## www.muppix.co explore directories [begin end last days minutes size greater]

DATABASES names & sizes of all connected hard-drives on this version of linux. TIP: goto using these harddrive names

## select lines with '*mytext*' in files [filename begin end ignore case *number* aswell *mysecondtext*]

| | |
|---|---|
| SELECT * FROM *mytable* WHERE *mycolumn* IN ('*mytext*', '*mysecondtext*','*mythirdtext*') | select lines with '*mytext*' or '*mysecondtext*' or '*mythirdtext*', ignore case, in all directories |
| SELECT t1.*,t*2*.* FROM *mytable* t1 JOIN mysecondtable t*2* ON t1.*mycolumn*=t*2*.*mycolumn* | select lines from a list of text/words in the file *mylist*.txt, (*mytext* or *mysecondtext* or ThirdText etc) all subdirectories from here on, ignore case TIP:on a Windows PC, ensure you run dos*2*unix dos*2*unix on *mylist*.txt! |

## select line with '*mytext*' [begin end before after aswell or *mysecondtext mythirdtext* word ignore case]

| | |
|---|---|
| SELECT * FROM *mytable* WHERE *mycolumn*='*mytext*' | select line with '*mytext*' ignore case. ie: could match MytEXT *mytext* or MYTEXT etc |
| SELECT * FROM *mytable* WHERE (*mycolumn* LIKE '%*mytext*%') AND (*mycolumn* LIKE '%*mysecondtext*%') | select line with both '*mytext*' aswell as '*mysecondtext*' in any order on the line |
| SELECT * FROM *mytable* WHERE (*mycolumn* = '*mytext*') AND (*mysecondcolumn* = '*mysecondtext*') | select line with '*mytext*' aswell as '*mysecondtext*' on the line (ignore case) |
| SELECT * FROM *mytable* WHERE (*mycolumn* = '*mytext*') AND (*mysecondcolumn* = '*mysecondtext*') AND (*mythirdcolumn* = '*mythirdtext*') | select line with '*mytext*' aswell as '*mysecondtext*' aswell as '*mythirdtext*' in any order (ignore case) |
| SELECT * FROM *mytable* WHERE 0 < COALESCE( NULLIF( PATINDEX( '%*mytext*%', *mycolumn*),0), NULLIF( PATINDEX( '%*mysecondtext*%' ,*mycolumn*) ,0)) | select either '*mytext*' or '*mysecondtext*' |
| SELECT * FROM *mytable* WHERE 0 < COALESCE( NULLIF( PATINDEX( '%*mytext*%' ,*mycolumn*),0) ,NULLIF( PATINDEX( '%*mysecondtext*%' ,*mycolumn*),0) ,PATINDEX( '%*mythirdtext*%' ,*mycolumn*)) | select line with '*mytext*' or '*mysecondtext*' or '*mythirdtext*', ignore case |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* IN (SELECT *mythirdcolumn* FROM mysecondtable) --select a list from mysecondtable | select any of the texts in the file *mylist*.txt '*mytext*' or '*mysecondtext*' or '*mythirdtext*' etc TIP: in Windows ensure you run dos*2*unix on *mylist*.txt, so Linux can read it |
| SELECT * FROM *mytable* WHERE *mycolumn* LIKE '*mytext*%' | select line that begin with '*mytext*' TIP:may first want to ensure there are no leading spaces |
| SELECT * FROM *mytable* WHERE *mycolumn* LIKE '*mytext*[A-D]%' | select line that begin with (range) '*mytext*A' or '*mytext*B' or '*mytext*C' or '*mytext*D' |
| SELECT * FROM *mytable* WHERE *mycolumn* LIKE '%*mytext*' | select line ending with '*mytext*' |
| SELECT * FROM *mytable* WHERE *mycolumn* LIKE '[A-D]%' | select line that begin with character 'A','B','C' or 'D' (range) |
| SELECT * FROM *mytable* WHERE *mycolumn* NOT LIKE '[A-D]%' | delete line that begin with character 'A','B','C' or 'D' (range) |
| SELECT *mycolumn* FROM *mytable* WHERE LEFT( *mycolumn*, CHARINDEX( '*mysecondtext*' , *mycolumn*)) LIKE '%*mytext*%' | select line where '*mytext*' is before '*mysecondtext*', '*mysecondtext*' after '*mytext*' |
| SELECT * FROM *mytable* WHERE SUBSTRING( *mycolumn*, LEN( '*mytext*')+ CHARINDEX( '*mytext*' , *mycolumn*) ,9999) LIKE '%*mytext*%' | select line if '*mytext*' appears atleast twice duplicated - also for a second time on each line |

