Table of Contents

[1. Row filters 1](#_Toc391497655)

[2. Testing for nulls 2](#_Toc391497656)

[3. The IN list test 3](#_Toc391497657)

[3.1. Simple In lists 3](#_Toc391497658)

[3.2. In lists that contain nulls 5](#_Toc391497659)

[3.3. In lists that contain row values 5](#_Toc391497660)

[3.4. Testing a literal against an in list 6](#_Toc391497661)

[4. The BETWEEN test 7](#_Toc391497662)

[4.1. Gotchas 8](#_Toc391497663)

1. Row filters

Most of our queries so far have returned all of the data from the table in the From clause. We are working with very small tables. Imagine the output if we had thousands of rows in our tables. We seldom want to see all of the data in a table. Rather we want to see only a subset of the data- a subset that matches some search condition. If you are using the CCSF WebStars system to see your current schedule- you do not want to see everyone's schedule- just yours.

To filter the output add a WHERE clause to the SQL statement after the From clause. The Where clause specifies which rows should be returned. The Where clause contains a logical expression (a predicate or test) that is applied to each row in the table. If the logical expression evaluates to true for a row from the table (if that row passes the test), that row is returned in the result. The row is not returned if the predicate evaluates as False or as Unknown. . We will cover a variety of comparison operators and conditional expressions in this unit and in the following units. We will also investigate the concept of Unknown.

Our query model is now

Select col\_expressions

From table\_expression

WHERE predicate

Order By sort\_keys;

In this unit, we will look at several filters: we will discuss more filters soon.

* the Is Null and Is Not Null filters
* the In list filter
* the Between filter

A few general rules for writing filters

* You cannot test a column alias in a Where clause; you need to test using the column name.
* Character literals are enclosed in single quotes: 'CA', 'Anderson'
* In tests against character literals, leading blanks are significant. The value ' CA' does not match the value 'CA'. Trailing blanks are not significant.
* Numbers are not enclosed in quotes; do not include punctuation such as commas or dollar signs when you write numeric literals
* Dates are more complex; for now write date literals in the standard default format: '2008-04-01'. Note that this is a string literal; MySQL will cast it to a date for date testing.

For these queries the table expressions will follow the format of databaseName.tableName such as

from a\_oe.order\_headers

This means you can run these queries from within any of your databases.

1. Testing for nulls

You may want to write your queries to skip any rows where certain column values are null. The way to test for this is to add a Where clause after the From clause that filters for the nulls.

Testing for missing values requires the use of the IS NULL operator. Think of IS NULL and IS NOT NULL as single operators. You use the same operator IS NULL for any data type that you are testing.

If you try to test using the syntax = NULL, you will get no rows returned. This is not flagged as an error.

1. Test for empty attributes by using the test IS NULL. You use the same test for attributes of any data type. This tests for a null shipping\_mode which is a char(6) attribute.

Select ord\_id, ord\_date, sales\_rep\_id, shipping\_mode

From a\_oe.order\_headers

WHERE shipping\_mode IS NULL;

+--------+---------------------+--------------+---------------+

| ord\_id | ord\_date | sales\_rep\_id | shipping\_mode |

+--------+---------------------+--------------+---------------+

| 116 | 2013-11-12 00:00:00 | 155 | NULL |

| 117 | 2013-11-28 00:00:00 | 150 | NULL |

| 118 | 2013-11-28 00:00:00 | 150 | NULL |

| 119 | 2013-11-28 00:00:00 | 155 | NULL |

| 120 | 2014-01-02 00:00:00 | NULL | NULL |

| 121 | 2014-01-03 00:00:00 | NULL | NULL |

| 550 | 2014-08-02 00:00:00 | NULL | NULL |

| 551 | 2014-08-03 00:00:00 | NULL | NULL |

+--------+---------------------+--------------+---------------+

8 rows in set (0.00 sec)

1. Test for empty attributes by using the test IS NULL. This is testing an integer column.

Select ord\_id, ord\_date, sales\_rep\_id

From a\_oe.order\_headers

WHERE sales\_rep\_id IS NULL;

+--------+---------------------+--------------+

| ord\_id | ord\_date | sales\_rep\_id |

+--------+---------------------+--------------+

| 115 | 2013-11-08 00:00:00 | NULL |

| 120 | 2014-01-02 00:00:00 | NULL |

| 121 | 2014-01-03 00:00:00 | NULL |

| 225 | 2013-08-09 00:00:00 | NULL |

| 227 | 2013-08-01 00:00:00 | NULL |

| 525 | 2014-05-09 00:00:00 | NULL |

| 527 | 2014-05-01 00:00:00 | NULL |

| 550 | 2014-08-02 00:00:00 | NULL |

| 551 | 2014-08-03 00:00:00 | NULL |

+--------+---------------------+--------------+

9 rows in set (0.00 sec)

