Backwoods Burgers Service Management

Avery Smith, Tyler Vincent

Url to index.html page:
http://classwork.engr.oregonstate.edu:6798

TA and Peer Review Feedback:

TA:

"This is a really good start!

However, there are some errors regarding your use of the incorrect foreign keys in the Reservations table (you should use the primary key CustomerID from the Customers table as the foreign key). You are also missing other necessary Foreign Keys in your some tables. Also, your many to many is not set up properly right now."

TA:

"This is really solid work!

Feedback:

There's a mismatch in your outline vs diagrams regarding the FK "serverID" listed in your ServersTables. It should be employeeID.

You list "Status" for your Reservations table, but that is not in your diagrams or your outline.

Your DDL has tableID as a FK in reservations. This would mean that there's a 1:M relationship between tables and reservations that is not depicted in the diagrams.

Please briefly touch on the normalization process/what your schema is currently in (1NF/2NF/3NF).

Please expand the overview section with more details about your Backwoods Burger's, your design strengths, etc.

Overall great work!

TA:

This is a good start group 139!

Feedback:

You are missing the DML file in your zip. Please be sure to include that moving forward.

You are missing a significant amount of CRUD requirements. It should not be functional for this step of the project, but you should still show the buttons and forms for CRUD. From the project guide:

- 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. It is generally not appropriate to have only a single query that joins all tables and displays them.
- 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 SELECT & INSERT functionalities for all relationships as well as entities. And DELETE & UPDATE for least one M:M relationship. Put another way, you need a method to SELECT, INSERT, UPDATE and DELETE the rows of an intersection table in your project.

Please update your Reservations entity in the outline to reflect the DDL (you are missing table tableID, a discussion about the relationship with tables, and some of your FK null/not null values don't match.

Make sure to include drop downs for FKs in your UI when you add more insert functionality.

Let me know if you have any questions or need assistance with anything.

Thanks!

Feedback

You are missing date-time in your current parties interface.

This line in your DML should be currentID not customerID - INSERT INTO serversTables (customerID, tableID, employeeID)

I took points off since the DDL wasn't hand authored like the rubric said.

Peer Review Feedback:

Steps 1&2:

https://edstem.org/us/courses/70689/discussion/6120608

https://edstem.org/us/courses/70689/discussion/5999817

Step 3:

https://edstem.org/us/courses/70689/discussion/6173825?answer=14352794 https://edstem.org/us/courses/70689/discussion/6173825?answer=14362338

https://edstem.org/us/courses/70689/discussion/6173825?answer=14362350

https://edstem.org/us/courses/70689/discussion/6173825?answer=14362441

Actions based on feedback:

- Step 1 & 2
    - Clarified relationships between tables, in the descriptions. We chose to rework the servers:reservations relationship to be 1:M to help with simplicity and avoid confusion.
    - Reworked names to maintain consistency (camelCase), we had previous inconsistencies with capitalization..
    - Made Servers:Tables Many to Many. Properly facilitating the relationship with an intersection table now, ServersTables.
    - Corrected reservations to reference customerID FK instead of individual attributes from Customers table.
    - Ensured that we are now using employeeID instead of also using serverID on the outline and ERD.
    - Confirmed 3NF for the outline, ERD, and schema.
    - Further adjusted the schema models for discrepancies between the files and the outline.
    - Fixed discrepancies regarding naming, standardizing case as well as using employeeID instead of serverID in the schema, matching the outline.
    - Cemented a 1:M relationship between tables and reservations.
- Step 3
    - *Feedback based changes:*
        - Was initially hosting on heroku for testing for ease and control. Have now hotels it on OSU servers as someone suggested that in peer reviews.
        - Ensured all Not null attributes correlated between outline and Schema
        - Updated Schema Diagram
        - All pages now have insert functionality
        - Deleting employee button now works
        - Added a tableID attribute to reservations and discussed the relationship
        - Seat New Party form was hidden when no data was present. Overhauled entire page so this form no longer exists

        - Accidentally turned current parties page into a 1:M functionality for servers to tables, so changed design to allow multiple servers to be attached to a single table, and for a single server to be attached to multiple tables
    - *Changes we made on our own/upgrades:*
        - Placed Cascading Deletes to prevent data anomalies.
        - Now, current table assignment page(reprsenting M:N serversTables table), has a row for every active table, and allows you to assign as many servers as desired, no duplicates allowed. In addition, a server can be assigned to many tables, maintaining the M:N relationship.
        - Email field for customers is allowed to be nullable and is optional on the insert form.

