

# **Foresta Documentation |**

## **Decision Spot**

---

**None**

*Decision Spot*

© <a href="https://decisionspot.com" target="\_blank" rel="noopener">Copyright Decision Spot, LLC | All Rights Reserved</a>

## Table of contents

---

1. Home	3
1.1 Welcome	3
2. User Guide	4
2.1 User Guide	4
2.2 Organizing your analytics	5
2.3 App Operations	15
2.4 Reporting	53
2.5 Access Control	106
2.6 REST API	111
3. App Builder	119
3.1 App Builder	119
3.2 Getting Started	121
3.3 Create Applications	123
3.4 Reporting Automation	131
3.5 Deprecate and Delete Applications	134
4. Applications	141
4.1 Applications	141
4.2 Supply Chain Network Optimization	142
4.3 Transportation Optimization	262
4.4 Shipday Optimizer	307
4.5 SMC Rater	320
4.6 Geo Coder	327
4.7 Seasonality Detector	331
4.8 Inventory Simulator	337
5. Release Notes	350
5.1 Foresta Release Notes Summary	350

# 1. Home

---

## 1.1 Welcome

---

### Knowledge-base for Foresta Users and Developers

Foresta (developed by Decision Spot) is a highly scalable cloud-based application deployment platform. It is built by world class developers and data scientists using the latest technology in advanced analytics. Foresta offers pre-built data-science and optimization applications as well as capabilities to rapidly deploy production grade custom applications.

-  Exploring applications? Go to [User Guide](#)
-  Building applications? Go to [App Builder](#)

*Didn't find what you're looking for?*

Contact our support 

### What is an application (aka App)?

An app is a tool that helps you make decision. It provides the user with information and data relevant to a business process and embed frameworks that can help an organization make data-driven decisions.

### Who is a Foresta user?

Foresta addresses two kinds of users:

Business Users	App Builders
example: Division Manager, Leader, Planner, Analyst etc.	example: Data Scientist, Data Engineer, etc.
They develop and evaluate strategies, Determine business scenarios, Make corporate and operational decisions	They evaluate strategies through algorithms and statistics, develop data pipelines to generate business metrics, Establish complex data workflows
No programming capacity	Can code
Business users are the end consumers of data apps	Develop apps for business users

## 2. User Guide

---

### 2.1 User Guide

---



Foresta apps are designed to foster best-in-class analytics practice in an organization. Besides a seamless and secure IT integration, these apps are rich in features that enable business-users to extract maximum value from the data.

👉 Use the navigation to jump to the relevant section.

## 2.2 Organizing your analytics

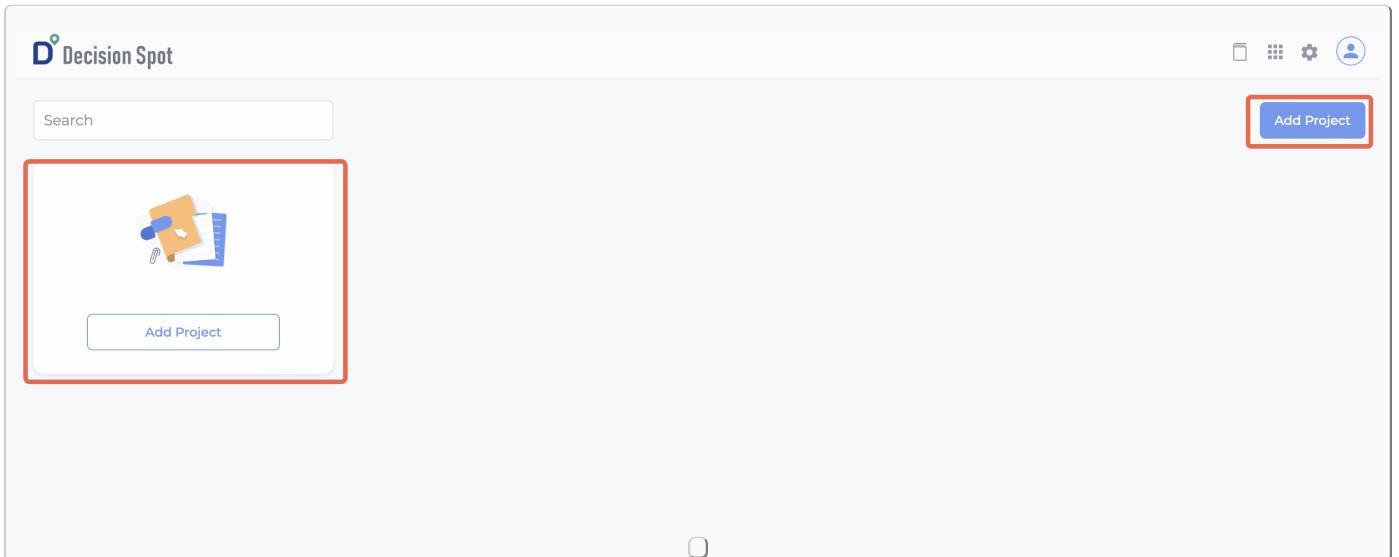
### 2.2.1 Projects

Foresta has a **Projects** functionality, which allows the users to organize, manage and access control their modeling efforts as per their need.

A **Project** in essence is a container for a group of applications, that are being used for a specific analytics effort.

#### Creating a new Project

On Foresta homepage, Projects can be added by either using the **Projects** card or the button on top right corner as seen in the screenshot below.



*Foresta Homepage*

On-clicking **Add Project**, a dialog box opens up with several fields and settings:

## Add Project

**1**

Name

**2**

Description

**3**

Choose background color (or card view):

**4**

Access Settings

Public

Private

Restricted

**5**

Applications

Viz Testing

TTS Diet

Opportunity Management

Supply Chain Network Optimizer

Costing and Pricing

**6**

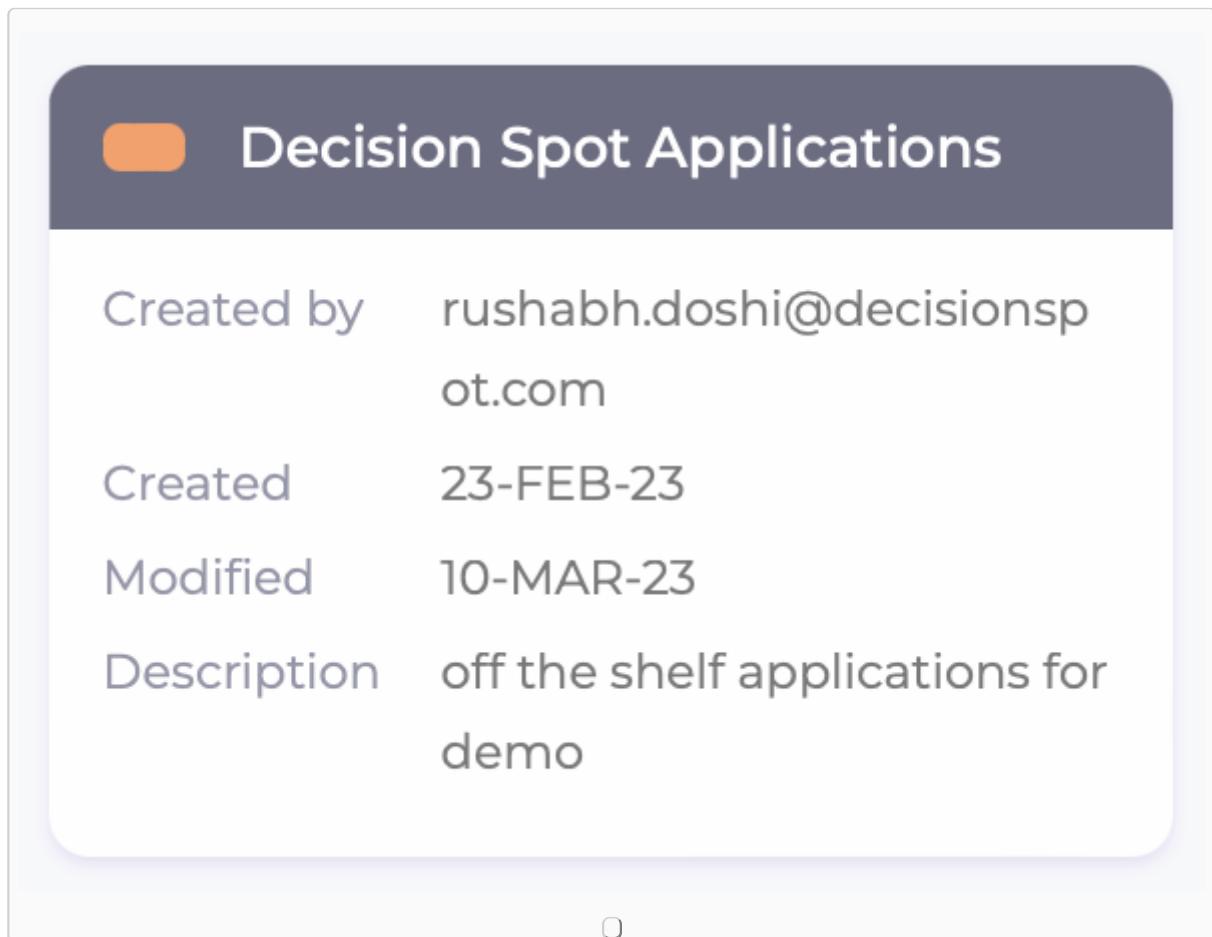
[Cancel](#)

[Add Project](#)

*Add Project Dialog Box*

1. **Name** for the project
2. **Description** for the project
3. **Background color** for the project card
4. **Access Settings** for the project
5. Add **Applications** to the project
6. **Add Project** button to confirm the creation of project

Once the project is created it appears as a card on the homepage, and can be accessed by clicking on the card. The project card also provides some metadata to the user.

*Project Card***Editing an existing Project**

A project can be edited (add / remove applications, change access settings etc.) by using the **Edit Project** option from the dropdown menu on the projects card as shown in the screenshot below.

**Decision Spot Applications**

Created by	rus ot.c	Edit
Created	23-	Delete
Modified	10-MAR-23	
Description	off the shelf applications for demo	

*Project Card Actions***Warning**

A user can also delete the project using the same dropdown menu. All the apps, scenarios and data associated will be deleted upon deletion of the project.

**Project customization**

<<<<< HEAD This feature is available to the 'App Builder' and 'Administrator' roles. In the project page, the users can customize the way apps are positioned and named. It can be very useful in many situations. In the example below, there are two versions of the same app Supply Chain Network Optimizer. Since the newest version is on the bottom and the oldest one at the top, hover over the newer version, a six dot icon will appear, Click the icon and Drag this version to an the top position. ===== At the Project page, you can continue customizing the way that your apps are positioned and named. That feature can be very useful in many situations. In the example below, there are two versions of the same app Supply Chain Network Optimizer. Since the newest version is on the bottom and the oldest one at the top, by hovering over the app a six dot icon will appear and it will allow you to drag to newest version to the top position.

3e47ce11310a2c92254bf376baebd98e631cf9b6

The screenshot shows the 'Project Page' interface. At the top, there's a navigation bar with 'Decision Spot', 'My Project > Supply Chain Network Optimizer', and various icons for search, filters, and project management. Below the navigation is a search bar labeled 'Search scenarios' and a dropdown for 'Apply labels'. To the right of these are buttons for 'Newest' and several small icons. The main content area lists scenarios: 'Supply Chain Network Opti...' (highlighted with a red box around its pencil icon), 'Transportation Optimization', 'Inventory Optimization', and another 'Supply Chain Network Opti...'. There's also a small red box around the three-dot menu icon next to the second scenario.

*Project Page*

<<<<< HEAD Now that the newest version is on the top of the apps list, users can rename it to differentiate it from the older version. Hover over the app name and click on the pencil icon to rename the app.

===== Now that the newest version is on the top of the apps list, you can rename it to differentiate it from the oldest version. Hover over the app name and click on the pencil icon.

3e47ce11310a2c92254bf376baebd98e631cf9b6

This screenshot shows the 'Project Page' after renaming. The scenario 'Supply Chain Network Optimizer' has been changed to 'SCNO 0.0.24'. A red box highlights the pencil icon next to the original name. A tooltip box shows the new name and some metadata: 'In project name: Supply Chain Network Optimizer', 'Global name: Supply Chain Network Optimizer', 'Version: 0.0.24', 'In project ID: 355', and 'Global ID: 104'. The rest of the interface is similar to the first screenshot.

*Project Page*

<<<<< HEAD At shown in the example, it is renamed with the acronym and the release version. Any app can be reordered or renamed, not necessarily upon deployment. ===== At this case, we renamed it with the acronym and the release version of the app. Reorder and rename it by anytime.

3e47ce11310a2c92254bf376baebd98e631cf9b6

This screenshot shows the 'Project Page' after the rename and a subsequent change. The scenario 'SCNO 0.0.24' has been renamed back to 'Supply Chain Network Optimizer'. A red box highlights the pencil icon next to the original name. The rest of the interface is consistent with the previous screenshots.

*Project Page*

<<<<< HEAD Renaming the app on the project will only change its name for that project. The Global App name will continue as it is on the App Builder page.  
===== Renaming the app on the project will only change its name for that project. The Global App name will continue as it is on the Apps page.

3e47ce11310a2c92254bf376baebd98e631cf9b6

## 2.2.2 Scenarios

Foresta provides the user with a scenario framework to enable what-if analysis. It also has a powerful functionality to create / manage scenarios.

### Ways to Create a Scenario

Foresta has 5 different ways to create a scenario.

1. Add Scenario Button

2. Duplicate an existing scenario

3. Add Scenario menu

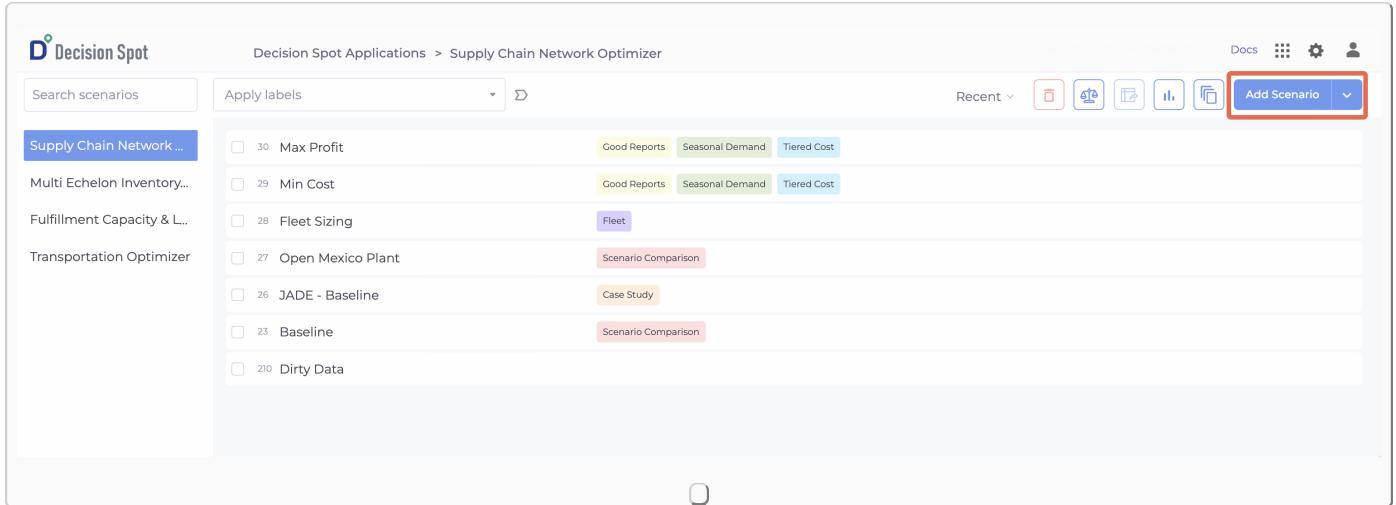
4. Using REST-API

5. Bulk Data Operations

Each method has been described below.

#### 1. ADD SCENARIO BUTTON

User can create a **New** scenario by clicking on the add scenario button on the Application Home Screen.



*ADD SCENARIO button*

On clicking the **ADD SCENARIO** button, a pop-up appears where a user needs to provide a name for the scenario, and optional description for the scenario.

### Scenario

**Name**

**Description**

Public Read

Public Write

Cancel
Add

*Scenario Dialog Box*

After providing the name and description, when the user clicks on **Add**, it creates a new scenario with a unique **Scenario ID** attached to it. This scenario contains an empty copy of all the tables that are configured in the app schema, which the user can then populate using data using a host of tools (xlsx / csv upload, using RESTful API, or app-to-app piping etc.).

The **Scenario** card on the UI looks like:

The screenshot shows the Decision Spot interface with the following elements:

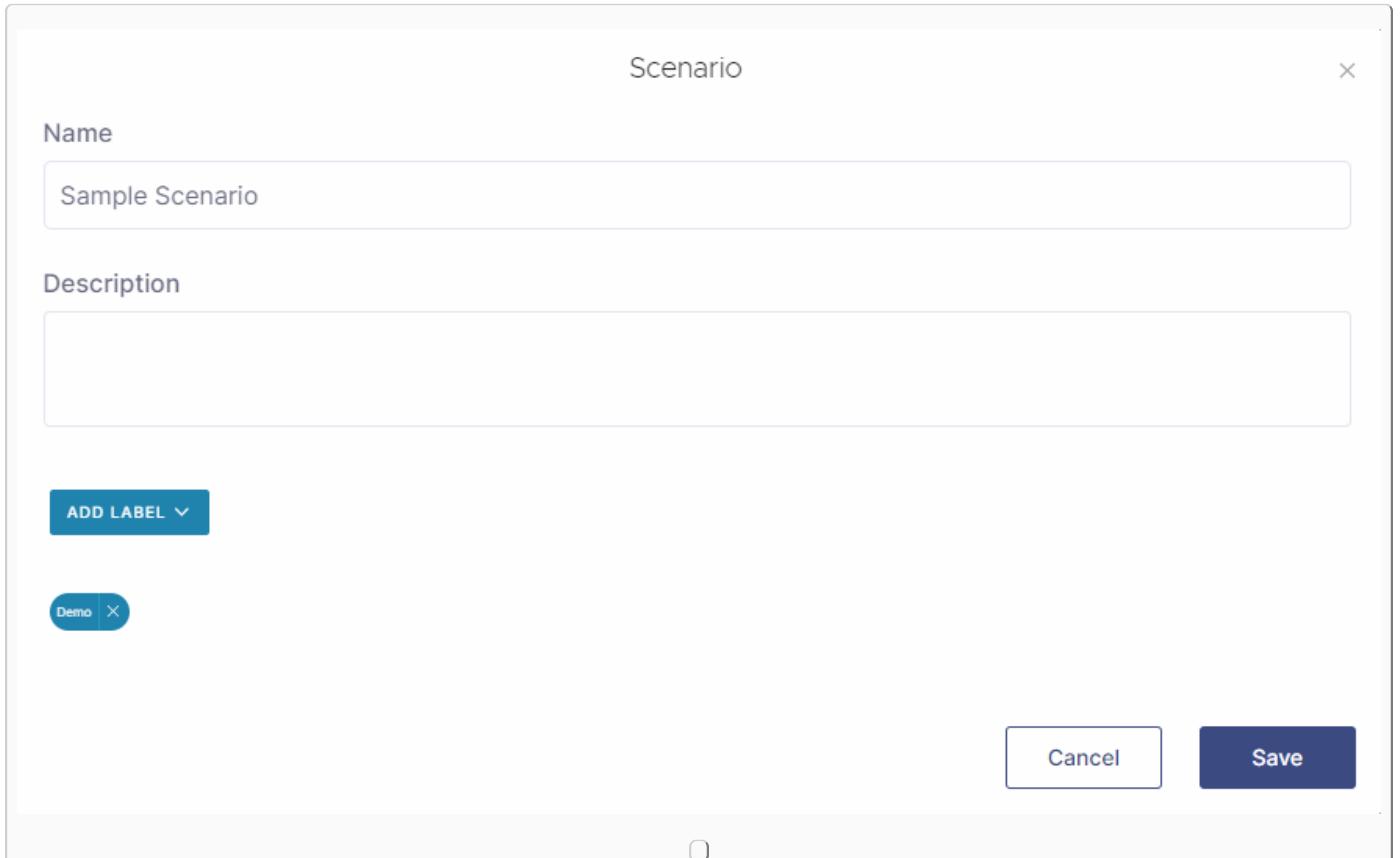
- Search scenarios**: A search bar with a placeholder "Search scenarios".
- Apply labels**: A dropdown menu with a placeholder "Apply labels".
- Recent**: A list of recent scenarios with small preview cards.
- Action buttons**: A row of buttons for managing scenarios: **Edit**, **Compare With**, and **Duplicate**.
- Scenario Card**: A detailed view of the "Grocery Store - Cereals" scenario, showing its name, ID (219), labels ("Cereals", "Chicago"), and description.

*Scenario Card*

It provides the user with some information, as well as a few actions.

1. Item 1: Scenario ID
2. Item 2: Scenario Name
3. Item 3: Scenario Tooltip, contains Name and Description
4. Item 4: Labels
5. Item 5: Edit Scenario
6. Item 6: Compare Scenario (Input Data)
7. Item 7: Duplicate Scenario

The user can assign **Labels** to a scenario by using the **EDIT** functionality.



*Edit Scenario Dialog Box*

## 2. DUPLICATE SCENARIO BUTTON ON SCENARIO CARD

The **Duplicate** button on Scenario Card, creates a copy of an existing scenario.

You can duplicate a scenario in different ways though, by selecting the arrow pointing down just at the right of Duplicate.

***DUPPLICATE Scenario*****3. ADD SCENARIO MENU**

Add Scenario dropdown is a powerful capability to create scenarios in a variety of ways. Unlike the "Add Scenario" button, a user can create a new scenario, and populate it using an existing scenario or pull the inputs from a different app.

Click the arrow next to the **ADD SCENARIO** button, it opens a dropdown selection menu to create a new scenario using several options as configured by the app builder.

***Add-Scenario Menu***

The actions available in the App-to-App menu will depend on what the creator of the app has configured, and it can vary for each application.

**4. USING REST API**

Please refer to the API documentation for details.

**5. BULK DATA OPERATIONS**

Bulk copy is a smart way to create scenarios by combining data from several different sources. [More details are discussed here](#)

## 2.3 App Operations

### 2.3.1 Navigating an application

The user clicks on a scenario of an app to view the app window that enables them to view, upload and manipulate data, run models, and see the scenario specific information. This is where users spend the most time in Foresta.

Name	Active	Latitude	Longitude	Sourcing Status	Open Close Status	Inventory Status
C_Springfield	True	37.21527800000001	-93.298056	Multiple	Fixed	Flow Through Only
C_Phoenix-Mesa	True	33.448333	-112.073333	Multiple	Fixed	Flow Through Only
C_Houston	True	29.763056	-95.363056	Multiple	Fixed	Flow Through Only
C_Brazoria	True	29.044167	-95.568889	Multiple	Fixed	Flow Through Only
C_Knoxville	True	35.060556	-83.920833	Multiple	Fixed	Flow Through Only
C_San Diego	True	32.71527800000005	-117.156389	Multiple	Fixed	Flow Through Only
C_Seattle	True	47.606389	-122.330833	Multiple	Fixed	Flow Through Only
C_Orange County	True	33.472791	-117.694517	Multiple	Fixed	Flow Through Only
C_Lexington	True	38.049167	-84.50027800000002	Multiple	Fixed	Flow Through Only
C_Milwaukee	True	43.03888900000001	-87.906389	Multiple	Fixed	Flow Through Only
C_San Antonio	True	29.42388900000006	-98.49333299999999	Multiple	Fixed	Flow Through Only
C_Brownsville	True	25.901389	-97.497222	Multiple	Fixed	Flow Through Only
C_Dallas	True	32.783333	-96.8	Multiple	Fixed	Flow Through Only
C_Omaha	True	41.258611	-95.9375	Multiple	Fixed	Flow Through Only
C_Los Angeles	True	33.472791	-117.694517	Multiple	Fixed	Flow Through Only
C_Columbus	True	39.961111	-82.998889	Multiple	Fixed	Flow Through Only
C_Santa Rosa	True	38.440556	-122.713333	Multiple	Fixed	Flow Through Only
C_Reno	True	39.529722	-119.812778	Multiple	Fixed	Flow Through Only

#### App Navigation

1. **Solve:** Click this to execute the model from the GUI.
2. **Data Validation:** Click this to run data validation over the input data. This is a powerful functionality that highlights the integrity issues in the data such as invalid values, duplicates, nulls etc. The data integrity errors are populated in the validation tables.
3. **Bulk Upload:** This is used to upload data in the input tables of the app.
4. **Input Tables:** This is the landing page of the scenario that views the input tables of the app.
5. **Validation Tables:** The data integrity issues found by Solve or Data Validation are populated in a group of tables here.
6. **Solution Tables:** The solution tables and reports from the model output are viewed in Solution Tables.
7. **Visualization:** The integrated visualizations are viewed by going into the visualization tab.
8. **Logs:** While the model is solving, the users can access the logging to view the real-time status of the execution.

### 2.3.2 Data Grid

The data grid on Foresta enables the user to view the input data / solution data on the UI. It also has a host of features that enables the user to perform a variety of operations.

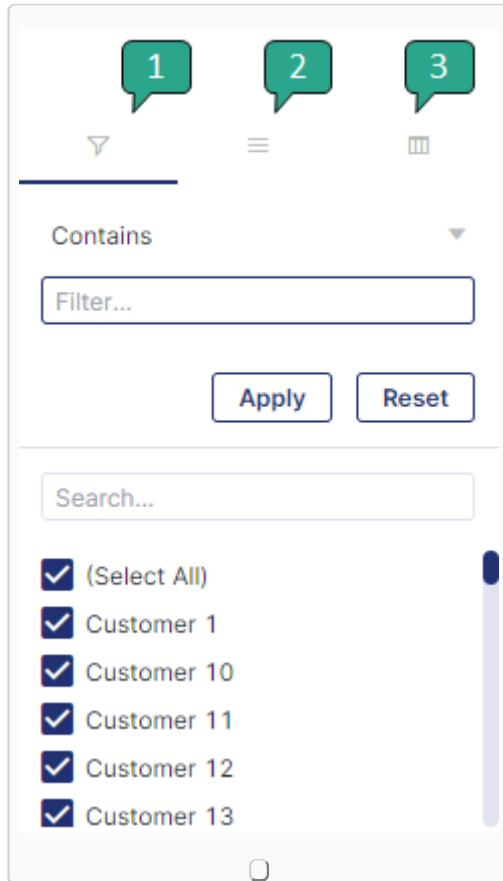
On the screenshot below:

Name	Active	Latitude	Longitude	Sourcing Status	Open Close Status	Inventory Status	Opening Cost	Operating Cost	Closing Cost
Plant A	True	41.878114	-87.629798	Multiple	Fixed	Flow Through Only	0	0	0
Plant B	True	32.688333	-96.938424	Multiple	Fixed	Flow Through Only	0	0	0
Plant C	True	39.736603	-105.02123	Multiple	Fixed	Flow Through Only	0	0	0
Plant D	True	33.748752	-84.887684	Multiple	Fixed	Flow Through Only	0	70000	0
Customer 1	True	39.438881	-75.22817499999998	Multiple	Fixed	Flow Through Only	0	0	0
Customer 2	True	30.355851	-81.694137	Multiple	Fixed	Flow Through Only	0	0	0
Customer 3	True	38.350903	-78.955919	Multiple	Fixed	Flow Through Only	0	0	0
Customer 4	True	33.979568	-117.898437	Multiple	Fixed	Flow Through Only	0	0	0
Customer 5	True	32.688333	-96.938424	Multiple	Fixed	Flow Through Only	0	0	0
Customer 6	True	32.722794	-97.084692	Multiple	Fixed	Flow Through Only	0	0	0
Customer 7	True	42.831055	-78.639808	Multiple	Fixed	Flow Through Only	0	0	0
Customer 8	True	43.580123	-116.219041	Multiple	Fixed	Flow Through Only	0	0	0
Customer 9	True	37.818495	-121.286548	Multiple	Fixed	Flow Through Only	0	0	0
Customer 10	True	44.872086	-92.997031	Multiple	Fixed	Flow Through Only	0	0	0
Customer 11	True	44.814135	-68.778574	Multiple	Fixed	Flow Through Only	0	0	0
Customer 12	True	39.529321	-119.804953	Multiple	Fixed	Flow Through Only	0	0	0
Customer 13	True	42.287672	-74.561318	Multiple	Fixed	Flow Through Only	0	0	0
Customer 14	True	39.736603	-105.02123	Multiple	Fixed	Flow Through Only	0	0	0
Customer 15	True	45.5414	-122.498239	Multiple	Fixed	Flow Through Only	0	0	0
Customer 16	True	33.128095	-95.598595	Multiple	Fixed	Flow Through Only	0	0	0
Customer 17	True	33.128095	-95.598595	Multiple	Fixed	Flow Through Only	0	0	0
Customer 18	True	33.605722	-101.873513	Multiple	Fixed	Flow Through Only	0	0	0
Customer 19	True	41.061041	-86.208192	Multiple	Fixed	Flow Through Only	0	0	0

Foresta Data Grid

1. Adding a row to the grid
2. Deleting a row from the grid
3. Uploading xlsx / csv file to the grid
4. Downloading xlsx / csv file from the grid
5. Updating the data in the grid using **Google Sheets**
6. Setting page size (number of rows per page)
7. Row count indicator of the grid
8. Page navigation, to go to the next/previous page or to the last/first page

As shown below, users can also apply some other commands in the UI by clicking the column headers:



*Filters, Pin and Visible columns*

1. Apply Data Filters
2. Pin columns in the GUI
3. Select visible columns in the GUI

### 2.3.3 Scenario Comparison

The **COMPARE WITH** feature allows the user to compare the scenario input data sets from the same app. This can be useful in identifying changes across scenarios.

The screenshot shows the Decision Spot Applications interface for the Supply Chain Network Optimizer. At the top, there's a navigation bar with 'Decision Spot Applications > Supply Chain Network Optimizer'. Below it is a search bar labeled 'Search scenarios' and a dropdown for 'Apply labels'. On the right side of the header are 'Recent' dropdown, several small icons (red square, blue gear, etc.), and a 'Docs' link. The main area displays a list of scenarios:

- 30 Max Profit (tags: Good Reports, Seasonal Demand, Tiered Cost)
- 29 Min Cost (tags: Good Reports, Seasonal Demand, Tiered Cost)
- 28 Fleet Sizing (tag: Fleet) - this row has a red box around the 'Edit' and 'Compare With' buttons
- Fleet Sizing i Mexico Plant (tag: Scenario Comparison)
- 26 JADE - Baseline (tag: Case Study)
- 23 Baseline (tag: Scenario Comparison)
- 210 Dirty Data

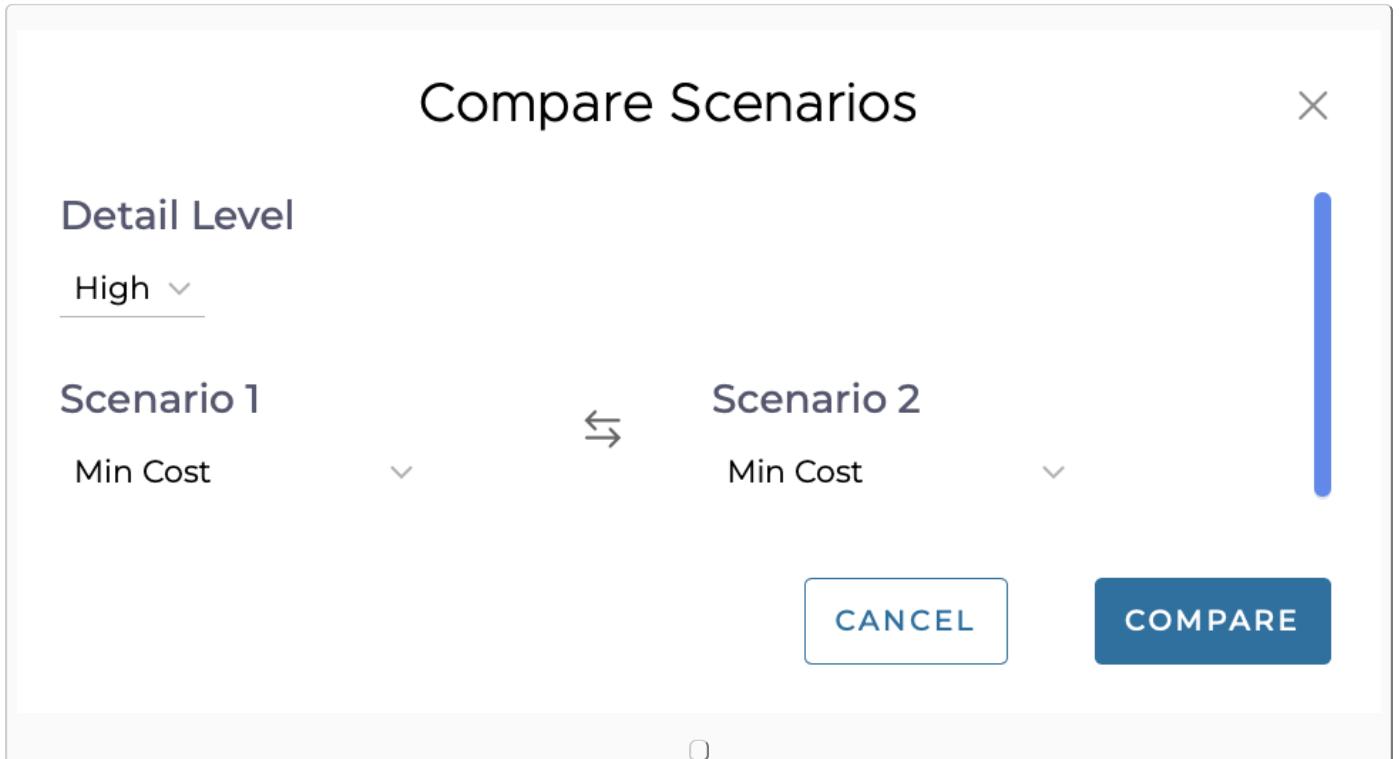
At the bottom right of the scenario list, there is a red box highlighting the 'Compare With' button.

*Compare Scenarios Button*

The comparison can happen at two different levels of detail:

1. Low
2. High

To compare inputs of scenarios, click on the **COMPARE WITH** button as shown in screenshot above and then select the **Detail level**, **Scenario 1** and **Scenario 2**.



*Compare Scenarios Dialog box*

**The differences identified for low level of details are:**

1. Row Difference Summary - This table indicates the number of extra rows in either of the two compared scenarios.

Found 7 differences between scenarios		
Table	Open Mexico Plant Extra Rows	Baseline Extra Rows
Carrier Specific Failure Logging	1	0

2. Value Difference Summary - The value difference summary table provides the user with information about number of different values at a Table -> Field level between the compared scenarios.

Found 7 differences between scenarios X

**Table**  
Value Difference Summary ▼

Table	Field	Number of Differences
Demand	Landed Cost X-Ray	2123
Sites	Longitude	37
Sites	Active	2
Sites	Latitude	5
Sites	Opening Cost	1
Sites	Open Close Status	2

**The differences identified for High level of details are:**

Along with the **Row Difference Summary** and **Value Difference Summary**, the **High** detail comparison provides the exhaustive list of all the value differences for each table (i.e. Product value difference details, Parameters value difference details etc.).

Found 2 differences between scenarios X

**Table**  
Products Value Difference Details ▼

Name	Data Field Name	Max Profit Value	Min Cost Value
Fibre Grade 1	Weight	1	10

*Detail Comparison*

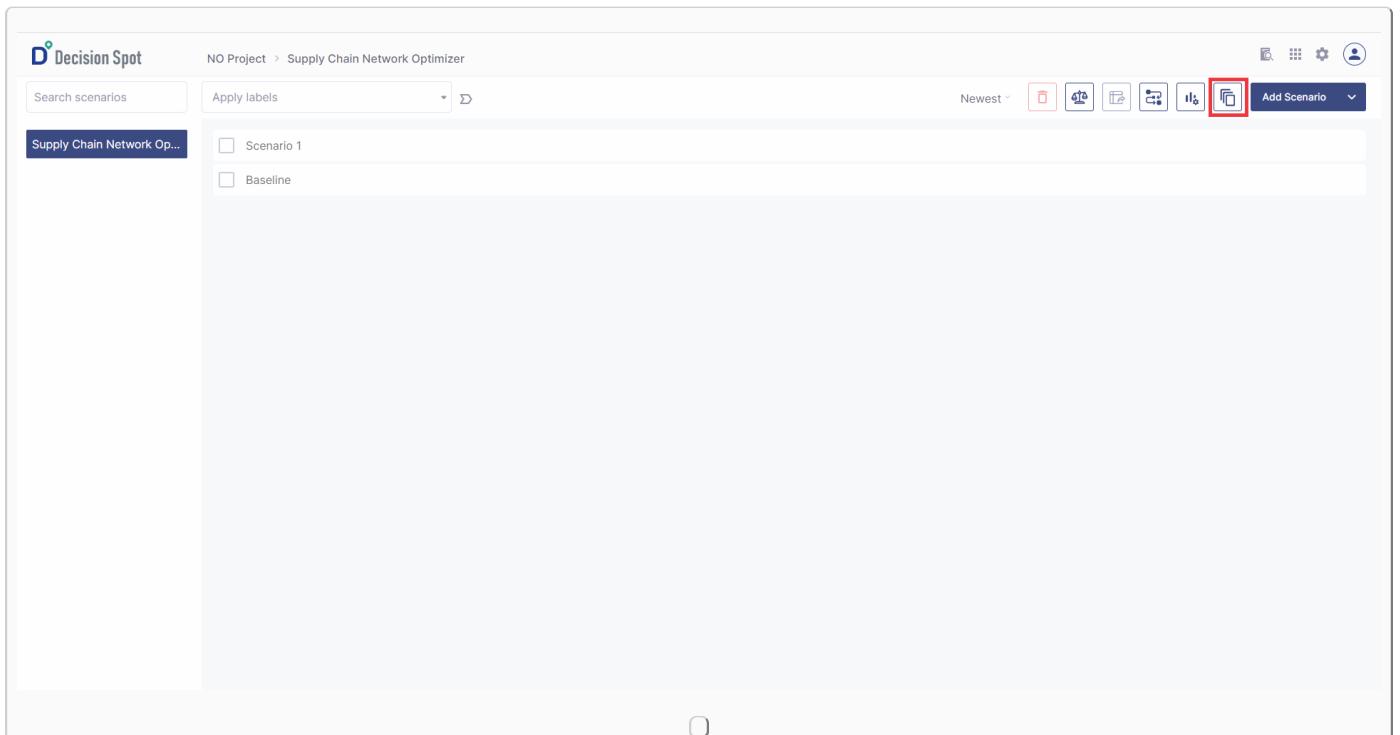


The **High** level of detail comparison can take a long time depending on the size of the input data in scenarios.

### 2.3.4 Bulk Data Operations

Foresta provides users with **Scenarios** functionalities to build various scenarios for their models out-of-the-box. There are several ways to create a scenario on Foresta application (Uploading from Excel or CSV files, Importing from other applications, Duplicating existing scenarios, etc). But if in case, a user desires to build a scenario blending data from multiple scenarios, or maybe you want to copy the data from a scenario and change part of the data at the same time, The **Bulk Data Operations (BDO)** feature is perfect to achieve that. It makes these tedious tasks of moving bulk quantities of data - easy, efficient, fast and reproducible.

Users can apply filters and transformations above the data locally within an app without having to manually do any edits. To use the BDO, on the App page select the **Bulk Data Operations** option in the top right corner of the screen.



*Project Window*

#### Filters, Transformations and Configurations

Before starting to use BDO, it is important to understand what are **Filters** and **Transformations** and **Configurations**

- **Filter** is a user-defined condition that slices input data based on entries in a column.
- **Transformation** is a user-defined operation that modifies the data
- **Configuration** is a user-defined sequence of filters and transformations applied on data.

Refer to the following examples:

##### Example 1:

Imagine if there were a data table called `Sites` which contains the location of different manufacturing plants (based in Chicago, Dallas etc.) and their corresponding operating costs.

Sites		
Plant	Operating Cost	
Chicago	20,000,000	
Dallas	50,000,000	
...		

Now, Let's say one is to change the operating cost of chicago plant to 100 million.

A **Configuration** is a setting where the table `Sites` is selected, the `Plant='Chicago'` is **Filtered** and then the `Operating Cost` is **Transformed** to a value = 100,00000.

#### Example 2:

A Planner in a Glass Manufacturing company is preparing the production plan for the next 3 years of production (2024, 2025, 2026). After running the Baseline scenario, the analyst is ready to deliver the outputs of their Foresta Supply Chain Network Optimization model to the stakeholders but, in the last moment, planner is requested to run a new scenario:

*"What is the impact on the production if Chicago Plant will operate only one shift for the first and second quarter of 2024"*

Now, one (not-smart) way to do this analysis is:

1. Create a new scenario in Foresta
2. Download the data from another existing scenario
3. Determine appropriate updates in the data
4. Manually do the updates
5. Uploaded the inputs back into the new scenario, re-run the model.

Instead of doing that whole process, the user wisely decides to use the Bulk Data Operations (BDO) feature to be faster and effective!

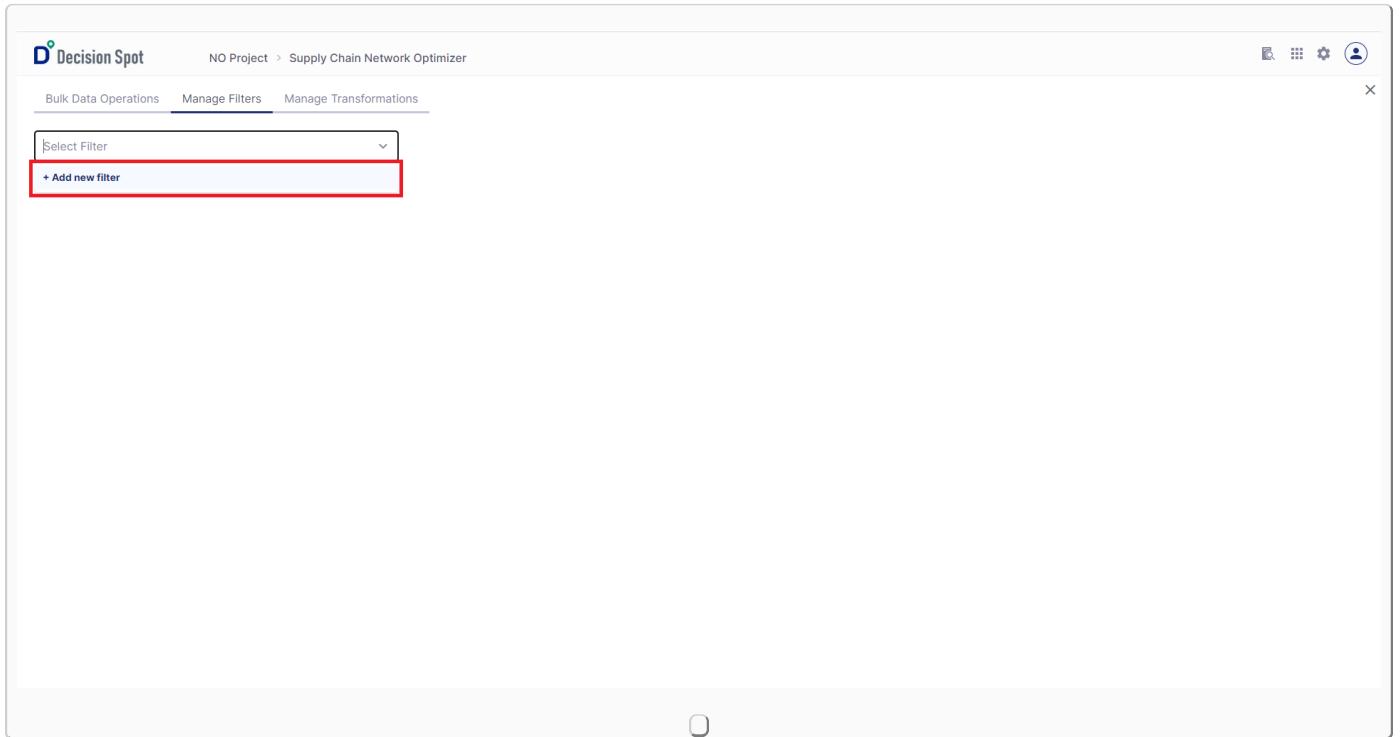
Let's call new scenario created in that process as `Scenario 1`.

#### Creating a Filter

To create filters, go to the tab **Manage Filters**, on the BDO page.

Bulk Data Operations Page

In the dropdown **Select Filter**, click + Add new filter.



*Manage Filters page*

Here, users can create filters based combining the different logical operators. Follow the steps described below to create a Filter:

1. Give a name to the Filter
2. Select the target input table
3. Select the Column to apply the filter
4. Select the Operator type
5. Select the value related to the operator

Recall stakeholders' request from [Example 2](#). The analyst needs to reduce production hours (shifts), which is done in the [Aggregate Production](#) table. The change is to be made to the `Site - Chicago` and in the `Time Period - Q1-2024 and Q2-2024`.

Create a filter on the table `Aggregate Production` and subset this data based on two columns `Site` and `Time Period`. First, Select the column `Site` where entry in this column is equal to `Chicago`

Manage Filters page

Then filter by `Time Period`, Notice the **Nested** (1) and **Sequential** (2) filters combined to filter `Site=Chicago Plant AND` (3) the `Time Period=Q1-2024 OR` (4) `Time Period=Q2-2024`.

Manage Filters page

Click the **Save** button to create the filter.

The screenshot shows the 'Manage Filters' page in the Decision Spot interface. A filter named 'Chicago plant for Q1-2023 and Q2-2024' is selected, joined to the 'Aggregate Production' table. The 'Join' button is highlighted with a red box. The 'Save' button at the bottom right is also highlighted with a red box.

Manage Filters page

### Note

Users can create **Join Filters** too. This is helpful in situations where filter is created based on information in a secondary table.

**For example:** Consider the case above, instead of filtering just the `Plant Chicago` if some user were to filter all the plants in a particular `State`. Given this information is available in a secondary `Sites` table instead of `Aggregate Production`, the filter is created, by clicking on **Join** option. It is available at the right of Table drop-down as shown below.

The screenshot shows the 'Manage Filters' page in the Decision Spot interface. A join filter is being configured, with the 'Join' button highlighted with a red box. The 'Save' button at the bottom right is also highlighted with a red box.

## Creating a Transformation

To create transformations, go to the **Manage Transformations** tab on the BDO page.

The screenshot shows the BDO interface with the 'Manage Transformations' tab selected. The 'Source scenario' section has a dropdown labeled 'Select Scenario'. The 'Destination scenario(s)' section includes a 'Filter by labels' dropdown with options for 'Baseline' and 'Scenario 1'. The 'Select type' section has a toggle switch set to 'Applied for all tables' and three radio button options: 'Full Replacement' (selected), 'Update Without Append', and 'Update and Append'. At the bottom right are 'Save & Preview' and 'Preview' buttons.

*Bulk Data Operations page*

In the dropdown **Select Transformation**, click **+ Add new transformation**.

The screenshot shows the 'Manage Transformations' page with a dropdown menu labeled 'Select Transformation' and a red box highlighting the '+ Add new transformation' button below it. The page header indicates 'NO Project'.

*Manage Transformations page*

Here, users can create transformations by defining a modification to the data using the different logical operators. Follow the steps described below to create the Transformation:

1. Give a name to the Transformation
2. Select the target input table
3. Select the Column where the data will be transformed
4. Select the transformation operator
5. Determine the value related to the transformation
6. Click the 'Save' button

The screenshot shows the 'Decision Spot' application interface. At the top, it says 'NO Project > Supply Chain Network Optimizer'. Below that are tabs for 'Bulk Data Operations', 'Manage Filters', and 'Manage Transformations', with 'Manage Transformations' being the active tab. A sub-header '+ Add new transformation' is visible. The main area contains a form for creating a transformation. The transformation name is 'Dividing shifts by 2'. The table is set to 'Aggregate Production'. The column is 'Max Hours'. The operator is '/'. The value is '2'. There is a '+' button to add more steps. At the bottom right are 'Cancel' and 'Save' buttons, with 'Save' highlighted.

*Manage Transformations page*

Referring to the case presented in [Example 2](#), the `Max Hours` was calculated considering 2 shifts per day. To reduce the the production hours to 1 shift per day divide the existing value of `Max Hours` by 2.

### Setting up a Configuration

Once [filters](#) and [transformations](#) are created, users proceed to setting up the **Configuration**.

Go to the **Bulk Data Operations** tab.

*Bulk Data Operations page*

First, select the **Source Scenario** from the **Select Scenario** dropdown. This is the scenario on which the Filters and Transformations will be applied.

*Bulk Data Operations page*

Select the input tables that will be copied.

### Note

1. Utilize the **Hide empty tables** (1) slider to show only input tables that are populated with data in the Source Scenario
2. Click **Select All** (2) to select all tables from the Source Scenario.

In the example shown below, all tables that are populated with data have been selected.

The screenshot shows the 'Bulk Data Operations' page in the Decision Spot interface. The 'Source scenario' section is highlighted with a red box. Inside, the 'Select all tables' checkbox is checked, and a green box labeled '2' points to it. Next to it is a 'Hide empty tables' toggle switch. The 'Operators' section is highlighted with a green box labeled '1'. The 'Destination scenario(s)' section shows 'Baseline' and 'Scenario 1' with checkboxes. A green box labeled '6' points to the 'Select type' dropdown menu, which contains 'Applied for all tables', 'Full Replacement' (selected), 'Update Without Append', and 'Update and Append'. At the bottom right are 'Save & Preview' buttons.

*Bulk Data Operations page*

A **Configuration** is a composed of a set of **Action's** which is defined as combination of Filter and Transformation applied onto a table.

```
* Apply filters and transformations On 1 Table:
  Apply Filter/s
  + ... (several filters)
  + Apply Transformation/s
  + ... (several transformations)
  -----
  = 1 Action

* Combine different actions on a set of tables:
  Action
  + Action
  + .... (several actions)
  -----
  = Configuration
```

To define an **Action**, at the target input table, select **+ Add Action**

The screenshot shows the Bulk Data Operations interface. On the left, the 'Source scenario' section lists tables under 'Baseline': Core Data (Sites 54, Products 5, Time Periods 12, Parameters 26), Demand and Production (Demand 3000, Production 60, Aggregate Production 48, Tiered Costs 480), and Transportation (Lanes 1, Truck Carrier 1). To the right, the 'Destination scenario(s)' section shows 'Selected tables (48)' and 'Operators'. Below these are 'Filter by labels' and 'Search' fields, and checkboxes for 'Baseline' and 'Scenario 1'. A red box highlights the 'Add Action' dropdown menu for the 'Aggregate Production' table. At the bottom right are 'Save & Preview' and 'Preview' buttons.

*Bulk Data Operations page*

An Action menu will be open below the selected table. From the drop down select the **Filter** and the **Transformation** to be applied.

This screenshot is similar to the previous one but focuses on the 'Aggregate Production' table. A red box highlights the 'Action 1' dropdown menu, which contains 'Add Filter' and 'Add Transformation' options. The rest of the interface is identical to the first screenshot, showing the table selection, destination scenarios, and operation settings.

*Bulk Data Operations page*

**Note**

Users can also create Filters and Transformations within the Configuration itself. Notice the options **+ Add filter** and **+ Add transformation** are available when setting up an Action.

**Warning**

User-defined Filters and Transformations can be reused or re-ordered in different ways to create a variety of actions. However, the Filters and Transformations created within the Action, are not re-usable across different Configurations.

Once all the actions are defined, select the **Type** of replacement to be done in the target scenario from section in the bottom right.

The screenshot shows the 'Decision Spot' application interface for 'Supply Chain Network Optimizer'. The top navigation bar includes 'My Project > Supply Chain Network Optimizer', 'Bulk Data Operations' (selected), 'Manage Filters', and 'Manage Transformations'. Below the navigation is a search bar 'Choose configuration' and a 'Save as New' button. The main area is divided into 'Source scenario' and 'Destination scenario(s)' sections. The 'Source scenario' contains a tree view of selected tables: 'Baseline' (Parameters 26, Demand and Production (Demand 3000, Production 60), Aggregate Production 48 (Chicago Plant for First Semester of year 2024, Dividing shifts by 2), Tiered Costs 480, Transportation). The 'Destination scenario(s)' section shows 'Selected tables (48)' and a 'Filter by labels' dropdown. On the right, a 'Select type' dropdown is highlighted with a red box, showing options: 'Applied for all tables' (selected), 'Full Replacement' (radio button selected), 'Update Without Append', and 'Update and Append'. At the bottom right are 'Save & Preview' and 'Preview' buttons.

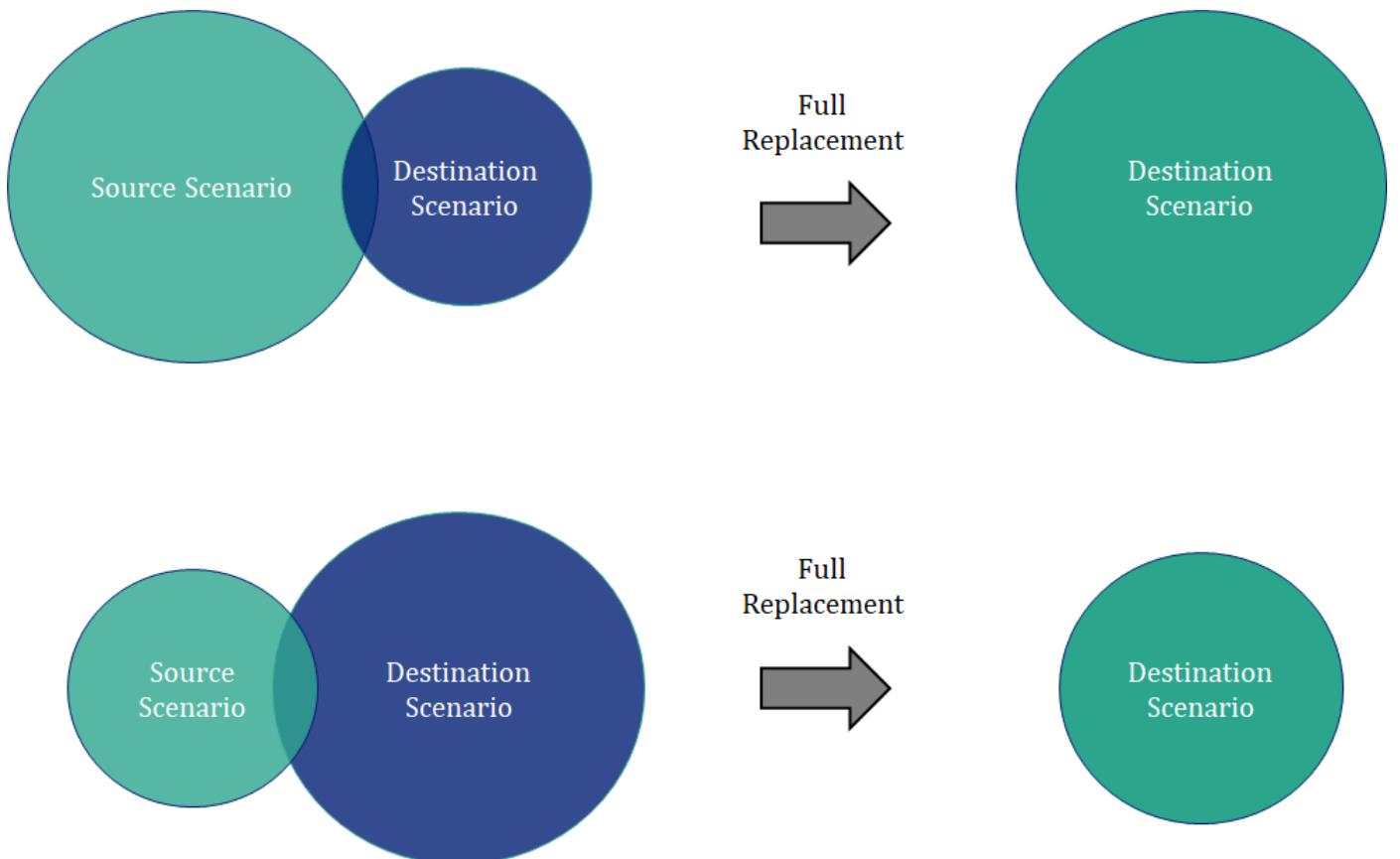
*Bulk Data Operations page*

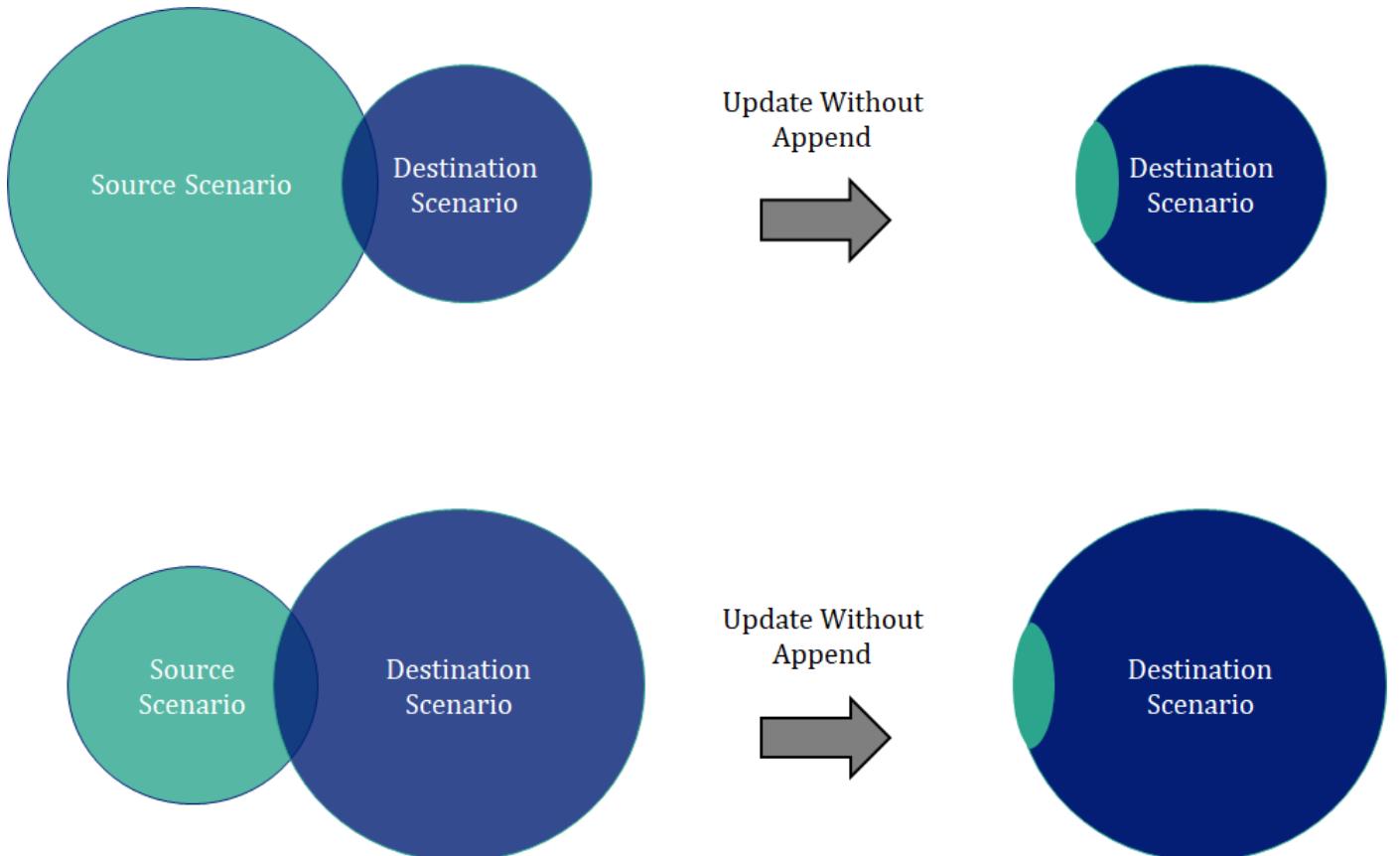
- **Full Replacement:** The source scenario data will replace ALL destination scenario data.
- **Update without Append:** Replace only the records that ARE MATCHING between the source and destination scenarios.
- **Update and Append:** Replace the records that ARE MATCHING and, if the source scenario has additional records, new records will be ADDED into the destination scenario.

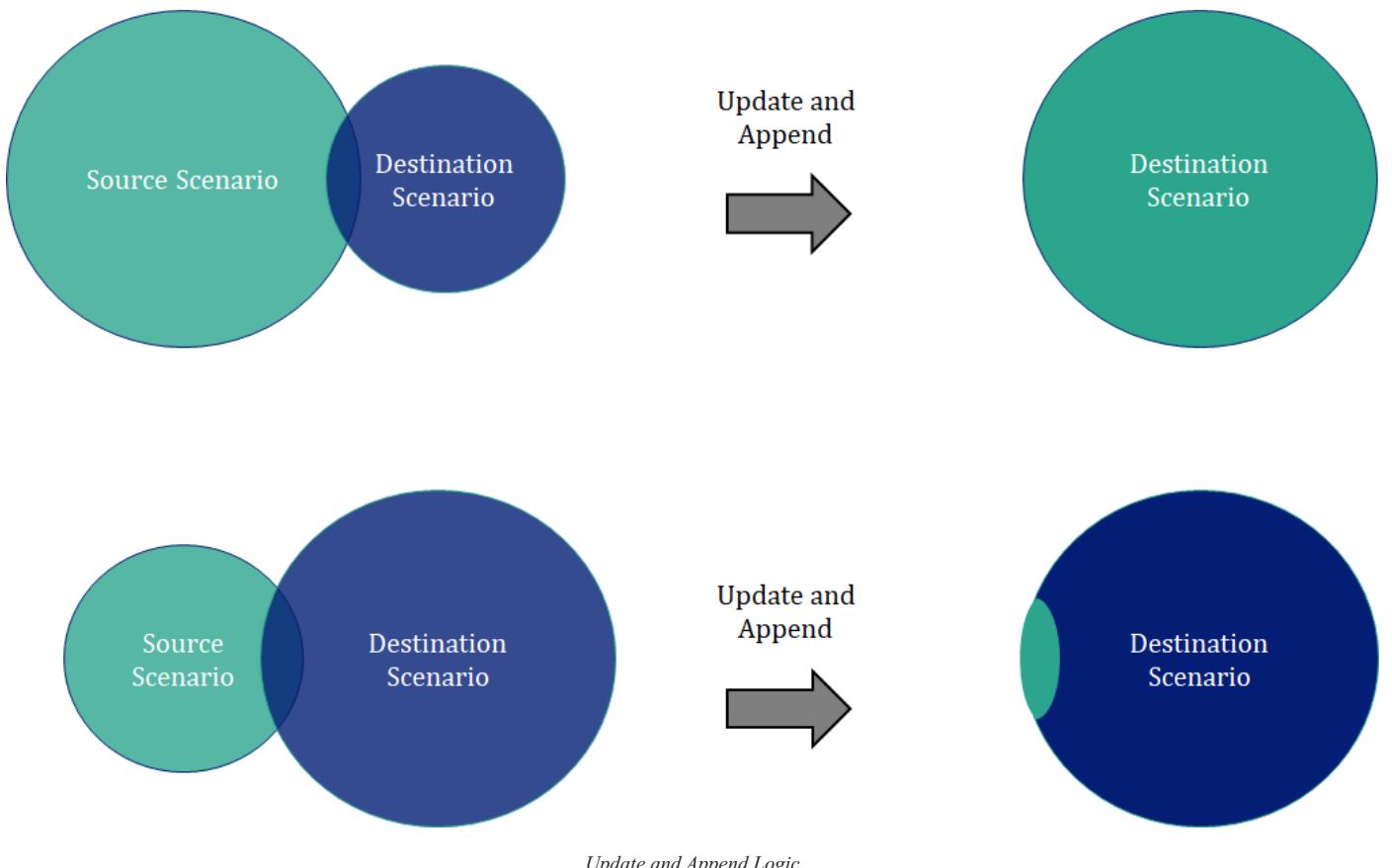
**Note**

By deactivating the **Applied for all tables** slider, you can customize the replacement type by each table separately.

Take a look at the illustration below to visually understand how each replacement type works.







The **Configuration** is ready, click the **Save as New** button.

The screenshot shows the 'Bulk Data Operations' page in the Decision Spot interface. The 'Save as New' button is highlighted with a red box. The configuration interface includes sections for 'Source scenario' (with 'Baseline' selected), 'Selected tables (48)', 'Destination scenario(s)' (with 'Baseline' and 'Scenario 1' selected), and 'Select type' (with 'Full Replacement' selected). The main area displays a tree view of selected tables and actions, such as 'Parameters 26', 'Demand and Production' (with 'Demand 3000' and 'Production 60' checked), 'Aggregate Production 48' (with 'Chicago Plant for First Semester of year 2024' and 'Dividing shifts by 2' actions), 'Tiered Costs 480', and 'Transportation'.

*Bulk Data Operations page*

A pop-up will appear, give a **Name** to the configuration and click **Save**.

The screenshot shows the 'Bulk Data Operations' page in the Decision Spot interface. On the left, under 'Source scenario', there's a tree view with 'Baseline' selected. Under 'Demand and Production', several items like 'Demand 3000' and 'Production 60' are checked. In the center, a modal dialog is open with the title 'Save new configuration as' and the content 'Reducing shifts to one at Chicago Plant for first Semester'. At the bottom of the dialog are 'Cancel' and 'Save' buttons. A red box highlights this dialog. On the right, the 'Destination scenario(s)' section shows 'Selected tables (48)' and 'Operators'. Below it, 'Filter by labels' and 'Search' fields are present. At the bottom right are 'Save & Preview' and 'Preview' buttons.

*Bulk Data Operations page*

A green toast notification will appear at the bottom left corner indicating that the Configuration was saved successfully, and the same will be now available in Configuration dropdown.

This screenshot shows the same 'Bulk Data Operations' page after the configuration has been saved. The 'Destination scenario(s)' section now includes the newly saved configuration under 'Selected tables (48)'. A red box highlights the green toast message 'Successfully added new configuration' at the bottom left. The rest of the interface is identical to the previous screenshot, showing the configuration tree and save buttons.

*Bulk Data Operations page*

Once a **Configuration** is created and saved, it is persisted and can be utilized as many times as needed!

### Using a Configuration

A configuration is used to perform the following sequence of steps:

1. Get the data from a **Source Scenario**
2. Apply a set of logical and mathematical operations (Filters/Transformations) on this data
3. Load the result in a **Destination Scenario**

This is the essence of Bulk Data Operations (BDO). Select the saved **Configuration** from the dropdown (1) then, Select the **Destination Scenario** from the menu on the right corner (2).

