**Q1. EER DIAGRAM**

Please find the file “EERdiagram.svg”

**Q2. (E)ER Reducing, Functional Dependencies, and Normalization**

**Q2.1. RELATIONAL SCHEMA**

**2.1.1.** WEBSITE **(URL)**

**2.1.2.** AIRPORT **(PORT IATA CODE,** LATITUDE, LONGITUDE,INFO**)**

**2.1.3.** AIRLINE **(LINE IATA CODE,** POLICIE, TRAVEL GUIDE LINES, AIRWEBSITE**)**

**2.1.4.** LOCATION **(**SOURCE, DESTINATION**)**

\* NB. I considered the departure of the flight as “SOURCE” and arrive of it as “DESTINATION”

**2.1.5.** TICKET **(TICKET NO.** , **FLIGHT NUMBER, PASSANGERS ID,** **OPERATOR ID,**TOTAL PRICE,SCR/DST LOCATION**)**

\* NB. I considered the departure of the flight as “SOURCE”(SCR) and arrive of it as “DESTINATION”(DST) for ticket.

**2.1.6.** ECONIMIC **(**NUMBER OF PASSANGERS**)**

**2.1.7.** FIRST CLASS **(**NUMBER OF PASSANGERS**)**

**2.1.8.** BUSINESS **(**NUMBER OF PASSANGERS**)**

**2.1.9.** PASSANGERS **(PASSPORT NO. ,USRE ID,** NAME**)**

**2.1.10.** FLIGHTROUT **(FLIGHT NO.,LINE IATA CODE, PORT IATA CODE**, SOURCE, DESTINATION**)**

**2.1.11.** FLIGHT **(FLIGHT NO., AIRLINE IATA CODE, AIRPORT IATA CODE**, DEPART, ARRIVE PRICE,OPTION ID, LANDING, TAKEOFF)

**2.1.12.** OPTION **(ID**, NAME,PRICE**)**

**2.1.13.** CUSTOMER **(ID USER)**

**2.1.14.** OPERATOR **(ID USER,** NAME, PHONE NUMBER, ADDRESS**)**

**2.1.15.** MODERATOR **(ID USER)**

**2.1.16.** BOOK **(ID USER ,FLIGHT NO,** RETURN, CONNECTING, FOR OTHERS**)**

**Q2.2. FUNCTIONAL DEPENDENCY**

**2.2.1.** NO DEPENDENCIES

**2.2.2.** {AIRPORT IATA CODE}{LATITUDE, LONGITUDE, INFO}

**2.2.3.** {AIRLINE IATA CODE}{POLICIE, TRAVEL GUIDE LINES, AIRWEBSITE}

**2.2.4.** {SOURCE, DESTINATION} { DESTINATION }

{SOURCE, DESTINATION} { SOURCE }

**2.2.5. {**TICKET NO. } {FLIGHT NUMBER, PASSANGERS ID, OPERATOR ID,TOTAL PRICE,SCR/DST LOCATION}

**2.2.6.** NO DEPENDENCIES

**2.2.7.** NO DEPENDENCIES

**2.2.8.** NO DEPENDENCIES

**2.2.9.** {PASSPORT NO.}{NAME}

**2.2.10.** {FLIGHT NO.} {LINE IATA CODE, PORT IATA CODE, SOURCE, DESTINATION, PRICE}

{FLIGHT NO. , LINE IATA CODE, PORT IATA CODE, SOURCE, DESTINATION } {PRICE}

**2.2.11.** {FLIGHT NO.} {LINE IATA CODE, PORT IATA CODE, SOURCE, DESTINATION, PRICE}

{FLIGHT NO. , LINE IATA CODE, PORT IATA CODE, SOURCE, DESTINATION, OPTIONTYPE} {PRICE}

**2.2.12.** {ID.} {NAME, PRICE}

**2.2.13.** NO DEPENDENCIES

**2.2.14.** NO DEPENDENCIES

**2.2.15.** NO DEPENDENCIES

**2.2.16.** {ID USER , FLIGHT NO.} {RETURN, CONNECTING, FOR OTHERS}

**Q2.3. NORMALIZATION**

There are some changes to the relational schema and EER after normalization in my opinion. First of all, I had considered several entities to make a less messy diagram but after thinking about the normalization some would join to make a concise entity. All these entities are now in ***BCNF***.

