MySQL UNION 操作符 MySQL IFNULL() 函数

MySQL 函数

MySQL 有很多内置的函数，以下列出了这些函数的说明。

MySQL 字符串函数

函数	描述	实例
ASCII(s)	返回字符串 s 的第一个字符的 ASCII 码。	返回 CustomerName 字段第一个字母的 ASCII 码： `SELECT ASCII(CustomerName) AS NumCodeOfFirstChar FROM Customers;`
CHAR_LENGTH(s)	返回字符串 s 的字符数	返回字符串 RUNOOB 的字符数 `SELECT CHAR_LENGTH("RUNOOB") AS LengthOfString;`
CHARACTER_LENGTH(s)	返回字符串 s 的字符数	返回字符串 RUNOOB 的字符数 `SELECT CHARACTER_LENGTH("RUNOOB") AS LengthOfString;`
CONCAT(s1,s2...sn)	字符串 s1,s2 等多个字符串合并为一个字符串	合并多个字符串 `SELECT CONCAT("SQL ", "Runoob ", "Google ", "Facebook") AS ConcatenatedString;`
CONCAT_WS(x, s1,s2...sn)	同 CONCAT(s1,s2,...) 函数，但是每个字符串直接要加上 x，x 可以是分隔符	合并多个字符串，并添加分隔符： `SELECT CONCAT_WS("-", "SQL", "Tutorial", "is", "fun!")AS ConcatenatedString;`

FIELD(s,s1,s2...)	返回第一个字符串 s 在字符串列表(s1,s2...)中的位置	返回字符串 c 在列表值中的位置： `SELECT FIELD("c", "a", "b", "c", "d", "e");`
FIND_IN_SET(s1,s2)	返回在字符串s2中与s1匹配的字符串的位置	返回字符串 c 在指定字符串中的位置： `SELECT FIND_IN_SET("c", "a,b,c,d,e");`
FORMAT(x,n)	函数可以将数字 x 进行格式化 "#,###.##", 将 x 保留到小数点后 n 位，最后一位四舍五入。	格式化数字 "#,###.##" 形式： `SELECT FORMAT(250500.5634, 2);  -- 输出 250,500.56`
INSERT(s1,x,len,s2)	字符串 s2 替换 s1 的 x 位置开始长度为 len 的字符串	从字符串第一个位置开始的 6 个字符替换为 runoob： `SELECT INSERT("google.com", 1, 6, "runnob");  -- 输出：runoob.com`
LOCATE(s1,s)	从字符串 s 中获取 s1 的开始位置	获取 b 在字符串 abc 中的位置： `SELECT LOCATE('st','myteststring');  -- 5`
LCASE(s)	将字符串 s 的所有字母变成小写字母	字符串 RUNOOB 转换为小写： `SELECT LOWER('RUNOOB') -- runoob`
LEFT(s,n)	返回字符串 s 的前 n 个字符	返回字符串 runoob 中的前两个字符： `SELECT LEFT('runoob',2) -- ru`
LEFT(s,n)	返回字符串 s 的前 n 个字符	返回字符串 abcde 的前两个字符：

		`SELECT LEFT('abcde',2) -- ab`
LOCATE(s1,s)	从字符串 s 中获取 s1 的开始位置	返回字符串 abc 中 b 的位置： `SELECT LOCATE('b', 'abc') -- 2`
LOWER(s)	将字符串 s 的所有字母变成小写字母	字符串 RUNOOB 转换为小写： `SELECT LOWER('RUNOOB') -- runoob`
LPAD(s1,len,s2)	在字符串 s1 的开始处填充字符串 s2，使字符串长度达到 len	将字符串 xx 填充到 abc 字符串的开始处： `SELECT LPAD('abc',5,'xx') -- xxabc`
LTRIM(s)	去掉字符串 s 开始处的空格	去掉字符串 RUNOOB开始处的空格： `SELECT LTRIM("  RUNOOB") AS LeftTrimmedString;-- RUNOOB`
MID(s,n,len)	从字符串 s 的 start 位置截取长度为 length 的子字符串，同 SUBSTRING(s,n,len)	从字符串 RUNOOB 中的第 2 个位置截取 3个 字符： `SELECT MID("RUNOOB", 2, 3) AS ExtractString; -- UNO`
POSITION(s1 IN s)	从字符串 s 中获取 s1 的开始位置	返回字符串 abc 中 b 的位置： `SELECT POSITION('b' in 'abc') -- 2`
REPEAT(s,n)	将字符串 s 重复 n 次	将字符串 runoob 重复三次： `SELECT REPEAT('runoob',3) -- runoobrunoobrunoob`
REPLACE(s,s1,s2)	将字符串 s2 替代字符串 s 中的字符串 s1	将字符串 abc 中的字符 a 替换为字符 x：

