

LIS-542: Database Applications Development

Autumn 2006
Master of Library and Information Science
Information School
University of Washington

Introduction to relational database theory and technology from an information science perspective. Focus on traditional transactional database theory, architecture and implementation in a user-centered systems context. Introduces set and graph theory, relational algebra and introduces data warehouses.

Prerequisite: LIS 511 or 540

Course website & Listserv

<http://courses.washington.edu/lis54206/>

lis542a_au06@u.washington.edu

(Archive: https://mailman1.u.washington.edu/mailman/listinfo/lis542a_au06
Registered students are subscribed automatically using their UW mail account.

Credit Hours

5 (3 lecture hours; 2 lab hours; 10 outside hours) – credit/no credit

Meeting times

Lecture Tuesday/Thursday 1130-0120 – MGH 238

Lab Thursday 0130-0220 – MGH 4230

Instructor

David Hendry, Assistant Professor
330J Mary Gates Hall
dhendry@u.washington.edu | <http://faculty.washington.edu/dhendry>

Office hours: Thursdays, 0230 – 0330 or by appointment.

Student services

Marie Potter, Academic Advisor
470E Mary Gates Hall
mardup@u.washington.edu
Tel: (206) 616-2544

Overview

Relational Database Management Systems are an enabling technology. In business, they underlie mission critical systems for inventory control, order processing, and decision support. On the web, they underlie much of what we do when we consume and produce information. While they don't solve all the problems related to information storage and access, their importance in commercial and civic institutions is hard to overestimate.

The aim of this course is to develop an understanding for how relational database systems are used to store and access information. To do this we shall examine the functions that relational databases provide, how information systems are built using relational databases, how SQL is used to specify and query databases, and how database systems can be designed by following a methodology of stepwise refinement. Most of all, we shall develop a technical foundation so that we can think and talk like a database designer – at the same time, we shall reflect upon what we learn as information professionals who, in the future, are likely to be members of teams that are developing database applications.

Textbooks and readings

The textbook for this is course is:

Connolly, T. M. & Begg, C. E. (2005). *Database Systems: A Practical Approach to Design, Implementation, and Management* (4th Edition) New York: Addison-Wesley Publishing. [ISBN: 0321210255]

Why this book? This book is primary designed for undergraduate students in computer science. Nevertheless, I've used this book with good success with Informatics students and I believe it will work well for LIS-542 because it gives a very thorough treatment of the technical topics that we shall cover. And, because it takes an engineering orientation, I believe it will help us think like engineers and appreciate the elegance – and complexity – of database systems.

To give perspective, additional readings may be assigned from time to time.

Learning

Aims

The general aims of this course are to:

1. Develop a conceptual understanding for relational database systems
2. Develop skills in designing, implementing and testing database systems.

Objectives

On the successful completion of this course, you should be able to:

1. Describe the functions and organization of database management systems
2. Describe the relational model, including the data structure and algebra
3. From problem statements, derive SQL statements for querying, updating and creating databases
4. Create Entity-Relationship and Enhanced-Entity Relationship models for small systems
5. Read an ER diagram as a specification and implement a database system for it

6. Describe the problems of data redundancy and update anomalies and be able to normalize a database to 3NF to avoid these problems
7. Describe a three-tier information system
8. Outline and follow a methodology for designing database applications
9. Be comfortable with interacting with databases from the command line and from graphical user interfaces.

Assessment

This five credit course will be graded as credit/no-credit. Over the quarter, you will complete 10 exercises. To receive credit for the course, each exercise must be satisfactorily completed. Work that is graded needs improvement can be resubmitted for a second level of review.

Exercise	Due	Week
A1 - SQL Queries I	Oct 12	#3
A2 - Relational algebra	Oct 19	#4
A3 - SQL Queries II	Oct 26	#5
A4 - SQL Data Definition	Nov 2	#6
A5 – Reflections on System-Centric Methodologies	Nov 9	#7
A6 - Entity Relationship Modeling I	Nov 16	#8
A7 - Entity Relationship Modeling II	Nov 28	#9
A8 - Database Normalization	Nov 30	#10
A9 - Reflections of Learning	Dec 6	#11
A10 - Database Design	Dec 11	#12

Exercise “A9-Reflections on Learning” is special. This exercise takes place throughout the quarter. For this exercise, you are to keep a personal journal and document the technical, skills-oriented, and conceptual blocks that you encounter in learning this material. The goal is to document these difficulties and to reflect upon why they occurred and how you overcame them. On the last day of class we shall draw upon these documented experiences and our reflections of them to discuss what we’ve learned and what remains to be learned. We might, for example, discuss our reflections under the theme “Thinking like an engineer”.

