Team: **Bryan Wu**, **Susan Joh**

Team name: Busy Beavers

Project title: **Busy Beaver Burgers**

Project URL: **https://web.engr.oregonstate.edu/~johsus/CS340/index.html**

https://flip1.engr.oregonstate:9544

Project Outline and Database Outline, ERD, Schema & Sample Data Updated Version

Overview:

Busy Beaver Burgers (BBB) is a family-owned fast food restaurant that wants to implement a website with a database backend to keep track of its **orders** and **customers**. Busy Beaver Burgers serves roughly 500 customers a day and at the moment only takes to-go orders, since it just opened. BBB also wants to know which dishes are popular at the moment, so they asked us to implement a rating system from 1 to 5. Currently, the restaurant has 7 **dishes** on the menu and on average 200 **ratings** per dish. Once we keep track of all of the ratings on each dish, we can find the highest rated dish and reorganize the menu accordingly. Additionally, with the new database, we can keep track of a customer's **dietary restrictions** so they can be alerted if they order something that they might be allergic to. We can now record how much money each customer has spent and generate reports of what dishes they ordered and when they ordered the dishes. These reports will be beneficial to making business decisions and they fall under the realm of business analytics. Overall, establishing a backend database for Busy Beaver Burgers will point the restaurant in the right direction financially for the future.

Project Database Outline Updated:

- Customers: The customers that order food or will prospectively order food from Busy Beaver Burgers. We want to record customer details to provide a more customized experience (especially with regards to allergies). Additionally, we want to keep track of customers who leave poor reviews to follow up with them afterwards.
    - customer_id: int, auto_increment, unique, not NULL, PK
    - customer_name: varchar(50), not NULL
    - phone_number: varchar(15), unique, not NULL
    - Relationships:
        - 1:M relationship with Orders, customer_id is an FK in Orders
        - (optional) M: (optional) M relationship with Dietary_Restrictions
        - 1: (optional) M relationship with Ratings, customer_id is an FK in Ratings

- Orders: Orders are made by customers, and they are typically transactions, meaning they are recorded and generally not changed. When recorded, they contain the date and time of the transaction, the customer that ordered the food, the quantity of each dish ordered, and the cumulative price of the food (not shown, because it is a calculation of individual food items).
    - order_id: int, auto_increment, unique, not NULL, PK
    - time: datetime, not NULL
    - customer_id: int not NULL, FK
    - **We removed dish_quantity from Orders to comply with 3NF, since dish_quantity was directly derived from the number of dishes and thus was a transitive dependency.**
    - Relationships:
        - M:1 relationship with Customers, customer_id is an FK in Orders
        - (optional) M:M relationship with Dishes

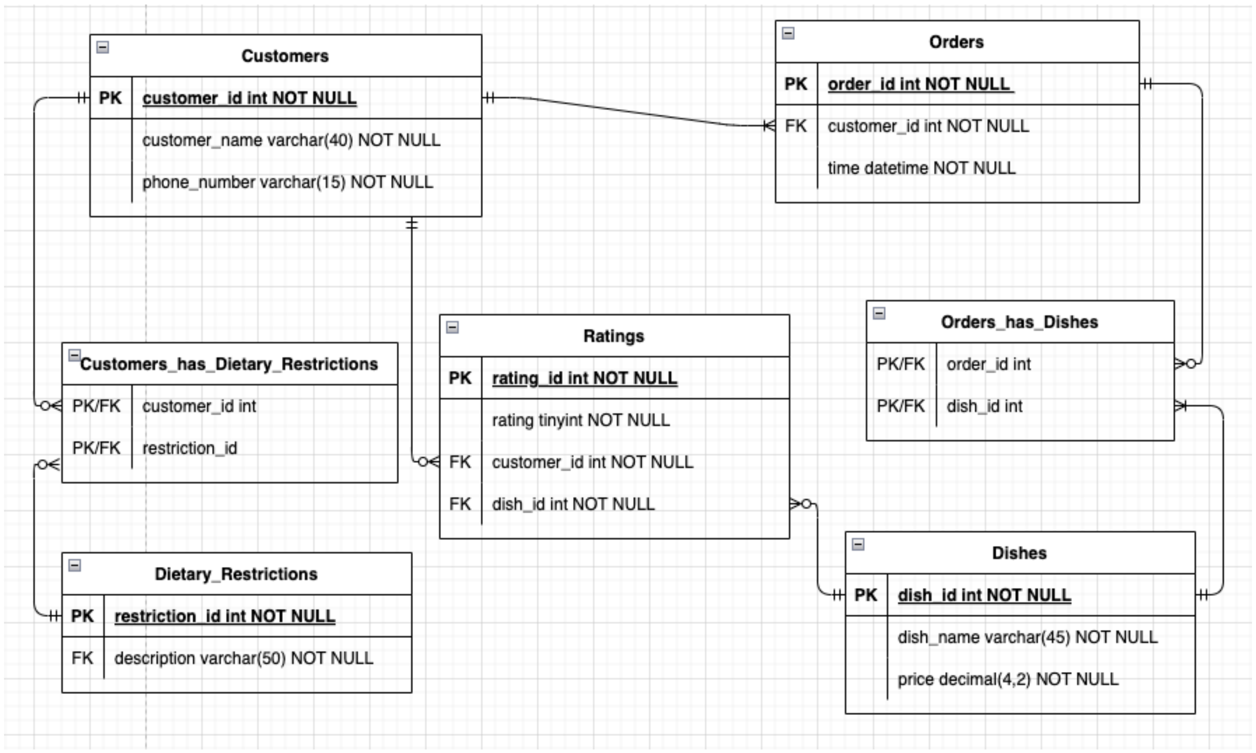
- Dishes: These are the dishes on the menu, of which there are currently 7. Each dish has a name, price, order quantity (the number of the same dish that is ordered in one order). Each dish also can be associated with one or more dietary restrictions.
    - dish_id: int, auto_increment, unique, not NULL, PK
    - dish_name: varchar(45) NOT NULL
    - price: decimal(4, 2), not NULL
    - Relationships:
        - 1:(optional) M relationship with Ratings
        - **We removed the relationship with Dietary_Restrictions**
        - M: (optional) M relationship with Orders

