The purpose of our system

- *What is the purpose of the database? Why is it needed? What should it do?*

The purpose of our system is to ensure the efficient and easy data manipulation of the parking lot system for a particular organization. Our system's main idea is about storing certain essential information about the customer of the shopping center's parking lot and making sure that the amount of the parking cost is consistent with the amount of the parking rate for that particular customer.

Description: In our database we will have 10 distinct entities which are parking lot, block, floor, parking slot, vehicle, parking card, customer, parking rate, payment and gate. Each entity will have from 2 to 7 attributes. We will be dealing with the parking lots of several shopping centers and with customers who parked their vehicles at those parking lots.

- *Who are the users and what are their information needs?*

The users of our system are the administration staff of the parking lot of the certain shopping center.

Their information needs are the amount of time customers spend in the parking lot, the amount they pay for that parking, and the place where that customer parks his or her vehicle.

- *What are the problems that the system should solve?*

Our system is designed in such a way that it is easy to add and retrieve data, it requires only the essential information about the customer, and it ensures efficient parking regulation of the parking lot management system.

So, the problems that the system should solve are the complicated structure, the data redundancy, and the poor parking regulation of the parking lot management system.

- *What input data is available to the database?*

Date and time of entry, time of departure, vehicle ID and type, parking card ID, method of payment, status of payment, and the amount paid.

- *What kind of information should be stored in the database?*

In our database system, we will be dealing with several shopping centers and their parking lots. We will store such information about the customer such as the date of entry, time of entry, time of departure, vehicle ID and type, parking card ID, payment type and status, and the amount paid. We will also store some information about the parking lot such as the name, address of the organization and the number and list of floors, blocks, and parking slots.

Scenario: At the entrance to the shopping center's parking lot (for example, Mega center "Alma-Ata") a parking card, which opens the gate, is issued to the customer. The database stores main information about the customer such as the date and time of entry, as well as the other type of essential information. The payment amount is calculated according to these rules: Staying on the parking from 0-15 minutes - Free of charge; from 15-60 minutes - 100 tenge; for each subsequent hour - 100 tenge. When leaving, the customer enters the parking card and, if the payment is made, the gate opens.

Business rules:

The parking lot contains one or more floors.

Floor contains one or more blocks.

Block contains one or more parking slots.

One customer parks one vehicle.

One vehicle occupies one parking slot.

There are two types of parking slots: two-wheel and four-wheel.

The parking lot has 2 gates: one at the entrance panel and one at exit panel.

Customers receive a parking card at the entrance panel.

Each customer receives a unique parking card.

Each parking card has a unique parking rate (amount of time passed since the start of the parking).

Payment amount is calculated according to the parking rate.

Parking rate rules: Staying on the parking from 0-15 minutes – Free of charge. From 15-60 minutes – 100 tenge. For each subsequent hour – 100 tenge.

Customer can choose only one type of payment method: cash or credit card.

Customer enters the parking card at the exit panel to make a payment.

Once the payment is made, the gate opens.

Customers can stay in the parking lot maximum for 15 hours (from the start of the working hours till the end of the working hours: from 9 am to 00 am).

2. Create ERD using Crow's Foot notation (min.10 well-organized entities; their attributes, and types of relations);

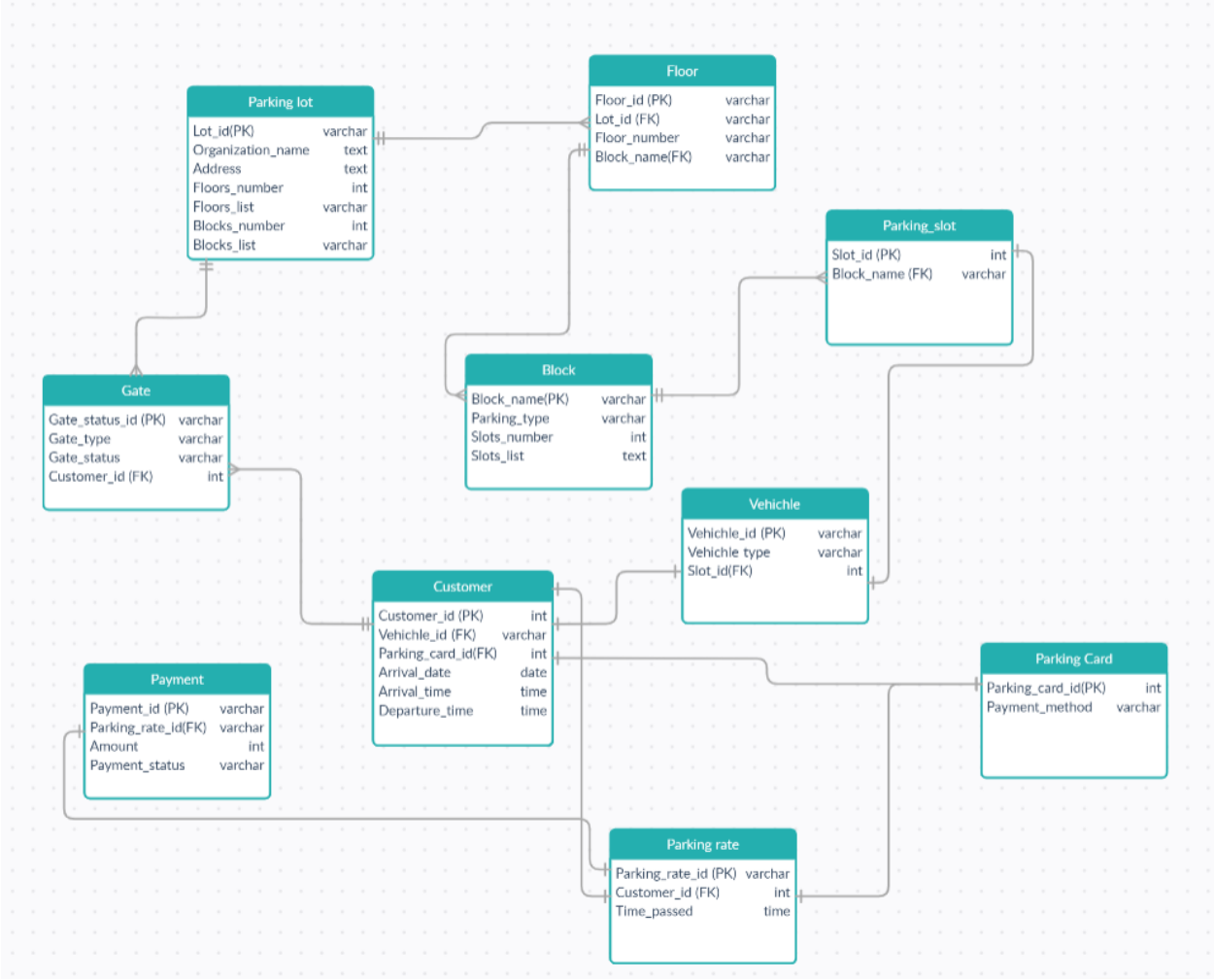

Figure 1 – ER diagram of our database

3. Create database: tables with entities (tables) and constraints (PK, FK, UK, and etc.);

CODE:

```
CREATE DATABASE parking_lot;

CREATE TABLE Parking_lot(
Lot_id varchar,
Organization_name text,
Address text,
Floors_number int,
Floors_list varchar,
Blocks_number int,
Blocks_list varchar,
        PRIMARY KEY(Lot_id)
);

CREATE TABLE Block(
Block_name varchar,
Parking_type varchar,
Slots_number int,
Slots_list text,
        PRIMARY KEY(Block_name)
);

CREATE TABLE Floor(
Floor_id varchar,
Lot_id varchar,
Floor_number varchar,
Block_name varchar,
        FOREIGN KEY(Block_name) REFERENCES Block(Block_name)
);
```

```
CREATE TABLE Parking_slot(
Slot_id int,
Block_name varchar,
        PRIMARY KEY(Slot_id)
);

CREATE TABLE Vehicle(
Vehicle_id varchar,
Vehicle_type varchar,
Slot_id int,
        PRIMARY KEY(Vehicle_id),
        FOREIGN KEY(Slot_id) REFERENCES Parking_slot(Slot_id)
);

CREATE TABLE Parking_card(
Parking_card_id int,
Payment_method varchar,
        PRIMARY KEY(Parking_card_id)
);

CREATE TABLE Customer(
Customer_id int,
Vehicle_id varchar,
Parking_card_id int,
Arrival_date date,
Arrival_time time,
Departure_time time,
        PRIMARY KEY(Customer_id),
        FOREIGN KEY(Vehicle_id) REFERENCES Vehicle(Vehicle_id),
        FOREIGN KEY(Parking_card_id) REFERENCES Parking_card(Parking_card_id)
);
```

```
CREATE TABLE Parking_rate(
Parking_rate_id varchar,
Customer_id int,
Time_passed time,
        PRIMARY KEY(Parking_rate_id),
        FOREIGN KEY(Customer_id) REFERENCES Customer (Customer_id)
);

CREATE TABLE Payment(
Payment_id varchar,
Parking_rate_id varchar,
Amount int,
Payment_status varchar,
        PRIMARY KEY(Payment_id),
        FOREIGN KEY(Parking_rate_id) REFERENCES Parking_rate(Parking_rate_id)
);

CREATE TABLE Gate(
Gate_status_id varchar,
Gate_type varchar,
Gate_status varchar,
Customer_id int,
        PRIMARY KEY(Gate_status_id),
        FOREIGN KEY(Customer_id) REFERENCES Customer(Customer_id)
);
```

4. Write 5 different (add, drop and constraints) ALTER TABLE statements;

1) The administration wants to identify each floor with the unique ID assigned to it.