1. We can use IS NOT NULL to find rows that have a data value.

Select ord\_id, ord\_date, shipping\_mode

From a\_oe.order\_headers

WHERE shipping\_mode IS NOT NULL

Order by ord\_date

Limit 5;

+--------+---------------------+---------------+

| ord\_id | ord\_date | shipping\_mode |

+--------+---------------------+---------------+

| 227 | 2013-08-01 00:00:00 | USPS1 |

| 223 | 2013-08-05 00:00:00 | USPS1 |

| 224 | 2013-08-07 00:00:00 | FEDEX1 |

| 218 | 2013-08-08 00:00:00 | USPS1 |

| 225 | 2013-08-09 00:00:00 | FEDEX1 |

+--------+---------------------+---------------+

5 rows in set (0.00 sec)

1. Note what happens if you use the test = null instead of Is null

Select ord\_id, ord\_date, shipping\_mode

From a\_oe.order\_headers

WHERE shipping\_mode = NULL

;

Empty set

The null indicator does not equal any value in the table. So testing with = Null or <> Null is not a true test and that filter will return no rows. The use of Nulls in tables is important but null testing is not always obvious.

1. The IN list test

Suppose we want to display all customers named Morris or Morse or Morise. This is testing against a specific set of values and we can use an IN list for this. The list of values is enclosed in parentheses and the values are separated by commas.

Simple In lists

1. Using the IN list for text values. Each text value is enclosed in quotes .

Select cust\_id

, cust\_name\_last

, cust\_name\_first

From a\_oe.customers

Where cust\_name\_last in( 'Morise', 'Morris', 'Morse' )

;

+---------+----------------+-----------------+

| cust\_id | cust\_name\_last | cust\_name\_first |

+---------+----------------+-----------------+

| 401250 | Morse | Samuel |

| 402100 | Morise | William |

| 404950 | Morris | William |

| 409010 | Morris | William |

+---------+----------------+-----------------+

4 rows in set (0.00 sec)

1. Using the IN list for numeric values. It is not an error to have a value in the list which does not match any rows in the table. . It is not an error to have a in the list appear more than once.

Select ord\_id

, ord\_date

, cust\_id

From a\_oe.order\_headers

WHERE ord\_id IN (101, 107, 95, 125)

;

+--------+---------------------+---------+

| ord\_id | ord\_date | cust\_id |

+--------+---------------------+---------+

| 107 | 2013-10-02 00:00:00 | 403050 |

| 125 | 2013-12-09 00:00:00 | 409160 |

+--------+---------------------+---------+

1. You can use the NOT IN test to exclude specified data values.

Select prod\_id, prod\_name, catg\_id

From a\_prd.products

WHERE catg\_id NOT IN ('HW', 'PET');

+---------+---------------------+---------+

| prod\_id | prod\_name | catg\_id |

+---------+---------------------+---------+

| 1010 | Weights | SPG |

| 1020 | Dartboard | SPG |

. . . rows omitted

| 1130 | Mini Freezer | APL |

| 4569 | Mini Dryer | APL |

| 5000 | Fingerling Potatoes | GFD |

| 5001 | Ginger Preserve | GFD |

| 5002 | Ball-Peen Hammer | HD |

| 5004 | Dead Blow hammer | HD |

| 5005 | Shingler Hammer | HD |

| 5008 | Claw Framing | HD |

+---------+---------------------+---------+

1. You can use a list that contains only one item.

Select job\_id, job\_title, max\_salary

From a\_emp.jobs

WHERE max\_salary IN (120000 );

+--------+------------+------------+

| job\_id | job\_title | max\_salary |

+--------+------------+------------+

| 16 | Programmer | 120000.00 |

+--------+------------+------------+

1 row in set (0.01 sec)

1. You can use a NOT IN list that contains only one item.

Select job\_id, job\_title, max\_salary

From a\_emp.jobs

WHERE max\_salary NOT IN (120000 );

+--------+---------------+------------+

| job\_id | job\_title | max\_salary |

+--------+---------------+------------+

| 1 | President | 100000.00 |

| 2 | Marketing | 75000.00 |

| 4 | Sales Manager | 60000.00 |

| 8 | Sales Rep | 30000.00 |

+--------+---------------+------------+

4 rows in set (0.00 sec)

The above two queries may look like they should return all rows in one or the other of these queries. But we have 8 rows in the jobs table; one was returned with the IN (12000) test and four with the NOT IN (12000) test. What happened to the other three rows? Neither the IN test nor the NOT IN test return the rows where the max\_salary is null. For that you need to test with the IS NULL test.

