

Project Name: Airline Reservation System.

This Airline reservation system is to maintain the booking record of the customers and its payments and show them the flights according to their choice (source, destination). It also includes records of airlines and its manufacturers. This database will provide an easy access to airline reservation system and its findings such as total revenues by all airliner. We can easily find the airline which has generated largest revenues.

For this database, there are several tables such as

Customer table:

```
DROP TABLE IF EXISTS `customer`;
```

```
CREATE TABLE `customer` (  
  `CustomerId` int(11) NOT NULL,  
  `FirstName` varchar(45) DEFAULT NULL,  
  `LastName` varchar(45) DEFAULT NULL,  
  `Mobilen0` int(11) DEFAULT NULL,  
  `EmailId` varchar(45) DEFAULT NULL,  
  `AddressID` int(11) NOT NULL,  
  `CreditCard_CardID` int(11) NOT NULL,  
  PRIMARY KEY (`CustomerId`),  
  KEY `fk_Customer_Address_idx` (`AddressID`),  
  KEY `fk_Customer_CreditCard1_idx` (`CreditCard_CardID`),  
  CONSTRAINT `AddressID` FOREIGN KEY (`AddressID`) REFERENCES `address` (`AddressID`) ON DELETE NO ACTION ON  
  UPDATE NO ACTION,  
  CONSTRAINT `fk_Customer_CreditCard1` FOREIGN KEY (`CreditCard_CardID`) REFERENCES `creditcard` (`CardID`) ON  
  DELETE NO ACTION ON UPDATE NO ACTION  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

2.Flight

```
DROP TABLE IF EXISTS `flight`;  
CREATE TABLE `flight` (  
  `FlightID` varchar(23) NOT NULL,  
  `Destination` varchar(45) DEFAULT NULL,  
  `Source` varchar(45) DEFAULT NULL,  
  `DepartureTime` datetime DEFAULT NULL,  
  `ArrivalTime` datetime DEFAULT NULL,  
  `CertificateId` int(11) NOT NULL,  
  `AirlinerId` int(11) NOT NULL,  
  `TotalSeatCapacity` int(11) DEFAULT NULL,  
  `Price` varchar(45) DEFAULT NULL,  
  `TotalSeatBooked` int(11) DEFAULT NULL,  
  PRIMARY KEY (`FlightID`),  
  KEY `fk_Flight_MaintainanceCertificate1_idx` (`CertificateId`),  
  KEY `fk_Flight_Airliner1_idx` (`AirlinerId`),  
  CONSTRAINT `AirlinerId` FOREIGN KEY (`AirlinerId`) REFERENCES `airliner` (`AirlinerId`) ON DELETE NO  
  ACTION ON UPDATE NO ACTION,  
  CONSTRAINT `CertificateId` FOREIGN KEY (`CertificateId`) REFERENCES `maintainancecertificate`  
  (`CertificateId`) ON DELETE NO ACTION ON UPDATE NO ACTION  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Project Name: Airline Reservation System.

3. Address Table:

```
CREATE TABLE `address` (  
  `AddressID` int(11) NOT NULL,  
  `Zipcode` int(11) DEFAULT NULL,  
  `State` varchar(45) DEFAULT NULL,  
  `City` varchar(45) DEFAULT NULL,  
  `Country` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`AddressID`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

4.Airliner Table

```
CREATE TABLE `airliner` (  
  `AirlinerId` int(11) NOT NULL,  
  `AirlinerName` varchar(45) DEFAULT NULL,  
  `AirlinerType` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`AirlinerId`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

5. Airliner_has_manufaturer Table

```
DROP TABLE IF EXISTS `airliner_has_manufaturer`;  
CREATE TABLE `airliner_has_manufaturer` (  
  `ManuID` int(11) NOT NULL,  
  `Airliner_AirlinerId` int(11) NOT NULL,  
  PRIMARY KEY (`Airliner_AirlinerId`),  
  KEY `fk_Airliner_has_Manufaturer_Manufaturer1_idx` (`ManuID`),  
  CONSTRAINT `ManuID` FOREIGN KEY (`ManuID`) REFERENCES `manufaturer` (`ManuID`) ON DELETE NO  
ACTION ON UPDATE NO ACTION,  
  CONSTRAINT `fk_Airliner_has_Manufaturer_Airliner1` FOREIGN KEY (`Airliner_AirlinerId`) REFERENCES  
`airliner` (`AirlinerId`) ON DELETE NO ACTION ON UPDATE NO ACTION  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

6.Credit Card Table

```
DROP TABLE IF EXISTS `creditcard`;  
/*!40101 SET @saved_cs_client = @@character_set_client */;  
/*!40101 SET character_set_client = utf8 */;  
CREATE TABLE `creditcard` (  
  `CardID` int(11) NOT NULL,  
  `CardType` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`CardID`)  
) ENGINE=InnoDB DEFAULT CHARSET=cp850;
```

7.Maintainace Certificate Table:

Project Name: Airline Reservation System.