Exercise handouts will be available on the course website and distributed in class.

Standard cover sheet

To protect your privacy when exercises are returned and to facilitate communication, submitted work must have a cover sheet. The cover sheet must include the following information and be formatted nicely:

- Course name
- Quarter, program, department, and university
- Assignment name
- Your name and e-mail address
- A date
- A web site address (if relevant).

Staple the exercise pages to the cover sheet.

Late policy

If you will miss the deadline for a piece of work, you should inform the instructor as soon as you can, indicating when you will submit the work. The instructor will seek to accommodate your needs.

Right to revise

The instructor reserves the right to revise this syllabus.

Guidelines on using e-mail

When communicating with the instructor, please follow these guidelines:

- You are welcome to give feedback to the instructor about the course, to ask a question about an assignment, to share an interesting article or resource, to report that you will be absent from a class/lab, to request additional time for an assignment (because of significant health, personal, or educational matter), or similar communication ;
- Whenever appropriate, please copy the class listserv with your question or comment;
- E-mail concerning assignments might not be replied to if it is sent within 36hr of the assignment due date;
- If your e-mail concerns your grade, please follow the re-grading policy (see above);
- E-mail that is sent on Friday afternoon or over the weekend is not replied to until Monday or Tuesday of the following week;
- If you don't receive a reply within 2 days or so, please resend your e-mail or ask about it during class or lab.

Class Schedule

Week 1: Overview (Sept 25 – 29)

No readings
L1 *** No class meeting
L2 Greetings and introduction
Lab Introduction to tools

Week 2: Database systems – why do we need them? (Oct 2 – 6)

Read Chapters 1 – 2
L1 Functions & components
L2 Data models & three-tier architecture
Lab Introduction to SQL and Postgresql

Week 3: Relational model (Oct 9 – 13)

Read Chapters 3 – 4 (4.2 optional)
L1 Relational data model
L2 Relational Algebra
Lab SQL Practice

(continued ...)

Week 4: SQL, I (Oct 16 – 20)

Read Chapters 5 – 6 [*please read for basic overview*]

L1 “Simple” queries (***) Dave to miss class (***)

L2 Grouping queries

Lab SQL Practice

Week 5: SQL, II (Oct 23 – 27)

Read Chapter 7

L1 Queries on multiple tables

L2 Database definition

Lab SQL and MS Access

Week 6: Database planning and design (Oct 30 – Nov 3)

Read Chapters 9 – 10

L1 Modeling systems

L2 Systems-centric and human-centric design

Lab Requirements Gathering Exercise

Week 7: Entity-relationship modeling (Nov 6 – 10)

Read Chapter 11

L1 Entities, attributes, relationships

L2 Structural constraints

Lab Drawing and modeling tools

Week 8: Enhanced entity-relationship modeling (Nov 13 – 17)

Read Chapter 12

L1 Modeling object structures

L2 Generalization/specialization and aggregation

Lab Modeling practice

Week 9: Normalization (Nov 21 – 24)

Read Chapter 13

L1 The normalization process and 3rd normal form

L2 *** No class meeting – Thanksgiving

Lab *** No class meeting – Thanksgiving

Week 10: Database design methodology, I (Nov 27 – Dec 1)

Read Chapters 15 – 16

L1 System-oriented methodologies – The conceptual, logical & physical

L2 Conceptual database design

Lab —

(continued ...)

Week 11: Database design methodology, II (Dec 4 – 8)

No readings

L1 Logical database design

L2 Reflections on “Thinking like an engineer” and good-byes

Lab —

Students with Disabilities

To request academic accommodations due to a disability, please contact Disabled Student Services: 448 Schmitz, 206-543-8924 (V/TTY). If you have a letter from DSS indicating that you have a disability which requires academic accommodations, please present the letter to me so we can discuss the accommodations you might need in the class.

