**CECS 323 Term Project**

## This is a team Project

Each team will be three to four students. You will need at least three strong participants to get this all done in the time allotted. Please let me know the team roster as soon as you form the team so that I can get group accounts set up promptly. Each of you will fill out a peer evaluation of the other members on the team. You can find that here. I also ask each team to agree among themselves on who will function as team lead. The team lead will be in charge of making assignments of specific tasks to specific team members, and getting status on a regular basis. I will be looking to the team leads for a weekly status report on how the team is doing. There will be extra credit points awarded to the team lead if their team feels that they are doing a good job.

## Lessons Learned by Your Predecessors

At the end of the Fall 2017 semester, I asked students to share with me lessons learned from performing on this project. They came up with the following (in no particular order):

- Unless you plan your time carefully, and stay on top of the deliverables, you **will** run out of time.
- Make sure that your relation scheme diagram is neat and easy to read early on. This will make everything much easier throughout the project.
- The data manipulation language (both the inserts and the selects) will take an immense amount of time if you are not careful. To keep that under control:
    - Read the queries in this rubric first, and plan your UML model around how you will answer the queries.
    - Use the MySQL workbench to interactively populate the data, and then export that data as insert statements.
    - Split the sample data up between at least two of the team members. This takes a long time to do because of all of the referential integrity constraints.
- Get the DDL done as soon as you are reasonably sure that the UML is solid. Start this at least two weeks ahead of the due date.
- Have someone in the team who has not done a given task, such as the queries, do a QA check before you move forward.
- Make use of a tool such as Google Drive to manage all of the files for the project.
- Use a group-messaging tool to keep in touch with each other.
- Start using the MySQL environment as soon it is issued so that you get familiar with it.
- Do not wait until the end to start on your triggers.
- Within the team, be very clear who has what responsibility. Do not change things without going through the person who is responsible for that part of the project.

- Write out your create table statements in the proper order so that the referential integrity constraints are a part of the create table statement.
- When you get feedback from me (your friendly professor) be sure to share it with the whole team.
- Get feedback early. Face to face is always best, but I take E-mail, as you doubtless know by now.
- Make a "to do" list of the project deliverables, and track % completion on all of them from the very beginning.
- Read this rubric several times. There are no "throw away" statements here!

Above all, have fun!

## A note about reading this rubric

I always tell my students that I am teaching a great number of things in this course, and not all of them are strictly technical. One skill that I want to impart to you is the ability to read a lengthy narrative (like this one) and sift out those portions of it that directly affect your objective, and which do not. Be on the lookout for the occasional mention of some fact that is irrelevant to this project. At the same time, I am also trying to teach each of you to know what you do not know. If you find that portions of this are unclear, or subject to multiple interpretations, ask for clarification. The best development efforts that I have served on were always a dialog between the analysts, customers, and developers. Dialogs take time, and often require multiple sessions to get it right, but the benefits in the end are always worth it.

## Project Description

You are designing a database for Miming's Chinese Cuisine restaurant, a small family-owned restaurant that specializes in authentic Chinese food for parties large and small.

### Orders

A given party may be too large to sit at just one table, so the guests within a party might be divided up between two or more tables. For privacy reasons, we do not take the names of any of our guests unless they are paying for the food. Therefore, it is a **seat at a table** within a party that places an order, not an individual person. It would take us too long to gather the information about each customer within the party. After all, we just want to make sure that we get the money at the end. Each party has just one bill. Just one Customer pays for a given bill.

### Miming's Money

We also encourage repeat customers. Each time that a customer places an order with us and pays cash, we put Miming's Money put into an "account" with us that they can apply to their next order. For every $10 of money that they spend, they get a $1 of Miming's Money deposited into their Miming's account. We only deal in whole numbers, so an order for $137.42, for instance, would give that customer $13 of Miming's Money. If the customers has enough Miming's money to pay for the entire bill, then that amount is deducted from their Miming's account and they do not pay us anything for that bill. Each bill only has one type of payment (e.g. credit card, debit card, Miming's money, money order, …).

