Applied Notional Project

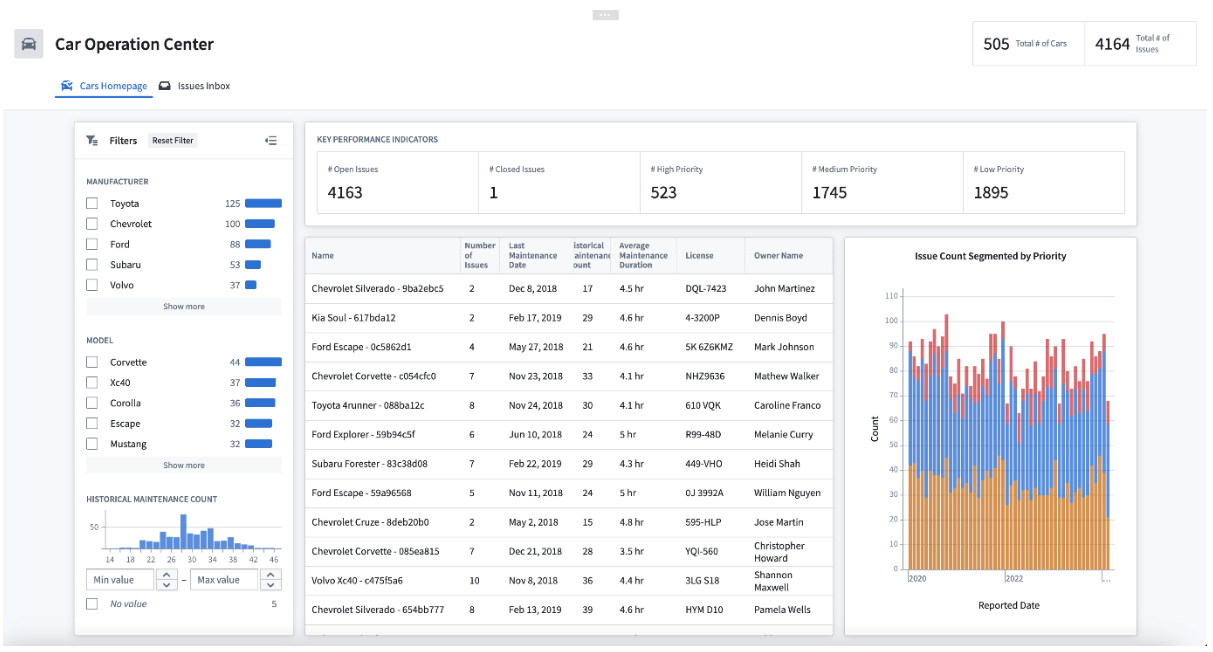
In this exercise, you will take on the role of a Foundry engineer building a dashboard for a used car leasing company.

This exercise designed to mimic a real-world scenario, so you won't be given step by step instructions on how to build the workflow. Instead, you are only given raw datasets, along with requirements for what the application should do.

The application requirements come from the perspective of car leasing manager who needs help tracking his cars and their related issues.

This exercise is designed to challenge you so that you come out with all the skills needed to develop on your own in the real world!

By the end of the exercise, you will have built an entire working application that allows a company to view their automobile fleet, inspect their cars, and resolve issues.



KEY PERFORMANCE INDICATORS

# Open Issues	# Closed Issues	# High Priority	# Medium Priority	# Low Priority
4163	1	523	1745	1895

Name	Number of Issues	Last Maintenance Date	istorical aintenanc ount	Average Maintenance Duration	License	Owner Name
Chevrolet Silverado - 9ba2ebc5	2	Dec 8, 2018	17	4.5 hr	DQL-7423	John Martinez
Kia Soul - 617bda12	2	Feb 17, 2019	29	4.6 hr	4-3200P	Dennis Boyd
Ford Escape - 0c5862d1	4	May 27, 2018	21	4.6 hr	5K 6Z6KMZ	Mark Johnson
Chevrolet Corvette - c054cfc0	7	Nov 23, 2018	33	4.1 hr	NHZ9636	Mathew Walker
Toyota 4runner - 088ba12c	8	Nov 24, 2018	30	4.1 hr	610 VQK	Caroline Franco
Ford Explorer - 59b94c5f	6	Jun 10, 2018	24	5 hr	R99-48D	Melanie Curry
Subaru Forester - 83c38d08	7	Feb 22, 2019	29	4.3 hr	449-VHO	Heidi Shah
Ford Escape - 59a96568	5	Nov 11, 2018	24	5 hr	0J 3992A	William Nguyen
Chevrolet Cruze - 8deb20b0	2	May 2, 2018	15	4.8 hr	595-HLP	Jose Martin
Chevrolet Corvette - 085ea815	7	Dec 21, 2018	28	3.5 hr	YQI-560	Christopher Howard
Volvo Xc40 - c475f5a6	10	Nov 8, 2018	36	4.4 hr	3LG S18	Shannon Maxwell
Chevrolet Silverado - 654bb777	8	Feb 13, 2019	39	4.6 hr	HYM D10	Pamela Wells

Benefits

This exercise is a great way to learn Foundry development. Since this exercise comes from the perspective of an end user, you will gain real-world skills on how to decompose and solve a problem. This means that you will learn how to critically think about application requirements and convert them into development tasks.

- This will translate to real world use case scoping.
- This will also give you the skills you need to think about Foundry not as a single-purpose application, but as a set of tools designed to work together and be used on top of each other.

Most importantly, you will gain the both the Foundry intuition and Foundry development skills for how to build a use case.

- This means that you will learn how to architect and develop in the platform without instructions or guidelines. Like all types of engineering, building something new entails reading documentation, debugging issues, and sometimes trial and error. By the end of this exercise, you will be comfortable building in Foundry as an independent, self sufficient developer.
- While developing in Foundry might be hard at first, being able to "just figure things out" is the quickest way to improve your skills and build even the most advanced use cases.

Project Objectives and Requirements

Project Summary

I own a car leasing company and I am looking to make an application to support my business. Recently, my customers have been reporting lots of issues with their cars. I need a system to help me keep track of this! I would like you to create an application in Foundry for me and my employees to use.

In the first tab, I would like a high level dashboard of my cars. I have provided you with raw data that should have everything you need to build this application.

In the second tab, I would like a list of outstanding issues on my cars. In addition to viewing details about existing issues, I want my employees to be able to update the priority of an issue, and mark an issue as closed.

Keep in mind, the raw data is a bit messy, so you might have to clean it up before it can be application ready!

Tips Before Getting Started

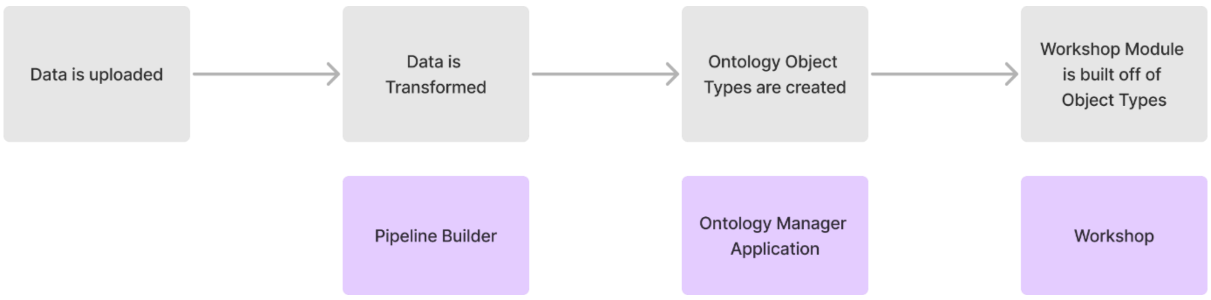
Before diving into the project, it’s important to have a general plan for how to build a use case, as well as to know what resources you have at your disposal. This page outlines how to think about Foundry development, and how to find and read documentation regarding the product. It also gives instructions on how to use Foundry’s custom AI integrations to help you find documentation and create pipeline builder transformations.

Foundry Development Paradigm

Foundry use case generally starts with data being uploaded or ingested. From there, data must be cleaned and transformed. Once the data is application-ready, it needs to be registered into the Ontology. From there, a front end application can be created.

This project is no different. The only Foundry applications that you need to use in order to complete this project are Pipeline Builder, Ontology Manger Application, and Workshop.

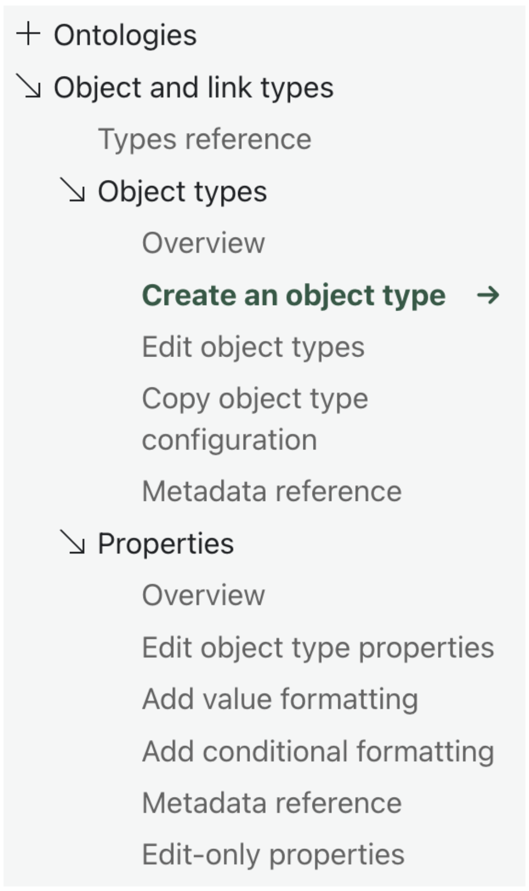
Here is a diagram detailing the Foundry development paradigm, along with the applications used at each stage.


Data is uploaded
Data is Transformed
Ontology Object Types are created
Workshop Module is built off of Object Types
Pipeline Builder
Ontology Manager Application
Workshop

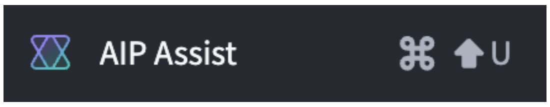
An important note is that the process is iterative. This means that as you are building your application, you may realize you created a column incorrectly or you are missing a column that you need in your application. That is okay. It is very normal to return to your Pipeline Builder and make a pipeline modification after your dataset has already been built.

Foundry Documentation

- All Foundry documentation can be accessed at https://www.palantir.com/docs/
- Building in Foundry means knowing how to navigate the documentation. Try searching for a key word or phrase, or browsing through the links presented on the left. For example, in order to learn how to create an object in the ontology, locate the "Ontologies" section of the documentation, and click on "Create an object type". You can also find the link here.
- This will come in handy when exploring new topics in Foundry, such as aggregating data or creating conditional formatting.

