George He gyh226
Zhuo Chen zc2958
Wenxuan Hu wh2407
CS-GY 6083 Section B
11/30/23

Table of Contents

Summary

World on Wheels (WOW) is a rapidly growing car rental company with a widespread presence across airports, towns, and cities in the United States. In response to its expanding operations and the need for more efficient management, WOW has decided to transition from a file system-based data management approach to a sophisticated centralized database system. This is where my company comes in and assists WOW with their data management needs. The primary solution involves the creation of a centralized database system to replace the existing file system-based data management. This transition is crucial for streamlining operations, enhancing data accessibility, and improving overall business efficiency. Furthermore, a relational database model has been chosen for its ability to efficiently manage complex relationships between different entities within the business. The model includes tables for corporations, offices, customers, individuals, coupons, vehicles, rentals, invoices, payments, and other relevant entities.

While creating this model, there were a few key assumptions that were made. These assumptions include the following: for each rental service there is only one invoice. A customer can have multiple rental services. Each vehicle can be used in one or more rental services. A customer may or may not use a coupon to pay for a rental service. Each invoice must be paid in one payment. Each customer only has one address associated with them. Daily limit for a car rental is optional, customers may not go over the limit. Rental and office will have 2 relationships due to drop off and pick up location. Each office will have more than one vehicle available to rent. Each vehicle class has more than one vehicle.

This data management system will also help WOW improve business performance as well as scale the business for future growth. The centralized database system ensures that all relevant data is stored in a single, accessible location. This improves the speed and efficiency of data retrieval and contributes to faster decision-making. By moving from a file system to a relational database, WOW can streamline its operations. The database allows for efficient management of customer information, rental transactions, and financial data, leading to a more organized and agile business process. With this new database, WOW can better understand customer preferences and behaviors. This enables personalized services, targeted marketing, and

improved customer satisfaction. Finally, as WOW continues to grow, the relational database model provides scalability. New entities and relationships can be easily added to accommodate the evolving needs of the business. This is extremely important to a business like WOW. A scalable database expands the capabilities of the system and ensures a positive user experience even as demand increases.

Brief details of software

For our project, we chose PHP to handle the backend tasks, seamlessly integrating with the MySQL database. As for the frontend, we opted for HTML to create the user interface of the website. This straightforward combination aligns well with our development strategy, ensuring a smooth connection between server-side functionalities and the user experience.