## Customer Information

Customers **can** remain anonymous if they pay in cash. However, if they give us some basic information about them, we can award them Miming's Money (see below) to reward their loyalty. In addition, we will be sure to let them know of specials that we have from time to time. What we would like to know about each private customer is:

- Name
- E-mail address
- Snail-mail address

If they represent a **corporation** (for instance, they are buying for a corporate function) –

- The name of the person
- The name of the corporation
- The name of the organization within the corporation (say Sales, Fabrication, Engineering, …)
- The address of their office
- The contact information for someone that we can advise of specials

A specific person **could** come into Miming's as a private individual and as a representative of a corporation. But they will have two separate Miming's accounts, one as a private individual and one as a member of a corporation.

## Scheduling of staff

To keep things simple, the employees work shifts. Each day has just two shifts: morning and evening. A given employee always works a full shift, no matter what their function is. Each shift is eight hours. Many of our employees are part-time, so we pay them by the hour. Others are on salary and receive a weekly set rate. Only the salaried employees receive health care benefits. It's not that we do not care about our employees, it is just that we care a lot more about our customers.

## Staff

We treat our staff like family, because most of them are. Each employee at Miming's has a specific title and function:

- Chef – prepares the food. In turn, the Chef can fall into one and only one of the following categories.
    - Head Chef
        - Designs new recipes and oversees the rest of the chefs. We keep track of which of our recipes were developed by a given head chef
        - We must have one and only one of our head chefs in the kitchen whenever Miming's is open.
    - Sous Chef

        - Works with the head chef but tends to be more “hands on”.
        - Each sous chef is expert in preparing a set of menu items. A sous chef who wishes to learn how to prepare a given menu item must enter a mentoring relationship with one of the other sous chefs who is an expert in that menu item. When they do, Miming’s records the start date, the menu item, and the end date of the mentorship.
    - Line Cook
        - Works in at station (butcher, fry cook, grill chef, pantry chef, pastry chef, roast chef, sauté chef (in charge of the sauces and gravies) and vegetable chef (this is the one who earns the highest celery).
        - **During a given shift**, more than one line cook could staff any one of these stations, and a given line cook might be in charge of more than one station. Typically, during a time of day that is usually slow, we will assign one line cook to several stations so that we get the best value for their time.
    - All our cooks are full time employees and work on salary.
- Maître d’ – welcomes guests, gets them seated, and evenly distributes the patrons across the wait staff who are on duty.
    - All our Maître d’s have a fixed, hourly rate.
    - Each shift needs at least one Maître d’.
- The wait staff – takes the orders, brings the food and drink and helps the customers with any questions that they might have.
    - During a given shift, each wait staff member covers some collection of tables. That wait staff team member has those tables for their entire shift. No other wait staff member serves any of those tables during a given shift.
    - In Miming’s, we believe in the safety of our customers, so we bolt the tables to the floor, which means that we can never “push the tables together” to make a larger table. But we **can** break a given party up and spread them across several tables when needed.
    - The wait staff also has a fixed hourly rate that they receive in addition to their tips.
- Dishwasher – fixed hourly rate.
- Manager – Full-time salaried
    - Each shift needs one and only one manager.

## Menu Items

- Each menu item has a single spiciness associated with it. The spiciness values are:
    - Mild

    - Tangy
    - Piquant
    - Hot
    - Oh My God
- Each menu item falls into one of several categories[1]:
    - Appetizer
    - Soup
    - Meat Entrees
        - Chow Mein
        - Egg Foo Young
        - Chop Suey
        - Each of the meat entrees comes with a particular type of meat. The meat must be one of: Chef Special, Pork, Chicken, Beef, Seafood, or Vegetables. Do not ask what is in the Chef Special. It's better that you don't know.

