1. **New;**

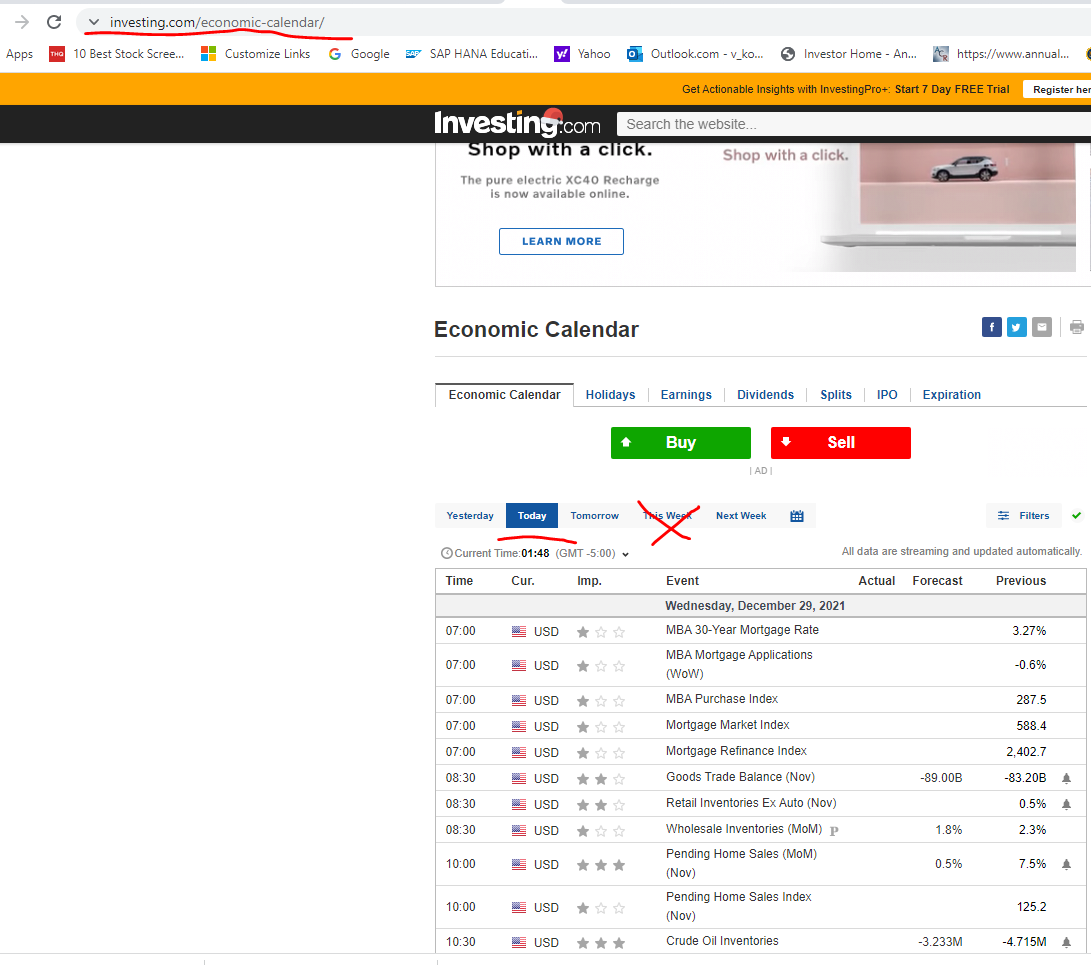
Create a new script, **investing\_download\_today\_login.py** based oninvesting\_download\_thisweek\_login.py. This script will not create any investing\_output.csv file. I will just update the Tables;

* **eco\_events**
* **news\_headlines**

with actual values, if required (corresponding event\_name in the Table eco\_events\_impact has no\_actual column value = False).

**url** same as current script investing\_download\_thisweek\_login.py); https://www.investing.com/economic-calendar/

The script will be using **Today Tab instead of This Week Tab** for downloading Data.



This script will be iterative in nature and will have sleep time in minutes specified in Control.csv.

**Table definitions** are attached in;

* Table\_Definition\_eco\_events.sql
* Table\_Definition\_eco\_events\_impact.sql
* Table\_Definition\_news\_headlines.sql

**Logic for conversion of data to Table column values** is provided in;

* Correspondence\_EcoEvents\_to\_Output\_CSV.xlsx in the Correspondence sheet.

**Table news\_headlines Insert;** Script should also insert a row in the news\_headlines Table based on logic provided in;

* Correspondence\_EcoEvents\_to\_Output\_CSV.xlsx in the news\_headlines\_correspondence sheet.

news\_headlines Table insert will be triggered by availability of Actual values for the first time (Not in eco\_events Table) for the event\_name. In case the matching event\_name in eco\_events\_impact Table has no\_actual column value = True then script will not wait for the Actual values. Script will insert the corresponding row in the Table news\_headlines on the event\_date and event\_time.

news\_headlines Table insert will be driven primarily by the Table values found in the Table eco\_events\_impact for the matching event\_name (except for the headline). Symbols & url will be pulled from eco\_events\_impact based on matching the eco\_events.event\_name with eco\_events\_impact.event\_name as follows;

eco\_events.event\_name LIKE eco\_events\_impact.event\_name

eco\_events\_impact.event\_name column values contain wildcard and end with percent sing ("%")

1. **Upgrade;**

**Script;** investing\_download\_thisweek\_login.py

1. Before creating investing\_output.csv in the Data sub-folder. Script should **UpSert (Insert if no row found, else Update) the Table;**

* **eco\_events**

**in PostgreSQL**.

This script will be executed once a day.

**Logic for conversion of data to Table column values** is provided in;

* Correspondence\_EcoEvents\_to\_Output\_CSV.xlsx in the Correspondence sheet.

1. Script should use Control.csv in Control sub-folder to pick up the **database login parameters** from the Control.csv.