Gabe Golden, Chad Tolleson

CS586 - Spring 2019 - Graduate Project - Database Implementation - Part 3

<u>Your ER diagram, showing any changes you made during the implementation Process</u>

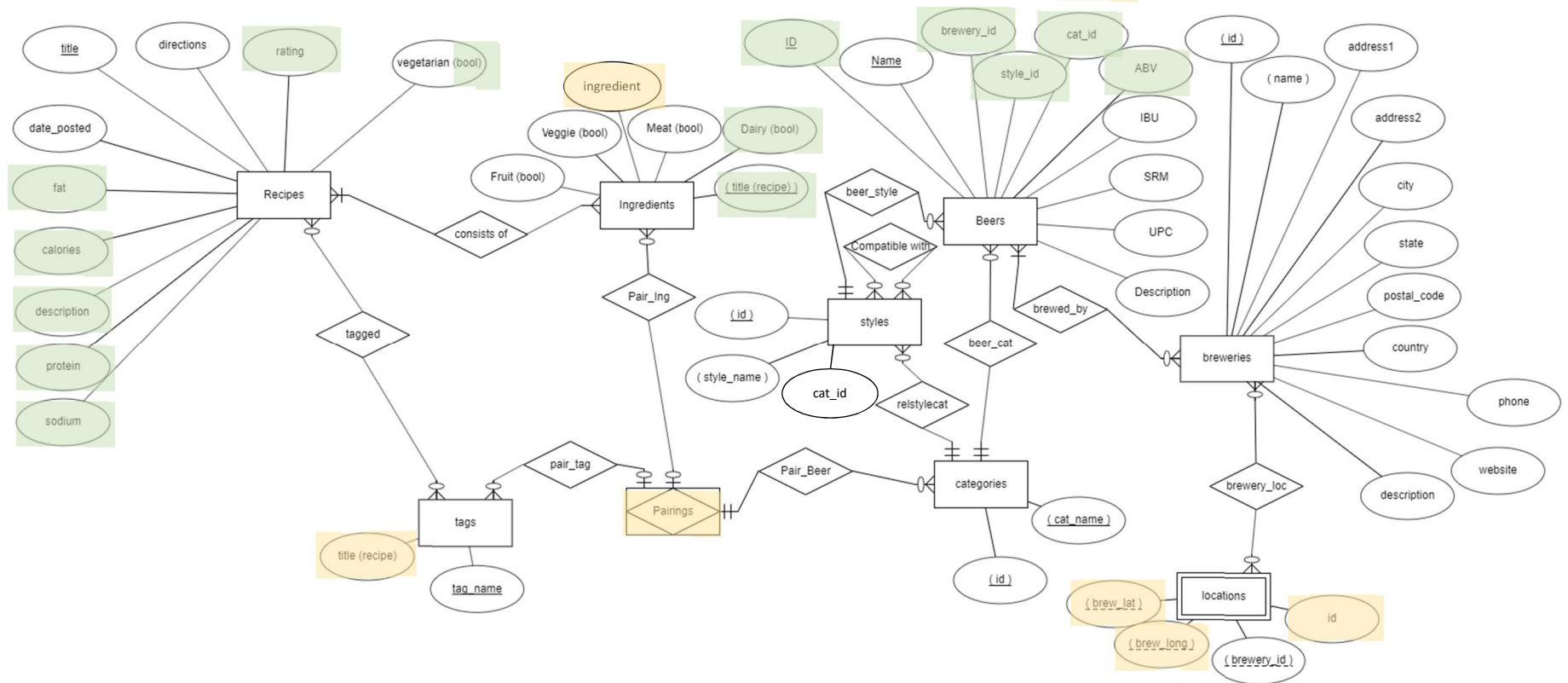ERD Submitted for Part 2, changes highlighted in canary, removals highlighted in rose:

Current ERD, additions highlighted in sage, removals highlighted in rose, and modifications highlighted in yellow:

The CREATE TABLE statements for your database

```
CREATE TABLE beers (
        id int PRIMARY KEY,
        brewery_id int REFERENCES breweries(id),
        name text,
        cat_id int REFERENCES categories(id),
        style_id int REFERENCES styles(id),
        abv double DEFAULT 0.0,
        ibu double DEFAULT 0.0,
        srm double DEFAULT 0.0,
        upc double DEFAULT 0.0,
        filepath text,
        description text,
        add_user text,
        last_mod text
);

CREATE TABLE breweries (
        id int NOT NULL,
        name text NOT NULL,
        address1 text,
        address2 text,
        city text,
        state text,
        code text,
        country text,
        phone text,
        website text,
        description text,
        PRIMARY KEY(id, name)
);

CREATE TABLE categories (
        id int NOT NULL,
        cat_name text NOT NULL,
        PRIMARY KEY(id, cat_name)
);

CREATE TABLE ingredients (
        title text REFERENCES recipes(title),
        ingredient text,
        meat bool,
        veggie bool,
        fruit bool,
        dairy bool,
        vegetarian bool
```

```
);

CREATE TABLE locations (
        id int,
        brewery_id int NOT NULL,
        brew_lat text NOT NULL,
        brew_long text NOT NULL,
        PRIMARY KEY(brewery_id,brew_lat,brew_long)
);

CREATE TABLE pairings (
        cat_id int REFERENCES categories(id),
        pair_type text NOT NULL,
        pair_link text
);

CREATE TABLE recipes (
        title text PRIMARY KEY,
        directions text,
        fat text,
        dateload text,
        calories text,
        description text,
        protein text,
        rating text,
        sodium text,
        vegetarian bool
);

CREATE TABLE styles (
        id int NOT NULL,
        cat_id int REFERENCES categories(id),
        style_name text NOT NULL,
        PRIMARY KEY(id,style_name)
);

CREATE TABLE tags (
        title text,
        tag_name text
);
```

A brief description of how you populated the database

We initially designed the tables based on our original expectations, and refined in the process of creating the ERD.

The data we used for this database came from two sources, the Open Beer Database [1] for most beer related information, and most recipe data came from a Kaggle repository of Epicurious recipes [2]. All of the data required some cleaning and manipulation before being suitable for importing into the database. Some data little more than copying and pasting into a text file, but many of the tables contained unexpected characters such as new line and register return that required a not insignificant amount of time to cleanse. Furthermore, much of the data did not fit into the constraints we planned and anticipated. These situations required that we either remove the non-conforming data records or loosen the constraints we initially set on the relations between tables.

Once we moved beyond those hurtles, we loaded data into the database using one of these three methods:

1) Smaller tables were loaded using long INSERT INTO statements like this:

```
INSERT INTO categories(id,cat_name)
VALUES
('1','British Ale'),
('2','Irish Ale'),
('3','North American Ale'),
('4','German Ale'),
('5','Belgian and French Ale'),
('6','International Ale'),
('7','German Lager'),
('8','North American Lager'),
('9','Other Lager'),
('10','International Lager'),
('11','Other Style');
```

2) Larger tables were formatted and saved into CSV files which were then loaded using statements like this (breweries.csv included separately for reference):

```
\copy breweries from breweries.csv with csv;
```

3) Other large files were fortunately available in .sql files (albeit for other database besides PostgreSQL) and were easily converted into statements we were able to execute from within the database such as this (beers2.sql included separately for reference):

```
\i beers2.sql
```

On a similar topic, there are several Boolean fields in the ingredients and recipe tables that were populated by us using update queries to set the values to true/false. Some examples of these fields are indicators as to whether an ingredient is a vegetable, meat, dairy, etc. And if a recipe is vegetarian. Additionally, the pairings table which links the recipe and beer data together was manually compiled using data found online. [3]

For each of your 20 questions, the question in English, its translation to SQL and the (full) answer to the query. (If you needed to change any of your original questions, also list the originals and why you needed to change or replace them.) & A listing of 5 rows from each of your tables

1) How many recipes are in the database?

a. Query
SELECT COUNT(*) FROM recipes;

b. First 5 rows

```
dbgroup1=> SELECT COUNT(*) FROM recipes;
 count
-------
 15730
(1 row)
```

2) How many different types of ingredients are in the database?
   a. Query
   SELECT COUNT(*)
   FROM(
   SELECT ingredient
   FROM ingredients
   GROUP BY ingredient
) AS sub;

   b. First 5 rows

```
dbgroup1=> SELECT COUNT(*)
dbgroup1-> FROM(
dbgroup1(> SELECT ingredient
dbgroup1(> FROM ingredients
dbgroup1(> GROUP BY ingredient
dbgroup1(> ) AS sub;
 count
-------
 83462
(1 row)
```

