Temat: Wypożyczalnia samochodów

Autorzy: Piotr Kalisz, Tomasz Kostrzewa, Konrad Gromala, Jakub Kraj, Mateusz Oleksy

1. Zakres i krótki opis systemu

System wypożyczalni samochodów, o którym mowa, jest skomplikowanym i wielofunkcyjnym narzędziem, które służy do obsługi klientów chcących wynająć pojazdy, zarządzania samochodami, oferowania dodatkowych usług oraz zarządzania relacjami z firmami ubezpieczeniowymi.

Klienci, korzystając z systemu, mogą przeglądać dostępne pojazdy, wybierać różne typy i klasy samochodów zgodnie z własnymi potrzebami. System umożliwia także monitorowanie i zarządzanie historią wynajmów każdego klienta, co obejmuje daty rozpoczęcia i zakończenia wynajmu. Klienci mają również możliwość skorzystania ze zniżek, które mogą być aplikowane do transakcji za pomocą specjalnych kodów rabatowych. Kody rabatowe mogą być użyte jedynie raz.

System dba także o bieżące zarządzanie pojazdami, monitorując ich stan techniczny, przebieg, częstotliwość wymiany oleju i inne parametry techniczne, co pomaga w utrzymaniu pojazdów w dobrym stanie oraz planowaniu przeglądów i napraw. Każdy samochód jest ubezpieczony, a system zarządza również polisami ubezpieczeniowymi, w tym terminami ich ważności oraz współpracą z firmami ubezpieczeniowymi.

Dodatkowo, system posiada funkcje zarządzania danymi klientów, w tym ich informacjami kontaktowymi, adresami oraz innymi danymi osobowymi, które są niezbędne do procesu wynajmu. Klienci mogą być zlokalizowani w różnych miastach, a system umożliwia zarządzanie tymi informacjami w sposób zorganizowany.

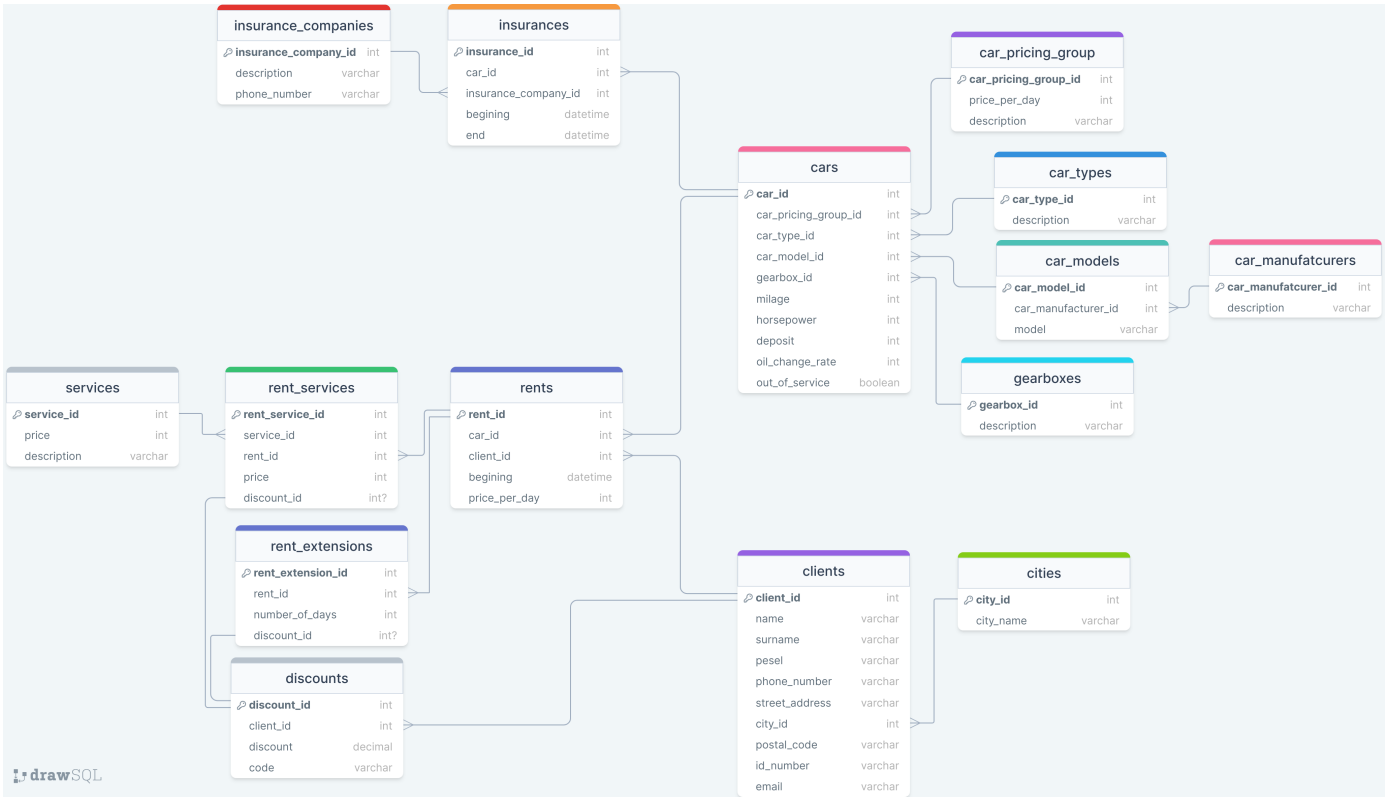
Podsumowując, system jest kompleksowym rozwiązaniem dla wypożyczalni samochodów, które ułatwia zarządzanie wynajmem, klientami, pojazdami oraz usługami dodatkowymi, a wszystko to w celu stworzenia wygodnego i efektywnego serwisu dla klientów.