CODE:

```
ALTER TABLE Floor
ADD PRIMARY KEY(Floor_id);
```

2) The administration wants to make sure that the vehicle ID of each vehicle is inserted.

CODE:

```
ALTER TABLE Vehicle
ALTER Vehicle_id SET NOT NULL;
```

3) The administration wants to collect one more information about the payment – its currency.

CODE:

```
ALTER TABLE Payment
ADD COLUMN Currency varchar;
```

4) The administration wants to connect the name of the block from the parking slot table with the name of the block from the block table.

CODE:

```
ALTER TABLE Parking_Slot
ADD FOREIGN KEY(Block_name) REFERENCES Block(Block_name);
```

5) The administration decided that the currency of the payment is an unnecessary information and decided to delete it.

CODE:

```
ALTER TABLE Payment
DROP COLUMN Currency;
```

5. Write SQL query for DML statements (insert, delete, update). Insert - for all tables at least 10 rows, Update – for each table with a condition, Delete – for each table with a condition;

INSERTING VALUES into tables:

1)Inserting values into the table Parking_lot. Including lot id, shopping center name and address, number of floors and blocks, list of floors and blocks.

We've inserted 10 distinct shopping centers because it was stated in the requirements that we should input at least 10 rows for all tables. However, since those 10 shopping centers are too much, we'll be dealing in the further tables mainly with 3 of them.

CODE:

```
INSERT INTO Parking_lot(lot_id,
organization_name,address,floors_number,floors_list,blocks_number,

blocks_list)

VALUES

('ALA01','Mega center «Alma-Ata»','Rozybakiev street 247A, Almaty',2,'1st floor, 2nd floor',4,'A, B, C, D'),

('ALA02','Mega Park','Makataeva street 127/1, Almaty',2,'1st floor, 2nd floor',4,'E, F, G, H'),

('ALA03','Dostyk Plaza',' Samal-2 111, Almaty',2,'1st floor, 2nd floor',4,'I, J, K, L'),

('ALA04','Colibri','Tashkent highway 17k, Almaty,',2,'1st floor, 2nd floor',2,'Q,R'),

('ALA05','Aport Mall','Ave. Tauelsizdik 34, Nur-Sultan',2,'1st floor, 2nd floor',2,'S,T'),

('AST01','Astana Mall','Ave. Tauelsizdik 34, Nur-Sultan',2,'1st floor, 2nd floor',2,'U,V'),

('AST02','Mega-Astana','4 \ 010000, Korgalzhinskoe highway 1, Nur-Sultan',2,'1st floor, 2nd floor',2,'W,X'),

('AST03','Khan Shatyr','Ave. Turan 37, Nur-Sultan',2,'1st floor, 2nd floor',2,'Y,Z'),

('AST04','Keruen City','Kurgalzhinskoe highway 1, Nur-Sultan',2,'1st floor, 2nd floor',2,'A1,A2'),

('AST05','MEGA Silk Way','Ave. Kabanbay Batyr 62, Nur-Sultan',2,'1st floor, 2nd floor',2,'B1,B2');
```

	lot_id [PK] character varying	organization_name text	address text	floors_number integer	floors_list character varying	blocks_number integer	blocks_list character varying
1	ALA01	Mega center «Alma-Ata»	Rozybakiev street 2…	2	1st floor, 2nd floor	4	A, B, C, D
2	ALA02	Mega Park	Makataeva street 1…	2	1st floor, 2nd floor	4	E, F ,G ,H
3	ALA03	Dostyk Plaza	Samal-2 111, Almaty	2	1st floor, 2nd floor	4	I, J, K, L
4	ALA04	Colibri	Tashkent highway …	2	1st floor, 2nd floor	2	Q,R
5	ALA05	Aport Mall	Ave. Tauelsizdik 34,…	2	1st floor, 2nd floor	2	S,T

	lot_id [PK] character varying	organization_name text	address text	floors_number integer	floors_list character varying	blocks_number integer	blocks_list character varying
5	ALA05	Aport Mall	Ave. Tauelsizdik 34,...	2	1st floor, 2nd floor	2	S,T
6	AST01	Astana Mall	Ave. Tauelsizdik 34,...	2	1st floor, 2nd floor	2	U,V
7	AST02	Mega-Astana	4 \ 010000, Korgalz...	2	1st floor, 2nd floor	2	W,X
8	AST03	Khan Shatyr	Ave. Turan 37, Nur-...	2	1st floor, 2nd floor	2	Y,Z
9	AST04	Keruen City	Kurgalzhinskoe hig...	2	1st floor, 2nd floor	2	A1,A2
10	AST05	MEGA Silk Way	Ave. Kabanbay Bat...	2	1st floor, 2nd floor	2	B1,B2

Figure 3 – List of parking lots we are working with

2)Inserting values into the table Block. Including the name of the block, number of slots, list of slots, and parking type.

CODE:

```
INSERT INTO Block(block_name,parking_type,slots_number,slots_list)
VALUES
('A','four-wheel',25,'101,102,103,104,105,...,125'),
('B','four-wheel',25,'126,127,128,129,130,...,150'),
('C','four-wheel',25,'201,202,203,204,205,...,225'),
('D','two-wheel',50,'226,227,228,229,230,...,275'),
('E','four-wheel',25,'101,102,103,104,105,...,125'),
('F','four-wheel',25,'126,127,128,129,130,...,150'),
('G','four-wheel',25,'201,202,203,204,205,...,225'),
('H','two-wheel',50,'226,227,228,229,230,...,275'),
('I','four-wheel',25,'101,102,103,104,105,...,125'),
('J','four-wheel',25,'126,127,128,129,130,...,150'),
('K','four-wheel',25,'201,202,203,204,205,...,225'),
('L','two-wheel',50,'226,227,228,229,230,...,275'),
('M','two-wheel',25,'275,276,277,278,279,...,300');
```

	block_name [PK] character varying	parking_type character varying	slots_number integer	slots_list text
1	A	four-wheel	25	101,102,103,104,105,...,125
2	B	four-wheel	25	126,127,128,129,130,...,150
3	C	four-wheel	25	201,202,203,204,205,...,225
4	D	two-wheel	50	226,227,228,229,230,...,275
5	E	four-wheel	25	101,102,103,104,105,...,125
6	F	four-wheel	25	126,127,128,129,130,...,150

	block_name [PK] character varying	parking_type character varying	slots_number integer	slots_list text
7	G	four-wheel	25	201,202,203,204,205,...,225
8	H	two-wheel	50	226,227,228,229,230,...,275
9	I	four-wheel	25	101,102,103,104,105,...,125
10	J	four-wheel	25	126,127,128,129,130,...,150
11	K	four-wheel	25	201,202,203,204,205,...,225
12	L	two-wheel	50	226,227,228,229,230,...,275
13	M	two-wheel	25	275,276,277,278,279,...,300

Figure 4 – List of blocks we are working with

3)Inserting values into the table Floor. Including the id of the floor, id of the parking lot, number of the floor and the blocks that are located in this floor.

CODE:

```
INSERT INTO Floor(floor_id,lot_id,floor_number,block_name)
VALUES
('1_A','ALA01','1st floor','A'),
('1_B','ALA01','1st floor','B'),
('1_C','ALA01','2nd floor','C'),
('1_D','ALA01','2nd floor','D'),
('2_E','ALA02','1st floor','E'),
('2_F','ALA02','1st floor','F'),
('2_G','ALA02','2nd floor','G'),
('2_H','ALA02','2nd floor','H'),
('3_I','ALA03','1st floor','I'),
('3_J','ALA03','1st floor','J'),
('3_K','ALA03','2nd floor','K'),
('3_L','ALA03','2nd floor','L'),
('4_M','ALA04','2nd floor','M');
```

	floor_id [PK] character varying	lot_id character varying	floor_number character varying	block_name character varying
1	1_A	ALA01	1st floor	A
2	1_B	ALA01	1st floor	B
3	1_C	ALA01	2nd floor	C
4	1_D	ALA01	2nd floor	D
5	2_E	ALA02	1st floor	E
6	2_F	ALA02	1st floor	F

	floor_id [PK] character varying	lot_id character varying	floor_number character varying	block_name character varying
7	2_G	ALA02	2nd floor	G
8	2_H	ALA02	2nd floor	H
9	3_I	ALA03	1st floor	I
10	3_J	ALA03	1st floor	J
11	3_K	ALA03	2nd floor	K
12	3_L	ALA03	2nd floor	L
13	4_M	ALA04	2nd floor	M

Figure 5 – List of floors that belong to each floor

4)Inserting values into the table Parking_slot. Including the id of the slot and the name of the block to which it belongs.

