Extract, Transfer, Load

Pandas & Sqlalchemy

Background

We chose these data sets because the ability to join them will be straightforward given we have a primary key of zip code for the population density data table that can match the foreign key of zip code for the IRS tax return data. In addition, we contemplated what sort of analysis could be done based on the tables in this database and thought that interesting insights could be distilled relating to income, tax bracket, zip code, and the population density within a zip code.

Data Sources

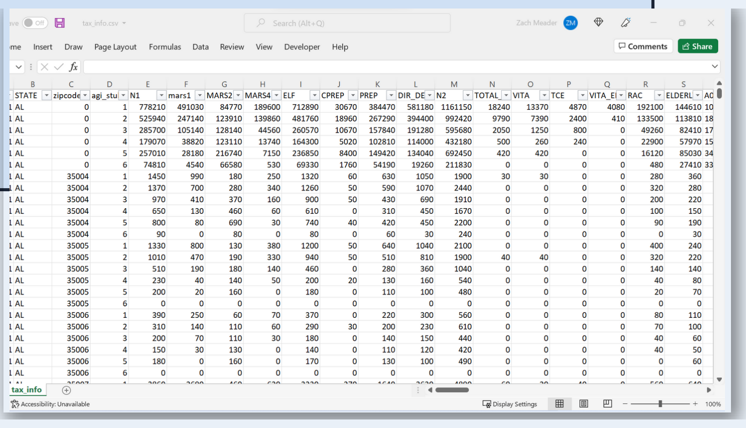
IRS Tax Returns:
Source: *https://www.irs.gov/e-file-providers/definition-of-adjusted-gross-income*

Population Density
Source: *https://simplemaps.com/data/us-zips*

01

Extract

- **Imported:** Dependencies for Pandas and Sqlalchemy and CSV files into Pandas environment
- **Transformed:** read CSV files and converted them into data frame



tax_info.csv

	STATE	zipcode	agi_stu	N1	mars1	MARS2	MARS4	ELF	CPREP	PREP	DIR_DE	N2	TOTAL_	VITA	TCE	VITA_EI	RAC	ELDERL	A0
1	AL	0	1	778210	491030	84770	189600	712890	30670	384470	581180	1161150	18240	13370	4870	4080	192100	144610	10
1	AL	0	2	525940	247140	123910	139860	481760	18960	267290	394400	992420	9790	7390	2400	410	133500	113810	18
1	AL	0	3	285700	105140	128140	44560	260570	10670	157840	191280	595680	2050	1250	800	0	49260	82410	17
1	AL	0	4	179070	38820	123110	13740	164300	5020	102810	114000	432180	500	260	240	0	22900	57970	15
1	AL	0	5	257010	28180	216740	7150	236850	8400	149420	134040	692450	420	420	0	0	16120	85030	34
1	AL	0	6	74810	4540	66580	530	69330	1760	54190	19260	211830	0	0	0	0	480	27410	33
1	AL	35004	1	1450	990	180	250	1320	60	630	1050	1900	30	30	0	0	280	360	
1	AL	35004	2	1370	700	280	340	1260	50	590	1070	2440	0	0	0	0	320	280	
1	AL	35004	3	970	410	370	160	900	50	430	690	1910	0	0	0	0	200	220	
1	AL	35004	4	650	130	460	60	610	0	310	450	1670	0	0	0	0	100	150	
1	AL	35004	5	800	80	690	30	740	40	420	450	2200	0	0	0	0	90	190	
1	AL	35004	6	90	0	80	0	80	0	60	30	240	0	0	0	0	0	30	
1	AL	35005	1	1330	800	130	380	1200	50	640	1040	2100	0	0	0	0	400	240	
1	AL	35005	2	1010	470	190	330	940	50	510	810	1900	40	40	0	0	320	220	
1	AL	35005	3	510	190	180	140	460	0	280	360	1040	0	0	0	0	140	140	
1	AL	35005	4	230	40	140	50	200	20	130	160	540	0	0	0	0	40	80	
1	AL	35005	5	200	20	160	0	180	0	110	100	480	0	0	0	0	20	70	
1	AL	35005	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1	AL	35006	1	390	250	60	70	370	0	220	300	560	0	0	0	0	80	110	
1	AL	35006	2	310	140	110	60	290	30	200	230	610	0	0	0	0	70	100	
1	AL	35006	3	200	70	110	30	180	0	140	150	440	0	0	0	0	40	60	
1	AL	35006	4	150	30	130	0	140	0	110	120	420	0	0	0	0	40	50	
1	AL	35006	5	180	0	160	0	170	0	130	100	490	0	0	0	0	0	60	
1	AL	35006	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

tax_info

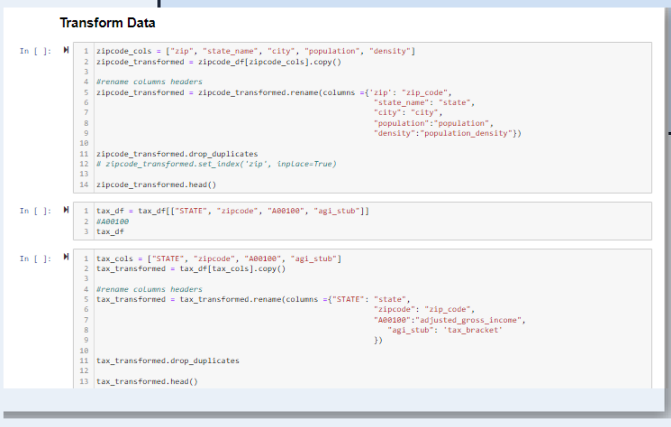
02

Transfer

Created: defined lists to hold values for data sources & return based on selected columns

