```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Scanner;

public class DatabaseManager {
    private static final String DB_URL = "jdbc:mysql://localhost:3306/mydatabase";
    private static final String USER = "user";
    private static final String PASS = "password";

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int option;
        do {
            System.out.println("\n--- Database Management System ---");
            System.out.println("1. Add User");
            System.out.println("2. Update User");
            System.out.println("3. Delete User");
            System.out.println("4. View All Users");
            System.out.println("5. Add Role");
            System.out.println("6. Assign Role to User");
            System.out.println("7. Remove Role from User");
            System.out.println("8. List All Roles");
            System.out.println("9. Search User by Name");
            System.out.println("10. Search User by Email");
            System.out.println("11. Exit");
            System.out.print("Choose an option: ");
            option = scanner.nextInt();
            switch (option) {
                case 1:
                    addUser(scanner);
                    break;
                case 2:
                    updateUser(scanner);
                    break;
                case 3:
                    deleteUser(scanner);
                    break;
                case 4:
                    viewUsers();
                    break;
                case 5:
                    addRole(scanner);
                    break;
                case 6:
                    assignRole(scanner);
                    break;
                case 7:
                    removeRole(scanner);
                    break;
                case 8:
                    listRoles();
                    break;
                case 9:
```

```
                    searchUserByName(scanner);
                    break;
                case 10:
                    searchUserByEmail(scanner);
                    break;
                case 11:
                    System.out.println("Exiting the program...");
                    break;
                default:
                    System.out.println("Invalid option. Please enter a valid
number.");
            }
        } while (option != 11);
        scanner.close();
    }

    private static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(DB_URL, USER, PASS);
    }

    private static void addUser(Scanner scanner) {
        System.out.print("Enter user name: ");
        scanner.nextLine(); // clear buffer
        String name = scanner.nextLine();
        System.out.print("Enter email: ");
        String email = scanner.next();
        String sql = "INSERT INTO users (name, email) VALUES (?, ?)";
        try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql)) {
            conn.setAutoCommit(false);
            pstmt.setString(1, name);
            pstmt.setString(2, email);
            executeUpdate(pstmt, "User added successfully!");
            conn.commit();
        } catch (SQLException e) {
            System.out.println("Error adding user: " + e.getMessage());
        }
    }

    private static void updateUser(Scanner scanner) {
        System.out.print("Enter user id to update: ");
        int id = scanner.nextInt();
        scanner.nextLine(); // clear buffer
        System.out.print("Enter new email: ");
        String email = scanner.nextLine();
        String sql = "UPDATE users SET email = ? WHERE id = ?";
        try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql)) {
            conn.setAutoCommit(false);
            pstmt.setString(1, email);
            pstmt.setInt(2, id);
            executeUpdate(pstmt, "User updated successfully!");
            conn.commit();
        } catch (SQLException e) {
            System.out.println("Error updating user: " + e.getMessage());
        }
```

```
110     }
111
112     private static void deleteUser(Scanner scanner) {
113         System.out.print("Enter user id to delete: ");
114         int id = scanner.nextInt();
115         String sql = "DELETE FROM users WHERE id = ?";
116         try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql)) {
117             conn.setAutoCommit(false);
118             pstmt.setInt(1, id);
119             executeUpdate(pstmt, "User deleted successfully!");
120             conn.commit();
121         } catch (SQLException e) {
122             System.out.println("Error deleting user: " + e.getMessage());
123         }
124     }
125
126     private static void viewUsers() {
127         String sql = "SELECT * FROM users";
128         try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql); ResultSet rs = pstmt.executeQuery()) {
129             System.out.println("List of all users:");
130             while (rs.next()) {
131                 int id = rs.getInt("id");
132                 String name = rs.getString("name");
133                 String email = rs.getString("email");
134                 System.out.printf("ID: %d, Name: %s, Email: %s\n", id, name, email);
135             }
136         } catch (SQLException e) {
137             System.out.println("Error retrieving users: " + e.getMessage());
138         }
139     }
140
141     private static void addRole(Scanner scanner) {
142         System.out.print("Enter role name: ");
143         scanner.nextLine(); // clear buffer
144         String roleName = scanner.nextLine();
145         String sql = "INSERT INTO roles (role_name) VALUES (?)";
146         try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql)) {
147             conn.setAutoCommit(false);
148             pstmt.setString(1, roleName);
149             executeUpdate(pstmt, "Role added successfully!");
150             conn.commit();
151         } catch (SQLException e) {
152             System.out.println("Error adding role: " + e.getMessage());
153         }
154     }
155
156     private static void assignRole(Scanner scanner) {
157         System.out.print("Enter user id for role assignment: ");
158         int userId = scanner.nextInt();
159         System.out.print("Enter role id to assign: ");
160         int roleId = scanner.nextInt();
161         String sql = "INSERT INTO user_roles (user_id, role_id) VALUES (?, ?)";
```

```
        try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql)) {
            conn.setAutoCommit(false);
            pstmt.setInt(1, userId);
            pstmt.setInt(2, roleId);
            executeUpdate(pstmt, "Role assigned to user successfully!");
            conn.commit();
        } catch (SQLException e) {
            System.out.println("Error assigning role: " + e.getMessage());
        }
    }

    private static void removeRole(Scanner scanner) {
        System.out.print("Enter user id for role removal: ");
        int userId = scanner.nextInt();
        System.out.print("Enter role id to remove: ");
        int roleId = scanner.nextInt();
        String sql = "DELETE FROM user_roles WHERE user_id = ? AND role_id = ?";
        try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql)) {
            conn.setAutoCommit(false);
            pstmt.setInt(1, userId);
            pstmt.setInt(2, roleId);
            executeUpdate(pstmt, "Role removed from user successfully!");
            conn.commit();
        } catch (SQLException e) {
            System.out.println("Error removing role: " + e.getMessage());
        }
    }

    private static void listRoles() {
        String sql = "SELECT * FROM roles";
        try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql); ResultSet rs = pstmt.executeQuery()) {
            System.out.println("List of all roles:");
            while (rs.next()) {
                int roleId = rs.getInt("role_id");
                String roleName = rs.getString("role_name");
                System.out.printf("Role ID: %d, Role Name: %s\n", roleId, roleName);
            }
        } catch (SQLException e) {
            System.out.println("Error listing roles: " + e.getMessage());
        }
    }

    private static void searchUserByName(Scanner scanner) {
        System.out.print("Enter user name to search: ");
        scanner.nextLine(); // clear buffer
        String name = scanner.nextLine();
        String sql = "SELECT * FROM users WHERE name LIKE ?";
        try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql)) {
            pstmt.setString(1, "%" + name + "%");
            try (ResultSet rs = pstmt.executeQuery()) {
                System.out.println("Search results:");
                while (rs.next()) {
```

```
                    int id = rs.getInt("id");
                    String email = rs.getString("email");
                    System.out.printf("ID: %d, Name: %s, Email: %s\n", id, name,
email);
                }
            }
        } catch (SQLException e) {
            System.out.println("Error searching for user: " + e.getMessage());
        }
    }

    private static void searchUserByEmail(Scanner scanner) {
        System.out.print("Enter email to search for: ");
        scanner.nextLine(); // clear buffer
        String email = scanner.nextLine();
        String sql = "SELECT * FROM users WHERE email LIKE ?";
        try (Connection conn = getConnection(); PreparedStatement pstmt =
conn.prepareStatement(sql)) {
            pstmt.setString(1, "%" + email + "%");
            try (ResultSet rs = pstmt.executeQuery()) {
                System.out.println("Search results:");
                while (rs.next()) {
                    int id = rs.getInt("id");
                    String name = rs.getString("name");
                    System.out.printf("ID: %d, Name: %s, Email: %s\n", id, name,
email);
                }
            }
        } catch (SQLException e) {
            System.out.println("Error searching for email: " + e.getMessage());
        }
    }

    private static void executeUpdate(PreparedStatement pstmt, String
successMessage) throws SQLException {
        int affectedRows = pstmt.executeUpdate();
        if (affectedRows > 0) {
            System.out.println(successMessage);
            System.out.println("Operation was successful.");
        } else {
            System.out.println("Operation failed, no changes were made.");
        }
    }
}
```