		`SELECT REPLACE('abc','a','x') --xbc`
REVERSE(s)	将字符串s的顺序反过来	将字符串 abc 的顺序反过来： `SELECT REVERSE('abc') -- cba`
RIGHT(s,n)	返回字符串 s 的后 n 个字符	返回字符串 runoob 的后两个字符： `SELECT RIGHT('runoob',2) -- ob`
RPAD(s1,len,s2)	在字符串 s1 的结尾处添加字符串 s2，使字符串的长度达到 len	将字符串 xx 填充到 abc 字符串的结尾处： `SELECT RPAD('abc',5,'xx') -- abcxx`
RTRIM(s)	去掉字符串 s 结尾处的空格	去掉字符串 RUNOOB 的末尾空格： `SELECT RTRIM("RUNOOB ") AS RightTrimmedString; -- RUNOOB`
SPACE(n)	返回 n 个空格	返回 10 个空格： `SELECT SPACE(10);`

STRCMP(s1,s2)	比较字符串 s1 和 s2，如果 s1 与 s2 相等返回 0 ，如果 s1>s2 返回 1，如果 s1<s2 返回 -1	比较字符串：
		`SELECT STRCMP("runoob", "runoob");`
		`-- 0`

SUBSTR(s, start, length)	从字符串 s 的 start 位置截取长度为 length 的子字符串	从字符串 RUNOOB 中的第 2 个位置截取 3个 字符： `SELECT SUBSTR("RUNOOB", 2, 3) AS ExtractString; -- UNO`
SUBSTRING(s, start,	从字符串 s 的 start 位置截取长度为 length	从字符串 RUNOOB 中的第 2 个位置截取 3个 字

length)	的子字符串	符： `SELECT SUBSTRING("RUNOOB", 2, 3) AS ExtractString; -- UNO`

SUBSTRING_INDEX(s, delimiter, number)	返回从字符串 s 的第 number 个出现的分隔符 delimiter 之后的子串。
	如果 number 是正数，返回第 number 个字符左边的字符串。
	如果 number 是负数，返回第(number 的绝对值(从右边数))个字符右边的字符串。
		`SELECT SUBSTRING_INDEX('a*b','*',1) -- a`
		`SELECT SUBSTRING_INDEX('a*b','*',-1) -- b`
		`SELECT SUBSTRING_INDEX(SUBSTRING_INDEX('a*b*c*d*e','*',3),'*',-1) -- c`

TRIM(s)	去掉字符串 s 开始和结尾处的空格	去掉字符串 RUNOOB 的首尾空格： `SELECT TRIM('  RUNOOB  ') AS TrimmedString;`
UCASE(s)	将字符串转换为大写	将字符串 runoob 转换为大写： `SELECT UCASE("runoob"); -- RUNOOB`
UPPER(s)	将字符串转换为大写	将字符串 runoob 转换为大写： `SELECT UPPER("runoob"); -- RUNOOB`

MySQL 数字函数

函数名	描述	实例
ABS(x)	返回 x 的绝对值	返回 -1 的绝对值： `SELECT ABS(-1) -- 返回1`
ACOS(x)	求 x 的反余弦值(参数是弧度)	`SELECT ACOS(0.25);`

ASIN(x)	求反正弦值(参数是弧度)	`SELECT ASIN(0.25);`
ATAN(x)	求反正切值(参数是弧度)	`SELECT ATAN(2.5);`
ATAN2(n, m)	求反正切值(参数是弧度)	`SELECT ATAN2(-0.8, 2);`
AVG(expression)	返回一个表达式的平均值，expression 是一个字段	返回 Products 表中Price 字段的平均值： `SELECT AVG(Price) AS AveragePrice FROM Products;`
CEIL(x)	返回大于或等于 x 的最小整数	`SELECT CEIL(1.5) -- 返回2`
CEILING(x)	返回大于或等于 x 的最小整数	`SELECT CEIL(1.5) -- 返回2`
COS(x)	求余弦值(参数是弧度)	`SELECT COS(2);`
COT(x)	求余切值(参数是弧度)	`SELECT COT(6);`
COUNT(expression)	返回查询的记录总数，expression 参数是一个字段或者 * 号	返回 Products 表中 products 字段总共有多少条记录： `SELECT COUNT(ProductID) AS NumberOfProducts FROM Products;`
DEGREES(x)	将弧度转换为角度	`SELECT DEGREES(3.1415926535898) -- 180`

n DIV m	整除，n 为被除数，m 为除数	计算 10 除于 5： `SELECT 10 DIV 5;  -- 2`
EXP(x)	返回 e 的 x 次方	计算 e 的三次方： `SELECT EXP(3) -- 20.085536923188`
FLOOR(x)	返回小于或等于 x 的最大整数	小于或等于 1.5 的整数： `SELECT FLOOR(1.5) -- 返回1`