## delete lines [begin end above below duplicate blanklines]

| | |
|---|---|
| TRUNCATE TABLE *mytable* | empty-out entire contents/delete all lines in .txt |
| SELECT * FROM *mytable* WHERE NOT *mycolumn* LIKE '%*mytext*%' | delete line if '*mytext*' is somewhere on the line (ignore case) TIP: first dble check which line will be deleted by running: fgrep -i '*mytext*' |
| SELECT * FROM *mytable* WHERE NOT *mycolumn* LIKE '*mytext*%' | delete lines that begin with '*mytext*' |
| SELECT * FROM *mytable* WHERE NOT *mycolumn* LIKE '%*mytext*' | delete lines that end with '*mytext*' |
| DELETE FROM *mytable* WHERE *mycolumn* = '*mytext*' OR *mycolumn*= '*mysecondtext*' | delete lines with '*mytext*' or '*mysecondtext*' |
| SELECT * FROM *mytable* WHERE NOT *mycolumn* ='*mytext*' OR *mycolumn* ='*mysecondtext*' | delete line with '*mytext*' or '*mysecondtext*' anywhere on the line (ignore case) |
| SELECT t1.*,t*2*.* FROM *mytable* t1 FULL OUTER JOIN mysecondtable t*2* ON t1.*mycolumn* =t*2*.*mycolumn* WHERE t1.*mycolumn* IS NULL | delete lines if any of the texts in the file *mylist*.txt are found, '*mytext*' etc TIP: in Windows ensure you run dos*2*unix on *mylist*.txt |
| SELECT * FROM *mytable* WHERE NOT (*mycolumn* ='*mytext*' AND *mysecondcolumn* ='*mysecondtext*') | delete line with '*mytext*' aswell '*mysecondtext*' anywhere on the line |
| SELECT *mycolumn* FROM *mytable* WHERE NOT LEFT( *mycolumn*, CHARINDEX( '*mysecondtext*' , *mycolumn*)) LIKE '%*mytext*%' | delete lines with '*mytext*' before '*mysecondtext*' ('*mysecondtext*' after '*mytext*' |
| SELECT * FROM *mytable* WHERE LTRIM(RTRIM(ISNULL(*mycolumn*,''))<>'' | truly delete all blanklines which may have some spaces or tabs or no spaces at all |
| SELECT *mycolumn* ,*mysecondcolumn* ,*mythirdcolumn* FROM *mytable* GROUP BY *mycolumn* ,*mysecondcolumn* ,*mythirdcolumn* ORDER BY *mycolumn* ,*mysecondcolumn* ,*mythirdcolumn* | sort & delete duplicate lines (dont maintain the original order & is a lot faster) |
| SELECT *mycolumn*,*mysecondcolumn* ,COUNT(*) FROM *mytable* GROUP BY *mycolumn*, *mysecondcolumn* HAVING COUNT(*)>1 | select only the duplicate lines, ie: those lines that occur twice or more |
| SELECT *mycolumn* ,*mysecondcolumn* ,*mythirdcolumn* FROM *mytable* GROUP BY *mycolumn* ,*mysecondcolumn* ,*mythirdcolumn* | delete duplicate lines, but maintain the original order ( without sorting) select begin occurrence of each line |
| SELECT *mysecondcolumn* FROM *mytable* GROUP BY *mysecondcolumn* | delete duplicate lines, based on duplicates in second column only, select begin occurrence of *2*nd column,preserve order of lines |
| SELECT * FROM (SELECT ROW_NUMBER() OVER (ORDER BY *mycolumn*) AS myrow, * FROM *mytable* ) XX WHERE myrow NOT BETWEEN *2* AND 8 | delete between second line to eigth line : (fixed) lines *2* 3 4 5 6 7 8 |
| SELECT *mycolumn*,*mysecondcolumn*, COUNT(*) FROM *mytable* GROUP BY *mycolumn* ,*mysecondcolumn* HAVING COUNT(*) >1 | how many/occurrence of duplicate lines - pivot table |

## delete '*mytext*' in the line [begin end before after between *number* second *mychar mydelimiter* word occurrence]

| | |
|---|---|
| SELECT REPLACE(*mycolumn*,'*mytext*','') FROM *mytable* | delete '*mytext*' on the line if found |
| | select everything before '*mytext*' on |

| | |
|---|---|
| SELECT LEFT( *mycolumn*, CHARINDEX( '*mytext*', *mycolumn* + '*mytext*')-1 ) FROM *mytable* | the line, (delete '*mytext*' & everything after) |
| SELECT *mycolumn*, SUBSTRING( *mycolumn*, CHARINDEX(' *mytext*',*mycolumn*) ,9999 ) FROM *mytable* | delete everything before '*mytext*' on the line, (select all text after '*mytext*') |
| SELECT REVERSE( SUBSTRING( REVERSE( ' '+ *mycolumn*) ,CHARINDEX( ' ',REVERSE( ' '+ *mycolumn*)) ,999)) FROM *mytable* | delete end word / end column |
| SELECT REVERSE( SUBSTRING( REVERSE( ','+*mycolumn*) ,CHARINDEX( ',', REVERSE( ',' +*mycolumn*)) ,999)) FROM *mytable* | delete end word / end column with comma ',' as *mydelimiter* |
| SELECT LEFT( *mycolumn*,C ASE WHEN 0 > LEN( *mycolumn*) -*2* THEN 0 ELSE LEN( *mycolumn*) -*2* END) FROM *mytable* | delete the end *2* (fixed) characters on each line. (end & second from end character) |
| SELECT SUBSTRING( *mycolumn* ,PATINDEX( '% %' ,*mycolumn* +' ') ,999) FROM *mytable* | delete beginning word / column |
| SELECT LTRIM(*mycolumn*) FROM *mytable* | left align /justify, delete beginning/leading spaces and or tabs on each line |
| SELECT RTRIM(*mycolumn*) FROM *mytable* | delete spaces or tabs at end of each line. right align. also deletes extra spaces on blanklines |
| SELECT LTRIM(RTRIM(*mycolumn*)) FROM *mytable* | delete leading/beginning space aswell as ending/trailing spaces on the line(left align, trim ) |
| SELECT CASE WHEN 0= COALESCE ( NULLIF( PATINDEX( '%*mytext*%' ,*mycolumn* ),0),0) THEN *mycolumn* ELSE '' END FROM *mytable* | delete words/columns anywhere on the line, with '*mytext*' somewhere inside/between the word ie: will delete words such as 'all*mytext*' or '*mytext*ing' or '*mytext*' |
| SELECT RTRIM(LTRIM( REPLACE(REPLACE(REPLACE( *mycolumn* ,' ','<>'),'><',''),'<>',' '))) FROM *mytable* | delete/replace all multiple/duplicate/consecutive spaces with single space/blank |
| SELECT REPLACE( REPLACE( REPLACE( *mycolumn*,'a', '') ,'b','') ,'c' ,'') FROM *mytable* | delete all occurrence of any of these 3 single (*mychar*) character 'a','b' or 'c' (ie: also delete 'abc', dan' etc) delete multiple/duplicate characters |
| CREATE FUNCTION dbo.fnStripPunct ( @Text varchar(100) ) RETURNS varchar(100) AS BEGIN DECLARE @Result varchar(100) SET @Result = '' IF @Text LIKE '%[^a-z^0-9]%' BEGIN SELECT @Result = @Result + SUBSTRING( @Text ,N.Number,1) FROM dbo.Numbers N WHERE N.Number <= LEN(@Text) AND SUBSTRING( @Text ,N.Number ,1) LIKE '[0-9a-z]' END ELSE BEGIN SET @Result = @Text END RETURN @Result END GO; SELECT dbo.fnStripPunct( *mycolumn*) from *mytable* | replace punctuation with space (delete all punctuation) |

