SE3309a - Assignment Three

Samuel Mallabone - 250844429

Robert Northmore - 250838145

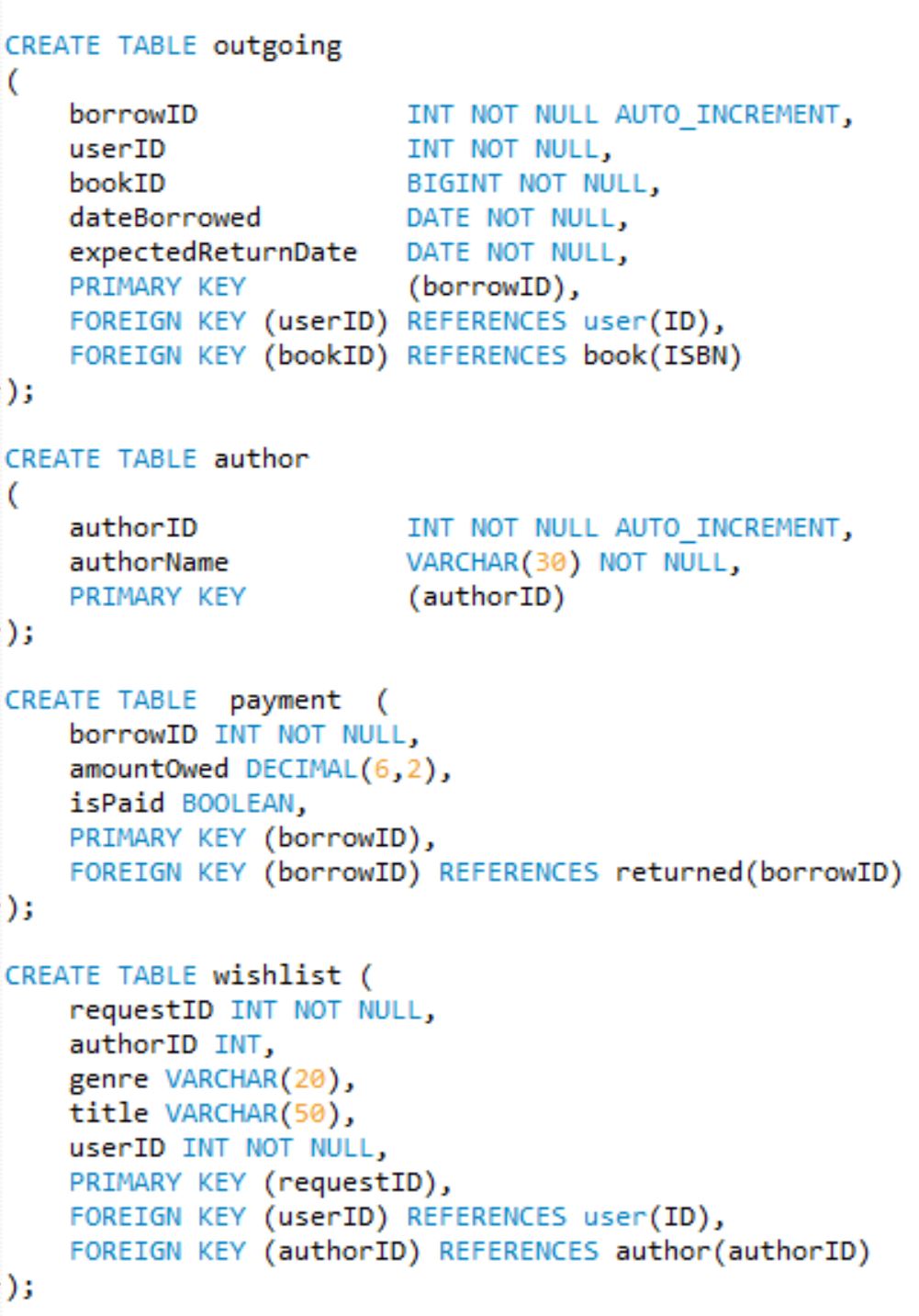
Jak Terpak - 250846924

Craig Cook - 250866685

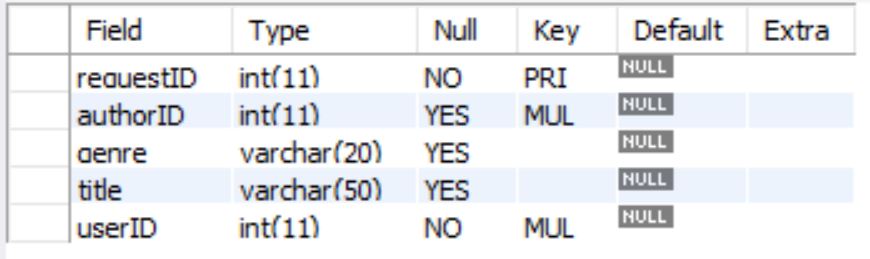
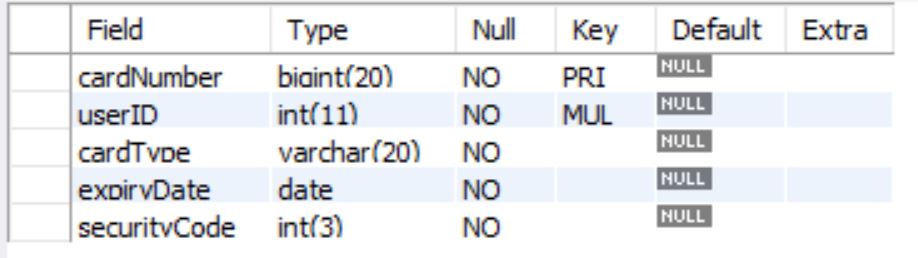
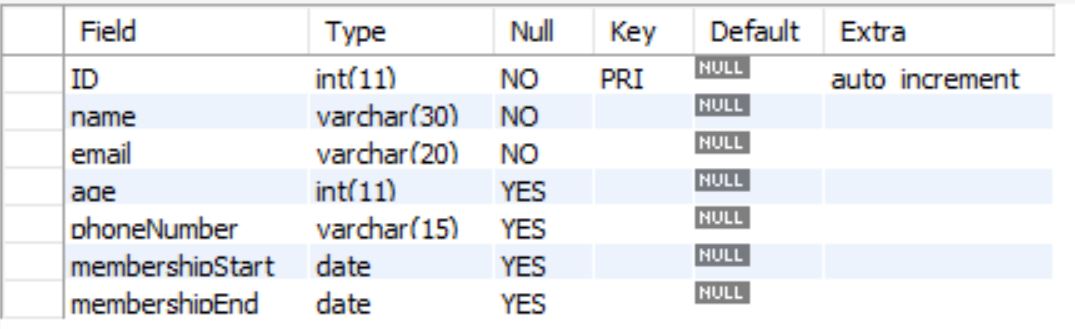
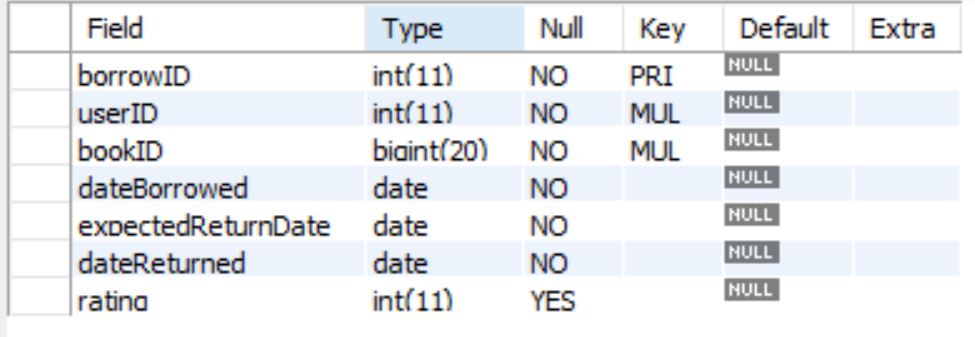
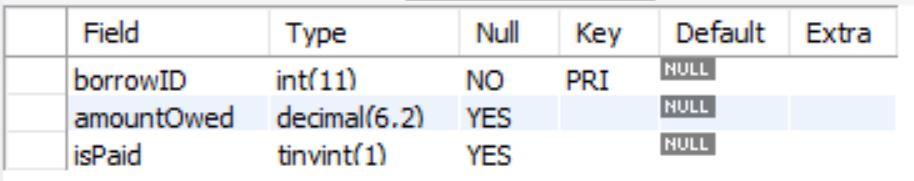
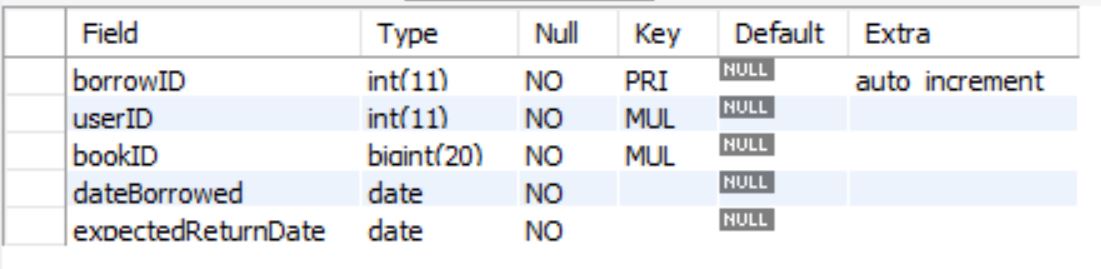
