Developer SQL Scripts to Upgrade Database Schema

Yogesh S. Patil[1], Abhijit P.Ingale[2], Aniket D.Pathak[3]

Department of Computer Science & Engineering
S.S.G.B. College of Engineering & Technology, Bhusawal, Jalgaon, India

[1]*yogeshe146@gmail.com*
[2]*apingale83@gmail.com*
[3]*aniketpathak89@gmail.com*

Abstract: ***As life cycle of any software product does not remain same for long time because as time passes we required certain modification in existing product. In software database product it is difficult to do modification because we want existing database in next version it is difficult task to move existing database to next version in this paper we have given different developer SQL scripts that will help us in writing upgrade scripts. Developer will write down upgrade scripts to move their database to next version.***

Keywords: schema, upgrades, SQL Scripts, Database Application

1. Introduction

When we move our database to next version the schema of database changes we have listen down all the possible schema changes.

1. Add table - add a new database table.

The CREATE DATABASE Statement
The CREATE DATABASE statement is used to create a database.
SQL CREATE DATABASE Syntax
CREATE DATABASE database_name

CREATE DATABASE Example

Now we want to create a database called "my_db".

We use the following CREATE DATABASE statement:

CREATE DATABASE my_db

Database tables can be added with the CREATE TABLE statement.

2. Delete table - delete an existing database table.

The DROP TABLE Statement
The DROP TABLE statement is used to delete a table.
DROP TABLE table_name

3. Rename table - rename an existing database table.

Renaming a table

To rename a table, the SQL ALTER TABLE syntax is:

ALTER TABLE table_name
RENAME TO new_table_name;
For example:
ALTER TABLE suppliers
RENAME TO vendors;

This will rename the suppliers table to vendors.

4. Add column - add a database column.

Adding column(s) to a table
Syntax #1

To add a column to an existing table, the SQL ALTER TABLE syntax is:

ALTER TABLE table_name
ADD column_name column-definition;

For example:

ALTER TABLE supplier
ADD supplier_name varchar2 (50);

This will add a column called supplier_name to the supplier table.

Syntax #2

To add multiple columns to an existing table, the SQL ALTER TABLE syntax is:

ALTER TABLE table_name
ADD (column_1 column-definition,
column_2 column-definition,
...
column_n column_definition);

For example:

ALTER TABLE supplier
ADD (supplier_name varchar2 (50), city varchar2 (45));

This will add two columns (supplier_name and city) to the supplier table.

5. Delete column - delete a database column.

To delete a column in a table, use the following syntax (notice that some database systems don't allow deleting a column):
ALTER TABLE table_name

DROP COLUMN column_name

6. Rename column - rename a database column.

The script for renaming any column:

sp_RENAME 'TableName' [OldColumnName]' , '[NewColumnName]', 'COLUMN'

The script for renaming any object (table, sp etc):

sp_RENAME '[OldTableName]', '[NewTableName]'

Here are two examples of renaming database object.

Renaming database table column to new name.
Renaming database table to new name.

In both the cases we will first see existing table. Rename the object. Test object again with new name.

a) Renaming database table column to new name.

Example uses Adventure Works database. A small table with name "Table_First" is created. Table has two fields ID and Name as shown Figure 6.1

```
Use AdventureWorks
GO

CREATE TABLE Table_First(ID INT, Name VARCHAR(20))
GO

INSERT
INTO    Table_First
VALUES  (1,'Test')
GO

SELECT  *
FROM    Table_First
GO
```

Results | Messages

ID	Name
1	Test

Figure 6.1: Table_First

Now, to change the Column Name from "Name" to "NameChange" we can use command:

USE AdventureWorks

GO
sp_RENAME 'Table_First.Name', 'NameChange', 'COLUMN'

GO
Following Figure 6.2 show use of SP_RENAME Command

```
Use AdventureWorks
GO

SP_RENAME 'Table_First.Name', 'NameChange' , 'COLUMN'
GO
```

Messages

Caution: Changing any part of an object name could break scripts and

Figure 6.2: SP_RENAME Command

You can see the column name "Name" is now changed to "NameChange".

