Todd Lunter

1 & 2. These problems were pretty straight forward inserts. I chose a subquery for getting the only id for the particular applications to deal with auto incrementing and making it easy to stack multiple inserts. I split the productivity and games inserts for the two parts.

3. I used regular inserts again for the Developer, but for the Role, I uses a INSERT INTO … SELECT to take care of inserting into all the applications for the categories.

4. I again used a INSERT INTO … SELECT to get the application and developer information.

5. Since I just needed to update the role from PROJECT MANAGER to PRODUCT MANAGER, I did a simple updated with a join on the particular Developer.

6. Rather than try to update or insert into the roles for the Director role of Gregory Peck, I just deleted all his old roles and inserted all Director roles for all web applications. The INSERT INTO … SELECT took care of getting all WebApplications.

7. Revoking Edward Norton's Update privileges on Scripts on Outlook was just a delete based upon a couple joins on Developer and and Application stemming from the Privilege table.

8. Since I needed to remove the PowerPoint application and all things related to it with foreign keys, I did a LEFT JOIN on all things that could be related and performed a DELETE for that particular application. Because of the way MySQL performs the JOIN, I needed application to be joined early on and that would cause a foreign key constraint to fail. I therefore did the Application delete immediately following the deletion of all other Assets, Roles, Privileges, Sales, and WebApplications.

9. First, I determined that I needed to create a view that would look at the Sales for WebApplications where the Developer that did the sell was a Product Manager. I took the Application Id, the Developer Id, the month of the sell and the quantity sold. I needed to only get the web application for the current sale, the role developer for the sale developer, the role application for the sale application and the role PRODUCT MANAGER.

I then performed a join between the sells for this month and the sells for last month of the same year for the same web applications and the same developer and found the ratio between this month to last months sells. I only select the rows that have a ratio equal to or less than 0.9 since that means a 10 percent loss from the previous month.

I then list the product managers name, the view assets names' and the ratio of sales from this month to the last month.

10. I select all the assets and join the applications, to get all the roles for applications, and then join developer to target only developer Alice Wonderland. I also select only the asset type “controller”. I did not select where Alice Wonderland was a “developer” type developer, and chose the general form of the table.

11. To get the highest grossing application, I created a view of all applications, and joined the various forms of applications to get their price. Since a left join of an application that doesn't exist for that type is all NULLs, I could use an IFNULL(...) function to get the price and then got the sames for that application and used the price times all the quantities. I could SUM these for a total gross sale by grouping them by category.

In a second select, I got the category and price, ordering by the price descending and limiting to 1.

12. Again, I created a view, this time getting all the developers, their corresponding sales for mobile applications. I selected the sales for only this month and this year, group the data by the developer and sum the quantity of sales.

In a second select, I order the mobile application sells by the quantity and limit it to only one (gets the highest).