

# Chương 4. CÁC HÀM CHỨC NĂNG

- [4.1. Hàm thời gian](#)
- [4.2. Hàm toán học](#)
- [4.3. Hàm tổng hợp](#)
- [4.4. Hàm chuyển đổi](#)
- [4.5. Hàm chuỗi](#)

# 4.1. Hàm thời gian

## 4.1.1. ADD\_SECOND

Thêm một khoảng thời gian (tính bằng giây) vào một trường dữ liệu ngày tháng.

### Cú pháp:

**ADD\_SECOND** (truong\_du\_lieu\_thoi\_gian **DATETIME**, gia\_tri *INT*) => *DATETIME*

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.
- **gia\_tri:** Số giây cần thêm.

### Ví dụ:

```
SELECT ADD_SECOND('15-08-2019 10:01:30',30)
-- 15-08-2019 10:02:00
```

## 4.1.2. ADD\_MINUTE

Thêm một khoảng thời gian (tính bằng phút) vào một trường dữ liệu ngày tháng.

### Cú pháp:

**ADD\_MINUTE** (truong\_du\_lieu\_thoi\_gian *DATETIME*, gia\_tri *INT*) => *DATETIME*

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.
- **gia\_tri:** Số phút cần thêm.

### Ví dụ:

```
SELECT ADD_MINUTE('15-08-2019 10:01:30',30)
```

## 4.1.3. ADD\_HOUR

Thêm một khoảng thời gian (tính bằng giờ) vào một trường dữ liệu ngày tháng.

### Cú pháp:

**ADD\_HOUR** (truong\_du\_lieu\_thoi\_gian DATETIME, gia\_tri INT) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.
- **gia\_tri:** Số giờ cần thêm.

### Ví dụ:

```
SELECT ADD_HOUR('15-08-2019 10:01:30',5)
-- 15-08-2019 15:01:30
```

## 4.1.4. ADD\_DAY

Thêm một khoảng thời gian (tính bằng ngày) vào một trường dữ liệu ngày tháng.

### Cú pháp:

**ADD\_DAY** (truong\_du\_lieu\_thoi\_gian DATETIME, gia\_tri INT) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.
- **gia\_tri:** Số ngày cần thêm.

### Ví dụ:

```
SELECT ADD_DAY('15-08-2019 10:01:30',30)
-- 15-08-2019 10:01:30
```

## 4.1.5. ADD\_WEEK

Thêm một khoảng thời gian (tính bằng tuần) vào một trường dữ liệu ngày tháng.

### Cú pháp:

**ADD\_WEAK** (truong\_du\_lieu\_thoi\_gian DATETIME, gia\_tri INT) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.
- **gia\_tri:** Số tuần cần thêm.

### Ví dụ:

```
SELECT ADD_WEAK('15-08-2019 10:01:30',30)
-- 15-08-2019 10:02:00
```

## 4.1.6. ADD\_MONTH

Thêm một khoảng thời gian (tính bằng tháng) vào một trường dữ liệu ngày tháng.

### Cú pháp:

**ADD\_MONTH** (truong\_du\_lieu\_thoi\_gian DATETIME, gia\_tri INT) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.
- **gia\_tri:** Số tháng cần thêm.

### Ví dụ:

```
SELECT ADD_MONTH('15-08-2019 10:01:30',6)
-- 15-12-2019 10:01:30
```

## 4.1.7. ADD\_QUARTER

Thêm một khoảng thời gian (tính bằng quý) vào một trường dữ liệu ngày tháng.

### Cú pháp:

**ADD\_QUARTER** (truong\_du\_lieu\_thoi\_gian DATETIME, gia\_tri INT) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

- **gia\_tri:** Số quý cần thêm.

**Ví dụ:**

```
SELECT ADD_QUARTER('15-08-2019 10:01:30',1)
-- 15-11-2019 10:01:30
```

## 4.1.8. ADD\_YEAR

Thêm một khoảng thời gian (tính bằng năm) vào một trường dữ liệu ngày tháng.

**Cú pháp:**

**ADD\_YEAR** (truong\_du\_lieu\_thoi\_gian DATETIME, gia\_tri INT) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.
- **gia\_tri:** Số năm cần thêm.

**Ví dụ:**

```
SELECT ADD_YEAR('15-08-2019 10:01:30',5)
-- 15-08-2024 10:01:30
```

## 4.1.9. SECOND

Trả về giá trị giây (từ 0 -59) trong trường dữ liệu thời gian.

**Cú pháp:**

**SECOND** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT SECOND('15-08-2019 10:01:30')
-- 30
```

## 4.1.10. MINUTE

Trả về giá trị phút (từ 0 -59) trong trường dữ liệu thời gian.

**Cú pháp:**

**MINUTE** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT MINUTE('15-08-2019 10:01:30')  
-- 1
```

## 4.1.11. HOUR

Trả về giá trị giờ (từ 0- 23) trong trường dữ liệu thời gian.

**Cú pháp:**

**HOUR** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT HOUR('15-08-2019 10:01:30')  
-- 10
```

## 4.1.12. DAY

Trả về giá trị ngày (từ 1- 31) trong trường dữ liệu thời gian.

**Cú pháp:**

**DAY** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT DAY('15-08-2019 10:01:30')  
-- 15
```

## 4.1.13. MONTH

Trả về giá trị tháng (từ 1- 12) trong trường dữ liệu thời gian.

