CS 325 - Project Handout - Fall 2016

When studying the design and implementation of databases, it greatly aids understanding to actually experience the process being studied. To that end, during this semester you will be modeling, designing, and implementing a database for a scenario. (Think of it as modeling, designing, and implementing a demonstrable prototype of a database for that scenario.)

This project will use the Oracle RDBMS (Relational Database Management System) on campus. The project consists of a number of milestones, with pieces turned in for each; its final milestone must be turned in by:

11:59 pm on **Friday, December 9th**

Remember -- like all course work, it will not be accepted late. However, IF:

- you present your project in lab on **Wednesday, December 7th**,

  AND!!

- you submit the *final* milestone before **11:59 pm** on **Thursday, December 8th**,

  ...it will be considered **early**, and will receive **a 5 point BONUS** added to the overall project grade.

(Note that if you don't present your project in lab on December 7th, you CANNOT receive this bonus.)

Project Grading Comments

Your final project should be robust enough to **demonstrate** as a **prototype** -- do not attempt to implement a "production quality" system! (And if you cannot implement everything you hoped to, implement as much as you can, aiming for an **interesting, demonstrable prototype**.) A project that meets all of the minimum requirements mentioned in this handout, and that has met all of the minimum requirements all down the line (throughout the milestones), and that does so well, would receive a grade of **90**.

The other **10** points I will reward based on merit -- did you come up with a particularly interesting, original proposal? is a database particularly well-designed and implemented? Is something about a project above-and-beyond the minimal requirements, or exemplary? Were useful extra features included, or was some major aspect particularly well-done? Did something about a project just stand out, or make a strong impression?

Please note, also, that you may receive grading comments for a milestone. These comments may include corrections or additions that must be made to your project. **If these are not done, subsequent milestones' grades may be affected.**

Milestone grading breakdowns:

- the overall project grade -- the sum of the parts below -- makes up **20%** of the final course grade, as noted in the syllabus. (this sum is multiplied by 0.20 to compute the project portion of your semester course grade.)
- the overall project grade is made up of:

Project Milestone	Milestone's Worth	Additional Information
Scenario Selection	up to 2.5 points	
Project Model	up to 20 points	up to 3 points for sub-milestone 1, up to 5 points for sub-milestone 2, and up to 12 points for the final model milestone
Project Design	up to 15 points	

Project Milestone	Milestone's Worth	Additional Information
Project Population	up to 7.5 points	
Project Presentation	up to 5 points	
Project Final Milestone	up to 50 points	up to 40 pts baseline, and (see above) up to 10 points for doing more than the minimum

Project Scenario Selection (worth: up to 2.5 points)

One important aspect of this project is that you are not just "building a database" -- you are modeling a **scenario** in which a database could be useful. This distinction is important, and needs to be kept in mind, starting at this Project Scenario Selection phase. You need a scenario with multiple players interacting in various ways.

As I write this handout, four project scenarios are available for this semester. The current versions of these are posted along with this project handout. You will select one of these, unless you discuss this with me otherwise before the beginning of lecture on **Thursday, September 29**.

NOTICE that several of these have a **Variations allowed:** section -- these are variations you are allowed to make to the scenario if you would like. See below for how to indicate these.

Project Scenario Selection milestone (up to 2.5 points)

Due:

by beginning of lecture on **Thursday, September 29**

How to submit:

Submit the file `325scenario.txt` using the tool `~st10/325submit` and using a homework number of **21**.

What to submit:

In a file `325scenario.txt`:

- include your name and the last-modified date
- give your scenario selection as follows:
  - IF you are simply selecting one of the four scenarios with no modification, give the name of your chosen scenario.
  - IF you are choosing to make one or more of the variations permitted for one of the scenarios, you should give the name of the scenario you are "starting" from, then give the name of your variation of that scenario, followed by a rewritten version of that scenario now including your variation.
- Consider -- what are some QUESTIONS people might ask within your chosen scenario? (For example, for the little rental-videos scenario our homework tables are based on, some example questions might be:
  - Does customer X currently have any unreturned rentals?
  - In what formats is movie Y available for rental?

