Chapter 1

INTRODUCTION

1.1 Background

A database is an organized collection of data, generally stored and accessed electronically from a computer system. Where databases are more complex they are often developed using formal design and modeling techniques.

The database management system (DBMS) is the software that interacts with end users, applications, the database itself to capture and analyze the data and provides facilities to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a "database system". Often the term "database" is also used to loosely refer to any of the DBMS, the database system or an application associated with the database. The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified and the database schema, which defines the database's logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform administration procedures. Typical database administration tasks supported by the DBMS include change management, performance monitoring/tuning and backup and recovery. Many database management systems are also responsible for automated rollbacks, restarts and recovery as well as the logging and auditing of activity.

1.2 Introduction to Airline Reservation System

An airline reservation system (ARS) is part of the so-called passenger service systems (PSS), which are applications supporting the direct contact with the passenger.

ARS eventually evolved into the computer reservations system (CRS). A computer reservation system is used for the reservations of a particular airline and interfaces with a global distribution system (GDS) which supports travel agencies and other distribution channels in making reservations for most major airlines in a single system.

Today all persons are busy with their schedule and no one have time to make a trip for holidays with their family. And this Airline Reservation Process is very difficult to understand in General meaning. But we are providing a Solution for that Problem. This system provides a facility to easy access towards a customers and a real time users. They

can easily connected through it and just 3 steps. There is no requirement for any type of Agent.

The Airline Reservation System project is an implementation of a general Airline Ticketing website like Goibibo, which helps the customers to search the availability and prices of various airline tickets, along with the different packages available with the reservations. This project also covers various features like online registration of the users, modifying the details of the website by the management staff or administrator of the website, by adding, deleting or modifying the customer details, flights or packages information.

In general, this website would be designed to perform like any other airline ticketing website available online. It demonstrates the use of payment gateways to accept payment and shows the implementation of transaction management if multiple transactions are made at the same time. It also demonstrates the use of AJAX calls for dynamic control over webpages. This project also implements the use of PHPMailer which is used to automatically send emails of their ticket confirmation and boarding passes. All the data for this project is collected using the Flight Data API which results in a very realistic experience when using the website.