MySQL DDL

```
-- phpMyAdmin SQL Dump
-- version 5.2.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Dec 10, 2023 at 08:09 PM
-- Server version: 10.4.32-MariaDB
-- PHP Version: 8.0.30

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
```

```
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS
*/;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `project`
--

-- --------------------------------------------------------

--
-- Table structure for table `users`
--

CREATE TABLE `users` (
  `id` int(11) NOT NULL,
  `username` varchar(50) NOT NULL,
  `password` varchar(255) NOT NULL,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `customerid` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `users`
--

INSERT INTO `users` (`id`, `username`, `password`, `created_at`, `customerid`) VALUES
(3, 'zc2958', '$2y$10$EG7yntQEaRWEYfGeat.Mhe5S0Kre/khZC/ZUcFByuPH5IoSIJU9fa',
'2023-12-09 20:25:32', 36);
```

```
-- ----------------------------------------------------


--
-- Table structure for table `zc_corpcust`
--

CREATE TABLE `zc_corpcust` (
  `customerid` int(11) NOT NULL COMMENT 'THE ID OF A CUSTOMER',
  `empid` int(11) NOT NULL COMMENT 'EMPLOYEE ID OF CORPRATION',
  `corpid` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;


--
-- Triggers `zc_corpcust`
--
DELIMITER $$
CREATE TRIGGER `arc_fkarc_1_zc_corpcust` BEFORE INSERT ON `zc_corpcust` FOR
EACH ROW BEGIN
    DECLARE d CHAR(1);

    SELECT
        a.type
    INTO d
    FROM
        zc_customer a
    WHERE
        a.customerid = NEW.customerid;

    IF (d IS NULL OR d <> 'C') THEN
        SIGNAL SQLSTATE '45000'
```

```
      SET MESSAGE_TEXT = 'FK ZC_CORPCUST_ZC_CUSTOMER_FK in Table
ZC_CORPCUST violates Arc constraint on Table ZC_CUSTOMER - discriminator column
TYPE doesn"t have value "C"';
   END IF;
END
$$
DELIMITER ;

-- ----------------------------------------------------------


--
-- Table structure for table `zc_corporation`
--

CREATE TABLE `zc_corporation` (
  `corpname` varchar(20) NOT NULL COMMENT 'COPORATION NAME',
  `regnum` varchar(10) NOT NULL COMMENT 'CORPORATION REGISTRATION
NUMBER ',
  `corpid` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- ----------------------------------------------------------


--
-- Table structure for table `zc_coupons`
--

CREATE TABLE `zc_coupons` (
  `couponid` int(11) NOT NULL COMMENT 'THE ID OF A COUPON',
  `discount` tinyint(4) NOT NULL COMMENT 'THE DISCOUNT OF PRODUCT',
  `startdate` datetime NOT NULL COMMENT 'THE START DATE OF COUPON',
```

```
  `enddate` datetime NOT NULL COMMENT 'THE END DATE OF COUPON'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `zc_coupons`
--

INSERT INTO `zc_coupons` (`couponid`, `discount`, `startdate`, `enddate`) VALUES
(36, 50, '2023-08-01 00:00:00', '2023-05-31 00:00:00');

-- --------------------------------------------------------

--
-- Table structure for table `zc_customer`
--

CREATE TABLE `zc_customer` (
  `customerid` int(11) NOT NULL COMMENT 'THE ID OF A CUSTOMER',
  `street` varchar(20) NOT NULL COMMENT 'STREET ADDRESS',
  `city` varchar(20) NOT NULL COMMENT 'CITY OF ADDRESS',
  `state` varchar(2) NOT NULL COMMENT 'STATE OF ADDRESS LIKE NY',
  `zipcode` varchar(5) NOT NULL,
  `email` varchar(30) NOT NULL COMMENT 'THE EMAIL OF A CUSTOMER',
  `phonenum` varchar(10) NOT NULL COMMENT 'PHONE NUMBER OF A CUSTOMER',
  `type` varchar(1) NOT NULL COMMENT 'I FOR INDIVIDUAL, C FOR CORPORATION
CUSTOMER'
) ;

--
-- Dumping data for table `zc_customer`
--
```

```
INSERT INTO `zc_customer` (`customerid`, `street`, `city`, `state`, `zipcode`, `email`,
`phonenum`, `type`) VALUES
(36, '86 schermerhorn', 'Brooklyn', 'Ne', '11201', 'zc2958@nyu.edu', '3477683265', 'I');

-- --------------------------------------------------------

--
-- Table structure for table `zc_customer_coupon`
--

CREATE TABLE `zc_customer_coupon` (
  `coupontype` varchar(1) NOT NULL COMMENT 'TYPE OF COUPON',
  `couponid` int(11) NOT NULL,
  `customerid` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `zc_customer_coupon`
--

INSERT INTO `zc_customer_coupon` (`coupontype`, `couponid`, `customerid`) VALUES
('I', 36, 36);

-- --------------------------------------------------------

--
-- Table structure for table `zc_individual`
--

CREATE TABLE `zc_individual` (
```

```
  `customerid` int(11) NOT NULL COMMENT 'THE ID OF A CUSTOMER',
  `fname` varchar(20) NOT NULL COMMENT 'FIRST NAME OF INDIVIDUAL',
  `lname` varchar(20) NOT NULL COMMENT 'LAST NAME OF INDIVIDUAL',
  `dlnum` varchar(9) NOT NULL COMMENT 'DRIVER LICENSE OF INDIVIDUAL',
  `insurance` varchar(20) NOT NULL COMMENT 'INSURANCE COMPANY NAME',
  `policynum` varchar(11) NOT NULL COMMENT 'INSURANCE POLICY NUMBER'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Triggers `zc_individual`
--
DELIMITER $$
CREATE TRIGGER `arc_fkarc_1_zc_individual` BEFORE INSERT ON `zc_individual` FOR
EACH ROW BEGIN
    DECLARE d CHAR(1);

    SELECT
        a.type
    INTO d
    FROM
        zc_customer a
    WHERE
        a.customerid = NEW.customerid;

    IF (d IS NULL OR d <> 'I') THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'FK ZC_INDIVIDUAL_ZC_CUSTOMER_FK in Table
ZC_INDIVIDUAL violates Arc constraint on Table ZC_CUSTOMER - discriminator column
TYPE doesn"t have value "I"';
    END IF;
END
```

```
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `zc_invoice`
--

CREATE TABLE `zc_invoice` (
  `invoiceid` int(11) NOT NULL,
  `invoicedate` datetime NOT NULL,
  `rentalid` int(11) NOT NULL,
  `amount` decimal(7,2) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `zc_invoice`
--

INSERT INTO `zc_invoice` (`invoiceid`, `invoicedate`, `rentalid`, `amount`) VALUES
(1, '2023-12-10 00:00:00', 1, 500.00);

--
-- Triggers `zc_invoice`
--
DELIMITER $$
CREATE TRIGGER `calculate_invoice_amount_on_insert` BEFORE INSERT ON `zc_invoice`
FOR EACH ROW BEGIN
    SET @v_rental_duration := 0;
    SET @v_used_mileage := 0;
```

```
    SET @v_daily_rate := 0;
    SET @v_over_mileage_fee := 0;
    SET @v_total_amount := 0;
    SET @v_discount := 0;
    SET @v_dailylimit := 0;
    SET @v_typeid := '';
    SET @v_couponid := 0;

    SELECT r.dailylimit, v.typeid, r.couponid
    INTO @v_dailylimit, @v_typeid, @v_couponid
    FROM zc_rental r
    INNER JOIN zc_vehicle v ON v.vin = r.vin
    WHERE r.rentalid = NEW.rentalid;

    SELECT DATEDIFF(dropoffdate, pickupdate), (endodo - startodo)
    INTO @v_rental_duration, @v_used_mileage
    FROM zc_rental
    WHERE rentalid = NEW.rentalid;

    SELECT rentalcharge, extracharge
    INTO @v_daily_rate, @v_over_mileage_fee
    FROM zc_vehicleclass
    WHERE typeid = @v_typeid;

    SELECT COALESCE(discount, 0) INTO @v_discount FROM zc_coupons WHERE
couponid = @v_couponid;

    IF @v_used_mileage > (@v_dailylimit * @v_rental_duration) THEN
        SET @v_total_amount := (@v_daily_rate * @v_rental_duration) +
                    ((@v_used_mileage - (@v_dailylimit * @v_rental_duration)) *
@v_over_mileage_fee);
```

```
    ELSE
        SET @v_total_amount := @v_daily_rate * @v_rental_duration;
    END IF;

    SET NEW.amount := @v_total_amount;
END
$$
DELIMITER ;
DELIMITER $$
CREATE TRIGGER `generate_invoiceid` BEFORE INSERT ON `zc_invoice` FOR EACH
ROW BEGIN
  DECLARE max_invoiceid INT;
  SET max_invoiceid = (SELECT MAX(invoiceid) FROM zc_invoice);
  IF max_invoiceid IS NULL THEN
    SET NEW.invoiceid = 1;
  ELSE
    SET NEW.invoiceid = max_invoiceid + 1;
  END IF;
END
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `zc_office`
--

CREATE TABLE `zc_office` (
  `officeid` int(11) NOT NULL COMMENT 'THE ID OF OFFICE',
  `street` varchar(20) NOT NULL,
```

```
  `city` varchar(20) NOT NULL,
  `state` varchar(2) NOT NULL,
  `zipcode` varchar(5) NOT NULL,
  `officenum` varchar(10) NOT NULL COMMENT 'THE NUMBER OF OFFICE'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `zc_office`
--

INSERT INTO `zc_office` (`officeid`, `street`, `city`, `state`, `zipcode`, `officenum`) VALUES
(33, '400 Pine St', 'Seattle', 'WA', '98101', 'Office202'),
(34, 'Some Address', 'Some City', 'SC', '98102', 'Office103'),
(35, '293 Maple Ave', 'Seattle', 'WA', '98104', 'Office202'),
(36, '149 Oak St', 'New York', 'NY', '92342', 'Office103'),
(37, '4112 W 37th St', 'Los Angeles', 'CA', '90001', 'Office104'),
(38, '230 Pine Ave', 'Chicago', 'IL', '60007', 'Office108'),
(39, '489 Nick St', 'Boston', 'MA', '02835', 'Office109');

-- --------------------------------------------------------

--
-- Table structure for table `zc_payment`
--

CREATE TABLE `zc_payment` (
  `paymentid` int(11) NOT NULL,
  `method` varchar(10) NOT NULL COMMENT 'CREDIT,DEBIT,GIFTCARD',
  `DATE` datetime NOT NULL COMMENT 'THE DATE OF PAYMENT',
  `cardnum` varchar(16) NOT NULL COMMENT 'THE NUMBER OF A CARD',
  `cvv` varchar(4) NOT NULL COMMENT 'cvv of the card',
```

```
  `expdate` datetime NOT NULL COMMENT 'THE EXPIRATION DATE OF THE CARD',
  `paymentfname` varchar(20) NOT NULL COMMENT 'PAYMENT FIRST NAME ',
  `paymentlname` varchar(20) NOT NULL COMMENT 'PAYMENT LAST NAME',
  `invoiceid` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Triggers `zc_payment`
--
DELIMITER $$
CREATE TRIGGER `generate_paymentid` BEFORE INSERT ON `zc_payment` FOR EACH
ROW BEGIN
  DECLARE max_paymentid INT;
  SET max_paymentid = (SELECT MAX(paymentid) FROM zc_payment);
  IF max_paymentid IS NULL THEN
    SET NEW.paymentid = 1;
  ELSE
    SET NEW.paymentid = max_paymentid + 1;
  END IF;
END
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `zc_rental`
--

CREATE TABLE `zc_rental` (
  `rentalid` int(11) NOT NULL,
```

```
  `pickupdate` datetime NOT NULL COMMENT 'THE DATE OF PICKUP',
  `dropoffdate` datetime NOT NULL,
  `startodo` int(11) NOT NULL COMMENT 'THE START MILES OF ODOMETER',
  `endodo` int(11) NOT NULL COMMENT 'THE END MILES OF ODOMETER',
  `pickuploc` int(11) NOT NULL,
  `customerid` int(11) NOT NULL,
  `vin` varchar(18) NOT NULL,
  `dropoffloc` int(11) NOT NULL,
  `couponid` int(11) DEFAULT NULL,
  `dailylimit` smallint(6) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `zc_rental`
--

INSERT INTO `zc_rental` (`rentalid`, `pickupdate`, `dropoffdate`, `startodo`, `endodo`,
`pickuploc`, `customerid`, `vin`, `dropoffloc`, `couponid`, `dailylimit`) VALUES
(1, '2023-12-12 00:00:00', '2023-12-22 00:00:00', 1500, 2500, 33, 36, 'VIN23456789012345', 33,
36, 100);

--
-- Triggers `zc_rental`
--
DELIMITER $$
CREATE TRIGGER `generate_rentalid` BEFORE INSERT ON `zc_rental` FOR EACH ROW
BEGIN
  DECLARE max_rentalid INT;
  SET max_rentalid = (SELECT MAX(rentalid) FROM zc_rental);
  IF max_rentalid IS NULL THEN
    SET NEW.rentalid = 1;
```

```
  ELSE
    SET NEW.rentalid = max_rentalid + 1;
  END IF;
END
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `zc_vehicle`
--

CREATE TABLE `zc_vehicle` (
  `vin` varchar(18) NOT NULL COMMENT 'VIN number of a car',
  `make` varchar(20) NOT NULL COMMENT 'brand of a car',
  `model` varchar(20) NOT NULL COMMENT 'MODEL OF VEHICLE ',
  `year` smallint(6) NOT NULL COMMENT 'the year of a car',
  `plate` varchar(10) NOT NULL COMMENT 'license plate of a car',
  `officeid` int(11) NOT NULL,
  `typeid` smallint(6) NOT NULL,
  `dailylimit` smallint(6) DEFAULT NULL,
  `odometer` smallint(6) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `zc_vehicle`
--

INSERT INTO `zc_vehicle` (`vin`, `make`, `model`, `year`, `plate`, `officeid`, `typeid`,
`dailylimit`, `odometer`) VALUES
```

