Semester Project – Fall 2017
Professor Shoemaker

Instructions

Follow the steps and let me know if you have any questions.

The intent of the project is to create a project you can use in your portfolio for your use in the future.

You can use the **Brenda's Bakeries** problem detailed below or, if you want, create a self-designed project with a database of your choice.

What to use a system of your own?

To use a database of your own choosing you must get my okay before you start.

Send me an email with a paragraph outlining the database with a list of possible tables with their definitions. Your system must have 12-15 tables with at least three many-to-many relationships.

I want to check that the approach your thinking about is neither too small nor too big. If you submit a project other than Brenda's Bakery in Phase 1 without getting my okay first, you'll earn zero points on Phase 1 of the project and you'll be off to very bad start on Phase 2.

The Project has Three Phases

There are three phases to this project and you'll make a separate submission in Blackboard for each one; they each have their own deadlines - see Blackboard for the deadlines.

The phases are patterned after the normal progress of a project: design first followed by construction.

In **Phase 1 - Design** you'll submit your design which will be your EER model along with your commentary and definitions of your entities.

In **Phase 2 – DB Create** you'll submit your updated EER model and the working SQL that creates the database from your EER diagram.

In **Phase 3** – **Working Database** you'll submit your completed database together with a set of queries, stored procedures and triggers.

Phase 1 is worth 40% of the project grade, Phase 2 is worth 20% and Phase 3 is worth 40%. It doesn't matter that these phases may have point values in the assignments different than shown here. The weighted total column in Blackboard scales them properly.

Deadlines and Submitting your Work

For all but the final phase there is a 33% deduction for each whole or part day late. Because I'm under a deadline to get final grades submitted at the end of the semester, I'll start grading the last phase shortly after the deadline. If your submission isn't there when I finish grading everyone's last phase, you'll earn a grade of zero for that phase. Partial work may earn partial credit. If you're not finished by the deadline, I suggest you submit what you've got.

Make sure you use the answer document provided for each phase; don't start your own document from scratch.

For each phase, submit all the files specified. Submit each file separately; **don't combine them in a zip file**. Your document must have an extension of .DOC or .DOCX. Any other extension, including .ODT or .ZIP will result in zero credit for that phase.

TIP: Read all the way through this document before you start. E.g. a query in the last phase that requires an OUTER JOIN will require that you have records setup in your that work properly for the join.

Problem Description – Brenda's Bakeries

You've just started your independent computer consulting business and have landed your first client: Brenda's Bakeries. Brenda Bool started her bakery business 20 years ago out of her home kitchen and has since grown it to five retail shops and a central kitchen were all the baked goods are prepared and delivered to the shops each morning.

Brenda runs an upscale bakery. She uses only the best ingredients she can find and likes to offer unusual pastries that other shops don't offer.

Brenda feels she is beginning to lose track of what's going on in the business and contracted with you to create a database to bring some organization on things.

How the bakery works

The bakery buys **ingredients** from **vendors**. Ingredients are the materials that go into making the baked goods such as flour, salt, yeast, milk, eggs, cinnamon, sugar, etc. The bakery can buy an ingredient from more than one vendor. For example, it buys milk and cream from two different dairies, depending upon which has what's needed on the day of an order what that day's prices are at each dairy.

When an **employee** places an **order** with a vendor it is typically for a number of different ingredients at various prices and **units of measure**. For example, an order to a dairy might have 100 gallons of milk at $2.00/gallon, 5 gallons of heavy whipping cream $3.50/gallon and 10 pounds of butter at $4.25/pound.

The bakery sells **products** in its **shops**. Products include bread, cakes, cupcakes, pies and many types of pastries. Brenda employs many **employees** and some employees work in more than one of her shops.

For each bakery product there is a **recipe** that describes ingredients that go into the product and the quantity and unit of measure of each. For example, the recipe for a particular bread might have 25 pounds of wheat flower, 4 cups of salt, 3 gallons of milk, and a dozen eggs, etc. Note that each ingredient in a recipe has a quantity and unit of measure. (gallons, cups, pounds, etc.)

