Table of Contents

[1. Selecting columns 1](#_Toc391133435)

[2. Selecting all columns 2](#_Toc391133436)

[3. Column aliases 3](#_Toc391133437)

[4. Sorting the output display 4](#_Toc391133438)

In this discussion we will examine a few features of the Select statement. These are:

* selecting individual columns
* selecting all columns
* using column aliases
* sorting the rows displayed

There are additional inserts in the demo file for this document.

1. Selecting columns

The first few queries use only two clauses: the FROM clause to identify the table that supplies the data and the SELECT clause to identify the columns to be returned. For these queries, all rows from the table are returned. This set of demos uses the zoo\_2014 table. Your data set might be different depending on the rows you inserted.

You indicate which columns you want displayed and the order of the columns by listing the column names in the Select clause.

1. You can display the columns in any order. Note that rows for the animals with no name displays the word NULL with this client.

Select

z\_type

, z\_name

From zoo\_2014;

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

| z\_type | z\_name |

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

| Giraffe | Sam |

| Armadillo | Abigail |

| Lion | Leon |

| Lion | Lenora |

| Giraffe | Sally |

| Zebra | Huey |

| Zebra | Dewey |

| Zebra | Louie |

| Horse | NULL |

| Giraffe | Dewey |

| armadillo | Anders |

| armadillo | Anne |

| Lion | Leon |

| Lion | NULL |

| Lion | NULL |

| Lion | |

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

16 rows in set (0.00 sec)

1. Display dates and numeric values.

Select

z\_dob

, z\_cost

, z\_name

From zoo\_2014;

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

| z\_dob | z\_cost | z\_name |

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

| 2002-05-15 10:45:00 | 5000.00 | Sam |

| 2010-01-15 08:30:00 | 490.00 | Abigail |

| 2009-02-25 15:00:00 | 5000.00 | Leon |

| 2009-03-25 15:35:00 | 5000.00 | Lenora |

| 2009-05-15 02:02:00 | 5000.25 | Sally |

| 2012-06-02 02:02:00 | 2500.25 | Huey |

| 2012-06-02 02:10:00 | 2500.25 | Dewey |

| 2013-01-02 02:25:00 | 2500.25 | Louie |

| 2010-05-15 08:30:00 | 490.00 | NULL |

| 2013-06-06 02:10:00 | 3750.00 | Dewey |

| 2010-01-15 08:30:00 | 490.00 | Anders |

| 2010-01-15 08:30:00 | 490.01 | Anne |

| 2009-02-25 15:00:00 | 1850.00 | Leon |

| 2009-02-25 15:00:00 | 1850.00 | NULL |

| 2009-02-25 15:00:00 | 1850.00 | NULL |

| 2009-02-25 15:00:00 | 1850.00 | |

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

1. Selecting all columns

The symbol \* is used to indicate that all columns should be returned. This is inefficient if you do not need to see all of the columns but is helpful for a quick look at a small table.

Using Select \* can be a bad idea with embedded SQL if the table design is changed. Embedded SQL refers to SQL statement that might be included inside other units of code. You also have to consider that someone might reorder the column positions in the table and then your query produces a different result.

1. Display all columns, all rows.

Select \*

From zoo\_2014;

