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,
  `Mobileno` 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,
```

```
  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:

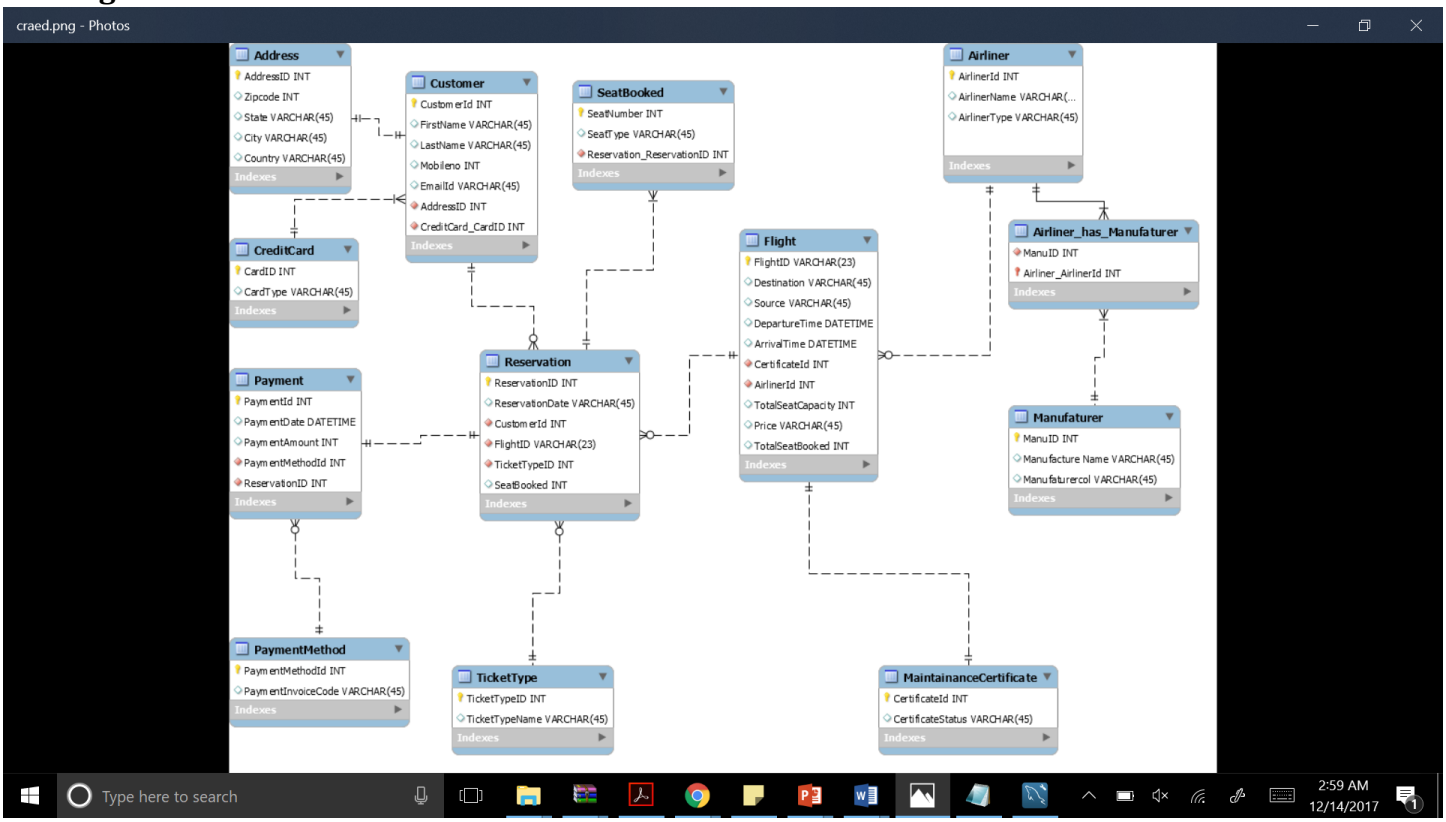
```
DROP TABLE IF EXISTS `seatbooked`;
CREATE TABLE `seatbooked` (
  `SeatNumber` int(11) NOT NULL,
  `SeatType` varchar(45) DEFAULT NULL,
  `Reservation_ReservationID` int(11) NOT NULL,