3) How many recipes use onions as an ingredient? How many recipes use apples as an ingredient?
   a. Query
   SELECT COUNT(*)
   FROM (
   SELECT title
   FROM ingredients
   WHERE ingredient LIKE '%onion%'
   OR ingredient LIKE '%apple%'
   GROUP BY title
) AS sub;

   b. First 5 rows

```
dbgroup1=> SELECT COUNT(*)
dbgroup1-> FROM (
dbgroup1(> SELECT title
dbgroup1(> FROM ingredients
dbgroup1(> WHERE ingredient LIKE '%onion%'
dbgroup1(> OR ingredient LIKE '%apple%'
dbgroup1(> GROUP BY title
dbgroup1(> ) AS sub;
 count
-------
  6084
(1 row)
```

4) How many recipes use both apples and onions as ingredients?
   a. Query
   SELECT COUNT(*)
   FROM (
   SELECT title
   FROM ingredients
   WHERE ingredient LIKE '%onion%'
   INTERSECT
   SELECT title
   FROM ingredients
   WHERE ingredient LIKE '%apple%'
   GROUP BY title
) AS sub;

   b. First 5 rows
```
dbgroup1=> SELECT COUNT(*)
dbgroup1-> FROM (
dbgroup1(> SELECT title
dbgroup1(> FROM ingredients
dbgroup1(> WHERE ingredient LIKE '%onion%'
dbgroup1(> INTERSECT
dbgroup1(> SELECT title
dbgroup1(> FROM ingredients
dbgroup1(> WHERE ingredient LIKE '%apple%'
dbgroup1(> GROUP BY title
dbgroup1(> ) AS sub;
 count
-------
   320
(1 row)
```

5) Which beers pair with the recipes using beer as an ingredient and the most beer pairings?
   a. Query
   CREATE VIEW vw_beerrecipepairings AS (

```
SELECT i.title, COUNT(b.name) AS beer_count
FROM
ingredients i, pairings p, beers b,
(SELECT title
FROM ingredients
WHERE ingredient like '%beer%'
GROUP BY title) AS sub1
WHERE i.title = sub1.title
AND p.pair_type='ingredient'
AND i.ingredient=p.pair_link
AND p.cat_id=b.cat_id
GROUP BY (i.title, b.name));

SELECT bw.name as brewery, b.name as beer
FROM beers b, pairings p, ingredients i,
breweries bw,(SELECT title
FROM vw_beerrecipepairings
WHERE beer_count=(
SELECT MAX(beer_count)
FROM vw_beerrecipepairings)) as sub
WHERE sub.title=i.title
AND i.ingredient=p.pair_link
AND p.pair_type='ingredient'
AND p.cat_id=b.cat_id
AND b.brewery_id=bw.id
GROUP BY (bw.name, b.name);
```

    b. First 5 rows

6) Show recipes that are Italian but do not include tomatoes.
    a. Query

```
(SELECT DISTINCT r.title
FROM recipes r, tags t, ingredients i
WHERE r.title=t.title
AND r.title=i.title
AND ( LOWER(t.tag_name) LIKE LOWER('%Ital%')
OR LOWER(i.ingredient) LIKE LOWER('%Ital%') ) )
EXCEPT
(SELECT r.title
FROM recipes r, tags t, ingredients i
```

```
WHERE r.title=t.title
AND r.title=i.title
AND LOWER(t.tag_name) LIKE LOWER('%tomat%')
AND LOWER(i.ingredient) LIKE LOWER('%tomat%') );
```

b. First 5 rows

7) Show the recipes with >= 1000mg Sodium, scotch as an ingredient, and does not have dairy as an ingredient.

a. Query

```
SELECT DISTINCT r.title, i.ingredient
FROM recipes r, ingredients i
WHERE r.title=i.title
AND LOWER(i.ingredient) LIKE LOWER('%scotch%')
AND LOWER(i.ingredient) NOT LIKE LOWER('%butterscotch%')
AND LOWER(i.ingredient) NOT LIKE LOWER('%scotch bonnet%')
AND r.sodium >= 1000
EXCEPT
SELECT DISTINCT r.title, i.ingredient
FROM recipes r, ingredients i
WHERE r.title=i.title
AND i.dairy=FALSE;
```

b. First 5 rows

8) Show all paleo recipes with < 400 calories.