**Cú pháp:**

**MONTH** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT MONTH('15-08-2019 10:01:30')  
-- 8
```

## 4.1.14. QUARTER

Trả về giá trị quý (từ 1- 4) trong trường dữ liệu thời gian.

**Cú pháp:**

**QUARTER** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT QUARTER('15-08-2019 10:01:30')  
-- 3
```

## 4.1.15. YEAR

Trả về giá trị năm trong trường dữ liệu thời gian.

**Cú pháp:**

**YEAR** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT YEAR('15-08-2019 10:01:30')
-- 2019
```

## 4.1.16. LASTDAY

Trả về ngày cuối cùng trong tháng với trường dữ liệu thời gian.

**Cú pháp:**

**LASTDAY** (truong\_du\_lieu\_thoi\_gian DATETIME) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT LASTDAY('15-08-2019 10:01:30')
-- 31-08-2019 10:01:30
```

## 4.1.17. TRUNC\_SECOND

Trả về giá trị thời gian với độ chính xác được tính theo giây.

**Cú pháp:**

**TRUNC\_SECOND** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT TRUNC_SECOND('15-08-2019 10:01:30')  
-- 15-08-2019 10:01:30
```

## 4.1.18. TRUNC\_MINUTE

Trả về giá trị thời gian với độ chính xác được tính theo phút.

**Cú pháp:**

**TRUNC\_MINUTE** (truong\_du\_lieu\_thoi\_gian) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT TRUNC_MINUTE('15-08-2019 10:01:30')  
-- 15-08-2019 10:01:00
```

## 4.1.19. TRUNC\_HOUR

Trả về giá trị thời gian với độ chính xác được tính theo giờ.

**Cú pháp:**

**TRUNC\_HOUR** (truong\_du\_lieu\_thoi\_gian DATETIME) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT TRUNC_HOUR('15-08-2019 10:01:30')  
-- 15-08-2019 10:00:00
```

## 4.1.20. TRUNC\_DAY

Trả về giá trị thời gian với độ chính xác được tính theo ngày.

### Cú pháp:

**TRUNC\_DAY** (trung\_du\_lieu\_thoi\_gian DATETIME) => DATETIME

- **trung\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

### Ví dụ:

```
SELECT TRUNC_DAY ('15-08-2019 10:01:30')  
-- 15-08-2019 00:00:00
```

## 4.1.21. TRUNC\_WEEK

Trả về giá trị thời gian với độ chính xác được tính theo tuần.

### Cú pháp:

**TRUNC\_WEEK** (trung\_du\_lieu\_thoi\_gian DATETIME) => DATETIME

- **trung\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

### Ví dụ:

```
SELECT TRUNC_WEEK('15-08-2019 10:01:30')  
-- 04-08-2019 00:00:00
```

## 4.1.22. TRUNC\_MONTH

Trả về giá trị thời gian với độ chính xác được tính theo tháng.

### Cú pháp:

**TRUNC\_MONTH** (truong\_du\_lieu\_thoi\_gian DATETIME) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT TRUNC_MONTH('15-08-2019 10:01:30')  
-- 01-08-2019 00:00:00
```

## 4.1.23. TRUNC\_QUARTER

Trả về giá trị thời gian với độ chính xác được tính theo quý.

**Cú pháp:**

**TRUNC\_QUARTER** (truong\_du\_lieu\_thoi\_gian DATETIME) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT TRUNC_QUARTER('15-08-2019 10:01:30')  
-- 01-07-2019 00:00:00
```

## 4.1.24. TRUNC\_YEAR

Trả về giá trị thời gian với độ chính xác được tính theo năm.

**Cú pháp:**

**TRUNC\_YEAR** (truong\_du\_lieu\_thoi\_gian DATETIME) => DATETIME

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT TRUNC_YEAR('15-08-2019 10:01:30')  
-- 01-01-2019 00:00:00
```

## 4.1.25. DOM

Trả về thứ tự ngày trong tháng từ một biểu thức thời gian.

### Cú pháp:

**DOM** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

### Ví dụ:

```
SELECT DOM('15-08-2019 10:01:30')
-- 15
```

## 4.1.26. DOW

Trả về thứ tự ngày trong tuần từ một biểu thức thời gian.

### Cú pháp:

**DOW** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

```
SELECT DOW('15-08-2019 10:01:30')
-- 4(15-08-2019 là thứ 5, 4 là thứ tự của thứ 5 trong tuần)
```

## 4.1.27. DOY

Trả về thứ tự ngày trong năm từ một biểu thức thời gian.

### Cú pháp:

**DOY** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

```
SELECT DOY('15-08-2019 10:01:30')  
-- 227(Năm 2019 có 365 ngày, 15-08-2019 là ngày thứ 227)
```

## 4.1.28. WOY

Trả về thứ tự tuần trong năm từ một biểu thức thời gian.

**Cú pháp:**

**WOY** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

```
SELECT WOY('15-08-2019 10:01:30')  
-- 33(Một năm có 54 tuần, ngày 15-08-2019 là tuần thứ 33 trong năm)
```

## 4.1.29. YWEEK

Trả về thứ tự tuần trong năm từ một biểu thức thời gian.

**Cú pháp:**

**YWEEK** (truong\_du\_lieu\_thoi\_gian DATETIME) => INT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

**Ví dụ:**

```
SELECT YWEEK('15-08-2019 10:01:30')  
-- 33
```

## 4.1.30. EPOCH

Tính số giây đã trôi qua từ một mốc dữ liệu thời gian so với ngày 1 tháng 1 năm 1970 lúc 00:00:00 giờ UTC.

### Cú pháp:

**EPOCH** (truong\_du\_lieu\_thoi\_gian DATETIME) => BIGINT

- **truong\_du\_lieu\_thoi\_gian:** Biểu thức hoặc cột trả về kiểu dữ liệu thời gian/ngày tháng.