  - What are the titles of Comedy movies for which we have copies?

  Come up with at least 5 such example questions for your chosen scenario, and include a numbered list of these questions.

- Also come up with at least 3 initial business rules for your scenario. (For example, for the little rental-videos scenario our homework tables are based on, some example business rules might be:
  - While a store manager may authorize a different rental return date than the usual, all rentals must have a return date specified at the time of the rental.
  - A customer with an overdue rental may not rent any more videos until taking care of that overdue rental.
  - A customer may indicate a favorite movies category, but they may only indicate one such favorite category.

  Come up with at least 3 initial reasonable business rules for your chosen scenario, and include a numbered list of these initial business rules. (Note that you will be expanding this in a separate file as the project continues.)

Project Model Milestones (worth: up to 3 points + up to 5 points + up to 12 points = up to 20 points

NOTE: you should **not** be mentioning tables or relations at this point **at all**, nor should you be mentioning primary or foreign keys. Those are appropriate in the next milestone!

Project Model sub-milestone 1 (up to 3 points)

Due:

by beginning of lecture on **Thursday, September 29**

How to submit:

Bring THREE copies of the items given below to LECTURE on **Thursday, September 29**.

What to submit:

Bring THREE COPIES, on paper, of the following to LECTURE on **Thursday, September 29** -- each copy should contain:

- your name
- the name of your scenario
  - IF you are using a variation, ALSO include your scenario description
- a list of the entity classes you believe should be part of your database model at this point.
- an example of a 1:N relationship class you believe should be part of your model -- for this sub-milestone, simply give its name AND the names of the two entity classes it is between
- an example of a M:N relationship class you believe should be part of your model -- for this sub-milestone, simply give its name AND the names of the two entity classes it is between
- an example of a multi-valued attribute, and the entity class it is an attribute of, that you believe should be part of your model

(One of these copies will be given to me, to check-off whether you have included attempts at each of the required pieces; the other two copies will be used for part of an in-lecture activity.)

Project Model sub-milestone 2 (up to 5 points)

Due:

by beginning of YOUR lab session on **Wednesday, October 5**

How to submit:

Bring THREE copies of the items given below to LAB on **Wednesday, October 5**.

What to submit:

Bring THREE COPIES, on paper, of the following to LAB on **Wednesday, October 5** -- each copy should contain:

- your name
- the name of your scenario
  - IF you are using a variation, ALSO include your scenario description
- your current-as-of-this-point Entity Relationship Diagram (ERD) for your project, depicting your database model
- make sure this includes the lists of attributes for each entity class, in which multi-valued attributes are indicated, as discussed in lecture
  - Remember: since these are not tables or relations, there are no primary keys or foreign keys yet. These are NOT part of the E-R model stage.
- also include the current version of your business rules

(One of these copies will be given to me, to check-off whether you have included attempts at each of the required pieces; the other two copies will be used for part of in-lab model draft reviews.)

Project Model final milestone (up to 12 points)

Due:

by **11:59 pm** on **Friday, October 21st**

How to submit:

Submit the files noted below using `~st10/325submit` and using a homework number of **22**.

What to submit:

Create a PDF file named `325model.pdf` containing the following:

- your name(s)
- CS 325 - Fall 2016
- the date the model was last modified
- your current-as-of-this-point Entity Relationship Diagram (ERD) for your project, depicting your database model
- make sure this includes the lists of attributes for each entity class, in which multi-valued attributes are indicated, as discussed in lecture
- remember, entity classes are NOT tables or relations. Do NOT refer to them as such, nor include information about tables or relations in the model. The project is NOT at the table/relation stage yet!
  - Likewise, since these are not tables or relations, there are no primary keys or foreign keys yet, either. These are NOT part of the E-R model stage.