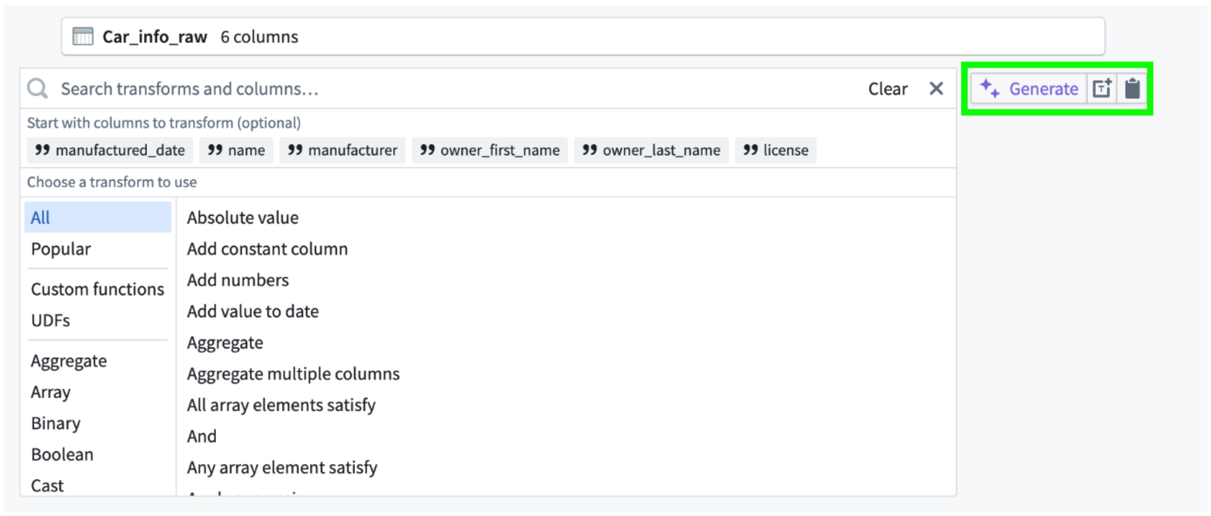


AIP Assist

- If your stack has enabled AIP Assist, you should see a button in the bottom left corner of your foundry navigation bar. The button looks like this :
    - 

- AIP Assist is trained on the foundry documentation and can be a great resource for answering questions when you don't know where in the foundry documentation to look.
- Some tools have helpful AI assistants built in. For example, when creating a transform in Pipeline Builder, you'll notice an option called "generate".

Pipeline Builder AIP Assist



- By typing in a description of what type of data transformation you are looking for, AIP can generate a transformation with filled in values.
- To make the most of the "generate" tool, try being as specific as possible with your prompt.
    - For example: "I want to update the data in column XYZ to be title cased instead of fully upper cased."
    - Another example: "I have a string formatted like this "2024-10-20", in column XYZ and I would like to convert it to a date."
- Note: Don't phrase prompts as a question, and try not to be vague. Avoid things like "How do I make a transformation that does XYZ".

Raw Data

In order to build my application, I will give you raw datasets from my source system. I have four datasets that should be pertinent for this project.

1. The first dataset is a cars dataset called *Car info raw*. This dataset contains basic information about each of the cars.
2. The second dataset is a maintenance history dataset called *Historical maintenance log raw*. This dataset contains historical information on maintenance performed on the cars. I don't need row level information from this dataset, I instead want aggregated information from this dataset, such as number of maintenance events per car.
3. The third dataset is called *Model information raw*. This dataset provides additional information on a car based on the manufacturer and model of that car.
4. The fourth and final dataset is called *Issues raw*. This dataset is a running list of active issues on the cars.

[Suggestion] Before starting implementation on the workflow, its important to inspect and understand these datasets. Click on each of the datasets and look at the different columns. Make sure you understand what data each dataset contains. That way you will be best equipped to clean and transform them. Some questions to ask yourself when inspecting the data include the following:

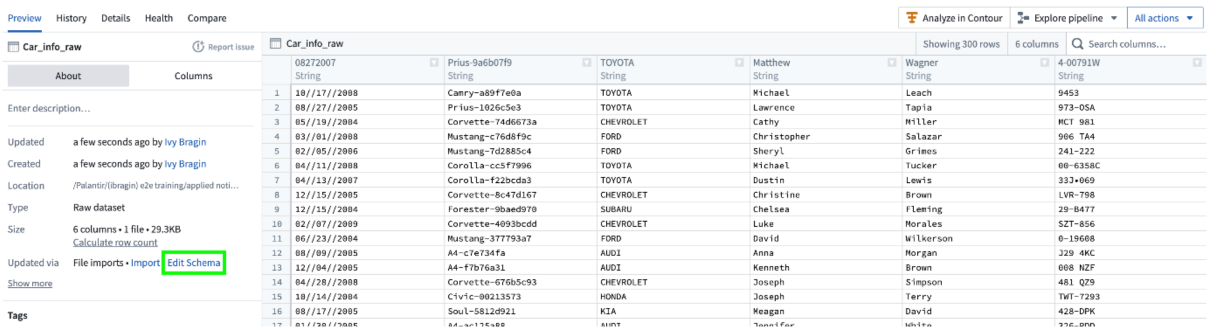
- Is there a unique identifier in this data? To view the uniqueness of a column of type "string", click the column name once you've loaded it into Foundry and view the presented statistics below. A histogram will appear and it will show you the frequency that different values appear at.
- What are the relationships between the datasets? See if you can identify any foreign keys. A foreign key allows you to link one dataset to another.
- Are there any data quality issues? Perhaps the column name or content is formatted poorly. It's important to clean any poorly formatted data.

Before beginning your work, it's encouraged to read through *all* of the requirements presented in the exercise. It is also recommended that you read through general hints, and save the specific hints section for when you get stuck. This will ensure your understanding of what is being asked of you, and may save you time down the road.

Download the files below to begin!

*Note: It's possible that some of the data may be interpreted incorrectly when you upload to Foundry. If you notice your column headers look strange, follow the steps below.

- Open the dataset in Foundry. On the left side of the dataset, you should see metadata about your upload. Select "Edit schema".



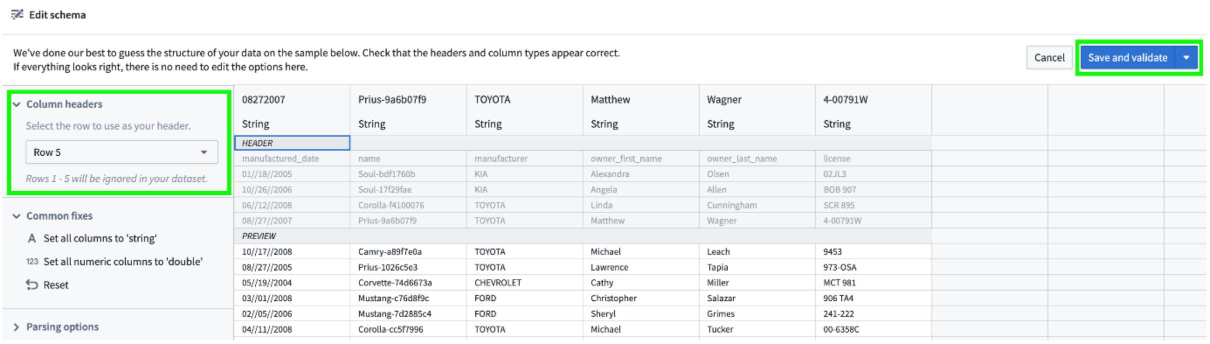
Enter description…

Updated	a few seconds ago by Ivy Bragin
Created	a few seconds ago by Ivy Bragin
Location	/Palantir/(ibragin) e2e training/applied noti…
Type	Raw dataset
Size	6 columns • 1 file • 29.3KB Calculate row count
Updated via	File imports • Import Edit Schema

Car_info_raw — Showing 300 rows — 6 columns — Search columns…

	08272007 String	Prius-9a6b07f9 String	TOYOTA String	Matthew String	Wagner String	4-00791W String
1	10//17//2008	Camry-a89f7e0a	TOYOTA	Michael	Leach	9453
2	08//27//2005	Prius-1026c5e3	TOYOTA	Lawrence	Tapia	973-OSA
3	05//19//2004	Corvette-74d6673a	CHEVROLET	Cathy	Miller	MCT 981
4	03//01//2008	Mustang-c76d8f9c	FORD	Christopher	Salazar	906 TA4
5	02//05//2006	Mustang-7d2885c4	FORD	Sheryl	Grimes	241-222
6	04//11//2008	Corolla-cc5f7996	TOYOTA	Michael	Tucker	00-6358C
7	04//13//2007	Corolla-f22bcda3	TOYOTA	Dustin	Lewis	33J•069
8	12//15//2005	Corvette-8c47d167	CHEVROLET	Christine	Brown	LVR-798
9	12//15//2004	Forester-9baed970	SUBARU	Chelsea	Fleming	29-B477
10	02//07//2009	Corvette-4093bcdd	CHEVROLET	Luke	Morales	SZT-856
11	06//23//2004	Mustang-377793a7	FORD	David	Wilkerson	0-19608
12	08//09//2005	A4-c7e734fa	AUDI	Anna	Morgan	J29 4KC
13	12//04//2005	A4-f7b76a31	AUDI	Kenneth	Brown	008 NZF
14	04//28//2008	Corvette-676b5c93	CHEVROLET	Joseph	Simpson	481 QZ9
15	10//14//2004	Civic-00213573	HONDA	Joseph	Terry	TWT-7293
16	08//17//2005	Soul-5812d921	KIA	Meagan	David	428-DPK

- Notice that Foundry auto-selected row 5 to be your header. Change the value under "column header'" to say Row 1. Then click save and validate.



We've done our best to guess the structure of your data on the sample below. Check that the headers and column types appear correct.
If everything looks right, there is no need to edit the options here.

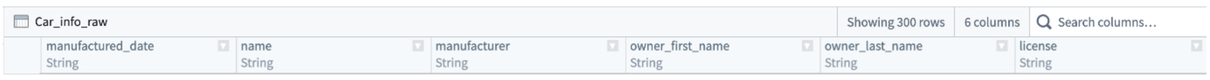
Select the row to use as your header.

Row 5

Rows 1 - 5 will be ignored in your dataset.

08272007 String	Prius-9a6b07f9 String	TOYOTA String	Matthew String	Wagner String	4-00791W String
HEADER					
manufactured_date	name	manufacturer	owner_first_name	owner_last_name	license
01//18//2005	Soul-bdf1760b	KIA	Alexandra	Olsen	02JL3
10//26//2006	Soul-17f29fae	KIA	Angela	Allen	BOB 907
06//12//2008	Corolla-f4100076	TOYOTA	Linda	Cunningham	SCR 895
08//27//2007	Prius-9a6b07f9	TOYOTA	Matthew	Wagner	4-00791W
PREVIEW					
10//17//2008	Camry-a89f7e0a	TOYOTA	Michael	Leach	9453
08//27//2005	Prius-1026c5e3	TOYOTA	Lawrence	Tapia	973-OSA
05//19//2004	Corvette-74d6673a	CHEVROLET	Cathy	Miller	MCT 981
03//01//2008	Mustang-c76d8f9c	FORD	Christopher	Salazar	906 TA4
02//05//2006	Mustang-7d2885c4	FORD	Sheryl	Grimes	241-222
04//11//2008	Corolla-cc5f7996	TOYOTA	Michael	Tucker	00-6358C

- Once your headers look like they have appropriate names, you may continue.


Car_info_raw — Showing 300 rows — 6 columns — Search columns…

manufactured_date String	name String	manufacturer String	owner_first_name String	owner_last_name String	license String

Downloads

- Car info raw.csv
- Model information raw.csv
- Issues raw.csv
- Historical maintenance log raw.csv

Objective 1 - Data Cleaning

Before you can create my application, you will need to clean the raw data provided. My datasets are poorly maintained and I am already aware of some outstanding issues with each table. In addition to solving the issues below, feel free to make any other data transformations necessary to fulfill my application requirements.