- Dietary_Restrictions: The dietary restrictions table is a category table that lists categories such as “vegan”, “gluten-free”, etc., and it describes foods on the menu, as well as customers should they choose to list their dietary preferences.
    - restriction_id: int, auto_increment, unique, not NULL, PK
    - description: varchar(40), not NULL
    - Relationships:
        - (optional) M:M relationship with Customers
        - **We removed the relationship with Dishes**

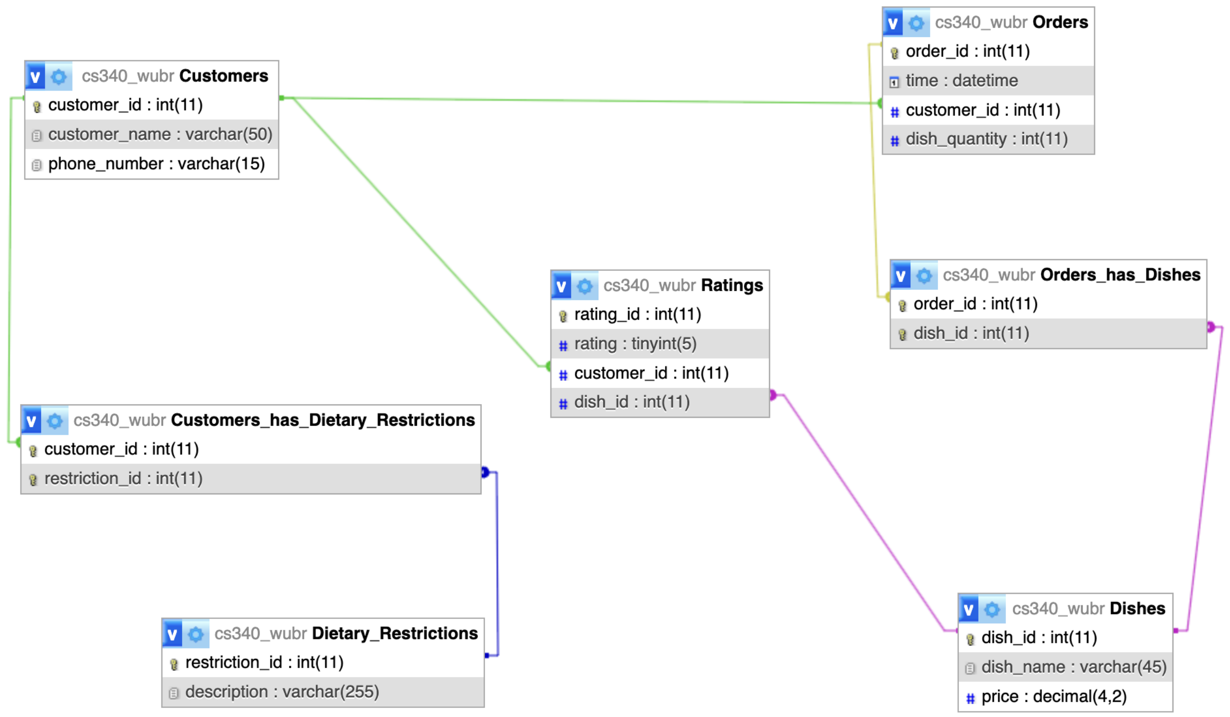
- Ratings: Ratings are between 1 and 5, and they give the restaurant an idea of which dishes are popular at the moment.
    - rating_id: int, auto_increment, unique, not NULL, PK
    - rating: TINYINT(5), not NULL
    - customer_id: int, not NULL, FK
    - dish_id: int, not NULL, FK
    - Relationships:
        - (optional) M:1 relationship with Customer, customer_id is an FK in Ratings