### Ví dụ:

```
SELECT EPOCH('24/06/2024 10:01:30')  
-- 1719223290
```

```
SELECT EPOCH('01/01/1970 00:00:00')  
-- 0
```

## 4.2. Hàm toán học

### 4.2.1. ABS

Tính giá trị tuyệt đối của một biểu thức số.

**Cú pháp:**

**ABS** (truong\_du\_lieu\_so Số) => Số

- **truong\_du\_lieu\_so:** Trường dữ liệu dạng số cần tính giá trị tuyệt đối.

**Ví dụ về ABS:**

```
SELECT ABS(-10)
-- 10
```

```
SELECT ABS(0)
-- 0
```

### 4.4.2. ACOS

Tính arccosine của một biểu thức số.

**Cú pháp:**

**ACOS** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu dạng số.

**Ví dụ về ACOS:**

```
SELECT ACOS(0)
-- 1.5707963267948966
```

```
SELECT ACOS(1.0)
```

```
-- 0
```

```
SELECT ACOS(-1)
```

```
-- 3.141592653589793
```

## 4.2.3. ADD

Tính giá trị của phép cộng giữa hai trường dữ liệu dạng số hoặc một trường dữ liệu và một số.

### Cú pháp:

**ADD** (truong\_du\_lieu\_so\_1 Số, truong\_du\_lieu\_so\_2 Số) => Số

- **truong\_du\_lieu\_so\_1:** DOUBLE, INTEGER, BIGINT, DECIMAL, hoặc FLOAT.
- **truong\_du\_lieu\_so\_2:** DOUBLE, INTEGER, BIGINT, DECIMAL, hoặc FLOAT.

### Ví dụ về ADD với dữ liệu số :

```
Select ADD(2, 3)
```

```
-- 6
```

### Ví dụ về ADD cột "col" có hai giá trị [2,3,4]:

```
Select ADD(col, 3)
```

```
-- 5
```

```
-- 6
```

```
-- 7
```

## 4.2.4. ASIN

Tính arcsine của một biểu thức số.

### Cú pháp:

**ASIN** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu dạng số.

#### Ví dụ về Asin:

```
SELECT ASIN(0)
-- 0.0
```

```
SELECT ASIN(1)
-- 1.5707963267948966
```

```
SELECT ASIN(-1)
-- -1.5707963267948966
```

## 4.2.5. ATAN

Tính arctang của một biểu thức số.

#### Cú pháp:

**ATAN** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu dạng số.

#### Ví dụ về ATAN:

```
SELECT ATAN(-1)
-- -0.7853981633974483
```

```
SELECT ATAN(19564.7)
-- 1.5707452143321894
```

## 4.2.6. ATAN2

Tính arctang của tỷ số giữa hai trường dữ liệu.

#### Cú pháp:

**ATAN2** (y Số, x Số) => DOUBLE

- **y:** Giá trị đầu vào kiểu số thực đại diện cho tọa độ y, nằm trong khoảng từ âm vô cùng đến dương vô cùng.
- **x:** Giá trị đầu vào kiểu số thực đại diện cho tọa độ x, nằm trong khoảng từ âm vô cùng đến dương vô cùng.

**Ví dụ về ATAN2:**

```
SELECT ATAN2(1,0)
-- 1.5707452143321894
```

```
SELECT ATAN2(0.0,1.0)
-- 0
```

```
SELECT ATAN2(0.0,-1.0)
-- 3.141592653589793
```

```
SELECT ATAN2(-0.000000000001,-1.0)
-- -3.141592653579793
```

## 4.2.7. CBRT

Tính căn bậc 3 của các hàng trong một trường dữ liệu.

**Cú pháp:**

**CBRT** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** trường dữ liệu dạng số cần tính căn bậc 3.

**Ví dụ về CBRT:**

```
SELECT CBRT(5)
-- 1.709975946676697
```

```
SELECT CBRT(120)
-- 4.932424148653812
```

```
SELECT CBRT(99.5)
-- 4.638049208321277
```

## 4.2.8. CEIL

Trả về giá trị bằng hoặc lớn hơn gần nhất nếu là số thập phân của các hàng trong trường dữ liệu đầu vào.

### Cú pháp:

**CEIL** (truong\_du\_lieu\_so Số) => INT

### Ví dụ về CEIL:

```
SELECT CEIL(37.775420706711)
-- 38
```

```
SELECT CEIL(3.1459)
-- 4
```

```
SELECT CEIL(-37.775420706711)
-- -37
```

```
SELECT CEIL(0)
-- 0
```

## 4.2.9. COS

Tính cosine của một biểu thức số.

### Cú pháp:

**COS** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu dạng số.

### Ví dụ về COS:

```
SELECT COS(0)
```

```
-- 1.0
```

```
SELECT COS(1.0)
```

```
-- 0.5403023058681398
```

```
SELECT COS(-1)
```

```
-- 0.5403023058681398
```

## 4.2.10. COSH

Tính hyperbolic cosine của một biểu thức số.

**Cú pháp:**

**COSH** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu dạng số.

**Ví dụ về COSH:**

```
SELECT COSH(0)
```

```
-- 1.0
```

```
SELECT COSH(1.0)
```

```
-- 1.543080634815244
```

```
SELECT COSH(-1)
```

```
-- 1.543080634815244
```

## 4.2.11. COT

Tính cotang của một biểu thức số.

