

SQL Programming

Questions and Answers

te that when using PDO you have to do this. To fix this you have to do this:

```
$connection = new PDO('mysql:dbname=dbtest;host=127.0.0.1; charset=utf8', $username, $password);
$connection->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
$connection->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

In the above example the error mode isn't strictly necessary, **but it is advised to add it**. The script will not stop with a Fatal Error when something goes wrong. And it gives the chance to catch any error(s) which are thrown as PDOException s.

What is **mandatory** however is the first `setAttribute()` line, which tells PDO to dis-

able prepared statements and use real prepared statements. This makes sure the statement values aren't parsed by PHP before sending it to the MySQL server (giving a possibility to inject malicious SQL).

Although you can set the `charset` in the options of the constructor, it's important to note that though you can set the `charset` in the options of the constructor, it's important to note that in earlier versions of PHP (< 5.3.6) silently ignored the `charset` parameter in the DSN.

Explanation

What happens is that the SQL statement you pass to `prepare` is parsed and compiled by the database server. By specifying parameters (either a `?` or a named parameter like `:name` above) you tell the database engine where you want to filter on. Then when you `execute`, the prepared statement is combined with the parameter values you specified.

The important thing here is that the parameter values are combined with the compiled SQL string. SQL injection works by tricking the script into including malicious strings into the SQL to send to the database. So by sending the actual SQL separately from the prepared statement will just be treated as strings (although the database engine may optimize so parameters may end up as numbers too, of course). In the example above, if the `:name` variable contains `'Sarah'; DELETE FROM employees`, the result would simply be `string "Sarah"; DELETE FROM employees`, and you will not end up with an employee named Sarah.

Another benefit with using prepared statements is that if you execute the same statement in the same session it will only be parsed and compiled once, giving you some speed. And since you asked about how to do it for an insert, here's an example:

```
$preparedStatement = $db->prepare('INSERT INTO table (column1, column2) VALUES (:value1, :value2)');
$preparedStatement->execute(array('column1' => $value1, 'column2' => $value2));
```

An Prepared Statement Example

by George Duckett

Table of Contents

[About this book](#)

[SQL](#) (3 questions)

[SQL Server](#) (129 questions)

[MySQL](#) (85 questions)

[Tsql](#) (59 questions)

[Database](#) (47 questions)

[SQL Server 2005](#) (26 questions)

[Oracle](#) (23 questions)

[SQL Server 2008](#) (18 questions)

[PostgreSQL](#) (17 questions)

[Performance](#) (15 questions)

[JOIN](#) (14 questions)

[DateTime](#) (13 questions)

[Database Design](#) (13 questions)

[Select](#) (10 questions)

[PHP](#) (9 questions)

[C#](#) (8 questions)

[Java](#) (8 questions)

[GROUP BY](#) (8 questions)

[SQLite](#) (6 questions)

[DISTINCT](#) (6 questions)

[Django](#) (5 questions)

[SSMS](#) (5 questions)

[LINQ To SQL](#) (5 questions)

[Random](#) (4 questions)

[Duplicates](#) (4 questions)

[NoSQL](#) (4 questions)

[LINQ](#) (3 questions)

[ORM](#) (3 questions)

[Foreign Keys](#) (3 questions)

[CASE](#) (3 questions)

[String Concatenation](#) (3 questions)

[Case Sensitive](#) (3 questions)

[Indexing](#) (2 questions)

[Triggers](#) (2 questions)

[Sql Order By](#) (2 questions)

[Hierarchical Data](#) (2 questions)

[HTML](#) (1 question)

[Eclipse](#) (1 question)

[Sorting](#) (1 question)

[Console](#) (1 question)

[UNION](#) (1 question)

[Ms Access](#) (1 question)

[Anti Patterns](#) (1 question)

[Core Data](#) (1 question)

[Dynamic SQL](#) (1 question)

[Cursor](#) (1 question)

[Copyright](#)

About this book

This book has been divided into categories where each question belongs to one or more categories. The categories are listed based on how many questions they have; the question appears in the most popular category. Everything is linked internally, so when browsing a category you can easily flip through the questions contained within it. Where possible links within questions and answers link to appropriate places within in the book. If a link doesn't link to within the book, then it gets a special icon, like [this](#).

SQL

[Skip to questions,](#)

Wiki by user [john-saunders](#) 

SQL stands for *Structured Query Language*.

One subset of the SQL standard is **DDL** (Data Definition Language), which is used to create tables and constraints. These include:

- CREATE
- DROP
- ALTER

Another subset is **DML** (Data Manipulation Language), which is used to modify and view data within the database:

- SELECT
- INSERT
- UPDATE
- DELETE

The final “standard” subset of commands is **DCL** (Data Control Language):

- GRANT
- REVOKE

Many database implementations require the use of SQL, and over the years, vendors have implemented dialects of SQL to provide more functionality as well as simplify it. Because of these deviations from the standard, SQL is fractured - syntax that works on one implementation does not necessarily work on another.

ISO/IEC (formerly ANSI) standards have been beneficial in resolving such situations, but adoption is selective. Queries conforming to these standards should be portable to other databases, though performance may vary.

Most DBMSs have additional languages for writing stored procedures. In Oracle this is PL/SQL (Procedural Language/Structured Query Language), in PostgreSQL it's PL/pgSQL (Procedural Language/PostgreSQL). Outside of stored procedures or functions, Oracle and PostgreSQL use SQL. Thus the tags [plsql](#)  and [plpgsql](#)  should only be used for problems directly related to writing stored procedures. Microsoft SQL Server uses the term T-SQL (Transact-SQL)([tsql](#)) for both “plain” SQL (queries, DML, ..) and the language used for stored procedures.

Tagging Recommendation

This tag should be used for general SQL programming language questions, in addition to tags for specific products. For example, questions about Microsoft SQL Server should use the [sql-server](#) tag, while questions regarding MySQL should use the [mysql](#) tag. SQL is the umbrella under which these products exist; tagging them by product (including version, e.g [oracle11g](#), [sql-server-2008](#)) is the easiest way to know what functionality is available for the task at hand. It is very common for [mysql](#) questions to omit this tag because query discussions on MySQL are more often stated as MySQL rather than SQL in general.

Please read this [summary](#) about the SQL standard (the 1992 one in this case, broadly implemented) and if you can, refer to the [book itself](#).

Free SQL Programming Books

- [Developing Time-Oriented Database Applications in SQL](#)
- [Use The Index, Luke!: A Guide To SQL Database Performance](#)
- [Learn SQL The Hard Way](#)
- [SQL Tutorial For Starters](#)
- [SQL - Free books](#)
- [SQL - Free books 2](#)

Free SQL/Database Online Courses

- [Coursera-Introduction to Databases](#)
- [Stanford Online-DB: Introduction to Databases](#)

SQL/Database Online Tutorial.

- [Codeschool Try SQL](#)

Useful Resources

While you should always provide complete code examples (e.g. schema, data sample and expected result) in your question or answer, you can also isolate problematic code and reproduce it in an online environment such as [SQL Fiddle](#), and link to that.

MySQL is managed and maintained by Oracle and in-depth documentation can be found at the [MySQL website](#).

More specific tags

When you are asking a question about SQL you can also add more specific tags. Here is the list of available tags:

- [sqlbulkcopy](#)
- [sqlconnection](#)
- [sqlcommand](#)
- [sql-copy](#)

- [sql-convert](#) 
- [sql-delete](#) 
- [sql-drop](#) 
- [sqldatetime](#) 
- [sql-date-functions](#) 
- [sql-function](#) 
- [sql-job](#) 
- [sql-like](#) 
- [sql-limit](#) 
- [sql-merge](#) 
- [sql-order-by](#)
- [sql-returning](#) 
- [sql-server-job](#) 
- [to-date](#) 
- [triggers](#)
- [sql-timestamp](#) 
- [sql-update](#) 
- [sql-view](#) 
- [stored-procedures](#) 
- [having](#) 
- [where-clause](#) 
- [count](#) 
- [group-by](#)
- [join](#)
- [left-join](#) 
- [inner-join](#) 
- [outer-join](#) 
- [self-join](#) 
- [cross-join](#) 
- [right-join](#) 
- [full-outer-join](#) 
- [natural-join](#) 

Implementation specific tags

You can specify your question by adding the implementation you used as a tag.

- [mysql](#)
 - [mysqli](#) 
 - [sql-server](#)
 - [oracle](#)
 - [postgresql](#)
 - [sqlite](#)
 - [db2](#) 
-

Questions

[Q: In SQL, what's the difference between count\(column\) and count\(*\)?](#)

Tags: [sql](#) ([Next Q](#))

I have the following query:

```
select column_name, count(column_name)
from table
group by column_name
having count(column_name) > 1;
```

What would be the difference if I replaced all calls to count(column_name) to count(*)?

This question was inspired by [How do I find duplicate values in a table in Oracle?](#).

To clarify the accepted answer (and maybe my question), replacing count(column_name) with count(*) would return an extra row in the result that contains a null and the count of null values in the column.

Tags: [sql](#) ([Next Q](#))

User: [bill-the-lizard](#) 

[Answer](#)  by [sqlmenace](#) 

count(*) counts NULLs and count(column) does not

[edit] added this code so that people can run it

[Skip code block](#)

```
create table #bla(id int,id2 int)
insert #bla values(null,null)
insert #bla values(1,null)
insert #bla values(null,1)
insert #bla values(1,null)
insert #bla values(null,1)
insert #bla values(1,null)
insert #bla values(null,null)

select count(*),count(id),count(id2)
from #bla
```

results 7 3 2

Tags: [sql](#) ([Next Q](#))

[Q: What is the reason not to use select *?](#)

Tags: [sql](#) ([Prev Q](#)) ([Next Q](#))

I've seen a number of people claim that you should specifically name each column you want in your select query.

Assuming I'm going to use all of the columns anyway, why would I not use SELECT *?

Even considering the question [*SQL query - Select * from view or Select col1, col2, ... colN from view](#)*, I don't think this is an exact duplicate as I'm approaching the issue from a slightly different perspective.

One of our principles is to not optimize before it's time. With that in mind, it seems like using `SELECT *` should be the **preferred** method until it is proven to be a resource issue or the schema is pretty much set in stone. Which, as we know, won't occur until development is completely done.

That said, is there an overriding issue to not use `SELECT *`?

Tags: [sql](#) ([Prev Q](#)) ([Next Q](#))

User: [notme](#)

[Answer](#) by [robert-paulson](#)

The essence of the quote of not prematurely optimizing is to go for simple and straightforward code and **then** use a profiler to point out the hot spots, which you can then optimize to be efficient.

When you use `select *` you're making it impossible to profile, therefore you're not writing clear & straightforward code and you are going against the spirit of the quote. `select *` is an anti-pattern.

So selecting columns is not a premature optimization. A few things off the top of my head

....

1. If you specify columns in a SQL statement, the SQL execution engine will error if that column is removed from the table and the query is executed.
2. You can more easily scan code where that column is being used.
3. You should always write queries to bring back the least amount of information.
4. As others mention if you use ordinal column access you should never use `select *`
5. If your SQL statement joins tables, `select *` gives you all columns from all tables in the join

The corollary is that using `select * ...`

1. The columns used by the application is opaque
 2. DBA's and their query profilers are unable to help your application's poor performance
 3. The code is more brittle when changes occur
 4. Your database and network are suffering because they are bringing back too much data (I/O)
 5. Database engine optimizations are minimal as you're bringing back all data regardless (logical).
-

Writing correct SQL is just as easy as writing `Select *`. So the real lazy person writes proper SQL because they don't want to revisit the code and try to remember what they

were doing when they did it. They don't want to explain to the DBA's about every bit of code. They don't want to explain to their clients why the application runs like a dog.

Tags: [sql](#) ([Prev Q](#)) ([Next Q](#))

Q: What's the difference between NOT EXISTS vs. NOT IN vs. LEFT JOIN WHERE IS NULL?

Tags: [sql](#) ([Prev Q](#))

It seems to me that you can do the same thing in a SQL query using either NOT EXISTS, NOT IN, or LEFT JOIN WHERE IS NULL. For example:

```
SELECT a FROM table1 WHERE a NOT IN (SELECT a FROM table2)
SELECT a FROM table1 WHERE NOT EXISTS (SELECT * FROM table2 WHERE table1.a = table2.a)
SELECT a FROM table1 LEFT JOIN table2 ON table1.a = table2.a WHERE table1.a IS NULL
```

I'm not sure if I got all the syntax correct, but these are the general techniques I've seen. Why would I choose to use one over the other? Does performance differ...? Which one of these is the fastest / most efficient? (If it depends on implementation, when would I use each one?)

Tags: [sql](#) ([Prev Q](#))

User: [froadie](#)

Answer by [quassnoi](#)

- [NOT IN vs. NOT EXISTS vs. LEFT JOIN / IS NULL: SQL Server](#)
- [NOT IN vs. NOT EXISTS vs. LEFT JOIN / IS NULL: PostgreSQL](#)
- [NOT IN vs. NOT EXISTS vs. LEFT JOIN / IS NULL: Oracle](#)
- [NOT IN vs. NOT EXISTS vs. LEFT JOIN / IS NULL: MySQL](#)

In a nutshell:

NOT IN is a little bit different: it never matches if there is but a single NULL in the list.

- In MySQL, NOT EXISTS is a little bit less efficient
- In SQL Server, LEFT JOIN / IS NULL is less efficient
- In PostgreSQL, NOT IN is less efficient
- In Oracle, all three methods are the same.

Tags: [sql](#) ([Prev Q](#))

SQL Server

[Skip to questions](#),

Wiki by user [blorgbeard](#) 

[Microsoft's SQL Server](#)  is a suite of ***relational database management system*** (RDBMS) products providing multi-user database access functionality. It originated from the Sybase SQL Server 4.x codebase and *Transact-SQL dialect* (T-SQL), but it has forked significantly since then.

SQL Server is available in multiple versions, typically identified by release year, and versions are subdivided into *editions* to distinguish between product functionality. The latest released version is [SQL Server 2014](#) .

The SQL Server product range is split broadly into six categories:

1. [SQL Server](#)  ([sql-server](#)) is the main suite of enterprise and developer server products. Primary differences are licensing costs, capacities, and components included in the product, with some minor differences supported language features. Standard components include database language and storage server, developer tools, [ETL](#)  tools, schedulers, and replication. Other components include OLAP, reporting, and parallel computation. Components runs as NT Services.
2. [SQL Server Express](#)  ([sql-server-express](#)) is free for use and distribution but has reduced engine performance, functionality and capacity than found in its other server siblings. It is focused on small deployments and runs as an NT Service.
3. [SQL Server Compact Edition](#) ([sql-server-ce](#)) is an embeddable subset of SQL Server. Like the Express edition it has a reduced language, functionality and capacity, but it is free to distribute. It's focused on small installations and desktop applications where its small footprint and no-management-required features are a great advantage.

Note: [SQL Server Compact Edition](#) is [deprecated](#). Customers should use SQL Server Express (and possibly [LocalDB](#)).

4. [Azure SQL Database](#)  ([sql-azure](#)) is a completely managed, hosted, high-availability instance of SQL Server 2005 with some language syntax support for federated queries, operated in [Microsoft Azure](#)  datacenters.
5. [SQL Server Analytics Platform System](#)  (or APS), formerly known as **SQL Server Parallel Data Warehouse (PDW)**, is a pre-built data warehouse appliance that offers massively parallel processing for SQL Server, allowing support for many hundreds of terabytes.
6. [Azure SQL Data Warehouse](#)  is an enterprise-class distributed database in the Azure Cloud capable of processing up to petabyte volumes of relational and non-relational data. It is the industry's first cloud data warehouse with grow, shrink, and pause in seconds.

SQL Server Release History

[Skip code block](#)

Version	Year	Release Name	Codename
1.0 (OS/2)	1989	SQL Server 1.0 (16 bit)	Ashton-Tate
1.1 (OS/2)	1991	SQL Server 1.1 (16 bit)	-
4.21(WinNT)	1993	SQL Server 4.21	SQLNT
6.0	1995	SQL Server 6.0	SQL95
6.5	1996	SQL Server 6.5	Hydra
7.0	1998	SQL Server 7.0	Sphinx
-	1999	SQL Server 7.0 OLAP Tools	Palato mania
8.0	2000	SQL Server 2000	Shiloh
8.0	2003	SQL Server 2000 x64	Liberty
9.0	2005	SQL Server 2005	Yukon
10.0	2008	SQL Server 2008	Katmai
10.25	2010	SQL Azure DB	CloudDatabase
10.5	2010	SQL Server 2008 R2	Kilimanjaro (aka KJ)
11.0	2012	SQL Server 2012	Denali
12.0	2014	SQL Server 2014	Hekaton

References

- [MSDN SQL Server 2005 Transact-SQL Reference](#) 
- [MSDN SQL Server 2008 Transact-SQL Reference](#) 
- [MSDN SQL Server 2008 R2 Transact-SQL Reference](#) 
- [MSDN SQL Server 2012 Transact-SQL Reference](#) 
- [MSDN SQL Server 2014 Transact-SQL Reference](#) 
- [SQL Server Wikipedia Article](#) 
- [SQL Azure Stack-Overflow Tag](#) 
- [MSDN SQL Server DevCenter](#) 
- [SQL Server Best Practices](#) 

Tagging Recommendation

There are several version- and edition-specific tags. It is recommended to use the [`sql-server`](#) tag together with the version- and/or edition-specific tag; for example, [`sql-server-2005`](#) or [`sql-server-2014-express`](#). Do not use this tag for other types of DBMS ([`mysql`](#), [`postgresql`](#), etc.).

Questions

[Q: How do I split a string so I can access item x?](#)

Tags: [sql-server](#) ([Next Q](#)), [sql-server-2005](#) ([Next Q](#))

Using SQL Server 2005, how do I split a string so I can access item x?

For example

Take the string "Hello John Smith". How can I split the string by a space and access the item at Index 1 which should return "John"?

Tags: [sql-server](#) ([Next Q](#)), [sql-server-2005](#) ([Next Q](#))

User: [gatekiller](#)

[Answer](#) by [jonesinator](#)

You may find the solution in [SQL User Defined Function to Parse a Delimited String](#) helpful (from [The Code Project](#)).

You can use this simple logic:

[Skip code block](#)

```
Declare @products varchar(200) = '1|20|3|343|44|6|8765'
Declare @individual varchar(20) = null

WHILE LEN(@products) > 0
BEGIN
    IF PATINDEX('%|%',@products) > 0
    BEGIN
        SET @individual = SUBSTRING(@products, 0, PATINDEX('%|%',@products))
        SELECT @individual

        SET @products = SUBSTRING(@products, LEN(@individual + '|') + 1,
                                  LEN(@products))
    END
    ELSE
    BEGIN
        SET @individual = @products
        SET @products = NULL
        SELECT @individual
    END
END
END
```

[Answer](#) by [nathan-bedford](#)

I don't believe SQL Server has a built-in split function, so other than a UDF, the only other answer I know is to hijack the PARSENAME function:

```
SELECT PARSENAME(REPLACE('Hello John Smith', ' ', '.'), 2)
```

PARSENAME takes a string and splits it on the period character. It takes a number as its second argument, and that number specifies which segment of the string to return (working from back to front).

```
SELECT PARSENAME(REPLACE('Hello John Smith', ' ', '.'), 3) --return Hello
```

Obvious problem is when the string already contains a period. I still think using a UDF is

the best way...any other suggestions?

[Answer](#) by [vzczc](#)

First, create a function (using CTE, common table expression does away with the need for a temp table)

[Skip code block](#)

```
create function dbo.SplitString
(
    @str nvarchar(4000),
    @separator char(1)
)
returns table
AS
return (
    with tokens(p, a, b) AS (
        select
            1,
            1,
            charindex(@separator, @str)
        union all
        select
            p + 1,
            b + 1,
            charindex(@separator, @str, b + 1)
        from tokens
        where b > 0
    )
    select
        p-1 zeroBasedOccurance,
        substring(
            @str,
            a,
            case when b > 0 then b-a ELSE 4000 end)
        AS s
    from tokens
)
GO
```

Then, use it as any table (or modify it to fit within your existing stored proc) like this.

```
select s
from dbo.SplitString('Hello John Smith', ' ')
where zeroBasedOccurrence=1
```

Update

Previous version would fail for input string longer than 4000 chars. This version takes care of the limitation:

[Skip code block](#)

```
create function dbo.SplitString
(
    @str nvarchar(max),
    @separator char(1)
)
returns table
AS
return (
    with tokens(p, a, b) AS (
        select
            cast(1 as bigint),
            cast(1 as bigint),
            charindex(@separator, @str)
        union all
        select
            p + 1,
            b + 1,
            charindex(@separator, @str, b + 1)
    )
    select
        p-1 zeroBasedOccurance,
        substring(
            @str,
            a,
            case when b > 0 then b-a ELSE maxlen end)
        AS s
    from tokens
)
```

```

        from tokens
        where b > 0
)
select
    p-1 ItemIndex,
    substring(
        @str,
        a,
        case when b > 0 then b-a ELSE LEN(@str) end)
    AS Item
from tokens
);
GO

```

Usage remains the same.

Tags: [sql-server](#) ([Next Q](#)), [sql-server-2005](#) ([Next Q](#))

[Q: How to create a SQL Server function to “join” multiple rows from a subquery into a single delimited field?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

To illustrate, assume that I have two tables as follows:

[Skip code block](#)

VehicleID	Name	
1	Chuck	
2	Larry	
LocationID	VehicleID	City
1	1	New York
2	1	Seattle
3	1	Vancouver
4	2	Los Angeles
5	2	Houston

I want to write a query to return the following results:

VehicleID	Name	Locations
1	Chuck	New York, Seattle, Vancouver
2	Larry	Los Angeles, Houston

I know that this can be done using server side cursors, ie:

[Skip code block](#)

```

DECLARE @VehicleID int
DECLARE @VehicleName varchar(100)
DECLARE @LocationCity varchar(100)
DECLARE @Locations varchar(4000)
DECLARE @Results TABLE
(
    VehicleID int
    Name varchar(100)
    Locations varchar(4000)
)

DECLARE VehiclesCursor CURSOR FOR
SELECT
    [VehicleID]
    ,[Name]
FROM [Vehicles]

OPEN VehiclesCursor

FETCH NEXT FROM VehiclesCursor INTO

```

```

@VehicleID
, @VehicleName
WHILE @@FETCH_STATUS = 0
BEGIN

    SET @Locations = ''

    DECLARE LocationsCursor CURSOR FOR
    SELECT
        [City]
    FROM [Locations]
    WHERE [VehicleID] = @VehicleID

    OPEN LocationsCursor

    FETCH NEXT FROM LocationsCursor INTO
        @LocationCity
    WHILE @@FETCH_STATUS = 0
    BEGIN
        SET @Locations = @Locations + @LocationCity

        FETCH NEXT FROM LocationsCursor INTO
            @LocationCity
    END
    CLOSE LocationsCursor
    DEALLOCATE LocationsCursor

    INSERT INTO @Results (VehicleID, Name, Locations) SELECT @VehicleID, @Name, @Locations
END
CLOSE VehiclesCursor
DEALLOCATE VehiclesCursor

SELECT * FROM @Results

```

However, as you can see, this requires a great deal of code. What I would like is a generic function that would allow me to do something like this:

```

SELECT VehicleID
, Name
, JOIN(SELECT City FROM Locations WHERE VehicleID = Vehicles.VehicleID, ', ') AS Locations
FROM Vehicles

```

Is this possible? Or something similar?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [templar](#) 

[Answer](#)  by [mun](#) 

If you're using SQL Server 2005, you could use the FOR XML PATH command.

```

SELECT [VehicleID]
, [Name]
, (STUFF((SELECT CAST(', ' + [City] AS VARCHAR(MAX))
    FROM [Location]
    WHERE (VehicleID = Vehicle.VehicleID)
    FOR XML PATH ('')), 1, 2, '')) AS Locations
FROM [Vehicle]

```

It's a lot easier than using a cursor, and seems to work fairly well.

[Answer](#)  by [mike-powell](#) 

Note that [Matt's code](#) will result in an extra comma at the end of the string; using COALESCE (or ISNULL for that matter) as shown in the link in Lance's post uses a similar method but doesn't leave you with an extra comma to remove. For the sake of

completeness, here's the relevant code from Lance's link on sqlteam.com:

```
DECLARE @EmployeeList varchar(100)
SELECT @EmployeeList = COALESCE(@EmployeeList + ', ', '') +
    CAST(EmpUniqueID AS varchar(5))
FROM SalesCallsEmployees
WHERE SalCal_Uuid = 1
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: What are the pros and cons to keeping SQL in Stored Procs versus Code](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Next Q](#))

What are the advantages/disadvantages of keeping SQL in your C# source code or in Stored Procs? I've been discussing this with a friend on an open source project that we're working on (C# ASP.NET Forum). At the moment, most of the database access is done by building the SQL inline in C# and calling to the SQL Server DB. So I'm trying to establish which, for this particular project, would be best.

So far I have:

Advantages for in Code:

- Easier to maintain - don't need to run a SQL script to update queries
- Easier to port to another DB - no procs to port

Advantages for Stored Procs:

- Performance
- Security

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Next Q](#))

User: [guy](#)

[Answer](#) by [orion-edwards](#)

I am not a fan of stored procedures

Stored Procedures are MORE maintainable because: * You don't have to recompile your C# app whenever you want to change some SQL

You'll end up recompiling it anyway when datatypes change, or you want to return an extra column, or whatever. The number of times you can 'transparently' change the SQL out from underneath your app is pretty small on the whole

- You end up reusing SQL code.

Programming languages, C# included, have this amazing thing, called a function. It means

you can invoke the same block of code from multiple places! Amazing! You can then put the re-usable SQL code inside one of these, or if you want to get really high tech, you can use a library which does it for you. I believe they're called Object Relational Mappers, and are pretty common these days.

Code repetition is the worst thing you can do when you're trying to build a maintainable application!

Agreed, which is why storedprocs are a bad thing. It's much easier to refactor and decompose (break into smaller parts) code into functions than SQL into... blocks of SQL?

You have 4 webservers and a bunch of windows apps which use the same SQL code
Now you realized there is a small problem with the SQL code so do you rather.....
change the proc in 1 place or push the code to all the webservers, reinstall all the
desktop apps(clickonce might help) on all the windows boxes

Why are your windows apps connecting directly to a central database? That seems like a
HUGE security hole right there, and bottleneck as it rules out server-side caching.
Shouldn't they be connecting via a web service or similar to your web servers?

So, push 1 new sproc, or 4 new webservers?

In this case it *is* easier to push one new sproc, but in my experience, 95% of 'pushed
changes' affect the code and not the database. If you're pushing 20 things to the
webservers that month, and 1 to the database, you hardly lose much if you instead push 21
things to the webservers, and zero to the database.

More easily code reviewed.

Can you explain how? I don't get this. Particularly seeing as the sprocs probably aren't in
source control, and therefore can't be accessed via web-based SCM browsers and so on.

More cons:

Storedprocs live in the database, which appears to the outside world as a black box.
Simple things like wanting to put them in source control becomes a nightmare.

There's also the issue of sheer effort. It might make sense to break everything down into a
[million tiers](#) if you're trying to justify to your CEO why it just cost them 7 million
dollars to build some forums, but otherwise creating a storedproc for every little thing is
just extra donkeywork for no benefit.

[Answer](#) by [eric-z-beard](#)

This is being discussed on a few other threads here currently. I'm a consistent proponent
of stored procedures, although some good arguments for Linq to Sql are being presented.

Embedding queries in your code couples you tightly to your data model. Stored
procedures are a good form of contractual programming, meaning that a DBA has the

freedom to alter the data model and the code in the procedure, so long as the contract represented by the stored procedure's inputs and outputs is maintained.

Tuning production databases can be extremely difficult when the queries are buried in the code and not in one central, easy to manage location.

[Edit] Here is another [current discussion](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Next Q](#))

[Q: How can I remove duplicate rows?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Next Q](#))

What is the best way to remove duplicate rows from a fairly large table (i.e. 300,000+ rows)?

The rows of course will not be perfect duplicates because of the existence of the RowID identity field.

```
MyTable
-----
RowID int not null identity(1,1) primary key,
Col1 varchar(20) not null,
Col2 varchar(2048) not null,
Col3 tinyint not null
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Next Q](#))

User: [seibar](#)

[Answer](#) by [mark-brackett](#)

Assuming no nulls, you GROUP BY the unique columns, and SELECT the MIN (or MAX) RowId as the row to keep. Then, just delete everything that didn't have a row id:

[Skip code block](#)

```
DELETE MyTable
FROM MyTable
LEFT OUTER JOIN (
    SELECT MIN(RowId) as RowId, Col1, Col2, Col3
    FROM MyTable
    GROUP BY Col1, Col2, Col3
) as KeepRows ON
    MyTable.RowId = KeepRows.RowId
WHERE
    KeepRows.RowId IS NULL
```

In case you have a GUID instead of an integer, you can replace

```
MIN(RowId)
```

with

```
CONVERT(uniqueidentifier, MIN(CONVERT(char(36), MyGuidColumn)))
```

[Answer](#) by [martin-smith](#)

Another possible way of doing this is

```

;WITH cte
    AS (SELECT ROW_NUMBER() OVER (PARTITION BY Col1, Col2, Col3
                                ORDER BY ( SELECT 0)) RN
        FROM    #MyTable)
DELETE FROM cte
WHERE   RN > 1

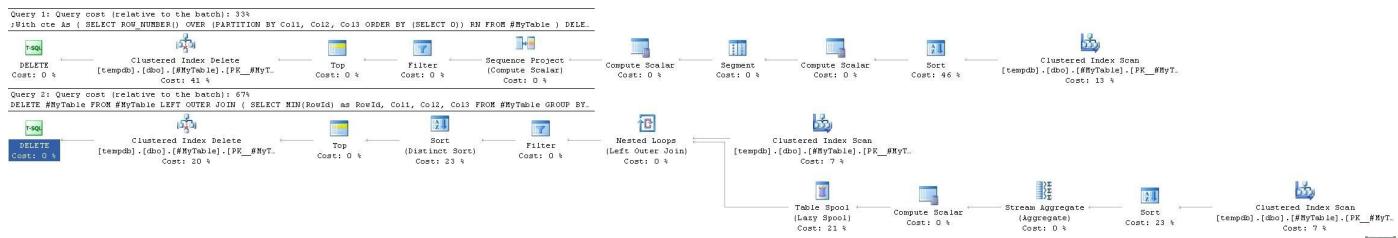
```

I am using ORDER BY (SELECT 0) above as it is arbitrary which row to preserve in the event of a tie.

To preserve the latest one in RowID order for example you could use ORDER BY RowID DESC

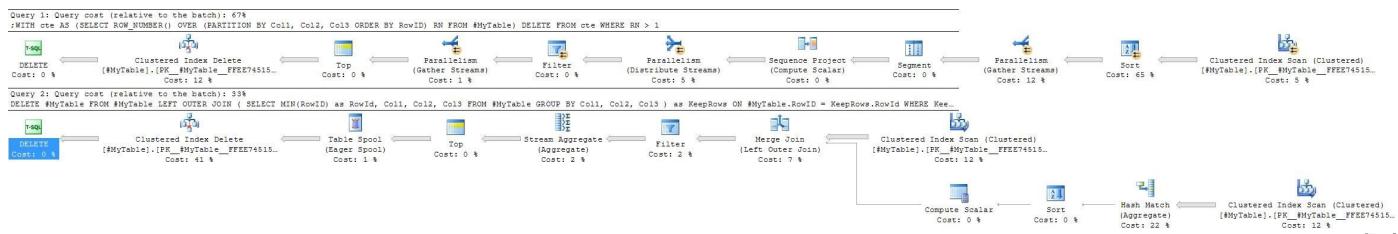
Execution Plans

The execution plan for this is often simpler and more efficient than that in the accepted answer as it does not require the self join.



This is not always the case however. One place where the GROUP BY solution might be preferred is situations where a [hash aggregate](#) would be chosen in preference to a stream aggregate.

The ROW_NUMBER solution will always give pretty much the same plan whereas the GROUP BY strategy is more flexible.



Factors which might favour the hash aggregate approach would be

- No useful index on the partitioning columns
- relatively fewer groups with relatively more duplicates in each group

In extreme versions of this second case (if there are very few groups with many duplicates in each) one could also consider simply inserting the rows to keep into a new table then TRUNCATE-ing the original and copying them back to minimise logging compared to deleting a very high proportion of the rows.

[Answer](#) by [jon-galloway](#)

There's a good article on [removing duplicates](#) on the Microsoft Support site. It's pretty conservative - they have you do everything in separate steps - but it should work well against large tables.

I've used self-joins to do this in the past, although it could probably be prettied up with a

HAVING clause:

```
DELETE dupes
FROM      MyTable dupes,
          MyTable fullTable
WHERE     dupes.dupField      = fullTable.dupField
AND       dupes.secondDupField = fullTable.secondDupField
AND       dupes.uniqueField    > fullTable.uniqueField
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Next Q](#))

[Q: How to insert a line break in a SQL Server VARCHAR/NVARCHAR string](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I didn't see any similar questions asked on this topic, and I had to research this for something I'm working on right now. Thought I would post the answer for it in case anyone else had the same question.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [mark-struzinski](#) 

Answer  by [sören-kuklau](#) 

char(13) is CR. For DOS-/Windows-style CRLF linebreaks, you want char(13)+char(10), like:

```
'This is line 1.' + CHAR(13)+CHAR(10) + 'This is line 2.'
```

Answer  by [mark-struzinski](#) 

I found the answer here: <http://blog.sqlauthority.com/2007/08/22/sql-server-t-sql-script-to-insert-carriage-return-and-new-line-feed-in-code/> 

You just concatenate the string and insert a CHAR(13) where you want your line break.

Example:

```
DECLARE @text NVARCHAR(100)
SET @text = 'This is line 1.' + CHAR(13) + 'This is line 2.'
SELECT @text
```

This prints out the following:

This is line 1.
This is line 2.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: What are the main performance differences between varchar and nvarchar SQL Server data types?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I'm working on a database for a small web app at my school using SQL Server 2005. I see a couple of schools of thought on the issue of varchar vs nvarchar:

1. Use varchar unless you deal with a lot of internationalized data, then use nvarchar.
2. Just use nvarchar for everything.

I'm beginning to see the merits of view 2. I know that nvarchar does take up twice as much space, but that isn't necessarily a huge deal since this is only going to store data for a few hundred students. To me it seems like it would be easiest not to worry about it and just allow everything to use nvarchar. Or is there something I'm missing?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [jason-baker](#) 

[Answer](#)  by [joe-barone](#) 

Always use nvarchar.

You may never need the double-byte characters for most applications. However, if you need to support double-byte languages and you only have single-byte support in your database schema it's really expensive to go back and modify throughout your application.

The cost of migrating one application from varchar to nvarchar will be much more than the little bit of extra disk space you'll use in most applications.

[Answer](#)  by [gbn](#) 

Disk space is not the issue... but memory and performance will be. Double the page reads, double index size, strange LIKE and = constant behaviour etc

Do you need to store Chinese etc script? Yes or no...

And from MS BOL "[Storage and Performance Effects of Unicode](#)" 

Edit:

Recent SO question highlighting how bad nvarchar performance can be...

[SQL Server uses high CPU when searching inside nvarchar strings](#) 

[Answer](#)  by [thomas-harlan](#) 

Be consistent! JOIN-ing a VARCHAR to NVARCHAR has a big performance hit.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

Q: Best way to get identity of inserted row? 

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

What is the best way to get identity of inserted row?

I know about @@IDENTITY and IDENT_CURRENT and SCOPE_IDENTITY but don't understand the pros and cons attached to each.

Can someone please explain the differences and when I should be using each?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [oded](#) 

[Answer](#)  by [bdukes](#) 

- [@@IDENTITY](#)  returns the last identity value generated for any table in the current session, across all scopes. **You need to be careful here**, since it's across scopes. You could get a value from a trigger, instead of your current statement.
- [SCOPE_IDENTITY](#)  returns the last identity value generated for any table in the current session and the current scope. **Generally what you want to use**.
- [IDENT_CURRENT](#)  returns the last identity value generated for a specific table in any session and any scope. This lets you specify which table you want the value from, in case the two above aren't quite what you need (**very rare**). Also, as [@Guy Starbuck](#) mentioned, "You could use this if you want to get the current IDENTITY value for a table that you have not inserted a record into."
- The [OUTPUT clause](#)  of the `INSERT` statement will let you access every row that was inserted via that statement. Since it's scoped to the specific statement, it's **more straightforward** than the other functions above. However, it's a little **more verbose** (you'll need to insert into a table variable/temp table and then query that) and it gives results even in an error scenario where the statement is rolled back. That said, if your query uses a parallel execution plan, this is the **only guaranteed method** for getting the identity (short of turning off parallelism). However, it is executed *before* triggers and cannot be used to return trigger-generated values.

[Answer](#)  by [orry](#) 

I believe the safest and most accurate method of retrieving the inserted id would be using the output clause.

for example (taken from the following [MSDN](#)  article)

[Skip code block](#)

```
USE AdventureWorks2008R2;
GO
DECLARE @MyTableVar table( NewScrapReasonID smallint,
                           Name varchar(50),
                           ModifiedDate datetime);
INSERT Production.ScrapReason
    OUTPUT INSERTED.ScrapReasonID, INSERTED.Name, INSERTED.ModifiedDate
    INTO @MyTableVar
VALUES (N'Operator error', GETDATE());

--Display the result set of the table variable.
SELECT NewScrapReasonID, Name, ModifiedDate FROM @MyTableVar;
--Display the result set of the table.
SELECT ScrapReasonID, Name, ModifiedDate
```

```
FROM Production.ScrapReason;
GO
```

[Answer](#) by [orion-edwards](#)

I'm saying the same thing as the other guys, so everyone's correct, I'm just trying to make it more clear.

`@@IDENTITY` returns the id of the last thing that was inserted by your client's connection to the database.

Most of the time this works fine, but sometimes a trigger will go and insert a new row that you don't know about, and you'll get the ID from this new row, instead of the one you want

`SCOPE_IDENTITY()` solves this problem. It returns the id of the last thing that *you inserted* in the SQL code *you sent* to the database. If triggers go and create extra rows, they won't cause the wrong value to get returned. Hooray

`IDENT_CURRENT` returns the last ID that was inserted by anyone. If some other app happens to insert another row at an unfortunate time, you'll get the ID of that row instead of your one.

If you want to play it safe, always use `SCOPE_IDENTITY()`. If you stick with `@@IDENTITY` and someone decides to add a trigger later on, all your code will break.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Truncate \(not round\) decimal places in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I'm trying to determine the best way to truncate or drop extra decimal places in SQL without rounding. For example:

```
declare @value decimal(18,2)
set @value = 123.456
```

This will auto round `@Value` to be 123.46....which in most cases is good. However, for this project I don't need that. Is there a simple way to truncate the decimals I don't need? I know I can use the `left()` function and convert back to a decimal...any other ways?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [ryan-eastabrook](#)

[Answer](#) by [jimmy](#)

```
select round(123.456, 2, 1)
```

[Answer](#) by [jeff-cuscutis](#)

```
ROUND ( 123.456 , 2 , 1 )
```

When the third parameter != 0 it truncates rather than rounds

[http://msdn.microsoft.com/en-us/library/ms175003\(SQL.90\).aspx](http://msdn.microsoft.com/en-us/library/ms175003(SQL.90).aspx) 

Syntax

ROUND (numeric_expression , length [,function])

Arguments

- *numeric_expression* Is an expression of the exact numeric or approximate numeric data type category, except for the bit data type.
- *length* Is the precision to which numeric_expression is to be rounded. length must be an expression of type tinyint, smallint, or int. When length is a positive number, numeric_expression is rounded to the number of decimal positions specified by length. When length is a negative number, numeric_expression is rounded on the left side of the decimal point, as specified by length.
- *function* Is the type of operation to perform. function must be tinyint, smallint, or int. When function is omitted or has a value of 0 (default), numeric_expression is rounded. When a value other than 0 is specified, numeric_expression is truncated.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Q: How do I create a foreign key in SQL Server?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I have never “hand-coded” object creation code for SQL Server and foreign key declaration is seemingly different between SQL Server and Postgres. Here is my sql so far:

[Skip code block](#)

```
drop table exams;
drop table question_bank;
drop table anwser_bank;

create table exams
(
    exam_id uniqueidentifier primary key,
    exam_name varchar(50),
);
create table question_bank
(
    question_id uniqueidentifier primary key,
    question_exam_id uniqueidentifier not null,
    question_text varchar(1024) not null,
    question_point_value decimal,
    constraint question_exam_id foreign key references exams(exam_id)
);
create table anwser_bank
(
    anwser_id      uniqueidentifier primary key,
    anwser_question_id uniqueidentifier,
    anwser_text    varchar(1024),
    anwser_is_correct bit
);
```

When I run the query I get this error:

Msg 8139, Level 16, State 0, Line 9 Number of referencing columns in foreign key differs from number of referenced columns, table 'question_bank'.

Can you spot the error?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [mmattax](#) 

[Answer](#)  by [john-boker](#) 

```
create table question_bank
(
    question_id uniqueidentifier primary key,
    question_exam_id uniqueidentifier not null,
    question_text varchar(1024) not null,
    question_point_value decimal,
    foreign key fk_questionbank_exams ( question_exam_id ) references exams ( exam_id )
);
```

[Answer](#)  by [alexcuse](#) 

And if you just want to create the constraint on its own, you can use ALTER TABLE

```
alter table MyTable
add constraint MyTable_MyColumn_FK FOREIGN KEY ( MyColumn ) references MyOtherTable(PKColumn)
```

I wouldn't recommend the syntax mentioned by Sara Chipps for inline creation, just because I would rather name my own constraints.

[Answer](#)  by [sara-chipps](#) 

You can also name your foreign key constraint by using:

```
CONSTRAINT your_name_here FOREIGN KEY ( question_exam_id ) REFERENCES EXAMS ( exam_id )
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Why would I ever pick CHAR over VARCHAR in SQL?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I realize that CHAR is recommended if all my values are fixed-width. But, so what? Why not just pick VARCHAR for all text fields just to be safe.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [skunkspinner](#) 

[Answer](#)  by [jim-mckeeth](#) 

Generally pick **CHAR** if all rows will have close to the *same length*. Pick **VARCHAR** when the *length varies* significantly. CHAR may also be a bit faster because all the rows are of the same length.

It varies by DB implementation, but generally VARCHAR uses one or two more bytes of

storage (for length or termination) in addition to the actual data. So (assuming you are using a one byte character set) storing the word “FooBar”

- CHAR(6) = 6 bytes (no overhead)
- VARCHAR(10) = 8 bytes (2 bytes of overhead)
- CHAR(10) = 10 bytes (4 bytes of overhead)

Bottom line is **CHAR** can be *faster* and more *space efficient* for data of relatively the same length (within two characters length difference).

Note: Microsoft SQL has 2 bytes of overhead for a VARCHAR. This may vary from DB to DB, but generally there is at least 1 byte of overhead needed to indicate length or EOL on a VARCHAR.

As was pointed out by Gaven in the comments, if you are using a multi-byte, variable length character set like UTF8 then CHAR stores the maximum number of bytes necessary to store the number of characters. So if UTF8 needs at most 3 bytes to store a character, then CHAR(6) will be fixed at 18 bytes, even if only storing latin1 characters. So in this case VARCHAR becomes a much better choice.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Q: How to perform an IF...THEN in an SQL SELECT?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [case](#) ([Next Q](#))

How do I perform an IF...THEN in an SQL SELECT statement?

For example:

```
SELECT IF(Obsolete = 'N' OR InStock = 'Y' ? 1 : 0) AS Saleable, * FROM Product
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [case](#) ([Next Q](#))

User: [eric-labashosky](#)

[Answer](#) by [darrel-miller](#)

The CASE statement is the closest to IF in SQL and is supported on all versions of SQL Server

```
SELECT CAST(
    CASE
        WHEN Obsolete = 'N' or InStock = 'Y'
            THEN 1
        ELSE 0
    END AS bit) as Saleable, *
FROM Product
```

You only need to do the CAST if you want the result as a boolean value, if you are happy with an int, this works:

```
SELECT CASE
    WHEN Obsolete = 'N' or InStock = 'Y'
        THEN 1
    ELSE 0
END as Saleable, *
FROM Product
```

CASE statements can be embedded in other CASE statements and even included in aggregates.

SQL Server Denali (SQL Server 2012) adds the [IIF](#) statement which is also available in [access](#): (pointed out by [Martin Smith](#))

```
SELECT IIF(Obsolete = 'N' or InStock = 'Y', 1, 0) as Selable, * from Product
```

[Answer](#) by [jonathan](#)

The case statement is your friend in this situation, and takes one of two forms:

The simple case:

```
SELECT CASE <variable> WHEN <value>      THEN <returnvalue>
                           WHEN <othervalue> THEN <returnthis>
                           ELSE <returndefaultcase>
    END AS <newcolumnname>
FROM <table>
```

The extended case:

```
SELECT CASE WHEN <test>      THEN <returnvalue>
               WHEN <othertest> THEN <returnthis>
               ELSE <returndefaultcase>
```

```
END AS <newcolumnname>
FROM <table>
```

You can even put case statements in an order by clause for really fancy ordering.

[Answer](#)  by [martin-smith](#) 

From SQL Server 2012 you can use the [IIF function](#)  for this.

```
SELECT IIF(Obsolete = 'N' OR InStock = 'Y', 1, 0) AS Salable, *
FROM Product
```

This is effectively just a shorthand (albeit not standard SQL) way of writing CASE.

I prefer the conciseness when compared with the expanded CASE version.

Both IIF() and CASE resolve as expressions within a SQL Statement and can only be used in well defined places.

The CASE expression cannot be used to control the flow of execution of Transact-SQL statements, statement blocks, user-defined functions, and stored procedures.

If your needs can not be satisfied by these limitations (for example a need to return differently shaped result sets dependant on some condition) then SQL Server does also have a procedural [IF](#)  Keyword.

[Skip code block](#)

```
IF @IncludeExtendedInformation = 1
BEGIN
    SELECT A,B,C,X,Y,Z
    FROM T
END
ELSE
BEGIN
    SELECT A,B,C
    FROM T
END
```

[Care must sometimes be taken to avoid parameter sniffing issues with this approach however.](#) 

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [case](#) ([Next Q](#))

[Q: SQL Server 2005 How Create a Unique Constraint?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

How do I create a unique constraint on an existing table in SQL Server 2005?

I am looking for both the TSQL and how to do it in the Database Diagram.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [david-basarab](#) 

[Answer](#)  by [rory](#) 

The SQL command is:

```
ALTER TABLE <tablename> ADD CONSTRAINT  
    <constraintname> UNIQUE NONCLUSTERED  
(  
    <columnname>  
)
```

See the full syntax [here](#) 

If you want to do it from a Database Diagram:

- right-click on the table and select ‘Indexes/Keys’
- click the Add button to add a new index
- enter the necessary info in the Properties on the right hand side:
 - the columns you want (click the ellipsis button to select)
 - set Is Unique to Yes
 - give it an appropriate name

[Answer](#)  by [james-lawruk](#) 

In SQL Server Management Studio Express:

- Right-click table, choose **Modify or Design(For Later Versions)**
- Right-click field, choose **Indexes/Keys...**
- Click **Add**
- For **Columns**, select the **field name** you want to be unique.
- For **Type**, choose **Unique Key**.
- Click **Close, Save** the table.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL MAX of multiple columns?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

How do you return 1 value per row of the max of several columns:

TableName

```
[Number, Date1, Date2, Date3, Cost]
```

I need to return something like this:

```
[Number, Most_Recent_Date, Cost]
```

Query?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [benb](#) 

[Answer](#)  by [lasse-v.-karlsen](#) 

Well, you can use the CASE statement:

```
SELECT
  CASE
    WHEN Date1 >= Date2 AND Date1 >= Date3 THEN Date1
    WHEN Date2 >= Date1 AND Date2 >= Date3 THEN Date2
    WHEN Date3 >= Date1 AND Date3 >= Date2 THEN Date3
    ELSE Date1
  END AS MostRecentDate
```

[Answer](#) by [sven](#)

Here is another nice solution for the Max functionality using T-SQL and SQL Server

```
SELECT [Other Fields],
  (SELECT Max(v)
   FROM (VALUES (date1), (date2), (date3),...) AS value(v)) as [MaxDate]
FROM [YourTableName]
```

[Answer](#) by [bajafresh4life](#)

If you're using MySQL, you can use

```
SELECT GREATEST(col1, col2...) FROM table
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to convert DateTime to VarChar](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Next Q](#))

I am working on a query in Sql Server 2005 where I need to convert a value in DateTime variable into a varchar variable in yyyy-mm-dd format (without time part). How do I do that?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Next Q](#))

User: [ali](#)

[Answer](#) by [tonyossa](#)

With Microsoft Sql Server:

[Skip code block](#)

```
--Create test case--DECLARE @myDateTime DATETIME
SET @myDateTime = '2008-05-03'
--- Convert string--SELECT LEFT(CONVERT(VARCHAR, @myDateTime, 120), 10)
```

[Answer](#) by [joel-coehoorn](#)

Try the following:

```
CONVERT(varchar(10), [MyDateTimecolumn], 20)
```

For a full date time and not just date do:

```
CONVERT(varchar(23), [MyDateTimecolumn], 121)
```

See this page for convert styles:

<http://msdn.microsoft.com/en-us/library/ms187928.aspx>

OR

[SQL Server CONVERT\(\) Function](#)

[Answer](#) by [colin](#)

Here's some test sql for all the styles. Make nvarchar(max) shorter to trim (e.g. nvarchar(10)).

[Skip code block](#)

```
DECLARE @now datetime
SET @now = GETDATE()
select convert(nvarchar(MAX), @now, 0) as output, 0 as style
union select convert(nvarchar(MAX), @now, 1), 1
union select convert(nvarchar(MAX), @now, 2), 2
union select convert(nvarchar(MAX), @now, 3), 3
union select convert(nvarchar(MAX), @now, 4), 4
union select convert(nvarchar(MAX), @now, 5), 5
union select convert(nvarchar(MAX), @now, 6), 6
union select convert(nvarchar(MAX), @now, 7), 7
union select convert(nvarchar(MAX), @now, 8), 8
union select convert(nvarchar(MAX), @now, 9), 9
union select convert(nvarchar(MAX), @now, 10), 10
union select convert(nvarchar(MAX), @now, 11), 11
union select convert(nvarchar(MAX), @now, 12), 12
union select convert(nvarchar(MAX), @now, 13), 13
union select convert(nvarchar(MAX), @now, 14), 14
--15 to 19 not valid
union select convert(nvarchar(MAX), @now, 20), 20
union select convert(nvarchar(MAX), @now, 21), 21
union select convert(nvarchar(MAX), @now, 22), 22
union select convert(nvarchar(MAX), @now, 23), 23
union select convert(nvarchar(MAX), @now, 24), 24
union select convert(nvarchar(MAX), @now, 25), 25
--26 not valid
union select convert(nvarchar(MAX), @now, 100), 100
union select convert(nvarchar(MAX), @now, 101), 101
union select convert(nvarchar(MAX), @now, 102), 102
union select convert(nvarchar(MAX), @now, 103), 103
union select convert(nvarchar(MAX), @now, 104), 104
union select convert(nvarchar(MAX), @now, 105), 105
union select convert(nvarchar(MAX), @now, 106), 106
union select convert(nvarchar(MAX), @now, 107), 107
union select convert(nvarchar(MAX), @now, 108), 108
union select convert(nvarchar(MAX), @now, 109), 109
union select convert(nvarchar(MAX), @now, 110), 110
union select convert(nvarchar(MAX), @now, 111), 111
union select convert(nvarchar(MAX), @now, 112), 112
union select convert(nvarchar(MAX), @now, 113), 113
union select convert(nvarchar(MAX), @now, 114), 114
union select convert(nvarchar(MAX), @now, 120), 120
union select convert(nvarchar(MAX), @now, 121), 121
--122 to 125 not valid
union select convert(nvarchar(MAX), @now, 126), 126
union select convert(nvarchar(MAX), @now, 127), 127
--128, 129 not valid
union select convert(nvarchar(MAX), @now, 130), 130
union select convert(nvarchar(MAX), @now, 131), 131
--132 not valid
order BY style
```

Here's the result

[Skip code block](#)

output	style
Apr 28 2014 9:31AM	0
04/28/14	1
14.04.28	2
28/04/14	3
28.04.14	4

28-04-14	5
28 Apr 14	6
Apr 28, 14	7
09:31:28	8
Apr 28 2014 9:31:28:580AM	9
04-28-14	10
14/04/28	11
140428	12
28 Apr 2014 09:31:28:580	13
09:31:28:580	14
2014-04-28 09:31:28	20
2014-04-28 09:31:28.580	21
04/28/14 9:31:28 AM	22
2014-04-28	23
09:31:28	24
2014-04-28 09:31:28.580	25
Apr 28 2014 9:31AM	100
04/28/2014	101
2014.04.28	102
28/04/2014	103
28.04.2014	104
28-04-2014	105
28 Apr 2014	106
Apr 28, 2014	107
09:31:28	108
Apr 28 2014 9:31:28:580AM	109
04-28-2014	110
2014/04/28	111
20140428	112
28 Apr 2014 09:31:28:580	113
09:31:28:580	114
2014-04-28 09:31:28	120
2014-04-28 09:31:28.580	121
2014-04-28T09:31:28.580	126
2014-04-28T09:31:28.580	127
28 9:31:28:580 1435 الثانية 1435 ١٤٣٥ ٩:٣١:٢٨:٥٨٠AM جمادى ١٣٠	130
28/06/1435 9:31:28:580AM	131

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Next Q](#))

[Q: SQL: IF clause within WHERE clause](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Is it possible to use an **IF** clause within a **WHERE** clause in MS SQL?

Example:

```
WHERE
IF IsNumeric(@OrderNumber) = 1
    OrderNumber = @OrderNumber
ELSE
    OrderNumber LIKE '%' + @OrderNumber + '%'
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [bryan-roth](#) 

[Answer](#)  by [bdukes](#) 

Use a [CASE](#)  statement

UPDATE: The previous syntax (as pointed out by a few people) doesn't work. You can use CASE as follows:

```
WHERE OrderNumber LIKE
CASE WHEN IsNumeric(@OrderNumber) = 1 THEN
```

```
@OrderNumber  
ELSE  
  '%' + @OrderNumber  
END
```

Or you can use an IF statement like [@N. J. Reed](#) points out.

[Answer](#) by [njr101](#)

You should be able to do this without any IF or CASE

```
WHERE  
  (IsNumeric(@OrderNumber) AND  
    (CAST OrderNumber AS VARCHAR) = (CAST @OrderNumber AS VARCHAR))  
OR  
  (NOT IsNumeric(@OrderNumber) AND  
    OrderNumber LIKE ('%' + @OrderNumber))
```

Depending on the flavour of SQL you may need to tweak the casts on the order number to an INT or VARCHAR depending on whether implicit casts are supported.

This is a very common technique in a WHERE clause. If you want to apply some “IF” logic in the WHERE clause all you need to do is add the extra condition with an boolean AND to the section where it needs to be applied.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Add a column, with a default value, to an existing table in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Next Q](#))

How can a column with a default value be added to an existing table in [SQL Server 2000](#) / [SQL Server 2005](#)?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Next Q](#))

User: [mathias](#)

[Answer](#) by [james-boother](#)

```
ALTER TABLE {TABLENAME}  
ADD {COLUMNNAME} {TYPE} {NULL|NOT NULL}  
CONSTRAINT {CONSTRAINT_NAME} DEFAULT {DEFAULT_VALUE}  
[WITH VALUES]
```

[Answer](#) by [dbugger](#)

```
ALTER TABLE Protocols  
ADD ProtocolTypeID int NOT NULL DEFAULT(1)  
GO
```

[Answer](#) by [phunk_munkie](#)

WITH VALUES handles the NOT NULL part...

```
ALTER TABLE table  
ADD column BIT
```

```
CONSTRAINT Constraint_name DEFAULT 0 WITH VALUES
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Next Q](#))

Q: Solutions for INSERT OR UPDATE on SQL Server

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

Assume a table structure of MyTable(KEY, datafield1, datafield2...).

Often I want to either update an existing record, or insert a new record if it doesn't exist.

Essentially:

```
IF (key exists)
    run update command
ELSE
    run insert command
```

What's the best performing way to write this?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [chris-cudmore](#) 

[Answer](#)  by [aku](#) 

don't forget about transactions. Performance is good, but simple (IF EXISTS..) approach is very dangerous.

When multiple threads will try to perform Insert-or-update you can easily get primary key violation.

Solutions provided by @Beau Crawford & @Esteban show general idea but error-prone.

To avoid deadlocks and PK violations you can use something like this:

[Skip code block](#)

```
begin tran
if exists (select * from table with (updlock,serializable) where key = @key)
begin
    update table set...
    where key = @key
end
else
begin
    insert into table (key, ...)
    values (@key, ...)
end
commit tran
```

Or

```
begin tran
    update table with (serializable) set...
    where key = @key

    if @@rowcount = 0
    begin
        insert into table (key, ...) values (@key,...)
    end
commit tran
```

[Answer](#) by [keith](#)

See my [detailed answer to a very similar previous question](#)

@Beau Crawford's is a good way in SQL 2005 and below, though if you're granting rep it should go to the [first guy to SO it](#). The only problem is that for inserts it's still two IO operations.

MS Sql2008 introduces merge from the SQL:2003 standard:

[Skip code block](#)

```
merge tablename as target
using (values ('new value', 'different value'))
    as source (field1, field2)
  on target.idfield = 7
when matched then
  update
    set field1 = source.field1,
        field2 = source.field2,
        ...
when not matched then
  insert ( idfield, field1, field2, ... )
  values ( 7, source.field1, source.field2, ... )
```

Now it's really just one IO operation, but awful code :-(

[Answer](#) by [beau-crawford](#)

Do an UPSERT:

```
UPDATE MyTable SET FieldA=@FieldA WHERE Key=@Key
IF @@ROWCOUNT = 0
  INSERT INTO MyTable (FieldA) VALUES (@FieldA)
```

<http://en.wikipedia.org/wiki/Upsert>

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: What is the best way to paginate results in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Next Q](#))

What is the best way (performance wise) to paginate results in SQL Server 2000, 2005, 2008, 2012 if you also want to get the total number of results (before paginating)?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Next Q](#))

User: [panagiotis-korros](#)

[Answer](#) by [mdb](#)

Getting the total number of results and paginating are two different operations. For the sake of this example, let's assume that the query you're dealing with is

```
SELECT * FROM Orders WHERE OrderDate >= '1980-01-01' ORDER BY OrderDate
```

In this case, you would determine the total number of results using:

```
SELECT COUNT(*) FROM Orders WHERE OrderDate >= '1980-01-01'
```

...which may seem inefficient, but is actually pretty performant, assuming all indexes etc. are properly set up.

Next, to get actual results back in a paged fashion, the following query would be most efficient:

```
SELECT *  
FROM ( SELECT ROW_NUMBER() OVER ( ORDER BY OrderDate ) AS RowNum, *  
      FROM Orders  
      WHERE OrderDate >= '1980-01-01'  
    ) AS RowConstrainedResult  
WHERE RowNum >= 1  
  AND RowNum < 20  
ORDER BY RowNum
```

This will return rows 1-19 of the original query. The cool thing here, especially for web apps, is that you don't have to keep any state, except the row numbers to be returned.

[Answer](#) by [jama](#)

I was also very curious about why Microsoft doesn't support simple queries with **offset/limit** like in MySQL or PostgreSQL. Finally, released **Microsoft SQL Server 2012**, I really like its simplicity for pagination, you don't have to use complex queries like answered here.

For getting the next 10 rows just run this query:

```
SELECT * FROM TableName ORDER BY id OFFSET 10 ROWS FETCH NEXT 10 ROWS ONLY;
```

<http://technet.microsoft.com/en-us/library/gg699618.aspx>

Key points to consider when using it:

- ORDER BY is mandatory to use OFFSET and FETCH clause.
- OFFSET clause is mandatory with FETCH. You can never use, ORDER BY ... FETCH.
- TOP cannot be combined with OFFSET and FETCH in the same query expression.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Next Q](#))

Q: How to return the date part only from a SQL Server datetime datatype

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

```
SELECT GETDATE()
```

Returns: 2008-09-22 15:24:13.790

I want that date part without the time part: 2008-09-22 00:00:00.000

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

User: [eddiegroves](#)

[Answer](#) by [aku](#)

On SQL Server 2008 and higher, you should convert to date:

```
SELECT CONVERT(date, getdate())
```

On older versions, you can do the following:

```
SELECT DATEADD(dd, 0, DATEDIFF(dd, 0, @your_date))
```

for example

```
SELECT DATEADD(dd, 0, DATEDIFF(dd, 0, GETDATE()))
```

gives me

```
2008-09-22 00:00:00.000
```

Pros:

- No varchar<->datetime conversions required
- No need to think about locale

[Answer](#) by [benr](#)

SQLServer 2008 now has a Date datatype which contains only a date with no time.

Anyone using SQLServer 2008 and beyond can do the following:

```
select CONVERT(date, getdate())
```

[Answer](#) by [abatishchev](#)

If using SQL 2008 and above:

```
select cast(getdate() as date)
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

Q: Most efficient T-SQL way to pad a varchar on the left to a certain length?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

As compared to say:

```
REPLICATE(@padchar, @len - LEN(@str)) + @str
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [cade-roux](#)

[Answer](#) by [alexciuse](#)

This is simply an inefficient use of SQL, no matter how you do it.

perhaps something like

```
right('XXXXXXXXXXXX'+ rtrim(@str), @n)
```

where X is your padding character and @n is the number of characters in the resulting string (assuming you need the padding because you are dealing with a fixed length).

But as I said you should really avoid doing this in your database.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Q: Is there a Max function in SQL Server that takes two values like Math.Max in .NET?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I want to write a query like this:

```
SELECT o.OrderId, MAX(o.NegotiatedPrice, o.SuggestedPrice)  
FROM Order o
```

But this isn't how the MAX function works, right? It is an aggregate function so it expects a single parameter and then returns the MAX of all rows.

Does anyone know how to do it my way?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [skb](#) 

[Answer](#)  by [kevin-crumley](#) 

You'd need to make a User-Defined Function if you wanted to have syntax similar to your example, but could you do what you want to do, inline, fairly easily with a CASE statement, as the others have said.

The UDF could be something like this:

```
create function dbo.InlineMax(@val1 int, @val2 int)  
returns int  
as  
begin  
    if @val1 > @val2  
        return @val1  
    return isnull(@val2,@val1)  
end
```

... and you would call it like so ...

```
SELECT o.OrderId, dbo.InlineMax(o.NegotiatedPrice, o.SuggestedPrice)  
FROM Order o
```

[Answer](#)  by [miketeeyee](#) 

If you're using SQL Server 2008 (or above), then this is the better solution:

```
SELECT o.OrderId,  
       (SELECT MAX(Price)  
        FROM (VALUES (o.NegotiatedPrice),(o.SuggestedPrice)) AS AllPrices(Price))  
  FROM Order o
```

All credit and votes should go to [Sven's answer to a related question, "SQL MAX of multiple columns?"](#)

I say it's the “*best answer*” because:

1. It doesn't require complicating your code with UNION's, PIVOT's, UNPIVOT's, UDF's, and crazy-long CASE statements.
 2. It isn't plagued with the problem of handling nulls, it handles them just fine.
 3. It's easy to swap out the “MAX” with “MIN”, “AVG”, or “SUM”. You can use any aggregate function to find the aggregate over many different columns.
 4. You're not limited to the names I used (i.e. “AllPrices” and “Price”). You can pick your own names to make it easier to read and understand for the next guy.
 5. You can find multiple aggregates using SQL Server 2008's [derived tables](#) like so:
SELECT MAX(a), MAX(b) FROM (VALUES (1, 2), (3, 4), (5, 6), (7, 8), (9, 10)) AS MyTable(a, b)
-

[Answer](#) by [splattnet](#)

Can be done in one line:

```
-- the following expression calculates ==> max(@val1, @val2)
SELECT 0.5 * (@val1 + @val2) + ABS(@val1 - @val2))
```

Edit: If you're dealing with very large numbers you'll have to convert the value variables into bigint in order to avoid an integer overflow.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: NOT IN clause and NULL values](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

This issue came up when I got different records counts for what I thought were identical queries one using a `not in` where constraint and the other a `left join`. The table in the `not in` constraint had one null value (bad data) which caused that query to return a count of 0 records. I sort of understand why but I could use some help fully grasping the concept.

To state it simply, why does query A return a result but B doesn't?

```
A: select 'true' where 3 in (1, 2, 3, null)
B: select 'true' where 3 not in (1, 2, null)
```

This was on SQL Server 2005. I also found that calling `set ansi_nulls off` causes B to return a result.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [jamie-ide](#)

[Answer](#) by [brannon](#)

Query A is the same as:

```
select 'true' where 3 = 1 or 3 = 2 or 3 = 3 or 3 = null
```

Since `3 = 3` is true, you get a result.

Query B is the same as:

```
select 'true' where 3 <> 1 and 3 <> 2 and 3 <> null
```

When `ansi_nulls` is on, `3 <> null` is UNKNOWN, so the predicate evaluates to UNKNOWN, and you don't get any rows.

When `ansi_nulls` is off, `3 <> null` is true, so the predicate evaluates to true, and you get a row.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Are there any disadvantages to always using nvarchar\(MAX\)?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

In SQL Server 2005, are there any disadvantages to making all character fields `nvarchar(MAX)` rather than specifying a length explicitly, e.g. `nvarchar(255)`? (Apart from the obvious one that you aren't able to limit the field length at the database level)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [stucampbell](#)

[Answer](#) by [david-kreps](#)

Same question was asked on MSDN Forums:

- [Varchar\(max\) vs Varchar\(255\)](#)

From the original post (much more information there):

When you store data to a VARCHAR(N) column, the values are physically stored in the same way. But when you store it to a VARCHAR(MAX) column, behind the screen the data is handled as a TEXT value. So there is some additional processing needed when dealing with a VARCHAR(MAX) value. (only if the size exceeds 8000)

VARCHAR(MAX) or NVARCHAR(MAX) is considered as a ‘large value type’. Large value types are usually stored ‘out of row’. It means that the data row will have a pointer to another location where the ‘large value’ is stored...

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: Convert Month Number to Month Name Function in SQL](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I have months stored in SQL Server as 1,2,3,4,...12. I would like to display them as January,February etc. Is there a function in SQL Server like MonthName(1) = January? I am trying to avoid a CASE statement, if possible.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [saif-khan](#)

[Answer](#) by [alexander-kojevnikov](#)

A little hacky but should work:

```
SELECT DATENAME(month, DATEADD(month, @mydate-1, CAST('2008-01-01' AS datetime)))
```

[Answer](#) by [leoinfo](#)

I think this is the best way to get the **month name** when you have the **month number**

```
Select DateName( month , DateAdd( month , @MonthNumber , 0 ) - 1 )
```

[EDIT]

As Asif said below, another way of doing this would be

```
Select DateName( month , DateAdd( month , @MonthNumber , -1 ) )
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#))

[\(Next Q\)](#)

[Q: Copy tables from one database to another in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I have a database called foo and a database called bar. I have a table in foo called tblFoobar that I want to move (data and all) to database bar from database foo. What is the SQL statement to do this?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [ryankeeter](#) 

[Answer](#)  by [david-b](#) 

On SQL Server? and on the same database server? Use three part naming.

```
INSERT INTO bar..tblFoobar( *fieldlist* )
SELECT *fieldlist* FROM foo..tblFoobar
```

This just moves the data. If you want to move the table definition (and other attributes such as permissions and indexes), you'll have to do something else.

[Answer](#)  by [david](#) 

SQL Server Management Studio's "Import Data" task (right-click on the DB name, then tasks) will do most of this for you. Run it from the database you want to copy the data into.

If the tables don't exist it will create them for you, but you'll probably have to recreate any indexes and such. If the tables do exist, it will append the new data by default but you can adjust that (edit mappings) so it will delete all existing data.

I use this all the time and it works fairly well.

[Answer](#)  by [leoinfo](#) 

This should work:

```
SELECT *
INTO DestinationDB..MyDestinationTable
FROM SourceDB..MySourceTable
```

It will **not** copy constraints, defaults or indexes. The table created will **not** have a clustered index.

Alternatively you could:

```
INSERT INTO DestinationDB..MyDestinationTable
SELECT * FROM SourceDB..MySourceTable
```

If your destination table exists and is empty.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: Row Offset in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Is there any way in SQL Server to get the results starting at a given offset? For example, in another type of SQL database, it's possible to do:

```
SELECT * FROM MyTable OFFSET 50 LIMIT 25
```

to get results 50-74. This construct does not appear to exist in SQL Server.

How can I accomplish this without loading all the rows I don't care about? Thanks!

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [alex](#)

[Answer](#) by [brian-kim](#)

I would avoid using `SELECT *`. Specify columns you actually want even though it may be all of them.

SQL Server 2005+

```
SELECT col1, col2
FROM (
    SELECT col1, col2, ROW_NUMBER() OVER (ORDER BY ID) AS RowNum
    FROM MyTable
) AS MyDerivedTable
WHERE MyDerivedTable.RowNum BETWEEN @startRow AND @endRow
```

SQL Server 2000

[Efficiently Paging Through Large Result Sets in SQL Server 2000](#)

[A More Efficient Method for Paging Through Large Result Sets](#)

[Answer](#) by [martin-smith](#)

If you will be processing all pages in order then simply remembering the last key value seen on the previous page and using `TOP (25) ... WHERE Key > @last_key ORDER BY Key` can be the best performing method if suitable indexes exist to allow this to be seeked efficiently - or [an API cursor](#) if they don't.

For selecting an arbitrary page the best solution for SQL Server 2005 - 2008 R2 is probably `ROW_NUMBER` and `BETWEEN`

For SQL Server 2012+ you can use the enhanced [ORDER BY](#) clause for this need.

```
SELECT *
FROM MyTable
ORDER BY OrderingColumn ASC
OFFSET 50 ROWS
FETCH NEXT 25 ROWS ONLY
```

Though [it remains to be seen how well performing this option will be](#).

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: Effect of NOLOCK hint in SELECT statements](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I guess the real question is:

If I don't care about dirty reads, will adding the **with (NOLOCK)** hint to a SELECT statement affect the performance of:

1. the current SELECT statement
2. other transactions against the given table

Example:

```
Select *  
from aTable with (NOLOCK)
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [bob-probst](#)

[Answer](#) by [tom.dietrich](#)

- 1) Yes, a select with NOLOCK will complete faster than a normal select.
- 2) Yes, a select with NOLOCK will allow other queries against the effected table to complete faster than a normal select.

Why would this be?

NOLOCK typically (depending on your DB engine) means give me your data, and I don't care what state it is in, and don't bother holding it still while you read from it. It is all at once faster, less resource-intensive, and very very dangerous.

You should be warned to never do an update from or perform anything system critical, or where absolute correctness is required using data that originated from a NOLOCK read. It is absolutely possible that this data contains rows that were deleted during the query's run or that have been deleted in other sessions that have yet to be finalized. It is possible that this data includes rows that have been partially updated. It is possible that this data contains records that violate foreign key constraints. It is possible that this data excludes rows that have been added to the table but have yet to be committed.

You really have no way to know what the state of the data is.

If you're trying to get things like a Row Count or other summary data where some margin of error is acceptable, then NOLOCK is a good way to boost performance for these queries and avoid having them negatively impact database performance.

Always use the NOLOCK hint with great caution and treat any data it returns suspiciously.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Q: In SQL, how can you “group by” in ranges?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Suppose I have a table with a numeric column (lets call it “score”).

I’d like to generate a table of counts, that shows how many times scores appeared in each range.

For example:

score range		number of occurrences
0-9		11
10-19		14
20-29		3
...		...

In this example there were 11 rows with scores in the range of 0 to 9, 14 rows with scores in the range of 10 to 19, and 3 rows with scores in the range 20-29.

Is there an easy way to set this up? What do you recommend?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [hugh](#)

[Answer](#) by [ron-tuffin](#)

Neither of the highest voted answers are correct on SQLServer 2000. Perhaps they were using a different version.

Here are the correct versions of both of them on SQLServer 2000.

```
select t.range as [score range], count(*) as [number of occurences]
from (
    select case
        when score between 0 and 9 then '0- 9'
        when score between 10 and 19 then '10-19'
        else '20-99' end as range
    from scores) t
group by t.range
```

or

```
select t.range as [score range], count(*) as [number of occurences]
from (
    select user_id,
        case when score >= 0 and score< 10 then '0-9'
        when score >= 10 and score< 20 then '10-19'
        else '20-99' end as range
    from scores) t
group by t.range
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Q: Create a date with T-SQL

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I am trying to convert a date with individual parts such as 12, 1, 2007 into a datetime in SQL Server 2005. I have tried the following:

```
CAST(DATEPART(year, DATE) + '-' + DATEPART(month, DATE) + '-' + DATEPART(day, DATE) AS DATETIME)
```

but this results in the wrong date. What is the correct way to turn the three date values into a proper datetime format.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [brandon](#) 

[Answer](#)  by [cade-roux](#) 

Assuming y, m, d are all int, how about:

```
CAST(CAST(y AS varchar) + '-' + CAST(m AS varchar) + '-' + CAST(d AS varchar) AS DATETIME)
```

Please see [my other answer](#)  for SQL Server 2012 and above

[Answer](#)  by [charles-bretana](#) 

Try this:

```
Declare @DayOfMonth TinyInt Set @DayOfMonth = 13
Declare @Month TinyInt Set @Month = 6
Declare @Year Integer Set @Year = 2006-----
Select DateAdd(day, @DayOfMonth - 1,
               DateAdd(month, @Month - 1,
                         DateAdd(Year, @Year-1900, 0)))
```

It works as well, has added benefit of not doing any string conversions, so it's pure arithmetic processing (very fast) and it's not dependant on any date format. This capitalizes on the fact that SQL Server's internal representation for datetime and smalldatetime values is a two part value the first part of which is an integer representing the number of days since 1 Jan 1900, and the second part is a decimal fraction representing the fractional portion of one day (for the time) – So the integer value 0 (zero) always translates directly into Midnight morning of 1 Jan 1900...

or, thanks to suggestion from @brinary,

```
Select DateAdd(yy, @Year-1900,
               DateAdd(m, @Month - 1, @DayOfMonth - 1))
```

Edited October 2014. As Noted by @cade Roux, SQL 2012 now has a built-in function: DATEFROMPARTS(year, month, day) that does the same thing.

[Answer](#)  by [cade-roux](#) 

SQL Server 2012 has a wonderful and long-awaited new DATEFROMPARTS function (which will raise an error if the date is invalid - my main objection to a DATEADD-based solution to this problem):

<http://msdn.microsoft.com/en-us/library/hh213228.aspx> 

```
DATEFROMPARTS(ycolumn, mcolumn, dcolumn)
```

or

```
DATEFROMPARTS(@y, @m, @d)
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

Q: How to use GROUP BY to concatenate strings in SQL Server?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [string-concatenation](#) ([Next Q](#))

How do I get:

id	Name	Value
1	A	4
1	B	8
2	C	9

to

id	Column
1	A:4, B:8
2	C:9

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [string-concatenation](#) ([Next Q](#))

User: [eldila](#) 

[Answer](#)  by [kevin-fairchild](#) 

No CURSOR, WHILE loop, or User-Defined Function needed.

Just need to be creative with FOR XML and PATH.

[Note: This solution only works on SQL 2005 and later. Original question didn't specify the version in use.]

[Skip code block](#)

```
CREATE TABLE #YourTable ([ID] INT, [Name] CHAR(1), [Value] INT)

INSERT INTO #YourTable ([ID],[Name],[Value]) VALUES (1,'A',4)
INSERT INTO #YourTable ([ID],[Name],[Value]) VALUES (1,'B',8)
INSERT INTO #YourTable ([ID],[Name],[Value]) VALUES (2,'C',9)

SELECT
    [ID],
    STUFF((
        SELECT ', ' + [Name] + ':' + CAST([Value] AS VARCHAR(MAX))
        FROM #YourTable
        WHERE (ID = Results.ID)
        FOR XML PATH(''),TYPE).value('.//text())[1]', 'VARCHAR(MAX')
    ,1,2,'') AS NameValues
FROM #YourTable Results
GROUP BY ID

DROP TABLE #YourTable
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [string-concatenation](#) ([Next Q](#))

Q: How to deal with SQL column names that look like SQL keywords?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

One of my columns is called `from`. I can't change the name because I didn't make it. Am I allowed to do something like `SELECT from FROM TableName` or is there a special syntax to avoid the SQL Server being confused?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [nathan-h](#) 

[Answer](#)  by [tvanfosson](#) 

Wrap the column name in brackets like so, `from` becomes `[from]`.

```
select [from] from table;
```

It is also possible to use the following (useful when querying multiple tables):

```
select table.[from] from table;
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: What's the best practice for primary keys in tables?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

When designing tables, I've developed a habit of having one column that is unique and that I make the primary key. This is achieved in three ways depending on requirements:

1. Identity integer column that auto increments.
2. Unique identifier (GUID)
3. A short character(x) or integer (or other relatively small numeric type) column that can serve as a row identifier column

Number 3 would be used for fairly small lookup, mostly read tables that might have a unique static length string code, or a numeric value such as a year or other number.

For the most part, all other tables will either have an auto-incrementing integer or unique identifier primary key.

The Question :-)

I have recently started working with databases that have no consistent row identifier and primary keys are currently clustered across various columns. Some examples:

- datetime/character
- datetime/integer
- datetime/varchar
- char/nvarchar/nvarchar

Is there a valid case for this? I would have always defined an identity or unique identifier

column for these cases.

In addition there are many tables without primary keys at all. What are the valid reasons, if any, for this?

I'm trying to understand why tables were designed as they were, and it appears to be a big mess to me, but maybe there were good reasons for it.

EDIT

OK... Wow! A lot of great responses and discussion. I guess I hit on a topic that's a little bit religious without realizing it. :-)

A third question to sort of help me decipher the answers: In cases where multiple columns are used to comprise the compound primary key, is there a specific advantage to this method vs. a surrogate/artificial key? I'm thinking mostly in regards to performance, maintenance, administration, etc.?

EDIT 2

There are lots of good answers here, and it was hard to choose the “best” one, so I’ve chosen one I thought was helpful, but didn’t receive as many votes, and up-voted the others that helped answer my question.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [lloyd-cotten](#) 

[Answer](#)  by [logicalmind](#) 

I follow a few rules:

1. Primary keys should be as small as necessary. Prefer a numeric type because numeric types are stored in a much more compact format than character formats. This is because most primary keys will be foreign keys in another table as well as used in multiple indexes. The smaller your key, the smaller the index, the less pages in the cache you will use.
2. Primary keys should never change. Updating a primary key should always be out of the question. This is because it is most likely to be used in multiple indexes and used as a foreign key. Updating a single primary key could cause of ripple effect of changes.
3. Do NOT use “your problem primary key” as your logic model primary key. For example passport number, social security number, or employee contract number as these “primary key” can change for real world situations.

On surrogate vs natural key, I refer to the rules above. If the natural key is small and will never change it can be used as a primary key. If the natural key is large or likely to change I use surrogate keys. If there is no primary key I still make a surrogate key because experience shows you will always add tables to your schema and wish you’d put a primary key in place.

[Answer](#)  by [tony-andrews](#) 

Natural verses artifical keys is a kind of religious debate among the database community - see [this article](#) and others it links to. I'm neither in favour of **always** having artifical keys, nor of **never** having them. I would decide on a case-by-case basis, for example:

- US States: I'd go for state_code ('TX' for Texas etc.), rather than state_id=1 for Texas
- Employees: I'd usually create an artifical employee_id, because it's hard to find anything else that works. SSN or equivalent may work, but there could be issues like a new joiner who hasn't supplied his/her SSN yet.
- Employee Salary History: (employee_id, start_date). I would **not** create an artifical employee_salary_history_id. What point would it serve (other than "[foolish consistency](#)")

Wherever artificial keys are used, you should always also declare unique constraints on the natural keys. For example, use state_id if you must, but then you'd better declare a unique constraint on state_code, otherwise you are sure to eventually end up with:

state_id	state_code	state_name
137	TX	Texas...
249	TX	Texas		

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: Parameterize an SQL IN clause](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

How do I parameterize a query containing an `IN` clause with a variable number of arguments, like this one?

```
select * from Tags  
where Name in ('ruby', 'rails', 'scruffy', 'rubyonrails')  
order by Count desc
```

In this query, the number of arguments could be anywhere from 1 to 5.

I would prefer not to use a dedicated stored procedure for this (or XML), but if there is some elegant way specific to [SQL Server 2008](#), I am open to that.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [jeff-atwood](#)

[Answer](#) by [joel-spolsky](#)

Here's a quick-and-dirty technique I have used:

```
SELECT *  
FROM Tags  
WHERE '|ruby|rails|scruffy|rubyonrails|'  
    LIKE '%' + Name + '%'
```

So here's the C# code:

```
string[] tags = new string[] { "ruby", "rails", "scruffy", "rubyonrails" };  
const string cmdText = "select * from tags where '|' + @tags + '|' like '%|' + Name + '|%'";  
  
using (SqlCommand cmd = new SqlCommand(cmdText)) {  
    cmd.Parameters.AddWithValue("@tags", string.Join("|", tags));  
}
```

Two caveats:

- The performance is terrible. `like "%...%"` queries are not indexed.
- Make sure you don't have any |, blank, or null tags or this won't work

There are other ways to accomplish this that some people may consider cleaner, so please keep reading.

[Answer](#) by [mark-brackett](#)

You can parameterize *each* value, so something like:

[Skip code block](#)

```
string[] tags = new string[] { "ruby", "rails", "scruffy", "rubyonrails" };  
string cmdText = "SELECT * FROM Tags WHERE Name IN ({0})";  
  
string[] paramNames = tags.Select(  
    (s, i) => "@tag" + i.ToString()  
).ToArray();  
  
string inClause = string.Join(", ", paramNames);  
using (SqlCommand cmd = new SqlCommand(string.Format(cmdText, inClause))) {
```

```

    for(int i = 0; i < paramNames.Length; i++) {
        cmd.Parameters.AddWithValue(paramNames[i], tags[i]);
    }
}

```

Which will give you:

```

cmd.CommandText = "SELECT * FROM Tags WHERE Name IN (@tag0,@tag1,@tag2,@tag3)"
cmd.Parameters["@tag0"] = "ruby"
cmd.Parameters["@tag1"] = "rails"
cmd.Parameters["@tag2"] = "scruffy"
cmd.Parameters["@tag3"] = "rubyonrails"

```

No, this is not open to [SQL injection](#). The only injected text into CommandText is not based on user input. It's solely based on the hardcoded "@tag" prefix, and the index of an array. The index will *always* be an integer, is not user generated, and is safe.

The user inputted values are still stuffed into parameters, so there is no vulnerability there.

Edit:

Injection concerns aside, take care to note that constructing the command text to accomodate a variable number of parameters (as above) impede's SQL server's ability to take advantage of cached queries. The net result is that you almost certainly lose the value of using parameters in the first place (as opposed to merely inserting the predicate strings into the SQL itself).

Not that cached query plans aren't valuable, but IMO this query isn't nearly complicated enough to see much benefit from it. While the compilation costs may approach (or even exceed) the execution costs, you're still talking milliseconds.

If you have enough RAM, I'd expect SQL Server would probably cache a plan for the common counts of parameters as well. I suppose you could always add five parameters, and let the unspecified tags be NULL - the query plan should be the same, but it seems pretty ugly to me and I'm not sure that it'd worth the micro-optimization (although, on Stack Overflow - it may very well be worth it).

Also, SQL Server 7 and later will [auto-parameterize queries](#), so using parameters isn't really necessary from a performance standpoint - it is, however, *critical* from a security standpoint - especially with user inputted data like this.

[Answer](#) by [mark-brackett](#)

For SQL Server 2008, you can use a [table valued parameter](#). It's a bit of work, but it is arguably cleaner than [my other method](#).

First, you have to create a type

```
CREATE TYPE dbo.TagNamesTableType AS TABLE ( Name nvarchar(50) )
```

Then, your ADO.NET code looks like this:

[Skip code block](#)

```

string[] tags = new string[] { "ruby", "rails", "scruffy", "rubyonrails" };
cmd.CommandText = "SELECT Tags.* FROM Tags JOIN @tagNames as P ON Tags.Name = P.Name";
// value must be IEnumerable<SqlDataRecord>
cmd.Parameters.AddWithValue("@tagNames", tags.AsSqlDataRecord("Name")).SqlDbType = SqlDbType.Structure

```

```

cmd.Parameters["@tagNames"].TypeName = "dbo.TagNamesTableType";

// Extension method for converting IEnumarable<string> to IEnumarable<SqlDataRecord>
public static IEnumarable<SqlDataRecord> AsSqlDataRecord(this IEnumarable<string> values, string columnname)
{
    if (values == null || !values.Any()) return null; // Annoying, but SqlClient wants null instead of empty list
    var firstRecord = values.First();
    var metadata = SqlMetaData.InferFromValue(firstRecord, columnname);
    return values.Select(v =>
    {
        var r = new SqlDataRecord(metadata);
        r.SetValues(v);
        return r;
    });
}

```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: Table Naming Dilemma: Singular vs. Plural Names](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Convention has it that table names should be the singular of the entity that they store attributes of.

I dislike any T-SQL that requires square brackets around names, but I have renamed a users table to the singular, forever sentencing those using the table to sometimes have to use brackets.

My gut feel is that it is more correct to stay with the singular, but my gut feel is also that brackets indicate undesirables like column names with spaces in them etc.

Should I stay, or should I go?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [profk](#) 

[Answer](#)  by [nestor](#) 

I had same question, and after reading all answers here I definitely stay with SINGULAR, reasons:

Reason 1 (Concept). You can think of bag containing apples like “AppleBag”, it doesn’t matter if contains 0, 1 or a million apples, it is always the same bag. Tables are just that, containers, the table name must describe what it contains, not how much data it contains. Additionally, the plural concept is more about a spoken language one (actually to determine whether there is one or more), a table is not intended to be read by a human.

Reason 2. (Convenience). it is easier come out with singular names, than with plural ones. Objects can have irregular plurals or not plural at all, but will always have a singular one (with few exceptions like News).

- Customer
- Order
- User
- Status

- News

Reason 3. (Aesthetic and Order). Specially in master-detail scenarios, this reads better, aligns better by name, and have more logical order (Master first, Detail second):

- 1.Order
- 2.OrderDetail

Compared to:

- 1.OrderDetails
- 2.Orders

Reason 4 (Simplicity). Put all together, Table Names, Primary Keys, Relationships, Entity Classes... is better to be aware of only one name (singular) instead of two (singular class, plural table, singular field, singular-plural master-detail...)

- Customer
- Customer.CustomerID
- CustomerAddress
- public Class Customer { ... }
- SELECT FROM Customer WHERE CustomerID = 100

Once you know you dealing with “Customer”, you can be sure you will use the same word for all your database interaction needs.

Reason 5. (Globalization). The world is getting smaller, you may have a team of different nationalities, not everybody has English as native language. Would be easier for a non-native English language programmer to think of “Repository” than of “Repositories”, or avoid them type “Statuses” instead of “Status”. Having singular names can lead to less errors caused by typos, save time by avoid spending extra seconds to think “is it Child or Children?”, hence improving productivity.

Reason 6. (Why not?). It can even save you writing time, save you disk space, and even make your computer keyboard lasts more!

- SELECT Customer.CustomerName FROM Customer WHERE Customer.CustomerID = 100
- SELECT Customers.CustomerName FROM Customers WHERE Customers.CustomerID = 100

You have saved 3 letters, 3 bytes, 3 extra keyboard hits :)

And finally, you can name those ones messing up with reserved names like:

- User > LoginUser, AppUser, SystemUser, CMSUser,...

Or use the infamous squared brackets [User]

[Answer](#) by [brian-boatright](#)

If you use Object Relational Mapping tools or will in the future I suggest **Singular**.

Some tools like LLBLGen can automatically correct plural names like Users to User without changing the table name itself. Why does this matter? Because when it's mapped you want it to look like User.Name instead of Users.Name or worse from some of my old databases tables naming tblUsers.strName which is just confusing in code.

My new rule of thumb is to judge how it will look once it's been converted into an object. one table I've found that does not fit the new naming I use is UsersInRoles. But there will always be those few exceptions and even in this case it looks fine as UsersInRoles.Username.

[Answer](#) by [bill-karwin](#)

In his book "[SQL Programming Style](#)," Joe Celko suggests that a collection (e.g. a table) should be named in the plural, while a scalar data element (e.g. a column) should be named in the singular.

He cites [ISO-11179-4](#) as a standard for metadata naming, which supports this guideline.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: Left join and Left outer join in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

What is the difference between left join and left outer join?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [kg-sosa](#)

[Answer](#) by [lasse-v.-karlsen](#)

As per the documentation: [FROM \(Transact-SQL\)](#):

```
<join_type> ::=  
[ { INNER | { { LEFT | RIGHT | FULL } [ OUTER ] } } [ <join_hint> ] ]  
JOIN
```

The keyword OUTER is marked as optional (enclosed in square brackets), and what this means in this case is that whether you specify it or not makes no difference. Note that while the other elements of the join clause is also marked as optional, leaving *them* out will of course make a difference.

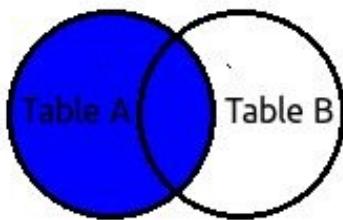
For instance, the entire type-part of the JOIN clause is optional, in which case the default is INNER if you just specify JOIN. In other words, this is legal:

```
SELECT *  
FROM A JOIN B ON A.X = B.Y
```

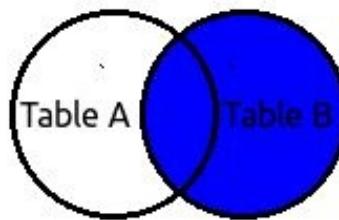
Here's a list of equivalent syntaxes:

A LEFT JOIN B	A LEFT OUTER JOIN B
A RIGHT JOIN B	A RIGHT OUTER JOIN B
A FULL JOIN B	A FULL OUTER JOIN B
A INNER JOIN B	A JOIN B

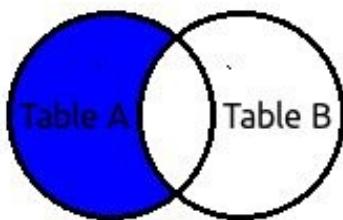
Also take a look at the answer I left on this other SO question: [SQL left join vs multiple tables on FROM line?](#)



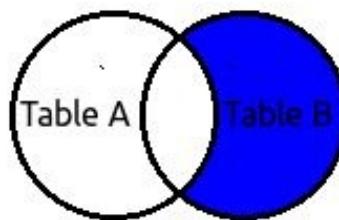
```
SELECT [list] FROM
[Table A] A
LEFT JOIN
[Table B] B
ON A.Value = B.Value
```



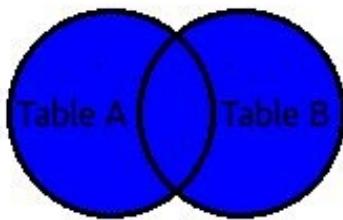
```
SELECT [list] FROM
[Table A] A
RIGHT JOIN
[Table B] B
ON A.Value = B.Value
```



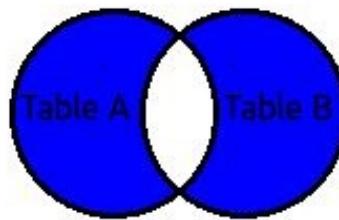
```
SELECT [list] FROM
[Table A] A
LEFT JOIN
[Table B] B
ON A.Value = B.Value
WHERE B.Value IS NULL
```



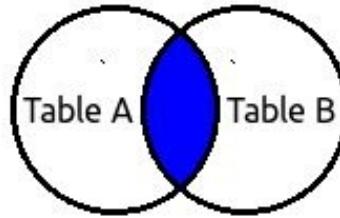
```
SELECT [list] FROM
[Table A] A
RIGHT JOIN
[Table B] B
ON A.Value = B.Value
WHERE A.Value IS NULL
```



```
SELECT [list] FROM
[Table A] A
FULL OUTER JOIN
[Table B] B
ON A.Value = B.Value
```



```
SELECT [list] FROM
[Table A] A
FULL OUTER JOIN
[Table B] B
ON A.Value = B.Value
WHERE A.Value IS NULL
OR B.Value IS NULL
```



```
SELECT [list] FROM
[Table A] A
INNER JOIN
[Table B] B
ON A.Value = B.Value
```

[Answer](#) by [sactiw](#)

To answer your question **there is no difference between LEFT JOIN and LEFT OUTER JOIN, they are exactly same** that said...

At the top level there are mainly 3 types of joins:

1. INNER
2. OUTER
3. CROSS

1. **INNER JOIN** - fetches data if present in both the tables.

2. **OUTER JOIN** are of 3 types:

1. LEFT OUTER JOIN - fetches data if present in the left table.
2. RIGHT OUTER JOIN - fetches data if present in the right table.
3. FULL OUTER JOIN - fetches data if present in either of the two tables.

3. **CROSS JOIN**, as the name suggests, does $[n \times m]$ that joins everything to everything.

Similar to scenario where we simply lists the tables for joining (in the `FROM` clause of the `SELECT` statement), using commas to separate them.

Points to be noted:

- If you just mention `JOIN` then by default it is a `INNER JOIN`.
 - An `OUTER` join has to be `LEFT | RIGHT | FULL` you can not simply say `OUTER JOIN`.
 - You can drop `OUTER` keyword and just say `LEFT JOIN` or `RIGHT JOIN` or `FULL JOIN`.
-

For those who want to visualise these in a better way, please go to this link: [A Visual Explanation of SQL Joins](#)

[Answer](#) by [mitch-wheat](#)

What is the difference between left join and left outer join?

Nothing. `LEFT JOIN` and `LEFT OUTER JOIN` are equivalent.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How do you import a large MS SQL .sql file?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I use RedGate SQL data compare and generate a .sql file, so I can run it on my local machine. But the problem is the file is over 300mb and I can't do copy and paste because the clipboard won't able to handle it and when I try to open the file in the SQL Server Management Studio I get an error about the file being too large.

Is there a way to run a large .sql file? The file basically contain data for two new tables.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [jack](#)

[Answer](#) by [ray-booyesen](#)

From the command prompt, start up `sqlcmd`:

```
sqlcmd -S <server> -i C:\<your file here>.sql
```

Just replace `<server>` with the location of your SQL box and `<your file here>` with the

name of your script. Don't forget, if you're using a SQL instance the syntax is:

```
sqlcmd -S <server>\instance.
```

Here is the list of all arguments you can pass sqlcmd:

[Skip code block](#)

```
Sqlcmd      [-U login id]      [-P password]
[-S server]      [-H hostname]      [-E trusted connection]
[-d use database name]  [-l login timeout]      [-t query timeout]
[-h headers]      [-s colseparator]      [-w screen width]
[-a packetsize]     [-e echo input]      [-I Enable Quoted Identifiers]
[-c cmdend]      [-L[c] list servers[clean output]]
[-q "cmdline query"]  [-Q "cmdline query" and exit]
[-m errorlevel]     [-V severitylevel]      [-W remove trailing spaces]
[-u unicode output]  [-r[0|1] msgs to stderr]
[-i inputfile]      [-o outputfile]      [-z new password]
[-f | i:[,o:]]  [-Z new password and exit]
[-k[1|2] remove[replace] control characters]
[-y variable length type display width]
[-Y fixed length type display width]
[-p[1] print statistics[:colon format]]
[-R use client regional setting]
[-b On error batch abort]
[-v var = "value"...]  [-A dedicated admin connection]
[-X[1] disable commands, startup script, environment variables [and exit]]
[-x disable variable substitution]
[-? show syntax summary]
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: Is a view faster than a simple query?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

Is a

```
select * from myView
```

faster than the query itself to create the view (in order to have the same resultSet):

```
select * from ([query to create same resultSet as myView])
```

?

It's not totally clear to me if the view uses some sort of caching making it faster compared to a simple query.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

User: [johnidol](#) 

[Answer](#)  by [mark-brittingham](#) 

Yes, views can have a clustered index assigned and, when they do, they'll store temporary results that can speed up resulting queries.

Update: At least three people have voted me down on this one. With all due respect, I think that they are just wrong; Microsoft's own documentation makes it very clear that Views can improve performance.

First, simple views are expanded in place and so do not directly contribute to performance improvements - that much is true. **However**, indexed views can *dramatically* improve performance.

Let me go directly to the documentation:

After a unique clustered index is created on the view, the view's result set is materialized immediately and persisted in physical storage in the database, saving the overhead of performing this costly operation at execution time.

Second, these indexed views can work *even when they are not directly referenced by another query* as the optimizer will use them in place of a table reference when appropriate.

Again, the documentation:

The indexed view can be used in a query execution in two ways. The query can reference the indexed view directly, or, more importantly, the query optimizer can select the view if it determines that the view can be substituted for some or all of the query in the lowest-cost query plan. In the second case, the indexed view is used instead of the underlying tables and their ordinary indexes. The view does not need to be referenced in the query for the query optimizer to use it during query execution. This allows existing applications to benefit from the newly created indexed views without changing those applications.

This documentation, as well as charts demonstrating performance improvements, can be found [here](#) .

Update 2: the answer has been criticized on the basis that it is the “index” that provides the performance advantage, not the “View.” However, this is easily refuted.

Let us say that we are a software company in a small country; I'll use Lithuania as an example. We sell software worldwide and keep our records in a SQL Server database. We're very successful and so, in a few years, we have 1,000,000+ records. However, we often need to report sales for tax purposes and we find that we've only sold 100 copies of our software in our home country. By creating an indexed view of just the Lithuanian records, we get to keep the records we need in an indexed cache as described in the MS documentation. When we run our reports for Lithuanian sales in 2008, our query will search through an index with a depth of just 7 ($\text{Log}_2(100)$ with some unused leaves). If we were to do the same without the VIEW and just relying on an index into the table, we'd have to traverse an index tree with a search depth of 21!

Clearly, the View itself would provide us with a performance advantage (3x) over the simple use of the index alone. I've tried to use a real-world example but you'll note that a simple list of Lithuanian sales would give us an even greater advantage.

Note that I'm just using a straight b-tree for my example. While I'm fairly certain that SQL Server uses some variant of a b-tree, I don't know the details. Nonetheless, the point holds.

Update 3: The question has come up about whether an Indexed View just uses an index

placed on the underlying table. That is, to paraphrase: “an indexed view is just the equivalent of a standard index and it offers nothing new or unique to a view.” If this was true, of course, then the above analysis would be incorrect! Let me provide a quote from the Microsoft documentation that demonstrate why I think this criticism is not valid or true:

Using indexes to improve query performance is not a new concept; however, indexed views provide additional performance benefits that cannot be achieved using standard indexes.

Together with the above quote regarding the persistence of data in physical storage and other information in the documentation about how indices are created on Views, I think it is safe to say that an Indexed View is **not** just a cached SQL Select that happens to use an index defined on the main table. Thus, I continue to stand by this answer.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

[Q: How do I set a column value to NULL in SQL Server Management Studio?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

How do I clear the value from a cell and make it NULL?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [zack-peterson](#) 

[Answer](#)  by [jeff-martin](#) 

I think [@Zack](#)  properly answered the question but just to cover all the bases:

```
Update myTable set MyColumn = NULL
```

This would set the entire column to null as the Question Title asks.

To set a specific row on a specific column to null use:

```
Update myTable set MyColumn = NULL where Field = Condition.
```

This would set a specific cell to null as the inner question asks.

[Answer](#)  by [zack-peterson](#) 

If you've opened a table and you want to clear an existing value to NULL, click on the value, and press **Ctrl+0**.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: How do I drop a foreign key constraint only if it exists in sql server?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I can drop a table if it exists using the following code but do not know how to do the same with a constraint:

```
IF EXISTS(SELECT 1 FROM sys.objects WHERE OBJECT_ID = OBJECT_ID(N'TableName') AND type = (N'U')) DROP TABLE [TableName]
go
```

I also add the constraint using this code:

```
ALTER TABLE [dbo].[TableName]
    WITH CHECK ADD CONSTRAINT [FK_TableName_TableName2] FOREIGN KEY([FK_Name])
        REFERENCES [dbo].[TableName2] ([ID])
go
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [solrevdev](#) 

[Answer](#)  by [james-l](#) 

```
IF EXISTS (SELECT *
    FROM sys.foreign_keys
    WHERE object_id = OBJECT_ID(N'dbo.FK_TableName_TableName2')
    AND parent_object_id = OBJECT_ID(N'dbo.TableName'))
) ALTER TABLE [dbo.TableName] DROP CONSTRAINT [FK_TableName_TableName2]
```

[Answer](#)  by [eric-isaacs](#) 

This is a lot simpler than the current proposed solution:

```
IF (OBJECT_ID('FK_ConstraintName', 'F') IS NOT NULL)
BEGIN
    ALTER TABLE dbo.TableName DROP CONSTRAINT FK_ConstraintName
END
```

If you need to drop another type of constraint, these are the applicable codes to pass into the OBJECT_ID() function in the second parameter position:

```
C = CHECK constraint
D = DEFAULT (constraint or stand-alone)
F = FOREIGN KEY constraint
PK = PRIMARY KEY constraint
UQ = UNIQUE constraint
```

You can also use OBJECT_ID without the second parameter.

Full List of types [here](#) 

Object type:

```
AF = Aggregate function (CLR)
C = CHECK constraint
D = DEFAULT (constraint or stand-alone)
F = FOREIGN KEY constraint
FN = SQL scalar function
FS = Assembly (CLR) scalar-function
FT = Assembly (CLR) table-valued function
IF = SQL inline table-valued function
IT = Internal table
P = SQL Stored Procedure
PC = Assembly (CLR) stored-procedure
```

```
PG = Plan guide
PK = PRIMARY KEY constraint
R = Rule (old-style, stand-alone)
RF = Replication-filter-procedure
S = System base table
SN = Synonym
SO = Sequence object
```

Applies to: SQL Server 2012 through SQL Server 2014.

```
SQ = Service queue
TA = Assembly (CLR) DML trigger
TF = SQL table-valued-function
TR = SQL DML trigger
TT = Table type
U = Table (user-defined)
UQ = UNIQUE constraint
V = View
X = Extended stored procedure
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

Q: How can I list all foreign keys referencing a given table in SQL Server?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I need to remove a highly referenced table in a SQL Server database. How can I get a list of all the foreign key constraints I will need to remove in order to drop the table?

(SQL answers preferable over clicking about in the GUI of the management studio.)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [chillitom](#) 

[Answer](#)  by [recep](#) 

Not sure why no one suggested but I use sp_fkeys to query foreign keys for a given table:

```
EXEC sp_fkeys 'TableName'
```

[Answer](#)  by [gishu](#) 

I'd use the Database Diagramming feature in SQL Server Management studio.. but since you ruled that out..