The screenshot shows the Bulk Data Operations page in the Decision Spot interface. At the top, there's a navigation bar with 'My Project > Supply Chain Network Optimizer'. Below it is a toolbar with 'Bulk Data Operations' (selected), 'Manage Filter' (with a red box around it and a green '1' over it), 'Manage Transformations', and other buttons like 'Save' and 'Save as New'. The main area is divided into two main sections: 'Source scenario' and 'Destination scenario(s)'.

**Source scenario:** This section contains a dropdown menu showing 'Reducing shifts to one at Chicago Plant for first Semester o...', a 'Baseline' dropdown, and a 'Hide empty tables' toggle. It lists several checked items under 'Demand and Production' and 'Aggregate Production', each with an 'Add Action' button. There's also a section for 'Tiered Costs' and 'Transportation'.

**Destination scenario(s):** This section has a 'Selected tables (48)' header and an 'Operators' button. It includes a 'Filter by labels' dropdown and a 'Search' input field. A red box highlights the 'Scenario 1' checkbox, which is checked. Below this is a 'Select type' section with radio buttons for 'Full Replacement' (selected), 'Update Without Append', and 'Update and Append'. At the bottom right of the main area are 'Save & Preview' and 'Preview' buttons.

Bulk Data Operations page

After selecting the Destination Scenario, the **Preview** option will be available (or the **Save and Preview**, if in case additional changes have been made to a saved Configuration).

Select this option preview the data in the Destination Scenario.

The screenshot shows the Bulk Data Operations page in the Decision Spot interface. The left panel displays a hierarchical configuration tree with sections such as Core Data, Demand and Production, Aggregate Production, and Transportation. The central area features a grid showing specific actions like "Dividing shifts by 2". The right side includes tabs for "Selected tables (36)", "Operators (1)", "Destination scenario(s)", and "Select type" (set to "Full Replacement"). A red box highlights the "Preview" button at the bottom right.

*Bulk Data Operations page***Pro-tip**

Users don't necessarily need to save a configuration to preview and execute it but it is always a good practice to save. Building configurations require critical thinking. Therefore, users shouldn't spend time building the same configurations over and over again.

In the **Preview**, users can select any table that is copied over (1) and navigate through the **Grid** (2), and verify if the Configuration was properly applied on the input data.

Refer to the case continued from [Example 2](#), Notice that the filters and transformations were effective, because the `Site=Chicago plant` has its `Max Hours` divided by 2 for `Time Period`'s of Q1-2024 and Q2-2024 (3).

In the bottom (4), one can also check the SQL statement related to all filters and transformations that were set up in the Configuration.

The screenshot shows the 'Data Preview' section with a red box highlighting the table. The table contains data for Plant Denver and Plant Chicago across various time periods. A green callout '3' points to the last row of the table. The 'SQL Script' section below it contains two lines of SQL code for updating the 'max\_hours' value for Plant Chicago in Q1 and Q2 2024.

Site	Product Group	Time Period	Min H...	Max H...	Tiered Cost
Plant Denver	*all*	Q2-2025	0	1080	
Plant Denver	*all*	Q2-2026	0	1080	
Plant Denver	*all*	Q3-2024	0	1080	
Plant Denver	*all*	Q3-2025	0	1080	
Plant Denver	*all*	Q3-2026	0	1080	
Plant Denver	*all*	Q4-2024	0	1080	
Plant Denver	*all*	Q4-2025	0	1080	
Plant Denver	*all*	Q4-2026	0	1080	
Plant Chicago	*all*	Q1-2024	0	540	
Plant Chicago	*all*	Q2-2024	0	540	

**SQL Script**

```
Select * Into l_aggregate_production from l_aggregate_production;
Update l_aggregate_production SET max_hours = (max_hours/2) Where ( site = 'Plant Chicago' ) and (( time_period = 'Q1-2024' ) or ( time_period = 'Q2-2024' ));
```

*Preview page*

Once verified, click **Confirm** to finish loading the data in the Destination Scenario or click **Back** if there are any additional changes required.

This screenshot is identical to the one above, showing the 'Data Preview' and 'SQL Script' sections. The 'SQL Script' section contains the same update query for Plant Chicago's maximum hours.

*Preview page*

A green toast message will be shown at the left confirming the Bulk Data Operation.

Source scenario

Selected tables (48) Destination scenario(s)

Operators

Filter by labels Search

Baseline Scenario 1

Select type

Applied for all tables

Full Replacement

Update Without Append

Update and Append

Save & Preview Preview

Bulk Data Operations page

The input data now is ready to be used at your **Destination Scenario** now. Have fun!

Site	Product Group	Time Period	Min Hours	Max Hours	Tiered Cost
Plant Dallas	*all*	Q1-2025	0	1080	
Plant Dallas	*all*	Q1-2026	0	1080	
Plant Dallas	*all*	Q2-2024	0	1080	
Plant Dallas	*all*	Q2-2025	0	1080	
Plant Dallas	*all*	Q2-2026	0	1080	
Plant Dallas	*all*	Q3-2024	0	1080	
Plant Dallas	*all*	Q3-2025	0	1080	
Plant Dallas	*all*	Q3-2026	0	1080	
Plant Dallas	*all*	Q4-2024	0	1080	
Plant Dallas	*all*	Q4-2025	0	1080	
Plant Dallas	*all*	Q4-2026	0	1080	
Plant Denver	*all*	Q1-2024	0	1080	
Plant Denver	*all*	Q1-2025	0	1080	
Plant Denver	*all*	Q1-2026	0	1080	
Plant Denver	*all*	Q2-2024	0	1080	
Plant Denver	*all*	Q2-2025	0	1080	
Plant Denver	*all*	Q2-2026	0	1080	
Plant Denver	*all*	Q3-2024	0	1080	
Plant Denver	*all*	Q3-2025	0	1080	
Plant Denver	*all*	Q3-2026	0	1080	
Plant Denver	*all*	Q4-2024	0	1080	
Plant Denver	*all*	Q4-2025	0	1080	
Plant Denver	*all*	Q4-2026	0	1080	
Plant Chicago	*all*	Q1-2024	0	540	
Plant Chicago	*all*	Q2-2024	0	540	

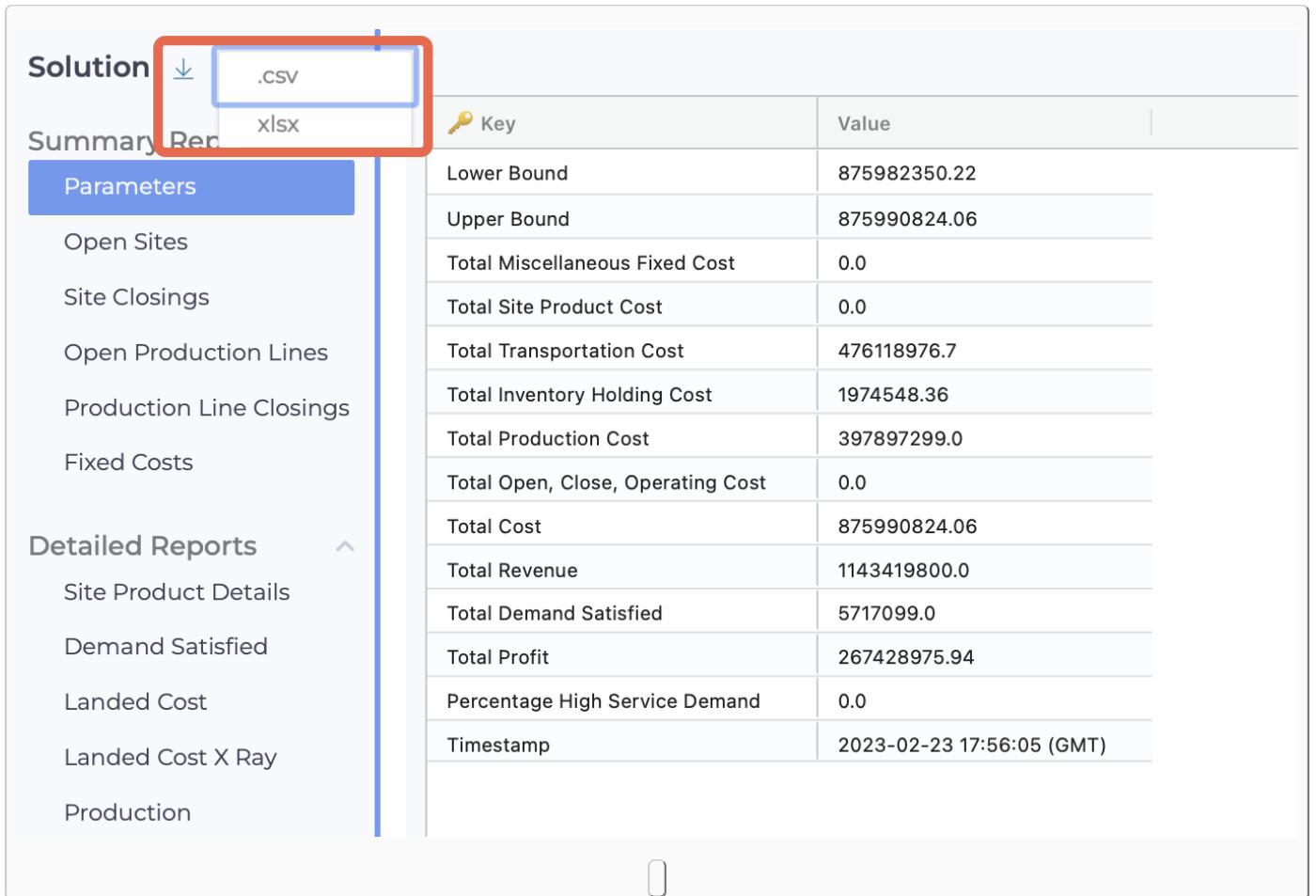
Scenarios page

### 2.3.5 Exporting Solutions

---

Foresta applications provides user with a way to export solutions to xlsx / csv format. There are three ways to export solutions from an app.

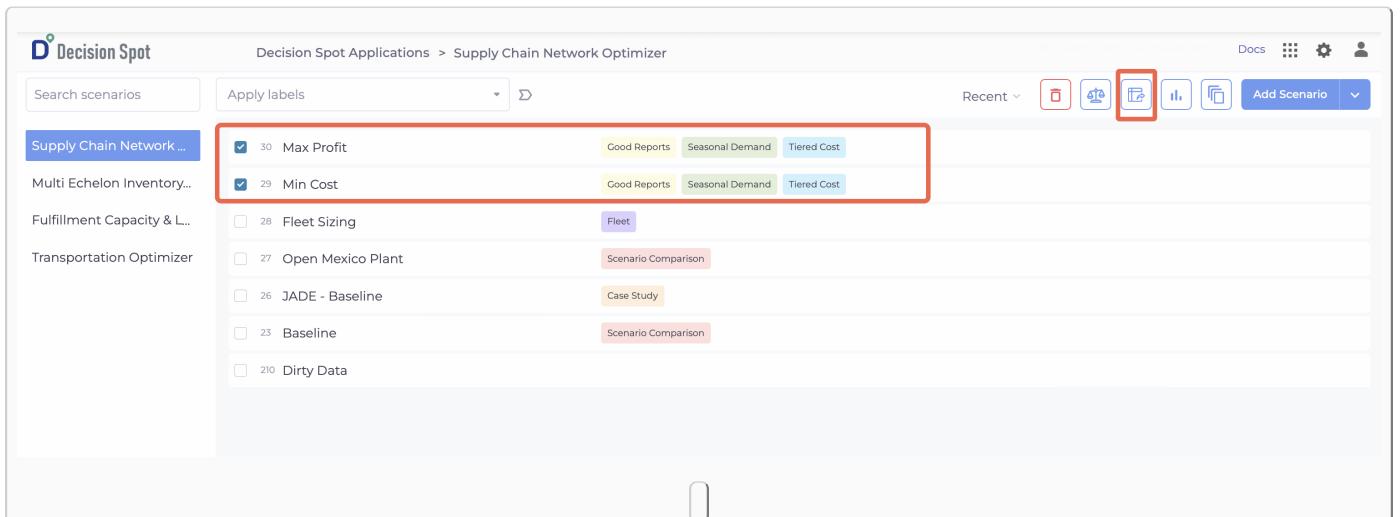
1. Export **all** output tables for 1 scenario. Navigate to Scenario > Solution section and click on the download button next to the **Solution** and select whether to export it as csv / xlsx.



The screenshot shows the 'Solution' section of a software interface. On the left, there's a sidebar with sections like 'Parameters', 'Open Sites', 'Site Closings', 'Open Production Lines', 'Production Line Closings', 'Fixed Costs', and 'Detailed Reports'. Under 'Detailed Reports', there are items like 'Site Product Details', 'Demand Satisfied', 'Landed Cost', 'Landed Cost X Ray', and 'Production'. On the right, there's a table titled 'Key' and 'Value' with various metrics. At the top left of the main area, there are download buttons for '.CSV' and 'xlsx', both enclosed in a red box. The table data includes:

Key	Value
Lower Bound	875982350.22
Upper Bound	875990824.06
Total Miscellaneous Fixed Cost	0.0
Total Site Product Cost	0.0
Total Transportation Cost	476118976.7
Total Inventory Holding Cost	1974548.36
Total Production Cost	397897299.0
Total Open, Close, Operating Cost	0.0
Total Cost	875990824.06
Total Revenue	1143419800.0
Total Demand Satisfied	5717099.0
Total Profit	267428975.94
Percentage High Service Demand	0.0
Timestamp	2023-02-23 17:56:05 (GMT)

2. Export **all** output tables for multiple scenarios. On the application homepage, select the scenarios to export and click on **EXPORT** button. This will export a **UNION** by scenario of each of the output tables in an xlsx file.



The screenshot shows the 'Decision Spot Applications' homepage. On the left, there's a sidebar with sections like 'Supply Chain Network ...', 'Multi Echelon Inventory...', 'Fulfillment Capacity & L...', 'Transportation Optimizer', and a list of scenarios: '30 Max Profit', '29 Min Cost', '28 Fleet Sizing', '27 Open Mexico Plant', '26 JADE - Baseline', '23 Baseline', and '210 Dirty Data'. The first two scenarios, '30 Max Profit' and '29 Min Cost', are selected and highlighted with a red box. On the right, there are several export and management buttons: 'Recent', a download icon, a refresh icon, a settings icon, and a 'Docs' button. Below these are buttons for 'Add Scenario', 'Edit Scenario', 'Delete Scenario', and 'Run Scenario'.

3. Individual tables from a solution in a scenario can be exported to csv format using the **Download** button at the top of the grid.

Hide empty tables

Search

**Solution**

- Summary Reports
  - Parameters 14
  - Open Sites 648
- Detailed Reports
  - Site Product Details 3216
  - Demand Satisfied 2976
    - Landed Cost 2976**
    - Production 240
    - Aggregate Production 48
  - Transportation Reports
    - Arcs 3036
    - Transportation Details 253
  - Infeasibility Reports
    - Bottlenecks 60

 **Site** | **Product** | **Time Period** | **Type** | **Units** | **Production Cost** | **Inventor**

Site	Product	Time Period	Type	Units	Production Cost	Inventor
Customer 1	Clear	Q1-2012	Demand Satisfied	1162.57	581285	
Customer 1	Clear	Q2-2012	Demand Satisfied	1162.57	581285	
Customer 1	Clear	Q3-2012	Demand Satisfied	1162.57	581285	
Customer 1	Clear	Q4-2012	Demand Satisfied	1162.57	581285	
Customer 2	Clear	Q1-2012	Demand Satisfied	2720.68	1360340	
Customer 2	Clear	Q2-2012	Demand Satisfied	2720.68	1360340	
Customer 2	Clear	Q3-2012	Demand Satisfied	2720.68	1360340	
Customer 2	Clear	Q4-2012	Demand Satisfied	2720.68	1360340	
Customer 3	Clear	Q1-2012	Demand Satisfied	1980.17	990085	
Customer 3	Clear	Q2-2012	Demand Satisfied	1980.17	990085	
Customer 3	Clear	Q3-2012	Demand Satisfied	1980.17	990085	
Customer 3	Clear	Q4-2012	Demand Satisfied	1980.17	990085	
Customer 4	Clear	Q1-2012	Demand Satisfied	524.77	262385	
Customer 4	Clear	Q2-2012	Demand Satisfied	524.77	262385	
Customer 4	Clear	Q3-2012	Demand Satisfied	524.77	262385	
Customer 4	Clear	Q4-2012	Demand Satisfied	524.77	262385	
Customer 5	Clear	Q1-2012	Demand Satisfied	1230.08	615040	
Customer 5	Clear	Q2-2012	Demand Satisfied	1230.08	615040	
Customer 5	Clear	Q3-2012	Demand Satisfied	1230.08	615040	
Customer 5	Clear	Q4-2012	Demand Satisfied	1230.08	615040	
Customer 6	Clear	Q1-2012	Demand Satisfied	1269.45	634725	
Customer 6	Clear	Q2-2012	Demand Satisfied	1269.45	634725	
Customer 6	Clear	Q3-2012	Demand Satisfied	1269.45	634725	

## 2.3.6 Workflows

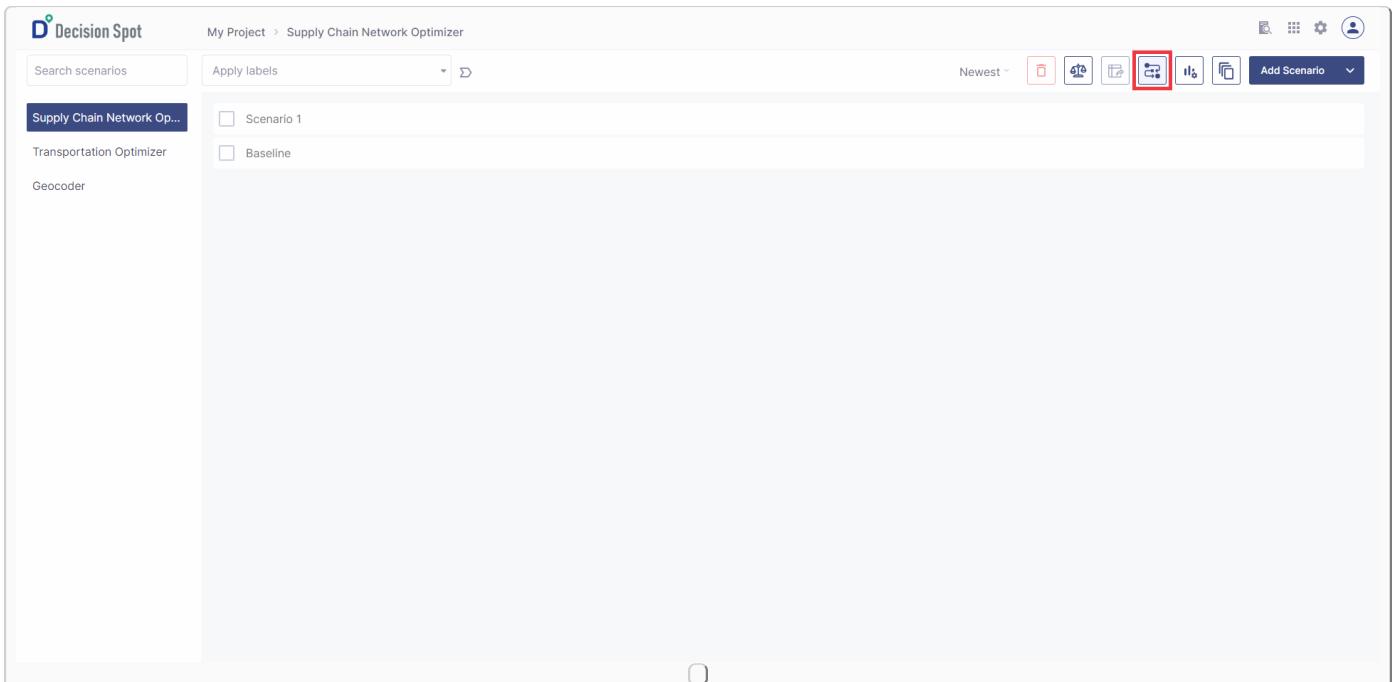
Consider the following situation: A '*Supply Chain Distribution Planning*' project requires 4 steps. First, run a Geocoding function to obtain Latitudes and Longitudes for locations followed by running a Network Optimization model to determine the optimal lanes, then transfer the the optimal lanes in a Transportation Optimization app to finally run another model to generate an operational plan.

Now, a user can either open each one individually, performing extractions, uploads, validations, and solves on different screens, or can benefit from the **Workflow** feature. Workflow integrates all the functionalities (actions, solve, validate, visualizations) of a project in a single screen. This enables rapid execution of scenarios in a project.

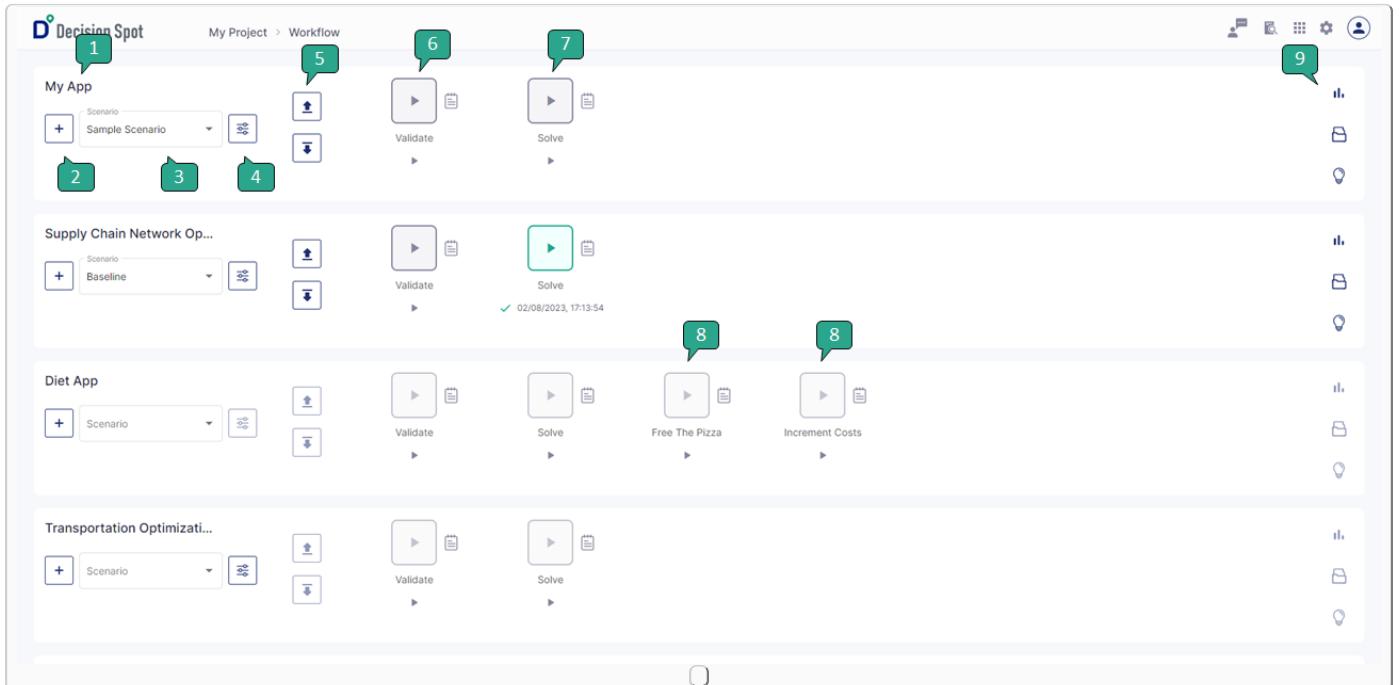
If you're a pilot and your project is an airplane, the Workflow is your cockpit. You can access and control all the different controls (apps and scenarios) of the airplane at one single box.

### Using Workflows

Within your **Project**, select **Open Workflow** in top right corner of the page.



In the Workflow page, the users will see all existing **Apps** (1) and **Scenarios** (3) of the project in the left.



Workflow Screen

- Add a New Scenario (2) on this page
- Control the Parameters (4) of a selected scenario
- Download/Upload (5) data into a selected scenario of an app
- Run Validate (6) and Solve (7) function,
- Access the Logs generated by functions (notebook icons at the right of validation and solve buttons).
- If the App has embedded Actions, users can run them here (8).
- Utilize the shortcuts on the right (9), to directly go to Visualizations, the Input Data and the Solution Tables of the selected scenario.

## 2.3.7 Validation Tables

Validation Tables in Foresta are designed to ensure the data integrity of the input data. A summary of errors is displayed, along with the option to locate the exact rows where these errors occur. Concise error messages are also provided, which are either system generated or user-defined, making it easy to understand and diagnose the errors.

### Using Validation Tables

To run data validation on the input data, use the **Start data validation** option as shown below.

Name	Active	Latitude	Longitude	Sourcing Status	Open Close Status	Inventory Status	Opening
C_Chattanooga	True	35.045556	-85.309722	Multiple	Fixed	Flow Through Only	
C_Springfield	True	37.215278000000005	-93.298056	Multiple	Fixed	Flow Through Only	
C_Phoenix-Mesa	True	33.448333	-112.073333	Multiple	Fixed	Flow Through Only	
C_Houston	True	29.763056	-95.363056	Multiple	Fixed	Flow Through Only	
C_Brazoria	True	29.044166999999998	-95.568889	Multiple	Fixed	Flow Through Only	
C_Knoxville	True	35.960556	-83.920833	Multiple	Fixed	Flow Through Only	
C_San Diego	True	32.715278000000005	-117.1563889999999	Multiple	Fixed	Flow Through Only	
C_Seattle	True	47.606389	-122.33083300000001	Multiple	Fixed	Flow Through Only	
C_Orange County	True	33.472791	-117.6945169999999	Multiple	Fixed	Flow Through Only	
C_Lexington	True	38.049167	-84.50027800000002	Multiple	Fixed	Flow Through Only	
C_Milwaukee	True	43.03889000000005	-87.906389	Multiple	Fixed	Flow Through Only	
C_San Antonio	True	29.423889000000003	-98.493333	Multiple	Fixed	Flow Through Only	
C_Brownsville	True	25.901389	-97.497222	Multiple	Fixed	Flow Through Only	
C_Dallas	True	32.783333	-96.8	Multiple	Fixed	Flow Through Only	
C_Omaha	True	41.258611	-95.9375	Multiple	Fixed	Flow Through Only	
C_Los Angeles	True	33.472791	-117.6945169999999	Multiple	Fixed	Flow Through Only	
C_Columbus	True	39.96111099999995	-82.998889	Multiple	Fixed	Flow Through Only	
C_Santa Rosa	True	38.440556	-122.713333	Multiple	Fixed	Flow Through Only	
C_Reno	True	39.529722	-119.81277800000001	Multiple	Fixed	Flow Through Only	
C_Tacoma	True	47.253056	-122.443056	Multiple	Fixed	Flow Through Only	
C_Killeen	True	31.116944	-97.7275	Multiple	Fixed	Flow Through Only	
C_Wichita	True	37.69222199999994	-97.337222	Multiple	Fixed	Flow Through Only	
C_Amarillo	True	35.221944	-101.830833	Multiple	Fixed	Flow Through Only	
C_Boise City	True	43.613611	-116.202863	Multiple	Fixed	Flow Through Only	
C_Denver	True	39.739167	-104.984167	Multiple	Fixed	Flow Through Only	

Data Validation option on Project Screen

If there are no validation errors in the input data, a green slider message appears indicating successful validation as shown below. The user can then proceed further.

The screenshot shows a table titled "Production" with columns: Site, Product, Time Period, Min Supply, Max Supply, Cost, Rate, and Tiered Cost. All rows have values of 0 or 25, and the cost and rate are 99999. A green message bar at the bottom left says "Validation success! No validation problems found."

Site	Product	Time Period	Min Supply	Max Supply	Cost	Rate	Tiered Cost
P_Ahoskie	Pellet	2022-2	0	7885	25	99999	
P_Ahoskie	Pellet	2022-3	0	7885	25	99999	
P_Ahoskie	Pellet	2022-4	0	7885	25	99999	
P_Ahoskie	Pellet	2022-5	0	7885	25	99999	
P_Ahoskie	Pellet	2022-6	0	7885	25	99999	
P_Ahoskie	Pellet	2022-7	0	7885	25	99999	
P_Ahoskie	Pellet	2022-8	0	7885	25	99999	
P_Ahoskie	Pellet	2022-9	0	7885	25	99999	
P_Ahoskie	Pellet	2022-10	0	7885	25	99999	
P_Ahoskie	Pellet	2022-11	0	7885	25	99999	
P_Ahoskie	Pellet	2022-12	0	7885	25	99999	
P_Ahoskie	Pellet	2022-13	0	7885	25	99999	
P_Ahoskie	Pellet	2022-14	0	7885	25	99999	
P_Ahoskie	Pellet	2022-15	0	7885	25	99999	
P_Ahoskie	Pellet	2022-16	0	7885	25	99999	
P_Ahoskie	Pellet	2022-17	0	7885	25	99999	
P_Ahoskie	Pellet	2022-18	0	7885	25	99999	
P_Ahoskie	Pellet	2022-19	0	7885	25	99999	
P_Ahoskie	Pellet	2022-20	0	7885	25	99999	
P_Ahoskie	Pellet	2022-21	0	7885	25	99999	
P_Ahoskie	Pellet	2022-22	0	7885	25	99999	
P_Ahoskie	Pellet	2022-23	0	7885	25	99999	
P_Ahoskie	Pellet	2022-24	0	7885	25	99999	
P_Ahoskie	Pellet	2022-25	0	7885	25	99999	
P_Ahoskie	Pellet	2022-26	0	7885	25	99999	

*Data Validation Successful*

In case validation errors are detected, a green slider message with the number of validation problems identified appears as shown below.

The screenshot shows a table titled "Production" with columns: Site, Product, Time Period, Min Supply, Max Supply, Cost, Rate, and Tiered Cost. The first row has a value of 0 for Min Supply instead of 7885. A green message bar at the bottom left says "Validation found 1 problem."

Site	Product	Time Period	Min Supply	Max Supply	Cost	Rate	Tiered Cost
P_Ahoskie	Pellet	2022-2	7885	0	25	99999	
P_Ahoskie	Pellet	2022-3	0	7885	25	99999	
P_Ahoskie	Pellet	2022-4	0	7885	25	99999	
P_Ahoskie	Pellet	2022-5	0	7885	25	99999	
P_Ahoskie	Pellet	2022-6	0	7885	25	99999	
P_Ahoskie	Pellet	2022-7	0	7885	25	99999	
P_Ahoskie	Pellet	2022-8	0	7885	25	99999	
P_Ahoskie	Pellet	2022-9	0	7885	25	99999	
P_Ahoskie	Pellet	2022-10	0	7885	25	99999	
P_Ahoskie	Pellet	2022-11	0	7885	25	99999	
P_Ahoskie	Pellet	2022-12	0	7885	25	99999	
P_Ahoskie	Pellet	2022-13	0	7885	25	99999	
P_Ahoskie	Pellet	2022-14	0	7885	25	99999	
P_Ahoskie	Pellet	2022-15	0	7885	25	99999	
P_Ahoskie	Pellet	2022-16	0	7885	25	99999	
P_Ahoskie	Pellet	2022-17	0	7885	25	99999	
P_Ahoskie	Pellet	2022-18	0	7885	25	99999	
P_Ahoskie	Pellet	2022-19	0	7885	25	99999	
P_Ahoskie	Pellet	2022-20	0	7885	25	99999	
P_Ahoskie	Pellet	2022-21	0	7885	25	99999	
P_Ahoskie	Pellet	2022-22	0	7885	25	99999	
P_Ahoskie	Pellet	2022-23	0	7885	25	99999	
P_Ahoskie	Pellet	2022-24	0	7885	25	99999	
P_Ahoskie	Pellet	2022-25	0	7885	25	99999	
P_Ahoskie	Pellet	2022-26	0	7885	25	99999	

*Data Validation Errors Found*

To locate these errors, click on the **Validation tables** button as shown below.

Site	Product	Time Period	Min Supply	Max Supply	Cost	Rate	Tiered Cost
P_Ahoskie	Pellet	2022-2	7885	0	25	99999	
P_Ahoskie	Pellet	2022-3	0	7885	25	99999	
P_Ahoskie	Pellet	2022-4	0	7885	25	99999	
P_Ahoskie	Pellet	2022-5	0	7885	25	99999	
P_Ahoskie	Pellet	2022-6	0	7885	25	99999	
P_Ahoskie	Pellet	2022-7	0	7885	25	99999	
P_Ahoskie	Pellet	2022-8	0	7885	25	99999	
P_Ahoskie	Pellet	2022-9	0	7885	25	99999	
P_Ahoskie	Pellet	2022-10	0	7885	25	99999	
P_Ahoskie	Pellet	2022-11	0	7885	25	99999	
P_Ahoskie	Pellet	2022-12	0	7885	25	99999	
P_Ahoskie	Pellet	2022-13	0	7885	25	99999	
P_Ahoskie	Pellet	2022-14	0	7885	25	99999	
P_Ahoskie	Pellet	2022-15	0	7885	25	99999	
P_Ahoskie	Pellet	2022-16	0	7885	25	99999	
P_Ahoskie	Pellet	2022-17	0	7885	25	99999	
P_Ahoskie	Pellet	2022-18	0	7885	25	99999	
P_Ahoskie	Pellet	2022-19	0	7885	25	99999	
P_Ahoskie	Pellet	2022-20	0	7885	25	99999	
P_Ahoskie	Pellet	2022-21	0	7885	25	99999	
P_Ahoskie	Pellet	2022-22	0	7885	25	99999	
P_Ahoskie	Pellet	2022-23	0	7885	25	99999	
P_Ahoskie	Pellet	2022-24	0	7885	25	99999	
P_Ahoskie	Pellet	2022-25	0	7885	25	99999	
P_Ahoskie	Pellet	2022-26	0	7885	25	99999	

[Go to Validation Tables](#)

This takes you to the summary table **Validation Failure Summary**, containing the different error types and their respective counts as shown below.

Error Type	Error Count
Data Row Failure	1
Data Type Failure	3

### Validation Failures Summary

Next in the **Normal Validation Details** section, clicking on the tables shows the exact error type and the corresponding location as shown in the following two examples.

Name	Error Type	Latitude	Longitude	Sourcing Status	Open Close Status	Inventory Status
C_Dallas	Data Type Failure : Longitude Data Type Failure : Open Close Status Illegal Active value of True.	32.783333	-196.8	Multiple	Fixed	Flow Through Only
C_San Antonio		29.42388900000003	-98.493333	Multiple	Fixed	Flow Through Only
C_Chattanooga		35.045556	-85.309722	Multiple	Fixed	Flow Through Only

*Locating the Error - Example 1*

The screenshot shows the Decision Spot application interface. On the left, there's a sidebar with a tree view under the 'Validation' node, including 'Summary', 'Validation Failure', 'Normal Validation Details' (which is expanded), and 'Production' (which is selected and highlighted with an orange border). The main area displays a table titled 'Validation'. The table has columns: Site, Product, Time Period, Error Type, Min Supply, Max Supply, and Cost. One row is highlighted with a red border, showing the following data:

Site	Product	Time Period	Error Type	Min Supply	Max Supply	Cost
P_Ahoskie	Pellet	2022-2	Data row failure: Max Supply cannot be smaller than Min Su...	7885	0	25

At the top right of the main area, there are buttons for 'idle', 'download', 'refresh', and 'next'. A 'Quick filter' input field and a '1 Row' indicator are also present.

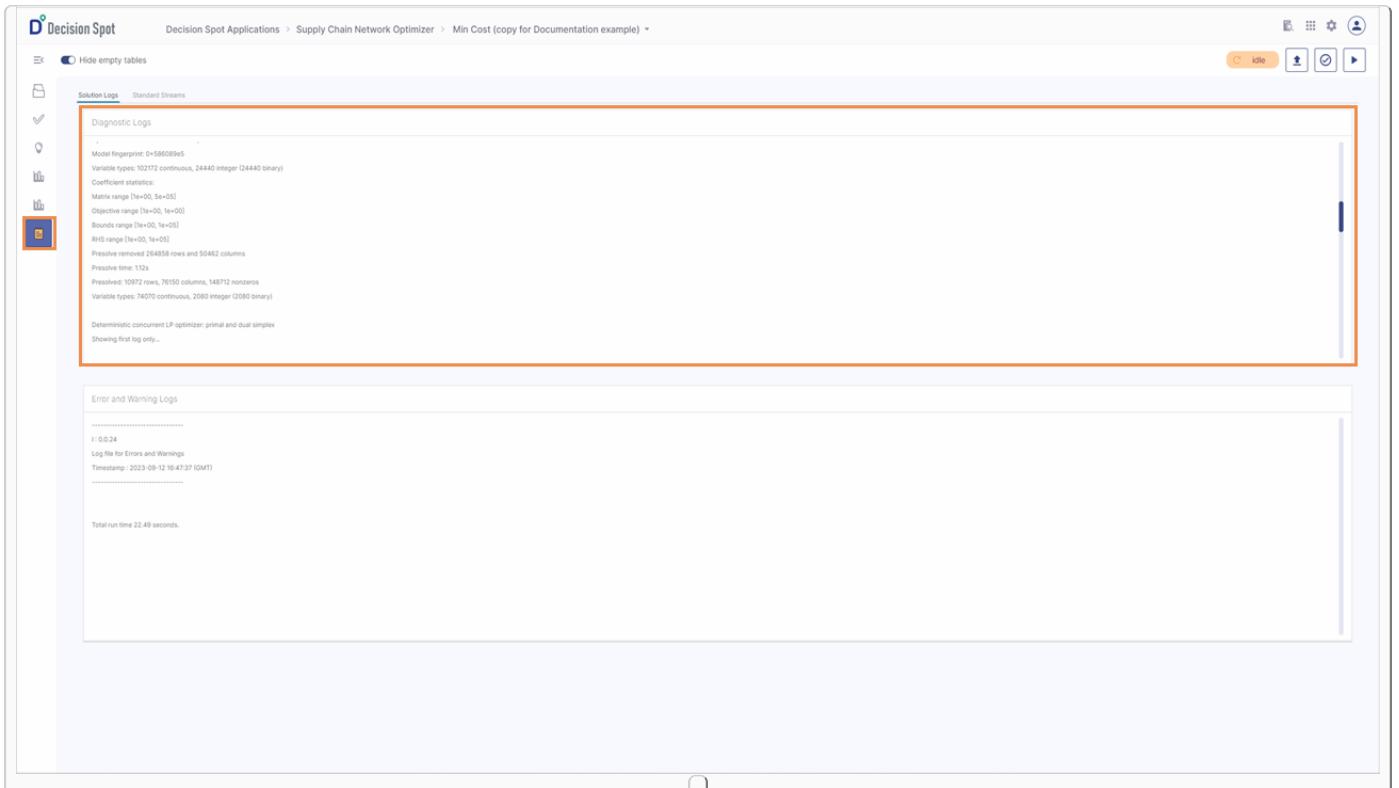
*Locating the Error - Example 2*

## 2.3.8 Logs and Status

Foresta allow users to track the performance of the apps and quickly identify errors and issues as they occur in real time through the **Logs** feature.

### Type of Logs

**1. Diagnostic Logs** - By clicking the Logs button and navigating to Solution Logs, as shown in the image below, users can access Diagnostic Logs that show the real time execution status of the app Solve function. This contains execution times for all sub processes, processor usage, optimization model statistics and parameters, and more.



The screenshot shows the Decision Spot application interface. At the top, it displays the path: Decision Spot Applications > Supply Chain Network Optimizer > Min Cost (copy for Documentation example). Below this, there are two tabs: "Solution Logs" and "Standard Streams". The "Solution Logs" tab is selected and highlighted with a blue border. Under "Solution Logs", the "Diagnostic Logs" section is expanded, showing detailed log output. The log content includes:

```

Model Fingerprint: 0+586089e5
Variable types: 32172 continuous, 24440 integer (24440 binary)
Coefficient statistics:
Matrix range [1e+00, 5e+05]
Objective range [1e+00, 1e+00]
Bounds range [1e+00, 1e+05]
RHS range [1e+00, 1e+05]
Presolve removed 264858 rows and 50462 columns
Presolve time 112s
Presolved: 10971 rows, 76190 columns, 14872 nonzeros
Variable types: 74070 continuous, 2080 integer (2080 binary)

Deterministic concurrent LP optimizer: primal and dual simplex
Showing first log entry...

```

Below the Diagnostic Logs, there is another section titled "Error and Warning Logs" which is currently empty.

Diagnostic Logs in Logs > Solution Logs

**2. Error and Warning Logs** - Below the Diagnostic Logs, as shown in the image below, users can view the logged errors and warnings as they occur while app execution.

**Solution Logs**

Diagnostic Logs

```

Model Fingerprint: 0+58609e5
Variable types: 32172 continuous, 24440 integer (24440 binary)
Coefficient statistics:
Matrix range [1e-05, 5e-05]
Objective range [1e-05, 1e-05]
Bounds range [1e-05, 1e-05]
RHS range [1e-05, 1e-05]
Presolve removed 264858 rows and 50462 columns
Presolve time: 1.12s
Presolved: 10972 rows, 76150 columns, 148712 nonzeros
Variable types: 74070 continuous, 2080 integer (2080 binary)

Deterministic concurrent LP optimizer: primal and dual simplex
Showing first log only...

```

Error and Warning Logs

```

I: 0.0.24
Log file for Errors and Warnings
Timestamp: 2023-09-12 16:47:37 (GMT)
-----
Total run time 22.49 seconds.

```

#### Error and Warning Logs in Logs > Solution Logs

**3. Status Bar** - The status bar at the top right of the Foresta window constantly displays the app status. It shows *Idle* when the app is not running. This bar also shows the particular subprocess that is currently running, as shown in the image below.

**Solution Logs**

Diagnostic Logs

```

Coefficient statistics:
Matrix range [1e-05, 5e-05]
Objective range [1e-05, 1e-05]
Bounds range [1e-05, 1e-05]
RHS range [1e-05, 1e-05]
Presolve removed 264858 rows and 50462 columns
Presolve time: 1.12s
Presolved: 10972 rows, 76150 columns, 148712 nonzeros
Variable types: 74070 continuous, 2080 integer (2080 binary)

Deterministic concurrent LP optimizer: primal and dual simplex
Showing first log only...

```

Concurrent spin time: 0.20s

Error and Warning Logs

```

I: 0.0.24
Log file for Errors and Warnings
Timestamp: 2023-09-12 16:47:37 (GMT)
-----
Total run time 22.49 seconds.

```

*Status Bar*

## 2.4 Reporting

### 2.4.1 Connect with external reporting tools

Foresta applications can seamlessly integrate with popular BI and visualization tools available in the market today. Users can build reports or pull data directly from the Foresta database.

Any external agency (ETL, BI Software, Visualization etc.) would require 3 things to connect with Foresta. Enter the appropriate details in the tool of your choice, and connect!

#### 1. Database Type/Driver:

This is `postgresql`



#### 2. Hostname

The hostname is the server URL that is seen in the browser.

##### Note

The usual value of hostname is `foresta-<organization_name>.decisionspot.com`

#### 3. Database, Username & Password

Create a database connection as instructed to get Database, Username and Password.

##### Foresta ❤️ PowerBI and tableau

Custom reports created using Microsoft PowerBI and Tableau (by Salesforce) can be integrated within Foresta apps and made available to business users. Learn more about integrating PowerBI and Tableau visualizations in Foresta apps.

- [Click here](#) for PowerBI
- [Click here](#) for Tableau

## 2.4.2 Create a database connection

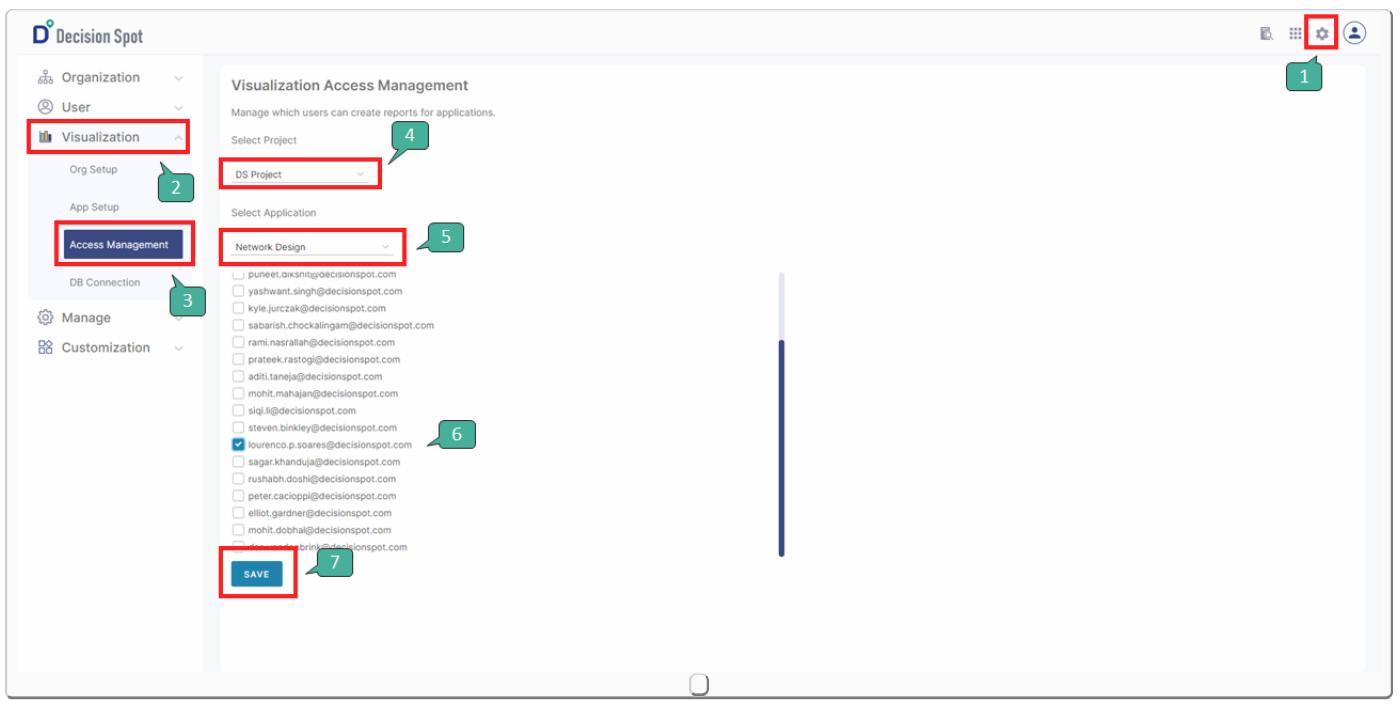
This is done in 2 steps:

1. Assign a user access to projects/applications for visualization
  2. Get the application database name and setup user credentials

## **1. Assign user access to projects/applications for visualization**

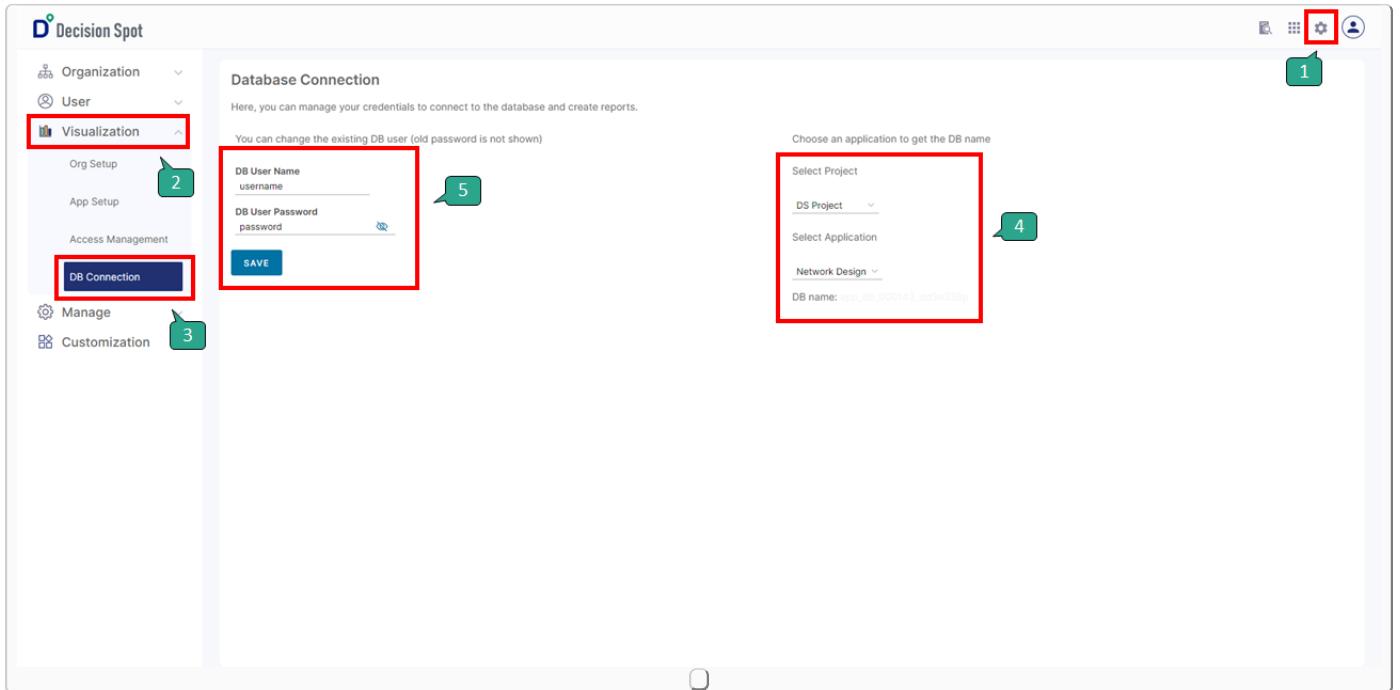
The first step would be to obtain a user access to the projects/applications for establishing secure connection from to the Foresta servers. The **Visualization Access Management** section can be found in **Settings** (1) > **Visualization** (2) > **Access Management** (3).

On selecting the relevant project and application (4 & 5), a list of all users will appear (6). Select the appropriate users to permit access to and click **Save** (7).



The database credentials for the application can be found within the **Settings** (1) of the Foresta server. Within Foresta settings page, navigate to the **DB Connection** (2) section under the **Visualizations** (3).

Select the **Project** and **Application** (4) on the right dropdown to get the Database name (DB Name) for the application the user wants to create visualization for, and enter the Application (5) Name and Project ID that will be used to connect with the database.



DB Connection Page

**Note**

If the required app is not available in the dropdown, please reach out to [Foresta support](#) to get the DB name.

**Warning**

If the credentials are already set up, user is required to change the credentials.

## 2.4.3 PowerBI Integration

### PowerBI



Foresta offers a seamless integration with Microsoft PowerBI that allows users to develop their own reports and visualizations. Further, those reports can be easily embedded into the applications to serve the business needs effectively.

To integrate power BI visualizations into Foresta, Users first connect PowerBI to the Foresta database, and build their dashboards.

Now, there are two types of visualizations that users can make:

1. **Individual Scenario:** To plot charts and derive insights based on data in a particular [scenario](#) of the app
2. **Scenario Comparison:** To plot charts and determine the difference across several scenarios of the app

Everything about building the visualizations is available [here](#). Once the visualizations are finalized in the powerBI, the users can use these visualizations in their system or they have an option to integrate/embed these visualizations in Foresta app. The visualizations are rendered and refreshed after each solve to rapidly surface critical insights from different aspects of data and models.

There are multiple ways in which users can integrate their dashboards within the apps.

#### 1. Manual Publishing Methods

- via. Upload
- via. Linking the server

#### 2. Automatic Methods

Further details are discussed [here](#).

👉 Use the navigation for related information.

## Building PowerBI Visualizations

### NEW TO POWER BI?

We understand this can be overwhelming at times. Here are a few resources for to get basic familiarity with powerBI.

1. Power BI Tutorial - Create Your First Dashboard Now (Practice Files included)
2. How to use Microsoft Power BI - Tutorial for Beginners

### SOURCING DATA

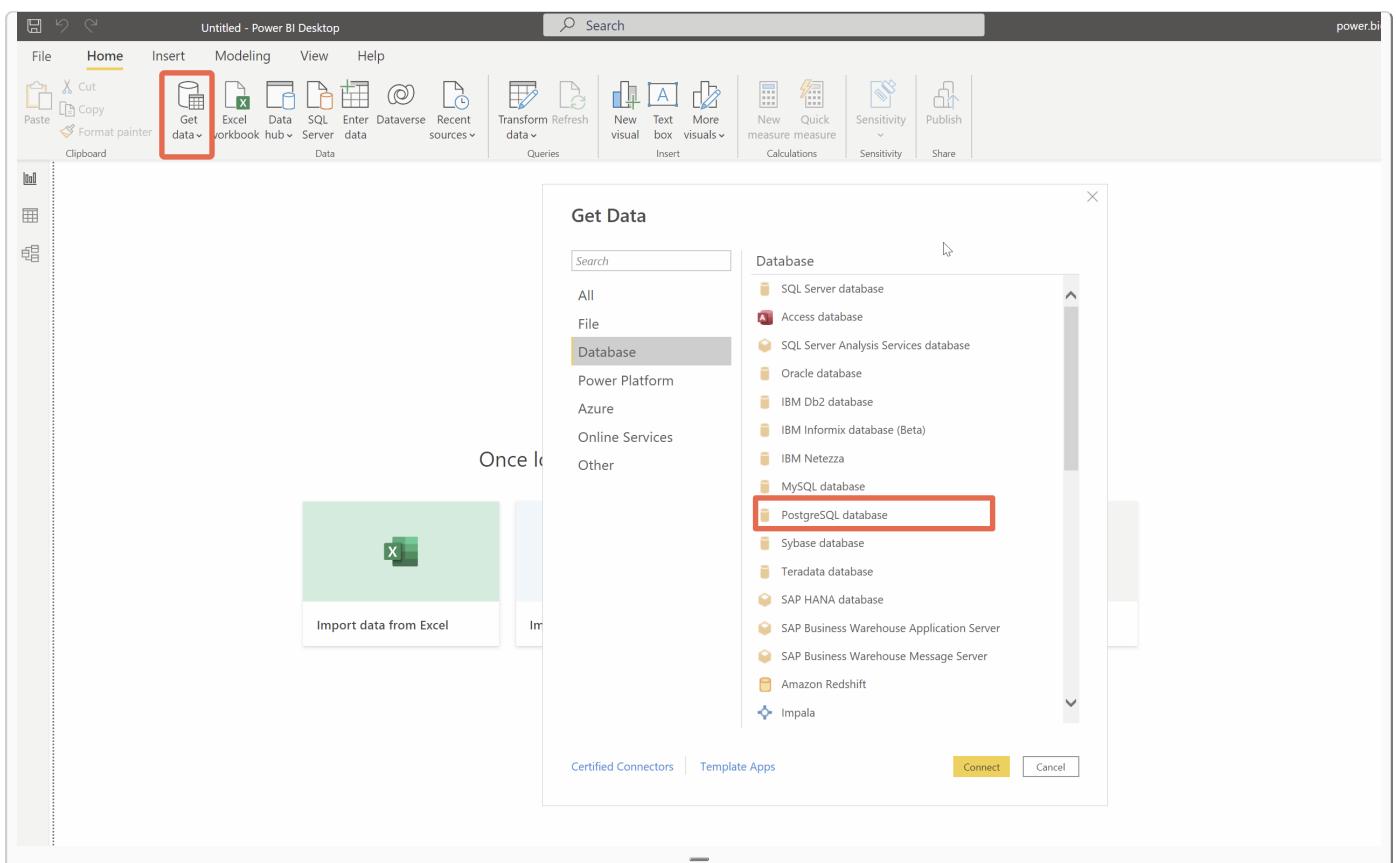
#### 1. Create a database connection

Follow the steps provided [here](#) to get Database Name , Username and Password

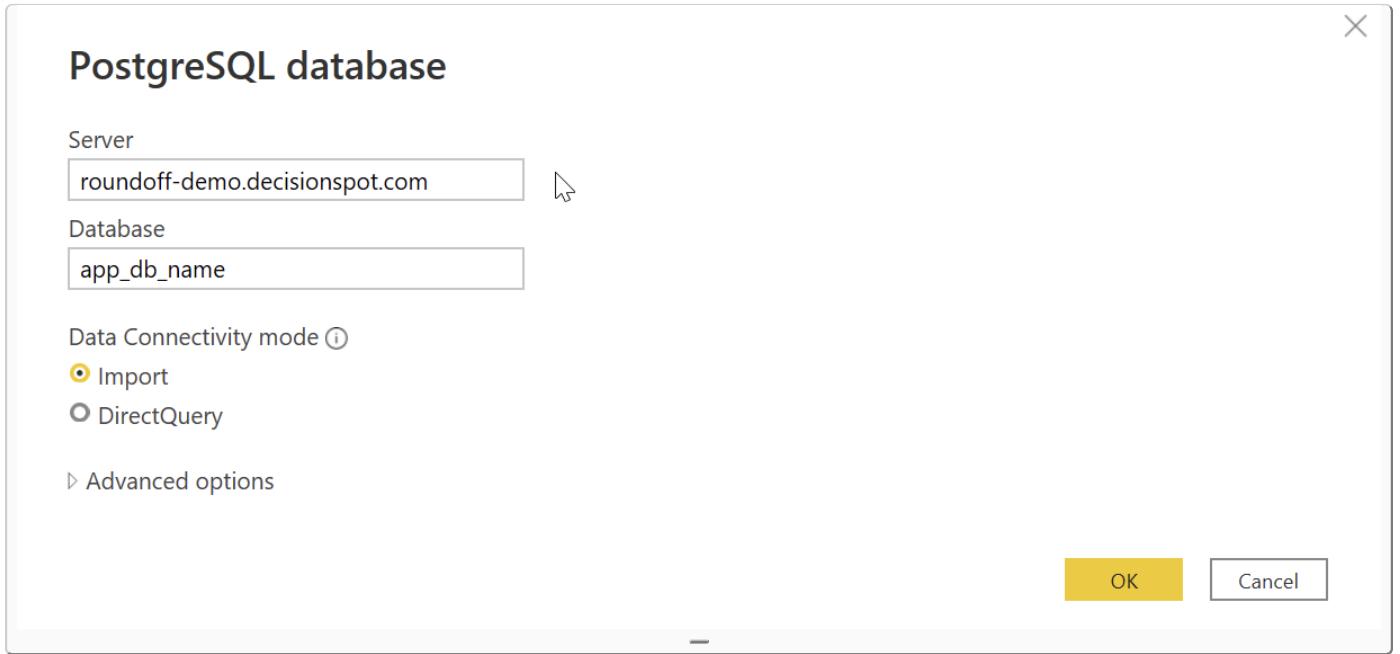
#### 2. Add the connection details

Start PowerBI on your system, In the **Home** ribbon, click **Get Data**. You'll see various types of data sources.

Foresta apps utilize PostgreSQL to store data, click **PostgreSQL database**



Fill-in the details for **Server** and **Database**, Select **Data Connectivity Mode** in the dialog box and click **OK**.



### 3. Set the data connectivity mode

There are two modes to connect to the data source from PowerBI, **DirectQuery** and **Import**.

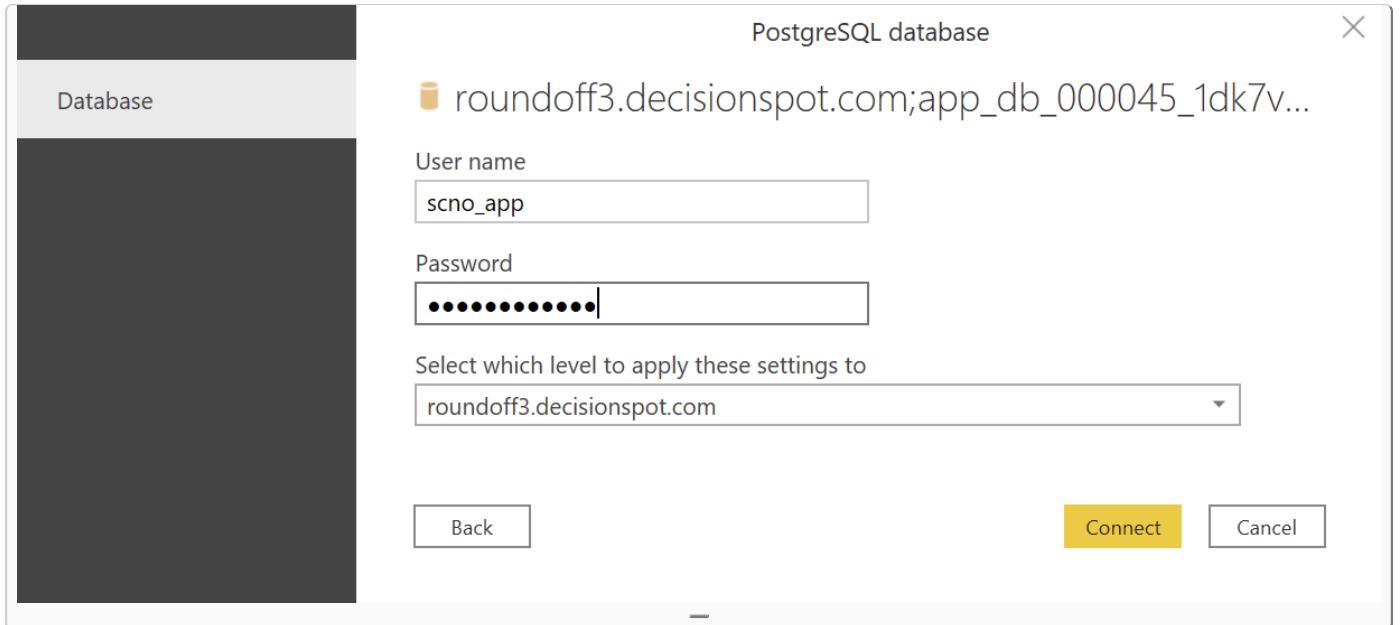
- 1. DirectQuery** - When connecting a data source using the Direct Query technique, the dashboard will query the data source immediately during runtime. Each filter and interaction with the report will result in a series of new queries. Since no data is imported into Power BI, the user can always query the data that already exists in the data source.
- 2. Import** - Using the Import mode of connection, Power BI will cache the data you're linked to, providing a snapshot of the data at a specific point in time. All of data interactions and filters will be applied to this compressed cache source rather than the original data source.

#### Recommendation

**Import** is the preferred mode of connection for Foresta applications since it provides faster dashboard performance while interacting in the GUI. Further, once the dashboard is integrated in an app Foresta has built-in triggers to refresh the cache on PowerBI server at the end of every successful scenario solve.

### 4. Supply user credentials

Enter the **User Name** and **Password** created in Step 1.



Now, click **Connect**.

##### 5. Select data tables

Users will see a `rpts` database with both input and output tables, containing data from all the scenarios in the app.

Select the relevant tables required for creating the visualization from the **Navigator** dialog box.

### Important Note

Ensure `rpts.app_scenarios` is selected as one of the tables while sourcing the data

The screenshot shows the Microsoft PowerBI Navigator interface. On the left, there is a tree view of tables under the connection 'roundoff-demo.decisionspot.com: app\_db...'. The table 'rpts.app\_scenarios' is selected, indicated by a checked checkbox next to its name. To the right of the tree view is a preview of the 'rpts.app\_scenarios' table, which contains the following data:

<b>id</b>	<b>name</b>
205	Ten Cities
209	Open Mexico Plant
210	Open Mexico Plant + DCs
207	UPS Test
208	Baseline
188	Min Cost
204	Big Bakery
206	Three Tier

At the bottom of the interface, there are three buttons: 'Select Related Tables' (disabled), 'Load' (highlighted in yellow), 'Transform Data', and 'Cancel'.

Click on **Load** and start building dashboards in Foresta!

### Important Note

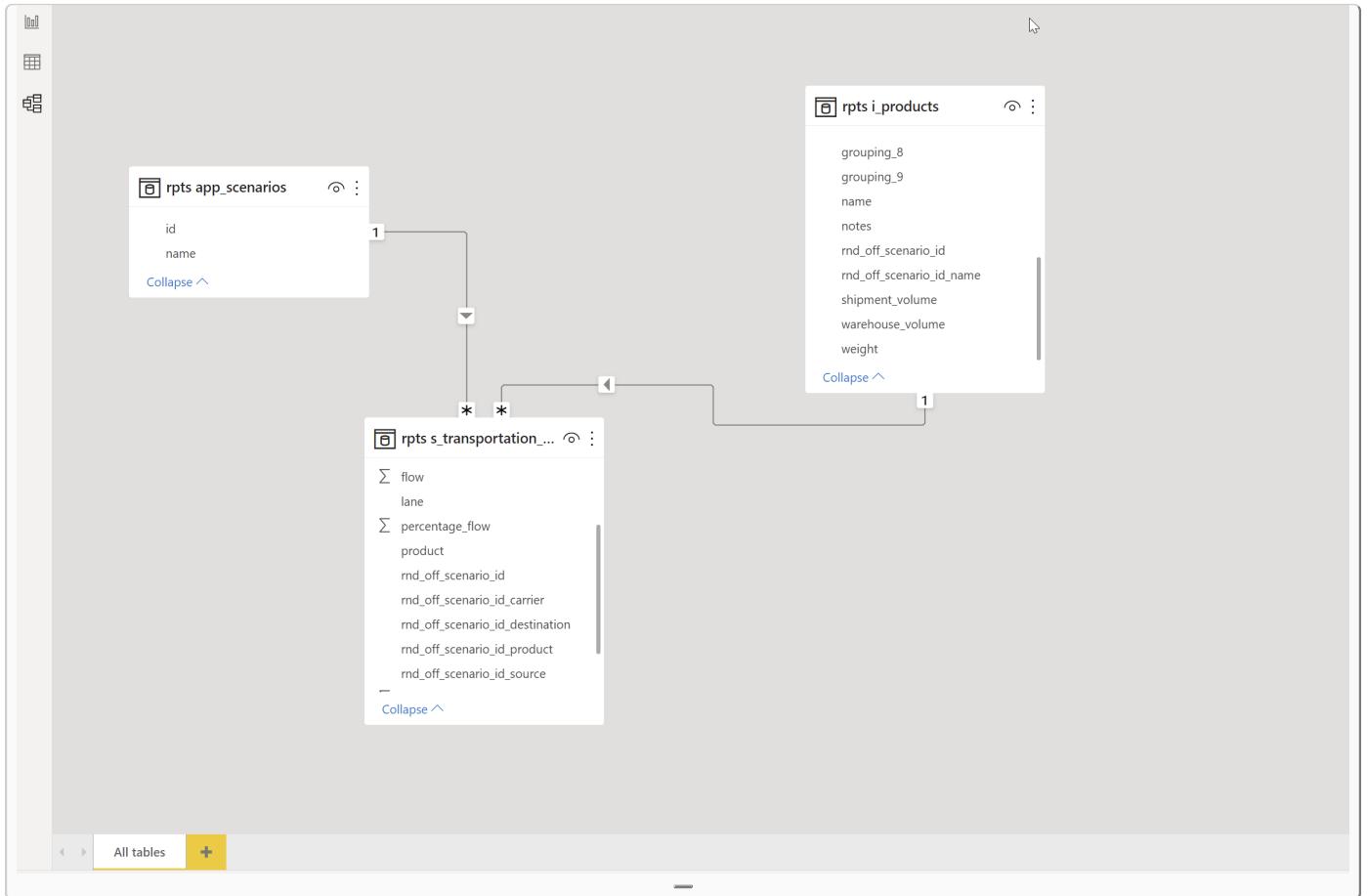
In order to view the input and output tables at this stage, users are required to execute the 'Solve' function.

