**Documentation for the Database/System: Hederlige Harrys Bilar**

1. **System Overview**  
   The "Honest Harry's Assortment" system is a database solution for managing users, logins, password resets, and roles. The system is designed to ensure that user data is secure and that users can manage their accounts effectively. The system includes the following main components:  
   • **User Management**: Create, update, and manage user accounts.  
   • **Login and Authentication**: Manage login attempts, verification, and account locking.  
   • **Password Reset**: Allows users to reset their passwords via email.  
   • **Role Management**: Administrators can update user roles (Customer or Admin).  
   • **Reporting**: Generate reports on login attempts and user activity.
2. **Table Descriptions**  
   **Users**  
   • **Description**: Stores information about users.  
   • **Columns**:  
   o **UserID**: Unique ID for each user (primary key).  
   o **Email**: User's email address (unique).  
   o **PasswordHash**: Hashed password.  
   o **SALT**: Salt used to strengthen password hashing.  
   o **FirstName**: User's first name.  
   o **LastName**: User's last name.  
   o **Address**: User's address.  
   o **PostalCode**: Postal code.  
   o **City**: City.  
   o **Country**: Country.  
   o **Role**: User's role (Customer or Admin).  
   o **PhoneNumber**: Phone number.  
   o **ValidTo**: Date when the account expires.  
   o **IsLocked**: Indicates if the account is locked (1 = locked, 0 = unlocked).

**LoginAttempts**  
• **Description**: Logs all login attempts.  
• **Columns**:  
o **AttemptID**: Unique ID for each login attempt (primary key).  
o **UserID**: User's ID (foreign key to Users).  
o **Email**: Email address used for the attempt.  
o **IpAddress**: IP address from which the attempt was made.  
o **IsSuccessful**: Indicates whether the attempt was successful (1 = successful, 0 = failed).  
o **AttemptTime**: Timestamp of the attempt.

**PasswordReset**  
• **Description**: Manages password resets.  
• **Columns**:  
o **ResetID**: Unique ID for each reset request (primary key).  
o **UserID**: User's ID (foreign key to Users).  
o **ResetCode**: Unique reset code.  
o **ExpiryTime**: Timestamp when the code expires.  
o **Expired**: Indicates if the code has expired (1 = expired, 0 = valid).

**EmailVerification**  
• **Description**: Manages email verification for new accounts.  
• **Columns**:  
o **VerificationID**: Unique ID for each verification request (primary key).  
o **UserID**: User's ID (foreign key to Users).  
o **VerificationCode**: Unique verification code.  
o **ExpiryTime**: Timestamp when the code expires.  
o **IsVerified**: Indicates if the email address is verified (1 = verified, 0 = not verified).

1. **Stored Procedures and Views**  
   **Stored Procedures**  
   • **CreateAccount**: Creates a new user account and sends a verification link via email.  
   • **ResetPassword**: Triggers a password reset and sends a reset link via email.  
   • **SetForgottenPassword**: Updates the user's password after a successful reset.  
   • **LockAccount**: Locks a user account.  
   • **UnLockAccount**: Unlocks a user account.  
   • **LoginAttempt**: Manages login attempts and logs them.  
   • **UpdateUserRole**: Updates a user's role (for administrators only).

**Views**  
• **LoginAttemptInfo**: Displays the most recent successful and unsuccessful login for each user.  
• **AttemptsPerIpAddress**: Displays the number of login attempts per IP address, including both successful and failed attempts, along with the average successful frequency.

1. **Optimization and Performance**  
   **Optimization**  
   • **Indexing**: All primary keys and foreign keys are indexed to speed up searches and joins.  
   • **Batch Updates**: Usage of SET NOCOUNT ON to reduce unnecessary network traffic.  
   • **Hashing**: Passwords are hashed with SHA-256 and a unique salt for each user to ensure security.  
   • **Caching**: Use of variables to reduce database load.  
   • **Window Functions**: Usage of ROW\_NUMBER and OVER to create efficient reports without losing detailed information.

**Future Improvements**  
• **Improved Error Handling**: Add clear error messages to all stored procedures to facilitate troubleshooting and provide better user feedback in case of problems.  
• **Automatic Cleanup of Old Data**: Create a scheduled job that regularly removes old login attempts, expired password reset requests, and unused verification codes. This will keep the database clean and efficient.  
• **Improved Security**: Add two-factor authentication (2FA) for administrators or users who want additional security on their accounts.

**Performance Testing**  
• **Load Testing**: Test how the system performs when many users are active at the same time to assess its handling of high loads.

1. **Testing and Debugging**  
   • **Unit Testing**: Test each stored procedure and function individually to ensure they work correctly.  
   • **Integration Testing**: Test entire flows, such as from account creation to login and password reset.

**Conclusion**  
The system is designed to be both secure and efficient. By using best practices to design and optimize the database, the system can handle many users and transactions.