SQL Test Automation

Product Manual

Ara Institute of Canterbury

Developed By Aditya Raj – Semester One, 2021

Table of Contents

Getting Started

This product manual serves to provide information regarding the SQL Test Automation program for Ara Institute of Canterbury. The information in this document includes the products recommended system requirements, installation, setup, and usage instructions. Please read the entire document before doing anything.

What is SQL Test Automation?

The SQL Test Automation program serves to automate the bulk marking of SQL Server practical tests done by the students of database courses at Ara Institute of Canterbury. This program has been developed to speed up the marking process for tutors of SQL Server database courses in which student work is marked automatically and instantly, not only reducing time and manual work for tutors, but students also benefit by receiving their results much faster. This program reduces the time taken to mark a class of 25 students work from 10 working days, down to approximately 1 minute.

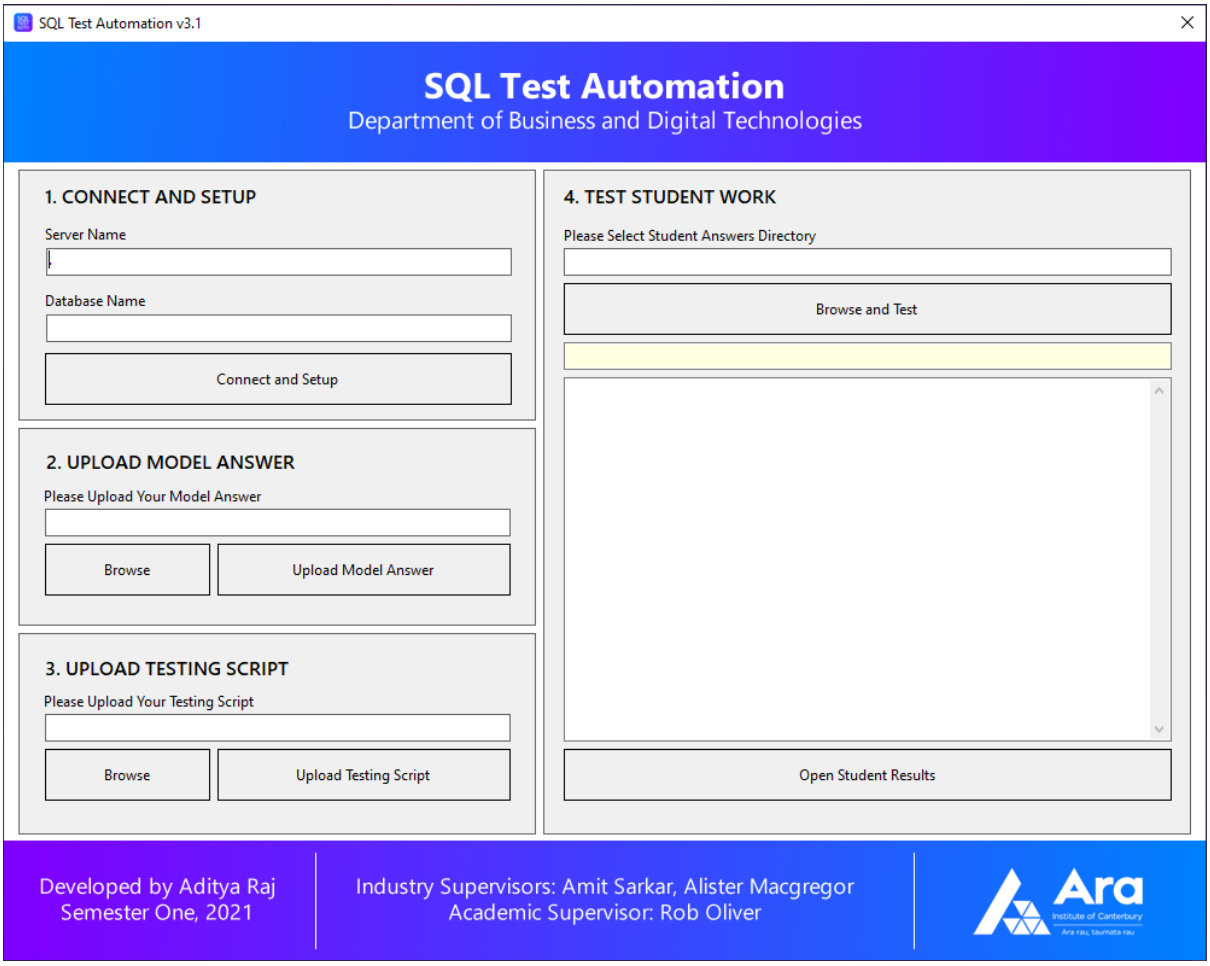
Recommended System Requirements

- Microsoft Windows 10 operating system or newer
- Microsoft SQL Server 2017 or newer
- Microsoft SQL Server Management Studio 17 or newer
- Microsoft .NET Framework 4.5 or newer

Installation

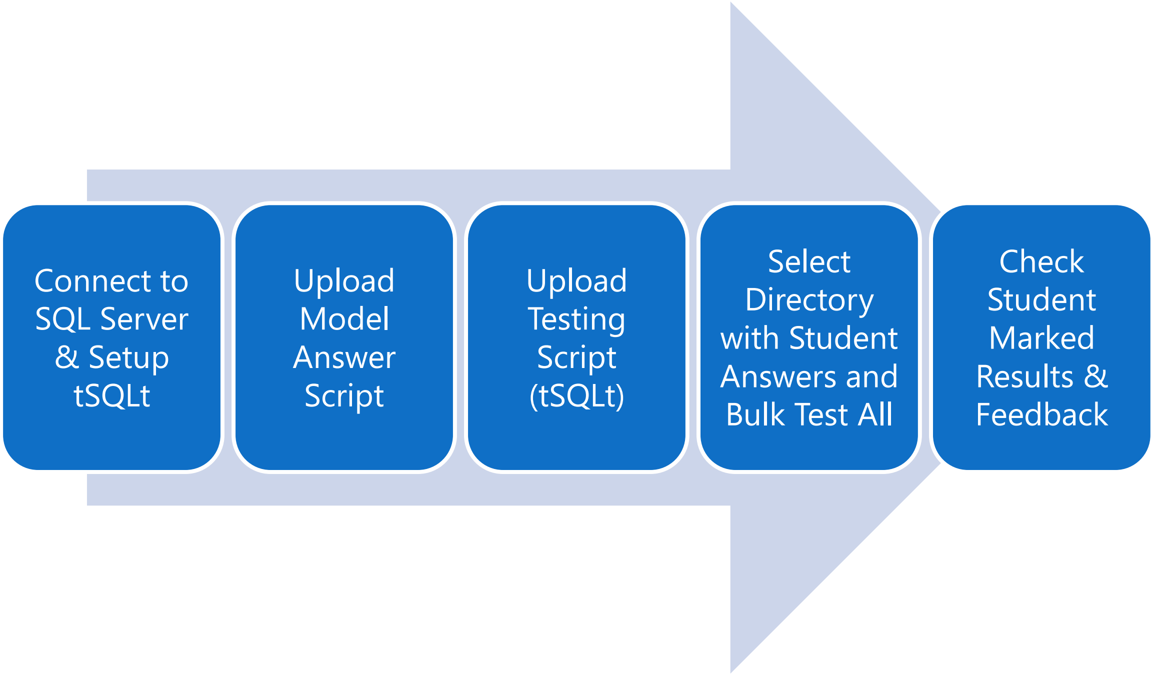
SQL Test Automation has been packaged as a portable program so that the directory with the program executable file and supporting files can be copied to any PC that meets the recommended system requirements and can be run. The package directory has been named *SQL Test Automation v3.1*.

