Project part 3

December 12, 2023

Database Management Systems I CSC_6710_2309_001

Amanda Stead – Working Alone

YouTube link: https://youtu.be/q3BJLX93Yd8

GitHub repo: https://github.com/AmandaStead/databaseProject

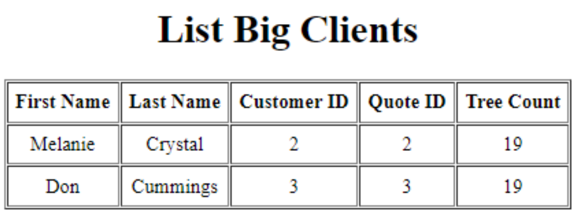
How to run project configurations located in Github

SQL Statements:

2. [Big clients]: List the clients that David Smith cuts the most number of trees. List one client if there is no tie, list all the top clients if there is a tie. Please verify your query result is correct.

```
public List<BigClients> BigClients() throws SQLException {
      List<BigClients> BigClients = new ArrayList<>();
       String sql = "SELECT\r\n"
                      + "    u.firstname,\r\n"
                      + "    u.lastname,\r\n"
                      + "    u.customerid,\r\n"
                      + "    q.quoteid,\r\n"
                      + "    q.customerid AS quote_customerid,\r\n"
                      + "    q.tree_count\r\n"
                      + "FROM\r\n"
                      + "    user u\r\n"
                      + "JOIN\r\n"
                      + "    quote q ON u.customerid = q.customerid\r\n"
                      + "JOIN\r\n"
                      + "    tree t ON q.quoteid = t.quoteid\r\n"
                      + "WHERE\r\n"
                      + "    q.tree_count = (\r\n"
                      + "        SELECT MAX(tree_count)\r\n"
                      + "        FROM quote\r\n"
                      + "    );";
```

Solution:

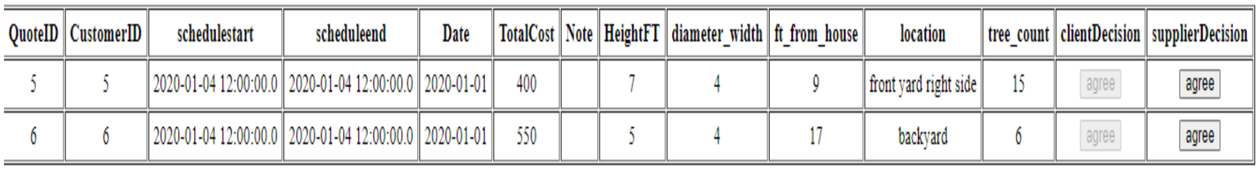


List Big Clients

First Name	Last Name	Customer ID	Quote ID	Tree Count
Melanie	Crystal	2	2	19
Don	Cummings	3	3	19

3. [Easy clients]: List the clients who simply accept David Smith's initial quotes without further negotiations.

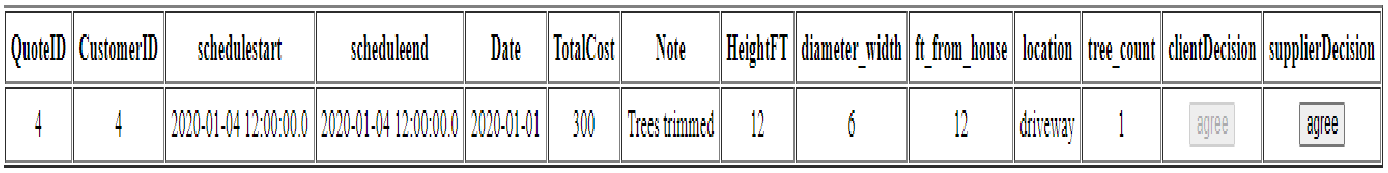
```
public List<quote> EasyClients() throws SQLException {
        List<quote> listquote = new ArrayList<quote>();
        String sql = "SELECT *\r\n"
                    + "FROM quote\r\n"
                    + "WHERE (custnote IS NULL OR custnote = '')\r\n"
                    + "  AND clientdecision = 'agree'\r\n"
                    + "  AND supplierdecision = 'agree';";
```



