8. Self Join

In this article we are going to learn of self joined tables. So far we are going to learn why we need Self joined tables and how to create them? Included examples can be executed in both SQL Server & MySQL.

Self Join Tables

A table joins with itself is called a self join. When a foreign key related to the same table it is called a self joined table. We can create a query to join with the same table, like a join with two tables.

Self join mainly uses an INNER JOIN or a LEFT JOIN. But it can create any kind of join. The syntax for a self-join table can be written in one of the following ways.

Syntax
SELECT TableA.*, TableB.* FROM Table1 TableA LEFT JOIN Table1 TableB ON TableA.ForiegnKeyColumn = TableB.PrimaryKeyColumn;
SELECT TableA.*, TableB.* FROM Table1 TableA INNER JOIN Table1 TableB ON TableA.ForiegnKeyColumn = TableB.PrimaryKeyColumn;

Table 1. Syntax for Self Join

Refer to the following self joined table for our examples.

Subjects Table:

SubjectCode	Name	RequiredCode	Level
201	Systems Software	NULL	2
202	Mathematics	NULL	2
301	Computer Architecture	**201**	3
302	Discrete Maths	**202**	3

303	Engineering Maths I	**202**	3
401	Processor Design	**301**	4
402	Engineering Maths II	**303**	4
403	Compiler Design	**302**	4

Table 2. Subjects

We can create a table and insert data using the following query.

Sample Data Script:

```
CREATE TABLE Subjects(
    SubjectCode VARCHAR (10) NOT NULL,
    Name VARCHAR (50) NOT NULL,
    Level INT NOT NULL,
    RequiredCode VARCHAR (10) NULL,
    PRIMARY KEY (SubjectCode )
);

INSERT Subjects (SubjectCode, Name, Level, RequiredCode) VALUES ('401',
'Processor Design', 4,'301');
INSERT Subjects (SubjectCode, Name, Level, RequiredCode) VALUES ('402',
'Engineering Mathematics II', 4,'303');
INSERT Subjects (SubjectCode, Name, Level, RequiredCode) VALUES ('403',
'Compiler Design', 4,'302');
INSERT Subjects (SubjectCode, Name, Level, RequiredCode) VALUES ('301',
'Computer Architecture', 3,'201');
INSERT Subjects (SubjectCode, Name, Level, RequiredCode) VALUES ('302',
'Discrete Mathematics', 3,'202');
INSERT Subjects (SubjectCode, Name, Level, RequiredCode) VALUES ('303',
'Engineering Mathematics I', 3,'202');
INSERT Subjects (SubjectCode, Name, Level, RequiredCode) VALUES ('201',
'Systems Software', 2, NULL);
INSERT Subjects (SubjectCode, Name, Level, RequiredCode) VALUES ('202',
'Mathematics', 2, NULL);
```

Script 1. Sample Data

The relation of the self-join table can be represented graphically in the following way. SubjectCode is the primary key and RequiredCode is the foreign key.

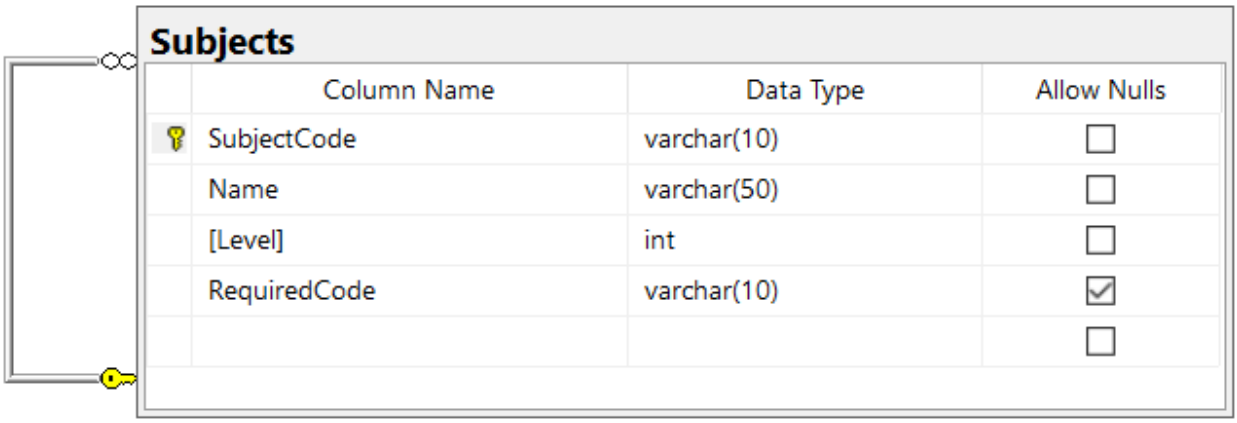

Figure 1. Subjects Table

Example 1. Self Join Tables

Students have a list of subjects at the university. They should have been completed the prerequisite subject before applying for the new subject. The main set will be Subjects. Prerequisite subjects will be a subset of it, but available in the same table.

By using the below query we can create a self-join.

SQL Script:

```
SELECT Subjects .*, RequiredSubject.Name AS RequiredName
FROM Subjects
LEFT JOIN Subjects RequiredSubject
ON Subjects.RequiredCode = RequiredSubject.SubjectCode;
```

Script 2. Self Join

Joined data is available in the following format.

SubjectCode	Name	RequiredCode	Level	RequiredName
201	Systems Software	**NULL**	2	NULL
202	Mathematics	**NULL**	2	NULL
301	Computer Architecture	**201**	3	Systems Software
302	Discrete Maths	**202**	3	Mathematics
303	Engineering Maths I	**202**	3	Mathematics
401	Processor Design	**301**	4	Computer

				Architecture
402	Engineering Maths II	**303**	4	Engineering Maths I
403	Compiler Design	**302**	4	Discrete Maths

Table 4. Joined Subject Data

Subjects Table is the left table. The right Table is shown below.

Right Table:

SubjectCode	Name	RequiredCode	Level
201	Systems Software	NULL	2
202	Mathematics	NULL	2
301	Computer Architecture	**201**	3
302	Discrete Maths	**202**	3
303	Engineering Maths I	**202**	3

Table 5. Right Table