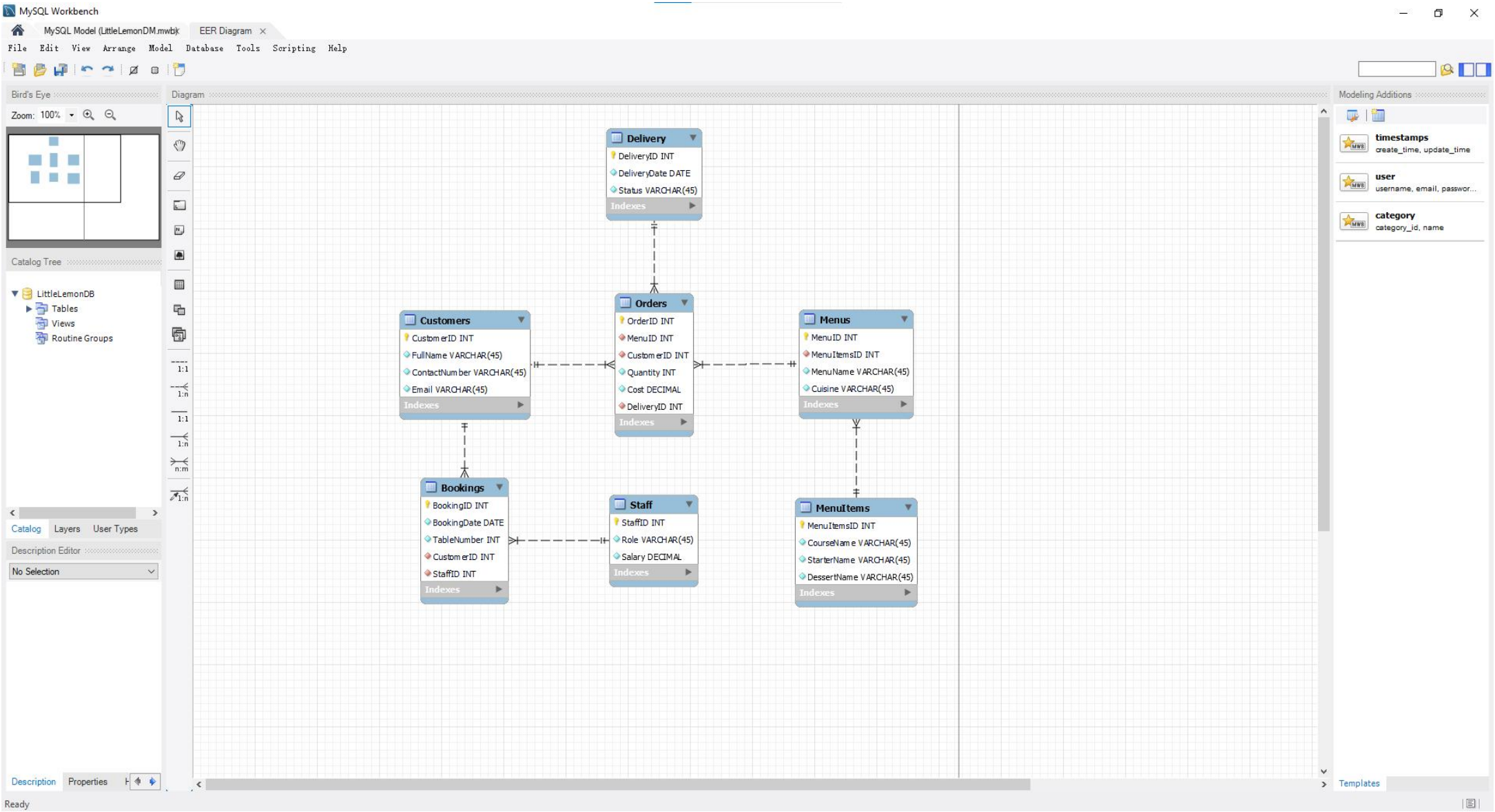
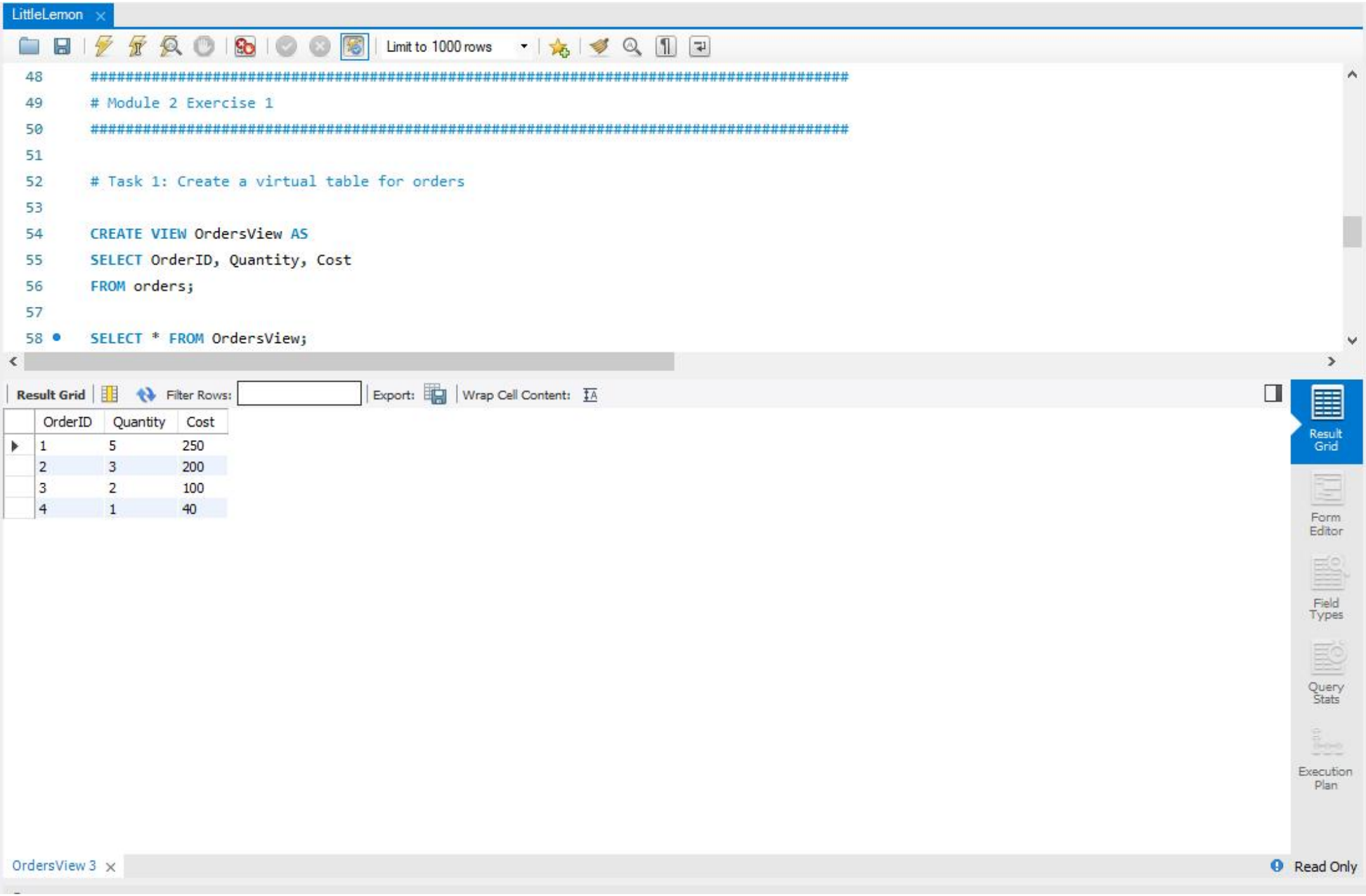