2. Wymagania i funkcje systemu

- **Wynajem i Rezerwacja Pojazdów**:
    - Kategoryzacja pojazdów z uwzględnieniem szczegółowych opisów ich parametrów technicznych i wyposażenia.
    - Wynajem samochodów na określony okres.
    - Ustalanie i aktualizacja stawek wynajmu oraz cen za usługi dodatkowe.
- **Zarządzanie Klientami**:
    - Gromadzenie, aktualizacja i zarządzanie danymi osobowymi klientów.
    - Monitorowanie historii wynajmów dla każdego klienta.
    - Zarządzanie danymi dotyczącymi lokalizacji klientów.
- **Zarządzanie Flotą Pojazdów**:
    - Śledzenie dostępności pojazdów w czasie rzeczywistym, zapewniające efektywne zarządzanie flotą.
    - Monitorowanie stanu technicznego i serwisowania pojazdów.
    - Zarządzanie polisami ubezpieczeniowymi przypisanymi do pojazdów.
- **Zarządzanie Zniżkami**:
    - Tworzenie i zarządzanie ofertami promocyjnymi i zniżkami dla klientów.
    - Implementacja systemu kodów rabatowych przy transakcjach wynajmu.
- **Analiza i Raportowanie Danych**:
    - Rejestrowanie danych o terminach wynajmu i kompleksowa historia transakcji.
    - Analiza zebranych danych w celu optymalizacji procesów i oferty wypożyczalni.

3. Projekt bazy danych

Schemat bazy danych



Opis poszczególnych tabel

- **rent_extensions:** Tabela przechowuje informacje o przedłużeniach wynajmu. Każde przedłużenie jest powiązane z określonym wynajmem i może obejmować zniżkę.
    - `rent_extension_id`: Unikalny identyfikator przedłużenia wynajmu.
    - `rent_id`: Identyfikator wynajmu, do którego odnosi się przedłużenie.
    - `number_of_days`: Liczba dni przedłużenia wynajmu.
    - `discount_id`: Identyfikator zniżki zastosowanej do przedłużenia (opcjonalnie).
- **rents:** Tabela zawiera informacje o wynajmach samochodów. Każdy wpis odnosi się do konkretnego samochodu i klienta.
    - `rent_id`: Unikalny identyfikator wynajmu.
    - `car_id`: Identyfikator wynajmowanego samochodu.
    - `client_id`: Identyfikator klienta wynajmującego samochód.
    - `begining`: Data i godzina rozpoczęcia wynajmu.
    - `price_per_day`: Cena wynajmu za dzień.
- **car_pricing_group:** Tabela przechowuje informacje o grupach cenowych samochodów.
    - `car_pricing_group_id`: Unikalny identyfikator grupy cenowej.
    - `price_per_day`: Cena wynajmu za dzień w tej grupie cenowej.
    - `description`: Opis grupy cenowej.
- **discounts:** Tabela zawiera informacje o zniżkach przyznawanych klientom.
    - `discount_id`: Unikalny identyfikator zniżki.
    - `client_id`: Identyfikator klienta, który otrzymał zniżkę.
    - `discount`: Wartość zniżki procentowo.
    - `code`: Kod zniżki.
- **car_models:** Tabela przechowuje informacje o modelach samochodów.
    - `car_model_id`: Unikalny identyfikator modelu samochodu.
    - `car_manufacturer_id`: Identyfikator producenta samochodu.
    - `model`: Nazwa modelu samochodu.
- **car_types:** Tabela zawiera informacje o typach samochodów.
    - `car_type_id`: Unikalny identyfikator typu samochodu.
    - `description`: Opis typu samochodu.

