U.S. Bureau of Labor Statistics
(Data Engineering Track)

-Yen- -Nebiat- -Bryan- -Adebola- -Stephanie-

Occupational Employment and Wage Statistics (OEWS) program

- Occupational Employment and Wage Statistics (OEWS) program produces employment and wage estimates in nonfarm establishments for about 830 occupations
- The estimates are available for the nationwide, for individual States, and for metropolitan areas and nonmetropolitan areas
- The estimates are constructed from a sample of about 1.1 million establishments
- OEWS survey is a semiannual survey, and the data is published annually. Each publishment includes three-year pooled samples
- Our project focuses on the data for individual States from the most recent data released in May 2023 which includes data from 6 surveys collected from November 2020 - May 2023

ETL

Extract: OEWS Data from BLS database (EXCEL) --> Jupyter Notebook

Transform: Clean & Aggregate Data into readable CSVs using Pandas

Load: Pull CSVs into psycopg2 - Build SQL database

Extract

The May 2023 OEWS is stored in the 'Data' folder, and we extract the raw data using a Pandas DataFrame

```
file_path = 'Data/all_data_M_2023.xlsx'
all_owes_df = pd.read_excel(file_path, sheet_name='All May 2023 data')
all_owes_df.head()
```

✓ 2m 9.5s

	AREA	AREA_TITLE	AREA_TYPE	PRIM_STATE	NAICS	NAICS_TITLE	I_GROUP	OWN_CODE	OCC_CODE	OCC_TITLE	...	H_MEDIAN	H_PCT75	H_PCT90	A_PCT10	A_PCT25	A_MEDIAN	A_PCT75
0	99	U.S.	1	US	000000	Cross-industry	cross-industry	1235	00-0000	All Occupations	...	23.11	37.01	58.4	29050	35660	48060	76980
1	99	U.S.	1	US	000000	Cross-industry	cross-industry	1235	11-0000	Management Occupations	...	56.19	81.29	111.36	54550	78330	116880	169090
2	99	U.S.	1	US	000000	Cross-industry	cross-industry	1235	11-1000	Top Executives	...	49.74	79.57	#	46400	66170	103460	165500
3	99	U.S.	1	US	000000	Cross-industry	cross-industry	1235	11-1010	Chief Executives	...	99.37	#	#	80000	130840	206680	#
4	99	U.S.	1	US	000000	Cross-industry	cross-industry	1235	11-1011	Chief Executives	...	99.37	#	#	80000	130840	206680	#

5 rows × 32 columns

```
state_oews_df = all_owes_df.loc[all_owes_df['AREA_TYPE'] == 2, :]
state_oews_df.head()
```

	AREA	AREA_TITLE	AREA_TYPE	PRIM_STATE	NAICS	NAICS_TITLE	I_GROUP	OWN_CODE	OCC_CODE	OCC_TITLE	...	H_MEDIAN	H_PCT75	H_PCT90	A_PCT10	A_PCT25	A_MEDIAN	A_PCT75
177501	1	Alabama	2	AL	000000	Cross-industry	cross-industry	1235	00-0000	All Occupations	...	19.88	30.09	46.18	22620	29580	41350	62580
177502	2	Alaska	2	AK	000000	Cross-industry	cross-industry	1235	00-0000	All Occupations	...	26.99	40.52	58.35	31200	38720	56140	84280
177503	4	Arizona	2	AZ	000000	Cross-industry	cross-industry	1235	00-0000	All Occupations	...	22.92	35.05	51.67	30870	36150	47680	72900
177504	5	Arkansas	2	AR	000000	Cross-industry	cross-industry	1235	00-0000	All Occupations	...	18.78	28.32	40.5	26360	30000	39060	58900
177505	6	California	2	CA	000000	Cross-industry	cross-industry	1235	00-0000	All Occupations	...	25.98	44.83	73.07	34170	37890	54030	93250

5 rows × 32 columns

We filter the 'AREA_TYPE' column to extract and narrow down the data to State level.

