1.) The task of deleting a customer form the customers table can be achieved through a stored procedure or a stored function. I chose to implement this task using a stored function in which a parameter for a customer ID was accepted. I start a transaction in which a variable that I declare is able to detect an empty string as well as an exception handler is declared for a non valid ID. I then began to delete foreign key dependencies in other tables before deleting the customer record that matches the customer ID parameter. The final step is to check to see if my exception handler or empty string variable is set to true. If it is I rollback the changes and if not I commit the changes to the database.

4.) READ UNCOMMITTED would be implemented to allow full access to the data with no locks. This would allow you to read rows that other transactions are currently working on. READ COMMITTED prevents dirty reads in which a select statement would not be able to observe a row unless a transaction has fully committed the new data. This would be useful in preserving the integrity of the data history. REPEATABLE READ is the default isolation level in which all transaction locks are held until the transaction is complete this would still allow the real time insertion of newer rows when another transaction is simultaneously deleting rows that are contingent on the same fields accessed during insertion. SERIALIZABLE isolation level would place a lock on all transactions. Transactions would have to occur in a serialized fashion one after another for the strictest level of transaction isolation. This would be useful when you need strict data integrity. A DB administrator with the authority to grant privileges would create a role using CREATE ROLE, then use the GRANT statement to give specific privileges to the new role. Then, assign an existing user to a role.