

```
1 ID - The unique ID used to keep track of the sample at NASA.
2 Mission - The mission responsible for retrieving the sample.
3 Type - The type of sample (type of rock or other classification).
4 Subtype - A more specific type classification.
5 Weight (g) - The original weight of the sample, in grams.
6 Pristine (%) - The percentage of the sample that remains (some sample is used up during research).
```

```
In [199]: 1 import pandas as pd
```

```
In [200]: 1 rock_samples = pd.read_csv("rocksamples.csv")
2 rock_samples.head()
```

Out[200]:

	ID	Mission	Type	Subtype	Weight (g)	Pristine (%)
0	10001	Apollo11	Soil	Unsieved	125.8	88.36
1	10002	Apollo11	Soil	Unsieved	5629.0	93.73
2	10003	Apollo11	Basalt	Ilmenite	213.0	65.56
3	10004	Apollo11	Core	Unsieved	44.8	71.76
4	10005	Apollo11	Core	Unsieved	53.4	40.31

```
In [201]: 1 rock_samples.shape
```

Out[201]: (2229, 6)

```
In [202]: 1 rock_samples.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2229 entries, 0 to 2228
Data columns (total 6 columns):
#   Column          Non-Null Count  Dtype
---  -
0   ID              2229 non-null   int64
1   Mission         2229 non-null   object
2   Type            2229 non-null   object
3   Subtype         2226 non-null   object
4   Weight (g)      2229 non-null   float64
5   Pristine (%)    2229 non-null   float64
dtypes: float64(2), int64(1), object(3)
memory usage: 104.6+ KB
```

```
In [203]: 1 rock_samples.isna().sum()
```

Out[203]: ID 0
Mission 0
Type 0
Subtype 3
Weight (g) 0
Pristine (%) 0
dtype: int64

```
In [204]: 1 rock_samples[rock_samples['Subtype'].isnull()]
```

Out[204]:

	ID	Mission	Type	Subtype	Weight (g)	Pristine (%)
89	12023	Apollo12	Special	NaN	407.90	66.89
355	15014	Apollo15	Special	NaN	333.20	100.00
1513	70149	Apollo17	Basalt	NaN	0.95	0.00

Convert the sample weight

While details of rocket design are proprietary, some information is publicly available. For example, you can gather data such as the weight of the spacecraft modules that carried the samples back to Earth and the total amount of weight that each rocket can transport. These values can be used to calculate the maximum weight of rock samples that can be collected and returned to Earth.

To prepare the rock sample data for later calculations, we need to understand that rocket weight is often measured in kilograms, not grams. Therefore, we need to convert the original rock sample weights from grams to kilograms for easier data analysis later.

```
In [205]: 1 rock_samples['Weight (g)'] = rock_samples['Weight (g)'].apply(lambda x: x*0.001)
```

```
In [206]: 1 rock_samples.rename(columns={'Weight (g)' : 'Weight(Kg)'}, inplace=True)
```

```
In [207]: 1 rock_samples.head()
```

Out[207]:

	ID	Mission	Type	Subtype	Weight(Kg)	Pristine (%)
0	10001	Apollo11	Soil	Unsieved	0.1258	88.36
1	10002	Apollo11	Soil	Unsieved	5.6290	93.73
2	10003	Apollo11	Basalt	Ilmenite	0.2130	65.56
3	10004	Apollo11	Core	Unsieved	0.0448	71.76
4	10005	Apollo11	Core	Unsieved	0.0534	40.31

Create a new DataFrame

Pandas, the Python library we are using to do our data analysis, has a structure called a DataFrame, and it's really effective for representing 2D data. You might have recognized that, when you run the `rock_samples.head()` code, what is printed out looks almost like a snapshot of an Excel worksheet, which is a great way to think about DataFrames in pandas.

The `rock_samples` DataFrame has a row for every sample that was collected but, as we mentioned earlier, we want to understand the rock samples in total as they relate to the specific rockets that brought them back.

