Wayne State University

Project 2: DrivewayDash

Sumaiya Ahmed (hd1015) | Dawlat Hamad (gv5450)
Professor Lu
CSC 4710
15 Dec. 2024

Table of Content

Overview	2
Entity Relationship Diagram	3
Assumptions and Justifications	4
Relationships	7
Relational Model	8
SQL	9

Overview

Project Description

This project involves designing and implementing a database-driven web application for managing driveway-sealing operations for a contractor, David Smith. The system includes functionalities for registration, request for quotes, negotiations, order creation, and billing. Clients can submit requests, negotiate quotes, track their progress, and pay bills. The contractor (David) manages incoming requests, generates work forms, order forms, bills, and tracks orders through a dashboard. The system also provides detailed queries, such as identifying high-value clients, overdue bills, and quote history.

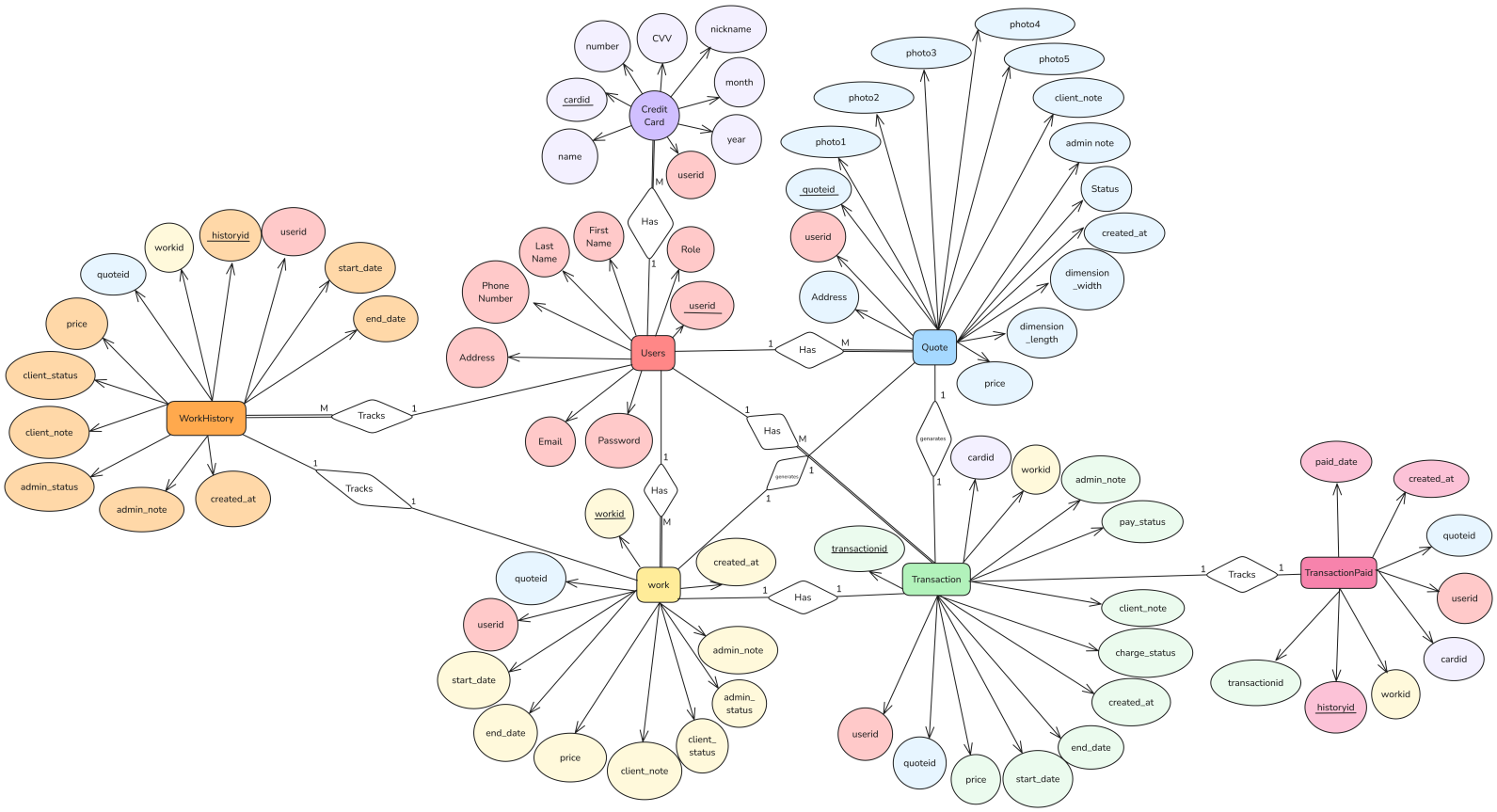
Project History

Sumaiya Ahmed => User Panel and Technical Report
Dawlat Hamad => Admin Panel and Login/Register Screen
Overall Work Time ≈ 65 hours

Youtube Link

[CSC4710-Project2] (https://youtu.be/Vc7ir4f1nSE)

Entity Relationship Diagram



Assumptions and Justifications

Entities:

- Users
  - Primary Key: userid