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

| z\_id | z\_name | z\_type | z\_cost | z\_dob | z\_acquired |

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

| 23 | Sam | Giraffe | 5000.00 | 2002-05-15 10:45:00 | 2002-05-15 |

| 25 | Abigail | Armadillo | 490.00 | 2010-01-15 08:30:00 | 2010-04-15 |

| 56 | Leon | Lion | 5000.00 | 2009-02-25 15:00:00 | 2010-03-25 |

| 57 | Lenora | Lion | 5000.00 | 2009-03-25 15:35:00 | 2011-01-15 |

| 85 | Sally | Giraffe | 5000.25 | 2009-05-15 02:02:00 | 2012-05-15 |

| 43 | Huey | Zebra | 2500.25 | 2012-06-02 02:02:00 | 2012-06-02 |

| 44 | Dewey | Zebra | 2500.25 | 2012-06-02 02:10:00 | 2012-06-02 |

| 45 | Louie | Zebra | 2500.25 | 2013-01-02 02:25:00 | 2013-01-02 |

| 47 | NULL | Horse | 490.00 | 2010-05-15 08:30:00 | 2010-04-15 |

| 52 | Dewey | Giraffe | 3750.00 | 2013-06-06 02:10:00 | 2013-07-12 |

| 70 | Anders | armadillo | 490.00 | 2010-01-15 08:30:00 | 2010-04-15 |

| 71 | Anne | armadillo | 490.01 | 2010-01-15 08:30:00 | 2010-04-15 |

| 72 | Leon | Lion | 1850.00 | 2009-02-25 15:00:00 | 2010-03-25 |

| 73 | NULL | Lion | 1850.00 | 2009-02-25 15:00:00 | 2010-03-25 |

| 74 | NULL | Lion | 1850.00 | 2009-02-25 15:00:00 | 2010-03-25 |

| 75 | | Lion | 1850.00 | 2009-02-25 15:00:00 | 2010-03-25 |

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

1. Column aliases

By default, the column headers are the attribute names. Column aliases can be used to supply different headers for the output display.

Notice in the demos below how case issues are handled in the various ways of creating column aliases.

1. Display column headers other than the attribute names. The word AS is optional and may be omitted.

Select

z\_id

, z\_dob AS BirthDate

, z\_cost AS Price

, z\_name AS NAME

From zoo\_2014;

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

| z\_id | birthdate | price | name |

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

| 23 | 2002-05-15 10:45:00 | 5000.00 | Sam |

| 25 | 2010-01-15 08:30:00 | 490.00 | Abigail |

| 56 | 2009-02-25 15:00:00 | 5000.00 | Leon |

| 57 | 2009-03-25 15:35:00 | 5000.00 | Lenora |

| 85 | 2009-05-15 02:02:00 | 5000.25 | Sally |

| 43 | 2012-06-02 02:02:00 | 2500.25 | Huey |

| 44 | 2012-06-02 02:10:00 | 2500.25 | Dewey |

| 45 | 2013-01-02 02:25:00 | 2500.25 | Louie |

| 47 | 2010-05-15 08:30:00 | 490.00 | NULL |

| 52 | 2013-06-06 02:10:00 | 3750.00 | Dewey |

| 70 | 2010-01-15 08:30:00 | 490.00 | Anders |

| 71 | 2010-01-15 08:30:00 | 490.01 | Anne |

| 72 | 2009-02-25 15:00:00 | 1850.00 | Leon |

| 73 | 2009-02-25 15:00:00 | 1850.00 | NULL |

| 74 | 2009-02-25 15:00:00 | 1850.00 | NULL |

| 75 | 2009-02-25 15:00:00 | 1850.00 | |

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

1. The use of double quotes for your aliases allows you to use spaces or special characters in the header.

Select

z\_id

, z\_dob AS "Date of Birth"

, z\_cost AS "Price $"

, z\_name As "Name"

From zoo\_2014;

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

| z\_id | Date of Birth | Price $ | Name |

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

| 23 | 2002-05-15 10:45:00 | 5000.00 | Sam |

| 25 | 2010-01-15 08:30:00 | 490.00 | Abigail |

| 56 | 2009-02-25 15:00:00 | 5000.00 | Leon |

| 57 | 2009-03-25 15:35:00 | 5000.00 | Lenora |

| 85 | 2009-05-15 02:02:00 | 5000.25 | Sally |

| 43 | 2012-06-02 02:02:00 | 2500.25 | Huey |

| 44 | 2012-06-02 02:10:00 | 2500.25 | Dewey |

| 45 | 2013-01-02 02:25:00 | 2500.25 | Louie |

| 47 | 2010-05-15 08:30:00 | 490.00 | NULL |

| 52 | 2013-06-06 02:10:00 | 3750.00 | Dewey |

| 70 | 2010-01-15 08:30:00 | 490.00 | Anders |

| 71 | 2010-01-15 08:30:00 | 490.01 | Anne |

| 72 | 2009-02-25 15:00:00 | 1850.00 | Leon |

| 73 | 2009-02-25 15:00:00 | 1850.00 | NULL |

| 74 | 2009-02-25 15:00:00 | 1850.00 | NULL |

| 75 | 2009-02-25 15:00:00 | 1850.00 | |

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

1. Sorting the output display

If you want to control the order in which the rows are displayed, you use an ORDER BY clause.