QuoteID	CustomerID	schedulestart	scheduleend	Date	TotalCost	Note	HeightFT	diameter_width	ft_from_house	location	tree_count	clientDecision	supplierDecision
5	5	2020-01-04 12:00:00.0	2020-01-04 12:00:00.0	2020-01-01	400		7	4	9	front yard right side	15	agree	agree
6	6	2020-01-04 12:00:00.0	2020-01-04 12:00:00.0	2020-01-01	550		5	4	17	backyard	6	agree	agree

4. [One tree quotes]. List all the AGREED quotes that only involved one tree.

```
public List<quote> Onetreequotes() throws SQLException {
        List<quote> listquote = new ArrayList<quote>();
        String sql = "SELECT *\r\n"
                    + "FROM quote\r\n"
                    + "WHERE tree_count = 1 and clientdecision = 'agree' and 
supplierdecision = 'agree';";
```



QuoteID	CustomerID	schedulestart	scheduleend	Date	TotalCost	Note	HeightFT	diameter_width	ft_from_house	location	tree_count	clientDecision	supplierDecision
4	4	2020-01-04 12:00:00.0	2020-01-04 12:00:00.0	2020-01-01	300	Trees trimmed	12	6	12	driveway	1	agree	agree

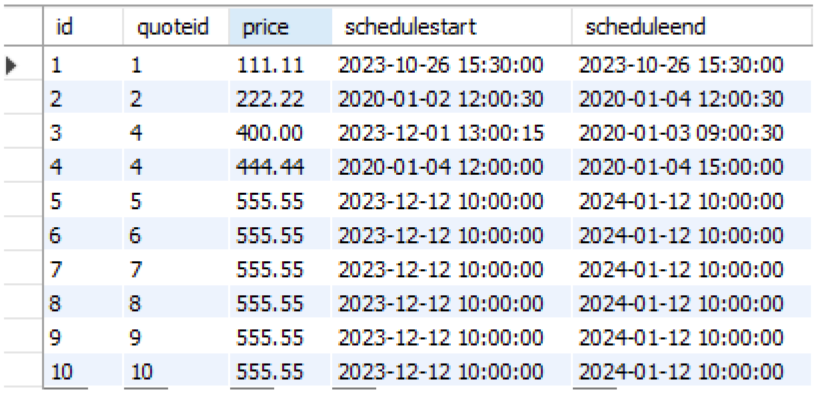
5. [Prospective clients]: List all the clients that submitted some quotes but never produced any orders of work.

```
public List<BigClients> Prospectiveclients() throws SQLException {
       List<BigClients> Prospectiveclients = new ArrayList<>();
        String sql = "SELECT user.customerid, user.firstname, user.lastname\r\n"
                    + "FROM user user\r\n"
                    + "WHERE user.customerid IN (\r\n"
                    + "    SELECT DISTINCT quote.customerid\r\n"
                    + "    FROM quote \r\n"
                    + "    WHERE quote.customerid IS NOT NULL\r\n"
                    + "    AND quote.quoteid NOT IN (\r\n"
                    + "        SELECT DISTINCT orderofwork.quoteid\r\n"
                    + "        FROM orderofwork\r\n"
                    + "    )\r\n"
                    + "    );";
```

List all Prospective clients

quoteid	customerid	firstname	lastname
3	3	Don	Cummings

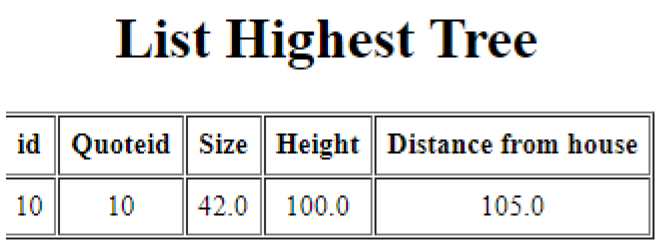
CustomerID = QuoteID for orderofwork table



