**SQL Queries for Mere Mortals**

**Fourth Edition**

Copyright © 2018 John L. Viescas.

ISBN-10: 0-134-85833-6  
ISBN-13: 978-0-134-85833-3

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the CD or programs accompanying it.

When reviewing corrections, always check the print number of your book. Corrections are made to printed books with each subsequent printing.

First Printing: December 2018

**Corrections for July 15, 2019**

| **Pg** | **Error** | **Correction** |
| --- | --- | --- |
| 133 | Example at the bottom of the page:  SQL 'The Vendor's name is:'  Displayed as The Vendor's name is: | Should read:  SQL 'The Vendor''s name is:'  Displayed as The Vendor's name is: |
| 154 | Text from top of the page:  a column called DOB and are concerned that some of your users might  not be familiar with this abbreviation. You can eliminate any possible  misinterpretation of the name by using an alias, as shown here:  SQL SELECT EmpFirstName || ' ' || EmpLastName AS  EmployeeName, DOB AS DateOfBirth  FROM Employees  This SELECT statement produces a result set with two columns called  EmployeeName and DateOfBirth. You’ve now effectively eliminated any  possible confusion of the information displayed in the result set.  Providing names for your calculated columns has a minor effect on  the translation process. For example, here’s one possible version of the  translation process for the previous example:  *“Give me a list of employee names and their dates of birth.”*  Translation Select first name and last name as employee name and  DOB as date of birth from the employees table  Clean Up Select first name ~~and~~ || ‘ ’ || last name as EmployeeName  ~~and~~ DOB as DateOfBirth from ~~the~~ employees ~~table~~  SQL SELECT EmpFirstName || ' ' || EmpLastName  AS EmployeeName, DOB AS DateOfBirth  FROM Employees | Should read:  a column called EmpDOB and are concerned that some of your users might  not be familiar with this abbreviation. You can eliminate any possible  misinterpretation of the name by using an alias, as shown here:  SQL SELECT EmpFirstName || ' ' || EmpLastName AS  EmployeeName, EmpDOB AS DateOfBirth  FROM Employees  This SELECT statement produces a result set with two columns called  EmployeeName and DateOfBirth. You’ve now effectively eliminated any  possible confusion of the information displayed in the result set.  Providing names for your calculated columns has a minor effect on  the translation process. For example, here’s one possible version of the  translation process for the previous example:  *“Give me a list of employee names and their dates of birth.”*  Translation Select first name and last name as employee name and  DOB as date of birth from the employees table  Clean Up Select first name ~~and~~ || ‘ ’ || last name as EmployeeName  ~~and~~ DOB as DateOfBirth from ~~the~~ employees ~~table~~  SQL SELECT EmpFirstName || ' ' || EmpLastName  AS EmployeeName, EmpDOB AS DateOfBirth  FROM Employees |
| 170 | Example code at top of the page:  Translation/ Select first name || ‘ ’ || ~~and~~ last name as FullName,  Clean Up BowlerAddress, city || ‘, ’ || state || ‘ ’ || ~~and~~ ZIP Code as  CityStateZip, BowlerZip from ~~the~~ bowlers ~~table and~~ order  by ZIP Code  SQL SELECT BowlerFirstName || ' ' || BowlerLastName AS  FullName,  Bowlers.BowlerAddress,  BowlerCity || ', ' || BowlerState || ' ' ||  BowlerZip AS CityStateZip, BowlerZip  FROM Bowlers  ORDER BY BowlerZip | Should read:  Translation/ Select first name || ‘ ’ || ~~and~~ last name as FullName,  Clean Up BowlerAddress, city || ‘, ’ || state || ‘ ’ || ~~and~~ ZIP Code as  CityStateZip, BowlerZip from ~~the~~ bowlers ~~table and~~ order  by ZIP Code  SQL SELECT BowlerFirstName || ' ' || BowlerLastName AS  FullName,  BowlerAddress,  BowlerCity || ', ' || BowlerState || ' ' ||  BowlerZip AS CityStateZip, BowlerZip  FROM Bowlers  ORDER BY BowlerZip |
| 173 | Third problem under Sales Orders Database:  3. *“Compile a complete list of vendor names and addresses in vendor*  *name order.”*  You can find the solution in CH05\_Vendor\_Addresses (10 rows). | Should read:  3. *“Compile a complete list of vendor names, addresses, and phone numbers in vendor name order.”*  You can find the solution in CH05\_Vendor\_Addresses (10 rows). |
| 186 | SQL at top of the page:  SQL SELECT VendName, VendPhone  FROM Vendors  WHERE VendCity <> 'Bellevue' | Should read:  SQL SELECT VendName, VendPhoneNumber  FROM Vendors  WHERE VendCity <> 'Bellevue' |