- **insurances:** Tabela przechowuje informacje o ubezpieczeniach samochodów.
    - `insurance_id`: Unikalny identyfikator ubezpieczenia.
    - `car_id`: Identyfikator ubezpieczonego samochodu.
    - `insurance_company_id`: Identyfikator firmy ubezpieczeniowej.
    - `begining`: Data rozpoczęcia ubezpieczenia.
    - `end`: Data zakończenia ubezpieczenia.
- **cities:** Tabela zawiera informacje o miastach.
    - `city_id`: Unikalny identyfikator miasta.
    - `city_name`: Nazwa miasta.
- **insurance_companies:** Tabela przechowuje informacje o firmach ubezpieczeniowych.
    - `insurance_company_id`: Unikalny identyfikator firmy ubezpieczeniowej.
    - `phone_number`: Numer telefonu firmy ubezpieczeniowej.
    - `description`: Opis firmy ubezpieczeniowej.
- **car_manufacurers:** Tabela zawiera informacje o producentach samochodów.
    - `car_manufacurer_id`: Unikalny identyfikator producenta samochodu.
    - `description`: Opis producenta samochodu.
- **services:** Tabela przechowuje informacje o usługach dodatkowych.
    - `service_id`: Unikalny identyfikator usługi.
    - `price`: Cena usługi.
    - `description`: Opis usługi.
- **cars:** Tabela zawiera informacje o samochodach dostępnych do wynajmu.
    - `car_id`: Unikalny identyfikator samochodu.
    - `car_pricing_group_id`: Identyfikator grupy cenowej samochodu.
    - `car_type_id`: Identyfikator typu samochodu.
    - `car_model_id`: Identyfikator modelu samochodu.
    - `gearbox_id`: Identyfikator skrzyni biegów.
    - `milage`: Przebieg samochodu.
    - `horsepower`: Moc silnika w koniach mechanicznych.
    - `deposit`: Kaucja za wynajem samochodu.
    - `oil_change_rate`: Częstotliwość wymiany oleju w kilometrach.
    - `out_of_service`: Status dostępności samochodu (0 - dostępny, 1 - niedostępny).
- **gearboxes:** Tabela przechowuje informacje o skrzyniach biegów.
    - `gearbox_id`: Unikalny identyfikator skrzyni biegów.
    - `description`: Opis skrzyni biegów.
- **rent_services:** Tabela zawiera informacje o usługach dodatkowych powiązanych z wynajmem.
    - `rent_service_id`: Unikalny identyfikator usługi wynajmu.
    - `service_id`: Identyfikator usługi.
    - `rent_id`: Identyfikator wynajmu.
    - `price`: Cena usługi wynajmu.
    - `discount_id`: Identyfikator zniżki na usługę wynajmu (opcjonalnie).
- **clients:** Tabela przechowuje informacje o klientach.
    - `client_id`: Unikalny identyfikator klienta.
    - `name`: Imię klienta.
    - `surname`: Nazwisko klienta.
    - `pesel`: Numer PESEL klienta.
    - `phone_number`: Numer telefonu klienta.
    - `street_address`: Adres zamieszkania klienta.
    - `city_id`: Identyfikator miasta, w którym mieszka klient.
    - `postal_code`: Kod pocztowy klienta.
    - `id_number`: Numer dowodu osobistego klienta.
    - `email`: Adres e-mail klienta.

