Tandon School of Engineering

Principles of Database Systems
(CS-GY 6083 – B, Spring 2022)

Personal Project Part 2 Report

World of Wheels

Name: Shunyi Zhu NetId: sz3719

Date: 10 May 2022

[rev. 2.0]

Contents

1 Business Case

In the first part of the project, we design a database system helping the car rental company. We define two types of the customer, the individual customer with the discriminator as "I" and the member in the corporation with the discriminator as "C".

They share the common property in customer id, location (using street, city and zip-code), email, phone and customer type. For individual members, we have personal information such as name, insurance information and license number. For corporation users, we simply record employ id and connect the entity to the corporation entity with corporation information included. In this relationship, it is one-to-many since one corporation can have multiple employers. We also use two separate entities for coupon. All the other relationship is one-to-one.

For the location and vehicle part, we set the office entity for the office information, vehicle entity for the vehicle information and class entity to generalize the vehicle type. The relationship is one-to-many since one office can hold multiple vehicles and one class can have many vehicles.

For each vehicle, it can be rent for several times, therefore, the relationship is one-to-many. We can also apply the similar process to the relationship between customer and order. For each order, we have one invoice so the relationship is one-to-one. As is mentioned in the project requirement, one can use multiple payments for the invoice, therefore the relationship is one-to-many.

Business Assumptions:

1. We use 2 bit char to define the class for the vehicle type, for example, using "01" for small size car, "02" for medium size car, etc.
2. We record the daily rate of service and fees for over mileage in the order table so that we can track the information and calculate the total value for the bill together.
3. We define the coupon for individual members and corporation separately. We record the rate, start date and end date for individual and only the rate for corporation users.
4. We use street, city and zipcode for defining the location for the office.
5. We use first name, middle name, last name to define the basic information for individual member where first name and last name are mandatory. We also included insurance.
6. For corporation members, we simply use employ id for the information while connected to the corporation block with corporation id and name.
7. We use customer type to define the sub-types for individual user and corporation user.

2 Model Design

In order the system can work better in practical application, we've improved our model design of project part I.

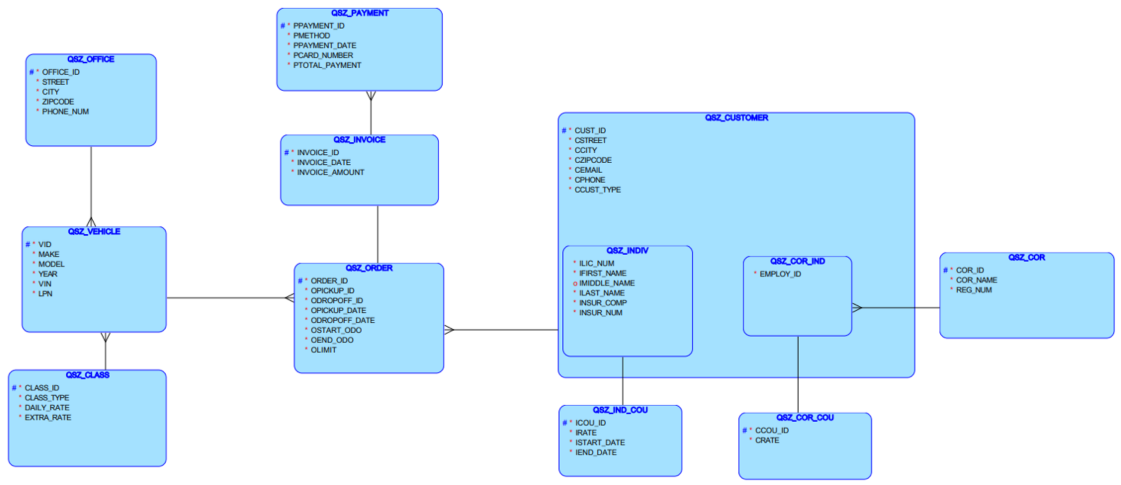


Figure 1: Logical Model

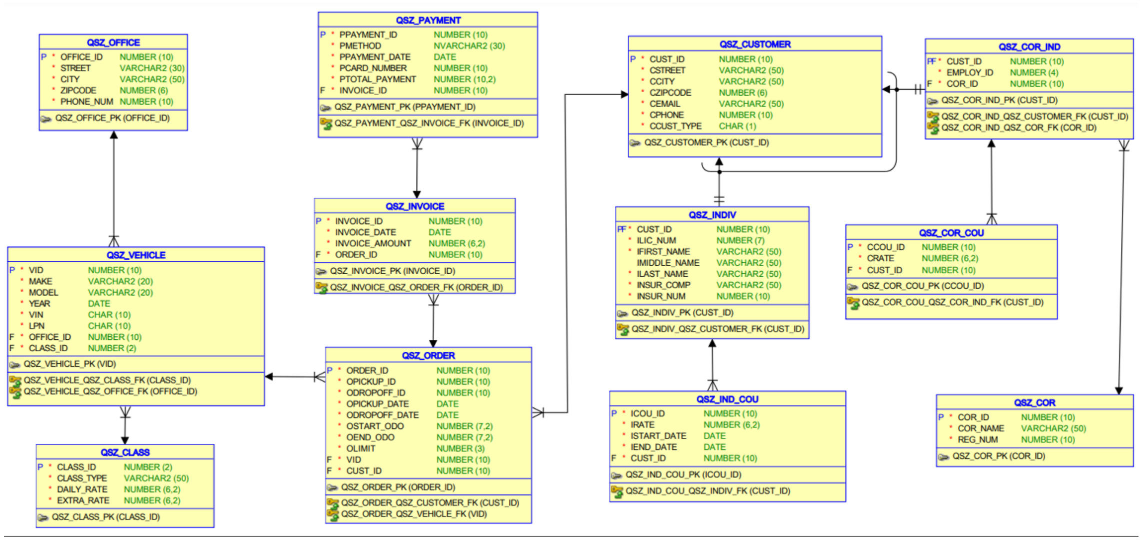
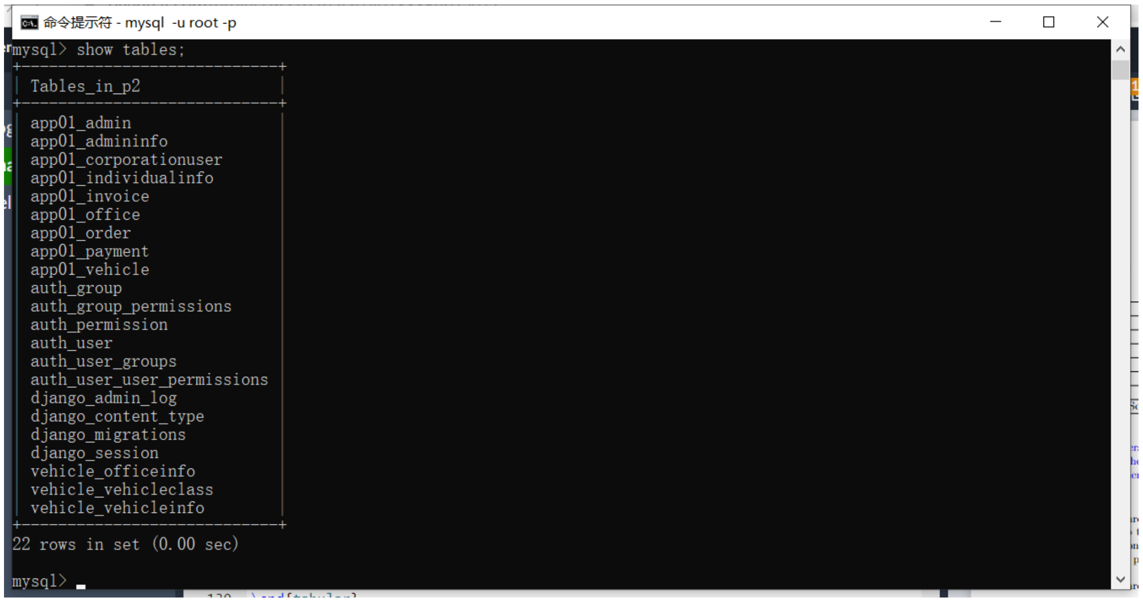


Figure 2: Relational Model

3 Technology Stack

FrontEnd	JavaScript	Html	css	js	Bootstrap
BackEnd	Python				
FrameWork	Django				
Encryption	md5				
ul protection	middleware	cookie	session		
Design	Oracle Data Modeler				
Database	mysql	ORM	BootStrapModelForm		
Self-designed Tool	pagination	CAPTCHA			

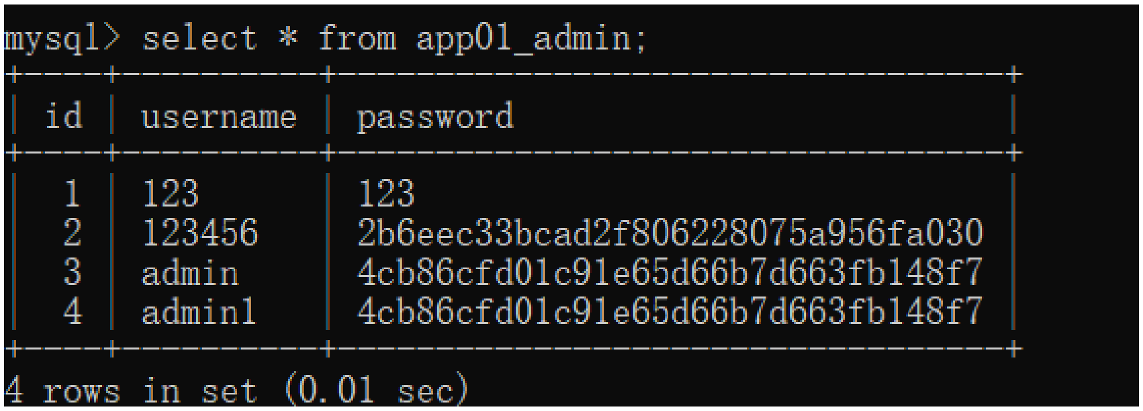
4 Table Information



命令提示符 - mysql -u root -p

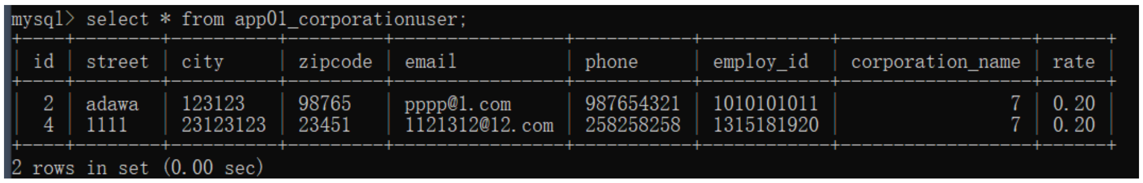
```
mysql> show tables;
+----------------------------+
| Tables_in_p2               |
+----------------------------+
| app01_admin                |
| app01_admininfo            |
| app01_corporationuser      |
| app01_individualinfo       |
| app01_invoice              |
| app01_office               |
| app01_order                |
| app01_payment              |
| app01_vehicle              |
| auth_group                 |
| auth_group_permissions     |
| auth_permission            |
| auth_user                  |
| auth_user_groups           |
| auth_user_user_permissions |
| django_admin_log           |
| django_content_type        |
| django_migrations          |
| django_session             |
| vehicle_officeinfo         |
| vehicle_vehicleclass       |
| vehicle_vehicleinfo        |
+----------------------------+
22 rows in set (0.00 sec)

mysql> _
```

Figure 3: Table Information



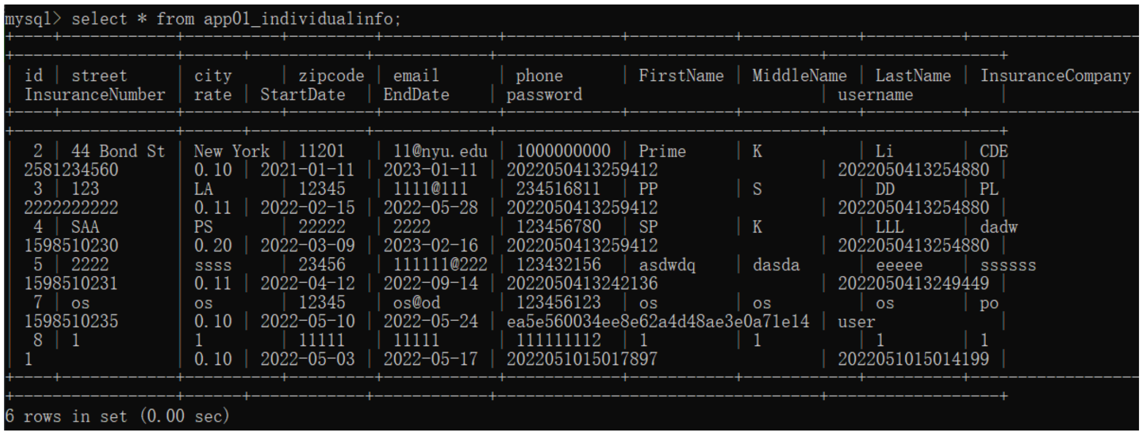
```
mysql> select * from app01_admin;
+----+-----------+----------------------------------+
| id | username  | password                         |
+----+-----------+----------------------------------+
|  1 | 123       | 123                              |
|  2 | 123456    | 2b6eec33bcad2f806228075a956fa030 |
|  3 | admin     | 4cb86cfd01c91e65d66b7d663fb148f7 |
|  4 | admin1    | 4cb86cfd01c91e65d66b7d663fb148f7 |
+----+-----------+----------------------------------+
4 rows in set (0.01 sec)
```

Figure 4: Admin Information



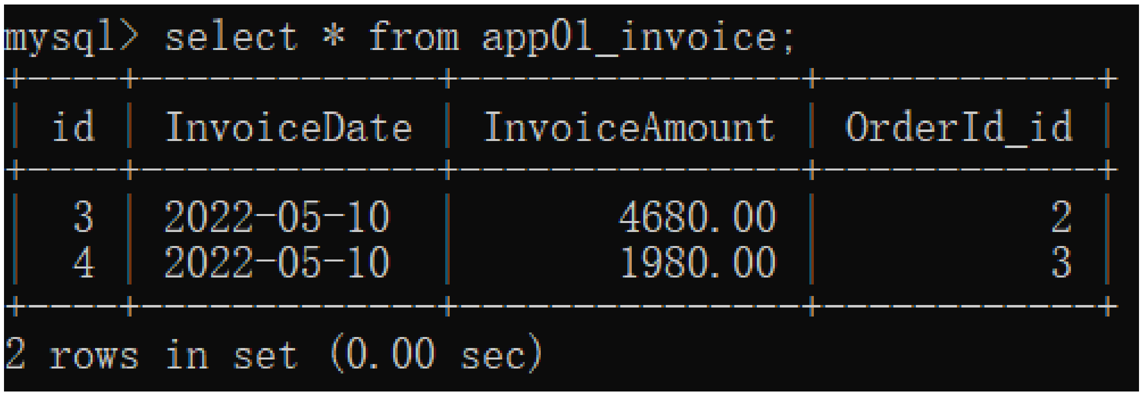
```
mysql> select * from app01_corporationuser;
+----+--------+----------+---------+----------------+------------+------------+------------------+------+
| id | street | city     | zipcode | email          | phone      | employ_id  | corporation_name | rate |
+----+--------+----------+---------+----------------+------------+------------+------------------+------+
|  2 | adawa  | 123123   | 98765   | pppp@1.com     | 987654321  | 1010101011 |                7 | 0.20 |
|  4 | 1111   | 23123123 | 23451   | 1121312@12.com | 258258258  | 1315181920 |                7 | 0.20 |
+----+--------+----------+---------+----------------+------------+------------+------------------+------+
2 rows in set (0.00 sec)
```

