HO CHI MINH UNIVERSITY OF TECHNOLOGY AND EDUCATION

FACULTY FOR HIGH-QUALITY TRAINING

COURSE NAME: Database Management System

□□✵□□

FINAL PROJECT REPORT

Project name:

BUS TICKET BOOKING MANAGEMENT SYSTEM

Lecturer: Prof. Nguyễn Thành Sơn

Course ID: DBMS330284E_22_2_01FIE

Group: 6

Date: 2nd Sem/2022-2023

Ho Chi Minh city, May, 2023

LIST OF STUDENTS – GROUP 6

Project: Bus ticket booking management system

ID	*Full name*
21110758	Lê Xuân Cường
21110092	Bùi Quốc Thông
21110066	Phạm Vũ Bảo Nhân
21110785	Mai Nguyễn Nhật Nam

Professor's comment

Ho Chi Minh city, May …, 2023

Grading


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

Contents


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

Prologue

Firstly, we would like to express our gratitude to Prof. Nguyễn Thành Sơn for his whole-hearted instructions that helped us finish our final project for the Database Management System course. Thanks to the knowledge the professor has provided us, we were able to firmly grasp the basic knowledge and foundation for building a database management system. And through this project, our group would like to present the development process of a database management system and demonstrate by programming a related project once again.

During the process of executing this project, it will be hard to avoid mistakes. Because of that, we would love to get the professor's suggestion on improving our work so it would be more functional and complete. We wish you good health and the best of luck pursuing the path of teaching.

Finally, we would like to thank all the teachers and classmates who studied with us on this course and offered us support while we carried out our final project.


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

INTRODUCTION

In recent years, the Information and Technology (IT) area has been integrated into our society and daily lives, regardless of any field and/or occupations. It also plays an important part of booking management in Vietnam and especially in almost every country as there are many applications made to help fix problems that big organizations frequently face.

The creation of the bus ticket booking management system is the result of many developers' creativity and hard work with the aim of aiding companies in managing their businesses.

With that in mind, to better understand the application and role of Information and Technology (IT) in Database Management, we have decided on the **"Bus ticket booking management system"** as our final project.


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

CHAPTER I: SYSTEM OVERVIEW

1. Specifications

1.1. Problem statement

The bus ticket booking management system will:

- Manage the employees, passengers, bus, trips, routes easier.
- Convenient for users to check and book trips.
- Check the state and location of the trip more clearly through a map.
- More convenient for the bus company to obtain statistics: revenue, number of passengers, number of trips, employee salary, outcome, etc. per day, per month, per year.

Vehicle management: Manage travel vehicles including their location, date and time of arrival/departure, price, etc.

System management: Manage employees, drivers, customers, travel curriculum.

Statistics: Employee statistics, vehicle statistics, daily sales, etc.

1.2. Overview

A bus company needs to have a bus ticket reservation system. The bus ticket reservation system should contain the following data:

The bus company manages a lot of agents. Each agent has: agent ID, cash reserve ID, address, agent name.

Each agent has only one cash reserve. A cash reserve includes cash reserve ID and counter.

An agent has many employees. Each employee has: employee ID, position ID, account ID, agent ID, name, address, phone number, identity number, salary, email, date of birth. Each employee is provided with an account to access into the system (username and password). Each employee type has a different position.

The information of the position group contains: position ID, type.


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

There are several types:

- Administrator
- Travel planner
- Travel supervisor
- Driver
- Ticket seller
- Service guide
- Security guard
- Porter

Each position group has separate privileges. The information of the privileges group includes: privilege ID, name.

The agent manages passengers. Each passenger has: passenger ID, name, phone number, address, identity number, gender, email.

The gender attribute of passenger above has two options:

- Male
- Female

Easily manage and filter the address of stations in the general local area, there is information of places: place ID, region.

Each passenger can choose a pickup station and drop-off station. Each station has: station ID, detailed address, name, capacity, parked bus number.

The bus of each brand has: bus ID, registration number, model, capacity, status, type.

Status of the bus can be:

- Ongoing
- Idle
- Break
- Incident

Type of the bus can be:


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

- Interprovince
- Transit

Routes involving the journey have: route ID, start bus station ID, final bus station ID, travel distance.

Each trip is set up by the travel planner which includes: trip ID, drivers ID, bus ID, route ID, departure time, duration, number of booked seats, state.

The state attribute of trip above has three options:

- Waiting
- Going
- Finish

The drivers ID in the trip relation is an attribute of TRIP_DRIVER relation: trip ID, driver ID. Note that driver ID is a multivalued attribute.

The agent distributes tickets to the passenger. Each ticket has: ticket ID, trip ID, passenger ID, status, fare, type, seat number.

The status of the ticket can be:

- Available
- Bought

The type of ticket has two options:

- Seat ticket
- Sleeper ticket

The agent manages the booking transaction. Each booking transaction includes: transaction ID, ticket ID, passenger ID, employee ID, booking time.

Each driver has an employee ID number, license level and type of driver (long-haul driver and transit driver).

Each employee can take on more than 1 position.

Each passenger can book more than 1 ticket.

Each trip can have more than 1 driver.


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

The bus company provides a delivery service so that the customers can send a package without booking a ticket. They must provide information about their packages such as: mass, the phone number of sender and receiver. This package will have an ID and price. The package's price is determined by PACKAGE_PRICE_POLICY: ID, mass of package and price_per_km.

When a big event happens, the bus company has discount periods to lower the price of tickets.

Besides, the refund policy can help the passengers receive part of the fare when they cancel their trip and tickets.

2. Problem process

2.1. Booking period

** Offline booking:*

The service guide records the passenger's full field information including: their name, ID number, phone number, address, gender, email. Then, the ticket seller checks again to guarantee all the required fields are correctly fielded.

Then, the passenger picks a trip by choosing from multiple options: destination, pickup station, drop-off station, departure time, the available seat, ticket type. Options will be planned by the travel planner, so the passenger must follow this template.

