"Airline Reservation System"

Submitted as Final Project Report for

CS 6360.004 – Database Design

Submitted by:

Shobhit Agarwal (sxa138130)

Kunal Arora (kxa142230)

Indervir Singh (isb140030)

Bala Yadav (bxy140430)

DEPARTMENT OF ENGINEERING & COMPUTER SCIENCE

University of Texas at Dallas

Dallas, Texas

Dec 2014

ACKNOWLEDGEMENT

It is a great pleasure to have the opportunity to extend our heartiest felt gratitude to everybody who helped us throughout the course of this project.

It is distinct pleasure to express our deep sense of gratitude and indebtedness to our learned professor, **Ms. Nurcan Yuruk** for their invaluable guidance, encouragement and patient reviews. With their continuous inspiration only, it becomes possible to complete this Project.

Shobhit Agarwal (sxa138130)

Kunal Arora (kxa142230)

Indervir Singh (isb140030)

Bala Yadav (bxy140430)

Table of Content

1.INTRODUCTION

1.1 Overview

Airline reservation system is one of the most used database system in the world. It is an example of Transaction processing systems. **Transaction processing systems** are systems with large databases and hundreds of concurrent users executing database transactions. These systems require high availability and fast response time for hundreds of concurrent users.

We define the concept of a transaction, which is used to represent a logical unit of database processing, that must be completed in its entirety to ensure correct- ness. A transaction is typically implemented by a computer program, which includes database commands such as retrievals, insertions, deletions, and updates.

In this project, we deal with the database part of the whole system with insertions, deletions and updates as the primary task of our system along with maintenance of the integrity of the system at all stages of a transaction.

The system that is being described in this report handles everything that the most practical systems would do. The complexity of the database system has been handled by trying to make most basic entities resembling the real world objects.

Our database handles the most basic functions of an airline reservation system, including reservation, cancellation and updating of a flight trip transaction.

1.2 Feasibility Study & Risk Analysis:

It is the most difficult area to assess because objectives, functions, and performance are somewhat hazy; anything seems possible if right assumptions are made.

A clinical attitude should prevail during an evaluation of technical feasibility. The considerations normally attached with the technical feasibility:

Development Risk:

- Can the system element be designed so that necessary function and performance are achieved within the constraints uncovered during analysis

Resource availability:

- Are skilled staffs available to develop the system element in Question? Are any other necessary resources (hardware and software) available to build the system?

Technology:

- Has the relevant technology progressed to a state that will support the system?
- All of the above consideration also applies to the work we have done. As far as developments, risks are concerned, yes necessary functions and the constraints under which they have to perform have been identified and divided into modules so that each module performs its own assigned task.
- As for skilled staff for development is concerned, we are the only person performing this task and we have fully understood the problem. We are sufficiently equipped with the use of programming and can perform this tasks in the given time constraints.
- The use of programming language enables the programmer to develop software that can help end-user to operate the system more easily. We can use GUI tool VB.NET or Java to build a GUI for the system but the report deals with only the database design part of the system and for that we use Oracle SQL developer. The tool comprises of all the components required to solve the problem system.

Economical Feasibility:

An evaluation of development cost weighed against the ultimate income or benefit derived from the development system or product. It includes a broad range of concerns such as:

- Cost-benefit Analysis
- Long-term corporate income strategies
- Impact on other profits/products
- Cost of resources needed for development
- Potential market growth

The work being done is economically feasible since the work is not being done at very large scale, although it might be a bit complex. The cost of resources needed to do the work would not be big. The whole task could be completed by a single resource in given time.

Operational Feasibility:

This study helps us in finding whether the work to be done will be operational with the available staff and with in the given time. The staff is fully capable of handling the database system. The IT literacy is of good order and the software has been made in such a way that it becomes easier for the user to answer queries being asked. This will facilitate easy use and adoptability of the system. Based on this, it was felt that the proposed system would be operationally feasible. With the use of menus, simple command buttons and proper validation required it become fully understandable to the common user and operational with the user. Moreover, we can pack and associate different canned queries with the GUI components to make it more friendly for users and also apply principles from cognitive science to deal with the user interaction with our system.

2. ER-DIAGRAM

Day by day, IT is becoming an important part of our life. The way we manage our travel is no longer the same. Considering the impact of IT on Airline Reservation Systems, we present one of the most practical scenarios of Airline Reservation System where users tend to book their tickets nowadays.

We start with the airport. Airport is an independent entity, which is uniquely identified by a code with address as an important attribute. Address contains different attributes under it; we make this as a composite attribute. Similarly we will define the primary attributes and other attributes for different entities in the report subsequently. We will lay more stress on the relationships. Now we move to airline. We uniquely identify the plane and assume that a plane has an access to the airport or not. First we thought of giving the access to Airline Company but practically, it is a very strict criterion. Sometimes planes might be landed on an airport because of emergency landing or other issues and because the company is not allowed, it might lead to a disaster. So we limit it to a plane. If we give access to a company, we are giving the access to all the planes of that company.

So we state our first assumption:

1) The access to an airport is dependent on the airplane and not the airline company. An airport can give access to multiple planes and a plane can have access to multiple airports.

Moving to Airline Company, every company owns multiple planes and every plane must belong to a company. We assume that every airplane belongs to a company and needs to be classified like that for properly managing the data. Even the private jets and chartered flights belong to some company. Even the planes belonging to Ministry of Defense belong to companies like Lockheed Martin, Rockwell etc.

So we state our second assumption:

2) Every plane must belong to an airline company. So a full participation of plane is justified. A company can have multiple planes but a plane must belong to one company. A plane cannot be shared across multiple companies as this scenario is very rarely observed. Also, users should have clear idea about which airline companies are they flying in. For e.g. Qatar, Etihad are better

service providers than Lufthansa or British Airways. Users should have the freedom to chose the company and each airplane must belong to a unique company.

Seat in a plane is the basic element of ticket reservation. The mode of travel is airplane. So we assume every seat belongs to an airplane.

Otherwise there might be some other mode of travel, which can have seats. But as we strictly provide air travel services to our customers using this database, each seat belongs to a plane and a plane can have multiple seats. Its number and the airline number can define each seat. The reason is that different airlines can have same seat numbers. So only a combination of seat number and the plane number can define a seat uniquely.

So we state our third assumption:

3) Each seat belongs to a plane and plane has multiple seats. Without a plane, seat cannot exist. Seat is a dependent attribute (plane).

We introduce an entity called traveller who occupies the seat. Please note that it is not the user because a user can be agent or anybody booking the tickets on behalf of the traveller. A traveller is a person for whom the ticket has been booked and he is travelling in a flight trip. Hence he will occupy a seat.

So we state our next assumption:

4) Each traveller must have a seat. However, seats can be empty so its not necessary that every seat belongs to a traveller but the opposite is possible. So a full participation of traveller is justified. Before understanding the definition of a flight trip, lets understand what a hop is. A travel from A to C can be via B. So we define a hop as a travel between a take off and immediate next landing. So this trip has two hops: A-B and B-C. Each hop has a ID and is not dependent on that trip because we do not know if this hop belongs to multiple journeys or not. Some traveller may be traveling from B-C in the same plane at the same time but his journey might be B-C-D. So each hop has a unique ID (with a take off and landing airport). Every hop must have a departure airport, arrival airport and a plane. These are the elements, which define a hop.

