SQL Assessment

* 1. **Display all available vehicles on a given date.**

SELECT \* FROM vehicles WHERE Vehicle\_id NOT IN (

SELECT Vehicle\_id FROM reservations

WHERE ‘given date’ BETWEEN Departure\_Date AND Return\_Date );

Date used ‘2024-01-24’:

A screenshot of a computer

Description automatically generated

A reservation of the vehicle with the Vehicle\_id of 16 is made within the set time so the vehicle is not show.

* 1. **Display how many vehicles each department has used, and who are the most recurrent faculty members using the service.**

SELECT d.Department\_Name, COUNT(r.Vehicle\_id) AS Used\_Vehicles\_Count, f.Members\_Name AS Recurrent\_Faculty

FROM departments d

LEFT JOIN faculty\_members f ON d.Department\_id = f.Department\_id

LEFT JOIN reservations r ON f.Faculty\_id = r.Faculty\_id

GROUP BY d.Department\_id, f.Faculty\_id

ORDER BY d.Department\_Name, Used\_Vehicles\_Count DESC;

A screenshot of a computer

Description automatically generated

* 1. **Display the total mileage driven by a department or faculty member this year.**

SELECT

CASE WHEN f.Faculty\_id IS NOT NULL THEN f.Members\_Name ELSE d.Department\_Name END AS Entity,

SUM(co.Odometer\_End - co.Odometer\_Start) AS Total\_Mileage\_This\_Year

FROM departments d

LEFT JOIN faculty\_members f ON d.Department\_id = f.Department\_id

LEFT JOIN reservations r ON f.Faculty\_id = r.Faculty\_id

LEFT JOIN checkout\_forms ch ON r.Reservation\_id = ch.Reservation\_id

LEFT JOIN completion\_forms co ON ch.Checkout\_id = co.Checkout\_id

WHERE YEAR(r.Return\_Date) = YEAR(CURRENT\_DATE())

GROUP BY Entity;

A screenshot of a data

Description automatically generated

* 1. **Show details of a particular bill.**

SELECT f.Members\_Name AS Faculty\_Name,

SUM(co.Odometer\_End - co.Odometer\_Start) \* 1.44 AS Total\_Spent,

co.Card\_Number AS Card\_Used

FROM faculty\_members f

LEFT JOIN reservations r ON f.Faculty\_id = r.Faculty\_id

LEFT JOIN checkout\_forms ch ON r.Reservation\_id = ch.Reservation\_id

LEFT JOIN completion\_forms co ON ch.Checkout\_id = co.Checkout\_id

WHERE f.Members\_Name = 'A Members\_Name'

GROUP BY Faculty\_Name, Card\_Used;



Mr Smith was used as the ‘A Members\_Name’. I used 1.44 as its the current conversion rate for petrol per litre.

**1.5. Display those who booked vehicles but do not use them.**

SELECT f.Members\_Name AS Faculty\_Name, r.Reservation\_id

FROM faculty\_members f

JOIN reservations r ON f.Faculty\_id = r.Faculty\_id

WHERE r.Reservation\_id NOT IN (

SELECT ch.Reservation\_id

FROM checkout\_forms ch);

A screenshot of a computer

Description automatically generated

**1.6. Display the summary of vehicle usage per month, per faculty member, and department since the very beginning.**

SELECT

YEAR(r.Return\_Date) AS Year,

MONTH(r.Return\_Date) AS Month,

f.Members\_Name AS Faculty\_Name,

d.Department\_Name,

SUM(co.Odometer\_End - co.Odometer\_Start) AS Total\_Mileage

FROM completion\_forms co

LEFT JOIN checkout\_forms ch ON co.Checkout\_id = ch.Checkout\_id

LEFT JOIN reservations r ON ch.Reservation\_id = r.Reservation\_id

LEFT JOIN faculty\_members f ON r.Faculty\_id = f.Faculty\_id

LEFT JOIN departments d ON f.Department\_id = d.Department\_id

GROUP BY Year, Month, Faculty\_Name, Department\_Name

ORDER BY Year, Month;

A screenshot of a computer

Description automatically generated

**1.7. View the most used vehicle in my department that is not the most used by any other department.**

**2a. Database Diagram**

**A computer screen shot of a computer

Description automatically generated**

**2b. functions, procedure and triggers**

CREATE DEFINER=`root`@`localhost` PROCEDURE `AddReservation`(

IN faculty\_id INT,

IN vehicle\_id INT,

IN departure\_date DATE,

IN destination VARCHAR(45))

BEGIN

INSERT INTO Reservations (Faculty\_id, Vehicle\_id, Departure\_Date, Destination)

VALUES (faculty\_id, vehicle\_id, departure\_date, destination);

SELECT 'Reservation added successfully' AS Message;

END

CREATE DEFINER=`root`@`localhost` TRIGGER `maintenance\_jobs\_AFTER\_UPDATE` AFTER UPDATE ON `maintenance\_jobs` FOR EACH ROW BEGIN

IF NEW.Vehicle\_Fixed = TRUE THEN

UPDATE maintenance\_logs

SET Exit\_Date = CURRENT\_DATE()

WHERE Log\_ID = NEW.Log\_ID;

END IF;

END

CREATE DEFINER=`root`@`localhost` TRIGGER `maintenance\_logs\_AFTER\_UPDATE` AFTER UPDATE ON `maintenance\_logs` FOR EACH ROW BEGIN

UPDATE Vehicles

SET Last\_Maintenance = NEW.Exit\_Date

WHERE NEW.Exit\_Date != NULL;

END

**2c. Tools and Techniques used to generate data**

<https://www.randomlists.com/random-date>

**2d. SQL Queries**

INSERT INTO vehicles (Vehicle\_id, Vehicle\_Type, Last\_Maintenance) VALUES (8, 'car', '2005-11-10');

UPDATE departments SET Department\_Name = 'Economics' WHERE Department\_id = 1;

SELECT \* FROM reservations;

CREATE DEFINER=`root`@`localhost` TRIGGER `maintenance\_logs\_AFTER\_UPDATE` AFTER UPDATE ON `maintenance\_logs` FOR EACH ROW BEGIN

UPDATE Vehicles

SET Last\_Maintenance = NEW.Exit\_Date

WHERE NEW.Exit\_Date != NULL;

END