GREATEST(expr1, expr2, expr3, ...)	返回列表中的最大值	返回以下数字列表中的最大值：
		`SELECT GREATEST(3, 12, 34, 8, 25); -- 34`
		返回以下字符串列表中的最大值：
		`SELECT GREATEST("Google", "Runoob", "Apple");  -- Runoob`

LEAST(expr1, expr2, expr3, ...)	返回列表中的最小值	返回以下数字列表中的最小值：
		`SELECT LEAST(3, 12, 34, 8, 25);  -- 3`
		返回以下字符串列表中的最小值：
		`SELECT LEAST("Google", "Runoob", "Apple");  -- Apple`

LN	返回数字的自然对数	返回 2 的自然对数： `SELECT LN(2);  -- 0.6931471805599453`

LOG(x)	返回自然对数(以 e 为底的对数)	`SELECT LOG(20.085536923188) -- 3`
LOG10(x)	返回以 10 为底的对数	`SELECT LOG10(100) -- 2`
LOG2(x)	返回以 2 为底的对数	返回以 2 为底 6 的对数： `SELECT LOG2(6);  -- 2.584962500721156`
MAX(expression)	返回字段 expression 中的最大值	返回数据表 Products 中字段 Price 的最大值： `SELECT MAX(Price) AS LargestPrice FROM Products;`
MIN(expression)	返回字段 expression 中的最小值	返回数据表 Products 中字段 Price 的最小值： `SELECT MIN(Price) AS LargestPrice FROM Products;`
MOD(x,y)	返回 x 除以 y 以后的余数	5 除于 2 的余数： `SELECT MOD(5,2) -- 1`
PI()	返回圆周率(3.141593)	`SELECT PI() --3.141593`
POW(x,y)	返回 x 的 y 次方	2 的 3 次方： `SELECT POW(2,3) -- 8`

POWER(x,y)	返回 x 的 y 次方	2 的 3 次方： `SELECT POWER(2,3) -- 8`
RADIANS(x)	将角度转换为弧度	180 度转换为弧度： `SELECT RADIANS(180) -- 3.1415926535898`
RAND()	返回 0 到 1 的随机数	`SELECT RAND() --0.93099315644334`
ROUND(x)	返回离 x 最近的整数	`SELECT ROUND(1.23456) --1`
SIGN(x)	返回 x 的符号，x 是负数、0、正数分别返回 -1、0 和 1	`SELECT SIGN(-10) -- (-1)`
SIN(x)	求正弦值(参数是弧度)	`SELECT SIN(RADIANS(30)) -- 0.5`
SQRT(x)	返回x的平方根	25 的平方根： `SELECT SQRT(25) -- 5`
SUM(expression)	返回指定字段的总和	计算 OrderDetails 表中字段 Quantity 的总和： `SELECT SUM(Quantity) AS TotalItemsOrdered FROM OrderDetails;`
TAN(x)	求正切值(参数是弧度)	`SELECT TAN(1.75);  -- -5.52037992250933`

TRUNCATE(x,y)	返回数值 x 保留到小数点后 y 位的值（与 ROUND 最大的区别是不会进行四舍五入）	`SELECT TRUNCATE(1.23456,3) -- 1.234`

MySQL 日期函数

函数名	描述	实例
ADDDATE(d,n)	计算起始日期 d 加上 n 天的日期	`SELECT ADDDATE("2017-06-15", INTERVAL 10 DAY);` `->2017-06-25`
ADDTIME(t,n)	时间 t 加上 n 秒的时间	`SELECT ADDTIME('2011-11-11 11:11:11', 5)` `->2011-11-11 11:11:16 (秒)`
CURDATE()	返回当前日期	`SELECT CURDATE();` `-> 2018-09-19`
CURRENT_DATE()	返回当前日期	`SELECT CURRENT_DATE();` `-> 2018-09-19`
CURRENT_TIME	返回当前时间	`SELECT CURRENT_TIME();` `-> 19:59:02`
CURRENT_TIMESTAMP()	返回当前日期和时间	`SELECT CURRENT_TIMESTAMP()` `-> 2018-09-19 20:57:43`
CURTIME()	返回当前时间	`SELECT CURTIME();` `-> 19:59:02`