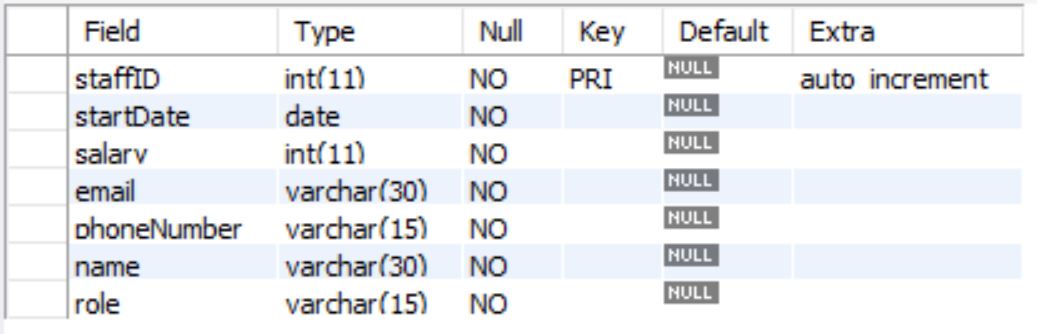
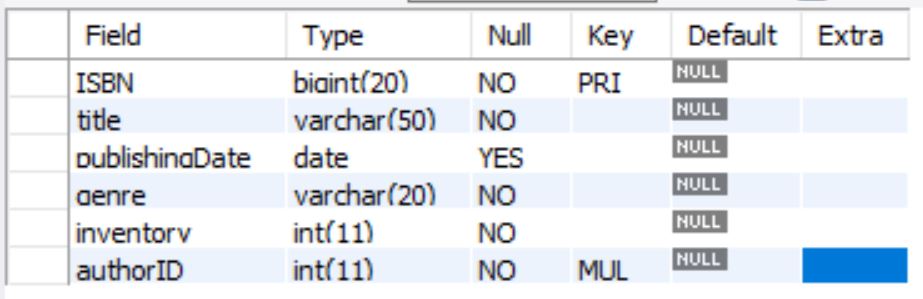
Note: We also created a Github repository name SE3309Databases and included all the TA’s in it. Github link: <https://github.com/sam-mallabone/SE3309Databases> (If this link doesn’t work, the Github Repository’s name is SE3309ADatabases and my username is sam-mallabone, here is a clonable link for the repository too: https://github.com/sam-mallabone/SE3309Databases.git)