## www.muppix.co select / delete columns [ *mytext* begin end second or delete *mychar mydelimiter* split]

| | |
|---|---|
| SELECT SUBSTRING( *mycolumn*,0 ,CHARINDEX( ' ',*mycolumn*+' ')) FROM *mytable* | select beginning column only |
| SELECT COALESCE( *mycolumn*, *mysecondcolumn* ,*mythirdcolumn*) FROM *mytable* | select beginning word/column from 3 columns |
| SELECT *mycolumn*,x.i.value('.','VARCHAR(*2*0)') AS data FROM ( SELECT *mycolumn*, CONVERT(XML,''+ *REPLACE(mycolumn ,' ','')*+*) AS data FROM *mytable* )x1 CROSS APPLY data.nodes('i[position()= *2* ]') AS x(i) | select second column |
| SELECT LTRIM( RIGHT( *mycolumn*, CHARINDEX(' ', REVERSE( ' ' +*mycolumn*)))) FROM *mytable* | select only the end column, delete all columns before the end column |
| SELECT *mycolumn*,myendcolumn FROM *mytable* | select second column and end column |
| SELECT * FROM *mytable* WHERE LEFT( *mycolumn* ,LEN( '*mytext*')) = '*mytext*' | select line if begin column is '*mytext*' |
| SELECT * FROM *mytable* WHERE right(*mycolumn*,len('*mytext*')) = '*mytext*' | select line if end column is '*mytext*' |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* = '*mytext*' | select line if second column is '*mytext*' |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* = '*mytext*' | select line if second column is '*mytext*', but column *mydelimiter* is '\|' |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* LIKE '%*mytext*%' OR *mysecondcolumn* LIKE '%*mysecondtext*%' OR *mysecondcolumn* LIKE '%*mythirdtext*%' | select whole line if '*mytext*' or '*mysecondtext*' or '*mythirdtext*' is somewhere in the second column (wildcard) |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* LIKE '*mytext*%' | select line if second column begins with '*mytext*' |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* LIKE '%*mytext*' | select line if second column ends with '*mytext*' |
| SELECT *mycolumn* ,*mysecondcolumn* ,*mythirdcolumn* FROM *mytable* WHERE (*mysecondcolumn* = '*mytext*') | select column 1,*2*,3,4 if second column is '*mytext*' |
| SELECT * FROM *mytable* WHERE SUBSTRING(*mycolumn*,*2*,6)='*mytext*' | select line if (fixed) character columns *2*-7 is '*mytext*' (from second character, for 6 characters , as length of *mytext* is 6 ) |
| DELETE * FROM *mytable* WHERE *mysecondcolumn* = '*mytext*' | delete line if second column is '*mytext*' |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* NOT LIKE '[a-zA-Z0-9] | delete lines if second column not alphanumeric (range) |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* LIKE '[A-Z]%' | select line if second column begins with uppercase (range) character (*mychar*) |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* LIKE '[A-Z]' | select line if the entire second column is all uppercase |
| SELECT * FROM *mytable* WHERE *mysecondcolumn* LIKE '[0-9]%' | select line if second column begins with '*mytext*' or begins with a *number* [0-9] |
| SELECT FROM *mytable* WHERE NOT *mysecondcolumn* LIKE '*mytext*%' | delete line if second column begins with '*mytext*' |
| SELECT * FROM *mytable* WHERE LEN(CAST(*mysecondcolumn* AS VARCHAR))>10 | if length of second column is greater 10 (or less) , select the line |
| SELECT * FROM *mytable* WHERE ISNULL(*mysecondcolumn*,'')='' | if length of second column is 0 / empty ( or less ) , select the line |
| SELECT myeigthcolumn,*mycolumn*,*mysecondcolumn* FROM *mytable* WHERE *mysecondcolumn* LIKE '%*mytext*' | select if '*mytext*' is at the end of second column and select eigth , beginning aswell second columns |
| SELECT * FROM *mytable* WHERE NOT SUBSTRING(*mycolumn*,*2*,6)='*mytext*' | delete lines if (fixed) character columns *2*-7 is '*mytext*' (from second character, for 6 characters , as length of *mytext* is 6 ) |
| SELECT REPLACE(*mycolumn*,'*mytext*','') FROM *mytable* | delete '*mytext*' if it is at the beginning of the line |
| SELECT REPLACE(myendcolumn,'*mytext*','') FROM *mytable* | delete '*mytext*' if it is at the end of line |
| SELECT * FROM *mytable* WHERE myendcolumn <>'*mytext*' | delete the whole line if '*mytext*' is at the end of line |
| SELECT TOP *2* * FROM *mytable* ORDER BY *mycolumn* ASC | select the beginning (fixed) begin and second lines (above), delete lines below second line |
| SELECT TOP *2* * FROM *mytable* ORDER BY *mycolumn* DESC | select (fixed) end line and second from end line , delete beginning/above lines. ie: tail -100 , end 100 lines TIP:useful for selecting *mytext* on live log files |

| | |
|---|---|
| SELECT * FROM (SELECT ROW_NUMBER() OVER (ORDER BY ***mycolumn***) AS myrow, * FROM ***mytable*** ) XX WHERE NOT myrow >=***2*** | select the second (fixed) lines & below , delete lines above second line |
| SELECT TOP 88 * FROM (SELECT TOP 90 * FROM ***mytable*** ORDER BY ***mycolumn*** ASC) ORDER BY ***mycolumn*** DESC -- 90=88+***2*** | select fixed line, between second line to 88th line, useful in splitting up a file |

