Jeffrey Schachtsick
Saul Gonzalez
CS340 - Introduction to Databases
Final Project

Outline

Homebrew database project. The database stores the quantity of ingredients a homebrewer has on hand. The homebrewer uses the ingredient on hand to create new beers. He can add new ingredients to the database as needed. He tracks all the ingredients that make up each beer by creating unique recipes for each beer. A recipe is made up of all the ingredients that make up the beer. The homebrewer can also modify an existing recipe to enhance the flavor by adding more ingredients to the recipe.

Keeping track of the ingredients each beer is made of will enhance the beer making process. This will help make changes to the recipe more efficient. Having an organized way to keep track of this information will give the homebrewer the ability to focus on creating great tasting beers.

Database Outline in Words

You are starting an at home brewing project, and you want to implement a database that represents the following description.

You make beers. A beer has a BNum, BeerName, and Alc.

A beer has exactly one type of hop. A hop has HNum, HopName, DatePick, DateExpired, WeightOz, NationGrown, and Form.

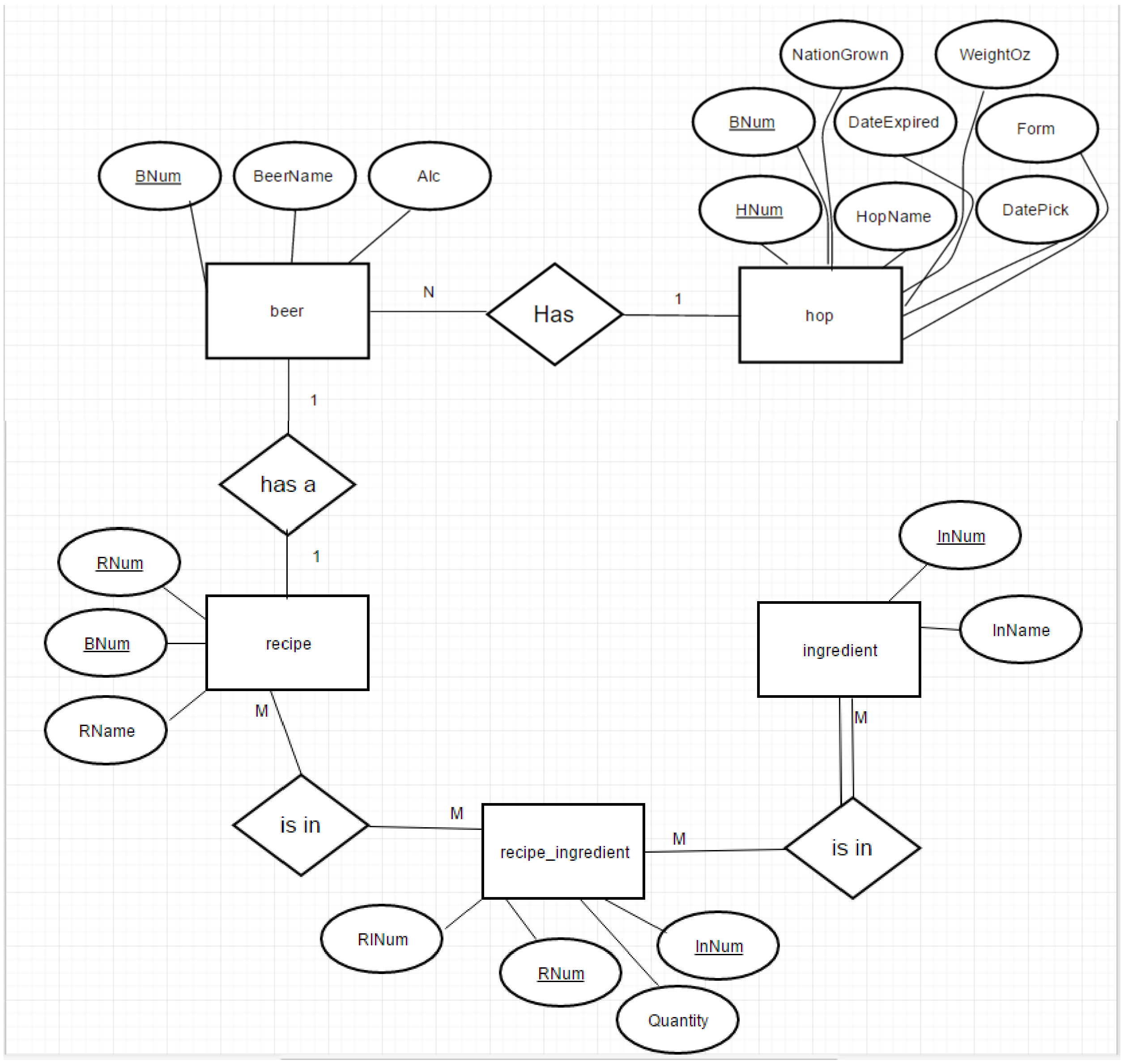
You keep track of how each beer is made. A beer is made up of exactly one recipe. A recipe has RNum, RName.

