University of Washington

iSchool Info 330

# Module 08 - Data-Driven Applications

In this module, we will look at various applications that use a database**. Our focus will be on reporting applications since these the most used.**

## Outline

Here is a general outline of what we will be doing this module:

|  |
| --- |
| **Module08: Data-Driven Applications** |
| Session01 Lectures and Labs < 110 mins |
| Reporting Applications |
| Excel Reports |
| Lab 1: Creating Excel Reports |
| Power BI Reports  Tableau Reports |
| Lab 2: Creating Tableau Reports |
| Session02 - Lab |
| Lab 4: Final Project - Milestone02 |
| Session03 Lectures and Labs < 110 mins |
| Python |
| Lab 3: Installing Python |
| Python Basics |
| Lab 4: Creating a Report with Python |
| Other Programming Languages |
| Ethics in Reporting |

# Session01 < 110 mins

In this session, we explore **how reporting applications are used with databases**. We will use start with two simple examples of reporting software by Microsoft's Excel and Report Builder.

## Reporting Applications

There are **many companies** building reporting applications. In fact, I would not be surprised if there are **over a hundred different reporting applications** out there. Most of them are **small** companies trying to get market share, but there are a lot of **large** companies that are also interested in getting people to use their software.

You can **specialize** **in a few reporting software applications and feel confident** that you will be able to **learn and use other reporting software** as needed.

That is because in general, they all need to follow the same pattern:

* Make a **connection** to the database
* Access send data via a SQL **select** statement
* Allow for **displaying** data in a table or graphic format.

An internet search will bring up **many different websites** directing you to their **proprietary** reporting Solutions. However, some websites **will let you review a collection of reporting applications**. On the surface, these **often** seem to be independent, some are, but **often** you'll find that they are **biased**.

"Find the best Reporting Software for your business. Compare product reviews and features to build your list."( <https://www.capterra.com/reporting-software/>, 2017)

### Microsoft

Microsoft has spent a lot of money creating and improving reporting applications and offers **many to choose**. Most applications' **features overlap** with each other, and Microsoft will likely combine some of them in the future. Currently, all of them may be considered valid options for reporting applications.

#### Excel

Excel is a spreadsheet application that includes **many reporting features**. Microsoft considers this a vital part of its BI applications stack. **Easy to learn** and earlier versions have been **used for more than a decade**. Excel must be **purchased** and installed on PCs and Mac and is designed to be a single user application.

#### Power BI

Power BI is a **more complex** piece of software that comes in **both free and paid-for versions**. The **free** desktop version **provides a lot of built-in functions,** more so than most other free versions of similar software in the industry.

## Excel Reports

Excel is Microsoft's frontline reporting software. Starting initially as a commercial spreadsheet application, it has become much more.

* Can display data in different forms such as in **charts, graphs**, or other types of reports
* Has programming features using "Visual Basic for Applications (**VBA**)"
* Reports are most commonly created by users with little to **no programming skills**
* Capable of creating reports from **many different data sources**

### Creating a Connection

To make a connection to Data:

1. Create or open an Excel spreadsheet
2. Look for the **Data tab** on the ribbon interface
3. The **get data** button and select the data source you want to use from the context menu
4. Advance through the **wizard** answering its questions

For example, if I wish to connect to Microsoft's SQL server, choose the option, **From database -> SQL Server Database.**

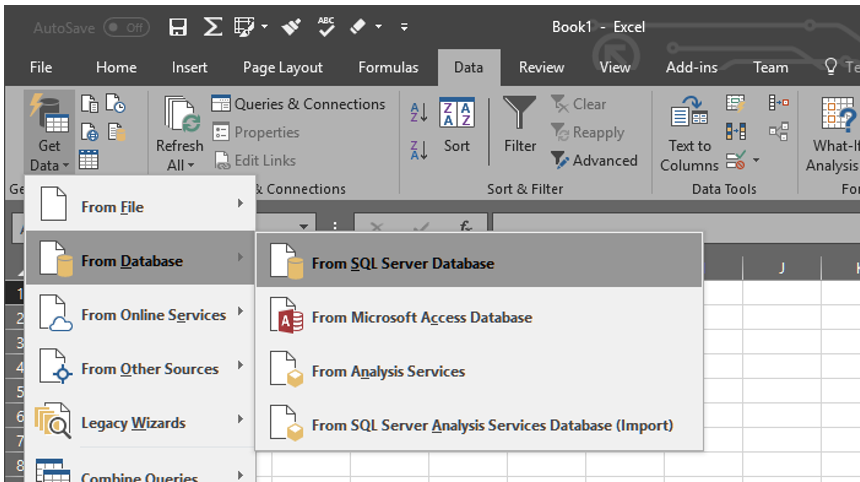
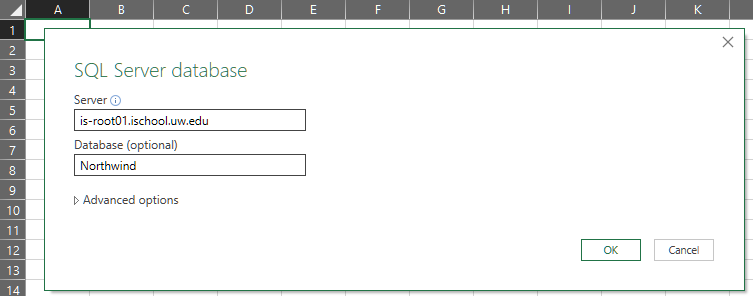


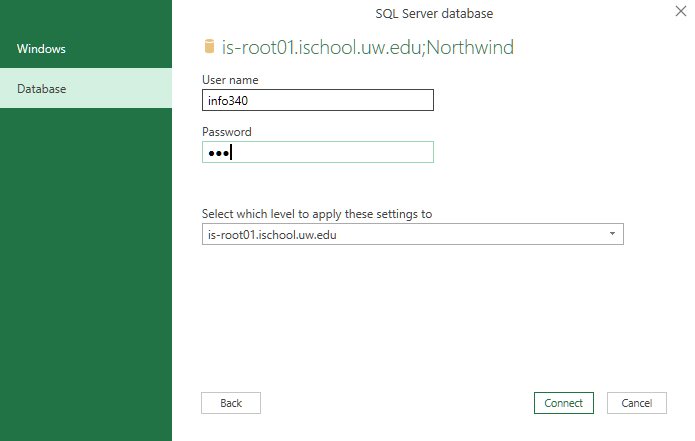
Figure: Using the Get Data button in Excel

At first, I would have to enter the name of the **server** and optionally the name of the **database** I wish to use.

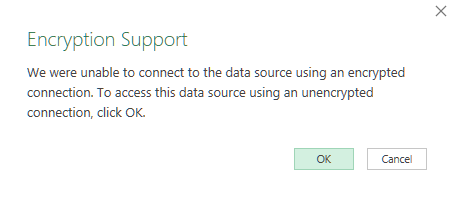


Next, choose the type of **authentication**, either Windows or database-based authentication.

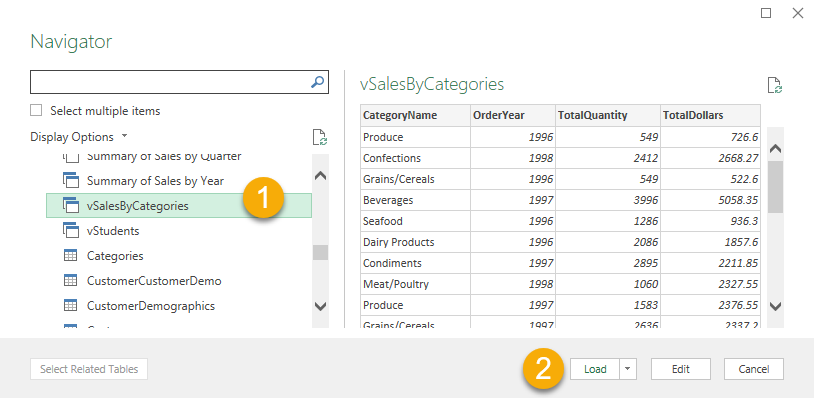
With Windows-based, I only need to indicate a Windows account, but with database authentication, I will need to enter in a username and password.



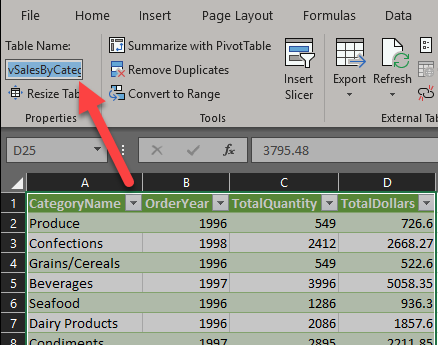
Microsoft's will warn you that the data will **not** be automatically **encrypted** as it passes from client to server unless you encrypt it yourself.



After that, you will be presented with a list of **objects to choose** from that contain reporting data.



After selecting an object and using the **load** button, you should see data in the spreadsheet.



**Important**: Note the name of the table for later use in the Tableau lab.

## Lab 1: Creating Excel Reports

In this lab, you will use Excel to create a report using data from the Northwind database, but through a view in your personal lab database.

You will work on your own for the first 10 minutes and then review the answers in the last 10 minutes.

**Note**: This lab should be done individually or in groups of three or less.

### Step 1: Create the database

Run the following SQL code into a code window, then modify it to use your own name instead of "YourNameHere." Afterward, execute the code to make the Module 06 labs database.

