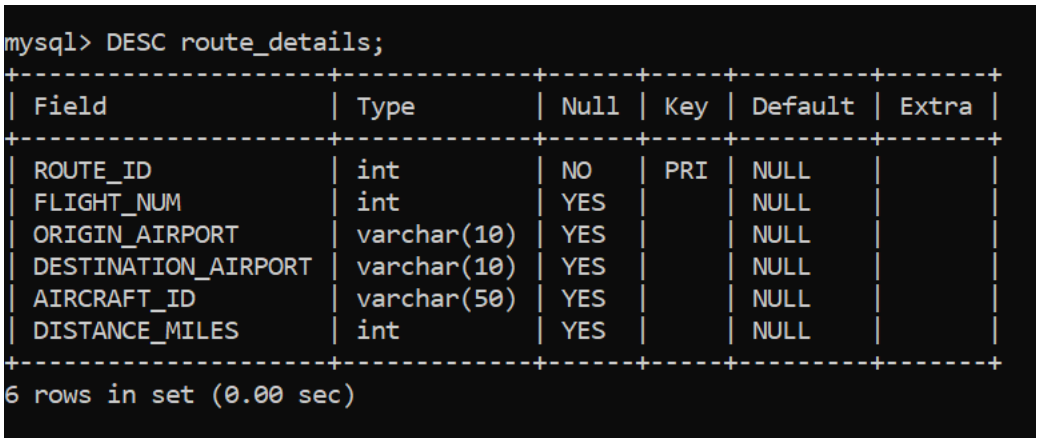
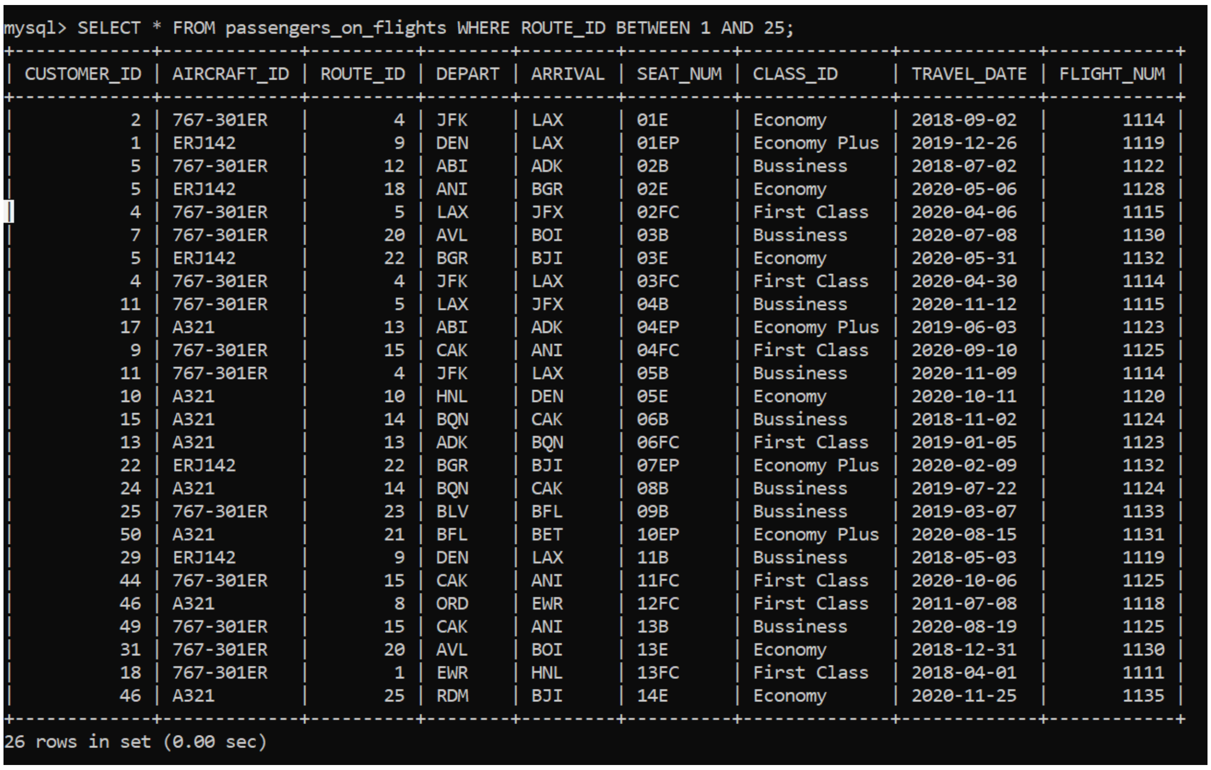
Write a query to create a route_details table using suitable data types for the fields, such as route_id, flight_num, origin_airport, destination_airport, aircraft_id, and distance_miles. Implement the check constraint for the flight number and unique constraint for the route_id fields. Also, make sure that the distance miles field is greater than 0.