Report Banner - Edit rsm.cfg File

Resource Standard Metrics™ for C, C++, C# and Java
Version 7.75 - mSquaredTechnologies.com

License Type: Shareware Evaluation License
Licensed To : Shareware End User - Distribute Freely
License No. : SW1380 License Date: Dec 05, 1998
Build Date : Sep 2 2009 Run Date: Apr 22, 2024

License File: C:\Program Files (x86)\MSquared\M2 RSM\rsm.lic
Config. File: C:\Program Files (x86)\MSquared\M2 RSM\rsm.cfg
Command Line: -H -OC:\Users\DELL\M2 RSM Wizard\output\complexity_metrics_report_for_databasemanager_java_code.htm -c -fd -FC:\Users\DELL\M2 RSM Wizard\input\rsm_file_list_for_databasemaneger_java_code.lst

~~ Function Metrics ~~
~~ Complexity Detail Analysis ~~

File: C:\Program Files (x86)\MSquared\M2 RSM Wizard\example code\DatabaseManager\DatabaseManager.java

---

Function: DatabaseManager.main
Parameters: (String[] args)
Cyclomatic Complexity Vg Detail
Function Base : 1
Loops while / do : 1
Selection case : 11

Complexity	Param 1	Return 1	Cyclo Vg 13	Total	15
LOC 58	eLOC 56	lLOC 41	Comment 0	Lines	58

Function: DatabaseManager.getConnection
Parameters: ()

Complexity	Param 0	Return 1	Cyclo Vg 1	Total	2
LOC 3	eLOC 2	lLOC 1	Comment 0	Lines	3

Function: DatabaseManager.addUser
Parameters: (Scanner scanner)

Complexity	Param 1	Return 1	Cyclo Vg 1	Total	3
LOC 14	eLOC 14	lLOC 12	Comment 1	Lines	14

Function: DatabaseManager.updateUser
Parameters: (Scanner scanner)

