Project Step 5 Draft

Team Members:
Ryunosuke Yano
Kyle Donovan
CS 340 Group 52

Link: http://flip1.engr.oregonstate.edu:51410/

**a) Project Outline and Database Outline - Updated Version:**

Project Outline

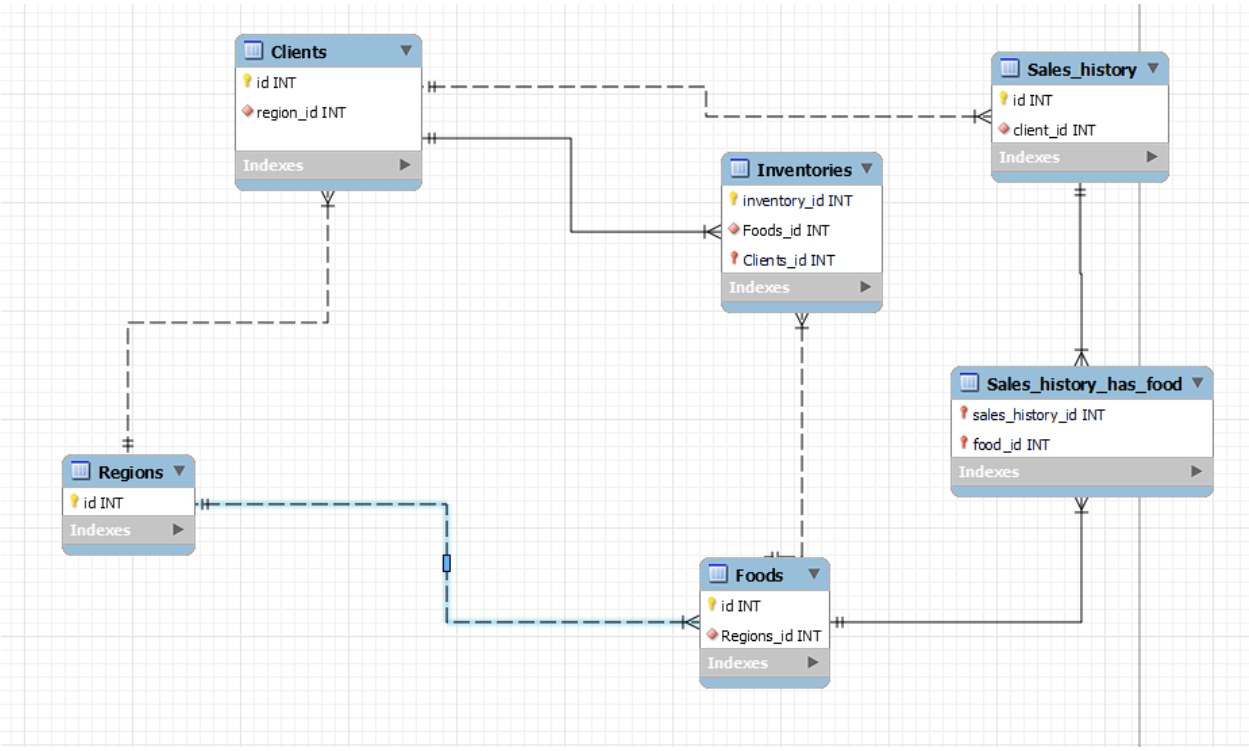
YanoDanoCo (maintained by Ryunosuke Yano and Kyle Donovan) is a growing food distribution platform where our clients track wholesale orders of frozen food for their restaurants. Our client base includes over 100 restaurants spending over 17 million through YanoDanoCo annually and these restaurants need a way to maintain their inventory in a streamlined manner as these restaurants order on a biweekly basis to fulfill the needs of their customers. In addition, although most foods can be purchased in all 7 regions, some are geographically restricted due to import laws or financial restrictions due to transport costs. Our software meets the needs of our clients by grouping our client base by regions in order to better assist our clients' purchases with regards to regional restrictions. Our database will provide the backend for our web application to track inventory and transactions for YanoDanoCo's many clients. For the database, we have 6 entities, regions, clients, inventories, sales history, foods and an intersection table between sales history and foods.