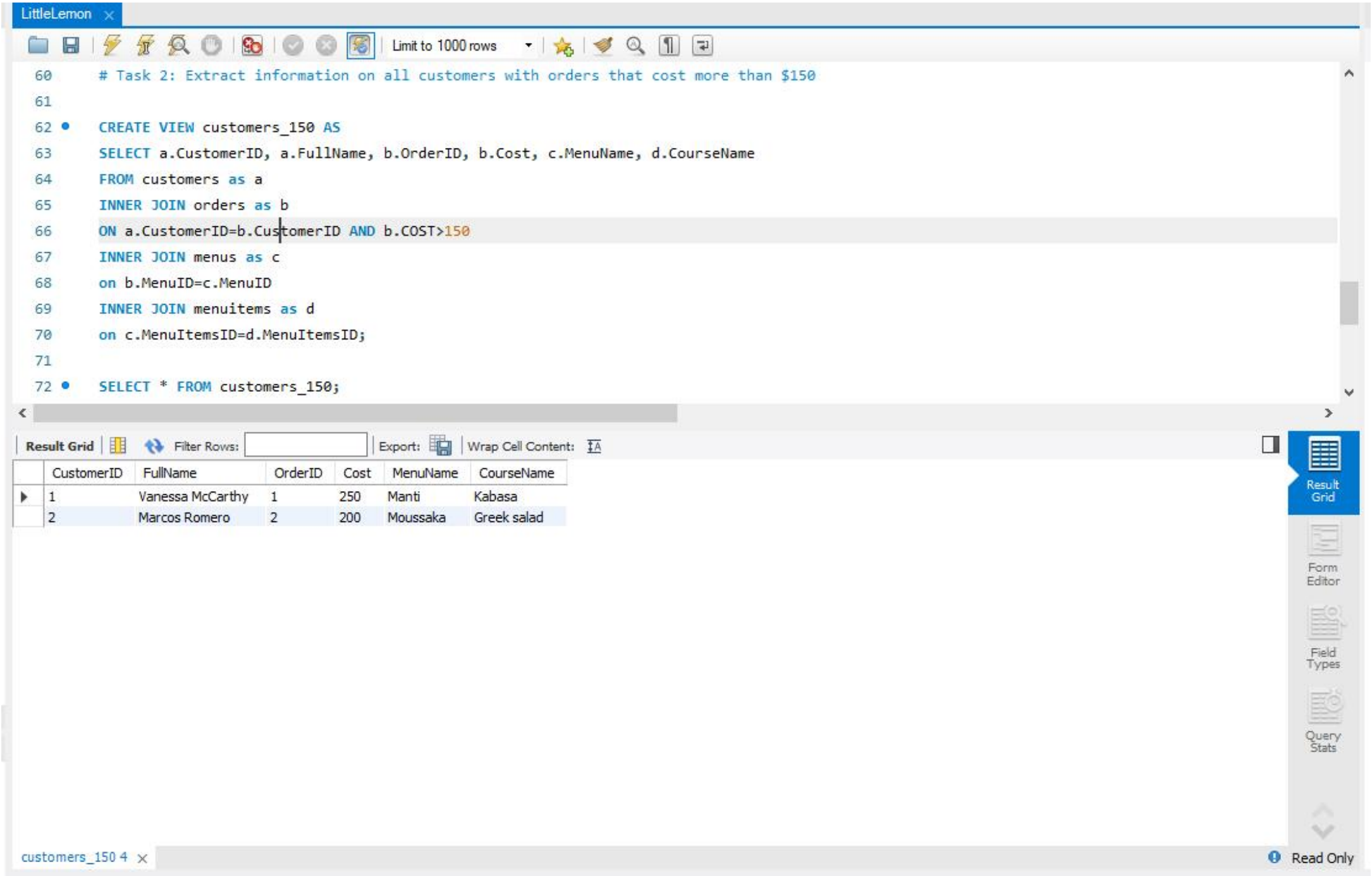
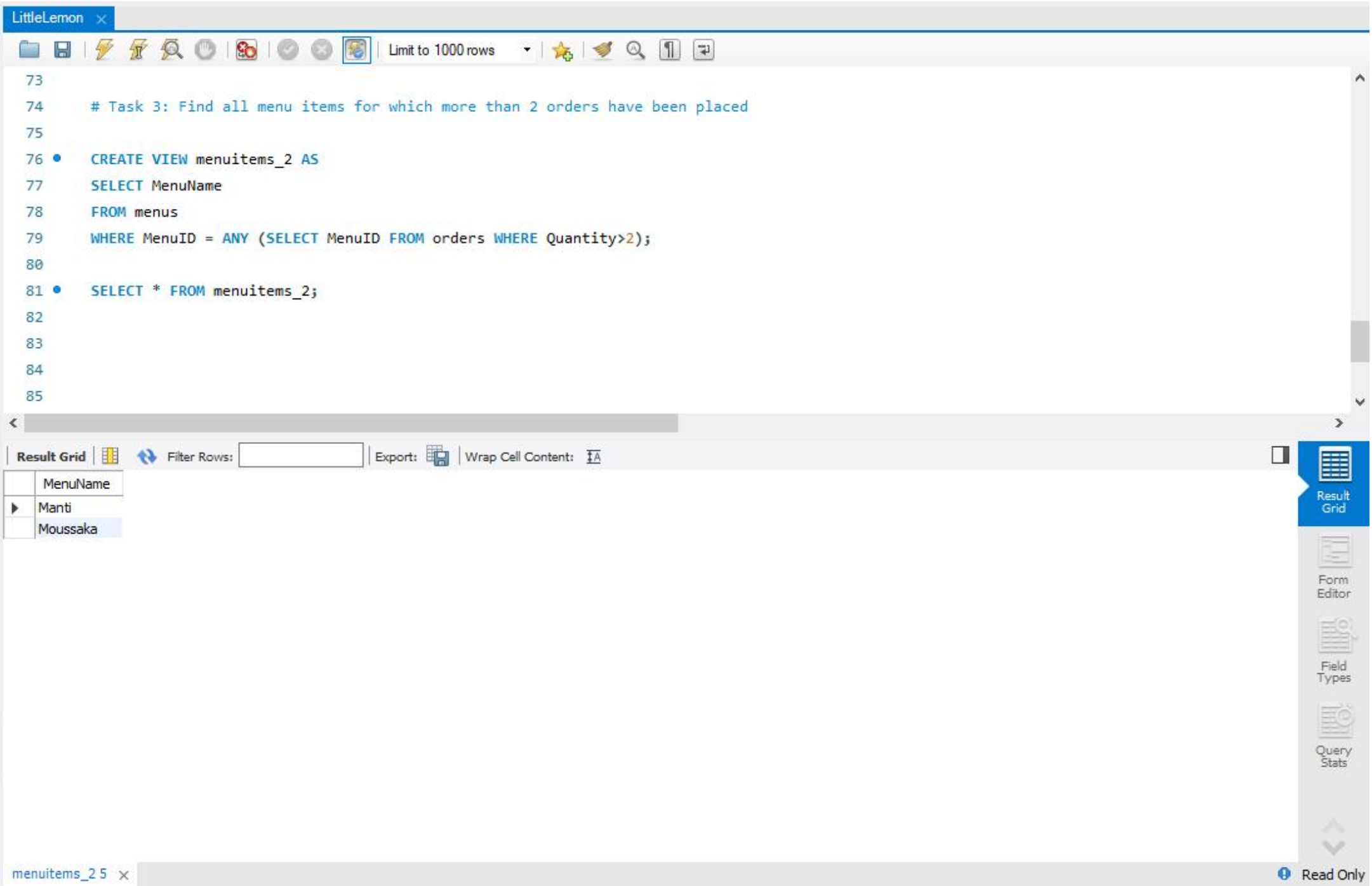