  - first Name
  - last Name
  - address
  - phone Number
  - email (unique)
  - password
  - role
- Card
  - Primary Key: cardid
  - Foreign Key: userid
  - nickname
  - number
  - Name
  - month
  - year
  - cvv
- Quote
  - Primary Key: quoteid
  - Foreign Key: userid
  - Address
  - dimension_length
  - dimension_width
  - price
  - photo1
  - photo2
  - photo3
  - photo4
  - photo5
  - client_note
  - status
  - Admin_note
  - created_at
- Transaction
  - Primary Key: transactionid
  - Foreign Key: userid
  - Foreign Key: quoteid
  - Foreign Key: workid
  - price

    - starte_date
    - end_date
    - pay_status
    - admin_note
- Work
    - Primary Key: workid
    - Foreign Key: userid
    - Foreign Key: quoteid
    - start_date
    - end_date
    - price
    - client_status
    - client_note
    - Admin_note
    - created_at
- WorkHistory
    - Primary Key: historyid
    - Foreign Key: workid
    - Foreign Key: userid
    - quoteid
    - start_date
    - end_date
    - price
    - client_status
    - client_note
    - admin_status
    - admin_note
    - created_at
- TransactionPaid
    - Primary Key: historyid
    - transactionid
    - userid
    - quoteid
    - workid
    - cardid
    - Created_at
    - paid_date

Justifications:

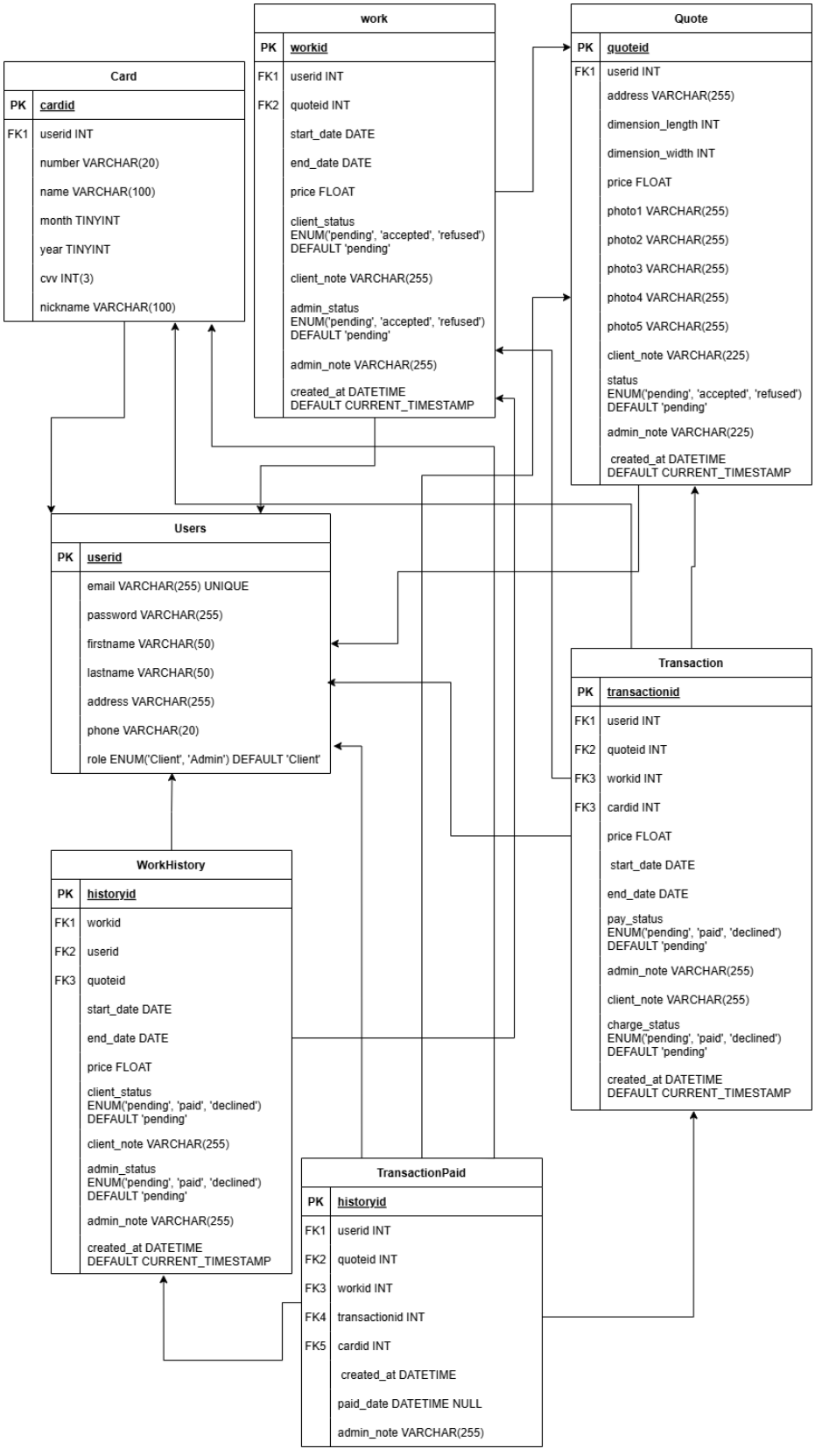
- Users: these attributes are standard for user profiles in most systems.
- Quote: these attributes cover details for requesting a quote.
- Work: these attributes reflect order tracking and scheduling.
- Transaction: these attributes cover details of a successful transaction.

- Card: these attributes cover the details needed for a payment method.
- WorkHistory: attributes keep track of changes in quotes, works, and orders.
- TransactionPaid: attributes keep track of users paying bill at what time for which order using which card.

Relationships

- Users to Credit Card: this is a one-to-many relationship. A user can have multiple credit cards, but a credit card can only be assigned to one user.
- Users to Quote: this is a one-to-many relationship. A user can have multiple quotes, but a quote can only belong to one user.
- Users to Work: this is also a one-to-many relationship. A user can have multiple work orders, but a work order can only belong to one user.
- User to Transaction: This is also another one-to-many relationship. Similar to other entities, a user can have multiple transactions, but a transaction can only belong to one user.
- Quote to work: This is a one-to-one relationship, because one quote can generate only one work order.
- Quote to Transaction: This is a one-to-one relationship, a quote can only be used for one transaction.
- WorkHistory to Users: This is a one-to-many relationship, as users can have multiple work histories but one work history can belong to only one user.
- WorkHistory to work: This is a one-to-one relationship, as one work can only be tracked by one work history.
- Transaction to work: This is a one-to-one relationship because one work can only have one transaction history.
- TransactionPaid to transaction: This is a one-to-one relationship as users can only pay one time.