- Make sure that your completed model meets the model aspects of the **Project minimum structural requirements** given below.

Also submit the latest version of your business rules, in a separate file now named `325biz-rules.pdf`

- (Hint: you SHOULD find yourself adding to your initial set of business rules as part of the modeling process!)
- Be sure that it includes an appropriate last-modified date.

Only submit your `325scenario.txt` file if you have made any corrections or changes to your project scenario.

Because an important topic in this course is database modeling and design, your project is required to meet certain criteria to help ensure that it is at least somewhat interesting in model and structural senses. **Final projects that do not meet these minimum criteria will be severely penalized.**

The **database model** for your database must include:

- at least 5 **distinct**, **significant** entity classes
  - a superclass entity along with all of its subclass entities count as **one** combined entity toward the five-entity-class minimum, unless some of the subtypes have relationships in which only that subtype participates. Be careful, and **ask me** if you have concerns about this.
- at least 4 **distinct**, **significant** relationship classes, at least one that is **1:N** and at least one that is **M:N**
- at least one **multi-valued** attribute

And, the corresponding **database design/schema** must eventually **correctly implement** all of the above.

Project Design Milestone (worth: up to 15 points)

Due:

by **11:59 pm** on **Sunday, November 13th**

How to submit:

Submit the files noted below using `~st10/325submit` and using a homework number of **23**.

What to submit:

Create a SQL script named `325design.sql` containing the following:

- your name(s) (in a comment)
- CS 325 - Fall 2016 (in a comment)
- the date the design was last modified (in a comment)
- your current-as-of-this-point database schema/design, in which the structure of your database's tables are expressed as **nicely-formatted SQL** `create table` **statements**.
  - precede each SQL `create table` statement with a **neat comment** describing the table's **purpose**, explaining any attribute whose meaning is not immediately clear from its name, and elaborating on the domain of any attribute whose logical domain needs more description than is apparent from its physical domain
  - (for convenience later during population, go ahead and also precede each SQL `create table` statement with a corresponding `drop table` statement as well)
  - remember: primary keys must be explicitly specified for each table, and all foreign keys necessary should be explicitly specified as well
  - (thus, these SQL `create table` statements give us 3 of the 4 components of a database design/schema: the tables' structure, their relationships, and at least some indication of each attribute's physical domain, if not its logical domain)
- What about the 4th component of a database design/schema? That's the business rules, and so...

Also submit the latest version of your business rules, `325biz-rules.pdf`:

- **Yes**, submit it **even if it has not changed**. (But, hint: you SHOULD find yourself adding to your business rules as part of developing the design/schema!) Also, business rules are officially part of a database design/schema, as you should recall from earlier this semester.
- Be sure that, as always, it includes an appropriate last-modified date.

Also submit the latest version of your model, `325model.pdf`:

- **Yes**, submit it **even if it has not changed**. The design/schema should be developed directly from the model, and so one cannot really evaluate the design/schema independent of that model.
- **Make sure** that the model you submit corresponds to your submitted design/schema! Your design/schema will be graded on the basis of the model version that you submit along with it.

  - This does **NOT** mean to add table-related aspects to your model! It means, if you change any maximum or minimum cardinalities, for example, or add or remove any entity classes, etc., based either on comments from me or from insights you get during the design process, that you modify the model to reflect them.

- Be sure that, as always, the submitted model includes an appropriate last-modified date

Additional notes/reminders:

- **PLEASE NOTE**: `insert` statements **DO NOT** belong with the database design. Please save them for the population milestone, and do **NOT** include them here.
- Remember, if previous milestone(s) required that certain changes or additions be made to the proposal, to the previous set of business rules, and/or to the model, then those changes should be reflected in newly-submitted versions for this milestone, or **this** milestone's grade may be affected

Project Population Milestone (worth: up to 7.5 points)

Due:

by **11:59 pm** on **Tuesday, November 29th**

How to submit:

Submit the files noted below using `~st10/325submit` and using a homework number of **24**.