- Regions
    - Purpose: to keep track of each restaurants in a general geographical area
    - Attributes:
        - region_id (int, primary key, auto_increment, unique): Unique identifier for each region
        - region_name (VARCHAR(255), not NULL): name of the region
    - Relationships:
        - 1:M relationship between region and client with region_id as FK inside client
        - 1:M relationship with foods as region_id as FK inside foods
- Inventories
    - Purpose: to keep track of available item stock levels
    - Attributes:
        - inventory_id (int, auto_increment, unique not NULL, PK)
        - Client_id (int, not Null): id of client
        - item_count (int, not NULL): inventory of item

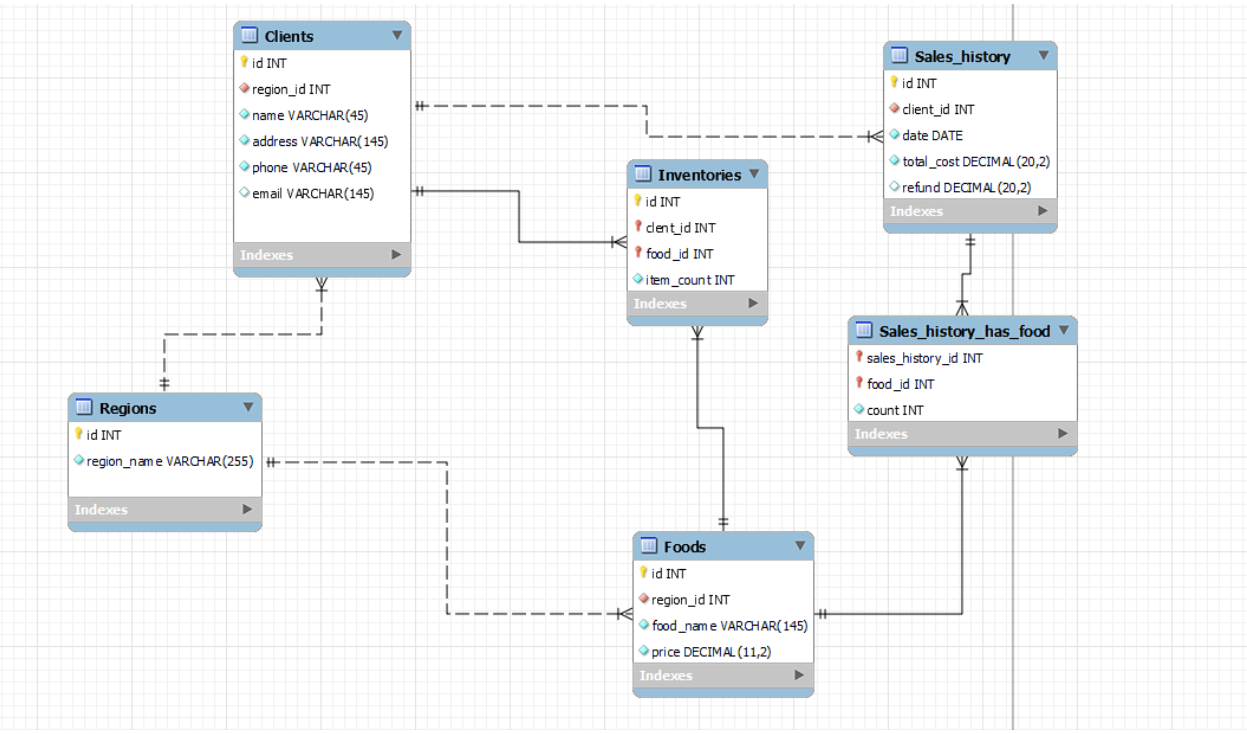
        - Units (varchar(45), not null): units for item_count
        - food_id (int, not NULL, FK): id of food
    - Relationships:
        - 1:M with client with inventory_id as FK inside inventories
        - M:1 with food with food_id as FK inside inventory
- Clients
    - Purpose: to provide details about the client
    - Attributes:
        - client_id (INT, primary key, unique, auto_increment, PK): unique identifier
        - name (VARCHAR(45), not NULL): name of the client
        - address (VARCHAR(145), not NULL): address of the client
        - phone (VARCHAR(45), not NULL): phone number of the client
        - email (VARCHAR(145)): email address for the client
        - region_id (int, foreign key, not NULL, FK): unique identifier of the region
    - Relationships:
        - 1:M with sales_history as FK inside sales_history
        - 1:M with inventories as FK inside inventories
        - M:1 with region as FK inside client.
- Sales_history
    - Purpose: to track previous orders from clients
    - Attributes:
        - sales_history_id (int, not NULL, auto_increment, unique, PK): unique identifier for each sale
        - date (date, not NULL): date of purchase
        - client_id (int, not NULL, FK): id of client
        - total_cost(decimal(20,2), not NULL): total cost of transaction
        - refund (decimal(20, 2)): amount of refund, can be NULL in the case of no refund
    - Relationships:
        - M:1 with clients as FK inside sales history
        - M:N with foods implemented with sales_history_has_food as intersection table. Sales_history_id and food_id as foreign keys inside the intersection table
- Sales_history_has_food
    - Purpose: intersection table between sales history and Foods to keep track of each individual food sales and the amount involved in transaction
    - Attributes:
        - Sales_history_id (int, FK, not Null): unique identifier for each sales history
        - food_id(int, FK, not NULL): unique identifier for each food
        - count(int, not NULL): amount of food transacted in sale
    - Relationships:
        - 1:M with sales history with sales_history_id as FK
        - 1:M with Foods with food_id as FK