  PRIMARY KEY (`SeatNumber`),
  KEY `fk_SeatBooked_Reservation1_idx` (`Reservation_ReservationID`),
  CONSTRAINT `fk_SeatBooked_Reservation1` 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 :



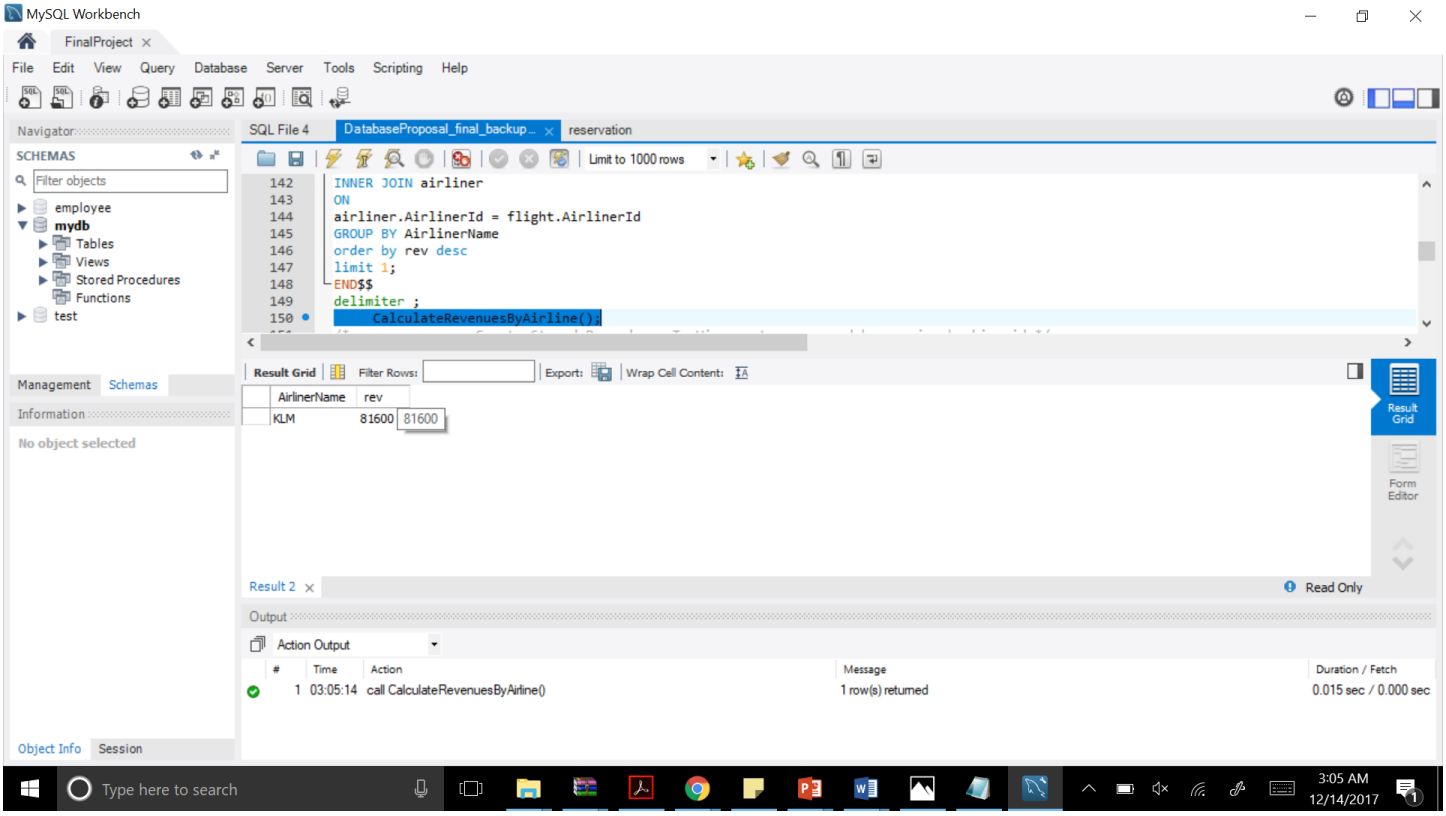
Stored procedures:

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.

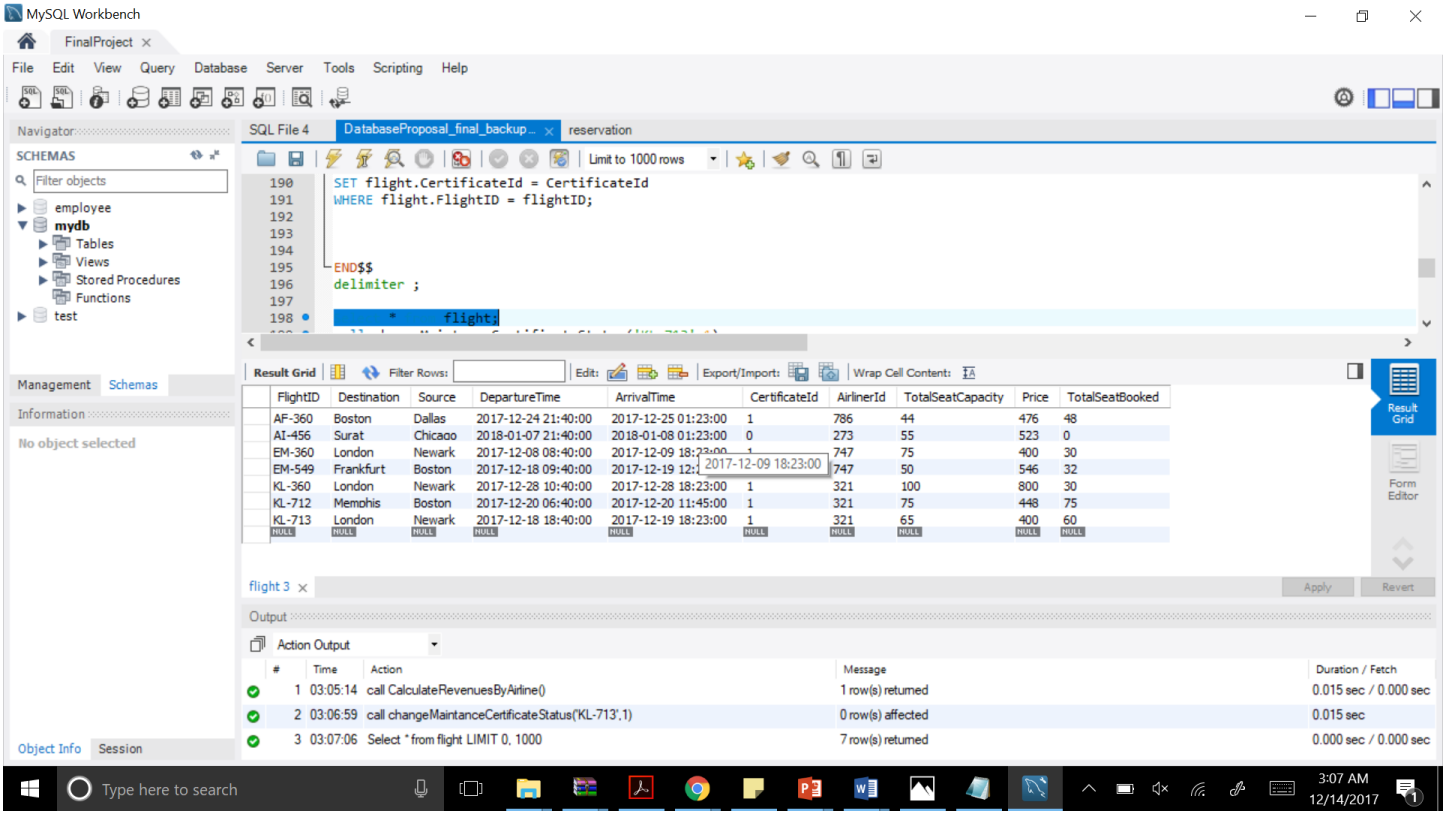


2. This will enables the maintaince certificate of airliner