Then, the ticket seller verifies the customer's selection. If valid, the ticket seller informs the passenger and waits for their confirmation. If they confirm, the ticket seller prints the ticket, gives it to them and reminds them to arrive at the correct time on the ticket. Else if they refuse, the customer needs to modify the information.

** Online booking:*

First of all, the passengers must have an account to access the bus ticket booking application. If they don't have an account yet, they have to register and log in to book the ticket. If they have an account, they only need to log in to book.


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

Afterwards, passengers will access the system to book their ticket. They will fill in the information about their name, ID number, phone number, address, gender, email, destination, pickup station, drop-off station, departure time, the available seat, ticket type. The system will send a verification code through email, then passengers fill in the app to verify their booking action.

Next, the system will provide information about the ticket.

2.2. Departure period

The passengers wait for the agent. 15 minutes before the departure time of the trip, the vehicle will take the passengers to the bus station.

At the bus station, the porter put the passengers' luggage into the trunk.

When it's time, the service guide instructs passengers to the vehicle, and provides water and tissues to them.

2.3. Drop-off period

When the bus arrives at the last bus station, the porter takes passengers' luggage from the bus and gives it to the passenger.

2.4. Ticket cancellation period

2.5. Delivery period

3. Main functions

Administrator (global):

- Add, modify, delete, authorize for positions
- Add, modify, delete employee of the position
- Statistic information about trip, the number of sold tickets

Travel planner:

- Add, modify, delete trips
- Add, modify, delete routes


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

- Add (distribute the tickets of the trip), modify, delete tickets

Travel supervisor:

- Add, delete passengers of the trip
- Report errors (trip, route, passenger, booking)

Ticket-selling:

- Add, modify, delete passenger
- Export bill
- Export ticket
- Change the state attribute of trips

Passenger (when booking online):

- Check price ticket of each route
- Check the information about booked tickets
- Book one or many tickets
- Change the information about ticket (information of passenger, the route, departure time, departure date)
- Cancel their tickets
- Export their tickets

* Authorization:

- Admin: Full control on the whole system – Global privilege
- […] (Other privileges): Local privilege

4. Attributes and operations reference

Bus relation:

* ID attribute:

Format:

bu[model]_[number]

[ ] : Ignore this notation


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

Example: **bu5272f29s28_5**

- model: 5272f29s28
- number: 5

Note:

- model: Model of the bus
- number: Order of the bus

* Capacity attribute of bus relation is fixed (default of model).

Trip relation:

* ID attribute:

Format:

tr[ID_route]_[departure_time]

[ ] : Ignore this notation

Example: **trr78_20230918**

- ID_route: r78
- departure_time: 2023/09/18

Note:

- ID_route: ID of route in route relation
- departure_time: Written in continuous form (No special characters)

* Duration attribute is calculated using the following formula:

Duration = distance / average_speed

Note:

- distance: The distance attribute of route relation
- average_speed: The moving value is statistic by system


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

* booked_seat attribute: Updated after every insert or delete statement of the passenger of the trip. It must satisfy the following rule:

$$booked\_seat \leq capacity - inherent\_seat$$

Note:

- capacity: Get from the bus relation
- inherent_seat: This value is set by the planner or ticket seller

Ticket relation:

* ID attribute:

Format:

ti[type]_[seat_number]

[ ] : Ignore this notation

Example: **tiseat_a14**

- type: seat
- seat_number: a14

Note:

- type: Ticket type
- seat_number: Number of tickets

* Fare attribute: Set by the travel planner.

* Seat number attribute has a limit: 0 < seat_number < 15

Format:

{[A | B | C]} {[1 | 2 | ...]}

Note:


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

- A: Floor 1
- B: Floor 2
- C: Rear seats
- 1, 2, …, 5: column 1 (after driver's seat)
- 6, 7, …,10: column 2
- 11, 12, …, 15: column 3
- For C (rear seat): 1,2, …, 5 (left to right)

* Status attribute:

- Available
- Bought

* Type attribute:

- Seat
- Bed

Agent relation:

* ID attribute:

Format:

a[ID_agent]

Employee relation:

* ID attribute:

Format:

e[ID_employee]

* Salary attribute: Unit in (đồng/VND)

* Birthdate:

Format:


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

dd/MM/yyyy

Driver relation:

* License_level attribute:

List of levels:

- A1
- A2
- B1
- B2
- … (To be updated)

* Type attribute:

Available driver types:

- Transit
- Interprovince

Route relation:

* ID attribute:

Format:

r[route_number]

[ ] : Ignore this notation

Example: **r78**

- route_number: 78

Note:

- route_number: The route number

* Distance attribute: Set manually by the system


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

Booking relation:

* ID attribute:

Format:

b[booking_time]

[ ] : Ignore this notation

Example: **b20230719**

- booking_time: 2023/07/19

Note:

- booking_time: The booking time attribute of this relation. Written in continuous form (No special characters)

* Booking_time attribute: Written in GMT+7 format.

Place relation:

* ID attribute: ID of province

* Region attribute: Name of province

BusStation relation:

* Cur_bus_number attribute: Updated after the related trip state change.

* Capacity attribute: Maximum capacity of the station (For buses)

Passenger relation:

* ID attribute:

Format:

pass[ID_passenger]

* Phone_number attribute: 10-11-digit length.


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

* Gender attribute: Male/Female

SystemAccount relation:

* Username attribute:

Format:

admin: sysad_[optional_name] system employees: sys_[optional_name]

[ ] : Ignore this notation

* Password attribute:

Conditions:

- At least 6 characters in length
- All characters must not be the same

Position relation:

* ID attribute:

Format:

p[ID_position]_[type]

* Type attribute:

List of account types:

- admin: Administrator
- plan: Travel planner
- visor: Travel supervisor
- driver: Driver
- seller: Ticket seller
- guide: Service guide
- guard: Security guard


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

- porter: Porter

Privilege relation:

* Name attribute:

List of names:

- Add passenger
- Add trip
- Add route
- Add bus
- Add ticket
- All control
- … (To be updated)

CashReserve relation:

* Counter: Updated after a transaction (booking) occurs.