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*--

-- Title: Mod08 Labs Database

-- Author: YourNameHere

-- Desc: This file creates reporting structures

-- Change Log: When,Who,What

-- 2017-01-01,YourNameHere,Created File

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*--

Use Master;

go

If Exists(Select Name from SysDatabases Where Name = 'MyLabsDB\_YourNameHere')

Begin

Alter Database [MyLabsDB\_YourNameHere] set Single\_user With Rollback Immediate;

Drop Database MyLabsDB\_YourNameHere;

End

go

Create Database MyLabsDB\_YourNameHere;

go

Use MyLabsDB\_YourNameHere;

go

### Step 2: Create a reporting view

Run the following SQL code, understand what it does, and then use it to create a reporting view called, "vSalesByCategories";

Create View vSalesByCategories

as

Select

[CategoryName]

,[OrderYear] = Year(o.OrderDate)

,[TotalQuantity] = Sum(od.Quantity)

,[TotalDollars] = Sum(od.UnitPrice)

From Northwind.dbo.Categories as c

Join Northwind.dbo.Products as p

On c.CategoryID = p.CategoryID

Join Northwind.dbo.[Order Details] as od

On p.ProductID = od.ProductID

Join Northwind.dbo.Orders as o

On od.OrderID = o.OrderID

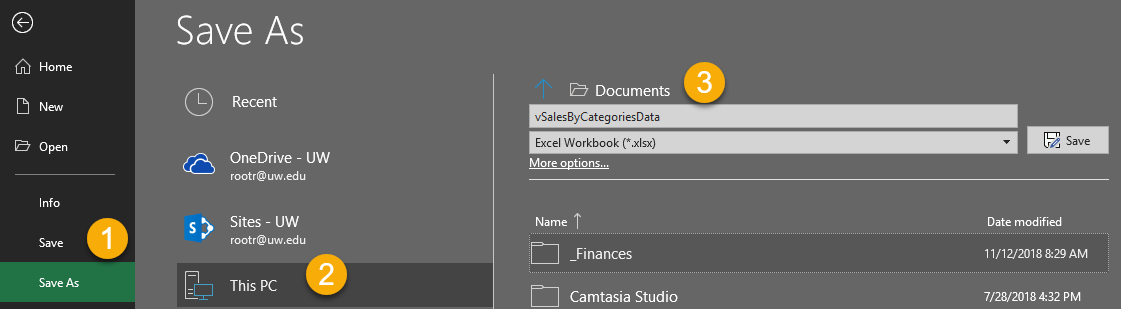
### Group By c.CategoryName, Year(o.OrderDate)

### Step 3: Create an Excel report

Create an Excel report that connects to your database and displays data from your new view.

### Step 4: Save Your File

Save this file in an easy place to locate, like the Documents folder. We will use this Excel data in the Tableau lab later.



### Step 4: Review Your Work

Now, you will review your work with your instructor.

## Power BI Reports

"Power BI is **a suite** of business **analytics tools** that deliver insights throughout your organization. Connect to hundreds of data sources, simplify data prep, and drive **ad hoc analysis**. Produce beautiful reports, then publish them for your organization to consume on the **web and across mobile devices**. Everyone can create personalized dashboards with a unique, 360-degree view of their business. And scale across the **enterprise**, with governance and security built-in." (<https://powerbi.microsoft.com/en-us/>, 2017)

It is hard to find good information about the different components that make up the Power BI family of applications. That seems to be due to marketing and rapid changes. However, here is a list of the general components:

#### Power BI Service

Power BI web service is a cloud-based application that allows subscribers to **store, manage, and view Power BI reports and custom components**.

#### Power BI Web Site

The Power BI website acts as a **front-end to the Power BI cloud** web service.

#### Power BI Desktop

The Power BI Desktop application allows users to **create reports that can be viewed with the desktop application or uploaded to the Power BI web service**. Once uploaded to the Power BI web service, these reports can be accessed through the Power BI website, several Microsoft applications, and custom applications accessing the web service.

#### Power BI Mobile

The Power BI mobile application allows users to **view and interact with Power BI reports** from tablets and cellphones.

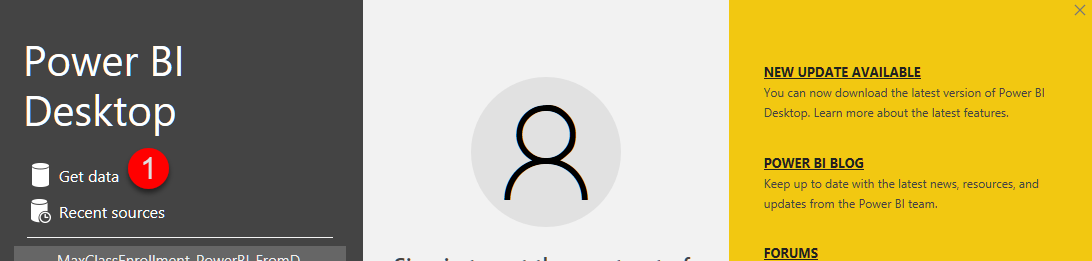
#### Power BI Reporting Server

The Power BI Reporting Server is an on-premise server to **store and manage reports without a cloud**.

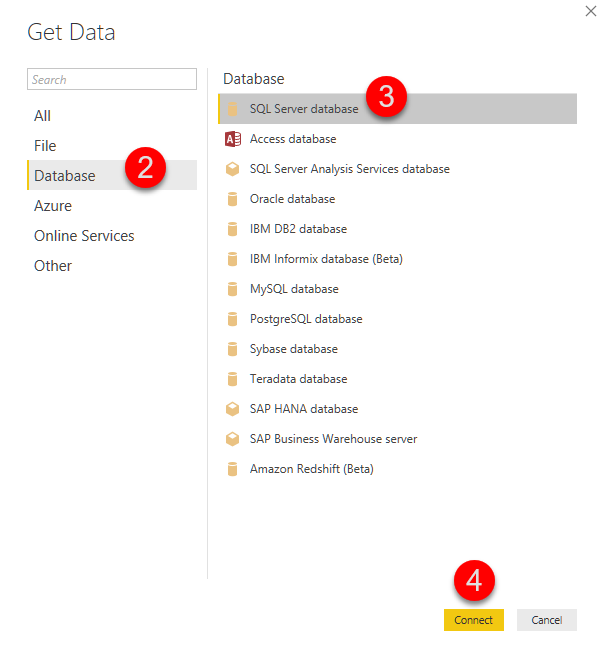
### Creating a Power BI Report

"The **3 major building blocks** of Power BI are: **dashboards**, **reports**, and **datasets**. You can't have dashboards or reports without data (well, you can have empty dashboards and empty reports, but they're not very useful until they have data), so let's start with datasets." (<https://docs.microsoft.com/en-us/power-bi/service-basic-concepts#power-bi-concepts>, 2017)

When you first **open the Power BI desktop** application, you will see a **splash screen** that includes a "**Get data**" icon. clicking this icon will launch several dialogue windows that will help you create a new report.

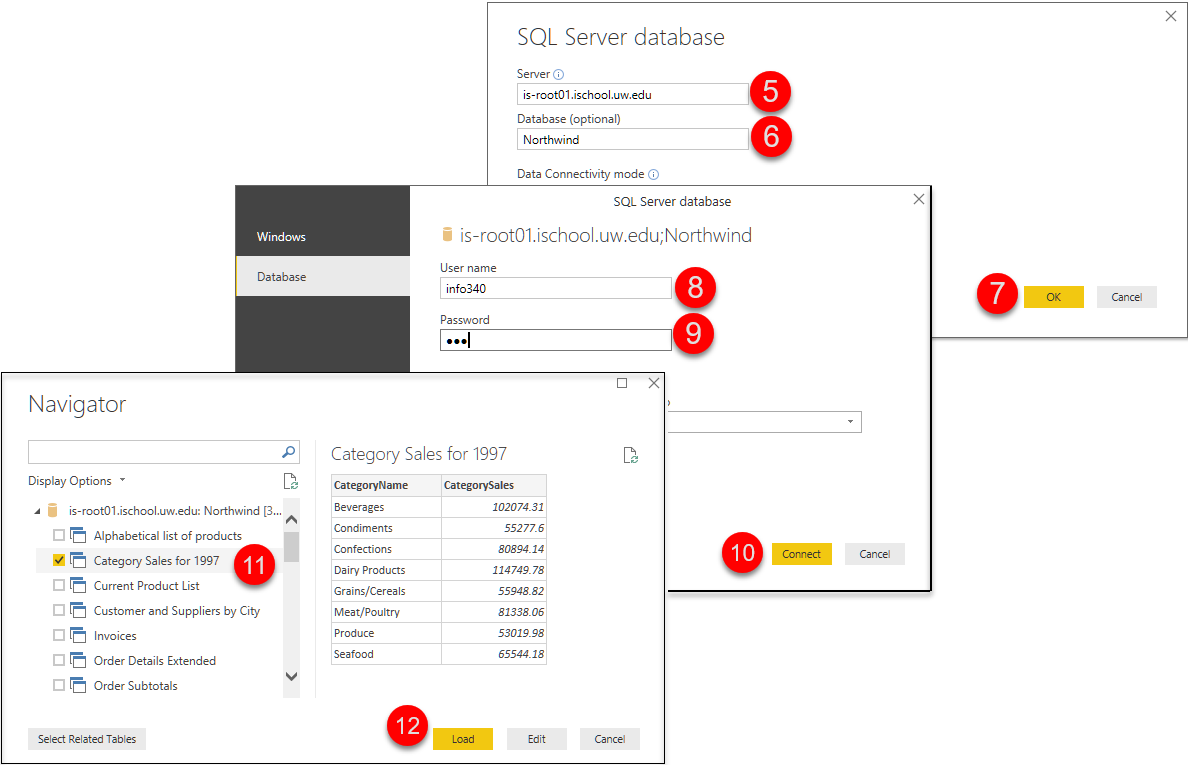


The get data dialogue window allows you to **choose between many different data sources**. More data sources are being added all the time by Microsoft and custom ones can be created as well. For our examples we will be using a SQL Server database connection. To make this selection choose from the **database** category on the **left** side of the dialogue and the **SQL Server** database option on the **right**.

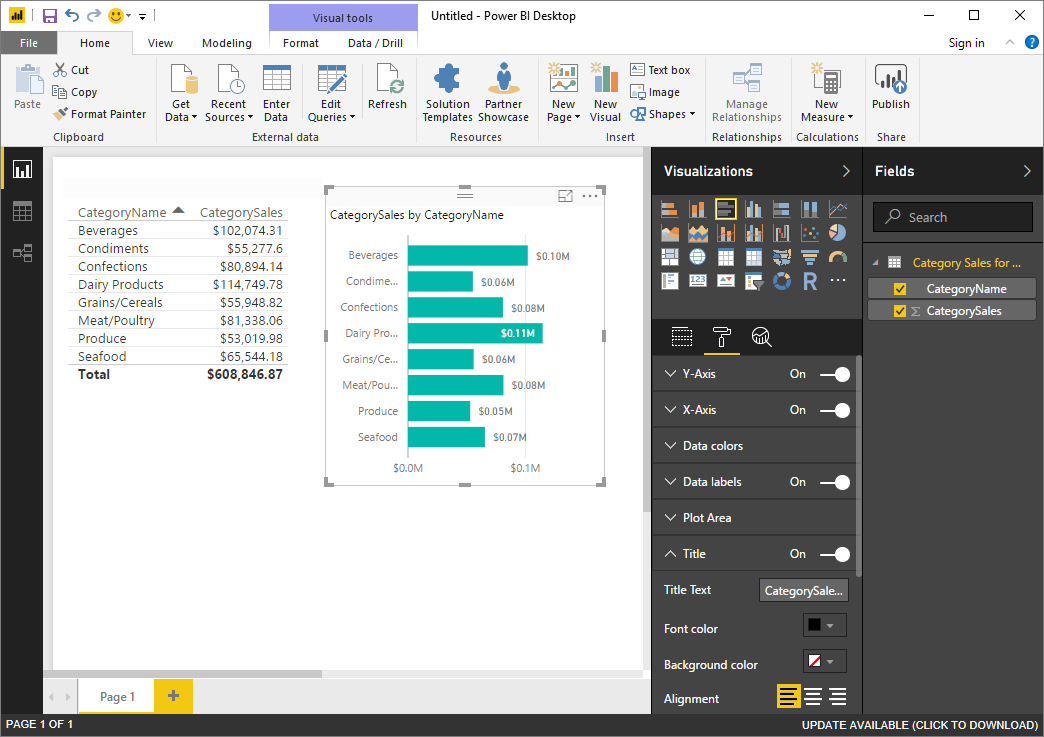


The next several **screens look like** the database **connector in Excel**. This because Power BI functionality was integrated into Excel's most recent versions.

