DBMS Project Report

PES University

Database Management Systems

UE18CS252

Submitted By

SRN : PES1201801143	Name : Sruthy S	Evaluation date and time:

Functional Dependencies **2**

Identifying Keys based on FDs **2**

Normalization & testing for lossless join property **2+2**

DDL: Table creation with all constraints **2+2**

Triggers **2**

SQL Queries **2**
Viva / modifications (Unit III/ IV concepts)2+

2

AIRLINE RESERVATION SYSTEM

Problem statement:
An airline reservation database needs to store information about the airports it is working on and different routes it can take, it's passenger details(name,address,phNo) , their travel details(ticketNo,flightNo,flightDate) as well as the aircraft (type,number of seats) flight details(from,to,departure and arrival timings and route).Design a database for the same along with the fare variation for the different classes.

DATABASE SCHEMA

Airport
(airport_id(int), airport_name(varchar))

Route
(RouteNo(int),From(varchar),Via(varchar),To(varchar))

Fares
(FareType(varchar), FareDescription(varchar))

Tariff
(RouteNo(int),FareType(varchar),Price(int))

Aircraft
(aid(int),AircraftType(varchar),NoSeats(int))

Flight
(FlightNo(int),*FromAirport(varchar),ToAirport(varchar)*,DepTime(time),ArrTime(time), *aid(int),RouteNo(int)*)

Passenger
(pid(int),Name(varchar),Address(varchar),PhNo(int))

Ticket
(TicketNo(varchar),Date(date),*pid(int)*)

Itinerary
(TicketNo(varchar),*FlightNo(int)*,FlightDate(date),*FareType*(varchar),Amount(int))

*Primary Keys considered are underlined and foreign keys are in *itallics*

Functional Dependencies: (List based on your application constraints)

A functional dependency A->B in a relation holds if two tuples having same value of attribute A also have same value for attribute B.

1. Table: Airport (airport_id,aname)
Airport_id -> Aname
(Every airport id has only one name ie.it's own name and can never possess two names)
Aname -> Airport_id
Attribute closure: (Airport_id)+={Airport_id,Aname}
(Aname)+={Airport_id,Aname}
Since both airport_id and aname are superkeys(functionally gives every other attribute in the table hence both can be candidates for choosing the primary key)
Candidate keys: Airport_id,Aname
Chosen Primary Key: Airport_id

2. Table: Route (RouteNo,From,Via,To)
RouteNo -> From
RouteNo ->Via
RouteNo ->To
(From,Via,to) -> RouteNo
Attribute closure: (RouteNo)+={RouteNo,From,Via,To}
(From,Via,To)+={RouteNo,From,Via,To}
Every route number specifies a definite from, destination and via. Similarly ,every distinct from,to and via corresponds to that route number hence both the aforementioned can act as candidate keys.
Candidate keys: RouteNo,(From,Via,To)
Chosen Primary key: RouteNo

3. Table: Fares(Faretype,FareDescription)
FareType -> FareDesc
FareDesc ->FareType
Attribute closure: (FareType)+={FareType,FareDesc}
(FareDesc)+={FareType,FareDesc}
Candidate Keys: FareType,FareDesc
Chosen Primary Key: FareType

4. Table : Tariff(RouteNo,FareType,Price)
FD: RouteNo,FareType ->Price
Attribute closure : (RouteNo,FareType)+={RouteNo,FareType,Price}
In this table, RouteNo alone cannot decide the price neither can faretype individually.However by combining both,we can get a unique key to access the table.
Price cannot functionally determine route or faretype.
Candidate key:RouteNo,FareType
Chosen Primary Key: RouteNo,FareType

5. Table: Aircraft(aid,AircraftType,NoSeats)
Aid -> AircraftType
Aid -> NoSeats
AircraftType -> NoSeats

Attribute closure: (Aid)+={Aid,AircrftType,NoSeats}
(AircraftType)+={AircraftType, NoSeats}

In this table, there is an unique I given to each aircraft hence aid can determine both aircraft type and seats whereas aircraft cannot functionally determine it's id but can do so for the seats.(As the aircraft type can be the same with different seat capacities)
Candidate Key: aid
Primary Key: aid