You can order by

* a column
* a column alias
* the numeric position of the column in the Select ( not always a good idea)
* a calculated column expression ( we will discuss this in the next unit)

If you have two columns with the same alias and try to sort by the alias, you will get an error message.

1. Controlling the order in which the rows are displayed. This is sorted by price with the lower values first; this is an ascending sort which is the default sort order.

Select

z\_id

, z\_dob AS "BirthDate"

, z\_cost AS "Price"

, z\_name As "Name"

From zoo\_2014

ORDER BY z\_cost;

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

| z\_id | BirthDate | Price | Name |

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

| 47 | 2010-05-15 08:30:00 | 490.00 | NULL |

| 25 | 2010-01-15 08:30:00 | 490.00 | Abigail |

| 70 | 2010-01-15 08:30:00 | 490.00 | Anders |

| 71 | 2010-01-15 08:30:00 | 490.01 | Anne |

| 74 | 2009-02-25 15:00:00 | 1850.00 | NULL |

| 73 | 2009-02-25 15:00:00 | 1850.00 | NULL |

| 72 | 2009-02-25 15:00:00 | 1850.00 | Leon |

| 75 | 2009-02-25 15:00:00 | 1850.00 | |

| 45 | 2013-01-02 02:25:00 | 2500.25 | Louie |

| 44 | 2012-06-02 02:10:00 | 2500.25 | Dewey |

| 43 | 2012-06-02 02:02:00 | 2500.25 | Huey |

| 52 | 2013-06-06 02:10:00 | 3750.00 | Dewey |

| 57 | 2009-03-25 15:35:00 | 5000.00 | Lenora |

| 56 | 2009-02-25 15:00:00 | 5000.00 | Leon |

| 23 | 2002-05-15 10:45:00 | 5000.00 | Sam |

| 85 | 2009-05-15 02:02:00 | 5000.25 | Sally |

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

1. Using DESC to specify a descending sort.

Select

z\_id

, z\_dob AS "BirthDate"

, z\_cost AS "Price"

, z\_name As "Name"

From zoo\_2014

ORDER BY z\_cost DESC;

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

| z\_id | BirthDate | Price | Name |

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

| 85 | 2009-05-15 02:02:00 | 5000.25 | Sally |

| 23 | 2002-05-15 10:45:00 | 5000.00 | Sam |

| 56 | 2009-02-25 15:00:00 | 5000.00 | Leon |

| 57 | 2009-03-25 15:35:00 | 5000.00 | Lenora |

| 52 | 2013-06-06 02:10:00 | 3750.00 | Dewey |

| 45 | 2013-01-02 02:25:00 | 2500.25 | Louie |

| 44 | 2012-06-02 02:10:00 | 2500.25 | Dewey |

| 43 | 2012-06-02 02:02:00 | 2500.25 | Huey |

| 73 | 2009-02-25 15:00:00 | 1850.00 | NULL |

| 74 | 2009-02-25 15:00:00 | 1850.00 | NULL |

| 72 | 2009-02-25 15:00:00 | 1850.00 | Leon |

| 75 | 2009-02-25 15:00:00 | 1850.00 | |

| 71 | 2010-01-15 08:30:00 | 490.01 | Anne |

| 70 | 2010-01-15 08:30:00 | 490.00 | Anders |

| 25 | 2010-01-15 08:30:00 | 490.00 | Abigail |

| 47 | 2010-05-15 08:30:00 | 490.00 | NULL |

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

If two rows have the same value for z\_cost, then we have not specified an exact order for those rows

1. This is a two level sort. The first sort key is the z\_type. If the z\_type values of two rows match, then the z\_cost value is used for the second sort level.

Select

z\_type As "Type"

, z\_cost AS "Price"

, z\_name As "Name"

From zoo\_2014

ORDER BY z\_type, z\_cost;

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

| Type | Price | Name |

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

| Armadillo | 490.00 | Abigail |

| armadillo | 490.00 | Anders |

| armadillo | 490.01 | Anne |

| Giraffe | 3750.00 | Dewey |

| Giraffe | 5000.00 | Sam |

| Giraffe | 5000.25 | Sally |

| Horse | 490.00 | NULL |

| Lion | 1850.00 | Leon |

| Lion | 1850.00 | NULL |

| Lion | 1850.00 | NULL |

| Lion | 1850.00 | |

| Lion | 5000.00 | Lenora |

| Lion | 5000.00 | Leon |

| Zebra | 2500.25 | Louie |

| Zebra | 2500.25 | Dewey |

| Zebra | 2500.25 | Huey |

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

1. This is a two level sort. The first sort key is the z\_type and it is ascending. The second sort key z\_cost uses a descending sort.