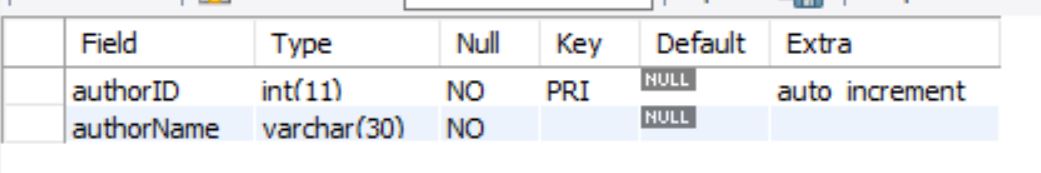
Question 2: Creating Tables:

*The top three screenshots show the commands to create the tables. All these commands ran properly and created the tables as we intended. Below the create table statements, we used the describe <table\_name> command to show that the tables were successfully created and had the schema we intended.*



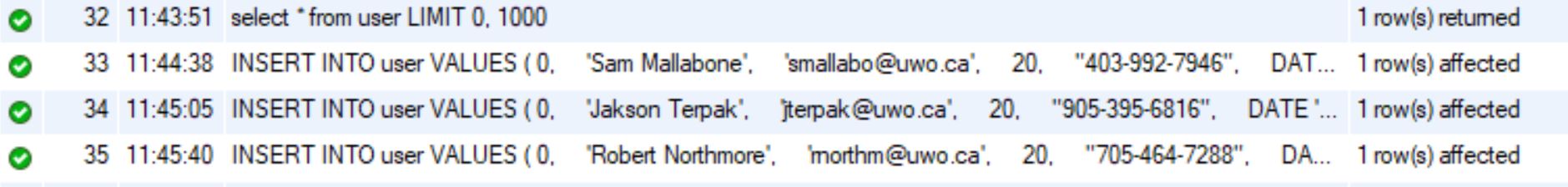
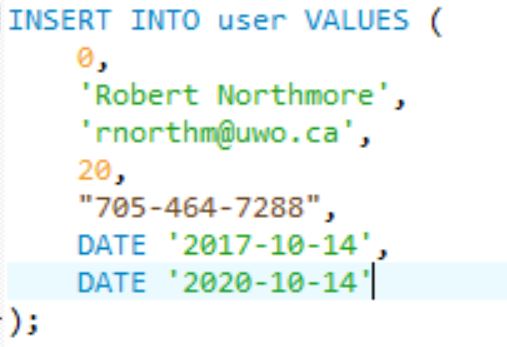
*This section is when we were running the describe <table\_name> commands*