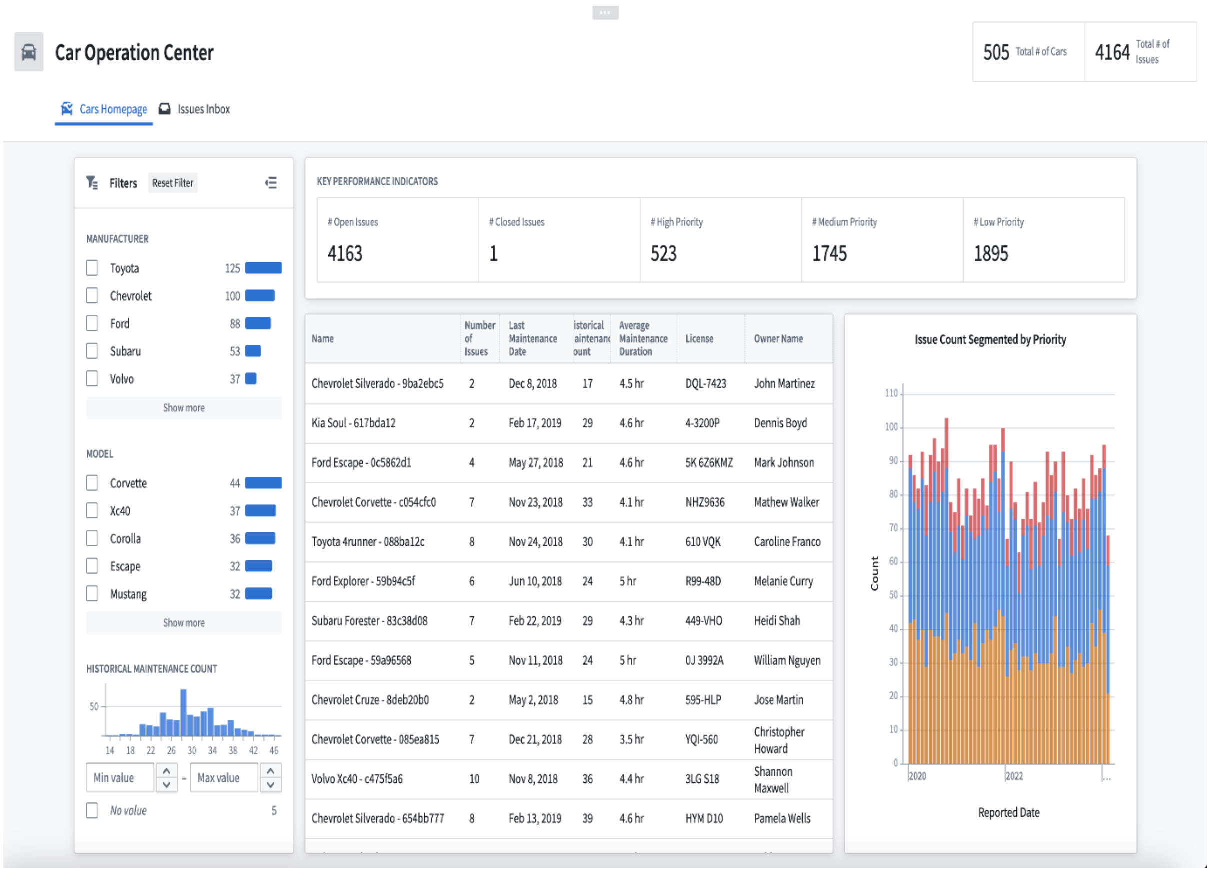
Some known issues with my raw data are:

- **Expired issues**. My issues dataset has tons of expired issues. An expired issue is any issue before January 1st, 2020. I would like those rows to be filtered out, since these issues are no longer valid.
- **Date formatting problems**. My source system has messed with some of my date columns. Instead of dates being saved as date fields, they are saved as string fields and formatted in weird ways. For example, the "reported_date" field in the issue's table looks like "2021_10_22" and the "manufactured_date" field in the car's table looks like "05//19//2004".
- **Capitalization issues**. Some of my columns have data that is accidentally capitalized when it shouldn't be. The data in these columns should be title-cased or sentence-cased instead.
- **Abbreviations**. The "transmission" column in the model information table holds abbreviated values ("A", "M"). "A" should be switched to "Automatic", and M should be switched to "Manual".
- **Foreign key issues**. Lots of my datasets should be able to link together, but because of formatting mistakes, they don't. For example, the *car_id* field in the Issues table should be able to link to the *name* field in the Car table. The problem is, the *car_id* field and the *name* field have prefixes that prevent the column from being a valid foreign key.
- **Ghost issues**: Some issues in the issues dataset don’t actually link to any car. I would like those issues to be filtered out.

Objective 2 - Cars Homepage

Tab 1 - Cars Homepage

The cars homepage will provide me with an overview page of my cars, along with metrics and charts to view related issues on a high level. I have created a mockup of what I want my cars homepage to look like. Here it is.



Car Operation Center

505 Total # of Cars | 4164 Total # of Issues

Cars Homepage | Issues Inbox

Filters | Reset Filter

MANUFACTURER

Toyota	125
Chevrolet	100
Ford	88
Subaru	53
Volvo	37

Show more

MODEL

Corvette	44
Xc40	37
Corolla	36
Escape	32
Mustang	32

Show more

HISTORICAL MAINTENANCE COUNT

Min value – Max value

No value 5

KEY PERFORMANCE INDICATORS

# Open Issues	# Closed Issues	# High Priority	# Medium Priority	# Low Priority
4163	1	523	1745	1895

Name	Number of Issues	Last Maintenance Date	istorical aintenanc ount	Average Maintenance Duration	License	Owner Name
Chevrolet Silverado - 9ba2ebc5	2	Dec 8, 2018	17	4.5 hr	DQL-7423	John Martinez
Kia Soul - 617bda12	2	Feb 17, 2019	29	4.6 hr	4-3200P	Dennis Boyd
Ford Escape - 0c5862d1	4	May 27, 2018	21	4.6 hr	5K 6Z6KMZ	Mark Johnson
Chevrolet Corvette - c054cfc0	7	Nov 23, 2018	33	4.1 hr	NHZ9636	Mathew Walker
Toyota 4runner - 088ba12c	8	Nov 24, 2018	30	4.1 hr	610 VQK	Caroline Franco
Ford Explorer - 59b94c5f	6	Jun 10, 2018	24	5 hr	R99-48D	Melanie Curry
Subaru Forester - 83c38d08	7	Feb 22, 2019	29	4.3 hr	449-VHO	Heidi Shah
Ford Escape - 59a96568	5	Nov 11, 2018	24	5 hr	0J 3992A	William Nguyen
Chevrolet Cruze - 8deb20b0	2	May 2, 2018	15	4.8 hr	595-HLP	Jose Martin
Chevrolet Corvette - 085ea815	7	Dec 21, 2018	28	3.5 hr	YQI-560	Christopher Howard
Volvo Xc40 - c475f5a6	10	Nov 8, 2018	36	4.4 hr	3LG S18	Shannon Maxwell
Chevrolet Silverado - 654bb777	8	Feb 13, 2019	39	4.6 hr	HYM D10	Pamela Wells

Lets dive into the “Cars Homepage“ tab requirements and features.

Application Section Configuration

- I want all of the widgets in my Cars Homepage to be contained in a section with padding so that the widgets don’t touch each other.
- In the section header, I would like “Car Operation Center” in big letters. Underneath that I want tabs that allow users to switch between the Cars Homepage and the Issues Inbox.
- In the top right corner I want two metrics. The first metric should show the total number of cars. The next metric should show the total number of issues.

Filter bar (left hand side)

- I want to have a filter bar that allows me to filter my cars. I want the filter bar to control all of the metrics, table, and chart. This means that if I apply a filter in the filter bar, all the metrics, table, and chart should respond accordingly. In my filter bar, I want to be able to filter my cars data by “Manufacturer”, “Model”, and “Historical Maintenance Count”.
- I realize these fields aren’t in the cars table naturally, so you may need to join them in within the pipeline.
- For the dimensions of the filter bar, I want it to take up the full amount of available vertical space, and take up 300px horizontally.

Metrics (“Key Performance Indicators” at the top of the application)

- There are two groups of metrics in my mockup. In this section I am only talking about the metrics with the title “Key Performance Indicators”.
- I want all of these metrics to update based on the filter bar on the left.
- For the first metric (on the left) I want to show the total number of open issues for the filtered cars.
- For the second metric I want to show the total number of closed issues for the filtered cars.
- For the third fourth and fifth metric, I want to show the total number of open issues segmented by priority. That means the third metric should be total number of open issues that are high priority for the filtered cars. The fourth and fifth metric should display the same statistic for medium priority and low priority issues, respectively. I will give you guidance on how to determine the “priority“ of an issue in objective 3.

Table (underneath metrics to the right of the filter bar)

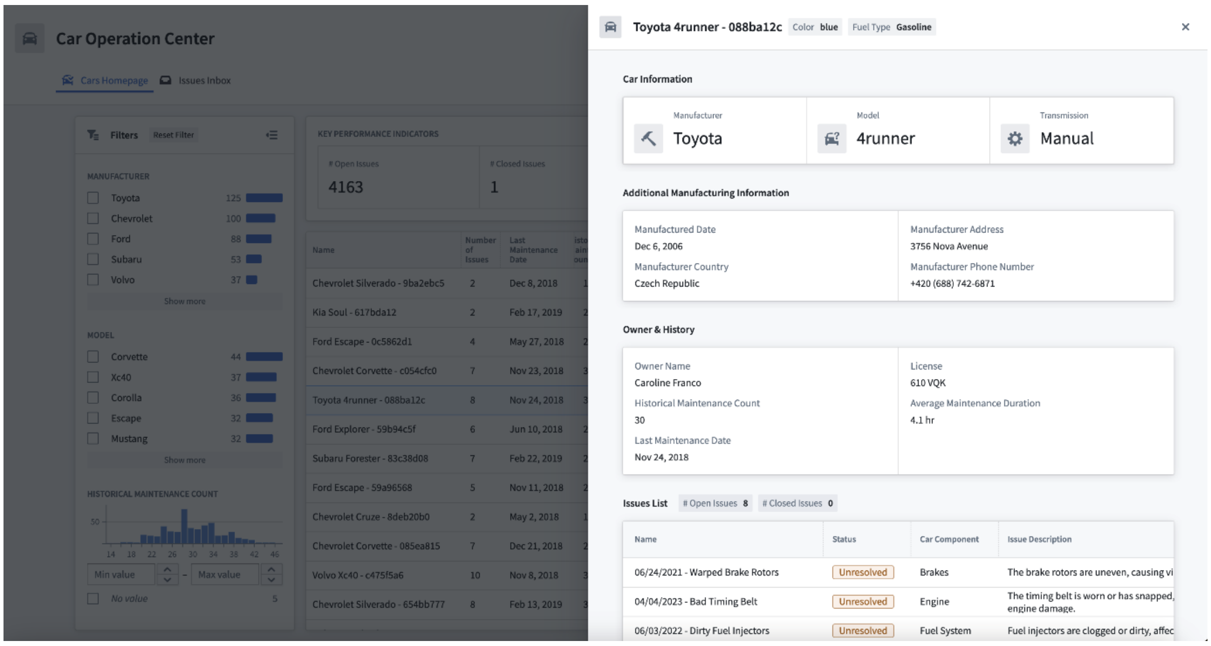
- This table should list all filtered cars and relevant properties. I will define the properties that I want below.
    - The starting column called “Name” should be a readable identifier for the car. I want the property to be ”<Manufacturer> <Model> - <Car ID>”.
    - Next, I want a column called “Number of Issues” that shows me the number of related issues to that row’s car.
    - After that, I want a column called “Last Maintenance Date” that gives me the most recent maintenance date for that car. This information can be calculated from the historical maintenance log table.
    - Next, I want a “Historical Maintenance Count” column. This column should show a count of maintenance events per car. This information can be calculated from the historical maintenance log table.
    - I also want a “Average Maintenance Duration” column. This information can be calculated from the historical maintenance log table.
    - Next, I want to see the license plate number of the car in a column called “License”.
    - Finally, I want a column called “Owner Name“. The Owner Name should come from the “owner_first_name“ and the “owner_last_name“ fields.
- When a user clicks a row, I want an overlay to appear with more information about the selected row. You can find more information on the overlay in the below section “Overlay - Selected Car View“.