Since baked goods are perishable, each night the kitchen makes only what will be sold the following day; Brenda does not store inventory of baked goods but she does store ingredients. For example, she might buy all the flour she needs for a week in a single order.

Customers come into the shops and the employees make **sales** of products to the customers. A single sale to a customer can have any number of **products** on it. For example, a dad with three kids comes in and order a muffin for $2.50, two bagels at $1.75 each and a cupcake at $2.00, all on a single order. The system keeps track of which employee made the sale and in which shop the sale was made.

Brenda runs a customer loyalty program that rewards customers for repeated purchases. If customers are willing to give the information, she records their name, email address and phone number. The system records purchases by customers on the loyalty program.

Big Deal

Tips for Avoiding Pitfalls

These are tips on avoiding pitfalls that students in the past have encountered with the project. These tips will save you time, frustration and re-work.

Don’t Delay Getting Started

Do not underestimate the amount of work this project will require. This project is a significant amount of your semester grade and will require far more time than any of the assignments. You have more than enough time, if you don’t put it off. Many students in their final comments said they wished they hadn’t put off getting started on this.

Work for a few of hours many times

You’ll get more done in less time if you break up your work into a series of short work periods over weeks rather than trying to get the project done in one weekend.

Understand the Problem

Many students in their final comments mentioned that if they had read the problem more thoroughly they would have had to do a lot less rework.

The Problem Description section above is your blueprint for the design. Make sure you understand it thoroughly before you start building your database. Think of it this way: A home builder wouldn’t start the digging the foundation of a new home when they’re still not certain if it’s going to be three bedrooms or four.

Be careful not to Gold Plate

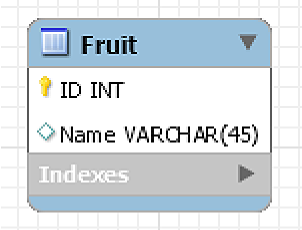
“Gold plating” is when designers and programmers add cool features to a system that were never requested by the business users. Gold plating is a real-world problem because it causes projects to be late and to run over budget. No matter how cool, do not add features not requested in the Problem Description. For example, nothing about scheduling employees is in the Problem Description, so don’t add scheduling to your design. Gold plating will not increase your grade.

Think of it this way: A home builder wouldn't add an extra bathroom to a house that the homeowner didn't request just because the builder thinks it would be cool.

Keep your Design Simple and Relentlessly Consistent

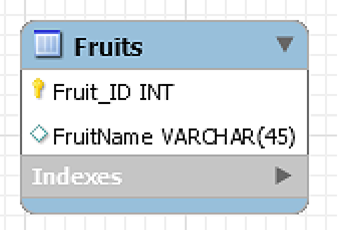
When designing tables, you wind up doing that same sort of thing repeatedly. For example, each table will need a primary key and most tables need to record the name of the things is stores.   It's best to do the same thing the same way, everywhere.   For example, in my own designs, the primary key of a table is always named ID, it's an integer and auto-incremented. Period. And if I need to store the name of the things in the table I always just name the field 'Name.'

This



Fruit
ID INT
Name VARCHAR(45)
Indexes

Not This

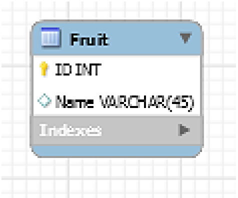


Fruits
Fruit_ID INT
FruitName VARCHAR(45)
Indexes

I've seen many different Customer tables in my career and many variations on customer's names such as: CustomerName, Customer_Name, CustName, CustNam, CName, C_Name, etc., etc. Why not just Name? Another guideline I follow is I almost never use the name of the entity in the name of its fields. E.g. Name not FruitName.

