

Встроенные функции и условная логика

Встроенные функции

Строки:

экранирование: дополнительная кавычка " или \
quote() + кавычки при выборке текста:

```
SELECT quote(text_fld) FROM string_tbl; -> 'This string didn\'t work, but it does now'
```

char() берет номера, и объединяет в строку

```
char(97,98,99) -> abc
```

concat() (+ в MS SQL) объединяет строки

```
UPDATE string_tbl SET text_fld = CONCAT(text_fld, 'but now it is longer');  
select concat(first_name, ' ', last_name, ' has been customer since ', date(create_date)) narrative from customer;
```

length() длина строки

position начало подстроки SELECT POSITION('characters' IN vchar_fld) FROM string_tbl; Первый с 1. 0 если не найдено.

locate как position, но 3 аргумент - старт поиска

insert вставка

replace заменяет

substring находит подстроку

объединение данных из группировки в столбец

```
SELECT id, GROUP_CONCAT(data) FROM yourtable GROUP BY id
```

Числа:

ceil(), floor() округление в большую или меньшую сторону к ближайшему целому числу

round() десятичная часть $\geq 0,5$ округлено в большую сторону и наоборот ROUND(72.0909,3)
- 3 знака оставить

TRUNCATE(72.0909, 1) - простое усечение, оставит 72.0

аргумент может быть < 0 TRUNCATE(17, -1) = 10, round(17, -1) = 20

sign() знак числа
abs() абсолютное значение

Даты:

cast() строку формата YYYY-MM-DD HH:MM:SS в дату

```
SELECT HOUR(@dt), MINUTE(@dt), SECOND(@dt), DAY(@dt), WEEK(@dt), MONTH(@dt), QUARTER(@dt),  
YEAR(@dt);
```

str_to_date(str, 'format') Формат:

| | |
|----|---|
| %M | Полное имя месяца (January..December) |
| %m | Числовое значение месяца |
| %d | Числовое значение дня месяца (00..31) |
| %j | День года (001..366) |
| %W | Полное имя дня недели (Sunday.Saturday) |
| %Y | Значение года (четыре цифры) |
| %y | Значение года (две цифры) |
| %H | Час дня в 24-часовом формате (00..23) |
| %h | Час дня в 12-часовом формате (01..12) |
| %i | Минуты в часе (00..59) |
| %s | Число секунд (00..59) |
| %f | Число микросекунд (000000..999999) |
| %p | АМ или РМ |
| %a | Краткое имя дня недели — Sun, Mon,... |
| %b | Краткое имя месяца — Jan,Feb,... |

CURRENT_DATE(), CURRENT_TIME(), CURRENT_TIMESTAMP()

interval

```
SELECT DATE ADD(CURRENT DATE(), INTERVAL 5 DAY)
```

последний день месяца

```
LAST_DAY('2019-09-17')
```

имя месяца

```
DAYNAME('2019-09-18')
```

извлекают элемент даты

```
EXTRACT(YEAR FROM '2019-09-18 22:19:05')
```

кол-во полных дней

```
DATEDIFF('2019-09-03', 2019-06-21)
```

```
SELECT  
CURRENT_TIME() AS 'CUR_TIME',  
ADDTIME(CURRENT_TIME(), 020000) AS 'ADDTIME',  
SUBTIME(CURRENT_TIME(), 020000) AS 'SUBTIME';
```

```
☐ CUR_TIME | ADDTIME | SUBTIME |
```

```
☐ 10:12:34 | 12:12:34 | 08:12:34 |
```

Условная логика

case:

```
SELECT c.first_name, c.last_name,  
CASE  
WHEN active = 0 THEN 0  
ELSE (select count(*) from rental r where r.customer_id = c.customer_id)  
END activity_type  
FROM customer c;
```

преимущество перед if-then в возможности использовать внутри select, insert, update, delete возвращаемые значения одного типа

Использование case для разделения по столбцам

```
SELECT  
☐SUM(CASE WHEN monthname(rental_date) = 'May' THEN 1 ELSE 0 END) May_rentals,  
☐SUM(CASE WHEN monthname(rental_date) = 'June' THEN 1 ELSE 0 END) June_rentals,  
☐SUM(CASE WHEN monthname(rental_date) = 'July' THEN 1 ELSE 0 END) July_rentals  
FROM rental WHERE rental date BETWEEN 2005-05-01' AND '2005-08-01';
```

использование case для получения факта участия актера в фильмах статуса G

```

SELECT a.first_name, a.last_name,
CASE
WHEN EXISTS (SELECT 1 FROM film_actor fa
INNER JOIN film f ON fa.film_id = f.film_id
WHERE fa.actor_id = a.actor_id
AND f.rating = 'G') THEN 'Y' ELSE 'N' END g_actor
FROM actor a WHERE a.last_name LIKE 'S%' OR a.first_name LIKE 'S%';

```

условные обновления

```

UPDATE customer
SET active = CASE
WHEN 90 <= (SELECT datediff(now(), max(rental_date) FROM rental r WHERE r.customer_id =
customer.customer_id))
THEN 0
ELSE 1
END
WHERE active = 1;

```

Аналитические функции

Постобработка сформированных данных. Использование: функция(столбец) over (дополнительное условие группировки)

| Функция | Описание |
|-------------|--|
| RANK | Аналогична функции DENSE_RANK() за исключением того, что в последовательности ранжированных значений есть пробелы, когда две или более строки имеют одинаковый ранг. |
| DENSE_RANK | Присваивает ранг каждой строке в своем разделе на основе предложения ORDER BY. Он присваивает одинаковый ранг строкам с одинаковыми значениями. Если две или более строки имеют одинаковый ранг, то в последовательности ранжированных значений не будет пробелов. |
| ROW_NUMBER | Назначает последовательное целое число каждой строке в своем разделе |
| CUME_DIST | Вычисляет совокупное распределение значения в наборе значений. |
| FIRST_VALUE | Возвращает значение указанного выражения относительно первой строки в рамке окна. |

| Функция | Описание |
|--------------|--|
| LAG | Возвращает значение N-й строки перед текущей строкой в разделе. Возвращает NULL, если предшествующей строки не существует. |
| LAST_VALUE | Возвращает значение указанного выражения относительно последней строки в рамке окна. |
| LEAD | Возвращает значение N-й строки после текущей строки в разделе. Возвращает NULL, если никакой последующей строки не существует. |
| NTH_VALUE | Возвращает значение аргумента из N-й строки рамки окна |
| NTILE | Распределяет строки для каждого раздела окна в указанное количество ранжированных групп. |
| PERCENT_RANK | Вычисляет процентильный ранг строки в разделе или наборе результатов |

Revision #3

Created 7 April 2025 05:45:35 by Admin

Updated 7 April 2025 13:38:59 by Admin