CS 6360 Database Design

Project Report

Group 15

Harish Srinivasan

Pratik Deepak Naik

Jay Virendra Soni

a. 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 timetable. The timetable 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.

b. 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?

- Yes, using the superclass/subclass relationships allows us to define common relations and attributes for the super classes reducing redundancy as described below:
- We can see in EER that it is very necessary to divide person class into different types. Also, employee class is divided into 3 types which is needed separate roles of different employees.
- It becomes easy to manage different entities with different roles using superclass/subclass relationships.

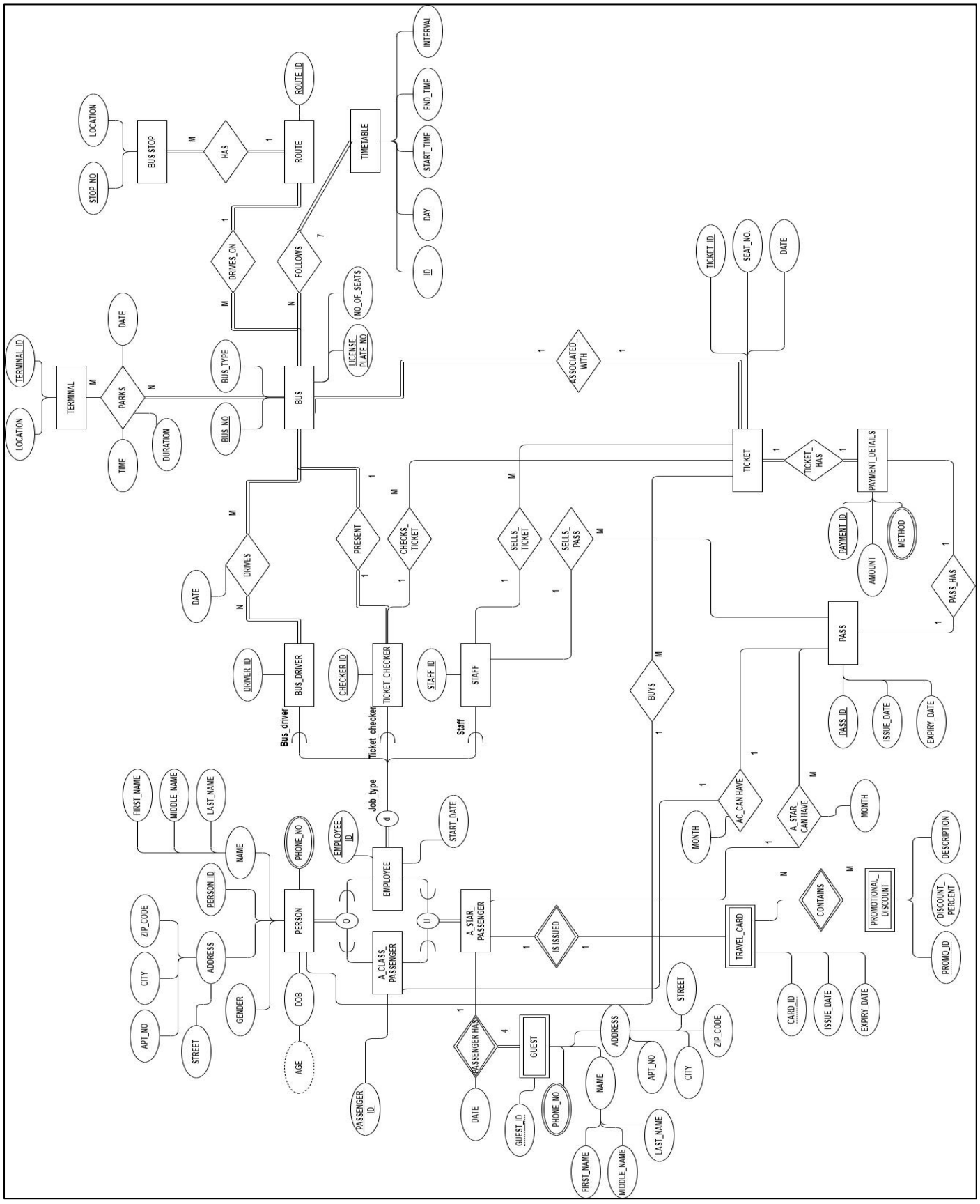
2. Can you think of 5 more business rules (other than those explicitly described above) that are likely to be used in a DART environment? Describe how your EER will change to represent the rules.

- Separate Student Passes can be issued based on student id => Add “pass_type” attribute in pass class.
- Group Bookings can also be added => Add new class “group_booking” is to be associated with “staff” and “bus” classes.
- Special Buses on festivals => Add “special_bus” weak entity with particular dates.
- Free tickets for children below age of 10, senior citizens and handicap persons => Add “ticket_type” attribute with values child, normal, senior, disable.
- Advertisements on buses 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 the final report).

- We use MySQL as our RDBMS due to its speed, security, and ease of use. We design our Relational database system in the third normal form. The design is shown in the “RDB.png” file submitted along with this report.