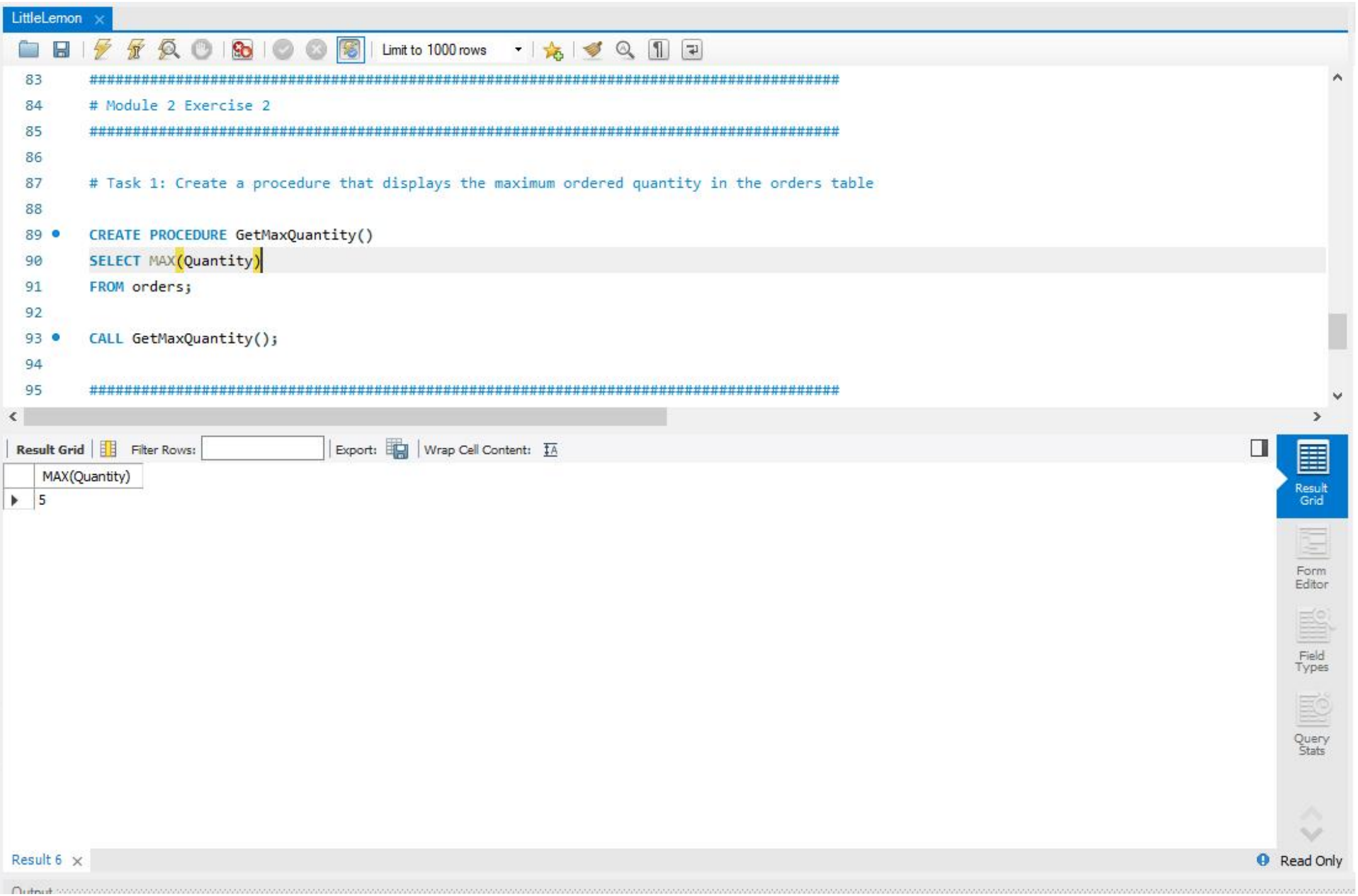
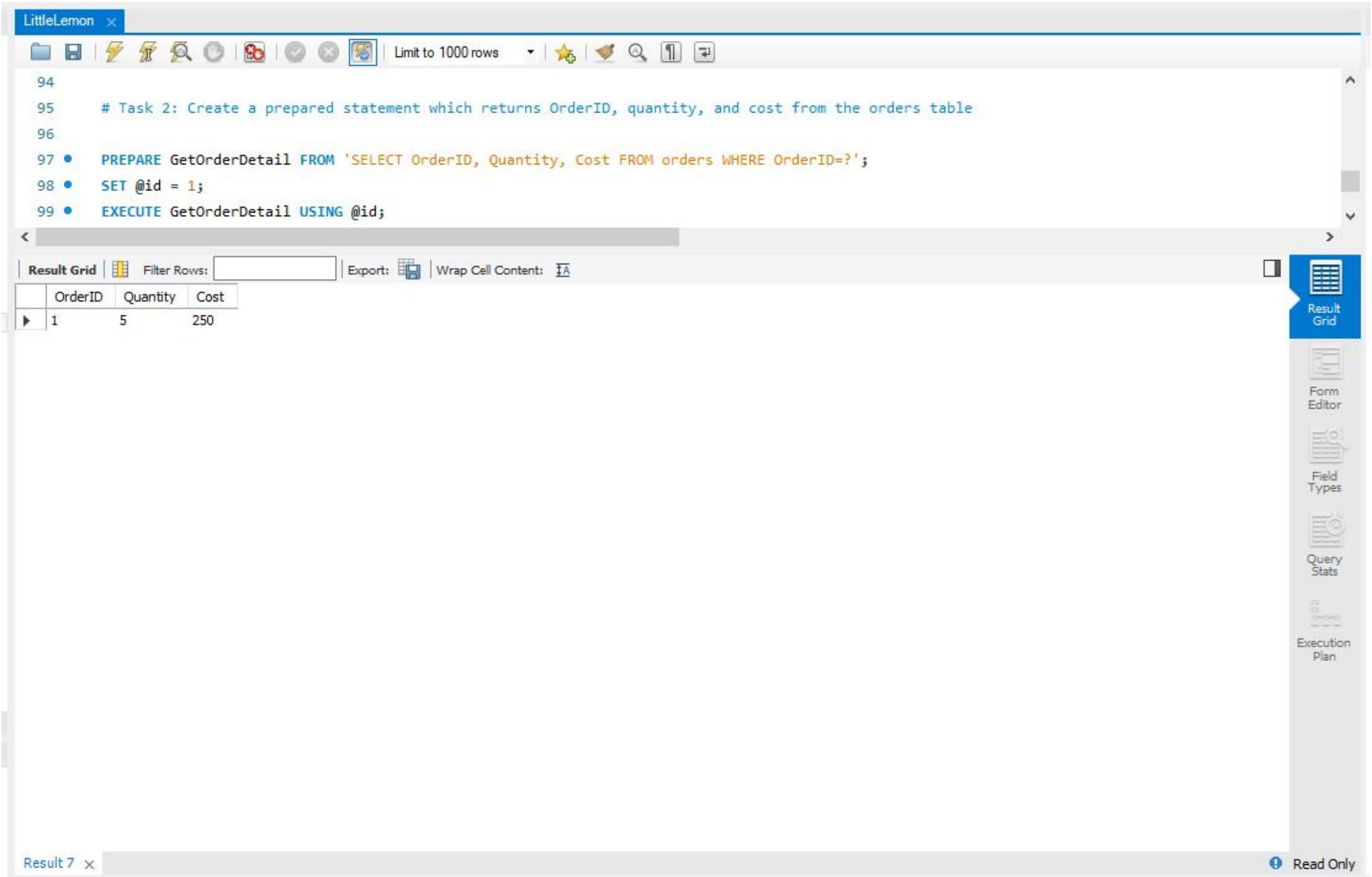

