CSCI 443: *Database Management Systems*
Final Exam -- open book section
29 April, 1993

The entire exam is to be turned in at 8:30PM. Work the closed book section first and turn it in before you consult your books and notes to work on the open book section.

Name: _____

Problem 1. (8 points)

Newly passed government regulations require that UNCA monitor garbage collected at all buildings. Monitoring consists of recording the weight of garbage collected at each site.

Draw an entity-relationship diagram for a database to store gathered the information. It is important that your database be able to produce summaries (monthly and yearly) of collected garbage and that it also store the dates on which the measurements were made. [It is believed that garbage output is especially high at the end of a semester, and the university wants to be able to record these differences.]

Problem 2. (16 points)

Given the following SQL table definition

```
CREATE TABLE PETS
     ( PET-ID  CHAR(10) NOT NULL UNIQUE,
       NAME    CHAR(20),
       SPECIES CHAR(5)
       WEIGHT  DECIMAL(6,2),
     ) ;
```

Subproblem 2a.

Show how to generate a list of all `PET-ID`s and `NAME`s using SQL.

Subproblem 2b.

Show how to generate a list of all snakes (`SPECIES` equal to 'SNAKE').

Subproblem 2c.

Show how to generate a list of the number of pets of each species. That is, list the number of snakes, number of cats, *etc*.

Subproblem 2d.

Show how to generate a list of all dogs that weigh less than 50 pounds.

Subproblem 2e.

Show how to generate a list of all dogs that weigh less than the weight of the average bear. [Note: There is a `AVG` operator in SQL. See the last paragraph for 510.]

Subproblem 2f.

Show how to generate a list of all cats that have the same name as any dog. [Note: This task is similar to problem 5 of homework 4.]

Problem 3. (14 points)

Consider the following table, which is being used to keep up with modules written in a multi-person programming project.

NAME	FILE	OWNER	AUTHOR	LINES
RETRIEVE	HASH.PAS	KNUTH	SMITH	105
RETRIEVE	HASH.PAS	KNUTH	WESSON	105
DELETE	HASH.PAS	JONES	JONES	15
DELETE	HASH.PAS	JONES	SMITH	15
UPDATE	HASH.PAS	CLARK	ARTHUR	1037
LIST	SORT.PAS	CLARK	CLARK	700

Subproblem 3A. (4 points)

Draw functional dependencies (pp. 220-222) for the module database. Make sure that your functional dependencies are consistent with the table and your knowledge of how multi-person programming projects are administered.

Subproblem 3B. (4 points)

Draw multivalued dependencies (pp. 223) for the module database.

Base your answers to the remaining subproblems on the dependencies you gave in subproblems 3A and 3B.

Subproblem 3C. (4 points)

Give examples of how insertion anomalies could result using a table of the form given at the start of the problem.

Subproblem 3D. (4 points)

Show how to transform this relation into a set of fourth normal form relations.

Problem 4. (6 points)

Draw a B-tree of order 2 containing 11 records.

Problem 5. (10 points)

The next version of ORACLE will support ``database triggers." Triggers are a recently introduced method of storing SQL procedures inside database servers. The database application sends the SQL procedure to the database server which executes the procedure and then returns the result to the application.

Although implementing triggers adds considerable complexity to database servers, triggers are considered a ``must have" feature in the competitive market for high-end database management systems.

Your job for the next 15 minutes is to, first, think for 10 minutes about what sort of applications would benefit from using triggers and, second, write for 5 minutes about these applications. Please don't write more than two paragraphs!

Problem 6. (12 points)

This problem has a Pascal version and a C version. You can choose which one you want to answer.

Pascal version

A file TESTDATA has been opened and contains records of the form

```
RECTYPE = RECORD
     count:  integer ;
     data:  array[1..100] of char ;
END
```

Write the Pascal code needed to read the 100'th record of TESTDATA, increment the count field by one, and write the record back into the file at the same place where you read it.

C version

A stream file (FILE *) testdata has been opened and contains records of the form

```
struct rectype {
     int count ;
     char data[100] ;
}
```

Write the C code needed to read the 100'th record of testdata, increment the count field by one, and write the record back into the file at the same place where you read it.