- (optional) M:1 relationship with Dishes, dish_id is an FK in Ratings

Entity Relationship Diagram:


Customers
PK customer_id int NOT NULL
customer_name varchar(40) NOT NULL
phone_number varchar(15) NOT NULL
Orders
PK order_id int NOT NULL
FK customer_id int NOT NULL
time datetime NOT NULL
Customers_has_Dietary_Restrictions
PK/FK customer_id int
PK/FK restriction_id
Ratings
PK rating_id int NOT NULL
rating tinyint NOT NULL
FK customer_id int NOT NULL
FK dish_id int NOT NULL
Orders_has_Dishes
PK/FK order_id int
PK/FK dish_id int
Dietary_Restrictions
PK restriction_id int NOT NULL
FK description varchar(50) NOT NULL
Dishes
PK dish_id int NOT NULL
dish_name varchar(45) NOT NULL
price decimal(4,2) NOT NULL

Schema:


cs340_wubr Customers
customer_id : int(11)
customer_name : varchar(50)
phone_number : varchar(15)
cs340_wubr Orders
order_id : int(11)
time : datetime
customer_id : int(11)
dish_quantity : int(11)
cs340_wubr Ratings
rating_id : int(11)
rating : tinyint(5)
customer_id : int(11)
dish_id : int(11)
cs340_wubr Orders_has_Dishes
order_id : int(11)
dish_id : int(11)
cs340_wubr Customers_has_Dietary_Restrictions
customer_id : int(11)
restriction_id : int(11)
cs340_wubr Dietary_Restrictions
restriction_id : int(11)
description : varchar(255)
cs340_wubr Dishes
dish_id : int(11)
dish_name : varchar(45)
price : decimal(4,2)

Example Data:

Customers		
customer_id	customer_name	phone_number
1	John Mayer	(212) 555-1234
2	Ethan Wright	(312) 555-5678
3	Samantha Lee	(415) 555-9012
4	Alexander Patel	(305) 555-2345
5	Olivia Kim	(206) 555-6789

Orders		
order_id	time	customer_id
1	2023-05-02 10:15:30	1
2	2023-05-01 10:20:45	3
3	2023-04-30 09:00:00	3
4	2023-04-29 13:59:59	4

Dishes		
dish_id	dish_name	price
1	Hamburger	3.50
2	Cheeseburger	3.70
3	Fries	2.00
4	Tater tots	2.50

Orders_has_Dishes	
order_id	dish_id
1	1
2	2
3	1
3	3
4	2

Ratings			
rating_id	rating	customer_id	dish_id
1	5	1	1
2	4	3	2
3	5	3	1
4	3	4	2

Customers_has_Dietary_Restrictions	
customer_id	restriction_id
1	2
1	3
2	3
3	3

Dietary_Restrictions	
restriction_id	description
1	Vegetarian
2	Gluten-free
3	Dairy-free

Fixes based on Feedback from Previous Steps

Step 3 Feedback:

#1:

- *Does the UI utilize a SELECT for every table in the schema?* In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.
  - Yes! All tables have a corresponding select. However, I do not think that the intersection tables need to be present on the UI. I am not 100% sure on this though. It looks like the customers_has_dietary_restrictions was left out, but I'm not sure if you need to add it in since I don't think we need intersection tables in the UI. It is present in the SQL. There are 7 tables in the ER diagram and 8 in the schema, so you may want to go back and update either the diagram or the schema with the version that is correct.
- *Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?*
  - No, this has not yet been implemented.
- *Does the UI implement an INSERT for every table in the schema?* In other words, there should be UI input fields that correspond to each table and attribute in that table.
  - Yes, every page has an insert capability.
- *Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?* In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line_total).
  - I don't think this has been implemented. Each intersection table can be updated, but it does not look like it's automatically updated when the tables it connects are updated.
- *Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship?* In other words, if an order is deleted from the Orders table, it

should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.
    - Yes! The delete functionality is implemented.
- *Is there at least one UPDATE for any one entity?* In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?
    - Yes! All tables have update functionality.
- *Is at least one relationship NULLable?* In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus it should be feasible to edit an Order and change the value of Employee to be empty.
    - Yes! There are multiple nullable relationships. An example is between customers and ratings where ratings is optional.
- *Do you have any other suggestions for the team to help with their HTML UI? For example using AS aliases to replace obscure column names such as fname with First Name.*
    - No.