4. Implementacja

Kod poleceń DDL

```
CREATE TABLE car_pricing_group(
    car_pricing_group_id INT NOT NULL IDENTITY(1, 1),
    price_per_day INT NOT NULL CHECK (price_per_day > 0),
    description VARCHAR(255) NOT NULL,
    PRIMARY KEY (car_pricing_group_id),
    CONSTRAINT car_pricing_group_unique_description UNIQUE (description)
);
```

```
CREATE TABLE car_types(
    car_type_id INT NOT NULL IDENTITY(1, 1),
    description VARCHAR(255) NOT NULL,
    PRIMARY KEY (car_type_id),
    CONSTRAINT car_types_unique_description UNIQUE (description)
);
```

```
CREATE TABLE car_manufacturers(
    car_manufacturer_id INT NOT NULL IDENTITY(1, 1),
    description VARCHAR(255) NOT NULL,
    PRIMARY KEY (car_manufacturer_id),
    CONSTRAINT car_manufacturers_unique_description UNIQUE (description)
);
```

```
CREATE TABLE car_models(
    car_model_id INT NOT NULL IDENTITY(1, 1),
    car_manufacturer_id INT NOT NULL,
    model VARCHAR(32) NOT NULL,
    PRIMARY KEY (car_model_id),
    CONSTRAINT car_models_unique_model UNIQUE (car_manufacturer_id, model),
    CONSTRAINT car_models_car_manufacturer_id FOREIGN KEY (car_manufacturer_id) REFERENCES
car_manufacturers(car_manufacturer_id)
);
```

```
CREATE TABLE gearboxes(
    gearbox_id INT NOT NULL IDENTITY(1, 1),
    description VARCHAR(255) NOT NULL,
    PRIMARY KEY (gearbox_id),
    CONSTRAINT gearboxes_unique_description UNIQUE (description)
);
```

```
CREATE TABLE cars(
    car_id INT NOT NULL IDENTITY(1, 1),
    car_pricing_group_id INT NOT NULL,
    car_type_id INT NOT NULL,
    car_model_id INT NOT NULL,
    gearbox_id INT NOT NULL,
    milage INT NOT NULL CHECK (milage >= 0),
    horsepower INT NOT NULL CHECK (horsepower > 0),
    deposit INT NOT NULL CHECK (deposit >= 0),
    oil_change_rate INT NOT NULL CHECK (oil_change_rate > 0),
    out_of_service BIT NOT NULL,
    PRIMARY KEY (car_id),
    CONSTRAINT cars_car_pricing_group_id FOREIGN KEY (car_pricing_group_id) REFERENCES
car_pricing_group(car_pricing_group_id),
    CONSTRAINT cars_car_type_id FOREIGN KEY (car_type_id) REFERENCES car_types(car_type_id),
```

```
    CONSTRAINT cars_car_model_id FOREIGN KEY (car_model_id) REFERENCES car_models(car_model_id),
    CONSTRAINT cars_gearbox_id FOREIGN KEY (gearbox_id) REFERENCES gearboxes(gearbox_id)
);
```

```
CREATE TABLE cities(
    city_id INT NOT NULL IDENTITY(1, 1),
    city_name VARCHAR(32) NOT NULL,
    PRIMARY KEY (city_id),
    CONSTRAINT cities_unique_city_name UNIQUE (city_name)
);
```

```
CREATE TABLE clients(
    client_id INT NOT NULL IDENTITY(1, 1),
    name VARCHAR(32) NOT NULL,
    surname VARCHAR(32) NOT NULL,
    pesel VARCHAR(11) NOT NULL,
    phone_number VARCHAR(16) NOT NULL,
    street_address VARCHAR(32) NOT NULL,
    city_id INT NOT NULL,
    postal_code VARCHAR(6) NOT NULL,
    id_number VARCHAR(16) NOT NULL,
    email VARCHAR(32) NOT NULL,
    PRIMARY KEY (client_id),
    CONSTRAINT clients_unique_email UNIQUE (name, surname, email),
    CONSTRAINT clients_city_id FOREIGN KEY (city_id) REFERENCES cities(city_id)
);
```

```
CREATE TABLE rents(
    rent_id INT NOT NULL IDENTITY(1, 1),
    car_id INT NOT NULL,
    client_id INT NOT NULL,
    begining DATETIME NOT NULL,
    price_per_day INT NOT NULL CHECK (price_per_day > 0),
    PRIMARY KEY (rent_id),
    CONSTRAINT rents_car_id FOREIGN KEY (car_id) REFERENCES cars(car_id),
    CONSTRAINT rents_client_id FOREIGN KEY (client_id) REFERENCES clients(client_id)
);
```

```
CREATE TABLE discounts(
    discount_id INT NOT NULL IDENTITY(1, 1),
    client_id INT NOT NULL,
    discount DECIMAL(3, 2) NOT NULL CHECK (discount > 0 AND discount < 1),
    code VARCHAR(8) NOT NULL,
    PRIMARY KEY (discount_id),
    CONSTRAINT discounts_unique_code UNIQUE (client_id, code),
    CONSTRAINT discounts_client_id FOREIGN KEY (client_id) REFERENCES clients(client_id)
);
```

```
CREATE TABLE rent_extensions(
    rent_extension_id INT NOT NULL IDENTITY(1, 1),
    rent_id INT NOT NULL,
    number_of_days INT NOT NULL CHECK (number_of_days > 0),
    discount_id INT NULL,
    PRIMARY KEY (rent_extension_id),
    CONSTRAINT rent_extensions_rent_id FOREIGN KEY (rent_id) REFERENCES rents(rent_id),
    CONSTRAINT rent_extensions_discount_id FOREIGN KEY (discount_id) REFERENCES discounts(discount_id)
);
```

```
CREATE TABLE insurance_companies(
    insurance_company_id INT NOT NULL IDENTITY(1, 1),
    phone_number VARCHAR(16) NOT NULL,
    description VARCHAR(255) NOT NULL,
    PRIMARY KEY (insurance_company_id),
    CONSTRAINT insurance_companies_unique_description UNIQUE (description)
);
```

```
CREATE TABLE insurances(
    insurance_id INT NOT NULL IDENTITY(1, 1),
    car_id INT NOT NULL,
    insurance_company_id INT NOT NULL,
    begining_date DATETIME NOT NULL,
    end_date DATETIME NOT NULL,
    PRIMARY KEY (insurance_id),
    CONSTRAINT insurances_car_id FOREIGN KEY (car_id) REFERENCES cars(car_id),
    CONSTRAINT insurances_insurance_company_id FOREIGN KEY (insurance_company_id) REFERENCES
insurance_companies(insurance_company_id),
    CONSTRAINT date_check CHECK (begining_date < end_date)
);
```

```
CREATE TABLE services(
    service_id INT NOT NULL IDENTITY(1, 1),
    price INT NOT NULL CHECK (price > 0),
    description VARCHAR(255) NOT NULL,
    PRIMARY KEY (service_id),
    CONSTRAINT services_unique_description UNIQUE (description)
);
```

```
CREATE TABLE rent_services(
    rent_service_id INT NOT NULL IDENTITY(1, 1),
    service_id INT NOT NULL,
    rent_id INT NOT NULL,
    price INT NOT NULL CHECK (price > 0),
    discount_id INT NULL,
    PRIMARY KEY (rent_service_id),
    CONSTRAINT rent_services_service_id FOREIGN KEY (service_id) REFERENCES services(service_id),
    CONSTRAINT rent_services_rent_id FOREIGN KEY (rent_id) REFERENCES rents(rent_id),
    CONSTRAINT rent_services_discount_id FOREIGN KEY (discount_id) REFERENCES discounts(discount_id)
);
```