Academic accommodations due to disability will not be made unless the student has a letter from DSS specifying the type and nature of accommodations needed.

Grading Criteria

General grading information for the University of Washington is available at:

http://www.washington.edu/students/gencaat/front/Grading_Sys.html

The iSchool has adopted its own criteria for grading graduate courses. The grading criteria used by the iSchool is available at:

<http://www.ischool.washington.edu/resources/academic/grading.aspx>

The UW undergraduate grading guidelines, used by the iSchool and available at

<http://depts.washington.edu/grading/practices/guidelin.htm>, may be used in this class.

Academic Integrity

The essence of academic life revolves around respect not only for the ideas of others, but also their rights to those ideas and their promulgation. It is therefore essential that all of us engaged in the life of the mind take the utmost care that the ideas and expressions of ideas of other people always be appropriately handled, and, where necessary, cited. For writing assignments, when ideas or materials of others are used, they must be cited. The format is not that important—as long as the source material can be located and the citation verified, it’s OK. What is important is that the material be cited. In any situation, if you have a question, please feel free to ask. Such attention to ideas and acknowledgment of their sources is central not only to academic life, but life in general.

Please acquaint yourself with the University of Washington's resources on [academic honesty](http://depts.washington.edu/grading/issue1/honesty.htm) (<http://depts.washington.edu/grading/issue1/honesty.htm>).

Students are encouraged to take drafts of their writing assignments to the Writing Center for assistance with using citations ethically and effectively. Information on scheduling an appointment can be found at: <http://depts.washington.edu/iwrite/>

Copyright

All of the expressions of ideas in this class that are fixed in any tangible medium such as digital and physical documents are protected by copyright law as embodied in title 17 of the United States Code. These expressions include the work product of both: (1) your student colleagues (e.g., any assignments published here in the course environment or statements committed to text in a discussion forum); and, (2) your instructors (e.g., the syllabus, assignments, reading lists, and lectures). Within the constraints of "fair use", you may copy these copyrighted expressions for your personal intellectual use in support of your education here in the iSchool. Such fair use by you does not include further distribution by any means of copying, performance or presentation beyond the circle of your close acquaintances, student colleagues in this class and your family. If you have any questions regarding whether a use to which you wish to put one of these expressions violates the creator's copyright interests, please feel free to ask the instructor for guidance.

Privacy

To support an academic environment of rigorous discussion and open expression of personal thoughts and feelings, we, as members of the academic community, must be committed to the inviolate right of privacy of our student and instructor colleagues. As a result, we must forego sharing personally identifiable information about any member of our community including information about the ideas they express, their families, life styles and their political and social affiliations. If you have any questions regarding whether a disclosure you wish to make regarding anyone in this course or in the iSchool community violates that person's privacy interests, please feel free to ask the instructor for guidance.

Knowing violations of these principles of academic conduct, privacy or copyright may result in University disciplinary action under the Student Code of Conduct.

Student Code of Conduct

Good student conduct is important for maintaining a healthy course environment. Please familiarize yourself with the University of Washington's Student Code of Conduct at: <http://www.washington.edu/students/handbook/conduct.html>

A1: Getting Familiar with PostgreSQL

Worth: 10%

Creating and running SQL scripts

It can be very helpful to create SQL scripts so that you can group together a series of SQL commands. For example, consider the following SQL script.

```
/* filename: mkbranch.psql table */

/* Connect to my database -- CHANGE THIS !!!*/
\c dhendry

/* Remove the Branch table (in case an old version exists) */
drop table Branch;

/* Define the Branch table */
create table Branch(
    branchNo char(5) not null,
    street varchar(35),
    city varchar(20),
    postcode varchar(10)
);

insert into Branch(branchNo,city) values ('10', 'Toronto');
insert into Branch(branchNo,city) values ('20', 'Seattle');

/* Check to see how many rows are in the Branch table */
select count(*) from Branch;

/* Show all the rows in the Branch table */
select * from Branch;
```

To run this script you need to do the following:

1. Use WordPad (or some other editor) and type in the script
2. Save the script to a file, say **mkbranch.psql**
3. Copy this file into a directory on your Linux account
4. Start psql in that directory
5. Issue the following PSQL command
`\i mkbranch.psql`

