**Part 1**

1) What is a good choice for a primary key here? In contrast, give an example of an

attribute (or composite) that would not be a valid primary key.

Ans : **index\_id is the primary key as it can identify tuple uniquely.**

(We can also consider index\_id,company as PK)

Example of invalid primary key

* category would not be primary key as it consists of multiple values and cannot identify tuple uniquely.
* numEmps cannot be primary key because it has null values.

prerequisites for primary key:

* The value of primary key should be unique for each row of the table.
* The column(s) that makes the key cannot contain duplicate values.
* The attribute(s) that is marked as primary key is not allowed to have null values.

2) For your choice of primary key, do the data satisfy 1NF? Why or why not?

index\_id is the pimary key.

Ans : **Yes the data is in 1 NF as all the values are atomic.**

For the data to be in 1 NF there should be no composite or multi valued data which is not present in the table.

Hence we conclude the data is in 1NF.

3) For your choice of primary key, do the data satisfy 2NF? Why or why not?

Ans: **No the data violates 2nf.**

There are repeated values for company, raised\_currency, round etc and normalization is done to reduce redundancy at each form.

company -> city,state

company fund\_date -> Raised\_curency, Round

(***Theory states that to violate 2nf the primary key must be composite (to get partial dependency). However the PK is singular so I am uncertain??***

***so i felt index\_id,company should be the primary key as this would be appropriate.so in this case***

***index\_id company -> city,state***

***company -> city,state***

***so there exists partial dependency hence violates 2nf***)

4) For your choice of primary key, do the data satisfy 3NF? Why or why not?

Ans : **No the data does not satisfy 3nf.**

because there exists transitive FD

Examples :

* index\_id -> company

company -> city,state (for this dataset at least)

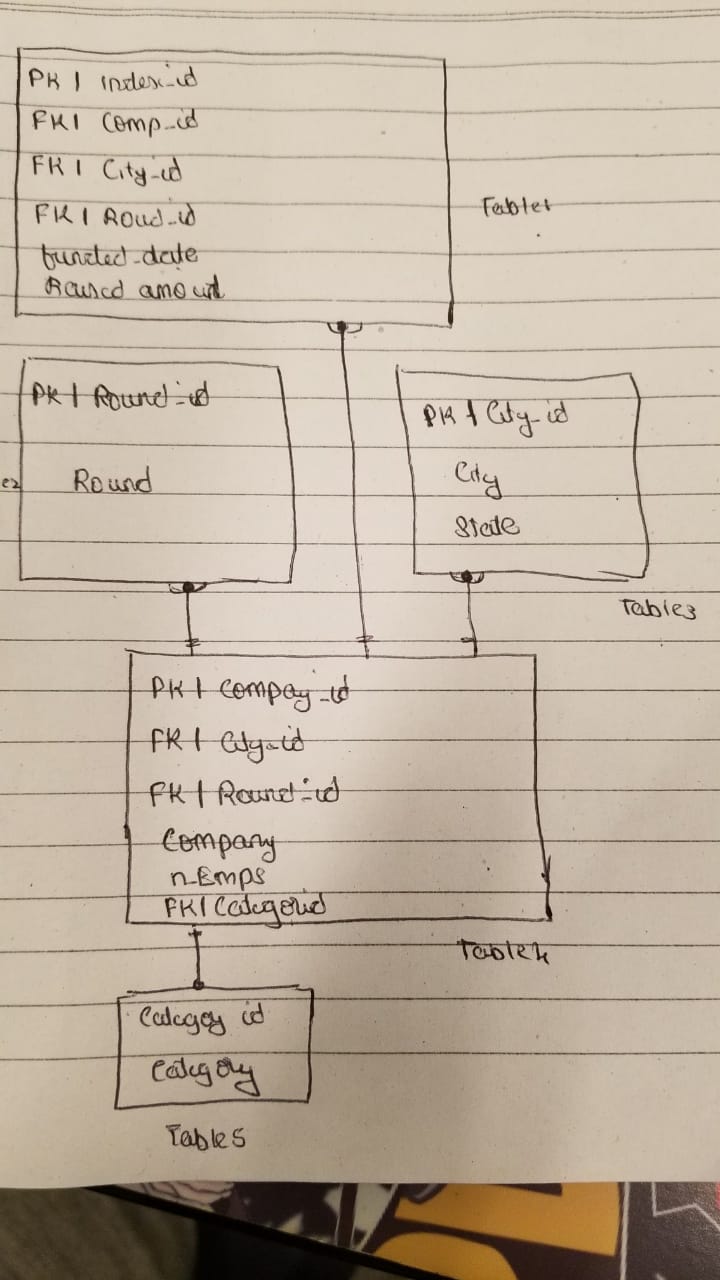
* index\_id -> city

City -> state

Q5) (10 pts) Sketch a proposed Entity-Relationship diagram that would bring this dataset into

3NF. If you answered “yes” to (4), for example, your ERD would just be the raw data table with

no changes.



