```
################### DATABASE SETUP ###################


# Setup for the Passengers table
CREATE TABLE IF NOT EXISTS Passengers (
    PassengerID INT AUTO_INCREMENT,
    PassengerName VARCHAR (50) NOT NULL,
    PassengerEmail VARCHAR (50),
    PRIMARY KEY(PassengerID)
);

# Setup for the Airplanes table

CREATE TABLE IF NOT EXISTS Airplanes(
    TailNumber INT NOT NULL,
    Manufacturer VARCHAR(50) NOT NULL,
    ModelNumber INT NOT NULL,
    SeatCount INT,
    PRIMARY KEY(TailNumber)
);

# Setup for the Mechanics table

CREATE TABLE IF NOT EXISTS Mechanics(
    MechanicID INT AUTO_INCREMENT,
    MechanicName VARCHAR(50) NOT NULL,
    MechanicSpecialty VARCHAR(50),
    MechanicPhone INT(10),
    PRIMARY KEY(MechanicID)
);

# Setup for the FlightCrews table

CREATE TABLE IF NOT EXISTS FlightCrews(
    FlightCrewID INT AUTO_INCREMENT,
    FlightCrewName VARCHAR(50) NOT NULL,
    CrewCount INT,
    PRIMARY KEY(FlightCrewID)
);

# Setup for the Flights table
CREATE TABLE IF NOT EXISTS Flights(
```

```
    FlightID INT NOT NULL,
    TailNumber INT,
    FlightCrewID INT,
    FlightDate DATE NOT NULL,
    Origin VARCHAR(3) NOT NULL,
    DepartureTime TIME UNIQUE NOT NULL,
    Destination VARCHAR(3),
    ArrivalTime TIME UNIQUE,
    PRIMARY KEY(FlightID),
    FOREIGN KEY(TailNumber) REFERENCES
Airplanes(TailNumber) ON DELETE SET NULL,
    FOREIGN KEY(FlightCrewID) REFERENCES
FlightCrews(FlightCrewID) ON DELETE SET NULL

);


# Setup for the Tickets table
CREATE TABLE IF NOT EXISTS Tickets (
    TicketNumber INT AUTO_INCREMENT,
    PassengerID INT,
    FlightID INT,
    Price DECIMAL (10,2)DEFAULT 200,
    PRIMARY KEY(TicketNumber),
    FOREIGN KEY(PassengerID) REFERENCES
Passengers(PassengerID) ON DELETE SET NULL,
    FOREIGN KEY(FlightID) REFERENCES Flights(FlightID)ON
DELETE SET NULL
);


# Setup for the WorkOrders table

CREATE TABLE IF NOT EXISTS WorkOrders(
    WorkOrderID INT AUTO_INCREMENT,
    TailNumber INT,
    MechanicID INT,
    Description VARCHAR(50),
    Status VARCHAR(10),
    PRIMARY KEY(WorkOrderID),
    FOREIGN KEY(TailNumber) REFERENCES
Airplanes(TailNumber) ON DELETE SET NULL,
```

```
    FOREIGN KEY(MechanicID) REFERENCES
Mechanics(MechanicID) ON DELETE SET NULL
);
```

```
    ##################### QUERIES ####################
```

1. List the flight ID, tail number, departure time, and origin for all flights between January 1, 2021 and March 1, 2021 with a destination of “DCA”. Sort the result by the flight ID in ascending order.

   Code –

```
    SELECT FlightID,TailNumber, DepartureTime,Origin
     FROM Flights
     WHERE FlightDate BETWEEN ‘2021/01/01’ AND
‘2021/03/01’
      AND Destination = 'DCA'
      ORDER BY FlightID ASC;

      ************************************************
```

2. List the passenger IDs of all passengers that have purchased tickets over $1,000 in price. If the same passenger ID has purchased multiple tickets that meet this criterion, then only list the passenger ID once.

   Code –

```
   SELECT DISTINCT(PassengerID)
   FROM Tickets
   where Price > 1000;

     ************************************************
```

3. List all fields for work orders whose descriptions mention “oil”. Order these fields by status descending, then by tail number ascending.