Chapter 2

E R DIAGRAM AND RELATIONAL SCHEMA DIAGRAM

2.1 Description of ER Diagram

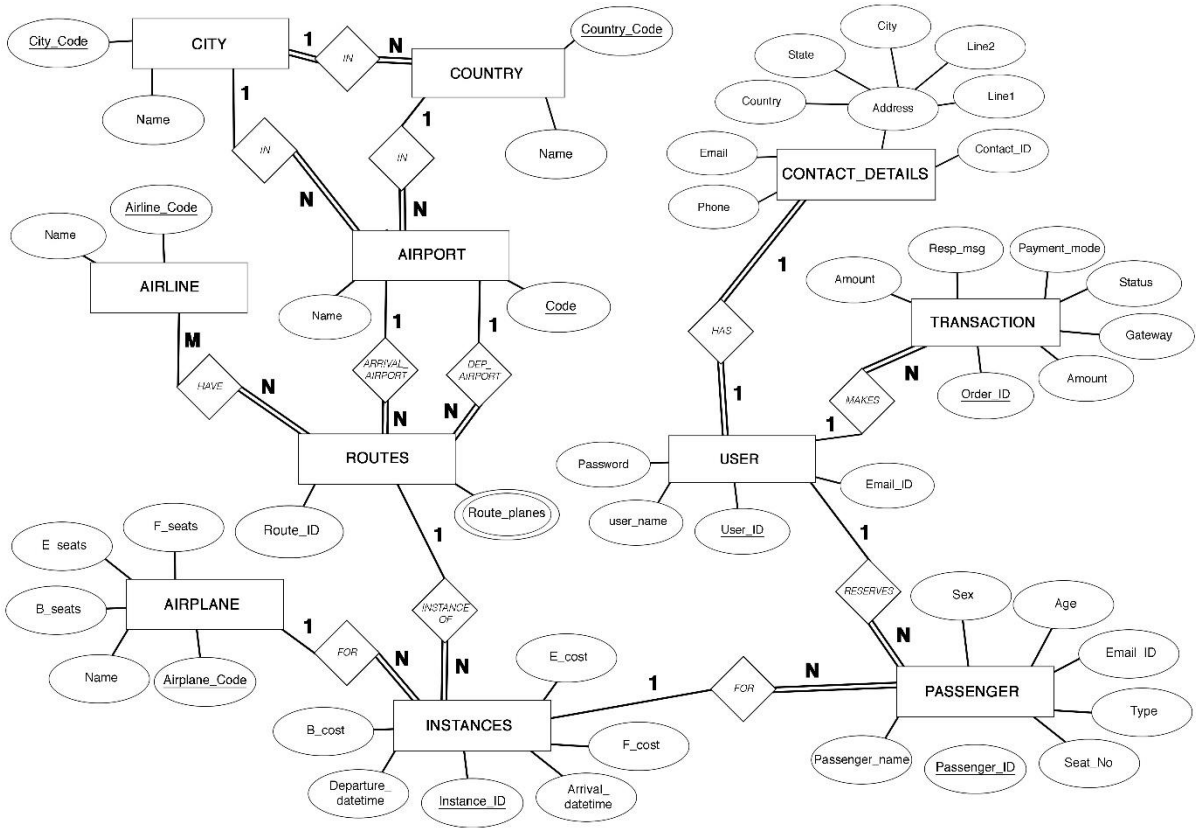

Figure 2.1: E-R Diagram for Airline Reservation System

Entity relationship diagram displays the relationships of entity set stored in a database. In other words, we can say that ER diagrams help you to explain the logical structure of databases. At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique.

2.1.1 Components of Airline Reservation System, E-R Diagram

1. Entity types like **ROUTES** and **PASSENGER** are in rectangular boxes.

2. Relationships like **ARRIVAL_AIRPORT** and **HAS** are in diamond boxes, attached to entity types with straight lines.
3. Attributes are shown in ovals like **Name** and **Seat_No**, each attached by a straight line to entity or relationship type.
4. Key attributes like **Passenger_ID** and **Airport_code** are underlined.
5. Component attributes of a composite attribute like **address** are attached to oval representing it.
6. Multivalued attributes like in **Route_planes,** are represented using double layered oval.

2.1.2 E-R Diagram Relationships Description

1. **CITY:AIRPORT** is of cardinality 1:N as one city can possibly have more than one airport and therefore connected by an **IN** relationship. There is total participation from AIRPORT as all airports are located in some city, but partial participation from CITY as a city may exist without an airport.
2. **COUNTRY:AIRPORT** is of cardinality 1:N as one country can have more than one airport and therefore connected by an **IN** relationship. There is total participation from AIRPORT as all airports are location is some country, but partial participation from COUNTRY as a country may exist without an airport.
3. **AIRPORT:ROUTES** is of cardinality 1:N in both the relationships **ARRIVAL_AIRPORT** and **DEPARTURE_AIRPORT** as one airport can be involved in multiple routes, but one route can have only one departure and arrival airport. There is total participation from routes as all routes are with respect to some airport, and partial participation from AIRPORT as an airport can exist without being assigned to a route.
4. **AIRLINE:ROUTES** is of cardinality M:N as one route can have multiple airlines assigned to it and one airline can service multiple routes and therefore connected by **HAVE** relationship. There is total participation from ROUTES as all routes must be serviced by some airline, but partial participation from AIRLINE as an airline may or may not service a route.
5. **ROUTES:INSTANCES** is of cardinality of 1:N as one route can have multiple instances depending on the departure date and time and therefore connected by an **INSTANCE_OF** relationship. There is total participation from INSTANCES as all

instances are derived from a route, but partial participation from ROUTES as a route may not have an instance.

6. **AIRPLANE:INSTANCES** is of cardinality 1:N as one airplane can be assigned to multiple instances, but one instance is assigned only one airplane and therefore connected by **FOR** relationship. There is total participation from INSTANCES as all instances are assigned some airplane, but partial participation from AIRPLANE as one airplane may not service any flight instance.
7. **CONTACT_DETAILS:USER** is of cardinality 1:1 as one user has only one contact detail and one contact details in for one user and is therefore connected by **HAS** relationship. There is total participation from both CONTACT_DETAILS and USER as all contact details are assigned to some user and all users have contact details.
8. **USER:TRANSACTION** is of cardinality 1:N as one user can make many transactions and one transaction is made by only one user and therefore connected by **MAKES** relationship. There is total participation from TRANSACTION as all transactions are made by some user, but partial participation from USER as may not make a transaction.
9. **USER:PASSENGER** is of cardinality 1:N as one user can book many passenger tickets and one passenger ticket is booked by only one user, and therefore connected by **RESERVES** relationship. There is a total participation from PASSENGER as all passenger tickets are booked by some user, but partial participation from USER as one user may or may not reserve a seat.
10. **INSTANCES:PASSENGER** is of cardinality 1:N as one instance can have multiple passengers and one passenger is for only one instance and therefore connected by **FOR** relationship. There is a total participation from PASSENGER as all passengers belong to some instance, but partial participation from INSTANCE as few instances may or may not have passengers assigned to them.