What to submit:

Create and submit a `325populate.sql` file (that is, a SQL script) that populates your tables with some example data (which can be fictitious!)

*******THIS MUST BE a SEPARATE FILE FROM YOUR** `325design.sql` file!!!!!**********

- include your name(s) in a comment in the SQL script
- include `CS 325 - Fall 2016` in a comment in the SQL script
- include the date that the SQL script was last modified in a comment within the SQL script
- how much data should you include? The goal is to include enough to make your project at least a **non-trivial, demonstrable prototype**. Trying to give some firmer guidelines:
  - have at least 10 rows per table,
  - ...with additional rows as needed to make your database a reasonable prototype. (You'll find that some tables need additional rows so that others can have sufficient contents.)
  - Again, fictitious data is fine, but make it "look" realistic -- don't use names like 'fg^s&A#', etc.
  - Don't use "real" data that is sensitive! (No "real" social security numbers, credit card numbers, etc.!)

Also create and submit a SQL script `325show-contents.sql` that contains:

- your name(s) (in a comment)
- CS 325 - Fall 2016 (in a comment)
- the date the contents script was last modified (in a comment)
- `spool` commands to output its results to a file `325result-contents.txt`
- one or more `select *` statements for each table
  - precede each `select` statement with a `prompt` command to display the name of the table whose rows are being displayed
  - make sure that the rows are readable, and do not "wrap-around" in the `325result-contents.txt` file. There are several ways to achieve this; for example:
  - you can show a very-wide table's contents using several `select` statements, each projecting the primary key and a few of its other attributes;
  - you can use the `truncate` feature of the SQL*Plus `column` command (ask me if you need help with this)

   - you can increase the number of characters printed per line before wrapping by changing `linesize`

Also submit the file `325result-contents.txt` resulting from running `325show-contents.sql`

Also submit the latest version of your business rules, `325biz-rules.pdf`

- **Yes**, submit it **even if it has not changed**. (you MAY find yourself adding to your business rules as part of the population process)
- Be sure, as always, that it includes an appropriate last-modified date

Only submit your `325design.sql` file if it has been changed since the database design/schema milestone.

Only submit your `325model.pdf` file if is has been changed since the database design/schema milestone.

- (make sure that your latest model corresponds with your latest database design/schema!)

Remember: if previous milestone(s) required that certain changes or additions be made to the proposal, to the previous set of business rules, to the model, and/or to the design/schema, then those changes should be reflected in newly-submitted versions for this milestone, or **this** milestone's grade may be affected

Project presentation Milestone (worth: up to 5 points)

Due:

by beginning of YOUR lab session on **Wednesday, December 7th**

How to submit:

Present the following during lab on Wednesday, December 7th.

What to present:

To provide you with some practice preparing and giving a short live-program-demonstration related to your semester database project, and to give you a little extra encouragement as you polish the final project milestone's example queries and example reports, you are to prepare and present a short presentation meeting the following requirements.

Note that this presentation will also be the Week 15 Lab Exercise.

Remember, also, that doing this presentation is required as part of the early-completion 5-point bonus.

Your presentation must meet the following requirements:

- You should plan for it to be **no more than 3 minutes** in length.
- In your presentation:
  - start by **briefly** describing your scenario
  - then include a **brief** display of your final model's E-R diagram
  - then run, **live**, one of your example queries that you believe would be helpful to people within that scenario
    - first **display the query already typed within a SQL script**, and then **run that SQL script live** to show its result
    - also **say why you think this query would be useful** to people within that scenario
  - then run, **live**, one of your example reports that you believe would be helpful to people within that scenario
    - first **display the report already typed within a SQL script**, and then **run that SQL script live** to show its result
    - also **say why you think this report would be useful** to people within that scenario
- Just to make sure this is clear: this must be a **live-program-demonstration** -- you are expected to **actually run** your selected example query script and your selected example report script as part of your demonstration, **projecting** your scripts' output.