**Cú pháp:**

**COT** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu dạng số.

**Ví dụ về COT:**

```
SELECT COT(0)
```

```
-- 1.0
```

```
SELECT COT(1.0)
```

```
-- 0.6420926159343306
```

```
SELECT COT(-1)
```

```
-- -0.6420926159343306
```

## 4.2.12. DEGRESS

Chuyển đổi đơn vị từ Radians sang Degress (Độ)

**Cú pháp:**

**DEGRESS** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và có kiểu dữ liệu dạng số.

**Ví dụ về DEGRESS:**

```
SELECT DEGRESS(0)
```

```
-- 0.0
```

```
SELECT DEGRESS(-1)
```

```
-- 57.29577951308232
```

## 4.2.13. DIV

Tính kết quả của phép chia giữa hai biểu thức đầu vào

### Cú pháp:

**DIV** (truong\_du\_lieu\_so\_1 Số, truong\_du\_lieu\_so\_2 Số) => Số

- **truong\_du\_lieu\_so\_1:** Số bị chia
- **truong\_du\_lieu\_so\_2:** Số chia, có thể là một số cụ thể hoặc một trường dữ liệu dạng số.

### Ví dụ về DIV:

```
SELECT DIV(6,2)
-- 3
```

```
SELECT DIV(revenue,order)
-- 3
-- 8
-- 9
```

## 4.2.14. DIVIDE

Tính kết quả của phép chia lấy số nguyên giữa hai biểu thức số

### Cú pháp:

**DIVIDE** (truong\_du\_lieu\_so\_1 Số, truong\_du\_lieu\_so\_2 Số) => Giá trị dạng số

- **truong\_du\_lieu\_so\_1:** Số bị chia
- **truong\_du\_lieu\_so\_2:** Số chia

### Ví dụ về DIVIDE:

```
SELECT DIVIDE(revenue,2)
-- 1
-- 9
-- 20
```

## 4.2.15. EXP

Tính lũy thừa của số e (cơ số của logarit tự nhiên, xấp xỉ 2.718281...) với số mũ lần lượt là các giá trị trong một trường dữ liệu.

### Cú pháp:

**EXP** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Giá trị số mũ để nâng e lên

### Ví dụ về EXP:

```
SELECT EXP(1)
-- 2.718281828459045
```

```
SELECT EXP(10.0)
-- 22026.465794806718
```

## 4.2.16. FACTORIAL

Tính giai thừa của một giá trị số.

### Cú pháp:

**FACTORIAL** (gia\_tri INT) => BIGINT

- **gia\_tri:** Giá trị số nguyên từ 0- 20 trong một trường dữ liệu.

### Ví dụ về FACTORIAL:

```
SELECT FACTORIAL(5)
-- 120
```

```
SELECT FACTORIAL(20)
-- 2432902008176640000
```

## 4.2.17. FLOOR

Trả về giá trị bằng hoặc nhỏ hơn gần nhất nếu là số thập phân của trường dữ liệu đầu vào.

### Cú pháp:

**FLOOR** (truong\_du\_lieu\_so Số) => INT

- **truong\_du\_lieu\_so:** Giá trị số lớn hơn 0

### Ví dụ về FLOOR:

```
SELECT FLOOR(0)
-- 0
```

```
SELECT CEIL(45.76)
-- 45
```

```
SELECT FLOOR(-1.3)
-- -2
```

## 4.2.18. LOG

Tính logarit tự nhiên của các giá trị trong một trường dữ liệu.

### Cú pháp:

**LOG** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Giá trị số lớn hơn 0.

### Ví dụ về LOG:

```
SELECT LOG(0)
-- null
```

```
SELECT LOG(1)
-- 0
```

```
SELECT LOG(5)
-- 1.6094379124341003
```

## 4.2.19. LOG10

Tính logarit cơ số 10 của trường dữ liệu đầu vào dạng số.

### Cú pháp:

**LOG10** (truong\_du\_lieu\_so Số) => Giá trị dạng số

- **truong\_du\_lieu\_so:** Giá trị số lớn hơn 0.

### Ví dụ về LOG10:

```
SELECT LOG10(20.5)
-- 1.3117538610557542
```

```
SELECT LOG10(100)
-- 2.0
```

## 4.2.20. LOG1P

Tính logarit tự nhiên của 1 cộng một giá trị trong trường dữ liệu dạng số.

### Cú pháp:

**LOG1P** (truong\_du\_lieu\_so Số) => Giá trị dạng số

- **truong\_du\_lieu\_so:** Giá trị số lớn hơn -1.

### Ví dụ về LOG1P

```
SELECT LOG1P(0)
-- 0
```

```
SELECT LOG1P(0.6931471805599453)
-- 0
```

```
SELECT LOG1P(5)
-- 1.791759469228055
```

## 4.2.21. MULTIPLY

Tính kết quả của phép nhân giữa các dòng trong hai trường dữ liệu dạng số.

**Cú pháp:**

**MULTIPLY** (truong\_du\_lieu\_so Số) => Giá trị dạng số

- **truong\_du\_lieu\_so:** Giá trị của biểu thức muốn tính logarit.

**Ví dụ về MULTIPLY:**

```
SELECT MULTIPLY(10,2)
-- 20
```

```
SELECT MULTIPLY(-2.0,2.0)
-- -4.0
```

```
SELECT MULTIPLY(0,2)
-- 0
```

## 4.2.22. RADIANS

Chuyển đổi đơn vị từ Degress (Độ) sang Radians

**Cú pháp:**

**RADIANS** (x Số) => DOUBLE

- **x:** Số có đơn vị là degress và kiểu dữ liệu là DOUBLE, INTEGER, BIGINT, DECIMAL, hoặc FLOAT.

