Chapter 1

INTRODUCTION

1.1 Purpose of the Project

The objective of our project is to provide and maintain information about the different airport, airlines and passengers associated with various domains. Our main focus is to design a unique Airport Management System that will improve user's experience. Users will have the felicity to log in from any place with internet connection. After that they will be able to perform different tasks designed for them such as view information about availability of airlines in their airport, airline schedules, flight details, passenger details, tickets, employee details. Also there is provision for users to add new passenger, tickets and update other details. This will make the system useful and convenient for everyone.

1.2 Motivation

The existing system was suffering from serious drawbacks. Since whole of the system was to be maintained with respective department the process of keeping, maintaining and retrieving the information was tedious and lengthy. The records about airlines were never used to be in a systematic order, there used to be lots of difficulties in finding information about doctors associated with different domains. If any information was needed, it was required to go through different websites records of different departments. For this reason, we have provided users with automated system for easier access.

1.3 Existing System

The current system has users going through various websites, records to view information about airlines, passengers and employees. This requires a lot of time from the user side. Currently, there is no system that provides information of all the airport in one place. Using this system, we are eliminating the inconvenience caused to the airport management staff.

1.3 Proposed System

Airport management system primarily deals with management of airport, airlines and passengers. The system provides broad overview of underlying operational factors that influence the airport management.

- The database system has the data of all commercial service airports.
- An airport is located in a city.
- All International airlines operating through various countries across the world have their offices located in all major cities and airports they cover. Hence, an airport may have many airline offices.
- Every airline is identified uniquely by an airline code. Airline code is a two-letter airline designator. Airline also has three-digit code which is printed on an air ticket.
- Every flight is uniquely identified by a flight code. Flight code is a combination of an airline code and number. It contains attributes like current status, source, destination and timings.
- A passenger is uniquely identified by a passenger id and a passport number. Every passenger has details such as name, address, age, sex, and phone.

Chapter 2

PROJECT REQUREMENTS

2.1 Hardware

- Processor: Pentium IV or higher
- RAM**:** 256 MB or higher
- Hard Disk**:** 10 GB or higher
- Cache Memory: 512 KB or higher
- Keyboard: Microsoft Compatible 101 or higher

2.2 Software

- Operating System: Windows
- Front-End (Interface): HTML, PHP, CSS
- Back-End (Database): MySQL
- Web Server: XAMPP
- Web Browser Google Chrome or Mozilla Firefox or Microsoft Edge

Chapter 3

LITERATURE SURVEY

After a lot of research and experience we were able to understand how airports work and the consequences they face in their daily operations. Based on the information gathered we made the database with required attributes. Various domains present in an airport are explained below.

Airline companies serve flights. Every flight is uniquely identified by a flight code. Flight code is a combination of an airline code and four-digit number.

Flight takes off from one airport and lands on another airport. Therefore, most important aspect of a flight is, its source and destination. Source and destination airports are identified using an airport's IATA code. International Air Transport Airport code is simply a location identifier. IATA code is a three-letter code designating many airports across the world. These codes are prominently displayed on baggage tags and printed on an air ticket.

Flight has an arrival time, departure time, duration. Flight has three types of classes business, economy and first class. There can be of two types such non-stop flight and a connecting flight. Connecting flight is a flight which takes intermediate stop and changes a flight possibly change of an airline. But we are assuming that connecting flight does not change a flight that is at each stop, after layover time gets over, passengers aboard the same flight

Flight serves passengers. Flight carries passengers from source to destination. A passenger is uniquely identified by a passenger id and a passport number. Every passenger has details such as name, address, age, sex, and phone. For a passenger to travel by a flight, he needs a ticket.

A ticket or air ticket is used to confirm that an individual has reserved a seat on a flight. With the ticket, a passenger is allowed to board the flight. An air ticket has information such as the passenger's name, the issuing airline, ticket number, source, destination, journey date, seat no, class, fare. Ticket number is the combination of airline's 3-digit code, 4-digit form number and 6-digit serial number.

Hence, depending on airline, source, destination, journey date and most importantly class, which a passenger chooses fare or price of an air ticket is determined. A passenger can book one or multiple tickets. The day on which he books an air ticket is a booking date.

Similarly, a passenger can cancel one or multiple tickets. The day on which he cancels an air ticket is cancellation date and there will be a surcharge that a passenger has to pay after cancelling a ticket.

Every airport has employees working for it. Every employee is identified by SSN. Every employee has an information such name, address, phone, age, sex, salary. Employees in the role of administrative support, engineer, traffic controller and airport authority work at the airport.

Every airline needs administrative support staff to keep the office running smoothly. The different positions include secretaries, data entry workers, receptionists, communications and PR specialists and human resources department. There are different types of engineers who work specifically with information technologies, electronics, flight structure, environmental regulations, etc.

Traffic Monitor works in different shifts such as day or night. There are different positions that airport authorities might work at such as manager, attendee, assistant, pilot, etc. Employees working in the role of administrative support may help passengers with various tasks such as booking a flight ticket, solving passenger's questions, etc.

Chapter 4

LITERATURE SURVEY AND PROBLEM STATEMENT

Through the detailed study of the literature survey we arrived at a conclusion that maintenance is important and integral part of any system and hence maintenance should be made more convenient as the time progresses.

We made the fowling assumptions for the project:

- We are not considering privately managed airports. We are only considering publicly owned airports.
- Several Categories of airports:
  - Commercial Service airport:
    These are publicly owned airports that serve aircrafts which provide scheduled passenger service.
  - Cargo Service airports:
    These airports serve aircrafts carrying cargo only
- The system is designed only for international flights.
- A city has at most one international airport.
- For Connecting flights, flight and airline remains same at layover stops
- There are different types of jobs available at the airport. For simplicity our system considers a few jobs only.

Approach to creating a Airport Management System as a web application

Chapter 5

SYSTEM DESIGN

Design is the first step in the development phase for any technique or principle. It helps in defining a device, process or system. It enables its physical realization.

After the software requirements have been analysed, the software design involves three technical activities - design, coding, implementation and testing.

The design activities are very important because it affects the success of the software implementation. A good system design helps in making the maintenance of system easy. It increases the reliability and maintainability of the system. Design can be used to translate the customer's requirements into finished software.

Through good design, quality of development can be improved. Software design is a process through which requirements can be translated into a software.

5.1 E-R Diagram

The ER model is a conceptual data model that views the real-world objects as entities and defines its relationships. A basic component of this model is the Entity-Relationship diagram. It is used to visually represent data objects. Today it is commonly used for database design by the database designer.

The utility of the ER model is:

- It maps well to the relational model.
- The constructs used in the ER model can easily be transformed into relational tables.
- It is simple and easy to understand with a minimum of training.
- The model can be used by the database designer to communicate the design to the end user.

- The model can be used as a design plan by the database developer to implement a data model in a database management software.

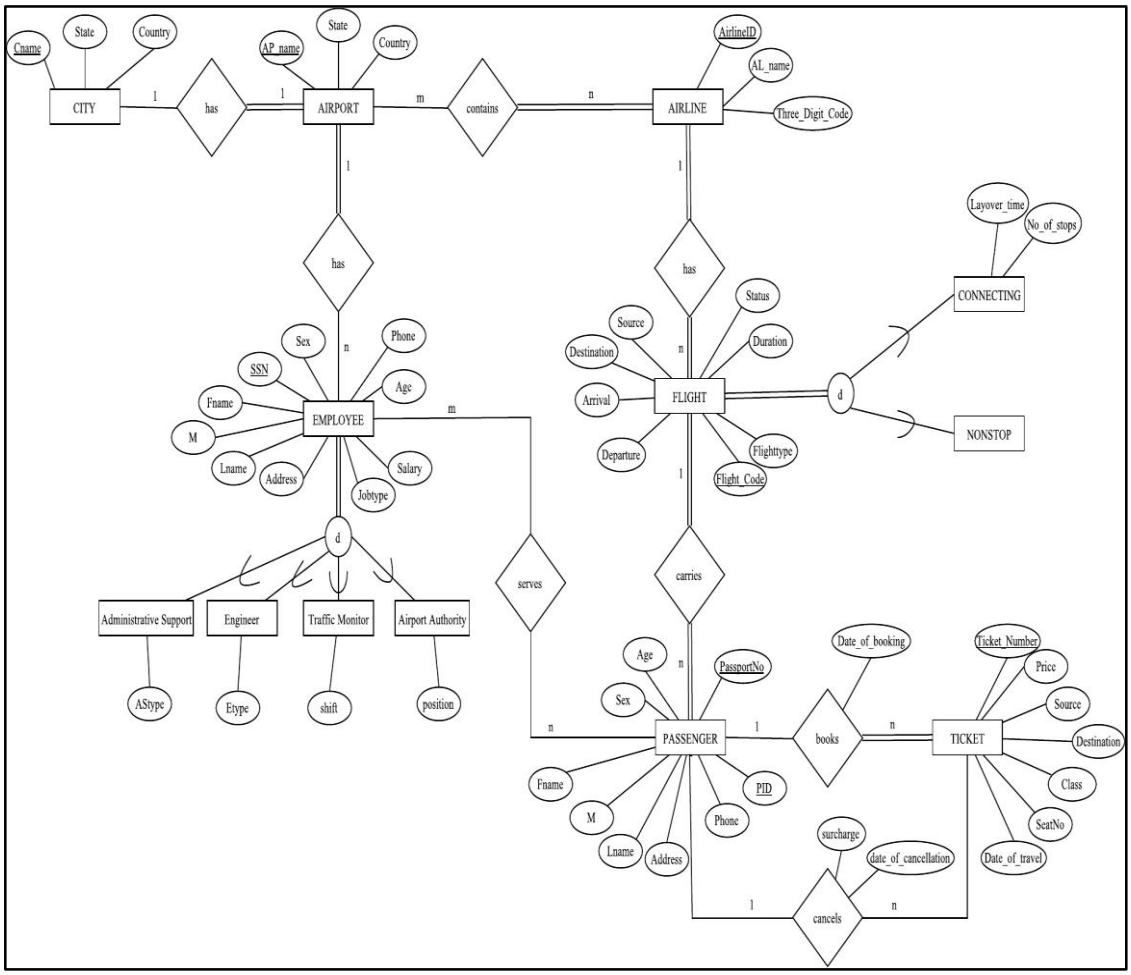


Fig. 5.1 E-R Diagram

5.2 Schema Diagram

A schema contains schema objects, which could be tables, columns, data types, views, stored procedures, relationships, primary keys, foreign keys, etc. A database schema can be represented in a visual diagram, which shows the database objects and their relationship with each other.

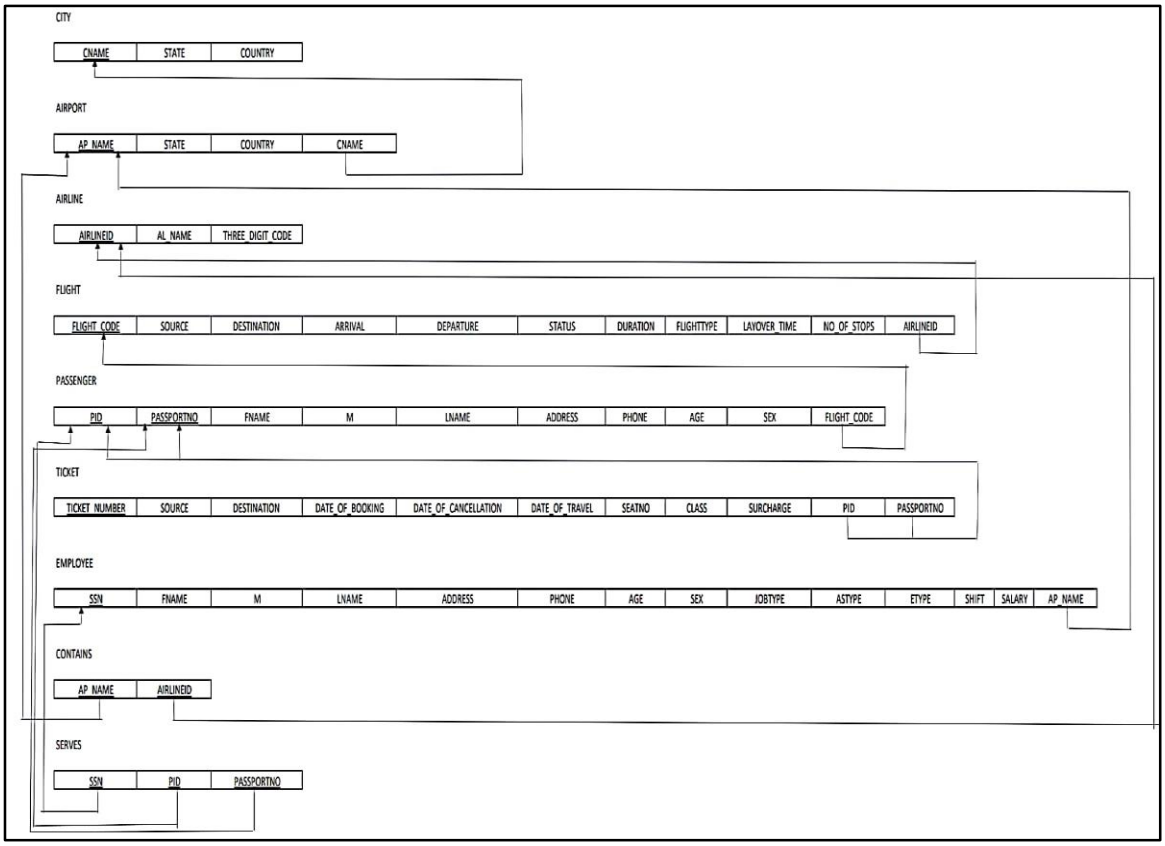


Fig. 5.2 Schema Diagram

5.3 Normalization Theory

Normalization is a database design technique which organizes tables in a manner that reduces redundancy and dependency of data. It divides larger tables to smaller tables and links them using relationships. A relation is said to be in a particular form if it satisfies certain specified constraints.