Select

z\_type As "Type"

, z\_cost AS "Price"

, z\_name As "Name"

From zoo\_2014

ORDER BY z\_type, z\_cost desc

;

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

| Type | Price | Name |

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

| armadillo | 490.01 | Anne |

| Armadillo | 490.00 | Abigail |

| armadillo | 490.00 | Anders |

| Giraffe | 5000.25 | Sally |

| Giraffe | 5000.00 | Sam |

| Giraffe | 3750.00 | Dewey |

| Horse | 490.00 | NULL |

| Lion | 5000.00 | Lenora |

| Lion | 5000.00 | Leon |

| Lion | 1850.00 | Leon |

| Lion | 1850.00 | NULL |

| Lion | 1850.00 | NULL |

| Lion | 1850.00 | |

| Zebra | 2500.25 | Louie |

| Zebra | 2500.25 | Dewey |

| Zebra | 2500.25 | Huey |

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

1. The default is that nulls sort as a low-valued data item. We have animals with no name value. They are sorting at the top of this display.

Select

z\_type As "Type"

, z\_name As "Name"

From zoo\_2014

ORDER BY z\_name;

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

| Type | Name |

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

| Horse | NULL |

| Lion | NULL |

| Lion | NULL |

| Lion | |

| Armadillo | Abigail |

| armadillo | Anders |

| armadillo | Anne |

| Giraffe | Dewey |

| Zebra | Dewey |

| Zebra | Huey |

| Lion | Lenora |

| Lion | Leon |

| Lion | Leon |

| Zebra | Louie |

| Giraffe | Sally |

| Giraffe | Sam |

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

1. With a Desc z\_name sort the nulls are at the end of the result set.

Select

z\_type As "Type"

, z\_name As "Name"

From zoo\_2014

ORDER BY z\_name DESC;

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

| Type | Name |

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

| Giraffe | Sam |

| Giraffe | Sally |

| Zebra | Louie |

| Lion | Leon |

| Lion | Leon |

| Lion | Lenora |

| Zebra | Huey |

| Giraffe | Dewey |

| Zebra | Dewey |

| armadillo | Anne |

| armadillo | Anders |

| Armadillo | Abigail |

| Lion | |

| Lion | NULL |

| Lion | NULL |

| Horse | NULL |

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

1. You can sort on a date value. If two rows have the same value for the z\_dob column, we are not specifying which of the rows is first in the display. MySQL can return rows that are tied for z\_dob in nay order..

Select

z\_id

, z\_dob as "BirthDate"

, z\_name as "Name"

From zoo\_2014

ORDER BY z\_dob DESC;

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

| z\_id | BirthDate | Name |

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

| 52 | 2013-06-06 02:10:00 | Dewey |

| 45 | 2013-01-02 02:25:00 | Louie |

| 44 | 2012-06-02 02:10:00 | Dewey |

| 43 | 2012-06-02 02:02:00 | Huey |

| 47 | 2010-05-15 08:30:00 | NULL |

| 25 | 2010-01-15 08:30:00 | Abigail |

| 70 | 2010-01-15 08:30:00 | Anders |

| 71 | 2010-01-15 08:30:00 | Anne |

| 85 | 2009-05-15 02:02:00 | Sally |

| 57 | 2009-03-25 15:35:00 | Lenora |

| 74 | 2009-02-25 15:00:00 | NULL |

| 73 | 2009-02-25 15:00:00 | NULL |

| 72 | 2009-02-25 15:00:00 | Leon |

| 75 | 2009-02-25 15:00:00 | |

| 56 | 2009-02-25 15:00:00 | Leon |

| 23 | 2002-05-15 10:45:00 | Sam |

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

1. You can sort by a column alias. Since this alias includes spaces, it needs to be quoted and you need to use the back tick.