## research: select lines with '*mytext*' and also lines above or below

| | |
|---|---|
| SELECT * FROM ***mytable*** WHERE LEN(CAST(***mycolumn*** AS VARCHAR))>***2*** | select line greater than (fixed) ***2*** characters length (second) , delete lines smaller than 1 ***2*** ( < less than) |
| SELECT TOP 1 * FROM ***mytable*** ORDER BY LEN(CAST(***mycolumn*** AS VARCHAR)) DESC | select the longest line |

## *number*s or values [greater smaller equals *number* end begin second column delete]

| | |
|---|---|
| SELECT * FROM ***mytable*** WHERE ***mycolumn*** LIKE '%[0-9]%' | select lines with a ***number*** (range) somewhere on the line |
| SELECT * FROM ***mytable*** WHERE NOT ***mycolumn*** LIKE '%[0-9]%' | delete lines with a ***number*** (range) somewhere on the line |
| SELECT * FROM ***mytable*** WHERE ***mycolumn*** > ***2***.0 | select line if begin column has a ***number***/value : is greater than ***2***.0 |
| SELECT CAST( LEFT( ***mycolumn*** +'A', PATINDEX( '%[^0-9]%' ,***mycolumn*** +'A')- 1) AS BIGINT) ,***mycolumn*** FROM ***mytable*** | select ***number***s before characters , delete characters after the ***number***s |
| SELECT REPLACE( REPLACE( REPLACE( REPLACE( REPLACE( REPLACE( REPLACE( REPLACE( ***mycolumn***,'0','') ,'1', ''), '***2***', ''), '3', ''), '4','') ,'5' ,'') ,'6',''),'7',''),'8',''),'9','') FROM ***mytable*** | delete all ***number***s/digits |
| SELECT * FROM ***mytable*** WHERE ***mysecondcolumn*** = ***2***.0 | if second column has a ***number***/value : is exactly equals to ***2***.0 ie: column could select 10.000, 10 or 010, select whole line |
| SELECT (CONVERT( decimal(15,***2***) ,REPLACE( REPLACE( REPLACE( ***mycolumn***,'(','-') ,')','') ,',' ,''))) FROM ***mytable*** | convert/clean up ***number***s in second column so can be summed up,ie negative ***number*** (1***2***3,456) |
| SELECT * FROM ***mytable*** WHERE ***mysecondcolumn*** BETWEEN ***2*** AND 10.1 | select lines if second column ***number*** /value is between ***2***.0 - 10.1 , it is greater than ***2***.0 and less than 10.1 |
| SELECT * ,***mysecondcolumn*** + ***mythirdcolumn*** AS mytotal FROM ***mytable*** | select text & insert end column - sum up/ add second & third columns |
| SELECT ***mycolumn***,***mysecondcolumn***,summed FROM (SELECT t1.***mycolumn***, t1.***mysecondcolumn***, SUM(t***2***.***mysecondcolumn***) AS summed FROM ***mytable*** t1 INNER JOIN ***mytable*** t***2*** ON t1.***mycolumn*** >= t***2***.***mycolumn*** GROUP BY t1.***mycolumn***, t1.***mysecondcolumn***) X ORDER BY summed | cumulative sum up of second column |
| SELECT SUM(***mysecondcolumn***) FROM ***mytable*** | sum second column |
| SELECT AVG(***mysecondcolumn***) FROM ***mytable*** | average of second column |
| SELECT MIN(***mysecondcolumn***) FROM ***mytable*** | line with minimum ***number*** in second column |
| SELECT MAX(***mysecondcolumn***) FROM ***mytable*** | maximum of second column |
| SELECT ***mycolumn***,ROUND( ***mysecondcolumn***,***2***) ,***mythirdcolumn*** FROM ***mytable*** | round second column to ***2*** decimal places |

## replace or convert text [*mysecondtext* beginning ignore case *mythirdtext* begin end line *mychar* duplicate space list]

