Updates from Python scripting

- We have written an outline of the menu options and the function stubs
- Roughly half of the functions are filled out and ready to be tested once our database is ready
- Some of the functions are yet to be determined how to handle because we need to figure out how we should handle lists
    - What to do with list outputs
        - We are leaning towards saving in a pickle file or another runnable format for the admin/user to load in
    - Handling duplicate returns for restaurant_name and location.
        - Probably should make it a “search” format, where we give the user the options and they select
- Also, we are still debating on how to best handle the ranking formula.
    - This will need to be implemented in the database as a default value (for when the user has not rated a restaurant yet)
    - But this means that we need to have default rows for all users and all restaurants, which needs to be updated in our code
    - Also, we need to figure out how the admin can even enter a ranking formula
        - Parsing string into a formula?
- Need to write our function for finding chains and handling that
    - This will likely just call a stored procedure but may necessitate a small update to our DDL, hence tabled at the moment

Updates from DDL

- We have made our DDL
- All tables are fully defined, with following the requirements:
    - There are 6 tables (more than 4), and a lot more than 12 attributes
    - We have identified places for triggers and materialized views instead of having derived relations
    - Keys are consistent (i.e. BIGINT UNSIGNED for FKs based on SERIAL PRIMARY KEYs, etc.)
    - Types are defined reasonably and documented if not obvious (e.g. price_range)
    - NOT NULL and UNIQUE are specified according to the ER diagrams
    - Appropriate attribute names avoiding keywords
    - Documentation for each table and non-obvious attributes
    - A useful index over an attribute that is not a primary or foreign key
- The index is over the location attribute in the restaurant table–justification:
    - Not a primary or foreign key
    - Will be queried on a lot, for restaurant recommendations based on user location, best rated restaurants in particular locations, etc.
- In our original plans, we had certain mutli-valued attributes in the cuisines and categories tables (top restaurants and locations for both), but we have decided to turn these into materialized views

One-on-One Meeting and Other Feedback Incorporation

- Our original diagrams had unnecessary primary keys, i.e. both a unique primary key over user_id, and a primary key on username. We have taken out all instances of this kind of redundancy, and instead used UNIQUE for non-PK attributes like usernames.
- Our original diagrams had lists, non-atomic multi-valued attributes, in certain tables. We have decided to turn these into materialized views instead since in our context, we would be recomputing these and displaying them reasonably frequently.
- Suggestion: turn categories and cuisines into materialized views or compute on the fly. We have actually retained the categories and cuisines tables, to have a dedicated place to keep track of unique types of these items, but some of the columns are indeed planned to be materialized views now.