SQL Connection

[Skip to content](https://chatgpt.com/c/68778ca6-12e4-8002-82c8-8e0bdc8fdf42#main)

**Chat history**

**You said:**

Microsoft.Data.SqlClient.SqlException (0x80131904): Login failed for user 'sa'. at Microsoft.Data.SqlClient.TdsParser.ThrowExceptionAndWarning(TdsParserStateObject stateObj, Boolean callerHasConnectionLock, Boolean asyncClose) at Microsoft.Data.SqlClient.TdsParser.TryRun(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj, Boolean& dataReady) at Microsoft.Data.SqlClient.TdsParser.Run(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.CompleteLogin(Boolean enlistOK) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.LoginNoFailover(ServerInfo serverInfo, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance, SqlConnectionString connectionOptions, SqlCredential credential, TimeoutTimer timeout) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.OpenLoginEnlist(TimeoutTimer timeout, SqlConnectionString connectionOptions, SqlCredential credential, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance) at Microsoft.Data.SqlClient.SqlInternalConnectionTds..ctor(DbConnectionPoolIdentity identity, SqlConnectionString connectionOptions, SqlCredential credential, Object providerInfo, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance, SqlConnectionString userConnectionOptions, SessionData reconnectSessionData, Boolean applyTransientFaultHandling, String accessToken, DbConnectionPool pool) at Microsoft.Data.SqlClient.SqlConnectionFactory.CreateConnection(DbConnectionOptions options, DbConnectionPoolKey poolKey, Object poolGroupProviderInfo, DbConnectionPool pool, DbConnection owningConnection, DbConnectionOptions userOptions) at Microsoft.Data.ProviderBase.DbConnectionPool.CreateObject(DbConnection owningObject, DbConnectionOptions userOptions, DbConnectionInternal oldConnection) at Microsoft.Data.ProviderBase.DbConnectionPool.TryGetConnection(DbConnection owningObject, UInt32 waitForMultipleObjectsTimeout, Boolean allowCreate, Boolean onlyOneCheckConnection, DbConnectionOptions userOptions, DbConnectionInternal& connection) at Microsoft.Data.ProviderBase.DbConnectionPool.TryGetConnection(DbConnection owningObject, TaskCompletionSource1 retry, DbConnectionOptions userOptions, DbConnectionInternal& connection) at Microsoft.Data.ProviderBase.DbConnectionFactory.TryGetConnection(DbConnection owningConnection, TaskCompletionSource1 retry, DbConnectionOptions userOptions, DbConnectionInternal oldConnection, DbConnectionInternal& connection) at Microsoft.Data.ProviderBase.DbConnectionInternal.TryOpenConnectionInternal(DbConnection outerConnection, DbConnectionFactory connectionFactory, TaskCompletionSource1 retry, DbConnectionOptions userOptions) at Microsoft.Data.SqlClient.SqlConnection.TryOpen(TaskCompletionSource1 retry, SqlConnectionOverrides overrides) at Microsoft.Data.SqlClient.SqlConnection.Open(SqlConnectionOverrides overrides) at Microsoft.EntityFrameworkCore.Storage.RelationalConnection.OpenInternal(Boolean errorsExpected) at Microsoft.EntityFrameworkCore.Storage.RelationalConnection.Open(Boolean errorsExpected) at Microsoft.EntityFrameworkCore.SqlServer.Storage.Internal.SqlServerDatabaseCreator.<>c\_\_DisplayClass18\_0.<Exists>b\_\_0(DateTime giveUp) at Microsoft.EntityFrameworkCore.ExecutionStrategyExtensions.<>c\_\_DisplayClass12\_02.<Execute>b\_\_0(DbContext \_, TState s) at Microsoft.EntityFrameworkCore.SqlServer.Storage.Internal.SqlServerExecutionStrategy.Execute[TState,TResult](TState state, Func3 operation, Func3 verifySucceeded) at Microsoft.EntityFrameworkCore.ExecutionStrategyExtensions.Execute[TState,TResult](IExecutionStrategy strategy, TState state, Func2 operation, Func2 verifySucceeded) at Microsoft.EntityFrameworkCore.SqlServer.Storage.Internal.SqlServerDatabaseCreator.Exists(Boolean retryOnNotExists) at Microsoft.EntityFrameworkCore.SqlServer.Storage.Internal.SqlServerDatabaseCreator.Exists() at Microsoft.EntityFrameworkCore.Migrations.Internal.Migrator.Migrate(String targetMigration) at Microsoft.EntityFrameworkCore.Design.Internal.MigrationsOperations.UpdateDatabase(String targetMigration, String connectionString, String contextType) at Microsoft.EntityFrameworkCore.Design.OperationExecutor.UpdateDatabaseImpl(String targetMigration, String connectionString, String contextType) at Microsoft.EntityFrameworkCore.Design.OperationExecutor.UpdateDatabase.<>c\_\_DisplayClass0\_0.<.ctor>b\_\_0() at Microsoft.EntityFrameworkCore.Design.OperationExecutor.OperationBase.Execute(Action action) ClientConnectionId:e7f18cc6-92bb-4257-8a69-fff8f0d1ea4a Error Number:18456,State:1,Class:14 Login failed for user 'sa'. (base) Saurabh@

**ChatGPT said:**

The error message you're seeing:

Microsoft.Data.SqlClient.SqlException (0x80131904): Login failed for user 'sa'.

Error Number:18456,State:1,Class:14

means that the SQL Server login for the user sa has failed. Here's how you can troubleshoot and resolve it:

**🔍 Root Cause**

The SQL Server is rejecting the connection attempt due to one of the following:

1. **Incorrect password** for user sa
2. **SQL Server authentication mode** is not configured correctly
3. sa**login is disabled**
4. **SQL Server is unreachable**
5. **Firewall or network issue**

**✅ Steps to Fix**

**✅ 1. Check Username and Password**

* Double-check the password you're using for the sa user.
* Try connecting via **SQL Server Management Studio (SSMS)** using the same credentials.