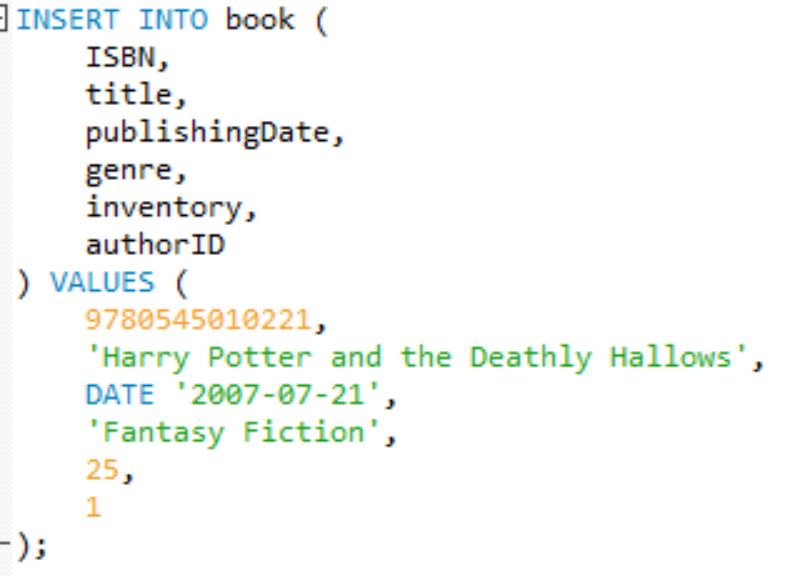


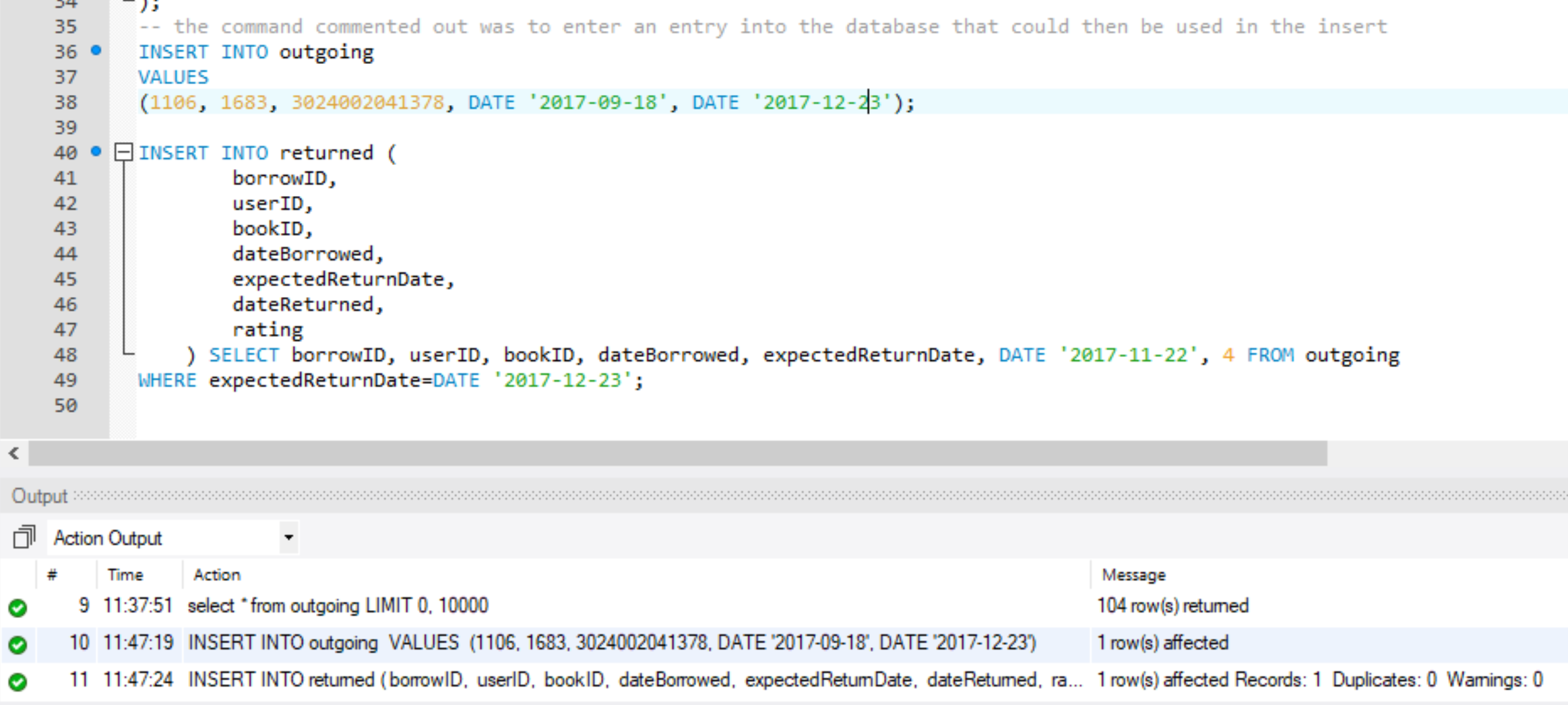


Question 3: Three different types of Insert

*Below shows the three different insert commands for entering data into tables. All three of these different types of commands worked for us and the screenshots show both the command and the output. Instead of running the select \* statement for the table to show that the values had been entered, we showed the success statement of the output because we thought this would exemplify that the command worked as intended.*