The entities and their related attributes are as follows in the below picture:

**Diagram

Description automatically generated**

**After Normalization the entities and attributes could be as follows in the next page, I have put them all in one page for easier tracking.**

1. **Airport (AIRPORT IATA CODE,** LATITUDE, LONGITUDE, INFO**)**

AIRPORT IATA CODE is the Primary key and candidate key.

1. **Flight (FlightNumber, AIRLINE IATA CODE, OPTION ID, SRC AIRPORT IATA CODE, DEST AIRPORT IATA,** TAKEOFF, CLASS, PASSENGERSLIMIT**)**

OPTION ID: FORIGN KEY

FLIGHT NUMBER: PRIMARY KEY

SRC AIRPORT IATA CODE, DEST AIRPORT IATA: CANDIDATE KEY

1. **FlightOption** (**ID**, NAME, PRICE)

ID: PRIMARY KEY and CANDIDATE KEY

1. **Airline** (**AIRLINE IATA CODE,** POLICIE, TRAVEL GUIDE LINES, AIRWEBSITE)

AIRLINE IATA CODE: PRIMARY KEY and CANDIDATE KEY

1. **Passenger** (**ID**, **PASSPORT NUMBER, USER ID**)

USER ID: FORIGN KEY

ID: PRIMARY KEY

PASSPORT NUMBER: CANDIDATE KEY

1. **TICKET** (**ID**, **FLIGHT NUMBER, PASSANGERS ID,** **OPERATOR ID,** TOTAL PRICE)

FLIGHT NUMBER, PASSANGERS ID, OPERATOR ID: FORIGN KEY

ID: PRIMARY KEY and CANDIDATE KEY

1. **OPERATOR (ID, ID USER, AIRLINE IATA CODE)**

ID: PRIMARY KEY

ID USER and AIRLINE IATA CODE: FORIGN KEY

1. **USER** (**ID USER,** NAME, PHONE NUMBER, ADDRESS)

**ID USER**: PRIMARY KEY and CANDIDATE KEY

**Q3. SQL Creation**

CREATE DATABASE WebsiteDB;

CREATE TABLE Airport(

IATA VARCHAR(100) UNIQUE NOT NULL,

Latitude VARCHAR(100) NOT NULL,

Longitude VARCHAR(100) NOT NULL

INFO VARCHAR(250),

PRIMARY KEY(IATA)

);

CREATE TABLE Flight(

FlightNumber VARCHAR(50) UNIQUE NOT NULL,

AirlineIATA VARCHAR(20) NOT NULL,

SRCAIRPORTIATA VARCHAR(50) NOT NULL,

DESTAIRPORTIATA VARCHAR(50) NOT NULL,

TAKEOFF VARCHAR(250),

CLASS VARCHAR(250),

PASSENGERSLIMIT VARCHAR(250)

PRIMARY KEY(FlightNumber),

FOREIGN KEY(AirlineIATA) REFERENCES Airline(IATA),

FOREIGN KEY(SRCAIRPORTIATA) REFERENCES Airport(IATA),

FOREIGN KEY(DESTAIRPORTIATA) REFERENCES Airport(IATA),

);

CREATE TABLE Airline(

AIRLINEIATA VARCHAR(20) UNIQUE NOT NULL,

AIRWebsite VARCHAR(50) NOT NULL,

TravelGuide VARCHAR(250),

POLICIES VARCHAR(250),

PRIMARY KEY(AIRLINEIATA)

);

CREATE TABLE FlightOptions(

ID INT IDENTITY(1,1) NOT NULL,

Name VARCHAR(250) NOT NULL,

Price REAL NOT NULL,

PRIMARY KEY(ID),

);