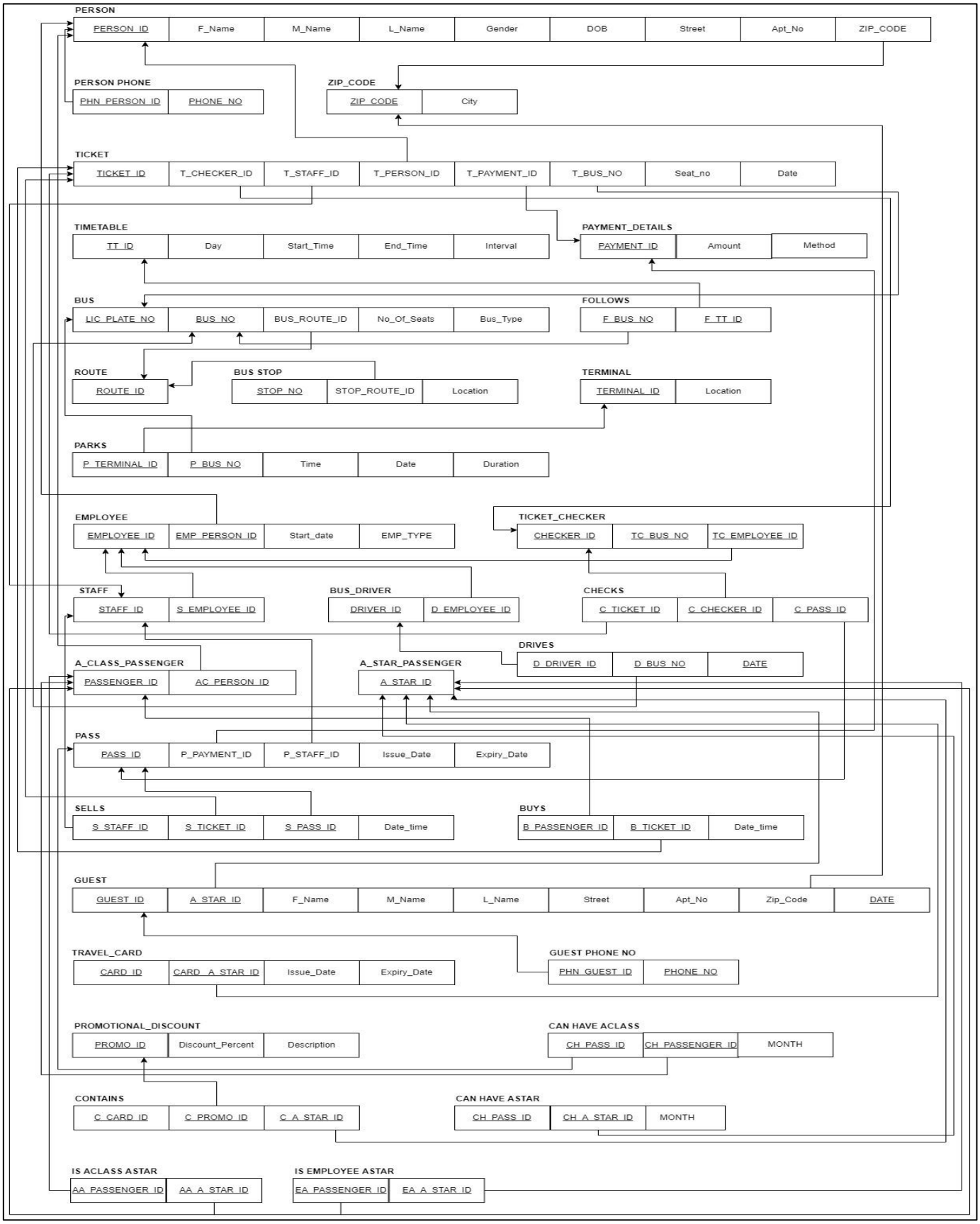
c. EER diagram with all assumptions (Solution for Phase II).



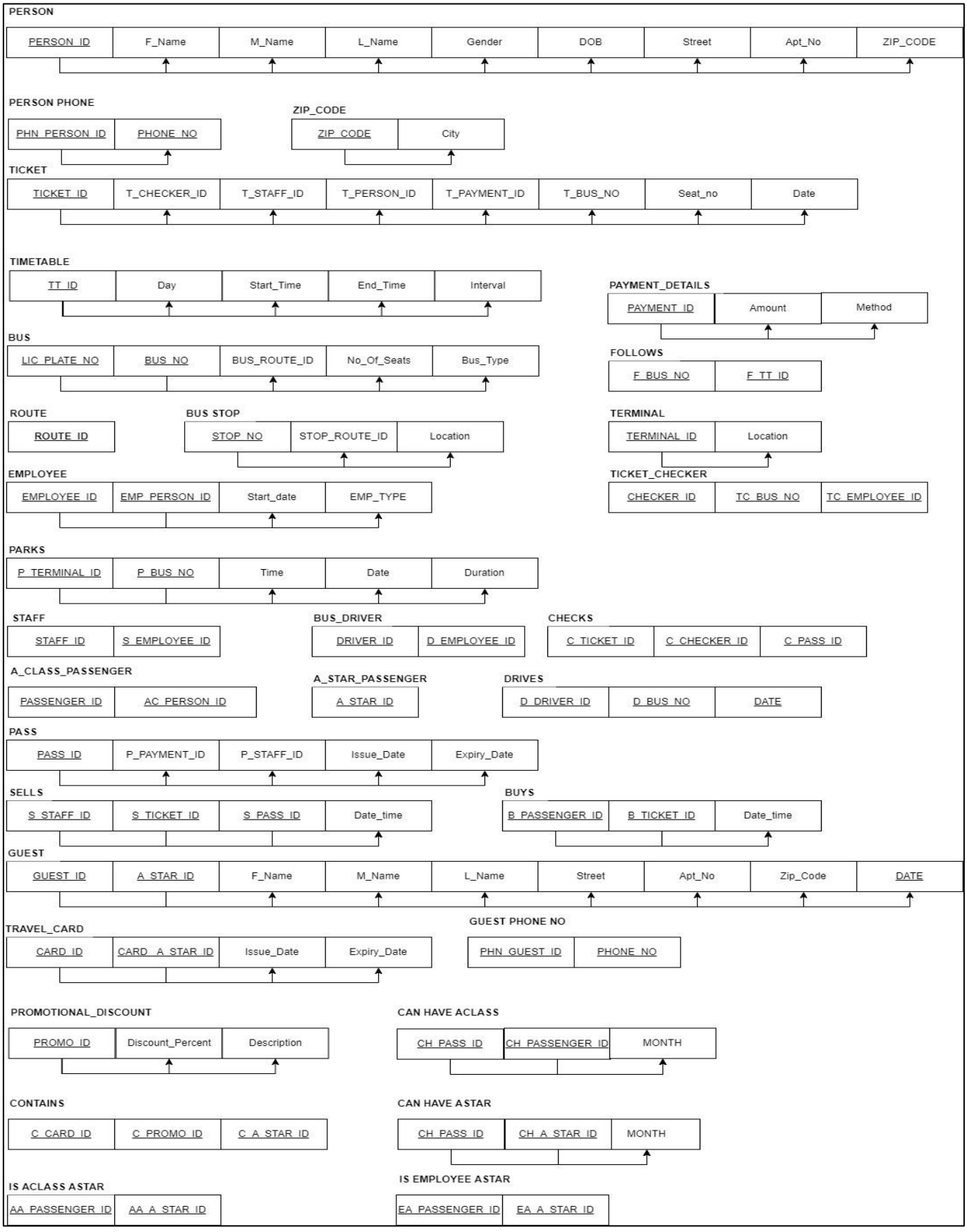
ASSUMPTIONS for EER Diagram:

- We have assumed that employees do not take any holidays from work. Therefore, all employees do work each day.
- Promotional discounts on travel cards are offered only for a particular day and not for an entire week or month. Therefore, if promotions are offered for three simultaneous dates, there will be three entities in the promotional entity type.
- Each guest belongs to only one A-star passenger. Two or more A-star passengers cannot have the same guest.
- A ticket is valid for the entire day and the passenger can use the ticket for travel at any time of the day.
- All A-Star passengers are A-class passengers
- All bus stops have unique bus stop numbers which can be used to identify them.
- Buses parked in the terminal can be parked for specific durations of time.

d. Relational Schema after normalization



e. Dependency diagram



f. SQL Statements (Solution of Phase 3 – c d e)

```
CREATE SCHEMA dart;
```

Table Creation

No.	Table	SQL
1	person	CREATE TABLE `person` (`person_id` varchar(4) NOT NULL, `f_name` varchar(45) NOT NULL, `m_name` varchar(45) DEFAULT NULL, `l_name` varchar(45) NOT NULL, `gender` varchar(1) NOT NULL, `dob` date NOT NULL, `street` varchar(45) NOT NULL, `apt_no` varchar(5) NOT NULL, `zip_code` varchar(5) NOT NULL, PRIMARY KEY (`person_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
2	person_ phone	CREATE TABLE `person_phone` (`phn_person_id` varchar(4) NOT NULL, `phone_no` varchar(10) NOT NULL, PRIMARY KEY (`phn_person_id`,`phone_no`), CONSTRAINT `fk_person_phone_1` FOREIGN KEY (`phn_person_id`) REFERENCES `person` (`person_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
3	zip_code	CREATE TABLE `dart`.`zip_code` (`zip_code` VARCHAR(5) NOT NULL, `city` VARCHAR(45) NOT NULL, PRIMARY KEY (`zip_code`));
4	a_star_ passenger	CREATE TABLE `a_star_passenger` (`a_star_id` varchar(5) NOT NULL, PRIMARY KEY (`a_star_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

5	employee	CREATE TABLE `employee` (`employee_id` varchar(5) NOT NULL, `emp_person_id` varchar(5) NOT NULL, `start_date` date NOT NULL, `e_type` varchar(45) NOT NULL, PRIMARY KEY (`employee_id`,`emp_person_id`), KEY `emp_person_id` (`emp_person_id`), CONSTRAINT `employee_ibfk_1` FOREIGN KEY (`emp_person_id`) REFERENCES `person` (`person_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
6	a_class_ passenger	CREATE TABLE `a_class_passenger` (`passenger_id` varchar(5) NOT NULL, `ac_person_id` varchar(5) NOT NULL, PRIMARY KEY (`passenger_id`,`ac_person_id`), KEY `fk_a_class_passenger_1_idx` (`ac_person_id`), CONSTRAINT `fk_a_class_passenger_1` FOREIGN KEY (`ac_person_id`) REFERENCES `person` (`person_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
7	staff	CREATE TABLE `staff` (`staff_id` varchar(5) NOT NULL, `s_employee_id` varchar(5) NOT NULL, PRIMARY KEY (`staff_id`,`s_employee_id`), KEY `fk_staff_1_idx` (`s_employee_id`), CONSTRAINT `fk_staff_1` FOREIGN KEY (`s_employee_id`) REFERENCES `employee` (`employee_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
8	ticket_ checker	CREATE TABLE `ticket_checker` (`checker_id` varchar(5) NOT NULL, `tc_bus_no` varchar(7) NOT NULL, `tc_employee_id` varchar(5) NOT NULL, PRIMARY KEY (`checker_id`,`tc_bus_no`,`tc_employee_id`), KEY `fk_ticket_checker_1_idx` (`tc_employee_id`), KEY `fk_ticket_checker_2_idx` (`tc_bus_no`), CONSTRAINT `fk_ticket_checker_1` FOREIGN KEY (`tc_employee_id`) REFERENCES `employee` (`employee_id`), CONSTRAINT `fk_ticket_checker_2` FOREIGN KEY (`tc_bus_no`) REFERENCES `bus` (`license_plate_no`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