#### BUILDING AN INDIVIDUAL SCENARIO DASHBOARD

This type of visualizations are built on data in a specific scenario.

To achieve this, [create relationships](#) between the `rpts.app_scenarios` table and the other model tables based on **Scenario ID** field (aka. `rndoff_scenario_id`, `scenario_id`)

The cardinality of these relationships is `one-to-many` type. These relationships may not be auto-detected, in such a case users will need to manually create.



### Important Note

Establishing relationships between `rpts.app_scenarios` table and the other model tables automatically enable Foresta to filter scenarios in **Individual Scenarios**

Ensure establishing these relationships if the aim is to integrate dashboards in Foresta.

### BUILDING SCENARIO COMPARISON DASHBOARDS

While building scenario comparison reports follow these instructions:

1. Deactivate `rpts_app_scenario` relationships
2. Add `scenario_id` fields to either the axis or slicer of your visuals as needed.

### REMARK

Once the dashboards are built, the users can use these visualizations in their system or they have an option to integrate/embed these visualizations in Foresta app. The visualizations are rendered and refreshed after each solve to rapidly surface critical insights from different aspects of data and models.

Technical instructions on How to integrate visualizations is available [here](#)

## Integrating PowerBI Visualizations

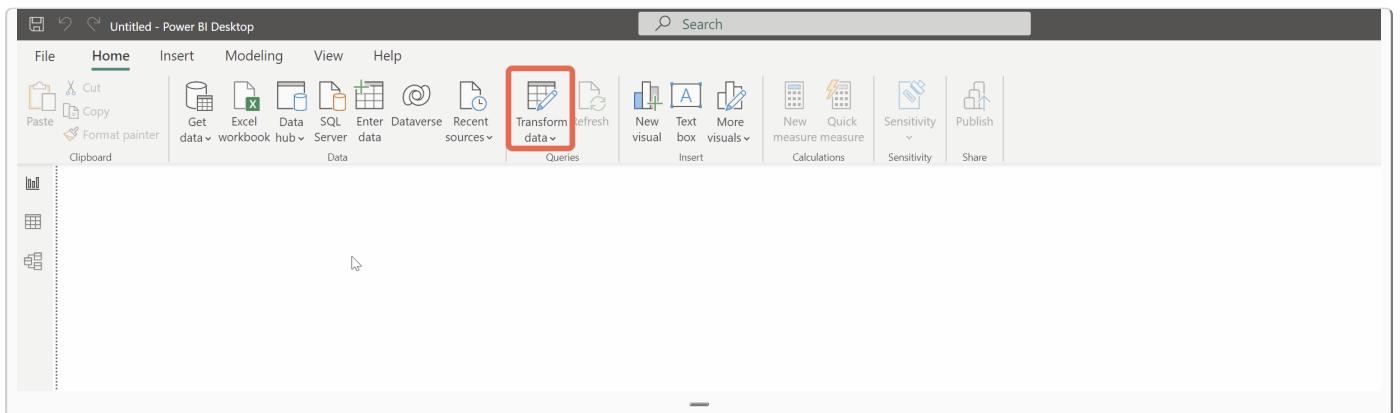
This section is relevant if users have a powerBI dashboard on their system and would like to integrate the dashboard in Foresta Apps.

Looking for tips and instructions on building the dashboards? please go [this section](#).

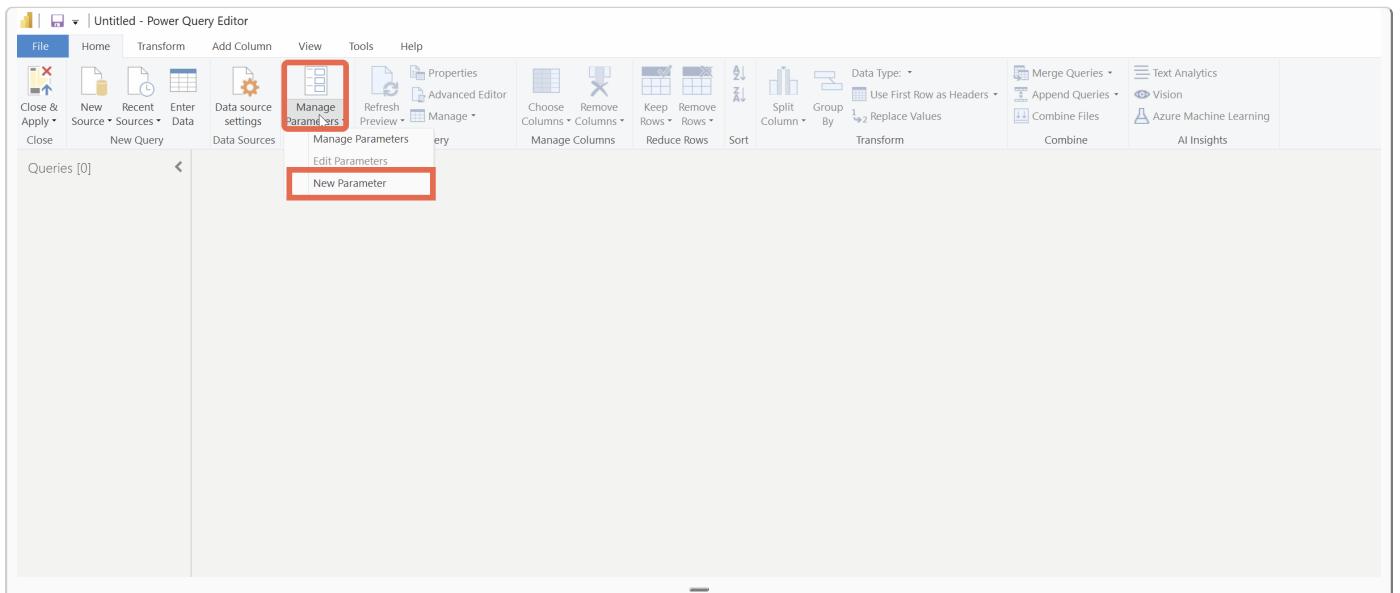
To integrate visualizations in Foresta, the users must parameterize their connection details. This ensures backend services of the app can seamlessly interact with the powerBI visualization.

### ESSENTIAL STEP: SET UP DATABASE CONNECTIVITY PARAMETERS

In PowerBI, click **Transform Data**



Click on **Manage Parameters** and then **New Parameter**



In the pop-up, Create a parameter with name - **Hostname**

## Manage Parameters

New

A <sup>B</sup> C Hostname	X
---------------------------	---

Name: Hostname

Description:

Required

Type: Text

Suggested Values: Any value

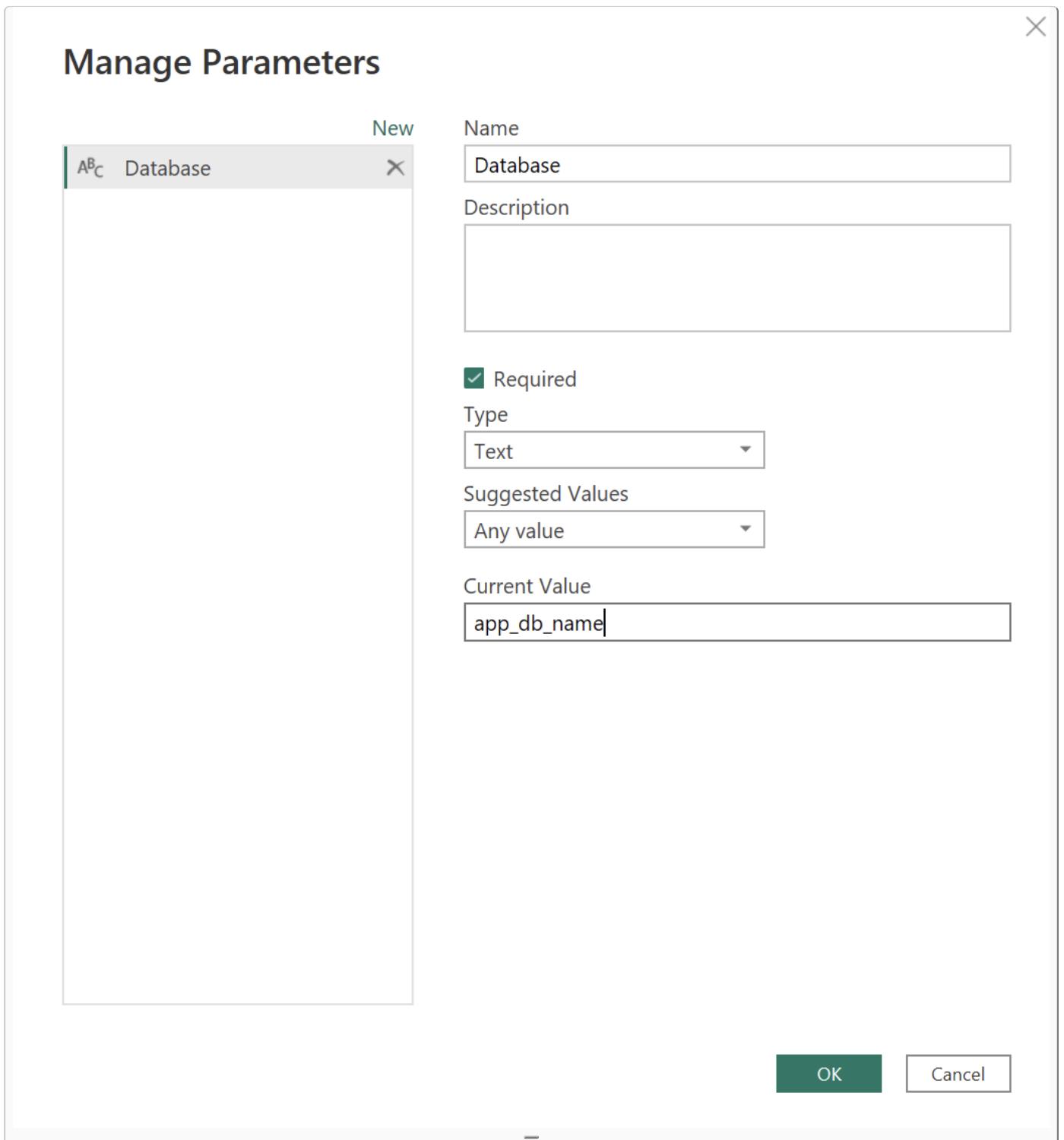
Current Value: foresta server url

OK Cancel



Please use the name string as indicated in the screenshot. Also, make sure the type is set to **Text**.

Add another parameter with name - **Database**



### ⚠️ Warning

Please use the name string as indicated in the screenshot. Also, make sure the type is set to **Text**.

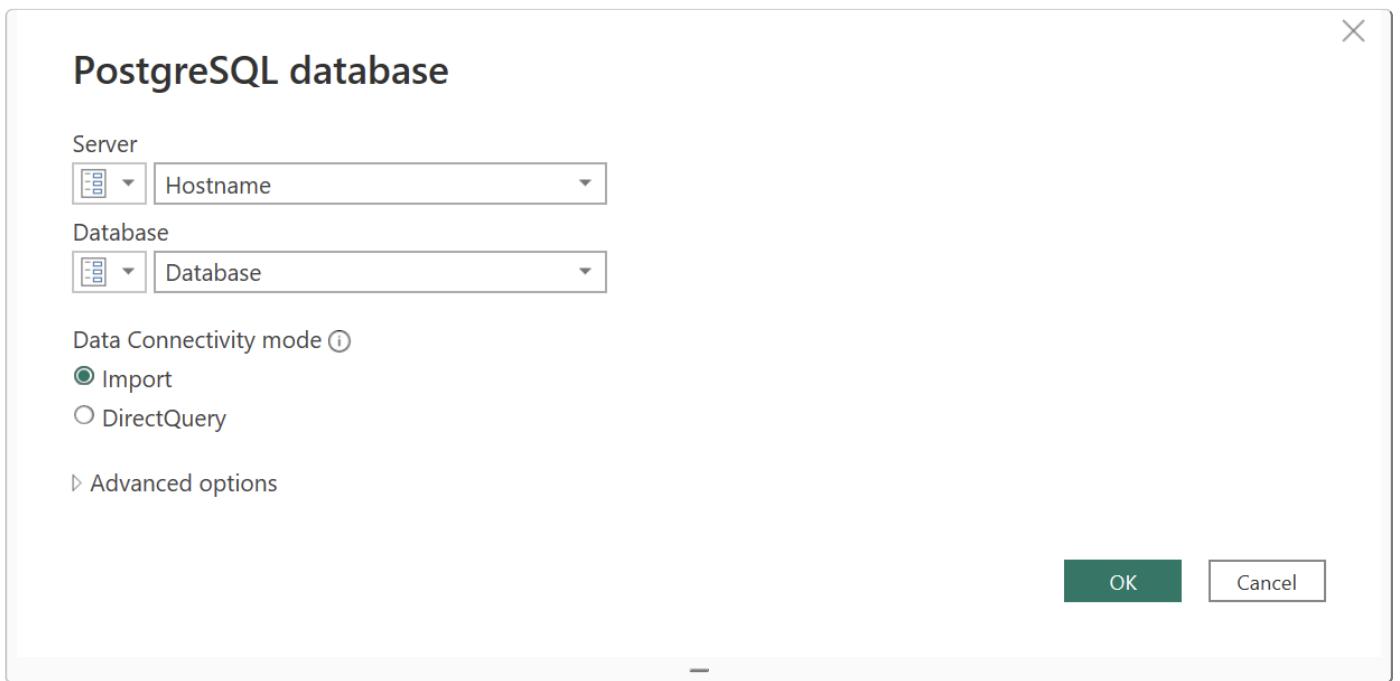
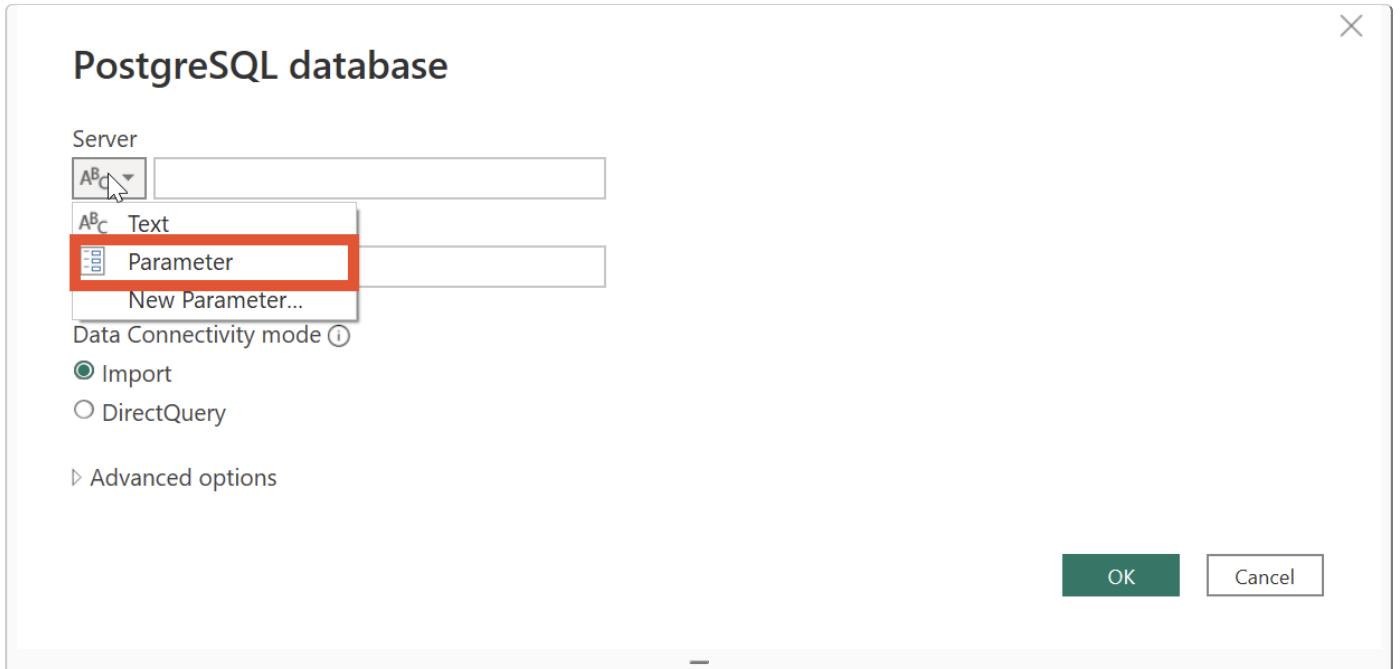
Now, Select **Data Source Settings** at the top ribbon.

The screenshot shows the Power Query Editor interface. The ribbon at the top has the 'Data Sources' tab highlighted with a red box. The main area displays a table with columns: #, site, product, time\_period, L2\_min\_supply, L2\_max\_supply, L2\_cost, L2\_rate, and tiered\_cost. The table contains data for various plants across different quarters and colors (Red, Gray, Clear). A query bar at the top shows the formula: = Table.AddColumn(#"Changed Type", "scenario\_id\_site\_product", each [rnd\_off\_scenario\_id\_text]&[site]&[product]). The bottom status bar indicates 14 COLUMNS, 199+ ROWS, and PREVIEW DOWNLOADED AT 12:33 PM.

Select the database and go to Change Source...

The screenshot shows the Power Query Editor with the 'Data Sources' ribbon tab selected. A 'Data source settings' dialog box is open over the main table area. The dialog box title is 'Data source settings' and it says 'Manage settings for data sources that you have connected to using Power BI Desktop.' It has tabs for 'Data sources in current file' (selected) and 'Global permissions'. A search bar contains 'foresta-qadecisionspot.com/app\_db\_000004\_9lz743m0'. At the bottom of the dialog box are buttons for 'Change Source...', 'Export PBIDS', 'Edit Permissions...', and 'Clear Permissions...'. The main table below the dialog box shows the same data as the previous screenshot. The bottom status bar indicates 14 COLUMNS, 199+ ROWS, and PREVIEW DOWNLOADED AT 12:33 PM.

Configure the PostgreSQL connection with parameters as shown below.

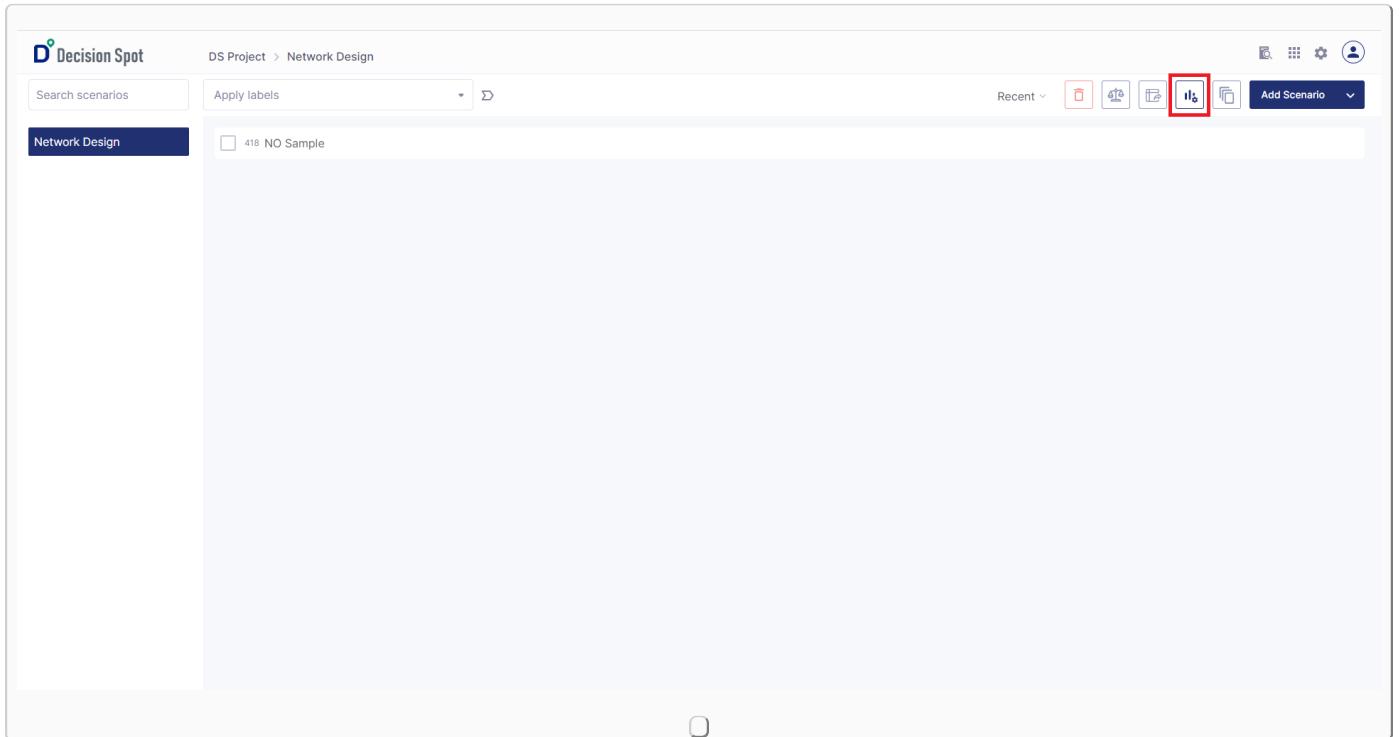


Now the reports are ready to be published using on the following methods.

#### MANUAL PUBLISHING METHODS:

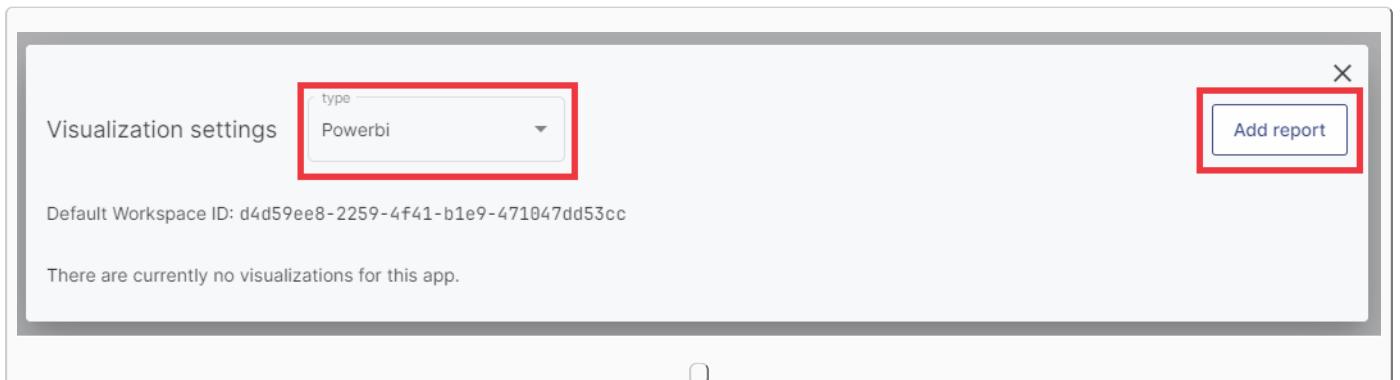
via. Upload

PowerBI (.pbix) files can uploaded directly into Foresta, as described ahead. Go to the Project page, select **Visualization Settings** at the top right corner.

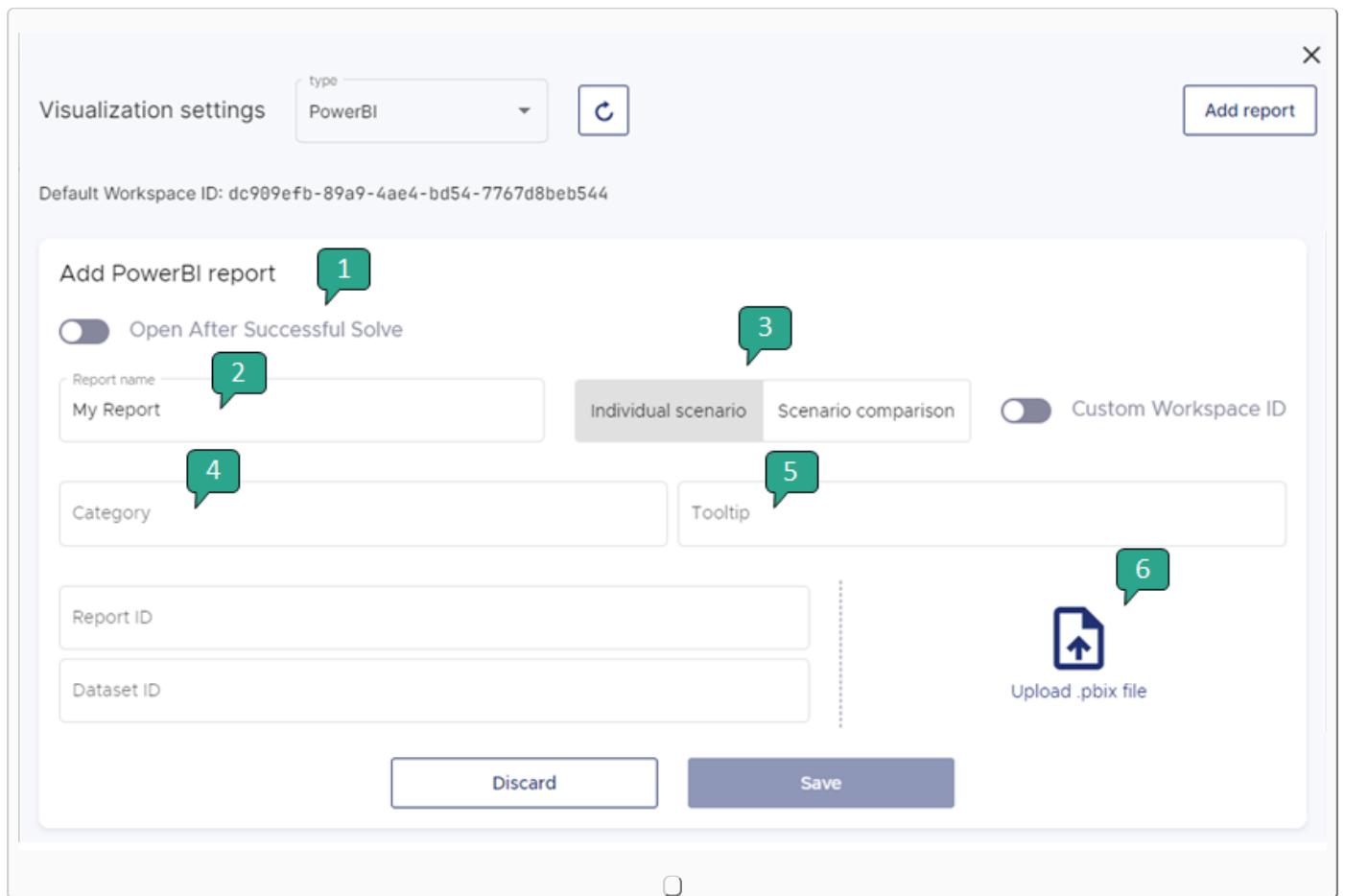


*Visualization Settings*

A pop-up will appear. In **type**, select PowerBI and click **Add report**.



*Visualization Settings*

*Visualization Settings*

1. Toggle if the visualization should automatically open after Solving the scenario (available only for Individual Scenario reports (3))
2. Set up the **Report name**.
3. Select if the report refers to an **Individual Scenario** or a **Scenario Comparison** analysis. See [here](#) for more details.
4. Inform the **Category** of the report (optional). The reports will be grouped following that category.
5. Write a **Tooltip** for the report (optional). The tooltip will be shown on hovering the report name.
6. **Upload** the PowerBI file (.pbix) here.

**Note**

Notice that users don't need to worry about publishing features here (Workspace ID, Report ID and Dataset ID), since they are uploading the file directly into the server.

After loading the file, hit the **Save** button.

Visualization settings

Add report

Open After Successful Solve

Report name: My Report  Individual scenario  Scenario comparison  Custom Workspace ID

Category  Tooltip

Report ID  Dataset ID

 NO Sample - Individual Scenario.pbix

Visualization Settings

If the upload was effective, it will be shown a new tab of the report with its informations.

**My Report**

Report ID	e44f0a30-8943-4d28-93e0-5a3019982eb2	<input type="radio" value="Individual scenario"/> Individual scenario
Dataset ID	49464433-bc4f-4f44-b8bc-01ba2f58860d	
Category	-	
Tooltip	-	

Visualization Settings

**Note**

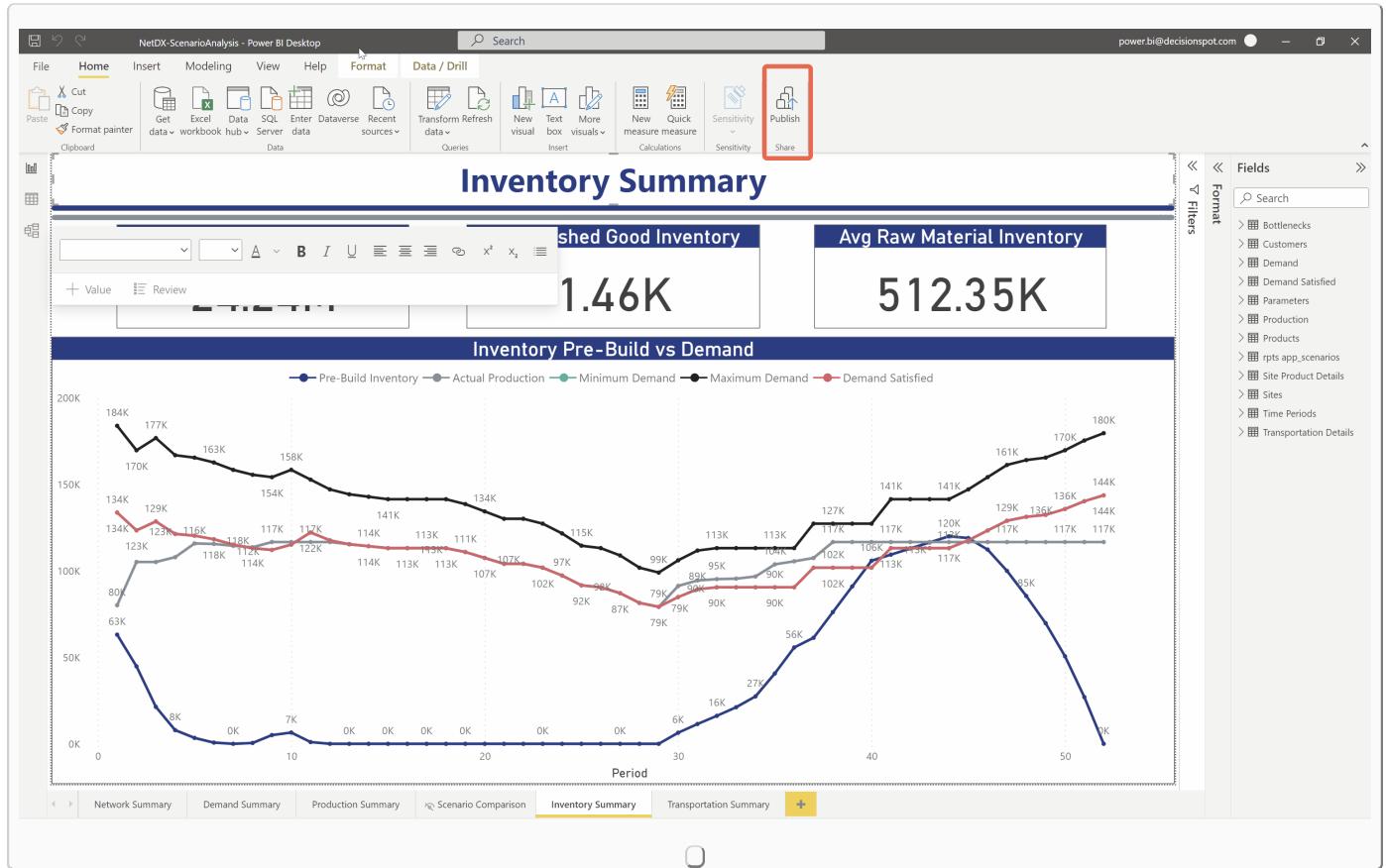
Alternatively, you can **Publish** your report manually into PowerBI server, and point these configurations to that server. See the [Manual Publishing Methods](#) to know more about it.

#### via. Linking the server

This is a 2 step process.

##### 1. Publish reports to PowerBI server

To publish the reports to the server, select the **Publish** option in **Home** menu in PowerBI.

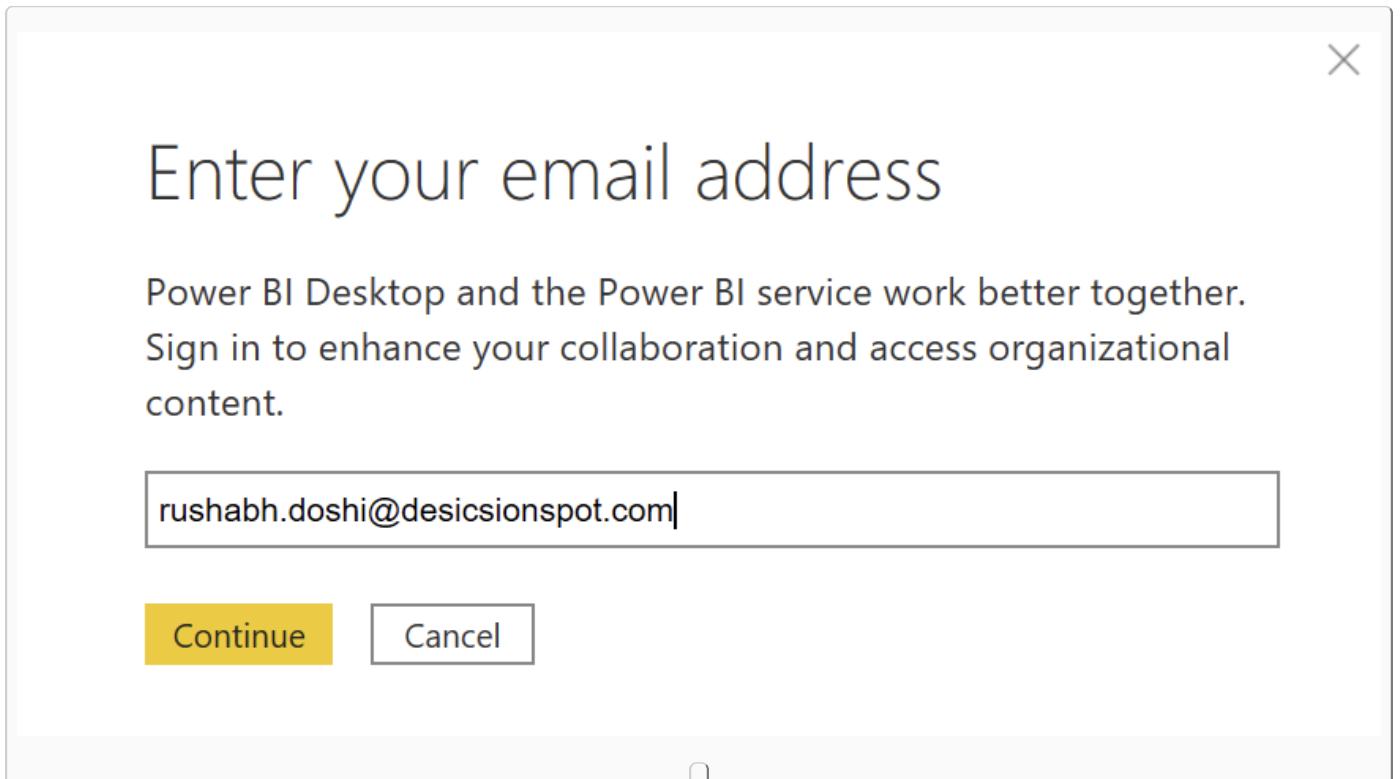


*PowerBI Publish*

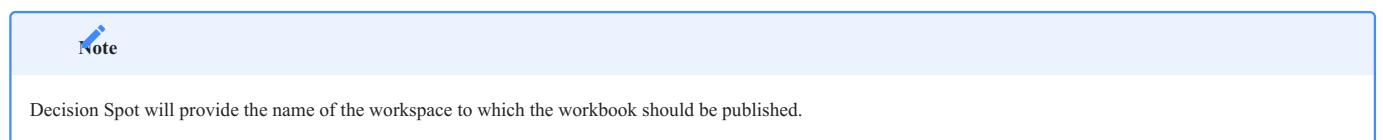
Enter the e-mail address of your PowerBI service account, and in the following dialog box enter the password to login to the server.

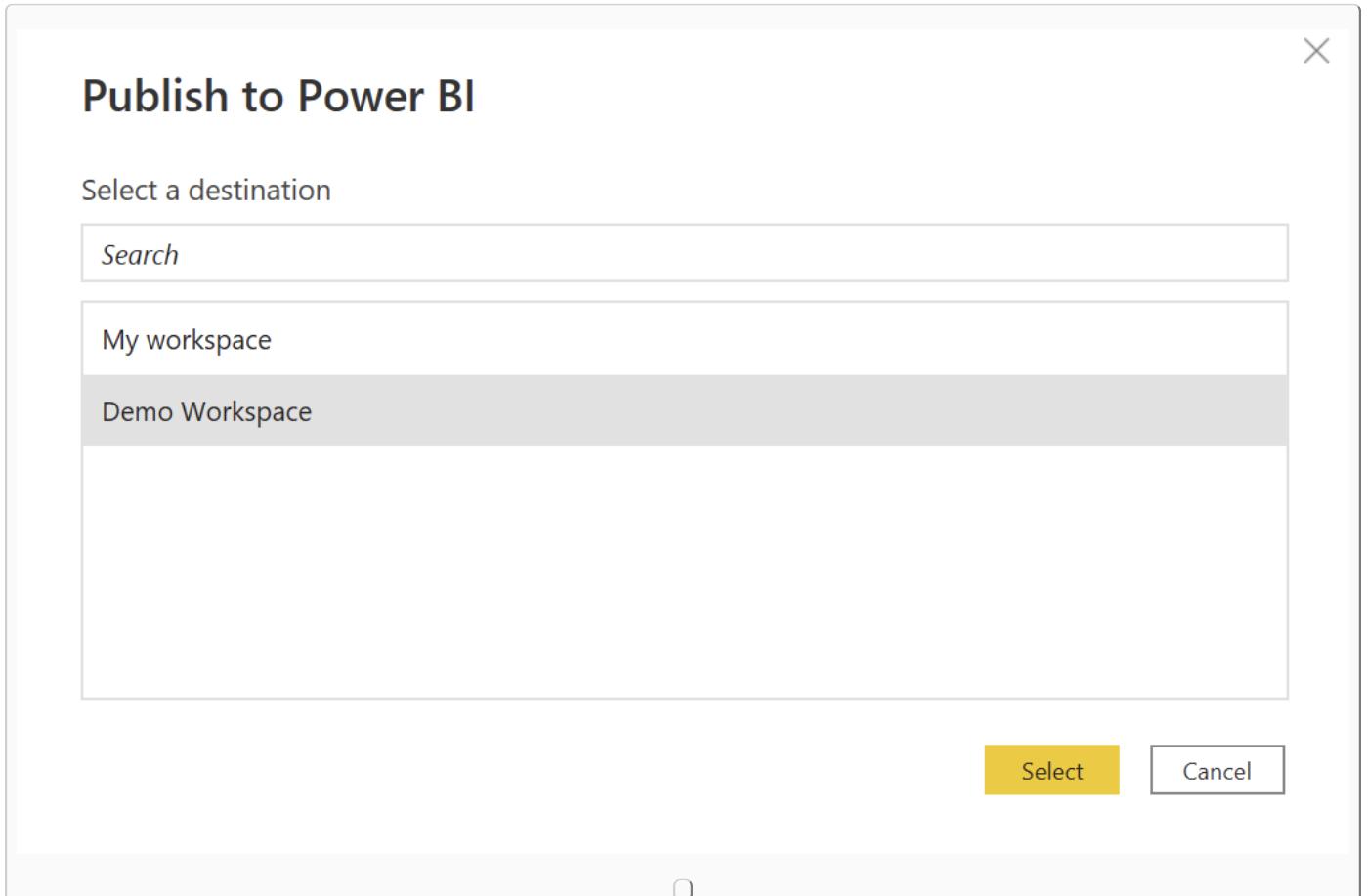


Please reach out to Decision Spot for setting up a PowerBI service account.

*PowerBI Service Login*

On the **Publish to PowerBI** dialog box, please select the Destination workspace to publish the workbook.





*PowerBI Publish Dialog Box*

## 2. Integrate PowerBI report details in Foresta application

After successfully publishing the workbook from desktop application, Login to PowerBI Web. Navigate to the workspace to which the report was published from the left hand navigation tree.

The screenshot shows the Power BI Home page. On the left, there's a sidebar with navigation links: Home, Create, Browse, Data hub, Metrics, Apps, Deployment pipelines, Learn, and Workspaces (which is currently selected and highlighted with a red box). Below the sidebar, there's a dropdown menu set to 'My workspace'. In the center, there's a search bar and a 'New report' button. A 'Workspaces' section is highlighted with a red box, containing a list of workspaces: 'Demo Workspace'. Below this, there are four cards: 'Popular in your org' (2nd\_approach\_params), 'You frequently open this' (FCLP), 'You frequently open this' (Demo\_DS), and 'You frequently open this' (TBC\_VRP\_viz\_). At the bottom, there's a table showing recent activity:

Type	Opened	Location	Endorsement	Sensitivity
Workspace	14 hours ago	Workspaces	—	—
Dataset	14 hours ago	Demo Workspace	—	—
Report	a day ago	Demo Workspace	—	—
Dataset	5 days ago	Demo Workspace	—	—
Report	5 days ago	Demo Workspace	—	—

**Create a workspace**

PowerBI Web Homepage

Open the dataset that was created for published report.

The screenshot shows the Power BI Demo Workspace interface. On the left, there's a sidebar with various options like Home, Create, Browse, Data hub, Metrics, Apps, Deployment pipelines, Learn, Workspaces, and the current workspace, Demo Workspace. The main area displays a list of items under 'All'. The columns are Name, Type, Owner, Refreshed, Next refresh, and Endorsement. A dataset named 'NetDX-ScenarioAnalysis' is selected and highlighted with a red border. Other datasets listed include FCLP, NetDX-ScenarioComp, TBC\_VRP\_viz\_db\_connected, TBC\_VRP\_viz\_db\_connected.pbix, Test NO Viz, and another entry for Test NO Viz.

Name	Type	Owner	Refreshed	Next refresh	Endorsement
FCLP	Dataset	Demo Workspace	11/1/23, 9:59:59 AM	N/A	—
NetDX-ScenarioAnalysis	Report	Demo Workspace	11/1/23, 9:53:02 AM	—	—
NetDX-ScenarioAnalysis	Dataset	Demo Workspace	11/1/23, 9:53:02 AM	N/A	—
NetDX-ScenarioComp	Report	Demo Workspace	8/12/22, 6:38:15 PM	—	—
NetDX-ScenarioComp	Dataset	Demo Workspace	8/12/22, 6:38:15 PM	N/A	—
TBC_VRP_viz_db_connected	Report	Demo Workspace	19/10/22, 6:11:34 PM	—	—
TBC_VRP_viz_db_connected	Dataset	Demo Workspace	19/10/22, 6:11:34 PM	N/A	—
TBC_VRP_viz_db_connected.pbix	Dashboard	Demo Workspace	—	—	—
Test NO Viz	Report	Demo Workspace	31/10/22, 11:25:36 PM	—	—
Test NO Viz	Dataset	Demo Workspace	31/10/22, 11:25:36 PM	N/A	—

#### *Demo Workspace Index*

Navigate to **File > Settings > Datasets > Data Source Credentials** and provide the **User Name** and **Password** for secure connection to the Foresta application database.

The screenshot shows the Power BI Dataset Homepage for the 'NetDX-ScenarioComp' dataset. The 'Settings' option in the context menu is highlighted with a red box. The homepage includes sections for visualizing and sharing the data, and a table of related reports.

**Dataset Homepage**

The screenshot shows the 'Dataset Settings' page for the 'NetDX-ScenarioComp' dataset. The 'Data source credentials' section is highlighted with a red box. It displays the connection information: 'app\_db\_000045\_1dk7vv8b-roundoff-demo.decisionspot.com'. There are also sections for 'Parameters', 'Query Caching', 'Scheduled refresh', and 'Server settings'.

**Dataset Settings**

Configure NetDX-ScenarioCo...

server  
roundoff-demo.decisionspot.com

database  
app\_db\_000045\_1dk7vv8b

Authentication method  
Basic

User name  
scno\_app

Password  
..... 

Privacy level setting for this data source  
Private

Dataset Credentials

After successful sign-in, navigate to the published workbook from Workspace homepage. Copy the URL of the page after the workbook loads on the screen.

The screenshot shows a Microsoft Edge browser window with a Power BI report titled "Network at a Glance". The report displays various metrics and a network map. The URL in the address bar is highlighted.

*Workbook View*

The URL of the page looks like this:

```
https://app.powerbi.com/groups/d4d59ee8-2259-4f41-b1e9-471047dd53cc/reports/9288289e-148f-4ffc-9e85-995044b95204/ReportSectionbcc2d0f4191299075108?experience=power-bi
```

The string after **https://app.powerbi.com/groups/** is the workspace ID of the PowerBI service.

For example, **Workspace ID** from this URL is `d4d59ee8-2259-4f41-b1e9-471047dd53cc`

The string after **reports/** is the report ID of the published report in PowerBI service.

For example, **Report ID** from this URL is `9288289e-148f-4ffc-9e85-995044b95204`

Navigate back to the Dataset homepage. Grab the URL of the Dataset homepage which looks like this:

The screenshot shows the Power BI Dataset details page. The URL in the browser's address bar is highlighted with a red box and reads: <https://app.powerbi.com/groups/648944df-fd54-40b8-985e-33f7840844d7/datasets/8>. The page displays dataset details, workspace information, and options to visualize or share the data.

Dataset URL

```
https://app.powerbi.com/groups/d4d59ee8-2259-4f41-b1e9-471047dd53cc/datasets/fea0d758-636e-4287-9158-50c51425dca0/details?experience=power-bi
```

The string after **datasets/** is the dataset ID of the published dataset in PowerBI service.

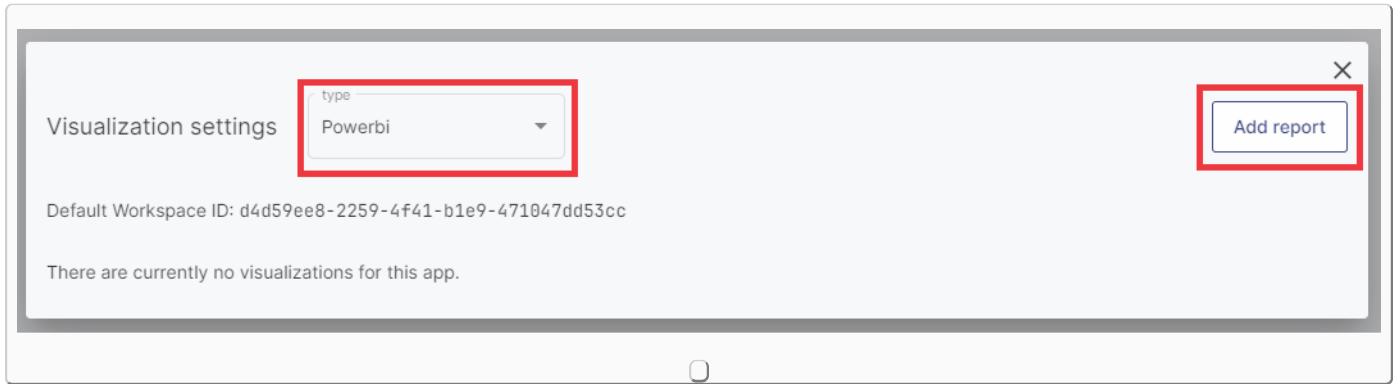
For example the **Dataset ID** from this URL is `fea0d758-636e-4287-9158-50c51425dca0`

The above three strings are required for embedding the report in the Foresta Application. Navigate to the Foresta Application home page, and click on the **Visualization Settings** button as shown in the screenshot below.

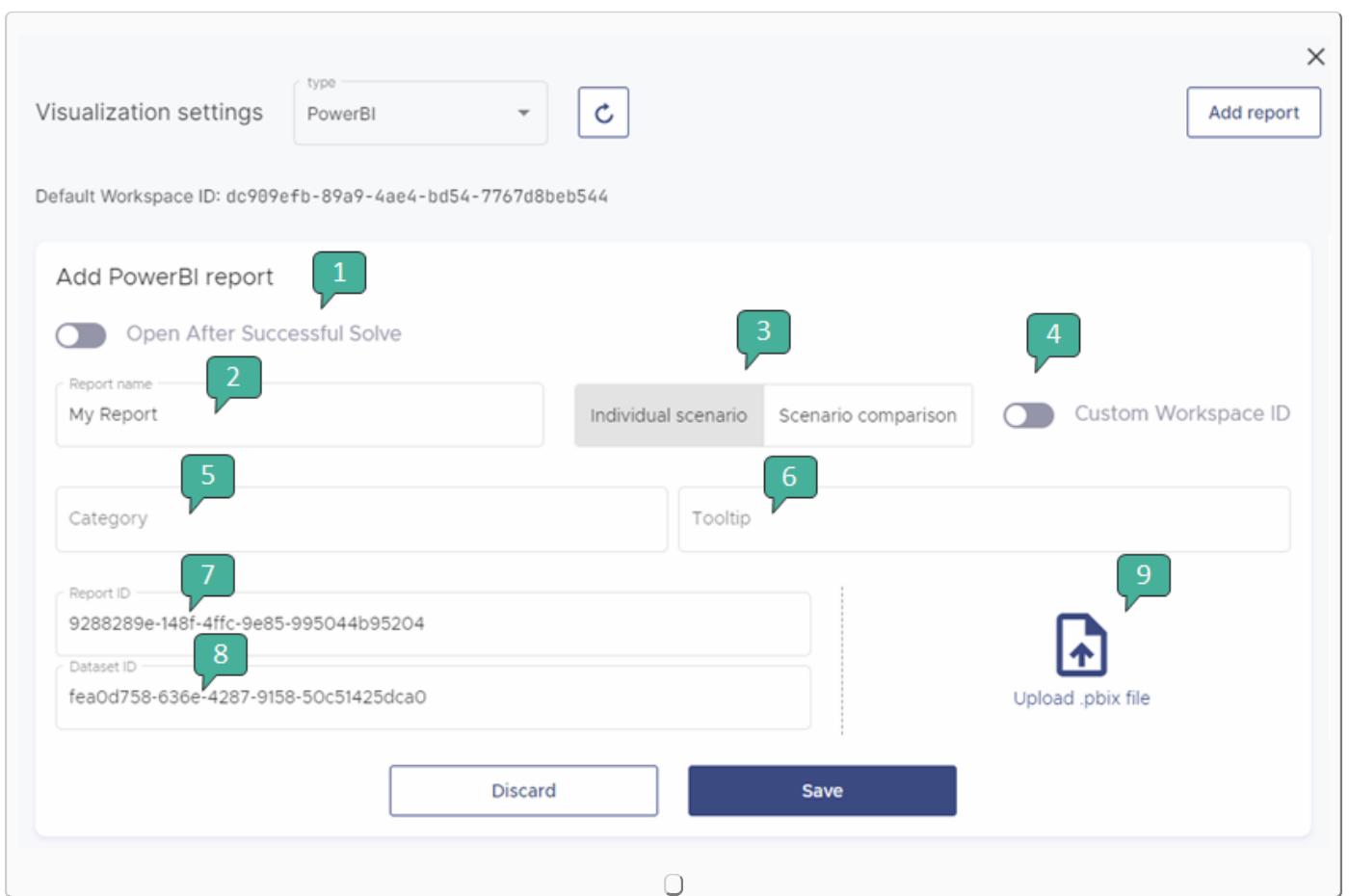
The screenshot shows the Foresta Decision Spot application interface. The top navigation bar includes 'Decision Spot' and 'DS Project > Network Design'. On the right side of the header, there is a toolbar with several icons, one of which is highlighted with a red box and labeled 'Visualization Settings'.

Foresta Visualization Settings Button

A pop-up will appear. At **type**, select Powerbi and select **Add report**.



*Visualization Settings*



*Visualization Settings*

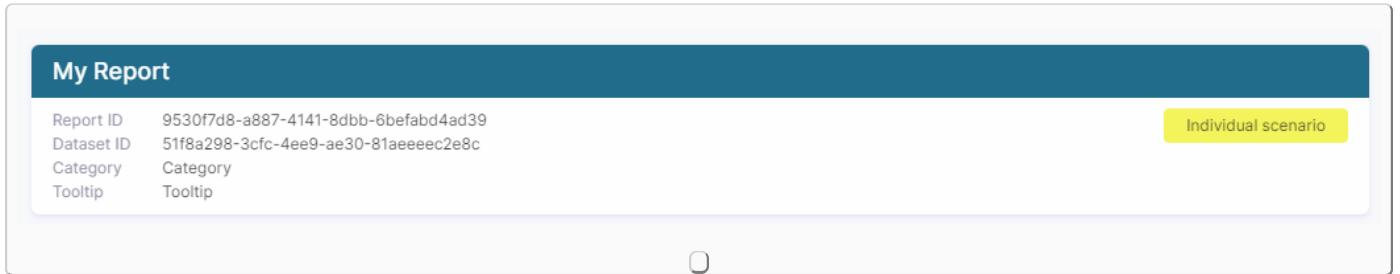
Follow the steps:

1. Toggle if the visualization should automatically open after Solving the scenario (available only for Individual Scenario reports (3))
2. Set up the **Report name**.
3. Select if the report refers to an **Individual Scenario** or a **Scenario Comparison** analysis (take a look at the end of the page to understand more about it).
4. If your report was published at the **Default Workspace**, leave the slider for **Custom Workspace ID** unchecked. If not, click on the slider and inform the **Workspace ID** of your report (the Default Workspace ID is shown just in the top left corner).
5. Inform the **Category** of the report (optional). The reports will be grouped following that category.
6. Write a **Tooltip** for the report (optional). The tooltip will be shown as you hover over the report name.
7. Fill the **Report ID**.
8. Fill the **Dataset ID**.

#### Note

Alternatively, you can **Upload** (9) your report manually instead of having to follow all these publishing steps. See the [Manual Publishing Methods](#) to learn more.

Click **Save** to embed the report into Foresta application. If the publish was effective, it will be shown as a new tab of the report with its information.



The screenshot shows a 'My Report' page with a dark blue header bar containing the title 'My Report'. Below the header, there is a table with the following data:

Report ID	9530f7d8-a887-4141-8dbb-6befabd4ad39	Individual scenario
Dataset ID	51f8a298-3fc-4ee9-ae30-81aeeeec2e8c	
Category	Category	
Tooltip	Tooltip	

Below the table, there is a small preview window showing a single visualization. At the bottom of the page, there is a section titled 'Visualization Settings' with a small icon.

#### AUTO-PUBLISH VISUALIZATIONS

This feature is available to the App-Builders. Find more details [here](#)

#### NAVIGATION FOR INTEGRATED VISUALIZATIONS

##### Scenario Comparison Report

If the reports are comparing results across 2 or multiple scenarios. Navigate to the Scenario Comparison Reports from the **Compare scenarios** button on application home page.

### Scenario Comparison Reports

#### Individual Scenario Report

If the report analyzes only results and KPIs from 1 scenario. Navigate to a **Scenario > Visualization** as shown in screenshot for individual scenario reports.

### Individual Scenario Reports

## 2.4.4 Tableau



Similar to PowerBI, Foresta has integrated visualization and reporting capability with embedded Tableau dashboards. Users can create and publish their own reports built from the Foresta application database with minimal configuration. The steps to connect reports can be listed as:

- Building Visualizations:
  - a. Assign user access to projects/applications for visualization
  - b. Connect to the Foresta application database from Tableau
  - c. Create the Data Model in Tableau
  - d. Create the dashboards
  - e. Setting up Connection Type
- Publishing Visualizations:
  - a. Publish the Data Source to Tableau server
  - b. Publish the Workbook to Tableau server
  - c. Integrate the Tableau report details in Foresta application

### **Building Visualizations**

#### 1. CREATE A DATABASE CONNECTION

Follow the steps provided [here](#) to get Database Name, Username and Password

#### 2. CONNECT TO THE FORESTA APPLICATION DATABASE FROM TABLEAU

Foresta application uses a PostgreSQL database. The first screen that you access when opening Tableau is the **Home** menu.

Connect to a PostgreSQL server under the **Connect** menu at the right pane.

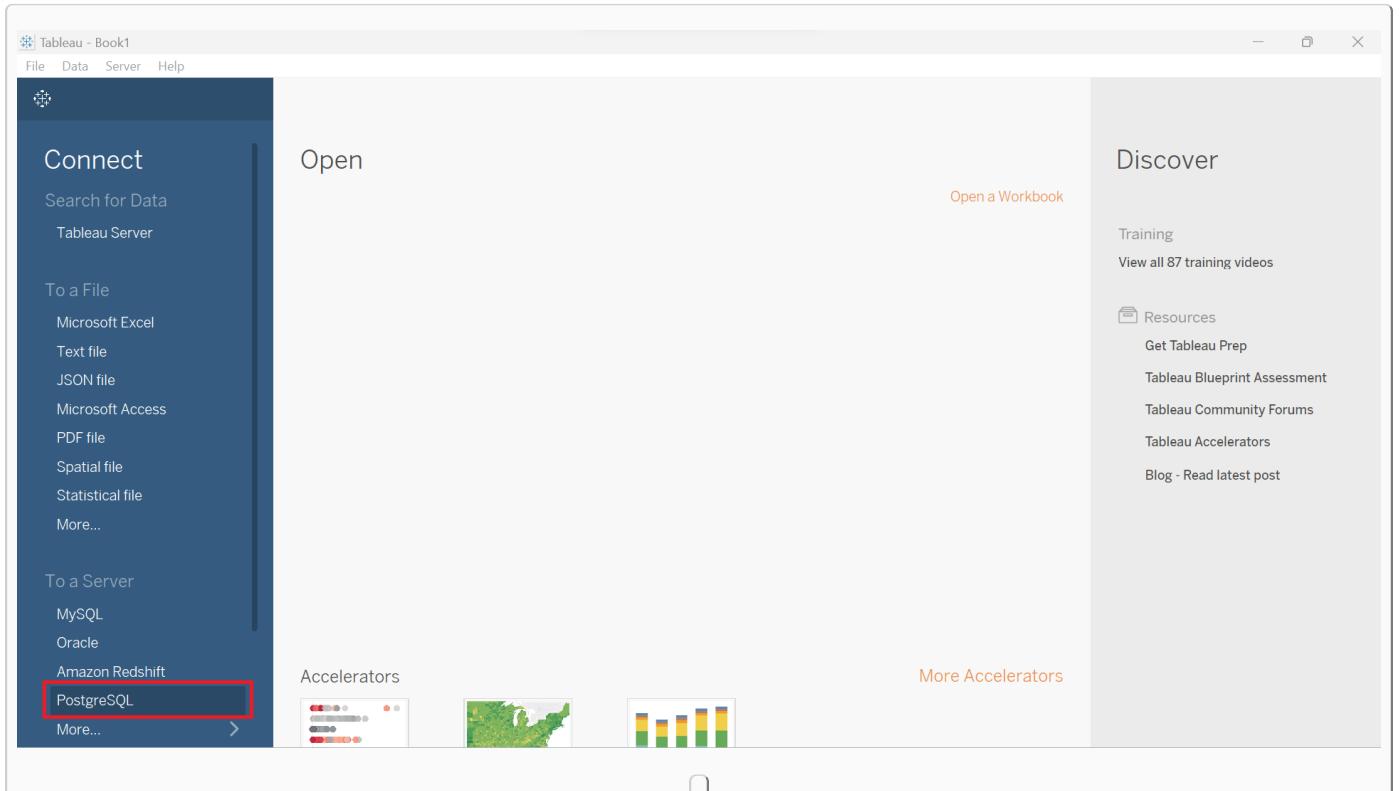


Tableau -> Connect -> To a Server

### Note

If PostgreSQL server is not on the right menu, select the last option **More** at the right pane (Connect > To a Server > More...) and search for PostgreSQL server.

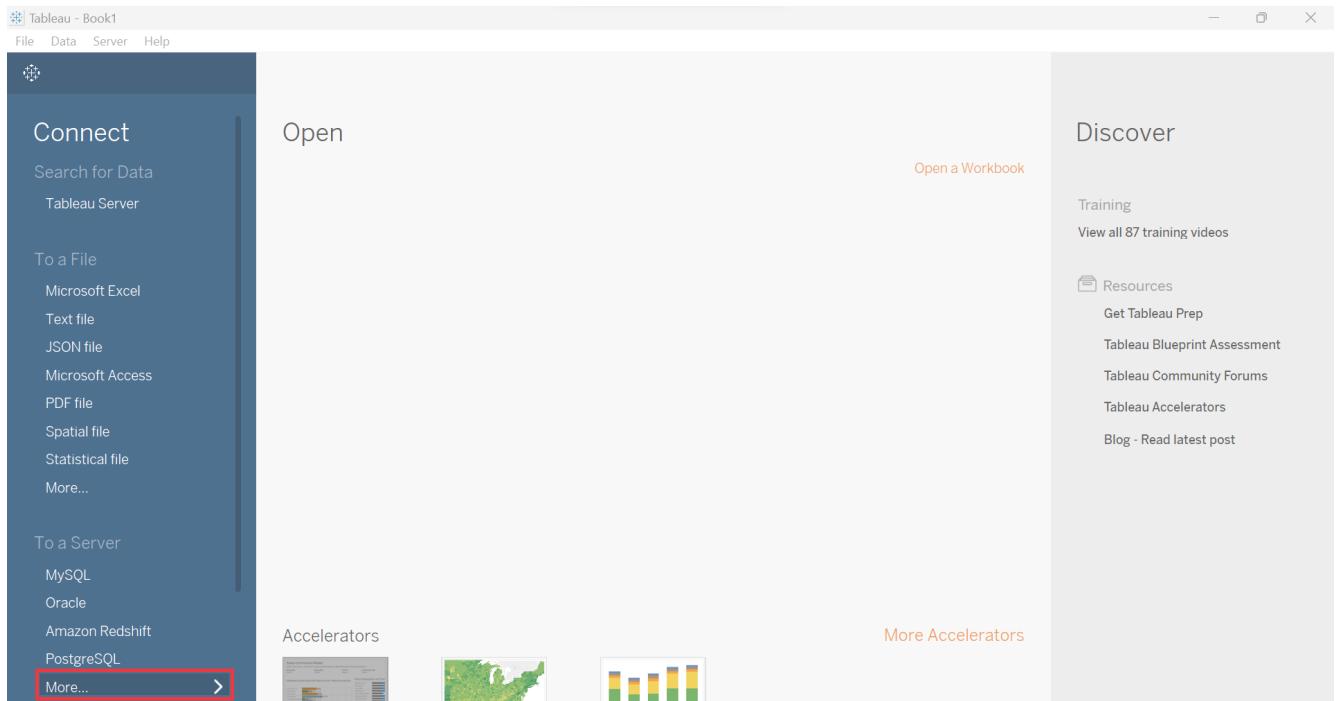
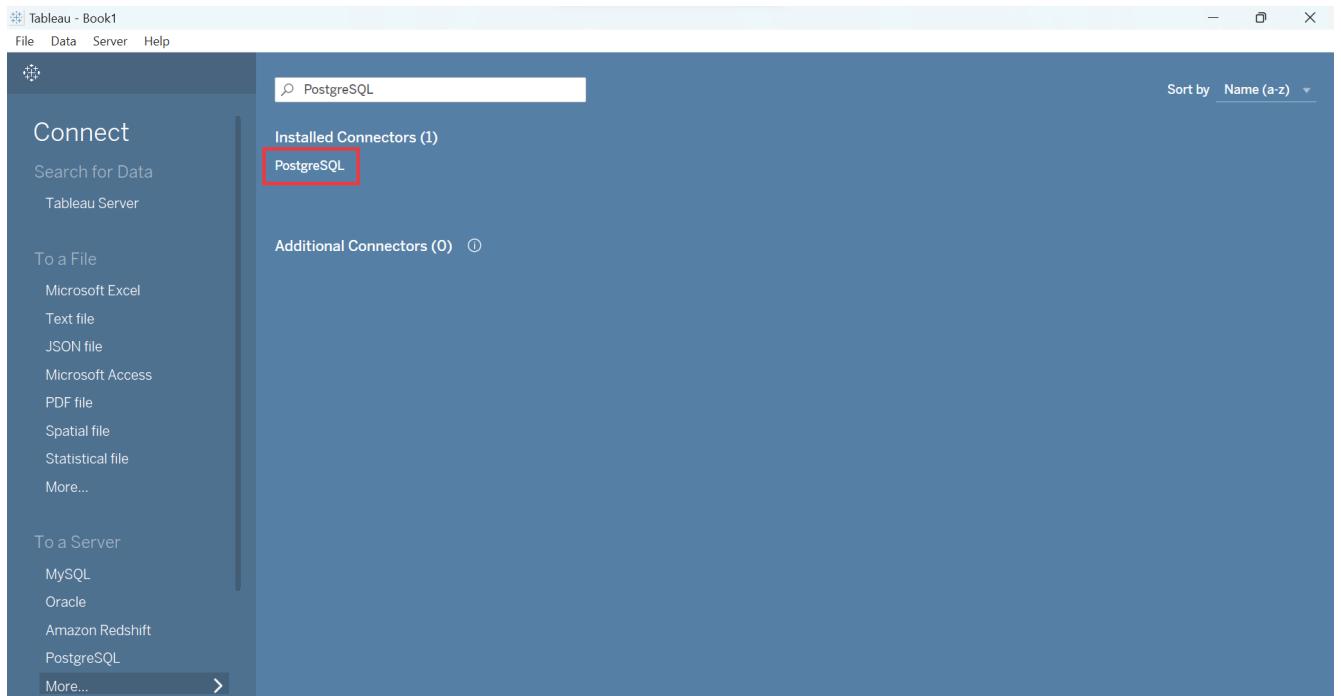
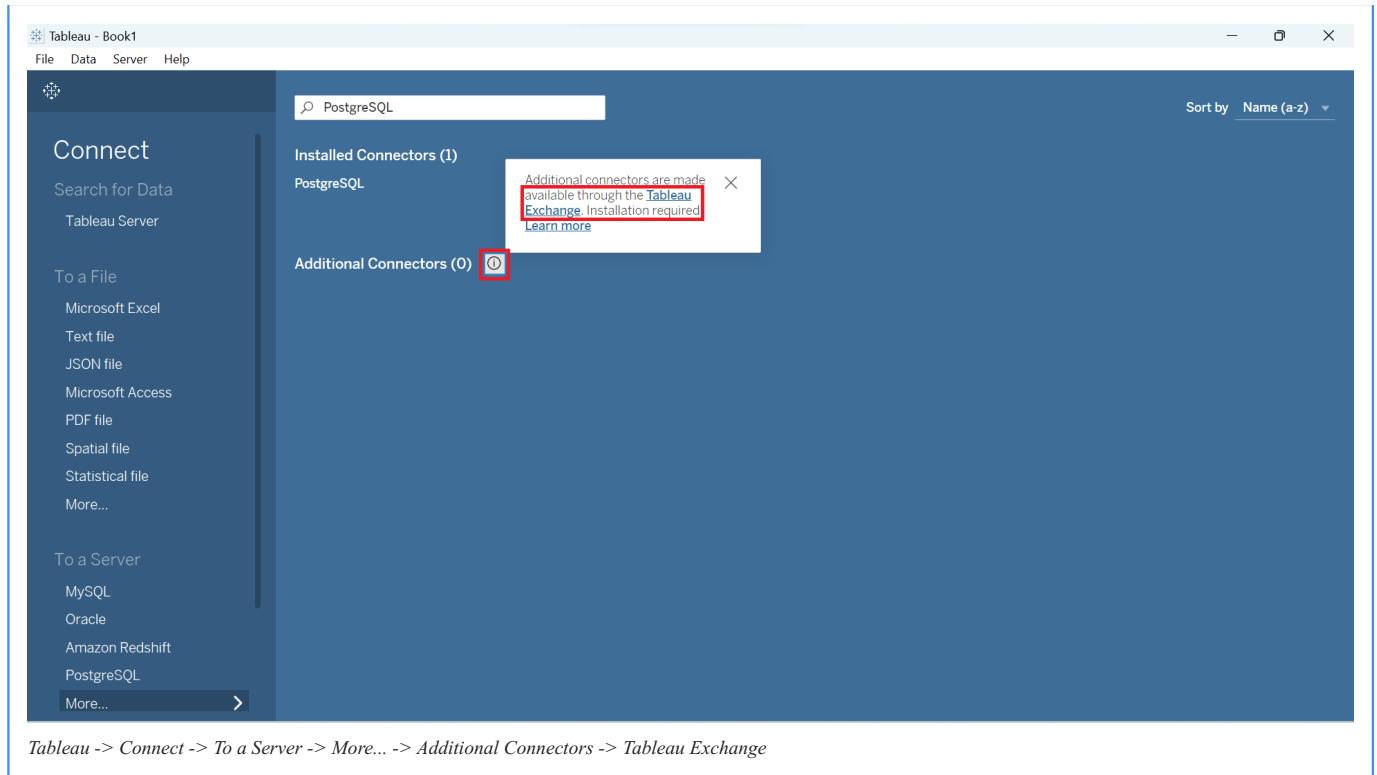


Tableau -> Connect -> To a Server -> More...



