

PROJECT PHASE II GUIDELINE

Phase II consists of the relational schema, create table statements, sets of SQL statements for tasks in IFD.

1. COVER PAGE:

You should fill the group member information into this cover page (http://www-static.cc.gatech.edu/classes/AY2008/cs4400_fall/coverpage.pdf) and attach it to the project. We assume that the group members are not changed from phase I. You need to email the professor and your TA if there is a change in your group. No change is allowed during phase III.

2. ER-DIAGRAM:

The solution for ER-Diagram from the class website can be used for phase II. You need to indicate in the project which ER-Diagram is used for phase II. If your own solution is used, the revised ER-Diagram from phase I has to be included.

3. CREATE TABLE STATEMENTS:

This part contains the create table SQL statements for all the tables in the relational schema. All the integrity constraints, where applicable (primary key, foreign key and null constraint), must be included.

4. SQL STATEMENTS:

This part consists of the sets of SQL statements to perform all tasks in the IFD and Task Decomposition posted on the class website. **The tasks should be numbered according to the task numbers in the IFD solution.**

The format of this part is as follows:

TASK 1: User Login

<Set of SQL Statements>

TASK 2a:

<Set of SQL Statements>

Each task will be associated with one GUI. Given a task, multiple SQL statements may be necessary in order to generate the output for the GUI related to the task. In this case, the output of the last SQL statement in the task should be ready to display in the GUI where applicable, i.e. For examples, derived values, records ordered by id, record contains all the attributes (in order as in the GUIs) to fill in the GUIs, etc.

For long SQL statements, you may want to break those statements into shorter ones by using views so that they become more readable.

The SQL statements should be able to HANDLE SPECIAL CASES. For examples, the total of hours is zero, the total number of members is zero.

HINTS

Group by, views, outer join are useful. In some cases, left/right outer join instead of inner join will give correct results. A view can be defined and shared by multiple tasks. It also makes the query easier. For example, a view can be used to count the number of late payments for all users. Another view can be used to render the expiration date. Two views can be used to render the final information that consists of #late and expiration date.

In this project, the SQL statements can be written much easier if the payment entity is associated with the lease entity instead of the tenant entity.