A1: Getting Familiar with PostgreSQL

Worth: 10%

System catalog commands

You can determine the names of all the existing tables with this command

`\d`

Then, you can show all information about a table by naming the table as in

`\d building`

Although it probably won't be helpful to you, you can find the names of all users with

`\du`

There are many uses to which the system catalog can be put. To learn more about the system catalog commands type

`\?`

A1: Getting Familiar with PostgreSQL

Worth: 10%

The deliverable

For this assignment, please answer the following questions in a paper-based report:

1. In your own words, please explain lines (1) – (6) that are associated with the cat command.
 2. Type in a few Linux commands. Now, press the UP and DOWN arrow keys a few times. Explain what is happening? Do the arrow keys work in the PostgreSQL command shell? How might this feature be useful?
 3. Log into psql and create your own table, add some data to it, and query it. Use the SQL command CREATE TABLE, INSERT INTO, and SELECT. To show your work, please copy the SQL statements from the PostgreSQL command shell and paste them into your report.
 4. Type in the PostgreSQL script shown above in the section *Creating and running SQL scripts* and run the script. Now, go to the course home page. You will find scripts and data for creating the BoatClub and DreamHome databases. Run these scripts.
 5. In your own words describe the system catalog (hint: see chapter #2, Connolly & Begg). Using system catalog commands, show all tables in your database and show all details for one of your tables.
 6. We have been using the command line to create and issue SQL commands. Many Graphical User Interface (GUI) applications have been developed for making the process of creating and editing SQL commands a little easier. Do a little research and identify several GUI applications for query PostgreSQL databases. Describe a couple of options.
 7. Reflect on the learning objectives for this assignment. What is their nature? What approach to learning do they represent? Can you think of an alternative approach? Write a brief reflective statement on these and similar questions.
- * In your personal journal describe what it was like logging into Linux, exploring the command line, and using the PostgreSQL interpreter. What was it like to learn these commands? Did you make mistakes? How were you informed of them? Was the computer helpful? Was all of this easy or was it challenging.

Learning objectives

- To use the system catalog to discover the structure of a database
- To reverse engineer and describe a database
- To learn some basic features of the SQL SELECT statement.

Getting started

- On the course website you will find a database called: "usa_pop.zip". To complete the assignment, you will need to load the database into your PostgreSQL account.

Questions

1. Create a description of the data. Your description should include 'map' that shows all the tables, the tables' attributes, the connections between the tables, the size of the tables. Include any other information that might be helpful.
2. Analyze the tables and answer the following question: Could the database consist of fewer tables? Explain the rationale for your answer.
3. Write SQL statements for the following queries:
 - a. List all the zip codes and cities for the state code WA.
 - b. List all the zip codes and cities for the state codes WA, MA, and OR.
 - c. List all details for all cities named Boston or London.
 - d. List the state that has a population of 447100 in 2000.
 - e. List the states that have a population greater than 800,000 but less than 3,000,000.
 - f. List all details for all cities named Boston or London but only four 3 states of your choosing.
 - g. List the states (state codes only) with more than 1000 zip codes and list states alphabetically.

Deliverable

Please submit a concise report that answers these questions.

Learning objectives

- To describe the relational model conceptually and using set notation.

Questions

- a. Given these three domains, show their cross product
S_domain = {1, 2, 3}
T_domain = {X, Y}
V_domain = {'hello', 'fred'}
 - b. State the general rule for a valid relation given these domains.
 - c. Give two examples of a valid relation.
 - d. Give one example of an invalid relation.
2. Suppose a student in grade 8 came to you and asked: "What is a relational database?" Using a pencil and one piece of paper, draw a labeled sketch that answers the student. As necessary, use annotations and captions. Be creative! Be concise! Be clear!
3. Consider the following scenario, involving three tables:
Student <sname, scode>
{<fred, 10>, <mary, 20>, <martha,20>}

Course <ccode, start_time, cname>
{<math-101, 1130,"Basic Math" >, <english-101, 1400, "Basic English">}
Registration <sname, ccode>
{<fred, math-101>, <fred, english-101>, <mary, english-101>, <martha, math-101>}