Transform – remove unnecessary columns

```
columns_to_keep = [
    'AREA',
    'AREA_TITLE',
    'PRIM_STATE',
    'OCC_CODE',
    'OCC_TITLE',
    'O_GROUP',
    'TOT_EMP',
    'EMP_PRSE',
    'JOBS_1000',
    'LOC_QUOTIENT',
    'H_MEAN',
    'A_MEAN',
    'MEAN_PRSE',
    'H_PCT25',
    'H_MEDIAN',
    'H_PCT75',
    'A_PCT25',
    'A_MEDIAN',
    'A_PCT75'
]

cleaned_df = state_oews_df[columns_to_keep]
cleaned_df.head()
```

- Refined the number of columns down from 32 to 19
- AREA: state FIPS code
- AREA_TITLE: state names
- PRIM_STATE: abbreviation of state names
- OCC_CODE: Standard Occupational Classification (SOC) code
- OCC_TITLE: occupation title
- O_GROUP: occupation level, including 'Total', 'Major', and 'Detailed'
- TOT_EMP: estimated total employment
- EMP_PRSE: Percent relative standard error for the employment estimate
- JOBS_1000: The number of jobs in the given occupation per 1,000 jobs in the given area
- LOC_QUOTIENT: ratio of an occupation's share of employment in a given area to that occupation's share of employment in the U.S. as a whole
- H_MEAN & A_MEAN: mean hourly wage and mean annual wage
- H_PCT & A_PCT: hourly percentile wage and annual percentile wage

Transform – Rename Columns

```
renamed_oews_df = cleaned_df.rename(columns={
    'AREA': 'area',
    'AREA_TITLE': 'area_title',
    'PRIM_STATE': 'prim_state',
    'OCC_CODE': 'occ_code',
    'OCC_TITLE': 'occ_title',
    'O_GROUP': 'o_group',
    'TOT_EMP': 'tot_emp',
    'EMP_PRSE': 'emp_prse',
    'JOBS_1000': 'jobs_1000',
    'LOC_QUOTIENT': 'loc_quotient',
    'H_MEAN': 'h_mean',
    'A_MEAN': 'a_mean',
    'MEAN_PRSE': 'mean_prse',
    'H_PCT25': 'h_pct25',
    'H_MEDIAN': 'h_median',
    'H_PCT75': 'h_pct75',
    'A_PCT25': 'a_pct25',
    'A_MEDIAN': 'a_median',
    'A_PCT75': 'a_pct75'
})
renamed_oews_df.head()
```

- We chose PostgreSQL as our database to ensure optimal performance.
- To meet its requirements, we standardized the column names by using lowercase letters and underscores.
- This approach enhances consistency, improves efficiency, and minimizes potential errors.

Transform – convert data types

Before the conversion

```
<class 'pandas.core.frame.DataFrame'>
Index: 36643 entries, 177501 to 214143
Data columns (total 19 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   area          36643 non-null  int64 
 1   area_title    36643 non-null  object
 2   prim_state    36643 non-null  object
 3   occ_code      36643 non-null  object
 4   occ_title     36643 non-null  object
 5   o_group       36643 non-null  object
 6   tot_emp       36643 non-null  object
 7   emp_prse      36643 non-null  object
 8   jobs_1000     36643 non-null  object
 9   loc_quotient  36643 non-null  object
 10  h_mean        36643 non-null  object
 11  a_mean        36643 non-null  object
 12  mean_prse     36643 non-null  object
 13  h_pct25       36643 non-null  object
 14  h_median      36643 non-null  object
 15  h_pct75       36643 non-null  object
 16  a_pct25       36643 non-null  object
 17  a_median      36643 non-null  object
 18  a_pct75       36643 non-null  object
dtypes: int64(1), object(18)
memory usage: 5.6+ MB
```

After the conversion

```
<class 'pandas.core.frame.DataFrame'>
Index: 36643 entries, 177501 to 214143
Data columns (total 19 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   area          36643 non-null  int64  
 1   area_title    36643 non-null  object 
 2   prim_state    36643 non-null  object 
 3   occ_code      36643 non-null  object 
 4   occ_title     36643 non-null  object 
 5   o_group       36643 non-null  object 
 6   tot_emp       35775 non-null  float64
 7   emp_prse      35775 non-null  float64
 8   jobs_1000     35775 non-null  float64
 9   loc_quotient  35775 non-null  float64
 10  h_mean        33535 non-null  float64
 11  a_mean        35951 non-null  float64
 12  mean_prse     36150 non-null  float64
 13  h_pct25       33400 non-null  float64
 14  h_median      33121 non-null  float64
 15  h_pct75       32851 non-null  float64
 16  a_pct25       35816 non-null  float64
 17  a_median      35531 non-null  float64
 18  a_pct75       35244 non-null  float64
dtypes: float64(13), int64(1), object(5)
memory usage: 5.6+ MB
```