**✅ 2. Check SQL Server Authentication Mode**

SQL Server supports two authentication modes:

* **Windows Authentication**
* **Mixed Mode Authentication** (Windows + SQL Server Logins)

If you're using sa, you **must** enable **Mixed Mode**.

To verify:

1. Open **SSMS**
2. Right-click the server → **Properties**
3. Go to **Security** tab
4. Ensure **SQL Server and Windows Authentication mode** is selected

**✅ 3. Enable the**sa**Login**

Sometimes the sa account is disabled by default.

To enable:

1. In SSMS → Go to **Security > Logins**
2. Right-click on sa → **Properties**
3. Go to **Status**
4. Set **Login: Enabled**
5. Set/Reset the password if needed

ALTER LOGIN sa ENABLE; ALTER LOGIN sa WITH PASSWORD = 'YourStrongPassword';

**✅ 4. Restart SQL Server**

After changing authentication mode or enabling login, **restart the SQL Server**:

services.msc → SQL Server (MSSQLSERVER) → Restart

**✅ 5. Check Connection String**

Ensure your connection string in .NET or appsettings.json is correct:

"ConnectionStrings": { "DefaultConnection": "Server=localhost;Database=YourDB;User Id=sa;Password=YourPassword;" }

**✅ 6. Firewall or Port Issues**

* Ensure SQL Server is listening on port **1433**
* If remote connection: check firewall or SQL Server's TCP/IP settings in **SQL Server Configuration Manager**

**🛠 Verify with a Manual SQL Connection**

Try logging in manually using SQLCMD or SSMS:

sqlcmd -S localhost -U sa -P YourPassword

If this fails, it confirms a login or configuration issue.

**ℹ️ Extra Info from the Error**

* Error Number: 18456: SQL Server login failed
* State: 1: Generic error – check SQL Server logs for more precise cause (different states give more info)

If you'd like help modifying your EF Core configuration or .NET connection string, feel free to share the relevant snippet.

**You said:**

Microsoft.Data.SqlClient.SqlException (0x80131904): Login failed for user 'sa'. at Microsoft.Data.SqlClient.TdsParser.ThrowExceptionAndWarning(TdsParserStateObject stateObj, Boolean callerHasConnectionLock, Boolean asyncClose) at Microsoft.Data.SqlClient.TdsParser.TryRun(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj, Boolean& dataReady) at Microsoft.Data.SqlClient.TdsParser.Run(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.CompleteLogin(Boolean enlistOK) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.LoginNoFailover(ServerInfo serverInfo, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance, SqlConnectionString connectionOptions, SqlCredential credential, TimeoutTimer timeout) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.OpenLoginEnlist(TimeoutTimer timeout, SqlConnectionString connectionOptions, SqlCredential credential, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance) at Microsoft.Data.SqlClient.SqlInternalConnectionTds..ctor(DbConnectionPoolIdentity identity, SqlConnectionString connectionOptions, SqlCredential credential, Object providerInfo, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance, SqlConnectionString userConnectionOptions, SessionData reconnectSessionData, Boolean applyTransientFaultHandling, String accessToken, DbConnectionPool pool) at Microsoft.Data.SqlClient.SqlConnectionFactory.CreateConnection(DbConnectionOptions options, DbConnectionPoolKey poolKey, Object poolGroupProviderInfo, DbConnectionPool pool, DbConnection owningConnection, DbConnectionOptions userOptions) at Microsoft.Data.ProviderBase.DbConnectionPool.CreateObject(DbConnection owningObject, DbConnectionOptions userOptions, DbConnectionInternal oldConnection) at Microsoft.Data.ProviderBase.DbConnectionPool.TryGetConnection(DbConnection owningObject, UInt32 waitForMultipleObjectsTimeout, Boolean allowCreate, Boolean onlyOneCheckConnection, DbConnectionOptions userOptions, DbConnectionInternal& connection) at Microsoft.Data.ProviderBase.DbConnectionPool.WaitForPendingOpen() --- End of stack trace from previous location --- at Microsoft.Data.SqlClient.SqlRetryLogicProvider.ExecuteAsync(Object sender, Func1 function, CancellationToken cancellationToken) at Microsoft.Data.SqlClient.SqlRetryLogicProvider.ExecuteAsync(Object sender, Func1 function, CancellationToken cancellationToken) at Microsoft.SqlTools.ServiceLayer.Connection.ReliableConnection.ReliableSqlConnection.<>c\_\_DisplayClass31\_0.<<OpenAsync>b\_\_0>d.MoveNext() in /\_/src/Microsoft.SqlTools.ManagedBatchParser/ReliableConnection/ReliableSqlConnection.cs:line 342 --- End of stack trace from previous location --- at Microsoft.SqlTools.ServiceLayer.Connection.ConnectionService.TryOpenConnection(ConnectionInfo connectionInfo, ConnectParams connectionParams) in /\_/src/Microsoft.SqlTools.ServiceLayer/Connection/ConnectionService.cs:line 712 ClientConnectionId:a987933c-c2ae-4348-971a-0d51d895991d Error Number:18456,State:1,Class:14

**ChatGPT said:**

You're again seeing the SqlException: Login failed for user 'sa' with error number 18456, State:1, Class:14. This confirms the login is failing due to an authentication issue.

Let’s break it down and guide you toward a fix.