## Pricing

Miming's has four menus:

- Evening
- Lunch
- Sunday brunch buffet
- Children's

A given menu item can appear on any of the four menus, but it must appear on no less than one of them. The same menu item showing up on the **evening** menu will cost the customer more money than the same menu item on the **lunch** menu. We reduce the portion size for the lunch menu, so we can offer them for less. The All You Can Eat Sensational Suicide Sunday Brunch has no prices for the individual items at all, since each customer pays a fixed rate to partake of our superlatively sumptuous buffet. As you would guess, the portions on the Children's menu are even smaller than they are on the Lunch menu. None of our spicy items go into the children's menu.

Each menu item comes in only one portion size within a given menu.

## Orders

An order can come in via the web, the phone, or "eat in". They must pay for their order when they place the order on the web. The customer pays for their phone order when the pick it up. Miming's records the type of payment used on each bill. The phone order and eat in orders can be paid for by cash, debit

[1] I borrowed from https://www.goldenphoenixinla.com/ for these.

card, or credit card. Evil customers have taken advantage of Miming's with bad checks too often in the past to allow customers to use them as a form of payment. For the "to go" orders, we give the customer a time when they can come and pick it up. We also record exactly when the order was ready so that we can be sure to have the food hot and ready, no matter how much they order.

# Output

**This project does <u>not</u>** require any graphical user interface front end. All the interaction between your database and the user will be through the command line.

## Additional Business Rules

You will add five business rules to the above business rules. You will have to provide some means to enforce these business rules in your database, either by means of one or more triggers or a database constraint such as referential integrity, uniqueness constraint, not null constraint or the like. The business rule needs to be something that will show up in the model of your design. For instance, your business rule might be that a given sous chef cannot be mentoring on more than three menu items at the same time. None of your additional business rules can contradict any of the business rules provide in the project definition.

## Denormalization

Denormalization is a conscious, deliberate change of a design from 3rd normal form to some lower normal form in order to meet some objective. To give you some practice at this, I will require that you select some specific place in your model to denormalize. You will describe the denormalization part of phase 1 of the project. Then you will have to explain how you are going to maintain data integrity despite the redundancy that you have introduced into the design by the denormalization. Most often, one or more triggers will synchronize the redundant copies of the data.

Please be sure that your **UML** model depicts a **3rd** Normal Form design. The changes to your design to reflect the denormalization that you have selected will appear in the **relation scheme diagram**.

For the purposes of this project, a denormalization must introduce redundancy of some sort into the physical structure of the dataset. For instance, merging a child table and its parent together and creating a subkey in the resulting table, or creating a multi-valued attribute while maintaining a junction table to represent those values as well would also introduce redundancy into the structure. You will be required to create the necessary triggers to ensure that the redundancies in your structure do not allow conflicting data to be stored.

## Other Triggers

As you probably remember, the UML class model captures semantics that we cannot reflect in the Relation Scheme diagram. The {} constraints (such as {complete, disjoint} on the categorization is one. The requirement that the category classes can never have a multiplicity other than 0..1 on them is another such rule that we can capture explicitly in the UML class model, but there is no way to reflect that in the Relation Scheme diagram.

The physical tables essentially match one for one with the Relation Scheme diagram relation schemes. Those rules that we cannot represent in the Relation Scheme diagram, we cannot represent in the physical tables either. In the past, various teams have written a large number of triggers to enforce the business

rules surrounding the various categorizations in their UML class diagram. While such triggers are a wonderful idea, I will not require that of you in this term project because they represent a good deal of code, they are tedious to write, and I do not think that the amount of learning that you will get from that exercise is worth the amount of time that you would have to put in.

## Project Phases

I have found that students package their deliverables in an apparently endless variety of possible configurations. This makes the grading of these projects much more difficult, particularly if the team makes more than one submission for either of the phases. By giving you the outline of what I expect to see in your drop boxes, I hope that the teams will each structure their deliverables in the same fashion and it will be easy for me to find what I need when it comes time to do the grading.

