Table of Contents

[1. Syntax for outer joins (Left, Right, Full) 1](#_Toc330936900)

[2. Queries using altgeld\_mart tables 3](#_Toc330936901)

[2.1. Customers and orders 3](#_Toc330936902)

[2.2. Products and orders 5](#_Toc330936903)

These are the tables we are using. Note that we have employees with no projects and a department with no employees and employees with no department.

z\_em\_dept z\_em\_emp z\_em\_empproj

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **d\_id** | **d\_name** |  | **e\_id** | **e\_name** | **d\_id** |  | **p\_id** | **e\_id** |
| 100 | Manufacturing |  | 1 | Jones | 150 |  | ORDB-10 | 3 |
| 150 | Accounting |  | 2 | Martin | 150 |  | ORDB-10 | 5 |
| 200 | Marketing |  | 3 | Gates | 250 |  | Q4-SALES | 2 |
| 250 | Research |  | 4 | Anders | 100 |  | Q4-SALES | 4 |
|  |  |  | 5 | Bossy |  |  | ORDB-10 | 2 |
|  |  |  | 6 | Perkins |  |  | Q4-SALES | 5 |

1. Syntax for outer joins (Left, Right, Full)

Outer joins can use the syntax Left Join or Right Join. A join written as

From tblA LEFT JOIN tblB

will include all rows from table tblA and any matching rows from tblB. The table to the left of the phrase Left Join will have all of its rows returned.

A join written as

From tblA RIGHT JOIN tblB

will include all rows from table tblB and any matching rows from tblA. The table to the right of the phrase Right Join will have all of its rows returned.

The outer joins are not symmetric.

The word OUTER is optional; you can use Left Outer Join or LeftJoin.

You will still need to identify the joining columns.

1. All departments; employees of those departments if they exist.

Select z\_em\_dept.d\_id, d\_name, e\_id, e\_name

From z\_em\_dept

LEFT JOIN z\_em\_emp on z\_em\_dept.d\_id = z\_em\_emp.d\_id

;

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

| d\_id | d\_name | e\_id | e\_name |

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

| 100 | Manufacturing | 4 | Anders |

| 150 | Accounting | 1 | Jones |

| 150 | Accounting | 2 | Martin |

| 200 | Marketing | NULL | NULL |

| 250 | Research | 3 | Gates |

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

1. All employees; assigned departments if they exist. Outer joins are not commutative

Select d\_id, d\_name, e\_id, e\_name

From z\_em\_emp

LEFT JOIN z\_em\_dept using(d\_id);

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

| d\_id | d\_name | e\_id | e\_name |

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

| 150 | Accounting | 1 | Jones |

| 150 | Accounting | 2 | Martin |

| 250 | Research | 3 | Gates |

| 100 | Manufacturing | 4 | Anders |

| NULL | NULL | 5 | Bossy |

| NULL | NULL | 6 | Perkins |

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

1. All employees; assigned departments if they exist.

Select d\_id, d\_name, e\_id, e\_name

From z\_em\_dept

RIGHT JOIN z\_em\_emp using(d\_id)

;

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

| d\_id | d\_name | e\_id | e\_name |

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

| 150 | Accounting | 1 | Jones |

| 150 | Accounting | 2 | Martin |

| 250 | Research | 3 | Gates |

| 100 | Manufacturing | 4 | Anders |

| NULL | NULL | 5 | Bossy |

| NULL | NULL | 6 | Perkins |

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

MySQL does not yet support the Full Outer join syntax that you might know from another dbms.

1. Three table outer join. This is all of the departments and their employees if there are any in the department and the projects if the employees have a project.

Select d\_id, d\_name, e\_id, e\_name, p\_id

From z\_em\_dept

LEFT JOIN z\_em\_emp using(d\_id)

LEFT JOIN z\_em\_empproj using(e\_id);

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

| d\_id | d\_name | e\_id | e\_name | p\_id |

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

| 100 | Manufacturing | 4 | Anders | Q4-SALES |

| 150 | Accounting | 1 | Jones | NULL |

| 150 | Accounting | 2 | Martin | Q4-SALES |

| 150 | Accounting | 2 | Martin | ORDB-10 |

| 200 | Marketing | NULL | NULL | NULL |

| 250 | Research | 3 | Gates | ORDB-10 |

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

1. Three table outer join. This is all of the employees and their departments if they have one and their projects if they have one

Select e\_id, e\_name,d\_id, d\_name, p\_id

From z\_em\_emp

LEFT JOIN z\_em\_dept using(d\_id)

LEFT JOIN z\_em\_empproj using(e\_id)

Order by z\_em\_emp.e\_id;

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

| e\_id | e\_name | d\_id | d\_name | p\_id |

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

| 1 | Jones | 150 | Accounting | NULL |

| 2 | Martin | 150 | Accounting | Q4-SALES |

| 2 | Martin | 150 | Accounting | ORDB-10 |

| 3 | Gates | 250 | Research | ORDB-10 |

| 4 | Anders | 100 | Manufacturing | Q4-SALES |

| 5 | Bossy | NULL | NULL | ORDB-10 |

| 5 | Bossy | NULL | NULL | Q4-SALES |

| 6 | Perkins | NULL | NULL | NULL |

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

1. Suppose we want to see all employees and their departments if they have one and the names of their projects if they have one. The following query does not do that. We start with an outer join but then use an inner join which eliminates employees with no projects.

