**Avoid ORDER BY in SQL Server views**

By: [Aaron Bertrand](https://www.mssqltips.com/sqlserverauthor/49/aaron-bertrand/)   |   Last Updated: 2016-10-04

FROM: <https://www.mssqltips.com/sqlservertip/4488/avoid-order-by-in-sql-server-views/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20190724>

**Problem**

For as long as I have been working with SQL Server, I have watched people add ORDER BY to views for various reasons - most commonly, they are referencing the view in multiple queries, and want to avoid having to repeat an ORDER BY clause in each of those outer queries.

**Solution**

ORDER BY in a view is not something you can rely on. In fact, it's not even something you can accomplish on its own. Let's look at a few examples using the new [WideWorldImporters sample database](https://blogs.technet.microsoft.com/dataplatforminsider/2016/06/09/wideworldimporters-the-new-sql-server-sample-database):

USE WideWorldImporters;

GO

CREATE VIEW dbo.CustomersByName

AS

SELECT CustomerID, CustomerName, DeliveryCityID

FROM Sales.Customers

ORDER BY CustomerName;

GO

This results in a very explicit error message:

Msg 1033, Level 15, State 1, Procedure CustomersByName  
The ORDER BY clause is invalid in views, inline functions, derived tables, subqueries, and common table expressions, unless TOP, OFFSET or FOR XML is also specified.

Sure, there are kludges to get it in there, most of which I see in the wild are \*exactly\* this:

SELECT TOP (100) PERCENT <columns>

FROM dbo.<table>

ORDER BY <column>;

If we do that with our query above:

CREATE VIEW dbo.CustomersByName

AS

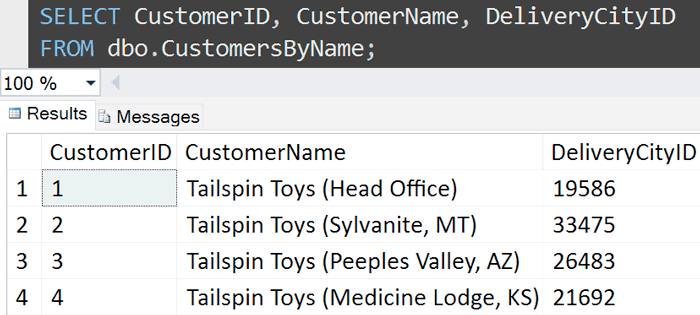
SELECT TOP (100) PERCENT CustomerID, CustomerName, DeliveryCityID

FROM Sales.Customers

ORDER BY CustomerName;

GO

The view is successfully created, but we can easily see that an outer query against the view, without an ORDER BY, won't obey the ORDER BY from inside the view:



This used to work, back in the SQL Server 2000 days, but not in modern versions. Let's compare the execution plans between a query against the view (which returns data sorted by CustomerID) and running the query \*inside\* the view directly (which returns data sorted by CustomerName):

SELECT CustomerID, CustomerName, DeliveryCityID

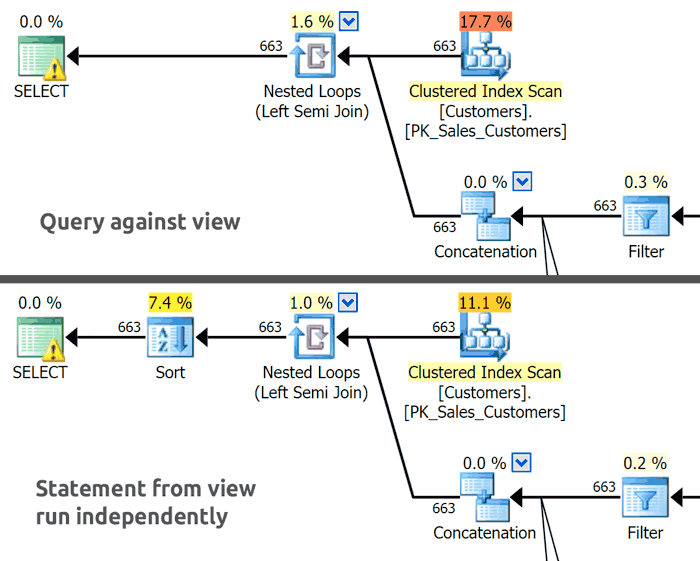
FROM dbo.CustomersByName;

SELECT TOP (100) PERCENT CustomerID, CustomerName, DeliveryCityID

FROM Sales.Customers

ORDER BY CustomerName;

The execution plans are virtually identical, except for one little thing (see if you can spot it):



When we run the query from inside the view (without actually referencing the view itself), the results are ordered as we desired, and this is facilitated by that extra Sort operator. This doesn't happen when we query against the view because, essentially, SQL Server looks at our outer query, sees there is no ORDER BY, and says, "they don't care about the order of results," so feels free to return the data in the most efficient order (which happens to be by CustomerID in this example, but that won't always be true, depending on the columns in the query, other indexes, and other factors).