Complexity	Param 1	Return 1	Cyclo Vg 1	Total	3
LOC 14	eLOC 14	lLOC 12	Comment 1	Lines	14

Function: DatabaseManager.deleteUser
Parameters: (Scanner scanner)

Complexity	Param 1	Return 1	Cyclo Vg 1	Total	3
LOC 10	eLOC 10	lLOC 8	Comment 0	Lines	10

Function: DatabaseManager.viewUsers
Parameters: ()
Cyclomatic Complexity Vg Detail
Function Base : 1
Loops while / do : 1

Complexity	Param 0	Return 1	Cyclo Vg 2	Total	3
LOC 11	eLOC 10	lLOC 8	Comment 0	Lines	11

Function: DatabaseManager.addRole
Parameters: (Scanner scanner)

Complexity	Param 1	Return 1	Cyclo Vg 1	Total	3
LOC 11	eLOC 11	lLOC 9	Comment 1	Lines	11

Function: DatabaseManager.assignRole
Parameters: (Scanner scanner)

Complexity	Param 1	Return 1	Cyclo Vg 1	Total	3

```
LOC 13       eLOC 13       lLOC 11       Comment 0        Lines       13

Function: DatabaseManager.removeRole
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 13       eLOC 13       lLOC 11       Comment 0        Lines       13

Function: DatabaseManager.listRoles
Parameters: ()
  Cyclomatic Complexity Vg Detail
    Function Base           : 1
    Loops while / do        : 1
Complexity   Param 0       Return 1      Cyclo Vg 2       Total        3
LOC 10       eLOC 9        lLOC 7        Comment 0        Lines       10

Function: DatabaseManager.searchUserByName
Parameters: (Scanner scanner)
  Cyclomatic Complexity Vg Detail
    Function Base           : 1
    Loops while / do        : 1
Complexity   Param 1       Return 1      Cyclo Vg 2       Total        4
LOC 15       eLOC 13       lLOC 10       Comment 1        Lines       15

Function: DatabaseManager.searchUserByEmail
Parameters: (Scanner scanner)
  Cyclomatic Complexity Vg Detail
    Function Base           : 1
    Loops while / do        : 1
Complexity   Param 1       Return 1      Cyclo Vg 2       Total        4
LOC 15       eLOC 13       lLOC 10       Comment 1        Lines       15

Function: DatabaseManager.executeUpdate
Parameters: (PreparedStatement pstmt, String successMessage)
  Cyclomatic Complexity Vg Detail
    Function Base           : 1
    Conditional if / else if: 1
Complexity   Param 2       Return 1      Cyclo Vg 2       Total        5
LOC 9        eLOC 7        lLOC 4        Comment 0        Lines        9

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

                      ~~ Total File Summary ~~

LOC 239      eLOC 207      lLOC 163      Comment 5        Lines      253
--------------------------------------------------------------------------

                    ~~ File Functional Summary ~~

File Function Count....:         13
Total Function LOC.....:        196  Total Function Pts LOC :        4.5
Total Function eLOC....:        185  Total Function Pts eLOC:        3.9
Total Function lLOC....:        144  Total Function Pts lLOC:        3.1
Total Function Params .:         11  Total Function Return .:         13
Total Cyclo Complexity :         30  Total Function Complex.:         54
       ------     -----     -----     ------     ------     -----
Max Function LOC ......:         58  Average Function LOC ..:      15.08
Max Function eLOC .....:         56  Average Function eLOC .:      14.23
Max Function lLOC .....:         41  Average Function lLOC .:      11.08
       ------     -----     -----     ------     ------     -----
Max Function Parameters:          2  Avg Function Parameters:       0.85
Max Function Returns ..:          1  Avg Function Returns ..:       1.00
Max Interface Complex. :          3  Avg Interface Complex. :       1.85
Max Cyclomatic Complex.:         13  Avg Cyclomatic Complex.:       2.31
Max Total Complexity ..:         15  Avg Total Complexity ..:       4.15

--------------------------------------------------------------------------
End of File: C:\Program Files (x86)\MSquared\M2 RSM Wizard\example code\
             DatabaseManager\DatabaseManager.java



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

```
                    ~~ Total Metrics For 1 Files ~~
-----------------------------------------------------------------------

                     ~~ Project Directory Summary ~~

Files 1 - C:\Program Files (x86)\MSquared\M2 RSM Wizard\example code\DatabaseManager
LOC 239      eLOC 207      lLOC 163      Comment 5        Lines      253

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

                       ~~ Total Project Summary ~~

LOC 239      eLOC 207      lLOC 163      Comment 5        Lines      253
Average per File, metric/1 files
LOC 239      eLOC 207      lLOC 163      Comment 5        Lines      253

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

                     ~~ Project Functional Metrics ~~

Function: DatabaseManager.main
Parameters: (String[] args)
Complexity   Param 1       Return 1      Cyclo Vg 13      Total       15
LOC 58       eLOC 56       lLOC 41       Comment 0        Lines       58

Function: DatabaseManager.getConnection
Parameters: ()
Complexity   Param 0       Return 1      Cyclo Vg 1       Total        2
LOC 3        eLOC 2        lLOC 1        Comment 0        Lines        3