To be more pragmatic, these airports, plane and timestamp will define a hop but this will involve too many attributes as a part of PK. So we define a hop ID.

Please note that it has a base fare also, which will be used subsequently.

So we assume:

5) A unique hop can belong to multiple trips and a trip can have multiple hops. Every hop has a departure airport, arrival airport and a plane. Now, a flight trip is a journey with all possible details against traveller(s) (with multiple hops). It is like a transaction but a transaction contains many more details, which we are not including because of complexities and this may get out of context, as it will involve banks and intermediate merchants. This is where our database gives the flexibility of expansion. We can further link the airline reservation system with other systems and then make a complete, more complex system. Each flight trip has a fare and obviously, a fare cannot be defined without a flight trip. Each trip can have multiple hops and a hop has a base fare. When a user books a flight trip, the total fare can be calculated as the sum of the fares of all the hop-fares minus discount.

Now as all travellers are defined against a trip, no traveller can exist without a trip.

So:

6) Traveller and fare cannot exist without a trip and a trip has one fare and multiple travellers (depending on for how many travellers, the user books). Finally, the user is the one who is booking the trip and has a unique id. He can book as many flight trips he wants (agent) for as many travellers.

User can be of three types:

a) Agent: Usually books for multiple travellers regularly.

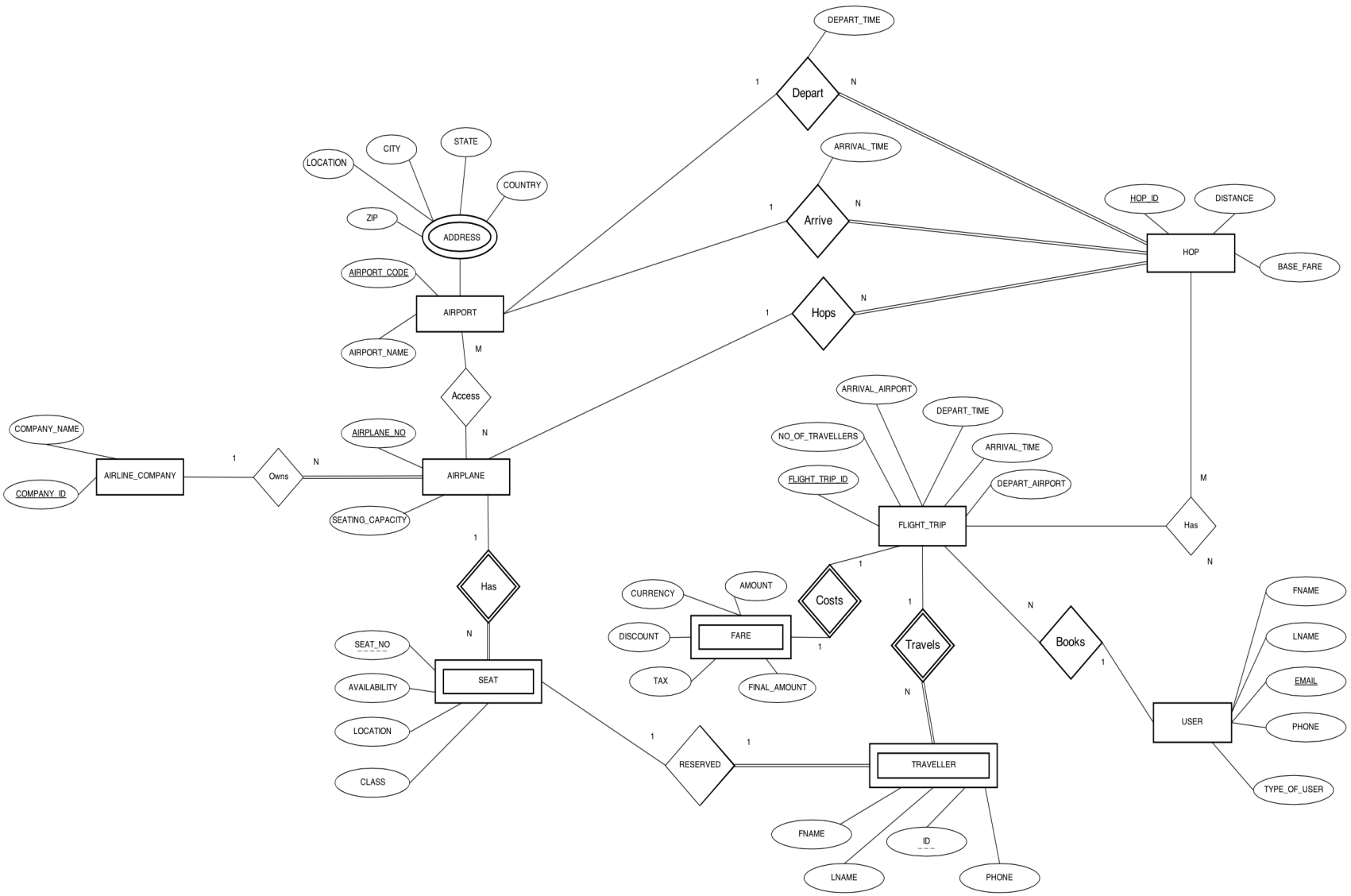
b) Traveller: Usually books for himself or at most family/friends.

c) Airline agent: Can be found at company counter usually at the airport.

Following assumption arises:

7) A user books tickets for multiple travellers and is an independent entity unlike traveller.

Based on all the above-mentioned assumptions and the assumptions for relations mentioned in the next section, we come up with the ER diagram as follows:

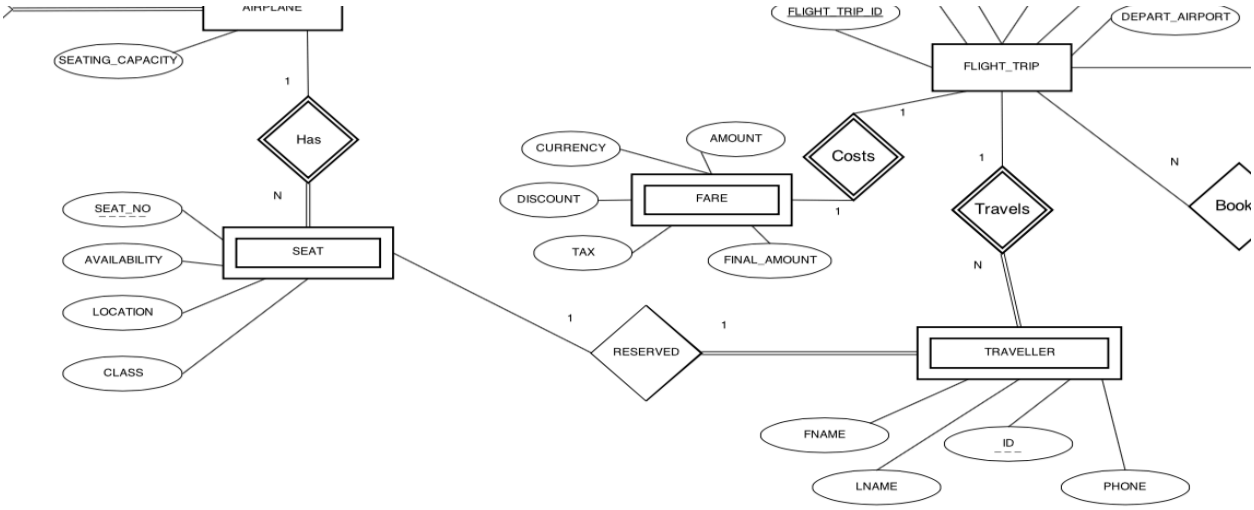

DEPART_TIME
Depart
1
N
ARRIVAL_TIME
Arrive
HOP_ID
DISTANCE
HOP
BASE_FARE
Hops
LOCATION
CITY
STATE
COUNTRY
ZIP
ADDRESS
AIRPORT_CODE
AIRPORT
AIRPORT_NAME
M
Access
COMPANY_NAME
COMPANY_ID
AIRLINE_COMPANY
Owns
AIRPLANE_NO
AIRPLANE
SEATING_CAPACITY
ARRIVAL_AIRPORT
DEPART_TIME
NO_OF_TRAVELLERS
ARRIVAL_TIME
FLIGHT_TRIP_ID
DEPART_AIRPORT
FLIGHT_TRIP
Has
Has
SEAT_NO
AVAILABILITY
LOCATION
CLASS
SEAT
CURRENCY
AMOUNT
DISCOUNT
FARE
TAX
FINAL_AMOUNT
Costs
Travels
Books
RESERVED
TRAVELLER
FNAME
ID
LNAME
PHONE
FNAME
LNAME
EMAIL
PHONE
USER
TYPE_OF_USER

3. RELATIONSHIP ASSUMPTIONS FOR ER-DIAGRAM

- An airline company can own many airplanes, but no airplanes can be without an airline company.
- Address is a multi-value attribute
- Multiple airplanes can land on an airport and also multiple airports can handle multiple airplanes
- An airplane-seat cannot exists without an airplane
- A single seat will be reserved for a single traveller
- A traveller can have multiple flight trips and there is no traveller without a flight trip i.e. if there is no flight trip, there wont be any traveller for it
- Fare exists only if there is a flight trip i.e. if there is no flight trip, there wont be a cost for it
- A user can book multiple flight trips
- A flight trip can have multiple hops and a particular hop can be used by multiple flight trips

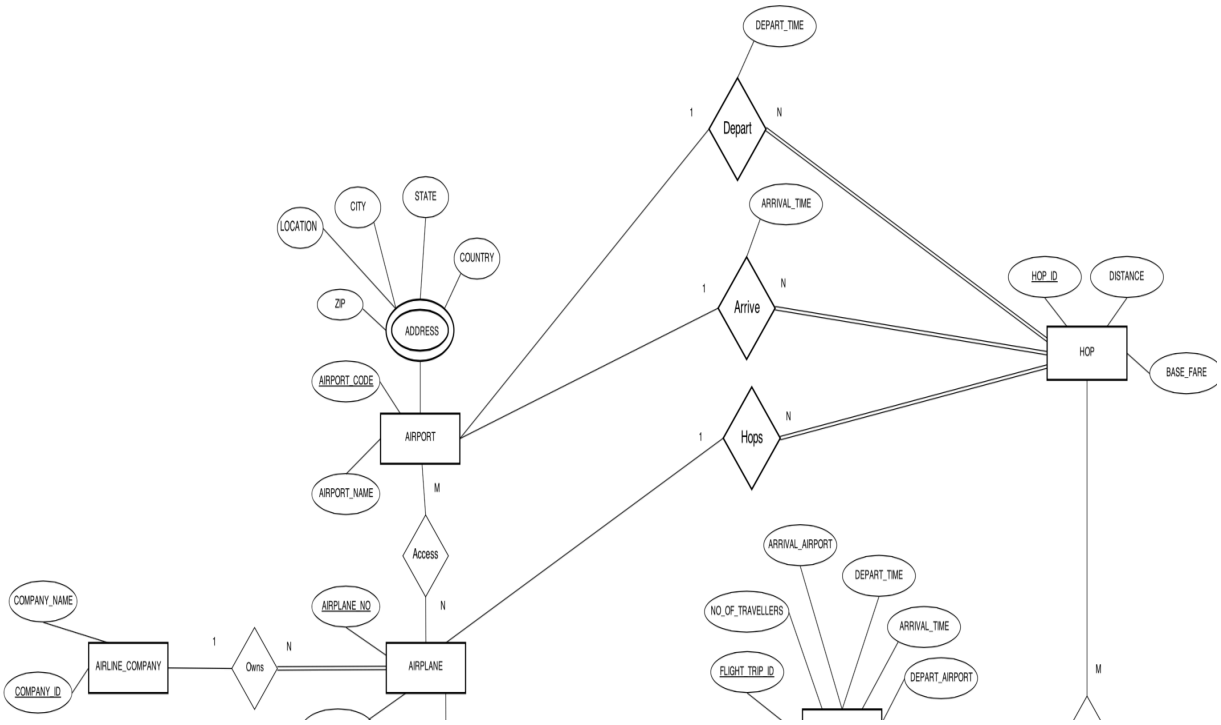
4. RELATIONSHIPS ACCORDING TO REQUIREMENTS