6. Table: Flight(FlightNo,FromAirport,ToAirport,DeptTime,ArrTime,aid,RouteNo)
(FlightNo)+={FlightNo,FromAirport,ToAirport,DeptTime,ArrTime,aid,RouteNo}

Candidate Key: FlightNo
Primary Key: FlightNo

7. Table :Passenger(pid,name,address,phNo)
(pid)+={pid,name,address,phNo}
(phNo)+={name,pid,phNo,address}

As we can observe,the unique passenger id and the passenger phone number can functionally determine every other attribute in the table.The candidate and the primary keys are as follows:
Candidate Keys: pid,phNo
Chosen Primary Key: pid

8. Table: Ticket(TicketNo,Date,pid)
(TicketNo)+={TicketNo,Date,pid}
Each pid can have multiple tickets hence pid cannot functionally determine any of the attributes other than itself (due to reflexivity)
Candidate Key: TicketNo
Primary Key: TicketNo

9. Table: Itinerary(TicketNo,FlightNo,FlightDate,FareType,Amount)
(TicketNo)+={TicketNo,FlightNo,FlightDate,FareType,Amount}
Candidate Key: TicketNo
Primary Key : TicketNo

Normalization and testing for lossless join property:

Normalization is the process of minimizing redundancy from a relation or set of relations.

TRANSITION FROM MY INITIAL DATABASE TO THE NORMALIZED VERSION

1. First Normal Form

A relation is in first normal form if and only if the domain of each attribute contains only atomic (indivisible) values, and the value of each attribute contains only a single value from that domain.
Every attribute in the current database design is single valued and hence abides by the rules of the first normal form.

2. Second Normal Form

To be in second normal form, a relation must be in first normal form and relation must not contain any partial dependency.(no non-prime attribute is dependent on any proper subset of any candidate key of the table)

Violation of Second Normal Form

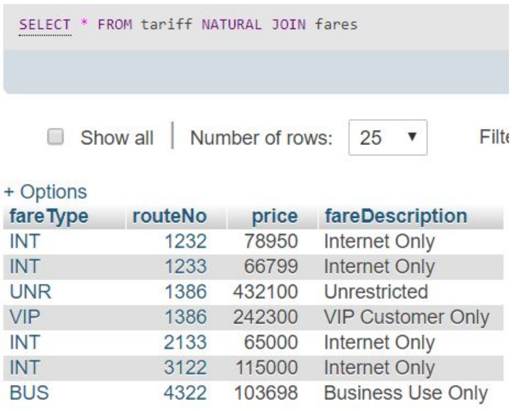
- My initial database consisted all attributes from the current Fares table and the Tariff table under one table named Bill details

BillDetails(RouteNo,FareType,FareDesc,Price)
The primary key in that case would be RouteNo,FareType as both of these combined determine the price.However then,FareDesc, a non prime attribute would have been partially dependent on FareType. Hence to follow the rules of the second normal form,the table was decomposed to Fares and Tariff

3. Third Normal Form

A relation is said to be in third normal form if it satisfies all conditions of second normal form **and** does not house any transitive dependency.As we can clearly observe from the schema,there is no transitive dependency in any of the tables hence third normal form is achieved.

TESTING LOSSLESS JOIN PROPERTY
To test this particular property,I have created a dummy table billdetails(as described above in second normal form) and displayed that along with the natural join of the decomposed tables fares and tariff. If the latter is equal to the former,we say that the join is lossless which can be seen in the images below.