a. Query

```
SELECT DISTINCT r.title
FROM recipes r, tags t
WHERE r.title=t.title
AND LOWER(t.tag_name) LIKE LOWER('%paleo%')
AND r.calories < 400;
```

b. First 5 rows

```
                                              title
-------------------------------------------------------------
Butternut Squash with Pumpkin-Seed Pesto
Coleslaw with Bay Shrimp and Tarragon
Carrot and Squash Ribbons
Deviled Green Eggs With Roasted Red Pepper and Capers
Sautéed Zucchini
```

9) Show all recipes that require prep.
   a. Query
      SELECT DISTINCT r.title
      FROM recipes r
      WHERE LOWER(r.directions) LIKE LOWER('%prep%');

   b. First 5 rows

```
                                              title
-------------------------------------------------------------
Spiced Nectarine Cake
Thai-Spiced Pork Tenderloin with Orange Curry Sauce
Strawberries with Cassis
Grilled Pork Chops with Cherry Relish
Praline Cheesecake with Hazelnut Crust
```

10) How many recipes are vegetarian and have 5-star ratings?
    a. Query
       SELECT COUNT(*)
       FROM recipes r
       WHERE vegetarian is TRUE
       AND rating >= 5;

    b. First 5 rows

```
dbgroup1=> select title, vegetarian, rating from recipes where vegetarian=TRUE and rating >=5;
dbgroup1=>
dbgroup1=>
dbgroup1=> SELECT COUNT(*)
FROM recipes r
WHERE vegetarian is TRUE
AND rating >= 5;
 count
-------
   658
(1 row)
```

11) Show all recipes with the word 'simple' in the description.
    a. Query
       SELECT DISTINCT r.title
       FROM recipes r
       WHERE LOWER(r.description) LIKE LOWER('%simple%');

b. First 5 rows

12) Show all high (> 700) calorie recipes with 'holiday' in the description.

a. Query

SELECT DISTINCT r.title

FROM recipes r

WHERE r.calories > 700

AND LOWER(r.description) LIKE LOWER('%holiday%');

b. First 5 rows

13) Show all no-cook desserts

a. Query

SELECT recipes.title

FROM recipes

WHERE recipes.title IN (SELECT tags.title FROM public.tags

WHERE tags.tag_name = 'No-Cook'

GROUP BY tags.title)

AND recipes.title IN (SELECT tags.title FROM public.tags

WHERE tags.tag_name = 'Dessert'

GROUP BY tags.title);

b. First 5 rows

14) Show all grill-related recipes rated >=4.

a. Query

SELECT recipes.title

FROM recipes

```
WHERE recipes.rating >= 4
AND (recipes.title IN (SELECT tags.title FROM public.tags
WHERE tags.tag_name LIKE '%Grill%'
GROUP BY tags.title)
OR recipes.title LIKE '%Grill%');
```

b. First 5 rows

```
                                   title
---------------------------------------------------------------------------
Grilled Shrimp, Corn and Black Bean Tostada Salad
Brined and Barbecued Turkey
Shrimp and Mango Skewers with Guava-Lime Glaze
Buffalo Grilled Shrimp with Blue Cheese Dip and Celery
Mango Salad with Grilled Shrimp
```

15) Which beer pairs well with the beer I am currently drinking? Take a beer type, find all the recipes that pair with that beer, select a recipe, and then find all of the beers that pair with that recipe.