This worked for me in SQL2008 (don't have 2005)

To get list of referring table and column names...

```
select t.name as TableWithForeignKey, fk.constraint_column_id as FK_PartNo , c.name as ForeignKeyColu
from sys.foreign_key_columns as fk
inner join sys.tables as t on fk.parent_object_id = t.object_id
inner join sys.columns as c on fk.parent_object_id = c.object_id and fk.parent_column_id = c.column_i
where fk.referenced_object_id = (select object_id from sys.tables where name = 'TableOthersForeignKey'
order by TableWithForeignKey, FK_PartNo
```

To get names of foreign key constraints

```
select distinct name from sys.objects where object_id in
( select fk.constraint_object_id from sys.foreign_key_columns as fk
  where fk.referenced_object_id =
    (select object_id from sys.tables where name = 'Table0OthersForeignKeyInto')
)
```

[Answer](#) by [bankz](#)

Try this :

```
sp_help 'TableName'
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Best way to do nested case statement logic in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Next Q](#))

I'm currently writing an SQL Query, where a few of the columns returned need to be calculated depending on quite a lot of conditions.

I'm currently using nested case statements, but its getting messy. Is there a better (more organised and/or readable) way?

(I am using Microsoft SQL Server, 2005)

A simplified example:

[Skip code block](#)

```
SELECT
  col1,
  col2,
  col3,
CASE
    WHEN condition
    THEN
      CASE
        WHEN condition1
        THEN
          CASE
            WHEN condition2
            THEN calculation1
            ELSE calculation2
          END
        ELSE
          CASE
            WHEN condition2
            THEN calculation3
            ELSE calculation4
          END
        END
      END
    ELSE
      CASE
        WHEN condition1
        THEN
          CASE
            WHEN condition2
            THEN calculation5
            ELSE calculation6
          END
        ELSE
          CASE
```

```
        WHEN condition2
        THEN calculation7
        ELSE calculation8
    END
END AS 'calculatedcol1',
col4,
col5-etc
FROM table
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Next Q](#))

User: [sophia](#) 

[Answer](#)  by [chris-kl](#) 

You could try some sort of COALESCE trick, eg:

```
SELECT COALESCE(
CASE WHEN condition1 THEN calculation1 ELSE NULL END,
CASE WHEN condition2 THEN calculation2 ELSE NULL END,
etc...
)
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Next Q](#))

[Q: Drop all the tables, stored procedures, triggers, constraints and all the dependencies in one sql statement](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

Is there any way in which I can clean a database in SQL Server 2005 by dropping all the tables and deleting stored procedures, triggers, constraints and all the dependencies in one SQL statement?

REASON FOR REQUEST:

I want to have a DB script for cleaning up an existing DB which is not in use rather than creating new ones, especially when you have to put in a request to your DB admin and wait for a while to get it done!

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [renegademind](#) 

[Answer](#)  by ivan

this script cleans all views, SPS, functions PKs, FKs and tables.

[Skip code block](#)

```
/* Drop all non-system stored procs */
DECLARE @name VARCHAR(128)
DECLARE @SQL VARCHAR(254)

SELECT @name = (SELECT TOP 1 [name] FROM sysobjects WHERE [type] = 'P' AND category = 0 ORDER BY [name])
WHILE @name is not null
```

```

BEGIN
    SELECT @SQL = 'DROP PROCEDURE [dbo].[ ' + RTRIM(@name) +' ]'
    EXEC (@SQL)
    PRINT 'Dropped Procedure: ' + @name
    SELECT @name = (SELECT TOP 1 [name] FROM sysobjects WHERE [type] = 'P' AND category = 0 AND [name]
END
GO

/* Drop all views */
DECLARE @name VARCHAR(128)
DECLARE @SQL VARCHAR(254)

SELECT @name = (SELECT TOP 1 [name] FROM sysobjects WHERE [type] = 'V' AND category = 0 ORDER BY [name]

WHILE @name IS NOT NULL
BEGIN
    SELECT @SQL = 'DROP VIEW [dbo].[ ' + RTRIM(@name) +' ]'
    EXEC (@SQL)
    PRINT 'Dropped View: ' + @name
    SELECT @name = (SELECT TOP 1 [name] FROM sysobjects WHERE [type] = 'V' AND category = 0 AND [name]
END
GO

/* Drop all functions */
DECLARE @name VARCHAR(128)
DECLARE @SQL VARCHAR(254)

SELECT @name = (SELECT TOP 1 [name] FROM sysobjects WHERE [type] IN (N'FN', N'IF', N'TF', N'FS', N'FT')

WHILE @name IS NOT NULL
BEGIN
    SELECT @SQL = 'DROP FUNCTION [dbo].[ ' + RTRIM(@name) +' ]'
    EXEC (@SQL)
    PRINT 'Dropped Function: ' + @name
    SELECT @name = (SELECT TOP 1 [name] FROM sysobjects WHERE [type] IN (N'FN', N'IF', N'TF', N'FS',
END
GO

/* Drop all Foreign Key constraints */
DECLARE @name VARCHAR(128)
DECLARE @constraint VARCHAR(254)
DECLARE @SQL VARCHAR(254)

SELECT @name = (SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS WHERE constraint_ca
WHILE @name is not null
BEGIN
    SELECT @constraint = (SELECT TOP 1 CONSTRAINT_NAME FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS WHE
    WHILE @constraint IS NOT NULL
    BEGIN
        SELECT @SQL = 'ALTER TABLE [dbo].[ ' + RTRIM(@name) +' ] DROP CONSTRAINT [' + RTRIM(@constraint
        EXEC (@SQL)
        PRINT 'Dropped FK Constraint: ' + @constraint + ' on ' + @name
        SELECT @constraint = (SELECT TOP 1 CONSTRAINT_NAME FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS
    END
SELECT @name = (SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS WHERE constraint_ca
END
GO

/* Drop all Primary Key constraints */
DECLARE @name VARCHAR(128)
DECLARE @constraint VARCHAR(254)
DECLARE @SQL VARCHAR(254)

SELECT @name = (SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS WHERE constraint_ca
WHILE @name IS NOT NULL
BEGIN
    SELECT @constraint = (SELECT TOP 1 CONSTRAINT_NAME FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS WHE
    WHILE @constraint is not null
    BEGIN
        SELECT @SQL = 'ALTER TABLE [dbo].[ ' + RTRIM(@name) +' ] DROP CONSTRAINT [' + RTRIM(@constraint
        EXEC (@SQL)
        PRINT 'Dropped PK Constraint: ' + @constraint + ' on ' + @name
        SELECT @constraint = (SELECT TOP 1 CONSTRAINT_NAME FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS
    END

```

```

SELECT @name = (SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLE_CONSTRAINTS WHERE constraint_ca
END
GO

/* Drop all tables */
DECLARE @name VARCHAR(128)
DECLARE @SQL VARCHAR(254)

SELECT @name = (SELECT TOP 1 [name] FROM sysobjects WHERE [type] = 'U' AND category = 0 ORDER BY [name]
WHILE @name IS NOT NULL
BEGIN
    SELECT @SQL = 'DROP TABLE [dbo].[ ' + RTRIM(@name) +' ]'
    EXEC (@SQL)
    PRINT 'Dropped Table: ' + @name
    SELECT @name = (SELECT TOP 1 [name] FROM sysobjects WHERE [type] = 'U' AND category = 0 AND [name]
END
GO

```

[Answer](#) by [vlad](#)

I'm using this script by Adam Anderson, updated to support objects in other schemas than dbo.

[Skip code block](#)

```

declare @n char(1)
set @n = char(10)

declare @stmt nvarchar(max)
--procedures
select @stmt = isnull( @stmt + @n, '' ) +
'drop procedure [' + schema_name(schema_id) + '].[' + name + ']'
from sys.procedures

--check constraints
select @stmt = isnull( @stmt + @n, '' ) +
'alter table [' + schema_name(schema_id) + '].[' + object_name( parent_object_id ) + ']' drop const
from sys.check_constraints
--functions
select @stmt = isnull( @stmt + @n, '' ) +
'drop function [' + schema_name(schema_id) + '].[' + name + ']'
from sys.objects
where type in ( 'FN', 'IF', 'TF' )
--views
select @stmt = isnull( @stmt + @n, '' ) +
'drop view [' + schema_name(schema_id) + '].[' + name + ']'
from sys.views
--foreign keys
select @stmt = isnull( @stmt + @n, '' ) +
'alter table [' + schema_name(schema_id) + '].[' + object_name( parent_object_id ) + ']' drop cons
from sys.foreign_keys
--tables
select @stmt = isnull( @stmt + @n, '' ) +
'drop table [' + schema_name(schema_id) + '].[' + name + ']'
from sys.tables
--user defined types
select @stmt = isnull( @stmt + @n, '' ) +
'drop type [' + schema_name(schema_id) + '].[' + name + ']'
from sys.types
where is_user_defined = 1

exec sp_executesql @stmt

```

Source: http://blog.falafel.com/Blogs/AdamAnderson/09-01-06/SQL_Drop_All_Objects_in_a_SQL_Server_Database.aspx

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

Q: efficient way to implement paging

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [linq-to-sql](#) ([Next Q](#))

Should I use LINQ's `skip()` and `Take()` method for paging, or implement my own paging with a SQL query?

Which is most efficient? Why would I choose one over the other?

I'm using SQL Server 2008, ASP.NET MVC and LINQ.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [linq-to-sql](#) ([Next Q](#))

User: [stoneheart](#)

[Answer](#) by [rodrigoelp](#)

Trying to give you a brief answer to your doubt, if you execute the `skip(n).take(m)` methods on linq (with SQL 2005 / 2008 as database server) your query will be using the `Select ROW_NUMBER() Over...` statement, with is somehow direct paging in the SQL engine.

Giving you an example, I have a db table called `mtcity` and I wrote the following query (work as well with linq to entities):

```
using (DataClasses1DataContext c = new DataClasses1DataContext())
{
    var query = (from MtCity2 c1 in c.MtCity2s
                 select c1).Skip(3).Take(3);
    //Doing something with the query.
}
```

The resulting query will be:

[Skip code block](#)

```
SELECT [t1].[CodCity],
       [t1].[CodCountry],
       [t1].[CodRegion],
       [t1].[Name],
       [t1].[Code]
  FROM (
    SELECT ROW_NUMBER() OVER (
        ORDER BY [t0].[CodCity],
        [t0].[CodCountry],
        [t0].[CodRegion],
        [t0].[Name],
        [t0].[Code]) AS [ROW_NUMBER],
        [t0].[CodCity],
        [t0].[CodCountry],
        [t0].[CodRegion],
        [t0].[Name],
        [t0].[Code]
   FROM [dbo].[MtCity] AS [t0]
  ) AS [t1]
 WHERE [t1].[ROW_NUMBER] BETWEEN @p0 + 1 AND @p0 + @p1
 ORDER BY [t1].[ROW_NUMBER]
```

Which is a windowed data access (pretty cool, btw cuz will be returning data since the very begining and will access the table as long as the conditions are met). This will be very similar to:

[Skip code block](#)

```
With CityEntities As
()
```

```

Select ROW_NUMBER() Over (Order By CodCity) As Row,
      CodCity /*here is only accessed by the Index as CodCity is the primary
From dbo.mtcity
)
Select [t0].[CodCity],
      [t0].[CodCountry],
      [t0].[CodRegion],
      [t0].[Name],
      [t0].[Code]
From CityEntities c
Inner Join dbo.Mtcity t0 on c.CodCity = t0.CodCity
Where c.Row Between @p0 + 1 AND @p0 + @p1
Order By c.Row Asc

```

With the exception that, this second query will be executed faster than the linq result because it will be using exclusively the index to create the data access window; this means, if you need some filtering, the filtering should be (or must be) in the Entity listing (where the row is created) and some indexes should be created as well to keep up the good performance.

Now, what's better?

If you have pretty much solid workflow in your logic, implementing the proper SQL way will be complicated. In that case LINQ will be the solution.

If you can lower that part of the logic directly to SQL (in a stored procedure), it will be even better because you can implement the second query I showed you (using indexes) and allow SQL to generate and store the Execution Plan of the query (improving performance).

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [linq-to-sql](#) ([Next Q](#))

Q: Difference between JOIN and INNER JOIN

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Next Q](#))

Both these joins will give me the same results:

```
SELECT * FROM table JOIN otherTable ON table.ID = otherTable.FK
```

VS

```
SELECT * FROM table INNER JOIN otherTable ON table.ID = otherTable.FK
```

Is there any difference between the statements in performance or otherwise?

Does it differ between different SQL implementations?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Next Q](#))

User: [driis](#)

[Answer](#) by [palehorse](#)

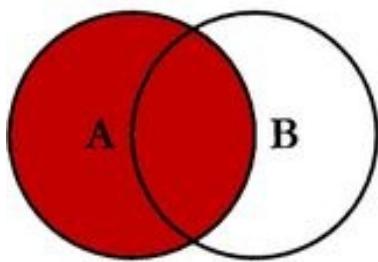
They are functionally equivalent, but `INNER JOIN` can be a bit clearer to read, especially if the query has other join types (i.e. `LEFT` or `RIGHT` or `CROSS`) included in it.

[Answer](#) by [damian-drygiel](#)

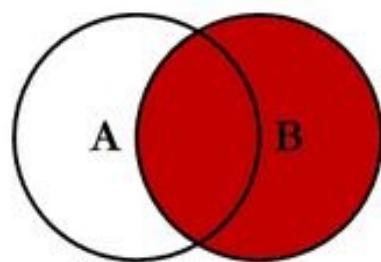
Just typing `JOIN` performs an `INNER JOIN` by default.

For all others, one picture is sometimes worth more than hundreds of words:

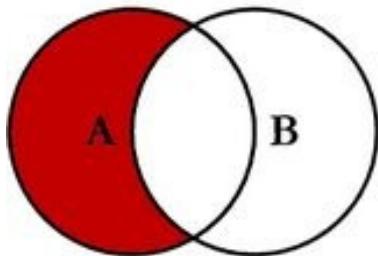
SQL JOINS



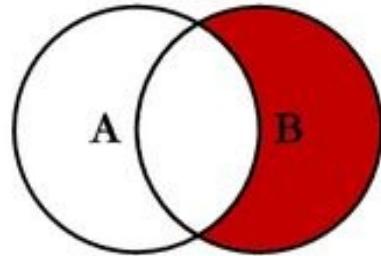
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```

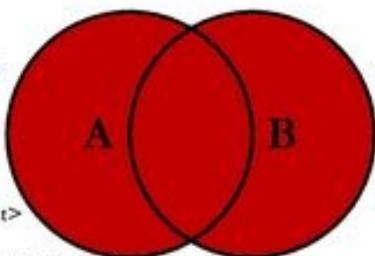


```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```

```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

© C.L. Moffatt, 2008

Image courtesy of [CodeProject](#)

[Answer](#) by [quassnoi](#)

No, there is no difference, pure [syntactic sugar](#).

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Next Q](#))

Q: SQL Server: How do you return the column names from a table?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Please can someone show me how I would return the column names of a table using SQL server 2008?

i.e. a table contains the columns id, name, address, country and I want to return these as data?

Thank you

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [belliez](#)

[Answer](#) by [gulzar-nazim](#)

Not sure if there is an easier way in 2008 version.

```
USE [Database Name]
SELECT COLUMN_NAME, *
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = 'YourTableName' AND TABLE_SCHEMA='YourSchemaName'
```

[Answer](#) by [diggdev](#)

This is the easiest way

```
exec sp_columns [tablename]
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL Server - stop or break execution of a SQL script](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Is there a way to immediately stop execution of a SQL script in SQL server, like a “break” or “exit” command?

I have a script that does some validation and lookups before it starts doing inserts, and I want it to stop if any of the validations or lookups fail.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [andy-white](#)

[Answer](#) by [blorgbeard](#)

The [raiserror](#) method

```
raiserror('Oh no a fatal error', 20, -1) with log
```

This will terminate the connection, thereby stopping the rest of the script from running.

This even works with GO statements, eg.

```
print 'hi'
go
raiserror('Oh no a fatal error', 20, -1) with log
go
print 'ho'
```

Will give you the output:

```
hi
Msg 2745, Level 16, State 2, Line 1
Process ID 51 has raised user error 50000, severity 20. SQL Server is terminating this process.
Msg 50000, Level 20, State 1, Line 1
Oh no a fatal error
Msg 0, Level 20, State 0, Line 0
A severe error occurred on the current command. The results, if any, should be discarded.
```

Notice that ‘ho’ is not printed.

CAVEATS:

- This only works if you are logged in as admin ('sysadmin' role), and also leaves you with no database connection.
- If you are NOT logged in as admin, the RAISEERROR() call itself will fail *and the script will continue executing*.
- When invoked with sqlcmd.exe, exit code 2745 will be reported.

Reference: <http://www.mydatabasesupport.com/forums/ms-sqlserver/174037-sql-server-2000-abort-whole-script.html#post761334>

The noexec method

Another method that works with GO statements is set noexec on. This causes the rest of the script to be skipped over. It does not terminate the connection, but you need to turn noexec off again before any commands will execute.

Example:

Skip code block

```
print 'hi'  
go  
  
print 'Fatal error, script will not continue!'  
set noexec on  
  
print 'ho'  
go  
--last line of the script  
set noexec off--Turn execution back on; only needed in SSMS, so as to be able  
--to run this script again in the same session.
```

Answer [by gordon-bell](#)

Just use a RETURN (it will work both inside and outside a stored procedure).

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Q: What is best tool to compare two SQL Server databases (schema and data)?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

Possible Duplicate:

[Free Tool to compare Sql Server tables](#)

I would like to compare two SQL Server databases including schema (table structure) and data in tables too. What is best tool to do this?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [tomaszs](#)

Answer [by joakim-backman](#)

I am using Red-Gate's software: <http://www.red-gate.com>

[Answer](#) by mauro

I like [Open DBDiff](#). It's not the most complete tool, but it works great and it's free. And it's very easy to use.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to combine date from one field with time from another field - MS SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

In an extract I am dealing with, I have 2 datetime columns. One column stores the dates and another the times as shown.

How can I query the table to combine these two fields into 1 column of type datetime?

Dates

```
2009-03-12 00:00:00.000
2009-03-26 00:00:00.000
2009-03-26 00:00:00.000
```

Times

```
1899-12-30 12:30:00.000
1899-12-30 10:00:00.000
1899-12-30 10:00:00.000
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

User: [jon-winstanley](#)

[Answer](#) by [lieven-keersmaekers](#)

You can simply add the two.

- if the Time part of your Date column is always zero
- and the Date part of your Time column is also always zero (*base date: January 1, 1900*)

Adding them returns the correct result.

```
SELECT Combined = MyDate + MyTime FROM MyTable
```

Rationale (kudos to ErikE/dnolan)

```
It works like this due to the way the date is stored as two 4-byte `Integers` with the left 4-bytes being the `date` and the right 4-bytes being the `time`.
```

```
Its like doing $0001 0000 + $0000 0001 = $0001 0001
```

[Edit regarding new SQL Server 2008 types](#)

Date and Time are types introduced in SQL Server 2008. If you insist on adding, you can use Combined = CAST(MyDate AS DATETIME) + CAST(MyTime AS DATETIME)

Edit2 regarding loss of precision in SQL Server 2008 and up (kudos to Martin Smith)

Have a look at [How to combine date and time to datetime2 in SQL Server?](#) to prevent loss of precision using SQL Server 2008 and up.

[Answer](#) by [lukeh](#)

If the time element of your date column *and* the date element of your time column are both zero then [Lieven's answer](#) is what you need. If you can't guarantee that will always be the case then it becomes slightly more complicated:

```
SELECT DATEADD(day, 0, DATEDIFF(day, 0, your_date_column)) +
       DATEADD(day, 0 - DATEDIFF(day, 0, your_time_column), your_time_column)
FROM your_table
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

[Q: T-SQL Cast versus Convert](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

What is the general guidance on when you should use CAST versus CONVERT? Is there any performance issues related to choosing one versus the other? Is one closer to ANSI-SQL?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [buddyjoe](#)

[Answer](#) by [matthew-farwell](#)

CONVERT is SQL Server specific, CAST is ANSI.

CONVERT is more flexible in that you can format dates etc. Other than that, they are pretty much the same. If you don't care about the extended features, use CAST.

EDIT:

As noted by @beruic and @C-F in the comments below, there is possible loss of precision when an implicit conversion is used (that is one where you use neither CAST nor CONVERT). For further information, see [CAST and CONVERT](#) and in particular this graphic: [SQL Server Data Type Conversion Chart](#). With this extra information, the original advice still remains the same. Use CAST where possible.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: Should I use != or <> for not equal in TSQL?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I have seen SQL that uses both != and <> for *not equal*. What is the preferred syntax and why?

I like != because <> reminds me of Visual Basic.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [bob-the-janitor](#) 

[Answer](#)  by [dbandrew](#) 

Technically they function the same if you're using MS SQL aka T-SQL. If you're using it in stored procedures there is no performance reason to use one over the other. It then comes down to personal preference. I prefer to use <> as it is ANSI compliant.

You can find links to the various ANSI standards at...

<http://en.wikipedia.org/wiki/SQL> 

[Answer](#)  by [bill-karwin](#) 

Most databases support != (popular programming languages) and <> (ANSI).

Databases that support both != and <>:

- MySQL 5.1: [!= and <>](#) 
- PostgreSQL 8.3: [!= and <>](#) 
- SQLite: [!= and <>](#) 
- Oracle 10g: [!= and <>](#) 
- Microsoft SQL Server 2000/2005/2008: [!=](#)  and [<>](#) 
- IBM Informix Dynamic Server 10: [!= and <>](#) 
- InterBase/Firebird: [!= and <>](#) 
- Apache Derby 10.6: [!= and <>](#) 
- Sybase Adaptive Server Enterprise 11.0: [!= and <>](#) 

Databases that support the ANSI standard operator, **exclusively**:

- IBM DB2 UDB 9.5: [<>](#) 
 - Microsoft Access 2010: [<>](#) 
-

[Answer](#)  by [adam-lassek](#) 

'<>' is from the [SQL-92 standard](#) , '!= is a [proprietary](#)  T-SQL operator. It's available in other databases as well, but since it isn't standard you have to take it on a case-by-case basis.

In most cases, you'll know what database you're connecting to so this isn't really an issue. At worst you might have to do a search and replace in your sql.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Turn off constraints temporarily \(MS SQL\)](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I'm looking the way to temporarily turn off all DB's constraints (eg table relationships)

I need to copy (using INSERTs) one DBs tables to another DB I know I can archive that by executing commands in proper order (to do not break relationships)

But it would be easier if I could turn off checking constraints temporarily and turn on it back after operation finish.

Is this possible?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [maciej](#) 

[Answer](#)  by [gbn](#) 

You can disable FK and CHECK constraints ~~only in SQL 2005+~~. See [ALTER TABLE](#) 

```
ALTER TABLE foo NOCHECK CONSTRAINT ALL
```

or

```
ALTER TABLE foo NOCHECK CONSTRAINT CK_foo_column
```

Primary keys and unique constraints can not be disabled, but this should be OK if I've understood you correctly.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: Does MS SQL Server's “between” include the range boundaries?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

For instance can

```
SELECT foo
FROM bar
WHERE foo BETWEEN 5 AND 10
```

select 5 and 10 or they are excluded from the range?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [lea-verou](#) 

[Answer](#)  by [d.j.](#) 

The BETWEEN operator is inclusive.

From Books Online:

BETWEEN returns TRUE if the value of test_expression is greater than or equal to the value of begin_expression and less than or equal to the value of end_expression.

[Answer](#) by [bradc](#)

Yes, but be careful when using between for dates.

```
BETWEEN '01/01/2009' AND '01/31/2009'
```

is really interpreted as 12am, or

```
BETWEEN '01/01/2009 00:00:00' AND '01/31/2009 00:00:00'
```

so will miss anything that occurred during the day of Jan 31st. In this case, you will have to use:

```
myDate >= '01/01/2009 00:00:00' AND myDate < '02/01/2009 00:00:00' --CORRECT!
```

or

```
BETWEEN '01/01/2009 00:00:00' AND '01/31/2009 23:59:59' --WRONG! (see update!)
```

UPDATE: It is entirely possible to have records created within that last second of the day, with a datetime as late as 01/01/2009 23:59:59.997!!

For this reason, the BETWEEN (firstday) AND (lastday 23:59:59) approach is not recommended.

Use the myDate >= (firstday) AND myDate < (Lastday+1) approach instead.

Good [article on this issue here](#).

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to get first character of a string in SQL?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I have a SQL column with length of 6. Now want to take only first char of that column. Is there any string function in SQL to do this?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [vinodtiru](#)

[Answer](#) by [eric](#)

LEFT(colName, 1) will also do this, also. It's equivalent to SUBSTRING(colName, 1, 1).

I like LEFT, since I find it a bit cleaner, but really, there's no difference either way.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to replace a string in a SQL Server Table Column](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

I have a table (SQL Server) which references paths (UNC or otherwise), but now the path is going to change. In the path column, I have many records and I need to change just a portion of the path, but not the entire path. And I need to change the same string to the new one, in every record.

How can I do this with a simple update?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [dada](#) 

[Answer](#)  by [cjk](#) 

It's this easy:

```
update my_table  
set path = replace(path, 'oldstring', 'newstring')
```

[Answer](#)  by [marc-gravell](#) 

```
UPDATE [table]  
SET [column] = REPLACE([column], '/foo/', '/bar/')
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: Select n random rows from SQL Server table](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [random](#) ([Next Q](#))

I've got a SQL Server table with about 50,000 rows in it. I want to select about 5,000 of those rows at random. I've thought of a complicated way, creating a temp table with a "random number" column, copying my table into that, looping through the temp table and updating each row with RAND(), and then selecting from that table where the random number column < 0.1. I'm looking for a simpler way to do it, in a single statement if possible.

[This article](#) suggest using the NEWID() function. That looks promising, but I can't see how I could reliably select a certain percentage of rows.

Anybody ever do this before? Any ideas?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [random](#) ([Next Q](#))

User: [john-m-gant](#)

[Answer](#) by [ralph-shillington](#)

```
select top 10 percent * from [yourtable] order by newid()
```

In response to the "pure trash" comment concerning large tables: you could do it like this to improve performance.

```
select * from [yourtable] where [yourPk] in  
(select top 10 percent [yourPk] from [yourtable] order by newid())
```

The cost of this will be the key scan of values plus the join cost, which on a large table with a small percentage selection should be reasonable.

[Answer](#) by [patrick-taylor](#)

Depending on your needs, TABLESAMPLE will get you nearly as random and better performance. this is available on MS SQL server 2005 and later.

TABLESAMPLE will return data from random pages instead of random rows and therefore does not even retrieve data that it will not return.

On a very large table I tested

```
select top 1 percent * from [tablename] order by newid()
```

took more than 20 minutes.

```
select * from [tablename] tablesample(1 percent)
```

took 2 minutes.

Performance will also improve on smaller samples in TABLESAMPLE whereas it will not with newid().

Please keep in mind that this is not as random as the newid() method but will give you a decent sampling.

See the [MSDN page](#).

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [random](#) ([Next Q](#))

Q: Foreign key constraint may cause cycles or multiple cascade paths?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I have a problem when I try to add constraints to my tables. I get the error:

Introducing FOREIGN KEY constraint 'FK74988DB24B3C886' on table 'Employee' may cause cycles or multiple cascade paths. Specify ON DELETE NO ACTION or ON UPDATE NO ACTION, or modify other FOREIGN KEY constraints.

My constraint is between a Code table and an employee table. The Code table contains Id, Name, FriendlyName, Type and a Value. The employee has a number of fields that reference codes, so that there can be a reference for each type of code.

I need for the fields to be set to null if the code that is referenced is deleted.

Any ideas how I can do this?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: christian-nielsen

[Answer](#) by [onedaywhen](#)

SQL Server does simple counting of cascade paths and, rather than trying to work out whether any cycles actually exist, it assumes the worst and refuses to create the referential actions (CASCADE): you can and should still create the constraints without the referential actions. If you can't alter your design (or doing so would compromise things) then you should consider using triggers as a last resort.

FWIW resolving cascade paths is a complex problem. Other SQL products will simply ignore the problem and allow you to create cycles, in which case it will be a race to see which will overwrite the value last, probably to the ignorance of the designer (e.g. ACE/Jet does this). I understand some SQL products will attempt to resolve simple cases. Fact remains, SQL Server doesn't even try, plays it ultra safe by disallowing more than one path and at least it tells you so.

[Answer](#) by [hans-riesebos](#)

A typical situation with multiple cascading paths will be this: A master table with two details, let's say "Master" and "Detail1" and "Detail2". Both details are cascade delete. So far no problems. But what if both details have a one-to-many-relation with some other table (say "SomeOtherTable"). SomeOtherTable has a Detail1ID-column AND a Detail2ID-column.

```
Master { ID, masterfields }
```

```
Detail1 { ID, MasterID, detail1fields }

Detail2 { ID, MasterID, detail2fields }

SomeOtherTable {ID, Detail1ID, Detail2ID, someothertablefields }
```

In other words: some of the records in SomeOtherTable are linked with Detail1-records and some of the records in SomeOtherTable are linked with Detail2 records. Even if it is guaranteed that SomeOtherTable-records never belong to both Details, it is now impossible to make SomeOtherTable's records cascade delete for both details, because there are multiple cascading paths from Master to SomeOtherTable (one via Detail1 and one via Detail2). Now you may already have understood this. Here is a possible solution:

```
Master { ID, masterfields }

DetailMain { ID, MasterID }

Detail1 { DetailMainID, detail1fields }

Detail2 { DetailMainID, detail2fields }

SomeOtherTable {ID, DetailMainID, someothertablefields }
```

All ID fields are key-fields and auto-increment. The crux lies in the DetailMainId fields of the Detail tables. These fields are both key and referential constraint. It is now possible to cascade delete everything by only deleting master-records. The downside is that for each detail1-record AND for each detail2 record, there must also be a DetailMain-record (which is actually created first to get the correct and unique id).

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Q: How to connect to SQL Server database from JavaScript in the browser?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Can anybody give me some sample source code showing how to connect to a SQL Server 2005 database from JavaScript locally? I am learning web programming on my desktop.

Or do I need to use any other scripting language? Suggest some alternatives if you have them, but I am now trying to do it with JavaScript. My SQL Server is locally installed on my desktop — SQL Server Management Studio 2005 and IE7 browser.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [enjoy-coding](#) 

[Answer](#)  by [fbinder](#) 

You shouldn't use client javascript to access databases for several reasons (bad practice, security issues, etc) but if you really want to do this, here is an example:

[Skip code block](#)

```
var connection = new ActiveXObject("ADODB.Connection") ;

var connectionstring="Data Source=<server>;Initial Catalog=<catalog>;User ID=<user>;Password=<password>";
```

```

connection.Open(connectionstring);
var rs = new ActiveXObject("ADODB.Recordset");

rs.Open("SELECT * FROM table", connection);
rs.MoveFirst
while(!rs.eof)
{
    document.write(rs.fields(1));
    rs.movenext;
}

rs.close;
connection.close;

```

A better way to connect to a sql server would be to use some server side language like PHP, Java, .NET, among others. Client javascript should be used only for the interfaces.

And there are rumors of an ancient legend about the existence of server javascript, but this is another story. ;)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to avoid the “divide by zero” error in SQL?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Next Q](#))

I have this error message:

```

Msg 8134, Level 16, State 1, Line 1
Divide by zero error encountered.

```

What is the best way to write SQL code so that I will never see this error message again?

I could do either of the following:

- Add a where clause so that my divisor is never zero

Or

- I could add a case statement, so that there is a special treatment for zero.

Is the best way to use a NullIf clause?

Is there better way, or how can this be enforced?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Next Q](#))

User: [henrik-staun-poulsen](#) 

[Answer](#)  by [henrik-staun-poulsen](#) 

In order to avoid a “Division by zero” error we have programmed it like this:

```

Select Case when divisor=0 then null
Else dividend / divisor
End ,,

```

But here is a much nicer way of doing it:

```
Select dividend / nullif(divisor, 0) ...
```

Now the only problem is to remember the NullIf bit, if I use the “/” key.

[Answer](#) by [tobias-domhan](#)

In case you want to return zero, in case a zero devision would happen, you can use:

```
SELECT COALESCE(dividend / NULLIF(divisor,0), 0) FROM sometable
```

For every divisor that is zero, you will get a zero in the result set.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Next Q](#))

[Q: How do I execute a stored procedure once for each row returned by query?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I have a stored procedure that alters user data in a certain way. I pass it user_id and it does it's thing. I want to run a query on a table and then for each user_id I find run the stored procedure once on that user_id

How would I write query for this?

SQL SERVER

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [iosamurai](#)

[Answer](#) by [steven-a.-lowe](#)

use a cursor

ADDENDUM: [MS SQL cursor example]

[Skip code block](#)

```
declare @field1 int
declare @field2 int
declare cur CURSOR LOCAL for
    select field1, field2 from sometable where someotherfield is null

open cur

fetch next from cur into @field1, @field2

while @@FETCH_STATUS = 0 BEGIN

    --execute your sproc on each row
    exec uspYourSproc @field1, @field2

    fetch next from cur into @field1, @field2
END

close cur
deallocate cur
```

in MS SQL, [here's an example article](#)

note that cursors are slower than set-based operations, but faster than manual while-loops; more details [in this SO question](#)

ADDENDUM 2: if you will be processing more than just a few records, pull them into a temp table first and run the cursor over the temp table; this will prevent SQL from escalating into table-locks and speed up operation

ADDENDUM 3: and of course, if you can inline whatever your stored procedure is doing to each user ID and run the whole thing as a single SQL update statement, that would be optimal

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: SELECT DISTINCT on one column](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Next Q](#))

Using SQL Server, I have...

```
ID  SKU PRODUCT
=====
1   FOO-23 Orange
2   BAR-23 Orange
3   FOO-24 Apple
4   FOO-25 Orange
```

I want

```
1   FOO-23 Orange
3   FOO-24 Apple
```

This query isn't getting me there. How can I SELECT DISTINCT on just one column?

```
SELECT
[ID], [SKU], [PRODUCT]
FROM [TestData]
WHERE ([PRODUCT] =
(SELECT DISTINCT [PRODUCT] FROM [TestData] WHERE ([SKU] LIKE 'FOO-%'))
ORDER BY [ID]
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Next Q](#))

User: [mmcglynn](#)

[Answer](#) by [aaron-alton](#)

Assuming that you're on SQL Server 2005 or greater, you can use a CTE with ROW_NUMBER():

```
SELECT *
FROM   (SELECT ID, SKU, Product,
              ROW_NUMBER() OVER (PARTITION BY Product ORDER BY ID) AS RowNumber
        FROM   MyTable
        WHERE   SKU LIKE 'FOO%') AS a
WHERE   a.RowNumber = 1
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Next Q](#))

Q: LIMIT 10..20 in SQL Server

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I'm trying to do something like :

```
SELECT * FROM table LIMIT 10,20
```

or

```
SELECT * FROM table LIMIT 10 OFFSET 10
```

but using SQL Server

The only [solution I found](#) looks like overkill:

```
SELECT * FROM (
    SELECT *, ROW_NUMBER() OVER (ORDER BY name) as row FROM sys.databases
) a WHERE row > 5 and row <= 10
```

I also [found](#):

```
SELECT TOP 10 * FROM stuff;
```

... but it's not what I want to do since I can't specify the starting limit.

Is there another way for me to do that ?

Also, just curious, is there a reason why doesn't SQL Server support the `LIMIT` function or something similar? I don't want to be mean, but that really sounds like something a DBMS needs ... If it does, then I'm sorry for being so ignorant! I've been working with MySQL and SQL+ for the past 5 years so...

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [marcgg](#)

[Answer](#) by [bill-karwin](#)

The `LIMIT` clause is not part of standard SQL. It's supported as a vendor extension to SQL by MySQL, PostgreSQL, and SQLite.

Other brands of database may have similar features (e.g. `TOP` in Microsoft SQL Server), but these don't always work identically.

It's hard to use `TOP` in Microsoft SQL Server to mimic the `LIMIT` clause. There are cases where it just doesn't work.

The solution you showed, using `ROW_NUMBER()` is available in Microsoft SQL Server 2005 and later. This is the best solution (for now) that works solely as part of the query.

Another solution is to use `TOP` to fetch the first `count + offset` rows, and then use the API to seek past the first `offset` rows.

See also:

- “[Emulate MySQL LIMIT clause in Microsoft SQL Server 2000](#)”

- “[Paging of Large Resultsets in ASP.NET](#)”
-

[Answer](#) by [martin-smith](#)

For SQL Server 2012 + [you can use](#).

```
SELECT *  
FROM sys.databases  
ORDER BY name  
OFFSET 5 ROWS  
FETCH NEXT 5 ROWS ONLY
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL update query using joins](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I have to update a field with a value which is returned by a join of 3 tables.

Example:

[Skip code block](#)

```
select  
    im.itemid  
    ,im.sku as iSKU  
    ,gm.SKU as GSku  
    ,mm.ManufacturerId as ManuID  
    ,mm.ManufacturerName  
    ,im.mf_item_number  
    ,mm.ManufacturerID  
from  
    item_master im, group_master gm, Manufacturer_Master mm  
where  
    im.mf_item_number like 'STA%'  
    and im.sku=gm.sku  
    and gm.ManufacturerID = mm.ManufacturerID  
    and gm.manufacturerID=34
```

I want to update the `mf_item_number` field values of table `item_master` with some other value which is joined in the above condition.

How can I do this in MS SQL Server?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [shyju](#)

[Answer](#) by [gbn](#)

Edit: changed to ANSI joins

[Skip code block](#)

```
UPDATE  
    im  
SET  
    mf_item_number = gm.SKU --etc  
FROM  
    item_master im
```

```
JOIN
group_master gm ON im.sku=gm.sku
JOIN
Manufacturer_Master mm ON gm.ManufacturerID=mm.ManufacturerID
WHERE
im.mf_item_number like 'STA%'
AND
gm.manufacturerID=34
```

Edit, Nov 2011

To make it clear... The UPDATE clause can refer to an table alias specified in the FROM clause. So `im` in this case is valid

Edit, Aug 2012

Generic example

[Skip code block](#)

```
UPDATE
A
SET
foo = B.bar
FROM
TableA A
JOIN
TableB B ON A.col1 = B.colx
WHERE
...
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: Adding an identity to an existing column](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I need to change the primary key of a table to an identity column, and there's already a number of rows in table.

I've got a script to clean up the IDs to ensure they're sequential starting at 1, works fine on my test database.

What's the SQL command to alter the column to have an identity property?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [kirschstein](#) 

[Answer](#)  by [john-sansom](#) 

You can't alter the existing columns for identity.

You have 2 options,

1. Create a new table with identity & drop the existing table
2. Create a new column with identity & drop the existing column

Approach 1. (*New table*) Here you can retain the existing data values on the newly created

identity column.

[Skip code block](#)

```
CREATE TABLE dbo.Tmp_Names
(
    Id int NOT NULL
        IDENTITY(1, 1),
    Name varchar(50) NULL
)
ON [PRIMARY]
go

SET IDENTITY_INSERT dbo.Tmp_Names ON
go

IF EXISTS ( SELECT   *
            FROM     dbo.Names )
    INSERT INTO dbo.Tmp_Names ( Id, Name )
        SELECT  Id,
                Name
            FROM    dbo.Names TABLOCKX
go

SET IDENTITY_INSERT dbo.Tmp_Names OFF
go

DROP TABLE dbo.Names
go

Exec sp_rename 'Tmp_Names', 'Names'
```

Approach 2 (*New column*) You can't retain the existing data values on the newly created identity column, The identity column will hold the sequence of number.

```
Alter Table Names
Add Id_new Int Identity(1, 1)
Go

Alter Table Names Drop Column ID
Go

Exec sp_rename 'Names.Id_new', 'ID', 'Column'
```

See the following Microsoft SQL Server Forum post for more details:

[How to alter column to identity\(1,1\)](#)

[Answer](#) by [justin-grant](#)

In SQL 2005 and above, there's a trick to solve this problem without changing the table's data pages. This is important for large tables where touching every data page can take minutes or hours. The trick also works even if the identity column is a primary key, is part of a clustered or non-clustered index, or other gotchas which can trip up the simpler "add/remove/rename column" solution.

Here's the trick: you can use SQL Server's [ALTER TABLE...SWITCH](#) statement to change the schema of a table without changing the data, meaning you can replace a table with an IDENTITY with an identical table schema, but without an IDENTITY column. The same trick works to add IDENTITY to an existing column.

Normally, [ALTER TABLE...SWITCH](#) is used to efficiently replace a full partition in a partitioned table with a new, empty partition. But it can also be used in non-partitioned tables too.

I've used this trick to convert, in under 5 seconds, a column of a 2.5 billion row table from IDENTITY to a non-IDENTITY (in order to run a multi-hour query whose query plan worked better for non-IDENTITY columns), and then restored the IDENTITY setting, again in less than 5 seconds.

Here's a code sample of how it works.

[Skip code block](#)

```
CREATE TABLE Test
(
    id int identity(1,1),
    somecolumn varchar(10)
);

INSERT INTO Test VALUES ('Hello');
INSERT INTO Test VALUES ('World');

--copy the table. use same schema, but no identity
CREATE TABLE Test2
(
    id int NOT NULL,
    somecolumn varchar(10)
);

ALTER TABLE Test SWITCH TO Test2;

--drop the original (now empty) table
DROP TABLE Test;

--rename new table to old table's name
EXEC sp_rename 'Test2', 'Test';

--see same records
SELECT * FROM Test;
```

This is obviously more involved than the solutions in other answers, but if your table is large this can be a real life-saver. There are some caveats:

- you'll need to drop foreign keys before you do the switch and restore them after.
- same for WITH SCHEMABINDING functions, views, etc.
- new table's indexes need to match exactly (same columns, same order, etc.)
- old and new tables need to be on the same filegroup.
- only works on SQL Server 2005 or later
- I previously believed that this trick only works on the Enterprise or Developer editions of SQL Server (because partitions are only supported in Enterprise and Developer versions), but Mason G. Zhwiti in his comment below says that it also works in SQL Standard Edition too. I assume this means that the restriction to Enterprise or Developer doesn't apply to ALTER TABLE...SWITCH.

There's a good [article on TechNet](#) detailing the requirements above.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I get column names from a table in SQL Server?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I would like to query the name of all columns of a table. I found how to do this in:

- [Oracle](#)
- [MySQL](#)
- [PostgreSQL](#)

But I need to know: **how can this be done in Microsoft SQL Server (2008 in my case)?**

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [odiseh](#)

[Answer](#) by anon

You can obtain this information and much, much more by querying the [Information Schema views](#).

This sample query:

```
SELECT *
FROM Northwind.INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = N'Customers'
```

Can be made over all these DB objects:

- CHECK_CONSTRAINTS
- COLUMN_DOMAIN_USAGE - COLUMN_PRIVILEGES
- COLUMNS
- CONSTRAINT_COLUMN_USAGE
- CONSTRAINT_TABLE_USAGE
- DOMAIN_CONSTRAINTS
- DOMAINS
- KEY_COLUMN_USAGE
- PARAMETERS
- REFERENTIAL_CONSTRAINTS
- ROUTINES
- ROUTINE_COLUMNS
- SCHEMATA
- TABLE_CONSTRAINTS
- TABLE_PRIVILEGES
- TABLES
- VIEW_COLUMN_USAGE
- VIEW_TABLE_USAGE
- VIEWS

[Answer](#) by [arnkrishn](#)

You can use the stored procedure `sp_columns` which would return information pertaining to all columns for a given table. More info can be found here

<http://msdn.microsoft.com/en-us/library/ms176077.aspx>

You can also do it by a SQL query. Some thing like this should help -

```
SELECT * FROM sys.columns WHERE object_id = OBJECT_ID('dbo.yourTableName')
```

Or a variation would be:

```
SELECT      o.Name, c.Name
FROM        sys.columns c
JOIN        sys.objects o ON o.object_id = c.object_id
WHERE       o.type = 'U'
ORDER BY    o.Name, c.Name
```

This gets all columns from all tables, ordered by table name and then on column name.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

Q: Difference between numeric,float and decimal in sql server

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I searched in google and also visited the

[decimal and numeric](#)  and [SQL Server Helper](#) 

to glean the difference between numeric , float and decimal datatypes and also to find out which one should be used in which situation.

For any kind of financial transaction, which one is prefered and why? e.g. for salary field

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [priyanka.sarkar](#) 

[Answer](#)  by [imanabidi](#) 

use the *float* or *real* data types **only if** the precision provided by *decimal* (up to 38 digits) is insufficient

- Approximate numeric data types do not store the exact values specified **for many** numbers; they store **an extremely close** approximation of the value. ([Technet](#) - Avoid using float or real columns in WHERE clause search conditions, especially the = and <> operators ([Technet](#) 

so generally because the *precision provided by decimal* is [10E38 ~ 38 digits] if your number can fit in it, and smaller storage space (and maybe speed) of Float is not important and dealing with abnormal behaviors and issues of approximate numeric types are not acceptable, **use Decimal generally.**

more useful information

- numeric = decimal (5 to 17 bytes) (**Exact** Numeric Data Type)
 - will map to Decimal in .NET
 - both have (18, 0) as default (precision,scale) parameters in SQL server
 - scale = maximum number of decimal digits that can be stored to the right of the decimal point.
 - kindly note that money(8 byte) and smallmoney(4 byte) are also exact and map to Decimal In .NET and have 4 decimal points([MSDN](#) 

- [decimal and numeric \(Transact-SQL\) - MSDN](#)
- real (4 byte) (**Approximate** Numeric Data Type)
 - will map to Single in .NET
 - The ISO synonym for real is float(24)
 - [float and real \(Transact-SQL\) - MSDN](#)
- float (8 byte) (**Approximate** Numeric Data Type)
 - will map to Double in .NET
- All **exact** numeric types always produce the same result, regardless of which kind of processor architecture is being used **or the magnitude of the numbers**
- The parameter supplied to the float data type defines the number of bits that are used to store the **mantissa of the floating point number**.
- Approximate Numeric Data Type usually uses less storage and have better speed (up to 20x) and you should also consider when they got converted in .NET
 - [What is the difference between Decimal, Float and Double in C#](#)
 - [Decimal vs Double Speed](#)
 - [SQL Server - .NET Data Type Mappings \(From MSDN\)](#)

TABLE 3-1 Exact Numeric Data Types

DATA TYPE	STORAGE SIZE	POSSIBLE VALUES	COMMENTS
<i>tinyint</i>	1 byte	0 to 255	Equal to the <i>byte</i> data type in most programming languages, cannot store negative values
<i>smallint</i>	2 bytes	-32768 to 32767	A signed 16-bit integer
<i>int</i>	4 bytes	-2,147,483,648 to 2,147,483,647	A signed 32-bit integer
<i>bigint</i>	8 bytes	-2E63 to 2E63 – 1	A signed 64-bit integer
<i>decimal</i> (precision, scale)	5 to 17 bytes depending on precision	-10E38 + 1 to 10E38 – 1	A decimal number containing up to 38 digits
<i>numeric</i> (precision, scale)	Functionally equivalent to the <i>decimal</i> data type		

TABLE 3-3 Approximate Numeric Data Types

DATA TYPE	STORAGE SIZE	POSSIBLE VALUES
<i>float</i> (n <= 24)	4 bytes	-3.40E38 to -1.18E-38, 0 and 1.18E-38 to 3.40E38
<i>float</i> (24 > n <= 53)	8 bytes	-1.79E308 to -2.23E-308, 0 and 2.23E-308 to 1.79E308
<i>real</i>	Functionally equivalent to <i>float(24)</i>	

main source : [MCTS Self-Paced Training Kit \(Exam 70-433\): Microsoft® SQL Server® 2008 Database Development](#) - Chapter 3 - Tables , Data Types , and Declarative Data

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: Get day of week in SQL 2005/2008](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

If I have a date 01/01/2009, I want to find out what day it was e.g. Monday, Tuesday, etc...

Is there a built-in function for this in SQL 2005/2008? Or do I need to use an auxiliary table?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

User: aaaa

[Answer](#) by [sqlmenace](#)

Use [DATENAME](#) or [DATEPART](#):

```
SELECT DATENAME(dw, GETDATE())-Friday  
SELECT DATEPART(dw, GETDATE())-6
```

[Answer](#) by [sung](#)

Even though SQLMenace's answer has been accepted, there is one important SET option you should be aware of

[SET DATEFIRST](#)

[DATENAME](#) will return correct date *name* but not the same [DATEPART](#) value if the first day of week has been changed as illustrated below.

[Skip code block](#)

```
declare @DefaultDateFirst int  
set @DefaultDateFirst = @@datefirst  
--; 7 First day of week is "Sunday" by default  
select [@DefaultDateFirst] = @DefaultDateFirst  
  
set datefirst @DefaultDateFirst  
select datename(dw, getdate())-Saturday  
select datepart(dw, getdate())-7  
  
--; Set the first day of week to * TUESDAY *  
--; (some people start their week on Tuesdays...)  
set datefirst 2  
select datename(dw, getdate())-Saturday  
--; Returns 5 because Saturday is the 5th day since Tuesday.  
--; Tue 1, Wed 2, Th 3, Fri 4, Sat 5  
select datepart(dw, getdate())-5 <-- It's not 7!  
set datefirst @DefaultDateFirst
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

[Q: Difference between a User and a Login in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I have recently been running into many different areas of SQL Server that I normally don't mess with. One of them that has me confused is the area of Logins and Users. Seems like it should be a pretty simple topic...

It appears that each login can only have 1 user and each user can only have 1 login.

A login can be associated to multiple tables thus associating that user to many tables.

So my question is why even have a login and a user? they seem to be pretty much one in the same. What are the differences, or what is it that I seem to be missing?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [corymathews](#) 

[Answer](#)  by [scott-ivey](#) 

A "Login" grants the principal entry into the SERVER.

A "User" grants a login entry into a single DATABASE.

One "Login" can be associated with many users (one per database).

Each of the above objects can have permissions granted to it at its own level. See the following articles for an explanation of each

- [Principals](#) 
 - [Database Users](#) 
-

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[**Q: When should I use Cross Apply over Inner Join?**](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

What is the main purpose of using [CROSS APPLY](#) ?

I have read (vaguely, through posts on the Internet) that cross apply can be more efficient when selecting over large data sets if you are partitioning. (Paging comes to mind)

I also know that [CROSS APPLY doesn't require a UDF as the right-table.](#) 

In most `INNER JOIN` queries (one-to-many relationships), I could rewrite them to use `CROSS APPLY`, but they always give me equivalent execution plans.

Can anyone give me a good example of when `CROSS APPLY` makes a difference in those cases where `INNER JOIN` will work as well?

Edit:

Here's a trivial example, where the execution plans are exactly the same. (Show me one

where they differ and where cross apply is faster/more efficient)

Skip code block

```
create table Company (
    companyId int identity(1,1)
,   companyName varchar(100)
,   zipcode varchar(10)
,   constraint PK_Company primary key (companyId)
)
GO

create table Person (
    personId int identity(1,1)
,   personName varchar(100)
,   companyId int
,   constraint FK_Person_CompanyId foreign key (companyId) references dbo.Company(companyId)
,   constraint PK_Person primary key (personId)
)
GO

insert Company
select 'ABC Company', '19808' union
select 'XYZ Company', '08534' union
select '123 Company', '10016'

insert Person
select 'Alan', 1 union
select 'Bobby', 1 union
select 'Chris', 1 union
select 'Xavier', 2 union
select 'Yoshi', 2 union
select 'Zambrano', 2 union
select 'Player 1', 3 union
select 'Player 2', 3 union
select 'Player 3', 3

/* using CROSS APPLY */
select *
from Person p
cross apply (
    select *
    from Company c
    where p.companyid = c.companyId
) Czip

/* the equivalent query using INNER JOIN */
select *
from Person p
inner join Company c on p.companyid = c.companyId
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

User: [jeff-meatball-yang](#) 

[Answer](#)  by [quassnoi](#) 

Can anyone give me a good example of when CROSS APPLY makes a difference in those cases where INNER JOIN will work as well?

See the article in my blog for detailed performance comparison:

- [INNER JOIN vs. CROSS APPLY](#) 

CROSS APPLY works better on things that have no simple JOIN condition.

This one selects 3 last records from t2 for each record from t1:

[Skip code block](#)

```
SELECT t1.*, t2o.*  
FROM t1  
CROSS APPLY  
(  
    SELECT TOP 3 *  
    FROM t2  
    WHERE t2.t1_id = t1.id  
    ORDER BY  
        t2.rank DESC  
) t2o
```

It cannot be easily formulated with an INNER JOIN condition.

You could probably do something like that using CTE's and window function:

[Skip code block](#)

```
WITH t2o AS  
(  
    SELECT t2.*, ROW_NUMBER() OVER (PARTITION BY t1_id ORDER BY rank) AS rn  
    FROM t2  
)  
SELECT t1.*, t2o.*  
FROM t1  
INNER JOIN  
    t2o  
ON t2o.t1_id = t1.id  
AND t2o.rn <= 3
```

, but this is less readable and probably less efficient.

Update:

Just checked.

master is a table of about 20,000,000 records with a PRIMARY KEY on id.

This query:

[Skip code block](#)

```
WITH q AS  
(  
    SELECT *, ROW_NUMBER() OVER (ORDER BY id) AS rn  
    FROM master  
)  
, t AS  
(  
    SELECT 1 AS id  
    UNION ALL  
    SELECT 2  
)  
SELECT *  
FROM t  
JOIN q  
ON q.rn <= t.id
```

runs for almost 30 seconds, while this one:

[Skip code block](#)

```
WITH t AS  
(  
    SELECT 1 AS id  
    UNION ALL  
    SELECT 2  
)  
SELECT *  
FROM t  
CROSS APPLY
```

```

(
SELECT TOP (t.id) m.*
FROM master m
ORDER BY
      id
) q

```

is instant.

[Answer](#) by [nurettin](#)

cross apply sometimes enables you to do things that you cannot do with inner join.

Example (a syntax error):

```

select F.* from sys.objects O
inner join dbo.myTableFun(O.name) F
on F.schema_id= O.schema_id

```

This is a **syntax error**, because table functions can only take variables or constants as parameters when using inner join.

However:

```

select F.* from sys.objects O
cross apply ( select * from dbo.myTableFun(O.name) ) F
where F.schema_id= O.schema_id

```

This is legal.

Edit: Or alternatively, shorter syntax: (by ErikE)

```

select F.* from sys.objects O
cross apply dbo.myTableFun(O.name) F
where F.schema_id= O.schema_id

```

Edit:

Note: Informix 12.10 xC2+ has [Lateral Derived Tables](#) and Postgresql (9.3+) has [Lateral Subqueries](#) which can be used to a similar effect.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL to determine minimum sequential days of access?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

The following User History table contains **one record for every day a given user has accessed a website** (in a 24 hour UTC period). It has many thousands of records, but only one record per day per user. If the user has not accessed the website for that day, no record will be generated.

Id	UserId	CreationDate
750997	12	2009-07-07 18:42:20.723
750998	15	2009-07-07 18:42:20.927
751000	19	2009-07-07 18:42:22.283

What I'm looking for is a SQL query on this table *with good performance*, that tells me

which userids have accessed the website for (n) continuous days without missing a day.

In other words, **how many users have (n) records in this table with sequential (day-before, or day-after) dates?** If any day is missing from the sequence, the sequence is broken and should restart again at 1; we're looking for users who have achieved a continuous number of days here with no gaps.

Any resemblance between this query and [a particular Stack Overflow badge](#) is purely coincidental, of course.. :)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [jeff-atwood](#)

[Answer](#) by [spencer-rupert](#)

The answer is obviously:

```
SELECT DISTINCT UserId
FROM UserHistory uh1
WHERE (
    SELECT COUNT(*)
    FROM UserHistory uh2
    WHERE uh2.CreationDate
        BETWEEN uh1.CreationDate AND DATEADD(d, @days, uh1.CreationDate)
) = @days OR UserId = 52551
```

EDIT:

Okay here's my serious answer:

[Skip code block](#)

```
DECLARE @days int
DECLARE @seconds bigint
SET @days = 30
SET @seconds = (@days * 24 * 60 * 60) - 1
SELECT DISTINCT UserId
FROM (
    SELECT uh1.UserId, Count(uh1.Id) as Conseq
    FROM UserHistory uh1
    INNER JOIN UserHistory uh2 ON uh2.CreationDate
        BETWEEN uh1.CreationDate AND
            DATEADD(s, @seconds, DATEADD(dd, DATEDIFF(dd, 0, uh1.CreationDate), 0))
    AND uh1.UserId = uh2.UserId
    GROUP BY uh1.Id, uh1.UserId
) as Tbl
WHERE Conseq >= @days
```

EDIT:

[Jeff Atwood] This is a great fast solution and deserves to be accepted, but [Rob Farley's solution is also excellent](#) and arguably even faster (!). Please check it out too!

[Answer](#) by [rob-farley](#)

How about (and please make sure the previous statement ended with a semi-colon):

[Skip code block](#)

```
WITH numberedrows
AS (SELECT ROW_NUMBER() OVER (PARTITION BY UserID
                                ORDER BY CreationDate)
    - DATEDIFF(day, '19000101', CreationDate) AS TheOffset,
    CreationDate,
    UserID
    FROM tablename)
```

```
SELECT MIN(CreationDate),
       MAX(CreationDate),
       COUNT(*) AS NumConsecutiveDays,
       UserID
  FROM numberedrows
 GROUP BY UserID,
          TheOffset
```

The idea being that if we have list of the days (as a number), and a row_number, then missed days make the offset between these two lists slightly bigger. So we're looking for a range that has a consistent offset.

You could use “ORDER BY NumConsecutiveDays DESC” at the end of this, or say “HAVING count(*) > 14” for a threshold...

I haven't tested this though - just writing it off the top of my head. Hopefully works in SQL2005 and on.

...and would be very much helped by an index on tablename(UserID, CreationDate)

Edited: Turns out Offset is a reserved word, so I used TheOffset instead.

Edited: The suggestion to use COUNT(*) is very valid - I should've done that in the first place but wasn't really thinking. Previously it was using datediff(day, min(CreationDate), max(CreationDate)) instead.

Rob

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: Best approach to remove time part of datetime in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

Which method provides the best performance when removing the time portion from a datetime field in SQL Server?

```
a) select DATEADD(dd, DATEDIFF(dd, 0, getdate()), 0)
```

or

```
b) select cast(convert(char(11), getdate(), 113) as datetime)
```

The second method does send a few more bytes either way but that might not be as important as the speed of the conversion.

Both also appear to be very fast, but there might be a difference in speed when dealing with hundreds-of-thousands or more rows?

Also, is it possible that there are even better methods to get rid of the time portion of a datetime in SQL?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

User: [stephen-perelson](#) 

[Answer](#)  by [gbn](#) 

Strictly, method a is the least resource intensive:

a) `select DATEADD(dd, DATEDIFF(dd, 0, getdate()), 0)`

Proven less CPU intensive for same total duration a million rows by some one with way too much time on their hands: [Most efficient way in SQL Server to get date from date+time?](#)

I saw a similar test elsewhere with similar results too.

I prefer the DATEADD/DATEDIFF because:

- varchar is subject to language/dateformat issues
Example: [Why is my CASE expression non-deterministic?](#)
- float relies on internal storage
- it extends to work out first day of month, tomorrow etc by changing “0” base

Edit, Oct 2011

For SQL Server 2008+, you can CAST to date. Or just use date so no time to remove.

Edit, Jan 2012

A worked example of how flexible this is: [Need to calculate by rounded time or date figure in sql server](#)

Edit, May 2012

Do not use this in WHERE clauses and the like without thinking: adding a function or CAST to a column invalidates index usage. See number 2 here: <http://www.simple-talk.com/sql/t-sql-programming/ten-common-sql-programming-mistakes/>

Now, this does have an example of later SQL Server optimiser versions managing CAST to date correctly, but *generally* it will be a bad idea ...

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

Q: Function vs. Stored Procedure in SQL Server

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I've been learning Functions and Stored Procedure for quite a while but I don't know why and when I should use a function or a stored procedure. They look same to me, maybe because I am kinda newbie about that.

Can some one tell me why?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [tarik](#)

[Answer](#) by [myitchychin](#)

Functions are computed values and cannot perform permanent environmental changes to

SQL Server (i.e. no INSERT or UPDATE statements allowed).

A Function can be used inline in SQL Statements if it returns a scalar value or can be joined upon if it returns a result set.

[Answer](#)  by [chris-j](#) 

Functions and stored procedures serve separate purposes. Although it's not the best analogy, functions can be viewed literally as any other function you'd use in any programming language, but stored procs are more like individual programs or a batch script.

Functions normally have an output and optionally inputs. The output can then be used as the input to another function (a SQL Server built-in such as DATEDIFF, LEN, etc) or as a predicate to a SQL Query - e.g., `SELECT a, b, dbo.MyFunction(c) FROM table` or `SELECT a, b, c FROM table WHERE a = dbo.MyFunc(c)`.

Stored procs are used to bind SQL queries together in a transaction, and interface with the outside world. Frameworks such as ADO.NET, etc. can't call a function directly, but they can call a stored proc directly.

Functions do have a hidden danger though: they can be misused and cause rather nasty performance issues: consider this query:

```
SELECT * FROM dbo.MyTable WHERE col1 = dbo.MyFunction(col2)
```

Where MyFunction is declared as:

[Skip code block](#)

```
CREATE FUNCTION MyFunction (@someValue INTEGER) RETURNS INTEGER
AS
BEGIN
    DECLARE @retval INTEGER

    SELECT localValue
        FROM dbo.localToNationalMapTable
        WHERE nationalValue = @someValue

    RETURN @retval
END
```

What happens here is that the function MyFunction is called for every row in the table MyTable. If MyTable has 1000 rows, then that's another 1000 ad-hoc queries against the database. Similarly, if the function is called when specified in the column spec, then the function will be called for each row returned by the SELECT.

So you do need to be careful writing functions. If you do SELECT from a table in a function, you need to ask yourself whether it can be better performed with a JOIN in the parent stored proc or some other SQL construct (such as CASE ... WHEN ... ELSE ... END).

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Q: how can I Update top 100 records in sql server 

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I want to update Top 100 records in sql server . I have a table T1 with fields F1 and F2. T1 has 200 records. I want to update F1 field of Top 100 records. How can i can update in sql server.

Thanks

Based on comments there is a where clause that prevents re-processing the same records.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [rajesh-](#) 

[Answer](#)  by [umair-ahmed](#) 

Note, the parentheses are required for UPDATE statements:

```
update top (100) table1 set field1 = 1
```

[Answer](#)  by [martin-smith](#) 

Without an ORDER BY the whole idea of TOP doesn't make much sense. You need to have a consistent definition of which direction is "up" and which is "down" for the concept of top to be meaningful.

Nonetheless SQL Server allows it but [doesn't guarantee a deterministic result](#) .

The UPDATE TOP syntax in the accepted answer does not support an ORDER BY clause but it is possible to get deterministic semantics here by using a CTE or derived table to define the desired sort order as below.

```
;WITH CTE AS
(
SELECT TOP 100 *
FROM T1
ORDER BY F2
)
UPDATE CTE SET F1='foo'
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: What represents a double in sql server?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#))

I have a couple of properties in c# which are double and I want to store these in a table in SQL Server, but noticed there is no double type, so what is best to use, decimal or float?

This will store latitude and longitude values, so I need the most accurate precision.

Thanks for the responses so far.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#))

User: [xaisoft](#) 

[Answer](#)  by [richardtallent](#) 

float

Or if you want to go old-school:

real

You can also use float(53), but it means the same thing as float.

("real" is equivalent to float(24), not float/float(53).)

The **decimal(x,y)** SQL Server type is for when you want *exact* decimal numbers rather than floating point (which can be approximations). This is in contrast to the C# "decimal" data type, which is more like a 128-bit floating point number.

MSSQL **float** does not have *exactly* the same precision as the 64-bit **double** type in .NET (slight difference in mantissa IIRC), but it's a close enough match most uses.

To make things more confusing, a "float" in C# is only 32-bit, so it would be more equivalent in SQL to the real/float(24) type in MSSQL than float/float(53).

In your specific use case... All you need is 5 places after the decimal point to represent latitude and longitude within about one-meter precision, and you only need up to three digits before the decimal point for the degrees. Float(24) or decimal(8,5) will best fit your needs in MSSQL, and using float in C# is good enough, you don't need double. In fact, your users will probably thank you for rounding to 5 decimal places rather than having a bunch of insignificant digits coming along for the ride.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#))

[Q: Count\(*\) vs Count\(1\)](#) 

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

Just wondering if any of you guys use Count(1) over Count(*) and if there is a noticeable difference in performance or if this is just a legacy habit that has been brought forward from days gone past?

(The specific database is SQL Server 2005.)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

User: [super9](#) 

[Answer](#)  by [gbn](#) 

There is no difference.

Reason:

[Books on-line](#) says “COUNT ({ [[ALL | DISTINCT] expression] | * })”

“1” is a non-null expression: so it’s the same as COUNT(*). The optimizer recognizes it for what it is: trivial.

The same as EXISTS (SELECT * ... or EXISTS (SELECT 1...

Example:

```
SELECT COUNT(1) FROM dbo.tab800krows
SELECT COUNT(1),FKID FROM dbo.tab800krows GROUP BY FKID

SELECT COUNT(*) FROM dbo.tab800krows
SELECT COUNT(*),FKID FROM dbo.tab800krows GROUP BY FKID
```

Same IO, same plan, the works

Edit, Aug 2011

[Similar question on DBA.SE](#).

Edit, Dec 2011

COUNT(*) is mentioned specifically in [ANSI-92](#) (look for “Scalar expressions 125”)

Case:

a) If COUNT(*) is specified, then the result is the cardinality of T.

That is, the ANSI standard recognizes it as bleeding obvious what you mean. COUNT(1) has been optimized out by RDBMS vendors *because* of this superstition. Otherwise it would be evaluated as per ANSI

b) Otherwise, let TX be the single-column table that is the result of applying the <value expression> to each row of T and eliminating null values. If one or more null values are eliminated, then a completion condition is raised: warning-

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

[Q: how to see active SQL Server connections?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I am using SQL Server 2008 Enterprise. I want to see any active SQL Server connections, and the related information of all the connections, like from which IP address, connect to which DB or something.

Any existing tools to solve this issue?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [george2](#)

Answer by [mehrdad-afshari](#)

You can use [sp_who](#) stored procedure.

Provides information about current users, sessions, and processes in an instance of the Microsoft SQL Server Database Engine. The information can be filtered to return only those processes that are not idle, that belong to a specific user, or that belong to a specific session.

[Answer](#) by [syed-umar-ahmed](#)

[Skip code block](#)

```
SELECT
    DB_NAME(dbid) as DBName,
    COUNT(dbid) as NumberOfConnections,
    loginname as LoginName
FROM
    sys.sysprocesses
WHERE
    dbid > 0
GROUP BY
    dbid, loginname
;
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

[Q: Why do you create a View in a database?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

When and Why does some one decide that they need to create a View in their database?
Why not just run a normal stored procedure or select?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [medicineman](#)

[Answer](#) by [dave-carlile](#)

A view provides several benefits.

1. Views can hide complexity

If you have a query that requires joining several tables, or has complex logic or calculations, you can code all that logic into a view, then select from the view just like you would a table.

2. Views can be used as a security mechanism

A view can select certain columns and/or rows from a table, and permissions set on the view instead of the underlying tables. This allows surfacing only the data that a user needs to see.

3. Views can simplify supporting legacy code

If you need to refactor a table that would break a lot of code, you can replace the table with a view of the same name. The view provides the exact same schema as the original

table, while the actual schema has changed. This keeps the legacy code that references the table from breaking, allowing you to change the legacy code at your leisure.

These are just some of the many examples of how views can be useful.

[Answer](#)  by [graeme-perrow](#) 

Among other things, it can be used for security. If you have a “customer” table, you might want to give all of your sales people access to the name, address, zipcode, etc. fields, but not credit_card_number. You can create a view that only includes the columns they need access to and then grant them access on the view.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

Q: How can I do an UPDATE statement with JOIN in SQL?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I need to update this table in SQL Server 2005 with data from its ‘parent’ table, see below:

sale

```
id (int)
udid (int)
assid (int)
```

ud

```
id (int)
assid (int)
```

sale.assid contains the correct value to update ud.assid. What query will do this? I’m thinking a join but I’m not sure if it’s possible.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [ant-swift](#)

[Answer](#) by [eric](#)

It very much depends on which database you’re using. Here are the ways to do it in ANSI (aka should work on any database), MySQL, SQL Server, and Oracle. Be advised that the ANSI method will be much slower than the other two methods, but if you’re not using MySQL, SQL Server, or Oracle, it’s the only way to go.

ANSI:

[Skip code block](#)

```
update ud
  set assid = (
    select sale.assid
      from sale
     where sale.udid = ud.id
  )
where exists (
  select *
    from sale
   where sale.udid = ud.id
);
```

MySQL:

```
update ud u
inner join sale s on
  u.id = s.udid
set u.assid = s.assid
```

SQL Server:

```
update u
  set u.assid = s.assid
  from ud u
    inner join sale s on
      u.id = s.udid
```

Oracle:

```
update
  (select
    u.assid as new_assid,
    s.assid as old_assid
   from ud u
    inner join sale s on
      u.id = s.udid) up
set up.new_assid = up.old_assid
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: Function to Calculate Median in Sql Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

According to [MSDN](#) , Median is not available as an aggregate function in Transact-Sql. However, I would like to find out whether it is possible to create this functionality (using the [Create Aggregate](#)  function, user defined function, or some other method).

What would be the best way (if possible) to do this - allow for the calculation of a median value (assuming a numeric data type) in an aggregate query?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [yaakov-ellis](#) 

[Answer](#)  by [justin-grant](#) 

There are lots of ways to do this, with dramatically varying performance. Here's one particularly well-optimized solution, from

http://sqlblog.com/blogs/adam_machanic/archive/2006/12/18/medians-row-numbers-and-performance.aspx . This is a particularly optimal solution when it comes to actual I/Os generated during execution—it looks more costly than other solutions but is actually much faster.

That page also contains a discussion of other solutions and perf testing details. Note the use of a unique column as a disambiguator in case there are multiple rows with the same value of the median column.

As with all database performance scenarios, always try to test a solution out with real data on real hardware—you never know when a change to SQL Server's optimizer or a peculiarity in your environment will make a normally-speedy solution slower.

[Skip code block](#)

```
SELECT
  CustomerId,
  AVG(TotalDue)
FROM
(
  SELECT
    CustomerId,
    TotalDue,
    ROW_NUMBER() OVER (
      PARTITION BY CustomerId
```

```
        ORDER BY TotalDue ASC, SalesOrderId ASC) AS RowAsc,
    ROW_NUMBER() OVER (
        PARTITION BY CustomerId
        ORDER BY TotalDue DESC, SalesOrderId DESC) AS RowDesc
    FROM Sales.SalesOrderHeader SOH
) x
WHERE
    RowAsc IN (RowDesc, RowDesc - 1, RowDesc + 1)
GROUP BY CustomerId
ORDER BY CustomerId;
```

[Answer](#)  by [jeff-atwood](#) 

If you're using SQL 2005 or better this is a nice, simple-ish median calculation for a single column in a table:

```
SELECT
(
    (SELECT MAX(Score) FROM
        (SELECT TOP 50 PERCENT Score FROM Posts ORDER BY Score) AS BottomHalf)
    +
    (SELECT MIN(Score) FROM
        (SELECT TOP 50 PERCENT Score FROM Posts ORDER BY Score DESC) AS TopHalf)
) / 2 AS Median
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: Is it possible to specify condition in Count\(\)?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Is it possible to specify a condition in `Count()`? I would like to count only the rows that have for example "Manager" value in Position column.

Edit: *please* read carefully, I want to do it `IN` count statement, not using `WHERE`; I'm asking about it because I need to count both Managers and Other in the same select (something like `Count(Position = Manager)`, `Count(Poisiont = Other)`) so `WHERE` is no use for me in this example

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [agnieszka](#) 

[Answer](#)  by [guffa](#) 

If you can't just limit the query itself with a `where` clause, you can use the fact that the `count` aggregate only counts the non-null values:

```
select count(case Position when 'Manager' then 1 else null end)
from...
```

You can also use the `sum` aggregate in a similar way:

```
select sum(case Position when 'Manager' then 1 else 0 end)
from...
```

[Answer](#)  by [redfilter](#) 

Assuming you do not want to restrict the rows that are returned because you are

aggregating other values as well, you can do it like this:

```
select count(case when Position = 'Manager' then 1 else null end) as ManagerCount  
from...
```

Let's say within the same column you had values of Manager, Supervisor, and Team Lead, you could get the counts of each like this:

```
select count(case when Position = 'Manager' then 1 else null end) as ManagerCount,  
        count(case when Position = 'Supervisor' then 1 else null end) as SupervisorCount,  
        count(case when Position = 'Team Lead' then 1 else null end) as TeamLeadCount,  
from...
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to drop SQL default constraint without knowing its name?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

In Microsoft SQL Server, I know the query to check if a default constraint exists for a column and drop a default constraint is:

```
IF EXISTS(SELECT * FROM sysconstraints  
        WHERE id=OBJECT_ID('SomeTable')  
        AND COL_NAME(id,colid)='ColName'  
        AND OBJECTPROPERTY(constid, 'IsDefaultCnst')=1)  
ALTER TABLE SomeTable DROP CONSTRAINT DF_SomeTable_ColName
```

But due to typo in previous versions of the database, the name of the constraint could be DF_SomeTable_ColName or DF_SmoetTable_ColName.

How can I delete the default constraint without any SQL errors? Default constraint names don't show up in INFORMATION_SCHEMA table, which makes things a bit trickier.

So, something like 'delete the default constraint in this table/column', or 'delete DF_SmoetTable_ColName', but don't give any errors if it can't find it.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [robo](#)

[Answer](#) by [philip-kelley](#)

Expanding on Mitch Wheat's code, the following script will generate the command to drop the constraint and dynamically execute it.

[Skip code block](#)

```
declare @table_name nvarchar(256)  
declare @col_name nvarchar(256)  
declare @Command nvarchar(1000)  
  
set @table_name = N'Department'  
set @col_name = N'ModifiedDate'  
  
select @Command = 'ALTER TABLE ' + @table_name + ' drop constraint ' + d.name  
from sys.tables t  
join sys.default_constraints d  
on d.parent_object_id = t.object_id  
join sys.columns c  
on c.object_id = t.object_id  
and c.column_id = d.parent_column_id
```

```
where t.name = @table_name
and c.name = @col_name

--print @Command

execute (@Command)
```

[Answer](#) by [mitch-wheat](#)

Rob Farley's blog post might be of help:

- [Two ways to find / drop a default constraint without knowing its name](#)

Something like:

```
declare @table_name nvarchar(256)
declare @col_name nvarchar(256)
set @table_name = N'Department'
set @col_name = N'ModifiedDate'

select t.name, c.name, d.name, d.definition
from
    sys.tables t
    join sys.default_constraints d on d.parent_object_id = t.object_id
    join sys.columns c on c.object_id = t.object_id
                                and c.column_id = d.parent_column_id
where
    t.name = @table_name
    and c.name = @col_name
```

[Answer](#) by [scubasteve](#)

I found that this works and uses no joins:

```
DECLARE @ObjectName NVARCHAR(100)
SELECT @ObjectName = OBJECT_NAME([default_object_id]) FROM SYS.COLUMNS
WHERE [object_id] = OBJECT_ID('[tableSchema].[tableName]') AND [name] = 'columnName';
EXEC( 'ALTER TABLE [tableSchema].[tableName] DROP CONSTRAINT ' + @ObjectName)
```

Just make sure that columnName does not have brackets around it because the query is looking for an exact match and will return nothing if it is [columnName].

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Is the NOLOCK \(Sql Server hint\) bad practice?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I'm in the business of making website and applications that are *not* mission critical -> eg. banking software, space flight, intensive care monitoring application, etc. You get the idea.

So, with that massive disclaimer, is it bad using the NOLOCK hint in some Sql statement? A number of years ago, it was suggested by a fellow Sql Administrator that I should use NOLOCK if I'm happy with a "dirty read" which will give me a bit more performance out of my system because each read doesn't lock the table/row/whatever.

I was also told that it's a great solution if I'm experiencing dead-locks. So, I started

following that thought for a few years until a Sql guru was helping me with some random code and noticed all the NOLOCKS in my sql code. I was politely scolded and he tried to explain it to me (why it's not a good thing) and I sorta got lost. I felt that the essence of his explanation was 'it's a band-aid solution to a more serious problem .. especially if you're experiencing deadlocking. As such, fix the root of the problem'.

I did some googling recently about it and came across [this post](#).

So, can some sql db guru sensei's please enlighten me?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [pure.krome](#)

[Answer](#) by [omg-ponies](#)

With NOLOCK hint, the transaction isolation level for the SELECT statement is READ UNCOMMITTED. This means that the query may see dirty and inconsistent data.

This is not a good idea to apply as a rule. Even if this dirty read behavior is OK for your mission critical web based application, a NOLOCK scan can cause 601 error which will terminate the query due to data movement as a result of lack of locking protection.

I suggest reading [When Snapshot Isolation Helps and When It Hurts](#) - the MSDN recommends using READ COMMITTED SNAPSHOT rather than SNAPSHOT under most circumstances.

[Answer](#) by [geoff-dalgas](#)

Prior to working on Stack Overflow, I was against NOLOCK on the principal that you could potentially perform a SELECT with NOLOCK and get back results with data that may be out of date or inconsistent. A factor to think about is how many records may be inserted/updated at the same time another process may be selecting data from the same table. If this happens a lot then there's a high probability of deadlocks unless you use a database mode such as [READ COMMITTED SNAPSHOT](#).

I have since changed my perspective on the use of NOLOCK after witnessing how it can improve SELECT performance as well as eliminate deadlocks on a massively loaded SQL Server. There are times that you may not care that your data isn't exactly 100% committed and you need results back quickly even though they may be out of date.

Ask yourself a question when thinking of using NOLOCK:

Does my query include a table that has a high number of INSERT/UPDATE commands and do I care if the data returned from a query may be missing these changes at a given moment?

If the answer is no, then use NOLOCK to improve performance.

I just performed a quick search for the NOLOCK keyword within the code base for Stack Overflow and found 138 instances, so we use it in quite a few places.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

[Q: SET NOCOUNT ON usage](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Inspired by [this question](#) where there are differing views on SET NOCOUNT...

Should we use SET NOCOUNT ON for SQL Server? If not, why not?

What it does Edit 6, on 22 Jul 2011

It suppresses the “xx rows affected” message after any DML. This is a resultset and when sent, the client must process it. It’s tiny, but measurable (see answers below)

For triggers etc, the client will receive multiple “xx rows affected” and this causes all manner of errors for some ORMs, MS Access, JPA etc (see edits below)

Background:

General accepted best practice (I thought until this question) is to use `SET NOCOUNT ON` in triggers and stored procedures in SQL Server. We use it everywhere and a quick google shows plenty of SQL Server MVPs agreeing too.

MSDN says this can break a [.net SQLDataAdapter](#).

Now, this means to me that the SQLDataAdapter is limited to utterly simply CRUD processing because it expects the “n rows affected” message to match. So, I can’t use:

- IF EXISTS to avoid duplicates (no rows affected message) *Note: use with caution*
- WHERE NOT EXISTS (less rows than expected)
- Filter out trivial updates (eg no data actually changes)
- Do any table access before (such as logging)
- Hide complexity or denormalisation
- etc

In the question marc_s (who knows his SQL stuff) says do not use it. This differs to what I think (and I regard myself as somewhat competent at SQL too).

It’s possible I’m missing something (feel free to point out the obvious), but what do you folks out there think?

Note: it’s been years since I saw this error because I don’t use SQLDataAdapter nowadays.

Edits after comments and questions:

Edit: More thoughts...

We have multiple clients: one may use a C# SQLDataAdaptor, another may use nHibernate from Java. These can be affected in different ways with `SET NOCOUNT ON`.

If you regard stored procs as methods, then it's bad form (anti-pattern) to assume some internal processing works a certain way for your own purposes.

Edit 2: a [trigger breaking nHibernate question](#), where `SET NOCOUNT ON` can not be set (and no, it's not a duplicate of [this](#))

Edit 3: Yet more info, thanks to my MVP colleague

- [KB 240882](#), issue causing disconnects on SQL 2000 and earlier
- [Demo of performance gain](#)

Edit 4: 13 May 2011

[Breaks Linq 2 SQL too when not specified?](#)

Edit 5: 14 Jun 2011

Breaks JPA, stored proc with table variables: [Does JPA 2.0 support SQL Server table variables?](#)

Edit 6: 15 Aug 2011

The SSMS “Edit rows” data grid requires `SET NOCOUNT ON`: [Update trigger with GROUP BY](#)

Edit 7: 07 Mar 2013

More in depth details from @RemusRusanu:

[Does SET NOCOUNT ON really make that much of a performance difference](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [gbn](#)

[Answer](#) by [sedat-kapanoglu](#)

Ok now I've done my research, here is the deal:

- In TDS protocol [SET NOCOUNT ON only saves 9-bytes per query](#), which doesn't seem that much. I used to think that 9 row(s) affected was returned from server in plain text but it doesn't seem to be the case. It's in fact a small message packet called `DONE_IN_PROC`.
- Microsoft actually encourages the use of `SET NOCOUNT ON` in Stored Procedures, as per the book “[Improving .NET Application Performance and Scalability](#)” (page 541). I would not base my design decisions upon `SqlDataAdapter`.

So I think you can stick with `SET NOCOUNT ON`'s if the cost is less than switching to another technology. I would still consider abandoning `SqlDataAdapter` since you still don't know what kind of design quirk you'll encounter next.

EDIT: @racingsnail pointed out that the network roundtrip delay is a bigger performance killer than the packet size. He has a point but a second network packet wouldn't cause the same delay as the roundtrip latency because the packets would be sent in tandem and

would not require acknowledgement. So it may cause far less delay than the actual network roundtrip latency.

EDIT 2: Here is a very detailed analysis about insignificant overhead of SET NOCOUNT setting: <http://daleburnett.com/2014/01/everything-ever-wanted-know-set-nocount/>

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL Server - SELECT FROM stored procedure](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I have a stored procedure that returns rows:

```
CREATE PROCEDURE MyProc
AS
BEGIN
    SELECT * FROM MyTable
END
```

My actual procedure is a little more complicated, which is why a sproc is necessary.

Is it possible to select the output by calling this procedure?

Something like:

```
SELECT * FROM (EXEC MyProc) AS TEMP
```

I need to use SELECT TOP X, ROW_NUMBER, and an additional WHERE clause to page my data, and I don't really want to pass these values as parameters.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [jonathanpeppers](#) 

[Answer](#)  by [mehrdad-afshari](#) 

You can use a [User-defined function](#)  or a [view](#)  instead of a procedure.

A procedure can return multiple result sets, each with its own schema. It's not suitable for using in a SELECT statement.

[Answer](#)  by [kristof](#) 

You should look at this excellent article by Erland Sommarskog:

- [How to Share Data Between Stored Procedure](#) 

It basically lists all available options for your scenario.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I select the first day of a month in SQL?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

I just need to select the first day of the month of a given datetime variable.

I know it's quite easy to do using this kind of code :

```
select CAST(CAST(YEAR(@mydate) AS VARCHAR(4))  
+ '/' + CAST(MONTH(@mydate) AS VARCHAR(2)) + '/01' AS DATETIME)
```

but this is not very elegant, and probably not very fast either.

Is there a 'better way to do this ? (I'm using SQL Server 2008)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

User: [brann](#) 

[Answer](#)  by [lukeh](#) 

```
SELECT DATEADD(month, DATEDIFF(month, 0, @mydate), 0) AS StartOfMonth
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL Server query - Selecting COUNT\(*\) with DISTINCT](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

In SQL Server 2005 I have a table cm_production that lists all the code that's been put into production. The table has a ticket_number, program_type, and program_name and push_number along with some other columns.

GOAL: Count all the DISTINCT program names by program type and push number

What I have so far is:

```
SELECT DISTINCT COUNT(*) AS Count, program_type AS [Type]  
FROM cm_production  
WHERE push_number=@push_number  
GROUP BY program_type
```

This gets me partway there, but it's counting all the program names, not the distinct ones (which I don't expect it to do in that query). I guess I just can't wrap my head around how to tell it to count only the distinct program names without selecting them. Or something.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [samacore](#) 

[Answer](#)  by [remus-rusanu](#) 

Count all the DISTINCT program names by program type and push number

```
SELECT COUNT(DISTINCT program_name) AS Count,
       program_type AS [Type]
  FROM cm_production
 WHERE push_number=@push_number
 GROUP BY program_type
```

DISTINCT COUNT(*) will return a row for each unique count. What you want is [COUNT\(DISTINCT expression\)](#): evaluates expression for each row in a group and returns the number of unique, nonnull values.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: How do I escape a single quote in SQL Server?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I'm trying to insert some text data into a table in SQL Server 9.

The text includes a single quote.

How do I escape that?

I tried using two single quotes, but it threw me some errors.

eg. `insert into my_table values('hi, my name''s tim.');`

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [tim_wonil](#)

[Answer](#) by [CORY](#)

Single quotes are escaped by doubling them up, just as you've shown us in your example. The following SQL illustrates this functionality. I tested it on SQL Server 2008:

```
DECLARE @my_table TABLE (
    [value] VARCHAR(200)
)

INSERT INTO @my_table VALUES ('hi, my name''s tim.')

SELECT * FROM @my_table
```

Results

```
value
=====
hi, my name's tim.
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL Server - transactions roll back on error?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

We have client app that is running some SQL on a SQL Server 2005 such as the following:

```
BEGIN TRAN;
INSERT INTO myTable (myColumns...) VALUES (myValues...);
INSERT INTO myTable (myColumns...) VALUES (myValues...);
INSERT INTO myTable (myColumns...) VALUES (myValues...);
COMMIT TRAN;
```

It is sent by one long string command.

If one of the inserts fail, or any part of the command fails, does SQL Server roll back the transaction? If it does not rollback, do I have to send a second command to roll it back?

I can give specifics about the api and language I'm using, but I would think SQL Server should respond the same for any language.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [jonathanpeppers](#)

[Answer](#) by [dyingcactus](#)

You can put set xact_abort on before your transaction to make sure sql rolls back automatically in case of error.

[Answer](#) by [raj-more](#)

You are correct in that the entire transaction will be rolled back. You should issue the command to roll it back.

You can wrap this in a TRY CATCH block as follows

[Skip code block](#)

```
BEGIN TRY
    BEGIN TRANSACTION

        INSERT INTO myTable (myColumns...) VALUES (myValues...);
        INSERT INTO myTable (myColumns...) VALUES (myValues...);
        INSERT INTO myTable (myColumns...) VALUES (myValues...);

        COMMIT TRAN--Transaction Success!
    END TRY
    BEGIN CATCH
        IF @@TRANCOUNT > 0
            ROLLBACK TRAN --RollBack in case of Error

        --you can Raise ERROR with RAISEERROR() Statement including the details of the exception
        RAISERROR(ERROR_MESSAGE(), ERROR_SEVERITY(), 1)
    END CATCH
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: IndexOf function in T-SQL](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Given an email address column, I need to find the position of the @ sign for substringing.

What is the `indexof` function, for strings in T-SQL?

Looking for something that returns the position of a substring within a string.

in C#

```
var s = "abcde";
s.IndexOf('c'); // yields 2
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [developingchris](#)

[Answer](#) by [scott-ivey](#)

[CHARINDEX](#) is what you are looking for

```
select CHARINDEX('@', 'someone@somewhere.com')
-----
8
(1 row(s) affected)
```

-or-

```
select CHARINDEX('c', 'abcde')
-----
3
(1 row(s) affected)
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Use variable with TOP in select statement in SQL Server without making it dynamic](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

```
declare @top int
set @top = 5
select top @top * from tablename
```

Is it possible?

Or any idea for such a logic (i don't want to use dynamic query)?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [paresh](#)

[Answer](#) by [guffa](#)

Yes, in SQL Server 2005 it's possible to use a variable in the top clause.

```
select top (@top) * from tablename
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: Saving changes after table edit in SQL Server Management Studio](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [ssms](#) ([Next Q](#))

If I want to save any changes in a table, previously saved in SQL Server Management Studio (no data in table present) I get an error message:

“Saving changes is not permitted. The changes you have made require the following tables to be dropped and re-created. You have either made changes to a table that can't be re-created or enabled the option Prevent saving changes that require the table to be re-created.”

What can prevent the table to be easily edited? Or, is it the usual way for SQL Server Management Studio to require re-creating table for editing? What is it - this “**option Prevent saving changes**”?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [ssms](#) ([Next Q](#))

User: [rem](#)

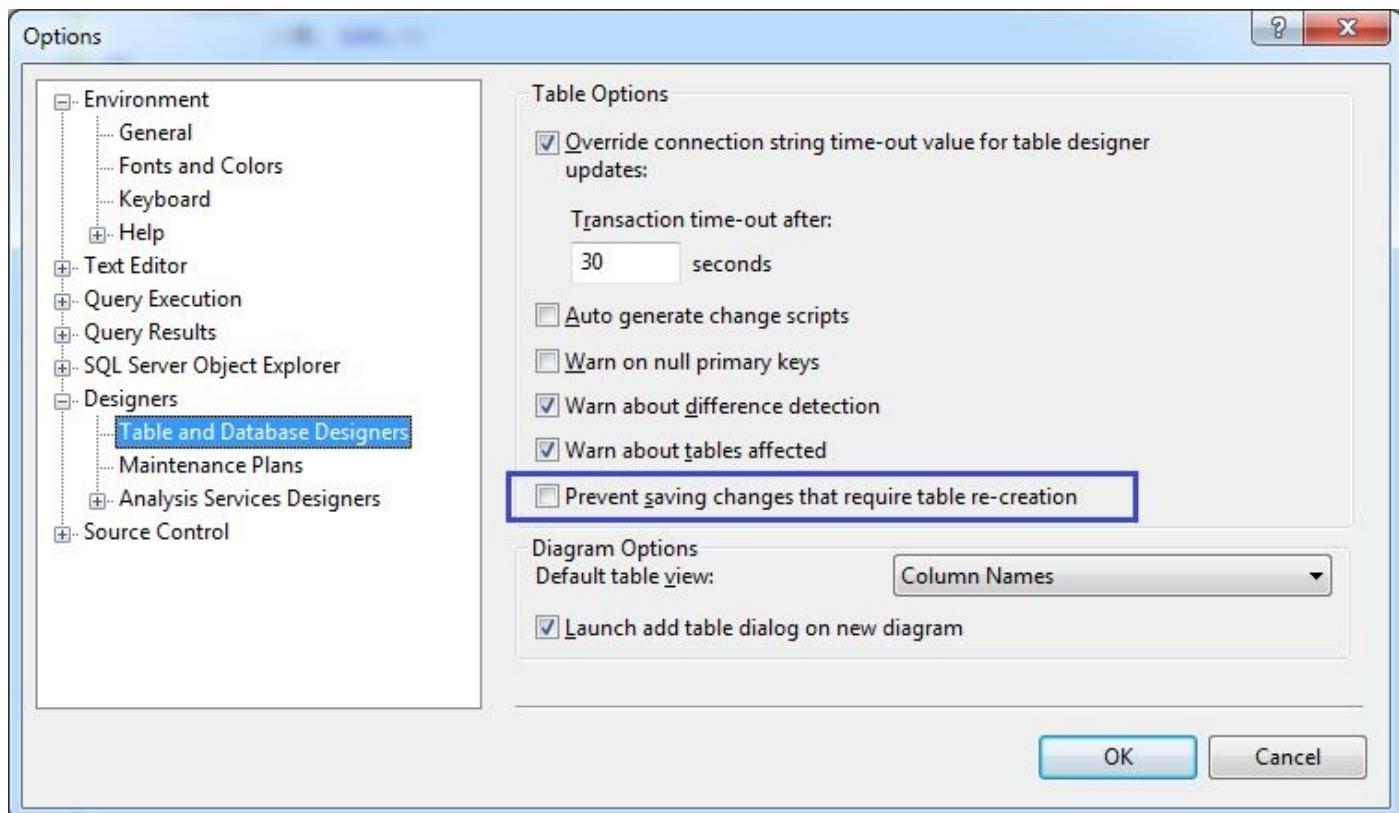
[Answer](#) by [pedro](#)

Go into Tools -> Options -> Designers-> Uncheck “Prevent saving changes that require table re-creation”. Voila.

That happens because sometimes it is necessary to drop and recreate a table in order to change something. This can take a while, since all data must be copied to a temp table and then re-inserted in the new table. Since SQL Server by default doesn't trust you, you need to say “OK, I know what I'm doing, now let me do my work.”

[Answer](#) by [prasanna](#)

Tools>Options



Uncheck above option

Answer by [daniel-vassallo](#)

To work around this problem, use SQL statements to make the changes to the metadata structure of a table.

This problem occurs when “**Prevent saving changes that require table re-creation**” option is enabled.

Source: [Error message when you try to save a table in SQL Server 2008: “Saving changes is not permitted”](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [ssms](#) ([Next Q](#))

[Q: SQL Server: How to Join to first row](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I'll use a concrete, but hypothetical, example.

Each **Order** normally has only one **line item**:

Orders:

OrderGUID	OrderNumber
{FFB2...}	STL-7442-1
{3EC6...}	MPT-9931-8A

LineItems:

LineItemGUID	Order ID	Quantity	Description
--------------	----------	----------	-------------

{098FBE3...}	1	7	prefabulated amulite
{1609B09...}	2	32	spurving bearing

But occasionally there will be an order with two line items:

LineItemID	Order ID	Quantity	Description
{A58A1...}	6,784,329	5	pentametric fan
{0E9BC...}	6,784,329	5	differential girdlespring

Normally when showing the orders to the user:

```
SELECT Orders.OrderNumber, LineItems.Quantity, LineItems.Description
FROM Orders
INNER JOIN LineItems
ON Orders.OrderID = LineItems.OrderID
```

I want to show the single item on the order. But with this occasional order containing two (or more) items, the orders would *appear* be **duplicated**:

OrderNumber	Quantity	Description
STL-7442-1	7	prefabulated amulite
MPT-9931-8A	32	differential girdlespring
KSG-0619-81	5	panametric fan
KSG-0619-81	5	differential girdlespring

What I really want is to have SQL Server *just pick one*, as it will be *good enough*:

OrderNumber	Quantity	Description
STL-7442-1	7	prefabulated amulite
MPT-9931-8A	32	differential girdlespring
KSG-0619-81	5	panametric fan

If I get adventurous, I might show the user, an ellipsis to indicate that there's more than one:

OrderNumber	Quantity	Description
STL-7442-1	7	prefabulated amulite
MPT-9931-8A	32	differential girdlespring
KSG-0619-81	5	panametric fan, ...

So the question is how to either

- eliminate “duplicate” rows
- only join to one of the rows, to avoid duplication

First attempt

My first naive attempt was to only join to the “**TOP 1**” line items:

```
SELECT Orders.OrderNumber, LineItems.Quantity, LineItems.Description
FROM Orders
INNER JOIN (
    SELECT TOP 1 LineItems.Quantity, LineItems.Description
    FROM LineItems
    WHERE LineItems.OrderID = Orders.OrderID) LineItems2
ON 1=1
```

But that gives the error:

The column or prefix ‘Orders’ does not match with a table name or alias name used in the query.

Presumably because the inner select doesn’t see the outer table.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [ian-boyd](#) 

Answer  by [quassnoi](#) 

```
SELECT Orders.OrderNumber, LineItems.Quantity, LineItems.Description
FROM Orders
JOIN LineItems
ON LineItems.LineItemGUID =
(
    SELECT TOP 1 LineItemGUID
    FROM LineItems
    WHERE OrderID = Orders.OrderID
)
```

In SQL Server 2005 and above, you could just replace INNER JOIN with CROSS APPLY:

```
SELECT Orders.OrderNumber, LineItems2.Quantity, LineItems2.Description
FROM Orders
CROSS APPLY
(
    SELECT TOP 1 LineItems.Quantity, LineItems.Description
    FROM LineItems
    WHERE LineItems.OrderID = Orders.OrderID
) LineItems2
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Q: How to check if a stored procedure exists before creating it

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I have a SQL script that has to be run every time a client executes the “database management” functionality. The script includes creating stored procedures on the client database. Some of these clients might already have the stored procedure upon running the script, and some may not. I need to have the missing stored procedures added to the client database, but it doesn’t matter how much I try to bend T-SQL syntax, I get

CREATE/ALTER PROCEDURE’ must be the first statement in a query batch

I’ve read that dropping before creating works, but I don’t like doing it that way.

```
IF EXISTS (SELECT * FROM sys.objects WHERE type = 'P' AND name = 'MyProc')
DROP PROCEDURE MyProc
GO

CREATE PROCEDURE MyProc...
```

How can I add check for the existence of a stored procedure and create it if it doesn’t exist but alter it if it does exist?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [the-shaper](#) 

[Answer](#)  by [quassnoi](#) 

You can run procedural code anywhere you are able to run a query.

Just copy everything after AS:

```
BEGIN
    DECLARE @myvar INT
    SELECT *
    FROM   mytable
    WHERE  @myvar...
END
```

This code does exactly same things a stored proc would do, but is not stored on the database side.

That's much like what is called anonymous procedure in PL/SQL.

Update:

Your question title is a little bit confusing.

If you only need to create a procedure if it not exists, then your code is just fine.

Here's what SSMS outputs in the create script:

```
IF EXISTS ( SELECT *
            FROM   sys.objects
            WHERE  object_id = OBJECT_ID(N'myproc')
                   AND type IN ( N'P', N'PC' ) )
DROP ...
CREATE ...
```

Update:

Example of how to do it when including the schema:

```
IF EXISTS ( SELECT *
            FROM   sysobjects
            WHERE  id = object_id(N'[dbo].[MyProc]')
                   AND OBJECTPROPERTY(id, N'IsProcedure') = 1 )
BEGIN
    DROP PROCEDURE [dbo].[MyProc]
END
```

In the example above, *dbo* is the schema.

[Answer](#)  by [geoff](#) 

I realize this has already been marked as answered, but we used to do it like this:

```
IF NOT EXISTS (SELECT * FROM sys.objects WHERE type = 'P' AND OBJECT_ID = OBJECT_ID('dbo.MyProc'))
    exec('CREATE PROCEDURE [dbo].[MyProc] AS BEGIN SET NOCOUNT ON; END')
GO

ALTER PROCEDURE [dbo].[MyProc]
AS
    ...
```

Just to avoid dropping the procedure.

[Answer](#)  by [mrchips](#) 

If you're looking for the simplest way to check for a database object's existence before removing it, here's one way (example uses a SPROC, just like your example above but could be modified for tables, indexes, etc...):

```
IF (OBJECT_ID('MyProcedure') IS NOT NULL)
    DROP PROCEDURE MyProcedure
GO
```

This is quick and elegant, but you need to make sure you have unique object names across all object types since it does not take that into account.

I Hope this helps!

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Finding duplicate rows in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [duplicates](#) ([Next Q](#))

I have a SQL Server database of organizations, and there are many duplicate rows. I want to run a select statement to grab all of these and the amount of dupes, but also return the ids that are associated with each organization.

A statement like:

```
SELECT      orgName, COUNT(*) AS dupes
FROM        organizations
GROUP BY    orgName
HAVING      (COUNT(*) > 1)
```

Will return something like

orgName	dupes
ABC Corp	7
Foo Federation	5
Widget Company	2

But I'd also like to grab the IDs of them. Is there any way to do this? Maybe like a

orgName	dupeCount	id
ABC Corp	1	34
ABC Corp	2	5
...		
Widget Company	1	10
Widget Company	2	2

The reason being that there is also a separate table of users that link to these organizations, and I would like to unify them (therefore remove dupes so the users link to the same organization instead of dupe orgs). But I would like part manually so I don't screw anything up, but I would still need a statement returning the IDs of all the dupe orgs so I can go through the list of users.

Any help would be greatly appreciated. Thanks :)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [duplicates](#) ([Next Q](#))

User: [xtine](#) 

[Answer](#) by [redfilter](#)

```
select o.orgName, oc.dupeCount, o.id
from organizations o
inner join (
    SELECT orgName, COUNT(*) AS dupeCount
    FROM organizations
    GROUP BY orgName
    HAVING COUNT(*) > 1
) oc on o.orgName = oc.orgName
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [duplicates](#) ([Next Q](#))

Q: How to fetch the row count for all tables in a SQL SERVER database

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I am searching for a SQL Script that can be used to determine if there is any data (i.e. row count) in any of the tables of a given database.

The idea is to re-incarnate the database in case there are any rows existing (in any of the database).

The database being spoken of is Microsoft SQL SERVER.

Could someone suggest a sample script?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [vijaysylvester](#)

[Answer](#) by [adrianbanks](#)

The following SQL will get you the row count of all tables in a database:

```
CREATE TABLE #counts
(
    table_name varchar(255),
    row_count int
)

EXEC sp_MSForEachTable @command1='INSERT #counts (table_name, row_count) SELECT ''?'' , COUNT(*) FROM
SELECT table_name, row_count FROM #counts ORDER BY table_name, row_count DESC
```

The output will be a list of tables and their row counts.

If you just want the total row count across the whole database, appending:

```
SELECT SUM(row_count) AS total_row_count FROM #counts
```

will get you a single value for the total number of rows in the whole database.

[Answer](#) by [keng](#)

If you want to bypass the time and resources it takes to count(*) your 3million row tables. Try this per SQL SERVER Central by Kendal Van Dyke.

