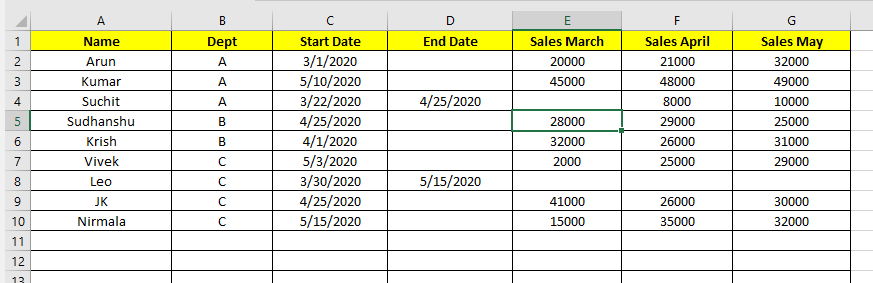
**Excel vs Python**

1. **Importing Our Data**

**Excel:**

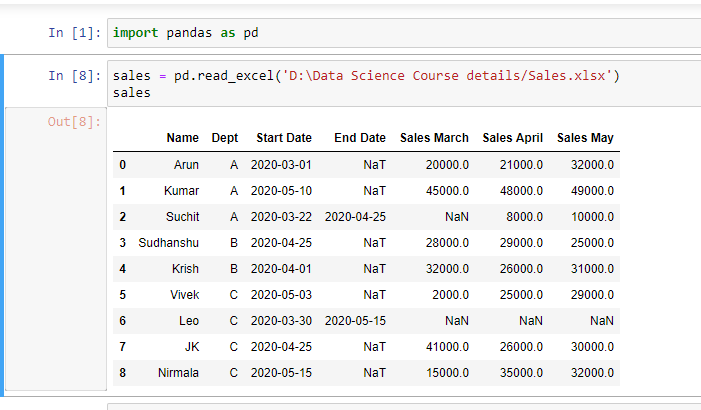


Python:

*Import pandas as pd*

*sales = pd.read\_excel(‘sales.xlsx’)*

*sales*



We can notice a few differences between how pandas represent the data vs what we saw in Excel:

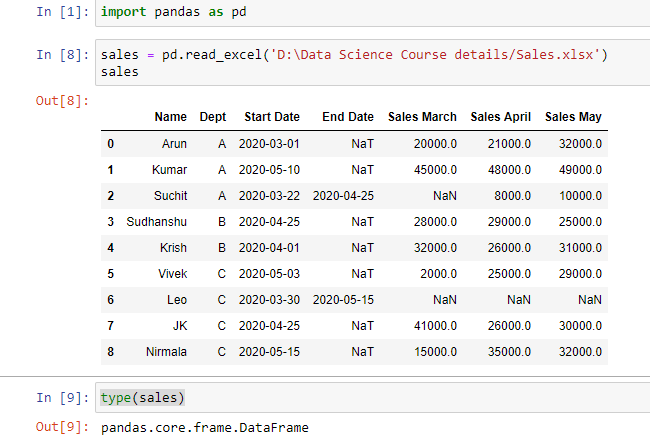
* In pandas, **the row numbers start at 0 versus 1 in Excel**.
* The column names in pandas are taken from the data, versus Excel where columns are labelled using letters.
* Where there is a missing value in the original data, pandas has the placeholder **NaN** which indicates that the value is missing, or **null**.
* The sales data has a decimal point added to each value, because pandas stores numeric values that include null (NaN) values as numeric type known as **float** (this doesn’t effect anything for us, but we just wanted to explain why this is).