1. Test with IS NULL

Select job\_id, job\_title, max\_salary

From a\_emp.jobs

WHERE max\_salary is null;

+--------+---------------+------------+

| job\_id | job\_title | max\_salary |

+--------+---------------+------------+

| 32 | Code Debugger | NULL |

| 64 | DBA | NULL |

| 128 | RD | NULL |

+--------+---------------+------------+

3 rows in set (0.00 sec)

In lists that contain nulls

If you try putting a Null in a list you will find that it does not match a row with a null. The rule is that a row is returned if the Where predicate evaluates as True; a null in the table does not match a null in the list. A null value does not match another null value. The logical value of a null matching a null is Unknown; the value of the filter expression must be True for the row to be returned.

1. Trying to use Null in a list.

Select job\_id, job\_title, max\_salary

From a\_emp.jobs

WHERE max\_salary IN (120000, null );

+--------+------------+------------+

| job\_id | job\_title | max\_salary |

+--------+------------+------------+

| 16 | Programmer | 120000.00 |

+--------+------------+------------+

1 row in set (0.01 sec)

What may seem more surprising is the following query, which returns no rows. The rule is that a row is returned if the Where predicate evaluates as True; the row is not returned if the predicate evaluates as False or as Unknown. Here we are negating the Unknown value which is still Unknown.

1. Nulls always get interesting

Select job\_id, job\_title, max\_salary

From a\_emp.jobs

WHERE max\_salary NOT IN (120000, null );

Empty set (0.00 sec)

* 1. In lists that contain row values

You can test constructed rows with an In test. In MySQL you can use the expression row(30, 101) to refer to a two part value. The word row is optional; I will use it here to emphasis that we are comparing multi-part row values.

1. Suppose we wanted to find employees in dept 30 with manager 101. We could use the following. The row values is (30, 101) and it is enclosed in parentheses for the In list. The two columns we are comparing are also enclosed in parentheses.

Select emp\_id, name\_last, dept\_id, emp\_mng

From a\_emp.employees

Where row(dept\_id, emp\_mng) IN( row(30, 101) );

+--------+-----------+---------+---------+

| emp\_id | name\_last | dept\_id | emp\_mng |

+--------+-----------+---------+---------+

| 108 | Green | 30 | 101 |

| 203 | Mays | 30 | 101 |

| 205 | Higgs | 30 | 101 |

+--------+-----------+---------+---------+

1. Now suppose we wanted to find employees in dept 30 with manager 101 and also employees in dept 35 with manager 101.

Select emp\_id

, name\_last

, dept\_id

, emp\_mng

From a\_emp.employees

Where row( dept\_id, emp\_mng ) in (

row( 30, 101 )

, row( 35, 101 )

)

Order by dept\_id, emp\_mng;

+--------+-----------+---------+---------+

| emp\_id | name\_last | dept\_id | emp\_mng |

+--------+-----------+---------+---------+

| 108 | Green | 30 | 101 |

| 203 | Mays | 30 | 101 |

| 205 | Higgs | 30 | 101 |

| 162 | Holme | 35 | 101 |

| 200 | Whale | 35 | 101 |

+--------+-----------+---------+---------+

1. You could rewrite this without the keyword row- but include the parentheses which make this a row.

Select emp\_id, name\_last, dept\_id, emp\_mng

From a\_emp.employees

Where (dept\_id, emp\_mng) IN( (30, 101), (35, 101) )

Order by dept\_id, emp\_mng;

* 1. Testing a literal against an in list

Commonly we think of testing a column against an In list of literals. But with some tests we can reverse that type of thinking. Suppose we are working with the vets database and we are told that the vet got a message that "Edgar is sick" but the message does not say if Edgar is a client or an animal. You can use a Where clause such as this to look for a match for the literal "Edgar with any of these columns.

where 'Edger' in ( cl\_name\_last, cl\_name\_first, an\_name);

1. And then run this query which looks for the literal Edger in three columns(don't worry about the join in the From clause at this time)

Select cl\_name\_last, cl\_name\_first, an\_name

From vt\_clients C

left join vt\_animals A on C.cl\_id = a.cl\_id

Where 'Edger' in ( cl\_name\_last, cl\_name\_first, an\_name);

+--------------+---------------+---------+

| cl\_name\_last | cl\_name\_first | an\_name |

+--------------+---------------+---------+

| Harris | Eddie | Edger |

| Boston | Edger | NULL |

+--------------+---------------+---------+

This could use useful for searches where you do not know if a value was entered as a person's first or last name; that happens fairly frequently when people fill out forms.