Select

z\_id

, z\_dob as "Date of Birth"

, z\_name as "Name"

From zoo\_2014

ORDER BY **`**Date of Birth**`**;

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

| z\_id | Date of Birth | Name |

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

| 23 | 2002-05-15 10:45:00 | Sam |

| 74 | 2009-02-25 15:00:00 | NULL |

| 73 | 2009-02-25 15:00:00 | NULL |

| 72 | 2009-02-25 15:00:00 | Leon |

| 75 | 2009-02-25 15:00:00 | |

| 56 | 2009-02-25 15:00:00 | Leon |

| 57 | 2009-03-25 15:35:00 | Lenora |

| 85 | 2009-05-15 02:02:00 | Sally |

| 70 | 2010-01-15 08:30:00 | Anders |

| 25 | 2010-01-15 08:30:00 | Abigail |

| 71 | 2010-01-15 08:30:00 | Anne |

| 47 | 2010-05-15 08:30:00 | NULL |

| 43 | 2012-06-02 02:02:00 | Huey |

| 44 | 2012-06-02 02:10:00 | Dewey |

| 45 | 2013-01-02 02:25:00 | Louie |

| 52 | 2013-06-06 02:10:00 | Dewey |

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

1. What happens if you use double quotes on the sort key identifier? Are these rows sorted in date order? To get sorting on a quoted alias you need to use the back ticks.

Select

z\_id

, z\_dob as "Date of Birth"

, z\_name as "Name"

From zoo\_2014

ORDER BY "Date of Birth";

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

| z\_id | Date of Birth | Name |

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

| 23 | 2002-05-15 10:45:00 | Sam |

| 25 | 2010-01-15 08:30:00 | Abigail |

| 56 | 2009-02-25 15:00:00 | Leon |

| 57 | 2009-03-25 15:35:00 | Lenora |

| 85 | 2009-05-15 02:02:00 | Sally |

| 43 | 2012-06-02 02:02:00 | Huey |

| 44 | 2012-06-02 02:10:00 | Dewey |

| 45 | 2013-01-02 02:25:00 | Louie |

| 47 | 2010-05-15 08:30:00 | NULL |

| 52 | 2013-06-06 02:10:00 | Dewey |

| 70 | 2010-01-15 08:30:00 | Anders |

| 71 | 2010-01-15 08:30:00 | Anne |

| 72 | 2009-02-25 15:00:00 | Leon |

| 73 | 2009-02-25 15:00:00 | NULL |

| 74 | 2009-02-25 15:00:00 | NULL |

| 75 | 2009-02-25 15:00:00 | |

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

1. MySQL allows you to sort by the column number. This is generally considered poor style since it is easy to rearrange the column in the select and forget to adjust the Order By clause. You want to write SQL that is easier to write correctly and harder to write incorrectly.  
   This will sort by the z\_type values then by the z\_name values.

Select

z\_id

, z\_type

, z\_name

From zoo\_2014

ORDER BY 2,3;

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

| z\_id | z\_type | z\_name |

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

| 25 | Armadillo | Abigail |

| 70 | armadillo | Anders |

| 71 | armadillo | Anne |

| 52 | Giraffe | Dewey |

| 85 | Giraffe | Sally |

| 23 | Giraffe | Sam |

| 47 | Horse | NULL |

| 73 | Lion | NULL |

| 74 | Lion | NULL |

| 75 | Lion | |

| 57 | Lion | Lenora |

| 56 | Lion | Leon |

| 72 | Lion | Leon |

| 44 | Zebra | Dewey |

| 43 | Zebra | Huey |

| 45 | Zebra | Louie |

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

You can sort on calculated columns, either by using the alias or repeating the calculation as the sort key. We discuss calculation later; this is included here for completeness. Extract (month..) gives us the numerical value of the month.

Select z\_id

, extract( Month from z\_dob) AS "Birth Month"

, z\_name As "Name"

From zoo\_2014

ORDER BY extract( Month from z\_dob);

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

| z\_id | Birth Month | Name |

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

| 70 | 1 | Anders |

| 25 | 1 | Abigail |

| 71 | 1 | Anne |

| 45 | 1 | Louie |

| 72 | 2 | Leon |

| 73 | 2 | NULL |

| 74 | 2 | NULL |

| 56 | 2 | Leon |

| 75 | 2 | |

| 57 | 3 | Lenora |

| 47 | 5 | NULL |

| 85 | 5 | Sally |

| 23 | 5 | Sam |

| 52 | 6 | Dewey |

| 44 | 6 | Dewey |

| 43 | 6 | Huey |

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