/* ----------------Change Maintaince Certificate ------*/

```
DELIMITER $$
create procedure changeMaintanceCertificateStatus(flightID varchar(30),CertificateId INT)

BEGIN
UPDATE flight
SET flight.CertificateId = CertificateId
WHERE flight.FlightID = flightID;
END$$
delimiter ;

Select * from flight;
call changeMaintanceCertificateStatus('KL-713',1);
```

Project Name: Airline Reservation System.

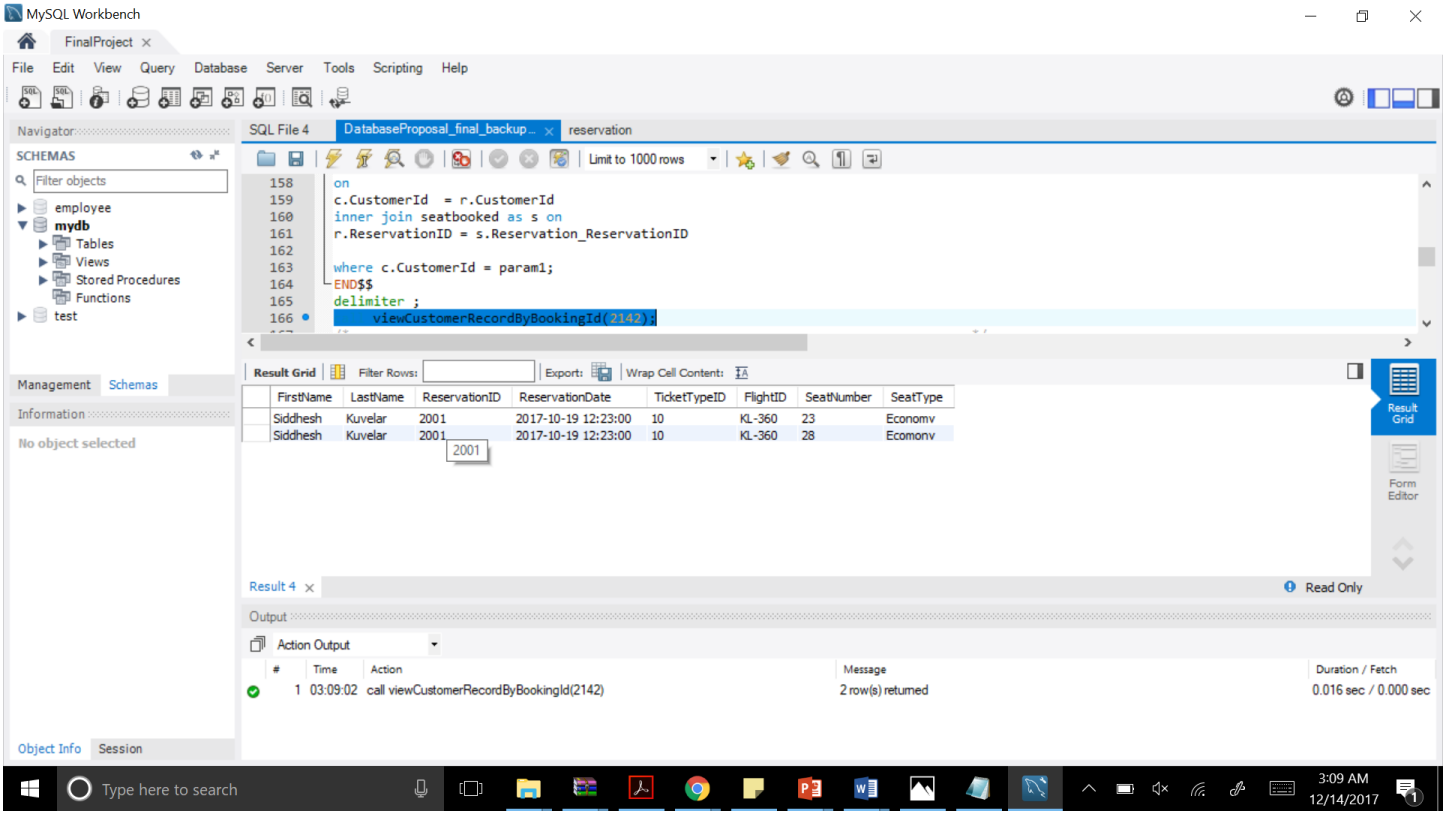