Figure 5: Corporation User Information



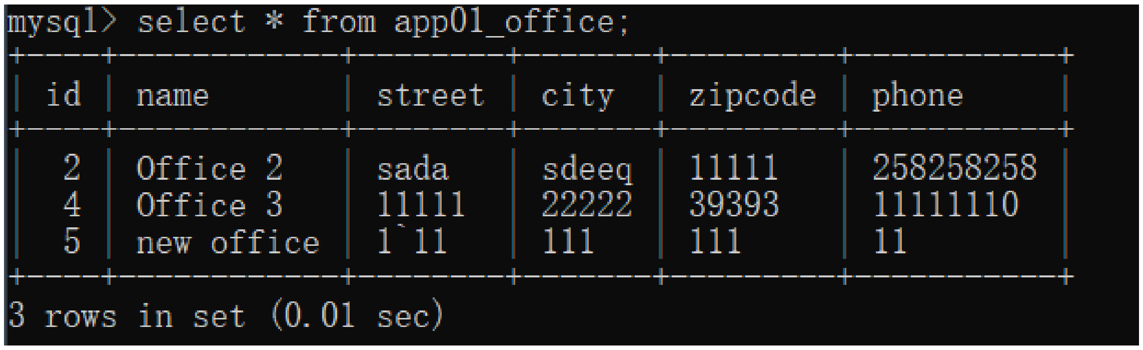
```
mysql> select * from app01_individualinfo;
+----+------------+----------+---------+-----------+-----------+-----------+------------+----------+-----------------
+-----------------+------+------------+-----------+----------------------------------+------------------+
| id | street          | city     | zipcode    | email       | phone        | FirstName | MiddleName | LastName | InsuranceCompany
 InsuranceNumber | rate | StartDate  | EndDate    | password                          | username         |
+----+------------+----------+---------+-----------+-----------+-----------+------------+----------+-----------------
+-----------------+------+------------+-----------+----------------------------------+------------------+
|  2 | 44 Bond St      | New York | 11201      | 11@nyu.edu  | 1000000000 | Prime     | K          | Li       | CDE
  2581234560      | 0.10 | 2021-01-11 | 2023-01-11 | 2022050413259412                 | 2022050413254880 |
|  3 | 123             | LA       | 12345      | 1111@111    | 234516811  | PP        | S          | DD       | PL
  2222222222      | 0.11 | 2022-02-15 | 2022-05-28 | 2022050413259412                 | 2022050413254880 |
|  4 | SAA             | PS       | 22222      | 2222        | 123456780  | SP        | K          | LLL      | dadw
  1598510230      | 0.20 | 2022-03-09 | 2023-02-16 | 2022050413259412                 | 2022050413254880 |
|  5 | 2222            | ssss     | 23456      | 111111@222  | 123432156  | asdwdq    | dasda      | eeeee    | ssssss
  1598510231      | 0.11 | 2022-04-12 | 2022-09-14 | 2022050413242136                 | 2022050413249449 |
|  7 | os              | os       | 12345      | os@od       | 123456123  | os        | os         | os       | po
  1598510235      | 0.10 | 2022-05-10 | 2022-05-24 | ea5e560034ee8e62a4d48ae3e0a71e14 | user             |
|  8 | 1               | 1        | 11111      | 11111       | 111111112  | 1         | 1          | 1        | 1
  1               | 0.10 | 2022-05-03 | 2022-05-17 | 2022051015017897                 | 2022051015014199 |
+----+------------+----------+---------+-----------+-----------+-----------+------------+----------+-----------------
+-----------------+------+------------+-----------+----------------------------------+------------------+
6 rows in set (0.00 sec)
```

Figure 6: Individual User Information



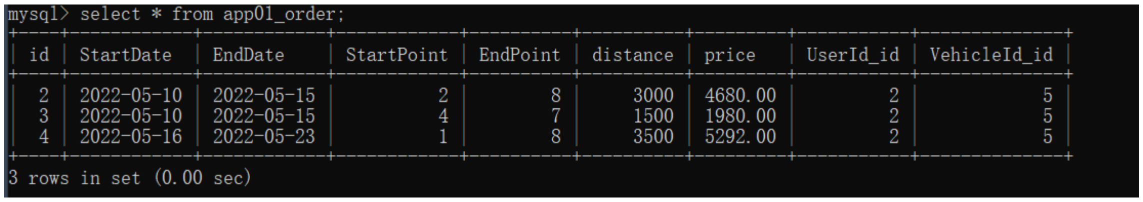
```
mysql> select * from app01_invoice;
+----+-------------+---------------+-------------+
| id | InvoiceDate | InvoiceAmount | OrderId_id  |
+----+-------------+---------------+-------------+
|  3 | 2022-05-10  |       4680.00 |           2 |
|  4 | 2022-05-10  |       1980.00 |           3 |
+----+-------------+---------------+-------------+
2 rows in set (0.00 sec)
```

Figure 7: Invoice Information



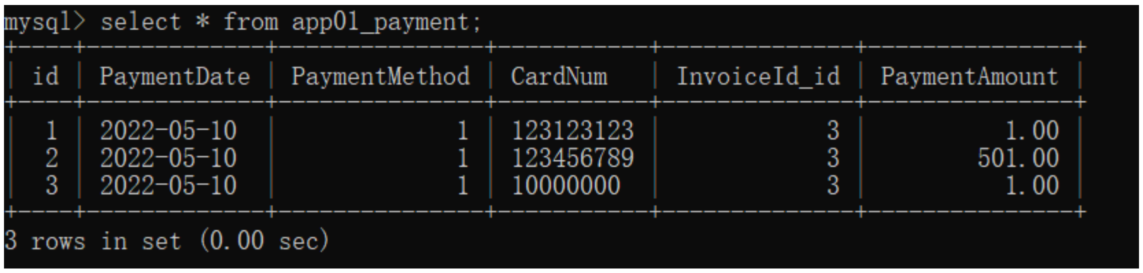
```
mysql> select * from app01_office;
+----+------------+--------+--------+----------+-----------+
| id | name       | street | city   | zipcode  | phone     |
+----+------------+--------+--------+----------+-----------+
|  2 | Office 2   | sada   | sdeeq  | 11111    | 258258258 |
|  4 | Office 3   | 11111  | 22222  | 39393    | 11111110  |
|  5 | new office | 1`11   | 111    | 111      | 11        |
+----+------------+--------+--------+----------+-----------+
3 rows in set (0.01 sec)
```

Figure 8: Office Information



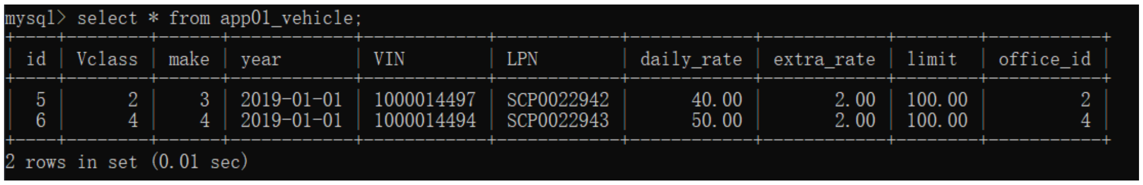
```
mysql> select * from app01_order;
+----+------------+------------+------------+----------+----------+---------+----------+--------------+
| id | StartDate  | EndDate    | StartPoint | EndPoint | distance | price   | UserId_id | VehicleId_id |
+----+------------+------------+------------+----------+----------+---------+----------+--------------+
|  2 | 2022-05-10 | 2022-05-15 |          2 |        8 |     3000 | 4680.00 |         2 |            5 |
|  3 | 2022-05-10 | 2022-05-15 |          4 |        7 |     1500 | 1980.00 |         2 |            5 |
|  4 | 2022-05-16 | 2022-05-23 |          1 |        8 |     3500 | 5292.00 |         2 |            5 |
+----+------------+------------+------------+----------+----------+---------+----------+--------------+
3 rows in set (0.00 sec)
```

Figure 9: Order Information



```
mysql> select * from app01_payment;
+----+-------------+---------------+-----------+------------+---------------+
| id | PaymentDate | PaymentMethod | CardNum   | InvoiceId_id | PaymentAmount |
+----+-------------+---------------+-----------+------------+---------------+
|  1 | 2022-05-10  |             1 | 123123123 |          3 |          1.00 |
|  2 | 2022-05-10  |             1 | 123456789 |          3 |        501.00 |
|  3 | 2022-05-10  |             1 | 10000000  |          3 |          1.00 |
+----+-------------+---------------+-----------+------------+---------------+
3 rows in set (0.00 sec)
```

Figure 10: Payment Information



```
mysql> select * from app01_vehicle;
+----+--------+------+------------+------------+------------+------------+------------+--------+-----------+
| id | Vclass | make | year       | VIN        | LPN        | daily_rate | extra_rate | limit  | office_id |
+----+--------+------+------------+------------+------------+------------+------------+--------+-----------+
|  5 |      2 |    3 | 2019-01-01 | 1000014497 | SCP0022942 |      40.00 |       2.00 | 100.00 |         2 |
|  6 |      4 |    4 | 2019-01-01 | 1000014494 | SCP0022943 |      50.00 |       2.00 | 100.00 |         4 |
+----+--------+------+------------+------------+------------+------------+------------+--------+-----------+
2 rows in set (0.01 sec)
```

Figure 11: Vehicle Information

5 DDL Code

```
--      Oracle      SQL Developer Data Modeler 21.4.1.349.1605
--         :         2022-04-08 19:07:52 EDT
--         :       Oracle Database 11g
--         :       Oracle Database 11g



-- predefined type, no DDL - MDSYS.SDO_GEOMETRY

-- predefined type, no DDL - XMLTYPE

CREATE TABLE qsz_class (
    class_id  NUMBER(2) NOT NULL,
    class_type VARCHAR2(50) NOT NULL,
    daily_rate NUMBER(6, 2) NOT NULL,
    extra_rate NUMBER(6, 2) NOT NULL
);

COMMENT ON COLUMN qsz_class.class_id IS
    'uniquely identify the car class';
```

```
COMMENT ON COLUMN qsz_class.class_type IS
    'various classes of vehicle such as small car, mid-size car,
        luxury car, SUV, Premium SUV, Mini Van, and Station Wagon
        ';

COMMENT ON COLUMN qsz_class.daily_rate IS
    'rental rate per day in odometer limit';

COMMENT ON COLUMN qsz_class.extra_rate IS
    'rental rate per day out of odometer limit';

ALTER TABLE qsz_class ADD CONSTRAINT qsz_class_pk PRIMARY KEY (
    class_id );

CREATE TABLE qsz_cor (
    cor_id  NUMBER(10) NOT NULL,
    cor_name VARCHAR2(50) NOT NULL,
    reg_num NUMBER(10) NOT NULL
);

COMMENT ON COLUMN qsz_cor.cor_id IS
    'uniquely identify the corporation';

COMMENT ON COLUMN qsz_cor.cor_name IS
    'Name of the corporation';

COMMENT ON COLUMN qsz_cor.reg_num IS
    'Registration number of the corporation';

ALTER TABLE qsz_cor ADD CONSTRAINT qsz_cor_pk PRIMARY KEY (
    cor_id );

CREATE TABLE qsz_cor_cou (
    ccou_id NUMBER(10) NOT NULL,
    crate  NUMBER(6, 2) NOT NULL,
    cust_id NUMBER(10) NOT NULL
);

COMMENT ON COLUMN qsz_cor_cou.ccou_id IS
    'uniquely identify the coupon of the corporate customer';

COMMENT ON COLUMN qsz_cor_cou.crate IS
    'rate of discount';

ALTER TABLE qsz_cor_cou ADD CONSTRAINT qsz_cor_cou_pk PRIMARY KEY
     ( ccou_id );
```

```
CREATE TABLE qsz_cor_ind (
    cust_id  NUMBER(10) NOT NULL,
    employ_id NUMBER(4) NOT NULL,
    cor_id   NUMBER(10) NOT NULL
);

COMMENT ON COLUMN qsz_cor_ind.employ_id IS
    'Employee ID of the customer who rents the car on a corporate
        account.';

ALTER TABLE qsz_cor_ind ADD CONSTRAINT qsz_cor_ind_pk PRIMARY KEY
     ( cust_id );

CREATE TABLE qsz_customer (
    cust_id    NUMBER(10) NOT NULL,
    cstreet    VARCHAR2(50) NOT NULL,
    ccity      VARCHAR2(50) NOT NULL,
    czipcode   NUMBER(6) NOT NULL,
    cemail     VARCHAR2(50) NOT NULL,
    cphone     NUMBER(10) NOT NULL,
    ccust_type CHAR(1) NOT NULL
);

ALTER TABLE qsz_customer
    ADD CONSTRAINT ch_inh_qsz_customer CHECK ( ccust_type IN ( 'C
        ', 'I', 'QSZ_CUSTOMER' ) );

COMMENT ON COLUMN qsz_customer.cust_id IS
    'uniquely identify the customer';

COMMENT ON COLUMN qsz_customer.cstreet IS
    'street address of the customer.';

COMMENT ON COLUMN qsz_customer.ccity IS
    'city address of the customer.';

COMMENT ON COLUMN qsz_customer.czipcode IS
    'zipcode of the customer.';

COMMENT ON COLUMN qsz_customer.cemail IS
    'email of the customer';

COMMENT ON COLUMN qsz_customer.cphone IS
    'email of the customer';
```

```
COMMENT ON COLUMN qsz_customer.ccust_type IS
    'customers of types Individual or Corporate.';

ALTER TABLE qsz_customer ADD CONSTRAINT qsz_customer_pk PRIMARY
    KEY ( cust_id );

CREATE TABLE qsz_ind_cou (
    icou_id    NUMBER(10) NOT NULL,
    irate      NUMBER(6, 2) NOT NULL,
    istart_date DATE NOT NULL,
    iend_date  DATE NOT NULL,
    cust_id    NUMBER(10) NOT NULL
);

COMMENT ON COLUMN qsz_ind_cou.icou_id IS
    'uniquely identify the coupon of the individual customer';

COMMENT ON COLUMN qsz_ind_cou.irate IS
    'discount rate';

COMMENT ON COLUMN qsz_ind_cou.istart_date IS
    'discount valid start date';

COMMENT ON COLUMN qsz_ind_cou.iend_date IS
    'discount valid end date';

ALTER TABLE qsz_ind_cou ADD CONSTRAINT qsz_ind_cou_pk PRIMARY KEY
     ( icou_id );

CREATE TABLE qsz_indiv (
    cust_id     NUMBER(10) NOT NULL,
    ilic_num    NUMBER(7) NOT NULL,
    ifirst_name VARCHAR2(50) NOT NULL,
    imiddle_name VARCHAR2(50),
    ilast_name  VARCHAR2(50) NOT NULL,
    insur_comp  VARCHAR2(50) NOT NULL,
    insur_num   NUMBER(10) NOT NULL
);

COMMENT ON COLUMN qsz_indiv.ilic_num IS
    'Driver License Number of the individual customer.';

COMMENT ON COLUMN qsz_indiv.insur_comp IS
    'Insurance Company Name ';

COMMENT ON COLUMN qsz_indiv.insur_num IS
```

```
    'and Insurance Policy Number.';

ALTER TABLE qsz_indiv ADD CONSTRAINT qsz_indiv_pk PRIMARY KEY (
    cust_id );

CREATE TABLE qsz_invoice (
    invoice_id     NUMBER(10) NOT NULL,
    invoice_date   DATE NOT NULL,
    invoice_amount NUMBER(6, 2) NOT NULL,
    order_id       NUMBER(10) NOT NULL
);

COMMENT ON COLUMN qsz_invoice.invoice_id IS
    'uniquely identify the invoice';

COMMENT ON COLUMN qsz_invoice.invoice_date IS
    'date of the invoice';

COMMENT ON COLUMN qsz_invoice.invoice_amount IS
    'amount of the invoice';

ALTER TABLE qsz_invoice ADD CONSTRAINT qsz_invoice_pk PRIMARY KEY
     ( invoice_id );

CREATE TABLE qsz_office (
    office_id NUMBER(10) NOT NULL,
    street    VARCHAR2(30) NOT NULL,
    city      VARCHAR2(50) NOT NULL,
    zipcode   NUMBER(6) NOT NULL,
    phone_num NUMBER(10) NOT NULL
);

COMMENT ON COLUMN qsz_office.office_id IS
    'uniquely identify a office';

COMMENT ON COLUMN qsz_office.street IS
    'street address of the office';

COMMENT ON COLUMN qsz_office.city IS
    'city address of the office';

COMMENT ON COLUMN qsz_office.zipcode IS
    'zip code of the office';

COMMENT ON COLUMN qsz_office.phone_num IS
    'phone number of the office';
```

```
ALTER TABLE qsz_office ADD CONSTRAINT qsz_office_pk PRIMARY KEY (
     office_id );

CREATE TABLE qsz_order (
    order_id     NUMBER(10) NOT NULL,
    opickup_id   NUMBER(10) NOT NULL,
    odropoff_id  NUMBER(10) NOT NULL,
    opickup_date DATE NOT NULL,
    odropoff_date DATE NOT NULL,
    ostart_odo   NUMBER(7, 2) NOT NULL,
    oend_odo     NUMBER(7, 2) NOT NULL,
    olimit       NUMBER(3) NOT NULL,
    vid          NUMBER(10) NOT NULL,
    cust_id      NUMBER(10) NOT NULL
);

COMMENT ON COLUMN qsz_order.order_id IS
    'uniquely identify the order';

ALTER TABLE qsz_order ADD CONSTRAINT qsz_order_pk PRIMARY KEY (
    order_id );

CREATE TABLE qsz_payment (
    ppayment_id   NUMBER(10) NOT NULL,
    pmethod       NVARCHAR2(30) NOT NULL,
    ppayment_date DATE NOT NULL,
    pcard_number  NUMBER(10) NOT NULL,
    ptotal_payment NUMBER(10, 2) NOT NULL,
    invoice_id    NUMBER(10) NOT NULL
);

COMMENT ON COLUMN qsz_payment.ppayment_id IS
    'uniquely identify the payment id ';

COMMENT ON COLUMN qsz_payment.pmethod IS
    'payment method redit/debit/gift card';

COMMENT ON COLUMN qsz_payment.ppayment_date IS
    'payment date';

COMMENT ON COLUMN qsz_payment.pcard_number IS
    'payment card number';

COMMENT ON COLUMN qsz_payment.ptotal_payment IS
    'total amount of payment';
```

```
ALTER TABLE qsz_payment ADD CONSTRAINT qsz_payment_pk PRIMARY KEY
     ( ppayment_id );

CREATE TABLE qsz_vehicle (
    vid       NUMBER(10) NOT NULL,
    make      VARCHAR2(20) NOT NULL,
    model     VARCHAR2(20) NOT NULL,
    year      DATE NOT NULL,
    vin       CHAR(10) NOT NULL,
    lpn       CHAR(10) NOT NULL,
    office_id NUMBER(10) NOT NULL,
    class_id NUMBER(2) NOT NULL
);

COMMENT ON COLUMN qsz_vehicle.vid IS
    'uniquely identify the car ';

COMMENT ON COLUMN qsz_vehicle.model IS
    'model of the car';

COMMENT ON COLUMN qsz_vehicle.year IS
    'year of the car';

COMMENT ON COLUMN qsz_vehicle.vin IS
    'Vehicle Identification Number';

COMMENT ON COLUMN qsz_vehicle.lpn IS
    'License Plate number';

ALTER TABLE qsz_vehicle ADD CONSTRAINT qsz_vehicle_pk PRIMARY KEY
     ( vid );

ALTER TABLE qsz_cor_cou
    ADD CONSTRAINT qsz_cor_cou_qsz_cor_ind_fk FOREIGN KEY (
        cust_id )
        REFERENCES qsz_cor_ind ( cust_id );

ALTER TABLE qsz_cor_ind
    ADD CONSTRAINT qsz_cor_ind_qsz_cor_fk FOREIGN KEY ( cor_id )
        REFERENCES qsz_cor ( cor_id );

ALTER TABLE qsz_cor_ind
    ADD CONSTRAINT qsz_cor_ind_qsz_customer_fk FOREIGN KEY (
        cust_id )
        REFERENCES qsz_customer ( cust_id );
```

```
ALTER TABLE qsz_ind_cou
    ADD CONSTRAINT qsz_ind_cou_qsz_indiv_fk FOREIGN KEY ( cust_id
        )
        REFERENCES qsz_indiv ( cust_id );

ALTER TABLE qsz_indiv
    ADD CONSTRAINT qsz_indiv_qsz_customer_fk FOREIGN KEY ( cust_id
         )
        REFERENCES qsz_customer ( cust_id );

ALTER TABLE qsz_invoice
    ADD CONSTRAINT qsz_invoice_qsz_order_fk FOREIGN KEY ( order_id
         )
        REFERENCES qsz_order ( order_id );

ALTER TABLE qsz_order
    ADD CONSTRAINT qsz_order_qsz_customer_fk FOREIGN KEY ( cust_id
         )
        REFERENCES qsz_customer ( cust_id );

ALTER TABLE qsz_order
    ADD CONSTRAINT qsz_order_qsz_vehicle_fk FOREIGN KEY ( vid )
        REFERENCES qsz_vehicle ( vid );

ALTER TABLE qsz_payment
    ADD CONSTRAINT qsz_payment_qsz_invoice_fk FOREIGN KEY (
        invoice_id )
        REFERENCES qsz_invoice ( invoice_id );

ALTER TABLE qsz_vehicle
    ADD CONSTRAINT qsz_vehicle_qsz_class_fk FOREIGN KEY ( class_id
         )
        REFERENCES qsz_class ( class_id );

ALTER TABLE qsz_vehicle
    ADD CONSTRAINT qsz_vehicle_qsz_office_fk FOREIGN KEY (
        office_id )
        REFERENCES qsz_office ( office_id );

CREATE OR REPLACE TRIGGER arc_fkarc_1_qsz_cor_ind BEFORE
    INSERT OR UPDATE OF cust_id ON qsz_cor_ind
    FOR EACH ROW
DECLARE
    d CHAR(1);
BEGIN
```

```
    SELECT
        a.ccust_type
    INTO d
    FROM
        qsz_customer a
    WHERE
        a.cust_id = :new.cust_id;

    IF ( d IS NULL OR d <> 'C' ) THEN
        raise_application_error(
                                -20223,
                                'FK QSZ_COR_IND_QSZ_CUSTOMER_FK in
                                    Table QSZ_COR_IND violates Arc
                                    constraint on Table QSZ_CUSTOMER -
                                     discriminator column CCUST_TYPE
                                    doesn''t have value ''C'''
        );
    END IF;

EXCEPTION
    WHEN no_data_found THEN
        NULL;
    WHEN OTHERS THEN
        RAISE;
END;
/

CREATE OR REPLACE TRIGGER arc_fkarc_1_qsz_indiv BEFORE
    INSERT OR UPDATE OF cust_id ON qsz_indiv
    FOR EACH ROW
DECLARE
    d CHAR(1);
BEGIN
    SELECT
        a.ccust_type
    INTO d
    FROM
        qsz_customer a
    WHERE
        a.cust_id = :new.cust_id;

    IF ( d IS NULL OR d <> 'I' ) THEN
        raise_application_error(
                                -20223,
                                'FK QSZ_INDIV_QSZ_CUSTOMER_FK in
                                    Table QSZ_INDIV violates Arc
```

```
                                  constraint on Table QSZ_CUSTOMER -
                                   discriminator column CCUST_TYPE
                                  doesn''t have value ''I'''
        );
    END IF;

EXCEPTION
    WHEN no_data_found THEN
        NULL;
    WHEN OTHERS THEN
        RAISE;
END;
/



-- Oracle SQL Developer Data Modeler        :
--
-- CREATE TABLE                           12
-- CREATE INDEX                            0
-- ALTER TABLE                            24
-- CREATE VIEW                             0
-- ALTER VIEW                              0
-- CREATE PACKAGE                          0
-- CREATE PACKAGE BODY                     0
-- CREATE PROCEDURE                        0
-- CREATE FUNCTION                         0
-- CREATE TRIGGER                          2
-- ALTER TRIGGER                           0
-- CREATE COLLECTION TYPE                  0
-- CREATE STRUCTURED TYPE                  0
-- CREATE STRUCTURED TYPE BODY             0
-- CREATE CLUSTER                          0
-- CREATE CONTEXT                          0
-- CREATE DATABASE                         0
-- CREATE DIMENSION                        0
-- CREATE DIRECTORY                        0
-- CREATE DISK GROUP                       0
-- CREATE ROLE                             0
-- CREATE ROLLBACK SEGMENT                 0
-- CREATE SEQUENCE                         0
-- CREATE MATERIALIZED VIEW                0
-- CREATE MATERIALIZED VIEW LOG            0
-- CREATE SYNONYM                          0
-- CREATE TABLESPACE                       0
-- CREATE USER                             0
```

```
--
-- DROP TABLESPACE                    0
-- DROP DATABASE                      0
--
-- REDACTION POLICY                   0
--
-- ORDS DROP SCHEMA                   0
-- ORDS ENABLE SCHEMA                 0
-- ORDS ENABLE OBJECT                 0
--
-- ERRORS                             0
-- WARNINGS                           0
```