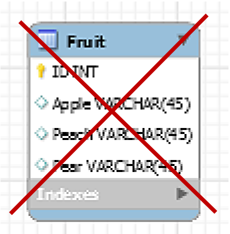
Don't Confuse Records with Table Fields

Suppose in some system you needed a Fruit table meant to keep track of Apples, Peaches and Pears and possibly other types of fruit. You design a table with a name field and would put three records in the table, one for each of the fruits. Your table design would look like this. When new types of fruit need to be added, you just add new records in the table.



Fruit
ID INT
Name VARCHAR(45)
Indexes

You wouldn't put three fields like this - you'd need to change the design every time a new fruit was added!



Fruit
ID INT
Apple VARCHAR(45)
Peach VARCHAR(45)
Pear VARCHAR(45)
Indexes

Deleting Entities from your Model

If you decide to delete an entity from your model, it's not enough to just remove it from the diagram. You must actually delete it from the model.   To delete an entity, right click on the entity in the diagram and select Delete. If you remove by pressing the Delete keyboard button and don't actually delete it, it will still be there and will be generated into a table when you forward engineer your model.

Watch out for Phase 3

Phase 3 requires a lot of SQL coding and that can be time-consuming. Start early!

Getting Help

Please always feel free to email me questions about the project as you work on it. You should always attach your model file and/or SQL to help me understand your question.

I'm always happy to answer your questions. If you ask a specific question like "Do I have the many-to-many relationship between A and B right?" I will take a close look at that issue.

If you ask me a general question such as "Is my diagram in good shape?" I'll be glad to look it over for obvious problems, but since I don't "pre-grade" work, I won't go through a preliminary version of a diagram with the attention to detail that I do when grading. It's your responsibility to go through the details of your work before you submit it.

Please remember that I may take up to 24 hours to respond to a question, so don't wait until the evening of a deadline.

Steps for Phase 1 - Design

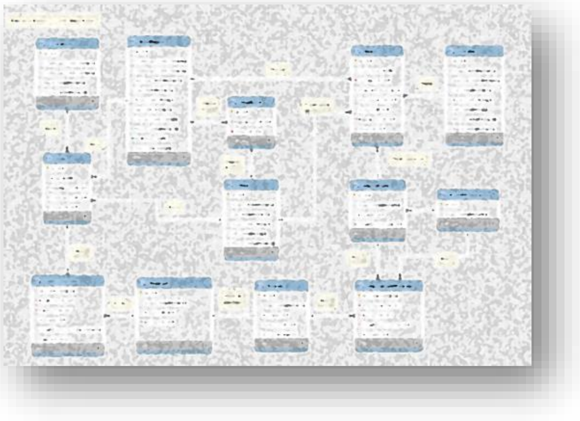
For the written parts, use good grammar, spelling, capitalization, punctuation and write in complete sentences. You must use the answer document from Blackboard. Any other document results in zero credit.

Step	Points	Instructions
1.	10	Develop an executive summary for your project. This is a business, not a technical statement. Imagine you are writing this for a Brenda who is not technical at all, doesn't want to know about tech stuff, and will look at you like you're from Mars if you start tossing technical terms around. Include your topic (Brenda's or your own) as well as scope. Scope is extremely important in projects. When addressing scope be sure to explain what your project is going to include and, equally as important, what it is not going to include. When you're writing a summary like in this the workplace you want to be sure to highlight the benefits the system will bring the business.

This summary should 150-200 words and should be broken up into at least three or four paragraphs – don’t write one long run-on paragraph.

Which you rather have to read?