To decide a suitable logical structure for given database design the concept of normalization, which are briefly described below.

1. **1st Normal Form (1 NF):** A relation is said to be in 1 NF if and only if all unaligned domains contain one value only. That is the fields of an n-set should have no group items and no repeating groups.
2. **2nd Normal Form (2 NF):** A relation is said to be in 2 NF if and only if it is in 1 NF and every non key attribute is fully dependent on primary key. This normal takes care of functional dependencies on non-key attributes.
3. **3rd Normal Form (3 NF):** A relation is said to be in 3 NF if and only if it is in 2 NF and every non key attribute is non transitively dependent on the primary key. This normal form avoids the transitive dependencies on the primary key.
4. **Boyce code Normal Form (BCNF):** This is a stronger definition than that of NF. A relation is said to be in BCNF if and only if every determinant is a Candidate key.
5. **4th Normal Form (4 NF):** A relation is said to be in 4 NF if and only if whenever there exists a multi valued dependency in a relation say A- >>B then all of the relation is also functionally dependent on A (i.e. A- >X for all attributes x of the relation.).
6. **5th Normal Form (5 NF) OR Projection Join Normal Form (PJNF):** A relation R is in 5 NF if and only if every join dependency in R is implied by the candidate key on R . A relation can't be non-loss split into two tables but can be split into three tables. This is called Join Dependency.

All the tables in our database are normalized till 3 NF as mentioned below:

5.3.1 Functional Dependencies

PASSPORTNO -> FNAME, M, LNAME, ADDRESS, PHONE, AGE, SEX Violates 2NF

PID -> FLIGHT_CODE Violates 2NF

DATE_OF_BOOKING, SOURCE, DESTINATION, CLASS -> PRICE Violates 3NF

DATE_OF_CANCELLATION -> SURCHARGE Violates 3NF

JOBTYPE -> SALARY Violates 3NF

5.3.2 Tables after Normalization to 3NF

CITY (CNAME, STATE, COUNTRY)

AIRPORT (AP_NAME, STATE, COUNTRY, CNAME)

AIRLINE (AIRLINEID, AL_NAME, THREE_DIGIT_CODE)

CONTAINS (AIRLINEID, AP_NAME)

FLIGHT (FLIGHT_CODE, SOURCE, DESTINATION, ARRIVAL, DEPARTURE, CUR_STATUS, DURATION, FLIGHTTYPE, LAYOVER_TIME, NO_OF_STOPS, AIRLINEID)

PASSENGER1 (PID, PASSPORTNO)

PASSENGER2(PASSPORTNO, FNAME, M, LNAME, ADDRESS, PHONE, AGE, SEX)

PASSENGER3 (PID, FLIGHT_CODE)

TICKET1 (TICKET_NUMBER, SOURCE, DESTINATION, DATE_OF_BOOKING, DATE_OF_TRAVEL, SEATNO, CLASS, DATE_OF_CANCELLATION, PID, PASSPORTNO)

TICKET2 (DATE_OF_BOOKING, SOURCE, DESTINATION, CLASS, PRICE)

TICKET3 (DATE_OF_CANCELLATION, SURCHARGE)

EMPLOYEE1 (SSN, FNAME, M, LNAME, ADDRESS, PHONE, AGE, SEX, JOBTYPE, ASTYPE, ETYPE, SHIFT, POSITION, AP_NAME)

EMPLOYEE2(JOBTYPE, SALARY)

SERVES (SSN, PID, PASSPORTNO)

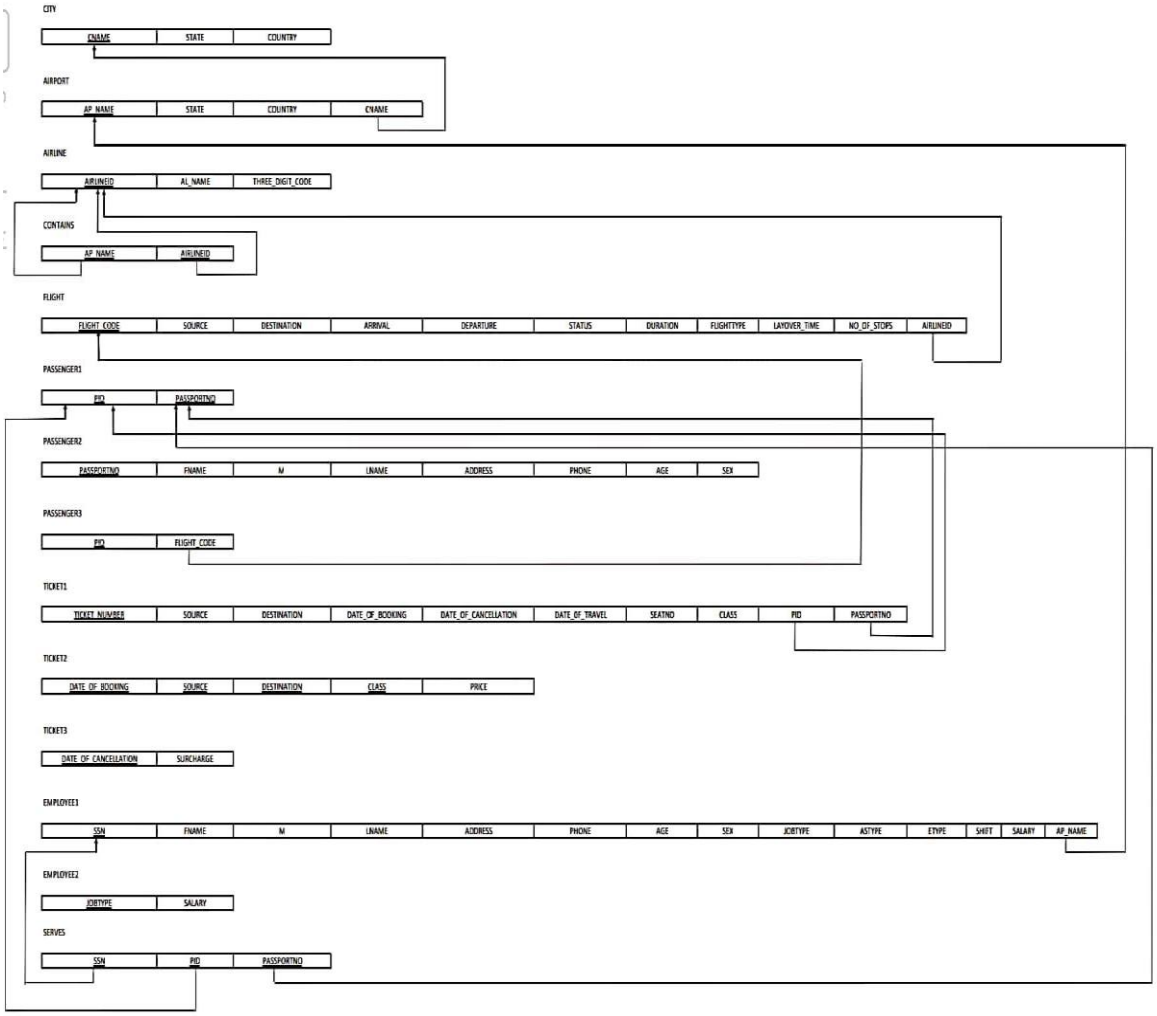


Fig. 5.3 Normalized Schema Diagram

Chapter 6

IMPLEMENTATION

6.1 Introduction

For implementing the system, we need to first set up the apache server. XAMPP is an application server for hosting PHP websites. XAMPP is a cross platform web server stack that comes with built in apache, MySQL, PHP and Perl. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server extremely easy as well. Once this is done, we choose a text editor to our convenience. For this project we have used Visual Studio Code, which is a cross platform text editor for code and mark up. We have developed a web application for this problem statement.

6.2 Front-End Technology

6.2.1 HTML

HTML stands for Hyper Text Markup Language, which is the most widely used language on Web to develop web pages. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.

6.2.2 PHP

PHP: Hypertext Preprocessor is a general-purpose programming language originally designed for web development. It was originally created by Rasmus Lerdorf in 1994 the PHP reference implementation is now produced by The PHP Group.

It is a programming language that allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web based software applications.

6.2.3 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML.[1] CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.[2]

It is a simply designed language intended to simplify the process of making web pages presentable how HTML elements are to be displayed on screen, paper, or in other media**.** CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page.

6.3 Back-End Technology

6.3.1 MySQL

MySQL is the most popular Open Source Relational SQL Database Management System. MySQL is one of the best RDBMS being used for developing various web- based software applications. With its proven performance, reliability, and ease-of-use, MySQL has become the leading database choice for web-based applications, used by high profile web properties including Facebook, Twitter, YouTube, etc.

Some of the key features of SQL:

- Client-Server Architecture
- SQL Compatibility

- Easy to use
- It is scalable
- Allows roll-back
- Compatible on many operating systems
- High Performance

6.3.2 XAMPP

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible.