Row Counts Using sysindexes If you're using SQL 2000 you'll need to use sysindexes

like so:

```
-- Shows all user tables and row counts for the current database --Remove OBJECTPROPERTY function call
SELECT o.NAME,
       i.rowcnt
  FROM sysindexes AS i
 INNER JOIN sysobjects AS o ON i.id = o.id
 WHERE i.indid < 2 AND OBJECTPROPERTY(o.id, 'IsMSShipped') = 0
 ORDER BY o.NAME
```

If you're using SQL 2005 or 2008 querying sysindexes will still work but Microsoft advises that sysindexes may be removed in a future version of SQL Server so as a good practice you should use the DMVs instead, like so:

[Skip code block](#)

```
-- Shows all user tables and row counts for the current database --Remove is_ms_shipped = 0 check to i
SELECT o.name,
       ddps.row_count
  FROM sys.indexes AS i
 INNER JOIN sys.objects AS o ON i.OBJECT_ID = o.OBJECT_ID
 INNER JOIN sys.dm_db_partition_stats AS ddps ON i.OBJECT_ID = ddps.OBJECT_ID
   AND i.index_id = ddps.index_id
 WHERE i.index_id < 2 AND o.is_ms_shipped = 0 ORDER BY o.NAME
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: UPDATE from SELECT using SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#))

In SQL Server, it's possible to *insert* into a table using a SELECT statement:

```
INSERT INTO Table (col1, col2, col3)
  SELECT col1, col2, col3 FROM other_table WHERE sql = 'cool'
```

Is it also possible to *update* via a SELECT? I have a temporary table containing the values, and would like to update another table using those values. Perhaps something like this:

```
UPDATE Table SET col1, col2
  SELECT col1, col2 FROM other_table WHERE sql = 'cool'
 WHERE Table.id = other_table.id
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#))

User: [sparkyfied](#) 

[Answer](#)  by [robin-day](#) 

[Skip code block](#)

```
UPDATE
      Table_A
SET
      Table_A.col1 = Table_B.col1,
      Table_A.col2 = Table_B.col2
FROM
      Some_Table Table_A
INNER JOIN
      Other_Table Table_B
ON
      Table_A.id = Table_B.id
WHERE
      Table_A.col3 = 'cool'
```

[Answer](#) by [onedaywhen](#)

In SQL Server 2008 (or better), use [MERGE](#)

```
MERGE INTO YourTable T
  USING other_table S
    ON T.id = S.id
      AND S.tsq1 = 'cool'
WHEN MATCHED THEN
  UPDATE
    SET col1 = S.col1,
        col2 = S.col2;
```

Alternatively:

[Skip code block](#)

```
MERGE INTO YourTable T
  USING (
    SELECT id, col1, col2
      FROM other_table
     WHERE tsq1 = 'cool'
  ) S
    ON T.id = S.id
WHEN MATCHED THEN
  UPDATE
    SET col1 = S.col1,
        col2 = S.col2;
```

[Answer](#) by [jamal](#)

```
UPDATE table
SET Col1 = i.Col1,
    Col2 = i.Col2
FROM (
  SELECT ID, Col1, Col2
    FROM other_table) i
WHERE
  i.ID = table.ID
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#))

Q: SQL server query to get the list of columns in a table along with Data types, NOT NULL, and PRIMARY KEY constraints

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

I need to write a query on SQL server to get the list of columns in a particular table, its associated data types and their length and if they are not null. I have managed to do this much. But now i also need to get in the same table against a column - TRUE if it is a primary key. How do i do this ?

This is how the output should be:

Columns_name----Data type----Length----isnull----Pk

please help me!

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [shrayas](#)

[Answer](#) by [marc_s](#)

To avoid duplicate rows for some columns, use user_type_id instead of system_type_id.

[Skip code block](#)

```
SELECT
    c.name 'Column Name',
    t.Name 'Data type',
    c.max_length 'Max Length',
    c.precision ,
    c.scale ,
    c.is_nullable,
    ISNULL(i.is_primary_key, 0) 'Primary Key'
FROM
    sys.columns c
INNER JOIN
    sys.types t ON c.user_type_id = t.user_type_id
LEFT OUTER JOIN
    sys.index_columns ic ON ic.object_id = c.object_id AND ic.column_id = c.column_id
LEFT OUTER JOIN
    sys.indexes i ON ic.object_id = i.object_id AND ic.index_id = i.index_id
WHERE
    c.object_id = OBJECT_ID('YourTableName')
```

Just replace YourTableName with your actual table name - works for SQL Server 2005 and up.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: Multi-statement Table Valued Function vs Inline Table Valued Function](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

A few examples to show, just incase:

Inline Table Valued

```
CREATE FUNCTION MyNS.GetUnshippedOrders()
RETURNS TABLE
AS
RETURN SELECT a.SaleId, a.CustomerID, b.Qty
        FROM Sales.Sales a INNER JOIN Sales.SaleDetail b
            ON a.SaleId = b.SaleId
            INNER JOIN Production.Product c ON b.ProductID = c.ProductID
            WHERE a.ShipDate IS NULL
GO
```

Multi Statement Table Valued

[Skip code block](#)

```
CREATE FUNCTION MyNS.GetLastShipped(@CustomerID INT)
RETURNS @CustomerOrder TABLE
(SaleOrderID      INT          NOT NULL,
CustomerID       INT          NOT NULL,
OrderDate        DATETIME     NOT NULL,
OrderQty         INT          NOT NULL)
AS
BEGIN
    DECLARE @MaxDate DATETIME

    SELECT @MaxDate = MAX(OrderDate)
    FROM Sales.SalesOrderHeader
    WHERE CustomerID = @CustomerID
```

```

INSERT @CustomerOrder
SELECT a.SalesOrderID, a.CustomerID, a.OrderDate, b.OrderQty
FROM Sales.SalesOrderHeader a INNER JOIN Sales.SalesOrderHeader b
    ON a.SalesOrderID = b.SalesOrderID
    INNER JOIN Production.Product c ON b.ProductID = c.ProductID
WHERE a.OrderDate = @MaxDate
    AND a.CustomerID = @CustomerID
RETURN
END
GO

```

Is there an advantage to using one type (in-line or multi statement) over the other? Is there certain scenarios when one is better than the other or are the differences purely syntactical? I realise the two example queries are doing different things but is there a reason I would write them in that way?

Reading about them and the advantages/differences haven't really been explained.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [andrewc](#) 

[Answer](#)  by [thomas](#) 

In researching Matt's comment, I have revised my original statement. He is correct, there will be a difference in performance between an inline table valued function (ITVF) and a multi-statement table valued function (MSTVF) even if they both simply execute a SELECT statement. SQL Server will treat an ITVF somewhat like a VIEW in that it will calculate an execution plan using the latest statistics on the tables in question. A MSTVF is equivalent to stuffing the entire contents of your SELECT statement into a table variable and then joining to that. Thus, the compiler cannot use any table statistics on the tables in the MSTVF. So, all things being equal, (which they rarely are), the ITVF will perform better than the MSTVF. In my tests, the performance difference in completion time was negligible however from a statistics standpoint, it was noticeable.

In your case, the two functions are not functionally equivalent. The MSTV function does an extra query each time it is called and, most importantly, filters on the customer id. In a large query, the optimizer would not be able to take advantage of other types of joins as it would need to call the function for each customerId passed. However, if you re-wrote your MSTV function like so:

[Skip code block](#)

```

CREATE FUNCTION MyNS.GetLastShipped()
RETURNS @CustomerOrder TABLE
(
    SaleOrderID      INT          NOT NULL,
    CustomerID       INT          NOT NULL,
    OrderDate        DATETIME     NOT NULL,
    OrderQty         INT          NOT NULL
)
AS
BEGIN
    INSERT @CustomerOrder
    SELECT a.SalesOrderID, a.CustomerID, a.OrderDate, b.OrderQty
    FROM Sales.SalesOrderHeader a
        INNER JOIN Sales.SalesOrderHeader b
            ON a.SalesOrderID = b.SalesOrderID
        INNER JOIN Production.Product c
            ON b.ProductID = c.ProductID

```

```

        WHERE a.OrderDate = (
            Select Max(SH1.OrderDate)
            FROM Sales.SalesOrderHeader As SH1
            WHERE SH1.CustomerID = A.CustomerId
        )
    RETURN
END
GO

```

In a query, the optimizer would be able to call that function once and build a better execution plan but it still would not be better than an equivalent, non-parameterized ITVS or a VIEW.

ITVFs should be preferred over a MSTVFs when feasible because the datatypes, nullability and collation from the columns in the table whereas you declare those properties in a multi-statement table valued function and, importantly, you will get better execution plans from the ITVFs. In my experience, I have not found many circumstances where an ITVFs was a better option than a VIEW but mileage may vary.

Thanks to Matt.

Addition

Since I saw this come up recently, here is an excellent analysis done by Wayne Sheffield comparing the performance difference between Inline Table Valued functions and Multi-Statement functions.

[His original blog post.](#) 

[Copy on SQL Server Central](#) 

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

[Q: INNER JOIN vs LEFT JOIN performance in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

I've created SQL command that use INNER JOIN for 9 tables, anyway this command take a very long time (more than five minutes). So my folk suggest me to change INNER JOIN to LEFT JOIN because the performance of LEFT JOIN is better, at first time its despite what I know. After I changed, the speed of query is significantly improve.

I would like to know why LEFT JOIN is faster than INNER JOIN?

My SQL command look like below: `SELECT * FROM A INNER JOIN B ON... INNER JOIN C ON... INNER JOIN D` and so on

Update: This is brief of my schema.

[Skip code block](#)

```

FROM sidisalesdrmlly a-NOT HAVE PK AND FK
    INNER JOIN sidisalesdetmly b-THIS TABLE ALSO HAVE NO PK AND FK
        ON a.CompanyCd = b.CompanyCd
        AND a.SPRNo = b.SPRNo
        AND a.SuffixNo = b.SuffixNo
        AND a.dnno = b.dnno
    INNER JOIN exFSlipDet h-PK = CompanyCd, FSlipNo, FSlipSuffix, FSlipLine

```

```

        ON a.CompanyCd = h.CompanyCd
        AND a.sprno = h.AcctSPRNo
    INNER JOIN exFSlipHdr c-PK = CompanyCd, FSlipNo, FSlipSuffix
        ON c.CompanyCd = h.CompanyCd
        AND c.FSlipNo = h.FSlipNo
        AND c.FSlipSuffix = h.FSlipSuffix
    INNER JOIN coMappingExpParty d-NO PK AND FK
        ON c.CompanyCd = d.CompanyCd
        AND c.CountryCd = d.CountryCd
    INNER JOIN coProduct e-PK = CompanyCd, ProductSalesCd
        ON b.CompanyCd = e.CompanyCd
        AND b.ProductSalesCd = e.ProductSalesCd
    LEFT JOIN coUOM i-PK = UOMId
        ON h.UOMId = i.UOMId
    INNER JOIN coProductOldInformation j-PK = CompanyCd, BFStatus, SpecCd
        ON a.CompanyCd = j.CompanyCd
        AND b.BFStatus = j.BFStatus
        AND b.ProductSalesCd = j.ProductSalesCd
    INNER JOIN coProductGroup1 g1-PK = CompanyCd, ProductCategoryCd, UsedDepartment, ProductGroup1Cd
        ON e.ProductGroup1Cd = g1.ProductGroup1Cd
    INNER JOIN coProductGroup2 g2-PK = CompanyCd, ProductCategoryCd, UsedDepartment, ProductGroup2Cd
        ON e.ProductGroup1Cd = g2.ProductGroup1Cd

```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

User: [anonymous](#) 

Answer  by [aaronnaught](#) 

A `LEFT JOIN` is absolutely not faster than an `INNER JOIN`. In fact, it's slower; by definition, an outer join (`LEFT JOIN` or `RIGHT JOIN`) has to do all the work of an `INNER JOIN` plus the extra work of null-extending the results. It would also be expected to return more rows, further increasing the total execution time simply due to the larger size of the result set.

(And even if a `LEFT JOIN` were faster in *specific* situations due to some difficult-to-imagine confluence of factors, it is not functionally equivalent to an `INNER JOIN`, so you cannot simply go replacing all instances of one with the other!)

Most likely your performance problems lie elsewhere, such as not having a candidate key or foreign key indexed properly. 9 tables is quite a lot to be joining so the slowdown could literally be almost anywhere. If you post your schema, we might be able to provide more details.

Edit:

Reflecting further on this, I could think of one circumstance under which a `LEFT JOIN` might be faster than an `INNER JOIN`, and that is when:

- Some of the tables are *very* small (say, under 10 rows);
- The tables do not have sufficient indexes to cover the query.

Consider this example:

[Skip code block](#)

```

CREATE TABLE #Test1
(
    ID int NOT NULL PRIMARY KEY,
    Name varchar(50) NOT NULL
)
INSERT #Test1 (ID, Name) VALUES (1, 'One')
INSERT #Test1 (ID, Name) VALUES (2, 'Two')
INSERT #Test1 (ID, Name) VALUES (3, 'Three')

```

```

INSERT #Test1 (ID, Name) VALUES (4, 'Four')
INSERT #Test1 (ID, Name) VALUES (5, 'Five')

CREATE TABLE #Test2
(
    ID int NOT NULL PRIMARY KEY,
    Name varchar(50) NOT NULL
)
INSERT #Test2 (ID, Name) VALUES (1, 'One')
INSERT #Test2 (ID, Name) VALUES (2, 'Two')
INSERT #Test2 (ID, Name) VALUES (3, 'Three')
INSERT #Test2 (ID, Name) VALUES (4, 'Four')
INSERT #Test2 (ID, Name) VALUES (5, 'Five')

SELECT *
FROM #Test1 t1
INNER JOIN #Test2 t2
ON t2.Name = t1.Name

SELECT *
FROM #Test1 t1
LEFT JOIN #Test2 t2
ON t2.Name = t1.Name

DROP TABLE #Test1
DROP TABLE #Test2

```

If you run this and view the execution plan, you'll see that the `INNER JOIN` query does indeed cost more than the `LEFT JOIN`, because it satisfies the two criteria above. It's because SQL Server wants to do a hash match for the `INNER JOIN`, but does nested loops for the `LEFT JOIN`; the former is *normally* much faster, but since the number of rows is so tiny *and* there's no index to use, the hashing operation turns out to be the most expensive part of the query.

You can see the same effect by writing a program in your favourite programming language to perform a large number of lookups on a list with 5 elements, vs. a hash table with 5 elements. Because of the size, the hash table version is actually slower. But increase it to 50 elements, or 5000 elements, and the list version slows to a crawl, because it's $O(N)$ vs. $O(1)$ for the hashtable.

But change this query to be on the `ID` column instead of `Name` and you'll see a very different story. In that case, it does nested loops for both queries, but the `INNER JOIN` version is able to replace one of the clustered index scans with a seek - meaning that this will literally be *an order of magnitude* faster with a large number of rows.

So the conclusion is more or less what I mentioned several paragraphs above; this is almost certainly an indexing or index coverage problem, possibly combined with one or more very small tables. Those are the only circumstances under which SQL Server *might* sometimes choose a worse execution plan for an `INNER JOIN` than a `LEFT JOIN`.

[Answer](#)  by [dbenham](#) 

There is one important scenario that can lead to an outer join being faster than an inner join that has not been discussed yet.

When using an outer join, the optimizer is always free to drop the outer joined table from the execution plan if the join columns are the PK of the outer table, and none of the columns are selected from the outer table. For example `SELECT A.* FROM A LEFT OUTER JOIN B ON A.KEY=B.KEY` and `B.KEY` is the PK for `B`. Both Oracle (I believe I was using

release 10) and Sql Server (I used 2008 R2) prune table B from the execution plan.

The same is not necessarily true for an inner join: `SELECT A.* FROM A INNER JOIN B ON A.KEY=B.KEY` may or may not require B in the execution plan depending on what constraints exist.

If A.KEY is a nullable foreign key referencing B.KEY, then the optimizer cannot drop B from the plan because it must confirm that a B row exists for every A row.

If A.KEY is a mandatory foreign key referencing B.KEY, then the optimizer is free to drop B from the plan because the constraints guarantee the existence of the row. But just because the optimizer can drop the table from the plan, doesn't mean it will. SQL Server 2008 R2 does NOT drop B from the plan. Oracle 10 DOES drop B from the plan. It is easy to see how the outer join will out-perform the inner join on SQL Server in this case.

This is a trivial example, and not practical for a stand-alone query. Why join to a table if you don't need to?

But this could be a very important design consideration when designing views. Frequently a "do-everything" view is built that joins everything a user might need related to a central table. (Especially if there are naive users doing ad-hoc queries that do not understand the relational model) The view may include all the relevant columns from many tables. But the end users might only access columns from a subset of the tables within the view. If the tables are joined with outer joins, then the optimizer can (and does) drop the un-needed tables from the plan.

It is critical to make sure that the view using outer joins gives the correct results. As Aaronaught has said - you cannot blindly substitute OUTER JOIN for INNER JOIN and expect the same results. But there are times when it can be useful for performance reasons when using views.

One last note - I haven't tested the impact on performance in light of the above, but in theory it seems you should be able to safely replace an INNER JOIN with an OUTER JOIN if you also add the condition `<FOREIGN_KEY> IS NOT NULL` to the where clause.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

Q: What does ON [PRIMARY] mean?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Next Q](#))

I'm creating an SQL setup script and I'm using someone else's script as an example. Here's an example of the script:

[Skip code block](#)

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[be_Categories](
    [CategoryID] [uniqueidentifier] ROWGUIDCOL NOT NULL CONSTRAINT [DF_be_Categories_CategoryID]  DE
    [CategoryName] [nvarchar](50) NULL,
    [Description] [nvarchar](200) NULL,
    [ParentID] [uniqueidentifier] NULL,
CONSTRAINT [PK_be_Categories] PRIMARY KEY CLUSTERED
(
    [CategoryID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
) ON [PRIMARY]
GO
```

Does anyone know what the ON [PRIMARY] command does?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Next Q](#))

User: [icono123](#)

[Answer](#) by [blowdart](#)

When you create a database in Microsoft SQL Server you can have multiple file groups, where storage is created in multiple places, directories or disks. Each file group can be named. The PRIMARY file group is the default one, which is always created, and so the SQL you've given creates your table ON the PRIMARY file group.

See [MSDN](#) for the full syntax.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Next Q](#))

Q: Is there a combination of “LIKE” and “IN” in SQL?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Next Q](#))

In SQL I (sadly) often have to use “LIKE” conditions due to databases that violate nearly every rule of normalization. I can't change that right now. But that's irrelevant to the question.

Further, I often use conditions like `WHERE something in (1,1,2,3,5,8,13,21)` for better readability and flexibility of my SQL statements.

Is there any possible way to combine these two things without writing complicated sub-selects?

I want something as easy as `WHERE something LIKE ('bla%', '%foo%', 'batz%')`

instead of

```
WHERE something LIKE 'bla%'  
OR something LIKE '%foo%'  
OR something LIKE 'batz%'
```

I'm working with SQL Server and Oracle here but I'm interested if this is possible in any RDBMS at all.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Next Q](#))

User: [lnwdr](#) 

[Answer](#)  by [omg-ponies](#) 

There is no combination of LIKE & IN in SQL, much less in TSQL (SQL Server) or PLSQL (Oracle). Part of the reason for that is because Full Text Search (FTS) is the recommended alternative.

Both Oracle and SQL Server FTS implementations support the CONTAINS keyword, but the syntax is still slightly different:

Oracle:

```
WHERE CONTAINS(t.something, 'bla OR foo OR batz', 1) > 0
```

SQL Server:

```
WHERE CONTAINS(t.something, '"bla*" OR "foo*" OR "batz*"')
```

Reference:

- [Building Full-Text Search Applications with Oracle Text](#) 
 - [Understanding SQL Server Full-Text](#) 
-

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Next Q](#))

[Q: SQL Server altering a column to be nullable](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I want to alter a table column to be nullable. I have used:

```
ALTER TABLE Merchant_Pending_Functions Modify NumberOfLocations NULL
```

This gives an error at Modify. What is the correct syntax?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [ram](#) 

[Answer](#)  by [quassnoi](#) 

Assuming SQL Server (based on your previous questions):

```
ALTER TABLE Merchant_Pending_Functions ALTER COLUMN NumberOfLocations INT NULL
```

Replace INT with your actual datatype.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Q: How can you represent inheritance in a database?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

I'm thinking about how to represent a complex structure in a SQL Server database.

Consider an application that needs to store details of a family of objects, which share some attributes, but have many others not common. For example, a commercial insurance package may include liability, motor, property and indemnity cover within the same policy record.

It is trivial to implement this in C#, etc, as you can create a Policy with a collection of Sections, where Section is inherited as required for the various types of cover. However, relational databases don't seem to allow this easily.

I can see that there are two main choices:

1. Create a Policy table, then a Sections table, with all the fields required, for all possible variations, most of which would be null.
2. Create a Policy table and numerous Section tables, one for each kind of cover.

Both of these alternatives seem unsatisfactory, especially as it is necessary to write queries across all Sections, which would involve numerous joins, or numerous null-checks.

What is the best practice for this scenario?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [steve-jones](#) 

[Answer](#)  by [daniel-vassallo](#) 

@Bill Karwin  describes three inheritance models in his [SQL Antipatterns](#)  book, when proposing solutions to the SQL [Entity-Attribute-Value](#)  antipattern. This is a brief overview:

Single Table Inheritance (aka Table Per Hierarchy Inheritance):

Using a single table as in your first option is probably the simplest design. As you mentioned, many attributes that are subtype-specific will have to be given a NULL value on rows where these attributes do not apply. With this model, you would have one policies table, which would look something like this:

[Skip code block](#)

id	date_issued	type	vehicle_reg_no	property_address
1	2010-08-20 12:00:00	MOTOR	01-A-04004	NULL
2	2010-08-20 13:00:00	MOTOR	02-B-01010	NULL
3	2010-08-20 14:00:00	PROPERTY	NULL	Oxford Street
4	2010-08-20 15:00:00	MOTOR	03-C-02020	NULL

\----- COMMON FIELDS -----/ \----- SUBTYPE SPECIFIC FIELDS -----/

Keeping the design simple is a plus, but the main problems with this approach are the following:

- When it comes to adding new subtypes, you would have to alter the table to accommodate the attributes that describe these new objects. This can quickly become problematic when you have many subtypes, or if you plan to add subtypes on a regular basis.
- The database will not be able to enforce which attributes apply and which don't, since there is no metadata to define which attributes belong to which subtypes.
- You also cannot enforce NOT NULL on attributes of a subtype that should be mandatory. You would have to handle this in your application, which in general is not ideal.

Concrete Table Inheritance:

Another approach to tackle inheritance is to create a new table for each subtype, repeating all the common attributes in each table. For example:

[Skip code block](#)

```
--// Table: policies_motor
+-----+
| id | date_issued | vehicle_reg_no |
+-----+
| 1 | 2010-08-20 12:00:00 | 01-A-04004 |
| 2 | 2010-08-20 13:00:00 | 02-B-01010 |
| 3 | 2010-08-20 15:00:00 | 03-C-02020 |
+-----+



--// Table: policies_property
+-----+
| id | date_issued | property_address |
+-----+
| 1 | 2010-08-20 14:00:00 | Oxford Street |
+-----+
```

This design will basically solve the problems identified for the single table method:

- Mandatory attributes can now be enforced with NOT NULL.
- Adding a new subtype requires adding a new table instead of adding columns to an existing one.
- There is also no risk that an inappropriate attribute is set for a particular subtype, such as the vehicle_reg_no field for a property policy.
- There is no need for the type attribute as in the single table method. The type is now

defined by the metadata: the table name.

However this model also comes with a few disadvantages:

- The common attributes are mixed with the subtype specific attributes, and there is no easy way to identify them. The database will not know either.
- When defining the tables, you would have to repeat the common attributes for each subtype table. That's definitely not [DRY](#).
- Searching for all the policies regardless of the subtype becomes difficult, and would require a bunch of `UNIONS`.

This is how you would have to query all the policies regardless of the type:

```
SELECT      date_issued, other_common_fields, 'MOTOR' AS type
FROM        policies_motor
UNION ALL
SELECT      date_issued, other_common_fields, 'PROPERTY' AS type
FROM        policies_property;
```

Note how adding new subtypes would require the above query to be modified with an additional `UNION ALL` for each subtype. This can easily lead to bugs in your application if this operation is forgotten.

Class Table Inheritance (aka Table Per Type Inheritance):

This is the solution that [@David mentions in the other answer](#). You create a single table for your base class, which includes all the common attributes. Then you would create specific tables for each subtype, whose primary key also serves as a [foreign key](#) to the base table. Example:

Skip code block

```
CREATE TABLE policies (
    policy_id      int,
    date_issued    datetime,
    -- other common attributes...
);

CREATE TABLE policy_motor (
    policy_id      int,
    vehicle_reg_no varchar(20),
    -- other attributes specific to motor insurance...
    FOREIGN KEY (policy_id) REFERENCES policies (policy_id)
);

CREATE TABLE policy_property (
    policy_id      int,
    property_address varchar(20),
    -- other attributes specific to property insurance...
    FOREIGN KEY (policy_id) REFERENCES policies (policy_id)
);
```

This solution solves the problems identified in the other two designs:

- Mandatory attributes can be enforced with NOT NULL.
- Adding a new subtype requires adding a new table instead of adding columns to an existing one.
- No risk that an inappropriate attribute is set for a particular subtype.
- No need for the type attribute.
- Now the common attributes are not mixed with the subtype specific attributes anymore.
- We can stay DRY, finally. There is no need to repeat the common attributes for each subtype table when creating the tables.
- Managing an auto incrementing id for the policies becomes easier, because this can be handled by the base table, instead of each subtype table generating them independently.
- Searching for all the policies regardless of the subtype now becomes very easy: No UNIONs needed - just a SELECT * FROM policies.

I consider the class table approach as the most suitable in most situations.

The names of these three models come from [Martin Fowler's](#)  book [Patterns of Enterprise Application Architecture](#) .

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

[**Q: Get all table names of a particular database by SQL query?**](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I working on application which can deal with multiple database servers like “MySQL” and “MS SQL Server”.

I want to get tables names of a particular database using a general query which should suitable for all database types. I have tried following:

```
SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_TYPE='BASE TABLE'
```

But it is giving table names of all databases of a particular server but I want to get tables names of selected database only. How can I restrict this query to get tables of a particular database?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [awan](#) 

[Answer](#)  by [michael-baylon](#) 

Probably due to the way different sql dbms deal with schemas.

Try the following

For SQL Server:

```
SELECT TABLE_NAME  
FROM INFORMATION_SCHEMA.TABLES  
WHERE TABLE_TYPE = 'BASE TABLE' AND TABLE_CATALOG='dbName'
```

For MySQL:

```
SELECT TABLE_NAME  
FROM INFORMATION_SCHEMA.TABLES  
WHERE TABLE_TYPE = 'BASE TABLE' AND TABLE_SCHEMA='dbName'
```

For Oracle I think the equivalent would be to use DBA_TABLES.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I backup a remote SQL Server database to a local drive?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

I need to copy a database from a remote server to a local one. I tried to use SQL Server Management Studio, but it only backs up to a drive on the remote server.

Some points:

- I do not have access to the remote server in a way that I could copy files;
- I do not have access to setup a UNC path to my server;

Any ideas of how can I copy this database? Will I have to use 3rd party tools?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [fernando](#) 

[Answer](#)  by [daniel-gill](#) 

In Microsoft SQL Server Management Studio you can right-click on the database you wish to backup and click Tasks -> Generate Scripts.

This pops open a wizard where you can set the following in order to perform a decent backup of your database, **even on a remote server**:

- Select the database you wish to backup and hit next,
- In the options it presents to you:
 1. In 2010: under the Table/View Options, change ‘Script Data’ and ‘Script Indexes’ to True and hit next,
 2. In 2012: under ‘General’, change ‘Types of data to script’ from ‘Schema only’ to ‘Schema and data’
 3. In 2014: the option to script the data is now “hidden” in step “Set Scripting Options”, you have to click the “Advanced” and set “Types of data to script” to “Schema and data” value
- In the next *four* windows, hit ‘select all’ and then next,
- Choose to script to a new query window

Once it's done its thing, you'll have a backup script ready in front of you. Create a new local (or remote) database, and change the first 'USE' statement in the script to use your new database. Save the script in a safe place, and go ahead and run it against your new empty database. This should create you a (nearly) duplicate local database you can then backup as you like.

If you have *full* access to the remote database, you can choose to check 'script all objects' in the wizard's first window and then change the 'Script Database' option to True on the next window. Watch out though, you'll need to perform a full search & replace of the database name in the script to a new database which in this case you won't have to create before running the script. This should create a more accurate duplicate but is sometimes not available due to permissions restrictions.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: SET versus SELECT when assigning variables?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

What are the differences between the `SET` and `SELECT` statements when assigning variables in T-SQL?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [juur](#) 

[Answer](#)  by [omg-ponies](#) 

[Quote](#) , which summarizes from [this article](#) :

1. `SET` is the ANSI standard for variable assignment, `SELECT` is not.
2. `SET` can only assign one variable at a time, `SELECT` can make multiple assignments at once.
3. If assigning from a query, `SET` can only assign a scalar value. If the query returns multiple values/rows then `SET` will raise an error. `SELECT` will assign one of the values to the variable and hide the fact that multiple values were returned (so you'd likely never know why something was going wrong elsewhere - have fun troubleshooting that one)
4. When assigning from a query if there is no value returned then `SET` will assign `NULL`, where `SELECT` will not make the assignment at all (so the variable will not be changed from its previous value)
5. As far as speed differences - there are no direct differences between `SET` and `SELECT`. However `SELECT`'s ability to make multiple assignments in one shot does give it a slight speed advantage over `SET`.

[Answer](#)  by [joe-stefanelli](#) 

I believe SET is ANSI standard whereas the SELECT is not. Also note the different behavior of SET vs. SELECT in the example below when a value is not found.

```
declare @var varchar(20)
set @var = 'Joe'
set @var = (select name from master.sys.tables where name = 'qwerty')
select @var /* @var is now NULL */

set @var = 'Joe'
select @var = name from master.sys.tables where name = 'qwerty'
select @var /* @var is still equal to 'Joe' */
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL Server SELECT into existing table](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I am trying to select some fields from one table and insert them into an existing table from a stored procedure. Here is what I am trying:

```
SELECT col1, col2
INTO dbo.TableTwo
FROM dbo.TableOne
WHERE col3 LIKE @search_key
```

I think select into is for temporary tables which is why I get an error that dbo.TableTwo already exists.

How can I insert multiple rows from dbo.TableOne into dbo.TableTwo?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [daniel](#) 

[Answer](#)  by [omg-ponies](#) 

SELECT... INTO... only works if the table specified in the INTO clause does not exist - otherwise, you have to use:

```
INSERT INTO dbo.TABLETWO
SELECT col1, col2
  FROM dbo.TABLEONE
 WHERE col3 LIKE @search_key
```

This assumes there's only two columns in dbo.TABLETWO - you need to specify the columns otherwise:

```
INSERT INTO dbo.TABLETWO
  (col1, col2)
SELECT col1, col2
  FROM dbo.TABLEONE
 WHERE col3 LIKE @search_key
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Q: Need to list all triggers in SQL Server database with table name and table's schema

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#)), [triggers](#) ([Next Q](#))

I need to list all triggers in SQL Server database with table name and table's schema.

I'm almost there with this:

```
SELECT trigger_name = name, trigger_owner = USER_NAME(uid), table_schema = , table_name = OBJECT_NAME(
    isupdate = OBJECTPROPERTY( id, 'ExecIsUpdateTrigger'), isdelete = OBJECTPROPERTY( id, 'ExecIsDeleteTrigger'),
    isinsert = OBJECTPROPERTY( id, 'ExecIsInsertTrigger'), isafter = OBJECTPROPERTY( id, 'ExecIsAfterTrigger'),
    isinsteadof = OBJECTPROPERTY( id, 'ExecIsInsteadOfTrigger'),
    [disabled] = OBJECTPROPERTY(id, 'ExecIsTriggerDisabled')
) FROM sysobjects INNER JOIN sysusers ON sysobjects.uid = sysusers.uid
WHERE type = 'TR'
```

I just need to get the table's schema also.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#)), [triggers](#) ([Next Q](#))

User: [ronnie-overby](#)

[Answer](#) by [joe-stefanelli](#)

Here's one way:

[Skip code block](#)

```
SELECT
    sysobjects.name AS trigger_name
    ,USER_NAME(sysobjects.uid) AS trigger_owner
    ,s.name AS table_schema
    ,OBJECT_NAME(parent_obj) AS table_name
    ,OBJECTPROPERTY( id, 'ExecIsUpdateTrigger') AS isupdate
    ,OBJECTPROPERTY( id, 'ExecIsDeleteTrigger') AS isdelete
    ,OBJECTPROPERTY( id, 'ExecIsInsertTrigger') AS isinsert
    ,OBJECTPROPERTY( id, 'ExecIsAfterTrigger') AS isafter
    ,OBJECTPROPERTY( id, 'ExecIsInsteadOfTrigger') AS isinsteadof
    ,OBJECTPROPERTY(id, 'ExecIsTriggerDisabled') AS [disabled]
FROM sysobjects
INNER JOIN sysusers
    ON sysobjects.uid = sysusers.uid
INNER JOIN sys.tables t
    ON sysobjects.parent_obj = t.object_id
INNER JOIN sys.schemas s
    ON t.schema_id = s.schema_id
WHERE sysobjects.type = 'TR'
```

EDIT: Commented out join to sysusers for query to work on AdventureWorks2008.

[Skip code block](#)

```
SELECT
    sysobjects.name AS trigger_name
    ,USER_NAME(sysobjects.uid) AS trigger_owner
    ,s.name AS table_schema
    ,OBJECT_NAME(parent_obj) AS table_name
    ,OBJECTPROPERTY( id, 'ExecIsUpdateTrigger') AS isupdate
    ,OBJECTPROPERTY( id, 'ExecIsDeleteTrigger') AS isdelete
    ,OBJECTPROPERTY( id, 'ExecIsInsertTrigger') AS isinsert
    ,OBJECTPROPERTY( id, 'ExecIsAfterTrigger') AS isafter
    ,OBJECTPROPERTY( id, 'ExecIsInsteadOfTrigger') AS isinsteadof
```

```

,OBJECTPROPERTY(id, 'ExecIsTriggerDisabled') AS [disabled]
FROM sysobjects
/*
INNER JOIN sysusers
    ON sysobjects.uid = sysusers.uid
*/
INNER JOIN sys.tables t
    ON sysobjects.parent_obj = t.object_id

INNER JOIN sys.schemas s
    ON t.schema_id = s.schema_id
WHERE sysobjects.type = 'TR'

```

EDIT 2: For SQL 2000

[Skip code block](#)

```

SELECT
    o.name AS trigger_name
    , 'x' AS trigger_owner
    /*USER_NAME(o.uid)*/
    ,s.name AS table_schema
    ,OBJECT_NAME(o.parent_obj) AS table_name
    ,OBJECTPROPERTY(o.id, 'ExecIsUpdateTrigger') AS isupdate
    ,OBJECTPROPERTY(o.id, 'ExecIsDeleteTrigger') AS isdelete
    ,OBJECTPROPERTY(o.id, 'ExecIsInsertTrigger') AS isinsert
    ,OBJECTPROPERTY(o.id, 'ExecIsAfterTrigger') AS isafter
    ,OBJECTPROPERTY(o.id, 'ExecIsInsteadOfTrigger') AS isinsteadof
    ,OBJECTPROPERTY(o.id, 'ExecIsTriggerDisabled') AS [disabled]
FROM sysobjects AS o
/*
INNER JOIN sysusers
    ON sysobjects.uid = sysusers.uid
*/
INNER JOIN sysobjects AS o2
    ON o.parent_obj = o2.id

INNER JOIN sysusers AS s
    ON o2.uid = s.uid

WHERE o.type = 'TR'

```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#)), [triggers](#) ([Next Q](#))

Q: Select statement to find duplicates on certain fields

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

Can you help me with SQL statements to find duplicates on multiple fields?

For example, in pseudo code:

```

select count(field1,field2,field3)
from table
where the combination of field1, field2, field3 occurs multiple times

```

and from the above statement *if there are multiple occurrences* I would like to select *every record except the first one*.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [joe-skeet](#) 

[Answer](#) by [rajesh-chamarthi](#)

To get the list of fields for which there are multiple records, you can use..

```
select field1,field2,field3, count(*)  
  from table_name  
 group by field1,field2,field3  
 having count(*) > 1
```

Check this link for more information on how to delete the rows.

<http://support.microsoft.com/kb/139444>

Edit : As the other users mentioned, there should be a criterion for deciding how you define “first rows” before you use the approach in the link above. Based on that you’ll need to use an order by clause and a sub query if needed. If you can post some sample data, it would really help.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

[Q: Nested select statement in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Can someone help me understand why the following doesn't work?

```
SELECT name FROM (SELECT name FROM agentinformation)
```

I guess my understanding of SQL is wrong, because I would have thought this would return the same thing as

```
SELECT name FROM agentinformation
```

Doesn't the inner select statement create a result set which the outer SELECT statement then queries?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [brennan-vincent](#)

[Answer](#) by [joe-stefanelli](#)

You need to alias the subquery.

```
SELECT name FROM (SELECT name FROM agentinformation) a
```

or to be more explicit

```
SELECT a.name FROM (SELECT name FROM agentinformation) a
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

Q: SQL Server SELECT INTO @variable?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I have the following code in one of my Sql (2008) Stored Procs which executes perfectly fine:

[Skip code block](#)

```
CREATE PROCEDURE [dbo].[Item_AddItem]
    @CustomerId uniqueidentifier,
    @Description nvarchar(100),
    @Type int,
    @Username nvarchar(100),
AS
BEGIN

    DECLARE @TopRelatedItemId uniqueidentifier;
    SET @TopRelatedItemId =
    (
        SELECT top(1) RelatedItemId
        FROM RelatedItems
        WHERE CustomerId = @CustomerId
    )

    DECLARE @TempItem TABLE
    (
        ItemId uniqueidentifier,
        CustomerId uniqueidentifier,
        Description nvarchar(100),
        Type int,
        Username nvarchar(100),
        TimeStamp datetime
    );

    INSERT INTO Item
    OUTPUT INSERTED.* INTO @TempItem
    SELECT NEWID(), @CustomerId, @Description, @Type, @Username, GETDATE()

    SELECT
        ItemId,
        CustomerId,
        @TopRelatedItemId,
        Description,
        Type,
        Username,
        TimeStamp
    FROM
        @TempItem
END
GO
```

So the question for you guys is is there a possibility to do something along the lines of:

[Skip code block](#)

```
DECLARE @TempCustomer TABLE
(
    CustomerId uniqueidentifier,
    FirstName nvarchar(100),
    LastName nvarchar(100),
    Email nvarchar(100)
);
SELECT
    CustomerId,
    FirstName,
    LastName,
    Email
INTO
```

```
@TempCustomer
FROM
    Customer
WHERE
    CustomerId = @CustomerId
```

So that I could reuse this data from memory in other following statements? SQL Server throws a fit with the above statement, however i don't want to have to create separate variables and initialize each one of them via a separate SELECT statement against the same table.... UGH!!!

Any suggestions on how to achieve something along the lines without multiple queries against the same table?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [bleepzter](#) 

[Answer](#)  by [richardthekiw](#) 

You cannot SELECT .. INTO .. a TABLE VARIABLE. The best you can do is create it first, then insert into it. Your 2nd snippet has to be

[Skip code block](#)

```
DECLARE @TempCustomer TABLE
(
    CustomerId uniqueidentifier,
    FirstName nvarchar(100),
    LastName nvarchar(100),
    Email nvarchar(100)
);
INSERT INTO
    @TempCustomer
SELECT
    CustomerId,
    FirstName,
    LastName,
    Email
FROM
    Customer
WHERE
    CustomerId = @CustomerId
```

[Answer](#)  by [dougajmcdonald](#) 

If you wanted to simply assign some variables for later use, you can do them in one shot with something along these lines:

[Skip code block](#)

```
declare @var1 int,@var2 int,@var3 int;

select
    @var1 = field1,
    @var2 = field2,
    @var3 = field3
from
    table
where
    condition
```

If that's the type of thing you're after

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to select date without time in SQL](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

When I select date in SQL it is returned as 2011-02-25 21:17:33.933. But I need only the Date part, that is 2011-02-25. How can I do this?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [neeraj](#) 

[Answer](#)  by [bernd_k](#) 

I guess he wants a string

```
select convert(varchar(10), '2011-02-25 21:17:33.933', 120)
```

[Answer](#)  by [prad9](#) 

```
Convert(date, getdate())
```

Please refer to <http://msdn.microsoft.com/en-us/library/ms188383.aspx> 

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to get last inserted id?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#))

I have this code:

[Skip code block](#)

```
string insertSql =
    "INSERT INTO aspnet_GameProfiles(UserId,GameId) VALUES(@UserId, @GameId)";

using (SqlConnection myConnection = new SqlConnection(myConnectionString))
{
    myConnection.Open();

    SqlCommand myCommand = new SqlCommand(insertSql, myConnection);

    myCommand.Parameters.AddWithValue("@UserId", newUserId);
    myCommand.Parameters.AddWithValue("@GameId", newGameId);

    myCommand.ExecuteNonQuery();

    myConnection.Close();
}
```

When I insert into this table, I have an auto_increment int primary key column called GamesProfileId, how can i get the last inserted one after this so I can use that id to insert into another table?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#))

User: [anthonypliu](#) 

[Answer](#)  by [gbn](#) 

Change the insert statement (all one line, split for clarity here) to this for SQL Server 2005+

```
INSERT INTO aspnet_GameProfiles(UserId,GameId)
OUTPUT INSERTED.ID
VALUES(@UserId, @GameId)
```

...or SQL Server 2000

```
INSERT INTO aspnet_GameProfiles(UserId,GameId)
VALUES(@UserId, @GameId);
SELECT SCOPE_IDENTITY()
```

And then

```
Int32 newId = (Int32) myCommand.ExecuteScalar();
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL Server: CASE WHEN OR THEN ELSE END => the OR is not supported](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [case](#) ([Prev Q](#)) ([Next Q](#))

The OR in the WHEN clause of a CASE statement is not supported. How can I do this?

```
CASE ebv.db_no
WHEN 22978 OR 23218 OR 23219 THEN 'WECS 9500'
ELSE 'WECS 9520'
END as wecs_system
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [case](#) ([Prev Q](#)) ([Next Q](#))

User: [werner](#) 

[Answer](#)  by [omg-ponies](#) 

That format requires you to use either:

```
CASE ebv.db_no
WHEN 22978 THEN 'WECS 9500'
WHEN 23218 THEN 'WECS 9500'
WHEN 23219 THEN 'WECS 9500'
ELSE 'WECS 9520'
END as wecs_system
```

Otherwise, use:

```
CASE
WHEN ebv.db_no IN (22978, 23218, 23219) THEN 'WECS 9500'
ELSE 'WECS 9520'
END as wecs_system
```

[Answer](#)  by [darren](#) 

```
CASE
```

```
WHEN ebv.db_no = 22978 OR
    ebv.db_no = 23218 OR
    ebv.db_no = 23219
THEN 'WECS 9500'
ELSE 'WECS 9520'
END as wecs_system
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [case](#) ([Prev Q](#)) ([Next Q](#))

[Q: Not equal <> != operator on NULL](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Could someone please explain the following behavior in SQL?

```
SELECT * FROM MyTable WHERE MyColumn != NULL (0 Results)
SELECT * FROM MyTable WHERE MyColumn <> NULL (0 Results)
SELECT * FROM MyTable WHERE MyColumn IS NOT NULL (568 Results)
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [maxim-gershkovich](#) 

Answer  by [omg-ponies](#) 

<> is Standard SQL-92; != is its equivalent. Both evaluate for values, which NULL is not — NULL is a placeholder to say there is the absence of a value.

Which is why you can only use IS NULL/IS NOT NULL as predicates for such situations.

This behavior is not specific to SQL Server. All standards-compliant SQL dialects work the same way.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: T-SQL: Deleting all duplicate rows but keeping one](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Possible Duplicate:

[SQL - How can I remove duplicate rows?](#)

I have a table with a very large amount of rows. Duplicates are not allowed but due to a problem with how the rows were created I know there are some duplicates in this table. I need to eliminate the extra rows from the perspective of the key columns. Some other columns may have *slightly* different data but I do not care about that. I still need to keep one of these rows however. SELECT DISTINCT won't work because it operates on all columns and I need to suppress duplicates based on the key columns.

How can I delete the extra rows but still keep one efficiently?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [nuit9](#)

Answer by [ben-thul](#)

You didn't say what version you were using, but in SQL 2005 and above, you can use a common table expression with the [OVER Clause](#). It goes a little something like this:

```
WITH cte AS (
    SELECT [foo], [bar],
        row_number() OVER(PARTITION BY foo, bar ORDER BY baz) AS [rn]
    FROM TABLE
)
DELETE cte WHERE [rn] > 1
```

Play around with it and see what you get.

(Edit: In an attempt to be helpful, someone edited the ORDER BY clause within the CTE. To be clear, you can order by anything you want here, it needn't be one of the columns returned by the cte. In fact, a common use-case here is that "foo, bar" are the group identifier and "baz" is some sort of time stamp. In order to keep the latest, you'd do ORDER BY baz desc)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I group by date time column without taking time into consideration](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I have a bunch of product orders and I'm trying to group by the date and sum the quantity for that date. How can I group by the month/day/year without taking the time part into consideration?

3/8/2010 7:42:00 should be grouped with 3/8/2010 4:15:00

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [the-muffin-man](#)

Answer by [oded](#)

Cast/Convert the values to a [Date](#) type for your group by.

```
GROUP BY CAST(myDateTime AS DATE)
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

[Q: Syntax of for-loop in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

What is the syntax for a `FOR` loop in TSQL?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [macho](#) 

[Answer](#)  by [jams](#) 

T-SQL doesn't have a `FOR` loop, it has a `WHILE` loop

[WHILE \(Transact-SQL\)](#) 

```
WHILE Boolean_expression
BEGIN
END
```

[Answer](#)  by [tcks](#) 

There is no `for-loop`, only the `while-loop`:

```
DECLARE @i int = 0
WHILE @i < 20
BEGIN
    SET @i = @i + 1
    /* do some work */
END
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to drop a table if it exists in SQL Server?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

The table name is Scores.

Is it correct to do the following?

```
IF EXISTS(SELECT *
          FROM    dbo.Scores)
    DROP TABLE dbo.Scores
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

User: [tmaster](#) 

[Answer](#)  by [martin-smith](#) 

Is it correct to do the following?

```
IF EXISTS(SELECT *
          FROM    dbo.Scores)
    DROP TABLE dbo.Scores
```

No. That will drop the table only if it contains any rows (and will raise an error if the table does not exist).

Instead, for a permanent table you can use

```
IF OBJECT_ID('dbo.Scores', 'U') IS NOT NULL  
DROP TABLE dbo.Scores;
```

Or, for a temporary table you can use

```
IF OBJECT_ID('tempdb.dbo.#T', 'U') IS NOT NULL  
DROP TABLE #T;
```

For anyone on SQL Server 2016+ I'll point out [this answer](#) - which has a better way.

[Answer](#) by [jveazey](#)

The ANSI SQL/cross-platform way is to use the [INFORMATION_SCHEMA](#), which was specifically designed to query meta data about objects within SQL databases.

```
if exists (select * from INFORMATION_SCHEMA.TABLES where TABLE_NAME = 'Scores' AND TABLE_SCHEMA = 'dbo')  
drop table dbo.Scores;
```

Most modern RDBMS servers provide, at least, basic INFORMATION_SCHEMA support, including: [MySQL](#), [Postgres](#), [Oracle](#), [IBM DB2](#), and [Microsoft SQL Server 7.0 \(and greater\)](#).

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL Server - Return value after INSERT](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I'm trying to get a the key-value back after an INSERT-statement. Example: I've got a table with the attributes name and id. id is a generated value.

```
INSERT INTO table (name) VALUES('bob');
```

Now I want to get the id back in the same step. How is this done?

We're using Microsoft SQL Server 2008.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [melbic](#)

[Answer](#) by [gbn](#)

No need for a separate SELECT...

```
INSERT INTO table (name)  
OUTPUT Inserted.ID  
VALUES('bob');
```

This works for non-IDENTITY columns (such as GUIDs) too

[Answer](#) by [curt](#)

Use SCOPE_IDENTITY() to get the new ID value

```
INSERT INTO table (name) VALUES('bob');
```

```
SELECT SCOPE_IDENTITY()
```

<http://msdn.microsoft.com/en-us/library/ms190315.aspx> 

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

Q: Best practices for SQL varchar column length

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [mysql](#) ([Next Q](#)), [postgresql](#) ([Next Q](#))

Every time I set up a new SQL table or add a new varchar column to an existing table, I am wondering one thing: what is the best value for the length.

So, let's say, you have a column called name of type varchar. So, you have to choose the length. I cannot think of a name > 20 chars, but you will never know. But instead of using 20, I always round up to the next 2^n number. In this case, I would choose 32 as the length. I do that, because from a computer scientist point of view, a number 2^n looks more even to me than other numbers and I'm just assuming that the architecture underneath can handle those numbers slightly better than others.

On the other hand, MSSQL server for example, sets the default length value to 50, when you choose to create a varchar column. That makes me thinking about it. Why 50? Is it just a random number, or based on average column length, or what?

It could also be - or probably is - that different SQL servers implementations (like MySQL, MSSQL, Postgres, ...) have different best column length values.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [mysql](#) ([Next Q](#)), [postgresql](#) ([Next Q](#))

User: [esskar](#) 

[Answer](#)  by [a_horse_with_no_name](#) 

No DBMS I know of has any “optimization” that will make a VARCHAR with a 2^n length perform better than one with a max. length that is not a power of 2.

I think early SQL Server versions actually treated a VARCHAR with length 255 differently than one with a higher maximum length. I don't know if this is still the case.

For almost all DBMS the actual storage that is required is only determined by the number of characters you put into it, not the max. length you define. So from a storage point of view (and most probably a performance one as well) it does not make any difference whether you declare a column as VARCHAR(100) or VACHAR(500).

You should see the max. length provided for a VARCHAR column as a kind of constraint (or business rule) rather than a technical/physical thing.

For PostgreSQL the best setup is to use text without a length restriction and a CHECK CONSTRAINT that limits the number of characters to whatever your business requires.

If that requirement changes, altering the check constraint is much faster than altering the table (because the table does not need to be re-written)

The same can be applied for Oracle and others - in Oracle it would be VARCHAR(4000) instead of text though.

I don't know if there is a physical storage difference between VARCHAR(max) and e.g. VARCHAR(500) in SQL Server. But apparently there is a performance impact when using varchar(max) as compared to varchar(8000).

See [this link](#) (posted by Erwin Brandstetter as a comment)

Edit 2013-09-22

Regarding bigown's comment:

In Postgres versions before 9.2 (which was not available when I wrote the initial answer) a change to the column definition *did* rewrite the whole table, see e.g. [here](#). Since 9.2 this is no longer the case and a quick test confirmed that increasing the column size for a table with 1.2 million rows indeed only took 0.5 seconds.

For Oracle this seems to be true as well, judging by the time it takes to alter a big table's varchar column. But I could not find any reference for that.

For MySQL [the manual says](#) “*In most cases, ALTER TABLE makes a temporary copy of the original table*”. And my own tests confirm that: running an ALTER TABLE on a table with 1.2 million rows (the same as in my test with Postgres) to increase the size of a column took 1.5 minutes. In MySQL however you can *not* use the “workaround” to use a check constraint to limit the number of characters in a column.

For SQL Server I could not find a clear statement on this but the execution time to increase the size of a varchar column (again the 1.2 million rows table from above) indicates that **no** rewrite takes place.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [mysql](#) ([Next Q](#)), [postgresql](#) ([Next Q](#))

[Q: What is the meaning of the prefix N in T-SQL statements?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I have seen prefix N in some insert T-SQL queries. Many people have used N before inserting the value in a table.

I searched, but I was not able to understand what is the purpose of including the N before inserting any strings into the table.

```
INSERT INTO Personnel.Employees  
VALUES(N'29730', N'Philippe', N'Horsford', 20.05, 1),
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [kartik-patel](#)

[Answer](#) by [curt](#)

It's declaring the string as nvarchar data type, rather than varchar

You may have seen Transact-SQL code that passes strings around using an N prefix. This denotes that the subsequent string is in Unicode (the N actually stands for National language character set). Which means that you are passing an NCHAR, NVARCHAR or NTEXT value, as opposed to CHAR, VARCHAR or TEXT.

To quote [from Microsoft](#):

Prefix Unicode character string constants with the letter N. Without the N prefix, the string is converted to the default code page of the database. **This default code page may not recognize certain characters.**

If you want to know the difference between these two data types, see this SO post:

[What is the difference between varchar and nvarchar?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How do I query for all dates greater than a certain date in SQL Server?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

I'm trying:

```
SELECT *  
FROM dbo.March2010 A  
WHERE A.Date >= 2010-04-01;
```

A.Date looks like: 2010-03-04 00:00:00.000

However, this is not working. Can anyone provide a reference for why?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

User: [eric-francis](#) 

[Answer](#)  by [david](#) 

```
select *
from dbo.March2010 A
where A.Date >= Convert(datetime, '2010-04-01' )
```

In your query, 2010-4-01 is treated as a mathematical expression, so in essence it reads

```
select *
from dbo.March2010 A
where A.Date >= 2005;
```

(2010 minus 4 minus 1 is 2005 Converting it to a proper datetime, and using single quotes will fix this issue.)

Technically, the parser might allow you to get away with

```
select *
from dbo.March2010 A
where A.Date >= '2010-04-01'
```

it will do the conversion for you, but in my opinion it is less readable than explicitly converting to a DateTime for the maintenance programmer that will come after you.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

[Q: Search text in stored procedure in SQL Server](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I want to search a text from all my database stored procedures. I use the below SQL:

```
SELECT DISTINCT
    o.name AS Object_Name,
    o.type_desc
FROM sys.sql_modules m
INNER JOIN
    sys.objects o
        ON m.object_id = o.object_id
WHERE m.definition Like '%[ABD]%' ;
```

I want to search for [ABD] in all stored procedures including square brackets, but it's not giving the proper result. How can I change my query to achieve this?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [dharappatel](#) 

[Answer](#)  by [mahmoud-gamal](#) 

Escape the square brackets:

```
...
WHERE m.definition Like '%\[ABD\]%' ESCAPE '\'
```

Then the square brackets will be treated as a string literals not as wild cards.

[Answer](#) by [user27332](#)

Try this request:

Query

```
SELECT name  
FROM sys.procedures  
WHERE Object_definition(object_id) LIKE '%strHell%'
```

[For more details](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

[Q: Recover unsaved SQL query scripts](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [ssms](#) ([Prev Q](#)) ([Next Q](#))

How to recover the unsaved scripts if the SSMS crashes / unsaved tab gets accidentally closed?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [ssms](#) ([Prev Q](#)) ([Next Q](#))

User: [bumblebee](#)

[Answer](#) by [bumblebee](#)

Posting this in case if somebody stumbles into same problem.

Googled for [Retrieve unsaved Scripts](#) and found a solution.

Run the following **select** script. It provides a list of scripts and its time of execution in the last 24 hours. This will be helpful to retrieve the scripts, if we close our query window in SQL Server management studio without saving the script. It works for all executed scripts not only a view or procedure.

```
Use <database>  
SELECT execquery.last_execution_time AS [Date Time], execsql.text AS [Script] FROM sys.dm_exec_query_  
CROSS APPLY sys.dm_exec_sql_text(execquery.sql_handle) AS execsql  
ORDER BY execquery.last_execution_time DESC
```

[Answer](#) by [matt](#)

You *may* be able to find them in one of these locations (depending on the version of Windows you are using).

XP

```
C:\Documents and Settings\YourUsername\My Documents\SQL Server Management Studio\Backup Files\
```

Vista/7

```
C:\Users\YourUsername\Documents\SQL Server Management Studio\Backup Files
```

OR

C:\Users\YourUsername\AppData\Local\Temp

[Googled from this source](#)  and [this source](#) 

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#)), [ssms](#) ([Prev Q](#)) ([Next Q](#))

Q: Efficiently convert rows to columns in sql server

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I'm looking for an efficient way to convert rows to columns in SQL server, I heard that PIVOT is not very fast, and I need to deal with lot of records.

This is my example:

Id	Value	ColumnName
1	John	FirstName
2	2.4	Amount
3	ZH1E4A	PostalCode
4	Fork	LastName
5	857685	AccountNumber

This is my result:

FirstName	Amount	PostalCode	LastName	AccountNumber
John	2.4	ZH1E4A	Fork	857685

How can I build the result?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [tbag](#) 

[Answer](#)  by [bluefeet](#) 

There are several ways that you can transform data from multiple rows into columns. In SQL Server you can use the PIVOT function to transform the data from rows to columns:

[Skip code block](#)

```
select Firstname, Amount, PostalCode, LastName, AccountNumber
from
(
  select value, columnname
  from yourtable
) d
pivot
(
  max(value)
  for columnname in (Firstname, Amount, PostalCode, LastName, AccountNumber)
) piv;
```

See [SQL Fiddle with Demo](#) .

If you have an unknown number of columnnames that you want to transpose, then you can use dynamic SQL:

[Skip code block](#)

```
DECLARE @cols AS NVARCHAR(MAX),
        @query AS NVARCHAR(MAX)

select @cols = STUFF((SELECT ',' + QUOTENAME(ColumnName)
        from yourtable
    
```

```

        group by ColumnName, id
        order by id
    FOR XML PATH(''), TYPE
    ).value('.','NVARCHAR(MAX)')
,1,1,'')

set @query = N'SELECT ' + @cols + N' from
(
    select value, ColumnName
    from yourtable
) x
pivot
(
    max(value)
    for ColumnName in (' + @cols + N')
) p '

exec sp_executesql @query;

```

See [SQL Fiddle with Demo](#).

If you do not want to use the PIVOT function, then you can use an aggregate function with a CASE expression:

```

select
    max(case when columnname = 'FirstName' then value end) Firstname,
    max(case when columnname = 'Amount' then value end) Amount,
    max(case when columnname = 'PostalCode' then value end) PostalCode,
    max(case when columnname = 'LastName' then value end) LastName,
    max(case when columnname = 'AccountNumber' then value end) AccountNumber
from yourtable

```

See [SQL Fiddle with Demo](#).

This could also be completed using multiple joins, but you will need some column to associate each of the rows which you do not have in your sample data. But the basic syntax would be:

[Skip code block](#)

```

select fn.value as FirstName,
    a.value as Amount,
    pc.value as PostalCode,
    ln.value as LastName,
    an.value as AccountNumber
from yourtable fn
left join yourtable a
    on fn.somecol = a.somecol
    and a.columnname = 'Amount'
left join yourtable pc
    on fn.somecol = pc.somecol
    and pc.columnname = 'PostalCode'
left join yourtable ln
    on fn.somecol = ln.somecol
    and ln.columnname = 'LastName'
left join yourtable an
    on fn.somecol = an.somecol
    and an.columnname = 'AccountNumber'
where fn.columnname = 'Firstname'

```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

Q: Rename column SQL Server 2008

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I am using SQL Server 2008 and navicat. I need to rename a column in a table using SQL.

```
ALTER TABLE table_name RENAME COLUMN old_name to new_name;
```

This statement doesn't work.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [serhio-g.-lazin](#) 

[Answer](#)  by [habib](#) 

Use sp_rename

```
EXEC sp_RENAME 'TableName.oldColumnName' , 'NewColumnName', 'COLUMN'
```

See: [SQL SERVER – How to Rename a Column Name or Table Name](#) 

For your case it would be:

```
EXEC sp_RENAME 'table_name.old_name' , 'new_name', 'COLUMN'
```

Remember to use single quotes to enclose your values.

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to Delete using INNER JOIN with SQL Server?](#)

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I want to delete using INNER JOIN in SQL Server 2008.

But I get this error,

Msg 156, Level 15, State 1, Line 15 Incorrect syntax near the keyword 'INNER'.

My code,

```
DELETE FROM WorkRecord2 INNER JOIN Employee ON EmployeeRun=EmployeeNo  
Where Company = '1' AND Date = '2013-05-06'
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [nettoon493](#) 

[Answer](#)  by [bluefeet](#) 

You need to specify what table you are deleting from, here is a version with an alias:

```
DELETE w  
FROM WorkRecord2 w  
INNER JOIN Employee e  
    ON EmployeeRun=EmployeeNo  
Where Company = '1' AND Date = '2013-05-06'
```

[Answer](#)  by [hims056](#) 

Just add the name of the table between DELETE and FROM from where you want to delete records because we have to specify the table to delete. Also remove ORDER BY clause

because there is nothing to order while deleting records.

So your final query should be like this:

```
DELETE WorkRecord2
  FROM WorkRecord2
INNER JOIN Employee
    ON EmployeeRun=EmployeeNo
 WHERE Company = '1'
   AND Date = '2013-05-06';
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

Q: Pad a string with leading zeros so it's 3 characters long in SQL Server 2008

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

I have a string that is up to 3 characters long when it's first created in SQL Server 2008 R2.

I would like to pad it with leading zeros, so if its original value was '1' then the new value would be '001'. Or if its original value was '23' the new value is '023'. Or if its original value is '124' then new value is the same as original value.

I am using SQL Server 2008 R2. How would I do this using T-SQL?

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [sunil](#) 

Answer  by [hogan](#) 

If the field is already a string, this will work

```
SELECT RIGHT('000'+ISNULL(field,''),3)
```

If you want nulls to show as '000'

It might be an integer — then you would want

```
SELECT RIGHT('000'+CAST(field AS VARCHAR(3)),3)
```

Tags: [sql-server](#) ([Prev Q](#)) ([Next Q](#)), [tsql](#) ([Prev Q](#)) ([Next Q](#))

Q: Reset Identity Seed after deleting records in SQL Server

Tags: [sql-server](#) ([Prev Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

I have inserted records in a SQL Server database table. The table had a primary key defined and the auto increment identity seed is set to YES. This is done primarily because in SQL Azure, each table has to have a primary key and Identity Defined.

But, since I have to delete some records from the table, the identity seed for those tables

will be disturbed and the Index column (which is auto-generated with an increment of 1) will get disturbed.

Therefore, kindly let me know how can reset the Identity column after I delete the records so that the column has sequence in ascending numerical order.

P.S. : The Identity column is not used as a FK anywhere in database.

Tags: [sql-server](#) ([Prev Q](#)) ([database](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

User: [romil-n](#) 

[Answer](#)  by [petr-abdulin](#) 

The [DBCC CHECKIDENT](#)  management command is used to reset identity counter. However it's not supported by SQL Azure.

Example: DBCC CHECKIDENT ('[TestTable]', RESEED, 0) GO

A possible workaround is a [SET IDENTITY INSERT](#)  statement which allows you to insert specific values into identity column.

[Answer](#)  by [anil-shah](#) 

```
DBCC CHECKIDENT ('TestTable', RESEED, 0)  
GO
```

Where 0 is identity Start value

Tags: [sql-server](#) ([Prev Q](#)) ([database](#) ([Prev Q](#)) ([Next Q](#)), [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#))

MySQL

[Skip to questions](#),

Wiki by user [paul-dixon](#)

[MySQL](#) is an Open Source [rdbms](#) (Relational Database Management System) that runs as a server providing multi-user access to a number of databases. Its source code is available under the [GNU](#) (General Public License).

[MySQL](#) offers standard database driver connectivity for using MySQL with applications and tools that are compatible with industry standards [odbc](#) and [jdbc](#).

How to ask a good and answerable question

Please see this post: [Tips for asking a good Structured Query Language \(SQL\) question](#) to get Tips on how to ask a question about MySQL - following some of those steps will help people to help you solve your problem.

Pronunciation

Officially it is /maɪ̯, ɛskjuː'ɛl/ ("My S-Q-L"), but is often pronounced /maɪ̯'si:kwəl/ ("My Sequel").

Latest Version : [5.7.11](#) released on 2016-02-05.

Example

```
SELECT * FROM users ORDER BY Name
```

Resources

- [Official homepage](#)
- [Official blog](#)
- [MySQL Developer Zone](#)
- [MySQL documentation](#)
- [Planet MySQL](#)
- [SQL Fiddle](#)
- [PHP MySQL Documentation](#)

MySQL Tutorials

- [Official Tutorial \(5.7\)](#)

Popular questions

- [Hidden Features of MySQL](#) 
- [SQLite vs MySQL](#) 
- [Would you recommend PostgreSQL over MySQL?](#) 
- [Best way to prevent SQL injection in PHP?](#)
- [How to do a regular expression replace in MySQL?](#) 
- [How do you set a default value for a MySQL Datetime column?](#) 

Popular administration tools

- [PhpMyAdmin](#) 
- [MySQL Workbench](#) 
- [HeidiSQL](#) 
- [Navicat](#) 
- [RazorSQL](#) 

Chat Room

- [Stack Overflow chat room for MySQL](#) 

Related tags

- [database](#)
- [mysqli](#) 
- [mysqldump](#) 
- [mysql-connector](#) 
- [mysql2](#) 
- [mysql5](#) 
- [mysql-real-escape-string](#) 
- [mysql-management](#) 
- [mysql-error-1005](#) 
- [mysql-error-1045](#) 
- [mysql-error-1054](#) 
- [mysql-error-1062](#) 
- [mysql-error-1064](#) 
- [mysql-connect](#) 
- [mysqld](#) 
- [mysqlimport](#) 
- [mysql-management](#) 
- [mysql-workbench](#) 



Questions

[Q: Multiple Updates in MySQL](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I know that you can insert multiple rows at once, is there a way to update multiple rows at once (as in, in one query) in MySQL?

Edit: For example I have the following

Name	id	Col1	Col2
Row1	1	6	1
Row2	2	2	3
Row3	3	9	5
Row4	4	16	8

I want to combine all the following Updates into one query

```
UPDATE table SET Col1 = 1 WHERE id = 1;
UPDATE table SET Col1 = 2 WHERE id = 2;
UPDATE table SET Col2 = 3 WHERE id = 3;
UPDATE table SET Col1 = 10 WHERE id = 4;
UPDATE table SET Col2 = 12 WHERE id = 4;
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [teifion](#) 

[Answer](#)  by [michiel-de-mare](#) 

Yes, that's possible - you can use INSERT ... ON DUPLICATE KEY UPDATE.

Using your example:

```
INSERT INTO table (id,Col1,Col2) VALUES (1,1,1),(2,2,3),(3,9,3),(4,10,12)
ON DUPLICATE KEY UPDATE Col1=VALUES(Col1),Col2=VALUES(Col2);
```

[Answer](#)  by [harrison-fisk](#) 

Since you have dynamic values, you need to use an IF or CASE for the columns to be updated. It gets kinda ugly, but it should work.

Using your example, you could do it like:

```
UPDATE table SET Col1 = CASE id
                  WHEN 1 THEN 1
                  WHEN 2 THEN 2
                  WHEN 4 THEN 10
                  ELSE Col1
                  END,
            Col2 = CASE id
                  WHEN 3 THEN 3
                  WHEN 4 THEN 12
                  ELSE Col2
                  END
            WHERE id IN (1, 2, 3, 4);
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Q: How can I prevent SQL-injection in PHP?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Next Q](#))

If user input is inserted without modification into an SQL query, then the application becomes vulnerable to [SQL injection](#), like in the following example:

```
$unsafe_variable = $_POST['user_input'];  
mysql_query("INSERT INTO `table` (`column`) VALUES ('$unsafe_variable')");
```

That's because the user can input something like `value');` `DROP TABLE table;--`, and the query becomes:

```
INSERT INTO `table` (`column`) VALUES('value'); DROP TABLE table;--')
```

What can be done to prevent this from happening?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Next Q](#))

User: [andrew-g.-johnson](#)

[Answer](#) by [theo](#)

Use prepared statements and parameterized queries. These are SQL statements that are sent to and parsed by the database server separately from any parameters. This way it is impossible for an attacker to inject malicious SQL.

You basically have two options to achieve this:

1. Using [PDO](#) (for any supported database driver):

```
$stmt = $pdo->prepare('SELECT * FROM employees WHERE name = :name');  
$stmt->execute(array('name' => $name));  
  
foreach ($stmt as $row) {  
    // do something with $row  
}
```

2. Using [MySQLi](#) (for MySQL):

```
$stmt = $dbConnection->prepare('SELECT * FROM employees WHERE name = ?');  
$stmt->bind_param('s', $name);  
  
$stmt->execute();  
  
$result = $stmt->get_result();  
while ($row = $result->fetch_assoc()) {  
    // do something with $row  
}
```

If you're connecting to a database other than MySQL, there is a driver-specific second option that you can refer to (e.g. `pg_prepare()` and `pg_execute()` for PostgreSQL). PDO is the universal option.

Correctly setting up the connection

Note that when using PDO to access a MySQL database *real* prepared statements are **not**

used by default. To fix this you have to disable the emulation of prepared statements. An example of creating a connection using PDO is:

```
$dbConnection = new PDO('mysql:dbname=dbtest;host=127.0.0.1;charset=utf8', 'user', 'pass');  
$dbConnection->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);  
$dbConnection->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

In the above example the error mode isn't strictly necessary, **but it is advised to add it.** This way the script will not stop with a Fatal Error when something goes wrong. And it gives the developer the chance to catch any error(s) which are thrown as PDOExceptions.

What is **mandatory** however is the first setAttribute() line, which tells PDO to disable emulated prepared statements and use *real* prepared statements. This makes sure the statement and the values aren't parsed by PHP before sending it to the MySQL server (giving a possible attacker no chance to inject malicious SQL).

Although you can set the charset in the options of the constructor, it's important to note that 'older' versions of PHP (< 5.3.6) [silently ignored the charset parameter](#) in the DSN.

Explanation

What happens is that the SQL statement you pass to prepare is parsed and compiled by the database server. By specifying parameters (either a ? or a named parameter like :name in the example above) you tell the database engine where you want to filter on. Then when you call execute, the prepared statement is combined with the parameter values you specify.

The important thing here is that the parameter values are combined with the compiled statement, not an SQL string. SQL injection works by tricking the script into including malicious strings when it creates SQL to send to the database. So by sending the actual SQL separately from the parameters, you limit the risk of ending up with something you didn't intend. Any parameters you send when using a prepared statement will just be treated as strings (although the database engine may do some optimization so parameters may end up as numbers too, of course). In the example above, if the \$name variable contains 'Sarah'; DELETE FROM employees the result would simply be a search for the string "'Sarah'; DELETE FROM employees", and you will not end up with [an empty table](#).

Another benefit with using prepared statements is that if you execute the same statement many times in the same session it will only be parsed and compiled once, giving you some speed gains.

Oh, and since you asked about how to do it for an insert, here's an example (using PDO):

```
$preparedStatement = $db->prepare('INSERT INTO table (column) VALUES (:column)');  
$preparedStatement->execute(array('column' => $unsafeValue));
```

Can Prepared Statements Be Used For Dynamic Queries?

While you can still use prepared statements for the query parameters, the structure of the

dynamic query itself cannot be parametrized and certain query features cannot be parametrized.

For these specific scenarios, the best thing to do is use a whitelist filter that restricts the possible values.

```
// Value whitelist
// $dir can only be 'DESC' or 'ASC'
$dir = !empty($direction) ? 'DESC' : 'ASC';
```

[Answer](#) by [matt-sheppard](#)

You've got two options - escaping the special characters in your `unsafe_variable`, or using a parameterized query. Both would protect you from SQL injection. The parameterized query is considered the better practice, but escaping characters in your variable will require fewer changes.

We'll do the simpler string escaping one first.

```
//Connect
$unsafe_variable = $_POST["user-input"];
$safe_variable = mysql_real_escape_string($unsafe_variable);
mysql_query("INSERT INTO table (column) VALUES ('" . $safe_variable . "')");
//Disconnect
```

See also, the details of the [mysql_real_escape_string](#) function.

Warning:

As of PHP 5.5.0 `mysql_real_escape_string` and the `mysql` extension are *deprecated*. Please use `mysqli` extension and `mysqli::escape_string` function instead

To use the parameterized query, you need to use [MySQLi](#) rather than [MySQL](#) functions. To rewrite your example, we would need something like the following.

[Skip code block](#)

```
<?php
$mysqli = new mysqli("server", "username", "password", "database_name");

// TODO - Check that connection was successful.

$unsafe_variable = $_POST["user-input"];

$stmt = $mysqli->prepare("INSERT INTO table (column) VALUES (?)");

// TODO check that $stmt creation succeeded

// "s" means the database expects a string
$stmt->bind_param("s", $unsafe_variable);

$stmt->execute();

$stmt->close();

$mysqli->close();
?>
```

The key function you'll want to read up on there would be [mysqli::prepare](#).

Also, as others have suggested, you may find it useful/easier to step up a layer of abstraction with something like [PDO](#).

Please note that the case you asked about is a fairly simple one, and that more complex cases may require more complex approaches. In particular:

- If you want to alter the structure of the SQL based on user input, parameterised queries are not going to help, and the escaping required is not covered by `mysql_real_escape_string`. In this kind of case you would be better off passing the user's input through a whitelist to ensure only 'safe' values are allowed through.
- If you use integers from user input in a condition and take the `mysql_real_escape_string` approach, you will suffer from the problem described by [Polynomial](#) in the comments below. This case is trickier because integers would not be surrounded by quotes, so you could deal with by validating that the user input contains only digits.
- There are likely other cases I'm not aware of. You might find <http://webappsec.org/projects/articles/091007.txt> a useful resource on some of the more subtle problems you can encounter.

[Answer](#) by [kibbee](#)

I'd recommend using [PDO](#) (PHP Data Objects) to run parameterized SQL queries.

Not only does this protect against SQL injection, it also speeds up queries.

And by using PDO rather than `mysql_`, `mysqli_`, and `pgsql_` functions, you make your app a little more abstracted from the database, in the rare occurrence that you have to switch database providers.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Next Q](#))

[Q: Subqueries vs joins](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

I refactored a slow section of an application we inherited from another company to use an inner join instead of a subquery like

```
where id in (select id from... )
```

The refactored query runs about 100x faster. (~50 seconds to ~0.3) I expected an improvement, but can anyone explain why it was so drastic? The columns used in the where clause were all indexed. Does SQL execute the query in the where clause once per row or something?

Update - Explain results:

The difference is in the second part of the "where id in ()" query -

```
2  DEPENDENT SUBQUERY submission_tags ref st_tag_id st_tag_id 4 const 2966 Using where
```

vs 1 indexed row with the join:

```
SIMPLE s eq_ref PRIMARY PRIMARY 4 newsladder_production.st.submission_id 1 Using index
```

Thanks everyone!

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [palmsey](#) 

[Answer](#)  by [jeffrey-l-whitledge](#) 

A “correlated subquery” (i.e., one in which the where condition depends on values obtained from the rows of the containing query) will execute once for each row. A non-correlated subquery (one in which the where condition is independent of the containing query) will execute once at the beginning. The SQL engine makes this distinction automatically.

But, yeah, explain-plan will give you the dirty details.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

Q: Comparing date ranges 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

In MySQL, If I have a list of date ranges (range-start and range-end). e.g.

```
10/06/1983 to 14/06/1983  
15/07/1983 to 16/07/1983  
18/07/1983 to 18/07/1983
```

And I want to check if another date range contains ANY of the ranges already in the list, how would I do that?

e.g.

```
06/06/1983 to 18/06/1983 = IN LIST  
10/06/1983 to 11/06/1983 = IN LIST  
14/07/1983 to 14/07/1983 = NOT IN LIST
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [kieran-benton](#) 

[Answer](#)  by [lasse-v.-karlsen](#) 

This is a classical problem, and it's actually easier if you reverse the logic.

Let me give you an example.

I'll post one period of time here, and all the different variations of other periods that overlap in some way.

[Skip code block](#)

-----	compare to this one
----- -----	contained within
----- ----- -----	contained within, equal start
----- ----- -----	contained within, equal end
----- ----- -----	contained within, equal start+end
----- ----- ----- -----	not fully contained, overlaps start
----- ----- ----- -----	not fully contained, overlaps end
----- ----- ----- -----	overlaps start, bigger
----- ----- ----- -----	overlaps end, bigger
----- ----- ----- -----	overlaps entire period

on the other hand, let me post all those that doesn't overlap:

---	-----	compare to this one
---	-----	ends before
---	-----	starts after

So if you simple reduce the comparison to:

```
starts after end
ends before start
```

then you'll find all those that doesn't overlap, and then you'll find all the non-matching periods.

For your final NOT IN LIST example, you can see that it matches those two rules.

You will need to decide wether the following periods are IN or OUTSIDE your ranges:

-----	equal end with start of comparison period
-----	equal start with end of comparison period

If your table has columns called range_end and range_start, here's some simple SQL to retrieve all the matching rows:

```
SELECT *
FROM periods
WHERE NOT (range_start > @check_period_end
            OR range_end < @check_period_start)
```

Note the *NOT* in there. Since the two simple rules finds all the *non-matching* rows, a simple NOT will reverse it to say: *if it's not one of the non-matching rows, it has to be one of the matching ones.*

Applying simple reversal logic here to get rid of the NOT and you'll end up with:

```
SELECT *
FROM periods
WHERE range_start <= @check_period_end
      AND range_end >= @check_period_start
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to use GROUP BY to concatenate strings in MySQL?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Basically the question is how to get from this:

id	string
1	A
1	B

to this:

```
id      string
1          A B
2          C
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [paweł-hajdan](#) 

[Answer](#)  by [scott-noyes](#) 

```
SELECT id, GROUP_CONCAT(string SEPARATOR ' ') FROM table GROUP BY id;
```

http://dev.mysql.com/doc/refman/5.0/en/group-by-functions.html#function_group-concat 

[Answer](#)  by [graeme-perrow](#) 

```
SELECT id, GROUP_CONCAT( string SEPARATOR ' ') FROM table GROUP BY id
```

More details [here](#) 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Getting raw SQL query string from PDO prepared statements](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

Is there a way to get the raw SQL string executed when calling `PDOStatement::execute()` on a prepared statement? For debugging purposes this would be extremely useful.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

User: [wilco](#) 

[Answer](#)  by [bill-karwin](#) 

I assume you mean that you want the final SQL query, with parameter values interpolated into it. I understand that this would be useful for debugging, but it is not the way prepared statements work. Parameters are not combined with a prepared statement on the client-side, so PDO should never have access to the query string combined with its parameters.

The SQL statement is sent to the database server when you do `prepare()`, and the parameters are sent separately when you do `execute()`. MySQL's general query log does show the final SQL with values interpolated after you `execute()`. Below is an excerpt from my general query log. I ran the queries from the mysql CLI, not from PDO, but the principle is the same.

```
081016 16:51:28 2 Query      prepare s1 from 'select * from foo where i = ?'
                      2 Prepare    [2] select * from foo where i = ?
081016 16:51:39 2 Query      set @a =1
081016 16:51:47 2 Query      execute s1 using @a
                      2 Execute   [2] select * from foo where i = 1
```

You can also get what you want if you set the PDO attribute PDO::ATTR_EMULATE_PREPARES. In this mode, PDO interpolate parameters into the SQL query and sends the whole query when you execute(). **This is not a true prepared query.** You will circumvent the benefits of prepared queries by interpolating variables into the SQL string before execute().

Re comment from @afilina:

No, the textual SQL query is *not* combined with the parameters during execution. So there's nothing for PDO to show you.

Internally, if you use PDO::ATTR_EMULATE_PREPARES, PDO makes a copy of the SQL query and interpolates parameter values into it before doing the prepare and execute. But PDO does not expose this modified SQL query.

The PDOStatement object has a property \$queryString, but this is set only in the constructor for the PDOStatement, and it's not updated when the query is rewritten with parameters.

It would be a reasonable feature request for PDO to ask them to expose the rewritten query. But even that wouldn't give you the "complete" query unless you use PDO::ATTR_EMULATE_PREPARES.

This is why I show the workaround above of using the MySQL server's general query log, because in this case even a prepared query with parameter placeholders is rewritten on the server, with parameter values backfilled into the query string. But this is only done during logging, not during query execution.

[Answer](#) by [bigwebguy](#)

[Skip code block](#)

```
/**  
 * Replaces any parameter placeholders in a query with the value of that  
 * parameter. Useful for debugging. Assumes anonymous parameters from  
 * $params are in the same order as specified in $query  
 *  
 * @param string $query The sql query with parameter placeholders  
 * @param array $params The array of substitution parameters  
 * @return string The interpolated query  
 */  
public static function interpolateQuery($query, $params) {  
    $keys = array();  
  
    # build a regular expression for each parameter  
    foreach ($params as $key => $value) {  
        if (is_string($key)) {  
            $keys[] = '/:'.$key.'/';  
        } else {  
            $keys[] = '/[?]/';  
        }  
    }  
  
    $query = preg_replace($keys, $params, $query, 1, $count);  
    #trigger_error('replaced '.$count.' keys');  
  
    return $query;  
}
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

[Q: Best database field type for a URL](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

I need to store a url in a MySQL table. What's the best practice for defining a field that will hold a URL with an undetermined length?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [arkanciscan](#) 

Answer  by [micahwittman](#) 

1. [Lowest common denominator max URL length among popular web browsers: 2,083 \(Internet Explorer\)](#) 

2. <http://dev.mysql.com/doc/refman/5.0/en/char.html> 

Values in VARCHAR columns are variable-length strings. The length can be specified as a value from 0 to 255 before MySQL 5.0.3, and 0 to 65,535 in 5.0.3 and later versions. The effective maximum length of a VARCHAR in MySQL 5.0.3 and later is subject to the maximum row size (65,535 bytes, which is shared among all columns) and the character set used.

3. So ...

< MySQL 5.0.3 use **TEXT**

or

=> MySQL 5.0.3 use **VARCHAR(2083)**

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: Can I concatenate multiple MySQL rows into one field?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Using MySQL, I can do something like:

```
SELECT hobbies FROM peoples_hobbies WHERE person_id = 5;
```

and get:

```
shopping
fishing
coding
```

but instead I just want 1 row, 1 col:

```
shopping, fishing, coding
```

The reason is that I'm selecting multiple values from multiple tables, and after all the joins I've got a lot more rows than I'd like.

I've looked for a function on [MySQL Doc](#)  and it doesn't look like the CONCAT or

CONCAT_WS functions accept result sets, so does anyone here know how to do this?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [dean-rather](#) 

[Answer](#)  by [che](#) 

You can use [GROUP_CONCAT](#) .

As in:

```
SELECT person_id, GROUP_CONCAT(hobbies SEPARATOR ', ')  
FROM peoples_hobbies GROUP BY person_id
```

[Death](#) : As Dag stated in his comment, there is a 1024 byte limit on result. To solve this, run this query before your query:

```
SET group_concat_max_len = 2048
```

Of course, you can change 2048 according to your needs.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I tell when a MySQL table was last updated?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

In the footer of my page, I would like to add something like “last updated the xx/xx/200x” with this date being the last time a certain MySQL table has been updated.

What is the best way to do that? Is there a function to retrieve the last updated date? Should I access to the database every time I need this value?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [rage](#) 

[Answer](#)  by [alnitak](#) 

In later versions of MySQL you can use the `information_schema` database to tell you when another table was updated:

```
SELECT UPDATE_TIME  
FROM information_schema.tables  
WHERE TABLE_SCHEMA = 'dbname'  
AND TABLE_NAME = 'tabname'
```

This does of course mean opening a connection to the database.

An alternative option would be to “touch” a particular file whenever the MySQL table is updated:

On database updates:

- Open your timestamp file in O_RDWR mode

- close it again

or alternatively

- use [touch\(\)](#), the PHP equivalent of the `utimes()` function, to change the file timestamp.

On page display:

- use `stat()` to read back the file modification time.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL/SQL: Group by date only on a Datetime column](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Having a table with a column like: `mydate DATETIME ...`

I have a query such as:

```
SELECT SUM(foo), mydate FROM a_table GROUP BY a_table.mydate;
```

This will group by the full datetime, including hours and minutes. I wish to make the group by, only by the date `YYYY/MM/DD` not by the `YYYY/MM/DD/HH/mm`.

Anyone know how to do this? I can still do it (as I am atm), dynamically in my code, but I'm cleaning trash code and this can be made through the SQL I just can't find out how :(.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [fmsf](#)

[Answer](#) by [michael-haren](#)

Cast the datetime to a date, then GROUP BY using this syntax:

```
SELECT SUM(foo), DATE(mydate) FROM a_table GROUP BY DATE(a_table.mydate);
```

Or you can GROUP BY the alias as @orlandu63 suggested:

```
SELECT SUM(foo), DATE(mydate) DateOnly FROM a_table GROUP BY DateOnly;
```

Though I don't think it'll make any difference to performance, it is a little clearer.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL - find records from one table which don't exist in another](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I've got the following two SQL tables (in MySQL):

[Skip code block](#)

```
Phone_book
+----+-----+-----+
| id | name | phone_number |
+----+-----+-----+
| 1  | John | 1111111111111 |
+----+-----+-----+
| 2  | Jane | 2222222222222 |
+----+-----+-----+


Call
+----+-----+-----+
| id | date | phone_number |
+----+-----+-----+
| 1  | 0945 | 1111111111111 |
+----+-----+-----+
| 2  | 0950 | 2222222222222 |
+----+-----+-----+
| 3  | 1045 | 3333333333333 |
+----+-----+-----+
```

How do I find out which calls were made by people whose phone_number is not in the Phone_book? The desired output would be:

```
Call
+----+-----+-----+
| id | date | phone_number |
+----+-----+-----+
| 3  | 1045 | 3333333333333 |
+----+-----+-----+
```

Any help would be much appreciated.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [philip-morton](#)

[Answer](#) by [alnitak](#)

There's several different ways of doing this, with varying efficiency, depending on how good your query optimiser is, and the relative size of your two tables:

This is the shortest statement, and may be quickest if your phone book is very short:

```
SELECT *
FROM   Call
WHERE  phone_number NOT IN (SELECT phone_number FROM Phone_book)
```

alternatively (thanks to [Alterlife](#))

```
SELECT *
FROM   Call
WHERE  NOT EXISTS
      (SELECT *
       FROM   Phone_book
       WHERE  Phone_book.phone_number = Call.phone_number)
```

or (thanks to [Kieran](#))

```
SELECT *
FROM Call
LEFT OUTER JOIN Phone_Book
ON (Call.phone_number = Phone_book.phone_number)
WHERE Phone_book.phone_number IS NULL
```

(ignoring that, as others have said, it's normally best to select just the columns you want, not `*`)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Skip certain tables with mysqldump](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

Is there a way to restrict certain tables from the mysqldump command?

For example, I'd use the following syntax to dump *only* table1 and table2:

```
mysqldump -u username -p database table1 table2 > database.sql
```

But is there a similar way to dump all the tables *except* table1 and table2? I haven't found anything in the mysqldump documentation, so is brute-force (specifying all the table names) the only way to go?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [zac](#) 

[Answer](#)  by [brian-fisher](#) 

You can use the `--ignore-table` option. So you could do

```
mysqldump -u username -p database --ignore-table=database.table1 > database.sql
```

If you want to ignore multiple tables you can use a simple script like this

[Skip code block](#)

```
#!/bin/bash
PASSWORD=XXXXXX
HOST=XXXXXX
USER=XXXXXX
DATABASE=databasename
DB_FILE=dump.sql
EXCLUDED_TABLES=(
table1
table2
table3
table4
tableN
)

IGNORED_TABLES_STRING=''
for TABLE in "${EXCLUDED_TABLES[@]}"
do :
    IGNORED_TABLES_STRING+=" --ignore-table=${DATABASE}.${TABLE}"
done

echo "Dump structure"
mysqldump --host=${HOST} --user=${USER} --password=${PASSWORD} --single-transaction --no-data ${DATA

echo "Dump content"
mysqldump --host=${HOST} --user=${USER} --password=${PASSWORD} ${DATABASE} ${IGNORED_TABLES_STRING} >
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL Query GROUP BY day / month / year](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Next Q](#))

Is it possible I make a simple query to count how many records I have in a determined period of time like a Year, month or day, having a `TIMESTAMP` field, like:

```
SELECT COUNT(id)
FROM stats
WHERE record_date.YEAR = 2009
GROUP BY record_date.YEAR
```

Or even:

```
SELECT COUNT(id)
FROM stats
GROUP BY record_date.YEAR, record_date.MONTH
```

To have a monthly statistic.

Thanks!

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Next Q](#))

User: [fernando-barrocal](#) 

[Answer](#)  by [codelogic](#) 

```
GROUP BY YEAR(record_date), MONTH(record_date)
```

Check out the [date and time functions](#)  in MySQL.

[Answer](#)  by [andriy-m](#) 

```
GROUP BY DATE_FORMAT(record_date, '%Y%m')
```

Note (primarily, to potential downvoters). Presently, this may not be as efficient as other suggestions. Still, I leave it as an alternative, and a one, too, that can serve in seeing how faster other solutions are. (For you can't really tell fast from slow until you see the difference.) Also, as time goes on, changes could be made to MySQL's engine with regard to optimisation so as to make this solution, at some (perhaps, not so distant) point in future, to become quite comparable in efficiency with most others.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Next Q](#))

[Q: How to create a simple 'Hello World' module in Magento?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

How can the following be accomplish in Magento?

- Display a “Hello World” message using a controller/view/model approach. So, if I went to `http://example.com/myController` it would show the string ‘Hello World’. Being able to show this string within the template of my website (for example, the header, footer, etc.) will be a bonus.
- How to add a method to this controller (or a new controller if necessary), which interacts with a model, and performs the query `Select * FROM articles where id='10'` and returns the row (containing the columns `id`, `title`, `content`) to the controller? And then use the controller to include a view, which would display this row. So going to `http://example.com/myController/show_row` (or something similar) would display the row within a view. (No need to be fancy, just a echo `$row->id;` or something similar would work.)

Any other information about Magento’s code structure will also be very helpful.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

User: [click-upvote](#) 

[Answer](#)  by [alan-storm](#) 

First and foremost, I highly recommend you buy the [PDF/E-Book from PHP Architect](#). It’s US\$20, but is the only straightforward “Here’s how Magento works” resource I’ve been able to find. I’ve also started writing [Magento tutorials at my own website](#).

Second, if you have a choice, and aren’t an experienced programmer or don’t have access to an experienced programmer (ideally in PHP and Java), **pick another cart**. Magento is well engineered, but it was engineered to be a shopping cart solution that other programmers can build modules on top of. It was not engineered to be easily understood by people who are smart, but aren’t programmers.

Third, Magento MVC is very different from the [Ruby on Rails](#), [Django](#), [CodeIgniter](#), [CakePHP](#), etc. MVC model that’s popular with PHP developers these days. I think it’s based on the [Zend](#) model, and the whole thing is very Java OOP-like. There’s **two** controllers you need to be concerned about. The module/frontName controller, and then the MVC controller.

Fourth, the Magento application itself is built using the same module system you’ll be using, so poking around the core code is a useful learning tactic. Also, a lot of what you’ll be doing with Magento is **overriding** existing classes. What I’m covering here is **creating** new functionality, not overriding. Keep this in mind when you’re looking at the code samples out there.

I’m going to start with your first question, showing you how to setup a controller/router to respond to a specific URL. This will be a small novel. I might have time later for the model/template related topics, but for now, I don’t. I will, however, briefly speak to your SQL question.

Magento uses an [EAV](#) database architecture. Whenever possible, try to use the model objects the system provides to get the information you need. I know it’s all there in the SQL tables, but it’s best not to think of grabbing data using raw SQL queries, or you’ll go

mad.

Final disclaimer. I've been using Magento for about two or three weeks, so caveat emptor. This is an exercise to get this straight in my head as much as it is to help Stack Overflow.

Create a module

All additions and customizations to Magento are done through modules. So, the first thing you'll need to do is create a new module. Create an XML file in app/modules named as follows

```
cd /path/to/store/app  
touch etc/modules/MyCompanyName_Helloworld.xml
```

```
<?xml version="1.0"?>  
<config>  
    <modules>  
        <MyCompanyName_Helloworld>  
            <active>true</active>  
            <codePool>local</codePool>  
        </MyCompanyName_Helloworld>  
    </modules>  
</config>
```

MyCompanyName is a unique namespace for your modifications, it doesn't have to be your company's name, but that the recommended convention by magento. HelloWorld is the name of your module.

Clear the application cache

Now that the module file is in place, we'll need to let Magento know about it (and check our work). In the admin application

1. Go to System->Cache Management
2. Select Refresh from the All Cache menu
3. Click Save Cache settings

Now, we make sure that Magento knows about the module

1. Go to System->Configuration
2. Click Advanced
3. In the "Disable modules output" setting box, look for your new module named "MyCompanyName_Helloworld"

If you can live with the performance slow down, you might want to turn off the application cache while developing/learning. Nothing is more frustrating than forgetting to clear out the cache and wondering why your changes aren't showing up.

Setup the directory structure

Next, we'll need to setup a directory structure for the module. You won't need all these directories, but there's no harm in setting them all up now.

```
mkdir -p app/code/local/MyCompanyName/HelloWorld/Block  
mkdir -p app/code/local/MyCompanyName/HelloWorld/controllers  
mkdir -p app/code/local/MyCompanyName/HelloWorld/Model  
mkdir -p app/code/local/MyCompanyName/HelloWorld/Helper  
mkdir -p app/code/local/MyCompanyName/HelloWorld/etc  
mkdir -p app/code/local/MyCompanyName/HelloWorld/sql
```

And add a configuration file

```
touch app/code/local/MyCompanyName/HelloWorld/etc/config.xml
```

and inside the configuration file, add the following, which is essentially a “blank” configuration.

```
<?xml version="1.0"?>  
<config>  
    <modules>  
        <MyCompanyName_HelloWorld>  
            <version>0.1.0</version>  
        </MyCompanyName_HelloWorld>  
    </modules>  
</config>
```

Oversimplifying things, this configuration file will let you tell Magento what code you want to run.

Setting up the router

Next, we need to setup the module’s routers. This will let the system know that we’re handling any URLs in the form of

```
http://example.com/magento/index.php/helloworld
```

So, in your configuration file, add the following section.

Skip code block

```
<config>  
<!-- ... -->  
    <frontend>  
        <routers>  
            <!-- the <helloworld> tagname appears to be arbitrary, but by  
            convention is should match the frontName tag below-->  
            <helloworld>  
                <use>standard</use>  
                <args>  
                    <module>MyCompanyName_HelloWorld</module>  
                    <frontName>helloworld</frontName>  
                </args>  
            </helloworld>  
        </routers>  
    </frontend>  
<!-- ... -->  
</config>
```

What you’re saying here is “any URL with the frontName of helloworld ...”

```
http://example.com/magento/index.php/helloworld
```

should use the frontName controller MyCompanyName_HelloWorld”.

So, with the above configuration in place, when you load the helloworld page above, you’ll get a 404 page. That’s because we haven’t created a file for our controller. Let’s do that now.

```
touch app/code/local/MyCompanyName/HelloWorld/controllers/IndexController.php
```

Now try loading the page. Progress! Instead of a 404, you'll get a PHP/Magento exception

```
Controller file was loaded but class does not exist
```

So, open the file we just created, and paste in the following code. The name of the class needs to be based on the name you provided in your router.

```
class MyCompanyName_Helloworld_IndexController extends Mage_Core_Controller_Front_Action{
    public function indexAction(){
        echo "We're echoing just to show that this is what's called, normally you'd have some kind of
    }
}
```

What we've just setup is the module/frontName controller. This is the default controller and the default action of the module. If you want to add controllers or actions, you have to remember that the tree first part of a Magento URL are immutable they will always go this way

<http://example.com/magento/index.php/frontName/controllerName/actionName>

So if you want to match this url

```
http://example.com/magento/index.php/helloworld/foo
```

You will have to have a FooController, which you can do this way :

```
touch app/code/local/MyCompanyName/Helloworld/controllers/FooController.php
```

[Skip code block](#)

```
class MyCompanyName_Helloworld_FooController extends Mage_Core_Controller_Front_Action{
    public function indexAction(){
        echo 'Foo Index Action';
    }

    public function addAction(){
        echo 'Foo add Action';
    }

    public function deleteAction(){
        echo 'Foo delete Action';
    }
}
```

Please note that the default controller IndexController and the default action indexAction can be implicit but have to be explicit if something come after it. So

<http://example.com/magento/index.php/helloworld/foo> will match the controller FooController and the action indexAction and NOT the action fooAction of the IndexController. If you want to have a fooAction, in the controller IndexController you then have to call this controller explicitly like this way :

<http://example.com/magento/index.php/helloworld/index/foo> because the second part of the url is and will always be the controllerName. This behaviour is an inheritance of the Zend Framework bundled in Magento.

You should now be able to hit the following URLs and see the results of your echo statements

```
http://example.com/magento/index.php/helloworld/foo
http://example.com/magento/index.php/helloworld/foo/add
http://example.com/magento/index.php/helloworld/foo/delete
```

So, that should give you a basic idea on how Magento dispatches to a controller. From here I'd recommended poking at the existing Magento controller classes to see how

models and the template/layout system should be used.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

[Q: What's faster, SELECT DISTINCT or GROUP BY in MySQL?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

If I have a table

```
CREATE TABLE users (
    id int(10) unsigned NOT NULL auto_increment,
    name varchar(255) NOT NULL,
    profession varchar(255) NOT NULL,
    employer varchar(255) NOT NULL,
    PRIMARY KEY (id)
)
```

and I want to get all unique values of profession field, what would be faster (or recommended):

```
SELECT DISTINCT u.profession FROM users u
```

or

```
SELECT u.profession FROM users u GROUP BY u.profession
```

?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [vava](#) 

[Answer](#)  by [squarecog](#) 

They are essentially equivalent to each other (in fact this is how some databases implement DISTINCT under the hood).

If one of them is faster, it's going to be DISTINCT. This is because, although the two are the same, a query optimizer would have to catch the fact that your GROUP BY is not taking advantage of any group members, just their keys. DISTINCT makes this explicit, so you can get away with a slightly dumber optimizer.

When in doubt, test!

[Answer](#)  by [quassnoi](#) 

If you have an index on profession, these two are synonyms.

If you don't, then use DISTINCT.

GROUP BY in MySQL sorts results. You can even do:

```
SELECT u.profession FROM users u GROUP BY u.profession DESC
```

and get your professions sorted in DESC order.

DISTINCT creates a temporary table and uses it for storing duplicates. GROUP BY does the

same, but sorts the distinct results afterwards.

So

```
SELECT DISTINCT u.profession FROM users u
```

is faster, if you don't have an index on profession.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

Q: How can I SELECT rows with MAX(Column value), DISTINCT by another column in SQL?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#))

My table is:

[Skip code block](#)

id	home	datetime	player	resource
1	10	04/03/2009	john	399
2	11	04/03/2009	juliet	244
5	12	04/03/2009	borat	555
3	10	03/03/2009	john	300
4	11	03/03/2009	juliet	200
6	12	03/03/2009	borat	500
7	13	24/12/2008	borat	600
8	13	01/01/2009	borat	700

I need to select each distinct home holding the maximum value of datetime.

Result would be:

id	home	datetime	player	resource
1	10	04/03/2009	john	399
2	11	04/03/2009	juliet	244
5	12	04/03/2009	borat	555
8	13	01/01/2009	borat	700

I have tried:

```
-- 1 ..by the MySQL manual:  
  
SELECT DISTINCT home, id, datetime as dt, player, resource  
  FROM topten t1  
 WHERE datetime = (SELECT MAX(t2.datetime) FROM topten t2  
                   GROUP BY home )  
 GROUP BY datetime  
 ORDER BY datetime DESC
```

Doesn't work. Result-set has 130 rows although database holds 187. Result includes some duplicates of home.

```
-- 2 ..join  
  
SELECT s1.id, s1.home, s1.datetime, s1.player, s1.resource  
  FROM topten s1 JOIN  
(SELECT id, MAX(datetime) AS dt  
    FROM topten  
   GROUP BY id) AS s2
```

```
ON s1.id = s2.id  
ORDER BY datetime
```

Nope. Gives all the records.

```
-- 3 ..something exotic:
```

With various results.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#))

User: [kaptah](#) 

[Answer](#)  by [michael-la-voie](#) 

You are so close! All you need to do is select BOTH the home and it's max date time, then join back to the topten table on BOTH fields:

```
SELECT tt.*  
FROM topten tt  
INNER JOIN  
    (SELECT home, MAX(datetime) AS MaxDateTime  
     FROM topten  
     GROUP BY home) groupedtt  
ON tt.home = groupedtt.home  
AND tt.datetime = groupedtt.MaxDateTime
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#))

[Q: Best data type for currency values](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

What is the best SQL data type for currency values? I'm using MySQL but would prefer a database independent type.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [brian-fisher](#) 

[Answer](#)  by [kibbee](#) 

Something like Decimal(19, 4) usually works pretty well in most cases. You can adjust the scale and precision to fit the needs of the numbers you need to store. Even in SQL Server, I tend not to use "money" as it's non-standard.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to change the default collation of a table?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

```
create table check2(f1 varchar(20), f2 varchar(20));
```

creates a table with the default collation latin1_general_ci;

```
alter table check2 collate latin1_general_ci;
show full columns from check2;
```

shows the individual collation of the columns as ‘latin1_general_ci’.

Then what is the effect of the alter table command?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [dharma0us](#) 

[Answer](#)  by [nikki-erwin-ramirez](#) 

To change the default character set and collation of a table *including those of existing columns* (note the **convert to** clause):

```
alter table <some_table> convert to character set utf8 collate utf8_unicode_ci;
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MYSQL OR vs IN performance](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

I am wondering if there is any difference in regards to performance between the following

```
SELECT... FROM... WHERE someFIELD IN(1,2,3,4)
SELECT... FROM... WHERE someFIELD between 0 AND 5
SELECT... FROM... WHERE someFIELD = 1 OR someFIELD = 2 OR someFIELD = 3...
```

or will MySQL optimize the SQL in the same way compilers will optimize code ?

EDIT: Changed the AND's to OR's for the reason stated in the comments.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

User: [scott](#) 

[Answer](#)  by [sunseeker](#) 

I needed to know this for sure, so I benchmarked both methods. I consistently found IN to be much faster than using OR.

Do not believe people who give their “opinion”, science is all about testing and evidence.

I ran a loop of 1000x the equivalent queries (for consistency, I used sql_no_cache):

IN: 2.34969592094s

OR: 5.83781504631s

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL INSERT INTO table VALUES.. vs INSERT INTO table](#)

[SET](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

What is main difference between `INSERT INTO table VALUES ...` and `INSERT INTO table SET ...`?

Example:

```
INSERT INTO table (a, b, c) VALUES (1,2,3)  
INSERT INTO table SET a=1, b=2, c=3
```

And what about performance of these two?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

User: [irmantas](#)

Answer by [vinko-vrsalovic](#)

As far as I can tell, both syntaxes are equivalent. The first is SQL standard, the second is MySQL's extension.

So they should be exactly equivalent performance wise.

<http://dev.mysql.com/doc/refman/5.6/en/insert.html> says:

INSERT inserts new rows into an existing table. The `INSERT ... VALUES` and `INSERT ... SET` forms of the statement insert rows based on explicitly specified values. The `INSERT ... SELECT` form inserts rows selected from another table or tables.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL dump by query](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

Is it possible to do mysqldump by single SQL query?

I mean to dump the **whole** database, like phpmyadmin does when you do export to SQL

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [jakub-arnold](#)

Answer by [zak-](#)

not mysqldump, but mysql cli...

```
mysql -e "select * from myTable" -u myuser -pxxxxxxxxxx mydatabase
```

you can redirect it out to a file if you want :

```
mysql -e "select * from myTable" -u myuser -pxxxxxxxxxx mydatabase > mydumpfile.txt
```

Update: Original post asked if he could dump from the database by query. What he asked and what he meant were different. He really wanted to just mysqldump all tables.

```
mysqldump --tables myTable --where="id < 1000"
```

[Answer](#)  by [thomas-ahle](#) 

This should work

```
mysqldump --databases X --tables Y --where="1 limit 1000000"
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: Using column alias in WHERE clause of MySQL query produces an error](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

The query I'm running is as follows, however I'm getting this error:

```
#1054 - Unknown column 'guaranteed_postcode' in 'IN/ALL/ANY subquery'
```

[Skip code block](#)

```
SELECT `users`.`first_name`, `users`.`last_name`, `users`.`email`,  
SUBSTRING(`locations`.`raw`, -6, 4) AS `guaranteed_postcode`  
FROM `users` LEFT OUTER JOIN `locations`  
ON `users`.`id` = `locations`.`user_id`  
WHERE `guaranteed_postcode` NOT IN #this is where the fake col is being used  
(  
    SELECT `postcode` FROM `postcodes` WHERE `region` IN  
(  
        'australia'  
    )  
)
```

My question is: why am I unable to use a fake column in the where clause of the same DB query?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [james](#)

Answer by [victor-hugo](#)

You can only use column aliases in GROUP BY, ORDER BY, or HAVING clauses.