Widoki

Widok rents_timespan:

- **Implementacja:**

```
CREATE VIEW rents_timespan AS
SELECT
    rents.rent_id,
    rents.car_id,
    rents.client_id,
    rents.begining as begining_date,
    DATEADD(DAY, SUM(rent_extensions.number_of_days), rents.begining) as ending_date
FROM rents
JOIN rent_extensions
    ON rent_extensions.rent_id = rents.rent_id
GROUP BY rents.rent_id, rents.car_id, rents.client_id, rents.begining;
```

- **Effekt użycia:**

	rent_id	car_id	client_id	begining_date	ending_date
1	1	1	1	2023-06-01 10:00:00.000	2023-06-11 10:00:00.000
2	2	2	2	2023-06-02 11:00:00.000	2023-07-02 11:00:00.000
3	3	3	3	2023-06-03 12:00:00.000	2023-06-05 12:00:00.000
4	4	4	4	2023-06-04 13:00:00.000	2023-06-11 13:00:00.000
5	5	5	5	2023-06-05 14:00:00.000	2024-09-01 14:00:00.000
6	6	6	6	2023-06-06 15:00:00.000	2023-06-12 15:00:00.000
7	7	7	7	2023-06-07 16:00:00.000	2023-06-12 16:00:00.000
8	8	8	8	2023-06-08 17:00:00.000	2023-06-11 17:00:00.000
9	9	9	9	2023-06-09 18:00:00.000	2023-06-13 18:00:00.000
10	10	10	10	2023-06-10 19:00:00.000	2023-06-12 19:00:00.000
11	11	11	11	2023-06-11 10:00:00.000	2023-06-17 10:00:00.000
12	12	12	12	2023-06-12 11:00:00.000	2023-06-15 11:00:00.000
13	13	1	13	2023-06-13 12:00:00.000	2024-08-13 12:00:00.000
14	14	2	14	2023-06-14 13:00:00.000	2023-06-19 13:00:00.000
15	15	3	15	2023-06-15 14:00:00.000	2024-08-10 14:00:00.000

Widok rents_active:

- **Implementacja:**

```
CREATE VIEW rents_active AS
SELECT
    rent_id,
    car_id,
    client_id,
    begining_date,
    ending_date
FROM rents_timespan
WHERE begining_date < GETDATE() AND ending_date > GETDATE();
```

