AD Applied Database Systems

Assignment 1. Due Friday, 22 October, 2010

Please staple your answers and put your NAME and STUDENT ID on the front page.

Unless the web site says otherwise (please check it) hand your written answers into the Informatics Teaching Office by **16:00 Friday, 22 October.** The project is due at the same time; please consult the lab website for instructions on handing in your work.

This coursework has three parts. (1) Some questions for you to think about. They do not require answers and there will be no credit for them, however trying to answer them may help with the short essay questions on the exam. (2) Some short exercises which require answers to be handed in. (3) The start of a term project which requires you to build a database and do some amount of SQL programming. *It is important that you do these on your own.* The final exam will consist partly of some short essays and partly of some "technical" questions like those in this coursework.

Things to think about and discuss in class.

1. Scientists have been, generally speaking, much slower to adopt relational database technology than business. Why do you think this is?
2. Forgetting flash memory, suppose main memory were as cheap as disk memory. Would we still need databases? If so what characteristics might change?
3. Roughly how much unused disk space is there in the Informatics Forum?
4. People, mostly scientists, are talking about keeping an exabyte (that's 10^{18} bytes) of on-line data – presumably on disk storage. Roughly how much would an exabyte weigh, how much would it cost and how much heat would it put out?

Short written answers required to the following questions.

1. Assume a database Student(Sid, Name, Degree), Course(Cid, Cname, Room, Teacher), Enrolled(Sid, Cid, Mark)
   (a) Write SQL queries for
      i. The names of the courses in which a MSc student or a BSc student are enrolled.
      Answer:
      ```
      SELECT Cname
      FROM Student, Course, Enrolled
      WHERE Student.Sid = Enrolled.Sid AND Enrolled.Cid = Course.Cid
      AND (Degree = "MSc" OR Degree = "BSc")
      ```
      ii. The names of the coursess in which a MSc student and a BSc student are enrolled.
      Answer:
      ```
      SELECT Cname
      FROM Student, Course, Enrolled
      WHERE Student.Sid = Enrolled.Sid AND Enrolled.Cid = Course.Cid
      AND Degree = "MSc" INTERSECT
      SELECT Name
      FROM Student, Course, Enrolled
      WHERE Student.Sid = Enrolled.Sid AND Enrolled.Cid = Course.Cid
      AND Degree = "BSc"
      ```
   (b) Translate the following relational algebra queries into SQL. Where possible, make use of those features of SQL that are designed to make relational queries more understandable,

i. $\pi_{\mathsf{Room,Teacher}}(\mathsf{Course})$
Answer:
```
SELECT (DISTINCT) Room, Teacher
FROM Student
```

ii. $\pi_{\mathsf{Sid}}(\mathsf{Enrolled}) - \pi_{\mathsf{Sid}}(\sigma_{\mathsf{Marks}<\mathsf{M1}}(\mathsf{Enrolled} \bowtie \pi_{\mathsf{M1}}(\rho_{\mathsf{Marks}\to\mathsf{M1}}(\mathsf{Enrolled}))))$
Recall that ρ is the rename operator.
Answer:
```
SELECT Sid
FROM Enrolled
WHERE Sid NOT IN (
   SELECT Sid FROM Enrolled
   WHERE Mark < ANY (SELECT Mark FROM Enrolled))
```

iii. $\pi_{\mathsf{Name}}(\sigma_{\mathsf{Mark}>70}(\mathsf{Student} \bowtie \mathsf{Course} \bowtie \mathsf{Enrolled}))$ (Do *not* use SQL's natural join operator.)
Answer:
```
SELECT Name
FROM Student, Course, Enrolled
WHERE Student.Sid = Enrolled.Sid AND Enrolled.Cid = Course.Cid
AND Mark > 70
```

(c) Suppose that Enrolled contains 1034 tuples and that Courses contains 7 tuples. What is the smallest number of Student tuples for which such a database would make sense.

Answer: 148. The number of Enrolled tuples cannot be larger than the product of the numbers of Student and Course tuples.