3. According to booking id this will populate all the information required to see the customer reservation.

/*----------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.

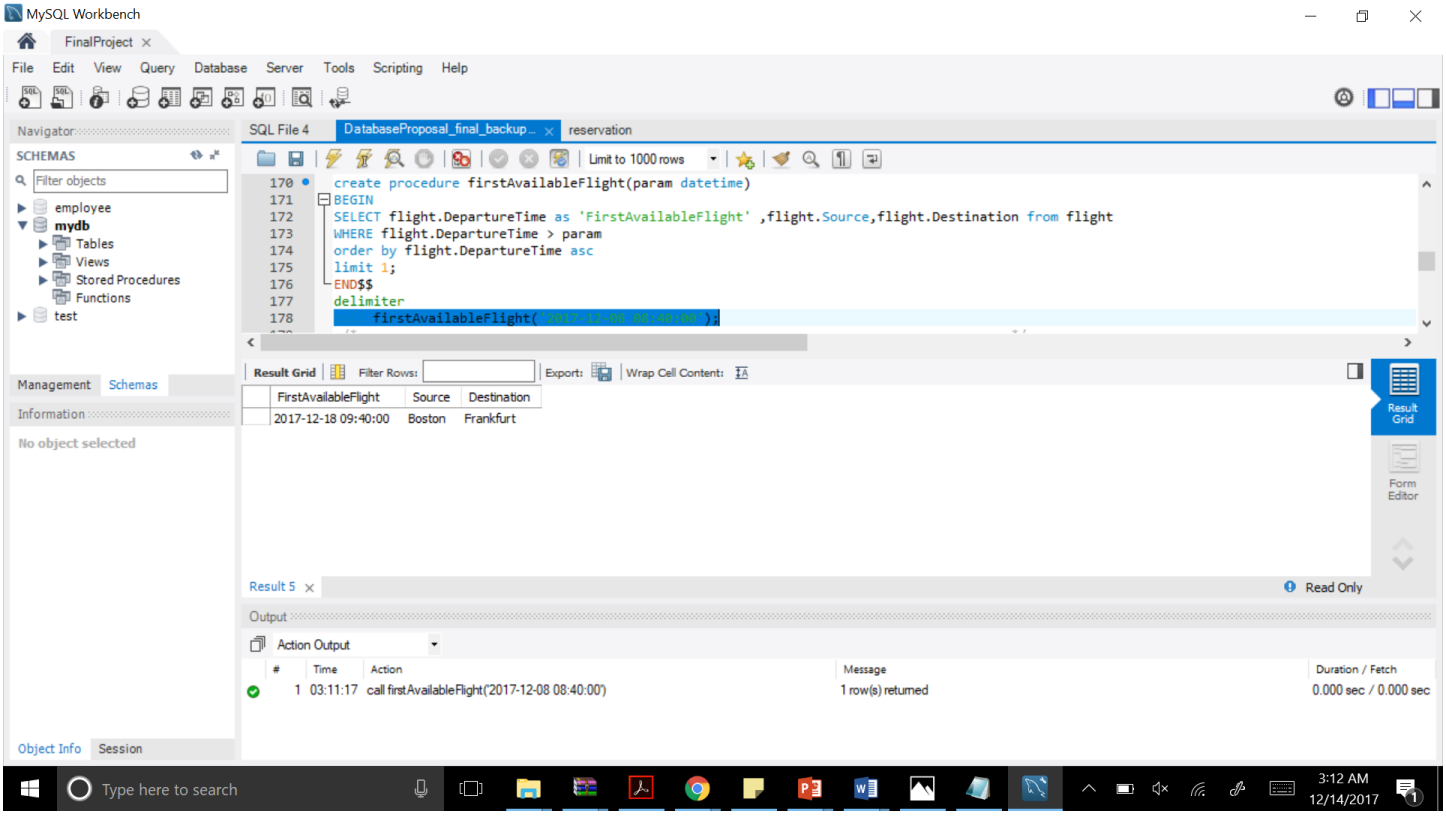


4.This procedure will tell the customer about the first available plane.