6 DML Code

```
-- DML TO POPULATE DATA FOR QSZ_OFFICE
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000000','1 St', 'New York', '100000','1000000000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000001','2 St', 'Los Angeles',
    '100100','1000010000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000002','3 St', 'Chicago', '100200','1000020000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000003','4 St', 'Houston', '100300','1000030000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000004','5 St', 'Phoenix', '100400','1000050000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000005','5 St', 'Phoenix', '100400','1000050000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000006','6 St', 'Philadelphia',
    '100500','1000060000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000007','7 St', 'San Antonio',
    '100600','1000070000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000008','8 St', 'San Diego', '100700','1000080000');
```

```
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000009','9 St', 'Dallas', '100800','1000090000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('1000000010','10 St', 'San Jose', '100900','2000000000');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('2000000010','11 St', 'Austin', '200000','2000000001');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('2000000020','12 St', 'Jacksonville',
    '200001','2000000002');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('2000000030','13 St', 'Fort Worth',
    '200002','2000000003');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('2000000040','14 St', 'Columbus', '200003','2000000004');
insert into QSZ_OFFICE (OFFICE_ID, STREET, CITY, ZIPCODE,
    PHONE_NUM)
values ('2000000050','15 St', 'Indianapolis',
    '200004','2000000005');
COMMIT;


-- DML TO POPULATE DATA FOR QSZ_CLASS
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('01', 'small', 10, 10);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('02', 'mid-size', 15, 20);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('03', 'luxury', 18, 25);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('04', 'SUV', 20, 28);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('05', 'premium', 25, 28);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('06', 'premium suv', 25, 30);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('07', 'mini van', 15, 25);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('08', 'midium van', 20, 25);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('09', 'large van', 25, 20);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('10', 'station wagon', 30, 30);
insert into QSZ_CLASS( CLASS_ID, CLASS_TYPE, DAILY_RATE,
    EXTRA_RATE) values ('99', 'special', 40, 40);
COMMIT;




-- DML TO POPULATE DATA FOR QSZ_VEHICLE

insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000018796', 'Audi', 'A3', to_date('17-JUN-20','DD-MON-
    RR'),'1000001546', 'ACD0000045', '1000000000', '02');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000003465', 'Audi', 'A4', to_date('05-MAY-19','DD-MON-
    RR'),'1000013704', 'AEF0023750', '1000000000', '03');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000008513', 'Audi', 'A4 allroad', to_date('02-JAN-21','
    DD-MON-RR'),'1000005363', 'BCD0000162', '1000000000', '04');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000018249', 'Audi', 'A7', to_date('19-DEC-21','DD-MON-
    RR'),'1000000954', 'AGK0008292', '1000000000', '05');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000011991', 'GMC', 'Acadia', to_date('25-MAY-18','DD-
    MON-RR'),'1000028803', 'CEF0006847', '1000000000', '06');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000022454', 'Nissan', 'Altima', to_date('21-FEB-18','DD
    -MON-RR'),'1000021183', 'CJK0019663', '1000000000', '07');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000008107', 'Avalon', 'Toyota', to_date('11-APR-18','DD
    -MON-RR'),'1000025239', 'AKD1018189', '1000000000', '08');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000023906', 'Avalon Hybrid', 'Toyota', to_date('12-MAR
    -18','DD-MON-RR'),'1000026324', 'PKG0926029', '1000000000',
    '09');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000009609', 'Aviator', 'Lincoln', to_date('25-APR-18','
    DD-MON-RR'),'1000011659', 'PVG0008550', '1000000000', '10');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000008933', 'Bentayga', 'Bentley', to_date('23-NOV
    -18','DD-MON-RR'),'1000005348', 'SPC0007613', '1000000000',
    '99');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000009188', 'Blazer', 'Chevrolet', to_date('16-OCT
    -18','DD-MON-RR'),'1000014497', 'SCP0022942', '1000000000',
    '01');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000020187', 'Bolt EUV', 'Chevrolet', to_date('05-SEP
    -18','DD-MON-RR'),'1000032322', 'APG0027629', '1000000000',
    '02');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000010764', 'Bronco', 'Ford', to_date('13-SEP-18','DD-
    MON-RR'),'1000018985', 'AKG0012068', '1000000000', '03');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000022014', 'Cayenne Coupe', 'Porsche', to_date('21-DEC
    -18','DD-MON-RR'),'1000004901', 'QAL0030293', '1000000000',
    '04');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000027781', 'Cherokee', 'Jeep', to_date('22-NOV-18','DD
    -MON-RR'),'1000012948', 'KAL0014881', '1000000000', '05');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000022092', 'Plug-in Hybrid', 'Honda', to_date('14-OCT
    -18','DD-MON-RR'),'1000031074', 'LAG0016234', '1000000000',
    '06');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000010302', 'CT5', 'Cadillac', to_date('27-NOV-18','DD-
    MON-RR'),'1000012040', 'RPA0027213', '1000000000', '07');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000002199', 'CT6', 'Cadillac', to_date('11-JAN-18','DD-
    MON-RR'),'1000021756', 'GKA0004421', '1000000000', '08');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
```

```
values ('1000012384', 'CX-70', 'MAZDA', to_date('21-FEB-18','DD-
    MON-RR'),'1000020288', 'SKA0014619', '1000000000', '09');
insert into QSZ_VEHICLE( VID, MAKE, MODEL, YEAR, VIN, LPN,
    OFFICE_ID, CLASS_ID)
values ('1000024544', 'Mercedes-Benz', 'A-Class', to_date('17-JUN
    -21','DD-MON-RR'),'1000012892', 'ACD0018826', '1000000001',
    '01');



COMMIT;





-- DML TO POPULATE DATA FOR QSZ_CUSTOMER

insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000001','18 Densmore Dri','Essex','573895','
    UUuSb@gmail.com','9368835840','I');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000002','661 West Corint','Washington','960424','
    ZEuJl@gmail.com','9139868857','I');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000003','7717 Everett St','Arvada','702635','
    RraCz@gmail.com','9644206517','I');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000004','1622 Edgar D Ni','Montgomery','842166','
    fTHHV@gmail.com','9695020174','I');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000005','7564 Moore Cour','Arvada','562454','
    OBdyJ@gmail.com','9418976382','I');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000006','2611 Bluefield ','Nashville','285268','
    JDkAt@gmail.com','9746045388','I');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000007','6021 Yarrow Str','Nashville','819758','
    yASdp@gmail.com','9713225596','I');
```

```
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000008','4107 East 68th ','Anchorage','616060','
    EDBAc@gmail.com','9064013309','I');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000009','2721 Lindsay Av','Anchorage','779863','
    IJakg@gmail.com','9796059308','I');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000000','307 Joel Street','Manchester','272434','
    mdvzA@gmail.com','9509132656','I');

insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000011','80 South Main S','Manchester','877085','
    GthIO@gmail.com','8369334046','C');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000012','864 Main Street','Randolph','321617','
    cceLI@gmail.com','8736846239','C');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000013','1745 T Street S','Randolph','278380','
    DTPdb@gmail.com','8914953540','C');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000014','1008 Rhode Isla','Orange','660821','
    qiTvH@gmail.com','8641562209','C');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000015','700 Revels Driv','Orange','751670','
    QoAvO@gmail.com','8364732123','C');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000016','721 Bay Ridge A','Tewksbury','520706','
    eNZNj@gmail.com','8067943383','C');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000017','225 Kennedy Roa','Tewksbury','251309','
    kTQnl@gmail.com','8387930171','C');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000018','80 South Main S','Essex','325439','
    DEQIy@gmail.com','8959081943','C');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000019','3318 East Woodb','Essex','446422','
    KiCrg@gmail.com','8049480715','C');
insert into QSZ_CUSTOMER(CUST_ID, CSTREET, CCITY, CZIPCODE,
    CEMAIL, CPHONE, CCUST_TYPE)
values ('6000000010','41 Fairway Driv','Arvada','371990','
    wUAOa@gmail.com','8511417704','C');

COMMIT;




-- DML TO POPULATE DATA FOR QSZ_INDIV
insert into QSZ_CUSTOMER values (
QSZ_INDIV('6000000001','7777001','Rita','A','Mahurin','GEICOCC
    ','7777818542'));

insert into QSZ_CUSTOMER values (
QSZ_INDIV(CUST_ID, ILIC_NUM, IFIRST_NAME, IMIDDLE_NAME,
    ILAST_NAME, INSUR_COMP, INSUR_NUM)
values ('6000000002','7777002','Gerald','R','Roberts','GEICOCC
    ','7777475309');
insert into QSZ_CUSTOMER values (
QSZ_INDIV(CUST_ID, ILIC_NUM, IFIRST_NAME, IMIDDLE_NAME,
    ILAST_NAME, INSUR_COMP, INSUR_NUM)
values ('6000000003','7777003','Valerie','D','Keith','GEICOCC
    ','7777729112');
insert into QSZ_CUSTOMER values (
QSZ_INDIV(CUST_ID, ILIC_NUM, IFIRST_NAME, IMIDDLE_NAME,
    ILAST_NAME, INSUR_COMP, INSUR_NUM)
values ('6000000004','7777004','Mildred','C','Patrick','GEICOCC
    ','7777915308');
insert into QSZ_CUSTOMER values (
QSZ_INDIV(CUST_ID, ILIC_NUM, IFIRST_NAME, IMIDDLE_NAME,
    ILAST_NAME, INSUR_COMP, INSUR_NUM)
values ('6000000005','7777005','Gerald','R','Young','GEICOCC
    ','7777542254');
insert into QSZ_CUSTOMER values (
QSZ_INDIV(CUST_ID, ILIC_NUM, IFIRST_NAME, IMIDDLE_NAME,
    ILAST_NAME, INSUR_COMP, INSUR_NUM)
values ('6000000006','7777006','Michael','B','Mullins','GEICOCC
    ','7777262539');
insert into QSZ_CUSTOMER values (
QSZ_INDIV(CUST_ID, ILIC_NUM, IFIRST_NAME, IMIDDLE_NAME,
    ILAST_NAME, INSUR_COMP, INSUR_NUM)
```

```
values ('6000000007','7777007','Chad','M','Lamus','GEICOCC
    ','7777979520');
insert into QSZ_CUSTOMER values (
QSZ_INDIV(CUST_ID, ILIC_NUM, IFIRST_NAME, IMIDDLE_NAME,
    ILAST_NAME, INSUR_COMP, INSUR_NUM)
values ('6000000008','7777008','Tasha','V','Kaba','GEICOCC
    ','7777457648');
insert into QSZ_CUSTOMER values (
QSZ_INDIV(CUST_ID, ILIC_NUM, IFIRST_NAME, IMIDDLE_NAME,
    ILAST_NAME, INSUR_COMP, INSUR_NUM)
values ('6000000009','7777009','Timothy','O','Mendez','GEICOCC
    ','7777622031');
insert into QSZ_CUSTOMER values (
QSZ_INDIV(CUST_ID, ILIC_NUM, IFIRST_NAME, IMIDDLE_NAME,
    ILAST_NAME, INSUR_COMP, INSUR_NUM)
values ('6000000000','7777000','Joel','J','Goodson','GEICOCC
    ','7777342541');
COMMIT;

-- DML TO POPULATE DATA FOR QSZ_IND_COU

insert into QSZ_IND_COU(ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000001','0.05',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000001');

insert into QSZ_IND_COU (ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000002','0.01',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000002');

insert into QSZ_IND_COU (ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000003','0.10',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000003');

insert into QSZ_IND_COU (ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000004','0.15',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000004');

insert into QSZ_IND_COU (ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000005','0.05',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000005');
```

```
insert into QSZ_IND_COU (ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000006','0.05',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000006');

insert into QSZ_IND_COU (ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000007','0.05',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000007');

insert into QSZ_IND_COU (ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000008','0.05',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000008');

insert into QSZ_IND_COU (ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000009','0.05',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000009');

insert into QSZ_IND_COU (ICOU_ID, IRATE, ISTART_DATE, IEND_DATE,
    CUST_ID)
values ('8880000000','0.05',to_date('05-MAY-20','DD-MON-RR'),
    to_date('05-DEC-20','DD-MON-RR'),'6000000000');
COMMIT;

-- DML TO POPULATE DATA FOR QSZ_COR
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
values ('1234123412', 'Walmart','0000000000');
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
values ('1234123413', 'Amazon','0000000001');
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
values ('1234123414', 'Apple','0000000002');
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
values ('1234123415', 'CVS Health','0000000003');
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
values ('1234123416', 'UnitedHealth Group','0000000004');
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
values ('1234123417', 'McKesson','0000000005');
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
values ('1234123418', 'Alphabet','0000000006');
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
values ('1234123419', 'Exxon Mobil','0000000007');
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
values ('1234123420', 'AT&T','0000000008');
insert into QSZ_COR(COR_ID, COR_NAME, REG_NUM)
```

```
values ('1234123421', 'Costco','0000000009');
COMMIT;

-- DML TO POPULATE DATA FOR QSZ_COR_IND
insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000011','9540','1234123412');

insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000012','6255','1234123413');

insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000013','8070','1234123414');

insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000014','5395','1234123415');

insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000015','7892','1234123416');

insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000016','6732','1234123417');

insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000017','9308','1234123418');

insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000018','6007','1234123419');

insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000019','4625','1234123420');

insert into QSZ_CUSTOMER values (
QSZ_COR_IND('6000000010','5936','1234123421');
COMMIT;

-- DML TO POPULATE DATA FOR QSZ_COR_COU
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888897762','0.15','6000000001');
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888864261','0.15','6000000002');
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888889697','0.15','6000000003');
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888834957','0.10','6000000004');
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888836715','0.15','6000000005');
```

```
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888873040','0.15','6000000006');
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888892204','0.15','6000000007');
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888834640','0.20','6000000008');
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888864156','0.15','6000000009');
insert into QSZ_COR_COU(CCOU_ID, CRATE, CUST_ID)
values ('8888881699','0.10','6000000000');
COMMIT;
```

```
-- DML TO POPULATE DATA FOR QSZ_ORDER
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900001','4444686609','6866094444',to_date('05-MAY
    -20','DD-MON-RR'),to_date('05-JUN-20','DD-MON-RR')
    ,'10000.00','11627.00','100','1000018796','6000000001');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900002','4444741303','7413034444',to_date('04-JAN
    -20','DD-MON-RR'),to_date('04-FEB-20','DD-MON-RR')
    ,'10000.00','11303.00','100','1000003465','6000000002');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900003','4444356548','3565484444',to_date('03-FEB
    -20','DD-MON-RR'),to_date('03-MAR-20','DD-MON-RR')
    ,'10000.00','10223.00','100','1000008513','6000000003');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900004','4444311673','3116734444',to_date('02-MAR
    -20','DD-MON-RR'),to_date('02-APR-20','DD-MON-RR')
    ,'10000.00','10933.00','100','1000018249','6000000004');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900005','4444456817','4568174444',to_date('01-APR
    -20','DD-MON-RR'),to_date('01-MAY-20','DD-MON-RR')
    ,'10000.00','10269.00','100','1000011991','6000000005');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900006','4444211849','2118494444',to_date('05-MAY
    -20','DD-MON-RR'),to_date('05-JUN-20','DD-MON-RR')
    ,'10000.00','11609.00','100','1000022454','6000000006');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900007','4444237743','2377434444',to_date('05-MAY
    -20','DD-MON-RR'),to_date('05-JUN-20','DD-MON-RR')
    ,'10000.00','11140.00','100','1000008107','6000000007');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900008','4444979698','9796984444',to_date('12-AUG
    -20','DD-MON-RR'),to_date('12-SEP-20','DD-MON-RR')
    ,'10000.00','11358.00','100','1000023906','6000000008');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900009','4444461000','4610004444',to_date('04-MAY
    -20','DD-MON-RR'),to_date('04-JUN-20','DD-MON-RR')
    ,'10000.00','10366.00','100','1000009609','6000000009');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900000','4444252011','2520114444',to_date('03-JUN
    -20','DD-MON-RR'),to_date('03-JUL-20','DD-MON-RR')
    ,'10000.00','10761.00','100','1000008933','6000000000');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900011','4444365600','3656004444',to_date('05-JUL
    -20','DD-MON-RR'),to_date('05-AUG-20','DD-MON-RR')
    ,'10000.00','10683.00','100','1000009188','6000000011');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900012','4444943922','9439224444',to_date('03-AUG
    -20','DD-MON-RR'),to_date('03-SEP-20','DD-MON-RR')
    ,'10000.00','10092.00','100','1000020187','6000000012');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
```

```
values ('9090900013','4444746515','7465154444',to_date('02-MAY
    -20','DD-MON-RR'),to_date('02-JUN-20','DD-MON-RR')
    ,'10000.00','10215.00','100','1000010764','6000000013');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900014','4444844245','8442454444',to_date('05-SEP
    -20','DD-MON-RR'),to_date('05-OCT-20','DD-MON-RR')
    ,'10000.00','10947.00','100','1000022014','6000000014');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900015','4444584337','5843374444',to_date('03-JUL
    -20','DD-MON-RR'),to_date('03-AUG-20','DD-MON-RR')
    ,'10000.00','11707.00','100','1000027781','6000000015');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900016','4444732492','7324924444',to_date('04-MAY
    -20','DD-MON-RR'),to_date('04-JUN-20','DD-MON-RR')
    ,'10000.00','10255.00','100','1000022092','6000000016');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900017','4444241711','2417114444',to_date('11-AUG
    -20','DD-MON-RR'),to_date('11-SEP-20','DD-MON-RR')
    ,'10000.00','10502.00','100','1000010302','6000000017');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900018','4444411592','4115924444',to_date('06-SEP
    -20','DD-MON-RR'),to_date('06-OCT-20','DD-MON-RR')
    ,'10000.00','11840.00','100','1000002199','6000000018');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900019','4444958426','9584264444',to_date('10-DEC
    -20','DD-MON-RR'),to_date('10-JAN-21','DD-MON-RR')
    ,'10000.00','10326.00','100','1000012384','6000000019');
insert into QSZ_ORDER(ORDER_ID, OPICKUP_ID, ODROPOFF_ID,
    OPICKUP_DATE, ODROPOFF_DATE, OSTART_ODO, OEND_ODO, OLIMIT,
    VID, CUST_ID)
values ('9090900010','4444796984','7969844444',to_date('15-SEP
    -20','DD-MON-RR'),to_date('15-OCT-20','DD-MON-RR')
    ,'10000.00','10272.00','100','1000024544','6000000010');
COMMIT;
```

```
-- DML TO POPULATE DATA FOR QSZ_INVOICE
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911101',to_date('05-MAY-20','DD-MON-RR')
    ,'2341','9090900001');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911102',to_date('04-JAN-20','DD-MON-RR')
    ,'1980','9090900002');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911103',to_date('03-FEB-20','DD-MON-RR')
    ,'2381','9090900003');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911104',to_date('02-MAR-20','DD-MON-RR')
    ,'2637','9090900004');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911105',to_date('01-APR-20','DD-MON-RR')
    ,'2627','9090900005');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911106',to_date('05-MAY-20','DD-MON-RR')
    ,'1955','9090900006');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911107',to_date('05-MAY-20','DD-MON-RR')
    ,'2317','9090900007');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911108',to_date('12-AUG-20','DD-MON-RR')
    ,'2416','9090900008');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911109',to_date('04-MAY-20','DD-MON-RR')
    ,'2041','9090900009');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911100',to_date('03-JUN-20','DD-MON-RR')
    ,'2144','9090900000');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911111',to_date('05-JUL-20','DD-MON-RR')
    ,'2300','9090900011');
```

```
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911112',to_date('03-AUG-20','DD-MON-RR')
    ,'2128','9090900012');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911113',to_date('02-MAY-20','DD-MON-RR')
    ,'2655','9090900013');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911114',to_date('05-SEP-20','DD-MON-RR')
    ,'2274','9090900014');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911115',to_date('03-JUL-20','DD-MON-RR')
    ,'2516','9090900015');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911116',to_date('04-MAY-20','DD-MON-RR')
    ,'2239','9090900016');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911117',to_date('11-AUG-20','DD-MON-RR')
    ,'2112','9090900017');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911118',to_date('06-SEP-20','DD-MON-RR')
    ,'2089','9090900018');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911119',to_date('10-DEC-20','DD-MON-RR')
    ,'2422','9090900019');
insert into QSZ_INVOICE(INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT,
     ORDER_ID)
values ('9090911110',to_date('15-SEP-20','DD-MON-RR')
    ,'2297','9090900010');
COMMIT;

-- DML TO POPULATE DATA FOR QSZ_PAYMENT
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333693473','credit',to_date('05-MAY-20','DD-MON-RR')
    ,'1693935880','2341','9090911101');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333632374','giftcard',to_date('04-JAN-20','DD-MON-RR')
    ,'7098388408','1980','9090911102');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333649750','giftcard',to_date('03-FEB-20','DD-MON-RR')
    ,'1178107425','2381','9090911103');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333469581','giftcard',to_date('02-MAR-20','DD-MON-RR')
    ,'7396765923','2637','9090911104');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333541092','credit',to_date('01-APR-20','DD-MON-RR')
    ,'9548980724','2627','9090911105');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333805475','giftcard',to_date('05-MAY-20','DD-MON-RR')
    ,'8273038802','1955','9090911106');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333776865','giftcard',to_date('05-MAY-20','DD-MON-RR')
    ,'6877512417','2317','9090911107');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333351946','credit',to_date('12-AUG-20','DD-MON-RR')
    ,'1190749689','2416','9090911108');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333845511','giftcard',to_date('04-MAY-20','DD-MON-RR')
    ,'4853851575','2041','9090911109');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333458467','giftcard',to_date('03-JUN-20','DD-MON-RR')
    ,'4962098200','2144','9090911100');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333184943','credit',to_date('05-JUL-20','DD-MON-RR')
    ,'9147870476','2300','9090911111');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333524938','credit',to_date('03-AUG-20','DD-MON-RR')
    ,'9847043871','2128','9090911112');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333218854','credit',to_date('02-MAY-20','DD-MON-RR')
    ,'8989732109','2655','9090911113');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333540045','giftcard',to_date('05-SEP-20','DD-MON-RR')
    ,'4035112430','2274','9090911114');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333498029','giftcard',to_date('03-JUL-20','DD-MON-RR')
    ,'4477648085','2516','9090911115');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333615893','debit',to_date('04-MAY-20','DD-MON-RR')
    ,'9609516012','2239','9090911116');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333775105','debit',to_date('11-AUG-20','DD-MON-RR')
    ,'7339406210','2112','9090911117');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333465982','debit',to_date('06-SEP-20','DD-MON-RR')
    ,'7369425383','2089','9090911118');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333564340','debit',to_date('10-DEC-20','DD-MON-RR')
    ,'1223630188','2422','9090911119');
insert into QSZ_PAYMENT(PPAYMENT_ID, PMETHOD, PPAYMENT_DATE,
    PCARD_NUMBER, PTOTAL_PAYMENT, INVOICE_ID)
values ('2333611338','debit',to_date('15-SEP-20','DD-MON-RR')
    ,'5020891558','2297','9090911110');
COMMIT;
```