CODE:

```
INSERT INTO Parking_slot(slot_id,block_name)
VALUES
(101,'A'),
(131,'B'),
(221,'C'),
(231,'D'),
(102,'E'),
(132,'F'),
(222,'G'),
(232,'H'),
(103,'I'),
(133,'J'),
(223,'K'),
(233,'L'),
(288,'M');
```

	slot_id [PK] integer	block_name character varying
1	101	A
2	102	E
3	103	I
4	131	B
5	132	F
6	133	J

	slot_id [PK] integer	block_name character varying
7	221	C
8	222	G
9	223	K
10	231	D
11	232	H
12	233	L
13	288	M

Figure 6 – List of parking slots that belong to each block

5) Inserting values into the table Vehicle. Including the id of the vehicle, type of the vehicle (two-wheel, four-wheel) and the id of the slot where it is parked.

CODE:

```
INSERT INTO Vehicle(vehicle_id,vehicle_type,slot_id)
VALUES
('111ONE', 'four-wheel', 101),
('222TWO', 'four-wheel', 131),
('333TRI', 'four-wheel', 221),
('123ABC', 'two-wheel', 231),
('444FOR', 'four-wheel', 102),
('555FIV', 'four-wheel', 132),
('666SIX', 'four-wheel', 222),
('456DEF', 'two-wheel', 232),
('777SEV', 'four-wheel', 103),
('888EIT', 'four-wheel', 133),
('999NIN', 'four-wheel', 223),
('789HIJ', 'two-wheel', 233),
('941HEX', 'two-wheel', 288);
```

	vehicle_id [PK] character varying	vehicle_type character varying	slot_id integer
1	111ONE	four-wheel	101
2	123ABC	two-wheel	231
3	222TWO	four-wheel	131
4	333TRI	four-wheel	221
5	444FOR	four-wheel	102
6	456DEF	two-wheel	232

	vehicle_id [PK] character varying	vehicle_type character varying	slot_id integer
7	555FIV	four-wheel	132
8	666SIX	four-wheel	222
9	777SEV	four-wheel	103
10	789HIJ	two-wheel	233
11	888EIT	four-wheel	133
12	941HEX	two-wheel	288
13	999NIN	four-wheel	223

Figure 7 – List of vehicles we are working with

6)Inserting values into the table Parking_card. Including the id of the parking card and the payment method the customer owning this card prefers.

CODE:

```
INSERT INTO Parking_card(parking_card_id,payment_method)
VALUES
(8801, 'cash'),
(8802, 'credit card'),
(8803, 'cash'),
(8804, 'credit card'),
(8805, 'cash'),
(8806, 'credit card'),
(8807, 'cash'),
(8808, 'credit card'),
(8809, 'cash'),
(8810, 'credit card'),
(8811, 'credit card'),
(8812, 'credit card'),
(8813, 'cash'),
(8814, 'credit card'),
(8815, 'cash');
```

	parking_card_id [PK] integer	payment_method character varying
1	8801	cash
2	8802	credit card
3	8803	cash
4	8804	credit card
5	8805	cash
6	8806	credit card

	parking_card_id [PK] integer	payment_method character varying
7	8807	cash
8	8808	credit card
9	8809	cash
10	8810	credit card
11	8811	credit card
12	8812	credit card
13	8813	cash
14	8814	credit card
15	8815	cash

Figure 8 – List of parking cards we are working with

7)Inserting values into the table Customer. Including the id of the customer, id of the parking card of the customer, date and time when this customer has arrived into the parking lot, and the departure time.

Considering that today is the 17th of November, there are some customers who haven't left the parking lot yet (as they're still in the shopping center). For those customers the departure time is undefined.

CODE:

```
INSERT INTO
Customer(customer_id,vehicle_id,parking_card_id,arrival_date,arrival_time,departure_time)

VALUES

(801, '111ONE', 8801,'2020-11-15', '10:00', '11:00'),

(802, '222TWO', 8802, '2020-11-10', '11:15', '11:29'),

(803, '333TRI', 8803, '2020-10-28', '13:20', '15:20'),

(805, '444FOR', 8805, '2020-10-15', '16:00', '20:10'),

(806, '555FIV', 8806, '2020-10-22', '20:15', '20:30'),

(808, '456DEF', 8808, '2020-11-10', '20:00', '21:00'),

(809, '777SEV', 8809, '2020-11-08', '11:00',' 17:00'),

(810, '888EIT', 8810, '2020-10-07', '13:46', '14:36'),

(811, '999NIN', 8811, '2020-11-17', '11:00', '15:05');


INSERT INTO Customer(customer_id,vehicle_id,parking_card_id,arrival_date,arrival_time)

VALUES

(804, '123ABC', 8804, '2020-11-17', '14:12'),

(807, '666SIX', 8807, '2020-11-17', '16:00'),
```

(812, '789HIJ', 8812, '2020-11-17', '15:12'),

(813, '941HEX', 8813, '2020-11-17', '13:35');

	customer_id [PK] integer	vehicle_id character varying	parking_card_id integer	arrival_date date	arrival_time time without time zone	departure_time time without time zone
1	801	111ONE	8801	2020-11-15	10:00:00	11:00:00
2	802	222TWO	8802	2020-11-10	11:15:00	11:29:00
3	803	333TRI	8803	2020-10-28	13:20:00	15:20:00
4	804	123ABC	8804	2020-11-17	14:12:00	[null]
5	805	444FOR	8805	2020-10-15	16:00:00	20:10:00
6	806	555FIV	8806	2020-10-22	20:15:00	20:30:00

	customer_id [PK] integer	vehicle_id character varying	parking_card_id integer	arrival_date date	arrival_time time without time zone	departure_time time without time zone
7	807	666SIX	8807	2020-11-17	16:00:00	[null]
8	808	456DEF	8808	2020-11-10	20:00:00	21:00:00
9	809	777SEV	8809	2020-11-08	11:00:00	17:00:00
10	810	888EIT	8810	2020-10-07	13:46:00	14:36:00
11	811	999NIN	8811	2020-11-17	11:00:00	15:05:00
12	812	789HIJ	8812	2020-11-17	15:12:00	[null]
13	813	941HEX	8813	2020-11-17	13:35:00	[null]

Figure 9 – List of customers we are working with

8)Inserting values into the table Parking_rate. Including the id of the customer, parking rate id, and the time that particular customer has spent in the parking lot.

Considering that today is the 17th of November, there are some customers who haven't left the parking lot yet (as they're still in the shopping center). For those customers the time passed is undefined.

CODE:

INSERT INTO Parking_rate(parking_rate_id,customer_id,time_passed)

VALUES

('R801RE', 801, '01:00'),

('R802FR', 802,'00:14'),

('R803ED', 803, '03:00'),

('R805GH', 805, '04:10'),

('R806DL', 806, '00:15'),

('R808LD', 808, '01:00'),

('R809BV', 809, '06:00'),

('R810LW', 810, '00:50'),

('R811SA', 811, '01:05');

INSERT INTO Parking_rate(parking_rate_id,customer_id)

VALUES

('R804JK', 804),

('R807BG', 807),

('R812KE', 812),

('R813ME', 813);

	parking_rate_id [PK] character varying	customer_id integer	time_passed time without time zone
1	R801RE	801	01:00:00
2	R802FR	802	00:14:00
3	R803ED	803	03:00:00
4	R804JK	804	[null]
5	R805GH	805	04:10:00
6	R806DL	806	00:15:00

	parking_rate_id [PK] character varying	customer_id integer	time_passed time without time zone
7	R807BG	807	[null]
8	R808LD	808	01:00:00
9	R809BV	809	06:00:00
10	R810LW	810	00:50:00
11	R811SA	811	01:05:00
12	R812KE	812	[null]
13	R813ME	813	[null]

Figure 10 – List of customers and their parking rates

9)Inserting values into the table Payment. Including the payment id, the parking rate id, amount paid (in tenges), and the status of the payment.

Considering that today is the 17th of November, there are some customers who haven't left the parking lot yet (as they're still in the shopping center). For those customers the payment amount is undefined and payment status is unpaid.

