Database Design Term Project

Phase 4

CS 6360

Group 20

Anuj Patil - AAP190027

Omkar Poyekar - ODP200000

Poojan Patel – PRP200000

Project Description

Dallas Area Road Transport or DART would like one relational database to store the information about their bus transportation system to be able to carry out their work in an organized way. The DART has some major modules such as Bus, Person (Employee and Passenger) and Ticket Sales. A Person can be an Employee or an A-class Passenger.

A person can be both an employee and an A-Class passenger. Details of a person such as Person ID, Name (First, Middle, Last), Address, Gender, Date of Birth (Must be 16 years or older), and Phone number (one person can have more than one phone number) are recorded. The Person ID should have the format "PXXX" where X is a number from 0 to 9. (Hint: you can use regexp_like() function).

Employee can only be one of following three types: classified as Bus Drivers, Staff (sells tickets or passes) and Ticket checkers. The start date of the employee is recorded. One bus driver can drive multiple buses and multiple drivers can drive one bus but on different dates. But for each day, a driver can only drive one particular bus. One bus will always have one particular ticket checker.

Payment information such as ID, method (cash or card), amount and other information are recorded. Ticket details such as Ticket ID, Bus ID, seat number and price are stored. The staff sells daily tickets to a person and the staff details, ticket details, person details and payment details are stored.

An A-Star passenger is someone who has some extra privileges than an A-Class passenger. An A-Star Passenger can be an Employee or an A-Class passenger or both. Different passes are issued by DART. An A-Class passenger can buy only one pass in a month, but an A-Star Passenger can buy multiple passes in a month.

Each A-Star Passenger is issued a travel card. The travel card details such as card ID, date of issue and other information are stored. Card ID is not unique, and is associated with A-Star Passenger.

Sometimes promotional discounts are offered on the travel cards and details such promotion ID and promotion description are recorded. The Promotional IDs are not unique and different travel cards may have discounts with the same Promotional IDs.

A-Star passengers can have guests who travel for free with them four times a month. A Guest info log is maintained which stores information such as passenger ID, guest ID, guest name, guest address, and guest contact information. Guest IDs are temporary IDs that a person gets when they travel as a guest of an A-Star passenger. Each guest ID is not unique and cannot be used to identify a guest.

Bus details such as Bus Number, License plate number, number of seats and other information are stored. Each route has many bus stops. One bus stop is part of only one route. The route and bus stop details are stored. Each bus is parked in a terminal and the information of the terminal such as Terminal ID, Location, Date and Time are stored.

Each bus drives on one particular route and follows a particular time table. The time table information such as day and start time, end time and intervals (15 min, 20 min, 30 min) are recorded. Values for 'day' can be {M,T,W,Th,F,Sat,Sun}. A unique ID in the form of "DTXX" is given to each unique record in the timetable. For example, Day-{M}, StartTime- 10:00, EndTime – 20:00, Interval - 15m can have ID DT01 and so on. A bus may have different StartTime, EndTime or Interval for different days.

Project Questions

1. Is the ability to model superclass/subclass relationships likely to be important in a transportation system environment such as DART? Why or why not?

   In the process of designing our entity relationship diagram for a database, we may find that attributes of two or more entities overlap, meaning that these entities seem very similar but still have a few differences. In this case, we may create a subtype of the parent entity that contains distinct attributes. A parent entity becomes a super type that has a relationship with one or more subtypes. These are like "Disjoint" and "Union" Operation of an EER diagram model also known as Specialization and Generalization.

   The following problem condition directs us to use subclass-superclass structure in the environment:

   - A Person can be an Employee or an A-Class Passenger.
     Employee and A-Class Passenger are subclass of Person OR a Person is superclass of Employee and A-Class Passenger.
   - An A-Star Passenger can be an Employee or an A-Class passenger or both.
     Employee, and an A-Class passenger are subclass of an A-Star Passenger.
   - Employee is further classified as either Staff, Ticket Checker, or Bus Drivers.
     Staff, Ticket Checker, and Bus Drivers are subclasses of the Employee class.

2. Can you think of 5 more business rules (other than those explicitly described above) that are likely to be used in DART system? Describe how your EER will change to represent the rules.