TripDriver relation:

* ID_driver attribute references the ID_employee of employee relation where position is driver.

Package relation:

*ID attribute:

Format:

pack[ID_package]

Example: **packa1231bs**

- ID_package: a1231bs

PackagePricePolicy relation:

* ID attribute:

Format:


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

pacp[ID_policy]

Example: **pacp12a32**

- ID_policy: 12a32

* Price_per_km attribute: Set by system

* Mass attribute: Set by system

Event relation:

* Discount_type attribute:

Types of discounts:

- Special
- Normal
- Sale (Can be inserted, deleted, updated in some cases)

* Discount_percent attribute is calculated using the following formula:

Unit % (number / 100)

Refund relation:

* Refund_name attribute: Cancel

* Refund_percent attribute is calculated using the following formula:

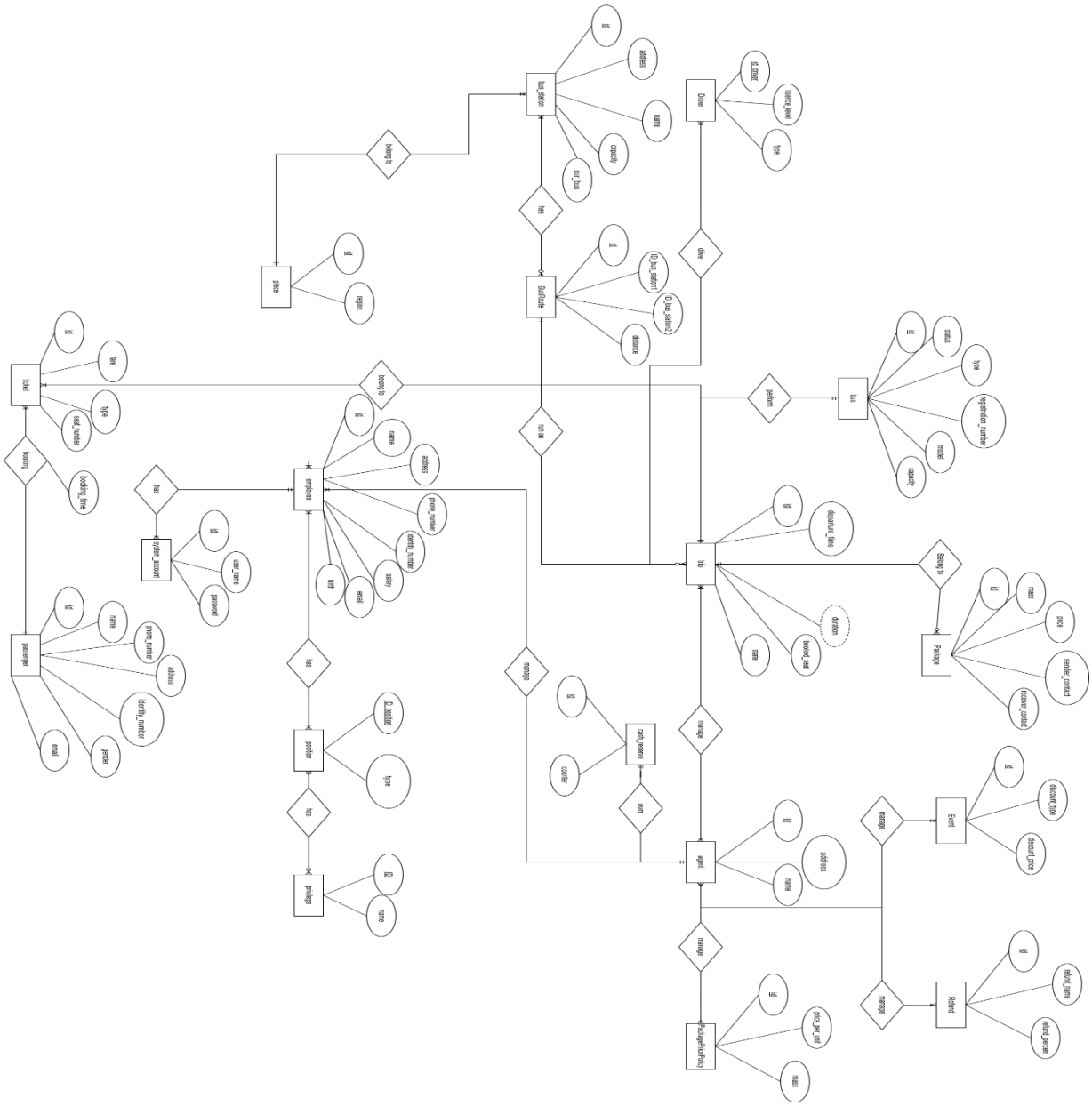
Unit % (number / 100)