2.2 Description of Relational Schema Diagram

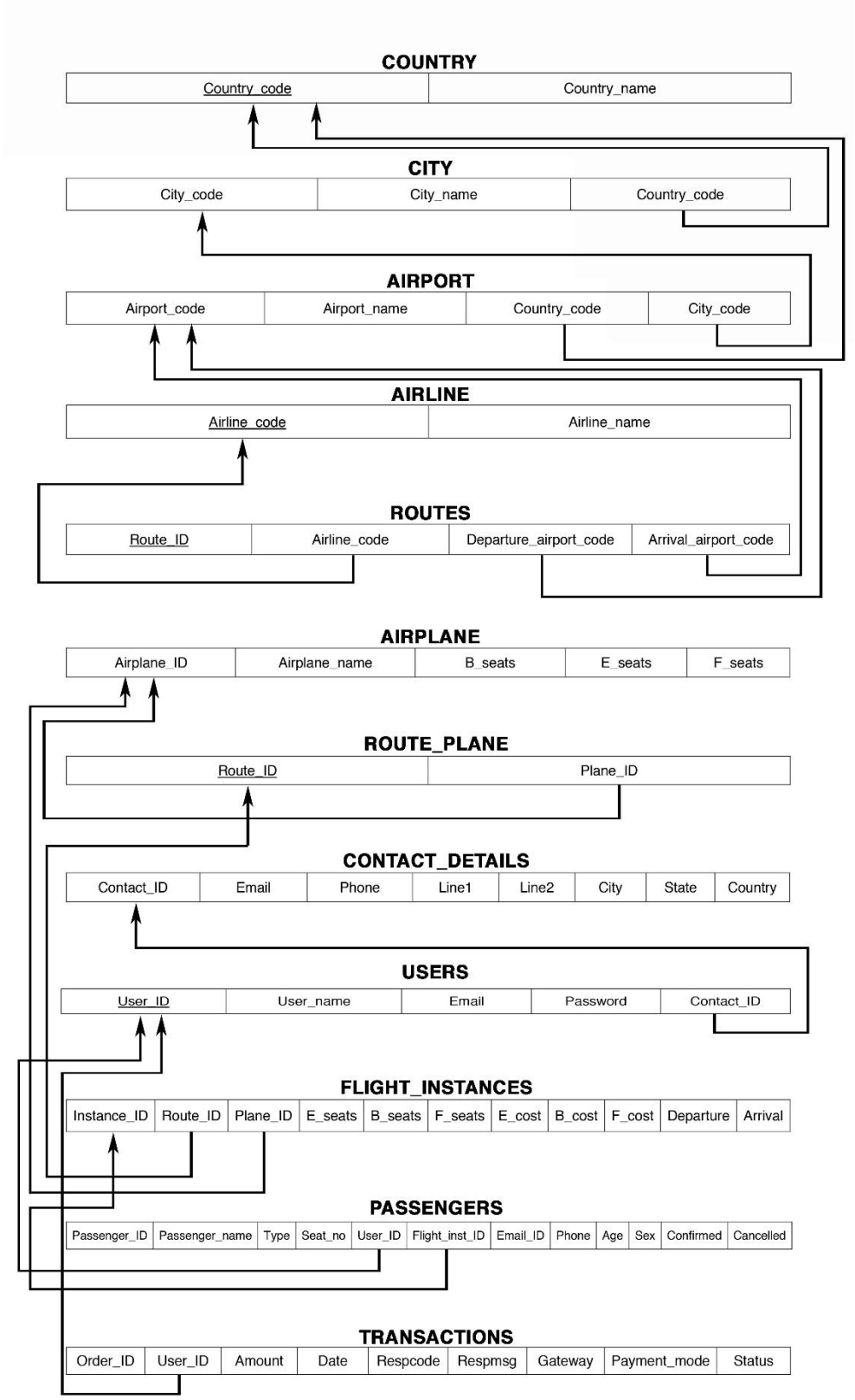

Figure 2.2 Relational Schema Diagram for Airline Reservation System

2.2.1 General Constraints

1. **NULL Constraint**: Attributes that are under NOT NULL constraints have to be filled compulsorily. Almost all the attributes in the project are under NOT NULL constraint.
2. **Entity Integrity Constraint**: This constraint makes sure that no primary key can have a NULL value assigned to it. The primary keys involved in the project include:
    - Code
    - Flight_inst_ID
    - Passenger_ID
3. **Referential Integrity Constraints**: A table in the back end of the project may have references pointing to an attribute in another table. For example: FLIGHT_INST_ID in the PASSENGER table refers to INSTANCE_ID in FLIGHT_INSTANCES table. The various tables are also linked with multiple foreign keys which are all set to cascade any update or delete operation on the attribute in the main table. The various Foreign Key attributes are:
    - Flight_inst_ID
    - Plane_ID
    - City_code
    - Country_code

2.2.2 Schema Description

With reference to the Figure.2.2 shows the relational schema of Airline Reservation System. It has the following entities.

1. **COUNTRY:** This table contains the details like IATA code, name. IATA code is the primary key.
2. **CITY:** This table contains the names of all the cities in the world. It has 3 columns, IATA city code, name, Country code, referring to entry in previous table.
3. **AIRPORT:** This table consists the list of airports passengers can board and arrive at. It has 4 columns IATA airport code, name, country code and city code.
4. **AIRLINE:** This table consists of all the airlines and has IATA airline code and name as attributes.
5. **ROUTES:** This table consists of list of all the routes serviced by each airline registered with the system. It has the following columns, Route ID, airline code,

departure airport code and arrival airport code. An assumption is made that, all flights taken into consideration are direct flights from source to destination.

6. **AIRPLANE:** This table contains a list of airplanes that can be used for any routes. It has airplane ID, name, economy seats, business seats and first class seats as its columns.
7. **ROUTE_PLANES:** This table is used to represent the MVD in the ROUTES entity, it has 2 columns route ID and airplane ID.