An ingredient consists of one or more recipes. A recipe can be in zero or more ingredients An ingredient is made up of InNum, and InName. The quantity of each ingredient used should be stored.

Assume IDs are unique for each entity.

ER Diagram of Database

Below is the ER Diagram for the Homebrew Database.



Database Schema

Below is a capture of every attribute of every table within the Homebrew Database. Primary Keys and Foreign Keys are referenced within each of the tables.

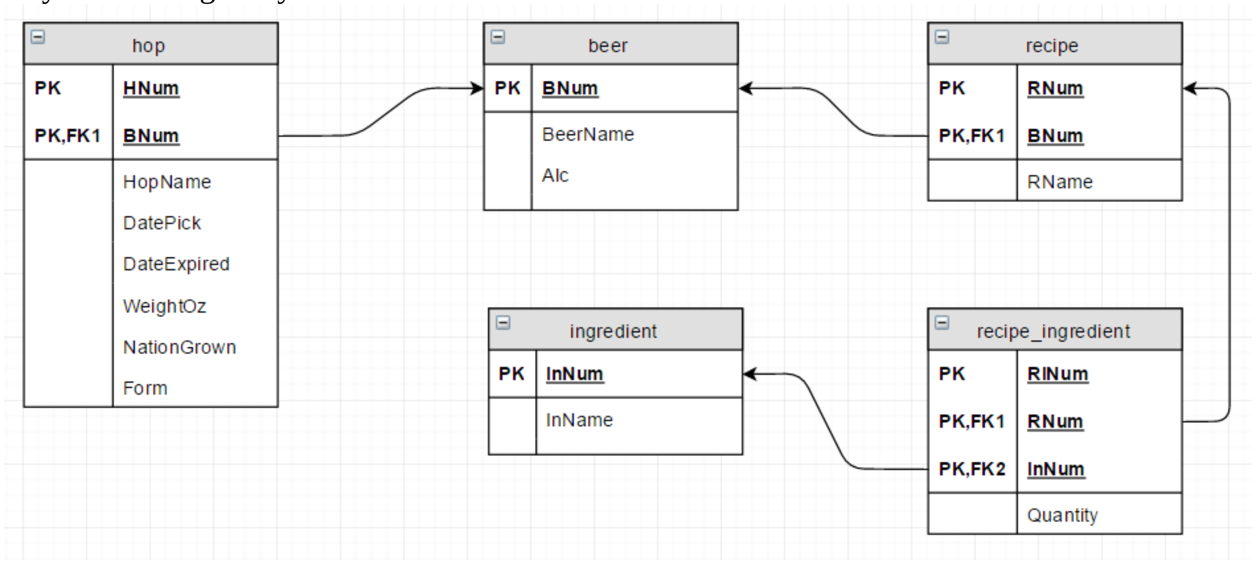


Table Creation Queries

-- beer table with the following properties:
-- BNum - an auto incrementing integer which is the primary key
-- BeerName - a varchar with a maximum length of 255 characters, cannot be null
-- Alc - a double

```
DROP TABLE IF EXISTS `beer`;
CREATE TABLE `beer` (
 `BNum` int(11) NOT NULL AUTO_INCREMENT,
 `BeerName` varchar(255) NOT NULL,
 `Alc` double,
 PRIMARY KEY (`BNum`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

-- hop table with the following properties:
-- HNum - an auto incrementing integer which is the primary key
-- BNum- an integer which is a foreign key reference to beer
-- HopName - a varchar with a maximum length of 255 characters, cannot be null
-- DatePick - a date type, cannot be null
-- DateExpired - a date type, cannot be null
-- WeightOz - a double
-- NationGrown - a varchar with a maximum length of 25 characters
-- Form - a varchar with a maximum length of 25 characters

```
DROP TABLE IF EXISTS `hop`;
CREATE TABLE `hop` (
 `HNum` int(11) NOT NULL AUTO_INCREMENT,
 `BNum` int(11) NOT NULL,
 `HopName` varchar(255) NOT NULL,
 `DatePick` DATE NOT NULL,
 `DateExpired` DATE NOT NULL,
 `WeightOz` double,
 `NationGrown` varchar(25),
 `Form` varchar(25),
 PRIMARY KEY (`HNum`,`BNum`),
 KEY `idx_fk_BNum` (`BNum`),
 CONSTRAINT `fk_beer_hop` FOREIGN KEY (`BNum`) REFERENCES `beer` (`BNum`) ON
UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


-- recipe table with the following properties:
-- RNum - an auto incrementing integer which is the primary key
-- BNum- an integer which is a foreign key reference to beer
-- name - a varchar with a maximum length of 255 characters, cannot be null

DROP TABLE IF EXISTS `recipe`;
CREATE TABLE `recipe` (
 `RNum` int(11) NOT NULL AUTO_INCREMENT,
 `BNum` int(11) NOT NULL,
 `RName` varchar(255) NOT NULL,
 PRIMARY KEY (`RNum`,`BNum`),
 KEY `idx_fk_bid` (`BNum`),
 CONSTRAINT `fk_beer_recipe` FOREIGN KEY (`BNum`) REFERENCES `beer` (`BNum`)
ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ingredient table with the following properties:
-- INum - an auto incrementing integer which is the primary key
-- InName - a varchar with a maximum length of 255 characters, cannot be null