Let’s use the type() function to look at the type of our sales variable

*Python:*

*type(sales)*

*pandas.core.frame.DataFrame*



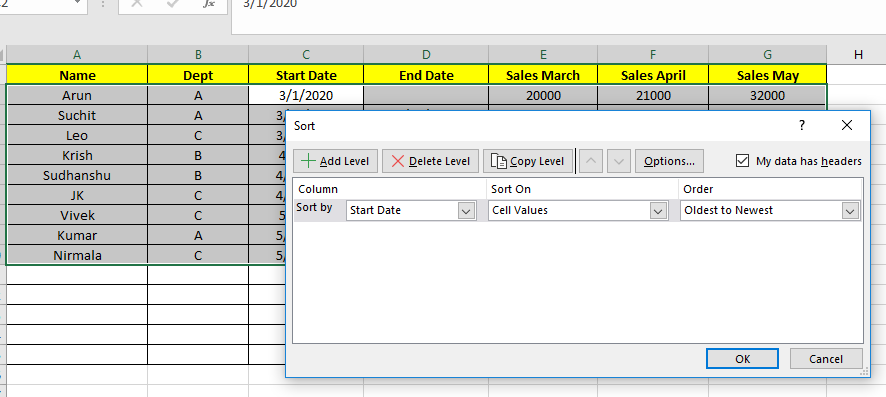
This output tells us that our sales variable is a **DataFrame** object, which is a specific type of object in pandas. Most of the time in pandas when we want to modify a dataframe, we’ll use special syntax called a data frame **method**, which allows us to access specific functionality that relates to the dataframe objects. We’ll see an example of that in a moment when we complete our first task in pandas

## Sorting Data

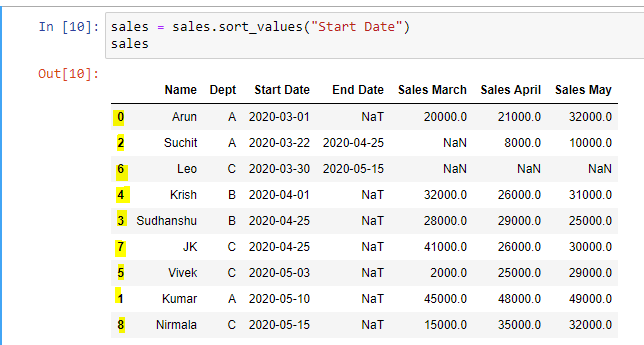
How to sort our data in Excel and Python

In Excel, if we wanted to sort our data by the "Start Date" column, we would:

* Select our data.
* Click the ‘Sort’ button on the toolbar.
* Select ‘Start Date’ in the dialog box that opens.



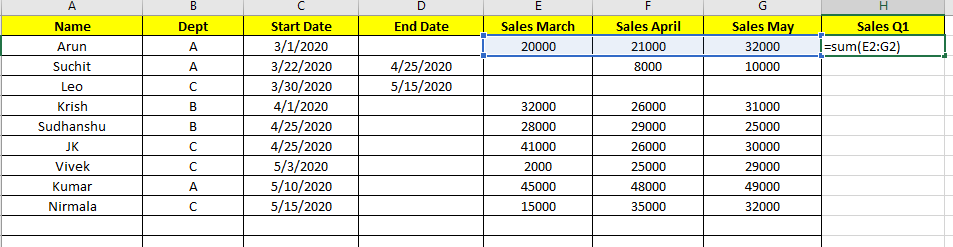
In Pandas, we use the DataFrame.sort\_values() method.

*sales = sales.sort\_values("Start Date")*

**Summing the Sales Values**

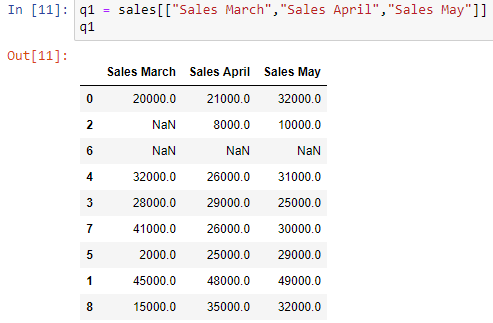
***Excel***

* Enter a new column name "Sales Q1" in cell H1.
* In cell H2, use the SUM() formula and specify the range of cells using their coordinates.
* Drag the formula down to all rows.