a. Query
Note, this ran with Deschutes Mirror Pond Pale Ale, beer id 3587

```
CREATE VIEW vw_mirrorp_recipes AS
SELECT DISTINCT r.title
FROM recipes r, ingredients i, tags t,
(SELECT p.pair_link, b.name
FROM pairings p, beers b
WHERE b.id=3587
AND p.cat_id=b.cat_id) AS pb
WHERE r.title=i.title AND r.title=t.title
AND LOWER(pb.pair_link) = LOWER(i.ingredient)
UNION
SELECT DISTINCT r.title
FROM recipes r, ingredients i, tags t,
(SELECT p.pair_link, b.name
FROM pairings p, beers b
WHERE b.id=3587
AND p.cat_id=b.cat_id) AS pb
WHERE r.title=i.title AND r.title=t.title
AND LOWER(pb.pair_link) = LOWER(t.tag_name);

SELECT DISTINCT bw2.name AS Brewery, b2.name AS beer
FROM beers b2, breweries bw2, pairings p2, ingredients i2
WHERE b2.brewery_id=bw2.id
AND b2.cat_id=p2.cat_id
AND p2.pair_type='ingredient'
AND LOWER(p2.pair_link) = LOWER(i2.ingredient)
```

```
AND i2.title IN (select * FROM vw_mirrorp_recipes)
UNION
SELECT DISTINCT bw2.name AS Brewery, b2.name AS beer
FROM beers b2, breweries bw2, pairings p2, tags t2
WHERE b2.brewery_id=bw2.id
AND b2.cat_id=p2.cat_id
AND p2.pair_type='tag'
AND LOWER(p2.pair_link) = LOWER(t2.tag_name)
AND t2.title IN (select * FROM vw_mirrorp_recipes);
```

b. First 5 rows

```
                      brewery                          |                          beer
-------------------------------------------------------+------------------------------------------
21st Amendment Brewery Cafe                            | 21A IPA
21st Amendment Brewery Cafe                            | 563 Stout
21st Amendment Brewery Cafe                            | Amendment Pale Ale
21st Amendment Brewery Cafe                            | Bitter American
21st Amendment Brewery Cafe                            | Double Trouble IPA
```

16) Which beers pair with the most recipes?

a. Query
```
SELECT cats.cat_id, categories.cat_name, cats.cnt
FROM categories,
(SELECT cat_id, COUNT(*) AS cnt
FROM pairings
GROUP BY cat_id) AS cats
WHERE categories.id = cats.cat_id
ORDER BY cnt DESC;
```

b. First 5 rows

```
cat_id |         cat_name         |  cnt
-------+--------------------------+-------
     3 | North American Ale       | 18318
     5 | Belgian and French Ale   | 15590
     1 | British Ale              | 14783
     7 | German Lager             | 10598
     4 | German Ale               |  6332
```

17) Which tags do not pair with the beer with the highest pairing count found in number 16, above.
(*We updated this question from the original because we ended up not having comprehensive
cuisine data in our tags.*)

a. Query
```
SELECT DISTINCT(tags.tag_name)
FROM tags
WHERE TRIM(tags.tag_name) NOT IN
(SELECT DISTINCT TRIM(pairings.pair_link)
FROM pairings
WHERE pairings.pair_type = 'tag'
```

AND pairings.cat_id = 3);

b. First 5 rows

```
        tag_name
-------------------------
Fontina
Leek
Lychee
Pomegranate Juice
Mixer
```

18) Which beers pair well with Thai food but not with Japanese food? (We updated this from the original question which asked about Thai and Chinese. That combination produced no results.)
    a. Query
    SELECT DISTINCT bw.name AS brewery, b.name AS beer
    FROM beers b, breweries bw, pairings p
    WHERE b.brewery_id=bw.id
    AND b.cat_id=p.cat_id
    AND LOWER(p.pair_link) LIKE LOWER('%thai%')
    EXCEPT
    SELECT DISTINCT bw.name AS brewery, b.name AS beer
    FROM beers b, breweries bw, pairings p
    WHERE b.brewery_id=bw.id
    AND b.cat_id=p.cat_id
    AND LOWER(p.pair_link) LIKE LOWER('%cjapanese%');

    b. First 5 rows

```
              brewery                    |                      beer
-----------------------------------------+------------------------------------------------
21st Amendment Brewery Cafe              | General Pippo's Porter
(512) Brewing Company                    | (512) Pecan Porter
(512) Brewing Company                    | (512) Whiskey Barrel Aged Double Pecan Porter
Alameda Brewhouse                        | Irvington Porter
Alaskan Brewing                          | Alaskan Smoked Porter
```