7 Basic Web Application

The login interface for admin:

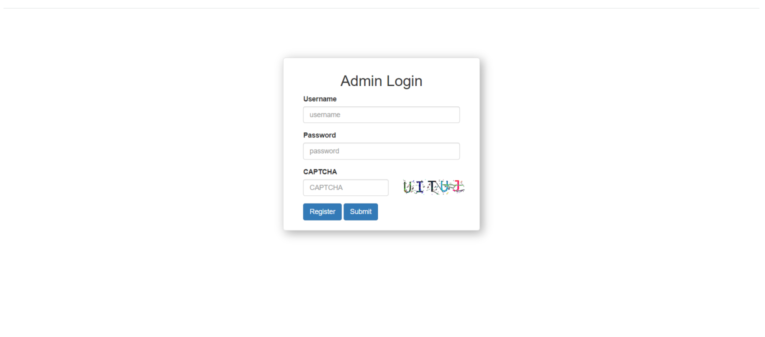


Figure 12: Admin Login

Register for admin:

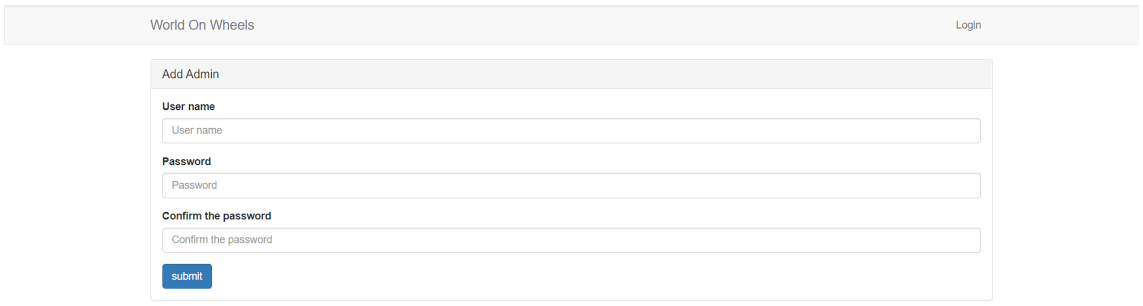
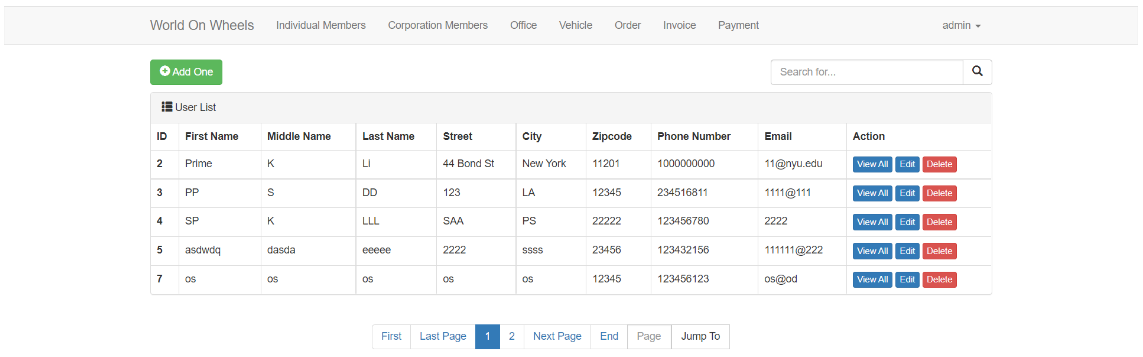


Figure 13: Admin Login

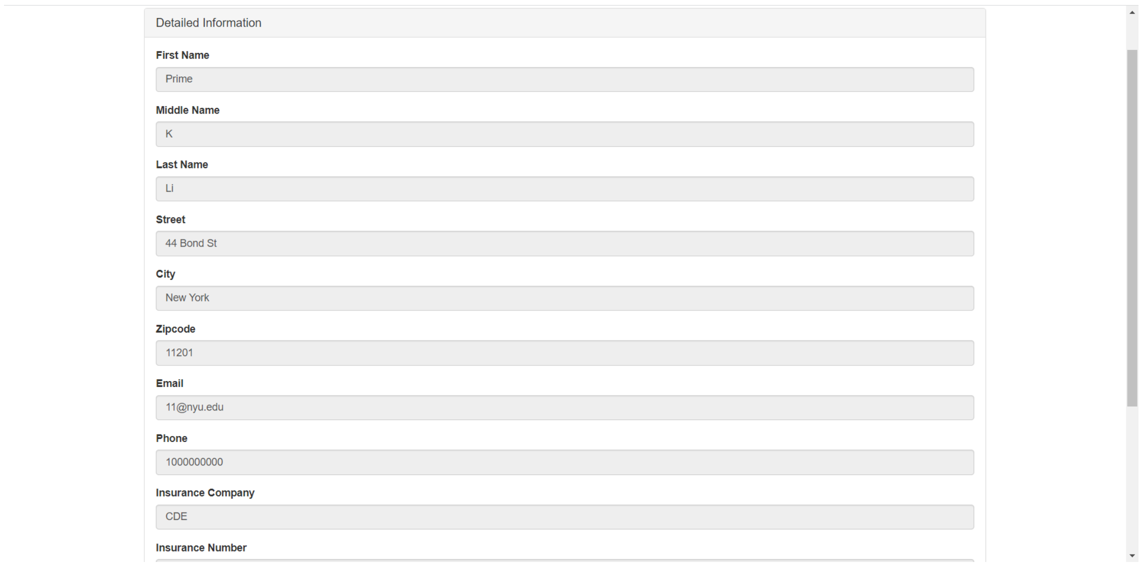
Individual Member in admin view:



ID	First Name	Middle Name	Last Name	Street	City	Zipcode	Phone Number	Email	Action
2	Prime	K	Li	44 Bond St	New York	11201	1000000000	11@nyu.edu	View All Edit Delete
3	PP	S	DD	123	LA	12345	234516811	1111@111	View All Edit Delete
4	SP	K	LLL	SAA	PS	22222	123456780	2222	View All Edit Delete
5	asdwdq	dasda	eeeee	2222	ssss	23456	123432156	111111@222	View All Edit Delete
7	os	os	os	os	os	12345	123456123	os@od	View All Edit Delete

Figure 14: Admin Individual

In the previous website, we only show the selected information for the individual member, we use the view all button to show all the information.