1. Copy the package directory to a local location, for example, *C:\ drive*.
2. Within this directory, the *template_scripts* subdirectory contains empty templates of the model answer, the student answer, and the testing script – please read the next section *Using SQL Test Automation* to make use of these templates.
3. Open **SQLTestAutomation_V3.exe** in the package directory to run the program.
4. The program is now ready for use - please read the next section *Using SQL Test Automation*.



Using SQL Test Automation

The following section serves to provide information regarding the usage of the SQL Test Automation program. The usage of this product involves a process starting with the connection of the program to the SQL Server and setup of the tSQLt Unit Testing Framework, then the upload of the Model answer, the tSQLt Testing script and finally the bulk testing of multiple student answers uploaded onto the program using directory selection. The process ends with the results of the students' answers being outputted into a new subdirectory in text format for each student, this subdirectory can be directly accessed from this program at the end of this process.



Connect to SQL Server & Setup tSQLt

Upload Model Answer Script

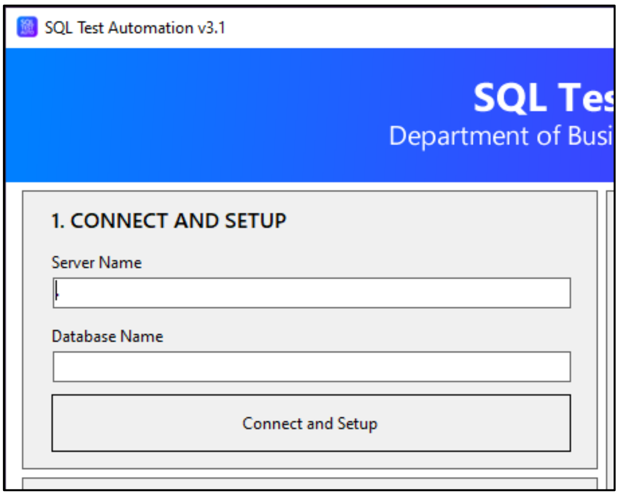
Upload Testing Script (tSQLt)

Select Directory with Student Answers and Bulk Test All

Check Student Marked Results & Feedback

Connect to SQL Server and Setup tSQLt

The pre-requisite to this step is to have Microsoft SQL Server installed, this can be version 2017 or newer as per the recommended system requirements. SQL Server will install with at least one instance; the user can use this instance or create and use another instance of their choice. Within this instance, the user will need to either restore a backed-up database or execute a SQL script that will create the new database using Microsoft SQL Server Management Studio. The database will need to be the same as the one the students used in their practical tests.

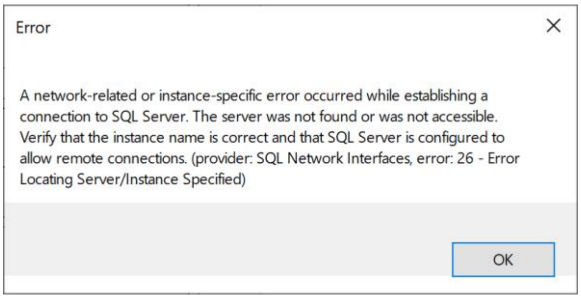


Once the program has been run as per the *Installation* section, the first required step is to connect the program to Microsoft SQL Server. The first panel of the program allows the user to create a custom connection string to the SQL Server instance and the database within this instance of the user’s choice. The single dot represents the root server instance, another server name can be “.\SQLEXPRESS”.

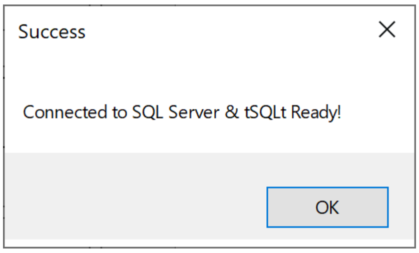
The user will need to type in the Server Name and the Database Name in their appropriate fields and click “Connect and Setup”.

The program will now attempt to connect to the SQL Server and once connected, automatically set up the tSQLt unit testing framework which will be needed later.

