

# Variable SQL Server: Déclarer, Définir, Sélectionner, Global, Local [Exemples TSQL]



## Quelle est la variable?

Dans MS SQL, les variables sont l'objet qui sert d'espace réservé à un emplacement mémoire. Variable contient une valeur de donnée unique.

Dans ce tutoriel, vous apprendrez:

Quelle est la variable?

Types de variable: Local, Global

- Comment déclarer une variable
- Assigner une valeur à un VARIABLE
  - Pendant la déclaration de variable à l'aide du mot clé DECLARE.
  - Utiliser SET
  - UTILISER SELECT
  - Autres exemples
- Faits intéressants!

## Types de variable: Local, Global

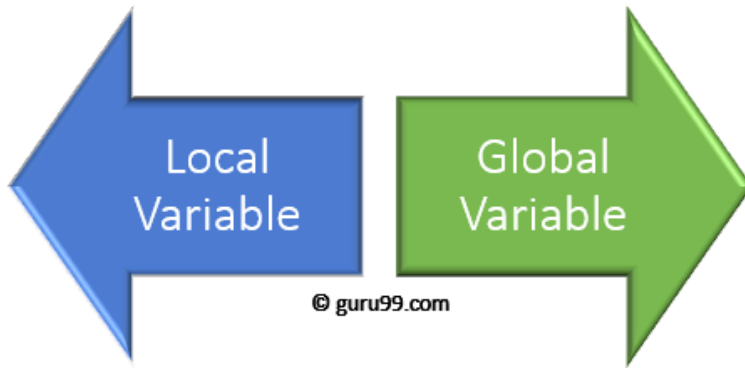
MS SQL a deux types de variables:

1. Variable locale
2. Variable globale.

Cependant, l'utilisateur ne peut créer qu'une variable locale.

La figure ci-dessous explique deux types de variables disponibles dans le serveur MS SQL.

@ User Defined



@@System Defined



(./images/1/030819\_0723\_SQLServerVa1.png).

Type de variables dans SQL Server

## Variable locale:

- Un utilisateur déclare la variable locale.
- Par défaut, une variable locale commence par @.
- La portée de chaque variable locale est limitée au **lot ou à la procédure en cours** dans une session donnée.

## Variable globale:

- Le système gère la variable globale . Un utilisateur ne peut pas les déclarer.
- La variable globale commence par @@
- Il stocke **les informations relatives à la session** .

## Comment déclarer une variable

- Avant d'utiliser une variable dans un lot ou une procédure, vous devez **déclarer la variable**.
- DECLARE command is used to DECLARE variable which acts as a placeholder for the memory location.
- Only once the declaration is made, a variable can be used in the subsequent part of batch or procedure.

### TSQL Syntax:

```
DECLARE { @LOCAL_VARIABLE[AS] data_type [ = value ] }
```

### Rules:

- Initialization is an optional thing while declaring.



- By default, DECLARE initializes variable to NULL.
- Using the keyword 'AS' is optional.
- To declare more than one local variable, use a comma after the first local variable definition, and then define the next local variable name and data type.

## Examples of Declaring a variable:

### Query: With 'AS'

```
DECLARE @COURSE_ID AS INT;
```

### Query: Without 'AS'

```
DECLARE @COURSE_NAME VARCHAR (10);
```

### Query: DECLARE two variables

```
DECLARE @COURSE_ID AS INT, @COURSE_NAME VARCHAR (10);
```

## Assigning a value to a VARIABLE

You can assign a value to a variable in the following **three** ways:

1. During variable declaration using DECLARE keyword.
2. Using SET
3. Using SELECT

Let's have a look at all three ways in detail:

### During variable declaration using DECLARE keyword.

#### T-SQL Syntax:

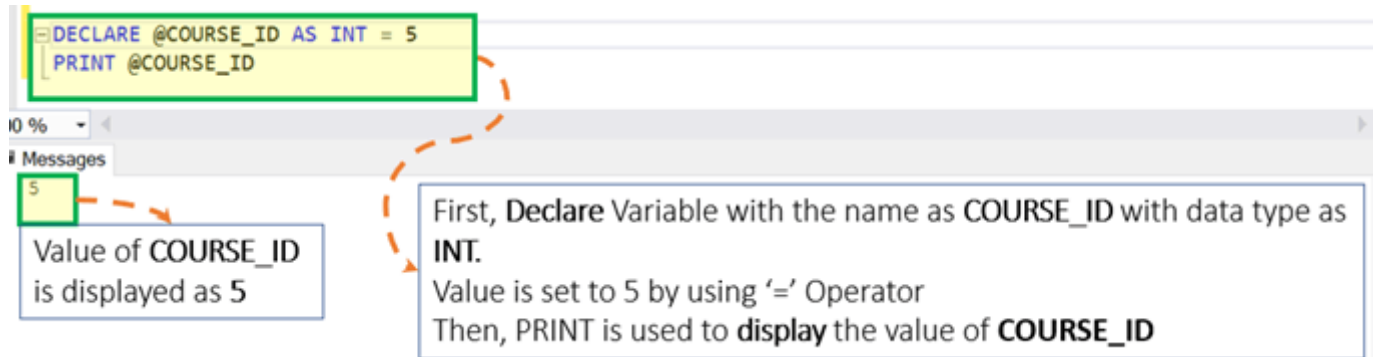
```
DECLARE { @Local_Variable [AS] Datatype [ = value ] }
```

Here, after datatype we can use '=' followed by value to be assigned

#### Query:



```
DECLARE @COURSE_ID AS INT = 5  
PRINT @COURSE_ID
```



(./images/1/030819\_0723\_SQLServerVa2.png).

## Using SET

Sometimes we want to keep declaration and initialization separate. SET can be used to assign values to the variable, post declaring a variable. Below are the different ways to assign values using SET:

**Example:** Assigning a value to a variable using SET

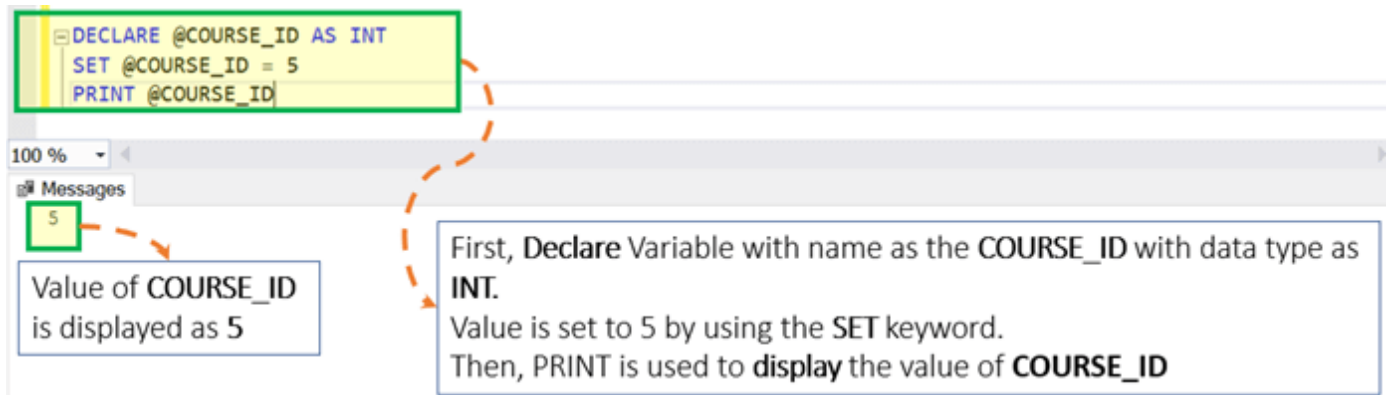
### Syntax:

```
DECLARE @Local_Variable <Data_Type>  
SET @Local_Variable = <Value>
```

### Query:

```
DECLARE @COURSE_ID AS INT  
SET @COURSE_ID = 5  
PRINT @COURSE_ID
```





(./images/1/030819\_0723\_SQLServerVa3.png).