| | |
|---|---|
| UPDATE ***mytable*** SET ***mysecondcolumn*** = 'mynewtext'WHERE ***mycolumn*** = '***mytext***' | replace every '***mytext***' with '***mysecondtext***' |
| SELECT REPLACE(***mycolumn***,'***mytext***','***mysecondtext***') FROM ***mytable*** | replace every '***mytext***' with '***mysecondtext***', ignore case of '***mytext***' |
| SELECT REPLACE(myendcolumn, '***mytext***','***mysecondtext***') FROM ***mytable*** | if '***mytext***' is at the end on the line , replace with '***mysecondtext***' |
| SELECT ***mycolumn***, CASE WHEN 0= CHARINDEX('***mytext***', ***mycolumn***,1) THEN ***mycolumn*** ELSE LEFT(***mycolumn*** ,CHARINDEX('***mytext***' ,***mycolumn***,1)-1) +'***mysecondtext***' END AS mynewcol FROM ***mytable*** | replace everything after '***mytext***' with '***mysecondtext***'. replacing ***mytext*** and everything after ***mytext*** |
| SELECT ISNULL(***mycolumn***,'***mytext***'),***mysecondcolumn*** FROM ***mytable*** | replace blanklines with '***mytext***'. insert '***mytext***' TIP:may need to ensure is truly blankline |
| SELECT IIF( ***mycolumn*** LIKE '***mytext***%' ,'***mysecondtext***' + SUBSTRING( ***mycolumn*** ,LEN( '***mytext***') +1 ,999), ***mycolumn***) AS mynewcolumn,* FROM ***mytable*** | if begin column is '***mytext***', replace with '***mysecondtext***' |
| UPDATE ***mytable*** SET ***mysecondcolumn*** = '***mysecondtext***' WHERE (***mysecondcolumn*** ='***mytext***') | if second column is '***mytext***', replace with '***mysecondtext***' |
| SELECT *,CASE ***mysecondcolumn*** WHEN '***mytext***' THEN 'myfourthtext' WHEN '***mysecondtext***' THEN 'myfourthtext' ELSE '' END AS mynewcolumn FROM ***mytable*** | if second column is '***mytext***' or '***mysecondtext***' or '***mythirdtext***' , replace with 'myfourthtext' |
| SELECT SUM(***mythirdcolumn***) ,mynew ,***mycolumn*** FROM( SELECT ***mycolumn*** ,***mythirdcolumn*** , CASE ***mycolumn*** WHEN '***mytext***' THEN '1' WHEN '***mysecondtext***' THEN '***2***' ELSE '3' END AS mynew FROM ***mytable***) X GROUP BY ***mycolumn*** ,mynew | make new column and then sum on the new colunm |
| UPDATE ***mytable*** SET ***mysecondcolumn*** = '***mysecondtext***', ***mythirdcolumn*** = '***mythirdtext***' WHERE (***mysecondcolumn*** ='***mytext***') | if second column is '***mytext***', replace second column with '***mysecondtext***', third column with '***mythirdtext***' |
| SELECT IIF(***mycolumn*** LIKE '%***mytext***',LEFT(m ycolumn,LEN(***mycolumn***) -LEN( '***mytext***'))+ '***mysecondtext***', ***mycolumn***) AS mynewcolumn,* FROM ***mytable*** | if end column is '***mytext***', replace with '***mysecondtext***' |
| SELECT REPLACE(***mysecondcolumn***,'***mytext***','***mysecondtext***') FROM ***mytable*** | if '***mytext***' is anywhere in second column, replace with '***mysecondtext***' ($NF if ***mytext*** is in end column) |
| SELECT *,CASE WHEN ***mycolumn*** =***mytext*** THEN '***mythirdtext***' WHEN ***mycolumn*** ='***mysecondtext*** THEN 'myfourthtext' ELSE 'myothertext' END AS mynewcolumn FROM ***mytable*** | replace or delete or insert ***mytext*** or ***mysecondtext*** (many texts) using a list of multiple/duplicate texts in a file 'myreplacelist.txt'. ie: convert/normalise/clean text |
| SELECT UCASE(***mycolumn***) FROM ***mytable*** | replace/convert lines to uppercase |
| SELECT LCASE(***mycolumn***) FROM ***mytable*** | replace/convert lines to lowercase |
| SELECT ***mycolumn***,CASE WHEN ***mycolumn*** LIKE '%***mytext***%' THEN ***mycolumn*** ELSE CASE WHEN ***mycolumn*** LIKE '%***mysecondtext***%' THEN '***mythirdtext***' ELSE ***mycolumn*** END END AS mynewcol FROM ***mytable*** | replace '***mysecondtext***' with '***mythirdtext***', but not for lines with '***mytext***' |

## insert lines / append text [begin end between before after *mysecondtext* blankline file]

| | |
|---|---|
| SELECT (CASE WHEN ***mycolumn*** LIKE '%***mytext***%' THEN '***mysecondtext***' ELSE '' END)+ ***mycolumn*** as mynewcolumn ,* FROM ***mytable*** | if '***mytext***' on line, insert word/column '***mysecondtext*** ' at beginning of line |
| INSERT INTO ***mytable***(***mycolumn***) VALUES ('***mytext***') | insert '***mytext***' below end of all lines |

## insert text on the line[*mytext* before after column blankline]

| | |
|---|---|
| SELECT ('***mytext***') AS mynewcolumn,* FROM ***mytable*** | insert '***mytext*** ' / column before beginning of the line ie: sed 's/^/ /' #indent lines |
| SELECT *,('***mytext***') AS mynewcolumn FROM ***mytable*** | insert '***mytext***' or column after the end of the line |
| SELECT REPLACE(***mycolumn***, '***mytext***','***mysecondtextmytext***') FROM ***mytable*** | insert '***mysecondtext***' before '***mytext***' |