Function: DatabaseManager.addUser
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 14       eLOC 14       lLOC 12       Comment 1        Lines       14

Function: DatabaseManager.updateUser
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 14       eLOC 14       lLOC 12       Comment 1        Lines       14

Function: DatabaseManager.deleteUser
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 10       eLOC 10       lLOC 8        Comment 0        Lines       10

Function: DatabaseManager.viewUsers
Parameters: ()
Complexity   Param 0       Return 1      Cyclo Vg 2       Total        3
LOC 11       eLOC 10       lLOC 8        Comment 0        Lines       11

Function: DatabaseManager.addRole
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 11       eLOC 11       lLOC 9        Comment 1        Lines       11

Function: DatabaseManager.assignRole
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 13       eLOC 13       lLOC 11       Comment 0        Lines       13

Function: DatabaseManager.removeRole
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 13       eLOC 13       lLOC 11       Comment 0        Lines       13

Function: DatabaseManager.listRoles
Parameters: ()
Complexity   Param 0       Return 1      Cyclo Vg 2       Total        3
LOC 10       eLOC 9        lLOC 7        Comment 0        Lines       10

Function: DatabaseManager.searchUserByName
Parameters: (Scanner scanner)
```

```
Complexity   Param 1       Return 1      Cyclo Vg 2       Total        4
LOC 15       eLOC 13       lLOC 10       Comment 1        Lines       15

Function: DatabaseManager.searchUserByEmail
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 2       Total        4
LOC 15       eLOC 13       lLOC 10       Comment 1        Lines       15

Function: DatabaseManager.executeUpdate
Parameters: (PreparedStatement pstmt, String successMessage)
Complexity   Param 2       Return 1      Cyclo Vg 2       Total        5
LOC 9        eLOC 7        lLOC 4        Comment 0        Lines        9

Total: Functions
LOC 196      eLOC 185      lLOC 144      InCmp 24       CycloCmp      30

Function Points         FP(LOC) 3.7     FP(eLOC) 3.5    FP(lLOC)     2.7

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

                    ~~ Project Functional Analysis ~~

Total Functions .......:         13  Total Physical Lines ..:        196
Total LOC .............:        196  Total Function Pts LOC :        3.7
Total eLOC ............:        185  Total Function Pts eLOC:        3.5
Total lLOC.............:        144  Total Function Pts lLOC:        2.7
Total Cyclomatic Comp. :         30  Total Interface Comp. .:         24
Total Parameters ......:         11  Total Return Points ...:         13
Total Comment Lines ...:          5  Total Blank Lines .....:          0
       ------     -----     -----      ------     ------     -----
Avg Physical Lines ....:      15.08
Avg LOC ...............:      15.08  Avg eLOC ..............:      14.23
Avg lLOC ..............:      11.08  Avg Cyclomatic Comp. ..:       2.31
Avg Interface Comp. ...:       1.85  Avg Parameters ........:       0.85
Avg Return Points .....:       1.00  Avg Comment Lines .....:       0.38
       ------     -----     -----      ------     ------     -----
Max LOC ...............:         58
Max eLOC ..............:         56  Max lLOC ..............:         41
Max Cyclomatic Comp. ..:         13  Max Interface Comp. ...:          3
Max Parameters ........:          2  Max Return Points .....:          1
Max Comment Lines .....:          1  Max Total Lines .......:         58
       ------     -----     -----      ------     ------     -----
Min LOC ...............:          3
Min eLOC ..............:          2  Min lLOC ..............:          1
Min Cyclomatic Comp. ..:          1  Min Interface Comp. ...:          1
Min Parameters ........:          0  Min Return Points .....:          1
Min Comment Lines .....:          0  Min Total Lines .......:          3

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

                          ~~ File Summary ~~

C Source Files *.c ....:          0  C/C++ Include Files *.h:          0
C++ Source Files *.c* .:          0  C++ Include Files *.h* :          0
C# Source Files *.cs ..:          0  Java Source File *.jav*:          1
Other Source Files ....:          0
Total File Count ......:          1


