Project Description, Advanced Database Management, INFO606, Winter 2022

Il-Yeol Song

Students will be required to do a term project. This document describes teaming, topics, proposal, grading components, and final project report submission guidelines related to the term project. The purpose of this project list is to provide you with choices of different aspects of database systems, both theoretically and practically.

Sumamry:

- A team may consist of up to **3 to 4 members**. A team may consist of students from either online or F2F sections. A solo project is allowed only in a special circumstance (e.g., the last team or a reason that can be justified) and must be approved in advance via emails.
- *Teaming must be done by the end of Week 2 (*Sunday Jan 16, 2022, 11:59PM) *and each team must post their topic and team members by Monday of Week 3.* ***If you do not join a team by the end of Week 2, you will get -5 in the project grading.***
- *If your name is not shown by the Monday 9AM of Week 3, I will randomly assign you to a left-over topic. In order to avoid this randomly assigned teaming and topic and the penalty of 5 points, make sure you participate in teaming in Discussion Board of Week 2.*
- The proposal should include the title, goal, scope, description, methods employed, hardware and software involved, the procedure to work on, experiment/implementation plan, expected significance, and the items to be delivered at the end of the project.
- **The proposal must include only those names who have collaborated to the proposal effort and exclude those names who did not collaborate**. If any one's name is not included in the submitted proposal by the due date, you will lose 10% of your project grade, but you still have to submit your own proposal as **a solo project**.
- A term report should be organized in a logical sequence. References should be properly cited in the contents and listed at the end of a report or at the bottom of the slide.
- ***Project grading*** *will consist of the following four components:*
  - *Teaming: 5%*
  - *Proposal: 10%*
  - *Final report: 85%:*
    - *Written report:60%;*
    - *presentation 25%*
- **Important dates:**
  - **Teaming for the term project:** The end of Week 2, Jan 16, 2022
  - **Project proposal**: Wednesday of the 4th week, Jan 26, 2022 (10% of your project grade)
  - **Project due**: Thursday of Exam Week, March 17, 2022 (I will create a link for each team to submit in Blackboard)
  - **PEF:** Friday of Exam Week, March 18, 2022

- *If you are interested in a few choices but need my advice in deciding a topic, send me an email with the top 2 or 3 choices with a short description of your study goal or career goal as well as your background. I will recommend one topic that could best suit your goal. Send me an email (song@drexel.edu) with your email heading of* ***INFO606 project****.*

Project Description, Advanced Database Management, INFO606, Winter 2022
Il-Yeol Song

Topic

Here are some recommended project topics. They are:

- **Relational application development**: You will develop an application running in relational database system such as Oracle or MySQL. If appropriate, your project should consist of SQL, working triggers, stored procedures, and stored functions that demonstrate your competency of the course material. ***Each*** member is required to create a total of **four** *working and meaningful PL/SQL program units* (any combination of triggers, stored procedures or stored function with at least one of each unit) that are useful for the application. You could choose to continue working on your INFO 605 project to save time in developing the ERD and relational schema etc. Or you can begin a brand new project. Some examples are listed below.

  • ***Multimedia DBMS Implementation***: Implement a small multimedia database system in a chosen application. Here, a multimedia database means you should include at least two of audio, image, text, and video data type in your schema. You can choose any domain. You can use any tool such as Oracle, MySQL, or SQL Server.

  • **Startup.com. Develop a web application with SQL database on the backend.** Assume you are creating a start-up company for E-business. Limit the scope to reasonable core functionalities and create a sample prototype. For example, assuming you are selling authentic CDs (or books, paintings, dolls, personal collections) you collected. For simplicity, assume all the payments are done via credit cards. You can use any software and DB system.

  • **Unstructured Data Management in Oracle (MySQL)**: Experiment with unstructured data such as XML or JSON. Learn to create, store, load, index, query, update XML (or JSON) data in Oracle or MySQL. You must document your schema and data used for (ERD, RDB schema, DDL), the data set used for testing, SQL queries tested, output from the DB system, and interpretation of the tests. You may or may not use PL/SQL for this topic.

  • **A Simplified Patient Care Clinic System (PCCS):**
  A small local clinic wants to computerize the clinic. The main purpose is to keep track of patients, their appointments, medical histories such as symptoms, treatments, and payments. Assume the clinic has only 2-3 doctors (as well as a few nurses and staff) and wants to have a minimum functionality, instead of a comprehensive system for major hospitals. You can simplify the environment to make the system a reasonable size. (Note that about 80% of small clinics in USA do not use a computerized system yet. The main reason is that most patient managements are targeted for big hospitals and too complex for them to use and afford them. Thus, target a simple, usable, and minimal system for real-world setting.)