- **Effekt użycia:**

	rent_id	car_id	client_id	begining_date	ending_date
1	5	5	5	2023-06-05 14:00:00.000	2024-09-01 14:00:00.000
2	13	1	13	2023-06-13 12:00:00.000	2024-08-13 12:00:00.000
3	15	3	15	2023-06-15 14:00:00.000	2024-08-10 14:00:00.000
4	18	6	3	2023-06-18 17:00:00.000	2024-11-04 17:00:00.000
5	19	7	4	2023-06-19 18:00:00.000	2024-10-28 18:00:00.000
6	21	9	6	2023-06-21 10:00:00.000	2024-10-27 10:00:00.000
7	26	2	11	2023-06-26 15:00:00.000	2024-08-21 15:00:00.000
8	30	6	15	2023-06-30 19:00:00.000	2024-08-08 19:00:00.000
9	31	14	3	2024-06-28 19:47:56.353	2024-06-30 19:47:56.353
10	32	13	3	2024-06-28 19:51:11.200	2024-06-30 19:51:11.200
11	38	11	3	2024-06-28 20:28:42.757	2024-06-30 20:28:42.757
12	39	4	5	2024-06-28 20:32:17.050	2024-07-05 20:32:17.050

Widok rents_expired:

- **Implementacja:**

```
CREATE VIEW rents_expired AS
SELECT
    rent_id,
    car_id,
    client_id,
    begining_date,
    ending_date
FROM rents_timespan
WHERE ending_date < GETDATE();
```

- **Effekt użycia:**

	rent_id	car_id	client_id	begining_date	ending_date
1	1	1	1	2023-06-01 10:00:00.000	2023-06-11 10:00:00.000
2	2	2	2	2023-06-02 11:00:00.000	2023-07-02 11:00:00.000
3	3	3	3	2023-06-03 12:00:00.000	2023-06-05 12:00:00.000
4	4	4	4	2023-06-04 13:00:00.000	2023-06-11 13:00:00.000
5	6	6	6	2023-06-06 15:00:00.000	2023-06-12 15:00:00.000
6	7	7	7	2023-06-07 16:00:00.000	2023-06-12 16:00:00.000
7	8	8	8	2023-06-08 17:00:00.000	2023-06-11 17:00:00.000
8	9	9	9	2023-06-09 18:00:00.000	2023-06-13 18:00:00.000
9	10	10	10	2023-06-10 19:00:00.000	2023-06-12 19:00:00.000
10	11	11	11	2023-06-11 10:00:00.000	2023-06-17 10:00:00.000
11	12	12	12	2023-06-12 11:00:00.000	2023-06-15 11:00:00.000
12	14	2	14	2023-06-14 13:00:00.000	2023-06-19 13:00:00.000
13	16	4	1	2023-06-16 15:00:00.000	2023-06-22 15:00:00.000
14	17	5	2	2023-06-17 16:00:00.000	2023-06-20 16:00:00.000
15	20	8	5	2023-06-20 19:00:00.000	2023-06-22 19:00:00.000

Widok discounts_used:

- **Implementacja:**

```
CREATE VIEW discounts_used AS
SELECT discount_id FROM rent_services
WHERE discount_id IS NOT NULL
UNION
SELECT discount_id FROM rent_extensions
WHERE discount_id IS NOT NULL;
```

- **Effekt użycia:**

	discount_id
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15

Widok discounts_available:

- **Implementacja:**

```
CREATE VIEW discounts_available AS
SELECT client_id, discount_id, code
FROM discounts
WHERE (SELECT COUNT(*) FROM discounts_used WHERE discount_id = discounts.discount_id) = 0;
```

- **Effekt użycia:**

	client_id	discount_id	code
1	3	18	DISC2X
2	5	20	DISC3X
3	7	22	DISC15X
4	8	23	DISC20X
5	9	24	DISC25X
6	10	25	DISC30X
7	11	26	DISC35X
8	15	30	DISC55X

Widok car_list:

- **Implementacja:**

```
CREATE VIEW car_list AS
SELECT
    cars.car_id,
    car_types.description as car_type,
    car_models.model as car_model,
    car_manufacturers.description as car_manufacturer,
```

```
    car_pricing_group.description as car_pricing_group,
    gearboxes.description as car_gearbox,
    cars.out_of_service as car_out_of_service
FROM cars
JOIN car_types
    ON cars.car_type_id = car_types.car_type_id
JOIN car_models
    ON cars.car_model_id = car_models.car_model_id
JOIN car_manufacturers
    ON car_models.car_manufacturer_id = car_manufacturers.car_manufacturer_id
JOIN car_pricing_group
    ON cars.car_pricing_group_id = car_pricing_group.car_pricing_group_id
JOIN gearboxes
    ON cars.gearbox_id = gearboxes.gearbox_id;
```

