Sample Project 2: Do WIC households purchase similar foods compared to households that do not participate in the WIC program but are eligible?

Notebook Setup

In [ ]:

```
# pandas-related imports
import pandas as pd

# database interaction imports
from pyathenajdbc import connect

# visualization
import matplotlib as plt
%matplotlib inline

# to use weights
from statsmodels.stats.weightstats import DescrStatsW
```

In [ ]:

```
# connection to the database
conn = connect(s3_staging_dir = 's3://usda-iri-2019-queryresults/',
               region_name = 'us-gov-west-1',
               LogLevel = '0',
               workgroup = 'workgroup-iri_usda')
```

Define Cohort

Since the sample project's question focuses exclusively on the difference in 100% whole wheat bread purchases between WIC-participant and WIC-eligible households in 2017, we will restrict our study cohort to just households who were either WIC participants or WIC-eligible in 2017 and had sufficient purchasing data. Luckily, the demographic table limited to this cohort has already been created in the `project_q2_cohort` table within the `iri_usda_2019_db` database, so we will begin by analyzing this table.

Data Exploration:

Review of 2nd Data Exploration Notebook

The second data exploration (02_01_Data_Exploration_Food_Expense.ipynb) notebook contains code that we will slightly alter here for our data exploration. Our main goal is to get a better sense of our cohort and their characteristics in ways that may impact our future analysis in the project.

In [ ]:

```
#check out project_q2_cohort
qry = '''
select *
from iri_usda_2019_db.project_q2_cohort
limit 10
'''

pd.read_sql(qry, conn)
```

In [ ]:

```
# count total amount of rows
qry = '''
select count(*) as total_wic_and_eligible_count
from iri_usda_2019_db.project_q2_cohort
'''

pd.read_sql(qry, conn)
```

In [ ]:

```
# count total amount of WIC participants
qry = '''
select count(*) as wic_count
from iri_usda_2019_db.project_q2_cohort
where wic_june = 1
'''

pd.read_sql(qry, conn)
```

In [ ]:

```
# count total amount of WIC-eligible households
qry = '''
select count(*) as wic_eligible_count
from iri_usda_2019_db.project_q2_cohort
where wic_june != 1
'''

pd.read_sql(qry, conn)
```

Visualization Important Distributions

In [ ]:

```
qry = '''
select *
from iri_usda_2019_db.project_q2_cohort
'''

cohort = pd.read_sql(qry, conn)
```

In [ ]:

```
wic = cohort[cohort['wic_june'] == 1]
```

In [ ]:

```
wic_eligible = cohort[cohort['wic_june'] != 1]
```

Household size distribution in WIC and WIC-eligible households

Non-weighted distributions

In [ ]:

```
wic_hhsize = wic.groupby('hhsize').size().reset_index().rename(columns={0:'count'})
wic_eligible_hhsize = wic_eligible.groupby('hhsize').size().reset_index().rename(columns={0:'count'})
```

In [ ]:

```
wic_hhsize.plot(x='hhsize', y='count', kind='bar', title='Household size distribution in WIC-households')
wic_eligible_hhsize.plot(x='hhsize', y='count', kind='bar', title='Household size distribution in WIC-eligible households')
```

Weighted distributions

In [ ]:

```
weighted_wic_hhsize = DescrStatsW(wic.hhsize, weights=wic.projection61k)
weighted_wic_hhsize.quantile([.1,.25,.5,.75,.9])
```

In [ ]:

```
weighted_wic_eligible_hhsize = DescrStatsW(wic_eligible.hhsize, weights=wic_eligible.projection61k)
weighted_wic_eligible_hhsize.quantile([.1,.25,.5,.75,.9])
```

Income distribution in WIC and WIC-eligible households

Non-weighted distributions

In [ ]:

```
wic_hhinc = wic.groupby('hhinc').size().reset_index().rename(columns={0:'count'})
wic_eligible_hhinc = wic_eligible.groupby('hhinc').size().reset_index().rename(columns={0:'count'})
```

In [ ]:

```
wic_hhinc.plot(x='hhinc', y='count', kind='bar', title='Income distribution in WIC-households')
wic_eligible_hhinc.plot(x='hhinc', y='count', kind='bar', title='Income distribution in WIC-eligible households')
```