CODE:

INSERT INTO Payment(payment_id,parking_rate_id,amount,payment_status)

VALUES

('P801RE', 'R801RE', 100, 'paid'),

('P802FR', 'R802FR', 0, 'paid'),

('P803ED', 'R803ED', 300, 'paid'),

('P805GH', 'R805GH', 400, 'paid'),

('P806DL', 'R806DL', 0, 'paid'),

('P808LD', 'R808LD', 100, 'paid'),

('P809BV', 'R809BV', 600, 'paid'),

('P810LW', 'R810LW', 100, 'paid'),

('P811SA', 'R811SA', 200, 'paid');

INSERT INTO Payment(payment_id,parking_rate_id,payment_status)

VALUES

('P804JK', 'R804JK', 'unpaid'),

('P807BG', 'R807BG', 'unpaid'),

('P812KE', 'R812KE', 'unpaid'),

('P813ME', 'R813ME', 'unpaid');

	payment_id [PK] character varying	parking_rate_id character varying	amount integer	payment_status character varying
1	P801RE	R801RE	100	paid
2	P802FR	R802FR	0	paid
3	P803ED	R803ED	300	paid
4	P804JK	R804JK	[null]	unpaid
5	P805GH	R805GH	400	paid
6	P806DL	R806DL	0	paid

	payment_id [PK] character varying	parking_rate_id character varying	amount integer	payment_status character varying
7	P807BG	R807BG	[null]	unpaid
8	P808LD	R808LD	100	paid
9	P809BV	R809BV	600	paid
10	P810LW	R810LW	100	paid
11	P811SA	R811SA	200	paid
12	P812KE	R812KE	[null]	unpaid
13	P813ME	R813ME	[null]	unpaid

Figure 11 – List of payment information we are working with

10)Inserting values into the table Gate. Including the gate status id, type of the gate, status of the gate and the customer id to whom it refers to.

Considering that today is the 17th of November, there are some customers who haven't left the parking lot yet (as they're still in the shopping center). For those customers the gate status is closed.

CODE:

INSERT INTO Gate(gate_status_id,gate_type,gate_status,customer_id)

VALUES

('ALA01_E1', 'Exit gate', 'opened', 801),

('ALA01_E2', 'Exit gate', 'opened', 802),

('ALA01_E3', 'Exit gate', 'opened', 803),

('ALA01_E4', 'Exit gate', 'closed', 804),

('ALA02_E5', 'Exit gate', 'opened', 805),

('ALA02_E6', 'Exit gate', 'opened', 806),

('ALA02_E7', 'Exit gate', 'closed', 807),

('ALA02_E8', 'Exit gate', 'opened', 808),

('ALA03_E9', 'Exit gate', 'opened', 809),

('ALA03_E10', 'Exit gate', 'opened', 810),

('ALA03_E11', 'Exit gate', 'closed', 811),

('ALA03_E12', 'Exit gate', 'closed', 812),

('ALA03_E13','Exit gate', 'closed', 813);

	gate_status_id [PK] character varying	gate_type character varying	gate_status character varying	customer_id integer
1	ALA01_E1	Exit gate	opened	801
2	ALA01_E2	Exit gate	opened	802
3	ALA01_E3	Exit gate	opened	803
4	ALA01_E4	Exit gate	closed	804
5	ALA02_E5	Exit gate	opened	805
6	ALA02_E6	Exit gate	opened	806

	gate_status_id [PK] character varying	gate_type character varying	gate_status character varying	customer_id integer
7	ALA02_E7	Exit gate	closed	807
8	ALA02_E8	Exit gate	opened	808
9	ALA03_E9	Exit gate	opened	809
10	ALA03_E10	Exit gate	opened	810
11	ALA03_E11	Exit gate	closed	811
12	ALA03_E12	Exit gate	closed	812
13	ALA03_E13	Exit gate	closed	813

Figure 12 – List of gates we are working with

UPDATING and DELETING from tables

Table 10:

The administration has decided to change the gate status of the payment with the gate status id ‘ALA03_E11’ to ‘opened’.

CODE:

UPDATE Gate

SET gate_status = 'opened'

WHERE gate_status_id = 'ALA03_E11';

The administration has decided to delete the gate status with id ‘ALA03_E13’ from the table Gate.

CODE:

DELETE FROM Gate

```
WHERE gate_status_id = 'ALA03_E13';
```

Table 9:

The administration has decided to change the payment status of the payment with id ‘P813ME’ to ‘paid’.

CODE:

```
UPDATE Payment
SET payment_status = 'paid'
WHERE payment_id = 'P813ME';
```

The administration has decided to delete the payment with id ‘P813ME’ from the table Payment.

CODE:

```
DELETE FROM Payment
WHERE payment_id = 'P813ME';
```

Table 8:

The administration has decided to change the amount of time spent in the parking lot (time-passed) of the customer with id 811 to ’02:05’.

CODE:

```
UPDATE Parking_rate
SET time_passed = '02:05'
WHERE customer_id = 811;
```

The administration has decided to delete the customer with id 813 from the table Parking_rate.

CODE:

```
DELETE FROM Parking_rate
WHERE customer_id = 813;
```

Table 7:

The administration has decided to change the arrival time of the customer with id 813 to’13:55’.

CODE:

```
UPDATE Customer
SET arrival_time = '13:55'
```

```
WHERE customer_id = 813;
```

The administration has decided to delete the customer with id 813 from the table Customer.

CODE:

```
DELETE FROM Customer
WHERE customer_id = 813;
```

Table 6:

The administration has decided to change the payment method of the parking card with id 8813 to ‘credit card’.

CODE:

```
UPDATE Parking_card
SET payment_method = 'credit card'
WHERE parking_card_id = 8813;
```

The administration has decided to delete the parking card with id 8813 from the table Parking_card.

CODE:

```
DELETE FROM Parking_card
WHERE parking_card_id = 8813;
```

Table 5:

The administration has decided to change the vehicle type of the slot with the id 288 to ‘four-wheel’.

CODE:

```
UPDATE Vehicle
SET vehicle_type = 'four-wheel'
WHERE slot_id = 288;
```

The administration has decided to delete the slot with id 288 from the table Vehicle.

CODE:

```
DELETE FROM Vehicle
WHERE slot_id = 288;
```

Table 4:

The administration has decided to change the slot id of the block named M to 299.

CODE:

```
UPDATE Parking_slot
SET slot_id = 299
WHERE block_name = 'M';
```

The administration has decided to delete the block named M from the table Parking_slot.

CODE:

```
DELETE FROM Parking_slot
WHERE block_name = 'M';
```

Table 3:

The administration has decided to change the floor_number of the floor with the id = '4_M' to the '1st floor'.

CODE:

```
UPDATE Floor
SET floor_number = '1st floor'
WHERE floor_id = '4_M';
```

The administration has decided to delete the floor with the id = '4_M' from the table Floor.

CODE:

```
DELETE FROM Floor
WHERE floor_id = '4_M';
```

Table 2:

The administration has decided to change the parking type of the block named M to 'four-wheel'.

CODE:

```
UPDATE Block
SET parking_type = 'four-wheel'
WHERE block_name = 'M';
```

The administration has decided to delete the block named M from the table Block.

CODE:

```
DELETE FROM Block
WHERE block_name = 'M';
```

Table 1:

The administration has decided to change the name of the shopping center ‘MEGA Silk Way’ with the id ‘AST05’ to ‘Mega Silk Way’.

CODE:

```
UPDATE parking_lot
SET organization_name = 'Mega Silk Way'
WHERE lot_id = 'AST05';
```

The administration has decided to delete the shopping center Colibri from the table of parking_lot.

CODE:

```
DELETE FROM Parking_lot
WHERE lot_id = 'ALA04';
```

6. Write at least 10 queries: using DISTINCT, conditions (,=), OR, AND, BETWEEN, IN, LIKE, LENGHT, COUNT, MAX, MIN, SUM, AVG, INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN and etc. The queries should be coherent and complex.

1) The administration wants to select shopping centers which start with 'Mega', their addresses and, and their parking lot's block names with the corresponding floor numbers.Left join returns all rows from the table parking_lot specified in the ON condition and only those rows from the other table floor where the joined fields are equal.

CODE:

```
SELECT
parking_lot.organization_name,parking_lot.address,floor.floor_number,floor.block_name

FROM parking_lot

LEFT JOIN floor

ON parking_lot.lot_id=floor.lot_id

WHERE organization_name LIKE 'Mega%';
```

	organization_name text	address text	floor_number character varying	block_name character varying
1	Mega center «Alma-Ata»	Rozybakiev street 247A, Almaty	1st floor	A
2	Mega center «Alma-Ata»	Rozybakiev street 247A, Almaty	1st floor	B
3	Mega center «Alma-Ata»	Rozybakiev street 247A, Almaty	2nd floor	C
4	Mega center «Alma-Ata»	Rozybakiev street 247A, Almaty	2nd floor	D
5	Mega Park	Makataeva street 127/1, Almaty	1st floor	E
6	Mega Park	Makataeva street 127/1, Almaty	1st floor	F
7	Mega Park	Makataeva street 127/1, Almaty	2nd floor	G
8	Mega Park	Makataeva street 127/1, Almaty	2nd floor	H
9	Mega-Astana	4 \ 010000, Korgalzhinskoe hig...	[null]	[null]
10	Mega Silk Way	Ave. Kabanbay Batyr 62, Nur-Su...	[null]	[null]

Figure 13 – List of shopping centers

2) The administration wants to select some information about customers (their parking card id, vehicle id, arrival date and payment method) who prefer to pay with a cash. Right join returns all rows from the table parking_card specified in the ON condition and only those rows from the other table customer where the joined fields are equal.