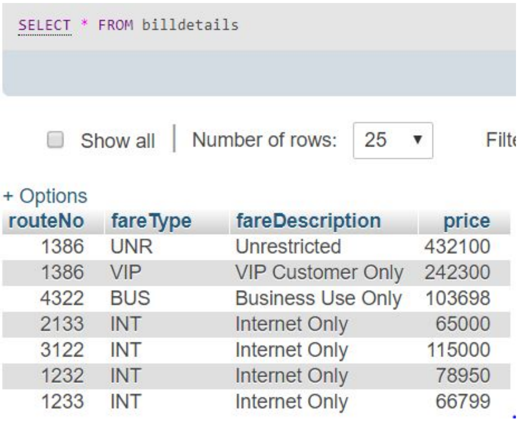


```
SELECT * FROM tariff NATURAL JOIN fares
```

□ Show all | Number of rows: 25 ▾ Filte

+ Options

fareType	routeNo	price	fareDescription
INT	1232	78950	Internet Only
INT	1233	66799	Internet Only
UNR	1386	432100	Unrestricted
VIP	1386	242300	VIP Customer Only
INT	2133	65000	Internet Only
INT	3122	115000	Internet Only
BUS	4322	103698	Business Use Only



```
SELECT * FROM billdetails
```

□ Show all | Number of rows: 25 ▾ Filte

+ Options

routeNo	fareType	fareDescription	price
1386	UNR	Unrestricted	432100
1386	VIP	VIP Customer Only	242300
4322	BUS	Business Use Only	103698
2133	INT	Internet Only	65000
3122	INT	Internet Only	115000
1232	INT	Internet Only	78950
1233	INT	Internet Only	66799

Hence the lossless join property is tested.

TABLE SCRIPTS

```
CREATE TABLE Airport (
   airport_id varchar(3),
   aname varchar(255),
   PRIMARY KEY(airport_id)

);
CREATE TABLE Route(
          routeNo int NOT NULL,
          fromPlace varchar(255),
          via varchar(255),
          toPlace varchar(255),
          PRIMARY KEY(routeNo)
 );
 CREATE TABLE Fares (
   fareType varchar(3),
          fareDescription varchar(255),
   PRIMARY KEY(fareType)
);
CREATE TABLE Tariff (
   routeNo int NOT NULL,
   fareType varchar(3),
          price int,
   PRIMARY KEY(routeNo,fareType),
   FOREIGN KEY (routeNo) REFERENCES Route(routeNo),
   FOREIGN KEY (fareType) REFERENCES Fares(fareType),
          CHECK (price>0)
);
CREATE TABLE Aircraft (
    air_id int NOT NULL,
    atype varchar(255),
    no_seats int,
    PRIMARY KEY(air_id),
    CHECK (no_seats>0)
);
CREATE TABLE Flight (
    flightNo int NOT NULL,
    fromAirport varchar(3),
            toAirport varchar(3),
            depTime time,
            arrTime time,
            air_id int,
            routeNo int,
    PRIMARY KEY(flightNo),
    FOREIGN KEY (routeNo) REFERENCES Route(routeNo),
            FOREIGN KEY (air_id) REFERENCES Aircraft(air_id),
            FOREIGN KEY (fromAirport) REFERENCES Airport(airport_id),
    FOREIGN KEY (toAirport) REFERENCES Airport(airport_id)

);
CREATE TABLE Passenger (
            pid int NOT NULL,
            pName varchar(255),
            pAddress varchar(255),
            phNo varchar(10),
            vip int(1),
            PRIMARY KEY(pid),
            CONSTRAINT chk_phone CHECK (phNo not like '%[^0-9]%'),
            CONSTRAINT chk_vip CHECK (vip=0 or vip=1)

);
CREATE TABLE Ticket (
            ticketNo varchar(255),
            date DATE,
            pid int,
            PRIMARY KEY (ticketNo),
            FOREIGN KEY (pid) REFERENCES Passenger(pid)



);
CREATE TABLE Itinerary (
            ticketNo varchar(255),
            flightNo int,
            flightDate DATE,
            fareType varchar(3),
            amount int,
            PRIMARY KEY (ticketNo),
            FOREIGN KEY (ticketNo) REFERENCES Ticket(ticketNo),
            FOREIGN KEY (fareType) REFERENCES Fares(fareType),
            FOREIGN KEY (flightNo) REFERENCES Flight(flightNo)

);
```