        Shareware evaluation licenses process only 20 files.
        Paid licenses enable processing for an unlimited number of files.
```

Report Banner - Edit rsm.cfg File

Report Banner - Edit rsm.cfg File

Resource Standard Metrics™ for C, C++, C# and Java
Version 7.75 - mSquaredTechnologies.com

License Type: Shareware Evaluation License
Licensed To : Shareware End User - Distribute Freely
License No. : SW1380 License Date: Dec 05, 1998
Build Date : Sep 2 2009 Run Date: Apr 22, 2024

License File: C:\Program Files (x86)\MSquared\M2 RSM\rsm.lic
Config. File: C:\Program Files (x86)\MSquared\M2 RSM\rsm.cfg
Command Line: -H -OC:\Users\DELL\M2 RSM Wizard\output\functional_quality_metrics_report_for_databasemanager_java_code.htm -c -fa -n -o -SFunctional Quality Metrics -FC:\Users\DELL\M2 RSM Wizard\input\rsm_file_list_for_databasemaneger_java_code.lst
UDQN File : C:\Program Files (x86)\MSquared\M2 RSM\rsm_udqn.cfg

Functional Quality Metrics

~~ Function Metrics ~~
~~ Class/Struct Metrics ~~
~~ Complexity Analysis ~~
~~ Quality Analysis ~~

File: C:\Program Files (x86)\MSquared\M2 RSM Wizard\example code\DatabaseManager\DatabaseManager.java

---

---------- Class Begin Line: 8 ----------
Class: DatabaseManager

Notice #52: Line 8: A class has been identified which does not have a preceding comment. Comments that detail the purpose, algorithms, and parameter/return definitions are suggested.

Notice #1: Line 9: Line character length = 82. This width exceeds the standard terminal width of 80 characters.

---------- Function Begin Line: 13 ----------
Function: DatabaseManager.main
Parameters: (String[] args)

Notice #51: Line 13: A function has been identified which does not have a preceding comment. Comments that detail the purpose, algorithms, and parameter/return definitions are suggested.

Notice #1: Line 66: Line character length = 87. This width exceeds the standard terminal width of 80 characters.

Notice #17: Function comments, 0.0% are less than 10.0%.

Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Notice #28: The function cyclomatic complexity of 13 exceeds the specified limit of 10.

Function: DatabaseManager.main
Complexity Param 1 Return 1 Cyclo Vg 13 Total 15
LOC 58 eLOC 56 lLOC 41 Comment 0 Lines 58
---------- Function End Line: 70 ----------

---------- Function Begin Line: 72 ----------
Function: DatabaseManager.getConnection
Parameters: ()

```
  Notice #51: Line 72: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #49: The function contains no input parameters or void.
         Suggest using explict parameters for interface clarity.

Function: DatabaseManager.getConnection
Complexity   Param 0       Return 1      Cyclo Vg 1       Total        2
LOC 3        eLOC 2        lLOC 1        Comment 0        Lines        3
------------------------ Function End Line: 74 -----------------------

------------------------ Function Begin Line: 76 -----------------------
Function: DatabaseManager.addUser
Parameters: (Scanner scanner)

  Notice #51: Line 76: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #1: Line 83: Line character length = 103.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 83: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #17: Function comments, 6.7% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Function: DatabaseManager.addUser
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 14       eLOC 14       lLOC 12       Comment 1        Lines       14
------------------------ Function End Line: 89 -----------------------

------------------------ Function Begin Line: 94 -----------------------
Function: DatabaseManager.updateUser
Parameters: (Scanner scanner)

  Notice #51: Line 94: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #1: Line 101: Line character length = 103.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 101: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #17: Function comments, 6.7% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Function: DatabaseManager.updateUser
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 14       eLOC 14       lLOC 12       Comment 1        Lines       14
------------------------ Function End Line: 107 -----------------------

------------------------ Function Begin Line: 112 -----------------------
Function: DatabaseManager.deleteUser
Parameters: (Scanner scanner)

  Notice #51: Line 112: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.
```

```
  Notice #1: Line 116: Line character length = 103.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 116: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #17: Function comments, 0.0% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Function: DatabaseManager.deleteUser
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 10       eLOC 10       lLOC 8        Comment 0        Lines       10
----------------------- Function End Line: 121 -----------------------

---------------------- Function Begin Line: 126 ----------------------
Function: DatabaseManager.viewUsers
Parameters: ()

  Notice #51: Line 126: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #1: Line 128: Line character length = 140.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 128: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #1: Line 134: Line character length = 84.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #17: Function comments, 0.0% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

  Notice #49: The function contains no input parameters or void.
         Suggest using explict parameters for interface clarity.

Function: DatabaseManager.viewUsers
Complexity   Param 0       Return 1      Cyclo Vg 2       Total        3
LOC 11       eLOC 10       lLOC 8        Comment 0        Lines       11
----------------------- Function End Line: 136 -----------------------

---------------------- Function Begin Line: 141 ----------------------
Function: DatabaseManager.addRole
Parameters: (Scanner scanner)

  Notice #51: Line 141: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #1: Line 146: Line character length = 103.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 146: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #17: Function comments, 8.3% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Function: DatabaseManager.addRole
```

```
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 11       eLOC 11       lLOC 9        Comment 1        Lines       11
----------------------- Function End Line: 151 -----------------------

---------------------- Function Begin Line: 156 ----------------------
Function: DatabaseManager.assignRole
Parameters: (Scanner scanner)

  Notice #51: Line 156: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #1: Line 162: Line character length = 103.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 162: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #17: Function comments, 0.0% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Function: DatabaseManager.assignRole
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 13       eLOC 13       lLOC 11       Comment 0        Lines       13
----------------------- Function End Line: 168 -----------------------

---------------------- Function Begin Line: 173 ----------------------
Function: DatabaseManager.removeRole
Parameters: (Scanner scanner)

  Notice #51: Line 173: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #1: Line 179: Line character length = 103.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 179: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #17: Function comments, 0.0% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Function: DatabaseManager.removeRole
Complexity   Param 1       Return 1      Cyclo Vg 1       Total        3
LOC 13       eLOC 13       lLOC 11       Comment 0        Lines       13
----------------------- Function End Line: 185 -----------------------

---------------------- Function Begin Line: 190 ----------------------
Function: DatabaseManager.listRoles
Parameters: ()

  Notice #51: Line 190: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #1: Line 192: Line character length = 140.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 192: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.
```

```
  Notice #1: Line 197: Line character length = 84.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #17: Function comments, 0.0% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

  Notice #49: The function contains no input parameters or void.
         Suggest using explict parameters for interface clarity.