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

CHAPTER 2: SYSTEM ANALYSIS AND DESIGN

1. Conceptual level database design

From the necessary data in description of the problem, the following Entity Relationship Model (ERD) is formed.




```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

Sharp image: ERD sharp image (busticketbookingerd.netlify.app)

2. Logical level database design

From the Entity Relationship Model (ERD), we have:

- Bus (id_bus, registration_number, model, capacity, status, type)
- Trip (id_trip, id_bus, id_route, departure_time, duration, booked_seat, status)
- TripDriver (id_trip, id_driver)
- Agent (id_agent, id_place, id_cash_reserve, address, name)
- CashReserve (id_cash_reserve, counter)
- BusStation (id_bus_station, id_place, address, name, bus_capacity, count_current_bus)
- Driver (id_driver, license_level, type, state)
- Route (id_route, id_start_station, id_end_station, distance)
- Place (id_place, region)
- Employee (id_employee, id_account, id_agent, name, address, phone_number, identity_number, salary, email, birthdate)
- Position (id_position, type)
- Privilege (id_privilege, name)
- SystemAccount (id_account, user_name, password)
- Ticket (id_ticket, id_trip, status, fare, type, seat_number)
- Event (id_event, discount_type, discounted_price)
- Refund (id_refund, refund_name, refund_percent)
- Passenger (id_passenger, name, phone_number, address, identity_number, gender, email)
- Booking (id_booking, id_ticket, id_passenger, id_employee, booking_time)
- Package (id_package, id_trip, mass, price, sender_contact, receiver_contact, id_route)
- PackagePricePolicy (id_policy, price_per_km, mass_unit)
- Agent_Trip (id_agent, id_trip)
- Agent_Event (id_agent, id_event)


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

- Agent_Refund (id_agent, id_refund)
- Agent_Policy (id_agent, id_policy)
- BusRoute_BusStation (id_busRoute, id_busStation)
- Employee_Position (id_employee, id_position)
- Employee_Ticket (id_employee, id_ticket)
- Position_Privilege (id_position, id_privilege)

3. Required constraints

No.	Table	Constraint
1	Bus	Primary key: id_bus
2	Trip	Primary key: id_trip
3	TripDriver	Foreign keys: id_trip references Trip(id_trip), id_driver references Driver(id_driver), respectively
4	Driver	Primary key: id_driver
5	Agent	Primary key: id_agent Foreign key: id_cash_reserve references CashReserve(id_cash_reserve)
6	CashReserve	Primary key: id_cash_reserve


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

7	BusStation	Primary key: id_bus_station
8	BusRoute	Primary key: id_route Foreign keys: id_bus_station1 and id_bus_station2 references BusStation(id_bus_station)
9	Place	Primary key: id_place
10	Employee	Primary key: id_employee Foreign keys: id_account references SystemAccount(id_account), id_position references Position(id_position), id_agent references Agent(id_agent), respectively
11	Position	Primary key: id_position
12	Privilege	Primary key: id_privilege Foreign key: id_position references Position(id_position)
13	SystemAccount	Primary key: id_account
14	Ticket	Primary key: id_ticket


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

		Foreign key: id_trip references Trip(id_trip)
15	Event	No constraints
16	Refund	No constraints
17	Passenger	Primary key: id_passenger
18	Booking	Primary key: id_booking Foreign keys: id_ticket references Ticket(id_ticket), id_passenger references Passenger(id_passenger), id_employee references Employee(id_employee), respectively
19	Package	Primary key: id_package Foreign keys: id_route references Route(id_route), id_trip references Trip(id_trip), respectively
20	PackagePricePolicy	Primary key: id_policy
21	AgentAndTrip	Primary key: id_agent, id_trip Foreign key: id_agent references Agent(id_agent), id_trip references Trip(id_trip)


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

22	AgentAndEvent	Primary key: id_agent, id_event Foreign key: id_agent references Agent(id_agent) id_event references Event(id_event)
23	AgentAndRefund	Primary key: id_agent, id_refund Foreign key: id_agent references Agent(id_agent), id_refund references Refund(id_refund)
24	AgentAndPPP	Primary key: id_agent, id_ppp Foreign key: id_agent references Agent(id_agent), id_ppp references PackagePricePolicy(id_policy)
25	BusRouteAndBusStation	Primary key: id_busRoute, id_busStation Foreign key: id_busRoute references BusRoute(id_route), id_busStation references BusStation(id_busStation)
26	EmployeeAndPosition	Primary key: id_employee, id_postion Foreign key: id_employee references Employee(id_employee), id_position references Position(id_position)
27	EmployeeAndTicket	Primary key: id_employee; id_ticket


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

		Foreign key: id_employee references Employee(id_employee), id_ticket references Ticket(id_ticket)
28	PositionAndPrivilege	Primary key: id_position, id_privilege Foreign key: id_position references Position(id_position), id_privilege references Privilege(id_privilege)

4. Database settings and constraints

[BUS]

```
CREATE TABLE Bus(
	ID_bus nchar(10) NOT NULL PRIMARY KEY,
	registration_number nchar(12) NOT NULL,
	model nvarchar(20) NOT NULL,
	capacity tinyint NOT NULL,
	status nvarchar(10) NULL,
	type nvarchar(20) NULL,
)
```

[TRIP]

```
CREATE TABLE Trip(
	ID_trip nchar(10) NOT NULL PRIMARY KEY,
	ID_bus nchar(10) NOT NULL REFERENCES Bus(ID_bus),
	ID_route nchar(10) NOT NULL REFERENCES Route(ID_route),
	departure_time datetime NOT NULL,
	duration nvarchar(20) NULL,
	booked_seat TINYINT NOT NULL,
	status nvarchar(10) NULL
)
```

[TRIP_DRIVER]

```
CREATE TABLE TripDriver(
	ID_trip nchar(10) NOT NULL REFERENCES Trip(ID_trip),
	ID_driver nchar(10) NOT NULL REFERENCES Ddriver(ID_driver)
)
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

[AGENT]

```
CREATE TABLE Agent(
	ID_agent nchar(10) NOT NULL PRIMARY KEY,
	address nvarchar(max) NOT NULL,
	name nvarchar(30) NOT NULL,
	ID_cash_reserve nchar(10) NOT NULL REFERENCES CashReserve(ID_cash_reserve)
)
```

[CASH_RESERVE]

```
CREATE TABLE CashReserve(
	ID_cash_reserve nchar(10) NOT NULL PRIMARY KEY,
	counter real NOT NULL
)
```

[BUS_STATION]

```
CREATE TABLE BusStation(
	ID_bus_station nchar(10) NOT NULL PRIMARY KEY,
	address nvarchar(max) NOT NULL,
	name nvarchar(30) NOT NULL,
	capacity tinyint NOT NULL,
	cur_bus_number nchar(10) NOT NULL
)
```

[DRIVER]