You would configure these screens by **filling in** the name of your **server**, the name of your **database**, the **username** and **password**, and the **database object** or query you wish to use for your reports data.



Upon using the "**Load**" button on the last dialogue window, you will be presented with the Power BI editing tools. Here you will be able to **change the fonts**, and **other visual** aspects of your report.



## Non-Microsoft

There are plenty of other companies out there providing reporting software. Some examples include:

* IBM **Cognos**
* **Tableau** Business Intelligence
* **SAS** Business Intelligence
* SAP **Business Objects** Web Intelligence
* SAP **Crystal Reports**

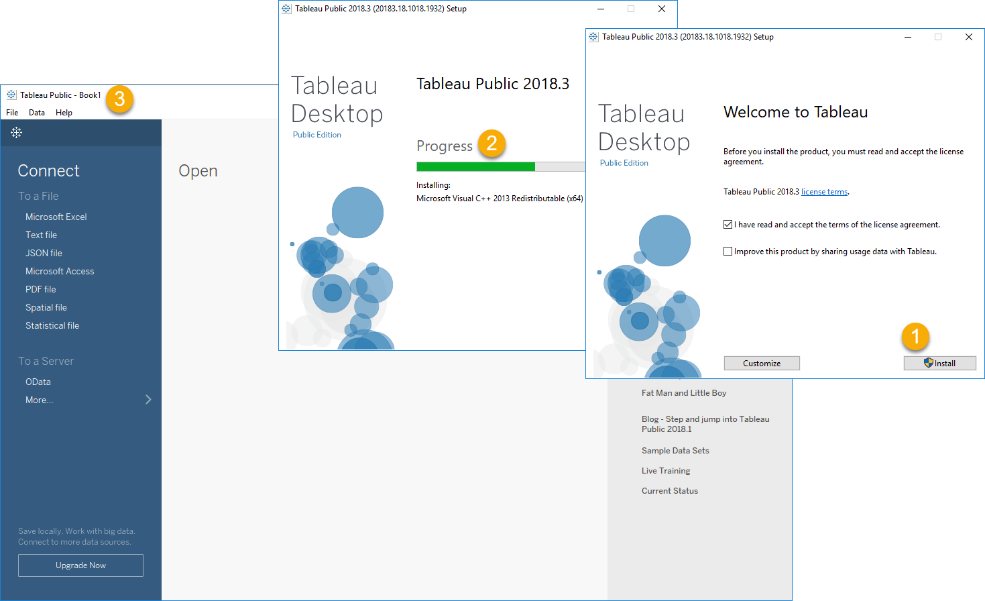
### Tableau Reports

**Tableau** **has become the most popular reporting tool out there**, followed by Microsoft's Power BI for its early use of cloud-based reporting. **Still**, the **most common** reporting application you are likely to see is Microsoft's **Excel due to it being readily available on most users' computers!**

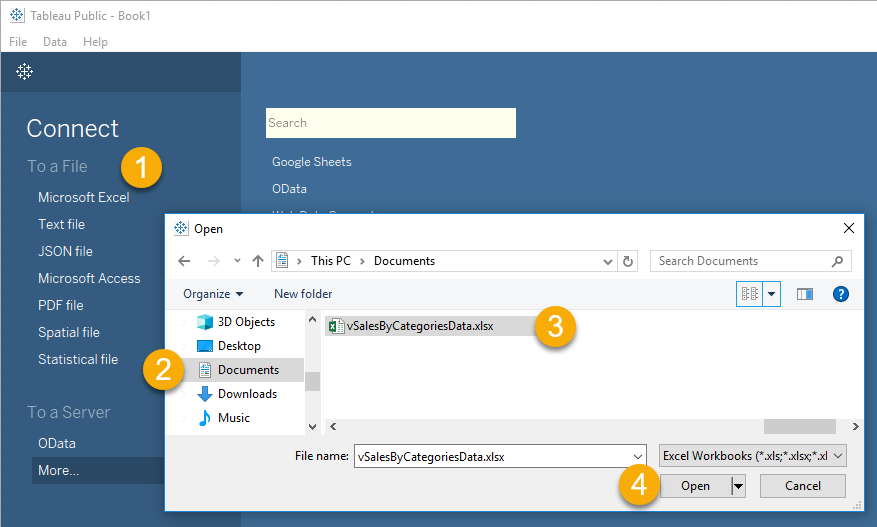
"Create and share interactive charts and graphs, stunning maps, live dashboards and fun applications in minutes, then publish anywhere on the web. Anyone can do it, it's that easy—and it's free." (<https://public.tableau.com/en-us/s/download>, 2018)

**Important:** Unlike Microsoft's PowerBI, the **Tableau public edition does not let you save your work as a file or connect directly to a database**, so you have to pull data from a file and save your report using a screenshot!

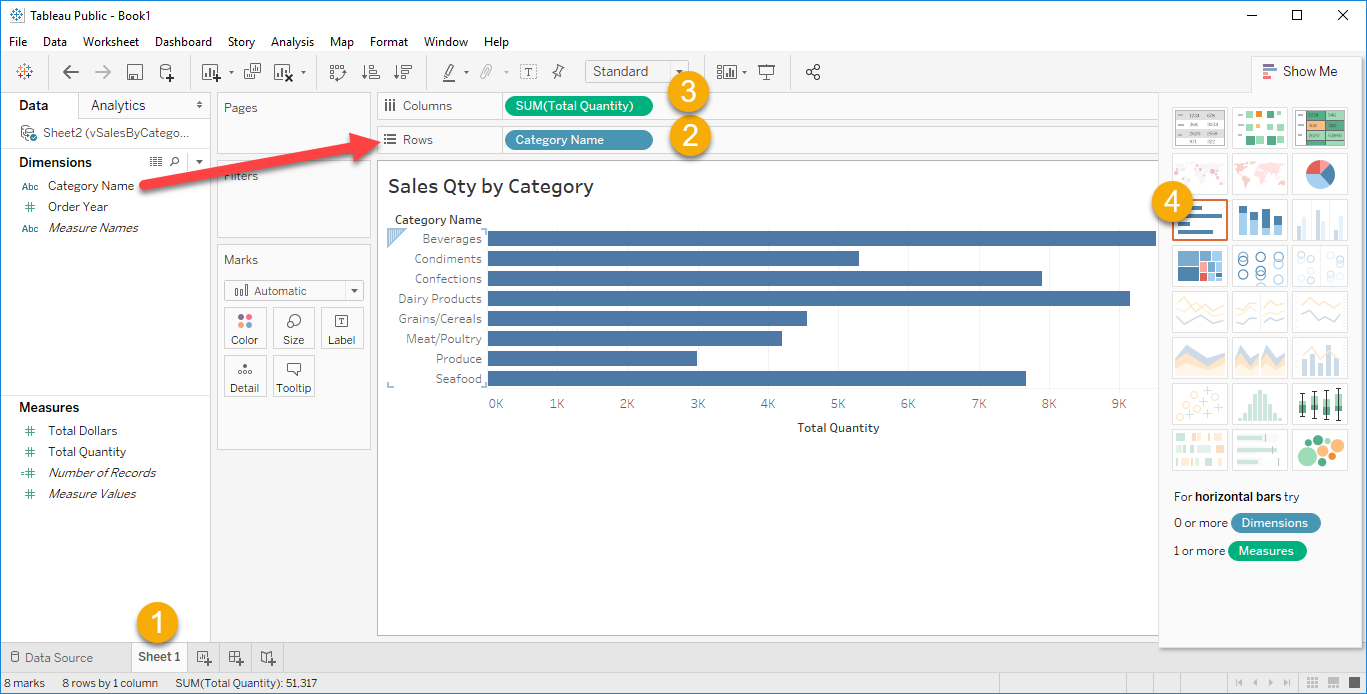
You can download Tableau Public edition from their website and start the Installer. When it finishes it should launch the application, so you can start build reports right away!



The first step in building a report is to connect to a data source. Tableau public does not let you connect to a SQL Server database (You need to pay more to do that!), but it can connect to both Excel and Access.

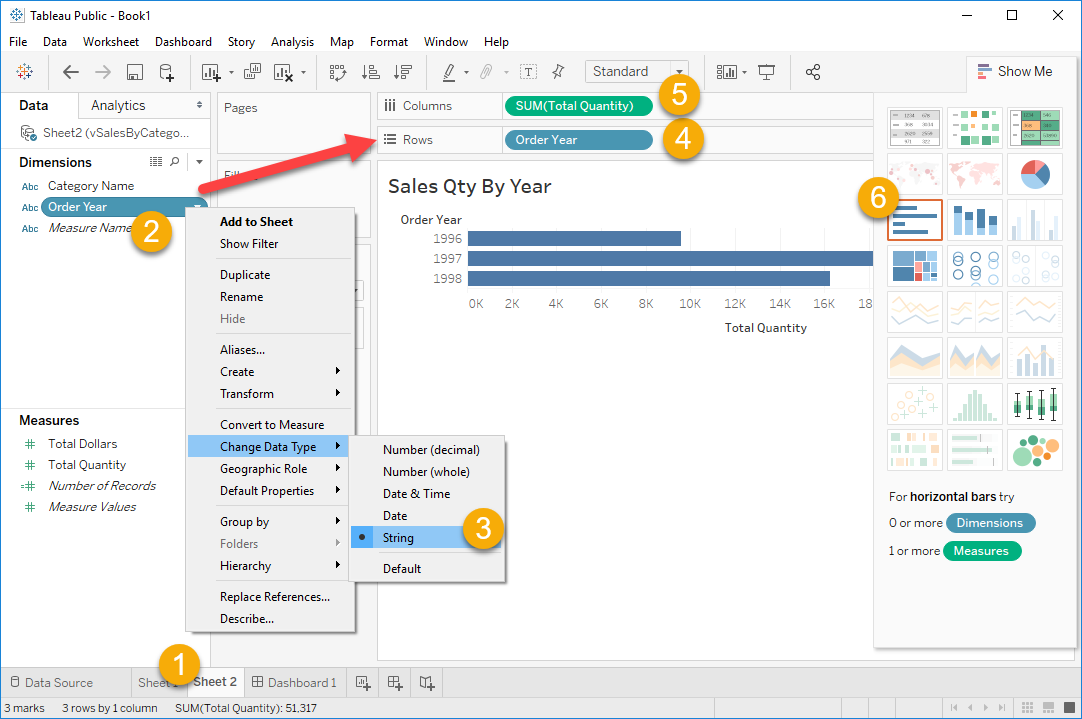


Once you have connected to a data source you configure a Tableau's Book's Sheet to create reporting objects. You can create many different sheets in a single Book. Each can show a different aspect of the data.