  • **Relational Database and Python**
  Develop a small application where Python can be used to as an application program with MySQL or SQLite. SQLite is a light version of an open-source SQL system. Study SQLite in terms of installation, administration, SQL features, interface with other software such as Python, R, or other languages/applications.

  • **Database in a Cloud:** Experiment how to develop a database application in the cloud environment such as AWS, MS Azure, or Google.

• **Data Warehouse in a Cloud:** Experiment how to develop a data warehouse in the cloud environment such as AWS, MS Azure, or Google.

• **Data Lake Technology: A data lake is a big data repository that stores unprocessed raw data.** Study how data lakes store the data, search the data, and are implemented. Examples of data lake for Covid19 data is presented at https://c3.ai/products/c3-ai-covid-19-data-lake/.

• **Data Lakehouse Technology: A data lakehouse combines best elements of data warehouses and data lakes.** Study how data lakehouse technology integrates and storse the data, searchs and analyzes the data, and supports big data management. A well-known data lakehouse platform is The Databricks Lakehouse Platform, https://databricks.com/product/data-lakehouse.

• **MongoDB Applications:** MongoDB is the most widely used document-based NoSQL database system. We will discuss installation, modeling, basic CRUD operations, and indexing in this class. Your team will develop a meaningful application using MongoDB. You may study PyMongo to study how to use MongoDB with Python.

• **Cassandra:** Cassandra is the most widely used column-based NoSQL database. Learn installation, modeling constructs, CRUD operations, query processing, management, and administration in Cassandra. Experiment its schema features and play with it.

- **Spark**

Spark is a unified cluster computing framework that supports in-memory computing for analytics, machine learning, SQL, real-time, and graph data applications with multiple language interface. Study the two of SparkAPIs from SparkSQL, MLlib, SparkStreaming, GraphX. Experiment with a small, but meaningful application in Spark. There are many resources available on Spark at Google or Keggle. I recommend you use Google Colab to simplify Spark installation. PySpark is an interactive Python shell that can work with Spark for big data analytics. The PySpark shell allow you to link the Python API to the spark core and initializing the spark context. A good tutorial for PySpark is here: https://www.dezyre.com/apache-spark-tutorial/pyspark-tutorial.

• **DBT:** DBT (Data Build Tool) is data transformation tool based on SQL. It can be used in transforming the data for data warehouse. Study and experiment it. Learn more at https://getdbt.com.

• **Google Big Query:** Study how to build a data warehouse using Google Big Query and analyze using Google Analytics.

- **Case Study** : Do an in-depth case study of a real-world big data project. Study its goals, business environments, tools and systems used, architecture, data types and data, results, impacts, etc.

C. Your own topic

If you have other topic under either implementation or research category, submit a proposal, for approval, including nature and scope of the problem, methodology, involved hardware, software, deliverable at the end of project completion, and any known references.

Project Design/Draft Review

Project Description, Advanced Database Management, INFO606, Winter 2022
Il-Yeol Song

If your project involves design and implementation or includes ERD and RDB schema, I recommend you submit your project ERD and RDB schema for my review by the end of Week 6. This is to enforce you to start early and keep you on track for the rest of your project.

Submit a self-contained single word file that contains the problem statement, ERD, and relational schema, or any other material you want me to review. Submit via email.

Proposal

Every team must submit a proposal. It could be approximately about 3 page long (in a single space using time 12 font) with the proper format. Your proposal should have a proper structure divided into sections such as the title (short, but meaningful), goals of the study, scope (**IN-scope** that will be included in your project and **OUT-Scope** that will not be included), project plan (how you will proceed to learn the topic), knowledge acquisition to learn about your project, examples of input and output, experimentation plan (what data set to use, which techniques to learn, etc, if the project includes implementation), HW and SW environments, significance once the project is complete, deliverables upon completion, and any known references (at the time of writing the proposal).

- ***Submit a word file, not PDF***, for my easy commenting.

I will create a link to submit the proposal in Week 4 Assignment folder.

Basic Principle of Documentation and Grading Policy

The basic principle of documentation in any report is very simple: readers should be able to understand your work or grade your project by just reading your report without you. This principle is called *reviewer-friendly* writing. For this purpose I suggest you keep in mind the following four points.

(a) Readers of your report should be able to tell what you have accomplished. For example, you can include the following ideas if they are relevant to your problems: what problems you have worked on, what assumptions you have made in your work, how much you have accomplished, what parts you did not finish (or limitation of the result of your work: do not try to hide the problems; nobody's work is perfect), how well you did, how important your work is, why it is important, where it can be applied, what kinds of difficult problems you had during your work, what the remaining problems are, what parts can be done as the next steps, etc.