/*----------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.

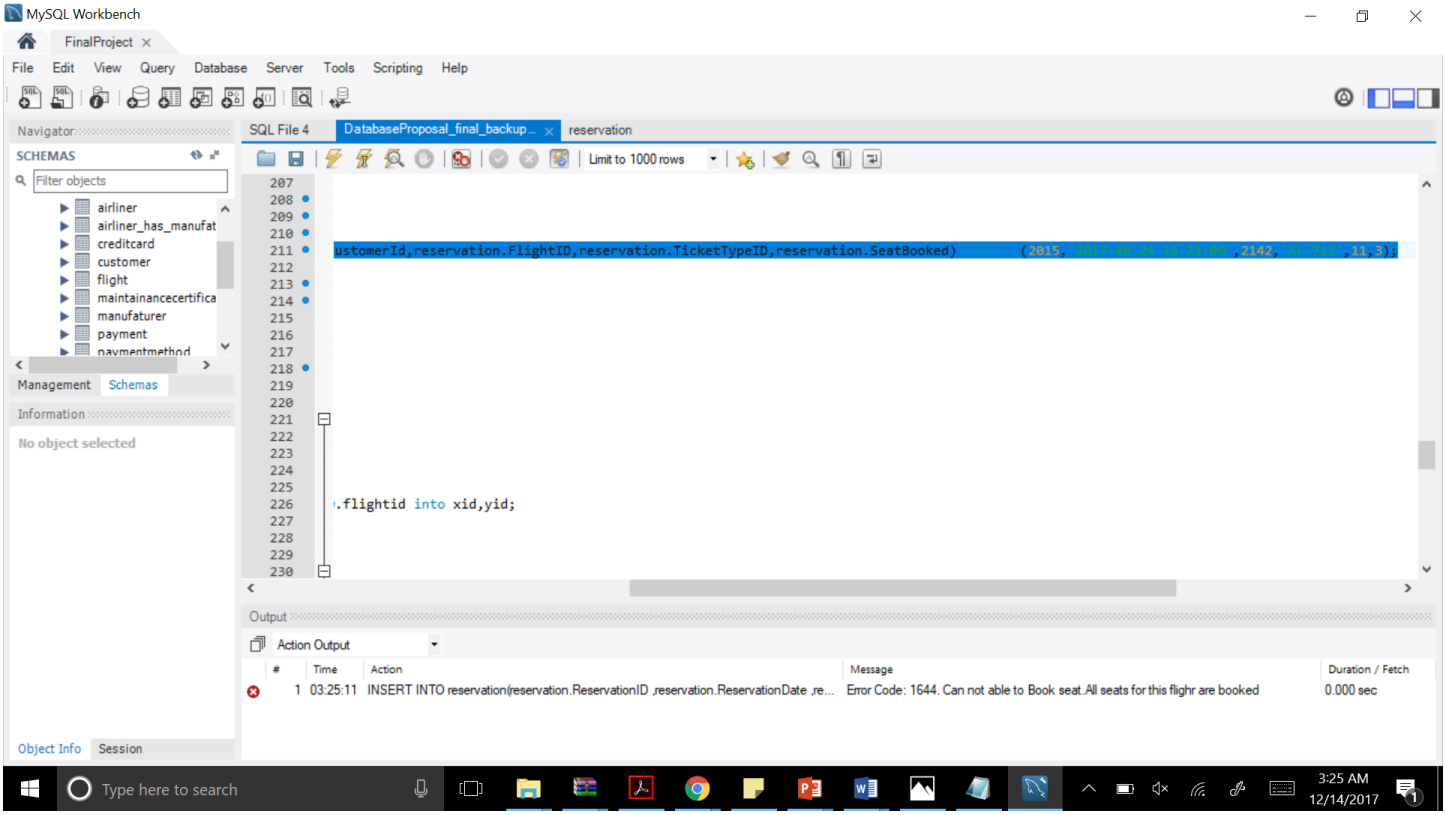
/*----------Triggers ----------*/

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



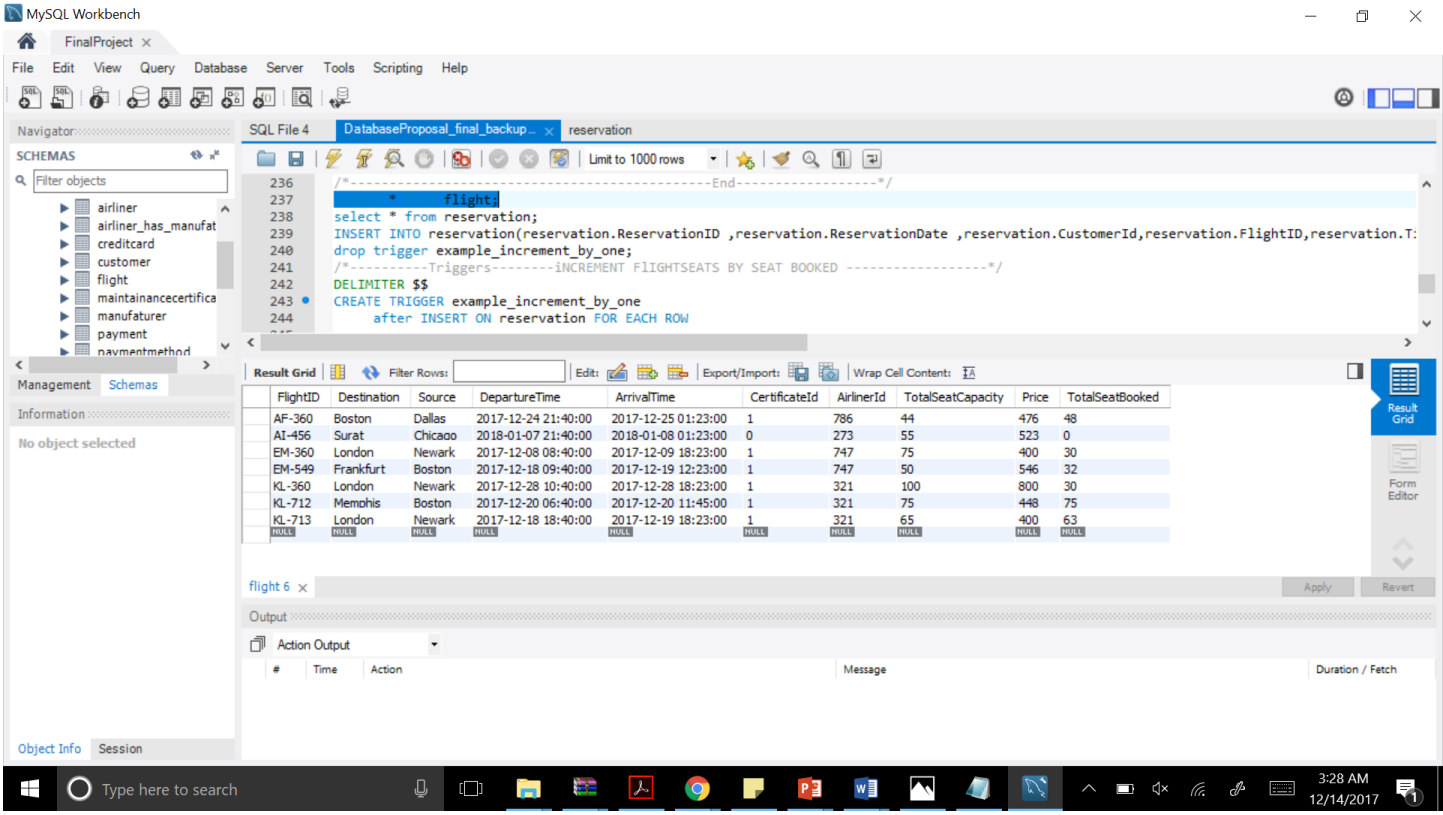
2.

```
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.