CODE:

```
SELECT
parking_card.parking_card_id,customer.vehicle_id,customer.arrival_date,parking_card.payment
_method

FROM customer

RIGHT JOIN parking_card

ON customer.parking_card_id=parking_card.parking_card_id

WHERE payment_method IN ('cash');
```

	parking_card_id integer	vehicle_id character varying	arrival_date date	payment_method character varying
1	8801	111ONE	2020-11-15	cash
2	8803	333TRI	2020-10-28	cash
3	8805	444FOR	2020-10-15	cash
4	8809	777SEV	2020-11-08	cash
5	8807	666SIX	2020-11-17	cash
6	8815	[null]	[null]	cash

Figure 14 – List of customers

3) The administration wants to count the number of gates with the status 'opened' and the number of gates with the status 'closed'

CODE:

```
SELECT COUNT(gate_status) AS num_of_gates,gate_status

FROM gate

GROUP BY gate_status;
```

	num_of_gates bigint	gate_status character varying
1	9	opened
2	3	closed

Figure 15 – List of gates

4) The administration wants to select some information about customers (their customer id, payment method, arrival date, arrival time, and departure time) who arrived at the parking_lot after the 1st of November, 2020 before the 4 P.M. Full join returns all rows from the LEFT-hand table and RIGHT-hand table with nulls in place where the join condition is not met.

CODE:

```
SELECT
customer.customer_id,parking_card.payment_method,customer.arrival_date,customer.arrival_ti
me,customer.departure_time

FROM customer

FULL JOIN parking_card

ON customer.parking_card_id=parking_card.parking_card_id

WHERE arrival_date > '2020-11-01' AND arrival_time < '16:00';
```

	customer_id integer	payment_method character varying	arrival_date date	arrival_time time without time zone	departure_time time without time zone
1	801	cash	2020-11-15	10:00:00	11:00:00
2	802	credit card	2020-11-10	11:15:00	11:29:00
3	809	cash	2020-11-08	11:00:00	17:00:00
4	811	credit card	2020-11-17	11:00:00	15:05:00
5	804	credit card	2020-11-17	14:12:00	[null]
6	812	credit card	2020-11-17	15:12:00	[null]

Figure 16 – List of customers

5) The administration wants to select the maximum, minimum, average amount payed by the customers for the parking as well as the total sum paid by all listed customers.

CODE:

```
SELECT MAX(amount) AS max_amount,MIN(amount) AS
min_amount,AVG(amount)::numeric(10,2)

AS avg_amount,SUM(amount) AS sum

FROM payment;
```

	max_amount integer	min_amount integer	avg_amount numeric (10,2)	sum bigint
1	600	0	200.00	1800

Figure 17 – List of calculations

6) The administration wants to select some information about shopping centers (name, address, lot_id, and floor_number) where the floor_number is not equal to '2nd floor' and the length of the address lies between 25 and 35 range.Inner join returns records that have matching values in both tables.

CODE:

```
SELECT
parking_lot.organization_name,parking_lot.address,floor.lot_id,floor.floor_id,floor.floor_numbe
r

FROM parking_lot

INNER JOIN floor

ON parking_lot.lot_id=floor.lot_id

WHERE (Length(address) BETWEEN 25 AND 35) AND floor_number NOT IN ('1st floor');
```

	organization_name text	address text	lot_id character varying	floor_id character varying	floor_number character varying
1	Mega center «Alma-Ata»	Rozybakiev street 247A, Almaty	ALA01	1_C	2nd floor
2	Mega center «Alma-Ata»	Rozybakiev street 247A, Almaty	ALA01	1_D	2nd floor
3	Mega Park	Makataeva street 127/1, Almaty	ALA02	2_G	2nd floor
4	Mega Park	Makataeva street 127/1, Almaty	ALA02	2_H	2nd floor

If we remove the length limitation we will get such table:

	organization_name text	address text	lot_id character varying	floor_id character varying	floor_number character varying
1	Mega center «Alma-Ata»	Rozybakiev street 247A, Almaty	ALA01	1_C	2nd floor
2	Mega center «Alma-Ata»	Rozybakiev street 247A, Almaty	ALA01	1_D	2nd floor
3	Mega Park	Makataeva street 127/1, Almaty	ALA02	2_G	2nd floor
4	Mega Park	Makataeva street 127/1, Almaty	ALA02	2_H	2nd floor
5	Dostyk Plaza	Samal-2 111, Almaty	ALA03	3_K	2nd floor
6	Dostyk Plaza	Samal-2 111, Almaty	ALA03	3_L	2nd floor

Figure 18 – List of shopping centers

7) The administration wants to select some information about the payment (payment_status, amount) that certain vehicle has made (vehicle_id) and the time that vehicle has spent in the parking_lot (time_passed). Selecting only the ones with the payment amount larger than average or the one with the vehicle id = '999NIN'. Inner join of the payment and parking_rate tables returns records that have matching values in both tables. Same for the inner join case between parking_rate and customer tables.

CODE:

```
SELECT
payment.amount,payment.payment_status,parking_rate.time_passed,customer.vehicle_id

FROM payment

INNER JOIN parking_rate ON payment.parking_rate_id=parking_rate.parking_rate_id

INNER JOIN customer ON parking_rate.customer_id=customer.customer_id

WHERE amount >(SELECT AVG(amount) FROM payment) OR vehicle_id = '999NIN';
```

	amount integer	payment_status character varying	time_passed time without time zone	vehicle_id character varying
1	300	paid	03:00:00	333TRI
2	400	paid	04:10:00	444FOR
3	600	paid	06:00:00	777SEV
4	200	paid	02:05:00	999NIN

Figure 19 – List of payment information

8) The administration wants to select some information about customers (customer id, departure time) and the payment status, gate status, and time spent in the parking lot for that particular customer.Order by the customer_id in the ascending order. Inner join of the payment and parking_rate tables returns records that have matching values in both tables. Same for the inner join case between parking_rate and customer tables, gate and customer tables.

CODE:

SELECT
time_passed,customer.customer_id,customer.departure_time,payment.payment_status,gate.gate_status

FROM parking_rate

INNER JOIN payment ON parking_rate.parking_rate_id=payment.parking_rate_id

INNER JOIN customer ON parking_rate.customer_id=customer.customer_id

INNER JOIN gate ON customer.customer_id=gate.customer_id

ORDER BY customer_id;

	time_passed time without time zone	customer_id integer	departure_time time without time zone	payment_status character varying	gate_status character varying
1	01:00:00	801	11:00:00	paid	opened
2	00:14:00	802	11:29:00	paid	opened
3	03:00:00	803	15:20:00	paid	opened
4	[null]	804	[null]	unpaid	closed
5	04:10:00	805	20:10:00	paid	opened
6	00:15:00	806	20:30:00	paid	opened
7	[null]	807	[null]	unpaid	closed
8	01:00:00	808	21:00:00	paid	opened
9	06:00:00	809	17:00:00	paid	opened
10	00:50:00	810	14:36:00	paid	opened
11	02:05:00	811	15:05:00	paid	opened
12	[null]	812	[null]	unpaid	closed

Figure 20 – List of customers

9) The administration wants to select distinct time_passed values (amount of time spent in the parking lot) where the values are not null and value of the hour is larger than 1.

CODE:

SELECT DISTINCT time_passed

FROM parking_rate

WHERE time_passed IS NOT NULL AND EXTRACT(hour from time_passed)>1;

	time_passed time without time zone
1	04:10:00
2	02:05:00
3	06:00:00
4	03:00:00

Figure 21 – List of time values

10) The administration wants to select some information about customers (customer_id, vehicle_id, parking_card_id, arrival_time, departure_time) and finding the sum of the hours of the arrival and departure times. The gate_status has to be repeated more than 4 times in the table gate

and the customer_id should not be equal to 810.Inner join of the customer and gate tables returns records that have matching values in both tables.

