**Project Grading Scheme**

Specific deliverables:

Video

Report

Code

**Project-Component Requirements**

Server:

Client:

Data Mining:

--------------------------------------------------------------------------------

All grading is by letter grade, out of the following scale:

A-, A, A+: excellent work

B-, B, B+: very good work (trending to good, if in the B- range)

C, C+: acceptable work (note that there is no C-)

D: barely acceptable work

F: unacceptable or absent work

In the comments below half-a-letter-grade means a + or a -. So if the project

is a B, then half-a-letter-grade means B- or B+. It doesn't mean that two half

letter grades puts the project into a different category. So two increases of

half-a-letter-grade doesn't mean a B becomes an A-. An A grade is excellent

work; a B grade is not excellent work, but good to very-good work. The project

assessor will use his/her judgement in how this is determined. The + and - are

in the context of that judgement.

--------------------------------------------------------------------------------

There are three deliverables, per se: video, report, code

Deliverable Coherenece: Are the three deliverables coherent in-and-of-themselves? This is not marking for content in particular.

**By deliverable:**

**Video:** Good introduction; flows well, good description of whatever they are

doing, appropriate demo of whatever they are doing, good conclusion.

Highlighting particular interesting or tricky aspect of the project, and/or

particularly relevant design decisions is good and should add

half-a-letter-grade to your evaluation.

It should be ~20 minutes; coherence matters more than length: a well-done ~15+

minute presentation is better than a poorly done 20 minutes presentation;

likewise for ~25- minutes. Below 15 minutes or above 25 minutes are not good

and will be penalized by half a letter grade.

**Report:** Appropriate introduction, flows well, good description of whatever the

project is doing. There should be an appropriate ER model with entity sets,

relationship sets, etc. as shown in the course slides. The correctness of the

ER diagram is not evaluated in this section of grading; it is simply observed

that there is one there and it is well done. Appropriate descriptions of design

choices and/or tradeoffs. Appropriate conclusions, summary, as necessary.

**Code:** Well-structured: clear delineation of (a) client code (b) server code (c)

testcases (ideally also separated by client and server). Code looks well

written, decomposed into appropaite functions/modules/etc., with comments as

necessary, appropiate indentation, etc. E.g., we should not be seeing multiline

SQL code that is written as one single line that wraps a lot. This is not

judging the functionality of the code, per se.

--------------------------------------------------------------------------------

**Project Components:** there are three components to the project:

Client

Server

Data-Mining Exercise

They are judged for function as follows (per the project description):

**Server:**

The following items must be present:

(a) ER design with appropriate entity sets, relationship sets, cardinality

constraints using n:m notation, primary keys should be underlined, any optional

attributes, weak entity sets, etc. as necessary.

Notes:

(1) A design with very few entity sets/relationship sets/attributes in

not excellent, by definition. It is at best "good" (B). The

datasets needed a minimum of 50 attributes, and so appropriate

engagement with the project should produce something of

significance.

(2) The ER design is not simply the diagram: any additional descriptive

material may be necessary. For example, a specialization should

say if it is partial or total, as that is not visible in an ER

diagram.

(3) Design choices should be explained and justified. This can be

limited to major choices or significant issues. There should be

something in the report and/or video that covers design choices.

Minus half-a-letter grade if absent.

(b) Relational schema: should be in code in one or more .sql files, though

possibly in some other source code. Explanation of any non-obvious translations

from ER model to relational schema should be in the report and/or video.

Example of a non-obvious translation: a weak entity set where rather than using

the primary key of the entity set on which it is dependent together with

whatever the discriminator is, the project chose to use a defined new key for the

relation. A good example of where this is a bad design choice is the GamePlays

relation in the NHL database: it would have been much better to use gameID and a

playNumber within the game.

Notes:

(1) Appropriate indexes should be present

(2) Appropriate data-loading operations should be present

(3) Any necessary data errors should be described and fixed

(4) Appropriate foreign keys, constraint checks(), etc.

(c) Test plan and test cases necessary to validate the above. These do not have

to be exhausive. They should exist, and by exist this does not mean in code:

manual test cases are acceptable, provided they are detailed in the test plan.

Describing purpose, input, expected output, and result is sufficient.

**Client:**

The following items should be present:

(a) Ideal client requirements

(b) Actual client proposed

(c) Actual client implemented

(d) Test cases necessary to validate the client

(e) Justification for the client design

Client should include both the ability to query data and to update/input data,

as appropriate to the project area.

**Data-Mining Exercise:**

The following items should be present:

(a) a domain-appropriate question that the data-mining exercise attempts to

answer

(b) a technique or techniques that will be appropriate to the question

(c) a description of how the technique was implemented/used

(d) a desciption of how the model was validated

(e) a report of the results (the results do not need to be good)

Note: picking two or three variables and doing simple-linear regression is NOT a

data-mining exercise. In particular, in item (c) above the project should make

clear how attributes are selected and pre-processed (feature selection)

Overall Quality of the Project:

When looking at deliverables and required components of a project it is often the case that the point grades do not capture the overall quality of the project.

By way of example, consider building a car: you could build a complete, high quality, but limited functionality vehicle: manual windows, manual door locks, no A/C, three gears, etc. But it is extremely well put together. In such a case, the overall quality would be judged as some category of A.

Conversely, you could try to build a Cadillac, but the end product built is lacking a steering wheel, the automated windshield wipers do not actually work, and the doors are missing. In such a case the project may have excellent individual components, but the overall quality is poor and would be judged accordingly.