SQL

Consider the following schema for an airline database (primary key attributes are in **bold**):

- FLIGHTS (flight_num, source_city, destination_city)
- DEPARTURES (flight_num, date, plane_type)
- PASSENGERS (passenger_id, passenger_name, passenger_address)
- BOOKINGS (passenger_id, flight_num, date, seat_number)

Express the following queries in SQL (feel free to abbreviate relation and attribute names and to use INTERSECT and EXCEPT if you need to):

a) Find the passenger_id of all passengers who have a seat booked on a plane of type "747" from San Francisco to Washington. Do not return any duplicate values.

```sql
SELECT DISTINCT B.passenger_id
FROM Flights F, Departures D, Bookings B
WHERE B.flight_num = D.flight_num
AND B.date = D.date
AND F.flight_num = D.flight_num
AND F.source_city = "San Francisco"
AND F.destination_city = "Washington"
AND D.plane_type = "747";
```

*Since key of Departures is flight_num and date, you need both of these to do the join and find out what type of plane the passenger is booked on.*

*We can also write the above using NATURAL JOIN*

b) Find the cities that have direct (non-stop) flights to both Honolulu and Newark

```sql
SELECT DISTINCT source_city
FROM Flights F
WHERE F.dest_city = "Honolulu"
AND F.source_city IN
(SELECT source_city
FROM Flights F2
WHERE dest_city = "Newark");
```

*Could also be done with a self join on Flights, or with INTERSECT, or..., just can’t use a simple selection with "AND" in the Where clause --- this would return no tuples*

c) Find the flight_num and date of all flights for which there are no reservations.
SELECT flight_num, date
FROM Departures D
WHERE NOT EXISTS
   (SELECT *
    FROM Bookings B
    WHERE B.flight_num = D.flight_num
    AND B.date = D.date);

Alternately here we could have used EXCEPT clause and taken the difference of Departures and Bookings.

Another alternate solution using OUTER JOINS
SELECT D.flight_num, D.date
FROM Departures D NATURAL LEFT OUTER JOIN Bookings B
WHERE B.passenger_id = NULL;

d) Find the passenger_name of all passengers who have a seat booked on at least one plane of every type.

SELECT DISTINCT passenger_name
FROM Passengers P
WHERE
   (SELECT COUNT(DISTINCT D.plane_type)
    FROM Departures D, Bookings B
    WHERE D.flight_num = B.flight_num
    AND D.date = B.date
    AND B.passenger_id = P.passenger_id)
   =
   (SELECT COUNT(DISTINCT D.plane_type)
    FROM Departures D);

Alternatively:
SELECT DISTINCT P.passenger_name
FROM Passengers P
WHERE NOT EXISTS
   (SELECT D.plane_type
    FROM Departures D
    WHERE NOT EXISTS
       (SELECT *
        FROM Departures D2, Bookings B
        WHERE D2.flight_num = B.flight_num
        AND D2.date = B.date
        AND B.passenger_id = P.passenger_id
        AND D.plane_type = D2.plane_type));

The first one is easier and does it by counting the number of plane types. The second is trickier but is similar to the technique used in the book.

e) Print an ordered list of all source cities and the number of distinct destination cities
that they have direct (non-stop) flights to. The list should be ordered in decreasing number of destinations and should contain only those source cities that have flights to 25 or more distinct destinations.

```
SELECT source_city,
COUNT(DISTINCT destination_city) AS NumDestinations
FROM Flights F
GROUP BY source_city
HAVING NumDestinations >= 25
ORDER BY NumDestinations DESC;
```

For example, the output should look like:

<table>
<thead>
<tr>
<th>Source_City</th>
<th>NumDestinations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chicago</td>
<td>120</td>
</tr>
<tr>
<td>Atlanta</td>
<td>106</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Austin</td>
<td>25</td>
</tr>
</tbody>
</table>