# C:\Users\stefano.maggi.CONBIPELSPA\Desktop\prom38.png

**Date**: 19/06/2012

**Procedure:** SQL Server - Useful Metadata queries

**Source:** [**LINK**](http://www.tech-recipes.com/rx/24343/sql-server-useful-metadata-queries/)

**Permalink:** [**LINK**](http://heelpbook.altervista.org/2012/sql-server-useful-metadata-queries/)

**Created by:** HeelpBook Staff

**Document Version:** 1.0

# **SQL Server – Useful Metadata queries**

**Metadata queries** are really helpful in discovering information for a given database schema. Database information including the tables, views, columns names, data types, indexes, and table constraints are all available using queries such as these.

During this tutorial, I want to explore some useful metadata queries.

Let us start by finding the list of tables created in the given database.

select \*

from information\_schema.tables

where table\_type='base table';

Now let us list the views created in the given database.

select \*

from information\_schema.tables

where table\_type='view';

Let us create a query that lists the column names, data types, whether the column allows null or not, and the maximum allowed characters in the row.

select column\_name, data\_type, is\_nullable,

character\_maximum\_length

from information\_schema.columns

where table\_name='emp';

This **query** shows the table name, object id, table creation date, and the last table modified time.

select name, object\_id, create\_date, modify\_date

from sys.tables;

Listing the created indexes for a table with the column names is frequently required. In this query **a.name** is the table name for which you are listing the indexes. By removing the **a.name** condition, you can see all the created indexes in your database.

SELECT a.name table\_name,

b.name index\_name,

d.name column\_name

FROM sys.tables a,

sys.indexes b,

sys.index\_columns c,

sys.columns d

WHERE a.object\_id = b.object\_id

AND b.object\_id = c.object\_id

AND b.index\_id = c.index\_id

AND c.object\_id = d.object\_id

AND c.column\_id = d.column\_id

AND a.name = 'emp';

This query will list the defined constraints on tables with the column names. In thie example, we can see the emp table’s unique, primary or foreign key constraints.

SELECT a.table\_name,

a.constraint\_name,

b.column\_name,

a.constraint\_type

FROM information\_schema.table\_constraints a,

information\_schema.key\_column\_usage b

WHERE a.table\_name = 'EMP'

AND a.table\_name = b.table\_name

AND a.table\_schema = b.table\_schema

AND a.constraint\_name = b.constraint\_name;

Suppose you want to write a ***‘select count(1) from table\_name’*** query for each table in your database, but you have more than 100 tables in your database. Instead of writing a separate query for each table, you can generate those queries using **SQL**. Therefore, you can write **SQL** code to generate **SQL**.

SELECT 'select count(1) from [' + table\_name + '];'

FROM information\_schema.tables;