**Ví dụ về RADIANS:**

```
SELECT RADIANS(45)
-- 0.7853981633974483
```

## 4.2.23. ROUND

Trả về giá trị sau khi làm tròn cho các giá trị đầu vào. Nếu không nhập số lượng chữ số thập phân số nguyên gần nhất sẽ được trả về.

### Cú pháp:

**ROUND** (truong\_du\_lieu\_so Số, Gia\_tri INT) => Số

- **truong\_du\_lieu\_so:** Số cần làm tròn.
- **gia\_tri:** Số lượng chữ số ở phần thập phân.

### Ví dụ về ROUND:

```
SELECT ROUND(24,0)
-- 24
```

```
SELECT ROUND(24,-2)
-- 0
```

```
SELECT ROUND(24.35,1)
-- 24.4
```

## 4.2.24. SIN

Tính sine của một biểu thức số.

### Cú pháp:

**SIN** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu là DOUBLE, INTEGER, DECIMAL, hoặc FLOAT.

### Ví dụ về SIN:

```
SELECT SIN(360)
-- 0.9589157234143065
```

```
SELECT SIN(510.89)
-- 0.9282211721815067
```

```
SELECT SIN(-1)
-- -0.8414709848078965
```

## 4.2.25. SINH

Tính hyperbolic sine của một biểu thức số.

### Cú pháp:

**SINH** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu dạng số.

### Ví dụ về SINH:

```
SELECT SINH(1)
-- 1.1752011936438014
```

```
SELECT SINH(1.5)
-- 2.1292794550948173
```

## 4.2.26. SQRT

Tính căn bậc hai của trường dữ liệu dạng số không âm.

### Cú pháp:

**SQRT** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Trường dữ liệu cần tính căn bậc hai

### Ví dụ về SQRT:

```
SELECT (25.25)
-- 5.024937810560445
```

```
SELECT (25)
-- 5.0
```

## 4.2.26. SUBTRACT

Tính kết quả của phép trừ giữa các dòng trong hai trường dữ liệu dạng số.

### Cú pháp:

**SUBTRACT** (truong\_du\_lieu\_so Số, so\_tru Số) => Giá trị dạng số

- **truong\_du\_lieu\_so:** Giá trị của trường dữ liệu được coi như số bị trừ.
- **so\_tru:** Số trừ, có thể là một trường dữ liệu dạng số hoặc một số cụ thể

### Ví dụ về SUBTRACT:

```
SELECT SUBTRACT (10,2)
-- 8
```

```
SELECT SUBTRACT (-2.0,2.0)
-- -4.0
```

```
SELECT SUBTRACT(0,2)
-- -2
```

## 4.2.28. TAN

Tính tang của một biểu thức số.

### Cú pháp:

**TAN** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu là DOUBLE, INTEGER, DECIMAL, hoặc FLOAT

### Ví dụ về TAN:

```
SELECT TAN(180.8)
-- -6.259341891872157
```

```
SELECT TAN(1200)
-- -0.08862461268886584
```

## 4.2.29. TANH

Tính hyperbolic tang của một biểu thức số.

### Cú pháp:

**TANH** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so:** Số có đơn vị là radians và kiểu dữ liệu là: DOUBLE, INTEGER, BIGINT, DECIMAL, hoặc FLOAT.

### Ví dụ về TANH:

```
SELECT TANH(1.5)
-- 0.9051482536448664
```

```
SELECT TANH(1)
-- 0.7615941559557649
```

## 4.3. Hàm tổng hợp

### 4.3.1. AVG

Tính giá trị trung bình của tất cả các giá trị một trường dữ liệu.

#### Cú pháp:

**AVG** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so**: Trường dữ liệu cần tính giá trị trung bình với kiểu dữ liệu là DOUBLE, INTEGER, BIGINT, DECIMAL, hoặc FLOAT.

#### Ví dụ về AVG:

```
SELECT AVG(1.5)
-- 1.5
```

#### Ví dụ về AVG: cột val[0.6348, -1.301466]:

```
SELECT AVG("val")
-- -0.333333
```

#### Ví dụ về AVG: với hàm GROUP BY:

```
SELECT mat_hang, AVG(san_luong)
FROM inet.sample
GROUP BY mat_hang
-- Cà rốt, 3000
-- Cam sành, 25.25
-- Ổi, 70
```

### 4.3.2. COUNT

Đếm tổng số lượng của các giá trị trong một trường dữ liệu.

### Cú pháp:

**COUNT** (Truong\_du\_lieu\_so Số) => BIGINT

- **Truong\_du\_lieu\_so**: Trường dữ liệu cần đếm với kiểu dữ liệu dạng số.

### Ví dụ về COUNT:

```
SELECT COUNT(mat_hang)
-- 10
```

### Ví dụ về COUNT: với hàm GROUP BY:

```
SELECT mat_hang, COUNT(san_luong)
FROM inet.sample
GROUP BY mat_hang
-- Cà rốt, 3
-- Cam sành, 5
-- Ổi, 2
```

## 4.3.3. COUNTDISTINCT

Đếm tổng số lượng của các giá trị trong một trường dữ liệu.

### Cú pháp:

**COUNT** (truong\_du\_lieu\_so Số) => BIGINT

- **truong\_du\_lieu\_so**: Trường dữ liệu cần đếm với kiểu dữ liệu dạng số

### Ví dụ về COUNT: với hàm GROUP BY:

```
SELECT mat_hang, COUNT(san_luong)
FROM inet.sample
GROUP BY mat_hang
-- Cà rốt, 3
-- Cam sành, 5
-- Ổi, 2
```

## 4.3.4. COUNTEXISTING

## 4.3.5. COUNTMISING

## 4.3.6. FIRST

## 4.3.7. FIRSTQUATILE

## 4.3.8. LAST

## 4.3.9. MAX

Tính giá trị lớn nhất của các giá trị trong một trường dữ liệu.