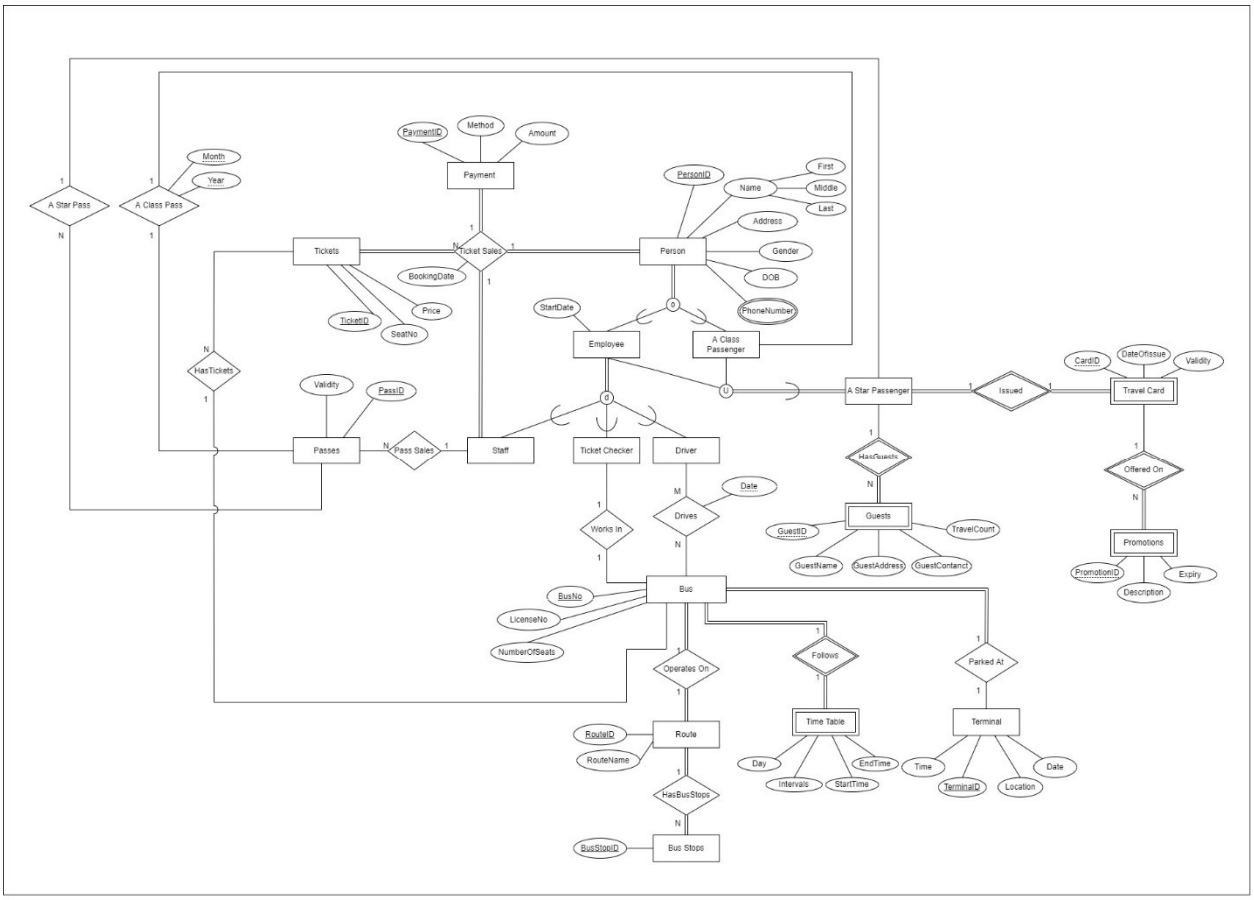
   Rules likely to be applied in a DART environment are:

   1. An Employee can also be a Student => Add new subclass “Student” of Employee.
   2. A Student will be responsible for cleaning and disinfecting a Bus => Add new relationship “Clean” from “Student” to “Bus”.
   3. Students can also have special student discount.
   4. Group passes can also be added => Add new class “Group Booking” in association with “Staff” and “Bus” classes.
   5. Free tickets for children below age 10, senior citizens and handicap persons => Add “ticket_type” attribute with values normal, child, senior, disabled
   6. Advertisements on uses to generate revenue => Add new class “Ads” and relate to “Bus” class.

3. Justify using a Relational DBMS like Oracle for this project (Successfully design a relational database system, show the design in final report).

   In the DART, there are several important entity types. These entities are related to one another, and in some cases depend on each other's data. For example, ticket sales contains information about tickets sold, the person buying the tickets, the person selling the tickets and the payment information used to buy those tickets. As we can see there is a lot of shared data, which in a non-relational environment would have a lot of data redundancy. Also, since this system is dynamic in nature, data integrity is of very high importance. The usage of relational DBMS like ORACLE/MYSQL would allow for relations to be established between the various entity, thus eliminating the data integrity and redundancy issues.