This syntax is not as obvious in meaning as testing a column against a list of values. Avoid this for a simple In list.

1. The BETWEEN test

To test data against a range of values, use BETWEEN. The Between test is an inclusive test. If the row being tested matches an end point of the range, the test has a true value, and the row will get into the output display. The range should be an increasing range. If you test WHERE salary BETWEEN 72000 and 3000 the query will run but no rows will be returned.

1. Using BETWEEN with a numeric range.

Select emp\_id, name\_last AS "Employee", salary

From a\_emp.employees

Where salary BETWEEN 65000 AND 70000;

+--------+----------+----------+

| emp\_id | Employee | salary |

+--------+----------+----------+

| 104 | Ernst | 65000.00 |

| 109 | Fiet | 65000.00 |

| 160 | Dorna | 65000.00 |

| 200 | Whale | 65000.00 |

| 103 | Hunol | 69000.00 |

+--------+----------+----------+

1. Using NOT BETWEEN to exclude values in the range.

Select emp\_id

, name\_last as "Employee"

, salary

From a\_emp.employees

Where salary not between 25000 and 105000

Order by salary ;

+--------+----------+-----------+

| emp\_id | Employee | salary |

+--------+----------+-----------+

| 201 | Harts | 15000.00 |

| 150 | Tuck | 20000.00 |

| 161 | Dewal | 120000.00 |

+--------+----------+-----------+

1. Using BETWEEN with a date range.

Select emp\_id, name\_last AS "Employee", hire\_date

From a\_emp.employees

Where hire\_date BETWEEN '2001-01-01' AND '2007-12-31';

+--------+----------+------------+

| emp\_id | Employee | hire\_date |

+--------+----------+------------+

| 150 | Tuck | 2001-10-28 |

| 155 | Hiller | 2004-03-05 |

| 201 | Harts | 2004-08-25 |

+--------+----------+------------+

1. Using BETWEEN with character range.

Select emp\_id, name\_last AS "Employee",dept\_id

From a\_emp.employees

Where name\_last BETWEEN 'J' and 'T'

Order by name\_last;

+--------+----------+---------+

| emp\_id | Employee | dept\_id |

+--------+----------+---------+

| 100 | King | 10 |

| 204 | King | 30 |

| 101 | Koch | 30 |

| 203 | Mays | 30 |

| 146 | Partne | 215 |

| 145 | Russ | 80 |

| 207 | Russ | 35 |

+--------+----------+---------+

You need to be careful with character range tests. If we had an employee with a last name composed of just the letter T, that employee would be returned. But the employee with the name Tuck is not returned.

1. Customer with a low credit rating

Select cust\_id, cust\_name\_last, cust\_name\_first, credit\_limit

From a\_oe.customers

Where credit\_limit between 0 and 1000;

+---------+----------------+-----------------+--------------+

| cust\_id | cust\_name\_last | cust\_name\_first | credit\_limit |

+---------+----------------+-----------------+--------------+

| 400801 | Washington | Geo | 750 |

| 401250 | Morse | Samuel | 750 |

| 402100 | Morise | William | 750 |

| 402110 | Coltrane | John | 750 |

| 402120 | McCoy | Tyner | 750 |

+---------+----------------+-----------------+--------------+

1. But customers with no credit rating were not returned by the previous query.

Select cust\_id, cust\_name\_last, cust\_name\_first, credit\_limit

From a\_oe.customers

Where credit\_limit is null;

+---------+----------------+-----------------+--------------+

| cust\_id | cust\_name\_last | cust\_name\_first | credit\_limit |

+---------+----------------+-----------------+--------------+

| 402500 | Jones | Elton John | NULL |

| 405000 | Day | David | NULL |

+---------+----------------+-----------------+--------------+

The word "between" is one of those words that can be ambiguous in English, If I ask you how many animals appear between the Cat and the Dog, you will probably say 3.



But if I asked you how many integers there are between 12 and 16 { 12, 13, 14, 15, 16} some people will say 3 (13,14,15) , some will say 5 [12, 13, 14, 15, 16], and a few people will say 4 [12, 13, 14, 15) or (13, 14, 15, 16]. So if someone asks you to write a query that finds values between two points, it is a good idea to ask if they want to include the end points.

The SQL between operator is always inclusive of the end points. Your job is to write a query that does the job you were asked to do.

* 1. Gotchas

Some tests to watch out for with the use of Between.

The Between operator will match no rows if the range is descending or if one of the range points is a null.