Project Final Milestone
(worth: up to 40 points baseline + up to 10 points for work above and beyond = up to 50 points)

Due:

by **11:59 pm** on **Friday, December 9th**

(**But remember**: the **early completion 5 POINT BONUS** applies to projects presented in lab on **Wednesday, December 7th** AND then completely submitted by **11:59 pm** on **Thursday, December 8th**)

How to submit:

Submit the files noted below using `~st10/325submit` and using a homework number of **25**.

What to submit:

Create and submit a `325queries.sql` file that contains a set of **example queries** using your prototype database, including:

- your name(s) (in a comment)
- CS 325 - Fall 2016 (in a comment)
- the date the example queries script was last modified (in a comment)
- `spool` commands to output its results to a file `325query-results.txt`
- at least **eight** substantially-different and structurally-different representative queries, including at least some "innovative" ones, that meet the following requirements:
  – each query must be potentially meaningful/useful to users of your database
  – precede each query statement with a `prompt` command explaining the query's purpose and **numbering** the query
  – include at least one **join**
  – include at least one **appropriately-nested** query
  – include at least one appropriate use of an **aggregate function** (such as `count, min, max, avg, sum,` etc.)
  – include at least one appropriate use of a `group by` clause
  – include at least one **compound** `where` condition with at least a couple of sub-conditions **other** than join conditions
  – make sure that enough example data is included so that these queries' results are a meaningful demonstration -- you may need to add additional rows for this to be the case.
- These queries could be related to the questions given in your database proposal, but they do not have to be, and you are not limited to answering those original questions.
- The **most important** criterion is that they show **clearly** how the database could be **useful** to end-users within

the scenario.

Also submit the file `325query-results.txt` resulting from running `325queries.sql`

Create and submit one or more files `325report1.sql`, `325report2.sql`, ... that create **example reports** from your prototype database, including:

- your name(s) (in a comment)
- CS 325 - Fall 2016 (in a comment)
- the date that example reports script was last modified (in a comment)
- `spool` commands to output that script's results to a file `325report1-results.txt` (or `325report2-results.txt`, etc.)
- at least three substantially-different and structurally-different representative reports, including at least some "innovative" ones, that meet the following requirements:
  - each report must be potentially meaningful/useful to users of your database
  - the code for each report must be preceded by a neat **comment** explaining its purpose
  - each report should be **well-designed** and **well-laid-out**. Human-readability is an important characteristic of a well-designed report.
  - "nice"/"pretty" column formatting -- especially for numeric columns -- is expected
  - "nice"/"pretty" heading formatting is expected
  - concatenation should be used to make reports more pleasant to read -- (for example, instead of having separate first and last name columns, a report should display a single column with the last name concatenated with the first name, or vice versa, depending on the report's purpose)
  - rows should be explicitly ordered in a meaningful way within reports (and this includes appropriate secondary ordering as applicable, etc.)
  - at least a top title is expected (for **each** report)
  - include at least one appropriate `break` command whose results are obvious
  - include at least one appropriate `compute` command whose results are obvious
  - at least two of your reports should be based on queries involving more than one table (or on a view created from more than one table)
  - at least one report should contain at least one column whose contents are appropriate, meaningful numeric data with a well-formatted **fractional** part
  - `break` should be used to avoid "ugly" repetition in consecutive report rows
  - `skip` should be used judiciously to separate results generated using breaks and computes. (Avoid too much skipping of lines, however -- e.g., avoid having a blank line after every row in a report.)
  - make sure that enough example data is included so that these reports' results are a meaningful demonstration -- you may need to add additional rows for this to be the case.
- These reports could be related to the questions given in your database proposal, but they do not have to be, and you are not limited to reports that answer those original questions.
- The **most important** criterion is that they show **clearly** how the database could be **useful** to end-users within the scenario.