| 188 | Example in the middle of the page:  You use a “greater than or equal to” comparison for this because you want  to retrieve all hire dates from January 1, 1989, to the present, including  employees hired on that date. As you run through the translation process,  be sure to identify all the columns you need for the SELECT clause.  Translation Select first name and last name as EmployeeName from the  employees table for all employees hired since January 1, 1989  Clean Up Select first name ~~and~~ || ' ' || last name as EmployeeName  from ~~the~~ employees ~~table for all employees hired since~~ where  date hired >= ~~January 1, 1989~~ ‘1989-01-01’  SQL SELECT FirstName || ' ' || LastName  AS EmployeeName  FROM Employees  WHERE DateHired >= '1989-01-01' | Should read:  You use a “greater than or equal to” comparison for this because you want  to retrieve all hire dates from January 1, 1989, to the present, including  agents hired on that date. As you run through the translation process,  be sure to identify all the columns you need for the SELECT clause.  Translation Select first name and last name as EmployeeName from the  agents table for all agents hired since January 1, 1989  Clean Up Select first name ~~and~~ || ' ' || last name as EmployeeName  from ~~the~~ agents ~~table for all agents hired since~~ where  date hired >= ~~January 1, 1989~~ ‘1989-01-01’  SQL SELECT AgtFirstName || ' ' || AgtLastName  AS EmployeeName  FROM Agents  WHERE DateHired >= '1989-01-01' |
| 190 | Example in the middle of the page:  *“Which staff members were hired in July 1986?”*  The range condition is appropriate here because you want to retrieve the  names of everyone who was hired within a specific set of dates, in this  case, between July 1, 1986, and July 31, 1986. Let’s now run this through  the translation process and build the appropriate SELECT statement.  Translation Select first name and last name from the staff table where  the date hired is between July 1, 1986, and July 31, 1986  Clean Up Select first name and last name from the staff ~~table~~ where  ~~the~~ date hired is between ~~July 1, 1986~~ '1986-07-01' and  ~~July 31, 1986~~ '1986-07-31'  SQL SELECT FirstName, LastName  FROM Staff  WHERE DateHired  BETWEEN '1986-07-01' AND '1986-07-31' | Should read:  *“Which staff members were hired in July 1994?”*  The range condition is appropriate here because you want to retrieve the  names of everyone who was hired within a specific set of dates, in this  case, between July 1, 1994, and July 31, 1994. Let’s now run this through  the translation process and build the appropriate SELECT statement.  Translation Select first name and last name from the staff table where  the date hired is between July 1, 1994, and July 31, 1994  Clean Up Select first name ~~and~~ last name from ~~the~~ staff ~~table~~ where  ~~the~~ date hired ~~is~~ between ~~July 1, 1994~~ '1994-07-01' and  ~~July 31, 1994~~ '1994-07-31'  SQL SELECT StfFirstName, StfLastName  FROM Staff  WHERE DateHired  BETWEEN '1994-07-01' AND '1994-07-31' |
| 198 | Example in the middle of the page:  Here’s an example of how you might use the ESCAPE option:  *“Show me a list of products that have product codes beginning with*  *‘G\_00’ and ending in a single number or letter.”*  Translation Select product name and product code from the products table  where the product code begins with ‘G\_00’ and ends in a single  number or letter | Should read:  Here’s an example of how you might use the ESCAPE option:  (Note that there is no product code in any sample database; this is just an example.)  *“Show me a list of products that have product codes beginning with*  *‘G\_00’ and ending in a single number or letter.”*  Translation Select product name and product code from the products table  where the product code begins with ‘G\_00’ and ends in a single  number or letter |
| 203 | Example in the middle of the page:  *“Show me a list of all the orders we’ve taken, except for those posted in*  *October.”*  Translation Select order ID and order date from the orders table where  the order date does not fall between October 1, 2017, and  October 31, 2017  Clean Up Select order ID ~~and~~ order date from ~~the~~ orders ~~table~~  where the order date ~~does~~ not ~~fall~~ between ~~October 1, 2017,~~  '2017-10-01' and ~~October 31, 2017~~ '2017-10-31'  SQL SELECT OrderID, OrderDate  FROM Orders  WHERE OrderDate NOT BETWEEN '2017-10-01'  AND '2017-10-31' | Should read:  *“Show me a list of all the orders we’ve taken, except for those posted in*  *October.”*  Translation Select order number and order date from the orders table where  the order date does not fall between October 1, 2017, and  October 31, 2017  Clean Up Select order number ~~and~~ order date from ~~the~~ orders ~~table~~  where the order date ~~does~~ not ~~fall~~ between ~~October 1, 2017,~~  '2017-10-01' and ~~October 31, 2017~~ '2017-10-31'  SQL SELECT OrderNumber, OrderDate  FROM Orders  WHERE OrderDate NOT BETWEEN '2017-10-01'  AND '2017-10-31' |