ER Diagram



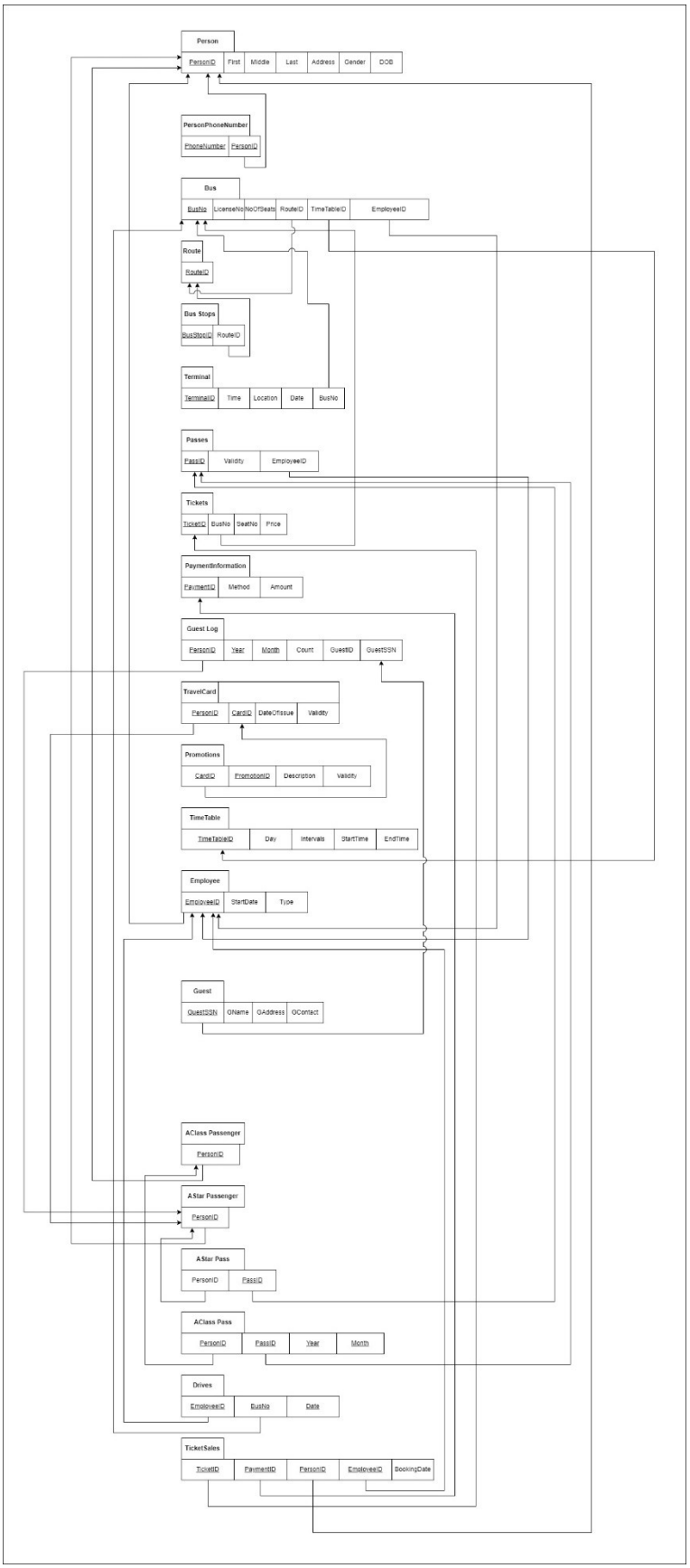
Assumptions

1. Person entity is specialized to Employee entity and A-Class Passenger entity.
2. A-Star Passenger is Union Type of A-Class Passenger and Employee
3. Travel Card is a weak entity depending upon A-Star entity with the identifying relationship ‘Issued’.
4. Promotion is a weak entity depending on Travel Card with the identifying relationship ‘Offered On’.
5. Guests is a weak entity depending on A-Star entity with an identifying relationship “HasGuests”.
6. TicketSales is a weak entity depending on Tickets, Person, Staff(Employee) and Payment Information.
7. Passes entity has a unique Pass ID, and has a Validity.
8. Timetable is a weak entity depending on Bus with the identifying relationship ‘Follow’.
9. A Class Pass has Month and Year as weak attributes.

