**Database Systems 22C:144 Palani Andiappan**

**General Project Description and Guidelines**

**Database Project (25% of total grade)**

Each group will identify a project using these brief descriptions. Your goal is to design and build an appropriate database. The project descriptions are purposely kept short and simple. This offers you considerable latitude in terms of how to define the specifics of the project. It also gives you the experience of going through “requirement analysis”. The projects are also kept intentionally close to real-world organizations. Thus most of us are likely to have some idea of how these run and the kinds of information they would need to track. Use this background knowledge. If you have any questions, you may ask me or ask the course TA through normal channels of communication.

*As a first step, each group must send the TA an email specifying the group’s top 3 project choices. The TA needs to receive this as soon as possible but* ***no later than******a week after project team selection date****. We will assign to your group the best possible choice on a first come first serve basis. If your 3 choices are already taken then we will assign an open option.*

**Timeline/Deliverables**

All deliverables are to be uploaded to our ICON site only. Deliverables are at the group level and not at the individual level. If multiple submissions are made for the same group we will use the latest submitted. Please us this file naming convention for all files

**<group#\_deliverable\_name\_project\_title>**. For example the file you upload for project description could be “Group11\_Project\_Description\_Amusement\_Park”.

Sample project deliverables will be provided in ICON. Please use these samples as guidelines to prepare your project deliverables. Please refer to <http://www.palaniandiappan.com/evaluation.html> for project deliverable deadlines.

1. *A one to two page project description.* This is an initial, but reasonably well-thought out, one to two page description of your project. Provide enough details for me to determine if the scope of your project is sufficient or not.

* Description of project
* Requirements of the project. What is the business purpose or need? What is the the problem you are trying to solve? What questions you will be answering?. For example a coffee shop owner might want to create a database that tracks all sales and links sales to registered customers using loyalty card. Owner may be interested in improving customer loyalty by sending them coupons. Owner may try to answer questions “What is the best selling soup?” or “What is the average sale of chicken sandwiches by weekday?” or “What is the most ordered you pick 2 combination?” or “What do registered customers order the most?”.
* Assumptions of the project. For example a coffee shop owner might have an assumption that all customers register online only once and use their loyalty card every time they make a purchase.
* Scope of the project. Here you will define what data you will be capturing and also what is out of scope. For example the coffee shop owner database will have all sales data and customer data. The database will not have purchases, raw material cost, profit loss statements, assets etc.
* Project Design Approach. This is a high level description of the design. This could be a what data will be captured and the logical groups of data you will be creating. Please refer to the data grouping exercise. The groups are nothing but relations.

A second aim of this deliverable is to have you begin your work together as a group. *This deliverable will not be graded*.

*2. Entity-Relationship Model.* Provide a complete design for your project using the Entity-Relationship Model. 30 points

*3. Relational Design.* Provide a complete relational design for your project. Include designs for your user views as well as the data dictionary. 30 points

Include the following:

* An updated description and ER model as per feedback given earlier (Note that the submission is a cumulative document).
* A complete relational schema that derives from your ER model and is normalized to the 3rd normal form.
* Relational integrity constraints in your design.
* All non-trivial functional dependencies for each relation in your schema.
* A 1-page justification for your normalized relational schema.
* A data dictionary: which will include domain definitions. Note: you do not need to define data that is generally understood (e.g., age, Social Security Number). Approximate structure:

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Definition** | **Data Type** | **Notes** |
| Salary | Income (per year) in dollars | Numeric | Cannot be null |
| Evaluation | 1 sentence about employee performance | String |  |
| Status | one of (high, medium, low) | String | Default: medium |

* Authorization decisions: For each type of user (customer, employee, etc.) in your problem, identify activities they will be able to perform on each relation (read, insert, update, delete) in an authorization table. Example of such a table:

Assume we have a schema with just these two relations:

Person(ID, age, email)

Job(JobID, ID, salary, job\_description, evaluation)

|  |  |
| --- | --- |
| **User role** | **Privilege** |
| Customer role | Person: Select, Update (email)  Job: Select (JobID, ID, salary, job\_description) |
| Employee role | Person: Select, Update, Insert  Job: Select |
| Manager role | Person: All  Job: Select, Update (job\_description, evaluation) |
| Owner role | Person: All  Job: All |
|  |  |

* Additional normalization problem to solve: Place all the attributes in your project into a single relation R (first make sure that each has a unique name). Then derive a relational scheme design that satisfies the best possible normal form. Provide the justification and show the derivation of this design as appropriate. For example, if you do a chase test for lossy/non lossy decomposition, show this.

*4. Database Implementation.*

Relational database implementation

* Implement your database design with sample data and a Web accessible interface. You must use the CS database server for this. Provide written documentation about the decisions you have made regarding the physical design (indexes for example). Estimate sizes of your relations. 40 points
* This is really straightforward. Implement all your relations, add a few tuples of data in each (make up the data). Create enough sample data to solve your original questions or problems that you started with. Keep the number of rows in any table to a minimum of 2 and a maximum of 20. Implement at least 3 SQL queries of fair degree of complexity involving multiple relations (join/sub query). Make sure these work correctly. Submit appropriate documentation. Interface may be simple. **Please refer to implementation instructions at the bottom of this document.**

MongoDB Model

* Submit a MongoDB model(NoSQL model) of your final ER model. You will submit the model only in a word document. You do not have to build a database. For the Mongo DB model use the minimum amount of data necessary to show the model (one tuple should be enough).