#2:

- *Does the UI utilize a SELECT for every table in the schema?* In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.
    - Yes, each table presented in the schema appears on the UI.
- *Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?*
    - No, I did not see one.
- *Does the UI implement an INSERT for every table in the schema?* In other words, there should be UI input fields that correspond to each table and attribute in that table.
    - Yes. each table includes an insert in the UI.

- *Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?* In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line_total).
    - There are fields for the user to add the corresponding FK on an insert, but the user has to manually enter the FK. It does not appear that a new row is added to the relative intersection tables when a new entry is made to corresponding tables.
- *Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship?* In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.
    - All entities include deletes and it appears that all deletes are of the form ON DELETE CASCADE.
- *Is there at least one UPDATE for any one entity?* In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?
    - Yes, all entities include updates.
- *Is at least one relationship NULLable?* In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus it should be feasible to edit an Order and change the value of Employee to be empty.
    - It appears that all attributes are NOT NULL so there would be no way with the current setup to make a relationship nullable. However, there is a comment in the create Orders section that if a customer is deleted they still want to have the order available so the intention is there but the constraint of NOT NULL on customer ID may need to be updated.
- *Do you have any other suggestions for the team to help with their HTML UI? For example using AS aliases to replace obscure column names such as fname with First Name.*

    - The project is looking good! I am impressed that you chose to implement insert/delete/updates for each entity. As this instruction mentions, I do suggest adding the “as alias” to the SQL for the inserts so that the tables in the UI match the column titles from the entities.

#3:

- *Does the UI utilize a SELECT for every table in the schema?* In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.
    - The HTML tables model the use of SELECT from all DB tables in the schema except the intersection table “Dietary_Restrictions_has_Dishes.” All other tables include all listed schema attributes.
- *Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?*
    - While individual attributes are named as opposed to wildcard use in the DML.sql , there are no table joins in the select operations. This is recommended to connect IDs to names.
- *Does the UI implement an INSERT for every table in the schema?* In other words, there should be UI input fields that correspond to each table and attribute in that table.
    - All tables presented in the UI support INSERT/Create functionality with the exception of the intersection table “Customer_has_Dietary_Restrictions” which includes the “New” button, but does not allow creation. This is a project requirement and is suggested to be implemented.
- *Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?* In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line_total).
    - Intersection table “Orders_has_Dishes” includes this functionality in both the UI and DML. The intersection table

“Customer_has_Dietary_Restrictions” has this functionality included in the DML but the table is not included in the UI.

- *Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship?* In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.
    - Delete is included in the UI and also provides a confirmation (very nice!). The DDL does include “ON DELETE CASCADE” functionality.
- *Is there at least one UPDATE for any one entity?* In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?
    - UPDATE functionality is incorporated into all tables except the intersection table “Customers_has_Dietary_Restrictions.”
- *Is at least one relationship NULLable?* In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus it should be feasible to edit an Order and change the value of Employee to be empty.
    - All foreign keys are defined as NOT NULL in the DDL. I would recommend allowing optional relationships with Orders in the case where the customer is not in the system and wants to remain anonymous or in Ratings where a customer wants to remain anonymous (even though this may lead to troublesome ratings w/out accountability).
- *Do you have any other suggestions for the team to help with their HTML UI? For example using AS aliases to replace obscure column names such as fname with First Name.*
    - While it is easy to assume column names to be directly associated with their UI table, in the case of multiple IDs I had to avert my eyes from the table to the table title. The names could be more specific detailing the columns so they might stand alone.

#4:

- Does the UI utilize a SELECT for every table in the schema? In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.
    - Yes, each table in the schema is displayed on the UI.
- Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?
    - No, there isn't a select, but it could be ID to names.
- Does the UI implement an INSERT for every table in the schema? In other words, there should be UI input fields that correspond to each table and attribute in that table.
    - Yes, each table implements an insert in the UI except the "Customer_has_Dietary_Restrictions" table. It has a new button but it doesn't work yet.
- Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship? In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line_total).
    - "Orders_has_Dishes" does in UI and DML, but "Customer_has_Dietary_Restrictions" only does on the DML.
- Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship? In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.
    - Yes, all entities include updates and includes ON DELETE CASCADE.
- Is there at least one UPDATE for any one entity? In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?
    - Yes, it is on all tables besides "Customers_has_Dietary_Restrictions".

- Is at least one relationship NULLable? In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus it should be feasible to edit an Order and change the value of Employee to be empty.
    - No, all are defined as NOT NULL. One suggestion could be to have the option for employees or customers to be empty or optional like it says.
- Do you have any other suggestions for the team to help with their HTML UI? For example using AS aliases to replace obscure column names such as fname with First Name.
    - The only other suggestion I have would be to change the column names so that you are easily able to understand what they are, for example, under “orders” you could make it “order date and time” so that you know exactly what it is.