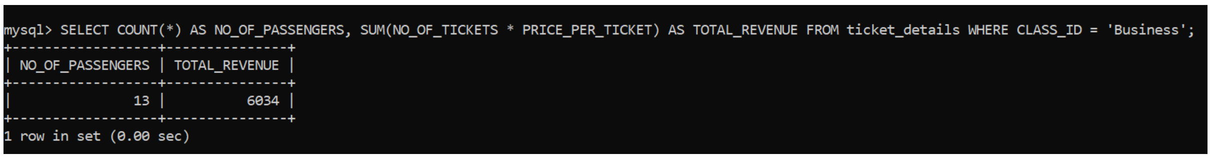
```
mysql> DESC route_details;
+---------------------+-------------+------+-----+---------+-------+
| Field               | Type        | Null | Key | Default | Extra |
+---------------------+-------------+------+-----+---------+-------+
| ROUTE_ID            | int         | NO   | PRI | NULL    |       |
| FLIGHT_NUM          | int         | YES  |     | NULL    |       |
| ORIGIN_AIRPORT      | varchar(10) | YES  |     | NULL    |       |
| DESTINATION_AIRPORT | varchar(10) | YES  |     | NULL    |       |
| AIRCRAFT_ID         | varchar(50) | YES  |     | NULL    |       |
| DISTANCE_MILES      | int         | YES  |     | NULL    |       |
+---------------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

Write a query to display all the passengers (customers) who have travelled in routes 01 to 25. Take data from the passengers_on_flights table.



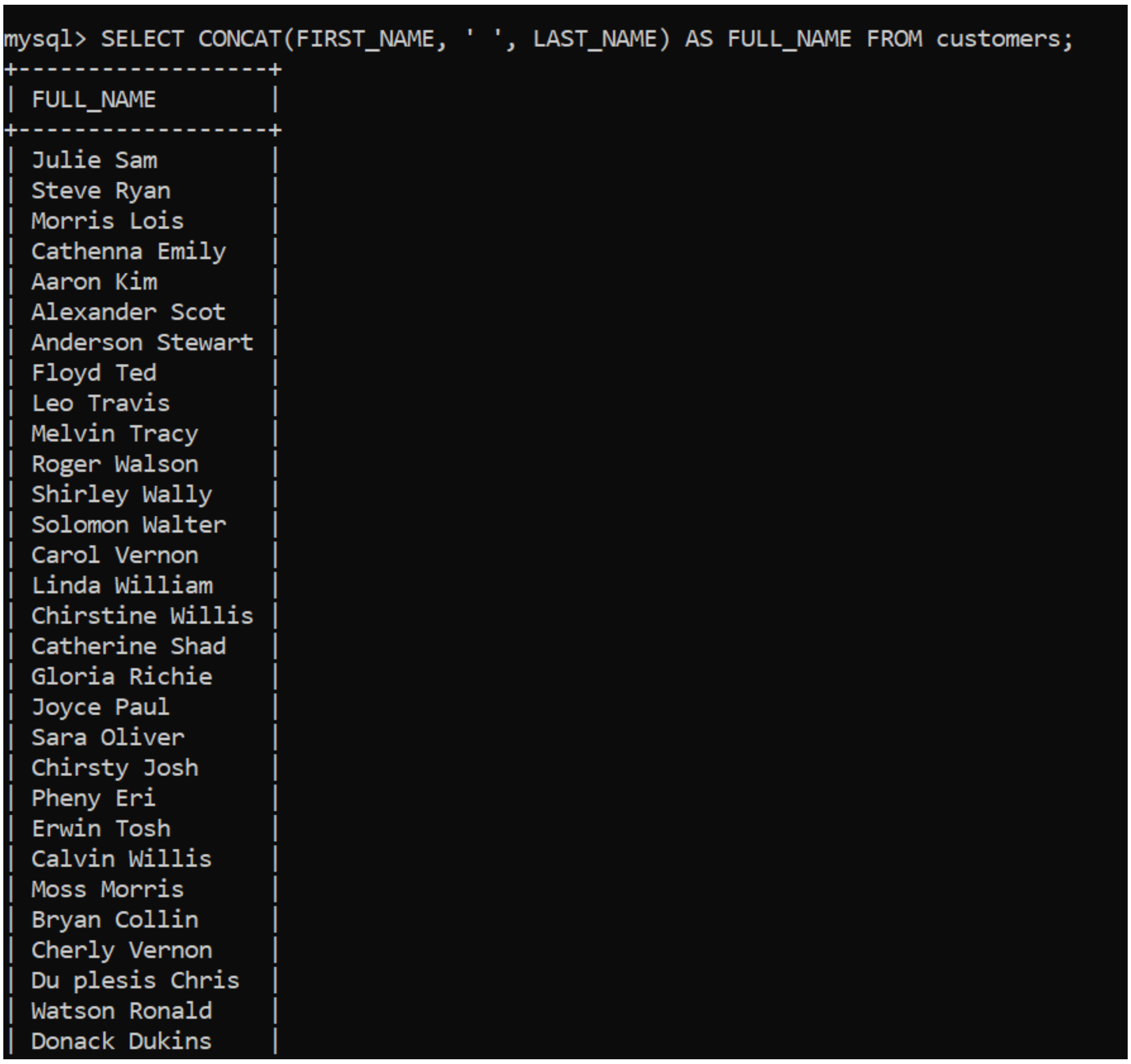
```
mysql> SELECT * FROM passengers_on_flights WHERE ROUTE_ID BETWEEN 1 AND 25;
+-------------+-------------+----------+--------+---------+----------+--------------+-------------+------------+
| CUSTOMER_ID | AIRCRAFT_ID | ROUTE_ID | DEPART | ARRIVAL | SEAT_NUM | CLASS_ID     | TRAVEL_DATE | FLIGHT_NUM |
+-------------+-------------+----------+--------+---------+----------+--------------+-------------+------------+
|           2 | 767-301ER   |        4 | JFK    | LAX     | 01E      | Economy      | 2018-09-02  |       1114 |
|           1 | ERJ142      |        9 | DEN    | LAX     | 01EP     | Economy Plus | 2019-12-26  |       1119 |
|           5 | 767-301ER   |       12 | ABI    | ADK     | 02B      | Bussiness    | 2018-07-02  |       1122 |
|           5 | ERJ142      |       18 | ANI    | BGR     | 02E      | Economy      | 2020-05-06  |       1128 |
|           4 | 767-301ER   |        5 | LAX    | JFX     | 02FC     | First Class  | 2020-04-06  |       1115 |
|           7 | 767-301ER   |       20 | AVL    | BOI     | 03B      | Bussiness    | 2020-07-08  |       1130 |
|           5 | ERJ142      |       22 | BGR    | BJI     | 03E      | Economy      | 2020-05-31  |       1132 |
|           4 | 767-301ER   |        4 | JFK    | LAX     | 03FC     | First Class  | 2020-04-30  |       1114 |
|          11 | 767-301ER   |        5 | LAX    | JFX     | 04B      | Bussiness    | 2020-11-12  |       1115 |
|          17 | A321        |       13 | ABI    | ADK     | 04EP     | Economy Plus | 2019-06-03  |       1123 |
|           9 | 767-301ER   |       15 | CAK    | ANI     | 04FC     | First Class  | 2020-09-10  |       1125 |
|          11 | 767-301ER   |        4 | JFK    | LAX     | 05B      | Bussiness    | 2020-11-09  |       1114 |
|          10 | A321        |       10 | HNL    | DEN     | 05E      | Economy      | 2020-10-11  |       1120 |
|          15 | A321        |       14 | BQN    | CAK     | 06B      | Bussiness    | 2018-11-02  |       1124 |
|          13 | A321        |       13 | ADK    | BQN     | 06FC     | First Class  | 2019-01-05  |       1123 |
|          22 | ERJ142      |       22 | BGR    | BJI     | 07EP     | Economy Plus | 2020-02-09  |       1132 |
|          24 | A321        |       14 | BQN    | CAK     | 08B      | Bussiness    | 2019-07-22  |       1124 |
|          25 | 767-301ER   |       23 | BLV    | BFL     | 09B      | Bussiness    | 2019-03-07  |       1133 |
|          50 | A321        |       21 | BFL    | BET     | 10EP     | Economy Plus | 2020-08-15  |       1131 |
|          29 | ERJ142      |        9 | DEN    | LAX     | 11B      | Bussiness    | 2018-05-03  |       1119 |
|          44 | 767-301ER   |       15 | CAK    | ANI     | 11FC     | First Class  | 2020-10-06  |       1125 |
|          46 | A321        |        8 | ORD    | EWR     | 12FC     | First Class  | 2011-07-08  |       1118 |
|          49 | 767-301ER   |       15 | CAK    | ANI     | 13B      | Bussiness    | 2020-08-19  |       1125 |
|          31 | 767-301ER   |       20 | AVL    | BOI     | 13E      | Economy      | 2018-12-31  |       1130 |
|          18 | 767-301ER   |        1 | EWR    | HNL     | 13FC     | First Class  | 2018-04-01  |       1111 |
|          46 | A321        |       25 | RDM    | BJI     | 14E      | Economy      | 2020-11-25  |       1135 |
+-------------+-------------+----------+--------+---------+----------+--------------+-------------+------------+
26 rows in set (0.00 sec)
```

Write a query to identify the number of passengers and total revenue in business class from the ticket_details table.



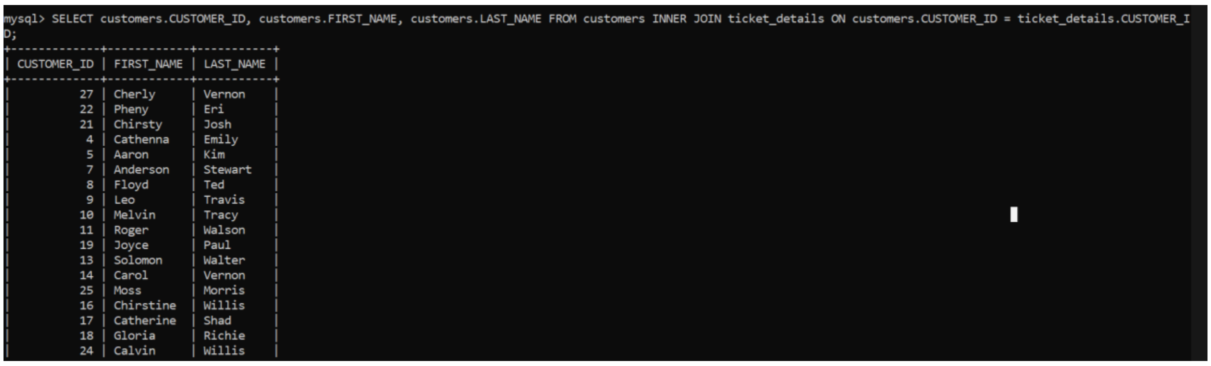
```
mysql> SELECT COUNT(*) AS NO_OF_PASSENGERS, SUM(NO_OF_TICKETS * PRICE_PER_TICKET) AS TOTAL_REVENUE FROM ticket_details WHERE CLASS_ID = 'Business';
+------------------+---------------+
| NO_OF_PASSENGERS | TOTAL_REVENUE |
+------------------+---------------+
|               13 |          6034 |
+------------------+---------------+
1 row in set (0.00 sec)
```

Write a query to display the full name of the customer by extracting the first name and last name from the customer table.



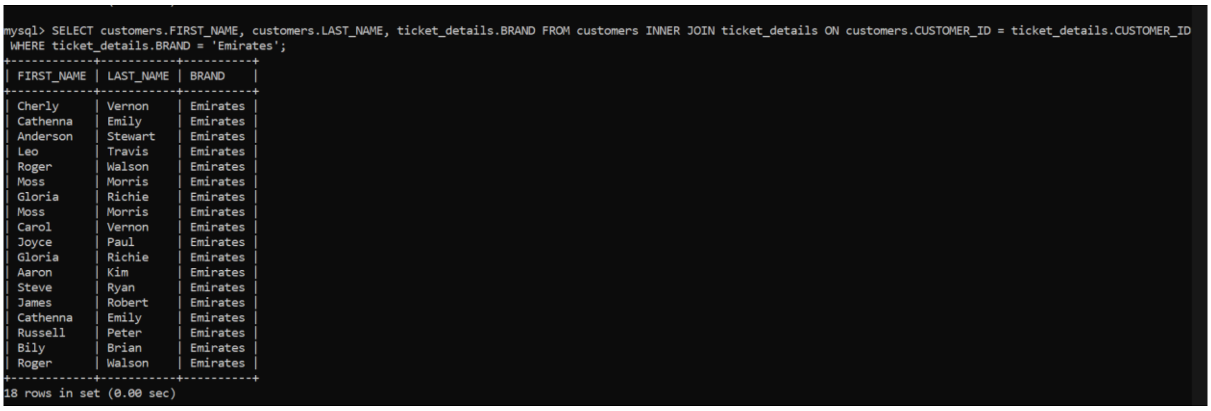
```
mysql> SELECT CONCAT(FIRST_NAME, ' ', LAST_NAME) AS FULL_NAME FROM customers;
+-------------------+
| FULL_NAME         |
+-------------------+
| Julie Sam         |
| Steve Ryan        |
| Morris Lois       |
| Cathenna Emily    |
| Aaron Kim         |
| Alexander Scot    |
| Anderson Stewart  |
| Floyd Ted         |
| Leo Travis        |
| Melvin Tracy      |
| Roger Walson      |
| Shirley Wally     |
| Solomon Walter    |
| Carol Vernon      |
| Linda William     |
| Chirstine Willis  |
| Catherine Shad    |
| Gloria Richie     |
| Joyce Paul        |
| Sara Oliver       |
| Chirsty Josh      |
| Pheny Eri         |
| Erwin Tosh        |
| Calvin Willis     |
| Moss Morris       |
| Bryan Collin      |
| Cherly Vernon     |
| Du plesis Chris   |
| Watson Ronald     |
| Donack Dukins     |
```

Write a query to extract the customers who have registered and booked a ticket. Use data from the customer and ticket_details tables



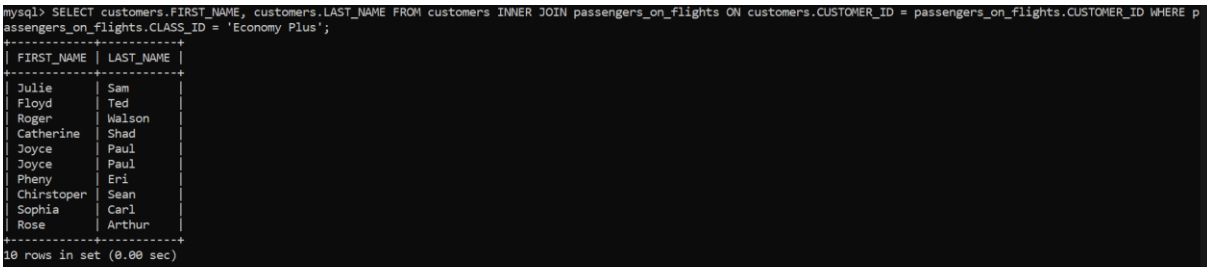
```
mysql> SELECT customers.CUSTOMER_ID, customers.FIRST_NAME, customers.LAST_NAME FROM customers INNER JOIN ticket_details ON customers.CUSTOMER_ID = ticket_details.CUSTOMER_I
D;
+-------------+------------+-----------+
| CUSTOMER_ID | FIRST_NAME | LAST_NAME |
+-------------+------------+-----------+
|          27 | Cherly     | Vernon    |
|          22 | Pheny      | Eri       |
|          21 | Chirsty    | Josh      |
|           4 | Cathenna   | Emily     |
|           5 | Aaron      | Kim       |
|           7 | Anderson   | Stewart   |
|           8 | Floyd      | Ted       |
|           9 | Leo        | Travis    |
|          10 | Melvin     | Tracy     |
|          11 | Roger      | Walson    |
|          19 | Joyce      | Paul      |
|          13 | Solomon    | Walter    |
|          14 | Carol      | Vernon    |
|          25 | Moss       | Morris    |
|          16 | Chirstine  | Willis    |
|          17 | Catherine  | Shad      |
|          18 | Gloria     | Richie    |
|          24 | Calvin     | Willis    |
```

Write a query to identify the customer's first name and last name based on their customer ID and brand (Emirates) from the ticket_details table.



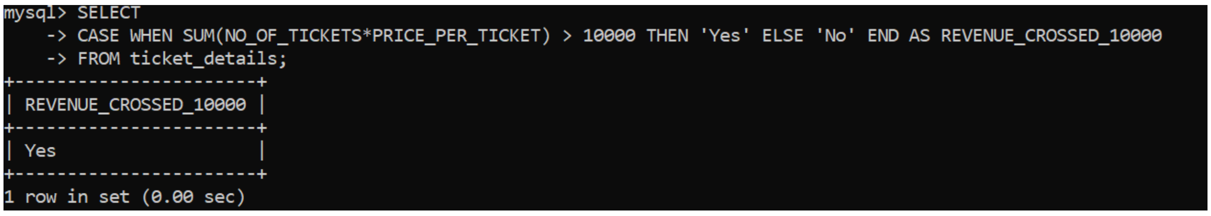
```
mysql> SELECT customers.FIRST_NAME, customers.LAST_NAME, ticket_details.BRAND FROM customers INNER JOIN ticket_details ON customers.CUSTOMER_ID = ticket_details.CUSTOMER_ID
 WHERE ticket_details.BRAND = 'Emirates';
+------------+-----------+----------+
| FIRST_NAME | LAST_NAME | BRAND    |
+------------+-----------+----------+
| Cherly     | Vernon    | Emirates |
| Cathenna   | Emily     | Emirates |
| Anderson   | Stewart   | Emirates |
| Leo        | Travis    | Emirates |
| Roger      | Walson    | Emirates |
| Moss       | Morris    | Emirates |
| Gloria     | Richie    | Emirates |
| Moss       | Morris    | Emirates |
| Carol      | Vernon    | Emirates |
| Joyce      | Paul      | Emirates |
| Gloria     | Richie    | Emirates |
| Aaron      | Kim       | Emirates |
| Steve      | Ryan      | Emirates |
| James      | Robert    | Emirates |
| Cathenna   | Emily     | Emirates |
| Russell    | Peter     | Emirates |
| Bily       | Brian     | Emirates |
| Roger      | Walson    | Emirates |
+------------+-----------+----------+
18 rows in set (0.00 sec)
```

Write a query to identify the customers who have travelled by *Economy Plus* class using Group By and Having clause on the passengers_on_flights table.