If the provided information does not match the SQL Server, the following error or similar message will appear. Please check for the correct information in SQL Server Management Studio and try again.



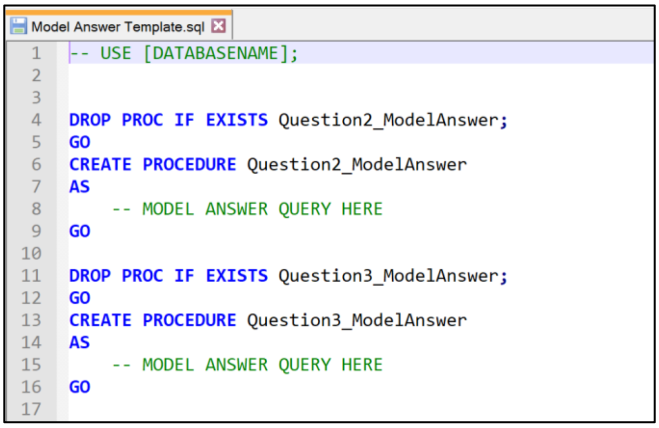
A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured to allow remote connections. (provider: SQL Network Interfaces, error: 26 - Error Locating Server/Instance Specified)



Connected to SQL Server & tSQLt Ready!

Once the provided information matches with the SQL Server, the program will automatically set up the tSQLt unit testing framework and this Success message will appear, please click OK to continue.

Upload Model Answer Script

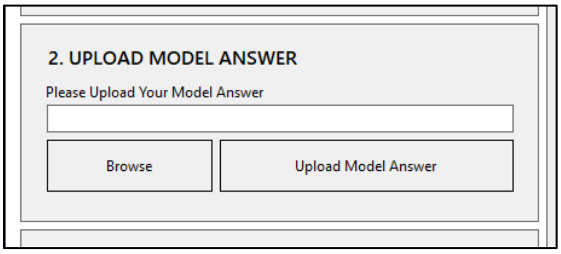


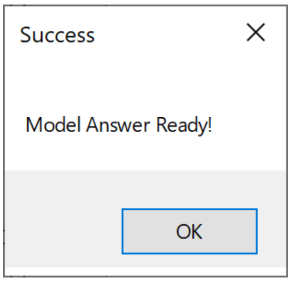
```sql
-- USE [DATABASENAME];


DROP PROC IF EXISTS Question2_ModelAnswer;
GO
CREATE PROCEDURE Question2_ModelAnswer
AS
    -- MODEL ANSWER QUERY HERE
GO

DROP PROC IF EXISTS Question3_ModelAnswer;
GO
CREATE PROCEDURE Question3_ModelAnswer
AS
    -- MODEL ANSWER QUERY HERE
GO

```

The pre-requisite to this step is to use the *template_scripts\Model Answer Template.sql* file to write the model answers from the questions from the practical test the students in the database course have completed. The user will need to follow the instructions within this script to fill it in and save a copy. This model answer script will be used by the program to compare it to the students' answers.

Once the program is connected to the SQL Server and tSQLt is ready for use, this step will be possible.

On the program, click the "Browse" button and select your model answer SQL file. Finally, click the "Upload Model Answer" button to execute it into the program and SQL Server.