DATE()	从日期或日期时间表达式中提取日期值	`SELECT DATE("2017-06-15");` `-> 2017-06-15`
DATEDIFF(d1,d2)	计算日期 d1->d2 之间相隔的天数	`SELECT DATEDIFF('2001-01-01','2001-02-02')` `-> -32`

DATE_ADD(d，INTERVAL expr type)	计算起始日期 d 加上一个时间段后的日期	`SELECT ADDDATE('2011-11-11 11:11:11',1)`
		`-> 2011-11-12 11:11:11 (默认是天)`
		
		`SELECT ADDDATE('2011-11-11 11:11:11', INTERVAL 5 MINUTE)`
		`-> 2011-11-11 11:16:11 (TYPE的取值与上面那个列出来的函数类似)`

DATE_FORMAT(d,f)	按表达式 f的要求显示日期 d	`SELECT DATE_FORMAT('2011-11-11 11:11:11','%Y-%m-%d %r')` `-> 2011-11-11 11:11:11 AM`

DATE_SUB(date,INTERVAL expr type)	函数从日期减去指定的时间间隔。	Orders 表中 OrderDate 字段减去 2 天：
		
		`SELECT OrderId,DATE_SUB(OrderDate,INTERVAL 2 DAY) AS OrderPayDate`
		`FROM Orders`

DAY(d)	返回日期值 d 的日期部分	`SELECT DAY("2017-06-15");` `-> 15`
DAYNAME(d)	返回日期 d 是星期几，如 Monday,Tuesday	`SELECT DAYNAME('2011-11-11 11:11:11')` `->Friday`

DAYOFMONTH(d)	计算日期 d 是本月的第几天	`SELECT DAYOFMONTH('2011-11-11 11:11:11')` `->11`
DAYOFWEEK(d)	日期 d 今天是星期几，1 星期日，2 星期一，以此类推	`SELECT DAYOFWEEK('2011-11-11 11:11:11')` `->6`
DAYOFYEAR(d)	计算日期 d 是本年的第几天	`SELECT DAYOFYEAR('2011-11-11 11:11:11')` `->315`
EXTRACT(type FROM d)	从日期 d 中获取指定的值，type 指定返回的值。 type可取值为： • MICROSECOND • SECOND • MINUTE • HOUR • DAY • WEEK • MONTH • QUARTER • YEAR • SECOND_MICROSECOND • MINUTE_MICROSECOND • MINUTE_SECOND • HOUR_MICROSECOND • HOUR_SECOND • HOUR_MINUTE • DAY_MICROSECOND	`SELECT EXTRACT(MINUTE FROM '2011-11-11 11:11:11')` `-> 11`

	• DAY_SECOND • DAY_MINUTE • DAY_HOUR • YEAR_MONTH	
FROM_DAYS(n)	计算从 0000 年 1 月 1 日开始 n 天后的日期	`SELECT FROM_DAYS(1111)` `-> 0003-01-16`
HOUR(t)	返回 t 中的小时值	`SELECT HOUR('1:2:3')` `-> 1`
LAST_DAY(d)	返回给给定日期的那一月份的最后一天	`SELECT LAST_DAY("2017-06-20");` `-> 2017-06-30`
LOCALTIME()	返回当前日期和时间	`SELECT LOCALTIME()` `-> 2018-09-19 20:57:43`
LOCALTIMESTAMP()	返回当前日期和时间	`SELECT LOCALTIMESTAMP()` `-> 2018-09-19 20:57:43`
MAKEDATE(year, day-of-year)	基于给定参数年份 year 和所在年中的天数序号 day-of-year 返回一个日期	`SELECT MAKEDATE(2017, 3);` `-> 2017-01-03`
MAKETIME(hour, minute, second)	组合时间，参数分别为小时、分钟、秒	`SELECT MAKETIME(11, 35, 4);` `-> 11:35:04`
MICROSECOND(date)	返回日期参数所对应的毫秒数	`SELECT MICROSECOND("2017-06-20 09:34:00.000023");` `-> 23`