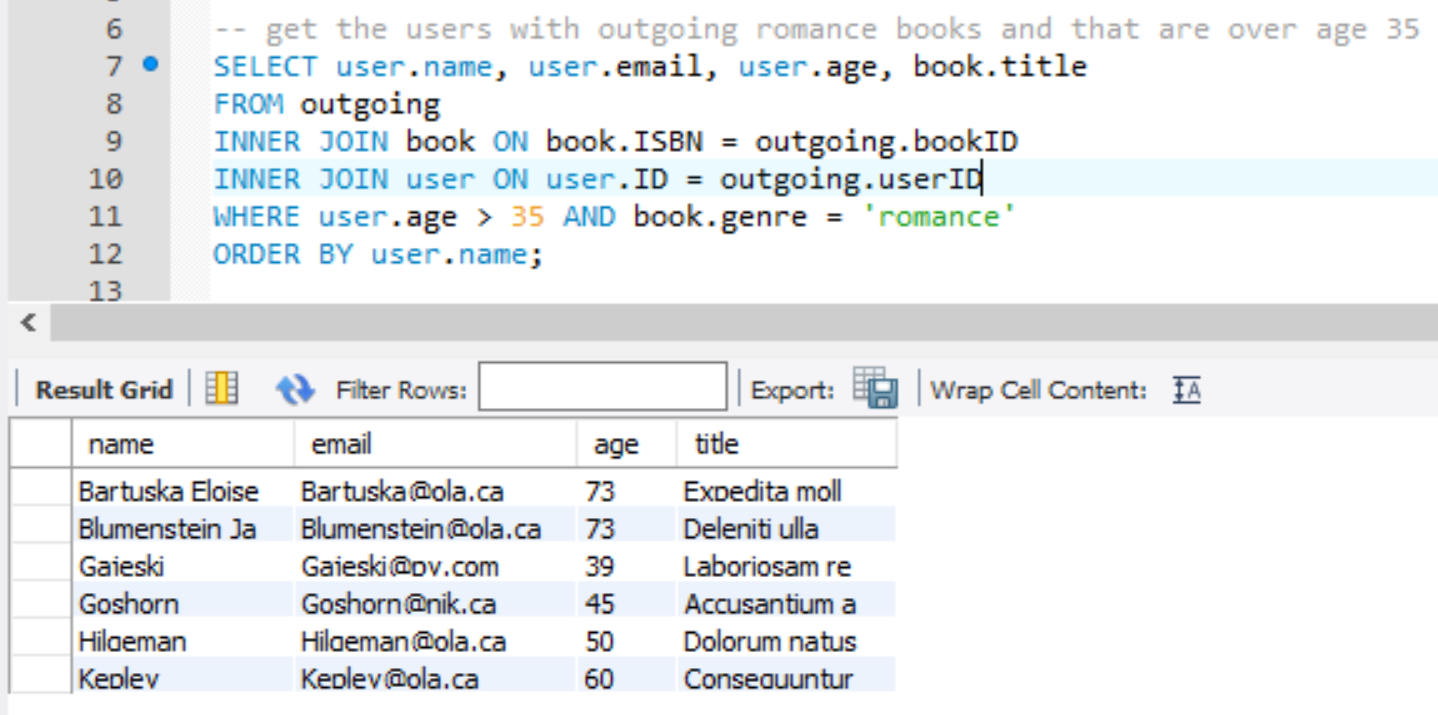


Question 4: Inserting data into the tuples.

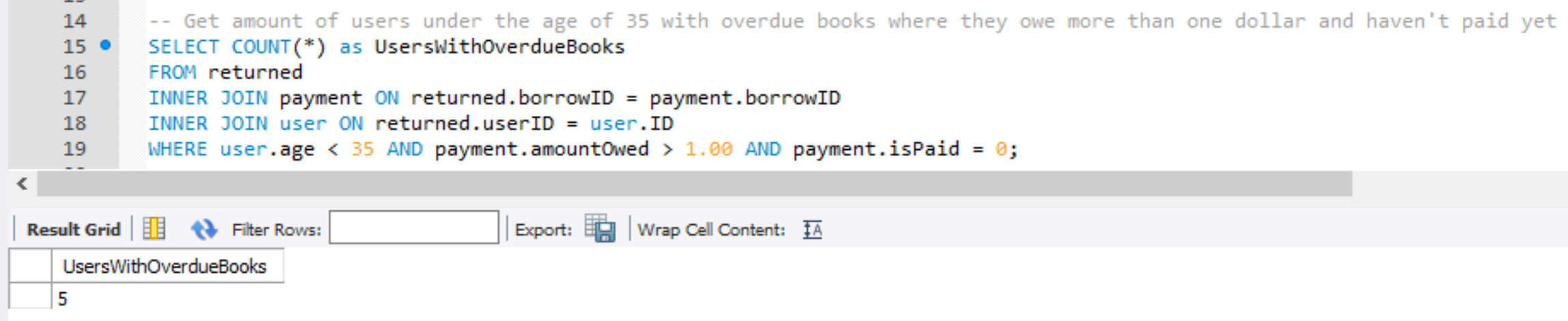
*To insert our data into the tuples, we used python (the files name is insertmysql.py). This python file would run the script that enters data into the database. We ensured to create the tables in such a way that we could use existing primary keys from a created table for the foreign keys on another table. This ensured that we could properly use all the joins and that our queries would run as expected. In the python file a lot of the code is commented out and in weird order. We ran several sections of the code to insert data which accounts for the seemingly random ordering of the code. It worked perfectly and we were able to enter over 5000 tuples into two tables, several hundred into another table and over one hundred tuples the remaining tables. To get our data, we used a combination of csv files containing names as well as a python library called Faker which allowed us to generate random names and random text which we took advantage of.*