Standard SQL doesn't allow you to refer to a column alias in a WHERE clause. This restriction is imposed because when the WHERE code is executed, the column value may not yet be determined.

Copied from [MySQL documentation](#)

As pointed in the comments, using HAVING instead may do the work. Make sure to give a read at this [WHERE vs HAVING](#) though.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL: @variable vs. variable. Whats the difference?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

In another question I posted someone told me that there is a difference between:

```
@variable
```

and:

```
variable
```

in MySQL. He also mentioned how MSSQL has batch scope and MySQL has session scope. Can someone elaborate on this for me?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [djtriplethreat](#) 

[Answer](#)  by [quassnoi](#) 

MySQL has the concept of [*user-defined variables*](#) .

They are loosely typed variables that may be initialized somewhere in a session and keep their value until the session ends.

They are prepended with an @ sign, like this: @var

You can initialize this variable with a SET statement or inside in a query:

```
SET @var = 1
SELECT @var2 := 2
```

When you develop a stored procedure in MySQL, you can pass the input parameters and declare the local variables:

[Skip code block](#)

```
DELIMITER //
CREATE PROCEDURE prc_test (var INT)
BEGIN
    DECLARE var2 INT;
    SET var2 = 1;
    SELECT var2;
END;
// 
DELIMITER ;
```

These variables are not prepended with any prefixes.

The difference between a procedure variable and a session-specific user-defined variable is that procedure variable is reinitialized to NULL each time the procedure is called, while the session-specific variable is not:

[Skip code block](#)

```
CREATE PROCEDURE prc_test ()
BEGIN
    DECLARE var2 INT DEFAULT 1;
    SET var2 := var2 + 1;
    SET @var2 := @var2 + 1;
    SELECT var2, @var2;
END;

SET @var2 = 1;
CALL prc_test();
var2  @var2
---  ---
2      2

CALL prc_test();
var2  @var2
```

```
--  --
2    3

CALL prc_test();

var2  @var2
--  --
2    4
```

As you can see, `var2` (procedure variable) is reinitialized each time the procedure is called, while `@var2` (session-specific variable) is not.

(In addition to user-defined variables, MySQL *also* has some predefined “system variables”, which may be “global variables” such as `@@global.port` or “session variables” such as `@@session.sql_mode`; these “session variables” are unrelated to session-specific user-defined variables.)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: INNER JOIN ON vs WHERE clause](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

For simplicity, assume all relevant fields are NOT NULL.

You can do:

```
SELECT
  table1.this, table2.that, table2.somethingelse
FROM
  table1, table2
WHERE
  table1.foreignkey = table2.primarykey
  AND (some other conditions)
```

Or else:

```
SELECT
  table1.this, table2.that, table2.somethingelse
FROM
  table1 INNER JOIN table2
  ON table1.foreignkey = table2.primarykey
WHERE
  (some other conditions)
```

Do these two work on the same way in MySQL?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

User: [jccyc](#) 

[Answer](#)  by [quassnoi](#) 

`INNER JOIN` is ANSI syntax which you should use.

It is generally considered more readable, especially when you join lots of tables.

It can also be easily replaced with an `OUTER JOIN` whenever a need arises.

The `WHERE` syntax is more relational model oriented.

A result of two tables JOIN'ed is a cartesian product of the tables to which a filter is applied which selects only those rows with joining columns matching.

It's easier to see this with the WHERE syntax.

As for your example, in MySQL (and in SQL generally) these two queries are synonyms.

Also note that MySQL also has a STRAIGHT_JOIN clause.

Using this clause, you can control the JOIN order: which table is scanned in the outer loop and which one is in the inner loop.

You cannot control this in MySQL using WHERE syntax.

[Answer](#) by [carl-manaster](#)

Others have pointed out that INNER JOIN helps human readability, and that's a top priority; I agree. Let me try to explain *why* the join syntax is more readable.

A basic SELECT query is this:

```
SELECT stuff  
FROM tables  
WHERE conditions
```

The SELECT clause tells us **what** we're getting back; the FROM clause tells us **where** we're getting it from, and the WHERE clause tells us **which** ones we're getting.

JOIN is a statement about the tables, how they are bound together (conceptually, actually, into a single table). Any query elements that control the tables - where we're getting stuff from - semantically belong to the FROM clause (and of course, that's where JOIN elements go). Putting joining-elements into the WHERE clause conflates the **which** and the **where-from**; that's why the JOIN syntax is preferred.

[Answer](#) by [rafidheen](#)

Applying conditional statements in ON / WHERE

Here I have explained about the logical query processing steps.

Reference : Inside Microsoft® SQL Server™ 2005 T-SQL Querying

Publisher: Microsoft Press

Pub Date: March 07, 2006

Print ISBN-10: 0-7356-2313-9

Print ISBN-13: 978-0-7356-2313-2

Pages: 640

[Inside Microsoft® SQL Server™ 2005 T-SQL Querying](#)

```
(8)  SELECT (9) DISTINCT (11) TOP <top_specification> <select_list>  
(1)   FROM <left_table>  
(3)     <join_type> JOIN <right_table>  
(2)       ON <join_condition>  
(4)   WHERE <where_condition>  
(5)   GROUP BY <group_by_list>  
(6)   WITH {CUBE | ROLLUP}  
(7)   HAVING <having_condition>  
(10)  ORDER BY <order_by_list>
```

The first noticeable aspect of SQL that is different than other programming languages is the order in which the code is processed. In most programming languages, the code is processed in the order in which it is written. In SQL, the first clause that is processed is the FROM clause, while the SELECT clause, which appears first, is processed almost last.

Each step generates a virtual table that is used as the input to the following step. These virtual tables are not available to the caller (client application or outer query). Only the table generated by the final step is returned to the caller. If a certain clause is not specified in a query, the corresponding step is simply skipped.

Brief Description of Logical Query Processing Phases

Don't worry too much if the description of the steps doesn't seem to make much sense for now. These are provided as a reference. Sections that come after the scenario example will cover the steps in much more detail.

1. FROM: A Cartesian product (cross join) is performed between the first two tables in the FROM clause, and as a result, virtual table VT1 is generated.
2. ON: The ON filter is applied to VT1. Only rows for which the <join_condition> is TRUE are inserted to VT2.
3. OUTER (join): If an OUTER JOIN is specified (as opposed to a CROSS JOIN or an INNER JOIN), rows from the preserved table or tables for which a match was not found are added to the rows from VT2 as outer rows, generating VT3. If more than two tables appear in the FROM clause, steps 1 through 3 are applied repeatedly between the result of the last join and the next table in the FROM clause until all tables are processed.
4. WHERE: The WHERE filter is applied to VT3. Only rows for which the <where_condition> is TRUE are inserted to VT4.
5. GROUP BY: The rows from VT4 are arranged in groups based on the column list specified in the GROUP BY clause. VT5 is generated.
6. CUBE | ROLLUP: Supergroups (groups of groups) are added to the rows from VT5, generating VT6.
7. HAVING: The HAVING filter is applied to VT6. Only groups for which the <having_condition> is TRUE are inserted to VT7.
8. SELECT: The SELECT list is processed, generating VT8.
9. DISTINCT: Duplicate rows are removed from VT8. VT9 is generated.
10. ORDER BY: The rows from VT9 are sorted according to the column list specified in the ORDER BY clause. A cursor is generated (VC10).
11. TOP: The specified number or percentage of rows is selected from the beginning of VC10. Table VT11 is generated and returned to the caller.

Therefore, (INNER JOIN) ON will filter the data (the data count of VT will be reduced here itself) before applying WHERE clause. The subsequent join conditions will be executed with filtered data which improves performance. After that only the WHERE condition will apply filter conditions.

(Applying conditional statements in ON / WHERE will not make much difference in few cases. This depends how many tables you have joined and number of rows available in each join tables)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL LIKE IN\(\)?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

My current query looks like this:

```
SELECT * FROM fiberbox f WHERE f.fiberBox LIKE '%1740 %' OR f.fiberBox LIKE '%1938 %' OR f.fiberBox L
```

I did some looking around and can't find anything similar to a LIKE IN() - I envision it working like this:

```
SELECT * FROM fiberbox f WHERE f.fiberbox LIKE IN('%140 %', '%1938 %', '%1940 %')
```

Any ideas? Am I just thinking of the problem the wrong way - some obscure command I've never seen.

MySQL 5.0.77-community-log

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [michael-wales](#) 

[Answer](#)  by [paul-dixon](#) 

A [REGEXP](#)  might be more efficient, but you'd have to benchmark it to be sure, e.g.

```
SELECT * from fiberbox where field REGEXP '1740|1938|1940';
```

[Answer](#)  by [david-carroll](#) 

Paul Dixon's answer worked brilliantly for me. To add to this, here are some things I observed for those interested in using REGEXP:

To Accomplish multiple LIKE filters with Wildcards:

```
SELECT * FROM fiberbox WHERE field LIKE '%1740 %'  
                      OR field LIKE '%1938 %'  
                      OR field LIKE '%1940 %';
```

Use REGEXP Alternative:

```
SELECT * FROM fiberbox WHERE field REGEXP '1740 |1938 |1940 ';
```

Values within REGEXP quotes and between the | (OR) operator are treated as wildcards. Typically, REGEXP will require wildcard expressions such as (*.*)1740 (*.*) to work as

%1740 %.

If you need more control over placement of the wildcard, use some of these variants:

To Accomplish LIKE with Controlled Wildcard Placement:

```
SELECT * FROM fiberbox WHERE field LIKE '1740 %'  
      OR field LIKE '%1938 '  
      OR field LIKE '%1940 % test';
```

Use:

```
SELECT * FROM fiberbox WHERE field REGEXP '^1740 |1938 $|1940 (.*) test';
```

- Placing ^ in front of the value indicates start of the line.
- Placing \$ after the value indicates end of line.
- Placing (.*) behaves much like the % wildcard.
- The . indicates any single character, except line breaks. Placing . inside () with * (.*) adds a repeating pattern indicating any number of characters till end of line.

There are more efficient ways to narrow down specific matches, but that requires more review of Regular Expressions. NOTE: Not all regex patterns appear to work in MySQL statements. You'll need to test your patterns and see what works.

Finally, To Accomplish Multiple LIKE and NOT LIKE filters:

```
SELECT * FROM fiberbox WHERE field LIKE '%1740 %'  
      OR field LIKE '%1938 %'  
      OR field NOT LIKE '%1940 %'  
      OR field NOT LIKE 'test %'  
      OR field = '9999';
```

Use REGEXP Alternative:

```
SELECT * FROM fiberbox WHERE field REGEXP '1740 |1938 |^9999$'  
      OR field NOT REGEXP '1940 |^test ';
```

OR Mixed Alternative:

```
SELECT * FROM fiberbox WHERE field REGEXP '1740 |1938 '  
      OR field NOT REGEXP '1940 |^test '  
      OR field NOT LIKE 'test %'  
      OR field = '9999';
```

Notice I separated the NOT set in a separate WHERE filter. I experimented with using negating patterns, forward looking patterns, and so on. However, these expressions did not appear to yield the desired results. In the first example above, I use ^9999\$ to indicate exact match. This allows you to add specific matches with wildcard matches in the same expression. However, you can also mix these types of statements as you can see in the second example listed.

Regarding performance, I ran some minor tests against an existing table and found no differences between my variations. However, I imagine performance could be an issue with bigger databases, larger fields, greater record counts, and more complex filters.

As always, use logic above as it makes sense.

If you want to learn more about regular expressions, I recommend www.regular-expressions.info

expressions.info as a good reference site.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Q: How do I update if exists, insert if not (AKA “upsert” or “merge”) in MySQL?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Is there an easy way to `INSERT` a row when it does not exist, or to `UPDATE` if it exists, using one MySQL query?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [blub](#) 

[Answer](#)  by [chaos](#) 

Yes, [`INSERT... ON DUPLICATE KEY UPDATE`](#). For example:

```
INSERT INTO `usage`
(`thing_id`, `times_used`, `first_time_used`)
VALUES
(4815162342, 1, NOW())
ON DUPLICATE KEY UPDATE
`times_used` = `times_used` + 1
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Q: MySQL, better to insert NULL or empty string?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have a form on a website which has a lot of different fields. Some of the fields are optional while some are mandatory. In my DB I have a table which holds all these values, is it better practice to insert a `NULL` value or an empty string into the DB columns where the user didn't put any data?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [roflwaffle](#) 

[Answer](#)  by [quassnoi](#) 

By using `NULL` you can distinguish between “put no data” and “put empty data”.

Some more differences:

- A `LENGTH` of `NULL` is `NULL`, a `LENGTH` of an empty string is `0`.
- `NULLS` are sorted before the empty strings.
- `COUNT(message)` will count empty strings but not `NULLS`

- You can search for an empty string using a bound variable but not for a NULL. This query:

```
SELECT *
FROM mytable
WHERE mytext = ?
```

will never match a NULL in `mytext`, whatever value you pass from the client. To match NULLs, you'll have to use other query:

```
SELECT *
FROM mytable
WHERE mytext IS NULL
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Simple way to calculate median with MySQL](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

What's the simplest (and hopefully not too slow) way to calculate the median with MySQL? I've used `AVG(x)` for finding the mean, but I'm having a hard time finding a simple way of calculating the median. For now, I'm returning all the rows to PHP, doing a sort, and then picking the middle row, but surely there must be some simple way of doing it in a single MySQL query.

Example data:

id	val
1	4
2	7
3	2
4	2
5	9
6	8
7	3

Sorting on `val` gives 2 2 3 4 7 8 9, so the median should be 4, versus `SELECT AVG(val)` which == 5.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [davr](#) 

[Answer](#)  by [velcrow](#) 

The problem with the proposed solution (TheJacobTaylor) is runtime. Joining the table to itself is slow as molasses for large datasets. My proposed alternative runs in mysql, has awesome runtime, uses an explicit ORDER BY statement, so you don't have to hope your indexes ordered it properly to give a correct result, and is easy to unroll the query to debug.

[Skip code block](#)

```
SELECT avg(t1.val) as median_val FROM (
SELECT @rownum:=@rownum+1 as `row_number`, d.val
FROM data d, (SELECT @rownum:=0) r
WHERE 1
```

```

-put some where clause here
ORDER BY d.val
) as t1,
(
  SELECT count(*) as total_rows
  FROM data d
  WHERE 1
-put same where clause here
) as t2
WHERE 1
AND t1.row_number in ( floor((total_rows+1)/2), floor((total_rows+2)/2) );

```

[edit] Added avg() around t1.val and row_number in(...) to correctly produce a median when there are an even number of records. Reasoning:

```

SELECT floor((3+1)/2),floor((3+2)/2);#total_rows is 3, so avg row_numbers 2 and 2
SELECT floor((4+1)/2),floor((4+2)/2);#total_rows is 4, so avg row_numbers 2 and 3

```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Retrieving the last record in each group](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#))

There is a table messages that contains data as shown below:

Id	Name	Other_Columns
1	A	A_data_1
2	A	A_data_2
3	A	A_data_3
4	B	B_data_1
5	B	B_data_2
6	C	C_data_1

If I run a query `select * from messages group by name`, I will get the result as:

```

1  A      A_data_1
4  B      B_data_1
6  C      C_data_1

```

What query will return the following result?

```

3  A      A_data_3
5  B      B_data_2
6  C      C_data_1

```

That is, the last record in each group should be returned.

At present, this is the query that I use:

```
select * from (select * from messages ORDER BY id DESC) AS x GROUP BY name
```

But this looks highly inefficient. Any other ways to achieve the same result?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#))

User: [vijay-dev](#)

[Answer](#) by [bill-karwin](#)

I write the solution this way:

```
SELECT m1.*
```

```
FROM messages m1 LEFT JOIN messages m2
  ON (m1.name = m2.name AND m1.id < m2.id)
 WHERE m2.id IS NULL;
```

Regarding performance, one solution or the other can be better, depending on the nature of your data. So you should test both queries and use the one that is better at performance given your database.

For example, I have a copy of the StackOverflow August data dump. I'll use that for benchmarking. There are 1,114,357 rows in the Posts table. This is running on MySQL 5.0.75 on my Macbook Pro 2.40GHz.

I'll write a query to find the most recent post for a given user ID (mine).

First using the technique [shown](#) by @Eric with the GROUP BY in a subquery:

```
SELECT p1.postid
  FROM Posts p1
 INNER JOIN (SELECT pi.owneruserid, MAX(pi.postid) AS maxpostid
              FROM Posts pi GROUP BY pi.owneruserid) p2
            ON (p1.postid = p2.maxpostid)
 WHERE p1.owneruserid = 20860;

1 row in set (1 min 17.89 sec)
```

Even the EXPLAIN analysis takes over 16 seconds:

id	select_type	table	type	possible_keys	key	key_len	ref
1	PRIMARY	<derived2>	ALL	NULL	NULL	NULL	NULL
1	PRIMARY	p1	eq_ref	PRIMARY,PostId,OwnerUserId	PRIMARY	8	p2.ma
2	DERIVED	pi	index	NULL	OwnerUserId	8	NULL

3 rows in set (16.09 sec)

Now produce the same query result using my technique with LEFT JOIN:

```
SELECT p1.postid
  FROM Posts p1 LEFT JOIN posts p2
    ON (p1.owneruserid = p2.owneruserid AND p1.postid < p2.postid)
 WHERE p2.postid IS NULL AND p1.owneruserid = 20860;

1 row in set (0.28 sec)
```

The EXPLAIN analysis shows that both tables are able to use their indexes:

id	select_type	table	type	possible_keys	key	key_len	ref	rows
1	SIMPLE	p1	ref	OwnerUserId	OwnerUserId	8	const	1384
1	SIMPLE	p2	ref	PRIMARY,PostId,OwnerUserId	OwnerUserId	8	const	1384

2 rows in set (0.00 sec)

Here's the DDL for my Posts table:

[Skip code block](#)

```
CREATE TABLE `posts` (
  `PostId` bigint(20) unsigned NOT NULL auto_increment,
  `PostTypeId` bigint(20) unsigned NOT NULL,
  `AcceptedAnswerId` bigint(20) unsigned default NULL,
  `ParentId` bigint(20) unsigned default NULL,
  `CreationDate` datetime NOT NULL,
  `Score` int(11) NOT NULL default '0',
  `ViewCount` int(11) NOT NULL default '0',
  `Body` text NOT NULL,
```

```

`OwnerUserId` bigint(20) unsigned NOT NULL,
`OwnerDisplayName` varchar(40) default NULL,
`LastEditorUserId` bigint(20) unsigned default NULL,
`LastEditDate` datetime default NULL,
`LastActivityDate` datetime default NULL,
`Title` varchar(250) NOT NULL default '',
`Tags` varchar(150) NOT NULL default '',
`AnswerCount` int(11) NOT NULL default '0',
`CommentCount` int(11) NOT NULL default '0',
`FavoriteCount` int(11) NOT NULL default '0',
`ClosedDate` datetime default NULL,
PRIMARY KEY (`PostId`),
UNIQUE KEY `PostId`(`PostId`),
KEY `PostTypeId`(`PostTypeId`),
KEY `AcceptedAnswerId`(`AcceptedAnswerId`),
KEY `OwnerUserId`(`OwnerUserId`),
KEY `LastEditorUserId`(`LastEditorUserId`),
KEY `ParentId`(`ParentId`),
CONSTRAINT `posts_ibfk_1` FOREIGN KEY (`PostTypeId`) REFERENCES `posttypes`(`PostTypeId`)
) ENGINE=InnoDB;

```

[Answer](#) by [newtover](#)

Bill Karwin's solution above works fine when item count within groups is rather small, but the performance of the query becomes bad when the groups are rather large, since the solution requires about $n^2/2 + n/2$ of only IS NULL comparisons.

I made my tests on a InnoDB table of 18684446 rows with 1182 groups. The table contains testresults for functional tests and has the (test_id, request_id) as the primary key. Thus, test_id is a group and I was searching for the last request_id for each test_id.

Bill's solution has already been running for several hours on my dell e4310 and I do not know when it is going to finish even though it operates on a coverage index (hence using index in EXPLAIN).

I have a couple of other solutions that are based on the same ideas:

- if the underlying index is BTREE index (which is usually the case), the largest (group_id, item_value) pair is the last value within each group_id, that is the first for each group_id if we walk through the index in descending order;
- if we read the values which are covered by an index, the values are read in the order of the index;
- each index implicitly contains primary key columns appended to that (that is the primary key is in the coverage index). In solutions below I operate directly on the primary key, in your case, you will just need to add primary key columns in the result.
- in many cases it is much cheaper to collect the required row ids in the required order in a subquery and join the result of the subquery on the id. Since for each row in the subquery result MySQL will need a single fetch based on primary key, the subquery will be put first in the join and the rows will be output in the order of the ids in the subquery (if we omit explicit ORDER BY for the join)

[3 ways MySQL uses indexes](#) is a great article to understand some details.

Solution 1

This one is incredibly fast, it takes about 0,8 secs on my 18M+ rows:

```

SELECT test_id, MAX(request_id), request_id
FROM testresults

```

```
GROUP BY test_id DESC;
```

If you want to change the order to ASC, put it in a subquery, return the ids only and use that as the subquery to join to the rest of the columns:

```
SELECT test_id, request_id
FROM (
    SELECT test_id, MAX(request_id), request_id
    FROM testresults
    GROUP BY test_id DESC) as ids
ORDER BY test_id;
```

This one takes about 1,2 secs on my data.

Solution 2

Here is another solution that takes about 19 seconds for my table:

```
SELECT test_id, request_id
FROM testresults, (SELECT @group:=NULL) as init
WHERE IF(IFNULL(@group, -1)=@group:=test_id, 0, 1)
ORDER BY test_id DESC, request_id DESC
```

It returns tests in descending order as well. It is much slower since it does full index scan but it is here to give you idea how to output N max rows for each group.

The disadvantage of the query is that its result cannot be cached by the query cache.

[Answer](#)  by [eric](#) 

Use your [subquery](#)  to return the correct grouping, because you're halfway there.

Try this:

```
select
  a./*
from
  messages a
  inner join
    (select name, max(id) as maxid from messages group by name) as b on
      a.id = b.maxid
```

If it's not id you want the max of:

```
select
  a./*
from
  messages a
  inner join
    (select name, max(other_col) as other_col
     from messages group by name) as b on
      a.name = b.name
      and a.other_col = b.other_col
```

This way, you avoid correlated subqueries and/or ordering in your subqueries, which tend to be very slow/inefficient.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#))

Q: How to ‘insert if not exists’ in MySQL? 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

I started by googling, and found this [article](#) which talks about mutex tables.

I have a table with ~14 million records. If I want to add more data in the same format, is there a way to ensure the record I want to insert does not already exist without using a pair of queries (ie, one query to check and one to insert is the result set is empty)?

Does a unique constraint on a field guarantee the insert will fail if it's already there?

It seems that with *merely* a constraint, when I issue the insert via php, the script croaks.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

User: [warren](#)

[Answer](#) by [knittl](#)

use INSERT IGNORE INTO table

see <http://bogdan.org.ua/2007/10/18/mysql-insert-if-not-exists-syntax.html>

there's also INSERT ... ON DUPLICATE KEY UPDATE syntax, you can find explanations on dev.mysql.com

Post from bogdan.org.ua according to [Google's webcache](#):

18th October 2007

To start: as of the latest MySQL, syntax presented in the title is not possible. But there are several very easy ways to accomplish what is expected using existing functionality.

There are 3 possible solutions: using INSERT IGNORE, REPLACE, or INSERT ... ON DUPLICATE KEY UPDATE.

Imagine we have a table:

```
CREATE TABLE `transcripts` (
`ensembl_transcript_id` varchar(20) NOT NULL,
`transcript_chrom_start` int(10) unsigned NOT NULL,
`transcript_chrom_end` int(10) unsigned NOT NULL,
PRIMARY KEY (`ensembl_transcript_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Now imagine that we have an automatic pipeline importing transcripts meta-data from Ensembl, and that due to various reasons the pipeline might be broken at any step of execution. Thus, we need to ensure two things: 1) repeated executions of the pipeline will not destroy our database, and 2) repeated executions will not die due to 'duplicate primary key' errors.

Method 1: using REPLACE

It's very simple:

```
REPLACE INTO `transcripts`
SET `ensembl_transcript_id` = 'ENSORGTO00000000001',
`transcript_chrom_start` = 12345,
`transcript_chrom_end` = 12678;
```

If the record exists, it will be overwritten; if it does not yet exist, it will be created.

However, using this method isn't efficient for our case: we do not need to overwrite existing records, it's fine just to skip them.

Method 2: using INSERT IGNORE Also very simple:

```
INSERT IGNORE INTO `transcripts`  
SET `ensembl_transcript_id` = 'ENSORGT000000000001',  
`transcript_chrom_start` = 12345,  
`transcript_chrom_end` = 12678;
```

Here, if the 'ensembl_transcript_id' is already present in the database, it will be silently skipped (ignored). (To be more precise, here's a quote from MySQL reference manual: "If you use the IGNORE keyword, errors that occur while executing the INSERT statement are treated as warnings instead. For example, without IGNORE, a row that duplicates an existing UNIQUE index or PRIMARY KEY value in the table causes a duplicate-key error and the statement is aborted.". If the record doesn't yet exist, it will be created.

This second method has several potential weaknesses, including non-abortion of the query in case any other problem occurs (see the manual). Thus it should be used if previously tested without the IGNORE keyword.

There is one more option: to use INSERT ... ON DUPLICATE KEY UPDATE syntax, and in the UPDATE part just do nothing do some meaningless (empty) operation, like calculating 0+0 (Geoffray suggests doing the id=id assignment for the MySQL optimization engine to ignore this operation). Advantage of this method is that it only ignores duplicate key events, and still aborts on other errors.

As a final notice: this post was inspired by Xaprb. I'd also advise to consult his other post on writing flexible SQL queries.

Answer  by [server](#) 

```
INSERT INTO `table` (value1, value2)  
SELECT 'stuff for value1', 'stuff for value2' FROM `table`  
WHERE NOT EXISTS (SELECT * FROM `table`  
    WHERE value1='stuff for value1' AND value2='stuff for value2')  
LIMIT 1
```

Alternatively, the outer SELECT statement can refer to DUAL in order to handle the case where the table is initially empty:

```
INSERT INTO `table` (value1, value2)  
SELECT 'stuff for value1', 'stuff for value2' FROM DUAL  
WHERE NOT EXISTS (SELECT * FROM `table`  
    WHERE value1='stuff for value1' AND value2='stuff for value2')  
LIMIT 1
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

Q: Fulltext Search with InnoDB 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I'm developing a high-volume web application, where part of it is a MySQL database of

discussion posts that will need to grow to 20M+ rows, smoothly.

I was originally planning on using MyISAM for the tables (for the built-in [fulltext search capabilities](#)), but the thought of the *entire table* being locked due to a single write operation makes me shutter. Row-level locks make so much more sense (not to mention InnoDB's other speed advantages when dealing with huge tables). So, for this reason, I'm pretty determined to use InnoDB.

The problem is... InnoDB doesn't have built-in fulltext search capabilities.

Should I go with a third-party search system? Like [Lucene\(c++\)](#) / [Sphinx](#)? Do any of you database ninjas have any suggestions/guidance? LinkedIn's [zoie](#) (based off Lucene) looks like the best option at the moment... having been built around realtime capabilities (which is pretty critical for my application.) I'm a little hesitant to commit yet without some insight...

(FYI: going to be on EC2 with high-memory rigs, using PHP to serve the frontend)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [brianreavis](#)

[Answer](#) by [jeremy-smyth](#)

Along with the general phasing out of MyISAM, [InnoDB full-text search \(FTS\) is finally available in MySQL 5.6.4 release.](#)

From <http://dev.mysql.com/doc/refman/5.6/en/innodb-table-and-index.html#innodb-fulltext-index>:

These indexes are physically represented as entire InnoDB tables, which are acted upon by SQL keywords such as the FULLTEXT clause of the CREATE INDEX statement, the MATCH() ... AGAINST syntax in a SELECT statement, and the OPTIMIZE TABLE statement.

While other engines have lots of different features, this one is InnoDB, so it's native (which means there's an upgrade path), and that makes it a worthwhile option.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL “NOT IN” query](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I wanted to run a simple query to throw up all the rows of Table1 where a principal column value is not present in a column in another table (Table2).

I tried using:

```
SELECT * FROM Table1 WHERE Table1.principal NOT IN Table2.principal
```

This is instead throwing a syntax error. Google search led me to forums where people

were saying that MySQL does not support NOT IN and something extremely complex needs to be used. Is this true? Or am I making a horrendous mistake?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [kshitij-saxena—kj-](#) 

[Answer](#)  by [julien-lebosquain](#) 

To use IN, you must have a set, use this syntax instead:

```
SELECT * FROM Table1 WHERE Table1.principal NOT IN (SELECT principal FROM table2)
```

[Answer](#)  by [lukáš-lalinský](#) 

The subquery option has already been answered, but note that in many cases a LEFT JOIN can be a faster way to do this:

```
SELECT table1.*  
FROM table1 LEFT JOIN table2 ON table2.principal=table1.principal  
WHERE table2.principal IS NULL
```

If you want to check multiple tables to make sure it's not present in any of the tables (like in SRKR's comment), you can use this:

```
SELECT table1.*  
FROM table1  
LEFT JOIN table2 ON table2.name=table1.name  
LEFT JOIN table3 ON table3.name=table1.name  
WHERE table2.name IS NULL AND table3.name IS NULL
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Get table column names in mysql?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

Is there a way to grab the columns name of a table in mysql? using php

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

User: [an-employee](#)

[Answer](#) by [greg](#)

You can use [DESCRIBE](#):

```
DESCRIBE my_table;
```

Or in newer versions you can use [INFORMATION_SCHEMA](#):

```
SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_SCHEMA = 'my_database' AND TABLE_NAME
```

Or you can use [SHOW COLUMNS](#):

```
SHOW COLUMNS FROM my_table;
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

[Q: Best way to test if a row exists in a MySQL table](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

I'm trying to find out if a row exists in a table. Using MySQL, is it better to do a query like this:

```
SELECT COUNT(*) AS total FROM table1 WHERE...
```

and check to see if the total is non-zero or is it better to do a query like this:

```
SELECT * FROM table1 WHERE... LIMIT 1
```

and check to see if any rows were returned?

In both queries, the WHERE clause uses an index.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

User: [bernard-chen](#)

[Answer](#) by [chris-thompson](#)

You could also try using

```
SELECT EXISTS(SELECT * FROM table1 WHERE...)
```

per [the documentation](#)

Per a comment below:

```
SELECT EXISTS(SELECT 1 FROM table1 WHERE...)
```

[Answer](#) by [laurent-w.](#)

I have made some researches on this subject recently. The way to implement it has to be different if the field is a TEXT field, a non unique field.

I have made some tests with a TEXT field. Considering the fact that we have a table with 1M entries. 37 entries are equal to 'something':

- `SELECT * FROM test WHERE texte LIKE '%something%' LIMIT 1` with `mysql_num_rows()` : 0.039061069488525s. (**FASTER**)
- `SELECT count(*) as count FROM test WHERE text LIKE '%something%':` 16.028197050095s.
- `SELECT EXISTS(SELECT 1 FROM test WHERE text LIKE '%something%')` : 0.87045907974243s.
- `SELECT EXISTS(SELECT 1 FROM test WHERE text LIKE '%something%' LIMIT 1)` : 0.044898986816406s.

But now, with a BIGINT PK field, only one entry is equal to '321321':

- `SELECT * FROM test2 WHERE id ='321321' LIMIT 1` with `mysql_num_rows()` : 0.0089840888977051s.
- `SELECT count(*) as count FROM test2 WHERE id ='321321' :` 0.00033879280090332s.
- `SELECT EXISTS(SELECT 1 FROM test2 WHERE id ='321321')` : 0.00023889541625977s.
- `SELECT EXISTS(SELECT 1 FROM test2 WHERE id ='321321' LIMIT 1)` : 0.00020313262939453s. (**FASTER**)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL error: key specification without a key length](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have a table with a primary key that is a varchar(255). Some cases have arisen where 255 characters isn't enough. I tried changing the field to a text, but I get the following error:

```
BLOB/TEXT column 'message_id' used in key specification without a key length
```

how can I fix this?

edit: I should also point out this table has a composite primary key with multiple columns.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [gsto](#)

[Answer](#) by [omg-ponies](#)

The error happens because MySQL can index only the first N chars of a BLOB or TEXT column. So The error mainly happen when there is a field/column type of TEXT or BLOB or those belongs to TEXT or BLOB types such as TINYBLOB, MEDIUMBLOB, LONGBLOB, TINYTEXT, MEDIUMTEXT, and LONGTEXT that you try to make as primary key or index. With full BLOB or TEXT without the length value, MySQL is unable to guarantee the uniqueness of the column as it's of variable and dynamic size. So, when using BLOB or TEXT types as index, the value of N must be supplied so that MySQL can determine the key length. However, MySQL doesn't support a key length limit on TEXT or BLOB. TEXT(88) simply won't work.

The error will also pop up when you try to convert a table column from non-TEXT and non-BLOB type such as VARCHAR and ENUM into TEXT or BLOB type, with the column already been defined as unique constraints or index. The Alter Table SQL command will fail.

The solution to the problem is to remove the TEXT or BLOB column from the index or unique constraint, or set another field as primary key. If you can't do that, and wanting to place a limit on the TEXT or BLOB column, try to use VARCHAR type and place a limit of length on it. By default, VARCHAR is limited to a maximum of 255 characters and its limit must be specified implicitly within a bracket right after its declaration, i.e VARCHAR(200) will limit it to 200 characters long only.

Sometimes, even though you don't use TEXT or BLOB related type in your table, the Error 1170 may also appear. It happens in situation such as when you specify VARCHAR column as primary key, but wrongly set its length or characters size. VARCHAR can only accept up to 256 characters, so anything such as VARCHAR(512) will force MySQL to auto-convert the VARCHAR(512) to a SMALLTEXT datatype, which subsequently fail with error 1170 on key length if the column is used as primary key or unique or non-unique index. To solve this problem, specify a figure less than 256 as the size for VARCHAR field.

Reference: [MySQL Error 1170 \(42000\): BLOB/TEXT Column Used in Key Specification Without a Key Length](#) 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)) 

[Q: What's the difference between VARCHAR and CHAR?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

What's the difference between VARCHAR and CHAR in MySQL?

I am trying to store MD5 hashes.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [steven](#) 

[Answer](#)  by [anon.](#) 

VARCHAR is variable-length.

CHAR is fixed length.

If your content is a fixed size, you'll get better performance with CHAR.

[Answer](#) by [rahul-bhadana](#)

CHAR

1. Used to store character string value of **fixed length**.
2. The maximum no. of characters the data type can hold is **255 characters**.
3. It's **50% faster** than VARCHAR.
4. Uses **static memory allocation**.

VARCHAR

1. Used to store **variable length** alphanumeric data.
 2. The maximum this data type can hold is up to
 - Pre-MySQL 5.0.3: **255 characters**.
 - In MySQL 5.0.3+: **65,535 characters** shared for the row.
 3. It's **slower** than CHAR.
 4. Uses **dynamic memory allocation**.
-

[Answer](#) by [p-sharma](#)

CHAR Vs VARCHAR

CHAR is used for Fixed Length Size Variable

VARCHAR is used for Variable Length Size Variable.

E.g.

```
Create table temp
(City CHAR(10),
Street VARCHAR(10));

Insert into temp
values('Pune', 'Oxford');

select length(city), length(street) from temp;
```

Output will be

length(City)	Length(street)
10	6

Conclusion: To use storage space efficiently must use VARCHAR Instead CHAR if variable length is variable

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: ROW_NUMBER\(\) in MySQL](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Is there a nice way in MySQL to replicate the SQL Server function ROW_NUMBER()?

For example:

```
SELECT
    col1, col2,
    ROW_NUMBER() OVER (PARTITION BY col1, col2 ORDER BY col3 DESC) AS intRow
FROM Table1
```

Then I could, for example, add a condition to limit intRow to 1 to get a single row with the highest col3 for each (col1, col2) pair.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [paul](#)

[Answer](#) by [omg-ponies](#)

There is no ranking functionality in MySQL. The closest you can get is to use a variable:

```
SELECT t.*,
       @rownum := @rownum + 1 AS rank
  FROM YOUR_TABLE t,
       (SELECT @rownum := 0) r
```

so how would that work in my case? I'd need two variables, one for each of col1 and col2? Col2 would need resetting somehow when col1 changed..?

Yes. If it were Oracle, you could use the LEAD function to peak at the next value.

Thankfully, Quassnoi covers [the logic for what you need to implement in MySQL](#).

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: generate days from date range](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

I would like to run a query like

```
select... as days where `date` is between '2010-01-20' and '2010-01-24'
```

And return data like:

```
days
-----
2010-01-20
2010-01-21
2010-01-22
2010-01-23
2010-01-24
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

User: [pentium10](#)

[Answer](#) by [redfilter](#)

This solution uses **no loops, procedures, or temp tables**. The subquery generates dates for the last thousand days, and could be extended to go as far back or forward as you wish.

```
select a.Date
from (
    select curdate() - INTERVAL (a.a + (10 * b.a) + (100 * c.a)) DAY as Date
    from (select 0 as a union all select 1 union all select 2 union all select 3 union all select 4 u
          cross join (select 0 as a union all select 1 union all select 2 union all select 3 union all sele
          cross join (select 0 as a union all select 1 union all select 2 union all select 3 union all sele
) a
where a.Date between '2010-01-20' and '2010-01-24'
```

Output:

```
Date
-----
2010-01-24
2010-01-23
2010-01-22
2010-01-21
2010-01-20
```

Notes on Performance

Testing it out [here](#), the performance is surprisingly good: **the above query takes 0.0009 sec.**

If we extend the subquery to generate approx. 100,000 numbers (and thus about 274 years worth of dates), it runs in 0.0458 sec.

Incidentally, this is a very portable technique that works with most databases with minor adjustments.

[SQL Fiddle example returning 1,000 days](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

[Q: latitude/longitude find nearest latitude/longitude - complex sql or complex calculation!](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

this is little hard one. i have latitude and longitude and i want to pull the record from the database, which has nearest latitude and longitude by the distance, if that distance gets longer then specified one, then don't retrieve it.

[Skip code block](#)

```
Table structure:
id
latitude
longitude
place name
city
country
state
zip
```

sealevel

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [basit](#) 

[Answer](#)  by [igor-zevaka](#) 

What you need is to translate the distance into degrees of longitude and latitude, filter based on those to bound the entries that are roughly in the bounding box, then do a more precise distance filter. Here is great paper that explains how to do all this:

<http://www.scribd.com/doc/2569355/Geo-Distance-Search-with-MySQL> 

[Answer](#)  by [kaletha](#) 

```
SELECT latitude, longitude, SQRT(
    POW(69.1 * (latitude - [startlat]), 2) +
    POW(69.1 * ([startlng] - longitude) * COS(latitude / 57.3), 2)) AS distance
FROM TableName HAVING distance < 25 ORDER BY distance;
```

where **[starlat]** and **[startlng]** is the position where to start measuring the distance.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL - Get row number on select](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Can I run a select statement and get the row number if the items are sorted?

I have a table like this:

```
mysql> describe orders;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| orderID | bigint(20) unsigned | NO | PRI | NULL | auto_increment |
| itemID | bigint(20) unsigned | NO | | NULL | |
+-----+-----+-----+-----+-----+
```

I can then run this query to get the number of orders by ID:

```
SELECT itemID, COUNT(*) as ordercount
FROM orders
GROUP BY itemID ORDER BY ordercount DESC;
```

This gives me a count of each `itemID` in the table like this:

```
+-----+-----+
| itemID | ordercount |
+-----+-----+
| 388 | 3 |
| 234 | 2 |
| 3432 | 1 |
| 693 | 1 |
| 3459 | 1 |
+-----+-----+
```

I want to get the row number as well, so I could tell that `itemID=388` is the first row, `234` is second, etc (essentially the ranking of the orders, not just a raw count). I know I can do

this in Java when I get the result set back, but I was wondering if there was a way to handle it purely in SQL.

Update

Setting the rank adds it to the result set, but not properly ordered:

[Skip code block](#)

```
mysql> SET @rank=0;
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT @rank:=$rank+1 AS rank, itemID, COUNT(*) as ordercount
   -> FROM orders
   -> GROUP BY itemID ORDER BY rank DESC;
+-----+-----+
| rank | itemID | ordercount |
+-----+-----+
|    5 |    3459 |         1 |
|    4 |     234 |         2 |
|    3 |     693 |         1 |
|    2 |    3432 |         1 |
|    1 |     388 |         3 |
+-----+-----+
5 rows in set (0.00 sec)
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [george](#) 

[Answer](#)  by [mike-cialowicz](#) 

Take a look at [this](#) .

Change your query to:

```
SET @rank=0;
SELECT @rank:=$rank+1 AS rank, itemID, COUNT(*) as ordercount
   FROM orders
   GROUP BY itemID
   ORDER BY ordercount DESC;
```

[Answer](#)  by [swamibebop](#) 

```
SELECT @rn:=$rn+1 AS rank, itemID, ordercount
FROM (
  SELECT itemID, COUNT(*) AS ordercount
  FROM orders
  GROUP BY itemID
  ORDER BY ordercount DESC
) t1, (SELECT @rn:=0) t2;
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL query String contains](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I've been trying to figure out how I can make a query with MySQL that checks if the value (string \$haystack) in a certain column contains certain data (string \$needle), like this:

```
mysql_query("
```

```
SELECT *
FROM `table`
WHERE `column`.contains('{$needle}')
");
```

In PHP, the function is called `substr($haystack, $needle)`, so maybe:

```
WHERE substr(`column`, '{$needle}')=1
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [arik](#) 

[Answer](#)  by [wolph](#) 

Quite simple actually:

```
mysql_query("
SELECT *
FROM `table`
WHERE `column` LIKE '%{$needle}%'"
");
```

The % is a wildcard for any character. Do note that this can get slow on very large datasets so if your database grows you'll need to use fulltext indices.

[Answer](#)  by [omg-ponies](#) 

Use:

```
SELECT *
FROM `table`
WHERE INSTR(`column`, '{$needle}') > 0
```

Reference:

- [INSTR](#) 
-

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL ON DUPLICATE KEY UPDATE for multiple rows insert in single query](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [duplicates](#) ([Prev Q](#)) ([Next Q](#))

I have a sql query where I want to insert multiple rows in single query. so I used something like:

```
$sql = "INSERT INTO beautiful (name, age)
VALUES
('Helen', 24),
('Katrina', 21),
('Samia', 22),
('Hui Ling', 25),
('Yumie', 29);

mysql_query( $sql, $conn );
```

The problem is when I execute this query, I want to check whether a UNIQUE key (which

is not the PRIMARY KEY), for eg. ‘name’ in above case, should be checked and if such a ‘name’ already exists, the corresponding whole row should be updated otherwise inserted.

For instance, in below eg., if ‘Katrina’ is already present in database, the whole row, irrespective of number of fields, should be updated. Again if ‘Samia’ is not present, the row should be inserted.

I thought of using:

```
INSERT INTO beautiful (name, age)
VALUES
('Helen', 24),
('Katrina', 21),
('Samia', 22),
('Hui Ling', 25),
('Yumie', 29) ON DUPLICATE KEY UPDATE
```

Here is the trap. I got stuck and confused how to proceed. I have multiple rows to insert/update at a time. Please give me the direction. Thanks.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [duplicates](#) ([Prev Q](#)) ([Next Q](#))

User: [prashant](#) 

[Answer](#)  by [peter-lang](#) 

Use keyword VALUES to refer to new values (see [documentation](#) .

[Skip code block](#)

```
INSERT INTO beautiful (name, age)
VALUES
('Helen', 24),
('Katrina', 21),
('Samia', 22),
('Hui Ling', 25),
('Yumie', 29)
ON DUPLICATE KEY UPDATE
age = VALUES(age),
...
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [duplicates](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I get around MySQL Errcode 13 with SELECT INTO OUTFILE?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I am trying to dump the contents of a table to a csv file using a MySQL SELECT INTO OUTFILE statement. If I do:

```
SELECT column1, column2
INTO OUTFILE 'outfile.csv'
FIELDS TERMINATED BY ','
FROM table_name;
```

outfile.csv will be created on the server in the same directory this database’s files are stored in.

However, when I change my query to:

```
SELECT column1, column2
INTO OUTFILE '/data/outfile.csv'
FIELDS TERMINATED BY ','
FROM table_name;
```

I get:

```
ERROR 1 (HY000): Can't create/write to file '/data/outfile.csv' (Errcode: 13)
```

Errcode 13 is a permissions error, but I get it even if I change ownership of /data to mysql:mysql and give it 777 permissions. MySQL is running as user “mysql”.

Strangely I can create the file in /tmp, just not in any other directory I’ve tried, even with permissions set such that user mysql should be able to write to the directory.

This is MySQL 5.0.75 running on Ubuntu.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [ryan-olson](#) 

[Answer](#)  by [vin-g](#) 

Which particular version of Ubuntu is this and is this Ubuntu Server Edition?

Recent Ubuntu Server Editions (such as 10.04) ship with AppArmor and MySQL’s profile might be in enforcing mode by default. You can check this by executing `sudo aa-status` like so:

[Skip code block](#)

```
# sudo aa-status
5 profiles are loaded.
5 profiles are in enforce mode.
  /usr/lib/connman/scripts/dhclient-script
  /sbin/dhclient3
  /usr/sbin/tcpdump
  /usr/lib/NetworkManager/nm-dhcp-client.action
  /usr/sbin/mysqld
0 profiles are in complain mode.
1 processes have profiles defined.
1 processes are in enforce mode :
  /usr/sbin/mysqld (1089)
0 processes are in complain mode.
```

If mysqld is included in enforce mode, then it is the one probably denying the write. Entries would also be written in /var/log/messages when AppArmor blocks the writes/accesses. What you can do is edit `/etc/apparmor.d/usr.sbin.mysqld` and add `/data/` and `/data/*` near the bottom like so:

```
...
/usr/sbin/mysqld {
  ...
  /var/log/mysql/ r,
  /var/log/mysql/* rw,
  /var/run/mysqld/mysqld.pid w,
  /var/run/mysqld/mysqld.sock w,
  /data/ r,
  /data/* rw,
}
```

And then make AppArmor reload the profiles.

```
# sudo /etc/init.d/apparmor reload
```

WARNING: the change above will allow MySQL to read and write to the /data directory.
We hope you've already considered the security implications of this.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Q: WHERE vs HAVING

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Why do you need to place columns you create yourself (for example `select 1 as "number"`) after HAVING and not WHERE in MySQL?

And are there any downsides instead of doing WHERE 1 (writing the whole definition instead of a column name)?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [baloo](#)

[Answer](#) by [quassnoi](#)

Why is it that you need to place columns you create yourself (for example “`select 1 as number`”) after HAVING and not WHERE in MySQL?

WHERE is applied before GROUP BY, HAVING is applied after (and can filter on aggregates).

In general, you can reference aliases in neither of these clauses, but MySQL allows referencing SELECT level aliases in GROUP BY, ORDER BY and HAVING.

And are there any downsides instead of doing “WHERE 1” (writing the whole definition instead of a column name)

If your calculated expression does not contain any aggregates, putting it into the WHERE clause will most probably be more efficient.

[Answer](#) by [fishdrowned](#)

All answers upon didn't hit the key point.

Assume we have a table:

```
CREATE TABLE `table` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `value` int(10) unsigned NOT NULL,
  PRIMARY KEY (`id`),
  KEY `value` (`value`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

And have 10 rows with both id and value from 1 to 10:

```
INSERT INTO `table`(`id`, `value`) VALUES (1, 1),(2, 2),(3, 3),(4, 4),(5, 5),(6, 6),(7, 7),(8, 8),(9,
```

Try the following 2 queries:

```
SELECT `value` v FROM `table` WHERE `value`>5;--Get 5 rows
SELECT `value` v FROM `table` HAVING `value`>5;--Get 5 rows
```

You will get exactly the same results, you can see the HAVING clause can work without GROUP BY clause.

Here's the difference:

```
SELECT `value` v FROM `table` WHERE `v`>5;
```

Error #1054 - Unknown column 'v' in 'where clause'

```
SELECT `value` v FROM `table` HAVING `v`>5;--Get 5 rows
```

WHERE clause requires a condition to be a column in a table, but HAVING clause can use either column or alias.

This is because WHERE clause filters data before select, but HAVING clause filters data after select.

So put the conditions in WHERE clause will be more efficient if you have many many rows in a table.

Try EXPLAIN to see the key difference:

[Skip code block](#)

```
EXPLAIN SELECT `value` v FROM `table` WHERE `value`>5;
+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | type | possible_keys | key | key_len | ref | rows | Extra
+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE      | table | range | value        | value | 4       | NULL | 5 | Using where; Usi
+-----+-----+-----+-----+-----+-----+-----+
EXPLAIN SELECT `value` v FROM `table` having `value`>5;
+-----+-----+-----+-----+-----+-----+-----+
| id | select_type | table | type | possible_keys | key | key_len | ref | rows | Extra
+-----+-----+-----+-----+-----+-----+-----+
| 1 | SIMPLE      | table | index | NULL         | value | 4       | NULL | 10 | Using index |
+-----+-----+-----+-----+-----+-----+-----+
```

You can see either WHERE or HAVING uses index, but the rows are different.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Q: [mysql_fetch_array\(\)](#)/[mysql_fetch_assoc\(\)](#)/[mysql_fetch_row\(\)](#) expects parameter 1 to be resource or mysqli_result, boolean given 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

I am trying to select data from a MySQL table, but I get one of the following error messages:

mysql_fetch_array() expects parameter 1 to be resource, boolean given

or

mysqli_fetch_array() expects parameter 1 to be mysqli_result, boolean given

This is my code:

```
$username = $_POST['username'];
$password = $_POST['password'];

$result = mysql_query('SELECT * FROM Users WHERE UserName LIKE $username');

while($row = mysql_fetch_array($result)) {
    echo $row['FirstName'];
```

```
}
```

The same applies to code like

```
$result = mysqli_query($mysqli, 'SELECT...');  
// mysqli_fetch_array() expects parameter 1 to be mysqli_result, boolean given  
while( $row=mysqli_fetch_array($result) ) {  
    ...
```

and

```
$result = $mysqli->query($mysqli, 'SELECT...');  
// Call to a member function fetch_assoc() on a non-object  
while( $row=$result->fetch_assoc($result) ) {  
    ...
```

and

```
$result = $pdo->query('SELECT...', PDO::FETCH_ASSOC);  
// Invalid argument supplied for foreach()  
foreach( $result as $row ) {  
    ...
```

and

```
$stmt = $mysqli->prepare('SELECT...');  
// Call to a member function bind_param() on a non-object  
$stmt->bind_param(...)
```

and

```
$stmt = $pdo->prepare('SELECT...');  
// Call to a member function bindParam() on a non-object  
$stmt->bindParam(...)
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

User: [iamjonesy](#) 

[Answer](#)  by [scompt.com](#) 

A query may fail for various reasons in which case both the `mysql_*` and the `mysqli` extension will return `false` from their respective query functions/methods. You need to test for that error condition and handle it accordingly.

[mysql *](#) extension 

NOTE The [mysql functions are deprecated](#)  and will be removed with the upcoming php version 7.

Check `$result` before passing it to `mysql_fetch_array`. You'll find that it's `false` because the query failed. See the [mysql_query](#)  documentation for possible return values and suggestions for how to deal with them.

[Skip code block](#)

```
$username = mysql_real_escape_string($_POST['username']);  
$password = $_POST['password'];  
$result = mysql_query("SELECT * FROM Users WHERE UserName LIKE '$username'");  
  
if($result === FALSE) {  
    die(mysql_error()); // TODO: better error handling  
}  
  
while($row = mysql_fetch_array($result))
```

```
{  
    echo $row['FirstName'];  
}
```

[mysqli extension](#)

procedural style:

[Skip code block](#)

```
$username = mysqli_real_escape_string($mysqli, $_POST['username']);  
$result = mysqli_query($mysqli, "SELECT * FROM Users WHERE UserName LIKE '$username'");  
  
// mysqli_query returns false if something went wrong with the query  
if($result === FALSE) {  
    yourErrorHandler(mysqli_error($mysqli));  
}  
else {  
    // as of php 5.4 mysqli_result implements Traversable, so you can use it with foreach  
    foreach( $result as $row ) {  
        ...  
    }  
}
```

oo-style:

[Skip code block](#)

```
$username = $mysqli->escape_string($_POST['username']);  
$result = $mysqli->query("SELECT * FROM Users WHERE UserName LIKE '$username'");  
  
if($result === FALSE) {  
    yourErrorHandler($mysqli->error); // or $mysqli->error_list  
}  
else {  
    // as of php 5.4 mysqli_result implements Traversable, so you can use it with foreach  
    foreach( $result as $row ) {  
        ...  
    }  
}
```

using a prepared statement:

[Skip code block](#)

```
$stmt = $mysqli->prepare('SELECT * FROM Users WHERE UserName LIKE ?');  
if ( !$stmt ) {  
    yourErrorHandler($mysqli->error); // or $mysqli->error_list  
}  
else if ( !$stmt->bind_param('s', $_POST['username']) ) {  
    yourErrorHandler($stmt->error); // or $stmt->error_list  
}  
else if ( !$stmt->execute() ) {  
    yourErrorHandler($stmt->error); // or $stmt->error_list  
}  
else {  
    $result = $stmt->get_result();  
    // as of php 5.4 mysqli_result implements Traversable, so you can use it with foreach  
    foreach( $result as $row ) {  
        ...  
    }  
}
```

These examples only illustrate *what* should be done (error handling), not how to do it. Production code shouldn't use [or die](#) when outputting HTML, else it will (at the very least) generate invalid HTML. Also, database error messages shouldn't be displayed to non-admin users, as it [discloses too much information](#).

[Answer](#) by [john-conde](#)

This error message is displayed when you have an error in your query which caused it to fail. It will manifest itself when using:

- `mysql_fetch_array/mysqli_fetch_array()`
- `mysql_fetch_assoc()/mysqli_fetch_assoc()`

- `mysql_num_rows()`/`mysqli_num_rows()`

Note: This error does *not* appear if no rows are affected by your query. Only a query with an invalid syntax will generate this error.

Troubleshooting Steps

- Make sure you have your development server configured to display all errors. You can do this by placing this at the top of your files or in your config file: [error_reporting\(-1\);](#). If you have any syntax errors this will point them out to you.
- Use [mysql_error\(\)](#). `mysql_error()` will report any errors MySQL encountered while performing your query.

Sample usage:

```
mysql_connect($host, $username, $password) or die("cannot connect");
mysql_select_db($db_name) or die("cannot select DB");

$sql = "SELECT * FROM table_name";
$result = mysql_query($sql);

if (false === $result) {
    echo mysql_error();
}
```

- Run your query from the MySQL command line or a tool like [phpMyAdmin](#). If you have a syntax error in your query this will tell you what it is.
- Make sure your quotes are correct. A missing quote around the query or a value can cause a query to fail.
- Make sure you are escaping your values. Quotes in your query can cause a query to fail (and also leave you open to SQL injections). Use [mysql_real_escape_string\(\)](#) to escape your input.
- Make sure you are not mixing `mysqli_*` and `mysql_*` functions. They are not the same thing and cannot be used together. (If you're going to choose one or the other stick with `mysqli_*`. See below for why.)

Other tips

`mysql_*` functions should not be used for new code. They are no longer maintained and the community has begun the [deprecation process](#). Instead you should learn about [prepared statements](#) and use either [PDO](#) or [MySQLi](#). If you can't decide, [this article](#) will help to choose. If you care to learn, here is [good PDO tutorial](#).

[Answer](#) by [nik](#)

Error occurred here was due to the use of single quotes (''). You can put your query like this:

```
mysql_query(
    "SELECT * FROM Users
     WHERE UserName
      LIKE '".mysql_real_escape_string ($username)."'
    ");
```

It's using `mysql_real_escape_string` for prevention of SQL injection. Though we should use MySQLi or PDO_MYSQL extension for upgraded version of PHP (PHP 5.5.0 and later), but for older versions `mysql_real_escape_string` will do the trick.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

[Q: Remove duplicate rows in MySQL](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have a table with the following fields:

```
id (Unique)
url (Unique)
title
company
site_id
```

Now, I need to remove rows having same title, company and site_id. One way to do it will be using the following SQL along with a script (PHP):

```
SELECT title, site_id, location, id, count( * )
FROM jobs
GROUP BY site_id, company, title, location
HAVING count( * ) >1
```

After running this query, I can remove duplicates using a server side script. But, I want to know if this can be done only using SQL query.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [chetan](#) 

[Answer](#)  by [chris-henry](#) 

A really easy way to do this is to add a `UNIQUE` index on the 3 columns. When you write the `ALTER` statement, include the `IGNORE` keyword. Like so:

```
ALTER IGNORE TABLE jobs
ADD UNIQUE INDEX idx_name (site_id, title, company);
```

This will drop all the duplicate rows. As an added benefit, future `INSERTs` that are duplicates will error out. As always, you may want to take a backup before running something like this...

[Answer](#)  by [andomar](#) 

MySQL has restrictions about referring to the table you are deleting from. You can work around that with a temporary table, like:

[Skip code block](#)

```
create temporary table tmpTable (id int);

insert tmpTable
(id)
select id
from YourTable yt
where exists
```

```

(
  select  *
  from    YourTable yt2
  where   yt2.title = yt.title
          and yt2.company = yt.company
          and yt2.site_id = yt.site_id
          and yt2.id > yt.id
);
delete
from  YourTable
where ID in (select id from tmpTable);

```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Rank function in MySQL](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I am not an expert in MySQL . I need to find out rank of customers. Here I am adding the corresponding ANSI standard SQL query for my requirement. Please help me to convert it to MySQL .

```

SELECT RANK() OVER (PARTITION BY Gender ORDER BY Age) AS [Partition by Gender],
       FirstName,
       Age,
       Gender
  FROM Person

```

Is there any function to find out rank in MySQL?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [abhis](#) 

[Answer](#)  by [daniel-vassallo](#) 

One option is to use a ranking variable, such as the following:

```

SELECT      first_name,
            age,
            gender,
            @curRank := @curRank + 1 AS rank
  FROM        person p, (SELECT @curRank := 0) r
 ORDER BY    age;

```

The (SELECT @curRank := 0) part allows the variable initialization without requiring a separate SET command.

Test case:

[Skip code block](#)

```

CREATE TABLE person (id int, first_name varchar(20), age int, gender char(1));

INSERT INTO person VALUES (1, 'Bob', 25, 'M');
INSERT INTO person VALUES (2, 'Jane', 20, 'F');
INSERT INTO person VALUES (3, 'Jack', 30, 'M');
INSERT INTO person VALUES (4, 'Bill', 32, 'M');
INSERT INTO person VALUES (5, 'Nick', 22, 'M');
INSERT INTO person VALUES (6, 'Kathy', 18, 'F');
INSERT INTO person VALUES (7, 'Steve', 36, 'M');
INSERT INTO person VALUES (8, 'Anne', 25, 'F');

```

Result:

[Skip code block](#)

first_name	age	gender	rank
Kathy	18	F	1
Jane	20	F	2
Nick	22	M	3
Bob	25	M	4
Anne	25	F	5
Jack	30	M	6
Bill	32	M	7
Steve	36	M	8

8 rows in set (0.02 sec)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: What do column flags mean in MySQL Workbench?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

In MySQL Workbench table editor there are 7 column flags available: PK, NN, UQ, BIN, UN, ZF, AI.

PK obviously stands for Primary Key. What about others?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [ivan](#) 

[Answer](#)  by [vasiliy-kulakov](#) 

PK - Primary Key

NN - Not Null

BIN - Binary (stores data as binary strings. There is no character set so sorting and comparison is based on the numeric values of the bytes in the values.)

UN - Unsigned (non-negative numbers only. so if the range is -500 to 500, instead its 0 - 1000, the range is the same but it starts at 0)

UQ - Create/remove Unique Key

ZF - Zero-Filled (if the length is 5 like INT(5) then every field is filled with 0's to the 5th value. 12 = 00012, 400 = 00400, etc.)

AI - Auto Increment

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to convert all tables from MyISAM into InnoDB?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I know I can issue an alter table individually to change the table storage from MyISAM to InnoDB.

I am wondering if there is a way to quickly change all of them to InnoDB?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [pentium10](#) 

[Answer](#)  by [gajendra-bang](#) 

[Skip code block](#)

```
<?php
    // connect your database here first
    //

    // Actual code starts here

    $sql = "SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES
        WHERE TABLE_SCHEMA = 'your_database_name'
        AND ENGINE = 'MyISAM'";

    $rs = mysql_query($sql);

    while($row = mysql_fetch_array($rs))
    {
        $tbl = $row[0];
        $sql = "ALTER TABLE `{$tbl}` ENGINE=INNODB";
        mysql_query($sql);
    }
?>
```

[Answer](#)  by [will-jones](#) 

Run this SQL statement (in the mysql client, phpMyAdmin, or wherever) to retrieve all the MyISAM tables in your database.

Replace value of the name_of_your_db variable with your database name.

```
SET @DATABASE_NAME = 'name_of_your_db';

SELECT CONCAT('ALTER TABLE `', table_name, '` ENGINE=InnoDB;') AS sql_statements
FROM information_schema.tables AS tb
WHERE table_schema = @DATABASE_NAME
AND `ENGINE` = 'MyISAM'
AND `TABLE_TYPE` = 'BASE TABLE'
ORDER BY table_name DESC;
```

Then, copy the output and run as a new SQL query.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Q: MySQL's now() +1 day 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

I'm using now() in MySQL query.

```
INSERT INTO table SET data = '$data', date = now()
```

But I want to add 1 day to this date (so that date should contain tomorrow).
Is it possible?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

User: [qiao](#) 

[Answer](#)  by [mark-byers](#) 

You can use:

```
NOW() + INTERVAL 1 DAY
```

If you are only interested in the date, not the date and time then you can use CURDATE instead of NOW:

```
CURDATE() + INTERVAL 1 DAY
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to rename a table column in MySQL](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

How to rename a table column in table xyz, the columns are:

```
Manufacurerid, name, status, AI, PK, int
```

I want to rename to manufacturerid

I tried using PHPMyAdmin panel which is not working. It shows an error:

```
MySQL said: Documentation  
#1025 - Error on rename of '.\shopping\#sql-c98_26' to '.\shopping\tblmanufacturer' (errno: 150)
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [bharanikumar](#) 

[Answer](#)  by [matt-diamond](#) 

Lone Ranger is very close... in fact, you also need to specify the datatype of the renamed column. For example:

```
ALTER TABLE xyz CHANGE manufacurerid manufacturerid INT
```

(replacing INT with whatever your column definition is)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL: Can't create table \(errno: 150\)](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I am trying to import a .sql file and its failing on creating tables.

Here's the query that fails:

```
CREATE TABLE `data` (
  `id` int(10) unsigned NOT NULL,
  `name` varchar(100) NOT NULL,
  `value` varchar(15) NOT NULL,
  UNIQUE KEY `id` (`id`, `name`),
  CONSTRAINT `data_ibfk_1` FOREIGN KEY (`id`) REFERENCES `keywords` (`id`) ON DELETE CASCADE ON UPDATE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

I exported the .sql from the same database, I dropped all the tables and now im trying to import it, why is it failing?

MySQL: Can't create table './dbname/data.frm' (errno: 150)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [gtlx](#) 

[Answer](#)  by [omg-ponies](#) 

From the [MySQL - FOREIGN KEY Constraints Documentation](#): 

If you re-create a table that was dropped, it must have a definition that conforms to the foreign key constraints referencing it. It must have the correct column names and types, and it must have indexes on the referenced keys, as stated earlier. ***If these are not satisfied, MySQL returns Error 1005 and refers to Error 150 in the error message, which means that a foreign key constraint was not correctly formed.***

Similarly, if an ALTER TABLE fails due to Error 150, this means that a foreign key definition would be incorrectly formed for the altered table.

[Answer](#)  by [dan-mcgrath](#) 

Error 150 means you have a problem with your foreign key. Possibly the key on the foreign table isn't the exact same type?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Insert into a MySQL table or update if exists](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I want to add a row to a database table, but a row exists with the same unique key I want to update the row.

For example,

```
insert into table (id, name, age) values(1, "A", 19)
```

Let's say the unique key is `id` and in my database there is a row with `id = 1`. In that case, I want to update that row with these values. Normally this gives an error. If I use `INSERT IGNORE` it will ignore the error but still won't update.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [keshan](#) 

[Answer](#)  by [donnie](#) 

Use [INSERT... ON DUPLICATE KEY UPDATE](#) 

QUERY:

```
INSERT INTO table (id, name, age) VALUES(1, "A", 19) ON DUPLICATE KEY UPDATE  
name="A", age=19
```

[Answer](#)  by [martin-schapendonk](#) 

Check out REPLACE

<http://dev.mysql.com/doc/refman/5.0/en/replace.html> 

```
REPLACE into table (id, name, age) values(1, "A", 19)
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: mysql :: insert into table, data from another table?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I was wondering if there is a way to do this purely in sql:

```
q1 = SELECT campaign_id, from_number, received_msg, date_received  
      FROM `received_txts` WHERE `campaign_id` = '8';  
INSERT INTO action_2_members (campaign_id, mobile, vote, vote_date)  
VALUES(q1.campaign_id, q1.from_number, q1.received_msg, q1.date_received);
```

Note: q1 would return about 30k rows.

Is there any way to do what I am attempting above in straight sql? To just pull the data straight from one table (basically a raw data table) and insert into another table (basically a processed data table)?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [hailwood](#) 

[Answer](#)  by [zerkms](#) 

```
INSERT INTO action_2_members (campaign_id, mobile, vote, vote_date)  
SELECT campaign_id, from_number, received_msg, date_received  
      FROM `received_txts`  
     WHERE `campaign_id` = '8'
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL select 10 random rows from 600K rows fast](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#)), [random](#) ([Prev Q](#)) ([Next Q](#))

How can I best write a query that selects 10 rows randomly from a total of 600k?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#)), [random](#) ([Prev Q](#)) ([Next Q](#))

User: [francisc](#)

[Answer](#) by [riedsio](#)

A great post handling several cases, from simple, to gaps, to non-uniform with gaps.

<http://jan.kneschke.de/projects/mysql/order-by-rand/>

For most general case, here is how you do it:

```
SELECT name
  FROM random AS r1 JOIN
       (SELECT CEIL(RAND() *
                    (SELECT MAX(id)
                     FROM random)) AS id)
    AS r2
 WHERE r1.id >= r2.id
 ORDER BY r1.id ASC
 LIMIT 1
```

This supposes that the distribution of ids is equal, and that there can be gaps in the id list.
See the article for more advanced examples

[Answer](#) by [preetam-purbia](#)

```
SELECT column FROM table
ORDER BY RAND()
LIMIT 10
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#)), [random](#) ([Prev Q](#)) ([Next Q](#))

[Q: You can't specify target table for update in FROM clause](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have a simple mysql table:

[Skip code block](#)

```
CREATE TABLE IF NOT EXISTS `pers` (
  `persID` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(35) NOT NULL,
  `gehalt` int(11) NOT NULL,
  `chefID` int(11) DEFAULT NULL,
  PRIMARY KEY (`persID`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=4 ;

INSERT INTO `pers` (`persID`, `name`, `gehalt`, `chefID`) VALUES
(1, 'blb', 1000, 3),
(2, 'as', 1000, 3),
(3, 'chef', 1040, NULL);
```

I tried to run following update, but I get only the error 1093:

[Skip code block](#)

```
UPDATE pers P
SET P.gehalt = P.gehalt * 1.05
WHERE (P.chefID IS NOT NULL
OR gehalt <
(SELECT (
    SELECT MAX(gehalt * 1.05)
    FROM pers MA
    WHERE MA.chefID = MA.chefID)
    AS _pers
))
```

I searched for the error and found from mysql following page

<http://dev.mysql.com/doc/refman/5.1/en/subquery-restrictions.html>, but it doesn't help me.

What shall I do to correct the sql query?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [cshulz](#)

[Answer](#) by [blueraja-danny-pflughoeft](#)

The problem is that MySQL, for whatever inane reason, doesn't allow you to write queries like this:

```
UPDATE myTable
SET myTable.A =
(
    SELECT B
    FROM myTable
    INNER JOIN...
```

That is, if you're doing an UPDATE/INSERT/DELETE on a table, you can't reference that table in an inner query (*you can however reference a field from that outer table...*)

The solution is to replace the instance of `myTable` in the sub-query with `(SELECT * FROM myTable)`, like this

```
UPDATE myTable
SET myTable.A =
(
    SELECT B
    FROM (SELECT * FROM myTable) AS something
    INNER JOIN...
```

This apparently causes the necessary fields to be implicitly copied into a temporary table, so it's allowed.

I found this solution [here](#). A note from that article:

You don't want to just `SELECT * FROM table` in the subquery in real life; I just wanted to keep the examples simple. In reality you should only be selecting the columns you need in that innermost query, and adding a good `WHERE` clause to limit the results, too.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: INSERT ... ON DUPLICATE KEY \(do nothing\)](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have a table with a unique key for two columns:

```
CREATE TABLE `xpo`.`user_permanent_gift` (
`id` INT UNSIGNED NOT NULL AUTO_INCREMENT ,
`fb_user_id` INT UNSIGNED NOT NULL ,
`gift_id` INT UNSIGNED NOT NULL ,
`purchase_timestamp` TIMESTAMP NULL DEFAULT now() ,
PRIMARY KEY (`id`),
UNIQUE INDEX `user_gift_UNIQUE` (`fb_user_id` ASC, `gift_id` ASC) );
```

I want to insert a row into that table, but if the key exists, to do nothing! I don't want an error to be generated because the keys exist.

I know that there is the following syntax:

```
INSERT... ON DUPLICATE KEY UPDATE...
```

but is there something like:

```
INSERT... ON DUPLICATE KEY DO NOTHING
```

?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [ufk](#) 

[Answer](#)  by [ceejayoz](#) 

Yes, there's `INSERT IGNORE`, or you can do `ON DUPLICATE KEY UPDATE id=id`.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Delete all Duplicate Rows except for One in MySQL?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Possible Duplicate:

[Remove duplicate rows in MySQL](#)

How would I delete all duplicate data from a MySQL Table?

For example, with the following data:

```
SELECT * FROM names;
```

[Skip code block](#)

id	name
1	google

```
| 2 | yahoo |
| 3 | msn   |
| 4 | google|
| 5 | google|
| 6 | yahoo |
+---+-----+
```

I would use `SELECT DISTINCT name FROM names`; if it were a `SELECT` query. How would I do this with `DELETE` to only remove duplicates and keep just one record of each?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [highway-of-life](#) 

[Answer](#)  by [mtendotnet](#) 

1) If you want to keep the row with the lowest `id` value:

```
DELETE n1 FROM names n1, names n2 WHERE n1.id > n2.id AND n1.name = n2.name
```

2) If you want to keep the row with the highest `id` value:

```
DELETE n1 FROM names n1, names n2 WHERE n1.id < n2.id AND n1.name = n2.name
```

I used this method in MySQL 5.1

Not sure about other versions.

NB - You need to do this first on a test copy of your table!

When I did it, I found that unless I also included `AND n1.id <> n2.id`, it deleted every row in the table.

[Answer](#)  by [omg-ponies](#) 

If you want to keep the row with the lowest `id` value:

```
DELETE FROM NAMES
WHERE id NOT IN (SELECT *
                  FROM (SELECT MIN(n.id)
                        FROM NAMES n
                       GROUP BY n.name) x)
```

If you want the `id` value that is the highest:

```
DELETE FROM NAMES
WHERE id NOT IN (SELECT *
                  FROM (SELECT MAX(n.id)
                        FROM NAMES n
                       GROUP BY n.name) x)
```

The subquery in a subquery is necessary for MySQL, or you'll get a 1093 error.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Q: MYSQL - ORDER BY & LIMIT 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [sql-order-by](#) ([Next Q](#))

I have a query that looks like this:

```
SELECT article FROM table1 ORDER BY publish_date LIMIT 20
```

How does ORDER BY work? Will it order all records, then get the first 20, or will it get 20 records and order them by the publish_date field?

If it's the last one, you're not guaranteed to really get the most recent 20 articles.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [sql-order-by](#) ([Next Q](#))

User: [alex](#) 

[Answer](#)  by [james](#) 

It will order first, then get the first 20. A database will also process anything in the WHERE clause before ORDER BY.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [sql-order-by](#) ([Next Q](#))

[Q: Full Outer Join in MySQL](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

I want to do a Full Outer Join in MySQL. Is this possible? Is a Full Outer Join supported by MySQL?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

User: [peter](#) 

[Answer](#)  by [pablo-santa-cruz](#) 

You don't have FULL JOINS on MySQL, but you can sure [emulate them](#) .

For a code SAMPLE transcribed from [this SO question](#)  you have:

with two tables t1, t2:

```
SELECT * FROM t1
LEFT JOIN t2 ON t1.id = t2.id
UNION
SELECT * FROM t1
RIGHT JOIN t2 ON t1.id = t2.id
```

[Answer](#)  by [nathan-long](#) 

The answer that [Pablo Santa Cruz](#)  gave is correct; however, in case anybody stumbled on this page and wants more clarification, here is a detailed breakdown.

Example Tables

Suppose we have the following tables:

```
-- t1
id  name
1   Tim
2   Marta
```

```
-t2
id  name
1   Tim
3   Katarina
```

Inner Joins

An inner join, like this:

```
SELECT *
FROM `t1`
INNER JOIN `t2` ON `t1`.`id` = `t2`.`id`;
```

Would get us only records that appear in both tables, like this:

```
1 Tim  1 Tim
```

Inner joins don't have a direction (like left or right) because they are explicitly bidirectional - we require a match on both sides.

Outer Joins

Outer joins, on the other hand, are for finding records that may not have a match in the other table. As such, you have to specify **which side** of the join is allowed to have a missing record.

`LEFT JOIN` and `RIGHT JOIN` are shorthand for `LEFT OUTER JOIN` and `RIGHT OUTER JOIN`; I will use their full names below to reinforce the concept of outer joins vs inner joins.

Left Outer Join

A left outer join, like this:

```
SELECT *
FROM `t1`
LEFT OUTER JOIN `t2` ON `t1`.`id` = `t2`.`id`;
```

...would get us all the records from the left table regardless of whether or not they have a match in the right table, like this:

```
1 Tim  1   Tim
2 Marta NULL NULL
```

Right Outer Join

A right outer join, like this:

```
SELECT *
FROM `t1`
RIGHT OUTER JOIN `t2` ON `t1`.`id` = `t2`.`id`;
```

...would get us all the records from the right table regardless of whether or not they have a match in the left table, like this:

```
1   Tim  1   Tim
NULL NULL 3   Katarina
```

Full Outer Join

A full outer join would give us all records from both tables, whether or not they have a match in the other table, with NULLs on both sides where there is no match. The result would look like this:

```
1    Tim    1    Tim
2    Marta  NULL  NULL
NULL NULL  3    Katarina
```

However, as Pablo Santa Cruz pointed out, MySQL doesn't support this. We can emulate it by doing a UNION of a left join and a right join, like this:

```
SELECT *
FROM `t1`
LEFT OUTER JOIN `t2` ON `t1`.`id` = `t2`.`id`

UNION

SELECT *
FROM `t1`
RIGHT OUTER JOIN `t2` ON `t1`.`id` = `t2`.`id`;
```

You can think of a UNION as meaning "run both of these queries, then stack the results on top of each other"; some of the rows will come from the first query and some from the second.

It should be noted that a UNION in MySQL will eliminate exact duplicates: Tim would appear in both of the queries here, but the result of the UNION only lists him once. My database guru colleague feels that this behavior should not be relied upon. So to be more explicit about it, we could add a WHERE clause to the second query:

[Skip code block](#)

```
SELECT *
FROM `t1`
LEFT OUTER JOIN `t2` ON `t1`.`id` = `t2`.`id`

UNION

SELECT *
FROM `t1`
RIGHT OUTER JOIN `t2` ON `t1`.`id` = `t2`.`id`
WHERE `t1`.`id` IS NULL;
```

On the other hand, if you **wanted** to see duplicates for some reason, you could use UNION ALL.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

Q: Import SQL file into mysql

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

I have a database `nitm`. I haven't created any tables there. But I have an sql file which contains all the necessary data for the database. The file is `nitm.sql` which is in `C:\drive`. This file has size of about 103M. I am using wamp server.

I have used the following syntax in mysql console to import the file.

```
mysql>c:/nitm.sql;
```

But this didn't work.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [coosal](#) 

[Answer](#)  by [d_-b](#) 

From the mysql console:

```
mysql> use DATABASE_NAME;
```

```
mysql> source path/to/file.sql;
```

make sure there is no slash before path ... it took me a while to realize that! lol

[Answer](#)  by [coosal](#) 

Finally, i solved the problem. I placed the `nitm.sql` file in `bin` file of the `mysql` folder and used the following syntax.

```
C:\wamp\bin\mysql\mysql5.0.51b\bin>mysql -u root nitm < nitm.sql
```

And this worked.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL SELECT only not null values](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#))

Is it possible to do a select statement that takes only NOT NULL values.

Right now I am using this:

```
SELECT * FROM table
```

And then I have to filter out the null values with a php loop

is there a way to do:

```
SELECT * (that are NOT NULL) FROM table?
```

Right now when I select * I get val1,val2,val3,null,val4,val5,null,null etc.... but I just want to get the values that are not null in my result. Is this possible without filtering with a loop?

I've been searching for a while now, cant figure it out

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#))

User: [bryan-sammon](#) 

[Answer](#) by [martin-smith](#)

You should use `IS NOT NULL`. (The comparison operators = and <> both give UNKNOWN with `NULL` on either side of the expression.)

```
SELECT *
FROM table
WHERE YourColumn IS NOT NULL;
```

Just for completeness I'll mention that in MySQL you can also negate the [null safe equality operator](#) but this is not standard SQL.

```
SELECT *
FROM table
WHERE NOT (YourColumn <=> NULL);
```

Edited to reflect comments. It sounds like your table may not be in first normal form in which case changing the structure may make your task easier. A couple of other ways of doing it though...

```
SELECT val1 AS val
FROM your_table
WHERE val1 IS NOT NULL
UNION ALL
SELECT val2
FROM your_table
WHERE val2 IS NOT NULL
/*And so on for all your columns*/
```

The disadvantage of the above is that it scans the table multiple times once for each column. That may possibly be avoided by the below but I haven't tested this in MySQL.

[Skip code block](#)

```
SELECT CASE idx
      WHEN 1 THEN val1
      WHEN 2 THEN val2
    END AS val
FROM your_table
  /*CROSS JOIN*/
  JOIN (SELECT 1 AS idx
        UNION ALL
        SELECT 2) t
HAVING val IS NOT NULL /*Can reference alias in Having in MySQL*/
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL keys, MUL vs PRI vs UNI](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

What is the difference between MUL, PRI and UNI in MySQL?

I'm working on a MySQL query, using the command:

```
desc mytable;
```

One of the fields is shown as being a MUL key, others show up as UNI or PRI.

I know that if a key is PRI, only one record per table can be associated with that key. If a key is MUL, does that mean that there could be more than one associated record?

Here's the response of mytable.

Field	Type	Null	Key	Default	Extra
courseid	int(11)	YES	MUL	NULL	
dept	char(3)	YES		NULL	
coursenum	char(4)	YES		NULL	

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [themaestro](#) 

[Answer](#)  by [matt-healy](#) 

It means that the field is (part of) a non-unique index. You can issue

```
show create table <table>;
```

To see more information about the table structure.

[Answer](#)  by [robguinness](#) 

```
DESCRIBE <table>;
```

This is actually a shortcut for:

```
SHOW COLUMNS FROM <table>;
```

In any case, there are three possible values for the “Key” attribute:

1. PRI
2. UNI
3. MUL

The meaning of PRI and UNI are quite clear:

- PRI=> primary key
- UNI=> unique key

The third possibility, MUL, (which you asked about) is basically an index that is neither a primary key nor a unique key. The name comes from “multiple” because multiple occurrences of the same value are allowed. Straight from the [MySQL documentation](#) :

“If Key is MUL, the column is the first column of a nonunique index in which multiple occurrences of a given value are permitted within the column.”

There is also a final caveat:

“If more than one of the Key values applies to a given column of a table, Key displays the one with the highest priority, in the order PRI, UNI, MUL.”

As a general note, the MySQL documentation is quite good. When in doubt, check it out!

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: INSERT with SELECT](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Say I have a query that inserts using a select as such:

```
INSERT INTO courses name, location, gid  
    SELECT (name, location, gid)  
        FROM courses  
    WHERE cid = $cid
```

Is it possible to only select “name,location” for the insert, and set gid to something else in the query?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [kyle](#)

[Answer](#) by [andrew](#)

Yes, absolutely, but check your syntax.

```
INSERT INTO courses (name, location, gid)  
SELECT name, location, 1  
FROM   courses  
WHERE  cid = 2
```

You can put a constant of the same type as gid in its place, not just 1, of course. And, I just made up the cid value.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySQL OPTIMIZE all tables?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

MySQL has an [OPTIMIZE TABLE](#) command which can be used to reclaim unused space in a MySQL install. Is there a way (built-in command or common stored procedure) to run this optimization for every table in the database and/or server install, or is this something you'd have to script up yourself?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [alan-storm](#)

[Answer](#) by [ike-walker](#)

You can use [mysqlcheck](#) to do this at the command line.

One database:

```
mysqlcheck -o <db_schema_name>
```

All databases:

```
mysqlcheck -o --all-databases
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I make SQL case sensitive string comparison on MySQL?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [case-sensitive](#) ([Next Q](#))

I have a function that returns five characters with mixed case. If I do a query on this string it will return the value regardless of case.

How can I make MySQL string queries case sensitive?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [case-sensitive](#) ([Next Q](#))

User: [stevenb](#) 

[Answer](#)  by [drudge](#) 

<http://dev.mysql.com/doc/refman/5.0/en/case-sensitivity.html> 

The default character set and collation are latin1 and latin1_swedish_ci, so nonbinary string comparisons are case insensitive by default. This means that if you search with col_name LIKE 'a%', you get all column values that start with A or a. To make this search case sensitive, make sure that one of the operands has a case sensitive or binary collation. For example, if you are comparing a column and a string that both have the latin1 character set, you can use the COLLATE operator to cause either operand to have the latin1_general_cs or latin1_bin collation:

```
col_name COLLATE latin1_general_cs LIKE 'a%'  
col_name LIKE 'a%' COLLATE latin1_general_cs  
col_name COLLATE latin1_bin LIKE 'a%'  
col_name LIKE 'a%' COLLATE latin1_bin
```

If you want a column always to be treated in case-sensitive fashion, declare it with a case sensitive or binary collation.

[Answer](#)  by [craig-white](#) 

The good news is that if you need to make a case-sensitive query, it is very easy to do:

```
SELECT * FROM `table` WHERE BINARY `column` = 'value'
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [case-sensitive](#) ([Next Q](#))

[Q: What's the difference between INNER JOIN, LEFT JOIN, RIGHT JOIN and FULL JOIN?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

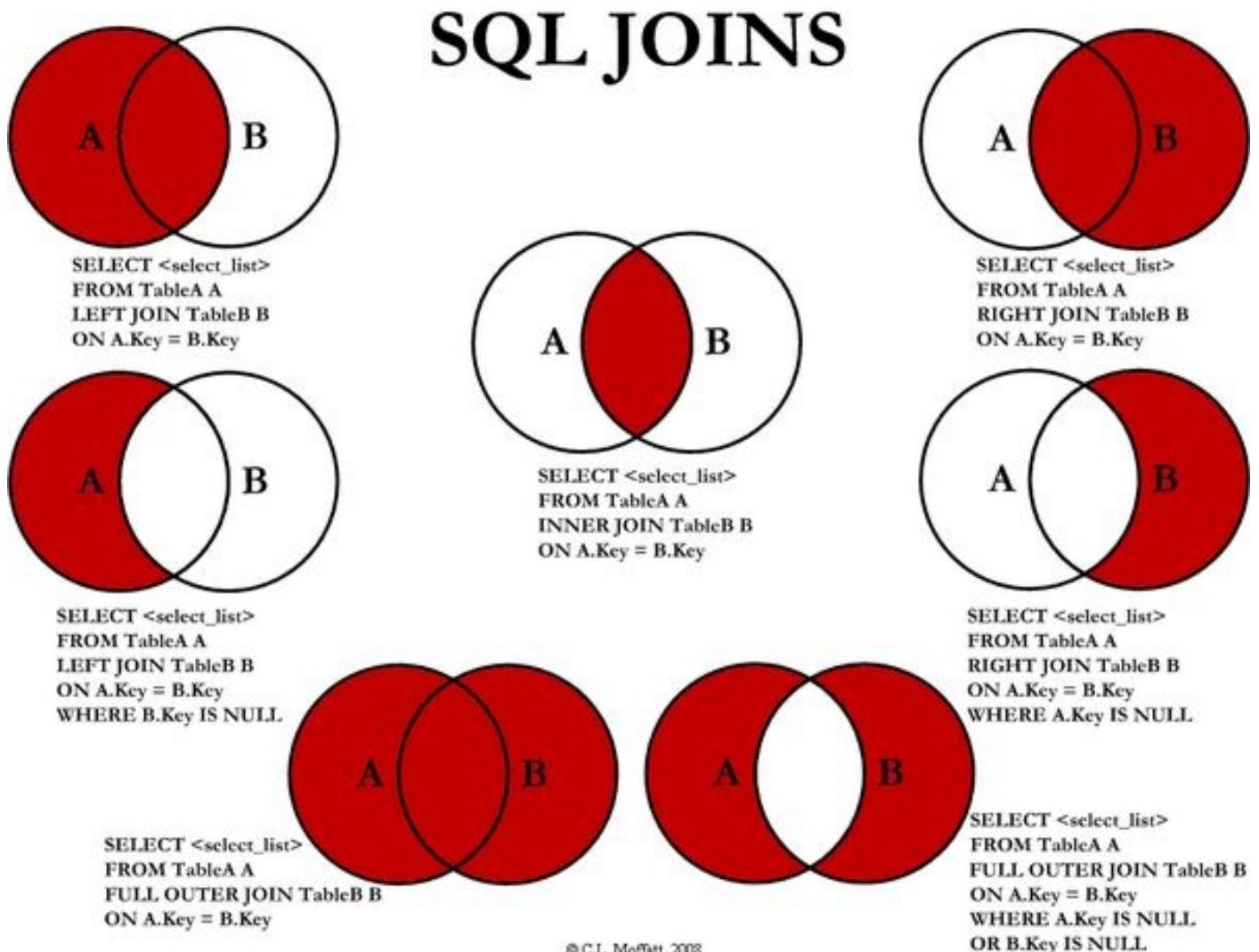
What's the difference between INNER JOIN, LEFT JOIN, RIGHT JOIN and FULL JOIN in MySQL?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

User: [lion-king](#)

[Answer](#) by [pranay-rana](#)

Reading this original article on **The Code Project** will help you a lot: [Visual Representation of SQL Joins](#).



Also check this post: [SQL SERVER – Better Performance – LEFT JOIN or NOT IN?](#)

Find original one at: [Difference between JOIN and OUTER JOIN in MySQL](#).

[Answer](#) by [brian-leeming](#)

INNER JOIN gets all records from one table that have some related entry in a second table

LEFT JOIN gets all records from the LEFT linked table but if you have selected some columns from the RIGHT table, if there is no related records, these columns will contain NULL

RIGHT JOIN is like the above but gets all records in the RIGHT table

FULL JOIN gets all records from both tables and puts NULL in the columns where related records do not exist in the opposite table

[Answer](#) by [arunprasanth-kv](#)

An SQL JOIN clause is used to combine rows from two or more tables, based on a common field between them.

There are different types of joins available in SQL:

INNER JOIN: returns rows when there is a match in both tables.

LEFT JOIN: returns all rows from the left table, even if there are no matches in the right table.

RIGHT JOIN: returns all rows from the right table, even if there are no matches in the left table.

FULL JOIN: returns rows when there is a match in one of the tables.

SELF JOIN: is used to join a table to itself as if the table were two tables, temporarily renaming at least one table in the SQL statement.

CARTESIAN JOIN: returns the Cartesian product of the sets of records from the two or more joined tables.

We can take each first four joins in Details :

We have two tables with the following values.

TableA

		lastName.....
1	arun	prasanth
2	ann	antony
3	sruthy	abc
6	new	abc

TableB

	id2	age	Place.....
1	24	kerala	
2	24	usa	
3	25	ekm	
5	24	chennai	

INNER JOIN

Note : it gives the intersection of the two tables, i.e. rows they have common in TableA and TableB

Syntax

```
SELECT table1.column1, table2.column2...
FROM table1
INNER JOIN table2
ON table1.common_field = table2.common_field;
```

Apply it in our sample table :

```
SELECT TableA.firstName, TableA.lastName, TableB.age, TableB.Place
FROM TableA
```

```
INNER JOIN TableB  
ON TableA.id = TableB.id2;
```

Result Will Be

firstName	lastName	age	Place.....
arun	prasanth	24	kerala
ann	antony	24	usa
sruthy	abc	25	ekm

LEFT JOIN

Note : will give all selected rows in TableA, plus any common selected rows in TableB.

Syntax

```
SELECT table1.column1, table2.column2...  
FROM table1  
LEFT JOIN table2  
ON table1.common_field = table2.common_field;
```

Apply it in our sample table :

```
SELECT TableA.firstName, TableA.lastName, TableB.age, TableB.Place  
FROM TableA  
LEFT JOIN TableB  
ON TableA.id = TableB.id2;
```

Result

firstName	lastName	age	Place.....
arun	prasanth	24	kerala
ann	antony	24	usa
sruthy	abc	25	ekm
new	abc	NULL	NULL

RIGHT JOIN

Note : will give all selected rows in TableB, plus any common selected rows in TableA.

Syntax

```
SELECT table1.column1, table2.column2...  
FROM table1  
RIGHT JOIN table2  
ON table1.common_field = table2.common_field;
```

Apply it in our sample table :

```
SELECT TableA.firstName, TableA.lastName, TableB.age, TableB.Place  
FROM TableA  
RIGHT JOIN TableB  
ON TableA.id = TableB.id2;
```

Result

firstName	lastName	age	Place.....
arun	prasanth	24	kerala
ann	antony	24	usa
sruthy	abc	25	ekm
NULL	NULL	24	chennai

FULL JOIN

Note : It is same as union operation, it will return all selected values from both tables.

Syntax

```
SELECT table1.column1, table2.column2...
```

```
FROM table1
FULL JOIN table2
ON table1.common_field = table2.common_field;
```

Apply it in our sample table :

```
SELECT TableA.firstName, TableA.lastName, TableB.age, TableB.Place
FROM TableA
FULL JOIN TableB
ON TableA.id = TableB.id2;
```

Result

firstName	lastName	age	Place.....
arun	prasanth	24	kerala
ann	antony	24	usa
sruthy	abc	25	ekm
new	abc	NULL	NULL
NULL	NULL	24	chennai

Interesting Fact

For INNER joins the order doesn't matter

For (LEFT, RIGHT or FULL) OUTER joins, the order matter

Better to go check this [Link](#) it will give you interesting details about join order

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL injection that gets around mysql_real_escape_string\(\)](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

Is there an SQL injection possibility even when using `mysql_real_escape_string()` function?

Consider this sample situation. SQL is constructed in PHP like this:

```
$login = mysql_real_escape_string(GetFromPost('login'));
$password = mysql_real_escape_string(GetFromPost('password'));

$sql = "SELECT * FROM table WHERE login='$login' AND password='$password'" ;
```

I have heard numerous people say to me that a code like that is still dangerous and possible to hack even with `mysql_real_escape_string()` function used. But I cannot think of any possible exploit?

Classic injections like this:

```
aaa' OR 1=1-
```

do not work.

Do you know of any possible injection that would get through the PHP code above?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

User: [richard-knop](#)

[Answer](#) by [wesley-van-opdorp](#)

Consider the following query:

```
$iId = mysql_real_escape_string("1 OR 1=1");
$sql = "SELECT * FROM table WHERE id = $iId";
```

`mysql_real_escape_string()` will not protect you against this. The fact that you use single quotes (' ') around your variables inside your query is what protects you against this. The following is also an option:

```
$iId = (int)"1 OR 1=1";
$sql = "SELECT * FROM table WHERE id = $iId";
```

[Answer](#) by [ircmaxell](#)

The short answer is **yes, yes there is a way to get around `mysql_real_escape_string()`.**

For Very OBSCURE EDGE CASES!!!

The long answer isn't so easy. It's based off an attack [demonstrated here](#).

The Attack

So, let's start off by showing the attack...

```
mysql_query('SET NAMES gbk');
$var = mysql_real_escape_string("\xbf\x27 OR 1=1 /*");
mysql_query("SELECT * FROM test WHERE name = '$var' LIMIT 1");
```

In certain circumstances, that will return more than 1 row. Let's dissect what's going on here:

1. Selecting a Character Set

```
mysql_query('SET NAMES gbk');
```

For this attack to work, we need the encoding that the server's expecting on the connection both to encode ' as in ASCII i.e. 0x27 and to have some character whose final byte is an ASCII \ i.e. 0x5c. As it turns out, there are 5 such encodings supported in MySQL 5.6 by default: big5, cp932, gb2312, gbk and sjis. We'll select gbk here.

Now, it's very important to note the use of `SET NAMES` here. This sets the character set **ON THE SERVER**. If we used the call to the C API function `mysql_set_charset()`, we'd be fine (on MySQL releases since 2006). But more on why in a minute...

2. The Payload

The payload we're going to use for this injection starts with the byte sequence 0xbf27. In gbk, that's an invalid multibyte character; in latin1, it's the string ¿'. Note that in latin1 **and** gbk, 0x27 on its own is a literal ' character.

We have chosen this payload because, if we called `addslashes()` on it, we'd insert

an ASCII \ i.e. 0x5c, before the ' character. So we'd wind up with 0xbf5c27, which in gbk is a two character sequence: 0xbf5c followed by 0x27. Or in other words, a *valid* character followed by an unescaped '. But we're not using addslashes(). So on to the next step...

3. mysql_real_escape_string()

The C API call to mysql_real_escape_string() differs from addslashes() in that it knows the connection character set. So it can perform the escaping properly for the character set that the server is expecting. However, up to this point, the client thinks that we're still using latin1 for the connection, because we never told it otherwise. We did tell the *server* we're using gbk, but the *client* still thinks it's latin1.

Therefore the call to mysql_real_escape_string() inserts the backslash, and we have a free hanging ' character in our “escaped” content! In fact, if we were to look at \$var in the gbk character set, we'd see:

```
續' OR 1=1 /*
```

Which is [exactly what](#) the attack requires.

4. The Query

This part is just a formality, but here's the rendered query:

```
SELECT * FROM test WHERE name = '續' OR 1=1 /*' LIMIT 1
```

Congratulations, you just successfully attacked a program using mysql_real_escape_string()...

The Bad

It gets worse. PDO defaults to *emulating* prepared statements with MySQL. That means that on the client side, it basically does a sprintf through mysql_real_escape_string() (in the C library), which means the following will result in a successful injection:

```
$pdo->query('SET NAMES gbk');
$stmt = $pdo->prepare('SELECT * FROM test WHERE name = ? LIMIT 1');
$stmt->execute(array("\xbf\x27 OR 1=1 /*"));
```

Now, it's worth noting that you can prevent this by disabling emulated prepared statements:

```
$pdo->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
```

This will *usually* result in a true prepared statement (i.e. the data being sent over in a separate packet from the query). However, be aware that PDO will silently [fallback](#) to emulating statements that MySQL can't prepare natively: those that it can are [listed](#) in the manual, but beware to select the appropriate server version).

The Ugly

I said at the very beginning that we could have prevented all of this if we had used mysql_set_charset('gbk') instead of SET NAMES gbk. And that's true provided you are

using a MySQL release since 2006.

If you're using an earlier MySQL release, then a [bug](#) in `mysql_real_escape_string()` meant that invalid multibyte characters such as those in our payload were treated as single bytes for escaping purposes *even if the client had been correctly informed of the connection encoding* and so this attack would still succeed. The bug was fixed in MySQL [4.1.20](#), [5.0.22](#) and [5.1.11](#).

But the worst part is that PDO didn't expose the C API for `mysql_set_charset()` until 5.3.6, so in prior versions it **cannot** prevent this attack for every possible command! It's now exposed as a [DSN parameter](#).

The Saving Grace

As we said at the outset, for this attack to work the database connection must be encoded using a vulnerable character set. [utf8mb4](#) is *not vulnerable* and yet can support *every* Unicode character: so you could elect to use that instead—but it has only been available since MySQL 5.5.3. An alternative is [utf8](#), which is also *not vulnerable* and can support the whole of the Unicode [Basic Multilingual Plane](#).

Alternatively, you can enable the [NO_BACKSLASH_ESCAPES](#) SQL mode, which (amongst other things) alters the operation of `mysql_real_escape_string()`. With this mode enabled, `\x27` will be replaced with `\x2727` rather than `\x5c27` and thus the escaping process *cannot* create valid characters in any of the vulnerable encodings where they did not exist previously (i.e. `\xbf27` is still `\bf27` etc.)—so the server will still reject the string as invalid. However, see [@eggyal's answer](#) for a different vulnerability that can arise from using this SQL mode.

Safe Examples

The following examples are safe:

```
mysql_query('SET NAMES utf8');
$var = mysql_real_escape_string("\xbf\x27 OR 1=1 /*");
mysql_query("SELECT * FROM test WHERE name = '$var' LIMIT 1");
```

Because the server's expecting utf8...

```
mysql_set_charset('gbk');
$var = mysql_real_escape_string("\xbf\x27 OR 1=1 /*");
mysql_query("SELECT * FROM test WHERE name = '$var' LIMIT 1");
```

Because we've properly set the character set so the client and the server match.

```
$pdo->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);
$pdo->query('SET NAMES gbk');
$stmt = $pdo->prepare('SELECT * FROM test WHERE name = ? LIMIT 1');
$stmt->execute(array("\xbf\x27 OR 1=1 /*"));
```

Because we've turned off emulated prepared statements.

```
$pdo = new PDO('mysql:host=localhost;dbname=testdb;charset=gbk', $user, $password);
$stmt = $pdo->prepare('SELECT * FROM test WHERE name = ? LIMIT 1');
$stmt->execute(array("\xbf\x27 OR 1=1 /*"));
```

Because we've set the character set properly.

```
$mysqli->query('SET NAMES gbk');
$stmt = $mysqli->prepare('SELECT * FROM test WHERE name = ? LIMIT 1');
$param = "\xbf\x27 OR 1=1 /*";
$stmt->bind_param('s', $param);
$stmt->execute();
```

Because MySQLi does true prepared statements all the time.

Wrapping Up

If you:

- Use Modern Versions of MySQL (late 5.1, all 5.5, 5.6, etc) **AND**
mysql_set_charset() / \$mysqli->set_charset() / PDO's DSN charset parameter
(in PHP ≥ 5.3.6)

OR

- Don't use a vulnerable character set for connection encoding (you only use utf8 / latin1 / ascii / etc)

You're 100% safe.

Otherwise, you're vulnerable **even though you're using mysql_real_escape_string()**

...

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

[Q: Set NOW\(\) as Default Value for datetime datatype?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

I have two columns in table users namely registerDate and lastVisitDate which consist of datetime data type. I would like to do the following.

1. Set registerDate defaults value to MySQL NOW()
2. Set lastVisitDate default value to 0000-00-00 00:00:00 Instead of null which it uses by default.

Because the table already exists and has existing records, I would like to use Modify table. I've tried using the two piece of code below, but neither works.

```
ALTER TABLE users MODIFY registerDate datetime DEFAULT NOW();
ALTER TABLE users MODIFY registerDate datetime DEFAULT CURRENT_TIMESTAMP;
```

It gives me Error : ERROR 1067 (42000): Invalid default value for 'registerDate'

Is it possible for me to set the default datetime value to NOW() in MySQL?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

User: [ibrahim-azhar-armar](#) 

[Answer](#) by [johan](#)

Set the datatype to timestamp.

```
CREATE TABLE mytable (
    mydate TIMESTAMP,
    ....)
```

See: <http://dev.mysql.com/doc/refman/5.1/en/create-table.html>

If you want to prevent MySQL from updating the timestamp value on UPDATE you can change the definition to:

```
CREATE TABLE mytable (
    mydate TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ....)
```

In 5.6.5 and above you can use the DATETIME type with a dynamic default value:

```
CREATE TABLE foo (
    creation_time DATETIME DEFAULT CURRENT_TIMESTAMP,
    modification_time DATETIME ON UPDATE CURRENT_TIMESTAMP
)
```

Reference: <http://optimize-this.blogspot.com/2012/04/datetime-default-now-finally-available.html>

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [datetime](#) ([Prev Q](#)) ([Next Q](#))

Q: 'IF' in 'SELECT' statement - choose output value based on column values

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

```
SELECT id, amount FROM report
```

I need amount to be amount if report.type='P' and -amount if report.type='N'. How do I add this to the above query?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [michael](#)

[Answer](#) by [felipe-alcacibar](#)

```
SELECT
    id
    , IF(type = 'P', amount, amount * -1) as amount
FROM report
```

See <http://dev.mysql.com/doc/refman/5.0/en/control-flow-functions.html>.

Additionally, you could handle when the condition is null. In the case of a null amount:

```
SELECT
    id
    , IF(type = 'P', IFNULL(amount,0), IFNULL(amount,0) * -1) as amount
FROM report
```

The part `IFNULL(amount, 0)` means *when amount is not null return amount else return 0*.

[Answer](#) by [mellamokb](#)

Use a case statement:

```
select id,
       case report.type
           when 'P' then amount
           when 'N' then -amount
       end as amount
  from `report`
```

[Answer](#) by [user1210826](#)

```
SELECT CompanyName,
       CASE WHEN Country IN ('USA', 'Canada') THEN 'North America'
            WHEN Country = 'Brazil' THEN 'South America'
            ELSE 'Europe' END AS Continent
  FROM Suppliers
 ORDER BY CompanyName;
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: MySql export schema without data](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I'm using a MySql database with a Java program, now I want to give the program to somebody else.

How to export the MySql database structure without the data in it, just the structure?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [darth-blue-ray](#)

[Answer](#) by [daric](#)

You can do with the `--no-data` option with mysqldump command

```
mysqldump -u root -p --no-data dbname > schema.sql
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Foreign key constraints: When to use ON UPDATE and ON DELETE](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#)), [foreign-keys](#) ([Next Q](#))

I'm designing my database schema using MySQL Workbench, which is pretty cool because you can do diagrams and it converts them :P

Anyways, I've decided to use InnoDB because of it's Foreign Key support. One thing I

noticed though is that it allows you to set On Update and on Delete options for foreign keys. Can someone explain where “Restrict”, “Cascade” and set null could be used in a simple example?

For example, say I have a user table which includes a user ID. And say I have a message table message which is a many-to-many which has 2 foreign keys (which reference the same primary key, user ID in the user table). Is setting the On Update and On Delete options any useful in this case? If so, which one do I choose? If this isn't a good example, could you please come up with a good example to illustrate how these could be useful?