Weighted distributions

In [ ]:

```
weighted_wic_hhinc = DescrStatsW(wic.hhinc, weights=wic.projection61k)
weighted_wic_hhinc.quantile([.1,.25,.5,.75,.9])
```

In [ ]:

```
weighted_wic_eligible_hhinc = DescrStatsW(wic_eligible.hhinc, weights=wi
c_eligible.projection61k)
weighted_wic_eligible_hhinc.quantile([.1,.25,.5,.75,.9])
```

Join Tables

Household to Trip

Now, let's find all of the purchasing data in 2017 for our cohort. After retreiving the data, we will be able to filter for just bread products. Since we are going to use this table later in our analysis, we will create a table `project_q2_purchases` in the `iri_usda_2019_db` database.

In [ ]:

```
# see existing table list
table_list = pd.read_sql('show tables IN iri_usda_2019_db;', conn)
print(table_list)

# get a series of tab_name values
s = pd.Series(list(table_list['tab_name']))
```

In [ ]:

```
# create money spent per household table for WIC households with suffici
ent purchasing data in 2017
if('project_q2_purchases' not in s.unique()):
    query = """
    CREATE table iri_usda_2019_db.project_q2_purchases
    WITH (
    format = 'Parquet',
    parquet_compression = 'SNAPPY'
    )
    AS
    select trip.panid, trip.upc, trip.quantity, demo.wic_june
    from iri_usda_2019_db.project_q2_cohort demo
    left join iri_usda.trip_all trip
    on trip.panid = demo.panid
    where trip.year = '2017'
    """
    with conn.cursor() as cursor:
        cursor.execute(query)
```

In [ ]:

```
#check to see if everyone was matched to purchase data
```

In [ ]:

```
# check project_q2_purchases
qry = '''
select *
from iri_usda_2019_db.project_q2_purchases
limit 10
'''

pd.read_sql(qry, conn)
```

Trip data for our cohort to Product data

To filter for just bread products, we will need to add the description and category corresponding to the upc code to our current table. In doing so, we will join the trip data for our cohort to `pd_master_all` .

In [ ]:

```
qry = '''
select trip.panid, trip.upc, trip.quantity, trip.wic_june, product.upcdesc, product.category
from iri_usda_2019_db.project_q2_purchases trip
left join iri_usda.pd_master_all product
on product.upc = trip.upc
where category like '%BREAD%'
'''

all_bread = pd.read_sql(qry, conn)
```

Proportion of 100% whole wheat bread purchases in WIC and WIC-eligible households

Let's first subset our `all_bread` table by WIC and WIC-eligible households

In [ ]:

```
# All bread puchases in WIC-households
all_bread_wic = all_bread[all_bread['wic_june'] == 1]

# All bread purchases in WIC-eligible households
all_bread_wic_eligible = all_bread[all_bread['wic_june'] != 1]
```

Now let's find a proportion of 100% whole wheat bread purchases in WIC households

In [ ]:

```
# Number of purchases of bread products that contain '100% WHOLE WHEAT' string
len(all_bread_wic[all_bread_wic['upcdesc'].str.contains('100% WHOLE WHEAT')])
```

In [ ]:

```
# Number of total bread purchases by WIC-households
len(all_bread_wic)
```

```
In [ ]: # Find a proportion of 100% WHOLE WHEAT to all bread purchases by WIC-households
        len(all_bread_wic[all_bread_wic['upcdesc'].str.contains('100% WHOLE WHEAT')]) / len(all_bread_wic) * 100
```

Now let's find a proportion of 100% whole wheat bread purchases by WIC-eligible households

```
In [ ]: # Number of 100% WHOLE WHEAT purchases by WIC-eligible households
        len(all_bread_wic_eligible[all_bread_wic_eligible['upcdesc'].str.contains('100% WHOLE WHEAT')])
```

```
In [ ]: # Number of total bread purchases by WIC-eligible households
        len(all_bread_wic_eligible)
```

```
In [ ]: # Proportion of 100% WHOLE WHEAT to all bread purchases by WIC-eligible households
        len(all_bread_wic_eligible[all_bread_wic_eligible['upcdesc'].str.contains('100% WHOLE WHEAT')]) / len(all_bread_wic_eligible) * 100
```