This	Or This
Develop an executive summary for your project. This is a business, not a technical statement. Imagine you are writing this for a Brenda who is not technical at all, doesn’t want to know about tech stuff, and will look at you like you’re from Mars if you start tossing technical terms around. Include your topic (Brenda’s or your own) as well as scope. Scope is extremely important in projects. When addressing scope be sure to explain what your project is going to include and, equally as important, what it is not going to include. This summary should 150-200 words and should be broken up into two or three paragraphs – don’t write one long run-on paragraph. When you’re writing a summary like in this the workplace you want to be sure to highlight the benefits the system will bring the business.	Develop an executive summary for your project. This is a business, not a technical statement. Imagine you are writing this for a Brenda who is not technical at all, doesn’t want to know about tech stuff, and will look at you like you’re from Mars if you start tossing technical terms around. Include your topic (Brenda’s or your own) as well as scope. Scope is extremely important in projects. When addressing scope be sure to explain what your project is going to include and, equally as important, what it is not going to include. This summary should 150-200 words and should be broken up into two or three paragraphs – don’t write one long run-on paragraph. When you’re writing a summary like in this the workplace you want to be sure to highlight the benefits the system will bring the business.

2.	30	Create an **EER diagram** for Brenda's in MySQL Workbench. A couple important points on your diagram: • The entities will come from the **nouns** in the description above. For example, **customer** is a noun and will wind up being an entity in your diagram. Be sure to include primary and foreign keys. Add a textbox by each relationship line with a verb or verb phrase that titles the relationship. An example of a verb phrase might be: An employee **works in** a store. • All entities must be in <u>third normal form</u> or higher. • Every entity should be in a relationship with at least one other entity and you should be able to traverse relationships from any entity to every other entity on the diagram. I.e. No entity or group of entities should be an "island" on the diagram, separate from other entities. [-5] • Neatness Counts ▪ Add a text box in the upper left of your diagram with the name of the project and your name. [-3] ▪ Orient the diagram in landscape. (File menu, Page Setup) [-3] ▪ Arrange your diagram to fit on one page – you can see the page edges in the Workbench diagram screen. [-3] ▪ <u>No line should cross another line or entity</u> as that makes the diagram very hard to understand. [-10] • My design for Brenda's has 13 tables. Four of the tables are needed only because of many-to-many relationships. The diagram fits comfortably on one page in landscape orientation. I've obscured it so I don't spoil all your fun.  • Paste a screen shot of your diagram in your answer document.

3.	20	Write up the **Crayon Definitions** for each entity on your diagram. A crayon definition is simply a definition that's short enough to be written with a crayon. The idea is to quickly give others a sense of why you have the entity in your design. It should describe the records you'll hold in the table. An example might be "The Vehicle table contains records for both cars, trucks and rickshaws."
4.	10	Develop a set of **business rules** to support your executive summary. I would like you to have at least seven rules that directly address the data (things like an employee works for only one department) that will be converted to constraints. Don't write something trivial like "An Employee's name can be up to 45 characters." Or "A customer can have one email address." Be careful not to cook up requirements that create strange restrictions on what users can do in the system. For example, one student made this rule: "While Brenda can buy the same product from multiple vendors, she will only buy a product once a day at most." There's no purpose in this restriction and the system shouldn't have it.
5.	10	Write at least five **output requirements** - what kind of data output is necessary such as: show a list of customers with the names of sales reps. Think of reports
	80	Total

Submitting this Phase

When you submit Phase 1, please submit these files in this order. If you don't submit your document first, I may not be able to see it in Blackboard. Submit the files individually, don't zip them up.

1. Document file (.DOC or .DOCX only)
2. Your model file. (Be careful not accidently upload the backup file of the model. Double check that the file you're uploading doesn't end in .bak.)

Steps for Phase 2 – Database Create

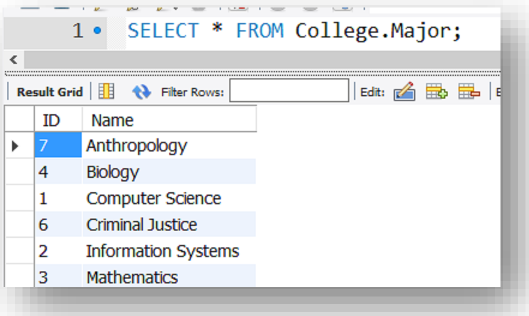
Don't start this until you get my feedback on your Phase 1 submission. You may wind up having to make significant changes in your ERR diagram as a result of my feedback. If I make suggestions for changes in your Phase 1 submission, you must incorporate them in Phase 2.