[ROUTE]

```
CREATE TABLE Route(
	ID_route nchar(10) NOT NULL PRIMARY KEY,
	ID_bus_station1 nchar(10) NOT NULL REFERENCES BusStation(ID_bus_station),
	ID_bus_station2 nchar(10) NOT NULL REFERENCES BusStation(ID_bus_station),
	distance float NULL
)
```

[PLACE]

```
CREATE TABLE Place(
	ID_place nchar(10) NOT NULL PRIMARY KEY,
	region nvarchar(20) NOT NULL
)
```

[EMPLOYEE]


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
CREATE TABLE Employee(
	ID_employee nchar(10) NOT NULL PRIMARY KEY,
	ID_position nchar(10) NOT NULL REFERENCES Position(ID_position),
	ID_account nchar(10) NOT NULL REFERENCES SystemAccount(ID_account),
	ID_agent nchar(10) NOT NULL REFERENCES Agent(ID_agent),
	name nvarchar(30) NOT NULL,
	address nvarchar(max) NOT NULL,
	phone_number nchar(15) NOT NULL,
	identity_number nchar(10) NOT NULL,
	salary money NOT NULL,
	email nvarchar(30) NULL,
	birthdate date NOT NULL
)
```

[POSITION]

```
CREATE TABLE Position(
	ID_position nchar(10) NOT NULL PRIMARY KEY,
	ID_privilege nchar(10) NOT NULL REFERENCES Privilege(ID_privilege),
	type nvarchar(20) NULL
)
```

[PRIVILEGE]

```
CREATE TABLE Privilege(
	ID_privilege nchar(10) NOT NULL PRIMARY KEY,
	name nvarchar(30) NOT NULL
)
```

[SYSTEM_ACCOUNT]

```
CREATE TABLE SystemAccount(
	ID_account nchar(10) NOT NULL PRIMARY KEY,
	user_name nvarchar(30) NOT NULL,
	password nvarchar(20) NOT NULL
)
```

[TICKET]

```
CREATE TABLE Ticket(
	ID_ticket nchar(10) NOT NULL PRIMARY KEY,
	ID_trip nchar(10) NOT NULL REFERENCES Trip(ID_trip),
	status nvarchar(10) NULL,
	fare money NOT NULL,
	type nvarchar(20) NULL,
	seat_number nchar(5) NOT NULL
)
```

[EVENT]


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
CREATE TABLE Event(
	discount_type nvarchar(20) NOT NULL,
	discounted_price float NOT NULL
)
```

[REFUND]

```
CREATE TABLE Refund(
	refund_name nvarchar(20) NOT NULL,
	refund_percent float NOT NULL
)
```

[PASSENGER]

```
CREATE TABLE Passenger(
	ID_passenger nchar(10) NOT NULL PRIMARY KEY,
	name nvarchar(30) NOT NULL,
	phone_number nchar(15) NOT NULL,
	address nvarchar(max) NOT NULL,
	identity_number nchar(10) NOT NULL,
	gender nchar(10) NOT NULL,
	email nvarchar(max) NULL
)
```

[BOOKING]

```
CREATE TABLE Booking(
	ID_booking nchar(10) NOT NULL PRIMARY KEY,
	ID_ticket nchar(10) NOT NULL REFERENCES Ticket(ID_ticket),
	ID_passenger nchar(10) NOT NULL REFERENCES Passenger(ID_passenger),
	ID_employee nchar(10) NOT NULL REFERENCES Employee(ID_employee),
	booking_time datetime NOT NULL
)
```

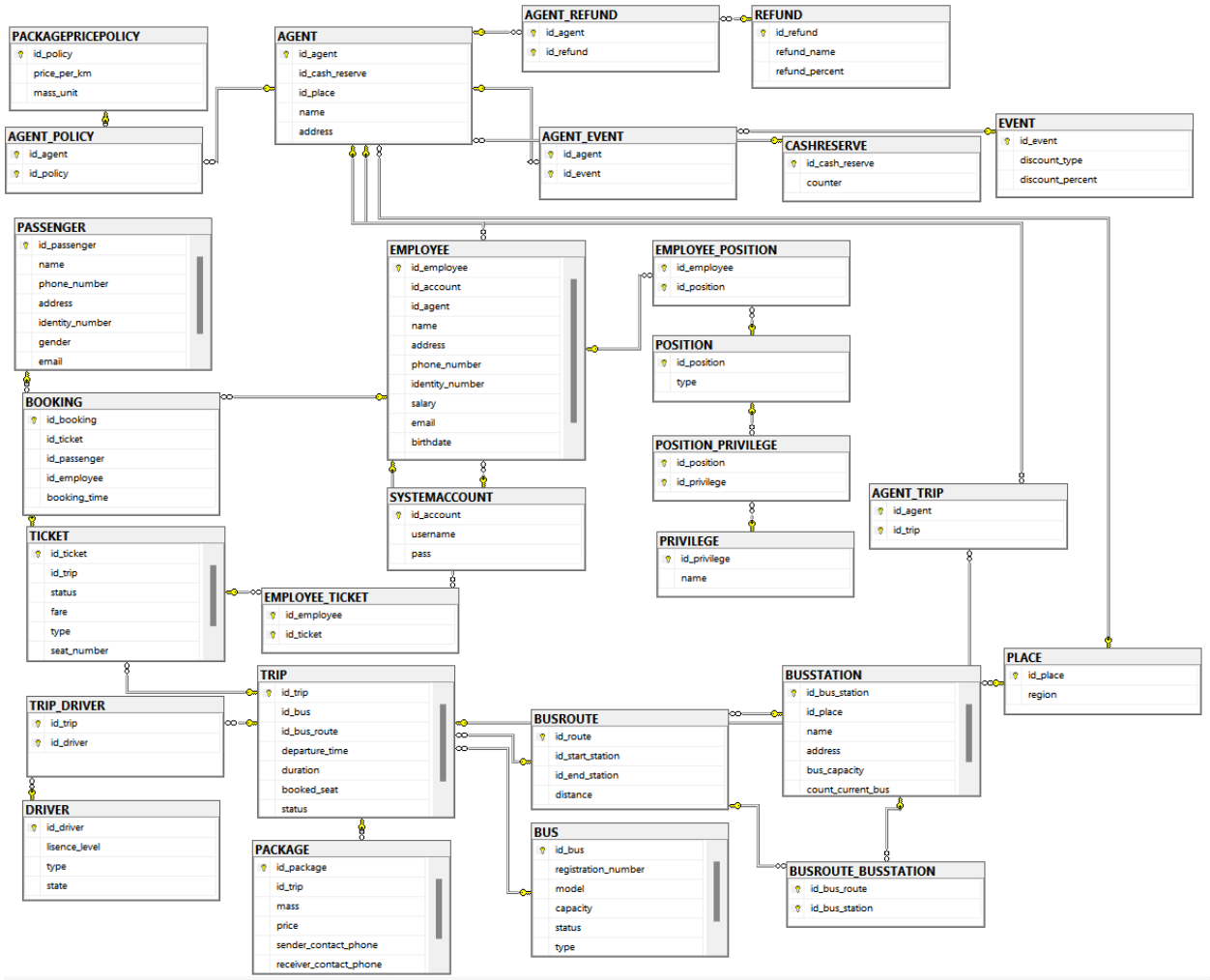
[PACKAGE]

```
CREATE TABLE Package(
	ID_package nchar(10) NOT NULL PRIMARY KEY,
	ID_route nchar(10) NOT NULL REFERENCES Route(ID_route),
	ID_trip nchar(10) NOT NULL REFERENCES Trip(ID_trip),
	mass float NOT NULL,
	price money NOT NULL,
	sender_contact nchar(15) NOT NULL,
	receiver_contact nchar(15) NOT NULL
)
```