Detailed Information

First Name
Prime

Middle Name
K

Last Name
Li

Street
44 Bond St

City
New York

Zipcode
11201

Email
11@nyu.edu

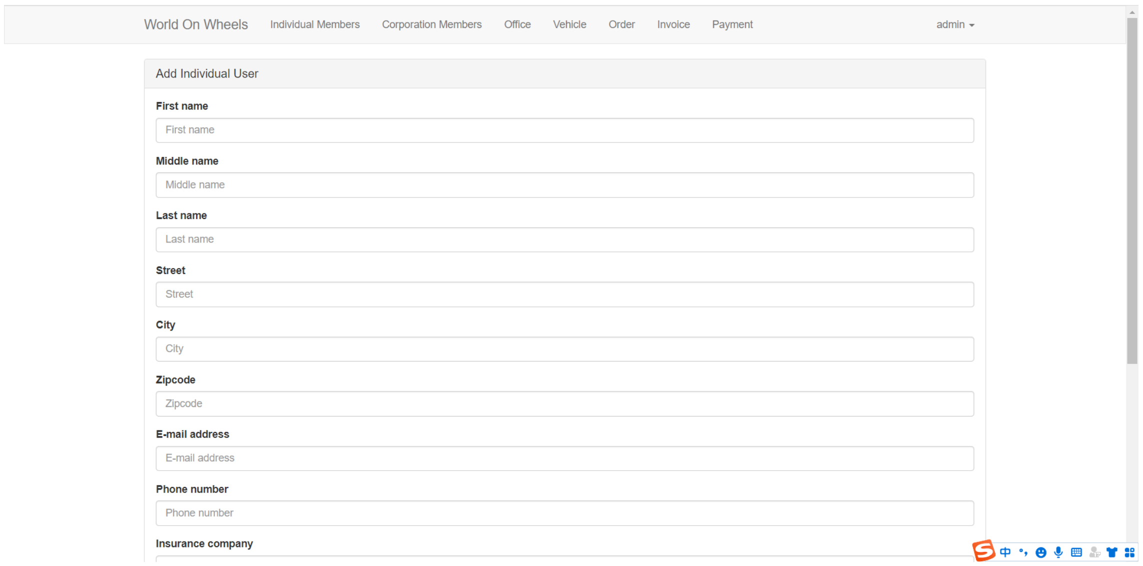
Phone
1000000000

Insurance Company
CDE

Insurance Number

Figure 15: View Detailed Information

A benefit provided by the admin user is that it can add members such as individual members and corporation members, one sample is that,



Add Individual User

First name
First name

Middle name
Middle name

Last name
Last name

Street
Street

City
City

Zipcode
Zipcode

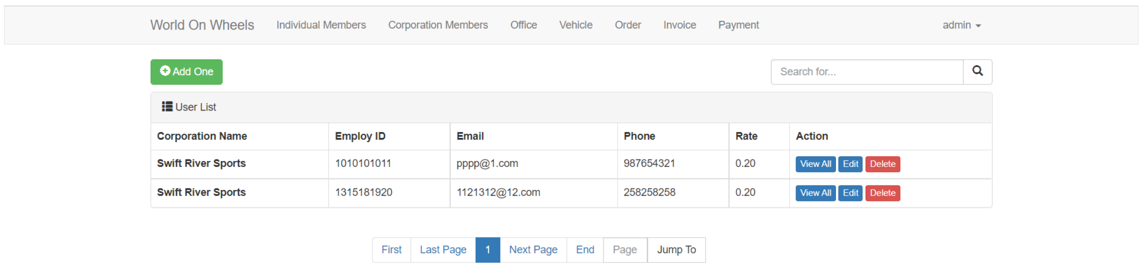
E-mail address
E-mail address

Phone number
Phone number

Insurance company

Figure 16: Add User

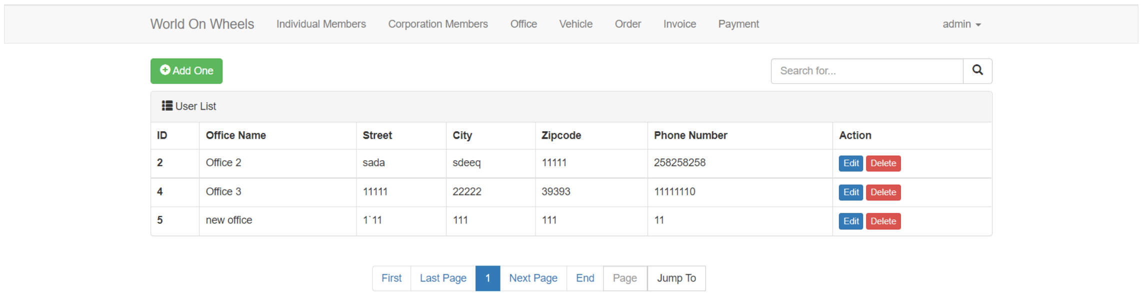
The following is the website view for the corporation members,



Corporation Name	Employ ID	Email	Phone	Rate	Action
Swift River Sports	1010101011	pppp@1.com	987654321	0.20	View All Edit Delete
Swift River Sports	1315181920	1121312@12.com	258258258	0.20	View All Edit Delete

Figure 17: Corporation User

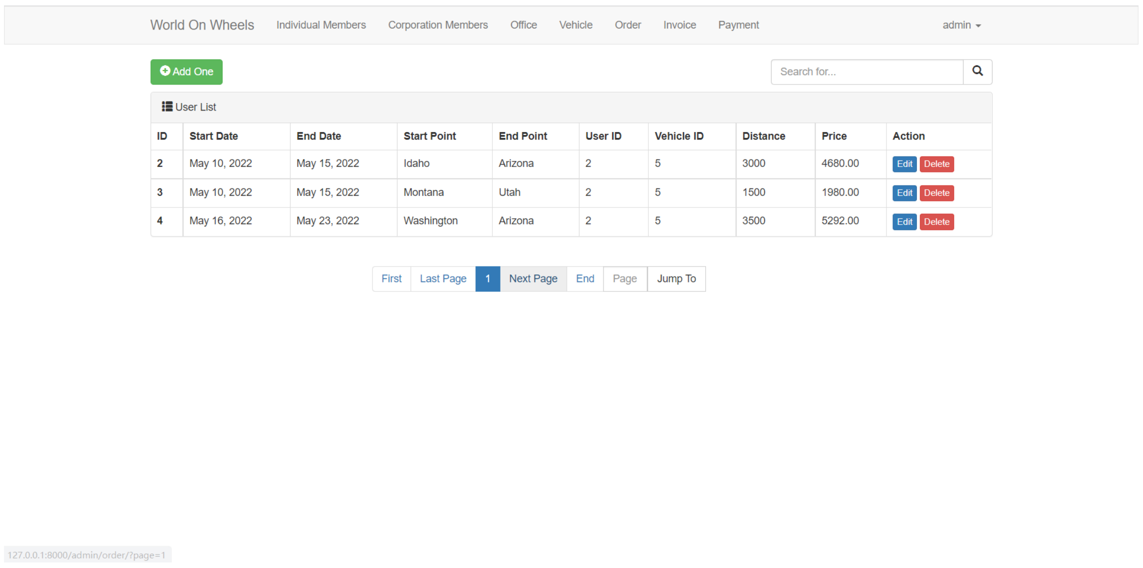
The following is the website view for the office,



ID	Office Name	Street	City	Zipcode	Phone Number	Action
2	Office 2	sada	sdeeq	11111	258258258	Edit Delete
4	Office 3	11111	22222	39393	11111110	Edit Delete
5	new office	1`11	111	111	11	Edit Delete

Figure 18: Office

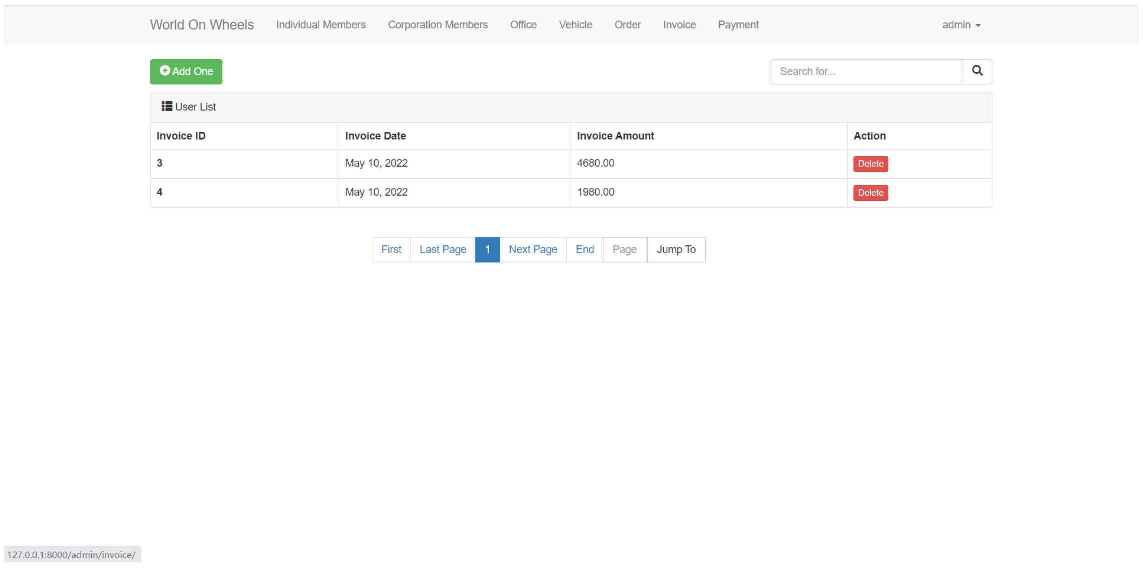
The following is the website view for the order,



ID	Start Date	End Date	Start Point	End Point	User ID	Vehicle ID	Distance	Price	Action
2	May 10, 2022	May 15, 2022	Idaho	Arizona	2	5	3000	4680.00	Edit Delete
3	May 10, 2022	May 15, 2022	Montana	Utah	2	5	1500	1980.00	Edit Delete
4	May 16, 2022	May 23, 2022	Washington	Arizona	2	5	3500	5292.00	Edit Delete

Figure 19: Order

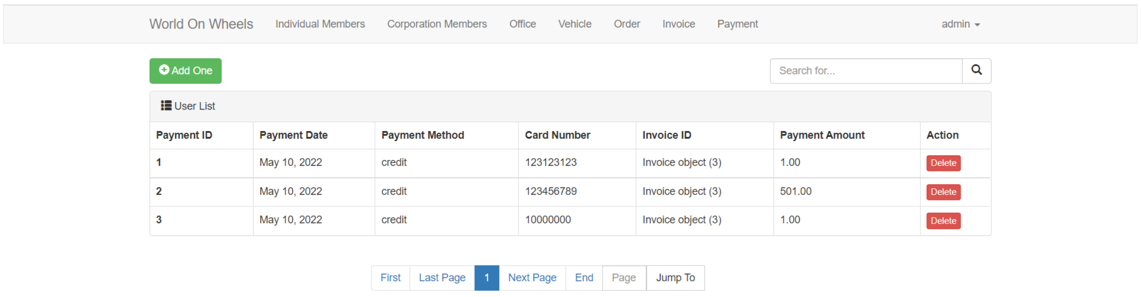
The following is the website view for the invoice,



Invoice ID	Invoice Date	Invoice Amount	Action
3	May 10, 2022	4680.00	Delete
4	May 10, 2022	1980.00	Delete

Figure 20: Invoice

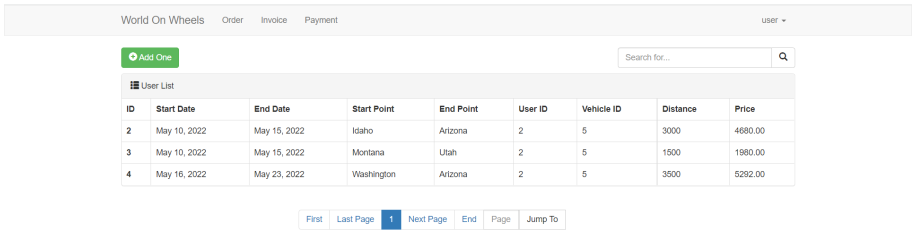
The following is the website view for the payment,



Payment ID	Payment Date	Payment Method	Card Number	Invoice ID	Payment Amount	Action
1	May 10, 2022	credit	123123123	Invoice object (3)	1.00	Delete
2	May 10, 2022	credit	123456789	Invoice object (3)	501.00	Delete
3	May 10, 2022	credit	10000000	Invoice object (3)	1.00	Delete

Figure 21: Payment

Next, we are going to show the system for the user end instead of the admin end. For the layout used for the user end, there will be fewer information, only order, invoice and payment will be included.



ID	Start Date	End Date	Start Point	End Point	User ID	Vehicle ID	Distance	Price
2	May 10, 2022	May 15, 2022	Idaho	Arizona	2	5	3000	4680.00
3	May 10, 2022	May 15, 2022	Montana	Utah	2	5	1500	1980.00
4	May 16, 2022	May 23, 2022	Washington	Arizona	2	5	3500	5292.00

Figure 22: Payment

8 User-friendly Function

To make our system more user-friendly, we define some function that can help user such as selecting time and calculate the total amount for the order.

8.1 Start Date and End Date

Here, we use the date picker to help user to select the date from the graph. Also, we set the start date to be before today and end date after today that is suitable for the inner logic.