```
DROP TABLE IF EXISTS `maintainancecertificate`;  
CREATE TABLE `maintainancecertificate` (  
  `CertificateId` int(11) NOT NULL,  
  `CertificateStatus` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`CertificateId`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

8. Payment Table:

```
DROP TABLE IF EXISTS `payment`;  
CREATE TABLE `payment` (  
  `PaymentId` int(11) NOT NULL,  
  `PaymentDate` datetime DEFAULT NULL,  
  `PaymentAmount` int(11) DEFAULT NULL,  
  `PaymentMethodId` int(11) NOT NULL,  
  `ReservationID` int(11) NOT NULL,  
  PRIMARY KEY (`PaymentId`),  
  KEY `fk_Payment_Payment Method1_idx` (`PaymentMethodId`),  
  KEY `fk_Payment_Reservation1_idx` (`ReservationID`),  
  CONSTRAINT `PaymentMethodId` FOREIGN KEY (`PaymentMethodId`) REFERENCES `paymentmethod`  
  (`PaymentMethodId`) ON DELETE NO ACTION ON UPDATE NO ACTION,  
  CONSTRAINT `ReservationID` FOREIGN KEY (`ReservationID`) REFERENCES `reservation` (`ReservationID`) ON  
  DELETE NO ACTION ON UPDATE NO ACTION  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

9. Payment Method Table:

```
DROP TABLE IF EXISTS `paymentmethod`;  
CREATE TABLE `paymentmethod` (  
  `PaymentMethodId` int(11) NOT NULL,  
  `PaymentInvoiceCode` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`PaymentMethodId`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

10. Reservation Table:

```
DROP TABLE IF EXISTS `reservation`;  
  
CREATE TABLE `reservation` (  
  `ReservationID` int(11) NOT NULL,  
  `ReservationDate` varchar(45) DEFAULT NULL,  
  `CustomerId` int(11) NOT NULL,  
  `FlightID` varchar(23) NOT NULL,  
  `TicketTypeID` int(11) NOT NULL,  
  `SeatBooked` int(11) DEFAULT NULL,  
  PRIMARY KEY (`ReservationID`),  
  KEY `fk_Reservation_Customer1_idx` (`CustomerId`),  
  KEY `fk_Reservation_Flight1_idx` (`FlightID`),  
  KEY `TicketTypeID_idx` (`TicketTypeID`),  
  CONSTRAINT `CustomerId` FOREIGN KEY (`CustomerId`) REFERENCES `customer` (`CustomerId`) ON DELETE  
  NO ACTION ON UPDATE NO ACTION,
```

Project Name: Airline Reservation System.

```
CONSTRAINT `FlightID` FOREIGN KEY (`FlightID`) REFERENCES `flight` (`FlightID`) ON DELETE NO ACTION ON  
UPDATE NO ACTION,  
CONSTRAINT `TicketTypeID` FOREIGN KEY (`TicketTypeID`) REFERENCES `tickettype` (`TicketTypeID`) ON  
DELETE NO ACTION ON UPDATE NO ACTION  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

11. SeatBooked Table:

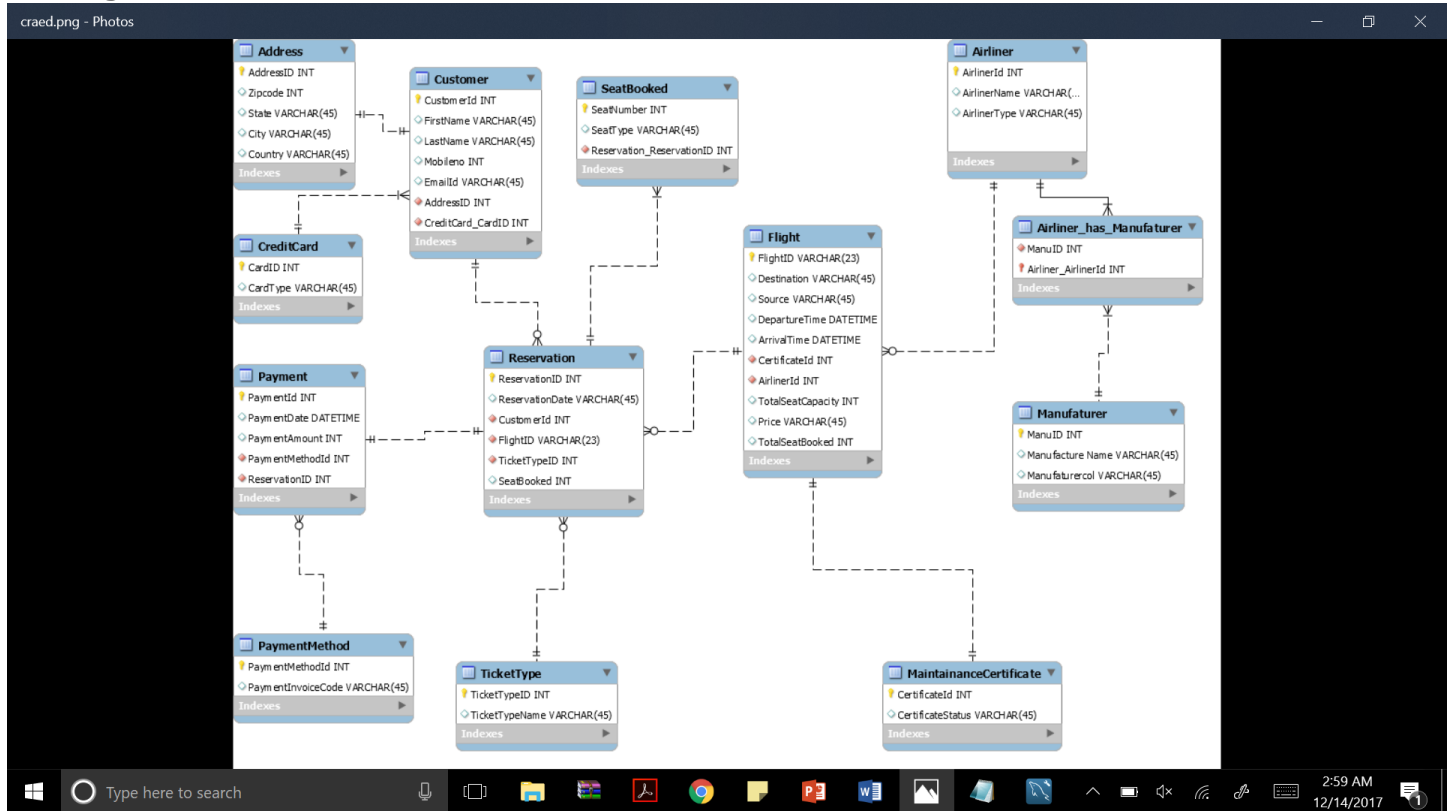
```
DROP TABLE IF EXISTS `seatbooked`;  
CREATE TABLE `seatbooked` (  
  `SeatNumber` int(11) NOT NULL,  
  `SeatType` varchar(45) DEFAULT NULL,  
  `Reservation_ReservationID` int(11) NOT NULL,  
  PRIMARY KEY (`SeatNumber`),  
  KEY `fk_SeatBooked_Reservation1_idx` (`Reservation_ReservationID`),  
  CONSTRAINT `fk_SeatBooked_Reservation1` FOREIGN KEY (`Reservation_ReservationID`) REFERENCES  
  `reservation` (`ReservationID`) ON DELETE NO ACTION ON UPDATE NO ACTION  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