| 205 | Example at the bottom of the page:  *“Give me the first and last names of customers who live in Seattle and*  *whose last names start with the letter ‘H’.”*  Translation Select first name and last name from the customers table  where the city is ‘Seattle’ and the last name begins with ‘H’  Clean Up Select first name ~~and~~ last name from ~~the~~ customers ~~table~~  where ~~the~~ city ~~is~~ = 'Seattle' and ~~the~~ last name ~~begins with~~  like ‘H%’  SQL SELECT CustFirstName, CustLastName  FROM Customers  WHERE CustCity = 'Seattle'  AND CustLastName LIKE 'H%' | Should read:  *“Give me the first and last names of customers who live in Redmond and*  *whose last names start with the letter ‘V’.”*  Translation Select first name and last name from the customers table  where the city is ‘Redmond’ and the last name begins with ‘V’  Clean Up Select first name ~~and~~ last name from ~~the~~ customers ~~table~~  where ~~the~~ city ~~is~~ = 'Redmond' and ~~the~~ last name ~~begins with~~  like ‘V%’  SQL SELECT CustFirstName, CustLastName  FROM Customers  WHERE CustCity = 'Redmond'  AND CustLastName LIKE 'V%' |
| 211 | Example near top of the page:  *“I need the name and title of every professor or associate professor*  *who was hired on May 16, 1989.”*  Translation Select first name, last name, title, and date hired from the staff  table where the title is ‘professor’ or ‘associate professor’ and the  date hired is May 16, 1989  Clean Up Select first name, last name, title, ~~and~~ date hired from ~~the~~ staff  ~~table~~ where the title ~~is~~ = 'professor' or title = 'associate professor'  and ~~the~~ date hired ~~is~~ = May 16, 1989 '1989-05-16'  SQL SELECT StfFirstName, StfLastName, Title, DateHired  FROM Staff  WHERE (Title = 'Professor'  OR Title = 'Associate Professor')  AND DateHired = '1989-05-16' | Should read:  *“I need the name and title of every faculty member or secretary*  *who was hired on May 16, 1989.”*  Translation Select first name, last name, title, and date hired from the staff  table where the position is ‘faculty’ or ‘secretary’ and the  date hired is May 16, 1989  Clean Up Select first name, last name, title, ~~and~~ date hired from ~~the~~ staff  ~~table~~ where the position ~~is~~ = 'faculty' or position = 'secretary'  and ~~the~~ date hired ~~is~~ = May 16, 1989 '1989-05-16'  SQL SELECT StfFirstName, StfLastName, Title, DateHired  FROM Staff  WHERE (Position = 'Faculty'  OR Title = 'Secretary')  AND DateHired = '1989-05-16' |
| 213 | Example in middle of the page:  *“Which staff members are not teachers or teacher’s aides?”*  Translation Select first name, last name, and title from the staff table where  the title is not ‘teacher’ or ‘teacher’s aide’  Clean Up Select first name, last name, ~~and~~ title from ~~the~~ staff ~~table~~ where  ~~the~~ title is not in ('teacher', ~~or~~ 'teacher"s aide')  SQL SELECT StfFirstName, StfLastName, Title  FROM Staff  WHERE NOT Title  NOT IN ('Teacher', 'Teacher''s Aide') | Should read:  *“Which staff members are not registrars or teacher’s aides?”*  Translation Select first name, last name, and title from the staff table where  the position is not ‘registrar’ or ‘teacher''s aide’  Clean Up Select first name, last name, ~~and~~ title from ~~the~~ staff ~~table~~ where  ~~the~~ position is not in ('registrar', ~~or~~ ' teacher''s aide ')  SQL SELECT StfFirstName, StfLastName, Title  FROM Staff  WHERE Title  NOT IN ('Registrar', 'Secretary') |
| 215 | SQL at bottom of the page:  SQL SELECT CustFirstName, CustLastName, CustState,  CustZipCode  FROM Orders  WHERE CustLastName = 'Patterson'  AND CustState = 'CA'  OR CustZipCode LIKE '%9' | Should read:  SQL SELECT CustFirstName, CustLastName, CustState,  CustZipCode  FROM Customers  WHERE CustLastName = 'Patterson'  AND CustState = 'CA'  OR CustZipCode LIKE '%9' |