```
mysql> SELECT customers.FIRST_NAME, customers.LAST_NAME FROM customers INNER JOIN passengers_on_flights ON customers.CUSTOMER_ID = passengers_on_flights.CUSTOMER_ID WHERE p
assengers_on_flights.CLASS_ID = 'Economy Plus';
+------------+-----------+
| FIRST_NAME | LAST_NAME |
+------------+-----------+
| Julie      | Sam       |
| Floyd      | Ted       |
| Roger      | Walson    |
| Catherine  | Shad      |
| Joyce      | Paul      |
| Joyce      | Paul      |
| Pheny      | Eri       |
| Chirstoper | Sean      |
| Sophia     | Carl      |
| Rose       | Arthur    |
+------------+-----------+
10 rows in set (0.00 sec)
```

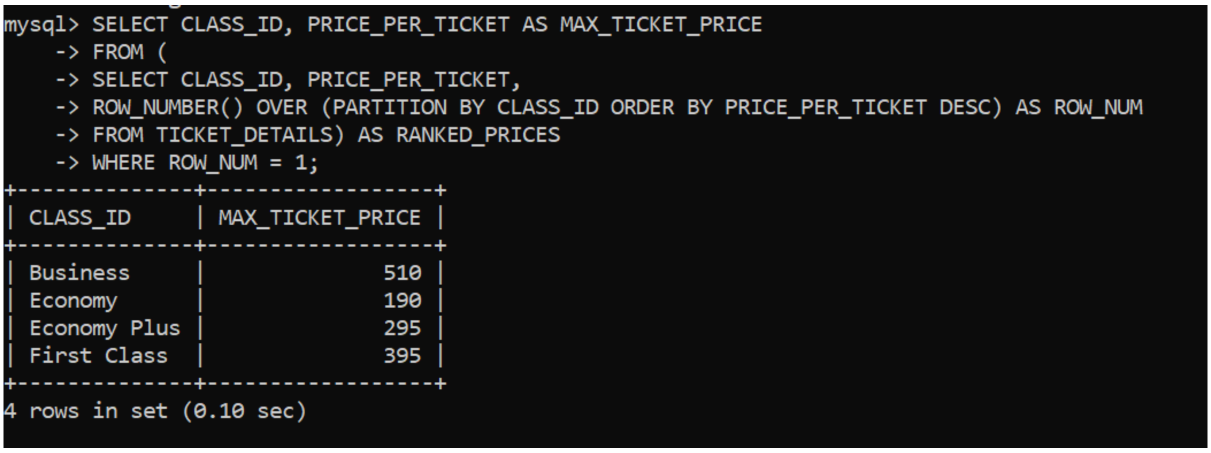
Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket_details table.