8. **CONTACT DETAILS:** This table contains the contact details of all the users registered with the reservation platform. It has contact ID, email, phone, line1, line2, city, state and country as attributes.
9. **USERS:** This contains the credentials of all the users registered with the booking system. It has user ID, username, email, password and contact ID as its columns.
10. **FLIGHT INSTANCES:** This is the master table, it contains the list of all routes and its schedules along with the number of seats available in each travel class and the airplane assigned to this instance. It has instance ID, route ID, plane ID, e seats, b seats, f seats, e cost, b cost, f cost, departure and arrival as its attributes.
11. **PASSENGER:** This table contains the list of passengers who have booked for multiple flight instances. It contains, passenger ID, name, type, seat no. , user ID, email ID, phone, age, sex as its attributes.
12. **TRANSACTIONS:** This table contains all the transactions made by the users for any particular bookings. It has the following columns order ID, user ID, amount, date, response code, response message, gateway payment mode and status.

Chapter 3

SYSTEM DESIGN

3.1 Table Description

COUNTRY

Table 3.1 Country table

FIELD	TYPE	NULL?	KEY	DEFAULT
Code	VARCHAR(10)	NO	PRIMARY	None
Name	VARCHAR(100)	NO		None

The Table 3.1 contains the list of all countries. It has code and name as the attributes where Code is the primary key.

CITY

Table 3.2 City Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Code	VARCHAR(10)	NO	PRIMARY	None
Name	VARCHAR(100)	NO		None
Country_code	VARCHAR(10)	NO	FOREIGN	None

The Table 3.2 contains the list of all cities, which are in a country. It has 3 attributes Code, Name and Country_code. Code is the primary key and Country_code is the foreign key which references code of country table.

AIRPORT

Table 3.3 Airport Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Code	VARCHAR(10)	NO	PRIMARY	None
Name	VARCHAR(100)	NO		None
Country_code	VARCHAR(10)	NO	FOREIGN	None
City_code	VARCHAR(10)	NO	FOREIGN	None

The Table 3.3 contains the list of all airports. It has 4 attributes Code, Name, Country_code and City_code. Code is the primary key, Country_code is a foreign key which references Code of country table and City_code is a foreign key which references Code of City table.