USE AdventureWorks

GO
SELECT * FROM Table_First
GO

Following Figure 6.3 verify that the column name has been changed.

Figure 6.3 Column name has been changed.

b) Renaming database table to new name.

We can change the table name too with the same command.
sp_RENAME 'Table_First', 'Table_Last'
GO

Following Figure 6.4 Shows how we can change Table Name.

```
Use AdventureWorks
GO

SP_RENAME 'Table_First', 'Table_Last'
GO
```

Messages

Caution: Changing any part of an object name could break scripts and

Figure 6.4: Table Name Change

Now, the table name "Table_First" is renamed as "Table_Last".
"Table_First" will no longer be available in database. We can verify this by running script:

USE AdventureWorks
GO
SELECT * FROM Table_First
GO

The Messages shows an error "Invalid object name 'Table_First'."

To check that the new renamed table exist in database run script:
USE AdventureWorks
GO
SELECT * FROM Table_Last
GO

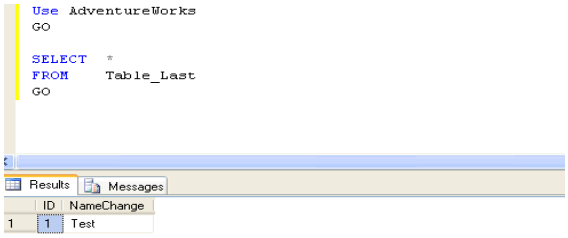

Figure 6.5: Table_Last

You can see the same data now available in new table named “Table_Last” as shown in Figure 6.5

2. Complex Kinds of Schema Change

1. MANIPULATE DATA IN PLACE - UPDATING THE EXISTING DATABASE CONTENT.

The UPDATE Statement

The UPDATE statement is used to update existing records in a table.

SQL UPDATE Syntax

UPDATE table_name SET column1=value, column2=value2,... WHERE some_column=some_value

Note: Notice the WHERE clause in the UPDATE syntax. The WHERE clause specifies which record or records that should be updated. If you omit the WHERE clause, all records will be updated!

SQL UPDATE Example

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob		

Table 1: Persons table

Now we want to update the person "Tjessem, Jakob" in the "Persons" table.
We use the following SQL statement:

UPDATE Persons SET Address='Nissestien 67', City='Sandnes' WHERE LastName='Tjessem' AND FirstName='Jakob'

The "Persons" table will now look like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob	Nissestien 67	Sandnes

Table 2: Persons table after UPDATE

SQL UPDATE Warning

Be careful when updating records. If we had omitted the WHERE clause in the example above, like this:

UPDATE Persons SET Address='Nissestien 67', City='Sandnes'

The "Persons" table would have looked like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Nissestien 67	Sandnes
2	Svendson	Tove	Nissestien 67	Sandnes
3	Pettersen	Kari	Nissestien 67	Sandnes
4	Nilsen	Johan	Nissestien 67	Sandnes
5	Tjessem	Jakob	Nissestien 67	Sandnes

Table 3: Persons table after UPDATE& without where

2. Column type changes - data type migration e.g. changing column type from textual to numeric. Here's a little sample script that shows on possible method:

```
create table test (id int)
create table test_tmp (id ntext)
insert into test
values (1)
insert into test
values (2)
insert into test_tmp
select convert(ntext,cast(id as nvarchar)) from test
drop table test
exec sp_rename 'test_tmp','test'
```

Basically we create a copy of the table, then populate it. We first convert the int to nvarchar then we take to a text value. Finally we drop the old table and rename the temp table.

3. Update of foreign keys - If a primary key change then all its foreign keys may require updates.

Sql alter table statement to change primary key and

foreign key constraint

Example
Sample table: agent1

Suppose there is a PRIMARY KEY CONSTRAINT named 'pk_ag_code' for the column 'agent_code' of the 'agent1' table.