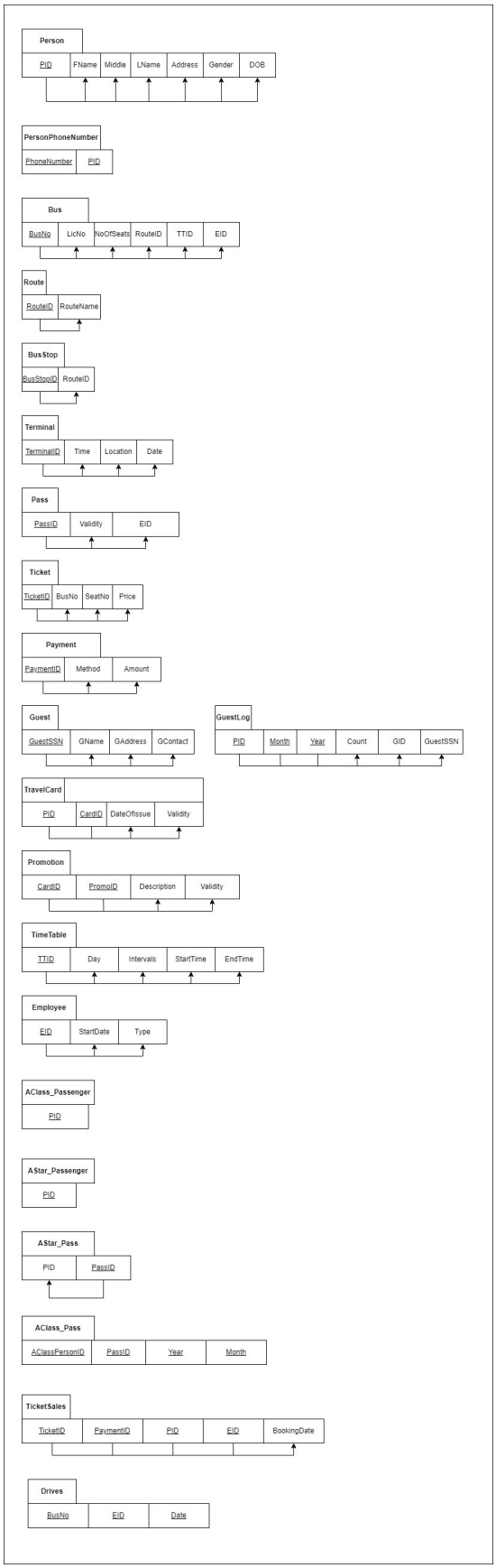
Relational Schema

Primary Keys and Foreign Keys for relational mapping of DART Schema.

Relation	PKs	FKs
Person	PersonID	
PersonPhoneNumber	PhoneNumber, PersonID	PersonID
Bus	BusNo	TimeTableID, EmployeeID, RouteID
Route	RouteID	
BusStops	BusStopID	RouteID
Terminal	TerminalID	BusNo
Passes	PassID	EmployeeID
Tickets	TicketID	BusNo
PaymentInformation	PaymentID	
Guest Log	PersonID, Month, Year	GuestSSN
TravelCard	PersonID, CardID	PersonID
Promotions	CardID, PromotionID	CardID
TimeTable	TimeTableID	
Employee	EmployeeID	EmployeeID
AClass Passenger	PersonID	PersonID
AStar Passenger	PersonID	PersonID
AStar Pass	PassID	PassID, PersonID
AClass Pass	PassID, PersonID, Month, Year	PassID, PersonID
Drives	PersonID, BusNo, Date	PersonID, BusNo
TicketSales	TicketID	TicketID, PaymentID, PersonID, PersonID


Person
PersonID
First
Middle
Last
Address
Gender
DOB
PersonPhoneNumber
PhoneNumber
PersonID
Bus
BusNo
LicenseNo
NoOfSeats
RouteID
TimeTableID
EmployeeID
Route
RouteID
Bus Stops
BusStopID
RouteID
Terminal
TerminalID
Time
Location
Date
BusNo
Passes
PassID
Validity
EmployeeID
Tickets
TicketID
BusNo
SeatNo
Price
PaymentInformation
PaymentID
Method
Amount
Guest Log
PersonID
Year
Month
Count
GuestID
GuestSSN
TravelCard
PersonID
CardID
DateOfIssue
Validity
Promotions
CardID
PromotionID
Description
Validity
TimeTable
TimeTableID
Day
Intervals
StartTime
EndTime
Employee
EmployeeID
StartDate
Type
Guest
GuestSSN
GName
GAddress
GContact
AClass Passenger
PersonID
AStar Passenger
PersonID
AStar Pass
PersonID
PassID
AClass Pass
PersonID
PassID
Year
Month
Drives
EmployeeID
BusNo
Date
TicketSales
TicketID
PaymentID
PersonID
EmployeeID
BookingDate

Functional Dependency



SQL STATEMENTS

CREATING SCHEMA

Database

```
CREATE DATABASE dartdb;
```

CREATING TABLES

AClass_Pass

```
CREATE TABLE `aclass_pass` (
  `passid` int NOT NULL,
  `pid` varchar(4) NOT NULL,
  `month` varchar(10) NOT NULL,
  `year` varchar(10) NOT NULL,
  PRIMARY KEY (`month`,`pid`,`year`),
  KEY `passid` (`passid`),
  KEY `pid` (`pid`),
  CONSTRAINT `aclass_pass_ibfk_1` FOREIGN KEY (`passid`) REFERENCES `pass` (`passid`),
  CONSTRAINT `aclass_pass_ibfk_2` FOREIGN KEY (`pid`) REFERENCES `aclass_passenger` (`pid`)
)
```