**Cú pháp:**

**MAX** (truong\_du\_lieu\_so Số) => BIGINT

- **truong\_du\_lieu\_so:** Trường dữ liệu cần đếm với kiểu dữ liệu dạng số

**Ví dụ về MAX:**

```
SELECT MAX(san_luong)
-- 300
```

**Ví dụ về MAX: với hàm GROUP BY:**

```
SELECT mat_hang, MAX(san_luong)
FROM inet.sample
GROUP BY mat_hang
-- Cà rốt, 300
-- Cam sành, 59
-- Ổi, 272
```

## 4.3.10.MEAN

## 4.3.11.MEDIAN

## 4.3.12.MIN

Tính giá trị nhỏ nhất của các giá trị trong một trường dữ liệu.

**Cú pháp:**

**MIN** (truong\_du\_lieu\_so Số) => BIGINT

- **truong\_du\_lieu\_so:** Trường dữ liệu cần đếm với kiểu dữ liệu dạng số

**Ví dụ về MIN:**

```
SELECT MIN(san_luong)
-- 10
```

**Ví dụ về MIN: với hàm GROUP BY:**

```
SELECT mat_hang, MIN(san_luong)
FROM inet.sample
GROUP BY mat_hang
-- Cà rốt, 46
-- Cam sành, 10
-- Ổi, 39
```

## 4.3.13.P25TH

## 4.3.14.P50TH

## 4.3.15.P75TH

## 4.3.16.P90TH

## 4.3.17.P99TH

## 4.3.18.SECONDQUARTILE

## 4.3.19.STD

## 4.3.20.SUM

Tính tổng của các giá trị trong một trường dữ liệu.

### Cú pháp:

**SUM** (truong\_du\_lieu\_so Số) => DOUBLE

- **truong\_du\_lieu\_so**: Trường dữ liệu tính tổng đếm với kiểu dữ liệu dạng số.

### Ví dụ về SUM:

```
SELECT SUM(san_luong)
-- 10
```

### Ví dụ về SUM: với hàm GROUP BY:

```
SELECT mat_hang, SUM(san_luong)
FROM inet.sample
GROUP BY mat_hang
-- Cà rốt, 46
-- Cam sành, 10
-- Ổi, 39
```

## 1.1.21.THIRDQUATILE

## 1.1.22.VARIANCE

## 4.4. Hàm chuyển đổi

### 4.4.1. CASTBIGINT

Chuyển đổi kiểu dữ liệu của một cột thành số nguyên BIGINT.

**Cú pháp:**

**CASTBIGINT** (truong\_du\_lieu\_so Số) => BIGINT

- truong\_du\_lieu\_so: trường dữ liệu cần chuyển với kiểu dữ liệu là DOUBLE, INTEGER, BIGINT, DECIMAL, hoặc FLOAT.

**Ví dụ về CASTBIGINT: cột col[1,2,10.5]:**

```
SELECT CASTBIGINT(col)
-- 1, 2, 10
```

### 4.4.2. CASTDECIMAL

Chuyển đổi kiểu dữ liệu của một cột thành số thực DECIMAL.

**Cú pháp:**

**CASTDECIMAL** (truong\_du\_lieu\_so Số) => DECIMAL

- truong\_du\_lieu\_so: Trường dữ liệu cần đếm với kiểu dữ liệu là DOUBLE, INTEGER, BIGINT, DECIMAL, hoặc FLOAT.

**Ví dụ về CASTDECIMAL: cột col[1,2,10.5]:**

```
SELECT CASTDECIMAL(col)
-- 1.0 ,2.0 , 10.5
```

### 4.4.3. CASTDOUBLE

Chuyển đổi kiểu dữ liệu của một cột thành nguyên Double.

**Cú pháp:**

**CASTDECIMAL** (truong\_du\_lieu\_so Số) => DOUBLE

- truong\_du\_lieu\_so: Trường dữ liệu cần đếm với kiểu dữ liệu là DOUBLE, INTEGER, BIGINT, DECIMAL, hoặc FLOAT.

**Ví dụ về CASTDECIMAL: cột col[1,2,10.5]:**

```
SELECT CASTDECIMAL(col)
-- 1.0 ,2.0 , 10.5
```

## 4.4.4. CASTFLOAT

Chuyển đổi kiểu dữ liệu của một cột thành số thực Float.

**Cú pháp:**

**CASTDECIMAL** (truong\_du\_lieu\_so Số) => FLOAT

- truong\_du\_lieu\_so: Trường dữ liệu cần đếm với kiểu dữ liệu là DOUBLE, INTEGER, BIGINT, DECIMAL, hoặc FLOAT.

**Ví dụ về CASTDECIMAL: cột col[1,2,10.5]:**

```
SELECT CASTDECIMAL(col)
-- 1.0 ,2.0 , 10.5
```

## 4.4.5. CASTINT

Chuyển đổi kiểu dữ liệu của một cột thành số nguyên Int.

**Cú pháp:**

**CASTDECIMAL** (truong\_du\_lieu\_so Số) => INT

- `truong_du_lieu_so`: Trường dữ liệu cần đếm với kiểu dữ liệu là `DOUBLE`, `INTEGER`, `BIGINT`, `DECIMAL`, hoặc `FLOAT`.