SAMPLE INSERT VALUES

```
INSERT INTO Airport VALUES
('EZE','Ezeiza International Airport Argentina'),
('BAJ','Bali Indonesia'),
('ACY','Atlantic City International USA'),
('MAA','Chennai International India'),
('MAN','Manchester England'),
('MCI','Kansas City International USA'),
('JFK','John F Kennedy International Airport USA'),
('CLT','Charlotte Douglas International Airport USA'),
('BOM','Chatrapathi Sivaji International Airport India'),
('CCU','Netaji Subhash Chandra Bose International Airport India');

INSERT INTO Route VALUES
(1232,'New York','Miami','Buenos Aires'),
(1233,'New York','Santa Cruz','Buenos Aires'),
(1234,'New York',NULL,'Buenos Aires'),
(1386,'Chennai','Paris','North Carolina'),
(2133,'Manchester','Abu Dhabi','Mumbai'),
(4322,'Bali',NULL,'Kansas'),
(3122,'Kansas',NULL,'Kolkata');

INSERT INTO Fares VALUES
('UNR','Unrestricted'),
('BUS','Business Use Only'),
('CPC','Capacity Control'),
('VIP','VIP Customer Only'),
('INT','Internet Only');

INSERT INTO Tariff VALUES
(1386,'UNR',432100),
(1386,'BUS',242300),
(4322,'BUS',103698),
(4322,'INT',96860),
(2133,'INT',65000),
(3122,'INT',115000),
(1232,'INT',78950),
(1233,'INT',66799),
(1234,'INT',102780),
(2133,'BUS',104321);

INSERT INTO Aircraft VALUES
(101, 'BOEING', 450),
(102, 'RAYTHEON', 320),
(103, 'SAFRAN', 270),
(104,'DOUGHLAS',200),
(105,'AIRBUS',300);

INSERT INTO Flight VALUES
(1,'JFK','EZE','21:00:00','11:15:00',101,1232),
(2,'JFK','EZE','23:50:00','14:00:00',103,1233),
(3,'BAJ','MCI','13:30:00','09:45:00',105,4322),
(4,'MAA','CLT','21:00:00','09:15:00',104,1386),
(5,'MAN','BOM','13:00:00','23:00:00',102,2133),
(6,'JFK','EZE','21:00:00','05:30:00',104,1234);

INSERT INTO Passenger VALUES
(1000,'George Minelli','Nor Road,New Jersey','9121002336',0),
(1001,'Ronnie Jackson','Mor Road,Buenos Aires','9007754636',0),
(1002,'Sashi Rao','Dombvilli,Mumbai','9121300036',0),
(1003,'Sarah Tiong','Rose road,New York','6123452345',0),
(1004,'Jess Carter','Apple Road,North Carolina','6324567890',0),
(1005,'Brendan Martin','Annanagar,Chennai','9121364636',0),
(1006,'Harry Potter','Manchester','6210213456',0),
(1007,'Percy Jackson','New York','8234907645',0),
(1008,'Enid Blyton','North Carolina','6129678900',0);

INSERT INTO Ticket VALUES
('TKT1','2015-12-17',1004),
('TKT2','2015-12-14',1005),
('TKT3','2015-12-22',1001),
('TKT4','2015-12-20',1003),
('TKT5','2015-12-21',1006),
('TKT6','2015-12-19',1007),
('TKT7','2015-12-20',1008),
('TKT8','2015-12-24',1005);
```

```
INSERT INTO Itinerary VALUES
('TKT1',1,'2015-12-17','INT',78950),
('TKT2',2,'2015-12-14','INT',66799),
('TKT3',3,'2015-12-22','BUS',103698),
('TKT4',4,'2015-12-20','UNR',432100),
('TKT5',5,'2015-12-21','BUS',104312),
('TKT6',6,'2015-12-19','INT',102780),
('TKT7',4,'2015-12-20','BUS',242300),
('TKT8',3,'2015-12-24','INT',96860);
```