9	bus_driver	CREATE TABLE `bus_driver` (`driver_id` varchar(5) NOT NULL, `d_employee_id` varchar(5) NOT NULL, PRIMARY KEY (`driver_id`,`d_employee_id`), KEY `fk_bus_driver_1_idx` (`d_employee_id`), CONSTRAINT `fk_bus_driver_1` FOREIGN KEY (`d_employee_id`) REFERENCES `employee` (`employee_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
10	route	CREATE TABLE `route` (`route_id` varchar(5) NOT NULL, PRIMARY KEY (`route_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;CREATE TABLE `route` (`route_id` varchar(5) NOT NULL, PRIMARY KEY (`route_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
11	bus	CREATE TABLE `bus` (`license_plate_no` varchar(7) NOT NULL, `bus_no` varchar(5) NOT NULL, `bus_route_id` varchar(5) NOT NULL, `no_of_seats` int NOT NULL, `bus_type` varchar(5) NOT NULL, PRIMARY KEY (`license_plate_no`,`bus_no`), KEY `fk_bus_2_idx` (`bus_route_id`), CONSTRAINT `fk_bus_2` FOREIGN KEY (`bus_route_id`) REFERENCES `route` (`route_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

12	ticket	CREATE TABLE `ticket` (`ticket_id` varchar(5) NOT NULL, `t_bus_no` varchar(7) NOT NULL, `t_seat_no` int NOT NULL, `t_checker_id` varchar(5) NOT NULL, `t_staff_id` varchar(5) NOT NULL, `t_person_id` varchar(4) NOT NULL, `t_payment_id` varchar(5) NOT NULL, `date` date NOT NULL, PRIMARY KEY (`ticket_id`), KEY `fk_ticket_1_idx` (`t_bus_no`), KEY `fk_ticket_2_idx` (`t_checker_id`), KEY `fk_ticket_3_idx` (`t_staff_id`), KEY `fk_ticket_4_idx` (`t_person_id`), KEY `fk_ticket_5_idx` (`t_payment_id`), CONSTRAINT `fk_ticket_1` FOREIGN KEY (`t_bus_no`) REFERENCES `bus` (`license_plate_no`), CONSTRAINT `fk_ticket_2` FOREIGN KEY (`t_checker_id`) REFERENCES `ticket_checker` (`checker_id`), CONSTRAINT `fk_ticket_3` FOREIGN KEY (`t_staff_id`) REFERENCES `staff` (`staff_id`), CONSTRAINT `fk_ticket_4` FOREIGN KEY (`t_person_id`) REFERENCES `person` (`person_id`), CONSTRAINT `fk_ticket_5` FOREIGN KEY (`t_payment_id`) REFERENCES `payment_details` (`payment_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
13	bus_stop	CREATE TABLE `bus_stop` (`stop_no` varchar(5) NOT NULL, `stop_route_id` varchar(5) NOT NULL, `location` varchar(45) NOT NULL, PRIMARY KEY (`stop_no`), KEY `fk_bus_stop_1_idx` (`stop_route_id`), CONSTRAINT `fk_bus_stop_1` FOREIGN KEY (`stop_route_id`) REFERENCES `route` (`route_id`) ON DELETE RESTRICT ON UPDATE RESTRICT) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
14	timetable	CREATE TABLE `timetable` (`tt_id` varchar(4) NOT NULL, `day` varchar(3) NOT NULL,

		`start_time` time NOT NULL, `end_time` time NOT NULL, `interval` int NOT NULL, PRIMARY KEY (`tt_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
15	payment_ details	CREATE TABLE `payment_details` (`payment_id` varchar(5) NOT NULL, `amount` float NOT NULL, `method` varchar(5) NOT NULL, PRIMARY KEY (`payment_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
16	follows	CREATE TABLE `follows` (`f_bus_no` varchar(7) NOT NULL, `f_tt_id` varchar(5) NOT NULL, PRIMARY KEY (`f_bus_no`,`f_tt_id`), KEY `fk_follows_1_idx` (`f_tt_id`), CONSTRAINT `fk_follows_1` FOREIGN KEY (`f_tt_id`) REFERENCES `timetable` (`tt_id`), CONSTRAINT `fk_follows_2` FOREIGN KEY (`f_bus_no`) REFERENCES `bus` (`license_plate_no`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
17	terminal	CREATE TABLE `terminal` (`terminal_id` varchar(5) NOT NULL, `location` varchar(45) NOT NULL, PRIMARY KEY (`terminal_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
18	parks	CREATE TABLE `parks` (`p_terminal_id` varchar(5) NOT NULL, `p_bus_no` varchar(7) NOT NULL, `time` time NOT NULL, `date` date NOT NULL, `duration` int NOT NULL, PRIMARY KEY (`p_terminal_id`,`p_bus_no`), KEY `fk_parks_2_idx` (`p_bus_no`),