MySQL Workbench
MySQL Model (LittleLemonDM.mwb)
EER Diagram
File Edit View Arrange Model Database Tools Scripting Help
Bird's Eye
Zoom: 100%
Catalog Tree
LittleLemonDB
Tables
Views
Routine Groups
Catalog Layers User Types
Description Editor
No Selection
Description Properties
Diagram
Delivery
DeliveryID INT
DeliveryDate DATE
Status VARCHAR(45)
Indexes
Orders
OrderID INT
MenuID INT
CustomerID INT
Quantity INT
Cost DECIMAL
DeliveryID INT
Indexes
Customers
CustomerID INT
FullName VARCHAR(45)
ContactNumber VARCHAR(45)
Email VARCHAR(45)
Indexes
Menus
MenuID INT
MenuItemsID INT
MenuName VARCHAR(45)
Cuisine VARCHAR(45)
Indexes
Bookings
BookingID INT
BookingDate DATE
TableNumber INT
CustomerID INT
StaffID INT
Indexes
Staff
StaffID INT
Role VARCHAR(45)
Salary DECIMAL
Indexes
MenuItems
MenuItemsID INT
CourseName VARCHAR(45)
StarterName VARCHAR(45)
DessertName VARCHAR(45)
Indexes
Modeling Additions
timestamps
create_time, update_time
user
username, email, passwor...
category
category_id, name
Templates
Ready