Create a new DataFrame called missions that will be a summary of data for each of the six Apollo

```
In [208]: 1 mission = pd.DataFrame()
```

```
In [209]: 1 mission['Mission'] = df['Mission'].unique()
```

```
In [210]: 1 mission.head()
```

Out[210]:

	Mission
0	Apollo11
1	Apollo12
2	Apollo14
3	Apollo15
4	Apollo16

```
In [211]: 1 mission.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 1 columns):
#   Column   Non-Null Count  Dtype
---  -
0    Mission  6 non-null      object
dtypes: object(1)
memory usage: 176.0+ bytes
```

Sum total sample weight by mission

Now you can add a new column to the missions DataFrame to represent the sum of all samples collected on that mission.

```
In [212]: 1 sample_total_weight= df.groupby('Mission')['Weight(Kg)'].sum()
```

```
In [213]: 1 missions = pd.merge(mission, sample_total_weight, on='Mission')
2 missions.rename(columns={'Weight(Kg)': 'Sample weight (kg)'}, inplace=True)
3 missions
```

Out[213]:

	Mission	Sample weight (kg)
0	Apollo11	21.55424
1	Apollo12	34.34238
2	Apollo14	41.83363
3	Apollo15	75.39910
4	Apollo16	92.46262
5	Apollo17	109.44402

Get the difference in weights across missions

We're not rocket experts, so it's important to take a look at a lot of different cross sections of data that are available to you. In this case, we can see that the total weight of the samples increased with each mission, but it's hard to immediately see by how much. We can add one more column to the missions DataFrame that simply grabs the difference between the current row and the row preceding it:

```
In [214]: 1 missions['Difference'] = missions['Sample weight (kg)'].diff()  
          2 missions
```

Out[214]:

	Mission	Sample weight (kg)	Difference
0	Apollo11	21.55424	NaN
1	Apollo12	34.34238	12.78814
2	Apollo14	41.83363	7.49125
3	Apollo15	75.39910	33.56547
4	Apollo16	92.46262	17.06352
5	Apollo17	109.44402	16.98140

```
In [215]: 1 missions['Difference'] = missions['Difference'].fillna(value=0)  
          2 missions
```

Out[215]:

	Mission	Sample weight (kg)	Difference
0	Apollo11	21.55424	0.00000
1	Apollo12	34.34238	12.78814
2	Apollo14	41.83363	7.49125
3	Apollo15	75.39910	33.56547
4	Apollo16	92.46262	17.06352
5	Apollo17	109.44402	16.98140

```
In [ ]: 1
```

```
In [216]: 1 missions['Lunar module (LM)'] = ['Eagle (LM-5)', 'Intrepid (LM-6)', 'Antares (LM-
2 missions['LM mass (kg)'] = [15103, 15235, 15264, 16430, 16445, 16456]
3 missions['LM mass diff'] = missions['LM mass (kg)'].diff()
4 missions['LM mass diff'] = missions['LM mass diff'].fillna(value=0)
5
6 missions['Command module (CM)'] = ['Columbia (CSM-107)', 'Yankee Clipper (CM-108)
7 missions['CM mass (kg)'] = [5560, 5609, 5758, 5875, 5840, 5960]
8 missions['CM mass diff'] = missions['CM mass (kg)'].diff()
9 missions['CM mass diff'] = missions['CM mass diff'].fillna(value=0)
10
11 missions
```

Out[216]:

	Mission	Sample weight (kg)	Difference	Lunar module (LM)	LM mass (kg)	LM mass diff	Command module (CM)	CM mass (kg)	CM mass diff
0	Apollo11	21.55424	0.00000	Eagle (LM-5)	15103	0.0	Columbia (CSM-107)	5560	0.0
1	Apollo12	34.34238	12.78814	Intrepid (LM-6)	15235	132.0	Yankee Clipper (CM-108)	5609	49.0
2	Apollo14	41.83363	7.49125	Antares (LM-8)	15264	29.0	Kitty Hawk (CM-110)	5758	149.0
3	Apollo15	75.39910	33.56547	Falcon (LM-10)	16430	1166.0	Endeavor (CM-112)	5875	117.0
4	Apollo16	92.46262	17.06352	Orion (LM-11)	16445	15.0	Casper (CM-113)	5840	-35.0
5	Apollo17	109.44402	16.98140	Challenger (LM-12)	16456	11.0	America (CM-114)	5960	120.0

```
In [217]: 1 missions['Total weight (kg)'] = missions['LM mass (kg)'] + missions['CM mass (kg)']
2 missions['Total weight diff'] = missions['LM mass diff'] + missions['CM mass diff']
3 missions
```