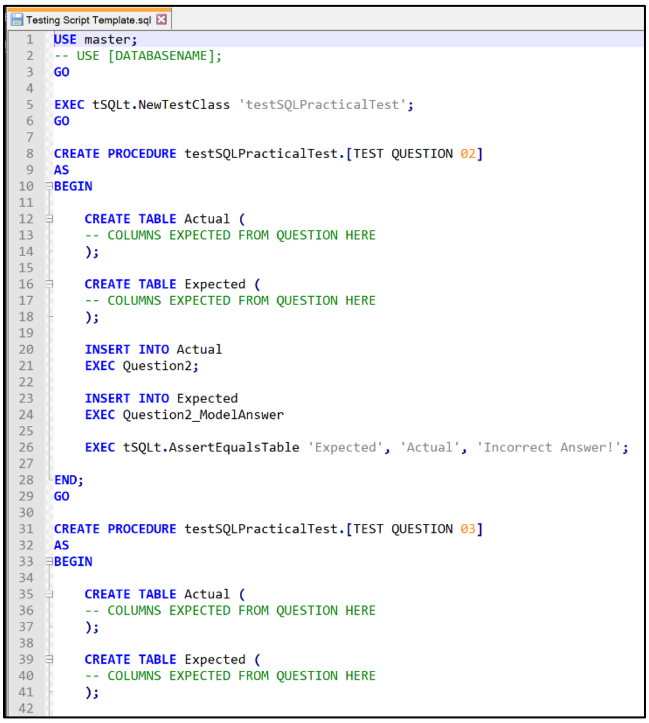


Model Answer Ready!

If an incorrect SQL file is selected, the process of the program will break, and student answer testing will not show the appropriate results. If your file is not selected, the program will throw an error, please try again.

Once the model answer script is ready the user should receive a success message, please click OK to continue.

Upload tSQLt Testing Script

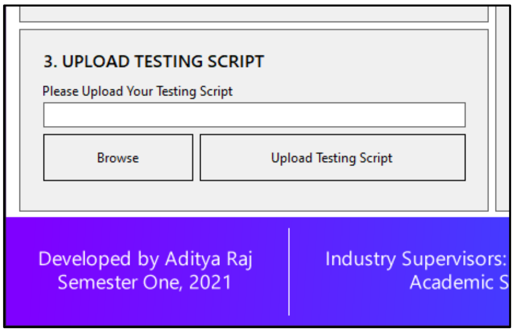


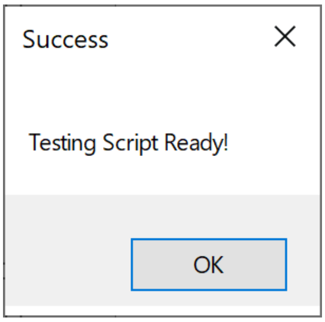
```
USE master;
-- USE [DATABASENAME];
GO

EXEC tSQLt.NewTestClass 'testSQLPracticalTest';
GO

CREATE PROCEDURE testSQLPracticalTest.[TEST QUESTION 02]
AS
BEGIN

    CREATE TABLE Actual (
    -- COLUMNS EXPECTED FROM QUESTION HERE
    );

    CREATE TABLE Expected (
    -- COLUMNS EXPECTED FROM QUESTION HERE
    );

    INSERT INTO Actual
    EXEC Question2;

    INSERT INTO Expected
    EXEC Question2_ModelAnswer

    EXEC tSQLt.AssertEqualsTable 'Expected', 'Actual', 'Incorrect Answer!';

END;
GO

CREATE PROCEDURE testSQLPracticalTest.[TEST QUESTION 03]
AS
BEGIN

    CREATE TABLE Actual (
    -- COLUMNS EXPECTED FROM QUESTION HERE
    );

    CREATE TABLE Expected (
    -- COLUMNS EXPECTED FROM QUESTION HERE
    );

```

The pre-requisite to this step is to use the *template_scripts\Testing Script Template.sql* file to write a tSQLt testing script that takes account of the model answer and the student answers to output the students results later. The user will need to follow the instructions within this script to fill it in and save a copy. This testing script creates a tSQLt Test Class with many stored procedures which act as each question in the practical test, these execute the actual (student answer) and expected (model answer) stored procedures to compare and provide the results. The correct columns must be added to each question stored procedure for the appropriate results to be output.

Once the program is connected to the SQL Server and tSQLt is ready for use, this step will be possible.

On the program, click the "Browse" button and select your testing script SQL file. Finally, click the "Upload Testing Script" button to execute it into the program and SQL Server.