		CONSTRAINT `fk_parks_1` FOREIGN KEY (`p_terminal_id`) REFERENCES `terminal` (`terminal_id`), CONSTRAINT `fk_parks_2` FOREIGN KEY (`p_bus_no`) REFERENCES `bus` (`license_plate_no`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
19	pass	CREATE TABLE `pass` (`pass_id` varchar(5) NOT NULL, `issue_date` date NOT NULL, `expiry_date` date NOT NULL, `p_staff_id` varchar(5) NOT NULL, `p_payment_id` varchar(5) NOT NULL, PRIMARY KEY (`pass_id`), KEY `fk_pass_1_idx` (`p_staff_id`), KEY `fk_pass_3_idx` (`p_payment_id`), CONSTRAINT `fk_pass_1` FOREIGN KEY (`p_staff_id`) REFERENCES `staff` (`staff_id`) ON DELETE RESTRICT ON UPDATE RESTRICT, CONSTRAINT `fk_pass_3` FOREIGN KEY (`p_payment_id`) REFERENCES `payment_details` (`payment_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
20	Sells_pass	CREATE TABLE `sells_passes` (`sp_staff_id` varchar(5) NOT NULL, `sp_pass_id` varchar(5) NOT NULL, `date` date NOT NULL, PRIMARY KEY (`sp_staff_id`,`sp_pass_id`), KEY `fk_sells_passes_2_idx` (`sp_pass_id`), CONSTRAINT `fk_sells_passes_1` FOREIGN KEY (`sp_staff_id`) REFERENCES `staff` (`staff_id`), CONSTRAINT `fk_sells_passes_2` FOREIGN KEY (`sp_pass_id`) REFERENCES `pass` (`pass_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
21	Sells_tickets	CREATE TABLE `sells_tickets` (`st_staff_id` varchar(5) NOT NULL, `st_ticket_id` varchar(5) NOT NULL, `date` date NOT NULL, PRIMARY KEY (`st_staff_id`,`st_ticket_id`), KEY `fk_sells_1_idx` (`st_ticket_id`),

		CONSTRAINT `fk_sells_1` FOREIGN KEY (`st_ticket_id`) REFERENCES `ticket` (`ticket_id`), CONSTRAINT `fk_sells_3` FOREIGN KEY (`st_staff_id`) REFERENCES `staff` (`staff_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
22	buys	CREATE TABLE `buys` (`b_passenger_id` varchar(5) NOT NULL, `b_ticket_id` varchar(5) NOT NULL, `date_time` varchar(45) DEFAULT NULL, PRIMARY KEY (`b_passenger_id`,`b_ticket_id`), KEY `fk_buys_2_idx` (`b_ticket_id`), CONSTRAINT `fk_buys_1` FOREIGN KEY (`b_passenger_id`) REFERENCES `a_class_passenger` (`passenger_id`), CONSTRAINT `fk_buys_2` FOREIGN KEY (`b_ticket_id`) REFERENCES `ticket` (`ticket_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
23	Checks_pass	CREATE TABLE `checks_pass` (`cp_checker_id` varchar(5) NOT NULL, `cp_pass_id` varchar(5) NOT NULL, PRIMARY KEY (`cp_checker_id`,`cp_pass_id`), KEY `fk_checkspass_1_idx` (`cp_pass_id`), CONSTRAINT `fk_checks_pass_1` FOREIGN KEY (`cp_checker_id`) REFERENCES `ticket_checker` (`checker_id`), CONSTRAINT `fk_checkspass_1` FOREIGN KEY (`cp_pass_id`) REFERENCES `pass` (`pass_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
24	Checks_ticket	CREATE TABLE `checks_ticket` (`ct_checker_id` varchar(5) NOT NULL, `ct_ticket_id` varchar(5) NOT NULL, PRIMARY KEY (`ct_checker_id`,`ct_ticket_id`), KEY `fk_checks_ticket_1_idx` (`ct_ticket_id`), CONSTRAINT `fk_checks_ticket_1` FOREIGN KEY (`ct_ticket_id`) REFERENCES `ticket_checker` (`checker_id`), CONSTRAINT `fk_checks_ticket_2` FOREIGN KEY (`ct_ticket_id`) REFERENCES `ticket` (`ticket_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

25	drives	CREATE TABLE `drives` (`d_driver_id` varchar(5) NOT NULL, `d_bus_no` varchar(7) NOT NULL, `date` date NOT NULL, PRIMARY KEY (`d_driver_id`,`d_bus_no`,`date`), KEY `fk_drives_2_idx` (`d_bus_no`), CONSTRAINT `fk_drives_1` FOREIGN KEY (`d_driver_id`) REFERENCES `bus_driver` (`driver_id`), CONSTRAINT `fk_drives_2` FOREIGN KEY (`d_bus_no`) REFERENCES `bus` (`license_plate_no`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
26	guest	CREATE TABLE `guest` (`guest_id` varchar(5) NOT NULL, `g_a_star_id` varchar(5) NOT NULL, `f_name` varchar(45) NOT NULL, `m_name` varchar(45) DEFAULT NULL, `l_name` varchar(45) NOT NULL, `street` varchar(45) NOT NULL, `apt_no` varchar(5) NOT NULL, `zip_code` varchar(5) NOT NULL, `date` date NOT NULL, `month` varchar(2) NOT NULL, PRIMARY KEY (`guest_id`,`g_a_star_id`,`date`), KEY `fk_guest_1_idx` (`g_a_star_id`), KEY `fk_guest_2_idx` (`zip_code`), CONSTRAINT `fk_guest_1` FOREIGN KEY (`g_a_star_id`) REFERENCES `a_star_passenger` (`a_star_id`), CONSTRAINT `fk_guest_2` FOREIGN KEY (`zip_code`) REFERENCES `zip_code` (`zip_code`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
27	travel_card	CREATE TABLE `travel_card` (`card_id` varchar(5) NOT NULL, `card_a_star_id` varchar(5) NOT NULL, `issue_date` date NOT NULL, `expiry_date` date NOT NULL, PRIMARY KEY (`card_id`,`card_a_star_id`), KEY `fk_travel_card_1_idx` (`card_a_star_id`),