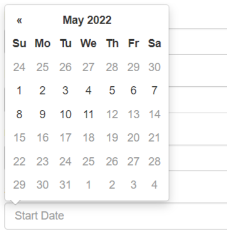


Figure 23: StartDate

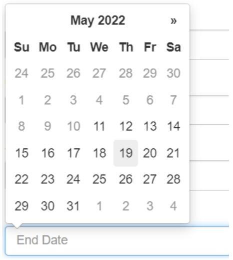
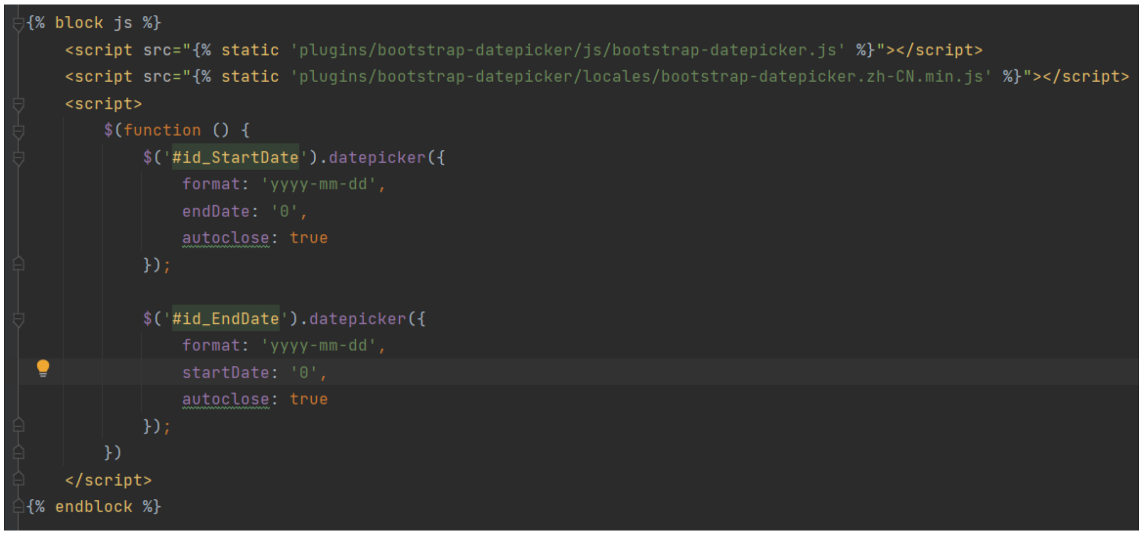


Figure 24: EndDate

I have also attached the corresponding code,

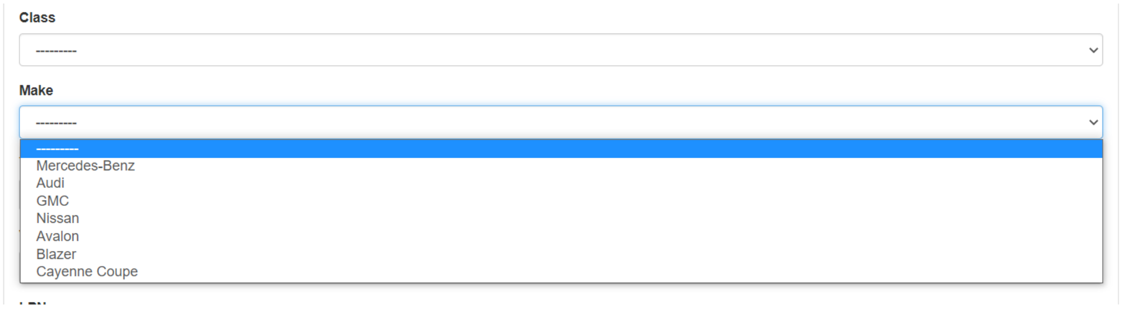


```
{% block js %}
    <script src="{% static 'plugins/bootstrap-datepicker/js/bootstrap-datepicker.js' %}"></script>
    <script src="{% static 'plugins/bootstrap-datepicker/locales/bootstrap-datepicker.zh-CN.min.js' %}"></script>
    <script>
        $(function () {
            $('#id_StartDate').datepicker({
                format: 'yyyy-mm-dd',
                endDate: '0',
                autoclose: true
            });

            $('#id_EndDate').datepicker({
                format: 'yyyy-mm-dd',
                startDate: '0',
                autoclose: true
            });
        })
    </script>
{% endblock %}
```

Figure 25: Date js code

8.2 Quick Selection

When typing in certain values, we are actually choosing among a few choices, instead of simply typing we, we can help users to do quick selection.



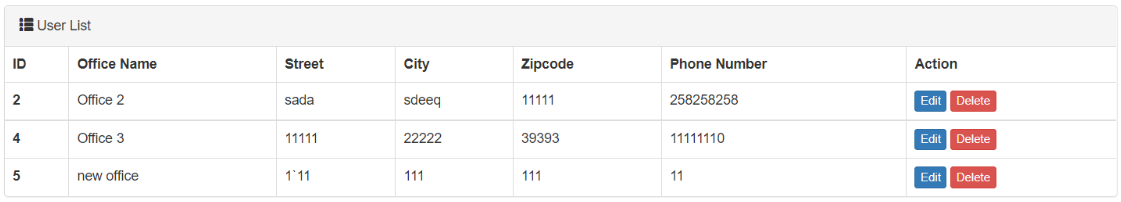
Class

Make

Mercedes-Benz
Audi
GMC
Nissan
Avalon
Blazer
Cayenne Coupe

Figure 26: Quick Selection

On the other hand, for example, if we add a new office called "new office", we will add this choice to the vehicle choices, this is implement by the the foreign key and choices in the django framework.


User List

ID	Office Name	Street	City	Zipcode	Phone Number	Action
2	Office 2	sada	sdeeq	11111	258258258	Edit Delete
4	Office 3	11111	22222	39393	11111110	Edit Delete
5	new office	1`11	111	111	11	Edit Delete

Figure 27: Adding "New Office"

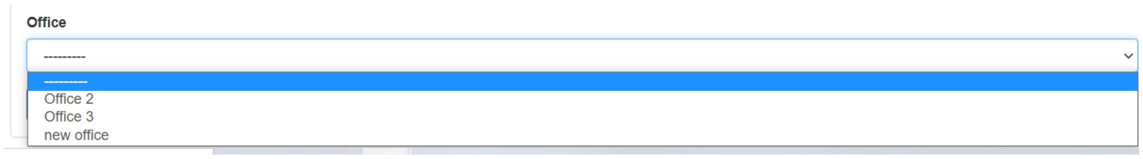


Figure 28: Vehicle Choice Among Office

The detailed code is here,

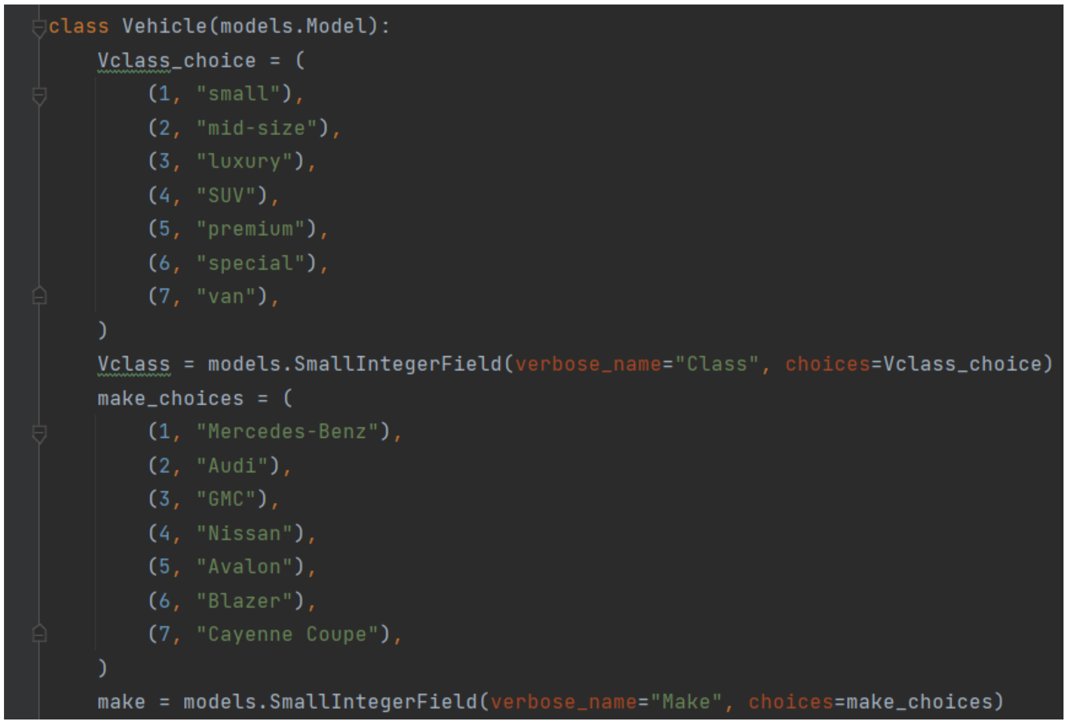
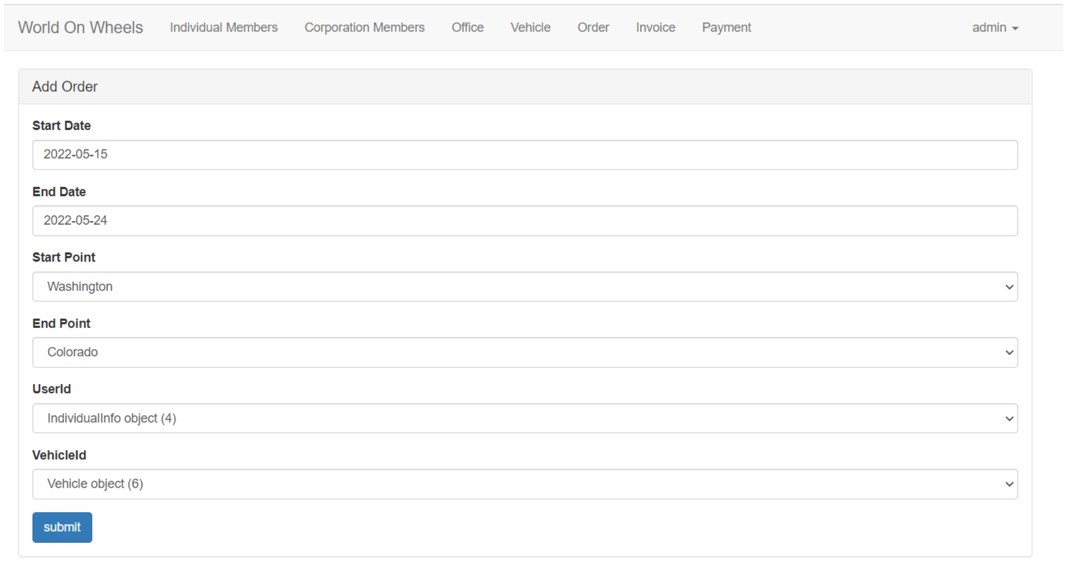

```
class Vehicle(models.Model):
    Vclass_choice = (
        (1, "small"),
        (2, "mid-size"),
        (3, "luxury"),
        (4, "SUV"),
        (5, "premium"),
        (6, "special"),
        (7, "van"),
    )
    Vclass = models.SmallIntegerField(verbose_name="Class", choices=Vclass_choice)
    make_choices = (
        (1, "Mercedes-Benz"),
        (2, "Audi"),
        (3, "GMC"),
        (4, "Nissan"),
        (5, "Avalon"),
        (6, "Blazer"),
        (7, "Cayenne Coupe"),
    )
    make = models.SmallIntegerField(verbose_name="Make", choices=make_choices)
```

Figure 29: Code for Choices

8.3 Order Automation

Here, I introduce a function that the customer only need to select the starting point and the ending point, it will calculate the distance between the two points. After the user select the vehicle, the database will calculate the order amount based on the daily rate, extra rate, limit from the vehicle and the coupon rate from the user, calculate the date difference between the end date and the start date.



Add Order

Start Date: 2022-05-15

End Date: 2022-05-24

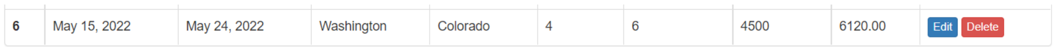
Start Point: Washington

End Point: Colorado

UserId: IndividualInfo object (4)

VehicleId: Vehicle object (6)

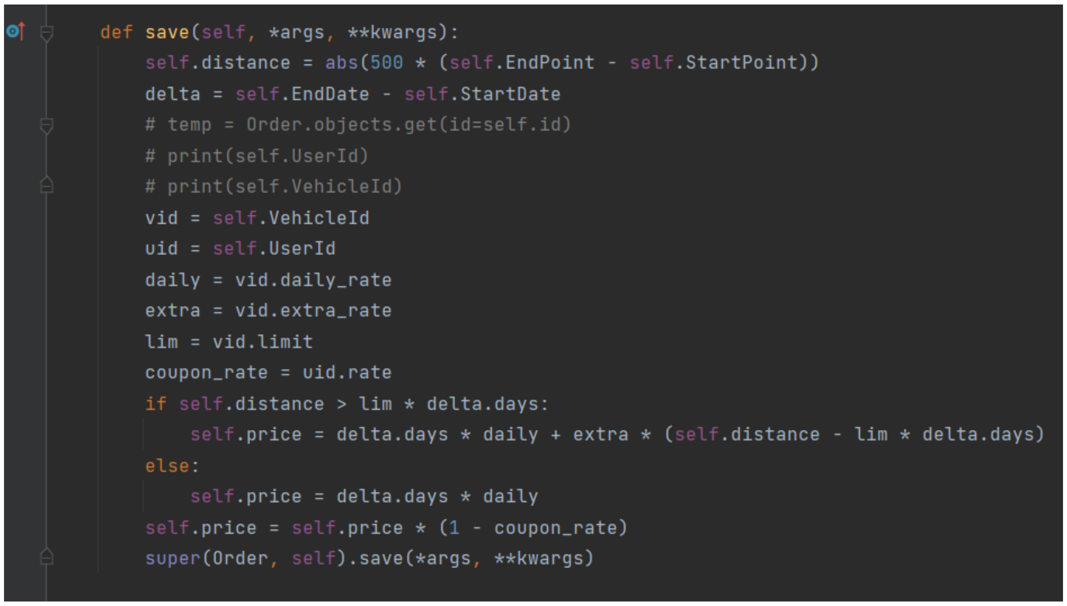
Figure 30: Generating a new order



6	May 15, 2022	May 24, 2022	Washington	Colorado	4	6	4500	6120.00	Edit Delete

Figure 31: Corresponding Order

The code that is dealing with the database is shown below, We overwrite the save function.



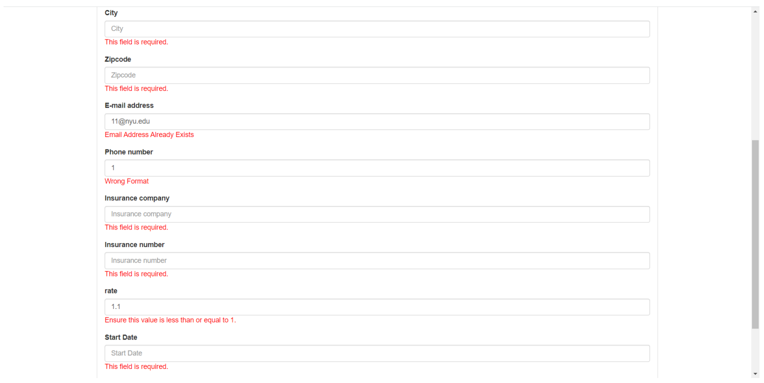
```python
def save(self, *args, **kwargs):
    self.distance = abs(500 * (self.EndPoint - self.StartPoint))
    delta = self.EndDate - self.StartDate
    # temp = Order.objects.get(id=self.id)
    # print(self.UserId)
    # print(self.VehicleId)
    vid = self.VehicleId
    uid = self.UserId
    daily = vid.daily_rate
    extra = vid.extra_rate
    lim = vid.limit
    coupon_rate = uid.rate
    if self.distance > lim * delta.days:
        self.price = delta.days * daily + extra * (self.distance - lim * delta.days)
    else:
        self.price = delta.days * daily
    self.price = self.price * (1 - coupon_rate)
    super(Order, self).save(*args, **kwargs)
```

Figure 32: Save Function Overwrite

9 Security Protections

9.1 Prevent SQL injection attacks

To prevent SQL injection attacks, for example, I built a validation function for user inputs to check whether the input is valid or not. Every data will have its own error message. For example, we will check whether the phone has the correct format, whether the email has been used before, whether the coupon rate is in the correct region and whether the information needed is filled.



City
City
This field is required.

Zipcode
Zipcode
This field is required.

E-mail address
11@nyu.edu
Email Address Already Exists

Phone number
1
Wrong Format

Insurance company
Insurance company
This field is required.

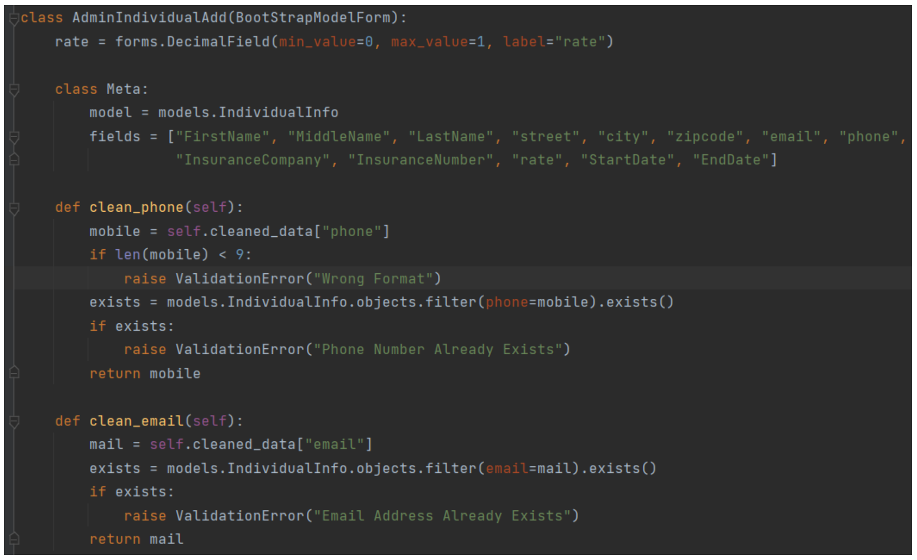
Insurance number
Insurance number
This field is required.

rate
1.1
Ensure this value is less than or equal to 1.