In case PostgreSQL connector is not installed, select the **Information** symbol just at the right of **Additional Connectors**, and proceed to the **Tableau Exchange** web page to install the PostgreSQL connector.



Fill the required details for **Host**, **Database**, **Username** and **Password** in the dialog box and click **Sign In**. This information must match what was configured in the Step 1.

- Server: Host
- Database: DB Name
- Username: User Name
- Password: Password

PostgreSQL X

General Initial SQL

---

Server  
foresta-demo.decisionspot.com

Port  
5432

Database  
app\_db\_name

Authentication  
Username and Password ▾

Username  
username

Password  
.....

Require SSL

Sign In

Tableau -> Connect -> To a Server -> PostgreSQL

### 3. CREATE THE DATA MODEL IN TABLEAU

Now that you are connected to the data source, Tableau will automatically redirect to the **Data Source** tab, at the bottom left corner. Here you can create your data model, by dragging required tables to the middle of the screen and creating relationships between them.

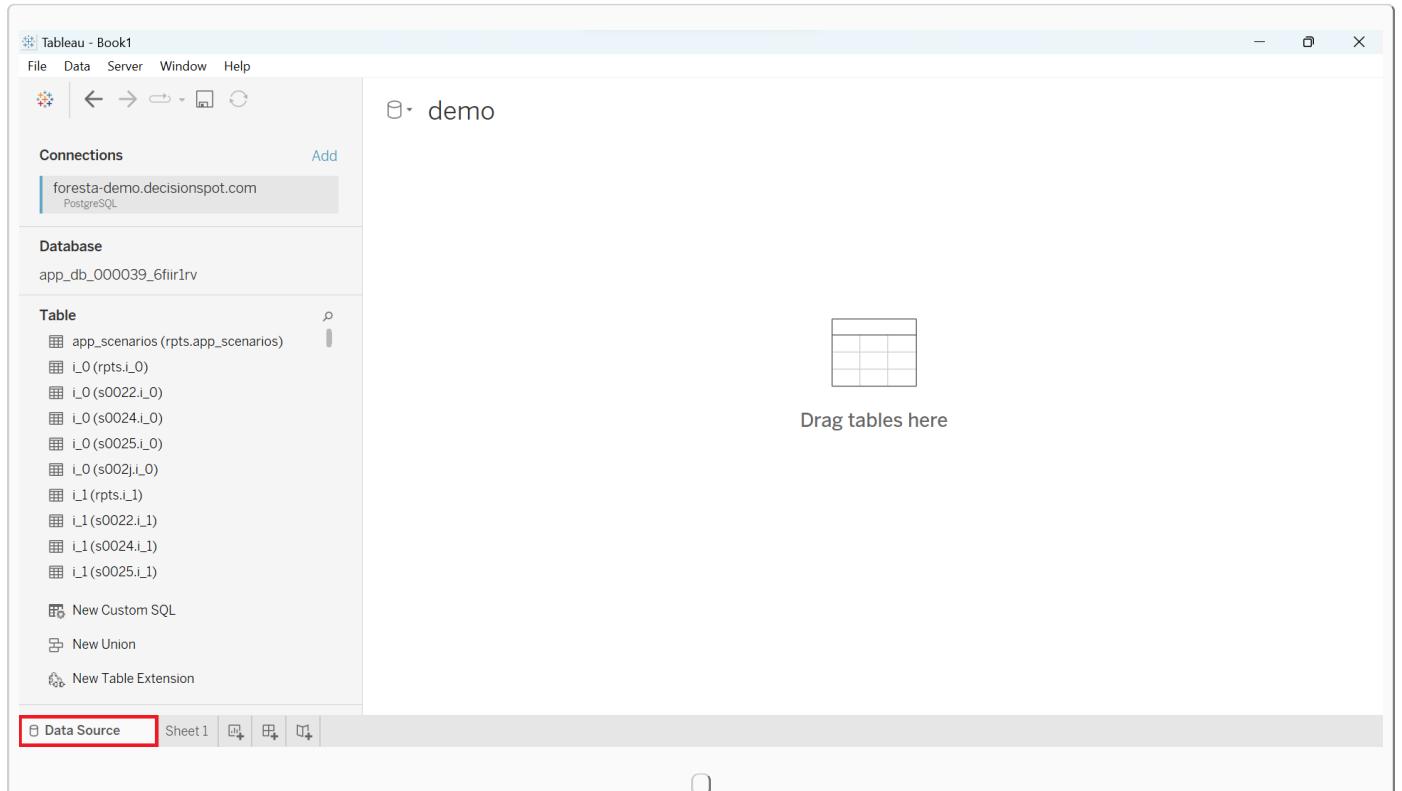


Tableau -&gt; Data Source



Please make sure to select the **rpts.app\_scenarios** table for automated scenario filters to work for **Individual Scenario reports**.

The screenshot shows the Tableau Data Source configuration interface. On the left, the 'Connections' pane lists a connection to 'foresta-demo.decisionspot.com' (PostgreSQL). The 'Database' pane shows the 'app\_db\_000039\_6fir1rv' database with a 'site' table selected. The main workspace displays a relationship diagram where 'app\_scenarios' is connected to 's\_site\_product\_details'. Below the diagram, a tooltip explains the difference between relationships and joins. A preview pane shows three columns: 'Site' (Abc), 'Product' (Abc), and 'Time Period' (Abc). At the bottom right, there are 'Update Now' and 'Update Automatically' buttons.

Tableau -&gt; Data Source

#### 4. CREATE THE DASHBOARDS

Create the required dashboards and configure the filters / slicers as per requirements.

#### 5. SETTING UP CONNECTION TYPE

Before publishing the report, there are two possible configurations that you can set that will dictate how the data will be loaded into your Tableau model. At the **Data Source** section (bottom left corner of the main menu), you can find the **Connection** configuration, at the upper-right corner of the screen:

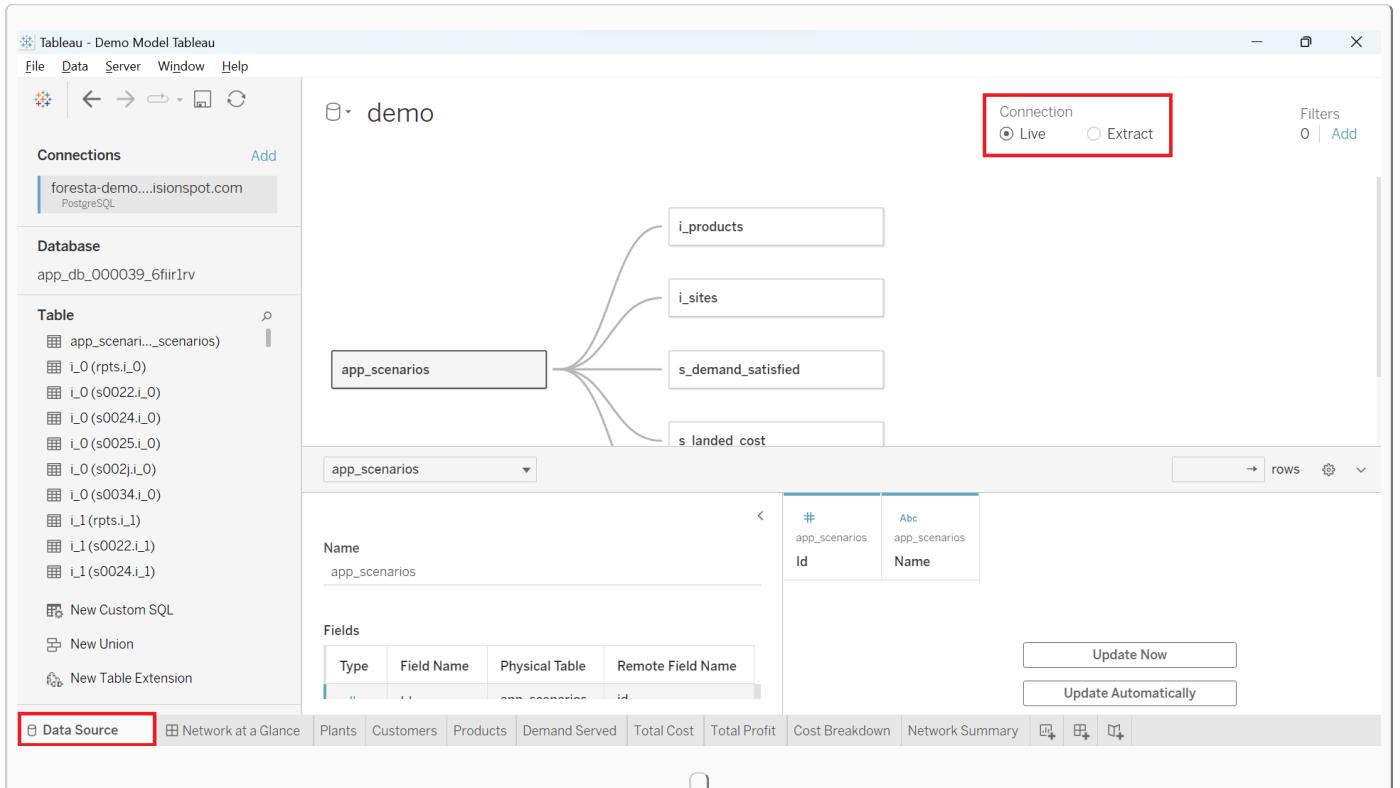


Tableau -&gt; Data Source -&gt; Connection

- **Live** - When you connect a data source with this configuration, your dashboard will query the data source immediately during runtime. Each filter and interaction with the report will result in a series of new queries. Since no data is imported into Tableau, the user can always query the data that already exists in the data source.
- **Extract** - Using the Extract mode of connection, Tableau will cache the data you're linked to, providing a snapshot of your data at a specific point in time. All of your data's interactions and filters will be applied to this compressed cache source rather than the original data source.

### Recommendation

**Extract** is the preferred mode of connection for Foresta applications since it gives better dashboard loading, interactivity and filter performance. Foresta has a built-in automated trigger to refresh the **Extract** dataset on Tableau server at the end of every successful scenario solve.

By default Tableau selects the Live connection, but as described above, we *HIGHLY* recommend an **Extract** connection, for performance purposes. When changing the configuration from Live to Extract, probably Tableau will ask you to save the extraction file in a pre-defined path (.../My Tableau Repository/Data Source). Proceed selecting the **Save** option.

### Warning

The **Connection Type** will dictate how you will proceed on the next steps. So keep in mind that was the connection type defined before proceeding to the next section.

## Publishing Visualizations

### 1. PUBLISH THE DATA SOURCE TO TABLEAU SERVER

Before anything, you only should follow this section if your **Connection Type** was set as **Extract**, if the connection was set as **Live** you can proceed to the next section [Publish the Workbook to Tableau server](#)

With the extract connection set up, **publish the data source**. At the main menu, select **Server** and **Publish Data Source**.

The screenshot shows the Tableau interface with the 'Server' menu highlighted in red. A sub-menu is open under 'Server' with the option 'Publish Data Source' also highlighted in red. The main dashboard view includes a summary card with counts for Plants (2), Customers (193), and Prod (1), and a total profit of 16,758,666. Below this are two visualizations: a bar chart titled 'Cost Breakdown' showing various cost components like Inventory Holding Cost, Miscellaneous Fixed Cost, etc., and a map titled 'Network Summary' showing locations across North America.

Tableau -&gt; Server -&gt; Publish Data Source

Your data source will appear just at the right as an option, select it and a pop-up will appear. If you configured the connection as Extract, as we recommended, the pop-up will be displayed as below. It will be necessary to configure two features, **Authentication** and **More Options**.

## Publish Data Source to Tableau Cloud X

Name  
demo

Description

Tags  
Add

Permissions  
Same as project (**Default**) [Edit](#)

Authentication  
Refresh not enabled [Edit](#)

Tableau Bridge required for on-premises data  
If Tableau Cloud can't connect directly to this data source, it will use a Tableau Bridge client to keep this data fresh.

More Options  
 Update workbook to use the published data source

Tableau Cloud will temporarily access the credentials provided for  
① '[foresta-dev.decisionspot.com]' to confirm it can maintain a live data connection.

 **Workbook Optimizer** Publish

© <a href="https://decisionspot.com" target="\_blank" rel="noopener">Copyright Decision Spot, LLC | All Rights Reserved</a>

Tableau -> Server -> Publish Data Source

At **Authentication**, select **Edit** and change the authentication configuration to **Allow refresh access**.

**Publish Data Source to Tableau Cloud**

Name: demo

Description:

Tags: Add

Permissions: Same as project (**Default**) [Edit](#)

Authentication: Refresh not enabled [Edit](#)

Data Source	Connection	Authentication	Extract	Username
demo	foresta-demo.decisionspot.com	Refresh not enabled Refresh not enabled <b>Allow refresh access</b>	Yes	lourencods

Update workbook to use the published data source

Tableau Cloud will temporarily access the credentials provided for  
 ⓘ '[foresta-dev.decisionspot.com]' to confirm it can maintain a live data connection.

[Workbook Optimizer](#) [Publish](#)

Tableau -> Server -> Publish Data Source -> Authentication

At the **More Options** section, make sure that the box **Update workbook to use the published data source** stay **UNCHECKED**.

## Publish Data Source to Tableau Cloud

**Name**  
demo

**Description**

**Tags**  
Add

**Permissions**  
Same as project (**Default**) [Edit](#)

**Authentication**

**Allow refresh access** [Edit](#)

Tableau Bridge required for on-premises data  
If Tableau Cloud can't connect directly to this data source, it will use a Tableau Bridge client to keep this data fresh.

[More Options](#)

**Update workbook to use the published data source**

**i** Tableau Cloud will temporarily access the credentials provided for '[foresta-dev.decisionspot.com]' to confirm it can maintain a live data connection.  
Allowing refresh access embeds credentials for that connection.

 **Workbook Optimizer**

**Publish**

© <a href="http://decisionspot.com" target="\_blank" rel="noopener">Copyright Decision Spot, LLC | All Rights Reserved</a>

## Tableau -&gt; Server -&gt; Publish Data Source

Select the option **Publish**, and a web page will automatically be opened informing that the publishing was completed.

The screenshot shows the Tableau Server interface. On the left, there's a sidebar with various icons. In the center, a data source named "demo" is listed under "Explore / default / demo". The data source is described as a "Tableau Viz" and was last updated on "1 Jun 2023, 15:41". Below the data source, there are buttons for "New", "Edit Data Source", "Ask Data", "Connections 1", and "Extract Refresh". A search bar at the top right says "Search for views, metrics, workbooks and more". A "Publishing Complete" dialog box is open in the center, stating "demo has been published to the server." It also suggests setting up a schedule for refreshes and includes a "Schedule" button. Below the dialog, there's a section titled "Create a Lens to Use Ask Data" with a "Create New Lens" button.

*Data Source publishing completed*

## 2. PUBLISH THE WORKBOOK TO TABLEAU SERVER

Now you need to **publish the report into the server**. At Tableau main menu, look for the **Server** section at the top bar and select the option **Publish Workbook**.

The screenshot shows the Tableau interface with the 'Server' menu highlighted and the 'Publish Workbook...' option selected. The main dashboard displays various metrics and visualizations, including a summary card, a bar chart for 'Cost Breakdown', and a map for 'Network Summary'.

**Server Menu Options:**

- Signed In to https://prod-useast-b.online.tableau.com (Decision Spot - Foresta)
- Run Optimizer...
- Open Workbook...
- Publish Workbook...** (highlighted with a red box)
- Publish Data Source
- Create User Filter
- Install Tableau Bridge Client...
- Tableau Public

**Dashboard Metrics:**

- Plants: 2
- Customers: 193
- Prod: 1
- Total Profit: 16,758,666

**Cost Breakdown Bar Chart:**

Category	Value
Inventory Holding Cost (\$ Landed Cost)	~8M
Miscellaneous Fixed Cost	~7M
Open Close Operating Cost	~6M
Production Cost	~14M
Site Product Cost	~1M
Transportation Cost	~12M

**Network Summary Map:**

A map of North America showing numerous blue dots representing locations, with two red dots highlighting specific points. Labels include 'United States' and 'Mexico'.

Tableau -> Server -> Publish Workbook...

A pop-up will appear. Configure most of the fields as desired and go to the **Data Sources** section. Click on **Edit**.

## Publish Workbook to Tableau Cloud

Location  
Default

Name  
Demo Model Tableau

Description

Tags  
Add

Sheets  
1 of 9 selected [Edit](#)

Permissions  
Same as project (**Default**) [Edit](#)

Data Sources

1 embedded in workbook [Edit](#)

More Options

Show sheets as tabs

Show selections

 **Workbook Optimizer**

**Publish**

Tableau -> Server -> Publish Workbook... -> Data Sources -> Edit

If your **Connection Type** was set as **Extract**, configure the Data Source as below (Publish Type = Embedded in workbook, Authentication = Allow refresh access).

Manage Data Sources

Data Source

demo

Tableau Cloud will temporarily access the credentials provided for '[forestad-dev.decisionspot.com]' to confirm it can maintain a live data connection. Allowing refresh access embeds credentials for that connection.

Publish Type ⓘ

Embedded in workbook ▾

Authentication

Allow refresh access ▾

Tableau -> Server -> Publish Workbook... -> Data Sources -> Edit

But if your **Connection Type** was set as **Live**, configure the Data Source as below (Publish Type = Embedded in workbook, Authentication = Embedded password).

Manage Data Sources

Data Source

app\_scenarios (rpts.app\_s...)

Tableau Cloud will temporarily access the credentials provided for '[forestad-dev.decisionspot.com]' to confirm it can maintain a live data connection.

Publish Type ⓘ

Embedded in workbook ▾

Authentication

Embedded password ▾

Tableau -> Server -> Publish Workbook... -> Data Sources -> Edit

Go back to the **Publish Workbook to Tableau Cloud** pop-up and click **Publish**.

## Publish Workbook to Tableau Cloud X

Location  
Default

Name  
Demo Model Tableau

Description

Tags  
Add

Sheets  
All Edit

Permissions  
Same as project (**Default**) [Edit](#)

Data Sources  
1 embedded in workbook [Edit](#)

More Options

Show sheets as tabs  
 Show selections

 **Workbook Optimizer**

Publish

Tableau -> Server -> Publish Workbook... -> Publish Workbook to Tableau Cloud

A web page will be opened confirming that the publishing was completed.

The screenshot shows the Tableau Cloud interface. On the left, there's a sidebar with various icons. In the center, there's a search bar and a main content area showing a workbook titled "Demo Model Tableau". The workbook has one view named "Network at a Glance" which displays some data and charts. A modal window titled "Publishing Complete" is overlaid on the page. It contains the message "Publishing Complete" and "Demo Model Tableau". Below that, it says "Data Sources" and "Data extracts can be scheduled to refresh regularly. Schedule data refresh". There are also links for "Preview different device layouts" and "Share this workbook". The overall interface is clean and modern, typical of a cloud-based application.

*Publishing completed*

### 3. INTEGRATE THE TABLEAU REPORT DETAILS IN FORESTA APPLICATION

After successfully publishing the workbook from desktop application, login to [Tableau Cloud](#) (or continue from the web page that was opened in the moment that you published the workbook).

Go to the address where the report was published. If not changed, Tableau probably saved on the path Explore > default.

This screenshot shows the "Network at a Glance" report details page in Tableau Cloud. The top navigation bar shows "Explore / default / Demo Model Tableau". The main content area displays the report's details: "Workbook: Demo Model Tableau", "Location: default", "Owner: Tableau Viz", and "Modified: 1 Jun 2023, 16:55". Below this, the report's visualizations are shown, including a summary table and two charts. At the bottom of the page, the URL "https://prod-useast-b.online.tableau.com/#/site/decisionspotforesta/redirect..." is visible. The interface is consistent with the previous screenshot, showing the same sidebar and search bar.

*Report address on Tableau Cloud*

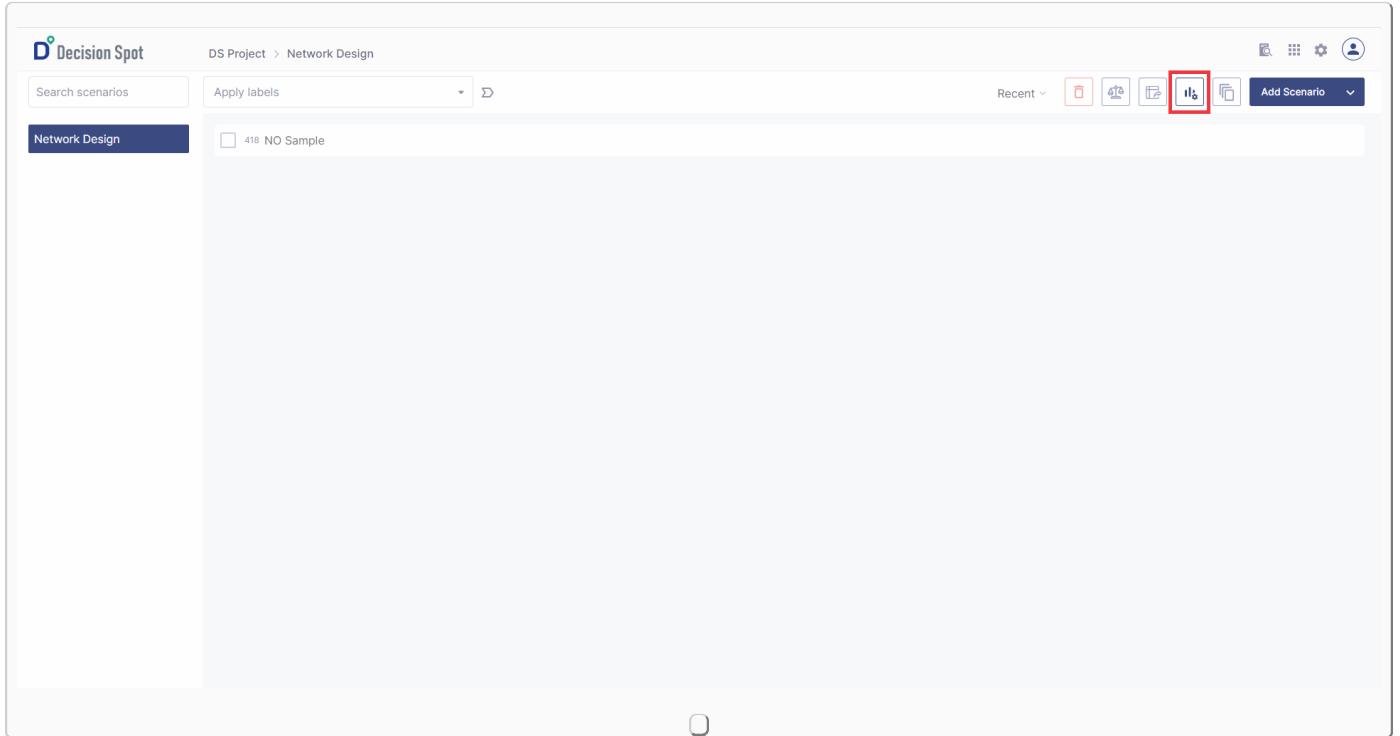
Select the report and, at the right top corner, select **Share**.

*Report opened at Tableau Cloud*

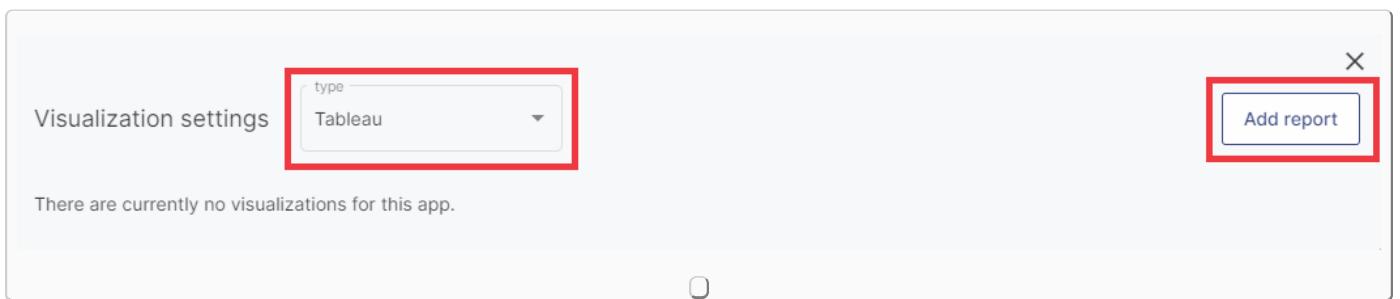
A pop-up named **Share View** will appear at the middle of the screen. At the bottom right corner, select **Copy Link** to copy the URL containing the address of the visualization.

*Pop-up with the visualization link*

Navigate to the Foresta Application home page, and click on the **Visualization Settings** button, at the top right corner, as shown in the screenshot below.

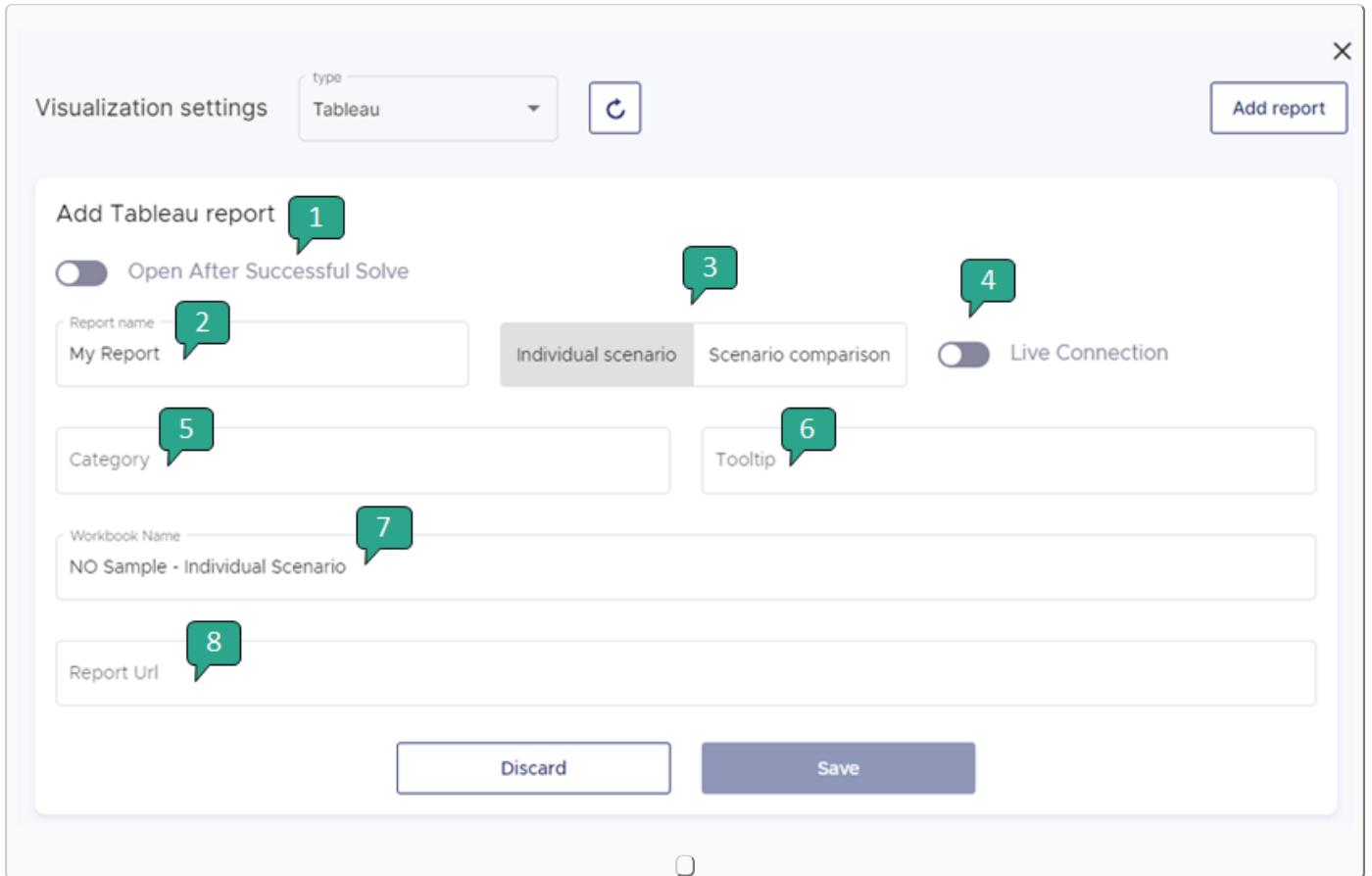
*Foresta Visualization Settings Button*

At the **Visualization Settings** pop-up, select **Tableau** option at the right corner and click **Add report**.

*Foresta Visualization Settings Pop-up*

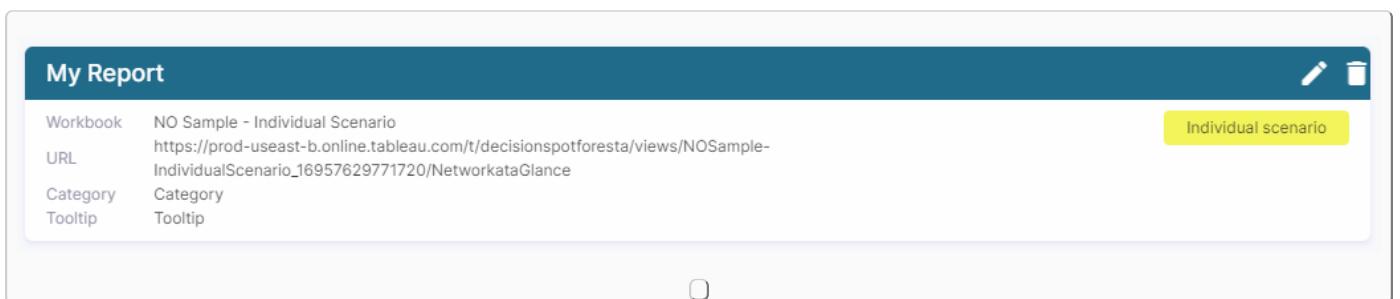
The pop-up will expand. Follow the steps:

1. Define if you the visualization will automatically open after Solving the scenario (available only for Individual Scenario reports (3)).
2. Give a name to the Report.
3. Select if the report refers to an **Individual Scenario** or a **Scenario Comparison** analysis (take a look at the end of the page to understand more about it).
4. Active **Live Connection** toggle if your **Connection Type** was set as Live.
5. Inform the **Category** of the report (optional). The reports will be grouped following that category.
6. Write a **Tooltip** for the report (optional). The tooltip will be shown as you hover over the report name.
7. Set up the **Workbook name**. Needs to be the **EXACT** name of the published workbook.
8. Paste the link copied at the **Report Url** field.



Foresta Visualization Settings Menu for Tableau

Click the **Save** button to create the reports at Foresta Application. The pop-up will change with the new report embedded.



Foresta Visualization Settings Menu

- 1. Scenario Comparison Report:** If the reports are comparing results across 2 or multiple scenarios, use the *Scenario comparison report* section of the *Visualization Settings* configuration.

Navigate to the Scenario Comparison Reports from the *Compare scenarios* button on application home page.

#### Scenario Comparison Reports

- 2. Individual Scenario Report:** If the report analyzes only results and KPIs from 1 scenario, use the *Individual scenario reports* section of the *Visualization Settings* configuration.

Navigate to a Scenario - > Visualization as shown in screenshot for individual scenario reports.

Plants	Customers	Demand Served	Total Cost	Total Profit
4	50	1,247,186	\$868,657,732	\$1,264,548,909,593

## 2.4.5 Set up Organization Credentials

### Power BI

Create the **User Name** and **Password** (5) for auto-deployment service within **Settings (1) -> Visualization (2) -> Org Setup (3) -> Power BI (4)** section as shown in the screenshot below.

The screenshot shows the 'Organization Dashboard Setup' page in the Decision Spot application. The left sidebar has a tree view with nodes like 'Organization', 'User', 'Visualization' (highlighted with a red box and labeled 2), 'Org Setup' (highlighted with a red box and labeled 3), 'App Setup', 'Access Management', 'DB Connection', 'Manage', and 'Customization'. The main area shows 'Organization Dashboard Setup' with a 'Select visualization type' dropdown set to 'PowerBi' (highlighted with a red box and labeled 4). Below it are fields for Client Id, Client Secret, Tenant Id, and Workspace Id. A note says 'Here, you can manage the credentials used to publish the reports. Publish report credentials already exist. This is one time setup and cannot be changed.' A red box highlights the 'User Name' field containing 'qapublishing' (labeled 5). The top right corner has a gear icon (highlighted with a red box and labeled 1).

*Settings > Visualization > Org Setup > Power BI*

### Tableau

Create the **Username** and **Password** for auto-deployment user within **Settings > Visualization > Org Setup > Tableau** section as shown in the screenshot below.

The screenshot shows the 'Organization Dashboard Setup' page for Tableau. The left sidebar has 'Org Setup' selected under 'Visualization'. The main area shows Tableau Server details: https://prod-useast-b.online.tableau.com, API Version 3.19, Site Url decisionspotforesta, Site Name Decision Spot - Foresta, Project Name Foresta QA, Personal Access Token Name APIAutomation, and Personal Access Token Secret \*\*\*\*\*. A note at the bottom says 'Here, you can manage the credentials used to publish the reports. Publish report credentials already exist. This is one time setup and cannot be changed.' A red box highlights the 'User Name qapublishing' and 'User Password \*\*\*\*\*' fields, which are also numbered 5. Other numbered callouts point to the 'Select visualization type' dropdown (2), the edit icon (4), and the top right corner (1).

*Settings > Visualization > Org Setup > Tableau*

#### Note

This is just a One-Time setup.

## 2.5 Access Control

---

### 2.5.1 User-Roles on Foresta

Foresta provides several pre-defined user roles which can be assigned to a user to control access depending on the tasks they need to perform.

There are five hierarchical roles:

1. Administrator
2. App Builder
3. Modeler
4. Analyst
5. Viewer

The permissions among the five roles can be broken down as shown in the table below:

<b>Actions</b>	<b>Administrator</b>	<b>App Builder</b>	<b>Modeler</b>	<b>Analyst</b>	<b>Viewer</b>
Add Application	1	1			
Add Project	1	1	1		
Add Scenario	1	1	1	1	
Add Teams	1				
Add Users	1				
App Customization	1	1			
Bulk Copy Scenario	1	1	1	1	
Copy Application	1	1	1		
Create Access Token	1	1	1	1	
Create Dashboard	1	1	1		
Create Label	1	1	1		
Delete Application	1	1			
Delete Label	1	1	1		
Delete Project	1	1	1		
Delete Scenario	1	1	1	1	
Delete Teams	1				
Delete Users	1				
Duplicate Scenario	1	1	1	1	
Edit Application	1	1			
Googlesheets Edit	1	1	1	1	
Edit Label	1	1	1		
Edit Project	1	1	1		
Edit Scenario	1	1	1	1	
Edit Teams	1				
Edit Users	1				
Export Solutions	1	1	1	1	1
Lock Scenario	1	1	1	1	
Manage Application Dashboard	1	1	1		
Manage Org Dashboard	1				
Solve Scenario	1	1	1	1	
Upload Table Data	1	1	1	1	
Validate Scenario	1	1	1	1	

<b>Actions</b>	<b>Administrator</b>	<b>App Builder</b>	<b>Modeler</b>	<b>Analyst</b>	<b>Viewer</b>
View Application	1	1	1	1	1
View Dashboard	1	1	1	1	1
View Global Application	1	1	1	1	1
View Organization Info	1	1	1	1	1
View Project	1	1	1	1	1
View Organization Roles	1				
View Scenario	1	1	1	1	1
View Server Info	1	1	1		
View Teams	1				
View Users	1				

## 2.5.2 Managing User Access to Projects

Projects created by users on Foresta can be shared with other users for collaboration. However, there may be cases where not everyone needs to be able to access all the projects or restrict access to projects within teams.

Foresta has functionality built in to manage user access to projects. There are three types of Access Levels:

1. **Public:** Everyone with login credentials to the server can access this project. However, what a user can change in each project is dependent on their user role.
2. **Private:** Only the creator of the project has access to this project.
3. **Restricted:** This allows the creator of the project to manage who all has access by adding explicit access using their user id of the server.

The creator can define the access level at the time of creation or project or it can be updated later by using **Edit Project**.

### Update Project

**Name**

**Description**

**Choose background (for card view):**


✓

**Metadata**

Creator rushabh.doshi@decisionspot.com  
 Date created 02-23-2023      Last modified 03-10-2023

**Access Settings**

Public  
 Private  
 Restricted

prateek.rastogi@decisionspot.com X
mohit.dobhal@decisionspot.com X
Type here...

*Update Projects Card*

## 2.6 REST API

### 2.6.1 DSpotConnect

APIs allow users perform variety of actions programmatically using Python. The API endpoints are accessible through a python library called `dspotconnect`. Built with best in class security protocols, `dspotconnect` requires PAT based authentication to establish connection with Foresta servers.

Source: [dspotconnect](#)

#### Installation

Install through pypi:

```
pip install git+https://<KEY>@github.com/decision-spot/dspotconnect@0.0.3
```

Get in touch with [Foresta Support](#) to get a valid key for your organization.

#### Usage

Select the appropriate doc for your package version.

Release	Contents
0.0.3	<ul style="list-style-type: none"> <li>- <a href="#">Package Index</a></li> <li>- <a href="#">Reference</a></li> </ul>

#### Creating Access Tokens

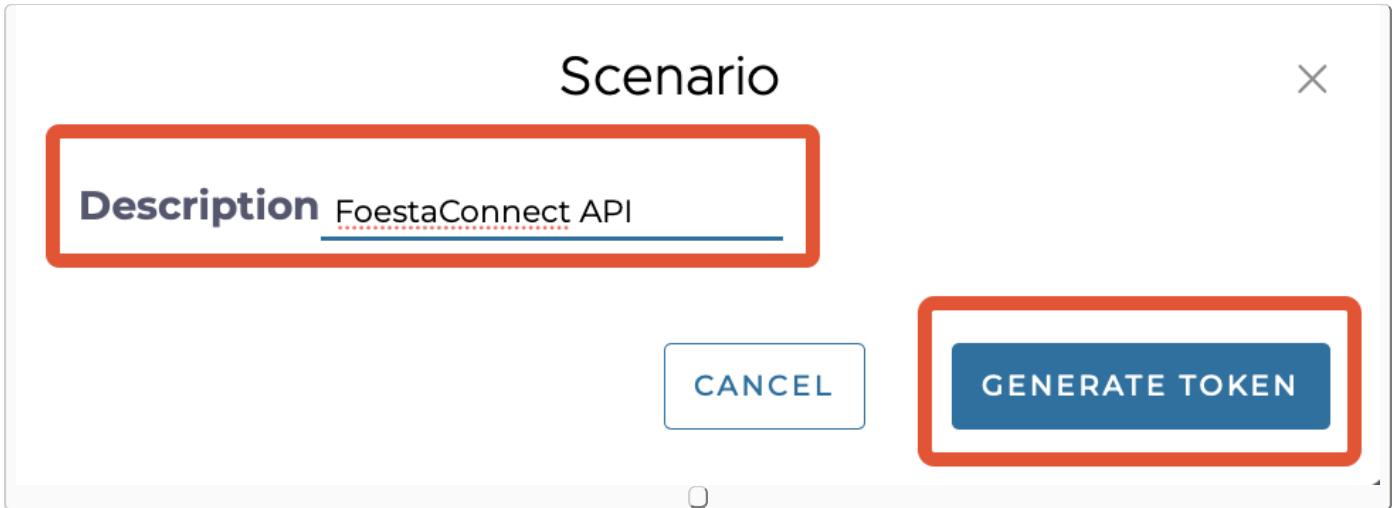
In order to use API, user needs a Personal Access Token (similar to GitHub PATs) which can be generated from the settings page.

Navigate to > **User** > **Access Token** section from the home page.

The screenshot shows the 'Personal Access Tokens' page in the Decision Spot web application. The left sidebar includes sections for 'Organization', 'Visualization', and 'Manage'. Under 'User', the 'Access Tokens' option is selected and highlighted with a red box. The main content area displays a table with columns 'Email' and 'Description', showing the message 'No Rows To Show'. In the top right corner of the main area, there is a button labeled '+ CREATE TOKEN'.

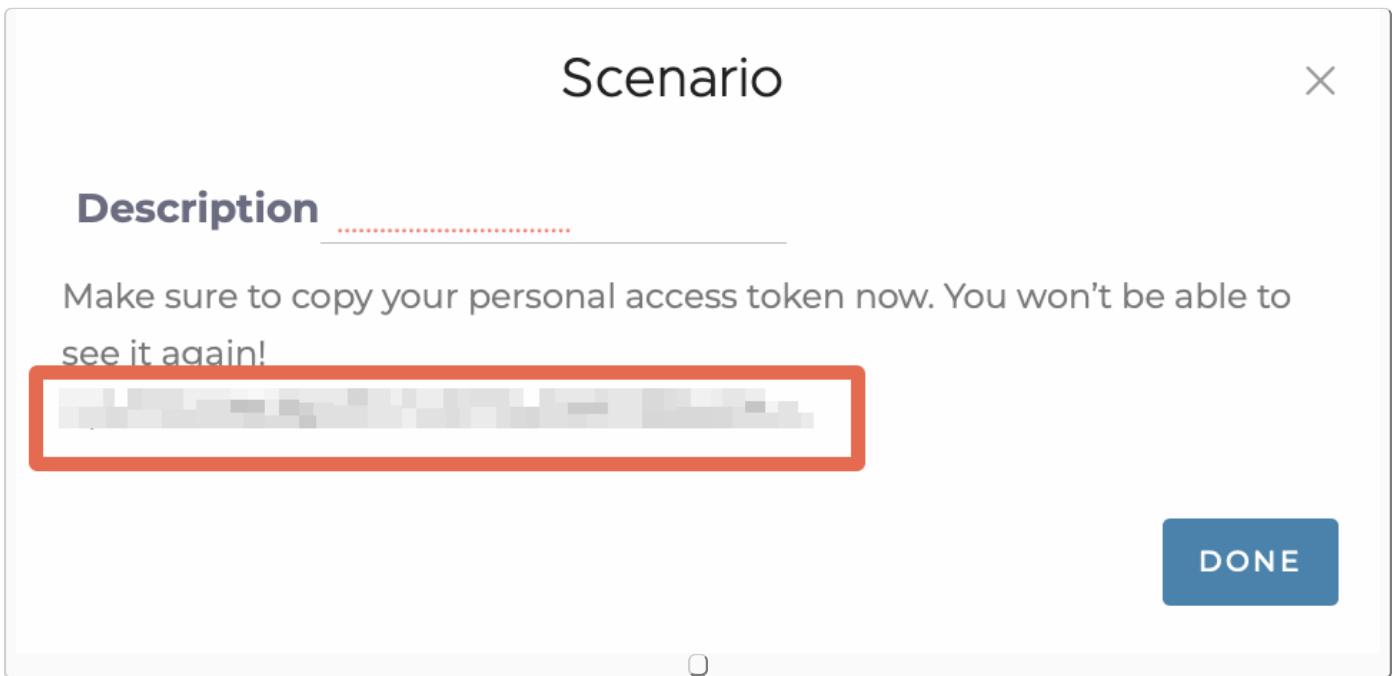
*Foresta Settings*

Click on the **CREATE TOKEN** button in the top right corner of the screen. On the following pop-up, provide the description for the token and click **GENERATE TOKEN**.



Create Token Pop-up

This generates a token as displayed on the following pop-up.



Generated Token

**Warning**

The token will be displayed only once, so the user is advised to copy it and save it in a secure location.

The tokens can be **Revoke**'ed in the case of not being used anymore.

## Personal Access Tokens

Email	Description
rushabh.doshi@decisionspot.com	FoestaConnect API

Revoke Token

*Revoke Token*

## Revoke Token

Are you sure you want to revoke this token? Any applications or scripts using this token will no longer be able to access the Roundoff API. This action cannot be undone.

Cancel

Ok

*Confirm Revoke Token*

## 2.6.2 0.0.3

---

### dspotconnect-0.0.3

- `AppConnect`
- `AppConnect.app_id`
- `AppConnect.app_name`
- `AppConnect.app_to_app()`
- `AppConnect.create_label()`
- `AppConnect.current_scenarios()`
- `AppConnect.delete_label()`
- `AppConnect.delete_scenario()`
- `AppConnect.download_table()`
- `AppConnect.duplicate_scenario()`
- `AppConnect.get_column_names()`
- `AppConnect.get_label_color()`
- `AppConnect.get_labels()`
- `AppConnect.get_logs()`
- `AppConnect.get_notes()`
- `AppConnect.get_table_names()`
- `AppConnect.is_solving_underway()`
- `AppConnect.launch_solve()`
- `AppConnect.new_scenario()`
- `AppConnect.perform_validation()`
- `AppConnect.rename_scenario()`
- `AppConnect.set_labels()`
- `AppConnect.set_notes()`
- `AppConnect.upload_input_table()`
- `TicDatConnector`
- `TicDatConnector.download_input_dat()`
- `TicDatConnector.download_solution()`
- `TicDatConnector.upload_input_dat()`

### dspotconnect-0.0.3

```
class dspotconnect.AppConnect(server, token, app_id, upload_encoding='utf-8')
```

Bases: `object`

*property app\_id property app\_name app\_to\_app(app\_to\_app\_label, src\_app\_id, src\_scenario\_id)*

Create a new scenario from the scenario of a different app using the `app_to_app` functionality deployed with this app.

- **Parameters:**

- **app\_to\_app\_label** – String associated with `self.app_name`. Apps deployed on DSpot have a non-negative number of app-to-app options associated with them. An app-to-app option can be used to move data from one app to another via scenario creation.
- **src\_app\_id** – The ID of the app we are sourcing from.
- **src\_scenario\_id** – The scenario ID we are sourcing from. This is a scenario of the source app.
- **Returns:** the scenario ID of the new scenario.

**NB** - the onus is on you to select an appropriate `app_to_app_label`, `src_app_id`, `src_scenario_id` triplet.

Certain `app_to_app_labels` will only work with certain sourcing apps, and even with a proper sourcing app selection, there might be a scenario data requirement. If there is a failure in converting, then a new scenario will still be created, but it might only be partially populated.

```
create_label(label, label_color="#0072A3")
```

creates a new label :type label: :param label: the name of the new label

WARNING - mayhem will result if label refers to a label that already exists. \* **Parameters:** `label_color` – optional. string setting the color of the label \* **Returns:**

```
current_scenarios()
```

- **Returns:** a dict mapping scenario ID to scenario name

```
delete_label(label)
```

deletes a label

- **Parameters:** `label` – the label that is to be deleted

- **Returns:**

```
delete_scenario(scenario)
```

- **Parameters:** `scenario` – if an int, then a scenario ID. If a string, then a scenario name.

- **Returns:**

```
download_table(scenario, sub_schema_type, table_name, download_type="DataFrame")
```

- **Parameters:**

- `scenario` – if an int, then a scenario ID. If a string, then a scenario name.

- `sub_schema_type` – one of “input”, “validation”, “solution”

- `table_name` – a table name for the appropriate sub\_schema

- `download_type` – either “DataFrame” or “list-of-lists”

- **Returns:** the downloaded table in the appropriate format.

```
duplicate_scenario(scenario)
```

- **Parameters:** `scenario` – if an int, then a scenario ID. If a string, then a scenario name.

- **Returns:** the scenario ID of the new scenario. It will be a duplicate of the scenario identified by the scenario argument.

get\_column\_names(sub\_schema\_type, table\_name)

- **Parameters:**

- **sub\_schema\_type** – one of “input”, “validation”, “solution”

- **table\_name** – a table name for the appropriate sub\_schema

- **Returns:** a list of column names (in order). The column names will be the display names.

get\_label\_color(label)

returns the label color for a given label (as a string)

- **Parameters: label** – a string that represents a given label (i.e. name of the label)

- **Returns:** the color of the label, as a string

get\_labels(scenario)

returns the labels (as a list of strings) for a given scenario :type scenario: :param scenario: if an int, then a scenario ID. If a string, then a scenario name. :return: a list of strings. Empty list if there are no labels for this scenario.

get\_logs(scenario)

Retrieve the information displayed in the Logging section of the GUI

- **Parameters: scenario** – if an int, then a scenario ID. If a string, then a scenario name.

- **Returns:** A dict with keys “standardStreams” and “solutionLogs”

get\_notes(scenario)

returns the scenario notes for a given scenario

- **Parameters: scenario** – if an int, then a scenario ID. If a string, then a scenario name.

- **Returns:** the notes for this scenario. Will be empty string if there are no notes.

get\_table\_names(sub\_schema\_type)

- **Parameters: sub\_schema\_type** – one of “input”, “validation”, “solution”

- **Returns:** a list of table names. The table names will be the display names.

is\_solving\_underway(scenario)

Companion to launch\_solve. After a solve has launched, use this subroutine to determine if the solve is still underway or if it has finished already.

- **Parameters: scenario** – if an int, then a scenario ID. If a string, then a scenario name.

- **Returns:** True if solving is still underway, false if finished.

launch\_solve(scenario)

This returns right away. Solving has a high liklihood of being a long running task, and thus we finish right away. Use is\_solving\_underway to see if the solve is still ongoing, or if it has finished yet.

- **Parameters: scenario** – if an int, then a scenario ID. If a string, then a scenario name.

- **Returns:**

new\_scenario(name)

Create a new scenario

- **Parameters: name** – name of the scenario to create

- **Returns:** the scenario ID of the new scenario

perform\_validation(scenario)

This doesn't return until the validation step is completed. Validation is considered something akin to uploading - its unlikely to be a long running task and thus the finish delay is built into the API.

- **Parameters:** `scenario` – if an int, then a scenario ID. If a string, then a scenario name.
- **Returns:** no need for a return status check - just look in the validation tables to see if there was a validation failure

```
rename_scenario(scenario, name)
```

- **Parameters:**
- `scenario` – if an int, then a scenario ID. If a string, then a scenario name.
- `name` – the new name for the scenario
- **Returns:**

```
set_labels(scenario, labels)
```

sets the labels for a scenario. Will overwrite any previous labeling of the scenario

- **Parameters:**
- `scenario` – scenario: if an int, then a scenario ID. If a string, then a scenario name.
- `labels` – list of label strings. can be empty list, in which case the scenario will be cleared of labels
- **Returns:**

```
set_notes(scenario, notes)
```

sets the scenario notes for a given scenario :type scenario: :param scenario: if an int, then a scenario ID. If a string, then a scenario name. :type notes: :param notes: the string that should be the new scenario notes :return:

```
upload_input_table(scenario, table_name, table_data)
```

Will overwrite the entire table. An empty table\_data will delete the entire table. To edit a subset of the table, download the entire table, edit the appropriate section, upload the entire table.

- **Parameters:**
- `scenario` – if an int, then a scenario ID. If a string, then a scenario name.
- `table_name` – a table name for the “input” sub\_schema (only input tables are editable).
- `table_data` – Either a pandas.DataFrame or a list-of-lists.
- **Returns:**

```
class dspotconnect.TicDatConnector(con, engine)
```

Bases: `object`

Helper class to support Python developer running a ticdat based engine remotely on DSpot.

```
download_input_dat(scenario, process_active_tables='deactivate cascading')
```

download the input dat object. Always returns an input dat object, even if all the tables are empty.

- **Parameters:**
- `scenario` – if an int, then a scenario ID. If a string, then a scenario name.
- `process_active_tables` – one of the following values ‘raw’ : keep both active and inactive entries. See `duplicates_ticdat_init` if using `TicDatFactory`. ‘deactivate immediate’: remove inactive rows from active\_tables. ‘deactivate cascading’: remove inactive rows from active tables, and then cascade those deletions downward by removing the FK failures. The object returned with this option will thus mirror the object passed to the solve function. Don’t use this option if the input data has foreign key failures as-is, as you might end up removing the FK fails that were present prior to the FK fails. \* **Returns:** a dat object associated with the `input_schema`

```
download_solution(scenario)
```

download the solution dat object from DSpot. Don't call until after a solution has been performed. Will return None if all the solution tables are empty

- **Parameters:** **scenario** – if an int, then a scenario ID. If a string, then a scenario name.
- **Returns:** a solution\_schema compatible dat object (or None if all solution tables are empty)

upload\_input\_dat(dat, scenario\_name='Autogenerated by dspotconnect')

creates a new scenario on DSpot from the dat object

- **Parameters:**
- **dat** – an input\_schema compatible dat object
- **scenario\_name** – the name of the scenario to create. NB - the scenario created by this routine might be this string, or it might be this string with a number postpended. This routine always creates a new scenario with a unique name.
- **Returns:** the ID of the scenario that was created.

## 3. App Builder

---

### 3.1 App Builder

---



#### Who is an app builder?

- A Data-Scientist/OR Scientist, Data Engineer, Analyst etc.
- Can code in python
- Understands mathematical modeling

**So what now?**

If 'App Builder' is a good description of yourself and want to create apps for your organization/clients, you're at the right place!

- 👉 Use the navigation to jump to the relevant section.

## 3.2 Getting Started

### 3.2.1 Getting started

Some context to build your intuition about apps.



Enterprise decision makers often need softwares that hides the ML or Optimization algorithm under the hood of rich analytics features like Data Grids, Clean and simple GUI, Scenarios, Dashboards, etc. The integration of such softwares in an enterprise further requires solid administrative tools such as User-Access Control, Package Management, Server Management, Security Compliance etc.

Data scientists (The App Builders) are super skilled at developing models and visualizations but are not really equipped with enough skills or in some cases with enough time to develop feature rich web applications. So, data-scientists rely on engineering to serve the needs of stakeholders. This, unfortunately, creates many inter-dependencies and restricts the impact a data-scientist can make.

**So, How do we solve this?**

While designing Foresta, our team carefully thought ... how we could empower data scientists to build production grade decision apps without really having to develop web-applications? How we can make this a self-serve process? How to take away the complex engineering involved?

Turns out, a lot of those essential components are reproducible and from one app to another!

What varies is the data model, core algorithm and the visualizations. Data Model is essential component of app building, data model logically defines the template for input and outputs of the algorithm. The algorithm is the core logic of the app and visualizations enable interpretation of complex data using visual tools such as charts, and graphs.

## 3.3 Create Applications

### 3.3.1 Create using GitHub

To create an app via GitHub:

1. Navigate to the Applications page from Foresta Home page.

The screenshot shows the 'Decision Spot' application management interface. On the left is a search bar. The main area displays a grid of project cards. Each card contains the project name, created by, created on, modified on, and a brief description. The projects listed are: Ehsan Playground, QA Work, Lourenço QA, Elliot TO, Aditi's Projects, Slim Learning, DS Project, Bernardo Project, Spot Women PlayGround, and Sabarish Projects. In the top right corner of the interface, there is a blue 'Add Project' button. A red box highlights this button, indicating it is the target for the first step in creating a new application.

2. On the upper-right corner, select **Add App** and a dropdown menu will expand. Select the **From GitHub** option.

This screenshot shows the same 'Decision Spot' application management interface as the previous one, but with a different focus. The 'Add Project' button has been clicked, which has triggered a dropdown menu. The menu is titled 'Add App' and contains two options: 'From File' and 'From GitHub'. A red box highlights the 'From GitHub' option, indicating it is the selected path for creating a new application. The rest of the interface remains the same, showing the list of existing projects.

 Note

Alternatively, you can scroll down the screen and do the same process by the **Add Application** box.

 Decision Spot

[Profile](#)
[Dashboard](#)
[Grid](#)
[Settings](#)
[Logout](#)

[Add App](#)

**Diet App to App Overwrite**

Created By	lourenco.p.soares@decisionspot.com
Created On	21-AUG-23
App Version	2
App Id	145
Projects	3

**App Custom Column Format**

Created By	lourenco.p.soares@decisionspot.com
Created On	21-AUG-23
App Version	
App Id	146
Projects	1

**test html tooltips**

Created By	mohit.mahajan@decisionspot.co
Created On	22-AUG-23
App Version	0.0.1
App Id	163
Projects	1

**Diet Formatting**

Created By	sagar.khanduja@decisionspot.c
Created On	23-AUG-23
App Version	0.0.1
App Id	167
Projects	1

**dro 13**

Created By	rushabh.doshi@decisionspot.co
Created On	24-AUG-23
App Version	main
App Id	170
Projects	0

**dro 14**

Created By	rushabh.doshi@decisionspot.co
Created On	24-AUG-23
App Version	main
App Id	171
Projects	0

**so\_with\_twbx**

Created By	rushabh.doshi@decisionspot.co
Created On	25-AUG-23
App Version	main
App Id	172
Projects	0

**facility\_location**

Created By	mohit.mahajan@decisionspot.co
Created On	12-SEP-23
App Version	0.0.1
App Id	185
Projects	1

**Landed Cost 0.58**

Created By	sagar.khanduja@decisionspot.c
Created On	18-SEP-23
App Version	b812331
App Id	196
Projects	3

**NO test**

Created By	lourenco.p.soares@decisionspot.co
Created On	22-SEP-23
App Version	0.0.24
App Id	203
Projects	1

**Tree Visualization Example**

Created By	lourenco.p.soares@decisionspot.co
Created On	26-SEP-23
App Version	1
App Id	205
Projects	1

**param test**

Created By	peter.cacioppi@decisionspot.co
Created On	28-SEP-23
App Version	blah
App Id	212
Projects	1

  
**Add Application**  

[File](#)
[Github](#)

1. Fill the app **Name**, link to the **GitHub** repository, inform the **Release Tag** for the application version to be used and the GitHub **Personal Access Token** if the repository is **Private**. Set the access level using **Access Settings** and click **Confirm** to deploy the app.

### New Application from Github

**Name**

**Repo Url**

**Release Tag**

**Token (optional)**

**Access Settings**

Public

Private

Restricted

Cancel
Confirm

**Note**

The release tag value can also be one the following (if you're using pre-release or just testing an application):

- Complete or partial SHA for the default branch.
- The name of the feature branch itself (in which case it uses the most recent commit for the feature branch)

Once the app is created, the user will see a card for the deployed application as shown below.

## TTS Diet

Created By rushabh.doshi@decisionspot.com

Created On 16-MAR-23

App Version 0.0.2

App Id 7

Projects 1

### 3.3.2 Create using File

To create an app via File Upload:

1. Navigate to the Applications page from Foresta Home page.

2. On the upper-right corner, select **Add App** and a dropdown menu will expand. Select the **From File** option.

 Note

Alternatively, you can scroll down the screen and do the same process by the **Add Application** box.

 Decision Spot

[Profile](#)
[Logout](#)

**Add App**

**Diet App to App Overwrite**

Created By	lourenco.p.soares@decisionspot.com
Created On	21-AUG-23
App Version	2
App Id	145
Projects	3

**App Custom Column Format**

Created By	lourenco.p.soares@decisionspot.com
Created On	21-AUG-23
App Version	
App Id	146
Projects	1

**test html tooltips**

Created By	mohit.mahajan@decisionspot.com
Created On	22-AUG-23
App Version	0.0.1
App Id	163
Projects	1

**Diet Formatting**

Created By	sagar.khanduja@decisionspot.com
Created On	23-AUG-23
App Version	0.0.1
App Id	167
Projects	1

**dro 13**

Created By	rushabh.doshi@decisionspot.com
Created On	24-AUG-23
App Version	main
App Id	170
Projects	0

**dro 14**

Created By	rushabh.doshi@decisionspot.com
Created On	24-AUG-23
App Version	main
App Id	171
Projects	0

**so\_with\_twbx**

Created By	rushabh.doshi@decisionspot.com
Created On	25-AUG-23
App Version	main
App Id	172
Projects	0

**facility\_location**

Created By	mohit.mahajan@decisionspot.com
Created On	12-SEP-23
App Version	0.0.1
App Id	185
Projects	1

**Landed Cost 0.58**

Created By	sagar.khanduja@decisionspot.com
Created On	18-SEP-23
App Version	b812331
App Id	196
Projects	3

**NO test**

Created By	lourenco.p.soares@decisionspot.com
Created On	22-SEP-23
App Version	0.0.24
App Id	203
Projects	1

**Tree Visualization Example**

Created By	lourenco.p.soares@decisionspot.com
Created On	26-SEP-23
App Version	1
App Id	205
Projects	1

**param test**

Created By	peter.cacioppi@decisionspot.com
Created On	28-SEP-23
App Version	blah
App Id	212
Projects	1

  
**Add Application**  

[File](#)
[Github](#)

1. Fill the app **Name** and **Version** and then click on **Choose file**. Select the `.zip` or `.py` of the python package. Set the access level using **Access Settings** and click **Confirm** to deploy the package.

### New Application from File

Name

App Version

Choose file

Access Settings

Public

Private

Restricted

Cancel Confirm

Once the app is created, the user will see a card for the deployed application as shown below.

## TTS Diet

Created By rushabh.doshi@decisionspot.com

Created On 16-MAR-23

App Version 0.0.2

App Id 7

Projects 1

## 3.4 Reporting Automation

### 3.4.1 Automated Publishing of PowerBI Reports

Follow the steps provided [here](#) to create visualizations. The app builders can set up **Automated Publishing of PowerBI reports** into Foresta applications.

Automated Publishing means that you can create apps with pre-defined published reports. So you would not need to publish again the same report every time that a new project is created.

#### Organise the package repository in required structure

After creating the reports, take the PowerBI files (.pbix) and organise it in the package repository as shown in representation below.

```
{app_package_name}
--> config.json
--> {app_package_name}
--> {app_package_name}_pbix
-->--> scenario
-->--> comparison
```

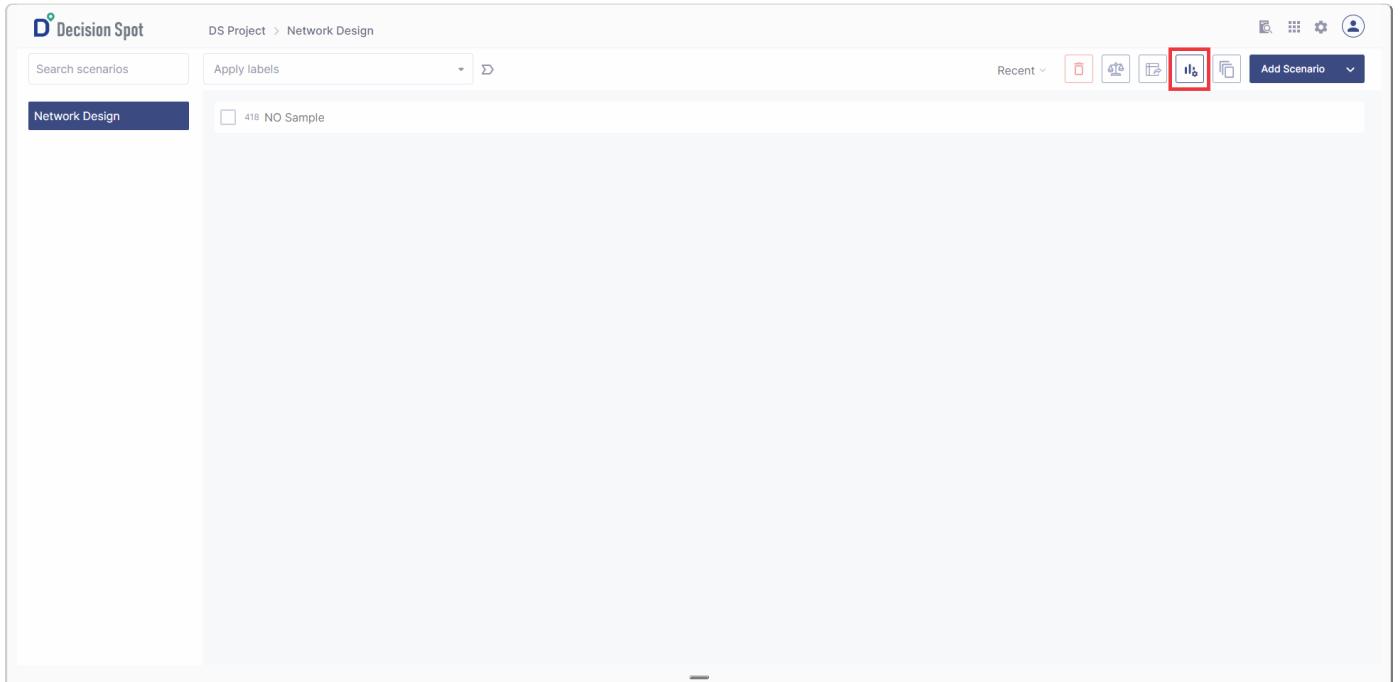
The **Individual Scenario** reports go in the **scenario** folder and the **Scenario Comparison** reports go in the **comparison** folder.