 - A. Using a diagram and caption, please explain the concepts: **operand**, **operator**, and the property called **closure**.
 - B. Write a relational algebra statement to show:
 - a. Just the names of the courses.
 - b. Just the names and codes of courses that start after 1200.
 - c. The student codes (scode) of students that are registered for 'english-101'.
 - C. What are the primary and foreign keys in this example? Why are they needed?
 - D. Why would using a single table to represent the above data be a poor choice?
Registration <sname, scode, ccode, start_time, cname>

A3: Relational Algebra (2 pages)

Worth: 10%

4. Rewrite the following SQL expression in the relational algebra:

```
SELECT      x, y, z
FROM        t1, t2
WHERE       x > 10;
```

5. Explain how UNION, INTERSECTION, and DIFFERENT operators are different from the CROSS PRODUCT and JOIN operators.

6. Consider the SB Project and these two tables:

User <uid: integer, uname: string, upassword: string, email:string, website_url:string>

Resource<rid: integer, url:string, title:string, description:string>

Discuss the relationship between User and Resource. Describe an approach for representing this relationship. Now, critique your approach – I will use your critiques to help drive the development process.

Deliverable

Please hand in a concise report that answers the above questions.

Please attach your sketch in an appendix, after your answers ...

P.S.

Please don't neglect A9 – your ongoing reflections on what you are learning in this class.

Learning objectives

- Continue to develop knowledge for SQL queries (LIKE, JOIN, GROUP BY, HAVING, Aggregate functions, nested subqueries, and so on).
- Develop knowledge for creating a simple database and using the database to drive a user-interface dialog.

Getting started

- To complete these tasks, you need to use the database "usa_pop.zip", which you created in A2.

Questions

1. Write an SQL query that:
 - a. Lists all the states that have a city that begins with "West" or "South". Only show a state once in the listing.
 - b. Shows the number of cities that each state has.
 - c. Shows the average, maximum and average state population.
 - d. Shows the percentage change in population between 2000 and 1990. (Suppose X and Y are the populations for 2000 and 1990. Then, the percentage change is given by this calculation $(X - Y)/X * 100$.)
 - e. Given a state name, list all the cities in that state that have names longer than 8 characters. (Hint: See the PostgreSQL documentation and look for a "string function that computes the number of characters in a string." For documentation, see <http://www.postgresql.org/docs/7.4/static>)
 - f. Shows the city names, state code, and state name for all cities in the database.
 - g. Find the number of zip codes in states with populations between 2.5 – 5.0 million in 2000. List the state name, state code, population in 2000, and number of zip codes. Order the results by state code.
 - h. Show the statement that makes the above query a view. Show an example query on your new view. Discuss why this view might be useful.
 - i. Lists all states that exceed that exceed the average state population.
2. Propose an interesting query that one might ask of the database; then, write the SQL for it.
3. In order to correctly perform pattern matching with the LIKE operator, what must the database do? You may want to frame your answer by comparing the LIKE operator to the equality operator. In other words, how does the database need to treat these two expressions in rather different ways:
 - 1) SELECT ... WHERE A = 100;
 - 2) SELECT ... WHERE B IS LIKE '%mlis%';Discuss your answer. If something seems puzzling discuss that too. (*Hint*: You may find useful information in Appendix C.)

A4: Beginning SQL

Worth: 10%

4. Assume that we have the following tables:

Tag	(<u>tid</u> , term)
Resource	(<u>sid</u> , name, url)

Add a table that can be used to 'bridge' these two tables. Demonstrate your solution by writing a psql script that does the following:

- Defines the database, including tables, domains (if useful), and primary and secondary keys;
 - Adds some representative data to the tables;
 - Demonstrates some interesting queries.
5. Consider how the above database could be used to drive a user-interface for social bookmarking. Select three tasks that users might want to complete with a social bookmarking system.

Now, on one or more blank pieces of paper and with and with a pencil, please do the following. For each task:

- State the task concisely;
- Sketch a user-interface dialog (two or more screens) for the task;
- Show how the above database could be used to drive the user-interface.

Finally, propose a task and user-interface dialog that cannot be completed with the above database – and describe how the database would have to be modified to enable the dialog.

Deliverable

Please submit a concise report that answers these questions.

P.S.

Please remember A9 – your ongoing reflections of your learning.