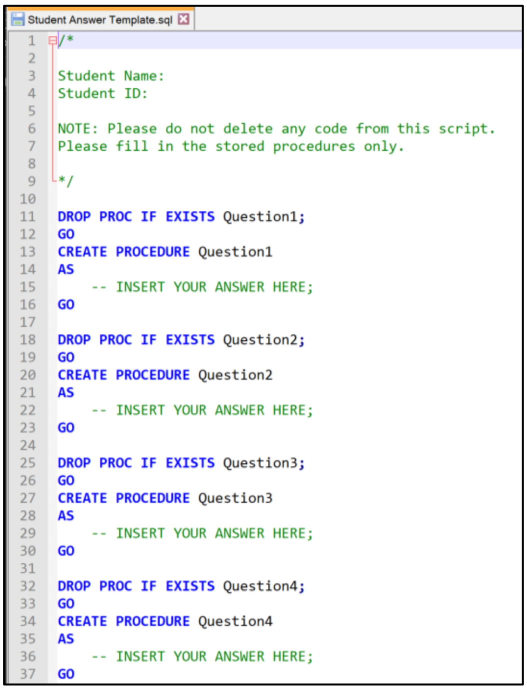


Testing Script Ready!

If an incorrect SQL file is selected, the process of the program will break, and student answer testing will not show the appropriate results. If your file is not selected, the program will throw an error, please try again.

Once the testing script is ready the user should receive a success message, please click OK to continue.

Testing Student Answers from Directory



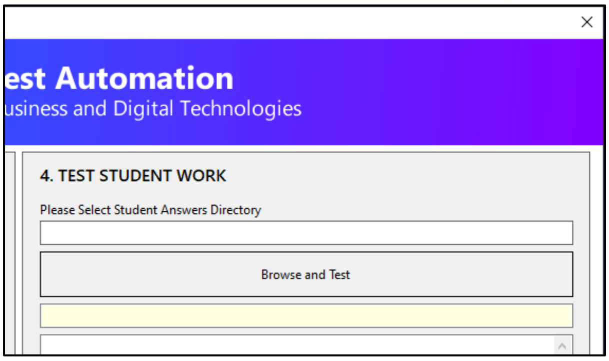
```
/*

Student Name:
Student ID:

NOTE: Please do not delete any code from this script.
Please fill in the stored procedures only.

*/

DROP PROC IF EXISTS Question1;
GO
CREATE PROCEDURE Question1
AS
    -- INSERT YOUR ANSWER HERE;
GO

DROP PROC IF EXISTS Question2;
GO
CREATE PROCEDURE Question2
AS
    -- INSERT YOUR ANSWER HERE;
GO

DROP PROC IF EXISTS Question3;
GO
CREATE PROCEDURE Question3
AS
    -- INSERT YOUR ANSWER HERE;
GO

DROP PROC IF EXISTS Question4;
GO
CREATE PROCEDURE Question4
AS
    -- INSERT YOUR ANSWER HERE;
GO
```

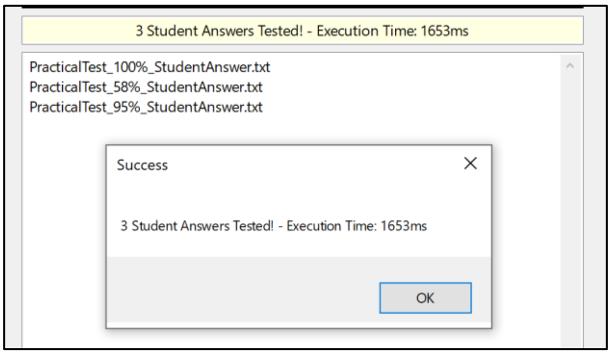
All the students in the database course sitting for their practical test will need to use the supplied file: *template_scripts\Student Answer Template.sql*. This template script will allow the program to execute and test the students work, if this template is not used, the program will not be able to appropriately mark the students' work.