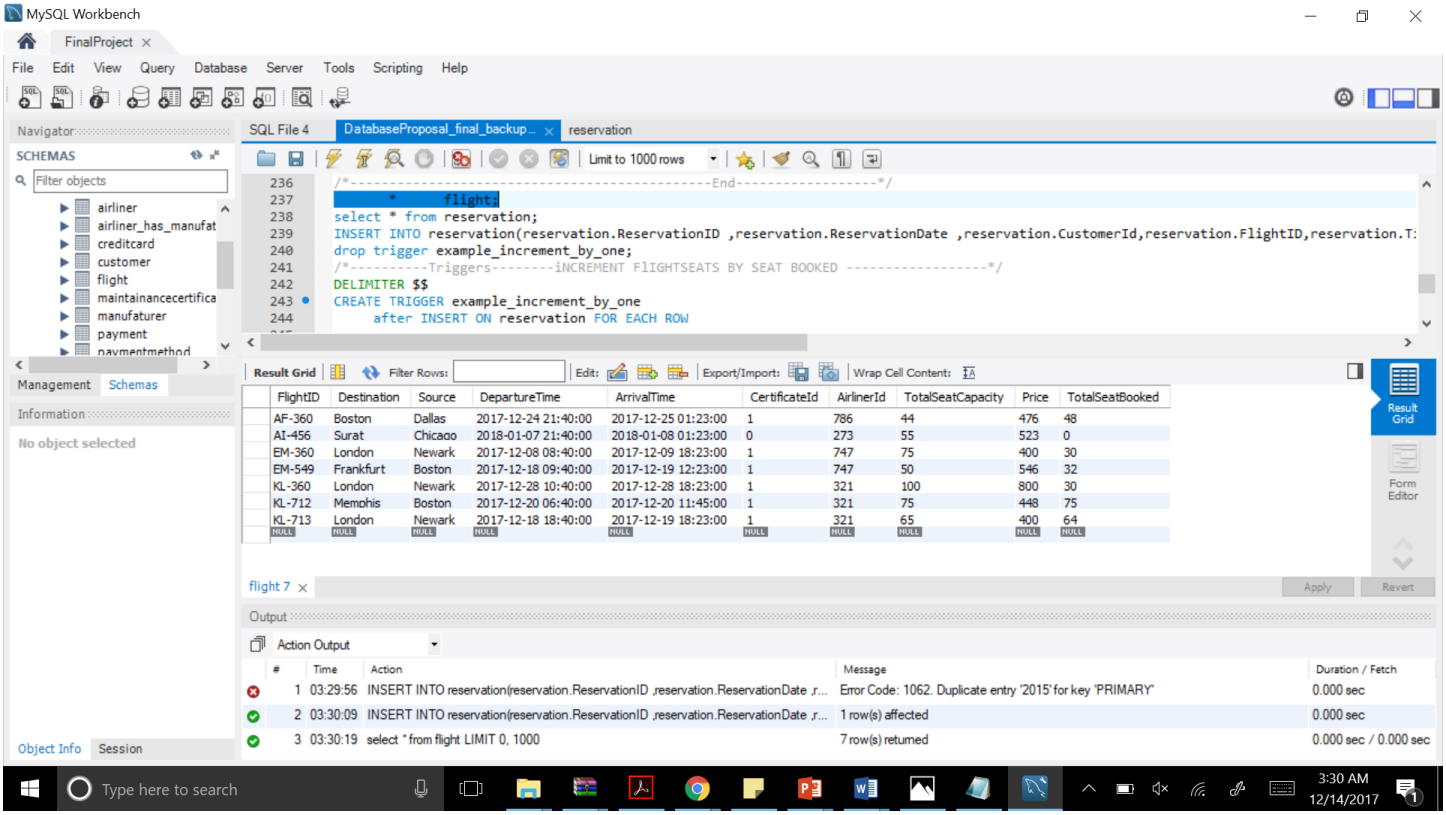
```
                declare zid Int;
        select TotalSeatBooked,TotalSeatCapacity  from flight  where flight.FlightID  = new.flightid into
xid,yid;


      if(yid > xid + new.SeatBooked)

        THEN
            UPDATE flight