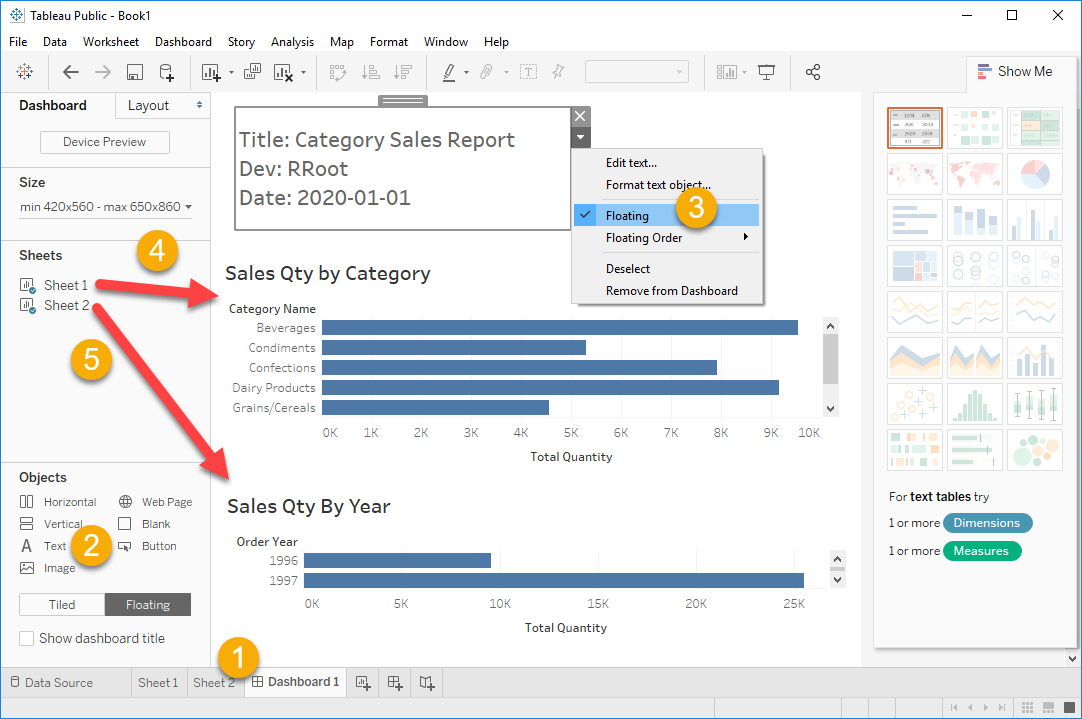


***Tip:*** *The use of Dimension and Measures are terms we saw when we looked at the Data Warehouse design.*

Tableau automatically chooses data types for your dimensions and measures. Sometimes you will need to change these data types to create the reports you want.



Individual Sheets are collected into Dashboards. Dashboards can also contain other objects, like a Text area where a report header can be added.



## Lab 2: Creating Tableau Reports

In this lab, you will use Tableau to create a report using data from the Excel Spreadsheet.

You will work on your own for the first 10 minutes, then we will review the answers together in the last 10 minutes.

**Note**: This lab should be done individually or in groups of three or less.

### Step 1: Install Tableau

Download Tableau Public and Run the installer program and follow the prompts to install the Tableau Software.

### Step 2: Create a Tableau report

Create a Tableau report that connects to your Excel Spreadsheet and displays data from the "vSalesByCategories" view.

### Step 3: Review Your Work

Now, you will review your work with your instructor.

**Important:** Unlike Microsoft's PowerBI, the **Tableau public edition does not let you save your work as a file or connect directly to a database**, so you have to pull data from a file and save your report using a screenshot!

# Session02 Lectures and Labs < 50 mins

In this session you will continue to work on the final. You will work on your own for the whole 50 minutes of this lab but may ask questions whenever you would like help.

# Session03 Lectures and Labs < 110 mins

# Custom Applications

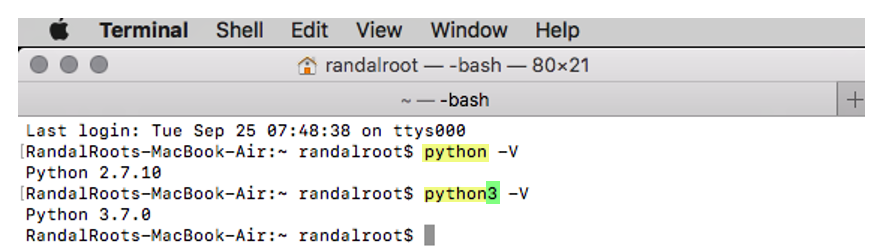
In addition to using Pre-made **database applications**, you can **create your own using languages**, like Microsoft's **C# or** Open Source **Python**. **Both** languages are **simple to learn**, but **Python** is more forgiving, while C# has more advanced programming options. **Both** languages now **run on Windows, Linux, and Mac OS**. **Both** languages are **Free** to use for production development.

In this session, you will learn **more about custom data-driven applications** by looking at examples using Python scripting and C#. Lastly, we will talk briefly about ethics in reporting, a subject that you will cover more during your studies.

## Python

Python runs on **Windows, Linux/Unix, Mac OS X**. Python is **free** to use, even for commercial products, because of its OSI-approved open source license.

There are two main versions of Python: *"New to Python or choosing between Python 2 and Python 3? Read Python 2 or Python 3."* -- Python Programming Language – Official Website <http://wiki.python.org/moin/Python2orPython3>

Mac already has 2.x installed. I recommend installing and use 3.x. You will just have to remember to use the correct version when you are running your scripts (run "python.exe" for 2.x and run "python3.exe" for 3.x

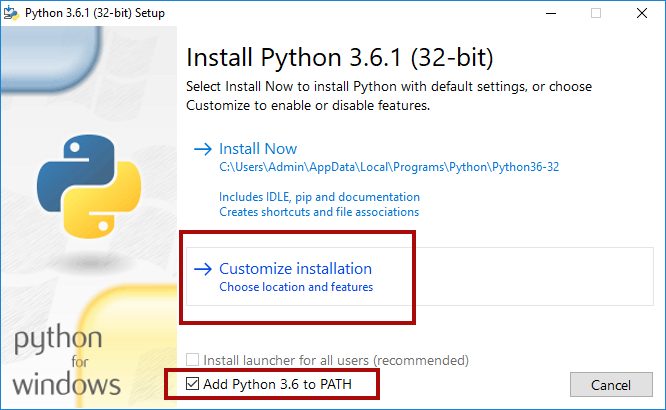
### Installing Python

To create Python scripts, you **must install Python**.

Although installation is easy, you may need to search for video tutorials based on your chosen OS. <https://www.google.com/search?q=How+to+install+python> (opens an external site)

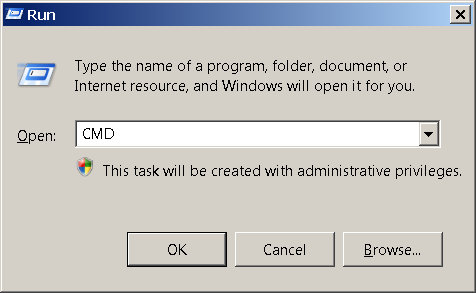
#### Windows

On Windows, I recommend installing it in a **custom, easy to access, location**. I also recommend including the Python executable in the **OS path**.

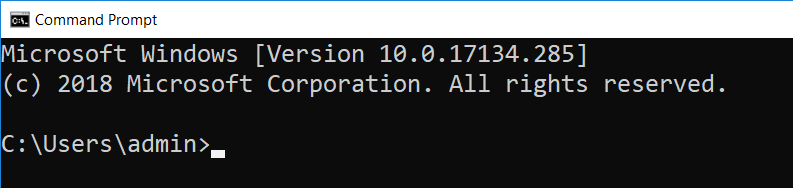


**Once you have installed Python,** you can create a program that runs as a Console application in the OS Command Console/Terminal.

To open a command console in **Windows**, use the Start Menu ➤ Run (windows key + r) and type in the command "CMD".

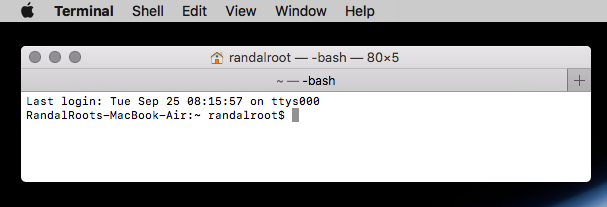


You will then be presented with a command prompt in the command console window that looks like the following image.



