Name 1: Date:
Name 2:

We have two relations, Products(maker, model, type) and PC(model, speed, ram, hd, price). Consider the following transactions. Assume that the transactions might abort.

a) Given a speed and amount of RAM (as arguments to the function) look up the PC 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, hd size and price, check that there is no product with that model. If there is, print an error. If no such model exists, insert it into the PC and product tables.

1) If only transactions **a** and **b** are allowed to run in our database. What is its minimal isolation level for each transaction without compromising the integrity of the database? What is the impact of this isolation level on the user running each transactions? Assume multiple instances of each transaction can occur.

2) If only **c** and **d** are allowed to run, what is the impact (for the user running the transaction and for the integrity of the database) of the 4 different isolation levels on transaction **d**? Assume c runs serializable.

1) a & b are allowed to run
a is Read only, so it will never compromise the database. We can run it READ UNCOMMITTED.
For user, it might print a model that is being deleted, or one that is being added and then trans. aborts, disappearing (phantom)

a will never interfere with another a

b can compromise the database? NO

It only deletes a tple, which is not a potential consistency problem: Run READ UNCOMMITTED

For user: Worst it can happen

1) 2 transr. try to delete the same tple
One will succeed.

2) 2 try to delete same tple.
T_1 deletes it
T_2 tries, but doesn't find it
T_1 abortr.
$\Rightarrow T_2$ didn't find it.

---

2) T_c decreases price by 100
T_d insert new model.

First assume T_c never aborts

READ UNCOMMITTED:

User's view:
- T_c can decrease the price of a part being added, but appears serializable.
- T_d can try to add the same model, but only one will succeed.

This does not affect consistency

If T_c or T_d can abort.

T_c nor T_d affect consistency.

T_d aborts T_c modifies price of a part that was never really added.

T_c aborts, nothing really happens to other transactions.

READ COMMITTED and above:

Transactions are serializable.

They don't affect consistency.