AClass_Passenger

```
CREATE TABLE `aclass_passenger` (
  `pid` varchar(4) NOT NULL,
  PRIMARY KEY (`pid`),
  CONSTRAINT `aclass_passenger_ibfk_1` FOREIGN KEY (`pid`) REFERENCES `person` (`pid`)
)
```

AStar_Pass

```
CREATE TABLE `astar_pass` (
  `passid` int NOT NULL,
  `pid` varchar(4) NOT NULL,
  PRIMARY KEY (`passid`),
  KEY `pid` (`pid`),
  CONSTRAINT `astar_pass_ibfk_1` FOREIGN KEY (`passid`) REFERENCES `pass` (`passid`),
  CONSTRAINT `astar_pass_ibfk_2` FOREIGN KEY (`pid`) REFERENCES `astar_passenger` (`pid`)
)
```

AStar_Passenger

```
CREATE TABLE `astar_passenger` (
  `pid` varchar(4) NOT NULL,
  PRIMARY KEY (`pid`),
  CONSTRAINT `astar_passenger_ibfk_1` FOREIGN KEY (`pid`) REFERENCES `person` (`pid`)
)
```

Bus

```
CREATE TABLE `bus` (
  `busno` int NOT NULL,
  `licno` varchar(20) NOT NULL,
  `noofseats` int NOT NULL,
  `ttid` varchar(4) NOT NULL,
  `routeid` int NOT NULL,
  `eid` varchar(4) NOT NULL,
  PRIMARY KEY (`busno`),
  KEY `ttid` (`ttid`),
  KEY `routeid` (`routeid`),
  KEY `eid` (`eid`),
```

```
  CONSTRAINT `bus_ibfk_1` FOREIGN KEY (`ttid`) REFERENCES `timetable` (`ttid`),
  CONSTRAINT `bus_ibfk_2` FOREIGN KEY (`routeid`) REFERENCES `route` (`routeid`),
  CONSTRAINT `bus_ibfk_3` FOREIGN KEY (`eid`) REFERENCES `employee` (`eid`)
)
```

BusStop

```
CREATE TABLE `busstop` (
  `routeid` int NOT NULL,
  `busstopid` int NOT NULL,
  PRIMARY KEY (`routeid`,`busstopid`),
  CONSTRAINT `busstop_ibfk_1` FOREIGN KEY (`routeid`) REFERENCES `route` (`routeid`)
)
```

Drives

```
CREATE TABLE `drives` (
  `eid` varchar(4) NOT NULL,
  `busno` int NOT NULL,
  `date` date NOT NULL,
  PRIMARY KEY (`busno`,`eid`,`date`),
  KEY `eid` (`eid`),
  CONSTRAINT `drives_ibfk_1` FOREIGN KEY (`busno`) REFERENCES `bus` (`busno`),
  CONSTRAINT `drives_ibfk_2` FOREIGN KEY (`eid`) REFERENCES `employee` (`eid`)
)
```

Employee

```
CREATE TABLE `employee` (
  `eid` varchar(4) NOT NULL,
  `startdate` date NOT NULL,
  `type` varchar(20) NOT NULL,
```

```
  PRIMARY KEY (`eid`),
  CONSTRAINT `employee_ibfk_1` FOREIGN KEY (`eid`) REFERENCES `person` (`pid`),
  CONSTRAINT `employee_chk_1` CHECK ((`type` in (’Staff', 'Ticket Checker', 'Driver')))
)
```

Guest

```
CREATE TABLE `guest` (
  `guestSSN` varchar(15) NOT NULL,
  `gname` varchar(30) NOT NULL,
  `gaddress` varchar(120) NOT NULL,
  `gcontact` int NOT NULL,
  PRIMARY KEY (`guestSSN`)
)
```

GuestLog

```
CREATE TABLE `guestlog` (
  `pid` varchar(4) NOT NULL,
  `month` varchar(20) NOT NULL,
  `year` int NOT NULL,
  `count` int NOT NULL,
  `guestid` varchar(20) NOT NULL,
  `guestSSN` varchar(15) NOT NULL,
  PRIMARY KEY (`pid`,`month`,`year`),
  KEY `guestSSN` (`guestSSN`),
  CONSTRAINT `guestlog_ibfk_1` FOREIGN KEY (`pid`) REFERENCES `astar_passenger` (`pid`),
  CONSTRAINT `guestlog_ibfk_2` FOREIGN KEY (`guestSSN`) REFERENCES `guest` (`guestSSN`),
  CONSTRAINT `guestlog_chk_1` CHECK ((`count` between 1 and 4))
)
```