### Phase 1 – Preliminary conceptual design

Think of this as a "mock" turn in. I will grade your work and provide you feedback as though you received a grade for your work, but it will not contribute to your final grade. Not until you get to Phase 1 Final will I award your team a "real" grade for phase 1. For Phase 1 Preliminary, turn in:

1. The description of your five additional business rules. Please call this BusinessRules.docx, txt, …
2. The explanation of your denormalization and how you are going to enforce data integrity in spite of the redundancy of the denormalized structure. Please call this document Denormalization.docx, .txt, …
3. The normalized UML class diagram – either as a DIA model or draw.io. If you have a different tool that you would like to use, please check with me first.
4. English description of all classes and associations. Please call this ClassAndAssociationDefinitions.txt, or docx, or …
5. English description of all of the attributes. Please call this AttributeDefinitions.txt, or docx, or …

### Phase 1 – Final conceptual design

Turn in everything that you turned in for Phase 1 Preliminary. This time it counts toward your final grade.

### Phase 2 – Physical Design

1. Everything that you turned in for Phase 1.
2. The relation scheme diagram. Please pay attention to the layout of your relation scheme diagram. Please use the same layout in the relation scheme diagram that you did in the UML diagram. If a given class is in the upper left-hand corner of your UML diagram, put the corresponding relation scheme in the same place in the relation scheme diagram. This makes my job enormously easier.

### Phase 3 - Implementation

1. Everything from Phase 2. If you have updated your UML or Relation Scheme diagram since Phase 2, be sure that I get the latest copy of those models.
2. The DDL – This is all of the create table statements. Please put the primary key constraint and the foreign key constraint into the create table statement. Remember to document your DDL so that

it is easy to understand what each table does and the meaning of the columns. Please call this file create_table.sql.

3. The Views – Put the DDL for creating the views into a file called create_view.sql.

    1) MenuItem_v – For each menu item, give it's spiciness, and all of the different costs for that item. If a given item is not on a particular menu, then report "N/A" for that particular item for that particular menu. Also, if an item only appears as a single serving portion, put in "N/A" into the report for the gallon, … prices.

    2) Customer_addresses_v – for each customer, indicate whether they are an individual or a corporate account, and display all of the information that we are managing for that customer.

    3) Sous_mentor_v – reports all the mentor/mentee relationships at Miming's, sorted by the name of the mentor, then the name of the mentee. Show the skill that the mentorship passes, as well as the start date.

    4) Customer_Sales_v – On a year by year basis, show how much each customer has spent at Miming's.

    5) Customer_Value_v – List each customer and the total $ amount of their orders for the past year (365 days), in order of the value of customer orders, from highest to the lowest.

    Please perform a select * from each view and put the results of that select into a single file called view_output.docx, or .pdf. If you like, you can put the SQL for the view in the same file with the output.

4. The DML – The insert statements used to populate the tables. Please remember to put the column names into your insert statements and use just one insert statement for each table. Order your insert statements so that the execution of the inserts meet all of the referential integrity constraints. Remember to document your DML.

5. The queries – In all cases, remember:

    - Each query is a **single** SQL statement.
    - **Never** return just the ID of a given thing in your queries, always do any necessary joins so that you can display a proper name.
    - I will dock points for using literals in your queries. For instance, use the now() function to get the current date when asked to find visits within the past year, do not use a literal and put in the due date of the assignment for the current date.
    - Be sure that the sample data that you insert into your tables is adequate to return **some** data from each of these queries:

        1) List the customers. For each customer, indicate which category he or she fall into, and his or her contact information. If you have more than one independent categorization of

customers, please indicate which category the customer falls into for all of the categorizations.

2) List the top three customers in terms of their net spending for the past **two** years (last 730 days), and the total that they have spent in that period.

3) Find all of the sous chefs who have three or more menu items that they can prepare. For each sous chef, list their name, the number of menu items that they can prepare, and each of the menu items. You can use group_concat to get all of a given sous chef's data on one row, or print out one row per sous chef per menu item.

4) Find all of the sous chefs who have three or more menu items in common.

    i. Please give the name of each of the two sous chefs sharing three or more menu items.

    ii. Please make sure that any given pair of sous chefs only shows up once.

    iii. Please list the items that the two Sous Chefs have in common. Again, you can use group_concat to get all of those items into one value in the output.