4.1 One-to-One Binary Relationships

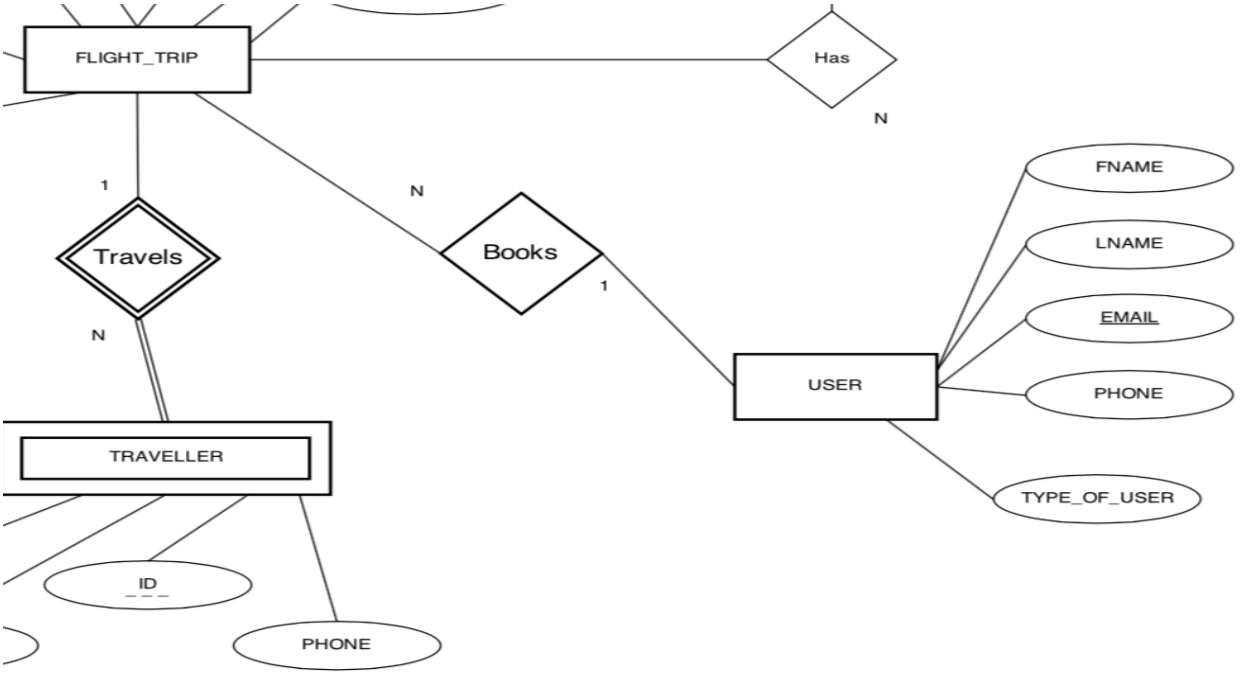


- SEAT-reserved-TRAVELLER
- FLIGHT_TRIP - costs – FARE

4.2 One-to-Many Binary Relationships

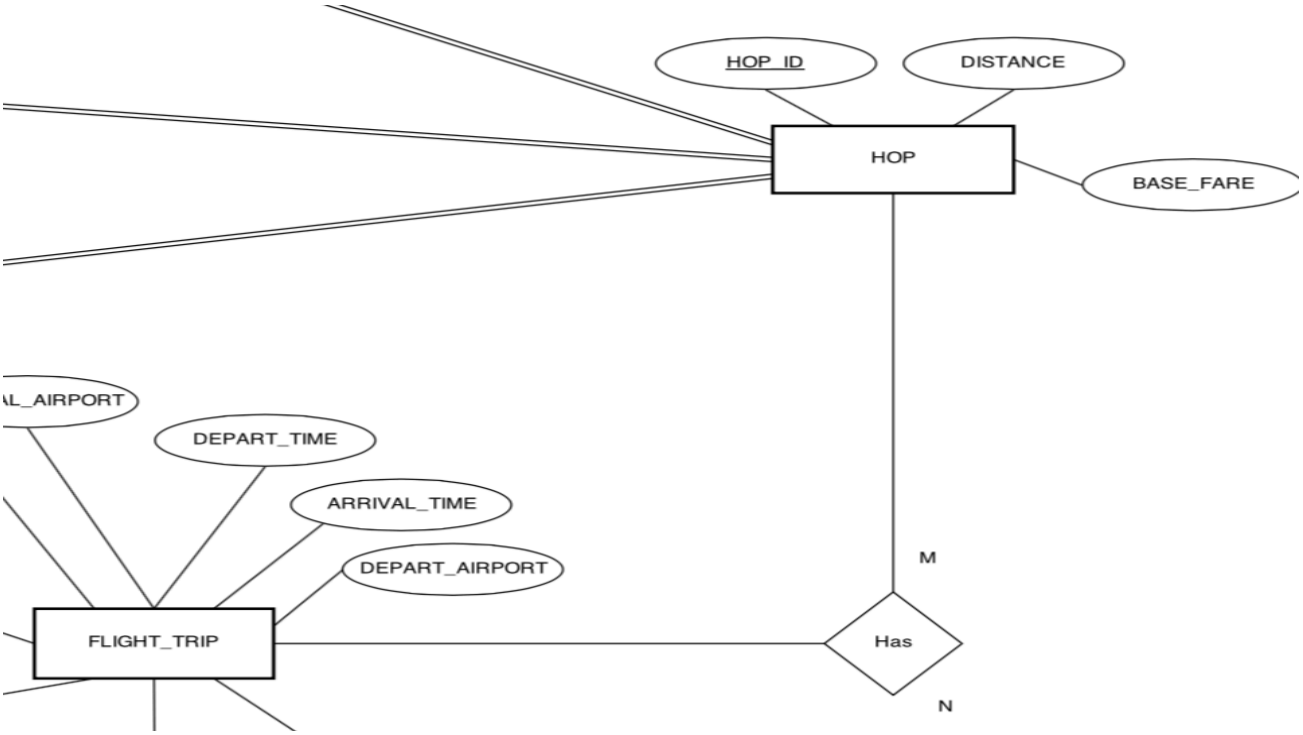


- AIRPLANE - hops - HOP
- AIRPORT - arrive - HOP
- AIRPORT - depart – HOP
- AIRLINE_COMPANY - owns - AIRPLANE
- AIRPLANE - has - SEATS



- FLIGHT_TRIP - travels – TRAVELLER
- USER - books - FLIGHT_TRIP

4.3 Many-to-Many Binary Relationships



- AIRPORT - access - AIRPLANE
- FLIGHT_TRIP - has - HOP

5. RELATIONAL SCHEMA

A relational database schema is the tables, columns and relationships that make up a relational database.

Purpose:

A relational database schema helps us to organize and understand the structure of a database. This is particularly useful when designing a new database, modifying an existing database to support more functionality, or building integration between databases.

Creating the Schema:

There are two steps to creating a relational database schema: creating the logical schema and creating the physical schema. The logical schema depicts the structure of the database, showing the tables, columns and relationships with other tables in the database and is a direct mapping of Entity-Relationship diagram.

The physical schema is created by actually generating the tables, columns and relationships in the relational database management software (RDBMS) i.e SQL queries to create the database tables and relationships defined.

Rules for ER-Relational Mapping:

Please refer the ER Diagram from previous section

Step 1: Mapping of Regular Entity Types

- For regular entities AIRPORT,AIRPLANE,AIRLINE_COMPANY,FLIGHT_TRIP,USER,HOP, the relations AIRPORT,AIRPLANE,AIRLINE_COMPANY,FLIGHT_TRIP,USER,HOP have been created respectively. For each of the above relations, the primary keys are selected as AIRPORT.AIRPORT_CODE,AIRPLANE.AIRPLANE_NO,AIRLINE_COMPANY.COMPANY_ID,FLIGHT_TRIP.FLIGHT_TRIP_ID,USER.EMAIL,HOP.HOP_ID. All other attributes for each of the entities have been forwarded as simple attributes.

Step 2: Mapping of Weak Entity Types

- For weak entities (SEAT,TRAVELLER,FARE), the relations SEAT,TRAVELLER,FARE have beencreated with primary keys as SEAT(AIRPLANE,SEAT_NO),TRAVELLER_ITINERARY(FLIGHT,SEAT_NO,AIRPLANE),TRAVELLER_INFORMATION(ID)[Please note we have implemented 2NF decomposition here],FARE(FLIGHT_TRIP_ID) and all the simple attributes as shown in ER diagram.

Step 3: Mapping of Binary 1:1 relationships

- For 1:1 relationship between SEAT : TRAVELLER, Following the foreign key approach, the primary key of SEAT (AIRPLANE,SEAT_NO) have been placed in relation TRAVELLER(AIRPLANE,SEAT_NO) as foreign key.
- For 1:1 relationship between FLIGHT_TRIP : FARE, Following the foreign key approach, the primary key of FLIGHT_TRIP(FLIGHT_TRIP_ID) has been placed in relation FARE(FLIGHT_TRIP_ID). As this relation is also a Weak relation, the above mapping was already completed in Step2.