Start Date
Start Date
This field is required.

Figure 33: Error Message

For this part, the code is attached here.

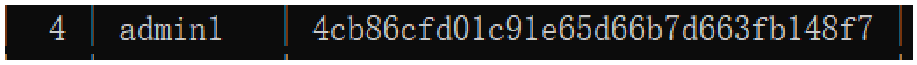


```
class AdminIndividualAdd(BootStrapModelForm):
    rate = forms.DecimalField(min_value=0, max_value=1, label="rate")

    class Meta:
        model = models.IndividualInfo
        fields = ["FirstName", "MiddleName", "LastName", "street", "city", "zipcode", "email", "phone",
                  "InsuranceCompany", "InsuranceNumber", "rate", "StartDate", "EndDate"]

    def clean_phone(self):
        mobile = self.cleaned_data["phone"]
        if len(mobile) < 9:
            raise ValidationError("Wrong Format")
        exists = models.IndividualInfo.objects.filter(phone=mobile).exists()
        if exists:
            raise ValidationError("Phone Number Already Exists")
        return mobile

    def clean_email(self):
        mail = self.cleaned_data["email"]
        exists = models.IndividualInfo.objects.filter(email=mail).exists()
        if exists:
            raise ValidationError("Email Address Already Exists")
        return mail
```

Figure 34: Error Code

9.2 Data Encryption

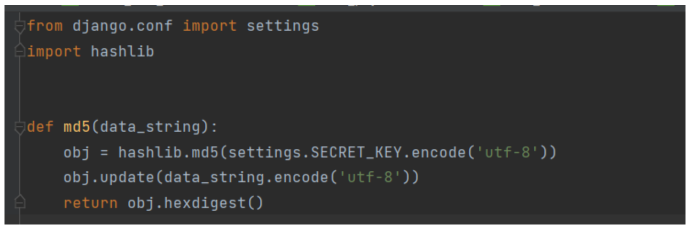
To protect the user information, we will use the md5 to encrypt the message.



```
4 | admin1 | 4cb86cfd01c91e65d66b7d663fb148f7
```

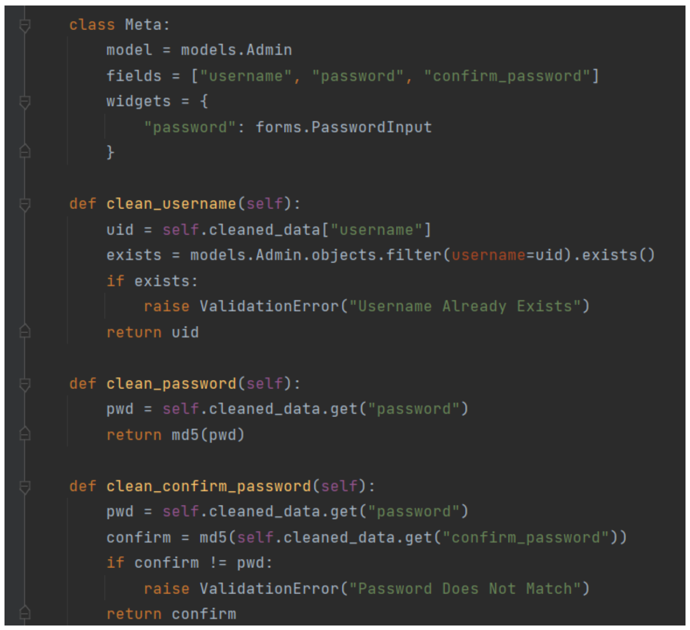
Figure 35: Encrypted Password

We use the encrypted method to encrypted the password and use the check function to see if the password matches.



```python
from django.conf import settings
import hashlib


def md5(data_string):
    obj = hashlib.md5(settings.SECRET_KEY.encode('utf-8'))
    obj.update(data_string.encode('utf-8'))
    return obj.hexdigest()
```

Figure 36: Encrypted Method

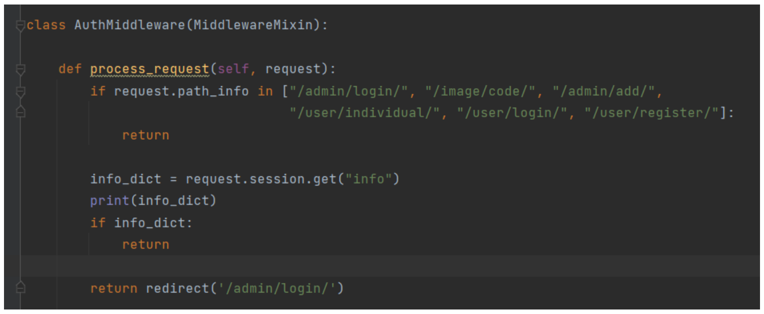


```python
    class Meta:
        model = models.Admin
        fields = ["username", "password", "confirm_password"]
        widgets = {
            "password": forms.PasswordInput
        }

    def clean_username(self):
        uid = self.cleaned_data["username"]
        exists = models.Admin.objects.filter(username=uid).exists()
        if exists:
            raise ValidationError("Username Already Exists")
        return uid

    def clean_password(self):
        pwd = self.cleaned_data.get("password")
        return md5(pwd)

    def clean_confirm_password(self):
        pwd = self.cleaned_data.get("password")
        confirm = md5(self.cleaned_data.get("confirm_password"))
        if confirm != pwd:
            raise ValidationError("Password Does Not Match")
        return confirm
```

Figure 37: Check Function

9.3 URL Protection Using MiddleWare

To help protect the security of the websites, I use the middle to constrain the websites urls that can be visited directly. For example, if someone want to visit the url 'admin/order/' without login, he will be redirected to the url 'admin/login/' to login. The way to realize this function is to use the middleware.

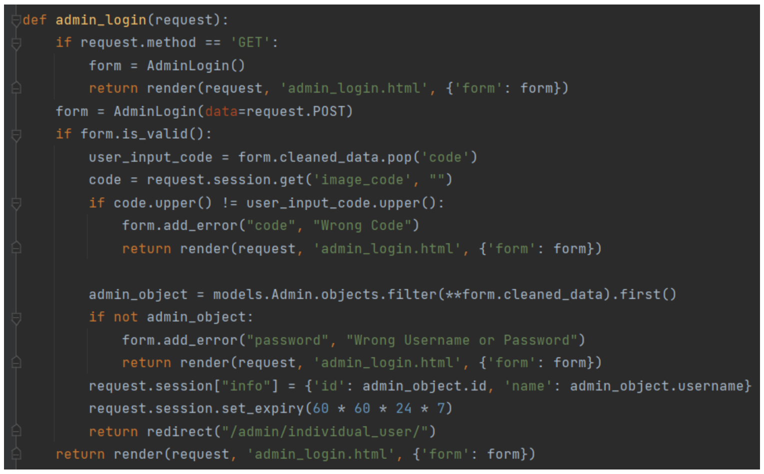


```python
class AuthMiddleware(MiddlewareMixin):

    def process_request(self, request):
        if request.path_info in ["/admin/login/", "/image/code/", "/admin/add/",
                                 "/user/individual/", "/user/login/", "/user/register/"]:
            return

        info_dict = request.session.get("info")
        print(info_dict)
        if info_dict:
            return

        return redirect('/admin/login/')
```

Figure 38: Middle Ware

9.4 Cookies and Sessions

On the other hand, considering that the same user will use the system repeatedly, after login, we use cookie to generate random string and session to remember the user information. Therefore, the same user do not need to login again. On the other hand, we set the expiry time to be 7 days.



```python
def admin_login(request):
    if request.method == 'GET':
        form = AdminLogin()
        return render(request, 'admin_login.html', {'form': form})
    form = AdminLogin(data=request.POST)
    if form.is_valid():
        user_input_code = form.cleaned_data.pop('code')
        code = request.session.get('image_code', "")
        if code.upper() != user_input_code.upper():
            form.add_error("code", "Wrong Code")
            return render(request, 'admin_login.html', {'form': form})

        admin_object = models.Admin.objects.filter(**form.cleaned_data).first()
        if not admin_object:
            form.add_error("password", "Wrong Username or Password")
            return render(request, 'admin_login.html', {'form': form})
        request.session["info"] = {'id': admin_object.id, 'name': admin_object.username}
        request.session.set_expiry(60 * 60 * 24 * 7)
        return redirect("/admin/individual_user/")
    return render(request, 'admin_login.html', {'form': form})
```

Figure 39: Cookie and Session

9.5 Captcha

I also introduced captcha to further ensure the safety of our whole system. Every time when we refresh the website, it will generate a random new captcha.

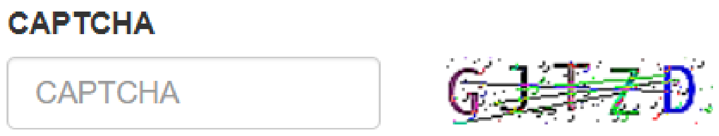
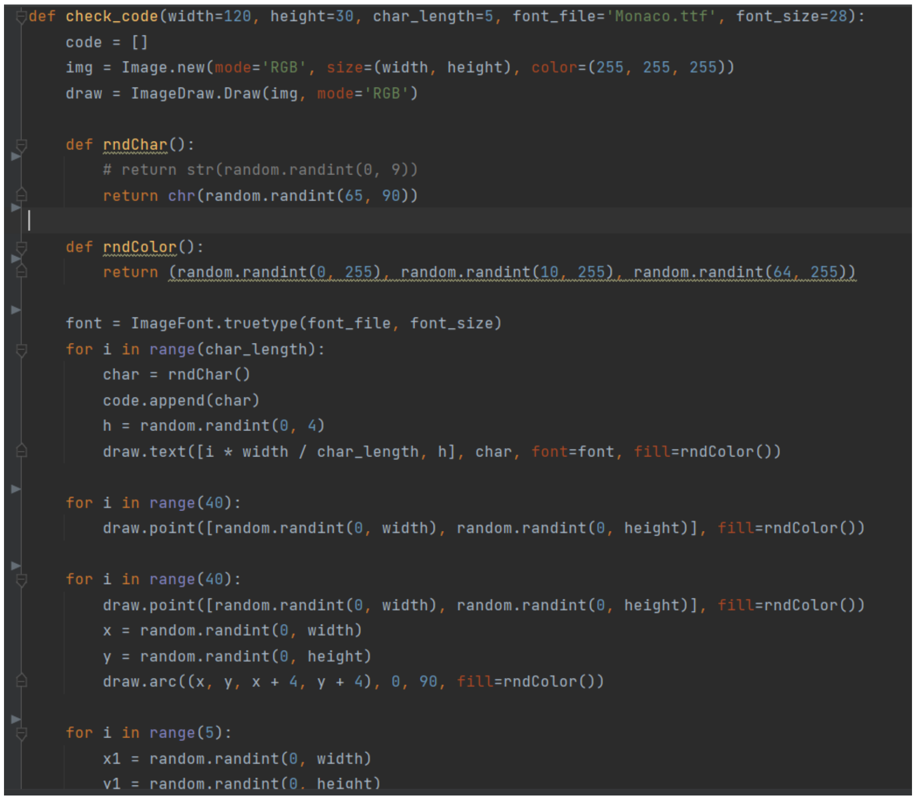


Figure 40: Captcha

The code is attached,



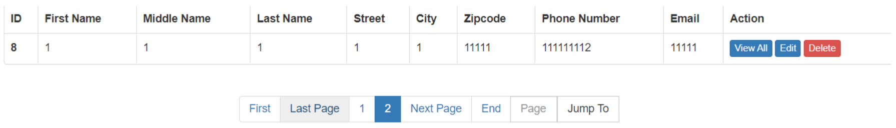
```python
def check_code(width=120, height=30, char_length=5, font_file='Monaco.ttf', font_size=28):
    code = []
    img = Image.new(mode='RGB', size=(width, height), color=(255, 255, 255))
    draw = ImageDraw.Draw(img, mode='RGB')

    def rndChar():
        # return str(random.randint(0, 9))
        return chr(random.randint(65, 90))

    def rndColor():
        return (random.randint(0, 255), random.randint(10, 255), random.randint(64, 255))

    font = ImageFont.truetype(font_file, font_size)
    for i in range(char_length):
        char = rndChar()
        code.append(char)
        h = random.randint(0, 4)
        draw.text([i * width / char_length, h], char, font=font, fill=rndColor())

    for i in range(40):
        draw.point([random.randint(0, width), random.randint(0, height)], fill=rndColor())

    for i in range(40):
        draw.point([random.randint(0, width), random.randint(0, height)], fill=rndColor())
        x = random.randint(0, width)
        y = random.randint(0, height)
        draw.arc((x, y, x + 4, y + 4), 0, 90, fill=rndColor())

    for i in range(5):
        x1 = random.randint(0, width)
        y1 = random.randint(0, height)
```

Figure 41: Captcha Code

10 More Self-Designed Tools

10.1 Pagination

I add the pagination on the website to show multiple data.



ID	First Name	Middle Name	Last Name	Street	City	Zipcode	Phone Number	Email	Action
8	1	1	1	1	1	11111	111111112	11111	View All Edit Delete

Figure 42: Pagination

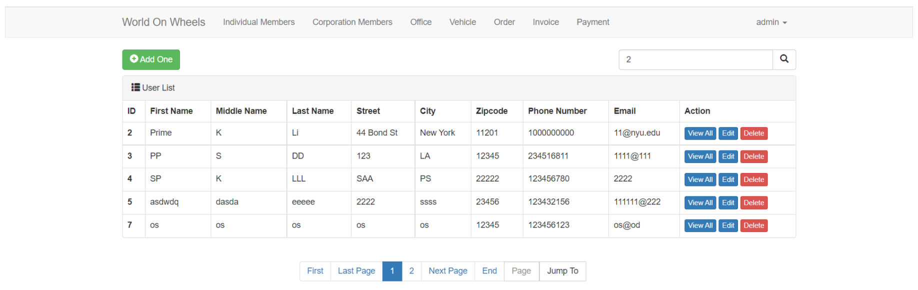
Code is attached,

```
"""
    def pretty_list(request):

        queryset = models.PrettyNum.objects.all()

        page_object = Pagination(request, queryset)

        context = {
            "queryset": page_object.page_queryset, # date
            "page_string": page_object.html()    # page size
        }
        return render(request, 'pretty_list.html', context)


In html file

    {% for obj in queryset %}
        {{obj.xx}}
    {% endfor %}

    <ul class="pagination">
        {{ page_string }}
    </ul>

"""

from django.utils.safestring import mark_safe


class Pagination(object):

    def __init__(self, request, queryset, page_size=10, page_param="page", plus=5
                                                ):
        """
        :param request:
        :param queryset:
        :param page_size:
        :param page_param: page size in the url
        :param plus: range for the page
        """
        from django.http.request import QueryDict
        import copy
        query_dict =copy.deepcopy(request.GET)
        query_dict._mutable =True
        self.query_dict =query_dict

        self.page_param =page_param
        page =request.GET.get(page_param, "1")

        if page.isdecimal():
            page =int(page)
        else:
            page =1

        self.page =page
        self.page_size =page_size

        self.start =(page -1) *page_size
        self.end =page *page_size

        self.page_queryset =queryset[self.start:self.end]

        total_count =queryset.count()
        total_page_count, div =divmod(total_count, page_size)
        if div:
            total_page_count +=1
        self.total_page_count =total_page_count
        self.plus =plus

    def html(self):
        if self.total_page_count <=2 *self.plus +1:
            start_page =1
            end_page =self.total_page_count
        else:
            if self.page <=self.plus:
                start_page =1
                end_page =2 * self.plus +1
            else:
                if (self.page +self.plus) >self.total_page_count:
                    start_page =self.total_page_count -2 *self.plus
                    end_page =self.total_page_count
                else:
                    start_page =self.page -self.plus
                    end_page =self.page +self.plus

        page_str_list =[]

        self.query_dict.setlist(self.page_param, [1])
        page_str_list.append('<li><a href="?{}">First</a></li>'.format(self.
                                                query_dict.urlencode()))

        if self.page >1:
            self.query_dict.setlist(self.page_param, [self.page -1])
            prev ='<li><a href="?{}">Last Page</a></li>'.format(self.query_dict.
                                                   urlencode())
        else:
```

```
        self.query_dict.setlist(self.page_param, [1])
        prev = '<li><a href="?{}">Last Page</a></li>'.format(self.query_dict.
                                              urlencode())
    page_str_list.append(prev)

    for i in range(start_page, end_page +1):
        self.query_dict.setlist(self.page_param, [i])
        if i ==self.page:
            ele = '<li class="active"><a href="?{}">{}</a></li>'.format(self.
                                                  query_dict.urlencode(), i)
        else:
            ele = '<li><a href="?{}">{}</a></li>'.format(self.query_dict.
                                                  urlencode(), i)
        page_str_list.append(ele)

    if self.page <self.total_page_count:
        self.query_dict.setlist(self.page_param, [self.page +1])
        prev = '<li><a href="?{}">Next Page</a></li>'.format(self.query_dict.
                                              urlencode())
    else:
        self.query_dict.setlist(self.page_param, [self.total_page_count])
        prev = '<li><a href="?{}">Next Page</a></li>'.format(self.query_dict.
                                              urlencode())
    page_str_list.append(prev)

    self.query_dict.setlist(self.page_param, [self.total_page_count])
    page_str_list.append('<li><a href="?{}">End</a></li>'.format(self.
                                          query_dict.urlencode()))

    search_string ="""
        <li>
            <form style="float: left;margin-left: -1px" method="get">
                <input name="page"
                       style="position: relative;float:left;display: inline-
                                                     block;width:
                                                     60px;border-
                                                     radius: 0;"
                       type="text" class="form-control" placeholder="Page">
                <button style="border-radius: 0" class="btn btn-default" type="
                                              submit">Jump To</
                                              button>
            </form>
        </li>
        """

    page_str_list.append(search_string)
    page_string =mark_safe("".join(page_str_list))
    return page_string
```

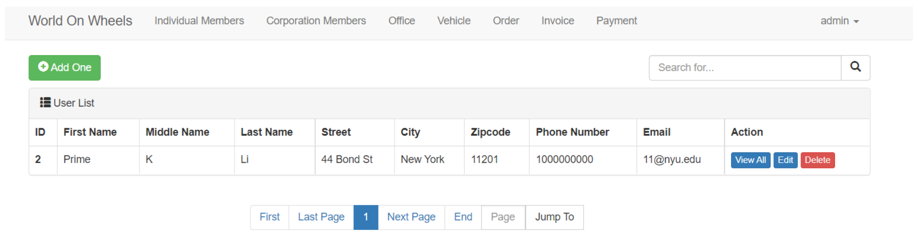
10.2 Search

On the other hand, for the admin user, if there exist too many data, it will be quite difficult to search for a specific data. Therefore, I add the search function so that we can find the data quickly based on the id.



