Step 1:

- My hypothetical new startup is a an extra-terrestrial themed bakery named "Martin Marshmallow" that is doing really well in our local area. We see a strong relationship due to our area having a lot of UFO sightings and extra-terrestrial events reported along with recent unclassified UFO/extraterrestrial documents that were officially released.
- We just raised some venture funding to expand our reach to other parts of the country, but we want to validate where our next store locations would be.
- As the technical cofounder and sole data engineer I want to build a warehouse of restaurants/bakeries that do well and enrich it with data that is similar theme of our current bakery.
- We have collected some starter data from Yelp's publicly accessible data and some UFO sighting data to begin our product market fit validation to new locations.

Yelp Data (JSON files)	
Business:	209K rows
Checkin:	175K rows
Review:	8M
Tip:	1.3M
User:	1.9M
UFOS (CSV File) :	80k

Roughly 13 Million Rows of Data

- The use case of this data is to create an Analytics Database a Data Analyst or Analytic Engineer to do some analysis and create some business metric's in the future with our new locations in relation to this data we have sourced.
- We want to build a warehouse of the data to continually enrichment with data.

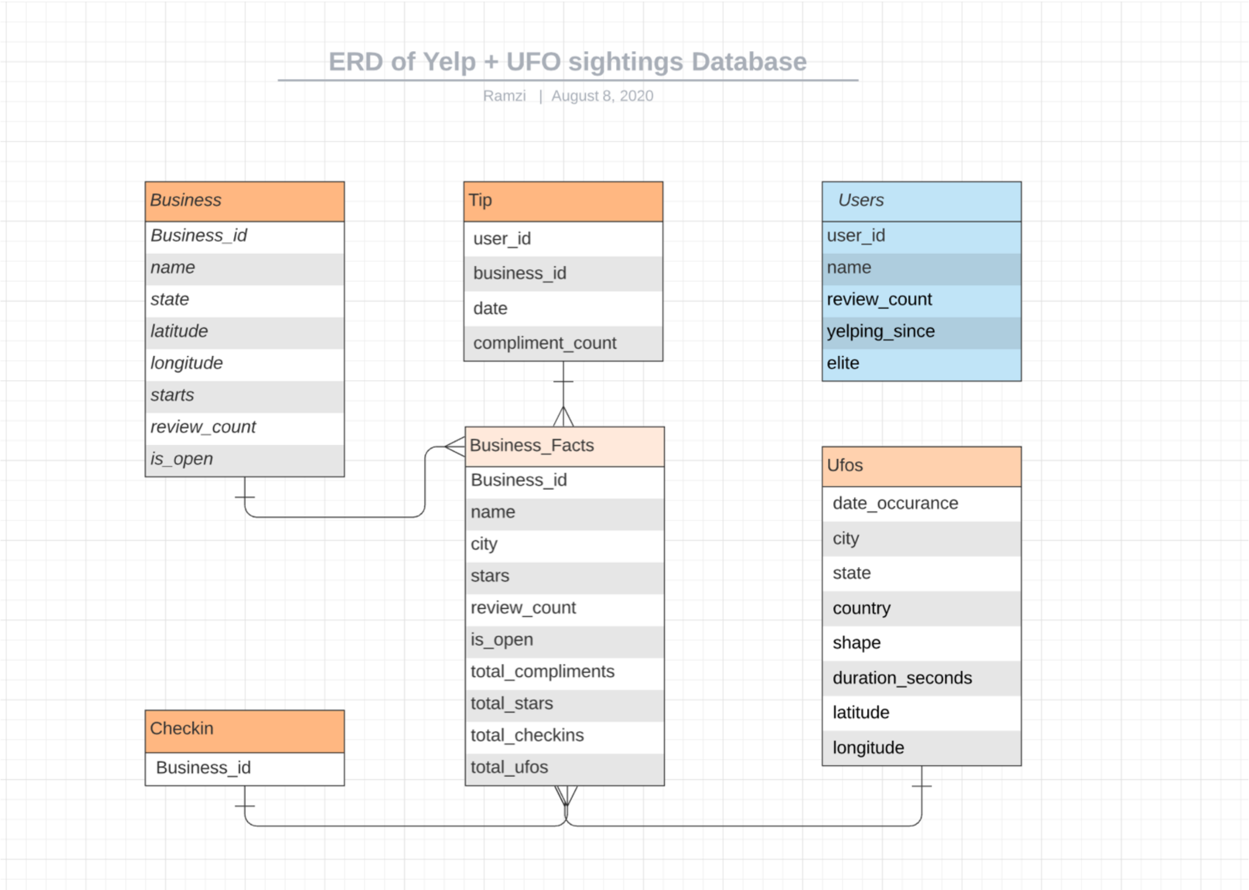
Step 2:
Yelp Data Cleaning/Limiting:

- Reviewing the data, the Yelp Data didn't have any missing data values but it had a lot of data we would not need initially but we might find valuable at a later date. Because we still a relatively young startup we want to run as lean as possible in our investments in our new data warehouse in size and compute so we will limit what data we bring it initially and scale it up as we find more value in the data.
- In my ETL process I will limit what base tables I bring in the staging and finalize production tables.

UFO data cleaning:

- The UFO data was one of the tables that required a lot of cleaning.
    - There were missing values of locations that were missing that I removed
    - Columns of text I didn’t find value that I dropped.
    - I Removed the header of the CSV file in order to read it properly as CSV in airflow (I had difficulty with the header)

Step 3: ERD Diagram and ETL Process.