12. Ticket Type Table:

```
DROP TABLE IF EXISTS `tickettype`;  
CREATE TABLE `tickettype` (  
  `TicketTypeID` int(11) NOT NULL,  
  `TicketTypeName` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`TicketTypeID`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Project Name: Airline Reservation System.

ER Diagram :



Stored procedures:

1, This will tell you the which is top most revenue generated airline.

/*-----Create Stored Procedure--To Calculate Revenue of Top AirLiner*/

DELIMITER \$\$

create procedure CalculateRevenuesByAirline()

BEGIN

SELECT airliner.AirlinerName , sum(Price * flight.TotalSeatBooked) as rev from flight

INNER JOIN airliner

ON

airliner.AirlinerId = flight.AirlinerId

GROUP BY AirlinerName

order by rev desc

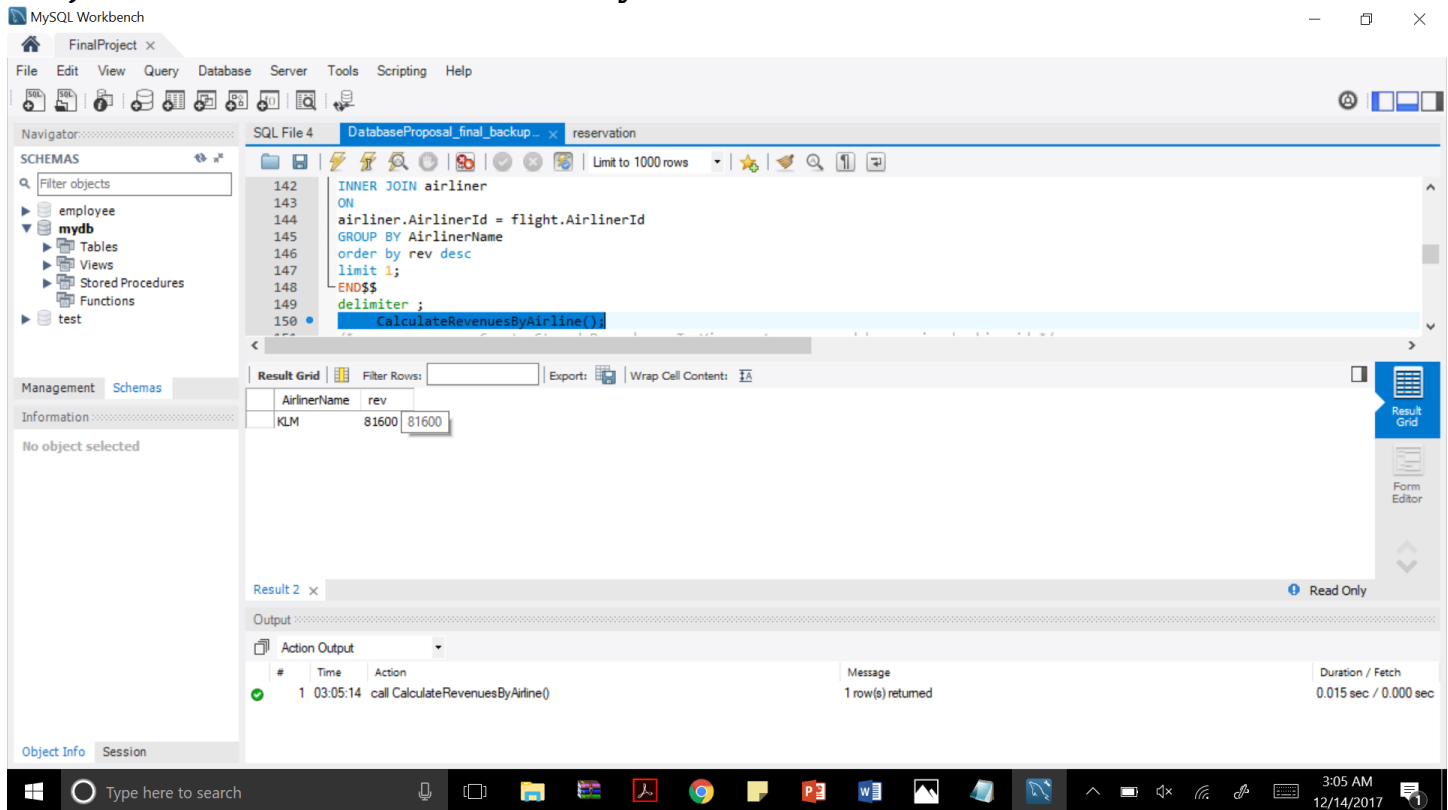
limit 1;

END\$\$

delimiter ;

call CalculateRevenuesByAirline();

Project Name: Airline Reservation System.



2. This will enable the maintenance certificate of the airline

/* -----Change Maintenance Certificate -----*/

DELIMITER \$\$

create procedure changeMaintenanceCertificateStatus(flightID varchar(30),CertificateId INT)

BEGIN

UPDATE flight

SET flight.CertificateId = CertificateId

WHERE flight.FlightID = flightID;

END\$\$

delimiter ;

Select * from flight;

call changeMaintenanceCertificateStatus('KL-713',1);

Project Name: Airline Reservation System.

MySQL Workbench

FinalProject x

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS

- Filter objects
- employee
- mydb
 - Tables
 - Views
 - Stored Procedures
 - Functions
- test

SQL File 4 DatabaseProposal_final_backup... reservation