       SET TotalSeatBooked = xid + NEW.SeatBooked
       WHERE flight.FlightID = NEW.flightid ;

        END IF;

    END;
$$
```

1. **First**



Project Name: Airline Reservation System.

2.

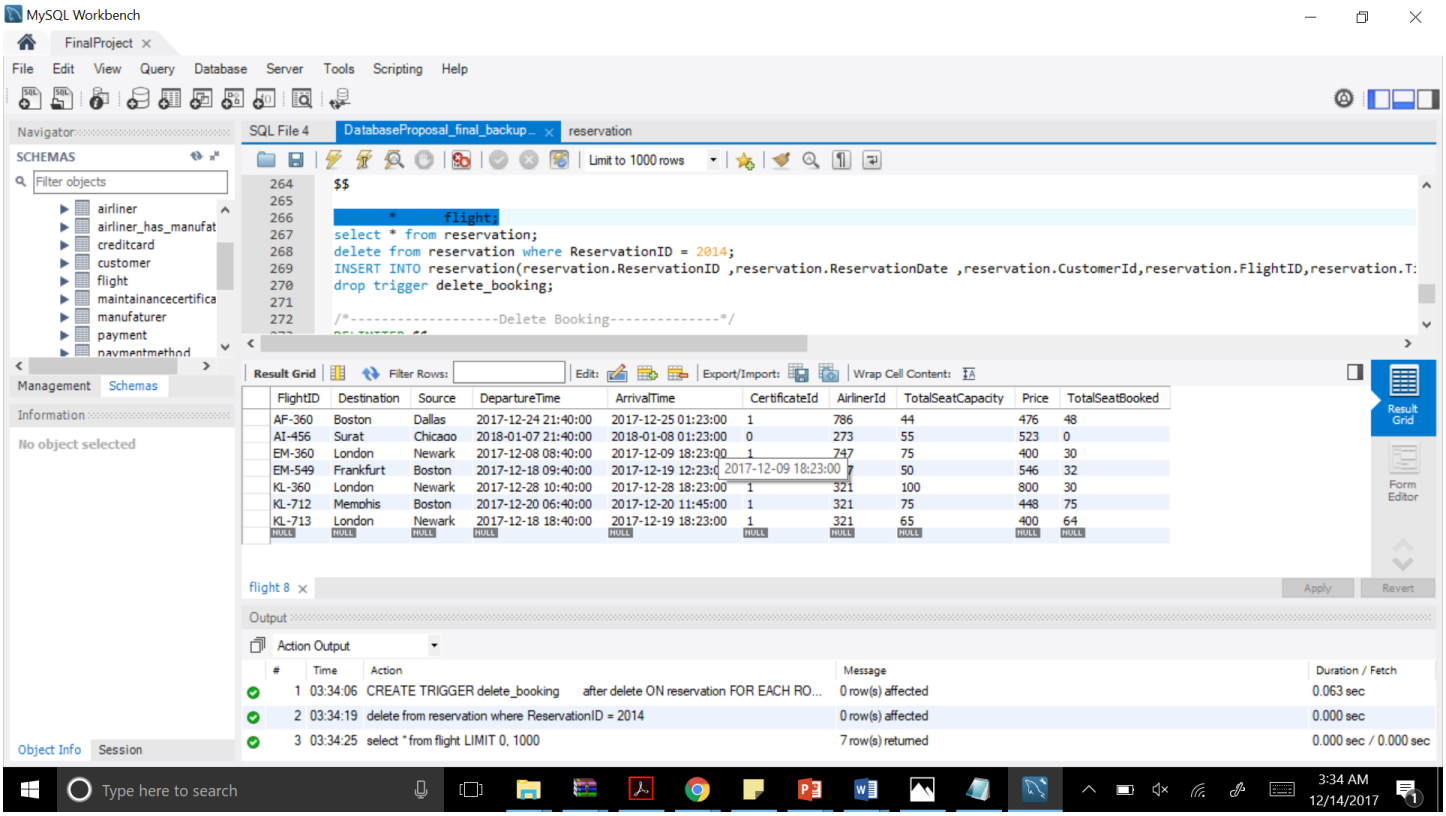


3 To delete the seat

```
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.

