Error Handling with IF

IF statements can be used to handle errors or unexpected conditions.

IF @@ERROR <> 0

BEGIN

    PRINT 'An error occurred.'

END

Key Characteristics of @@ERROR

1. Value: After each Transact-SQL statement, @@ERROR holds the error number of that statement. If the statement executed successfully, @@ERROR returns 0.
2. Reset After Each Statement: @@ERROR is reset to 0 after each T-SQL statement, regardless of whether an error occurred. This means you must check @@ERROR immediately after the statement that might have caused an error.
3. Usage: It's often used in conjunction with the IF statement to check for errors and take appropriate action.

Limitations and Best Practices

1. Immediate Check: Always check @@ERROR immediately after the statement you're interested in, because its value is reset after each SQL statement.
2. Superseded by TRY...CATCH: In modern T-SQL programming, @@ERROR is often replaced by the more robust TRY...CATCH construct, which offers better error handling capabilities.
3. Not Always Reliable: @@ERROR might not catch all types of errors, especially those that are severe enough to terminate the connection.
4. Use in Transactions: In transaction processing, use @@ERROR to decide whether to commit or roll back a transaction.

Conclusion

While @@ERROR provides a basic mechanism for error checking in T-SQL, its limitations mean it's often better to use the TRY...CATCH block for more comprehensive error handling. Understanding @@ERROR is still useful, particularly for maintaining and understanding legacy SQL Server code.