Function: DatabaseManager.listRoles
Complexity   Param 0       Return 1      Cyclo Vg 2       Total        3
LOC 10       eLOC 9        lLOC 7        Comment 0        Lines       10
----------------------- Function End Line: 199 -----------------------

---------------------- Function Begin Line: 204 ----------------------
Function: DatabaseManager.searchUserByName
Parameters: (Scanner scanner)

  Notice #51: Line 204: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #1: Line 209: Line character length = 103.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 209: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #38: Line 211: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #1: Line 216: Line character length = 88.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #17: Function comments, 6.3% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Function: DatabaseManager.searchUserByName
Complexity   Param 1       Return 1      Cyclo Vg 2       Total        4
LOC 15       eLOC 13       lLOC 10       Comment 1        Lines       15
----------------------- Function End Line: 218 -----------------------

---------------------- Function Begin Line: 224 ----------------------
Function: DatabaseManager.searchUserByEmail
Parameters: (Scanner scanner)

  Notice #51: Line 224: A function has been identified which does not
         have a preceding comment.  Comments that detail the purpose,
         algorithms, and parameter/return definitions are suggested.

  Notice #1: Line 229: Line character length = 103.  This width
         exceeds the standard terminal width of 80 characters.

  Notice #38: Line 229: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
         upon the code.

  Notice #38: Line 231: Exception handling 'try' - 'catch' has been
         identified.  Exception handling can be a form indirect logic
         control similar to a goto.  The use of exception handling
         should be driven by the design and not casually imposed
```

```
        upon the code.

  Notice #1: Line 236: Line character length = 88.  This width
        exceeds the standard terminal width of 80 characters.

  Notice #17: Function comments, 6.3% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Function: DatabaseManager.searchUserByEmail
Complexity   Param 1       Return 1      Cyclo Vg 2       Total        4
LOC 15       eLOC 13       lLOC 10       Comment 1        Lines       15
----------------------- Function End Line: 238 -----------------------

  Notice #1: Line 244: Line character length = 107.  This width
        exceeds the standard terminal width of 80 characters.

----------------------- Function Begin Line: 244 -----------------------
Function: DatabaseManager.executeUpdate
Parameters: (PreparedStatement pstmt, String successMessage)

  Notice #51: Line 244: A function has been identified which does not
        have a preceding comment.  Comments that detail the purpose,
        algorithms, and parameter/return definitions are suggested.

  Notice #17: Function comments, 0.0% are less than 10.0%.

  Notice #46: Function blank line percent, 0.0% is less than 10.0%.

Function: DatabaseManager.executeUpdate
Complexity   Param 2       Return 1      Cyclo Vg 2       Total        5
LOC 9        eLOC 7        lLOC 4        Comment 0        Lines        9
----------------------- Function End Line: 252 -----------------------

  Notice #31: Class comments, 2.1% are less than 10.0%.

  Notice #46: Class blank line percent, 5.3% is less than 10.0%.

Class: DatabaseManager
Attributes   Publ 0        Prot 0        Private 3        Total        3
Methods      Publ 1        Prot 0        Private 22       Total       23
LOC 233      eLOC 201      lLOC 157      Comment 5        Lines      245
------------------------- Class End Line: 253 -------------------------

  Notice #47: File blank line percentage, 5.9% is less than 10.0%

  Notice #20: File comment line percentage, 2.0% is less than 10.0%
--------------------------------------------------------------------------

                      ~~ Total File Summary ~~

LOC 239      eLOC 207      lLOC 163      Comment 5        Lines      253
--------------------------------------------------------------------------