19) Show all beer that would pair well with a meal containing fish. [4]
    a. Query
    SELECT DISTINCT bw.name AS brewery, b.name AS beer
    FROM breweries bw, beers b, pairings p
    WHERE b.brewery_id=bw.id
    AND b.cat_id=p.cat_id
    AND LOWER(p.pair_link) IN ('shad', 'sole', 'anchovy', 'cod', 'arrowtooth eel', 'carps', 'atka mackerel', 'bonito', 'eel', 'herring', 'salmon', 'trout', 'jack', 'fish', 'barramundi', 'basa fish', 'mackerel', 'bluegill', 'bluefish', 'duck', 'brook trout', 'butterfish', 'halibut', 'sheephead', 'capelin', 'carp', 'catfish', 'chinook salmon', 'chum salmon', 'cobia', 'coho salmon', 'coley', 'crappie', 'crawfish', 'dory', 'discus', 'drum', 'dino', 'flounder', 'flathead', 'flatfish', 'flying fish', 'giant gourami', 'gilt-head bream', 'dorado', 'groundfish', 'grouper', 'gar', 'haddock',

'hake', 'harvestfish', 'hilsa', 'hoki', 'shark', 'basa', 'kapenta', 'kingklip', 'largemouth bass', 'maori cod', 'mahi-mahi', 'marlin', 'milkfish', 'monkfish', 'mullet', 'mullus surmuletus', 'pike', 'snakehead', 'roughy', 'oscar', 'saury', 'panfish', 'pangasius', 'toothfish', 'pelagic cod', 'perch', 'pollock', 'pomfret', 'pilchard', 'pufferfish', 'paddlefish', 'plaice', 'quoy fish', 'rainbow trout', 'redfish', 'snapper', 'sturgeon', 'sardine', 'scrod', 'sea bass', 'seer fish', 'shrimpfish', 'skipjack tuna', 'rainbow sardine', 'snakeskin gourami', 'snook', 'snoek', 'surf sardine', 'swordfish', 'skate', 'sunfish', 'smallmouth bass', 'spoonbill', 'thresher shark', 'tilapia', 'tilefish', 'tuna', 'turbot', 'yellowfin tuna', 'zander');

b. First 5 rows

```
                    brewery                 |                      beer
--------------------------------------------+--------------------------------------
 21st Amendment Brewery Cafe                | 21A IPA
 21st Amendment Brewery Cafe                | 563 Stout
 21st Amendment Brewery Cafe                | Amendment Pale Ale
 21st Amendment Brewery Cafe                | Bitter American
 21st Amendment Brewery Cafe                | Double Trouble IPA
```

20) Show recipes that pair well with IPAs.
   a. Query
   SELECT DISTINCT(ingredients.title)
   FROM ingredients
   WHERE TRIM(ingredients.ingredient) IN (SELECT TRIM(pairings.pair_link)
   FROM pairings,
   (SELECT DISTINCT(cat_id)
   FROM public.styles
   WHERE styles.style_name LIKE '%India Pale Ale%') AS IPAs
   WHERE pairings.cat_id = IPAs.cat_id AND pairings.pair_type = 'ingredient')
   UNION
   SELECT DISTINCT(tags.title)
   FROM tags
   WHERE TRIM(tags.tag_name) IN (SELECT TRIM(pairings.pair_link)
   FROM pairings,
   (SELECT DISTINCT(cat_id)
   FROM public.styles
   WHERE styles.style_name LIKE '%India Pale Ale%') AS IPAs
   WHERE pairings.cat_id = IPAs.cat_id AND pairings.pair_type = 'tag');

   b. First 5 rows

```
                                title
------------------------------------------------------------------
 Light Vegetable Broth
 Sunday Frittata with Frizzled Leeks
 Creamy Black Bean Spread
 Sea Bass and Tomato Ceviche
 Ayran
```

References

[1] Open Beer Database, "Open Beer Database," unknown, [Online]. Available:
https://openbeerdb.com/. [Accessed 12 May 2019].

[2] H. Darwood, "Kaggle," Kaggle Inc., [Online]. Available:
https://www.kaggle.com/hugodarwood/epirecipes. [Accessed 12 May 2019].

[3] CraftBeer.com Contributors, "CraftBeer.com," CraftBeer.com and Brewers Association, [Online].
Available: https://www.craftbeer.com/educational-resources/beer-food-chart. [Accessed 20 May
2019].

[4] Wikibin, contriubtors, "Wikibin," Wikibin, [Online]. Available: http://wikibin.org/articles/list-of-
edible-fish.html. [Accessed 9 June 2019].