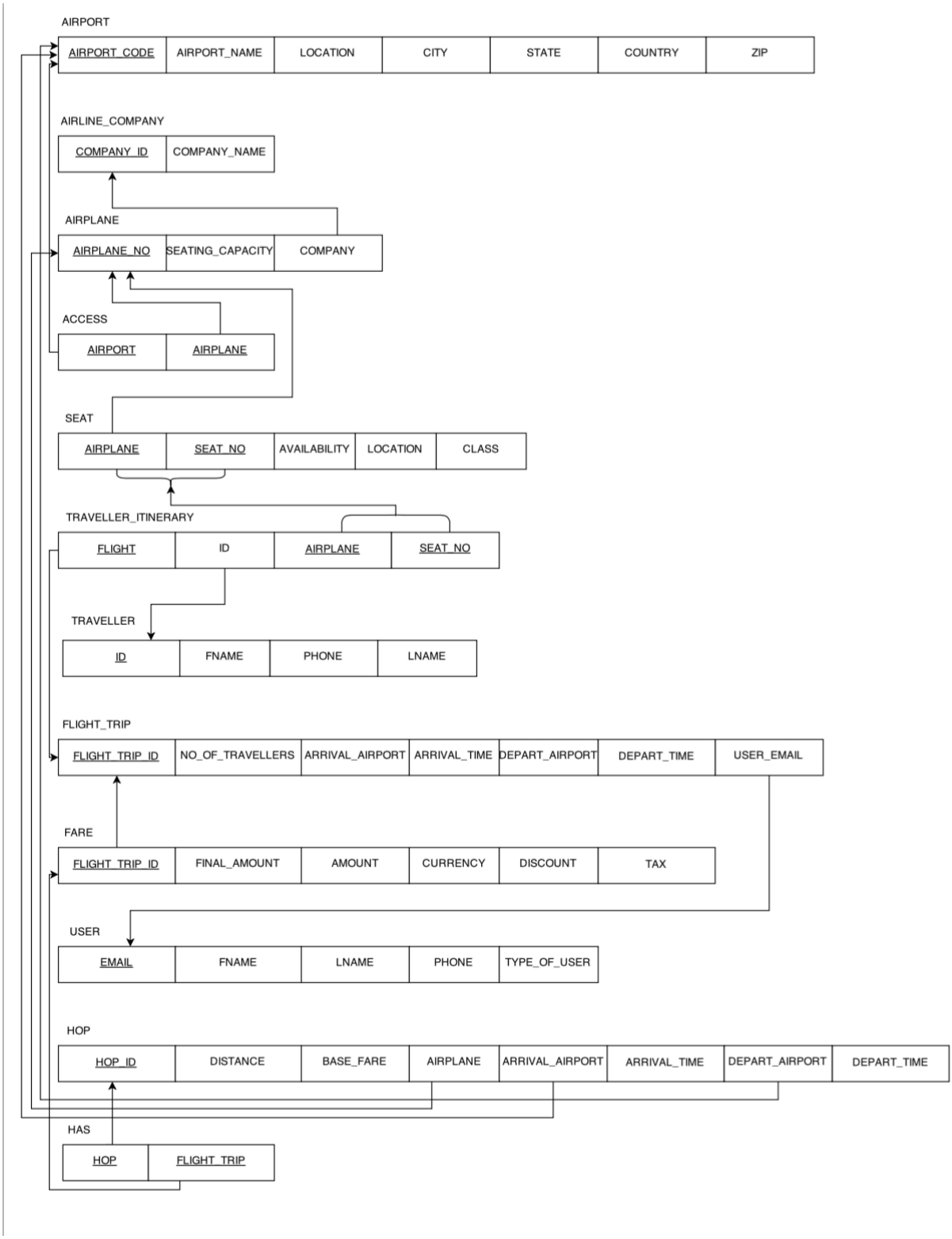
Step 4: Mapping of Binary 1:N relationships

- For 1:N relationship (namely 'Owns') between AIRLINE_COMPANY:AIRPLANE, the primary key of AIRLINE_COMPANY(COMPANY_ID) is included in AIRPLANE(COMPANY) as foreign key.
- For 1:N relationship (namely 'Has') between AIRPLANE:SEAT, the primary key of AIRPLANE(AIRPLANE_NO) is included in SEAT(AIRPLANE) as foreign key.
- For 1:N relationship (namely 'Travels') between FLIGHT_TRIP:TRAVELLER, the primary key of FLIGHT_TRIP(FLIGHT_TRIP_ID) is included in TRAVELLER(FLIGHT) as foreign key.
- For 1:N relationship (namely 'Books') between USER:FLIGHT_TRIP, the primary key of USER(EMAIL) is included in FLIGHT_TRIP(USER_EMAIL) as foreign key.
- For 1:N relationship (namely 'Depart') between AIRPORT:HOP, the primary key of AIRPORT(AIRPORT_CODE) is included in HOP(DEPART_AIRPORT) as foreign key and attribute(DEPART_TIME) of relation as simple attribute.
- For 1:N relationship (namely 'Arrive') between AIRPORT:HOP (Arrival), the primary key of AIRPORT(AIRPORT_CODE) is included in HOP(ARRIVAL_AIRPORT) as foreign key and attribute(ARRIVAL_TIME) of relation as simple attribute.
- For 1:N relationship (namely 'Hops') between AIRPLANE:HOP, the primary key of AIRPLANE(AIRPLANE_NO) is included in HOP(AIRPLANE) as foreign key.

Step 5: Mapping of Binary M: N Relationships

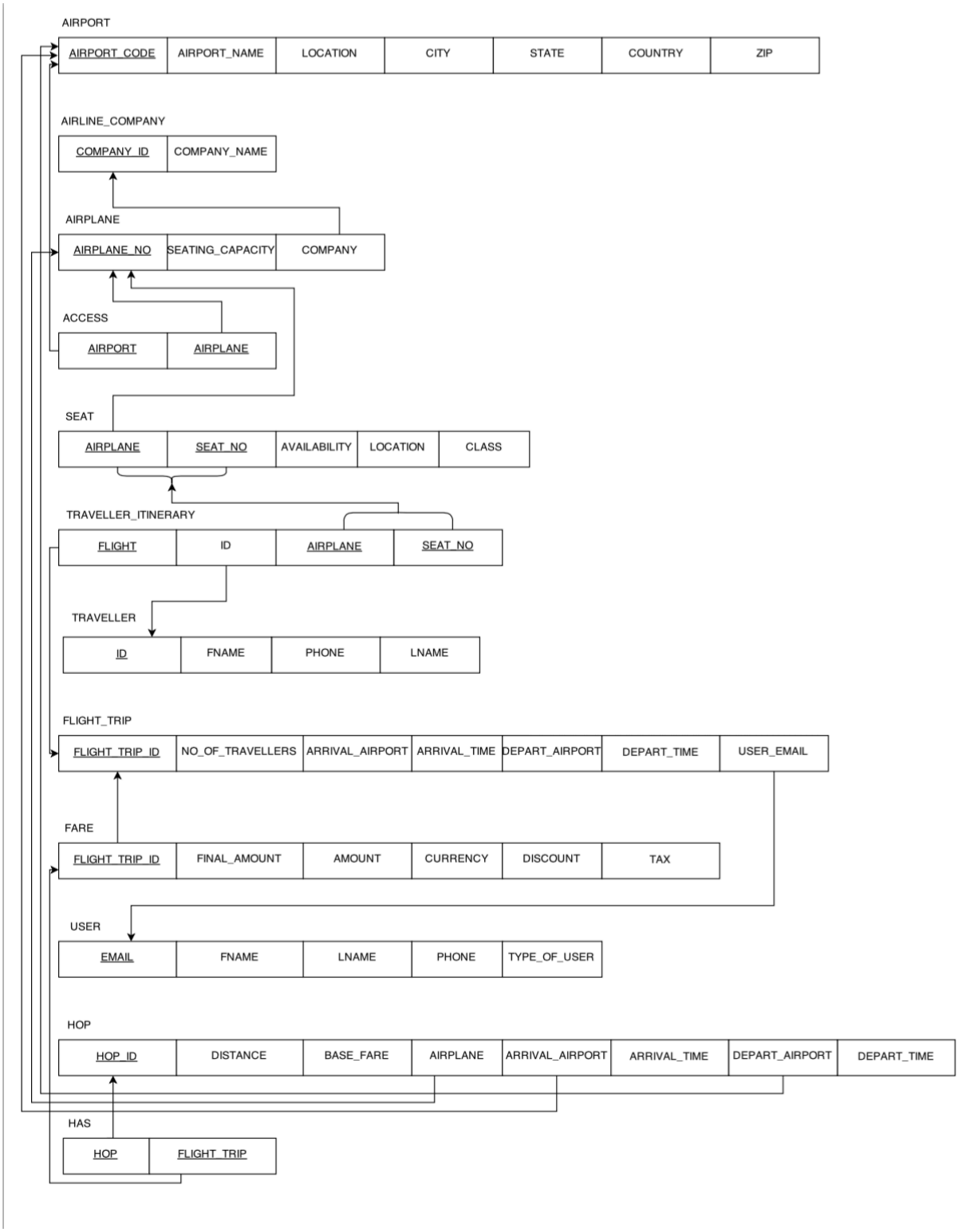
- For M: N relationship (Access) between AIRPLANE:AIRPORT, a new relation ACCESS has been created and primary keys of AIRPORT(AIRPORT_CODE) and AIRPLANE(AIRPLANE_NO) are included as AIRPORT,AIRPLANE respectively.
- For M:N relationship(Access) between HOP:FLIGHT_TRIP, a new relation 'HAS' has been created and primary keys of HOP(HOP_ID) and FLIGHT_TRIP(FLIGHT_TRIP_ID) are included as HOP,FLIGHT_TRIP respectively.