CODE:

```
SELECT
customer.customer_id,customer.vehicle_id,customer.parking_card_id,customer.arrival_time,

customer.departure_time,EXTRACT(HOUR from arrival_time)+EXTRACT(HOUR from departure_time) AS sum_of_hours_of_arrival_departure_times

FROM customer

INNER JOIN gate ON customer.customer_id=gate.customer_id

WHERE gate_status IN (SELECT gate_status FROM gate GROUP BY gate_status HAVING COUNT(*)>4) AND customer.customer_id != 810;
```

	customer_id [PK] integer	vehicle_id character varying	parking_card_id integer	arrival_time time without time zone	departure_time time without time zone	sum_of_hours_of_arrival_departure_times double precision
1	801	111ONE	8801	10:00:00	11:00:00	21
2	802	222TWO	8802	11:15:00	11:29:00	22
3	803	333TRI	8803	13:20:00	15:20:00	28
4	805	444FOR	8805	16:00:00	20:10:00	36
5	806	555FIV	8806	20:15:00	20:30:00	40
6	808	456DEF	8808	20:00:00	21:00:00	41
7	809	777SEV	8809	11:00:00	17:00:00	28
8	811	999NIN	8811	11:00:00	15:05:00	26

Figure 22 – List of customers

11) The administration wants to select some information about the customer (customer_id,arrival_time) and his/her vehicle(vehicle_id), the block and slot where that customer parked his/her vehicle, and the gate status for that particular customer. Selecting the ones who parked in October and has a customer_id which is divisible by 2.Left join returns all rows from the table block specified in the ON condition and only those rows from the other table parking_slot where the joined fields are equal. Same for the left join case between parking_slot and vehicle tables, vehicle and customer tables, customer and gate tables.

CODE:

```
SELECT
parking_slot.block_name,parking_slot.slot_id,vehicle.vehicle_id,customer.customer_id,block.parking_type,customer.arrival_time,gate.gate_status

FROM block

LEFT JOIN parking_slot ON block.block_name=parking_slot.block_name

LEFT JOIN vehicle ON parking_slot.slot_id= vehicle.slot_id

LEFT JOIN customer ON vehicle.vehicle_id=customer.vehicle_id

LEFT JOIN gate ON customer.customer_id=gate.customer_id

WHERE EXTRACT(month from arrival_date)=10 AND customer.customer_id % 2 =0;
```

	block_name character varying	slot_id integer	vehicle_id character varying	customer_id integer	parking_type character varying	arrival_time time without time zone	gate_status character varying
1	F	132	555FIV	806	four-wheel	20:15:00	opened
2	J	133	888EIT	810	four-wheel	13:46:00	opened

Figure 23 – List of customer information

7. Write at least 5 subqueries: single-row, multiple-row and multiple-column subqueries, and etc.;

1) The administration wants to select all information about customer from the customer table where parking_card_id is larger than 8805 (Single-row subquery).

CODE:

```
SELECT *

FROM customer

WHERE parking_card_id > (SELECT parking_card_id FROM parking_card WHERE
parking_card_id=8805);
```

	customer_id [PK] integer	vehicle_id character varying	parking_card_id integer	arrival_date date	arrival_time time without time zone	departure_time time without time zone
1	806	555FIV	8806	2020-10-22	20:15:00	20:30:00
2	808	456DEF	8808	2020-11-10	20:00:00	21:00:00
3	809	777SEV	8809	2020-11-08	11:00:00	17:00:00
4	810	888EIT	8810	2020-10-07	13:46:00	14:36:00
5	811	999NIN	8811	2020-11-17	11:00:00	15:05:00
6	807	666SIX	8807	2020-11-17	16:00:00	[null]
7	812	789HIJ	8812	2020-11-17	15:12:00	[null]

Figure 24 – List of customers

2) The administration wants to select all rows from the table block where the parking type is equal to ‘four-wheel’(Multiple-row subquery).

CODE:

```
SELECT *

FROM block

WHERE block_name=ANY(SELECT block_name FROM block WHERE parking_type='four-
wheel');
```

	block_name [PK] character varying	parking_type character varying	slots_number integer	slots_list text
1	A	four-wheel	25	101,102,103,104,105,...,125
2	B	four-wheel	25	126,127,128,129,130,...,150
3	C	four-wheel	25	201,202,203,204,205,...,225
4	E	four-wheel	25	101,102,103,104,105,...,125
5	F	four-wheel	25	126,127,128,129,130,...,150
6	G	four-wheel	25	201,202,203,204,205,...,225
7	I	four-wheel	25	101,102,103,104,105,...,125
8	J	four-wheel	25	126,127,128,129,130,...,150
9	K	four-wheel	25	201,202,203,204,205,...,225

Figure 25 – List of blocks

3) The administration wants to select customer id, vehicle id, arrival date, arrival time, departure time from the table customer where arrival_date is on the 17th of November and customer_id is an odd number (Multiple_column subquery).

CODE:

```
SELECT customer_id,vehicle_id,arrival_date,arrival_time,departure_time

FROM customer

WHERE (customer_id,arrival_date) IN (SELECT customer_id,arrival_date FROM customer
WHERE arrival_date='2020-11-17' AND customer_id%2=1);
```

	customer_id [PK] integer	vehicle_id character varying	arrival_date date	arrival_time time without time zone	departure_time time without time zone
1	811	999NIN	2020-11-17	11:00:00	15:05:00
2	807	666SIX	2020-11-17	16:00:00	[null]

Figure 26 – List of customers

4) The administration wants to select some information about shopping centers (their id, name, address and, by left joining table parking_lot with the table floor, also their parking lot's block name with the corresponding floor number) where lot_id starts with 'ALA'(Multiple-row subquery).

CODE:

```
SELECT
parking_lot.organization_name,parking_lot.address,floor.lot_id,floor.floor_number,floor.block_
name

FROM parking_lot

LEFT JOIN floor ON parking_lot.lot_id=floor.lot_id
```

WHERE organization_name IN(SELECT organization_name FROM parking_lot WHERE

lot_id LIKE 'ALA%');

	organization_name text	address text	lot_id character varying	floor_number character varying	block_name character varying
1	Mega center «Alma-Ata»	Rozybakiev ...	ALA01	1st floor	A
2	Mega center «Alma-Ata»	Rozybakiev ...	ALA01	1st floor	B
3	Mega center «Alma-Ata»	Rozybakiev ...	ALA01	2nd floor	C
4	Mega center «Alma-Ata»	Rozybakiev ...	ALA01	2nd floor	D
5	Mega Park	Makataeva ...	ALA02	1st floor	E
6	Mega Park	Makataeva ...	ALA02	1st floor	F
7	Mega Park	Makataeva ...	ALA02	2nd floor	G
8	Mega Park	Makataeva ...	ALA02	2nd floor	H
9	Dostyk Plaza	Samal-2 11...	ALA03	1st floor	I
10	Dostyk Plaza	Samal-2 11...	ALA03	1st floor	J
11	Dostyk Plaza	Samal-2 11...	ALA03	2nd floor	K
12	Dostyk Plaza	Samal-2 11...	ALA03	2nd floor	L
13	Aport Mall	Ave. Tauelsi...	[null]	[null]	[null]

Figure 27 – List of shopping centers

5) The administration wants to select some information about customers (customer_id, arrival_date,arrival_time,departure_time) who arrived at the parking lot after the 24th of October, 2020 and left the parking lot after 2 P.M. (Multiple-column subquery).

CODE:

SELECT customer_id,arrival_date,arrival_time,departure_time

FROM customer

WHERE (arrival_date,departure_time) IN (SELECT arrival_date,departure_time FROM customer WHERE arrival_date >'2020-10-24' AND departure_time >'14:00');

	customer_id [PK] integer	arrival_date date	arrival_time time without time zone	departure_time time without time zone
1	803	2020-10-28	13:20:00	15:20:00
2	808	2020-11-10	20:00:00	21:00:00
3	809	2020-11-08	11:00:00	17:00:00
4	811	2020-11-17	11:00:00	15:05:00

Figure 28 – List of customers

6) The administration wants to select some information about customers (vehicle id, arrival time) and the payment status, amount paid, and time spent in the parking lot for that particular customer. Selecting specifically those who have already paid for their parking. Order by the amount in the descending order (Multiple-row subquery).

CODE:

```
SELECT
customer.vehicle_id,customer.arrival_time,parking_rate.time_passed,payment.payment_status,payment.amount

FROM payment

INNER JOIN parking_rate ON payment.parking_rate_id=parking_rate.parking_rate_id

INNER JOIN customer ON parking_rate.customer_id=customer.customer_id

WHERE payment_status IN (SELECT payment_status FROM payment WHERE payment_status= 'paid')

ORDER BY amount DESC;
```

	vehicle_id character varying	arrival_time time without time zone	time_passed time without time zone	payment_status character varying	amount integer
1	777SEV	11:00:00	06:00:00	paid	600
2	444FOR	16:00:00	04:10:00	paid	400
3	333TRI	13:20:00	03:00:00	paid	300
4	999NIN	11:00:00	02:05:00	paid	200
5	111ONE	10:00:00	01:00:00	paid	100
6	456DEF	20:00:00	01:00:00	paid	100
7	888EIT	13:46:00	00:50:00	paid	100
8	222TWO	11:15:00	00:14:00	paid	0
9	555FIV	20:15:00	00:15:00	paid	0