MINUTE(t)	返回 t 中的分钟值	`SELECT MINUTE('1:2:3')` `-> 2`
MONTHNAME(d)	返回日期当中的月份名称，如 Janyary	`SELECT MONTHNAME('2011-11-11 11:11:11')` `-> November`
MONTH(d)	返回日期d中的月份值，1 到 12	`SELECT MONTH('2011-11-11 11:11:11')` `->11`
NOW()	返回当前日期和时间	`SELECT NOW()` `-> 2018-09-19 20:57:43`
PERIOD_ADD(period, number)	为 年-月 组合日期添加一个时段	`SELECT PERIOD_ADD(201703, 5);` `-> 201708`
PERIOD_DIFF(period1, period2)	返回两个时段之间的月份差值	`SELECT PERIOD_DIFF(201710, 201703);` `-> 7`
QUARTER(d)	返回日期d是第几季节，返回 1 到 4	`SELECT QUARTER('2011-11-11 11:11:11')` `-> 4`
SECOND(t)	返回 t 中的秒钟值	`SELECT SECOND('1:2:3')` `-> 3`
SEC_TO_TIME(s)	将以秒为单位的时间 s 转换为时分秒的格式	`SELECT SEC_TO_TIME(4320)`

		-> 01:12:00
STR_TO_DATE(string, format_mask)	将字符串转变为日期	SELECT STR_TO_DATE("August 10 2017", "%M %d %Y"); -> 2017-08-10
SUBDATE(d,n)	日期 d 减去 n 天后的日期	SELECT SUBDATE('2011-11-11 11:11:11', 1) ->2011-11-10 11:11:11 (默认是天)
SUBTIME(t,n)	时间 t 减去 n 秒的时间	SELECT SUBTIME('2011-11-11 11:11:11', 5) ->2011-11-11 11:11:06 (秒)
SYSDATE()	返回当前日期和时间	SELECT SYSDATE() -> 2018-09-19 20:57:43
TIME(expression)	提取传入表达式的时间部分	SELECT TIME("19:30:10"); -> 19:30:10
TIME_FORMAT(t,f)	按表达式 f 的要求显示时间 t	SELECT TIME_FORMAT('11:11:11','%r') 11:11:11 AM
TIME_TO_SEC(t)	将时间 t 转换为秒	SELECT TIME_TO_SEC('1:12:00') -> 4320
TIMEDIFF(time1, time2)	计算时间差值	SELECT TIMEDIFF("13:10:11", "13:10:10"); -> 00:00:01

TIMESTAMP(expression, interval)	单个参数时，函数返回日期或日期时间表达式；有2个参数时，将参数加和	`SELECT TIMESTAMP("2017-07-23", "13:10:11");` `-> 2017-07-23 13:10:11`
TO_DAYS(d)	计算日期 d 距离 0000 年 1 月 1 日的天数	`SELECT TO_DAYS('0001-01-01 01:01:01')` `-> 366`
WEEK(d)	计算日期 d 是本年的第几个星期，范围是 0 到 53	`SELECT WEEK('2011-11-11 11:11:11')` `-> 45`
WEEKDAY(d)	日期 d 是星期几，0 表示星期一，1 表示星期二	`SELECT WEEKDAY("2017-06-15");` `-> 3`
WEEKOFYEAR(d)	计算日期 d 是本年的第几个星期，范围是 0 到 53	`SELECT WEEKOFYEAR('2011-11-11 11:11:11')` `-> 45`
YEAR(d)	返回年份	`SELECT YEAR("2017-06-15");` `-> 2017`
YEARWEEK(date, mode)	返回年份及第几周（0到53），mode 中 0 表示周天，1表示周一，以此类推	`SELECT YEARWEEK("2017-06-15");` `-> 201724`

MySQL 高级函数

函数名	描述	实例
BIN(x)	返回 x 的二进制编码	15 的 2 进制编码:

		`SELECT BIN(15); -- 1111`
BINARY(s)	将字符串 s 转换为二进制字符串	`SELECT BINARY "RUNOOB";` `-> RUNOOB`

`CASE expression`
`WHEN condition1`
`THEN result1`
`WHEN condition2`
`THEN result2`
`...`
`WHEN conditionN`
`THEN resultN`
`ELSE result`
`END`
	CASE 表示函数开始，END 表示函数结束。如果 condition1 成立，则返回 result1, 如果 condition2 成立，则返回 result2，当全部不成立则返回 result，而当有一个成立之后，后面的就不执行了。	`SELECT CASE`
		`WHEN 1 > 0`
		`THEN '1 > 0'`
		`WHEN 2 > 0`
		`THEN '2 > 0'`
		`ELSE '3 > 0'`
		`END`
		`->1 > 0`