#5 (Feedback from grader):

- Missing Dietary_Restrictions_Has_Dishes

Step 3 Summary of Changes:

Actions based on feedback:
From feedback #1:

- [UI SELECT:] All tables have a corresponding select. However, I do not think that the intersection tables need to be present on the UI. I am not 100% sure on this though. It looks like the customers_has_dietary_restrictions was left out, but I'm not sure if you need to add it in since I don't think we need intersection tables in the UI. It is present in the SQL. There are 7 tables in the ER diagram and 8 in the schema, so you may want to go back and update either the diagram or the schema with the version that is correct.
  [response:] The Dietary_Restrictions_has_Dishes table has been removed from the outline, the sample data, and the data manipulation queries. It was kept in the schema on accident, but we have since updated it. As for the intersection tables on the UI, we have removed them and implemented them on the corresponding entity pages.

- [UI SELECT search/filter:] No, [the SELECT search/filter] has not yet been implemented.
  [response:] We have since implemented a SELECT statement with a filter for both customers whose name is "Olivia Kim" and for all orders of hamburgers.

- [UI INSERT:] I don't think [INSERT FK attributes, including at least one M:M relationship] has been implemented. Each intersection table can be updated, but it does not look like it's automatically updated when the tables it connects are updated.
  [response:] The update to the table will be implemented in a later step.

From feedback #2:

- [UI SELECT search/filter:] No, I did not see [a SELECT utilizing a search/filter].
  [response:] Repeated suggestion.

- [UI INSERT:] There are fields for the user to add the corresponding FK on an insert, but the user has to manually enter the FK. It does not appear that a new row is added to the relative intersection tables when a new entry is made to corresponding tables.
  [response:] To our knowledge, there is no way to automatically insert rows into two tables with only one insert statement. As such, we added two insert statements whenever an intersection table is involved. One for the entity and one for the intersection table.

- [NULLable:] It appears that all attributes are NOT NULL so there would be no way with the current setup to make a relationship nullable. However, there is a comment in the create Orders section that if a customer is deleted they still want to have the order available so the intention is there but the constraint of NOT NULL on customer ID may need to be updated.
  [response:] We have updated the Orders table to accommodate for the deletion of a customer. Now, when a customer is deleted, the customer_id in the Orders table is set to NULL using ON DELETE SET NULL.

- [UI Other:] The project is looking good! I am impressed that you chose to implement insert/delete/updates for each entity. As this instruction mentions, I do suggest adding the “as alias” to the SQL for the inserts so that the tables in the UI match the column titles from the entities.
  [response:] We have implemented the AS keyword in all of our SELECT statements.

From feedback #3:

- [UI SELECT:] The HTML tables model the use of SELECT from all DB tables in the schema except the intersection table “Dietary_Restrictions_has_Dishes.” All other tables include all listed schema attributes.
  [response:] Repeated suggestion.

- [UI SELECT search/filter:] While individual attributes are named as opposed to wildcard use in the DML.sql , there are no table joins in the select operations. This is recommended to connect IDs to names.
  [response:] We have implemented table joins, as well as search and filter queries.

- [UI INSERT:] All tables presented in the UI support INSERT/Create functionality with the exception of the intersection table "Customer_has_Dietary_Restrictions" which includes the "New" button, but does not allow creation. This is a project requirement and is suggested to be implemented.
  [response:] We have fixed the functionality of the "New" button in the UI page for "Customer_has_Dietary_Restrictions"

- [UI INSERT FK attributes/M:M:] Intersection table "Orders_has_Dishes" includes this functionality in both the UI and DML. The intersection table "Customer_has_Dietary_Restrictions" has this functionality included in the DML but the table is not included in the UI.
  [response:] The UI for the "Customer_has_Dietary_Restrictions" has been updated.

- [UI UPDATE:] UPDATE functionality is incorporated into all tables except the intersection table "Customers_has_Dietary_Restrictions."
  [response:] The UI for the "Customer_has_Dietary_Restrictions" has been updated.

- [NULLable:] All foreign keys are defined as NOT NULL in the DDL. I would recommend allowing optional relationships with Orders in the case where the customer is not in the system and wants to remain anonymous or in Ratings where a customer wants to remain anonymous (even though this may lead to troublesome ratings w/out accountability).

[response:] Repeated suggestion.

- [UI Other:] While it is easy to assume column names to be directly associated with their UI table, in the case of multiple IDs I had to avert my eyes from the table to the table title. The names could be more specific detailing the columns so they might stand alone.
  [response:] Because SQL actually does not let column names be more than one word, we named things using camelCase. We did rename all the SELECT statement columns

From feedback #4:

- [UI SELECT search/filter:] No, there isn't a select, but it could be ID to names.
  [response:] We have since implemented multiple SELECT statements with filters.