```
SET flight.CertificateId = CertificateId
WHERE flight.FlightID = flightID;

END$$
delimiter ;

* flight;
```

Result Grid

FlightID	Destination	Source	DepartureTime	ArrivalTime	CertificateId	AirlineId	TotalSeatCapacity	Price	TotalSeatBooked
AF-360	Boston	Dallas	2017-12-24 21:40:00	2017-12-25 01:23:00	1	786	44	476	48
AI-456	Surat	Chicago	2018-01-07 21:40:00	2018-01-08 01:23:00	0	273	55	523	0
EM-360	London	Newark	2017-12-08 08:40:00	2017-12-09 18:23:00	1	747	75	400	30
EM-549	Frankfurt	Boston	2017-12-18 09:40:00	2017-12-19 12:18:23:00	1	747	50	546	32
KL-360	London	Newark	2017-12-28 10:40:00	2017-12-28 18:23:00	1	321	100	800	30
KL-712	Memphis	Boston	2017-12-20 06:40:00	2017-12-20 11:45:00	1	321	75	448	75
KL-713	London	Newark	2017-12-18 18:40:00	2017-12-19 18:23:00	1	321	65	400	60

flight 3 x Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	03:05:14	call CalculateRevenuesByAirline()	1 row(s) returned	0.015 sec / 0.000 sec
2	03:06:59	call changeMaintenanceCertificateStatus(KL-713,1)	0 row(s) affected	0.015 sec
3	03:07:06	Select * from flight LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

3:07 AM 12/14/2017

3. According to booking id this will populate all the information required to see the customer reservation.

/*-----Create Stored Procedure--To View customer record by passing booking id */

DELIMITER \$\$

create procedure viewCustomerRecordByBookingId(param1 INT)

BEGIN

select

c.FirstName,c.LastName,r.ReservationID,r.ReservationDate,r.TicketTypeId,r.FlightID,s.SeatNumber,s.Seat
Type

from customer as c

inner join reservation as r

on

c.CustomerId = r.CustomerId

inner join seatbooked as s on

r.ReservationID = s.Reservation_ReservationID

where c.CustomerId = param1;

END\$\$

delimiter ;

call viewCustomerRecordByBookingId(2142);

Project Name: Airline Reservation System.