Since the ORDER BY inside the view is only allowed to exist as a way to determine which TOP rows to include, and since TOP (100) PERCENT means "return all the rows," these are two pieces of logic SQL Server feels confident about completely throwing away and not considering at all. This is why there is no Sort operator in the first plan, and no Top operator in \*either\* plan.

The underlying problem here is that ORDER BY can serve these two functions - one to determine TOP inclusion, and one to determine presentation order. There is no way to prioritize these; in the second query above, the ORDER BY is actually serving both functions. Ideally, SQL Server's TOP clause should have been implemented with its own ORDER BY, so that if we wanted to return the first 10 customers alphabetically but have the results sorted by DeliveryCityID, we could say something like this:

SELECT TOP (10) OVER (ORDER BY CustomerName)

CustomerID, CustomerName, DeliveryCityID

FROM Sales.Customers

ORDER BY DeliveryCityID;

Instead, we have to write our queries a little more elaborately, where we have to perform the two different functions of ORDER BY in two different stages:

SELECT CustomerID, CustomerName, DeliveryCityID

FROM

(

SELECT TOP (10)

CustomerID, CustomerName, DeliveryCityID

FROM Sales.Customers

ORDER BY CustomerName

) AS x

ORDER BY DeliveryCityID;

*(A more common example of this is when you want to present, say, a top 10 list ordered from 10 to 1.)*

[OFFSET/FETCH syntax](https://www.mssqltips.com/sqlservertip/2362/overview-of-offset-and-fetch-feature-of-sql-server-2012/) was introduced in SQL Server 2012, which seems like it could help to solve the problem, but unfortunately it still uses the same overloaded ORDER BY clause.

**Are there other kludges to get the "desired" behavior?**

Of course. There are always going to be ways to get SQL Server to bend to your will, even if only temporarily - by using a percentage so close to 100 percent that it will round up, or by using the upper bound of the BIGINT type:

CREATE VIEW dbo.CustomersByName\_KludgeOne

AS

SELECT TOP (99.9999999999999999) PERCENT CustomerID, CustomerName, DeliveryCityID

FROM Sales.Customers

ORDER BY CustomerName;

GO

CREATE VIEW dbo.CustomersByName\_KludgeTwo

AS

SELECT TOP (9223372036854775807) CustomerID, CustomerName, DeliveryCityID

FROM Sales.Customers

ORDER BY CustomerName;

GO

Sure enough, if you select from each view without an ORDER BY, the results are ordered by name, and the Sort operator is re-introduced into the plan. You'll also notice a new Top operator that wasn't present before, even though no rows are filtered out in either case.

But **please** don't learn from this or rely on it; like the original functionality that was phased away starting in SQL Server 2005, there is always a chance that some future build of SQL Server will "fix" this loophole, and it will stop behaving as you observe. Never mind that even when you \*can\* get the view to return data in a certain order, this may not be the order desired by all queries that reference the view (whether they do so explicitly or not). SQL Server will not be able to determine how to resolve conflicts or prioritization issues with outer queries that have their own, different, ORDER BY clause. You could guess that the outer ORDER BY will always win, but this isn't documented, so is just as unreliable as the other behavior.

Summary

Views are not meant to dictate presentation order; if you expect queries against a view from presenting the data in a predictable order, please stop looking for kludges and add those ORDER BY clauses to your outer queries. Having ORDER BY inside the view is not going to work in all cases, and is going to be confusing for people reading or maintaining the code.