**Example:** Assign a value to **multiple variables** using SET.

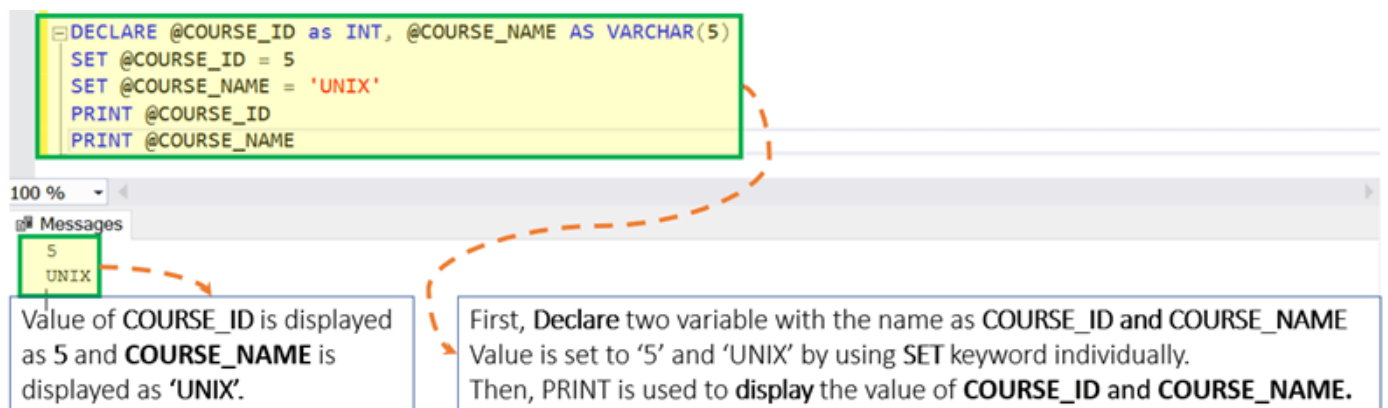
**Syntax:**

```
DECLARE @Local_Variable_1 <Data_Type>, @Local_Variable_2 <Data_Type>,
SET @Local_Variable_1 = <Value_1>
SET @Local_Variable_2 = <Value_2>
```

**Rule:** One SET Keyword can be used to assign a value to only **one variable**.

**Query:**

```
DECLARE @COURSE_ID as INT, @COURSE_NAME AS VARCHAR(5)
SET @COURSE_ID = 5
SET @COURSE_NAME = 'UNIX'
PRINT @COURSE_ID
PRINT @COURSE_NAME
```



(./images/1/030819\_0723\_SQLServerVa4.png).

**Example:** Assigning a value to a variable with a **Scalar Subquery** using SET

**Syntax:**

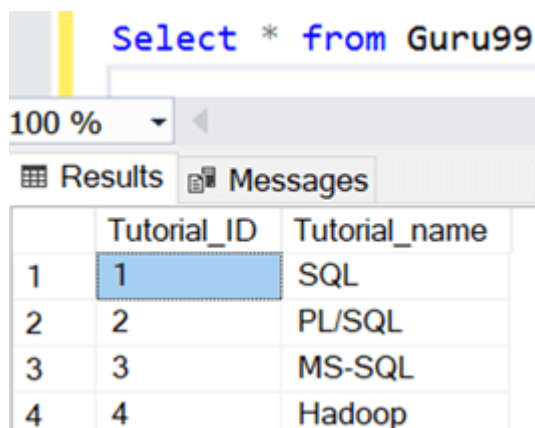


```
DECLARE @Local_Variable_1 <Data_Type>, @Local_Variable_2 <Data_Type>, SET @Local_Variable_1  
= (SELECT <Column_1> from <Table_Name> where <Condition_1>)
```

## Rules:

- Enclose the query in parenthesis.
- The query should be a scalar query. A scalar query is a query with results as just one row and one column. Otherwise, the query will throw an error.
- If the query returns zero rows, then the variable is set to EMPTY, i.e., NULL.