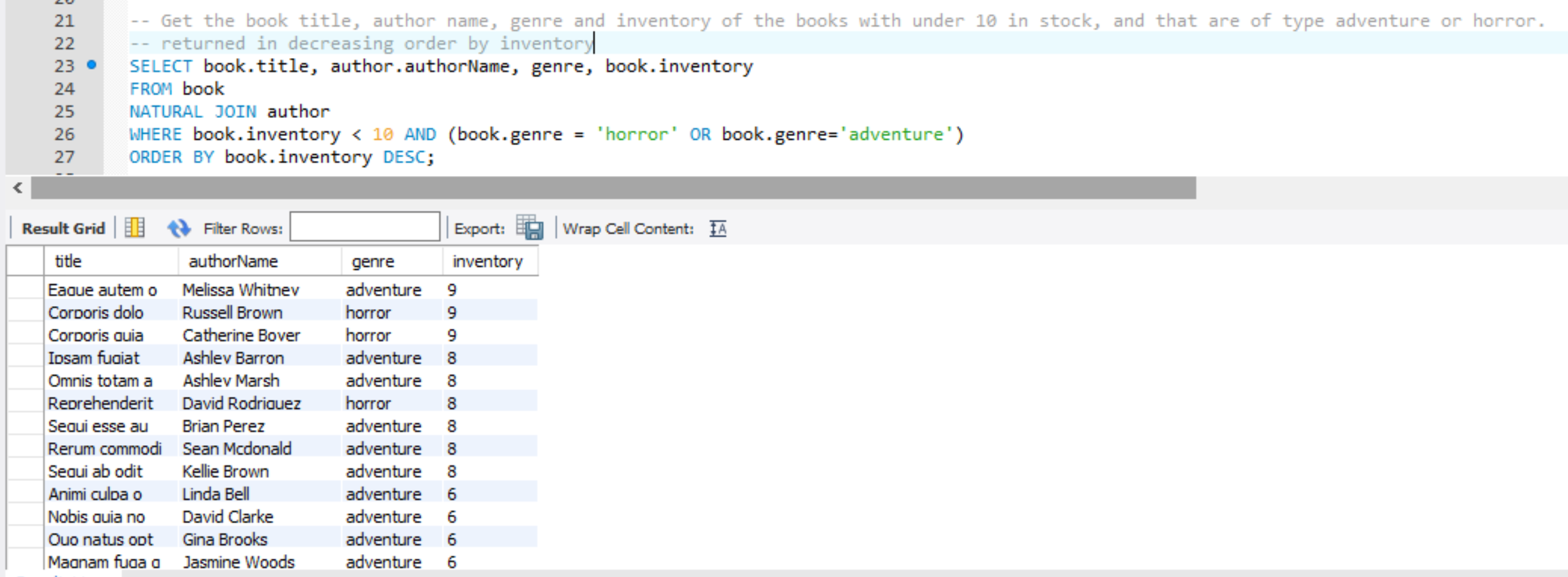
Question 5: Six Interesting queries

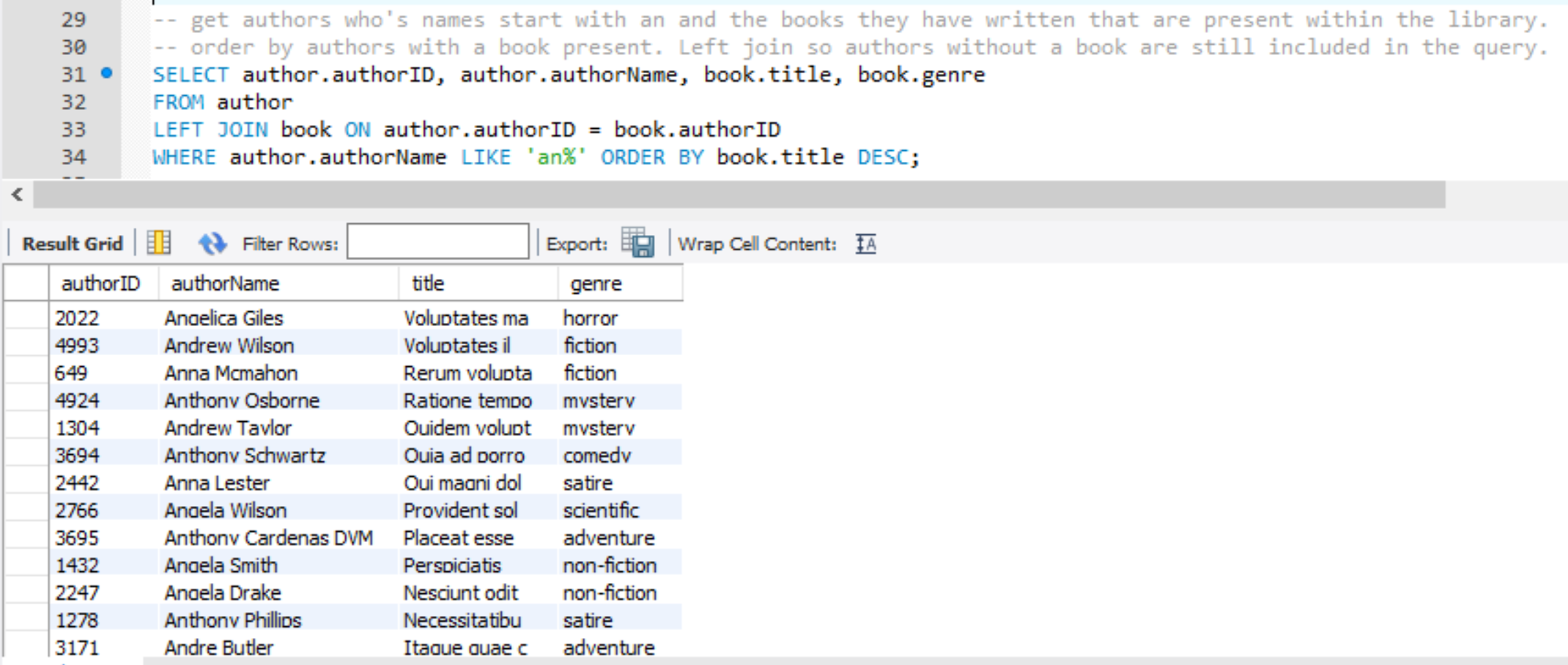
*Below are screenshots of our six interesting queries and their outputs/results.* 