Thanks

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#)), [foreign-keys](#) ([Next Q](#))

User: [meltuhamy](#) 

[Answer](#)  by [regilero](#) 

Do not hesitate to put constraints on the database. You'll be sure to have a consistent database, and that's one of the good reasons to use a database. Especially if you have several applications requesting it (or just one application but with a direct mode and a batch mode using different sources).

With MySQL you do not have advanced constraints like you would have in PostgreSQL but at least the foreign key constraints are quite advanced.

We'll take an example, a company table with a user table containing people from these company

```
CREATE TABLE COMPANY (company_id INT NOT NULL,
                     company_name VARCHAR(50),
                     PRIMARY KEY (company_id)
) ENGINE=INNODB;;
CREATE TABLE USER (user_id INT, user_name VARCHAR(50),
                   company_id INT,
                   INDEX company_id_idx (company_id),
                   FOREIGN KEY (company_id) REFERENCES COMPANY (company_id) ON...
) ENGINE=INNODB;
```

Let's look at the **ON UPDATE** clause:

- **ON UPDATE RESTRICT** : *the default* : if you try to update a company_id in table COMPANY the engine will reject the operation if one USER at least links on this company.
- **ON UPDATE NO ACTION** : same as RESTRICT.
- **ON UPDATE CASCADE** : *the best one usually* : if you update a company_id in a row of table COMPANY the engine will update it accordingly on all USER rows referencing this COMPANY (but no triggers activated on USER table, warning). The engine will track the changes for you, it's good.
- **ON UPDATE SET NULL** : if you update a company_id in a row of table COMPANY the engine will set related USERS company_id to NULL (should be available in USER company_id field). I cannot see any interesting thing to do with that on an update, but I may be wrong.

And now on the **ON DELETE** side:

- **ON DELETE RESTRICT** : *the default* : if you try to delete a company_id Id in table COMPANY the engine will reject the operation if one USER at least links on this company, can save your life.
- **ON DELETE NO ACTION** : same as RESTRICT
- **ON DELETE CASCADE** : *dangerous* : if you delete a company row in table COMPANY the engine will delete as well the related USERS. This is dangerous but can be used to make automatic cleanups on secondary tables (so it can be something you want, but quite certainly not for a COMPANY<->USER example)
- **ON DELETE SET NULL** : * handful* : if you delete a COMPANY row the related USERS will automatically have the relationship to NULL. If Null is your value for users with no company this can be a good behavior, for example maybe you need to keep the users in your application, as authors of some content, but removing the company is not a problem for you.

usually my default is: **ON DELETE RESTRICT ON UPDATE CASCADE**. with some ON DELETE CASCADE for track tables (logs—not all logs—, things like that) and ON DELETE SET NULL when the master table is a ‘simple attribute’ for the table containing the foreign key, like a JOB table for the USER table.

Edit

It's been a long time since I wrote that. Now I think I should add one important warning. MySQL has one big documented limitation with cascades. **Cascades are not firing triggers**. So if you were over confident enough in that engine to use triggers you should avoid cascades constraints.

- <http://dev.mysql.com/doc/refman/5.6/en/triggers.html> 

MySQL triggers activate only for changes made to tables by SQL statements. They do not activate for changes in views, nor by changes to tables made by APIs that do not transmit SQL statements to the MySQL Server

- <http://dev.mysql.com/doc/refman/5.6/en/stored-program-restrictions.html#stored-routines-trigger-restrictions> 

Triggers are not activated by foreign key actions.

And I do not think this will get fixed one day. Foreign key constraints are managed by the InnoDB storage and Triggers are managed by the MySQL SQL engine. Both are separated. Innodb is the only storage with constraint management, maybe they'll add triggers directly in the storage engine one day, maybe not.

But I have my own opinion on which element you should choose between the poor trigger implementation and the very useful foreign keys constraints support. And once you'll get used to database consistency you'll love PostgreSQL.

[Q: How to move columns in a MySQL table?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Currently I am having the following MySQL table: Employees (empID, empName, department);

I want to change the table to the following: Employees (empID, department, empName);

How can this be done using ALTER statements?

Note: I want to change only column positions.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [isumitg](#)

[Answer](#) by [ted-hopp](#)

If empName is a VARCHAR(50) column:

```
ALTER TABLE Employees MODIFY COLUMN empName VARCHAR(50) AFTER department;
```

EDIT

Per the comments, you can also do this:

```
ALTER TABLE Employees CHANGE COLUMN empName empName VARCHAR(50) AFTER department;
```

Note that the repetition of empName is deliberate. You have to tell MySQL that you want to keep the same column name.

You should be aware that both syntax versions are specific to MySQL. They won't work, for example, in PostgreSQL or many other DBMSs.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL Select only rows with Max Value on a Column](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have this table for documents (simplified version here):

id	rev	content
1	1	...
2	1	...
1	2	...
1	3	...

How do I select one row per id and only the greatest rev?

With the above data, the result should contain two rows: [1, 3, ...] and [2, 1, ...].

I'm using MySQL.

Currently I use checks in the while loop to detect and over-write old revs from the resultset. But is this the only method to achieve the result? Isn't there a **SQL** solution?

Update

As the answers suggest, there *is* a SQL solution, and [here a sqlfiddle demo](#).

Update 2

I noticed after adding the above sqlfiddle, the rate at which the question is upvoted has surpassed the upvote rate of the answers. That has not been the intention! The fiddle *is* based on the answers, especially the accepted answer.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [majid-fouladpour](#)

[Answer](#) by [adrian-carneiro](#)

At first glance...

All you need is a GROUP BY clause with the MAX aggregate function:

```
SELECT id, MAX(rev)
FROM YourTable
GROUP BY id
```

It's never that simple, is it?

I just noticed you need the content column as well.

This is a very common question in SQL: find the whole data for the row with some max value in a column per some group identifier. I heard that a lot during my career. Actually, it was one of the questions I answered in my current job's technical interview.

It is, actually, so common that StackOverflow community has created a single tag just to deal with questions like that: [greatest-n-per-group](#).

Basically, you have two approaches to solve that problem:

Joining with simple group-identifier, max-value-in-group Sub-query

In this approach, you first find the group-identifier, max-value-in-group (already solved above) in a sub-query. Then you join your table to the sub-query with equality on both group-identifier and max-value-in-group:

```
SELECT a.id, a.rev, a.contents
FROM YourTable a
INNER JOIN (
    SELECT id, MAX(rev) rev
    FROM YourTable
    GROUP BY id
) b ON a.id = b.id AND a.rev = b.rev
```

Left Joining with self, tweaking join conditions and filters

In this approach, you left join the table with itself. Equality, of course, goes in the group-identifier. Then, 2 smart moves:

1. The second join condition is having left side value less than right value
2. When you do step 1, the row(s) that actually have the max value will have `NULL` in the right side (it's a `LEFT JOIN`, remember?). Then, we filter the joined result, showing only the rows where the right side is `NULL`.

So you end up with:

```
SELECT a.*  
FROM YourTable a  
LEFT OUTER JOIN YourTable b  
    ON a.id = b.id AND a.rev < b.rev  
WHERE b.id IS NULL;
```

Conclusion

Both approaches bring the exact same result.

If you have two rows with `max-value-in-group` for `group-identifier`, both rows will be in the result in both approaches.

Both approaches are SQL ANSI compatible, thus, will work with your favorite RDBMS, regardless of its “flavor”.

Both approaches are also performance friendly, however your mileage may vary (RDBMS, DB Structure, Indexes, etc.). So when you pick one approach over the other, *benchmark*. And make sure you pick the one which make most of sense to you.

[Answer](#)  by [kevin-burton](#) 

My preference is to use as little code as possible...

You can do it using `IN` try this:

```
SELECT *  
FROM t1 WHERE (id,rev) IN  
( SELECT id, MAX(rev)  
    FROM t1  
   GROUP BY id  
)
```

to my mind it is less complicated... easier to read and maintain.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[**Q: “where 1=1” statement**](#) 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

Possible Duplicate:

[Why would someone use WHERE 1=1 AND <conditions> in a SQL clause?](#)

I saw some people use a statement to query a table in a MySQL database like the following:

```
select * from car_table where 1=1 and value="TOYOTA"
```

But what does 1=1 mean here?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

User: [mellan](#) 

[Answer](#)  by [gbn](#) 

It's usually when folks build up SQL statements.

When you add `and value = "Toyota"` you don't have to worry about whether there is a condition before or just WHERE. The optimiser should ignore it

No magic, just practical

Example Code:

```
commandText = "select * from car_table where 1=1";  
  
if (modelYear <> 0)      commandText += " and year="+modelYear  
if (manufacturer <> "")  commandText += " and value="+QuotedStr(manufacturer)  
if (color <> "")        commandText += " and color="+QuotedStr(color)  
if (california)          commandText += " and hasCatalyticic=1"
```

Otherwise you would have to have a complicated set of logic:

[Skip code block](#)

```
commandText = "select * from car_table"  
whereClause = "";  
if (modelYear <> 0)  
{  
    if (whereClause <> "")  
        whereClause = whereClause + " and ";  
    commandText += "year="+modelYear;  
}  
if (manufacturer <> "")  
{  
    if (whereClause <> "")  
        whereClause = whereClause + " and ";  
    commandText += "value="+QuotedStr(manufacturer)  
}  
if (color <> "")  
{  
    if (whereClause <> "")  
        whereClause = whereClause + " and ";  
    commandText += "color="+QuotedStr(color)  
}  
if (california)  
{  
    if (whereClause <> "")  
        whereClause = whereClause + " and ";  
    commandText += "hasCatalyticic=1"  
}  
  
if (whereClause <> "")  
    commandText = commandText + "WHERE "+whereClause;
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: How do I check if a column is empty or null in mysql](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have a column in a table which might contain null or empty values. How do I check if a column is empty or null in the rows present in a table.

(e.g. null or '' or ' ' or ' ' and...)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [priya](#)

[Answer](#) by [maček](#)

This will select all rows where some_col is NULL or '' (empty string)

```
SELECT * FROM table WHERE some_col IS NULL OR some_col = '';
```

[Answer](#) by [onedaywhen](#)

As defined by the SQL-92 Standard, when comparing two strings of differing widths, the narrower value is right-padded with spaces to make it is same width as the wider value. Therefore, all values that consist exclusively involving zero or more spaces will be deemed to be equal e.g.

```
'' = ' ' IS TRUE
'' = '   ' IS TRUE
' ' = '   ' IS TRUE
'   ' = '       ' IS TRUE
etc
```

Therefore, this should work regardless of how many spaces make up the some_col value:

```
SELECT *
  FROM T
 WHERE some_col IS NULL
   OR some_col = '';
```

or more succinctly:

```
SELECT *
  FROM T
 WHERE NULLIF(some_col, '') IS NULL;
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: ALTER TABLE to add a composite primary key](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have a table called provider. I have three columns called person, place, thing. There can be duplicate persons, duplicate places, and duplicate things, but there can never be a duplicate person-place-thing combination.

How would I ALTER TABLE to add a composite primary key for this table in MySQL

with the these three columns?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [david542](#) 

[Answer](#)  by [adrian-cornish](#) 

```
ALTER TABLE provider ADD PRIMARY KEY(person,place,thing);
```

If a primary key already exists then you want to do this

```
ALTER TABLE provider DROP PRIMARY KEY, ADD PRIMARY KEY(person, place, thing);
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Q: How to reset AUTO_INCREMENT in MySQL?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

How can I reset the auto-increment of a field? I want it to start counting from 1 again.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [some-folk](#) 

[Answer](#)  by [niels](#) 

You can reset the counter with:

```
ALTER TABLE tablename AUTO_INCREMENT = 1
```

For InnoDB you cannot set the auto_increment value lower or equal to the highest current index. (quote from [ViralPatel](#) ):

Note that you cannot reset the counter to a value less than or equal to any that have already been used. For MyISAM, if the value is less than or equal to the maximum value currently in the AUTO_INCREMENT column, the value is reset to the current maximum plus one. For InnoDB, if the value is less than the current maximum value in the column, no error occurs and the current sequence value is not changed.

See [How to Reset an MySQL AutoIncrement using a MAX value from another table?](#)  on how to dynamically get an acceptable value.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

Q: How to copy a row and insert in same table with a autoincrement field in MySQL?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have a table Eg- tab . What I am trying to do is copying a row with an autoincrement column ID=1 and insert the data into same table with a row and column ID=2.

Using MySql. How can I do this in a single query?Please help

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [navdroid](#) 

[Answer](#)  by [mu-is-too-short](#) 

Use `INSERT... SELECT`:

```
insert into your_table (c1, c2, ...)
select c1, c2, ...
from your_table
where id = 1
```

where `c1, c2, ...` are all the columns except `id`. If you want to explicitly insert with an `id` of 2 then include that in your `INSERT` column list and your `SELECT`:

```
insert into your_table (id, c1, c2, ...)
select 2, c1, c2, ...
from your_table
where id = 1
```

You'll have to take care of a possible duplicate `id` of 2 in the second case of course.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: When to use single quotes, double quotes, and backticks?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

I am trying to learn the best way to write queries. I also understand the importance of being consistent, and up until now I have essentially randomly used single quotes, double quotes, and backticks without any real thought.

Example:

```
$query = 'INSERT INTO table (id, col1, col2) VALUES (NULL, val1, val2)';
```

Also, in the above example, consider that "table," "col[n]," and "val[n]" may be variables.

What is the standard for this? What do you do?

I've been reading answers to similar questions on here for about 20 minutes, but it seems like there is no definitive answer to this question.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

User: [nate](#) 

[Answer](#)  by [michael-berkowski](#) 

Backticks are to be used for table and column identifiers, but are only necessary when the identifier is a [MySQL reserved keyword](#), or when the identifier contains whitespace

characters or characters beyond a limited set (see below) It is often recommended to avoid using reserved keywords as column or table identifiers when possible, avoiding the quoting issue.

Single quotes should be used for string values like in the VALUES() list. Double quotes are supported by MySQL for string values as well, but single quotes are more widely accepted by other RDBMS, so it is a good habit to use single quotes instead of double.

MySQL also expects DATE and DATETIME literal values to be single-quoted as strings like
'2001-01-01 00:00:00'

So using your example, I would double-quote the PHP string and use single quotes on the values 'val1', 'val2'. NULL is a MySQL keyword, and a special (non)-value, and is therefore unquoted.

None of these table or column identifiers are reserved words or make use of characters requiring quoting, but I've quoted them anyway with backticks (more on this later...).

Functions native to the RDBMS (for example, `NOW()` in MySQL) should not be quoted, although their arguments are subject to the same string or identifier quoting rules already mentioned.

```
$query = "INSERT INTO `table` (`id`, `col1`, `col2`, `date`, `updated`) VALUES (NULL, 'val1', 'val2',  
//scroll >>>  
//-----  
//-----  
//----- ^^^^^^ Unquoted keyword  
//----- ^^^^^^ Backtick table & column
```

Variable interpolation

The quoting patterns for variables do not change, although if you intend to interpolate the variables directly in a string, it must be double-quoted in PHP. Just make sure that you have properly escaped the variables for use in SQL. ([It is recommended to use an API supporting prepared statements instead, as protection against SQL injection](#)).

```
// Same thing with some variable replacements
// Here, a variable table name $table is backtick-quoted, and variables
// in the VALUES list are single-quoted
$query = "INSERT INTO `{$table}` (`id`, `col1`, `col2`, `date`) VALUES (NULL, '$val1', '$val2', '$date')
```

Prepared statements

When working with prepared statements, consult the documentation to determine whether or not the statement's placeholders must be quoted. The most popular APIs available in PHP, PDO and MySQLi, expect *unquoted* placeholders, as do most prepared statement APIs in other languages:

```
// PDO example with named parameters, unquoted
$query = "INSERT INTO `table` (`id`, `col1`, `col2`, `date`) VALUES (:id, :col1, :col2, :date)";

// MySQLi example with ? parameters, unquoted
$query = "INSERT INTO `table` (`id`, `col1`, `col2`, `date`) VALUES (?, ?, ?, ?)";
```

Characters requiring backtick quoting in identifiers:

[According to MySQL documentation](#), you do not need to quote (backtick) identifiers using the following character set:

ASCII: [0-9, a-z, A-Z\$_] (basic Latin letters, digits 0-9, dollar, underscore)

You can use characters beyond that set as table or column identifiers, including whitespace for example, but then you *must* quote (backtick) them.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [php](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can an SQL query return data from multiple tables](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#))

I would like to know the following:

- how to get data from multiple tables in my database?
- what types of methods are there to do this?
- what are joins and unions and how are they different from one another?
- When should I use each one compared to the others?

I am planning to use this in my (for example - PHP) application, but don't want to run multiple queries against the database, what options do I have to get data from multiple tables in a single query?

Note: I am writing this as I would like to be able to link to a well written guide on the numerous questions that I constantly come across in the PHP queue, so I can link to this for further detail when I post an answer.

The answers cover off the following:

1. [Part 1 - Joins and Unions](#)
2. [Part 2 - Subqueries](#)
3. [Part 3 - Tricks and Efficient Code](#)
4. [Part 4 - Subqueries in the From Clause](#)
5. [Part 5 - Mixed Bag of John's Tricks](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#))

User: [fluffeh](#)

[Answer](#) by [john-woo](#)

Ok, I found this post very interesting and I would like to share some of my knowledge on creating a query. Thanks for this [Fluffeh](#). Others who may read this and may feel that I'm wrong are 101% free to edit and criticise my answer. (*Honestly, I feel very thankful for correcting my mistake(s).*)

I'll be posting some of the frequently asked questions in MySQL tag.

Trick No. 1 (*rows that matches to multiple conditions*)

Given this schema

[Skip code block](#)

```
CREATE TABLE MovieList
(
    ID INT,
    MovieName VARCHAR(25),
    CONSTRAINT ml_pk PRIMARY KEY (ID),
    CONSTRAINT ml_uq UNIQUE (MovieName)
);

INSERT INTO MovieList VALUES (1, 'American Pie');
INSERT INTO MovieList VALUES (2, 'The Notebook');
INSERT INTO MovieList VALUES (3, 'Discovery Channel: Africa');
INSERT INTO MovieList VALUES (4, 'Mr. Bean');
INSERT INTO MovieList VALUES (5, 'Expendables 2');

CREATE TABLE CategoryList
(
    MovieID INT,
    CategoryName VARCHAR(25),
    CONSTRAINT cl_uq UNIQUE(MovieID, CategoryName),
    CONSTRAINT cl_fk FOREIGN KEY (MovieID) REFERENCES MovieList(ID)
);

INSERT INTO CategoryList VALUES (1, 'Comedy');
INSERT INTO CategoryList VALUES (1, 'Romance');
INSERT INTO CategoryList VALUES (2, 'Romance');
INSERT INTO CategoryList VALUES (2, 'Drama');
INSERT INTO CategoryList VALUES (3, 'Documentary');
INSERT INTO CategoryList VALUES (4, 'Comedy');
INSERT INTO CategoryList VALUES (5, 'Comedy');
INSERT INTO CategoryList VALUES (5, 'Action');
```

QUESTION

Find *all movies* that belong to at least *both* Comedy and Romance categories.

Solution

This question can be very tricky sometimes. It may seem that a query like this will be the answer:-

```
SELECT DISTINCT a.MovieName
FROM MovieList a
INNER JOIN CategoryList b
    ON a.ID = b.MovieID
WHERE b.CategoryName = 'Comedy' AND
    b.CategoryName = 'Romance'
```

[SQLFiddle Demo](#)

which is definitely very wrong because it produces *no result*. The explanation of this is that there is only one valid value of `CategoryName` on *each row*. For instance, the first condition returns *true*, the second condition is always false. Thus, by using `AND` operator, both condition should be true; otherwise, it will be false. Another query is like this,

```
SELECT DISTINCT a.MovieName
FROM MovieList a
INNER JOIN CategoryList b
    ON a.ID = b.MovieID
WHERE b.CategoryName IN ('Comedy', 'Romance')
```

[SQLFiddle Demo](#)

and the result is still incorrect because it matches to record that has *at least* one match on the categoryName. The **real solution** would be by counting the number of record instances per movie. The number of instance should match to the total number of the values supplied in the condition.

```
SELECT a.MovieName
FROM MovieList a
    INNER JOIN CategoryList b
        ON a.ID = b.MovieID
WHERE b.CategoryName IN ('Comedy', 'Romance')
GROUP BY a.MovieName
HAVING COUNT(*) = 2
```

[SQLFiddle Demo \(the answer\)](#)

- [SQL of Relational Division](#) 
-

Trick No. 2 (*maximum record for each entry*)

Given schema,

[Skip code block](#)

```
CREATE TABLE Software
(
    ID INT,
    SoftwareName VARCHAR(25),
    Descriptions VARCHAR(150),
    CONSTRAINT sw_pk PRIMARY KEY (ID),
    CONSTRAINT sw_uq UNIQUE (SoftwareName)
);

INSERT INTO Software VALUES (1, 'PaintMe', 'used for photo editing');
INSERT INTO Software VALUES (2, 'World Map', 'contains map of different places of the world');
INSERT INTO Software VALUES (3, 'Dictionary', 'contains description, synonym, antonym of the words');

CREATE TABLE VersionList
(
    SoftwareID INT,
    VersionNo INT,
    DateReleased DATE,
    CONSTRAINT sw_uq UNIQUE (SoftwareID, VersionNo),
    CONSTRAINT sw_fk FOREIGN KEY (SoftwareID) REFERENCES Software(ID)
);

INSERT INTO VersionList VALUES (3, 2, '2009-12-01');
INSERT INTO VersionList VALUES (3, 1, '2009-11-01');
INSERT INTO VersionList VALUES (3, 3, '2010-01-01');
INSERT INTO VersionList VALUES (2, 2, '2010-12-01');
INSERT INTO VersionList VALUES (2, 1, '2009-12-01');
INSERT INTO VersionList VALUES (1, 3, '2011-12-01');
INSERT INTO VersionList VALUES (1, 2, '2010-12-01');
INSERT INTO VersionList VALUES (1, 1, '2009-12-01');
INSERT INTO VersionList VALUES (1, 4, '2012-12-01');
```

QUESTION

Find the latest version on each software. Display the following columns:
SoftwareName, Descriptions, LatestVersion (*from VersionNo column*), DateReleased

Solution

Some SQL developers mistakenly use MAX() aggregate function. They tend to create like this,

```
SELECT a.SoftwareName, a.Descriptions,
       MAX(b.VersionNo) AS LatestVersion, b.DateReleased
  FROM Software a
    INNER JOIN VersionList b
      ON a.ID = b.SoftwareID
 GROUP BY a.ID
ORDER BY a.ID
```

[SQLFiddle Demo](#)

(most RDBMS generates a syntax error on this because of not specifying some of the non-aggregated columns on the group by clause) the result produces the correct LatestVersion on each software but obviously the DateReleased are incorrect. MySQL doesn't support Window Functions and Common Table Expression yet as some RDBMS do already. The workaround on this problem is to create a subquery which gets the individual maximum versionNo on each software and later on be joined on the other tables.

[Skip code block](#)

```
SELECT a.SoftwareName, a.Descriptions,
       b.LatestVersion, c.DateReleased
  FROM Software a
    INNER JOIN
    (
        SELECT SoftwareID, MAX(VersionNo) LatestVersion
        FROM VersionList
        GROUP BY SoftwareID
    ) b ON a.ID = b.SoftwareID
    INNER JOIN VersionList c
        ON c.SoftwareID = b.SoftwareID AND
           c.VersionNo = b.LatestVersion
 GROUP BY a.ID
ORDER BY a.ID
```

[SQLFiddle Demo \(the answer\)](#)

So that was it. I'll be posting another soon as I recall any other FAQ on MySQL tag. Thank you for reading this little article. I hope that you have atleast get even a little knowledge from this.

UPDATE 1

Trick No. 3 (*Finding the latest record between two IDs*)

Given Schema

[Skip code block](#)

```
CREATE TABLE userList
(
    ID INT,
    NAME VARCHAR(20),
    CONSTRAINT us_pk PRIMARY KEY (ID),
    CONSTRAINT us_uq UNIQUE (NAME)
);
```

```

INSERT INTO userList VALUES (1, 'Fluffeh');
INSERT INTO userList VALUES (2, 'John Woo');
INSERT INTO userList VALUES (3, 'hims056');

CREATE TABLE CONVERSATION
(
    ID INT,
    FROM_ID INT,
    TO_ID INT,
    MESSAGE VARCHAR(250),
    DeliveryDate DATE
);

INSERT INTO CONVERSATION VALUES (1, 1, 2, 'hi john', '2012-01-01');
INSERT INTO CONVERSATION VALUES (2, 2, 1, 'hello fluff', '2012-01-02');
INSERT INTO CONVERSATION VALUES (3, 1, 3, 'hey hims', '2012-01-03');
INSERT INTO CONVERSATION VALUES (4, 1, 3, 'please reply', '2012-01-04');
INSERT INTO CONVERSATION VALUES (5, 3, 1, 'how are you?', '2012-01-05');
INSERT INTO CONVERSATION VALUES (6, 3, 2, 'sample message!', '2012-01-05');

```

QUESTION

Find the latest conversation between two users.

Solution

[Skip code block](#)

```

SELECT      b.Name SenderName,
            c.Name RecipientName,
            a.Message,
            a.DeliveryDate
FROM        Conversation a
            INNER JOIN userList b
                ON a.From_ID = b.ID
            INNER JOIN userList c
                ON a.To_ID = c.ID
WHERE       (LEAST(a.FROM_ID, a.TO_ID), GREATEST(a.FROM_ID, a.TO_ID), DeliveryDate)
IN
(
    SELECT  LEAST(FROM_ID, TO_ID) minFROM,
            GREATEST(FROM_ID, TO_ID) maxTo,
            MAX(DeliveryDate) maxDate
    FROM    Conversation
    GROUP BY minFROM, maxTo
)

```

[SQLFiddle Demo](#) 

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to temporarily disable a foreign key constraint in MySQL?](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [django](#) ([Next Q](#))

Is it possible to temporarily disable constraints in MySQL?

I have two Django models, each with a ForeignKey to the other one. Deleting instances of a model returns an error because of the ForeignKey constraint:

```

cursor.execute("DELETE FROM myapp_item WHERE n = %s", n)
transaction.commit_unless_managed() #a foreign key constraint fails here

cursor.execute("DELETE FROM myapp_style WHERE n = %s", n)
transaction.commit_unless_managed()

```

Is it possible to temporarily disable constraints and delete anyway?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [django](#) ([Next Q](#))

User: [jul](#) 

[Answer](#)  by [andrew-campbell](#) 

Try DISABLE KEYS or

```
SET FOREIGN_KEY_CHECKS=0;
```

make sure to

```
SET FOREIGN_KEY_CHECKS=1;
```

after.

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#)), [django](#) ([Next Q](#))

[**Q: How to import an SQL file using the command line in MySQL?**](#)

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

I have a .sql file with an export from phpMyAdmin. I want to import it into a different server using the command line.

I have a [Windows Server 2008](#)  R2 installation. I placed the .sql file on the C drive, and I tried this command

```
database_name < file.sql
```

It is not working I get syntax errors.

How can I import this file without a problem?

Do I need to create a database first?

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

User: [jaylen](#) 

[Answer](#)  by [bansi](#) 

Try:

```
mysql -u username -p database_name < file.sql
```

Check [MySQL Options](#) .

Note: It is better to use the full path of the SQL file file.sql.

[Answer](#)  by [vladkras](#) 

A common use of [mysqldump](#)  is for making a backup of an entire database:

```
shell> mysqldump db_name > backup-file.sql
```

You can load the dump file back into the server like this:

UNIX

```
shell> mysql db_name < backup-file.sql
```

The same in **Windows** command prompt:

```
mysql -p -u[user] [database] < backup-file.sql
```

PowerShell

```
C:\> cmd.exe /c "mysql -u root -p db_name < backup-file.sql"
```

MySQL command line

```
mysql> use db_name;
mysql> source backup-file.sql;
```

Tags: [mysql](#) ([Prev Q](#)) ([Next Q](#))

[Q: What is this operator <=> in MySQL?](#)

Tags: [mysql](#) ([Prev Q](#))

I'm working on code written by a previous developer and in a query it says,

```
WHERE p.name <=> NULL
```

What does `<=>` mean in this query? Is it something equal to `=`? Or is it a syntax error?

But it is not showing any errors or exceptions. I already know that `<>` = `!=` in [MySQL](#) .

Tags: [mysql](#) ([Prev Q](#))

User: [zzlalani](#) 

[Answer](#)  by [jack](#) 

Similarity with `=` operator

Like the regular `=` operator, two values are compared and the result is either `0` (not equal) or `1` (equal); in other words: `'a' <=> 'b'` yields `0` and `'a' <=> 'a'` yields `1`.

Difference with `=` operator

Unlike the regular `=` operator, values of `NULL` don't have a special meaning and so it never yields `NULL` as a possible outcome; so: `'a' <=> NULL` yields `0` and `NULL <=> NULL` yields `1`.

Contrary to `=`, whereby `'a' = NULL` yields `NULL` and even `NULL = NULL` yields `NULL`; BTW, almost all operators and functions in MySQL work in this manner, because comparing against `NULL` is basically undefined.

Usefulness

This is very useful for when both operands may contain `NULL` and you need a consistent comparison result between two columns.

Another use-case is with prepared statements, for example:

```
... WHERE col_a <=> ? ...
```

Here, the placeholder can be either a scalar value or `NULL` without having to change anything about the query.

Related operators

Besides `<=>` there are also two other operators that can be used to compare against `NULL`, namely `IS NULL` and `IS NOT NULL`; they're part of the ANSI standard and therefore supported on other databases, unlike `<=>`, which is MySQL-specific.

You can think of them as specializations of MySQL's `<=>`:

```
'a' IS NULL      ==> 'a' <=> NULL
'a' IS NOT NULL ==> NOT('a' <=> NULL)
```

Based on this, your particular query (fragment) can be converted to the more portable:

```
WHERE p.name IS NULL
```

Tags: [mysql](#) ([Prev Q](#))

Tsql

[Skip to questions,](#)

Wiki by user [omg-ponies](#) 

[T-SQL](#)  (Transact Structured Query Language) is the extension of SQL functionality supported by Sybase ASE and Microsoft SQL Server. They share the name because Microsoft SQL Server was based on Sybase ASE code in a partnership that later dissolved. While the name is shared, functionality is not necessarily identical. With the introduction of Microsoft SQL Server 2005, the SQL Server code was completely re-written to provide integration with .Net.

Transact-SQL is central to using Microsoft SQL Server. All applications that communicate with an instance of SQL Server do so by sending Transact-SQL statements to the server, regardless of the user interface of the application.

It adheres to nearly all SQL standards, yet supplies an additional set of functionality not a part of the SQL standard, with concepts such as common table expressions, index hints, and the ability to SELECT scalars.

Questions about T-SQL should be tagged appropriately with either [sql-server](#) or [sybase](#), and include the version as well.

More information can be found in [SQL Server Transact-SQL Reference](#)  and [Sybase Transact-SQL User's Guide](#) .

Questions

[Q: What is the difference between SQL, PL-SQL and T-SQL?](#)

Tags: [tsql](#) ([Prev Q](#)) ([Next Q](#))

What is the difference between SQL, PL-SQL and T-SQL?

Can anyone explain what the differences between these three are, and provide scenarios where each would be relevantly used?

Tags: [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [goober](#) 

[Answer](#)  by [quassnoi](#) 

- SQL is a query language to operate on sets.
It is more or less standardized, and used by almost all relational database management systems: SQL Server, Oracle, MySQL, PostgreSQL, DB2, Informix, etc.
- PL/SQL is a proprietary procedural language used by Oracle
- PL/pgSQL is a procedural language used by PostgreSQL
- TSQL is a proprietary procedural language used by Microsoft in SQL Server.

Procedural languages are designed to extend SQL's abilities while being able to integrate well with SQL. Several features such as local variables and string/data processing are added. These features make the language Turing-complete.

They are also used to write stored procedures: pieces of code residing on the server to manage complex business rules that are hard or impossible to manage with pure set-based operations.

Tags: [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How do I DROP multiple columns with a single ALTER TABLE statement?](#)

Tags: [tsql](#) ([Prev Q](#)) ([Next Q](#))

I would like to write a single SQL command to drop multiple columns from a single table in one ALTER TABLE statement.

From [MSDN's ALTER TABLE documentation](#)...

```
DROP { [CONSTRAINT] constraint_name | COLUMN column_name }
```

Specifies that constraint_name or column_name is removed from the table. DROP COLUMN is not allowed if the compatibility level is 65 or earlier. Multiple columns and constraints can be listed.

It says that multiple columns can be listed in the the statement but the syntax doesn't show an optional comma or anything that would even hint at the syntax.

How should I write my SQL to drop multiple columns in one statement (if possible)?

Tags: [tsql](#) ([Prev Q](#)) ([Next Q](#))

User: [jesse-webb](#)

[Answer](#) by [alex-aza](#)

```
alter table TableName  
drop column Column1, Column2
```

The syntax is

```
DROP { [ CONSTRAINT ] constraint_name | COLUMN column } [ ,...n ]
```

[Answer](#) by [dir](#)

So summarizing

[Oracle](#):

```
ALTER TABLE table_name DROP (column_name1, column_name2);
```

[MS SQL](#):

```
ALTER TABLE table_name DROP COLUMN column_name1, column_name2
```

[MySql](#):

```
ALTER TABLE table_name DROP column_name1, DROP column_name2;
```

[Postgre SQL](#)

```
ALTER TABLE table_name DROP COLUMN column_name1, DROP COLUMN column_name2;
```

Tags: [tsql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Get top 1 row of each group](#)

Tags: [tsql](#) ([Prev Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#))

I have a table which I want to get the latest entry for each group. Here's the table:

DocumentStatusLogs Table

ID	DocumentID	Status	DateCreated
2	1	S1	7/29/2011
3	1	S2	7/30/2011
6	1	S1	8/02/2011
1	2	S1	7/28/2011
4	2	S2	7/30/2011
5	2	S3	8/01/2011
6	3	S1	8/02/2011

The table will be grouped by DocumentID and sorted by DateCreated in descending order. For each DocumentID, I want to get the latest status.

My preferred output:

DocumentID	Status	DateCreated
1	S1	8/02/2011
2	S3	8/01/2011
3	S1	8/02/2011

- Is there any aggregate function to get only the top from each group? See pseudo-code `GetOnlyTheTop` below:

```
select DocumentID, GetOnlyTheTop(Status), GetOnlyTheTop(DateCreated)
from DocumentStatusLogs group by DocumentID order by DateCreated desc
```

- If such function doesn't exist, is there any way I can achieve the output I want?
- Or at the first place, could this be caused by unnormalized database? I'm thinking, since what I'm looking for is just one row, should that status also be located in the parent table?

Please see the parent table for more information:

Current Documents Table

DocumentID	Title	Content	DateCreated
1	TitleA
2	TitleB
3	TitleC

Should the parent table be like this so that I can easily access its status?

DocumentID	Title	Content	DateCreated	CurrentStatus
1	TitleA	s1
2	TitleB	s3
3	TitleC	s1

UPDATE I just learned how to use “apply” which makes it easier to address such problems.

Tags: [tsql](#) ([Prev Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#))

User: [dpp](#)

[Answer](#) by [gbn](#)

```
;WITH cte AS
(
    SELECT *,
        ROW_NUMBER() OVER (PARTITION BY DocumentID ORDER BY DateCreated DESC) AS rn
    FROM DocumentStatusLogs
)
SELECT *
FROM cte
WHERE rn = 1
```

If you expect 2 entries per day, then this will arbitrarily pick one. To get both entries for a day, use DENSE_RANK instead

As for normalised or not, it depends if you want to:

- maintain status in 2 places
- preserve status history
- ...

As it stands, you preserve status history. If you want latest status in the parent table too (which is denormalisation) you'd need a trigger to maintain "status" in the parent. or drop this status history table.

Tags: [tsql](#) ([Prev Q](#)), [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#))

Database

[Skip to questions](#),

Wiki by user [hlovdal](#) 

From [Wikipedia](#) :

A database is an organized collection of data. It is the collection of tables, queries, reports, views and other objects. The data is typically organized to model aspects of reality in a way that supports processes requiring information, such as modelling the availability of rooms in hotels in a way that supports finding a hotel with vacancies.

A large proportion of websites and applications rely on databases. They are a crucial component of telecommunications systems, banking systems, video games, and just about any other software system or electronic device that maintains some amount of persistent information. In addition to persistence, database systems provide a number of other properties that make them exceptionally useful and convenient: reliability, efficiency, scalability, concurrency control, data abstraction, and high-level query languages.

Databases are so ubiquitous and important that computer science graduates frequently cite their database class as the one most useful to them in their industry or graduate-school careers.² 

The term *database* should not be confused with [Database Management System](#) (DBMS). A [dbms](#)  is the system software used to create and manage databases and provide users and applications with access to the database(s). A database is to a DBMS as a document is to a word processor.

Database Languages

Database languages are special-purpose languages, which do one or more of the following:

- [Data definition language](#)  : defines data types and the relationships among them
- [Data manipulation language](#)  : performs tasks such as inserting, updating, or deleting data occurrences. The most widely used [dml](#)  is [sql](#).
- [Query language](#)  : allows searching for information and computing derived information

A few notable DBMSs:

- [4d-database](#)  - [4D Website](#) 
- [mysql](#) - [MySQL Website](#) 
- [postgresql](#) - [PostgreSQL Website](#) 
- [oracle](#) - [Oracle Database Website](#) 
- [sqlite](#) - [SQLite Website](#) 

- [sqlserver](#) - [SQL Server Website](#)
- [sybase](#) - [Sybase Database Management Website](#)
- [mongodb](#) - [mongoDB Website](#)
- [mariadb](#) - [MariaDB Website](#)
- [derby](#) - [Apache Derby Website](#)
- [couchdb](#) - [Apache CouchDB Website](#)

Popular Database tools

- [Microsoft SQL Server Management Studio](#)
- [Oracle SQL Developer](#)
- [SQuirreL SQL](#)
- [Toad](#)
- [phpMyAdmin](#)

Some useful references:

- [Database design](#)
- [Managing a database](#)
- [Tuning of a database](#)
- [Database security](#)
- [Database connections](#)
- [Database normalization](#)

Free online database courses:

- [Introduction to Databases](#)
-

Questions

[Q: What is the best way to avoid SQL injection attacks?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

I've provided a solution for Python... please flesh this out with examples for other languages.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [mark-harrison](#) 

[Answer](#)  by [wedge](#) 

Never trust user input, never trust client input at all, client side sanitation efforts are worthless, it doesn't take much sophistication to access your web app directly. **Always perform sanitation server-side**, don't even bother with client-side sanitation, it's a waste of effort. There are a couple techniques that are useful for avoiding SQL injection attacks, I'll go from least favored to most favored.

- **3: Custom-Written Value Sanitation.** Avoid writing your own sanitation routines as much as possible except when it's absolutely the only option remaining (which is very unlikely in any modern language). Input sanitation is a hard problem, and the costs of getting it wrong are huge. It's best to leave that job to someone else. When you are forced to rely on this method use white-listing rather than black-listing to sanitize input.
- **2: Framework / Library Based Value Sanitation.** Leave the sanitation routines to the domain experts. Sanitation routines in 1st party frameworks and libraries are typically created by the same folks that wrote the DB or the SQL API. They have a much higher chance of knowing, and properly handling, all of the edge cases than you do. An example of this technique would be using php's mysql_escape_string() function to sanitize values that will be inserted into strings that serve as dynamic SQL statements.
- **1: Parameter Binding.** (aka prepared statements) Instead of constructing a SQL statement as a raw string which includes user data as in-place literal values, create a SQL statement with tokens where the user data would be. Then bind the user supplied data to the appropriate parameters. The key here is that this binds the provided data to a specific type and a specific use and eliminates any opportunity to change the logic of the SQL statement. This can be used in conjunction with library based input sanitation as well.

Here's an example in C# (Java and Python have similar functionalities, see some of the other answers):

```
SqlCommand userInfoQuery = new SqlCommand(  
    "SELECT id, name, email FROM users WHERE id = @UserName",  
    someSqlConnection);
```

```
SqlParameter userNameParam = userInfoQuery.Parameters.Add("@UserName",
    SqlDbType.VarChar, 25 /* max length of field */);

// userName is some string valued user input variable
userNameParam.Value = userName;
```

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to select the nth row in a SQL database table?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

I'm interested in learning some (ideally) database agnostic ways of selecting the *n*th row from a database table. It would also be interesting to see how this can be achieved using the native functionality of the following databases:

- SQL Server
- MySQL
- PostgreSQL
- SQLite
- Oracle

I am currently doing something like the following in SQL Server 2005, but I'd be interested in seeing other's more agnostic approaches:

```
WITH Ordered AS (
SELECT ROW_NUMBER() OVER (ORDER BY OrderID) AS RowNumber, OrderID, OrderDate
FROM Orders)
SELECT *
FROM Ordered
WHERE RowNumber = 1000000
```

Credit for the above SQL: [Firoz Ansari's Weblog](#) 

Update: See [Troels Arvin's answer](#) regarding the SQL standard. *Troels, have you got any links we can cite?*

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [charles-roper](#) 

[Answer](#)  by [henrik-gustafsson](#) 

There are ways of doing this in optional parts of the standard, but a lot of databases support their own way of doing it.

A really good site that talks about this and other things is <http://troels.arvin.dk/db/rdbms/#select-limit> .

Basically, PostgreSQL and MySQL supports the non-standard:

```
SELECT...
LIMIT y OFFSET x
```

Oracle, DB2 and MSSQL supports the standard windowing functions:

```
SELECT * FROM (
```

```
SELECT
    ROW_NUMBER() OVER (ORDER BY key ASC) AS rownumber,
    columns
    FROM tablename
) AS foo
WHERE rownumber <= n
```

(which I just copied from the site linked above since I never use those DBs)

Update: As of PostgreSQL 8.4 the standard windowing functions are supported, so expect the second example to work for PostgreSQL as well.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL Insert into ... values \(SELECT ... FROM ... \)](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

I am trying to insert into a table using the input from another table. Although this is entirely feasible for many database engines, I always seem to struggle to remember the correct syntax for the SQL engine of the day ([MySQL](#) , [Oracle](#) , [SQL Server](#) , [Informix](#) , and [DB2](#) ).

Is there a silver-bullet syntax coming from an SQL standard (for example, [SQL-92](#) ) that would allow me to insert the values without worrying about the underlying database?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [shadow_x99](#) 

[Answer](#)  by [shadow_x99](#) 

Try:

```
INSERT INTO table1 ( column1 )
SELECT col1
FROM table2
```

This works for:

```
Oracle
MS SQL Server
MySQL
Postgres
SQLite v3
TeraData
DB2
SyBase
```

[Answer](#)  by [travis](#) 

@[Shadow_x99](#) : That should work fine, and you can also have multiple columns and other data as well:

```
INSERT INTO table1 ( column1, column2, someInt, someVarChar )
SELECT table2.column1, table2.column2, 8, 'some string etc.'
FROM table2
WHERE table2.ID = 7;
```

Edit: I should mention that I've only used this syntax with Access, SQL

2000/2005/Express, MySql, and PostGres, so those should be covered. A commenter has pointed out that it'll work with SqlLite3.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

Q: Good reasons NOT to use a relational database?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [nosql](#) ([Next Q](#))

Can you please point to alternative data storage tools and give good reasons to use them instead of good-old relational databases? In my opinion, most applications rarely use the full power of SQL—it would be interesting to see how to build an SQL-free application.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [nosql](#) ([Next Q](#))

User: [caustic](#) 

[Answer](#)  by [matt-sheppard](#) 

Plain text files in a filesystem

- Very simple to create and edit
 - Easy for users to manipulate with simple tools (i.e. text editors, grep etc)
 - Efficient storage of binary documents
-

XML or JSON files on disk

- As above, but with a bit more ability to validate the structure.
-

Spreadsheet / CSV file

- Very easy model for business users to understand
-

Subversion (or similar disk based version control system)

- Very good support for versioning of data
-

[Berkeley DB](#)  (Basically, a disk based hashtable)

- Very simple conceptually (just un-typed key/value)
 - Quite fast
 - No administration overhead
 - Supports transactions I believe
-

[Amazon's Simple DB](#) 

- Much like Berkeley DB I believe, but hosted
-

[Google's App Engine Datastore](#)

- Hosted and highly scalable
 - Per document key-value storage (i.e. flexible data model)
-

[CouchDB](#)

- Document focus
 - Simple storage of semi-structured / document based data
-

Native language collections (stored in memory or serialised on disk)

- Very tight language integration
-

Custom (hand-written) storage engine

- Potentially very high performance in required uses cases
-

I can't claim to know anything much about them, but you might also like to look into [object database systems](#) .

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [nosql](#) ([Next Q](#))

[Q: Best way to do multi-row insert in Oracle?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#))

I'm looking for a good way to perform multi-row inserts into an Oracle 9 database. The following works in MySQL but doesn't seem to be supported in Oracle.

[Skip code block](#)

```
INSERT INTO TMP_DIM_EXCH_RT
(EXCH_WH_KEY,
EXCH_NAT_KEY,
EXCH_DATE, EXCH_RATE,
FROM_CURCY_CD,
TO_CURCY_CD,
EXCH_EFF_DATE,
EXCH_EFF_END_DATE,
EXCH_LAST_UPDATED_DATE)
VALUES
(1, 1, '28-AUG-2008', 109.49, 'USD', 'JPY', '28-AUG-2008', '28-AUG-2008', '28-AUG-2008'),
(2, 1, '28-AUG-2008', .54, 'USD', 'GBP', '28-AUG-2008', '28-AUG-2008', '28-AUG-2008'),
(3, 1, '28-AUG-2008', 1.05, 'USD', 'CAD', '28-AUG-2008', '28-AUG-2008', '28-AUG-2008'),
(4, 1, '28-AUG-2008', .68, 'USD', 'EUR', '28-AUG-2008', '28-AUG-2008', '28-AUG-2008'),
(5, 1, '28-AUG-2008', 1.16, 'USD', 'AUD', '28-AUG-2008', '28-AUG-2008', '28-AUG-2008'),
(6, 1, '28-AUG-2008', 7.81, 'USD', 'HKD', '28-AUG-2008', '28-AUG-2008', '28-AUG-2008');
```

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [jamey](#)

[Answer](#) by [espo](#)

This works in Oracle:

```
insert into pager (PAG_ID,PAG_PARENT,PAG_NAME,PAG_ACTIVE)
  select 8000,0,'Multi 8000',1 from dual
union all select 8001,0,'Multi 8001',1 from dual
```

The thing to remember here is to use the `from dual` statement.

([source](#))

[Answer](#) by [myto](#)

In Oracle, to insert multiple rows into table t with columns col1, col2 and col3 you can use the following syntax:

```
INSERT ALL
  INTO t (col1, col2, col3) VALUES ('val1_1', 'val1_2', 'val1_3')
  INTO t (col1, col2, col3) VALUES ('val2_1', 'val2_2', 'val2_3')
  INTO t (col1, col2, col3) VALUES ('val3_1', 'val3_2', 'val3_3')
  .
  .
SELECT 1 FROM DUAL;
```

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: How do I reset a sequence in Oracle?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#))

In [PostgreSQL](#), I can do something like this:

```
ALTER SEQUENCE serial RESTART WITH 0;
```

Is there an Oracle equivalent?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [mauli](#)

[Answer](#) by [doug-porter](#)

Here is a good procedure for resetting any sequence to 0 from Oracle guru [Tom Kyte](#). Great discussion on the pros and cons in the links below too.

[Skip code block](#)

```
tkyte@TKYTE901.US.ORACLE.COM>
create or replace
procedure reset_seq( p_seq_name in varchar2 )
is
  l_val number;
begin
  execute immediate
    'select '' || p_seq_name || '.nextval from dual' INTO l_val;
  execute immediate
    'alter sequence ' || p_seq_name || ' increment by -' || l_val ||
```

```

        ' minvalue 0';

execute immediate
'select '' || p_seq_name || '.nextval from dual' INTO l_val;

execute immediate
'alter sequence ' || p_seq_name || ' increment by 1 minvalue 0';
end;
/

```

From this page: [Dynamic SQL to reset sequence value](#)

Another good discussion is also here: [How to reset sequences?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Are Stored Procedures more efficient, in general, than inline statements on modern RDBMS's?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

Conventional wisdom states that stored procedures are always faster. So, since they're always faster, use them **ALL THE TIME**.

I am pretty sure this is grounded in some historical context where this was once the case. Now, I'm not advocating that Stored Procs are not needed, but I want to know in what cases stored procs are necessary in modern databases such as MySql, SqlServer, Oracle, or . Is it overkill to have ALL access through stored procs?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [casademora](#)

[Answer](#) by [matt-rogish](#)

NOTE that this is a general look at stored procedures not regulated to a specific DBMS. Some DBMS (and even, different versions of the same DBMS!) may operate contrary to this, so you'll want to double-check with your target DBMS before assuming all of this still holds.

I've been a Sybase ASE, MySQL, and SQL Server DBA on-and off since for almost a decade (along with application development in C, PHP, PL/SQL, C#.NET, and Ruby). So, I have no particular axe to grind in this (sometimes) holy war.

The historical performance benefit of stored procs have generally been from the following (in no particular order):

- Pre-parsed SQL
- Pre-generated query execution plan
- Reduced network latency
- Potential cache benefits

Pre-parsed SQL — similar benefits to compiled vs. interpreted code, except on a very

micro level.

Still an advantage? Not very noticeable at all on the modern CPU, but if you are sending a single SQL statement that is VERY large eleventy-billion times a second, the parsing overhead can add up.

Pre-generated query execution plan. If you have many JOINs the permutations can grow quite unmanageable (modern optimizers have limits and cut-offs for performance reasons). It is not unknown for very complicated SQL to have distinct, measurable (I've seen a complicated query take 10+ seconds just to generate a plan, before we tweaked the DBMS) latencies due to the optimizer trying to figure out the "near best" execution plan. Stored procedures will, generally, store this in memory so you can avoid this overhead.

Still an advantage? Most DBMS' (the latest editions) will cache the query plans for INDIVIDUAL SQL statements, greatly reducing the performance differential between stored procs and ad hoc SQL. There are some caveats and cases in which this isn't the case, so you'll need to test on your target DBMS.

Also, more and more DBMS allow you to provide optimizer path plans (abstract query plans) to significantly reduce optimization time (for both ad hoc and stored procedure SQL!!).

WARNING Cached query plans are not a performance panacea. Occasionally the query plan that is generated is sub-optimal. For example, if you send `SELECT * FROM table WHERE id BETWEEN 1 AND 99999999`, the DBMS may select a full-table scan instead of an index scan because you're grabbing every row in the table (so sayeth the statistics). If this is the cached version, then you can get poor performance when you later send `SELECT * FROM table WHERE id BETWEEN 1 AND 2`. The reasoning behind this is outside the scope of this posting, but for further reading see:

<http://www.microsoft.com/technet/prodtechnol/sql/2005/frcqupln.mspx> and
<http://msdn.microsoft.com/en-us/library/ms181055.aspx> and <http://www.simple-talk.com/sql/performance/execution-plan-basics/>

"In summary, they determined that supplying anything other than the common values when a compile or recompile was performed resulted in the optimizer compiling and caching the query plan for that particular value. Yet, when that query plan was reused for subsequent executions of the same query for the common values ('M', 'R', or 'T'), it resulted in sub-optimal performance. This sub-optimal performance problem existed until the query was recompiled. At that point, based on the @P1 parameter value supplied, the query might or might not have a performance problem."

Reduced network latency A) If you are running the same SQL over and over — and the SQL adds up to many KB of code — replacing that with a simple "exec foobar" can really add up. B) Stored procs can be used to move procedural code into the DBMS. This saves shuffling large amounts of data off to the client only to have it send a trickle of info back (or none at all!). Analogous to doing a JOIN in the DBMS vs. in your code (everyone's favorite WTF!)

Still an advantage? A) Modern 1Gb (and 10Gb and up!) Ethernet really make this negligible. B) Depends on how saturated your network is — why shove several megabytes

of data back and forth for no good reason?

Potential cache benefits Performing server-side transforms of data can potentially be faster if you have sufficient memory on the DBMS and the data you need is in memory of the server.

Still an advantage? Unless your app has shared memory access to DBMS data, the edge will always be to stored procs.

Of course, no discussion of Stored Procedure optimization would be complete without a discussion of parameterized and ad hoc SQL.

Parameterized / Prepared SQL

Kind of a cross between stored procedures and ad hoc SQL, they are embedded SQL statements in a host language that uses “parameters” for query values, e.g.:

```
SELECT .. FROM yourtable WHERE foo = ? AND bar = ?
```

These provide a more generalized version of a query that modern-day optimizers can use to cache (and re-use) the query execution plan, resulting in much of the performance benefit of stored procedures.

Ad Hoc SQL Just open a console window to your DBMS and type in a SQL statement. In the past, these were the “worst” performers (on average) since the DBMS had no way of pre-optimizing the queries as in the parameterized/stored proc method.

Still a disadvantage? Not necessarily. Most DBMS have the ability to “abstract” ad hoc SQL into parameterized versions — thus more or less negating the difference between the two. Some do this implicitly or must be enabled with a command setting (SQL server: <http://msdn.microsoft.com/en-us/library/ms175037.aspx>, Oracle: http://www.praetoriate.com/oracle_tips_cursor_sharing.htm).

Lessons learned? Moore’s law continues to march on and DBMS optimizers, with every release, get more sophisticated. Sure, you can place every single silly teeny SQL statement inside a stored proc, but just know that the programmers working on optimizers are very smart and are continually looking for ways to improve performance. Eventually (if it’s not here already) ad hoc SQL performance will become indistinguishable (on average!) from stored procedure performance, so any sort of *massive* stored procedure use ** solely for “performance reasons”** sure sounds like premature optimization to me.

Anyway, I think if you avoid the edge cases and have fairly vanilla SQL, you won’t notice a difference between ad hoc and stored procedures.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

Q: Which is faster/best? `SELECT *` or `SELECT column1, column2, column3, etc`

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

I’ve heard that `SELECT *` is generally bad practice to use when writing SQL commands

because it is more efficient to `SELECT` columns you specifically need.

If I need to `SELECT` every column in a table, should I use `SELECT *` or `SELECT column1, column2, column3, etc.`.

Does the efficiency really matter in this case? I'd think `SELECT *` would be more optimal internally if you really need all of the data, but I say this with no real understanding of databases.

I'm curious to know what the best practice is in this case.

UPDATE: I probably should specify that the only situation where I would really *want* to do a `SELECT *` is when I'm selecting data from one table where I know all columns will always need to be retrieved, even when new columns are added.

Given the responses I've seen however, this still seems like a bad idea and `SELECT *` should never be used for a lot more technical reasons that I ever thought about.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [dan-herbert](#) 

[Answer](#)  by [jon-galloway](#) 

One reason that selecting specific columns is better is that it raises the probability that SQL Server can access the data from indexes rather than querying the table data. Here's a post I wrote about it: <http://weblogs.asp.net/jgalloway/archive/2007/07/18/the-real-reason-select-queries-are-bad-index-coverage.aspx> 

It's also less fragile to change, since any code that consumes the data will be getting the same data structure regardless of changes you make to the table schema in the future.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

Q: How to list the tables in an SQLite database file that was opened with ATTACH?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [sqlite](#) ([Next Q](#))

What SQL can be used to list the tables, and the rows within those tables in a SQLite database file - once I have attached it with the `ATTACH` command on the SQLite 3 command line tool?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [sqlite](#) ([Next Q](#))

User: [izb](#) 

[Answer](#)  by [anthony-williams](#) 

The `.tables`, and `.schema` "helper" functions don't look into ATTACHED databases: they just query the `SQLITE_MASTER` table for the "main" database. Consequently, if you used

```
ATTACH some_file.db AS my_db;
```

then you need to do

```
SELECT name FROM my_db.sqlite_master WHERE type='table';
```

Note that temporary tables don't show up with `.tables` either: you have to list `sqlite_temp_master` for that:

```
SELECT name FROM sqlite_temp_master WHERE type='table';
```

[Answer](#) by [mark-janssen](#)

There are a few steps to see the tables in an SQLite database:

1. List the tables in your database:

```
.tables
```

2. List how the table looks:

```
.schema tablename
```

3. Print the entire table:

```
SELECT * FROM tablename;
```

4. List all of the available SQLite prompt commands:

```
.help
```

[Answer](#) by [lasse-v.-karlsen](#)

It appears you need to go through the `sqlite_master` table, like this:

```
SELECT * FROM dbname.sqlite_master WHERE type='table';
```

And then manually go through each table with a `SELECT` or similar to look at the rows.

The `.DUMP` and `.SCHEMA` commands doesn't appear to see the database at all.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [sqlite](#) ([Next Q](#))

Q: What's the difference between TRUNCATE and DELETE in SQL



Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

I wrote up an answer to this question by mistake in response to a question about the difference between DROP and TRUNCATE, but I thought that it's a shame not to share so I'll post my own answer to my own question ... is that even ethical? :)

Edit: If your answer is platform specific can you please indicate that.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [david-aldrige](#)

[Answer](#) by [david-aldrige](#)

Here's a list of differences. I've highlighted Oracle-specific features, and hopefully the community can add in other vendors' specific difference also. Differences that are common to most vendors can go directly below the headings, with differences highlighted below.

General Overview

If you want to quickly delete all of the rows from a table, and you're really sure that you want to do it, and you do not have foreign keys against the tables, then a TRUNCATE is probably going to be faster than a DELETE.

Various system-specific issues have to be considered, as detailed below.

Statement type

Delete is DML, Truncate is DDL

Commit and Rollback

Variable by vendor

SQL*Server

Truncate can be rolled back.

PostgreSQL

Truncate can be rolled back.

Oracle

Because a TRUNCATE is DDL it involves two commits, one before and one after the statement execution. Truncate can therefore not be rolled back, and a failure in the

truncate process will have issued a commit anyway.

However, see Flashback below.

Space reclamation

Delete does not recover space, Truncate recovers space

Oracle

If you use the REUSE STORAGE clause then the data segments are not de-allocated, which can be marginally more efficient if the table is to be reloaded with data. The high water mark is reset.

Row scope

Delete can remove only some rows. Truncate removes all rows.

Oracle

When a table is partitioned, the individual partitions can be truncated in isolation, thus a partial removal of all the table's data is possible.

Object types

Delete can be applied to tables and tables inside a cluster. Truncate applies only to tables or the entire cluster. (May be Oracle specific)

Data Object Identity

Oracle

Delete does not affect the data object id, but truncate assigns a new data object id *unless* there has never been an insert against the table since its creation Even a single insert that is rolled back will cause a new data object id to be assigned upon truncation.

Flashback (Oracle)

Flashback works across deletes, but a truncate prevents flashback to states prior to the operation.

However, from 11gR2 the FLASHBACK ARCHIVE feature allows this, except in Express Edition

[Use of FLASHBACK in Oracle](#)

http://docs.oracle.com/cd/E11882_01/appdev.112/e41502/adfns_flashback.htm#ADFNS63

Privileges

Variable

Oracle

Delete can be granted on a table to another user or role, but truncate cannot be without using a DROP ANY TABLE grant.

Redo/Undo

Delete generates a small amount of redo and a large amount of undo. Truncate generates a negligible amount of each.

Indexes

Oracle

A truncate operation renders unusable indexes usable again. Delete does not.

Foreign Keys

A truncate cannot be applied when an enabled foreign key references the table. Treatment with delete depends on the configuration of the foreign keys.

Table Locking

Oracle

Truncate requires an exclusive table lock, delete requires a shared table lock. Hence disabling table locks is a way of preventing truncate operations on a table.

Triggers

DML triggers do not fire on a truncate.

Oracle

DDL triggers are available.

Remote Execution

Oracle

Truncate cannot be issued over a database link.

Identity Columns

SQL*Server

Truncate resets the sequence for IDENTITY column types, delete does not.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: Storing money in a decimal column - what precision and scale?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

I'm using a decimal column to store money values on a database, and today I was wondering what precision and scale to use.

Since supposedly char columns of a fixed width are more efficient, I was thinking the same could be true for decimal columns. Is it?

And what precision and scale should I use? I was thinking precision 24/8. Is that overkill, not enough or ok?

This is what I've decided to do:

- Store the conversion rates (when applicable) in the transaction table itself, as a float
- Store the currency in the account table
- The transaction amount will be a DECIMAL(19, 4)
- All calculations using a conversion rate will be handled by my application so I keep control of rounding issues

I don't think a float for the conversion rate is an issue, since it's mostly for reference, and I'll be casting it to a decimal anyway.

Thank you all for your valuable input.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [ivan](#) 

[Answer](#)  by [onedaywhen](#) 

If you are looking for a one-size-fits-all, I'd suggest DECIMAL(19, 4) is a popular choice (a quick Google bears this out). I think this originates from the old VBA/Access/Jet Currency data type, being the first fixed point decimal type in the language; Decimal only came in 'version 1.0' style (i.e. not fully implemented) in VB6/VBA6/Jet 4.0.

The rule of thumb for *storage* of fixed point decimal values is to store at least one more decimal place than you actually require to allow for rounding. One of the reasons for mapping the old Currency type in the front end to DECIMAL(19, 4) type in the back end was that Currency exhibited bankers' rounding by nature, whereas DECIMAL(p, s) rounded by truncation.

An extra decimal place in storage for `DECIMAL` allows a custom rounding algorithm to be implemented rather than taking the vendor's default (and bankers' rounding is alarming, to say the least, for a designer expecting all values ending in .5 to round away from zero).

Yes, `DECIMAL(24, 8)` sounds like overkill to me. Most currencies are quoted to four or five decimal places. I know of situations where a decimal scale of 8 (or more) *is* required but this is where a 'normal' monetary amount (say four decimal places) has been pro rata'd, implying the decimal precision should be reduced accordingly (also consider a floating point type in such circumstances). And no one has that much money nowadays to require a decimal precision of 24 :)

However, rather than a one-size-fits-all approach, some research may be in order. Ask your designer or domain expert about accounting rules which may be applicable: GAAP, EU, etc. I vaguely recall some EU intra-state transfers with explicit rules for rounding to five decimal places, therefore using `DECIMAL(p, 6)` for storage. Accountants generally seem to favour four decimal places.

PS Avoid SQL Server's `MONEY` data type because it has serious issues with accuracy when rounding, among other considerations such as portability etc. See [Aaron Bertrand's blog](#) .

Microsoft and language designers chose banker's rounding because hardware designers chose it [citation?]. It is enshrined in the Institute of Electrical and Electronics Engineers (IEEE) standards, for example. And hardware designers chose it because mathematicians prefer it. See [Wikipedia](#) ; to paraphrase: The 1906 edition of Probability and Theory of Errors called this 'the computer's rule' ("computers" meaning humans who perform computations).

[Answer](#)  by [marcus-downing](#) 

We recently implemented a system that needs to handle values in multiple currencies and convert between them, and figured out a few things the hard way.

NEVER USE FLOATING POINT NUMBERS FOR MONEY

Floating point arithmetic introduces inaccuracies that may not be noticed until they've screwed something up. All values should be stored as either integers or fixed-decimal types, and if you choose to use a fixed-decimal type then make sure you understand exactly what that type does under the hood (ie, does it internally use an integer or floating point type).

When you do need to do calculations or conversions:

1. Convert values to floating point
2. Calculate new value
3. Round the number and convert it back to an integer

When converting a floating point number back to an integer in step 3, don't just cast it - use a math function to round it first. This will usually be `round`, though in special cases it could be `floor` or `ceil`. Know the difference and choose carefully.

Store the type of a number alongside the value

This may not be as important for you if you're only handling one currency, but it was important for us in handling multiple currencies. We used the 3-character code for a currency, such as USD, GBP, JPY, EUR, etc.

Depending on the situation, it may also be helpful to store:

- Whether the number is before or after tax (and what the tax rate was)
- Whether the number is the result of a conversion (and what it was converted from)

Know the accuracy bounds of the numbers you're dealing with

For real values, you want to be as precise as the smallest unit of the currency. This means you have no values smaller than a cent, a penny, a yen, a fen, etc. Don't store values with higher accuracy than that for no reason.

Internally, you may choose to deal with smaller values, in which case that's a *different type of currency value*. Make sure your code knows which is which and doesn't get them mixed up. Avoid using floating point values even here.

Adding all those rules together, we decided on the following rules. In running code, currencies are stored using an integer for the smallest unit.

[Skip code block](#)

```
class Currency {  
    String code;          // eg "USD"  
    int value;           // eg 2500  
    boolean converted;  
}  
  
class Price {  
    Currency grossValue;  
    Currency netValue;  
    Tax taxRate;  
}
```

In the database, the values are stored as a string in the following format:

```
USD:2500
```

That stores the value of \$25.00. We were able to do that only because the code that deals with currencies doesn't need to be within the database layer itself, so all values can be converted into memory first. Other situations will no doubt lend themselves to other solutions.

And in case I didn't make it clear earlier, **don't use float!**

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

[Q: The Next-gen Databases](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [nosql](#) ([Prev Q](#)) ([Next Q](#))

I'm learning traditional Relational Databases (with [PostgreSQL](#) ) and doing some

research I've come across some new types of databases. [CouchDB](#), [Drizzle](#), and [Scalaris](#) to name a few, what is going to be the next database technologies to deal with?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [nosql](#) ([Prev Q](#)) ([Next Q](#))

User: [randin](#)

[Answer](#) by [bill-karwin](#)

I would say next-gen *database*, not next-gen SQL.

SQL is a language for querying and manipulating relational databases. SQL is dictated by an international standard. While the standard is revised, it seems to always work within the relational database paradigm.

Here are a few new data storage technologies that are getting attention currently:

- [CouchDB](#) is a non-relational database. They call it a document-oriented database.
- [Amazon SimpleDB](#) is also a non-relational database accessed in a distributed manner through a web service. Amazon also has a distributed key-value store called **Dynamo**, which powers some of its S3 services.
- [Dynomite](#) and [Kai](#) are open source solutions inspired by Amazon Dynamo.
- [BigTable](#) is a proprietary data storage solution used by Google, and implemented using their Google File System technology. Google's MapReduce framework uses BigTable.
- [Hadoop](#) is an open-source technology inspired by Google's MapReduce, and serving a similar need, to distribute the work of very large scale data stores.
- [Scalaris](#) is a distributed transactional key/value store. Also not relational, and does not use SQL. It's a research project from the Zuse Institute in Berlin, Germany.
- [RDF](#) is a standard for storing semantic data, in which data and metadata are interchangeable. It has its own query language SPARQL, which resembles SQL superficially, but is actually totally different.
- [Vertica](#) is a highly scalable column-oriented analytic database designed for distributed (grid) architecture. It does claim to be relational and SQL-compliant. It can be used through Amazon's Elastic Compute Cloud.
- [Greenplum](#) is a high-scale data warehousing DBMS, which implements both MapReduce and SQL.
- [XML](#) isn't a DBMS at all, it's an interchange format. But some DBMS products work with data in XML format.
- [ODBMS](#), or Object Databases, are for managing complex data. There don't seem to be any dominant ODBMS products in the mainstream, perhaps because of lack of standardization. Standard SQL is gradually gaining some OO features (e.g. extensible data types and tables).
- [Drizzle](#) is a relational database, drawing a lot of its code from MySQL. It includes various architectural changes designed to manage data in a scalable "cloud computing" system architecture. Presumably it will continue to use standard SQL with some MySQL enhancements.
- [Cassandra](#) is a highly scalable, eventually consistent, distributed, structured key-value store, developed at Facebook by one of the authors of Amazon Dynamo, and

contributed to the Apache project.

- [Project Voldemort](#) is a non-relational, distributed, key-value storage system. It is used at LinkedIn.com
- [Berkeley DB](#) deserves some mention too. It's not "next-gen" because it dates back to the early 1990's. It's a popular key-value store that is easy to embed in a variety of applications. The technology is currently owned by Oracle Corp.

Also see this nice article by Richard Jones: "[Anti-RDBMS: A list of distributed key-value stores](#)." He goes into more detail describing some of these technologies.

Relational databases have weaknesses, to be sure. People have been arguing that they don't handle all data modeling requirements since the day it was first introduced.

Year after year, researchers come up with new ways of managing data to satisfy special requirements: either requirements to handle data relationships that don't fit into the relational model, or else requirements of high-scale volume or speed that demand data processing be done on distributed collections of servers, instead of central database servers.

Even though these advanced technologies do great things to solve the specialized problem they were designed for, relational databases are still a good general-purpose solution for most business needs. SQL isn't going away.

I've written an article in php|Architect magazine about the innovation of non-relational databases, and data modeling in relational vs. non-relational databases.

<http://www.phparch.com/magazine/2010-2/september/>

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [nosql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I get column names from a table in Oracle?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#))

I need to query the database to get the *column names*, not to be confused with data in the table. For example, if I have a table named EVENT_LOG that contains eventID, eventType, eventDesc, and eventTime, then I would want to retrieve those field names from the query and nothing else.

I found how to do this in:

- [Microsoft SQL Server](#)
- [MySQL](#)
- [PostgreSQL](#)

But I need to know: **how can this be done in Oracle?**

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [paxenos](#)

[Answer](#) by [bareta](#)

You can query the USER_TAB_COLUMNS table for table column metadata.

```
SELECT table_name, column_name, data_type, data_length
FROM USER_TAB_COLUMNS
WHERE table_name = 'MYTABLE'
```

[Answer](#) by [eppz](#)

In SQL Server...

```
SELECT [name] AS [Column Name]
FROM syscolumns
WHERE id = (SELECT id FROM sysobjects WHERE type = 'V' AND [Name] = 'Your table name')
```

Type = 'V' for views Type = 'U' for tables

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Strings as Primary Keys in SQL Database](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

I am not very familiar with databases and the theories behind how they work. Is it any slower from a performance standpoint (inserting/updating/querying) to use Strings for Primary Keys than integers?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [dubdubdubdot](#)

[Answer](#) by [kevin](#)

Technically yes, but if a string makes sense to be the primary key then you should probably use it. This all depends on the size of the table you're making it for and the size of the string that is going to be the primary key (longer strings == harder to compare). I wouldn't necessarily use a string for a table that has millions of rows, but the amount of performance slowdown you'll get by using a string on smaller tables will be minuscule to the headaches that you can have by having an integer that doesn't mean anything in relation to the data.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

[Q: Entity Attribute Value Database vs. strict Relational Model](#)

Ecommerce question

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

It is safe to say that the [EAV/CR](#)  database model is bad. That said,

Question: **What database model, technique, or pattern should be used to deal with “classes” of attributes describing e-commerce products which can be changed at run time?**

In a good E-commerce database, you will store classes of options (like TV resolution then have a resolution for each TV, but the next product may not be a TV and not have “TV resolution”). How do you store them, search efficiently, and allow your users to setup product types with variable fields describing their products? If the search engine finds that customers typically search for TVs based on console depth, you could add console depth to your fields, then add a single depth for each tv product type at run time.

There is a nice common feature among good e-commerce apps where they show a set of products, then have “drill down” side menus where you can see “TV Resolution” as a header, and the top five most common TV Resolutions for the found set. You click one and it only shows TVs of that resolution, allowing you to further drill down by selecting other categories on the side menu. These options would be the dynamic product attributes added at run time.

Further discussion:

So long story short, **are there any links out on the Internet or model descriptions that could “academically” fix the following setup?** I thank Noel Kennedy for suggesting a category table, but the need may be greater than that. I describe it a different way below, trying to highlight the significance. I may need a viewpoint correction to solve the problem, or I may need to go deeper in to the EAV/CR.

Love the positive response to the EAV/CR model. My fellow developers all say what Jeffrey Kemp touched on below: “new entities must be modeled and designed by a professional” (taken out of context, read his response below). The problem is:

- entities add and remove attributes weekly
(search keywords dictate future attributes)
- new entities arrive weekly
(products are assembled from parts)
- old entities go away weekly
(archived, less popular, seasonal)

The customer wants to add attributes to the products for two reasons:

- department / keyword search / comparison chart between like products
- consumer product configuration before checkout

The attributes must have significance, not just a keyword search. If they want to compare all cakes that have a “whipped cream frosting”, they can click cakes, click birthday theme, click whipped cream frosting, then check all cakes that are interesting knowing they all have whipped cream frosting. This is not specific to cakes, just an example.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [dr.-zim](#) 

[Answer](#)  by [jeffrey-kemp](#) 

There's a few general pros and cons I can think of, there are situations where one is better than the other:

Option 1, EAV Model:

- Pro: less time to design and develop a simple application
- Pro: new entities easy to add (might even be added by users?)
- Pro: “generic” interface components
- Con: complex code required to validate simple data types
- Con: much more complex SQL for simple reports
- Con: complex reports can become almost impossible
- Con: poor performance for large data sets

Option 2, Modelling each entity separately:

- Con: more time required to gather requirements and design
- Con: new entities must be modelled and designed by a professional
- Con: custom interface components for each entity
- Pro: data type constraints and validation simple to implement
- Pro: SQL is easy to write, easy to understand and debug
- Pro: even the most complex reports are relatively simple
- Pro: best performance for large data sets

Option 3, Combination (model entities “properly”, but add “extensions” for custom attributes for some/all entities)

- Pro/Con: more time required to gather requirements and design than option 1 but perhaps not as much as option 2 *
- Con: new entities must be modelled and designed by a professional
- Pro: new attributes might be easily added later on
- Con: complex code required to validate simple data types (for the custom attributes)
- Con: custom interface components still required, but generic interface components may be possible for the custom attributes
- Con: SQL becomes complex as soon as any custom attribute is included in a report
- Con: good performance generally, unless you start need to search by or report by the custom attributes

* I'm not sure if Option 3 would necessarily save any time in the design phase.

Personally I would lean toward option 2, and avoid EAV wherever possible. However, for some scenarios the users need the flexibility that comes with EAV; but this comes with a great cost.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: What is the difference between Join and Union?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

What is the difference between Join and Union? Can I get a sample?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [gold](#) 

[Answer](#)  by [alex-martelli](#) 

UNION puts lines from queries after each other, while JOIN makes a cartesian product and subsets it — completely different operations. Trivial example of UNION:

[Skip code block](#)

```
mysql> SELECT 23 AS bah
      -> UNION
      -> SELECT 45 AS bah;
+-----+
| bah |
+-----+
| 23 |
| 45 |
+-----+
2 rows in set (0.00 sec)
```

similary trivial example of JOIN:

[Skip code block](#)

```
mysql> SELECT * FROM
      -> (SELECT 23 AS bah) AS foo
      -> JOIN
      -> (SELECT 45 AS bah) AS bar
      -> ON (33=33);
+-----+-----+
| bah | bah |
+-----+-----+
| 23 | 45 |
+-----+-----+
1 row in set (0.01 sec)
```

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: What are best practices for multi-language database design?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

What is the best way to create multi-language database? To create localized table for every table is making design and querying complex, in other case to add column for each language is simple but not dynamic, please help me to understand what is the best choose

for enterprise applications

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [arsen-mkrtchyan](#) 

[Answer](#)  by [m4n](#) 

What we do, is to create two tables for each multilingual object.

E.g. the first table contains only language-neutral data (primary key, etc.) and the second table contains one record per language, containing the localized data plus the ISO code of the language.

In some cases we add a DefaultLanguage field, so that we can fall-back to that language if no localized data is available for a specified language.

Example:

[Skip code block](#)

```
Table "Product":  
-----  
ID : int  
<any other language-neutral fields>  
  
Table "ProductTranslations"  
-----  
ID : int      (foreign key referencing the Product)  
Language : varchar (e.g. "en-US", "de-CH")  
IsDefault : bit  
ProductDescription : nvarchar  
<any other localized data>
```

With this approach, you can handle as many languages as needed (without having to add additional fields for each new language).

Update (2014-12-14): please have a look at [this answer](#) , for some additional information about the implementation used to load multilingual data into an application.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

[Q: Postgres and Indexes on Foreign Keys and Primary Keys](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#))

Does Postgres automatically put indexes on Foreign Keys and Primary Keys? How can I tell? Is there a command that will return all indexes on a table?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#))

User: [dubdubdubdot](#) 

[Answer](#)  by [philipp](#) 

PostgreSQL automatically creates indexes on primary keys and unique constraints, but not on the referencing side of foreign key relationships.

When Pg creates an implicit index it will emit a NOTICE-level message that you can see in psql and/or the system logs, so you can see when it happens. Automatically created indexes are visible in \d output for a table, too.

The [documentation on unique indexes](#) says:

PostgreSQL automatically creates an index for each unique constraint and primary key constraint to enforce uniqueness. Thus, it is not necessary to create an index explicitly for primary key columns.

and the documentation on [constraints](#) says:

Since a DELETE of a row from the referenced table or an UPDATE of a referenced column will require a scan of the referencing table for rows matching the old value, it is often a good idea to index the referencing columns. Because this is not always needed, and there are many choices available on how to index, declaration of a foreign key constraint does not automatically create an index on the referencing columns.

Therefore you have to create indexes on foreign-keys yourself if you want them.

Note that if you use primary-foreign-keys, like 2 FK's as a PK in a M-to-N table, you will have an index on the PK and probably don't need to create any extra indexes.

While it's usually a good idea to create an index on (or including) your referencing-side foreign key columns, it isn't required. Each index you add slows DML operations down slightly, so you pay a performance cost on every INSERT, UPDATE or DELETE. If the index is rarely used it may not be worth having.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Max length for client ip address](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

Possible Duplicate:

[Maximum length of the textual representation of an IPv6 address?](#)

What would you recommend as the maximum size for a database column storing client ip addresses? I have it set to 16 right now, but could I get an ip address that is longer than that with IPv6, etc?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [tony-eichelberger](#)

[Answer](#) by [matt-bridges](#)

For IPv4, you *could* get away with storing the 4 raw bytes of the IP address (each of the

numbers between the periods in an IP address are 0-255, i.e., one byte). But then you would have to translate going in and out of the DB and that's messy.

IPv6 addresses are 128 bits (as opposed to 32 bits of IPv4 addresses). They are usually written as 8 groups of 4 hex digits separated by colons:

2001:0db8:85a3:0000:0000:8a2e:0370:7334. 39 characters is appropriate to store IPv6 addresses in this format.

Edit: However, there is a caveat, see @Deepak's answer for details about IPv4-mapped IPv6 addresses. (The correct maximum IPv6 string length is **45 characters**.)

[Answer](#) by [deepak](#)

There's a caveat. For IPv4 mapped IPv6 addresses, the string can be longer than 39 characters. Let me show with an example:

IPv6 (39 bytes) :

```
ABCD:ABCD:ABCD:ABCD:ABCD:ABCD:ABCD:ABCD
```

IPv4-mapped IPv6 (45 bytes) :

```
ABCD:ABCD:ABCD:ABCD:ABCD:ABCD:192.168.158.190
```

The last 32-bits (that correspond to IPv4 address) can need more than 10 characters.

The correct maximum IPv6 string length, therefore, is 45.

This was actually a quiz question in an IPv6 training I attended. (We all answered 39!)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

[Q: What datatype to use when storing latitude and longitude data in SQL databases?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

When storing latitude or longitude data in an ANSI SQL compliant database, what datatype would be most appropriate? Should float be used, or decimal, or ...?

I'm aware that Oracle, MySql, and SQL Server have added some special datatypes specifically for handling geo data, but I'm interested in how you would store the information in a "plain vanilla" SQL database.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [dthrasher](#)

[Answer](#) by [dotjoe](#)

Decimal(9, 6)

If you're not used to precision and scale parameters, here's a format string visual:

###.#####

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: What is the difference between single and double quotes in SQL?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

What is the difference between single quotes and double quotes in SQL?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [vineet](#)

[Answer](#) by [omg-ponies](#)

Single quotes are used to indicate the beginning and end of a string in SQL. Double quotes generally aren't used in SQL, but that can vary from database to database.

Stick to using single quotes.

That's the primary use anyways. You can use single quotes for a column alias - where you want the column name you reference in your application code to be something other than what the column is actually called in the database. For example: PRODUCTS.id would be more readable as product_id, so you use either of the following:

- SELECT PRODUCT.id AS product_id
- SELECT PRODUCT.id 'product_id'

Either works in Oracle, SQL Server, MySQL... But I know some have said that the

TOAD IDE seems to give some grief when using the single quotes approach.

You do have to use single quotes when the column alias includes a space character. IE: “product id”, but it’s not recommended practice for a column alias to be more than one word.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to do an INNER JOIN on multiple columns](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

I’m working on a homework project and I’m supposed to perform a database query which finds flights either by the city name or the airport code, but the `flights` table only contains the airport codes so if I want to search by city I have to join on the `airports` table.

The `airports` table has the following columns: `code`, `city`

The `flights` table has the following columns: `airline`, `flt_no`, `fairport`, `tairport`, `depart`, `arrive`, `fare`

The columns `fairport` and `tairport` are the **from** and **to** airport codes.

The columns `depart` and `arrive` are dates of departure and arrival.

I came up with a query which first joins the flights on the `fairport` column and the `airports.code` column. In order for me to match the `tairport` I have to perform another join on the previous matches from the first join.

```
SELECT airline, flt_no, fairport, tairport, depart, arrive, fare
  FROM (SELECT * FROM flights
        INNER JOIN airports
          ON flights.fairport = airports.code
          WHERE (airports.code = '?' OR airports.city='?')) AS matches
        INNER JOIN airports
          ON matches.tairport = airports.code
          WHERE (airports.code = '?' OR airports.city = '?')
```

My query returns the proper results and it will suffice for the purpose of the homework, but I’m wondering if I can `JOIN` on multiple columns? How would I construct the `WHERE` clause so it matches the departure and the destination city/code?

Below is a “pseudo-query” on what I want to achieve, but I can’t get the syntax correctly and I don’t know how to represent the `airports` table for the departures and the destinations:

```
SELECT * FROM flights
INNER JOIN airports
ON flights.fairport = airports.code AND flights.tairport = airports.code
WHERE (airports.code = 'departureCode' OR airports.city= 'departureCity')
      AND (airports.code = 'destinationCode' OR airports.city = 'destinationCity')
```

Update

I also found [this visual representation of SQL Join statements](#) to be very helpful as a general guide on how to construct SQL statements!

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [lirk](#) 

[Answer](#)  by [daniel-vassallo](#) 

You can JOIN with the same table more than once by giving the joined tables an [alias](#) , as in the following example:

[Skip code block](#)

```
SELECT
    airline, flt_no, fairport, tairport, depart, arrive, fare
FROM
    flights
INNER JOIN
    airports from_port ON (from_port.code = flights.fairport)
INNER JOIN
    airports to_port ON (to_port.code = flights.tairport)
WHERE
    from_port.code = '?' OR to_port.code = '?' OR airports.city='?'
```

Note that the `to_port` and `from_port` are aliases for the first and second copies of the `airports` table.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: Why is SELECT * considered harmful?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

Why is `SELECT *` bad practice? Wouldn't it mean less code to change if you added a new column you wanted?

I understand that `SELECT COUNT(*)` is a performance problem on some DBs, but what if you really wanted every column?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

User: [theodore-r.-smith](#) 

[Answer](#)  by [dave-markle](#) 

There are really three major reasons:

- **Inefficiency in moving data to the consumer.** When you `SELECT *`, you're often retrieving more columns from the database than your application really needs to function. This causes more data to move from the database server to the client, slowing access and increasing load on your machines, as well as taking more time to travel across the network. This is especially true when someone adds new columns to underlying tables that didn't exist and weren't needed when the original consumers coded their data access.
- **Indexing issues.** Consider a scenario where you want to tune a query to a high level of performance. If you were to use `*`, and it returned more columns than you actually needed, the server would often have to perform more expensive methods to retrieve

your data than it otherwise might. For example, you wouldn't be able to create an index which simply covered the columns in your SELECT list, and even if you did (including all columns [*shudder*]), the next guy who came around and added a column to the underlying table would cause the optimizer to ignore your optimized covering index, and you'd likely find that the performance of your query would drop substantially for no readily apparent reason.

- **Binding Problems.** When you SELECT *, it's possible to retrieve two columns of the same name from two different tables. This can often crash your data consumer. Imagine a query that joins two tables, both of which contain a column called "ID". How would a consumer know which was which? SELECT * can also confuse views (at least in some versions SQL Server) when underlying table structures change — [the view is not rebuilt, and the data which comes back can be nonsense](#). And the worst part of it is that you can take care to name your columns whatever you want, but the next guy who comes along might have no way of knowing that he has to worry about adding a column which will collide with your already-developed names.

But it's not all bad for SELECT *. I use it liberally for these use cases:

- **Ad-hoc queries.** When trying to debug something, especially off a narrow table I might not be familiar with, SELECT * is often my best friend. It helps me just see what's going on without having to do a boatload of research as to what the underlying column names are. This gets to be a bigger "plus" the longer the column names get.
- **When * means "a row".** In the following use cases, SELECT * is just fine, and rumors that it's a performance killer are just urban legends which may have had some validity many years ago, but don't now:

```
SELECT COUNT(*) FROM table;
```

in this case, * means "count the rows". If you were to use a column name instead of *, *it would count the rows where that column's value was not null*. COUNT(*), to me, really drives home the concept that you're counting rows, and you avoid strange edge-cases caused by NULLs being eliminated from your aggregates.

Same goes with this type of query:

```
SELECT a.ID FROM TableA a
WHERE EXISTS (
    SELECT *
    FROM TableB b
    WHERE b.ID = a.B_ID);
```

in any database worth its salt, * just means "a row". It doesn't matter what you put in the subquery. Some people use b's ID in the SELECT list, or they'll use the number 1, but IMO those conventions are pretty much nonsensical. What you mean is "count the row", and that's what * signifies. Most query optimizers out there are smart enough to know this. (Though to be honest, I only *know* this to be true with SQL Server and Oracle.)

The asterisk character, “*”, in the SELECT statement is shorthand for all the columns in the table(s) involved in the query.

Performance

The * shorthand can be slower because:

- Not all the fields are indexed, forcing a full table scan - less efficient
- What you save to send `SELECT *` over the wire risks a full table scan
- Returning more data than is needed
- Returning trailing columns using variable length data type can result in search overhead

Maintenance

When using `SELECT *`:

- Someone unfamiliar with the codebase would be forced to consult documentation to know what columns are being returned before being able to make competent changes. Making code more readable, minimizing the ambiguity and work necessary for people unfamiliar with the code saves more time and effort in the long run.
- If code depends on column order, `SELECT *` will hide an error waiting to happen if a table had its column order changed.
- Even if you need every column at the time the query is written, that might not be the case in the future
- the usage complicates profiling

Design

`SELECT *` is an *anti-pattern*:

- The purpose of the query is less obvious; the columns used by the application is opaque
- It breaks the modularity rule about using strict typing whenever possible. Explicit is almost universally better.

When Should “`SELECT *`” Be Used?

It's acceptable to use `SELECT *` when there's the explicit need for every column in the table(s) involved, as opposed to every column that existed when the query was written. The database will internally expand the * into the complete list of columns - there's no performance difference.

Otherwise, explicitly list every column that is to be used in the query - preferably while using a table alias.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#))

[Q: Use email address as primary key?](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

Is email address a bad candidate for primary when compared to auto incrementing numbers?

Our web application needs the email address to be unique in the system. So, I thought of using email address as primary key. However my colleague suggests that string comparison will be slower than integer comparison.

Is it a valid reason to not use email as primary key?

We are using PostgreSQL.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [robert](#) 

[Answer](#)  by [sjoerd](#) 

String comparison is slower than int comparison. However, this does not matter if you simply retrieve a user from the database using the e-mail address. It does matter if you have complex queries with multiple joins.

If you store information about users in multiple tables, the foreign keys to the users table will be the e-mail address. That means that you store the e-mail address multiple times.

[Answer](#)  by [hlgem](#) 

I will also point out that email is a bad choice to make a unique field, there are people and even small businesses that share an email address. And like phone numbers, **emails can get re-used.** Jsmith@somecompany.com can easily belong to John Smith one year and Julia Smith two years later.

Another problem with emails is that they change frequently. If you are joining to other tables with that as the key, then you will have to update the other tables as well which can be quite a performance hit when an entire client company changes their emails (which I have seen happen.)

[Answer](#)  by [steven-a.-lowe](#) 

the primary key should be *unique* and *constant*

email addresses change like the seasons. Useful as a secondary key for lookup, but a poor choice for the primary key.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [database-design](#) ([Prev Q](#))

Q: What are the Options for Storing Hierarchical Data in a Relational Database?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [hierarchical-data](#) ([Next Q](#))

Good Overviews

Generally speaking you're making a decision between fast read times (e.g. nested set) or fast write times (adjacency list). Usually you end up with a combination of the options below that best fit your needs. The following provides some in depth reading:

- [One more Nested Intervals vs. Adjacency List comparison](#): *the best comparison* of Adjacency List, Materialized Path, Nested Set and Nested Interval I've found.
- [Models for hierarchical data](#): slides with good explanations of tradeoffs and example usage
- [Representing hierarchies in MySQL](#): very good overview of Nested Set in particular
- [Hierarchical data in RDBMSs](#): most comprehensive and well organized set of links I've seen, but not much in the way on explanation

Options

Ones I am aware of and general features:

1. [Adjacency List](#):
 - Columns: ID, ParentID
 - Easy to implement.
 - Cheap node moves, inserts, and deletes.
 - Expensive to find level (can store as a computed column), ancestry & descendants (Bridge Table combined with level column can solve), path (Lineage Column can solve).
 - Use [Common Table Expressions](#) in those databases that support them to traverse.
2. [Nested Set](#) (a.k.a Modified Preorder Tree Traversal)
 - Popularized by Joe Celko in numerous articles and his book [Trees and Hierarchies in SQL for Smarties](#)
 - Columns: Left, Right
 - Cheap level, ancestry, descendants
 - Compared to Adjacency List, moves, inserts, deletes more expensive.
 - Requires a specific sort order (e.g. created). So sorting all descendants in a different order requires additional work.
3. [Nested Intervals](#)
 - Combination of Nested Sets and Materialized Path where left/right columns are floating point decimals instead of integers and encode the path information. In the later development of this idea nested intervals gave rise to [matrix](#)

[encoding](#)

4. [Bridge Table](#) (a.k.a. [Closure Table](#): some good ideas about how to use triggers for maintaining this approach)
 - Columns: ancestor, descendant
 - Stands apart from table it describes.
 - Can include some nodes in more than one hierarchy.
 - Cheap ancestry and descendants (albeit not in what order)
 - For complete knowledge of a hierarchy needs to be combined with another option.
5. [Flat Table](#)
 - A modification of the Adjacency List that adds a Level and Rank (e.g. ordering) column to each record.
 - Expensive move and delete
 - Cheap ancestry and descendants
 - Good Use: threaded discussion - forums / blog comments
6. [Lineage Column](#) (a.k.a. [Materialized Path](#), Path Enumeration)
 - Column: lineage (e.g. /parent/child/grandchild/etc...)
 - Limit to how deep the hierarchy can be.
 - Descendants cheap (e.g. LEFT(lineage, #) = '/enumerated/path')
 - Ancestry tricky (database specific queries)
7. [Multiple lineage columns](#)
 - Columns: one for each lineage level, refers to all the parents up to the root, levels down from the items level are set to NULL
 - Limit to how deep the hierarchy can be
 - Cheap ancestors, descendants, level
 - Cheap insert, delete, move of the leaves
 - Expensive insert, delete, move of the internal nodes

Database Specific Notes

MySQL

- [Use session variables for Adjacency List](#)

Oracle

- Use [CONNECT BY](#) to traverse Adjacency Lists

PostgreSQL

- [ltree datatype](#) for Materialized Path

SQL Server

- [General summary](#)
- 2008 offers [HierarchyId](#) data type appears to help with Lineage Column approach and expand the depth that can be represented.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [hierarchical-data](#) ([Next Q](#))

User: [orangeips](#) 

[Answer](#)  by [tegiri-nenashi](#) 

This is kind of a question that is still interesting even after all big 3 vendors implemented Recursive WITH clause. I'd suggest that different readers would be pleased with different answers.

1. [Comprehensive list of references](#)  by Troels Arvin.
 2. For the lack of competition, introductory textbook by Joe Celko "Trees and Hierarchies in SQL for Smarties" can indeed be considered a classics.
 3. Review of various tree encodings with emphasis to [nested intervals](#) .
-

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [hierarchical-data](#) ([Next Q](#))

Q: What does MongoDB not being ACID compliant really mean?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [nosql](#) ([Prev Q](#)) ([Next Q](#))

I am not a database expert and have no formal computer science background, so bear with me. I want to know the kinds of *real world* negative things that can happen if you use MongoDB, which is not [ACID](#) compliant. This applies to any ACID noncompliant database.

I understand that MongoDB can perform [Atomic Operations](#), but that they don't "support traditional locking and complex transactions", mostly for performance reasons. I also understand the importance of database transactions, and the example of when your database is for a bank, and you're updating several records that all need to be in sync, you want the transaction to revert back to the initial state if there's a power outage so credit equals purchase, etc.

But when I get into conversations about MongoDB, those of us that don't know the technical details of how databases are actually implemented start throwing around statements like:

MongoDB is way faster than MySQL and Postgres, but there's a tiny chance, like 1 in a million, that it "won't save correctly".

That "won't save correctly" part is referring to this understanding: If there's a power outage right at the instant you're writing to MongoDB, there's a chance for a particular record (say you're tracking pageviews in documents with 10 attributes each), that one of the documents only saved 5 of the attributes... which means over time your pageview counters are going to be "slightly" off. You'll never know by how much, you know they'll be 99.999% correct, but not 100%. This is because, unless you specifically made this a [mongodb atomic operation](#), the operation is not guaranteed to have been atomic.

So my question is, what is the correct interpretation of when and why MongoDB may not "save correctly"? What parts of ACID does it not satisfy, and under what circumstances, and how do you know when that 0.001% of your data is off? Can't this be fixed somehow? If not, this seems to mean that you shouldn't store things like your users table in MongoDB, because a record might not save. But then again, that 1/1,000,000 user might just need to "try signing up again", no?

I am just looking for maybe a list of when/why negative things happen with an ACID noncompliant database like MongoDB, and ideally if there's a standard workaround (like run a background job to cleanup data, or only use SQL for this, etc.).

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [nosql](#) ([Prev Q](#)) ([Next Q](#))

User: [lance-pollard](#)

[Answer](#) by [bryan-migliorisi](#)

One thing you lose with MongoDB is multi-collection (table) transactions. Atomic modifiers in MongoDB can only work against a single document.

If you need to remove an item from inventory and add it to someone's order at the same time - you can't. Unless those two things - inventory and orders - exist in the same document (which they probably do not).

I encountered this very issue in an application I am working on and two possible solutions exist:

1) Structure your documents as best you can and use atomic modifiers as best you can and for the remaining bit, use a background process to cleanup records that may be out of sync. For example, I remove items from inventory and add them to a reservedInventory array of the same document using atomic modifiers.

This lets me always know that items are NOT available in the inventory (because they are reserved by a customer). When the customer checks out, I then remove the items from the reservedInventory. It's not a standard transaction and since the customer could abandon the cart, I need some background process to go through and find abandoned carts and move the reserved inventory back into the available inventory pool.

This is obviously less than ideal, but it's the only part of a large application where MongoDB does not fit the need perfectly. Plus, it works flawlessly thus far. This may not be possible for many scenarios, but because of the document structure I am using, it fits well.

2) Use a transactional database in conjunction with MongoDB. It is common to use MySQL to provide transactions for the things that absolutely need them while letting MongoDB (or any other NoSQL) do what it does best.

If my solution from #1 does not work in the long run, I will investigate further into combining MongoDB with MySQL but for now #1 suits my needs well.

[Answer](#)  by [william-z](#) 

It's actually not correct that MongoDB is not ACID-compliant. On the contrary, MongoDB is ACID-compliant ***at the document level***.

Any update to a single document is

- Atomic: it either fully completes or it does not
- Consistent: no reader will see a "partially applied" update
- Isolated: again, no reader will see a "dirty" read
- Durable: (with the appropriate write concern)

What MongoDB doesn't have is ***transactions*** — that is, multiple-document updates that can be rolled back and are ACID-compliant.

Note that you can build transactions on top of the ACID-compliant updates to a single document, by [using two-phase commit](#) .

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [nosql](#) ([Prev Q](#)) ([Next Q](#))

Q: Best way to work with dates in Android SQLite 

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [sqlite](#) ([Prev Q](#)) ([Next Q](#))

I'm having some trouble working with dates on my Android application that uses SQLite. I have a couple questions:

1. What type should I use to store dates in SQLite (text, integer, ...)?
2. Given the best way to store dates how do I store it properly using ContentValues?
3. What's the best way to retrieve the date from the SQLite database?
4. How to make a sql select on SQLite, ordering the results by date?

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [sqlite](#) ([Prev Q](#)) ([Next Q](#))

User: [filipe](#) 

[Answer](#)  by [pearsonartphoto](#) 

The best way is to store the dates as an integer, received by using the Calendar command.

