Explicit vs implicit SQL joins

Asked 16 years, 3 months ago Modified 1 year, 10 months ago Viewed 240k times

502

Is there any efficiency difference in an explicit vs implicit inner join? For example:

```
SELECT * FROM
table a INNER JOIN table b
ON a.id = b.id;
```

vs.

```
SELECT a.*, b.*
FROM table a, table b
WHERE a.id = b.id;
```

sql join

Share

Improve this question

Follow

edited Oct 26, 2017 at 19:14

user8839064

17 • 3

asked Sep 4, 2008 at 22:49

dmanxiii

52.1k • 10 • 34 • 23

18 Good question. I'm curious why the explicit join is used at all. Is it not possible to do all queries without it? – andrew Jan 23, 2011 at 22:45

7 use EXPLAIN keyword to know the difference about both the queries.. use JOIN and see the difference.. If you try in a table more than 100k records you can see the difference... – Jey Mar 16, 2012 at 8:38

@andrew My question was actually whether implicit join was a form of "hack" (as in "A query involving more than one table, not using a join? That's a hack isn't it?") – bobobobo Apr 13, 2013 at 0:53

4 They are different, implicit joining will surprise you every once in a while when dealing with null values; use explicit joining and avoid bugs that arise when "nothing changed!" – BlackTigerX Sep 3, 2013 at 23:46

4 There is no difference. `,` is `CROSS JOIN` with looser binding & `INNER JOIN` is `CROSS JOIN` with `ON` like `WHERE` but tighter binding. What matters to execution is how the DBMS optimizes queries. – philipxy Aug 22, 2017 at 21:52

12 Answers

Sorted by: Highest score (default)

179

Performance-wise, they are exactly the same (at least in SQL Server).

PS: Be aware that the "implicit `OUTER JOIN`" syntax--using `*=` or `=*` in a `WHERE` after using comma--is deprecated since SQL Server 2005. (The "implicit (`CROSS`) `JOIN`" syntax using comma as used in the question is still supported.)

[Deprecation of "Old Style" JOIN Syntax: Only A Partial Thing]

Share Improve this answer Follow

edited Feb 21, 2023 at 21:18
philipxy
15.1k ● 6 ● 42 ● 94

answered Sep 4, 2008 at 22:56
lomaxx
116k ● 58 ● 147 ● 180

4 @lomaxx, just for clarity's sake, could you specify *which* syntax of the 2 in the question is deprecated? – J Wynia Sep 5, 2008 at 0:01

8 Can you provide supporting documentation? This sounds wrong on multiple levels. – ChrisLively May 20, 2009 at 14:28

27 How do you deprecate the SQL standard? – David Crawshaw Sep 30, 2009 at 9:10

10 @david Crenshaw, the implicit join is no longer in the standard and hasn't been for 18 years. – HLGEM Jun 21, 2010 at 20:00

13 So-called "implicit joins" of the 'inner' or 'cross' variety remain in the Standard. SQL Server is deprecating the "old-style" outer join syntax (i.e. `*=` and `=*`) which has never been Standard. – onedaywhen Sep 28, 2011 at 17:08

161

Personally I prefer the join syntax as its makes it clearer that the tables are joined and how they are joined. Try compare larger SQL queries where you selecting from 8 different tables and you have lots of filtering in the where. By using join syntax you separate out the parts where the tables are joined, to the part where you are filtering the rows.