6.4 Creating Tables

```
CREATE TABLE CITY
(CNAME VARCHAR(15) NOT NULL,
STATE VARCHAR(15),
COUNTRY VARCHAR(30),
PRIMARY KEY(CNAME));

CREATE TABLE AIRPORT
(AP_NAME VARCHAR(100) NOT NULL,
STATE VARCHAR(15),
COUNTRY VARCHAR(30),
CNAME VARCHAR(15),
PRIMARY KEY(AP_NAME),
FOREIGN KEY(CNAME) REFERENCES CITY(CNAME) ON DELETE
CASCADE);
```

```
CREATE TABLE AIRLINE
(AIRLINEID VARCHAR(3) NOT NULL,
AL_NAME VARCHAR(50),
THREE_DIGIT_CODE VARCHAR(3),
PRIMARY KEY(AIRLINEID));

CREATE TABLE CONTAINS
(AIRLINEID VARCHAR(3) NOT NULL,
AP_NAME VARCHAR(100) NOT NULL,
PRIMARY KEY(AIRLINEID, AP_NAME),
FOREIGN KEY(AIRLINEID) REFERENCES AIRLINE(AIRLINEID) ON
DELETE CASCADE,
FOREIGN KEY(AP_NAME) REFERENCES AIRPORT(AP_NAME) ON DELETE
CASCADE);

CREATE TABLE FLIGHT
(FLIGHT_CODE VARCHAR(10) NOT NULL,
SOURCE VARCHAR(3),
DESTINATION VARCHAR(3),
ARRIVAL VARCHAR(10),
DEPARTURE VARCHAR(10),
CUR_STATUS VARCHAR(10),
DURATION VARCHAR(30),
FLIGHTTYPE VARCHAR(10),
LAYOVER_TIME VARCHAR(30),
NO_OF_STOPS INT,
AIRLINEID VARCHAR(3),
PRIMARY KEY(FLIGHT_CODE),
FOREIGN KEY(AIRLINEID) REFERENCES AIRLINE(AIRLINEID) ON
DELETE CASCADE);

CREATE TABLE PASSENGER1
(PID INT NOT NULL,
```

```
PASSPORTNO VARCHAR(10) NOT NULL,
PRIMARY KEY(PID, PASSPORTNO));
CREATE TABLE PASSENGER2
(PASSPORTNO VARCHAR(10) NOT NULL,
FNAME VARCHAR(20),
M VARCHAR(1),
LNAME VARCHAR(20),
ADDRESS VARCHAR(100),
PHONE BIGINT,
AGE INT,
SEX VARCHAR(1),
PRIMARY KEY(PASSPORTNO));

CREATE TABLE PASSENGER3
(PID INT NOT NULL,
FLIGHT_CODE VARCHAR(10),
PRIMARY KEY(PID),
FOREIGN KEY(FLIGHT_CODE) REFERENCES FLIGHT(FLIGHT_CODE) ON
DELETE CASCADE);

CREATE TABLE EMPLOYEE1
(SSN INT NOT NULL,
FNAME VARCHAR(20),
M VARCHAR(1),
LNAME VARCHAR(20),
ADDRESS VARCHAR(100),
PHONE INT,
AGE INT,
SEX VARCHAR(1),
JOBTYPE VARCHAR(30),
ASTYPE VARCHAR(30),
ETYPE VARCHAR(30),
SHIFT VARCHAR(20),
POSITION VARCHAR(30),
```

```
AP_NAME VARCHAR(100),
PRIMARY KEY(SSN),
FOREIGN KEY(AP_NAME) REFERENCES AIRPORT(AP_NAME) ON DELETE
CASCADE);

CREATE TABLE EMPLOYEE2
(JOBTYPE VARCHAR(30) NOT NULL,
SALARY INT,
PRIMARY KEY(JOBTYPE));

CREATE TABLE SERVES
(SSN INT NOT NULL,
PID INT NOT NULL,
PASSPORTNO VARCHAR(10) NOT NULL,
PRIMARY KEY(SSN, PID, PASSPORTNO),
FOREIGN KEY(SSN) REFERENCES EMPLOYEE1(SSN) ON DELETE
CASCADE,
FOREIGN KEY(PID, PASSPORTNO) REFERENCES PASSENGER1(PID,
PASSPORTNO) ON DELETE CASCADE);

CREATE TABLE TICKET1
(TICKET_NUMBER BIGINT(15) NOT NULL,
SOURCE VARCHAR(3),
DESTINATION VARCHAR(3),
DATE_OF_BOOKING DATE,
DATE_OF_TRAVEL DATE,
SEATNO VARCHAR(5),
CLASS VARCHAR(15),
DATE_OF_CANCELLATION DATE,
PID INT,
PASSPORTNO VARCHAR(10),
FOREIGN KEY(PID, PASSPORTNO) REFERENCES PASSENGER1(PID,
PASSPORTNO) ON DELETE CASCADE);
```

```
CREATE TABLE TICKET2
(DATE_OF_BOOKING DATE NOT NULL,
SOURCE VARCHAR(3) NOT NULL,
DESTINATION VARCHAR(3) NOT NULL,
CLASS VARCHAR(15) NOT NULL,
PRICE INT,
PRIMARY KEY(DATE_OF_BOOKING, SOURCE, DESTINATION, CLASS));

CREATE TABLE TICKET3
(DATE_OF_CANCELLATION DATE NOT NULL,
SURCHARGE INT,
PRIMARY KEY(DATE_OF_CANCELLATION));
```

6.5 Trigger

Triggers in SQL Server are used to execute after, or before an INSERT, DELETE, or an UPDATE operation on a table. You can use these SQL triggers on Views, or Tables to perform any of the above-specified operations. Remember, you can associate a trigger to a single table only.

Trigger to update 'TICKET_PRICE_HISTORY' table when the price of an air ticket is updated in TICKET2 table.

```
CREATE TABLE TICKET_PRICE_HISTORY
(DATE_OF_BOOKING DATE NOT NULL,
SOURCE VARCHAR(3) NOT NULL,
DESTINATION VARCHAR(3) NOT NULL,
CLASS VARCHAR2(15) NOT NULL,
PRICE INT,
PRIMARY KEY(DATE_OF_BOOKING, SOURCE, DESTINATION, CLASS));


CREATE OR REPLACE TRIGGER TICKET_PRICE_HISTORY
BEFORE UPDATE OF PRICE
ON TICKET2
FOR EACH ROW
BEGIN
INSERT INTO TICKET_PRICE_HISTORY
VALUES(OLD.DATE_OF_BOOKING, OLD.SOURCE, OLD.DESTINATION,
OLD.CLASS, OLD.PRICE,NOW());
END;
```

Chapter 7

TESTING

7.1 Introduction

Software testing is an investigation conducted to provide stake-holders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

- meets the requirements that guided its design and development,
- responds correctly to all kinds of inputs,
- performs its functions within an acceptable time,
- is sufficiently usable,
- can be installed and run in its intended environments, and
- Achieves the general result its stakeholder's desire.

As the number of possible tests for even simple software components is practically infinite, all software testing uses some strategy to select tests that are feasible for the available time and resources. As a result, software testing typically (but not exclusively) attempts to execute a program or application with the intent of finding software bugs (errors or other

defects). The job of testing is an iterative process as when one bug is fixed; it can illuminate other, deeper bugs, or can even create new ones.

Software testing can provide objective, independent information about the quality of software and risk of its failure to users or sponsors.

Software testing can be conducted as soon as executable software (even if partially complete) exists. The overall approach to software development often determines when and how testing is conducted. For example, in a phased process, most testing occurs after system requirements have been defined and then implemented in testable programs. In contrast, under an Agile approach, requirements, programming, and testing are often done concurrently.

7.2 Testing Principles

- All tests should be traceable to end user requirements.
- Tests should be planned long before testing begins.
- Testing should begin on a small scale and progress towards resting in large.
- Exhaustive testing is not possible.
- To be most effective testing should be conducted by an independent third party.

The primary objective for test case design is to derive a set of tests than has the highest livelihood for uncovering defects in software. To accomplish this objective, two different categories of test case design techniques are used. They are:

- White box Testing
- Black box Testing

White-box Testing:

White box testing focus on the program control structure. Test cases are derived to ensure that all statements in the program have been executed at least once during testing and that all logical conditions have been executed.

Black-box Testing:

Black box testing is designed to validate functional requirements without regard to the internal workings of a program. Black box testing mainly focuses on the information domain of the software, deriving test cases by partitioning input and output in a manner that provides through test coverage. Incorrect and missing functions, interface errors. Errors in data structures, error in functional logic are the errors falling in this category.

7.3 Levels of Testing

The following are the main levels of testing:

- **Unit Testing:** A level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed.
- **Integration Testing:** A level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units.
- **System Testing:** A level of the software testing process where a complete, integrated system/software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements.
- **Acceptance Testing:** Acceptance Testing has been implemented after providing the system to different users for testing. The system is found to be working properly on the user systems without any errors/faults.

7.4 Test Cases

Test cases are derived to ensure that all statements in the program have been executed at least once during testing and that all logical conditions have been executed. Using White-Box testing method, the software engineer can drive test cases that:

- Guarantee the logical decisions on their true and false side.
- Exercise all logical decisions on their true and false sides.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structure to assure their validity.

The test case specification for system testing has to be submitted for review before system testing commences.