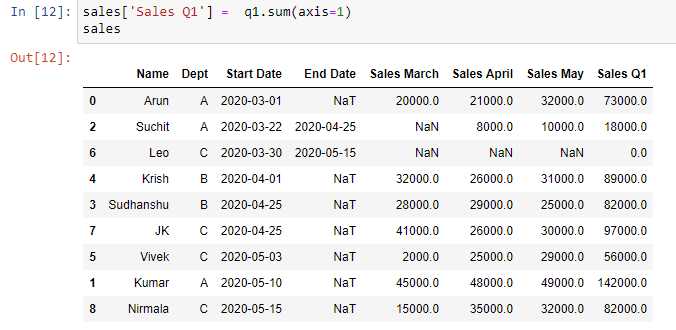
*Python:*

*q1 = sales[["Sales March","Sales April","Sales May"]]*



We’ll use the DataFrame.sum() method and specify axis=1, which tells pandas that we want to sum the rows and not the columns.

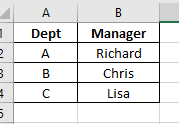
*sales['Sales Q1'] = q1.sum(axis=1)*



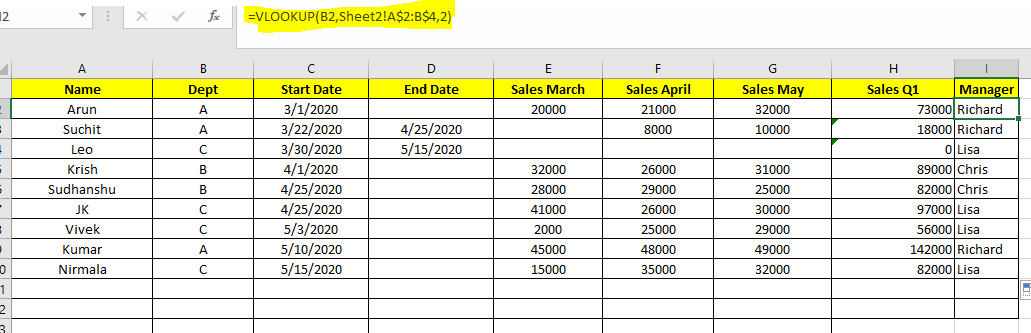
## Joining Manager Data

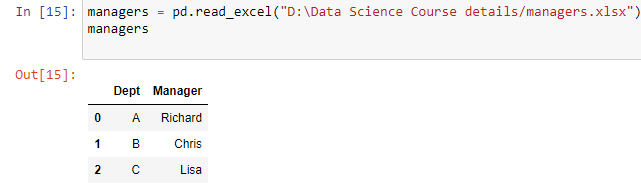
How to join this data in a "Manager" column in Excel and Python

Excel:



* start by adding the column name to cell I1.
* use the =**VLOOKUP(B2,Sheet2!A$2:B$4,2)**formula in cell I2, specifying:
  + to lookup the value from cell B2 (the Department)
  + in the selection of manager data, which we specify using coordinates
  + and that we want to select the value from the second column of that data.
* Click and drag the formula down to all cells.

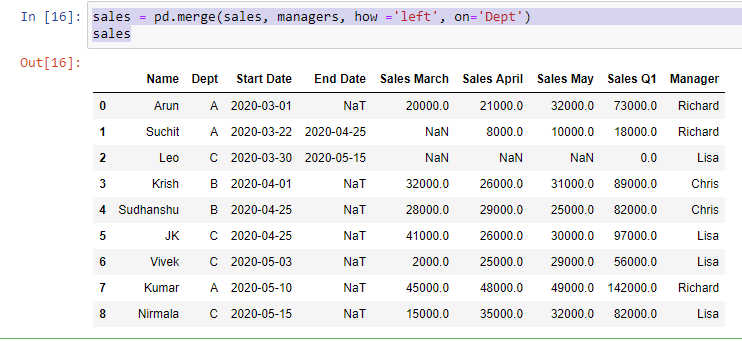




In order to join mangers data to sales using pandas, we’ll use the pandas.merge() function.

