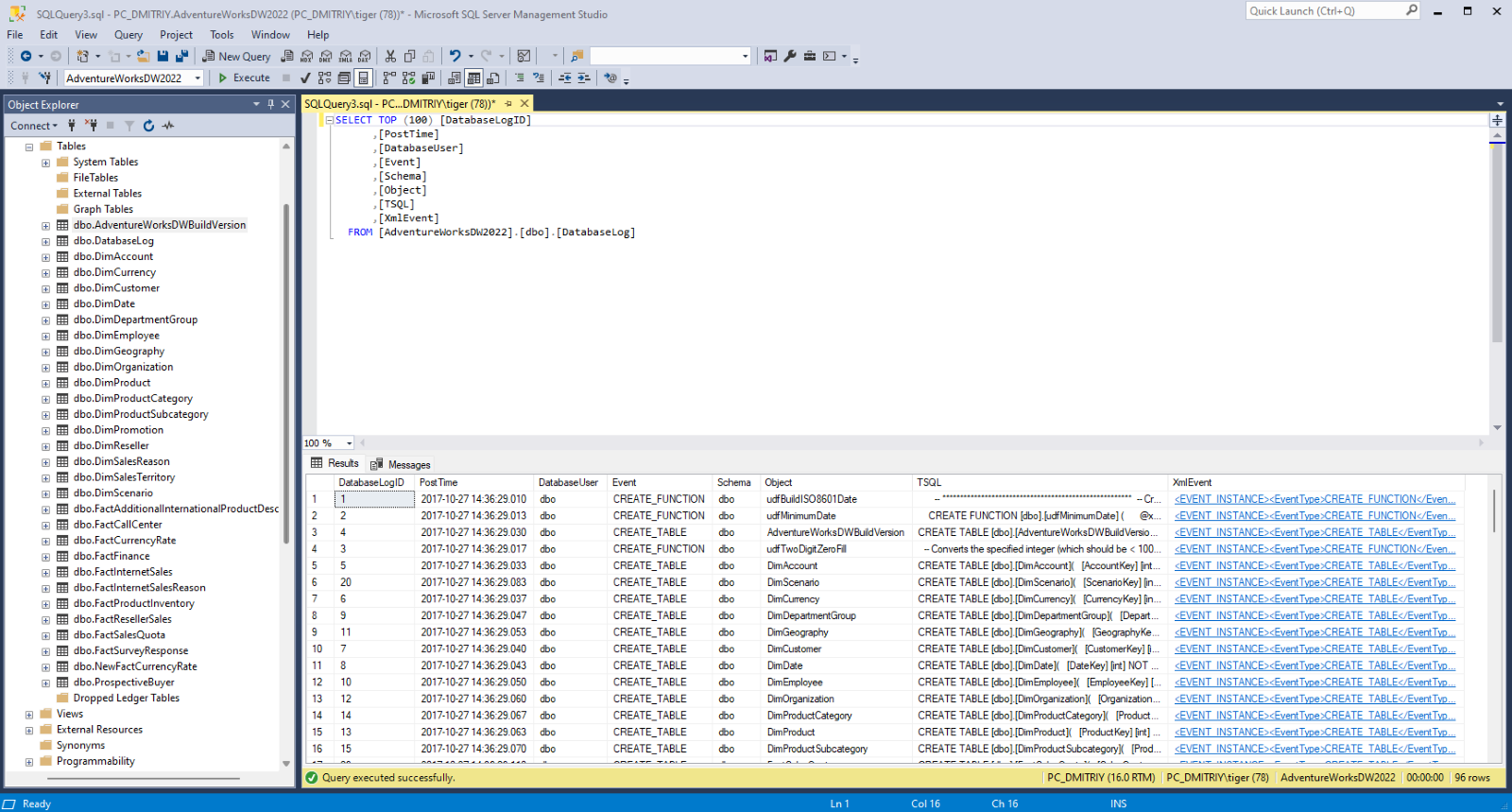
  


DB first look

After making the backup, we see the entire database, since we only need to data to work with, we do exploratory visual analysis of all the data tables and see what we might need in our project and what we would not. However, we have taken all the tables except the two:

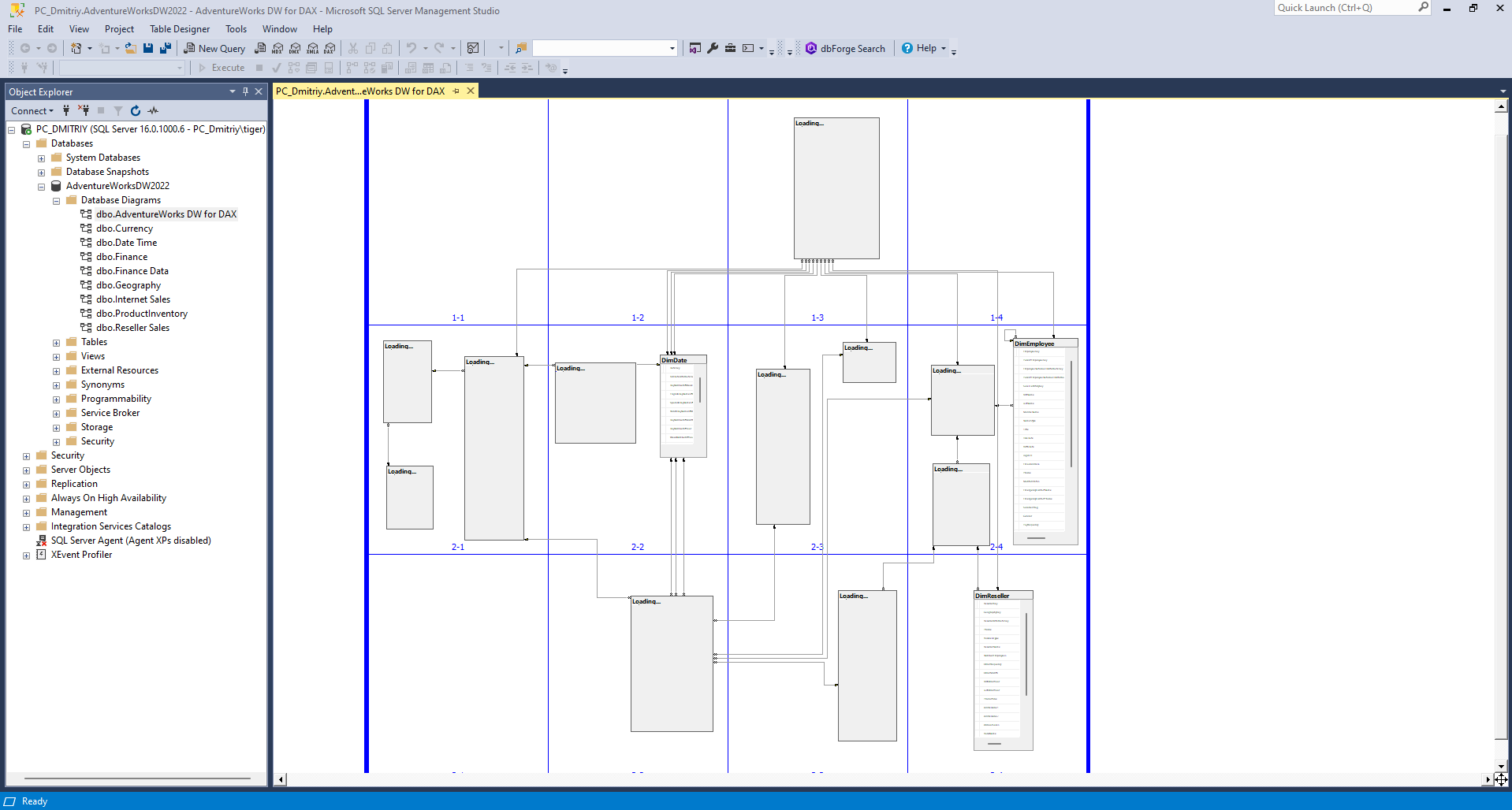
AdventureWorksDWBuildVersion – The table with 1 parameter stating the version of the database and the time it was patched.

DatabaseLog – History of changes made to various parts of the database stating time, action, and user.



Extracting the table’s relationships

To understand data at an appropriate level, after analysing all the database, we have found the relational scheme, where all the primary and foreign keys relationships are clearly stated. This will be useful in future EDA operations such as “table join”. We have captured all the available diagrams with windows “Snipping Tool” and have placed them into out GitHub repository.





