

CS 486/586 Midterm

Fall 2021

Updates & Clarifications:

- Q13 - No need to keep the foreign key constraint.
-

Instructions.

This file contains the Midterm Exam for CS 486-586 - Fall 2021.

To complete this exam, please make a copy of the google doc and edit it. When you are ready to turn in your exam, please save the doc as a pdf and turn it in on D2L or Canvas.

The midterm is ***due by midnight on Friday Nov 5***. The extended amount of time for the exam is intended so that students can work the exam into their schedules. I anticipate the exam will take on the order of 6-12 hours.

This exam is ***open book, open notes, open Internet, open everything***. However, if you use material from the Internet or sources other than the textbook, course slides, your notes or the PostgreSQL documentation, ***you must cite the web site or material that you used***. For example, you could include a link to the web site you used or the name of a book you used in your answer.

200 points total.

Please do be sure to read questions completely and note that some questions ask you to include the results in addition to the query. Please pay attention to the "Include in your answer" notes in the questions.

Breakfast Schema:

BreakfastFoods

Id	Name	Calories	Fat	Sodium	Carbs	MadeBy
1	Bacon, Sausage & Egg Wrap	640	33	1090	58	5
2	Scrambled Eggs (2 eggs)	149	11	145	1.6	--
3	Peanut Butter Homestyle Granola	140	7	65	18	3
4	Nonfat Greek Yogurt	90	0	65	5	4
5	Coffee with half and half	42	3.5	19	1.5	5

Consumers

Id	Name	FavoriteBreakfast
1	Olivia	3
2	Roman	--
3	Mikhail	4
4	Daniela	5
5	Miranda	5
6	Kiran	4

Companies

Id	Name	StockSymbol
1	Hostess Brands, Inc	TWNK
2	J.M. Smucker Company	SJM
3	Bob's Red Mill	--
4	Fage	--
5	Starbucks	SBUX

Notes:

- Consumers are people who eat breakfast.
- Calories, Fat, Sodium and Carbs are per serving.
- Bob's Red Mill and Fage are privately held and so do not have a stock symbol.
- Scrambled eggs are homemade, so 'Made by' is null

Sources:

- <https://www.starbucks.com>
- <https://www.bobsredmill.com/>
- <https://fdc.nal.usda.gov>
- <https://usa.fage>

Q1. (15 pts) Write Create Table commands to create the three tables above in your database and create a primary key for each table. Add any foreign keys as appropriate.

Include in your answer the table creation commands and any command to create keys and foreign keys.

Notes:

- You may create the primary keys and foreign keys either as part of the table creation statement or using Alter Table to add the keys after you create the table
- Assume that the names in all tables can be of arbitrary length
- Assume that the stock symbol can have a maximum length of 6 characters
- Assume that fat and carbs are floating point numbers
- All other fields should be assumed to be integers

Solution:

CREATE TABLE companies

```
(
  id      INT PRIMARY KEY,
  NAME    TEXT NOT NULL,
  stocksymbol VARCHAR(6)
);
```

CREATE TABLE breakfastfoods

```
(
  id      INT PRIMARY KEY,
  name    TEXT NOT NULL,
  calories INT NOT NULL,
  fat     FLOAT NOT NULL,
  sodium  INT NOT NULL,
  carbs   FLOAT NOT NULL,
  madeby  INT
);
```

CREATE TABLE consumers

```
(
  id          INT PRIMARY KEY,
  NAME        TEXT NOT NULL,
  favoritebreakfast INT);
```

---Adding FK---

ALTER TABLE breakfastfoods

ADD CONSTRAINT fk_id FOREIGN KEY (madeby) REFERENCES companies(id);

ALTER TABLE consumers

ADD CONSTRAINT fk_id FOREIGN KEY (favoritebreakfast) REFERENCES breakfastfoods(id);