**🔎 Summary of the Error**

* **Error Number: 18456** → SQL Server login failure.
* **State: 1** → General failure (no detailed reason shown).
* **User: 'sa'** → SQL Server system administrator account.
* **Cause**: Typically one of the following:
  + Wrong password
  + sa login is disabled
  + SQL Server is not in Mixed Mode Authentication
  + Connection string error

**✅ Step-by-Step Fix**

**🔧 Step 1: Confirm the Correct Password**

Try logging in using SSMS:

Server name: localhost or . or 127.0.0.1 Authentication: SQL Server Authentication Username: sa Password: [YourPassword]

If it fails, reset the password in SSMS or via script:

ALTER LOGIN sa WITH PASSWORD = 'NewStrongPassword';

**🔧 Step 2: Enable**sa**Login**

In SSMS:

1. Expand Security > Logins
2. Right-click on sa → Properties
3. Go to **Status**
   * Set **Login: Enabled**
4. Apply and click OK

**🔧 Step 3: Enable Mixed Mode Authentication**

SQL Server must allow **SQL Authentication** to use sa.

1. Right-click on the **server instance** → Properties
2. Go to **Security** tab
3. Select **SQL Server and Windows Authentication mode**
4. Click OK

👉 **Restart SQL Server** after making this change:

services.msc → SQL Server (MSSQLSERVER) → Restart

**🔧 Step 4: Check Your Connection String**

Make sure your connection string looks something like this:

"Server=localhost;Database=YourDbName;User Id=sa;Password=YourPassword;"

✅ Double-check:

* **Correct server name**
* **Correct credentials**
* No missing semicolons or special character issues

**🔧 Step 5: Check SQL Server Logs**

To get the **exact reason** for State: 1, check the SQL Server error log:

EXEC xp\_readerrorlog;

Look for entries like:

Login failed for user 'sa'. Reason: Password did not match...

**🧪 Optional: Test using**sqlcmd

You can try this in terminal to validate:

sqlcmd -S localhost -U sa -P YourPassword

**✅ If All Else Fails**

**Reset sa Password Using Windows Authentication**

If sa is locked or forgotten:

1. Connect with a **Windows Auth** login (e.g., your Admin user)
2. Run:

ALTER LOGIN sa WITH PASSWORD = 'NewStrongPassword'; ALTER LOGIN sa ENABLE;

Let me know if you'd like help:

* Reviewing your **connection string**
* Writing a quick **C#/.NET connection test**
* Enabling SQL Authentication from config files or SQL Server setup

We can solve this in minutes once we confirm the exact root cause.

**You said:**

in mac

**ChatGPT said:**

Thanks for clarifying — you're running this on **Mac**, which changes a few things since SQL Server doesn't run natively on macOS.

**✅ Likely Setup on Mac**

You're probably using one of these setups:

1. **Docker container** running SQL Server on Mac
2. **Remote SQL Server** (hosted on Windows or Azure)
3. **Local SQL Server via VM (less common)**

The error still means: **Login failed for user 'sa'** — so let’s walk through how to fix it depending on your setup.

**🐳 If You're Using SQL Server via Docker (Most Common on Mac)**

**🔍 Step 1: Check Your Docker Command**

You probably started the container like this:

docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=YourStrongPassword123' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2019-latest

✅ **Make sure** the SA\_PASSWORD in this command matches what you're using in your app/connection string.

**🧪 Step 2: Test Login from Mac Terminal**

Install sqlcmd if you haven’t:

brew tap microsoft/mssql-release https://github.com/Microsoft/homebrew-mssql-release brew install --no-sandbox msodbcsql17 mssql-tools

Then test the connection:

sqlcmd -S localhost -U sa -P 'YourStrongPassword123'

**🔁 Step 3: If You Forgot the Password**

1. Stop and remove the container:

docker stop sqlserver && docker rm sqlserver

1. Run it again with a **new password**:

docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=NewPassword123!' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2019-latest

**⚙️ Connection String Example in .NET on Mac**

If you're using EF Core or .NET Core on Mac, your appsettings.json should include:

"ConnectionStrings": { "DefaultConnection": "Server=localhost,1433;Database=YourDb;User Id=sa;Password=YourStrongPassword123;" }

⚠️ **Important**:

* Use localhost,1433 (comma instead of colon)
* Don't include Trusted\_Connection=True or Integrated Security=SSPI — those are for Windows Auth

**🔒 Common Issues on Mac Setup**

| **Issue** | **Fix** |
| --- | --- |
| SQL Server container not running | Run docker ps to check. Start if needed. |
| Wrong password or sa disabled | Recreate container with new SA\_PASSWORD |
| SQL port not exposed | Make sure Docker run command includes -p 1433:1433 |
| Using Integrated Security in connection string | Remove it; not supported on macOS |

**✅ Quick Fix Summary (Mac + Docker)**

docker stop sqlserver && docker rm sqlserver docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=MyStrongPass123!' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2022-latest

Then test:

sqlcmd -S localhost -U sa -P 'MyStrongPass123!'

Let me know if you want to:

* Check your connection string
* View Docker logs
* Connect from C# code on Mac

I can walk you through it based on your exact setup.

**You said:**

check my connectiom string

**ChatGPT said:**

Absolutely — please paste your **connection string** here so I can review it and spot any issues.

If it's in a .NET project, you'll likely find it in one of these places:

**✅ In**appsettings.json**:**

"ConnectionStrings": { "DefaultConnection": "Server=localhost,1433;Database=YourDb;User Id=sa;Password=YourPassword;" }

**✅ Or directly in C# code:**

var connectionString = "Server=localhost,1433;Database=YourDb;User Id=sa;Password=YourPassword;";

🔐 **Before you share**:  
If your real password is in there, you can mask it like this:

Password=\*\*\*\*\*\*\*\*

Once I see the full connection string format, I’ll let you know exactly what to fix.