*Note: a count was returned*







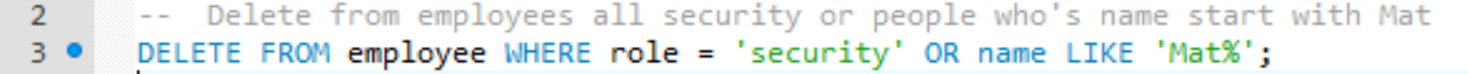
*Note: Only the one tuple was returned*



Question 6: Delete and Update

*Below shows three interesting modification commands. We choose to do one interesting delete and two interesting updates. For each of these commands we included the MySQL statement and the output from running the command to show that the command worked as intended.*

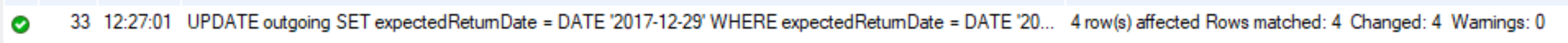
Delete:



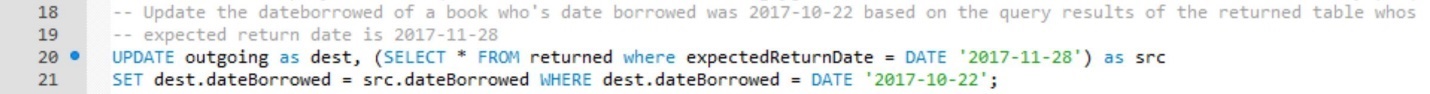


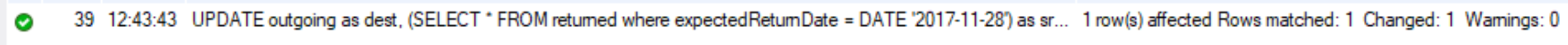
Update:





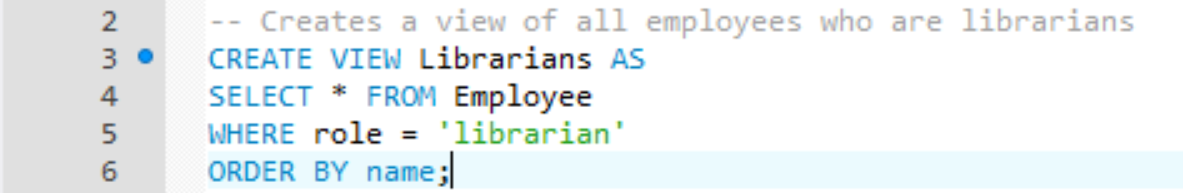
Update 2:



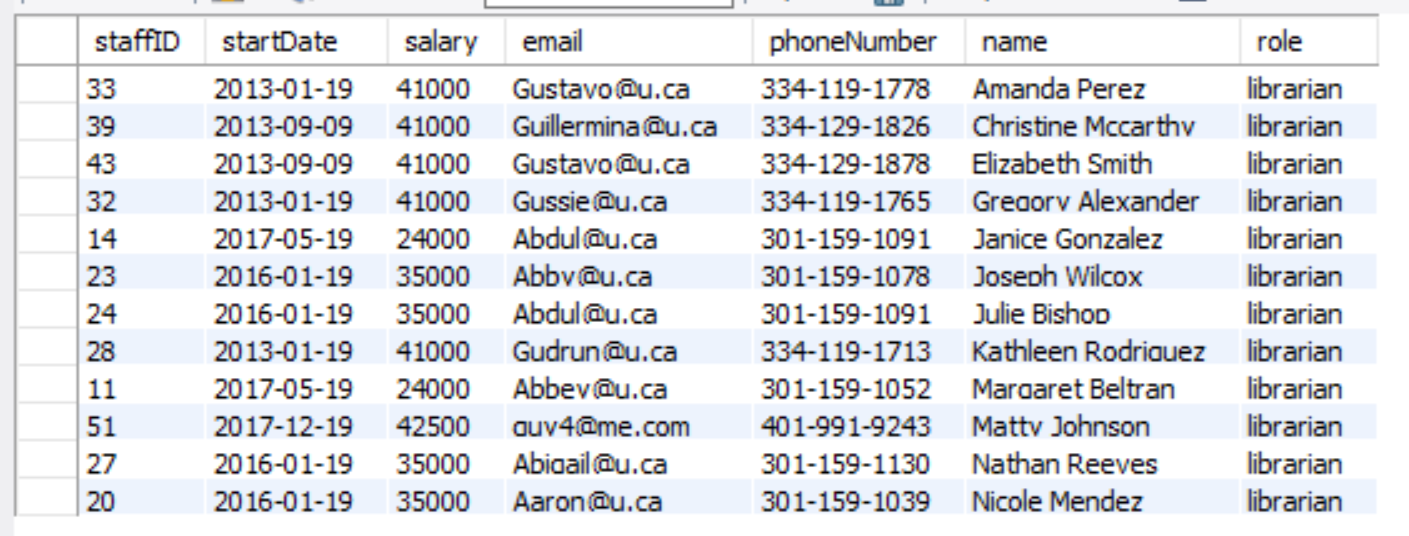


Question 7: Views

Below shows the View commands and the result tables from running the commands. We ran a Create View command and then a select \* command was run to see all the tuples within the View.