#### Create the application

Create an application on Foresta server.

#### Create a project and add the application

Create a project and add the application with PowerBI configured for auto-deployment. The instruction for that can be found [here](#). Users can check the whether the PowerBI workbooks were auto-deployed by clicking on the **Visualization Settings** button as shown below.



### 3.4.2 Automated Publishing of Tableau Reports

Similar to PowerBI, Foresta has the capability to automatically deploy Tableau workbooks with the application deployment process. This requires certain configuration settings to Tableau workbooks along with the application code repository. This document walks the user through all the steps required to setup auto-deployment.

The instructions to create and publish Tableau workbooks to a Foresta app can be found [here](#).

#### Create reports in Tableau connected with Foresta Database

Creating reports in Tableau requires connecting to the Foresta application database (PostgreSQL) from Tableau workbook, and the steps for it are shown [here](#).

#### Organise the package repository in required structure

After creating the reports, take the Tableau files (.twbx) and organise it in the package repository as shown in representation below.

```
{app_package_name}
--> config.json
--> {app_package_name}
--> {app_package_name}_twbx
-->> scenario
-->>> comparison
```



Notice that the Tableau files need to be at the packaged format (**.twbx**). To convert them to this format, you need to **Save As a Tableau Packaged Workbook**.

The **Individual Scenario** reports go in the **scenario** folder and the **Scenario Comparison** reports go in the **comparison** folder.

#### Create the application

Create an application on Foresta server.

#### Create a project and add the application

Create a project and add the application with Tableau configured for auto-deployment. The instruction for that can be found [here](#).

User can check the whether the Tableau workbooks were auto-deployed by clicking on the **Visualization Settings** button as shown below.

The screenshot shows the Decision Spot web application interface. At the top left is the logo 'Decision Spot'. The top navigation bar includes 'DS Project > Network Design', a 'Search scenarios' input field, an 'Apply labels' dropdown, and a 'Recent' dropdown with several icons: a red square, a blue square, a green square, a yellow square, a purple square, a grey square, and a 'Run' icon (highlighted with a red box). To the right of these are 'Add Scenario' and a user profile icon. On the left, a sidebar titled 'Network Design' contains a single item: '418 NO Sample' with a checkbox next to it. The main workspace is currently empty.

### 3.5 Deprecate and Delete Applications

---

### 3.5.1 Deprecating Apps

---

To **deprecate** an App, follow the steps below:

1. Navigate to the Applications page from Foresta Home page.

The screenshot shows a grid of ten application cards. Each card displays basic information such as the app name, created by, created on, modified on, and description. The cards are arranged in two rows of five. The top row includes cards for 'Ehsan Playground', 'QA Work', 'Lourenço QA', 'Elliot TO', and 'Aditi's Projects'. The bottom row includes cards for 'Slim Learning', 'DS Project', 'Bernardo Project', 'Spot Women PlayGround:D', and 'Sabarish Projects'. The interface has a light blue header with the 'Decision Spot' logo and a search bar. On the far right, there are several icons, with the three-dot menu icon being specifically highlighted with a red box.

2. In the App card, select the three dots icon on the upper-right corner.

This screenshot shows a detailed view of an application card titled 'My App'. The card lists the following information: Created By (dspotdev.support@decisionspot.com), Created On (28-SEP-23), App Version (1), App Id (419), and Projects (0). A red box highlights the three-dot menu icon located in the top right corner of the card.

3. Select **Deprecate**. A confirmation pop-up will ask for confirmation.

This screenshot shows the same 'My App' card from the previous step, but with a context menu open over it. The menu options are 'Edit', 'Deprecate', and 'Delete'. The 'Deprecate' option is highlighted with a red box. The menu has a light purple background and white text.

4. Deprecated apps will be shown as opaque cards in the bottom of the Apps navigation page, and marked as **Deprecated**.

Decision Spot

Add App

<b>dro 14</b>	<b>so_with_twbx</b>	<b>facility_location</b>	<b>Landed Cost 0.58</b>	<b>NO test</b>
Created By rushabh.doshi@decisionspot.com	Created By rushabh.doshi@decisionspot.com	Created By mohit.mahajan@decisionspot.com	Created By sagar.khanduja@decisionspot.com	Created By iourenco.p.soares@decisionspot.com
Created On 24-AUG-23	Created On 25-AUG-23	Created On 12-SEP-23	Created On 18-SEP-23	Created On 22-SEP-23
App Version main	App Version main	App Version 0.0.1	App Version b812331	App Version 0.0.24
App Id 171	App Id 172	App Id 185	App Id 196	App Id 203
Projects 0	Projects 0	Projects 1	Projects 3	Projects 1

<b>Tree Visualization Example</b>	<b>param test</b>
Created By iourenco.p.soares@decisionspot.com	Created By peter.cacioppi@decisionspot.com
Created On 26-SEP-23	Created On 28-SEP-23
App Version 1	App Version blah
App Id 205	App Id 212
Projects 1	Projects 1



Add Application

File    Github

<b>My App</b>	<b>Deprecated</b>
Created By dspotdev.support@decisionspot.com	
Created On 28-SEP-23	
App Version 1	
App Id 214	
Projects 0	



Deprecated App



You can **Activate** or **Delete** a deprecated app at any time.

### 3.5.2 Deleting Apps

---

To **delete** an App, follow the steps below:

1. Navigate to the Applications page from Foresta Home page.

The screenshot shows the 'Decision Spot' application list page. It displays ten app cards arranged in two rows of five. Each card contains the app name, created by, created on, modified on, and description. A red box highlights the three-dot menu icon in the top right corner of the first card.

App Name	Created By	Created On	Modified On	Description
Ehsan Playground	ehsan.khodabandeh@decisionspot.com	22-MAR-23	28-SEP-23	Deploy apps for the engines and projects I work on
QA Work	peter.cacioppi@decisionspot.co m	30-MAR-23	18-JUL-23	
Lourenço QA	lourenco.p.soares@decisionspo t.com	06-APR-23	27-SEP-23	
Elliot TO	ehsan.khodabandeh@decisionspo t.com	11-APR-23	11-APR-23	Mainly for NEMO
Adit's Projects	aditi.taneja@decisionspot.com	26-APR-23	28-SEP-23	
Slim Learning	mohit.mahajan@decisionspot.co m	09-MAY-23	12-SEP-23	
DS Project	lourenco.p.soares@decisionspo t.com	14-JUN-23	14-JUL-23	
Bernardo Project	bernardo.furtado@decisionspot. com	22-JUN-23	22-JUN-23	
Spot Women PlayGround :D	sonu.gupta@decisionspot.com	22-JUN-23	22-JUN-23	
Sabarish Projects	sabarish.chockalingam@decisio nspot.com	22-JUN-23	22-JUN-23	

2. In the App card, select the three dots icon on the upper-right corner.

The screenshot shows the 'My App' card details page. It lists the app's created by, created on, app version, app id, and projects. A red box highlights the three-dot menu icon in the top right corner of the card.

Attribute	Value
Created By	dspotdev.support@decisionspot.com
Created On	28-SEP-23
App Version	1
App Id	419
Projects	0

3. Click on **Delete**.

The screenshot shows the 'My App' card details page with a dropdown menu open. The menu includes options for Edit, Deprecate, and Delete. The Delete option is highlighted with a red box. The card itself lists the app's details: created by, created on, app version, app id, and projects.

Attribute	Value
Created By	dspotdev.support@decisionspot.com
Created On	28-SEP-23
App Version	1
App Id	419
Projects	0

**Note**

To delete deprecated apps, follow the same steps above on the deprecated app in the bottom of the apps navigation page.

**Decision Spot**

**Add App**

Search	App Details	App Details	App Details	App Details	
Search	<b>dro_14</b> Created By: rushabh.doshi@decisionspot.com Created On: 24-AUG-23 App Version: main App Id: 171 Projects: 0	<b>so_with_twbx</b> Created By: rushabh.doshi@decisionspot.com Created On: 25-AUG-23 App Version: main App Id: 172 Projects: 0	<b>facility_location</b> Created By: mohit.mahajan@decisionspot.com Created On: 12-SEP-23 App Version: 0.0.1 App Id: 185 Projects: 1	<b>Landed Cost 0.58</b> Created By: sagar.khanduja@decisionspot.com Created On: 18-SEP-23 App Version: b812331 App Id: 196 Projects: 3	<b>NO test</b> Created By: lourenco.p.soares@decisionspot.com Created On: 22-SEP-23 App Version: 0.0.24 App Id: 203 Projects: 1
	<b>Tree Visualization Example</b> Created By: lourenco.p.soares@decisionspot.com Created On: 26-SEP-23 App Version: 1 App Id: 205 Projects: 1	<b>param test</b> Created By: peter.cacioppi@decisionspot.com Created On: 28-SEP-23 App Version: blah App Id: 212 Projects: 1			
			 Add Application <a href="#">File</a> <a href="#">Github</a>		
	<b>My App</b> Created By: dspotdev.support@decisionspot.com Created On: 28-SEP-23 App Version: 1 App Id: 214 Projects: 0	<b>Deprecated</b>			

## 4. Applications

---

### 4.1 Applications

---

#### **What are Decision Spot applications?**

Decision Spot's Foresta platform is powered by applications that integrate the broad spectrum of open-source machine learning and AI libraries. These are useful in helping users minimize inventory, identify risks, streamline transportation, omnichannel fulfillment, and much more. These applications on Foresta unify all supply chain design capabilities onto a single, user-friendly interface, enabling you to evaluate operational levers swiftly and make data-driven decisions. With Foresta applications, a number of intelligent supply chain management solutions are made easily available.

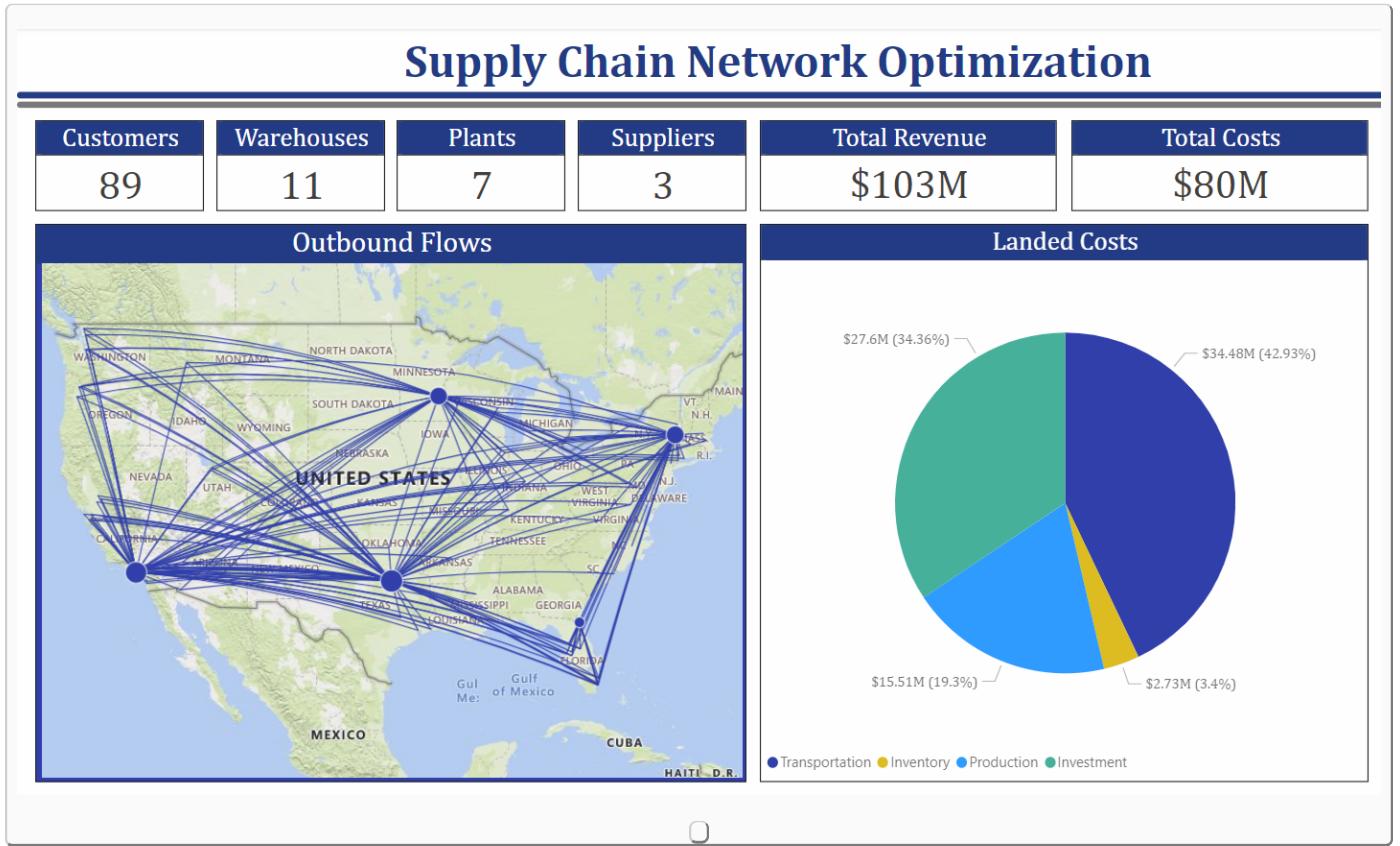
## 4.2 Supply Chain Network Optimization

### 4.2.1 Supply Chain Network Optimization

#### WHAT IS SUPPLY CHAIN NETWORK OPTIMIZATION?

Supply Chain Network Optimization is a strategic approach to designing and managing a company's supply chain to **maximize efficiency, minimize costs, and enhance overall performance**. It involves analyzing and fine-tuning various aspects of the supply chain network, such as sourcing of raw materials, capacity of production facilities, location of distribution centers, transportation lanes, and inventory throughput. The goal is to create a streamlined and cost-effective supply chain that can meet customer service levels while minimizing operational expenses and risks.

The supply chain network optimization app leverages mathematical modeling with powerful **Gurobi™** solver to calculate the best supply and demand scenarios. This tool offers essential input tables, output reports, and a mathematical optimization model to achieve optimal solutions for your supply chain design.



Supply Chain Network Optimization

👉 Use the navigation to explore further details

## 4.2.2 Input Tables

### Core Data

#### SITES

Populate this table to create sites (i.e. plants, warehouses, customers, DCs, suppliers, etc.). All models need at least one site. Site names should be unique in the model.

##### Name

*Type: Text*

Identifies the site. Will also create a singleton zone

##### Active

*Type: Boolean*

Used for Including or Excluding the site from Optimization.

True

False



Beware. Any string other than 'True' or 'true' will deactivate this record. The model does not accept binary values of 0 and 1 as False and True.

##### Latitude

*Type: Numeric*

Used for Haversine calculations and mapping

##### Longitude

*Type: Numeric*

Used for Haversine calculations and mapping

##### Sourcing Status

*Type: Text*

Use this column to define the sourcing strategy for the site.

**Single:** The site in scope can be supplied by a single site only

**Multiple:** The site in scope Can be supplied by more than one site

##### Open Close Status

*Type: Text*

This column can be used to determine the status of the site for Optimization.

**Potential:** The model will open or not the site based on the Optimization

**Fixed:** The site will be opened regardless of the Optimization

**Pre-Existing:** A Pre-Existing site is already open, the model can close it or not based on the Optimization

##### Inventory Status

*Type: Text*

Use this column to define the Inventory strategy for a site.

**Allows Inventory Between Periods:** Inventory can be stored at the site across periods

**Flow Through Only:** This is a flow through site only with no cross period inventory storage

##### Opening Cost

*Type: Text*

One off cost to open a previously unopened site. Non-negative, non-infinite.

**Operating Cost***Type: Text*

Fixed cost paid for every time period a site is open. Opening Cost and Operating Cost can both apply. Non-negative, non-infinite.

**Closing Cost***Type: Text*

One off cost to close a previously opened site. Non-negative, non-infinite

**Postal Code***Type: Text*

Postal Code of the site. Used for UPS / FedEx rating engines.

**Zone 1***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Zone 2***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Zone 3***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Zone 4***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Zone 5***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Zone 6***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Zone 7***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Zone 8***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Zone 9***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Zone 10***Type: Text*

Use this column to define a zone. Don't define singleton or all zones.

**Notes**

*Type: Text*

Descriptive Column to provide additional information about the Site.

Sample Input

Name	Active	Latitude	Longitude	Sourcing Status	Open Close Status	Inventoried
P_Dover	True	39.14	-75.58	Multiple	Fixed	Flow through
W_El Paso	False	31.76	-106.47	Multiple	Potential	Flow through
C_Albany	True	42.65	-73.75	Single	Fixed	Flow through

## PRODUCTS

Populate this table to create products. All models need at least one product.

### Name

*Type: Text*

Identifies the product. Will also create a singleton product grouping.

### Active

*Type: Boolean*

Used for Including or Excluding the product from Optimization.

True

False



Beware. Any string other than 'True' or 'true' will deactivate this record. The model does not accept binary values of 0 and 1 as False and True.

### Weight

*Type: Numeric*

Weight of one product unit. Used for transportation rating. Positive, non-infinite.

### Shipment Volume

*Type: Numeric*

Shipment volume of one product unit. Used for transportation rating. Positive, non-infinite.

### Warehouse Volume

*Type: Numeric*

Volume of one product unit used for inventory volume calculations. Positive, non-infinite.

### Adjusted Volume

*Type: Numeric*

Similar to Shipment Volume. Used for transportation rating. Positive, non-infinite.

### Freight Class

*Type: Text*

Used for CzarLite transportation rating. (Not implemented as of 0.0.8)

### Grouping X

*Type: Text*

Use this column to define a product grouping. Don't define singleton or all grouping.

### Notes

*Type: Text*

Descriptive Column to provide additional information about the Site.

## Sample Input

Name	Active	Weight	Shipment Volume	Warehouse Volume	Adjusted Volume	Freight
FG_Tuna Sandwich	True	0.66	1	1	1	50
PM_Tuna and Mayonnaise Mixture	True	1.00	1	1	1	50
RM_Canned Tuna	True	0.37	1	1	1	50
RM_Mayonnaise	True	0.55	1	1	1	50
RM_Onion	True	0.33	1	1	1	50
RM_Plastic Package	True	0.10	1	1	1	50

**TIME PERIODS**

Populate this table to create time periods. All models need at least one time period.

## Name

*Type: Text*

Identifies the time period. Will also create a singleton time period grouping.

## Chronological Order

*Type: Numeric*

Time period ordering will be determined by sorting this field, with ties broken by Name.

## Grouping X

*Type: Text*

Use this column to define a time period grouping. Don't define singleton or all grouping.

## Notes

*Type: Text*

Descriptive Column to provide additional information about the Site.

## Sample Input

Name	Chronological Order	Grouping 1	Grouping 2	Grouping 3	Grouping 4	Grouping 5
M+1	1	Month				
M+2	2	Month				
M+3	3	Month				

## PRODUCTION LINES

Populate this table to model different production facilities within the same site. If you populate this table, you should also populate Production By Production Lines.

### Site

*Type: Text*

Must match a valid site.

### Name

*Type: Text*

Identifies the production line within a site. Will NOT create a singleton production line grouping.

### Active

*Type: Boolean*

Used for Including or Excluding the site from Optimization.

`True`

`False`



Beware. Any string other than 'True' or 'true' will deactivate this record. The model does not accept binary values of 0 and 1 as False and True.

### Open Close Status

*Type: Text*

This column can be used to determine the status of the production line for Optimization.

`Potential`: The Optimization will consider to open or not this production line

`Fixed`: The production line will be opened, regardless of the Optimization

`Pre-Existing`: The Optimization will consider to close or not a pre-existing production line

### Opening Cost

*Type: Numeric*

One off cost to open a previously unopened production line. Non-negative, non-infinite.

### Operating Cost

*Type: Numeric*

Fixed cost paid for every time period a production line is open. Opening Cost and Operating Cost can both apply. Non-negative, non-infinite.

### Closing Cost

*Type: Numeric*

One off cost to close a previously opened production line. Non-negative, non-infinite

### Grouping X

*Type: Text*

Use this column to define a production line grouping. Don't define singleton or all grouping.

### Notes

*Type: Text*

Descriptive Column to provide additional information about the Site.

## Sample Input

<b>Site</b>	<b>Name</b>	<b>Active</b>	<b>Open Close Status</b>	<b>Opening Cost</b>	<b>Operating Cost</b>	<b>Closing</b>
P_Dover	P_Dover_Line_A	True	Fixed	1000000	50000	300000
P_Dover	P_Dover_Line_B	True	Fixed	1200000	60000	400000
P_Austin	P_Austin_Line_A	True	Fixed	2000000	75000	250000

## PARAMETERS

In this table the user can configurate how the optimization model will behave, setting up different parameters that will influence the optimization outputs.

At the end of this page, you can understand more about each parameter at the [Parameters Explanation](#) section.

### Note

On the upper-left corner, just next to Hide Empty Tables, you can change the display of this table, by using the toggle button **Tab View**. Deactivate that option to navigate in the Parameters table using the **Menu View**.

#### Key

*Type: Text*

Name of the parameter.

#### Value

*Type: Text*

Value of the parameter. Edit based on the description.

#### Category

*Type: Text*

A field used to categorize and organize parameters in sections.

#### Description

*Type: Text*

Description of the parameter. Provides information about the values parameters can take.

#### Parameters explanation

To complement the explanation below, take a look at the [Parameters](#) page to understand more about how each parameter works.

**MIP Gap:** Non-negative float. Sets a relative tolerance on the gap between the best integer objective and the objective of the best node remaining. Defined as a percentage, not as a ratio. Scaled by 0.01 before being passed to Gurobi.

**Objective:** Select the solution KPI to be the objective. Possible values are 'Total Cost', 'Total Profit', 'Total Revenue', 'Percentage High Service Demand', 'Critical Sites Opened', 'Total Demand Satisfied' or 'Total Infeasibility'.

**Max Total Cost:** Upper bound on the Total Cost KPI of a feasible solution. Use 'inf' to represent infinity. Negative numbers not allowed.

**Min Percentage High Service Demand:** Lower bound on the Percentage High Service Demand KPI of a feasible solution. Must be between 0 and 100, inclusive. This parameter can work together with the parameter High Service Demand Threshold. For example, if equals to 80, and High Service Demand Threshold = 500, 80% of the demand will be attended within the distance of 500 miles (assuming that Distance Type = Miles).

**Min Total Profit:** Lower bound on the Total Profit KPI of a feasible solution. Use '-inf' to represent negative infinity, and 'inf' to represent infinity.

**Min Total Demand Satisfied:** Lower bound on the Total Demand Satisfied KPI of a feasible solution.

**Zone For Critical Sites:** Used if the Objective is 'Critical Sites Opened'. Any site within this zone is a critical site.

**High Service Demand Threshold:** Final tier shipping distances that don't exceed this distance qualify as High Service. Non-negative, non-infinite number. This parameter can work together with the parameter Min Percentage High Service Demand. For example, if equals to 500, and Min Percentage High Service Demand = 80, 80% of the demand will be attended within the distance of 500 miles (assuming that Distance Type = Miles). Take a look at the [Modeling High Service Demand](#) and [Parameters](#) pages to understand more about it.

**Ending Inventory Constraint Type:** Describes what sort of constraint is applied by the Ending Inventory column in the [Site Product Inv Info](#) table.

**Generate Errors and Warnings:** Set to 'Instead Of Optimization' to return errors and warnings after a comprehensive sanity check of input data but without generating a solution.

Possible values: 'Instead Of Optimization' or 'Always"'.

Take a look at the [Parameters](#) page to understand more about this parameter.

**Empty Zones Or Groupings:** How should solver handle empty zones/groupings.  
Possible values: 'Remove With Warnings', 'Remove Without Warnings' or 'Exit With Error'.

Take a look at the [Parameters](#) page to understand more about this parameter.

**Bottleneck Tolerance:** The threshold by which constraint utilization is screened in order to go into the bottlenecks report. For example, if the user considers that 80% of production utilization is already an alert for a bottleneck situation, this field needs to be filled with the value of 80.

**Generate Bottlenecks:** Sets if the model should or not generate the Bottlenecks report. Possible values: 'Never', 'After Optimization', or 'Instead Of Optimization'.

**Open Sites Report Zone:** The Zone used to populate the Zone field on the Open Sites report. 'Zone 1' through Zone 10 allowed.

**Generate Landed Cost Report:** Parameter that sets if the model should or not generate Landed Cost Report. Possible values: 'Always', 'Never' or 'Instead Of Optimization'.

**Generate Transportation Details Report:** 'Always' or 'Never'.

**Generate All Possible Arcs Report:** Parameter that sets if the model should or not generate All Possible Arcs report. Possible values: 'After Optimization', 'Instead Of Optimization' or 'Never'.

**Distance Factor:** Default scaling factor used to adjust Haversine result.

**Distance Type:** Miles or Kilometers

**In Transit Inventory:** This parameter controls whether shipments will always start and end in the same time period. If so, then use 'Used For Holding Costs Only'. Otherwise, use 'Allowed Between Periods'. In other words, this parameter defines if the product can be shipped in one time period and arrive in the next one, or not.

**SMC Username:** Needed if your model uses the SMC Carrier.

**SMC License:** Needed if your model uses the SMC Carrier.

**SMC Password:** If you append '\_ENCRYPT\_ME' to this value, and then upload this table using Google Sheets then an encrypted version of the password will be stored. It is also encrypted if '\_ENCRYPT\_ME' is at the end of the value uploaded from CSV or Excel.

**Maximum CO2:** Upper bound on the amount of carbon that can be used without using purchasing carbon.

**Carbon Purchase Price:** Cost (per unit of carbon) to exceed the Maximum CO2 constraint.

**Carbon Sale Price:** Revenue (per unit of carbon) that is realized from slack in the Maximum CO2 constraint.

#### Sample Input

Key	Value	Category	Description
Zone For Critical Sites	*all*	Optimization	Used if the Objective is 'Critical Sites Opened'. Any site within this zone is a critical site.
Min Total Demand Satisfied	0	Optimization	Lower bound on the Total Demand Satisfied KPI of a feasible solution
Bottleneck Tolerance	Never	Reports	Default scaling factor used to adjust Haversine result.

## Demand and Production

### DEMAND

Demand can be associated with any site.

#### Site

*Type: Text*

Must match a valid site.

#### Product

*Type: Text*

Must match a valid product.

#### Time Period

*Type: Text*

Must match a valid time period.

#### Min Demand

*Type: Numeric*

The minimum amount of demand that a feasible solution must satisfy. Non-negative, non-infinite.

#### Max Demand

*Type: Numeric*

The maximum amount of demand that a feasible solution must satisfy. Non-negative, non-infinite. Null can be entered to flag Max Demand the same as Min Demand.



If the parameter **Objective**, in Parameters table, is set as **Total Infeasibility** (see [Optimization Objectives](#) page to understand more about Objectives configuration), set the Maximum Demand = Minimum Demand, and then reset the Minimum Demand field to be 0, to run infeasibility diagnostics.

#### Revenue

*Type: Numeric*

Per SKU revenue realized for satisfying demand. Non-negative, non-infinite.

#### Landed Cost X-Ray

*Type: Text*

'Enabled' or 'Disabled'

Enabled is needed for Landed Cost Tree visualization.

If Enabled then a visualization tree will be built, which can be accessed from the Landed Cost Report. If you change only this value, and then solve with Generate Landed Cost Report parameter set as Instead Of Optimization, then the solve will recycle the current solution to more quickly create the needed visualization trees.

#### Sample Input

Site	Product	Time Period	Min Demand	Max Demand	Revenue	Landed Ray
C_Pittsburgh	Bread	P1	700	1000	35	Disable
C_Newyork	Cookie	P1	1200	1600	20	Enabled
C_Albany	Chips	P1	4000	5000	15.5	Enabled

## PRODUCTION

Use this table to set up production features at the site, product and time period level.



To configure production features at the production line, site, product and time period level, it's recommended to use [Production Lines](#) and [Production By Production Lines](#) table.



This table can be used to set up procurement features as well, as Suppliers can be understood as Production Sites in the model.

### Site

*Type: Text*

Must match a valid site.

### Product

*Type: Text*

Must match a valid product.

### Time Period

*Type: Text*

Must match a valid time period.

### Min Supply

*Type: Numeric*

The minimum amount of supply that a feasible solution can use. Non-negative, non-infinite. Min is conditional on the production line being opened.

### Max Supply

*Type: Numeric*

The maximum amount of supply that a feasible solution can use. Non-negative, infinity allowed.

### Cost

*Type: Numeric*

Per SKU cost of production. Non-negative, non-infinite.

### Rate

*Type: Numeric*

Number of SKUs that can be produced with one aggregate production hour. Positive, infinity allowed.

### Tiered Cost

*Type: Text*

Can be NULL, or an entry from the Tiered Costs table.

### Sample Input

Site	Product	Time Period	Min Supply	Max Supply	Cost	Rate
P_Dover	Bread	P1	7000	10000	10	100
P_Dover	Cookie	P1	10000	16000	7	75
P_Dover	Chips	P1	10000	50000	5	200

## AGGREGATED PRODUCTION

The constraints created by this table apply to the production defined by the Production table. It doesn't constrain production line based production.



This table constrain the production based on production hours, instead of production volumes.

### Site

*Type: Text*

Must match a valid site.

### Product Group

*Type: Text*

Must match a valid product group.

### Time Period

*Type: Text*

Must match a valid time period.

### Min Hours

*Type: Numeric*

The minimum amount of aggregate production hours that a feasible solution can use. Non-negative, non-infinite. Min is conditional on the production line being opened.

### Max Hours

*Type: Numeric*

The maximum amount of aggregate production hours that a feasible solution can use. Non-negative, infinity allowed.

### Tiered Cost

*Type: Text*

Can be NULL, or an entry from the Tiered Costs table.

### Sample Input

Site	Product Group	Time Period	Min Hours	Max Hours	Tiered Cost
P_Dover	Bread	P1	7000	10000	Tier A
P_Dover	Cookie	P1	10000	16000	Tier B
P_Dover	Chips	P1	10000	50000	

**TIERED COSTS**

This table enables tiered production pricing.

**Note**

This table can be used to model purchase discounts too. In the Cost field, an Incremental Discount situation is when Cost = Marginal, and an All-Units Discount case corresponds to Cost = Simple.

## Name

*Type: Text*

Identifies the tiered cost. Will be repeated for all relevant records.

## Maximum Quantity

*Type: Numeric*

The upper limit (inclusive) of a quantity band. Positive, infinity allowed.

## Cost

*Type: Numeric*

`Marginal` : the per-unit-type cost incurred by production within this band. Each cost would be applied for the quantities that are within a band. For example, based on the sample input below, if the quantity produced in a time period is equal to 120000, the total production cost would be  $50000 * 25 + (100000 - 50000) * 22 + (120000 - 100000) * 20 = 2750000$ .

`Simple` : the per-unit-type cost incurred by all production if total production lies within this band.

Non-negative, non-infinite. In this case, the cost would be applied for all units, even for quantities that are not within the tier band. For example, based on the sample input below, if the quantity produced in a time period is equal to 120000, the total production cost would be  $120000 * 20 = 2400000$ .

## Unit Type

*Type: Text*

One of 'SKU', 'Weight', 'Shipment Volume', 'Adjusted Volume', 'Warehouse Volume', 'Hours'. All rows with same Name must have the same value for Unit Type.

## Tier Type

*Type: Text*

Either `Marginal` or `Simple`.

If `Simple`, please note that the total cost might drop when moving from one band to the next. All rows with same Name must have the same value for Tier Type.

## Sample Input

Name	Maximum Quantity	Cost	Unit Type	Tier Type
Tier A	50000	25	SKU	Marginal
Tier A	100000	22	SKU	Marginal
Tier A	999999999	20	SKU	Marginal

#### PRODUCTION BY PRODUCTION LINES

Use this table if you want production to occur at a production line (within a site). All production lines features can be set up in this table, and which products can be produced on those.



To use this table, the Production Lines table needs to be populated with the same production lines that are being configured here.

##### Site

*Type: Text*

Must match a valid site.

##### Production Line

*Type: Text*

Must match a valid production line.

##### Product

*Type: Text*

Must match a valid product.

##### Time Period

*Type: Text*

Must match a valid time period.

##### Min Supply

*Type: Numeric*

The minimum amount of supply that a feasible solution can use. Non-negative, non-infinite. Min is conditional on the production line being opened.

##### Max Supply

*Type: Numeric*

The maximum amount of supply that a feasible solution can use. Non-negative, infinity allowed.

##### Cost

*Type: Numeric*

Per SKU cost of production. Non-negative, non-infinite.

##### Rate

*Type: Numeric*

Number of SKUs that can be produced with one aggregate production hour. Positive, infinity allowed.

##### Sample Input

Site	Production Line	Product	Time Period	Min Supply	Max Supply	Cost
C_Pittsburgh	Line A	Bread	P1	7000	10000	10
C_Newyork	Line B	Cookie	P1	10000	16000	7
C_Albany	Line C	Chips	P1	10000	50000	5

#### AGGREGATED PRODUCTION BY PRODUCTION LINES

This table constrains production line based production.



This table constrain the production by production lines based on production hours, instead of production volumes.

##### Site

*Type: Text*

Must match a valid site.

##### Production Line

*Type: Text*

Must match a valid production line.

##### Product Group

*Type: Text*

Must match a valid product group.

##### Time Period

*Type: Text*

Must match a valid time period.

##### Min Hours

*Type: Numeric*

The minimum amount of aggregate production hours that a feasible solution can use. Non-negative, non-infinite. Min is conditional on the production line being opened.

##### Max Hours

*Type: Numeric*

The maximum amount of aggregate production hours that a feasible solution can use. Non-negative, infinity allowed.

##### Sample Input

Site	Production Lines	Product Group	Time Period	Min Hours	Max Hours
P_Dover	Line A	Bread	P1	7000	10000
P_Dover	Line B	Cookie	P1	10000	16000
P_Dover	Line C	Chips	P1	10000	50000

**BOM (BILL OF MATERIALS)**

Populate this table for simple Bill Of Materials based production. Use the Recipe Bill Of Materials instead if you need to model different BOM at different sites.

**Component**

*Type: Text*

Must match a valid product. The component is the product which is used to produce the Assembly.

**Assembly**

*Type: Text*

Must match a valid product. The assembly is the production output of the components that are associated with it.

**Quantity**

*Type: Numeric*

The amount of the component product needed to produce one unit of the assembly product. Positive, non-infinite.

**Sample Input**

In the example below, to produce 1 unit of Bread it's required 2 units of Dough.

Component	Assembly	Quantity
Dough	Bread	2
Dough	Cookie	1.5
Potato	Chips	4

**RECIPES**

Populate this table if you want to use the Recipe Bill Of Materials table.

**Name**

*Type: Text*

Identifies the recipe. Will also create a singleton recipe grouping.

**Grouping X**

*Type: Text*

Use this column to define a recipe grouping. Don't define singleton or all grouping. There are 10 grouping columns.

**Sample Input**

<b>Name</b>	<b>Grouping 1</b>
Bread Recipe 1	
Cookie Recipe 1	
Chips Recipe 1	

**RECIPE BOM (BILLS OF MATERIALS)**

Use this table if you need to model different BOM at different sites.

**Recipe**

*Type: Text*

Must match a valid recipe.

**Component**

*Type: Text*

Must match a valid product. The component is the product which is used to produce the Assembly.

**Assembly**

*Type: Text*

Must match a valid product. The assembly is the production output of the components that are associated with it.

**Quantity**

*Type: Numeric*

The amount of the component product needed to produce one unit of the assembly product. Positive, non-infinite.

**Sample Input**

Recipe	Component	Assembly	Quantity
Bread Recipe 1	Dough	Bread	2
Cookie Recipe 1	Dough	Cookie	1.5
Chips Recipe 1	Potato	Chips	4

**RECIPE BYPRODUCT**

Populate this table if you want to use the Recipe Bill Of Materials table **by-products**.

**Recipe**

*Type: Text*

Must match a valid [Recipe](#).

**Assembly**

*Type: Text*

Must match a valid [Product](#).

**Byproduct**

*Type: Text*

Must match a valid [Product](#).

**Quantity**

*Type: Numeric*

The amount of the byproduct that needs to be co-produced with one unit of the assembly product when the production is using this recipe. Positive, non-infinite.

**Byproduct Disposal Cost**

*Type: Numeric*

The cost per unit of immediately disposing the byproduct, when the production is using this recipe. Positive, infinity allowed.

**Sample Input**

Recipe	Assembly	Byproduct	Quantity	Byproduct Disposal Cost
Bread Recipe 1	Bread	Bread Crust	0.05	0.01
Cookie Recipe 1	Cookie	Trimming	0.03	0.02
Chips Recipe 1	Chips	Potato Peel	0.07	0.04

#### PRODUCTION LINE RECIPE PROHIBITIONS

By default all recipes are available at all production lines. Use this table to restrict recipes to certain production lines.

The (Production Line Group, Recipe Group) pairs specified here will prohibit the usage of specific recipes at specific production lines.

Otherwise, all production lines can use all recipes.

##### Production Line Group

*Type: Text*

Must match a valid group or production line defined in the Production Lines table.

##### Recipe Group

*Type: Text*

Must match a valid recipe group.

##### Sample Input

Production Line Group	Recipe Group
Line A	Bread Recipe 1
Line B	Cookie Recipe 1
P_Memphis Lines	Bread Recipes

**ZONE RECIPE PROHIBITIONS**

By default all recipes are available at all sites. Use this table to restrict recipes to certain sites. Does *NOT* restrict production line based production.

The (Zone Name, Recipe Group) pairs specified here will prohibit the usage of specific recipes at specific sites.

Otherwise, all sites can use all recipes.

**Note**

Note that this table is used to inform to the model what can NOT be done, instead of what CAN be done. At the [Production](#) table, for example, you will set up what products can be produced at each site. At the Zone Recipe Prohibitions, however, it is the inverse. You will set up what recipes can NOT be used at the specified zones.

**Zone Name**

*Type: Text*

Must match a valid zone.

**Recipe Group**

*Type: Text*

Must match a valid recipe group.

**Sample Input**

Zone Name	Recipe Group
P_Dover	Bread Recipe 1
P_Dover	Cookie Recipe 1
Warehouse	Bread Recipes

## Transportation

### ARC OVERRIDES

Use the Arc Overrides table as a more direct alternative to lanes and carriers tables.

#### Source

*Type: Text*

Must match a valid site.

#### Destination

*Type: Text*

Must match a valid site.

#### Product

*Type: Text*

Must match a valid product.

#### Cost

*Type: Numeric*

Per SKU cost of shipment. Non-negative, non-infinite.

#### Sample Input

Source	Destination	Product	Cost
P_Dover	C_Alban	Bread	2
W_Memphis	C_Atlanta	Cookie	1.23
W_Memphis	C_Nashville	Chips	0.78

**LANES**

Use lanes and carriers to generate arcs based on zones and product groupings.

**Name**

*Type: Text*

Identifies the lane. Also used to break ties with Precedence.

**Precedence**

*Type: Integer*

The Lanes table is processed as ordered by the Precedence field, with ties broken by Name.

For example, a record with Precedence 1 will be processed before a record with Precedence 2. This means that the lane with Precedence 1 will be *Prioritized* regarding the lane with Precedence 2. Take a look at the [Sample Input](#) section to understand more how this field works.

**Source Zone Name**

*Type: Text*

Must match a valid zone.

**Destination Zone Name**

*Type: Text*

Must match a valid zone.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Transit Time**

*Type: Numeric*

Time required for a shipment that uses an arc generated by this lane. Non-negative non-infinite.

**Holding Cost**

*Type: Numeric*

Per-sku, per-time cost for a shipment that uses an arc generated by this lane. Non-negative non-infinite.

**Carrier**

*Type: Text*

Must specify a valid carrier.

**Max Distance**

*Type: Numeric*

The maximum distance of an arc generated by this lane. Non-negative. Infinity allowed.

**Sample Input**

In the example below, for the product Bread, it is preferred to satisfy the demand directly by the production site to the customer at Albany, that's why Precedence = 1 for the lane with origin in Dover and Precedence = 2 for the one with origin at Memphis warehouse. If you run the model with the objective [Parameter](#) set as Total

Cost, and both lanes have *EQUAL COSTS*, the model will choose the lane with the *HIGHER* precedence number to flow products, in that case, the lane with Precedence 1, with origin at Dover plant.

<b>Name</b>	<b>Precedence</b>	<b>Source Zone Name</b>	<b>Destination Zone Name</b>	<b>Product Group</b>	...
Dover-Albany_Bread	2	P_Dover	C_Albanym	Bread	...
Memphis-Albany_Bread	1	W_Memphis	C_Albany	Bread	...
Memphis-Customers_Cookie		W_Memphis	Customer	Cookie	...

**BLENDED CARRIER**

This carrier rates by discovering the shipping costs of other carriers, and applying a weighted average of all the rates.

For example, this table can be used when a destination is supplied by multiple carriers by proportion.

## Carrier

*Type: Text*

Identifies the carrier. The carrier name will be repeated for all relevant records.

## Source Zone Name

*Type: Text*

Must match a valid zone.

## Destination Zone Name

*Type: Text*

Must match a valid zone.

## Product Group

*Type: Text*

Must match a valid product grouping.

## Blending In Carrier

*Type: Text*

Must specify a valid carrier. Here the user needs to populate an existing carrier that will compose the blended carrier defined at the Carrier field.

## Blend Weight

*Type: Numeric*

The relative proportion when performing a weighted average of each Blending In Carrier. In another words, this is the weight that the Blending In Carrier has on the blended carrier defined at the Carrier field.

## Avg Shipment

*Type: Numeric*

If an average shipment is provided here, it takes precedence over the Arc Shipment Data table.

This carrier doesn't use average shipment directly, but can pass this value on to the Blending In Carrier

Positive, non-infinite nullable.

## Sample Input

Carrier	Source Zone Name	Destination Zone Name	Product Group	Blending In Carrier	Blend Weight	Avg Shipment
Blended_Carrier	P_Dover	C_Albany	Bread	FTL	0.7	
Blended_Carrier	P_Dover	C_Albany	Bread	LTL	0.3	

**CHEAPEST CARRIER**

This carrier rates by discovering the shipping costs of other carriers, and applying the cheapest rate. Use this table to select a list of carriers, and from that list the model will calculate the cheapest carrier.

## Carrier

*Type: Text*

Identifies the carrier. The carrier name will be repeated for all relevant records.

## Source Zone Name

*Type: Text*

Must match a valid zone.

## Destination Zone Name

*Type: Text*

Must match a valid zone.

## Product Group

*Type: Text*

Must match a valid product grouping.

## Cheapest In Carrier

*Type: Text*

Must specify a valid carrier. For a given source, destination, product triplet, the cheapest carrier will be used.

## Avg Shipment

*Type: Numeric*

If an average shipment is provided here, it takes precedence over the Arc Shipment Data table.

This carrier doesn't use average shipment directly, but can pass this value on to the Cheapest In Carrier

Positive, non-infinite nullable.

## Sample Input

Carrier	Source Zone Name	Destination Zone Name	Product Group	Cheapest In Carrier	Avg Shipment
Cheapest_Road_Carrier	P_Dover	C_Albany	Bread	FTL	
Cheapest_Railroad_Carrier	W_Memphis	C_Atlanta	Cookie	Rail_Carrier	
Cheapest_Railroad_Carrier	W_Memphis	C_Nashville	Chips	Rail_Carrier	

**FEDEX CARRIER**

Use this carrier to perform ratings based on FedEx shipping costs. This carrier not yet implemented.

**Carrier**

*Type: Text*

Identifies the carrier. The carrier name will be repeated for all relevant records.

**Source Zone Name**

*Type: Text*

Must match a valid zone.

**Destination Zone Name**

*Type: Text*

Must match a valid zone.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Service Type**

*Type: Text*

Must be one of the following. Standard Overnight, Priority Overnight, Ground, First Overnight, 2 Day, 2 Day AM, Express Saver

**Package Weight**

*Type: Numeric*

Positive, non-infinite.

**Sample Input**

Carrier	Source Zone Name	Destination Zone Name	Product Group	Service Type	Package Weight
Fedex_Carrier	P_Dover	C_Alban	Bread	Standard Overnight	0.4
Fedex_Carrier	W_Memphis	C_Atlanta	Cookie	Priority Overnight	0.32
Fedex_Carrier	W_Memphis	C_Nashville	Chips	2 Day	0.63

**SMC CARRIER**

Use this carrier to perform ratings based on SMC3.

**Carrier**

*Type: Text*

Identifies the carrier. The carrier name will be repeated for all relevant records.

**Source Zone Name**

*Type: Text*

Must match a valid zone.

**Destination Zone Name**

*Type: Text*

Must match a valid zone.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Tariff Name**

*Type: Text*

Name of the Tariff the SMC rater should use to rate the shipment.

**Tariff Date**

*Type: Text*

The date the SMC rater should use to rate the shipment.

Only the year is needed, but any string that looks like a date will be understood.

**Percent Discount**

*Type: Numeric*

Percent by which the rating cost should be reduced. Number between 0 (inclusive) and 100 (exclusive).

**Avg Shipment**

*Type: Numeric*

Weight of the average shipment. Non-negative, non-infinite (Null is allowed). See Avg Shipment documentation for more information.

**Min Charge**

*Type: Numeric*

Minimum possible shipping cost.

Non-negative, non-infinite.

This is passed to the userMinimumChargeFloor component of the ltrateshipment payload.

**Sample Input**

Carrier	Source Zone Name	Destination Zone Name	Product Group	Tariff Name	Tariff Date	Percent
SMC	P_Dover	C_Albany	Bread	SMC Tariff	11/5/22	5
SMC	W_Memphis	C_Atlanta	Chips	SMC Tariff	11/5/22	10
SMC	W_Memphis	C_Nashville	Cookie	SMC Tariff	11/5/22	7

**TRUCK CARRIER**

The truck carrier is similar to the zone carrier, except rated based on truck capacity. Consult the [Average Shipment Logic](#) section of the documentation for more details.

**Carrier**

*Type: Text*

Identifies the carrier. The carrier name will be repeated for all relevant records.

**Source Zone Name**

*Type: Text*

Must match a valid zone.

**Destination Zone Name**

*Type: Text*

Must match a valid zone.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Unit Type**

*Type: Text*

The unit of measurement for the Truck Size fields.

Must be one of the following.

'SKU', 'Weight', 'Shipment Volume', 'Adjusted Volume'

**Cost Per Truck**

*Type: Numeric*

The cost per truck used. This adds to the Cost Per Truck Distance calculation.

**Cost Per Truck Distance**

*Type: Numeric*

The cost per truck used per distance shipped. This adds to the Cost Per Truck calculation.

**Truck Size**

*Type: Numeric*

Size of the truck, measured in Unit Type. Positive, non-infinite.

**Avg Shipment**

*Type: Numeric*

If an average shipment is provided here, it takes precedence over any other designation.

Weight of the average shipment. Non-negative, non-infinite (Null is allowed). See Avg Shipment documentation for more information.

If no average shipment can be resolved anywhere, than we assume trucks are fully utilized.

**Min Charge**

*Type: Numeric*

Minimum charge **per truck**

## Sample Input

<b>Carrier</b>	<b>Source Zone Name</b>	<b>Destination Zone Name</b>	<b>Product Group</b>	<b>Unit Type</b>	<b>Cost Per Truck</b>	<b>Cost Per Distance</b>
Truck_Carrier	P_Dover	C_Albany	Bread	Weight		1.70
Truck_Carrier	W_Memphis	C_Atlanta	Cookie	Weight	1000	
Truck_Carrier	W_Memphis	C_Nashville	Chips	Weight	1100	1.50

**UPS CARRIER**

Use this carrier to perform ratings based on UPS shipping costs.

**Carrier**

*Type: Text*

Identifies the carrier. The carrier name will be repeated for all relevant records.

**Source Zone Name**

*Type: Text*

Must match a valid zone.

**Destination Zone Name**

*Type: Text*

Must match a valid zone.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Service Code**

*Type: Text*

Must be one of the following. "Ground", "Three Day Select", "Three Day Air", "Second Day Air", "Second Day Air AM", "Next Day Air Saver", "Next Day Air"

**Percent Discount**

*Type: Numeric*

Percent by which the rating cost should be reduced. Number between 0 (inclusive) and 100 (exclusive).

**Package Weight**

*Type: Numeric*

Integer between 1 and 150, inclusive.

**Sample Input**

Carrier	Source Zone Name	Destination Zone Name	Product Group	Service Code	Percent Discount	Package Weight
P_Dover	C_Alban	Bread	2	C_Alban	Bread	Bread
W_Memphis	C_Atlanta	Cookie	1.23	C_Alban	Bread	Bread
W_Memphis	C_Nashville	Chips	0.78	C_Alban	Bread	Bread

**ZONE CARRIER**

Simplest carrier. Use this table to set transportation costs by units of products.

**Carrier**

*Type: Text*

Identifies the carrier. The carrier name will be repeated for all relevant records.

**Source Zone Name**

*Type: Text*

Must match a valid zone.

**Destination Zone Name**

*Type: Text*

Must match a valid zone.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Unit Type**

*Type: Text*

The unit of measurement for Truck utilization.

Must be one of the following.

'SKU', 'Weight', 'Shipment Volume','Adjusted Volume'

**Cost Per Unit**

*Type: Numeric*

The cost per unit shipped. This adds to the Cost Per Unit Distance calculation.

**Cost Per Unit Distance**

*Type: Numeric*

The cost per unit per distance shipped. This adds to the Cost Per Unit calculation.

**Sample Input**

Carrier	Source Zone Name	Destination Zone Name	Product Group	Unit Type	Cost Per Unit
Direct_Shipment_Carrier	P_Dover	C_Albany	Bread	Weight	5.23
Warehouse_Shipment_Carrier	W_Memphis	C_Albany	Bread	Weight	4.64
Warehouse_Shipment_Carrier	W_Memphis	C_Nashville	Cookie	Weight	

**DISTANCE WEIGHT BREAK CARRIER COST**

Allows you to rate based on distance and average shipment. Allows a user to manually enter information similar to that used by SMC3 ratings.

**Carrier**

*Type: Text*

Identifies the carrier. The carrier name will be repeated for all relevant records.

**Source Zone Name**

*Type: Text*

Must match a valid zone.

**Destination Zone Name**

*Type: Text*

Must match a valid zone.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Distance Break**

*Type: Numeric*

Specifies a distance band. Non-negative, infinity allowed.

**Weight Break**

*Type: Numeric*

Specifies a weight band. Non-negative, infinity allowed.

The average shipment is used to determine the weight break. See Avg Shipment documentation for more information.

**Cost Per Weight**

*Type: Numeric*

The cost per weight shipped. This adds to the Cost Per Weight Distance calculation.

**Cost Per Weight Distance**

*Type: Numeric*

The cost per weight per distance shipped. This adds to the Cost Per Weight calculation.

**Sample Input**

Carrier	Source Zone Name	Destination Zone Name	Product Group	Distance Break	Weight Break	Cost Per
Rail_Carrier	P_Dover	Customer	Bread	300	40000	0.23
Truck_Carrier	W_Memphis	Customer	Cookie	250	45000	
Truck_Carrier	Plant	Warehouse	Chips	600	38500	0.19

#### CARRIER SPECIFIC FAILURE LOGGING

Use the Carrier Specific Logging table to control the population of the Carrier Specific advanced validation tables. These tables will only be populated if the 'Generate Errors and Warnings' parameter is set to 'Instead of Optimization'.

##### Lane

*Type: Text*

Must match a valid lane.

##### Carrier

*Type: Text*

Must match a valid carrier.

##### Detail

*Type: Text*

Must be one of the following:

```
Shipment Weight Not Found
No Matching Distance/Weight Break
At Least One Blend Carrier Failed to Rate
All the Cheapest Carriers Failed to Rate
No Matching Zone/Zone/Group Record
```

##### Logging Level

Must be one of the following:

```
Itemized
Summarized
Ignored
```

##### Sample Input

Lane	Carrier	Detail	Logging Level
Dover-Albany_Bread	Direct_Shipment_Carrier	Shipment Weight Not Found	Summarized
Memphis-Albany_Bread	Warehouse_Shipment_Carrier	All the Cheapest Carriers Failed to Rate	Itemized
Memphis-Atlanta_Cookie	Warehouse_Shipment_Carrier	At Least One Blend Carrier Failed to Rate	Ignored

**ARC SHIPMENT DATA**

Consult the [Average Shipment Logic](#) section FAQs of the documentation to understand more about this table.

**Destination Site**

*Type: Text*

Must match a valid site.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Avg Shipment SKU**

*Type: Numeric*

If non-null, then the average shipment size measured in SKU. Non-negative, non-infinite. Null is allowed.

**Avg Shipment Weight**

*Type: Numeric*

If non-null, then the average shipment size measured in weight. Non-negative, non-infinite. Null is allowed.

**Avg Shipment Volume**

*Type: Numeric*

If non-null, then the average shipment size measured in product shipment volume. Non-negative, non-infinite. Null is allowed.

**Avg Shipment Adjusted Volume**

*Type: Numeric*

If non-null, then the average shipment size measured in product shipment adjusted volume. Non-negative, non-infinite. Null is allowed.

**Sample Input**

Destination Site	Product Group	Avg Shipment SKU	Avg Shipment Weight	Avg Shipment Volume	Avg Shipment Adjusted Volume
C_Albany	Bread	500	2500	10000	10156
C_Atlanta	Cookie	750	4000	20000	19978
C_Nashville	Chips	300	3200	15000	15040

**ZONE GROUP SHIPPING BOUNDS**

Use this table to restrict the shipping distances that are allowed, regardless of lane and carrier. For distance restrictions specific to lane and/or carrier, use the Max Distance column on the lanes form.

**Zone Name**

*Type: Text*

Must match a valid zone.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Max Inbound Distance**

*Type: Numeric*

Non-negative, infinity allowed.

No inbound arcs larger than this distance will be generated.

**Max Outbound Distance**

*Type: Numeric*

Non-negative, infinity allowed.

No outbound arcs larger than this distance will be generated.

**High Service Distance**

*Type: Numeric*

Non-negative, infinity not allowed.

Any inbound shipments this distance or smaller qualifies as high service. This field works in the same way as the parameter High Service Demand Threshold, in the Parameters table.

**Min Percentage Inbound High Service**

*Type: Numeric*

Between zero and 100, inclusive.

Lower bound on the percentage of inbound shipments that qualify as high service. This field works in the same way as the parameter Min Percentage High Service Demand, in the Parameters table.

**Sample Input**

Zone Name	Product Group	Max Inbound Distance	Max Outbound Distance	High Service Demand	Min Percentage Inbound High Service
Customer	Bread	300		150	80
Warehouse	Cookie	500	150	200	50

## MIN MAX FLOWS

This table is based on zones and product groupings. A wide range of shipping based requirements can be modeled with this table. Use this table to constrain the outbound and inbound volumes of a warehouse, for example. Please refer to [Flow Constraints](#) for examples on how to use this table.

### Source Zone Name

*Type: Text*

Must match a valid zone.

If a non-singleton zone is used, the constraint is set for the aggregate shipment across all the sites in the zone, **not** for each site in the zone.

### Destination Zone Name

*Type: Text*

Must match a valid zone.

If a non-singleton zone is used, the constraint is set for the aggregate shipment across all the sites in the zone, **not** for each site in the zone.

### Product Group

*Type: Text*

Must match a valid product grouping.

If a non-singleton grouping is used, the constraint is set for the aggregate shipment across all the products in the grouping, **not** for each product in the grouping.

### Time Period Group

*Type: Text*

Must match a valid time period grouping.

If a non-singleton grouping is used, the constraint is set for the aggregate shipment across all the time periods in the grouping, **not** for each time periods in the grouping.

### Active

*Type: Boolean*

Used for Including or Excluding the site from Optimization.

True

False



Beware. Any string other than 'True' or 'true' will deactivate this record. The model does not accept binary values of 0 and 1 as False and True.

### Min Flow

*Type: Numeric*

Minimum amount of flow that must be met by a feasible solution. This is a true min, not a conditional min.

. Non-negative, non-infinite.

### Binary Min

*Type: Numeric*

Flow quantity between zero and the Binary Min are prohibited.

### Max Flow

*Type: Numeric*

Maximum amount of flow that must be met by a feasible solution. Non-negative. Infinity allowed.

### Max Number Source Sites

*Type: Numeric*

Upper bound on the number of Source sites.

### Max Number Destination Sites

*Type: Numeric*

Upper bound on the number of Destination sites.

### Min Percentage Supplier

*Type: Numeric*

Lower bound on the percentage of flow relative to a row with `*all*` as source zone

### Max Percentage Supplier

*Type: Numeric*

Upper bound on the percentage of flow relative to a row with `*all*` as source zone

### Unit Type

*Type: Text*

If something other than SKU, then the cost is scaled by the appropriate value from the products table.

Possible values are:

SKU

Weight

Shipment Volume

Adjusted Volume

### Notes

*Type: Text*

Descriptive Column to provide additional information about the Site.

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	Active	Min Flow	Binary
W_Memphis	Customer	Bread	P1	True	30000	
W_El Paso	Customer	Cookie	P2	True		20000
*all*	DC_Chicago	Chips	P3	True		

## DISTANCE OVERRIDES

By default, distances are computed using the Haversine formula, adjusted by the Distance Factor parameter. Use this table if you want to override this behavior.

### Source

*Type: Text*

Must match a valid site.

### Destination

*Type: Text*

Must match a valid site.

### Distance

*Type: Numeric*

If provided, the distance between the source and destination. Non-negative, non-infinite. Null is allowed.

If non-null, then the Haversine calculation will **not** be used.

### Distance Factor

*Type: Numeric*

Scaling factor used to adjust Haversine result. Non-negative, non-infinite. Will be ignored if Distance is non-null.

### Sample Input

Source	Destination	Distance	Distance Factor
P_Dover	C_Albany	307	
W_Memphis	C_Atlanta	391	
W_Memphis	C_Nashville	212	

**SIMPLE DUTY TARIFF**

Populate this table to assess a duty tariff transit cost that does not require production zone tracking. In other words, use this table when it is desired to consider tariffs from A to B, for specific products.



This cost will not be applied if a shipping arc is not generated from lanes and carriers, or arc overrides.

**Source**

*Type: Text*

Must match a valid [Site](#).

**Destination**

*Type: Text*

Must match a valid [Site](#).

**Product**

*Type: Text*

Must match a valid [Product](#).

**Value**

*Type: Numeric*

The assessed duty tariff cost will be this value multiplied by the Duty Tariff Percentage and divided by 100. In other words, this is the value of one product unit, but will be used ONLY for duty tariff costs purposes.

**Duty Tariff Percentage**

*Type: Numeric*

The assessed duty tariff cost will be the Value column multiplied by this percentage and divided by 100.

**Sample Input**

At the example below, to import one unit of the product Smartphone, from Shanghai to Los Angeles, it is charged 10% of the product value of \$ 800 as duty tariffs. So the Duty Tariff cost would be \$ 80 in this case.

Source	Destination	Product	Value	Duty Tariff Percentage
Port_Shanghai	Port_Los Angeles	Smartphone	800	10
Port_Guangzhou	Port_Long Beach	Television	400	15

#### ADVANCED DUTY TARIFF

Populate this table to assess a duty tariff transit cost that requires production zone tracking. In other words, refer to this table when you need to account for duty tariffs applied to product groups transported from A to B to C. Zone A represents the production zone, while Zone B (Source Zone) is the final destination before reaching its ultimate destination (Zone C).

If there are stops at other locations between A to B (or B to C), and you want to specify duty tariffs between those points, for example, take a look at the [Simple Duty Tariff](#) table to model duty tariff costs for each lane.



If you want, you can populate the Zones (Production, Source and Destination) and Product Group columns with specific sites or products, respectively.



Production zone tracking might incur a significant memory overhead.

##### Production Zone Name

*Type: Text*

Must match a valid zone.

##### Source Zone Name

*Type: Text*

Must match a valid zone.

##### Destination Zone Name

*Type: Text*

Must match a valid zone.

##### Product Group

*Type: Numeric*

Must match a valid product grouping.

##### Value

*Type: Numeric*

The assessed duty tariff cost will be this value multiplied by the Duty Tariff Percentage and divided by 100. In other words, this is the value of one product unit, but will be used ONLY for duty tariff costs purposes.

##### Duty Tariff Percentage

*Type: Numeric*

The assessed duty tariff cost will be the Value column multiplied by this percentage and divided by 100.

##### Sample Input

At the example below, to import one tonne of the product Soybean, produced at Brazil zone and transported by the East Coast zone to get to the West Coast zone, it is charged 5% of the product value of \$400 as duty tariffs. So the Duty Tariff cost would be \$20 by unit, in this case.

Production Source Zone	Source Zone Name	Destination Zone Name	Product Group	Value	Duty Tariff Percentage
Brazil	East Coast	West Coast	Soybean	400	5
Mexico	Texas	Illinois	Strawberry	2500	2

**LANE DESTINATION ASSIGNMENTS**

Use this table to restrict the possible suppliers for a given site.

**Destination**

*Type: Text*

Must match a valid site.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Source**

*Type: Text*

Must match a valid site.

Every site product pair from (Destination, Product Group) will only be receiving inbound shipments from this source.

**Sample Input**

<b>Destination</b>	<b>Product Group</b>	<b>Source</b>
C_Albany	Bread	P_Dover
C_Atlanta	Cookie	W_Memphis
C_Nashville	Chips	W_El Paso

#### EXPECTED EMPTY ZONES GROUPINGS

Populate this table to disable the errors or warnings associated with zones or groupings that are known to be empty.



Modeling best practice is to set the Empty Zones or Groupings parameter to Exit With Error and then populate this table with the specific zones and groupings that are known to be empty. Checkout the [Best Modeling Practices](#) page to learn more about how to use this table.

Table Name

*Type: Text*

One of Sites, Products, Time Periods, Production Lines or Recipes.

Grouping or Zone Name

*Type: Text*

The actual zone or grouping that is empty. In other words, this entry is deliberately missing from the table referenced by Table Name, and references to this entry in other tables are not the result of misspellings.

Sample Input

Table Name	Grouping or Zone Name
Sites	Warehouse
Products	Bread

## Sites and Lines

### ZONE OPEN RESTRICTIONS

Use this table to restrict the opening and closing of sites based on time periods.

#### Zone Name

*Type: Text*

Must match a valid zone.

#### Time Period Group

*Type: Text*

Must match a valid time period grouping.

#### Min Number Open Sites

*Type: Numeric*

Minimum number of sites to open.

Non-negative, non-infinite, must be integral.

#### Max Number Open Sites

*Type: Numeric*

Maximum number of sites to open.

Non-negative, non-infinite, must be integral.

#### Sample Input

Zone Name	Time Period Grouping	Min Number Open Sites	Max Number Open Sites
Warehouse	P1	2	5
Warehouse	P2	2	5
Warehouse	P3	3	5

## SITE PRODUCT COSTS

This table is indexed by site, product-group, and thus there might be some site-product pairs that are referenced by multiple rows. The rule here is that the costs here are additive for such site-product pairs.

Take a look at the [Site Product Costs Table](#) documentation to understand more about this table.

### Site

*Type: Text*

Must match a valid site.

### Product Group

*Type: Text*

Must match a valid product grouping.

### SKU Cost

*Type: Numeric*

These costs are "handling" costs and not inventory costs. Goods pay these costs when they are physically inbound or outbound, but not when they are stored in starting or ending inventory. The concept of "inbound" is generalized to include goods that arrive via inbound shipments, or goods that are simply produced on site. Similarly, the concept of "outbound" is generalized to include goods that leave via outbound shipments, or goods that are consumed as demand satisfied on site. Finally, note that a full handling cost is applied only when the good experiences both "inbound" and "outbound". That is to say, if there are 50 units of inbound shipments and 50 units of outbound shipments, then the solution Site Product Details report will list a Flow of 50, and thus all 50 units will pay the appropriate handling penalty. But if there are 50 units of inbound shipments, and zero units of outbound shipments (and thus the ending inventory is 50 units higher than the starting inventory for that time period) then the flow is just 25 and those 25 units will pay the appropriate handling penalty.

### Weight Cost

*Type: Numeric*

These costs are similar to **SKU Cost** but are scaled by Weight field.

### Size Cost

*Type: Numeric*

This field creates an additional cost for the Average Inventory of the site. That is to say, for a given site product pair, you multiply the sum of all the relevant Size Cost entries by the product of the solution Average Inventory (for this site-product pair) and the Warehouse Volume (for the product). Recall that average inventory is itself computed as Flow/ITOR, plus the average of starting and ending inventory.

\[ \operatorname{TotalInventoryCost} = \sum \{ \operatorname{SizeCost} \} \* (\frac{\operatorname{Flow}}{\operatorname{ITOR}} + \frac{\operatorname{StartingInventory} + \operatorname{EndingInventory}}{2}) \* \operatorname{WarehouseVolume} \]

### Dwell Cost

*Type: Numeric*

Specifies a per-warehouse-volume cost for holding inventory. This cost doesn't apply to any form of inbound or outbound shipments. In other words, its like Size Cost, except without the ITOR scaled contribution of flow.

### Sample Input

Site	Product Group	SKU Cost	Weight Cost	Size Cost	Dwell Cost
W_Memphis	Bread	1.43			3.57
P_Dover	Cookie		5.61		
W_El Paso	Chips			2.91	

**HANDLING COSTS**

Use this table to specify source/destination specific handling costs.

**Source Zone Name**

*Type: Text*

Must match a valid zone.

**Destination Zone Name**

*Type: Text*

Must match a valid zone.

**Product Group**

*Type: Text*

Must match a valid product grouping.

**Handling Cost**

*Type: Numeric*

Cost to be applied based on total flow and Unit Type.

Non-negative, non-infinite.

**Unit Type**

*Type: Numeric*

If something other than SKU, then the cost is scaled by the appropriate value from the products table.

Possible values are:

- SKU
- Weight
- Shipment Volume
- Adjusted Volume
- Warehouse Volume

**Direction Type**

*Type: Text*

Determines which site should report on the incurred costs.