(d) Suppose that Student contains 500 tuples, what is the smallest number of tuples produced by the following query:
```
SELECT   Sid, Name
FROM     Student,Course
```

Answer: Zero – when there are no Course tuples.

2. Given tables $R(\underline{A}, B), R'(\underline{A}, B), S(\underline{B}, C), T(\underline{C}, A)$, for each of the following statetments say whether it is true or false. If it is false, provide an instance of the relevant tables showing that it is false.

   (a) A is a key for $R \cap R'$
   Answer: True

   (b) A is a key for $R \cup R'$
   Answer: False: $R = \{(1,2)\}, R' = \{(1,3)\}$

   (c) A is a key for $R \bowtie S$
   Answer: True

   (d) C is a key for $S \bowtie T$
   Answer: False. $S = \{(1,3),(2,3)\}, T = \{(3,4)\}$

   (e) C is a key for $R \bowtie S \bowtie T$
   Answer: True

3. Consider the following pairs of queries (same schema as question 1). Do they *always* give the same results? If not, explain how and when they would differ

   (a)
   ```
   SELECT  Room                    SELECT  Room
   FROM    Course                  FROM    Course
   WHERE   Teacher = 'Sneezy'      WHERE   Teacher = 'Sneezy'
   OR      Teacher = 'Grumpy'      UNION
                                   SELECT  Room
                                   FROM    Course
                                   WHERE   Teacher = 'Grumpy'
   ```
   Answer: They will always give the same result. (Not true if we had used UNION ALL)

   (b)
   ```
   SELECT  Name, Degree        SELECT  Name, Degree
   FROM    Student             FROM    Student, Course
   ```
   Answer: They will differ. Suppose Student is non-empty but Course is empty. The first query will have a non-empty output, but the second will produce an empty output.

4. Explain why SQL will object to the following query:
   ```
   SELECT      Name, Cid, AVG(Mark)
   FROM        Student, Enrolled
   WHERE       Student.Sid = Enrolled.Sid
   GROUP BY    Student.Name
   ```
   Answer: Cid will vary with Student.Name. We'd need to apply some aggregating function such as COUNT(Cid) to get the query to work.

5. Again using the schema of question 1, write a query to find the names of the students that are taking exactly 2 courses

   (a) in SQL, and
   Answer: SELECT Name FROM Student S
   WHERE COUNT(SELECT * FROM Enrolled E WHERE E.Sid = S.Sid) = 2

   (b) in relational algebra
   Answer: Use $E(S, C)$ as a shorthand for $\pi_{\mathsf{Sid},\mathsf{Cid}}(\mathsf{Enrolled})$ Now consider

$$
\begin{aligned}
E_1 &= \pi_S(E) \\
E_2 &= \pi_S(\sigma_{C \neq C'}(E \bowtie \rho_{C \to C'}(E))) \\
E_3 &= \pi_S(\sigma_{C \neq C' \wedge C' \neq C'' \wedge C'' \neq C}(E \bowtie \rho_{C \to C'}(E) \bowtie \rho_{C \to C''}(E)))
\end{aligned}
$$

E_1 gives the (ids of) students taking one or more courses; E_2 gives the students taking two or more; and E_3 gives the students taking three or more.

So $E_2 \setminus E_3$ gives the students taking exactly two courses. Finally do a join with the Students table to get theor names.

For the relational algebra question, be sure to decompose the query into small intermediate steps with a description and name for each intermediate table.

Project Part - Assignment 1[40pt]

You are starting a new job as a database administrator in a medical research institute. Your new employer has recently decided to redesign their existing genome database. The data for populating your new genome database will be provided by a partner institute that agreed to provide you with a dump of their own data they collected from the Web. The schema of the dumped database is given by the following DDL statements and brief field descriptions:

```
CREATE TABLE  entry(
              primacc      char(6) NOT NULL,
              name         varchar(11) NOT NULL,
              species_sci  varchar(80) NOT NULL,
              species_com  varchar(80),
              seqlength    int NOT NULL,
              sequence     varchar(400) NOT NULL,
              PRIMARY KEY(primacc));
```