- [UI INSERT:] Yes, each table implements an insert in the UI except the "Customer_has_Dietary_Restrictions" table. It has a new button but it doesn't work yet.
  [response:] Repeated suggestion.

- [UI INSERT FK attributes/M:M:] "Orders_has_Dishes" does in UI and DML, but "Customer_has_Dietary_Restrictions" only does on the DML.
  [response:] Repeated suggestion.

- [NULLable:] No, all are defined as NOT NULL. One suggestion could be to have the option for employees or customers to be empty or optional like it says.
  [response:] Repeated suggestion.

- [UI Other:] The only other suggestion I have would be to change the column names so that you are easily able to understand what they are, for example, under "orders" you could make it "order date and time" so that you know exactly what it is.

[response:] SQL does not allow column names to be longer than one word. Accordingly, we named things in camelCase.

Feedback from grader:

- Missing Dietary_Restrictions_Has_Dishes
  [response:] We decided to remove the Dietary_Restrictions_Has_Dishes table (by removing the relationship between Dietary_Restrictions and Dishes), but mistakenly made the wrong edit to the outline. We fixed our outline to reflect our update. We also updated our ERD.

Step 2 Feedback (comments on areas of improvement highlighted):

#1:

- Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?
  - Yes, the schema and ER diagram are consistent.
- Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?
  - Capitalization and plural/singular looks good. Only change maybe Dietary_Restrictions.restriction_id to dietary_restriction_id
- Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?
  - The schema is easy to read and follow along.
- Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?
  - Multiple intersection tables are properly formed.
- Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?
  - There are no non-normalized issues in the sample data
- Is the SQL file syntactically correct? This can be easily verified by using PhPMyAdmin and your CS 340 database (do not forget to take backup of your own database before you do this!)
  - Multiple lines for Primary key in the intersection table, use this instead:
    - PRIMARY KEY (customer_id, restriction_id)
  - Instead of "FOREIGN KEY (...) FROM .... ON DELETE CASCADE", it should be "FOREIGN KEY (...) REFERENCES .... ON DELETE CASCADE"
  - Typo in create table statement for Dietary_Restriction, row 44, "NUL"
  - Unable to create Orders and Ratings table due to "ON DELETE SET NULL"

    - In the create table statement for Dish, needs to update "name" to "dish_name" to match the PDF
    - Typo in insert statement for "Customers_has_Dietary_Restrictions"
    - Last two lines are causing an error
        - SET FOREIGN_KEY_CHECKS = 1;
        - COMMIT;
- In the SQL, are the data types appropriate considering the description of the attribute in the database outline?
    - Yes, the data types are appropriate.
- In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared?
    - The primary key and foreign keys are correctly defined, but need to fix typo for the CASCADE operations.
- In the SQL, are relationship tables present when compared to the ERD/Schema?
    - Yes, the relationship are present.
- In the SQL, is all example data shown in the PDF INSERTED?
    - Just missing the last record for the Customers table in the SQL statement.
- Is the SQL well structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)?
    - Well structured and enough comments to follow.

#2:

Good job overall! I would make sure the SQL runs and fix some syntax errors.

- Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?

  Yes, the outline and the ER diagram are entirely consistent.
- Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

  Capitalization and use of plural/singular looks good.

- Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?

  The schema is clear and easy to read.
- Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?

  Yes, there are two FKs for each M:N relationship table.
- Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?

  Does not appear to have normalization issues in the sample data.
- Is the SQL file syntactically correct? This can be easily verified by using PhPMyAdmin and your CS 340 database (do not forget to take backup of your own database before you do this!)

  There were some syntax errors.

  PRIMARY KEY (customer_id, restriction_id)
    - Typo in insert statement for "Customers_has_Dietary_Restrictions"
    - Last two lines are causing an error
      - SET FOREIGN_KEY_CHECKS = 1;
      - COMMIT;
    - Instead of "FOREIGN KEY (...) FROM .... ON DELETE CASCADE", it should be "FOREIGN KEY (...) REFERENCES .... ON DELETE CASCADE"
    - Typo in create table statement for Dietary_Restriction, row 44, "NUL"
    - Unable to create Orders and Ratings table due to "ON DELETE SET NULL"
- In the SQL, are the data types appropriate considering the description of the attribute in the database outline?

  Yes, the data types are appropriate and match the outline.
- In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared?

Yes, except there is a typo with the cascade operation.

- In the SQL, are relationship tables present when compared to the ERD/Schema?

  Yes, all relationship tables are present.
- In the SQL, is all example data shown in the PDF INSERTED?

  Looks like the last insert statement is not reflected in the pdf.
- Is the SQL well structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)?