Possible values are:

- Inbound
- Outbound

**Sample Input**

Source Zone Name	Destination Zone Name	Product Group	Handling Cost	Unit Type	Direction Type
P_Dover	Warehouse	Bread	14	SKU	Inbound
W_Memphis	C_Alban	Cookie	5	Weight	Outbound
Warehouse	Customer	Chips	10	Shipment Volume	Inbound

**SITE PRODUCT INVENTORY INFO**

Sets various inventory based information based on site/product. Site, product pairs not in this table will use the default value for each field.

**Site**

*Type: Text*

Must match a valid site.

**Product**

*Type: Text*

Must match a valid product.

**Itor**

*Type: Numeric*

Inventory Turnover Ratio. Positive, infinity allowed. Default of one.

\[ \operatorname{ITOR} = \text{CostOfGoodSold}\hspace{1mm} / \hspace{1mm} \text{AverageInventory} \]

**Holding Cost**

*Type: Numeric*

Per-SKU, per-time-period cost of holding inventory. Non-negative, non-infinite. Default of 0.

**Starting Inventory**

*Type: Numeric*

The starting inventory in the first time period. Non-negative, non-infinite. Default of 0.

**Ending Inventory**

*Type: Numeric*

The ending inventory in the first time period. Non-negative, non-infinite. Default of 0.

**Sample Input**

Site	Product	Itor	Holding Cost	Starting Inventory	Ending Inventory
P_Dover	Bread	10	0.1	10000	100000
P_Dover	Cookie	5	0.05	16000	7000
P_Dover	Chips	20	0.1	50000	60000

**SITE PRODUCT PROHIBITIONS**

Use this table to prevent a specific site from handling specific products.

**Site**

*Type: Text*

Must match a valid site.

**Product Group**

*Type: Text*

Must match a valid product grouping.

Site	Product
P_Dover	Bread
P_Dover	Cookie
P_Dover	Chips

#### PRODUCTION LINE OPEN RESTRICTIONS

Use this table to restrict the opening and closing of production lines based on time period.

##### Production Line Group

*Type: Text*

Must match a valid production line grouping.

##### Time Period Group

*Type: Text*

Must match a valid time period grouping.

##### Min Number Open Production Lines

*Type: Numeric*

Minimum number of production lines to open.

Non-negative, non-infinite, must be integral.

##### Max Number Open Production Lines

*Type: Numeric*

Maximum number of production lines to open.

Non-negative, non-infinite, must be integral.

##### Sample Input

Production Line Group	Time Period Group	Min Number Open Production Lines	Max Number Open Production Lines
Dover_PLines	P1	2	5
Dover_PLines	P2	2	5
Dover_PLines	P3	3	5

**MAX VOLUME**

Use this table to control the maximum average inventory volume and fixed cost based on site, product group and time period.

**Site**

*Type: Text*

Must match a valid site.

**Product Group**

*Type: Text*

Must match a valid product group.

**Time Period**

*Type: Text*

Must match a valid time period.

**Max Volume**

*Type: Numeric*

Limit on average inventory, measured in units of warehouse volume. Non-negative, infinity allowed.

**Max End Inv Volume**

*Type: Numeric*

Limit on ending inventory, measured in units of warehouse volume. Non-negative, infinity allowed.

**Fixed Cost**

*Type: Numeric*

Fixed cost paid if there is any throughput of goods for the (Site, Product Group, Time Period) triplet. Non-negative non-infinite. This cost will be

**Sample Input**

<b>Site</b>	<b>Product Group</b>	<b>Time Period</b>	<b>Max Volume</b>	<b>Max End Inv Volume</b>	<b>Fixed Cost</b>
P_Dover	Bread	P1	7000		10000
P_Dover	Cookie	P1		6500	16000
P_Dover	Chips	P1	10000	9000	50000

## Carbon

### CARBON BY PRODUCTION LINE

Assess a production based carbon penalty for production-line-based production.

#### Site

*Type: Text*

A [Site](#) from a site-line-product triplet from [Production By Production Lines](#) or [Aggregate Production By Production Lines](#) table.

#### Production Line

*Type: Text*

A [Production Line](#) from a site-line-product triplet from [Production By Production Lines](#) or [Aggregate Production By Production Lines](#) table.

#### Product

*Type: Text*

A [Product](#) from a site-line-product triplet from [Production By Production Lines](#) or [Aggregate Production By Production Lines](#) table.

#### Carbon Per Unit

*Type: Numeric*

Amount of carbon per unit of production.

#### Sample Input

Site	Production Line	Product	Carbon Per Unit
P_Dover	PL_1_A	Bread	1.2
P_Dover	PL_2_B	Cookie	2.4

**CARBON BY PRODUCTION**

Assess a production based carbon penalty for site-based production.

**Site**

*Type: Text*

A [Site](#) from a site-product pair from the [Production](#) table.

**Product**

*Type: Text*

A [Product](#) from a site-product pair from the [Production](#) table.

**Carbon Per Unit**

*Type: Numeric*

Amount of carbon per unit of production.

**Sample Input**

Site	Product	Carbon Per Unit
P_Dover	Bread	1.5
P_Dover	Cookie	2

**CARBON BY INVENTORY**

Assess an average inventory volume based carbon penalty.

**Site**

*Type: Text*

A site from the [Sites](#) table.

**Carbon Per Inventory Volume**

*Type: Numeric*

Amount of carbon used per-average-inventory-volume. Non-negative, non-infinite.

**Sample Input**

Site	Carbon Per Inventory Volume
W_Boston	0.8
W_Cleveland	0.35

**CARBON BY FREIGHT**

Assess a shipping based carbon penalty.

Lane

*Type: Text*

A lane from the [Lanes](#) table.

Carbon Per Weight Distance

*Type: Numeric*

Amount of carbon used per-weight-distance. Non-negative, non-infinite.

Sample Input

<b>Lane</b>	<b>Carbon Per Weight Distance</b>
Plant-Warehouse	0.3

## Manage Bulk Download

### DOWNLOAD EXCLUDE

Use this table to exclude table from bulk download operations.

#### Schema Type

*Type: Text*

'Input', 'Solution' or 'Validation'

#### Table Name

*Type: Text*

Must be a table name for this schema type

#### Sample Input

Schema Type	Table Name
Input	Demand
Validation	Demand
Solution	Demand Shortfall

## 4.2.3 Output Tables

### Summary Reports

#### PARAMETERS

Reports on the Key Performance Indicators for the solution, as well as various diagnostics.

#### Key

*Type: Text*

Name of a solution KPI.

#### Value

*Type: Text*

Value of a solution KPI

#### Parameters explanation

**Lower Bound:** upper bound is the objective of the best known feasible solution

**Upper Bound:** lower bound gives a bound on the best possible objective

**Total Miscellaneous Cost:** calculated based on the [Fixed Cost](#) field from the [Max Volume](#) table. Will be applied at the Site, Product Group and Time Period level. Since it is a *Fixed* cost, it does *NOT* depend of how many units flows by the site.

**Total Site Product Cost:** calculated based on the [Site Product Costs](#) table. Will be applied at the Site and Product Group level. This is a *Variable* cost, so it will be multiplied for the value of units that are flowing by the site.

**Total Transportation Cost:** this cost is calculated based on Carriers tables, as [Zone Carrier](#), [Truck Carrier](#), etc.

**Total Inventory Holding Cost:** result of the *Average Inventory* multiplied for the [Holding Cost](#) value at the [Site Product Inv Info](#) table. It is a *Variable* cost, applied on the Site, Product, Time Period and SKU (unit) level.

**Total Production Cost:** *Variable* cost applied on produced units. Unit production costs can be set in the input tables [Production](#), [Production By Production Lines](#), [Aggregate Production](#) (when using this table, would be necessary to populate the [Tiered Costs](#) table as well).

**Total Open, Close, Operating Cost:** this costs can be set up at the [Sites](#) table, and applied at the Site level, by populating the fields [Opening Cost](#), [Closing Cost](#) and [Operating Cost](#).

Opening and closing costs would be considered depending on the field [Open Close Status](#). If a site is set up as a Potential site and the solution decides to open it, Opening Costs will be accounted. But if a site is configured as Pre-Existing, Opening Costs will *NOT* be calculated, because the site is already open. Closing costs would be applied on sites that were closed by the solution, and this only can happen when the Open Close Status is equal to Fixed or Pre-Existing. Operating Costs are *Fixed* costs accounted for each time period that the site is open.

**Total Cost:** Is the sum of all costs described above (Miscellaneous Costs + Site Product Costs + Transportation Costs + Inventory Holding Cost + Production Costs + Open/Close/Operating Site Costs).

**Total Revenue:** the total income from the network. It is calculated as the unit revenue, populated at the [Revenue](#) field ([Demand](#) table), multiplied by the satisfied demand.

**Total Demand Satisfied:** sometimes the solution cannot satisfy the demand values populated at the [Maximum Demand](#) field ([Demand](#) table). This parameters returns the amount of the demand that was satisfied by the solution.

**Total Profit:** it is the difference between Total Revenue and Total Cost.

**Percentage High Service Demand:** the percentage of the demand that was satisfied by sources within the *High Demand Service Distance* (High Demand Service Threshold parameter). Take a look at the [High Service Demand](#) page to understand more about this indicator.

**Timestamp:** date and time when the solution was executed.

**OPEN SITES**

Details the open sites based on time period.



This table does not show *CLOSED* sites, which would be Potential sites that were not opened or Pre-Existing sites that were closed. If the solution decides to close sites, it will return data at the [Sites Closing](#) table.

**Site**

*Type: Text*

Identifies an open site.

**Time Period**

*Type: Text*

The site is open in this time period.

**Opening Cost**

*Type: Numeric*

The opening cost that was applied. Will be zero if the site was open in the previous time period. Refers to the [Opening Cost](#) at the [Sites](#) table.



A site configured as Fixed or Pre-Existing at the field [Open Close Status](#) ([Sites](#) table) will not have any Opening Cost associated with it, because in that case this site would be already opened.

**Operating Cost**

*Type: Numeric*

The operating cost that was applied.

**Open Close Status**

*Type: Text*

Copied over from the [Sites](#) input table.

**Zone**

*Type: Text*

Copied over from the [Sites](#) table. The Open Sites Report Zone parameter selects the Zone field to copy ([Parameters](#) table).

**SITE CLOSINGS**

Details the closed sites based on time period.

**Site**

*Type: Text*

Identifies a newly closed site.

**Time Period**

*Type: Text*

The closing event occurs in this time period.

**Closing Cost**

*Type: Numeric*

The closing cost that was applied. Refers to the [Closing Cost](#) field at the [Sites](#) input table.

## OPEN PRODUCTION LINES

Details the open production lines based on time period. Use the input table [Production Lines](#) to set up the costs explored here.

### Site

*Type: Text*

Identifies the site of an open production line.

### Name

*Type: Text*

The name of the production line within the site that is open.

### Time Period

*Type: Text*

The production line is open in this time period.

### Opening Cost

*Type: Numeric*

The opening cost that was applied. Will be zero if the production line was open in the previous time period.

### Operating Cost

*Type: Numeric*

The operating cost that was applied.

**PRODUCTION LINE CLOSINGS**

Details the closed production lines based on time period.

**Site**

*Type: Text*

Identifies the site of a newly closed production line.

**Name**

*Type: Text*

The name of the production line within the site that is newly closed.

**Time Period**

*Type: Text*

The closing event occurs in this time period.

**Closing Cost**

*Type: Numeric*

The closing cost that was applied.

**FIXED COSTS**

Reports on the fixed costs incurred based on the Max Volume input table, as well as the volume used relative to the max volume constraints.

**Site**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which is used by the solution.

**Product Group**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which is used by the solution.

**Time Period**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which is used by the solution.

**Fixed Cost**

*Type: Numeric*

The fixed cost applied.

**Volume Used**

*Type: Numeric*

The appropriate sub-total of Average Inventory from the Site Product Details report multiplied by product Warehouse Volume.

**Percent Volume Used**

*Type: Numeric*

100 multiplied by Volume Used divided by Max Volume from the Max Volume table. This would be the *Inventory Occupation* indicator.

**End Inv Volume Used**

*Type: Numeric*

The appropriate sub-total of Ending Inventory from the [Site Product Details](#) report multiplied by product Warehouse Volume.

**Percent End Inv Volume Used**

*Type: Numeric*

100 multiplied by End Inv Volume Used divided by Max End Inv Volume from the [Max Volume](#) table.

## Detailed Reports

### SITE PRODUCT DETAILS

Provides site usage details indexed by site, product and time period.

#### Site

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which is used by the solution.

#### Product

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which is used by the solution.

#### Time Period

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which is used by the solution.

#### Flow

*Type: Numeric*

The average of inbound shipments and outbound shipments. Local production and demand satisfied are included.

#### Average Inventory

*Type: Numeric*

Flow divided by ITOR plus the average of starting and ending inventory.

$$\text{Average Inventory} = \frac{\text{Flow} + (\text{Starting Inventory} + \text{Ending Inventory})}{2}$$

#### Starting Inventory

*Type: Numeric*

The inventory at the start of this time period.

#### Ending Inventory

*Type: Numeric*

The inventory at the end of this time period.

#### Total SKU Cost

*Type: Numeric*

Flow multiplied by the sum of the appropriate SKU Cost entries from Site Product Costs.

$$\text{Total SKU Cost} = \text{Flow} \times \sum \text{SKU Cost}$$

#### Total Weight Cost

*Type: Numeric*

Flow multiplied by the sum of the appropriate Weight Cost entries from Site Product Costs.

$$\text{Total Weight Cost} = \text{Flow} \times \sum \text{Weight Cost}$$

#### Total Size Cost

*Type: Numeric*

Flow multiplied by product Warehouse Volume multiplied by the sum of the appropriate Size Cost entries from Site Product Costs.

$$\text{Total Size Cost} = \text{Flow} \times \sum \text{Size Cost}$$

#### Total Dwell Cost

*Type: Numeric*

Ending Inventory multiplied by the sum of the appropriate Dwell Cost entries from [Site Product Costs](#) table.

$\text{Total Dwell Cost} = \text{EndingInventory} * \sum{\text{DwellCost}}$

**Inventory Holding Cost***Type: Numeric*

Average Inventory multiplied by product Holding Cost.

$\text{Average Holding Cost} = \text{Average Inventory} * \text{Holding Cost}$

**Inbound Handling Cost***Type: Numeric*

Flow multiplied by the appropriate per-unit costs inferred from the Handling Costs table.

Only the Handling Costs records that match Destination Zone for Inbound records will be used for this computation.

$\text{Inbound Handling Cost} = \text{Flow} * \text{Handling Cost}$

**Outbound Handling Cost***Type: Numeric*

Flow multiplied by the appropriate per-unit costs inferred from the Handling Costs table.

Only the Handling Costs records that match Source Zone for Outbound records will be used for this computation.

$\text{Outbound Handling Cost} = \text{Flow} * \text{Handling Cost}$

**DEMAND SATISFIED**

Details both the demand satisfied and the demand shortfalled.

**Site**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has positive Demand Satisfied or Demand Shortfall.

**Product**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has positive Demand Satisfied or Demand Shortfall.

**Time Period**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has positive Demand Satisfied or Demand Shortfall.

**Demand Satisfied**

*Type: Numeric*

The amount of demand satisfied.

**Revenue**

*Type: Numeric*

Demand Satisfied multiplied by [Revenue](#) from the [Demand](#) input table.

**Demand Shortfall**

*Type: Numeric*

Demand Satisfied subtracted from Max Demand from the Demand table.

## LANDED COST

Reports on the total *Cost-To-Serve* for satisfied demand. See more about the origin of the costs shown in this table in the reference of the output table [Parameters](#).

**Note**

You can display Landed Cost data as **Tree Charts** by setting the field [Landed Cost X-Ray](#) as **Enabled** at the [Demand](#) input table. To access the chart for a specific Site-Product-Time Period triplet, select the record and then select the icon at the top left corner, on the table grid page, as shown below.

Hide empty tables
 

C idle

Site	Product	Time Period	Type	Units	Production Cost	Inventory Holding Cost	Open Close
C_Rockford	Pellet	2022-9	Demand Satisfied	157.00	\$11,154.40	\$6.28	
C_Charleston	Pellet	2022-9	Demand Satisfied	159.00	\$11,296.50	\$6.36	
C_Rockford	Pellet	2022-8	Demand Satisfied	159.00	\$11,296.50	\$6.36	
C_Charleston	Pellet	2022-8	Demand Satisfied	161.00	\$11,438.59	\$6.44	
C_Killeen	Pellet	2022-9	Demand Satisfied	162.00	\$11,509.64	\$11.72	
C_Rockford	Pellet	2022-7	Demand Satisfied	162.00	\$11,509.64	\$6.48	
C_Rockford	Pellet	2022-10	Demand Satisfied	162.00	\$11,509.64	\$6.48	
C_Jackson	Pellet	2022-9	Demand Satisfied	163.00	\$11,580.68	\$6.52	
C_Killeen	Pellet	2022-8	Demand Satisfied	164.00	\$11,651.73	\$6.56	
C_Charleston	Pellet	2022-7	Demand Satisfied	164.00	\$11,651.73	\$6.56	
C_Charleston	Pellet	2022-10	Demand Satisfied	164.00	\$11,651.73	\$6.56	
C_Jackson	Pellet	2022-8	Demand Satisfied	165.00	\$11,722.78	\$6.60	
C_Killeen	Pellet	2022-7	Demand Satisfied	166.00	\$11,793.83	\$6.64	
C_Killeen	Pellet	2022-10	Demand Satisfied	166.00	\$11,793.83	\$6.64	

Page size: auto 1 to 14 of 6,188 < < Page 1 of 442 > >>

Total 1 visualization tree(s) displayed, out of 1 records.

↔ ☰ ✖ ✓ C\_Killeen - Pellet - 2022-9 - Demand Satisfied

```

graph LR
    A[C_Killeen  
Pellet  
2022-9] --> B[P_Lucedale  
Pellet  
2022-8]
    B --> C[P_Lucedale  
Fibre Grade 1  
2022-9]
    B --> D[P_Lucedale  
Fibre Grade 2  
2022-9]
  
```

### Site

Type: Text

Identifies a (Site, Product, Time Period) triplet which is used by the solution.

It either matches a record from the Demand Satisfied report or has ending inventory in the final time period.

### Product Group

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which is used by the solution.

It either matches a record from the Demand Satisfied report or has ending inventory in the final time period.

## Time Period

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which is used by the solution.

It either matches a record from the Demand Satisfied report or has ending inventory in the final time period.

## Type

*Type: Text*

Either 'Demand Satisfied' or 'Ending Inventory'

The latter refers to ending inventory in the final time period.

If cycles are detected, then this field will be used to populate cycle information and all the data fields are populated with null.

## Units

*Type: Numeric*

The total units served.

## Production Cost

*Type: Numeric*

The Production Cost component of total cost-to-service.

## Inventory Holding Cost

*Type: Numeric*

The Inventory Holding Cost component of total cost-to-service.

## Open Close Operating Cost

*Type: Numeric*

The Open Close Operating Cost component of total cost-to-service.

## Miscellaneous Fixed Cost

*Type: Numeric*

The Miscellaneous Fixed Cost component of total cost-to-service. This cost is related to the field **Fixed Cost**, at the **Max Volume** table.

## Site Product Cost

*Type: Numeric*

The Site Product Cost component of total cost-to-service.

## Transportation Cost

*Type: Numeric*

The Transportation Cost component of total cost-to-service. It is the sum of Shipping costs, In Transit Holding costs and Duty Tariff costs.

## Total Cost

*Type: Numeric*

The total cost-to-service.

## Viz ID

*Type: Numeric*

An internal ID field useful for the visualization feature.

**PRODUCTION**

Details both the site-based production and the violations of the site-based min supply constraint.

**Site**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

**Product**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

**Time Period**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

**Production**

*Type: Numeric*

The amount of site based production.

**Production Cost**

*Type: Numeric*

Production multiplied by Production Cost from the [Production](#) table. Will also include the Tiered Costs (if using the [Aggregate Production](#) table).

**Hours Used**

*Type: Numeric*

Relevant total of site based production divided by [Rate](#) from the [Production](#) input table.

**Min Supply Violation**

*Type: Numeric*

The violation of the Min Supply constraint used by the Total Infeasibility objective.

**PRODUCTION BY PRODUCTION LINES**

Details the production-line-based production and violations of the production-line-based min supply constraint.

**Site**

*Type: Text*

Identifies a (Production Line, Product, Time Period) triplet which has production line based Production.

**Production Line**

*Type: Text*

Identifies a (Production Line, Product, Time Period) triplet which has production line based Production.

**Product**

*Type: Text*

Identifies a (Production Line, Product, Time Period) triplet which has production line based Production.

**Time Period**

*Type: Text*

Identifies a (Production Line, Product, Time Period) triplet which has production line based Production.

**Production**

*Type: Numeric*

The amount of production line based production.

**Production Cost**

*Type: Numeric*

Production multiplied by Production Cost from the Production By Production Lines table.

**Hours Used**

*Type: Numeric*

Relevant total of production line based production divided by Rate from the Production By Production Lines input table.

**Min Supply Violation**

*Type: Numeric*

The violation of the Min Supply constraint used by the Total Infeasibility objective.

**AGGREGATED PRODUCTION**

Details the site-based aggregate production and violations of the site-based min hours constraint.

**Site**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

**Product Group**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

**Time Period**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

**Hours Used**

*Type: Numeric*

Relevant total of site based production divided by Rate from the [Production](#) table.

**Min Hours**

*Type: Numeric*

Relevant value from the Aggregate Production table.

**Max Hours**

*Type: Numeric*

Relevant value from the Aggregate Production table.

**Min Utilization**

*Type: Numeric*

100 multiplied by Min Hours divided by Hours Used.

**Max Utilization**

*Type: Numeric*

100 multiplied by Max Hours divided by Hours Used.

**Min Hours Violation**

*Type: Numeric*

The violation of the Min Hours constraint used by the Total Infeasibility objective.

**AGGREGATED PRODUCTION BY PRODUCTION LINES**

Details the production-line-based aggregate production and the production-line-based min hours constraint.

**Site**

*Type: Text*

Identifies a (Production Line, Product Group, Time Period) triplet which has production line based Production.

It matches a record from the Aggregate Production By Production Line table.

**Production Line**

*Type: Text*

Identifies a (Production Line, Product Group, Time Period) triplet which has production line based Production.

It matches a record from the Aggregate Production By Production Line table.

**Product Group**

*Type: Text*

Identifies a (Production Line, Product Group, Time Period) triplet which has production line based Production.

It matches a record from the Aggregate Production By Production Line table.

**Time Period**

*Type: Text*

Identifies a (Production Line, Product Group, Time Period) triplet which has production line based Production.

It matches a record from the Aggregate Production By Production Line table.

**Hours Used**

*Type: Numeric*

Relevant total of site based production divided by Rate from the Production By Production Lines table.

**Min Hours**

*Type: Numeric*

Relevant value from the Aggregate Production By Production Lines table.

**Max Hours**

*Type: Numeric*

Relevant value from the Aggregate Production By Production Lines table.

**Min Utilization**

*Type: Numeric*

100 multiplied by Min Hours divided by Hours Used.

**Max Utilization**

*Type: Numeric*

100 multiplied by Hours Used divided by Max Hours.

**Min Hours Violation**

*Type: Numeric*

The violation of the Min Hours constraint used by the Total Infeasibility objective.

## BOM DETAILS

Use this table to visualize the **Bill of Materials** table as a **Tree Chart**.

To access the BOM Tree Chart, select the record and then click on the icon at the top left corner, at the table grid, as shown below.

The screenshot shows the 'BOM Details' section of the application. On the left, there's a sidebar with various report categories like 'Parameters', 'Open Sites', 'Open Production Lines', etc. The 'BOM Details' section is selected. In the main area, there's a table with one row:

Product	Recipe	Number Immediate Components	Number Immediate and Derived Components
Default	Chocolate Bar	4	12

A blue square highlights the tree chart icon in the top-left corner of the table header. The status bar at the bottom right shows 'idle'.



### Product

Type: Text

An assembled Product.

### Recipe

Type: Text

An entry from [Recipes](#), or Default.

### Number Immediate Components

*Type: Numeric*

The immediate number of children in the visualization tree.

For example, in the Tree Chart image above, the assembly Chocolate Bar has 4 immediate childrens, that would be the components required to produce the assembly, given by Caramel, Chocolate, Nougat and Peanuts.

### Number Immediate and Derived Components

*Type: Numeric*

Total number of descendants in the visualization tree. That would be the number of all the different components required to produce the assembly, since sometimes a component can have its own components. In the Tree Chart example above, the component Chocolate is an assembly of Butter, Cocoa, Milk and Sugar.

**RECIPE PRODUCTION**

When a recipe is used for site based production, this table is populated with the appropriate details.

**Site**

*Type: Text*

Identifies a (Site, Product, Time Period, Recipe) 4-plet which has site-based production.

**Product Group**

*Type: Text*

Identifies a (Site, Product, Time Period, Recipe) 4-plet which has site-based production.

**Time Period**

*Type: Text*

Identifies a (Site, Product, Time Period, Recipe) 4-plet which has site-based production.

**Recipe**

*Type: Text*

Identifies a (Site, Product, Time Period, Recipe) 4-plet which has site-based production.

**Production**

*Type: Numeric*

The amount of production using this recipe at this site for this product in this time period.

**RECIPE PRODUCTION BY PRODUCTION LINES**

When a recipe is used for production line based production, this table is populated with the appropriate details.

**Site**

*Type: Text*

Identifies a (Site, Production Line, Product, Time Period, Recipe) 5-plet which has production-line-based production.

**Production Line**

*Type: Text*

Identifies a (Site, Production Line, Product, Time Period, Recipe) 5-plet which has production-line-based production.

**Product**

*Type: Text*

Identifies a (Site, Production Line, Product, Time Period, Recipe) 5-plet which has production-line-based production.

**Time Period**

*Type: Text*

Identifies a (Site, Production Line, Product, Time Period, Recipe) 5-plet which has production-line-based production.

**Recipe**

*Type: Text*

Identifies a (Site, Production Line, Product, Time Period, Recipe) 5-plet which has production-line-based production.

**Production**

*Type: Numeric*

The amount of production using this recipe at this production line for this product in this time period.

**RECIPE DETAILED DISPOSAL**

This table is populated when **Byproducts** are disposed of with site based production.

**Site**

*Type: Text*

Identifies a (Site, Assembly, Byproduct, Time Period, Recipe) 5-plet which has site-based by-product disposal.

**Assembly**

*Type: Text*

Identifies a (Site, Assembly, Byproduct, Time Period, Recipe) 5-plet which has site-based by-product disposal.

**Byproduct**

*Type: Text*

Identifies a (Site, Assembly, Byproduct, Time Period, Recipe) 5-plet which has site-based by-product disposal.

**Time Period**

*Type: Text*

Identifies a (Site, Assembly, Byproduct, Time Period, Recipe) 5-plet which has site-based by-product disposal.

**Recipe**

*Type: Text*

Identifies a (Site, Assembly, Byproduct, Time Period, Recipe) 5-plet which has site-based by-product disposal.

**Disposed**

*Type: Numeric*

The amount of the byproduct disposed of, immediately after production, using this recipe for this assembly in this time period.

**Disposal Cost**

*Type: Numeric*

Disposed multiplied by Byproduct Disposal Cost from the [Recipe Byproduct](#) input table.

**RECIPE DETAILED DISPOSAL BY PRODUCTION LINES**

This table is populated when **Byproducts** are disposed of with production line based production.

**Site**

*Type: Text*

Identifies a (Site, Production Line, Assembly, Byproduct, Time Period, Recipe) 6-plet which has site-based by-product disposal.

**Production Line**

*Type: Text*

Identifies a (Site, Production Line, Assembly, Byproduct, Time Period, Recipe) 6-plet which has site-based by-product disposal.

**Assembly**

*Type: Text*

Identifies a (Site, Production Line, Assembly, Byproduct, Time Period, Recipe) 6-plet which has site-based by-product disposal.

**Byproduct**

*Type: Text*

Identifies a (Site, Production Line, Assembly, Byproduct, Time Period, Recipe) 6-plet which has site-based by-product disposal.

**Time Period**

*Type: Text*

Identifies a (Site, Production Line, Assembly, Byproduct, Time Period, Recipe) 6-plet which has site-based by-product disposal.

**Recipe**

*Type: Text*

Identifies a (Site, Production Line, Assembly, Byproduct, Time Period, Recipe) 6-plet which has site-based by-product disposal.

**Disposed**

*Type: Numeric*

The amount of the byproduct disposed of, immediately after production, using this recipe at this production line for this assembly in this time period.

**Disposal Cost**

*Type: Numeric*

Disposed multiplied by Byproduct Disposal Cost from the [Recipe Byproduct](#) input table.

## Transportation Reports

### ALL POSSIBLE ARCS

This report is populated based on the parameter 'Generate All Possible Arcs Report' ([Parameters](#) input table). If populated as 'After Optimization' or 'Instead of Optimization', it will detail all the arcs generated by the lanes and carriers information, as well as the arcs from the Arc Overrides table.

Total per-sku cost is the sum of Shipping Cost and In Transit Holding Cost

#### Source

*Type: Text*

The source site for this arc (aka shipment option).

#### Destination

*Type: Text*

The destination site for this arc (aka shipment option).

#### Product

*Type: Text*

The product for this arc (aka shipment option).

#### Transit Time

*Type: Text*

The time offset this arc (aka shipment option).

Will always be zero if the 'In Transit Inventory' parameter is 'Used For Holding Costs Only'

#### Shipping Cost

*Type: Numeric*

Per-sku shipping cost.

Total per-sku cost is the sum of Shipping Cost and In Transit Holding Cost.

#### In Transit Holding Cost

*Type: Numeric*

Per-sku shipping cost.

Total per-sku cost is the sum of Shipping Cost and In Transit Holding Cost.

#### Duty Tariff Cost

*Type: Numeric*

Per-sku simple duty tariff cost. Is is related to the duty tariff costs populated at the [Simple Duty Tariff](#) table.



This Duty Tariff Cost DOES NOT include the advanced duty tariff cost that is based on plant production tracking. These advanced duty tariff costs are shown at the [Arcs Advanced Duty Tariff Cost](#) field, at the [Arcs Advanced Duty Tariff](#) output table.

#### Distance

*Type: Numeric*

Distance between Source and Destination.

#### Lane

*Type: Text*

If the arc is generated via the Lanes table, then the relevant Lane record is identified here.

**ARCS**

Details all the site-to-site shipments made by the solution.

**Source**

*Type: Text*

The source site for this shipment.

**Destination**

*Type: Text*

The destination site for this shipment.

**Product**

*Type: Text*

The product for this shipment.

**Start Time Period**

*Type: Text*

The time period for which this shipment started.

Start Time Period will always equal End Time Period if the 'In Transit Inventory' parameter is 'Used For Holding Costs Only' ([Parameters](#) table).

**End Time Period**

*Type: Text*

The time period during for which this time period ended.

Start Time Period will always equal End Time Period if the 'In Transit Inventory' parameter is 'Used For Holding Costs Only' ([Parameters](#) table).

**Flow**

*Type: Numeric*

The total SKU quantity (units) shipped by the shipment.

**Shipping Cost**

*Type: Numeric*

Per-sku shipping cost.

The shipping cost applied by this shipment.

**In Transit Holding Cost**

*Type: Numeric*

The in transit holding cost applied by this shipment.

**Duty Tariff Cost**

*Type: Numeric*

The duty tariff cost applied by this shipment. Will include BOTH the Simple and Advanced duty tariff cost applied.

**Lane**

*Type: Text*

If the arc is generated via the Lanes table, then the relevant Lane record is identified here.

**Distance**

*Type: Numeric*

Distance between Source and Destination.

**Map Code**

*Type: Text*

Used to label the map correctly.

## TRANSPORTATION DETAILS

This report is populated based on the 'Generate Transportation Details Report' parameter ([Parameters](#) table). If populated as 'Always', it details the usage of the lane-carrier based arcs.

### Source

*Type: Text*

The source site for this shipment.

### Destination

*Type: Text*

The destination site for this shipment.

### Product

*Type: Text*

Identifies the product.

### Carrier

*Type: Text*

Identifies the carrier.

### Lane

*Type: Text*

Identifies the appropriate record from the Lanes table.

### Flow

*Type: Numeric*

The total units shipped, across all time periods, using this (Source, Destination, Product, Carrier).

### Percentage Flow

*Type: Numeric*

100 for all carriers other than [Blended Carriers](#). For [Blended Carriers](#), will be the appropriate value 0 and 100 computed from Blend Weight.

### Shipping Cost

*Type: Numeric*

The shipping cost applied by this record.

### Avg Shipment

*Type: Numeric*

The average shipment or Null if no Average Shipment was applied.

See more about average shipments at the [Average Shipment Logic](#) page (FAQ section).

### Truck Utilization

*Type: Numeric*

Will be Null unless Carrier is a Truck Carrier. Otherwise will be computed from Avg Shipment if one was applied. Will default to 100 for a Truck Carrier which has Null Avg Shipment.

See more about average shipments at the [Average Shipment Logic](#) page (FAQ section).

### Distance

*Type: Numeric*

Distance between Source and Destination.

**ARCS ADVANCED DUTY TARIFF**

Provides the production zone breakdown of the site-site shipments. This table is populated only if the [Advanced Duty Tariff](#) input table is populated with valid data.

**Production Zone**

*Type: Text*

The zone of the site that originally produced the goods in the sub-shipment.

**Source**

*Type: Text*

The source site for the sub-shipment.

**Destination**

*Type: Text*

The destination site for the sub-shipment.

**Product**

*Type: Text*

The product site for the sub-shipment.

**Start Time Period**

*Type: Text*

The time period for which the sub-shipment started.

**End Time Period**

*Type: Text*

The time period for which the sub-shipment ended.



Start Time Period will always be equal to End Time Period if the 'In Transit Inventory' parameter is 'Used For Holding Costs Only'.

**Flow**

*Type: Numeric*

The total SKU shipped by this sub-shipment.

**Advanced Duty Tariff Cost (ADTC)**

*Type: Numeric*

The advanced duty tariff cost applied by the sub-shipment. It is the Flow column multiplied by the duty tariff cost per SKU.

$\text{ADTC} = \text{Flow} * \text{Duty Tariff Cost}$



This Advanced Duty Tariff Cost DOES NOT include the duty tariff cost populated at the [Simple Duty Tariff](#) table. The simple duty tariff cost will be shown at the [Duty Tariff Cost](#) field, at the [Arcs](#) table, summed up with this cost.

## Infeasibility Reports

### BOTTLENECKS

The bottlenecks report is especially useful when troubleshooting infeasible models. The standard response to "The solver can't generate a solution to this model" is to follow the "Total Infeasibility" guidance from the [Parameter Details](#) documentation.

#### Input Table

*Type: Text*

The input table that defines the constraint.

#### Input Field

*Type: Text*

The numerical field on the input table that defines constraint.

#### Primary Key Value

*Type: Text*

An entry from the primary key of the input table.

Will be comma delimited if Input Table has more than one primary key field.

#### Percent Utilization

*Type: Numeric*

The percentage utilization of the constraint.

**DEMAND SHORTFALL**

Has the records from the Demand Satisfied report where Demand Shortfall is positive.

**Site**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has positive Demand Shortfall.

**Product Group**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has positive Demand Shortfall.

**Time Period**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has positive Demand Shortfall.

**Demand Shortfall**

*Type: Numeric*

Max Demand subtracted from Demand Satisfied from the Demand table.

**MIN FLOW VIOLATIONS**

When the Total Infeasibility option for the Objective parameter is used, this table is populated with the various constraint violations from the Min Max Flows table.

**Source Zone Name**

*Type: Text*

Must match a valid zone.

copied from the Min Max Flows input table.

**Destination Zone Name**

*Type: Text*

Must match a valid zone.

copied from the Min Max Flows input table.

**Product Group**

*Type: Text*

Must match a valid product grouping.

copied from the Min Max Flows input table.

**Time Period Group**

*Type: Text*

Must match a valid time period grouping.

copied from the Min Max Flows input table.

**Min Flow Violation**

*Type: Numeric*

The violation of the Min Flow constraint used by the Total Infeasibility objective.

**Binary Min Violation**

*Type: Numeric*

The violation of the Binary Min constraint used by the Total Infeasibility objective.

**Min Percentage Supplier Violation**

*Type: Numeric*

The violation of the Min Percentage Supplier constraint used by the Total Infeasibility objective.

**MIN SUPPLY VIOLATIONS**

Has the records from the reports [Production](#) and [Production By Production Lines](#) where Min Supply Violation is positive.

**Site**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

Or

Identifies a (Production Line, Product, Time Period) triplet which has production line based Production.

**Production Line**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

Or

Identifies a (Production Line, Product, Time Period) triplet which has production line based Production.

**Product**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

Or

Identifies a (Production Line, Product, Time Period) triplet which has production line based Production.

**Time Period**

*Type: Text*

Identifies a (Site, Product, Time Period) triplet which has site based Production.

Or

Identifies a (Production Line, Product, Time Period) triplet which has production line based Production.

**Min Supply Violation**

*Type: Numeric*

The violation of the Min Supply constraint used by the Total Infeasibility objective.

**MIN HOURS VIOLATIONS**

Has the records from the [Aggregate Production](#) and [Aggregate Production By Production Lines](#) reports where Min Hours Violation is positive.

**Site**

*Type: Text*

Identifies a (Site, Product Group, Time Period) triplet which has site based Production and matches a record from the Aggregate Production table.

Or

Identifies a (Production Line, Product Group, Time Period) triplet which has production line based Production and matches a record from the Aggregate Production By Production Line table.

**Production Line**

*Type: Text*

Identifies a (Site, Product Group, Time Period) triplet which has site based Production and matches a record from the Aggregate Production table.

Or

Identifies a (Production Line, Product Group, Time Period) triplet which has production line based Production and matches a record from the Aggregate Production By Production Line table.

**Product Group**

*Type: Text*

Identifies a (Site, Product Group, Time Period) triplet which has site based Production and matches a record from the Aggregate Production table.

Or

Identifies a (Production Line, Product Group, Time Period) triplet which has production line based Production and matches a record from the Aggregate Production By Production Line table.

**Time Period**

*Type: Text*

Identifies a (Site, Product Group, Time Period) triplet which has site based Production and matches a record from the Aggregate Production table.

Or

Identifies a (Production Line, Product Group, Time Period) triplet which has production line based Production and matches a record from the Aggregate Production By Production Line table.

**Min Hours Violation**

*Type: Numeric*

The violation of the Min Hours constraint used by the Total Infeasibility objective.

**STARTING ENDING INVENTORY VIOLATIONS**

Has the record where the starting inventory violations and/or ending inventory violations are positive.

**Site**

*Type: Text*

Identifies a (Site, Product) pair that has a starting inventory violation or an ending inventory violation.

**Product**

*Type: Text*

Identifies a (Site, Product) pair that has a starting inventory violation or an ending inventory violation.

**Starting Inventory Violation**

*Type: Numeric*

The amount by which the starting inventory in the first time period dropped below the input requirement.

**Ending Inventory Violation**

*Type: Numeric*

The amount by which the ending inventory in the last time period dropped below the input requirement.

**Carbon****CARBON BY FREIGHT**

Reports on the shipping based carbon penalty.

**Lane**

*Type: Text*

Specifies a valid entry from the [Lanes](#) table.

**Total Carbon Used**

*Type: Numeric*

Amount of carbon used based on per-weight-distance.

**CARBON BY PRODUCTION**

Reports on the production based carbon penalty for site-based production.

**Site**

*Type: Text*

Specifies a site-product pair from the [Production](#) or [Aggregate Production](#) table.

**Product**

*Type: Text*

Specifies a site-product pair from the [Production](#) or [Aggregate Production](#) table.

**Total Carbon Used**

*Type: Numeric*

Amount of carbon used based on production.

**CARBON BY INVENTORY**

Reports on the average inventory volume based carbon penalty.

**Site**

*Type: Text*

Specifies a valid [Site](#).

**Total Carbon Used**

*Type: Numeric*

Amount of carbon used based on average inventory volume.

**CARBON BY PRODUCTION LINE**

Reports on the production based carbon penalty for production-line-based production.

**Site**

*Type: Text*

Specifies a site-line-product triplet from the [Production By Production Lines](#) or [Aggregate Production By Production Lines](#) table.

**Production Line**

*Type: Text*

Specifies a site-line-product triplet from the [Production By Production Lines](#) or [Aggregate Production By Production Lines](#) table.

**Product**

*Type: Text*

Specifies a site-line-product triplet from the [Production By Production Lines](#) or [Aggregate Production By Production Lines](#) table.

**Total Carbon Used**

*Type: Numeric*

Amount of carbon used based on production.

## 4.2.4 FAQs

---

### Index

This collection of files is the created documentation for Foresta Supply Chain Network Optimizer. More documentation is coming, but in the meantime, lets [KBO](#) together.

- **Average Shipment Logic** is a deep dive into one of trickier bits of Foresta Supply Chain Network Optimizer functionality. If you're not a sophisticated user, you can leave fields at their default values and just ignore this topic.
- **Bottlenecks Report** is a guide about how to track bottlenecks at the supply chain network and troubleshoot infeasible models.
- **High Level Summary Report** is a good place to start.
- **Introduction to Lanes, Carriers and Arcs** explains how the optimizer uses lanes and carriers to generate shipment arcs. This functionality is optional, but we anticipate it being used by the typical modeler. It's heavily influenced by the LogicNet Plus functionality that was well reviewed by a broad user community.
- **Parameters** is a additional section with more details about modeling parameters.
- **Site Product Costs Table** was generated based on questions from a Foresta Supply Chain Network Optimization early adopter.
- **Zone Grouping Failures** is a deep dive into how the solver diagnosis various data entry mistakes, since zones, groupings and carriers can be a little confusing.
- **Optimization Objectives** will explore how you can optimize your supply chain network using different optimization drivers.
- **Modeling Constraints** is a detailed section about which constraints can be considered in a Foresta Supply Chain Network Optimization model.
- **Flow Constraints** is a deep dive into this tricky type of constraint.
- **Zones and Groupings** is another section explaining how to use Zones and Groupings.
- **Modeling Constraints High Service Demand** is a deep dive into how to consider Service Levels in your optimization model.

## Average Shipment Logic

There are four carriers that will actually use an average shipment.

- The **Truck Carrier** can use an average shipment to determine the average utilization of a truck. If the average shipment isn't found, than the truck is assumed to be fully utilized.
- The unit type of the average shipment the truck carrier looks up is based on the unit type of the truck itself.
- The **Distance Weight Band Carrier** needs the average shipment, because it uses it to look up the appropriate weight band.
- The unit type of the average shipment for the distance weight band carrier is always weight.
- The **SMC Carrier**. This would require weight as the average shipment. The distance weight band carrier is actually very similar to the SMC carrier - they are based around converting a point to point shipment to a shipping cost, based on the geographic details of the two points and the weight of the shipment. The shipment cost is thus converted into an a per-unit cost based on cost-per-weight of the shipment and the weight of the product.
- The FreightWaves carrier also requires an average shipment measured in weight. It uses average shipment in a manner similar to the truck carrier.

Some people get confused and think the **UPS Carrier** uses an average shipment. It does not. The UPS carrier knows the package weight from the table itself, and thus uses that in a somewhat similar way to the SMC carrier. The package is priced out for the shipment points, which is converted to a cost-per-weight, which is then turned into a per-unit cost based on the weight of the product.

There is a three tier system for resolving the average shipment itself.

- The SMC, Truck and FW (FreightWaves) carriers have a field for setting the average shipment directly. If that is populated (i.e. set to a number instead of null) then that value is used regardless of any other values.
- The **Blending** and **Cheapest** carriers similarly have a field for setting average shipments. If that field is populated, then its assumed to be populated with the correct unit for any Truck/SMC/Distance-Weight-Break carrier that encapsulates and which doesn't set the average shipment directly. Thus, the second tier for resolving average shipment is to look into the encapsulating blending/cheapest carrier (if the carrier that uses average shipment is so encapsulated) and use that value if provided.
- Finally, the **Arc Shipment Data** table will be referenced, and if there is a unit appropriate non-null average shipment there, it will be used.

For the FreightWaves and truck carrier, a warning will be issued if the Average Shipment is larger than the capacity of the truck. In this case, a truck utilization of 100% will be used to price out the shipment arc. However, the user is urged to examine these warnings carefully as they are likely providing inappropriate average shipment values. The normal state of affairs is for trucks to carry loads that are less than the capacity of the truck. In this case, no warnings are issued and the truck capacity doesn't figure into the math that determines the per-sku shipping cost of the arc.

## Best Modeling Practices

### GENERAL ADVICE

The combination of [Lanes](#) and [Carriers](#) provides significant modeling power to the user. Lanes are iterated in order of precedence (with precedence 1 fully processing before precedence 2, and so on). Any lanes that have been generated by a previous lane record will not be overwritten by a subsequent lane record. This allows the user to generate shipment arcs that follow default-override patterns quite naturally (with the override lane being processed before the default lane).

That said, lanes and carriers do interact in complicated ways, and it's easy to get confused. For this reason, we recommend setting the "Generate Errors and Warnings" [parameter](#) to the non-default value of "Instead Of Optimization" for users just beginning the process of learning to model with lanes and carriers. At this setting, the tool will populate additional records in the validation tables, but it won't actually run an optimization or generate a solution.

This setting is helpful for following the guidance discussed below. Once your model is obeying "[Clean modeling](#)" (which will come naturally with experience), then return "Generate Errors and Warnings" to the default value of "Always" and you can begin generating solutions.

### ZONE GROUPING FAILURE TYPE ONE

This is the most obvious type of zone/grouping failure. We will demonstrate this with zones, but the same logic applies to product groupings and time period groupings.

The simplest way to demonstrate this failure is to put the same zone name in two different zone columns. Other similar problems arise when you put a site name in a zone column, or when you use *all* as a zone name or a site name.

The solver diagnosis this failure during validation. This is an advanced validation failure, which means that it is only checked if more basic conditions pass. The validation table that diagnosis this problem is called "Grouping Duplicates", and it is in the "Advanced Validation Details" subsection of the Validation tables.

### ZONE GROUPING FAILURE TYPE TWO

This is a more subtle form of grouping failure that only occurs with the carrier tables. For any given carrier, you cannot use zones and/or groupings to generate rating ambiguity.

So, for example, suppose the Source Zone, Destination Zone, Product Grouping triplets for the "per-mile" Zone Carrier were as follows.

```
(*all*, *all*, *all*)
(New York, *all*, *all*)
(*all*, Customer, *all*)
(*all*, *all*, P)
```

Then there the following questions cannot be answered one way or the other.

- Should outbound shipments from New York rate with the first or the second record?
- Should shipments that use the P product rate with the first or the last record?
- Which of the 4 records should be used to rate a shipment of P outbound from New York and inbound into a site from the Customer zone?

Note that under default parameter settings, these records are logged in Logging > Solution Logs > Error and Warning Logs. If the "Generate Errors and Warnings" parameter is set to "Instead Of Optimization", then they will be treated as advanced validation errors and will populate the "Carrier Multi Field References" table in the "Advanced Validation Details" subsection of the Validation tables.

### EMPTY ZONES OR GROUPINGS

This is a data condition that might reflect a data entry error similar to a Foreign Key failure, or it might be a natural consequence of a zone or a grouping that has no elements. Consider, for example, a model that organizes its sites into Plants, Warehouses, and Customers. Suppose a variety of data entry tables refer to the "Warehouse" zone. Further suppose, as part of what-if analysis, some copy of the scenario deletes all the sites belonging to the Warehouse zone. It's hard for the solver to predict how the model should validate. Should the data tables that refer to Warehouse be quietly ignored? Should they be ignored from core model building, but still logged as some form of warning so that the user is aware of the issue? Or should they block the solve from proceeding as if they were true Foreign Key failures?

The "Empty Zones Or Groupings" [parameter](#) provides these three choices. If the parameter is set to "Exit with Error" then the "Empty Zones Or Groupings" validation table (in the "Advanced Validation Details" subsection) is populated. If it is set to "Remove With Warnings", then the empty zones and grouping information will be logged in Logging > Solution Logs > Error and Warning Logs. If it is set to "Remove Without Warnings" then such records will be quietly ignored.

## CARRIER OR LANE RESTRICTIONS

The purpose of a carrier is to compute the per-sku price for an arc. A carrier attempts to rate arcs in response to the rating requests made by rows in the [Lanes](#) table. If the [Lanes](#) table attempts to rate an arc that the carrier is unable to rate, then the rating will fail and a carrier specific failure will be logged. For performance purposes, each row in the Lanes table will only attempt to attempt rate arcs which the carrier appears likely to rate successfully. For this reason, the carrier specific failure logging will often be less than you might expect.

To illustrate this concept, consider two similar approaches for generating two shipment tiers. Assume that all the sites belong to exactly one of the Plant, Warehouse and Customer zones, and that all arcs will be either Plant-to-Warehouse or Warehouse-to-Customer. Finally, assume the price of each arc should be one dollar per sku mile. Let's consider two similar modeling approaches to generating the appropriate arcs.

### Lane Based Restrictions

Here the Lanes table has two records, one where Source Zone Name is Plant, Destination Zone Name is Warehouse, and the other with Source Zone Name of Warehouse, Destination Zone Name of Customer. The value one-dollar-per-mile (which is just the name of the carrier) is used for the Carrier field for both rows.

The Zone Carrier table then has just one record. The Carrier field is also one-dollar-per-mile. `*all*` is used for Source Zone Name and Destination Zone Name. The Cost Per Distance is 1.

This configuration will generate arcs from plants to warehouses, and from warehouses to customers. It will not generate any other types of arcs. That is, warehouses will not be able to ship to warehouses, nor customers to warehouses, nor any of the other 7 permutations beyond Plant-Warehouse, Warehouse-Customer. This configuration will also not generate any carrier specific failures, as the one-dollar-per-mile carrier will not be asked to rate any shipments for which it has no information.

### Carrier Based Restrictions

Here we reverse the configuration described above. We will have just one Lanes record, with Source Zone Name and Destination Zone Name of `*all*`. The Zone Carrier table will have two records, one where Source Zone Name is Plant, Destination Zone Name is Warehouse, and the other with Source Zone Name of Warehouse, Destination Zone Name of Customer. All other values will be as before.

This configuration will generate the same transportation arcs as before. Only Plant-Warehouse and Warehouse-Customer arcs will be generated. Moreover, because this example is so simple, the [Lanes](#) table will be able to easily determine which arcs the carrier can rate. In this case, there will be no carrier specific failures. For more complicated restriction patterns, the [Lanes](#) table will be forced to attempt arc ratings that fail to match any carrier records. Any such failed rating will generate a carrier specific failure.

It is up to the modeler to determine whether to use **Lane Based Restrictions** or **Carrier Based Restrictions**. The former method is generally more readable. However, if the nature of the restrictions is so detailed that many thousands or rows will be needed (as opposed to just the two rows in this example) then the latter method will be far more performant. For performance reasons, the modeler should avoid having a large number of records in the Lanes table.

With default parameter settings, the carrier specific failures will be logged in Logging > Solution Logs > Error and Warning Logs. If the "Generate Errors and Warnings" [parameter](#) is set to "Instead Of Optimization", then they will be treated as advanced validation errors and will populate the "Carrier Specific Failure" and "Carrier Specific Failure Summary" tables in the "Advanced Validation Details" subsection of the Validation tables. This logging can be further fine-tuned by populating the "Carrier Specific Failure Logging" table in the Input tables.

## CLEAN MODELING

The recommended modeling here is as follows.

- To the extent that's possible, reduce the number of Carrier Specific Failure warning messages that are generated.
- Populate the Carrier Specific Failure Logging so that the expected Carrier Specific Failures are all ignored. Thus, there are no Carrier Specific Failures in the Error and Warning section of log files.
- Set the Empty Zones Or Groupings parameter to Exit With Error.
- Populate the [Expected Zones And Groupings](#) input table so that there are no validation failures.
- For empty zones and groupings that aren't specified at all in the Sites/Products/Time Periods table (since they are empty) but are referenced in other tables, the entries in Expected Zones And Groupings will be used to ensure that those other references are not themselves a data entry error.

For even more rigorous data integrity checking, you can do the following.

- Set the Expected Zones And Groupings **parameter** to 'Needed For All'.
- Further populate the [Expected Zones And Groupings](#) input table so that there are no validation failures.
- For non-empty zones and groupings (excepting the singleton groupings that are generated from the Name column) the name used in the Sites/Products/Time Periods tables will have to be repeated (just once) in Expected Zones And Groupings. This is to guard against non-obvious misspellings. For example, if 199 Zone 1 records are populated with "Customer" and 1 is populated with "customer", then this will be apparent when the single "Customer" record in Expected Zones And Groupings fails to remove all the validation errors.

## Bottleneck Reports

The bottlenecks report is especially useful when troubleshooting infeasible models. The standard response to "The solver can't generate a solution to this model" is to follow the "Total Infeasibility" guidance from the **Parameter Details** documentation.

### Bottleneck Tolerance

The Bottleneck Tolerance parameter sets the threshold by which constraint utilization is screened in order to go into the bottlenecks report. For example, if the threshold is 75, then all capacity constraints that are at 75% or higher will be itemized.

### Generate Bottlenecks

The Generate Bottlenecks parameter can be set to "Never", "After Optimization", and "Instead Of Optimization". The last choice can be a bit tricky, as explained below.

#### Generate Bottlenecks = Instead Of Optimization

The purpose of this parameter option would to re-generate the Bottlenecks Report based on a different Bottleneck Tolerance. When you run "Solve" with "Generate Bottlenecks" set to "Instead of Optimization", the normal solve process is almost completely bypassed. Subsequent to input data validation, the solution tables are also ingested into memory, and the input/solution pair is passed just to the bottlenecks report generation routine. As you can imagine, this process might fail if the input data has been significantly altered since the last true solve operation (i.e. since the solution was created).

For this reason, we recommend that you do **not** use this parameter option for anything other than regenerating the Bottlenecks Report with a different tolerance. In other words, if the only input data that has been altered since the solution was created is the Bottleneck Tolerance, then you can run with this parameter option as a time saving option to re-running the whole optimization process. As you might imagine, if you run with this option after deleting records for the input data then the solve process will fail, and is likely to generate an ugly error message when it fails. Rest assured, the current solution isn't destroyed when this happens, but you will need to re-run the solve in the normal way in order to create a new Bottlenecks Report.

### Development Notes

Here is an itemization of some constraints whose utilizations will be reported on with the Bottlenecks Report.

- Max Supply for Production table.
- Max Supply for Production By Production lines.
- Max Hours for Aggregate Production table.
- Max Hours for Aggregate Production By Production Line table.
- Max Volume for the Max Volume input table.
- Following fields from the Min Max Flows table:
  - Max Flow
  - Max Percentage Supplier

## High Level Table Summary

**Remember**, other than the [Parameters](#) table, all of the tables use 999999999 (nine 9s) as a flag for positive infinity, and similarly -999999999 for negative infinity. The Value field of the Parameters table is allowed to mix text and numbers, and thus uses "inf" and "-inf" as flag values for positive/negative infinity.

## Core Data

- **Sites** - Defines the sites, as well as the zones. The solver recognizes `*all*` as a zone containing all sites, and it also recognizes that the Name field of the Sites table is a singleton zone definition. Thus there is no need for a user to use a Zone X column to define either a zone with all sites or to define singleton zones.
- **Products** - Defines the products as well as the product groupings. Similar point applies regarding `*all*` and the Name field.
- **Production Lines** - Defines the production lines and the production line groupings. Each production line is associated with a site. Sites can either produce directly, or via their encapsulated production lines, or both. There is no automated way to make singleton production line groupings, but the `*all*` string is recognized as a comprehensive grouping.
- **Time Periods** - Defines the time periods as well as the time period groupings. Similar point applies regarding `"*all*"` and the Name field.
- **Parameters** - A simple Name/Value table. The non-advanced parameters will auto-populate with default values and descriptions. The auto-populated description entries should be helpful, but there is also a dedicated Parameters documentation page.

## Demand and Production

- **Demand** - Indexed by Site, Product, Time Period. Has Min Demand, Max Demand, Revenue. It's common to want Min Demand=Max Demand, which is why the data uploader for the Demand table is smart enough to recognize a table with the Demand field, but no Min Demand or Max Demand.
- **Production** - Indexed by Site, Product, Time Period. Has Min Supply, Max Supply, Cost and Rate. Rate is the units of goods that can be produced by hour of Aggregate Production (see below) and thus the default of infinity here indicates no Aggregate Production capacity used for production.
- **Aggregate Production** - Indexed by Site, Product Group, Time Period. Has Min Hours and Max Hours. If Min Hours is positive, then constraint only applies for time periods where the site is open.
- **Production By Production Lines** - Similar to the **Production** table, except for Production Lines.
- **Aggregate Production By Production Lines** - Similar to the **Aggregate Production** table, except for Production Lines.
- **BOM** - Bill of Materials table. Indexed by Component and Assembly product. Defines the Quantity of the Component needed to produce one unit of the Assembly. If you use the BOM table, then you can't use any of the recipe tables.
- **Recipes** - Defines Recipes for BOM production. Defines Recipe Name and groupings. Similar point applies regarding `*all*` and the Name field.
- **Recipe BOM** - Indexed by Recipe, Component and Assembly. Defines the Quantity of the Component needed to produce one unit of the Assembly when production is done with the Recipe. Use this field in lieu of **BOM** when recipe modeling is needed.
- **Production Line Recipe Prohibitions** - All recipes can be used at all production lines by default. Populate this table to black list any production line from using any recipe. Indexed by Production Line Group, Recipe Group.
- **Zone Recipe Prohibitions** - Similar to **Production Line Recipe Prohibitions** except for site based production. Indexed by Zone, Recipe Group.

## Transportation

There is a dedicated page just for Lanes/Carriers. Thus the Lanes/Carriers descriptions are kept short here.

- **Arc Overrides** - Can be used to provide a per-sku shipping cost directly, instead of by using lanes and carriers. Alternately, both techniques can be used, but the arcs defined here will have higher priority.
- **Lanes** - Similar to LNP (LogicNet Plus) modeling, lanes are iterated over in order of precedence. Lower precedence goes first, and sorting by Name breaks the tie if lanes shares precedence. Unlike LNP, a lane only specifies only one carrier. If you want to use blending or cheapest, then you can select a blending or cheapest carrier (i.e. composite carrier). Otherwise, select a normal carrier.
- **Blended Carrier** - Defines the Blending Weight with which different individual carriers should be blended together for rating.
- **Cheapest Carrier** - Defines which different individual carriers should be compared against each other for rating.
- **Fed Ex Carrier** - To be implemented.
- **SMC Carrier** - To be implemented.
- **Truck Carrier** - Defines Cost Per Truck, Cost Per Truck Distance (if both provided, they add together) and Truck Size. This carrier will use an Average Shipment if one is provided (checkout the [Average Shipment Logic](#) section to understand more about it).
- **UPS Carrier** - To be implemented.
- **Zone Carrier** - A simplification of Truck Carrier. A zone based way to define Cost Per Unit and Cost Per Unit Distance.
- **Distance Weight Break Carrier Costs** - Can define a weight by distance table. Needs average shipment to be provided (to look up along the weight axis).
- **Carrier Specific Logging** - Indexed by Lane, Carrier, Detail. Detail can be one of five flagging strings for the different types of lane/carrier error information. Defines the Logging Level, which must be one of "Itemized", "Summarized" or "Ignored".
- **Arc Shipment Data** - one of the ways to define Average Shipment values. (See separate documentation).
- **Zone Group Shipping Bounds** - Indexed Zone Name and Product Group. Defines Max Inbound Distance, Max Outbound Distance. Not yet fully implemented - **don't populate**.
- **Min Max Flows** - Indexed by Source Zone, Destination Zone, Product Group. Defines the Min Flow, Binary Min Flow (i.e. a floor that that applies only if there is a positive value), Max Flow, Max Number of Source Sites, Max Number of Destination Sites.
- **Distance Overrides** - Indexed by Source, Destination. Specifies Distance and Distance Factor. If you provide a Distance you don't need to provide a Distance Factor.
- **Lane Destination Assignments** - Indexed by Destination, Product Group. Defines the Source. Allows a destination site, product group to receive inbound shipments from only one source (this table is a alternative to the Min Max Flow table to constrain flows).

## Sites and Lines

- **Zone Open Restrictions** - Indexed by Zone, Time Period Group. Specifies Min Number Open Sites, Max Number Open Sites.
- **Site Product Costs** - Indexed by Site, Product Group. Specifies SKU Cost, Weight Cost, Size Cost. See the [Site Product Costs Table](#) documentation to understand more about it
- **Site Product Inv Info** - Indexed by Site, Product. Specifies ITOR (Inventory Turn Over Rate), Holding Cost, Starting Inventory, Ending Inventory.
- **Site Product Prohibitions** - Indexed by Site, Product Group. The primary key entries themselves define the data here - its a black list for site-product-group pairs.
- **Production Line Open Restrictions** - Indexed by Production Line Group, Time Period Group. Defines Min Number Open Production Lines, Max Number Open Production Lines.
- **Max Volume** - Indexed by Site, Product Group, Time Period. Defines the Max Volume (defined as average inventory volume) and Fixed Cost (to be applied if average inventory volume is positive).

## Terminology

### ARC

For a typical model, an [Arc](#) is a (source site, destination site, product) triplet with an associated per-sku transfer cost. For a multi-time period model with the In Transit Inventory parameter set to Allowed Between Periods, then an arc is a (source, destination, product, time-to-ship-offset) fourplet with an associated cost. Either way, when the core optimization engine runs, it looks to the arcs to determine if goods can be shipped between two sites, and how much that shipment will cost. These arcs are generated prior to launching the Gurobi-based solve step.

For the typical modeler, arcs represent an intermediate result. They are created from the information in the lanes and carriers, and then passed to a core Gurobi process in order to solve for a final solution.

That said, are a few alternatives to this usage pattern.

- A modeler might wish to eschew the lanes/carriers processing step altogether. In that case, arcs (with time-to-ship-offset values of zero) can be entered directly in the [Arc Overrides](#) input table.
- A modeler might wish to blend the two techniques. In this case, any arcs generated by the Arc Overrides table take precedence over arcs that would otherwise be generated by lanes and carriers.
- A modeler might wish to examine the arcs that were generated from lanes and carriers without going through the entire solve process. In this case, set the Generate All Possible Arcs Report parameter to Instead Of Optimization. If you are just learning how to use lanes and carriers to generate arcs, or if you are modeling something unusually tricky, you are strongly advised to take full advantage of this option. Lanes and carriers are powerful, but like all powerful things, they can be dangerous. The All Possible Arcs Report is an excellent tool for developing an expert level understanding of this functionality.

### LANES

Lanes are entries in the [Lanes](#) table. There is a collection of potential arcs implied by the Source Zone, Destination Zone, and Product Set of each lane. (If in-transit inventory is allowed between periods, then the Transit Time also effects the set of potential arcs).

When arcs are being generated by the app, it iterates over the Lanes table in lowest-to-highest order from the Precedence field. Specifically, this means that a row with Precedence 1 will be handled before a row with Precedence 2.

For each row of the [Lanes](#) table, the app will iterate over the potential arcs. If the arc was generated by a previous row in the Lanes table (or by a row from the [Arc Overrides](#) table) then this arc is passed over. Similarly, the arc is passed over if the source-destination distance is larger than the [Max Distance](#) value. Otherwise, the app asks the carrier referenced by the [Carrier](#) field to rate this arc. If the carrier successfully rates the arcs, then the arc is generated. Otherwise, the carrier specific failure is logged, and the app will be free to generate this arc based on a subsequent row from the Lanes table.

Note that a carrier look up failure is not always logged, as the app will apply filtering logic in order to minimize the potential arcs that need to be looped over. The logging of carrier specific failures is discussed in more detail in the [Best Modeling Practices](#) page.

Finally, we note that the modeler is discouraged from entering large number of records in the lanes table. As there is a fixed processing overhead for each lanes record, the run times can suffer if the number of lanes records is in the thousands (or higher). Fine grain rating information that requires 10,000 records or more should be handled at the carrier level.

### CARRIERS

A detailed documentation for Carriers can be found at the [High Level Table Summary](#) and the [Best Modeling Practices](#) pages.

## Parameter Details

The [Parameters](#) table contains a Description field where you can see a brief explanation of each parameter. This page contains additional information that complements the Description field. You can take a look at the [Parameter Explanation](#) section, at Input Tables -> Core Data -> [Parameters](#), to learn more about how each parameter works.

- The MIP Gap gets divided by 100 before being passed to the solver. That is to say, its a percent, not a ratio.
- The Total Infeasibility option for the Objective parameter is there to help [Troubleshoot Infeasible Models](#).
- **Autoscaling** can be important for numerical stability of Mixed-Integer Programming (MIP) models. If the user enters the demand in teaspoons and thus has huge numbers, rescaling what a "unit" means everywhere can really help.
- This is typically done automatically by the Optimization logic. "Autoscaling" is thus an advanced parameter that a sophisticated user would add to the Parameters table for troubleshooting.
- Some observations related to modeling [High Service Demand](#).
- When studying a feasible solution, we say the demand was met with high service if the site shipping into the demand point was within a specific distance. This distance is set via the "High Service Demand Threshold" parameter. Since this parameter defaults to zero, by default there is no way to meet demand with high service (unless its a zero distance shipment).
- You can set the Percentage High Service Demand as an objective, in which case it will maximize the demand that meets high service.
- You can set the Min Percentage High Service Demand as a global constraint for the model.
- As with any KPI, we advise you to not restrict it and set it as the objective at the same time.
- Like explained above for High Service Demand, the **Critical Zones** is similar.
- The basic use case here is to give the user the option to open as few "critical sites" as possible. This is what happens when you set the objective to be Critical Sites Opened.
- We don't give you a parameter to control the maximum number of critical sites opened, but you can control this with the [Zone Open Restrictions](#) table.
- By default, all sites are critical sites. The Zone For Critical Sites lets you make a more refined selection.
- Check the [Optimization Objectives](#) page to understand more about modeling Critical Zones.
- **Empty Zones Or Groupings** parameter:
  - If set as Remove With Warnings then you'll see a partial list of the empty references in Logging > Solution Logs > Error and Warning Logs.
  - If Exit With Error then you'll see a full list of the empty references in the "Empty Zones Or Groupings" validation table (in the "Advanced Validation Details" subsection).
- **Generate Errors and Warnings** parameter:
  - Like for the parameter above (Empty Zones or Groupings), Carrier and Arc errors can be controlled with this parameter. If this parameter is set as Always, then a partial list of carrier/arc errors will end up in Logging > Solution Logs > Error and Warning Logs. If its Instead Of Optimization, then a full list will end up in the "Carrier Specific Failure" and "Carrier Specific Failure Summary" tables in the "Advanced Validation Details" subsection of the Validation tables.

## Site Product Costs Table

[Site Product Costs](#) input table can be a little confusing. Bear in mind that this table is indexed by site, product-group, and thus there might be some site-product pairs that are referenced by multiple rows. The rule here is that the costs here are additive for such site-product pairs.