Figure 29 – List of vehicles

APPENDIX A

Create table statements:

```
CREATE TABLE Parking_lot(
Lot_id varchar,
Organization_name text,
Address text,
Floors_number int,
Floors_list varchar,
Blocks_number int,
Blocks_list varchar,
        PRIMARY KEY(Lot_id)
);

CREATE TABLE Block(
Block_name varchar,
Parking_type varchar,
Slots_number int,
Slots_list text,
        PRIMARY KEY(Block_name)
);

CREATE TABLE Floor(
Floor_id varchar,
Lot_id varchar,
Floor_number varchar,
Block_name varchar,
        FOREIGN KEY(Block_name) REFERENCES Block(Block_name)
);

CREATE TABLE Parking_slot(
Slot_id int,
Block_name varchar,
```

```
        PRIMARY KEY(Slot_id)
);

CREATE TABLE Vehicle(
Vehicle_id varchar,
Vehicle_type varchar,
Slot_id int,
        PRIMARY KEY(Vehicle_id),
        FOREIGN KEY(Slot_id) REFERENCES Parking_slot(Slot_id)
);

CREATE TABLE Parking_card(
Parking_card_id int,
Payment_method varchar,
        PRIMARY KEY(Parking_card_id)
);

CREATE TABLE Customer(
Customer_id int,
Vehicle_id varchar,
Parking_card_id int,
Arrival_date date,
Arrival_time time,
Departure_time time,
        PRIMARY KEY(Customer_id),
        FOREIGN KEY(Vehicle_id) REFERENCES Vehicle(Vehicle_id),
        FOREIGN KEY(Parking_card_id) REFERENCES Parking_card(Parking_card_id)
);

CREATE TABLE Parking_rate(
Parking_rate_id varchar,
```

```
Customer_id int,
Time_passed time,
        PRIMARY KEY(Parking_rate_id),
        FOREIGN KEY(Customer_id) REFERENCES Customer (Customer_id)
);

CREATE TABLE Payment(
Payment_id varchar,
Parking_rate_id varchar,
Amount int,
Payment_status varchar,
        PRIMARY KEY(Payment_id),
        FOREIGN KEY(Parking_rate_id) REFERENCES Parking_rate(Parking_rate_id)
);

CREATE TABLE Gate(
Gate_status_id varchar,
Gate_type varchar,
Gate_status varchar,
Customer_id int,
        PRIMARY KEY(Gate_status_id),
        FOREIGN KEY(Customer_id) REFERENCES Customer(Customer_id)
);
```

APPENDIX B

Alter table statements:

```
ALTER TABLE Floor
ADD PRIMARY KEY(Floor_id);

ALTER TABLE Vehicle
ALTER Vehicle_id SET NOT NULL;
ALTER TABLE Payment
ADD COLUMN Currency varchar;

ALTER TABLE Parking_Slot
ADD FOREIGN KEY(Block_name) REFERENCES Block(Block_name);

ALTER TABLE Payment
DROP COLUMN Currency;
```

APPENDIX C

Insert into statements:

INSERT INTO Parking_lot(lot_id,
organization_name,address,floors_number,floors_list,blocks_number,

blocks_list)

VALUES

('ALA01','Mega center «Alma-Ata»','Rozybakiev street 247A, Almaty',2,'1st floor, 2nd floor',4,'A, B, C, D'),

('ALA02','Mega Park','Makataeva street 127/1, Almaty',2,'1st floor, 2nd floor',4,'E, F, G, H'),

('ALA03','Dostyk Plaza',' Samal-2 111, Almaty',2,'1st floor, 2nd floor',4,'I, J, K, L'),

('ALA04','Colibri','Tashkent highway 17k, Almaty,',2,'1st floor, 2nd floor',2,'Q,R'),

('ALA05','Aport Mall','Ave. Tauelsizdik 34, Nur-Sultan',2,'1st floor, 2nd floor',2,'S,T'),

('AST01','Astana Mall','Ave. Tauelsizdik 34, Nur-Sultan',2,'1st floor, 2nd floor',2,'U,V'),

('AST02','Mega-Astana','4 \ 010000, Korgalzhinskoe highway 1, Nur-Sultan',2,'1st floor, 2nd floor',2,'W,X'),

('AST03','Khan Shatyr','Ave. Turan 37, Nur-Sultan',2,'1st floor, 2nd floor',2,'Y,Z'),

('AST04','Keruen City','Kurgalzhinskoe highway 1, Nur-Sultan',2,'1st floor, 2nd floor',2,'A1,A2'),

('AST05','MEGA Silk Way','Ave. Kabanbay Batyr 62, Nur-Sultan',2,'1st floor, 2nd floor',2,'B1,B2');

INSERT INTO Block(block_name,parking_type,slots_number,slots_list)

VALUES

('A','four-wheel',25,'101,102,103,104,105,...,125'),

('B','four-wheel',25,'126,127,128,129,130,...,150'),

('C','four-wheel',25,'201,202,203,204,205,...,225'),

('D','two-wheel',50,'226,227,228,229,230,...,275'),

('E','four-wheel',25,'101,102,103,104,105,...,125'),

('F','four-wheel',25,'126,127,128,129,130,...,150'),

('G','four-wheel',25,'201,202,203,204,205,...,225'),

('H','two-wheel',50,'226,227,228,229,230,...,275'),

('I','four-wheel',25,'101,102,103,104,105,...,125'),

('J','four-wheel',25,'126,127,128,129,130,...,150'),

('K','four-wheel',25,'201,202,203,204,205,...,225'),

('L','two-wheel',50,'226,227,228,229,230,...,275'),

('M','two-wheel',25,'275,276,277,278,279,...,300');

INSERT INTO Floor(floor_id,lot_id,floor_number,block_name)

VALUES

('1_A','ALA01','1st floor','A'),

('1_B','ALA01','1st floor','B'),

('1_C','ALA01','2nd floor','C'),

('1_D','ALA01','2nd floor','D'),

('2_E','ALA02','1st floor','E'),

('2_F','ALA02','1st floor','F'),

('2_G','ALA02','2nd floor','G'),

('2_H','ALA02','2nd floor','H'),

('3_I','ALA03','1st floor','I'),

('3_J','ALA03','1st floor','J'),

('3_K','ALA03','2nd floor','K'),

('3_L','ALA03','2nd floor','L'),

('4_M','ALA04','2nd floor','M');

INSERT INTO Parking_slot(slot_id,block_name)

VALUES

(101,'A'),

(131,'B'),

(221,'C'),

(231,'D'),

(102,'E'),

(132,'F'),

(222,'G'),

```
(232,'H'),
(103,'I'),
(133,'J'),
(223,'K'),
(233,'L'),
(288,'M');

INSERT INTO Vehicle(vehicle_id,vehicle_type,slot_id)
VALUES
('111ONE', 'four-wheel', 101),
('222TWO', 'four-wheel', 131),
('333TRI', 'four-wheel', 221),
('123ABC', 'two-wheel', 231),
('444FOR', 'four-wheel', 102),
('555FIV', 'four-wheel', 132),
('666SIX', 'four-wheel', 222),
('456DEF', 'two-wheel', 232),
('777SEV', 'four-wheel', 103),
('888EIT', 'four-wheel', 133),
('999NIN', 'four-wheel', 223),
('789HIJ', 'two-wheel', 233),
('941HEX', 'two-wheel', 288);


INSERT INTO Parking_card(parking_card_id,payment_method)
VALUES
(8801, 'cash'),
(8802, 'credit card'),
(8803, 'cash'),
(8804, 'credit card'),
(8805, 'cash'),
```

(8806, 'credit card'),

(8807, 'cash'),

(8808, 'credit card'),

(8809, 'cash'),

(8810, 'credit card'),

(8811, 'credit card'),

(8812, 'credit card'),

(8813, 'cash'),

(8814, 'credit card'),

(8815, 'cash');

INSERT INTO
Customer(customer_id,vehicle_id,parking_card_id,arrival_date,arrival_time,departure_time)

VALUES

(801, '111ONE', 8801,'2020-11-15', '10:00', '11:00'),

(802, '222TWO', 8802, '2020-11-10', '11:15', '11:29'),

(803, '333TRI', 8803, '2020-10-28', '13:20', '15:20'),

(805, '444FOR', 8805, '2020-10-15', '16:00', '20:10'),

(806, '555FIV', 8806, '2020-10-22', '20:15', '20:30'),

(808, '456DEF', 8808, '2020-11-10', '20:00', '21:00'),

(809, '777SEV', 8809, '2020-11-08', '11:00',' 17:00'),

(810, '888EIT', 8810, '2020-10-07', '13:46', '14:36'),

(811, '999NIN', 8811, '2020-11-17', '11:00', '15:05');

INSERT INTO Customer(customer_id,vehicle_id,parking_card_id,arrival_date,arrival_time)

