

Ex 6.6.1: This and the next exercises involve certain programs that operate on the two relations

Product (maker, model, type)

PC (model, speed, ram, hd, price)

from our running PC exercise. Sketch the following programs, including SQL statements and work done in a conventional language. Do not forget to issue BEGIN TRANSACTION, COMMIT, and ROLLBACK statements at the proper times and to tell the system your transactions are read-only if they are.

- a) Given a speed and amount of RAM (as arguments of the function), look up the PC's with that speed and RAM, printing the model number and price of each.
- b) Given a model number, delete the tuple for that model from both PC and Product.
- c) Given a model number, decrease the price of that model PC by \$100.
- d) Given a maker, model number, processor speed, RAM size, hard-disk size, and price, check that there is no product with that model. If there is such a model, print an error message for the user. If no such model existed in the database, enter the information about that model into the PC and Product tables.

Ex 6.6.2: For each of the programs of Ex 6.6.1, discuss the atomicity problems, if any, that could occur should the system crash in the middle of an execution of the program.

Ex 6.6.3: Suppose we execute as a transaction T one of the four programs of Exercise 6.6.1, while other transactions that are executions of the same or a different one of the four programs may also be executing at about the same time. What behaviors of transaction T may be observed if all the transactions run with isolation level READ UNCOMMITTED that would not be possible if they all ran with isolation level SERIALIZABLE ? Consider separately the case that T is any of the programs (a) through (d) of Exercise 6.6.1.