Assignment 4: Step 3 of Your PDA (Personal Database Application)

Note 1: This assignment is a slight modification of material developed by the Stanford Database Group.

Note 2: Remember to back up your work!

In this phase of your PDA you will make sure your relations have keys and foreign keys, create a substantial amount of data for you databases, and load your database with this data using the MySQL "load" command. First, make sure that your relations have keys and that the relations created from ER model relationships use foreign keys to specify the keys of the relations you are referencing. To create the data, write a program in any programming language you like that creates large files of records in a format acceptable to the load command, then load the data into your PDA relations. If you are using real data for your PDA, your program will need to transform the data into files of records conforming to your PDA schema. The rest of you will write a program to *fabricate* data: Your program will generate either random or nonrandom (e.g., sequential) records conforming to your schema. It is both fine and expected for your data values—strings especially—to be meaningless gibberish. The point of generating large amounts of data is so that you can experiment with a database of realistic size, rather than the small "toy" databases often used in classes. The data you generate and load should be on the order of

- At least one relations with tens of thousands of tuples
- At least two additional relations with thousands of tuples

If the semantics of your application include relations that are expected to be relatively small (e.g., schools within a university), it is fine to use some small relations, but please ensure that you have relations of the sizes prescribed above as well. When writing a program to fabricate data, keep in mind the following two points:

1. Make sure you do not generate duplicate values for attributes that serve as keys.

2. Your PDA almost certainly includes relations that are expected to join with each other. For example, you may have a STUDENT relation with attribute courseNo that's expected to join with attribute number in relation COURSES. In generating data, be sure to generate values that actually do join—otherwise, all of your interesting queries will have empty results! One way to guarantee joinability is to generate the values in one relation, then use the generated values in one relation to select joining values for the other relation. For example, you could generate course numbers first (either sequentially or randomly), then use these numbers to fill in the courseNo values in the STUDENT relation.

To help you generate data, I am supplying the following Python file that will generate datasets for the Sailors database described below. Turn in the following:

1. Your program code for generating or transforming data and a sample (small, not the entire file) data file and a sample control file. You can contact the TA if you are having trouble generating data.

2. The output of: 1) "show tables"; 2) a "describe table_name" for each table "table_name"; 3) a count(*) on each of your tables; and 4) the output of a select command on each table that returns less than 10 tuples. You can cut and paste or use "typescript." The output would look something the sample output below. Please upload a .pdf file to the LMS.

NOTE: You do not need to use Python! You can use any language to generate your data files. I am just supplying a Python file because that is the language most of you are most comfortable with.

HINTS
For all of these hints, I am assuming my database scheme is
Sailors(sid,sname,age,rating)
Boats(bid,name,ratingNeeded,bcolor)
Reserve(sid,bid,rdate)

Use the "load" command to load your data files. Here is an example of load commands used to load the relations:

```
load data local infile '/Users/leut/data_sailors' into table sailors
  fields terminated by ','
  lines terminated by '\n'
  ;

load data local infile '/Users/leut/data_boats' into table boats
  fields terminated by ','
  lines terminated by '\n'
  ;

load data local infile '/Users/leut/data_reserve' into table reserve
  fields terminated by ','
  lines terminated by '\n'
  ;
```

Note, I load sailors first, then boats, and then reserve. The order is important. The reason is that sid in reserve is a foreign key referencing sid in sailors, and for the tuple to insert, that sid value must exist in sailors—similar for bid in boats.

Similarly, when I want to call "drop table," I need to drop reserve first and then boats or sailors. If one tries to drop sailors first, it will cause an error because reserve tuples depend on the sailor tuples.

Notice the word "local" in the load commands. Without the word "local," the load will stop if there is an error loading one or more tuples. With the word "local," the load will continue and just ignore the offending tuple. The way I generated the reserve dataset does NOT guarantee that the primary key values will be unique. Thus, by using "local" I can just drop that tuple from the dataset. Note, it is not a reasonable option for "real" data to just ignore a tuple!

In order to use the load command, the data files need to be in the same directory as the database or, as in this case, they need to have permissions set to be readable.

Don't forget to **save a copy of your PDA** for reference as you do Step 4 of the PDA.

Sample Output:

```
mysql> use Sailors2 ;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A.

Database changed
mysql> show tables ;
+--------------------+
| Tables_in_sailors2 |
+--------------------+
| Boats              |
| Reserve            |
| Sailors            |
+--------------------+
3 rows in set (0.00 sec)

mysql> select count*[K(*) from Boats ;
+----------+
| count(*) |
+----------+
|     1000 |
+----------+
```

```
1 row in set (0.01 sec)

mysql> select count(*) from Reserve ;
+----------+
| count(*) |
+----------+
|    29999 |
+----------+
1 row in set (0.01 sec)

mysql> select count(*) from Sailors ;
+----------+
| count(*) |
+----------+
|     3000 |
+----------+
1 row in set (0.00 sec)

mysql> describe Boats ;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| bid          | int(11)     | NO   | PRI | 0       |       |
| name         | varchar(20) | YES  |     | NULL    |       |
| ratingNeeded | int(11)     | YES  |     | NULL    |       |
| bcolor       | varchar(20) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)

mysql> describe Sailors ;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| sid    | int(11)     | NO   | PRI | 0       |       |
| name   | varchar(20) | YES  |     | NULL    |       |
| age    | int(11)     | YES  |     | NULL    |       |
| rating | int(11)     | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> describe Reserve ;
+-------+---------+------+-----+------------+-------+
| Field | Type    | Null | Key | Default    | Extra |
+-------+---------+------+-----+------------+-------+
| bid   | int(11) | NO   | PRI | 0          |       |
| sid   | int(11) | NO   | PRI | 0          |       |
```

```
| rdate | date    | NO   | PRI | 0000-00-00 |       |
+-------+---------+------+-----+------------+-------+
3 rows in set (0.01 sec)

mysql> select * from sailors where sid < 5 ;
+-----+------+------+--------+
| sid | name | age  | rating |
+-----+------+------+--------+
|   1 | Bob1 |    9 |      4 |
|   2 | Bob2 |    8 |      1 |
|   3 | Bob3 |    2 |      2 |
|   4 | Bob4 |    2 |      7 |
+-----+------+------+--------+
4 rows in set (0.01 sec)

mysql> select * from boats where bid < 6 ;
+-----+--------+--------------+--------+
| bid | name   | ratingNeeded | bcolor |
+-----+--------+--------------+--------+
|   1 | guppy1 |            3 | red1   |
|   2 | guppy2 |           10 | red2   |
|   3 | guppy3 |            3 | red3   |
|   4 | guppy4 |            4 | red4   |
|   5 | guppy5 |            4 | red5   |
+-----+--------+--------------+--------+
5 rows in set (0.00 sec)

mysql> select * from reserve where sid = 1 ;
+-----+-----+------------+
| bid | sid | rdate      |
+-----+-----+------------+
|  53 |   1 | 2013-11-26 |
| 104 |   1 | 2013-03-26 |
| 356 |   1 | 2013-03-06 |
| 400 |   1 | 2013-01-03 |
| 437 |   1 | 2013-04-16 |
| 507 |   1 | 2013-12-27 |
| 678 |   1 | 2013-10-02 |
| 787 |   1 | 2013-07-08 |
| 828 |   1 | 2013-02-15 |
| 985 |   1 | 2013-12-15 |
+-----+-----+------------+
10 rows in set (0.01 sec)

mysql> quit ;
```