Out[217]:

	Mission	Sample weight (kg)	Difference	Lunar module (LM)	LM mass (kg)	LM mass diff	Command module (CM)	CM mass (kg)	CM mass diff	Total weight (kg)	Total weight diff
0	Apollo11	21.55424	0.00000	Eagle (LM-5)	15103	0.0	Columbia (CSM-107)	5560	0.0	20663	0.0
1	Apollo12	34.34238	12.78814	Intrepid (LM-6)	15235	132.0	Yankee Clipper (CM-108)	5609	49.0	20844	181.0
2	Apollo14	41.83363	7.49125	Antares (LM-8)	15264	29.0	Kitty Hawk (CM-110)	5758	149.0	21022	178.0
3	Apollo15	75.39910	33.56547	Falcon (LM-10)	16430	1166.0	Endeavor (CM-112)	5875	117.0	22305	1283.0
4	Apollo16	92.46262	17.06352	Orion (LM-11)	16445	15.0	Casper (CM-113)	5840	-35.0	22285	-20.0
5	Apollo17	109.44402	16.98140	Challenger (LM-12)	16456	11.0	America (CM-114)	5960	120.0	22416	131.0

Compare the data

The interesting thing about predicting how much sample each Artemis mission can bring back is that we don't yet know the full specs of the spacecraft that the Artemis mission plans on using. Using some information from the NASA Factsheet on the Space Launch System (SLS) and Orion Modules, we have data on weights and payloads.

A payload is basically the total amount of weight that a rocket can get up through our atmosphere and into space. So the likelihood that the payload number is more accurate than the exact weights of each module is high, because deciding the payload will likely affect each of the other design decisions.

We know that the Saturn V payload was 43,500 kg, and the weights of the modules varied from mission to mission. So, to determine the ratios that will allow us to make predictions about the Artemis missions, we can use:

Saturn V payload Mission sample weight Mission module weight

```
In [218]: 1 # Sample-to-weight ratio
2 saturnVPayload = 43500
3 missions['Crewed area : Payload'] = missions['Total weight (kg)'] / saturnVPayload
4 missions['Sample : Crewed area'] = missions['Sample weight (kg)'] / missions['Total weight (kg)']
5 missions['Sample : Payload'] = missions['Sample weight (kg)'] / saturnVPayload
6 missions
```

Out[218]:

	Mission	Sample weight (kg)	Difference	Lunar module (LM)	LM mass (kg)	LM mass diff	Command module (CM)	CM mass (kg)	CM mass diff	Total weight (kg)	Total weight diff	Ratio
0	Apollo11	21.55424	0.00000	Eagle (LM-5)	15103	0.0	Columbia (CSM-107)	5560	0.0	20663	0.0	0.4
1	Apollo12	34.34238	12.78814	Intrepid (LM-6)	15235	132.0	Yankee Clipper (CM-108)	5609	49.0	20844	181.0	0.4
2	Apollo14	41.83363	7.49125	Antares (LM-8)	15264	29.0	Kitty Hawk (CM-110)	5758	149.0	21022	178.0	0.4
3	Apollo15	75.39910	33.56547	Falcon (LM-10)	16430	1166.0	Endeavor (CM-112)	5875	117.0	22305	1283.0	0.5
4	Apollo16	92.46262	17.06352	Orion (LM-11)	16445	15.0	Casper (CM-113)	5840	-35.0	22285	-20.0	0.5
5	Apollo17	109.44402	16.98140	Challenger (LM-12)	16456	11.0	America (CM-114)	5960	120.0	22416	131.0	0.5

We can then use the mean() function to take the average of all those ratios across all the missions.

```
In [219]: 1 crewedArea_payload_ratio = missions['Crewed area : Payload'].mean()
2 sample_crewedArea_ratio = missions['Sample : Crewed area'].mean()
3 sample_payload_ratio = missions['Sample : Payload'].mean()
4 print(crewedArea_payload_ratio)
5 print(sample_crewedArea_ratio)
6 print(sample_payload_ratio)
```

```
0.4963026819923371
0.002848764392685611
0.0014369195019157087
```

As mentioned in the preceding unit, we can use the NASA Factsheet on the Space Launch System (SLS) and Orion Modules to gather estimated data on the rockets and modules that will be used in the Artemis program.