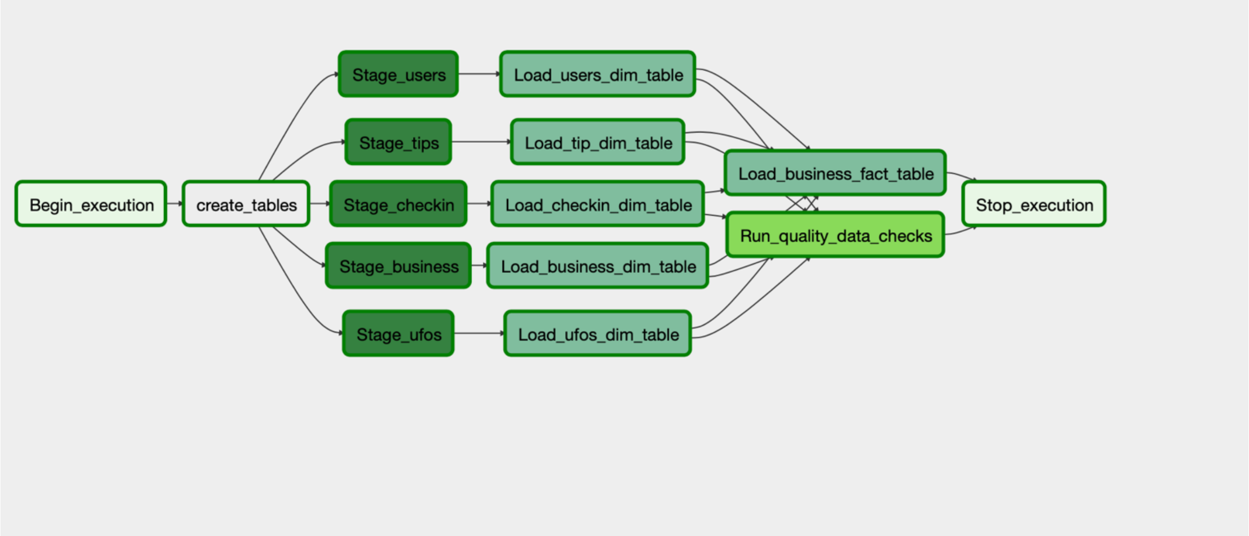
Joining the tables on the business ID, for the UFOS we are joining on the state column.
As for the Users table it’s currently unused but we might utilize it at a later date and will test joining it to the current star schema within the warehouse but will take the time to load it now.

Steps needed for the ETL Process:

1. Create and S3 Bucket and Redshift Warehouse in the same region
2. Add Files to S3 Bucket
    a. Using Boto 3 documented code found below
    b. Manually upload to S3 Bucket.
3. Create an Airflow Instance

    a. Locally on a machine
    b. Using Docker

4. Write the Airflow DAGS, Helpers and Operators.
5. Begin Testing ETL Workflow as Errors and issues occur in the DAG's in airflow cont. to edit and refine the process.
6. Finalize the process and being loading data in the set interval needed.



Data Dictionary:

- Business Table:
    - Business_id: Distinct Business ID
    - Name: Business Name
    - Latitude: Latitude Number for Business
    - Longitude: Longitude number for business
    - Stars: Star Rating for business (averaged)
    - Review_count: number of reviews
    - Is Open: if this business is currently open
- Checkin:
    - Business_id: Business ID that has gotten a checkin
- Tip:
    - User_id: Unique User ID
    - Business_id: Business ID
    - Date: Date of tip
    - Compliment_count: total complients users has given
- Ufos:
    - Date_occurrence: Date of the UFO sighting

  - City: City of UFO Sighting
  - State: State of UFO Sighting
  - Country: Country of UFO
  - Shape: Shape of UFO
  - Duration_seconds: Length of UFO sighting
  - Latitude: Latitude of ufo
  - Longitude: Longitude of UFO

- Users:
  - User_id: Unique User ID
  - Name: User Name
  - Review_count: Number of Reviews
  - Yelping_since: date since account creation
  - Elite: Has had elite status

- Business_Facts:
  - Business_Id: Unique Business ID (distinct)
  - Name: Name of Business:
  - City: City of Business
  - State: State of Business
  - Stars; Average Stars
  - Review_count: Number of Reviews
  - Is_open: Is open
  - Total_compliments: Number of compliments
  - Total_stars: Number of Stars
  - Total_checkins: Number of Checkins
  - Total_ufos: Number of UFOs

Step 5:

- The overall intention is to build a baseline data warehouse to accelerate our growth of our startup using yelp and ufo data.

Data Stack:

- Airflow allows us schedule and add additional data as we scale the warehouse
- S3 is a cheap way to add data and only use what we need.
- Redshift gives a quick query warehouse for analyst to do complex joins and analysis on data and eventually model new tables as we expand.
- Docker allows us to run the airflow instance locally until we need to expand and potentially run it on EC2
- Steps are documented above

Data Scaled to 100X

- We would need to run airflow in smaller batches along with scale up our redshift instance to have a larger capacity. We don’t want to reach a 100% capacity when Redshift is better optimized at a 70-80% capacity. We would also need to be very purposeful in how we model our tables going forward in terms of sort and dist keys in order to optimize query performance.

If Pipelines were run by 7AM Daily:

- We need to understand when the most common query times are if the Dag runs are going to limit the amount of queries I as the data engineer or other analysts can during the time the DAG runs. If they DAG completes at 8AM then perhaps it reasonable for business hours but if the DAG runs all day it can render our redshift instance useless due to the compute taking up so much resources from the DAG runs.

If the Database needed to be accessed by 100+ people:

- Going back to how we create our sort and dist keys need to be really optimized for our warehouse initially for so many people accessing the data. We would also need to begin creating potentially smaller data marts or OLAP cubes of the data for specific teams to get their queries to render quickly.
- This is something I currently deal with at my current job by creating data marts we can ensure dashboards and reports are useable without querying massive tables for example for the marketing team or product team.