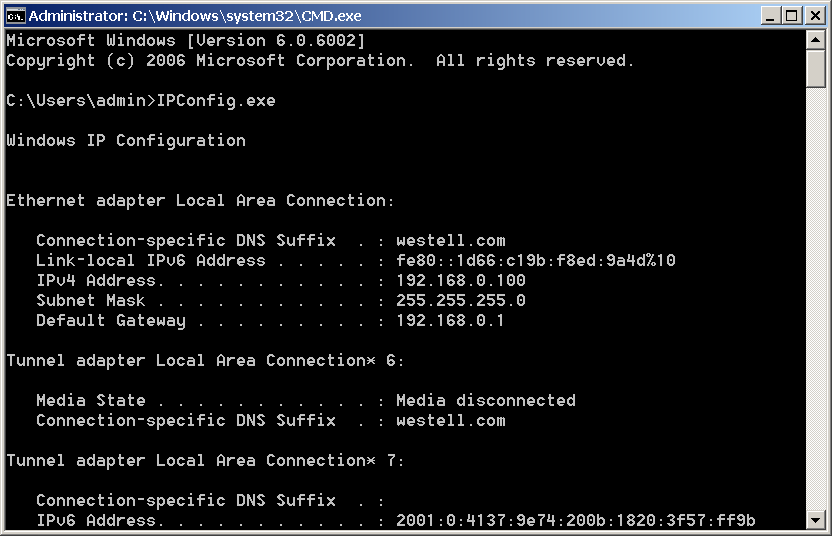
#### Mac

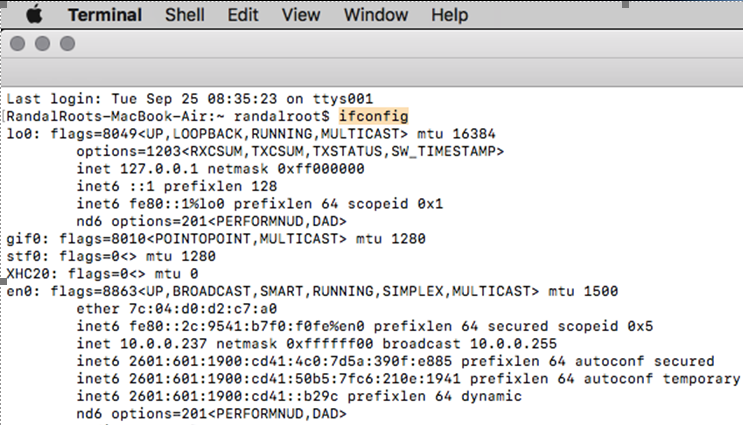
If you are using a **Mac**, this it is almost the same, but now it is called a Terminal window. Open a terminal window using **Finder > Applications > Utilities > Terminal.app** and you will see something like Figure 5.



With the Command Console/Terminal open you can run "Console Applications". These applications are not fancy, but they do allow you to accomplish useful tasks on a computer with the minimal fuss as to making your application look nice! IPConfig.exe is a good example of a Console application.

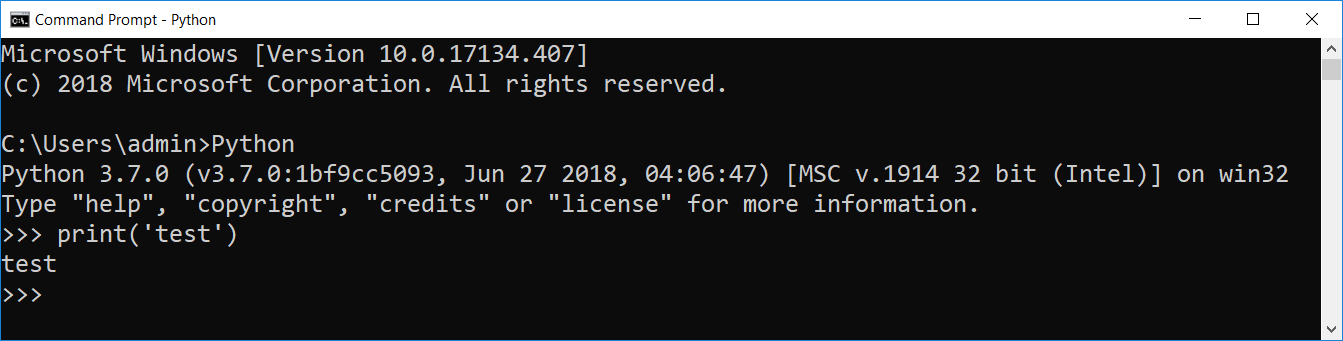
To see what it does, type in the command "**IPConfig.exe**" on the Windows OS and hit the Enter key to run the application or" **ifconfig"** on the **Mac** OS.

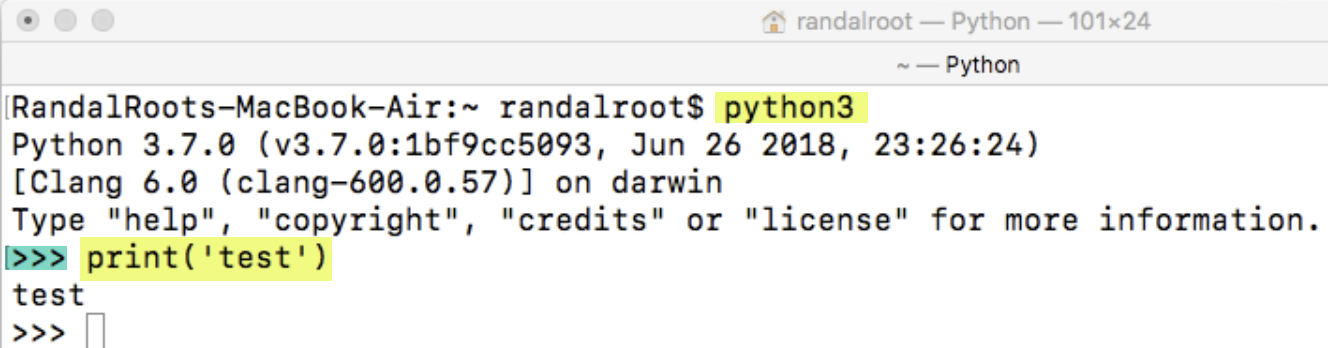




#### Both Windows and Mac

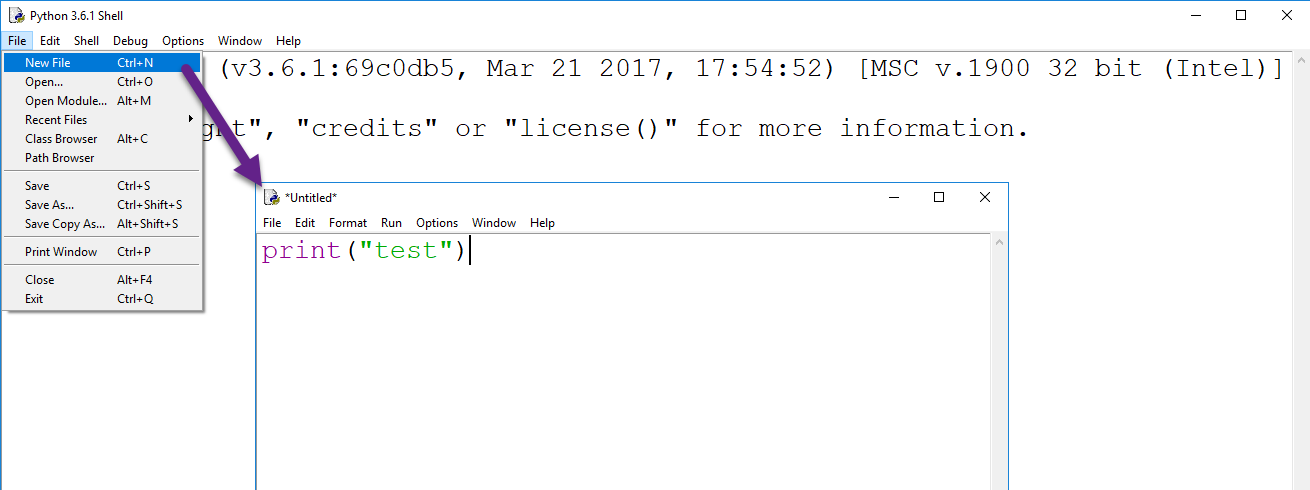
To run Python code, you would open the command/terminal window, type in the command "Python" (or "Python 3") wait for it to open with a >>> symbol. Afterward, type and execute Python commands.



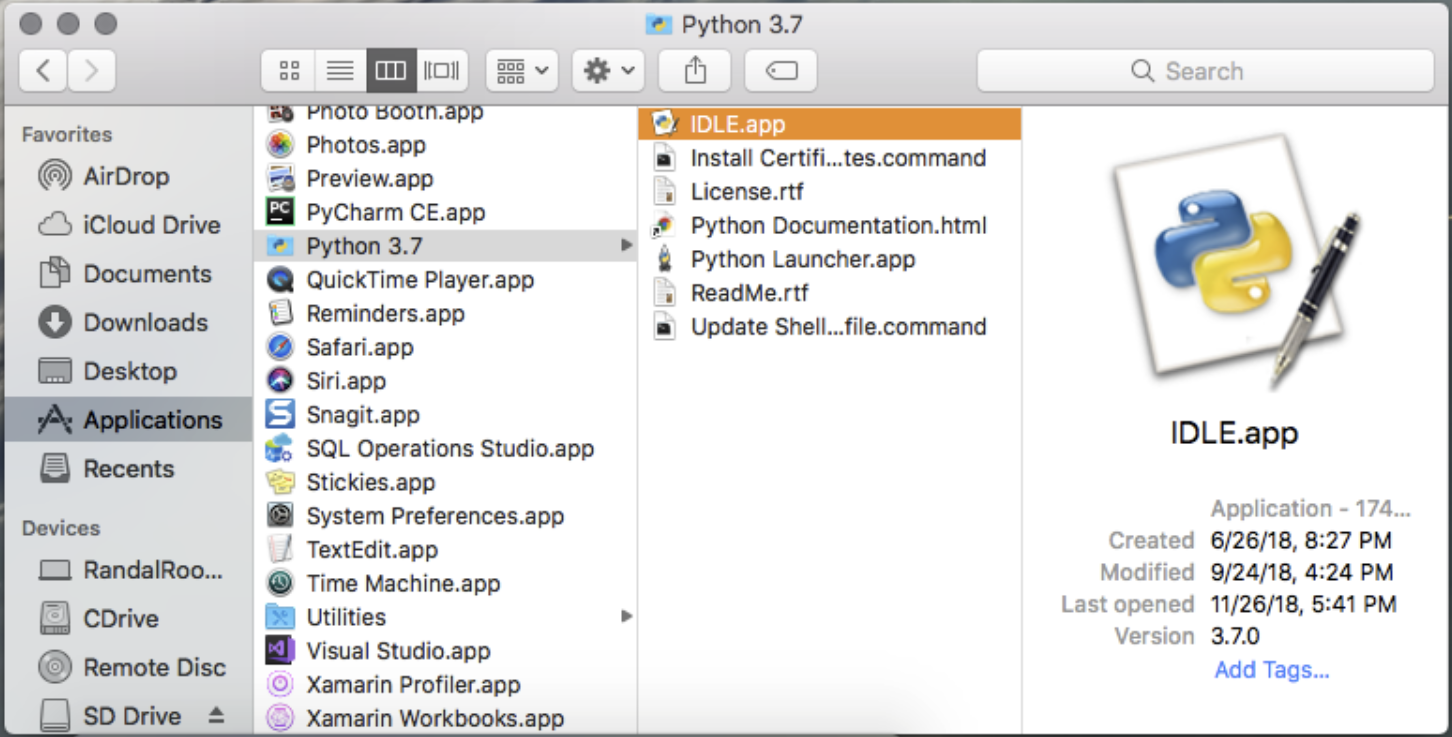


### Python Editor

On both **Mac and Windows**, the Python installation includes an **Integrated Development Environment (IDE)** used to write and test your code **called IDLE**.