[PACKAGE_PRICE_POLICY]


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
CREATE TABLE PackagePricePolicy(
	ID_policy nchar(10) NOT NULL PRIMARY KEY,
	price_per_km money NOT NULL,
	mass float NOT NULL,
)
```



5. Other constraints

Constrain bus identity after add one

```
-- Set constraint bus identity  automatically.
USE BusManagement
ALTER TABLE Bus
ADD CONSTRAINT AUTO_ID_Bus
DEFAULT DBO.AUTO_ID_Bus() FOR ID_bus;
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
GO
```

Constrain passenger identity and ticket identity after customer buy a ticket

```
 -- Set constraint ticket identity automatically.
USE BusManagement
ALTER TABLE TICKET
ADD CONSTRAINT AUTO_ID_ticket
DEFAULT DBO.AUTO_ID_ticket() FOR ID_ticket;
GO

 -- Set constraint passenger identity  automatically.
USE BusManagement
ALTER TABLE PASSENGER
ADD CONSTRAINT AUTO_ID_passenger
DEFAULT DBO.AUTO_ID_passenger() FOR ID_passenger;
GO
```

Constrain trip identity after add one

```
 -- Set constraint trip identity  automatically.
USE BusManagement
ALTER TABLE TRIP
ADD CONSTRAINT AUTO_ID_trip
DEFAULT DBO.AUTO_ID_trip() FOR ID_trip;
GO
```

Constrain route identity after add one

```
 -- Set constraint route identity  automatically.
USE BusManagement
ALTER TABLE BUS_ROUTE
ADD CONSTRAINT AUTO_ID_route
DEFAULT DBO.AUTO_ID_route() FOR ID_route;
GO
```

Constrain position identity after add one

```
 -- Set constraint route identity  automatically.
USE BusManagement
ALTER TABLE POSITION
ADD CONSTRAINT AUTO_ID_position
DEFAULT DBO.AUTO_ID_position() FOR ID_position;
GO
```



```
CREATE TABLE Driver(
       ID_driver nchar(10) NOT NULL PRIMARY KEY,
       license_level nchar(10) NULL,
       type nvarchar(20) NULL,
)
```

Constrain employee identity after add one

```
-- Set constraint route identity  automatically.
USE BusManagement
ALTER TABLE EMPLOYEE
ADD CONSTRAINT AUTO_ID_employee
DEFAULT DBO.AUTO_ID_employee() FOR ID_employee;
GO
```

Constrain agent identity after add one

```
-- Set constraint bus identity  automatically.
USE BusManagement
ALTER TABLE Agent
ADD CONSTRAINT AUTO_ID_Agent
DEFAULT DBO.AUTO_ID_Agent() FOR ID_agent;
GO
```

Constrain booking identity after add one

```
-- Set constraint bus identity  automatically.
USE BusManagement
ALTER TABLE Booking
ADD CONSTRAINT AUTO_ID_Booking
DEFAULT DBO.AUTO_ID_Booking() FOR ID_booking;
GO
```

Constrain package identity after add one

```
-- Set constraint bus identity  automatically.
USE BusManagement
ALTER TABLE Package
ADD CONSTRAINT AUTO_ID_Package
DEFAULT DBO.AUTO_ID_Package() FOR ID_package;
GO
```

Constrain package price policy identity after add one

```
-- Set constraint bus identity  automatically.
USE BusManagement
ALTER TABLE PackagePricePolicy
ADD CONSTRAINT AUTO_ID_PackagePricePolicy
DEFAULT DBO.AUTO_ID_PackagePricePolicy() FOR ID_policy;
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
GO
```

6. Trigger to check for constraints

***Update state of an employee on Employee and Driver relation**

```
CREATE TRIGGER tr_employee_update_stateEmployee
ON Employee
AFTER UPDATE
AS
BEGIN
    DECLARE @employee_id CHAR
    DECLARE @new_state CHAR
    SELECT @employee_id = ID_employee, @new_state = state FROM inserted
    -- Kiểm tra nếu trạng thái mới của nhân viên là 0
    IF @new_state = 0
    BEGIN
    -- Cập nhật trạng thái của nhân viên trong bảng Employee thành 0
    UPDATE Employee SET state = 0 WHERE ID_employee = @employee_id
    UPDATE Driver SET state = 0 WHERE ID_driver = @employee_id
    END
```