Transform – handle missing values

Before the handling

```
<class 'pandas.core.frame.DataFrame'>
Index: 36643 entries, 177501 to 214143
Data columns (total 19 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   area          36643 non-null  int64
 1   area_title    36643 non-null  object
 2   prim_state    36643 non-null  object
 3   occ_code      36643 non-null  object
 4   occ_title     36643 non-null  object
 5   o_group       36643 non-null  object
 6   tot_emp       35775 non-null  float64
 7   emp_prse      35775 non-null  float64
 8   jobs_1000     35775 non-null  float64
 9   loc_quotient  35775 non-null  float64
 10  h_mean        33535 non-null  float64
 11  a_mean        35951 non-null  float64
 12  mean_prse     36150 non-null  float64
 13  h_pct25       33400 non-null  float64
 14  h_median      33121 non-null  float64
 15  h_pct75       32851 non-null  float64
 16  a_pct25       35816 non-null  float64
 17  a_median      35531 non-null  float64
 18  a_pct75       35244 non-null  float64
dtypes: float64(13), int64(1), object(5)
memory usage: 5.6+ MB
```

After the handling

```
final_df = renamed_oews_df.dropna(how='any')
final_df.info()
```

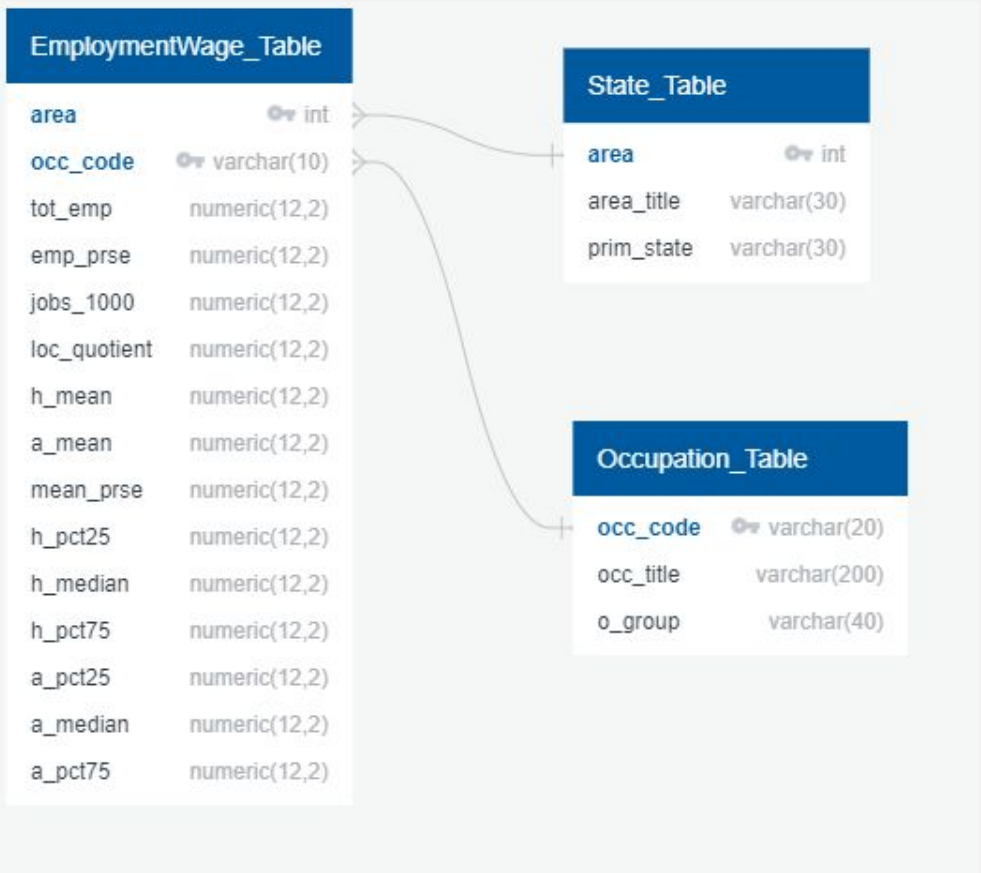
```
<class 'pandas.core.frame.DataFrame'>
Index: 31991 entries, 177501 to 214143
Data columns (total 19 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   area          31991 non-null  int64
 1   area_title    31991 non-null  object
 2   prim_state    31991 non-null  object
 3   occ_code      31991 non-null  object
 4   occ_title     31991 non-null  object
 5   o_group       31991 non-null  object
 6   tot_emp       31991 non-null  float64
 7   emp_prse      31991 non-null  float64
 8   jobs_1000     31991 non-null  float64
 9   loc_quotient  31991 non-null  float64
 10  h_mean        31991 non-null  float64
 11  a_mean        31991 non-null  float64
 12  mean_prse     31991 non-null  float64
 13  h_pct25       31991 non-null  float64
 14  h_median      31991 non-null  float64
 15  h_pct75       31991 non-null  float64
 16  a_pct25       31991 non-null  float64
 17  a_median      31991 non-null  float64
 18  a_pct75       31991 non-null  float64
dtypes: float64(13), int64(1), object(5)
memory usage: 4.9+ MB
```