In this phase you'll forward engineer your design into a database with all the tables. You're interested only in getting your tables created and putting just a few records in each table. Do not create more records than you're asked.

For the written parts, use good grammar, spelling, punctuation and write in complete sentences.

You must use the answer document from Blackboard. Any other document results in zero credit.

Step	Points	Instructions
1.	0	Paste in a screenshot of your EER diagram from Phase 1 with any markings I made on it and below the diagram also paste in any written comments I made. I want the diagram and comments in this document to refresh my memory of what you're doing. (10-point deduction if missing)
2.	15	Paste in a screenshot of your EER diagram with any changes as a result of my comments on Phase 1 or any changes you made as a result of the later steps in this phase. If there are no changes since Phase 1 just paste it in again.
3.	5	Enter your crayon definitions for your entities with any changes as a result your changes to the diagram from Phase 1. This is also to refresh my memory.
4.	10	Forward Engineer your MySQL Workbench EER diagram to obtain the SQL to create your database. Name your database BrendasXXX where XXX is your initials. Run the SQL to create your database. Include a screen shot of the tree view of database structure with <u>all the tables columns expanded</u> so I can see all the columns in every table.
5.	20	Create a SQL script file by hand to insert two or three records in each table. (And no more than two or three records!) You may find the database creation scripts from Lesson 2 helpful in getting going on this. This step is important because adding records to tables for the first time sometimes uncovers problems in your design. The limit on the number of records is meant to keep you have having to discard a lot of work if changes in the design are needed. Attach the file of INSERT statements in Blackboard. Plan on having to go back to your EER diagram to make changes and forward engineering again, maybe more than once. Tip: Insert records in tables in the same order that MySQL has the tables in the forward engineer script file. This will avoid problems with referential integrity. In the answer document, paste in a screen shot of the Result Grid showing the results of a SELECT * for each table the table. To save yourself time, you don't need to paste in the SELECT statement as text, instead just show both SELECT statement and the Result Grid as a single screen shot like this:

		 1 SELECT * FROM College.Major; ID Name 7 Anthropology 4 Biology 1 Computer Science 6 Criminal Justice 2 Information Systems 3 Mathematics
6.	5	Describe in a couple sentences or fewer, any change you may needed to make in your design as a result of inserting records in each table.
	55	Total

Submitting this Phase

When you submit this phase, please submit these files in this order. If you don't submit your document first, I may not be able to see it in Blackboard. Submit the files individually, don't zip them up.

1. Document file. (.DOC or .DOCX only)
2. SQL script file that creates the database and tables. (That resulted from the forward engineer)
3. The SQL script file that inserts two or three records in each table.
4. Your model file.

Steps for Phase 3 – Working Database

Don't start this until you get my feedback on your Phase 2 submission.

For the written parts, use good grammar, spelling, punctuation and write in complete sentences. You must use the answer document from Blackboard. Any other document results in zero credit.

Step	Points	Instructions
1.	10	Paste in a **screenshot** of your final EER diagram with any changes since the last phase.
2.	5	Paste in your **crayon definitions** from Phase 2 for your entities with any changes you've decided since then.