```
CREATE TABLE Driver(
    ID_driver nchar(10) NOT NULL PRIMARY KEY,
    license_level nchar(10) NULL,
    type nvarchar(20) NULL,
)
```

```
        SELECT state FROM Driver WHERE ID_driver = @employee_id


END
```

*** Delete account of employee and update state of Employee when his/her state =0**

```
CREATE TRIGGER tr_employee_deleteAccount

ON Employee

AFTER UPDATE

AS

BEGIN

        DECLARE @account_id CHAR

        DECLARE @new_state CHAR

        SELECT @account_id = ID_account, @new_state = inserted.state FROM inserted



        IF @new_state = 0

        BEGIN

        -- Delete account

        DELETE FROM SystemAccount WHERE @account_id = ID_account

        END
```



```
CREATE TABLE Driver(
        ID_driver nchar(10) NOT NULL PRIMARY KEY,
        license_level nchar(10) NULL,
        type nvarchar(20) NULL,
)
```

END

7. Views

View list of active employee which is working

CREATE VIEW [dbo].[ActiveEmployee] AS

SELECT Employee.ID_employee, Employee_ID_account, Employee.name, Employee.address, Employee.phone_number, Employee.identity_number, Employee.salary, Employee.email,

Employee.birthday, Agent.name, Position.type

FROM

Employee AS temp1 INNER JOIN Agent AS temp 2

ON temp1.ID_agent = temp2.ID_agent

INNER JOIN Position as temp3

ON temp1.ID_position = temp3.ID_position

WHERE temp1.status = 1

View list of waiting trip:

CREATE VIEW [dbo].[WaitingTrip] AS


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
temp1.ID_trip, temp1.departure_time, temp1.duration, temp1.booked_seat,
temp1.registration_number, temp1.type,

temp2.name AS start_point, temp3.name AS end_point

FROM

((SELECT Trip.*, Bus.registration_number, Bus.type

FROM Trip INNER JOIN Bus

ON Trip.ID_bus = Bus.ID_bus) AS temp0

INNER JOIN BusRoute

ON BusRoute.ID_route = temp0.ID_route) AS temp1

INNER JOIN (

SELECT temp1.ID_route, temp1.ID_bus_station1, BusStation.name

FROM temp1 INNER JOIN BusStation

ON temp1.ID_bus_station1 = BusStation.ID_bus_station

) AS temp2

ON temp1.ID_route = temp2.ID_route

INNER JOIN (

SELECT temp1.ID_route, temp1.ID_bus_station2, BusStation.name

FROM temp1 INNER JOIN BusStation

ON temp1.ID_bus_station2 = BusStation.ID_bus_station

) AS temp3
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
ON temp1.ID_route = temp3.ID_route

WHERE temp1.status = 'Waiting'
```

View list of going trip:

```
CREATE VIEW [dbo].[GoingTrip] AS

temp1.ID_trip, temp1.departure_time, temp1.duration, temp1.booked_seat,
temp1.registration_number, temp1.type,

temp2.name AS start_point, temp3.name AS end_point

FROM

((SELECT Trip.*, Bus.registration_number, Bus.type

FROM Trip INNER JOIN Bus

ON Trip.ID_bus = Bus.ID_bus) AS temp0

INNER JOIN BusRoute

ON BusRoute.ID_route = temp0.ID_route) AS temp1

INNER JOIN (

SELECT temp1.ID_route, temp1.ID_bus_station1, BusStation.name

FROM temp1 INNER JOIN BusStation

ON temp1.ID_bus_station1 = BusStation.ID_bus_station

) AS temp2
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
ON temp1.ID_route = temp2.ID_route
INNER JOIN (
SELECT temp1.ID_route, temp1.ID_bus_station2, BusStation.name
FROM temp1 INNER JOIN BusStation
ON temp1.ID_bus_station2 = BusStation.ID_bus_station
) AS temp3
ON temp1.ID_route = temp3.ID_route
WHERE temp1.status = 'Going'
```

View list of finished trip:

```
CREATE VIEW [dbo].[FinishTrip] AS
temp1.ID_trip, temp1.departure_time, temp1.duration, temp1.booked_seat, temp1.registration_number, temp1.type,
temp2.name AS start_point, temp3.name AS end_point
FROM
((SELECT Trip.*, Bus.registration_number, Bus.type
FROM Trip INNER JOIN Bus
ON Trip.ID_bus = Bus.ID_bus) AS temp0
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
INNER JOIN BusRoute
ON BusRoute.ID_route = temp0.ID_route) AS temp1
INNER JOIN (
SELECT temp1.ID_route, temp1.ID_bus_station1, BusStation.name
FROM temp1 INNER JOIN BusStation
ON temp1.ID_bus_station1 = BusStation.ID_bus_station
) AS temp2
ON temp1.ID_route = temp2.ID_route
INNER JOIN (
SELECT temp1.ID_route, temp1.ID_bus_station2, BusStation.name
FROM temp1 INNER JOIN BusStation
ON temp1.ID_bus_station2 = BusStation.ID_bus_station
) AS temp3
ON temp1.ID_route = temp3.ID_route
WHERE temp1.status = 'Finish'
```

View about list of idle interprovince bus:

```
CREATE VIEW [dbo].[IdleInterprovinceBus]
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
AS

SELECT rel.ID_bus, rel.registration_number, rel.model, rel.capacity

FROM Bus as rel

WHERE Bus.status = 'idle' AND Bus.type = 'interprovince'
```

View about list of break interprovince bus:

```
CREATE VIEW [dbo].[BreakInterprovinceBus]

AS

SELECT rel.ID_bus, rel.registration_number, rel.model, rel.capacity

FROM Bus as rel

WHERE Bus.status = 'break' AND Bus.type = 'interprovince'
```