   Code –

```
SELECT *
from WorkOrders
where Description LIKE BINARY "%oil%"
ORDER BY Status DESC, TailNumber ASC;
```

****

4. For each passenger name, list their name in addition to the date, origin, departure time, destination, and arrival time of matching all flights.

Code -

```
SELECT
P.PassengerName,P.Destination,F.FlightDate,F.Origin,F.
DepartureTime,F.ArrivalTime
FROM Passengers P
JOIN Tickets T
ON P.PassengerID=T.PassengerID
JOIN Flights F
ON F.FlightID=T.FlightID
ORDER BY P.PassengerName ASC,F.FlightDate ASC;     .
```

****

5. a) Create a view that computes the average ticket price.

Code -

```
CREATE VIEW Average_Prices AS SELECT AVG(Price) AS
Avg_Prices FROM Tickets;
```

b) Based on the view created in (a), list each passenger name alongside the number of tickets that the passenger has purchased with above-average ticket prices.

Code -

```
    SELECT
P.PassengerID,P.PassengerName,T.TicketNumber,T.Price
    FROM Passengers P
    JOIN Tickets T
    ON P.PassengerID = T.PassengerID AND T.Price >
(Select (Avg_Prices) FROM average_prices )
    GROUP BY (P.PassengerName);
    ***********************************************
```

6. List all airplane tail numbers alongside the number of work orders pertaining to that airplane that have a "pending"status. Order by the number of work orders from high to low.

Code -

```
    SELECT A.TailNumber,COUNT(W.WorkOrderID) AS
NUMBER_OF_WORKORDERS
    FROM Airplanes A
    LEFT JOIN WorkOrders W
    ON A.TailNumber = W.TailNumber WHERE W.Status LIKE
"%Pending%"
    GROUP BY W.TailNumber
    ORDER BY COUNT(W.WorkOrderID) DESC;

    ***********************************************
```

7. List all flight crew names alongside all flight IDs and tail numbers to which those flight crews are assigned with an origin of "SAN". If a flight crew has not been assigned to a flight yet, then still list it in the results. Sort by the date in descending order, then by the departure time in descending order.

Code -

```
SELECT C.FlightCrewName,F.TailNumber,F.FlightID
FROM FlightCrews C
LEFT JOIN Flights F
ON C.FlightCrewID = F.FlightCrewID WHERE F.Origin =
"San"
```

```
ORDER BY F.FlightDate DESC,F.DepartureTime DESC;
```

****

8. Foreach passenger name, list their name, all flight destinations that they have flown to, and the sum of the ticket prices foreach destination. Sort by the sum of ticket prices in descending order.

Code –

```
SELECT P.PassengerName,F.Destination,SUM(T.Price) AS
Sum_Prices
FROM Passengers P
JOIN Tickets T
ON P.PassengerID=T.PassengerID
JOIN Flights F
ON F.FlightID=T.FlightID
GROUP BY P.PassengerName,F.Destination
ORDER BY Sum_Prices DESC;
```

****

9. List all descriptions of work orders with a status of “completed” assigned to mechanics with a name that starts with “Eric”.

Code –

```
SELECT W.Status,W.Description, M.MechanicName
FROM WorkOrders W
JOIN Mechanics M
ON M.MechanicID = W.MechanicID
AND W.Status="Completed" and M.MechanicName LIKE
"Eric%";
```

****

10. For each flight destination, list each airplane manufacturer whose airplanes have flown to that destination alongside the number of flights that the

manufacturer's flights have flown to that destination. Sort by the manufacturer in ascending order, then the number of flights in descending order.

Code –

```
SELECT F.Destination,A.Manufacturer,COUNT(F.FlightID)
AS Number_of_Flights
FROM Airplanes A
JOIN Flights F
ON A.TailNumber = F.TailNumber
GROUP BY F.Destination, A.Manufacturer
ORDER BY A.Manufacturer ASC,Number_of_Flights DESC;
```

****