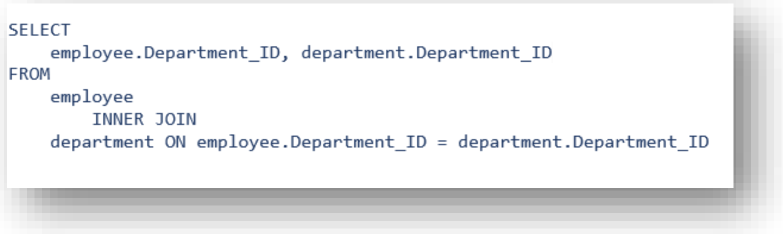
3.	15	Create a new script file to **INSERT records** into your tables. You also need to use the free site www.**mockaroo.com** to generate INSERT statements for at least your Customer table. See the section at the end of the document for more on mockaroo. You must use mockaroo.com to generate 100 INSERT statements for your Customer table. You can also use mockaroo for any of the other tables if you want, but for some of the smaller tables, like Ingredient, it probably is quickest to just create the insert statements by hand. You need to have at least 10 records in all the tables other than Customer. You'll paste a screen shot of the mockaroo page with your Customer table setup. Put all the INSERT statements for all of your tables in a single SQL file and attach it with your Blackboard submission. Tip: In this SQL file, order the tables in the same order the tables were created in your forward engineer SQL. For each table, execute a SELECT * on the table and paste in the SQL statement followed by the Result Grid in the answer document so I can see the name of the table in the SELECT statement and what your data looks like in the table. Snip just the first 10 or so records from each table.

Queries

Create the following types of queries for your database. For each query, provide a one or two sentence explanation of the business purpose of the query, the SQL and a screenshot of the Result Grid.

> Important: In your explanation tell me what the query is for – it's business purpose. Don't tell me how it works. Don't write "I joined A to B." I can see that by looking at the SQL. Instead write something like "This query gives a list of all employees who are currently on probation." Be sure to beautify each query and to snip just the result grid in your screen shot.

Your queries must have some non-trivial and useful business purpose. Here's a query that a student submitted in an earlier semester. It has no business purpose and would earn no credit.



```
SELECT
    employee.Department_ID, department.Department_ID
FROM
    employee
        INNER JOIN
    department ON employee.Department_ID = department.Department_ID
```

Every query must produce at least one record. I.e. don't write a query that shows sales for a customer who doesn't have any sales. Don't use SELECT * in any of these queries. Sort every query.

Each of these queries is a SELECT statement containing…		
4.	5	A **WHERE** clause.
5.	5	An **AND** or **OR** in the **WHERE** clause. Don’t use an OR where IN would work (~~State=’OH’ OR state=’PA’~~), and don’t use an AND where BETWEEN would work (~~date >= '2015-12-01' AND date <= '2015-12-31'~~).
6.	10	A **GROUP BY** with at least two of these: SUM, COUNT, MIN or MAX. Include a join, if needed to show the name, not the id, of whatever you’re grouping on. For example, if you’re joining in the Animal table, group on the Name field from the Animal table, not AnimalID. This query must include the HAVING clause.
7.	5	An **INNER JOIN** of at least three tables. The output needs to have at least one field from each table joined in.
8.	10	An **OUTER JOIN**. This query must result in at least one record that would not have appeared if the query used an INNER join. In the result grid highlight such a record when you take your screen shot. Note: You must have a mix of records – some that would and some that would not appear in an INNER join.
9.	10	A **Roll Your Own query** of your own design. This query should join at least four tables, have a GROUP BY, and have a meaningful business purpose.

View

10.	**10**	Create a View with a useful business purpose. Views are often used to supply data in reports, so think of a report and the columns on it and create a view to provide that data. Give your columns sensible headers. I don’t want to see a trivial query in the view. E.g. one that just selects * from some table or has only one table, and don’t use one of the queries you created in the previous steps. Include a one or two sentence explanation of the purpose of the view (tell me what it’s for – it’s business purpose, not how it works), the SQL you used to create the view and a screenshot of the Result Grid from the execution of the view.

Stored Procedure

11.	**10**	Create a Stored Procedure (not a function) that takes at least one IN argument and returns at least one OUT argument. The Stored Procedure should use the in argument to filter a SQL statement that calculates a sum, average or count that it returns in the out argument. For example, is a car dealership you might pass in the id of a model in the in argument and the Stored Procedure could count the number of cars of that model in the dealership and return the count in the out argument.