Pass

```
CREATE TABLE `pass` (
  `passid` int NOT NULL,
  `validity` date NOT NULL,
  `eid` varchar(4) NOT NULL,
  PRIMARY KEY (`passid`),
  KEY `eid` (`eid`),
  CONSTRAINT `pass_ibfk_1` FOREIGN KEY (`eid`) REFERENCES `employee` (`eid`)
)
```

Payment

```
CREATE TABLE `payment` (
  `paymentid` int NOT NULL AUTO_INCREMENT,
  `method` varchar(120) NOT NULL,
  `amount` float NOT NULL,
  PRIMARY KEY (`paymentid`),
  CONSTRAINT `payment_chk_1` CHECK ((`method` in ('Cash', 'Card')))
)
```

Person

```
CREATE TABLE `person` (
  `pid` varchar(4) NOT NULL,
  `fname` varchar(120) NOT NULL,
  `mname` varchar(120) DEFAULT NULL,
  `lname` varchar(120) NOT NULL,
  `address` varchar(300) NOT NULL,
  `gender` varchar(10) NOT NULL,
  `dob` date NOT NULL,
  PRIMARY KEY (`pid`),
```

```
  CONSTRAINT `GCHECK` CHECK ((`gender` in ('M', 'F'))),
  CONSTRAINT `PIDCHECK` CHECK (regexp_like(`pid`,'^[P][0-9]{3}$'))
)
```

PersonPhoneNumber

```
CREATE TABLE `personphonenumber` (
  `pid` varchar(4) NOT NULL,
  `phonenumber` int NOT NULL,
  PRIMARY KEY (`pid`,`phonenumber`),
  CONSTRAINT `personphonenumber_ibfk_1` FOREIGN KEY (`pid`) REFERENCES `person` (`pid`),
  CONSTRAINT `personphonenumber_chk_1` CHECK ((`phonenumber` between 100000000 and 9999999999))
)
```

Promotion

```
CREATE TABLE `promotion` (
  `promoid` int NOT NULL,
  `cardid` int NOT NULL,
  `description` varchar(60) DEFAULT NULL,
  `validity` date NOT NULL,
  PRIMARY KEY (`promoid`,`cardid`),
  KEY `cardid` (`cardid`),
  CONSTRAINT `promotion_ibfk_1` FOREIGN KEY (`cardid`) REFERENCES `travelcard` (`cardid`)
)
```

Route

```
CREATE TABLE `route` (
  `routeid` int NOT NULL AUTO_INCREMENT,
  `routename` varchar(120) NOT NULL,
```

```
  PRIMARY KEY (`routeid`)
)
```

Terminal

```
CREATE TABLE `terminal` (
  `terminalid` int NOT NULL AUTO_INCREMENT,
  `location` varchar(300) NOT NULL,
  `date` date NOT NULL,
  `time` time NOT NULL,
   `busno` int DEFAULT NULL,
  PRIMARY KEY (`terminalid`),
  KEY `terminal_ibfk_1` (`busno`),
  CONSTRAINT `terminal_ibfk_1` FOREIGN KEY (`busno`) REFERENCES `bus` (`busno`)
)
```

Ticket

```
CREATE TABLE `ticket` (
  `ticketid` int NOT NULL AUTO_INCREMENT,
  `busno` int NOT NULL,
  `seatno` int NOT NULL,
  `price` int NOT NULL,
  PRIMARY KEY (`ticketid`),
  KEY `busno_idx` (`busno`),
  CONSTRAINT `busno` FOREIGN KEY (`busno`) REFERENCES `bus` (`busno`)
)
```

TicketSales

```
CREATE TABLE `ticketsales` (
  `paymentid` int NOT NULL,
```

```
  `ticketid` int NOT NULL,
  `eid` varchar(4) NOT NULL,
  `pid` varchar(4) NOT NULL,
  `bookingdate` date NOT NULL,
  PRIMARY KEY (`paymentid`,`ticketid`,`eid`,`pid`),
  KEY `ticketid` (`ticketid`),
  KEY `eid` (`eid`),
  KEY `pid` (`pid`),
  CONSTRAINT `ticketsales_ibfk_1` FOREIGN KEY (`ticketid`) REFERENCES `ticket` (`ticketid`),
  CONSTRAINT `ticketsales_ibfk_2` FOREIGN KEY (`paymentid`) REFERENCES `payment` (`paymentid`),
  CONSTRAINT `ticketsales_ibfk_3` FOREIGN KEY (`eid`) REFERENCES `employee` (`eid`),
  CONSTRAINT `ticketsales_ibfk_4` FOREIGN KEY (`pid`) REFERENCES `person` (`pid`)
)
```