Q2. (10 pts) Create csv files for the data shown in the tables above. Load all the data into the tables. You may use either the command line or the phppgadmin gui. Include in your answer a screenshot to show that you have successfully loaded the data.

```
SELECT * FROM "fall2021db73"."breakfastfoods";
```

Submit

Actions		id	name	calories	fat	sodium	carbs	madeby
Edit	Delete	1	Bacon,Sausage & Egg Wrap	640	33	1090	58	5
Edit	Delete	2	Scrambled Eggs (2 eggs)	149	11	145	1.6	NULL
Edit	Delete	3	Peanut Butter Homestyle Granola	140	7	65	18	3
Edit	Delete	4	Nonfat Greek Yogurt	90	0	65	5	4
Edit	Delete	5	Coffee with half and half	42	3.5	19	1.5	5

5 row(s)

```
SELECT * FROM "fall2021db73"."companies";
```

Submit






Actions	id	name	stocksymbol
Edit Delete	1	Hostess Brands, Inc	TWNK
Edit Delete	2	J.M. Smucker Company	SJM
Edit Delete	3	Bob's Red Mill	NULL
Edit Delete	4	Fage	NULL
Edit Delete	5	Starbucks	SBUX

5 row(s)

[Cancel](#) | [Insert](#) | [Refresh](#)

```
SELECT * FROM "fall2021db73"."consumers";
```

Submit

Actions	id	name	favoritebreakfast
Edit Delete	1	Olivia	 3
Edit Delete	2	Roman	NULL
Edit Delete	3	Mikhail	 4
Edit Delete	4	Daniela	 5
Edit Delete	5	Miranda	 5
Edit Delete	6	Kiran	 4

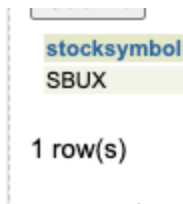
6 row(s)

Q3. (10 pts) Write a query to find the stock symbol of the company that makes Daniela's favorite breakfast food. (You must use the name 'Daniela' in the selection predicate, not Daniela's Id.)

Include your query and the result in your answer.

Solution:

```
SELECT stocksymbol
FROM companies
    JOIN breakfastfoods
        ON companies.id = breakfastfoods.madeby
    JOIN consumers
        ON breakfastfoods.id = consumers.favoritebreakfast
WHERE consumers.NAME = 'Daniela';
```



A screenshot of a database query result. It shows a single row with the column 'stocksymbol' and the value 'SBUX'. Below the table, it says '1 row(s)'.

stocksymbol
SBUX

1 row(s)

Q4. (10 pts) Write a SQL query to make a list of consumers and their favorite. Include consumers that do not have a favorite breakfast food.

Include your query and the result in your answer.

Solution:

```
SELECT consumers.NAME AS Cusumer_name,
    consumers.favoritebreakfast,
    breakfastfoods.NAME AS favorite
FROM consumers
    LEFT JOIN breakfastfoods
        ON consumers.favoritebreakfast = breakfastfoods.id
ORDER BY consumers.NAME;
```

cusumer_name	favoritebreakfast	favorite
Daniela	5	Coffee with half and half
Kiran	4	Nonfat Greek Yogurt
Mikhail	4	Nonfat Greek Yogurt
Miranda	5	Coffee with half and half
Olivia	3	Peanut Butter Homestyle Granola
Roman	NULL	NULL

6 row(s)

Q5: (10 pts) Write a SQL query to find the names and calories of breakfast foods that are *not* Kiran's favorite breakfast food. Use a subquery.

Include your query and the result in your answer.

Solution:

```
SELECT NAME,  
       calories  
FROM breakfastfoods  
WHERE NOT EXISTS (SELECT NAME  
                  FROM consumers  
                  WHERE breakfastfoods.id = consumers.favoritebreakfast  
                    AND NAME = 'Kiran');
```

name	calories
Bacon,Sausage & Egg Wrap	640
Scrambled Eggs (2 eggs)	149
Peanut Butter Homestyle Granola	140
Coffee with half and half	42

4 row(s)

Q6: (15 pts) Write two SQL queries to find the names of breakfast foods made by Starbucks. One query should use a subquery in the WHERE clause, one query should not use a subquery.