Share Improve this answer Follow

answered Sep 4, 2008 at 23:23
grom
16.1k ● 19 ● 65 ● 67

18 I completely agree, but this is kind of off-topic. OP asked about efficiency. – villasv Nov 29, 2017 at 13:01

75

On MySQL 5.1.51, both queries have identical execution plans:

```
mysql> explain select * from table1 a inner join table2 b on a.pid = b.pid;
+----+-------------+-------+------+---------------+------+---------+-----------
---+------+-------+
```

```
| id | select_type | table | type | possible_keys | key  | key_len | ref
| rows | Extra |
+----+-------------+-------+------+---------------+------+---------+----------
---+------+-------+
|  1 | SIMPLE      | b     | ALL  | PRIMARY       | NULL | NULL    | NULL
|  986 |       |
|  1 | SIMPLE      | a     | ref  | pid           | pid  | 4       |
schema.b.pid |   70 |       |
+----+-------------+-------+------+---------------+------+---------+----------
---+------+-------+
2 rows in set (0.02 sec)

mysql> explain select * from table1 a, table2 b where a.pid = b.pid;
+----+-------------+-------+------+---------------+------+---------+----------
---+------+-------+
| id | select_type | table | type | possible_keys | key  | key_len | ref
| rows | Extra |
+----+-------------+-------+------+---------------+------+---------+----------
---+------+-------+
|  1 | SIMPLE      | b     | ALL  | PRIMARY       | NULL | NULL    | NULL
|  986 |       |
|  1 | SIMPLE      | a     | ref  | pid           | pid  | 4       |
schema.b.pid |   70 |       |
+----+-------------+-------+------+---------------+------+---------+----------
---+------+-------+
2 rows in set (0.00 sec)
```

`table1` has 166208 rows; `table2` has about 1000 rows.

This is a very simple case; it doesn't by any means prove that the query optimizer wouldn't get confused and generate different plans in a more complicated case.

Share Improve this answer Follow

answered Apr 25, 2012 at 1:43

Matt Fenwick
49.1k ● 24 ● 129 ● 198

1 This should be the accepted answer. This is correct, the plan is the same (or close to with bigger statements) but the amount of records will be drastic, thus causing difference in performance. – SovietFrontier May 13, 2019 at 16:16

postgresql.org/docs/14/explicit-joins.html "Controlling the Planner with Explicit `JOIN` Clauses" has some comments that may be of interest for PostgreSQL. Also: dba.stackexchange.com/questions/198182/… – Ciro Santilli OurBigBook.com Sep 17, 2022 at 6:29

48

The second syntax has the unwanted possibility of a cross join: you can add tables to the FROM part without corresponding WHERE clause. This is considered harmful.

Share

Improve this answer

Follow

edited Jul 12, 2010 at 16:59

answered Nov 25, 2008 at 14:13

edosoft
17.3k ● 25 ● 80 ● 114

What if the table names in the from clause are generated from the tables used in the where clause? – Jus12 Sep 2, 2015 at 16:59

1 you can do a cross join with the explicit JOIN syntax as well. (stackoverflow.com/a/44438026/929164) you probably meant that it is less strict, thus more prone to user error. – Daniel Dror Jan 2, 2020 at 12:30

18

The first answer you gave uses what is known as ANSI join syntax, the other is valid and will work in any relational database.

I agree with grom that you should use ANSI join syntax. As they said, the main reason is for clarity. Rather than having a where clause with lots of predicates, some of which join tables and others restricting the rows returned with the ANSI join syntax you are making it blindingly clear which conditions are being used to join your tables and which are being used to restrict the results.

Share Improve this answer Follow

answered Sep 7, 2008 at 9:55

andy47

913 ● 2 ● 9 ● 18

7

@lomaxx: Just to clarify, I'm pretty certain that both above syntax are supported by SQL Serv 2005. The syntax below is NOT supported however

```
select a.*, b.*
from table a, table b
where a.id *= b.id;
```

Specifically, the outer join (*=) is not supported.

Share

Improve this answer

Follow

edited Sep 28, 2011 at 13:35

marc_s

753k ● 183 ● 1.4k ● 1.5k

answered Sep 4, 2008 at 23:39

deadbug

4,809 ● 4 ● 27 ● 23

2 Frankly I wouldn't use it even in SQL Server 2000, the *= syntax often gives wrong answers. Sometimes it interprets these as cross joins. – HLGEM Mar 18, 2009 at 17:21

> Performance wise, they are exactly the same (at least in SQL Server) but be aware that they are deprecating this join syntax and it's not supported by sql

6

> server2005 out of the box.