  Yes the sql is easy to follow and well structured with comments.

#3 (Feedback from grader):

Everything looks good so far but I think you might be missing 1 table of sample data and 1 set of insert statements!

Step 2 Summary of Changes:

Actions based on feedback and Upgrades to the draft version:

From feedback #1:

- [naming consistency] Only change maybe Dietary_Restrictions.restriction_id to dietary_restriction_id
  [response] We believe that restriction_id is descriptive enough as there are no other "restrictions" in our data.

- [SQL syntax] Multiple lines for Primary key in the intersection table, use this instead: PRIMARY KEY (customer_id, restriction_id)
  [response] We updated our SQL file to reflect this.

- [SQL syntax] Instead of "FOREIGN KEY (...) **FROM** .... ON DELETE CASCADE", it should be "FOREIGN KEY (...) **REFERENCES** .... ON DELETE CASCADE"
  [response] We updated our SQL file to reflect this.

- [SQL syntax] Typo in create table statement for Dietary_Restriction, row 44, "NUL"
  [response] We changed "NUL" to "NULL".

- [SQL syntax] Unable to create Orders and Ratings table due to "ON DELETE SET NULL"
  [response] Updated "ON DELETE SET NULL" to "ON DELETE CASCADE"

- [SQL syntax] In the create table statement for Dish, needs to update "name" to "dish_name" to match the PDF
  [response] Updated "name" to "dish_name" in the create table statement for Dish

- [SQL syntax] Typo in insert statement for "Customer**s**_has_Dietary_Restrictions"
  [response] Corrected to include plural form.

- [SQL syntax] Last two lines are causing an error
  SET FOREIGN_KEY_CHECKS = 1;
  COMMIT;
  [response] Removed line ‘COMMIT;’

- [SQL ex. data] missing the last record for the Customers table in the SQL statement
  [response] Added the last record for the Customers table in the SQL file.

From feedback #2:

- [SQL syntax] PRIMARY KEY (customer_id, restriction_id)
  [response] Repeated suggestion.

- [SQL syntax] Typo in insert statement for "Customer**s**_has_Dietary_Restrictions"
  [response] Repeated suggestion.

- [SQL syntax] Last two lines are causing an error
  SET FOREIGN_KEY_CHECKS = 1;
  COMMIT;
  [response] Repeated suggestion.

- [SQL syntax] Instead of "FOREIGN KEY (...) **FROM** .... ON DELETE CASCADE", it should be "FOREIGN KEY (...) **REFERENCES** .... ON DELETE CASCADE"
  [response] Repeated suggestion.

- [SQL syntax] Typo in create table statement for Dietary_Restriction, row 44, "NUL"
  [response] Repeated suggestion.

- [SQL syntax] Unable to create Orders and Ratings table due to "ON DELETE SET NULL"
  [response] Repeated suggestion.

- [SQL ex. data] Looks like the last insert statement is not reflected in the pdf.
  [response] Included the last insert statement in the PDF.

From grader:

- missing 1 table of sample data and 1 set of insert statements
  [response] Suggested by #1 and #2 respectively. Updates were made to the SQL file and PDF.

Other updates:

- Updated example data table name in PDF from "Customer_has_Dietary_Restrictions" to "Customer**s**_has_Dietary_Restrictions"
- We decided to remove the relationship between the Dietary_Restrictions and Dishes entities. By declaring that a dish has a dietary restriction would mean that each of that dish has that restriction. Instead we will get the information about dietary restrictions from the Customers entity (connection would be Dishes -> Orders_has_Dishes -> Orders -> Customers -> Customers_has_Dietary_Restrictions). We updated our ERD, schema, SQL file, and outline to reflect this change.

Step 1 Feedback (comments on areas of improvement highlighted):

#1:

“Looks good so far.

The overview is very clear about what problem is being solved and detailed on the scale and scope of the solution.

The outline is detailed and contains enough entities. Each entity is fully described per requirements. For Customers, should the phone number be a char(N)? To me, it seems ike order_quantity doesn't belong in Dishes but in Orders, since different orders would have different quantities of a dish in it. For DietaryRestrictions, it would seem customers would have restrictions on Ingredients, and Dishes would either contain those ingredients or not.

For the ER diagram, you might have some optional (O-|-) relationships, for example, between dishes and ratings (a new dish might have no ratings at first). The M:M relationships are there, including between Orders and Dishes, mitigated by the Orders_has_dishes table.”

#2:

“Great discussion of the problem you are trying to solve. It was very clear, and what a fun concept!

I only have a few small things (and there more just questions than anything else.)

In the Customers entity, your customer entity only has 30 characters. I think that might be a little short? I looked this up for my project, and it suggested 35 characters per name (first and last).