Chart (bottom right)

- The chart should show issue count segmented by priority. The y axis should be a count of issues, and the x axis should represent time.

Overlay - Selected Car View

When I select a car from the table, I want an overlay to pop up with additional information about the selected car.



MANUFACTURER	
Toyota	125
Chevrolet	100
Ford	88
Subaru	53
Volvo	37

MODEL	
Corvette	44
Xc40	37
Corolla	36
Escape	32
Mustang	32

Open Issues 4163

Closed Issues 1

Name	Number of Issues	Last Maintenance Date
Chevrolet Silverado - 9ba2ebc5	2	Dec 8, 2018
Kia Soul - 617bda12	2	Feb 17, 2019
Ford Escape - 0c5862d1	4	May 27, 2018
Chevrolet Corvette - c054cfc0	7	Nov 23, 2018
Toyota 4runner - 088ba12c	8	Nov 24, 2018
Ford Explorer - 59b94c5f	6	Jun 10, 2018
Subaru Forester - 83c38d08	7	Feb 22, 2019
Ford Escape - 59a96568	5	Nov 11, 2018
Chevrolet Cruze - 8deb20b0	2	May 2, 2018
Chevrolet Corvette - 085ea815	7	Dec 21, 2018
Volvo Xc40 - c475f5a6	10	Nov 8, 2018
Chevrolet Silverado - 654bb777	8	Feb 13, 2019

Toyota 4runner - 088ba12c Color blue Fuel Type Gasoline ×

Car Information

Manufacturer	Model	Transmission
Toyota	4runner	Manual

Additional Manufacturing Information

Manufactured Date: Dec 6, 2006
Manufacturer Address: 3756 Nova Avenue
Manufacturer Country: Czech Republic
Manufacturer Phone Number: +420 (688) 742-6871

Owner & History

Owner Name: Caroline Franco
License: 610 VQK
Historical Maintenance Count: 30
Average Maintenance Duration: 4.1 hr
Last Maintenance Date: Nov 24, 2018

Issues List # Open Issues 8 # Closed Issues 0

Name	Status	Car Component	Issue Description
06/24/2021 - Warped Brake Rotors	Unresolved	Brakes	The brake rotors are uneven, causing vi
04/04/2023 - Bad Timing Belt	Unresolved	Engine	The timing belt is worn or has snapped, engine damage.
06/03/2022 - Dirty Fuel Injectors	Unresolved	Fuel System	Fuel injectors are clogged or dirty, affec

Overlay Section Configuration

- For my selected car view section header, I want the car name to be displayed at the top. This should be the same name that is used as the title property for the table.
- Also in the header, I want two metrics. These metrics should be the car color and the car fuel type. On the right, I want a close button.
- In terms of spacing, I want the entire section to have padding so that no widgets touch each other.

Car Information (near the top)

- Below the section title at the top, I want a Car Information section.
- Within this section, I want three metrics. The first metric should show the manufacturer. The second metric should show the model. The third metric should show the transmission type for the car.

Additional Manufacturing Information

- In this section, I want information from the manufacturing side. This data should come from the car info dataset and the model information dataset.
- The property list should contain the following properties.
  - “Manufactured Date”, “Manufacturer Address”, “Manufacturer Country”, “Manufacturer Phone Number”.

Owner And History

- This section should provide more information on the owner and the history of the car. These fields will come from the car info dataset and the historical maintenance dataset.
- The property list should contain the following properties.
    - "Owner Name", "License", "Historical Maintenance Count", "Average Maintenance Duration", "Last Maintenance Date".

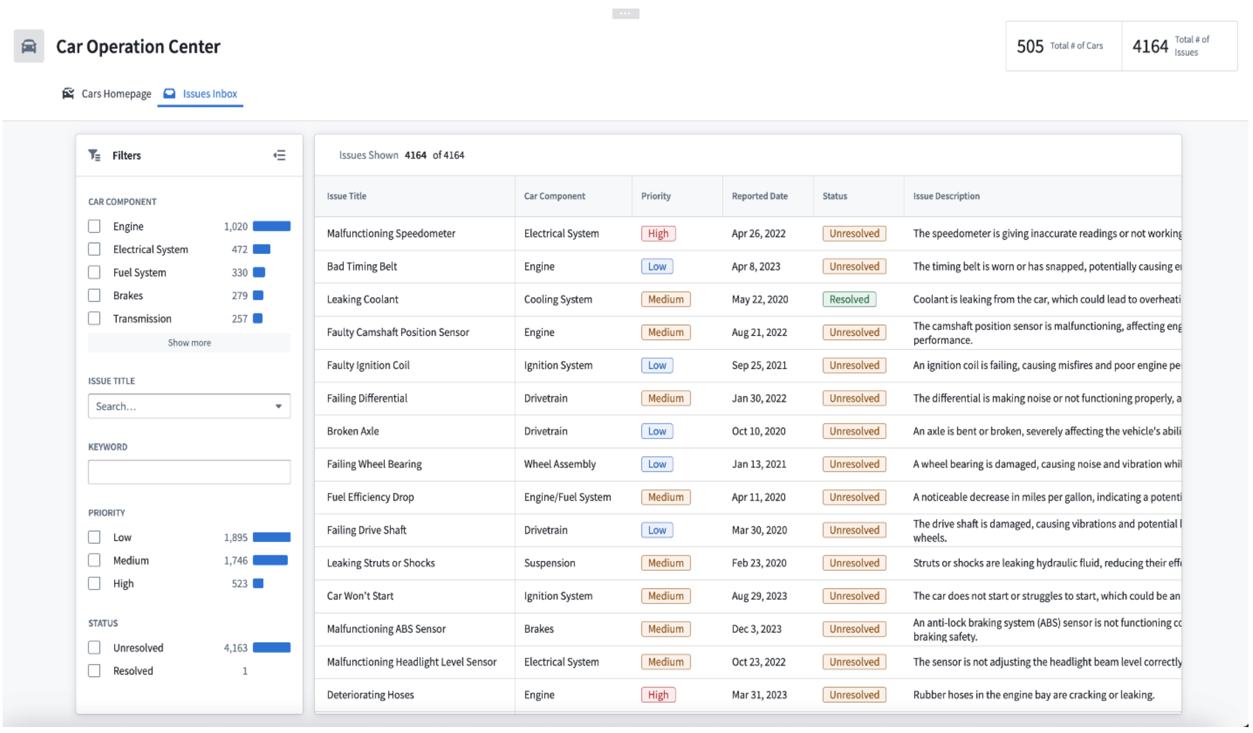
Issues List (bottom)

- In the Issues List header, you may notice two metrics. "# Open Issues" and "# Closed Issues". I would like these metrics to show the number of linked issues that are open and the number of linked issues that are closed. You can determine if a linked issue is open or closed using the "Status" field. I will explain the status field in the next objective "Objective 3 - Issue's Inbox".
- Finally I would like a table of all of the issues that are linked to this car.
- The table should contain the following properties
    - "Name", "Status", "Car Component", "Issue Description".
- A more detailed explanation of all of these columns can be found in "Objective 3 - Issue's Inbox".

Objective 3 - Issues Inbox

Tab 2 - Issues Inbox

The issues inbox will provide my employees with a filterable list of all issues with my cars. When an employee clicks an issue, I want a hidden section to appear with more information on that issue.



Car Operation Center

505 Total # of Cars | 4164 Total # of Issues

Cars Homepage | Issues Inbox

Filters

CAR COMPONENT

☐ Engine	1,020
☐ Electrical System	472
☐ Fuel System	330
☐ Brakes	279
☐ Transmission	257

Show more

ISSUE TITLE

Search...

KEYWORD

PRIORITY

☐ Low	1,895
☐ Medium	1,746
☐ High	523

STATUS

☐ Unresolved	4,163
☐ Resolved	1

Issues Shown **4164** of 4164

Issue Title	Car Component	Priority	Reported Date	Status	Issue Description
Malfunctioning Speedometer	Electrical System	High	Apr 26, 2022	Unresolved	The speedometer is giving inaccurate readings or not working
Bad Timing Belt	Engine	Low	Apr 8, 2023	Unresolved	The timing belt is worn or has snapped, potentially causing e
Leaking Coolant	Cooling System	Medium	May 22, 2020	Resolved	Coolant is leaking from the car, which could lead to overheati
Faulty Camshaft Position Sensor	Engine	Medium	Aug 21, 2022	Unresolved	The camshaft position sensor is malfunctioning, affecting eng performance.
Faulty Ignition Coil	Ignition System	Low	Sep 25, 2021	Unresolved	An ignition coil is failing, causing misfires and poor engine pe
Failing Differential	Drivetrain	Medium	Jan 30, 2022	Unresolved	The differential is making noise or not functioning properly, a
Broken Axle	Drivetrain	Low	Oct 10, 2020	Unresolved	An axle is bent or broken, severely affecting the vehicle's abili
Failing Wheel Bearing	Wheel Assembly	Low	Jan 13, 2021	Unresolved	A wheel bearing is damaged, causing noise and vibration whi
Fuel Efficiency Drop	Engine/Fuel System	Medium	Apr 11, 2020	Unresolved	A noticeable decrease in miles per gallon, indicating a potenti
Failing Drive Shaft	Drivetrain	Low	Mar 30, 2020	Unresolved	The drive shaft is damaged, causing vibrations and potential l wheels.
Leaking Struts or Shocks	Suspension	Medium	Feb 23, 2020	Unresolved	Struts or shocks are leaking hydraulic fluid, reducing their eff
Car Won't Start	Ignition System	Medium	Aug 29, 2023	Unresolved	The car does not start or struggles to start, which could be an
Malfunctioning ABS Sensor	Brakes	Medium	Dec 3, 2023	Unresolved	An anti-lock braking system (ABS) sensor is not functioning c braking safety.
Malfunctioning Headlight Level Sensor	Electrical System	Medium	Oct 23, 2022	Unresolved	The sensor is not adjusting the headlight beam level correctly
Deteriorating Hoses	Engine	High	Mar 31, 2023	Unresolved	Rubber hoses in the engine bay are cracking or leaking.