| | |
|---|---|
| SELECT REPLACE(*mycolumn*, '*mytext*','*mytextmysecondtext*') FROM *mytable* | insert '*mysecondtext*' after '*mytext*' |
| SELECT *mysecondcolumn* + '*mytext*' FROM *mytable* | insert '*mytext*' after second column. TIP: to insert a new column use '*mytext*' |
| SELECT '*mytext*' + *mysecondcolumn* FROM *mytable* | insert '*mytext*' before second column TIP: to insert a new column use '*mytext* ' |
| SELECT REPLACE(*mysecondcolumn*,'*mytext*','*mysecondtextmytext*') FROM *mytable* | if '*mytext*' is in second column, insert '*mysecondtext*' before the second column |
| SELECT REPLACE(*mysecondcolumn*,'*mytext*','*mytextmysecondtext*') FROM *mytable* | if '*mytext*' is in second column, insert '*mysecondtext*' after the second column |
| SELECT REPLACE(*mycolumn*,'*mytext*', '*mysecondtextmytext*') FROM *mytable* | if '*mytext*' at the beginning of a line, insert '*mysecondtext*' before '*mytext*' |
| SELECT ROW_NUMBER() OVER (ORDER BY *mycolumn*) AS linenumber,* FROM *mytable* | insert line*number*s at the beginning of each line ie: find out line*number*s with '*mytext*' : cat .txt\| nl -ba \|fgrep '*mytext*' |
| SELECT * FROM (SELECT ROW_NUMBER() OVER (ORDER BY *mycolumn*) AS myrow, * FROM *mytable* ) X WHERE *mycolumn* ='*mytext*' | select lines with '*mytext*' include line*number*s (usefull for large files & can delete section of lines , from fixed line*number*s ) |
| SELECT *,ROW_NUMBER() OVER (ORDER BY *mycolumn*) AS linenumber FROM *mytable* | insert line*number*s (formated to 9 *number*s) at end |

## sort & rearrange order [sort second column delimiter split]

| | |
|---|---|
| SELECT * FROM *mytable* ORDER BY *mycolumn* ASC | sort, but ignore case , uppercase or lowercase |
| SELECT * FROM *mytable* ORDER BY *mycolumn* | sort by *number*s ie: look at beginning column as numeric values and sort TIP: if there are punctuation characters, sort may not work & delete them |
| SELECT * FROM *mytable* ORDER BY *mysecondcolumn* | sort on the second column TIP:beware of multiple spaces between columns |
| SELECT * FROM *mytable* ORDER BY *mysecondcolumn* ORDER DESC | sort on second column but in reverse order |
| SELECT *mycolumn*,*mysecondcolumn* FROM *mytable* GROUP BY *mycolumn*,*mysecondcolumn* ORDER BY *mycolumn* -- (only showing *2* columns here) | sort lines and then delete duplicate lines |
| SELECT * FROM *mytable* ORDER BY LEN(CAST(*mycolumn* AS VARCHAR)) ASC | sort the lines of a file by length, shortest on top |
| SELECT SUBSTRING(*mycolumn*,*2*,3)+ SUBSTRING(*mycolumn*,8,7) FROM *mytable* | select fixed characters *2* - 5, 8 - 15 |
| SELECT SUBSTRING(*mycolumn*,*2* ,9999) FROM *mytable* | select fixed text after the second character onwards, delete beginning *2* characters |

## convert /split / change structure of lines

| | |
|---|---|
| ;WITH mytmp( mynewlist, *mycolumn*) AS (SELECT LEFT(*mycolumn*, CHARINDEX(',' ,*mycolumn*+',')-1) ,STUFF(*mycolumn*, 1, CHARINDEX(',' ,*mycolumn*+','), '') FROM *mytable* UNION all SELECT LEFT(*mycolumn*, CHARINDEX(',' ,*mycolumn*+',')-1) ,STUFF(*mycolumn*, 1, CHARINDEX(',' ,*mycolumn*+','), '') FROM mytmp WHERE *mycolumn* > '' ) SELECT mynewlist FROM mytmp -- *mycolumn* must be VARCHAR(MAX) | replace all commas / *mydelimiter* = ',' with a newline ie: split all text with commas into a table of words/columns (structure) |
| SELECT *mycolumn*, UPPER( *mysecondcolumn*) ,*mythirdcolumn* FROM *mytable* | convert second column to uppercase , '\|' as *mydelimiter* |
| SELECT *mycolumn*, COUNT(*) AS times FROM *mytable* GROUP BY *mycolumn* | select how many occurrence , ie: pivot table |
| SELECT *mycolumn*, SUM(*mysecondcolumn*) AS mytotal FROM *mytable* GROUP BY *mycolumn* | sum/add up second column for each unique occurrence of the beginning column (like SQL GROUP BY) |
| SELECT *mycolumn*, COUNT(*) OVER (PARTITION BY *mycolumn*) AS mynewcol, COUNT(*mysecondcolumn*) OVER (PARTITION BY *mycolumn*,*mysecondcolumn*) AS mynewcol*2*, COUNT(*) OVER () AS mytotal from *mytable* | how many different types of mycolumn in mysecondcolumn, how many total mycolumn |
| SELECT *mycolumn*, MAX( *mysecondcolumn*) AS mytotal FROM *mytable* GROUP BY *mycolumn* | maximum value of second column for each unique occurrence of the beginning column (like SQL GROUP BY) |
| SELECT *mycolumn*, MAX( *mysecondcolumn*) AS mytotal FROM *mytable* GROUP BY *mycolumn* | maximum value of second column for each unique occurrence of the beginning column, include all columns (like SQL GROUP BY) |
| SELECT *mycolumn*, MIN( *mysecondcolumn*) AS mytotal FROM *mytable* GROUP BY *mycolumn* | minimum value of second column for each unique occurrence of the beginning column (like SQL GROUP BY) |
| SELECT SUM(*mysecondcolumn*) OVER (ORDER BY *mycolumn*,*mythirdcolumn*) AS mytotal ,* FROM *mytable* | sum up second column - running total |
| SELECT *mycolumn* ,COUNT(*) FROM *mytable* GROUP BY *mycolumn* | occurrence based on begin column, include *number* of entriies |
| SELECT *mycolumn*,*mysecondcolumn*,COUNT(*) FROM *mytable* GROUP BY *mycolumn*,*mysecondcolumn* | occurrence based on begin column & secondcolumn, include *number* of entries |
| SELECT *mycolumn* ,*mysecondcolumn* ,*mythirdcolumn* FROM *mytable* GROUP BY *mycolumn* ,*mysecondcolumn* ,*mythirdcolumn* | based on begin column find 1st line of each unique group |

