High Level Design

Crop Production Data Anlysis- India

Author: Nivetha N

Contents

Abstract

The Agriculture business domain, as a vital part of the overall supply chain, is expected to highly evolve in the upcoming years via the developments, which are taking place on the side of the Future Internet. This paper presents a novel Business-to-Business collaboration platform from the agri-food sector perspective, which aims to facilitate the collaboration of numerous stakeholders belonging to associated business domains, in an effective and flexible manner. This dataset provides a huge amount of information on crop production in India ranging from several years. Based on the Information the ultimate goal would be to predict crop production and find important insights highlighting key indicators and metrics that influence the crop production.

Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level. The HLD will:

- Present all of the design aspects and define them in detail
- Describe the user interface being implemented
- Describe the hardware and software interfaces
- Describe the performance requirements
- Include design features and the architecture of the project

- List and describe the non-functional attributes like:
    - Security
    - Reliability
    - Maintainability
    - Portability
    - Reusability
    - Application compatibility
    - Resource utilization
    - Serviceability

1.2 Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

2 . General Description

2.1 Product Perspective & Problem Statement

The Agriculture business domain, as a vital part of the overall supply chain, is expected to highly evolve in the upcoming years via the developments, which are taking place on the side of the Future Internet. This paper presents a novel Business-to-Business collaboration platform from the agri-food sector perspective, which aims to facilitate the collaboration of numerous stakeholders belonging to associated business domains, in an effective and flexible manner. Based on the Information the ultimate goal would be to predict crop production and find important insights highlighting key indicators and metrics that influence the crop production. Make views and dashboards using Business Intelligence Tool Power BI.

2.2 Tools used

Excel, SQL Server, Business Intelligence tool Power BI are used to build the whole framework.

MS SQL Server

Microsoft Excel

Power BI Desktop

- Microsoft excel for transform and clean the data
- Ms SQL Server for Analyze the data
- Power BI desktop for Visualize the Data and create Dashboards as well