                    ~~ File Functional Summary ~~

File Function Count....:         13
Total Function LOC.....:        196  Total Function Pts LOC :        4.5
Total Function eLOC....:        185  Total Function Pts eLOC:        3.9
Total Function lLOC....:        144  Total Function Pts lLOC:        3.1
Total Function Params .:         11  Total Function Return .:         13
Total Cyclo Complexity :         30  Total Function Complex.:         54
       ------     -----     -----     ------     ------     -----
Max Function LOC ......:         58  Average Function LOC ..:      15.08
Max Function eLOC .....:         56  Average Function eLOC .:      14.23
Max Function lLOC .....:         41  Average Function lLOC .:      11.08
       ------     -----     -----     ------     ------     -----
Max Function Parameters:          2  Avg Function Parameters:       0.85
Max Function Returns ..:          1  Avg Function Returns ..:       1.00
Max Interface Complex. :          3  Avg Interface Complex. :       1.85
Max Cyclomatic Complex.:         13  Avg Cyclomatic Complex.:       2.31
Max Total Complexity ..:         15  Avg Total Complexity ..:       4.15
```

End of File: C:\Program Files (x86)\MSquared\M2 RSM Wizard\example code\DatabaseManager\DatabaseManager.java

---

~~ Total Metrics For 1 Files ~~

~~ Project Directory Summary ~~

Files 1 - C:\Program Files (x86)\MSquared\M2 RSM Wizard\example code\DatabaseManager
LOC 239 eLOC 207 lLOC 163 Comment 5 Lines 253

~~ Total Project Summary ~~

LOC 239 eLOC 207 lLOC 163 Comment 5 Lines 253
Average per File, metric/1 files
LOC 239 eLOC 207 lLOC 163 Comment 5 Lines 253

~~ Project Functional Metrics ~~

Function: DatabaseManager.main
Parameters: (String[] args)
Complexity 15 Param 1 Return 1 Cyclo Vg 13 Total 15
LOC 58 eLOC 56 lLOC 41 Comment 0 Lines 58

Function: DatabaseManager.getConnection
Parameters: ()
Complexity 2 Param 0 Return 1 Cyclo Vg 1 Total 2
LOC 3 eLOC 2 lLOC 1 Comment 0 Lines 3

Function: DatabaseManager.addUser
Parameters: (Scanner scanner)
Complexity 3 Param 1 Return 1 Cyclo Vg 1 Total 3
LOC 14 eLOC 14 lLOC 12 Comment 1 Lines 14

Function: DatabaseManager.updateUser
Parameters: (Scanner scanner)
Complexity 3 Param 1 Return 1 Cyclo Vg 1 Total 3
LOC 14 eLOC 14 lLOC 12 Comment 1 Lines 14

Function: DatabaseManager.deleteUser
Parameters: (Scanner scanner)
Complexity 3 Param 1 Return 1 Cyclo Vg 1 Total 3
LOC 10 eLOC 10 lLOC 8 Comment 0 Lines 10

Function: DatabaseManager.viewUsers
Parameters: ()
Complexity 3 Param 0 Return 1 Cyclo Vg 2 Total 3
LOC 11 eLOC 10 lLOC 8 Comment 0 Lines 11

Function: DatabaseManager.addRole
Parameters: (Scanner scanner)
Complexity 3 Param 1 Return 1 Cyclo Vg 1 Total 3
LOC 11 eLOC 11 lLOC 9 Comment 1 Lines 11

Function: DatabaseManager.assignRole
Parameters: (Scanner scanner)
Complexity 3 Param 1 Return 1 Cyclo Vg 1 Total 3
LOC 13 eLOC 13 lLOC 11 Comment 0 Lines 13

Function: DatabaseManager.removeRole
Parameters: (Scanner scanner)
Complexity 3 Param 1 Return 1 Cyclo Vg 1 Total 3
LOC 13 eLOC 13 lLOC 11 Comment 0 Lines 13

```
Function: DatabaseManager.listRoles
Parameters: ()
Complexity   Param 0       Return 1      Cyclo Vg 2       Total        3
LOC 10       eLOC 9        lLOC 7        Comment 0        Lines       10

Function: DatabaseManager.searchUserByName
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 2       Total        4
LOC 15       eLOC 13       lLOC 10       Comment 1        Lines       15

Function: DatabaseManager.searchUserByEmail
Parameters: (Scanner scanner)
Complexity   Param 1       Return 1      Cyclo Vg 2       Total        4
LOC 15       eLOC 13       lLOC 10       Comment 1        Lines       15

Function: DatabaseManager.executeUpdate
Parameters: (PreparedStatement pstmt, String successMessage)
Complexity   Param 2       Return 1      Cyclo Vg 2       Total        5
LOC 9        eLOC 7        lLOC 4        Comment 0        Lines        9

Total: Functions
LOC 196      eLOC 185      lLOC 144      InCmp 24       CycloCmp      30

Function Points         FP(LOC) 3.7     FP(eLOC) 3.5    FP(lLOC)     2.7

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

                  ~~ Project Functional Analysis ~~

Total Functions .......:         13  Total Physical Lines ..:        196
Total LOC .............:        196  Total Function Pts LOC :        3.7
Total eLOC ............:        185  Total Function Pts eLOC:        3.5
Total lLOC.............:        144  Total Function Pts lLOC:        2.7
Total Cyclomatic Comp. :         30  Total Interface Comp. .:         24
Total Parameters ......:         11  Total Return Points ...:         13
Total Comment Lines ...:          5  Total Blank Lines .....:          0
       ------     -----     -----     ------     ------     -----
Avg Physical Lines ....:      15.08
Avg LOC ...............:      15.08  Avg eLOC ..............:      14.23
Avg lLOC ..............:      11.08  Avg Cyclomatic Comp. ..:       2.31
Avg Interface Comp. ...:       1.85  Avg Parameters ........:       0.85
Avg Return Points .....:       1.00  Avg Comment Lines .....:       0.38
       ------     -----     -----     ------     ------     -----
Max LOC ...............:         58
Max eLOC ..............:         56  Max lLOC ..............:         41
Max Cyclomatic Comp. ..:         13  Max Interface Comp. ...:          3
Max Parameters ........:          2  Max Return Points .....:          1
Max Comment Lines .....:          1  Max Total Lines .......:         58
       ------     -----     -----     ------     ------     -----
Min LOC ...............:          3
Min eLOC ..............:          2  Min lLOC ..............:          1
Min Cyclomatic Comp. ..:          1  Min Interface Comp. ...:          1
Min Parameters ........:          0  Min Return Points .....:          1
Min Comment Lines .....:          0  Min Total Lines .......:          3

