# Directions

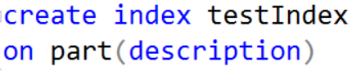
* Complete the following tasks using SQL Server
* Pay attention to the specified database
* Submit the working SQL and proof that the triggers work. The task may require additional screenshots

# Use your premiere database to solve the following

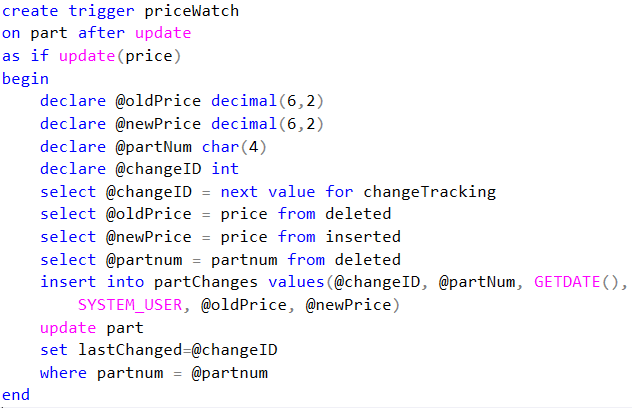
1. Create a DDL trigger that prevents adding an index. Prove it works by trying to create an index.  
   A close-up of a screen

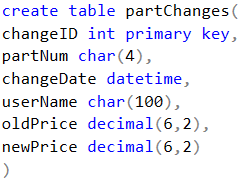
   Description automatically generated  
   A screenshot of a computer

   Description automatically generated



1. Create a change tracking table for the part table. This depends on you completing the first part of the sequence and function assignment from yesterday. I provide the code.

Create the table that holds the changed values

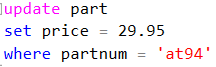


Create the trigger

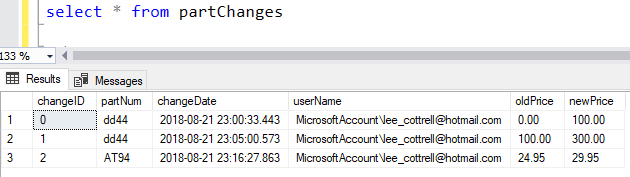
Test the trigger



Run several commands like this one, changing the prices for a few items in the database



Now look at the partChanges table – you should see output similar to mine



Look at the part table, the changed records will have a lastChanged with an integer

For number 2 submit screenshots of your partChanges table and part tables  
A screenshot of a computer

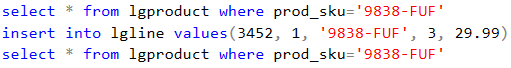
Description automatically generated  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generated

# Use your largeco database to solve the following

1. Create an after insert trigger for lgline. This trigger will subtract the line\_qty from the prod\_qoh in the lgproduct table, and update the lgproduct table for the prod\_sku. Prove it works. When I ran this update, the prod\_qoh went from 137 to 134. Submit both the trigger code and the proof it works.

  
A screenshot of a computer program

Description automatically generated  
  


1. Employees cannot be deleted from the database. We need to set them to be inactive. This is often referred to as ‘tombstoning’ the record. We will modify our database to handle this business rule. You will need to add a BIT[[1]](#footnote-1) field named active to lgemployee. You will then set the value of that field to 1. Then you will create an instead of trigger or delete on lgemployee. This trigger will simply update the active field to 0.
   1. Use alter table to add a bit column named active to lgemployee
   2. Use update to set all records’s active column to 1.
   3. Create an instead of delete trigger on lgemployee. This trigger will update lgemployee and set the active field to 0.
   4. select \* from lgemployee where emp\_num = 83304
      1. **Create a copy of this output, in case you accidentally nuke the record**
   5. Delete employee 83304
      1. 
   6. Rerun the select from above, prove that active is now 0
      1. If you nuked the record, rebuild it using an insert command and try again
   7. Submit the trigger code and the output proving that the employee was marked inactive  
      A screenshot of a computer program

      Description automatically generated  
      

## Bonus

1. Create an after update trigger for lgproduct(prod\_QOH). If the update takes the prod\_QOH less than the prod\_MIN then write the current date, prod\_sku and prod\_min into a new table called lgneedsordered. You need to build this table.  
   A screenshot of a computer code

   Description automatically generated  
   A screenshot of a computer code

   Description automatically generated  
   A screenshot of a computer

   Description automatically generated  
   A screenshot of a computer

   Description automatically generated

1. BIT fields are 1 bit long variables values that can hold1, 1 or null. The string TRUE is converted to 1, FALSE is converted to 0. [↑](#footnote-ref-1)