

Question 1:

List all passengers and the flights they have booked.

Screenshot 1:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'airlines_db' selected. The main query editor contains the following SQL code:

```
100 • SELECT * FROM bookings;
101
102 • Select P.PassengerID,P.FirstName, P.LastName,F.FlightID,F.FlightNumber, F.Origin, F.Destination
103 FROM Passengers P
104 JOIN
105     BOOKINGS B ON B.PassengerID=P.PassengerID
106 JOIN
107     Flights F ON B.FlightID = F.FlightID;
108
```

The 'Result Grid' shows 10 rows of data:

PassengerID	FirstName	LastName	FlightID	FlightNumber	Origin	Destination
101	John	Smith	1	AA123	New York	Los Angeles
102	Emily	Johnson	2	UA456	Chicago	Miami
103	Michael	Davis	3	DL789	Los Angeles	Seattle
101	John	Smith	2	UA456	Chicago	Miami
104	Olivia	Brown	4	BA789	London	New York
105	Daniel	Johnson	5	LH456	Berlin	Tokyo
106	Sophia	Lee	6	AF123	Paris	Dubai
107	Matthew	Wilson	7	SQ789	Singapore	Sydney
108	Ava	Jones	8	EX456	Dubai	New York
109	Ethan	Miller	9	AA456	New York	London

The bottom panel shows the 'Output' tab with a log of actions and messages. The first message is an error: 'Error Code: 1054. Unknown column 'P.PassengerID' in field list'. The second message is a success: '10 row(s) returned'.

Query 1:

```
SELECT P.PassengerID,P.FirstName, P.LastName,F.FlightID,F.FlightNumber, F.Origin, F.Destination
```

```
FROM Passengers P
```

```
JOIN
```

```
BOOKINGS B ON B.PassengerID=P.PassengerID
```

```
JOIN
```

```
Flights F ON B.FlightID = F.FlightID;
```

Question 2:

Display the flights with no bookings made.

Screenshot 2:

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left lists databases including 'airlines_db'. The 'Query Editor' contains the following SQL query:

```
103 FROM Passengers P
104 JOIN
105     BOOKINGS B ON B.PassengerID=P.PassengerID
106 JOIN
107     Flights F ON B.FlightID = F.FlightID;
108
109
110 • SELECT * FROM Flights F LEFT JOIN Bookings B ON F.FlightID=B.FlightID WHERE B.BookingID IS NULL;
111
```

The 'Result Grid' shows the results of the query. The first row is highlighted:

FlightID	FlightNumber	Origin	Destination	DepartureTime	ArrivalTime	BookingID	PassengerID	FlightID	SeatNumber	BookingDate
10	DL876	Seattle	Miami	2024-03-10 16:00:00	2024-03-10 19:30:00	NULL	NULL	NULL	NULL	NULL

The 'Output' pane at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
174	17:07:56	Select P.PassengerID, P.FirstName, P.LastName, F.FlightID, F.FlightNumber, F.Origin, F.Destination FROM Pas...	10 row(s) returned	0.000 sec / 0.000 sec
175	17:14:21	Select * FROM Rights F LEFT JOIN Bookings B ON F.FlightID=B.FlightID LIMIT 0, 1000	11 row(s) returned	0.016 sec / 0.000 sec
176	17:14:55	Select * FROM Rights F LEFT JOIN Bookings B ON F.FlightID=B.FlightID WHERE B.BookingID IS NULL LI...	1 row(s) returned	0.000 sec / 0.000 sec

Query 2:

SELECT * FROM Flights F

LEFT JOIN

Bookings B ON F.FlightID=B.FlightID

WHERE

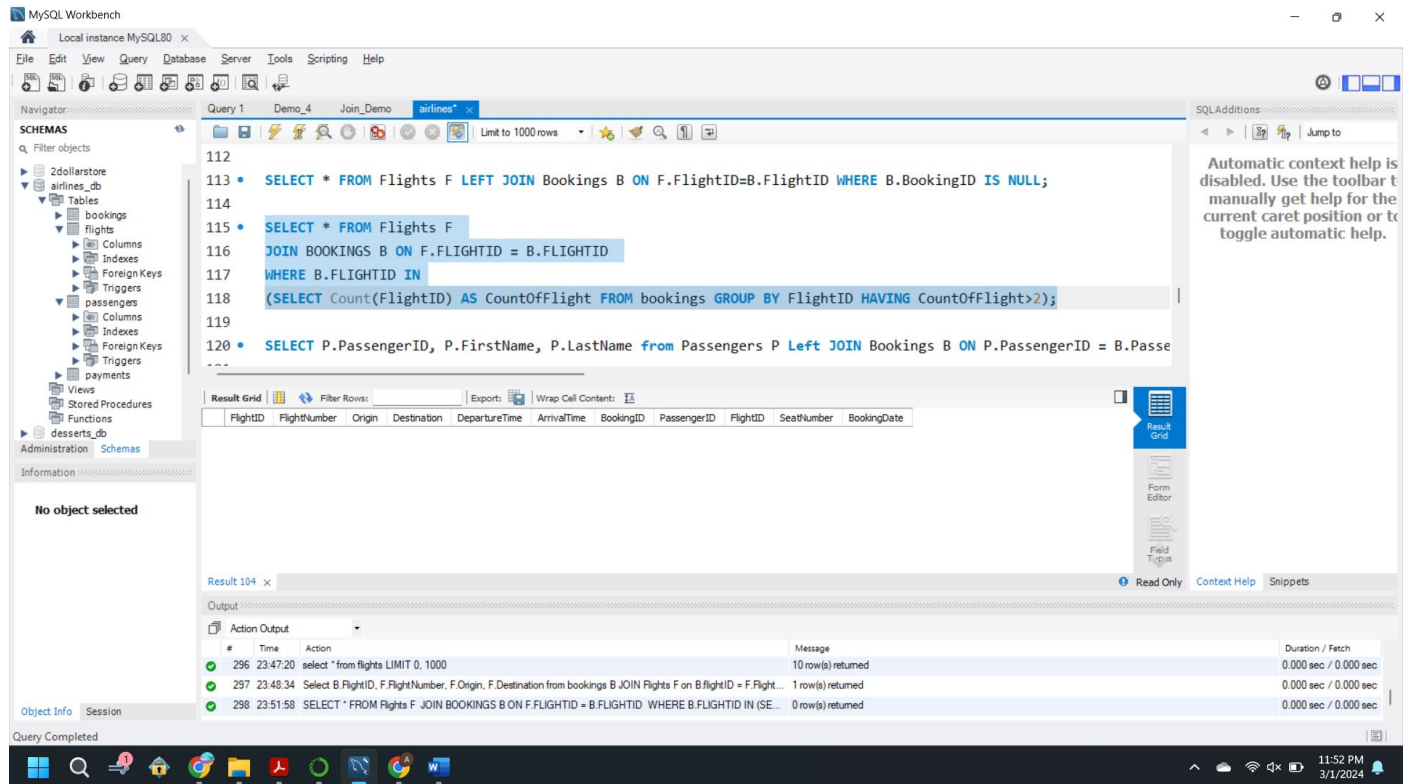
B.BookingID IS NULL;

Question 3:

Retrieve the flights with more than two bookings.

Screenshot 3:

No bookings were made more than 2 times for a flight.



Query 3:

SELECT * FROM Flights F

JOIN

BOOKINGS B ON F.FLIGHTID = B.FLIGHTID

WHERE

B.FLIGHTID IN

(SELECT Count(FlightID) AS CountOfFlight FROM bookings GROUP BY FlightID HAVING CountOfFlight>2);

Question 3 – Instead of more than 2 flights, we can find more than 1 flight.

Screenshot 3:

The screenshot shows the MySQL Workbench interface. The 'Query Editor' window contains the following SQL query:

```
Flights F ON B.FlightID = F.FlightID;  
  
110 • SELECT * FROM Flights F LEFT JOIN Bookings B ON F.FlightID=B.FlightID WHERE B.BookingID IS NULL;  
111  
112 • SELECT * FROM Flights F  
113 JOIN BOOKINGS B ON F.FLIGHTID = B.FLIGHTID  
114 WHERE B.FLIGHTID IN  
115 (SELECT Count(FlightID) AS CountOfFlight FROM bookings GROUP BY FlightID HAVING CountOfFlight>1);
```