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

                  ~~ Project Class/Struct Metrics ~~

Class: DatabaseManager
Attributes   Publ 0        Prot 0        Private 3        Total        3
Methods      Publ 1        Prot 0        Private 22       Total       23
Complexity   Param 11      Return 13     Cyclo Vg 30      Total       54
LOC 233      eLOC 201      lLOC 157      Comment 5        Lines      245

Total: All Classes/Structs
Attributes   Publ 0        Prot 0        Private 3        Total        3
Methods      Publ 1        Prot 0        Private 22       Total       23
Complexity   Param 11      Return 13     Cyclo Vg 30      Total       54
LOC 233      eLOC 201      lLOC 157      Comment 5        Lines      245
```

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

                ~~ Project Class/Struct Analysis ~~

Total Classes/Structs .:          1  Total Methods .........:         23
Total Public Methods ..:          1  Total Public Attributes:          0
Total Protected Methods:          0  Total Protected Attrib.:          0
Total Private Methods .:         22  Total Private Attrib. .:          3
Total Physical Lines ..:        245  Total LOC .............:        233
Total eLOC ............:        201  Total lLOC ............:        157
Total Cyclomatic Comp. :         30  Total Interface Comp. .:         24
Total Parameters ......:         11  Total Return Points ...:         13
Total Comment Lines ...:          5  Total Blank Lines .....:         13
       ------     -----     -----     ------     ------     -----
Avg Physical Lines ....:     245.00  Avg Methods ...........:      23.00
Avg Public Methods ....:       1.00  Avg Public Attributes .:       0.00
Avg Protected Methods .:       0.00  Avg Protected Attrib. .:       0.00
Avg Private Methods ...:      22.00  Avg Private Attributes :       3.00
Avg LOC ...............:     233.00  Avg eLOC ..............:     201.00
Avg lLOC ..............:     157.00  Avg Cyclomatic Comp. ..:      30.00
Avg Interface Comp. ...:      24.00  Avg Parameters ........:      11.00
Avg Return Points .....:      13.00  Avg Comment Lines .....:       5.00
       ------     -----     -----     ------     ------     -----
Max Physical Lines ....:        245  Max Methods ...........:         23
Max Public Methods ....:          1  Max Public Attributes .:          0
Max Protected Methods .:          0  Max Protected Attrib. .:          0
Max Private Methods ...:         22  Max Private Attributes :          3
Max LOC ...............:        233  Max eLOC ..............:        201
Max lLOC ..............:        157  Max Cyclomatic Comp. ..:         30
Max Interface Comp. ...:         24  Max Parameters ........:         11
Max Return Points .....:         13  Max Comment Lines .....:          5
       ------     -----     -----     ------     ------     -----
Min Physical Lines ....:        245  Min Methods ...........:         23
Min Public Methods ....:          1  Min Public Attributes .:          0
Min Protected Methods .:          0  Min Protected Attrib. .:          0
Min Private Methods ...:         22  Min Private Attributes :          3
Min LOC ...............:        233  Min eLOC ..............:        201
Min lLOC ..............:        157  Min Cyclomatic Comp. ..:         30
Min Interface Comp. ...:         24  Min Parameters ........:         11
Min Return Points .....:         13  Min Comment Lines .....:          5

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

                   ~~ Project Quality Profile ~~

Type  Count Percent  Quality Notice
-----------------------------------------------------------------

1        17   22.67  Physical line length > 80 characters
17       12   16.00  Function comment content less than 10.0%
20        1    1.33  File comment content < 10.0%
28        1    1.33  Cyclomatic complexity > 10
31        1    1.33  Class/Struct comments are < 10.0%
38       12   16.00  Exception Handling "try"- "catch" has been identified
46       13   17.33  Function/Class Blank Line content less < 10.0%
47        1    1.33  File Blank Line content < 10.0%
49        3    4.00  Function appears to have null or blank parameters
51       13   17.33  No comment preceding a function block
52        1    1.33  No comment preceding a class block
-----------------------------------------------------------------

         75  100.00  Total Quality Notices

                   ~~ Quality Notice Density ~~

Basis: 1000 (K)

Quality Notices/K LOC   =      313.8 ( 31.38%)
Quality Notices/K eLOC  =      362.3 ( 36.23%)
Quality Notices/K lLOC  =      460.1 ( 46.01%)

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

~~ File Summary ~~

C Source Files *.c ....:	0	C/C++ Include Files *.h:	0
C++ Source Files *.c* .:	0	C++ Include Files *.h* :	0
C# Source Files *.cs ..:	0	Java Source File *.jav*:	1
Other Source Files ....:	0		
Total File Count ......:	1		

Shareware evaluation licenses process only 20 files.
Paid licenses enable processing for an unlimited number of files.

Report Banner - Edit rsm.cfg File