```
mysql> SELECT
    -> CASE WHEN SUM(NO_OF_TICKETS*PRICE_PER_TICKET) > 10000 THEN 'Yes' ELSE 'No' END AS REVENUE_CROSSED_10000
    -> FROM ticket_details;
+-----------------------+
| REVENUE_CROSSED_10000 |
+-----------------------+
| Yes                   |
+-----------------------+
1 row in set (0.00 sec)
```

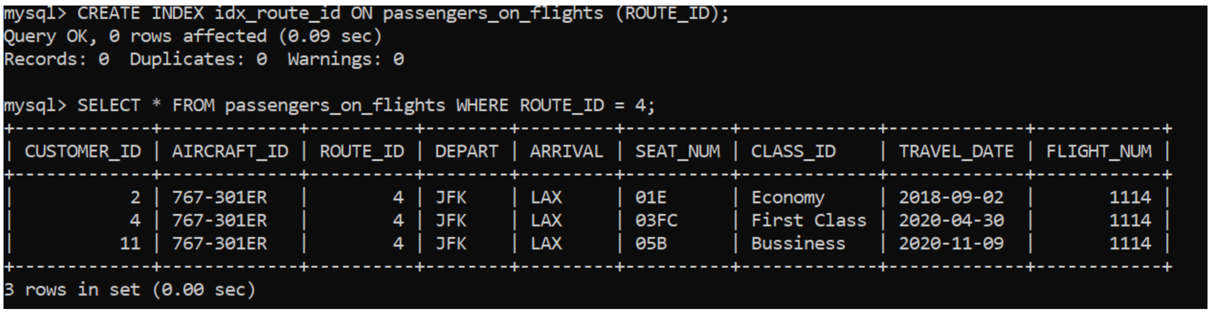
Write a query to create and grant access to a new user to perform operations on a database.

Write a query to find the maximum ticket price for each class using window functions on the ticket_details table.



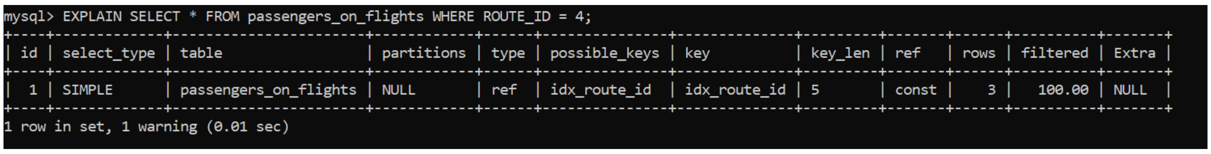
```
mysql> SELECT CLASS_ID, PRICE_PER_TICKET AS MAX_TICKET_PRICE
    -> FROM (
    -> SELECT CLASS_ID, PRICE_PER_TICKET,
    -> ROW_NUMBER() OVER (PARTITION BY CLASS_ID ORDER BY PRICE_PER_TICKET DESC) AS ROW_NUM
    -> FROM TICKET_DETAILS) AS RANKED_PRICES
    -> WHERE ROW_NUM = 1;
+--------------+------------------+
| CLASS_ID     | MAX_TICKET_PRICE |
+--------------+------------------+
| Business     |              510 |
| Economy      |              190 |
| Economy Plus |              295 |
| First Class  |              395 |
+--------------+------------------+
4 rows in set (0.10 sec)
```

Write a query to extract the passengers whose route ID is 4 by improving the speed and performance of the passengers_on_flights table.



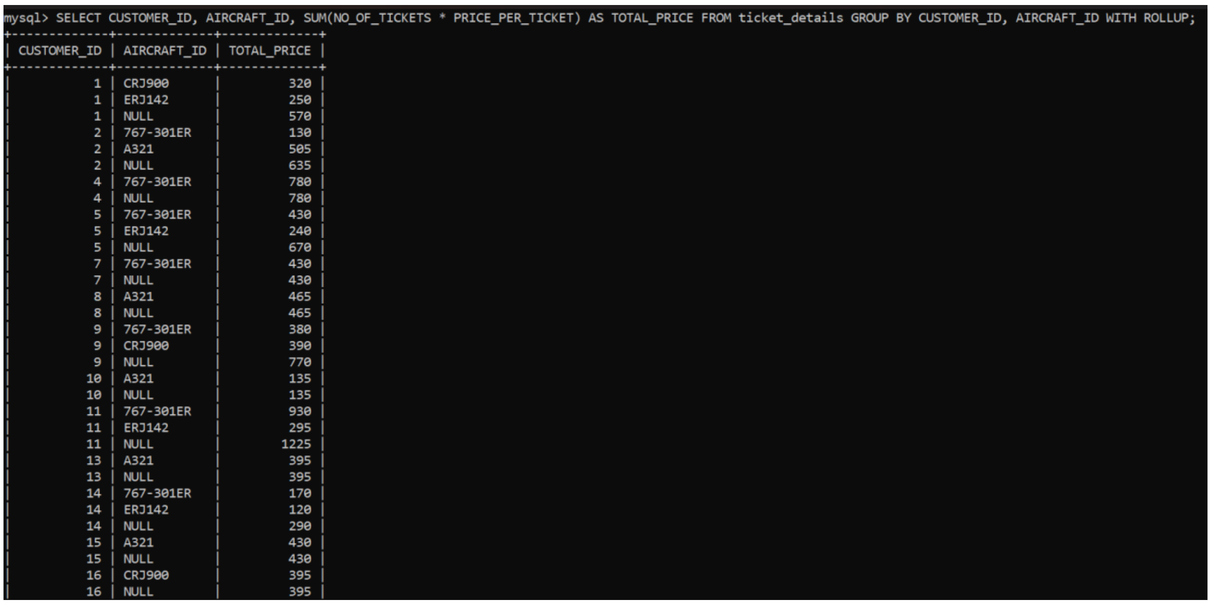
```
mysql> CREATE INDEX idx_route_id ON passengers_on_flights (ROUTE_ID);
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM passengers_on_flights WHERE ROUTE_ID = 4;
+-------------+-------------+----------+--------+---------+----------+-------------+-------------+------------+
| CUSTOMER_ID | AIRCRAFT_ID | ROUTE_ID | DEPART | ARRIVAL | SEAT_NUM | CLASS_ID    | TRAVEL_DATE | FLIGHT_NUM |
+-------------+-------------+----------+--------+---------+----------+-------------+-------------+------------+
|           2 | 767-301ER   |        4 | JFK    | LAX     | 01E      | Economy     | 2018-09-02  |       1114 |
|           4 | 767-301ER   |        4 | JFK    | LAX     | 03FC     | First Class | 2020-04-30  |       1114 |
|          11 | 767-301ER   |        4 | JFK    | LAX     | 05B      | Bussiness   | 2020-11-09  |       1114 |
+-------------+-------------+----------+--------+---------+----------+-------------+-------------+------------+
3 rows in set (0.00 sec)
```

For the route ID 4, write a query to view the execution plan of the passengers_on_flights table.



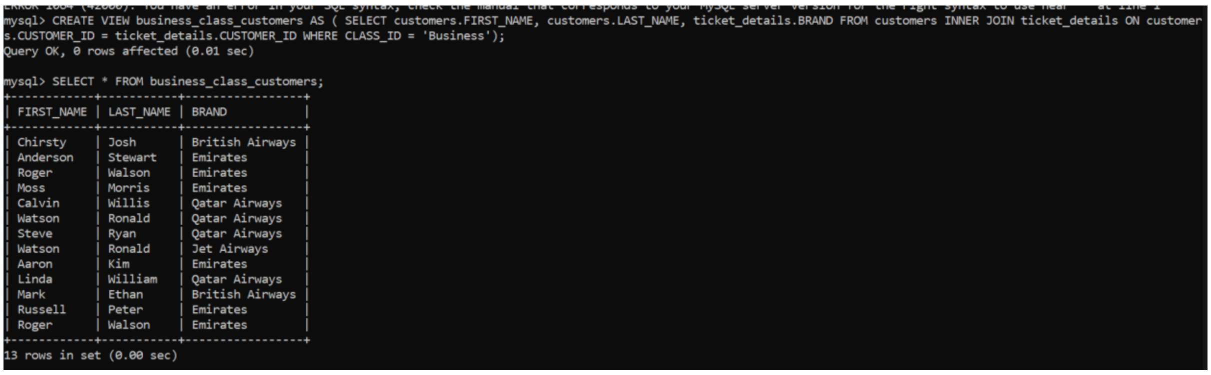
```
mysql> EXPLAIN SELECT * FROM passengers_on_flights WHERE ROUTE_ID = 4;
+----+-------------+-----------------------+------------+------+---------------+--------------+---------+-------+------+----------+-------+
| id | select_type | table                 | partitions | type | possible_keys | key          | key_len | ref   | rows | filtered | Extra |
+----+-------------+-----------------------+------------+------+---------------+--------------+---------+-------+------+----------+-------+
|  1 | SIMPLE      | passengers_on_flights | NULL       | ref  | idx_route_id  | idx_route_id | 5       | const |    3 |   100.00 | NULL  |
+----+-------------+-----------------------+------------+------+---------------+--------------+---------+-------+------+----------+-------+
1 row in set, 1 warning (0.01 sec)
```

Write a query to calculate the total price of all tickets booked by a customer across different aircraft IDs using rollup function.



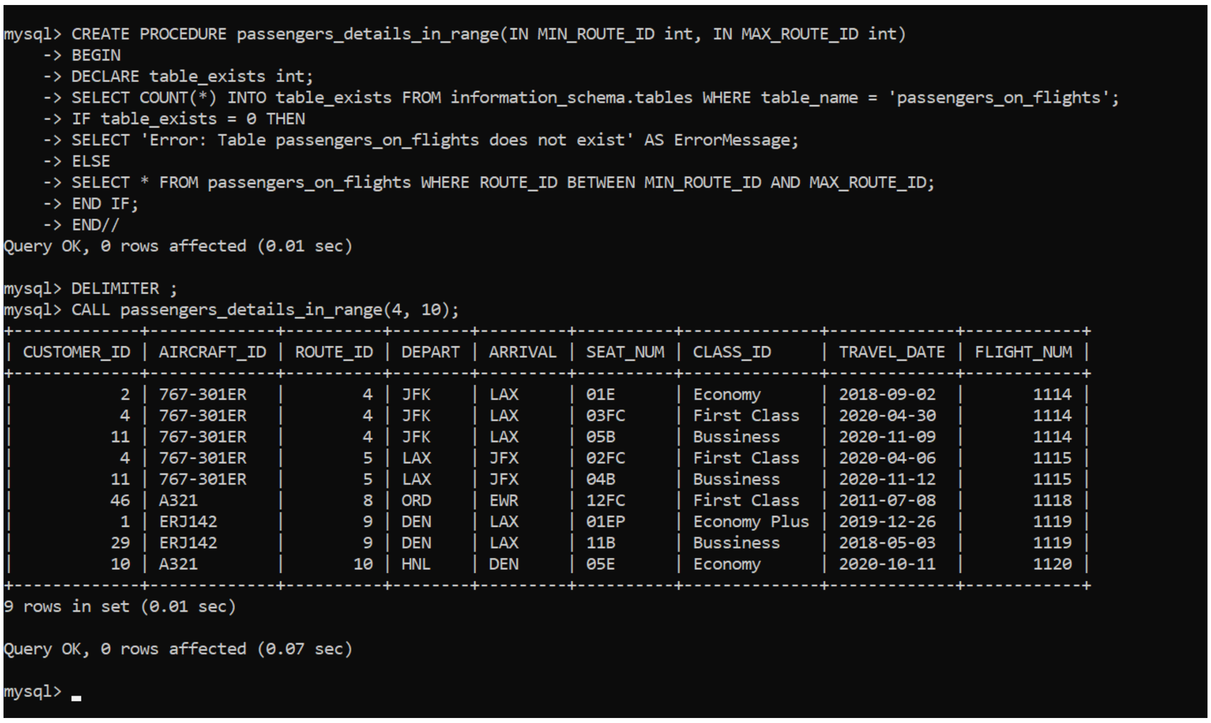
```
mysql> SELECT CUSTOMER_ID, AIRCRAFT_ID, SUM(NO_OF_TICKETS * PRICE_PER_TICKET) AS TOTAL_PRICE FROM ticket_details GROUP BY CUSTOMER_ID, AIRCRAFT_ID WITH ROLLUP;
+-------------+-------------+-------------+
| CUSTOMER_ID | AIRCRAFT_ID | TOTAL_PRICE |
+-------------+-------------+-------------+
|           1 | CRJ900      |         320 |
|           1 | ERJ142      |         250 |
|           1 | NULL        |         570 |
|           2 | 767-301ER   |         130 |
|           2 | A321        |         505 |
|           2 | NULL        |         635 |
|           4 | 767-301ER   |         780 |
|           4 | NULL        |         780 |
|           5 | 767-301ER   |         430 |
|           5 | ERJ142      |         240 |
|           5 | NULL        |         670 |
|           7 | 767-301ER   |         430 |
|           7 | NULL        |         430 |
|           8 | A321        |         465 |
|           8 | NULL        |         465 |
|           9 | 767-301ER   |         380 |
|           9 | CRJ900      |         390 |
|           9 | NULL        |         770 |
|          10 | A321        |         135 |
|          10 | NULL        |         135 |
|          11 | 767-301ER   |         930 |
|          11 | ERJ142      |         295 |
|          11 | NULL        |        1225 |
|          13 | A321        |         395 |
|          13 | NULL        |         395 |
|          14 | 767-301ER   |         170 |
|          14 | ERJ142      |         120 |
|          14 | NULL        |         290 |
|          15 | A321        |         430 |
|          15 | NULL        |         430 |
|          16 | CRJ900      |         395 |
|          16 | NULL        |         395 |
```

Write a query to create a view with only business class customers along with the brand of airlines.



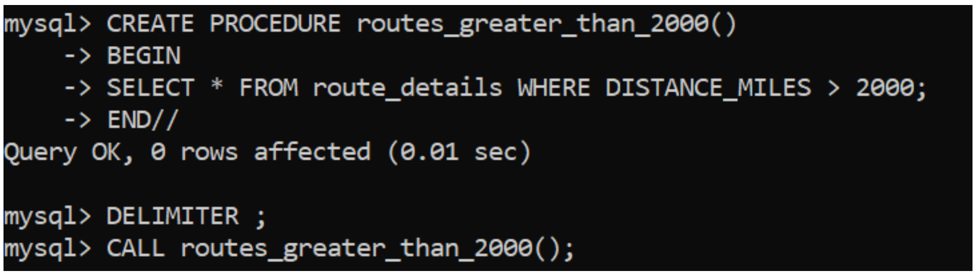
```
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '' at line 1
mysql> CREATE VIEW business_class_customers AS ( SELECT customers.FIRST_NAME, customers.LAST_NAME, ticket_details.BRAND FROM customers INNER JOIN ticket_details ON customer
s.CUSTOMER_ID = ticket_details.CUSTOMER_ID WHERE CLASS_ID = 'Business');
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM business_class_customers;
+------------+-----------+-----------------+
| FIRST_NAME | LAST_NAME | BRAND           |
+------------+-----------+-----------------+
| Chirsty    | Josh      | British Airways |
| Anderson   | Stewart   | Emirates        |
| Roger      | Walson    | Emirates        |
| Moss       | Morris    | Emirates        |
| Calvin     | Willis    | Qatar Airways   |
| Watson     | Ronald    | Qatar Airways   |
| Steve      | Ryan      | Qatar Airways   |
| Watson     | Ronald    | Jet Airways     |
| Aaron      | Kim       | Emirates        |
| Linda      | William   | Qatar Airways   |
| Mark       | Ethan     | British Airways |
| Russell    | Peter     | Emirates        |
| Roger      | Walson    | Emirates        |
+------------+-----------+-----------------+
13 rows in set (0.00 sec)
```

Write a query to create a stored procedure to get the details of all passengers flying between a range of routes defined in run time. Also, return an error message if the table doesn't exist.

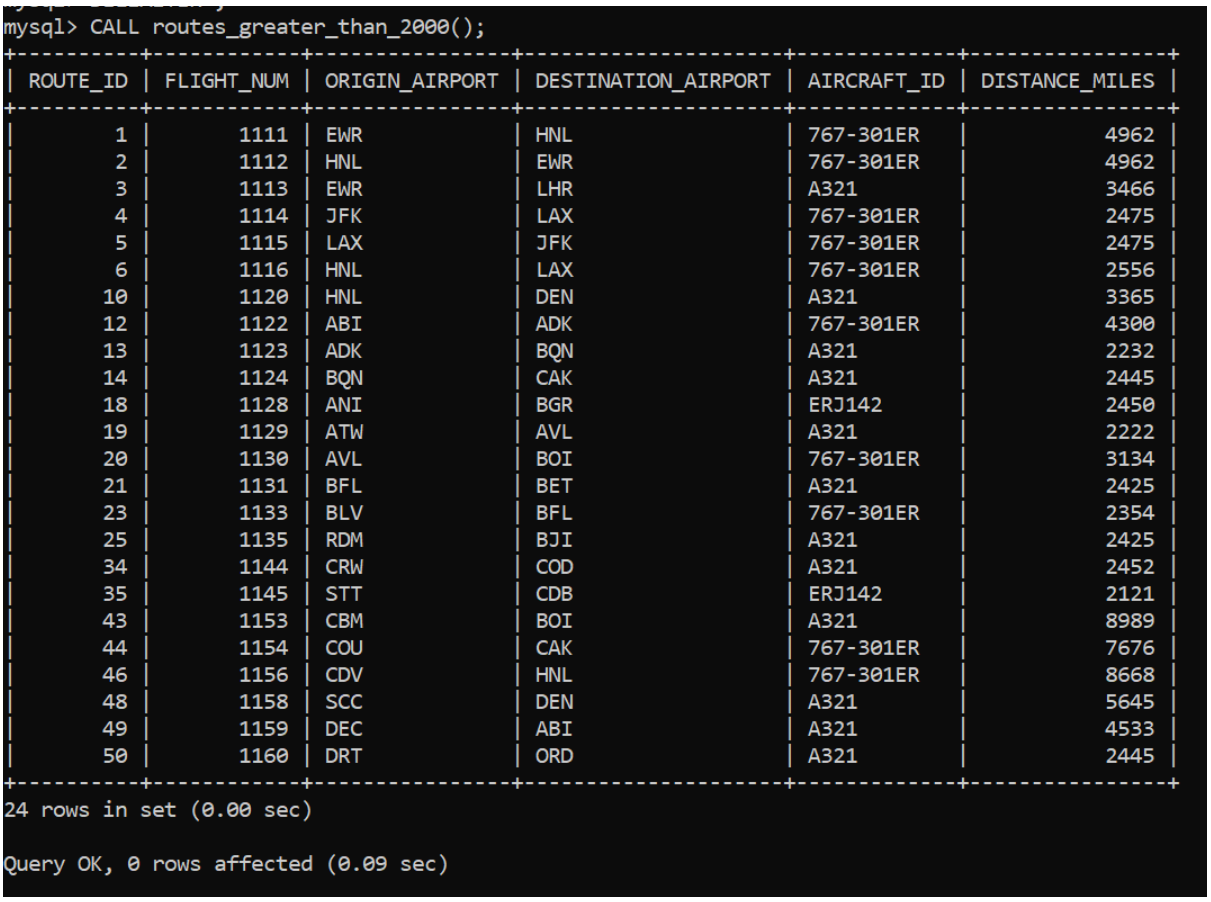