**Assumption:** Assume that we have the table as 'Guru99' with two columns as displayed below:



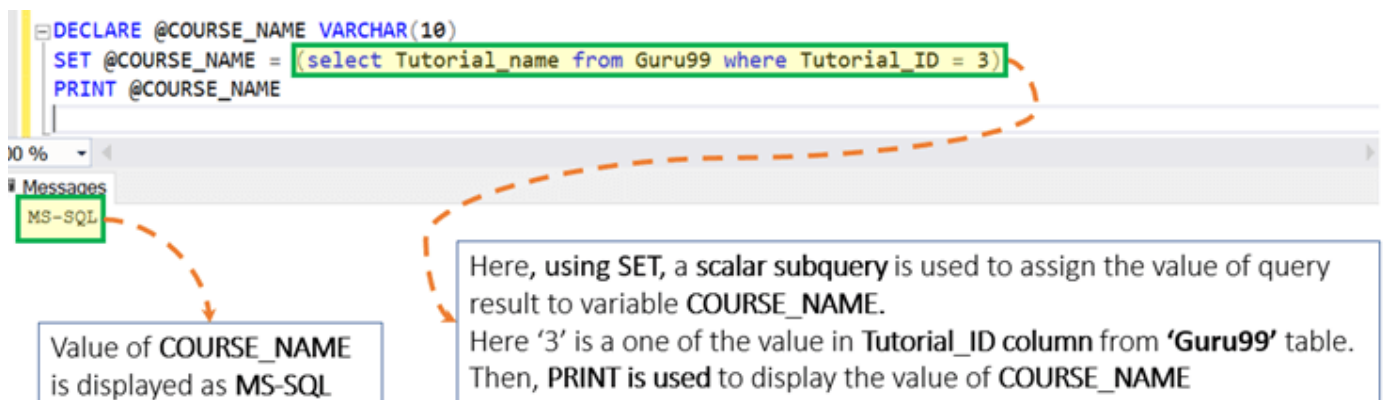
	Tutorial_ID	Tutorial_name
1	1	SQL
2	2	PL/SQL
3	3	MS-SQL
4	4	Hadoop

[./images/1/030819\\_0723\\_SQLServerVa5.png](#)

We will use 'Guru99' table in the further tutorials

**Example 1:** When subquery return one row as a result.

```
DECLARE @COURSE_NAME VARCHAR (10)  
SET @COURSE_NAME = (select Tutorial_name from Guru99 where Tutorial_ID = 3)  
PRINT @COURSE_NAME
```



Value of COURSE\_NAME is displayed as MS-SQL

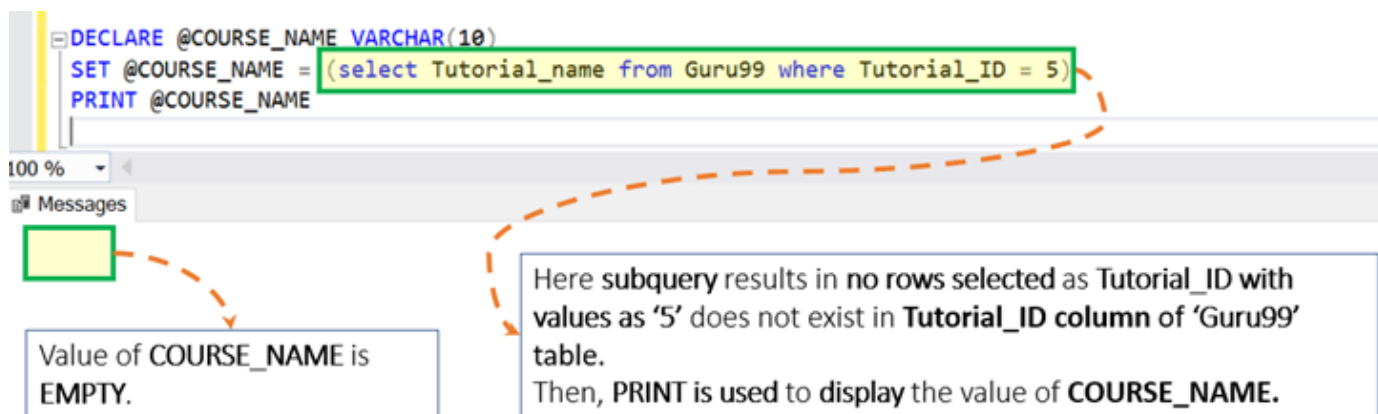
Here, using SET, a scalar subquery is used to assign the value of query result to variable COURSE\_NAME.  
Here '3' is a one of the value in Tutorial\_ID column from 'Guru99' table.  
Then, PRINT is used to display the value of COURSE\_NAME

[./images/1/030819\\_0723\\_SQLServerVa6.png](#)

**Example 2:** When subquery returns zero row as a result

```
DECLARE @COURSE_NAME VARCHAR (10)
SET @COURSE_NAME = (select Tutorial_name from Guru99 where Tutorial_ID = 5)
PRINT @COURSE_NAME
```

In this particular case, the variable value is EMPTY, i.e., NULL.