```
//Building the table includes:  
StringBuilder query=new StringBuilder();  
query.append("CREATE TABLE "+TABLE_NAME+ " (");  
query.append(COLUMN_ID+" int primary key autoincrement,");  
query.append(COLUMN_DATETIME+" int"));  
  
//And inserting the data includes this:  
values.put(COLUMN_DATETIME, System.currentTimeMillis());
```

Why do this? First of all, getting values from a date range is easy. Just convert your date into milliseconds, and then query appropriately. Sorting by date is similarly easy. The calls to convert among various formats are also likewise easy, as I included. Bottom line is, with this method, you can do anything you need to do, no problems. It will be slightly difficult to read a raw value, but it more than makes up that slight disadvantage with being easily machine readable and usable. And in fact, it is relatively easy to build a reader (And I know there are some out there) that will automatically convert the time tag to date as such for easy of reading.

It's worth mentioning that the values that come out of this should be long, not int. [Integer in sqlite](#)  can mean many things, anything from 1-8 bytes, but for almost all dates 64 bits, or a long, is what works.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [sqlite](#) ([Prev Q](#)) ([Next Q](#))

[Q: Optimise PostgreSQL for fast testing](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

I am switching to PostgreSQL from SQLite for a typical Rails application.

The problem is that running specs became slow with PG.

On SQLite it took ~34 seconds, on PG it's ~76 seconds which is **more than 2x slower**.

So now I want to apply some techniques to **bring the performance of the specs on par**

with SQLite with no code modifications (ideally just by setting the connection options, which is probably not possible).

Couple of obvious things from top of my head are:

- RAM Disk (good setup with RSpec on OSX would be good to see)
- Unlogged tables (can it be applied on the whole database so I don't have change all the scripts?)

As you may have understood I don't care about reliability and the rest (the DB is just a throwaway thingy here).

I need to get the most out of the PG and make it **as fast as it can possibly be**.

Best answer would ideally describe the *tricks* for doing just that, setup and the drawbacks of those tricks.

UPDATE: `fsync = off + full_page_writes = off` only decreased time to ~65 seconds (~16 secs). Good start, but far from the target of 34.

UPDATE 2: I [tried to use RAM disk](#) but the performance gain was within an error margin. So doesn't seem to be worth it.

UPDATE 3:* I found the biggest bottleneck and now my specs run as fast as the SQLite ones.

The issue was the database cleanup that did the **truncation**. Apparently SQLite is way too fast there.

To "fix" it I open a **transaction** before each test and roll it back at the end.

Some numbers for ~700 tests.

- Truncation: SQLite - 34s, PG - 76s.
- Transaction: SQLite - 17s, PG - 18s.

2x speed increase for SQLite. 4x speed increase for PG.

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

User: [dmytrii-nagirniak](#)

[Answer](#) by [craig-ringer](#)

First, always use the latest version of PostgreSQL. Performance improvements are always coming, so you're probably wasting your time if you're tuning an old version. For example, [PostgreSQL 9.2 significantly improves the speed of TRUNCATE](#) and of course adds index-only scans. Even minor releases should always be followed; see the [version policy](#).

Don'ts

[Do NOT put a tablespace on a RAMdisk or other non-durable storage](#)

If you lose a tablespace the whole database may be damaged and hard to use without significant work. There's very little advantage to this compared to just using UNLOGGED tables and having lots of RAM for cache anyway.

If you truly want a ramdisk based system, initdb a whole new cluster on the ramdisk by initdbing a new PostgreSQL instance on the ramdisk, so you have a completely disposable PostgreSQL instance.

PostgreSQL server configuration

When testing, you can configure your server for [non-durable but faster operation](#).

This is one of the only acceptable uses for the [fsync=off](#) setting in PostgreSQL. This setting pretty much tells PostgreSQL not to bother with ordered writes or any of that other nasty data-integrity-protection and crash-safety stuff, giving it permission to totally trash your data if you lose power or have an OS crash.

Needless to say, you should never enable fsync=off in production unless you're using Pg as a temporary database for data you can re-generate from elsewhere. If and only if you're doing to turn fsync off can also turn [full_page_writes](#) off, as it no longer does any good then. Beware that fsync=off and full_page_writes apply at the *cluster* level, so they affect *all* databases in your PostgreSQL instance.

For production use you can possibly use `synchronous_commit=off` and set a `commit_delay`, as you'll get many of the same benefits as `fsync=off` without the giant data corruption risk. You do have a small window of loss of recent data if you enable `async commit` - but that's it.

If you have the option of slightly altering the DDL, you can also use UNLOGGED tables in Pg 9.1+ to completely avoid WAL logging and gain a real speed boost at the cost of the tables getting erased if the server crashes. There is no configuration option to make all tables unlogged, it must be set during `CREATE TABLE`. In addition to being good for testing this is handy if you have tables full of generated or unimportant data in a database that otherwise contains stuff you need to be safe.

Check your logs and see if you're getting warnings about too many checkpoints. If you are, you should increase your [checkpoint_segments](#). You may also want to tune your `checkpoint_completion_target` to smooth writes out.

Tune `shared_buffers` to fit your workload. This is OS-dependent, depends on what else is going on with your machine, and requires some trial and error. The defaults are extremely conservative. You may need to increase the OS's maximum shared memory limit if you increase `shared_buffers` on PostgreSQL 9.2 and below; 9.3 and above changed how they use shared memory to avoid that.

If you're using a just a couple of connections that do lots of work, increase `work_mem` to give them more RAM to play with for sorts etc. Beware that too high a `work_mem` setting can cause out-of-memory problems because it's per-sort not per-connection so one query can have many nested sorts. You only *really* have to increase `work_mem` if you can see sorts

spilling to disk in EXPLAIN or logged with the [log_temp_files setting](#) (recommended), but a higher value may also let Pg pick smarter plans.

As said by another poster here it's wise to put the xlog and the main tables/indexes on separate HDDs if possible. Separate partitions is pretty pointless, you really want separate drives. This separation has much less benefit if you're running with fsync=off and almost none if you're using UNLOGGED tables.

Finally, tune your queries. Make sure that your random_page_cost and seq_page_cost reflect your system's performance, ensure your effective_cache_size is correct, etc. Use EXPLAIN (BUFFERS, ANALYZE) to examine individual query plans, and turn the auto_explain module on to report all slow queries. You can often improve query performance dramatically just by creating an appropriate index or tweaking the cost parameters.

AFAIK there's no way to set an entire database or cluster as UNLOGGED. It'd be interesting to be able to do so. Consider asking on the PostgreSQL mailing list.

Host OS tuning

There's some tuning you can do at the operating system level, too. The main thing you might want to do is convince the operating system not to flush writes to disk aggressively, since you really don't care when/if they make it to disk.

In Linux you can control this with the [virtual memory subsystem](#)'s dirty_* settings, like dirty_writeback_centisecs.

The only issue with tuning writeback settings to be too slack is that a flush by some other program may cause all PostgreSQL's accumulated buffers to be flushed too, causing big stalls while everything blocks on writes. You may be able to alleviate this by running PostgreSQL on a different file system, but some flushes may be device-level or whole-host-level not filesystem-level, so you can't rely on that.

This tuning really requires playing around with the settings to see what works best for your workload.

On newer kernels, you may wish to ensure that vm.zone_reclaim_mode is set to zero, as it can cause severe performance issues with NUMA systems (most systems these days) due to interactions with how PostgreSQL manages shared_buffers.

Query and workload tuning

These are things that DO require code changes; they may not suit you. Some are things you might be able to apply.

If you're not batching work into larger transactions, start. Lots of small transactions are expensive, so you should batch stuff whenever it's possible and practical to do so. If you're using async commit this is less important, but still highly recommended.

Whenever possible use temporary tables. They don't generate WAL traffic, so they're lots faster for inserts and updates. Sometimes it's worth slurping a bunch of data into a temp

table, manipulating it however you need to, then doing an `INSERT INTO... SELECT...` to copy it to the final table. Note that temporary tables are per-session; if your session ends or you lose your connection then the temp table goes away, and no other connection can see the contents of a session's temp table(s).

If you're using PostgreSQL 9.1 or newer you can use [UNLOGGED](#) tables for data you can afford to lose, like session state. These are visible across different sessions and preserved between connections. They get truncated if the server shuts down uncleanly so they can't be used for anything you can't re-create, but they're great for caches, materialized views, state tables, etc.

In general, don't `DELETE FROM blah;`. Use `TRUNCATE TABLE blah;` instead; it's a lot quicker when you're dumping all rows in a table. Truncate many tables in one `TRUNCATE` call if you can. There's a caveat if you're doing lots of `TRUNCATES` of small tables over and over again, though; see: [Postgresql Truncation speed](#)

If you don't have indexes on foreign keys, `DELETEs` involving the primary keys referenced by those foreign keys will be horribly slow. Make sure to create such indexes if you ever expect to `DELETE` from the referenced table(s). Indexes are not required for `TRUNCATE`.

Don't create indexes you don't need. Each index has a maintenance cost. Try to use a minimal set of indexes and let bitmap index scans combine them rather than maintaining too many huge, expensive multi-column indexes. Where indexes are required, try to populate the table first, then create indexes at the end.

Hardware

Having enough RAM to hold the entire database is a huge win if you can manage it.

If you don't have enough RAM, the faster storage you can get the better. Even a cheap SSD makes a massive difference over spinning rust. Don't trust cheap SSDs for production though, they're often not crashsafe and might eat your data.

Learning

Greg Smith's book, [PostgreSQL 9.0 High Performance](#) remains relevant despite referring to a somewhat older version. It should be a useful reference.

Join the PostgreSQL general mailing list and follow it.

Reading:

- [Tuning your PostgreSQL server - PostgreSQL wiki](#)
- [Number of database connections - PostgreSQL wiki](#)

Tags: [database](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

Q: SQL JOIN and different types of JOINS

Tags: [database](#) ([Prev Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#))

I have been going through many threads on **SO** and some other forums. So I thought I would summarize “**What is SQL JOIN?**” and “**What are different types of SQL JOINs?**”.

Tags: [database](#) ([Prev Q](#)), [oracle](#) ([Prev Q](#)) ([Next Q](#)), [postgresql](#) ([Prev Q](#)) ([Next Q](#))

User: [m-d](#)

[Answer](#) by [m-d](#)

What is SQL JOIN ?

SQL JOIN is a method to retrieve data from two or more database tables.

What are the different SQL JOINS ?

There are a total of five JOINS. They are :

[Skip code block](#)

1. JOIN or INNER JOIN
2. OUTER JOIN
 - 2.1 LEFT OUTER JOIN or LEFT JOIN
 - 2.2 RIGHT OUTER JOIN or RIGHT JOIN
 - 2.3 FULL OUTER JOIN or FULL JOIN
3. NATURAL JOIN
4. CROSS JOIN
5. SELF JOIN

1. JOIN or INNER JOIN :

In this kind of a JOIN, we get all records that match the condition in both the tables, and records in both the tables that do not match are not reported.

In other words, INNER JOIN is based on the single fact that : ONLY the matching entries in BOTH the tables SHOULD be listed.

Note that a JOIN without any other JOIN keywords (like INNER, OUTER, LEFT, etc) is an INNER JOIN. In other words, INNER JOIN is a Syntactic sugar for JOIN (see : [Difference between JOIN and INNER JOIN](#)).

2. OUTER JOIN :

OUTER JOIN retrieves

Either, the matched rows from one table and all rows in the other table Or, all rows in all tables (it doesn't matter whether or not there is a match).

There are three kinds of Outer Join :

2.1 LEFT OUTER JOIN or LEFT JOIN

This join returns all the rows from the left table in conjunction with the matching rows from the right table. If there are no columns matching in the right table, it returns NULL values.

2.2 RIGHT OUTER JOIN or RIGHT JOIN

This JOIN returns all the rows from the right table in conjunction with the matching rows from the left table. If there are no columns matching in the left table, it returns NULL values.

2.3 FULL OUTER JOIN or FULL JOIN

This JOIN combines LEFT OUTER JOIN and RIGHT OUTER JOIN. It returns row from either table when the conditions are met and returns NULL value when there is no match.

In other words, OUTER JOIN is based on the fact that : ONLY the matching entries in ONE OF the tables (RIGHT or LEFT) or BOTH of the tables(FULL) SHOULD be listed.

Note that `OUTER JOIN` is a loosened form of `INNER JOIN` .

3. NATURAL JOIN :

It is based on the two conditions :

1. the JOIN is made on all the columns with the same name for equality.
2. Removes duplicate columns from the result.

This seems to be more of theoretical in nature and as a result (probably) most DBMS don't even bother supporting this.

4. CROSS JOIN :

It is the Cartesian product of the two tables involved. The result of a CROSS JOIN will not make sense in most of the situations. Moreover, we wont need this at all (or needs the least, to be precise).

5. SELF JOIN :

It is not a different form of JOIN, rather it is a JOIN (INNER, OUTER, etc) of a table to itself.

JOINS based on Operators

Depending on the operator used for a JOIN clause, there can be two types of JOINS. They are

1. Equi JOIN

2. Theta JOIN

1. Equi JOIN :

For whatever JOIN type (INNER, OUTER, etc), if we use ONLY the equality operator (=), then we say that the JOIN is an EQUI JOIN.

2. Theta JOIN :

This is same as EQUI JOIN but it allows all other operators like >, <, >= etc.

Many consider both EQUI JOIN and Theta JOIN similar to INNER, OUTER etc JOINS. But I strongly believe that its a mistake and makes the ideas vague. Because INNER JOIN, OUTER JOIN etc are all connected with the tables and their data where as EQUI JOIN and THETA JOIN are only connected with the operators we use in the former.

Again, there are many who consider NATURAL JOIN as some sort of “peculiar” EQUI JOIN. In fact, it is true, because of the first condition I mentioned for NATURAL JOIN. However, we dont have to restrict that simply to NATURAL JOINS alone. INNER JOINS, OUTER JOINS etc could be an EQUI JOIN too.

Tags: [database](#) ([Prev Q](#)) [oracle](#) ([Prev Q](#)) ([Next Q](#)) [postgresql](#) ([Prev Q](#)) ([Next Q](#))

SQL Server 2005

[Skip to questions](#),

Wiki by user [bali-c](#) 

[SQL Server 2005](#)  (codename Yukon, version 9.00), released in October 2005, is the successor to SQL Server 2000. It included native support for managing XML data, in addition to relational data.

For this purpose, it defined an XML data type that could be used either as a data type in database columns or as literals in queries. XML columns can be associated with XSD schemas; XML data being stored is verified against the schema. XML is converted to an internal binary data type before being stored in the database. Specialized indexing methods were made available for XML data. XML data is queried using XQuery; Common Language Runtime (CLR) integration was a main feature with this edition, enabling one to write SQL code as Managed Code by the CLR. SQL Server 2005 added some extensions to the T-SQL language to allow embedding XQuery queries in T-SQL.

In addition, it also defines a new extension to XQuery, called XML DML, that allows query-based modifications to XML data. SQL Server 2005 also allows a database server to be exposed over web services using Tabular Data Stream (TDS) packets encapsulated within SOAP (protocol) requests. When the data is accessed over web services, results are returned as XML.

Resources

- [Microsoft SQL Server Wikipedia Article](#) 
- [Microsoft SQL Server 2005 Wikipedia Article](#) 

Source: [SQL Centre \(MS SQL Server 2005\)](#) 

Questions

[Q: Dynamic SELECT TOP @var In SQL Server](#)

Tags: [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

How can I have a dynamic variable setting the amount of rows to return in SQL Server?

Below is not valid syntax in SQL Server 2005+:

```
DECLARE @count int  
SET @count = 20  
  
SELECT TOP @count * FROM SomeTable
```

Tags: [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

User: [eddiegroves](#) 

[Answer](#)  by [brian-kim](#) 

```
SELECT TOP (@count) * FROM SomeTable
```

This will only work with SQL 2005+

Tags: [sql-server-2005](#) ([Prev Q](#)) ([Next Q](#))

Q: Difference between left join and right join in SQL Server

Tags: [sql-server-2005](#) ([Prev Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

I know about joins in SQL Server.

For example. there is two tables Table1, Table2.

There table structure are following.

```
create table Table1 (id int, Name varchar (10))
create table Table2 (id int, Name varchar (10))
```

Table1 Data as follows:

Id	Name
1	A
2	B

Table2 Data as follows:

Id	Name
1	A
2	B
3	C

If I execute both below mentioned SQL statements, both outputs will be the same

```
select *
from Table1
    left join Table2 on Table1.id = Table2.id

select *
from Table2
    right join Table1 on Table1.id = Table2.id
```

Please explain the difference between left and right join in above sql statements.

Tags: [sql-server-2005](#) ([Prev Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

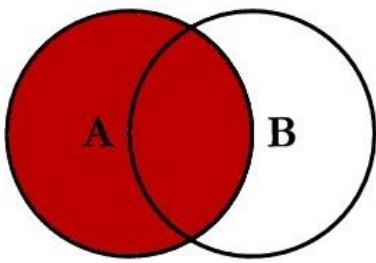
User: [pankaj-agarwal](#)

Answer by [daan-timmer](#)

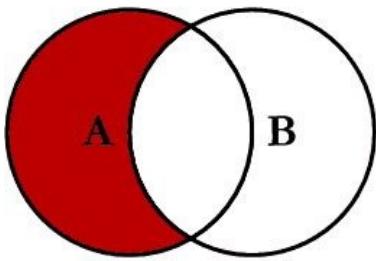
Codeproject has this image which explains the simple basics of SQL joins, taken from:

http://www.codeproject.com/KB/database/Visual_SQL_Joins.aspx

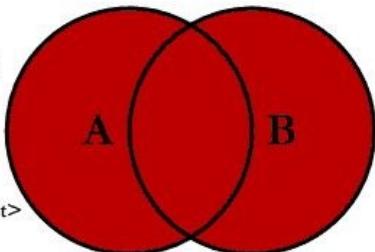
SQL JOINS



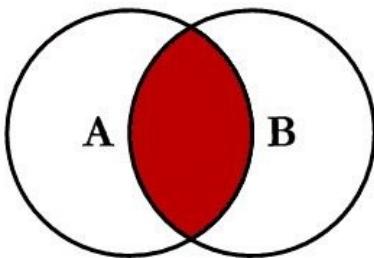
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



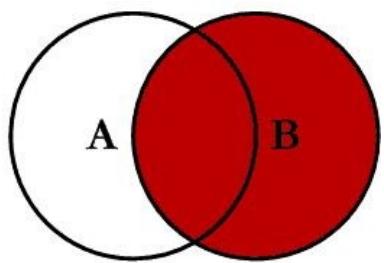
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```



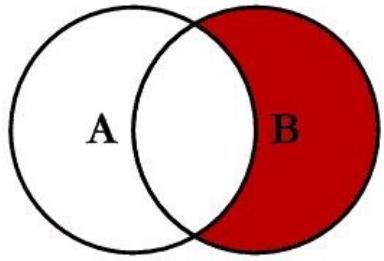
```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```



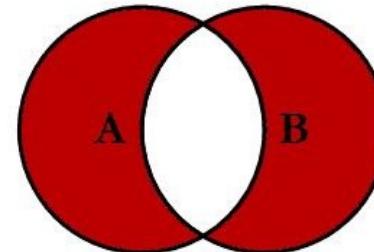
```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

© C.L. Moffatt, 2008

Tags: [sql-server-2005](#) ([Prev Q](#)), [join](#) ([Prev Q](#)) ([Next Q](#))

Oracle

[Skip to questions](#),

Wiki by user [jeffrey-kemp](#) 

[Oracle](#)  is an Object-relational Database Management System ([ORDBMS](#) ) created by [Oracle Corporation](#) .

Note: This tag is *not* for questions about other products owned by Oracle Corporation, such as Java and MySQL.

It supports a large number of languages and application development frameworks. Its primary languages are **SQL**, **PL/SQL** and **Java**.

When posting questions with the [oracle](#) tag, please be sure to **always** include Oracle edition and version.

How to tag questions

- about SQL in general should probably be tagged [sql](#).
- about PL/SQL in particular should probably be tagged [plsql](#)  as well as [oracle](#).
- specific to a particular database version *may* be tagged [oracle11g](#) , [oracle10g](#)  or [oracle9i](#) , as appropriate, but they should also be tagged [oracle](#) at least.
- about the free version of Oracle, Oracle XE, *may* be tagged [oracle10g](#)  or [oracle11g](#) , but should be tagged [oracle](#) at least.
- about Oracle HTMLDB or Application Express (any version) *may* be tagged [oracle-apex](#) , but should be tagged [oracle](#) at least.
- about other Oracle Corp. database products should *not* be tagged [oracle](#), rather use the more specific tag to the technology, e.g. [mysql](#).

Free version of Oracle database

Oracle provides a free version of the database called [Oracle Database Express Edition](#) .

Useful links

- [Oracle Documentation](#) 
- [SQL Language Reference](#) 
- [PL/SQL Language Reference](#) 
- [PL/SQL Packages and Types Reference](#) 
- [Oracle Ask Tom](#) 

Book recommendations

- [Expert Oracle Database Architecture: Oracle Database 9i, 10g, and 11g Programming Techniques and Solutions by Thomas Kyte](#) 

Questions

[Q: How do I find duplicate values in a table in Oracle?](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

What's the simplest SQL statement that will return the duplicate values for a given column and the count of their occurrences in an Oracle database table?

For example: I have a JOBS table with the column JOB_NUMBER. How can I find out if I have any duplicate JOB_NUMBERS, and how many times they're duplicated?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [andrew](#) 

[Answer](#)  by [bill-the-lizard](#) 

```
select column_name, count(column_name)
from table
group by column_name
having count (column_name) > 1;
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Difference between BYTE and CHAR in column datatypes](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

In Oracle, what is the difference between :

```
CREATE TABLE CLIENT
(
  NAME VARCHAR2(11 BYTE),
  ID_CLIENT NUMBER
)
```

and

```
CREATE TABLE CLIENT
(
  NAME VARCHAR2(11 CHAR),--or even VARCHAR2(11)
  ID_CLIENT NUMBER
)
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [guido-garcía](#) 

[Answer](#)  by [david-sykes](#) 

Let us assume the database character set is UTF-8, which is the recommended setting in recent versions of Oracle. In this case, some characters take more than 1 byte to store in the database.

If you define the field as VARCHAR2(11 BYTE), Oracle can use up to 11 bytes for storage,

but you may not actually be able to store 11 characters in the field, because some of them take more than one byte to store, e.g. non-English characters.

By defining the field as VARCHAR2(11 CHAR) you tell Oracle it can use enough space to store 11 characters, no matter how many bytes it takes to store each one. A single character may require up to 4 bytes.

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I confirm a database is Oracle & what version it is using SQL?](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

I'm building an installer for an application. The user gets to select a datasource they have configured and nominate what type of database it is. I want to confirm that the database type is indeed Oracle, and if possible, what version of Oracle they are running by sending a SQL statement to the datasource.

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [modius](#) 

[Answer](#)  by [tony-andrews](#) 

Run this SQL:

```
select * from v$version;
```

And you'll get a result like:

```
BANNER
-----
Oracle Database 10g Release 10.2.0.3.0 - 64bit Production
PL/SQL Release 10.2.0.3.0 - Production
CORE    10.2.0.3.0      Production
TNS for Solaris: Version 10.2.0.3.0 - Production
NLSRTL Version 10.2.0.3.0 - Production
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Fetch the row which has the Max value for a column](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

Table: UserId, Value, Date.

I want to get the UserId, Value for the max(Date) for each UserId. That is, the Value for each UserId that has the latest date. Is there a way to do this simply in SQL? (Preferably Oracle)

Update: Apologies for any ambiguity: I need to get ALL the UserIds. But for each UserId, only that row where that user has the latest date.

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [umang](#) 

[Answer](#)  by [david-aldrige](#) 

This will retrieve all rows for which the my_date column value is equal to the maximum value of my_date for that userid. This may retrieve multiple rows for the userid where the maximum date is on multiple rows.

[Skip code block](#)

```
select userid,
       my_date,
       ...
  from
  (
  select userid,
         my_date,
         ...
        max(my_date) over (partition by userid) max_my_date
   from   users
  )
 where my_date = max_my_date
```

“Analytic functions rock”

Edit: With regard to the first comment ...

“using analytic queries and a self-join defeats the purpose of analytic queries”

There is no self-join in this code. There is instead a predicate placed on the result of the inline view that contains the analytic function — a very different matter, and completely standard practice.

“The default window in Oracle is from the first row in the partition to the current one”

The windowing clause is only applicable in the presence of the order by clause. With no order by clause, no windowing clause is applied by default and none can be explicitly specified.

The code works.

[Answer](#)  by [bill-karwin](#) 

I see many people use subqueries or else vendor-specific features to do this, but I often do this kind of query without subqueries in the following way. It uses plain, standard SQL so it should work in any brand of RDBMS.

```
SELECT t1.*
  FROM mytable t1
  LEFT OUTER JOIN mytable t2
    ON (t1.UserId = t2.UserId AND t1."Date" < t2."Date")
 WHERE t2.UserId IS NULL;
```

In other words: fetch the row from t1 where no other row exists with the same UserId and a greater Date.

(I put the identifier “Date” in delimiters because it’s an SQL reserved word.)

In case if t1.“Date” = t2.“Date”, doubling appears. Usually tables has auto_inc(seq) key, eg Id. To avoid doubling can be used follows:

```
SELECT t1.*  
FROM mytable t1  
LEFT OUTER JOIN mytable t2  
ON t1.UserId = t2.UserId AND ((t1."Date" < t2."Date")  
OR (t1."Date" = t2."Date" AND t1.id < t2.id))  
WHERE t2.UserId IS NULL;
```

Re comment from @Farhan:

Here's a more detailed explanation:

An outer join attempts to join t1 with t2. By default, all results of t1 are returned, and if there is a match in t2, it is also returned. If there is no match in t2 for a given row of t1, then the query still returns the row of t1, and uses NULL as a placeholder for all of t2's columns. That's just how outer joins work in general.

The trick in this query is to design the join's matching condition such that t2 must match the *same* userid, and a *greater* date. The idea being if a row exists in t2 that has a greater date, then the row in t1 it's compared against *can't* be the greatest date for that userid. But if there is no match — i.e. if no row exists in t2 with a greater date than the row in t1 — we know that the row in t1 was the row with the greatest date for the given userid.

In those cases (when there's no match), the columns of t2 will be NULL — even the columns specified in the join condition. So that's why we use WHERE t2.UserId IS NULL, because we're searching for the cases where no row was found with a greater date for the given userid.

[Answer](#) by [dave-costa](#)

```
SELECT userid, MAX(value) KEEP (DENSE_RANK FIRST ORDER BY date DESC)  
FROM table  
GROUP BY userid
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Inner join vs Where](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

Is there a difference in performance (in oracle) between

```
Select * from Table1 T1  
Inner Join Table2 T2 On T1.ID = T2.ID
```

And

```
Select * from Table1 T1, Table2 T2  
Where T1.ID = T2.ID
```

?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

User: [juanformoso](#)

[Answer](#) by [kiewic](#)

No! The same execution plan, look at these two tables:

```
CREATE TABLE table1 (
    id INT,
    name VARCHAR(20)
);

CREATE TABLE table2 (
    id INT,
    name VARCHAR(20)
);
```

The execution plan for the query using the inner join:

[Skip code block](#)

```
-- with inner join

EXPLAIN PLAN FOR
SELECT * FROM table1 t1
INNER JOIN table2 t2 ON t1.id = t2.id;

SELECT *
FROM TABLE (DBMS_XPLAN.DISPLAY);
--0 select statement-1 hash join (access("T1"."ID"="T2"."ID"))-2 table access full table1-3 table acce
```

And the execution plan for the query using a WHERE clause.

[Skip code block](#)

```
-- with where clause

EXPLAIN PLAN FOR
SELECT * FROM table1 t1, table2 t2
WHERE t1.id = t2.id;

SELECT *
FROM TABLE (DBMS_XPLAN.DISPLAY);
--0 select statement-1 hash join (access("T1"."ID"="T2"."ID"))-2 table access full table1-3 table acce
```

[Answer](#) by [craig-trader](#)

If the query optimizer is doing its job right, there should be no difference between those queries. They are just two ways to specify the same desired result.

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#))

[Q: Why does Oracle 9i treat an empty string as NULL?](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

I know that it *does* consider '' as NULL, but that doesn't do much to tell me *why* this is the case. As I understand the SQL specifications, '' is not the same as NULL — one is a valid datum, and the other is indicating the absence of that same information.

Feel free to speculate, but please indicate if that's the case. If there's anyone from Oracle who can comment on it, that'd be fantastic!

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [chris-r](#)

[Answer](#) by [justin-cave](#)

I believe the answer is that Oracle is very, very old.

Back in the olden days before there was a SQL standard, Oracle made the design decision that empty strings in VARCHAR/VARCHAR2 columns were NULL and that there was only one sense of NULL (there are relational theorists that would differentiate between data that has never been prompted for, data where the answer exists but is not known by the user, data where there is no answer, etc. all of which constitute some sense of NULL).

By the time that the SQL standard came around and agreed that NULL and the empty string were distinct entities, there were already Oracle users that had code that assumed the two were equivalent. So Oracle was basically left with the options of breaking existing code, violating the SQL standard, or introducing some sort of initialization parameter that would change the functionality of potentially large number of queries. Violating the SQL standard (IMHO) was the least disruptive of these three options.

Oracle has left open the possibility that the VARCHAR data type would change in a future release to adhere to the SQL standard (which is why everyone uses VARCHAR2 in Oracle since that data type's behavior is guaranteed to remain the same going forward).

[Answer](#) by [brian-hart](#)

[Tom Kyte](#) VP of Oracle:

A ZERO length varchar is treated as NULL.

” is not treated as NULL.

” when assigned to a char(1) becomes ‘ ‘ (char types are blank padded strings).

” when assigned to a varchar2(1) becomes ” which is a zero length string and a zero length string is NULL in Oracle (it is no long ”)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Get list of all tables in Oracle?](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

How do I query an Oracle database to display the names of all tables in it?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [vitule](#)

[Answer](#) by [justin-cave](#)

```
SELECT owner, table_name  
FROM dba_tables
```

assuming that you have access to the DBA_TABLES data dictionary view. If you do not have those privileges but need them, you can request that the DBA explicitly grants you privileges on that table or that the DBA grants you the SELECT ANY DICTIONARY privilege or the SELECT_CATALOG_ROLE role (either of which would allow you to query any data dictionary table). Of course, you may want to exclude certain schemas like SYS and SYSTEM which have large numbers of Oracle tables that you probably don't care about.

Alternatively, if you do not have access to DBA_TABLES, you can see all the tables that your account has access to through the ALL_TABLES view

```
SELECT owner, table_name  
FROM all_tables
```

although that may be a subset of the tables available in the database (ALL_TABLES shows you the information for all the tables that your user has been granted access to).

If you are only concerned with the tables that you own, not those that you have access to, you could use USER_TABLES

```
SELECT table_name  
FROM user_tables
```

Since USER_TABLES only has information about the tables that you own, it does not have an OWNER column—the owner, by definition, is you.

Oracle also has a number of legacy data dictionary views—TAB, DICT, TABS, and CAT for example—that could be used. In general, I would not suggest using these legacy views unless you absolutely need to backport your scripts to Oracle 6. Oracle has not changed these views in a long time so they often have problems with newer types of objects. For example, the TAB and CAT views both show information about tables that are in the user's recycle bin while the [DBA|ALL|USER]_TABLES views all filter those out. CAT also shows information about materialized view logs with a TABLE_TYPE of "TABLE" which is unlikely to be what you really want. DICT combines tables and synonyms and doesn't tell you who owns the object.

[Answer](#) by [vitule](#)

Querying user_tables and dba_tables didn't work.
This one did:

```
select table_name from all_tables
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I create a copy of an Oracle table without copying the data?](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

I know the statement:

```
create table xyz_new as select * from xyz;
```

Which copies the structure and the data, but what if I just want the structure?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [andrew](#) 

[Answer](#)  by [jim-hudson](#) 

Just use a where clause that won't select any rows:

```
create table xyz_new as select * from xyz where 1=0;
```

[Answer](#)  by [dave-costa](#) 

I used the method that you accepted a lot, but as someone pointed out it doesn't duplicate constraints (except for NOT NULL, I think).

A more advanced method if you want to duplicate the full structure is:

```
SET LONG 5000
SELECT dbms_metadata.get_ddl( 'TABLE', 'MY_TABLE_NAME' ) FROM DUAL;
```

This will give you the full create statement text which you can modify as you wish for creating the new table. You would have to change the names of the table and all constraints of course.

(You could also do this in older versions using EXP/IMP, but it's much easier now.)

Edited to add If the table you are after is in a different schema:

```
SELECT dbms_metadata.get_ddl( 'TABLE', 'MY_TABLE_NAME', 'OTHER_SCHEMA_NAME' ) FROM DUAL;
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Oracle: how to UPSERT \(update or insert into a table?\)](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

The UPSERT operation either updates or inserts a row in a table, depending if the table already has a row that matches the data:

```
if table t has a row exists that has key X:  
    update t set mystuff... where mykey=X  
else  
    insert into t mystuff...
```

Since Oracle doesn't have a specific UPSERT statement, what's the best way to do this?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [mark-harrison](#) 

[Answer](#)  by [mark-harrison](#) 

The [MERGE statement](#) merges data between two tables. Using DUAL allows us to use this command.

[Skip code block](#)

```
create or replace  
procedure ups(xa number)  
as  
begin  
    merge into mergetest m using dual on (a = xa)  
        when not matched then insert (a,b) values (xa,1)  
            when matched then update set b = b+1;  
end ups;  
/  
drop table mergetest;  
create table mergetest(a number, b number);  
call ups(10);  
call ups(10);  
call ups(20);  
select * from mergetest;
```

A	B
10	2
20	1

[Answer](#)  by [mydeveloperday](#) 

The dual example above which is in PL/SQL was great because I wanted to do something similar, but I wanted it client side...so here is the SQL I used to send a similar statement direct from some C#

```
MERGE INTO Employee USING dual ON ( "id"=2097153 )  
WHEN MATCHED THEN UPDATE SET "last"="smith" , "name"="john"  
WHEN NOT MATCHED THEN INSERT ("id","last","name")  
    VALUES ( 2097153,"smith", "john" )
```

However from a C# perspective this provide to be slower than doing the update and seeing if the rows affected was 0 and doing the insert if it was.

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

Q: How do I limit the number of rows returned by an Oracle query after ordering? 

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

Is there a way to make an Oracle query behave like it contains a MySQL limit clause?

In MySQL, I can do this:

```
select *  
from sometable  
order by name  
limit 20,10
```

to get the 21st to the 30th rows (skip the first 20, give the next 10). The rows are selected after the `order by`, so it really starts on the 20th name alphabetically.

In Oracle, the only thing people mention is the `rownum` pseudo-column, but it is evaluated *before* `order by`, which means this:

```
select *  
from sometable  
where rownum <= 10  
order by name
```

will return a random set of ten rows ordered by name, which is not usually what I want. It also doesn't allow for specifying an offset.

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [mathieu-longtin](#) 

[Answer](#)  by [kosi2801](#) 

You can use a subquery for this like

```
select *  
from  
( select *  
      from emp  
     order by sal desc )  
where ROWNUM <= 5;
```

Have also a look at the topic [On ROWNUM and limiting results](#)  at Oracle/AskTom for more information.

Update: To limit the result with both lower and upper bounds things get a bit more bloated with

```
select * from  
( select a.* , ROWNUM rnum from  
    ( <your_query_goes_here, with order by> ) a  
   where ROWNUM <= :MAX_ROW_TO_FETCH )  
where rnum >= :MIN_ROW_TO_FETCH;
```

(Copied from specified AskTom-article)

Update 2: Starting with Oracle 12c (12.1) there is a syntax available to limit rows or start at offsets.

```
SELECT *  
FROM sometable  
ORDER BY name  
OFFSET 20 ROWS FETCH NEXT 10 ROWS ONLY;
```

See [this answer](#)  for more examples. Thanks to Krumia for the hint.

[Answer](#)  by [zeldi](#) 

I did some performance testing between:

A.) AskTom

```
select * from (
  select a.* , ROWNUM rnum from (
    <select statemenet with order by clause>
  ) a where rownum <= MAX_ROW
) where rnum >= MIN_ROW
```

B.) Analytic approach

```
select * from (
  <select statemenet with order by clause>
) where myrow between MIN_ROW and MAX_ROW
```

C.) Short alternative

```
select * from (
  select statement, rownum as RN with order by clause
) where a.rn >= MIN_ROW and a.rn <= MAX_ROW
```

Results:

Table had 10 million records, sort was on an unindexed datetime row:

- Explain plan showed same value for all three selects (323168)
- But the winner is AskTom (with analytic following close behind)

Selecting first 10 rows took:

- AskTom: 28-30 seconds
- Analytical: 33-37 seconds
- Short alternative: 110-140 seconds

Selecting rows betwwen 100.000 and 100.010:

- AskTom: 60 seconds
- Analytical: 100 seconds

Selecting rows between 9.000.000 and 9.000.010:

- AskTom: 130 seconds
- Analytical: 150 seconds

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Select count\(*\) from multiple tables](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

How can I select count(*) from two different tables (call them tab1 and tab2) having as result:

Count_1	Count_2
123	456

?

I've tried this:

```
select count(*) Count_1 from schema.tab1 union all select count(*) Count_2 from schema.tab2
```

But all I have is:

```
Count_1  
123  
456
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [user73118](#) 

[Answer](#)  by [quassnoi](#) 

```
SELECT  (  
    SELECT COUNT(*)  
    FROM tab1  
) AS count1,  
(  
    SELECT COUNT(*)  
    FROM tab2  
) AS count2  
FROM dual
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: How do I spool to a CSV formatted file using SQLPLUS?](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

I want to extract some queries to a CSV output format. Unfortunately, I can't use any fancy SQL client or any language to do it. I must use SQLPLUS.

How do I do it?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [daniel-c.-sobral](#) 

[Answer](#)  by [gabe](#) 

You could also use the following, although it does introduce spaces between fields.

[Skip code block](#)

```
set colsep ,      --separate columns with a comma  
set pagesize 0   --No header rows  
set trimspool on-remove trailing blanks  
set headsep off  --this may or may not be useful...depends on your headings.  
set linesize X    --X should be the sum of the column widths  
set numw X       --X should be the length you want for numbers (avoid scientific notation on IDs)  
  
spool myfile.csv  
  
select table_name, tablespace_name  
  from all_tables  
 where owner = 'SYS'  
   and tablespace_name is not null;
```

Output will be like:

TABLE_PRIVILEGE_MAP	, SYSTEM
SYSTEM_PRIVILEGE_MAP	, SYSTEM
STMT_AUDIT_OPTION_MAP	, SYSTEM
DUAL	, SYSTEM
...	

This would be a lot less tedious than typing out all of the fields and concatenating them with the commas. You could follow up with a simple sed script to remove whitespace that appears before a comma, if you wanted.

Something like this might work...(my sed skills are very rusty, so this will likely need work)

```
sed 's/\s+,/,/' myfile.csv
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Oracle Differences between NVL and Coalesce](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

Are there non obvious differences between NVL and Coalesce in Oracle?

The obvious differences are that coalesce will return the first non null item in its parameter list whereas nvl only takes two parameters and returns the first if it is not null, otherwise it returns the second.

It seems that NVL may just be a ‘Base Case’ version of coalesce.

Am I missing something?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [tom-hubbard](#)

[Answer](#) by [quassnoi](#)

COALESCE is more modern function that is a part of ANSI-92 standard.

NVL is Oracle specific, it was introduced in 80’s before there were any standards.

In case of two values, they are synonyms.

However, they are implemented differently.

NVL always evaluates both arguments, while COALESCE stops evaluation whenever it finds first non-NUL:

```
SELECT SUM(val)
FROM (
  SELECT NVL(1, LENGTH(RAWTOHEX(SYS_GUID()))) AS val
  FROM dual
  CONNECT BY
    level <= 10000
)
```

This runs for almost 0.5 seconds, since it generates SYS_GUID()’s, despite 1 being not a NULL.

```
SELECT SUM(val)
FROM (
  SELECT COALESCE(1, LENGTH(RAWTOHEX(SYS_GUID()))) AS val
  FROM dual
  CONNECT BY
    level <= 10000
)
```

This understands that 1 is not a NULL and does not evaluate the second argument.

SYS_GUID’s are not generated and the query is instant.

[Answer](#) by [gary-myers](#)

NVL will do an implicit conversion to the datatype of the first parameter, so the following does not error

```
select nvl('a',sysdate) from dual;
```

COALESCE expects consistent datatypes.

```
select coalesce('a',sysdate) from dual;
```

will throw a ‘inconsistent datatype error’

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: How can I find which tables reference a given table in Oracle SQL Developer?](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#)), [foreign-keys](#) ([Prev Q](#)) ([Next Q](#))

In [Oracle SQL Developer](#), if I’m viewing the information on a table, I can view the constraints, which let me see the foreign keys (and thus which tables are referenced by this table), and I can view the dependencies to see what packages and such reference the table. But I’m not sure how to find which tables reference the table.

For example, say I’m looking at the emp table. There is another table emp_dept which captures which employees work in which departments, which references the emp table through emp_id, the primary key of the emp table. Is there a way (through some UI element in the program, not through SQL) to find that the emp_dept table references the emp table, without me having to know that the emp_dept table exists?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#)), [foreign-keys](#) ([Prev Q](#)) ([Next Q](#))

User: [rudd-zwolinski](#)

Answer by [ferranb](#)

No. There is no such option available from Oracle SQL Developer.

You have to execute a query by hand or use other tool (For instance [PLSQL Developer](#) has such option). The following SQL is that one used by PLSQL Developer:

[Skip code block](#)

```
select table_name, constraint_name, status, owner
from all_constraints
where r_owner = :r_owner
and constraint_type = 'R'
and r_constraint_name in
(
  select constraint_name from all_constraints
  where constraint_type in ('P', 'U')
  and table_name = :r_table_name
  and owner = :r_owner
)
order by table_name, constraint_name
```

Where r_owner is the table which you are looking for references.

Be careful because on the reports tab of Oracle SQL Developer there is the option “All tables / Dependencies” this is from [ALL_DEPENDENCIES](#) which refers to “dependencies between procedures, packages, functions, package bodies, and triggers accessible to the current user, including dependencies on views created without any

database links.”. Then, this report have no value for your question.

[Answer](#) by [junaling](#)

To add this to SQL Developer as an extension do the following:

1. Save the below code into an xml file (e.g. fk_ref.xml):

[Skip code block](#)

```
<items>
  <item type="editor" node="TreeNode" vertical="true">
    <title><![CDATA[FK References]]></title>
    <query>
      <sql>
        <![CDATA[select a.owner,
          a.table_name,
          a.constraint_name,
          a.status
        from all_constraints a
        where a.constraint_type = 'R'
        and exists(
          select 1
          from all_constraints
          where constraint_name=a.r_constraint_name
          and constraint_type in ('P', 'U')
          and table_name = :OBJECT_NAME
          and owner = :OBJECT_OWNER)
        order by table_name, constraint_name]]>
      </sql>
    </query>
  </item>
</items>
```

2. Add the extension to SQL Developer:

- Tools > Preferences
- Database > User Defined Extensions
- Click “Add Row” button
- In Type choose “EDITOR”, Location is where you saved the xml file above
- Click “Ok” then restart SQL Developer

3. Navigate to any table and you should now see an additional tab next to SQL one, labelled FK References, which displays the new FK information.

4. Reference

- <http://www.oracle.com/technetwork/issue-archive/2007/07-jul/o47sql-086233.html>

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#)), [foreign-keys](#) ([Prev Q](#)) ([Next Q](#))

Q: Oracle: If Table Exists

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

I’m writing some migration scripts for an Oracle database, and was hoping Oracle had something similar to MySQL’s IF EXISTS construct.

Specifically, whenever I want to drop a table in MySQL, I do something like

```
DROP TABLE IF EXISTS `table_name`;
```

This way, if the table doesn't exist, the DROP doesn't produce an error, and the script can continue.

Does Oracle have a similar mechanism? I realize I could use the following query to check if a table exists or not

```
SELECT * FROM dba_tables where table_name = 'table_name';
```

but the syntax for tying that together with a DROP is escaping me.

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [alan-storm](#) 

[Answer](#)  by [jeffrey-kemp](#) 

The best and most efficient way is to catch the “table not found” exception: this avoids the overhead of checking if the table exists twice; and doesn't suffer from the problem that if the DROP fails for some other reason (that might be important) the exception is still raised to the caller:

```
BEGIN
  EXECUTE IMMEDIATE 'DROP TABLE mytable';
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -942 THEN
      RAISE;
    END IF;
END;
```

ADDENDUM For reference, here are the equivalent blocks for other object types:

Sequence

```
BEGIN
  EXECUTE IMMEDIATE 'DROP SEQUENCE ' || sequence_name;
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -2289 THEN
      RAISE;
    END IF;
END;
```

View

```
BEGIN
  EXECUTE IMMEDIATE 'DROP VIEW ' || view_name;
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -942 THEN
      RAISE;
    END IF;
END;
```

Trigger

```
BEGIN
  EXECUTE IMMEDIATE 'DROP TRIGGER ' || trigger_name;
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -4080 THEN
      RAISE;
```

```
    END IF;
END;
```

Index

```
BEGIN
    EXECUTE IMMEDIATE 'DROP INDEX ' || index_name;
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -1418 THEN
            RAISE;
        END IF;
END;
```

Column

```
BEGIN
    EXECUTE IMMEDIATE 'ALTER TABLE ' || table_name
                    || ' DROP COLUMN ' || column_name;
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -904 THEN
            RAISE;
        END IF;
END;
```

Database Link

```
BEGIN
    EXECUTE IMMEDIATE 'DROP DATABASE LINK ' || dblink_name;
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -2024 THEN
            RAISE;
        END IF;
END;
```

Materialized View

```
BEGIN
    EXECUTE IMMEDIATE 'DROP MATERIALIZED VIEW ' || mview_name;
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -12003 THEN
            RAISE;
        END IF;
END;
```

Type

```
BEGIN
    EXECUTE IMMEDIATE 'DROP TYPE ' || type_name;
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -4043 THEN
            RAISE;
        END IF;
END;
```

Constraint

```
BEGIN
    EXECUTE IMMEDIATE 'ALTER TABLE ' || table_name
                    || ' DROP CONSTRAINT ' || constraint_name;
EXCEPTION
    WHEN OTHERS THEN
        IF SQLCODE != -2443 THEN
            RAISE;
        END IF;
END;
```

Scheduler Job

```
BEGIN
  DBMS_SCHEDULER.drop_job(job_name);
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -27475 THEN
      RAISE;
    END IF;
END;
```

User / Schema

```
BEGIN
  EXECUTE IMMEDIATE 'DROP USER ' || user_name;
  /* you may or may not want to add CASCADE */
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -1918 THEN
      RAISE;
    END IF;
END;
```

Package

```
BEGIN
  EXECUTE IMMEDIATE 'DROP PACKAGE ' || package_name;
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -4043 THEN
      RAISE;
    END IF;
END;
```

Procedure

```
BEGIN
  EXECUTE IMMEDIATE 'DROP PROCEDURE ' || procedure_name;
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -4043 THEN
      RAISE;
    END IF;
END;
```

Function

```
BEGIN
  EXECUTE IMMEDIATE 'DROP FUNCTION ' || function_name;
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -4043 THEN
      RAISE;
    END IF;
END;
```

[Answer](#)  by [marius-burz](#) 

```
declare
  c int;
begin
  select count(*) into c from user_tables where table_name = upper('table_name');
  if c = 1 then
    execute immediate 'drop table table_name';
  end if;
end;
```

That's for checking whether a table in the current schema exists. For checking whether a given table already exists in a different schema, you'd have to use `all_tables` instead of

user_tables and add the condition all_tables.owner = upper('schema_name')

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to do top 1 in Oracle?](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

How to do this:

```
select top 1 Fname from MyTbl
```

In Oracle 11g?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

User: [gold](#) 

[Answer](#)  by [mcpeterson](#) 

If you want just a first selected row you can:

```
select fname from MyTbl where rownum = 1
```

you can also use analytic functions to order and take the top x

```
select max(fname) over (rank() order by some_factor) from MyTbl
```

[Answer](#)  by [damian-leszczyński–vash](#) 

```
SELECT *  
  FROM (SELECT * FROM MyTbl ORDER BY Fname )  
 WHERE ROWNUM = 1;
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#))

[Q: Case insensitive searching in Oracle](#)

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#)), [case-sensitive](#) ([Prev Q](#)) ([Next Q](#))

The default behaviour of LIKE and the other comparison operators, = etc is case-sensitive.

Is it possible make them case-insensitive?

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#)), [case-sensitive](#) ([Prev Q](#)) ([Next Q](#))

User: [sergionni](#) 

[Answer](#)  by [ben](#) 

There are 3 main ways to perform a case-insensitive search in Oracle without using full-text indexes.

Ultimately what method you choose is dependent on your individual circumstances; the

main thing to remember is that to improve performance you must index correctly for case-insensitive searching.

1. Case your column and your string identically.

You can force all your data to be the same case by using [UPPER\(\)](#) or [LOWER\(\)](#):

```
select * from my_table where upper(column_1) = upper('my_string');
```

or

```
select * from my_table where lower(column_1) = lower('my_string');
```

If `column_1` is not indexed on `upper(column_1)` or `lower(column_1)`, as appropriate, this may force a full table scan. In order to avoid this you can create a [function-based index](#).

```
create index my_index on my_table ( lower(column_1) );
```

If you're using LIKE then you have to concatenate a % around the string you're searching for.

```
select * from my_table where lower(column_1) LIKE lower('my_string') || '%';
```

[This SQL Fiddle](#) demonstrates what happens in all these queries. Note the Explain Plans, which indicate when an index is being used and when it isn't.

2. Use regular expressions.

From Oracle 10g onwards [REGEXP_LIKE\(\)](#) is available. You can specify the `_match_parameter_` 'i', in order to perform case-insensitive searching.

In order to use this as an equality operator you must specify the start and end of the string, which is denoted by the carat and the dollar sign.

```
select * from my_table where regexp_like(column_1, '^my_string$', 'i');
```

In order to perform the equivalent of LIKE, these can be removed.

```
select * from my_table where regexp_like(column_1, 'my_string', 'i');
```

Be careful with this as your string may contain characters that will be interpreted differently by the regular expression engine.

[This SQL Fiddle](#) shows you the same example output except using REGEXP_LIKE().

3. Change it at the session level.

The [NLS_SORT](#) parameter governs the collation sequence for ordering and the various comparison operators, including = and LIKE. You can specify a binary, case-insensitive, sort by altering the session. This will mean that every query performed in that session will perform case-insensitive parameters.

```
alter session set nls_sort=BINARY_CI
```

There's plenty of additional information around [linguistic sorting and string searching](#) if you want to specify a different language, or do an accent-insensitive search using

BINARY_AI.

You will also need to change the [NLS_COMP](#) parameter; to quote:

The exact operators and query clauses that obey the NLS_SORT parameter depend on the value of the NLS_COMP parameter. If an operator or clause does not obey the NLS_SORT value, as determined by NLS_COMP, the collation used is BINARY.

The default value of NLS_COMP is BINARY; but, LINGUISTIC specifies that Oracle should pay attention to the value of NLS_SORT:

Comparisons for all SQL operations in the WHERE clause and in PL/SQL blocks should use the linguistic sort specified in the NLS_SORT parameter. To improve the performance, you can also define a linguistic index on the column for which you want linguistic comparisons.

So, once again, you need to alter the session

```
alter session set nls_comp=LINGUISTIC
```

As noted in the documentation you may want to create a [linguistic index](#) to improve performance

```
create index my_linguistc_index on my_table
  (NLSSORT(column_1, 'NLS_SORT = BINARY_CI'));
```

Tags: [oracle](#) ([Prev Q](#)) ([Next Q](#)), [case-sensitive](#) ([Prev Q](#)) ([Next Q](#))

[Q: Oracle SQL: Update a table with data from another table](#)

Tags: [oracle](#) ([Prev Q](#))

Table 1:

id	name	desc
1	a	abc
2	b	def
3	c	adf

Table 2:

id	name	desc
1	x	123
2	y	345

In oracle SQL, how do I run an **sql update** query that can update Table 1 with Table 2's name and desc using the same id? So the end result I would get is

Table 1:

id	name	desc
1	x	123
2	y	345
3	c	adf

Question is taken from [sql update query with data from another table](#), but specifically for oracle SQL.

Tags: [oracle](#) ([Prev Q](#))

User: [muhd](#)

[Answer](#) by [justin-cave](#)

This is called a correlated update

```
UPDATE table1 t1
  SET (name, desc) = (SELECT t2.name, t2.desc
                        FROM table2 t2
                           WHERE t1.id = t2.id)
 WHERE EXISTS (
   SELECT 1
     FROM table2 t2
    WHERE t1.id = t2.id )
```

Assuming the join results in a key-preserved view, you could also

[Skip code block](#)

```
UPDATE (SELECT t1.id,
              t1.name name1,
              t1.desc desc1,
              t2.name name2,
              t2.desc desc2
            FROM table1 t1,
                 table2 t2
           WHERE t1.id = t2.id)
  SET name1 = name2,
      desc1 = desc2
```

Tags: [oracle](#) ([Prev Q](#))

SQL Server 2008

[Skip to questions,](#)

Wiki by user [sayak-banerjee](#) 

[SQL Server 2008](#)  (codename Katmai, version number 10.00) was released on August 6, 2008 and aims to make data management self-tuning, self organizing, and self maintaining with the development of SQL Server Always On technologies, to provide near-zero downtime. SQL Server 2008 also includes support for structured and semi-structured data, including digital media formats for pictures, audio, video and other multimedia data. In current versions, such multimedia data can be stored as BLOBS (binary large objects), but they are generic bitstreams. Intrinsic awareness of multimedia data will allow specialized functions to be performed on them. According to Paul Flessner, senior Vice President, Server Applications, Microsoft Corp., SQL Server 2008 can be a data storage backend for different varieties of data: XML, email, time/calendar, file, document, spatial, etc as well as perform search, query, analysis, sharing, and synchronization across all data types.

Other new data types include specialized date and time types and a Spatial data type for location-dependent data. Better support for unstructured and semi-structured data is provided using the new FILESTREAM data type, which can be used to reference any file stored on the file system. Structured data and metadata about the file is stored in SQL Server database, whereas the unstructured component is stored in the file system. Such files can be accessed both via Win32 file handling APIs as well as via SQL Server using T-SQL; doing the latter accesses the file data as a BLOB. Backing up and restoring the database backs up or restores the referenced files as well. SQL Server 2008 also natively supports hierarchical data, and includes T-SQL constructs to directly deal with them, without using recursive queries.

The Full-text search functionality has been integrated with the database engine. According to a Microsoft technical article, this simplifies management and improves performance. Spatial data will be stored in two types. A “Flat Earth” (GEOMETRY or planar) data type represents geospatial data which has been projected from its native, spherical, coordinate system into a plane. A “Round Earth” data type (GEOGRAPHY) uses an ellipsoidal model in which the Earth is defined as a single continuous entity which does not suffer from the singularities such as the international dateline, poles, or map projection zone “edges”. Approximately 70 methods are available to represent spatial operations for the Open Geospatial Consortium Simple Features for SQL, Version 1.1.

SQL Server includes better compression features, which also helps in improving scalability. It enhanced the indexing algorithms and introduced the notion of filtered indexes. It also includes Resource Governor that allows reserving resources for certain users or workflows. It also includes capabilities for transparent encryption of data (TDE) as well as compression of backups. SQL Server 2008 supports the ADO.NET Entity Framework and the reporting tools, replication, and data definition will be built around the Entity Data Model. SQL Server Reporting Services will gain charting capabilities from the

integration of the data visualization products from Dundas Data Visualization, Inc., which was acquired by Microsoft. On the management side, SQL Server 2008 includes the Declarative Management Framework which allows configuring policies and constraints, on the entire database or certain tables, declaratively. The version of SQL Server Management Studio included with SQL Server 2008 supports IntelliSense for SQL queries against a SQL Server 2008 Database Engine. SQL Server 2008 also makes the databases available via Windows PowerShell providers and management functionality available as Cmdlets, so that the server and all the running instances can be managed from Windows PowerShell.

[**Source**](#) 

Questions

Q: ALTER DATABASE failed because a lock could not be placed on database

Tags: [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#)), [ssms](#) ([Prev Q](#)) ([Next Q](#))

I need to restart a database because some processes are not working. My plan is to take it offline and back online again.

I am trying to do this in Sql Server Management Studio 2008:

```
use master;
go
alter database qcvalues
set single_user
with rollback immediate;
alter database qcvalues
set multi_user;
go
```

I am getting these errors:

```
Msg 5061, Level 16, State 1, Line 1
ALTER DATABASE failed because a lock could not be placed on database 'qcvalues'. Try again later.
Msg 5069, Level 16, State 1, Line 1
ALTER DATABASE statement failed.
Msg 5061, Level 16, State 1, Line 4
ALTER DATABASE failed because a lock could not be placed on database 'qcvalues'. Try again later.
Msg 5069, Level 16, State 1, Line 4
ALTER DATABASE statement failed.
```

What am I doing wrong?

Tags: [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#)), [ssms](#) ([Prev Q](#)) ([Next Q](#))

User: [joe-skeet](#)

[Answer](#) by [bobs](#)

After you get the error, run

```
EXEC sp_who2
```

Look for the database in the list. It's possible that a connection was not terminated. If you find any connections to the database, run

```
KILL <SPID>
```

where <SPID> is the SPID for the sessions that are connected to the database.

Try your script after all connections to the database are removed.

Unfortunately, I don't have a reason why you're seeing the problem, but here is a link that shows that the problem has occurred elsewhere.

<http://www.geakeit.co.uk/2010/12/11/sql-take-offline-fails-alter-database-failed-because-a-lock-could-not-error-5061/>

Tags: [sql-server-2008](#) ([Prev Q](#)) ([Next Q](#)), [ssms](#) ([Prev Q](#)) ([Next Q](#))



[Q: How to turn IDENTITY_INSERT on and off using SQL Server 2008?](#)

Tags: [sql-server-2008](#) ([Prev Q](#))

Why am I getting an error doing an insert when IDENTITY_INSERT is set to OFF?

How do I turn it on properly in SQL Server 2008? Is it by using SQL Server Management Studio?

I have run this query:

```
SET IDENTITY_INSERT Database. dbo. Baskets ON
```

Then I got the message back in the console that the Command(s) completed successfully. However when I run the application, it still gives me the error shown below:

```
Cannot insert explicit value for identity column in table 'Baskets' when  
IDENTITY_INSERT is set to OFF.
```

Tags: [sql-server-2008](#) ([Prev Q](#))

User: [beginner](#)

[Answer](#) by [gbn](#)

Via SQL as per [MSDN](#)

```
SET IDENTITY_INSERT sometableWithIdentity ON  
INSERT sometableWithIdentity (IdentityColumn, col2, col3, ...)  
VALUES (AnIdentityValue, col2value, col3value, ...)  
SET IDENTITY_INSERT sometableWithIdentity OFF
```

The complete error message tells you *exactly* what is wrong...

Cannot insert explicit value for identity column in table ‘sometableWithIdentity’ when IDENTITY_INSERT is set to OFF.

Tags: [sql-server-2008](#) ([Prev Q](#))

PostgreSQL

[Skip to questions](#),

Wiki by user [frank-heikens](#)

PostgreSQL (often Postgres, never Postgre), is an object-relational database management system (ORDBMS) available for all major operating systems. It is released under the PostgreSQL License, which is an MIT-style license, and is thus free and open source software. PostgreSQL is developed by the PostgreSQL Global Development Group, consisting of volunteers employed and supervised by companies such as Red Hat and EnterpriseDB.

PostgreSQL is pronounced as “post-grez-q-l”. Postgres is pronounced as “post-grez”.

Numerous forks of Postgres exist for specialized tasks, such as Greenplum Database, Amazon Redshift, ParAccel, Postgres-XC, Postgres-XL, PPAS, etc. Their features and syntax differ from stock PostgreSQL. Declare what you are using and add a tag.

Latest Release: [PostgreSQL 9.5](#)

PostgreSQL Features

- [Implements](#) the majority of the SQL:2011 standard
- [ACID](#)-compliant
- Fully [transactional](#) (including all DDL statements)
- Extensible data types, operators, indexing, functions, aggregates, procedural languages
- Large number of extensions both [bundled](#) and [supplied by third parties](#)
- [Supported operating systems](#) include: Linux, FreeBSD, Solaris, MS Windows, Mac OS X
- Detailed, comprehensive and clear [documentation](#)

How to ask good questions

For performance questions see the [tag info](#) of [postgresql-performance](#).

For questions related to a specific version of PostgreSQL please make sure to include the version tag([postgresql-9.4](#), [postgresql-9.2](#), [postgresql-8.4](#) etc.).

Questions concerning administration or advanced features are best directed to [dba.stackexchange.com](#).

The [Guide to Reporting Problems on the PostgreSQL wiki](#) is useful reading.

- Show your PostgreSQL version from `SELECT version()`.
- Include the full text of relevant error messages, SQL queries, etc.
- Where appropriate supply a test case with `CREATE TABLE` and `INSERT` statements for

sample data and a hand-crafted “expected results” table. [SQLFiddle](#) is useful for this.

- If you've asked previous, related questions, add a link.
- If you're re-posting a mailing list question here, link to the page in the [mailing list archives](#). Or vice versa, when posting an SO question to the mailing list.
- Know about PostgreSQL current settings/backend config using:

```
SELECT name, current_setting(name), source FROM pg_settings  
WHERE source NOT IN ('default', 'override');
```

Resources

- [The current manual](#)
- [PostgreSQL Wiki](#)
- [PostgreSQL GUI Tools](#)
- [Advantages of PostgreSQL](#)
- [PostgreSQL on Wikipedia](#)
- [Mailing list archives](#)
- [PostgreSQL IRC channels](#)
- [Books](#)

Useful links

- [PostgreSQL Tutorials](#)
- [PostgreSQL Online Tutorial](#)
- [PostgreSQL Online References](#)
- [PostgreSQL SQL Tricks](#)
- [PostgreSQL for Beginners](#)

Dedicated chat room for [PostgreSQL](#)

Related tags :

[databaserelational-database](#)

Questions

[Q: How to concatenate strings of a string field in a PostgreSQL ‘group by’ query?](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#)), [string-concatenation](#) ([Prev Q](#)) ([Next Q](#))

I am looking for a way to concatenate the strings of a field within a group by query. So for example, I have a table:

ID	COMPANY_ID	EMPLOYEE
1	1	Anna
2	1	Bill
3	2	Carol
4	2	Dave

and I wanted to group by company_id to get something like:

COMPANY_ID	EMPLOYEE
1	Anna, Bill
2	Carol, Dave

There is a built-in function in mySQL to do this [group_concat](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#)), [string-concatenation](#) ([Prev Q](#)) ([Next Q](#))

User: [guy-c](#)

[Answer](#) by [neall](#)

Update as of PostgreSQL 9.0:

Recent versions of Postgres (since late 2010) have the [string_agg\(expression, delimiter\)](#) function that will do exactly what the question asked for, even letting you specify the delimiter string:

```
SELECT company_id, string_agg(employee, ', ')
FROM mytable
GROUP BY company_id;
```

Update as of PostgreSQL 8.4:

PostgreSQL 8.4 (in 2009) introduced [the aggregate function array_agg\(expression\)](#) which concatenates the values into an array. Then `array_to_string()` can be used to give the desired result:

```
SELECT company_id, array_to_string(array_agg(employee), ', ')
FROM mytable
GROUP BY company_id;
```

Original Answer (for pre-8.4 PostgreSQL):

There is no built-in aggregate function to concatenate strings. It seems like this would be

needed, but it's not part of the default set. A web search however reveals some manual implementations [the same example](#):

```
CREATE AGGREGATE textcat_all(
    basetype   = text,
    sfunc      = textcat,
    stype      = text,
    initcond   = ''
);
```

[Here is the CREATE AGGREGATE documentation.](#)

In order to get the “,” inserted in between them without having it at the end, you might want to make your own concatenation function and substitute it for the “textcat” above. Here is one I put together but haven't tested (*update:* tested on 8.3.12 and working fine):

```
CREATE FUNCTION commacat(acc text, instr text) RETURNS text AS $$  
BEGIN  
    IF acc IS NULL OR acc = '' THEN  
        RETURN instr;  
    ELSE  
        RETURN acc || ', ' || instr;  
    END IF;  
END;  
$$ LANGUAGE plpgsql;
```

Note: The function above will output a comma even if the value in the row is null/empty, which outputs:

```
a, b, c, , e, , g
```

If you would prefer to remove extra commas to output:

```
a, b, c, e, g
```

just add an ELSIF check to the function:

[Skip code block](#)

```
CREATE FUNCTION commacat_ignore_nulls(acc text, instr text) RETURNS text AS $$  
BEGIN  
    IF acc IS NULL OR acc = '' THEN  
        RETURN instr;  
    ELSIF instr IS NULL OR instr = '' THEN  
        RETURN acc;  
    ELSE  
        RETURN acc || ', ' || instr;  
    END IF;  
END;  
$$ LANGUAGE plpgsql;
```

[Answer](#) by [markus-döring](#)

How about using Postgres built-in array functions? At least on 8.4 this works out of the box:

```
SELECT company_id, array_to_string(array_agg(employee), ',')  
FROM mytable  
GROUP BY company_id;
```

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#)), [string-concatenation](#) ([Prev Q](#)) ([Next Q](#))

Q: How do I (or can I) SELECT DISTINCT on multiple columns?

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#)), [duplicates](#) ([Prev Q](#)) ([Next Q](#))

I need to retrieve all rows from a table where 2 columns combined are all different. So I want all the sales that do not have any other sales that happened on the same day for the same price. The sales that are unique based on day and price will get updated to an active status.

So I'm thinking:

```
UPDATE sales
SET status = 'ACTIVE'
WHERE id IN (SELECT DISTINCT (saleprice, saledate), id, count(id)
              FROM sales
              HAVING count = 1)
```

But my brain hurts going any farther than that.

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#)), [duplicates](#) ([Prev Q](#)) ([Next Q](#))

User: [sheats](#)

[Answer](#) by [joel-coehoorn](#)

```
SELECT DISTINCT a,b,c FROM t
```

is *roughly* equivalent to:

```
SELECT a,b,c FROM t GROUP BY a,b,c
```

It's a good idea to get used to the GROUP BY syntax, as it's more powerful.

For your query, I'd do it like this:

[Skip code block](#)

```
UPDATE sales
SET status='ACTIVE'
WHERE id IN
(
    SELECT id
    FROM sales S
    INNER JOIN
    (
        SELECT saleprice, saledate
        FROM sales
        GROUP BY saleprice, saledate
        HAVING COUNT(*) = 1
    ) T
    ON S.saleprice=T.saleprice AND S.saledate=T.saledate
)
```

[Answer](#) by [erwin-brandstetter](#)

If you put together the answers so far, clean up and improve, you would arrive at this superior query:

```
UPDATE sales
SET status = 'ACTIVE'
WHERE (saleprice, saledate) IN (
    SELECT saleprice, saledate
```

```
FROM sales
GROUP BY saleprice, saledate
HAVING count(*) = 1
);
```

Which is *much* faster than either of them. Nukes the performance of the currently accepted answer by factor 10 - 15 (in my tests on PostgreSQL 8.4 and 9.1).

But this is still far from optimal. Use a [NOT EXISTS](#) (anti-)semi-join for even better performance. `EXISTS` is standard SQL, has been around forever (at least since PostgreSQL 7.2, long before this question was asked) and fits the presented requirements perfectly:

[Skip code block](#)

```
UPDATE sales s
SET status = 'ACTIVE'
WHERE NOT EXISTS (
    SELECT 1
    FROM sales s1
    WHERE s.saleprice = s1.saleprice
    AND s.saledate = s1.saledate
    AND s.id > s1.id           --except for row itself
)
AND s.status IS DISTINCT FROM 'ACTIVE'; --avoid empty updates. see below
```

[SQL Fiddle](#)

Unique key to identify row

If you don't have a primary or unique key for the table (`id` in the example), you can [substitute with the system column `ctid`](#) for the purpose of this query:

```
AND s.ctid <> s1.ctid
```

Every table should have a primary key. Add one if you didn't have one, yet. I suggest a [serial](#).

How is this faster?

The subquery in the `EXISTS` (anti-)semi-join can stop evaluating as soon as the first dupe is found (no point in looking further). For a base table with few duplicates this is only mildly more efficient. With lots of duplicates this becomes *way* more efficient.

Exclude empty updates

If some or many rows already have `status = 'ACTIVE'`, your update would not change anything, but still insert a new row version at full cost (minor exceptions apply). Normally, you do not want this. Add another `WHERE` condition like demonstrated above to make this even faster:

If `status` is defined `NOT NULL`, you can simplify to:

```
AND status <> 'ACTIVE';
```

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#)), [duplicates](#) ([Prev Q](#)) ([Next Q](#))

[Q: Insert, on duplicate update in PostgreSQL?](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

Several months ago I learned from an answer on Stack Overflow how to perform multiple updates at once in MySQL using the following syntax:

```
INSERT INTO table (id, field, field2) VALUES (1, A, X), (2, B, Y), (3, C, Z)
ON DUPLICATE KEY UPDATE field=VALUES(Col1), field2=VALUES(Col2);
```

I've now switched over to PostgreSQL and apparently this is not correct. It's referring to all the correct tables so I assume it's a matter of different keywords being used but I'm not sure where in the PostgreSQL documentation this is covered.

To clarify, I want to insert several things and if they already exist to update them.

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

User: [teifion](#)

Answer by [stephen-denne](#)

PostgreSQL starting from version 9.5 will have [UPSERT](#) syntax.

Searching postgresql's email group archives for "upsert" leads to finding [an example of doing what you possibly want to do, in the manual](#):

Example 38-2. Exceptions with UPDATE/INSERT

This example uses exception handling to perform either UPDATE or INSERT, as appropriate:

[skip code block](#)

```
CREATE TABLE db (a INT PRIMARY KEY, b TEXT);

CREATE FUNCTION merge_db(key INT, data TEXT) RETURNS VOID AS
$$
BEGIN
    LOOP
        --first try to update the key
        --note that "a" must be unique
        UPDATE db SET b = data WHERE a = key;
        IF found THEN
            RETURN;
        END IF;
        --not there, so try to insert the key
        --if someone else inserts the same key concurrently,
        --we could get a unique-key failure
        BEGIN
            INSERT INTO db(a,b) VALUES (key, data);
            RETURN;
        EXCEPTION WHEN unique_violation THEN
            --do nothing, and loop to try the UPDATE again
        END;
    END LOOP;
END;
$$
LANGUAGE plpgsql;

SELECT merge_db(1, 'david');
SELECT merge_db(1, 'dennis');
```

There's possibly an example of how to do this in bulk, using CTEs in 9.1 and above, in the [hackers mailing list](#):

```
WITH foos AS (SELECT (UNNEST(%foo[])).*)
updated as (UPDATE foo SET foo.a = foos.a... RETURNING foo.id)
INSERT INTO foo SELECT foos.* FROM foos LEFT JOIN updated USING(id)
WHERE updated.id IS NULL;
```

See [a horse with no name's answer](#) for a clearer example.

[Answer](#) by [bovine](#)

Warning: this is not safe if executed from multiple sessions at the same time (see caveats below).

Another clever way to do an “UPSERT” in postgresql is to do two sequential UPDATE/INSERT statements that are each designed to succeed or have no effect.

```
UPDATE table SET field='C', field2='Z' WHERE id=3;
INSERT INTO table (id, field, field2)
  SELECT 3, 'C', 'Z'
    WHERE NOT EXISTS (SELECT 1 FROM table WHERE id=3);
```

The UPDATE will succeed if a row with “id=3” already exists, otherwise it has no effect.

The INSERT will succeed only if row with “id=3” does not already exist.

You can combine these two into a single string and run them both with a single SQL statement execute from your application. Running them together in a single transaction is highly recommended.

This works very well when run in isolation or on a locked table, but is subject to race conditions that mean it might still fail with duplicate key error if a row is inserted concurrently, or might terminate with no row inserted when a row is deleted concurrently. A SERIALIZABLE transaction on PostgreSQL 9.1 or higher will handle it reliably at the cost of a very high serialization failure rate, meaning you'll have to retry a lot. See [why is upsert so complicated](#), which discusses this case in more detail.

This approach is also [subject to lost updates in read committed isolation unless the application checks the affected row counts and verifies that either the insert or the update affected a row](#).

[Answer](#) by [a horse with no name](#)

With PostgreSQL 9.1 this can be achieved using a writeable CTE ([common table expression](#)):

[Skip code block](#)

```
WITH new_values (id, field1, field2) as (
  values
    (1, 'A', 'X'),
    (2, 'B', 'Y'),
    (3, 'C', 'Z')
),
upsert as
(
  update mytable m
    set field1 = nv.field1,
```

```

        field2 = nv.field2
    FROM new_values nv
    WHERE m.id = nv.id
    RETURNING m.*
)
INSERT INTO mytable (id, field1, field2)
SELECT id, field1, field2
FROM new_values
WHERE NOT EXISTS (SELECT 1
                   FROM upsert up
                   WHERE up.id = new_values.id)

```

See these blog entries:

- [Upserting via Writeable CTE](#)
- [WAITING FOR 9.1 – WRITABLE CTE](#)
- [WHY IS UPSERT SO COMPLICATED?](#)

Note that this solution does **not** prevent a unique key violation but it is not vulnerable to lost updates.

See the [follow up by Craig Ringer on dba.stackexchange.com](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Save PL/pgSQL output from PostgreSQL to a CSV file](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

What is the easiest way to save PL/pgSQL output from a PostgreSQL database to a CSV file?

I'm using PostgreSQL 8.4 with pgAdmin III and PSQL plugin where I run queries from.

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

User: [hoff](#)

[Answer](#) by [imsop](#)

If you want something easy to re-use or automate, you can use Postgres's built in [COPY](#) command. e.g.

```
Copy (Select * From foo) To '/tmp/test.csv' With csv;
```

This approach runs entirely on the remote server - it can't write to your local PC. It also needs to be run as a Postgres "superuser" (normally called "root") because Postgres can't stop it doing nasty things with that machine's local filesystem.

That doesn't actually mean you have to be connected as a superuser (automating that would be a security risk of a different kind), because you can use [the SECURITY DEFINER option to CREATE FUNCTION](#) to make a function which *runs as though you were a superuser*.

The crucial part is that your function is there to perform additional checks, not just by-pass the security - so you could write a function which exports the exact data you need, or you

could write something which can accept various options as long as they meet a strict whitelist. You need to check two things:

1. Which **files** should the user be allowed to read/write on disk? This might be a particular directory, for instance, and the filename might have to have a suitable prefix or extension.
2. Which **tables** should the user be able to read/write in the database? This would normally be defined by GRANTS in the database, but the function is now running as a superuser, so tables which would normally be “out of bounds” will be fully accessible. You probably don’t want to let someone invoke your function and add rows on the end of your “users” table...

I’ve written [a blog post expanding on this approach](#), including some examples of functions that export (or import) files and tables meeting strict conditions.

The other approach is to **do the file handling on the client side**, i.e. in your application or script. The Postgres server doesn’t need to know what file you’re copying to, it just spits out the data and the client puts it somewhere.

The underlying syntax for this is the `COPY TO STDOUT` command, and graphical tools like pgAdmin will wrap it for you in a nice dialog.

The **psql command-line client** has a special “meta-command” called `\copy`, which takes all the same options as the “real” `COPY`, but is run inside the client:

```
\copy (Select * From foo) To '/tmp/test.csv' With csv
```

Note that there is no terminating `;`, because meta-commands are terminated by newline, unlike SQL commands.

From [the docs](#):

Do not confuse `COPY` with the `psql` instruction `\copy`. `\copy` invokes `COPY FROM STDIN` or `COPY TO STDOUT`, and then fetches/stores the data in a file accessible to the `psql` client. Thus, file accessibility and access rights depend on the client rather than the server when `\copy` is used.

Your application programming language *may* also have support for pushing or fetching the data, but you cannot generally use `COPY FROM STDIN/TO STDOUT` within a standard SQL statement, because there is no way of connecting the input/output stream. PHP’s PostgreSQL handler (*not PDO*) includes very basic [pg_copy_from](#) and [pg_copy_to](#) functions which copy to/from a PHP array, which may not be efficient for large data sets.

[Answer](#) by [sorin](#)

There are several solutions:

1 psql command

```
psql -d dbname -t -A -F"," -c "select * from users" > output.csv
```

This has the big advantage that you can use it via SSH, like `ssh postgres@host` command - enabling you to get

2 postgres copy command

```
COPY (SELECT * from users) To '/tmp/output.csv' With CSV;
```

3 psql interactive (or not)

```
>psql dbname
psql>\f ',' 
psql>\a
psql>\o '/tmp/output.csv'
psql>SELECT * from users;
psql>\q
```

All of them can be used in scripts, but I prefer #1.

4 pgadmin but that's not scriptable.

[Answer](#) by [marcin-wasiluk](#)

In terminal (while connected to the db) set output to the csv file

1) set field separator to ',' by:

```
\f ','
```

2) set output format unaligned:

```
\a
```

3) show only tuples

```
\t
```

4) set output

```
\o '/tmp/your outputFile.csv'
```

5) execute your query

```
select * from YOUR_TABLE
```

6) output

```
\o
```

you will be able to find your csv file in this location

```
cd //tmp
copy it using scp command
```

or edit using nano:

```
nano //tmp/your outputFile.csv
```

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to delete duplicate entries?](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

I have to add a unique constraint to an existing table. This is fine except that the table has millions of rows already, and many of the rows violate the unique constraint I need to add.

What is the fastest approach to removing the offending rows? I have an SQL statement which finds the duplicates and deletes them, but it is taking forever to run. Is there another way to solve this problem? Maybe backing up the table, then restoring after the constraint is added?

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

User: [gjrwebber](#)

[Answer](#) by [just-somebody](#)

For example you could:

```
CREATE TABLE tmp...
INSERT INTO tmp SELECT DISTINCT * FROM t;
DROP TABLE t;
ALTER TABLE tmp RENAME TO t;
```

[Answer](#) by [tim](#)

Some of these approaches seem a little complicated, and I generally do this as:

Given table `table`, want to unique it on `(field1, field2)` keeping the row with the max `field3`:

```
DELETE FROM table USING table alias
 WHERE table.field1 = alias.field1 AND table.field2 = alias.field2 AND
       table.max_field < alias.max_field
```

For example, I have a table, `user_accounts`, and I want to add a unique constraint on `email`, but I have some duplicates. Say also that I want to keep the most recently created one (max id among duplicates).

```
DELETE FROM user_accounts USING user_accounts ua2
 WHERE user_accounts.email = ua2.email AND user_account.id < ua2.id;
```

- Note - `USING` is not standard SQL, it is a PostgreSQL extension (but a very useful one), but the original question specifically mentions PostgreSQL.

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Select first row in each GROUP BY group?](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#)), [sqlite](#) ([Prev Q](#)) ([Next Q](#))

As the title suggests, I'd like to select the first row of each set of rows grouped with a

GROUP BY.

Specifically, if I've got a purchases table that looks like this:

```
SELECT * FROM purchases;
```

id	customer	total
1	Joe	5
2	Sally	3
3	Joe	2
4	Sally	1

I'd like to query for the `id` of the largest purchase (`total`) made by each customer. Something like this:

```
SELECT FIRST(id), customer, FIRST(total)
FROM purchases
GROUP BY customer
ORDER BY total DESC;
```

FIRST(id)	customer	FIRST(total)
1	Joe	5
2	Sally	3

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#)), [sqlite](#) ([Prev Q](#)) ([Next Q](#))

User: [david-wolever](#) 

Answer  by [omg-ponies](#) 

On Oracle 8i+, SQL Server 2005+, PostgreSQL 8.4+, DB2, Firebird 2.1+, Teradata, Sybase, Vertica:

[Skip code block](#)

```
WITH summary AS (
    SELECT p.id,
           p.customer,
           p.total,
           ROW_NUMBER() OVER(PARTITION BY p.customer
                             ORDER BY p.total DESC) AS rk
      FROM PURCHASES p)
SELECT s.*
  FROM summary s
 WHERE s.rk = 1
```

Supported by any database:

But you need to add logic to break ties:

[Skip code block](#)

```
SELECT MIN(x.id), --change to MAX if you want the highest
       x.customer,
       x.total
  FROM PURCHASES x
 JOIN (SELECT p.customer,
              MAX(total) AS max_total
         FROM PURCHASES p
        GROUP BY p.customer) y
    ON y.customer = x.customer
       AND y.max_total = x.total
 GROUP BY x.customer, x.total
```

[Answer](#) by [erwin-brandstetter](#)

In PostgreSQL this is typically **simpler and faster** (more performance optimization below):

```
SELECT DISTINCT ON (customer)
    id, customer, total
FROM purchases
ORDER BY customer, total DESC, id;
```

Or shorter with ordinal numbers of output columns:

```
SELECT DISTINCT ON (2)
    id, customer, total
FROM purchases
ORDER BY 2, 3 DESC, 1;
```

If total can be NULL (won't hurt either way, but you'll want to match existing indexes):

```
...
ORDER BY customer, total DESC NULLS LAST, id;
```

Major points

- [DISTINCT ON](#) is a PostgreSQL extension of the standard (where only DISTINCT on the whole SELECT list is defined).
- DISTINCT ON can be combined with ORDER BY. Leading expressions of ORDER BY have to match expressions in DISTINCT ON in that order, and you can add additional columns / expressions to pick a particular row from each group of peers. I added id as last item to ORDER BY to break ties:
“Pick the row with the smallest id from each group sharing the highest total.”

If total can be NULL, you *most probably* want the row with the greatest non-null value. Add NULLS LAST like demonstrated above. Details:

- [PostgreSQL sort by datetime asc, null first?](#)
- For more complex requirements (not needed in this simple case):
 - You *don't have to* include any of the columns / expression used in ORDER BY or DISTINCT ON in the SELECT list.
 - You *can* include any other column from the base tables in the SELECT list. This is instrumental in replacing much more complex queries with subqueries and aggregate / window functions.
- I tested with versions 8.3 – 9.5. But the feature has been there at least since version 7.1 (= for ever).

Index

The *perfect* index for the above query would be a [multi-column index](#) spanning all three columns in matching sequence and with matching sort order:

```
CREATE INDEX purchases_3c_idx ON purchases (customer, total DESC, id);
```

May be too specialized for real world applications. But use it if read performance is crucial. If you have `DESC NULLS LAST` in the query, use the same in the index so Postgres knows sort order matches.

Benchmark

I had a simple benchmark here for Postgres 9.1, which was outdated by now (2016). So I ran a new one with a better, reproducible setup for Postgres 9.4 and 9.5 and added the [detailed results in another answer](#).

Effectiveness / Performance optimization

You have to weigh cost and benefit before you create a tailored index for every query. The potential of above index largely depends on **data distribution**.

The index is used because it delivers pre-sorted data, and in Postgres 9.2 or later the query can also benefit from an [index only scan](#) if the width of the index is smaller than the underlying table. The index has to be scanned in its entirety, though.

- For **few rows per customer**, this is very efficient, even more so if you need sorted output anyway. The benefit shrinks with a growing number of rows per customer.
- For **many rows per customer**, the equivalent of a [loose index scan](#) would be much more efficient, but that's not currently implemented in Postgres (up to 9.5). There are **faster query techniques** to substitute for this. In particular, if you have a separate table holding unique customers (which often is the case). But also if you don't:
 - [Optimize GROUP BY query to retrieve latest record per user](#)
 - [Optimize groupwise maximum query](#)
 - [Query last N related rows per row](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [group-by](#) ([Prev Q](#)) ([Next Q](#)), [sqlite](#) ([Prev Q](#)) ([Next Q](#))

Q: What are the pros and cons of performing calculations in sql vs. in your application

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [java](#) ([Next Q](#))

shopkeeper table has following fields:

```
id (bigint), amount (numeric(19,2)), createddate (timestamp)
```

Let's say, I have the above table. I want to get the records for yesterday and generate a report by having the amount printed to cents.

One way of doing is to perform calculations in my java application and execute a simple query

```
Date previousDate ;// $1 calculate in application  
Date todayDate;// $2 calculate in application  
select amount where createddate between $1 and $2
```

and then loop through the records and convert amount to cents in my java application and generate the report

Another way is like performing calculations in sql query itself:

```
select cast(amount * 100 as int) as "Cents"  
from shopkeeper where createddate between date_trunc('day', now()) - interval '1 day' and date_tr
```

and then loop through the records and generate the report

In one way , all my processing is done in java application and a simple query is fired. In other case all the conversions and calculations is done in Sql query.

The above use case is just an example, in a real scenario a table can have many columns that require processing of the similar kind.

Can you please tell me which approach is better in terms of performance and other aspects and why?

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [java](#) ([Next Q](#))

User: [hellojava](#)

[Answer](#) by [marc-gravell](#)

It depends on a lot of factors - but most crucially:

- complexity of calculations (prefer doing complex crunching on an app-server, since that scales *out*; rather than a db server, which scales *up*)
- volume of data (if you need to access/aggregate a lot of data, doing it at the db server will save bandwidth, and disk io if the aggregates can be done inside indexes)
- convenience (sql is not the best language for complex work - especially not great for procedural work, but very good for set-based work; lousy error-handling, though)

As always, if you do bring the data back to the app-server, minimising the columns and

rows will be to your advantage. Making sure the query is tuned and appropriately indexed will help either scenario.

Re your note:

and then loop through the records

Looping through records is almost always the wrong thing to do in sql - writing a set-based operation is preferred.

As a general rule, I prefer to keep the database's job to a minimum "store this data, fetch this data" - however, there are always examples of scenarios where an elegant query at the server can save a lot of bandwidth.

Also consider: if this is computationally expensive, can it be cached somewhere?

If you want an *accurate* "which is better"; code it both ways and compare it (noting that a first draft of either is likely not 100% tuned). But factor in typical usage to that: if, in reality, it is being called 5 times (separately) at once, then simulate that: don't compare just a single "1 of these vs 1 of those".

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [java](#) ([Next Q](#))

Q: How to set auto increment primary key in PostgreSQL?

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

I have a table in PostgreSQL with 22 columns, and I want to add an auto increment primary key.

I tried to create a column called `id` of type BIGSERIAL but pgadmin responded with an error:

```
ERROR: sequence must have same owner as table it is linked to.
```

Does anyone know how to fix this issue? How do I add a create an auto-incrementing primary key in PostgreSQL without recreating the table again?

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

User: [mkn](#) 

[Answer](#)  by [a.h.](#) 

Try this command:

```
ALTER TABLE your_table ADD COLUMN key_column BIGSERIAL PRIMARY KEY;
```

Try it with the same DB-user as the one you have *created* the table.

[Answer](#)  by [eric-leschinski](#) 

Auto incrementing primary key in postgresql:

Step 1, create your table:

```
CREATE TABLE epictable
(
    mytable_key      serial primary key,
    moobars          VARCHAR(40) not null,
    foobars          DATE
);
```

Step 2, insert values into your table like this, notice that `mytable_key` is not specified in the first parameter list, this causes the default sequence to autoincrement.

```
insert into epictable(moobars,foobars) values('delicious moobars','2012-05-01')
insert into epictable(moobars,foobars) values('worldwide interblag','2012-05-02')
```

Step 3, select * from your table:

```
el@voyager$ psql -U pgadmin -d kurz_prod -c "select * from epictable"
```

Step 4, interpret the output:

mytable_key	moobars	foobars
1	delicious moobars	2012-05-01
2	world wide interblags	2012-05-02

(2 rows)

Observe that `mytable_key` column has been auto incremented.

ProTip:

You should always be using a primary key on your table because postgresql internally uses hash table structures to increase the speed of inserts, deletes, updates and selects. If a primary key column (which is forced unique and non-null) is available, it can be depended on to provide a unique seed for the hash function. If no primary key column is available, the hash function becomes inefficient as it selects some other set of columns as a key.

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

Q: Best way to select random rows PostgreSQL

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [random](#) ([Prev Q](#)) ([Next Q](#))

I want a random selection of rows in PostgreSQL, I tried this:

```
select * from table where random() < 0.01;
```

But some other recommend this:

```
select * from table order by random() limit 1000;
```

I have a very large table with 500 Million rows, I want it to be fast.

Which approach is better? What are the differences? What is the best way to select random rows?

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#)), [performance](#) ([Prev Q](#)) ([Next Q](#)), [random](#) ([Prev Q](#)) ([Next Q](#))

User: [nanounanue](#) 

[Answer](#)  by [erwin-brandstetter](#) 

Given your specifications (plus additional info in the comments),

- You have a numeric ID column with only few (or moderately few) gaps.
- Obviously no or few write operations.
- Your ID column has to be indexed! A primary key serves nicely.

The query below does not need a sequential scan of the big table, only an index scan.

First, get estimates for the main query:

```
SELECT count(*) AS ct          -optional  
      , min(id) AS min_id  
      , max(id) AS max_id  
      , max(id) - min(id) AS id_span  
FROM   big;
```

The only possibly expensive part is the `count(*)` (for huge tables). Given above specifications, you don't need it. An estimate will do just fine, available at almost no cost ([detailed explanation here](#) ):

```
SELECT reltuples AS ct FROM pg_class WHERE oid = 'schema_name.big'::regclass;
```

As long as `ct` isn't *much* smaller than `id_span`, the query will outperform other approaches.

[Skip code block](#)

```
WITH params AS (  
    SELECT 1           AS min_id          -minimum id <= current min id  
          , 5100000 AS id_span          -rounded up. (max_id - min_id + buffer)  
    )  
SELECT *  
FROM  (  
    SELECT p.min_id + trunc(random() * p.id_span)::integer AS id  
    FROM   params p  
          , generate_series(1, 1100) g -1000 + buffer  
    GROUP  BY 1                      -trim duplicates  
    ) r  
JOIN   big USING (id)  
LIMIT  1000;                         -trim surplus
```

- Generate random numbers in the `id` space. You have “few gaps”, so add 10 % (enough to easily cover the blanks) to the number of rows to retrieve.
- Each `id` can be picked multiple times by chance (though very unlikely with a big `id` space), so group the generated numbers (or use `DISTINCT`).
- Join the `ids` to the big table. This should be very fast with the index in place.
- Finally trim surplus `ids` that have not been eaten by dupes and gaps. Every row has a **completely equal chance** to be picked.

[Short version](#)

You can **simplify** this query. The CTE in the query above is just for educational purposes:

```
SELECT *
```

```

FROM (
  SELECT DISTINCT 1 + trunc(random() * 5100000)::integer AS id
  FROM generate_series(1, 1100) g
) r
JOIN big USING (id)
LIMIT 1000;

```

Refine with rCTE

Especially if you are not so sure about gaps and estimates.

[Skip code block](#)

```

WITH RECURSIVE random_pick AS (
  SELECT *
  FROM (
    SELECT 1 + trunc(random() * 5100000)::int AS id
    FROM generate_series(1, 1030) -1000 + few percent - adapt to your needs
    LIMIT 1030
      -hint for query planner
  ) r
  JOIN big b USING (id)
    -eliminate miss

  UNION
    -eliminate dupe

  SELECT b.*
  FROM (
    SELECT 1 + trunc(random() * 5100000)::int AS id
    FROM random_pick r
      -plus 3 percent - adapt to your needs
    LIMIT 999
      -less than 1000, hint for query planner
  ) r
  JOIN big b USING (id)
    -eliminate miss
)
SELECT *
FROM random_pick
LIMIT 1000; -actual limit

```

We can work with a smaller surplus in the base query. If there are too many gaps so we don't find enough rows, we continue to iterate with the recursive term.

Duplicates are eliminated by the UNION in the rCTE.

The outer LIMIT makes the CTE stop as soon as we have enough rows.

This query is carefully drafted to use the available index, generate actually random rows and not stop until we fulfill the limit. There are a number of pitfalls here if you are going to rewrite it.

Wrap into function

For repeated use with varying parameters:

[Skip code block](#)

```

CREATE OR REPLACE FUNCTION f_random_sample(_limit int = 1000, _gaps real = 1.03)
  RETURNS SETOF big AS
$func$
DECLARE
  _surplus int := _limit * _gaps;
  _estimate int := (
    -get current estimate from system
    SELECT c.reltuples * _gaps
    FROM pg_class c
    WHERE c.oid = 'big'::regclass);
BEGIN

  RETURN QUERY
  WITH RECURSIVE random_pick AS (
    SELECT *
    FROM (

```

```

SELECT 1 + trunc(random() * _estimate)::int
  FROM generate_series(1, _surplus) g
  LIMIT _surplus           --hint for query planner
 ) r (id)
JOIN big USING (id)      --eliminate misses

UNION                      --eliminate dupes
SELECT *
  FROM (
    SELECT 1 + trunc(random() * _estimate)::int
      FROM random_pick        --just to make it recursive
     LIMIT _limit             --hint for query planner
  ) r (id)
JOIN big USING (id)      --eliminate misses
)
SELECT *
  FROM random_pick
 LIMIT _limit;
END
$func$ LANGUAGE plpgsql VOLATILE ROWS 1000;

```

Call:

```

SELECT * FROM f_random_sample();
SELECT * FROM f_random_sample(500, 1.05);

```

You could even make this generic to work for any table: Take the name of the PK column and the table as polymorphic type and use EXECUTE ... But that's beyond the scope of this question. See:

- [Refactor a PL/pgSQL function to return the output of various SELECT queries](#)

Possible alternative

IF your requirements allow **identical sets for repeated** calls (and we are talking about repeated calls) I would consider a **materialized view**. Execute above query once and write the result to a table. Users get a quasi random selection at lightening speed. Refresh your random pick at intervals or events of your choosing.

Postgres 9.5 introduces [TABLESAMPLE SYSTEM \(n\)](#)

It's *very fast*, but the result is *not exactly random*. [The documentation:](#)

The SYSTEM method is significantly faster than the BERNOULLI method when small sampling percentages are specified, but it may return a less-random sample of the table as a result of clustering effects.

And the number of rows returned can vary wildly. For our example, try:

```
SELECT * FROM big TABLESAMPLE SYSTEM ((1000 * 100) / 5100000.0);
```

Related:

- [Fast way to discover the row count of a table](#)

[Q: How do I query using fields inside the new PostgreSQL JSON datatype?](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

I am looking for some docs and/or examples for the new JSON functions in PostgreSQL 9.2.

Specifically, given a series of JSON records:

```
[ {name: "Toby", occupation: "Software Engineer"},  
 {name: "Zaphod", occupation: "Galactic President"}]  
]
```

How would I write the SQL to find a record by name?

In vanilla SQL:

```
SELECT * from json_data WHERE "name" = "Toby"
```

The official dev manual is quite sparse:

- <http://www.postgresql.org/docs-devel/static/datatype-json.html> 
- <http://www.postgresql.org/docs-devel/static/functions-json.html> 

Update I

I've put together a [gist detailing what is currently possible with PostgreSQL 9.2](#) . Using some custom functions, it is possible to do things like:

```
SELECT id, json_string(data, 'name') FROM things  
WHERE json_string(data, 'name') LIKE 'G%';
```

Update II

I've now moved my JSON functions into their own project:

[PostSQL](#)  - a set of functions for transforming PostgreSQL and PL/v8 into a totally awesome JSON document store

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

User: [toby-hede](#) 

[Answer](#)  by [erwin-brandstetter](#) 

Postgres 9.2

I quote [Andrew Dunstan on the pgsql-hackers list](#) :

At some stage there will possibly be some json-processing (as opposed to json-producing) functions, but not in 9.2.

Doesn't prevent him from providing an [example implementation in PLV8](#) that should solve your problem.

Postgres 9.3

Offers an arsenal of new functions and operators to add “json-processing”.

- [The manual on new JSON functionality](#).
- [The Postgres Wiki on new features in pg 9.3](#).
- @Will posted a link to a [blog demonstrating the new operators](#) in a comments below.

The answer to the **original question** in Postgres 9.3:

```
SELECT *
FROM json_array_elements(
  '[{"name": "Toby", "occupation": "Software Engineer"}, {"name": "Zaphod", "occupation": "Galactic President"} ]'
) AS elem
WHERE elem->>'name' = 'Toby';
```

Advanced example:

- [Query combinations with nested array of records in JSON datatype](#)

For bigger tables you may want to add an expression index to increase performance:

- [Index for finding an element in a JSON array](#)

Postgres 9.4

Adds **jsonb** (b for “binary”, values are stored as native Postgres types) and yet more functionality for *both* types. In addition to expression indexes mentioned above, jsonb also supports [GIN, btree and hash indexes](#), GIN being the most potent of these.

- The manual on [json and jsonb data types](#) and [functions](#).
- [The Postgres Wiki on JSONB in pg 9.4](#)

The manual goes as far as suggesting:

In general, **most applications should prefer to store JSON data as jsonb**, unless there are quite specialized needs, such as legacy assumptions about ordering of object keys.

Bold emphasis mine.

[Performance benefits from general improvements to GIN indexes.](#)

Postgres 9.5

Complete jsonb functions and operators. Add more functions to manipulate jsonb in

place and for display.

- [Major good news in the release notes of Postgres 9.5.](#) 
-

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

[Q: postgres: upgrade a user to be a superuser?](#)

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

In postgres, how do I change an existing user to be a superuser? I don't want to delete the existing user, for various reasons.

```
# alter user myuser...?
```

Thanks for your help.

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

User: [flossfan](#) 

[Answer](#)  by [quassnoi](#) 

```
ALTER USER myuser WITH SUPERUSER;
```

[Docs.](#) 

Tags: [postgresql](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to speed up insertion performance in PostgreSQL](#)

Tags: [postgresql](#) ([Prev Q](#))

I am testing Postgres insertion performance. I have a table with one column with number as its data type. There is an index on it as well. I filled the database up using this query:

```
insert into aNumber (id) values (564),(43536),(34560) ...
```

I inserted 4 million rows very quickly 10,000 at a time with the query above. After the database reached 6 million rows performance drastically declined to 1 Million rows every 15 min. Is there any trick to increase insertion performance? I need optimal insertion performance on this project.

Using Windows 7 Pro on a machine with 5 GB RAM.

Tags: [postgresql](#) ([Prev Q](#))

User: [luke101](#) 

[Answer](#)  by [craig-ringer](#) 

See [populate a database](#)  in the PostgreSQL manual, [depesz's excellent-as-usual](#)

[article](#) on the topic, and [this SO question](#).

There's lots to be done. The ideal solution would be to import into an UNLOGGED table without indexes, then change it to logged and add the indexes. Unfortunately in PostgreSQL 9.4 there's no support for changing tables from UNLOGGED to logged. 9.5 adds `ALTER TABLE... SET LOGGED` to permit you to do this.

If you can take your database offline for the bulk import, use [pg_bulkload](#).

Otherwise:

- Disable any triggers on the table
- Drop indexes before starting the import, re-create them afterwards. (It takes *much* less time to build an index in one pass than it does to add the same data to it progressively, and the resulting index is much more compact).
- If doing the import within a single transaction, it's safe to drop foreign key constraints, do the import, and re-create the constraints before committing. Do not do this if the import is split across multiple transactions as you might introduce invalid data.
- If possible, use `COPY` instead of `INSERTS`
- If you can't use `COPY` consider using multi-valued `INSERTS` if practical. You seem to be doing this already. Don't try to list *too* many values in a single `VALUES` though; those values have to fit in memory a couple of times over, so keep it to a few hundred per statement.
- Batch your inserts into explicit transactions, doing hundreds of thousands or millions of inserts per transaction. There's no practical limit AFAIK, but batching will let you recover from an error by marking the start of each batch in your input data. Again, you seem to be doing this already.
- Use `synchronous_commit=off` and a huge `commit_delay` to reduce `fsync()` costs. This won't help much if you've batched your work into big transactions, though.
- `INSERT` or `COPY` in parallel from several connections. How many depends on your hardware's disk subsystem; as a rule of thumb, you want one connection per physical hard drive if using direct attached storage.
- Set a high `checkpoint_segments` value and enable `log_checkpoints`. Look at the PostgreSQL logs and make sure it's not complaining about checkpoints occurring too frequently.
- If and only if you don't mind losing your entire PostgreSQL cluster (your database and any others on the same cluster) to catastrophic corruption if the system crashes during the import, you can stop Pg, set `fsync=off`, start Pg, do your import, then (vitally) stop Pg and set `fsync=on` again. See [WAL configuration](#). **Do not do this if there is already any data you care about in any database on your PostgreSQL install.** If you set `fsync=off` you can also set `full_page_writes=off`; again, just remember to turn it back on after your import to prevent database corruption and data loss. See [non-durable settings](#) in the Pg manual.

You should also look at tuning your system:

- Use *good quality* SSDs for storage as much as possible. Good SSDs with reliable, power-protected write-back caches make commit rates incredibly faster. They're less beneficial when you follow the advice above - which reduces disk flushes / number of `fsync()`s - but can still be a big help. Do not use cheap SSDs without proper power-failure protection unless you don't care about keeping your data.
- If you're using RAID 5 or RAID 6 for direct attached storage, stop now. Back your data up, restructure your RAID array to RAID 10, and try again. RAID 5/6 are hopeless for bulk write performance - though a good RAID controller with a big cache can help.
- If you have the option of using a hardware RAID controller with a big battery-backed write-back cache this can really improve write performance for workloads with lots of commits. It doesn't help as much if you're using async commit with a `commit_delay` or if you're doing fewer big transactions during bulk loading.
- If possible, store WAL (`pg_xlog`) on a separate disk / disk array. There's little point in using a separate filesystem on the same disk. People often choose to use a RAID1 pair for WAL. Again, this has more effect on systems with high commit rates, and it has little effect if you're using an unlogged table as the data load target.

You may also be interested in [Optimise PostgreSQL for fast testing](#) .

Tags: [postgresql](#) ([Prev](#) [Q](#))

Performance

[Skip to questions](#),

Wiki by user [kiamlaluno](#) 

The performance of applications is often a paramount concern for mission critical systems. If your question pertains to optimization, whether it be database queries, algorithms, reducing network/transactional overhead, or anything that deals with speed or capacity, consider using this tag.

A good question states performance goals that need to be achieved as well as other restrictions. Trying to optimize something without measuring is not a “performance” question or work, but most likely personal entertainment - expect a question without goals/measurements to be treated as such.

Performance for many programs is represented in [big O notation](#) , which classifies how an algorithm’s resource requirements change in response to a change in input size.

This tag can also represent [system performance](#) , which is one of key non-functional requirements of an application or system.

The two main measures of performance are

- Throughput (how many in a time frame). Example of units: transactions per second (TPS), megabytes per second (MB/s), giga-bits per second (Gb/s), messages/request/pages per second.
- Latency (how long for an action). For example, seek time of 8 ms and search time of 100 ms.

Latency is often qualified with a statistical measure. Note: latencies usually don’t follow a [normal distribution](#)  and have very high upper limits compared with the average latency. As such the [standard deviation](#)  is not useful.

- Average latency. The average of all the latencies.
- Typical or [median](#)  latency. The mid-point of the range of possible latencies. This is usually 50% to 90% of the average latency. As this is the lowest figure it is often reported by vendors.
- Percentile latency. The figure it is less than or equal to N% of the time. That is, 99 percentile if the latency it is not more than this, 99 times out of 100.
- Worst or maximum latency. The highest latency measured.

When seeking to improve performance: prototype and measure first, optimize only if and where needed.

See also: [optimization](#)  [profiling](#)  [assembly](#)  [compiler](#)  [low-latency](#) 

Questions

[Q: Equals\(=\) vs. LIKE](#)

Tags: [performance](#) ([Prev Q](#))

When using SQL, are there any benefits of using = in a WHERE clause instead of LIKE?

Without any special operators, LIKE and = are the same, right?

Tags: [performance](#) ([Prev Q](#))

User: [travis](#)

[Answer](#) by [achinda99](#)

The equals (=) operator is a “comparison operator compares two values for equality.” In other words, in an SQL statement, it won’t return true unless both sides of the equation are equal. For example:

```
SELECT * FROM Store WHERE Quantity = 200;
```

The LIKE operator “implements a pattern match comparison” that attempts to match “a string value against a pattern string containing wild-card characters.” For example:

```
SELECT * FROM Employees WHERE Name LIKE 'Chris%';
```

LIKE is generally used only with strings and equals (I believe) is faster. The equals operator treats wild-card characters as literal characters. The difference in results returned are as follows:

```
SELECT * FROM Employees WHERE Name = 'Chris';
```

And

```
SELECT * FROM Employees WHERE Name LIKE 'Chris';
```

Would return the same result, though using LIKE would generally take longer as its a pattern match. However,

```
SELECT * FROM Employees WHERE Name = 'Chris%';
```

And

```
SELECT * FROM Employees WHERE Name LIKE 'Chris%';
```

Would return different results, where using “=” results in only results with “Chris%” being returned and the LIKE operator will return anything starting with “Chris”.

Hope that helps. Some good info can be found [here](#).

[Answer](#) by [mehaase](#)

Different Operators

LIKE and = are different operators. Most answers here focus on the wildcard support,

which is not the only difference between these operators!

= is a comparison operator that operates on numbers and strings. When comparing strings, the comparison operator compares *whole strings*.

LIKE is a string operator that compares *character by character*.

To complicate matters, both operators use a [collation](#) which can have important effects on the result of the comparison.

Motivating Example

Let's first identify an example where these operators produce obviously different results. Allow me to quote from the MySQL manual:

Per the SQL standard, LIKE performs matching on a per-character basis, thus it can produce results different from the = comparison operator:

[Skip code block](#)

```
mysql> SELECT 'ä' LIKE 'ae' COLLATE latin1_german2_ci;
+-----+
| 'ä' LIKE 'ae' COLLATE latin1_german2_ci |
+-----+
|          0 |
+-----+
mysql> SELECT 'ä' = 'ae' COLLATE latin1_german2_ci;
+-----+
| 'ä' = 'ae' COLLATE latin1_german2_ci |
+-----+
|          1 |
+-----+
```

Please note that this page of the MySQL manual is called *String Comparison Functions*, and = is not discussed, which implies that = is not strictly a string comparison function.

How Does = Work?

The [SQL Standard § 8.2](#) describes how = compares strings:

The comparison of two character strings is determined as follows:

a) If the length in characters of X is not equal to the length in characters of Y, then the shorter string is effectively replaced, for the purposes of comparison, with a copy of itself that has been extended to the length of the longer string by concatenation on the right of one or more pad characters, where the pad character is chosen based on CS. If CS has the NO PAD attribute, then the pad character is an implementation-dependent character different from any character in the character set of X and Y that collates less than any string under CS. Otherwise, the pad character is a .

b) The result of the comparison of X and Y is given by the collating sequence CS.

c) Depending on the collating sequence, two strings may compare as equal even if they are of different lengths or contain different sequences of characters. When the operations MAX, MIN, DISTINCT, references to a grouping column, and the UNION, EXCEPT, and INTERSECT operators refer to character strings, the specific

value selected by these operations from a set of such equal values is implementation-dependent.

(Emphasis added.)

What does this mean? It means that when comparing strings, the = operator is just a thin wrapper around the current collation. A collation is a library that has various rules for comparing strings. Here's an example of [a binary collation from MySQL](#):

```
static int my_strnncoll_binary(const CHARSET_INFO *cs __attribute__((unused)),
                                const uchar *s, size_t slen,
                                const uchar *t, size_t tlen,
                                my_bool t_is_prefix)
{
    size_t len= MY_MIN(slen,tlen);
    int cmp= memcmp(s,t,len);
    return cmp ? cmp : (int)((t_is_prefix ? len : slen) - tlen);
}
```

This particular collation happens to compare byte-by-byte (which is why it's called "binary" — it doesn't give any special meaning to strings). Other collations may provide more advanced comparisons.

For example, here is a [UTF-8 collation](#) that supports case-insensitive comparisons. The code is too long to paste here, but go to that link and read the body of `my_strnncollsp_utf8mb4()`. This collation can process multiple bytes at a time and it can apply various transforms (such as case insensitive comparison). The = operator is completely abstracted from the vagaries of the collation.

How Does LIKE Work?

The [SQL Standard § 8.5](#) describes how LIKE compares strings:

The <predicate>

M LIKE P

is true if there exists a partitioning of M into substrings such that:

- i) A substring of M is a sequence of 0 or more contiguous <character representation>s of M and each <character representation> of M is part of exactly one substring.
- ii) If the i-th substring specifier of P is an arbitrary character specifier, the i-th substring of M is any single <character representation>.
- iii) If the i-th substring specifier of P is an arbitrary string specifier, then the i-th substring of M is any sequence of 0 or more <character representation>s.
- iv) If the i-th substring specifier of P is neither an arbitrary character specifier nor an arbitrary string specifier, then the i-th substring of M is equal to that substring specifier according to the collating sequence of the <like predicate>, without the appending of <space> characters to M, and has the same length as that substring specifier.**
- v) The number of substrings of M is equal to the number of substring specifiers of P.

(Emphasis added.)

This is pretty wordy, so let's break it down. Items ii and iii refer to the wildcards _ and %, respectively. If P does not contain any wildcards, then only item iv applies. This is the case of interest posed by the OP.

In this case, it compares each “substring” (individual characters) in M against each substring in P using the current collation.

Conclusions

The bottom line is that when comparing strings, = compares the entire string while LIKE compares one character at a time. Both comparisons use the current collation. This difference leads to different results in some cases, as evidenced in the first example in this post.

Which one should you use? Nobody can tell you that — you need to use the one that's correct for your use case. Don't prematurely optimize by switching comparison operators.

Tags: [performance](#) ([Prev Q](#))

JOIN

[Skip to questions](#),

Wiki by user [scmsem](#) 

An SQL JOIN clause is used to combine rows from two or more tables, based on a common field between them. A JOIN is a means for combining fields from two tables (or more) by using values common to each.

ANSI-standard SQL specifies five types of JOIN: INNER, LEFT OUTER, RIGHT OUTER, FULL OUTER and CROSS. In [SQL](#), [join](#) is used to identify rows from tables that share a common key.

See also [A Visual Explanation of SQL Joins](#).

Specific join tags :

You can specify your question by adding extra tags:

[left-join](#) [inner-join](#) [outer-join](#) [self-join](#) [cross-join](#) [full-outer-join](#) [right-join](#) [natural-join](#)

Questions :

- [Join vs. sub-query](#)
 - [Difference between INNER and OUTER joins](#)
 - [Difference between JOIN and INNER JOIN](#)
 - [What is the difference between Left, Right, Outer and Inner Joins?](#)
 - [SQL join: where clause vs. on clause](#)
-

Questions

[Q: What is the syntax for an inner join in LINQ to SQL?](#)

Tags: [join](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#)), [linq-to-sql](#) ([Prev Q](#)) ([Next Q](#))

I'm writing a linq to sql statement & I'm just after the standard syntax for a normal inner join with an 'on' clause in C#.

ie how do you represent this in LINQ to SQL?:

```
select * from table1  
inner join table2 on table1.field = table2.field
```

EDIT: Real query to get all contacts for a dealer:

```
select DealerContact.*  
from Dealer  
    inner join DealerContact on Dealer.DealerID = DealerContact.DealerID
```

Tags: [join](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#)), [linq-to-sql](#) ([Prev Q](#)) ([Next Q](#))

User: [glenn-slaven](#) 

[Answer](#)  by [jon-limjap](#) 

It goes something like:

```
from t1 in db.Table1  
join t2 in db.Table2 on t1.field equals t2.field  
select new { t1.field2, t2.field3}
```

It would be nice to have sensible names and fields for your tables for a better example. :)

Update

I think for your query this might be more appropriate:

```
var dealercontacts = from contact in DealerContact  
                      join dealer in Dealer on contact.DealerId equals dealer.ID  
                      select contact;
```

Since you are looking for the contacts, not the dealers.

[Answer](#)  by [clever-human](#) 

And because I prefer the expression chain syntax, here is how you do it with that:

```
var dealerContracts = DealerContact.Join(Dealer,  
                                         contact => contact.DealerId,  
                                         dealer => dealer.DealerId,  
                                         (contact, dealer) => contact);
```

Tags: [join](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#)), [linq-to-sql](#) ([Prev Q](#)) ([Next Q](#))

[Q: Difference between INNER and OUTER joins](#)

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

What is the difference between INNER JOIN and OUTER JOIN?

How do LEFT JOIN, RIGHT JOIN, and FULL JOIN fit in?

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

User: [cdv](#) 

[Answer](#)  by [mark-harrison](#) 

Assuming you're joining on columns with no duplicates, which is a very common case:

- An inner join of A and B gives the result of A intersect B, i.e. the inner part of a [Venn diagram](#)  intersection.
- An outer join of A and B gives the results of A union B, i.e. the outer parts of a Venn diagram union.

Examples

Suppose you have two tables, with a single column each, and data as follows:

A	B
-	-
1	3
2	4
3	5
4	6

Note that (1,2) are unique to A, (3,4) are common, and (5,6) are unique to B.

Inner join

An inner join using either of the equivalent queries gives the intersection of the two tables, i.e. the two rows they have in common.