Learning objectives

- Continue to develop knowledge for SQL data definition;
- Continue to analyze the social bookmarking problem.

Questions

1. Relational integrity can be enforced by various kinds of constraints, including domain, entity, referential, and enterprise constraints. Explain the different between domain and enterprise constraints. How does one implement enterprise constraints?
2. Attributes in a table must be "typed". What does it mean for an attribute to have a "type" and why is it important? Are you able to change an attribute's type? If so, show how with some SQL.
3. PostgreSQL provide many options for date/time types. Please give a brief overview of the various options for representing date/time? Reflect on why there are so many choices.
Hint: See <http://www.postgresql.org/docs/7.4/static/datatype-datetime.html>
4. Suppose we need to keep of log of all actions that are completed by a user. The log table might look like this:

```
userlog<userid, action, time, comment>
```

Assume that action is a character code, such as "a" for 'add tag'; "d" for "delete tag", and so on.

What information should be stored in the comment field?

Create two tables, one for holding users and one for the userlog table. When specifying tables, please first create domains, and include the appropriate integrity constraints. Please discuss the rationale for the choices you've made.

Hint: Explore how the scalar operator CURRENT_TIME might be helpful.

5. Explain what transactions are for? Create a short SQL script that demonstrates how they work.
6. Key functions associated with social bookmarking seem to be the ability to 'rename' tags, 'copy' tags, 'clone' tags, 'swap' tags, and so on. In addition, we've explored the notion of a 'space' in which a set of tags is applicable so one might need, for example, to copy a whole space of tags to new space which is then changed. All of this is confusing – yet, until it's made clear it will be very difficult implement a database for social tagging!

Can you clarify the meaning of the above operations and the notion of a 'tag space'? In other words, can you provide precise definitions for these concepts?

The Deliverable – the usual, please.

(P.S. Please remember A9 – your ongoing reflections of your learning)

A6: Reflections on System-Centric Methodologies

Worth: 10%

Learning objectives

- List and discuss the risks associated with information system design;
- Describe an information system design process;
- Discuss some of the challenges with using and following a design methodology;
- Develop skills for analyzing and testing the correctness of SQL.

Questions

1. Find a case study that describes an information system failure and summarize the case study in 1 page. Please succinctly describe the reasons for the failure. Please write no more than approximately one page. [*Note:* Please look beyond the first page of Google results! In other words, try to find an interesting case study that has been published in a peer reviewed journal.]
2. Consider the design methodology presented in the textbook (chapters 9 – 10). Set that methodology against the process being followed in our class and discuss the similarities and differences. Are we doing it wrong? What could we be doing better? What are we doing well? Please write no more than one page?
3. Consider the problem of “tag space”. Based on our recent discussions and your own thinking, what are the various ways to define this concept? Name and briefly describe them. Now, take two options and carryout a cost-benefit analysis by completing the following chart:

OPTION #1: xxxx

+ a pro ...
+ a pro ...
- a con ...
- a con ...

OPTION #2: xxxxx

+ a pro ...
+ a pro ...
- a con ...
- a con ...

Given your analysis, which one do you think we implement.

4. *Optional* – Consider this code:

```
SELECT u.uid, u.username
FROM sbp.submission s, sbp.tagger u, sbp.tag t
WHERE t.term in ('mlis', '542')
AND t.tid = s.tid
AND s.uid = u.uid
GROUP BY s.uid, u.uid, u.username
HAVING Count(*) > 1;
```

We’ve discussed what it is supposed to do but does it actually work? Answer this question by conducting creating test cases and trying out the code. (Hint: Starting code and data is available on the website.)

The Deliverable – the usual, please.

(P.S. Please remember A9 – your ongoing reflections of your learning)

A7: Entity-Relationship Modeling I

Worth: 10%

Learning objectives

- Use ER modeling techniques to represent relatively simple models.

Preliminaries

- Please use pencil and blank paper to answer all questions.
- Please include a brief caption for explaining any tricky parts of your solutions.
- Please aim for rigor in how you use the notation.