AIRLINES

Table 3.4 Airplanes Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Code	VARCHAR(10)	NO	PRIMARY	None
Name	VARCHAR(100)	NO		None

The Table 3.4 contains a list of all the airlines. It has 2 attributes, Code and Name. Code is the primary key.

ROUTES

Table 3.5 Routes Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Route_ID	INTEGER	NO	PRIMARY	None
Airline_code	VARCHAR(10)	NO	FOREIGN	None
Departure_airport_code	VARCHAR(10)	NO	FOREIGN	None
Arrival_airport_code	VARCHAR(10)	NO	FOREIGN	None

The Table 3.5 contains the list of all routes serviced by different airlines from multiple origin and destination airports. It has 4 attributes Route_ID, Airline_code, Departure_airport_code, Arrival_airport_code. Route_ID is the primary key. Departure_airport_code and Arrival_airport_code and foreign keys which reference code of airport table and Airline_code references Code of airline table.

AIRPLANE

Table 3.6 Airplane Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Code	VARCHAR(10)	NO	PRIMARY	None
Name	VARCHAR(100)	NO		None
Eseats	INTEGER	YES		0

Bseats	INTEGER	YES		0
Fseats	INTEGER	YES		0

The Table 3.6 contains the list of all airplanes. It has 5 attributes Code, Name, Fseats, Eseats and Bseats. Code is the primary key.

ROUTE_PLANES

Table 3.7 Route_planes Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Route_ID	INTEGER	NO	PRIMARY FOREIGN	None
Plane_ID	VARCHAR(10)	NO	PRIMARY FOREIGN	None

The Table 3.7 is a table derived from multivalued attribute. It has 2 attributes Route_ID and Plane_ID. Route_ID and Plane_ID are primary keys. Plane_ID is the foreign key which references Plane_ID from Airplane table and Route_ID which references Route_ID of routes table.

INSTANCES

Table 3.8 Instances Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Instance_ID	BIG INTEGER	NO	PRIMARY	None
Route_ID	INTEGER	NO	FOREIGN	None
Plane_ID	VARCHAR(10)	NO	FOREIGN	None
Eseats	INTEGER	NO		None
Bseats	INTEGER	NO		None
Fseats	INTEGER	NO		None
ecost	INTEGER	NO		None
bcost	INTEGER	NO		None
fcost	INTEGER	NO		None
departure	DATETIME	NO		None
arrival	DATETIME	NO		None

The Table 3.8 contains the list of all flight instances. It has details about the departure and arrival date and time, the cost of seats etc. Instance_ID is the primary key. Route_ID is the foreign key which references Route_ID of Routes table and Plane_ID references Plane_ID from the airplane table.

CONTACT_DETAILS

Table 3.9 Contact details Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Contact_ID	INTEGER	NO	PRIMARY	None
Email	VARCHAR(40)	NO		None
Phone	VARCHAR(10)	NO		None
Line1	VARCHAR(40)	NO		None
Line2	VARCHAR(40)	NO		None
City	VARCHAR(40)	NO		None
State	VARCHAR(30)	NO		None
Country	VARCHAR(30)	NO		None

The Table 3.9 contains the list of all contact details of the users. It has a total of 8 attributes, Contact_ID, Email, Phone, Line1, Line2, City, State and Country. Contact_ID is the primary key.

USERS

Table 3.10 Users Table

FIELD	TYPE	NULL?	KEY	DEFAULT
User_ID	INTEGER	NO	PRIMARY	None
U_name	VARCHAR(50)	NO		None
Email	VARCHAR(40)	NO		None
Pass	VARCHAR(200)	NO		None
Contact_det	INTEGER	NO	FOREIGN	None

The Table 3.9 contains list of all users also with their login credentials. It has a 5 attributes, User_ID, U_name, Email, Pass and Conact_det. User_ID is the primary key. Contact_det is the foreign key which references Contact_ID of contact details table.