Include your query and the result in your answer.

Solution:

```
SELECT breakfastfoods.NAME  
FROM breakfastfoods  
      JOIN companies  
      ON breakfastfoods.madeby = companies.id  
WHERE companies.NAME = 'Starbucks';
```

name
Bacon,Sausage & Egg Wrap
Coffee with half and half

2 row(s)

```
SELECT NAME
FROM breakfastfoods
WHERE madeby in (SELECT id
                  FROM companies
                  WHERE name = 'Starbucks');
```

name
Bacon,Sausage & Egg Wrap
Coffee with half and half

2 row(s)

Q7: (10 pts) Write a query to list all breakfast foods that at least 2 consumers have listed as favorites. List breakfastfood name in your answer.

Include your query and the result in your answer.

Solution:

```
SELECT NAME
FROM breakfastfoods
WHERE id IN (SELECT favoritebreakfast
             FROM consumers
             GROUP BY favoritebreakfast
             HAVING Count(favoritebreakfast) >= 2);
```

name
Nonfat Greek Yogurt
Coffee with half and half

2 row(s)

Q8: (10 pts) Write two queries: 1) count the number of consumers and 2) count the number of consumers that have favorite breakfasts. Do not use NULL (IS NULL, IS NOT NULL, etc.) in your query.

Include your query and the result in your answer.

Solution:

1) `SELECT count(*)
FROM consumers;`

count
6

1 row(s)

2) `SELECT count(*)
FROM consumers
WHERE favoritebreakfast >= 1;`

count
5

1 row(s)

Q9: (15 pts) Write a query to find for each breakfast food, the number of consumers who list that food as their favorite. List breakfastfood name and count in your result. Name the count column NumFavorite. Be sure to include breakfast foods that are no one's favorite. Use UNION in your answer. (Using UNION is required.)

Include your query and the result in your answer.

Solution:

`(SELECT breakfastfoods.NAME,
Count(consumers.favoritebreakfast) AS NumFavorite
FROM breakfastfoods`

```

JOIN consumers
  ON breakfastfoods.id = consumers.favoritebreakfast
GROUP BY breakfastfoods.NAME
HAVING Count(consumers.favoritebreakfast) >= 1
ORDER BY NumFavorite )

```

UNION

```

(SELECT NAME, 0 as NumFavorite
FROM breakfastfoods
WHERE id NOT IN (SELECT consumers.favoritebreakfast
                 FROM consumers
                 WHERE breakfastfoods.id = consumers.favoritebreakfast));

```

name	numfavorite
Bacon,Sausage & Egg Wrap	0
Peanut Butter Homestyle Granola	1
Coffee with half and half	2
Scrambled Eggs (2 eggs)	0
Nonfat Greek Yogurt	2

5 row(s)

Q10: (15 pts) Write a SQL query to find the name of the breakfast foods that are the favorite of the most consumers. You may ignore nulls - that is, for this question focus only on breakfast foods that are the favorite of at least one consumer.

Include your query and the result in your answer.

Solution:

```

SELECT NAME
FROM breakfastfoods
WHERE id IN (SELECT favoritebreakfast
             FROM consumers
             GROUP BY favoritebreakfast
             HAVING Count(favoritebreakfast) >= 2);

```

name
Nonfat Greek Yogurt
Coffee with half and half

2 row(s)

Q11: (15 pts) Write two relational algebra expressions to find the name of and the amount of calories in Olivia's favorite breakfast food and the name of the company that makes that food. The two relational algebra expressions should be different, but should be guaranteed to give the same result.

Solution:

1.