The 'Result Grid' shows the following data:

FlightID	FlightNumber	Origin	Destination	DepartureTime	ArrivalTime	BookingID	PassengerID	FlightID	SeatNumber	BookingDate
2	UA456	Chicago	Miami	2024-03-02 10:30:00	2024-03-02 13:30:00	502	102	2	5C	2024-02-16 11:45:00
2	UA456	Chicago	Miami	2024-03-02 10:30:00	2024-03-02 13:30:00	504	101	2	7F	2024-02-18 16:30:00

The 'Output' window shows the following action output:

#	Time	Action	Message	Duration / Fetch
197	17:19:10	Select Count(FlightID) from bookings GROUP BY FlightID LIMIT 0, 1000	9 row(s) returned	0.000 sec / 0.000 sec
198	17:19:55	Select Count(FlightID) AS CountOfFlight from bookings GROUP BY FlightID HAVING CountOfFlight>1 LIMIT 0, ...	1 row(s) returned	0.000 sec / 0.000 sec
199	17:21:47	Select * FROM Flights F JOIN BOOKINGS B ON F.FLIGHTID = B.FLIGHTID WHERE B.FLIGHTID IN(Select ...	2 row(s) returned	0.000 sec / 0.000 sec

Query 3:

SELECT * FROM Flights F

JOIN

BOOKINGS B ON F.FLIGHTID = B.FLIGHTID

WHERE

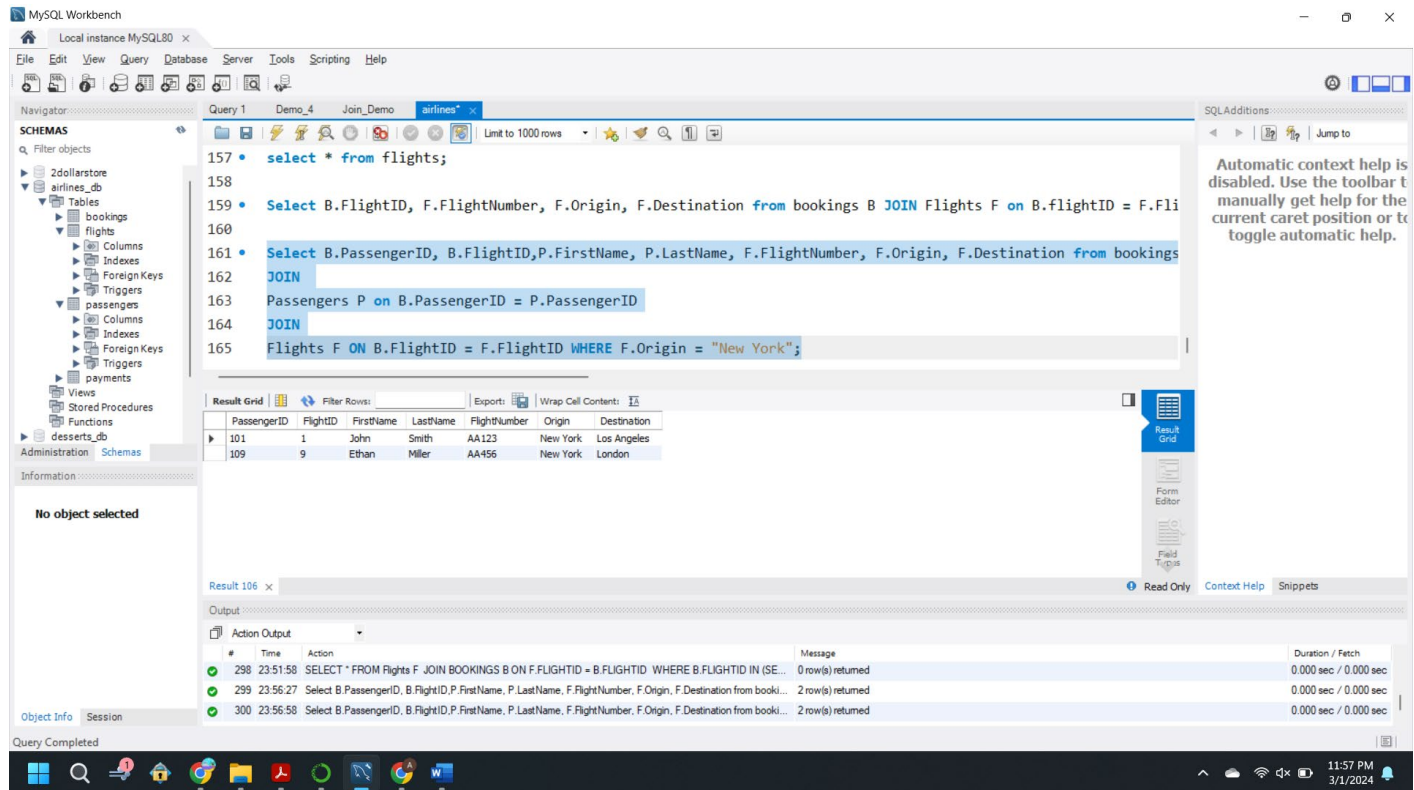
B.FLIGHTID IN

(SELECT Count(FlightID) AS CountOfFlight FROM bookings GROUP BY FlightID HAVING CountOfFlight>1);

Question 4:

List all passengers who booked a flight departing from New York.

Screenshot 4:



Query 4:

```
SELECT P.PassengerID,P.FirstName, P.LastName,F.FlightID,F.FlightNumber, F.Origin, F.Destination
FROM Passengers P
JOIN
BOOKINGS B ON B.PassengerID=P.PassengerID
JOIN
Flights F ON B.FlightID = F.FlightID
WHERE
F.Origin="New York";
```

Question 5:

Display the passengers who have not made any bookings.

Screenshot 5:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'airlines_db' selected. The main editor window shows a SQL query in the 'Query 1' tab. The query is as follows:

```
162 JOIN
163 Passengers P ON B.PassengerID = P.PassengerID
164 JOIN
165 Flights F ON B.FlightID = F.FlightID WHERE F.Origin = "New York";