**Next Steps**

See these tips and other resources:

* [Trick to Optimize TOP Clause in SQL Server](https://www.mssqltips.com/sqlservertip/2053/trick-to-optimize-top-clause-in-sql-server/)
* [Overview of OFFSET and FETCH Feature of SQL Server 2012](https://www.mssqltips.com/sqlservertip/2362/overview-of-offset-and-fetch-feature-of-sql-server-2012/)
* [Pagination with OFFSET / FETCH : A better way](http://sqlperformance.com/2015/01/t-sql-queries/pagination-with-offset-fetch)
* [Comparing performance for different SQL Server paging methods](https://www.mssqltips.com/sqlservertip/2696/comparing-performance-for-different-sql-server-paging-methods/)
* [TOP (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms189463.aspx) (MSDN)
* [ORDER BY Clause (Transact-SQL)](https://msdn.microsoft.com/en-us/library/ms188385.aspx) (MSDN)

Last Updated: 2016-10-04

[**View all my tips**](https://www.mssqltips.com/sqlserverauthor/49/aaron-bertrand/)

|  |  |
| --- | --- |
| **Friday, September 22, 2017 - 8:57:11 PM - parag** | [Back To Top](https://www.mssqltips.com/sqlservertip/4488/avoid-order-by-in-sql-server-views/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20190724#top) |
| use demo  go    SELECT TOP (10) OVER (ORDER BY custname desc)      custid, custname, city    FROM Sales.Customers    ORDER BY city;      Guys I have modified code as per my table but I am getting error running this query.    \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Msg 156, Level 15, State 1, Line 2  Incorrect syntax near the keyword 'OVER'.      I checked code online and think code is correct.I am using sql server 2012 ent edition.    Also below query is providing me desired output.    use demo  go      SELECT custid, contactname, city    FROM    (      SELECT TOP (10)        custid, contactname, city      FROM Sales.Customers      ORDER BY contactname    ) AS x    ORDER BY city;      custid      contactname                    city  ----------- ------------------------------ ---------------  1           Allen, Michael                 Berlin  28          Cavaglieri, Giorgio            Lisboa  4           Arndt, Torsten                 London  16          Birkby, Dana                   London  19          Boseman, Randall               London  22          Bueno, Janaina Burdan, Neville Madrid  13          Benito, Almudena               México D.F.  25          Carlson, Jason                 München  7           Bansal, Dushyant               Strasbourg  10          Bassols, Pilar Colome          Tsawassen    (10 row(s) affected)    Can anyone check first query and let me know if there is any error? | |

|  |  |
| --- | --- |
| **Monday, October 10, 2016 - 3:28:41 PM - Joe Celko** | [Back To Top](https://www.mssqltips.com/sqlservertip/4488/avoid-order-by-in-sql-server-views/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20190724#top) |
| Why do you think I am opposed to views? When I am teaching SQL, I tell people that ideally you want to give each class of users a collection of views that makes it look as if the schema were designed *for them and only for them* (the salesman can see sales data, but cannot go over to human resources and give himself a raise in the personnel tables).  Actually, I would guess the views in actual practice are used to make sure that everybody is singing from the same page. Complicated calculations will be done one and only one way with a view (I have a horror story from 25+ years ago about computing legal age two different ways in the same application). Access to only certain columns is restricted by a view.  Again, in order by clause on a view or a table makes no sense. They are supposed to model sets and sets by definition have no ordering. Today, with reporting servers and other display options, there is really no need to go back to the 1980's Sybase model built on top of UNIX sequential files. Do presentation in a presentation layer.  If you want to learn some neat tricks with views, look up the WITH CHECK OPTION, which almost nobody knows about. | |

|  |  |
| --- | --- |
| **Monday, October 10, 2016 - 12:12:47 PM - Greg Robidoux** | [Back To Top](https://www.mssqltips.com/sqlservertip/4488/avoid-order-by-in-sql-server-views/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20190724#top) |
| When selecting data from views, they should be handled like tables and not assumed the data is returned in any particular order. As Aaron points out there are ways to get around this to include an ORDER BY in the view, but these are solutions that present other issues. It's a cleaner approach to use an ORDER BY when querying data from the view instead of trying to include an ORDER BY in the view.  -Greg | |

|  |  |
| --- | --- |
| **Monday, October 10, 2016 - 8:06:19 AM - Aaron Bertrand** | [Back To Top](https://www.mssqltips.com/sqlservertip/4488/avoid-order-by-in-sql-server-views/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20190724#top) |
| Hey Sadheera, did you read the article or just the headline? It's tough to add context like that to the headline. Every single view I've ever seen with ORDER BY has had either TOP 100 PERCENT or TOP 2 billion. You have a different use case? You have views in production that is TOP and ORDER BY but actually use the ORDER BY as a filter? Congratulations! Then if you read the article you know the advice is not for you. I also did not suggest the extreme position that views don't belong at all. If you guys want to argue with Celko about that, don't hijack this article please. | |