3. Design Details

3.1 Functional Architecture

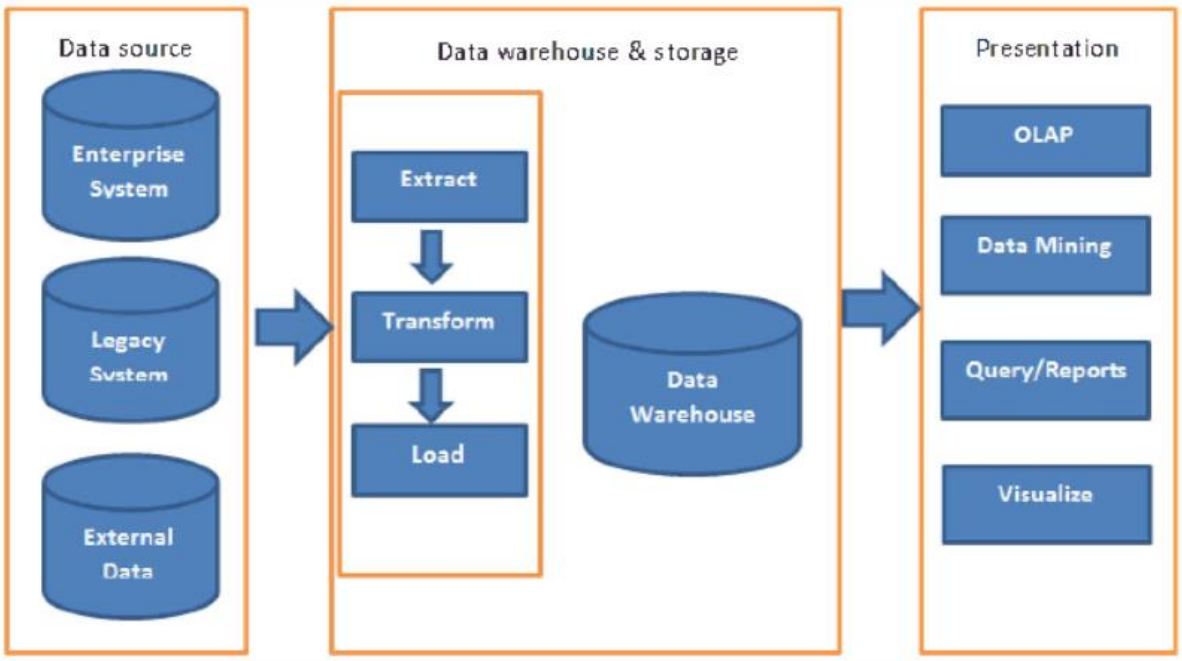
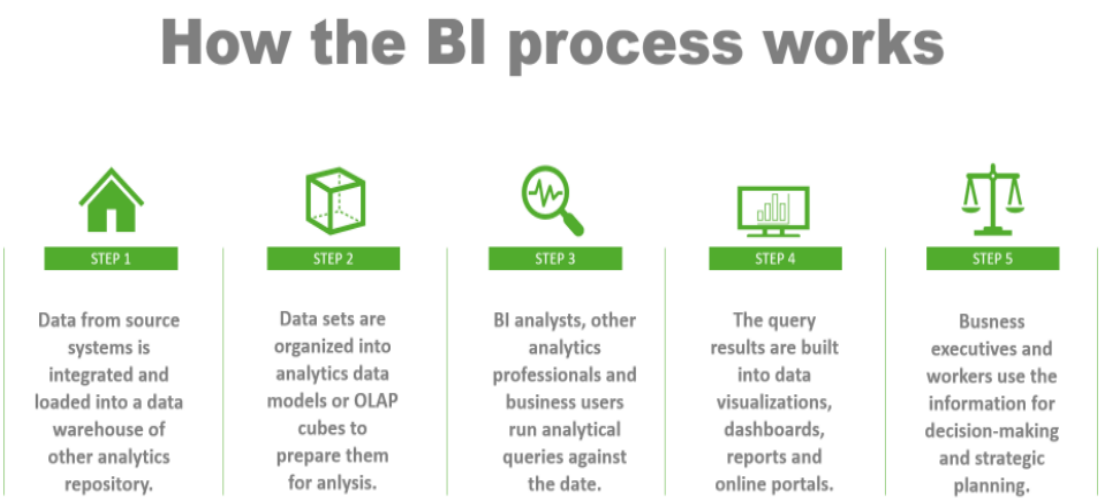


Fig: Functional Architecture of Business Intelligence



3.2 Optimizing Power BI Reports

Building performant reports is important in gaining trust with stakeholders and ensuring user adoption. Here are some practical and easy tips on how to optimize your Power BI reports! These tips will be broken down into two sections, **Data Model** and **Report View.**

Data Model

1. Use STAR Schema

Power BI is built to perform most efficiently when using a **STAR schema**. By breaking out your data model into fact and dimension tables, your queries will be optimized in your report. These fact and dimension tables should be related to one another, not joined (aka flat file format), which will result in more performant reports.

2. Limit Data Model

The fastest way to optimize your Power BI report is to limit the number of columns to only the ones you need in your data model. Go through your tables in Power Query and determine what fields are being used. Delete these columns if they are not being used in any of your reports or calculations. Another way to limit your data model is to use **Row Level Security** when applicable.

3. Push Data Transformations Upstream

Building on the previous step, if you are able to move your data manipulations to your query, this will help your performance. Whether you're doing calculations, formatting, or grouping at the appropriate granularity in your SQL, this will eliminate Power BI spinning its wheels on these items. If you end up doing your transformations in Power Query, try Query Folding.