As a reminder, the Artemis program is NASA's second set of missions to land humans on the surface of the Moon. The program will launch in 2025 and will send not only the next pair of humans, but also the first woman to set foot on the Moon. The preparation for this mission is even bigger than focusing on a Moon landing. It will also provide space for a commercial payload on the ship and is the first step along the Moon to Mars program. So, while the Artemis missions will likely bring home additional samples, there are other goals that might affect the amount of capacity that's required to do so.

Create an Artemis mission DataFrame We don't have all the details about the Artemis mission, but we do know currently that three iterations of the rocket will be cycled through for each mission. Each rocket will have one version meant to sustain a crew and one meant only for cargo. For the purposes of this module, we will focus only on the three rockets meant to house crew, to be more aligned with the Apollo missions. We also know that the expected payload of the Space Launch System (SLS) is expected to grow with each iteration, but that the current weight of Orion (the command and lunar modules combined) has one estimated weight today.

Again, we will call the command and lunar modules the crewed area, and we can create a DataFrame with the information we have about the three crewed missions:

```
In [220]: 1 artemis_crewedArea = 26520
2 artemis_mission = pd.DataFrame({'Mission': ['artemis1', 'artemis1b', 'artemis2'],
3                                     'Total weight (kg)': [artemis_crewedArea, artemis_
4                                                         'Payload (kg)': [26988, 37965, 42955]})
5 artemis_mission
```

Out[220]:

	Mission	Total weight (kg)	Payload (kg)
0	artemis1	26520	26988
1	artemis1b	26520	37965
2	artemis2	26520	42955

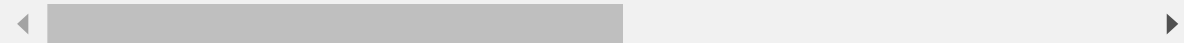
And we can estimate the weight of samples based on the ratios we determined from the Apollo missions:

```
In [221]: 1 artemis_mission['Sample weight from total (kg)'] = artemis_mission['Total weight  
2 artemis_mission['Sample weight from payload (kg)'] = artemis_mission['Payload (kg)  
3 artemis_mission
```

```
Out[221]:
```

	Mission	Total weight (kg)	Payload (kg)	Sample weight from total (kg)	Sample weight from payload (kg)
0	artemis1	26520	26988	75.549232	38.779584
1	artemis1b	26520	37965	75.549232	54.552649
2	artemis2	26520	42955	75.549232	61.722877

```
In [222]: 1 artemis_mission['Estimated sample weight (kg)'] = (artemis_mission['Sample weight  
2 artemis_mission
```



```
Out[222]:
```

	Mission	Total weight (kg)	Payload (kg)	Sample weight from total (kg)	Sample weight from payload (kg)	Estimated sample weight (kg)
0	artemis1	26520	26988	75.549232	38.779584	57.164408
1	artemis1b	26520	37965	75.549232	54.552649	65.050940
2	artemis2	26520	42955	75.549232	61.722877	68.636054

We can see now that the three Artemis missions can likely return 57.16 kg, 65.05 kg, and 68.64 kg, respectively.

Now the question is, what kinds of rocks should they prioritize?

Exercise - Prioritize Moon rock sample gathering based on data

- 1 Determining which types of samples to collect from the Moon requires expertise, but we can start to make some assumptions to learn how to clean and manipulate data.
- 2
- 3 First, we can determine how much remains of each sample that was returned from the Apollo missions by multiplying the weight of the sample that was originally collected by the percentage of remaining pristine sample.


```
In [223]: 1 rock_samples['Remaining (kg)'] = rock_samples['Weight(Kg)'] * (rock_samples['Pristine (%)'] - 1)
          2 rock_samples.head()
```

Out[223]:

	ID	Mission	Type	Subtype	Weight(Kg)	Pristine (%)	Remaining (kg)
0	10001	Apollo11	Soil	Unsieved	0.1258	88.36	0.111157
1	10002	Apollo11	Soil	Unsieved	5.6290	93.73	5.276062
2	10003	Apollo11	Basalt	Ilmenite	0.2130	65.56	0.139643
3	10004	Apollo11	Core	Unsieved	0.0448	71.76	0.032148
4	10005	Apollo11	Core	Unsieved	0.0534	40.31	0.021526

```
In [224]: 1 rock_samples.describe()
```