View about list of incident interprovince bus:

```
CREATE VIEW [dbo].[IncidentInterprovinceBus]

AS

SELECT rel.ID_bus, rel.registration_number, rel.model, rel.capacity

FROM Bus as rel

WHERE Bus.status = 'incident' AND Bus.type = 'interprovince'
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

View about list of ongoing interprovince bus:

```
CREATE VIEW [dbo].[OnGoingInterprovinceBus]
AS
SELECT rel.ID_bus, rel.registration_number, rel.model, rel.capacity
FROM Bus AS rel INNER JOIN (
Select Trip.ID_bus, Trip.ID_trip
FROM Trip
WHERE Trip.status = 'going'
) as rel2
ON rel.ID_bus = rel2.ID_bus
WHERE Bus.status = 'ongoing' AND Bus.type = 'interprovince'
```

View about list of passenger booking information (in detail):

```
CREATE VIEW [dbo].[BookingInfor]
AS
SELECT rel.ID_booking, temp1.ID_ticket, temp1.ID_trip, temp1.seat_number, temp1.type, temp1.fare, temp2.name AS passenger_name, temp2.phone_number AS passenger_phone_number, temp2.address AS passenger_address, temp2.email
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
AS passenger_email, temp2.gender AS passenger_gender, temp3.name AS employee_name, temp3.phone_number AS employee_phonee_number

FROM Booking AS rel INNER JOIN Ticket AS temp1

ON rel.ID_ticket = temp1.ID_ticket

INNER JOIN Passenger AS temp2

ON rel.ID_passenger = temp2.ID_passenger

INNER JOIN Employee AS temp3

ON rel.ID_employee = temp3.ID_employee
```

View about list of currently bus route information (in detail):

```
CREATE VIEW  [dbo].[BusRouteInfor]

AS

SELECT rel.ID_route, temp1.start_point, temp2.end_point, rel.distance

FROM BusRoute AS rel INNER JOIN (

SELECT rel.ID_Route, BusStation.name as start_point

FROM rel INNER JOIN BusStation

ON rel.ID_bus_station1 = BusStation.ID_bus_station

) AS temp1

ON rel.ID_route = temp1.ID_route
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
INNER JOIN (
SELECT rel.ID_Route, BusStation.name as end_point
FROM rel INNER JOIN BusStation
ON rel.ID_bus_station2 = BusStation.ID_bus_station
) AS temp2
ON rel.ID_route = temp2.ID_route
```

View about list of Going trip driver:

```
CREATE VIEW [dbo].[WaitingTripDriverInfor]
AS
SELECT rel.*, temp2.ID_driver, temp2.name AS driver_name, temp2.phone_number AS driver_phone_number
FROM WaitingTrip AS rel INNER JOIN TripDriver AS temp1
ON rel.ID_trip = temp1.ID_trip
INNER JOIN (
SELECT Driver.ID_driver, Employee.name, Employee.phone_number
FROM Driver INNER JOIN Employee
ON Driver.ID_driver = Employee.ID_employee
) AS temp2
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
ON temp1.ID_driver = temp2.ID_driver
```

View about list of Going trip driver:

```
CREATE VIEW [dbo].[GoingTripDriverInfor]

AS

SELECT rel.*, temp2.ID_driver, temp2.name AS driver_name, temp2.phone_number AS driver_phone_number

FROM GoingTrip AS rel INNER JOIN TripDriver AS temp1

ON rel.ID_trip = temp1.ID_trip

INNER JOIN (

SELECT Driver.ID_driver, Employee.name, Employee.phone_number

FROM Driver INNER JOIN Employee

ON Driver.ID_driver = Employee.ID_employee

) AS temp2

ON temp1.ID_driver = temp2.ID_driver
```

View about list of Going trip driver:

```
CREATE VIEW [dbo].[FinishTripDriverInfor]

AS
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
SELECT rel.*, temp2.ID_driver, temp2.name AS driver_name, temp2.phone_number AS driver_phone_number

FROM FinishTrip AS rel INNER JOIN TripDriver AS temp1

ON rel.ID_trip = temp1.ID_trip

INNER JOIN (

SELECT Driver.ID_driver, Employee.name, Employee.phone_number

FROM Driver INNER JOIN Employee

ON Driver.ID_driver = Employee.ID_employee

) AS temp2

ON temp1.ID_driver = temp2.ID_driver
```

View about list of employee accounts:

```
CREATE VIEW [dbo].[EmployeeAccount]

AS

SELECT temp1.ID_employee, temp1.name, temp2.username, temp2.password

FROM Employee AS temp1, SystemAccount AS temp2
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

View about list of waiting trip which is still empty chairs

```
CREATE VIEW [dbo].[TripWithAvailableChair]
AS
SELECT temp1.*, temp2.capacity - temp1.booked_seat AS available_position
FROM WaitingTrip AS temp1 INNER JOIN (
SELECT  Bus.capacity, Trip.ID_trip
FROM Trip INNER JOIN Bus
ON Trip.ID_bus = Bus.ID_bus
) as temp2
ON temp1.ID_trip temp2.ID_trip
WHERE temp2.capacity - temp1.booked_seat > 0
```

View about list of waiting trip which is full chairs

```
CREATE VIEW [dbo].[TripWithAvailableChair]
AS
SELECT temp1.*, temp2.capacity - temp1.booked_seat AS available_position
FROM WaitingTrip AS temp1 INNER JOIN (
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

```
SELECT  Bus.capacity, Trip.ID_trip

FROM Trip INNER JOIN Bus

ON Trip.ID_bus = Bus.ID_bus

) as temp2

ON temp1.ID_trip temp2.ID_trip

WHERE temp2.capacity - temp1.booked_seat = 0
```



```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```

CHAPTER 3: DESIGNING FUNCTIONS

1. Connect to database


```
CREATE TABLE Driver(
	ID_driver nchar(10) NOT NULL PRIMARY KEY,
	license_level nchar(10) NULL,
	type nvarchar(20) NULL,
)
```