CREATE TABLE Passenger(

ID INT IDENTITY(1,1) NOT NULL,

FlightNumber VARCHAR(50) UNIQUE NOT NULL,

PassportNumber VARCHAR(50) NOT NULL,

USER\_ID INT NOT NULL,

PRIMARY KEY(ID),

FOREIGN KEY(USER\_ID) REFERENCES USER(ID),

);

CREATE TABLE Ticket(

ID INT IDENTITY(1,1) NOT NULL,

OperatorID INT NOT NULL,

FlightNumber VARCHAR(50) NOT NULL,

PassengerID VARCHAR(20) NOT NULL,

TotalPrice FLOAT NOT NULL,

PRIMARY KEY(ID),

FOREIGN KEY(FlightNumber) REFERENCES Flight(ID),

FOREIGN KEY(OperatorID) REFERENCES OperatorUser(ID),

);

CREATE TABLE OperatorUser(

ID INT IDENTITY(1,1) NOT NULL,

USER\_ID INT NOT NULL,

AIRLINEIATA VARCHAR(20) NOT NULL

PRIMARY KEY(ID),

FOREIGN KEY(USER\_ID) REFERENCES User(ID),

FOREIGN KEY(AIRLINEIATA) REFERENCES Airline(IATA),

);

CREATE TABLE User(

ID INT IDENTITY(1,1) NOT NULL,

Name VARCHAR(50) NOT NULL

PhoneNumber VARCHAR(50) NOT NULL

Address VARCHAR(50) NOT NULL

PRIMARY KEY(ID),

);

**Q4. SQL, RELATIONAL ALGEBRA, OPTIMISATION**

**I have used “RA” as RELATIONAL ALGEBRA and “OP” as OPTIMISATION in follows**

1. ***List all flights from Brussels Airport to Rome, Italy for Friday the 2nd of July that still have empty seats.***

**SQL**:  
  
SELECT Flight.\*, sum(Class.PassengersLimit) as PassengersLimit from Flight  
Left join Passengers on Flight.ID = Passengers.ID  
group by Flight.ID

**RA**:

γ ID, FlightNumber, AIRLINE IATA CODE, OPTION ID, SRC AIRPORT IATA CODE, DEST AIRPORT IATA, TAKEOFF, CLASS, PASSENGERSLIMIT; count(PassengersLimit)→ PassengersLimit (Flight ⟕ Passengers)

**OP**:   
  
Optimization: There is no need for optimization because there is no WHERE clause and we have default index on the primary keys.

1. ***Ryanair wants to calculate how much money they got from a check-in luggage option for all their booked flights in 2020.***

**SQL:**  
SELECT sum(Option.price) as yearTotal   
FROM Option INNER join Flight on Flight.FlightNumber = Option. FlightNumber  
WHERE FlightNumber ('%Y',Flight.Date) == '2020'

**RA:**γ sum(Option.price)→ yearTotal ((σ FlightNumber ('%Y', Flight.Date) == '2020' (FlightNumber ⨝ Flight)))

**OP:**   
  
SQL Server supports B+-Tree index, so I would create an index, which is obviously non-clustered, on Option (ID, Flight.Date).

1. ***Find the total amount of seats available for all flights from airport A to airport B on one specific day.***

**SQL:**SELECT distinct Flight.\*, count(Passengers.PassangersLimit) as SeatCount from Flight   
INNER join Airport on Airport. AirportIATACode = Flight.FlightNumber  
INNER join Passengers on Flight. FlightNumber = Flight. FlightNumber  
GROUP BY Flight.FlightNumber  
Having Passengers. PassangersLimit in(‘Airport A’,'Airport B')

**RA:**σ total(SeatCount) (σ PassangersLimit = ‘Airport A’∨ PassangersLimit = 'Airport B' (γ FlightNumber, SrcAirportIATACode, Dest AirportIATACode ; count(AirportIATACode)→ SeatCount (Flight ⟕ Passenger ⟕ Airport)))