('VIN01234567890123', 'Ford', 'Fusion', 2023, 'UVW234', 39, 67, 200, 2835),
('VIN12345678901234', 'Honda', 'Odyssey', 2023, 'XYZ789', 39, 68, 150, 0),
('VIN23456789012345', 'Chevrolet', 'Camaro', 2023, 'ABC456', 39, 69, 100, 1500),
('VIN34567890123456', 'Ford', 'F-150', 2023, 'DEF789', 39, 70, 300, 0),
('VIN45678901234567', 'Mercedes-Benz', 'S-Class', 2023, 'GHI012', 39, 71, NULL, 0),
('VIN56789012345678', 'Toyota', 'Sienna', 2023, 'GHI789', 38, 62, 150, 7564),
('VIN67890123456789', 'BMW', 'Z4', 2023, 'JKL012', 38, 63, NULL, 0),
('VIN78901234567890', 'Nissan', 'Sentra', 2023, 'MNO345', 38, 64, 120, 1235),
('VIN89012345678901', 'Mercedes-Benz', 'GLE', 2023, 'PQR678', 38, 65, NULL, 0),
('VIN90123456789012', 'Toyota', 'Prius', 2023, 'STU902', 39, 66, 120, 0);
('VIN99423423789012', 'Toyota', 'Corolla', 2023, 'STU911', 39, 66, 140, 2987);
('VIN89018755678901', 'Mercedes-Benz', 'C300', 2021, 'PQR878', 38, 65, 200, 5980),
('VIN78976234567890', 'Tesla', 'Model Y', 2023, 'MNO345', 38, 64, 200, 982),
('VIN01234567890124', 'Honda', 'Accord', 2023, 'JKL345', 37, 64, 150, 4500),
('VIN12345678901235', 'Ford', 'Explorer', 2023, 'MNO678', 37, 65, 200, 7890),
('VIN23456789012346', 'Chevrolet', 'Equinox', 2023, 'PQR901', 37, 66, 120, 2300),
('VIN34567890123457', 'Toyota', 'Camry', 2023, 'STU234', 38, 62, 180, 3450),
('VIN45678901234568', 'Nissan', 'Rogue', 2023, 'ABC567', 38, 63, 200, 6780),
('VIN56789012345679', 'Tesla', 'Model 3', 2023, 'DEF890', 38, 64, 150, 1200),
('VIN67890123456780', 'BMW', 'X5', 2023, 'GHI123', 39, 67, 250, 5600),
('VIN78901234567891', 'Mercedes-Benz', 'A-Class', 2023, 'UVW567', 39, 68, NULL, 0),
('VIN89012345678902', 'Toyota', 'Highlander', 2023, 'XYZ890', 39, 69, 200, 8900),
('VIN90123456789013', 'Ford', 'Mustang', 2023, 'JKL123', 39, 70, 100, 3400),
('VIN99423423789014', 'Chevrolet', 'Malibu', 2023, 'MNO456', 39, 71, NULL, 0),
('VIN89018755678903', 'Nissan', 'Altima', 2023, 'PQR789', 37, 62, 150, 4567),
('VIN78976234567892', 'Honda', 'Pilot', 2023, 'STU012', 37, 63, 180, 6700);