The screenshot shows the MySQL Workbench interface. The SQL Editor contains a query that calls a stored procedure `viewCustomerRecordByBookingId(2142)`. The Results window displays a table with the following data:

FirstName	LastName	ReservationID	ReservationDate	TicketTypeID	FlightID	SeatNumber	SeatType
Siddhesh	Kuvelar	2001	2017-10-19 12:23:00	10	KL-360	23	Economy
Siddhesh	Kuvelar	2001	2017-10-19 12:23:00	10	KL-360	28	Economy

The Output window shows the execution of the procedure, indicating that 2 rows were returned.

4.This procedure will tell the customer about the first available plane.

/*-----Find First Available Plane-----*/

DELIMITER \$\$

create procedure firstAvailableFlight(param datetime)

BEGIN

SELECT flight.DepartureTime as 'FirstAvailableFlight' ,flight.Source,flight.Destination from flight

WHERE flight.DepartureTime > param

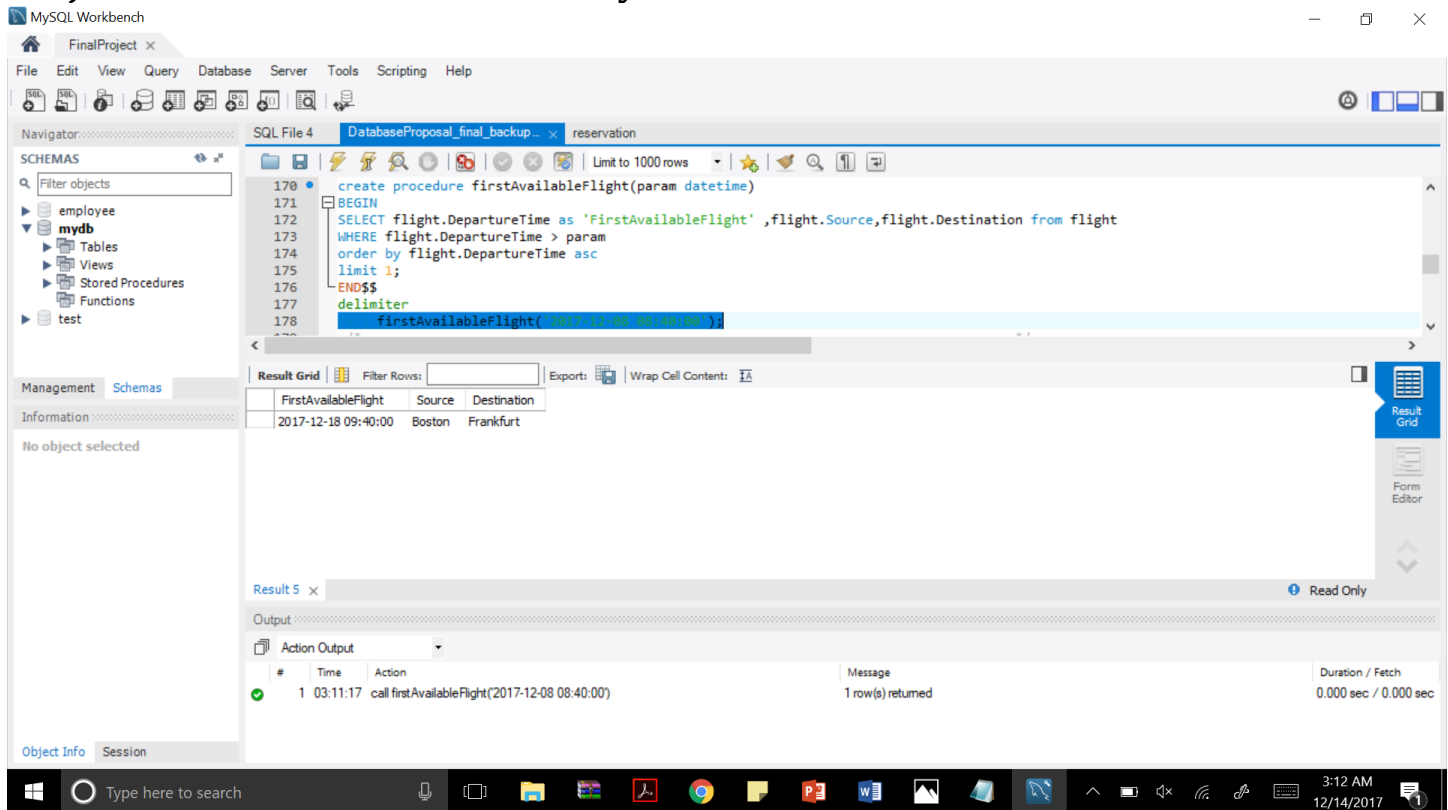
order by flight.DepartureTime asc

limit 1;

END\$\$

Delimiter

Project Name: Airline Reservation System.



Stored Triggers:

1.

/*-----Triggers -----*/

/*-----If Total seat capacity for the airplane is booked User should not able to Book Ticket*/

DROP TRIGGER example_before_book_seat;

select * from flight;

select * from reservation;

delete from reservation where ReservationID = 2015;

INSERT INTO reservation(reservation.ReservationID ,reservation.ReservationDate

,reservation.CustomerId,reservation.FlightID,reservation.TicketTypeID,reservation.SeatBooked) values (2015,'2017-10-24 12:23:00',2142,'KL-713',11,3);

DELIMITER \$\$

CREATE TRIGGER example_before_book_seat

BEFORE INSERT ON reservation FOR EACH ROW

BEGIN

DECLARE xid INT;

declare yid int;

select TotalSeatBooked,TotalSeatCapacity from flight where flight.FlightID = new.flightid into xid,yid;

Project Name: Airline Reservation System.