**You said:**

"HotelListingDbConnectionString": "Server=localhost,1433;User Id=sa;Password=StrongPa$$word@2025;Database=HotellistDB;Trusted\_Connection=false;Encrypt=false;MultipleActiveResultSets=True;TrustServerCertificate=true;"

**ChatGPT said:**

Thanks for sharing your connection string. Here's what you provided:

"HotelListingDbConnectionString": "Server=localhost,1433;User Id=sa;Password=StrongPa$$word@2025;Database=HotellistDB;Trusted\_Connection=false;Encrypt=false;MultipleActiveResultSets=True;TrustServerCertificate=true;"

Let's go through it step by step:

**✅ Breakdown of Your Connection String**

| **Key** | **Value** | **✅ / ❌** | **Notes** |
| --- | --- | --- | --- |
| **Server** | localhost,1433 | ✅ | Correct format for TCP on macOS/Docker |
| **User Id** | sa | ✅ | Correct |
| **Password** | StrongPa$$word@2025 | ✅ | Looks good (make sure it matches Docker's SA\_PASSWORD) |
| **Database** | HotellistDB | ✅ | Ensure this DB exists |
| **Trusted\_Connection** | false | ✅ | Correct for SQL Auth (remove or keep false) |
| **Encrypt** | false | ✅ | OK for local testing |
| **MultipleActiveResultSets** | True | ✅ | Optional but fine |
| **TrustServerCertificate** | true | ✅ | Required when Encrypt=false or using self-signed certs |

**🔍 Most Likely Cause of the Login Failure**

You're likely running SQL Server in a **Docker container** on macOS. That means:

1. The sa **password** in the Docker run command must match StrongPa$$word@2025.
2. If it doesn’t, SQL Server will reject the login with Error 18456.

**🧪 Test the Connection from Terminal**

Try this to verify your password is correct:

sqlcmd -S localhost -U sa -P 'StrongPa$$word@2025'

If it fails, do this:

**🔁 Restart the SQL Server Container with Correct Password**

docker stop sqlserver docker rm sqlserver docker run -e 'ACCEPT\_EULA=Y' \ -e 'SA\_PASSWORD=StrongPa$$word@2025' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2022-latest

Then try again.

**✅ Conclusion**

Your connection string is **correct**.

If you're still getting the login error:

* Confirm the Docker container was started with that **exact**SA\_PASSWORD
* Test with sqlcmd (or SSMS if you're using a remote instance)
* Confirm HotellistDB exists — or create it manually or via EF Core migrations

Let me know if you want to run EF Core migrations or seed data — I can help with that next.

**You said:**

docker logsJun 13 2023 12:34:33 Copyright (C) 2022 Microsoft Corporation Linux (Ubuntu 20.04.6 LTS) <X64> -d /var/opt/mssql/data/master.mdf -l /var/opt/mssql/data/mastlog.ldf -e /var/opt/mssql/log/errorlog ForceFlush is enabled for this instance. ForceFlush feature is enabled for log durability. The login info is removed from the store and destroyed.

**ChatGPT said:**

Thanks — based on the Docker logs snippet:

The login info is removed from the store and destroyed.

this line strongly suggests that the SQL Server instance **reset or wiped out login credentials**, possibly due to **invalid or weak**SA\_PASSWORD or **a permission issue when starting the container**.

Let’s get you up and running step-by-step. 💡

**🔍 Most Likely Problems**

1. ❌ **Your**SA\_PASSWORD**didn't meet requirements**, so sa was never created properly.
2. ❌ **Your Docker volume was corrupted or not persisted**.
3. ❌ You **didn’t pass the env variables** (SA\_PASSWORD, ACCEPT\_EULA) correctly.

**✅ Fix: Recreate the SQL Server Docker Container Correctly**

**🛠️ Step 1: Stop and Remove the Existing Container**

docker stop sqlserver docker rm sqlserver

**🧠 Step 2: Choose a Valid Strong Password**

SQL Server enforces strong passwords:

Minimum 8 characters, must contain uppercase, lowercase, numbers, and special characters

✅ Your password StrongPa$$word@2025 **should be valid**, but make sure it's quoted correctly in the terminal (to avoid issues with $ and @):

**✅ Step 3: Run the Container Properly**

docker run -e 'ACCEPT\_EULA=Y' \ -e 'SA\_PASSWORD=StrongPa$$word@2025' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2022-latest

🟡 **Important Notes**:

* Use **single quotes** for passwords with special characters.
* Do **not** use backticks or double quotes.
* Avoid volume mapping unless you want to persist the DB.

**🧪 Step 4: Test Connection**

Run:

sqlcmd -S localhost -U sa -P 'StrongPa$$word@2025'

If this works, your connection string from earlier will also start working.

**✅ Optional: Run EF Core Migration (if needed)**

If you’re using EF Core, run:

dotnet ef database update --connection "Server=localhost,1434;Database=HotellistDB;User Id=sa;Password=StrongPa$$word@2025;TrustServerCertificate=true;Encrypt=false"

**✅ TL;DR**

| **Task** | **Command** |
| --- | --- |
| Remove broken container | docker rm -f sqlserver |
| Start fresh | docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=StrongPa$$word@2025' -p 1433:1433 --name sqlserver -d mcr.microsoft.com/mssql/server:2022-latest |
| Test login | sqlcmd -S localhost -U sa -P 'StrongPa$$word@2025' |

If you still get errors, paste the full output of:

docker logs sqlserver

I’ll pinpoint it quickly.

**Unlock more with Plus**

ChatGPT Plus gives you higher limits, smarter models, and Sora for video.

Get Plus

Top of Form

Bottom of Form

ChatGPT can make mistakes. Check important info. See Cookie Preferences.