-- --

--

-- Table structure for table `zc_vehicleclass`

```
--

CREATE TABLE `zc_vehicleclass` (
  `typeid` smallint(6) NOT NULL COMMENT 'THE TYPE ID OF VEHICLE',
  `vehicletype` varchar(10) NOT NULL COMMENT 'SUV, SC(small car), MC(median car),LUX,PSUV(premier suv),MINI VAN, SW(station wagon), EV(electric vehicle), OTHER\r\n',
  `rentalcharge` decimal(6,2) NOT NULL,
  `extracharge` decimal(6,2) NOT NULL COMMENT 'EXTRA CHARGE OUT OF THE LIMITATION'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `zc_vehicleclass`
--

INSERT INTO `zc_vehicleclass` (`typeid`, `vehicletype`, `rentalcharge`, `extracharge`) VALUES
(33, 'SUV', 40.00, 8.00),
(35, 'SUV', 40.00, 8.00),
(36, 'Sedan', 35.00, 7.00),
(37, 'Truck', 50.00, 10.00),
(38, 'Conv', 55.00, 12.00),
(39, 'Compact', 30.00, 5.00),
(40, 'Luxury SUV', 70.00, 15.00),
(41, 'Sedan', 35.00, 7.00),
(42, 'Truck', 50.00, 10.00),
(43, 'Electric', 45.00, 9.00),
(44, 'Conv', 55.00, 12.00),
(45, 'Lux', 60.00, 15.00),
(46, 'Hybrid', 50.00, 10.00),
```

```
(47, 'Sedan', 45.00, 9.00),
(48, 'Sports Car', 55.00, 11.00),
(49, 'Compact', 40.00, 8.00),
(50, 'SUV', 50.00, 10.00),
(51, 'Hybrid', 55.00, 12.00),
(52, 'Conv', 60.00, 15.00),
(60, 'Sedan', 45.00, 9.00),
(61, 'Truck', 55.00, 12.00),
(62, 'Minivan', 50.00, 10.00),
(63, 'Conv', 60.00, 15.00),
(64, 'Compact', 35.00, 7.00),
(65, 'Luxury SUV', 70.00, 18.00),
(66, 'Hybrid', 45.00, 10.00),
(67, 'Sedan', 35.00, 7.00),
(68, 'Minivan', 55.00, 12.00),
(69, 'Conv', 50.00, 15.00),
(70, 'Truck', 60.00, 18.00),
(71, 'Luxury', 70.00, 20.00);

--
-- Indexes for dumped tables
--

--
-- Indexes for table `users`
--
ALTER TABLE `users`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `username` (`username`),
  ADD KEY `fk_customerid` (`customerid`);
```

```
--
-- Indexes for table `zc_corpcust`
--
ALTER TABLE `zc_corpcust`
  ADD PRIMARY KEY (`customerid`),
  ADD UNIQUE KEY `zc_corpcust_pkv1` (`empid`),
  ADD KEY `zc_corpcust_zc_corporation_fk` (`corpid`);

--
-- Indexes for table `zc_corporation`
--
ALTER TABLE `zc_corporation`
  ADD PRIMARY KEY (`corpid`);

--
-- Indexes for table `zc_coupons`
--
ALTER TABLE `zc_coupons`
  ADD PRIMARY KEY (`couponid`);

--
-- Indexes for table `zc_customer`
--
ALTER TABLE `zc_customer`
  ADD PRIMARY KEY (`customerid`);

--
-- Indexes for table `zc_customer_coupon`
--
ALTER TABLE `zc_customer_coupon`
  ADD PRIMARY KEY (`couponid`),
```

```
  ADD KEY `zc_customer_coupon_zc_fk` (`customerid`);

--
-- Indexes for table `zc_individual`
--
ALTER TABLE `zc_individual`
  ADD PRIMARY KEY (`customerid`);

--
-- Indexes for table `zc_invoice`
--
ALTER TABLE `zc_invoice`
  ADD PRIMARY KEY (`invoiceid`),
  ADD UNIQUE KEY `zc_invoice__idxv1` (`rentalid`);

--
-- Indexes for table `zc_office`
--
ALTER TABLE `zc_office`
  ADD PRIMARY KEY (`officeid`);

--
-- Indexes for table `zc_payment`
--
ALTER TABLE `zc_payment`
  ADD PRIMARY KEY (`paymentid`),
  ADD UNIQUE KEY `zc_payment__idx` (`invoiceid`);

--
-- Indexes for table `zc_rental`
--
```

```
ALTER TABLE `zc_rental`
  ADD PRIMARY KEY (`rentalid`),
  ADD KEY `zc_rental_zc_coupons_fk` (`couponid`),
  ADD KEY `zc_rental_zc_customer_fk` (`customerid`),
  ADD KEY `zc_rental_zc_office_fk` (`pickuploc`),
  ADD KEY `zc_rental_zc_office_fkv2` (`dropoffloc`),
  ADD KEY `zc_rental_zc_vehicle_fk` (`vin`);

--
-- Indexes for table `zc_vehicle`
--
ALTER TABLE `zc_vehicle`
  ADD PRIMARY KEY (`vin`),
  ADD KEY `zc_vehicle_zc_office_fk` (`officeid`),
  ADD KEY `zc_vehicle_zc_vehicleclass_fk` (`typeid`);

--
-- Indexes for table `zc_vehicleclass`
--
ALTER TABLE `zc_vehicleclass`
  ADD PRIMARY KEY (`typeid`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `users`
--
ALTER TABLE `users`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;
```

```
--
-- AUTO_INCREMENT for table `zc_corporation`
--
ALTER TABLE `zc_corporation`
  MODIFY `corpid` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=34;

--
-- AUTO_INCREMENT for table `zc_customer`
--
ALTER TABLE `zc_customer`
  MODIFY `customerid` int(11) NOT NULL AUTO_INCREMENT COMMENT 'THE ID OF A
CUSTOMER';

--
-- Constraints for dumped tables
--

--
-- Constraints for table `users`
--
ALTER TABLE `users`
  ADD CONSTRAINT `fk_customerid` FOREIGN KEY (`customerid`) REFERENCES
`zc_customer` (`customerid`);

--
-- Constraints for table `zc_corpcust`
--
ALTER TABLE `zc_corpcust`
  ADD CONSTRAINT `zc_corpcust_zc_corporation_fk` FOREIGN KEY (`corpid`)
REFERENCES `zc_corporation` (`corpid`),
```

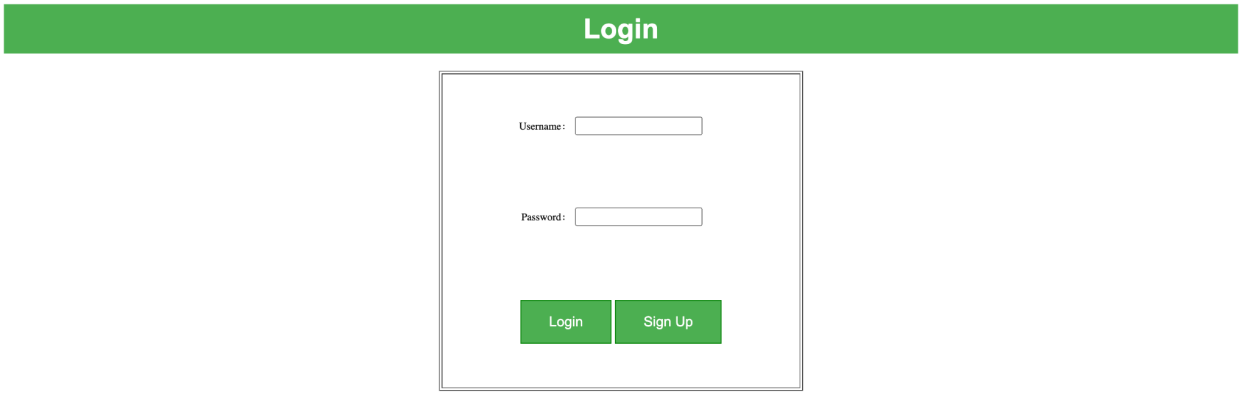
```
  ADD CONSTRAINT `zc_corpcust_zc_customer_fk` FOREIGN KEY (`customerid`)
REFERENCES `zc_customer` (`customerid`);

--
-- Constraints for table `zc_customer_coupon`
--
ALTER TABLE `zc_customer_coupon`
  ADD CONSTRAINT `zc_customer_coupon_fk` FOREIGN KEY (`couponid`) REFERENCES
`zc_coupons` (`couponid`),
  ADD CONSTRAINT `zc_customer_coupon_zc_fk` FOREIGN KEY (`customerid`)
REFERENCES `zc_customer` (`customerid`);

--
-- Constraints for table `zc_individual`
--
ALTER TABLE `zc_individual`
  ADD CONSTRAINT `zc_individual_zc_customer_fk` FOREIGN KEY (`customerid`)
REFERENCES `zc_customer` (`customerid`);

--
-- Constraints for table `zc_invoice`
--
ALTER TABLE `zc_invoice`
  ADD CONSTRAINT `zc_invoice_zc_rental_fk` FOREIGN KEY (`rentalid`) REFERENCES
`zc_rental` (`rentalid`);

--
-- Constraints for table `zc_payment`
--
ALTER TABLE `zc_payment`
```