166
167 • Select P.PassengerID, P.FirstName, P.LastName From Passengers P
168 Left join
169 Bookings B ON P.PassengerID = B.PassengerID
170 WHERE B.BookingID IS NULL;
```

The 'Result Grid' shows one row of data:

PassengerID	FirstName	LastName
110	Emma	Anderson

The 'Output' tab at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
302	00:01:37	SELECT P.PassengerID, P.FirstName, P.LastName from Passengers P Left JOIN Bookings B ON P.Passenger...	1 row(s) returned	0.000 sec / 0.000 sec
303	00:01:49	SELECT P.PassengerID, P.FirstName, P.LastName from Passengers P Left JOIN Bookings B ON P.Passenger...	11 row(s) returned	0.000 sec / 0.000 sec
304	00:02:32	Select P.PassengerID, P.FirstName, P.LastName From Passengers P Left join Bookings B ON P.PassengerID...	1 row(s) returned	0.000 sec / 0.000 sec

Query 5:

SELECT P.PassengerID, P.FirstName, P.LastName FROM Passengers P

LEFT JOIN

Bookings B ON P.PassengerID = B.PassengerID

WHERE

B.BookingID IS NULL;

Question 6:

Retrieve the average number of bookings per passenger

Screenshot 6:

The screenshot shows the MySQL Workbench interface. The main editor displays a SQL query with line numbers 108 to 120. The query is as follows:

```
108 WHERE F.Destination="New York";
109
110
111 • SELECT * FROM Flights F LEFT JOIN Bookings B ON F.FlightID=B.FlightID WHERE B.BookingID IS NULL;
112
113 • SELECT * FROM Flights F
114 JOIN BOOKINGS B ON F.FLIGHTID = B.FLIGHTID
115 WHERE B.FLIGHTID IN
116 (SELECT Count(FlightID) AS CountOfFlight FROM bookings GROUP BY FlightID HAVING CountOfFlight>1);
117
118 • SELECT P.PassengerID, P.FirstName, P.LastName from Passengers P Left JOIN Bookings B ON P.PassengerID = B.PassengerID WHERE B.PassengerID
119
120 • Select Avg(Count) from (Select COUNT(PassengerID) AS Count from Bookings GROUP BY PassengerID) as Avg;
```