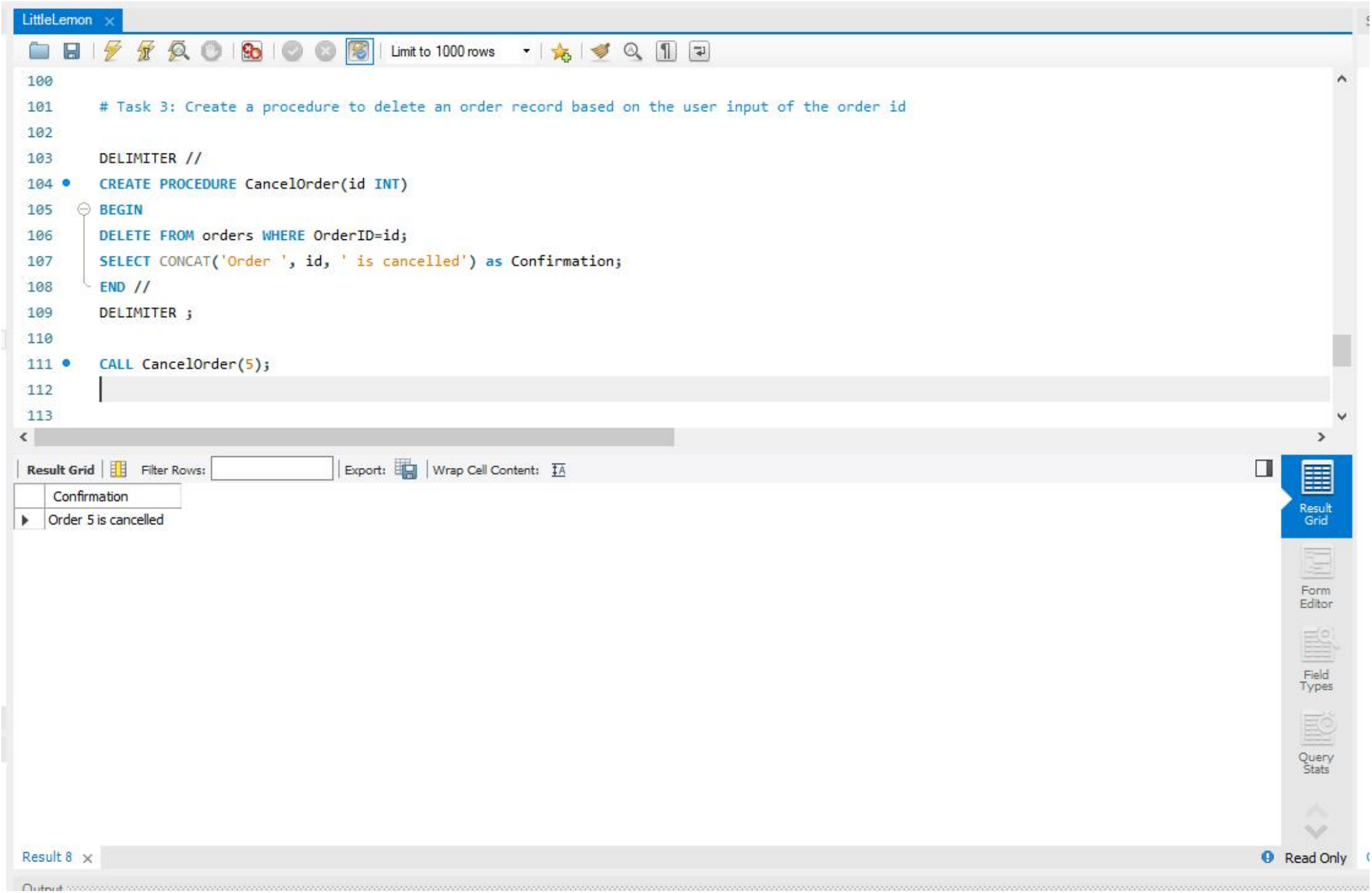

LittleLemon
Limit to 1000 rows
48 ##
49 # Module 2 Exercise 1
50 ##
51
52 # Task 1: Create a virtual table for orders
53
54 CREATE VIEW OrdersView AS
55 SELECT OrderID, Quantity, Cost
56 FROM orders;
57
58 SELECT * FROM OrdersView;
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
OrderID Quantity Cost
1 5 250
2 3 200
3 2 100
4 1 40
Result Grid
Form Editor
Field Types
Query Stats
Execution Plan
OrdersView 3
Read Only