Following all the above rules, the Relational data schema is as follows



6. NORMALISATION OF SCHEMA

The normalized relational schema is as follows:



7. TABLES AND PRIMARY KEYS

These are the final tables that we have created for the normalized relational schema mentioned in the above schema. The primary key of each table is chosen carefully to ensure that any other attribute of the table is not dependent on entity other than the primary key. The table is as follows:

- AIRPORT - Airport_Code
- AIRLINE_COMPANY - Company_ID
- AIRPLANE - Airplane_no
- ACCESS - Airplane, Airport
- SEAT - Airplane, Seat_no
- TRAVELLER - ID
- TRAVELLER_ITINERARY – FLIGHT, ID
- FLIGHT_TRIP - Flight_trip_id
- FARE - Flight_trip_id
- USER - Email
- HOP - Hop_id
- HAS (HOP) - Hop, Flight_trip

8. QUERIES

8.1 CREATE Table & INSERT Queries:

- **AIRPORT:**

```
CREATE TABLE AIRPORT(
        AIRPORT_CODE        VARCHAR(25),
        AIRPORT_NAME        VARCHAR(100)    NOT NULL,
        LOCATION            VARCHAR(50)     NOT NULL,
        CITY                VARCHAR(50)     NOT NULL,
        AP_STATE            VARCHAR(50)     NOT NULL,
        COUNTRY             VARCHAR(50)     NOT NULL,
        ZIP                 NUMBER          NOT NULL,
        CONSTRAINT AIRPORT_PK PRIMARY KEY (AIRPORT_CODE)
);

INSERT ALL
    INTO AIRPORT VALUES( 'DFW' , 'Dallas/Fort Worth International Airport',
    '3200 E Airfield Dr','Dallas','Texas','USA',75261)
    INTO AIRPORT VALUES( 'DAL' , 'Dallas Love Field Airport', '8008 Cedar
    Springs Rd','Dallas','Texas','USA',75235)
    INTO AIRPORT VALUES( 'RDU' , 'Raleigh-Durham International Airport',
    '2400 John Brantley Blvd, Morrisville','Raleigh','NC','USA',27560)
    INTO AIRPORT VALUES( 'CLT' , 'Charlotte Douglas International Airport',
    '5501 R C Josh Birmingham Pkwy','Charlotte','NC','USA',28208)
    INTO AIRPORT VALUES( 'LAX' , 'Los Angeles International Airport', '1 World
    Way','Los Angels','CA','USA',90045)
```

- **AIRLINE COMPANY:**

```
CREATE TABLE AIRLINE_COMPANY(
        COMPANY_ID                  VARCHAR(50) ,
        COMPANY_NAME        VARCHAR(100)    NOT NULL,
        CONSTRAINT COMPANY_PK PRIMARY KEY (COMPANY_ID)
);
```

```
INSERT ALL

    INTO AIRLINE_COMPANY VALUES( 'AA' , 'American Airlines')
    INTO AIRLINE_COMPANY VALUES( 'BA' , 'British Airways')
    INTO AIRLINE_COMPANY VALUES( 'EX' , 'Atlantic Airlines')
    INTO AIRLINE_COMPANY VALUES( 'JET' , 'Jet Airways')
    INTO AIRLINE_COMPANY VALUES( 'SJ' , 'Spice Jet')
```

- **AIRPLANE:**

```
CREATE TABLE AIRPLANE(
    AIRPLANE_NO             VARCHAR(25) ,
    SEATING_CAPACITY        NUMBER          NOT NULL,
    COMPANY                 VARCHAR(50)     NOT NULL,
    CONSTRAINT AIRPLANE_PK PRIMARY KEY (AIRPLANE_NO),
    CONSTRAINT PLANE_COMPANY_FK FOREIGN KEY (COMPANY)
    REFERENCES AIRLINE_COMPANY(COMPANY_ID)

);

INSERT ALL

    INTO AIRPLANE VALUES( 'AA751' , 40, 'AA' )
    INTO AIRPLANE VALUES( 'AA851' , 60, 'AA' )
    INTO AIRPLANE VALUES( 'EX100' , 40, 'EX' )
    INTO AIRPLANE VALUES( 'JET785' , 60, 'JET' )
    INTO AIRPLANE VALUES( 'BA747' , 40, 'BA' )
```

- **PLANE PORT ACCESS**:

```
CREATE TABLE PLANE_PORT_ACCESS(
    AIRPORT                 VARCHAR(25)     NOT NULL,
    AIRPLANE                VARCHAR(25)     NOT NULL,
    CONSTRAINT ACCESS_PK PRIMARY KEY (AIRPORT,AIRPLANE),
    CONSTRAINT ACCESS_PORT_FK FOREIGN KEY (AIRPORT)
    REFERENCES AIRPORT(AIRPORT_CODE)
    ON DELETE CASCADE,
    CONSTRAINT ACCESS_PLANE_FK FOREIGN KEY (AIRPLANE)
    REFERENCES AIRPLANE(AIRPLANE_NO)
    ON DELETE CASCADE
);
```

```
INSERT ALL

        INTO PLANE_PORT_ACCESS VALUES( 'DFW','AA751'  )
        INTO PLANE_PORT_ACCESS VALUES( 'RDU','AA751' )
        INTO PLANE_PORT_ACCESS VALUES( 'CLT','AA751'  )
        INTO PLANE_PORT_ACCESS VALUES( 'DFW','AA851'  )
        INTO PLANE_PORT_ACCESS VALUES( 'CLT','AA851' )
```