Triggers

Create a set of three triggers that helps insure data integrity in one of these tables: Customer, Vendor, or Employee (your choice). Include a one or two sentence explanation of the purpose of each trigger, the SQL you used to create the triggers and a screenshot of a Result Grid and/or Output window that shows the effect of each trigger. For example, if your trigger can throw an error you need to show a SQL statement that causes the error and the error itself in the Output window. If your trigger writes a log/audit record you need to show the record it wrote in the log table.

I expect these triggers to have a useful business purpose and some complexity. For example, don't submit a one-line trigger that just translates some field into upper case and don't reuse a trigger from the homework.

12.	10	**BEFORE INSERT**. This must be a BEFORE trigger that prepares data to make sure it complies to rules. Don't look for bad data and throw an error when you find, that's the next trigger. In other words, if a piece of information that's being inserted isn't correct, fix it in your trigger.
13.	10	**BEFORE UPDATE**. The UPDATE trigger must be a BEFORE trigger that tests data quality and throws an error when bad data comes in.
14.	10	**AFTER DELETE**. This needs to be an AFTER trigger that writes a record into a log/audit table.

Write Up

15.	5	Write a few sentences to answer these two questions: What is the most important thing you personally learned in this project? If you had to do this over again, what would you do differently?
	130	Total

Submitting this Phase

When you submit Phase 3, please submit these files in this order. If you don't submit your document first, I may not be able to see it in Blackboard. Submit the files individually, don't zip them up.

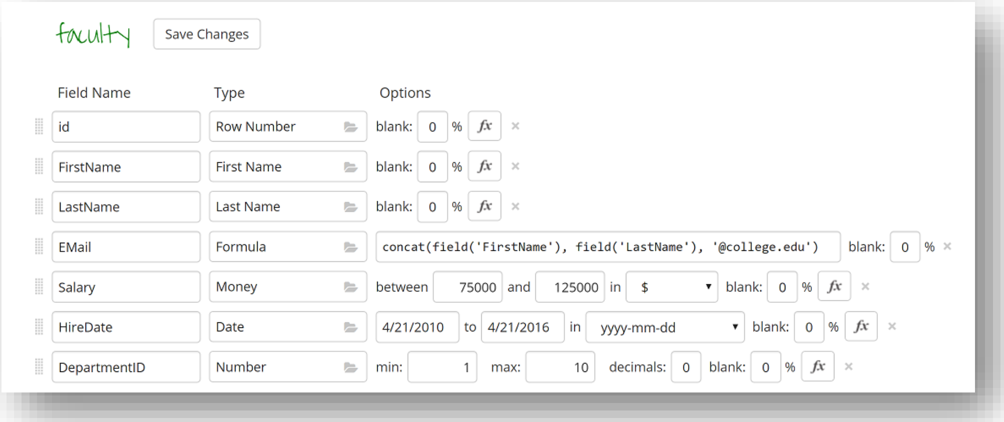
1. Document file (.DOC or .DOCX only)
2. SQL Script file that loaded records into all your tables.
3. Your model file

Background on Mockaroo

www.mockaroo.com is a free website that you can use to generate SQL INSERT statements for your tables, complete with random data. It's easy to learn to use and will save you a great deal of time.

If you create a free account, you can save your table definitions so that you can come back to make changes to the definitions and quickly regenerate data.

I used mockaroo to generate all the data in the College database. For example, here's my mockaroo setup for the Faculty table. Notice how I generated random dates for the HireDate and a random number between 1 and 10 for the DepartmentID.



faculty

Field Name	Type	Options
id	Row Number	blank: 0 % *fx* ×
FirstName	First Name	blank: 0 % *fx* ×
LastName	Last Name	blank: 0 % *fx* ×
EMail	Formula	`concat(field('FirstName'), field('LastName'), '@college.edu')` blank: 0 % ×
Salary	Money	between 75000 and 125000 in $ ▾ blank: 0 % *fx* ×
HireDate	Date	4/21/2010 to 4/21/2016 in yyyy-mm-dd ▾ blank: 0 % *fx* ×
DepartmentID	Number	min: 1 max: 10 decimals: 0 blank: 0 % *fx* ×