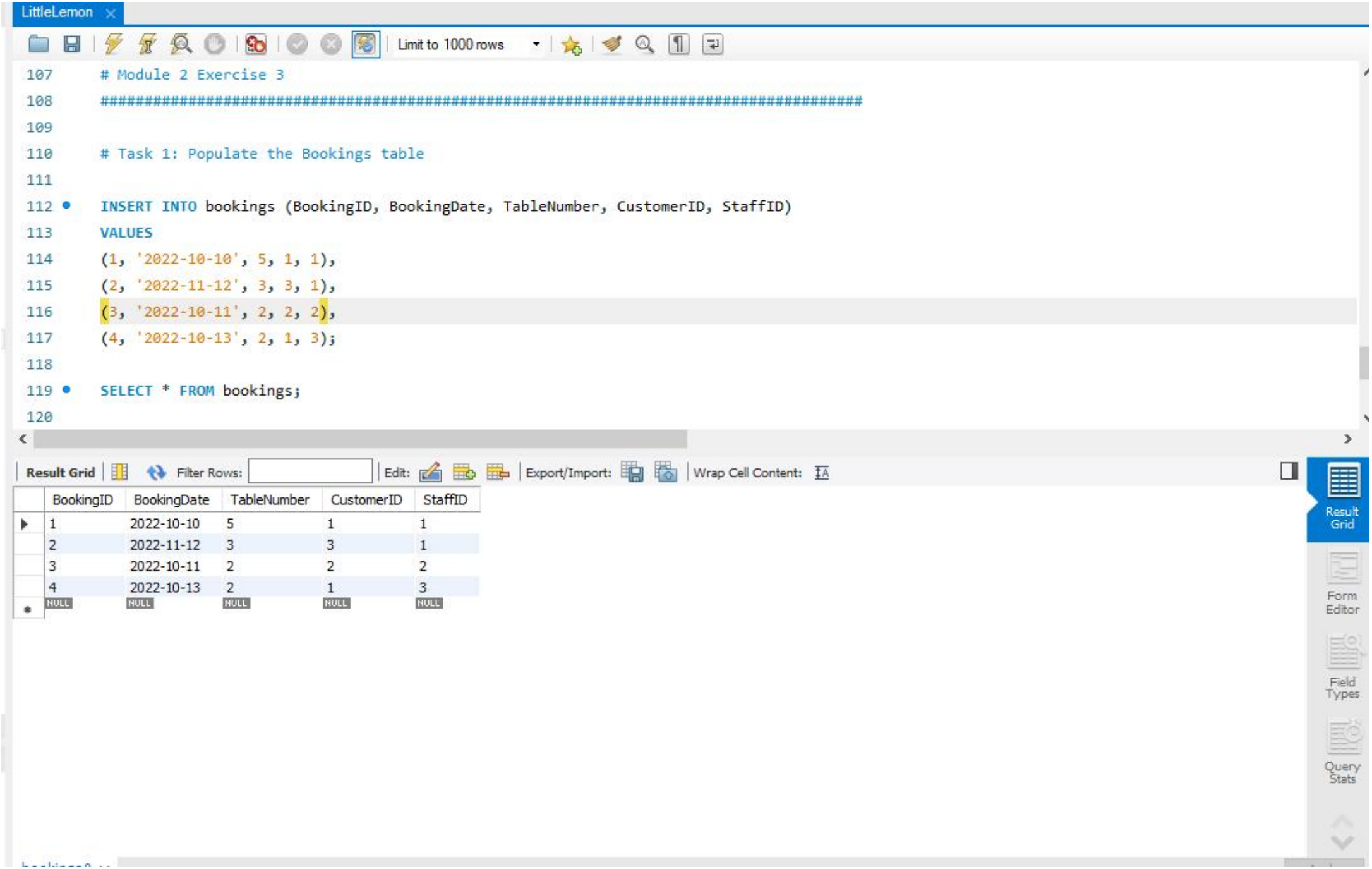

LittleLemon
Limit to 1000 rows
60 # Task 2: Extract information on all customers with orders that cost more than $150
61
62 CREATE VIEW customers_150 AS
63 SELECT a.CustomerID, a.FullName, b.OrderID, b.Cost, c.MenuName, d.CourseName
64 FROM customers as a
65 INNER JOIN orders as b
66 ON a.CustomerID=b.CustomerID AND b.COST>150
67 INNER JOIN menus as c
68 on b.MenuID=c.MenuID
69 INNER JOIN menuitems as d
70 on c.MenuItemsID=d.MenuItemsID;
71
72 SELECT * FROM customers_150;
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
CustomerID FullName OrderID Cost MenuName CourseName
1 Vanessa McCarthy 1 250 Manti Kabasa
2 Marcos Romero 2 200 Moussaka Greek salad
Result Grid
Form Editor
Field Types
Query Stats
customers_150 4
Read Only


LittleLemon
Limit to 1000 rows
73
74 # Task 3: Find all menu items for which more than 2 orders have been placed
75
76 CREATE VIEW menuitems_2 AS
77 SELECT MenuName
78 FROM menus
79 WHERE MenuID = ANY (SELECT MenuID FROM orders WHERE Quantity>2);
80
81 SELECT * FROM menuitems_2;
82
83
84
85
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
MenuName
Manti
Moussaka
Result Grid
Form Editor
Field Types
Query Stats
menuitems_2 5
Read Only


LittleLemon
Limit to 1000 rows
83 ##
84 # Module 2 Exercise 2
85 ##
86
87 # Task 1: Create a procedure that displays the maximum ordered quantity in the orders table
88
89 CREATE PROCEDURE GetMaxQuantity()
90 SELECT MAX(Quantity)
91 FROM orders;
92
93 CALL GetMaxQuantity();
94
95 ##
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
MAX(Quantity)
5
Result Grid
Form Editor
Field Types
Query Stats
Result 6
Read Only
Output


LittleLemon
Limit to 1000 rows
94
95 # Task 2: Create a prepared statement which returns OrderID, quantity, and cost from the orders table
96
97 PREPARE GetOrderDetail FROM 'SELECT OrderID, Quantity, Cost FROM orders WHERE OrderID=?';
98 SET @id = 1;
99 EXECUTE GetOrderDetail USING @id;
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
OrderID Quantity Cost
1 5 250
Result Grid
Form Editor
Field Types
Query Stats
Execution Plan
Result 7
Read Only


LittleLemon
Limit to 1000 rows
100
101 # Task 3: Create a procedure to delete an order record based on the user input of the order id
102
103 DELIMITER //
104 CREATE PROCEDURE CancelOrder(id INT)
105 BEGIN
106 DELETE FROM orders WHERE OrderID=id;
107 SELECT CONCAT('Order ', id, ' is cancelled') as Confirmation;
108 END //
109 DELIMITER ;
110
111 CALL CancelOrder(5);
112
113
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Confirmation
Order 5 is cancelled
Result Grid
Form Editor
Field Types
Query Stats
Result 8
Read Only
Output


LittleLemon
Limit to 1000 rows
107 # Module 2 Exercise 3
108 ##
109
110 # Task 1: Populate the Bookings table
111
112 INSERT INTO bookings (BookingID, BookingDate, TableNumber, CustomerID, StaffID)
113 VALUES
114 (1, '2022-10-10', 5, 1, 1),
115 (2, '2022-11-12', 3, 3, 1),
116 (3, '2022-10-11', 2, 2, 2),
117 (4, '2022-10-13', 2, 1, 3);
118
119 SELECT * FROM bookings;
120
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
BookingID BookingDate TableNumber CustomerID StaffID
1 2022-10-10 5 1 1
2 2022-11-12 3 3 1
3 2022-10-11 2 2 2
4 2022-10-13 2 1 3
NULL NULL NULL NULL NULL
Result Grid
Form Editor
Field Types
Query Stats

LittleLemon

```
121    # Task 2: Create a procedure to check whether a table in the restaurant is already booked
122
123    DELIMITER //
124    CREATE PROCEDURE CheckBooking(checkdate DATE, checktable INT)
125    BEGIN
126    DROP TABLE IF EXISTS BookingStatus;
127    CREATE TABLE BookingStatus AS
128    SELECT CONCAT('Table ', checktable, ' is already booked') as BookingStatus,
129    COUNT(*) AS booking FROM bookings WHERE BookingDate=checkdate and TableNumber=checktable;
130    UPDATE BookingStatus
131    SET BookingStatus=CONCAT('Table ', checktable, ' is not booked')
132    WHERE booking=0;
133    SELECT BookingStatus FROM BookingStatus;
134    END //
135    DELIMITER ;
136
137    CALL CheckBooking('2022-11-12', 3);
138
```