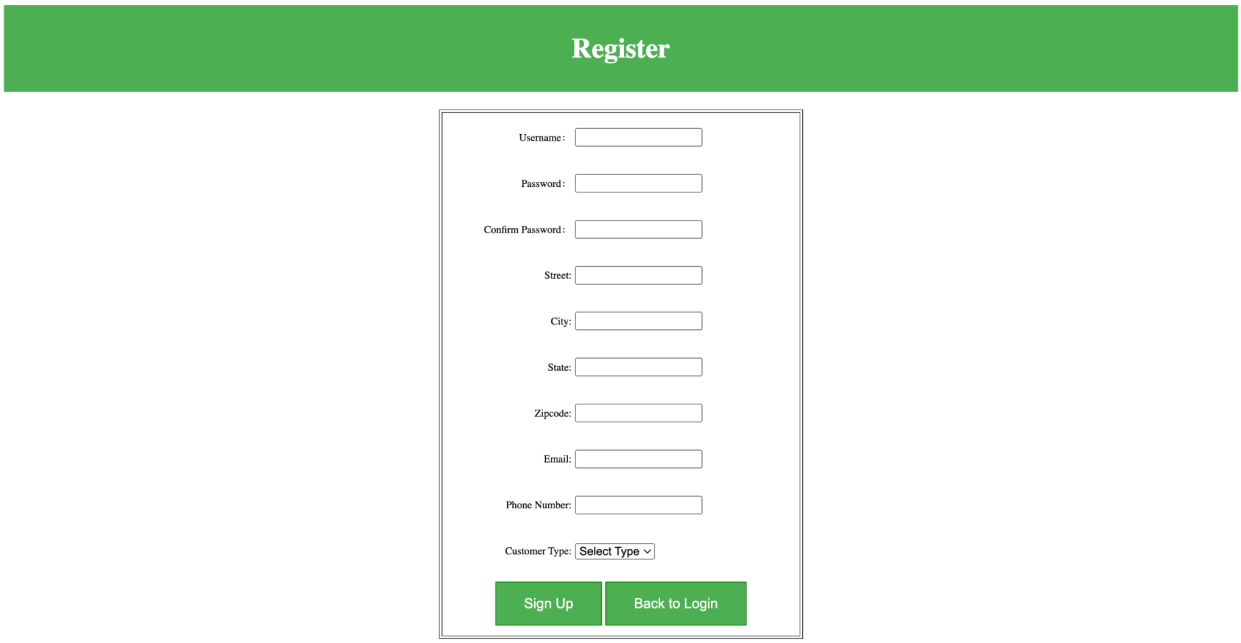
```
  ADD CONSTRAINT `zc_payment_zc_invoice_fk` FOREIGN KEY (`invoiceid`)
REFERENCES `zc_invoice` (`invoiceid`);

--
-- Constraints for table `zc_rental`
--
ALTER TABLE `zc_rental`
  ADD CONSTRAINT `zc_rental_zc_coupons_fk` FOREIGN KEY (`couponid`)
REFERENCES `zc_coupons` (`couponid`),
  ADD CONSTRAINT `zc_rental_zc_customer_fk` FOREIGN KEY (`customerid`)
REFERENCES `zc_customer` (`customerid`),
  ADD CONSTRAINT `zc_rental_zc_office_fk` FOREIGN KEY (`pickuploc`) REFERENCES
`zc_office` (`officeid`),
  ADD CONSTRAINT `zc_rental_zc_office_fkv2` FOREIGN KEY (`dropoffloc`)
REFERENCES `zc_office` (`officeid`),
  ADD CONSTRAINT `zc_rental_zc_vehicle_fk` FOREIGN KEY (`vin`) REFERENCES
`zc_vehicle` (`vin`);

--
-- Constraints for table `zc_vehicle`
--
ALTER TABLE `zc_vehicle`
  ADD CONSTRAINT `zc_vehicle_zc_office_fk` FOREIGN KEY (`officeid`) REFERENCES
`zc_office` (`officeid`),
  ADD CONSTRAINT `zc_vehicle_zc_vehicleclass_fk` FOREIGN KEY (`typeid`)
REFERENCES `zc_vehicleclass` (`typeid`);
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

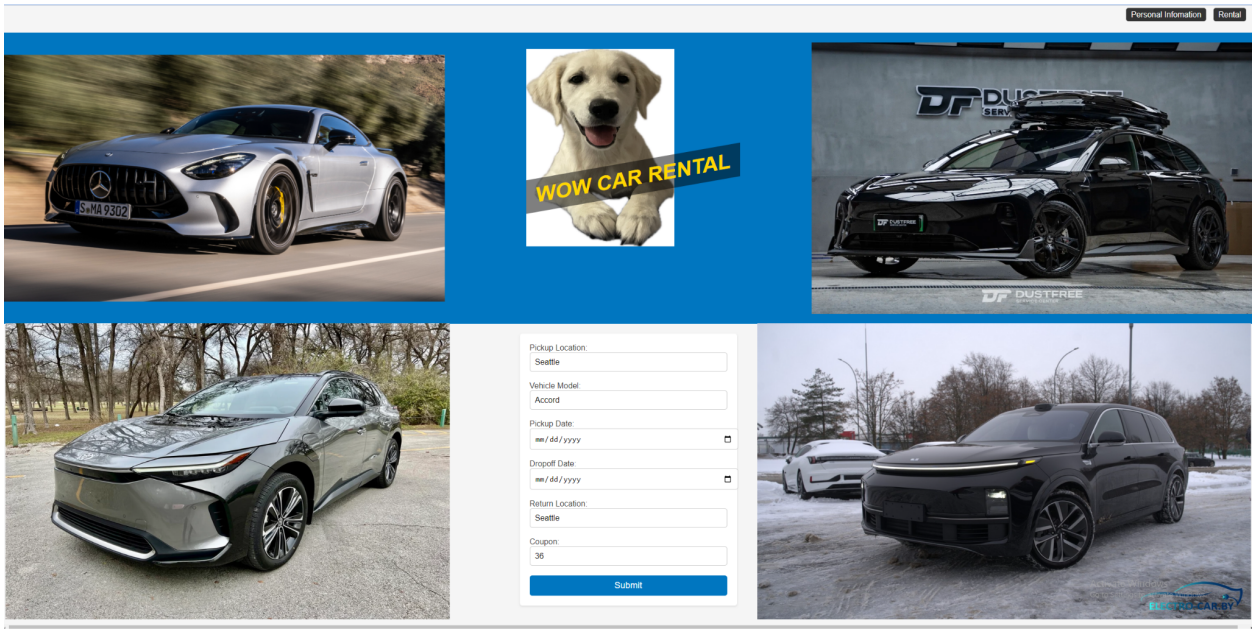
List of Tables

- users
    -
- zc_corpcust
- zc_corporation
- zc_coupons
- zc_customer
- zc_customer_coupon
- zc_individual
- zc_invoice
- zc_office
    - We will have 5 office locations around the US. Each office will have at least 6 cars available to be rented.
- zc_payment
- zc_rental
- zc_vehicle
    - There will be 30 available vehicles able to rent across the WOW office locations.
- zc_vehicleclass
    - There will be 9 vehicle classes available.

Screenshots







Rental Bill Information

Pickup location: 33

Dropoff location: 33

Pickup Date: 2023-12-14

Dropoff Date: 2023-12-21

Vehicle Model: Model Y

Starting Odometer: 982

Rental Days: 7 days

Daily Limit: 200 miles

Total Limit: 1400 miles

Confirm Payment

Price: 245.00

Please Enter Your Payment Information

First Name:
zhuo

Last Name:
chen

Payment Method:
Credit

Card Number:
12313412421

Expiration Month:
01

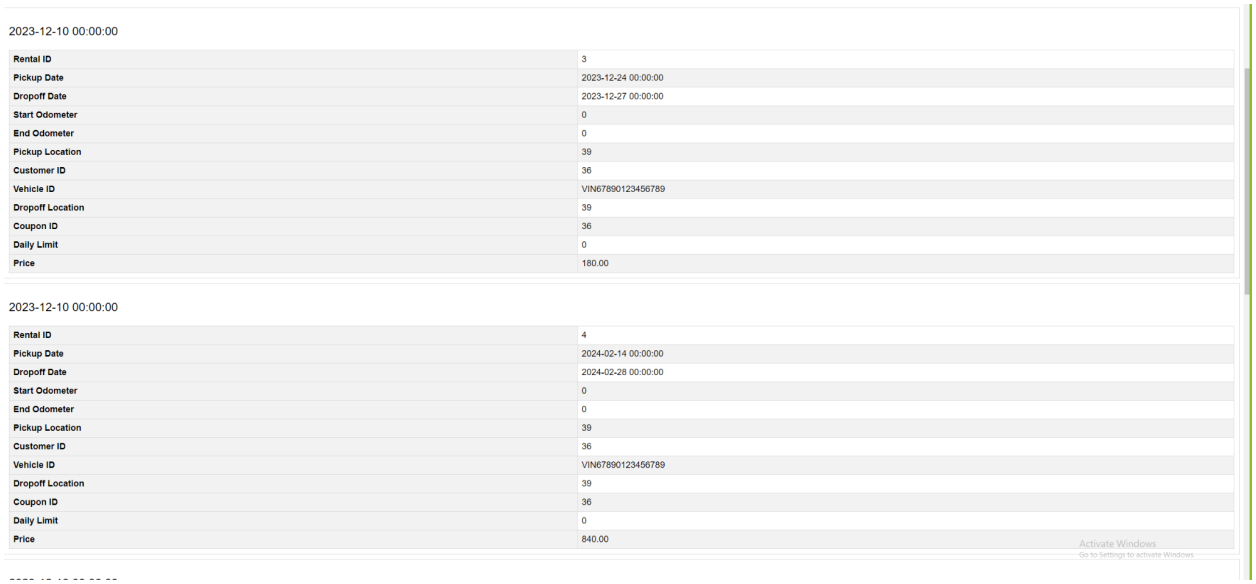
Expiration Year:
2023

CVV:
123

Submit Payment

Personal Information

Customer ID	Street	City	State	Zipcode	Email	Phone	Actions
36	86 schermerhorn	Brooklyn	Ne	11201	ze2958@nyu.edu	3477683265	Modify



2023-12-10 00:00:00

Rental ID	3
Pickup Date	2023-12-24 00:00:00
Dropoff Date	2023-12-27 00:00:00
Start Odometer	0
End Odometer	0
Pickup Location	39
Customer ID	36
Vehicle ID	VIN67890123456789
Dropoff Location	39
Coupon ID	36
Daily Limit	0
Price	180.00

2023-12-10 00:00:00

Rental ID	4
Pickup Date	2024-02-14 00:00:00
Dropoff Date	2024-02-28 00:00:00
Start Odometer	0
End Odometer	0
Pickup Location	39
Customer ID	36
Vehicle ID	VIN67890123456789
Dropoff Location	39
Coupon ID	36
Daily Limit	0
Price	840.00

Security features