Questions

1. Draw an ER model for the current database schema of SBP (please see last page).
 - a. Please discuss where a many-to-many relationship is represented by two one-to-many relationships. What, for example, are the merits of showing many-to-many relationships?
2. Consider a hotel chain with different hotels in different cities. Each hotel has a different name. Hotels obviously have rooms of different types and prices. And, of course, guests check into hotels and stay for holidays and business. Please draw an ER diagram for of this situation (include no more than eight entities).
3. Suppose you wanted to model the case of multiple guests checking into the same room. A family, for example, might need to reserve a room and everyone needs to be recorded in the database because multiple keys are handed out. Please extend the model done for #2 to handle this case.
4. It might be useful to support the notion of a directory in SBP. Extend your model in 1 so that bookmarks can be organized into categories.
5. *Optional.* Answer question #6 on A5 again. Discuss the issues – are we making progress?
6. *Optional.* Extend the model created in 4 to handle user groups.

The Deliverable – the usual, please.

(P.S. Please remember A9 – your ongoing reflections of your learning)

A7: Entity-Relationship Modeling I

Worth: 10%

```
/* Social bookmarking DB - Nov 9, 2006 */
create table sbp.item (
    iid          SERIAL PRIMARY KEY,
    iposted      date,
    itype        d_item_type,
    date         varchar(64),
    author1      d_author_text,
    title        varchar(512),
    pub          varchar(512), ...);

create table sbp.sbUser (
    uid          integer PRIMARY KEY,
    uname        varchar(40),
    upassword    varchar(40),
    e_mail       varchar(256),
    website_url  varchar(512),
    website_title varchar(512)
);

create table sbp.tag (
    tid          integer PRIMARY KEY,
    term         varchar(40)
);

create table sbp.bookmark(
    uid          integer references sbp.sbUser(uid),
    iid          integer references sbp.item(iid),
    posted       date,
    title        varchar(512),
    notes        text,
    PRIMARY KEY(uid, iid)
);

create table sbp.tagbookmark (
    uid          integer references sbp.sbUser(uid),
    tid          integer references sbp.tag(tid),
    iid          integer references sbp.item(iid),
    PRIMARY KEY(uid, iid)
);
```

Learning objectives

- Use ER modeling techniques to represent relatively simple models.

Preliminaries

- Please use pencil and blank paper to answer all questions.
- Please include a brief caption for explaining any tricky parts of your solutions.
- Please aim for rigor in how you use the notation.

Questions

1. Consider this database schema (underlined attributed indicate primary keys):

```
Flight (      fid: integer;
             from: string;
             to: string;
             distance: integer;
             departs: time;
             arrives: time;
             aid: integer )

Aircraft      (      aid: integer;
                 aname: string;
                 crusingrange: integer)

Certified      (eid:integer; aid:integer)
Employee      (eid: integer; ename: string; salary: integer)
Passenger      (pid: integer; lname: string; fname: string)
Seating        (pid: integer; fid: integer; seatid: integer)
```

Using many-to-many relationships, please propose an ER model. Now, break the many-to-many relationships into two one-to-many relationships can create another ER model.

2. If you haven't already done so, please extend your 'model' for SBP to represented 'groups' (question #6 from A7). How are you defining 'group'? What different options can be used to define groups? Can you see any way to use generalization/specialization relations to help model the notion of groups?
3. Suppose you wanted to enable people to create faceted structures. That is, you would like users to create a facet that can take on one of several values. Propose a model for enabling this. Extend the model developed in #2 to accommodate this new capability.

The Deliverable – the usual, please.

LIS-542: Database Applications Development

Autumn 2006

A10: Database Design

Worth: 10%

Reflections on Database Design

Through this quarter, we have investigated the Database Design Process. In this final exercise I would like you to step back and consider the material that we have covered and write an essay. Taking a professional role, say "reference librarian", please answer the question:

What should Database Design be in the Future?

Your essay might discuss the complexity of database design and explore how that complexity might be addressed. Or, it might discuss the how various different kinds of knowledge must be brought together in a database design project. To illustrate your views you might draw upon your own projects, the Social Bookmarking Project, or create your own scenario.

Deliverable

Your essay should be approximately 2,000 +/- 500 words or so. Concept maps, pencil sketches of user interfaces, and other visual materials are much appreciated.

Please e-mail your essay or put a printed copy of your essay in my mailbox (preferred) by 5pm on Monday December 11.