Chapter 8

RESULTS

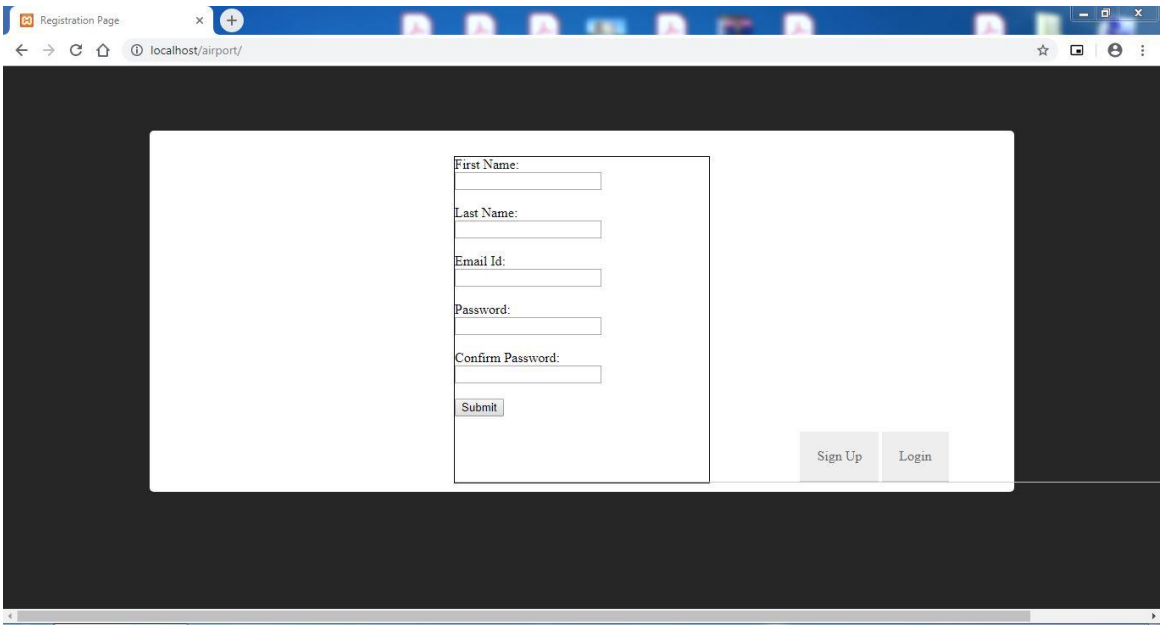


Fig. 8.1 Sign Up

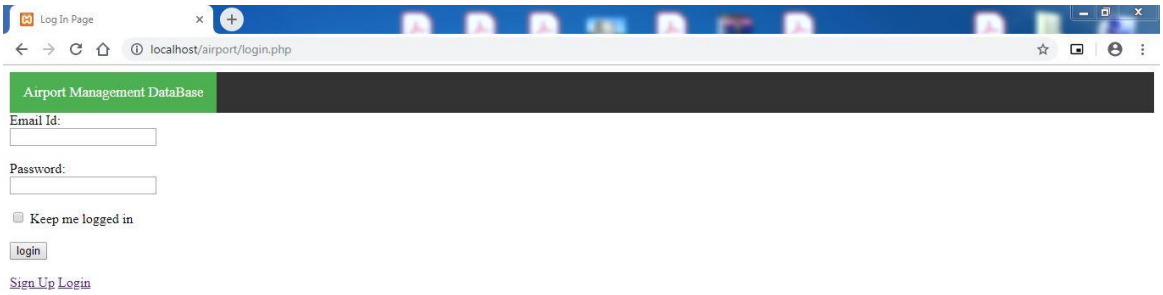


Fig. 8.2 Login

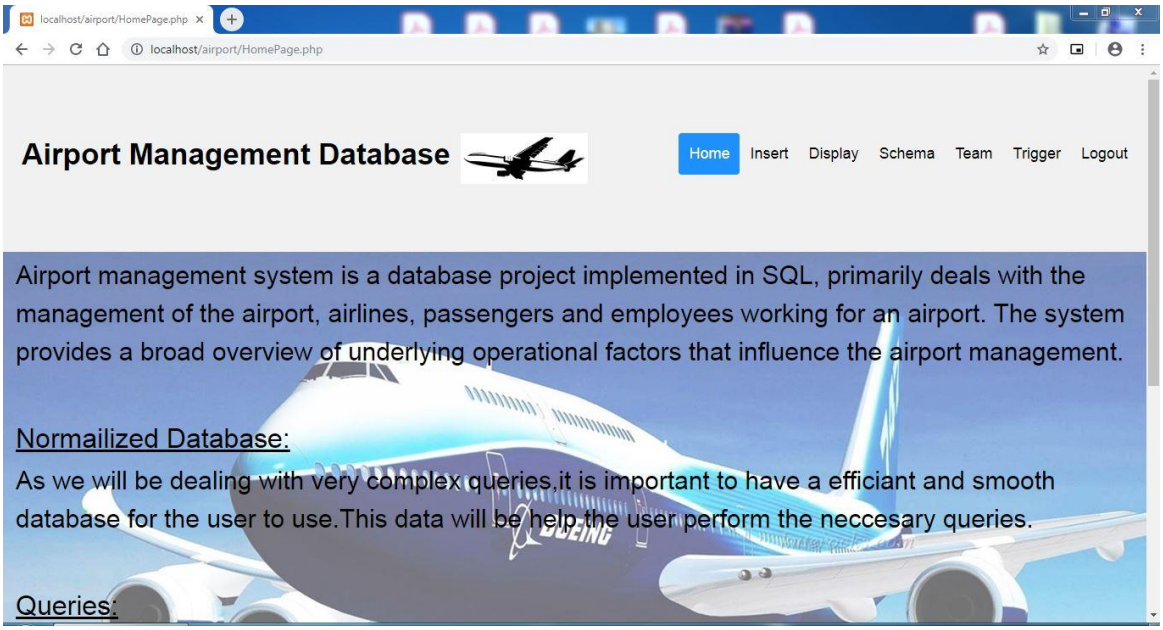


Fig. 8.3 Home Page

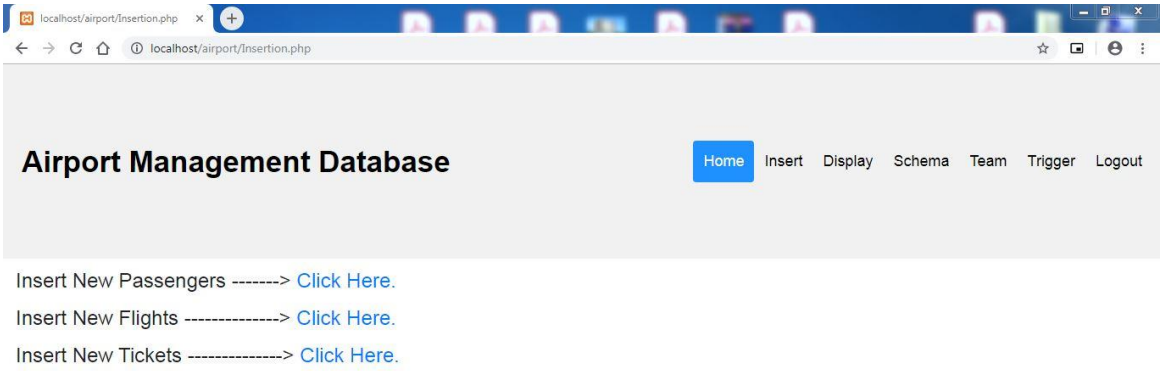


Fig. 8.4 Insert Page

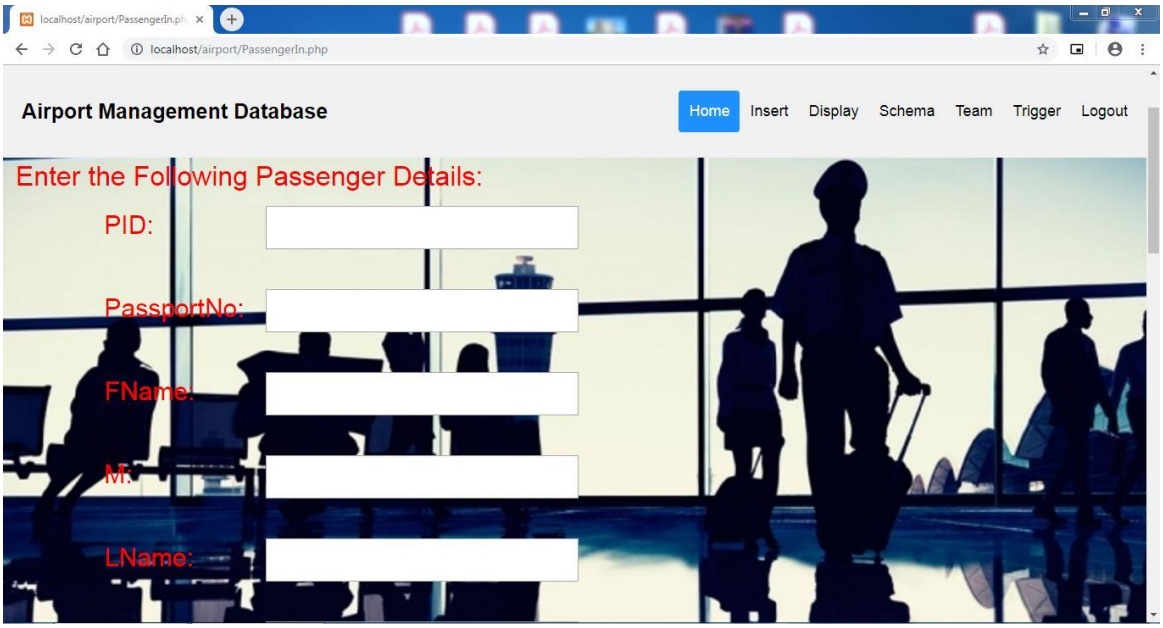


Fig. 8.5 Insert Passenger Details

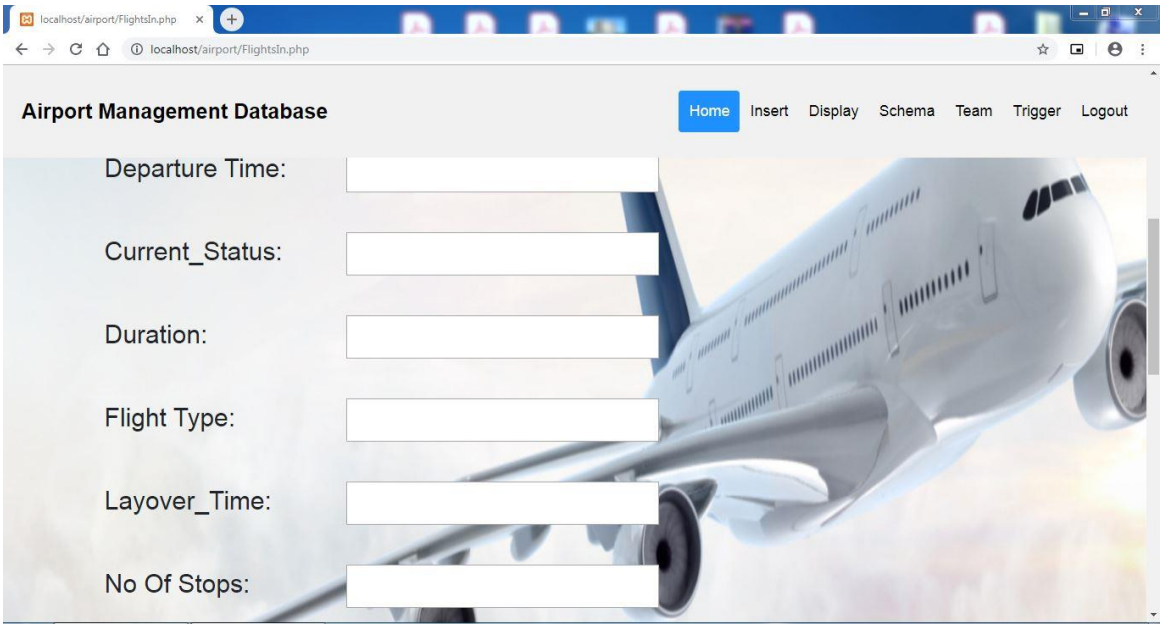


Fig. 8.6 Insert Flight Details

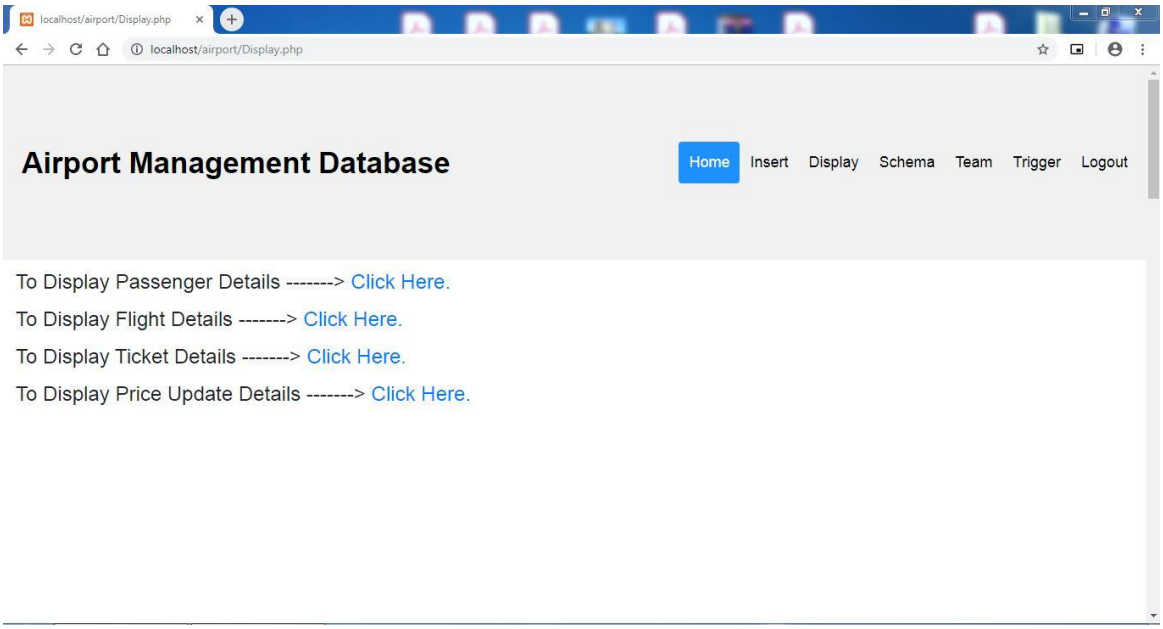
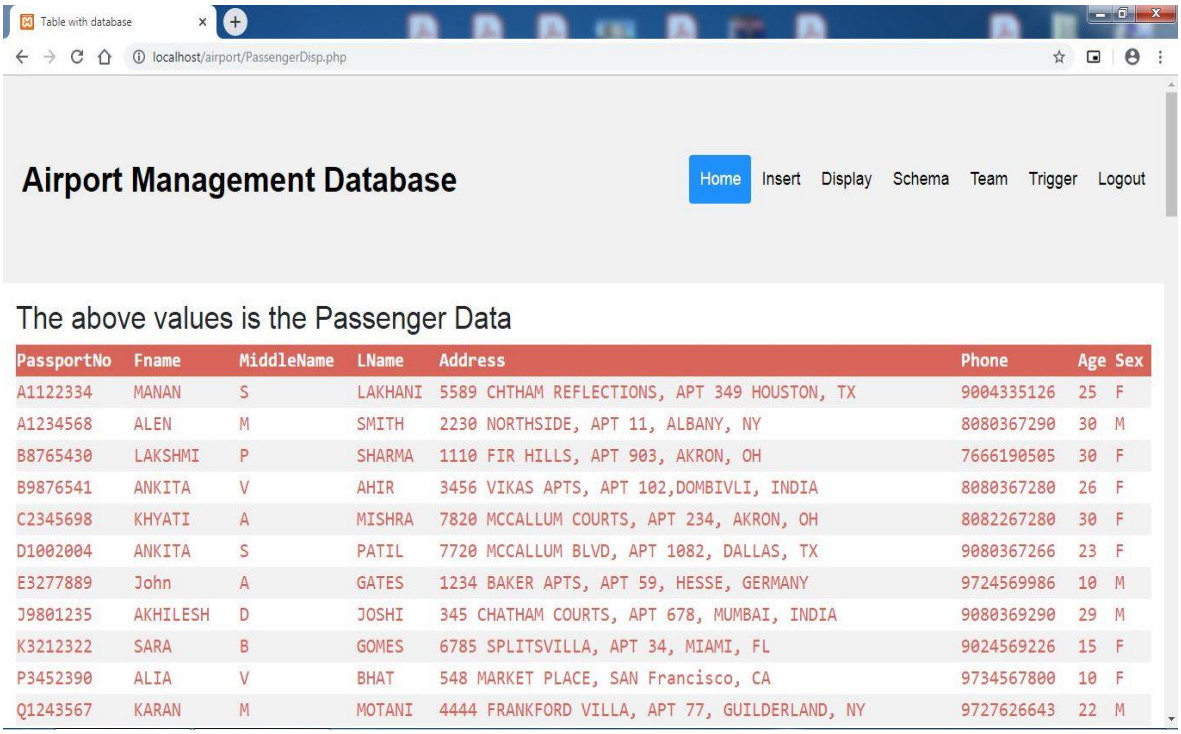
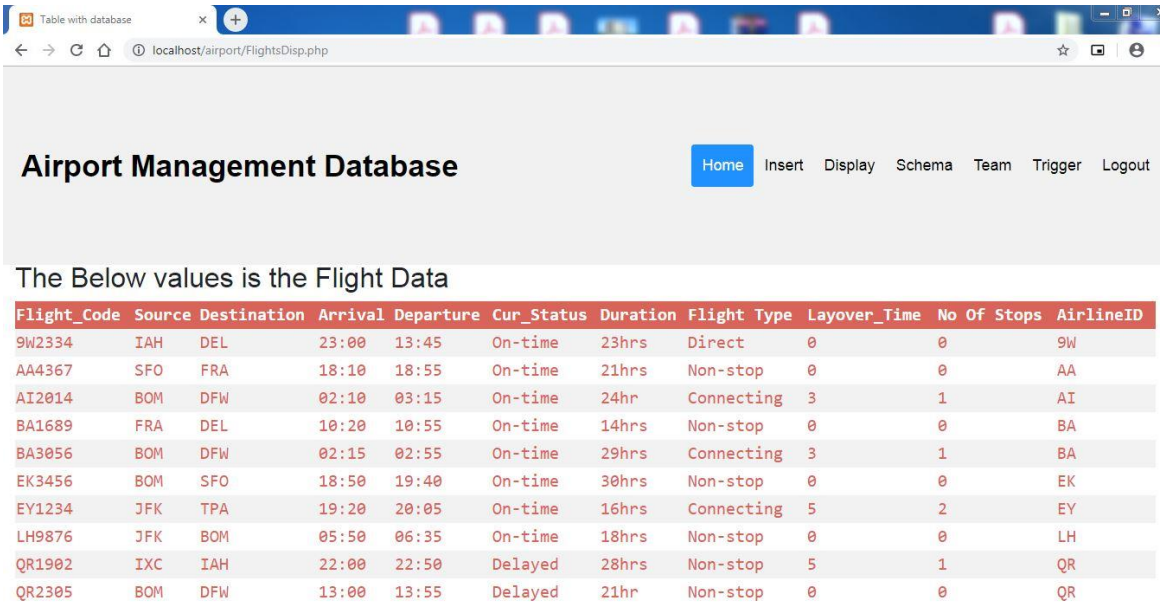