```php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST["username"];
    $password = $_POST["password"];

    // 准备和绑定
    $stmt = $conn->prepare("SELECT password FROM users WHERE username = ?");
    $stmt->bind_param("s", $username);
    $stmt->execute();
    $result = $stmt->get_result();
    $user = $result->fetch_assoc();
```

```php
if ($stmt->num_rows > 0) {
    // 用户名已存在, 防xss攻击 using htmlspecialchars
    $username = htmlspecialchars($username, ENT_QUOTES, 'UTF-8');
    echo "<script>alert('Username $username already exists.');window.location.href='register.html';</script>";
} else {
```

Above are two screenshots of the security features we have implemented to protect against SQL injection attacks. In the first screenshot ,our security feature is the prepare statement. When using prepared statements, you first send only the structure of the SQL query, not the actual data. For example,INSERT INTO table (column)VALUES (?).Here,?' acts as a placeholder for the data that will be inserted. Afterwards, you bind the data to these placeholders.

This process is managed by the DBS and ensures that the data passed to the SQL statement is not interpreted as part of the SQL code. This means that even if the input data contains potential malicious SQL code fragments, they will be treated solely as data, not as part of an SQL command.Prepared statements are often pre-compiled on the server. This pre-compilation feature can improve performance when the same SQL structure is executed multiple times, as the database doesn't need to parse and compile the SQL statement on each execution. By using prepared statements, you prevent malicious SQL code within user input from being executed by the database. This significantly reduces the risk of SQL injection attacks, as attackers cannot alter the structure of the SOL statement through user input. The second screenshot protects us against cross site scripting attacks. XSS attacks typically involve injecting malicious scripts into web pages, which are then executed in the context of the user's browser. By converting special characters to HTML entities, “htmlspecialchars” ensures that any script tags or other HTML elements injected into the page do not get executed but are rather displayed as plain text.
This function ensures that the data integrity is maintained and that user input is displayed exactly as entered, preventing any malicious interpretation or execution.

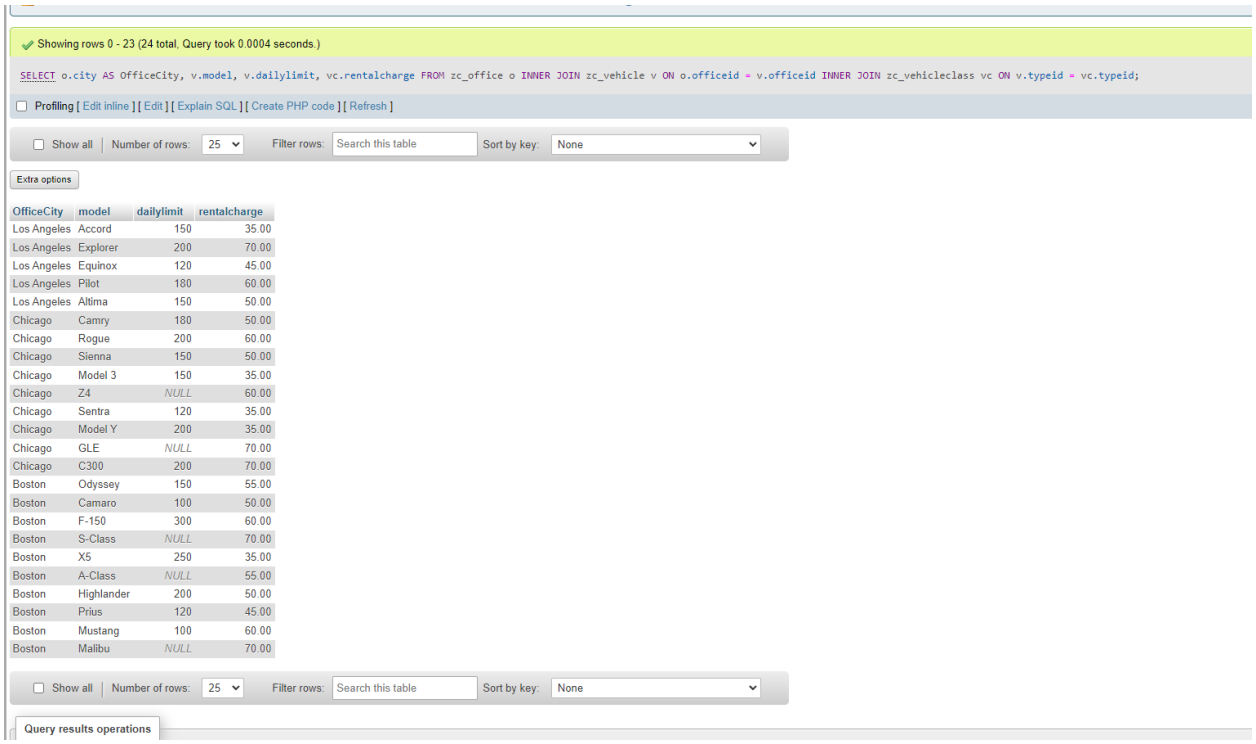
Lessons learned

All in all, I feel that the project went very well. I have gained valuable insights into collaborative coding and effective communication within a team. More specifically, I gained more knowledge on how to create a local website, connecting the frontend with the backend. Learning PHP and a bit of HTML will go a long way for me. The project's success can be attributed to our well-defined plan and the distribution of tasks among team members. We ensured that everyone had a clear understanding of their responsibilities, fostering a collaborative environment. Regular check-ins on Zoom facilitated seamless coordination, even though we were working remotely, we were very efficient with excellent communication between the three of us. One area for improvement could be refining our time management strategies to optimize productivity further. Overall, the project's positive outcome highlighted the importance of teamwork, communication, and efficient task allocation in achieving successful results.

SQL Queries

Q1) A1) SELECT o.city AS OfficeCity, v.model, v.dailylimit, vc.rentalcharge
FROM zc_office o
INNER JOIN zc_vehicle v ON o.officeid = v.officeid
INNER JOIN zc_vehicleclass vc ON v.typeid = vc.typeid;

A2)



Showing rows 0 - 23 (24 total, Query took 0.0004 seconds.)