## loop , repeat muppix commands [*mycommand* mysecondcommand]

| | |
|---|---|
| SELECT *mycolumn*, ( *mysecondcolumn*+*mythirdcolumn*+myfourthcolumn)/3 FROM *mytable* | sum/ add-up second, third & fourth column on each line - divide - average |

## save / append files [directory extension database insert]

| | |
|---|---|
| SELECT * FROM *mytable*; OUTPUT TO 'c:\\muppix*myfile*.txt' FORMAT TEXT QUOTE '"' WITH COLUMN NAMES | save results to .txt in this directory (TIP: pls note there is no "\|" with this command ) ie: ls -al >*myfile*.txt |
| SELECT * INTO #mynewtable FROM *mytable* -- #mytemptable only last during this SQL session | save results to temporary *mytable* |
| INSERT *mytable* SELECT '*mytext*','*mysecondtext*,*222* | insert results below end of .txt and save (even if it doesnt exist yet) ie: grep *mytext* * >>*myfile*.txt |
| SELECT * FROM *mytable* UNION ALL SELECT * FROM mysecondtable | insert below all lines |
| SELECT * INTO mysecondtable FROM *mytable* -- new SQL table | save as text file for viewing in notepad *.dat |
| SELECT * FROM *mytable*; OUTPUT TO 'c:\\muppix*myspreadsheet.csv*' FORMAT TEXT QUOTE '"' WITH COLUMN NAMES | save results to excell/spreadsheet or msaccess database in this directory. TIP: ensure the columns have a delimiter such as "\|" |
| SELECT * FROM *mytable* UNION ALL SELECT * FROM mysecondtable | insert all .txt lines at end/below *mysecondfile*.txt and *mysecondfile*.txt (even if *mysecondfile* doesnt exist yet) |
| SELECT t1.*,t*2*.* FROM *mytable* t1 JOIN mysecondtable t*2* ON t1.*mycolumn* = t*2*.*mycolumn* | insert *mysecondfile* after(to right of) . side by side as *2* columns |

| | |
|---|---|
| | with '\|' as ***mydelimiter*** between files |
| SELECT ***mytable***.* , mysecondtable.* FROM ***mytable*** INNER JOIN mysecondtable ON ***mytable***.***mycolumn*** = mysecondtable.***mycolumn*** | insert after columns from ***mysecondfile***, based on the begin column of each file. only select matching lines. excel VLOOKUP(A***2***,***mysecondfile***!A:B,***2***) TIP: ensure are linux files, ie: dos***2***unix ***myfile*** |
| SELECT ***mytable***.***mycolumn***, ***mytable***.***mysecondcolumn***, mysecondtable.***mythirdcolumn***,mysecondtable.myfourthcolumn FROM ***mytable*** LEFT JOIN mysecondtable ON ***mytable***.***mycolumn*** = mysecondtable.***mycolumn*** | insert after columns from ***mysecondfile***, based on the begin column of each file. include non-matching lines. excel VLOOKUP(A***2***,***mysecondfile***!A:B,***2***) TIP: ensure are linux files, ie: dos***2***unix ***myfile*** |
| SELECT ***mytable***.* , mysecondtable.* FROM ***mytable*** INNER JOIN mysecondtable ON ***mytable***.***mycolumn*** = mysecondtable.***mycolumn*** | insert after columns from ***mysecondfile***, based on the begin column of each file. include non-matching lines, ***mydelimiter*** = '\|' TIP: ensure are linux files, ie: dos***2***unix ***myfile*** |
| SELECT M.***mycolumn***,M.***mysecondcolumn***,D.mydatecolumn,D.mylastcolumn FROM ***mytable*** M CROSS APPLY( SELECT TOP ***2*** D.***mycolumn***,mydatecolumn,mylastcolumn FROM mysecondtable D WHERE M.***mycolumn*** = D.***mycolumn*** ORDER BY D.mydatecolumn DESC )D | select mycolumn and mysecondcolumn from ***mytable*** and lines of last (end) two dates for each mycolumn from mysecondtable |

## www.mupix.co linux basic navigation commands in the terminal window

| | |
|---|---|
| EXPLAIN ***mytable*** | select all filenames (& all its size/date information) in ***mydir*** & all sub directories, also select each subdirectory name with each file |
| SELECT * FROM ***mytable*** | select all lines of .txt ie: cat muppix.txt |
| SELECT TOP 10 * FROM ***mytable*** ORDER BY ***mycolumn*** ASC | begin 10 lines of file TIP: try all your commands on just these ***2***0 lines ie: cat muppix.txt \| head -***2***0 |
| SELECT TOP 10 * FROM ***mytable*** ORDER BY ***mycolumn*** DESC | end 10 lines of file TIP: use this subset of the file to try all your commands ie: cat muppix.txt \| tail |
| SELECT SUBSTRING(***mycolumn***,***2***,8-***2***) FROM ***mytable*** | delete character(s) before ***2***, select between character ***2*** (second) and 88. delete after 88th (fixed) |
| SELECT LEFT( ***mycolumn***,88) FROM ***mytable*** | only begin / less than 88 (fixed) characters of each line ie: cat .txt \| cut -c-88 |
| SELECT COUNT(*) FROM ***mytable*** | how many lines in the list / ie: how many ***mytext*** found in ***myfile***: cat ***myfile***.txt \| fgrep ***mytext*** \| wc -lc |
| SELECT RTRIM( LTRIM( REPLACE( REPLACE( ***mycolumn***,' ' ,' ' ),' ' ,' ' ))) FROM ***mytable*** | delete/replace all multiple/duplicate/consecutive spaces with single space/blank, also deletes begin spaces. easy to view ie: cat .txt \| awk -v OFS=" " '$1=$1' |
| SELECT * FROM ***mytable*** ORDER BY CONVERT(binary(***2***5),***mycolumn***) ASC | sort lines |
| SELECT * FROM (SELECT *, ROW_NUMBER () OVER (PARTITION BY ***mycolumn*** ORDER BY ***mycolumn*** ) AS myrowno | sort lines and then delete duplicate lines |