Fig. 8.7 Display Page



PassportNo	Fname	MiddleName	LName	Address	Phone	Age	Sex
A1122334	MANAN	S	LAKHANI	5589 CHTHAM REFLECTIONS, APT 349 HOUSTON, TX	9004335126	25	F
A1234568	ALEN	M	SMITH	2230 NORTHSIDE, APT 11, ALBANY, NY	8080367290	30	M
B8765430	LAKSHMI	P	SHARMA	1110 FIR HILLS, APT 903, AKRON, OH	7666190505	30	F
B9876541	ANKITA	V	AHIR	3456 VIKAS APTS, APT 102,DOMBIVLI, INDIA	8080367280	26	F
C2345698	KHYATI	A	MISHRA	7820 MCCALLUM COURTS, APT 234, AKRON, OH	8082267280	30	F
D1002004	ANKITA	S	PATIL	7720 MCCALLUM BLVD, APT 1082, DALLAS, TX	9080367266	23	F
E3277889	John	A	GATES	1234 BAKER APTS, APT 59, HESSE, GERMANY	9724569986	10	M
J9801235	AKHILESH	D	JOSHI	345 CHATHAM COURTS, APT 678, MUMBAI, INDIA	9080369290	29	M
K3212322	SARA	B	GOMES	6785 SPLITSVILLA, APT 34, MIAMI, FL	9024569226	15	F
P3452390	ALIA	V	BHAT	548 MARKET PLACE, SAN Francisco, CA	9734567800	10	F
Q1243567	KARAN	M	MOTANI	4444 FRANKFORD VILLA, APT 77, GUILDERLAND, NY	9727626643	22	M

Fig. 8.8 Passenger Details



Flight_Code	Source	Destination	Arrival	Departure	Cur_Status	Duration	Flight Type	Layover_Time	No Of Stops	AirlineID
9W2334	IAH	DEL	23:00	13:45	On-time	23hrs	Direct	0	0	9W
AA4367	SFO	FRA	18:10	18:55	On-time	21hrs	Non-stop	0	0	AA
AI2014	BOM	DFW	02:10	03:15	On-time	24hr	Connecting	3	1	AI
BA1689	FRA	DEL	10:20	10:55	On-time	14hrs	Non-stop	0	0	BA
BA3056	BOM	DFW	02:15	02:55	On-time	29hrs	Connecting	3	1	BA
EK3456	BOM	SFO	18:50	19:40	On-time	30hrs	Non-stop	0	0	EK
EY1234	JFK	TPA	19:20	20:05	On-time	16hrs	Connecting	5	2	EY
LH9876	JFK	BOM	05:50	06:35	On-time	18hrs	Non-stop	0	0	LH
QR1902	IXC	IAH	22:00	22:50	Delayed	28hrs	Non-stop	5	1	QR
QR2305	BOM	DFW	13:00	13:55	Delayed	21hr	Non-stop	0	0	QR