1. Range is decreasing

Select emp\_id, name\_last AS "Employee", salary

From a\_emp.employees

Where salary BETWEEN 70000 AND 65000

;

Empty set (0.00 sec)

1. One end of the range is a null

Select emp\_id, name\_last AS "Employee", salary

From a\_emp.employees

Where salary BETWEEN 30000 AND null

;

Empty set (0.00 sec)

1. One end of the range is a null

Select emp\_id, name\_last AS "Employee", salary

From a\_emp.employees

Where salary BETWEEN null AND 51000

;

Empty set (0.00 sec)

Using Between with datetime values can also be a problem. The ex\_date in the vets exam headers table is a date time value.

1. These are all the exam dates in date order

Select ex\_id, stf\_id, ex\_date

From a\_vets.vt\_exam\_headers

Order by ex\_date;

+-------+--------+---------------------+

| ex\_id | stf\_id | ex\_date |

+-------+--------+---------------------+

| 3105 | 29 | 2013-08-10 09:15:00 |

| 3010 | 29 | 2013-08-22 10:45:00 |

| 3001 | 29 | 2013-08-24 10:45:00 |

| 3203 | 29 | 2013-10-03 14:30:00 |

| 3202 | 29 | 2013-10-03 14:30:00 |

| 3304 | 15 | 2013-11-06 10:30:00 |

| 3282 | 15 | 2013-11-23 10:30:00 |

| 3306 | 29 | 2013-12-06 10:45:00 |

| 3390 | 15 | 2013-12-22 09:00:00 |

| 3393 | 29 | 2013-12-23 12:15:00 |

| 3392 | 15 | 2013-12-26 09:30:00 |

| 3409 | 29 | 2013-12-27 10:45:00 |

| 3486 | 15 | 2013-12-31 13:00:00 |

| 4101 | 15 | 2014-01-02 13:00:00 |

| 4102 | 15 | 2014-01-08 13:00:00 |

| 4103 | 38 | 2014-01-08 15:30:00 |

| 3288 | 25 | 2014-01-09 09:00:00 |

| 3104 | 38 | 2014-01-09 16:30:00 |

| 3322 | 29 | 2014-02-10 09:15:00 |

| 3321 | 29 | 2014-02-17 10:45:00 |

| 3324 | 29 | 2014-02-25 10:45:00 |

| 3323 | 29 | 2014-02-25 14:30:00 |

| 3325 | 29 | 2014-03-15 10:45:00 |

| 3420 | 15 | 2014-04-01 16:30:00 |

| 3494 | 25 | 2014-04-22 09:00:00 |

| 3411 | 29 | 2014-04-29 14:00:00 |

| 3412 | 29 | 2014-04-30 14:30:00 |

| 3413 | 15 | 2014-05-01 16:30:00 |

| 3612 | 15 | 2014-05-23 08:30:00 |

| 3513 | 15 | 2014-06-06 10:30:00 |

| 3552 | 15 | 2014-06-10 10:30:00 |

| 3514 | 29 | 2014-06-10 10:45:00 |

+-------+--------+---------------------+

1. But if I filter for exam dates in April 2014 by using between 2014-04-01 and 2014-04-30, I will not get the exam that occurred on April 30, 2014 at 14:30. If you do not include a time component, then the datetime value gets a default time component of midnight.

Select ex\_id, stf\_id, ex\_date

From a\_vets.vt\_exam\_headers

Where ex\_date between '2014-04-01' and '2014-04-30'

;

+-------+--------+---------------------+

| ex\_id | stf\_id | ex\_date |

+-------+--------+---------------------+

| 3411 | 29 | 2014-04-29 14:00:00 |

| 3420 | 15 | 2014-04-01 16:30:00 |

| 3494 | 25 | 2014-04-22 09:00:00 |

+-------+--------+---------------------+

4 rows in set (0.00 sec)

1. I could include a time components for the end of the range

Select ex\_id, stf\_id, ex\_date

From a\_vets.vt\_exam\_headers

Where ex\_date between '2014-04-01' and '2014-04-30 23:59:59';

+-------+--------+---------------------+

| ex\_id | stf\_id | ex\_date |

+-------+--------+---------------------+

| 3411 | 29 | 2014-04-29 14:00:00 |

| 3412 | 29 | 2014-04-30 14:30:00 |

| 3420 | 15 | 2014-04-01 16:30:00 |

| 3494 | 25 | 2014-04-22 09:00:00 |

+-------+--------+---------------------+

4 rows in set (0.00 sec)

You might try to solve this by using a filter

Where ex\_date between '2014-04-01' and '2013-04-01';

But that would include exam date on My 1, 2014 that have the time defaulted to midnight.

We will have a better way to write this query in Unit 05.