```
SELECT o.city AS OfficeCity, v.model, v.dailylimit, vc.rentalcharge FROM zc_office o INNER JOIN zc_vehicle v ON o.officeid = v.officeid INNER JOIN zc_vehicleclass vc ON v.typeid = vc.typeid;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

OfficeCity	model	dailylimit	rentalcharge
Los Angeles	Accord	150	35.00
Los Angeles	Explorer	200	70.00
Los Angeles	Equinox	120	45.00
Los Angeles	Pilot	180	60.00
Los Angeles	Altima	150	50.00
Chicago	Camry	180	50.00
Chicago	Rogue	200	60.00
Chicago	Sienna	150	50.00
Chicago	Model 3	150	35.00
Chicago	Z4	NULL	60.00
Chicago	Sentra	120	35.00
Chicago	Model Y	200	35.00
Chicago	GLE	NULL	70.00
Chicago	C300	200	70.00
Boston	Odyssey	150	55.00
Boston	Camaro	100	50.00
Boston	F-150	300	60.00
Boston	S-Class	NULL	70.00
Boston	X5	250	35.00
Boston	A-Class	NULL	55.00
Boston	Highlander	200	50.00
Boston	Prius	120	45.00
Boston	Mustang	100	60.00
Boston	Malibu	NULL	70.00

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Query results operations

A3) For this question, we decided to join together the zc_office, zc_vehicle and zc_vehicleclass tables. This will answer the business question of what vehicles are available at each office location.

Q2) A1) SELECT
    c.customerid,
    c.email,
    c.phonenum,
    COUNT(r.rentalid) AS total_rentals

```
FROM
    zc_customer c
LEFT JOIN
    zc_rental r ON c.customerid = r.customerid
GROUP BY
    c.customerid, c.email, c.phonenum
ORDER BY
    total_rentals DESC;
```

A2)

A3) This query will retrieve the details of the customer along with the total number of rentals they have. This will answer the business question of what customer rented which vehicles.

Q3) A1) SELECT
    c.customerid,
    c.email,
    c.phonenum,
    (SELECT COUNT(*) FROM zc_customer_coupon cc WHERE cc.customerid = c.customerid) AS total_coupons_used
FROM
    zc_customer c

ORDER BY

total_coupons_used DESC;

A2)

正在显示第 0 – 2 行 (共 3 行, 查询花费 0.0003 秒。)

```
SELECT c.customerid, c.email, c.phonenum, (SELECT COUNT(*) FROM zc_customer_coupon cc WHERE cc.customerid = c.customerid) AS total_coupons_used
FROM zc_customer c ORDER BY total_coupons_used DESC;
```

性能分析 [编辑内嵌] [编辑] [解析 SQL] [创建 PHP 代码] [刷新]

显示全部 | 行数: 25 过滤行: 在表中搜索

额外选项

customerid	email	phonenum	total_coupons_used 1
36	zc2958@nyu.edu	3477683265	1
37	wenxuanhu@gmail.com	7855922333	0
38	george@gmail.com	4323423453	0

显示全部 | 行数: 25 过滤行: 在表中搜索

A3) This query will retrieve the details of customers along with the total number of coupons they have redeemed. This will answer the business question of what customer used what coupons.

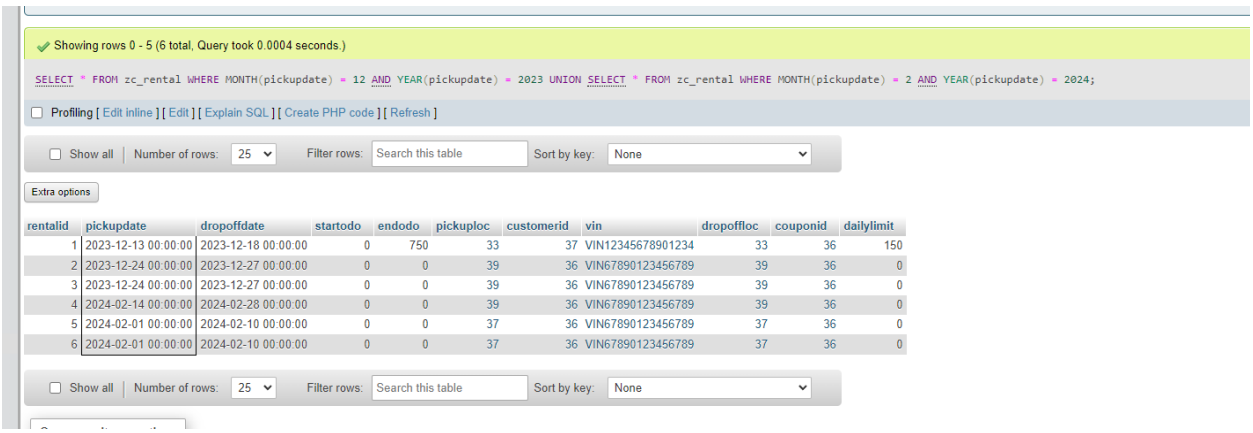
Q4) A1) SELECT * FROM zc_rental

WHERE MONTH(pickupdate) = 12 AND YEAR(pickupdate) = 2023

UNION SELECT * FROM zc_rental

WHERE MONTH(pickupdate) = 2 AND YEAR(pickupdate) = 2024;