[./images/1/030819\\_0723\\_SQLServerVa7.png](#)

## USING SELECT

Just like SET, we can also use SELECT to assign values to the variables, post declaring a variable using DECLARE. Below are different ways to assign a value using SELECT:

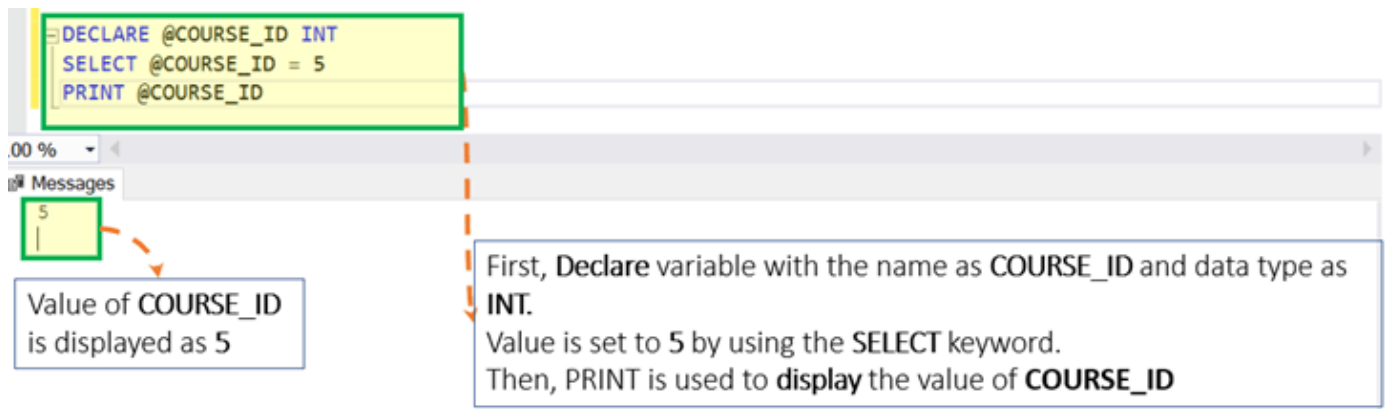
**Example:** Assigning a value to a variable using SELECT

**Syntax:**

```
DECLARE @LOCAL_VARIABLE <Data_Type>
SELECT @LOCAL_VARIABLE = <Value>
```

**Query:**

```
DECLARE @COURSE_ID INT
SELECT @COURSE_ID = 5
PRINT @COURSE_ID
```



(./images/1/030819\_0723\_SQLServerVa8.png).

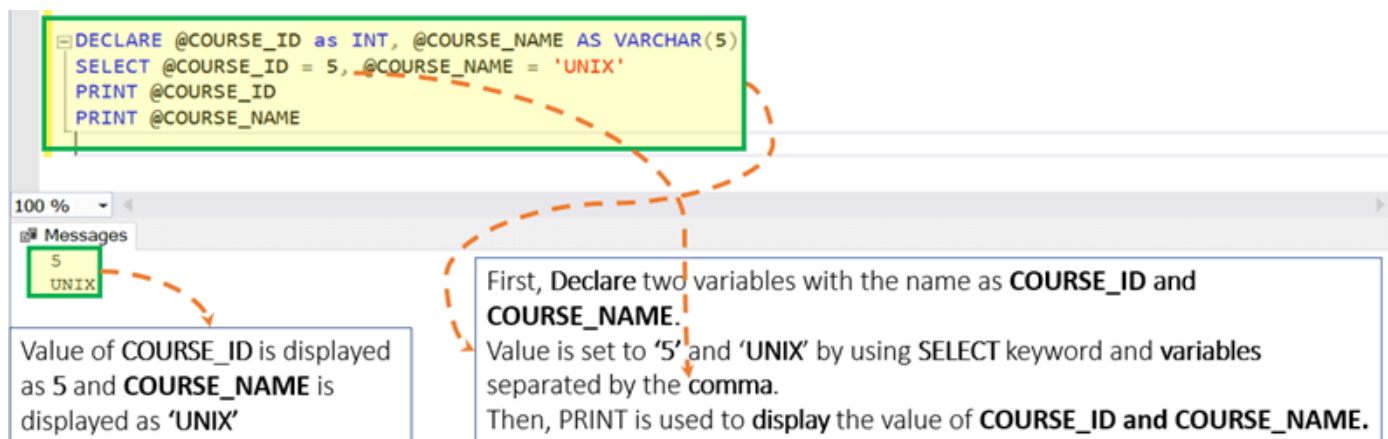
**Example:** Assigning a value to multiple variable using SELECT