id	quoteid	price	schedulestart	scheduleend
1	1	111.11	2023-10-26 15:30:00	2023-10-26 15:30:00
2	2	222.22	2020-01-02 12:00:30	2020-01-04 12:00:30
3	4	400.00	2023-12-01 13:00:15	2020-01-03 09:00:30
4	4	444.44	2020-01-04 12:00:00	2020-01-04 15:00:00
5	5	555.55	2023-12-12 10:00:00	2024-01-12 10:00:00
6	6	555.55	2023-12-12 10:00:00	2024-01-12 10:00:00
7	7	555.55	2023-12-12 10:00:00	2024-01-12 10:00:00
8	8	555.55	2023-12-12 10:00:00	2024-01-12 10:00:00
9	9	555.55	2023-12-12 10:00:00	2024-01-12 10:00:00
10	10	555.55	2023-12-12 10:00:00	2024-01-12 10:00:00

6. [Highest tree]: List the highest tree that David Smith ever cut, list all the trees if there is a tie or list just the one tree if there is no tie. (only listed the trees that have been cut)

```
public List<tree> HighestTree() throws SQLException {
        List<tree> HighestTree = new ArrayList<>();
         String sql = "SELECT t.*\r\n"
                    + "FROM tree t\r\n"
                    + "WHERE t.height = (\r\n"
                    + "    SELECT MAX(height)\r\n"
                    + "    FROM tree\r\n"
                    + ");\r\n"
                    + "";
```

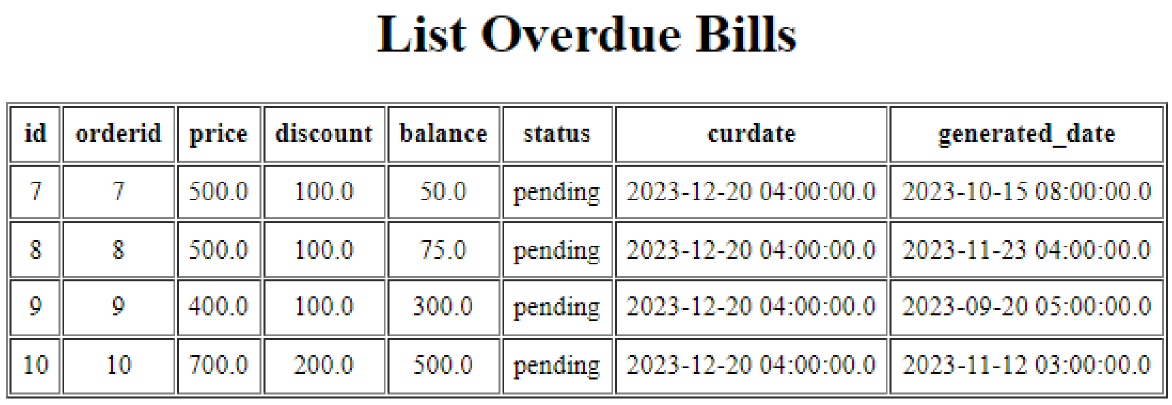


List Highest Tree

id	Quoteid	Size	Height	Distance from house
10	10	42.0	100.0	105.0

7. [Overdue bills]. List all the bills that have not been paid after one week the bill is generated.

```
public List<quote> Bills() throws SQLException {
        List<quote> Bills = new ArrayList<quote>();
        String sql = "SELECT *\r\n"
                   + "FROM bills\r\n"
                   + "WHERE status = 'pending'\r\n"
                   + "      AND DATEDIFF(CURDATE(), generated_date) > 7;\r\n"
                   + "\r\n"
                   + "\r\n"
                   + "";
```