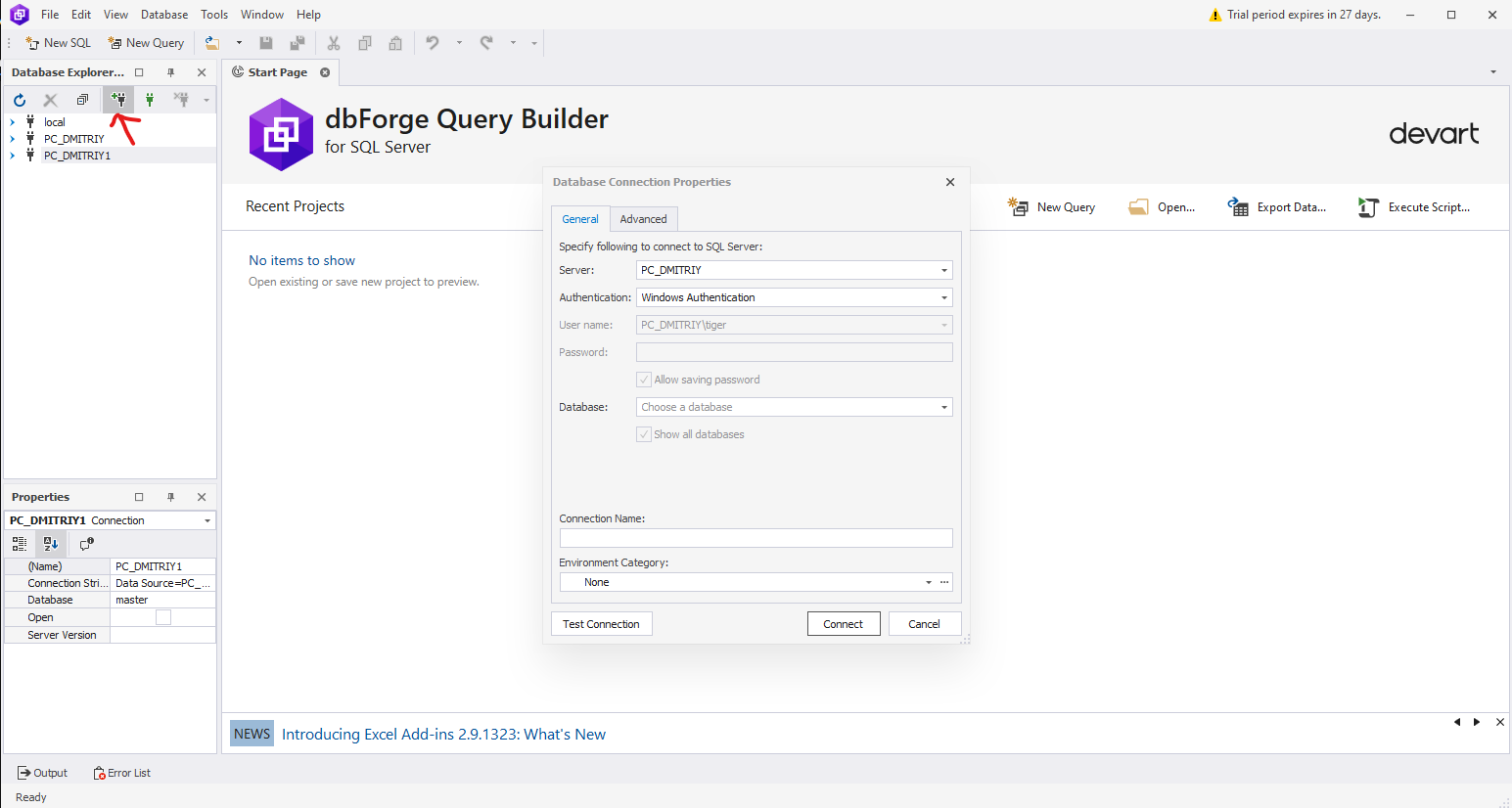
## Desired format

We are Data Scientist and primarily work with python and csv files, we have investigated ways of extracting the tables into csv files. We tend to believe, that we have found the most appropriate way of doing it.