```
mysql> CREATE PROCEDURE passengers_details_in_range(IN MIN_ROUTE_ID int, IN MAX_ROUTE_ID int)
    -> BEGIN
    -> DECLARE table_exists int;
    -> SELECT COUNT(*) INTO table_exists FROM information_schema.tables WHERE table_name = 'passengers_on_flights';
    -> IF table_exists = 0 THEN
    -> SELECT 'Error: Table passengers_on_flights does not exist' AS ErrorMessage;
    -> ELSE
    -> SELECT * FROM passengers_on_flights WHERE ROUTE_ID BETWEEN MIN_ROUTE_ID AND MAX_ROUTE_ID;
    -> END IF;
    -> END//
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> CALL passengers_details_in_range(4, 10);
+-------------+-------------+----------+--------+---------+----------+--------------+-------------+------------+
| CUSTOMER_ID | AIRCRAFT_ID | ROUTE_ID | DEPART | ARRIVAL | SEAT_NUM | CLASS_ID     | TRAVEL_DATE | FLIGHT_NUM |
+-------------+-------------+----------+--------+---------+----------+--------------+-------------+------------+
|           2 | 767-301ER   |        4 | JFK    | LAX     | 01E      | Economy      | 2018-09-02  |       1114 |
|           4 | 767-301ER   |        4 | JFK    | LAX     | 03FC     | First Class  | 2020-04-30  |       1114 |
|          11 | 767-301ER   |        4 | JFK    | LAX     | 05B      | Bussiness    | 2020-11-09  |       1114 |
|           4 | 767-301ER   |        5 | LAX    | JFX     | 02FC     | First Class  | 2020-04-06  |       1115 |
|          11 | 767-301ER   |        5 | LAX    | JFX     | 04B      | Bussiness    | 2020-11-12  |       1115 |
|          46 | A321        |        8 | ORD    | EWR     | 12FC     | First Class  | 2011-07-08  |       1118 |
|           1 | ERJ142      |        9 | DEN    | LAX     | 01EP     | Economy Plus | 2019-12-26  |       1119 |
|          29 | ERJ142      |        9 | DEN    | LAX     | 11B      | Bussiness    | 2018-05-03  |       1119 |
|          10 | A321        |       10 | HNL    | DEN     | 05E      | Economy      | 2020-10-11  |       1120 |
+-------------+-------------+----------+--------+---------+----------+--------------+-------------+------------+
9 rows in set (0.01 sec)

