# Assignment 6 – Table Population (2)

With the “easy” tables, completed, we now turn our heads to the tables with dependencies. We will add to the script from last week to populate the last of our tables.

There are two ways to deal with dependencies. The **wrong** way is this:

SELECT stateId FROM state WHERE stateName = ‘NY’

(returns 22)

INSERT INTO city VALUES (‘Albany’,22)

Why is this wrong? It uses human intervention to copy from the select into the insert. The “22” will also change between script runs. It is unsafe and it makes your script very fragile. It is sort of like writing a pointer address down in C and re-using it.

The correct approach:

INSERT INTO city SELECT ‘Albany’,stateId FROM state WHERE stateName = ‘NY’

What happens here?

Select runs first; it creates one row, containing the string ‘Albany’ and the stateId for ‘NY’. Insert then runs and takes that one row and inserts it into city.

Notice that the correct approach is safer, shorter, and easier!

What if I have multiple dependencies? We must change our select. We must be careful to make sure that the select only selects the combination we have in mind.

INSERT INTO city SELECT ‘Albany’, stateId, countyId FROM state, county WHERE stateName=’NY and countyName=’Albany’

**Make sure that your script uses your database (USE mydbName)**

**Your script must have a .sql extension and must be uploaded to blackboard. A script that doesn’t run is an automatic 0.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rubric | Poor | OK | Good | Great |
| Data Quantity | 0 average rows | 1-10 average rows(20) | 11-19 average rows(35) | 20+ average rows (50) |
| Tables Selected | No tables (0) | 1-60% of dependency tables(20) | 61-90% of dependency tables (35) | 91-100% of dependency tables (50) |