Out[224]:

	ID	Weight(Kg)	Pristine (%)	Remaining (kg)
count	2229.000000	2229.000000	2229.000000	2229.000000
mean	52058.432032	0.168253	84.512764	0.138103
std	26207.651471	0.637286	22.057299	0.525954
min	10001.000000	0.000000	0.000000	0.000000
25%	15437.000000	0.003000	80.010000	0.002432
50%	65527.000000	0.010200	92.300000	0.008530
75%	72142.000000	0.093490	98.140000	0.078240
max	79537.000000	11.729000	180.000000	11.169527

This information helps us see that, on average, each sample weighs about .16 kg and has about 84% of the original amount remaining. We can use this knowledge to determine which samples are likely running low, which means that they have been used a lot by researchers.

```
In [225]: 1 low_samples = rock_samples.loc[(rock_samples['Weight(Kg)'] >= .16) & (rock_sample
2 low_samples
```

Out[225]:

	ID	Mission	Type	Subtype	Weight(Kg)	Pristine (%)	Remaining (kg)
11	10017	Apollo11	Basalt	Ilmenite	0.9730	43.71	0.425298
14	10020	Apollo11	Basalt	Ilmenite	0.4250	27.88	0.118490
15	10021	Apollo11	Breccia	Regolith	0.2500	30.21	0.075525
29	10045	Apollo11	Basalt	Olivine	0.1850	12.13	0.022441
37	10057	Apollo11	Basalt	Ilmenite	0.9190	35.15	0.323028
39	10059	Apollo11	Breccia	Regolith	0.1880	36.94	0.069447
52	10072	Apollo11	Basalt	Ilmenite	0.4470	15.22	0.068033
59	10086	Apollo11	Soil	Unsieved	0.8230	0.01	0.000082
68	12002	Apollo12	Basalt	Olivine	1.5300	49.04	0.750312
69	12003	Apollo12	Soil	Unsieved	0.3000	28.52	0.085560
72	12006	Apollo12	Basalt	Olivine	0.2064	0.53	0.001094
94	12028	Apollo12	Core	Unsieved	0.1896	27.56	0.052254
113	12047	Apollo12	Basalt	Ilmenite	0.1930	33.70	0.065041
118	12053	Apollo12	Basalt	Pigeonite	0.8790	46.43	0.408120
120	12055	Apollo12	Basalt	Pigeonite	0.9120	26.00	0.237120
353	15012	Apollo15	Soil	Unsieved	0.3122	5.77	0.018014
354	15013	Apollo15	Soil	Unsieved	0.2962	8.80	0.026066
426	15205	Apollo15	Breccia	Regolith	0.3373	24.70	0.083313
534	15405	Apollo15	Breccia	ImpactMelt	0.5131	35.91	0.184254
570	15475	Apollo15	Basalt	Pigeonite	0.4068	49.60	0.201773
575	15498	Apollo15	Breccia	Regolith	2.3400	34.11	0.798174
607	15555	Apollo15	Basalt	Olivine	9.6140	46.61	4.481085
718	60016	Apollo16	Breccia	Fragmental	4.3070	49.57	2.134980
722	60025	Apollo16	Breccia	Anorthosite	1.8360	43.61	0.800680
1221	66075	Apollo16	Breccia	Fragmental	0.3471	32.97	0.114439
1645	71566	Apollo17	Basalt	Ilmenite	0.4144	45.06	0.186729
2183	79155	Apollo17	Basalt	Ilmenite	0.3188	25.97	0.082792

```
In [226]: 1 low_samples.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 27 entries, 11 to 2183
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   ID                    27 non-null    int64
1   Mission              27 non-null    object
2   Type                 27 non-null    object
3   Subtype              27 non-null    object
4   Weight(Kg)           27 non-null    float64
5   Pristine (%)         27 non-null    float64
6   Remaining (kg)       27 non-null    float64
dtypes: float64(3), int64(1), object(3)
memory usage: 1.7+ KB
```