```
if(yid < xid + new.SeatBooked)
```

```
THEN
```

```
    SIGNAL SQLSTATE '45000'
```

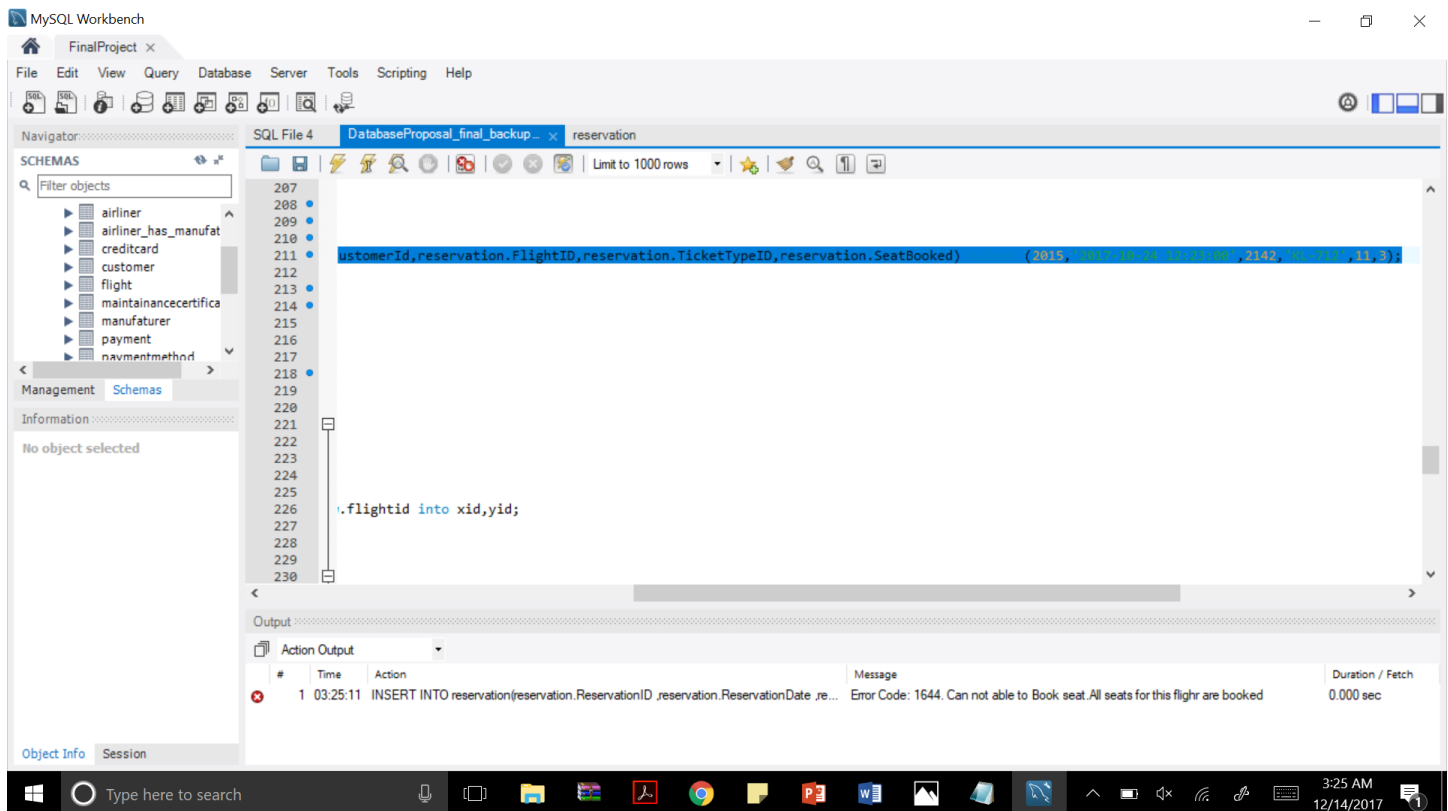
```
    SET MESSAGE_TEXT = 'Can not able to Book seat.All seats for this flighr are booked';
```

```
END IF;
```

```
END;
```

```
$$
```

```
/*-----End-----*/
```



```
2.
select * from flight;
select * from reservation;
INSERT INTO reservation(reservation.ReservationID ,reservation.ReservationDate
,reservation.CustomerId,reservation.FlightID,reservation.TicketTypeID,reservation.SeatBooked) values
(2015 ,'2017-10-24 12:23:00',2142,'KL-713',11,15);
drop trigger example_increment_by_one;
/*-----Triggers-----iNCREMENT FLIGHTSEATS BY SEAT BOOKED -----*/
```

```
DELIMITER $$
```

```
CREATE TRIGGER example_increment_by_one
after INSERT ON reservation FOR EACH ROW
```

```
BEGIN
```

```
    DECLARE xid INT;
```

```
    declare yid int;
```

Project Name: Airline Reservation System.