There exists transitional dependency between company, city, state. Breaking the table into 5 parts and assigning intermediate keys guarantees 3nf form with no interdependencies.

**PART2**

1) The database is mostly empty. Use the INSERT INTO statement to insert information

about two (2) complete recipes of your choosing into the database. You can make up your own

recipes or copy them from a website.

Ans: When we try to insert there is a foreign key constraint error.

To over come that

* Add Category id and values to category table
* Insert the 2 recipes after the category values is added.

Sql code :

* **insert into recipes.categories select 4, 'Entree', 'Main Dinner';**

**insert into recipes.categories select 3, 'Chilli2', 'Main course';**

* **select 4, 'boiled egg', 3, 'boiled egg', 20, 35, 3, 2, 'boil and eat', '', 4, 'author2', 2, '2019-04-21 10:01:10';**

**SELECT 3, 'omlet', 3, 'continental', 200, 30, 1, 1, 'fry and eat', '', 3, 'author1', 1, '2019-04-21 10:01:10'**

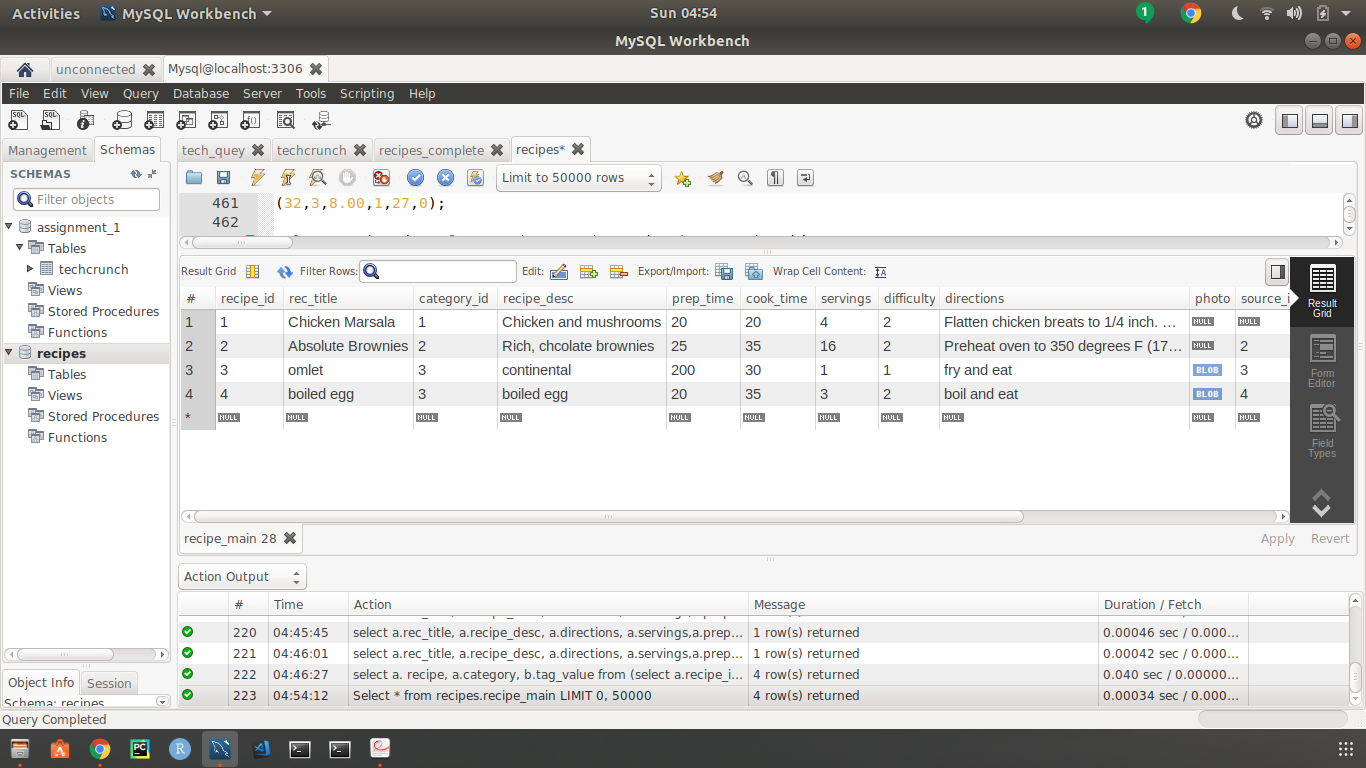


Table recipes.main After inserting 2 recipes.

