

Create a new procedure called AddBooking to add a new table booking record.

The screenshot shows the MySQL Workbench interface with the 'Query 1' editor. The SQL code defines a procedure named 'AddBooking' that takes four input parameters: 'booking_id' (INT), 'booking_date' (DATE), 'table_num' (INT), and 'customer_id' (INT). The procedure body includes an 'INSERT INTO Bookings' statement with values from the parameters and a 'select * from Bookings;' statement. The procedure is then called with the parameters (13, '2022-12-30', 3, 2).

```
1 USE littlelemondb;
2 DELIMITER //
3 CREATE PROCEDURE AddBooking(IN booking_id INT,IN booking_date DATE,IN table_num INT,IN customer_id INT)
4 BEGIN
5     INSERT INTO Bookings (bookingID, bookingDate, tableNumber, customerID)
6     VALUES (booking_id, booking_date, table_num, customer_id);
7     select * from Bookings;
8 END //
9 DELIMITER ;
10 CALL AddBooking(13,'2022-12-30',3, 2);
```

The 'Result Grid' shows the output of the procedure call, displaying the booking details for the new record.

bookingID	bookingDate	tableNumber	customerID
1	2022-10-10	5	1
2	2022-11-12	3	3
3	2022-10-11	2	2
4	2022-10-13	2	1
5	2022-12-17	5	1

The 'Output' pane shows the execution log, including the creation of the procedure and the successful call.

#	Time	Action	Message	Duration / Fetch
179	18:31:11	drop procedure AddBooking;	0 row(s) affected	0.000 sec
180	18:31:31	CREATE PROCEDURE AddBooking(IN booking_id INT, IN booking_date DATE, IN table_num INT, ...	0 row(s) affected	0.016 sec
181	18:31:39	CALL AddBooking(12,'2022-12-30',3, 2);	1 row(s) affected	0.000 sec
182	18:32:39	drop procedure AddBooking;	0 row(s) affected	0.000 sec
183	18:32:52	CREATE PROCEDURE AddBooking(IN booking_id INT, IN booking_date DATE, IN table_num INT, ...	0 row(s) affected	0.000 sec
184	18:32:59	CALL AddBooking(13,'2022-12-30',3, 2);	10 row(s) returned	0.016 sec / 0.000 sec

Create a new procedure called UpdateBooking that they can use to update existing bookings in the booking table.

The procedure should have two input parameters in the form of booking id and booking date.

The screenshot shows the MySQL Workbench interface with the 'Query 1' editor. The SQL code defines a procedure named 'UpdateBooking' that takes two input parameters: 'booking_id' (INT) and 'booking_date' (DATE). The procedure body includes an 'UPDATE Bookings' statement that sets 'bookingDate' to 'booking_date' where 'bookingID' equals 'booking_id'. The procedure is then called with the parameters (9, '2022-12-17').

```
1 USE littlelemondb;
2 DELIMITER //
3 CREATE PROCEDURE UpdateBooking(IN booking_id INT,IN booking_date DATE)
4 BEGIN
5     UPDATE Bookings
6     SET bookingDate = booking_date
7     WHERE bookingID = booking_id;
8     select * from Bookings;
9 END //
10 DELIMITER ;
11 CALL UpdateBooking(9,'2022-12-17');
```

The 'Result Grid' shows the output of the procedure call, displaying the updated booking details.

bookingID	bookingDate	tableNumber	customerID
3	2022-10-11	2	2
4	2022-10-13	2	1
5	2022-12-17	5	1
9	2022-12-17	5	1
10	2022-12-30	5	1

The 'Output' pane shows the execution log, including the creation of the procedure and the successful call.

#	Time	Action	Message	Duration / Fetch
184	18:32:59	CALL AddBooking(13,'2022-12-30',3, 2);	10 row(s) returned	0.016 sec / 0.000 sec
185	18:45:27	CREATE PROCEDURE UpdateBooking(IN booking_id INT,IN booking_date DATE) BEGIN UPDATE Bookin...	0 row(s) affected	0.016 sec
186	18:47:19	CALL UpdateBooking(9,'2022-12-17');	1 row(s) affected	0.015 sec
187	18:48:20	drop procedure UpdateBooking;	0 row(s) affected	0.015 sec
188	18:48:31	CREATE PROCEDURE UpdateBooking(IN booking_id INT,IN booking_date DATE) BEGIN UPDATE Bookin...	0 row(s) affected	0.000 sec
189	18:48:33	CALL UpdateBooking(9,'2022-12-17');	10 row(s) returned	0.000 sec / 0.000 sec

create a new procedure called CancelBooking that they can use to cancel or remove a booking.

The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'SCHEMAS' tree with 'littlelemondb' selected. The central editor shows a SQL query to create a stored procedure named 'CancelBooking' that takes a booking ID and deletes the corresponding record from the 'Bookings' table, returning a confirmation message. The query is as follows:

```
1 USE littlelemondb;
2 DELIMITER //
3 CREATE PROCEDURE CancelBooking(IN booking_id INT)
4 BEGIN
5     DECLARE cancelled_booking VARCHAR(100);
6     DELETE FROM Bookings
7     WHERE bookingId = booking_id;
8     SET cancelled_booking = CONCAT("Booking ", booking_id, " has been cancelled");
9     SELECT cancelled_booking as Confirmation;
10 END //
11 DELIMITER ;
12 CALL CancelBooking(8);
```

Below the query editor, the 'Result Grid' shows the output of the procedure call, displaying the confirmation message: 'Booking 8 has been cancelled'.

The bottom pane shows the 'Output' tab with a table of execution results:

#	Time	Action	Message	Duration / Fetch
191	19.04.30	CREATE PROCEDURE CancelBooking(IN booking_id INT) BEGIN DELETE FROM Bookings WHERE...	Error Code: 1193. Unknown system variable 'cancelled_booking'	0.016 sec
192	19.06.04	CREATE PROCEDURE CancelBooking(IN booking_id INT) BEGIN DECLARE cancelled_booking VARCHAR(100);	0 row(s) affected	0.016 sec
193	19.06.06	CALL CancelBooking(8);	1 row(s) returned	0.000 sec / 0.000 sec
194	19.06.57	drop procedure CancelBooking;	0 row(s) affected	0.016 sec
195	19.07.06	CREATE PROCEDURE CancelBooking(IN booking_id INT) BEGIN DECLARE cancelled_booking VARCHAR(100);	0 row(s) affected	0.016 sec
196	19.07.08	CALL CancelBooking(8);	1 row(s) returned	0.000 sec / 0.000 sec