Table entry contains exactly one tuple for each protein (gene) found in the current UniProt Knowledgebase (www.uniprot.org/uniprot/). Each entry has a unique primary accession number, a name, the scientific name of the species the gene is taken from, an optional common species name, the amino acid sequence, and a field indication the length of the amino acid sequence.

```
CREATE TABLE  comment(
              primacc   char(6) NOT NULL,
              type      varchar(20) NOT NULL,
              text      varchar(255) NOT NULL);
```

Table comment associates gene entries in the entry table with three different types of comments: SUBCELLULAR LOCATION indicates the location a gene was found in, CATALYTIC ACTIVITY links an entry with a catalytic activity, and DISEASE describes a disease in which an entry is involved in. The information is this table is extracted from the CC-lines of UniProt entries.

```
CREATE TABLE  feature(
              primacc      char(6) NOT NULL,
              type         varchar(20) NOT NULL,
              from_pos     int NOT NULL,
              to_pos       int NOT NULL,
              description  varchar(255));
```

Table feature represents sequence features. Each feature has a type, and a start and end position on the gene sequence. The information in this table is extracted from the FT-lines of UniProt entries.

```
CREATE TABLE  dbxref(
              primacc  char(6) NOT NULL,
              dbname   varchar(10) NOT NULL,
              refacc   varchar(10) NOT NULL,
              comment  varchar(128));
```

Table dbxref lists cross-references between entries in UniProt and other databases. Each tuple contains the

name of the referenced database and the primary accession number of the referenced entry. Note that this primary accession number is different from the accession number of UniProt entries. Each cross reference may contain an additional comment.

```
CREATE TABLE  go_term(
              id        int NOT NULL,
              name      varchar(200) NOT NULL,
              type      varchar(20),
              primacc   varchar(40) NOT NULL,
              obsolete  smallint NOT NULL,
              PRIMARY KEY(id));
```

Table go_term contains a comprehensive list of terms from the Gene Ontology. Each term has a unique identifier, a name (or description), a type (e.g., biological process), a primary accession, and a flag indicating whether the term is obsolete or not. The data in this table corresponds to table term in the Gene Ontology database schema:
(www.geneontology.org/GO.database.schema.shtml).

```
CREATE TABLE  omim(
              omimid  char(6) NOT NULL,
              title   varchar(180) NOT NULL);
```

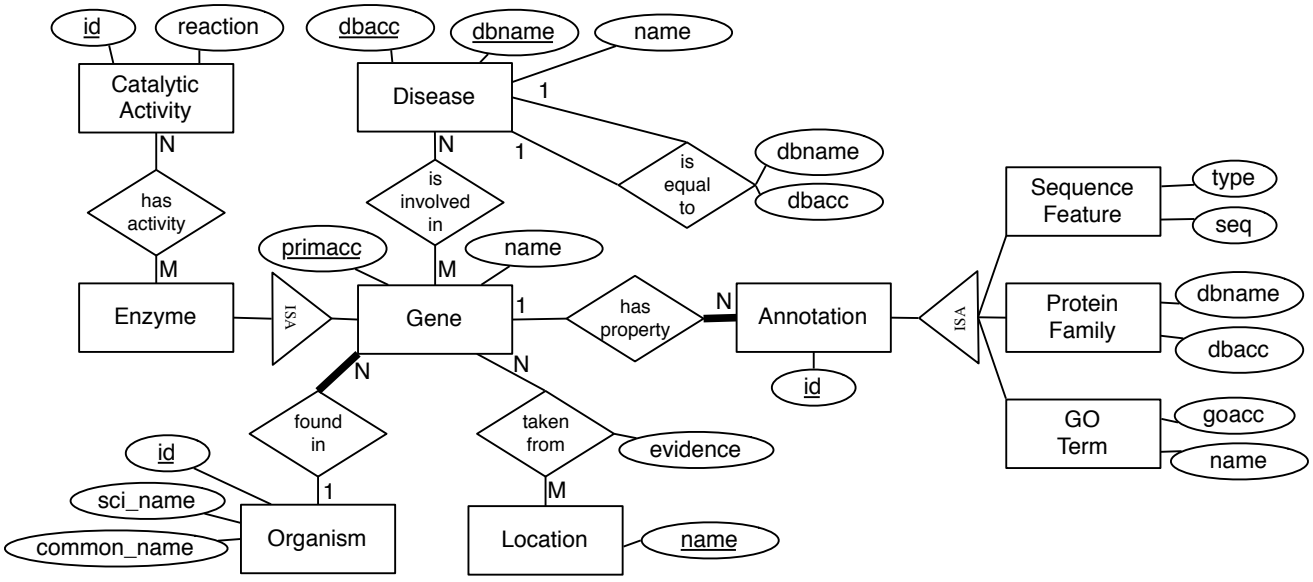
Table omim contains a list of diseases in the Mendelian Inheritance in Man Database. For each disease the unique identifier and the disease name (description) is listed.