- **Effekt użycia:**

	car_id	car_type	car_model	car_manufacturer	car_pricing_group	car_gearbox
1	1	Sedan	Corolla	Toyota	Economy	Automatic
2	2	SUV	Camry	Toyota	Compact	Automatic
3	3	Sedan	Civic	Honda	Standard	Manual
4	4	SUV	Accord	Honda	Full Size	Automatic
5	5	Convertible	Focus	Ford	Premium	CVT
6	6	Coupe	Mustang	Ford	Luxury	Automatic
7	7	SUV	Malibu	Chevrolet	Compact	Automatic
8	8	Coupe	Impala	Chevrolet	Standard	CVT
9	9	Coupe	3 Series	BMW	Full Size	Automatic
10	10	Hatchback	X5	BMW	Premium	Automatic
11	11	Sedan	C-Class	Mercedes-Benz	Luxury	Automatic
12	12	SUV	GLA	Mercedes-Benz	Economy	Automatic
13	13	SUV	RAV4	Toyota	Economy	Manual
14	14	Station Wagon	TIPO	FIAT	Economy	Manual
15	15	Sedan	TIPO	FIAT	Economy	Automatic

Widok car_prices:

- **Implementacja:**

```
CREATE VIEW car_prices AS
SELECT
    cars.car_id as car_id,
    car_pricing_group.price_per_day as car_price_per_day,
    car_pricing_group.description as car_price_description
FROM cars
    JOIN car_pricing_group
        ON cars.car_pricing_group_id = car_pricing_group.car_pricing_group_id;
```

- **Effect użycia:**

	car_id	car_price_per_day	car_price_description
1	1	50	Economy
2	2	75	Compact
3	3	100	Standard
4	4	150	Full Size
5	5	200	Premium
6	6	250	Luxury
7	7	75	Compact
8	8	100	Standard
9	9	150	Full Size
10	10	200	Premium
11	11	250	Luxury
12	12	50	Economy
13	13	50	Economy
14	14	50	Economy
15	15	50	Economy

Widok cars_available:

- **Implementacja:**

```sql
CREATE VIEW cars_available AS
SELECT
    car_id,
    car_type,
    car_model,
    car_manufacturer,
    car_pricing_group,
    car_gearbox
FROM car_list
WHERE car_id NOT IN (SELECT car_id FROM rents_active) AND car_out_of_service = 0;
```

- **Effekt użycia:**

	car_id	car_type	car_model	car_manufacturer	car_pricing_group	car_gearbox
1	1	Sedan	Corolla	Toyota	Economy	Automatic
2	2	SUV	Camry	Toyota	Compact	Automatic
3	3	Sedan	Civic	Honda	Standard	Manual
4	4	SUV	Accord	Honda	Full Size	Automatic
5	5	Convertible	Focus	Ford	Premium	CVT
6	7	SUV	Malibu	Chevrolet	Compact	Automatic
7	8	Coupe	Impala	Chevrolet	Standard	CVT
8	9	Coupe	3 Series	BMW	Full Size	Automatic
9	11	Sedan	C-Class	Mercedes-Benz	Luxury	Automatic
10	13	SUV	RAV4	Toyota	Economy	Manual
11	14	Station Wagon	TIPO	FIAT	Economy	Manual

Procedury/funkcje

Procedura AddRentExtension:

- **Implementacja:**

```sql
CREATE PROCEDURE AddRentExtension
    @rent_id INT,
    @number_of_days INT,
    @discount_code VARCHAR(8) = NULL
AS
BEGIN
    DECLARE @discount_id INT = NULL;
    DECLARE @client_id INT = NULL;

    IF @rent_id NOT IN (SELECT rent_id FROM rents_active)
    BEGIN
        THROW 50001, 'The rent has already expired!', 1;
    END

    SELECT @client_id = client_id
    FROM rents
    WHERE rent_id = @rent_id;

    IF @discount_code IS NOT NULL
    BEGIN
        IF @discount_code NOT IN (SELECT code FROM discounts_available WHERE client_id =
@client_id)
        BEGIN
            THROW 50000, 'Discount is not available!', 1;
        END
        SET @discount_id = (
            SELECT discount_id
            FROM discounts
            WHERE code = @discount_code AND client_id = @client_id
        );
    END

    INSERT INTO rent_extensions(rent_id, number_of_days, discount_id)
    VALUES (
        @rent_id,
        @number_of_days,
        @discount_id
    );
END;
```

Procedura AddRent:

- **Implementacja:**

```
CREATE PROCEDURE AddRent
    @car_id INT,
    @client_id INT,
    @number_of_days INT,
    @discount_code VARCHAR(8) = NULL
AS
BEGIN
    DECLARE @car_price_per_day INT;
    DECLARE @rent_id INT;
    DECLARE @discount_id INT = NULL;

    IF @car_id NOT IN (SELECT car_id FROM cars_available)
    BEGIN
        THROW 50002, 'Car is not available', 1;
    END

    IF @discount_code IS NOT NULL
    BEGIN
        IF @discount_code NOT IN (SELECT code FROM discounts_available WHERE client_id =
@client_id)
        BEGIN
            THROW 50000, 'Discount is not available!', 1;
        END
        SET @discount_id = (
            SELECT discount_id
            FROM discounts
            WHERE code = @discount_code AND client_id = @client_id
        );
    END

    SELECT @car_price_per_day = car_price_per_day
    FROM car_prices
    WHERE car_id = @car_id;

    INSERT INTO rents (car_id, client_id, begining, price_per_day)
    VALUES (
        @car_id,
        @client_id,
        GETDATE(),
        @car_price_per_day
    );

    SELECT @rent_id = SCOPE_IDENTITY();
    INSERT INTO rent_extensions(rent_id, number_of_days, discount_id)
    VALUES (
        @rent_id,
        @number_of_days,
        @discount_id
    );
END;
```

Procedura AddRentServicePurchase:

- **Implementacja:**

```
CREATE PROCEDURE AddRentServicePurchase
    @rent_id INT,
    @service_id INT,
    @discount_code VARCHAR(8) = NULL
AS
BEGIN
    DECLARE @discount_id INT = NULL;
    DECLARE @discount_value DECIMAL(3, 2) = 0.00;
    DECLARE @client_id INT;
```

```
    DECLARE @original_price INT;

    IF @rent_id NOT IN (SELECT rent_id FROM rents_active)
    BEGIN
        THROW 50001, 'The rent has already expired!', 1;
    END

    IF @discount_code IS NOT NULL
    BEGIN
        IF @discount_code NOT IN (SELECT code FROM discounts_available)
        BEGIN
            THROW 50000, 'Discount is not available!', 1;
        END
        SET @discount_id = (SELECT discount_id FROM discounts WHERE code = @discount_code);
        SET @discount_value = (SELECT discount FROM discounts WHERE code = @discount_code);
    END

    SELECT @client_id = client_id
    FROM rents
    WHERE rent_id = @rent_id;

    SELECT @original_price = price
    FROM services
    WHERE service_id = @service_id;

    INSERT INTO rent_services (rent_id, service_id, price, discount_id)
    VALUES (
        @rent_id,
        @service_id,
        @original_price * (1 - @discount_value),
        @discount_id
    );
END;
```

Procedura AddNewCar:

- **Implementacja:**

```
CREATE PROCEDURE AddNewCar
    @CarPricingGroupId INT,
    @CarTypeDescription VARCHAR(255),
    @CarManufacturerDescription VARCHAR(255),
    @CarModelDescription VARCHAR(32),
    @GearboxId INT,
    @Milage INT,
    @Horsepower INT,
    @Deposit INT,
    @OilChangeRate INT
AS
BEGIN
    DECLARE @CarTypeId INT;
    DECLARE @CarManufacturerId INT;
    DECLARE @CarModelId INT;

    IF @CarTypeDescription IN (SELECT description FROM car_types)
    BEGIN
        SELECT @CarTypeId = car_type_id FROM car_types WHERE description = @CarTypeDescription;
    END
    ELSE
    BEGIN
        INSERT INTO car_types (description) VALUES (@CarTypeDescription);
        SELECT @CarTypeId = SCOPE_IDENTITY();
    END

    IF @CarManufacturerDescription IN (SELECT description FROM car_manufacturers)
    BEGIN
        SELECT @CarManufacturerId = car_manufacturer_id FROM car_manufacturers WHERE description =
@CarManufacturerDescription;
    END
```

```
    ELSE
    BEGIN
        INSERT INTO car_manufacturers (description) VALUES (@CarManufacturerDescription);
        SELECT @CarManufacturerId = SCOPE_IDENTITY();
    END

    IF @CarModelDescription IN (SELECT model FROM car_models)
    BEGIN
        SELECT @CarModelId = car_model_id FROM car_models WHERE car_manufacturer_id =
@CarManufacturerId AND model = @CarModelDescription;
    END
    ELSE
    BEGIN
        INSERT INTO car_models (car_manufacturer_id, model) VALUES (@CarManufacturerId,
@CarModelDescription);
        SELECT @CarModelId = SCOPE_IDENTITY();
    END

    INSERT INTO cars (
        car_pricing_group_id,
        car_type_id,
        car_model_id,
        gearbox_id,
        milage,
        horsepower,
        deposit,
        oil_change_rate,
        out_of_service
    ) VALUES (
        @CarPricingGroupId,
        @CarTypeId,
        @CarModelId,
        @GearboxId,
        @Milage,
        @Horsepower,
        @Deposit,
        @OilChangeRate,
        0
    );
END;
```