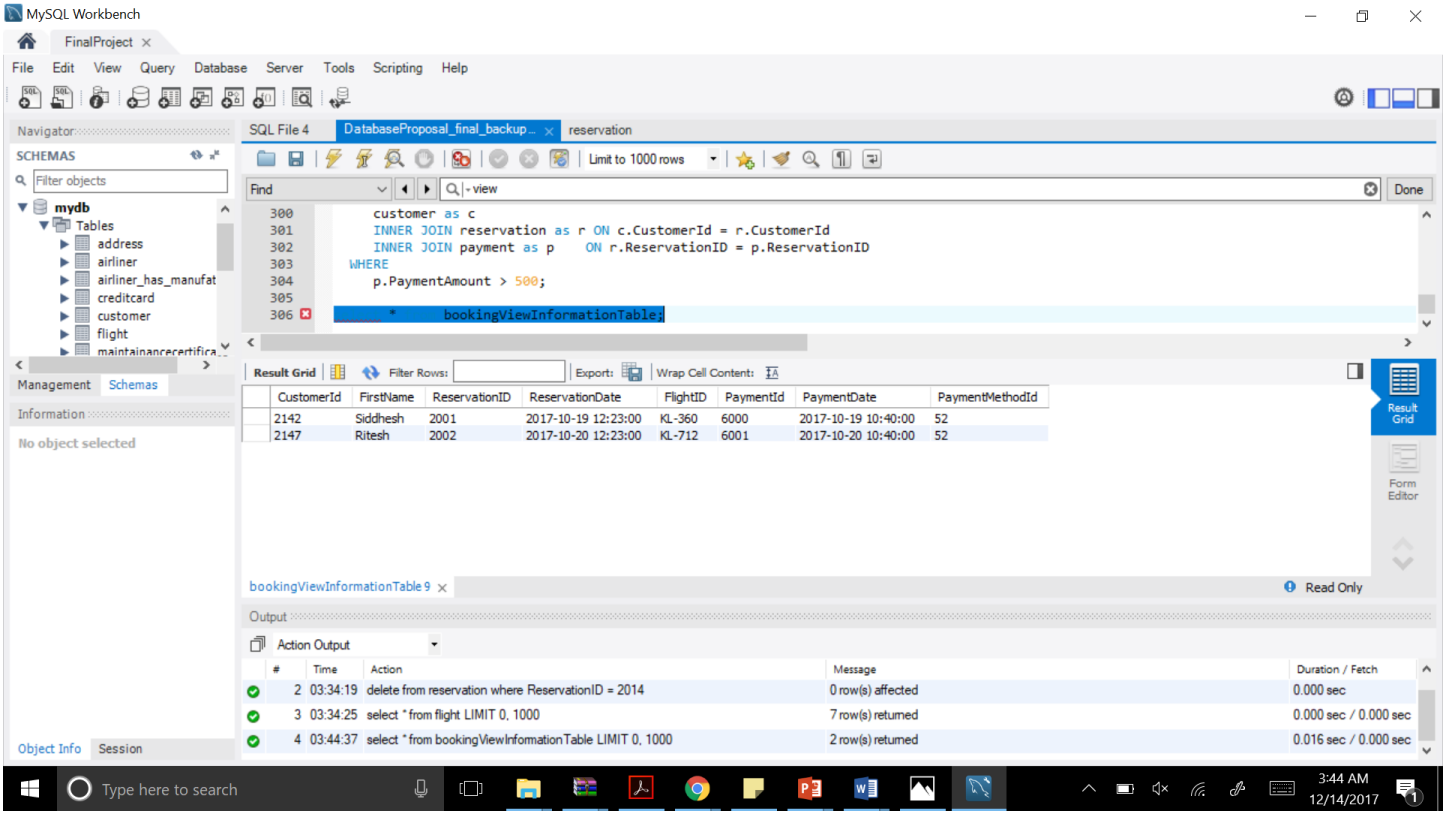
```
    SET TotalSeatBooked = xid - old.SeatBooked
    WHERE flight.FlightID = old.flightid ;


  END;
$$
```



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.



Privileges :

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