**Syntax:**

```
DECLARE @Local_Variable _1 <Data_Type>, @Local_Variable _2 <Data_Type>, SELECT @Local_Variable _1 = <Value_1>, @Local_Variable _2 = <Value_2>
```

**Rules:** Unlike SET, SELECT can be used to assign a value **to multiple variables** separated by the **comma**.

```
DECLARE @COURSE_ID as INT, @COURSE_NAME AS VARCHAR(5)
SELECT @COURSE_ID = 5, @COURSE_NAME = 'UNIX'
PRINT @COURSE_ID
PRINT @COURSE_NAME
```



(./images/1/030819\_0723\_SQLServerVa9.png).

**Example:** Assigning the value to a variable with a Subquery using SELECT

**Syntax:**



```
DECLARE @Local_Variable_1 <Data_Type>, @Local_Variable_2 <Data_Type>, SELECT @Local_Variable_1 = (SELECT <Column_1> from <Table_name> where <Condition_1>)
```

## Rules:



SH

>>

J'EN

- Enclose the query in Parenthesis.
- The query should be a scalar query. The scalar query is the query with the result as one row and one column. Otherwise, the query will throw an error.
- If the query returns zero rows, then the variable is EMPTY, i.e., NULL.
- Reconsider our 'Guru99' table

## Example 1: When subquery return one row as a result.

```
DECLARE @COURSE_NAME VARCHAR (10)
SELECT @COURSE_NAME = (select Tutorial_name from Guru99 where Tutorial_ID = 1)
PRINT @COURSE_NAME
```

Value of **COURSE\_NAME** is displayed as SQL

Here, using **SELECT**, the scalar subquery is used to assign the value of query result to variable **COURSE\_NAME**. Here '1' is a one of the value in **Tutorial\_ID** column from 'Guru99' table. Then, **PRINT** is used to display the value of **COURSE\_NAME**.

(/images/1/030819\_0723\_SQLServerVa10.png).

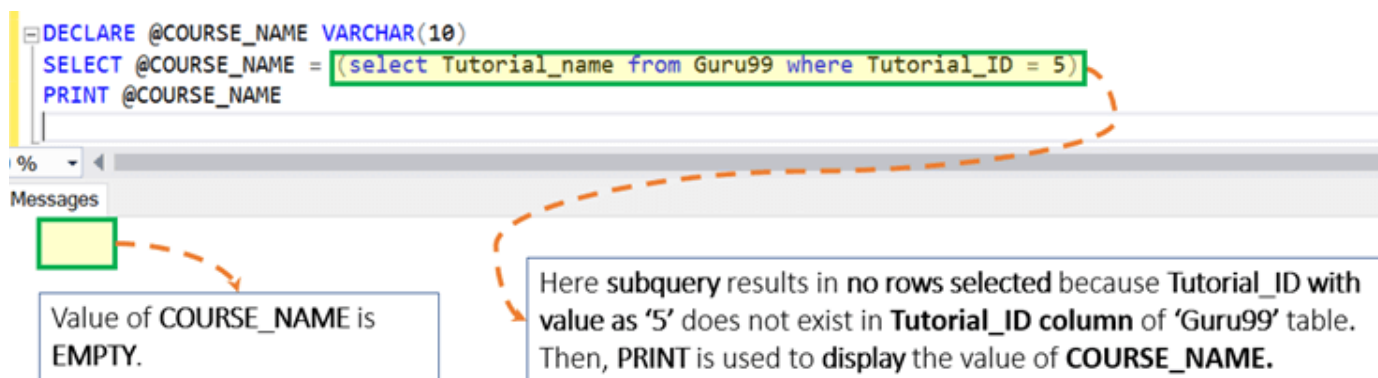
## Example 2: When subquery return zero row as a result

```

DECLARE @COURSE_NAME VARCHAR (10)
SELECT @COURSE_NAME = (select Tutorial_name from Guru99 where Tutorial_ID = 5)
PRINT @COURSE_NAME

```

In this particular case, the variable is to EMPTY, i.e., NULL.