```
declare zid Int;  
select TotalSeatBooked,TotalSeatCapacity from flight where flight.FlightID = new.flightid into  
xid,yid;
```

```
if(yid > xid + new.SeatBooked)
```

```
THEN
```

```
UPDATE flight
```

```
SET TotalSeatBooked = xid + NEW.SeatBooked
```

```
WHERE flight.FlightID = NEW.flightid ;
```

```
END IF;
```

```
END;
```

\$\$

1. First

The screenshot displays the MySQL Workbench interface. The SQL Editor window shows a script with the following content:

```
/*-----End-----*/  
select * from reservation;  
INSERT INTO reservation(reservation.ReservationID ,reservation.ReservationDate ,reservation.CustomerId,reservation.FlightID,reservation.T:  
drop trigger example_increment_by_one;  
/*-----Triggers-----INCREMENT FLIGHTSEATS BY SEAT BOOKED -----*/  
DELIMITER $$  
CREATE TRIGGER example_increment_by_one  
after INSERT ON reservation FOR EACH ROW
```

The Result Grid window shows a table with the following data:

FlightID	Destination	Source	DepartureTime	ArrivalTime	CertificateId	AirlineId	TotalSeatCapacity	Price	TotalSeatBooked
AF-360	Boston	Dallas	2017-12-24 21:40:00	2017-12-25 01:23:00	1	786	44	476	48
AI-456	Surat	Chicaco	2018-01-07 21:40:00	2018-01-08 01:23:00	0	273	55	523	0
EM-360	London	Newark	2017-12-08 08:40:00	2017-12-09 18:23:00	1	747	75	400	30
EM-549	Frankfurt	Boston	2017-12-18 09:40:00	2017-12-19 12:23:00	1	747	50	546	32
KL-360	London	Newark	2017-12-28 10:40:00	2017-12-28 18:23:00	1	321	100	800	30
KL-712	Memphis	Boston	2017-12-20 06:40:00	2017-12-20 11:45:00	1	321	75	448	75
KL-713	London	Newark	2017-12-18 18:40:00	2017-12-19 18:23:00	1	321	65	400	63
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

The bottom status bar shows the system time as 3:28 AM on 12/14/2017.

Project Name: Airline Reservation System.

2.

The screenshot shows the MySQL Workbench interface. The SQL Editor at the top contains a script with a comment, a SELECT statement, an INSERT statement, a DROP TRIGGER statement, a DELIMITER statement, and a CREATE TRIGGER statement. The Results window below shows a table with 10 columns: FlightID, Destination, Source, DepartureTime, ArrivalTime, CertificateId, AirlinerId, TotalSeatCapacity, Price, and TotalSeatBooked. The table contains 10 rows of data. The Action Output window at the bottom shows the execution of the script, with messages indicating the number of rows affected and returned.

```
/*-----Delete Booking-----*/
DELIMITER $$
CREATE TRIGGER delete_booking
after delete ON reservation FOR EACH ROW

BEGIN
    DECLARE xid INT;
    declare yid int;
    declare zid Int;
    select TotalSeatBooked,TotalSeatCapacity from flight where flight.FlightID = old.flightid into
    xid,yid;
```

FlightID	Destination	Source	DepartureTime	ArrivalTime	CertificateId	AirlinerId	TotalSeatCapacity	Price	TotalSeatBooked
AF-360	Boston	Dallas	2017-12-24 21:40:00	2017-12-25 01:23:00	1	786	44	476	48
AI-456	Surat	Chicago	2018-01-07 21:40:00	2018-01-08 01:23:00	0	273	55	523	0
EM-360	London	Newark	2017-12-08 08:40:00	2017-12-09 18:23:00	1	747	75	400	30
EM-549	Frankfurt	Boston	2017-12-18 09:40:00	2017-12-19 12:23:00	1	747	50	546	32
KL-360	London	Newark	2017-12-28 10:40:00	2017-12-28 18:23:00	1	321	100	800	30
KL-712	Memphis	Boston	2017-12-20 06:40:00	2017-12-20 11:45:00	1	321	75	448	75
KL-713	London	Newark	2017-12-18 18:40:00	2017-12-19 18:23:00	1	321	65	400	64
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

#	Time	Action	Message	Duration / Fetch
1	03:29:56	INSERT INTO reservation(reservation.ReservationID ,reservation.ReservationDate ,reservation.CustomerId,reservation.FlightID,reservation.TicketTypeID,reservation.SeatBooked) values (2014 ,	Error Code: 1062. Duplicate entry '2015' for key 'PRIMARY'	0.000 sec
2	03:30:09	INSERT INTO reservation(reservation.ReservationID ,reservation.ReservationDate ,reservation.CustomerId,reservation.FlightID,reservation.TicketTypeID,reservation.SeatBooked) values (2014 ,	1 row(s) affected	0.000 sec
3	03:30:19	select * from flight LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec

3 To delete the seat