To modify the PRIMARY KEY CONSTRAINT named 'pk_ag_code, the following sql statements can be used:

View plain copy to clip board print?

```
ALTER TABLE  agent1
DROP CONSTRAINT pk_ag_code;
ALTER TABLE  agent1
DROP CONSTRAINT pk_ag_code;
```

View plain copy to clip board print?

```
ALTER TABLE  agent1
ADD CONSTRAINT pk_ag_code
PRIMARY KEY(agent_code);
ALTER TABLE  agent1
ADD CONSTRAINT pk_ag_code
PRIMARY KEY(agent_code);
```

Sql alter table statement to change foreign key constraint

4. Table Merging and Splitting - e.g. one table becomes two or vice versa.

a) Merging:-The MERGE statement basically merges data from a source result set to a target table based on a condition that you specify and if the data from the source already exists in the target or not. The new SQL command combines the sequence of conditional INSERT, UPDATE and DELETE commands in a single atomic statement, depending on the existence of a record. The new MERGE SQL command looks like as below:

```
MERGE <target_table> [AS TARGET]
USING <table_source> [AS SOURCE]
ON <search_condition>
[WHEN MATCHED
THEN <merge_matched> ]
[WHEN NOT MATCHED [BY TARGET]
THEN <merge_not_matched> ]
[WHEN NOT MATCHED BY SOURCE
THEN <merge_ matched> ];
```

b) Splitting

```
Create PROCEDURE Proc_SplitWordsToChar
@Sentence VARCHAR(MAX)
AS
BEGIN
   SET NOCOUNT ON
   SET XACT_ABORT ON

DECLARE @Words VARCHAR(MAX)
DECLARE @t VARCHAR(MAX)
    DECLARE @I INT
  SET @Words = @Sentence
  SELECT @I = 0
  WHILE(@I < LEN(@Words)+1)
  BEGIN
   SELECT @t = SUBSTRING(@words,@I,1)

    Insert into #temp values (@t)
    SET @I = @I + 1
  END
END

--Execute Proc_SplitWordsToChar 'Hello'
```

3. Conclusion

Finally we conclude that this SQL upgrade scripts will help for everyone who want to upgrade their database to next version.

References

[1] Robert M. Marks," A Metadata Driven Approach to Performing Multi-Vendor Database Schema Upgrades",' 19th IEEE International Conference and Workshops on Engineering of Computer-Based Systems', 2012, PP 108-116.
[2] Praveena Mandapati , U. Tulasi ," Extended Schema Extraction and Evolution",'IEEE International Journal of Emerging Technology and Advanced Engineering', 2012, Vol. No. 2, PP 220-224.
[3] Hui Liu, Shengqi Wu ," Data Storage Schema Upgrade via Metadata Evolution in SaaS ",' IEEE International Conference on Consumer Electronics, Communications & Networks (CECNet)', 2012,PP 3148-3151.
[4] Shengfeng Wu, Iulian Neamtiu," Schema Evolution Analysis for Embedded Databases", 'IEEE International Conference on Data Engineering Workshops (ICDEW)', 2011, PP 151-156.
[5] Zenonas Theodosiou, Olga Georgiou, Nicolas Tsapatsoulis," Evaluating Annotators Consistency with the Aid of an Innovative Database Schema",' Sixth IEEE International Workshop on Semantic Media Adaptation and Personalization', 2011, PP 74-78.
[6] Hendrik Decker, Davide Martinenghi," Inconsistency-Tolerant Integrity Checking",'IEEE Transactions on Knowledge and Data Engineering ', Vol. 23, No. 2, February 2011.PP 218-234.
[7] Anuj Jaiswal, David J," Uninterpreted Schema Matching with Embedded Value Mapping under Opaque Column Names and Data Values ",' IEEE Transactions on Knowledge and Data Engineering', Vol. 22, No. 2, February 2010. PP 291-304.