```
select * from a INNER JOIN b on a.a = b.b;
select a.* , b.* from a,b where a.a = b.b;

a | b
---+---
3 | 3
4 | 4
```

Left outer join

A left outer join will give all rows in A, plus any common rows in B.

```
select * from a LEFT OUTER JOIN b on a.a = b.b;
select a.* , b.* from a,b where a.a = b.b(+);

a | b
---+---
1 | null
2 | null
3 | 3
4 | 4
```

Right outer join

A right outer join will give all rows in B, plus any common rows in A.

```
select * from a RIGHT OUTER JOIN b on a.a = b.b;
select a.* , b.* from a,b where a.a(+) = b.b;
```

a	b
3	3
4	4
null	5
null	6

Full outer join

A full outer join will give you the union of A and B, i.e. all the rows in A and all the rows in B. If something in A doesn't have a corresponding datum in B, then the B portion is null, and vice versa.

[Skip code block](#)

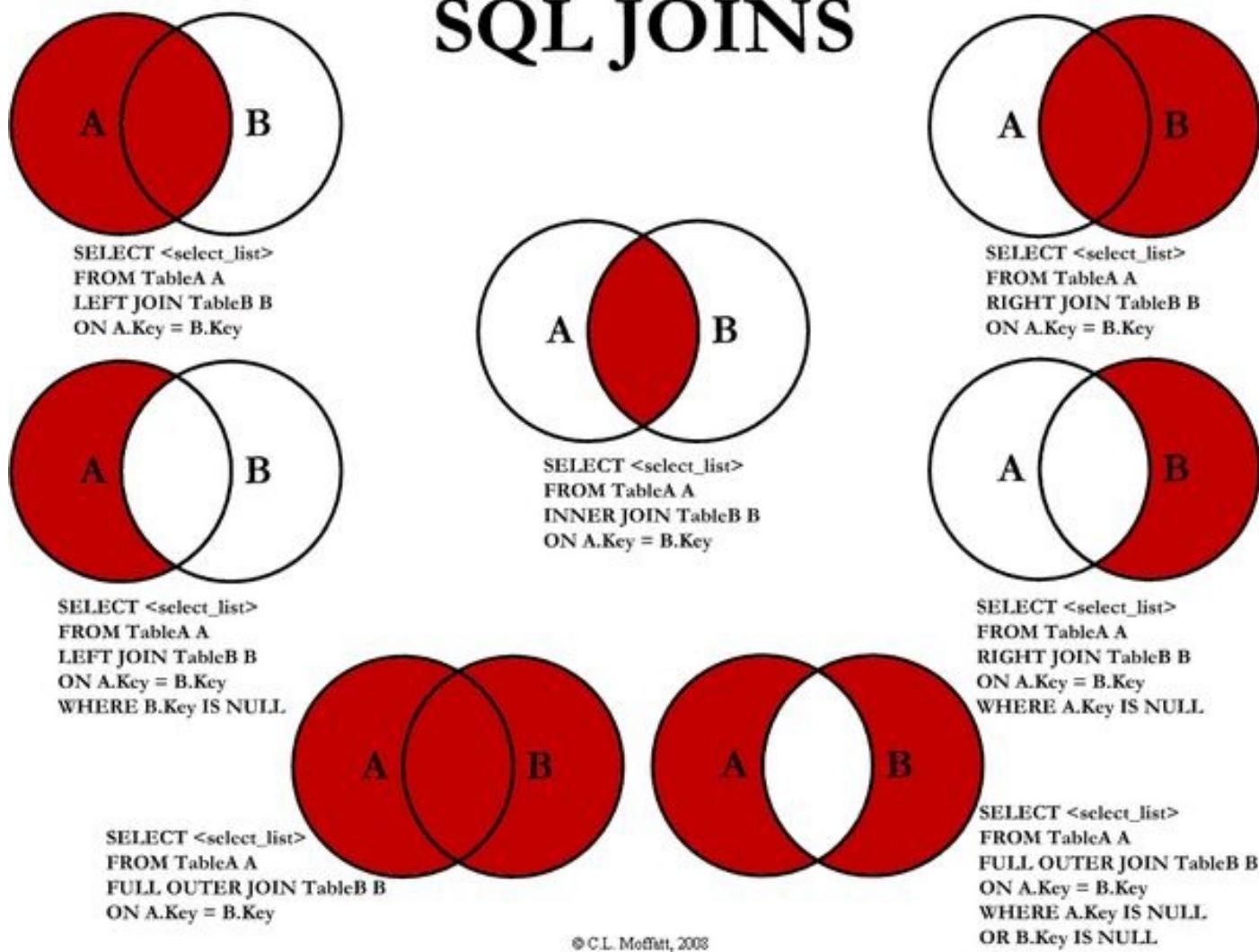
```
select * from a FULL OUTER JOIN b on a.a = b.b;
```

a	b
1	null
2	null
3	3
4	4
null	6
null	5

[Answer](#) by [teoman-shipahi](#)

Also you can consider the following schema for different join types;

SQL JOINS



[Answer](#) by [gt_ebuddy](#)

The following was taken from the article “[MySQL - LEFT JOIN and RIGHT JOIN, INNER JOIN and OUTER JOIN](#)” by Graham Ellis on his blog Horse’s Mouth.

In a database such as MySQL, data is divided into a number of tables which are then connected (Joined) together by JOIN in SELECT commands to read records from multiple tables. Read this example to see how it works.

First, some sample data:

[Skip code block](#)

```
people
mysql> select * from people;
+-----+-----+-----+
| name | phone | pid |
+-----+-----+-----+
| Mr Brown | 01225 708225 | 1 |
| Miss Smith | 01225 899360 | 2 |
| Mr Pullen | 01380 724040 | 3 |
+-----+-----+-----+
3 rows in set (0.00 sec)

property
mysql> select * from property;
+-----+-----+-----+
| pid | spid | selling |
+-----+-----+-----+
| 1 | 1 | Old House Farm |
| 3 | 2 | The Willows |
| 3 | 3 | Tall Trees |
| 3 | 4 | The Melksham Florist |
| 4 | 5 | Dun Roamin |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

REGULAR JOIN

If we do a regular JOIN (with none of the keywords INNER, OUTER, LEFT or RIGHT), then we get all records that match in the appropriate way in the two tables, and records in both incoming tables that do not match are not reported:

[Skip code block](#)

```
mysql> select name, phone, selling
from people join property
on people.pid = property.pid;
+-----+-----+-----+
| name | phone | selling |
+-----+-----+-----+
| Mr Brown | 01225 708225 | Old House Farm |
| Mr Pullen | 01380 724040 | The Willows |
| Mr Pullen | 01380 724040 | Tall Trees |
| Mr Pullen | 01380 724040 | The Melksham Florist |
+-----+-----+-----+
4 rows in set (0.01 sec)
```

LEFT JOIN

If we do a LEFT JOIN, we get all records that match in the same way and IN ADDITION we get an extra record for each unmatched record in the left table of the join - thus ensuring (in this example) that every PERSON gets a mention:

[Skip code block](#)

```

mysql> select name, phone, selling
from people left join property
on people.pid = property.pid;
+-----+-----+-----+
| name      | phone      | selling      |
+-----+-----+-----+
| Mr Brown  | 01225 708225 | Old House Farm |
| Miss Smith | 01225 899360 | NULL <<- unmatch |
| Mr Pullen  | 01380 724040 | The Willows   |
| Mr Pullen  | 01380 724040 | Tall Trees    |
| Mr Pullen  | 01380 724040 | The Melksham Florist |
+-----+-----+-----+
5 rows in set (0.00 sec)

```

RIGHT JOIN

If we do a RIGHT JOIN, we get all the records that match and IN ADDITION an extra record for each unmatched record in the right table of the join - in my example, that means that each property gets a mention even if we don't have seller details:

[Skip code block](#)

```

mysql> select name, phone, selling
from people right join property
on people.pid = property.pid;
+-----+-----+-----+
| name      | phone      | selling      |
+-----+-----+-----+
| Mr Brown  | 01225 708225 | Old House Farm |
| Mr Pullen  | 01380 724040 | The Willows   |
| Mr Pullen  | 01380 724040 | Tall Trees    |
| Mr Pullen  | 01380 724040 | The Melksham Florist |
| NULL      | NULL       | Dun Roamin   |
+-----+-----+-----+
5 rows in set (0.00 sec)

```

An INNER JOIN does a full join, just like the first example, and the word OUTER may be added after the word LEFT or RIGHT in the last two examples - it's provided for ODBC compatibility and doesn't add an extra capabilities.

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

Q: Explicit vs implicit SQL joins

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

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;

```

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

User: [dmanxiii](#) 

[Answer](#)  by [lomaxx](#) 

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

PS: Be aware that the `IMPLICIT OUTER JOIN` syntax is deprecated since SQL Server 2005.
(The `IMPLICIT OUTER JOIN` syntax as used in the question is still supported)

<http://blogs.technet.com/b/wardpond/archive/2008/09/13/deprecation-of-old-style-join-syntax-only-a-partial-thing.aspx?Redirected=true>

[Answer](#) by [grom](#)

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.

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL join: where clause vs. on clause](#)

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

After reading it, this is *not* a duplicate of [Explicit vs Implicit SQL Joins](#). The answer may be related (or even the same) but the **question** is different.

What is the difference and what should go in each?

If I understand the theory correctly, the query optimizer should be able to use both interchangeably.

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

User: [bcs](#)

[Answer](#) by [joel-coehoorn](#)

They are not the same thing.

Consider these queries:

```
SELECT *
FROM Orders
LEFT JOIN OrderLines ON OrderLines.OrderID=Orders.ID
WHERE Orders.ID = 12345
```

and

```
SELECT *
FROM Orders
LEFT JOIN OrderLines ON OrderLines.OrderID=Orders.ID
AND Orders.ID = 12345
```

The first will return an order and its lines, if any, for order number 12345. The second will return all orders, but only order 12345 will have any lines associated with it.

With an `INNER JOIN`, the clauses are *effectively* equivalent. However, just because they are functionally the same, in that they produce the same results, does not mean the two kinds

of clauses have the same semantic meaning.

[Answer](#) by [cade-roux](#)

On INNER JOINS they are interchangeable, and the optimizer will rearrange them at will.

On OUTER JOINS, they are not necessarily interchangeable, depending on which side of the join they depend on.

I put them in either place depending on the readability.

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL left join vs multiple tables on FROM line?](#)

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

Most SQL dialects accept both the following queries:

```
SELECT a.foo, b.foo  
FROM a, b  
WHERE a.x = b.x  
  
SELECT a.foo, b.foo  
FROM a  
LEFT JOIN b ON a.x = b.x
```

Now obviously when you need an outer join, the second syntax is required. But when doing an inner join why should I prefer the second syntax to the first (or vice versa)?

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

User: [jmucchiello](#)

[Answer](#) by [lasse-v.-karlsen](#)

The old syntax, with just listing the tables, and using the WHERE clause to specify the join criteria, is being deprecated in most modern databases.

It's not just for show, the old syntax has the possibility of being ambiguous when you use both INNER and OUTER joins in the same query.

Let me give you an example.

Let's suppose you have 3 tables in your system:

```
Company  
Department  
Employee
```

Each table contain numerous rows, linked together. You got multiple companies, and each company can have multiple departments, and each department can have multiple employees.

Ok, so now you want to do the following:

List all the companies, and include all their departments, and all their employees.

Note that some companies don't have any departments yet, but make sure you include them as well. Make sure you only retrieve departments that have employees, but always list all companies.

So you do this:

```
SELECT *--for simplicity
FROM Company, Department, Employee
WHERE Company.ID *= Department.CompanyID
AND Department.ID = Employee.DepartmentID
```

Note that the last one there is an inner join, in order to fulfill the criteria that you only want departments with people.

Ok, so what happens now. Well, the problem is, it depends on the database engine, the query optimizer, indexes, and table statistics. Let me explain.

If the query optimizer determines that the way to do this is to first take a company, then find the departments, and then do an inner join with employees, you're not going to get any companies that don't have departments.

The reason for this is that the `WHERE` clause determines which *rows* end up in the final result, not individual parts of the rows.

And in this case, due to the left join, the `Department.ID` column will be `NONE`, and thus when it comes to the `INNER JOIN` to `Employee`, there's no way to fulfill that constraint for the `Employee` row, and so it won't appear.

On the other hand, if the query optimizer decides to tackle the department-employee join first, and then do a left join with the companies, you will see them.

So the old syntax is ambiguous. There's no way to specify what you want, without dealing with query hints, and some databases have no way at all.

Enter the new syntax, with this you can choose.

For instance, if you want all companies, as the problem description stated, this is what you would write:

```
SELECT *
FROM Company
LEFT JOIN (
    Department INNER JOIN Employee ON Department.ID = Employee.DepartmentID
) ON Company.ID = Department.CompanyID
```

Here you specify that you want the department-employee join to be done as one join, and then left join the results of that with the companies.

Additionally, let's say you only want departments that contain the letter X in their name. Again, with old style joins, you risk losing the company as well, if it doesn't have any departments with an X in its name, but with the new syntax, you can do this:

```
SELECT *
FROM Company
LEFT JOIN (
    Department INNER JOIN Employee ON Department.ID = Employee.DepartmentID
) ON Company.ID = Department.CompanyID AND Department.Name LIKE '%X%'
```

This extra clause is used for the joining, but is not a filter for the entire row. So the row

might appear with company information, but might have NULLs in all the department and employee columns for that row, because there is no department with an X in its name for that company. This is hard with the old syntax.

This is why, amongst other vendors, Microsoft has deprecated the old outer join syntax, but not the old inner join syntax, since SQL Server 2005 and upwards. The only way to talk to a database running on Microsoft SQL Server 2005 or 2008, using the old style outer join syntax, is to set that database in 8.0 compatibility mode (aka SQL Server 2000).

Additionally, the old way, by throwing a bunch of tables at the query optimizer, with a bunch of WHERE clauses, was akin to saying “here you are, do the best you can”. With the new syntax, the query optimizer has less work to do in order to figure out what parts goes together.

So there you have it.

LEFT and INNER JOIN is the wave of the future.

Tags: [join](#) ([Prev Q](#)) ([Next Q](#))

Q: SQL join: selecting the last records in a one-to-many relationship

Tags: [join](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#)), [indexing](#) ([Next Q](#))

Suppose I have a table of customers and a table of purchases. Each purchase belongs to one customer. I want to get a list of all customers along with their last purchase in one SELECT statement. What is the best practice? Any advice on building indexes?

Please use these table/column names in your answer:

- customer: id, name
- purchase: id, customer_id, item_id, date

And in more complicated situations, would it be (performance-wise) beneficial to denormalize the database by putting the last purchase into the customer table?

If the (purchase) id is guaranteed to be sorted by date, can the statements be simplified by using something like LIMIT 1?

Tags: [join](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#)), [indexing](#) ([Next Q](#))

User: [netvope](#)

[Answer](#) by [bill-karwin](#)

This is an example of the greatest-n-per-group problem that has appeared regularly on StackOverflow.

Here's how I usually recommend solving it:

```
SELECT c.*, p1.*  
FROM customer c  
JOIN purchase p1 ON (c.id = p1.customer_id)  
LEFT OUTER JOIN purchase p2 ON (c.id = p2.customer_id AND  
    (p1.date < p2.date OR p1.date = p2.date AND p1.id < p2.id))  
WHERE p2.id IS NULL;
```

Explanation: given a row p1, there should be no row p2 with the same customer and a later date (or in the case of ties, a later id). When we find that to be true, then p1 is the most recent purchase for that customer.

Regarding indexes, I'd create a compound index in purchase over the columns (customer_id, date, id). That may allow the outer join to be done using a covering index. Be sure to test on your platform, because optimization is implementation-dependent. Use the features of your RDBMS to analyze the optimization plan. E.g. EXPLAIN on MySQL.

Some people use subqueries instead of the solution I show above, but I find my solution makes it easier to resolve ties.

Tags: [join](#) ([Prev Q](#)) ([Next Q](#)), [select](#) ([Prev Q](#)) ([Next Q](#)), [indexing](#) ([Next Q](#))

Q: Joins are for lazy people?

Tags: [join](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#)), [java](#) ([Prev Q](#)) ([Next Q](#))

I recently had a discussion with another developer who claimed to me that JOINs (SQL) are useless. This is technically true but he added that using joins is less efficient than making several requests and link tables in the code (C# or Java).

For him joins are for lazy people that don't care about performance. Is this true? Should we avoid using joins?

Tags: [join](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#)), [java](#) ([Prev Q](#)) ([Next Q](#))

User: [b413](#) 

[Answer](#)  by [michael-borgwardt](#) 

No, we should avoid developers who hold such incredibly wrong opinions.

In many cases, a database join is several orders of magnitude faster than anything done via the client, because it avoids DB roundtrips, and the DB can use indexes to perform the join.

~~Off the top of my head, I can't even imagine a single scenario where a correctly used join would be slower than the equivalent client-side operation.~~

Edit: There are some rare cases where custom client code can do things more efficiently than a straightforward DB join (see comment by meriton). But this is very much the exception.

[Answer](#)  by [marc-gravell](#) 

It sounds to me like your colleague would do well with a no-sql document-database or key-value store. Which are themselves very good tools and a good fit for many problems.

However, a *relational* database is heavily optimised for working with sets. There are many, many ways of querying the data based on joins that are *vastly* more efficient than lots of round trips. This is where the versatility of a rdbms comes from. You can achieve the same in a nosql store too, but you often end up building a separate structure suited for each different nature of query.

In short: I disagree. In a RDBMS, joins are ***fundamental***. If you aren't using them, you aren't using it as a RDBMS.

Tags: [join](#) ([Prev Q](#)) ([Next Q](#)), [c#](#) ([Prev Q](#)) ([Next Q](#)), [java](#) ([Prev Q](#)) ([Next Q](#))

[**Q: SQL Inner-join with 3 tables?**](#) 

Tags: [join](#) ([Prev Q](#))

I'm trying to join 3 tables in a view; here is the situation:

I have a table that contains information of students who are applying to live on this College Campus. I have another table that lists the Hall Preferences (3 of them) for each Student. But each of these preferences are merely an ID Number, and the ID Number has a

corresponding Hall Name in a third table (did not design this database...).

Pretty much, I have INNER JOIN on the table with their preferences, and their information, the result is something like...

```
John Doe | 923423 | Incoming Student | 005
```

Where 005 would be the HallID. So Now I want to match that HallID to a third table, where this table contains a HallID and HallName.

So pretty much, I want my result to be like...

```
John Doe | 923423 | Incoming Student | Foley Hall <---(INSTEAD OF 005)
```

EDIT here is what I currently Have

```
SELECT      s.StudentID, s.FName, s.LName, s.Gender, s.BirthDate, s.Email, r.HallPref1,    r.HallPref2,
            FROM        dbo.StudentSignUp AS s INNER JOIN
                        RoomSignUp.dbo.Incoming_Applications_Current AS r ON s.StudentID = r.StudentID
                        HallData.dbo.Halls AS h ON r.HallPref1 = h.HallID
```

Tags: [join](#) ([Prev Q](#))

User: [bob-sanders](#) 

[Answer](#)  by [bluefeet](#) 

You can do the following (I guessed on table fields,etc)

```
SELECT s.studentname
      , s.studentid
      , s.studentdesc
      , h.hallname
  FROM students s
 INNER JOIN hallprefs hp
   on s.studentid = hp.studentid
 INNER JOIN halls h
   on hp.hallid = h.hallid
```

EDIT:

Based on your request for multiple halls you could do it this way. You just join on your Hall table multiple times for each room pref id:

[Skip code block](#)

```
SELECT      s.StudentID
            , s.FName
            , s.LName
            , s.Gender
            , s.BirthDate
            , s.Email
            , r.HallPref1
            , h1.hallName as Pref1HallName
            , r.HallPref2
            , h2.hallName as Pref2HallName
            , r.HallPref3
            , h3.hallName as Pref3HallName
  FROM        dbo.StudentSignUp AS s
 INNER JOIN RoomSignUp.dbo.Incoming_Applications_Current AS r
   ON s.StudentID = r.StudentID
 INNER JOIN HallData.dbo.Halls AS h1
   ON r.HallPref1 = h1.HallID
 INNER JOIN HallData.dbo.Halls AS h2
   ON r.HallPref2 = h2.HallID
 INNER JOIN HallData.dbo.Halls AS h3
   ON r.HallPref3 = h3.HallID
```

Tags: [join](#) ([Prev Q](#))

DateTime

[Skip to questions](#),

Wiki by user [agf](#)

A DateTime object in many programming languages describes a date and a time of day. It can express either an instant in time or a position on a calendar, depending on the context in which it is used and the specific implementation.

- In [lua](#), [os.date](#) and [os.time](#) are provided as part of lua's standard library for working with date and time.
- In [sql-server](#), [datetime](#) is a type that represents a date and time of day. It carries no time zone context, so it is a position on a calendar, not an instant in time.
- In [.net](#), [DateTime](#) is a structure that is typically expressed as a date and time of day. It represents *either* an instant in time, or a position on a calendar. The precise meaning is dependent on the value of its [Kind](#) property.
- In [jodatime](#), a popular [java](#) library, [DateTime](#) is a class which represents a instant in time, with reference to a particular time zone.
- In [python](#), [datetime](#) is an object which represents a date and time of day. The precise meaning is dependent on the value of its [tzinfo](#) property. It is also the name of [the module](#) that provides [date](#), [time](#), [datetime](#), and related objects.
- In [php](#) the [DateTime](#) class is a representation of a instant in time, with reference to a particular time zone.
- In [fortran](#), the time and date manipulation library [datetime-fortran](#) can be used.
- In [javascript](#), a [Date](#) instance represents a single moment in time. Date objects are based on a time value that is the number of milliseconds since 1 January, 1970 UTC.
- In [r](#), the classes “POSIXlt” and “POSIXct” represent calendar dates and times. Subtraction of two date-time objects is equivalent to using [difftime](#). Class “POSIXct” represents the (signed) number of seconds since the beginning of 1970 (in the UTC time zone) as a numeric vector. Class “POSIXlt” is a named list of vectors representing sec, min, hour et al

Questions

[Q: How to convert java.util.date to java.sql.date?](#)

Tags: [datetime](#) ([Prev Q](#)) ([Next Q](#)), [java](#) ([Prev Q](#)) ([Next Q](#))

I am trying to use java.util.Date as input and then creating a query with it - so I need java.sql.Date.

I was surprised to find that it couldn't do the conversion implicitly or explicitly - but I don't even know how I would do this, as the Java API is still fairly new to me. Any suggestions? It seems like this should be an easy feat to accomplish.

Tags: [datetime](#) ([Prev Q](#)) ([Next Q](#)), [java](#) ([Prev Q](#)) ([Next Q](#))

User: [david-ackerman](#) 

[Answer](#)  by [david-ackerman](#) 

Nevermind....

[Skip code block](#)

```
public class MainClass {  
  
    public static void main(String[] args) {  
        java.util.Date utilDate = new java.util.Date();  
        java.sql.Date sqlDate = new java.sql.Date(utilDate.getTime());  
        System.out.println("utilDate:" + utilDate);  
        System.out.println("sqlDate:" + sqlDate);  
  
    }  
  
}
```

explains it. The link is http://www.java2s.com/Tutorial/Java/0040_Data-Type/ConvertfromjavaultDateObjecttojavasqlDateObject.htm 

Tags: [datetime](#) ([Prev Q](#)) ([Next Q](#)), [java](#) ([Prev Q](#)) ([Next Q](#))

[Q: java.util.Date vs java.sql.Date](#)

Tags: [datetime](#) ([Prev Q](#)), [java](#) ([Prev Q](#)) ([Next Q](#))

`java.util.Date` vs `java.sql.Date`: when to use which and why?

Tags: [datetime](#) ([Prev Q](#)), [java](#) ([Prev Q](#)) ([Next Q](#))

User: [flybywire](#)

[Answer](#) by [esko](#)

Congratulations, you've hit my favorite pet peeve with JDBC: Date class handling.

Basically databases usually support at least **three** forms of datetime fields which are date, time and timestamp. Each of these have a corresponding class in JDBC and **each of them extend** [java.util.Date](#). Quick semantics of each of these three are the following:

- [java.sql.Date](#) corresponds to SQL DATE which means it stores *years, months and days* while *hour, minute, second and millisecond* are ignored. Additionally `sql.Date` isn't tied to timezones.
- [java.sql.Time](#) corresponds to SQL TIME and as should be obvious, only contains information about *hour, minutes, seconds and milliseconds*.
- [java.sql.Timestamp](#) corresponds to SQL TIMESTAMP which is exact date to the nanosecond (*note that util.Date only supports milliseconds!*) with customizable precision.

One of the most common bugs when using JDBC drivers in relation to these three types is that the types are handled incorrectly. This means that `sql.Date` is timezone specific, `sql.Time` contains current year, month and day et cetera et cetera.

Finally: Which one to use?

Depends on the SQL type of the field, really. `PreparedStatement` has setters for all three values, `# setDate()` being the one for `sql.Date`, `# setTime()` for `sql.Time` and `# setTimestamp()` for `sql.Timestamp`.

Do note that if you use `ps.setObject(fieldIndex, utilDateObject)`; you can actually give a normal `util.Date` to most JDBC drivers which will happily devour it as if it was of the correct type but when you request the data afterwards, you may notice that you're actually missing stuff.

I'm really saying that none of the Dates should be used at all.

What I am saying that save the milliseconds/nanoseconds as plain longs and convert them to whatever objects you are using ([obligatory joda-time plug](#)). One hacky way which can be done is to store the date component as one long and time component as another, for example right now would be 20100221 and 154536123. These magic numbers can be used

in SQL queries and will be portable from database to another and will let you avoid this part of JDBC/Java Date API:s entirely.

Tags: [datetime](#) ([Prev Q](#)), [java](#) ([Prev Q](#)) ([Next Q](#))

Database Design

[Skip to questions,](#)

Wiki by user [walter-mitty](#) 

The result of [database design](#)  is a plan for the construction of a database as a model of the Universe of Discourse (the “business domain” or subject area about which information will be recorded in the database).

Most databases that capture and manage semi-permanent data operate under the control of a database management system (DBMS). Prominent DBMS products are Microsoft’s SQL Server, Oracle DBMS, and IBM’s DB2. There are dozens of others. Many of the questions and answers you’ll find under this tag relate to one of these DBMS products, but some design issues are DBMS independent.

The amount of preparation and education you’ll need before building your first successful database varies widely depending on many factors. Among other factors, it depends on how ambitious your database project is and on what prior experience you bring to bear on the project. Very experienced programmers sometimes underestimate the amount of material there is to learn about database design.

Sometimes programmers learn well by trial and error, or by postponing formal learning until their second or third project. Other times, database design neophytes make design decisions that lead into pitfalls that are very difficult to reverse.

There are many ways to measure the quality of a database design. Programmers building their first database are often primarily concerned with performance. There’s no question that performance is important. A bad design can easily result in database operations that take ten to a hundred times as much time as they should.

But don’t let performance issues blind you to other aspects of good design. In particular, future proofing of a database is enormously important. Failure to do this can result in a database that traps its users at the first level and prevents their data from evolving as their needs evolve.

Another aspect involves separating out the hidden features of a database (sometimes called physical design) from the public features visible across the application interface (sometimes called logical design). A neat separation of these features can result in a database that can be tweaked and tuned quite a bit with no changes to application code. A poor separation of these features can result in a database that makes a nightmare out of application development or database administration.

Another consideration is whether the proposed database will be embedded within a single application, or whether it will be an information hub that serves the needs of multiple applications. Some design decisions will be made very differently in these two cases.

Yet another consideration is whether the application is going to perform all data management functions on behalf of its clients, or whether custodial responsibility for the

database and its data is going to be vested in one or more DBAs (database administrators).

What kinds of questions will appear in the database-design tag?

You'll see a lot of questions about table design, data normalization, index design, query optimization, constraint declarations, and keys. A lot of questions, and many of the responses will address issues of speed or performance. There will be a lot of questions about key selection.

Most of the questions are about relational databases, including the SQL databases that are commonly called relational. A few questions are about "truly relational" databases or about "non-relational" or "post-relational" databases. A few are about semistructured or unstructured data.

A lot of questions tagged "database design" will also be tagged "data modeling". There is a huge overlap between the two subjects.

You'll see a lot of questions on the subject of table composition and decomposition. Closely related to table decomposition is the concept of data normalization. Indeed, many responders treat table decomposition and data normalization as though they are synonymous terms. They aren't quite synonymous. Nearly all improvements in data normalization result in table decomposition, but there are plenty of ways of decomposing tables that have nothing to do with normalization.

Data normalization is a brand new topic to many neophyte database designers. It's worth learning the rudiments of data normalization, even if the database you are building is small and simple. It's also sometimes worthwhile to disregard the rules of data normalization, but you really have to know what you are doing.

You'll also see a lot of questions on the subject of index design. Closely related to index design is query optimization. Many questions about either index design or query design have to do with how much effort the programmer should expend in getting the very best result out of the optimizer.

Three things are worth keeping in mind. First, optimization is often a matter of tradeoffs. Sometimes organizing things for rapid query will slow down data updates. Sometimes speed really matters in some database operations, but not others.

Second, you really need to pay attention to those things that slow operations down from seconds to minutes, or from minutes to hours, before you worry about 10% improvements.

Third, database delays vary enormously as the volume of data increases and as the number of concurrent users increases. Simple tests with one user and sample data can really mislead you about speed in a production environment.

What are common database development mistakes?

1. Not using appropriate indices

This is a relatively easy one, but it still happens all the time. Foreign keys should have indexes on them. If you're using a field in a WHERE you should (probably) have an index on it. Such indexes should often cover multiple columns based on the queries you need to

execute.

2. Not enforcing referential integrity

Your database may vary here, but if your database supports referential integrity - meaning that all foreign keys are guaranteed to point to an entity that exists - you should be using it.

It's quite common to see this failure on MySQL databases.

More here:

- [How important are constraints like NOT NULL and FOREIGN KEY if I'll always control my database input with PHP?](#) 
- [Are foreign keys really necessary in a database design?](#) 

3. Missing or inappropriately chosen keys

Keys are a fundamental part of database design and data integrity. Poorly chosen keys, failure to implement keys or misunderstandings about keys are very common problems and topics of discussion. The relative merits and use of surrogate and natural keys is just one aspect. Surrogate and natural keys can both be useful tools when applied correctly but they have distinct features and functions and one key cannot necessarily be regarded as a substitute for or alternative to another. Some relevant questions and answers are:

- [How do you like your primary keys?](#) 
- [What's the best practice for primary keys in tables?](#)
- [Use item specific prefixes and autonumber for primary keys?](#) 
- [Surrogate vs. natural/business keys](#) 
- [Should I have a dedicated primary key field?](#) 

4. Writing queries that require DISTINCT to work

You often see this in ORM-generated queries. Look at the log output from [Hibernate](#)  and you'll see all the queries begin with:

```
SELECT DISTINCT...
```

This is a bit of a shortcut to ensuring you don't return duplicate rows and thus get duplicate objects. You'll sometimes see people doing this as well. If you see it too much it's a real red flag. Not that DISTINCT is bad or doesn't have valid applications. It does (on both counts), but it's not a surrogate or a stopgap for writing correct queries.

From [Why I Hate DISTINCT](#) :

Where things start to go sour in my opinion is when a developer is building substantial query, joining tables together, and all of a sudden he realizes that it **looks** like he is getting duplicate (or even more) rows and his immediate response... his "solution" to this "problem" is to throw on the DISTINCT keyword and **POOF** all his troubles go away.

5. Favoring aggregation over joins

Another common mistake is to not realize how much more expensive aggregation (that is,

the GROUP BY clause) can be compared to joins.

For example:

From [SQL statement - “join” vs “group by and having”](#):

First query:

```
SELECT userid  
FROM userrole  
WHERE roleid IN (1, 2, 3)  
GROUP by userid  
HAVING COUNT(1) = 3
```

Query time: 0.312 s

Second query:

```
SELECT t1.userid  
FROM userrole t1  
JOIN userrole t2 ON t1.userid = t2.userid AND t2.roleid = 2  
JOIN userrole t3 ON t2.userid = t3.userid AND t3.roleid = 3  
AND t1.roleid = 1
```

Query time: 0.016 s

That's right. The join version I proposed is **twenty times faster than the aggregate version.**

6. Not simplifying complex queries through views

Not all database vendors support views but for those that do, they can greatly simplify queries if used judiciously. As an example, consider a [generic Party model](#) for CRM. This is an extremely powerful and flexible modeling technique, but it can lead to many joins. In this model there were:

- **Party:** people and organizations;
- **Party Role:** things those parties did, for example Employee and Employer;
- **Party Role Relationship:** how those roles related to each other.

Example:

- Ted is a Person, being a subtype of Party;
- Ted has many roles, one of which is Employee;
- Intel is an organization, being a subtype of a Party;
- Intel has many roles, one of which is Employer;
- Intel employs Ted, meaning there is a relationship between their respective roles.

So there are five tables joined to link Ted to his employer. We assume all employees are Persons (not organizations) and provide this helper view:

```
CREATE VIEW vw_employee AS  
SELECT p.title, p.given_names, p.surname, p.date_of_birth, p2.party_name employer_name  
FROM person p  
JOIN party py ON py.id = p.id  
JOIN party_role child ON p.id = child.party_id  
JOIN party_role_relationship prr ON child.id = prr.child_id AND prr.type = 'EMPLOYMENT'  
JOIN party_role parent ON parent.id = prr.parent_id = parent.id  
JOIN party p2 ON parent.party_id = p2.id
```

And suddenly we have a very simple view of the data you want, but on a highly flexible data model.

7. Not sanitizing input

This is a huge one. If you don't know what you're doing it's really easy to create sites vulnerable to attack. Nothing sums it up better than the [story of little Bobby Tables](#).

Data provided by the user by way of URLs, form data **and cookies** should always be treated as hostile and be sanitized. Make sure you're getting what you expect.

8. Not using prepared statements

Prepared statements are when you compile a query minus the data used in inserts, updates and WHERE clauses and then supply that later. For example:

```
SELECT * FROM users WHERE username = 'bob'
```

vs

```
SELECT * FROM users WHERE username = ?
```

or

```
SELECT * FROM users WHERE username = :username
```

depending on your platform.

Basically, each time any modern database encounters a new query it has to compile it. If it encounters a query it's seen before, you're giving the database the opportunity to cache the compiled query and the execution plan. By doing the query a lot you're giving the database the opportunity to figure that out and optimize accordingly (for example, by pinning the compiled query in memory).

Using prepared statements will also give you meaningful statistics about how often certain queries are used.

Prepared statements will also better protect you against [SQL injection](#) attacks.

9. Not normalizing enough

[Database normalization](#) is the process of optimizing database design, or how you organize your data into tables.

As an example, consider code where someone implode s an array and inserts it into a single field in a database. Normalizing this would be to treat each element of the array as a separate row in a child table (that is, a one-to-many relationship).

This came up in [Best method for storing a list of user IDs](#):

I've seen in other systems that the list is stored in a serialized PHP array.

But lack of normalization comes in many forms.

More:

- [Normalization: How far is far enough?](#)

- [SQL by Design: Why You Need Database Normalization](#)

10. Normalizing too much

This may seem like a contradiction to the previous point but normalization, like many things, is a tool. It is a means to an end and not an end in and of itself. Many developers forget this and start treating a “means” as an “end”. [Unit testing](#) is a prime example of this.

Careful and considered de-normalization can have huge performance benefits, but you have to be really careful when doing this.

More:

- [Why too much Database Normalization can be a Bad Thing](#)
- [How far to take normalization in database design?](#)
- [When Not to Normalize your SQL Database](#)
- [Maybe Normalizing Isn't Normal](#)
- [The Mother of All Database Normalization Debates on Coding Horror](#)

The trouble with following advice to “denormalize” is that it doesn’t tell you what to do. It’s like trying to get to [Los Angeles](#) by driving away from [Chicago](#). You could end up almost anywhere. A better plan is to find another design discipline that will function as an alternative to normalization, with different design goals. One such alternative is [star schema](#) design. Star schema design is widely used in [data warehousing](#) and reporting databases, where speed and ease of querying outweighs simplicity of update. There is another alternative, called [snowflake design](#) which looks sort of like a compromise between star schema and normalized design.

11. Using exclusive arcs

An exclusive arc is a common mistake where a table is created with two or more foreign keys where one and only one of them can be non-null. For one thing, it becomes that much harder to maintain data integrity. After all, even with referential integrity, nothing is preventing two or more of these foreign keys from being set (complex check constraints notwithstanding).

12. Not doing performance analysis on queries at all

Pragmatism reigns supreme, particularly in the database world. If you’re sticking to principles to the point that they’ve become a dogma then you’ve quite probably made mistakes. Take the example of the aggregate queries from above. The aggregate version might look “nice”, but its performance is woeful. A performance comparison should’ve ended the debate (but it didn’t), but more to the point: spouting such ill-informed views in the first place is ignorant, even dangerous.

13. Over-reliance on UNION ALL and particularly UNION constructs

A UNION in SQL terms merely concatenates congruent data sets, meaning they have the same type and number of columns. The difference between them is that UNION ALL is a simple concatenation and should be preferred wherever possible whereas a UNION will implicitly do a DISTINCT to remove duplicate tuples.

UNIONs, like DISTINCT, have their place. There are valid applications. But if you find yourself doing a lot of them, particularly in sub-queries, then you're probably doing something wrong. That might be a case of poor query construction or a poorly designed data model forcing you to do such things.

UNIONs, particularly when used in joins or dependent sub-queries, can cripple a database. Try to avoid them whenever possible.

14. Using OR conditions in queries

This might seem harmless. After all, ANDs are OK. OR should be OK too right? Wrong. Basically, an AND condition **restricts** the data set, whereas an OR condition **grows** it, but not in a way that lends itself to optimization. Particularly when the different OR conditions might intersect thus forcing the optimizer to effectively to a DISTINCT operation on the result.

Bad:

```
... WHERE a = 2 OR a = 5 OR a = 11
```

Better:

```
... WHERE a IN (2, 5, 11)
```

Now your SQL optimizer may effectively turn the first query into the second. But it might not. Just don't do it.

15. Not designing their data model to lend itself to high-performing solutions

This is a hard point to quantify. It is typically observed by its effect. If you find yourself writing complicated queries for relatively simple tasks or that queries for finding out relatively straightforward information are not efficient, then you probably have a poor data model.

In some ways this point summarizes all the earlier ones, but it's more of a cautionary tale that doing things like query optimization is often done first when it should be done second. First and foremost you should ensure you have a good data model before trying to optimize the performance. As [Donald Knuth](#) said:

Premature optimization is the root of all evil

16. Incorrect use of Database Transactions

All data changes for a specific process should be atomic. That is, if the operation succeeds, it does so fully. If it fails, the data is left unchanged. There should be no possibility of 'half-done' changes.

Ideally, the simplest way to achieve this is that the entire system design should strive to support all data changes through single INSERT/UPDATE/DELETE statements. In this case, no special transaction handling is needed, as your database engine should do so automatically.

However, if any processes do require multiple statements be performed as a unit to keep the data in a consistent state, then appropriate transaction control is necessary.

- Begin a transaction before the first statement.
- Commit the transaction after the last statement.
- On any error, rollback the transaction. And very newbie! Don't forget to skip/abort all statements that follow after the error.

Also pay careful attention to the subtleties of how your database connectivity layer, and database engine interact in this regard.

17. Not understanding the ‘set-based’ paradigm

The SQL language follows a specific paradigm suited to specific kinds of problems. Various vendor-specific extensions notwithstanding, the language struggles to deal with problems that are trivial in languages like Java, C#, [Delphi](#), etc.

This lack of understanding manifests itself in a few ways.

- Inappropriately imposing too much procedural or imperative logic on the database.
- Inappropriate or excessive use of [cursors](#). Especially when a single query would suffice.
- Incorrectly assuming that triggers fire once per row affected in multi-row updates.

Determine clear division of responsibility, and strive to use the appropriate tool to solve each problem.

Questions

[Q: Recommended SQL database design for tags or tagging](#)

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

I've heard of a few ways to implement tagging; using a mapping table between TagID and ItemID (makes sense to me, but does it scale?), adding a fixed number of possible TagID columns to ItemID (seems like a bad idea), Keeping tags in a text column that's comma separated (sounds crazy but could work). I've even heard someone recommend a sparse matrix, but then how do the tag names grow gracefully?

Am I missing a best practice for tags?

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [dlamblin](#) 

[Answer](#)  by [yaakov-ellis](#) 

Three tables (one for storing all items, one for all tags, and one for the relation between the two), properly indexed, with foreign keys set running on a proper database, should work well and scale properly.

```
Table: Item
Columns: ItemID, Title, Content

Table: Tag
Columns: TagID, Title

Table: ItemTag
Columns: ItemID, TagID
```

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

[Q: Database Design for Tagging](#)

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

How would you design a database to support the following tagging features:

- items can have a large number of tags
- searches for all items that are tagged with a given set of tags must be quick (the items must have ALL tags, so it's an AND-search, not an OR-search)
- creating/writing items may be slower to enable quick lookup/reading

Ideally, the lookup of all items that are tagged with (at least) a set of n given tags should be done using a single SQL statement. Since the number of tags to search for as well as the number of tags on any item are unknown and may be high, using JOINs is impractical.

Any ideas?

Thanks for all the answers so far.

If I'm not mistaken, however, the given answers show how to do an OR-search on tags. (Select all items that have one or more of n tags). I am looking for an efficient AND-search. (Select all items that have ALL n tags - and possibly more.)

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [christian-berg](#) 

[Answer](#)  by [jeff-atwood](#) 

Here's a good article on tagging Database schemas:

<http://tagging.pui.ch/post/37027745720/tags-database-schemas> 

along with performance tests:

<http://tagging.pui.ch/post/37027746608/tagsystems-performance-tests> 

Note that the conclusions there are very specific to MySQL, which (at least in 2005 at the time that was written) had very poor full text indexing characteristics.

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

Q: Is there ever a time where using a database 1:1 relationship makes sense?

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

I was thinking the other day on normalization, and it occurred to me, I cannot think of a time where there should be a 1:1 relationship in a database.

Name:SSN? I'd have them in the same table PersonID:AddressID? Again, same table.

I can come up with a zillion examples of 1:many or many:many (with appropriate intermediate tables), but never a 1:1.

Am I missing something obvious?

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [pulsehead](#) 

[Answer](#)  by [godeke](#) 

A 1:1 relationship typically indicates that you have partitioned a larger entity for some reason. Often it is because of performance reasons in the physical schema, but it can happen in the logic side as well if a large chunk of the data is expected to be "unknown" at the same time (in which case you have a 1:0 or 1:1, but no more).

As an example of a logical partition: you have data about an employee, but there is a larger set of data that needs to be collected, if and only if they select to have health coverage. I would keep the demographic data regarding health coverage in a different

table to both give easier security partitioning and to avoid hauling that data around in queries unrelated to insurance.

An example of a physical partition would be the same data being hosted on multiple servers. I may keep the health coverage demographic data in another state (where the HR office is, for example) and the primary database may only link to it via a linked server... avoiding replicating sensitive data to other locations, yet making it available for (assuming here rare) queries that need it.

Physical partitioning can be useful **whenever** you have queries that need consistent subsets of a larger entity.

[Answer](#) by [kevin](#)

One reason is database efficiency. Having a 1:1 relationship allows you to split up the fields which will be affected during a row/table lock. If table A has a ton of updates and table b has a ton of reads (or has a ton of updates from another application), then table A's locking won't affect what's going on in table B.

Others bring up a good point. Security can also be a good reason depending on how applications etc. are hitting the system. I would tend to take a different approach, but it can be an easy way of restricting access to certain data. It's really easy to just deny access to a certain table in a pinch.

[My blog entry about it.](#)

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

[Q: Is there common street addresses database design for all addresses of the world?](#)

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

I am a programmer and to be honest don't know street address structures of the world, just how in my country is structured :) so which is the best and common database design for storing street addresses? It should be so simple to use, fast to query and dynamic to store all street addresses of the world which is identifying just by one id

Thanks a lot

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

User: [arsen-mkrtchyan](#) 

[Answer](#)  by [edward-ross](#) 

It is possible to represent addresses from lots of different countries in a standard set of fields. The basic idea of a named access route (thoroughfare) which the named or numbered buildings are located on is fairly standard, except in China sometimes. Other near universal concepts include: naming the settlement (city/town/village), which can be generically referred to as a locality; naming the region and assigning an alphanumeric postcode. Note that postcodes, also known as zip codes, are purely numeric only in some countries. You will need lots of fields if you really want to be generic.

The UPU Universal Postal Union provides address data for lots of countries in a [standard format](#) . Note that the UPU format holds all addresses (down to the available field precision) for a whole country, it is therefore relational. If storing customer addresses, where only a small fraction of all possible addresses will be stored, its better to use a single table (or flat format) containing all fields and one address per row.

A reasonable format for storing addresses would be as follows:

- Address Lines 1-4
- Locality
- Region
- Postcode (or zipcode)
- Country

Address lines 1-4 can hold components such as:

- Building
- Sub-Building
- Premise number (house number)
- Premise Range
- Thoroughfare
- Sub-Thoroughfare
- Double-Dependent Locality
- Sub-Locality

Frequently only 3 address lines are used, but this is often insufficient. It is of course possible to require more lines to represent all addresses in the official format, but commas can always be used as line separators, meaning the information can still be captured.

Usually analysis of the data would be performed by locality, region, postcode and country and these elements are fairly easy for users to understand when entering data. This is why these elements should be stored as separate fields. However, don't force users to supply postcode or region, they may not be used locally.

Locality can be unclear, particularly the distinction between map locality and postal-locality. The postal locality is the one deemed by a postal authority which may sometimes be a nearby large town. However, the postcode will usually resolve any problems or discrepancies there, to allow correct delivery even if the official post-locality is not used.

Tags: [database-design](#) ([Prev Q](#)) ([Next Q](#))

Q: Using a Single Row configuration table in SQL Server database. Bad idea?



Tags: [database-design](#) ([Prev Q](#))

In developing a shopping cart application I've found that I needed to save settings and configurations based on the administrator's preferences and requirements. This information can be anything from company information, Shipping account IDs, PayPal API keys, notification preferences, etc.

It seems highly inappropriate to create a table to store a single row in a relational database system.

What is the appropriate way to store this information?

Note: my DBMS is SQL Server 2008 and programming layer is implemented with ASP.NET (in C#).

Tags: [database-design](#) ([Prev Q](#))

User: [david-murdoch](#)

[Answer](#) by [adrianbanks](#)

I have done this two ways in the past - a single row table and a key/value pair table - and there are positives and negatives to each approach.

Single Row

- positive: the values are stored in the correct type
- positive: it is easier to deal with in code (due to the above)
- positive: default values can be given to each setting individually
- negative: a schema change is required to add a new setting

- negative: the table can become very wide if there are lots of settings

Key/Value Pair

- positive: adding new settings does not require a schema change
- positive: the table schema is narrow, with extra rows being used for new settings
- negative: each setting has the same default value (null/empty?)
- negative: everything has to be stored as strings (ie. nvarchar)
- negative: when dealing with the settings in code, you have to know what type a setting is and cast it

The single row option is by far the easiest one to work with. This is because you can store each setting in its correct type in the database and not have to store the types of the settings as well as their lookup keys in code.

One thing I was concerned with using this approach was having multiple rows in the “special” single row settings table. I overcame this by (in SQL Server):

- adding a new bit column with a default value of 0
- creating a check constraint to ensure that this column has a value of 0
- creating a unique constraint on the bit column

This means that only one row can exist in the table because the bit column has to have a value of 0, but there can only be one row with that value because of the unique constraint.

Tags: [database-design](#) ([Prev Q](#))

Select

[Skip to questions](#),

Wiki by user [unixman83](#) 

In query languages (SQL, ...)

[SELECT](#)  is a statement in query languages, like SQL or SPARQL. It returns a result set of records from one or more tables.

SELECT queries require two essential parts. The *first part* is the **WHAT**, which determines what we want SQL to go and fetch. The *second part* of any SELECT command is the **FROM WHERE**. It identifies where to fetch the data from, which may be from a SQL table, a SQL view, or some other SQL data object.

Reference

- [SQL Server](#) 
 - [Oracle](#) 
 - [MySQL](#) 
 - [SQLite](#) 
 - [PostgreSQL](#) 
-

In HTML

<select> is also an HTML user interface element for choosing one or more option(s) from a finite collection of elements. **Do not use this tag for this purpose, use [html-select](#) instead.**

In C and C++

`select()` is an important system call used by C and C++ programs and libraries that avoids busy waiting for network activity and other I/O to complete.

In Perl

[In Perl 5](#) , `select()` has two different uses. When called with four parameters, it calls the same syscall as C and C++. In this form it has limited portability. Otherwise it is called with one or no parameters and sets or gets current default filehandle for output. The filehandle is used e.g. by `print` if no other specified. Special variables like `$|` relate to it, too. This second form is perfectly portable.

[In Perl 6](#) it has been completely dropped.

In Go

The [select statement](#) lets a goroutine wait on multiple channel operations. A select blocks until one of its cases can run, then it executes that case.

Questions

[Q: What's the difference between HAVING and WHERE?](#)

Tags: [select](#) ([Prev Q](#)) ([Next Q](#))

I must be googling in the wrong way or I'm having a stupid moment in time.

What's the difference between `HAVING` and `WHERE` in an SQL `SELECT` statement?

EDIT: I've marked Steven's answer as the correct one as it contained the key bit of information on the link:

When `GROUP BY` is not used, `HAVING` behaves like a `WHERE` clause

The situation I had seen the `WHERE` in did not have `GROUP BY` and is where my confusion started. Of course, until you know this you can't specify it in the question.

Many thanks for all the answers which were very enlightening.

Tags: [select](#) ([Prev Q](#)) ([Next Q](#))

User: [colinyounger](#) 

[Answer](#)  by [wcm](#) 

`HAVING` is used to check conditions after the aggregation takes place.

`WHERE` is used before the aggregation takes place.

This code:

```
select City, CNT=Count(1)
From Address
Where State = 'MA'
Group By City
```

Gives you a count of all the cities in MA.

This code:

```
select City, CNT=Count(1)
From Address
Where State = 'MA'
Group By City
Having Count(1)>5
```

Gives you the count of all the cities in MA that occur 6 or more times.

Tags: [select](#) ([Prev Q](#)) ([Next Q](#))

[Q: COUNT\(*\) vs. COUNT\(1\) vs. COUNT\(pk\): which is better?](#)

Tags: [select](#) ([Prev Q](#)) ([Next Q](#))

I often find these three variants:

```
SELECT COUNT(*) FROM Foo;
SELECT COUNT(1) FROM Foo;
SELECT COUNT(PrimaryKey) FROM Foo;
```

As far as I can see, they all do the same thing, and I find myself using the three in my codebase. However, I don't like to do the same thing different ways. To which one should I stick? Is any one of them better than the two others?

Tags: [select](#) ([Prev Q](#)) ([Next Q](#))

User: [zneak](#) 

[Answer](#)  by [michael-buen](#) 

Use * for all your queries that need to count everything, even for joins, use *

```
SELECT boss.boss_id, COUNT(subordinate.*)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

But don't use COUNT(*) for LEFT joins, as that will return 1 even if the subordinate table doesn't match anything from parent table

```
SELECT boss.boss_id, COUNT(*)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

Don't be fooled by those advising that when using * in COUNT, it fetches entire row from your table, saying that * is slow. The * on SELECT COUNT(*) and SELECT * has no bearing to each other, they are entirely different thing, they just share a common token, i.e. *. In fact, if it is not permitted to name a field as same as its table name, RDBMS language designer could give COUNT(tableNameHere) the same semantics as COUNT(*). Example:

For counting rows we could have this:

```
SELECT COUNT(emp) FROM emp
```

And they could make it simpler:

```
SELECT COUNT() FROM emp
```

And for LEFT JOINS, we could have this:

```
SELECT boss.boss_id, COUNT(subordinate)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

But they cannot do that (COUNT(tableNameHere)) since SQL standard permits naming a field with the same name as its table name:

```
CREATE TABLE fruit-ORM-friendly name
(
fruit_id int NOT NULL,
fruit varchar(50), /* same name as table name,
                     and let's say, someone forgot to put NOT NULL */
shape varchar(50) NOT NULL,
color varchar(50) NOT NULL
)
```

And also, it is not a good practice to make a field nullable, say you have values 'Banana',

‘Apple’, NULL, ‘Pears’ on fruit field. This will not count all fruits, it will only yield 3, not 4

```
SELECT count(fruit) FROM fruit
```

Though some RDBMS do that sort of principle(for counting the table’s rows, it accepts table name as COUNT’s parameter), this will work in Postgresql (if there is no subordinate field in any of the two tables below, i.e. as long as there is no name conflict between field name and table name):

```
SELECT boss.boss_id, COUNT(subordinate)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

But that could cause confusion later if we will add a subordinate field in the table, as it will count the field(which could be nullable), not the table rows.

So to be on the safe side, use:

```
SELECT boss.boss_id, COUNT(subordinate.*)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

[EDIT]

In particular to COUNT(1), it is a **one-trick pony**, it works well only on one table query:

```
SELECT COUNT(1) FROM tbl
```

But when you use joins, that trick won’t work on multi-table queries without its semantics being confused, and in particular you cannot write:

```
-- count the subordinates that belongs to boss
SELECT boss.boss_id, COUNT(subordinate.1)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

So what’s the meaning of COUNT(1) here?

```
SELECT boss.boss_id, COUNT(1)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

Is it this...?

```
-- counting all the subordinates only
SELECT boss.boss_id, COUNT(subordinate.boss_id)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

Or this...?

```
-- or is that COUNT(1) will also count 1 for boss regardless if boss has a subordinate
SELECT boss.boss_id, COUNT(*)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

Though it isn’t hard to infer (though some could be confused) that COUNT(1) is the same as COUNT(*) regardless of type of join. But for LEFT JOINs result, we cannot mold COUNT(1)

to work as: COUNT(subordinate.boss_id), COUNT(subordinate.*)

So just use either of the following:

```
-- count the subordinates that belongs to boss
SELECT boss.boss_id, COUNT(subordinate.boss_id)
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

Works on Postgresql, it's clear that you want to count the cardinality of the set

```
-- count the subordinates that belongs to boss
SELECT boss.boss_id, COUNT(subordinate.* )
FROM boss
LEFT JOIN subordinate on subordinate.boss_id = boss.boss_id
GROUP BY boss.id
```

Another way to count the cardinality of the set, very English-like (just don't make a column with a name same as its table name) : <http://www.sqlfiddle.com/#!1/98515/7> 

```
select boss.boss_name, count(subordinate)
from boss
left join subordinate on subordinate.boss_code = boss.boss_code
group by boss.boss_name
```

You cannot do this: <http://www.sqlfiddle.com/#!1/98515/8> 

```
select boss.boss_name, count(subordinate.1)
from boss
left join subordinate on subordinate.boss_code = boss.boss_code
group by boss.boss_name
```

You can do this, but this produces wrong result: <http://www.sqlfiddle.com/#!1/98515/9> 

```
select boss.boss_name, count(1)
from boss
left join subordinate on subordinate.boss_code = boss.boss_code
group by boss.boss_name
```

Bottom Line

Use either COUNT(field) or COUNT(*), and stick with it consistently, and if your database allows COUNT(tableHere) or COUNT(tableHere.*), use that. In short, don't use COUNT(1) for anything

Tags: [select](#) ([Prev Q](#)) ([Next Q](#))

Q: SQL/mysql - Select distinct/UNIQUE but return all columns?

Tags: [select](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#))

```
SELECT DISTINCT field1, field2, field3, ..... FROM table
```

I am trying to accomplish the following sql statement but I want it to return all columns is this possible? Something like:

```
SELECT DISTINCT field1, * from table
```

Tags: [select](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#))

User: [aryaxt](#) 

[Answer](#) by [denis-de-bernardy](#)

You're looking for a group by:

```
select *
from table
group by field1
```

Which can occasionally be written with a distinct on statement:

```
select distinct on field1 *
from table
```

On most platforms, however, neither of the above will work because the behavior on the other columns is unspecified. (The first works in MySQL, if that's what you're using.)

You could fetch the distinct fields and stick to picking a single arbitrary row each time.

On some platforms (e.g. PostgreSQL, Oracle, T-SQL) this can be done directly using window functions:

```
select *
from (
  select *,
    row_number() over (partition by field1 order by field2) as row_number
  from table
) as rows
where row_number = 1
```

On others (MySQL, SQLite), you'll need to write subqueries that will make you join the entire table with itself ([example](#)), so not recommended.

Tags: [select](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQL SELECT WHERE field contains words](#)

Tags: [select](#) ([Prev Q](#))

I need a select which would return results like this:

```
SELECT * FROM MyTable WHERE Column1 CONTAINS 'word1 word2 word3'
```

And I need all results, including the strings with 'word2 word3 word1' or 'word1 word3 word2' All words needs to be in the result.

Tags: [select](#) ([Prev Q](#))

User: [mario-m](#)

[Answer](#) by [mvp](#)

Rather slow, but working method to include **any** of words:

```
SELECT * FROM mytable
WHERE column1 LIKE '%word1%'
  OR column1 LIKE '%word2%'
  OR column1 LIKE '%word3%'
```

If you need **all** words to be present, use this:

```
SELECT * FROM mytable
WHERE column1 LIKE '%word1%'
AND column1 LIKE '%word2%'
AND column1 LIKE '%word3%'
```

If you want something faster, you need to look into full text search, and this is very specific for each database type.

Tags: [select](#) ([Prev Q](#))

PHP

[Skip to questions](#),

Wiki by user [charles](#)

[PHP: Hypertext Preprocessor](#) is a widely used, general-purpose [scripting-language](#) that is most suited for web development. The original PHP project, as invented by Rasmus Lerdorf, stood for Personal Home Page.

The latest stable release, development changes, and development branches can be found on [the PHP website](#). The PHP project is written in [c](#), and you can obtain its source code from [PHP's GitHub repository](#).

For a quick start with [php](#), you might use one of these applications, which include a web server and [PHP](#), among other components:

- [AMPPS](#)
- [XAMPP](#) (for Windows, Mac OS X & Linux)
- [MAMP](#) (for Mac OS)
- [WampServer](#) (for Windows)
- [easyPHP](#) (for Windows)
- [USBWebserver](#)
- [OpenServer](#) (for Windows)
- [PHPStudy](#) (for Windows)
- [WPN-XM](#) (for Windows)

As of version 5.4.0, [php](#) provides a built-in web server, which can be started using following command:

```
php -S localhost:8000
```

After executing the above command, the server will listen on port 8000 using current working directory as its document root. See the [PHP manual](#) for more information.

Notice: To make an online demo for your question, you may use [codepad](#), [3v4l](#) or [PHP Sandbox](#) which act like [jsfiddle](#), but for [php](#). However, all relevant code should still be included in your question or answer.

PHP Versions

Current Stable Version: 7.0.4 // *Release Date: 3 March 2016*

Old Stable Version: 5.6.19 // *Release Date: 3 March 2016*

Old Stable Version: 5.5.33 // *Release Date: 3 March 2016*

It is recommended to use the current stable released version. Only this version contains all

security and bug fixes. All versions below 5.5 are officially [unsupported and have been announced end-of-life](#). A list of supported branches and their maintenance status can be found [here](#).

For further information about new features and required changes in a new version see the official migration docs:

- [5.2.x->5.3.x](#)
- [5.3.x->5.4.x](#)
- [5.4.x->5.5.x](#)
- [5.5.x->5.6.x](#)
- [5.6.x->7.0.x](#)

PHP 6

On [July, 30th, 2014 a majority of the PHP steering group decided to skip version 6](#) to avoid confusion with an earlier but abandoned PHP 6 project (dubbed the Unicode release). While there never was any official release of PHP 6, many books and articles had been published already.

Sample PHP script

This script displays Hello World! on your screen.

```
<?php
    echo 'Hello World!';
?>
```

To run this script in a console, save it in current working directory in a file hello.php and simply execute command: php hello.php.

Interactive shells

For a interactive testing of [php](#) code in a [REPL](#) shell, check out <http://psysh.org/>. With that, you will be able to test functions and code snippets without losing context for variables, and execution won't be interrupted because of exceptions or errors.

PHP configuration information

This script displays config information:

```
<?php
    phpinfo();
?>
```

Save the file in your web server's document root as phpinfo.php, and run it from the browser. <http://localhost/phpinfo.php>

Remember to delete it afterwards for security.

Community

[php](#) has many active community forums, including:

- [Stack Overflow PHP chat room](#) 
- [Linux Fund](#) 
- [PHP Meetups](#) 
- freenode IRC channel [##php](#) 

More information

- [Wikipedia on PHP](#) 
- [PHP Official Page](#) 

Online documentation

The [PHP manual](#)  is the official documentation for [the language syntax](#) , featuring function search and URL shortcuts (for example <http://php.net/explode> ). [The API is well documented](#)  for bundled and additional extensions. Most additional extensions can be found in [PECL](#) . The [PEAR](#)  repository contains a plethora of community supplied classes. It is also possible to download an offline version of the documentation [here](#) .

Additionally, the PHP Framework Interop Group (PHP-FIG) has created sets of standards with regards to PHP coding styles and standards. These PHP Standard Recommendations (PSRs) can be found [here](#) .

PHP Tutorials

- [PHP Basics](#) 

PHP security related information

- [PHP Security Cheat Sheet](#) , by [OWASP](#) 

Free online tutorials

- [PHP-Codecademy](#) 
- [Codecourse](#) 
- [PHP Video Tutorial](#) 

Free PHP Programming Books

- [PHP Essentials](#) 
- [Practical PHP Programming](#)  (wiki containing O'Reilly's *PHP In a Nutshell*)
- [Symfony2](#) 
- [Zend Framework: Survive the Deep End](#) 

- [PHP: The Right Way](#) (a community-driven quick reference for PHP best practices and accepted coding standards)

Best PHP Books 2012

- [Learning PHP, MySQL, and JavaScript: A Step-By-Step Guide to Creating Dynamic Websites \(Animal Guide\)](#)
- [PHP Solutions: Dynamic Web Design Made Easy](#)
- [PHP and MySQL Web Development \(4th Edition\)](#)
- [Murach's PHP and MySQL \(Murach: Training & Reference\)](#)
- [PHP and MySQL for Dynamic Web Sites: Visual QuickPro Guide \(4th Edition\)](#)

Database support

PHP supports a wide range of [databases](#), [relational and non-relational alike](#).

PHP is often paired with the [MySQL](#) relational database. PHP also includes great database support for [PostgreSQL](#), [SQLite](#), [Microsoft SQL Server](#) ([API reference](#)), [Oracle](#), [IBM DB2 & Cloudscape](#), [Apache Derby](#) and even [ODBC](#).

All modern versions of PHP include [PDO](#), a built-in [data-access abstraction library](#) with [comprehensive connectivity options](#). More recently, PECL extensions have surfaced that offer “NoSQL” database support, including [Apache Thrift](#) (for [Apache Cassandra](#)), [MongoDB](#), [Redis](#), and others.

SAPI support

Like many other applications, web servers provide APIs (Application Programming Interfaces) to extend their functionality via third-party extensions. An API provided by a web server is generally referred as a SAPI (Server API). By utilizing different SAPIs, PHP can work on many different web servers as a server-side scripting language. The list of web servers associated with supported SAPIs includes:

- [Apache HTTP Server \(versions 1.3.x, 2.x\)](#)
- [Microsoft IIS](#)
- [thttpd \(tiny/turbo/throttling HTTP server\)](#)
- [Caudium](#)
- [AOLserver 3.1 or later](#)
- [Netscape Enterprise Web Server, Sun Java System Webserver, SUN/Netscape Alliance/Oracle iPlanet Web Server](#)
- [LiteSpeed Web Server](#)
- [Pi3Web](#)
- [Roxen WebServer](#)
- [WebJames HTTP server](#)
- [TUX web-server](#)
- Continuity

- Phttpd
- Embed (as of version 5.4.0 PHP provides built-in web server)

PHP can be run on many other servers. For PHP to be run on a given web server, it is only required that a server implements one of the supported SAPIs. For example, the Sambar web server can interact with PHP through the SAPI which is native to Microsoft IIS (internally called *isapi* in PHP). In addition, if specific web server SAPI is not supported, one of the generic interfaces, which are widely supported by web servers, can be used. These include:

- CGI (Common Gateway Interface)
- FastCGI
- FPM (FastCGI Process Manager)

PHP is intended for developing server-side web applications, but stand-alone and client-side applications can be developed as well. To be consistent with its server-oriented architecture, PHP defines and implements CLI (Command Line Interface) SAPI for these applications.

PHP can be also used as a Sendmail milter.

Useful Third-party Code and Tools

In addition to the vast functionality provided in the PHP Core and through PEAR and PECL, there are a number of noteworthy third party contributions to the PHP world, some of which are listed below.

Package Management with Composer

[Composer](#)  is a package management tool for PHP, inspired by npm for NodeJS and Bundler for Ruby. It allows for per-project dependencies to be specified in a JSON file.

Composer uses packages from [Packagist](#) , which is rapidly growing to contain many of the most popular PHP libraries.

Composer solves the following problems:

1. You have a project that depends on a number of libraries.
2. Some of those libraries depend on other libraries.
3. You declare the things you depend on.
4. Composer determines which versions of which packages need to be installed, and downloads them into a directory (usually `vendor`) in your project.

Nothing comes for free. Software downloaded with Composer may have bugs, just like any other, including security vulnerabilities. It is your responsibility to be aware of what you install, and update to include security fixes when necessary.

Frameworks

PHP has a variety of object-oriented web application frameworks that provide a lot of the common functionality required to build modern web application out of the box, with the more well-known being:

- [Agavi](#)
- [Aura](#)
- [CakePHP](#)
- [CodeIgniter](#)
- [Fat-Free](#)
- [Flow3](#)
- [FuelPHP](#) (Kohana-like HMVC Framework that comes with an ORM and Code Generator)
- [Kohana](#)
- [Laravel](#)
- [Limonade](#)
- [Lithium](#)
- [Nette](#)
- [Phalcon](#)
- [Symfony](#)
- [xAjax](#) (simplifies AJAX programming using PHP)
- [Yii](#)
- [Zend Framework](#)
- [Zeta Components \(formerly ezComponents\)](#)
- [more ...](#)

Content Management Systems

Many different content management systems have been developed for PHP, many of which can be used by people who are not developers. Some of them are open source while others are proprietary.

- [CMS Made Simple](#)
- [Drupal](#)
- [Joomla!](#)
- [WordPress](#)
- [more ...](#)

Object-Relational Mapping

Object relational maps try to solve or mitigate the [object-relational impedance mismatch](#) problem by transparently mapping between table structures in a database and business objects in an application. The more prominent ORMs in the PHP world are:

- [Doctrine](#)
- [Eloquent](#)
- [Propel](#)
- [Redbean](#)

- [more ...](#)

Popular questions and answers

- [Good PHP ORM Library?](#)
- [more ...](#)

Quality Assurance Tools

In recent years, there have been a steady increase of quality assurance tools in the PHP world. As PHP moved into the professional mainstream, tools to assert certain quality features and [metrics](#) were needed and provided by the PHP community. These tools include frameworks for [debugging](#), [unit-testing](#), [code analysis](#) and [coverage](#), [continuous integration](#) and other aspects of professional development, some of which are listed below:

- [PHPUnit](#)
- [PEAR CodeSniffer](#)
- [SimpleTest](#)
- [Xdebug](#)
- [more ...](#)

Popular questions and answers

- [What are unit tests and why should I care?](#)
- [How do you debug PHP scripts?](#)
- [Recommended server for continuous integration for a PHP project](#)

IDEs

An [Integrated Development Environment \(IDE\)](#) is software that provides most of the tools needed to produce other software in a convenient package. Standard features of IDEs are usually source code editors with syntax highlighting, code completion as well as debugging functionality, build assistance, version control system integration, etc. Some of the major known PHP IDEs are:

- [Eclipse with PDT](#) (free)
- [NetBeans](#) (free)
- [Aptana Studio](#) (free)
- [phpstorm](#) (commercial)
- [PhpED](#) (commercial)
- [PhpEdit](#) (commercial)
- [Zend Studio](#) (commercial, Eclipse-based)
- [Coda](#) (commercial, Mac users)
- [Sublime Text](#)
- [ActiveState Komodo IDE](#) (commercial, multi-platform)
- [ActiveState Komodo Edit](#) (free)

- [Adobe Dreamweaver](#) (commercial)
- [more ...](#) (includes editors as well)

Popular questions and answers

- [What is the best IDE for PHP?](#)
 - [more ...](#)
-

Frequently Asked Questions

Find some answers to some of the more frequently asked questions about PHP below.

I have a question about PHP 6

All development on the PHP 6 branch has been cancelled. Version 6 was skipped. The next major release after PHP 5 is PHP 7. Please refrain from asking questions about PHP 6.

I have a typical “does not work” problem. What should I do before asking a question?

The causes of a typical “does not work” or “works on localhost but does not work on production server” problem can easily be diagnosed by setting the [display_errors](#) value to On or 1.

An incorrectly configured development server and a typical production server has `display_errors` turned off. This suppresses all error messages, which is not helpful when trying to diagnose problems.

Before you post your question, find all error messages reported by PHP and see if you can fix them yourself. Errors such as “PHP Warning: ...” and “You have an error in your SQL syntax; check the manual ...” might be fairly simple to fix. Google the error message and search for it on Stack Overflow. Most error messages have well known solutions. Additionally, many of the most commonly encountered error messages have been compiled into one reference. Look it over to see if your error is already addressed there. See [Reference: What does this error mean in PHP?](#)

`display_errors` can be set to On via multiple ways. On your development machine you should set the value by editing `php.ini` file. On a production server, if it is not possible (or desirable) to edit this file, add these lines at the beginning of the script:

```
error_reporting(-1); // to enable all errors  
ini_set('display_errors', 1);  
ini_set('display_startup_errors', 1);
```

Make sure to list any errors you cannot fix on your own in your question.

Make sure you are not hiding errors by using the `@` syntax. This is not good practice, particularly during development when you want to be advised of errors. From the [manual](#):

Warning!

Currently the “@” error-control operator prefix will even disable error reporting for critical errors that will terminate script execution. Among other things, this means that if you use “@” to suppress errors from a certain function and either it isn’t available or has been mistyped, the script will die right there with no indication as to why.

You can quickly override this behaviour with `set_error_handler("var_dump");` which crudely prints all error messages irrespective of @ decorators, or disabled error_reporting.

Is PHP vulnerable? How do I secure my PHP application?

When it comes to security you can’t secure your application to be 100% safe. A 100% safe application does not exist in the real world, but you can reduce the chances of your application being hacked by following below mentioned standards:

- [Where PHP Security Could Fail Or Be Vulnerable Part 1](#)
- [Where PHP Security Could Fail Or Be Vulnerable Part 2](#)

In the above links, common vulnerabilities and countermeasures are mentioned.

How do I make my database queries secure from SQL injection?

In a nutshell, use *prepared statements* and *placeholders*. When you can’t, ensure values are properly escaped. Each database interface follows the same general pattern:

- **Prepare** a SQL statement containing placeholders
- **Bind** PHP variables to those placeholders
- **Execute** the prepared statement

Some interfaces allow you to skip the bind step by providing the list of replacements for the placeholders during execute. For reference:

- **PDO** [prepare](#), [bind](#) (optional), [execute](#)
- **MySQLi** [prepare](#), [bind](#), [execute](#)
- **PostgreSQL** [prepare](#), bind (none), [execute](#)
- **SQLite 3** [prepare](#), [bind](#), [execute](#)
- **MS SQL Server (5.3+)** [prepare and bind \(combined\)](#), [execute](#)
- **Oracle** [prepare](#), [bind](#), [execute](#)
- **IBM DB2** [prepare](#), [bind](#), [execute](#)

Notice: the old ext/mysql extension does not support prepared statements. It was [deprecated](#) in PHP 5.5 and removed in PHP 7. Use [PDO](#) or [MySql](#) instead.

See Also

- [Best way to prevent SQL Injection in PHP](#)

- <http://bobby-tables.com/php.html>

My special characters come out all screwed up. Why?

You need to learn about what character sets are and use encodings properly throughout your script. Some resources to get you started on the topic:

- [UTF-8 all the way through](#)
- [PHP charset/encoding FAQ](#)
- [What Every Programmer Absolutely, Positively Needs To Know About Encodings And Character Sets To Work With Text](#)
- [Handling Unicode Front To Back In A Web App](#)

“Headers have already been sent...”

You’re outputting content and triggering PHP’s default Content-type: text/html header before making your own call to `header()`. There are a few ways this can happen. Check for a stray echo or print, white space before and after the `<?php` and `?>` tags, or a Unicode BOM. Consider using output buffering. With many scripts, there is no need to include the ending `?>` and the problem is easily fixed by removing the ending `?>` tag from your files. See [How to fix “Headers already sent” error in PHP](#).

How do I make my E-Mails secure from [E-Mail injection](#)?

Validate your input! In recent versions of PHP, `mail()` is no longer vulnerable to e-mail header injection through the subject line or receiver, as it removes all the control characters, but you may want to make sure the receivers list doesn’t include multiple addresses if that’s not what you desire. If you allow the user to specify part of the body of the message, you must also validate or sanitize it so that he’s unable to, e.g. add new MIME parts or end any prematurely. Therefore, for building MIME messages, you’re strongly encouraged to use a library. See, for instance, [PEAR Mail_Mime](#).

Can I protect my PHP code from theft? If so, how?

There is no *effective* technical solution to protect, encode or encrypt PHP source code. There are many products that offer some levels of protection, like [IonCube](#) (commercial) and [Zend Guard](#) (commercial), but *all* can be broken with time and effort. Your best option is not a technical solution, but a legal solution in the form of a license agreement.

How can strings be written in PHP?

There are [four ways to write strings literals in PHP](#). Each method is slightly different in terms of escape sequences and string interpolation. For instance, `'\n'` is a literal two-character string with characters \ and n, while `"\n"` is a one-character string with a [LF](#) character. All of them produce values of type `string` (which is actually a byte array, as it

completely encoding unaware).

When/why does the `mail()` function not work?

The `mail()` function depends on correct server configuration. See [PHP mail form doesn't complete sending e-mail](#) for a list of common reasons. Then investigate [PHPMailer/SwiftMailer](#) as alternative.

What does a specific Operator mean in PHP?

See this [Community Wiki](#) for a helpful list.

How do I parse HTML/XML with PHP?

See [How do you parse and process HTML/XML in PHP?](#) and [Best XML Parser for PHP](#).

How do I turn on error reporting to find the source of problems?

See [PHP production server - turn on error messages](#). You typically need just `error_reporting(E_ALL);` at the top of your script, unless it's a parsing error there. As very crude method `set_error_handler("var_dump");` overrides all uncommon disable options. And in case of a HTTP 500 internal server error, or a completely blank page, you often want to look into the web servers `error.log` first.

How do I fix “eregi is deprecated” messages and switch to `preg_match()`?

See [How can I convert ereg expressions to preg in PHP?](#).

Why does `mysqli_fetch_array()` or `mysqli_fetch_assoc()` return only one row?

Many RDBMS fetching functions must be called in a loop until the query result resource returns FALSE.

[Skip code block](#)

```
// A common looping pattern, appending result rows onto the array $rowset
$result = mysqli_query($link, "SELECT column1, column2 FROM table");
if ($result) {
    $rowset = array();
    while ($row = mysqli_fetch_array($result)) {
        // Append the current row onto $rowset
        $rowset[] = $row;
    }
} else {
    echo mysqli_error($link);
}
```

Can I fill class properties with `new object()` calls or function calls?

No. See [Instance as a static class property](#); see [Why don't PHP attributes allow functions?](#) for an attempt at an explanation.

How do I fix the error mysqli_fetch_array() or mysqli_fetch_assoc() expects parameter 1 to be resource, boolean given?

This is caused by a failed query, which is typically indicative of a syntax error in the SQL statement. Another common cause is using a [MySQL reserved keyword](#) as a table or column name without surrounding it with backticks. For example:

```
$query = 'SELECT order FROM my_table'; // This will fail  
$query = 'SELECT `order` FROM my_table'; //This will work
```

Use `echo mysqli_error($link)` to view the error message that was generated by MySQL.

How do I validate an email address?

See [How to validate an email address in PHP](#). As of PHP 5.2 you can simply use the `filter_var()` function with the `FILTER_VALIDATE_EMAIL` filter.

How do I decode and extract JSON data?

See [How do I extract data from JSON with PHP?](#) and the [json_decode\(\)](#) examples in the manual on how to traverse the resulting object/array structures.

Why does my assigned session variable not appear in the next script?

Session variables require a session to be started. Please review the manual-page of [session_start\(\)](#).

Additionally, cookies need to be enabled, as explained on [this Google page](#).

What does my error message mean:

Many of the most commonly encountered error messages in PHP have been compiled into one reference.

See Reference: [What does this error mean in PHP?](#)

How do I sort arrays?

You wouldn't believe how often this question comes up, so we made a reference for it:

[Reference: all basic ways to sort arrays in PHP](#)

How can I retrieve a value from a complex nested arrays and/or object?

You must understand the structures reported by debugging utility functions `var_dump()` and `print_r()`, and the differences in how arrays and `stdClass` objects are displayed.

See Reference: [How can I access an array/object](#)

Built In Array and String Function in PHP

There are a number of built in functions in PHP to manipulate a string or a array. A few basics functions include:

[str_replace](#) - String Replace function

Replaces the contents and occurrence of something or a part of string with other.

[strlen](#) - Calculate string length

used to calculate the character length of the string

[and more string functions](#)

[is_array](#) - finds if the variable is array or not

[and more array functions](#)

Reference

- [PHP Manual](#)
 - [PHP Wikipedia page](#)
-

Questions

[Q: How to debug PDO database queries?](#)

Tags: [php](#) ([Prev Q](#))

Before moving to PDO, I created SQL queries in PHP by concatenating strings. If I got database syntax error, I could just echo the final SQL query string, try it myself on the database, and tweak it until I fixed the error, then put that back into the code.

Prepared PDO statements are faster and better and safer, but one thing bothers me: I never see the final query as it's sent to the database. When I get errors about the syntax in my Apache log or my custom log file (I log errors inside a catch block), I can't see the query that caused them.

Is there a way capture the complete SQL query sent by PDO to the database and log it to a file?

Tags: [php](#) ([Prev Q](#))

User: [nathan-long](#) 

[Answer](#)  by [pascal-martin](#) 

You say this :

I never see the final query as it's sent to the database

Well, actually, when using prepared statements, **there is no such thing as a “final query”** :

- First, a statement is sent to the DB, and prepared there
 - The database parses the query, and builds an internal representation of it
- And, when you bind variables and execute the statement, only the variables are sent to the database
 - And the database “injects” the values into its internal representation of the statement

So, to answer your question :

Is there a way capture the complete SQL query sent by PDO to the database and log it to a file?

No : as there is no “complete SQL query” anywhere, there is no way to capture it.

The best thing you can do, for debugging purposes, is “re-construct” an “real” SQL query, by injecting the values into the SQL string of the statement.

What I usually do, in this kind of situations, is :

- echo the SQL code that corresponds to the statement, with placeholders
- and use var_dump (*or an equivalent*) just after, to display the values of the parameters
- This is generally enough to see a possible error, even if you don't have any "real" query that you can execute.

This is not great, when it comes to debugging — but that's the price of prepared statements and the advantages they bring.

[Answer](#)  by [nathan-long](#) 

Looking in the database log

Although **Pascal MARTIN** is correct that PDO doesn't send the complete query to the database all at once, **ryeguy**'s suggestion to use the DB's logging function actually allowed me to see the complete query as assembled and executed by the database.

Here's how: (These instructions are for MySQL on a Windows machine - your mileage may vary)

- In `my.ini`, under the `[mysqld]` section, add a `log` command, like `log="C:\Program Files\MySQL\MySQL Server 5.1\data\mysql.log"`
- Restart MySQL.
- It will start logging every query in that file.

That file will grow quickly, so be sure to delete it and turn off logging when you're done testing.

Tags: [php](#) ([Prev Q](#))

C#

[Skip to questions](#),

Wiki by user [jon-skeet](#) 

[C#](#) is a *multi-paradigm, managed, object-oriented programming* language created by **Microsoft** in conjunction with the [.net](#) framework. C# can also be used with non-Microsoft implementations, such as [mono](#) and [xamarin](#).

Versions 1.0/1.2 and 2.0 of C# were submitted and approved as both [ECMA](#) and [ISO/IEC](#) standards. As of August 2013, there are no ECMA or ISO/IEC specifications for C# 3.0, 4.0 or 5.0, but language specifications are available from Microsoft for C# 3.0 and C# 5.0.

The language's type system was originally static, with only explicit variable declarations allowed. The introduction of [var](#) (C# 3.0) and [dynamic](#) (C# 4.0) allow it to use type inference for implicit variable typing, and to consume dynamic type systems, respectively. Delegates (especially with lexical closure support for anonymous methods (C# 2.0) and lambda expressions (C# 3.0)) allow the language to be used for functional programming. C# 5.0 introduced the [async](#) and [await](#) keywords to simplify the use of asynchronous function calls.

Compilation is usually done into the Microsoft Intermediate Language (MSIL), which is then JIT-compiled to native code (and cached) during execution in the Common Language Runtime (CLR). However, options like [NGen](#) (for the Microsoft .NET Framework) and [AOT](#) (for Mono) mean that C# code can be directly compiled into the native image. Additionally, some frameworks (e.g. the .NET Micro Framework) act as CIL interpreters, with no JIT.

Generics in [C#](#) are provided in part by the runtime, unlike C++ templates (templates are resolved at compile time), or Java's generics (which use type-erasure).

With the combination of [Microsoft .NET](#) for Windows (desktop/server/mobile), Mono (desktop/server/mobile), Silverlight / Moonlight (browser/mobile), Compact Framework (mobile), and Micro Framework (embedded devices), it is available for a wide range of platforms.

In November 2014, Microsoft announced the decision to [Open Source .NET](#) with Apache 2.0 Open Source licensing and to begin supporting iOS, Linux, and Android in addition to Windows as platforms for .NET 2015 (5.0) and ASP.NET 5.0. As a result C# can now target all these platforms using a single code base through Visual Studio 2015.

Hello World Example:

```
using System;  
  
class Hello  
{  
    static void Main()  
    {
```

```
        Console.WriteLine("Hello, World!");
    }
}
```

Hello World example using classes:

[Skip code block](#)

```
using System;

namespace HelloWorldUsingClasses
{
    class ExampleClass
    {
        string exampleString = "Hello World!";
        public ExampleClass()
        {
            Console.WriteLine(exampleString);
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            ExampleClass objHelloWorld = new ExampleClass();
        }
    }
}
```

Chat Room

- [Stack Overflow chat room for C#](#) 

FAQs

- [How to send email in ASP.NET C#](#) 
- [What are the correct version numbers for C#?](#) 
- [Why is Random giving the same results every time?](#) 
- [Captured variables in loops](#) 
- [In C#, is it possible to cast a List<Child> to List<Parent>?](#) 
- [Does C# pass objects by reference?](#) 
- [Why does the compiler complain about my conditional expression \(a == b ? x : y\)?](#) 
- [Why do I get a NullReferenceException? \(Object reference not set to instance of object\)](#) 
- [What is an “index out of range” exception, and how do I fix it?](#) 
- [What's new in C# 4.0?](#) 
- [What's new in .NET 4.5?](#) 
- [What's new for C# in Visual Studio 2012?](#) 
- [New Features in C# 6](#) 
- [Useful Features in C#](#) 

Resources

- [Specification](#)
 - [Wikipedia Article](#)
 - [.NET Projects on GitHub](#)
 - [Eric Lippert's old blog](#) || [Eric Lippert's new blog](#)
 - [Jon Skeet's blog](#)
 - [James Michael Hare's C#/NET Little Wonders & Little Pitfalls series](#)
 - [Programming Guide](#)
 - [Getting Started with Visual C#](#)
 - [C# Fundamentals: Development for Absolute Beginners](#)
 - [Visual C# .Net for Beginners](#)
 - [csharp.2000things](#)
 - [Dot Net Perls](#)
-

Books

- [CLR via C#](#)
 - [C# 5.0 in a Nutshell: The Definitive Reference](#)
 - [C# in Depth](#)
 - [Accelerated C#](#)
 - [Head First C#](#)
 - The C# Programming Language ([3rd Edition](#), [4th Edition](#))
 - [Framework Design Guidelines](#)
 - Essential C# ([4.0 \(3rd Edition\)](#), [5.0 \(4th Edition\)](#))
 - [Pro C# 2010 and the .NET 4 Platform](#)
 - [MCTS Self-Paced Training Kit \(Exam 70-536\): Microsoft® .NET Framework 2.0 Foundation](#)
 - [C# How to Program](#)
 - [Visual C# .NET Step by Step](#)
 - [Programming Microsoft Windows with C#](#)
 - [C# 4.0 Unleashed](#)
 - [Pro C# 5.0 and the .NET 4.5 Framework \(Professional Apress\)](#)
 - [Learning C# 3.0](#)
 - [Starting out with Visual C# 2012 \(with CD-Rom\) \(3rd Edition\)](#)
 - [Programming C# 5.0 — Building Windows 8, Web, and Desktop Applications for the .NET 4.5 Framework](#)
 - [C# 4.0 The Complete Reference](#)
 - [C# Design Pattern Essentials](#)
 - [C# 6.0 and the .NET 4.6 Framework](#)
-

Tutorials

- [MSDN](#)
 - [Microsoft Virtual Academy](#)
-

Demo Projects

- [Codeplex](#)
-

Questions

[Q: LINQ to SQL - Left Outer Join with multiple join conditions](#)

Tags: [c#](#) ([Prev Q](#)) ([Next Q](#)), [linq-to-sql](#) ([Prev Q](#)) ([Next Q](#)), [linq](#) ([Next Q](#))

I have the following SQL, which I am trying to translate to LINQ:

```
SELECT f.value
FROM period as p
LEFT OUTER JOIN facts AS f ON p.id = f.periodid AND f.otherid = 17
WHERE p.companyid = 100
```

I have seen the typical implementation of the left outer join (ie. `into x from y in x.DefaultIfEmpty()` etc.) but am unsure how to introduce the other join condition (`AND f.otherid = 17`)

EDIT

Why is the `AND f.otherid = 17` condition part of the JOIN instead of in the WHERE clause? Because f may not exist for some rows and I still want these rows to be included. If the condition is applied in the WHERE clause, after the JOIN - then I don't get the behaviour I want.

Unfortunately this:

```
from p in context.Periods
join f in context.Facts on p.id equals f.periodid into fg
from fgi in fg.DefaultIfEmpty()
where p.companyid == 100 && fgi.otherid == 17
select f.value
```

seems to be equivalent to this:

```
SELECT f.value
FROM period as p
LEFT OUTER JOIN facts AS f ON p.id = f.periodid
WHERE p.companyid = 100 AND f.otherid = 17
```

which is not quite what I'm after.

Tags: [c#](#) ([Prev Q](#)) ([Next Q](#)), [linq-to-sql](#) ([Prev Q](#)) ([Next Q](#)), [linq](#) ([Next Q](#))

User: [dan](#) 

[Answer](#)  by [dahlbyk](#) 

You need to introduce your join condition before calling [DefaultIfEmpty\(\)](#). I would just use extension method syntax:

```
from p in context.Periods
join f in context.Facts on p.id equals f.periodid into fg
from fgi in fg.Where(f => f.otherid == 17).DefaultIfEmpty()
where p.companyid == 100
select f.value
```

Or you could use a subquery:

```
from p in context.Periods
join f in context.Facts on p.id equals f.periodid into fg
from fgi in (from f in fg
             where f.otherid == 17
```

```
        select f).DefaultIfEmpty()
where p.companyid == 100
select f.value
```

Tags: [c#](#) ([Prev Q](#)) ([Next Q](#)), [linq-to-sql](#) ([Prev Q](#)) ([Next Q](#)), [linq](#) ([Next Q](#))

Q: How to use DbContext.Database.SqlQuery<TElement>(sql, params) with stored procedure? EF Code First CTP5

Tags: [c#](#) ([Prev Q](#)) ([Next Q](#))

I have a stored procedure that has three parameters and I've been trying to use the following to return the results:

```
context.Database.SqlQuery<myEntityType>("mySpName", param1, param2, param3);
```

At first I tried using SqlParameter objects as the params but this didn't work and threw a SQLException with the following message:

Procedure or function 'mySpName' expects parameter '@param1', which was not supplied.

So my question is how you can use this method with a stored procedure that expects parameters?

Thanks.

Tags: [c#](#) ([Prev Q](#)) ([Next Q](#))

User: [electricssheep](#)

[Answer](#) by [devart](#)

You should supply the SqlParameter instances in the following way:

```
context.Database.SqlQuery<myEntityType>(
    "mySpName @param1, @param2, @param3",
    new SqlParameter("param1", param1),
    new SqlParameter("param2", param2),
    new SqlParameter("param3", param3)
);
```

[Answer](#) by [dan-mork](#)

Also, you can use the "sql" parameter as a format specifier:

```
context.Database.SqlQuery<MyEntityType>("mySpName @param1 = {0}", param1)
```

[Answer](#) by [tom-halladay](#)

This solution is (only) for SQL Server 2005

You guys are lifesavers, but as @Dan Mork said, you need to add EXEC to the mix. What was tripping me up was:

- ‘EXEC ‘ before the Proc Name
- Commas in between Params
- Chopping off ‘@’ on the Param Definitions (not sure that bit is required though).

:

```
context.Database.SqlQuery<EntityType>(
    "EXEC ProcName @param1, @param2",
    new SqlParameter("param1", param1),
    new SqlParameter("param2", param2)
);
```

Tags: [c#](#) ([Prev Q](#)) ([Next Q](#))

[Q: Fastest Way of Inserting in Entity Framework](#)

Tags: [c#](#) ([Prev Q](#))

I'm looking for the fastest way of inserting into Entity Framework, I'm asking this because of the scenario where you have an active TransactionScope and the insertion is huge (4000+). It can potentially last more than 10 minutes (default timeout of transactions), and this will lead to an incomplete transaction.

Regards.

Tags: [c#](#) ([Prev Q](#))

User: [bongo-sharp](#) 

[Answer](#)  by [slauma](#) 

To your remark in the comments to your question:

“...SavingChanges (**for each record**)...”

That's the worst thing you can do! Calling SaveChanges() for each record slows bulk inserts extremely down. I would do a few simple tests which will very likely improve the performance:

- Call SaveChanges() once after ALL records.
- Call SaveChanges() after for example 100 records.
- Call SaveChanges() after for example 100 records and dispose the context and create a new one.
- Disable change detection

For bulk inserts I am working and experimenting with a pattern like this:

[Skip code block](#)

```
using (TransactionScope scope = new TransactionScope())
{
    MyDbContext context = null;
    try
    {
        context = new MyDbContext();
```

```

        context.Configuration.AutoDetectChangesEnabled = false;

        int count = 0;
        foreach (var entityToInsert in someCollectionOfEntitiesToInsert)
        {
            ++count;
            context = AddToContext(context, entityToInsert, count, 100, true);
        }

        context.SaveChanges();
    }
    finally
    {
        if (context != null)
            context.Dispose();
    }
}

scope.Complete();
}

private MyDbContext AddToContext(MyDbContext context,
    Entity entity, int count, int commitCount, bool recreateContext)
{
    context.Set<Entity>().Add(entity);

    if (count % commitCount == 0)
    {
        context.SaveChanges();
        if (recreateContext)
        {
            context.Dispose();
            context = new MyDbContext();
            context.Configuration.AutoDetectChangesEnabled = false;
        }
    }
}

return context;
}

```

I have a test program which inserts 560.000 entities (9 scalar properties, no navigation properties) into the DB. With this code it works in less than 3 minutes.

For the performance it is important to call `SaveChanges()` after “many” records (“many” around 100 or 1000). It also improves the performance to dispose the context after `SaveChanges` and create a new one. This clears the context from all entites, `SaveChanges` doesn’t do that, the entities are still attached to the context in state `Unchanged`. It is the growing size of attached entities in the context what slows down the insertion step by step. So, it is helpful to clear it after some time.

Here are a few measurements for my 560.000 entities:

- `commitCount = 1, recreateContext = false`: **many hours** (That’s your current procedure)
- `commitCount = 100, recreateContext = false`: **more than 20 minutes**
- `commitCount = 1000, recreateContext = false`: **242 sec**
- `commitCount = 10000, recreateContext = false`: **202 sec**
- `commitCount = 100000, recreateContext = false`: **199 sec**
- `commitCount = 1000000, recreateContext = false`: **out of memory exception**
- `commitCount = 1, recreateContext = true`: **more than 10 minutes**
- `commitCount = 10, recreateContext = true`: **241 sec**
- `commitCount = 100, recreateContext = true`: **164 sec**
- `commitCount = 1000, recreateContext = true`: **191 sec**

The behaviour in the first test above is that the performance is very non-linear and decreases extremely over time. (“Many hours” is an estimation, I never finished this test, I stopped at 50.000 entities after 20 minutes.) This non-linear behaviour is not so significant in all other tests.

[Answer](#) by [arkhivania](#)

This combination increase speed well enough.

```
context.Configuration.AutoDetectChangesEnabled = false;  
context.Configuration.ValidateOnSaveEnabled = false;
```

[Answer](#) by [adam-rackis](#)

You should look at using the `System.Data.SqlClient.SqlBulkCopy` for this. Here's the [documentation](#), and of course there are plenty of tutorials online.

Sorry, I know you were looking for a simple answer to get EF to do what you want, but bulk operations are not really what ORMs are meant for.

Tags: [c#](#) ([Prev](#) [Q](#))

Java

Questions

[Q: How do I get the size of a java.sql.ResultSet?](#)

Tags: [java](#) ([Prev Q](#)) ([Next Q](#))

Shouldn't this be a pretty straightforward operation? However, I see there's neither a `size()` nor `length()` method.

Tags: [java](#) ([Prev Q](#)) ([Next Q](#))

User: [jake](#) 

[Answer](#)  by [finnw](#) 

`resultSet.last()` followed by `resultSet.getRow()` will give you the row count, but it may not be a good idea as it can mean reading the entire table over the network and throwing away the data. Do a `SELECT COUNT(*) FROM...` query instead.

[Answer](#)  by [jeebee](#) 

```
ResultSet rs = ps.executeQuery();
int rowcount = 0;
if (rs.last()) {
    rowcount = rs.getRow();
    rs.beforeFirst(); // not rs.first() because the rs.next() below will move on, missing the first ele
}
while (rs.next()) {
    // do your standard per row stuff
}
```

Tags: [java](#) ([Prev Q](#)) ([Next Q](#))

[Q: Print query string in Hibernate with parameter values](#)

Tags: [java](#) ([Prev Q](#)) ([Next Q](#)), [orm](#) ([Next Q](#))

Is it possible in Hibernate to print generated SQL queries with real values instead of question marks?

How would you suggest to print queries with real values if it is not possible with Hibernate API?

Tags: [java](#) ([Prev Q](#)) ([Next Q](#)), [orm](#) ([Next Q](#))

User: [craftsman](#) 

[Answer](#)  by [pascal-thivent](#) 

You need to enable [logging](#) for the the following categories at debug and trace levels respectively:

- org.hibernate.SQL - Log all SQL DML statements as they are executed
- org.hibernate.type - Log all JDBC parameters

So a log4j configuration could look like:

```
# logs the SQL statements
log4j.logger.org.hibernate.SQL=debug

# Logs the JDBC parameters passed to a query
log4j.logger.org.hibernate.type=trace
```

The first is equivalent to `hibernate.show_sql=true` legacy [property](#), the second **prints the bound parameters among other things**.

Another solution (non hibernate based) would be to use a JDBC proxy driver like [P6Spy](#).

Tags: [java](#) ([Prev Q](#)) ([Next Q](#)), [orm](#) ([Next Q](#))

[Q: Java - escape string to prevent SQL injection](#)

Tags: [java](#) ([Prev Q](#)) ([Next Q](#))

I'm trying to put some anti sql injection in place in java and am finding it very difficult to work with the the "replaceAll" string function. Ultimately I need a function that will convert any existing \ to \\, any " to \", any ' to \', and any \n to \\n so that when the string is evaluated by MySQL SQL injections will be blocked.

I've jacked up some code I was working with and all the \\\\\\\\\\\\\\\\\ in the function are making my eyes go nuts. If anyone happens to have an example of this I would greatly appreciate it.

Tags: [java](#) ([Prev Q](#)) ([Next Q](#))

User: [isisagate](#)

[Answer](#) by [kaleb-brasee](#)

PreparedStatements are the way to go, because they make SQL injection impossible. Here's a simple example taking the user's input as the parameters:

[Skip code block](#)

```
public insertUser(String name, String email) {
    Connection conn = null;
    PreparedStatement stmt = null;
    try {
        conn = setupTheDatabaseConnectionSomehow();
        stmt = conn.prepareStatement("INSERT INTO person (name, email) values (?, ?)");
        stmt.setString(1, name);
        stmt.setString(2, email);
        stmt.executeUpdate();
    }
    finally {
        try {
```

```

        if (stmt != null) { stmt.close(); }
    }
    catch (Exception e) {
        // log this error
    }
    try {
        if (conn != null) { conn.close(); }
    }
    catch (Exception e) {
        // log this error
    }
}
}

```

No matter what characters are in name and email, those characters will be placed directly in the database. They won't affect the INSERT statement in any way.

There are different set methods for different data types — which one you use depends on what your database fields are. For example, if you have an INTEGER column in the database, you should use a `setInt` method. [The PreparedStatement documentation](#) lists all the different methods available for setting and getting data.

Tags: [java](#) ([Prev Q](#)) ([Next Q](#))

[Q: Hibernate show real SQL](#)

Tags: [java](#) ([Prev Q](#)), [orm](#) ([Prev Q](#)) ([Next Q](#))

if I set

```
<property name="show_sql">true</property>
```

in my **hibernate.cfg.xml** configuration file in the console I can see the SQL.

But it's not *real* SQL... Can I see the SQL code that will be passed directly to database?

Example:

I see

```
select this_.code from true.employee this_ where this_.code=?
```

Can I see

```
select employee.code from employee where employee.code=12
```

the *real* SQL?

Tags: [java](#) ([Prev Q](#)), [orm](#) ([Prev Q](#)) ([Next Q](#))

User: [tommaso-taruffi](#)

[Answer](#) by [pascal-thivent](#)

Can I see (...) the real SQL

If you want to see the SQL sent directly to the database (that is formatted **similar** to your example), you'll have to use some kind of jdbc driver proxy like [P6Spy](#) (or

[log4jdbc](#)

Alternatively you can enable logging of the following categories (using a `log4j.properties` file here):

```
log4j.logger.org.hibernate.SQL=DEBUG  
log4j.logger.org.hibernate.type=TRACE
```

The first is equivalent to `hibernate.show_sql=true`, the second prints the bound parameters among other things.

Reference

- Hibernate 3.5 Core Documentation
 - [3.5. Logging](#)
- Hibernate 4.1 Core Documentation
 - [3.5. Logging](#)

[Answer](#) by [tommaso-taruffi](#)

log4j.properties

[Skip code block](#)

```
log4j.logger.org.hibernate=INFO, hb  
log4j.logger.org.hibernate.SQL=DEBUG  
log4j.logger.org.hibernate.type=TRACE  
log4j.logger.org.hibernate.hql.ast.AST=info  
log4j.logger.org.hibernate.tool.hbm2ddl=warn  
log4j.logger.org.hibernate.hql=debug  
log4j.logger.org.hibernate.cache=info  
log4j.logger.org.hibernate.jdbc=debug  
  
log4j.appender.hb=org.apache.log4j.ConsoleAppender  
log4j.appender.hb.layout=org.apache.log4j.PatternLayout  
log4j.appender.hb.layout.ConversionPattern=HibernateLog --> %d{HH:mm:ss} %-5p %c - %m%n  
log4j.appender.hb.Threshold=TRACE
```

hibernate.cfg.xml

```
<property name="show_sql">true</property>  
<property name="format_sql">true</property>  
<property name="use_sql_comments">true</property>
```

Tags: [java](#) ([Prev Q](#)), [orm](#) ([Prev Q](#)) ([Next Q](#))

GROUP BY

[Skip to questions](#),

Wiki by user [cloudymarble](#) 

About

The GROUP BY statement is used in conjunction with the aggregate functions to group the result-set by one or more columns.

Aggregate functions can be performed on other fields in the group, such as SUM() or AVG(), to collate related data into a single value.

Syntax

```
SELECT column_name, aggregate_function(column_name)
FROM table_name
WHERE column_name operator value
GROUP BY column_name
```

[Source](#) 

MySQL Handling of GROUP BY

In standard SQL, a query that includes a GROUP BY clause cannot refer to nonaggregated columns in the select list that are not named in the GROUP BY clause. For example, this query is illegal in standard SQL because the name column in the select list does not appear in the GROUP BY:

```
SELECT o.custid, c.name, MAX(o.payment)
  FROM orders AS o, customers AS c
 WHERE o.custid = c.custid
 GROUP BY o.custid;
```

For the query to be legal, the name column must be omitted from the select list or named in the GROUP BY clause.

[Source](#) 

[GROUP BY \(Aggregate\) Functions](#) 

Related Tags :

- [aggregate](#) 
 - [sql](#)
 - [sql-query](#) 
 - [mysql](#)
-

Questions

[Q: Is there any difference between GROUP BY and DISTINCT](#)

Tags: [group-by](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#))

I learned something simple about SQL the other day:

```
SELECT c FROM myTb1 GROUP BY C
```

Has the same result as:

```
SELECT DISTINCT C FROM myTb1
```

What I am curious of, is there anything different in the way an SQL engine processes the command, or are they truly the same thing?

I personally prefer the distinct syntax, but I am sure it's more out of habit than anything else.

EDIT: This is not a question about aggregates. The use of GROUP BY with aggregate functions is understood.

Tags: [group-by](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#))

User: [brettski](#)

[Answer](#) by [skeolan](#)

[MusiGenesis](#)' response is functionally the correct one with regard to your question as stated; the SQL Server is smart enough to realize that if you are using "Group By" and not using any aggregate functions, then what you actually mean is "Distinct" - and therefore it generates an execution plan as if you'd simply used "Distinct."

However, I think it's important to note [Hank](#)'s response as well - cavalier treatment of "Group By" and "Distinct" could lead to some pernicious gotcha's down the line if you're not careful. It's not entirely correct to say that this is "not a question about aggregates" because you're asking about the functional difference between two SQL query keywords, one of which is **meant to be used with aggregates** and one of which is not.

A hammer can work to drive in a screw sometimes , but if you've got a screwdriver handy, why bother?

[Answer](#) by [glomek](#)

GROUP BY lets you use aggregate functions, like AVG, MAX, MIN, SUM, and COUNT. Other hand DISTINCT just removes duplicates.

For example, if you have a bunch of purchase records, and you want to know how much was spent by each department, you might do something like:

```
SELECT department, SUM(amount) FROM purchases GROUP BY department
```

This will give you one row per department, containing the department name and the sum of all of the amount values in all rows for that department.

Tags: [group-by](#) ([Prev Q](#)) ([Next Q](#)), [distinct](#) ([Prev Q](#)) ([Next Q](#))

Q: Using group by on multiple columns

Tags: [group-by](#) ([Prev Q](#)) ([Next Q](#))

I understand the point of group by x

But how does group by x, y work and what does it mean?

Tags: [group-by](#) ([Prev Q](#)) ([Next Q](#))

User: [l_____](#)

[Answer](#) by [smashery](#)

Group By X means **put all those with the same value for X in the one group.**

Group By X, Y means **put all those with the same values for both X and Y in the one group.**

To illustrate using an example, let's say we have the following table, to do with who is attending what subject at a university:

[Skip code block](#)

Table: Subject_Selection		
Subject	Semester	Attendee
ITB001	1	John
ITB001	1	Bob
ITB001	1	Mickey
ITB001	2	Jenny
ITB001	2	James
MKB114	1	John
MKB114	1	Erica

When you use a group by on the subject column only; say:

```
select Subject, Count(*)
from Subject_Selection
group by Subject
```

You will get something like:

Subject	Count
ITB001	5
MKB114	2

...because there are 5 entries for ITB001, and 2 for MKB114

If we were to group by two columns:

```
select Subject, Semester, Count(*)
from Subject_Selection
group by Subject, Semester
```

we would get this:

Subject	Semester	Count
ITB001	1	5

ITB001	1	3
ITB001	2	2
MKB114	1	2

This is because, when we group by two columns, it is saying “**Group them so that all of those with the same Subject and Semester are in the same group, and then calculate all the aggregate functions** (Count, Sum, Average, etc.) **for each of those groups**”. In this example, this is demonstrated by the fact that, when we count them, there are **three** people doing ITB001 in semester 1, and **two** doing it in semester 2. Both of the people doing MKB114 are in semester 1, so there is no row for semester 2 (no data fits into the group “MKB114, Semester 2”)

Hopefully that makes sense.

Tags: [group-by](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to use count and group by at the same select statement](#)

Tags: [group-by](#) ([Prev Q](#))

I have an sql select query that has a group by. I want to count all the records after the group by statement. Is there a way for this directly from sql? For example, having a table with users I want to select the different towns and the **total** number of users

```
select town, count(*) from user
group by town
```

I want to have a column with all the towns and another with the number of users in all rows.

An example of the result for having 3 towns and 58 users in total is :

Town	Count
Copenhagen	58
NewYork	58
Athens	58

Tags: [group-by](#) ([Prev Q](#))

User: [stavros](#) 

[Answer](#)  by [oded](#) 

This will do what you want (list of towns, with the number of users in each):

```
select town, count(town)
from user
group by town
```

You can use most [aggregate functions](#)  when using GROUP BY.

Update (following change to question and comments)

You can declare a variable for the number of users and set it to the number of users then select with that.