Fig. 8.9 Flight Details

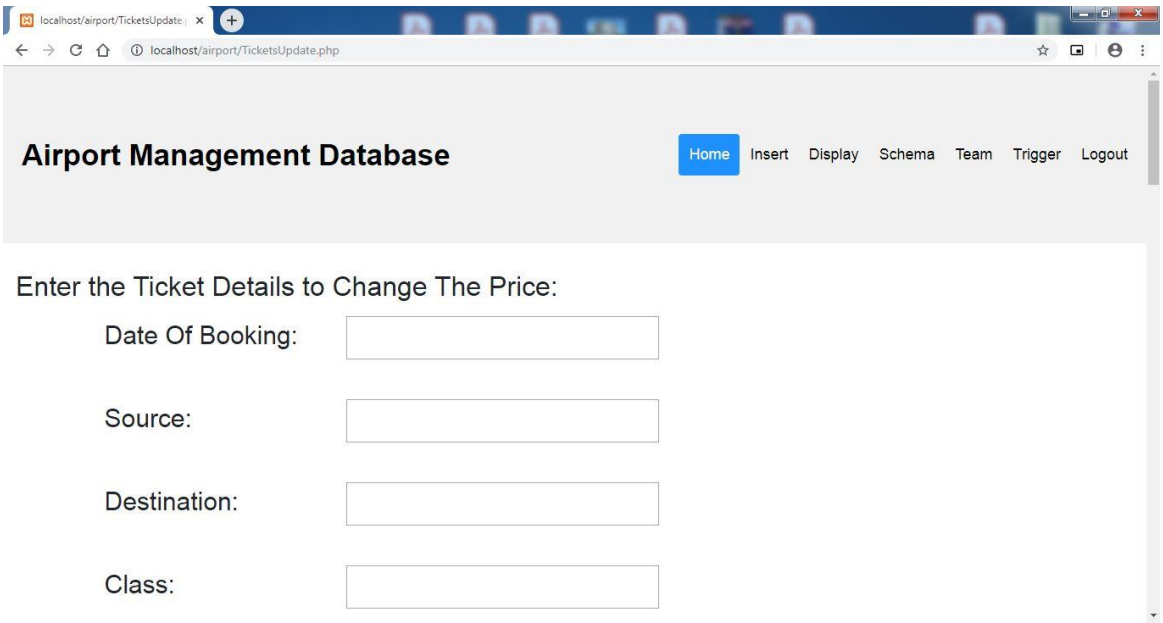


Fig. 8.10 Trigger

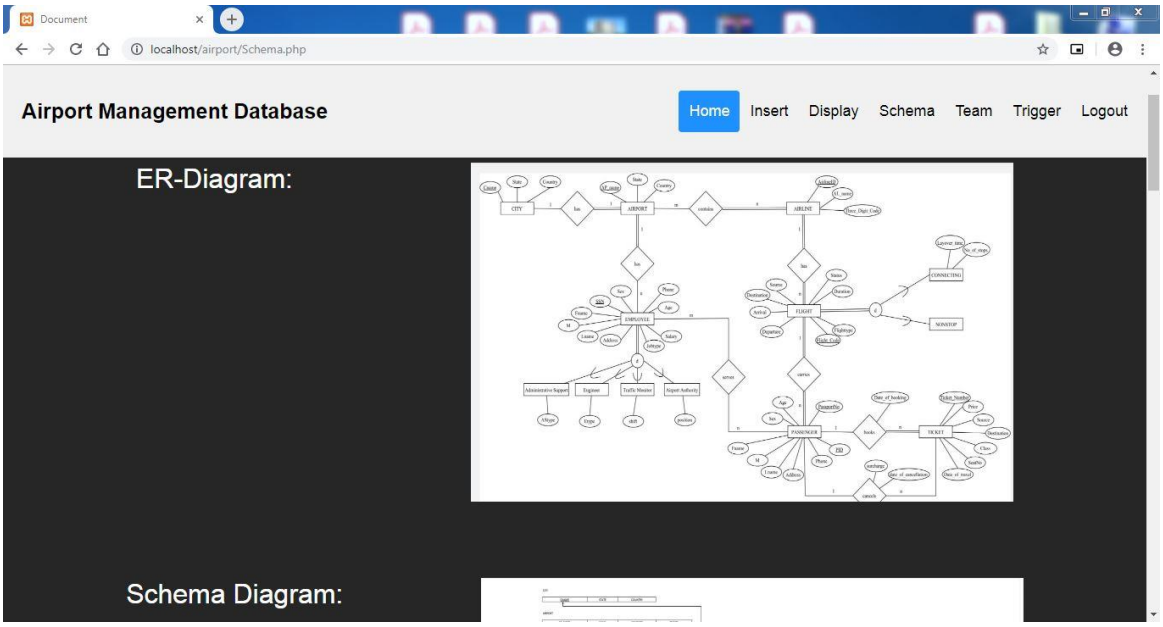


Fig. 8.11 Schema & ER Diagrams

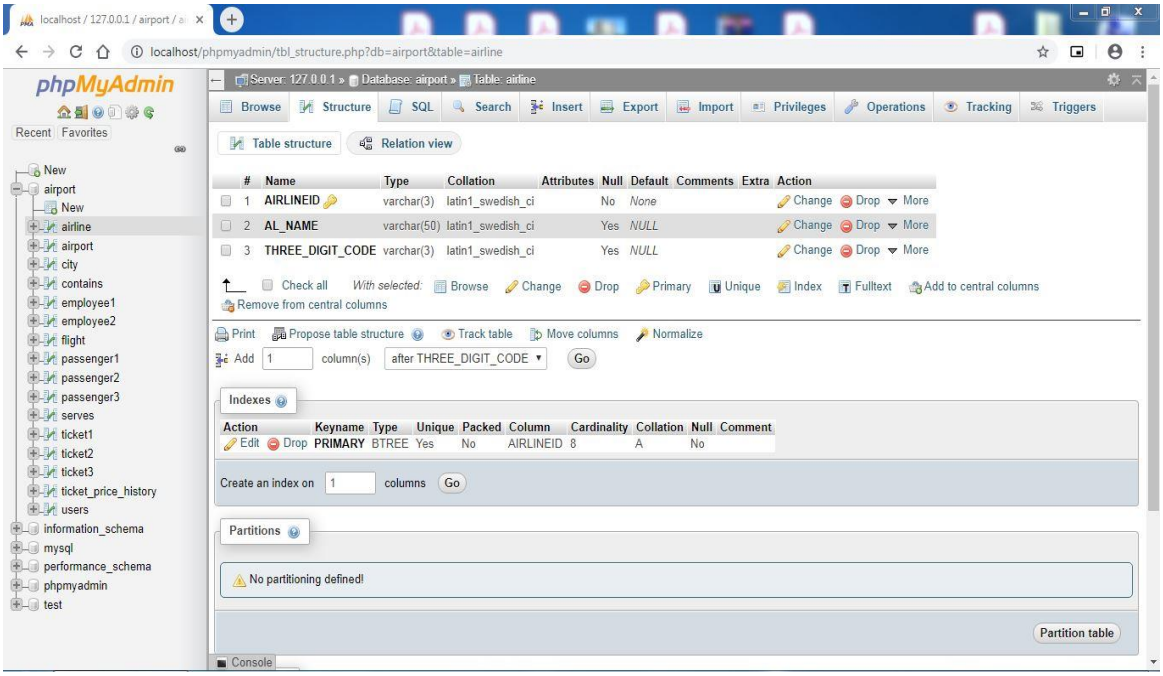


Table 8.1 Airline

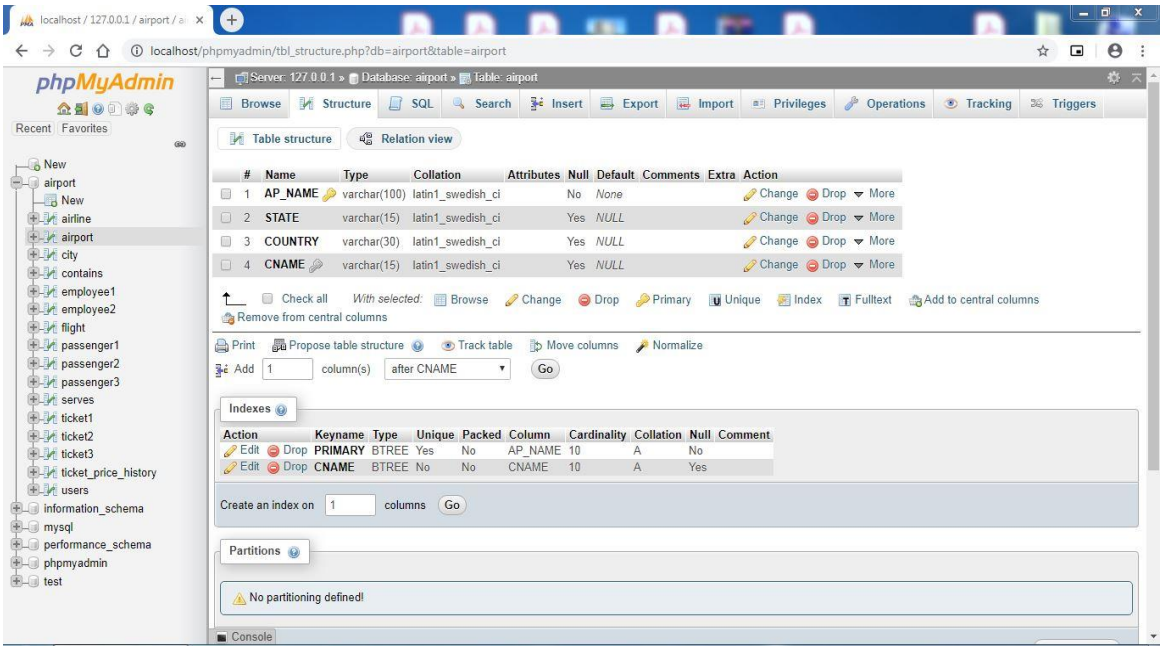


Table 8.2 Airport

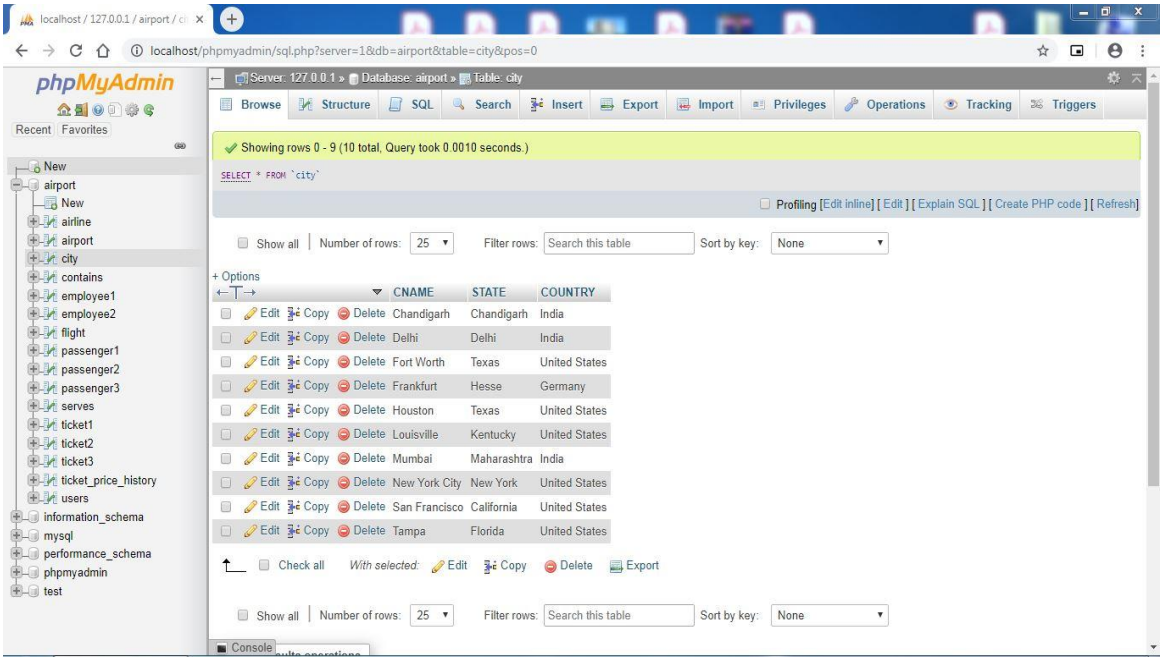


Table 8.3 City

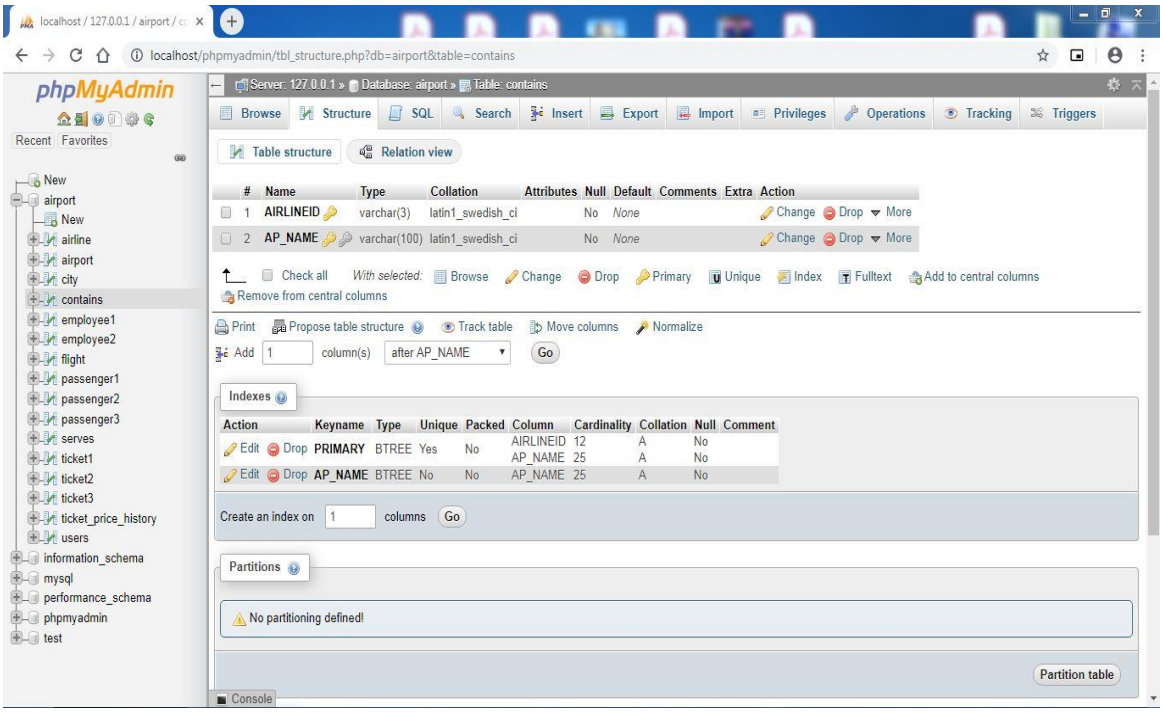


Table 8.4 Contains

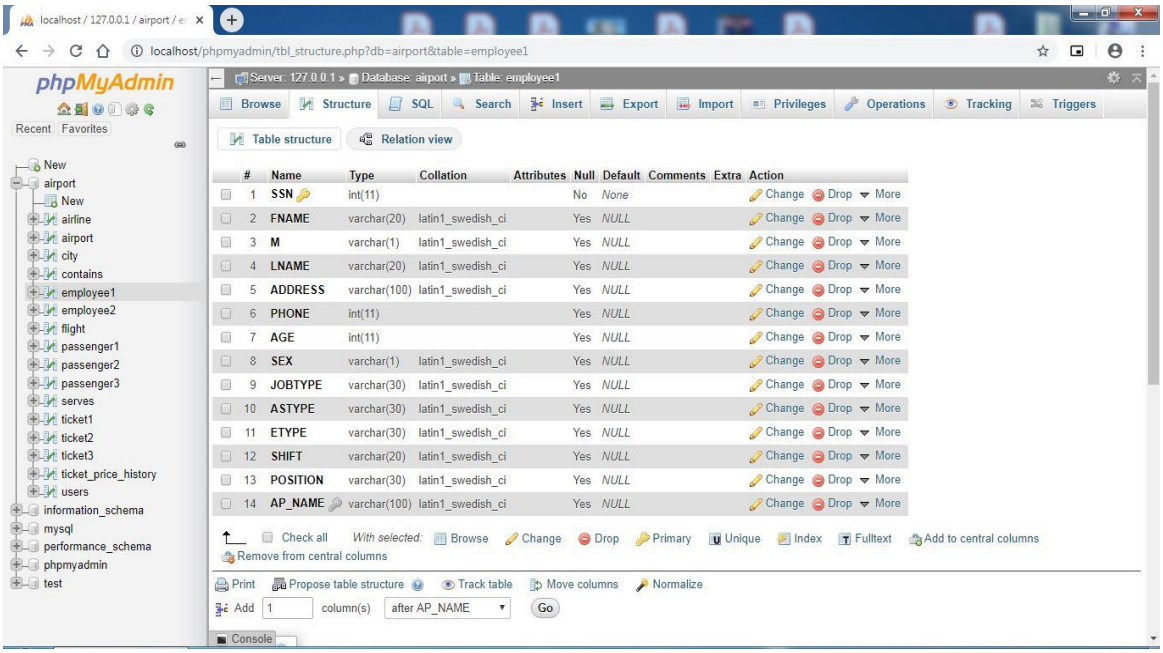


Table 8.5 Employee1

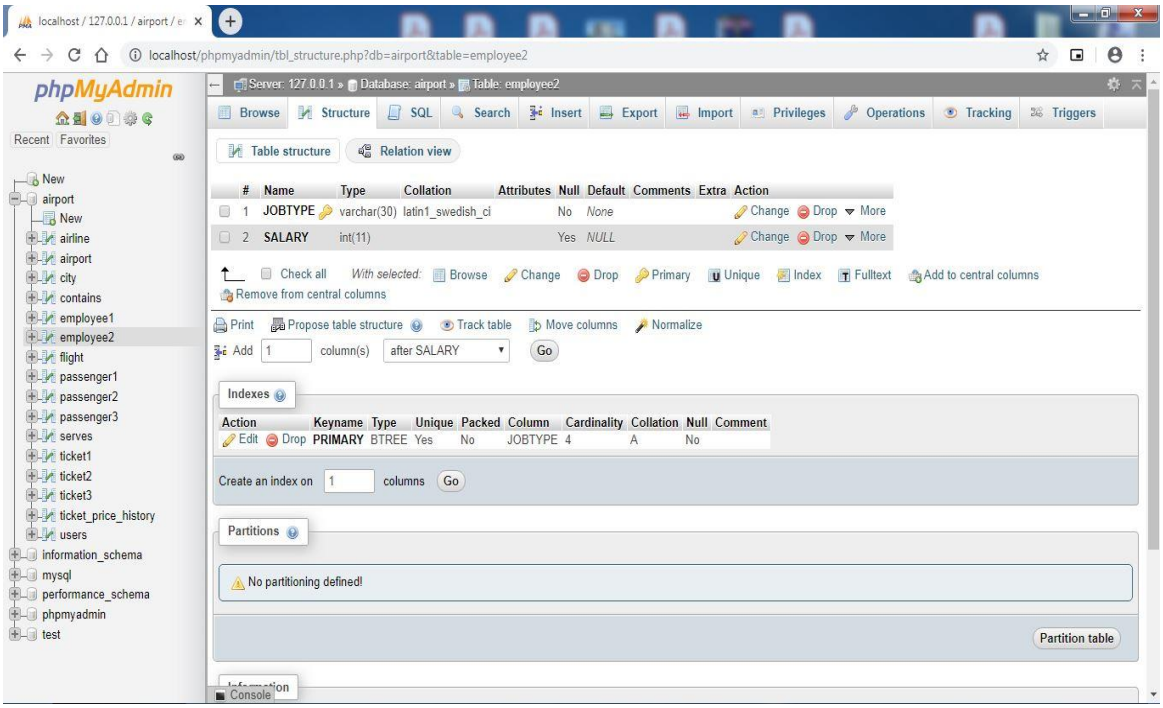


Table 8.6 Employee2

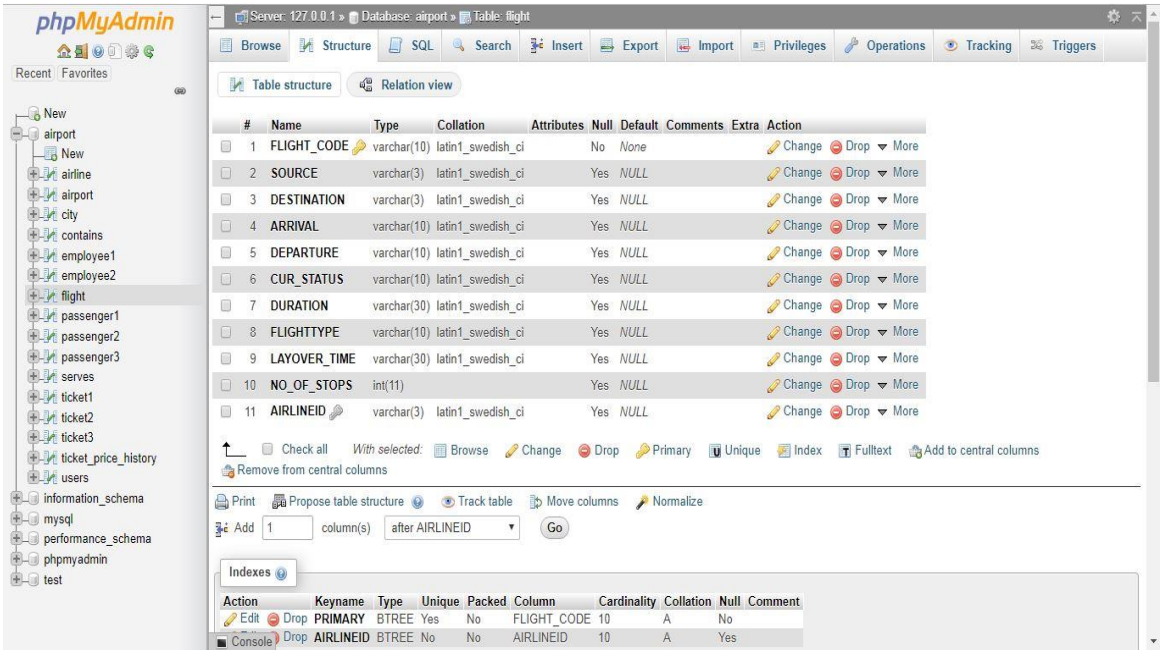


Table 8.7 Flight

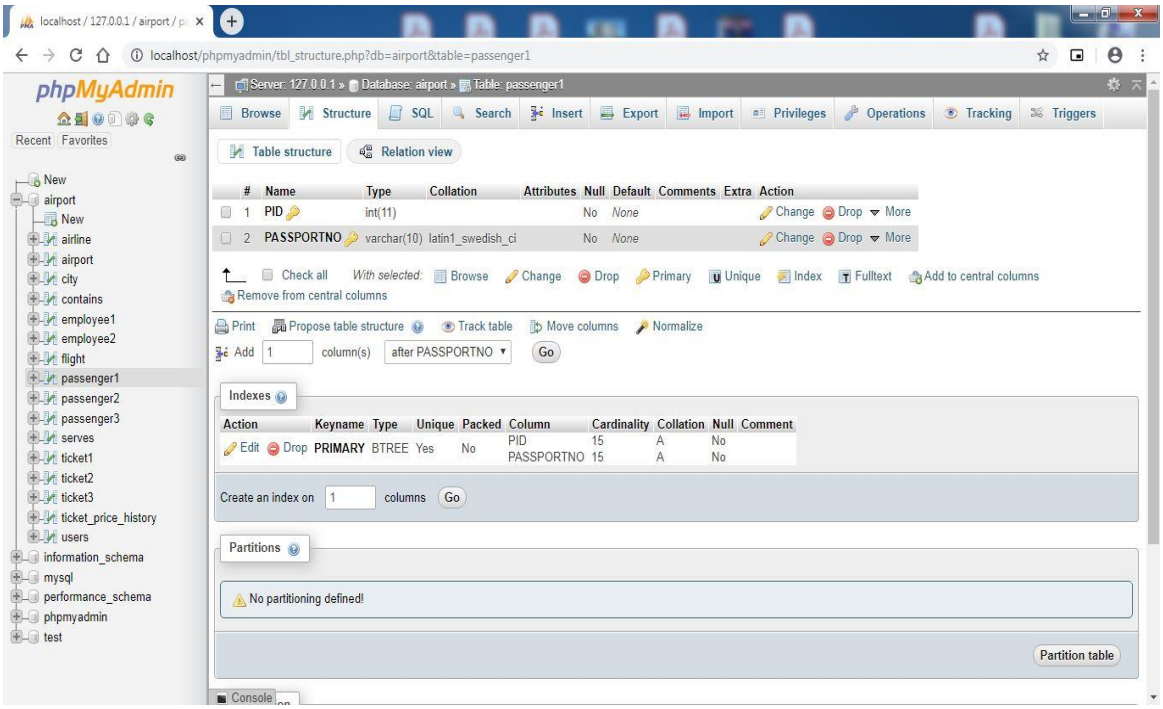


Table 8.8 Passenger1

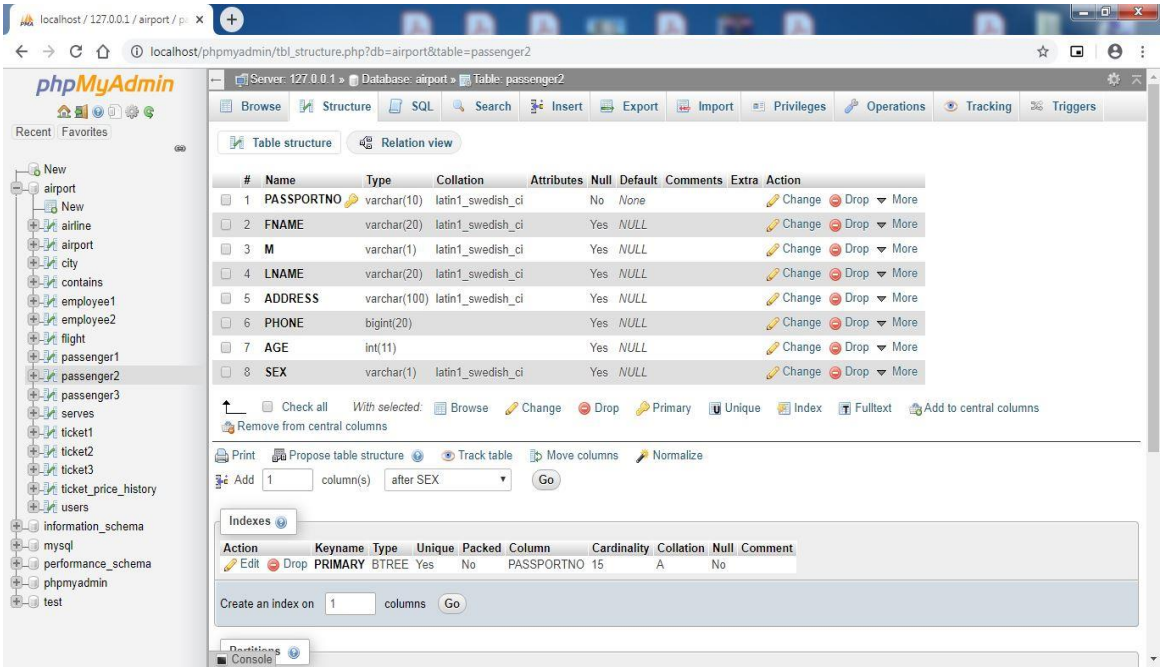


Table 8.9 Passenger2

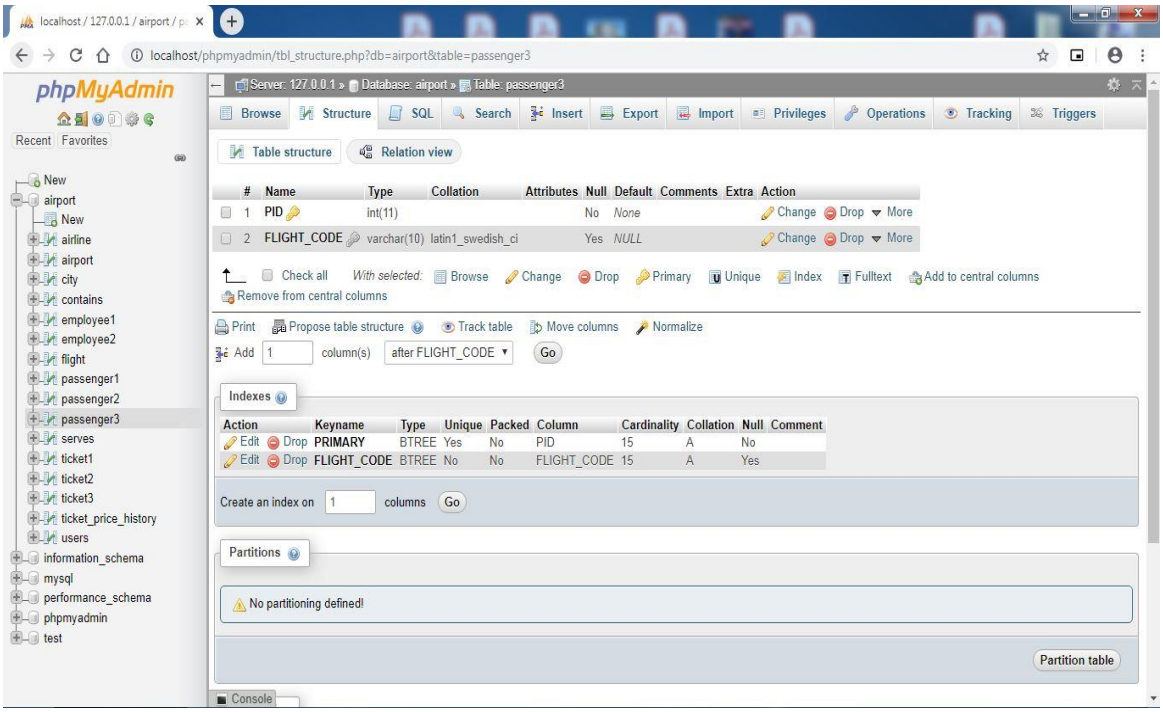


Table 8.10 Passenger3

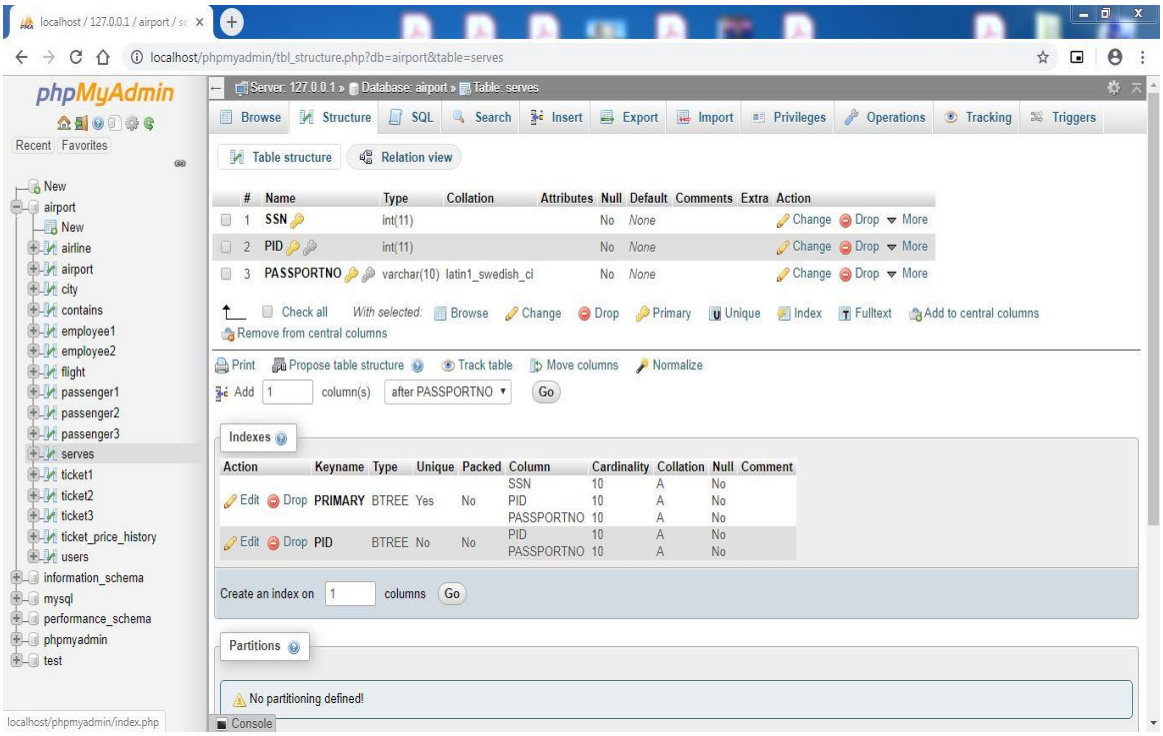


Table 8.11 Serves