- Food (referenced by inventory)
    - Purpose: to create category table for each food sold by YanoDanoCo
    - Attributes:
        - food_id(int, not NULL, auto_increment, unique, PK): unique identifier for each food type
        - food_name(VARCHAR(145), not NULL): name of each food type.
        - price (decimal(11,2), not NULL): price of each food in USD
        - region_id (int, FK): region id from where food originated
    - Relationships:
        - 1:M with inventory with food_id as FK inside inventory.
        - 1:M with sales_history with food_id as FK inside sales_history
        - M:1 with regions with regions_id as FK inside foods

**Entity-Relationship Diagram:**




**Schema:**




**Example Data**

**Regions**

| region_id | region_name |
|---|---|
| 1 | Central America |
| 2 | United States |
| 3 | Western Europe |

**Clients**

| client_id | region_id | name | address | phone | email |
|---|---|---|---|---|---|
| 1 | 2 | Ryu's Kitchen | 10000 Jollyville Rd Austin, TX 78759 | 804-493-2999 | ryukitchen@gmail.com |
| 2 | 2 | Kyle's Sandwich Shop | 3939 Parmer Lane Bend, Oregon 19394 | 757-321-4342 | KyleSandwich@gmail.com |
| 3 | 1 | #1 Tacos | 4939 45th Street, | 949-999-392 | Num1Tacos@gmail.co |

| | | | | | |
|---|---|---|---|---|---|
| | | | Mexico city, Mexico | 9 | m |
| 4 | 3 | Fish n’chips | 3991 6th street, Britain, England | 399-329-2945 | fishnchips@gmail.com |

**Inventories**

| inventory_id | client_id | food_id | item_count | units |
|---|---|---|---|---|
| 1 | 1 | 1 | 20 | lbs |
| 2 | 1 | 2 | 356 | lbs |
| 3 | 2 | 3 | 22 | lbs |
| 4 | 2 | 1 | 11 | lbs |

**Foods**

| food_id | region_id | food_name | price |
|---|---|---|---|
| 1 | 2 | corn | 1.29 |
| 2 | 2 | ribeye | 19.99 |
| 3 | 1 | beans | 1.09 |

**Sales History**

| sales_history_id | client_id | date | total_cost | refund |
|---|---|---|---|---|
| 1 | 1 | 2022-12-20 | 2044.64 | null |
| 2 | 3 | 2022-12-19 | 266666.66 | null |
| 3 | 3 | 2022-12-17 | 23256.56 | 23256.56 |

**Sales_history_has_food**

| sales_history_id | food_id | count |
|---|---|---|
| 1 | 2 | 356 |

| | | |
|---|---|---|
| 2 | 1 | 22 |
| 3 | 2 | 45 |

**b) Fixes based on Feedback from Step 4:**
**Actions:**
**-**Implementing the CRUD for 5 out of 6 tables
-Resolved internal server error issue for Clients, Inventories, and Sales_Histories
Upgrades to draft version:
None this time since most of our issues wee from implementation

**b) Fixes based on Feedback from Step 3:**

Actions based on the feedback:
-Made one of the Foreign keys optional
-Fixed the DDL script to behave properly
-Fixed DML script to add Join statement
-Inserted a units attribute to specify which units are used for count
-Added a searchable select field into the sales history page for a dynamically added list

Upgrades to the Draft version:
-Changed foreign key constraint for food from not null to nullable. Some types of food might be from multiple regions so it will be possible to source from other areas.
-added units entity in outline
-fixed inconsistency between DML script and example data

Changes not implemented:
-One reviewer asked why client_id is in inventory. Our logic is that single foods can be in multiple clients' inventories and a single client can have multiple food types. Therefore, inventories act as an intersection table while having use as an inventory by keeping track of the item_count and units.
-One reviewer indicated that add is not working -> This change will be incorporated at a later time (functional add is not yet necessitated per the program requirements).

**Fixes based on Feedback from Step 2:**

Actions based on the feedback:
-Added cascade operation to on delete and on update in .sql file
-fixed issue in .sql query that was causing issues
-Removed quotes for integer type in INSERT statements
-removed unnecessary info in comment section from sql file

Upgrades to the Draft version:

-Redid outline portion
-Corrected keys from PK to FK in intersection table (outline)