Π breakfastfoods.name, breakfastfoods.calories, companies.name ($\text{companies} \bowtie \text{id}=\text{madeby breakfastfoods (breakfastfoods} \bowtie \text{id=favoritebreakfast consumers))} \sigma \text{ Consumers.name = 'Olivia'}$

2.

Π breakfastfoods.name, breakfastfoods.calories, companies.name (σ breakfastfoods.id = (Π favoritebreakfast σ consumers.name='Olivia' consumers) breakfastfoods, companies, consumers)

\wedge

Π breakfastfoods.name, breakfastfoods.calories, companies.name (σ companies.id = breakfastfoods.madeby breakfastfoods, companies, consumers)

Q12: (10 pts) Give an example of a query that can be expressed in SQL, but which cannot be expressed in pure relational algebra. Write the SQL and explain why the query cannot be expressed in RA.

Solution:

Table1

Bag_Color	Count
Green	4

Yellow	5
Yellow	5

SQL Query:

Select Bag_Color,Count from Table1;

RA Query:

$\Pi_{\text{Bag_color,Count}}(\text{Table1})$

Explanation:

If we use an SQL query, it will select all the columns and values. Then give the result with duplicate values.

In relational algebra, It will eliminate the duplicates then give the result. Because Π is eliminating the duplicates. If we add distinct in SQL, it will give the result without duplication. But in RA, Π itself removes duplicates.

So, the above query can be expressed in SQL, but cannot be expressed in pure relational algebra.

Q13: (10 pts) Some consumers may have multiple favorite breakfast foods; for example, Miranda likes both Coffee and Nonfat Greek Yogurt. Modify the Breakfast Schema by adding a column called FavBreakArray that uses an *Array Type* to allow a Consumer to have multiple favorite breakfast foods. Write DDL statements to modify the schema and write an insert or update command to insert a row showing Miranda's favorite breakfast foods.

Include in your answer: your alter table statements and your insert or update statement.

No need to drop the existing FavoriteBreakfast column, you may leave it.

Postgres array documentation: <https://www.postgresql.org/docs/14/arrays.html>

Solution:

```
ALTER TABLE consumers
ADD FavBreakArray integer[];
```

```
UPDATE consumers
SET FavBreakArray = ARRAY[1, 2, 3, 4]
WHERE name = 'Miranda';
```

Actions		id	name	favoritebreakfast	favbreakarray
Edit	Delete	1	Olivia	☞3	NULL
Edit	Delete	2	Roman	NULL	NULL
Edit	Delete	3	Mikhail	☞4	NULL
Edit	Delete	4	Daniela	☞5	NULL
Edit	Delete	6	Kiran	☞4	NULL
Edit	Delete	5	Miranda	☞5	{1,2,3,4}

6 row(s)

Q14: (10 pts) First, insert the favorite breakfast information from the FavoriteBreakfast column into the new FavBreakArray column for all consumers other than Miranda. Using this new FavBreakArray column, find all consumers that have Coffee with half and half as one of their favorite breakfast.

Include your query and the result in your answer.

Solution:

```
UPDATE consumers SET favbreakarray = array[5] WHERE name = 'Miranda';
```

```
UPDATE consumers SET favbreakarray = array[3] WHERE name = 'Olivia';
```

```
UPDATE consumers SET favbreakarray = array[4] WHERE name = 'Mikhail';
```

```
UPDATE consumers SET favbreakarray = array[5] WHERE name = 'Daniela';
```

```
UPDATE consumers SET favbreakarray = array[4] WHERE name = 'Kiran';
```

```

SELECT consumers.NAME
FROM   consumers
       JOIN breakfastfoods
         ON breakfastfoods.id = ANY (consumers.favbreakarray)
WHERE  breakfastfoods.NAME = 'Coffee with half and half';

```

name
Daniela
Miranda

2 row(s)

Q15: (10 points). Create an enum type named 'FoodGroupEnum' to categorize foods. Enum values allowed should be: Fats, Dairy, Protein, Vegetables, Grains. Add a column FoodGroup to the Breakfastfood table of type FoodGroupEnum. Update the table to add the FoodGroups for each breakfastfood.