Relationship Model



SQL

```
CREATE TABLE Users (
    userid INT AUTO_INCREMENT PRIMARY KEY,
    email VARCHAR(255) UNIQUE,
    password VARCHAR(255),
    firstname VARCHAR(50),
    lastname VARCHAR(50),
    address VARCHAR(255),
    phone VARCHAR(20),
    role VARCHAR(20) DEFAULT 'client'
);

CREATE TABLE Quote (
    quoteid INT AUTO_INCREMENT PRIMARY KEY,
    userid INT,
    address VARCHAR(255),
    dimension_length INT,
    dimension_width INT,
    price FLOAT,
    photo1 VARCHAR(255),
    photo2 VARCHAR(255),
    photo3 VARCHAR(255),
    photo4 VARCHAR(255),
    photo5 VARCHAR(255),
    client_note VARCHAR(255),
    status ENUM('pending', 'accepted', 'refused') DEFAULT 'pending',
    admin_note VARCHAR(255),
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (userid) REFERENCES Users(userid)
);

CREATE TABLE Work (
    workid INT AUTO_INCREMENT PRIMARY KEY,
    userid INT,
    quoteid INT,
    start_date DATE,
    end_date DATE,
    price FLOAT,
    client_status ENUM('pending', 'accepted', 'refused') DEFAULT 'pending',
    client_note VARCHAR(255),
    admin_status ENUM('pending', 'accepted', 'refused') DEFAULT 'pending',
    admin_note VARCHAR(255),
```

```
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (userid) REFERENCES Users(userid),
    FOREIGN KEY (quoteid) REFERENCES Quote(quoteid)
);

CREATE TABLE WorkHistory (
    historyid INT AUTO_INCREMENT PRIMARY KEY,
    workid INT,
    userid INT,
    quoteid INT,
    start_date DATE,
    end_date DATE,
    price FLOAT,
    client_status ENUM('pending', 'accepted', 'refused') DEFAULT 'pending',
    client_note VARCHAR(255),
    admin_status ENUM('pending', 'accepted', 'refused') DEFAULT 'pending',
    admin_note VARCHAR(255),
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (workid) REFERENCES Work(workid),
    FOREIGN KEY (userid) REFERENCES Users(userid)
);

CREATE TABLE Card (
    cardid INT AUTO_INCREMENT PRIMARY KEY, -- admin to see
    userid INT,
    nickname VARCHAR(100), -- user to see
    number VARCHAR(20),
    name VARCHAR(100),
    month TINYINT,
    year TINYINT,
    cvv INT(3),
    FOREIGN KEY (userid) REFERENCES Users(userid)
);

CREATE TABLE Transactions ( -- Orders and Bills
    transactionid INT AUTO_INCREMENT PRIMARY KEY,
    userid INT,
    quoteid INT,
    workid INT,
    cardid INT,
    price FLOAT,
    start_date DATE,
    end_date DATE,
```

```
    charge_status ENUM('pending', 'charge', 'deny') DEFAULT 'pending',
    client_note VARCHAR(255),
    pay_status ENUM('pending', 'paid', 'declined') DEFAULT 'pending',
    admin_note VARCHAR(255),
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (userid) REFERENCES Users(userid),
    FOREIGN KEY (quoteid) REFERENCES Quote(quoteid),
    FOREIGN KEY (workid) REFERENCES Work(workid),
    FOREIGN KEY (cardid) REFERENCES Card(cardid)
);

CREATE TABLE TransactionPaid (
    historyid INT AUTO_INCREMENT PRIMARY KEY,
    transactionid INT,
    userid INT,
    quoteid INT,