Load - Entity Relationship Diagram(ERD)

```
EmployentWage_Table
-
area int PK FK >- State_Table.area
occ_code varchar(10) PK FK >- Occupation_Table.occ_code
tot_emp numeric(12,2)
emp_prse numeric(12,2)
jobs_1000 numeric(12,2)
loc_quotient numeric(12,2)
h_mean numeric(12,2)
a_mean numeric(12,2)
mean_prse numeric(12,2)
h_pct25 numeric(12,2)
h_median numeric(12,2)
h_pct75 numeric(12,2)
a_pct25 numeric(12,2)
a_median numeric(12,2)
a_pct75 numeric(12,2)

Occupation_Table
-
occ_code PK varchar(20)
occ_title varchar(200)
o_group varchar(40)

State_Table
-
area PK int
area_title varchar(30)
prim_state varchar(30)
```



- We used area and occ_code as compound keys to uniquely identify each row. Each row contains statistics about the occupations and types of jobs specific to each state.
- We used the area as a foreign key to connect to the State table and the occ_code as a foreign key to connect to the occupation table.

Load – insert data into PostgreSQL

What database did we use and why?

- We wanted a relational database that can operate on multiple operating systems and has components to ensure data integrity.
- We selected PostgreSQL which is a powerful and popular choice for relational databases.
  - Cross-Platform and Scalability: PostgreSQL runs on many operating systems (Linux, Windows, macOS), and is designed to scale
  - Strong Data Integrity and Security: PostgreSQL provides robust data integrity through constraints, including primary keys, foreign keys, and unique constraints.

Query – Examples of how to interact with the database

Query | Query History

```
1 SELECT s.area_title AS state, o.occ_title AS occupation, e.a_mean AS avg_annual_wage
2 FROM employmentwage_table e
3 JOIN state_table s ON e.area = s.area
4 JOIN occupation_table o ON e.occ_code = o.occ_code
5 WHERE s.area_title = 'North Carolina'
6 AND o.occ_title = 'Data Scientists';
```

Data Output | Messages | Notifications

	state character varying (30)	occupation character varying (200)	avg_annual_wage numeric (12,2)
1	North Carolina	Data Scientists	125950.00

Find the average wage of a "Data Scientist" in "North Carolina"

Find the Top 5 Highest Paying occupations in a specific state in "North Carolina"

Query | Query History

```
1 SELECT s.area_title AS state, o.occ_title AS occupation, e.a_mean AS avg_annual_wage
2 FROM employmentwage_table e
3 JOIN state_table s ON e.area = s.area
4 JOIN occupation_table o ON e.occ_code = o.occ_code
5 WHERE s.area_title = 'North Carolina'
6 ORDER BY e.a_mean DESC
7 LIMIT 5;
```