Q2. Write a set of queries to return all information on a specific recipe including main details,

ingredients, recipe tags, nutrition, comments, food warnings and any available substitutions. Use as few queries as possible. Your set of queries should be designed to create something similar to Figure 9.14 in the textbook—a complete recipe page (in that case, Chicken Marsala) that might appear on a website. Remember to use aliases on the field names that are returned so that the raw query results will be more readable.

Ans : To replicate what is there in the cookbook we fill up the tables ingredients,

rec\_ingredients, rec\_comments, rec\_tags, nutrition etc with data related to

our recipe.

Sql Code for insertion into various tables:

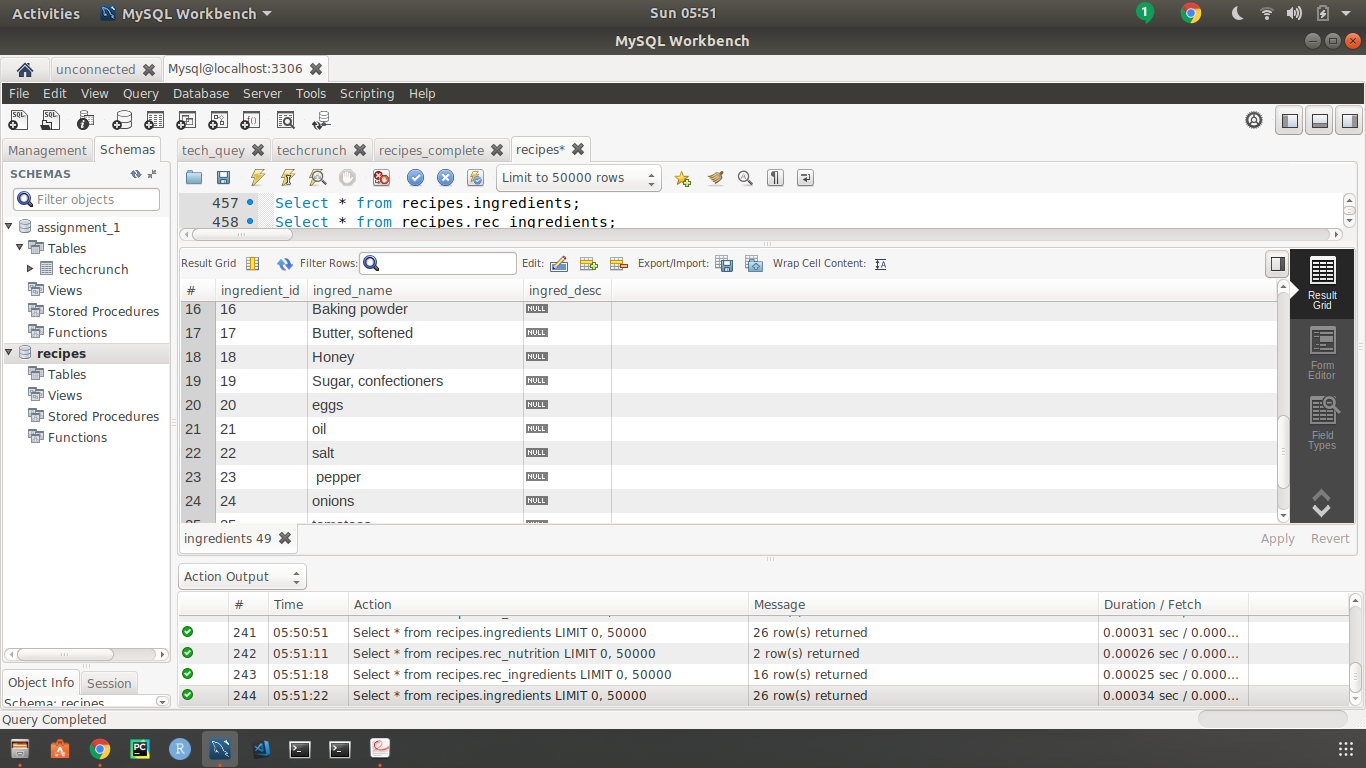
* i**nsert into recipes.food\_warnings select 4, 22, 'allergic',null;**
* **insert into recipes.rec\_nutrition select 3, 4, 300, 50, 95, 60, 10, 110, 40, 80, 75;**
* **insert into recipes.rec\_nutrition select 4, 4, 300, 50, 95, 60, 10, 110, 40, 80, 75;**
* **insert into recipes.substitutions select 2,20,1, 'flour instead of bread';**
* **insert into recipes.rec\_comments select 1, 3, '2019-04-04 18:10:16','use oil and butter';**