On Mac, remember that there are two versions of Idle installed **and make sure to open the 3.x version.**



## Lab 3: Install Python

In this lab, you will install Python and test that the editor works..

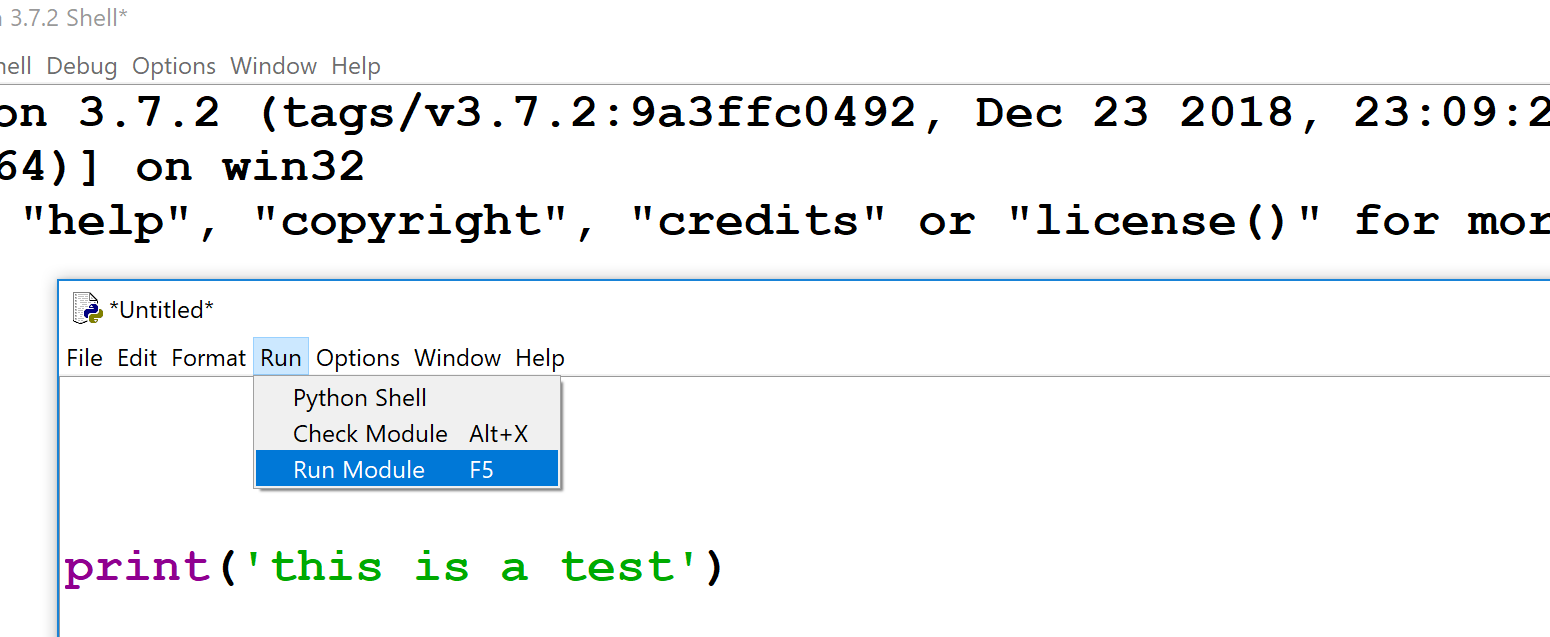
**Note**: This lab should be done individually or in groups of three or less. This lab can be done with Window or Mac OS.

### Step 1: Install Python

Download Python 3.x and Run the installer program. Follow the prompts to install the software.

### Step 2: Test the Idle Editor

Start the Idle editor and test that you can run the code shown in the following image:



## Python Basics

In a very general way, programs break down into two distinct categories:

* ***Data:*** the information you want to work with, such as a person's name and phone number.
* ***Operations:*** are the things you want to do with the data, such as printing out the data or adding two numbers together.

Program code consists of *Statements*, *comments*, *namespaces*, *directives*, and many other categories. Here are some examples from the Python language:

* ***Statements:*** A commands you add to a code file. A ***statement***is one instruction to the computer.

x = 4 *# This is one statement,*y = 5 *# this is another,*z= x + y *# and another as well*

* ***Comments: Single line***

**#** This is a standard, *inline*, Python comment.

* ***Comments: Multi-line***

**'''**  
C Style languages use a slash-star and star-slash pair for a block comment  
, but Python uses triple quotes  
int x = 5;  
int y = 10;  
**'''**

### Case-Sensitivity

Python is a **case-sensitive** language, so you must be careful as you type.

x = 4 *# This places the value of four into a variable called x***X** = 13 *# But, this places the value of four into a variable called X !*print(**X**) *# displays the value 13 to the user***PRINT(X)** *# this command is not understood by Python*

### Functions

***Function*** *in Python are* ***made using*** the define command, "**def**." After you create a function, you can run its group of statements by *calling* it.

**def** DemoFunction():  
 print(**"This is a statement in DemoFunction"**)  
 print(**"This is another statement in DemoFunction"**)  
*#End DemoFunction*

### The Main Script Body

Most applications run a "Main" script body as soon as a program is started. Within the Main script body, any code you type in will be processed one line after the other. If you call a function from the Main, it will jump to that function, run the statements inside, and return to the "Main body" when it is done. The example below outlines the order in which your statements will be processed.

**def** DemoFunction(): *# 3) jumps to here and run both statements…* print(**"This is a statement in DemoFunction"**)  
 print(**"This is another statement in DemoFunction"**)  
*#End DemoFunction  
  
# 1) Start of Main*print(**"This is a statement in the invisible Main body"**)  
DemoFunction() *# 2) call the method DemoFunction()...  
# 4) jumps back to here…*print(**"This is another statement in the invisible Main body"**)  
*# 5) End Main (the program ends! )*

### The print() Function

The **print()** function was created in Python to **print out information to the command window.** If you remember, IPConfig.exe wrote out its data to the command window **for a human user** to read.

**Note:** In Python 2.x you use the **print** function cannot use parentheses!

### The input() Function

The input() function "**pauses**" the program **to get data from the user**. It is also used to pause a script until the user presses the Enter key.

**Note:** In Python 2.x you use the **raw\_input()** function instead of the input() function to avoid and error!

### while loop

The While loop is the most common loop in programming. It will look and work very much the same in any language you decide to learn.

"The while statement is used for repeated execution as long as an expression is true:" -- Python help files

Using a counter to stop the loop from continuing indefinitely is the most common pattern for a while loop.

intCounter = 0

**while**(**intCounter < 3**):

print(intCounter)

intCounter = intCounter + 1

Another common pattern is using a "Flag" value to stop the loop.

strUserInput = input("Type in a string to echo (Enter 0 to quit!)")

**while**(**strUserInput != "0"**): #Make sure to use Quotes!!

print(strUserInput)

strUserInput = input("Type in a string to echo (Enter 0 to quit!)")

### The for loop

We have seen the while loop, but another type of loop available in Python is the "for" loop.

*"Python's for statement iterates over the items of any sequence (a list or a string), in the order that they appear in the sequence" -- Python help files*

Here is an example:

#Declare my variables

values = [1,2,3]

item = None

for item in values:

print(item) #Output the values

### Connecting to a File

You can connect to a data file using code that look like this:

objFile = open("test.txt", "**w**")

objFile.write("test" + "\n")

objFile.close()

Data can be read from the file using code like this:

objFile = open("test.txt", "**r**")

strData = objFile.read()# read() reads all the data at once

print(strData)

objFile.close()

You can read a single line at a time like this:

objFile = open("test.txt", "**r**")

strData = objFile.readline()# readline() reads a line of the data

print(strData)

objFile.close()

## Lab 4: Creating a Report with Python

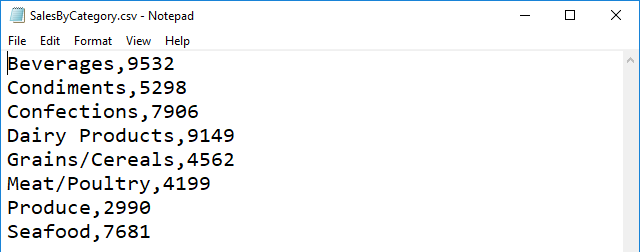
In this lab, you will use Python to create a report using data from the Northwind database, but through a view in your personal lab database.

You will work on your own for the first 5 minutes, and then we will review the answers together in the last 5 minutes.

**Note**: This lab should be done individually or in groups of three or less. This lab can be done with Windows or Mac OS.

### Step 1: Review a CSV File

Open the provided "Module08 Database Applications\Labs\SalesByCategory.csv" file using a simple text editor. Note the contents of the file.



### Step 2: Create a Report using Python Script

Create a simple console report that connects to the data file and displays data from the "SalesByCategory.csv" file using this code and the **Idle** integrated editing environment (IDE).

#-------------------------------------------------#  
# Title: <Type the name of the script here>  
# Dev: <Type your name here>  
# Date: <Type the day this script was first created>  
# Desc: <Type a description of the script>  
# ChangeLog: (Who, When, What)  
# <Example: RRoot, 01/15/2020, Added more code>  
#-------------------------------------------------#  
  
#-- Data --#  
# declare variables and constants  
FileName = **"SalesByCategory.csv"**RowOfData = {}  
TableOfData = []  
  
#-- Processing --#  
# perform tasks  
objFile = open(FileName, **"r"**)  
**for** line **in** objFile:  
 Data = line.split(**","**) # readline() reads a line of the data into 2 elements  
 RowOfData = {**"CategoryName"**: Data[0].strip(), **"TotalQty"**: Data[1].strip()}  
 TableOfData.append(RowOfData)  
objFile.close()  
  
#-- Presentation (I/O) --#  
print(**"\*\*\*\*\*\*\* Category Sales Quantities \*\*\*\*\*\*\*"**)  
print(**'Category Name (Total Quantity)'**) # adding a new line  
**for** Row **in** TableOfData:  
 print(Row[**"CategoryName"**] + **"("** + Row[**"TotalQty"**] + **")"**)  
print(**"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**)  
input(**"Press Enter to Exit!"**)

***NOTE:*** *We will look at* ***connecting and working with databases in Module09****.*

### Step 3: Review Your Work

Now, you will review your work with your instructor.

