**Candidate: Mehdi Malekzadeh Email:me.malekzadeh@gmail.com**

**Proposed BI Solution:**

**Assumption:**

I assume that the data from SQL Server and Postgre SQL is a query that is collected through various tables in those databases and then placed into one single table through multiple columns daily basis. Therefore, I assume that data records are shows the number of customers of each business entity for every single date.

I have designed the end-to-end BI solution using Microsoft Power BI dataflows (online service) and Microsoft Power BI Desktop to do this assessment.

**Introduction:**

Power BI Dataflows are a feature of Power BI that allows organizations to unify data from various sources, prepare the data for consumption, and publish it for users to consume in Power BI. The main idea behind dataflows is doing the Extract, Transform and Load (ETL) processes in an easy way. It can connect to multiple data sources such as Excel and familiar databases, then transform the data within a web browser, so it is ready for use in reports and dashboards.

The end result is a “dataflow” that can be used and reused as a source of data across multiple reports without being limited to specific Power BI desktop files. The dataflow data refresh can be scheduled in the Power BI service and is stored as Common Data Model entities in Azure Data Lake Storage Gen2, which is designed for big data analytics. The below image shows the design of the simplified proposed BI solution that I follow:

**Diagram

Description automatically generated**

*Design of proposed BI solution designed using Terrastruct online app*

1. Power BI Dataflow has connectors to connect to our Microsoft SQL Server database and PostgreSQL DB and fetch data from platform 1 and platform 2 accordingly. The CSV connector inside PBI Dataflows is used to import the country table. The Data is stored in Azure Data Lake Storage Gen2.
2. Prepare and transform the data using Power Query Online
3. A dataflow/s are created in the Power BI Service.
4. The Dataflow/Dataflows are consumed by Power BI Desktop.
5. Build the report in Power BI Desktop.

I follow the above steps to design my BI solution.

**Step 1: Creating Power BI Dataflows and connecting to Data sources**

* First, sign into Power BI at: [http://powerbi.com](http://powerbi.com/):
* create a new app workspace to create dataflows.
* I start creating a new Dataflow in the defined workspace by selecting “Define new tables” option.
* I connect the created dataflow to the given data sources through provided connectors and fetch the tables from platform1 and platform2. I also connect the Dataflow

**Graphical user interface, application, Teams

Description automatically generated**

*PBI Dataflow Datasource connectors page*

**Graphical user interface, application

Description automatically generated**

*Tables are added to Dataflow power query editor*

**Step 2:**

In my proposed design I have adopted an extension of a Star Schema which is called Snowflake Schema. A Snowflake Schema is an extension of a Star Schema, and it adds additional dimensions. The dimension tables are normalized which splits data into additional tables. In our design, we can use both Star and Snowflake schema. Snowflake schema due to normalizing attributes results in storage savings, however, it adds additional complexity in source query joins and might bring down data retrieval speed. The below image shows the design of our data model:

**Diagram

Description automatically generated**

*The Proposed Data model Generated by Microsoft Power BI desktop*

**Transform Data in Online power Query editor in PBI DF:**

1. Import the fact data tables from platform 1 and platform 2 to power bi Dataflow
2. Rename the columns to unify the column name of both tables
3. Remove the extra columns
4. Check the data type of all columns to be matched with the data values
5. Create Fact and Dimensions Tables from platform1 and Platform2 data :

* Dim\_BillingStatus from platform1
* Dim\_Business from platform1&2
* Dim\_Date: I use M code to create a dimensional date table ([DateGenerator.M-Language](https://github.com/mehdimkz/EzypayProject/blob/main/DateGenerator.M-Language" \o "DateGenerator.M-Language) in Github)
* Dim\_Country: I use a standard country table from <https://datahub.io/core/country-codes#data-cli> and bring it to Dataflow through connecting to a CSV file
* Fact\_BusinessCustomers: Append data from platform1&2 together to create a Fact table

The diagram view of our design is shown below (The steps are the transformation steps done to create the respected table). The detail of transformations for each table can be found in Transformations.M-Language in Github.

**Diagram

Description automatically generated**

*Diagram Generated by Microsoft Power BI Dataflow*

Table

Description automatically generated

*List of tables created for Ezy-Dataflow in power BI online service*

1. Add an incremental refresh for our fact table

Since Fact tables are always the largest tables in the data warehouse. I recommend that we reduce the number of rows transferred for this table. So I set incremental refresh for that entity. An incremental refresh can be done in the Power BI dataset and also in the PBI dataflow.

**Graphical user interface, text, application

Description automatically generated**

**Dataflows vs Datasets:**

Using Dataflows has some advantages over using Datasets and it’s mainly on Data refresh frequency. In dataflows, refreshes can be managed independently and efficiently. Different dataflows can be refreshed daily, weekly, and on-demand. This is especially helpful when some of your data sources are slow to refresh and the underlying data isn’t refreshed as often as for other data sources. In Datasets, Data is pulled from all the data sources when you refresh a dataset, even though some sources don’t need to be refreshed as frequently as other data sources. In case there’s a slow and infrequently-refreshed data source, the whole dataset pays the price.

**Step 3:**

The dataflow is ready in the Dataflow workspace. The below image shows the created dataflow ,”Ezy-Dataflow” in the PBI online workspace.

**Graphical user interface, application

Description automatically generated**

**Step 4 :**

Dataflow created in PBI Dataflow is consumed by Power BI Desktop as a Datasource. The below image shows that the created Dataflow with its tables can be seen in Microsoft Power BI desktop.

**Graphical user interface

Description automatically generated**

**Test Case:**

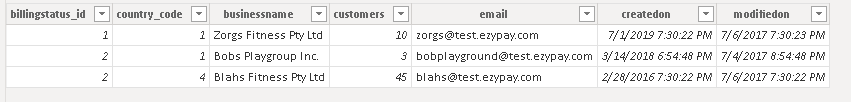
**Input :**

**Platform 1 Data:**

**Table

Description automatically generated**

**Platform 2 Data:**

****

**Output Report:**

**Graphical user interface, application

Description automatically generated**

**Suggestions for the proposed design improvement:**

It might be better to consider using different dataflows for different dimensions. The refreshes can be managed independently and efficiently. Different dataflows can be refreshed daily, weekly, and on-demand. This is especially helpful when some of the data sources are slow to refresh and the underlying data isn’t refreshed as often as for other data sources.