First, lets look **SKU Cost** and **Weight Cost**. These costs are most similar to each other, in that they are "handling" costs and not inventory costs. Goods pay these costs when they are physically inbound or outbound, but not when they are stored in starting or ending inventory. The concept of "inbound" is generalized to include goods that arrive via inbound shipments, or goods that are simply produced on site. Similarly, the concept of "outbound" is generalized to include goods that leave via outbound shipments, or goods that are consumed as demand satisfied on site or goods that are consumed as components during a Bill-Of-Materials production on site. Finally, note that a full handling cost is applied only when the good experiences both "inbound" and "outbound". That is to say, if there are 50 units of inbound shipments and 50 units of outbound shipments, then the solution Site Product Details report will list a Flow of 50, and thus all 50 units will pay the appropriate handling penalty. But if there are 50 units of inbound shipments, and zero units of outbound shipments (and thus the ending inventory is 50 units higher than the starting inventory for that time period) then the flow is just 25 and those 25 units will pay the appropriate handling penalty.

Now that we understand how the handling penalty works, the only difference between SKU Cost and Weight Cost is that the latter is scaled by the Weight field on the [Products](#) table and the former is not scaled by anything, being measured by units.

If you're wondering why there are fields to measure a handling cost by SKU and Weight, but not by Warehouse Volume, Shipment Volume, or Adjusted Volume, it is because the Warehouse Volume is taken into account with Size Cost (see below), and Shipment Volume and Adjusted Volume are both transportation units.

We also need to address the **Size Cost** field. This field creates an additional cost for the Average Inventory of the site. That is to say, for a given site-product pair, you multiply the sum of all the relevant Size Cost entries by the product of the solution Average Inventory (for this site-product pair) and the Warehouse Volume (for the product). Recall that average inventory is itself computed as Flow/ITOR (ITOR: Inventory Turn Over Ratio), plus the average of starting and ending inventory.

The **Dwell Cost** field specifies a per-warehouse-volume cost for holding inventory. This cost doesn't apply to any form of inbound or outbound shipments. In other words, its like Size Cost, except without the ITOR scaled contribution of flow.

Finally, note that the [Handling Costs](#) table is different. In that table, production, demand satisfied, and component consumption are not used when computing handling costs. That table requires you to specify "Inbound" and "Outbound" for the "Direction Type" field. Although local production often behaves similarly for inbound flow, and component consumption and demand satisfied behave similarly to outbound flow, they aren't the exact same thing. Inbound flow and outbound flow can always be fully computed from the arcs table.

## Optimization Objectives in Foresta Supply Chain Network Optimizer

### Introduction

When running an optimization model, it's possible to set different optimization objectives. You can optimize your supply chain by maximizing revenues, or you can just minimize costs, regardless of the incurred revenue of it, or maybe you just want to know if your supply chain can handle your customer necessities with the physical resources made available by the business. Based on that, the user can set different optimization objectives, by using the parameter Objective at the Parameters table (take a look at the section Input Tables/Parameters to learn about all the parameters that are available to configure within the model). Each time that the user changes the Objective parameter and runs the model, model outputs will be different, just because the model will prioritize one goal instead of the other one, and that's totally up to the user to decide which objective it's more important based on what the business wants to achieve.

In this section, each possible value for the parameter Objective will be explained, and its impacts in the optimization. Each possible value will require different inputs from the user, and will result on different model outputs. Let's start from the beginning.

Imagine a situation where you're running a Network Optimization model for the first time, you have these many customers demanding these different products and your supply chain has a lot of different constraints, as production lines capacities, inventory capacities, transportation capacities, etc. You're very excited to run the model for the first time and wants to show to your boss that it's possible to reduce supply chain costs of your network, so you set the parameter Objective as Total Cost to minimize costs of your network. After running the model, the system returns a message saying that the solution was not feasible, meaning that it's impossible to satisfy your customer demand values with your supply chain constraints. You have many warehouses and many plants in the network, and you really don't know what may be causing the infeasibility. In this situation, you would probably change the Objective parameter to Total Infeasibility, so the model can return a solution showing to you what exactly is causing the infeasibility, or a scenario where your supply chain can satisfy your customer demand (if feasible), even if this scenario is not the optimal one.

So changing this parameter and consequently the model objective will be part of your life as a modeler, and is up to you to have the necessary confidence to change it and understand its impacts. For that, you can consult whenever you want the section below, where each possible value is explained and its impacts at the optimization.

### Total Cost objective

The solution will minimize costs, *SATISFYING\** the Minimum Demand.

### Total Profit objective

The solution will maximize profits by balancing costs and revenue, *PURSUING\** the Maximum Demand values.

### Total Revenue objective

The solution will maximize the revenue, *SATISFYING\** the Maximum Demand, regardless of the costs incurred.

### Total Cost vs Total Profit vs Total Revenue

Note that minimizing costs is not the same as maximizing profits. When minimizing costs, the model will satisfy the *Minimum Demand* values, regardless of the *Maximum Demand* and *Revenue*, because, of course, that would infer at the scenario with the smallest costs, based on this minimum demand constraint (remember that without demand minimum constraints the optimal cost would be 0, which would be related to the scenario where the network is deactivated). However, when maximizing profits, the model will consider both Costs and Revenues and optimize maximizing the difference between Total Revenue and Total Costs. If Costs > Revenue (negative profit) at all demand values between minimum and maximum demand fields, the model will satisfy the *Minimum Demand*, to reduce the negative profit. Take a look at the example below to have more clarification on the matter.

Let's suppose that you have the situation described below, with a *Minimum Demand* of 10,000 units and a *Maximum Demand* of 20,000 units. The model was set considering only Production and Transportation costs. Setting up the Objective Parameter as Total Cost, Total Profit and Total Revenue, we can see its outputs.

### Input table: Demand

Site	Product	Time Period	Min Demand	Max Demand	Revenue
C_Albany	Bread	P1	10000	20000	5

Outputs:

Objective	Demand Satisfied	Demand Shortfall	Total Cost	Total Revenue	Total Profit
Total Cost	10,000	10,000	\$ 125,000	\$ 50,000	\$ -75,000
Total Profit	10,000	10,000	\$ 125,000	\$ 50,000	\$ -75,000
Total Revenue	20,000	0	\$ 250,000	\$ 100,000	\$ -150,000

We can see that increasing revenues is not the same as increasing profits, simply because the cost to produce and transfer the product is higher than the revenue of selling this same one product. Let's make a small change at the [Demand](#) table, increasing the [Revenue](#) by three (3) times its original value, and let's run again the model changing the objective parameter in the same way as before.

Input table: [Demand](#)

Site	Product	Time Period	Min Demand	Max Demand	Revenue
C_Albany	Bread	P1	10000	20000	<b>15</b>

Outputs:

Objective	Demand Satisfied	Demand Shortfall	Total Cost	Total Revenue	Total Profit
Total Cost	10,000	10,000	\$ 125,000	\$ 150,000	\$ 25,000
Total Profit	20,000	0	\$ 250,000	\$ 300,000	\$ 50,000
Total Revenue	20,000	0	\$ 250,000	\$ 300,000	\$ 50,000

Very interesting! Now the outputs of the Total Profit scenario are the same as the Total Revenue, that happens because now the unit [Revenue](#) is bigger than the unit cost. Therefore, we can get to some conclusions after this exercise:

- When Objective = Total Cost, the demand driver will be the [Minimum Demand](#) field\*.
- When Objective = Total Revenue, the demand driver will be the [Maximum Demand](#) field\*.
- When Objective = Total Profit, the demand satisfied will be somewhere between the Minimum and Maximum demand, or one of those values\*.
- If Unit Revenue < Unit Costs, Total Profit scenario outputs would be the same as the Total Cost scenario\*.
- If Unit Revenue > Unit Costs, Total Profit scenario outputs would be the same as the Total Revenue scenario\*.



Try to run the same model 3 times, as we did in the example above, with the Objective parameter set as Total Cost, Total Profit and Total Revenue, and see what happens. You can get good business insights by doing that, and understand a little bit more about how the model behaves!

#### Total Demand Satisfied objective

The model will satisfy the demand in the [Maximum Demand](#) field, if filled, or the [Minimum Demand](#), regardless of the revenue or profits incurred\*.

#### Percentage High Service Demand objective

With this objective, you can optimize the network by prioritizing Service Level indicators. You can do that by setting up the [Parameter](#) named High Service Demand Threshold, which will dictate what would be the distance between the source and the demand site that can be considered as High Service (go to the [High Service Demand](#) page to understand more about this feature). Let's explore a practical example to clarify more about how to use this objective:

There are 2 customers, one at Albany and one at Buffalo, and 2 warehouses that can supply these both customers, one at Boston and the other one at Cleveland. All 4 possible combinations of Source and Destinations are shown below, with its distances:

Source	Destination	Distance (miles)
W_Boston	C_Albany	139
W_Boston	C_Buffalo	391
W_Cleveland	C_Albany	414
W_Cleveland	C_Buffalo	180

Input table: Demand

Site	Product	Time Period	Min Demand
C_Albany	Bread	P1	10,000
C_Buffalo	Bread	P1	5,000

Imagine a case where the business believes that, to guarantee high service levels, it's necessary to satisfy the demand from sources within 150 miles distance from the customers. So we can set up the [Parameter](#) table with the following parameters:

- Objective = Percentage High Service Demand
- High Service Demand Threshold = 150

By using this configuration, the model will try to satisfy the demand from sources within 150 miles. Running the model and analyzing the outputs, we can see its results:

Output table: Parameters

Key	Value
...	...
Percentage High Service Demand	66.6667

Output table: Arcs

Source	Destination	...	Distance
W_Boston	C_Albany	...	139
W_Cleveland	C_Buffalo	...	180

We can see that the solution could satisfy 66.67% of the demand within the High Service Demand Threshold of 150 miles, that would be the demand of the customer at Albany ( $10000 / 15000 = 66.67$ ). As the lowest distance from any source (Boston or Cleveland) to the customer at Buffalo is 180 miles, the model could not reach a Percentage High Service Demand of 100%, simply because it is not possible with the sources available.

Sometimes it can be very straight forward what would be the best solution. In the case above, we can notice that Boston is closer to Albany than Cleveland, and Cleveland closer to Buffalo than Boston. But imagine in a network that you have 200 customers and 15 warehouses, it would be that easy to notice that based on all existing possibilities of lanes?



Try to compare the scenario above with the scenario where Objective = Total Cost. You can have good insights about the impacts on Costs when increasing Service Levels! That would be what people usually call a Trade-Off Analysis (Costs vs Service Levels).

#### Critical Sites Opened objective

With this parameter, the model will try to reduce by the maximum amount as possible the number of sites opened for what the user defines as the Critical Zone. To do that, you need to specify what zone would be that one using the [Parameter Zone For Critical Sites](#). Let's take a look at an example to clarify how does that objective works:

Imagine that your business has the potential to open 3 Warehouses at the East Coast. Cleveland, Boston and Springfield.

#### Input table: Sites

Name	...	Open Close Status	...	Zone 1
W_Cleveland	...	Potential	...	East US
W_Boston	...	Potential	...	East US
W_Springfield	...	Potential	...	East US

When you run the model with the Objective parameter set as Total Cost, the solution opens the Cleveland and Springfield warehouses. But when you set the objective parameter as Critical Sites Opened, and define East US as the Zone For Critical Sites, the model only opens the warehouse at Springfield. That happens because the model will minimize the number of Open Sites for Critical Zones, even if that returns a solution that is not optimal from a cost or revenue perspective.

#### Input table: Parameters

Key	Value	...
Objective	Critical Sites Opened	...
Zone For Critical Sites	East US	...

#### Total Infeasibility objective

The model will prioritize its feasibility, returning an infeasibility report that shows what's causing the infeasibility, and a feasible scenario in case if the model is feasible. When at this mode, the model *NOT* necessarily optimizes the network. The model will simultaneously seek to maximize demand satisfied (within the "Min Demand", "Max Demand" range, at the [Demand](#) table) and also to minimize the usage of the "Violation" variables.



The output of the model when running with the objective set as Total Infeasibility maybe *NOT* return the best scenario in terms of costs, profits or revenue. It will return a feasible scenario. So you should *NOT* expect a "good" or reasonable result when running in this mode. It's recommended to set the objective as Total Infeasibility only for feasibility diagnostic purposes.

\*These affirmations are based on a model without constraints related to maximums, as capacities, as these can be restrictive for fulfilling demand.

## Modeling Constraints in Foresta Supply Chain Network Optimizer

### Introduction

One of the most beautiful features of a Network Optimization model is the constraint modeling. In real life there may be a bunch of restrictions that can derail the perfect unrestricted solution. Maybe the customer demand for snow globes at Christmas can not be satisfied because your supply chain network does not have capacity to produce all these snow globes at November, so you need to consider that production capacity when modeling and then plan to start the production at September, to form pre-build stock, for example.

The example above explores constraints related to maximums, as capacities. But there may be constraints related to minimums too, these kind of constraints must be satisfied as well, if necessary. Safety stocks requirements, for example, are very common, and they are related to minimum inventory levels that must be satisfied by the network.

The Foresta Network Optimization solution allows you to model different supply chain and business constraints, as listed below.

### Types of constraints

- **Production constraints**
- Production capacity
- Production minimum usage
- Minimum and maximum number of open production lines
- **Inventory constraints**
- Inventory capacity
- Ending inventory requirements
- **Demand constraints**
- Maximum demand
- Minimum demand
- **Transportation constraints**
- Truck capacity
- Average shipment
- Maximum shipping distances
- **Flow constraints**
- Maximum flow
- Minimum flow
- **Open Sites constraints**
- Minimum number of Open Sites
- Maximum number of Open Sites
- **Handling constraints**
- Site-Product handling restrictions
- **Parameters constraints**
- Maximum Total Cost
- Minimum Percentage High Service Demand
- Minimum Total Profit
- Minimum Total Demand Satisfied

### Production constraints

Production constraints are very common in many supply chains. Production facilities usually have different production lines with different capacities and limitations, and it is possible to model these capacities by volume or by hours, in the levels of site, product and time period, or site, product, time period and production line. The modeling allows to fix minimum utilization as well, as a way to guarantee a minimum usage that sometimes is a business requirement.

**Production Capacities** and **Minimum Usage** can be modelled by using the following input tables:

- **Production**: to model constraints at the site, product and time period level, by flow units of measure.
- **Aggregate Production**: similar to the Production table, but the Product field would be aggregated by product groups, and the unit of measure would be hours, instead of flow units.
- **Production by Production Lines**: to model constraints at the site, product, time period and production line level, by flow units of measure.
- **Aggregate Production by Production Lines**: similar to the Production By Production Lines table, but the Product field would be aggregated by product groups, and the unit of measure would be hours, instead of flow units.



The main difference between the aggregated tables from the non-aggregated ones is the Product field. In the aggregated ones it is possible to model capacities by product groups, since it is very common in many supply chains sites or production lines that can produce different products. Try to use this feature to make your life simpler as a modeller!

For **Minimum and Maximum Number of Open Production Lines**, you can use the [Production Line Open Restrictions](#) table, where you can set, by groups of production lines and time periods, what would be the limits for minimums or maximums of production lines necessary to be opened.

#### Inventory constraints

Inventory constraints will dictate how much inventory each site have to hold (minimum requirements) or can hold (maximum capacities).

Use the [Max Volume](#) table to set **Inventory Capacities** at the site-product (or site-product group) level. To set **Ending Inventory** requirements (as safety stocks), use the [Site Product Inventory Information](#) table (site-product level).

#### Demand constraints

Demand constraints can be configured using the [Demand](#) table. Use the **Minimum Demand** field to set the demand that the model *MUST* satisfy, and the **Maximum Demand** one to set the upper bound limit of demand attendance (take a look at the [Optimization Objectives](#) page to learn more about how the demand table configuration will influence the modeling outputs based on the objective parameter).

#### Transportation constraints

Transportation constraints will guarantee that the model will flow products as the way as is in real life. If not modelled, the model can behave in some crazy ways, like shipping 200 bathtubs in just one truck, for example. That is why this kind of constraint is usually called *Soft Constraint*, because they will probably *NOT* cause infeasibility in the solution, but they are very important in terms of reasonableness and cost representation.

Fields as **Truck Size** and **Average Shipment** will dictate how many goods a shipment will transfer between sites (take a look at the [Average Shipment Logic](#) to understand a little more about how this field works within the model).

For **Distances**, you can set limits as well, specially to make your model more reasonable and to reduce transportation costs. You can use the [Zone Group Shipping Bounds](#) table to define **Maximums Inbound or Outbound Distances** in an aggregated way, by zones (groups of sites) and product group. In case you want to model the same limits but disaggregated, by source, destination and product group, you can use the [Lanes](#) table, using the **Maximum Distance** field.

#### Flow constraints

Flow constraints will dictate limits for how much product can flow between different sites. Usually distribution centers have specific quantity of docks, or maybe the quantity of docks is irrelevant, but there are restrictions of labor to ship all these products in a same day, for example. Use the [Min Max Flow](#) table to model the **Maximum or Minimum Flow** units that a site can ship, by source, destination, product and time period (or groups of these elements).

It can be tricky to use this table. Thinking about that, we made a whole page explaining how to use it, and how to avoid mistakes and conflicts involving different constraints fields. Take a look at the [Flow Constraints](#) page to understand more about how to model flow constraints using the [Min Max Flow](#) table.

#### Open Sites constraints

Sometimes the business do not desire to open to many sites, even if that would reduce supply chain costs. Or maybe you just want to force the model to open or not a quantity of sites within a range, to see what would happen with the network. To do that, you can restrict the **Minimum or Maximum Number of Open Sites**, per zone and time period group, by using the [Zone Open Restrictions](#) table.

#### Handling constraints

There are cases where a product can not be handled at a specific site, maybe for sanitary purposes or because simply there are no equipments to handle these products within the site. So, to model **Handling Restrictions**, you can use the [Site Product Prohibitions](#) table to do that, informing which products can *NOT* be handled within specific sites.



Maybe it is easier to tell the model what he can *NOT* do instead of what he *CAN* do. Think about that when building your model, you can facilitate a lot your work by applying this mindset!

#### Parameters constraints

Constraints that can be set by [Parameters](#) are the ones that will be applied at the highest level possible - at the scenario level - differently than the other constraints mentioned above. It is possible to restrain the indicators **Maximum Total Cost**, **Minimum Percentage High Service Demand** (go to the [Percentage High Service Demand](#) section at the [Optimization Objectives](#) page to understand more about how that percentage works within the model), **Minimum Total Profit** and **Minimum Total Demand Satisfied**.

## Modeling Flow Constraints in Foresta Supply Chain Network Optimizer

### Introduction

As explained in the [Modeling Constraints](#) page, flow constraints will limit how much product can flow between different sites. We have made a whole page about this kind of constraint because using the table [Min Max Flow](#) can be tricky sometimes. So we are going to take a deep dive in this matter to make everything regarding this modeling more clear for you.

### Minimum Flow and Binary Minimum

At the [Min Max Flow](#) table, you can model minimum flows by using two different fields, [Min Flow](#) or [Binary Min](#):

1. **Min Flow:** the model will flow quantities *ABOVE OR EQUAL TO* ( $\geq$ ) the value that you fill.
2. **Binary Min:** the model will flow quantities *ABOVE OR EQUAL TO* ( $\geq$ ) the value, *OR* the model will flow 0 (zero) units.

Let's explain the difference between both with an example. Imagine this situation, a customer in Albany, NY, can be supplied by both warehouses at Boston, MA, or Cleveland, OH. Boston is much closer to Albany than Cleveland, so the transportation costs are lower when shipping from Boston, as you can see at the Notes field. To ship products from Boston to Albany, it costs \$ 5 per ounce, and \$ 10 when shipping from Cleveland. Both warehouses have huge outbound capacities, so we fill the [Max Flow](#) field with a big number to *NOT* constrain maximums.

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	Active	Min Flow	Binary
W_Boston	C_Albany	*all*	P1	True	0	0
W_Cleveland	C_Albany	*all*	P1	True	0	0

When solving the model with the [Parameter](#) objective set as Total Cost and a [Demand](#) of 5000 units of Bread for the customer at Albany, we can see that all the demand it is being satisfied by Boston, as expected.

Output table: Arcs

Source	Destination	Product	...	Flow
W_Boston	C_Albany	*all*	...	5000

Ok, the model flows products by the cheaper path that it can find, very obvious. But imagine a situation where the business already has a warehouse at Cleveland, with an operation already running at fullest, and Boston warehouse is facing inventory space issues, because this warehouse is already with a high inventory occupation, and you need to clear some space at the site. So you set a [Min Flow](#) of 1000 ounces to force the model to flow products by Cleveland. The new configuration of the [Min Max Flow](#) table would be:

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	Active	Min Flow	Binary
W_Boston	C_Albany	*all*	P1	True	0	0
W_Cleveland	C_Albany	*all*	P1	True	1000	0

So now you run the model again to see what happens.

Output table: Arcs

Source	Destination	Product	...	Flow
W_Boston	C_Albany	*all*	...	4000
W_Cleveland	C_Albany	*all*	...	1000

Cool! The model now "reserves" 1000 ounces of demand to flow by Cleveland, and the remaining 4000 units of demand would flow by Boston, as expected.

Ok, this is very obvious. If the user sets a minimum the model will obey his/her wishes, of course. But let's imagine other scenario, the business at the moment only have the warehouse at Boston, and is considering to rent a warehouse at Cleveland, so you want to set up a situation where the model flows some minimum at Cleveland or nothing at all, because it does not worth to rent a warehouse in Cleveland to flow only 200 ounces of products, for example. The costs to rent it would be so much higher than the revenue obtained by satisfying the demand at Albany.

In this case you would probably use the field **Binary Min**. So let's change the input in the **Min Max Flow** table with value of 1000 units at the **Binary Min** field, instead of using the field **Min Flow**:

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	Active	Min Flow	Binary
W_Boston	C_Albany	*all*	P1	True	0	0
W_Cleveland	C_Albany	*all*	P1	True	0	<b>1000</b>

Running the model again, we can see the impacts of the alteration.

Output table: Arcs

Source	Destination	Product	...	Flow
W_Boston	C_Albany	*all*	...	5000

Now things get interesting. The model decides to flow nothing at all by Cleveland. In terms of costs, it does not worth it to flow products by Cleveland when you give an option to the model to do the minimum or flow 0 units by the site. The output would be the same of the scenario where there is no minimum! Think about how you can use the **Binary Min** field to help your business to get some insights.

You must be thinking: "If I fill both **Min Flow** and **Binary Min** fields, what would happen?". Let's try that!

First, let's consider a situation where **Min Flow < Binary Min**, setting up a value of 500 ounces in the **Min Flow** field at maintaining the value of 1000 ounces at the **Binary Min** field:

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	Active	Min Flow	Binary
W_Boston	C_Albany	*all*	P1	True	0	0
W_Cleveland	C_Albany	*all*	P1	True	<b>500</b>	<b>1000</b>

Running the model again, we can see the new results.

Output table: Arcs

Source	Destination	Product	...	Flow
W_Boston	C_Albany	*all*	...	4000
W_Cleveland	C_Albany	*all*	...	1000

Interesting... Now the model decides to flow the quantity that is configured at the **Binary Min** field by Cleveland. It makes sense, because the model can only flow 0 or a value equal or above than the value of the **Binary Min** field, but the **Min Flow** value is prohibiting the model to flow 0 units, because the minimum would be in fact 500 ounces, so reasonably the model decides to flow the quantity of 1000 ounces (remember by the last scenario that the optimal solution decides to flow 0 units when filling only the **Binary Min** field).



Be careful when using together **Binary Min** and **Min Flow** fields. In the case above, would make more sense just to set a value of 1000 units at the **Min Flow** field. The effect would be exactly the same!

Ok, now we know that the situation where **Min Flow < Binary Min** is the same as to set a Minimum Flow value at the **Min Flow** field. But if **Min Flow > Binary Min**, what would happen? Let's try it!

### Tip

It is always valid to remember that you can try to test each possible scenario within the model. You can just change data in the tables or you can create new scenarios, if you don't want to mess with the data of a specific scenario, for example. Be curious! You can find nice opportunities and understand more about the model as you manipulate it. Remember that you can always go back at any time!

The new configuration of the **Min Max Flow** table would be set as shown below. The **Min Flow** field would receive a value of 1000 ounces and the **Binary Min** the value of 500 ounces:

Source Zone Name	Destination Zone Name	Product Group *all*	Time Period Group	Active	Min Flow	Binary
W_Boston	C_Alternate	*all*	P1	True	0	0
W_Cleveland	C_Alternate	*all*	P1	True	1000	500

Running the model, we can see its outputs.

Output table: Arcs

Source	Destination	Product	...	Flow
W_Boston	C_Alternate	*all*	...	4000
W_Cleveland	C_Alternate	*all*	...	1000

How interesting... The output would be exactly the same as the last scenario. This happens simply because now the **Binary Min** value is ignored, since the minimum in fact is 1000 units. We can get a conclusion based on this exercise, **you should not set up data in both fields Min Flow and Binary Min, the output would be the same as just simply filling the Min Flow field. Use one or the another one, but not both together. This would push more data to the model and increase the time to process it, with no direct benefits.**

### Tip

Try by yourself to model the scenario where **Min Flow = Binary Min**. By now you should already have some idea of what would happen.

### Maximum Flow

This part is totally straight forward. As the Minimum Flow, you can limit the Maximum Flow of units that a source, destination, product and time period combination will flow, by using the **Max Flow** field.

### Maximum Number of Inbound/Outbound Sites

Maybe you will be required to model a scenario where you need to limit the amount of sources or destinations for a specific Zone (a group of **Sites**), you can use the **Max Number Inbound Sites** or **Max Number Outbound Sites** fields.

Let's use a similar example as the one explored above. There is this customer at Albany that can be supplied by warehouses at Boston, Cleveland and a new one, at Montreal, but you want to maintain the system where the customer demand is satisfied by a maximum of 2 warehouses. The only difference would be that now the model will consider a third option, at Montreal, as a candidate to fulfill the demand.

In the [Sites](#) table, you can create a zone called Warehouse to group up all the existing warehouses:

Name	...	<b>Zone 1</b>
W_Boston	...	Warehouse
W_Cleveland	...	Warehouse
W_Montreal	...	Warehouse

Now you can use the [Min Max Flow](#) table set up a record to create this constraint, as below, with the [Max Number Inbound Sites](#) field. This configuration would force the model to satisfy the customer demand at Albany by using a maximum of 2 warehouses as sources.

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	...	Max Number of Inbound Sites
Warehouse	C_Alternate	*all*	P1	...	2

You can do the opposite as well, by using the [Max Number Outbound Sites](#) field. If you have a zone for all customers named Customer, and you want to limit the number of Outbound Sites, you can apply the configuration below.

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	...	Max Number of Inbound Sites	Max Number of Outbound Sites
Warehouse	Customer	*all*	P1	...	2	3

This structure would limit the number of inbound sites by 2 and all warehouses would satisfy a maximum of 3 customers.

#### Constraining Flows Using Percentages

There are two fields in the [Min Max Flow](#) table that can be used to constrain flow units using percentages, [Min Percentage Supplier](#) and [Max Percentage Supplier](#).

Let's use the same example of the [Minimum and Binary Minimum](#) section. You want to force the model to satisfy 20% of the demand by the warehouse at Cleveland. In that case, 20% of the total demand of 5000 units is 1000 units. Suppose that you are planning a new month of distribution and the sales team sent you a forecast of 10,000 units of demand, the new value to be fixed would be of 2000 units. You probably do not want to update the model doing the same account every month to calculate how many units 20% represents on top of the total demand, so you should definitely use the [Min Percentage Supplier](#) field, in this case. You could just set the value of 20 at this field, and the model would force at least 20% of the total flow to be satisfied by the source specified.

#### Input table: Min Max Flow

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	...	Min Percentage Supplier	Max Percentage Supplier
W_Cleveland	C_Alternate	*all*	P1	...	20	100

#### Output table: Arcs

Source	Destination	Product	...	Flow
W_Boston	C_Alternate	*all*	...	8000
W_Cleveland	C_Alternate	*all*	...	2000

Regarding the field [Max Percentage Supplier](#), the logic is the same. If you want to force the model to satisfy a maximum of 80% of the demand by Boston warehouse, you can set up the [Min Max Flow](#) table as below:

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	...	Min Percentage Supplier	Max Percentage Supplier
W_Boston	C_Alternate	*all*	P1	...	0	80

Output table: Arcs

Source	Destination	Product	...	Flow
W_Boston	C_Alternate	*all*	...	8000
W_Cleveland	C_Alternate	*all*	...	2000

### Note

If you set values in both field **Min Flow** and **Min Percentage Supplier**, it will account the *BIGGEST* one. For example, if you set a **Min Percentage Supplier** value of 20% from Cleveland for a demand of 10,000 units at Albany, and fill a value of 1500 units in the **Min Flow** field, 2000 units would flow by Cleveland ( $20\% * 10000 = 2000 > 1500$ ). For the Maximum fields would be the opposite, the *SMALLEST* would prevail. The examples below can clarify this logic.

Min Flow > Min Percentage Supplier:

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	Active	Min Flow	...
W_Cleveland	C_Alternate	*all*	P1	True	<b>3000 (prevails)</b>	...

Min Flow < Min Percentage Supplier:

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	Active	Min Flow	...
W_Cleveland	C_Alternate	*all*	P1	True	1500	...

Max Flow > Max Percentage Supplier:

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	...	Max Flow	...
W_Boston	C_Alternate	*all*	P1	...	9000	...

Max Flow < Max Percentage Supplier:

Source Zone Name	Destination Zone Name	Product Group	Time Period Group	...	Max Flow	...
W_Boston	C_Alternate	*all*	P1	...	<b>7000 (prevails)</b>	...

## Zones and Groupings in Foresta

### Introduction

When modeling a supply chain network, it is possible to group up elements that have something in common. This could be a good alternative to simplify the modeling, to decrease processing time, and to reduce the amount of data to be analyzed and treated by the user.

Let's consider a network with Plants, Warehouses and Customers. Products are produced at the Plants, stored at Warehouses, and distributed to Customers by Warehouses only. So you want to include this logic in the model by using the table [Lanes](#), where you can set which sources can supply each destination. You have many plants, warehouses and customers, so maybe you do not want to create manually lane by lane, because the amount of combinations would be huge. If the network has 5 Plants, 10 Warehouses and 50 Customers, for example, the amount of combinations would be around 550 (5 Plants x 10 Warehouses + 10 Warehouses x 50 Customers). In case you want to consider that Plants can supply Customers as well, this number would increase to 800 (550 + 5 Plants x 50 Customers), so would be necessary to model the [Lanes](#) table with something around 800 rows of data.

To avoid all that work, you can just use Grouping modeling features. The [Sites](#) table have columns dedicated to that, called Zones, that are groups of sites. In the example above, you could just use a zone field to differentiate Plants, Warehouses and Customers, like shown below:

Input table: [Sites](#)

Name	Active	...	Zone 1
P_Dover	True	...	Plant
W_Boston	True	...	Warehouse
W_Cleveland	True	...	Warehouse
C_Albany	True	...	Customer
C_Buffalo	True	...	Customer

Therefore, you can simplify the process of modeling lanes, using each zone created, as below:

Input table: [Lanes](#)

Name	Precedence	Source Zone Name	Destination Zone Name	Product Group	...
Plants-Warehouses	0	Plant	Warehouse	*all*	...
Warehouses-Customers	0	Warehouse	Customer	*all*	...

instead of...

Name	Precedence	Source Zone Name	Destination Zone Name	Product Group	...
Dover-Boston	0	P_Dover	W_Boston	*all*	...
Dover-Cleveland	0	P_Dover	W_Cleveland	*all*	...
Boston-Albany	0	W_Boston	C_Albany	*all*	...
Boston-Buffalo	0	W_Boston	C_Buffalo	*all*	...
Cleveland-Albany	0	W_Cleveland	C_Albany	*all*	...
Cleveland-Buffalo	0	W_Cleveland	C_Buffalo	*all*	...



By using grouping features, you can facilitate your work in cases new sites are opened in the network, for example. It would not be necessary to update data in the [Lanes](#) table every time a new site is included, you would just need to include in the [Sites](#) table and associate it with the proper zone.

#### Creating Groups

It is possible to create Groups at [Input Tables](#) that have Grouping X columns (or Zone X for [Sites](#) table). For example, at the [Production Lines](#) table, there are columns, starting from Grouping 1, ending at Grouping 10, that you can use to create different groups. Sometimes you will need to create many groups for the same element, as shown:

Input table: [Sites](#)

Name	Active	...	Zone 1	Zone 2	Zone 3	Zone 4
W_Port of Virginia	True	...	Warehouse	Virginia_US	Southeast_US	US
S_Port of Santos	True	...	Supplier	Sao Paulo_BR	Southeast_BR	BR
P_Dover	True	...	Plant	Delaware_US	Northeast_US	US
C_Albany	True	...	Customer	New York_US	Northeast_US	US



Be careful when setting up different zones that have the same name. The example above shows that it is important to differentiate the Southeast region of United States (US) from the Southeast region of Brazil (BR), since in tables like [Lanes](#) and [Zone Carriers](#) you can create only rows with an *SINGLE* origin zone pointing to a *SINGLE* destination zone. Without the acronyms at the end of the regions, the model would understand the two as if they were the same, as the data of the other zones would not be informed. Take a look at the example below, without the concatenations.

Input table: [Sites](#)

Name	Active	...	Zone 1	Zone 2	Zone 3	Zone 4
W_Port of Virginia	True	...	Warehouse	Virginia	Southeast	US
S_Port of Santos	True	...	Supplier	Sao Paulo	Southeast	BR
P_Dover	True	...	Plant	Delaware	Northeast	US
C_Albany	True	...	Customer	New York	Northeast	US

Input table: [Zone Carriers](#)

Carrier	Source Zone Name	Destination Zone Name	Product Group	Unit Type	Cost Per Unit
Standard_Carrier	Southeast	Northeast	*all*	Weight	0.3

The model would understand that it would cost \$ 0.30 per pound to ship products from Southeast of US to Northeast of US, and the same cost would be applied when shipping from Southeast of Brazil to Northeast of US, what it would not make sense.



Concatenation is a good practice when grouping on different levels of an hierarchy!

## Modeling High Service Demand in Foresta Supply Chain Network Optimizer

### Introduction

The **Service Level** is a critical indicator of quality and efficiency in supply chain management, and it can serve as both an optimization driver and a requirement. In most supply chains, there is a strong correlation between service levels and transportation distances. As the **distance** between the origin and destination increases, the lead time required for delivery also increases, which can result in a higher risk of delivery delays and disruptions.

Based on this logic, we developed some feature to enable the consideration of Service Levels in your model. These features can be used to create new service level scenarios, which can help identify potential opportunities for business growth and improvement:

- High Service Demand Distance:
- High Service Demand Threshold input parameter
- Percentage High Service Demand (PHSD):
- PHSD output parameter
- Minimum PHSD
- PHSD as an optimization driver

### High Service Demand Distance

Imagine a business that considers a distance of 150 miles as the maximum distance from the customer which would guarantee high service levels. Let's call that distance *High Service Demand Distance*. If you configure, at the [Parameters](#) input table, the parameter named **High Service Demand Threshold** with the value of 150, the model will understand that a demand volume satisfied from within 150 miles can be considered as a *High Service Demand*. In other words, it is a demand value that was satisfied from a source that was "close" to it, and "close" means the same as "within 150 miles", in this example.

#### Input table: Parameters

Key	Value
...	...
High Service Demand Threshold	150

### Percentage High Service Demand

In the solution outputs, you can measure how much of the demand was satisfied within this *High Service Demand Distance* (High Service Demand Threshold), analyzing the [Parameters](#) output table, in the row **Percentage High Service Demand** (notice that this is a different table from the input table [Parameters](#)).

#### Output table: Parameters

Key	Value
...	...
Percentage High Service Demand	66.6667

In the case above, 66.67% of the demand was satisfied from sources within 150 miles distance from the demand locations. But in case if you want to model that percentage as a constraint, you can set a minimum percentage value as a requirement, using the parameter **Min Percentage High Service Demand** (at [Parameters](#)).

If you set up the [Parameters](#) table as below, the solution would optimize the network considering that at least 80% of the demand would be satisfied from sources within 150 miles distance from the demand locations.

#### Input table: Parameters

Key	Value
...	...
High Service Demand Threshold	150
Min Percentage High Service Demand	80

#### Percentage High Service Demand as an Optimization Driver

You can set the **Percentage High Service Demand** as an optimization driver. By doing that the solution will try to solve the network minimizing distances from sources to demand destinations, as a way to maximize this percentage.

Take a look at the [Optimization Objectives](#) page. At the section [Percentage High Service Demand objective](#), you can find a thorough explanation about how to configure the **Objective** parameter using this percentage.

## 4.3 Transportation Optimization

---

### 4.3.1 Transportation Optimization

#### WHAT IS TRANSPORTATION OPTIMIZATION?

Transportation Optimization (TO) is a comprehensive system that optimizes shipping operations. It manages input data such as orders, site information, and rates, then generates efficient delivery routes. TO can handle various transport scenarios, from single-pick single-drop routes to complex multi-pick multi-drop logistics, either for private fleets or third-party logistics providers. TO produces detailed reports, uncovering infeasible orders, and offers flexibility in configuration. It's a powerful tool offering essential input tables, output reports, and a mathematical optimization model to achieve optimal solutions for transportation routing.

👉 Use the navigation to explore further details

## 4.3.2 Input Tables

### Main Data

#### ORDERS

This table records necessary information of the orders that need to be shipped.

##### Order ID

*Type: Text*

Unique ID for the order that needs to be shipped.

##### Customer ID

*Type: Text*

Customer ID associated with the order.

##### Ship From ID

*Type: Text*

Origin location for the shipment.

##### Ship To ID

*Type: Text*

Destination location for the shipment

##### Weight

*Type: Numeric*

Weight of the order, if exists. This is a capacity metric.

##### Volume

*Type: Numeric*

Volume of the order, if exists. This is a capacity metric.

##### Cube Adjusted Weight

*Type: Numeric*

Cube adjusted weight of the order, if exists. This is a capacity metric.

##### Pallets

*Type: Numeric*

Size of the order in pallets, if exists. This is a capacity metric.



At least one of the four capacity metric for an order must be provided.

##### Earliest Pickup

*Type: DateTime*

The earliest date and time the shipment can be picked up at the Ship From location.

##### Latest Pickup

*Type: DateTime*

The latest date and time the shipment can be picked up at the Ship From location.

**Earliest Dropoff***Type: DateTime*

The earliest date and time the shipment can be dropped off at the Ship To location.

**Latest Dropoff***Type: DateTime*

The latest date and time the shipment can be dropped off at the Ship To location.

**Freight Class***Type: Text*

Use this column to define freight class for the shipment.

**Equipment ID***Type: Text*

Equipment that can ship this order. Acceptable values are *Equipment ID* defined in the Equipment Configurations table or `*all*`. `*all*` is suitable if the user wants the solver to decide the equipment to be used.

**First Pick on Route***Type: Text*

Whether this order must be the first order on the route to be picked. Can take values *yes* or *no*.

**Last Pick on Route***Type: Text*

Whether this order must be the last order on the route to be picked. Can take values *yes* or *no*.

**First Drop on Route***Type: Text*

Whether this order must be the first order on the route to be dropped off. Can take values *yes* or *no*.

**Last Drop on Route***Type: Text*

Whether this order should be the last order on the route to be dropped off. Can take values *yes* or *no*.

**Grouping 1***Type: Text*

Use this column to define an order grouping. Don't define singleton or all.

**Grouping 2***Type: Text*

Use this column to define an order grouping. Don't define singleton or all.

**Grouping 3***Type: Text*

Use this column to define an order grouping. Don't define singleton or all.

## Sample Input

<b>Order ID</b>	<b>Customer ID</b>	<b>Ship From ID</b>	<b>Ship To ID</b>	<b>Weight</b>	<b>Volume</b>	<b>Cube A Weight</b>
1	C_001	WH_A	C_001	2000		
2	C_002	WH_A	C_002			10000
3	C_003	WH_C	C_003		300	

**SITES**

Populate this table to create sites (i.e. plants, warehouses, customers, DCs, suppliers, etc.). All models need at least one site. Site names should be unique in the model.

**Site ID**

*Type: Text*

Identifies the site. Will also create a singleton zone.

**Depot ID**

*Type: Text*

Depot ID that the site is assigned to. All orders originating from this site must get an Equipment from the assigned Depot.

**City**

*Type: Text*

City of the site.

**State**

*Type: Text*

State of the site.

**Postal Code**

*Type: Text*

Postal code of the site.

**Country**

*Type: Text*

Country of the site.

**Latitude**

*Type: Numeric*

Used for Haversine distance calculations and mapping.

**Longitude**

*Type: Numeric*

Used for Haversine distance calculations and mapping.

**Fixed Load Time (Minutes)**

*Type: Numeric*

Fixed amount of time, in minutes, it takes to load a shipment at a site.

**Fixed Unload Time (Minutes)**

*Type: Numeric*

Fixed amount of time, in minutes, it takes to unload a shipment at a site.

**Variable load Time (Minutes)**

*Type: Numeric*

Variable time, in minutes, it takes to load a unit of shipment.

**Variable Unload Time (Minutes)**

*Type: Numeric*

Variable time, in minutes, it takes to unload a unit of shipment.

## Variable UOM

*Type: Text*

Unit of measure for variable load / unload times. Can be one of [Weight, Volume, Cube Adjusted Weight, Pallets]

## Max Early Arrival Wait Time (Minutes)

*Type: Numeric*

Amount of time a vehicle is allowed to wait, if it arrives early.

## Processing Time (Minutes)

*Type: Numeric*

Processing time in minutes. These are for facilities that serve as a Cross-Dock.

## Dwell Time (Hours)

*Type: Numeric*

Max amount of time, in hours, that an order may spend at a facility. These are for facilities that serve as a Cross-Dock.

## First Pick on Route

*Type: Text*Whether this order must be the first order on the route to be picked. Can take values *yes* or *no*.

## Last Pick on Route

*Type: Text*Whether this order must be the last order on the route to be picked. Can take values *yes* or *no*.

## First Drop on Route

*Type: Text*Whether this order must be the first order on the route to be dropped off. Can take values *yes* or *no*.

## Last Drop on Route

*Type: Text*Whether this order should be the last order on the route to be dropped off. Can take values *yes* or *no*.

## Zone 1 to Zone 10

*Type: Text*

Use these columns to define groupings / sets of sites. Descriptive Column to provide additional information about the Site.

## Sample Input

Site ID	Depot ID	City	State	Postal Code	Country	Latitude
P_Dover	WH_Dover	Dover	IA	19901	USA	39.14
W_El Paso	WH_Waco	El Paso	TX	79901	USA	31.76
C_Albany		Dover	NY	12288	USA	42.65

**EQUIPMENT CONFIGURATIONS**

Contains necessary information of the set of equipment that can be used to ship the order.

**Equipment ID**

*Type: Text*

Identifies the equipment. It should be unique.

**Active**

*Type: Boolean*

Used for Including or Excluding the site from Optimization.

True

False



Beware. Any string other than 'True' or 'true' will deactivate this record. The model does not accept binary values of 0 and 1 as False and True.

**Rank**

*Type: Numeric*

Used to define the priority of considering the given equipment, especially for private fleet cases

Lower the number, higher the rank. The magnitude is irrelevant.

**Equipment Type**

*Type: Text*

Equipment type that can be used to ship an order.

**Team Driver**

*Type: Text*

Whether this equipment requires team driver. The values it can take are **yes** or **no**.

**Reefer**

*Type: Text*

Whether this equipment is a refrigerated vehicle. The values it can take are **yes** or **no**.

**Hazmat**

*Type: Text*

Whether this equipment can handle hazardous material. The values it can take are **yes** or **no**.

**Max Weight**

*Type: Numeric*

Maximum capacity of the equipment in Weight.

**Max Volume**

*Type: Numeric*

Maximum capacity of the equipment in Volume.

**Max Cube Adjusted Weight**

*Type: Numeric*

Maximum capacity of the equipment in Cube Adjusted Weight.

**Max Pallets**

*Type: Numeric*

Maximum capacity of the equipment in Pallets.

## Speed

*Type: Numeric*

Speed of the equipment in miles / km per hour.

## Max Number of Picks

*Type: Numeric*

Maximum number of pickup stops the equipment can have.

## Max Number of Drops

*Type: Numeric*

Maximum number of delivery stops the equipment can have.

## Max Number of Stops

*Type: Numeric*

Maximum number of total stops the equipment can have. It should always be equal to **Max Number of Picks + Max Number of Drops**.

## Max Distance

*Type: Numeric*

Max total distance the equipment can travel for each route. It can be null, if it does not exist.

## Max Out of Route

*Type: Numeric*

Max out of route distance the equipment can travel for each route. It can be null, if it does not exist.

## Max Out of Route Percentage

*Type: Numeric*

Max out of route distance in percentage, the equipment can travel for each route. It can be null, if it does not exist.

## Max Distance between Picks

*Type: Numeric*

Max allowable distance between two pickup sites. It can be null, if it does not exist.

## Max Distance between Drops

*Type: Numeric*

Max allowable distance between two drop-off sites. It can be null, if it does not exist.

## Max First Pick to Last Pick

*Type: Numeric*

Max allowable distance between first pickup to last pickup sites. It can be null, if it does not exist.

## Max First Pick to First Drop

*Type: Numeric*

Max allowable distance between first pickup to first delivery on a route. It can be null, if it does not exist.

## Max First Pick to Lasp Drop

*Type: Numeric*

Max allowable distance between first drop to last drop sites. It can be null, if it does not exist.

## Stop Charges

*Type: Numeric*

The cost incurred of making a stop on the route. This is a charge that should be incurred on every extra stop a vehicle makes

Fuel Surcharge Per Unit Distance

*Type: Numeric*

Used in calculating the shipment cost.

**HOURS OF SERVICE RULES**

This table contains rules for single and team drivers.

**Team Driver**

*Type: Text*

Whether this rule is for team driver. Can take values *yes* or *no*.

**Driving hours**

*Type: Numeric*

Allowable driving time for a driver or team driver.

**Working hours**

*Type: Numeric*

Allowable operational hours for a driver or team driver

**Off Duty Hours**

*Type: Numeric*

Mandatory hours of off duty for a driver or team driver.

**Sample Input**

<b>Team Driver</b>	<b>Driving Hours</b>	<b>Working Hours</b>	<b>Off Duty Hours</b>
yes	11	14	10
no	11	14	10

**SITE OPERATING HOURS**

This table contains working days and operating hours of each site or zone.

**Site Zone**

*Type: Text*

Identifies the Site / Zone. This should match existing Site / Zone in the Sites table.

**Working Days**

*Type: Text*

Day of the week for which the time window is defined. Semi-colon (;) can be used to separate multiple days.

**Open Time**

*Type: Time*

Opening time for the Site / Zone.



The values should be in 00:00 (24-hr) format.

**Close Time**

*Type: Time*

Closing time for the Site / Zone.



The values should be in 00:00 (24-hr) format.

**Sample Input**

Site Zone	Working Days	Open Time	Close Time
Customer	mon;tue;wed;thu;fri	08:00	17:00
WH	mon;tue;wed;thu;fri	00:01	23:59

## PARAMETERS

In this table the user can configurate how the optimization model will behave, setting up different parameters that will influence the optimization outputs.

### Key

*Type: Text*

Name of the parameter.

### Value

*Type: Text*

Value of the parameter. Edit based on the description.

### Category

*Type: Text*

A field used to categorize and organize parameters in sections.

### Description

*Type: Text*

Description of the parameter. Provides information about the values parameters can take.

### Parameters explanation

Parameters are organized in their respective categories

### Transit

- **Distance Unit:** Miles or Kilometers
- **Distance Factor:** Road curvature factor, sometimes used to adjust Haversine distances

### Optimization

- **Use Hierarchical Optimization:** Whether to solve for multiple objectives sequentially
- **Cost Objective Percentage Weight:** Percentage of weight given to minimizing total cost. Provide an integer between 0 to 100
- **Distance Objective Percentage Weight:** Percentage of weight given to minimizing total distance. Provide an integer between 0 to 100
- **Number of Routes Objective Percentage Weight:** Percentage of weight given to minimizing total number of routes used. Provide an integer between 0 to 100
- **Solver Optimality Gap Percentage:** This is for the main mathematical optimization model, and it is a non-negative number that's defined as a percentage and will be passed to the solver. Acceptable values are in the range of (0.01 to 100), excluding 100. Default value is 0.5.
- **Solver Run Time Limit (Seconds):** Max time that solver can have. Leaving this blank tells the solver to take as much time as needed to solve the model. Be advised that setting a small value for time limit may result in sub-optimal solution. If the value is very small, the model may struggle to even find a feasible solution.

### Further Explanation

Currently, there are 3 objectives that can be minimized: total cost, total distance, and total number of routes. There are 3 ways to use them.

1. Minimize one objective (default): Allocate all the weight (i.e., 100) to one objective. For example, if `Cost Objective Percentage Weight` is 100, it signals an interest in minimizing total cost.
2. Minimize multiple objectives using "weighted sum" or "blended" method: One can give different weights to different objectives to signal that a weighted-sum of them should be minimized. For example, a weight of 80 is assigned to `Distance Objective Percentage Weight` and 20 to `Number of Routes Objective Percentage Weight`. The formula is simple:  $\sum_i w_i * Obj_i$ . Where  $w_i$  is the weight provided for objective  $i$  and  $Obj_i$  is the total value for the objective  $i$ . Note that  $Obj_i$  is normalized for cost and distance objectives. That is, each cost (distance) is divided by the max route's cost (distance) value, so they all become unitless and fall in the (0, 1) range. That way, they can be added together. Since number of routes is already a unitless value, normalization is not necessary. Using this method, multiple objectives boil down to one objective to minimize.
3. Minimize multiple objectives in a hierarchy (Hierarchical Optimization): The `Use Hierarchical Optimization` must be selected (or its value should be set to "Yes") but the rest of the setup is like #2. In this method, rather than adding the objectives, the weights are interpreted as priorities (higher weight, higher priority) and the desired objectives are solved one after the other. For example, provide a weight of 80 to `Distance Objective Percentage Weight` and 20 to `Number of Routes Objective Percentage Weight`. The mathematical optimization model, first minimizes total distance. Then take its optimal value and add it as a new constraint to the model, and then solve for minimizing number of routes (The solver's multi-objective optimization capability is used and the constraints are not added explicitly).

Note that the magnitude of weights in **hierarchical optimization** don't matter. So, providing weights of 80-20 to the two objectives is the same as 60-40 or 51-49. Because all that matters is that there is a clear ordering between them, so they can be sorted and the higher value can be given a higher priority.

Regardless of the method, ensure the sum of weights adds up to 100.

#### Route Generation

- **Ship Alone:** If enabled, create routes where each order is shipped alone
- **Enable Consolidation:** If enabled, consolidate orders with the same origin, destination, and compatible equipment and time windows into the same route.
- **Enable Neighborhood Search:** Default approach to generate multi-stop routes
- **Enable Rapid Route Generation:** Default approach to rapidly generate multi-stop routes. Use for quick analysis or for challenging cases, especially when using Neighborhood Search methods can take a long time.
- **Enable Expanded Search:** If enabled, "Neighborhood Search" is run with a larger neighborhood size
- **Enable Diversification Search:** If enabled, spend extra time to generate more diverse multi-stops routes
- **Diversification Runtime (Seconds):** Amount of extra time used for route generation if "Diversification Search" is enabled. Suggested range [60, 600]

More information about some nuances in route generation methods can be found [here](#).

#### Decomposition

- **Order Grouping Interval:** The interval by which the orders should be put into groups. (e.g., grouped by week, by day, or no grouping). Acceptable values are one of [Week, Day, No Group]
- **Grouping Time Field:** Which time field should be used to group the orders. Acceptable values are one of [Earliest Pickup, Latest Pickup, Earliest Dropoff, Latest Dropoff]. This field is used only if "Order Grouping Interval" is Week or Day.
- **Model Partitioning Criteria:** Which field should be used to decompose the model. Acceptable values can be Grouping 1, Grouping 2, Grouping 3, given they have values. If those fields are empty or this functionality is not desired, leave it empty.

#### Control

- **Max Backhaul Distance:** Max distance allowed for driving empty back to the origin location
- **Activate Backhaul Restriction:** Whether to activate or deactivate backhaul distance restriction or explore both options
- **Include Backhaul Leg In Cost:** Whether to include backhaul leg distance in cost calculation of closed loop routes
- **Enable Excess Equipment:** Whether the use of extra equipment is allowed
- **Enable Mode Selection:** Whether selecting routes with the best transportation mode is preferred over respecting configurations rank

## SMC3 Rater

- **SMC Username:** Username to load SMC3
- **SMC License:** License Key to load SMC3
- **SMC Password:** Password. If you append '\_ENCRYPT\_ME' to this value, and then upload this table using Google Sheets, then an encrypted version of the password will be stored. It is also encrypted if '\_ENCRYPT\_ME' is at the end of the value uploaded from CSV or Excel
- **Maximum Weight Break:** One of L5C, M5C, M1M, M5M, M10M, M20M, M30M, M40M to determine the maximum Weight Break to return for Transportation Optimizer app
- **Tariff Name:** Name of the Tariff (8 characters) the SMC rater should use to rate the shipment
- **Tariff Date:** The date the SMC rater should use to rate the shipment. Only the year is needed, but any string that looks like a date will be understood
- **Percent Discount:** Percent by which the LTL rating cost should be reduced. Number between 0 (inclusive) and 100 (exclusive)
- **Minimum Charge Floor:** User defined value that is an absolute minimum charge that is not subject to discounting. Values should be non-negative and finite.

## Sample Input

Key	Value	Category	Description
Distance Unit	Miles	Transit	Miles or Kilometers
Grouping Time Field	Latest Delivery	Decomposition	Which time field should be used to group the orders. Acceptable values are one of [Earliest Pickup, Latest Pickup, Earliest Dropoff, Latest Dropoff]. This field is used only if "Order Grouping Interval" is week or day.
Model Partitioning Criteria	Grouping 2	Decomposition	Which field should be used to decompose the model. Acceptable values can be Grouping 1, Grouping 2, Grouping 3, given they have values.

**DEPOTS**

Populate this table if routes should start from a centralized location. Usually depots are used when vehicles are located in a central location and the customer manages its own fleet.

## Depot ID

*Type: Text*

Identifies the depot. Depot should be defined in the sites table.

## Equipment ID

*Type: Text*

Identifies an Equipment. Equipment can be defined in Equipment Configurations.

## Closed Loop

*Type: Text*

Indicates whether the Equipment should return to the Depot after last stop. Can be *Yes* or *No*.

## Equipment Count

*Type: Numeric*

Count of Equipment available at the Depot.

## Sample Input

Depot ID	Equipment ID	Closed Loop	Equipment Count
Chicago	Dry-Van	Yes	100
Atlanta	Reefer	No	29

## Shipping Rates

### RATES

Contains rates for going between each pair of locations for any given equipment. If minimizing transportation costs is not of interest, this table can be left empty.

#### ID

*Type: Text*

Unique identifier for the defined cost.

#### Equipment ID

*Type: Text*

Identifies an Equipment. Equipment can be defined in the Equipment Configurations table.

#### Origin Zone

*Type: Text*

Identifies a Site / Zone.

#### Destination Zone

*Type: Text*

Identifies a Site / Zone.

#### Rate

*Type: Numeric*

Rate per unit distance for the Equipment ID - Origin Zone - Destination Zone combination.

#### Minium Charge

*Type: Numeric*

Absolute minimum charge for shipping from Origin to Destination.

#### Sample Input

ID	Equipment ID	Origin Zone	Destination Zone	Rate	Minimum Charge
1	Dry-Van	Chicago	New York	1.9	0
2	Reefer	Atlanta	TX	2.6	300
3	Flatbed	Florida	NC	1.7	250

**LTL RATES**

Contains LTL rates. Like Rates table, if cost minimization is not one of the objectives, it can be left empty.

**ID**

*Type: Text*

Unique identifier for the defined cost.

**Origin Postal Code**

*Type: Text*

Postal code for the origin location.

**Destination Postal Code**

*Type: Text*

Postal code for the destination location.

**Weight Break**

*Type: Text*

LTL weight break as defined by the SMC rater. Can be L5C, M5C, M1M, M5M, M10M, M20M, M30M, M40M.

**Tariff Name**

*Type: Text*

Name of the tariff SMC rater should use to rate the shipment.

**Tariff Date**

*Type: Text*

The date SMC rater should use to rate the shipment. Only the year is needed.

**Freight Class**

*Type: Text*

Shipment freight class of the LTL load.

**Percent Discount**

*Type: Numeric*

Percent by which the rate should be reduced. Can be numeric value between 0 and 100.

**Minimum Charge Floor**

*Type: Numeric*

User defined absolute minimum charge for shipment that should not be discounted.

**Rate**

*Type: Numeric*

Rate of shipping per 100 lbs.

**Minimum Charge**

*Type: Numeric*

Minimum charge of shipping from Origin to Destination.

## Sample Input

<b>ID</b>	<b>Origin Postal Code</b>	<b>Destination Postal Code</b>	<b>Weight Break</b>	<b>Tariff Name</b>	<b>Tariff Date</b>	<b>Freight</b>
1	27107	37814	M1M	LITECZ02	20220926	200
2	27107	29485	M5M	LITECZ02	20220926	200
3	27107	23175	M1M	LITECZ02	20220926	200

## Constraints

### ENTITY RELATIONS

Populate this table to define `*force*` or `*prevent*` relationships between different entities. Entities are Zone, Site, Customer, Order, Grouping, and Equipment. This is a very powerful table. Check entity relation table page to learn more.

#### Entity 1

*Type: Text*

The first entity for which the relationship needs to be defined.

#### Entity Type 1

*Type: Text*

The entity type of Entity 1 which can be: Zone, Site, Customer, Order, Grouping.

#### Entity 2

*Type: Text*

The second entity for which the relationship needs to be defined.

#### Entity Type 2

*Type: Text*

The entity type of Entity 2 which can be: Zone, Site, Customer, Order, Grouping, Equipment.

#### Relationship

*Type: Text*

Can be either `*prevent*` or `*force*`

#### Sample Input

Entity 1	Entity Type 1	Entity 2	Entity Type 2	Relationship
Customer 1	Site	Customer 2	Site	<code>*prevent*</code>
Order 1	Order	Dry-van	Equipment	<code>*prevent*</code>

More information for Entity Relationships can be found in the [Entity Relationships](#) page in FAQ section.

#### PREDEFINED ROUTES

If there are some predefined routes, where the sequence of stops for some orders is already determined, populate this table.

Two great use cases for this table are as follows:

**1. Baseline Modeling:** You can use this table to run a baseline scenario. For example, if all the current routes for covering the orders are available, you can populate this table and solve a model. If any route is infeasible, its orders are reported in the unrouted orders table. This way, we can see if all the constraints imposed by the user actually holds in their current system or not.

**2. Solution Fine-tuning:** You may run a model and are satisfied with X% of the routes in the solution (let's assume X=80 here). You can use this table, add the desired routes as Predefined Routes and run the model again. The model essentially optimizes for the remaining orders. For hard models, this can create a great opportunity to reduce the size of the model and explore even more powerful route generation algorithms since the size of the remaining orders to be routed are much smaller compared to the original model. For example, if currently you use Enable Neighborhood Search algorithm, you can still do that but with a larger Neighborhood Size.

##### Route ID

*Type: Text*

Unique ID to identify a Predefined Route.

##### Order ID

*Type: Text*

Unique ID of the Order

##### Route Type

*Type: Text*

Describes the type of route, which depends on the problem type.

Can be: "Multi Stop", "Inbound", "Outbound", "Linehaul"

##### Ship From ID

*Type: Text*

Ship From ID associated with the order.

##### Ship From Stop Number

*Type: Numeric*

Position of pick-up stop in the route.

##### Ship To ID

*Type: Text*

Ship To ID associated with the order

##### Ship To Stop Number

*Type: Numeric*

Position of the drop-off stop in the route.

##### Equipment ID

*Type: Text*

Equipment used to ship the order.

## Sample Input

<b>Route ID</b>	<b>Order ID</b>	<b>Route Type</b>	<b>Ship From ID</b>	<b>Ship From Stop Number</b>	<b>Ship To ID</b>	<b>Ship To Number</b>
R_001	Order_1	Multi Stop	W_El Paso	1	C_001	3
R_001	Order_2	Multi Stop	W_El Paso	1	C_002	4
R_001	Order_3	Multi Stop	W_El Paso	1	C_003	2

**TRANSIT OVERRIDE**

Populate this table for overriding distance or travel time.

**Origin Site ID**

*Type: Text*

Origin site for which the rates are available.

Must match a valid site, as defined in the Sites table.

**Destination Site ID**

*Type: Text*

Destination site for which the rates are available.

Must match a valid site, as defined in the Sites table.

**Distance**

*Type: Numeric*

Distance to use between origin and destination site.

Cannot be null if travel time is defined.

**Travel Time (Hours)**

*Type: Numeric*

Travel time to use between origin and destination site.

Can be null.

**Is Symmetric**

*Type: Text*

Whether distance and travel time between destination and origin are the same as origin to destination.

**Sample Input**

Origin Site ID	Destination Site ID	Distance	Travel Time (Hours)	Is Symmetric
W1	W2	1000	20	Yes
W1	W3	750	15	No
C1	W2	800		Yes

### 4.3.3 Solution

---

#### **Summary Reports**

##### **SUMMARY**

A report of the main KPIs

##### **Key**

*Type: Text*

Name of the KPI.

##### **Value**

*Type: Text*

Value of the KPI.

## Detailed Reports

### ROUTES

Contains information about generated routes

#### Route ID

*Type: Text*

Unique ID for the route.

#### Route Type

*Type: Text*

The algorithm that generated this route.

#### Equipment ID

*Type: Text*

Type of Equipment used for this route.

#### Origin

*Type: Text*

Origin location for the route.

#### Origin Location Type

*Type: Text*

Type of origin location. Possible options: **Ship From, Ship To, Cross dock or Depot**.

#### Destination

*Type: Text*

Destination location of the route.

#### Destination Location Type

*Type: Text*

Type of destination location. Possible options: **Ship From, Ship To, Cross dock or Depot**.

#### Cost

*Type: Numeric*

Total route cost.

#### Distance

*Type: Numeric*

Total route distance.

#### Loaded Distance

*Type: Numeric*

Route's total distance with load.

#### Unloaded Distance

*Type: Numeric*

Route's total distance without load.

#### Out of Route Distance

*Type: Numeric*

Route's total out of route distance.

**Out of Route Percentage***Type: Numeric*

Route's total out of route distance percentage.

**Earliest Start Time***Type: DateTime*

The earliest date and time the route can start from the first pickup stop.

**Latest Start Time***Type: DateTime*

The latest date and time the route can start from the first pickup stop.

**Earliest End Time***Type: DateTime*

The earliest date and time the route can end at the last dropoff location.

**Latest End Time***Type: DateTime*

The latest date and time the route can end at the last dropoff location.

**Num Picks***Type: Numeric*

Number of pick stops on this route.

**Num Drops***Type: Numeric*

Number of drop stops on this route.

**Num Stops***Type: Numeric*

Total stops on this route.

**Num Orders***Type: Numeric*

Total number of orders delivered by this route.

**Weight***Type: Numeric*

Total Weight of the route.

**Volume***Type: Numeric*

Total Volume of the route.

**Cube Adjusted Weight***Type: Numeric*

Total Cube adjusted weight of the route.

**Pallets***Type: Numeric*

Total Number of Pallets in a route.

**Utilization**

*Type: Numeric*

Max capacity utilized of the truck over max truck capacity.

## Weighted Average Utilization

*Type: Numeric*

Percentage of weighted average distance of space used in a route.

**ORDER ROUTE DETAILS**

Includes information from the "Orders" table along with route(s) that handle the order

**Route ID**

*Type: Text*

ID of the route that handles the order.

**Order ID**

*Type: Text*

Unique ID for the order that needs to be shipped.

**Customer ID**

*Type: Text*

Customer ID associated with the order.

**Cross Dock ID**

*Type: Text*

Unique ID of a Cross-dock, if the order is covered by a route through Cross-dock.

**Ship From ID**

*Type: Text*

Origin location for the shipment.

**Ship From Postal Code**

*Type: Text*

Postal code of the location where the order is picked.

**Ship From City**

*Type: Text*

City of the location where the order is picked.

**Ship From Country**

*Type: Text*

Country of the location where the order is picked.

**Ship From Latitude**

*Type: Numeric*

Latitude of the location where the order is picked.

**Ship From Longitude**

*Type: Numeric*

Longitude of the location where the order is picked.

**Ship To ID**

*Type: Text*

Destination location for the shipment.

**Ship To Postal Code**

*Type: Text*

Postal code of the location where the order is dropped.

**Ship To City**

*Type: Text*

City of the location where the order is dropped.

## Ship To Country

*Type: Text*

Country of the location where the order is dropped.

## Ship To Latitude

*Type: Numeric*

Latitude of the location where the order is dropped.

## Ship To Longitude

*Type: Numeric*

Longitude of the location where the order is dropped.

## Ship To Stop Number

*Type: Numeric*

Position of Dropoff in the route.

## Weight

*Type: Numeric*

Weight of the order, if exists.

## Volume

*Type: Numeric*

Volume of the order, if exists.

## Cube Adjusted Weight

*Type: Numeric*

Cube adjusted weight of the order, if exists.

## Pallets

*Type: Numeric*

Size of the order in pallets, if exists.

## Earliest Pickup

*Type: DateTime*

Earliest date and time that the order is picked up in the route.

## Latest Pickup

*Type: DateTime*

Latest date and time that the order is picked up in the route.

## Earliest Dropoff

*Type: DateTime*

Earliest date and time that the order is dropped in the route.

## Latest Dropoff

*Type: DateTime*

Latest date and time that the order is dropped in the route.

## Equipment ID

*Type: Text*

Equipment used to ship the order.

**STOPS**

Contains information about each stop of a route.

**Route ID**

*Type: Numeric*

Unique ID for the route.

**Stop Number**

*Type: Numeric*

Position of the stop on the route.

**Stop Action**

*Type: Text*

What action is performed at the stop. Possible options: **Pick**, **Drop**, **Backhaul** or **Deadhead**.

**Location ID**

*Type: Text*

Unique location ID of the stop.

**Location Type**

*Type: Text*

Type of Location for the stop. Possible options: **Ship From**, **Ship To**, **Crossdock** or **Depot**.

**Location City**

*Type: Text*

City of the stop location.

**Location State**

*Type: Text*

State of stop location.

**Location Postal Code**

*Type: Text*

Postal code of the stop location.

**Location Country**

*Type: Text*

Country of the stop location.

**Location Latitude**

*Type: Numeric*

Latitude of the stop location.

**Location Longitude**

*Type: Numeric*

Longitude of the stop location.

**Equipment ID**

*Type: Text*

Equipment used to ship this order.

**Change in Capacity**

*Type: Numeric*

The amount that is picked or dropped at the stop. Only one capacity unit is reported in order of importance: Weight, Volume, Cube Adjusted Weight, Pallets.

## Distance To Next Stop

*Type: Numeric*

Distance to the next stop in the route.