- Step 4
    - *Feedback based changes*
        - Added date/time in the parties interface.
        - Removed customer entirely from serversTables for now
        - Chose not to use currentID in insert statement into serversTables as this is an autogenerated PK that is generated automatically when a new serversTables transaction is created.
        - Hand authored the DDL

Entities: Tables, Servers, Tabs, Reservations, Customers

Relationships: Servers:Tables(Many-many), Tables:Tabs(one-many), Servers:Reservations(many-many), Customers:Reservations(one-many)

Overview: Our system will help Backwoods Burgers, which has 30 employees and serves 300+ customers a day, more efficiently manage their interconnected communications from reservations, payments, active tables, and ordering. Our implementation will keep all of their data in one centralized Database that will connect the various systems to create an easy-to-use interface for employees. This will speed up ordering, payments, table turnaround, and mitigate reservation conflicts. This design will allow employees or managers to organize employee designations for tables and reservations with ease, which will lead to a more efficient day-to-day business operation. With this implementation, tabs will get much smarter and make servers' lives easier. With digital tracking of tabs and linking to tables, it will be easier to split tabs than traditional notepads or memory. In addition, this will result in fewer accidental charges or incorrect tabs, and in the case there are, they will be much easier to amend.

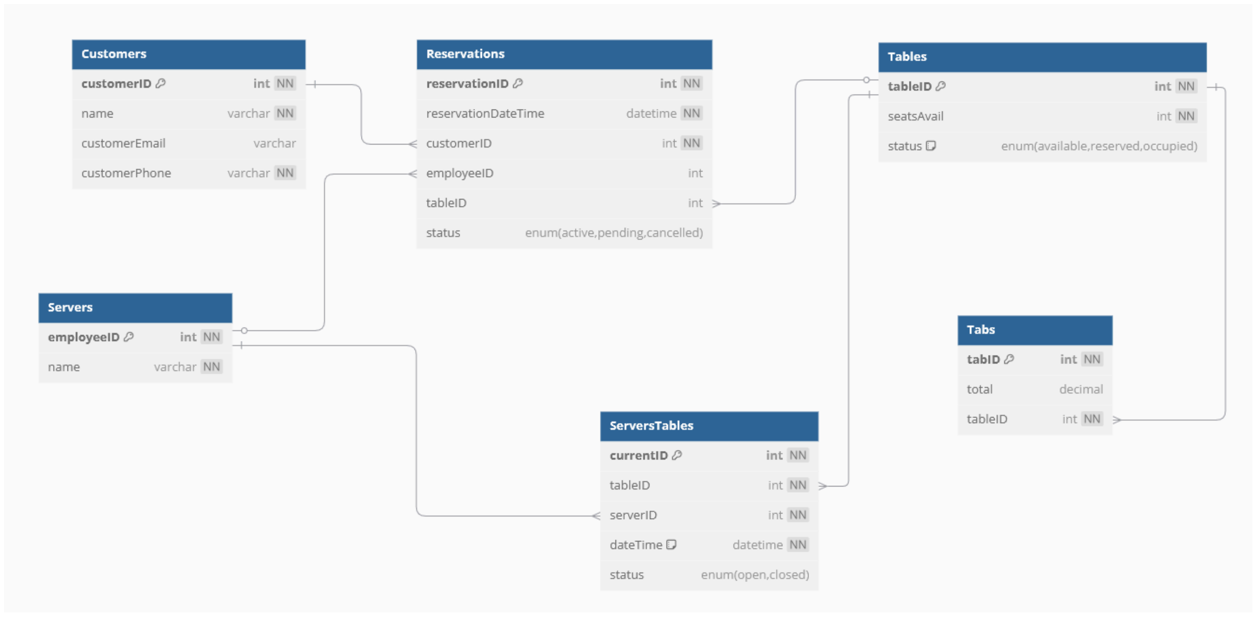
Database Outline:

- Servers
    - employeeID: int, unique, auto_increment, Not Null, PK
    - name: varchar, not null
    - Relationships: M:N servers to tables using employeeID and tableID. 1:M Servers:Reservations Allows one server to handle multiple reservations.
- Tables
    - tableID: int, unique, auto_increment, not Null, PK
    - seatsAvail: int, Not Null
    - status: enum('available', 'reserved', 'occupied'), Default 'available'
    - Relationships: 1:M tables to tabs to handle single or separate checks. M:N Tables:Servers, in the case of a large party multiple servers can handle subgroups of the main party. 1:M relationship with reservations, many reservations can be for a single table past, present, or future.
- Reservations