TimeTable

```
CREATE TABLE `timetable` (
  `ttid` varchar(4) NOT NULL,
  `day` varchar(10) NOT NULL,
  `starttime` time NOT NULL,
  `endtime` time NOT NULL,
  `interval` int NOT NULL,
  PRIMARY KEY (`ttid`),
  CONSTRAINT `timetable_chk_1` CHECK (regexp_like(`ttid`,'^(DT)?[0-9]{2}$')),
  CONSTRAINT `timetable_chk_2` CHECK ((`day` in ('M','T','W','Th','F','Sat', 'Sun'))),
  CONSTRAINT `timetable_chk_3` CHECK ((`interval` in (15,20,30)))
)
```

TravelCard

```
CREATE TABLE `travelcard` (
```

```
 `cardid` int NOT NULL AUTO_INCREMENT,
 `pid` varchar(4) NOT NULL,
 `date_of_issue` date NOT NULL,
 `validity` date NOT NULL,
 PRIMARY KEY (`cardid`,`pid`),
 KEY `travelcard_ibfk_1` (`pid`),
 CONSTRAINT `travelcard_ibfk_1` FOREIGN KEY (`pid`) REFERENCES `astar_passenger` (`pid`)
)
```

CREATING TRIGGERS

Person

```
CREATE DEFINER=`root`@`localhost` TRIGGER `person_BEFORE_INSERT` BEFORE INSERT ON `person` FOR EACH ROW BEGIN
IF TIMESTAMPDIFF(YEAR, NEW.dob,CURDATE()) < 16 THEN
 SIGNAL SQLSTATE '45000'
 SET MESSAGE_TEXT = 'AGE MUST BE GREATER THAN 16';
END IF;
END
```

QUERIES

1. For each employee class, list the employees belonging to that class.

```
SELECT e.type, e.eid, p.fname, p.lname, p.gender
FROM employee e, person p
WHERE p.pid = e.eid
ORDER BY e.type;
```

2. Find the names of employees who are also an A-Class Passenger.

```
SELECT p.fname, p.lname
FROM employee e, person p, aclass_passenger a
WHERE e.eid = a.pid AND e.eid = p.pid;
```

3. Find the average number of bookings made by the top five A-Star Passengers.

```
SELECT AVG(bookings)
FROM ( SELECT *
        FROM top_astar_passenger
        ORDER BY bookings DESC
        LIMIT 5 ) as TOPFIVE;
```

4. Find the Bus ID and Route names of the bus that is booked the most.

```
SELECT b.busno, b.routeid
FROM ticketsales ts, ticket t, bus b
WHERE ts.ticketid = t.ticketid AND t.busno = b.busno
GROUP BY  b.busno, b.routeid
ORDER BY COUNT(*) DESC
LIMIT 1;
```

6. Find the total number bookings for each bus in the system.

```
SELECT b.busno, count(ts.pid) AS Bookings
FROM ( ticketsales ts INNER JOIN ticket t ON ts.ticketid = t.ticketid ) RIGHT JOIN bus b ON t.busno = b.busno
GROUP BY  b.busno
ORDER BY COUNT(ts.pid);
```

7. Find the driver details who has driven every day of the past week.

```
SELECT d.eid
FROM DRIVES d, timetable t, bus b
WHERE d.busno = b.busno
        AND b.ttid = t.ttid
        AND d.date BETWEEN (CURRENT_DATE - INTERVAL 7 DAY) AND CURRENT_DATE
        AND t.day IN ( SELECT DISTINCT(day) FROM timetable )
GROUP BY d.eid
HAVING COUNT(d.eid) =  ( SELECT COUNT(DISTINCT(day)) FROM timetable )
```

8. Find the count of passengers who booked the most popular bus.

```
SELECT COUNT(DISTINCT(s.pid)) AS BOOKINGS
FROM popular_bus pb, ticket t, ticketsales s
WHERE s.ticketid = t.ticketid AND t.busno = pb.busno ;
```

9. List all the booking details issued after the most current employee was hired.

```
SELECT *
FROM ticketsales
WHERE bookingdate > ( SELECT startdate FROM employee ORDER BY startdate DESC LIMIT 1 );
```

10. List all the employees that have enrolled as A-Star Passengers within a month of being employed.

```
SELECT e.eid, p.fname, e.startdate, tc.date_of_issue, tc.cardid
FROM employee e,
astar_passenger asp, travelcard tc, person p
WHERE e.eid = p.pid
        AND e.eid = asp.pid
        AND tc.pid = asp.pid
        AND tc.date_of_issue BETWEEN e.startdate AND DATE(e.startdate + INTERVAL 1 MONTH);
```

11. Find the route with the highest number of bus stops.

```
SELECT routeid, count(*) AS BusStopCount
FROM busstop
 GROUP BY routeid
ORDER BY BusStopCount DESC
LIMIT 1;
```