Sql Code & explanation for getting information:

* **select concat(a.amount,' ',a.unit\_id,' ', b.ingred\_name) as ingredients**

**from recipes.rec\_ingredients as a join recipes.ingredients as b on a.ingredient\_id = b.ingredient\_id where a.recipe\_id = 4;**

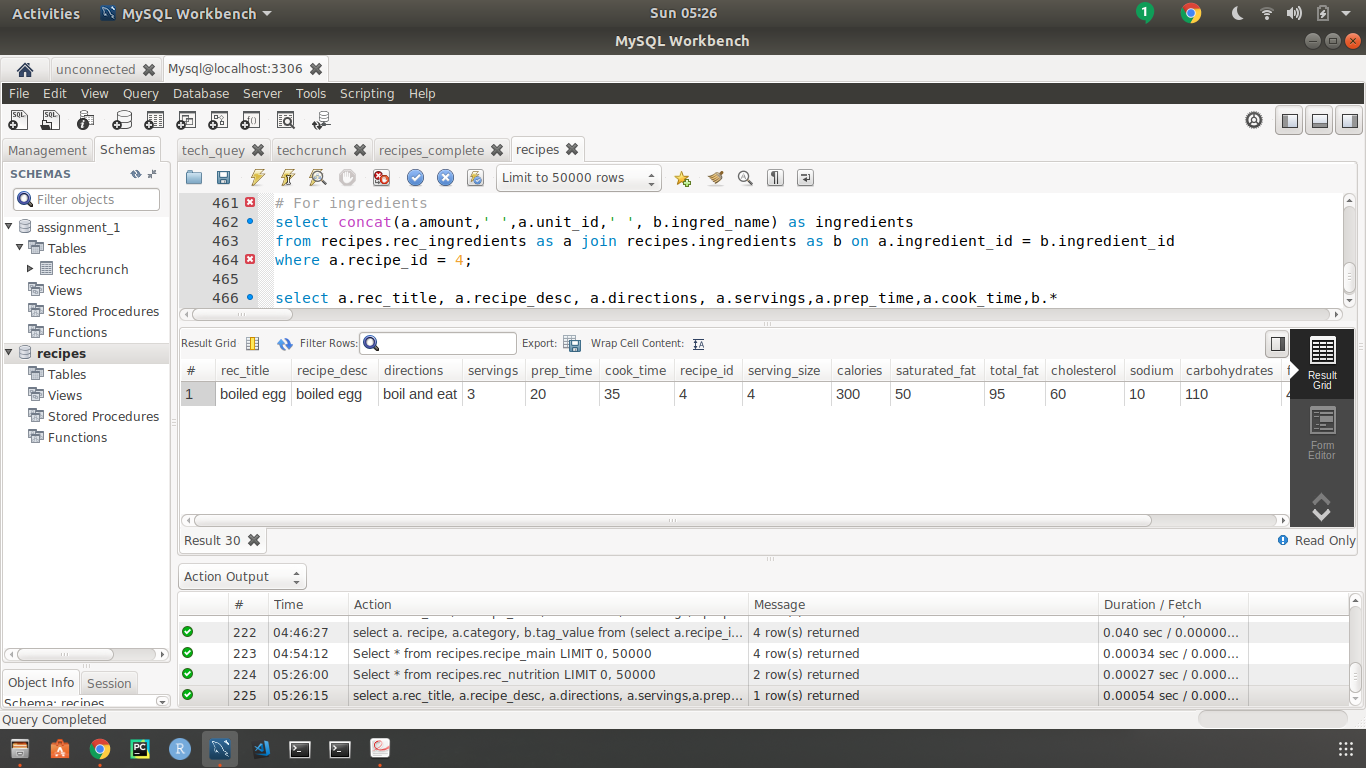
Left join on the ingredients and rec\_ingredients by using the parameter ingredient\_id gives new table which is used to concat function to get the output. Recipe\_id 4 is used to get details for that particular id

.