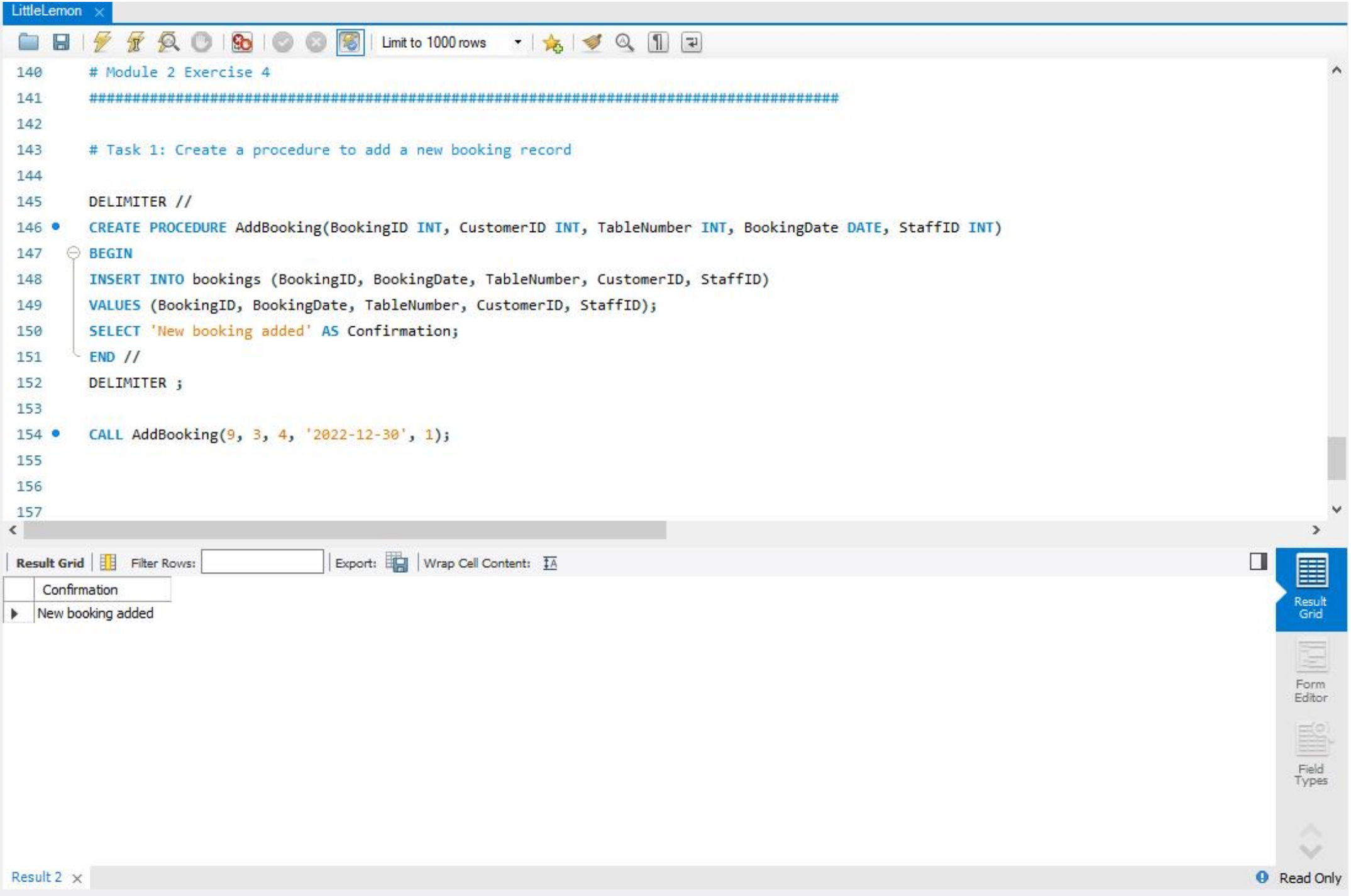
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