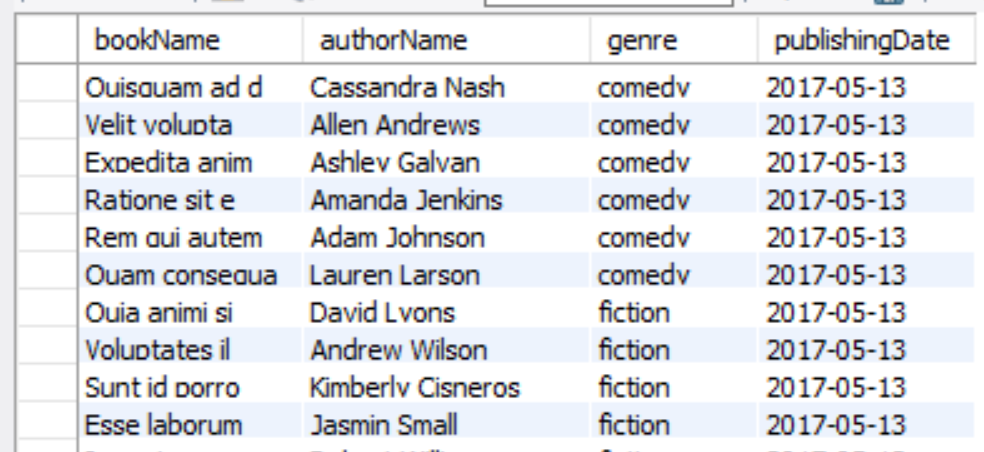
Result from querying all tuples in the above view



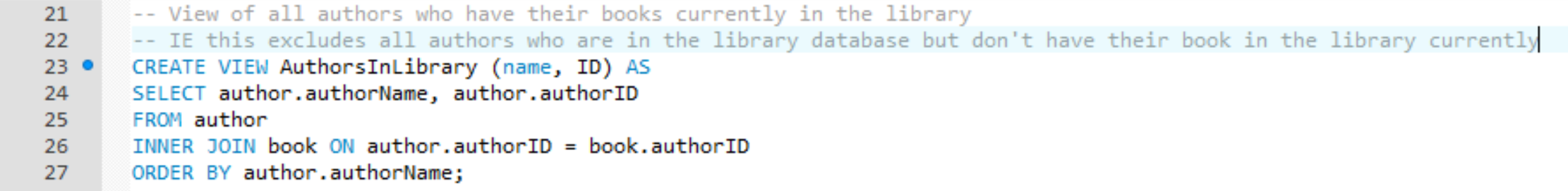
VIEW 2:



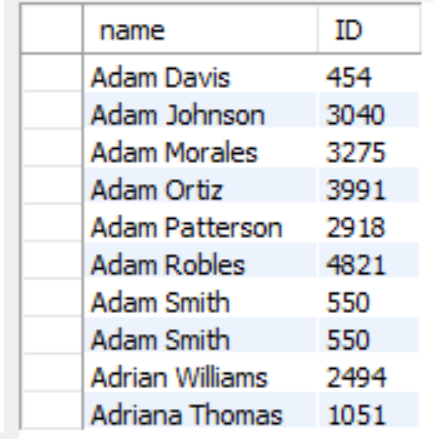
Result from querying all tuples within the view:



VIEW 3:



Resulting from querying all rows within the view.



Inserting into the views we created:

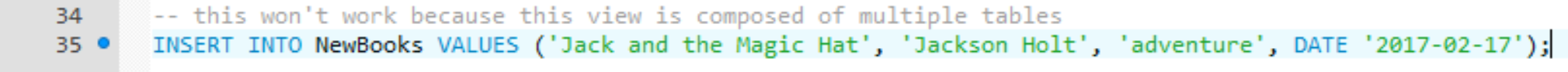
Note: see the comments above the command for an explanation of why the insertion worked or did not work.

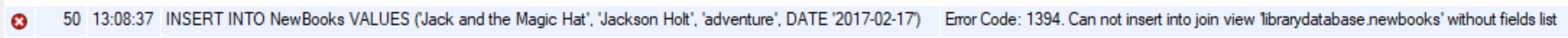
Working insert for view 1:





Not working insert for view 2:





The third view also will not be able to have values inserted because it is composed of a join between multiple tables. We chose not to include the screenshots of the failed insert because it is the same as the one above.

Question 8: Non-Existent MySQL clause.

*The SQL clause that we learned in class, but was not implemented in MySQL is the assertion statement. An assertion is a predicate expressing a condition we wish the database to always satisfy. Through our experience with MySQL we found out that this assertion was unavailable and thus we would not be able to use it if necessary.*