* **select a.rec\_title,a.recipe\_desc,a.directions,a.servings,a.prep\_time,a.cook\_time,b.\***

**from recipes.recipe\_main as a join recipes.rec\_nutrition as b on a.recipe\_id = b.recipe\_id where a.recipe\_id = 4;**

The recipe\_main table and the nutrition table is left joined with the parameter being the recipe\_id.The required values from each table is chosen by using conjunctions a and b.

****

Q3: Write a SELECT query that would supply the information that might be used in the back of a cookbook (the book’s index). Specifically, your results should be the recipe name, category,and all tag values. Your output should be sorted first by tag value and then by category. Show the results of your query on all data in your db.

Ans : To achieve this left join on the recipe\_main and the categories table with category id as the parameter.

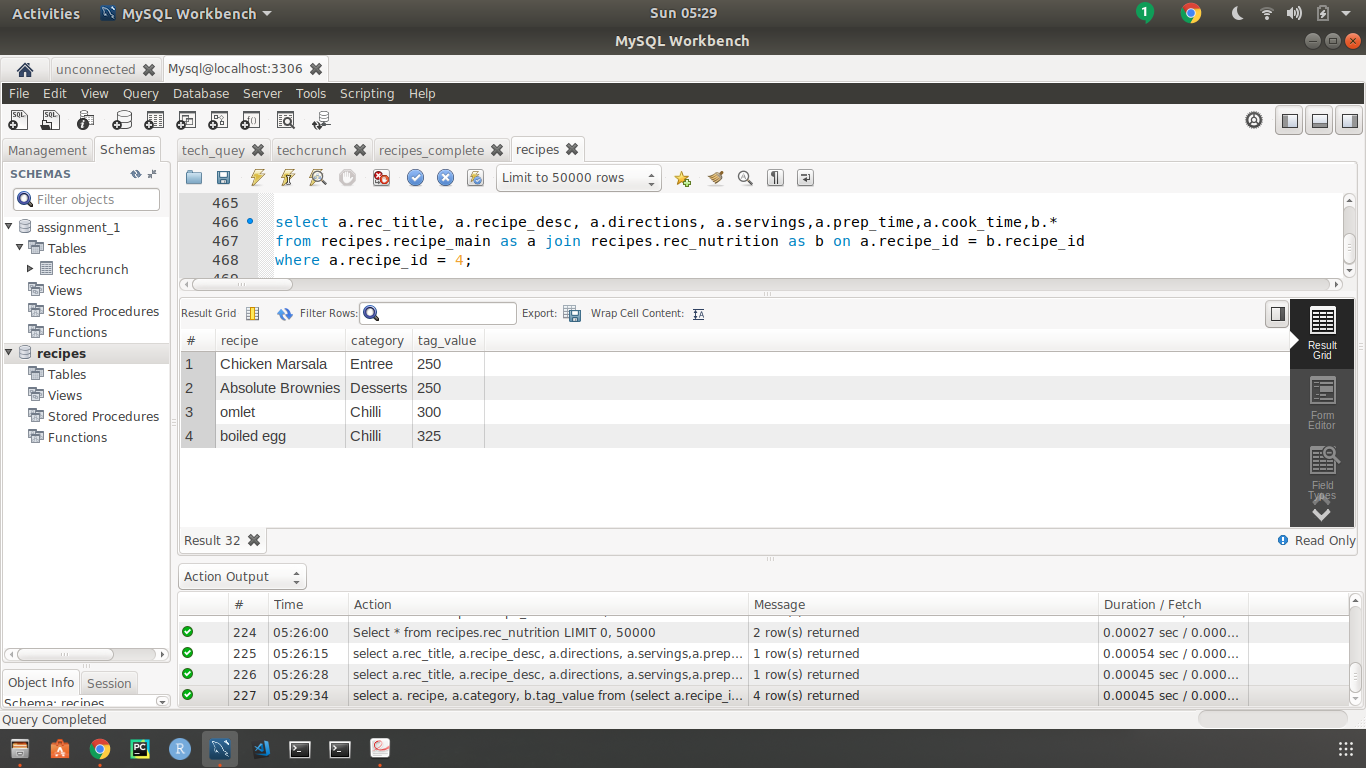
Then using this new formed table as a sub query to left join this table to the tags table using the recipe\_id as the parameter.

Sql Code :

**select a.rec\_title, a.recipe\_desc, a.directions, a.servings,a.prep\_time,a.cook\_time,b.\***

**from recipes.recipe\_main as a join recipes.rec\_nutrition as b on a.recipe\_id = b.recipe\_id**

**where a.recipe\_id = 4;**



Output showing sorted first by tag value and then by category