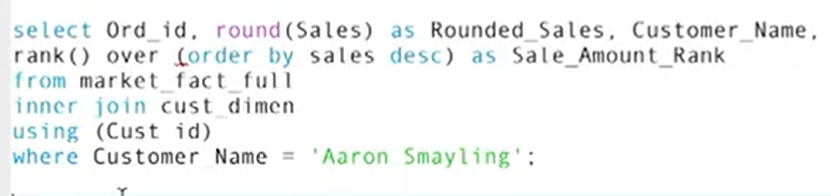
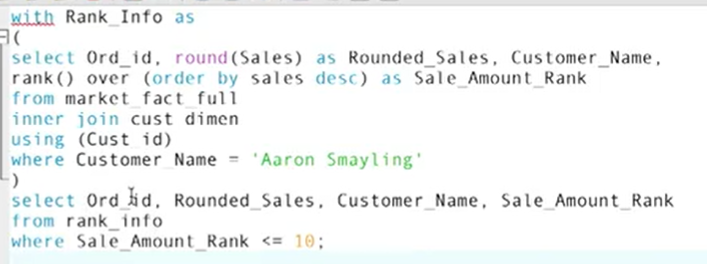
**Advanced SQL:**

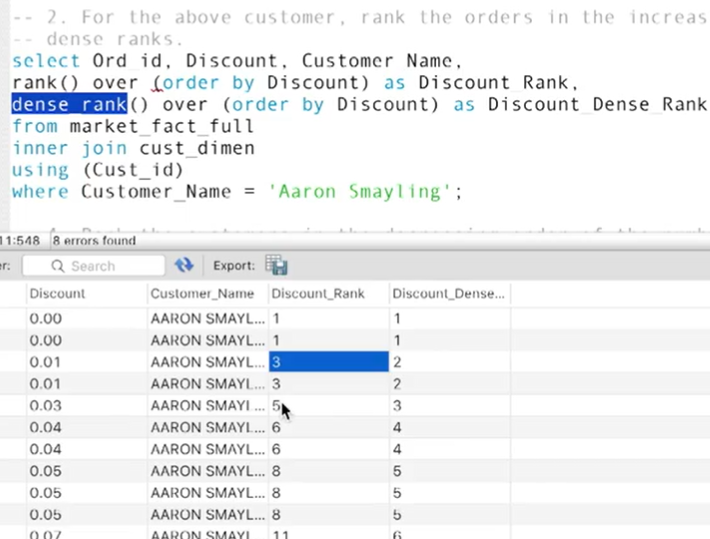
Window Functions, Query Optimization, Programming Constructs and Stored Functions

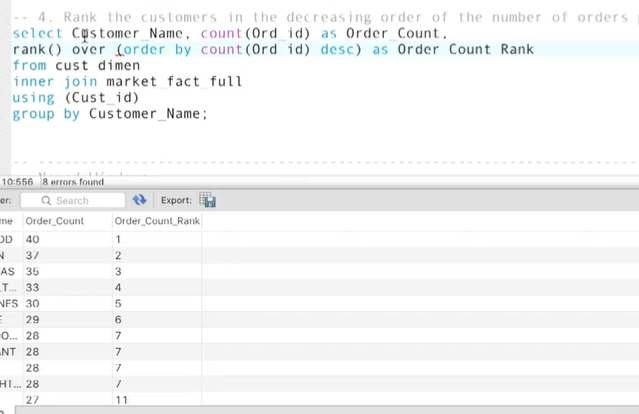
RANK Function:



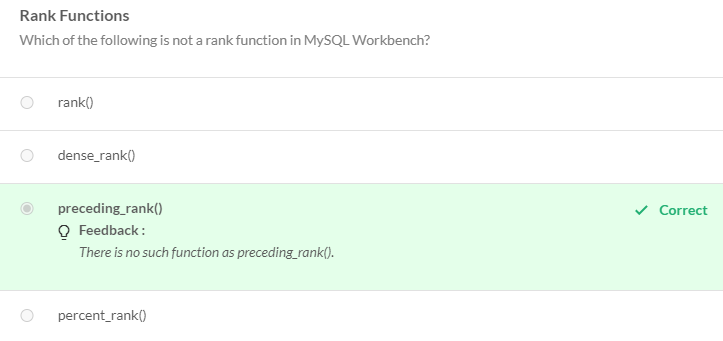


RANK() versus DENSE\_RANK() functions:

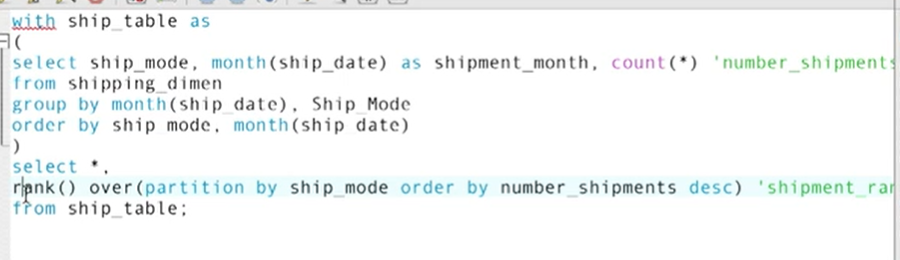




As you saw in the video, do keep in mind that while the rank() function need not have consecutive values, the dense\_rank() function must. For example, consider the table below.

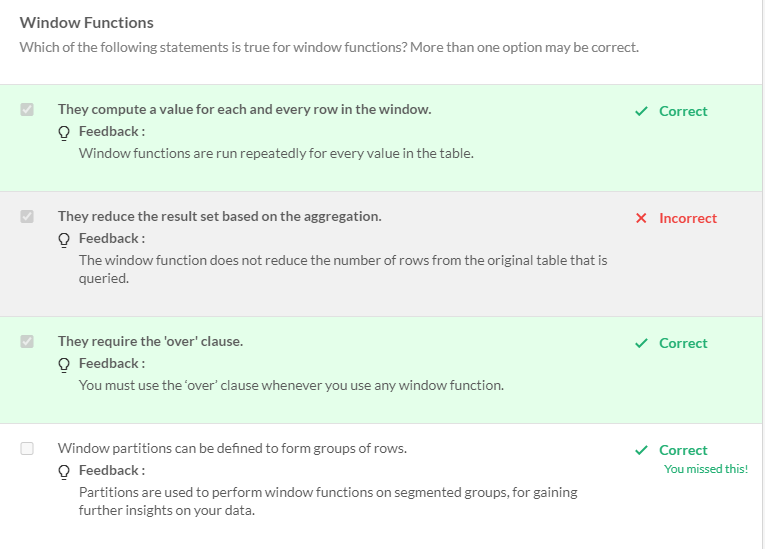


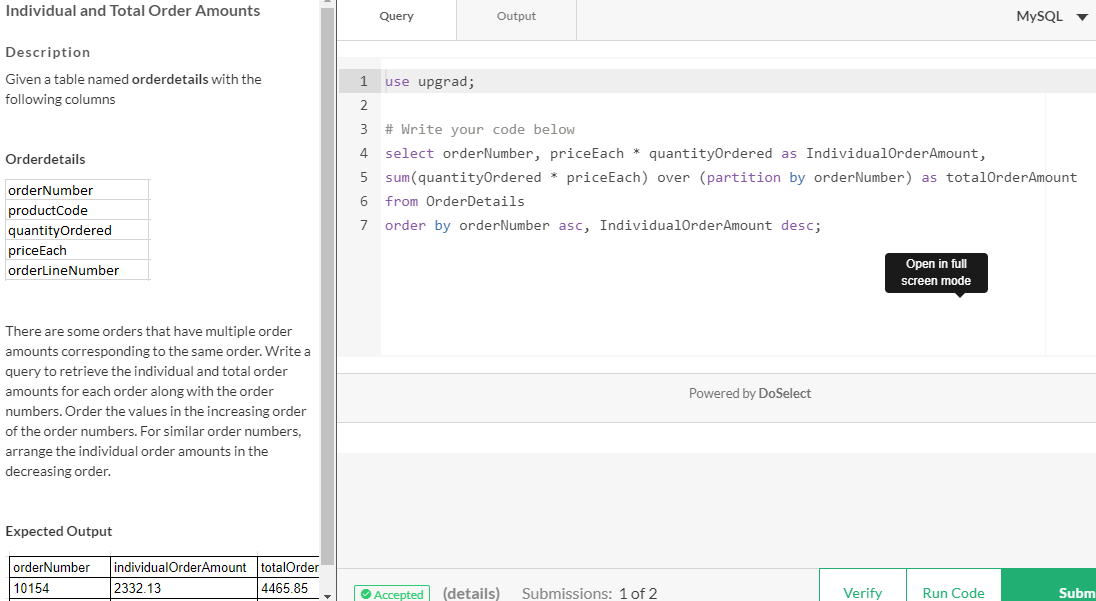
Partitioned Ranking:



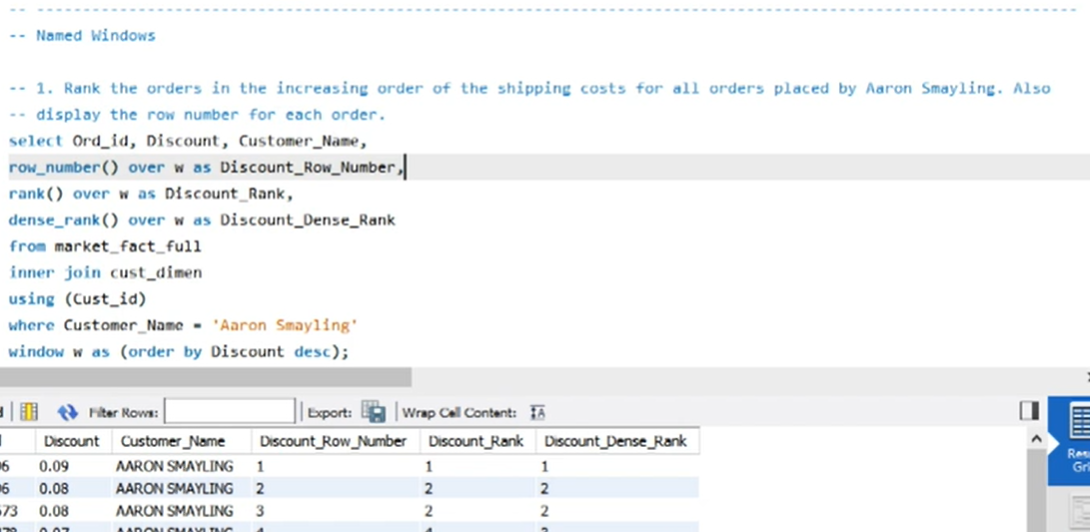
The syntax for the clause used for ranking using partitions in a query is as follows:

* rank() over (partition by <column\_to\_partition> order by <ordering\_criterion>)

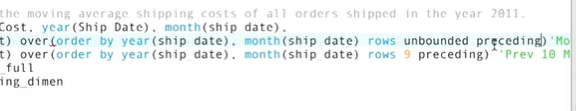




Named Window function:

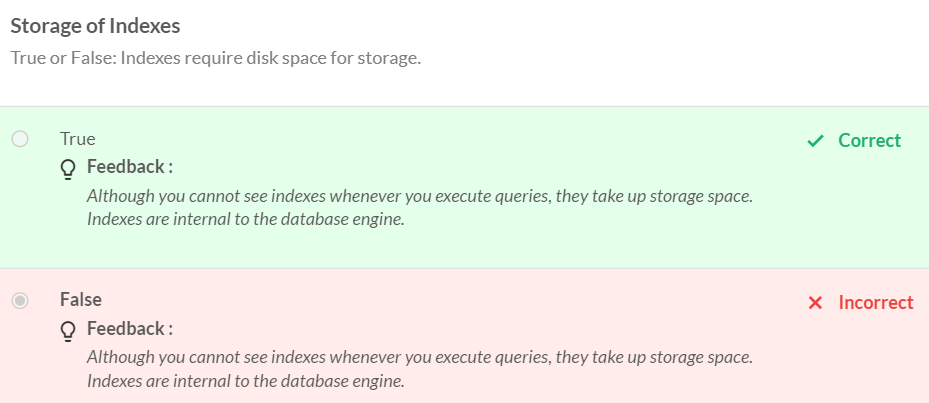


Frames or Moving Averages in SQL:

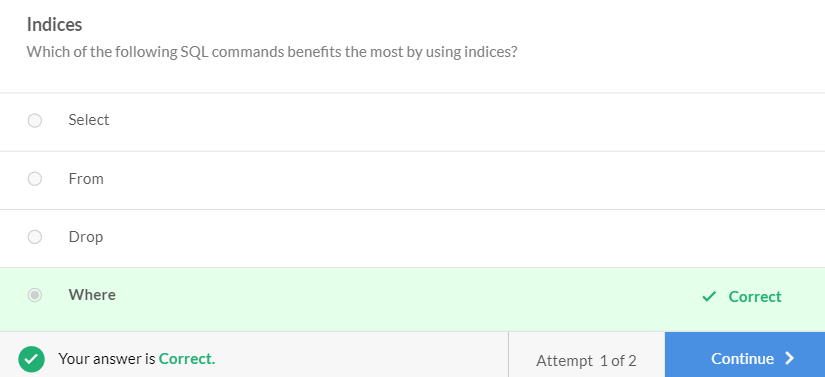


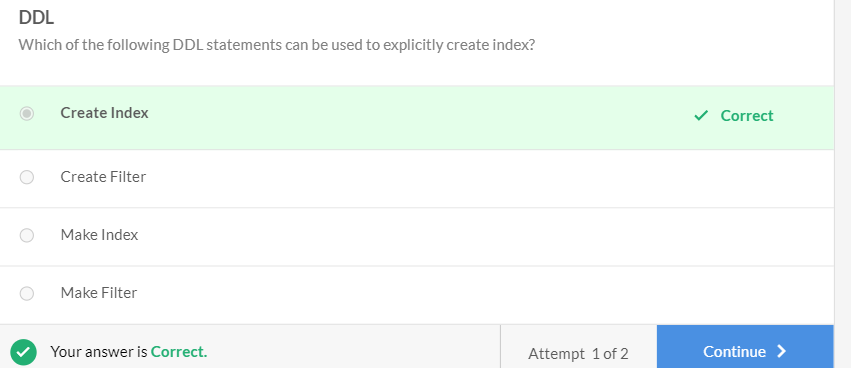
* Introduction to windowing and a discussion on the ‘over’ clause. The ‘over’ clause is used to specify the window on which the function executes.
* The use of aggregate functions - count() and average(), and non-aggregate functions - rank(), dense\_rank() and percent\_rank(), in windowing.
* The requirement and usage of the 'partition' clause.
* How to use named windows in queries in order to make the code more concise and easier to understand.
* The concept of moving averages and how they can be implemented using frames.

Indexing:



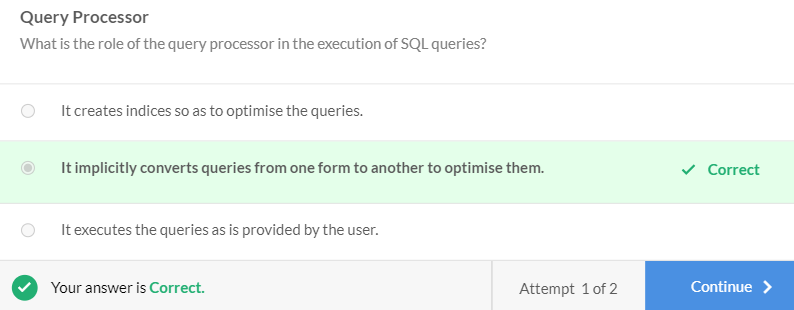
DDL Statement called CREATE INDEX



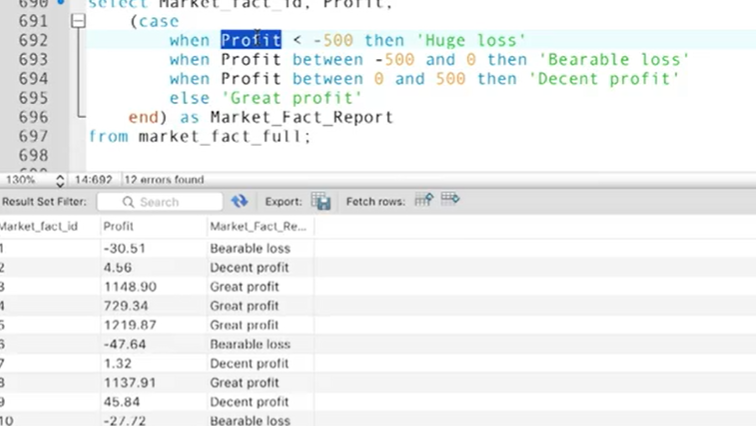


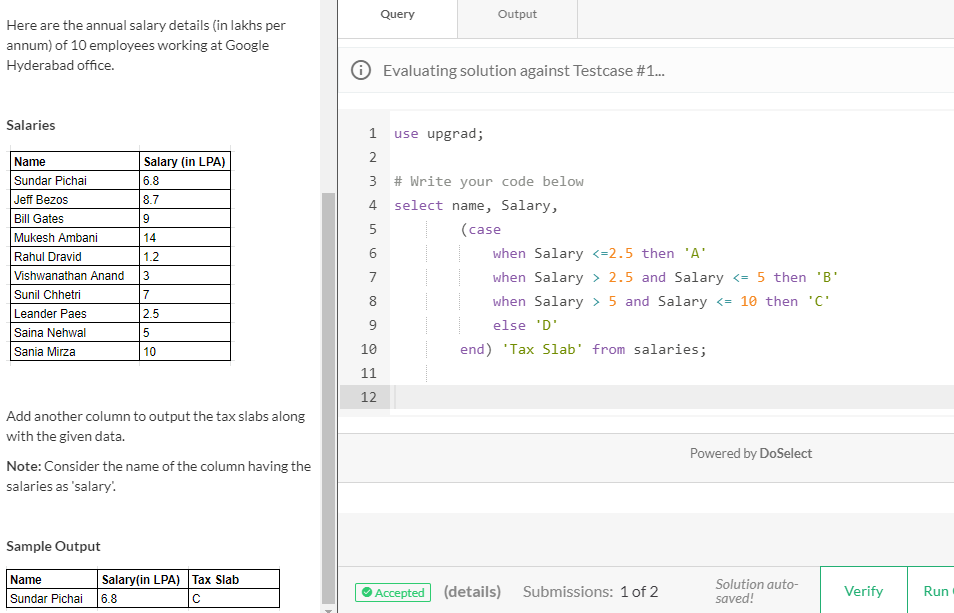
Executing a statement with the join clause creates a join index, which is an internal indexing structure. This makes it more efficient than a nested query. However, the nested query would perform better if you want to query data from a distributed database.

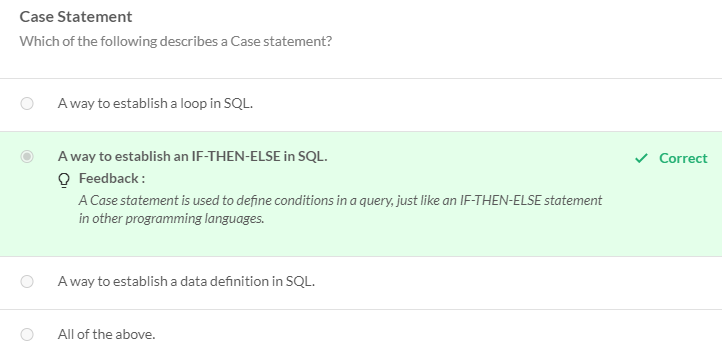
A distributed database has tables stored in different locations, instead of a local system. In this case, a nested query would perform better as we can extract the relevant information from different tables located on different computers. Then, we can merge the values to obtain the result. For a join, we would need to create a large table from the existing tables. Filtering from this large table would require comparatively more time.



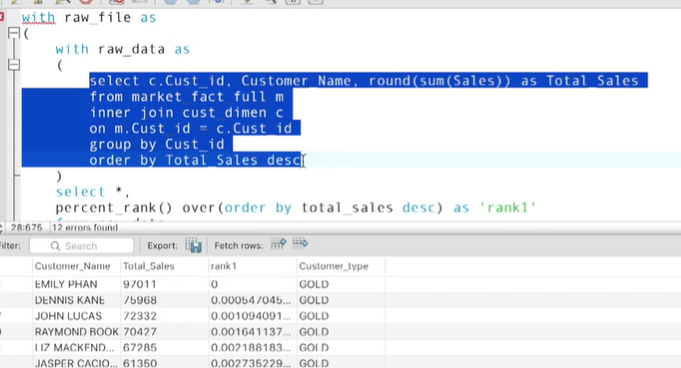
CASE WHEN STATEMENT:

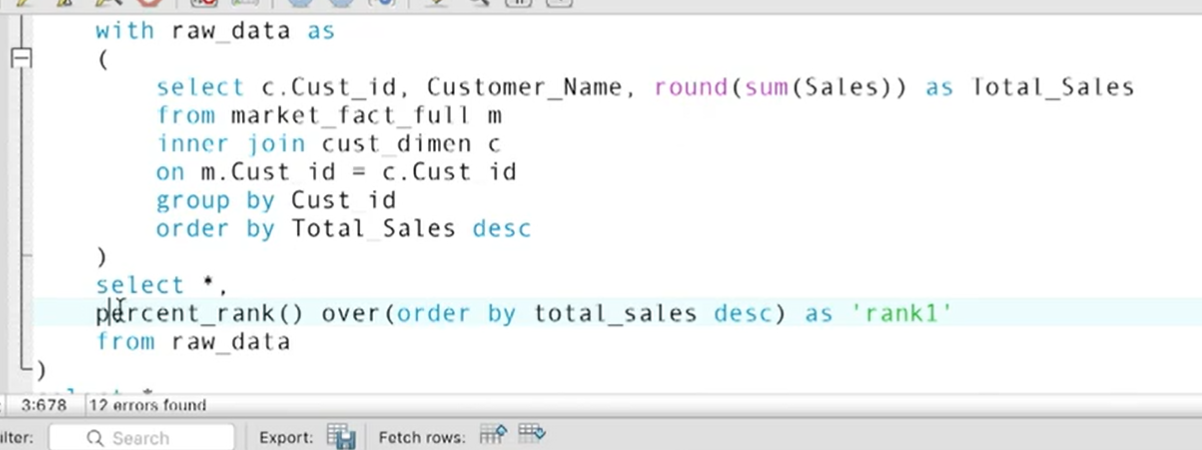


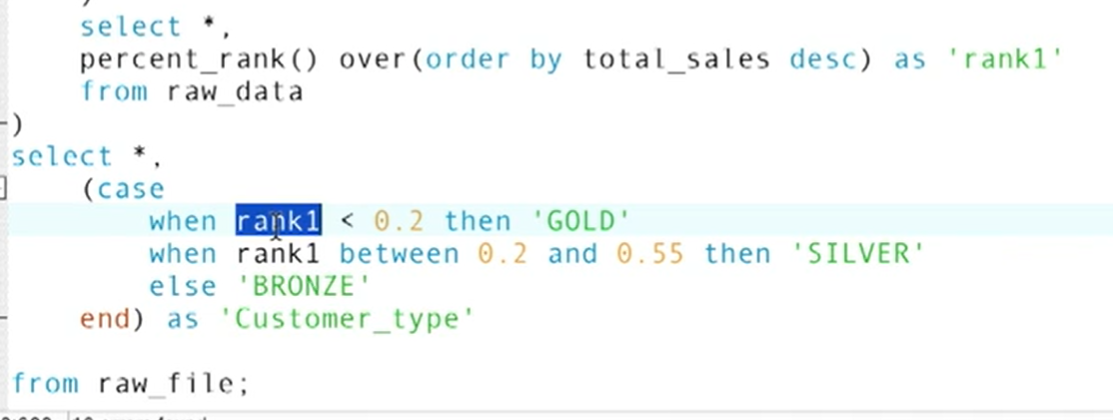


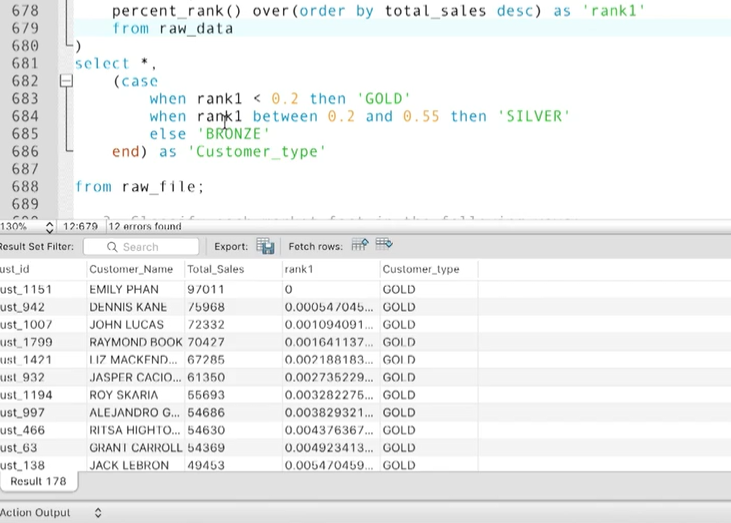


Demonstration of case statement

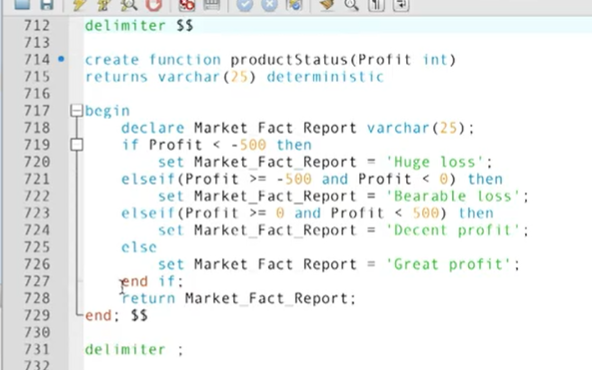




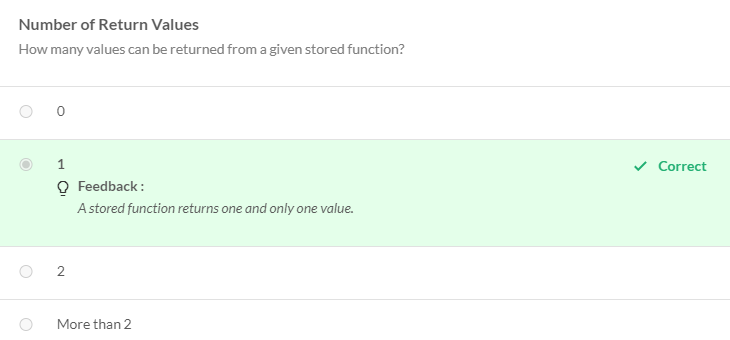


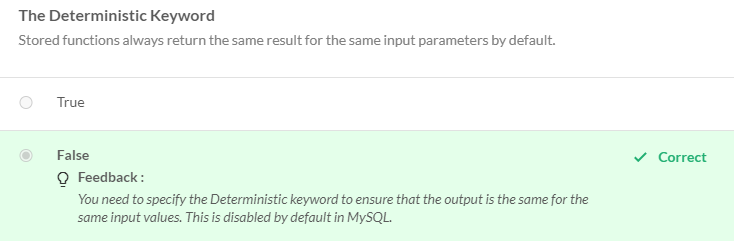


STORED FUNCTIONS:





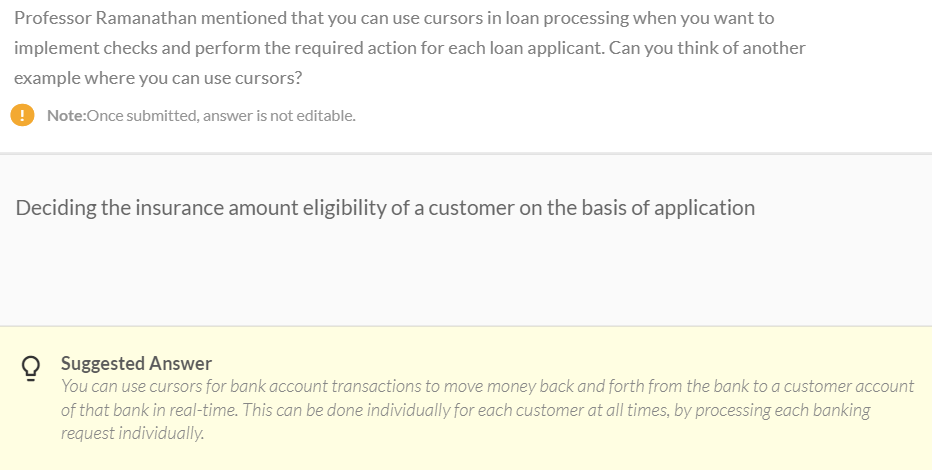




**Cursors**

Cursors are used to individually process each row that is returned in a query.

Stored functions are not system agnostic. Cursors can be used to iterate row by row.



* Case statements - used for applying conditions on your data and classifying your data based on those conditions.
* Stored functions - used to store complex logic that can be reused in more than one query.
* Cursors - used for performing specific operations on each row of the output that you obtain after running a query.
* Disadvantages of stored functions - they are not portable across different database engines.