- **SEAT:**

```
CREATE TABLE SEAT(
        AIRPLANE            VARCHAR(25)     NOT NULL,
        SEAT_NO             VARCHAR(5)      NOT NULL,
        AVAILABILITY        VARCHAR(10)     DEFAULT 'TRUE',
        LOCATION            VARCHAR(25)     NOT NULL,
        SEAT_CLASS          VARCHAR(25)     NOT NULL,
        CONSTRAINT SEAT_PK PRIMARY KEY (AIRPLANE,SEAT_NO),
        CONSTRAINT SEAT_PLANE_FK FOREIGN KEY (AIRPLANE)
        REFERENCES AIRPLANE(AIRPLANE_NO)
        ON DELETE CASCADE
);

INSERT ALL

  INTO SEAT VALUES( 'AA751','A23','TRUE','WINDOW','ECONOMIC' )
  INTO SEAT VALUES( 'AA751' ,'A24','TRUE','WINDOW','ECONOMIC' )
  INTO SEAT VALUES( 'AA751' ,'A25','TRUE','WINDOW','ECONOMIC'  )
  INTO SEAT VALUES( 'AA851' ,'A23','TRUE','WINDOW','ECONOMIC'  )
  INTO SEAT VALUES( 'AA851' ,'A24','TRUE','WINDOW','ECONOMIC' )
```

- **END USER:**

```
CREATE TABLE END_USER (
        EMAIL                       VARCHAR(25) ,
        FNAME                       VARCHAR(25) NOT NULL,
        LNAME                       VARCHAR(25) NOT NULL,
        PHONE                       NUMBER ,
        TYPE_OF_USER                VARCHAR(25) NOT NULL,
        CONSTRAINT USER_PK PRIMARY KEY (EMAIL)
);
```

```
INSERT ALL

    INTO END_USER VALUES
    ('kunal.jagdish@gmail.com','Kunal','Arora',4694529484,'NORMAL' )
    INTO END_USER VALUES
    ('kxa142230@utdallas.edu','Kunal','Arora',4694529484,'NORMAL' )
    INTO END_USER VALUES
    ('shobhit@gmail.com','Shobhit','Agrawal',4694527474,'NORMAL' )
    INTO END_USER VALUES
    ('indervirsingh@gmail.com' ,'Indervir','Singh',4694528585,'NORMAL' )
     INTO END_USER VALUES
    ( 'bala@yahoo.com' ,'Bala','Yadav',4694529595,'AGENT' )
```

- **FLIGHT_TRIP:**

```
CREATE TABLE FLIGHT_TRIP(
        FLIGHT_TRIP_ID          VARCHAR(15) ,
        NO_OF_TRAVELLERS        NUMBER          NOT NULL,
        ARRIVAL_AIRPORT         VARCHAR(25)     NOT NULL,
        ARRIVAL_TIME            DATE            NOT NULL,
        DEPART_AIRPORT          VARCHAR(25)     NOT NULL,
        DEPART_TIME             DATE            NOT NULL,
        USER_EMAIL              VARCHAR(25)     NOT NULL,
        CONSTRAINT FLIGHT_TRIP_PK PRIMARY KEY
        (FLIGHT_TRIP_ID),
        CONSTRAINT FLGHT_TRIP_USER_FK FOREIGN KEY
        (USER_EMAIL) REFERENCES END_USER(EMAIL)
        ON DELETE SET NULL
);

INSERT ALL

    INTO FLIGHT_TRIP
    VALUES('kunish17dec',1,'RDU',to_date('2014/12/27:12:00:00AM',
    'yyyy/mm/dd:hh:mi:ssam'),'DFW',to_date('2014/12/26:12:00:00AM',
    'yyyy/mm/dd:hh:mi:ssam'),'kunal.jagdish@gmail.com' )
```

```
INTO FLIGHT_TRIP
VALUES('kuniaa741',1,'RDU',to_date('2014/12/27:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam'),'DAL',to_date('2014/12/26:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam'),'kunal.jagdish@gmail.com' )

INTO FLIGHT_TRIP
VALUES('inder17dec',1,'RDU',to_date('2014/12/27:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam') ,'DFW',to_date('2014/12/26:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam'),'indervirsingh@gmail.com' )

INTO FLIGHT_TRIP
VALUES('shoaga17dec',1,'RDU',to_date('2014/12/27:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam'),'DFW',to_date('2014/12/26:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam'),'shobhit@gmail.com' )

INTO FLIGHT_TRIP

VALUES('bala17dec',1,'DFW',to_date('2014/12/27:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam')'LAX',to_date('2014/12/26:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam'),'bala@yahoo.com' )
```

- **FARE:**

```
CREATE TABLE FARE(
        FLIGHT_TRIP_ID          VARCHAR(15) ,
        FINAL_AMOUNT            DECIMAL(9,2)    NOT NULL,
        AMOUNT                  DECIMAL(9,2)    NOT NULL,
        CURRENCY                VARCHAR(5)      NOT NULL,
        DISCOUNT                DECIMAL(4,2)    NOT NULL,
        TAX                     DECIMAL(5,2)    NOT NULL,
        CONSTRAINT FARE_PK PRIMARY KEY (FLIGHT_TRIP_ID),
        CONSTRAINT FARE_FLT_TRIP_FK FOREIGN KEY
        (FLIGHT_TRIP_ID) REFERENCES FLIGHT_TRIP(FLIGHT_TRIP_ID)
        ON DELETE CASCADE
);

INSERT ALL

 INTO FARE VALUES( 'kunish17dec',176,200,'$',10,10 )
 INTO FARE VALUES( 'kuniaa741',176,200,'$',10,10 )
 INTO FARE VALUES( 'inder17dec',176,200,'$',10,10 )
 INTO FARE VALUES( 'shoaga17dec',176,200,'$',10,10 )
```

```
INTO FARE VALUES( 'bala17dec',176,200,'$',10,10 )
```

- **TRAVELLER:**

```
CREATE SEQUENCE TRAVELLER_ID_INCREMENT
        MINVALUE 1
        START WITH 1
        INCREMENT BY 1
        CACHE 10;


CREATE TABLE TRAVELLER(
        ID              NUMBER,
        PHONE           INTEGER,
        FNAME           VARCHAR(25) NOT NULL,
        LNAME           VARCHAR(25) NOT NULL,
        CONSTRAINT TRAVELLER_PK PRIMARY KEY (ID)
);


INSERT INTO TRAVELLER
        VALUES(traveller_id_increment.nextVal,4567891231,'Kunal','Arora' );
INSERT INTO TRAVELLER
        VALUES( traveller_id_increment.nextval,4567891241,'Indervir','Singh' );
INSERT INTO TRAVELLER
        VALUES( traveller_id_increment.nextVal,4567891274,'Bala','Yadav' );
INSERT INTO TRAVELLER
        VALUES(traveller_id_increment.nextVal,4567891278,'Shobhi','Agrawal);
INSERT INTO TRAVELLER
        VALUES( traveller_id_increment.nextVal,4567891214,'Nitish','Salwan' );
```