**OP:**

B+-tree index on Passenger. PassangersLimit

CREATE INDEX IF NOT EXISTS indexPassengersPassangersLimit ON model (PassangersLimit)

1. ***Find the airline that received the most customers in first class (no date range specified) and provide a column that shows how many percentages of the total amount of passenger’s travels in first class.***
2. **SQL:**SELECT DISTINCT Airline. AIRLINE IATA CODE, Airline, AIRWEBSITE, sum(CustomersID.Class) as maxnumberFirstClass from USERS  
   INNER join Airline on AIRLINE. AIRLINE IATA CODE = AirlineNumber  
   INNER join CustomerClass on UserClass.PassangersLimit = Customer.UserID   
   INNER join Passenger on Passenger. Passenger Id = PassengerCustomerUserID  
   WHERE Passenger. PassangersLimit is not NULL  
   GROUP by AIRLINE. AIRLINE IATA CODE  
   order by sum(CustomersID.Class)

**RA:**π max(numberFirstClassX) (γ CustomersID.Class X; Airline. AIRLINE IATA CODE → numberFirstClass (Customer ⨝ (σ CustomerClass = 'X' (Airline))))

**OP:** B+-tree index on Passenger. PassangersLimit

CREATE INDEX IF NOT EXISTS indexsale ON Passenger(PassangersLimit)

1. Find a pair of flights from BRU (Brussels) to SYD (Sydney) that have a connecting stop. Provide the total duration of the flight (including waiting time).  
   Note 1: An example is EK184/EK414 from Emirates that stops in DXB (Dubai)  
   Note 2: A connected flight means that the 2nd flight happens after the 1st flight landed in the connecting airport

**SQL:**SELECT Flight.\*, FlightNumber, SRC AIRPORT IATA CODE, DEST AIRPORT IATA, SUM(Flight. total duration) from employee  
inner join Flight ON AIRPORT.AIRPORT IATA CODE = AIRPORT.AIRPORT IATA CODE and Flight. FlightNumber = AIRPORT. FlightNumber

INNER JOIN Book on Book.connecting = Flight.AIRPORT IATA CODE  
GROUP by Flight.AIRPORT IATA CODE  
HAVING SUM(Flight. total duration) >= (select AVG(sumTotal) from

(SELECT SUM(F.totalduration) as sumTotal from Flight f  
INNER join duration d on Book.connecting = d. Book.connecting where Book.connecting = Flight.duration and f. AIRPORT = Flight.d AIRPORT GROUP by F.totalduration) td)

**OP:**   
B+-tree index on Flight.duration and Flight.d AIRPORT. CREATE INDEX IF NOT EXISTS indexsale ON Flight (dConnecting, dAirport)

**More About The EER diagram for Clarification:**

In my Diagram, Website is the mother of other entities. Main entities like : Airport, Airline, User, are the branches of website. Website is actually the database we are supposed to design in this project. I considered some real-world aspects in this project to make it smooth to understand.

Website should have and address or URL which uniquely identifies it. So it would be the key attribute and one and only attribute for website.

Airport is another entity which has Location. I considered Location as an separate entity but in normalization model location would be joined with airport. Beside, I added two attribute by my own to airport, which are Lattitude and Longitude, Airport has its own primary key too.

Airline is clear like airport with the key attribute and other related attributes.

The major part and important section of the diagram I think is the Flight and FlightRout. In diagram I considered each of them as a separate entity. Flight is the head of these entities: Flightrout, Passengers, Class, Ticket.

Passengers is one of the major entities as I used it in my SQL queries. Passengers are identified with and ID in my diagram.

User is an entity with three subEntities which are disjoint form it. I named Airline user as Operator. Operator is of the important entity which acts roles after normalization as it can book a flight and modify its option, so the operator is impressive on the price.

Location Entity will fade after normalization, but I have considered date and time on two branches of it which are Source and Destination. Flight and Airport are connected to the location so they can handle the location in them.

I considered Connecting and Returning as the attribute of the Book relation which can be linked to the customer.

August 2021

2nd Session Database Project

Mahtab Hosseinalizadehshalmani