ID	First Name	Middle Name	Last Name	Street	City	Zipcode	Phone Number	Email	Action
2	Prime	K	Li	44 Bond St	New York	11201	1000000000	11@nyu.edu	View All Edit Delete
3	PP	S	DD	123	LA	12345	234516811	1111@111	View All Edit Delete
4	SP	K	LLL	SAA	PS	22222	123456780	2222	View All Edit Delete
5	asdwdq	dasda	eeeee	2222	ssss	23456	123432156	111111@222	View All Edit Delete
7	os	os	os	os	os	12345	123456123	os@od	View All Edit Delete

Figure 43: Before Search



ID	First Name	Middle Name	Last Name	Street	City	Zipcode	Phone Number	Email	Action
2	Prime	K	Li	44 Bond St	New York	11201	1000000000	11@nyu.edu	View All Edit Delete

Figure 44: After Search

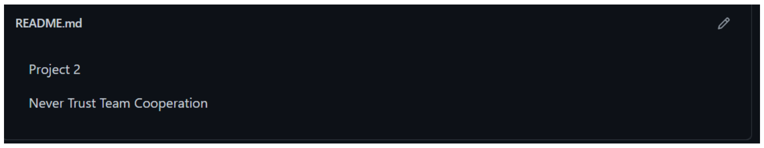
One of the example code,

```python
def admin_corporate(request):
    data_dict ={}
    search_data =request.GET.get('q', "")
    if search_data:
        data_dict["employ_id"] =search_data

    queryset =models.CorporationUser.objects.filter(**data_dict)
    page_object =Pagination(request, queryset, page_size=5)
    context ={
        "queryset": page_object.page_queryset,
        "page_string": page_object.html(),
    }
    return render(request, 'admin_corporate.html', context)
```

11 Harvest and Reflection

First of all, I would like to apologize for saying something unrelated to the technology stack, but doing this project did give me a bad experience. The main reason is that everything is done by MYSELF. My teammates did not do anything useful. I wrote all the codes myself, did the demo myself, preparing the PowerPoint myself. They did not write even A SINGLE LINE OF USEFUL CODE. I saw no reason why two graduate students count on others to do everything alone. Besides, I finished my part a week before the demo and asked them if they could finish the remaining part and they said yes. To avoid the tight deadline, I said that if they need any further assistance, they could inform me earlier. However, the accidents happened. They disappeared two days before the demo and appeared two minutes before the demo to ask me if I have prepared. How dare they.

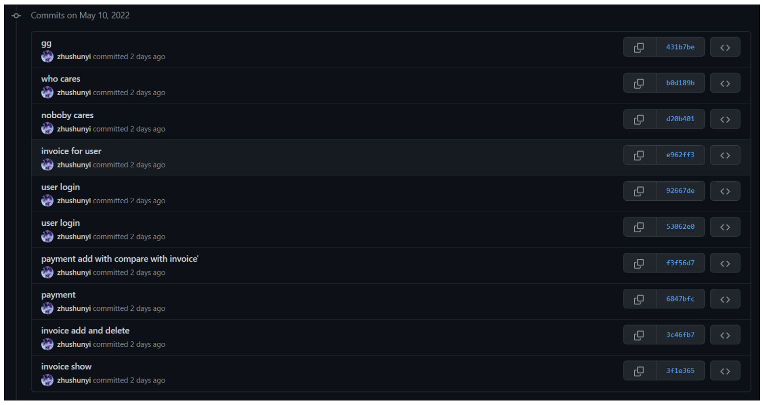
For the past two days, I only slept for half an hour and on the other hand, I was also suffering from the depression disorder and sometimes needs medical mental assistance. I felt myself struggling for these times but anyway, I have figured out anything on myself. I would not allow my teammates to take away my effort with seeming ease. I am asking if it is possible to grade individually for our group. I can provide any evidence if you want.


README.md

Project 2

Never Trust Team Cooperation

Figure 45: Read Me


Commits on May 10, 2022

gg
zhushunyi committed 2 days ago 431b7be

who cares
zhushunyi committed 2 days ago b0d189b

noboby cares
zhushunyi committed 2 days ago d20b401

invoice for user
zhushunyi committed 2 days ago e962ff3

user login
zhushunyi committed 2 days ago 92667de

user login
zhushunyi committed 2 days ago 53062e0

payment add with compare with invoice'
zhushunyi committed 2 days ago f3f56d7

payment
zhushunyi committed 2 days ago 6847bfc

invoice add and delete
zhushunyi committed 2 days ago 3c46fb7

invoice show
zhushunyi committed 2 days ago 3f1e365

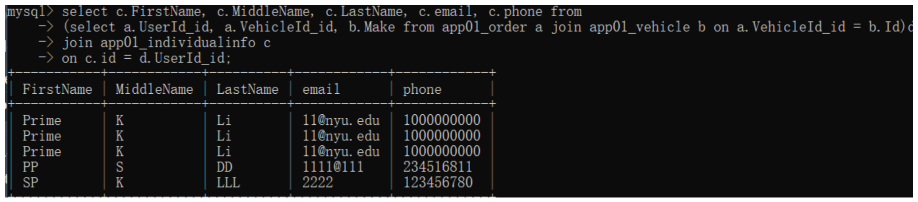
Figure 46: Commit History

Returning to technology, I really want to thank this project. For me, it is a practise with front-end and back-end. It is not only a database project but also a full-stack project. I learned a lot of tools from this project.

12 Business Analysis

12.1 Table joins with at least 3 tables in join

```
select c.FirstName, c.MiddleName, c.LastName, c.email, c.phone from
(select a.UserId_id, a.VehicleId_id, b.Make from app01_order a join app01_vehicle
                                          b on a.VehicleId_id =b.Id)d
join app01_individualinfo c
on c.id =d.UserId_id;
```



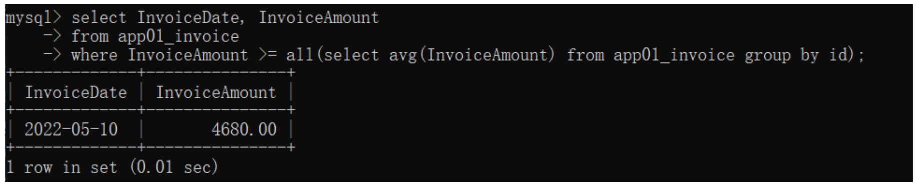
```
mysql> select c.FirstName, c.MiddleName, c.LastName, c.email, c.phone from
    -> (select a.UserId_id, a.VehicleId_id, b.Make from app01_order a join app01_vehicle b on a.VehicleId_id = b.Id)
    -> join app01_individualinfo c
    -> on c.id = d.UserId_id;
+-----------+------------+----------+-------------+------------+
| FirstName | MiddleName | LastName | email       | phone      |
+-----------+------------+----------+-------------+------------+
| Prime     | K          | Li       | 11@nyu.edu  | 1000000000 |
| Prime     | K          | Li       | 11@nyu.edu  | 1000000000 |
| Prime     | K          | Li       | 11@nyu.edu  | 1000000000 |
| PP        | S          | DD       | 1111@111    | 234516811  |
| SP        | K          | LLL      | 2222        | 123456780  |
```

Figure 47: Table Join

This query is used to show all the detailed information from the customers and the vehicle that has made an order.

12.2 Multi-row sub-query

```
select InvoiceDate, InvoiceAmount
from app01_invoice
where InvoiceAmount >=all(select avg(InvoiceAmount) from app01_invoice group by
                                         id);
```



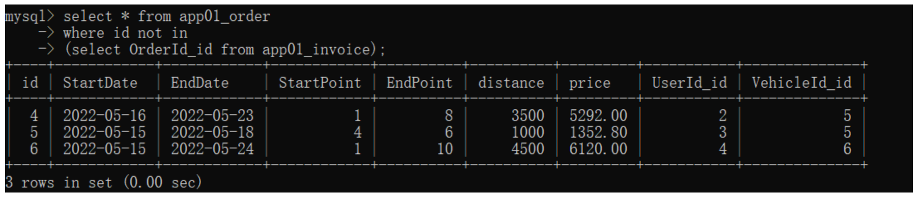
```
mysql> select InvoiceDate, InvoiceAmount
    -> from app01_invoice
    -> where InvoiceAmount >= all(select avg(InvoiceAmount) from app01_invoice group by id);
+-------------+---------------+
| InvoiceDate | InvoiceAmount |
+-------------+---------------+
| 2022-05-10  |       4680.00 |
+-------------+---------------+
1 row in set (0.01 sec)
```

Figure 48: Multi-row sub-query

This query is used to show the invoices whose amount is larger or equal to the highest average invoice amount of any specific customer.

12.3 Correlated subquery

```
select *from app01_order
where id not in
(select OrderId_id from app01_invoice);
```



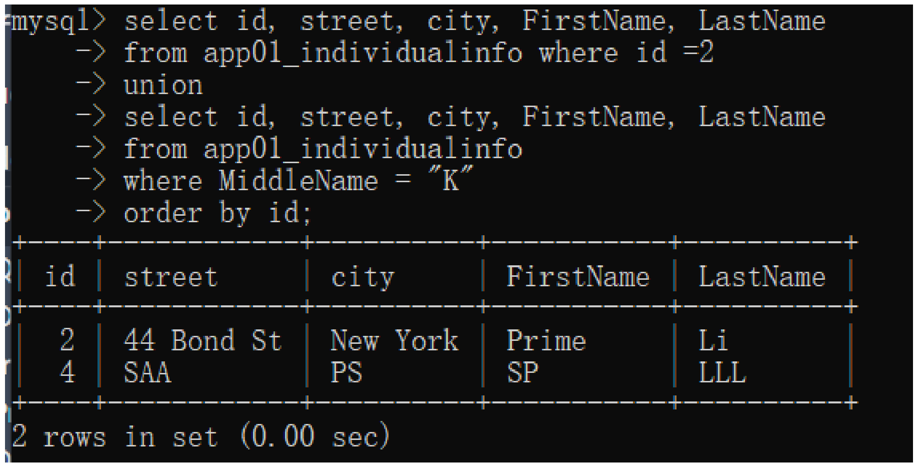
```
mysql> select * from app01_order
    -> where id not in
    -> (select OrderId_id from app01_invoice);
+----+------------+------------+------------+----------+----------+---------+-----------+--------------+
| id | StartDate  | EndDate    | StartPoint | EndPoint | distance | price   | UserId_id | VehicleId_id |
+----+------------+------------+------------+----------+----------+---------+-----------+--------------+
|  4 | 2022-05-16 | 2022-05-23 |          1 |        8 |     3500 | 5292.00 |         2 |            5 |
|  5 | 2022-05-15 | 2022-05-18 |          4 |        6 |     1000 | 1352.80 |         3 |            5 |
|  6 | 2022-05-15 | 2022-05-24 |          1 |       10 |     4500 | 6120.00 |         4 |            6 |
+----+------------+------------+------------+----------+----------+---------+-----------+--------------+
3 rows in set (0.00 sec)
```

Figure 49: Correlated subquery

This query is used to show the detailed information about orders that have not applied for the invoice in the database.

12.4 SET operator query

```
select id, street, city, FirstName, LastName
from app01_individualinfo where id =2
union
select id, street, city, FirstName, LastName
from app01_individualinfo
where MiddleName ="K"
order by id;
```



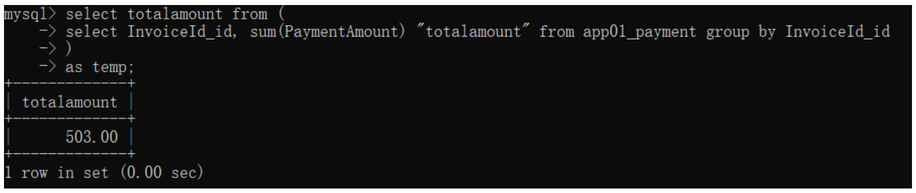
```
mysql> select id, street, city, FirstName, LastName
    -> from app01_individualinfo where id =2
    -> union
    -> select id, street, city, FirstName, LastName
    -> from app01_individualinfo
    -> where MiddleName = "K"
    -> order by id;
+----+------------+----------+-----------+----------+
| id | street     | city     | FirstName | LastName |
+----+------------+----------+-----------+----------+
|  2 | 44 Bond St | New York | Prime     | Li       |
|  4 | SAA        | PS       | SP        | LLL      |
+----+------------+----------+-----------+----------+
2 rows in set (0.00 sec)
```

Figure 50: SET operator query

This query is used to view the individual information with id is 2 and the individual with middle name as ”K” without duplicate record.

12.5 Query with in line view or WITH clause

```
select totalamount from (
select InvoiceId_id, sum(PaymentAmount) "totalamount" from app01_payment group by
                                           InvoiceId_id
)
as temp;
```



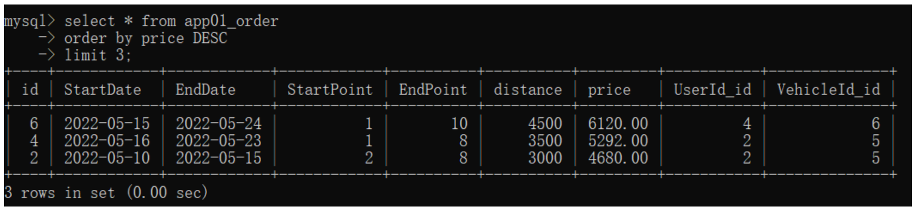
```
mysql> select totalamount from (
    -> select InvoiceId_id, sum(PaymentAmount) "totalamount" from app01_payment group by InvoiceId_id
    -> )
    -> as temp;
+-------------+
| totalamount |
+-------------+
|      503.00 |
+-------------+
1 row in set (0.00 sec)
```

Figure 51: Query with in line view

This query is used to view the total payment group by the invoice.

12.6 TOP-N Query

```
select *from app01_order
order by price DESC
limit 3;
```



```
mysql> select * from app01_order
    -> order by price DESC
    -> limit 3;
+----+------------+------------+------------+----------+----------+---------+------------+--------------+
| id | StartDate  | EndDate    | StartPoint | EndPoint | distance | price   | UserId_id  | VehicleId_id |
+----+------------+------------+------------+----------+----------+---------+------------+--------------+
|  6 | 2022-05-15 | 2022-05-24 |          1 |       10 |     4500 | 6120.00 |          4 |            6 |
|  4 | 2022-05-16 | 2022-05-23 |          1 |        8 |     3500 | 5292.00 |          2 |            5 |
|  2 | 2022-05-10 | 2022-05-15 |          2 |        8 |     3000 | 4680.00 |          2 |            5 |
+----+------------+------------+------------+----------+----------+---------+------------+--------------+
3 rows in set (0.00 sec)
```

Figure 52: Top-N Query

This query is used to view the top 3 order with amount price descending.