		CONSTRAINT `fk_travel_card_1` FOREIGN KEY (`card_a_star_id`) REFERENCES `a_star_passenger` (`a_star_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
28	guest_phone	CREATE TABLE `guest_phone` (`phn_guest_id` varchar(5) NOT NULL, `phone_no` varchar(10) NOT NULL, PRIMARY KEY (`phn_guest_id`,`phone_no`), CONSTRAINT `fk_guest_phone_1` FOREIGN KEY (`phn_guest_id`) REFERENCES `guest` (`guest_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
39	promotional_ discount	CREATE TABLE `promotional_discount` (`promo_id` varchar(5) NOT NULL, `discount_percent` int NOT NULL, `description` varchar(45) NOT NULL, PRIMARY KEY (`promo_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
30	contains	CREATE TABLE `contains` (`c_card_id` varchar(5) NOT NULL, `c_promo_id` varchar(5) NOT NULL, `c_a_star_id` varchar(5) NOT NULL, PRIMARY KEY (`c_card_id`,`c_promo_id`,`c_a_star_id`), KEY `fk_contains_2_idx` (`c_promo_id`), KEY `fk_contains_3_idx` (`c_a_star_id`), CONSTRAINT `fk_contains_1` FOREIGN KEY (`c_card_id`) REFERENCES `travel_card` (`card_id`), CONSTRAINT `fk_contains_2` FOREIGN KEY (`c_promo_id`) REFERENCES `promotional_discount` (`promo_id`), CONSTRAINT `fk_contains_3` FOREIGN KEY (`c_a_star_id`) REFERENCES `a_star_passenger` (`a_star_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
31	can_have_ a_class	CREATE TABLE `can_have_a_class` (`ch_pass_id` varchar(5) NOT NULL, `ch_passenger_id` varchar(5) NOT NULL, `month` varchar(2) NOT NULL,

		PRIMARY KEY (`ch_pass_id`,`ch_passenger_id`), KEY `fk_can_have_2_idx` (`ch_passenger_id`), CONSTRAINT `fk_can_have_1` FOREIGN KEY (`ch_pass_id`) REFERENCES `pass` (`pass_id`), CONSTRAINT `fk_can_have_2` FOREIGN KEY (`ch_passenger_id`) REFERENCES `a_class_passenger` (`passenger_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
32	Can_have_ a_star	CREATE TABLE `can_have_a_star` (`ch_pass_id` varchar(5) NOT NULL, `ch_a_star_id` varchar(5) NOT NULL, `month` varchar(2) NOT NULL, PRIMARY KEY (`ch_pass_id`,`ch_a_star_id`,`month`), KEY `fk_can_have_a_star_2_idx` (`ch_a_star_id`), CONSTRAINT `fk_can_have_a_star_1` FOREIGN KEY (`ch_pass_id`) REFERENCES `pass` (`pass_id`), CONSTRAINT `fk_can_have_a_star_2` FOREIGN KEY (`ch_a_star_id`) REFERENCES `a_star_passenger` (`a_star_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
33	is_a_class_ a_star	CREATE TABLE `is_a_class_a_star` (`aa_passenger_id` varchar(5) NOT NULL, `aa_a_star_id` varchar(5) NOT NULL, PRIMARY KEY (`aa_passenger_id`), KEY `fk_is_a_class_a_star_2_idx` (`aa_a_star_id`), CONSTRAINT `fk_is_a_class_a_star_1` FOREIGN KEY (`aa_passenger_id`) REFERENCES `a_class_passenger` (`passenger_id`), CONSTRAINT `fk_is_a_class_a_star_2` FOREIGN KEY (`aa_a_star_id`) REFERENCES `a_star_passenger` (`a_star_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
34	is_emp_a_star	CREATE TABLE `is_emp_a_star` (`ea_employee_id` varchar(5) NOT NULL, `ea_a_star_id` varchar(5) NOT NULL, PRIMARY KEY (`ea_employee_id`), KEY `fk_is_emp_a_star_2_idx` (`ea_a_star_id`),

		CONSTRAINT `fk_is_emp_a_star_1` FOREIGN KEY (`ea_employee_id`) REFERENCES `employee` (`employee_id`), CONSTRAINT `fk_is_emp_a_star_2` FOREIGN KEY (`ea_a_star_id`) REFERENCES `a_star_passenger` (`a_star_id`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
35	Zipcode	CREATE TABLE `zip_code` (`zip_code` varchar(5) NOT NULL, `city` varchar(45) NOT NULL, PRIMARY KEY (`zip_code`)) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

TRIGGERS

timetable	CREATE DEFINER=`root`@`localhost` TRIGGER `timetable_BEFORE_INSERT`
	BEFORE INSERT ON `timetable` FOR EACH ROW BEGIN
	IF NEW.tt_id REGEXP'^(DT)?[0-9]{2}$' = 0 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Timetable ID must have format`DTXX`';
	ELSEIF NEW.day NOT IN ('M', 'T', 'W', 'Th', 'F', 'Sat', 'Sun') THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Day is not correct';
	ELSEIF NEW.interval NOT IN ('15', '20', '30') THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Time Interval is not correct, please enter 15, 20 or 30 minute intervals';
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER `timetable_BEFORE_UPDATE` BEFORE UPDATE ON `timetable` FOR EACH ROW BEGIN IF NEW.tt_id REGEXP'^(DT)?[0-9]{2}$' = 0 THEN SIGNAL SQLSTATE '45000' SET message_text = 'Timetable ID must have format`DTXX`'; ELSEIF NEW.day NOT IN ('M', 'T', 'W', 'Th', 'F', 'Sat', 'Sun') THEN SIGNAL SQLSTATE '45000' SET message_text = 'Day is not correct'; ELSEIF NEW.interval NOT IN ('15', '20', '30') THEN SIGNAL SQLSTATE '45000'