4. Move Row Level Logic to Power Query

Move any logic that needs to be calculated at the row level to Power Query. Instead of using DAX to perform the more basic IF THEN calculations, use M to do the same thing. These fields will be included in your permanent data model brought in from Power Query instead of them being performed in the Power BI side.

5. Remove Calculated Columns

Calculated columns are a huge drain on performance. Every time you interact with your report, calculated columns are recalculated. Measures, on the other hand, are only recalculated when that specific measure are interacted with on the report. If possible move your calculated column to Power Query. If you can't do that, make it a Measure. If you can't do that, ask yourself why you need this calculated column.

6. Convert Multiple Measures to Variables

Instead of creating metrics that require using multiple measures, use variables. You can perform multiple calculations within a single measure using the VAR and RETURN functions. This will minimize the number of measures you use as well as increase performance for the metrics being calculated. If you are going to be repeating a variable multiple times then you can still create an independent measure for that value. See example of variables below.

7. Amend Dates and Column Types

Dates will show as mm/dd/yyyy 12:00:00am by default in Power BI. To cut down on characters in the data model, change the date types to Date to eliminate the Time portion. If you have integers with a large amount of decimal points, reduce the decimal places showing. Lastly, if you have text showing TRUE or FALSE, you can change these to be binary, so 1 and 0.

Report View

1. Minimize Visuals

When creating your report, try and use as few visual elements in your report as possible. This will decrease the amount of calculations that Power BI is performing when rendering your report.

2. Don’t Use Slicers (if possible)

Use slicers only where needed. Otherwise, use the filters available to users in the Filter Pane. Slicers are less efficient at returning required data than the Filter Pane. If you do use a slicer, make it a single drop down instead of a list. A list will pre-populate some queries in it and make it less efficient.

3. Use Performance Analyzer to Analyze

Power BI has a built in way for you to analyze the performance of your reports. In the View ribbon you can find the Performance Analyzer. By opening this pane and clicking Record, this will show you how fast your report renders when performing different functions in your report. You can also see what specific sections of your report are causing performance delays so that you can work on improving those sections.

4. KPIs

4.1 (Key Performance Indicators)

1. Area distribution impact of crop production
2. Area distribution on different crops across states
3. Top 3 crops impact on production
4. Top 3 crops impact on production in each year
5. Highest States Contribution by production
6. Seasons impact on crop production

5. Deployment

Power BI is both a self service BI and an enterprise BI tool. In a lot of cases, people publish reports from Power BI Desktop. This can be subject to mistakes when working in a full DTAP environment. You would always have to manually change your connection string if you are going to publish to a production workspace. Automating your deployments makes sense. With Power BI Actions, you can automate the deployment to any environment and workspace. You can:

1. Upload Power BI dashboard (pbix file).
2. Create a Power BI workspace.
3. Remove a Power BI workspace.
4. Add a new Admin user to a Power BI workspace.
5. Update the connection of a Power BI report.
6. Refresh a dataset.

One thing to notice is that you'd probably have the Power BI reports in source control and that they will have connectionstrings to your dev databases and cubes. If you automate the deployments to other environements, you'd have to first publish the report, and then change the connectionstring with Power BI Actions. This means that:

1. you overwrite the report in the production workspace which has a connection to production databases, with the latest version in source control, which still points to dev databases;
2. You then change the connection string.

If something goes wrong between step 1 and 2, you have a report on production that still connects to the wrong environment. Ideally, you would change the connectionstring *before* you publish it to production. Now, there is not yet (as far as I know) a way to do this. I have tried manipulating the pbix file programmatically, but that is unsupported behaviour. It did work for a while, but

doesn't anymore, which is exactly the problem with unsupported methods
Automating deployments is a great idea if you are delivering Power BI reports and the underlying datasets to the business, a datawarehouse or a cube for example. Any changes to the dataset means you have to change the report itself too.

Depending on your organizational roles and responsibilities, Power BI Desktop should be installed by a systems administrator and the Administrator in coordination with the appropriate IT roles.