Application Section Configuration

- In the same way that the Cars Homepage had padding, the Issues Inbox section should have padding that makes it so the widgets never touch each other, the top of the screen, sides of the screen, or the bottom of the screen.

Filter Bar (left hand side)

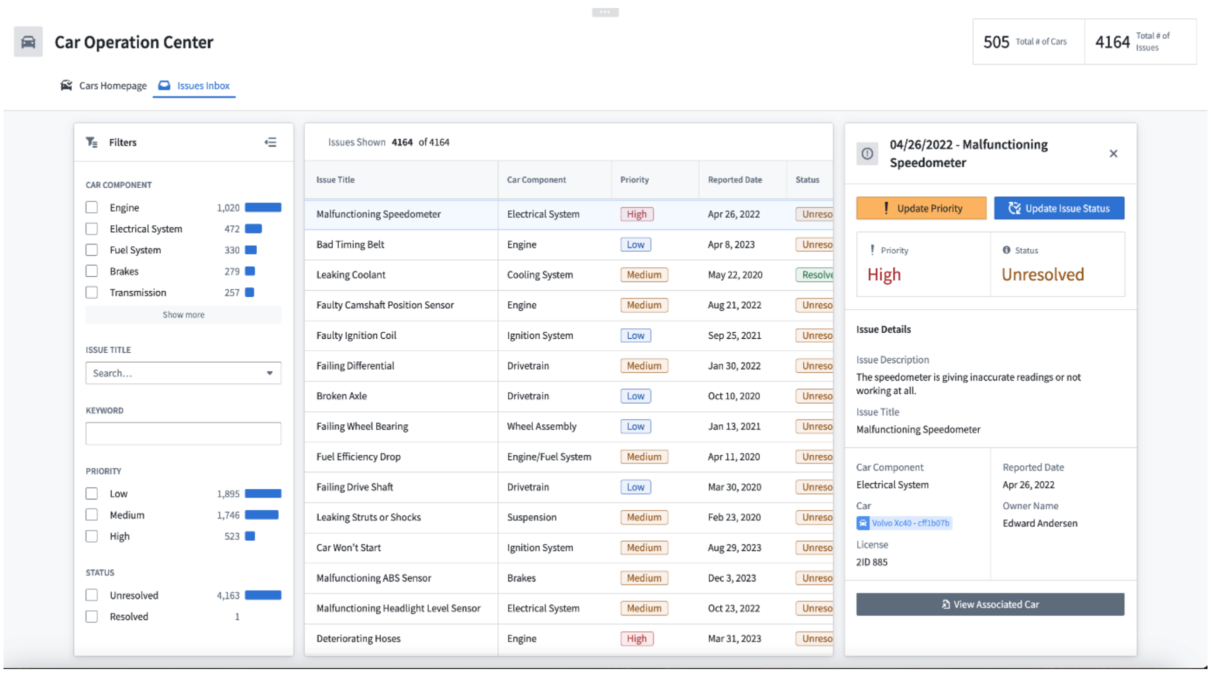
- I want to be able to filter my issues based on various properties.
- The first two properties are “car component“ and ”Issue Title“. These can be found in the issues table. Instead of having the issue title property be filterable as a histogram, I want the filter bar to show a multi select dropdown.
- Next I want to be able to search my issues in a search bar.
- Also, I want to be able to filter issues based on “Priority“. The ”Priority“ definition of an issue can be found below.
- Finally, I would like to be able to filter issues based on “Status”. The “Status“ definition can be found below.

Issue Table

- I would like the rows in the issue table to be dependent on my filter criteria.
- At the top of the table in the section header, I would like a metric that shows me how many issues I’m looking at, and how many possible issues there are.
- In my table, I would like the following columns.
    - “Issue Title“, ”Car Component“, and ”Issue Description“. These all come from the issues dataset.
    - “Reported Date“. This field comes from the issues dataset, but as I mentioned before, it needs to be formatted correctly.
    - “Priority“. I want the priority to be dependent on the manufacturer of the car that the issue is related to. That means if the manufacturer is ”Audi“ or ”Volvo“ then I want the priority to be ”High“. If the car is ”Ford“, ”Chevrolet“, or ”Jeep“ then I want the priority to be ”Medium. For every other manufacturer, I want the priority to be “Low”. The priority should be configurable via action. That is defined in the next section.
    - “Status”. I want a status column that is automatically set to Unresolved for all issues. The status should be configurable via action. That is defined in the next section.
- I want both the priority and the status to have conditional formatting. That means for “High“ priority issues, the value is red. For ”Medium“ priority issues, the value is orange, and for ”Low“ priority issues, the value is blue. For a status of ”Unresolved“ I want the value to be orange, and for a status of ”Resolved“ I want the value to be green.

Tab 2 - Selected Issue View

When I select an issue from the table, I want a "Selected Issue" section to appear on the right with more information about the selected issue.



Car Operation Center

505 Total # of Cars | 4164 Total # of Issues

Cars Homepage | Issues Inbox

Filters

CAR COMPONENT

Engine	1,020
Electrical System	472
Fuel System	330
Brakes	279
Transmission	257

Show more

ISSUE TITLE

Search...

KEYWORD

PRIORITY

Low	1,895
Medium	1,746
High	523

STATUS

Unresolved	4,163
Resolved	1

Issues Shown **4164** of 4164

Issue Title	Car Component	Priority	Reported Date	Status
Malfunctioning Speedometer	Electrical System	High	Apr 26, 2022	Unreso
Bad Timing Belt	Engine	Low	Apr 8, 2023	Unreso
Leaking Coolant	Cooling System	Medium	May 22, 2020	Resolve
Faulty Camshaft Position Sensor	Engine	Medium	Aug 21, 2022	Unreso
Faulty Ignition Coil	Ignition System	Low	Sep 25, 2021	Unreso
Failing Differential	Drivetrain	Medium	Jan 30, 2022	Unreso
Broken Axle	Drivetrain	Low	Oct 10, 2020	Unreso
Failing Wheel Bearing	Wheel Assembly	Low	Jan 13, 2021	Unreso
Fuel Efficiency Drop	Engine/Fuel System	Medium	Apr 11, 2020	Unreso
Failing Drive Shaft	Drivetrain	Low	Mar 30, 2020	Unreso
Leaking Struts or Shocks	Suspension	Medium	Feb 23, 2020	Unreso
Car Won't Start	Ignition System	Medium	Aug 29, 2023	Unreso
Malfunctioning ABS Sensor	Brakes	Medium	Dec 3, 2023	Unreso
Malfunctioning Headlight Level Sensor	Electrical System	Medium	Oct 23, 2022	Unreso
Deteriorating Hoses	Engine	High	Mar 31, 2023	Unreso

04/26/2022 - Malfunctioning Speedometer ×

Update Priority | Update Issue Status

Priority: High | Status: Unresolved

Issue Details

Issue Description
The speedometer is giving inaccurate readings or not working at all.

Issue Title
Malfunctioning Speedometer

Car Component
Electrical System

Reported Date
Apr 26, 2022

Car
Volvo Xc40 - cff1b07b

Owner Name
Edward Andersen

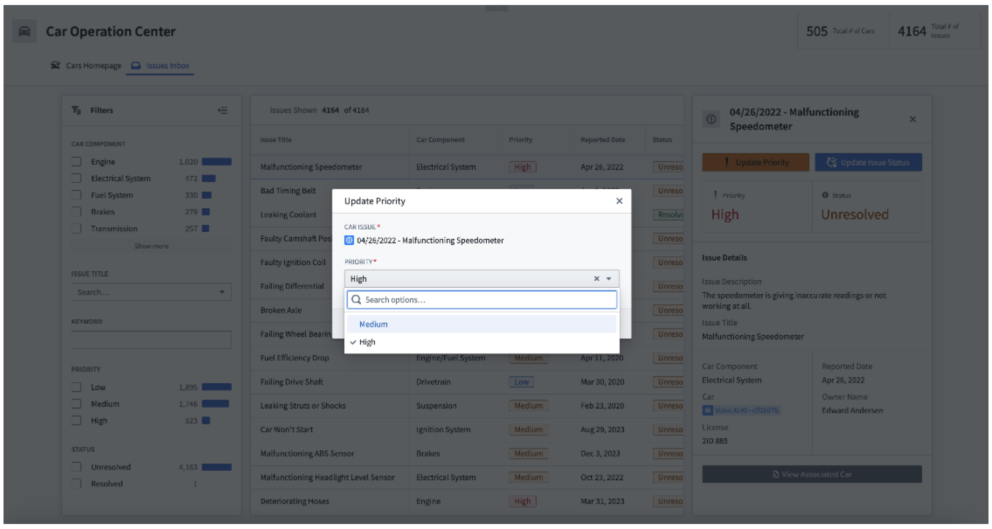
License
2ID 885

View Associated Car

Selected Issue Section

- The "Selected Issue" section should be hidden until I select an issue. That means it should be conditionally visible. When I select an issue from the table, this section should become visible.
- For the section header, I want to show the “Name“ property. The property should be a concatenation of the reported date and the issue title.
- Next to the “Name“ property, I want an X button. When clicked, the button should make the conditionally visible section hidden again.

Buttons - Update Priority and Update Issue Status



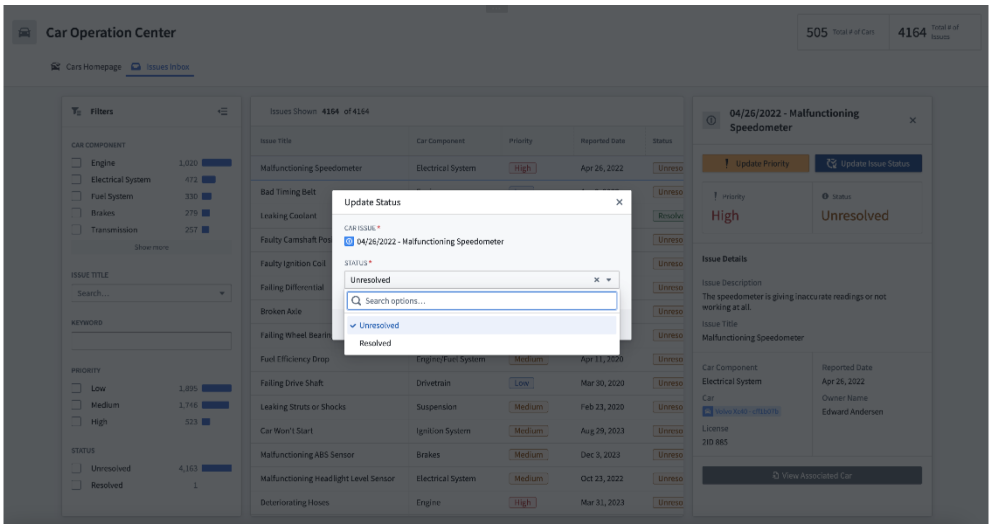
Update Priority ×

CAR ISSUE *
04/26/2022 - Malfunctioning Speedometer

PRIORITY *
High

Search options...