TRIGGER

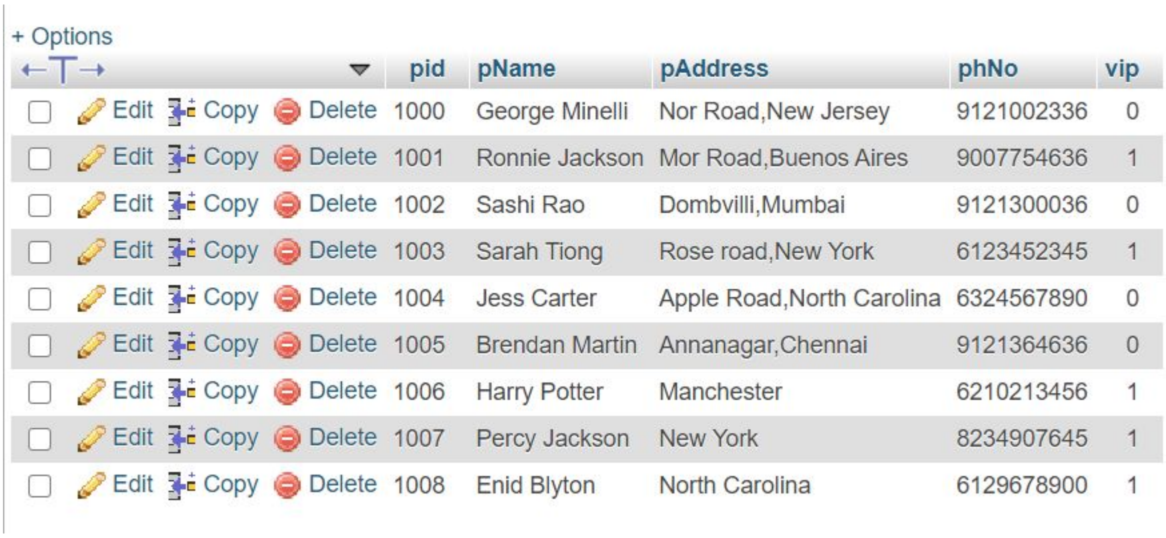
Triggers are event-driven specialized procedures that initiates or fires an action when an event occurs.

In my database, every passenger is not a vip by default (vip=0). However, I want to designate my passengers as vip when they spend more than 100000 with my firm.This is what my trigger does.

The name ofthe designed trigger is update_vip.

Write a trigger to designate passengers as vip when they spend more that 100000 with the company.

Working of the trigger : Initially,no customer is a vip, **on insert** into itinerary,the trigger is fired and the customer who's paid more that 100000 for their ticket becomes a vip.



+ Options

←T→				pid	pName	pAddress	phNo	vip
□	Edit	Copy	Delete	1000	George Minelli	Nor Road,New Jersey	9121002336	0
□	Edit	Copy	Delete	1001	Ronnie Jackson	Mor Road,Buenos Aires	9007754636	1
□	Edit	Copy	Delete	1002	Sashi Rao	Dombvilli,Mumbai	9121300036	0
□	Edit	Copy	Delete	1003	Sarah Tiong	Rose road,New York	6123452345	1
□	Edit	Copy	Delete	1004	Jess Carter	Apple Road,North Carolina	6324567890	0
□	Edit	Copy	Delete	1005	Brendan Martin	Annanagar,Chennai	9121364636	0
□	Edit	Copy	Delete	1006	Harry Potter	Manchester	6210213456	1
□	Edit	Copy	Delete	1007	Percy Jackson	New York	8234907645	1
□	Edit	Copy	Delete	1008	Enid Blyton	North Carolina	6129678900	1

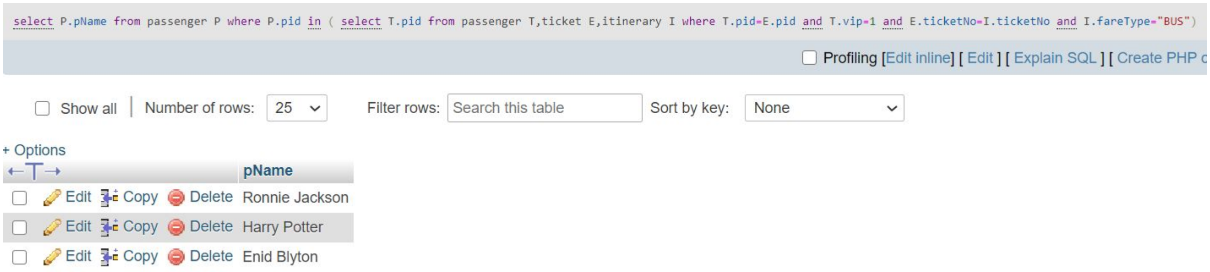
```
DELIMITER $$CREATE TRIGGER update_vip
    AFTER INSERT ON itinerary
    FOR EACH ROW BEGIN
    IF new.amount>100000
    then
        UPDATE passenger,ticket
        SET passenger.vip=1
        WHERE new.ticketNo=ticket.ticketNo and ticket.pid=passenger.pid;
       END IF;
       END;
$$
DELIMITER ;
```