Also submit the files `325report1-results.txt`, `325report2-results.txt`, ... resulting from running `325report1.sql`, `325report2.sql`, ...

Create and submit a file `325discussion1.pdf` that contains:

- your name(s)
- CS 325 - Fall 2016
- the date this first discussion was last modified
- a discussion on "**How can this implemented database now be used?**" that meets the following requirements:
  - it should contain **at least 300 words** (I will measure this using the UNIX `wc` command.)
  - discuss how your particular database, now implemented, can be used within your proposal's scenario
  - be specific!

Create and submit a file `325discussion2.pdf` that contains:

- your name(s)
- CS 325 - Fall 2016
- the date this second discussion was last modified
- a discussion on "**How can this implemented database now be maintained?**" that meets the following requirements:
  - it should contain **at least 300 words** (I will measure this using the UNIX `wc` command.)
  - discuss how your particular database could be **maintained** over time, again within the context of your proposal's scenario.
  - be specific!
  - do you have relations that must be updated periodically, or based on some event or transaction occurrence?
  - what kinds of issues might arise with regard to keeping the database current and useful over time?
  - would one person be able to take care of this database? which users would be permitted to perform maintenance activities? would a formal database administrator (DBA) be required?

Create and submit a file `325readme.pdf` that contains:

- your name(s)
- CS 325 - Fall 2016
- the date this readme file was last modified
- a list containing names and descriptions of all "code" files (SQL, SQL*Plus, PL/SQL, etc.) used in the final version of your submitted project
  - that is, follow **each** file name with a brief description of that file's contents
- a **separate** list of instructions for how to set up and use the database (which scripts are used to create and initially populate the database? etc.)
  - (these two lists are quite common contents in a README file -- they constitute very simple external

documentation for "installing" your project)

- Please note that I may or may not actually run your database -- however, I had better be able to use the instructions above to recreate your database from the files you submit, if necessary.

Also submit the final version of your business rules, `325biz-rules.pdf`

- **Yes**, submit it **even if it has not changed**. (you MAY find yourself adding to your business rules as part of the final milestone activities)
- Be sure that, as always, it includes an appropriate last-modified date

Only submit new population-related files if any have been changed since the population milestone.

Only submit your `325design.sql` file if it has been changed since the population milestone.

Only submit your `325model.pdf` file if is has been changed since the population milestone.

- HOWEVER -- NOTE that there could be a SUBSTANTIAL PENALTY if the latest versions of the `325model.pdf` and `325design.sql` files do not correspond/"go together"!!
- ALSO NOTE that there will be SUBSTANTIAL PENALTIES if the **Project minimum structural requirements** given in the Project Model milestone are not met in the latest versions of the `325model.pdf` and `325design.sql` files

Remember: if previous milestone(s) required that certain changes or additions be made to the proposal, to the previous set of business rules, to the model, to the design/schema, and/or to the population, then those changes should be reflected in newly-submitted versions for this milestone, or **this** milestone's grade may be affected

FINALLY, you may, if you wish, also create and submit a file `325misc.pdf`, accompanied by additional files as you desire, that contains:

- your name(s)
- CS 325 - Fall 2016
- the date this Miscellaneous file was last modified
- a description/discussion/list of anything else that you wish for me to consider while grading your project -- for example,
  - mention of features you have used in addition to those required (such as sequences, or views, or time/date functions, for example) and in which files those are used
  - lists of additional files you are submitting demonstrating additional features or functionality, such as PL/SQL triggers, files demonstrating that the triggers work, code for forms you have designed or implemented, etc.
  - If you have made some **extra** effort, and you want to **make sure** that I do not overlook it while grading your project, **then mention it/show it off here!**

The final project, if well-done, will be a package you can proudly show off as part of your "portfolio" for interviews, or possibly even use as a reference later for future databases that you design.