**Important Note 1**: All deliverables must have a cover sheet. In it each group member will briefly describe his/her contribution to the deliverable.

**Important Note 2**: If there are significant problems related to group dynamics etc. please let me know immediately. The sooner we solve such problems the better for you.

**Database Systems Projects**

Please design a database system for a gig or sharing economy type project. Some examples are

Uber

AirBnB

Handy

TaskRabbit

Instacart

Fiverr

You will have to track the following information

* Seller or service provider
  + Identification, demographics and contact information
  + Service/product they provide. How and when they provide the service
  + Price or compensation
* Customer
  + Identification, demographics and contact information
  + Service they want and when
* Administrator
  + This is you – your organization information
  + How you will make money
* Matching or Search
  + How customers will be matched with sellers
  + Both customers and sellers should be able to search
* Service tracking and completion
  + How will you track if the service has been provided or not
* Gather customer feedback on service

Feel free to expand upon this basic scenario as appropriate.  
  
  
**Relational Database Implementation Instructions**

For the last part of your project, you will be implementing your system in a MySQL database on an Iowa server. You will be writing a database and some sample data, and then you will write 3 queries that can be executed on your website.

Everyone in the class has their own webdev account and their own web page. Your web page is at

http://webdev.cs.uiowa.edu/~<hawkid>/

For example, my page is

http://webdev.cs.uiowa.edu/~rishsingh/

You will only be able to access the website from the campus network.

To be able to make any modifications to your website, you will need access to the DIVMS Linux machines. Everyone should already have made an account for these machines, and your username and password should just be your hawkID and password. There are different ways to log onto your Linux account.

1) If you want to connect to your Linux account from your own computer, you will need to install some additional software to make the connection. One option is NoMachine, which is available from CSG here.

<http://www.divms.uiowa.edu/clas_linux/help/nomachine/windows/>.

Simply follow the instructions on that page to get everything set up

2) You can also use an SSH client, but you will have to do everything through the command shell if you use that technique. One example of an SSH client that you can use is Putty, but you can use any other you find as well. You should use the following settings with an SSH client.

3) You can also connect to Linux account using any web browser. Here is the link:

<https://linux.divms.uiowa.edu>

Just use your hawkid and password to start a new session or resume a previous session.

Host = linux.divms.uiowa.edu

Port = 30022

Username = your hawkID

Password = the password associated with your hawkID

Once you have logged into your Linux account, you will have access to a "webdev" directory, which contains both the HTML files that make up your web page and several example scripts that demonstrate how to run execute SQL queries on your web page. To access your web directory, open a command terminal and use the following commands.

cd /webdev/user/yourusername

ls

You should see a list of files that looks like this

error\_log examples index.html mysql\_db\_info psql\_db\_info ssl\_error\_log

Most importantly, index.html is the home page of your website. You can make changes to it and see those changes reflected on your website. The other important file is mysql\_db\_info, which will have your MySQL username, password, and database name. You will need to use these to log into MySQL and build your database. Inside of examples, you will find several files that demonstrate how to make SQL calls through web scripts.

Next, let’s look at how to log into mysql and run commands. To connect to mysql, use the following command

mysql -h"dbdev.cs.uiowa.edu" -u"YOUR-USERNAME" -D"YOUR-DATABASE-NAME" -P"3306" -p"YOUR-PASSWORD"

You can also exclude your password by just typing the argument -p, and MySQL will prompt you to type it.

One MySQL is open, you can do several things. Do not try to change your password.

You can do anything in MySQL from the command prompt, which means it is possible to create your entire database through the interpreter. However, this is not easy to do, so a better option is to create a file containing your schema and then run that. An example SQL file has been placed on ICON for you to use.

Once you've created a SQL schema file, you can load the schema into your database using a command like the following.

mysql -h"dbdev.cs.uiowa.edu" -u"PUT-YOUR-USERNAME-HERE" -D"DATABASE-NAME" -P"3306" -p"PUT\_PASSWORD\_HERE" < YOUR\_FILE.sql

Importantly, you should make sure your sql file has the line USE <YOUR-DATABASE-NAME> at the top. You will only have permission to use a single database, so you cannot choose a database with an arbitrary name. Simply use the database name given in the mysql\_db\_info file.

After you've created your database, the last thing you need to do is write three queries and link them on your website. If you go into the /examples directory, there are examples of how to do this using both Python and Perl. You can use whichever you are more comfortable with.

Essentially, all you need to do is take one of the example files, substitute your database information for the example information, and write in your own query where the example query is. Then, in index.html, create new links to your scripts and remove the links to the example scripts.

**Submission Instructions**

For your submission, you need all of the following.

1) Cover page

2) The URL of your website (only 1 website per group)

3) Your SQL file with your schema and your sample data (having the schema and the sample data in separate files is fine if that is what you want).

4) Your SQL queries from your website.

5) Documentation of your schema (why did you select your indices, what design choices did you make, any points to clarify, and so on).

6) Updates to all of your prior work.