- **TRAVELLER_ITINERARY:**

```
CREATE TABLE TRAVELLER_ITINERARY(
        FLIGHT                  VARCHAR(15)     NOT NULL,
        ID                      VARCHAR(15),
        SEAT_NO                 VARCHAR(5),
        AIRPLANE                VARCHAR(25),
        CONSTRAINT TRAVELLER_PKK PRIMARY KEY
        (FLIGHT,SEAT_NO,AIRPLANE),
        CONSTRAINT TRVLR_FLT_TRIP_FK FOREIGN KEY (FLIGHT)
        REFERENCES FLIGHT_TRIP(FLIGHT_TRIP_ID)
        ON DELETE CASCADE,
        CONSTRAINT TRVLR_SEAT_FK FOREIGN
        KEY(SEAT_NO,AIRPLANE)
        REFERENCES SEAT(SEAT_NO,AIRPLANE)
);
```

```
INSERT ALL

    INTO TRAVELLER_ITINERARY VALUES( 'kunish17dec',2,'A23','AA751' )
    INTO TRAVELLER_ITINERARY VALUES( 'kuniaa741',6,'A24','AA751' )
    INTO TRAVELLER_ITINERARY VALUES( 'inder17dec',3,'A25','AA751' )
    INTO TRAVELLER_ITINERARY VALUES( 'shoaga17dec',4,'A23','AA851' )
    INTO TRAVELLER_ITINERARY VALUES( 'bala17dec',5,'A24','AA851' )
```

- **HOP:**

```
CREATE TABLE HOP(
        HOP_ID                  VARCHAR(15) ,
        DISTANCE                DECIMAL(7,2)    NOT NULL,
        BASE_FARE               DECIMAL(9,2)    NOT NULL,
        AIRPLANE                VARCHAR(25)     NOT NULL,
        ARRIVAL_AIRPORT         VARCHAR(25)     NOT NULL,
        ARRIVAL_TIME            DATE            NOT NULL,
        DEPART_AIRPORT          VARCHAR(25)     NOT NULL,
        DEPART_TIME             DATE            NOT NULL,
        CONSTRAINT HOP_PK PRIMARY KEY (HOP_ID),
        CONSTRAINT HOP_APLNE_FK FOREIGN KEY (AIRPLANE)
        REFERENCES AIRPLANE(AIRPLANE_NO),
        CONSTRAINT HOP_ARVL_ARPRT_FK
        FOREIGN KEY(ARRIVAL_AIRPORT)REFERENCES
        AIRPORT(AIRPORT_CODE),
        CONSTRAINT HOP_DRPT_ARPRT_FK FOREIGN
        KEY(DEPART_AIRPORT) REFERENCES
        AIRPORT(AIRPORT_CODE)
);
```

```
INSERT ALL

        INTO HOP VALUES
        ('AX7458',600,200,'AA751','RDU',to_date('2014/12/27:12:00:00AM','yyyy/mm/d
        d:hh:mi:ssam'),'DFW',to_date('2014/12/26:12:00:00AM',
        'yyyy/mm/dd:hh:mi:ssam') )

        INTO HOP VALUES
        ( 'AX7459',600,200,'AA851','RDU',to_date('2014/12/27:12:00:00AM',
        'yyyy/mm/dd:hh:mi:ssam'),'DAL',to_date('2014/12/26:12:00:00AM',
        'yyyy/mm/dd:hh:mi:ssam') )
        INTO HOP VALUES
        ( 'AX7460',600,200,'AA751','RDU',to_date('2014/12/27:12:00:00AM',
        'yyyy/mm/dd:hh:mi:ssam'),'LAX',to_date('2014/12/26:12:00:00AM',
        'yyyy/mm/dd:hh:mi:ssam') )
        INTO HOP VALUES
```

```
( 'AX7461',600,200,'AA751','RDU',to_date('2014/12/27:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam'),'CLT',to_date('2014/12/26:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam') )
INTO HOP VALUES
( 'AX7462',600,200,'AA751','DFW',to_date('2014/12/27:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam'),'CLT',to_date('2014/12/26:12:00:00AM',
'yyyy/mm/dd:hh:mi:ssam') )
```

- **HAS:**

```
CREATE TABLE HAS(
        HOP                     VARCHAR(25)     NOT NULL,
        FLIGHT_TRIP             VARCHAR(15)     NOT NULL,
        CONSTRAINT HP_HAS_FT_PK PRIMARY KEY
        (HOP,FLIGHT_TRIP),
        CONSTRAINT HOP_HAS_FK FOREIGN KEY (HOP) REFERENCES
        HOP(HOP_ID),
        CONSTRAINT FT_HAS_FK FOREIGN KEY(FLIGHT_TRIP)
        REFERENCES FLIGHT_TRIP(FLIGHT_TRIP_ID)
        ON DELETE CASCADE
);
```

```
INSERT ALL

    INTO HAS VALUES( 'AX7458','kunish17dec' )
    INTO HAS VALUES( 'AX7459','kuniaa741' )
    INTO HAS VALUES( 'AX7458','inder17dec' )
    INTO HAS VALUES( 'AX7458','shoaga17dec' )
    INTO HAS VALUES( 'AX7460','bala17dec' )
```

8.2 SQL Example Queries:

According to the requirements mentioned in the project specification, we have compiled some of the SQL queries using some of the basic operations.

- Simple SQL Select query:

    - SELECT FNAME, LNAME FROM TRAVELLER WHERE ID IN (SELECT ID FROM TRAVELLER_ITINERARY WHERE FLIGHT='kunish17dec');

- Example of GROUP BY & aggregate function:

    - SELECT USER_EMAIL, COUNT (NO_OF_TRAVELLERS) FROM FLIGHT_TRIP WHERE USER_EMAIL IN (SELECT EMAIL FROM END_USER WHERE TYPE_OF_USER ='NORMAL') GROUP BY USER_EMAIL;

- Example of JOIN:

    - SELECT * FROM FLIGHT_TRIP INNER JOIN FARE ON FLIGHT_TRIP.FLIGHT_TRIP_ID=FARE.FLIGHT_TRIP_ID;

- Example of ALTER table:

    - ALTER TABLE FARE ADD (CHECKING VARCHAR(25), CONSTRAINT FK FOREIGN KEY(CHECKING) REFERENCES AIRPORT(AIRPORT_CODE));

- Example of Dropping constraints and columns:

    - ALTER TABLE FARE DROP CONSTRAINT FK;
    - ALTER TABLE FARE DROP COLUMN CHECKING;

- Example of UNION operation:

    - SELECT FNAME, LNAME FROM TRAVELLER WHERE ID IN(SELECT ID FROM TRAVELLER_ITINERARY WHERE FLIGHT='kunish17dec')

  UNION

  SELECT FNAME, LNAME FROM TRAVELLER WHERE ID IN(SELECT ID FROM TRAVELLER_ITINERARY WHERE FLIGHT='inder17dec');

- Example of DROP table command:
    - DROP table fare;

All these queries work absolutely fine on the database that we have designed and at no stage, the integrity of the schema and the database is violated. The database schema needs to be consistent at all stages and these queries confirm that the database we define is maintaining the condition at all level.

We have used ON DELETE CASCADE conditions on the tables PLANE_PORT_ACCESS, SEAT, FARE, TRAVELLER_ITINERARY, HAS. This condition makes sure that whenever any attribute is deleted from these tables, the database will delete all the values that are linked to these attributes in these tables. This makes sure that the state of the database is consistent with our requirements and the schema is valid in all conditions.

9.REFERENCES

- en.wikipedia.org/wiki/Airline_reservations_system#Major_Systems
- http://www.sabreairlinesolutions.com/images/uploads/Hybrid_Model_Brochure.pdf
- aa.com – American Airlines
- www.britishairways.com