Maybe I'm missing something, but under Dishes you have an attribute order_quantity that's listed as an FK, but I'm not seeing it elsewhere. What is this linked to?

You have probably already thought about this, but for your rating int, would there need to be some sort of constraint for the datatype? Whether that be a lower or upper bound, or even if you want a certain number of decimals etc? I'm not sure what the protocol would be, but it did make me curious.

Great job so far!”

#3:

“Here's my review,

- **Does the overview describe what problem is to be solved by a website with DB back end?**

While your project doesn't specifically call out the problem is being solved via a website, it is definitely implied, and the database is called out also which is excellent!

- **Does the overview list specific facts?**

Lots of facts are specified which is awesome! One thing I would try to quantify is the 'total number of unique customers' because that will effect how you will ultimately size your DB, as well as your attribute types potentially.

- **Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities?**

The entities are well thought out and the data types and constraints are listed and look good. The inclusion of auto increments for customer id was a good move from what we've learned in the lectures. The relationships are spelled out also and look good.

- **Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?**

The relationships look correctly formulated and there are a few M:M relationships. The ER diagram looks good, In the instructions its called out as not being necessarily wrong for including your intersection tables in the ER, but that its preferred to use them later on. Just a nitpick, because I know the limitation of workbench wanting to solve the M:M with a intersection table.

- **Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?**

It looks like y'all did a good job with naming in the overview and entities, as well as the pluralization and singularity of the attributes, as well as the capitalizations. Nice job Great job!!”

#4 (TA comment):

“Hey team, good first draft! Going forward, remember to add the names of the team members and group number in the ED post. Additionally, I would also consider revising your ERD which should be less of a technical blueprint (aka a schema diagram) and more of a high-level logical overview of the database entities and their relationships. ERD diagrams often leave out the finer details such as attributes and intersection tables as its main focus is to demonstrate the rough structure and interconnectivity of the database tables. I would also consider removing some of the extra intersection tables from your project. I worry that implementing so many of these tables further in the quarter will be incredibly time consuming. I enjoy the effort you both put into the draft but I don't want you to have to spend all of your time on this class in the future!"

Step 1 Summary of Changes:

Actions based on feedback and Upgrades to the draft version:

From feedback #1:

- "For Customers, should the phone number be a char(N)?"
  [response] We agree, and have decided to store the phone number as a varchar(n).

- "To me, it seems like order_quantity doesn't belong in Dishes but in Orders, since different orders would have different quantities of a dish in it"
  [response] We removed order_quantity from our database.

- "For DietaryRestrictions, it would seem customers would have restrictions on Ingredients, and Dishes would either contain those ingredients or not"
  [response] Because we don't have an entity table tracking ingredients, we are only tracking basic dietary restriction categories (gluten-free, dairy-free, vegetarian, etc.)

- "For the ER diagram, you might have some optional (O-|-) relationships"
  [response] We had difficulty locating this feature on Workbench. We changed our diagram using draw.io and included optional relationships.

From feedback #2:

- "In the Customers entity, your customer entity only has 30 characters. I think that might be a little short? I looked this up for my project, and it suggested 35 characters per name (first and last)."
  [response] We took the feedback into consideration and increased our name field to 50 characters.

- "under Dishes you have an attribute order_quantity that's listed as an FK, but I'm not seeing it elsewhere. What is this linked to?"
  [response] This was a typo, and we removed that it is an FK.

- “for your rating int, would there need to be some sort of constraint for the datatype? Whether that be a lower or upper bound, or even if you want a certain number of decimals etc?”
  [response] We decided to implement a constraint for the rating int, to cap it at 5. Ratings should be between 1 and 5. There would be no decimals, since it is an integer.

  From feedback #3:
- “While your project doesn't specifically call out the problem is being solved via a website, it is definitely implied”
  [response] Edited the overview to include that the client wants a website with a database backend.

- “One thing I would try to quantify is the 'total number of unique customers' because that will affect how you will ultimately size your DB, as well as your attribute types potentially.”
  [response] We agree, and because customer phone numbers and IDs are unique, we are able to run table analytics and determine how many unique customers we have.

- “In the instructions its called out as not being necessarily wrong for including your intersection tables in the ER, but that its preferred to use them later on”
  [response] Edited ER diagram by using draw.io instead of MySQL Workbench.

- From feedback #4:
- ERD diagrams often leave out the finer details such as attributes and intersection tables as its main focus is to demonstrate the rough structure and interconnectivity of the database tables.
  [response] We have removed most intersection tables and understand that attributes are usually left out as well. Because we wanted to plan ahead, we

decided to keep the attributes in this time. We left an intersection table because it was included in the rubric.

Other updates:

- We changed the attribute ‘name’ under Dishes to ‘dish_name’ for clarification
- We changed ‘dietary_restriction_id’ to ‘restriction_id’ for brevity