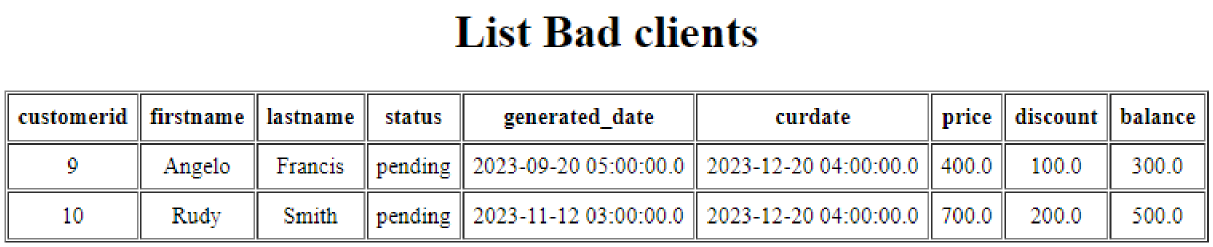
List Overdue Bills

id	orderid	price	discount	balance	status	curdate	generated_date
7	7	500.0	100.0	50.0	pending	2023-12-20 04:00:00.0	2023-10-15 08:00:00.0
8	8	500.0	100.0	75.0	pending	2023-12-20 04:00:00.0	2023-11-23 04:00:00.0
9	9	400.0	100.0	300.0	pending	2023-12-20 04:00:00.0	2023-09-20 05:00:00.0
10	10	700.0	200.0	500.0	pending	2023-12-20 04:00:00.0	2023-11-12 03:00:00.0

8. [Bad clients]. List all the clients that have never paid any bill after it is due. Suppose the bill is due one week after it is generated. A client is not bad if there is no bill for them at all.

```
public List<quote> BadClients() throws SQLException {
        List<quote> BadClients = new ArrayList<quote>();
        String sql =  "SELECT user.customerid, user.firstname, user.lastname, 
bills.status,bills.generated_date, 
bills.curdate,bills.price,bills.discount,bills.balance \r\n"
```

```
            + "FROM user JOIN quote ON user.customerid = quote.customerid JOIN 
orderofwork ON quote.quoteid = orderofwork.quoteid JOIN \r\n"
               +"bills ON orderofwork.quoteid = bills.orderid WHERE bills.status = 
'pending' AND bills.generated_date < curdate() - INTERVAL 7 DAY AND bills.price - 
bills.discount = bills.balance \r\n"
            + "GROUP BY user.customerid, user.firstname, user.lastname, 
bills.status, bills.generated_date, 
bills.curdate,bills.price,bills.discount,bills.balance"
            + "\r\n"
            + "\r\n"
            + "";
```

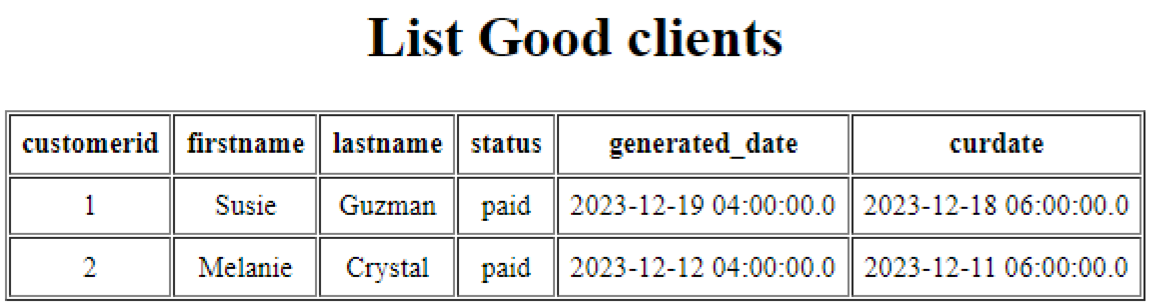


List Bad clients

customerid	firstname	lastname	status	generated_date	curdate	price	discount	balance
9	Angelo	Francis	pending	2023-09-20 05:00:00.0	2023-12-20 04:00:00.0	400.0	100.0	300.0
10	Rudy	Smith	pending	2023-11-12 03:00:00.0	2023-12-20 04:00:00.0	700.0	200.0	500.0