[./images/1/030819\\_0723\\_SQLServerVa11.png](#)

**Example 3:** Assign a value to a variable with a regular SELECT statement.

**Syntax:**

```

DECLARE @Local_Variable _1 <Data_Type>, @Local_Variable _2 <Data_Type>, SELECT @Local_Variable _1 = <Column_1> from <Table_name> where <Condition_1>

```

**Rules:**

- Unlike SET, if the query results in multiple rows then the variable value is set to the value of the last row.
- If the query returns zero rows, then the variable is set to EMPTY, i.e., NULL.

**Query 1:** The query returns one row.

```

DECLARE @COURSE_NAME VARCHAR (10)
SELECT @COURSE_NAME = Tutorial_name from Guru99 where Tutorial_ID = 3
PRINT @COURSE_NAME

```

```

DECLARE @COURSE_NAME VARCHAR (10)
SELECT @COURSE_NAME = Tutorial_name from Guru99 where Tutorial_ID = 3
PRINT @COURSE_NAME

```

Value of COURSE\_NAME is 'MS-SQL'.

Here regular SELECT clause is used to assign the value to the variable. Variable **COURSE\_NAME** is set to value from **Tutorial\_name** column of 'Guru99' table where **Tutorial\_ID = 3**. Then, **PRINT** is used to **display** the value of **COURSE\_NAME**.

(./images/1/030819\_0723\_SQLServerVa12.png).

## Query 2: The query returns multiple rows.

```

DECLARE @COURSE_NAME VARCHAR (10)
SELECT @COURSE_NAME = Tutorial_name from Guru99
PRINT @COURSE_NAME

```

In this special case, variable value is **set to the value of the last row**.

```

DECLARE @COURSE_NAME VARCHAR(10)
SELECT @COURSE_NAME = Tutorial_name from Guru99;
PRINT @COURSE_NAME

```

Value of COURSE\_NAME is displayed as Hadoop.

Here regular SELECT clause is used to assign the value to the variable. As query start processing, variable **COURSE\_NAME** is set to the current row. With every next row, new value override the current value in **COURSE\_NAME**. Finally, the **last row value** is stored in **COURSE\_NAME** from 'Guru99' table. Then, **PRINT** is used to **display** the final value of **COURSE\_NAME**.

Tutorial_ID	Tutorial_name
1	SQL
2	PL/SQL
3	MS-SQL
4	Hadoop

(./images/1/030819\_0723\_SQLServerVa13.png).

## Query 3: The query returns zero rows.

```

DECLARE @COURSE_NAME VARCHAR (10)
SELECT @COURSE_NAME = Tutorial_name from Guru99 where Tutorial_ID = 5
PRINT @COURSE_NAME

```

In this particular case, the variable is **EMPTY**, i.e., **NULL**.

```

DECLARE @COURSE_NAME VARCHAR (10)
SELECT @COURSE_NAME = Tutorial_name from Guru99 where Tutorial_ID = 5
PRINT @COURSE_NAME

```

Value of **COURSE\_NAME** is EMPTY.

Here regular **SELECT** Clause is used to assign the value to the variable. Variable **COURSE\_NAME** is set to **EMPTY**, as no rows selected because **Tutorial\_ID** with value as '5' does not exist in **Tutorial\_ID** column of 'Guru99' table. Then, **PRINT** is used to display the value of **COURSE\_NAME**.

(/images/1/030819\_0723\_SQLServerVa14.png).

## Other Examples

Using variable in the query

Query:

```

DECLARE @COURSE_ID Int = 1
SELECT * from Guru99 where Tutorial_id = @COURSE_ID

```

```

DECLARE @COURSE_ID Int = 1
SELECT * from Guru99 where Tutorial_id = @COURSE_ID

```

Tutorial_ID	Tutorial_name
1	SQL

Row where **Tutorial\_ID** is 1 is displayed

Here variable **@COURSE\_ID** is used in **where** clause to query from 'Guru99' table.

(/images/1/030819\_0723\_SQLServerVa15.png).

## Interesting Facts!

- A local variable can be displayed using **PRINT** as well as **SELECT** COMMAND
- Table Data type doesn't allow the use of 'AS' during declaration.
- **SET** complies with **ANSI** standards whereas **SELECT** does not.
- Creating a local variable with the name as **@** is also allowed. We can declare it as, for example:

```
'DECLARE @@ as VARCHAR (10)'
```

## Summary:

- Variables are the object which acts as a placeholder.