Query OK, 0 rows affected (0.07 sec)

mysql> _
```

Write a query to create a stored procedure that extracts all the details from the routes table where the travelled distance is more than 2000 miles.

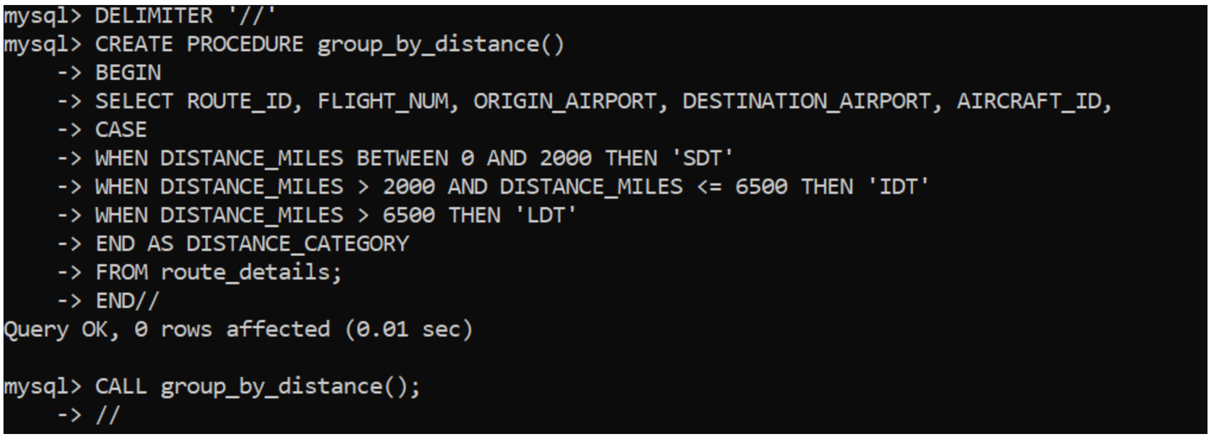


```
mysql> CREATE PROCEDURE routes_greater_than_2000()
    -> BEGIN
    -> SELECT * FROM route_details WHERE DISTANCE_MILES > 2000;
    -> END//
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> CALL routes_greater_than_2000();
```

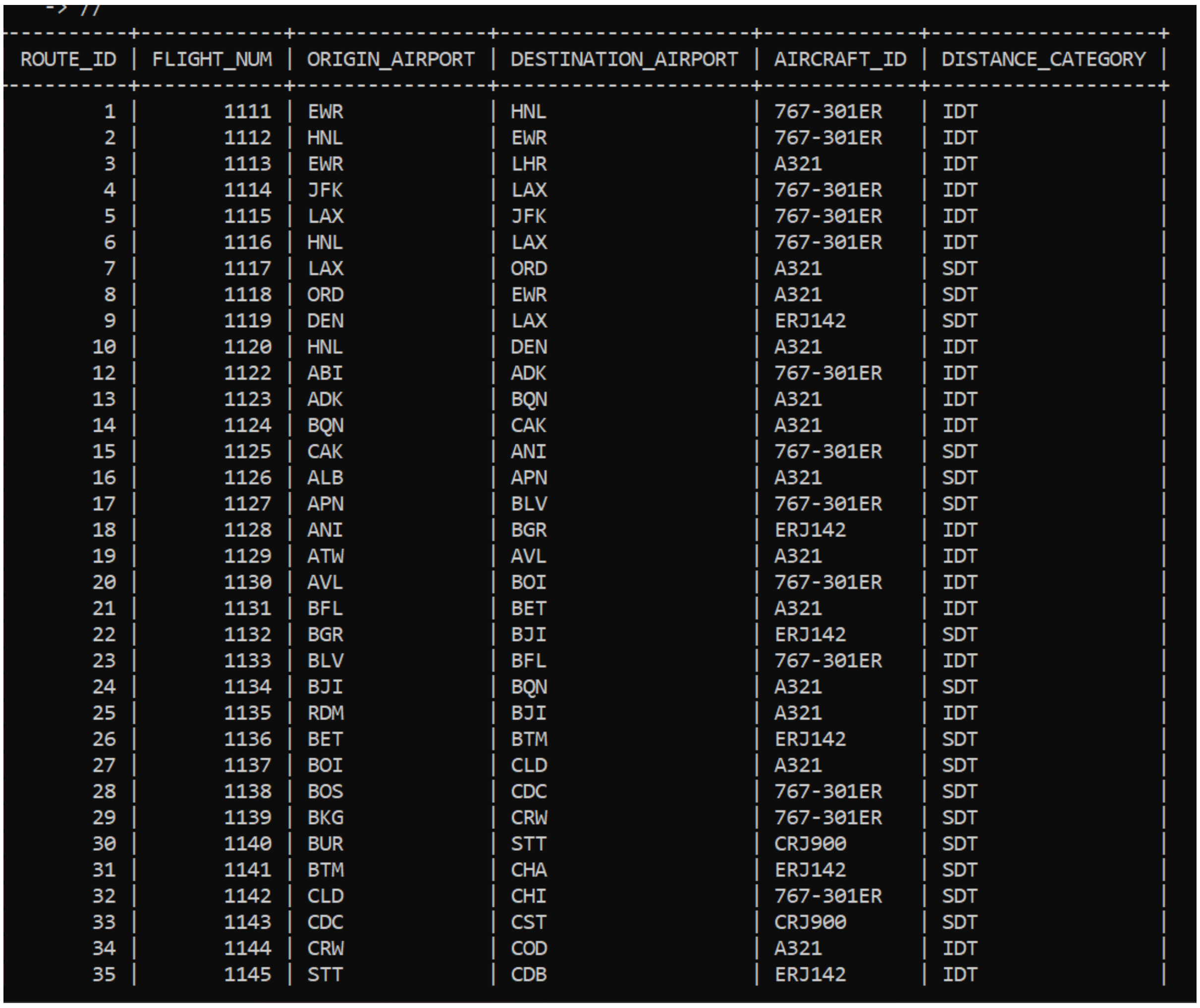


```
mysql> CALL routes_greater_than_2000();
+----------+------------+----------------+---------------------+--------------+----------------+
| ROUTE_ID | FLIGHT_NUM | ORIGIN_AIRPORT | DESTINATION_AIRPORT | AIRCRAFT_ID  | DISTANCE_MILES |
+----------+------------+----------------+---------------------+--------------+----------------+
|        1 |       1111 | EWR            | HNL                 | 767-301ER    |           4962 |
|        2 |       1112 | HNL            | EWR                 | 767-301ER    |           4962 |
|        3 |       1113 | EWR            | LHR                 | A321         |           3466 |
|        4 |       1114 | JFK            | LAX                 | 767-301ER    |           2475 |
|        5 |       1115 | LAX            | JFK                 | 767-301ER    |           2475 |
|        6 |       1116 | HNL            | LAX                 | 767-301ER    |           2556 |
|       10 |       1120 | HNL            | DEN                 | A321         |           3365 |
|       12 |       1122 | ABI            | ADK                 | 767-301ER    |           4300 |
|       13 |       1123 | ADK            | BQN                 | A321         |           2232 |
|       14 |       1124 | BQN            | CAK                 | A321         |           2445 |
|       18 |       1128 | ANI            | BGR                 | ERJ142       |           2450 |
|       19 |       1129 | ATW            | AVL                 | A321         |           2222 |
|       20 |       1130 | AVL            | BOI                 | 767-301ER    |           3134 |
|       21 |       1131 | BFL            | BET                 | A321         |           2425 |
|       23 |       1133 | BLV            | BFL                 | 767-301ER    |           2354 |
|       25 |       1135 | RDM            | BJI                 | A321         |           2425 |
|       34 |       1144 | CRW            | COD                 | A321         |           2452 |
|       35 |       1145 | STT            | CDB                 | ERJ142       |           2121 |
|       43 |       1153 | CBM            | BOI                 | A321         |           8989 |
|       44 |       1154 | COU            | CAK                 | 767-301ER    |           7676 |
|       46 |       1156 | CDV            | HNL                 | 767-301ER    |           8668 |
|       48 |       1158 | SCC            | DEN                 | A321         |           5645 |
|       49 |       1159 | DEC            | ABI                 | A321         |           4533 |
|       50 |       1160 | DRT            | ORD                 | A321         |           2445 |
+----------+------------+----------------+---------------------+--------------+----------------+
24 rows in set (0.00 sec)

Query OK, 0 rows affected (0.09 sec)
```

Write a query to create a stored procedure that groups the distance travelled by each flight into three categories. The categories are, short distance travel (SDT) for >=0 AND <= 2000 miles, intermediate distance travel (IDT) for >2000 AND <=6500, and long-distance travel (LDT) for >6500.