**Ví dụ về CASTDECIMAL: cột `col[1,2,10.5]`:**

```
SELECT CASTDECIMAL(col)
-- 1 ,2 , 10
```

## 4.4.6. CASTLONG

Chuyển đổi kiểu dữ liệu của một cột thành số nguyên Long.

**Cú pháp:**

**CASTDECIMAL** (`truong_du_lieu_so` Số) => LONG

- `truong_du_lieu_so`: Trường dữ liệu cần đếm với kiểu dữ liệu dạng số

**Ví dụ về CASTDECIMAL: cột `col[1,2,10.5]`:**

```
SELECT CASTDECIMAL(col)
-- 1, 2, 10
```

## 4.4.7. HASH

Trả về giá trị băm cho trường dữ liệu được truyền vào, hàm HASH có giá trị null dù cho giá trị đầu vào null

**Cú pháp:**

**HASH** (`truong_du_lieu`) => BIGINT

- `truong_du_lieu`: Giá trị được truyền vào để băm.

**Ví dụ về HASH:**

```
SELECT HASH('Ititan xin chào')
-- -1965350004
```

```
SELECT HASH('15/08/2019 10:01:30')
```

```
-- -44832748
```

# 4.5. Hàm chuỗi

## 4.5.1. ASCII

Chuyển kí tự dạng chuỗi thành mã ASCII tương ứng.

**Cú pháp:**

**BTRIM** (truong\_du\_lieu STRING) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để loại bỏ khoảng trắng.

**Ví dụ về ASCII:**

```
SELECT ASCII('iNet solutions')
```

```
-- 105(mã ASCII của ký tự 'i')
```

```
SELECT ASCII('W')
```

```
-- 87
```

## 4.5.2. BTRIM

Loại bỏ khoảng trắng ở đầu và cuối kí tự.

**Cú pháp:**

**BTRIM** (truong\_du\_lieu STRING) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để loại bỏ khoảng trắng.

**Ví dụ về BTRIM:**

```
SELECT BTRIM(' Ititan xin chào ')
```

```
-- Ititan xin chào
```

## 4.5.3. CONCAT

Nối hai hoặc nhiều chuỗi lại với nhau thành một chuỗi duy nhất.

### Cú pháp:

**CONCAT** (truong\_du\_lieu\_1 STRING, truong\_du\_lieu\_2 STRING) => Chuỗi

- **truong\_du\_lieu\_1:** Chuỗi ban đầu.
- **truong\_du\_lieu\_2:** Trường dữ liệu dạng chuỗi hoặc chuỗi.

### Ví dụ về CONCAT:

```
SELECT CONCAT('Ititan xin chào','iNet solutions')  
  
-- Ititan xin chàoiNet solutions
```

## 4.5.4. INITCAP

Chuyển đổi các chuỗi kí tự trong biểu thức với chữ cái đầu tiên của mỗi chữ được viết hoa.

### Cú pháp:

**INITCAP** (truong\_du\_lieu STRING) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để xử lý.

### Ví dụ về INITCAP:

```
SELECT INITCAP('ititan xin chào')  
  
-- Ititan Xin Chào
```

## 4.5.5. INSTR

Kiểm tra sự tồn tại của một chuỗi con bên trong một chuỗi lớn hơn.

**Cú pháp:**

**INITCAP** (truong\_du\_lieu STRING, chuoi\_con STRING) => BOOL

- **truong\_du\_lieu:** Chuỗi được truyền vào để xử lý.
- **chuoi\_con:** Chuỗi được truyền vào để so sánh

**Ví dụ về INSTR:**

```
SELECT INSTR('ititan xin chào','iti')
```

```
-- 1
```

```
SELECT INSTR('ititan xin chào','abc')
```

```
-- 0
```

## 4.5.6. LENGTH

Trả về độ dài của chuỗi kí tự.

**Cú pháp:**

**LENGTH** (truong\_du\_lieu STRING) => INT

- **truong\_du\_lieu:** Chuỗi được truyền vào để lấy độ dài.

**Ví dụ về LENGTH:**

```
SELECT LENGTH('Ititan xin chào')
```

```
-- 15
```

## 4.5.7. LOWER

Trả về tất cả các kí tự trong một chuỗi thành chữ không in hoa.

### Cú pháp:

**LOWER** (truong\_du\_lieu STRING) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để viết hoa.

### Ví dụ về LOWER:

```
SELECT LOWER('ITITAN XIN CHÀO')
```

```
-- ititan xin chào
```

## 4.5.8. LPAD

Thêm kí tự được chỉ định hoặc khoảng trắng cho đến khi chuỗi đạt đủ số lượng.

### Cú pháp:

**LPAD** (truong\_du\_lieu STRING, do\_dai INT, ki\_tu STRING) => Chuỗi

- **truong\_du\_lieu:** Chuỗi được truyền vào để thêm kí tự.
- **do\_dai:** Số lượng kí tự.
- **ki\_tu:** kí tự hoặc chuỗi để thêm.

### Ví dụ về LPAD:

```
SELECT LPAD('Ititan xin chào', 20, ' ')
```

```
--    Ititan xin chào
```

```
SELECT LPAD('iNet Solution', 18, 'i')
```

```
-- iiiiiNet Solution
```

**Lưu ý:** Nếu ki\_tu không được nhập vào, giá trị mặc định là khoảng trắng sẽ được thêm. Nếu chuỗi lớn hơn do\_dai sẽ trả về chuỗi gốc ban đầu.

## 4.5.9. LTRIM

Trả về giá trị lặp lại của các ký tự trong trường dữ liệu.

