1. Your database should be pre-populated with sample data. At least three rows per table is expected. The sample data should illustrate a table's functionality, e.g. if the table is part of a many-to-many relationship, the sample data should depict M:M.
2. Your database should have at least 4 entities and at least 4 relationships, 2  of which must be a many-to-many relationship.  The entities and relationships should implement the operational requirements of your project, see individual project steps for weekly instruction.
3. \*Note for this project you will be asked to identify 2 M:M relationships in the planning phase, however will only be required one for your final project execution.
   * Change Doctor\_Patients to Staff\_Patients
   * Change Schema Diagram in pdf
   * Indicate the change in pdf document
4. It should be possible to INSERT entries into every table individually.
5. Every table should be used in at least one SELECT query. For the SELECT queries, it is fine to just display the content of the tables, but your website needs to also have the ability to search using text or filter using a dynamically populated list of properties. This search/filter functionality should be present for at least one entity. It is generally not appropriate to have only a single query that joins all tables and displays them.
6. You need to include one DELETE and one UPDATE function in your website, for any one of the entities. In addition, it should be possible to add and remove things from at least one many-to-many relationship and it should be possible to add things to all relationships. This means you need INSERT functionality for all relationships as well as entities. And DELETE for at least one *many-to-many*relationship.
7. In a one-to-many relationship (like bsg\_people to bsg\_planets), you should be able to set the foreign key value to NULL (such as on a person in bsg\_people), that removes the relationship. In case none of the one-to-many relationships in your database has partial participation, you would need to change that to make sure they can have NULL values.
8. In a many-to-many relationship, to remove a relationship one would need to delete a row from a table. That would be the case with bsg\_people and bsg\_certifications. One should be able to add and remove certifications for a person without deleting either bsg\_people rows or bsg\_certification rows. If you implement DELETE functionality on at least (1) many-to-many relationship table, such that the rows in the relevant entity tables are not impacted, that is sufficient.
9. Other Work
   * Update DDQ and DMQ with queries used in backend.py
   * Post Step 6 Reviews to PDF
   * New PDF based on Step 7 guide