Each student shall fill in their answer in the supplied stored procedures on the script. For example, if a student does not know the answer to a question, they can leave the stored procedure empty, do not delete any of the template code. Tutors will simply need to add or remove the number of stored procedures depending on the number of questions in the practical test.

Once all previous steps have been completed before this section, this step is only possible. The pre-requisite to this step is that all the students SQL files with their answers will need to be stored in a single directory and for best performance, this directory should be stored on a local drive, for example, the C:/ drive on the Windows PC.

On the program, click the "Browse and Test" button, then locate and select the directory with all the student answer SQL files. Finally, please wait… the program will execute all the student answer SQL scripts and test them with the testing script and model answer using the SQL Server connection.


est Automation
usiness and Digital Technologies
4. TEST STUDENT WORK
Please Select Student Answers Directory
Browse and Test



3 Student Answers Tested! - Execution Time: 1653ms

PracticalTest_100%_StudentAnswer.txt
PracticalTest_58%_StudentAnswer.txt
PracticalTest_95%_StudentAnswer.txt

Success

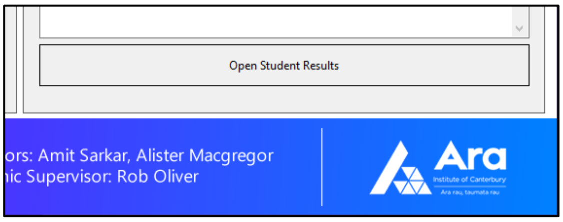
3 Student Answers Tested! - Execution Time: 1653ms

Once the process is complete, a success message will pop up notifying the user how many students' work the program marked and the time it's taken, which will vary depending on the number of questions in the practical test, the number of students in the class and the number of failures and errors the program produces in the results. Please click OK to continue.

NOTE: If there are any errors in the student code, the program will throw an error notifying the user with the student SQL file name that is responsible for the specific error. Please fix this and try again.

Viewing Student Marked Results

If the program has successfully marked all the students' work, then… Congratulations!


Open Student Results
ors: Amit Sarkar, Alister Macgregor
ic Supervisor: Rob Oliver
Ara
Institute of Canterbury

SQL Test Automation has saved you countless hours manually marking student work. You can now view all the results with feedback for each student by simply clicking the "Open Student Results" button, this will open the new subdirectory created named *StudentTestResults* within the student answers directory. This new subdirectory contains text files containing the results for each students' SQL file, you can simply open these text files using your favorite text editors such as Notepad or Notepad++.

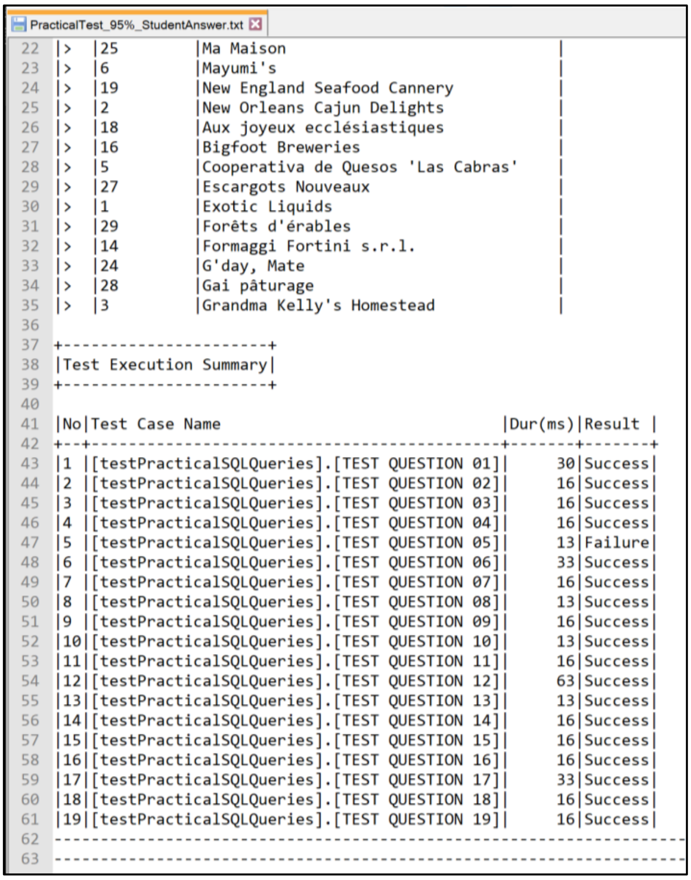
The user will be able to see specifically what the student got wrong for the incorrect answers using the feedback produced.