PASSENGER

Table 3.11 Passenger Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Passenger_ID	INTEGER	NO	PRIMARY	None
Passenger_name	VARCHAR(40)	NO		None
Type	VARCHAR(40)	NO		None
Seat_no	VARCHAR(5)	YES		None
User_ID	INTEGER	NO	FOREIGN	None
Flight_inst_ID	INTEGER	NO	FOREIGN	None
Email_ID	VARCHAR(40)	NO		None
Phone	VARCHAR(10)	NO		None
Age	INTEGER	NO		None
Sex	VARCHAR(10)	NO		None
confirmed	INTEGER	NO		0
cancelled	INTEGER	NO		0

The Table 3.11 contains the list of all the passengers for different flight instances. It has details about the type of seat, booking user_id and passenger details. It has 12 attributes. Passenger_ID is the primary key. Flight_inst_ID is the foreign key which references Instance_ID of Instances table and User_ID which references User_ID of users table.

TRANSACTION

Table 3.12 Transaction Table

FIELD	TYPE	NULL?	KEY	DEFAULT
Order_ID	VARCHAR(100)	NO	PRIMARY	None
User_ID	INTEGER	NO	FOREIGN	None
Amount	FLOAT	NO		None
Date	DATETIME	NO		None
Respcode	VARCHAR(100)	NO		None
Respmsg	VARCHAR(100)	NO		None
Gateway	VARCHAR(100)	NO		None

Payment_mode	VARCHAR(100)	NO		None
Status	VARCHAR(100)	NO		None

The Table 3.12 contains the list of all transactions made by the user. It has 10 attributes. Order_ID is the primary key. User_ID is the foreign key which references User_ID of Users table.

3.2 Stored Procedures

One stored is used to get the flight details when the user submits the initial form to view the tickets from source to destination on a specified date for a specified number of travelers and travel class type.

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `flight_data` (IN `origin_code`
VARCHAR(10), IN `destination_code` VARCHAR(10), IN `travellers` INT, IN `dep`
VARCHAR(20), IN `d` DATETIME, IN `type` VARCHAR(20))  NO SQL
BEGIN
  IF (type = 'eseats') THEN
    SELECT *
    FROM instances i, routes r, airplane ap
    WHERE i.Route_ID = r.Route_ID
    AND i.plane_ID = ap.Code
    AND r.departure_airport_code = origin_code
    AND r.arrival_airport_code = destination_code
    AND i.eseats > 0
    AND i.eseats >= travellers
    AND departure LIKE dep
    AND i.departure > d
    ORDER BY i.ecost;
  ELSEIF (type = 'bseats') THEN
    SELECT *
    FROM instances i, routes r, airplane ap
    WHERE i.Route_ID = r.Route_ID
    AND i.plane_ID = ap.Code
    AND r.departure_airport_code = origin_code
    AND r.arrival_airport_code = destination_code
    AND i.bseats > 0
    AND i.bseats >= travellers
```

```
    AND departure LIKE dep
    AND i.departure > d
    ORDER BY i.bcost;
  ELSE
    SELECT *
    FROM instances i, routes r, airplane ap
    WHERE i.Route_ID = r.Route_ID
    AND i.plane_ID = ap.Code
    AND r.departure_airport_code = origin_code
    AND r.arrival_airport_code = destination_code
    AND i.fseats > 0
    AND i.fseats >= travellers
    AND departure LIKE dep
    AND i.departure > d
    ORDER BY i.fcost;
  END IF;
END
```

It takes six parameters origin airport code, destination airport code, number of travelers, date of departure, current date and type of seats being booked.

3.3 Trigger

A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. For example, a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.

In this project a trigger is added to reduce the number of seats left when a passenger is confirmed.

```
CREATE TRIGGER update_inst
AFTER UPDATE
ON Passenger FOR EACH ROW
  BEGIN
    iF (OLD.confirmed = 0) AND (NEW.confirmed = 1) THEN
      IF (OLD.Type = 'Economy') THEN
        UPDATE instances SET eseats = eseats - 1 WHERE Instance_ID =
        NEW.Flight_inst_ID;
```

```
        ELSE IF (OLD.Type = 'Business') THEN
                UPDATE instances SET bseats = bseats - 1 WHERE Instance_ID =
                NEW.Flight_inst_ID;
        ELSE
                UPDATE instances SET fseats = fseats - 1 WHERE Instance_ID =
                NEW.Flight_inst_ID;
        END IF;
    END IF;
END
```

The above trigger is called whenever an update is made on the passenger table, and if the confirmed value is changed from 0 to 1, i.e. the passenger details are confirmed, then decrement the number of seats left in the instances table.