```
select * from flight;
```

```
select * from reservation;
```

```
delete from reservation where ReservationID = 2014;
```

```
INSERT INTO reservation(reservation.ReservationID ,reservation.ReservationDate
```

```
,reservation.CustomerId,reservation.FlightID,reservation.TicketTypeID,reservation.SeatBooked) values
```

```
(2014 , '2017-10-24 12:23:00',2142,'KL-713',11,4);
```

```
drop trigger delete_booking;
```

```
/*-----Delete Booking-----*/
```

```
DELIMITER $$
```

```
CREATE TRIGGER delete_booking
```

```
after delete ON reservation FOR EACH ROW
```

```
BEGIN
```

```
    DECLARE xid INT;
```

```
    declare yid int;
```

```
    declare zid Int;
```

```
    select TotalSeatBooked,TotalSeatCapacity from flight where flight.FlightID = old.flightid into
    xid,yid;
```

```
UPDATE flight
```

Project Name: Airline Reservation System.

SET TotalSeatBooked = xid - old.SeatBooked

WHERE flight.FlightID = old.flightid ;

END;

\$\$

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
select * from flight;
delete from reservation where ReservationID = 2014;
INSERT INTO reservation(reservation.ReservationID ,reservation.ReservationDate ,reservation.CustomerId,reservation.FlightID,reservation.TotalSeatBooked)
drop trigger delete_booking;
```

The Result Grid displays the following data:

FlightID	Destination	Source	DepartureTime	ArrivalTime	CertificateId	AirlineId	TotalSeatCapacity	Price	TotalSeatBooked
AF-360	Boston	Dallas	2017-12-24 21:40:00	2017-12-25 01:23:00	1	786	44	476	48
AI-456	Surat	Chicago	2018-01-07 21:40:00	2018-01-08 01:23:00	0	273	55	523	0
EM-360	London	Newark	2017-12-08 08:40:00	2017-12-09 18:23:00	1	747	75	400	30
EM-549	Frankfurt	Boston	2017-12-18 09:40:00	2017-12-19 12:23:00	1	50	50	546	32
KL-360	London	Newark	2017-12-28 10:40:00	2017-12-28 18:23:00	1	321	100	800	30
KL-712	Memphis	Boston	2017-12-20 06:40:00	2017-12-20 11:45:00	1	321	75	448	75
KL-713	London	Newark	2017-12-18 18:40:00	2017-12-19 18:23:00	1	321	65	400	64

The Output pane shows the following actions:

#	Time	Action	Message	Duration / Fetch
1	03:34:06	CREATE TRIGGER delete_booking after delete ON reservation FOR EACH ROW...	0 row(s) affected	0.063 sec
2	03:34:19	delete from reservation where ReservationID = 2014	0 row(s) affected	0.000 sec
3	03:34:25	select * from flight LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec

Views:

CREATE VIEW bookingViewInformationTable AS

SELECT

c.CustomerId,c.FirstName,

r.ReservationID, r.ReservationDate, r.FlightID, p.PaymentId,p.PaymentDate,p.PaymentMethodId

FROM

customer as c

INNER JOIN reservation as r ON c.CustomerId = r.CustomerId

INNER JOIN payment as p ON r.ReservationID = p.ReservationID

WHERE

p.PaymentAmount > 500;

Project Name: Airline Reservation System.

MySQL Workbench

FinalProject x

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS

Filter objects

mydb

Tables

- address
- airliner
- airliner_has_manufat
- creditcard
- customer
- flight
- maintenancecertifica

Management Schemas

Information

No object selected

SQL File 4 DatabaseProposal_final_backup... reservation

Limit to 1000 rows

Find

customer as c
INNER JOIN reservation as r ON c.CustomerId = r.CustomerId
INNER JOIN payment as p ON r.ReservationID = p.ReservationID
WHERE
p.PaymentAmount > 500;
* bookingViewInformationTable;

Result Grid

CustomerId	FirstName	ReservationID	ReservationDate	FlightID	PaymentId	PaymentDate	PaymentMethodId
2142	Siddhesh	2001	2017-10-19 12:23:00	KL-360	6000	2017-10-19 10:40:00	52
2147	Ritesh	2002	2017-10-20 12:23:00	KL-712	6001	2017-10-20 10:40:00	52

bookingViewInformationTable 9 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
2	03:34:19	delete from reservation where ReservationID = 2014	0 row(s) affected	0.000 sec
3	03:34:25	select * from flight LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec
4	03:44:37	select * from bookingViewInformationTable LIMIT 0, 1000	2 row(s) returned	0.016 sec / 0.000 sec

Object Info Session

3:44 AM
12/14/2017

Privileges :

create user 'reservationAdmin'@'localhost' identified by 'reservationAdmin';
grant select,update,delete on reservation to 'reservationAdmin'@'localhost';