9. [Good clients]. List all the clients that have paid their bills right after the bills are generated, meaning they paid the bill within 24 hours when the bill is generated.

```
public List<quote> GoodClients() throws SQLException {
        List<quote> GoodClients = new ArrayList<quote>();
        String sql = "SELECT user.customerid, user.firstname, 
user.lastname,bills.status,bills.generated_date,bills.curdate \r\n"
                  + "FROM user \r\n"
                  + "JOIN quote ON user.customerid = quote.customerid \r\n"
                  + "JOIN orderofwork ON quote.quoteid = orderofwork.quoteid \r\n"
                  + "JOIN bills ON orderofwork.quoteid = bills.orderid \r\n"
                  + "WHERE bills.status = 'paid' \r\n"
                  + "AND bills.generated_date >= DATE_SUB(NOW(), INTERVAL 1 
DAY)\r\n"
                  + "GROUP BY user.customerid, user.firstname, 
user.lastname,bills.status,bills.generated_date,bills.curdate \r\n"
                  + "\r\n"
                  + "\r\n"
                  + "";
```

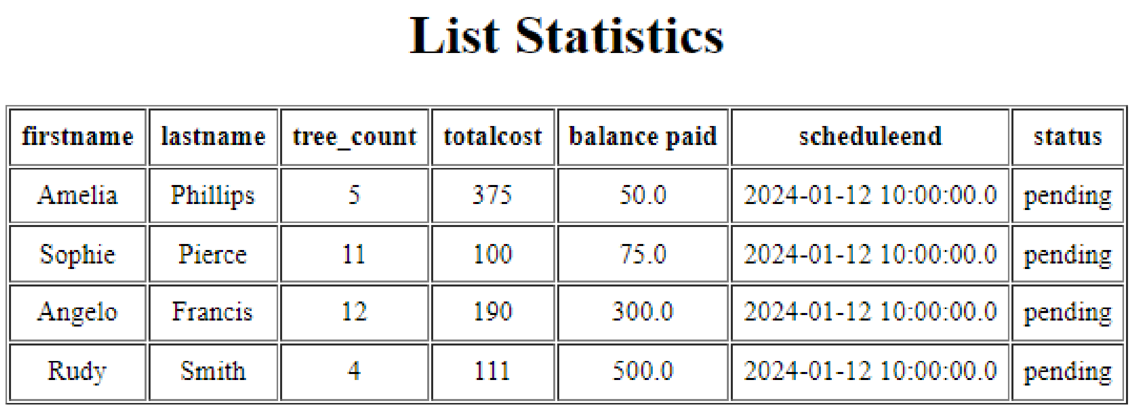


List Good clients

customerid	firstname	lastname	status	generated_date	curdate
1	Susie	Guzman	paid	2023-12-19 04:00:00.0	2023-12-18 06:00:00.0
2	Melanie	Crystal	paid	2023-12-12 04:00:00.0	2023-12-11 06:00:00.0

10. [Statistics] For each client, list the total numbers of trees, total due amount, total paid amount, and the date when the work is done for each tree. Only list the trees that have been cut.

```
public List<quote> Statistics() throws SQLException {
        List<quote> Statistics = new ArrayList<quote>();
        String sql = "SELECT\r\n"
                     + "     user.firstname,\r\n"
                     + "     user.lastname,\r\n"
                     + "     quote.tree_count,\r\n"
                     + "     quote.totalcost,\r\n"
                     + "     bills.balance,\r\n"
                     + "     orderofwork.scheduleend,\r\n"
                     + "     bills.status\r\n"
                     + "FROM\r\n"
                     + "     bills\r\n"
                     + "JOIN\r\n"
                     + "     orderofwork ON bills.orderid = orderofwork.quoteid\r\n"
                     + "JOIN\r\n"
                     + "     quote ON orderofwork.quoteid = quote.quoteid\r\n"
                     + "JOIN\r\n"
                     + "     user ON quote.customerID = user.customerID\r\n"
                     + "WHERE\r\n"
                     + "     bills.status = 'pending'"
                     + "\r\n"
                     + "\r\n"
                     + "";
```



List Statistics

firstname	lastname	tree_count	totalcost	balance paid	scheduleend	status
Amelia	Phillips	5	375	50.0	2024-01-12 10:00:00.0	pending
Sophie	Pierce	11	100	75.0	2024-01-12 10:00:00.0	pending
Angelo	Francis	12	190	300.0	2024-01-12 10:00:00.0	pending
Rudy	Smith	4	111	500.0	2024-01-12 10:00:00.0	pending