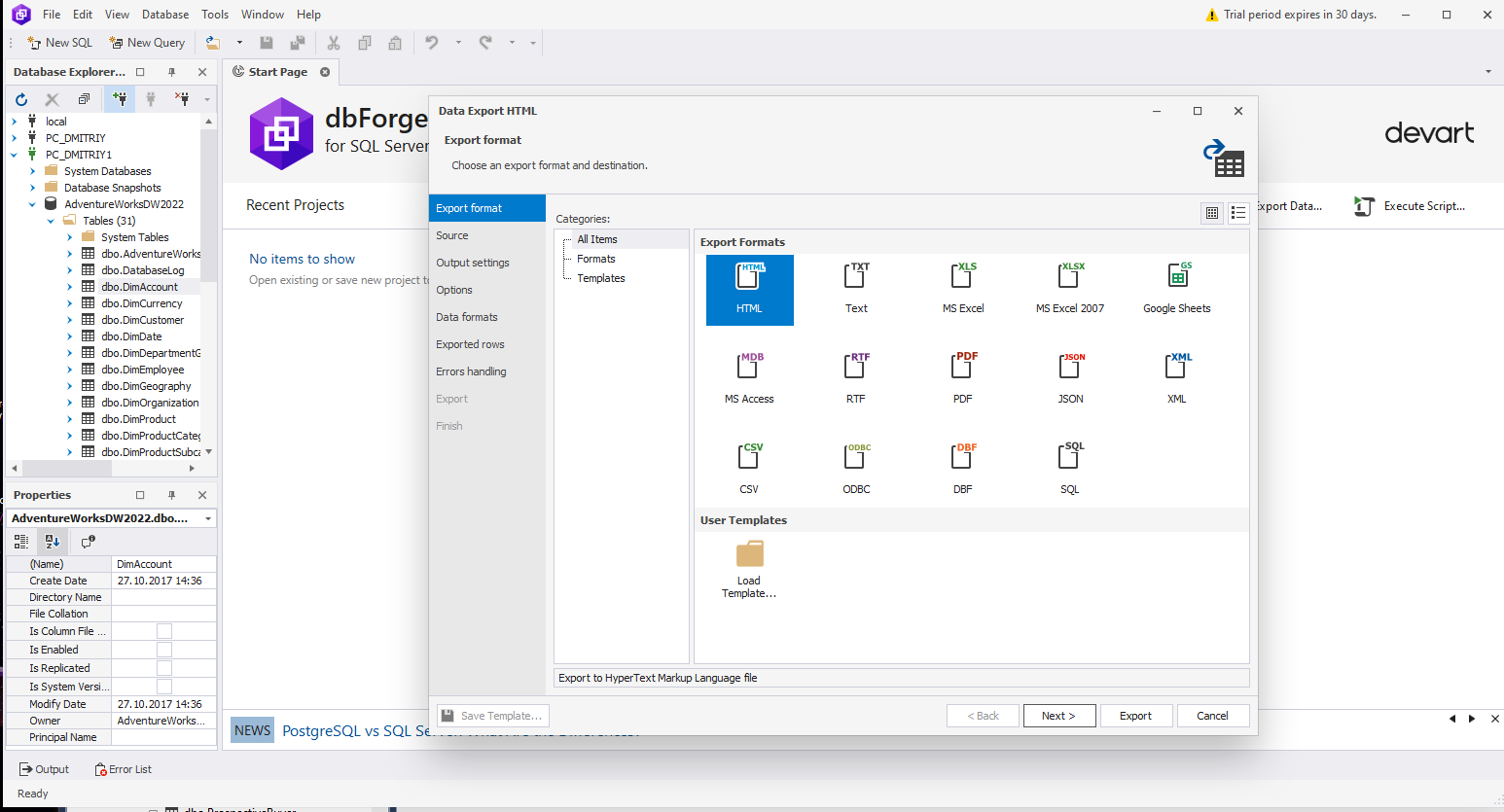
Since SSMS does not have the default option of converting the SQL relational database into CSV with keeping all the header and rows in place. We have found the plugin that significantly helped us and saved lots of time alternatively parsing through the available on GitHub csv (<https://github.com/microsoft/sql-server-samples/tree/master/samples/databases/adventure-works/oltp-install-script>) containing data and rules for SQL server.

The tool, which is called Data Pump from Devart, was easy to use and of quality. We have followed the following documentation from the developer (<https://docs.devart.com/data-pump/exporting-data/export-to-csv.html>). Attention this documentation was outdated since it does not show the integrated buttons in SSMS itself. So, we had to find alternative way of using this Data Pump.

After downloading and installing the free trial version of “dbForge SQL Tools Standard Trial” we need to find the application in our system called “dbForge Query Builder for SQL Server”. After opening the app, we would need to connect to our running SQL server by choosing the credentials from the image below. (red arrow shows the button to initialize new connection).



After we have established connection, we can keep up with developers' documentation and using the Data Pump.



Extracting all the necessary tables into our git repository.