```
DECLARE @numOfUsers INT
SET @numOfUsers = SELECT COUNT(*) FROM user
```

```
SELECT DISTINCT town, @numOfUsers  
FROM user
```

[Answer](#) by [milkovsky](#)

You can use [COUNT\(DISTINCT...\)](#):

```
SELECT COUNT(DISTINCT town)  
FROM user
```

Tags: [group-by](#) ([Prev Q](#))

SQLite

[Skip to questions,](#)

Wiki by user [doug-currie](#) 

[SQLite](#)  is a software library that implements a [self-contained](#) , [serverless](#) , [zero-configuration](#) , [transactional](#)  SQL database engine. SQLite is the most widely deployed SQL database engine in the world. The source code for SQLite is in the public domain.

SQLite is a relational database management system contained in a small (~350 KB) C programming library. In contrast to other database management systems, SQLite is not a separate process that is accessed from the client application, but an integral part of it.

SQLite is ACID-compliant and implements most of the SQL standard, using a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity.

Mobile Apps

SQLite is commonly used to store data on Android, iOS and Windows Phone apps, since it has a simple implementation, easy to adapt and quite fast.

Design

Unlike client–server database management systems, the SQLite engine has no standalone processes with which the application program communicates. Instead, the SQLite library is linked in and thus becomes an integral part of the application program.

The application program uses SQLite's functionality through simple function calls, which reduce latency in database access: function calls within a single process are more efficient than inter-process communication. SQLite stores the entire database as a single cross-platform file on a host machine.

References

- [SQLite Homepage](#) 
 - [SQLite Wikipedia Article](#) 
 - [Official SQLite FAQ](#) 
 - [SQL Lite Documentation](#) 
 - [SQLite In 5 Minutes Or Less](#) 
-

Questions

[Q: sqlite database default time value ‘now’](#)

Tags: [sqlite](#) ([Prev Q](#)) ([Next Q](#))

Is it possible in a sqlite database to craete a table that has a timestamp column that default to DATETIME('now') ?

Like this:

```
CREATE TABLE test (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    t TIMESTAMP DEFAULT DATETIME('now')
);
```

This gives an error... How to resolve?

Tags: [sqlite](#) ([Prev Q](#)) ([Next Q](#))

User: [joepie](#)

[Answer](#) by [owen](#)

i believe you can use

```
CREATE TABLE test (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    t TIMESTAMP
    DEFAULT CURRENT_TIMESTAMP
);
```

as of version 3.1 ([source](#))

[Answer](#) by [rev](#)

according to dr. hipp in a recent list post:

```
CREATE TABLE whatever(
    ...
    timestamp DATE DEFAULT (datetime('now','localtime')),
    ...
);
```

Tags: [sqlite](#) ([Prev Q](#)) ([Next Q](#))

[Q: SQLite - UPSERT *not* INSERT or REPLACE](#)

Tags: [sqlite](#) ([Prev Q](#)) ([Next Q](#))

<http://en.wikipedia.org/wiki/Upsert>

[Insert Update stored proc on SQL Server](#)

Is there some clever way to do this in SQLite that I have not thought of?

Basically I want to update three out of four columns if the record exists, If it does not

exists I want to INSERT the record with the default (NUL) value for the fourth column.

The ID is a primary key so there will only ever be one record to UPSERT.

(I am trying to avoid the overhead of SELECT in order to determine if I need to UPDATE or INSERT obviously)

Suggestions?

Tags: [sqlite](#) ([Prev Q](#)) ([Next Q](#))

User: [mike-trader](#) 

[Answer](#)  by [eric-b](#) 

Assuming 3 columns in the table.. ID, NAME, ROLE

BAD: This will insert or replace all columns with new values for ID=1:

```
INSERT OR REPLACE INTO Employee (id, name, role)
VALUES (1, 'John Foo', 'CEO');
```

BAD: This will insert or replace 2 of the columns... the NAME column will be set to NULL or the default value:

```
INSERT OR REPLACE INTO Employee (id, role)
VALUES (1, 'code monkey');
```

GOOD: This will update 2 of the columns. When ID=1 exists, the NAME will be unaffected. When ID=1 does not exist, the name will be default (NULL).

```
INSERT OR REPLACE INTO Employee (id, role, name)
VALUES ( 1,
        'code monkey',
        (SELECT name FROM Employee WHERE id = 1)
      );
```

This will update 2 of the columns. When ID=1 exists, the ROLE will be unaffected. When ID=1 does not exist, the role will be set to 'Benchwarmer' instead of the default value.

```
INSERT OR REPLACE INTO Employee (id, name, role)
VALUES ( 1,
        'Susan Bar',
        COALESCE((SELECT role FROM Employee WHERE id = 1), 'Benchwarmer')
      );
```

[Answer](#)  by [gregschlom](#) 

INSERT OR REPLACE is NOT equivalent to "UPSERT".

Say I have the table Employee with the fields id, name, and role:

```
INSERT OR REPLACE INTO Employee ("id", "name", "role") VALUES (1, "John Foo", "CEO")
INSERT OR REPLACE INTO Employee ("id", "role") VALUES (1, "code monkey")
```

Boom, you've lost the name of the employee number 1. SQLite has replaced it with a default value.

The expected output of an UPSERT would be to change the role and to keep the name.

[Answer](#) by [aristotle-pagaltzis](#)

[Eric B's answer](#) is OK if you want to preserve just one or maybe two columns from the existing row. If you want to preserve a lot of rows, it gets too cumbersome fast.

Here's an approach that will scale well to any amount of columns on either side. To illustrate it I will assume the following schema:

```
CREATE TABLE page (
    id      INTEGER PRIMARY KEY,
    name    TEXT UNIQUE,
    title   TEXT,
    content TEXT,
    author  INTEGER NOT NULL REFERENCES user (id),
    ts      TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Note in particular that `name` is the natural key of the row – `id` is used only for foreign keys, so the point is for SQLite to pick the ID value itself when inserting a new row. But when updating an existing row based on its `name`, I want it to continue to have the old ID value (obviously!).

I achieve a true UPSERT with the following construct:

[Skip code block](#)

```
INSERT OR REPLACE INTO page (id, name, title, content, author)
SELECT old.id, new.name, new.title, old.content, new.author
FROM ( SELECT
        "about"           AS name,
        "About this site" AS title,
        42                AS author
    ) AS new
LEFT JOIN (
    SELECT id, name, content
    FROM page
) AS old ON new.name = old.name;
```

Here, if a row did not previously exist `old.id` will be `NULL` and SQLite will then assign an ID automatically, but if there already was such a row, `old.id` will have an actual value and this will be reused. Which is exactly what I wanted.

In fact this is very flexible. Note how the `ts` column is completely missing on all sides – because it has a `DEFAULT` value, SQLite will just do the right thing in any case, so I don't have to take care of it myself.

You can also include a column on both the `new` and `old` sides and then use e.g. `COALESCE(new.content, old.content)` in the outer `SELECT` to say “insert the new content if there was any, otherwise keep the old content” – e.g. if you are using a fixed query and are binding the new values with placeholders.

Tags: [sqlite](#) ([Prev Q](#)) ([Next Q](#))

[Q: Is it possible to insert multiple rows at a time in an SQLite database?](#)

Tags: [sqlite](#) ([Prev Q](#))

In MySQL you can insert multiple rows like this:

```
INSERT INTO 'tablename' ('column1', 'column2') VALUES
  ('data1', 'data2'),
  ('data1', 'data2'),
  ('data1', 'data2'),
  ('data1', 'data2');
```

However, I am getting an error when I try to do something like this. Is it possible to insert multiple rows at a time in an SQLite database? What is the syntax to do that?

Tags: [sqlite](#) ([Prev Q](#))

User: [andrew](#) 

[Answer](#)  by [fearless_fool](#) 

update

As [BrianCampbell points out here](#) , **SQLite 3.7.11 and above now supports the simpler syntax of the original post.** However, the approach shown is still appropriate if you want maximum compatibility across legacy databases.

original answer

If I had privileges, I would bump [andy's reply](#) : You **can** insert multiple rows in SQLite, you just need **different syntax**. To make it perfectly clear, the OPs MySQL example:

```
INSERT INTO 'tablename' ('column1', 'column2') VALUES
  ('data1', 'data2'),
  ('data1', 'data2'),
  ('data1', 'data2'),
  ('data1', 'data2');
```

This can be recast into SQLite as:

```
INSERT INTO 'tablename'
  SELECT 'data1' AS 'column1', 'data2' AS 'column2'
UNION ALL SELECT 'data1', 'data2'
UNION ALL SELECT 'data1', 'data2'
UNION ALL SELECT 'data1', 'data2'
```

I've been using this to "broadside" large datasets from RoR and it's a big performance win.

a note on UNION vs UNION ALL

As several people commented, if you use `UNION ALL` (as shown above), all rows will be inserted, so in this case, you'd get four rows of `data1, data2`. If you omit the `ALL`, then duplicate rows will be eliminated (and the operation will presumably be a bit slower). We're using `UNION ALL` since it more closely matches the semantics of the original post.

in closing

P.S.: Please +1 [andy's reply](#) , not mine! He presented the solution first.

[Answer](#) by [andy](#)

Yes it is possible, but not with the usual comma-separated insert values.

Try this...

```
insert into myTable (col1,col2)
    select aValue as col1,anotherValue as col2
    union select moreValue,evenMoreValue
    union...
```

Yes, it's a little ugly but easy enough to automate the generation of the statement from a set of values. Also, it appears you only need to declare the column names in the first select.

Tags: [sqlite](#) ([Prev Q](#))

DISTINCT

[Skip to questions](#),

Wiki by user [sloth](#) 

The SQL DISTINCT keyword is used to remove duplicate values from a result of a query.

The [SQL 92 Standard](#)  defines DISTINCT as:

Two values are said to be not distinct if either: both are the null value, or they compare equal... Otherwise they are distinct. Two rows (or partial rows) are distinct if at least one of their pairs of respective values is distinct. Otherwise they are not distinct.

Questions

[Q: SQL to find the number of distinct values in a column](#)

Tags: [distinct](#) ([Prev Q](#))

I can select all the distinct values in a column in the following ways:

- `SELECT DISTINCT column_name FROM table_name;`
- `SELECT column_name FROM table_name GROUP BY column_name;`

But how do I get the row count from that query? Is a subquery required?

Tags: [distinct](#) ([Prev Q](#))

User: [christian-oudard](#) 

[Answer](#)  by [noah-goodrich](#) 

You can use the `DISTINCT` keyword within the `COUNT`  aggregate function:

```
SELECT COUNT(DISTINCT column_name) AS some_alias FROM table_name
```

This will count only the distinct values for that column.

[Answer](#)  by [paul-james](#) 

This will give you BOTH the distinct column values and the count of each value. I usually find that I want to know both pieces of information.

```
select distinct columnName, count(columnName) as CountOf from tableName group by columnName
```

Tags: [distinct](#) ([Prev Q](#))

Django

[Skip to questions](#),

Wiki by user [daniel-roseman](#) 

Django is the Web framework for perfectionists with deadlines.

Django is an open source Web 2.0 application framework, written in [Python](#)  and maintained by the non-profit [Django Software Foundation \(DSF\)](#) . Its primary goal is to ease the creation of complex data-driven websites.

Django follows the [mvc](#)  ([model-view-controller](#) ) architectural pattern. This consists of:

- an [object-relational mapper](#)  that mediates between data models (Python classes) and a relational database (“Model”)
- a system for processing requests with a web templating system (“View”)
- a regular-expression-based URL dispatcher (“Controller”)

Django’s latest **version is 1.9.1** and was released on the 2nd of January 2016.

Resources

- [Wikipedia Article](#) 
- [Django Project Homepage](#) 
- [Documentation](#) 
- [Download](#) 
- [GitHub Repository](#) 
- [Community](#) 
- [Django Packages](#) 

Useful books

- [2 Scoops of Django](#) 
- [Pro Django](#) 
- [Test Driven Development with Python](#) 
- [Lightweight Django](#) 
- [High Performance Django](#) 

Out of date resources

The following resources are out of date, and are not recommended for learning about Django.

- [The Django Book](#) 

Tutorials

- [The official Django tutorial](#) 
- [How to Tango with Django](#) 
- [DjangoGirls Tutorial](#) 

Newsletters

- [Django Round-Up](#) 

Podcasts

- [GoDjango \(weekly screencasts\)](#) 

Websites Using Django

- [Disqus](#) 
- [Pinterest](#) 
- [Instagram](#) 
- [Mahalo](#) 
- [Mozilla Support](#) 
- [PythonAnywhere](#) 
- [JSFiddle](#) 

See also:

- [django-models](#) 
 - [django-orm](#) 
 - [django-admin](#) 
 - [django-forms](#) 
 - [django-formwizard](#) 
 - [django-templates](#) 
 - [django-urls](#) 
 - [django-views](#) 
 - [django-settings](#) 
 - [mvc](#) 
 - [django-queryset](#) 
 - [django-rest-framework](#) 
 - [inline-formset](#) 
-

Questions

[Q: Show the sql Django is running?](#)

Tags: [django](#) ([Prev Q](#)) ([Next Q](#))

Is there a way to show the SQL that Django is running while performing a query?

Tags: [django](#) ([Prev Q](#)) ([Next Q](#))

User: [spence91](#) 

[Answer](#)  by [geowa4](#) 

Try the docs FAQ: [How can I see the raw SQL queries Django is running?](#) 

`django.db.connection.queries` contains a list of the SQL queries.

```
from django.db import connection
print connection.queries
```

Querysets also have a [query attribute](#)  containing the query to be executed.

```
print MyModel.objects.filter(name="my name").query
```

Tags: [django](#) ([Prev Q](#)) ([Next Q](#))

[Q: Getting the SQL from a Django QuerySet](#)

Tags: [django](#) ([Prev Q](#)) ([Next Q](#))

How do I get the SQL that Django will use on the database from a QuerySet object? I'm trying to debug some strange behavior, but I'm not sure what queries are going to the database. Thanks for your help.

Tags: [django](#) ([Prev Q](#)) ([Next Q](#))

User: [exupero](#) 

[Answer](#)  by [jpwatts](#) 

You print the queryset's query attribute.

```
>>> queryset = MyModel.objects.all()
>>> print queryset.query
SELECT "myapp_mymodel"."id", ... FROM "myapp_mymodel"
```

Tags: [django](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to perform OR condition in django queryset?](#)

Tags: [django](#) ([Prev Q](#)) ([Next Q](#))

I want to write a Django query equivalent to this SQL query:

```
SELECT * from user where income >= 5000 or income is NULL.
```

How to construct the Djagno queryset filter?

```
User.objects.filter(income__gte=5000, income=0)
```

This doesn't work, because it ANDs the filters. I want to OR the filters to get union of individual querysets.

Tags: [django](#) ([Prev Q](#)) ([Next Q](#))

User: [elisa](#) 

[Answer](#)  by [lakshman-prasad](#) 

```
from django.db.models import Q
User.objects.filter(Q(income__gte=5000) | Q(income__isnull=True))
```

[via Documentation](#) 

Tags: [django](#) ([Prev Q](#)) ([Next Q](#))

[Q: Django CharField vs TextField](#)

Tags: [django](#) ([Prev Q](#))

What is the difference between `CharField()` and `TextField()` in Django? The [documentation](#)  says that `CharField()` should be used for smaller strings and `TextField()` should be used for larger strings. Okay, but where is the line drawn between "small" and "large"? What's going on under the hood here that makes this the case?

Tags: [django](#) ([Prev Q](#))

User: [jonathan-gleason](#) 

[Answer](#)  by [cat-plus-plus](#) 

It's a difference between RDBMS's `varchar` (or similar) — those are usually specified with a maximum length, and might be more efficient in terms of performance or storage — and `text` (or similar) types — those are usually limited only by hardcoded implementation limits (not a DB schema).

PostgreSQL 9, specifically, states that "[There is no performance difference among these three types](#)" , but AFAIK there are some differences in e.g. MySQL, so this is something to keep in mind.

A good rule of thumb is that you use `CharField` when you need to limit the maximum length, `TextField` otherwise.

This is not really Django-specific, also.

Tags: [django](#) ([Prev Q](#))

SSMS

[Skip to questions](#),

Wiki by user [curt](#) 

SQL Server Management Studio is a tool included with Microsoft SQL Server 2005 and later versions for accessing, configuring, managing, administering and developing all components within Microsoft SQL Server. The tool includes both script editors and graphical tools which work with objects and features of the server.

SSMS combines the features of Enterprise Manager, Query Analyzer, and Analysis Manager, included in previous releases of SQL Server, into a single environment. In addition, SSMS works with all components of SQL Server such as Reporting Services and Integration Services. Developers get a familiar experience, and database administrators get a single comprehensive utility that combines easy-to-use graphical tools with rich scripting capabilities.

On Wikipedia: [SQL Server Management Studio](#) 

Questions

[Q: SQL Formatter for SQL Management Studio](#)

Tags: [ssms](#) ([Prev Q](#)) ([Next Q](#))

I was wondering if there is a plugin/tool for SQL Server Management Studio that will format your SQL?

I'm working with some large-ish stored procs that are a mangled mess of poorly formatted SQL and it'd be nice if I could just go "Select All -> Format SQL"

Tags: [ssms](#) ([Prev Q](#)) ([Next Q](#))

User: [lomaxx](#) 

[Answer](#)  by [maumen](#) 

Today I discovered Apex SQL Refactor. It is a free plugin. Integrates with SSMS. Downside is that it is an all or nothing process. It does not refactor as you type.

[Answer](#)  by [tao](#) 

Late answer, but hopefully worthwhile: The [Poor Man's T-SQL Formatter](#)  is an open-source (free) T-SQL formatter with complete T-SQL batch/script support (any DDL, any DML), SSMS Plugin, command-line bulk formatter, and other options.

It's available for immediate/online use at <http://poorsql.com> , and just today graduated to "version 1.0" (it was in beta for a few months), having just acquired support for MERGE statements, OUTPUT clauses, and other finicky stuff.

The SSMS Add-in allows you to set your own hotkey (default is `ctrl-K, ctrl-F`, to match Visual Studio), and formats the entire script or just the code you have selected/highlighted, if any. Output formatting is customizable.

In SSMS 2008 it combines nicely with the built-in intellisense, effectively providing more-or-less the same base functionality as Red Gate's SQL Prompt (SQL Prompt does of course have extra stuff, like snippets, quick object scripting, etc).

Feedback / feature requests are more than welcome, please give it a whirl if you get the chance!

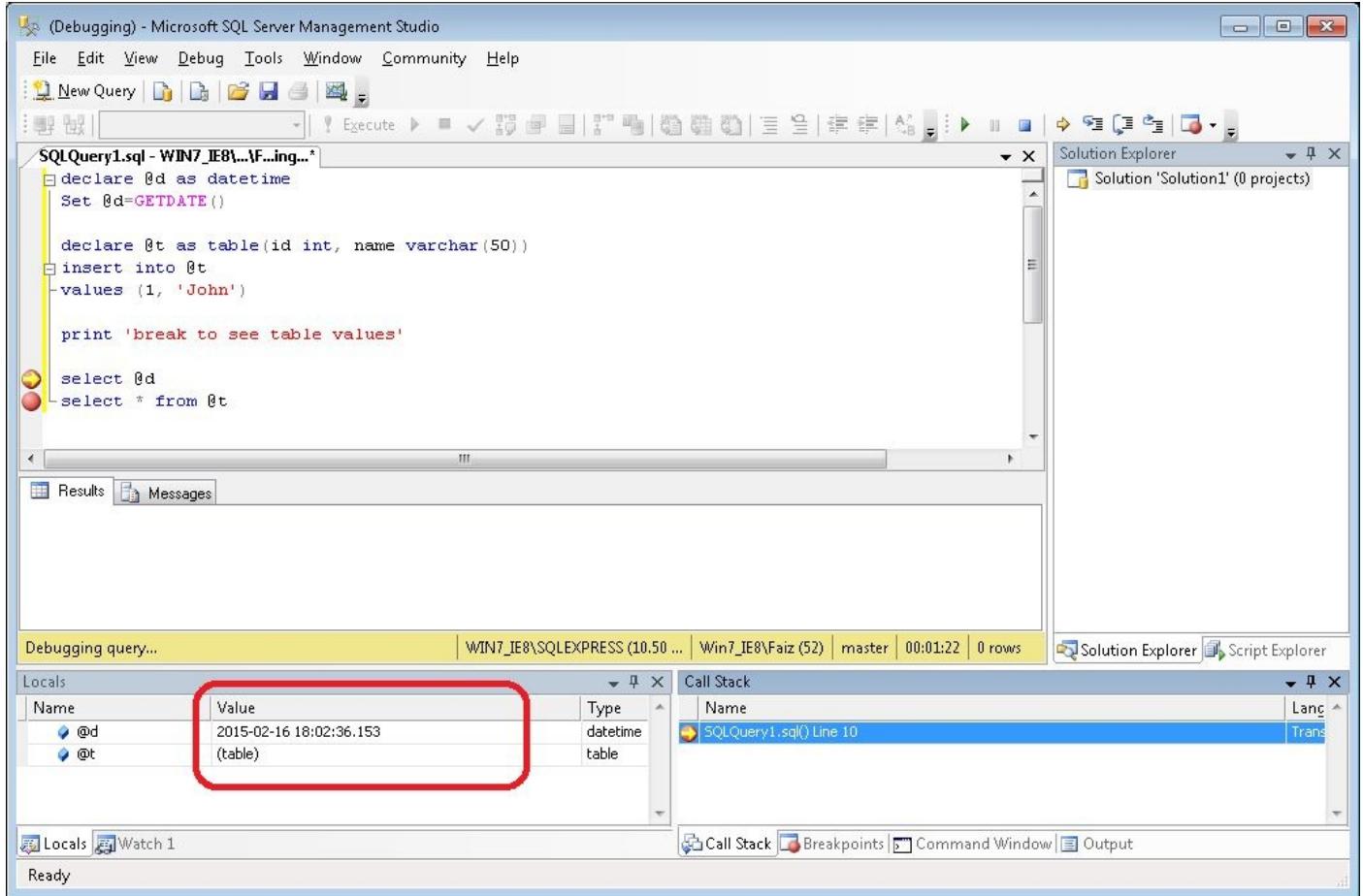
Disclosure: This is probably obvious already but I wrote this library/tool/site, so this answer is also shameless self-promotion :)

Tags: [ssms](#) ([Prev Q](#)) ([Next Q](#))

[Q: How to see the values of a table variable at debug time in T-SQL?](#)

Tags: [ssms](#) ([Prev Q](#))

Can we see the values (rows and cells) in a table valued variable in SQL Server Management Studio (SSMS) during debug time? If yes, how?



Tags: [ssms](#) ([Prev Q](#))

User: [faiz](#)

Answer by [kirby](#)

```
DECLARE @v XML = (SELECT * FROM <tablename> FOR XML AUTO)
```

Insert the above statement at the point where you want to view the table's contents. The table's contents will be rendered as XML in the locals window, or you can add @v to the watches window.

(Debugging) - Microsoft SQL Server Management Studio

File Edit View Debug Tools Window Community Help

New Query Execute

SQLQuery1.sql - WIN7_IE8\...ing...*

```
declare @d as datetime
Set @d=GETDATE()

declare @t as table(id int, name varchar(50))
insert into @t
values (1, 'John')

print 'break to see table values'

DECLARE @v XML = (SELECT * FROM @t FOR XML AUTO)
select @d
select * from @t
```

Results Messages

XML Visualizer

Expression: @v

Value:

```
<_x0040_t id="1" name="John" />
```

Close Help

Debugging query... | WIN7_IE8\SQLEXPRESS (10.50 ... | Win7_IE8\Faiz (52) | master | 00:06:36 | 0 rows

Solution Explorer Script Explorer

Locals

Name	Value	Type
@d	2015-02-16 18:06:20.060	datetime
@t	(table)	table
@v	<_x0040_t id="1" name="John" />	xml

Call Stack

Name	Lang
SQLQuery1.sql() Line 11	Trans

Locals Watch 1 Call Stack Breakpoints Command Window Output

Ready

Tags: [ssms](#) ([Prev Q](#))

LINQ To SQL

[Skip to questions](#),

Wiki by user [robert-harvey](#) 

LINQ to SQL is a component of .NET Framework version 3.5 that provides a run-time infrastructure for managing relational data as objects.

LINQ to SQL maps the data model of a relational database to an object model expressed in the programming language of the developer. When the application runs, LINQ to SQL translates the language-integrated queries in the object model into SQL and sends them to the database for execution. When the database returns the results, LINQ to SQL translates them back to objects that you can work with in your own programming language.

As of 2008, LINQ to SQL [is no longer being actively developed](#) , and Microsoft recommends using Entity Framework instead.

Introduction

- [LINQ to SQL](#)  (MSDN)
- [LINQ to SQL: .NET Language-Integrated Query for Relational Data](#)  (MSDN)

Using LINQ to SQL

- [Using LINQ to SQL \(Part 1\)](#)  by [Scott Guthrie](#) 

More information

- [LINQ to SQL \(formerly called DLINQ\) Wikipedia Article](#) 
 - [LINQ Wikipedia Article](#) 
-

Questions

[Q: Case insensitive string compare in LINQ-to-SQL](#)

Tags: [linq-to-sql](#) ([Prev Q](#)) ([Next Q](#)), [linq](#) ([Prev Q](#)) ([Next Q](#))

I've read that it's unwise to use ToUpper and ToLower to perform case-insensitive string comparisons, but I see no alternative when it comes to LINQ-to-SQL. The ignoreCase and CompareOptions arguments of String.Compare are ignored by LINQ-to-SQL (if you're using a case-sensitive database, you get a case-sensitive comparison even if you ask for a case-insensitive comparison). Is ToLower or ToUpper the best option here? Is one better than the other? I thought I read somewhere that ToUpper was better, but I don't know if that applies here. (I'm doing a lot of code reviews and everyone is using ToLower.)

```
Dim s = From row In context.Table Where String.Compare(row.Name, "test", StringComparison.InvariantCultureC
```

This translates to an SQL query that simply compares row.Name with "test" and will not return "Test" and "TEST" on a case-sensitive database.

Tags: [linq-to-sql](#) ([Prev Q](#)) ([Next Q](#)), [linq](#) ([Prev Q](#)) ([Next Q](#))

User: [bluemonkmn](#) 

[Answer](#)  by [andrew-arnott](#) 

As you say, there are some important differences between ToUpper and ToLower, and only one is dependably accurate when you're trying to do case insensitive equality checks.

Ideally, the best way to do a case-insensitive equality check is:

```
String.Equals(row.Name, "test", StringComparison.OrdinalIgnoreCase)
```

Note the **OrdinalIgnoreCase** to make it security-safe. But exactly the type of case (in)sensitive check you use depends on what your purposes is. But in general use Equals for equality checks and Compare when you're sorting, and then pick the right StringComparison for the job.

Michael Kaplan (a recognized authority on culture and character handling such as this) has relevant posts on ToUpper vs. ToLower:

- <http://www.siao2.com/2007/10/01/5218976.aspx> 
- <http://www.siao2.com/2005/03/10/391564.aspx> 

He says "String.ToUpper – **Use ToUpper rather than ToLower, and specify InvariantCulture in order to pick up OS casing rules**"

[Answer](#)  by [andrew-davey](#) 

I used `System.Data.Linq.SqlClient.SqlMethods.Like(row.Name, "test")` in my query.

This performs a case-insensitive comparison.

Tags: [linq-to-sql](#) ([Prev Q](#)) ([Next Q](#)), [linq](#) ([Prev Q](#)) ([Next Q](#))

Q: Entity Framework VS LINQ to SQL VS ADO.NET with stored procedures?

Tags: [linq-to-sql](#) ([Prev Q](#))

How would you rate each of them in terms of:

1. Performance
2. Speed of development
3. Neat, intuitive, maintainable code
4. Flexibility
5. Overall

I like my SQL and so have always been a die-hard fan of ADO.NET and stored procedures but I recently had a play with Linq to SQL and was blown away by how quickly I was writing out my DataAccess layer and have decided to spend some time really understanding either Linq to SQL or EF... or neither?

I just want to check, that there isn't a great flaw in any of these technologies that would render my research time useless. E.g. performance is terrible, it's cool for simple apps but can only take you so far.

Update: Can you concentrate on EF VS L2S VS SPs rather than ORM VS SPs. I'm mainly interested by EF VS L2S. But am keen to have them compared against stored procs too since plain SQL is something I know a lot about.

Tags: [linq-to-sql](#) ([Prev Q](#))

User: [britishdeveloper](#)

Answer by [dave-markle](#)

First off, if you're starting a new project, go with Entity Framework ("EF") - it now generates much better SQL (more like Linq to SQL does) and is easier to maintain and more powerful than Linq to SQL ("L2S"). As of the release of .NET 4.0, I consider Linq to SQL to be an obsolete technology. MS has been very open about not continuing L2S development further.

1) Performance

This is tricky to answer. For most single-entity operations (**CRUD**) you will find just about equivalent performance with all three technologies. You do have to know how EF and Linq to SQL work in order to use them to their fullest. For high-volume operations like polling queries, you may want to have EF/L2S "compile" your entity query such that the framework doesn't have to constantly regenerate the SQL, or you can run into scalability issues. (see edits)

For bulk updates where you're updating massive amounts of data, raw SQL or a stored

procedure will always perform better than an ORM solution because you don't have to marshal the data over the wire to the ORM to perform updates.

2) Speed of Development

In most scenarios, EF will blow away naked SQL/stored procs when it comes to speed of development. The EF designer can update your model from your database as it changes (upon request), so you don't run into synchronization issues between your object code and your database code. The only time I would not consider using an ORM is when you're doing a reporting/dashboard type application where you aren't doing any updating, or when you're creating an application just to do raw data maintenance operations on a database.

3) Neat/Maintainable code

Hands down, EF beats SQL/sprocs. Because your relationships are modeled, joins in your code are relatively infrequent. The relationships of the entities are almost self-evident to the reader for most queries. Nothing is worse than having to go from tier to tier debugging or through multiple SQL/middle tier in order to understand what's actually happening to your data. EF brings your data model into your code in a very powerful way.

4) Flexibility

Stored procs and raw SQL are more "flexible". You can leverage sprocs and SQL to generate faster queries for the odd specific case, and you can leverage native DB functionality easier than you can with an ORM.

5) Overall

Don't get caught up in the [false dichotomy](#) of choosing an ORM vs using stored procedures. You can use both in the same application, and you probably should. Big bulk operations should go in stored procedures or SQL (which can actually be called by the EF), and EF should be used for your CRUD operations and most of your middle-tier's needs. Perhaps you'd choose to use SQL for writing your reports. I guess the moral of the story is the same as it's always been. Use the right tool for the job. But the skinny of it is, EF is very good nowadays (as of .NET 4.0). Spend some real time reading and understanding it in depth and you can create some amazing, high-performance apps with ease.

EDIT: EF 5 simplifies this part a bit with [auto-compiled LINQ Queries](#), but for real high volume stuff, you'll definitely need to test and analyze what fits best for you in the real world.

[Answer](#) by user151323

Stored procedures:

(++)

- Great flexibility
- Full control over SQL
- The highest performance available

(—)

- Requires knowledge of SQL
- Stored procedures are out of source control
- Substantial amount of “repeating yourself” while specifying the same table and field names. The high chance of breaking the application after renaming a DB entity and missing some references to it somewhere.
- Slow development

ORM:

(+)

- Rapid development
- Data access code now under source control
- You’re isolated from changes in DB. If that happens you only need to update your model/mappings in one place.

(-)

- Performance may be worse
- No or little control over SQL the ORM produces (could be inefficient or worse buggy). Might need to intervene and replace it with custom stored procedures. That will render your code messy (some LINQ in code, some SQL in code and/or in the DB out of source control).
- As any abstraction can produce “high-level” developers having no idea how it works under the hood

The general tradeoff is between having a great flexibility and losing lots of time vs. being restricted in what you can do but having it done very quickly.

There is no general answer to this question. It’s a matter of holy wars. Also depends on a project at hand and your needs. Pick up what works best for you.

Tags: [linq-to-sql](#) ([Prev Q](#))

Random

[Skip to questions](#),

Wiki by user [thorsten-s.](#) 

This tag is for questions pertaining to creating random numbers or random distributions: using random number generators (RNGs), creating specific distributions (gaussian, log-normal) with given parameters from normal random numbers, problems and caveats with given implementations in different languages.

Since different languages may use different RNGs, a question with this tag must also be tagged with the language being used.

Frequently-asked questions

- How can I generate N uniformly-distributed random real numbers **that sum to a given value?**
 - [Language-agnostic](#)
 - [Python](#)
 - [Matlab](#)
 - [PHP](#)
 - [Javascript](#)
 - If uniformity is important, beware of [this popular algorithm](#)  (its output is **not** uniform.)
- How can I choose N elements, **avoiding duplicates**, from a list (or from the integer range 1..M)?
 - [Language-agnostic](#)
 - [Python](#)
 - [Java](#)
- If I create a new random number generator object for each request, why does it return the same value repeatedly? (Or: only seed the random generator once)
 - [Java](#)
 - [C#](#)
 - [C++](#)
- How do I generate a random integer value within a range?
 - [Java](#)



Questions

[Q: How to request a random row in SQL?](#)

Tags: [random](#) ([Prev Q](#))

How can I request a random row (or as close to truly random as is possible) in pure SQL?

Tags: [random](#) ([Prev Q](#))

User: [sverrejoh](#)

[Answer](#) by [yaakov-ellis](#)

See this post: [SQL to Select a random row from a database table](#). It goes through methods for doing this in MySQL, PostgreSQL, Microsoft SQL Server, IBM DB2 and Oracle (the following is copied from that link):

Select a random row with MySQL:

```
SELECT column FROM table  
ORDER BY RAND()  
LIMIT 1
```

Select a random row with PostgreSQL:

```
SELECT column FROM table  
ORDER BY RANDOM()  
LIMIT 1
```

Select a random row with Microsoft SQL Server:

```
SELECT TOP 1 column FROM table  
ORDER BY NEWID()
```

Select a random row with IBM DB2

```
SELECT column, RAND() as IDX  
FROM table  
ORDER BY IDX FETCH FIRST 1 ROWS ONLY
```

Select a random record with Oracle:

```
SELECT column FROM  
( SELECT column FROM table  
ORDER BY dbms_random.value )  
WHERE rownum = 1
```

[Answer](#) by [grey-panther](#)

Solutions like Jeremies:

```
SELECT * FROM table ORDER BY RAND() LIMIT 1
```

work, but they need a sequential scan of all the table (because the random value associated with each row needs to be calculated - so that the smallest one can be determined), which can be quite slow for even medium sized tables. My recommendation would be to use some kind of indexed numeric column (many tables have these as their primary keys), and then write something like:

```
SELECT * FROM table WHERE num_value >= RAND() *
  ( SELECT MAX (num_value ) FROM table )
ORDER BY num_value LIMIT 1
```

This works in logarithmic time, regardless of the table size, if `num_value` is indexed. One caveat: this assumes that `num_value` is equally distributed in the range $0 \dots \text{MAX}(\text{num_value})$. If your dataset strongly deviates from this assumption, you will get skewed results (some rows will appear more often than others).

[Answer](#) by [matt-hamilton](#)

I don't know how efficient this is, but I've used it before:

```
SELECT TOP 1 * FROM MyTable ORDER BY newid()
```

Because GUIDs are pretty random, the ordering means you get a random row.

Tags: [random](#) ([Prev Q](#))

Duplicates

[Skip to questions](#),

Wiki by user [alain](#) 

A duplicate is any re-occurrence of an item in a collection. This can be as simple as two identical strings in a list of strings, or multiple complex objects which are treated as the same object when compared to each other.

This tag may pertain to questions about preventing, detecting, removing, or otherwise dealing with unwanted duplicates, or adapting to safely allow duplicates.

Questions

[Q: Finding duplicate values in a SQL table](#)

Tags: [duplicates](#) ([Prev Q](#))

It's easy to find duplicates with one field:

```
SELECT name, COUNT(email)
FROM users
GROUP BY email
HAVING ( COUNT(email) > 1 )
```

So if we have a table

ID	NAME	EMAIL
1	John	asd@asd.com
2	Sam	asd@asd.com
3	Tom	asd@asd.com
4	Bob	bob@asd.com
5	Tom	asd@asd.com

This query will give us John, Sam, Tom, Tom because they all have the same email.

However, what I want is to get duplicates with the same email **and** name.

That is, I want to get "Tom", "Tom".

The reason I need this: I made a mistake, and allowed to insert duplicate name and email values. Now I need to remove/change the duplicates, so I need to *find* them first.

Tags: [duplicates](#) ([Prev Q](#))

User: [alex](#)

[Answer](#) by [gbn](#)

```
SELECT
    name, email, COUNT(*)
FROM
    users
GROUP BY
    name, email
HAVING
    COUNT(*) > 1
```

Simply group on the both columns

Note: the ANSI standard is to have all non aggregated columns in the GROUP BY.

MySQL allows you to avoid this, but results are unpredictable:

- [GROUP BY lname ORDER BY showing wrong results](#)
- [Which is the least expensive aggregate function in the absence of ANY\(\)](#) (see comments in accepted answer)

[Answer](#) by [km.](#)

try this:

[Skip code block](#)

```

declare @YourTable table (id int, name varchar(10), email varchar(50))

INSERT @YourTable VALUES (1,'John','John-email')
INSERT @YourTable VALUES (2,'John','John-email')
INSERT @YourTable VALUES (3,'fred','John-email')
INSERT @YourTable VALUES (4,'fred','fred-email')
INSERT @YourTable VALUES (5,'sam','sam-email')
INSERT @YourTable VALUES (6,'sam','sam-email')

SELECT
    name, email, COUNT(*) AS Countof
    FROM @YourTable
    GROUP BY name, email
    HAVING COUNT(*)>1

```

OUTPUT:

name	email	Countof
John	John-email	2
sam	sam-email	2

(2 row(s) affected)

if you want the IDs of the dups use this:

```

SELECT
    y.id, y.name, y.email
    FROM @YourTable y
    INNER JOIN (SELECT
                    name, email, COUNT(*) AS CountOf
                    FROM @YourTable
                    GROUP BY name, email
                    HAVING COUNT(*)>1
                ) dt ON y.name=dt.name and y.email=dt.email

```

OUTPUT:

id	name	email
1	John	John-email
2	John	John-email
5	sam	sam-email
6	sam	sam-email

(4 row(s) affected)

to delete the duplicates try:

[Skip code block](#)

```

DELETE d
    FROM @YourTable d
    INNER JOIN (SELECT
                    y.id, y.name, y.email, ROW_NUMBER() OVER(PARTITION BY y.name, y.email ORDER BY y.
                    FROM @YourTable y
                    INNER JOIN (SELECT
                        name, email, COUNT(*) AS Countof
                        FROM @YourTable
                        GROUP BY name, email
                        HAVING COUNT(*)>1
                    ) dt ON y.name=dt.name and y.email=dt.email
                    ) dt2 ON d.id=dt2.id
    WHERE dt2.RowRank!=1
select * FROM @YourTable

```

OUTPUT:

id	name	email
1	John	John-email
3	fred	John-email

4	fred	fred-email
5	sam	sam-email

(4 row(s) affected)

Tags: [duplicates](#) ([Prev Q](#))

NoSQL

[Skip to questions](#),

Wiki by user [coδεικεδις](#) 

NoSQL (sometimes expanded to “not only [sql](#)“) is a broad class of database management systems that differ from the classic model of the relational database management system ([rdbms](#) ) in some significant ways.

NoSQL systems:

- Specifically designed for high load
- Natively support horizontal scalability
- Fault tolerant
- Store data in denormalised manner
- Do not usually enforce strict database schema
- Do not usually store data in a table
- Sometimes provide eventual consistency instead of ACID transactions

In contrast to RDBMS, NoSQL systems:

- Do not guarantee data consistency
- Usually support a limited query language (subset of SQL or another custom query language)
- May not provide support for transactions/distributed transactions
- Do not usually use some advanced concepts of RDBMS, such as triggers, views, stored procedures

NoSQL implementations can be categorised by their manner of implementation:

- [Column-oriented](#) 
- [Document store](#) 
- [Graph](#) 
- [Key-value store](#) 
- [Multivalue databases](#) 
- [Object databases](#) 
- [Triplestore](#) 
- [Tuple store](#) 

Free NoSQL Books

- [CouchDB: The Definitive Guide](#) 
- [DBA's Guide to NoSQL and Apache Cassandra](#) 
- [The Little MongoDB Book](#) 

- [The Little Redis Book](#) 
-

Questions

[Q: Use cases for NoSQL](#)

Tags: [nosql](#) ([Prev Q](#))

NoSQL has been getting a lot of attention in our industry recently. I'm really interested in what peoples thoughts are on the best use-cases for its use over relational database storage. What should trigger a developer into thinking that particular datasets are more suited to a NoSQL solution. I'm particularly interested in [MongoDB](#)  and [CouchDB](#)  as they seem to be getting the most coverage with regard to PHP development and that is my focus.

Tags: [nosql](#) ([Prev Q](#))

User: [seengee](#) 

[Answer](#)  by [spacemonkey](#) 

Just promise yourself that you will never try to map a relational data model to a NoSQL database like MongoDB or CouchDB... This is the most common mistake developers make when evaluating emerging tech.

That approach is analogous to taking a car and trying to use it to pull your cart down the road like a horse.

It's a natural reaction due to everyone's experience of course, but the real value in using a document database is being able to simplify your datamodel and minimize your suffering as a developer. Your codebase will shrink, your bugs will be fewer and easier to find, performance is going to be awesome, and scale will be much simpler.

As a Joomla founder I'm biased :-) but coming from the CMS space, something like MongoDB is a silver bullet as content maps very naturally to document systems.

Another great case for MongoDB is real-time analytics, as MongoDB has very strong performance and scale particularly regarding concurrency. There are case studies at the MongoDB.org website that demonstrate those attributes.

I agree with the notion that each database has its own aims and use cases; take the purpose of each database for evaluation accordingly.

Tags: [nosql](#) ([Prev Q](#))

LINQ

[Skip to questions,](#)

Wiki by user [albin-sunnanbo](#) 

[linq](#) is a .NET-based DSL (Domain Specific Language), introduced in [.net-3.5](#), for querying data sources such as databases, XML files or in-memory object lists. All these data sources can be queried using the exact same, readable and easy-to-use syntax - or rather, *syntaxes*, because LINQ supports two notations:

- Inline LINQ or query syntax, where queries are expressed in a SQL-like language, with dialects in both [C#](#) and [VB.NET](#).
- Fluent LINQ or query operators, where queries are expressed as [lambda expressions](#) and can be linked (LINQed?) using a fluent syntax.

All LINQ query operations consist of three distinct actions: 1. Obtain the data source. 2. Create the query. 3. Execute the query.

Major Implementations:

.NET languages (C#, F#, VB.NET)

Some examples:

Fluent syntax (C#)

```
var result = dbContext.Products
    .Where(p => p.Category.Name == "Toys" && p.Price >= 2.50)
    .Select(p => p.Name);
```

Query syntax (C#)

```
var result = from product in dbContext.Products
    where product.Category.Name == "Toys"
    where product.Price >= 2.50
    select product.Name;
```

Query syntax (VB.NET)

```
Dim result = From product In dbContext.Products _
    Where product.Category.Name = "Toys" _
    Where product.Price >= 2.50 _
    Select product.Name
```

This query would return the name of all products in the “Toys” category with a price greater than or equal to 2.50.

Flavors

LINQ comes in many flavors, the most notable are

- [LINQ to Objects](#) - For querying collections of POCO (Plain old CLR objects)
- [LINQ to SQL](#) - For querying SQL databases
- [LINQ to Entities](#) - For querying SQL databases through [Entity Framework](#)

- [LINQ to XML](#) - For querying XML documents
- [PLINQ](#) - for querying in parallel.

LINQ has many extensions implemented online and a variety of extension open-source projects like [MORELINQ](#) which adds more operators to the .NET operators, and many others.

There are other implementations of LINQ which can be found on the Internet, such as [LINQ to SharePoint](#), [LINQ to Twitter](#), [LINQ to CSV](#), [LINQ to Excel](#), [LINQ to JSON](#) and [LINQ to Google](#).

Questions

[Q: SQL to LINQ Tool](#)

Tags: [linq](#) ([Prev Q](#))

Is there a tool out there which can convert SQL syntax to LINQ syntax?

I just want to rewrite basic queries with join, etc., to [LINQ](#). It would save me a lot of time.

Tags: [linq](#) ([Prev Q](#))

User: [chris](#) 

[Answer](#)  by [nikki9696](#) 

[Linqer](#) is a SQL to LINQ converter tool. It helps you to learn LINQ and convert your existing SQL statements.

Not every SQL statement can be converted to LINQ, but Linqer covers many different types of SQL expressions. Linqer supports both .NET languages - C# and Visual Basic.

Tags: [linq](#) ([Prev Q](#))

ORM

[Skip to questions](#),

Wiki by user [dean-kuga](#) 

Object-Relational Mapping

Object-Relational Mapping (ORM) is a technique for mapping object-oriented systems to relational databases. This creates, in effect, a “virtual object database” that can be used from within the programming language. There are both free and commercial packages available that perform object-relational mapping.

Resources

- [Wikipedia](#) 
- [Anders Hejlsberg](#) 
- [List of object-relational mapping software](#) 
- [Comparison of object-relational mapping software](#) 

Related tags

- [database](#)
 - [relational-database](#) 
 - [hibernate](#) 
 - [jpa](#) 
 - [toplink](#) 
 - [ebean](#) 
 - [peewee](#) 
 - [sugarorm](#) 
 - [eloquent](#) 
 - [nhibernate](#) 
 - [gorm](#) 
 - [idiorm](#) 
 - [propel](#) 
 - [sequelize.js](#) 
-

Questions

[Q: Using an ORM or plain SQL?](#)

Tags: [orm](#) ([Prev Q](#))

For some of the apps I've developed (then proceeded to forget about), I've been writing plain SQL, primarily for MySQL. Though I have used ORMs in python like [SQLAlchemy](#), I didn't stick with them for long. Usually it was either the documentation or complexity (from my point of view) holding me back.

I see it like this: use an ORM for portability, plain SQL if it's just going to be using one type of database. I'm really looking for advice on when to use an ORM or SQL when developing an app that needs database support.

Thinking about it, it would be far better to just use a lightweight wrapper to handle database inconsistencies vs. using an ORM.

Tags: [orm](#) ([Prev Q](#))

User: [hydrapheetz](#) 

[Answer](#)  by [cameron-pope](#) 

ORMs have some nice features. They can handle much of the dog-work of copying database columns to object fields. They usually handle converting the language's date and time types to the appropriate database type. They generally handle one-to-many relationships pretty elegantly as well by instantiating nested objects. I've found if you design your database with the strengths and weaknesses of the ORM in mind, it saves a lot of work in getting data in and out of the database. (You'll want to know how it handles polymorphism and many-to-many relationships if you need to map those. It's these two domains that provide most of the 'impedance mismatch' that makes some call ORM the 'vietnam of computer science'.)

For applications that are transactional, i.e. you make a request, get some objects, traverse them to get some data and render it on a Web page, the performance tax is small, and in many cases ORM can be faster because it will cache objects it's seen before, that otherwise would have queried the database multiple times.

For applications that are reporting-heavy, or deal with a large number of database rows per request, the ORM tax is much heavier, and the caching that they do turns into a big, useless memory-hogging burden. In that case, simple SQL mapping (LinQ or iBatis) or hand-coded SQL queries in a thin DAL is the way to go.

I've found for any large-scale application you'll find yourself using both approaches. (ORM for straightforward CRUD and SQL/thin DAL for reporting).

[Answer](#)  by [cletus](#) 

Speaking as someone who spent quite a bit of time working with JPA (Java Persistence API, basically the standardized ORM API for Java/J2EE/EJB), which includes Hibernate,

EclipseLink, Toplink, OpenJPA and others, I'll share some of my observations.

1. ORMs are not fast. They can be adequate and most of the time adequate is OK but in a high-volume low-latency environment they're a no-no;
2. In general purpose programming languages like Java and C# you need an awful lot of magic to make them work (eg load-time weaving in Java, instrumentation, etc);
3. When using an ORM, rather than getting further from SQL (which seems to be the intent), you'll be amazed how much time you spend tweaking XML and/or annotations/attributes to get your ORM to generate performant SQL;
4. For complex queries, there really is no substitute. Like in JPA there are some queries that simply aren't possible that are in raw SQL and when you have to use raw SQL in JPA it's not pretty (C#/Net at least has dynamic types—var—which is a lot nicer than an Object array);
5. There are an awful lot of "gotchas" when using ORMs. This includes unintended or unexpected behavior, the fact that you have to build in the capability to do SQL updates to your database (by using refresh() in JPA or similar methods because JPA by default caches everything so it won't catch a direct database update—running direct SQL updates is a common production support activity);
6. The object-relational mismatch is always going to cause problems. With any such problem there is a tradeoff between complexity and completeness of the abstraction. At times I felt JPA went too far and hit a real law of diminishing returns where the complexity hit wasn't justified by the abstraction.

There's another problem which takes a bit more explanation.

The traditional model for a Web application is to have a persistence layer and a presentation layer (possibly with a services or other layers in between but these are the important two for this discussion). ORMs force a rigid view from your persistence layer up to the presentation layer (ie your entities).

One of the criticisms of more raw SQL methods is that you end up with all these VOs (value objects) or DTOs (data transfer objects) that are used by simply one query. This is touted as an advantage of ORMs because you get rid of that.

Thing is those problems don't go away with ORMs, they simply move up to the presentation layer. Instead of creating VOs/DTOs for queries, you create custom presentation objects, typically one for every view. How is this better? IMHO it isn't.

I've written about this in [ORM or SQL: Are we there yet?](#) 

My persistence technology of choice (in Java) these days is ibatis. It's a pretty thin wrapper around SQL that does 90%+ of what JPA can do (it can even do lazy-loading of relationships although its not well-documented) but with far less overhead (in terms of complexity and actual code).

This came up last year in a GWT application I was writing. Lots of translation from EclipseLink to presentation objects in the service implementation. If we were using ibatis it would've been far simpler to create the appropriate objects with ibatis and then pass them all the way up and down the stack. Some purists might argue this is Bad™. Maybe so (in theory) but I tell you what: it would've led to simpler code, a simpler stack and

more productivity.

Tags: [orm](#) ([Prev Q](#))

Foreign Keys

[Skip to questions](#),

Wiki by user [juankysmith](#) 

The **foreign key** identifies a column or set of columns in one (referencing) table that refers to a column or set of columns in another (referenced) table. The columns in the referencing table must reference the columns of the primary key or other superkey in the referenced table. The values in one row of the referencing columns must occur in a single row in the referenced table.

Thus, a row in the referencing table cannot contain values that don't exist in the referenced table (except potentially NULL). This way references can be made to link information together and it is an essential part of database normalization.

Multiple rows in the referencing table may refer to the same row in the referenced table. Most of the time, it reflects the one (parent table or referenced table) to many (child table, or referencing table) relationship.

Composite foreign key consists of two or more columns. All the columns that form a single foreign key reference one table.

In some cases foreign key constraint can reference the same table. For example for *employee* table, having *employee_number*, *employee_name*, and *manager_employee_number*. Every manager is also an employee, so it can be built a foreign key constraints between *manager_employee_number* to the *employee_number*

Reference

- [Foreign key \(Wikipedia\)](#) 
 - [Foreign Key Constraint on technet](#) 
-

Questions

[Q: When to use “ON UPDATE CASCADE”](#)

Tags: [foreign-keys](#) ([Prev Q](#))

I use “ON DELETE CASCADE” regularly but I never use “ON UPDATE CASCADE” as I am not so sure in what situation it will be useful.

For the sake of discussion let see some code.

[Skip code block](#)

```
CREATE TABLE parent (
    id INT NOT NULL AUTO_INCREMENT,
    PRIMARY KEY (id)
);

CREATE TABLE child (
    id INT NOT NULL AUTO_INCREMENT, parent_id INT,
    INDEX par_ind (parent_id),
    FOREIGN KEY (parent_id)
        REFERENCES parent(id)
        ON DELETE CASCADE
);
```

For “ON DELETE CASCADE”, if a parent with an `id` is deleted, a record in child with `parent_id = parent.id` will be automatically deleted. This should be no problem.

- (1) This means that “ON UPDATE CASCADE” will do the same thing when `id` of the parent is updated?
- (2) If (1) is true, it means that there is no need to use “ON UPDATE CASCADE” if `parent.id` is not updatable (or will never be updated) like when it is `AUTO_INCREMENT` or always set to be `TIMESTAMP`. Is that right?
- (3) If (2) is not true, in what other kind of situation should we use “ON UPDATE CASCADE”?
- (4) What if I (for some reason) update the `child.parent_id` to be something not existing, will it then be automatically deleted?

Well, I know, some of the question above can be test programmically to understand but I want also know if any of this is database vendor dependent or not.

Please shed some light.

Tags: [foreign-keys](#) ([Prev Q](#))

User: [nawaman](#)

[Answer](#) by [c-pound-guru](#)

It’s true that if your primary key is just a identity value auto incremented, you would have no real use for ON UPDATE CASCADE.

However, let’s say that your primary key is a 10 digit UPC bar code and because of expansion, you need to change it to a 13-digit UPC bar code. In that case, ON UPDATE CASCADE would allow you to change the primary key value and any tables that have

foreign key references to the value will be changed accordingly.

In reference to #4—if you change the child ID to something that doesn’t exist in the parent table (and you have referential integrity), you should get a foreign key error.

Tags: [foreign-keys](#) ([Prev Q](#))

CASE

[Skip to questions,](#)

Wiki by user [mu-is-too-short](#) 

The CASE statement is an SQL control structure that is similar to switch from other languages. There are two forms of the CASE statement. The first is the simple CASE:

```
CASE search_expr
WHEN compare_expr1 THEN result_expr1
WHEN compare_expr2 THEN result_expr2
ELSE else_expr1
END
```

This will compare `search_expr` to each `compare_expr` in order until it finds a condition where `search_expr = compar_expr`, and return the `result_expr` of that condition. If no such condition is found the `else_expr` is returned.

The second form of CASE expression is the searched CASE:

```
CASE
WHEN boolean_expr1 then result_expr1
WHEN boolean_expr2 then result_expr1
ELSE else_expr1
END
```

This will test `boolean_expr` in order until it finds a condition where the `boolean_expr` is true, and return the `result_expr` of that condition. If no such condition is found the `else_expr` is returned.

Questions

[Q: SQL Switch/Case in where clause](#)

Tags: [case](#) ([Prev Q](#))

I tried searching around but couldn't find anything that would help me out.

I'm trying to do this in SQL:

[Skip code block](#)

```
declare @locationType varchar(50);
declare @locationID int;

SELECT column1, column2
FROM viewwhatever
WHERE
CASE @locationType
    WHEN 'location' THEN account_location = @locationID
    WHEN 'area' THEN xxx_location_area = @locationID
    WHEN 'division' THEN xxx_location_division = @locationID
```

I know that I shouldn't have to put '= @locationID' at the end of each one but I can't get the syntax even close to being correct. SQL keeps complaining about my '=' on the first WHEN line...

Help!

Tags: [case](#) ([Prev Q](#))

User: [miles](#)

[Answer](#) by [bob-probst](#)

[Skip code block](#)

```
declare @locationType varchar(50);
declare @locationID int;

SELECT column1, column2
FROM viewwhatever
WHERE
@locationID =
CASE @locationType
    WHEN 'location' THEN account_location
    WHEN 'area' THEN xxx_location_area
    WHEN 'division' THEN xxx_location_division
END
```

[Answer](#) by [lukek](#)

without a case statement...

```
SELECT column1, column2
FROM viewwhatever
WHERE
    (@locationType = 'location' AND account_location = @locationID)
    OR
    (@locationType = 'area' AND xxx_location_area = @locationID)
    OR
    (@locationType = 'division' AND xxx_location_division = @locationID)
```

Tags: [case](#) ([Prev Q](#))

String Concatenation

[Skip to questions,](#)

Wiki by user [jaguar](#) 

In particular in the theory of computation, the concatenation operation on strings is generalized to an operation on sets of strings as follows:

For two sets of strings S_1 and S_2 , the concatenation S_1S_2 consists of all strings of the form vw where v is a string from S_1 and w is a string from S_2 . In this definition, the string vw is the ordinary concatenation of strings v and w as defined in the introductory section. In this context, sets of strings are often referred to as formal languages.

Questions

[Q: Concatenate many rows into a single text string?](#)

Tags: [string-concatenation](#) ([Prev Q](#))

Consider a database table holding names, with three rows:

```
Peter  
Paul  
Mary
```

Is there an easy way to turn this into a single string of Peter, Paul, Mary?

Tags: [string-concatenation](#) ([Prev Q](#))

User: [johnnym](#)

[Answer](#) by ritesh

I had a similar issue when I was trying to join two tables with one-to-many relationships. In SQL 2005 I found that XML PATH method can handle the concatenation of the rows very easily.

If there is a table called STUDENTS

```
SubjectID      StudentName  
-----  
1              Mary  
1              John  
1              Sam  
2              Alaina  
2              Edward
```

Result I expected was:

```
SubjectID      StudentName  
-----  
1              Mary, John, Sam  
2              Alaina, Edward
```

I used the following T-SQL:

[Skip code block](#)

```
Select Main.SubjectID,  
       Left(Main.Students,Len(Main.Students)-1) As "Students"  
From  
  (  
    Select distinct ST2.SubjectID,  
    (  
      Select ST1.StudentName + ',' AS [text()]  
      From dbo.Students ST1  
      Where ST1.SubjectID = ST2.SubjectID  
      ORDER BY ST1.SubjectID  
      For XML PATH ('')  
    ) [Students]  
    From dbo.Students ST2  
  ) [Main]
```

You can do the same thing in a more compact way if you can concat the commas at the beginning and use substring to skip the first one so you don't need to do a subquery:

[Skip code block](#)

```

Select distinct ST2.SubjectID,
    substring(
        (
            Select ',' +ST1.StudentName AS [text()]
            From dbo.Students ST1
            Where ST1.SubjectID = ST2.SubjectID
            ORDER BY ST1.SubjectID
            For XML PATH ('')
        ), 2, 1000) [Students]
From dbo.Students ST2

```

[Answer](#) by [chris-shaffer](#)

Use COALESCE:

```

DECLARE @Names VARCHAR(8000)
SELECT @Names = COALESCE(@Names + ', ', '') + Name
FROM People

```

Just some explanation (since this answer seems to get relatively regular views):

- Coalesce is really just a helpful cheat that accomplishes two things:
 - 1) No need to initialize @Names with an empty string value.
 - 2) No need to strip off an extra separator at the end.
- The solution above will give incorrect results if a row has a *NULL* Name value (if there is a *NULL*, the *NULL* will make @Names *NULL* after that row, and the next row will start over as an empty string again. Easily fixed with one of two solutions:

```

DECLARE @Names VARCHAR(8000)
SELECT @Names = COALESCE(@Names + ', ', '') + Name
FROM People
WHERE Name IS NOT NULL

```

or:

```

DECLARE @Names VARCHAR(8000)
SELECT @Names = COALESCE(@Names + ', ', '') +
    ISNULL(Name, 'N/A')
FROM People

```

Depending on what behavior you want (the first option just filters *NULL*s out, the second option keeps them in the list with a marker message [replace 'N/A' with whatever is appropriate for you]).

[Answer](#) by [jens-frandsen](#)

One method not yet shown via the XML data() command in MS SQL Server is:

Assume table called NameList with one column called FName,

```

SELECT FName + ', ' AS 'data()'
FROM NameList
FOR XML PATH('')

```

returns:

```
"Peter, Paul, Mary, "
```

Only the extra comma must be dealt with.

Tags: [string-concatenation](#) ([Prev Q](#))

Case Sensitive

Questions

[Q: Is SQL syntax case sensitive?](#)

Tags: [case-sensitive](#) ([Prev Q](#))

Is SQL case sensitive. I've used MySQL and SQL Server which both seem to be case insensitive. Is this always the case? Does the standard define case-sensitivity?

Tags: [case-sensitive](#) ([Prev Q](#))

User: [steve-tranby](#) 

[Answer](#)  by [stefan-rusek](#) 

The SQL Keywords are case-insensitive (SELECT, FROM, WHERE, etc), but are often written in all caps. However in some setups table and column names are case-sensitive. MySQL has a configuration option to enable/disable it. Usually case-sensitive table and column names are the default on Linux MySQL and case-insensitive used to be the default on Windows, but now the installer asked about this during setup. For MSSQL it is a function of the database's collation setting.

Here is the [MySQL page about name case-sensitivity](#) 

Here is the [article in MSDN about collations for MSSQL](#) 

Tags: [case-sensitive](#) ([Prev Q](#))

Indexing

[Skip to questions](#),

Wiki by user [greeness](#) 

The purpose of storing an index is to optimize speed and performance in finding relevant documents for a search query. Without an index, the search process would scan every document in the corpus, which would require considerable time and computing power.

Indexes may benefit both read queries and updates. Many people wrongly believe indexes are only good for read queries. In general there are three methods of indexing - non-clustered, clustered and cluster.

References:

- [How do MySQL indexes work?](#) 
 - [Use the index, Luke](#) 
 - [Database index on Wiki](#) 
-

Questions

[Q: What is an index in SQL?](#)

Tags: [indexing](#) ([Prev Q](#))

What is an *index* in SQL? Can you explain or reference to understand clearly?

Where should I use an index?

Tags: [indexing](#) ([Prev Q](#))

User: [surya-sasidhar](#) 

[Answer](#)  by [emil-vikström](#) 

An index is used to speed up searching in the database. MySQL have some good documentation on the subject (which is relevant for other SQL servers as well):

<http://dev.mysql.com/doc/refman/5.0/en/mysql-indexes.html> 

An index can be used to efficiently find all row matching some column in your query and then walk through only that subset of the table to find exact matches. If you don't have indexes on any column in the WHERE clause, the SQL server have to walk through *the whole table* and check every row to see if it matches, which may be a slow operation on big tables.

The index can also be a UNIQUE index, which means that you cannot have duplicate values in that column, or a PRIMARY KEY which in some storage engines defines where in the database file the value is stored.

In MySQL you can use EXPLAIN in front of your SELECT statement to see if your query will make use of any index. This is a good start for troubleshooting performance problems.

Read more here: <http://dev.mysql.com/doc/refman/5.0/en/explain.html> 

Tags: [indexing](#) ([Prev Q](#))

Triggers

[Skip to questions,](#)

Wiki by user [dave-clemmer](#) 

Triggers are rules, that when evaluate to true perform one or more actions. Triggers can be defined at various tiers in applications and processes (such as database triggers to change data and ui triggers to change properties or fire events).

[WPF Trigger](#)  - The properties changed by triggers are automatically reset to their previous value when the triggered condition is no longer satisfied.

A trigger is a PL/SQL block structure which is fired when a DML statements like:

- INSERT
- DELETE
- UPDATE

is executed on a database table. A trigger is triggered automatically when an associated DML statement is executed.

We can also divide triggers in two groups:

- Row Level Triggers (executed for each row that is affected by triggering command)
- Statement Level Triggers (executed once for triggering command no matter how many rows are affected)

A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database. The trigger is mostly used for maintaining the integrity of the information on the database. Source : [Database trigger](#) 

A trigger is a special kind of a store procedure that executes in response to certain action on the table like insertion, deletion or update of data. It is a database object which is bound to a table and is executed automatically. You can't explicitly invoke triggers.

Reference

- [Oracle](#) 
 - [MySQL](#) 
 - [HSQLDB](#) 
 - [SQLite](#) 
 - [PostgreSQL](#) 
-

Questions

[Q: Are database triggers evil?](#)

Tags: [triggers](#) ([Prev Q](#))

Are database triggers a bad idea?

In my experience they are evil, because they can result in surprising side effects, and are difficult to debug (especially when one trigger fires another). Often developers do not even think of looking if there is a trigger.

On the other hand, it seems like if you have logic that must occur everytime a new FOO is created in the database then the most foolproof place to put it is an insert trigger on the FOO table.

The only time we're using triggers is for really simple things like setting the ModifiedDate.

Tags: [triggers](#) ([Prev Q](#))

User: [ww.](#) 

[Answer](#)  by [dkretz](#) 

The main problems with triggers are a) they are completely Global - they apply no matter what the context of the table activity; and b) they are stealthy; it's easy to forget they are there until they hurt you with unintended (and very mysterious) consequences.

Which just means they need to be carefully used for the proper circumstances; which in my experience is limited to relational integrity issues (sometimes with finer granularity than you can get declaratively); and usually not for business or transactional purposes. YMMV.

Tags: [triggers](#) ([Prev Q](#))

Sql Order By

[Skip to questions,](#)

Wiki by user [szyomon](#) 

An ORDER BY clause in [sql](#) specifies that a SQL SELECT statement returns a result set with the rows being sorted by the values of one or more columns.

Reference

- [Microsoft SQL Server](#) 
 - [Oracle](#) 
 - [MySQL](#) 
 - [SQLite](#) 
 - [PostgreSQL](#) 
-

Questions

[Q: SQL multiple column ordering](#)

Tags: [sql-order-by](#) ([Prev Q](#))

I am trying to sort by multiple columns in SQL, and in different directions. `column1` would be sorted descending, and `column2` ascending.

How can I do this?

Tags: [sql-order-by](#) ([Prev Q](#))

User: [señor-reginold-francis](#)

Answer by [ignacio-vazquez-abrams](#)

```
ORDER BY column1 DESC, column2
```

This sorts everything by `column1` (descending) first, and then by `column2` (ascending, which is the default) whenever the `column1` fields for two rows are equal.

Answer by [thomas-c.-g.-de-vilhena](#)

The other answers lack a concrete example, so here it goes:

Given the following **People** table:

FirstName	LastName	YearOfBirth
Thomas	Alva Edison	1847
Benjamin	Franklin	1706
Thomas	More	1478
Thomas	Jefferson	1826

If you execute the query below:

```
SELECT * FROM People ORDER BY FirstName DESC, YearOfBirth ASC
```

The result set will look like this:

FirstName	LastName	YearOfBirth
Thomas	More	1478
Thomas	Jefferson	1826
Thomas	Alva Edison	1847
Benjamin	Franklin	1706

Answer by [quassnoi](#)

```
SELECT *
FROM    mytable
ORDER BY
        coloumn1 DESC, coloumn2 ASC
```

Tags: [sql-order-by](#) ([Prev Q](#))

Hierarchical Data

[Skip to questions](#),

Wiki by user [iterator](#) 

Hierarchical data reflects a set of parent-child relationships. As Materials Resource Planning was developed by manufacturers in the 1950s, hierarchical data management emerged as a theoretical and practical discipline. A manufacturer uses the term Bill of Materials (BOM) for the set of items required to assemble a parent item; the term BOM was adopted by mathematicians and software developers, and it is used today in both realms.

Hierarchical data is ubiquitous and has called forth some of the most important and complex work of database management. It has also propelled object-oriented programming.

The key design element is recursion. Imagine starting at the trunk of a tree and visiting every leaf; this theoretical exercise would require an enormously large number of decision points occurring in a nested or recursive pattern. Requirements for this analysis are remarkably varied and have produced entire sub-disciplines of database design.

- [Wikipedia entry](#) 
-

Questions

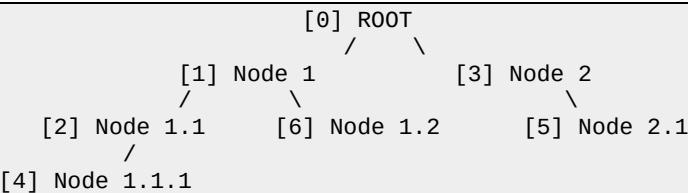
[Q: What is the most efficient/elegant way to parse a flat table into a tree?](#)

Tags: [hierarchical-data](#) ([Prev Q](#))

Assume you have a flat table that stores an ordered tree hierarchy:

Id	Name	ParentId	Order
1	'Node 1'	0	10
2	'Node 1.1'	1	10
3	'Node 2'	0	20
4	'Node 1.1.1'	2	10
5	'Node 2.1'	3	10
6	'Node 1.2'	1	20

Here's a diagram, where we have [id] Name. Root node 0 is fictional.



What minimalistic approach would you use to output that to HTML (or text, for that matter) as a correctly ordered, correctly indented tree?

Assume further you only have basic data structures (arrays and hashmaps), no fancy objects with parent/children references, no ORM, no framework, just your two hands. The table is represented as a result set, which can be accessed randomly.

Pseudo code or plain English is okay, this is purely a conceptional question.

Bonus question: Is there a fundamentally better way to store a tree structure like this in a RDBMS?

EDITS AND ADDITIONS

To answer one commenter's ([Mark Bessey](#)'s) question: A root node is not necessary, because it is never going to be displayed anyway. ParentId = 0 is the convention to express "these are top level". The Order column defines how nodes with the same parent are going to be sorted.

The "result set" I spoke of can be pictured as an array of hashmaps (to stay in that terminology). For my example was meant to be already there. Some answers go the extra mile and construct it first, but that's okay.

The tree can be arbitrarily deep. Each node can have N children. I did not exactly have a "millions of entries" tree in mind, though.

Don't mistake my choice of node naming ('Node 1.1.1') for something to rely on. The nodes could equally well be called 'Frank' or 'Bob', no naming structure is implied, this was merely to make it readable.

I have posted my own solution so you guys can pull it to pieces.

Tags: [hierarchical-data](#) ([Prev Q](#))

User: [tomalak](#) 

[Answer](#)  by [bill-karwin](#) 

There are several ways to store tree-structured data in a relational database. What you show in your example uses two methods:

- **Adjacency List** (the “parent” column) and
- **Path Enumeration** (the dotted-numbers in your name column).

Another solution is called **Nested Sets**, and it can be stored in the same table too. Read [“Trees and Hierarchies in SQL for Smarties”](#) by Joe Celko for a lot more information on these designs.

I usually prefer a design called **Closure Table** (aka “Adjacency Relation”) for storing tree-structured data. It requires another table, but then querying trees is pretty easy.

I cover Closure Table in my presentation [Models for Hierarchical Data with SQL and PHP](#) and in my book [SQL Antipatterns: Avoiding the Pitfalls of Database Programming](#).

```
CREATE TABLE ClosureTable (
    ancestor_id    INT NOT NULL REFERENCES FlatTable(id),
    descendant_id INT NOT NULL REFERENCES FlatTable(id),
    PRIMARY KEY (ancestor_id, descendant_id)
);
```

Store all paths in the Closure Table, where there is a direct ancestry from one node to another. Include a row for each node to reference itself. For example, using the data set you showed in your question:

```
INSERT INTO ClosureTable (ancestor_id, descendant_id) VALUES
(1,1), (1,2), (1,4), (1,6),
(2,2), (2,4),
(3,3), (3,5),
(4,4),
(5,5),
(6,6);
```

Now you can get a tree starting at node 1 like this:

```
SELECT f.*
FROM FlatTable f
JOIN ClosureTable a ON (f.id = a.descendant_id)
WHERE a.ancestor_id = 1;
```

The output (in MySQL client) looks like the following:

```
+----+
| id |
+----+
| 1 |
| 2 |
| 4 |
| 6 |
+----+
```

In other words, nodes 3 and 5 are excluded, because they’re part of a separate hierarchy,

not descending from node 1.

Re: comment from e-satis about immediate children (or immediate parent). You can add a “path_length” column to the ClosureTable to make it easier to query specifically for an immediate child or parent (or any other distance).

```
INSERT INTO ClosureTable (ancestor_id, descendant_id, path_length) VALUES
(1,1,0), (1,2,1), (1,4,2), (1,6,1),
(2,2,0), (2,4,1),
(3,3,0), (3,5,1),
(4,4,0),
(5,5,0),
(6,6,0);
```

Then you can add a term in your search for querying the immediate children of a given node. These are descendants whose path_length is 1.

[Skip code block](#)

```
SELECT f.*
FROM FlatTable f
JOIN ClosureTable a ON (f.id = a.descendant_id)
WHERE a.ancestor_id = 1
AND path_length = 1;

+---+
| id |
+---+
| 2 |
| 6 |
+---+
```

Re comment from @ashraf: “How about sorting the whole tree [by name]?”

Here’s an example query to return all nodes that are descendants of node 1, join them to the FlatTable that contains other node attributes such as name, and sort by the name.

```
SELECT f.name
FROM FlatTable f
JOIN ClosureTable a ON (f.id = a.descendant_id)
WHERE a.ancestor_id = 1
ORDER BY f.name;
```

Re comment from @Nate:

[Skip code block](#)

```
SELECT f.name, GROUP_CONCAT(b.ancestor_id ORDER BY b.path_length DESC) AS breadcrumbs
FROM FlatTable f
JOIN ClosureTable a ON (f.id = a.descendant_id)
JOIN ClosureTable b ON (b.descendant_id = a.descendant_id)
WHERE a.ancestor_id = 1
GROUP BY a.descendant_id
ORDER BY f.name

+-----+-----+
| name | breadcrumbs |
+-----+-----+
| Node 1 | 1           |
| Node 1.1 | 1,2        |
| Node 1.1.1 | 1,2,4      |
| Node 1.2 | 1,6        |
+-----+-----+
```

A user suggested an edit today. SO moderators approved the edit, but I am reversing it.

The edit suggested that the ORDER BY in the last query above should be ORDER BY b.path_length, f.name, presumably to make sure the ordering matches the hierarchy. But this doesn't work, because it would order "Node 1.1.1" after "Node 1.2".

If you want the ordering to match the hierarchy in a sensible way, that is possible, but not simply by ordering by the path length. For example, see my answer to [MySQL Closure Table hierarchical database - How to pull information out in the correct order](#).

Tags: [hierarchical-data](#) ([Prev Q](#))

HTML

[Skip to questions,](#)

Wiki by user [daniel-a.-white](#) 

Questions regarding HTML should include a minimal [Stack Snippet](#) , a [JSFiddle](#)  or a [JSBin](#)  [demo](#)  of relevant code if requesting debugging help.

HTML ([HyperText Markup Language](#) ) is the main markup language for creating web pages (similar to [xml](#) ) and other information to be displayed in a [web-browser](#) . It was invented by Sir [Tim Berners-Lee](#)  while at [CERN](#)  to enable researchers to share their findings. The latest version for HTML is [HTML5](#) . A new version (update) of HTML5 - [HTML 5.1](#)  is already in progress with many new features and bug fixes.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages. All the web pages that are written in different programming languages ([php](#), [jsp](#) , [vf](#)  etc.) get rendered as HTML on the browser.

Syntax

HTML is written in the form of HTML elements consisting of tags (and their attributes) enclosed in angle brackets (like `<html>`), within the web page content. Angle brackets are also referred to as left & right guillemets or chevrons.

HTML tags most commonly come in pairs like `<h1>` and `</h1>`, although some tags, known as empty elements, are unpaired, for example ``, these are usually closed by putting a / at the end (Ex. ``). The first tag in a pair is the start tag, the second tag is the end tag (they are also called opening tags and closing tags). In between these tags authors can add text, tags, comments and other types of text-based content.

The purpose of a web browser is to read HTML documents and compose them into visible or audible web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page.

Standards

HTML standards, as well as those for many other web technologies, are maintained by the [World Wide Web Consortium](#)  ([w3c](#) <img alt="W3C icon" data-bbox="385 935 405 950).

[html4](#) was introduced in 1997, and the latest iteration, [html5](#) was recently developed by the W3C. What W3C calls *HTML5* is a subset of the HTML-Living-Standard which is specified by the [Web Hypertext Application Technology Working Group](#) (WHATWG).

The language specification and standards documents for HTML5 are available online [here](#).

FAQ

- [HTML5 Elements List](#)
 - [HTML5-tagged on Stack OverFlow](#)
-

References :

- [W3C HTML/Specifications](#)
 - [HTML-Living-Standard](#)
 - [W3C HTML Validator](#)
 - [HTML Reference at the Mozilla Developer Network \(MDN\)](#)
 - [WebPlatform](#)
 - [Web development features - IE Dev Center](#)
 - [SitePoint HTML Reference](#)
 - [Way2Tutorial HTML Reference](#)
 - [Web Education Community Group Wiki](#)
 - [W3C Web Content Accessibility Guidelines](#)
 - [HTML Feature Requests](#)
-

Free HTML Books :

- [Dive Into HTML5](#)
 - [HTML Dog Tutorials](#)
 - [Shay Howe's Learn to Code HTML & CSS](#)
-

Related Tags :

[html5](#) [xhtml](#) [markup](#) [mhtml](#) [dhtml](#) [dhtmlx](#) [mshtml](#)

Questions

[Q: What is a reasonable length limit on person “Name” fields?](#)

Tags: [html](#)

I have a simple webform that will allow unauthenticated users to input their information, including name. I gave the name field a limit of 50 characters to coincide with my database table where the field is varchar(50), but then I started to wonder.

Is it more appropriate to use something like the Text column type or should I limit the length of the name to something reasonable?

I'm using SQL Server 2005, in case that matters in your response.

EDIT: I did not see [this broader question](#)  regarding similar issues.

Tags: [html](#)

User: [endangeredmassa](#) 

[Answer](#)  by [ian-nelson](#) 

[UK Government Data Standards Catalogue](#)  suggests 35 characters for each of Given Name and Family Name, or 70 characters for a single field to hold the Full Name.

Tags: [html](#)

Eclipse

Questions

[Q: Lightweight SQL editor for Eclipse](#)

Tags: [eclipse](#)

Is there any simple SQL editor plugin for Eclipse?

By simple I mean, the editor does **NOT** connect to any DB, does syntax highlighting and preferably formatting sql is a bonus.

Tags: [eclipse](#)

User: [nimcap](#) 

[Answer](#)  by [ben-k](#) 

for reference to add sql syntax highlighting / coloring to eclipse you can install `data tools platform extender sdk` from “install new software”, under `Database Development`

or enter site location directly <http://download.eclipse.org/datatools/updates> 

Tags: [eclipse](#)

Sorting

[Skip to questions](#),

Wiki by user [gustavo-mori](#)

Sorting is the process of applying some order to a collection of items. There are various [algorithms](#) for doing this, including [mergesort](#), [quicksort](#), and [heapsort](#).

From the [Sorting Algorithms Wikipedia entry](#):

In [computer-science](#) a [sorting-algorithm](#) is an *algorithm* that puts elements of a list in a certain order. The most-used orders are [numerical](#) order and [lexicographical](#) order. Efficient sorting is important for optimizing the use of other algorithms (such as search and merge algorithms) that require sorted lists to work correctly; it is also often useful for [canonicalizing](#) data and for producing human-readable [output](#). More formally, the output must satisfy two conditions:

1. The output is in nondecreasing order (each element is no smaller than the previous element according to the desired total order);
2. The output is a permutation (reordering) of the input.

Related tags

- [quicksort](#)
 - [mergesort](#)
 - [bubble-sort](#)
 - [insertion-sort](#)
 - [heapsort](#)
 - [bucket-sort](#)
 - [radix-sort](#)
 - [natural-sort](#)
 - [topological-sort](#)
 - [selection-sort](#)
 - [alphabetical-sort](#)
 - [stable-sort](#)
-

Questions

[Q: SQL how to make null values come last when sorting ascending](#)

Tags: [sorting](#)

I have a SQL table with a datetime field. The field in question can be null. I have a query and I want the results sorted ascendingly by the datetime field, however I want rows where the datetime field is null at the end of the list, not at the beginning.

Is there a simple way to accomplish that?

Tags: [sorting](#)

User: [david-božjak](#)

[Answer](#) by [redfilter](#)

```
select MyDate
from MyTable
order by case when MyDate is null then 1 else 0 end, MyDate
```

[Answer](#) by [a_horse_with_no_name](#)

(A “bit” late, but this hasn’t been mentioned at all)

You didn’t specify your DBMS.

In standard SQL (and most modern DBMS like Oracle, PostgreSQL, DB2, Firebird, Apache Derby, HSQLDB and H2) you can specify `NULLS LAST` or `NULLS FIRST`:

Use `NULLS LAST` to sort them to the end:

```
select *
from some_table
order by some_column DESC NULLS LAST
```

Tags: [sorting](#)

Console

[Skip to questions,](#)

Wiki by user [andrew-lambert](#) 

A console is an interactive text-mode or text-based user interface to a software application or operating system. It may be implemented as a character-mode application or embedded in a graphical user interface.

It may be used as the main user interface or as a secondary interface for debugging and administrative tasks. The degree of interactivity varies widely between consoles.

Questions

[Q: Disable Rails SQL logging in console](#)

Tags: [console](#)

Is there a way to disable SQL query logging when I'm executing commands in the console? Ideally, it would be great if I can just disable it and re-enable it with a command in the console.

I'm trying to debug something and using "puts" to print out some relevant data. However, the sql query output is making it hard to read.

Edit: I found another solution, since setting the logger to nil sometimes raised an error, if something other than my code tried to call logger.warn

Instead of setting the logger to `nil` you can set the level of the logger to `1`.

```
ActiveRecord::Base.logger.level = 1 # or Logger::INFO
```

Tags: [console](#)

User: [gylaz](#) 

[Answer](#)  by [ryan-bigg](#) 

To turn it off:

```
old_logger = ActiveRecord::Base.logger  
ActiveRecord::Base.logger = nil
```

To turn it back on:

```
ActiveRecord::Base.logger = old_logger
```

Tags: [console](#)

UNION

[Skip to questions](#),

Wiki by user [dheer](#) 

`UNION` is a keyword of SQL for combining the results of multiple SQL queries. The results are combined and duplicate rows are eliminated (similar to `DISTINCT`). If `UNION ALL` is used, the rows are combined but duplicates are not removed.

Use the tag [unions](#)  for unions in C, C++, and similar languages. Use [union-all](#)  for the `UNION-ALL` keyword.

Reference

- [SQL Server](#) 
 - [Oracle](#) 
 - [MySQL](#) 
-

Questions

[Q: What is the difference between UNION and UNION ALL?](#)

Tags: [union](#)

What is the difference between UNION and UNION ALL.

Tags: [union](#)

User: [brian-g](#)

[Answer](#) by [jim-harte](#)

UNION removes duplicate records (where all columns in the results are the same), UNION ALL does not.

There is a performance hit when using UNION vs UNION ALL, since the database server must do additional work to remove the duplicate rows, but usually you do not want the duplicates (especially when developing reports).

UNION Example:

```
SELECT 'foo' AS bar UNION SELECT 'foo' AS bar
```

Result:

```
+----+  
| bar |  
+----+  
| foo |  
+----+  
1 row in set (0.00 sec)
```

UNION ALL example:

```
SELECT 'foo' AS bar UNION ALL SELECT 'foo' AS bar
```

Result:

```
+----+  
| bar |  
+----+  
| foo |  
| foo |  
+----+  
2 rows in set (0.00 sec)
```

Tags: [union](#)

Ms Access

Questions

Q: How to do INSERT into a table records extracted from another table

Tags: [ms-access](#)

I'm trying to write a query that extracts and transforms data from a table and then insert those data into another table. Yes, this is a data warehousing query and I'm doing it in MS Access. So basically I want some query like this:

```
INSERT INTO Table2(LongIntColumn2, CurrencyColumn2) VALUES  
(SELECT LongIntColumn1, Avg(CurrencyColumn) as CurrencyColumn1 FROM Table1 GROUP BY LongIntColumn1)
```

I tried but get a syntax error message.

What would you do if you want to do this?

Tags: [ms-access](#)

User: [martin08](#) 

[Answer](#)  by [pilsetnieks](#) 

No “VALUES”, no parenthesis:

```
INSERT INTO Table2(LongIntColumn2, CurrencyColumn2)  
SELECT LongIntColumn1, Avg(CurrencyColumn) as CurrencyColumn1 FROM Table1 GROUP BY LongIntColumn1;
```

Tags: [ms-access](#)

Anti Patterns

[Skip to questions](#),

Wiki by user [bryanh](#) 

An anti-pattern can be thought of as a “worst of breed” solution to a problem in that it may “work” for its intended purpose, but have unwanted side effects or incur technical debt. Anti-patterns can arise from improper education, insufficient experience or simple ignorance.

SQL-Injection

In short, this is an anti-pattern of applying unfiltered user input directly to a database query string. This may seem like a quick and easy solution to querying data, however it can lead to data corruption, security breeches, and so-forth.

Example:

```
query-string = "select * from users where id='\" + userid + '\";"
```

Assuming the `userid` variable came directly from the user, an attacker can cause issues by setting the value to something that would cause unexpected behavior:

```
userid = '' or 1=1;drop table users"
```

Tight coupling

Instead of keeping two distinct parts of an application separate (CSS and HTML; Business Logic and the View in an MVC application), the parts are mingled such that a change in one necessitates a change in the other.

Example:

CSS

```
.yellow { color: yellow; }
```

HTML

```
<div class="yellow">foo</div>
```

In order to change the formatting, either the content (HTML) must be changed by substituting a different class name, or the class definition (CSS) must be changed to something that doesn't match its name.

Others

Wikipedia has [other examples](#) .

Questions

[Q: What are the most common SQL anti-patterns?](#)

Tags: [anti-patterns](#)

All of us who work with relational databases have learned (or are learning) that SQL is different. Eliciting the desired results, and doing so efficiently, involves a tedious process partly characterized by learning unfamiliar paradigms, and finding out that some of our most familiar programming patterns don't work here. What are the common antipatterns you've seen (or yourself committed)?

Tags: [anti-patterns](#)

User: [dkretz](#)

Answer by [juliet](#)

I am consistently disappointed by most programmers' tendency to mix their UI-logic in the data access layer:

[Skip code block](#)

```
SELECT
    FirstName + ' ' + LastName as "Full Name",
    case UserRole
        when 2 then "Admin"
        when 1 then "Moderator"
        else "User"
    end as "User's Role",
    case SignedIn
        when 0 then "Logged in"
        else "Logged out"
    end as "User signed in?",
    Convert(varchar(100), LastSignOn, 101) as "Last Sign On",
    DateDiff('d', LastSignOn, getDate()) as "Days since last sign on",
    AddrLine1 + ' ' + AddrLine2 + ' ' + AddrLine3 + ' ' +
        City + ', ' + State + ' ' + Zip as "Address",
    'XXX-XX-' + Substring(
        Convert(varchar(9), SSN), 6, 4) as "Social Security #"
FROM Users
```

Normally, programmers do this because they intend to bind their dataset directly to a grid, and its just convenient to have SQL Server format server-side than format on the client.

Queries like the one shown above are extremely brittle because they tightly couple the data layer to the UI layer. On top of that, this style of programming thoroughly prevents stored procedures from being reusable.

Answer by [david-b](#)

Here are my top 3.

Number 1. Failure to specify a field list. (Edit: to prevent confusion: this is a production code rule. It doesn't apply to one-off analysis scripts - unless I'm the author.)

```
SELECT *
Insert Into blah SELECT *
```

should be

```
SELECT fieldlist
Insert Into blah (fieldlist) SELECT fieldlist
```

Number 2. Using a cursor and while loop, when a while loop with a loop variable will do.

[Skip code block](#)

```
DECLARE @LoopVar int

SET @LoopVar = (SELECT MIN(TheKey) FROM TheTable)
WHILE @LoopVar is not null
BEGIN
    --Do Stuff with current value of @LoopVar
    ...
    --Ok, done, now get the next value
    SET @LoopVar = (SELECT MIN(TheKey) FROM TheTable
                    WHERE @LoopVar < TheKey)
END
```

Number 3. DateLogic through string types.

```
--Trim the time
Convert(Convert(theDate, varchar(10), 121), datetime)
```

Should be

```
--Trim the time
DateAdd(dd, DateDiff(dd, 0, theDate), 0)
```

I've seen a recent spike of "One query is better than two, amiright?"

```
SELECT *
FROM blah
WHERE (blah.Name = @name OR @name is null)
      AND (blah.Purpose = @Purpose OR @Purpose is null)
```

This query requires two or three different execution plans depending on the values of the parameters. Only one execution plan is generated and stuck into the cache for this sql text. That plan will be used regardless of the value of the parameters. This results in intermittent poor performance. It is much better to write two queries (one query per intended execution plan).

[Answer](#)  by [annakata](#) 

- **Human readable password fields**, egad. Self explanatory.
- Using **LIKE against indexed** columns, and I'm almost tempted to just say LIKE in general.
- Recycling SQL-generated PK values.
- Surprise nobody mentioned **the god-table** yet. Nothing says "organic" like 100 columns of bit flags, large strings and integers.
- Then there's **the "I miss .ini files"** pattern: storing CSVs, pipe delimited strings or other parse required data in large text fields.
- And for MS SQL server the use of cursors *at all*. There's a better way to do any given cursor task.

Edited because there's so many!

Tags: [anti-patterns](#)

Core Data

[Skip to questions](#),

Wiki by user [peter-hosey](#) 

Core Data is Apple's object modeling and persistence framework for [Cocoa](#)  ([cocoa](#) osx ) and [Cocoa Touch](#)  ([cocoa-touch](#) ios .

While offering some functionality that one might traditionally associate with a database, Core Data, itself, is not a relational database. It is an object-oriented framework for managing, storing and retrieving model objects of a [MVC](#)  (model-view-controller) design. Core Data provides the developer a wide array of features to simplify the management of these model objects. Instead of using file management and the arbitrary requests for the given store type, you interact with Objective-C and Swift objects.

The typical file Core Data stores its data into is a SQLite file. Although Core Data supports SQLite as one of its persistent store types, Core Data cannot manage any arbitrary SQLite database. In order to use a SQLite database, Core Data must create and manage the database itself. There is also a binary store type and a memory store type.

References

- [iOS Data Management](#)  and [Mac Data Management](#) 
 - [Core Data Programming Guide](#) 
-

Questions

[Q: XCode4 and Core Data: How to enable SQL Debugging](#)

Tags: [core-data](#)

I'm working on a universal iOS app and I'd like to see the raw SQL in the logs when I'm debugging. There is some info [in this blog post](#) about how to enable raw SQL logging for iOS Core Data development. The given example is for XCode 3 and it's just not clear to me how to enable this in XCode 4.

I've tried "Product" -> "Edit Scheme" and added "**-com.apple.CoreData.SQLDebug 1**" to "Arguments Passed on Launch", but I'm still not seeing any output in the logs. Not sure if I'm looking in the wrong place or just passing the arguments incorrectly.

Tags: [core-data](#)

User: [oalders](#) 

[Answer](#)  by [nicolas-s](#) 

You should be looking at the same place you get NSLOGS

And you should Go to Product -> Edit Scheme -> Then from the left panel select Run YOURAPP.app and go to the main panel's Arguments Tab.

There you can add an Argument Passed On Launch.

You should add `-com.apple.CoreData.SQLDebug 1`

Press OK and your are all set.

The key here is to edit the scheme you will be using for testing.

Tags: [core-data](#)

Dynamic SQL

Questions

Q: Why would someone use WHERE 1=1 AND <conditions> in a SQL clause? 

Tags: [dynamic-sql](#)

Why would someone use WHERE 1=1 AND <conditions> in a SQL clause (Either SQL obtained through concatenated strings, either view definition)

I've seen somewhere that this would be used to protect against SQL Injection, but it seems very weird.

If there is injection WHERE 1 = 1 AND injected OR 1=1 would have the same result as injected OR 1=1.

Later edit: What about the usage in a view definition?

Thank you for your answers.

Still, I don't understand why would someone use this construction for defining a view, or use it inside a stored procedure.

Take this for example:

```
CREATE VIEW vTest AS  
SELECT FROM Table WHERE 1=1 AND table.Field=Value
```

Tags: [dynamic-sql](#)

User: [bogdan-maxim](#) 

[Answer](#)  by [greg-hewgill](#) 

If the list of conditions is not known at compile time and is instead built at run time, you don't have to worry about whether you have one or more than one condition. You can generate them all like:

```
and <condition>
```

and concatenate them all together. With the 1=1 at the start, the initial and has something to associate with.

I've never seen this used for any kind of injection protection, as you say it doesn't seem like it would help much. I have seen it used as an implementation convenience. The SQL query engine will end up ignoring the 1=1 so it should have no performance impact.

[Answer](#)  by [eduardo-molteni](#) 

Just adding a example code to Greg's answer:

[Skip code block](#)

```
dim sqlstmt as new StringBuilder
sqlstmt.add("SELECT * FROM Products")
sqlstmt.add(" WHERE 1=1")

' '// From now on you don't have to worry if you must
' '// append AND or WHERE because you know the WHERE is there
If ProductCategoryID <> 0 then
    sqlstmt.AppendFormat(" AND ProductCategoryID = {0}", trim(ProductCategoryID))
end if
If MinimunPrice > 0 then
    sqlstmt.AppendFormat(" AND Price >= {0}", trim(MinimunPrice))
end if
```

Tags: [dynamic-sql](#)

Cursor

[Skip to questions,](#)

Wiki by user [óscar-lópez](#) 

cursor can mean many things, and it is [recommended](#)  to use more specific tags.

- For database cursors (that let you traverse records), it is better to use [database-cursor](#) .
 - For mouse cursors (onscreen pointer to let you know where the mouse is), it is better to use [text-cursor](#) .
 - For text-cursors (blinking vertical bar to let you know where text will be input), it is better to use [caret](#) .
-

For the database concept, quoting from the wikipedia [page](#) :

In computer science and technology, a database cursor is a control structure that enables traversal over the records in a database. Cursors facilitate subsequent processing in conjunction with the traversal, such as retrieval, addition and removal of database records. The database cursor characteristic of traversal makes cursors akin to the programming language concept of iterator.

Cursors are used by database programmers to process individual rows returned by database system queries. Cursors enable manipulation of whole result sets at once—a capability that most procedural programming languages lack. In this scenario, a cursor enables the rows in a result-set to be processed sequentially.

And for the user-interface concept, again from the wikipedia [page](#) :

In computing, a cursor is an indicator used to show the position on a computer monitor or other display device that will respond to input from a text input or pointing device. The flashing text cursor may be referred to as a caret in some cases.

Questions

[Q: Why do people hate SQL cursors so much?](#)

Tags: [cursor](#)

I can understand wanting to avoid having to use a cursor due to the overhead and inconvenience, but it looks like there's some serious cursor-phobia-mania going on where people are going to great lengths to avoid having to use one.

For example, one question asked how to do something obviously trivial with a cursor and the accepted answer proposed using a common table expression (CTE) recursive query with a recursive custom function, even though this limits the number of rows that could be processed to 32 (due to recursive function call limit in sql server). This strikes me as a terrible solution for system longevity, not to mention a tremendous effort just to avoid using a simple cursor.

What is the reason for this level of insane hatred? Has some ‘noted authority’ issued a fatwa against cursors? Does some unspeakable evil lurk in the heart of cursors that corrupts the morals of children or something?

Wiki question, more interested in the answer than the rep.

Related Info:

[SQL Server Fast Forward Cursors](#)

EDIT: let me be more precise: I understand that **cursors should not be used instead of normal relational operations**; that is a no-brainer. What I don’t understand is people going waaaaay out of their way to avoid cursors like they have cooties or something, even when a cursor is a simpler and/or more efficient solution. It’s the irrational hatred that baffles me, not the obvious technical efficiencies.

Tags: [cursor](#)

User: [steven-a.-lowe](#) 

Answer  by [s.lott](#) 

The “overhead” with cursors is merely part of the API. Cursors are how parts of the RDBMS work under the hood. Often `CREATE TABLE` and `INSERT` have `SELECT` statements, and the implementation is the obvious internal cursor implementation.

Using higher-level “set-based operators” bundles the cursor results into a single result set, meaning less API back-and-forth.

Cursors predate modern languages that provide first-class collections. Old C, COBOL, Fortran, etc., had to process rows one at a time because there was no notion of “collection” that could be used widely. Java, C#, Python, etc., have first-class list structures to contain result sets.

The Slow Issue

In some circles, the relational joins are a mystery, and folks will write nested cursors rather than a simple join. I've seen truly epic nested loop operations written out as lots and lots of cursors. Defeating an RDBMS optimization. And running really slowly.

Simple SQL rewrites to replace nested cursor loops with joins and a single, flat cursor loop can make programs run in 100th the time. [They thought I was the god of optimization. All I did was replace nested loops with joins. Still used cursors.]

This confusion often leads to an indictment of cursors. However, it isn't the cursor, it's the misuse of the cursor that's the problem.

The Size Issue

For really epic result sets (i.e., dumping a table to a file), cursors are essential. The set-based operations can't materialize really large result sets as a single collection in memory.

Alternatives

I try to use an ORM layer as much as possible. But that has two purposes. First, the cursors are managed by the ORM component. Second, the SQL is separated from the application into a configuration file. It's not that the cursors are bad. It's that coding all those opens, closes and fetches is not value-add programming.

Tags: [cursor](#)

Copyright

This book was compiled from [stackoverflow](#), using the latest data dump available at [archive.org](#). A selection of the best questions about SQL Programming, as voted by the site's users have been selected for inclusion in this book.

I am not affiliated or endorsed by StackExchange Inc, although I do have a [Stack Exchange](#) user profile there under the username [George Duckett](#).

All questions and content contained within this book are licensed under [cc-wiki](#) with [attribution required](#) as per Stack Exchange Inc's requirements.

If you have or know of copyrighted content included in this book and want it to be removed please let me know at georgeduckett@hotmail.com.