[Skip to content](https://chatgpt.com/c/68778ca6-12e4-8002-82c8-8e0bdc8fdf42#main)

**Chat history**

**You said:**

Microsoft.Data.SqlClient.SqlException (0x80131904): Login failed for user 'sa'. at Microsoft.Data.SqlClient.TdsParser.ThrowExceptionAndWarning(TdsParserStateObject stateObj, Boolean callerHasConnectionLock, Boolean asyncClose) at Microsoft.Data.SqlClient.TdsParser.TryRun(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj, Boolean& dataReady) at Microsoft.Data.SqlClient.TdsParser.Run(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.CompleteLogin(Boolean enlistOK) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.LoginNoFailover(ServerInfo serverInfo, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance, SqlConnectionString connectionOptions, SqlCredential credential, TimeoutTimer timeout) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.OpenLoginEnlist(TimeoutTimer timeout, SqlConnectionString connectionOptions, SqlCredential credential, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance) at Microsoft.Data.SqlClient.SqlInternalConnectionTds..ctor(DbConnectionPoolIdentity identity, SqlConnectionString connectionOptions, SqlCredential credential, Object providerInfo, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance, SqlConnectionString userConnectionOptions, SessionData reconnectSessionData, Boolean applyTransientFaultHandling, String accessToken, DbConnectionPool pool) at Microsoft.Data.SqlClient.SqlConnectionFactory.CreateConnection(DbConnectionOptions options, DbConnectionPoolKey poolKey, Object poolGroupProviderInfo, DbConnectionPool pool, DbConnection owningConnection, DbConnectionOptions userOptions) at Microsoft.Data.ProviderBase.DbConnectionPool.CreateObject(DbConnection owningObject, DbConnectionOptions userOptions, DbConnectionInternal oldConnection) at Microsoft.Data.ProviderBase.DbConnectionPool.TryGetConnection(DbConnection owningObject, UInt32 waitForMultipleObjectsTimeout, Boolean allowCreate, Boolean onlyOneCheckConnection, DbConnectionOptions userOptions, DbConnectionInternal& connection) at Microsoft.Data.ProviderBase.DbConnectionPool.TryGetConnection(DbConnection owningObject, TaskCompletionSource1 retry, DbConnectionOptions userOptions, DbConnectionInternal& connection) at Microsoft.Data.ProviderBase.DbConnectionFactory.TryGetConnection(DbConnection owningConnection, TaskCompletionSource1 retry, DbConnectionOptions userOptions, DbConnectionInternal oldConnection, DbConnectionInternal& connection) at Microsoft.Data.ProviderBase.DbConnectionInternal.TryOpenConnectionInternal(DbConnection outerConnection, DbConnectionFactory connectionFactory, TaskCompletionSource1 retry, DbConnectionOptions userOptions) at Microsoft.Data.SqlClient.SqlConnection.TryOpen(TaskCompletionSource1 retry, SqlConnectionOverrides overrides) at Microsoft.Data.SqlClient.SqlConnection.Open(SqlConnectionOverrides overrides) at Microsoft.EntityFrameworkCore.Storage.RelationalConnection.OpenInternal(Boolean errorsExpected) at Microsoft.EntityFrameworkCore.Storage.RelationalConnection.Open(Boolean errorsExpected) at Microsoft.EntityFrameworkCore.SqlServer.Storage.Internal.SqlServerDatabaseCreator.<>c\_\_DisplayClass18\_0.<Exists>b\_\_0(DateTime giveUp) at Microsoft.EntityFrameworkCore.ExecutionStrategyExtensions.<>c\_\_DisplayClass12\_02.<Execute>b\_\_0(DbContext \_, TState s) at Microsoft.EntityFrameworkCore.SqlServer.Storage.Internal.SqlServerExecutionStrategy.Execute[TState,TResult](TState state, Func3 operation, Func3 verifySucceeded) at Microsoft.EntityFrameworkCore.ExecutionStrategyExtensions.Execute[TState,TResult](IExecutionStrategy strategy, TState state, Func2 operation, Func2 verifySucceeded) at Microsoft.EntityFrameworkCore.SqlServer.Storage.Internal.SqlServerDatabaseCreator.Exists(Boolean retryOnNotExists) at Microsoft.EntityFrameworkCore.SqlServer.Storage.Internal.SqlServerDatabaseCreator.Exists() at Microsoft.EntityFrameworkCore.Migrations.Internal.Migrator.Migrate(String targetMigration) at Microsoft.EntityFrameworkCore.Design.Internal.MigrationsOperations.UpdateDatabase(String targetMigration, String connectionString, String contextType) at Microsoft.EntityFrameworkCore.Design.OperationExecutor.UpdateDatabaseImpl(String targetMigration, String connectionString, String contextType) at Microsoft.EntityFrameworkCore.Design.OperationExecutor.UpdateDatabase.<>c\_\_DisplayClass0\_0.<.ctor>b\_\_0() at Microsoft.EntityFrameworkCore.Design.OperationExecutor.OperationBase.Execute(Action action) ClientConnectionId:e7f18cc6-92bb-4257-8a69-fff8f0d1ea4a Error Number:18456,State:1,Class:14 Login failed for user 'sa'. (base) Saurabh@

**ChatGPT said:**

The error message you're seeing:

Microsoft.Data.SqlClient.SqlException (0x80131904): Login failed for user 'sa'.

Error Number:18456,State:1,Class:14

means that the SQL Server login for the user sa has failed. Here's how you can troubleshoot and resolve it:

**🔍 Root Cause**

The SQL Server is rejecting the connection attempt due to one of the following:

1. **Incorrect password** for user sa
2. **SQL Server authentication mode** is not configured correctly
3. sa**login is disabled**
4. **SQL Server is unreachable**
5. **Firewall or network issue**

**✅ Steps to Fix**

**✅ 1. Check Username and Password**

* Double-check the password you're using for the sa user.
* Try connecting via **SQL Server Management Studio (SSMS)** using the same credentials.