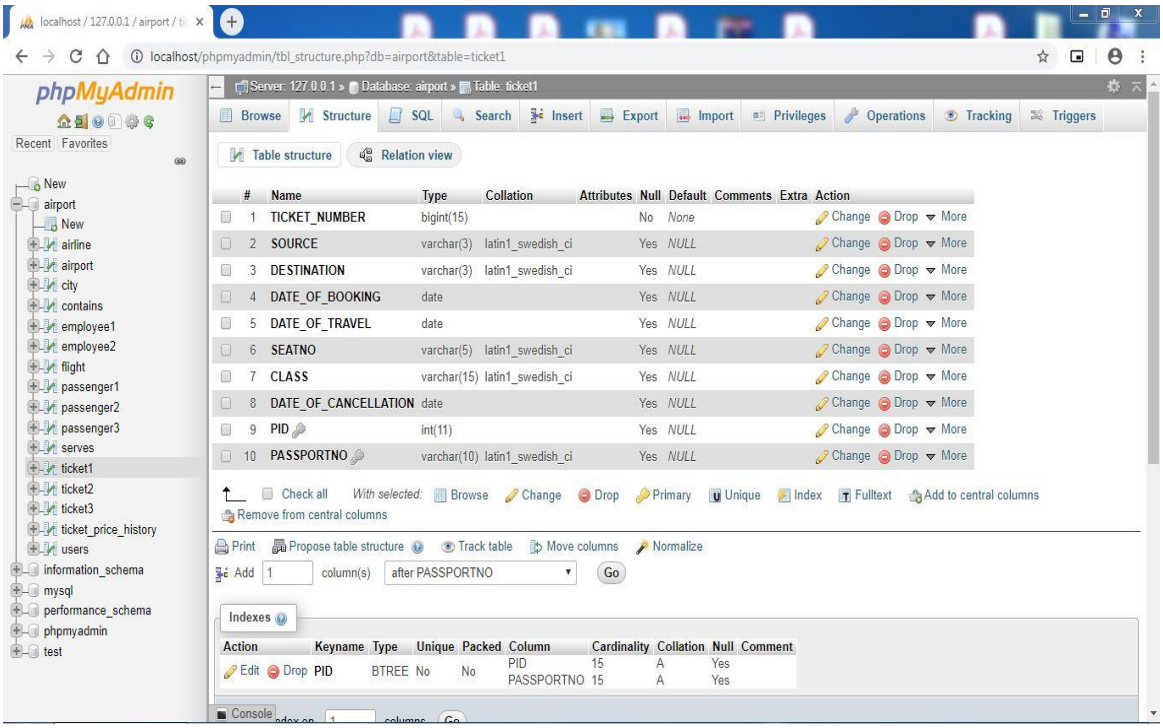


Table 8.12 Ticket1

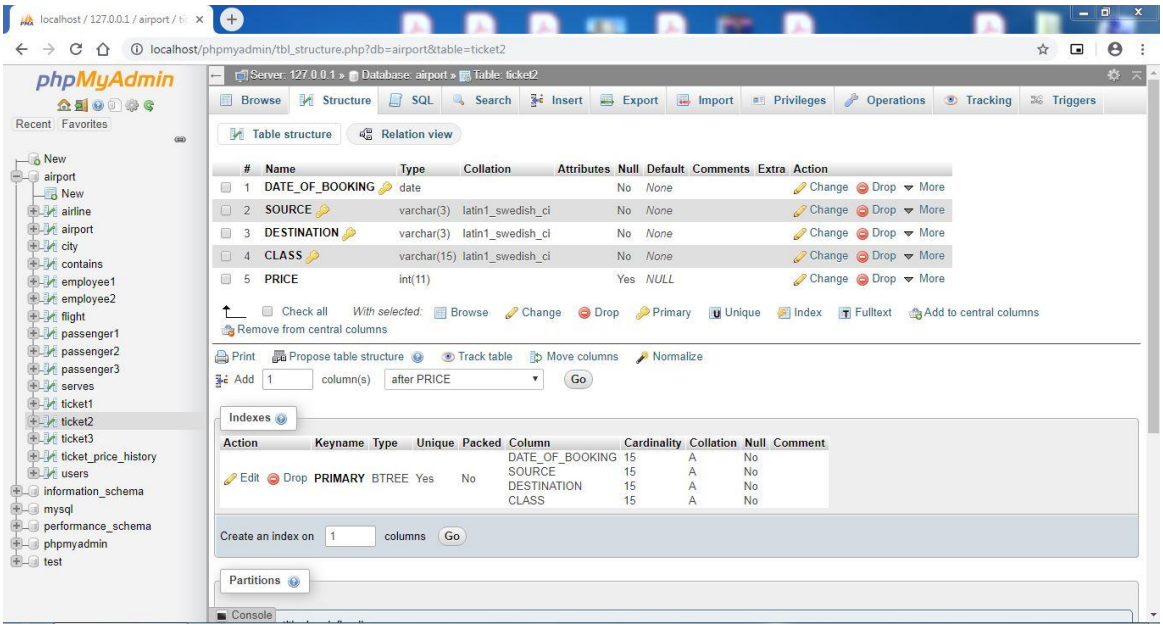


Table 8.13 Ticket2

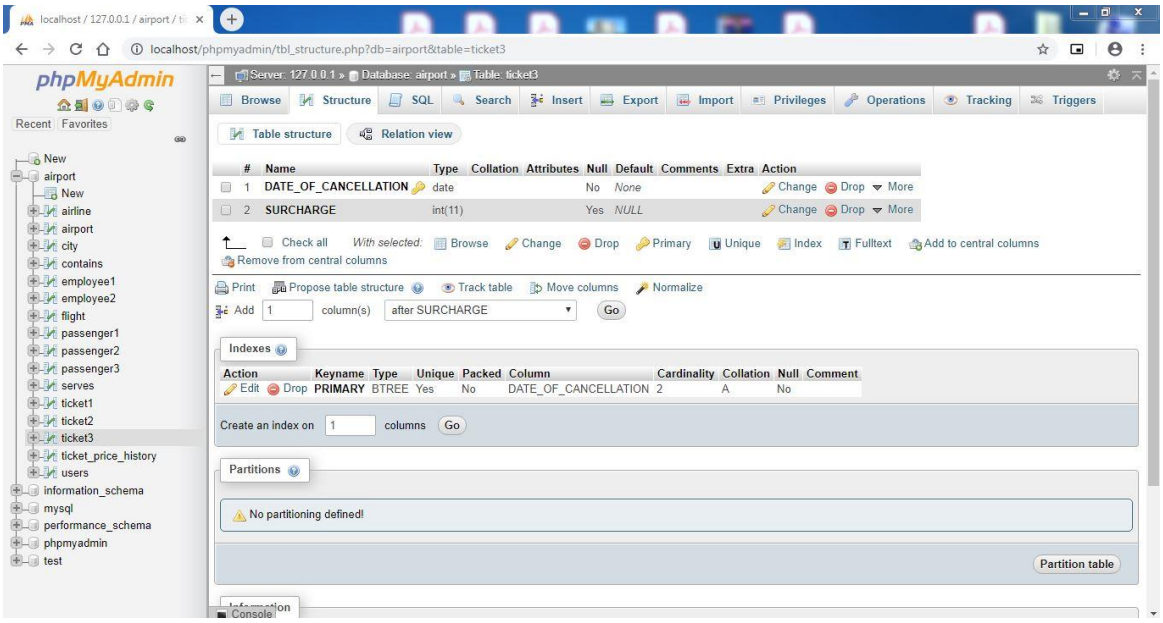


Table 8.14 Ticket3

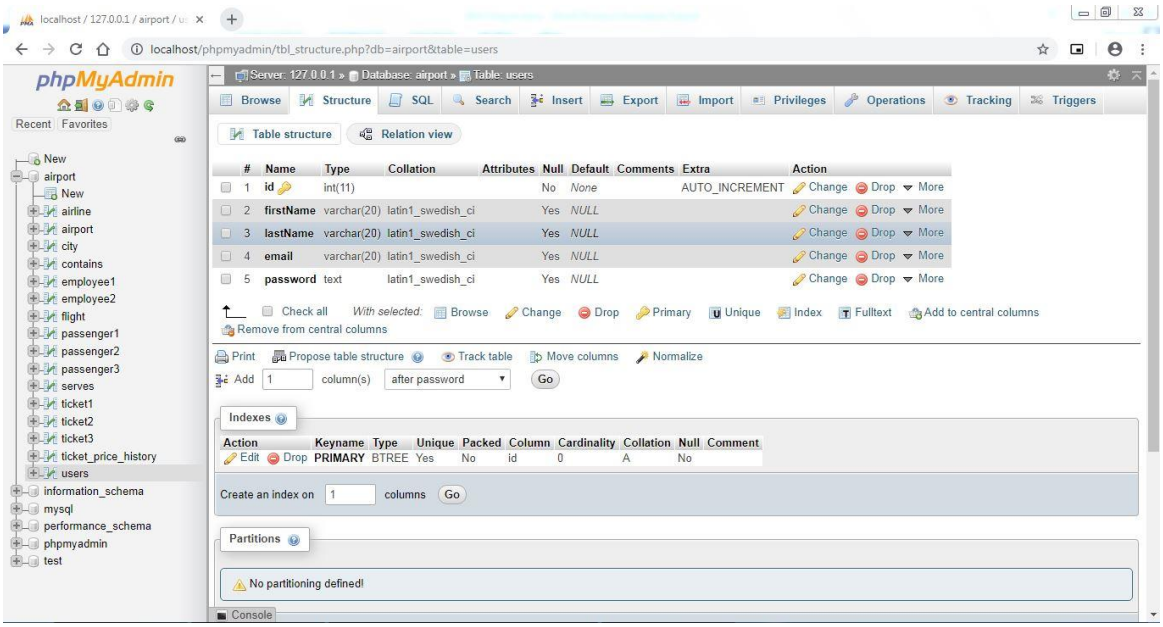


Table 8.15 Users

Conclusion

This project is designed to meet the requirements of Airport Management. It has been developed using MySQL, HTML and PHP, keeping in mind the specifications of the system.

We have provided all the basic operation required to run the application smoothly and meet the requirements. The project is designed in such a way that any future modification can be done easily. There is more scope to innovate in the web application as the database already contains most of the required fields.

FUTURE ENHANCEMENTS:

We can make a centralized system within the state/country. Security for the application can be improved as the security risks are very high. We can add more triggers and stored procedures to improve efficiency and more criteria in viewing the information. Functions like sort, search, delete can be added to make use of the database to its complete potential.

References

Websites:

https://www.geeksforgeeks.org/

https://www.w3schools.com/

https://www.tutorialspoint.com/

https://www.wikipedia.org/

Citations:

Airline Codes

http://www.iata.org/publications/Pages/code-search.aspx

Test Strategy Definition

https://www.tutorialspoint.com/software_testing_dictionary/test_strategy.htm

Unit Testing

http://softwaretestingfundamentals.com/unit-testing/

Integration Testing

http://softwaretestingfundamentals.com/integration-testing/

System Testing

http://softwaretestingfundamentals.com/system-testing/

Acceptance Testing

http://softwaretestingfundamentals.com/acceptance-testing/