## tools to help view the output

| | |
|---|---|
| SELECT TOP ***2*** * FROM ***mytable*** ORDER BY ***mycolumn*** ASC | select beginning (fixed) and second lines |
| SELECT LEFT( ***mycolumn***,***2***) FROM ***mytable*** | only select beginning and second characters of each line. ie: cut -c-77 to quickly view text, beginning 77 characters (not wrap long lines) |
| SELECT ROW_NUMBER() OVER (ORDER BY ***mycolumn***) AS linenumber,* FROM ***mytable*** | insert line***number***s at beginning of each line ie: find out line***number***s with '***mytext***' : cat .txt\| cat -n \|fgrep '***mytext***' |

## SQL table commands

| | |
|---|---|
| SELECT * INTO mynewtable FROM ***mytable*** | copy ***mytable*** |
| SELECT * INTO mynewtable FROM ***mytable*** WHERE 1 = 0 | copy table structure based on ***mytable*** |
| SELECT ***mycolumn***, ***mysecondcolumn*** FROM (SELECT '***mytext***' AS ***mycolumn*** ,1***2*** as ***mysecondcolumn*** UNION ALL SELECT 'any thing' AS ***mycolumn*** , 44 UNION ALL SELECT '***mytext***' AS ***mycolumn*** , 11 )X | SQL sandbox, create temporary test table |
| SELECT * FROM ***mytable*** M WHERE EXISTS (SELECT * FROM mysecondtable D WHERE M.***mycolumn*** = D.***mycolumn*** ) | lines with all matching mycolumn in mysecondtable |
| SELECT * INTO #T10 FROM (SELECT 1 AS ID) X WHERE 1 = 0; ;with mytemp(n) AS ( SELECT 1 UNION ALL SELECT n + 1 FROM mytemp WHERE n < 10 ) INSERT #T10 SELECT * FROM mytemp | create table 10 lines, loop ID 1 to 10 |
| SELECT CONVERT(varchar(8), mydatecolumn, 11***2***) as mynew,* from ***mytable*** | convert date to YYYYMMDD |
| SELECT ***mycolumn***, ***mythirdcolumn***, ***mythirdcolumn*** * 100.0 / COALESCE( NULLIF( SUM(***mythirdcolumn***) OVER (PARTITION BY ***mycolumn***),0),1) AS mypcent FROM ***mytable*** orig | percent of total, grouped by mycolumn |
| SELECT ,CAST( COALESCE( NULLIF( LTRIM( RTRIM( SUBSTRING( ***mycolumn*** ,PATINDEX( '%[0-9]%',***mycolumn*** +'0' ),999))) ,'') ,'0.0') AS numeric) FROM ***mytable*** | select ***number***s out of mixed column of characters and ***number***s |
| SELECT * FROM (SELECT ROW_NUMBER() OVER( PARTITION BY ***mycolumn*** ORDER BY ***mycolumn*** DESC,***mysecondcolumn*** DESC) AS myrownum ,* FROM ***mytable*** ) X WHERE myrownum = 1 | select begin (first) line of greatest group of mycolumn & mysecondcolumn |
| WITH mytemp AS ( SELECT * , ROW_NUMBER() OVER(PARTITION BY ***mycolumn*** ORDER BY ***mycolumn*** DESC,***mysecondcolumn*** DESC) AS myrownum FROM ***mytable*** ) SELECT * FROM mytemp WHERE myrownum = 1 | select begin (first) line of greatest group of mycolumn & mysecondcolumn |
| SELECT ***mycolumn***, ***mythirdcolumn*** ,* FROM (SELECT ***mycolumn***, ***mythirdcolumn***, DENSE_RANK() OVER (ORDER BY ***mycolumn*** DESC) myrank FROM ***mytable***) X WHERE myrank <=***2*** | select lines of ***2*** most greater values grouped by mycolumn / begin top ***2*** |
| SELECT ***mycolumn***, ***mythirdcolumn*** ,* FROM (SELECT ***mycolumn***, ***mythirdcolumn***, DENSE_RANK() OVER (ORDER BY ***mycolumn*** ASC) myrank FROM ***mytable***) X WHERE myrank <=***2*** | select lines of ***2*** most smaller values grouped by mycolumn / begin top ***2*** |
| SELECT mycur, SUM(mynumbs) AS myweight FROM (SELECT DISTINCT DENSE_RANK () OVER(ORDER BY M.mydatecolumn) mywind ,M.mydatecolumn mycur ,D.mydatecolumn mylast90 ,D.***mythirdcolumn*** mynumbs FROM ***mytable*** M, ***mytable*** D WHERE d.mydatecolumn BETWEEN (M.mydatecolumn-90) AND M.mydatecolumn ) X GROUP BY mycur | moving average over last 90 days |
| SELECT ***mycolumn*** ,***mysecondcolumn*** ,SUM(***mythirdcolumn***) AS mythird FROM ***mytable*** GROUP BY ***mycolumn*** ,***mysecondcolumn*** UNION ALL SELECT NULL,NULL ,SUM (***mythirdcolumn***) FROM ***mytable*** | sum grouped by mycolumn, and insert the total below |

## terminal keystrokes in linux window

---

**Backed by years of industry experience, the Muppix Team have developed a Free Unix Data Science Toolkit to extract and analyse multi-structured information from diverse data sources**

---

**Company**

Blog

**Training**

**Professional Services**

**Get Started**