Twenty-seven samples seem like a small amount to base a recommendation on. We can probably find some other samples that are needed for more research here on Earth. To discover them, we can use the `unique()` function to see how many unique types we have across the `low_samples` and `rock_samples` DataFrames.

```
In [227]: 1 low_samples.Type.unique()
```

```
Out[227]: array(['Basalt', 'Breccia', 'Soil', 'Core'], dtype=object)
```

```
In [228]: 1 rock_samples.Type.unique()
```

```
Out[228]: array(['Soil', 'Basalt', 'Core', 'Breccia', 'Special', 'Crustal'],
               dtype=object)
```

We can see that, although six unique types were collected across all samples, the samples that are running low are from only four unique types. But this data doesn't tell us everything about the samples we might want to focus on. For example, in our `low_samples` DataFrame, how many of each type are considered low?

```
In [229]: 1 low_samples.groupby('Type')['Weight(Kg)'].count()
```

```
Out[229]: Type
Basalt    14
Breccia    8
Core       1
Soil       4
Name: Weight(Kg), dtype: int64
```

Notice that there are more Basalt and Breccia type rocks with low samples than those of Core and Soil. Additionally, because the likelihood is high that every mission has some Core and Soil collection requirements, we can focus on the Basalt and Breccia rock types for the samples that we need to have collected:

```
In [230]: 1 needed_samples = low_samples[low_samples['Type'].isin(['Basalt', 'Breccia'])]
          2 needed_samples.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 22 entries, 11 to 2183
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype
---  -
0   ID                    22 non-null    int64
1   Mission               22 non-null    object
2   Type                  22 non-null    object
3   Subtype               22 non-null    object
4   Weight(Kg)            22 non-null    float64
5   Pristine (%)          22 non-null    float64
6   Remaining (kg)        22 non-null    float64
dtypes: float64(3), int64(1), object(3)
memory usage: 1.4+ KB
```

Exercise - Develop a recommendation of Moon rock samples to be collected

Let's take a step back and compare the samples that are running low to all the samples collected on Apollo missions. We can compare the total weight from the needed_samples DataFrame to the rock_samples DataFrame.

```
In [231]: 1 needed_samples.groupby('Type')['Weight(Kg)'].sum()
```

```
Out[231]: Type
Basalt      17.4234
Breccia     10.1185
Name: Weight(Kg), dtype: float64
```

```
In [232]: 1 rock_samples.groupby('Type')['Weight(Kg)'].sum()
```

```
Out[232]: Type
Basalt      93.14077
Breccia     168.88075
Core        19.93587
Crustal      4.74469
Soil        87.58981
Special      0.74410
Name: Weight(Kg), dtype: float64
```

```
1 This bit of information really stands out: we didn't have many Crustal rocks in
  the first place.
2
3 We can add Crustal rocks to the set of needed samples:
```

```
In [233]: 1 needed_samples = pd.concat([needed_samples, rock_samples.loc[rock_samples['Type']
2 needed_samples.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 68 entries, 11 to 2189
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype
---  -
0   ID                    68 non-null    int64
1   Mission              68 non-null    object
2   Type                 68 non-null    object
3   Subtype              68 non-null    object
4   Weight(Kg)          68 non-null    float64
5   Pristine (%)         68 non-null    float64
6   Remaining (kg)       68 non-null    float64
dtypes: float64(3), int64(1), object(3)
memory usage: 4.2+ KB
```

Summary of needed samples

The final step is to consolidate everything we know into one table that can be shared with the astronauts. First, we need a column for each type of rock that we want more samples of:

```
In [234]: 1 needed_samples_overview = pd.DataFrame()
2 needed_samples_overview['Type'] = needed_samples.Type.unique()
3 needed_samples_overview
```

```
Out[234]:
```

	Type
0	Basalt
1	Breccia
2	Crustal

Next, we want the total weight of each type of rock that was originally collected:

```
In [235]: 1 needed_sample_weights = needed_samples.groupby('Type')['Weight(Kg)'].sum().reset_
2 needed_samples_overview = pd.merge(needed_samples_overview, needed_sample_weights
3 needed_samples_overview.rename(columns={'Weight(Kg)': 'Total weight (kg)'}, inplace=
4 needed_samples_overview
```

```
Out[235]:
```