    workid INT,
    cardid INT,
    created_at DATETIME,
    paid_date DATETIME NULL,
    FOREIGN KEY (transactionid) REFERENCES Transactions(transactionid),
    FOREIGN KEY (userid) REFERENCES Users(userid),
    FOREIGN KEY (quoteid) REFERENCES Quote(quoteid),
    FOREIGN KEY (workid) REFERENCES Work(workid),
    FOREIGN KEY (cardid) REFERENCES Card(cardid)
);

INSERT INTO `Users` (`userid`, `email`, `password`, `firstname`, `lastname`, `address`,
`phone`, `role`) VALUES
(1, 'david_smith@gmail.com', '21232f297a57a5a743894a0e4a801fc3', 'David', 'Smith', '123
Maple St.', '2578467844', 'contractor'),
(2, 'dawlathamad@icloud.com', 'f6182f0359f72aae12fb90d305ccf9eb', 'Dawlat', 'Hamad', '683
Cherry Avenue', '3139709504', 'client'),
(3, 'ahmedsumaiya587@gmail.com', 'd7af994f1f1ef8b5e3beb9f7fb139f57', 'Sumaiya', 'Ahmed',
'826 Cherry Avenue', '4789463779', 'client'),
(4, 'janedoe1997@gmail.com', 'cd6c416546d256996e4941fe1170458e', 'Jane', 'Doe', '456 Sugar
Cane St', '3794125800', 'client'),
(5, 'john_doe_67@gmail.com', '64414f23baed90db1e20de4011131328', 'John', 'Doe', '456 Sugar
Cane St', '2347357556', 'client');

INSERT INTO `Quote` (`quoteid`, `userid`, `address`, `dimension_length`, `dimension_width`,
`price`, `photo1`, `photo2`, `photo3`, `photo4`, `photo5`, `client_note`, `status`, `admin_note`,
`created_at`) VALUES
```

```
 (1, 2, '683 Cherry Avenue', 12, 24, 25000, '1.png', '2.png', '3.png', '4.png', '5.png', 'Before the New Year', 'accepted', 'Okay', '2024-11-11 13:06:44'),
 (2, 3, '456 Sugar Cane St', 24, 24, 24000, '1.png', '2.png', '3.png', '4.png', '5.png', 'lala', 'accepted', '', '2024-11-19 16:12:36'),
 (3, 3, '356 Louis Lane', 25, 24, 50000, '1.png', '2.png', '3.png', '4.png', '5.png', 'lala', 'accepted', '', '2024-11-27 16:13:03'),
 (4, 3, '356 Louis Lane', 50, 24, 60000, '1.png', '2.png', '3.png', '4.png', '5.png', 'lala', 'accepted', '', '2024-12-11 16:13:40'),
 (5, 5, '456 Sugar Cane St', 24, 24, 24000, '1.png', '2.png', '3.png', '4.png', '5.png', 'Need Soon', 'accepted', '', '2024-12-14 16:56:15'),
 (6, 2, '932 Ice Cream St', 24, 24, 24000, '1.png', '2.png', '3.png', '4.png', '5.png', 'In March Please', 'accepted', '', '2024-12-15 17:18:45');

 INSERT INTO `Work` (`workid`, `userid`, `quoteid`, `start_date`, `end_date`, `price`, `client_status`, `client_note`, `admin_status`, `admin_note`, `created_at`) VALUES
 (1, 2, 1, '2024-12-23', '2024-12-30', 30000, 'accepted', 'I will pay more.', 'accepted', 'Faster shipping = more money', '2024-12-15 13:15:03'),
 (2, 3, 2, '2024-12-23', '2024-12-30', 24000, 'pending', NULL, 'pending', 'lala', '2024-12-15 16:15:02'),
 (3, 3, 3, '2024-12-23', '2024-12-23', 50000, 'pending', NULL, 'pending', 'lala', '2024-12-15 16:15:17'),
 (4, 3, 4, '2024-12-24', '2024-12-31', 75000, 'pending', NULL, 'pending', 'lala', '2024-12-15 16:15:44'),
 (5, 5, 5, '2024-12-17', '2024-12-23', 25000, 'accepted', 'Time is good.', 'accepted', 'Okay', '2024-12-15 16:57:00'),