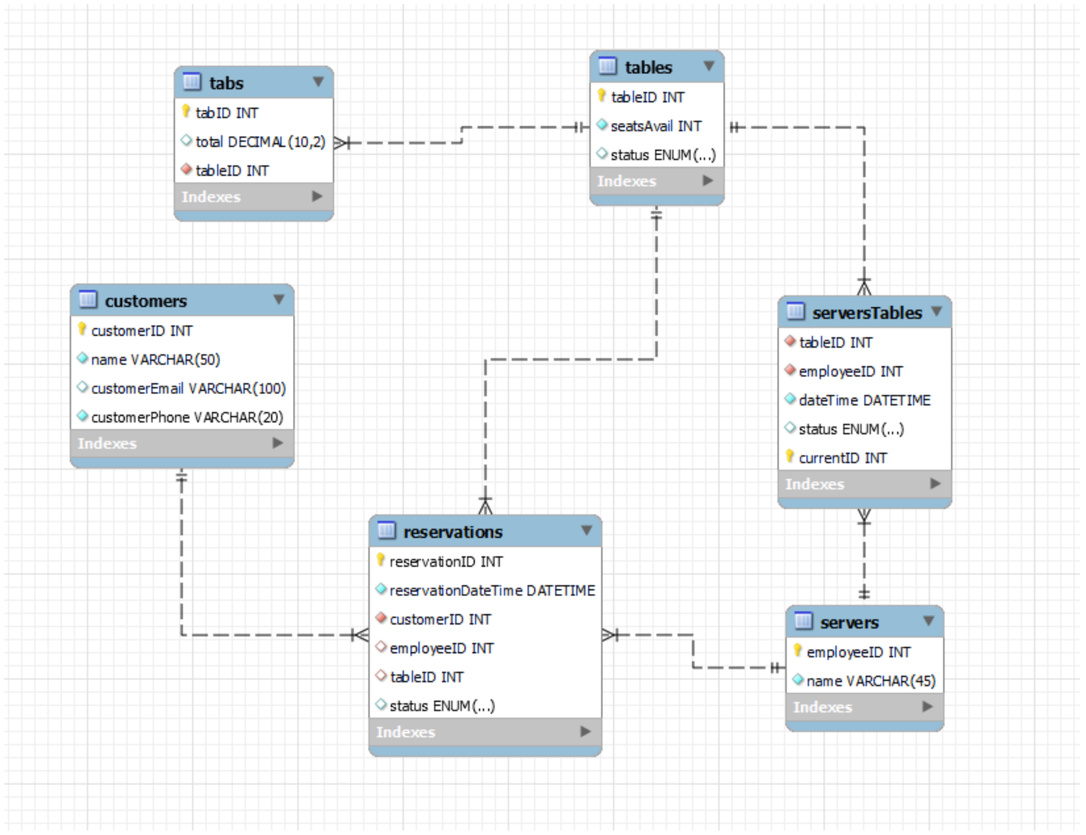
- reservationID: int, unique, auto_increment, not Null, PK
    - reservationDateTime: datetime, not null
    - customerID: foreign key, Not Null
    - employeeID: int, foreign key
    - tableID: foreign key
    - status: enum('active', 'pending', 'cancelled')
    - Relationships: 1:M Customers to Reservations, allows customers to have multiple reservations past or present. 1:M Each reservation is assigned one server, but a server can handle multiple reservations. M:1 relationship with tables, many reservations can be made for the same table.

- Tabs
    - tabID: int, unique, auto_increment, not Null, PK
    - total: decimal
    - tableID: int, not Null, FK
    - Relationships: 1:M tables:tabs to handle single or separate checks.
- Customers
    - customerID: int, unique, auto_increment, not Null, PK
    - name: varchar, Not Null
    - customerEmail: varchar
    - customerPhone: varchar, Not Null
    - Relationships: 1:M Customers to Reservations, allows customers to have multiple reservations past or present.
- ServersTables
    - currentID: primary key, auto_increment, Not Null
    - tableID: int, Not Null
    - serverID: int, Not Null
    - dateTime: current_timestamp, Not Null
    - status: enum("open", "closed")
    - Relationships: Intersection table between servers and tables to manage seating parties and assigning servers to tables. It facilities a many-to-many relationship between the two entities and carries a time stamp from when the party is first seated. The status is set to open when the party is seated and closed when the party leaves.

Entity-Relationship-Diagram



3NF Schema Diagram



Mock Data Table (Data has changed slightly while testing site functions, but all attributes and most data are the same):

Servers Table

employeeID	name
1	John Porker
2	Michael Chen
3	Emily Rodriguez
4	David Kim

Tables Table

tableID	seatsAvail	status
1	4	avail

2	2	avail
3	6	taken
4	8	avail

Reservations Table

reservationID	reservationDateTime	customerID	employeeID	tableID	status
1	2025-02-07 18:00:00	1	1	1	active
2	2025-02-08 19:30:00	2	2	2	pending
3	2025-02-09 20:00:00	3	3	3	cancelled
4	2025-02-13 21:00:00	4	4	4	active

Tabs Table

tabID	total	tableID
1	45.99	1
2	127.50	2
3	89.75	3
4	156.80	1

Customers Table

customerID	name	customerEmail	customerPhone
1	Emily Davis	emily.davis@email.com	555-123-4567

2	Benjamin Lee	benjamin.lee@email.com	555-234-5678
3	Chloe Scott	chloe.scott@email.com	555-345-6789
4	Noah Harris	noah.harris@email.com	555-456-7890

ServerTables Table (M:N Relationship)

employeeID	tableID	status	dateTime	currentID
1	1	open	current_timestamp	1
2	2	open	current_timestamp	2
3	3	open	current_timestamp	3
4	4	open	current_timestamp	4