(b) The report should be easily readable and understandable. No matter how significant your work may be, it is not worth full credit if you cannot convince readers. That is, you should simply write what you studied. Instead, you should write how to make your work understandable by readers. Use clear and concise English. **Keep in mind you should write to *communicate* with readers, not document the work for yourself.**

(c) The report should be well organized. It should have a clear logical sequence. Organize your work in chapters, sections, and sub-sections with meaningful headings. Include diagrams, tables, or figures whenever appropriate.

(d) The report should be self-contained, if possible. Do not ask readers to look around other references, unless you have a reason to do so.

(e) There should be no errors during the demonstration of the program. The implemented part should be robust against various inputs. Any known errors should be documented.

- The best method for documentation is *incremental documentation*. **DO NOT WAIT UNTIL YOU COMPLETE YOUR SYSTEM**. *It is very easy to forget the fresh idea you had by the time you finish your project*. Write whenever you have some material that you need to include in your final report.
- If you are in doubt in correctness of your final deliverables, send them to me for my review before you submit the final version to be graded.

Term Project Grading Criteria:

Your project will be graded by the following four criteria

(a) *Technical significance*: The contents must have some technical substances, not just concepts.
(b) *Clarity*: The contents must be presented with principles, specific examples, diagrams, and tables whenever possible, so that the ideas can be easily understood.
(c) *Up-to-date*: The contents must be up-to-date with proper references to existing materials.
(d) *Structure and style*: The report must be well organized
- Use proper chapter and section headings for easy grasp of topics
- Discussion style (Introduction-Main-Conclusion)
- Looks (page numbers, grammar, spelling, stapling or binding)
- Citation (*cite properly in the body* of the report, in addition to the list of references)

Project Deliverable and Submission

All the project teams will submit a soft copy to Blackboard. I will create a submission link in Week 11 Assignment folder:

(1) If you are working on PL/SQL project, **remember that each member must have four program units of his/her own responsible procedure, function, or trigger**. Hence, make sure to **add your name right above the procedure, function, and trigger names** to indicate who is responsible for each procedure, function, and trigger. You can help each other, but there must be one responsible member for each procedure, function, and trigger. Each member will get a different grade, depending on the nature, significance, and complexity of the program unit.

Here is an example style for a stored function:

//Developed by **Tom Cruise**
// **GET_USER_DURATION** function will determine how long the users have been registered (member since) in the application. This function will compute the membership duration through the creationDate in the ACCOUNT table.

(2) Create **one zip file** that includes **the report file in Word**, **the data set used** (if your project involved an experimentation), a **power point file**, or any other related files (e.g, Visio or draw.io diagram files, program codes, screen shots, input, output) used in the project.
(3) Your report should be comprehensive by consolidating all the materials into a single Word file including the final report, diagrams, program or SQL code, PL/SQL code, input, output, etc. That is, copy and

paste all the diagrams and programs into the final report. Organize them in a reviewer-friendly manner. Add page numbers.

(4) Use the naming convention of **606-22W-P-yourLastName_or_team_name-acronym-of-your-project.zip (e.g., 606-22W-Cloud.zip).** That is, by reading your zip file name, I should be able to easily identify your term and project topic.

(5) Submit project and peer evaluation form to Blackboard Assignment folder of Week 10. Your PEF also should have the naming convention of **606-22W_PEF_yourLastName_ProjectName (e.g., 606_22W_PEF_Song_Cloud.doc).**

- **PEF (Peer Evaluation Form)**
  - At the end of the term, each team member must submit a PEF (Peer Evaluation Form) that describes the division of team work and evaluates other team members. In the PEF, you have to specify the level of the effort done by each member of the group both in your proposal and in your term report.
  - A PEF must be submitted to the Assignment folder within 24 hrs of submission of the final report.
  - PEF submission link will be posted in Week 10
  - If you do not submit a PEF, you may get a lower grade than your team member.

- **How to Lose Some Points in the Documentation with Excellent Work**

I have noticed that many students who produced excellent implementation of applications have lost some credit because of poor documentation. It is not the thickness of the report that matters, but the structure and content of the report. I suggest you read and follow the guidelines above. Here are some ways to lose some points on documentation in spite of your excellent work.

• When there is no summary section in the first page of the report. The summary should show a brief overview of the whole project including goals, methods, conclusions and systems used.
• When there is no CONTENTS section with page numbers. I have to flip over pages to find info. that I need.
• When I don't know how to use your system or where to look to test your system by reading your documentation.
• When your documentation does not have any structure and it is hard to understand what you have done.
• When your documentation does not follow the logical order of the problem-solving process. It should follow the order of Introduction, Goals, Scope, Design, Methods employed, Experiments, Implementation Code, Conclusion, and Appendix.
• When it does not have a section on "How to install" and "How to run" section.
• When your direction in those sections does not match or does not have any actual screen shots.
• When you wrote your documentation by assuming that I am familiar with your database system and application problem. You have to write in a manner explaining your work to someone who is not familiar with your system and application. It should be self-contained as much as possible.