SQL Queries:

1. Display all names of vip passengers travelling by business class.

```
select P.pName
from passenger P
where P.pid in (
            select T.pid from passenger T,ticket E,itinerary I
            where T.pid=E.pid
    and T.vip=1
    and E.ticketNo=I.ticketNo
    and I.fareType="BUS");
```

Output:



```
select P.pName from passenger P where P.pid in ( select T.pid from passenger T,ticket E,itinerary I where T.pid=E.pid and T.vip=1 and E.ticketNo=I.ticketNo and I.fareType="BUS")
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP c

□ Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

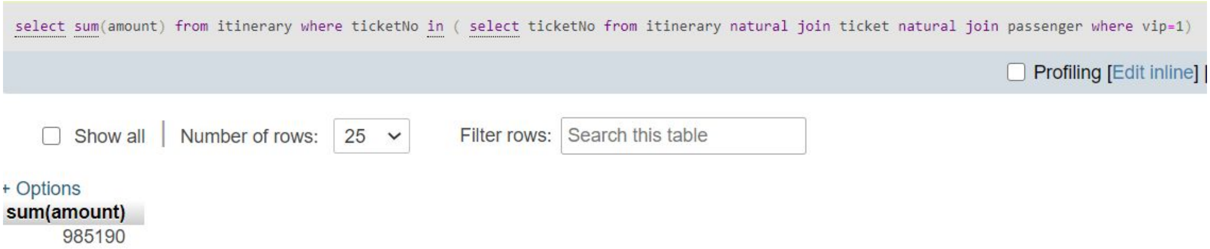
+ Options

←T→	pName
□ Edit Copy Delete	Ronnie Jackson
□ Edit Copy Delete	Harry Potter
□ Edit Copy Delete	Enid Blyton

2. Display the total amount received by the firm from vip customers only

```
select sum(amount)
from itinerary
where ticketNo in (
          select ticketNo from itinerary
          natural join ticket
          natural join passenger
          where vip=1);
```

Output:



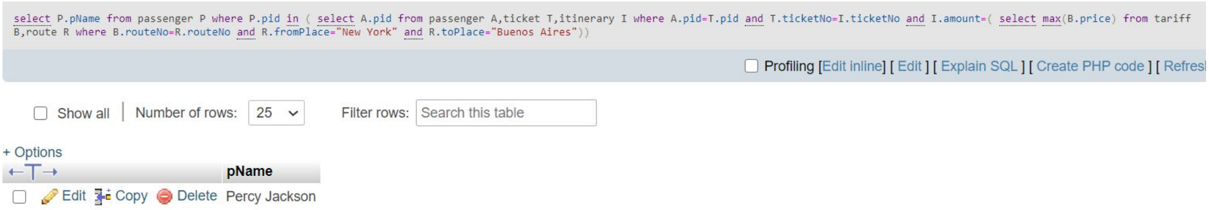
```
select sum(amount) from itinerary where ticketNo in ( select ticketNo from itinerary natural join ticket natural join passenger where vip=1)
```

sum(amount)
985190

3. Display the names of passengers who pay more for theflight from New York to Buenos Aires,and belong to the same class of faretype

```
select P.pName
from passenger P
where P.pid in (
        select A.pid
       from passenger A,ticket T,itinerary I
        where A.pid=T.pid
        and T.ticketNo=I.ticketNo
        and I.amount=(
                select max(B.price)
                from tariff B,route R
                where B.routeNo=R.routeNo
                and R.fromPlace="New York"
                and R.toPlace="Buenos Aires"));
```

Output:



```
select P.pName from passenger P where P.pid in ( select A.pid from passenger A,ticket T,itinerary I where A.pid=T.pid and T.ticketNo=I.ticketNo and I.amount=( select max(B.price) from tariff
B,route R where B.routeNo=R.routeNo and R.fromPlace="New York" and R.toPlace="Buenos Aires"))
```

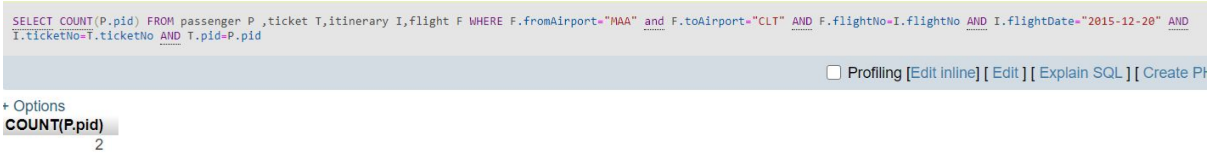
☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refres
☐ Show all
Number of rows: 25
Filter rows: Search this table
+ Options
←T→

	pName
☐ Edit Copy Delete	Percy Jackson

4. Count and return the number of passengers travelling from MAA to CLT on 20/12/2015

```
SELECT COUNT(P.pid)
FROM passenger P ,ticket T,itinerary I,flight F
WHERE F.fromAirport="MAA" and F.toAirport="CLT"
AND F.flightNo=I.flightNo
AND I.flightDate="2015-12-20"
AND I.ticketNo=T.ticketNo
AND T.pid=P.pid;
```

Output:



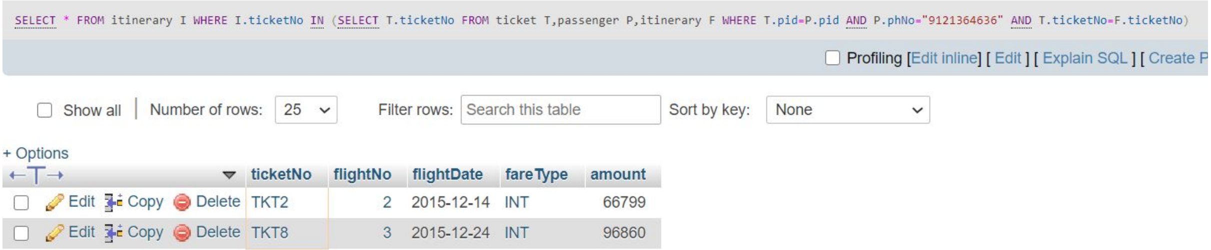
```
SELECT COUNT(P.pid) FROM passenger P ,ticket T,itinerary I,flight F WHERE F.fromAirport="MAA" and F.toAirport="CLT" AND F.flightNo=I.flightNo AND I.flightDate="2015-12-20" AND
I.ticketNo=T.ticketNo AND T.pid=P.pid
```

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PH
+ Options

COUNT(P.pid)
2

5. Given a person’s phone number, find his/her travel itineraries

```
SELECT * FROM itinerary I
WHERE I.ticketNo IN
          (SELECT T.ticketNo
    FROM ticket T,passenger P,itinerary F
    WHERE T.pid=P.pid AND P.phNo="9121364636" AND T.ticketNo=F.ticketNo);
```



```
SELECT * FROM itinerary I WHERE I.ticketNo IN (SELECT T.ticketNo FROM ticket T,passenger P,itinerary F WHERE T.pid=P.pid AND P.phNo="9121364636" AND T.ticketNo=F.ticketNo)
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create P
□ Show all
Number of rows: 25
Filter rows: Search this table
Sort by key: None
+ Options
Edit
Copy
Delete

ticketNo	flightNo	flightDate	fareType	amount
TKT2	2	2015-12-14	INT	66799
TKT8	3	2015-12-24	INT	96860