# ETL Project

Group 11 - Group Members:

Raymond Galan
Jeongdae JD Kwak
Rainer Perry

**Objective:**
Creation of a database for the cost of living in major cities around the world, combined with a breakdown of various costs and figures related to living in those cities, including: meals, various consumer goods, rent in apartments of various sizes, gym memberships, tuition, and net salaries. We then compared the data to NYC and calculated ratios.

## Data Sets Used:

**Cost of Living index by City 2020 Mid-Year**
https://www.numbeo.com/cost-of-living/rankings.jsp

**Cost of Living Worldwide**
https://www.kaggle.com/morriswongch/cost-of-living

## Libraries Imported:

- Pandas
- SQL Alchemy
- Psycopg2

## EXTRACT

Using Pandas, we retrieved both datasets through their respective URLs into the following dataframes, using the pd.read_html  method:

```
# retrieve data
url = 'https://www.numbeo.com/cost-of-living/rankings.jsp'

response = pd.read_html(url)
```

living_cost_df
breakdown_df

# TRANSFORM

1. The “lliving_cost_df” contained the cost of indexes, so we extracted the following columns and renamed them accordingly (using the .drop and .rename methods):

    - Cost_of_Living_Index'
    - Rent_Index
    - Living_Cost",
    - Groceries_Index",
    - Restaurant_Price_Index
    - Purchasing_Power_Index

```
# retrieve data
url = 'https://www.numbeo.com/cost-of-living/rankings.jsp'

response = pd.read_html(url)
```

```
# clean data
living_cost_df.drop('Rank', axis=1, inplace=True)
living_cost_df.head()
```

2. The “breakdown_df” contained the specific details to be combined with the indexes, hence we extracted the most relevant columns as follows by creating a cleaned-up dataframe with selected columns only, including the following examples:

    “Meal_for_1”’, “Meal_for_2”,  “Bottle_of_Wine”, “Gym_membership”, “Average_Monthly_net_Salary”, and others, a total of 17 items.

```
#rename columns
living_cost_df.rename(columns={
    "Cost of Living Index": 'Cost_of_Living_Index',
    "Rent Index": 'Rent_Index',
    "Cost of Living Plus Rent Index": "Living_Cost",
    "Groceries Index": "Groceries_Index",
    "Restaurant Price Index": "Restaurant_Price_Index",
    "Local Purchasing Power Index": "Purchasing_Power_Index"
}, inplace=True)
```

3. To set NYC as the standard to compare all other city data to, we created a new dataframe that contained the ratios for the individual items, computed with NYC as the standard.

```
# create new table to compare the price values to NYC by dividing and coming up with a ratio.
compare_nyc_df = clean_df.set_index("City")
nyc_value = compare_nyc_df.loc["New York, NY, United States", :].to_list()
compare_nyc_df = compare_nyc_df / nyc_value
```

# LOAD:

The final step was loading the data frames into Postgresql by creating an engine and a connection to SQL.

All three data frames were loaded under the names:

“Living Cost Index”
“Detailed Breakdown”
“Living Cost Compare to NYC”

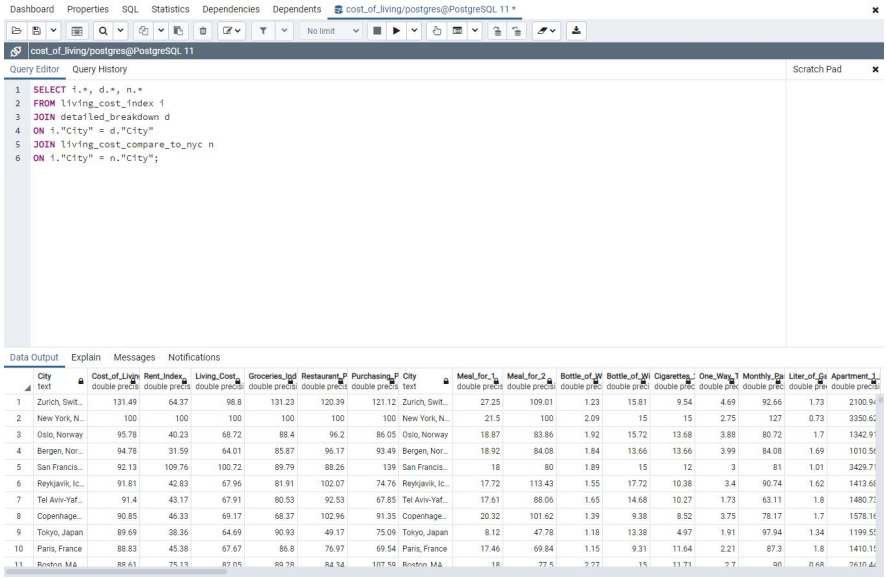
```
# create engine variable
engine = create_engine('postgresql://postgres:[redacted]@localhost:5432/cost_of_living')
conn = engine.connect()
```

```
# check current tables
engine.table_names()
```

```
# create table for living_cost_index
living_cost_df.to_sql(name="living_cost_index", con=engine, index=False)
```

```
# create table for detailed breakdown
clean_df.to_sql(name="detailed_breakdown", con=engine, index=False)
```

```
# create table for living_cost_compare_to_nyc
compare_nyc_df.to_sql(name="living_cost_compare_to_nyc", con=engine, index=True)
```




[image: join of 3 tables in pgAdmin]