12. Find the name of passengers who have been A-Star Passengers for over 5 years.

```
SELECT p.fname, tc.date_of_issue
FROM person p, astar_passenger asp, travelcard tc
WHERE p.pid = asp.pid
        AND asp.pid = tc.pid
        AND tc.date_of_issue < DATE(CURRENT_DATE - INTERVAL 5 YEAR);
```

13. Find the bookings made by the potential A-Star Passengers in the last year.

```
SELECT pap.pid, COUNT(*) AS Bookings
FROM potential_astar_passenger pap, ticketsales ts
WHERE pap.pid = ts.pid
        AND ts.bookingdate BETWEEN (CURRENT_DATE - INTERVAL 1 YEAR) AND CURRENT_DATE
GROUP BY pap.pid ;
```

VIEWS

1. Top A-Star Passenger- This view returns the First Name, Last Name and Date of membership enrollment of those passengers who have travelled more than 6 times in the last month.

```
CREATE VIEW `top_astar_passenger` AS
   SELECT
      `p`.`pid` AS `pid`,
      `p`.`fname` AS `fname`,
      `p`.`lname` AS `lname`,
      `tc`.`date_of_issue` AS `date_of_issue`,
      COUNT(0) AS `bookings`
   FROM
      ((`person` `p`
      JOIN `travelcard` `tc`)
      JOIN `ticketsales` `ts`)
   WHERE
      ((`p`.`pid` = `tc`.`pid`)
         AND (`ts`.`pid` = `tc`.`pid`)
         AND (`ts`.`bookingdate` BETWEEN CAST((CURDATE() - INTERVAL 1 MONTH) AS DATE) AND
CURDATE()))
   GROUP BY `p`.`pid` , `p`.`fname` , `p`.`lname` , `tc`.`date_of_issue`
   HAVING (COUNT(*) > 6)
```

2. Popular Bus- This view returns the details of the bus that the passenger has booked the most in the past 2 months

```
CREATE  VIEW `popular_bus` AS
   SELECT
      `b`.`busno` AS `busno`,
      `b`.`licno` AS `licno`,
      `b`.`noofseats` AS `noofseats`,
      `b`.`ttid` AS `ttid`,
      `b`.`routeid` AS `routeid`,
      `b`.`eid` AS `eid`
   FROM
      ((`ticketsales` `ts`
      JOIN `ticket` `t`)
      JOIN `bus` `b`)
   WHERE
      ((`ts`.`ticketid` = `t`.`ticketid`)
         AND (`t`.`busno` = `b`.`busno`)
         AND (`ts`.`bookingdate` BETWEEN (CURDATE() - INTERVAL 2 MONTH) AND CURDATE()))
   GROUP BY `b`.`busno` , `b`.`licno` , `b`.`noofseats` , `b`.`ttid` , `b`.`routeid` , `b`.`eid`
   ORDER BY COUNT(*) DESC
   LIMIT 1
```

4. Potential A-Star Passenger- This view returns the name, phone number and ID of the A-Class Passengers who travelled more than 4 time in the past 2 months.

```
CREATE VIEW `potential_astar_passenger` AS
  SELECT
    `p`.`fname` AS `fname`,
    `ppn`.`phonenumber` AS `phonenumber`,
    `p`.`pid` AS `pid`
  FROM
    (((`person` `p`
    JOIN `personphonenumber` `ppn`)
    JOIN `aclass_passenger` `acp`)
    JOIN `ticketsales` `ts`)
  WHERE
    ((`p`.`pid` = `ppn`.`pid`)
      AND (`p`.`pid` = `acp`.`pid`)
      AND (`acp`.`pid` = `ts`.`pid`)
      AND (`ts`.`bookingdate` BETWEEN (CURDATE() - INTERVAL 2 MONTH) AND CURDATE()))
  GROUP BY `p`.`fname` , `ppn`.`phonenumber` , `p`.`pid`
  HAVING (COUNT(*) > 4)
```

5. Top Employee- This view returns the details of the employee who has made the most number of bookings in the past month.

```
CREATE VIEW `top_employee` AS
   SELECT
      `p`.`pid` AS `pid`,
      `p`.`fname` AS `fname`,
      `p`.`mname` AS `mname`,
      `p`.`lname` AS `lname`,
      `p`.`address` AS `address`,
      `p`.`gender` AS `gender`
   FROM
      ((`person` `p`
      JOIN `employee` `e`)
      JOIN `ticketsales` `ts`)
   WHERE
      ((`p`.`pid` = `e`.`eid`)
         AND (`e`.`eid` = `ts`.`eid`)
         AND (`ts`.`bookingdate` BETWEEN (CURDATE() - INTERVAL 1 MONTH) AND CURDATE()))
   GROUP BY `p`.`fname` , `p`.`pid`
   ORDER BY COUNT(*) DESC
   LIMIT 1
```