Name: CBoisen
Date: November 8,2023
Course: IT FDN 130 A

Assignment 04- Processing Data

Introduction

Once a database is created, continued maintenance and manipulation are required to most effectively utilize the data being stored. In order to enact change within a database, different types of statements are used to create, read, update, and delete data. By performing successful transactions using SQL, data modifications can be made permanent helping databases to remain organized and up to date with the latest available information.

Transactions

To permanently commit data changes to a database, SQL must be used to create transaction statements. A SQL transaction (TRAN, TRANSACTION) contains a group of one or more statements used to create, read, update, or delete data. While each new transaction is implicit, it is still useful to be explicit, formally defining each new statement with the command BEGIN TRAN.or

```
-- Task 1 (20 pts): Add data to the Categories table
-- TODO: Add Insert Code
BEGIN TRY -- use a try/catch block to handle errors during transactions
    BEGIN TRAN; --- Transaction/TRAN is happening whenever INSERT, UPDATE, DELETE and should be used to formally define the statements
        INSERT INTO Categories -- can't insert into identity clause
            ([CategoryName]) --- skip identity (autonumber) column - automatically generates numbers for CategoryID
        VALUES
            ('Beverages'), ('Condiments');
    COMMIT TRAN; --- not completed, data not INSERT/UPDATE/DELETE until COMMIT
END TRY
BEGIN CATCH --- used to catch the error and rollback TRAN if fails
    IF @@TRANCOUNT > 0 ROLLBACK TRANSACTION; --- if trans did not work and cant be comitted, rollback so statement is closed
    PRINT 'There was an ERROR! Refer back to ERROR message'
    PRINT Error_Message();
END CATCH
GO

SELECT @@TRANCOUNT; -- check to see if transaction is still open and needs to be closed (ROLLBACK or COMMIT)
GO

SELECT @@IDENTITY AS [Last ID from current connection], --- can see what the identityID is for new/curent rows that are added
       IDENT_CURRENT('Categories') AS [Last ID for any connection]

SELECT * FROM Categories;
GO
```

Figure 1. An example of a transaction block containing an INSERT INTO statement, used to add data to the Categories table in the database

If an error is encountered within the transaction block the transaction will not be completed (evidenced by the query @@TRANCOUNT > 0) The operation can be then be rolled back (ROLLBACK TRANSACTION) and an error message displayed, preventing any unwanted changes from being made to the database. Until the transaction completes without error it will continue to rollback. Once no errors are encountered within the block, the modifications can be made permanent by committing the transaction (COMMIT TRANSACTION).

SQL Statements

Each transaction block can contain one or multiple different SQL statements. The CREATE command is primarily used to create a new database or tables within a database. When creating a new table, it is also necessary to designate the database from which data will be used. The READ statement often refers to SELECT. It is used for data retrieval from tables or views. The SELECT statement always requires the use of FROM and an object_name to specify where the data should be retrieved from. The UPDATE command is used to modify existing records stored in a table. When using the command it is necessary to first SET the column/columns to be updated and then specify the rows that need to be updated using the WHERE clause. The DELETE statement is used to delete existing records stored in a table. Similar to the UPDATE statement, the WHERE clause must always be used to specify which rows are to be modified. Without specifying the condition for the rows for both UPDATE and DELETE, changes may be unintentionally applied to all rows.

Identity

When creating a table, it is imperative that each row within a column have a unique identification number. The IDENTITY function is used in SQL to automatically create a numbered column containing unique identification numbers for each row in the table. The function defaults to a seed (initial value) of 1 and step (increment value) of 1 but both values can also be specified in the statement. When inserting, selecting, or copying tabular data, it is useful to use @@IDENTITY to verify the last IDENTITY value generated in the current session. If the query was not run properly or encountered an error, no IDENTITY values will be generated and the function will have a NULL output. IDENT_CURRENT() can also be used if you would like to know the last generated IDENTITY value of a specific table (not necessarily the last).

Conclusion

Through SQL, data present in database tables can be managed and manipulated using transact statements. A number of different operations can be included in a transact statement and beused to create, read(select) update, and delete data. Modifications within tables can also be made using functions such as identity, ensuring organization and ease of data access through unique number assignment.