|  |  |
| --- | --- |
| **Monday, October 10, 2016 - 7:36:54 AM - Vlad** | [Back To Top](https://www.mssqltips.com/sqlservertip/4488/avoid-order-by-in-sql-server-views/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20190724#top) |
| thanks**Sadheera,** but my reply was mainly not to dispute the article. Article simply provides the information about why ORDER BY should not be used today, and explains that new databases do not work as expected or as they used to work, and that is by design, not erroneous.  what I was disputing is the POV in "Joe Celko"  where implication was that using OREDR BY in a view was and is an anachronism left over from the days of mainframe and DOS computing, where flat file was the KING and should have been eliminated long before now.  I was trying to point out that there were and still are very good reasons why an ORDER BY in a view can or maybe even should be used at times. It is a useful tool, not an annoyance to be rid of, just as the view itself is a tool. | |

|  |  |
| --- | --- |
| **Sunday, October 09, 2016 - 10:41:12 PM - Sadheera** | [Back To Top](https://www.mssqltips.com/sqlservertip/4488/avoid-order-by-in-sql-server-views/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20190724#top) |
| Yes, this article is misleading as Vlad has mentioned above. Order by clause in the view, with a TOP clause can be useful when you want to retrieve most recent data etc.. | |

|  |  |
| --- | --- |
| **Friday, October 07, 2016 - 8:47:15 AM - Vlad** | [Back To Top](https://www.mssqltips.com/sqlservertip/4488/avoid-order-by-in-sql-server-views/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20190724#top) |
| **Joe Celko, I absolutely disagree with that comment of yours.**  **in many years working with databases, mostly with MS-SQL Server DB.**  **I have seen many reasons why people would want to use ORDER BY with in a View.**  **and none of them was because  "**because they truly believe that a view is a deck of punch cards**"**  **mostly, and that is because it has worked as expected in older versions, people would use ORDER BY in views is to ensure that a data-set presented to the querying client is always in the same order. no matter what. otherwise we can just as well query the table directly, why add the overhead.**  **in most cases Views are used to pre-process the data in some way or the other.**  **to have a specific data representation needed for a set of clients.**  **filter the data, add/remove things to minimize processing on client side.**  **it was done at the time when servers had more power than clients, when you could not expect that a data user (an application used to access the data) is run on hardware that have enough resources to actually do the data manipulation on site.**    **who uses the View locally (with in SQL jobs run on database itself)?  now one.**  **views are designed to be used for off-site(out side the database server) access.**  **to be used for client access not internal processing.**    **even IBM acknowledge and encourage the usefulness of Views.  ever hear about Logical Files?**  **yep, that is basically a view where you can do some data pre-processing, like \*\*GASP!!\*\* ordering records in specific way before passing it to the client.**    **many relational databases have been encouraging developers to use Views for some degree of data manipulation for years.**  **and now you expect everyone all of a sudden o kick the habit and move all to client side.**    **it is not such a big deal per-see, but it will take time.** | |

|  |  |
| --- | --- |
| **Tuesday, October 04, 2016 - 10:20:21 AM - Joe Celko** | [Back To Top](https://www.mssqltips.com/sqlservertip/4488/avoid-order-by-in-sql-server-views/?utm_source=dailynewsletter&utm_medium=email&utm_content=headline&utm_campaign=20190724#top) |
| I believe that the real reason people put an ORDER BY on a view is not to save sorting later, because they truly believe that a view is a deck of punch cards! Escaping from the old file system mentality (contiguous fields that make contiguous records that make contiguous stored files) is really hard to shake. Ever try to learn a totally different language, like Japanese? Sentences do not have subjects; they have topics; there are no articles; there are no plurals; and yet somehow Japan has been able to have a civilization without any of these things.  Today, the tiered architecture of client/server is getting really extended to what it should have been. The database server (literally a physical unit of hardware) passes a result set to a reporting server (dedicated and configured to serve something like Crystal Reports). The sorting and aggregation are done in the reporting server. Or the fancy crunching is done in the analytic server (fancy floating-point processor, tuned for statistical analysis), the broadcast server (designed to format and then send things to mobile phones), etc.  We never intended for ORDER BY to be viewed as a clause for a select statement. It was always part of the cursor declaration in the ANSI/ISO standards. But in the early days of 16-bit minicomputers, the SQL products made concessions. The Sybase MONEY datatype let COBOL programmers do display formatting in their query, for example. Likewise, the CONVERT() string function replaced the picture clause of COBOL.  Languages, much like all programming, follow the rule that there is nothing more permanent than a temporary fix :-) | |