Rahul Rajagopalan
6.170

# Adventure Game Schema Design Notes

As described in lecture, object models can be translated into relational database schema via a fairly mechanical process. Relations between objects can be translated into foreign key columns if they are one-to-one or one-to-many. Many-to-many relations require additional work, but there were no many-to-many relations in my model (because each player is operating in their own "world" with its own set of rooms, exits, and items).

The primary open question in translation was deciding how to model the subset relations in the database. There were some of these in the model. Items, Rooms, and Exits were classified as exhaustive subsets of the abstract GameEntity set. Item is also an abstract set, and its exhaustive subsets were item-unlocking Items and exit-unlocking Items, based on how each individual item contributed to the player's progress in the game.

GameEntity was not represented as a table; a table was made for all of its subsets. This did requre some duplication of column names; all GameEntities have ids, names, and descriptions. However, representing it as a table would require additional joins, which might have had a performance penalty.

In the case of items, the Item superset was represented as a table, and there were also tables for the subsets (3 tables in total). The items table held information common to all items; this held all the columns except the unlocks_* column, which was specific to the subsets' table. While the column is an integer in both cases, it is not really suitable to make a "polymorphic" items table, as the integer is a key that refers to a different table depending on the type of item. That is, unlocks_item in item_unlocking_items refers to rows in the items table, and unlocks_exit in exit_unlocking_items refers to rows in the exits table.

I wouldn't consider this design to be significantly better or worse than the other methods available for modeling subsets in a relational database. The ones described in class all have their individual advantages and disadvantages.

The schema can be found in code form in schema.sql.