```
mysql> DELIMITER '//'
mysql> CREATE PROCEDURE group_by_distance()
    -> BEGIN
    -> SELECT ROUTE_ID, FLIGHT_NUM, ORIGIN_AIRPORT, DESTINATION_AIRPORT, AIRCRAFT_ID,
    -> CASE
    -> WHEN DISTANCE_MILES BETWEEN 0 AND 2000 THEN 'SDT'
    -> WHEN DISTANCE_MILES > 2000 AND DISTANCE_MILES <= 6500 THEN 'IDT'
    -> WHEN DISTANCE_MILES > 6500 THEN 'LDT'
    -> END AS DISTANCE_CATEGORY
    -> FROM route_details;
    -> END//
Query OK, 0 rows affected (0.01 sec)

mysql> CALL group_by_distance();
    -> //
```

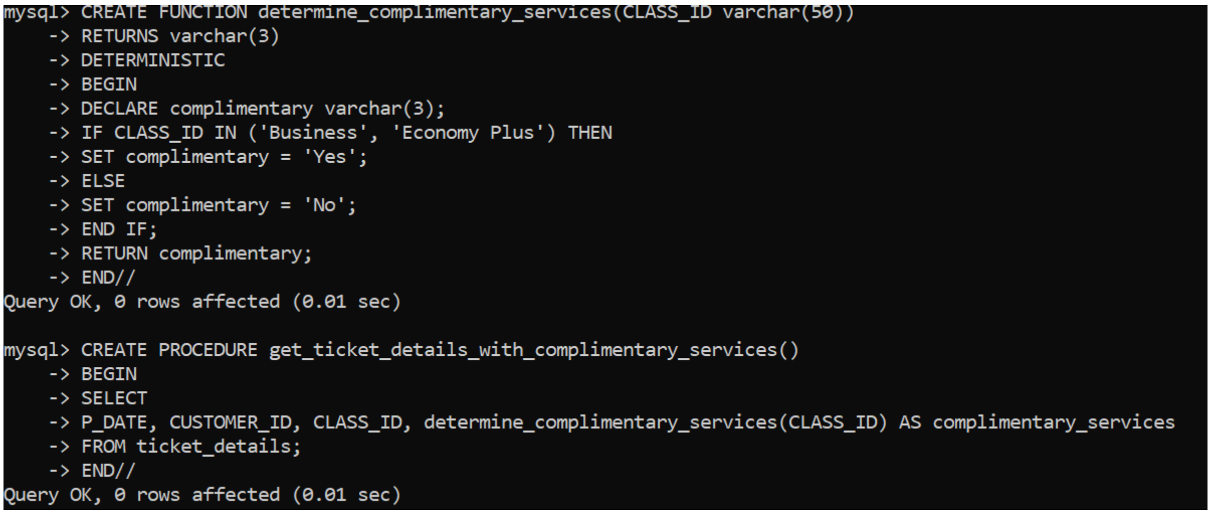


```
    -> //
+----------+------------+----------------+---------------------+--------------+-------------------+
| ROUTE_ID | FLIGHT_NUM | ORIGIN_AIRPORT | DESTINATION_AIRPORT | AIRCRAFT_ID  | DISTANCE_CATEGORY |
+----------+------------+----------------+---------------------+--------------+-------------------+
|        1 |       1111 | EWR            | HNL                 | 767-301ER    | IDT               |
|        2 |       1112 | HNL            | EWR                 | 767-301ER    | IDT               |
|        3 |       1113 | EWR            | LHR                 | A321         | IDT               |
|        4 |       1114 | JFK            | LAX                 | 767-301ER    | IDT               |
|        5 |       1115 | LAX            | JFK                 | 767-301ER    | IDT               |
|        6 |       1116 | HNL            | LAX                 | 767-301ER    | IDT               |
|        7 |       1117 | LAX            | ORD                 | A321         | SDT               |
|        8 |       1118 | ORD            | EWR                 | A321         | SDT               |
|        9 |       1119 | DEN            | LAX                 | ERJ142       | SDT               |
|       10 |       1120 | HNL            | DEN                 | A321         | IDT               |
|       12 |       1122 | ABI            | ADK                 | 767-301ER    | IDT               |
|       13 |       1123 | ADK            | BQN                 | A321         | IDT               |
|       14 |       1124 | BQN            | CAK                 | A321         | IDT               |
|       15 |       1125 | CAK            | ANI                 | 767-301ER    | SDT               |
|       16 |       1126 | ALB            | APN                 | A321         | SDT               |
|       17 |       1127 | APN            | BLV                 | 767-301ER    | SDT               |
|       18 |       1128 | ANI            | BGR                 | ERJ142       | IDT               |
|       19 |       1129 | ATW            | AVL                 | A321         | IDT               |
|       20 |       1130 | AVL            | BOI                 | 767-301ER    | IDT               |
|       21 |       1131 | BFL            | BET                 | A321         | IDT               |
|       22 |       1132 | BGR            | BJI                 | ERJ142       | SDT               |
|       23 |       1133 | BLV            | BFL                 | 767-301ER    | IDT               |
|       24 |       1134 | BJI            | BQN                 | A321         | SDT               |
|       25 |       1135 | RDM            | BJI                 | A321         | IDT               |
|       26 |       1136 | BET            | BTM                 | ERJ142       | SDT               |
|       27 |       1137 | BOI            | CLD                 | A321         | SDT               |
|       28 |       1138 | BOS            | CDC                 | 767-301ER    | SDT               |
|       29 |       1139 | BKG            | CRW                 | 767-301ER    | SDT               |
|       30 |       1140 | BUR            | STT                 | CRJ900       | SDT               |
|       31 |       1141 | BTM            | CHA                 | ERJ142       | SDT               |
|       32 |       1142 | CLD            | CHI                 | 767-301ER    | SDT               |
|       33 |       1143 | CDC            | CST                 | CRJ900       | SDT               |
|       34 |       1144 | CRW            | COD                 | A321         | IDT               |
|       35 |       1145 | STT            | CDB                 | ERJ142       | IDT               |
```

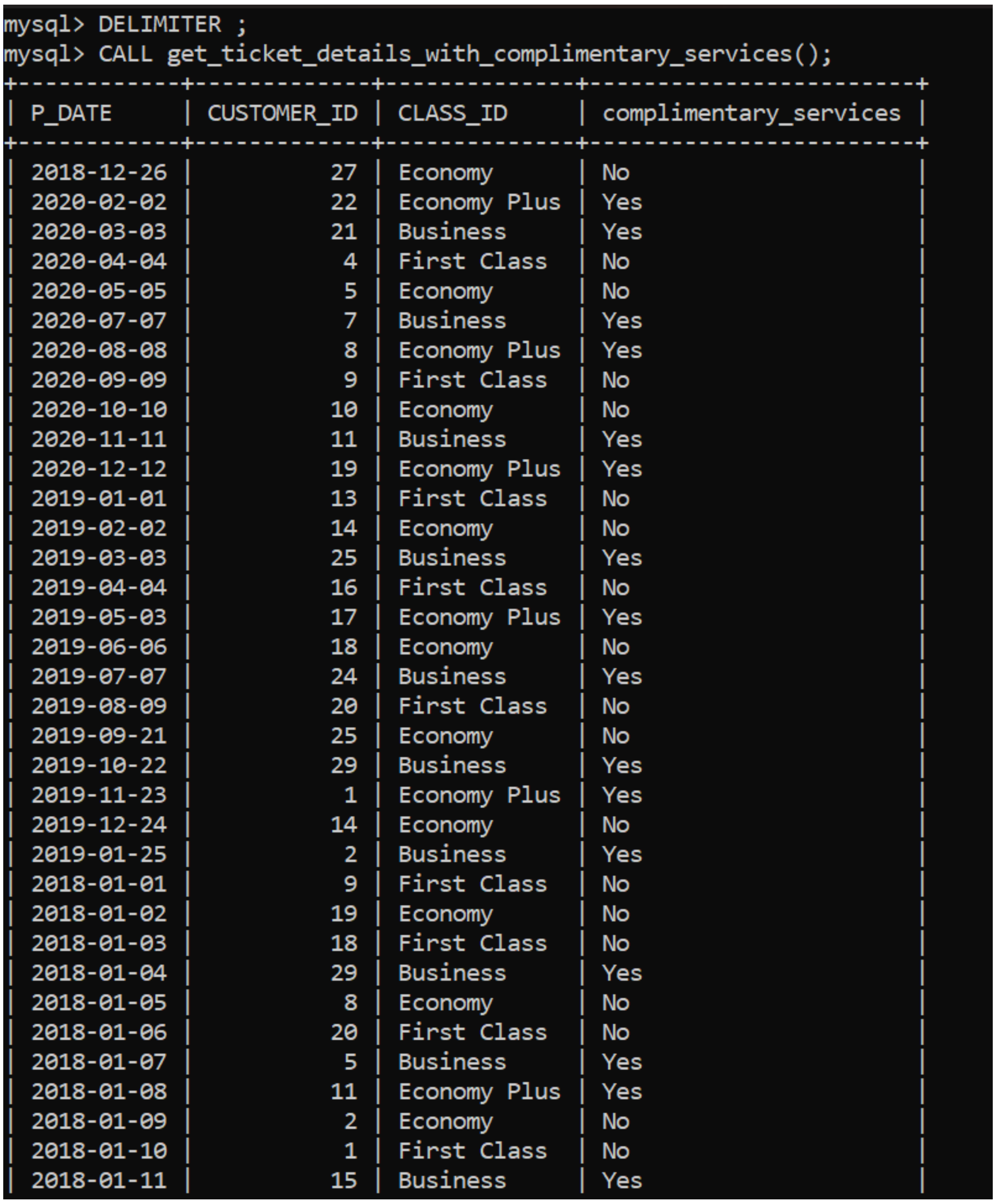
Write a query to extract ticket purchase date, customer ID, class ID and specify if the complimentary services are provided for the specific class using a stored function in stored procedure on the ticket_details table.

Condition:

- **If the class is *Business* and *Economy Plus,* then complimentary services are given as *Yes,* else it is *No***

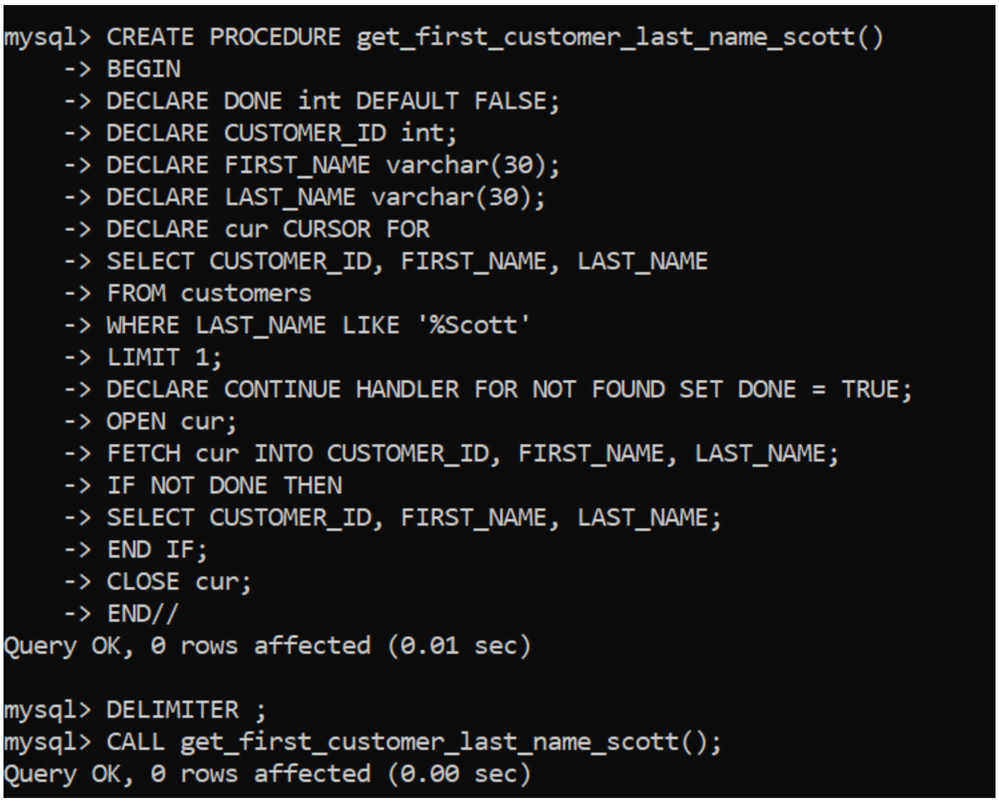