For each table a load file (containing the raw data) is provided at

http://www.inf.ed.ac.uk/teaching/courses/ad/project/data/data.html

The order of columns in the load files corresponds to their order in the above DDL statements. Columns in the load files are separated by '—'. Null Values are represented by the text string "null".

However, your job is to redesign this database and after a requirement analysis has been conducted, you are provided with the following ER diagram:



The following is a brief description of the identified entities and relationships:

- Genes are the central entity of the ER schema. Each gene is uniquely identified by a primary accession number and it has an additional name.

- A gene is taken from exactly one organism.
- Each organism has a unique identifier, a scientific name, and an optional common name.
- A gene may by found in different sub-cellular locations. Locations are uniquely identified by their name.
- For the occurrence of a gene in a sub-cellular location, an optional non-experimental qualifier indicates that the information given is not based on experimental findings. The three types of qualifiers are: Potential, Probable, and By similarity.
- Each gene is associated with a set of annotations describing different properties of the gene and its sequence. There are three classes of annotations:
  - Sequence features classify regions or sites of interest in the amino acid sequence of a gene. Each such annotation is composed of a class label or type and the classified substring of the protein sequence. An example sequence feature is MOTIF, indicating a short sequence motif of biological interest.
  - A gene may be annotated with a set of Gene Ontology terms. The Gene Ontology provides a controlled vocabulary of terms for describing gene characteristics. In general, there are three different classes of Gene Ontology terms, describing cellular components, biological processes, and molecular functions a gene is involved in. Gene Ontology terms have a unique identifier and a textual description.
  - Each gene may also belong to several different protein families. Each family is represented by the name and the primary identifier of a protein family database.
- Genes that catalyze chemical reactions are called enzymes. Enzymes can have more than one catalytic activity.
- The chemical reactions catalyzed by enzymes are given unique numerical identifiers as primary keys.
- Genes may be involved in the development of diseases. We call such genes disease genes. Disease genes are of primary interest for medical research and drug design. Each disease has a name and is uniquely identified by the name and the primary accession number of an entry in one of two disease databases, namely the Mendelian Inheritance in Man Database (MIM) (www.ncbi.nlm.nih.gov/omim/) and Orphanet, a database dedicated to information on rare diseases and orphan drugs (www.orpha.net/consor/cgi-bin/home.php). Correspondences between disease that appear in both databases are represented by links between their primary keys.

The following results are asked of you in this assignment:

1. Transform the given ER schema into a relational database schema. Provide a file with the DDL statements to create tables and specify column constraints and foreign key relationships.
2. Create a local copy of the provided genome database dump in your PostgreSQL database.
3. Answer the following questions (provide a file containing the queries and the resulting output).
   (a) How many entries have a ''CATALYTIC ACTIVITY"?
   (b) How many different catalytic activities are there?
   (c) List the entries having a DISEASE comment and i) a CATALYTIC ACTIVITY comment, ii) NO CATALYTIC ACTIVITY comment.
   (d) List the species' common name for those species that have more than one scientific name.

(e) List databases referenced by cross-references together with the number of references to them (in decreasing order).

(f) List the entry with the most i) features, ii) cross-references, iii) both (sum).

(g) How many GO references in dbxref are made to obsolete GO terms?

(h) How long is the longest sequence feature?

(i) Which sequence feature is most common for disease genes?