Pandas:

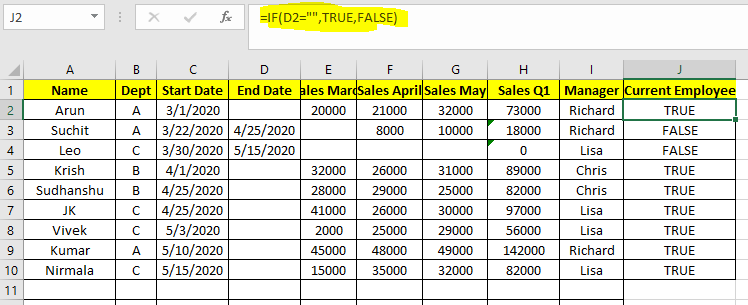
*sales = pd.merge(sales, managers, how ='left', on='Dept')*



## Adding a Conditional Column

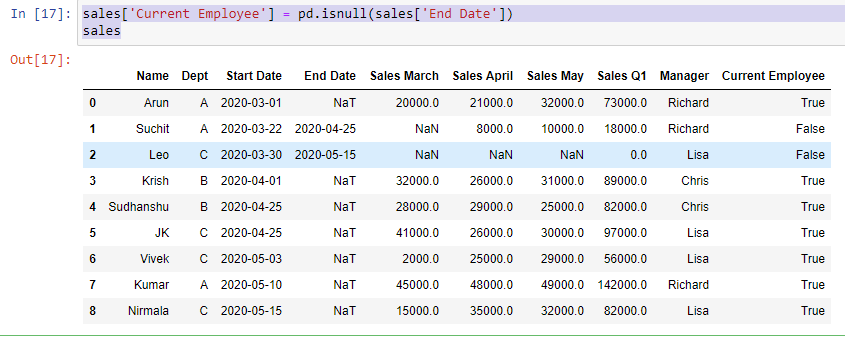
*Excel*

* Add a new column name to cell J1.
* Use the IF() formula to check if cell D1 (End Date) is empty, and if so fill J2 with TRUE, otherwise FALSE.



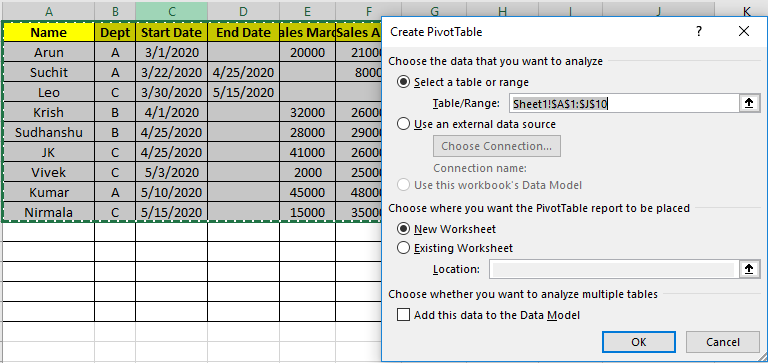
*Pandas:*

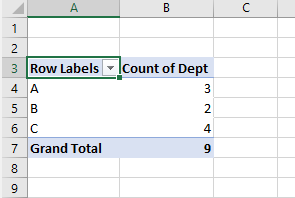
*sales['Current Employee'] = pd.isnull(sales['End Date'])*

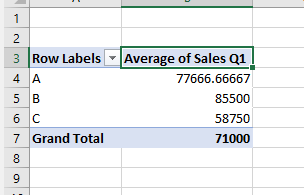


## Pivot Tables

*Excel:*

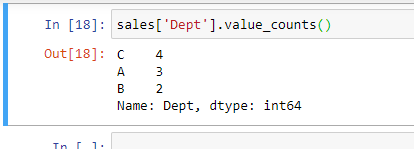






*Pandas:*

*sales['Dept'].value\_counts()*



to calculate this in pandas, we’ll use the DataFrame.pivot\_table() method. We need to specify some arguments:

* index: the column to aggregate by.
* values: the column we want to use the values for.
* aggfunc: the aggregation function we want to use, in this case 'mean' average.

*sales.pivot\_table(index='Dept', values='Sales Q1', aggfunc='mean')*