DROP TABLE IF EXISTS `ingredient`;
CREATE TABLE `ingredient` (
 `InNum` int(11) NOT NULL AUTO_INCREMENT,
 `InName` varchar(255) NOT NULL,
 PRIMARY KEY (`InNum`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- Recipe_ingredient table with the following properties, this is a table representing a many-to-
many relationship
```

```
-- between recipe and ingredient:
-- RINum - an auto incrementing integer which is the primary key
-- RNum - an integer which is a foreign key reference to beer
-- InNum - an integer which is a foreign key reference to ingredient
-- Quantity - a TINYINT which shows the quantity of the ingredient
-- The primary key is a combination of eid and pid

DROP TABLE IF EXISTS `recipe_ingredient`;
CREATE TABLE `recipe_ingredient` (
 `RINum` int(11) NOT NULL AUTO_INCREMENT,
 `RNum` INT(11) NOT NULL,
 `InNum` INT(11) NOT NULL,
 `Quantity` TINYINT NOT NULL,
 PRIMARY KEY (`RINum`),
 CONSTRAINT `fk_recipe_recipe_ingredient` FOREIGN KEY (`RNum`) REFERENCES
`recipe` (`RNum`) ON UPDATE CASCADE,
 CONSTRAINT `fk_ingredient_recipe_ingredient`FOREIGN KEY (`InNum`) REFERENCES
`ingredient` (`InNum`)
) ENGINE=INNODB DEFAULT CHARSET=utf8;
```

General Use Queries

Below are the general use queries which will be used in the Homebrew Database:

-- *List each beer from the database with Beer ID, Beer Name, and Beer's ABV*
SELECT BNum, BeerName, Alc FROM beer;

-- *Insert Beer Name and Beer's ABV into table*
INSERT INTO beer(BeerName, Alc) VALUES ([name], [abv]);

-- *List each hops from the database with hops id, hops name, date picked, expiration date, weight in oz., nation grown, form, and Beer's Name associated with the hop.*
SELECT hop.HNum, hop.HopName, hop.DatePick, hop.DateExpired, hop.WeightOz, hop.NationGrown, hop.Form, hop.BNum, beer.BeerName FROM hop INNER JOIN beer ON beer.BNum = hop.BNum;

-- *List each beer from the beer table to be dynamically selected to be inserted into hop*
SELECT BNum, BeerName FROM beer;

-- *Insert Hop name, date picked, expiration date, weight in oz, nation grown, hop form, and beer ID into hop table*
INSERT INTO hop(HNum, HopName, DatePick, DateExpired, WeightOz, NationGrown, Form, BNum) VALUES ([name], [date picked], [expiration date], [weight in oz], [nation grown], [hop form], [beer ID]);

-- *List each ingredient from the ingredient table*

```
SELECT InNum, InName FROM ingredient;

-- Insert Ingredient name into Ingredient table
INSERT INTO ingredient(InName) VALUES ([name]);

-- Update the Ingredient name using the ingredient ID in the ingredient table
UPDATE ingredient SET InName= [name] WHERE InNum= [ingredient id];

-- List each recipe from the recipe table and the beer that is associated with that recipe
SELECT recipe.RNum, recipe.RName, beer.BeerName, beer.Alc FROM recipe INNER JOIN
beer ON beer.BNum = recipe.BNum;

-- Insert recipe name and beer ID into the recipe table
INSERT INTO recipe(RName, BNum) VALUES ([name], [beer ID]);

-- List the current Recipe_Ingredient table with the recipe_ingredient ID, Recipe ID and Recipe
Name associated with the table, Ingredient ID and Ingredient Name associated with the
recipe_ingredient table, and the Ingredient Quantity
SELECT recipe_ingredient.RINum, recipe_ingredient.RNum, recipe.RName, ingredient.InNum,
ingredient.InName, recipe_ingredient.Quantity FROM recipe INNER JOIN recipe_ingredient
ON recipe.RNum = recipe_ingredient.RNum INNER JOIN ingredient ON ingredient.InNum =
recipe_ingredient.InNum ORDER BY recipe_ingredient.RINum;

-- List the recipe ID and recipe name to dynamically choose from the recipe table
SELECT RNum, RName FROM recipe;

-- List the ingredient ID and ingredient name to dynmaically choose from the ingredient table
SELECT InNum, InName FROM ingredient;

-- Insert recipe number, ingredient number, and quantity into the recipe_ingredient table
INSERT INTO recipe_ingredient(RNum, InNum, Quantity) VALUES ([recipe number],
[ingredient number], [quantity]);
```

Homebrew Database Link and Site Source Code

Please see our Homebrew database at the following link:
http://web.engr.oregonstate.edu/~schachtj/CS340/project/main.html

Included in the Gonzalez_Schachtsick_SiteSourceCode.zip is a folder containing the code for the Site Pages. There is another folder with SQL Queries containing table creation and general use queries in a .sql file.