## Earliest Arrival Time

*Type: DateTime*

The earliest date and time the equipment arrives at the stop location.

## Latest Arrival Time

*Type: DateTime*

The latest date and time the equipment arrives at the stop location.

**DEPOT ROUTE ASSIGNMENT**

When "Depots" table is provided along with the limit on each equipment at each location, generated routes should be assigned to different vehicles available at each depot. This table includes information about which routes are assigned to which equipment and which depot.

## Depot ID

*Type: Text*

Identifies the depot. Depot should be defined in the sites table.

## Equipment Number

*Type: Numeric*

Unique identifier for an equipment.

## Route ID

*Type: Text*

Unique ID for a route.

## Excess Equipment

*Type: Text*

Whether extra equipment is used on top of existing equipment. The values it can take are **true** or **false**.

## Route Sequence Number

*Type: Numeric*

Position of route visit on the Equipment.

## Equipment ID

*Type: Text*

Type of Equipment used to ship the order.

## Route Start Time

*Type: Numeric*

Date and Time the route starts on the equipment.

## Route End Time

*Type: Numeric*

Date and time the route ends on the equipment.

## Infeasibility Reports

### UNROUTED ORDERS

Includes information from the "Orders" table for those orders that could not be routed, along with the reason for this failure. If an order cannot be routed, it's because at least one constraint is being violated. For example, the order exceeds the total weight of a vehicle, or there is no rate information in the Rates table between origin and destination zone of this order, or any of the path-related Note that, even if an order violates several constraints, it's not possible to capture all or the most obvious ones either. Let's check a couple of examples:

- Order1 can go with any equipment (Equipment ID = `*all*`). There are two equipment available: TL and LTL. Assume it's first considered with TL. During the route generation, no rate is found between Order1's origin and destination zone. So, the reason for invalidity of the order is first saved as No TL Cost. Then, the order is considered with LTL. Now, assume the order is 15000 lb. All constraints are valid and the order can be rated and nothing will be reported then. But if the order is 25000 lb, it will hit Capacity Violation when running for LTL. So, in this case, if someone checks the unrouted reason of the order, the last attempt that was running with LTL, will be reported.
- Consider the case above but assume Order1 is first considered with LTL and then TL. In that case, the reason is reported as No TL Cost.
- Consider the case above but assume Order1 has a weight of 50000 lb where the Max Weight for TL is 40000 and for LTL is 20000. In that case, the reason is reported as Capacity Violation regardless of which equipment is the most recent one.
- Consider case above but assume that the order has one week to be picked up but it can only be picked up and dropped off when the locations are open between 8am to 5pm. The distance between Order1's origin and destination is 10 hours. There is no possible way to leave and arrive during this window. Even though there is still no cost, but due to order of constraint validation, time window of the order is checked first. So, this time, the reason will be an Invalid Transit Time Window.

### Order ID

*Type: Text*

Unique ID for the order that needs to be shipped.

### Customer ID

*Type: Text*

Customer ID associated with the order.

### Ship From ID

*Type: Text*

Origin location for the order.

### Ship From Postal Code

*Type: Text*

Postal code of the location where the order is picked.

### Ship From City

*Type: Text*

City of the location where the order is picked.

### Ship From Country

*Type: Text*

Country of the location where the order is picked.

### Ship From Latitude

*Type: Numeric*

Latitude of the location where the order is picked.

### Ship From Longitude

*Type: Numeric*

Longitude of the location where the order is picked.

**Ship To ID***Type: Text*

Destination location for the order.

**Ship To Postal Code***Type: Text*

Postal code of the location where the order is dropped.

**Ship To City***Type: Text*

City of the location where the order is dropped.

**Ship To Country***Type: Text*

Country of the location where the order is dropped.

**Ship To Latitude***Type: Numeric*

Latitude of the location where the order is dropped.

**Ship To Longitude***Type: Numeric*

Longitude of the location where the order is dropped.

**Weight***Type: Numeric*

Weight of the order, if exists.

**Volume***Type: Numeric*

Volume of the order, if exists.

**Cube Adjusted Weight***Type: Numeric*

Cube adjusted weight of the order, if exists.

**Pallets***Type: Numeric*

Size of the order in pallets, if exists.

**Earliest Pickup***Type: DateTime*

The earliest date and time the order can be picked up from the Ship From location.

**Latest Pickup***Type: DateTime*

The latest date and time the order can be picked up from the Ship From location.

**Earliest Dropoff***Type: DateTime*

The earliest date and time the order can be dropped off at the Ship To location.

**Latest Dropoff**

*Type: DateTime*

The latest date and time the order can be dropped off at the Ship To location.

## Equipment ID

*Type: Text*

Equipment that can ship this order.

## Reason

*Type: Text*

Reason due to which the order could not be routed.

## 4.3.4 FAQs

---

### Capacity And Utilization Metrics Calculations

There are four capacity metrics in the Routes table: Weight, Volume, Cube Adjusted Weight, and Pallets. There are also two utilization metrics: Utilization, and Weighted Utilization.

#### CAPACITY METRICS

They all report the maximum amount the route has carried at any given point. For example:

- Route 1 picks up 20,000 lb at location A and 15,000 lb at location B, and drops them at location C. Maximum weight = 35,000 lb.
- Route 2 picks up 20,000 lb at location A and 15,000 lb at location B, drops 10,000 lb at location C, picks 15,000 lb at location C, drops 25,000 at location D, and 15,000 at location E. Maximum weight = 40,000 lb.

#### UTILIZATION

For each of the capacity metrics that exist, utilization is the max capacity of the route over truck capacity. The utilization of the route is the max among these values. In the examples above, assume that the truck's max weight is 44,000 lb and its max volume is 3000. Let's also assume that the maximum volume is 2200 for Route 1 and 2750 for Route 2. The routes' utilization are:

- Route 1:  $\max \left( \frac{35,000}{44,000}, \frac{2,200}{3,000} \right) = 79.5\% \right)$
- Route 2:  $\max \left( \frac{40,000}{44,000}, \frac{2,750}{3,000} \right) = 91.6\% \right)$

#### WEIGHTED UTILIZATION

For each of the capacity metrics that exist, weighted utilization is calculated as the percentage of weighted average distance of used space in a truck.

The used capacity of the truck on each leg is calculated and multiplied by the leg's distance. The total of this value is then divided by total distance of the path to give a weighted average distance value. This value is then divided by max capacity of the truck to give a percentage of used space. This calculates used space based on capacity metric. The weighted utilization of the route is the max among the weighted utilization values. In other words:

$$\left[ \max_j \left( \frac{\sum_i c_{ij} * d_{ij}}{C_j} \right) \quad j \in \text{weight, volume, pallets, cube adjusted weight} \right]$$

where  $c_{ij}$  is the capacity metric  $j$  of the truck on leg  $i$ ,  $d_{ij}$  is the leg  $i$  distance, and  $C_j$  is the max truck capacity of the given metric  $j$ .

## Route Cost Calculation

Depending on whether the equipment is LTL or not, the cost calculation is different.

### NON-LTL EQUIPMENT

```
\[ \text{cost} = \max\left(\text{total distance} \times (\text{rate} + \text{fuel surcharge per distance}), \text{minimum charge}\right) + (\text{num stops} - 2) \times \text{stop charges} \]
```

- The total distance is the sum of distances from the origin to the last stop (i.e., last drop). If the route is a closed route and the backhaul distance (distance from the last stop to the origin) needs to be considered in the cost, then the total distance includes that leg as well.
- The "num stops" counts the number of picks and drops. So, in a closed loop route that requires a return to the depot, since there is no pick or drop happening at the depot, it's not considered. One can see that in a single-pick single-drop route, "num stops" becomes 2 and there are no extra stop-charges added to the cost.

### LTL EQUIPMENT

- LTL rates are based on the 5-digit zipcode of route's origin and destination, its freight class (which is considered as the freight class of the first order on the route), and its weight.
- The rates are in dollars per hundred pounds (a.k.a. CWT or hundred weight), which means we need to divide the route's weight by 100 before multiplying it by rate.
- To calculate the cost, we must first determine which weight break to use based on the route's weight.
- We also need to consider the next weight break and its rate since the cost is based on the smaller of the two values.

### Weight Breaks

Weight Break	Meaning
L5C	Less than 500 lbs
M5C	More than 500 lbs
M1M	More than 1000 lbs
M2M	More than 2000 lbs
M5M	More than 5000 lbs
M10M	More than 10,000 lbs
M20M	More than 20,000 lbs
M30M	More than 30,000 lbs
M40M	More than 40,000 lbs

Knowing that, the LTL cost is calculated as follows:

```
wb_cost = min(rate * weight / 100, next weight bracket rate * the lowerbound of next bracket weight / 100)
discounted cost = max(minimum charge, wb_cost) * (1 - percent discount)
cost = max(minimum charge floor, discounted cost)
```

Example:

- route's weight = 4500 lb
- weight break = M2M
- rate = \$190
- minimum charge = \$280
- minimum charge floor = \$100
- discount = 80%
- next weight break = M5M (so, its lowerbound is 5000)
- next weight break rate = \$150

```
wb_cost = min(190*4500/100, 150 * 5000/100) = 7500
discounted cost = max(280, 7500) * (1-0.8) = 1500
cost = max(100, 1500) = 1500
```

## Factors Affecting Run Time

*In essence, when there are more ways to arrange and combine various orders to create routes, more time will be consumed validating these options.*

This doesn't necessarily mean that a smaller dataset runs faster than a larger dataset. For example, processing 5000 orders where each order can almost fill up a truck will be much faster than processing 100 parcel-size orders in a close neighborhood.

Here are some important factors that are worth noting. Keep in mind that nothing below by itself can predict with certainty the run time or memory consumption. It's always the combination of factors that make a model easy or hard to solve. The only general rule is this:

The more relaxed the constraints, the greater the number of options to explore, leading to longer run times.

### IMPORTANT FACTORS

- Number of orders  $\uparrow \Rightarrow$  Run time  $\uparrow$
- Order's capacity (weight, volume, pallets, cube adjusted weight)  $\downarrow \Rightarrow$  Run time  $\uparrow$
- Because decrease of order's capacity, opens up a lot more options for the order to be combined with other orders.
- Order's pick time window (Latest Pickup - Earliest Pickup)  $\uparrow \Rightarrow$  Run time  $\uparrow$
- Order's Equipment ID options  $\uparrow \Rightarrow$  Run time  $\uparrow$
- For example, an order that can only use a dry van has fewer options for combinations than an order that can use \*all\*
- Number of configurations  $\uparrow \Rightarrow$  Run time  $\uparrow$
- Max Number of Picks/Drops/Stops  $\uparrow \Rightarrow$  Run time  $\uparrow$
- Values for path-related constraints  $\uparrow \Rightarrow$  Run time  $\uparrow$
- Path-related constraints are any of the constraints that only depend on the stops and their sequence and not the orders. They include:
  - All the constraints such as `Max Number of Picks`, `Max Distance`, or `Max Distance Between Picks` that are defined in Configurations table (exceptions are capacity constraints such as `Max Weight`).
  - Constraints such as "Max Backhaul Distance", "Disable Direction Change" (to be developed), "Enable Angle Check" (to be developed).
  - Site constraints: First/last pick on route, first/last drop on route.
- Note that in all of these constraints, a null value is equivalent of having a very large value, which means, the constraint is basically inactive and more options can be explored. This is also true for "yes/no" values. Anything that deactivates a constraint, increases the options, and hence, increases the run time.
- Values of `Working Days`  $\uparrow \Rightarrow$  Run time  $\uparrow$  (can also  $\downarrow$  it)
- Span of sites hours (`Close Time - Open Time`)  $\uparrow \Rightarrow$  Run time  $\uparrow$  (can also  $\downarrow$  it)
- If the number of working days and span between close time and open time increases, the window becomes wider which means there will be more feasible options for combining different orders. However, if the working days and open/close time together make the site operate on a 24/7 schedule (which is the loosest constraint you can have), basically the site is open all the time. So, there is no need to consider site's hours when validating time window, which in essence, converts the pick/drop window into one continuous window (from the earliest to the latest) rather than a series of discrete windows (i.e. Mon 8-17, Tue 8-17). So, the validation process will have fewer options to explore which will reduce the run time.
- `Grouping Time Field` parameter values in order of increasing run time: `day`, `week`, `no group`. In other words, the more relaxed the parameter, the longer the run time.
- If `Model Partitioning Criteria` parameter doesn't have any value, the run time will be more than when it has a value. It's because without any value, the data is not partitioned into separate groups and all are considered together.
- `Neighbor Size` parameter value  $\uparrow \Rightarrow$  Run time  $\uparrow$
- This is true for any other value that increase the size of the neighborhood search (such as the advanced parameter of `KNN Neighbor Size`, `KCORN Neighbor Size`, or `Num Iterations For Giant Tour`) or any value that increase the run time of search (such as `Diversification Runtime (Seconds)`).
- If we `Use Hierarchical Optimization` with multiple objectives, the optimization run time can increase. This is different from all the other factors above. Because all the above-mentioned factors increase the route generation process run time, but this one will affect the run time of the MIP model.

## Explanation and Use Cases of Entity Relations Table

### FIELDS

- Entity 1: Value of the first entity to establish the relationship
- Entity Type 1: It can be one of: **Zone, Site, Customer, Order, Grouping**
- Entity 2: Value of the second entity to establish the relationship
- Entity Type 2: It can be one of: **Zone, Site, Customer, Order, Grouping, Equipment**
- Relationship: Can be either \*force\* or \*prevent\*.

For brevity, Entity Type 1-Entity Type 2-Relationship is used in the following text.

### ACCEPTABLE ENTITY RELATIONSHIPS

- Any entity can have a \*prevent\* relationship with another entity of the same type. For example, `Order-Order-*prevent*` or `Grouping-Grouping-*prevent*`.
- Any entity, except Order, can have both \*force\* and \*prevent\* relationship with Equipment. For example, `Zone-Equipment-*force*` or `Order-Equipment-*prevent*`.
- Order can only have a \*prevent\* relationship with Equipment. Because, the Orders table has an `Equipment ID` field that can be used to force the type of equipment for shipping the order. Since the user has that option and to avoid any unforeseen conflict, `order-equipment-*force*` is not allowed in the `Entity Relations` table.

Note that for validation, all entities are converted to their respective orders. For example, `Zone-Equipment-*prevent*` relation is converted to `Order-Equipment-*prevent*`. The only exception is when the relation is `Grouping-Grouping-*prevent*`. That's why, if conflicting relations are entered in the table, the validation tables show those conflicts in the converted version.

### USE CASE EXAMPLES

- If there are some orders that can be delivered with any equipment, the `Equipment ID` field of `Orders` table can take \*all\*. But if there are a few orders that cannot use a certain equipment (for example, LTL), then those orders can be defined in the `Entity Relations` table with a \*prevent\* relationship with LTL.
- Orders of Amazon customers should never be paired with orders of Walmart customers on the same truck. Define a `customer-customer-*prevent*` relation.
- After analyzing the solution, the user notices that there are some routes that combine orders of far away locations. User decides to create new zones such as northwest, midwest, southeast, and so on. By defining a `zone-zone-*prevent*` relation, the user can prevent orders of northwest zone from going with southeast zone.

## Route Generation Nuances

First, two terms should be defined:

1. **Order infeasibility:** when an order cannot be routed due to various reasons such as tight time windows, no cost, large capacity, and so on.
2. **Model infeasibility:** when the mathematical optimization model cannot find a feasible solution that satisfies all the constraints.

There are several route generation algorithms that one can use, whether by themselves or in combination with other algorithms. Except for "Enable Neighborhood Search", others are disabled (have a value of "No") by default. Note that the route generation algorithms run for each equipment configuration separately.

### SHIP ALONE

There are some nuances in using this algorithm, which are explained here.

As the name suggests, if enabled, create routes where each order is shipped by itself. This method can be used as a quick way to check the order feasibility from a high-level. Because if there are records in the **Unrouted Orders** table, one can quickly see what are the underlying causes for their infeasibility and fix them, if needed. In some special setups, getting some unrouted orders by only running this algorithm, may not be conclusive of the order infeasibility. For example, it may be possible not to have the direct cost between two locations, but combining that order with other orders in a multi-stop route, will give us a cost, and thus a feasible route. Or it's possible that due to the working days and hours of the origin and destination sites of an order and considering HoS rules, the order cannot be shipped by itself. But combining it with other orders will avoid this issue.

Interestingly, "Ship Alone" can be used in combination with other methods in hope of helping them avoid some model infeasibility issues. Let's explain this with an example. Assume there are 3 orders and using some route generation methods, we could only create the following 2 routes for them: `R1 = {1, 2}, R2={2, 3}`. Since in the final solution each order should be covered, and it can also only be present in one route, this clearly creates an infeasible model. Because it's not possible to cover all orders without making order 2 be in two routes. In a case like this, having orders generated by "Ship Alone" can help avoid this issue since 3 more routes will be generated, each containing only one order. That way, the optimal solution may end up being `R2={2, 3}, R3={1}`.

Note that it's not suggested to always activate "Ship Alone" to help with issues like above. The modeler should just be aware that such situations can happen and there are tools at their disposal to explore. Activating "Ship Alone" or any of the other route generation methods for that matter can increase the run time of the model, without adding much value.

## Variability In Model Outputs

When running a model multiple times, it's possible to get different results, both in terms of the generated routes, and the solution KPIs like total cost and the number of routes. One may ask how is this possible?

To understand why this happens, one first needs to know the solution process steps.

### SOLUTION PROCESS

The following steps are taken to create the outputs from the input data:

1. **Data Preparation:** After the initial data validation, the raw input data is processed and converted into model-ready data.
2. **Model Decomposition:** Based on the [decomposition parameters](#), different *Equipment ID* in *Equipment Configurations* table, and whether *Depots* table is provided, the input data will be decomposed into independent partitions.
3. **Route Generation:** Based on the [route generation](#) parameters, different algorithms will be used for each partition to create a set of valid routes; i.e., routes that satisfy all the imposed constraints through different tables. They are called *candidate routes*.
4. **Route Selection:** A mathematical optimization model with the objectives and parameters defined in the [optimization parameters](#) is solved to find the best set of routes from the candidate routes.
5. **Vehicle Assignment:** If *Depots* table is provided, more likely than not, each depot has a limited amount of equipment to use. In these cases, the optimal set of routes cannot simply be returned. These routes need to be assigned to different vehicles, compatible with their equipment type. In those cases an additional step is required to assign routes to vehicles.
6. **Output Generation:** The optimal set of routes and vehicles (if applicable) are used to create the output reports.

There is a lot of variability in each of route generation, route selection, and, if needed, vehicle assignment steps.

Route generation algorithms are based on heuristics which can generate different routes in each run.

Route selection is done by a mathematical optimization model. However, the number of variables in this model is usually hundreds if not thousands of times more than the number of constraints which almost guarantees to have many solutions with the same objective values. Not to mention, this model also has an optimality gap and a time limit that also affects the obtained solutions.

Vehicle assignment is also done by a heuristic which means it can generate different results in different runs, especially when there are ties between different routes that can be assigned to a vehicle.

Putting it all together, except in some small models with very tight constraints, it's very unlikely to get the same solution from different runs.

## Inconsistent Unrouted Orders in Model Runs

When running the same model multiple times, the number of unrouted orders change from one run to another. How can this be explained?

First and foremost, the generated solutions are the results of many algorithms (most of them heuristics) and it's inevitable to obtain different solutions from one run to another, as explained [here](#).

Having said that, this question ultimately translates to:

**"How can an order be unrouted in one run and be routed in another?"**

An unrouted order has violated some constraints. But if it's routed in another run, then the violation can be narrowed down to a few constraints:

- *Transit time window:* This is the most likely scenario. The order could not be routed by itself since it violates the transit time window but when routed with other orders, the time window becomes feasible. So, in one run, the order didn't get the chance to be routed with the right orders to become feasible.
- For illustration, consider an order that can only be picked and dropped from 8:00 to 17:00 each day, but it takes 10 hours to go from its origin to destination. There is no feasible route that can ship this order alone. However, consider a second order with the same pick and drop window, the same pick location but a different destination, and a transit time of 6 hours. If these two orders are combined in a single-pick multi-drop route, where both are picked and then the second order is dropped first and the first order is dropped last, and considering the hours of service rules, one can easily see that the first order can be easily dropped without any time window violation. Here is how the math works:  
6-hour drive to second order's destination + 5-hour drive + 10-hour off-duty + 5-hour drive to first order's destination. Arrival will be the next day during the destination's open hours.
- *Cost:* The second most common scenario is due to cost. Similar to the explanation for time window, assume that the rate between an order's origin and destination does not exist. However, when combined with another order in a multi-stop route, the route's origin and destination will be different from the given order's origin and destination, and it's possible that the route for that route exists. Consider the example above with the two orders and this time, assume that there was no rate between origin and destination location of the second order. By being combined with the other order, the route's origin and destination change and finding a rate became possible.
- *Site sequence or order sequence:* These are constraints that force a given site/order to be first/last pick/drop on a route. Although not very common, if one of the site or order sequence constraints is active, it's possible that in one run, the order was only considered for shipment with other orders (and violating these constraints) while in another run, it might have been considered for being shipped alone, which could potentially lead to it being routed.
- *Insufficient equipment:* In cases where a vehicle assignment step is required and no excess vehicle is allowed, it's possible to have valid routes that could not be assigned to any of the available vehicles. These routes and their orders are then considered for being shipped with any available third party equipment (such as LTL). If then, for instance, any of these uncovered orders violate the third party capacity constraints, they become unrouted. Since the vehicle assignment is done by a heuristic, it can generate different results in different runs. This means, it's possible in one run to discard an extra vehicle with only one route that has a 30k order and in another run, discard a route that has several 5k orders.

In the first case, the 30k order violates the LTL capacity and becomes unrouted while in the second case, all can be routed.

## Unrouted Orders Despite Multiple Equipment

Multiple equipment exist and the orders should be covered by at least one of them. However, there are still unrouted orders and the given reason doesn't seem accurate. How can this be explained?

If the infeasibility reason doesn't seem right, more likely than not, either the order has an invalid transit time window or no cost.

Let's investigate the constraints that are validated one-by-one. The first set of constraints are *sequence-based* constraints that depend on the sequence of stops. These include:

- Max number of picks/drops/stops
- Site sequence (site first pick on route, last pick on route, first drop on route, last drop on route)
- Max distance between picks/drops
- Max distance
- Max out of route and out of route percentage
- Max first pick to last pick
- Max first pick to first drop
- Max first drop to last drop
- Max backhaul distance

If all of these are valid, then *order-based* constraints are validated. These include:

- Order pairing (orders don't appear in any `order-order-*prevent*` type constraints)
- Order grouping (orders don't appear in any `grouping-grouping-*prevent*` type constraints)
- Capacity constraints (weight/volume/cube adjusted weight/pallets)
- Order sequence (order first pick on route, last pick on route, first drop on route, last drop on route)
- Stop time window (orders combined together in a given stop, have common time windows for the pickup or delivery)
- Transit time window (it's possible to go from one stop to another, considering the available time window, service hours, transit time, and hours of service rules)
- Equipment Cost

If all of these constraints are also valid, then the route and consequently the orders are valid.

If there is a vehicle assignment step and no excess vehicle is allowed, it's possible to have routes that could not be assigned to any of the available vehicles, and thus, the routes and their orders become unrouted. In these cases, the reported reason is "Insufficient Equipment".

If there are predefined routes, then there are a few more constraints that should be checked:

- Orders that are part of the same route must belong to the same "Grouping Time Field". So, if the parameter value is *Day*, then all those orders must be from the same day.
- Orders that are part of the same route must belong to the same partition (partitioning happens based on the value of "Model Partitioning Criteria" parameter)

Keep in mind that if an order can be considered with multiple equipment, only one infeasibility reason can be saved. This is already explained with some examples in [Unrouted Orders](#). With that in mind, check the following for any of the unrouted orders to get to the root of infeasibility. Consider only one order at a time.

- Consider shipping the order alone.
- Check *capacity constraints* for any of the equipment and discard the ineligible ones (for example, if the order's weight is 30k, and the max weight of LTL equipment is 20k, LTL is invalid equipment for this order).
- Check the sequence-based constraints in the model if there are any. Try to calculate the value of the active constraints for the order. For a single order, only **Max Distance**, **Max First Pick To First Drop** and **Max Backhaul** (in case of having closed-loop route) need to be checked.
- If the *sequence-based* constraints are valid, check if the model has any order first/last pick/drop on route.
- For time window validation, keep in mind a few things:
  - Calculate the distance from origin to destination and divide it by speed of the equipment. This gives you the transit time.
  - Consider the values in the **Hours Of Service Rules** table
  - Every 8-hour of drive, there is half-an-hour of break
  - From **Site Operating Hours** take the open/close times and working days of origin and destination
  - With the above values, calculate the duration of time that's needed going from origin to destination
  - If the route is open, check these values to see if it's possible to leave the origin during the pickup window and arrive at the destination, given the calculated duration.
  - If the route is closed, continue with calculating duration from destination back to origin. Keep in mind that the values don't reset, but are added to what has already been calculated. So for example, if the duration from origin to destination is 28 hours and already 7 hours of drive and on-duty has been consumed, they should be considered in calculating the duration from destination back to origin. So, even though it may seem that from destination to origin should also take 28 hours, considering that 7 hours of drive time have already been consumed, only 4 hours can be driven before taking 10 hours of off-duty. Then another 11 hours of drive, before taking another 10 hours of off-duty, followed by another 3 hours of drive, which will bring the total duration to 38 hours (the 0.5-hour break was not considered here for simplicity).
  - If even time-window seems valid, check there is cost between origin and destination.
  - Please remember that if LTL is valid equipment, there should be costs available in the **LTL Rates** table for the order's weight break.

If after checking all of these constraints, the infeasibility reason for the order still seems inaccurate, please contact the support team to investigate further.

## 4.4 Shipday Optimizer

---

### 4.4.1 Shipday Optimizer

#### WHAT IS SHIPDAY OPTIMIZER?

Shipday Optimizer streamlines shipping logistics by intelligently assigning routes to specific days of the week. It optimizes two key objectives: minimizing the number of days to cover customer regions and evenly balancing routes across the week. Users can customize priorities based on their needs. This tool offers essential input tables, output reports, and a mathematical optimization model to achieve efficient route scheduling, reducing operational costs, and enhancing delivery efficiency.

👉 Use the navigation to explore further details

## 4.4.2 Input Tables

---

### Main Data

#### STOPS

Contains information about each stop of a route.

##### Route ID

*Type: Text*

Unique ID for the route.

##### Stop Number

*Type: Numeric*

Position of the stop on the route.

##### Stop Action

*Type: Text*

What action is performed at the stop. Possible options: **Pick**, **Drop**, **Backhaul** or **Deadhead**.

##### Location ID

*Type: Text*

Unique location ID of the stop.

##### Location Type

*Type: Text*

Type of Location for the stop. Possible options: **Ship From**, **Ship To**, **Crossdock** or **Depot**.

##### Location City

*Type: Text*

City of the stop location.

##### Location State

*Type: Text*

State of stop location

##### Location Postal Code

*Type: Text*

Postal code of the stop location.

##### Location Country

*Type: Text*

Country of the stop location.

##### Location Latitude

*Type: Numeric*

Latitude of the stop location.

##### Location Longitude

*Type: Numeric*

Longitude of the stop location.

##### Equipment ID

*Type: Text*

Equipment used to ship this order.

## Distance To Next Stop

*Type: Numeric*

Distance to the next stop in the route.

## Earliest Arrival Time

*Type: DateTime*

The earliest date and time the equipment arrives at the stop location.

## Latest Arrival Time

*Type: DateTime*

The latest date and time the equipment arrives at the stop location.

**REGIONS**

Table that maps each postal code to a region.

**Location Postal Code**

*Type: Text*

Postal code of a stop.

**Region**

*Type: Text*

Region that the postal code belongs to.

**Sample Input**

<b>Location Postal Code</b>	<b>Region</b>
12222	North-East
60660	Mid-West
32003	South

**DEPOTS**

Table that contains information on the origin location of the routes.

**Depot ID**

*Type: Text*

Identifies the depot.

**Equipment ID**

*Type: Text*

Identifies accessible equipment at the depot.

**Closed Loop**

*Type: Text*

Indicates whether the Equipment should return to the Depot after last stop. Can be *Yes* or *No*.

**Equipment Count**

*Type: Numeric*

Count of Equipment available at the Depot.

**Sample Input**

<b>Depot ID</b>	<b>Equipment ID</b>	<b>Closed Loop</b>	<b>Equipment Count</b>
Chicago	Dry-Van	Yes	100
Atlanta	Reefer	No	29

## PARAMETERS

In this table the user can configurate how the optimization model will behave, setting up different parameters that will influence the optimization outputs.

### Key

*Type: Text*

Name of the parameter.

### Value

*Type: Text*

Value of the parameter. Edit based on the description.

### Category

*Type: Text*

A field used to categorize and organize parameters in sections.

### Description

*Type: Text*

Description of the parameter. Provides information about the values parameters can take.

### Parameters explanation

**Truck Balance Objective Priority:** Priority of balancing number of trucks per day. Larger the number, higher the priority. The magnitude is irrelevant.

**Single Day Region Objective Priority:** Priority of having fewer ship days to cover each region. Larger the number, higher the priority. The magnitude is irrelevant.

**Solver Optimality Gap Percentage:** This is a non-negative number that's defined as a percentage and will be passed to Gurobi. Acceptable values are in the range of (0.01 to 100), excluding 100. Default value is 0.5.

**Solver Run Time Limit (Seconds):** Max time that the optimization model can run. Leaving this blank tells the solver to take as much time as needed to solve the model. Be advised that setting a small value for time limit may result in sub-optimal solution. If the value is very small, the model may struggle to even find a feasible solution. Default value is 3600.

**Relative Tolerance:** Allowable relative degradation for the highest priority objectives, when solving for lower priority objectives. Defined as a percentage, not as a ratio.

**Absolute Tolerance:** Allowable absolute degradation for the highest priority objectives, when solving for lower priority objectives.

### Sample Input

Key	Value	Category	Description
Truck Balance Objective Priority	2	Optimization	Priority of balancing number of trucks per day. Larger the number, higher the priority. The magnitude is irrelevant.
Single Day Region Objective Priority	1	Optimization	Priority of having fewer ship days to cover each region. Larger the number, higher the priority. The magnitude is irrelevant.
Solver Optimality Gap Percentage	0.01	Optimization	This is a non-negative number that's defined as a percentage and will be passed to Gurobi. Acceptable values are in the range of (0.01 to 100), excluding 100.

## Constraints

### REGION RESTRICTIONS

Populate this table if shipments for some regions must leave on certain ship days. Note that if you want to use this table, a Region cannot have more than one Ship Day.

Region

*Type: Text*

Region with a restriction.

Ship Day

*Type: Text*

The day of week that region must be shipped on.

Sample Input

Region	Ship Day
North-East	Monday
Mid-West	Wednesday
South	Thursday

#### PREDEFINED SHIP DAY PERCENTAGE

Populate this table to predefine percentage of routes per depot that should be assigned to a given ship day.

##### Depot ID

*Type: Text*

Unique value to identify the depot

##### Ship Day

*Type: Text*

Day of the week when the shipment leaves the origin location.

##### Route Percentage

*Type: Numeric*

A number between 0 and 100 (both inclusive) on percentage of routes that should leave on the given ship day.

If left empty, the model will decide based on objective priority.

##### Sample Input

Depot ID	Ship Day	Route Percentage
Chicago	Monday	10
Chicago	Wednesday	20
Chicago	Tuesday	70

##### Tips for populating this table

Sum of Route Percentage values must add up to 100%.

The Ship Day and Route Percentage values for a given Depot ID must match existing depots and span of routes' pickup windows in the Stops table, otherwise a mismatch occurs. Here are a few examples on the 3 types of mismatches that can happen:

- If the percentages are entered for depot "D10" but there is no pickup location associated with "D10" in the Stops table.
- If routes associated with depot "D1" (a valid depot), have a pickup time window between Monday to Friday, but the user only enters percentages for Monday to Wednesday. It's acceptable to enter a 0 percentage for Thursday and Friday if that's what the user wishes, but they cannot be ignored all together.
- If routes associated with depot "D1" in the Stops table only have a pickup window between Monday to Wednesday but in Predefined Ship Day Percentage, values of 25% is provided for Monday to Thursday. In this case, user wants 25% of routes to go on a Thursday, but Thursday doesn't even exist in the data.

##### Important Notes

- To avoid any problem, it's advised to populate the Predefined Ship Day Percentage table for the desired depots and enter the percentages for all the 7 days of the week, even if it's 0 for some of them. It's not necessary to put the values for all the depots, if that's not desired. So, in a dataset with 10 depots, it's totally acceptable for the user to only enter route percentages for two depots and leave the rest to the model to decide.
- Leaving out the entry for a depot is not necessarily equivalent to have an equal percentage for all days. For example, if for depot "D1", the routes span over 5 days, leaving the percentage information of "D1" out of Predefined Ship Day Percentage, doesn't necessarily mean that every location will have 20% of routes. To understand why this is the case, you need to consider how the objective of balancing number of routes is calculated. Leaving out the information, tells the model to try to minimize the difference between the max and min value of number of routes. So, the following three solutions (with 100 routes to distribute), assuming everything else is equal, provide the same value for the route-balance objective:
  - Solution 1: Mon 45, Tue 15, Wed 15, Thu 15, Fri 10; min = 10, max = 45. Diff = 35
  - Solution 2: Mon 40, Tue 30, Wed 10, Thu 10, Fri 10; min = 10, max = 45. Diff = 35
  - Solution 3: Mon 40, Tue 20, Wed 20, Thu 10, Fri 10; min = 10, max = 45. Diff = 35

## Visualization Data

### LOCATIONS LOOKUP

This table is populated to help with report generation, and expand 3-zips to county / 5-zips, and lower granularity.

#### Zip Code

*Type: Text*

Provide a zip code.

#### Zip 3

*Type: Text*

Left-3 characters of the Zip Code.

#### City

*Type: Text*

City of the Zip Code.

#### County

*Type: Text*

County of the Zip Code.

#### State Code

*Type: Text*

State Code of the Zip Code.

#### State Name

*Type: Text*

State Name of the Zip Code

#### Latitude

*Type: Numeric*

Used for Haversine distance calculations and mapping.

#### Longitude

*Type: Numeric*

Used for Haversine distance calculations and mapping.

#### Sample Input

Zip Code	Zip 3	City	County	State Code	State Name	Latitude
19901	199	Dover	Kent	DE	Delaware	39.14
79901	799	El Paso	El Paso	TX	Texas	31.76
12288	122	Albany	Albany	NY	New York	42.65

### 4.4.3 Solution

---

#### **Summary Reports**

##### **DEPOT ROUTE STATS**

Reports on number of routes per shipping day per depot.

Depot ID

*Type: Text*

Unique value to identify a Depot.

Ship Day

*Type: Text*

Day of the week when the shipment leaves the origin location.

Route Count

*Type: Numeric*

Number of Routes on a given Ship Day.

Route Percentage

*Type: Numeric*

Percentage of routes that leave on the given Ship Day.

**SUMMARY**

A report of the main KPIs. Currently, it only reports Total Runtime (Seconds) and Timestamp of the run.

**Key**

*Type: Text*

Name of the KPI.

**Value**

*Type: Text*

Value of the KPI.

## Detailed Reports

### ASSIGNMENT

Reports which route is assigned to which ship-day.

#### Route ID

*Type: Text*

Unique ID for a route.

#### Start Location ID

*Type: Text*

Start location of the route.

#### Location State

*Type: Text*

State of the stop on route.

#### Location Postal Code

*Type: Text*

Postal code of the stop on route.

#### Region

*Type: Text*

Region that the postal code belongs to.

#### Ship Day

*Type: Text*

Day of the week when the shipment leaves the origin location.

**REGION DAY ASSIGNMENT**

For each depot, it shows which region is associated to which state and ship-day.

**Depot ID**

*Type: Text*

Unique value to identify a Depot.

**Region**

*Type: Text*

Unique ID of a region.

**Ship To State**

*Type: Text*

Ship To State associated with the region.

**Ship Day**

*Type: Text*

Day of the week when the shipment leaves the origin location.

## 4.5 SMC Rater

---

### 4.5.1 SMC Rater

#### WHAT IS SMC RATER?

SMC Rater provides LTL ratings and costs for the shipments requests based on SMC3.

👉 Use the navigation to explore further details

## 4.5.2 Input Tables

---

### Details

Input table for details of the locations.

#### ID

*Type: Text*

Unique identification for each request.

#### Origin Postal Code

*Type: Text*

5-digit Postal Code for the Origin Location.

#### Destination Postal Code

*Type: Text*

5-digit Postal Code for the Destination Location.

#### Weight Break

*Type: Text*

One of {'L5C', 'L1M', 'M1M', 'M2M', 'M5M', 'M10M', 'M20M', 'M30M', 'M40M'} upto "Maximum Weight Break" parameter to get rating for a weight band. If an average shipment is provided, it takes precedence over Weight Break. See Average Shipment documentation page for more information.

#### Avg Shipment

*Type: Integer*

Weight of the average shipment. Non-negative, non-infinite, nullable. If an average shipment is provided here, it takes precedence over Weight Break.

#### Tariff Name

*Type: Text*

Name of the Tariff the SMC rater should use to rate the shipment.

#### Tariff Date

*Type: Text*

The date the SMC rater should use to rate the shipment. Only the year is needed, but any string that looks like a date will be understood.

#### Freight Class

*Type: Text*

The National Motor Freight Classification Code of the specified shipment line. One of (200, 150, 300, 60, 65, 70, 100, 92, 250, 92.5, 85, 77, 55, 175, 50, 110, 400, 77.5, 500, 125)

#### Percent Discount

*Type: Float*

Percent by which the rating cost should be reduced. Number between 0 (inclusive) and 100 (exclusive).

#### Minimum Charge Floor

*Type: Float*

User defined value that is an absolute minimum charge that is not subject to discounting. Non-negative, non-infinite. This is passed to the userMinimumChargeFloor component of the ltrateshipment payload.

## Sample Input

ID	Origin Postal Code	Destination Postal Code	Weight Break	Avg Shipment	Tariff Name	Tariff ID
1	43420	15084	M10M		DEMOLTLC	200707
2	43420	18071		15000	DEMOLTLC	200707
3	43420	16281	M20M		DEMOLTLC	200707
4	19403	15084		8000	DEMOLTLC	200707

## Parameters

In this table, user can enter their smc credentials to load SMC3 Rater and specify the maximum weight break (if applicable).

### Key

*Type: Text*

Name of the parameter.

### Value

*Type: Text*

Value of the parameter. Edit based on the description.

### Parameters explanation

**SMC Username:** Username to load SMC3.

**SMC License:** License Key to load SMC3.

**SMC Password:** If you append '\_ENCRYPT\_ME' to this value, and then upload this table using Google Sheets, then an encrypted version of the password will be stored. It is also encrypted if '\_ENCRYPT\_ME' is at the end of the value uploaded from CSV or Excel.

**Maximum Weight Break:** One of {'L5C', 'L1M', 'M1M', 'M2M', 'M5M', 'M10M', 'M20M', 'M30M', 'M40M} to determine the maximum Weight Break to return for Transportation Optimizer app. Default value is 'M20M'.

## 4.5.3 Solution

---

### Results

Detailed results table for LTL ratings based on SMC3.

#### ID

*Type: Text*

Unique identification for each request.

#### Origin Postal Code

*Type: Text*

5-digit Postal Code for the Origin Location.

#### Destination Postal Code

*Type: Text*

5-digit Postal Code for the Destination Location.

#### Weight Break

*Type: Text*

One of {'L5C', 'L1M', 'M1M', 'M2M', 'M5M', 'M10M', 'M20M', 'M30M', 'M40M'} upto "Maximum Weight Break" parameter to get rating for a weight band. If an average shipment was provided, it takes precedence over Weight Break. See Average Shipment documentation page for more information.

#### Avg Shipment

*Type: Integer*

Weight of the average shipment. Non-negative, non-infinite, nullable. If an average shipment was provided here, it takes precedence over Weight Break.

#### Tariff Name

*Type: Text*

Name of the Tariff the SMC rater used to rate the shipment.

#### Tariff Date

*Type: Text*

The date the SMC rater used to rate the shipment.

#### Freight Class

*Type: Text*

The National Motor Freight Classification Code of the specified shipment line. One of (200, 150, 300, 60, 65, 70, 100, 92, 250, 92.5, 85, 77, 55, 175, 50, 110, 400, 77.5, 500, 125)

#### Percent Discount

*Type: Float*

Percent by which the rating cost is reduced. Number between 0 (inclusive) and 100 (exclusive).

#### Minimum Charge Floor

*Type: Float*

User defined value that is an absolute minimum charge that is not subject to discounting. Non-negative, non-infinite. This is passed to the userMinimumChargeFloor component of the ltlrateshipment payload.

#### Rate

*Type: Float*

The rate for the specified shipment line.

**Minimum Charge***Type: Float*

Minimum Charge Amount for the shipment.

**Gross Charge***Type: Float*

Gross Charge Amount for the shipment.

**Total Charge***Type: Float*

The net charge for the shipment = Gross charge plus surcharges and applied discounts

**Billed Weight***Type: Float*

The weight at which the shipment is rated; Actual Weight or Actual Weight plus Deficit Weight. Deficit Weight is the amount of additional weight necessary for the use of lower rates.

**Discount Amount***Type: Float*

A dollar and cents amount of the calculated Discount.

**Currency***Type: Text*

Currency Alpha Code (USD, CAD, MXN)

## Unrated Requests

Output table that lists the requests that were not rated by SMC3 and corresponding error.

### ID

*Type: Text*

Unique identification for each request.

### Origin Postal Code

*Type: Text*

5-digit Postal Code for the Origin Location.

### Destination Postal Code

*Type: Text*

5-digit Postal Code for the Destination Location.

### Weight Break

*Type: Text*

One of {'L5C', 'L1M', 'M1M', 'M2M', 'M5M', 'M10M', 'M20M', 'M30M', 'M40M'} upto "Maximum Weight Break" parameter to get rating for a weight band. If an average shipment was provided, it takes precedence over Weight Break. See Average Shipment documentation page for more information.

### Avg Shipment

*Type: Integer*

Weight of the average shipment. Non-negative, non-infinite, nullable. If an average shipment was provided here, it takes precedence over Weight Break.

### Tariff Name

*Type: Text*

Name of the Tariff the SMC rater used to rate the shipment.

### Tariff Date

*Type: Text*

The date the SMC rater used to rate the shipment.

### Freight Class

*Type: Text*

The National Motor Freight Classification Code of the specified shipment line. One of (200, 150, 300, 60, 65, 70, 100, 92, 250, 92.5, 85, 77, 55, 175, 50, 110, 400, 77.5, 500, 125)

### Percent Discount

*Type: Float*

Percent by which the rating cost should be reduced. Number between 0 (inclusive) and 100 (exclusive).

### Minimum Charge Floor

*Type: Float*

User defined value that is an absolute minimum charge that is not subject to discounting. Non-negative, non-infinite. This is passed to the userMinimumChargeFloor component of the ltlrateshipment payload.

### Error

*Type: Text*

Error Code/Reason to not rate this request from SMC3.

## 4.6 Geo Coder

---

### 4.6.1 Geo Coder

#### WHAT IS GEO CODER?

Geo Coder generates latitude and longitude coordinates when provided with an address in a specific format (Street, City, State, Country, and Postal Code).

👉 Use the navigation to explore further details

## 4.6.2 Input Tables

---

### Details

Input table for details of the locations.

#### ID

*Type: Text*

Unique ID to identify the address.

#### Street

*Type: Text*

Street corresponding to the address. (Optional)

#### City

*Type: Text*

City corresponding to the address. (Optional)

#### State

*Type: Text*

State corresponding to the address. (Optional)

#### Country

*Type: Text*

2 digit Country Code corresponding to the address. (Default is US)

#### Postal Code

*Type: Text*

3-digit or 5-digit Postal code corresponding to the address.

#### Sample Input

ID	Street	City	State	Country	Location Postal Code
1	303 W Green St	Champaign	IL	US	61820
2		Chicago	IL	US	60614
3			GA	US	30320
4				IN	400053

## Parameters

In this table, users have the option to input their Google Maps API key in order to retrieve geographic coordinates using the Google Maps service.

### Key

*Type: Text*

Name of the parameter.

### Value

*Type: Text*

Value of the parameter. Edit based on the description.

### *Parameters explanation*

**GoogleMaps API Key:** Fill as required.

## 4.6.3 Solution

---

### Results

This table provides information about the longitude, latitude and error if relevant.

#### ID

*Type: Text*

Unique ID to identify the address.

#### Street

*Type: Text*

Street corresponding to the address.

#### City

*Type: Text*

City corresponding to the address.

#### State

*Type: Text*

State corresponding to the address.

#### Country

*Type: Text*

2 digit Country Code corresponding to the address.

#### Postal Code

*Type: Text*

3-digit or 5-digit Postal code corresponding to the address.

#### Latitude

*Type: Float*

Latitude of the location.

#### Longitude

*Type: Float*

Longitude of the location.

#### Error

*Type: Text*

Error subtext, if relevant.

## 4.7 Seasonality Detector

---

### 4.7.1 Seasonality Detector

#### WHAT IS SEASONALITY DETECTOR?

Seasonality Detector offers valuable insights into time series data. It identifies seasonal patterns, measures their strength, and automatically segments the year into seasons when seasonal strength surpasses a predefined threshold. Additionally, it groups items according to their seasonal patterns, providing information about prevalent seasonal trends and item similarities.

👉 Use the navigation to explore further details

## 4.7.2 Input Tables

### Parameters

In this table user can configure how seasonality is detected and how time series clustering is performed.

#### Key

*Type: Text*

Name of the parameter.

#### Value

*Type: Text*

Value of the parameter. Edit based on the description.

#### Category

*Type: Text*

A field used to categorize and organize parameters in sections.

#### Description

*Type: Text*

Description of the parameter. Provides information about the values parameters can take.

#### Parameters explanation

**Seasonality Time Units:** Level of granularity at which time series data will be grouped into seasons. Should be one of 'Day', 'Week', or 'Month'. If 'Month' is selected, for example, each month would be viewed as a unit, and each season is made up of several months.

**Seasonality Strength Threshold:** Seasonal strength varies between 0 and 1. Higher the seasonal strength, stronger the seasonality. This parameter defines the minimum seasonal strength required to perform steps to divide time series into seasons.

**Number of Seasons:** Number of seasons to group periods into. Periods can be grouped into a minimum of 2 seasons and a maximum of 4 seasons.

**Normalization Method:** The approach used to normalize seasonal components to make seasonal patterns comparable. Should be one of 'Standard', 'Min-Max', and 'Robust'.

**Number of Clusters:** Number of clusters for k-means clustering based on seasonal patterns. k clusters means the items will be grouped into k groups.

**Number of Runs:** Number of times the k-means algorithm is run with different centroid seeds.

**Number of Iterations:** Maximum number of iterations of the k-means algorithm for a single run.

#### Sample Input

Key	Value	Category	Description
Seasonality Time Units	Month	Seasonality Detection	Level of granularity at which time series data will be grouped into seasons. One of ('Day', 'Week', 'Month')
Seasonality Strength Threshold	0.8	Seasonality Detection	Minimum seasonality strength required to perform seasonal grouping (Between 0 and 1)
Normalization Method	Standard	Seasonality Detection	Number of seasons to group periods into. Periods can be grouped into a minimum of 2 seasons and a maximum of 4 seasons

## Time Series

Populate this table with the time series data that needs seasonality detection. Length of each time series need to be at least 2 years for detection to run.

### ID

*Type: Text*

Provide a unique ID (product name, etc.) for each time series.

### Date

*Type: Datetime*

Date of time series data.

### Value

*Type: Numeric*

Value of time series data. Value can be negative.

### Sample Input

ID	Date	Value
Product1	1/9/2022	100
Product1	2/15/2022	550
Product2	1/19/2022	0

## 4.7.3 Solution

### Seasonal Patterns

This table shows the seasonal pattern in a year for each item.

#### ID

*Type: Text*

Unique ID (product name, etc.) for each time series.

#### Period

*Type: Numeric*

Day number, week number or month number in a year.

#### Value

*Type: Numeric*

Average and normalized seasonal component for each day, week or month in a year.

## Seasonality Summary

This table show some summary data on the seasonality of each time series.

### ID

*Type: Text*

Unique ID (product name, etc.) for each time series.

### Seasonal Strength

Seasonal strength of each item, which varies between 0 and 1. Higher the seasonal strength, stronger the seasonality.

### Season 1 Start

*Type: Numeric*

The day number, week number or month number in a year that starts season 1. Time series with lower seasonality strength than threshold will not be divided into seasons, so this field is empty for them.

### Season 2 Start

*Type: Numeric*

The day number, week number or month number in a year that starts season 2. Time series with lower seasonality strength than threshold will not be divided into seasons, so this field is empty for them.

### Season 3 Start

*Type: Numeric*

The day number, week number or month number in a year that starts season 3. Time series with lower seasonality strength than threshold will not be divided into seasons, so this field is empty for them.

### Season 4 Start

*Type: Numeric*

The day number, week number or month number in a year that starts season 4. Time series with lower seasonality strength than threshold will not be divided into seasons, so this field is empty for them.

### Cost Value

*Type: Numeric*

Value of the cost function when searching for optimal breakpoints to divide time series into seasons. Lower cost value indicates smaller difference within seasons. Time series with lower seasonality strength than threshold will not be divided into seasons, so this field is empty for them.

### Cluster ID

*Type: Numeric*

Unique ID of the cluster each item is assigned to. Items with lower seasonality strength than threshold don't participate in clustering, so this field is empty for them.

## Clusters

This table contains information on the clusters. The starting period of seasons for each cluster is based on the average seasonal patterns of items assigned to each cluster.

### Cluster ID

*Type: Numeric*

Unique ID of each cluster.

### Season 1 Start

*Type: Numeric*

The day number, week number or month number in a year that starts season 1.

### Season 2 Start

*Type: Numeric*

The day number, week number or month number in a year that starts season 2.

### Season 3 Start

*Type: Numeric*

The day number, week number or month number in a year that starts season 3. Displays 0 if number of seasons is smaller than 3.

### Season 4 Start

*Type: Numeric*

The day number, week number or month number in a year that starts season 4. Displays 0 if number of seasons is smaller than 4.

## 4.8 Inventory Simulator

---

### 4.8.1 Inventory Simulator

#### WHAT IS INVENTORY SIMULATOR?

Inventory Simulator is a powerful tool designed to evaluate the efficiency of supply chain operations under specific safety stock configurations. It is capable of running simulations using actual demand and forecast data, as well as generating synthetic demand and forecast data. By mimicking supply chain processes, this simulator provides comprehensive insights into how well the supply chain performs under predefined conditions and delivers detailed performance reports.

👉 Use the navigation to explore further details

## 4.8.2 Input Tables

### Core Data

#### SITES

Populate this table to create sites. Sites should be unique.

Name

*Type: Text*

Provide a unique ID for each site to simulate.

Sample Input

Name	Active
Site1	True
Site2	True

**PRODUCTS**

Populate this table to create products. Product names should be unique.

Name

*Type: Text*

Provide a unique ID for each product to simulate.

Sample Input

<b>Name</b>	<b>Active</b>
Product1	True
Product2	True
Product3	True

## PARAMETERS

In this table user can provide more information on input data's format and configure parameters that will have an impact on simulation output.

### Key

*Type: Text*

Name of the parameter.

### Value

*Type: Text*

Value of the parameter. Edit based on the description.

### Description

*Type: Text*

Description of the parameter. Provides information about the values parameters can take.

### Parameters explanation

**Baseline Fill Rate:** A general target percentage of demand fulfilled on time from inventory. It is used to compare with the simulation result when producing reports and doesn't have an impact on the simulation outcome.

**Days Per Month:** Defines how many days a month has. It is used when converting between time units 'Day' and 'Month'. Value should fall between 28 and 31.

**Demand Forecast Source:** Source of demand and forecast data. If choosing 'Actual', data from table Actual Demand Forecast is used. If choosing 'Simulation', table Demand Forecast Parameters is used as input, and demand and forecast data by period is generated using normal distribution.

**Demand Time Units:** Defines the granularity of demand and forecast data as well as the granularity on which simulation is run. Must be one of 'Day', 'Week' or 'Month'.

**Service Time Units:** Defines the time units for lead time. Must be one of 'Day', 'Week' or 'Month'.

**Total Periods:** Total number of periods to simulate, in the same granularity of Demand Time Units.

**Warm-up Periods:** Number of warm-up periods to be excluded when evaluating simulation performance. Must be smaller than Total Periods.

### Sample Input

Name	Value	Description
Baseline Fill Rate	95	Baseline Fill Rate % (between 0 and 100)
Days Per Month	30	Number of days per month to convert between time units of 'Day' and 'Month'.
Demand Forecast Source	Actual	Source of demand and forecast data. One of ('Actual', 'Simulation'). Data from Actual Demand Forecast table or Demand Forecast Parameters table will be used accordingly.
Demand Time Units	Week	Demand and forecast granularity. One of ('Day', 'Week', 'Month'). Simulation also follows demand time units.
Service Time Units	Week	Time units for lead time. One of ('Day', 'Week', 'Month')
Total Periods	52	Total number of periods to simulate, in the same granularity of Demand Time Units.
Warm-up Periods	8	Number of warm-up periods to be excluded when evaluating simulation performance. Must be smaller than total periods.

## Site Product Data

### ACTUAL DEMAND FORECAST

If 'Actual' is selected as the source of demand and forecast data in the Parameters table, populate this table with actual demand and forecast data for each (Site, Product) combination to simulate.

#### Site

*Type: Text*

Identifies a site. Each site here must match a valid site from the Sites table.

#### Product

*Type: Text*

Identifies a product. Each product here must match a valid product from the Products table.

#### Time Period

*Type: Numeric*

Each number identifies a time period.

#### Demand

*Type: Numeric*

Actual demand in each time period.

#### Forecast

*Type: Numeric*

Forecast in each time period.

#### Sample Input

Site	Product	Time Period	Demand	Forecast
Site1	Product1	1	100	75
Site1	Product1	2	200	215

**DEMAND FORECAST PARAMETERS**

If 'Simulation' is selected as the source of demand and forecast data in the Parameters table, populate this table with parameters for demand and forecast for each (Site, Product) combination to simulate. Demand and forecast by period will be generated using normal distribution with provided parameters. When generating demand and forecast at each period, demand and forecast error are generated first, and then forecast is calculated by adding forecast error to demand.

**Site**

*Type: Text*

Identifies a site. Each site here must match a valid site from the Sites table.

**Product**

*Type: Text*

Identifies a product. Each product here must match a valid product from the Products table.

**Average Demand**

*Type: Numeric*

Average demand in the granularity specified in Parameters table.

**Std Dev Demand**

*Type: Text*

Standard deviation of demand in specified granularity.

**Average Forecast Error**

*Type: Numeric*

Average of forecast error (Forecast - Demand) in specified granularity.

**Std Dev Forecast Error**

*Type: Numeric*

Standard deviation of forecast error in specified granularity.

**Sample Input**

Site	Product	Average Demand	Std Dev Demand	Average Forecast Error	Std Dev Forecast Error
Site1	Product1	244	242.1	-10	224.6
Site2	Product2	1488.3	1134.8	192.8	610.4

**STOCKING NODES**

The Stocking Nodes table contains essential data for simulating and comparing outputs for each (Site, Product) pair. Simulations will only be conducted for the combinations found in this table.

**Site**

*Type: Text*

Identifies a site. Each site here must match a valid site from the Sites table.

**Product**

*Type: Text*

Identifies a product. Each product here must match a valid product from the Products table.

**Average Demand**

*Type: Numeric*

Average demand each period.

**Std Dev Demand**

*Type: Numeric*

Standard deviation of demand each period.

**Incoming Service Time**

*Type: Numeric*

Total lead time for product. If there are multiple sources for a product, this should be the weighted average lead time over all inbound arcs.

**Safety Stock**

*Type: Numeric*

Safety stock in units.

**Safety Stock Days**

*Type: Numeric*

Days of supply as safety stock.



For each row in this table, only one of Safety Stock and Safety Stock Days should have an input value. If both columns are empty or both have values in them, an error will be thrown.

**Fill Rate**

*Type: Numeric*

Target percentage of demand fulfilled on time from inventory. It is not a parameter for simulation and has no impact on simulation results. It is required as a baseline to compare with the simulation fill rate when producing simulation reports.

**Sample Input**

Site	Product	Average Demand	Std Dev Demand	Incoming Service Time	Safety Stock	Safety Stock Days
Site1	Product1	244	242.1	2.3571	517.5	
Site2	Product3	1000	578	2		19

## STOCKING DATA

This table stores replenishment settings for each (Site, Product) combination. Simulations will only be conducted for the combinations found in this table.

### Site

*Type: Text*

Identifies a site. Each site here must match a valid site from the Sites table.

### Product

*Type: Text*

Identifies a product. Each product here must match a valid product from the Products table.

### Review Period

*Type: Numeric*

Review Period defines the frequency inventory level is checked and replenishment is placed.



Currently the application only performs simulation based on periodic review policy, so Review Period must be positive integers.

### Receipt Period

*Type: Numeric*

Typically the same as review period. Cycle stock is controlled by receipt period.

### Minimum Order Size

*Type: Numeric*

Minimum possible size of reorder. If left empty or assigned zero, order can be of any size.

### Order Increment

*Type: Numeric*

Controls the granularity of a reorder.

### Sample Input

Site	Product	Review Period	Receipt Period	Minimum Order Size	Order Increment
Site1	Product1	1	1	100	10
Site1	Product2	1	1	0	1
Site2	Product3	2	2	0	0

**STARTING INVENTORY**

Settings for starting inventory for any (Site, Product) combinations can be specified in the Starting Inventory table. The starting inventory becomes the on-hand inventory for period 0 before simulation starts. If starting inventory is not provided, simulation runs based on the assumption that the starting inventory can fulfill all the demand in period 1 and leaves exactly the same inventory as reorder point (which equals safety stock + cycle stock in period 1) by the end of period 1.

**Site**

*Type: Text*

Identifies a site. Each site here must match a valid site from the Sites table.

**Product**

*Type: Text*

Identifies a product. Each product here must match a valid product from the Products table.

**Starting Inventory**

*Type: Numeric*

Initial inventory for each (Site, Product) combination.

**Sample Input**

Site	Product	Starting Inventory
Site1	Product1	3000
Site1	Product2	0
Site2	Product3	100

## 4.8.3 Solution

### Scenario Summary

This table shows metrics to evaluate overall simulation performance of the current scenario.

#### Metric

*Type: Text*

Name of metrics used to evaluate overall simulation performance.

#### Value

*Type: Numeric*

Metric values to evaluate simulation performance.

#### Metric explanation

**Number of Items:** Total number of items in simulation.

**Number of Items Meeting Target:** Number of items with simulation fill rate higher than or equal to each item's target fill rate.

**Number of Items Meeting Baseline:** Number of items with simulation fill rate higher than or equal to baseline fill rate specified in Parameters table.

## Site Product Details

This table contains detailed simulation output and shows how the inventory level changes by period for each item.

### Site

*Type: Text*

Identifies a site.

### Product

*Type: Text*

Identifies a product.

### Time Period

*Type: Numeric*

Each number identifies a period.

### Warm-up

*Type: Text*

Indicates whether the current period is within warm-up period and excluded from performance evaluation.

### Demand

*Type: Numeric*

Item demand of current period.

### Forecast

*Type: Numeric*

Forecast of item demand at current period.

### Safety Stock

*Type: Numeric*

Safety stock in units calculated at current period using future forecast.

### Cycle Stock

*Type: Numeric*

Cycle stock in units calculated at current period using future forecast.

### On Hand

*Type: Numeric*

On hand inventory at the end of current period after fulfilling demand and receiving orders.

### Reorder Point

*Type: Numeric*

Calculated by summing up safety stock and cycle stock at current period. If on hand inventory (plus order in transit) falls below reorder point, an order is placed to increase inventory level to reorder point.

### Order Placed

*Type: Numeric*

Size of order placed at current period to reach reorder point.

### Order in Transit

*Type: Numeric*

Sum of orders placed in previous periods that hasn't arrived.

**Order Arrived***Type: Numeric*

Order that arrives at the end of current period. It is assumed that these orders cannot be used to fulfilled demand at current period.

**Demand Fulfilled***Type: Numeric*

Demand fulfilled on time from inventory at current period.

**All Demand Fulfilled***Type: Text*

Indicates whether all demand has been successfully fulfilled from inventory.

**Below SS***Type: Text*

Indicates whether current on hand inventory falls below calculated safety stock level.

## Site Product Summary

This table contains summary data for each item's simulation.

### Site

*Type: Text*

Identifies a site.

### Product

*Type: Text*

Identifies a product.

### Safety Stock Days

*Type: Numeric*

Days of supply held as safety stock.

### Average Demand

*Type: Numeric*

Average demand during a period.

### COV

*Type: Numeric*

Standard deviation of forecast error divided by average demand. Measures forecast accuracy. Lower the COV, more accurate the forecast is.

### Lead Time Days

*Type: Numeric*

Lead time for each item converted to days.

### Target Fill Rate

*Type: Numeric*

Target percentage of demand fulfilled on time from inventory.

### Total Demand

*Type: Numeric*

Sum of demand throughout the simulated periods (excluding warm-up periods).

### Total Demand Fulfilled

*Type: Numeric*

Sum of all fulfilled demand throughout the simulated periods (excluding warm-up periods).

### Simulation Fill Rate

*Type: Numeric*

Percentage of demand fulfilled on time from inventory throughout the simulated periods (excluding warm-up periods).

### Simulation Service Level

*Type: Numeric*

Percentage of periods without stock-outs throughout the simulated periods (excluding warm-up periods).

## 5. Release Notes

---

### 5.1 Foresta Release Notes Summary

---

#### 1.1.8

Feature/Bug	Role Impacted
Define access settings for the application	AppBuilder, Administrator
Rename and reorder applications within a project	Analyst, Modeler, AppBuilder, Administrator
Add a new scenario from the workflow screen	Analyst, Modeler, AppBuilder, Administrator
Define Join between tables for bulk data operations	Analyst, Modeler, AppBuilder, Administrator
Preview and Execute without saving configuration for bulk data operations	Analyst, Modeler, AppBuilder, Administrator
Define category and tool-tip for both automated and manually deployed visualizations	Analyst, Modeler, AppBuilder, Administrator
Support for Tableau Live connection reports for both automated and manual deployment	Analyst, Modeler, AppBuilder, Administrator
Scenarios parameters table redesign defined by AppBuilder	All
Tree visualizations (for e.g Bill of Materials and Landed Cost) defined by AppBuilder	All
Tableau report adjustment on zoom-in/out and resolution change	All
Link to submit support issues	All

#### 1.1.7

Feature/Bug	Role Impacted
Bulk Data Operations UI enhancements	Analyst, Modeler, AppBuilder, Administrator
App-to-App-Overwrite actions implementation	Analyst, Modeler, AppBuilder, Administrator
Duplicate name validation for project, scenario and global app name	Analyst, Modeler, AppBuilder, Administrator
Option to set up a default visualization to be opened after solve is completed	Analyst, Modeler, AppBuilder, Administrator
Visualization refresh redesign	Analyst, Modeler, AppBuilder, Administrator
Bug fix for server memory usage metrics	AppBuilder, Administrator
Custom number and currency formatting defined by AppBuilder	AppBuilder
Pagination and advanced filtering implementation for scenario tables	All
Anaconda and Postgres upgrade to 2023 version	All

**1.1.6**

<b>Feature/Bug</b>	<b>Role Impacted</b>
Bulk Data Operations implementation (enhancement to existing bulk copy)	Analyst, Modeler, AppBuilder, Administrator
Workflow implementation	Analyst, Modeler, AppBuilder, Administrator
Added manual visualization refresh button	Analyst, Modeler, AppBuilder, Administrator
Visualizations cleanup on app deletion	Analyst, Modeler, AppBuilder, Administrator
Bug fix for delete labels not working from edit scenario pop-up	Analyst, Modeler, AppBuilder, Administrator
Bug fix for intermittent tableau refresh failure	Analyst, Modeler, AppBuilder, Administrator
Custom documentation preferences	AppBuilder, Administrator
Scenario sorting options update	All
Updated tool-tip for scenario	All
Bug fix for visible columns functionality not working	All
DB configuration update for performance improvements	All

**1.1.5**

<b>Feature/Bug</b>	<b>Role Impacted</b>
Ability to upload a new powerBi report for an application in single/multiple projects	Analyst, Modeler, AppBuilder, Administrator
Automated Tableau publishing with app creation	Analyst, Modeler, AppBuilder, Administrator
Support for multiple scenario comparison report	Analyst, Modeler, AppBuilder, Administrator
Customizable PBIX/Tableau configuration added in the visualization Org setup	AppBuilder, Administrator
Ability to upload custom gurobi.lic file	AppBuilder, Administrator
UI/UX update for app visualization settings	All
UI/UX update for scenario comparison modal pop-up	All
Fixed Tableau report overflow on escaping full screen	All
Fixed all pop ups to clear data on save/cancel	All
Fixed dspotconnect forbidden error issue	All
Fixed google-sheets error for tables with greater than 26 columns	All

**1.1.4**

<b>Feature/Bug</b>	<b>Role Impacted</b>
UI upgraded to darker color with new icons	All
Project and App search feature extended to search by createdBy	All
Admin menu remake	All
Scenario Table UI update	All
Scenarios can now be opened in new tab	All
Manual Tableau integration with automated scenario filter and refresh on solve	Analyst, Modeler, AppBuilder, Administrator
Overwrite action implementation	Analyst, Modeler, AppBuilder, Administrator
PBIX visualization refresh on duplicate-scenario and app-to-app	Analyst, Modeler, AppBuilder, Administrator
App Builder permissions extended to include visualization access management	AppBuilder

**1.1.3**

<b>Feature/Bug</b>	<b>Role Impacted</b>
Smart breadcrumbs with ability to switch between scenarios without navigating back	All
Scenario details screen status bar adjust to long text	All
Project and App cards are responsive to screen width change	All
Ability to search for scenario tables	All
Row count for each scenario table included with the name	All
Table tooltips are now displayed	All
Scenario duplicate actions implementation	Analyst, Modeler, AppBuilder, Administrator
Warning message on Solve and Validation	Analyst, Modeler, AppBuilder, Administrator
Bug fix for duplicate scenario not working	Analyst, Modeler, AppBuilder, Administrator
Bug fix for stop and kill button enabled even after being pressed	Analyst, Modeler, AppBuilder, Administrator

**1.1.2**

This is the first major release for the Foresta application with the following features:

<b>Feature/Bug</b>	<b>Role Impacted</b>
SSO Integration with major IDP providers	All
New feature rich UI	All
Project classification for organizing applications	Modeler, AppBuilder, Administrator
Automated PowerBI visualizations deployment	Analyst, Modeler, AppBuilder, Administrator
App Customization with custom python packages and whl files	AppBuilder, Administrator