CAST(x AS type)	转换数据类型	字符串日期转换为日期：
		`SELECT CAST("2017-08-29" AS DATE);`
		`-> 2017-08-29`

COALESCE(expr1, expr2, ..., expr_n)	返回参数中的第一个非空表达式（从左向右）	`SELECT COALESCE(NULL, NULL, NULL, 'runoob.com', NULL, 'google.com');` `-> runoob.com`
CONNECTION_ID()	返回服务器的连接数	`SELECT CONNECTION_ID();` `-> 4292835`
CONV(x,f1,f2)	返回 f1 进制数变成 f2 进制数	`SELECT CONV(15, 10`

		, 2); -> 1111
CONVERT(s USING cs)	函数将字符串 s 的字符集变成 cs	SELECT CHARSET('ABC') ->utf-8 SELECT CHARSET(CONVERT('ABC' USING gbk)) ->gbk
CURRENT_USER()	返回当前用户	SELECT CURRENT_USER(); -> guest@%
DATABASE()	返回当前数据库名	SELECT DATABASE(); -> runoob
IF(expr,v1,v2)	如果表达式 expr 成立，返回结果 v1；否则，返回结果 v2。	SELECT IF(1 > 0,'正确','错误') ->正确
IFNULL(v1,v2)	如果 v1 的值不为 NULL，则返回 v1，否则返回 v2。	SELECT IFNULL(null,'Hello Word') ->Hello Word
ISNULL(expression)	判断表达式是否为 NULL	SELECT ISNULL(NULL); ->1
LAST_INSERT_ID()	返回最近生成的 AUTO_INCREMENT 值	

		SELECT LAST_INSERT_ID(); ->6
NULLIF(expr1, expr2)	比较两个字符串，如果字符串 expr1 与 expr2 相等 返回 NULL，否则返回 expr1	SELECT NULLIF(25, 25); ->
SESSION_USER()	返回当前用户	SELECT SESSION_USER(); -> guest@%
SYSTEM_USER()	返回当前用户	SELECT SYSTEM_USER(); -> guest@%
USER()	返回当前用户	SELECT USER(); -> guest@%
VERSION()	返回数据库的版本号	SELECT VERSION() -> 5.6.34

← MySQL UNION 操作符 MySQL IFNULL() 函数 →

1 篇笔记

写笔记

需求：将数据库中每分钟一条的数据表，从 9:30 取到 22:00 ，以半小时为单位汇总，并输出 Excel。

数据表字段：id（序号）、incount（计数）、cdate（数据时间）

表名：m_temp

难点：时间处理

解决办法：使用 DATE_FORMAT、CONCAT、Date、Hour、Minute、Floor 函数将时间处理成半小时，在将这段时间的数据查询时 输出 cdate 统一变为 09:30，然后在 group by cdate ，用 sum（incount）求和。

1.处理时间：

示例：2018-08-21 09:30:00 至 2018-08-21 22:00:00

第一个半小时 09:30 至 10:00

将这段时间的数据查询时 输出 cdate 统一变为 09:30，然后在 group by cdate ，sum（incount）

将分钟取出 /30 ，0-29 分结果为 0 ，30-59分结果为 1 再乘以 30 分钟即变为 00 或者 30

date(cdate)，取出年月日；

hour(cdate)，取出小时；

minute(cdate)，取出分钟；

计算（minute(cdate)）/30)*30，结果为 0 或30；

floor(（minute(cdate)）/30)*30)， 处理成整数；

concat(date(cdate),' ',hour(cdate),':',floor(（minute(cdate)）/30)*30) ，按照日期格式进行拼接

DATE_FORMAT(concat(date(cdate),' ',hour(cdate),':',floor(minute(cdate)/30)*30+12) ,'%Y-%m-%d %H:%i') ，转换成date类型

完整 SQL：

```
select sum(incount),dataStartTime from (
select incount, DATE_FORMAT( concat(date(cdate),' ',hour(cdate),':',floor( minute(cdate)/
30 )*30+12) ,'%Y-%m-%d %H:%i')  as dataStartTime
 from m_temp WHERE cdate>='2018-08-21 09:30' and cdate<='2018-08-21 22:00'  ORDER BY data
StartTime
) a
group by DATE_FORMAT( dataStartTime ,'%Y-%m-%d %H:%i')
into outfile('c:/2018-08-21Data.xls');
```

宋新峰 2周前 (03-05)