	SET message_text = 'Time Interval is not correct, please enter 15, 20 or 30 minute intervals'; END IF; END

zipcode	CREATE DEFINER=`root`@`localhost` TRIGGER `zip_code_BEFORE_INSERT` BEFORE INSERT ON `zip_code` FOR EACH ROW BEGIN
	if new.zip_code regexp '^[0-9]{5}$' = 0 then
	signal sqlstate '45000'
	set message_text = 'Zip code format wrong';
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER `zip_code_BEFORE_UPDATE` BEFORE UPDATE ON `zip_code` FOR EACH ROW BEGIN
	if new.zip_code regexp '^[0-9]{5}$' = 0 then
	signal sqlstate '45000'
	set message_text = 'Zip code format wrong';
	END IF;
	END

person_phone	CREATE DEFINER=`root`@`localhost` TRIGGER `person_phone_BEFORE_INSERT` BEFORE INSERT ON `person_phone` FOR EACH ROW BEGIN
	if new.phone_no regexp '^[0-9]{10}$' = 0 then
	signal sqlstate '45000'
	set message_text = 'Phone number format is wrong';
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER `person_phone_BEFORE_UPDATE` BEFORE UPDATE ON `person_phone` FOR EACH ROW BEGIN
	if new.phone_no regexp '^[0-9]{10}$' = 0 then
	signal sqlstate '45000'
	set message_text = 'Phone number format is wrong';
	END IF;
	END

guest_phone	CREATE DEFINER=`root`@`localhost` TRIGGER `guest_phone_BEFORE_INSERT`
	BEFORE INSERT ON `guest_phone` FOR EACH ROW BEGIN
	if new.phone_no regexp '^[0-9]{10}$' = 0 then
	signal sqlstate '45000'
	set message_text = 'Phone number format is wrong';
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER `guest_phone_BEFORE_UPDATE`
	BEFORE UPDATE ON `guest_phone` FOR EACH ROW BEGIN
	if new.phone_no regexp '^[0-9]{10}$' = 0 then
	signal sqlstate '45000'
	set message_text = 'Phone number format is wrong';
	END IF;
	END

employee	CREATE DEFINER=`root`@`localhost` TRIGGER `employee_BEFORE_INSERT` BEFORE INSERT ON `employee` FOR EACH ROW BEGIN
	IF new.e_type not in ('Bus Driver', 'Staff', 'Ticket Checker') then
	signal sqlstate '45000'
	set message_text = 'Employee type must be Bus Driver, Staff, Ticket Checker';
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER `employee_BEFORE_UPDATE` BEFORE UPDATE ON `employee` FOR EACH ROW BEGIN
	IF new.e_type not in ('Bus Driver', 'Staff', 'Ticket Checker') then
	signal sqlstate '45000'
	set message_text = 'Employee type must be Bus Driver, Staff, Ticket Checker';
	END IF;
	END

promotional_ discount	CREATE DEFINER=`root`@`localhost` TRIGGER
	`promotional_discount_BEFORE_INSERT` BEFORE INSERT ON `promotional_discount`
	FOR EACH ROW BEGIN
	if new.discount_percent < 0 or new.discount_percent > 100 then
	signal sqlstate '45000'
	set message_text = 'Discount percent out of range';
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER
	`promotional_discount_BEFORE_UPDATE` BEFORE UPDATE ON `promotional_discount`
	FOR EACH ROW BEGIN
	if new.discount_percent < 0 or new.discount_percent > 100 then
	signal sqlstate '45000'
	set message_text = 'Discount percent out of range';
	END IF;
	END

payment_details	CREATE DEFINER=`root`@`localhost` TRIGGER `payment_details_BEFORE_INSERT`
	BEFORE INSERT ON `payment_details` FOR EACH ROW BEGIN
	IF NEW.method not in ('cash', 'card') THEN
	SIGNAL sqlstate '45000'
	SET message_text = 'Payment method is wrong';
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER `payment_details_BEFORE_UPDATE`
	BEFORE UPDATE ON `payment_details` FOR EACH ROW BEGIN
	IF NEW.method not in ('cash', 'card') THEN
	SIGNAL sqlstate '45000'
	SET message_text = 'Payment method is wrong';
	END IF;
	END

person	CREATE DEFINER=`root`@`localhost` TRIGGER `person_check_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 of person must be greater than 16 years.';
	ELSEIF NEW.person_id REGEXP'^[P][0-9]{3}$' = 0 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'PersonID must have format`PXXX`';
	ELSEIF NEW.gender NOT IN ('M', 'F') THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Gender is not correct.';
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER `person_check_before_update` BEFORE UPDATE ON `person` FOR EACH ROW BEGIN

	IF TIMESTAMPDIFF(YEAR, NEW.dob, CURDATE()) < 16 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Age of person must be greater than 16 years.';
	ELSEIF NEW.person_id REGEXP'^[P][0-9]{3}$' = 0 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'PersonID must have format`PXXX`';
	ELSEIF NEW.gender NOT IN ('M', 'F') THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Gender is not correct.';
	END IF;
	END

bus	CREATE DEFINER=`root`@`localhost` TRIGGER `bus_BEFORE_INSERT` BEFORE INSERT ON `bus` FOR EACH ROW BEGIN
	IF new.license_plate_no regexp '^[A-Z]{3}[0-9]{4}$' = 0 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'License plate number format is incorrect';
	ELSEIF new.no_of_seats > 100 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Number of seats cannot be more than 100';
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER `bus_BEFORE_UPDATE` BEFORE UPDATE ON `bus` FOR EACH ROW BEGIN
	IF new.license_plate_no regexp '^[A-Z]{3}[0-9]{4}$' = 0 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'License plate number format is incorrect';
	ELSEIF new.no_of_seats > 100 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Number of seats cannot be more than 100';
	END IF;
	END

guest	CREATE DEFINER=`root`@`localhost` TRIGGER `guest_BEFORE_INSERT` BEFORE INSERT ON `guest` FOR EACH ROW BEGIN IF (select COUNT(*) from guest where g_a_star_id = new.g_a_star_id and `month` = new.`month`) = 4 THEN SIGNAL SQLSTATE '45000' SET message_text = 'A-Star Passenger cannot have more than 4 guests in a month';