### Connecting to a SQL Server Database

You can **download and installed Python modules** that allow you to connect to databases. One that connects to databases is called **PyPyODBC**. (<https://pypi.python.org/pypi/pypyodbc>)

#### PIP

To install modules, you use the Python package installer (pip). This is updated so can get the latest version of pip like this:

$ python3 -m pip install --upgrade pip

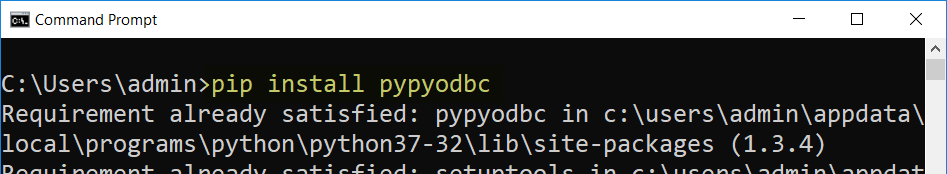
To use pip to install a package, you invoke it with this command:

python3 -m pip install the\_name\_of\_the\_package

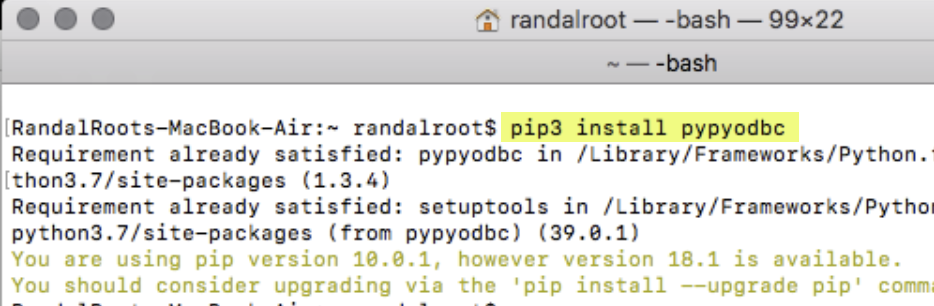
***NOTE: In Windows OS it is just Python, not python3***

#### Installing PyPyODBC

In a **Windows OS** you open can command window and run "**python -m pip install pypyodbc**" or "**python3 -m** **pip3** **install pypyodbc"** on **Mac OS**.



On **Mac OS**, you open a terminal window and run the command "**pip3 install pypyodbc**"



***Note: You can get more information here:***

* Windows -<https://pip.pypa.io/en/stable/user_guide/>
* Mac - <https://github.com/jiangwen365/pypyodbc/wiki/How-to-use-pypyodbc-on-MacOS-OSX>

#### Mac Issues

On Mac, things get a bit complicated! You need to perform a few additional installations to get things to work!

"This article explains how to install the Microsoft ODBC Driver for SQL Server on Linux and macOS, as well as the optional Command-Line Tools for SQL Server (bcp and sqlcmd) and the unixODBC Development Headers." (<https://docs.microsoft.com/en-us/sql/connect/odbc/linux-mac/installing-the-microsoft-odbc-driver-for-sql-server?view=sql-server-2017>, 2018)

##### HomeBrew (<https://brew.sh/>)

"Homebrew is a free and open-source software package management system that simplifies the installation of software on Apple's macOS operating system. Originally written by Max Howell, the package manager has gained popularity in the Ruby on Rails community and earned praise for its extensibility." ([Wikipedia](https://en.wikipedia.org/wiki/Homebrew_(package_management_software)), 2018)

To install on Mac OS, perform these commands in a terminal:

1. Install HomeBrew.

/usr/bin/ruby -e "$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/master/install>)"

2. Update HomeBrew.

brew update

3.Access the MS SQL ODBC Drivers by tapping into its GitHub files.

brew tap microsoft/mssql-release https://github.com/Microsoft/homebrew-mssql-release

4. Use Brew to download and install Microsoft's ODBC drivers for SQL.

brew install microsoft/msodbcsql mssql-tools

5. Restart your Mac!

After that, you should be able to connect and run SQL commands!

### Using an ODBC driver

“Microsoft have written and distributed multiple ODBC drivers for SQL Server:

* {SQL Server} - released with SQL Server 2000
* {SQL Native Client} - released with SQL Server 2005 (also known as version 9.0)
* {SQL Server Native Client 10.0} - released with SQL Server 2008
* {SQL Server Native Client 11.0} - released with SQL Server 2012
* {ODBC Driver 11 for SQL Server} - supports SQL Server 2005 through 2014
* {ODBC Driver 13 for SQL Server} - supports SQL Server 2005 through 2016
* {ODBC Driver 13.1 for SQL Server} - supports SQL Server 2008 through 2016
* {ODBC Driver 17 for SQL Server} - supports SQL Server 2008 through 2017”

(https://github.com/mkleehammer/pyodbc/wiki/Connecting-to-SQL-Server-from-Windows)

### Connecting Python to SQL Server

Then **create a Python script** **that connects to the database**, start by testing the connection.

**import** pypyodbc

db\_driver=**'{ODBC Driver 13 for SQL Server}'**db\_host = **'is-root01.ischool.uw.edu'**db\_name = **'Northwind'**db\_user = **'Info330'**db\_password = **'sql'**connection\_string = **'Driver='** + db\_driver  
connection\_string += **';Server='** + db\_host  
connection\_string += **';Database='** + db\_name  
connection\_string += **';UID='** + db\_user  
connection\_string += **';PWD='** + db\_password + **';'**

objCon = pypyodbc.connect(connection\_string)  
print(**'It worked!'**)  
objCon.close

### Creating Report Objects

Once this is done, you **create reporting objects** in then database, like this reporting stored procedure:

If (Object\_ID('pSelSalesByCategory') != '')

Drop Proc pSelSalesByCategory;

Go

Create Proc pSelSalesByCategory

(@CategoryName nvarchar(100))

AS

Select Year(o.OrderDate) as OrderYear

,Sum(od.Quantity) as TotalQuantity

From Northwind.dbo.Categories as c

Join Northwind.dbo.Products as p

On c.CategoryID = p.CategoryID

Join Northwind.dbo.[Order Details] as od

On p.ProductID = od.ProductID

Join Northwind.dbo.Orders as o

On od.OrderID = o.OrderID

Where c.CategoryName = @CategoryName

Group By c.CategoryName, Year(o.OrderDate)

Order By CategoryName, OrderYear

Go

Grant Exec on pSelSalesByCategory to Public;

### Running SQL Code from Python

Next, let's **create a Python script** that connects to the database and executes the SQL code.

**import** pypyodbc

db\_driver=**'{ODBC Driver 13 for SQL Server}'**db\_host = **'is-root01.ischool.uw.edu'**db\_name = **'Northwind'**db\_user = **'Info330'**db\_password = **'sql'**connection\_string = **'Driver='** + db\_driver  
connection\_string += **';Server='** + db\_host  
connection\_string += **';Database='** + db\_name  
connection\_string += **';UID='** + db\_user  
connection\_string += **';PWD='** + db\_password + **';'**

objCon = pypyodbc.connect(connection\_string)  
  
objCursor = objCon.cursor()  
strCategory = **"Seafood"**objCursor.execute(**"Exec pSelSalesByCategory @CategoryName = "** + strCategory )  
print(**"Data for category: "** + strCategory)  
**for** row **in** objCursor:  
 print(str(row[0]) + **','** + str(row[1]))  
  
objCursor.close  
objCon.close

## Other Programming Languages

There are **many** different **languages** and technologies used to create data-driven applications. You can **expect** them to follow the **same pattern** as seen in both the Python and C# examples:

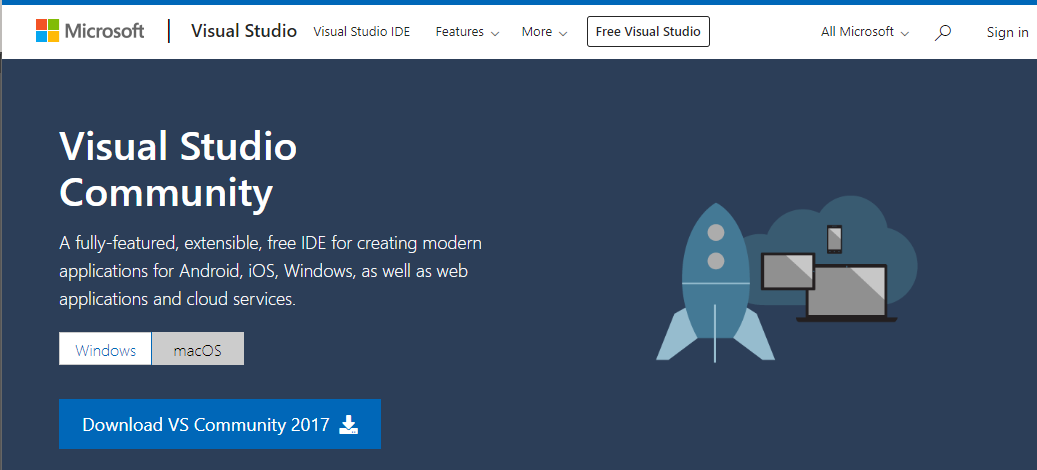
* **Open** a **connection**
* **Issue** a **command**
* **Process** any **results**

While the code may be a bit different in every language, if you **look for this pattern,** you should be able to figure out how the code works!