Include in your answer: enum creation command; a sample update statement and your new table.

Note: Count Granola is a Grain; Eggs and Bacon, Sausage & Egg wrap as Proteins; Yogurt and Coffee with half and half as Dairy.

Postgres enum documentation:

<https://www.postgresql.org/docs/14/datatype-enum.html>

Solution:

```

create type FoodGroupEnum as enum ('Fat' , 'Dairy' , 'Protein' , 'Vegetables' ,
'Grains');

```

```

[fall2021db73=> create type FoodGroupEnum as enum ('Fat' , 'Dairy' , 'Protein' , 'Vegetables' , 'Grains');
CREATE TYPE
[fall2021db73=> █

```

```

ALTER TABLE breakfastfoods ADD foodgroup foodgroupenum;

```

```

fall2021db73=> ALTER TABLE breakfastfoods ADD foodgroup foodgroupenum;
ALTER TABLE
_

```

UPDATE breakfastfoods SET foodgroup = 'Grains' where name = 'Peanut Butter Homestyle Granola';
 UPDATE breakfastfoods SET foodgroup = 'Protein' where name = 'Bacon, Sausage & Egg Wrap';
 UPDATE breakfastfoods SET foodgroup = 'Protein' where name = 'Scrambled Eggs (2 eggs)';
 UPDATE breakfastfoods SET foodgroup = 'Dairy' where name = 'Nonfat Greek Yogurt';
 UPDATE breakfastfoods SET foodgroup = 'Dairy' where name = 'Coffee with half and half';

Actions		id	name	calories	fat	sodium	carbs	madeby	foodgroup
Edit	Delete	3	Peanut Butter Homestyle Granola	140	7	65	18	3	Grains
Edit	Delete	4	Nonfat Greek Yogurt	90	0	65	5	4	Dairy
Edit	Delete	5	Coffee with half and half	42	3.5	19	1.5	5	Dairy
Edit	Delete	2	Scrambled Eggs (2 eggs)	149	11	145	1.6	NULL	Protein
Edit	Delete	1	Bacon,Sausage & Egg Wrap	640	33	1090	58	5	Protein

5 row(s)

Q16: (15 points). Find the name and calories of the breakfast food with the lowest amount of fat in each food group.

Include your query and the result in your answer.

Solution:

SELECT NAME, calories FROM breakfastfoods
 WHERE fat = (SELECT Min(fat) FROM breakfastfoods WHERE foodgroup='Grains')

UNION

SELECT NAME, calories FROM breakfastfoods
 WHERE fat = (SELECT Min(fat) FROM breakfastfoods WHERE foodgroup='Dairy')

UNION

SELECT NAME, calories FROM breakfastfoods
 WHERE fat = (SELECT Min(fat) FROM breakfastfoods WHERE foodgroup='Protein');

name	calories
Peanut Butter Homestyle Granola	140
Scrambled Eggs (2 eggs)	149
Nonfat Greek Yogurt	90

3 row(s)

Q17: (10 pts) The table/schema below representing a list of ingredients is not normalized. Please normalize the table/schema.

In your answer, include table names and attribute names as in the first line below. Underline keys and indicate any foreign keys you would create.

Ingredients(id, name, foodgroupid, foodgroup)

<u>id</u>	name	foodgroupid	foodgroup
1	flour	10	grains
2	oil	11	fats
3	oats	12	grains

Solution:

Ingredients_group(id, name, foodgroupid)

foodgroupid is a foreign key that references food_group.id

<u>id</u>	name	foodgroupid
1	flour	10
2	oil	11
3	oats	10

Food_group(id, name)

<u>id</u>	name
10	grains
11	fats