**✅ 2. Check SQL Server Authentication Mode**

SQL Server supports two authentication modes:

* **Windows Authentication**
* **Mixed Mode Authentication** (Windows + SQL Server Logins)

If you're using sa, you **must** enable **Mixed Mode**.

To verify:

1. Open **SSMS**
2. Right-click the server → **Properties**
3. Go to **Security** tab
4. Ensure **SQL Server and Windows Authentication mode** is selected

**✅ 3. Enable the**sa**Login**

Sometimes the sa account is disabled by default.

To enable:

1. In SSMS → Go to **Security > Logins**
2. Right-click on sa → **Properties**
3. Go to **Status**
4. Set **Login: Enabled**
5. Set/Reset the password if needed

ALTER LOGIN sa ENABLE; ALTER LOGIN sa WITH PASSWORD = 'YourStrongPassword';

**✅ 4. Restart SQL Server**

After changing authentication mode or enabling login, **restart the SQL Server**:

services.msc → SQL Server (MSSQLSERVER) → Restart

**✅ 5. Check Connection String**

Ensure your connection string in .NET or appsettings.json is correct:

"ConnectionStrings": { "DefaultConnection": "Server=localhost;Database=YourDB;User Id=sa;Password=YourPassword;" }

**✅ 6. Firewall or Port Issues**

* Ensure SQL Server is listening on port **1433**
* If remote connection: check firewall or SQL Server's TCP/IP settings in **SQL Server Configuration Manager**

**🛠 Verify with a Manual SQL Connection**

Try logging in manually using SQLCMD or SSMS:

sqlcmd -S localhost -U sa -P YourPassword

If this fails, it confirms a login or configuration issue.

**ℹ️ Extra Info from the Error**

* Error Number: 18456: SQL Server login failed
* State: 1: Generic error – check SQL Server logs for more precise cause (different states give more info)

If you'd like help modifying your EF Core configuration or .NET connection string, feel free to share the relevant snippet.

**You said:**

Microsoft.Data.SqlClient.SqlException (0x80131904): Login failed for user 'sa'. at Microsoft.Data.SqlClient.TdsParser.ThrowExceptionAndWarning(TdsParserStateObject stateObj, Boolean callerHasConnectionLock, Boolean asyncClose) at Microsoft.Data.SqlClient.TdsParser.TryRun(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj, Boolean& dataReady) at Microsoft.Data.SqlClient.TdsParser.Run(RunBehavior runBehavior, SqlCommand cmdHandler, SqlDataReader dataStream, BulkCopySimpleResultSet bulkCopyHandler, TdsParserStateObject stateObj) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.CompleteLogin(Boolean enlistOK) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.LoginNoFailover(ServerInfo serverInfo, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance, SqlConnectionString connectionOptions, SqlCredential credential, TimeoutTimer timeout) at Microsoft.Data.SqlClient.SqlInternalConnectionTds.OpenLoginEnlist(TimeoutTimer timeout, SqlConnectionString connectionOptions, SqlCredential credential, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance) at Microsoft.Data.SqlClient.SqlInternalConnectionTds..ctor(DbConnectionPoolIdentity identity, SqlConnectionString connectionOptions, SqlCredential credential, Object providerInfo, String newPassword, SecureString newSecurePassword, Boolean redirectedUserInstance, SqlConnectionString userConnectionOptions, SessionData reconnectSessionData, Boolean applyTransientFaultHandling, String accessToken, DbConnectionPool pool) at Microsoft.Data.SqlClient.SqlConnectionFactory.CreateConnection(DbConnectionOptions options, DbConnectionPoolKey poolKey, Object poolGroupProviderInfo, DbConnectionPool pool, DbConnection owningConnection, DbConnectionOptions userOptions) at Microsoft.Data.ProviderBase.DbConnectionPool.CreateObject(DbConnection owningObject, DbConnectionOptions userOptions, DbConnectionInternal oldConnection) at Microsoft.Data.ProviderBase.DbConnectionPool.TryGetConnection(DbConnection owningObject, UInt32 waitForMultipleObjectsTimeout, Boolean allowCreate, Boolean onlyOneCheckConnection, DbConnectionOptions userOptions, DbConnectionInternal& connection) at Microsoft.Data.ProviderBase.DbConnectionPool.WaitForPendingOpen() --- End of stack trace from previous location --- at Microsoft.Data.SqlClient.SqlRetryLogicProvider.ExecuteAsync(Object sender, Func1 function, CancellationToken cancellationToken) at Microsoft.Data.SqlClient.SqlRetryLogicProvider.ExecuteAsync(Object sender, Func1 function, CancellationToken cancellationToken) at Microsoft.SqlTools.ServiceLayer.Connection.ReliableConnection.ReliableSqlConnection.<>c\_\_DisplayClass31\_0.<<OpenAsync>b\_\_0>d.MoveNext() in /\_/src/Microsoft.SqlTools.ManagedBatchParser/ReliableConnection/ReliableSqlConnection.cs:line 342 --- End of stack trace from previous location --- at Microsoft.SqlTools.ServiceLayer.Connection.ConnectionService.TryOpenConnection(ConnectionInfo connectionInfo, ConnectParams connectionParams) in /\_/src/Microsoft.SqlTools.ServiceLayer/Connection/ConnectionService.cs:line 712 ClientConnectionId:a987933c-c2ae-4348-971a-0d51d895991d Error Number:18456,State:1,Class:14

**ChatGPT said:**

You're again seeing the SqlException: Login failed for user 'sa' with error number 18456, State:1, Class:14. This confirms the login is failing due to an authentication issue.

Let’s break it down and guide you toward a fix.