The Test Execution Summary table at the end of each text file shows the Success, Failure or Error results for the student's answers.

- **Success** – Correct Answer.
- **Failure** – Incorrect Answer, however close to the correct answer.
- **Error** – Completely incorrect answer.

NOTE: Any students that get all the answers correct in comparison to the model answer, their text file will read: *100% - No Errors or Failures Detected!*

These results and feedback may now be used to provide the student with a grade and/or shared with the student.



```
|>  |25        |Ma Maison                              |
|>  |6         |Mayumi's                               |
|>  |19        |New England Seafood Cannery            |
|>  |2         |New Orleans Cajun Delights             |
|>  |18        |Aux joyeux ecclésiastiques             |
|>  |16        |Bigfoot Breweries                      |
|>  |5         |Cooperativa de Quesos 'Las Cabras'     |
|>  |27        |Escargots Nouveaux                     |
|>  |1         |Exotic Liquids                         |
|>  |29        |Forêts d'érables                       |
|>  |14        |Formaggi Fortini s.r.l.                |
|>  |24        |G'day, Mate                            |
|>  |28        |Gai pâturage                           |
|>  |3         |Grandma Kelly's Homestead              |

+----------------------+
|Test Execution Summary|
+----------------------+

|No|Test Case Name                              |Dur(ms)|Result |
+--+--------------------------------------------+-------+-------+
|1 |[testPracticalSQLQueries].[TEST QUESTION 01]|     30|Success|
|2 |[testPracticalSQLQueries].[TEST QUESTION 02]|     16|Success|
|3 |[testPracticalSQLQueries].[TEST QUESTION 03]|     16|Success|
|4 |[testPracticalSQLQueries].[TEST QUESTION 04]|     16|Success|
|5 |[testPracticalSQLQueries].[TEST QUESTION 05]|     13|Failure|
|6 |[testPracticalSQLQueries].[TEST QUESTION 06]|     33|Success|
|7 |[testPracticalSQLQueries].[TEST QUESTION 07]|     16|Success|
|8 |[testPracticalSQLQueries].[TEST QUESTION 08]|     13|Success|
|9 |[testPracticalSQLQueries].[TEST QUESTION 09]|     16|Success|
|10|[testPracticalSQLQueries].[TEST QUESTION 10]|     13|Success|
|11|[testPracticalSQLQueries].[TEST QUESTION 11]|     16|Success|
|12|[testPracticalSQLQueries].[TEST QUESTION 12]|     63|Success|
|13|[testPracticalSQLQueries].[TEST QUESTION 13]|     13|Success|
|14|[testPracticalSQLQueries].[TEST QUESTION 14]|     16|Success|
|15|[testPracticalSQLQueries].[TEST QUESTION 15]|     16|Success|
|16|[testPracticalSQLQueries].[TEST QUESTION 16]|     16|Success|
|17|[testPracticalSQLQueries].[TEST QUESTION 17]|     33|Success|
|18|[testPracticalSQLQueries].[TEST QUESTION 18]|     16|Success|
|19|[testPracticalSQLQueries].[TEST QUESTION 19]|     16|Success|
--------------------------------------------------------------------
--------------------------------------------------------------------
```

The user has now completed the SQL Test Automation programs process, this process can now be redone for other practical tests. The program closes with the click of the X close button on the top right corner.

Thank you for using this product, I hope it helped you well.