Medium
✓ High



CAR COMPONENT

Engine	1,020
Electrical System	472
Fuel System	330
Brakes	279
Transmission	257

ISSUE TITLE

KEYWORD

PRIORITY

Low	1,895
Medium	1,746
High	523

STATUS

Unresolved	4,163
Resolved	1

Issues Shown 4164 of 4164

Issue Title	Car Component	Priority	Reported Date	Status
Malfunctioning Speedometer	Electrical System	High	Apr 26, 2022	Unreso
Bad Timing Belt	[illegible]	[illegible]	[illegible]	Unreso
Leaking Coolant				Resolv
Faulty Camshaft Posi				Unreso
Faulty Ignition Coil				Unreso
Failing Differential				Unreso
Broken Axle				Unreso
Failing Wheel Bearin				Unreso
Fuel Efficiency Drop	Engine/Fuel System	Medium	Apr 11, 2020	Unreso
Failing Drive Shaft	Drivetrain	Low	Mar 30, 2020	Unreso
Leaking Struts or Shocks	Suspension	Medium	Feb 23, 2020	Unreso
Car Won't Start	Ignition System	Medium	Aug 29, 2023	Unreso
Malfunctioning ABS Sensor	Brakes	Medium	Dec 3, 2023	Unreso
Malfunctioning Headlight Level Sensor	Electrical System	Medium	Oct 23, 2022	Unreso
Deteriorating Hoses	Engine	High	Mar 31, 2023	Unreso

Update Status

CAR ISSUE *
04/26/2022 - Malfunctioning Speedometer

STATUS *
Unresolved

Search options...

✓ Unresolved
Resolved

04/26/2022 - Malfunctioning Speedometer

Priority: High
Status: Unresolved

Issue Details

Issue Description
The speedometer is giving inaccurate readings or not working at all.

Issue Title
Malfunctioning Speedometer

Car Component
Electrical System

Reported Date
Apr 26, 2022

Car
Volvo Xc40 - cff1b07b

Owner Name
Edward Andersen

License
2ID 885

- I would like two buttons. One button to let me change the priority of an issue and one button to let me change the status of an issue.
- when these buttons are clicked, I want an action popup to appear. The popup should let me change the priority or the status of the issue.
- The popup should automatically know the issue that I want to change and show it to me in a greyed out box, like in the mockup above.
- The popup should also provide a dropdown for changing the status or priority. The dropdown options for status should be “Unresolved” and “Resolved”. The options for Priority should be “High”, “Medium”, “Low”.

Issue Details

- Below the buttons, I would like an Issue Details section. In this section, I want to first see my issue title and description.
- Next, I want to be able to see additional properties such as “Car Component” and “Reported Date”.
- Finally, I would like to see properties from the car. These properties include "Name", "Owner Name", and "License".

Button - View Associated Car

- When this button is clicked, I want the car overlay to popup. I want the overlay to show the car related to the issue that I was just looking at.

Hints

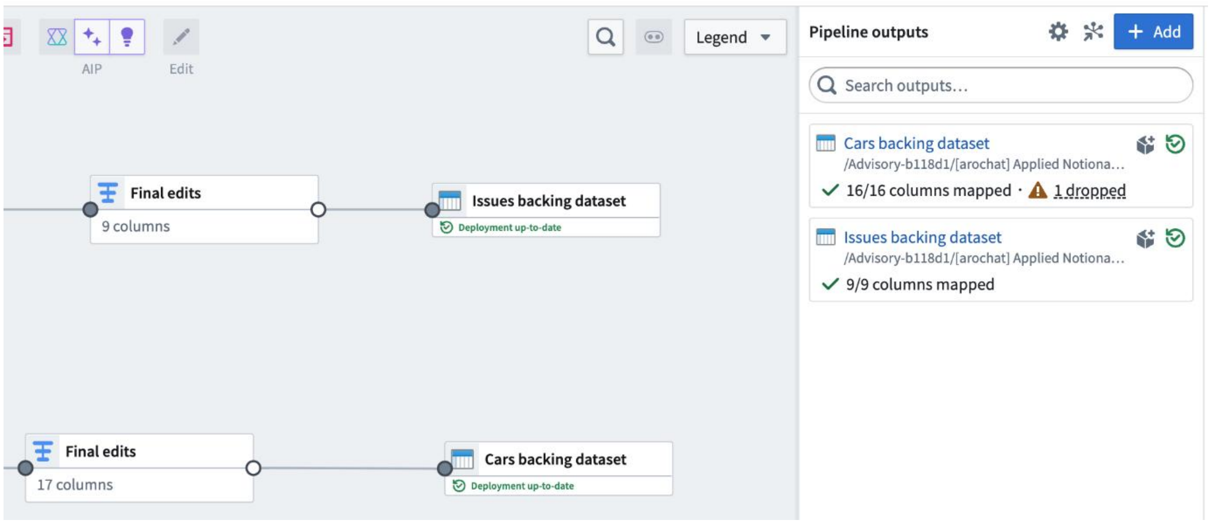
Hints Overview & General FAQ

Read Me!

The point of this learning module is to struggle and gain the skills to build without guidance or handholding. Only use these hints if you are stuck or if you feel like the exercise is taking longer then it should.

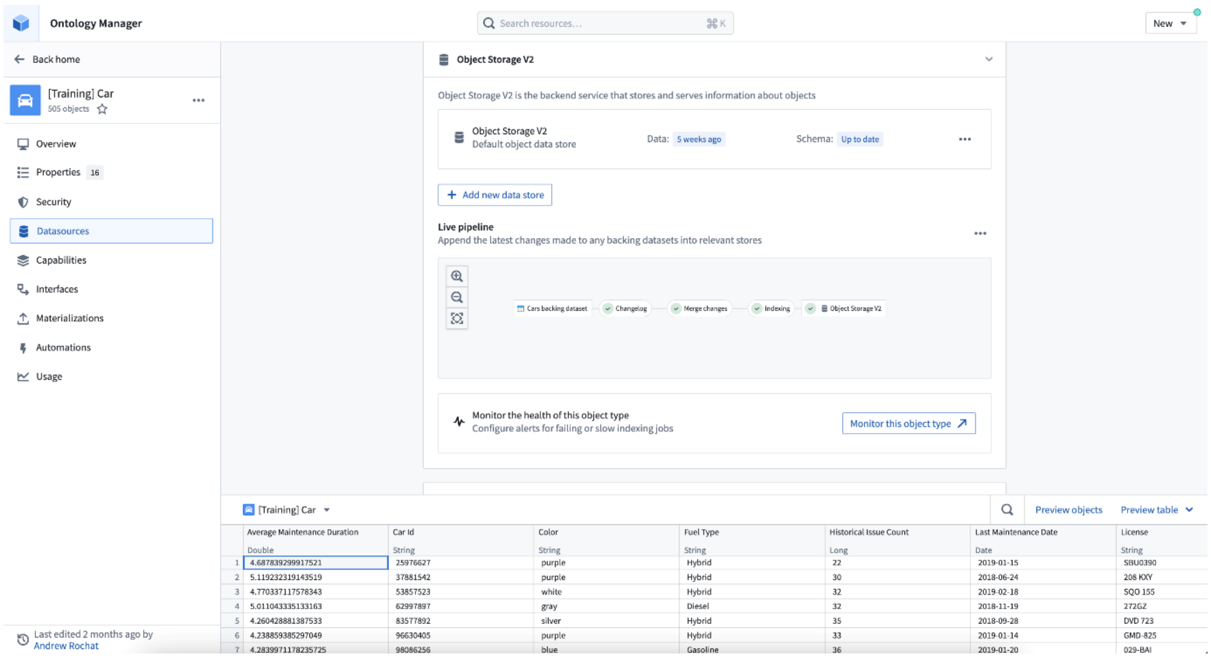
Frequently Asked Questions

I added a new column in my pipeline builder, why can't I see that new column downstream in my [output dataset / Ontology object type / Workshop module]?

- When you add a transformation that creates a new column in pipeline builder, there are a number of things that can go wrong, preventing you from seeing the column downstream.
- First, make sure you can see the column in the pipeline builder output dataset.
    - If you can't find the column in the output dataset, make sure that you have updated the output schema in your Pipeline Builder to account for the added column.
        - A dataset schema defines what columns your dataset has, so updating the schema is required in order to add a new column.
        - In the screenshot below, you can see a warning underneath the dataset in the top right labeled "Cars backing dataset". The warning says "1 dropped". That warning means 1 column is in the pipeline that is dropped from the output dataset.
        - 

        - When that warning is clicked, you are presented with a button that allows you to update your output schema.
    - Next, you need to save and deploy your change in order for your dataset to be rebuilt. Once your dataset finishes building, you will see your new column.

- Next, make sure you can see your column in the Ontology Manager Application.
    - First, check to make sure the sync between your dataset and the ontology was successful.
        - To do this, navigate to your ontology object type, and click on the "Datasources" tab on the left.
        - You should see a page similar to the screenshot below.



Object Storage V2 is the backend service that stores and serves information about objects

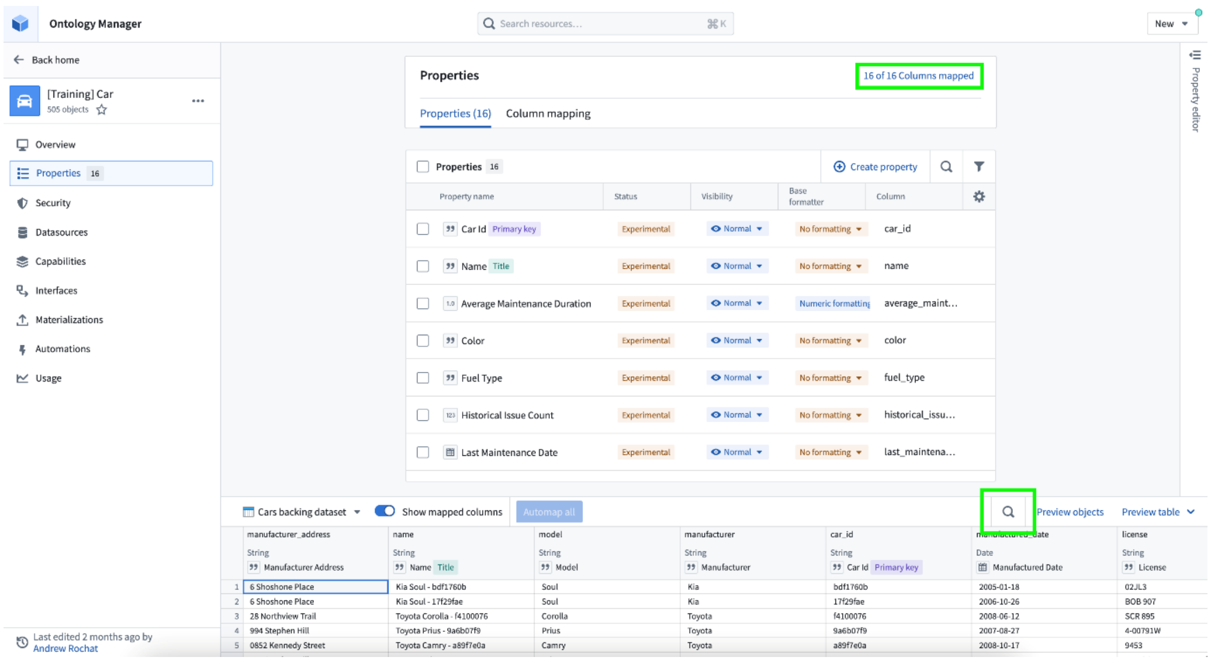
Object Storage V2
Default object data store
Data: 5 weeks ago
Schema: Up to date

Live pipeline
Append the latest changes made to any backing datasets into relevant stores

Monitor the health of this object type
Configure alerts for failing or slow indexing jobs

	Average Maintenance Duration Double	Car Id String	Color String	Fuel Type String	Historical Issue Count Long	Last Maintenance Date Date	License String
1	4.687839299917521	25976627	purple	Hybrid	22	2019-01-15	SBU0390
2	5.119232319143519	37881542	purple	Hybrid	30	2018-06-24	208 KXY
3	4.770337117578343	53857523	white	Hybrid	32	2019-02-18	SQO 155
4	5.011043335133163	62997897	gray	Diesel	32	2018-11-19	272GZ
5	4.260428881387533	83577892	silver	Hybrid	35	2018-09-28	DVD 723
6	4.238859385297049	96630405	purple	Hybrid	33	2019-01-14	GMD-825
7	4.2839971178235725	98086256	blue	Gasoline	36	2019-01-20	029-BAI

        - The important part of this page is the graph. Make sure the graph contains only check marks. If there are any X marks, click on them to investigate. If there are any loading icons, wait for them to finish.
    - Next, make sure the new column has been added in the "Properties" tab.
        - Go to the Properties tab and search for your property. If you are unable to find it, that means you have not mapped the property yet.