Below the query editor, the 'Result Grid' shows the execution results. The first result set, 'Avg(Count)', contains one row with the value 1.1111. The second result set, 'Output', shows the execution log with the following entries:

#	Time	Action	Message	Duration / Fetch
223	20:13:21	Select Avg(Count) from (Select COUNT(PassengerID) AS Count from Bookings GROUP BY PassengerID) as p LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
224	10:44:24	Select Avg(Count) from (Select COUNT(PassengerID) AS Count from Bookings GROUP BY PassengerID) LIMIT 0, 1000	Error Code: 1248. Every derived table must have its own alias	0.000 sec
225	10:44:38	Select Avg(Count) from (Select COUNT(PassengerID) AS Count from Bookings GROUP BY PassengerID) as Avg LIMIT 0, 1000	1 row(s) returned	0.015 sec / 0.000 sec

Query 6:

SELECT Avg(Count) FROM (SELECT COUNT(PassengerID) AS Count FROM Bookings GROUP BY PassengerID) as Avg;

Question 7:

List the flights where the total payment amount exceeds the average payment amount.

Screenshot 7:

The screenshot shows the MySQL Workbench interface. The SQL editor contains a query that filters flights based on whether their total payment amount exceeds the average payment amount. The query is as follows:

```
112
113 • SELECT * FROM Flights F LEFT JOIN Bookings B ON F.FlightID=B.FlightID WHERE B.BookingID IS NULL;
114
115 • SELECT * FROM Flights F
116 JOIN BOOKINGS B ON F.FLIGHTID = B.FLIGHTID
117 WHERE B.FLIGHTID IN
118 (SELECT Count(FlightID) AS CountOfFlight FROM bookings GROUP BY FlightID HAVING CountOfFlight>1);
119
120 • SELECT P.PassengerID, P.FirstName, P.LastName from Passengers P Left JOIN Bookings B ON P.PassengerID = B.PassengerID WHERE B.PassengerID
121
122 • Select Avg(Count) from (Select COUNT(PassengerID) AS Count from Bookings GROUP BY PassengerID) as Avg;
123
124 • Select p.BookingID,p.amount, b.flightID, f.FlightNumber, f.Origin, F.destination from Payments p JOIN Bookings b ON p.bookingID = b.bookingID
```

The Results window shows the output of the query, displaying a table with columns: BookingID, amount, flightID, FlightNumber, Origin, and destination. The data rows are:

BookingID	amount	flightID	FlightNumber	Origin	destination
502	450.50	2	UA456	Chicago	Miami
505	350.75	4	BA789	London	New York
506	500.25	5	LH156	Berlin	Tokyo
509	400.00	8	BK456	Dubai	New York

The bottom status bar shows the execution time and the number of rows returned for each statement.

Query 7:

```
SELECT p.BookingID,p.amount, b.flightID, f.FlightNumber, f.Origin, F.destination FROM Payments p
JOIN
Bookings b ON p.bookingID = b.bookingID
JOIN
Flights f ON b.flightID = f.FlightID
WHERE p.amount>(SELECT AVG(Amount) FROM payments);
```


Question 8:

Retrieve the details of passengers who booked flights with a departure time later than 12:00 PM.

Screenshot 8:

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
123
124 • Select p.BookingID,p.amount, b.flightID, f.FlightNumber, f.Origin, F.destination from Payments p JOIN Bookings b ON p.bookingID = b.booki
125
126 • SELECT P.PassengerID,P.FirstName, P.LastName,F.FlightID,F.FlightNumber, F.Origin, F.Destination,F.DepartureTime
127 FROM Passengers P
128 JOIN
129 BOOKINGS B ON B.PassengerID=P.PassengerID
130 JOIN
131 Flights F ON B.FlightID = F.FlightID
132 WHERE
133 F.FLIGHTID
134 IN
135 (SELECT FLIGHTID FROM Flights WHERE TIME(DepartureTime) > '12:00:00');
```

The Results grid shows the following data:

PassengerID	FirstName	LastName	FlightID	FlightNumber	Origin	Destination	DepartureTime
103	Michael	Davis	3	DL789	Los Angeles	Seattle	2024-03-03 15:45:00
104	Olivia	Brown	4	BA789	London	New York	2024-03-04 14:00:00
105	Daniel	Johnson	5	LH456	Berlin	Tokyo	2024-03-05 20:30:00
106	Sophia	Lee	6	AF123	Paris	Dubai	2024-03-06 12:45:00
108	Ava	Jones	8	EK456	Dubai	New York	2024-03-08 18:30:00

The Output pane shows the execution progress:

#	Time	Action	Message	Duration / Fetch
254	11:25:18	SELECT * FROM flights LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
255	11:26:39	SELECT * FROM bookings LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
256	11:26:58	SELECT P.PassengerID,P.FirstName, P.LastName,F.FlightID,F.FlightNumber, F.Origin, F.Destination,F.DepartureTime FROM Passengers P JOIN BO...	5 row(s) returned	0.000 sec / 0.000 sec

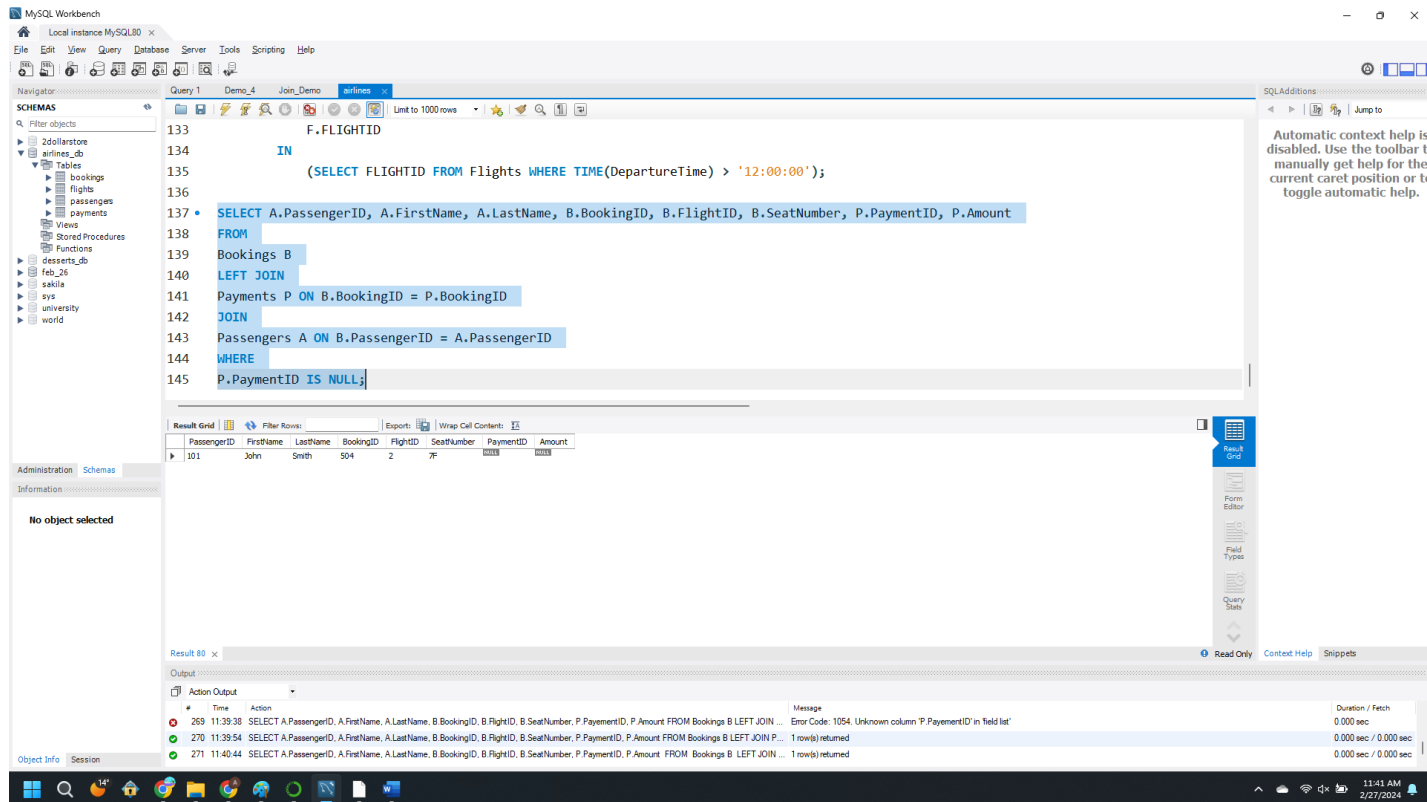
Query 8:

```
SELECT P.PassengerID,P.FirstName, P.LastName,F.FlightID,F.FlightNumber, F.Origin,
F.Destination,F.DepartureTime
FROM Passengers P
JOIN
BOOKINGS B ON B.PassengerID=P.PassengerID
JOIN
Flights F ON B.FlightID = F.FlightID
WHERE
F.FLIGHTID
IN
(SELECT FLIGHTID FROM P Flights WHERE TIME(DepartureTime) > '12:00:00');
```

Question 9:

Find the passengers who booked a seat on a flight with no associated payment.

Screenshot 9:



Query 9:

SELECT A.PassengerID, A.FirstName, A.LastName, B.BookingID, B.FlightID, B.SeatNumber, P.PaymentID, P.Amount FROM

Bookings B

LEFT JOIN

Payments P ON B.BookingID = P.BookingID

JOIN

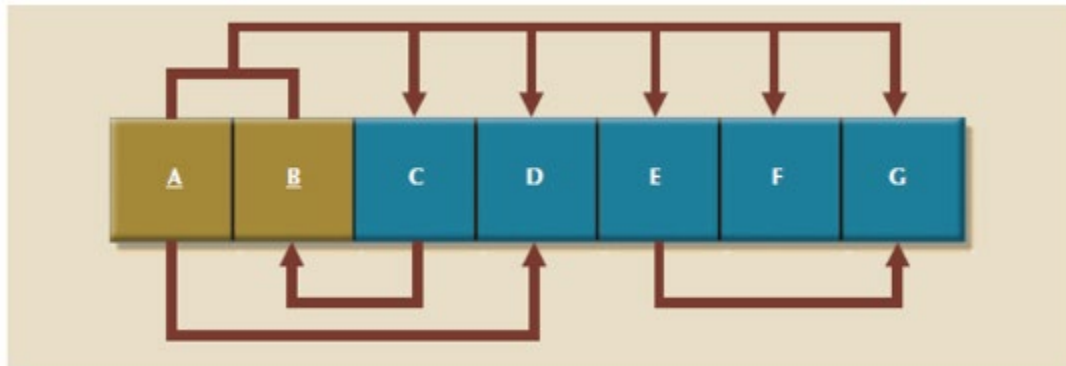
Passengers A ON B.PassengerID = A.PassengerID

WHERE

P.PaymentID IS NULL;

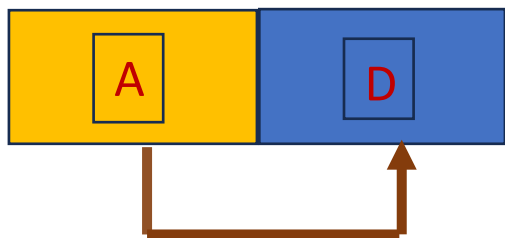
Question 10:

Break up the dependency diagram below to 3NF:

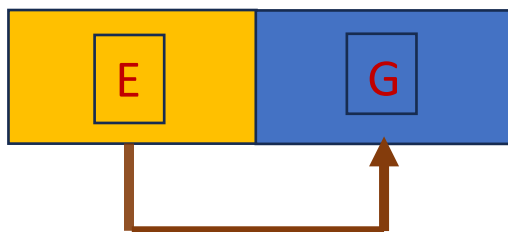


Answer:

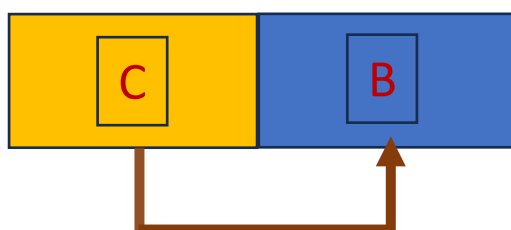
A and D has partial dependency, so they both form a new table.



E and G has transitive dependency, in 3NF transitive dependency should be removed, so E and G form a separate table.



C determines the value of the key attribute B. So C and B form a separate table.



A, C and E together can be used to determine F, so they all form a separate table.