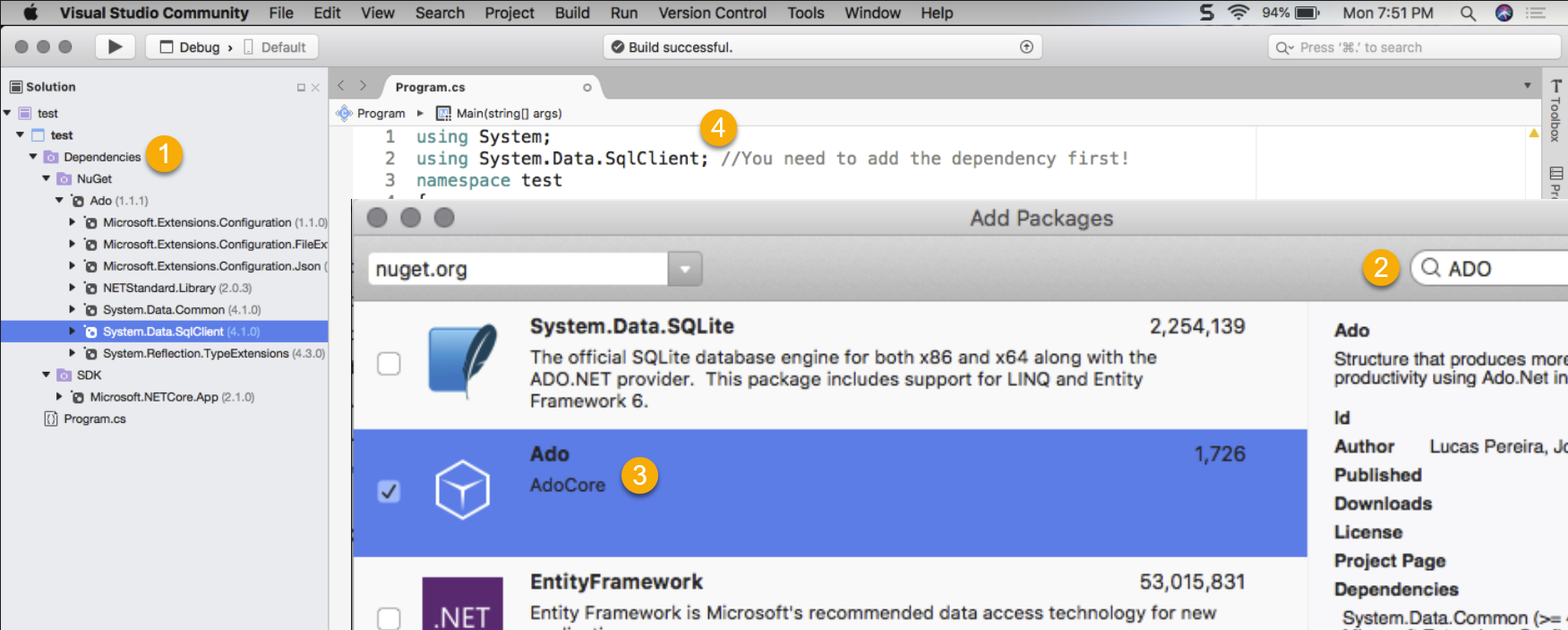
### Microsoft's C#

"You can use C# to create **Windows client applications, XML Web services, distributed components, client-server applications, database applications**, and much, much more." (<https://msdn.microsoft.com/en-us/library/z1zx9t92.aspx>)

You can now run C# on both Windows and Mac OS using Visual Studio (VS). The free version of VS is called the Community Edition, and it is, "A fully-featured, extensible, free IDE for creating modern applications for Android, iOS, Windows, as well as web applications and cloud services." (<https://visualstudio.microsoft.com/vs/community/>, 2018)

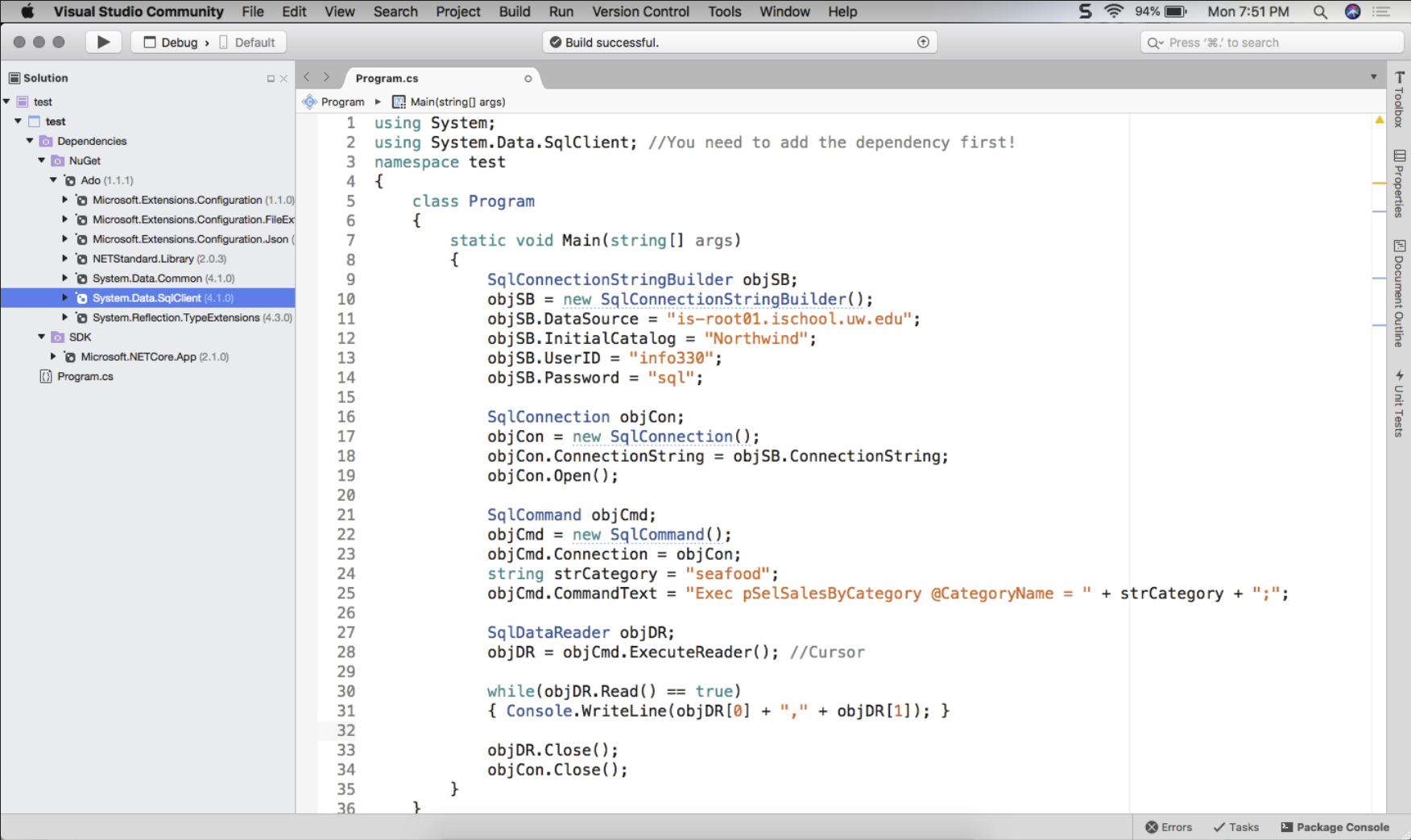


Once installed you will also need add an additional component called Active Data Objects to access SQL Server databases.



#### ADO

"ADO.NET is a **set of classes that expose data access** services to the .NET programmer." (<https://msdn.microsoft.com/en-us/library/aa286484.aspx>)



Here is some example code to try out:

using System;

using System.Data.SqlClient;

namespace test

{

class Program

{

static void Main(string[] args)

{

SqlConnectionStringBuilder objSB;

objSB = new SqlConnectionStringBuilder();

objSB.DataSource = "is-root01.ischool.uw.edu";

objSB.InitialCatalog = "Northwind";

objSB.UserID = "info330";

objSB.Password = "sql";

System.Data.SqlClient.SqlConnection objCon;

objCon = new System.Data.SqlClient.SqlConnection();

objCon.ConnectionString = objSB.ConnectionString;

objCon.Open();

System.Data.SqlClient.SqlCommand objCmd;

objCmd = new System.Data.SqlClient.SqlCommand();

objCmd.Connection = objCon;

string strCategory = "seafood";

objCmd.CommandText = "Exec pSelSalesByCategory @CategoryName = " + strCategory + ";";

System.Data.SqlClient.SqlDataReader objDR;

objDR = objCmd.ExecuteReader(); //Cursor

while (objDR.Read() == true)

{

Console.WriteLine(objDR[0] + "," + objDR[1]);

}

objDR.Close();

objCon.Close();

}

}

}

# Ethics in Reporting

**Just because you can do something does not mean you should**! But **how do you determine what you should and should not do?**

"Data science provides **huge opportunities to improve public and private life**, as well as our environment. However, such opportunities are also coupled to **significant ethical challenges**, posed by the ever-increasing volume of **data** – often personal, if not sensitive – and the growing reliance on **algorithms** to analyse them, combined with the gradual reduction of human involvement or even **oversight** over many automatic processes...

... [A paper published today](http://rsta.royalsocietypublishing.org/content/374/2083/20160360)  ...

In the paper, Floridi and Taddeo split the ethical challenges posed by data science into **three key areas of research**:

* the ethics of data (**how Data is generated, recorded and shared**)
* the ethics of **algorithms** (how artificial intelligence, machine learning and robots **interpret data**)
* the ethics of **practices** (devising **responsible innovation** and **professional codes** to guide this emerging science)

..." (<https://www.oii.ox.ac.uk/news/releases/what-is-data-ethics/> , 2017)

### The ethics of data

"The amount of data we produce every day is truly mind-boggling. There are 2.5 quintillion bytes of data created each day at our current pace, but that pace is only accelerating with the growth of the Internet of Things (IoT). **Over the last two years alone 90 percent of the data in the world was generated**.

…

Our current love affair with social media certainly fuels data creation. According to Domo's Data Never Sleeps 5.0 report, these are numbers **generated every minute of the day**:

Snapchat users share **527,760 photos**

…

**1.5 billion people** are active on Facebook daily

…

There are **600 million Instagrammers**; 400 million who are active every day

…

We send **16 million text messages**

…

**Once you are aware of all the Data you create as a single individual**, you can start to **imagine just how much Data we collectively generate every single day**."

(<https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/#2dac27d960ba>, 2020)

### The ethics of algorithms

"In today's AI community, the topic of bias is finally getting serious attention—in part because of high-profile incidents of algorithms going haywire. **The news is full of these examples, ranging from computer-judged beauty contests that took points off for dark skin to AI-driven recruiting algorithms that were biased against women.** Unless thoroughly examined, the very problems AIs are designed to solve (by removing humans from the equation) may perpetuate them."

(<https://www.forbes.com/sites/insights-intelai/2019/03/27/managing-the-ethics-of-algorithms/#7ff32e833481>, 2020)

### The ethics of practices

"What is clear is that **innovation is not just technological or economical, but also ethical and social.** Innovation **can contribute to the creation or destruction of jobs**, the **fight against crime**, the **invasion of privacy**, finding the cure for hereditary diseases or the manufacture of synthetic genomes. Innovation is related to **education, interpersonal dialogue, participative democracy** and new and innovative economic models.

**Responsible innovation is crucial. But it is not always easy.** We are in a complex era full of disruption that requires careful consideration, especially in relation to the positive and negative consequences associated." (<https://blog.iese.edu/ethics/2016/08/18/whats-ethics-role-in-responsible-innovation/>, 2020)