VALUES

(804, '123ABC', 8804, '2020-11-17', '14:12'),

(807, '666SIX', 8807, '2020-11-17', '16:00'),

(812, '789HIJ', 8812, '2020-11-17', '15:12'),

(813, '941HEX', 8813, '2020-11-17', '13:35');

```
INSERT INTO Parking_rate(parking_rate_id,customer_id,time_passed)
VALUES
('R801RE', 801, '01:00'),
('R802FR', 802,'00:14'),
('R803ED', 803, '03:00'),
('R805GH', 805, '04:10'),
('R806DL', 806, '00:15'),
('R808LD', 808, '01:00'),
('R809BV', 809, '06:00'),
('R810LW', 810, '00:50'),
('R811SA', 811, '01:05');

INSERT INTO Parking_rate(parking_rate_id,customer_id)
VALUES
('R804JK', 804),
('R807BG', 807),
('R812KE', 812),
('R813ME', 813);


INSERT INTO Payment(payment_id,parking_rate_id,amount,payment_status)
VALUES
('P801RE', 'R801RE', 100, 'paid'),
('P802FR', 'R802FR', 0, 'paid'),
('P803ED', 'R803ED', 300, 'paid'),
('P805GH', 'R805GH', 400, 'paid'),
('P806DL', 'R806DL', 0, 'paid'),
('P808LD', 'R808LD', 100, 'paid'),
('P809BV', 'R809BV', 600, 'paid'),
('P810LW', 'R810LW', 100, 'paid'),
```

('P811SA', 'R811SA', 200, 'paid');

INSERT INTO Payment(payment_id,parking_rate_id,payment_status)
VALUES
('P804JK', 'R804JK', 'unpaid'),
('P807BG', 'R807BG', 'unpaid'),
('P812KE', 'R812KE', 'unpaid'),
('P813ME', 'R813ME', 'unpaid');

INSERT INTO Gate(gate_status_id,gate_type,gate_status,customer_id)
VALUES
('ALA01_E1', 'Exit gate', 'opened', 801),
('ALA01_E2', 'Exit gate', 'opened', 802),
('ALA01_E3', 'Exit gate', 'opened', 803),
('ALA01_E4', 'Exit gate', 'closed', 804),
('ALA02_E5', 'Exit gate', 'opened', 805),
('ALA02_E6', 'Exit gate', 'opened', 806),
('ALA02_E7', 'Exit gate', 'closed', 807),
('ALA02_E8', 'Exit gate', 'opened', 808),
('ALA03_E9', 'Exit gate', 'opened', 809),
('ALA03_E10', 'Exit gate', 'opened', 810),
('ALA03_E11', 'Exit gate', 'closed', 811),
('ALA03_E12', 'Exit gate', 'closed', 812),
('ALA03_E13','Exit gate', 'closed', 813);

APPENDIX D

Update and Delete statements:

```
UPDATE Gate
SET gate_status = 'opened'
WHERE gate_status_id = 'ALA03_E11';

DELETE FROM Gate
WHERE gate_status_id = 'ALA03_E13';

UPDATE Payment
SET payment_status = 'paid'
WHERE payment_id = 'P813ME';

DELETE FROM Payment
WHERE payment_id = 'P813ME';

UPDATE Parking_rate
SET time_passed = '02:05'
WHERE customer_id = 811;

DELETE FROM Parking_rate
WHERE customer_id = 813;

UPDATE Customer
SET arrival_time = '13:55'
WHERE customer_id = 813;

DELETE FROM Customer
WHERE customer_id = 813;

UPDATE Parking_card
SET payment_method = 'credit card'
```

```
WHERE parking_card_id = 8813;
DELETE FROM Parking_card
WHERE parking_card_id = 8813;

UPDATE Vehicle
SET vehicle_type = 'four-wheel'
WHERE slot_id = 288;

DELETE FROM Vehicle
WHERE slot_id = 288;

UPDATE Parking_slot
SET slot_id = 299
WHERE block_name = 'M';

DELETE FROM Parking_slot
WHERE block_name = 'M';

UPDATE Floor
SET floor_number = '1st floor'
WHERE floor_id = '4_M';

DELETE FROM Floor
WHERE floor_id = '4_M';

UPDATE Block
SET parking_type = 'four-wheel'
WHERE block_name = 'M';

DELETE FROM Block
WHERE block_name = 'M';
```

```
UPDATE parking_lot
SET organization_name = 'Mega Silk Way'
WHERE lot_id = 'AST05';

DELETE FROM Parking_lot
WHERE lot_id = 'ALA04';
```

APPENDIX E

Queries:

```
SELECT
parking_lot.organization_name,parking_lot.address,floor.floor_number,floor.block_name

FROM parking_lot

LEFT JOIN floor

ON parking_lot.lot_id=floor.lot_id

WHERE organization_name LIKE 'Mega%';


SELECT
parking_card.parking_card_id,customer.vehicle_id,customer.arrival_date,parking_card.payment
_method

FROM customer

RIGHT JOIN parking_card

ON customer.parking_card_id=parking_card.parking_card_id

WHERE payment_method IN ('cash');


SELECT COUNT(gate_status) AS num_of_gates,gate_status

FROM gate

GROUP BY gate_status;


FROM customer

FULL JOIN parking_card

ON customer.parking_card_id=parking_card.parking_card_id

WHERE arrival_date > '2020-11-01' AND arrival_time < '16:00';


SELECT MAX(amount) AS max_amount,MIN(amount) AS
min_amount,AVG(amount)::numeric(10,2)

AS avg_amount,SUM(amount) AS sum

FROM payment;


SELECT
parking_lot.organization_name,parking_lot.address,floor.lot_id,floor.floor_id,floor.floor_numbe
r
```

```
FROM parking_lot

INNER JOIN floor

ON parking_lot.lot_id=floor.lot_id

WHERE (Length(address) BETWEEN 25 AND 35) AND floor_number NOT IN ('1st floor');


SELECT
payment.amount,payment.payment_status,parking_rate.time_passed,customer.vehicle_id

FROM payment

INNER JOIN parking_rate ON payment.parking_rate_id=parking_rate.parking_rate_id

INNER JOIN customer ON parking_rate.customer_id=customer.customer_id

WHERE amount >(SELECT AVG(amount) FROM payment) OR vehicle_id = '999NIN';


SELECT
time_passed,customer.customer_id,customer.departure_time,payment.payment_status,gate.gate_
status

FROM parking_rate

INNER JOIN payment ON parking_rate.parking_rate_id=payment.parking_rate_id

INNER JOIN customer ON parking_rate.customer_id=customer.customer_id

INNER JOIN gate ON customer.customer_id=gate.customer_id

ORDER BY customer_id;


SELECT DISTINCT time_passed

FROM parking_rate

WHERE time_passed IS NOT NULL AND EXTRACT(hour from time_passed)>1;


SELECT
customer.customer_id,customer.vehicle_id,customer.parking_card_id,customer.arrival_time,

customer.departure_time,EXTRACT(HOUR from arrival_time)+EXTRACT(HOUR from
departure_time) AS sum_of_hours_of_arrival_departure_times

FROM customer

INNER JOIN gate ON customer.customer_id=gate.customer_id

WHERE gate_status IN (SELECT gate_status FROM gate GROUP BY gate_status HAVING
COUNT(*)>4) AND customer.customer_id != 810;
```

```
SELECT
parking_slot.block_name,parking_slot.slot_id,vehicle.vehicle_id,customer.customer_id,block.parking_type,customer.arrival_time,gate.gate_status

FROM block

LEFT JOIN parking_slot ON block.block_name=parking_slot.block_name

LEFT JOIN vehicle ON parking_slot.slot_id= vehicle.slot_id

LEFT JOIN customer ON vehicle.vehicle_id=customer.vehicle_id

LEFT JOIN gate ON customer.customer_id=gate.customer_id

WHERE EXTRACT(month from arrival_date)=10 AND customer.customer_id % 2 =0;


SELECT *

FROM customer

WHERE parking_card_id > (SELECT parking_card_id FROM parking_card WHERE parking_card_id=8805);


SELECT *

FROM block

WHERE block_name=ANY(SELECT block_name FROM block WHERE parking_type='four-wheel');


SELECT customer_id,vehicle_id,arrival_date,arrival_time,departure_time

FROM customer

WHERE (customer_id,arrival_date) IN (SELECT customer_id,arrival_date FROM customer WHERE arrival_date='2020-11-17' AND customer_id%2=1);


SELECT
parking_lot.organization_name,parking_lot.address,floor.lot_id,floor.floor_number,floor.block_name

FROM parking_lot

LEFT JOIN floor ON parking_lot.lot_id=floor.lot_id

WHERE organization_name IN(SELECT organization_name FROM parking_lot WHERE lot_id LIKE 'ALA%');


SELECT customer_id,arrival_date,arrival_time,departure_time

FROM customer
```

```
WHERE (arrival_date,departure_time) IN (SELECT arrival_date,departure_time FROM
customer WHERE arrival_date >'2020-10-24' AND departure_time >'14:00');


SELECT
customer.vehicle_id,customer.arrival_time,parking_rate.time_passed,payment.payment_status,p
ayment.amount

FROM payment

INNER JOIN parking_rate ON payment.parking_rate_id=parking_rate.parking_rate_id

INNER JOIN customer ON parking_rate.customer_id=customer.customer_id

WHERE payment_status IN (SELECT payment_status FROM payment WHERE
payment_status= 'paid')

ORDER BY amount DESC;
```