	Type	Total weight (kg)
0	Basalt	17.42340
1	Breccia	10.11850
2	Crustal	4.74469

When astronauts are up on the Moon, one way they can identify rocks is by their size. If we can give them an estimated size of each type of rock that might make their collection process easier.

```
In [236]: 1 needed_sample_ave_weights = needed_samples.groupby('Type')['Weight(Kg)'].mean().r
2 needed_samples_overview = pd.merge(needed_samples_overview, needed_sample_ave_w
3 needed_samples_overview.rename(columns={'Weight(Kg)': 'Average weight (kg)'}, inpl
4 needed_samples_overview
```

Out[236]:

	Type	Total weight (kg)	Average weight (kg)
0	Basalt	17.42340	1.244529
1	Breccia	10.11850	1.264813
2	Crustal	4.74469	0.103145

Crustals are small! They're probably harder to spot, so no wonder we don't have many of them.

We probably want to give the astronauts some indication of how many of each type we want them to collect. So, for the three types we're looking for, we should grab the total number we have of each type and get the remaining percentage of each type of rock.

```
In [237]: 1 total_rock_count = rock_samples.groupby('Type')['ID'].count().reset_index()
2 needed_samples_overview = pd.merge(needed_samples_overview, total_rock_count, on=
3 needed_samples_overview.rename(columns={'ID': 'Number of samples'}, inplace=True)
4 total_rocks = needed_samples_overview['Number of samples'].sum()
5 needed_samples_overview['Percentage of rocks'] = needed_samples_overview['Number
6 needed_samples_overview
```

Out[237]:

	Type	Total weight (kg)	Average weight (kg)	Number of samples	Percentage of rocks
0	Basalt	17.42340	1.244529	351	0.258850
1	Breccia	10.11850	1.264813	959	0.707227
2	Crustal	4.74469	0.103145	46	0.033923

And finally, to tie it all back into a recommendation to the Artemis program, we can determine the average weight of samples we estimated in the preceding unit.

```
In [238]: 1 artemis_ave_weight = artemis_mission['Estimated sample weight (kg)'].mean()
2 artemis_ave_weight
```

Out[238]: 63.61713411579792

We can use this number to determine how many of each rock we want the astronauts to aim to collect:

```
In [240]: 1 needed_samples_overview['Weight to collect'] = needed_samples_overview['Percentage of rocks'] * 100
          2 needed_samples_overview['Rocks to collect'] = needed_samples_overview['Weight to collect'] / 100
          3 needed_samples_overview
```

Out[240]:

	Type	Total weight (kg)	Average weight (kg)	Number of samples	Percentage of rocks	Weight to collect	Rocks to collect
0	Basalt	17.42340	1.244529	351	0.258850	16.467267	13.231731
1	Breccia	10.11850	1.264813	959	0.707227	44.991764	35.571884
2	Crustal	4.74469	0.103145	46	0.033923	2.158103	20.922917

So, we might tell the Artemis astronauts to try to collect 13 Basalt rocks, 35 Breccia rocks, and 20 Crustal rocks. Whew!

Summary

Completed 100 XP 2 minutes You've watched Fei Fei design, prototype, test, and ultimately succeed at creating a rocket to reach the Moon. And you've seen some of what NASA scientists and engineers might analyze to determine how to account for every ounce of weight on a rocket. It's now clear why data-science skills and practices are so critical to space exploration.

NASA might have a lot more information and clearer standards than we did for this analysis, but the exploration and understanding of data from multiple sources is pervasive throughout many other industries. So you can apply the knowledge and skills you've acquired to solving countless other problems.

In this module, you have:

Gathered information about samples brought back from the Moon via the Apollo missions. Acquired data about the types of spacecraft and rockets used for the Apollo and upcoming Artemis missions. Compiled DataFrames, or tables, of that data, which tells stories and provides insights. Created a prediction of how much sample weight could be returned from the Artemis missions. Made a recommendation for the amount and types of rocks the Artemis astronauts should focus their efforts on, based on the rocks that are currently being used for research here on Earth. The data journey can take many turns. With your new data-wrangling skills, you should feel empowered to find other datasets or cut the data that's used here in new ways. By doing so, you'll uncover new information about space exploration and Moon sample recovery, or about another problem you might be interested in.

In []:

1

In []:

1