Select z\_em\_emp.e\_id, e\_name,z\_em\_dept.d\_id, d\_name, p\_id

From z\_em\_emp

LEFT JOIN z\_em\_dept on z\_em\_dept.d\_id = z\_em\_emp.d\_id

Join z\_em\_empproj on z\_em\_emp.e\_id = z\_em\_empproj.e\_id

Order by z\_em\_emp.e\_id;

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

| e\_id | e\_name | d\_id | d\_name | p\_id |

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

| 2 | Martin | 150 | Accounting | ORDB-10 |

| 2 | Martin | 150 | Accounting | Q4-SALES |

| 3 | Gates | 250 | Research | ORDB-10 |

| 4 | Anders | 100 | Manufacturing | Q4-SALES |

| 5 | Bossy | NULL | NULL | ORDB-10 |

| 5 | Bossy | NULL | NULL | Q4-SALES |

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

1. Queries using altgeld\_mart tables

Customers and orders

The cust\_id filter is simply to reduce the volume of output.

1. Customers with orders. This uses an inner join.

Select cust\_id

, cust\_name\_last

, ord\_id

From a\_oe.customers

Join a\_oe.order\_headers using(cust\_id)

Where cust\_id between 404900 and 409030

Order by cust\_id, ord\_id;

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

| cust\_id | cust\_name\_last | ord\_id |

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

| 404900 | Williams | 520 |

| 404950 | Morris | 110 |

| 404950 | Morris | 408 |

| 404950 | Morris | 510 |

| 404950 | Morris | 535 |

| 404950 | Morris | 540 |

| 405000 | Day | 116 |

| 408770 | Clay | 405 |

| 409030 | Mazur | 128 |

| 409030 | Mazur | 130 |

| 409030 | Mazur | 324 |

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

11 rows in set (0.03 sec)

1. Customers with and without orders. This uses an outer join; Customers Left Join Order Headers.  
   That means we get customers with orders and if the customer has several orders, that customer gets multiple lines in the result set.  
   We also get rows for the two customers in this cust\_id range who have no orders and the column for their order id value is null- these customers each get one row.

Select cust\_id

, cust\_name\_last

, ord\_id

From a\_oe.customers

left join a\_oe.order\_headers using(cust\_id)

Where cust\_id between 404900 and 409030

Order by cust\_id, ord\_id;

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

| cust\_id | cust\_name\_last | ord\_id |

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

| 404900 | Williams | 520 |

| 404950 | Morris | 110 |

| 404950 | Morris | 408 |

| 404950 | Morris | 510 |

| 404950 | Morris | 535 |

| 404950 | Morris | 540 |

| 405000 | Day | 116 |

| 408770 | Clay | 405 |

| 409010 | Morris | NULL |

| 409020 | Max | NULL |

| 409030 | Mazur | 128 |

| 409030 | Mazur | 130 |

| 409030 | Mazur | 324 |

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

13 rows in set (0.03 sec)

1. Now consider this join. I change the join to a right join. The result set is the same as the inner join used previously. Why?

Select cust\_id

, cust\_name\_last

, ord\_id

From a\_oe.customers

Right join a\_oe.order\_headers using(cust\_id)

Where cust\_id between 404900 and 409030

Order by cust\_id, ord\_id;

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

| cust\_id | cust\_name\_last | ord\_id |

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

| 404900 | Williams | 520 |

| 404950 | Morris | 110 |

| 404950 | Morris | 408 |

| 404950 | Morris | 510 |

| 404950 | Morris | 535 |

| 404950 | Morris | 540 |

| 405000 | Day | 116 |

| 408770 | Clay | 405 |

| 409030 | Mazur | 128 |

| 409030 | Mazur | 130 |

| 409030 | Mazur | 324 |

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

11 rows in set (0.03 sec)

In our database we have a foreign key in the order headers table that refers back to the customer table and to the cust\_id in the customer table.

create table a\_oe.order\_headers(

ord\_id int unsigned not null

, ord\_date datetime not null

, cust\_id int unsigned not null

. . .

, constraint ord\_cust\_fk foreign key(cust\_id) references a\_oe.customers(cust\_id)

. . . )

I also set the cust\_id in the order headers table as Not null. This means that every row in the order headers table must have a value for the cust\_id (it is Not null) and that cust\_id in the order header must match a cust\_id in the customers tables (foreign key reference) .

The outer join in this query is asking for all orders whether or not they match a customer. But our database is set up so that every order header rows is matched with a customer. So it does not make sense to ask to see order headers rows that do not match a customer. In this case you should use an inner join. Using an outer join when it is logically impossible to return unmatched rows is inefficient. Someone reading your query would assume you have made a mistake someplace but they would not know what the mistake is- is the database badly designed and allows the entry of orders that do not belong to a customer (who pays for those orders?), or did you get the join order incorrect?