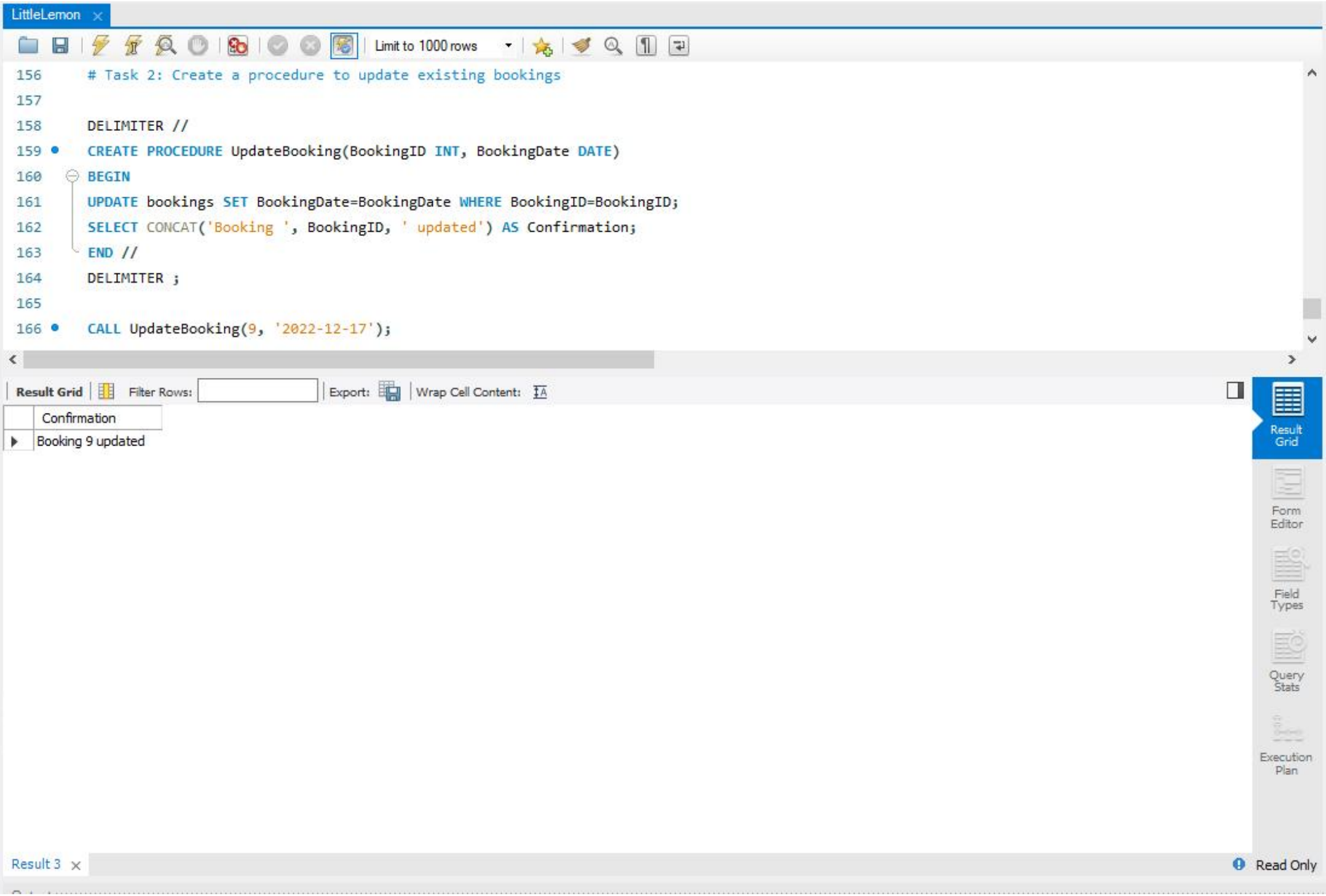
BookingStatus
Table 3 is already booked

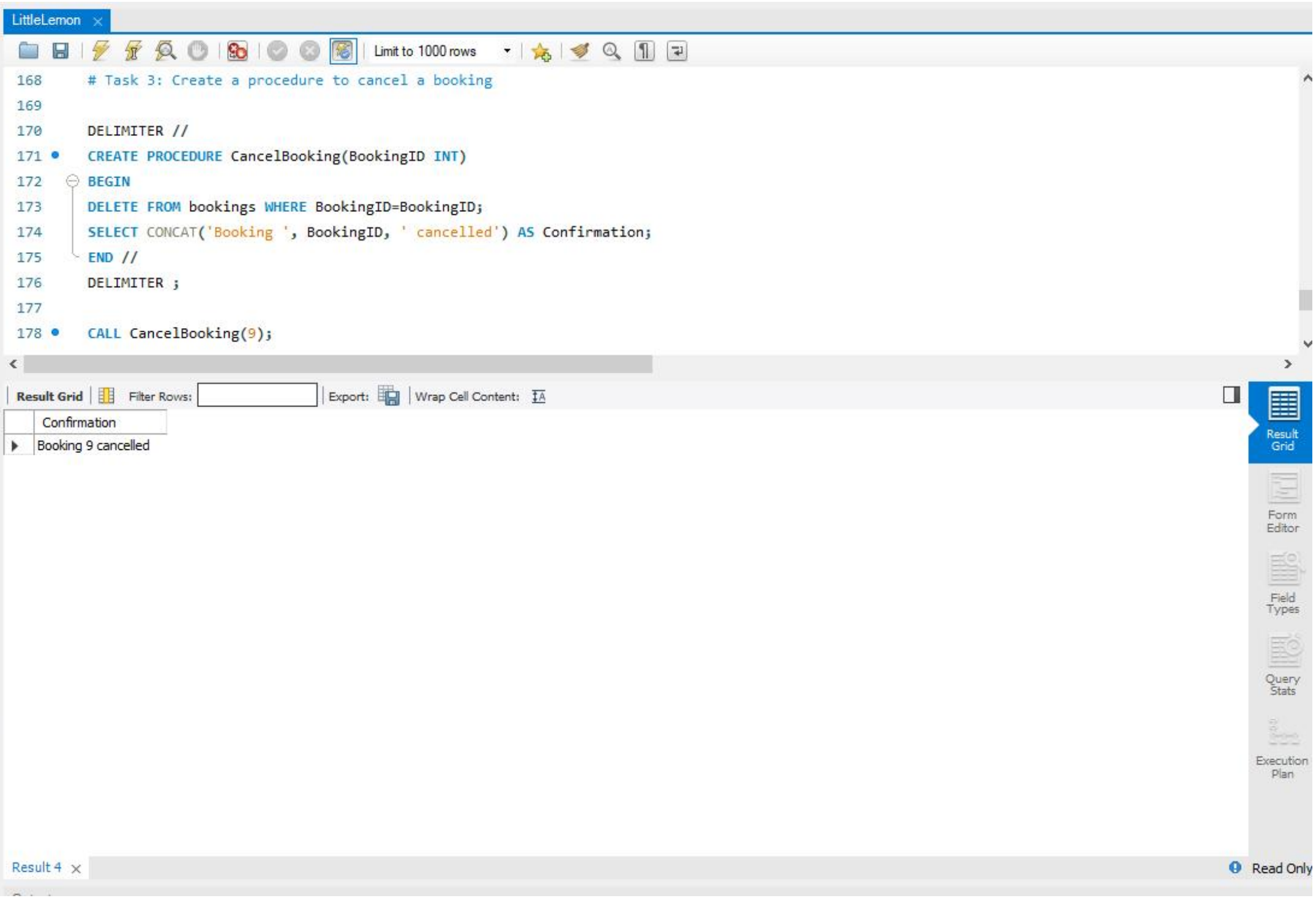
Result Grid
Form Editor
Field Types

Result 1

Read Only


LittleLemon
Limit to 1000 rows
Module 2 Exercise 4
##
Task 1: Create a procedure to add a new booking record
DELIMITER //
CREATE PROCEDURE AddBooking(BookingID INT, CustomerID INT, TableNumber INT, BookingDate DATE, StaffID INT)
BEGIN
INSERT INTO bookings (BookingID, BookingDate, TableNumber, CustomerID, StaffID)
VALUES (BookingID, BookingDate, TableNumber, CustomerID, StaffID);
SELECT 'New booking added' AS Confirmation;
END //
DELIMITER ;
CALL AddBooking(9, 3, 4, '2022-12-30', 1);
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Confirmation
New booking added
Result Grid
Form Editor
Field Types
Result 2
Read Only


LittleLemon
Limit to 1000 rows
156 # Task 2: Create a procedure to update existing bookings
157
158 DELIMITER //
159 CREATE PROCEDURE UpdateBooking(BookingID INT, BookingDate DATE)
160 BEGIN
161 UPDATE bookings SET BookingDate=BookingDate WHERE BookingID=BookingID;
162 SELECT CONCAT('Booking ', BookingID, ' updated') AS Confirmation;
163 END //
164 DELIMITER ;
165
166 CALL UpdateBooking(9, '2022-12-17');
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Confirmation
Booking 9 updated
Result Grid
Form Editor
Field Types
Query Stats
Execution Plan
Result 3
Read Only


LittleLemon
Limit to 1000 rows
Task 3: Create a procedure to cancel a booking
DELIMITER //
CREATE PROCEDURE CancelBooking(BookingID INT)
BEGIN
DELETE FROM bookings WHERE BookingID=BookingID;
SELECT CONCAT('Booking ', BookingID, ' cancelled') AS Confirmation;
END //
DELIMITER ;
CALL CancelBooking(9);
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Confirmation
Booking 9 cancelled
Result Grid
Form Editor
Field Types
Query Stats
Execution Plan
Result 4
Read Only