 (6, 2, 6, '2025-03-01', '2025-03-15', 30000, 'accepted', 'Good dates.', 'accepted', 'Okay', '2024-12-15 17:19:12');

 INSERT INTO `WorkHistory` (`historyid`, `workid`, `userid`, `quoteid`, `start_date`, `end_date`, `price`, `client_status`, `client_note`, `admin_status`, `admin_note`, `created_at`) VALUES
 (1, 1, 2, 1, '2024-12-23', '2024-12-30', 30000, 'pending', NULL, 'pending', 'Faster shipping = more money', '2024-12-15 13:15:03'),
 (2, 1, 2, 1, '2024-12-23', '2024-12-30', 30000, 'accepted', 'I will pay more.', 'accepted', 'Faster shipping = more money', '2024-12-15 13:22:45'),
 (3, 2, 3, 2, '2024-12-23', '2024-12-30', 24000, 'pending', NULL, 'pending', 'lala', '2024-12-15 16:15:02'),
 (4, 3, 3, 3, '2024-12-23', '2024-12-23', 50000, 'pending', NULL, 'pending', 'lala', '2024-12-15 16:15:17'),
 (5, 4, 3, 4, '2024-12-24', '2024-12-31', 75000, 'pending', NULL, 'pending', 'lala', '2024-12-15 16:15:44'),
 (6, 5, 5, 5, '2024-12-17', '2024-12-23', 25000, 'pending', NULL, 'pending', 'Okay', '2024-12-15 16:57:00'),
```

```
 (7, 5, 5, 5, '2024-12-17', '2024-12-23', 25000, 'accepted', 'Time is good.', 'accepted', 'Okay',
'2024-12-15 16:58:39'),
 (8, 6, 2, 6, '2025-03-01', '2025-03-15', 30000, 'pending', NULL, 'pending', 'Okay', '2024-12-15
17:19:12'),
 (9, 6, 2, 6, '2025-03-01', '2025-03-15', 30000, 'accepted', 'Good dates.', 'accepted', 'Okay',
'2024-12-15 17:20:03'),
 (10, 6, 2, 6, '2025-03-01', '2025-03-15', 30000, 'accepted', 'Good dates.', 'pending', 'Okay',
'2024-12-15 17:20:11'),
 (11, 6, 2, 6, '2025-03-01', '2025-03-15', 30000, 'accepted', 'Good dates.', 'accepted', 'Okay',
'2024-12-15 17:25:27');

 INSERT INTO `Card` (`cardid`, `userid`, `nickname`, `number`, `name`, `month`, `year`, `cvv`)
VALUES
 (1, 1, "David's Card", '1234 1234 1234 1234', 'David Smith', 8, 25, 428),
 (2, 2, 'Dawlat Discover', '5678 5678 5678 5678', 'Dawlat Hamad', 11, 25, 748),
 (3, 3, "Sumaiya's Card", '1234 5678 1234 5678', 'Sumaiya Ahmed', 4, 25, 856),
 (4, 4, "Jane's Master", '4567 4567 4567 4567', 'Jane Doe', 9, 25, 387),
 (5, 5, 'John Discover', '6789 6789 6789 6789', 'John Doe', 3, 25, 835),
 (6, 2, 'Dawlat Master', '3678 4856 2903 4783', 'Dawlat Hamad', 4, 24, 875);

 INSERT INTO `Transactions` (`transactionid`, `userid`, `quoteid`, `workid`, `cardid`, `price`,
`start_date`, `end_date`, `charge_status`, `client_note`, `pay_status`, `admin_note`, `created_at`)
VALUES
 (1, 2, 1, 1, 2, 30000, '2024-12-23', '2024-12-30', 'charge', 'Charge', 'paid', 'Please Pay',
'2024-12-15 13:22:40'),
 (2, 5, 5, 5, NULL, 25000, '2024-12-17', '2024-12-23', 'pending', NULL, 'pending', 'Please Pay',
'2024-12-02 16:58:26'),
 (3, 2, 6, 6, 2, 30000, '2025-03-01', '2025-03-15', 'charge', 'Charge', 'paid', 'Please Pay.',
'2024-12-15 17:25:17');

 INSERT INTO `TransactionPaid` (`historyid`, `transactionid`, `userid`, `quoteid`, `workid`,
`cardid`, `created_at`, `paid_date`) VALUES
 (1, 3, 2, 6, 6, 2, '2024-12-15 17:25:17', '2024-12-15 17:26:23');
```