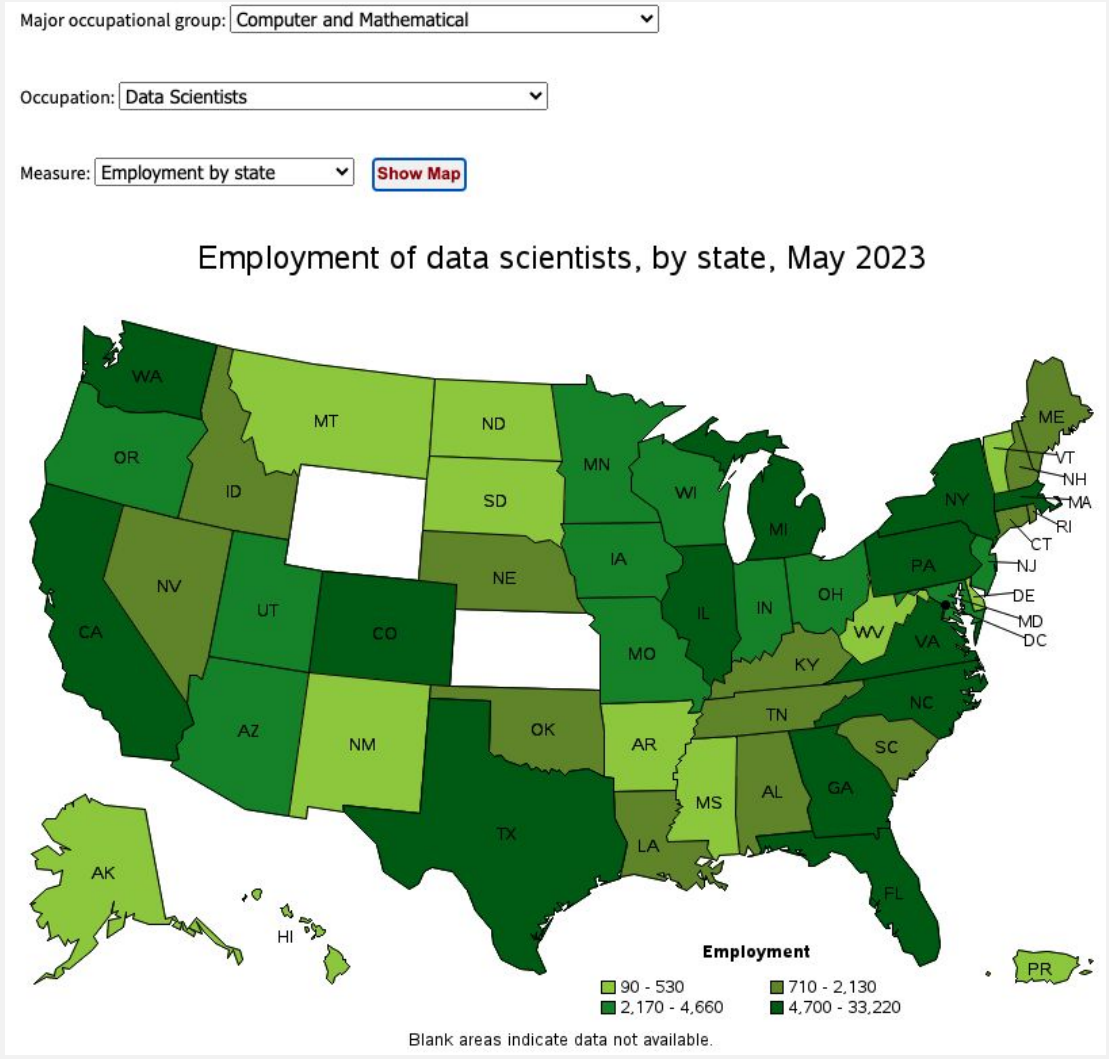
Data Output | Messages | Notifications

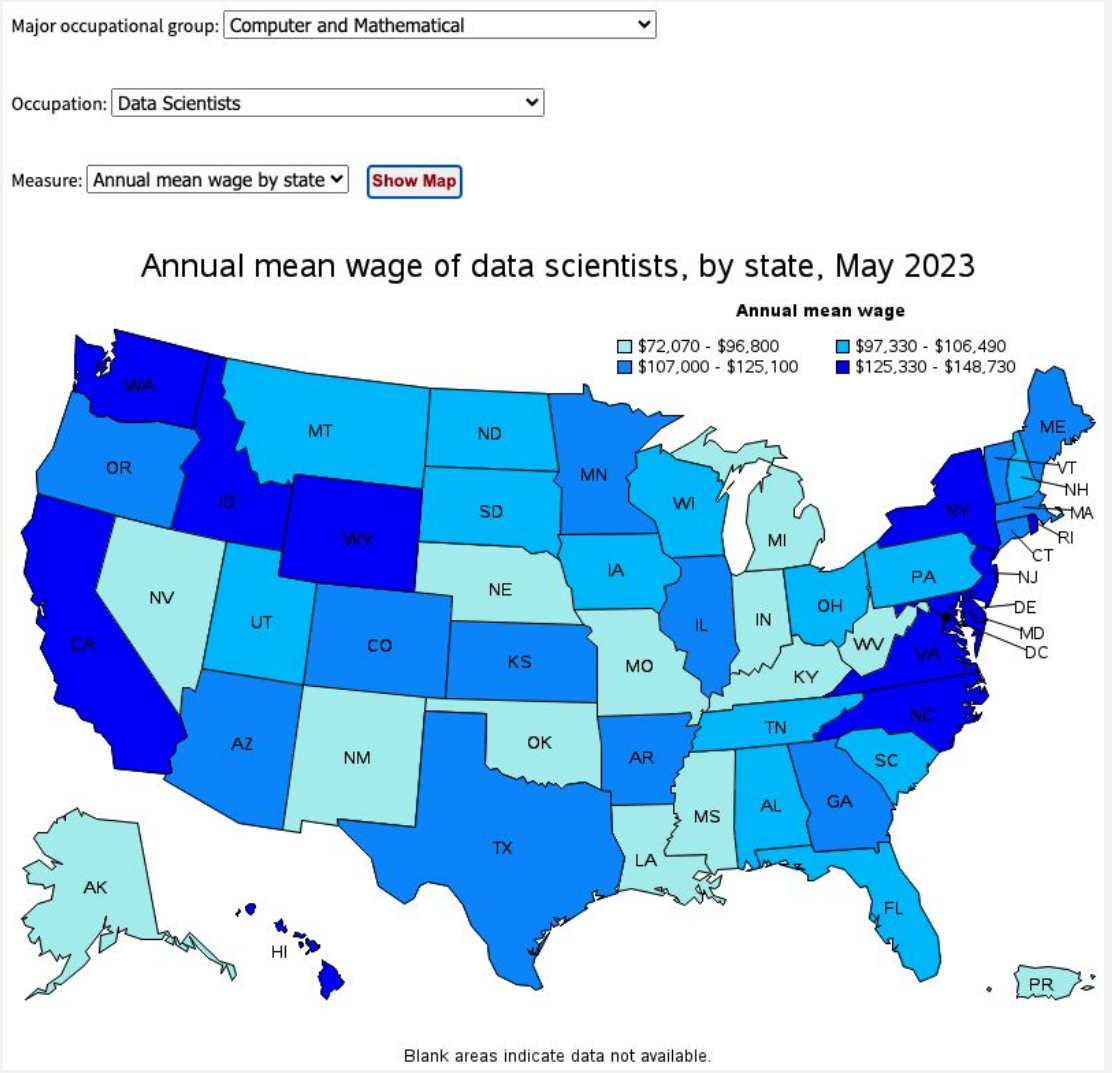
	state character varying (30)	occupation character varying (200)	avg_annual_wage numeric (12,2)
1	North Carolina	Dentists, All Other Specialists	198280.00
2	North Carolina	Natural Sciences Managers	177420.00
3	North Carolina	Computer and Information Systems Managers	176210.00
4	North Carolina	Optometrists	171170.00
5	North Carolina	Financial Managers	168610.00

Things you can do with the database
(Why we chose this dataset?)

- Give job seekers and employers an idea of salary ranges for different occupations in different locations
- Use in data-driven decision making for state workforce outcomes
- Explore career opportunities or assist in career decision making
- Examine wage and employment trends

Data Visualization





https://www.bls.gov/oes/current/map_changer.htm

Questions? Comments? Concerns? Compliments?