- Two types of Variable exist: Local and Global
- We can assign the variable in the following three ways:

While using DECLARE

Using SET

USING SELECT

◀ Prev (/sql-server-datatype.html)

[Report a Bug](#)

Next ▶ (/sql-server-table-create-alter-drop.html)

Guru99 is Sponsored by AQUAFOLD



(<https://bit.ly/2GiKGdS>).

Aqua Data Studio est fier de parrainer le site Web informatif de Guru99 destiné aux professionnels des bases de données. Amenez votre productivité à de nouveaux sommets avec des outils de requête, des analyses visuelles et des diagrammes ER pour plusieurs plates-formes.

[EN SAVOIR PLUS SUR AQUA DATA STUDIO](#)

([HTTPS://BIT.LY/2GIKGDS](https://bit.ly/2GIKGDS)).



## Tutoriel SQL Server

- 2) [Téléchargez et installez SQL Server \(/download-install-sql-server.html\)](/download-install-sql-server.html)
- 3) [Architecture SQL Server \(/sql-server-architecture.html\)](/sql-server-architecture.html)
- 4) [SQL Server Management Studio \(/sql-server-management-studio.html\)](/sql-server-management-studio.html)
- 5) [base de données SQL Server \(/sql-server-database-create-alter-drop-restore.html\)](/sql-server-database-create-alter-drop-restore.html)
- 6) [SQL Server DataTypes \(/sql-server-datatype.html\)](/sql-server-datatype.html)
- 7) [Variable SQL Server \(/sql-server-variable.html\)](/sql-server-variable.html)
- 8) [Table SQL Server \(/sql-server-table-create-alter-drop.html\)](/sql-server-table-create-alter-drop.html)
- 9) [SQL Server PRIMARY KEY \(/sql-server-primary-key.html\)](/sql-server-primary-key.html)
- 10) [SQL Server FOREIGN KEY \(/sql-server-foreign-key.html\)](/sql-server-foreign-key.html)

[11\) SI... autre déclaration \(/sql-server-if-else.html\)](/sql-server-if-else.html)

[12\) déclaration de cas \(/sql-server-case.html\)](/sql-server-case.html)

**f** [\(https://www.facebook.com/guru99com/\)](https://www.facebook.com/guru99com/)

**t** [\\_ \(https://twitter.com/guru99com\)](https://twitter.com/guru99com) 

[. \(https://www.youtube.com/channel/UC19i1XD6k88KqHlET8atqFQ\)](https://www.youtube.com/channel/UC19i1XD6k88KqHlET8atqFQ)



[. \(https://forms.aweber.com/form/46/724807646.htm\)](https://forms.aweber.com/form/46/724807646.htm)

## À propos

[À propos de nous \(/about-us.html\)](/about-us.html)

[Annoncez avec nous \(/advertise-us.html\)](/advertise-us.html)

[Écrivez pour nous \(/become-an-instructor.html\)](/become-an-instructor.html)

[Contactez-nous \(/contact-us.html\)](/contact-us.html)

## Suggestion de carrière

[\(/best-sap-module.html\)](/best-sap-module.html) [Test du logiciel \(/software-testing-career-complete-guide.html\)](/software-testing-career-complete-guide.html) [SAP Career](#)

[Suggestion Tool en \(/best-sap-module.html\)](/best-sap-module.html)

[tant que carrière \(/software-testing-career-complete-guide.html\)](/software-testing-career-complete-guide.html)

## Intéressant

[Livres à lire! \(/books.html\)](/books.html) [EBook de \(/ebook-pdf.html\)](/ebook-pdf.html)

[Blog \(/blog/\)](/blog/)

[Quiz \(/tests.html\)](/tests.html)

[. \(/ebook-pdf.html\)](/ebook-pdf.html)

## Exécuter en ligne

[Execute Java Online \(/try-java-editor.html\)](/try-java-editor.html)

[Execute Javascript \(/execute-javascript-online.html\)](/execute-javascript-online.html)

[Execute HTML \(/execute-html-online.html\)](/execute-html-online.html)

[Execute Python \(/execute-python-online.html\)](/execute-python-online.html)