5) Find the three menu items most often ordered from the Children's menu and order them from most frequently ordered to least frequently ordered.

6) Show by week, how many hours each employee works.

7) List the customers, sorted by the amount of Miming's Money that they have, from largest to smallest.

8) List the customers and the total that they have spent at Miming's ever, in descending order by the amount that they have spent.

9) Report on the customers at Miming's by the number of **times** that they come in by month and order the report from most frequent to the least frequent. Each row in the output should have the Customer name, the month, the year, and the number of times that customer came in during that month of that year.

10) List the three customers who have spent the most at Miming's over the past year (365 days). Order by the amount that they spent, from largest to smallest.

11) List the five menu items that have generated the most revenue for Miming's over the past year (365 days).

12) Find the sous chef who is mentoring the most other sous chef. List the menu items that the sous chef is passing along to the other sous chefs.

13) Find the three menu items that have the fewest sous chefs skilled in those menu items.

14) List all of the customers who eat at Miming's on their own as well as ordering for their corporation.

15) List the contents and prices of each of the menus.

16) Three additional queries that demonstrate the five additional business rules. Feel free to create additional views to support these queries if you so desire.

6. Sample Output from the Queries – **Be sure to include sample output** in with your queries. The sample output could just be text, or screen captures. Call this file query_output.docx or .pdf. Please include the SQL for the query immediately before the query output. This makes it immensely easier for me to review the query output.

7. Trigger Code – Please provide a listing all of your triggers.

    1) Please make sure that the triggers are adequately documented so that I know what their purpose is, and how they are achieving that purpose.

    2) Sample output showing an attempted insertion/update to the data in which the trigger prevents that wayward update to the database, and another demonstration showing the trigger allowing good data in.

# Hints

## Project Management

This is a big project make no mistake about that.

- Get as much done as you possibly can early on in the time that you have for this project, other classes will be crowding you for time near the end of the semester.
- Make good use of the preliminary phase one turn in. Be as complete in your preliminary design as you can so that I can give you as much feedback as possible.
- It is always possible that my feedback on your preliminary design will cause you to make significant changes to that design. So do not get ahead of the process and start building tables and queries before finalizing the design.
- **Design for the queries**. I have tried to be sure that the first part of the prompt alludes to all of the information that will be in the queries, but take a look at the queries and the views to be sure that your design is not making it unnecessarily difficult to perform the necessary queries or build the necessary views.
- Be **specific** about who on your team performs what and by when. It is even better if you can think of some measurable quality metrics to apply to the deliverables so that everyone on the team can feel secure that each contribution will be of high quality.
- Be sure to test the MySQL environment early:
    - Your individual account
        - You are able to create tables, indexes, and insert data into the tables
        - You are able to create and execute stored procedures, functions and triggers
    - Your team group account

        - You are able to do the same things as the individual accounts
        - That everyone on the team is able to access the team account
    - If you have any issues with the technology, let me know immediately and I will help you work it out.

## Configuration Management

In the business world, there is nothing more deadly to a project than sloppy or spotty configuration management. I will leave it up to the team how they manage the configuration of the various deliverables for this project, but please be certain that the team as a whole is 100% certain that what is going into the project drop box is what the team wants to turn in. I am not going to be sympathetic to the team that tells me the day after a given phase is due that the wrong version of something got turned in by mistake, or that one of the team members turned something in before the rest of the team was entirely ready.

## Technology

Complete this project in MySQL. Nothing else will meet the requirements. Each of you will get an individual MySQL account. Each team gets a group MySQL account as well. All of the team members of a given team will be able to login using their individual account, and then jointly access the team database within MySQL. For that reason, it's important that everyone on the team make sure that you can access the campus MySQL server both through BeachNet+ as well as from home so that you don't all have to be here on campus to work together on this. You will use your individual account for development work. Once you are confident that your work is ready to share, you will run those scripts in the group database so that the rest of the team can benefit from your work. You will **not** be able to grant access to any objects in your personal database to the rest of your team.