Identified: found zip codes within the Tax Returns dataset not present within the Population Density dataset and add zip codes to the Population Density dataset

Cleaned: renamed columns and dropped duplicates and dropped index



Transform Data

In [ ]:
```
zipcode_cols = ["zip", "state_name", "city", "population", "density"]
zipcode_transformed = zipcode_df[zipcode_cols].copy()

#rename columns headers
zipcode_transformed = zipcode_transformed.rename(columns ={'zip': "zip_code",
                                                          "state_name": "state",
                                                          "city": "city",
                                                          "population":"population",
                                                          "density":"population_density"})

zipcode_transformed.drop_duplicates
# zipcode_transformed.set_index('zip', inplace=True)

zipcode_transformed.head()
```

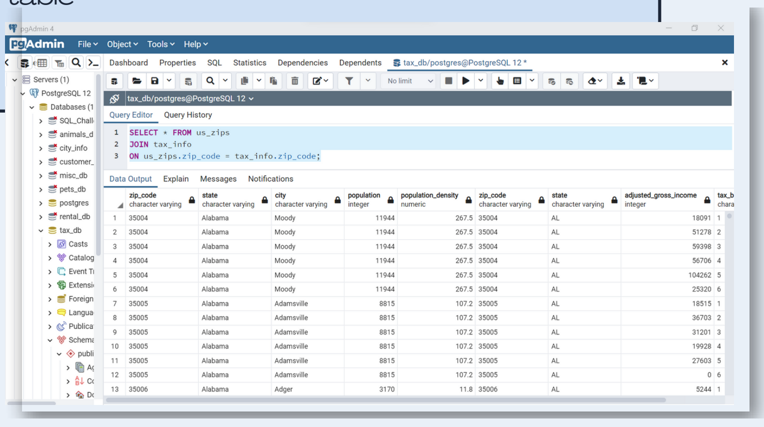
In [ ]:
```
tax_df = tax_df[["STATE", "zipcode", "A00100", "agi_stub"]]
#A00100
tax_df
```

In [ ]:
```
tax_cols = ["STATE", "zipcode", "A00100", "agi_stub"]
tax_transformed = tax_df[tax_cols].copy()

#rename columns headers
tax_transformed = tax_transformed.rename(columns ={"STATE": "state",
                                                  "zipcode": "zip_code",
                                                  "A00100":"adjusted_gross_income",
                                                     "agi_stub": 'tax_bracket'
                                                  })

tax_transformed.drop_duplicates

tax_transformed.head()
```

03

Load

- **Defined:** added a new database in SQL & used zip code in Population Density Primary Key linked it to zip code foreign key with IRS Tax Return
- **Created:** tables for Population Density and for IRS Tax Returns
- **Connected:** Connected SQL Database to python & insert data into Population Density and IRS Tax Returns tables using to_sql
- **Joined:** Join tables in SQL on zip codes & found Zip Codes unmapped and add them to the population density table



```
SELECT * FROM us_zips
JOIN tax_info
ON us_zips.zip_code = tax_info.zip_code;
```

	zip_code character varying	state character varying	city character varying	population integer	population_density numeric	zip_code character varying	state character varying	adjusted_gross_income integer	tax_b chara
1	35004	Alabama	Moody	11944	267.5	35004	AL	18091	1
2	35004	Alabama	Moody	11944	267.5	35004	AL	51278	2
3	35004	Alabama	Moody	11944	267.5	35004	AL	59398	3
4	35004	Alabama	Moody	11944	267.5	35004	AL	56706	4
5	35004	Alabama	Moody	11944	267.5	35004	AL	104262	5
6	35004	Alabama	Moody	11944	267.5	35004	AL	25320	6
7	35005	Alabama	Adamsville	8815	107.2	35005	AL	18515	1
8	35005	Alabama	Adamsville	8815	107.2	35005	AL	36703	2
9	35005	Alabama	Adamsville	8815	107.2	35005	AL	31201	3
10	35005	Alabama	Adamsville	8815	107.2	35005	AL	19928	4
11	35005	Alabama	Adamsville	8815	107.2	35005	AL	27603	5
12	35005	Alabama	Adamsville	8815	107.2	35005	AL	0	6
13	35006	Alabama	Adger	3170	11.8	35006	AL	5244	1