**Cú pháp:**

**LTRIM** (truong\_du\_lieu STRING) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để loại bỏ khoảng trắng.

**Ví dụ về LTRIM:**

```
SELECT LTRIM('iTitan xin chào')
```

```
-- iTitan xin chào
```

## 4.5.10. MASK\_FIRST

Giấu một số ký tự đầu tiên của trường dữ liệu dạng chuỗi.

**Cú pháp:**

**MASK\_FIRST** (truong\_du\_lieu STRING, do\_dai) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để loại bỏ khoảng trắng.
- **do\_dai:** số ký tự cần che giấu.

**Ví dụ về MASK\_FIRST:**

```
SELECT MASK_FIRST('Ititan xin chào',4)
```

```
-- Xxxxan xin chào
```

## 4.5.11. MASK\_LAST

Giấu một số ký tự đầu tiên của trường dữ liệu dạng chuỗi

### Cú pháp:

**MASK\_LAST** (truong\_du\_lieu String, do\_dai) => String

- **truong\_du\_lieu:** Chuỗi được truyền vào để loại bỏ khoảng trắng.
- **do\_dai:** số kí tự cần che giấu.

### Ví dụ về MASK\_LAST:

```
SELECT MASK_LAST('Ititan xin chào',5)
```

```
-- Ititan xinxxxxx
```

## 4.5.12. REPEAT

Trả về giá trị lặp lại của các kí tự trong trường dữ liệu.

### Cú pháp:

**REVERSE** (truong\_du\_lieu STRING, so\_lan\_lap INT) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để lặp.
- **so\_lan\_lap:** Số lần lặp lại các kí tự

### Ví dụ về REPEAT:

```
SELECT REPEAT('Ititan xin chào')
```

```
-- Ititan xin chàoItitan xin chàoItitan xin chào
```

## 4.5.13. REPLACE

Tìm kiếm và thay thế giá trị của một chuỗi kí tự.

### Cú pháp:

**REPLACE** (truong\_du\_lieu STRING, ki\_tu\_1 STRING, ki\_tu\_2 STRING) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để thêm kí tự.
- **ki\_tu\_1:** Kí tự tìm kiếm.
- **ki\_tu\_2:** Kí tự thay thế.

#### Ví dụ về REPLACE:

```
SELECT REPLACE('Ititan xin chào', 'i', 'y')
```

```
-- ltytan xyn chào
```

```
SELECT REPLACE('iNet Solution', 'Solution', 'xin chào')
```

```
-- iNet xin chào
```

## 4.5.14. REVERSE

Đảo ngược vị trí các kí tự của chuỗi trong biểu thức.

#### Cú pháp:

**REVERSE** (truong\_du\_lieu STRING) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để đảo ngược.

#### Ví dụ về REVERSE:

```
SELECT REVERSE('Ititan xin chào')
```

```
-- oàhC nix natitl
```

## 4.5.15. RTRIM

Loại bỏ các kí tự khoảng trắng ở cuối chuỗi kí tự.

#### Cú pháp:

**RTRIM** (truong\_du\_lieu STRING) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để loại bỏ khoảng trắng.

**Ví dụ về RTRIM:**

```
SELECT RTRIM('Ititan xin chào  ')
```

```
-- Ititan xin chào
```

## 4.5.16. STRPOS

Trả về vị trí lần đầu tiên xuất hiện của kí tự trong chuỗi.

**Cú pháp:**

**STRPOS** (truong\_du\_lieu STRING, ki\_tu STRING) => INT

- **truong\_du\_lieu:** Chuỗi được truyền vào để thêm kí tự.
- **ki\_tu:** Kí tự tìm kiếm.

**Ví dụ về STRPOS:**

```
SELECT STRPOS('Ititan xin chào', 'i')
```

```
-- 3
```

```
SELECT STRPOS('iNet Solution', 'iNet')
```

```
-- 1
```

```
SELECT STRPOS('iNet Solution', 'x')
```

```
-- 0
```

## 4.5.17. STR\_LEFT

Trích xuất một phần chuỗi kí tự bên trái của cột được truyền vào.

**Cú pháp:**

**STR\_LEFT** (truong\_du\_lieu STRING, do\_dai INT) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để trích xuất.
- **do\_dai:** Số kí tự muốn trích xuất.

**Ví dụ về STR\_LEFT:**

```
SELECT STR_LEFT('Ititan xin chào',3)

-- Iti
```

## 4.5.18. STR\_RIGHT

Trích xuất một phần chuỗi kí tự bên phải của cột được truyền vào.

**Cú pháp:**

**RTRIM** (truong\_du\_lieu STRING, do\_dai INT) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để trích xuất.
- **do\_dai:** Số kí tự muốn trích xuất.

**Ví dụ về RTRIM:**

```
SELECT RTRIM('Ititan xin chào',3)

-- hào
```

## 4.5.19. SUBSTR

Trích xuất một phần chuỗi kí tự bên của trường dữ liệu được truyền vào.

**Cú pháp:**

**STR\_LEFT** (truong\_du\_lieu STRING, vi\_tri INT, do\_dai INT) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để trích xuất.
- **do\_dai:** Số kí tự muốn trích xuất.

**Ví dụ về SUBSTR:**

```
SELECT SUBSTR('Ititan xin chào',3,4)
```

```
-- itan
```

## 4.5.20.UPPER

Trả về tất cả các kí tự trong một chuỗi thành chữ in hoa.

**Cú pháp:**

**UPPER** (truong\_du\_lieu STRING) => STRING

- **truong\_du\_lieu:** Chuỗi được truyền vào để in hoa.

**Ví dụ về UPPER:**

```
SELECT UPPER('Ititan xin chào')
```

```
-- ITITAN XIN CHÀO
```