QUERY CODE:

[Big clients]

```
    SELECT
        u.userid,
        u.email,
        u.firstname,
        u.lastname,
        u.address,
        u.phone,
```

```
    u.role,
    COUNT(t.transactionid) AS transaction_count
FROM
    Users u
JOIN
    Transactions t
ON
    u.userid = t.userid
GROUP BY
    u.userid,
    u.email,
    u.firstname,
    u.lastname,
    u.address,
    u.phone,
    u.role
HAVING
    transaction_count = (
        SELECT MAX(transaction_count)
        FROM (
            SELECT COUNT(t.transactionid) AS transaction_count
            FROM Users u
            JOIN Transactions t
            ON u.userid = t.userid
            GROUP BY u.userid
        ) AS Counts
    );
```

[Difficult clients]

```
SELECT
    u.userid,
    u.email,
    u.firstname,
    u.lastname,
    u.address,
    u.phone,
    u.role,
    COUNT(w.workid) AS pending_work_count
FROM
    Users u
JOIN
    Work w
ON
```

```
        u.userid = w.userid
    WHERE
        w.client_status = 'pending'
    GROUP BY
        u.userid,
        u.email,
        u.firstname,
        u.lastname,
        u.address,
        u.phone,
        u.role
    HAVING
        COUNT(w.workid) >= 3;
```

[This month quotes]

```
    SELECT
        quoteid,
        userid,
        address,
        dimension_length,
        dimension_width,
        price,
        client_note,
        status,
        admin_note,
        created_at
    FROM
        Quote
    WHERE
        MONTH(created_at) = MONTH(CURRENT_DATE) AND YEAR(created_at) =
YEAR(CURRENT_DATE);
```

[Prospective clients]

```
SELECT
        u.userid,
        u.email,
        u.firstname,
        u.lastname,
        u.address,
        u.phone,
        u.role
```

```
    FROM
      Users u
    LEFT JOIN
      Quote q
    ON
      u.userid = q.userid
    WHERE
      q.userid IS NULL AND u.userid != 1; -- Exclude user with userid = 1
```

[Largest driveway]

```
    SELECT
      q.quoteid,
      q.userid,
      q.address,
      q.dimension_length,
      q.dimension_width,
      (q.dimension_length * q.dimension_width) AS area
    FROM
      Quote q
    WHERE
      (q.dimension_length * q.dimension_width) = (
        SELECT MAX(q1.dimension_length * q1.dimension_width)
        FROM Quote q1
      );
```

[Overdue bills]

```
    SELECT
      t.transactionid,
      t.quoteid,
      t.workid,
      t.userid,
      t.start_date,
      t.end_date,
      t.price,
      t.created_at
    FROM
      Transactions t
    WHERE
      t.charge_status = 'pending'
      AND t.created_at <= DATE_SUB(CURRENT_DATE, INTERVAL 7 DAY)
```

[Bad clients]

```
SELECT
   u.userid,
   u.email,
   u.firstname,
   u.lastname,
   u.address,
   u.phone,
   u.role,
   COUNT(t.transactionid) AS pending_transaction_count
FROM
   Users u
JOIN
   Transactions t
ON
   u.userid = t.userid
WHERE
   t.charge_status = 'pending'
   AND t.created_at <= DATE_SUB(CURRENT_DATE, INTERVAL 7 DAY)
GROUP BY
   u.userid,
   u.email,
   u.firstname,
   u.lastname,
   u.address,
   u.phone,
   u.role;
```

[Good clients]

```
SELECT
   u.userid,
   u.email,
   u.firstname,
   u.lastname,
   u.address,
   u.phone,
   u.role
FROM
   Users u
JOIN
   TransactionPaid t
ON
```

```
    u.userid = t.userid
WHERE
    t.paid_date <= DATE_ADD(t.created_at, INTERVAL 7 DAY)
GROUP BY
    u.userid,
    u.email,
    u.firstname,
    u.lastname,
    u.address,
    u.phone,
    u.role;
```