Products and orders

These are limited to products in the MUS category to reduce the volume of output

1. First an inner join- these show products which have been ordered- each product id must match a product id on an order detail row

Select PR.prod\_id, PR.prod\_desc, PR.catg\_id, OD.ord\_id

From a\_prd.products PR

Join a\_oe.order\_details OD on PR.prod\_id = OD.prod\_id

Where PR.catg\_id in ('MUS')

Order by PR.prod\_id;

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

| prod\_id | prod\_desc | catg\_id | ord\_id |

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

| 2014 | Bix Beiderbecke - Tiger Rag | MUS | 518 |

| 2014 | Bix Beiderbecke - Tiger Rag | MUS | 525 |

| 2412 | David Newman - Davey Blue | MUS | 525 |

| 2746 | Charles Mingus - Blues & Politics | MUS | 525 |

| 2747 | Charles Mingus - Blues & Roots | MUS | 520 |

| 2947 | Ornette Coleman - Sound Grammer | MUS | 525 |

| 2984 | John Coltrane - Lush Life | MUS | 518 |

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

7 rows in set (0.00 sec)

1. How many products do we have in the MUS category?  
   We have 11 products; looking at the previous result set, 6 of these products were sold ( One was on two different orders)

Select PR.prod\_id, PR.prod\_desc, PR.catg\_id

From a\_prd.products PR

Where catg\_id in ('MUS')

Order by PR.prod\_id;

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

| prod\_id | prod\_desc | catg\_id |

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

| 2014 | Bix Beiderbecke - Tiger Rag | MUS |

| 2234 | Charles Mingus - Pithecanthropus Erectus | MUS |

| 2337 | John Coltrane - Blue Train | MUS |

| 2412 | David Newman - Davey Blue | MUS |

| 2487 | Stanley Turrentine - Don't Mess With Mr. T | MUS |

| 2746 | Charles Mingus - Blues & Politics | MUS |

| 2747 | Charles Mingus - Blues & Roots | MUS |

| 2933 | David Newman - I Remember Brother Ray | MUS |

| 2947 | Ornette Coleman - Sound Grammer | MUS |

| 2984 | John Coltrane - Lush Life | MUS |

| 2987 | Stanley Turrentine - Ballads | MUS |

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

11 rows in set (0.00 sec)

1. We can use an outer join to get both ordered and un-ordered products

Select PR.prod\_id, prod\_desc, catg\_id, ord\_id

From a\_prd.products PR

Left join a\_oe.order\_details OD on PR.prod\_id = OD.prod\_id

Where catg\_id in ('MUS')

Order by PR.prod\_id;

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

| prod\_id | prod\_desc | catg\_id | ord\_id |

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

| 2014 | Bix Beiderbecke - Tiger Rag | MUS | 518 |

| 2014 | Bix Beiderbecke - Tiger Rag | MUS | 525 |

| 2234 | Charles Mingus - Pithecanthropus Erectus | MUS | NULL |

| 2337 | John Coltrane - Blue Train | MUS | NULL |

| 2412 | David Newman - Davey Blue | MUS | 525 |

| 2487 | Stanley Turrentine - Don't Mess With Mr. T | MUS | NULL |

| 2746 | Charles Mingus - Blues & Politics | MUS | 525 |

| 2747 | Charles Mingus - Blues & Roots | MUS | 520 |

| 2933 | David Newman - I Remember Brother Ray | MUS | NULL |

| 2947 | Ornette Coleman - Sound Grammer | MUS | 525 |

| 2984 | John Coltrane - Lush Life | MUS | 518 |

| 2987 | Stanley Turrentine - Ballads | MUS | NULL |

1. This query gives us rows for the same products- why are we missing values in the first column which shows the product id? Every product has a product Id!

Select OD.prod\_id, prod\_desc, catg\_id, ord\_id

From a\_prd.products PR

Left join a\_oe.order\_details OD on PR.prod\_id = OD.prod\_id

Where catg\_id in ('MUS')

Order by OD.prod\_id;

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

| prod\_id | prod\_desc | catg\_id | ord\_id |

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

| NULL | Stanley Turrentine - Ballads | MUS | NULL |

| NULL | David Newman - I Remember Brother Ray | MUS | NULL |

| NULL | Stanley Turrentine - Don't Mess With Mr. T | MUS | NULL |

| NULL | Charles Mingus - Pithecanthropus Erectus | MUS | NULL |

| NULL | John Coltrane - Blue Train | MUS | NULL |

| 2014 | Bix Beiderbecke - Tiger Rag | MUS | 518 |

| 2014 | Bix Beiderbecke - Tiger Rag | MUS | 525 |

| 2412 | David Newman - Davey Blue | MUS | 525 |

| 2746 | Charles Mingus - Blues & Politics | MUS | 525 |

| 2747 | Charles Mingus - Blues & Roots | MUS | 520 |

| 2947 | Ornette Coleman - Sound Grammer | MUS | 525 |

| 2984 | John Coltrane - Lush Life | MUS | 518 |

What I did is switch the column alias for the first column and for the sort key to use the order detail s table. If I am looking for the product id in the order details table, the products which are not ordered do not have a value for that column and display as nulls.