Property name	Status	Visibility	Base formatter	Column
Car Id Primary key	Experimental	Normal	No formatting	car_id
Name Title	Experimental	Normal	No formatting	name
Average Maintenance Duration	Experimental	Normal	Numeric formatting	average_maint...
Color	Experimental	Normal	No formatting	color
Fuel Type	Experimental	Normal	No formatting	fuel_type
Historical Issue Count	Experimental	Normal	No formatting	historical_issu...
Last Maintenance Date	Experimental	Normal	No formatting	last_maintena...

	manufacturer_address String Manufacturer Address	name String Name Title	model String Model	manufacturer String Manufacturer	car_id String Car Id Primary key	manufactured_date Date Manufactured Date	license String License
1	6 Shoshone Place	Kia Soul - bdf1760b	Soul	Kia	bdf1760b	2005-01-18	02JL3
2	6 Shoshone Place	Kia Soul - 17f29fae	Soul	Kia	17f29fae	2006-10-26	BOB 907
3	28 Northview Trail	Toyota Corolla - f4100076	Corolla	Toyota	f4100076	2008-06-12	SCR 895
4	994 Stephen Hill	Toyota Prius - 9a6b07f9	Prius	Toyota	9a6b07f9	2007-08-27	4-00791W
5	0852 Kennedy Street	Toyota Camry - a89f7e0a	Camry	Toyota	a89f7e0a	2008-10-17	9453

- Before you can map your property, you need to confirm that the ontology is ready to map your new property.
- The first step is to **refresh your screen**.
- Next, search for your property using the hourglass in the dataset preview section at the bottom of the screen. Once you confirm your property is there, click the "X of Y Columns mapped" button and click "Automap". You can also get there by clicking the Column mapping tab button.

- Finally, make sure you can see the column in workshop.
  - First make sure the object set variable you are using in workshop matches the object type that you are seeing the problem with.
  - Next, refresh your workshop application to see the new property.

I can see my new column in workshop, but I can't see the data.

- Most likely the issue is that the ontology hasn't finished syncing the data from the backing dataset. To check on this, go to the "Datasources" tab in the Ontology Manager Application, and confirm that the data has finished syncing.
  - If the data has finished syncing, every node in the graph will have a check mark.
- If the data has successfully synced and you still do not see your data in workshop, refresh your workshop application.

If I have triggered Ontology actions that have updated my data, what happens when I rebuild my backing dataset?

- Ontology Object Types have an edits-win policy. That means if there is a cell that has an edited value, that cell will keep the edited value, even if it receives an updated value from the backing dataset.

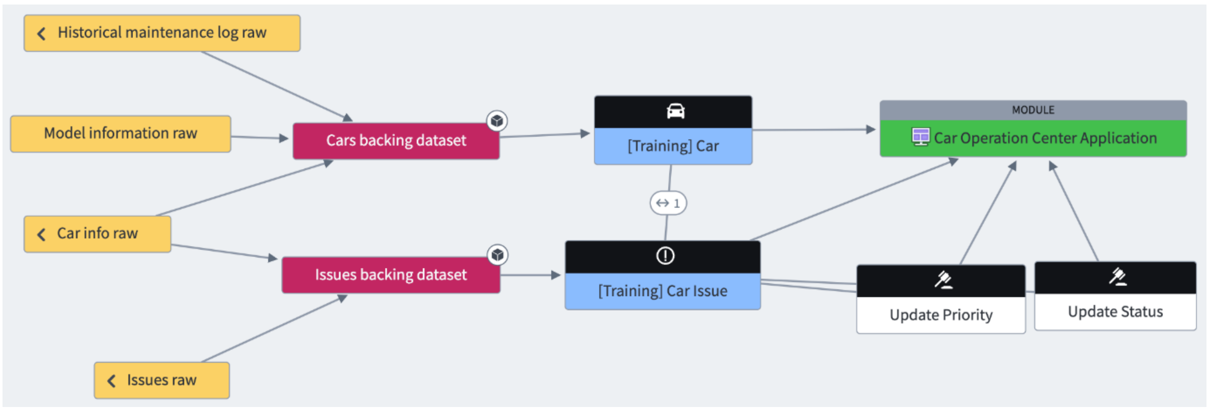
My Ontology object type is failing to sync.

- If you are in the "Datasources" tab in your Object Type, you should see a red icon that tells you that your sync has failed. If you click on that, you will see an error message that tells you why your sync has failed.
- Most likely, your sync is failing because your primary key is not unique.
    - To remediate this you first need to go step by step through your pipeline to see where the issue started.
        - Check the raw datasets. Was the primary key unique at that point?
        - If so, check your transformations. Where did your primary key lose its uniqueness.
    - Once you find the issue, think through ways to resolve it.
        - Are you joining your dataset with another dataset who's foreign key is not unique? If that is the case, perhaps the right approach is to fix the other dataset before your join transform.
        - Was your primary key never unique? If that is the case, you might need to use a drop duplicates transformation.

Architecture

Architecture Hints

- Here is a picture and explanation that shows what an example architecture for this workflow could look like.



- In the diagram, we have our four raw datasets on the left. Those datasets are transformed via pipeline builder into two ontology backing datasets. The arrows from the raw datasets to the backing datasets denote that a backing dataset is using that raw dataset as an input. This may be confusing, at first, but because of the concept of "joins", it makes sense.
- In this case, we are joining in historical maintenance logs and model information into the cars dataset to add columns that we will need in the workshop. For example, to get historical maintenance count, we need to bring in data from historical maintenance log raw. To get fuel type, transmission, etc, we need to bring in data from model information raw.
- Car info also appears to be an input in the issues backing dataset. That is because we need to use the car manufacturer to calculate the issue priority. We also need to join issues with cars so that we can filter out any issues that don't have cars.
- Within the ontology, we have a many to 1 link between car issue and car. This ontology link, which is configured in the ontology management application, is crucial for us to be able to jump back and forth between cars and issues. We use this link many times in the application.
- Finally, the car issue ontology object has two actions, "Update Priority" and "Update Status". Both of these actions are created in the ontology management application and used in the workshop app.
- The workshop module, called Car Operation Center Application is the crux of our architecture: this is our application.
- As a general rule of thumb, the less ontology object types, the better. That means if you have the option of doing an aggregation in the pipeline vs creating an extra ontology object type and doing the aggregation in workshop, it's better to do the aggregation in the pipeline. However, if you are aggregating over a property that might change (such as the number of open issues), you need to do this aggregation in workshop.

Architecture Warnings

- The maintenance table and the issues table are separate entities. You should never need to union them together.
- Although we are looking for a specific outcome, this project can be completed in a number of ways. For example, you might find yourself only needing two object types (Car & Issues) if you decide to join in maintenance data and aggregate in your pipeline. However, it's also possible to have 3 objects, Car, Issue, and Historical Maintenance, if you opt for aggregating maintenance data in workshop.

Raw Datasets

Raw Dataset Hints

These questions were already listed in an earlier section, but we are going to put them here as well to emphasize them.

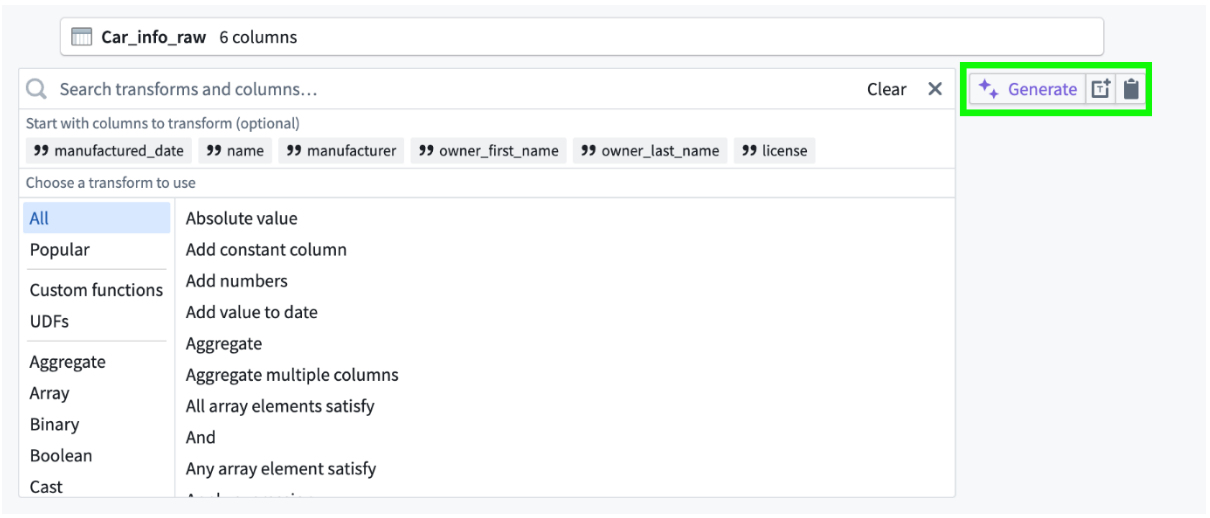
- Is there a unique identifier in this data? To view the uniqueness of a column of type "string", click the column name and view the presented statistics below.

- Are there any foreign keys to other datasets? Look at the column names along with the data contained in the columns. A foreign key allows you to link one dataset to another.
- Are there any data quality issues? Perhaps the column name or content is formatted poorly. Its important to clean any poorly formatted data.

Pipeline Builder

Pipeline Builder Hints

- Pipeline builder documentation can be found here https://www.palantir.com/docs/foundry/pipeline-builder/overview/
- Furthermore, an introduction on how to get started on pipeline builder can be found here https://www.palantir.com/docs/foundry/building-pipelines/create-batch-pipeline-pb/
- To get started, create a new pipeline builder within this project, and add the raw datasets as inputs into the pipeline builder.
- Next, you will need to create transforms for each of the raw datasets before they can be joined together. To transform a raw dataset, zoom in and hover over the raw dataset. Within the transform page, you can add "boards" which allow you to slice and dice the data depending the selected board. Boards (and transforms) build on top of each other, so each board you add will change the data for the following boards (and transforms).
- If you are unsure of what board to use but you know logically what you need to do, try searching the documentation or using the integrated AIP board generator tool in pipeline builder.