A2)



Showing rows 0 - 5 (6 total, Query took 0.0004 seconds.)

```
SELECT * FROM zc_rental WHERE MONTH(pickupdate) = 12 AND YEAR(pickupdate) = 2023 UNION SELECT * FROM zc_rental WHERE MONTH(pickupdate) = 2 AND YEAR(pickupdate) = 2024;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

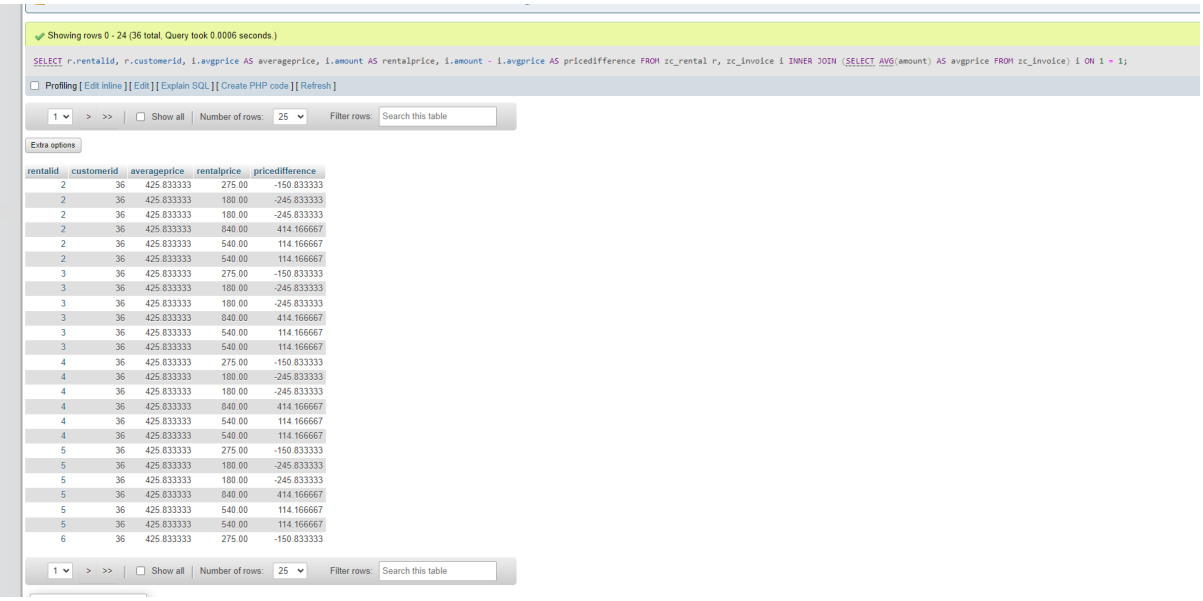
rentalid	pickupdate	dropoffdate	startodo	endodo	pickuploc	customerid	vin	dropoffloc	couponid	dailylimit
1	2023-12-13 00:00:00	2023-12-18 00:00:00	0	750	33	37	VIN12345678901234	33	36	150
2	2023-12-24 00:00:00	2023-12-27 00:00:00	0	0	39	36	VIN67890123456789	39	36	0
3	2023-12-24 00:00:00	2023-12-27 00:00:00	0	0	39	36	VIN67890123456789	39	36	0
4	2024-02-14 00:00:00	2024-02-28 00:00:00	0	0	39	36	VIN67890123456789	39	36	0
5	2024-02-01 00:00:00	2024-02-10 00:00:00	0	0	37	36	VIN67890123456789	37	36	0
6	2024-02-01 00:00:00	2024-02-10 00:00:00	0	0	37	36	VIN67890123456789	37	36	0

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

A3) This query still answers the business question: "How many rentals happened in December of 2023 and February of 2024.”

Q5) A1) SELECT r.rentalid, r.customerid, i.avgprice AS averageprice, i.amount AS rentalprice, i.amount - i.avgprice
AS pricedifference
FROM zc_rental r, zc_invoice i
INNER JOIN (SELECT AVG(amount) AS avgprice FROM zc_invoice) i ON 1 = 1;

A2)



A3) This query selects all data from the 'rental' table, retrieves the corresponding 'amount' from the 'invoice' table, and calculates the difference between the 'amount' for each record and the average 'amount'. This will answer a WOW business question.

Q6) A1) SELECT c.customerid, SUM(i.amount)

AS totalamountspent

FROM zc_customer c

INNER JOIN zc_rental r

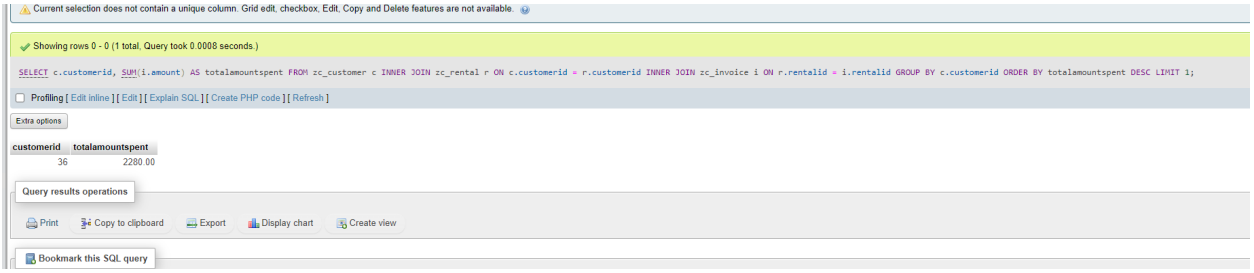
ON c.customerid = r.customerid

INNER JOIN zc_invoice i

ON r.rentalid = i.rentalid

GROUP BY c.customerid ORDER BY totalamountspent DESC LIMIT 1;

A2)



A3) This query finds the customer that spent the most amount of money on rentals at WOW. This is especially useful for when WOW is analyzing their customer data.

Protection against deadlocks and concurrent transactions

```php
    // Build SQL query to update customer information
    $sql = "UPDATE zc_customer SET street = '$street', city = '$city', state = '$state', zipcode = '$zipcode', email = '$email', phonenum = '$phonenum' WHERE customerid = '$customerid'";
    if ($conn->query($sql) === TRUE) {
        // Commit the transaction after a successful update
        $conn->commit();
        echo "Update successful";

        // After successful update, redirect to personal_info.php
        header("Location: personal_info.php");
        exit;
    } else {
        // Rollback the transaction in case of an error
        $conn->rollback();
        echo "Update failed: " . $conn->error;
    }
} else {
    echo "No customer records found.";
}

// Close the database connection
$conn->close();
?>
```

We added a commit to protect against deadlocks and concurrent transactions when updating customer information.