**🔎 Summary of the Error**

* **Error Number: 18456** → SQL Server login failure.
* **State: 1** → General failure (no detailed reason shown).
* **User: 'sa'** → SQL Server system administrator account.
* **Cause**: Typically one of the following:
  + Wrong password
  + sa login is disabled
  + SQL Server is not in Mixed Mode Authentication
  + Connection string error

**✅ Step-by-Step Fix**

**🔧 Step 1: Confirm the Correct Password**

Try logging in using SSMS:

Server name: localhost or . or 127.0.0.1 Authentication: SQL Server Authentication Username: sa Password: [YourPassword]

If it fails, reset the password in SSMS or via script:

ALTER LOGIN sa WITH PASSWORD = 'NewStrongPassword';

**🔧 Step 2: Enable**sa**Login**

In SSMS:

1. Expand Security > Logins
2. Right-click on sa → Properties
3. Go to **Status**
   * Set **Login: Enabled**
4. Apply and click OK

**🔧 Step 3: Enable Mixed Mode Authentication**

SQL Server must allow **SQL Authentication** to use sa.

1. Right-click on the **server instance** → Properties
2. Go to **Security** tab
3. Select **SQL Server and Windows Authentication mode**
4. Click OK

👉 **Restart SQL Server** after making this change:

services.msc → SQL Server (MSSQLSERVER) → Restart

**🔧 Step 4: Check Your Connection String**

Make sure your connection string looks something like this:

"Server=localhost;Database=YourDbName;User Id=sa;Password=YourPassword;"

✅ Double-check:

* **Correct server name**
* **Correct credentials**
* No missing semicolons or special character issues

**🔧 Step 5: Check SQL Server Logs**

To get the **exact reason** for State: 1, check the SQL Server error log:

EXEC xp\_readerrorlog;

Look for entries like:

Login failed for user 'sa'. Reason: Password did not match...

**🧪 Optional: Test using**sqlcmd

You can try this in terminal to validate:

sqlcmd -S localhost -U sa -P YourPassword

**✅ If All Else Fails**

**Reset sa Password Using Windows Authentication**

If sa is locked or forgotten:

1. Connect with a **Windows Auth** login (e.g., your Admin user)
2. Run:

ALTER LOGIN sa WITH PASSWORD = 'NewStrongPassword'; ALTER LOGIN sa ENABLE;

Let me know if you'd like help:

* Reviewing your **connection string**
* Writing a quick **C#/.NET connection test**
* Enabling SQL Authentication from config files or SQL Server setup

We can solve this in minutes once we confirm the exact root cause.

**You said:**

in mac

**ChatGPT said:**

Thanks for clarifying — you're running this on **Mac**, which changes a few things since SQL Server doesn't run natively on macOS.

**✅ Likely Setup on Mac**

You're probably using one of these setups:

1. **Docker container** running SQL Server on Mac
2. **Remote SQL Server** (hosted on Windows or Azure)
3. **Local SQL Server via VM (less common)**

The error still means: **Login failed for user 'sa'** — so let’s walk through how to fix it depending on your setup.

**🐳 If You're Using SQL Server via Docker (Most Common on Mac)**

**🔍 Step 1: Check Your Docker Command**

You probably started the container like this:

docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=YourStrongPassword123' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2019-latest

✅ **Make sure** the SA\_PASSWORD in this command matches what you're using in your app/connection string.

**🧪 Step 2: Test Login from Mac Terminal**

Install sqlcmd if you haven’t:

brew tap microsoft/mssql-release https://github.com/Microsoft/homebrew-mssql-release brew install --no-sandbox msodbcsql17 mssql-tools

Then test the connection:

sqlcmd -S localhost -U sa -P 'YourStrongPassword123'

**🔁 Step 3: If You Forgot the Password**

1. Stop and remove the container:

docker stop sqlserver && docker rm sqlserver

1. Run it again with a **new password**:

docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=NewPassword123!' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2019-latest

**⚙️ Connection String Example in .NET on Mac**

If you're using EF Core or .NET Core on Mac, your appsettings.json should include:

"ConnectionStrings": { "DefaultConnection": "Server=localhost,1433;Database=YourDb;User Id=sa;Password=YourStrongPassword123;" }

⚠️ **Important**:

* Use localhost,1433 (comma instead of colon)
* Don't include Trusted\_Connection=True or Integrated Security=SSPI — those are for Windows Auth

**🔒 Common Issues on Mac Setup**

| **Issue** | **Fix** |
| --- | --- |
| SQL Server container not running | Run docker ps to check. Start if needed. |
| Wrong password or sa disabled | Recreate container with new SA\_PASSWORD |
| SQL port not exposed | Make sure Docker run command includes -p 1433:1433 |
| Using Integrated Security in connection string | Remove it; not supported on macOS |

**✅ Quick Fix Summary (Mac + Docker)**

docker stop sqlserver && docker rm sqlserver docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=MyStrongPass123!' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2022-latest

Then test:

sqlcmd -S localhost -U sa -P 'MyStrongPass123!'

Let me know if you want to:

* Check your connection string
* View Docker logs
* Connect from C# code on Mac

I can walk you through it based on your exact setup.

**You said:**

check my connectiom string

**ChatGPT said:**

Absolutely — please paste your **connection string** here so I can review it and spot any issues.

If it's in a .NET project, you'll likely find it in one of these places:

**✅ In**appsettings.json**:**

"ConnectionStrings": { "DefaultConnection": "Server=localhost,1433;Database=YourDb;User Id=sa;Password=YourPassword;" }

**✅ Or directly in C# code:**

var connectionString = "Server=localhost,1433;Database=YourDb;User Id=sa;Password=YourPassword;";

🔐 **Before you share**:  
If your real password is in there, you can mask it like this:

Password=\*\*\*\*\*\*\*\*

Once I see the full connection string format, I’ll let you know exactly what to fix.