Car_info_raw 6 columns

Search transforms and columns… Clear ✕ Generate

Start with columns to transform (optional)

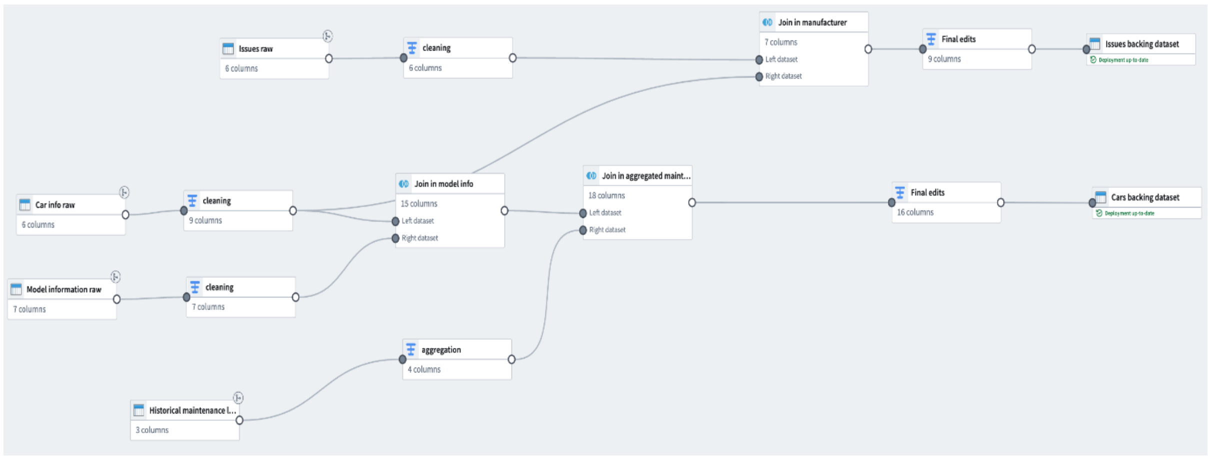
manufactured_date name manufacturer owner_first_name owner_last_name license

Choose a transform to use

- Your goal should be to transform the columns and datasets so that you have so that you have the necessary data format to support your front end workshop application. That means if a column is in all caps and you need the column to be titlecase, you should create a title case board for example.
- Some of the things you will need to do for each dataset are listed below. The following is not meant to be exhaustive, and should not be followed as instructions.
  - cars dataset

        - You will need to clean and transform the cars data so that you are able to join in historical maintenance issues and model information. The problem is there is no column in the cars dataset that corresponds to a column in the other dataset (without cleaning).
        - Try using the split string board + the array element board to slice up the name column.
        - You should also use the concatenate string board to create columns out of other columns. For example the name column in the frontend requires a format of <Manufacturer> <Model> - <Car ID>.
        - You should use the cast board to create a date column out of the badly formatted date.
    - historical maintenance log dataset
        - To get the historical maintenance date, historical maintenance count, and average maintenance duration, you should use an aggregation board in the pipeline builder, and then join the dataset into cars.
    - model information dataset
        - You should find a way to link the make_model_id column to something in the cars dataset. That way you can bring the model information properties into the cars dataset via join.
        - You should also use a case board to convert the transmission column from “A”, “M”, to “Automatic”, “Manual”.
    - Issues dataset
        - Within the issues dataset, you should perform some filtering to get rid of outdated issues. This should be done with the filter board.
        - You also may need to use the split string + array element boards to create a foreign key for the issues dataset.
        - Once you have pulled in the manufacturer information, you will need to use a case board to create a priority column.
        - You should also generate a static column for Status using the Constant Column board.
        - Theres a few ways that you can filter out issues that don't have an associated car. No matter what, you will need to join the issues dataset to the car's dataset. You can perform an inner join, or you can perform a left join and bring a column over. From there, you can then filter out rows that don't have a car.
- Once all of the datasets have been transformed, create your output datasets. Most likely, you should have a car output dataset and an issues output dataset. Finally, save and deploy your changes.

    - Remember that if you need to make a change to an output dataset at a later point in time, the change will only propagate once you save and deploy the changes. If you need to add a new column to the output dataset, make sure to click edit on the output dataset in the righthand panel and update the schema before saving a deploying.
- Here is an example of what your pipeline builder could look like:



Ontology Manager Application

Ontology Management Application (OMA) Hints

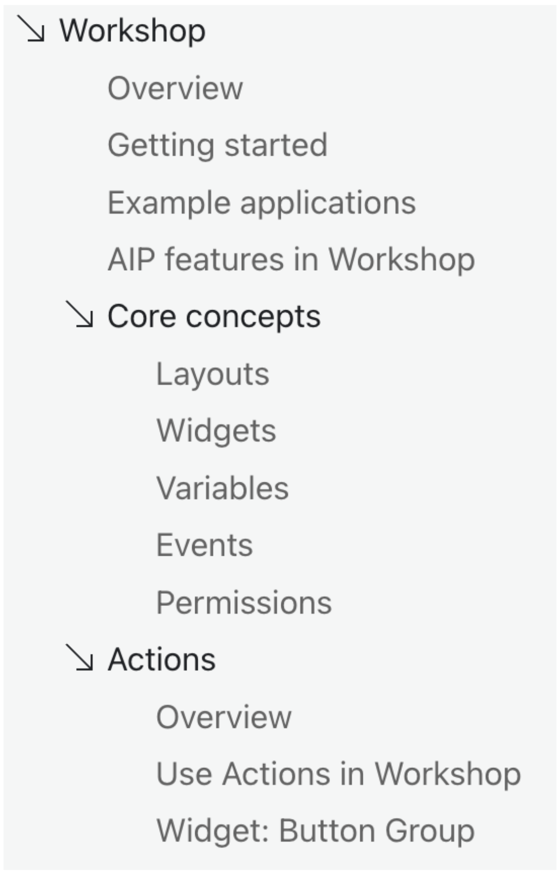
- Before creating ontology object types, read more about the ontology here https://www.palantir.com/docs/foundry/ontology/overview/
    - As a quick vocab lesson on the ontology: an object type is comparable to a dataset, an object is a row in a dataset, and a property is a column in a dataset.
- When creating an ontology object type, the first thing to do is ensure that you have a unique column that can be used as a primary key. To find a unique column, open the dataset, click on a column you suspect will be unique, and click "view stats" in the dropdown next to the column name. If the column is non-string, has any null values, or has any rows that appear twice, then that column should not be used as a primary key. In fact, if you try to create a primary key with a column that has null or duplicate values, you will get an error in the datasource tab of the OMA application.
    - As a hint, you should use car id for the car ontology object type and issue id for the ontology object type. Issue id may have some duplication issues that you need to sort out before it can work as a valid primary key.
- The next step is to create a title key. The title key is a readable short summary or "headline" of the object that distinguishes it from other objects.
    - In both the car and the issue object types, you should use the name property as defined above.
    - Using a concatenate string board in pipeline builder will also be useful for creating the name column in both the issue dataset and the car dataset.
- Finally, go into the ontology manager application and register a new object type.

- Once your new object type is setup, you will need to wait for your data to load into the ontology. While this is happening you can setup the necessary action you will need in your workshop app. An action is what allows a user to update data in the ontology from within a frontend application.
    - To setup the "Update Priority" and "Update Status" actions on the issues ontology object type, you will need to read through the action documentation here https://www.palantir.com/docs/foundry/action-types/overview/. Continue reading the following pages within the action-types documentation section for more information.
- You will also have to color format properties in the ontology management application. To format the priority field by color, go to the display section within the priority property. Search for conditional formatting to learn more. Search for "conditional formatting“ in the docs or via AIP Assist to learn more.
- Finally, make sure your fields are saved as the right type. For example, Any date fields should be saved as a date type, not a string type.

Workshop

Workshop Hints

- Before starting on your workshop module, review the documentation here https://www.palantir.com/docs/foundry/workshop/overview/. Workshop is a deceptively massive tool so it pays dividends to read up on the documentation before jumping in. The getting started guide can also be great. It's crucial to understand the **core concepts** before starting to develop in workshop.
- Reading through the **core concepts** section of the documentation before beginning development on your application will pay dividends.



- For the Update Status and Update Priority buttons, you will also have to understand actions in workshop.
- Below are some hints on how to create the different sections of the workshop module. The following is not meant to be exhaustive, and should not be followed as instructions.
- Cars Homepage
  - Filter bar
    - You should use the filter widget with a cars input object set to achieve this filter.
    - For the rest of this tab to depend on the outcome of you filter, you will need to create a new object set that starts from your initial object set and is filtered by your filter list variable.
  - Metric cards
    - You will need to create an numeric variables and select Object Set Aggregation to achieve most of the metrics. Before you can do that though, you will most likely need to perform a search around to get from “filtered cars“ to “linked issues of filterd cars”.
    - Once you are at “linked issues of filtered cars”, you will need to create even more object sets where you filter by property to get only open issues, or only high priority issues for example.

        - Finally, when you have the exact object sets you need, you can create numeric variables using the Object Set Aggregation variable creation option.
    - Table
        - To add “Number of Issues“ to the table, you can add a ”linked object / aggregation” and then perform a count aggregation in the workshop module.
    - Chart
        - You will need to use an object set that is of type issues to create the chart. You should be able to segment by priority. Once you have set up conditional formatting in the ontology management application, the chart colors will be the same as the mockup.
- Selected Car Overlay
    - General Hints
        - To create an overlay, look in the top left, under Layout.
    - Parent Section
        - In the parent section mockup, you will notice the X button is at the same level as the title and metrics. You will also notice that default overlay section title doesn’t allow for metrics or other widgets to be added. Therefore you should create a section header with the title and metrics that you want. Then you should put an X button in the top right of the header that closes the overlay.
        - To mimic the overlay padding style and design that the mockup uses, select regular padding with inner section style of minimal elevated.
    - Car Information
        - Use metric cards with string variables. When creating a string variable, select the object property option to be able to select a property from an object set. This only works if you have an object set with only 1 object in it. Make sure you are using the selected car object set.
    - Additional Manufacturing Information
        - Try using the property list widget.
    - Issues List
        - To create this table, you should create an object set that starts from the selected car, and gets all linked issues.

- Issues Inbox
    - Filter Bar
        - Keyword searching can be found in the widget configuration. This is not a property you have to add to the ontology object type.
    - Table
        - to create the metrics in the header section that show number of issues, create a numeric variable that performs an object set aggregation.
- Selected Issue Section
    - Parent Section
        - To make this section conditionally visible, you need to create a boolean variable. Create an event in the table that sets the variable to true when an object is clicked. From there, make the section conditionally visible based on the boolean variable.
    - Buttons - Update Priority & Update Status
        - You should create these buttons and then select Action as the on click event. From there, you can select the actions you configured in the ontology management application.
        - You should prefill the selected issue by clicking "Select parameters to configure" and then passing in the selected issue.
    - Metrics for priority and status
        - You should be able to create string variables and use the object property variable creator with the selected issue object set to create these metrics.
    - Issue details section
        - You should use a property list widget with the selected issue as the input object set. To get properties from the car, you can either pull them in via a pipeline builder join, or you can select car as a linked object, and choose the property from car you want to show.
    - Button - View Associated Car
        - To open the car overlay with the intended car, you need to set the table active object to be the car that you want. To do this, first create an object set which is the car you are interested in. Then use the set variable event to set the car table active object using the object set you created. Finally, create an event that opens the overlay.