Code Clarification:

!pip install censusdata

import censusdata

import pandas as pd

# Define the variables to be pulled

variables = [

'B25077\_001E', 'C24050\_001E', 'C24050\_011E', 'C24050\_014E', 'B25013\_009E',

'B25013\_008E', 'B25013\_007E', 'B25002\_001E', 'B25002\_002E', 'B01003\_001E',

'B25031\_001E', 'B25103\_001E', 'B25118\_014E', 'B25118\_015E', 'B25118\_016E',

'B25118\_017E', 'B25118\_018E', 'B25118\_019E', 'B25118\_020E', 'B25118\_021E',

'B25118\_022E', 'B25118\_023E', 'B25118\_024E', 'B25118\_025E', 'B25118\_001E',

'B25026\_010E', 'B25026\_011E', 'B25026\_012E', 'B25026\_013E', 'B25026\_014E',

'B25026\_015E', 'B25046\_003E', 'B08013\_001E', 'B25003I\_003E', 'B25003A\_003E',

'B25003B\_003E', 'B25003C\_003E', 'B25003D\_003E', 'B25003E\_003E', 'B25003F\_003E',

'B25003G\_003E', 'B25003H\_003E', 'B19013\_001E', 'B25119\_003E', 'B25010\_002E',

'B25010\_003E', 'B25009\_011E', 'B25009\_012E', 'B25009\_013E', 'B25009\_014E',

'B25009\_015E', 'B25009\_016E', 'B25009\_017E', 'B25032\_013E', 'B25032\_014E',

'B25032\_015E', 'B25032\_016E', 'B25032\_017E', 'B25032\_018E', 'B19057\_001E',

'B19057\_002E', 'B19057\_003E', 'B25042\_001E', 'B25042\_003E', 'B25042\_004E',

'B25042\_005E', 'B25042\_006E', 'B25042\_007E', 'B25042\_008E', 'B25042\_010E',

'B25042\_011E', 'B25042\_012E', 'B25042\_013E', 'B25042\_014E', 'B25042\_015E'

]

# Define the state and county codes

state\_code = '30' # Replace with the desired state code

county\_code = '\*' # '\*' indicates all counties within the state

# Download the data at the block group level

block\_group\_data = censusdata.download(

'acs5', 2019,

censusdata.censusgeo([('state', state\_code), ('county', county\_code), ('block group', '\*')]),

variables,

key='faad7551acd1bf70615d1197d3fac92869820e1a'

)

# Reset the index and rename the column

block\_group\_data = block\_group\_data.reset\_index()

block\_group\_data = block\_group\_data.rename(columns={'index': 'Block Group'})

# Convert 'Block Group' to string and sort

block\_group\_data['Block Group'] = block\_group\_data['Block Group'].astype(str)

block\_group\_data = block\_group\_data.sort\_values('Block Group')

# Display the data in a more formatted and structured way

formatted\_data = block\_group\_data[['Block Group'] + variables]

formatted\_data = formatted\_data.set\_index('Block Group')

# Set the display option to show all columns

pd.set\_option('display.max\_columns', None)

# Display the formatted data

formatted\_data  
  
  
The data requested is at the block group level - In your code, you're using the censusdata.download function to download data at the block group level (('block group', '\*')). This matches your request to get data at the block group level instead of the county level.

* You've mentioned that the block group data should include the state, county, census tract, and block group numbers - In your code, the 'Block Group' column which is created by resetting the index of block\_group\_data should include this information in the format you described.
* You want an extra column that identifies each block group with a long string of numbers - This is provided by the 'Block Group' column in your block\_group\_data DataFrame. This column is later set as the index in formatted\_data.
* You don't need data for every census tract or block, just the block groups - Your code requests data for all block groups within a given state and county (('block group', '\*')), and not for individual census tracts or blocks.
* Your request to retrieve all block groups for a specific state is also handled - You're using state\_code to specify the state, and 'county', '\*' to request data for all counties within the state, which implicitly fetches data for all block groups in these counties.