I think you are thinking of the deprecated *= and =* operators vs. "outer join".

I have just now tested the two formats given, and they work properly on a SQL Server 2008 database. In my case they yielded identical execution plans, but I couldn't confidently say that this would always be true.

Share Improve this answer Follow

answered Sep 4, 2008 at 23:33

Joshdan
156 ● 1 ● 3

4

On some databases (notably Oracle) the order of the joins can make a huge difference to query performance (if there are more than two tables). On one application, we had literally two orders of magnitude difference in some cases. Using the inner join syntax gives you control over this - if you use the right hints syntax.

You didn't specify which database you're using, but probability suggests SQL Server or MySQL where there it makes no real difference.

Share Improve this answer Follow

answered Sep 4, 2008 at 23:38

Leigh Caldwell
11k ● 4 ● 26 ● 31

1 Leigh, you can use the hints in implicit joins too. – SquareCog Oct 30, 2008 at 1:26

1 In Oracle it is extremely rare for the join order to affect the execution plan in a meaningful way. See this article by Jonathan Lewis for an explanation. – Jon Heller Jun 24, 2013 at 22:49

3

As Leigh Caldwell has stated, the query optimizer can produce different query plans based on what functionally looks like the same SQL statement. For further reading on this, have a look at the following two blog postings:-

One posting from the Oracle Optimizer Team

Another posting from the "Structured Data" blog

I hope you find this interesting.

Share Improve this answer Follow

answered Sep 17, 2008 at 17:44

Mike McAllister
1,549 ● 2 ● 12 ● 15

Mike, the difference they are talking about is that you need to be sure that if you specify an explicit join, you specify the *join condition* to join on, not the filter. You will note that for semantically correct queries, the exec plan is the same. – SquareCog Oct 30, 2008 at 1:34

2

Basically, the difference between the two is that one is written in the old way, while the other is written in the modern way. Personally, I prefer the modern script using the inner, left, outer, right definitions because they are more explanatory and makes the code more readable.

When dealing with inner joins there is no real difference in readability neither, however, it may get complicated when dealing with left and right joins as in the older method you would get something like this:

```
SELECT *
FROM table a, table b
WHERE a.id = b.id (+);
```

The above is the old way how a left join is written as opposed to the following:

```
SELECT *
FROM table a
LEFT JOIN table b ON a.id = b.id;
```

As you can visually see, the modern way of how the script is written makes the query more readable. (By the way same goes for right joins and a little more complicated for outer joins).

Going back to the boiler plate, it doesn't make a difference to the SQL compiler how the query is written as it handles them in the same way. I've seen a mix of both in Oracle databases which have had many people writing into it, both elder and younger ones. Again, it boils down to how readable the script is and the team you are developing with.

Share Improve this answer Follow

answered May 2, 2019 at 11:00

Michele La Ferla

6,884 ● 11 ● 55 ● 82

0

Performance wise, it should not make any difference. The explicit join syntax seems cleaner to me as it clearly defines relationships between tables in the from clause and does not clutter up the where clause.

Share Improve this answer Follow

answered Nov 30, 2011 at 18:30

David
29 • 2

-1

In my experience, using the cross-join-with-a-where-clause syntax often produces a brain damaged execution plan, especially if you are using a Microsoft SQL product. The way that SQL Server attempts to estimate table row counts, for instance, is savagely horrible. Using the inner join syntax gives you some control over how the query is executed. So from a practical point of view, given the atavistic nature of current database technology, you have to go with the inner join.

Share Improve this answer Follow

answered Aug 13, 2015 at 18:09

Sean
15 • 1

5 Do you have any proof of this? Because the accepted answer says otherwise. – cimmanon Aug 13, 2015 at 18:17

Highly active question. Earn 10 reputation (not counting the association bonus) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.