	ELSEIF new.`month` regexp '^[0][1-9]$' = 0 THEN
	IF new.`month` regexp '^[1][0-2]$' = 0 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Month format is incorrect';
	END IF;
	END IF;
	END

	CREATE DEFINER=`root`@`localhost` TRIGGER `guest_BEFORE_UPDATE` BEFORE UPDATE ON `guest` FOR EACH ROW BEGIN
	IF(select COUNT(*) from guest where g_a_star_id = new.g_a_star_id and `month` = new.`month`) = 4 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'A-Star Passenger cannot have more than 4 guests in a month';
	ELSEIF new.`month` regexp '^[0][1-9]$' = 0 THEN
	IF NEW.`month` regexp '^[1][0-2]$' = 0 THEN
	SIGNAL SQLSTATE '45000'
	SET message_text = 'Month format is incorrect';
	END IF;
	END IF;
	END

Phase III. d. Use the Create View statement to create the following 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_a_star_passengers` AS
select person_id, f_name as first_name, l_name as last_name, issue_date as date_of_membership
from person, ticket, travel_card
where (person_id, card_id) in
(select person_id, card_id
from person, travel_card
where exists
(select *
from a_class_passenger, is_a_class_a_star
where passenger_id = aa_passenger_id and person_id = ac_person_id and aa_a_star_id = card_a_star_id)
or exists
(select *
from employee, is_emp_a_star
```

```
where employee_id = ea_employee_id and person_id = emp_person_id and ea_a_star_id = card_a_star_id))
and person_id = t_person_id
and date > date(current_date - interval 1 month)
group by t_person_id
having count(ticket_id) > 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 license_plate_no, bus_no, no_of_seats, bus_type
from bus, ticket
where license_plate_no = t_bus_no
and date > date(current_date - interval 2 month)
group by t_bus_no
order by count(ticket_id)
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_a_star_passenger` AS
select f_name as first_name, m_name as middle_name, l_name as last_name, phone_no as phone_number, person_id
from person, person_phone, ticket
where not exists
(select *
from a_class_passenger, is_a_class_a_star
where passenger_id = aa_passenger_id and person_id = ac_person_id)
and person_id = phn_person_id
and person_id = t_person_id
and date > date(current_date - interval 2 month)
group by t_person_id
having count(ticket_id) > 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 f_name as first_name, l_name as last_name, start_date, e_type as employee_type
from person, employee, ticket
where person_id = emp_person_id
and person_id = t_person_id
and date > date(current_date - interval 1 month)
group by t_person_id
having max(ticket_id);
```

Phase III. e. Answer the following Queries. Feel free to use any of the views that you created in part (d.):

1	For each employee class, list the employees belonging to that class.
	select e.e_type, e.employee_id, p.f_name, p.l_name, p.gender from employee e, person p where p.person_id = e.emp_person_id ORDER BY e.e_type;
2	Find the names of employees who are also an A-Class Passenger.
	select p.person_id, p.f_name, p.m_name, p.l_name FROM employee e, person p, a_class_passenger a where e.emp_person_id = a.ac_person_id AND e.emp_person_id = p.person_id;
3	Find the average number of bookings made by the top five A-Star Passengers.
	select avg(count) as avg_no_bookings from (select count(ticket_id) as count from top_a_star_passengers, ticket where person_id = t_person_id group by t_person_id order by count(ticket_id) limit 5) as bookings;
4	Find the Bus ID and Route names of the bus that is booked the most.
	select b.license_plate_no as bus_id, b.bus_route_id as route_id from popular_bus as p, bus as b where p.license_plate_no = b.license_plate_no;
5	Find Bus ID that has been cancelled more than 3 times in the past month.
	NOT APPLICABLE

6	Find the total number bookings for each bus in the system.
	SELECT t.t_bus_no, COUNT(t.t_bus_no) FROM ticket t GROUP BY t.t_bus_no;
7	Find the driver details who has driven every day of the past week.
	select distinct f_name as first_name, l_name as last_name, dob as date_of_birth, street, city, p.zip_code as zip_code from person as p, zip_code as z, employee, bus_driver, drives where d_driver_id = driver_id and d_employee_id = employee_id and emp_person_id = person_id and p.zip_code = z.zip_code and date > date(current_date - interval 7 day) having count(d_driver_id) = 7;
8	Find the count of passengers who booked the most popular bus.
	select count(t_person_id) from ticket where t_bus_no = (select license_plate_no from popular_bus) group by t_bus_no;
9	List all the booking details issued after the most current employee was hired.
	select * from ticket where date > (select max(start_date) from employee);
10	List all the employees that have enrolled as A-Star Passengers within a month of being employed.
	select f_name, l_name, e_type from person, employee, is_emp_a_star, travel_card where person_id = emp_person_id and employee_id = ea_employee_id and ea_a_star_id = card_a_star_id and issue_date < date(start_date + interval 1 month);
11	Find the route with the highest number of bus stops.
	SELECT stop_route_id, COUNT(*) total FROM bus_stop GROUP BY stop_route_id ORDER BY COUNT(*) DESC LIMIT 1;
12	Find the name of passengers who have been A-Star Passengers for over 5 years.
	select f_name, l_name from person, a_class_passenger, is_a_class_a_star, travel_card where person_id = ac_person_id and passenger_id = aa_passenger_id and aa_a_star_id = card_a_star_id and issue_date > date(current_date - interval 5 year);

13	Find the bookings made by the potential A-Star Passengers in the last year.
	select * from ticket where t_person_id in (select person_id from potential_a_star_passenger) and date > date(current_date - interval 1 year);