**You said:**

"HotelListingDbConnectionString": "Server=localhost,1433;User Id=sa;Password=StrongPa$$word@2025;Database=HotellistDB;Trusted\_Connection=false;Encrypt=false;MultipleActiveResultSets=True;TrustServerCertificate=true;"

**ChatGPT said:**

Thanks for sharing your connection string. Here's what you provided:

"HotelListingDbConnectionString": "Server=localhost,1433;User Id=sa;Password=StrongPa$$word@2025;Database=HotellistDB;Trusted\_Connection=false;Encrypt=false;MultipleActiveResultSets=True;TrustServerCertificate=true;"

Let's go through it step by step:

**✅ Breakdown of Your Connection String**

| **Key** | **Value** | **✅ / ❌** | **Notes** |
| --- | --- | --- | --- |
| **Server** | localhost,1433 | ✅ | Correct format for TCP on macOS/Docker |
| **User Id** | sa | ✅ | Correct |
| **Password** | StrongPa$$word@2025 | ✅ | Looks good (make sure it matches Docker's SA\_PASSWORD) |
| **Database** | HotellistDB | ✅ | Ensure this DB exists |
| **Trusted\_Connection** | false | ✅ | Correct for SQL Auth (remove or keep false) |
| **Encrypt** | false | ✅ | OK for local testing |
| **MultipleActiveResultSets** | True | ✅ | Optional but fine |
| **TrustServerCertificate** | true | ✅ | Required when Encrypt=false or using self-signed certs |

**🔍 Most Likely Cause of the Login Failure**

You're likely running SQL Server in a **Docker container** on macOS. That means:

1. The sa **password** in the Docker run command must match StrongPa$$word@2025.
2. If it doesn’t, SQL Server will reject the login with Error 18456.

**🧪 Test the Connection from Terminal**

Try this to verify your password is correct:

sqlcmd -S localhost -U sa -P 'StrongPa$$word@2025'

If it fails, do this:

**🔁 Restart the SQL Server Container with Correct Password**

docker stop sqlserver docker rm sqlserver docker run -e 'ACCEPT\_EULA=Y' \ -e 'SA\_PASSWORD=StrongPa$$word@2025' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2022-latest

Then try again.

**✅ Conclusion**

Your connection string is **correct**.

If you're still getting the login error:

* Confirm the Docker container was started with that **exact**SA\_PASSWORD
* Test with sqlcmd (or SSMS if you're using a remote instance)
* Confirm HotellistDB exists — or create it manually or via EF Core migrations

Let me know if you want to run EF Core migrations or seed data — I can help with that next.

**You said:**

docker logsJun 13 2023 12:34:33 Copyright (C) 2022 Microsoft Corporation Linux (Ubuntu 20.04.6 LTS) <X64> -d /var/opt/mssql/data/master.mdf -l /var/opt/mssql/data/mastlog.ldf -e /var/opt/mssql/log/errorlog ForceFlush is enabled for this instance. ForceFlush feature is enabled for log durability. The login info is removed from the store and destroyed.

**ChatGPT said:**

Thanks — based on the Docker logs snippet:

The login info is removed from the store and destroyed.

this line strongly suggests that the SQL Server instance **reset or wiped out login credentials**, possibly due to **invalid or weak**SA\_PASSWORD or **a permission issue when starting the container**.

Let’s get you up and running step-by-step. 💡

**🔍 Most Likely Problems**

1. ❌ **Your**SA\_PASSWORD**didn't meet requirements**, so sa was never created properly.
2. ❌ **Your Docker volume was corrupted or not persisted**.
3. ❌ You **didn’t pass the env variables** (SA\_PASSWORD, ACCEPT\_EULA) correctly.

**✅ Fix: Recreate the SQL Server Docker Container Correctly**

**🛠️ Step 1: Stop and Remove the Existing Container**

docker stop sqlserver docker rm sqlserver

**🧠 Step 2: Choose a Valid Strong Password**

SQL Server enforces strong passwords:

Minimum 8 characters, must contain uppercase, lowercase, numbers, and special characters

✅ Your password StrongPa$$word@2025 **should be valid**, but make sure it's quoted correctly in the terminal (to avoid issues with $ and @):

**✅ Step 3: Run the Container Properly**

docker run -e 'ACCEPT\_EULA=Y' \ -e 'SA\_PASSWORD=StrongPa$$word@2025' \ -p 1433:1433 --name sqlserver \ -d mcr.microsoft.com/mssql/server:2022-latest

🟡 **Important Notes**:

* Use **single quotes** for passwords with special characters.
* Do **not** use backticks or double quotes.
* Avoid volume mapping unless you want to persist the DB.

**🧪 Step 4: Test Connection**

Run:

sqlcmd -S localhost -U sa -P 'StrongPa$$word@2025'

If this works, your connection string from earlier will also start working.

**✅ Optional: Run EF Core Migration (if needed)**

If you’re using EF Core, run:

dotnet ef database update --connection "Server=localhost,1433;Database=HotellistDB;User Id=sa;Password=StrongPa$$word@2025;TrustServerCertificate=true;Encrypt=false"

**✅ TL;DR**

| **Task** | **Command** |
| --- | --- |
| Remove broken container | docker rm -f sqlserver |
| Start fresh | docker run -e 'ACCEPT\_EULA=Y' -e 'SA\_PASSWORD=StrongPa$$word@2025' -p 1433:1433 --name sqlserver -d mcr.microsoft.com/mssql/server:2022-latest |
| Test login | sqlcmd -S localhost -U sa -P 'StrongPa$$word@2025' |

If you still get errors, paste the full output of:

docker logs sqlserver

I’ll pinpoint it quickly.

**Unlock more with Plus**

ChatGPT Plus gives you higher limits, smarter models, and Sora for video.

Get Plus

Top of Form

Bottom of Form

ChatGPT can make mistakes. Check important info. See Cookie Preferences.

Dotnet build