```
mysql> CREATE FUNCTION determine_complimentary_services(CLASS_ID varchar(50))
    -> RETURNS varchar(3)
    -> DETERMINISTIC
    -> BEGIN
    -> DECLARE complimentary varchar(3);
    -> IF CLASS_ID IN ('Business', 'Economy Plus') THEN
    -> SET complimentary = 'Yes';
    -> ELSE
    -> SET complimentary = 'No';
    -> END IF;
    -> RETURN complimentary;
    -> END//
Query OK, 0 rows affected (0.01 sec)

mysql> CREATE PROCEDURE get_ticket_details_with_complimentary_services()
    -> BEGIN
    -> SELECT
    -> P_DATE, CUSTOMER_ID, CLASS_ID, determine_complimentary_services(CLASS_ID) AS complimentary_services
    -> FROM ticket_details;
    -> END//
Query OK, 0 rows affected (0.01 sec)
```

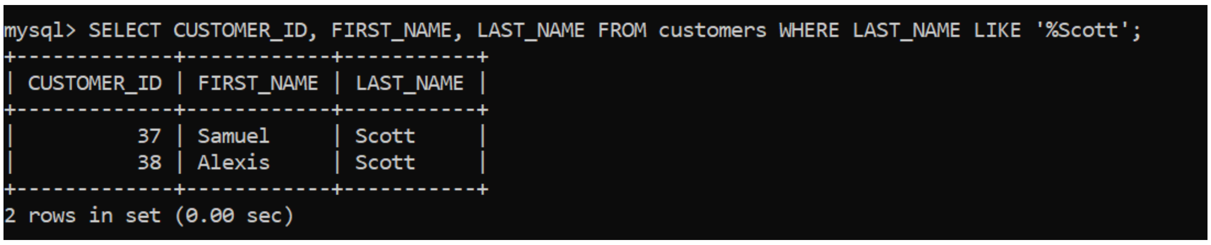


```
mysql> DELIMITER ;
mysql> CALL get_ticket_details_with_complimentary_services();
+------------+-------------+--------------+------------------------+
| P_DATE     | CUSTOMER_ID | CLASS_ID     | complimentary_services |
+------------+-------------+--------------+------------------------+
| 2018-12-26 |          27 | Economy      | No                     |
| 2020-02-02 |          22 | Economy Plus | Yes                    |
| 2020-03-03 |          21 | Business     | Yes                    |
| 2020-04-04 |           4 | First Class  | No                     |
| 2020-05-05 |           5 | Economy      | No                     |
| 2020-07-07 |           7 | Business     | Yes                    |
| 2020-08-08 |           8 | Economy Plus | Yes                    |
| 2020-09-09 |           9 | First Class  | No                     |
| 2020-10-10 |          10 | Economy      | No                     |
| 2020-11-11 |          11 | Business     | Yes                    |
| 2020-12-12 |          19 | Economy Plus | Yes                    |
| 2019-01-01 |          13 | First Class  | No                     |
| 2019-02-02 |          14 | Economy      | No                     |
| 2019-03-03 |          25 | Business     | Yes                    |
| 2019-04-04 |          16 | First Class  | No                     |
| 2019-05-03 |          17 | Economy Plus | Yes                    |
| 2019-06-06 |          18 | Economy      | No                     |
| 2019-07-07 |          24 | Business     | Yes                    |
| 2019-08-09 |          20 | First Class  | No                     |
| 2019-09-21 |          25 | Economy      | No                     |
| 2019-10-22 |          29 | Business     | Yes                    |
| 2019-11-23 |           1 | Economy Plus | Yes                    |
| 2019-12-24 |          14 | Economy      | No                     |
| 2019-01-25 |           2 | Business     | Yes                    |
| 2018-01-01 |           9 | First Class  | No                     |
| 2018-01-02 |          19 | Economy      | No                     |
| 2018-01-03 |          18 | First Class  | No                     |
| 2018-01-04 |          29 | Business     | Yes                    |
| 2018-01-05 |           8 | Economy      | No                     |
| 2018-01-06 |          20 | First Class  | No                     |
| 2018-01-07 |           5 | Business     | Yes                    |
| 2018-01-08 |          11 | Economy Plus | Yes                    |
| 2018-01-09 |           2 | Economy      | No                     |
| 2018-01-10 |           1 | First Class  | No                     |
| 2018-01-11 |          15 | Business     | Yes                    |
```

Write a query to extract the first record of the customer whose last name ends with Scott using a cursor from the customer table.



```
mysql> CREATE PROCEDURE get_first_customer_last_name_scott()
    -> BEGIN
    -> DECLARE DONE int DEFAULT FALSE;
    -> DECLARE CUSTOMER_ID int;
    -> DECLARE FIRST_NAME varchar(30);
    -> DECLARE LAST_NAME varchar(30);
    -> DECLARE cur CURSOR FOR
    -> SELECT CUSTOMER_ID, FIRST_NAME, LAST_NAME
    -> FROM customers
    -> WHERE LAST_NAME LIKE '%Scott'
    -> LIMIT 1;
    -> DECLARE CONTINUE HANDLER FOR NOT FOUND SET DONE = TRUE;
    -> OPEN cur;
    -> FETCH cur INTO CUSTOMER_ID, FIRST_NAME, LAST_NAME;
    -> IF NOT DONE THEN
    -> SELECT CUSTOMER_ID, FIRST_NAME, LAST_NAME;
    -> END IF;
    -> CLOSE cur;
    -> END//
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> CALL get_first_customer_last_name_scott();
Query OK, 0 rows affected (0.00 sec)
```



```
mysql> SELECT CUSTOMER_ID, FIRST_NAME, LAST_NAME FROM customers WHERE LAST_NAME LIKE '%Scott';
+-------------+------------+-----------+
| CUSTOMER_ID | FIRST_NAME | LAST_NAME |
+-------------+------------+-----------+
|          37 | Samuel     | Scott     |
|          38 | Alexis     | Scott     |
+-------------+------------+-----------+
2 rows in set (0.00 sec)
```