| 223 | Example at bottom of the page:  Suppose you’re making the following request to the database:  *“Let me see the names and phone numbers of King County residents*  *whose last names are Hernandez.”*  Translation Select first name, last name, and phone number from the  customers table where the county name is ‘King’ and the  last name is ‘Hernandez’  Clean Up Select first name, last name, ~~and~~ phone number from ~~the~~  customers ~~table~~ where ~~the~~ county ~~name~~ ~~is~~ = 'King' and ~~the~~  last name ~~is~~ = 'Hernandez'  SQL SELECT CustFirstName, CustLastName,  CustPhoneNumber  FROM Customers  WHERE CustCounty = 'King'  AND CustLastName = 'Hernandez'  As you know, a row must meet both conditions to be included in the  result set. If either the county name or the last name is Null, the database  disregards the row completely. | Should read:  Suppose you’re making the following request to the database:  *“Let me see the names and phone numbers of Auburn residents*  *whose last names are Hallmark.”*  Translation Select first name, last name, and phone number from the  customers table where the city name is ‘Auburn’ and the  last name is ‘Hallmark’  Clean Up Select first name, last name, ~~and~~ phone number from ~~the~~  customers ~~table~~ where ~~the~~ city ~~name~~ ~~is~~ = 'Auburn' and ~~the~~  last name ~~is~~ = 'Hallmark'  SQL SELECT CustFirstName, CustLastName,  CustPhoneNumber  FROM Customers  WHERE CustCity = 'Auburn'  AND CustLastName = 'Hallmark'  As you know, a row must meet both conditions to be included in the  result set. If either the city name or the last name is Null, the database  disregards the row completely. |
| 224 | Entire page:  Let’s now consider this request:  *“Show me the names of all staff members who are graduate counselors*  *or were hired on September 1, 2007.”*  Translation Select last name and first name from the staff table where the  title is ‘graduate counselor’ or date hired is September 1, 2007  Clean Up Select last name ~~and~~ first name from ~~the~~ staff ~~table~~ where ~~the~~  title ~~is~~ = 'graduate counselor' or date hired ~~is~~ = ~~September~~  ~~1,2007~~ '2007-09-01'  SQL SELECT StfLastName, StfFirstName  FROM Staff  WHERE Title = 'Graduate Counselor'  OR DateHired = '2007-09-01'  Although you might expect Nulls to have the same effect on conditions  combined with OR as they do on conditions combined with AND, that  is not necessarily the case. A row still has a chance of being included in  the result set as long as it meets either of these conditions. Take a look at  Figure 6-14 again. Based on the values of Title and DateHired, Table 6-2  shows how the database determines whether to send a row to the result  set when you combine the predicates with OR.  Table 6-2 Determining the Result Set with OR   |  |  |  | | --- | --- | --- | | Value of Title | Value of DateHired | Result | | Graduate Counselor | 2007-09-01 | The row is included in the result set because it meets both conditions. | | Graduate Counselor | 2007-11-15 | The row is included in the result set because it meets the first condition. | | Registrar | 2007-09-01 | The row is included in the result set because it meets the second condition. | | Graduate Counselor | Null | The row is included in the result set because it meets the first condition. | | Null | 2007-09-01 | The row is included in the result set because it meets the second condition. | | Null | Null | The row is excluded from the result set because it does not meet either condition. | | Should read:  Let’s now consider this request:  *“Show me the names of all staff members who are graduate advisors*  *or were hired on August 2, 1993.”*  Translation Select last name and first name from the staff table where the  title is ‘graduate advisor’ or date hired is August 2, 1993  Clean Up Select last name ~~and~~ first name from ~~the~~ staff ~~table~~ where ~~the~~  position ~~is~~ = 'graduate advisor' or date hired ~~is~~ = ~~August~~  ~~2, 1993~~ '1993-08-02'  SQL SELECT StfLastName, StfFirstName  FROM Staff  WHERE Position = 'Graduate Advisor'  OR DateHired = '1993-08-02'  Although you might expect Nulls to have the same effect on conditions  combined with OR as they do on conditions combined with AND, that  is not necessarily the case. A row still has a chance of being included in  the result set as long as it meets either of these conditions. Take a look at  Figure 6-14 again. Based on the values of Position and DateHired, Table 6-2 shows how the database determines whether to send a row to the result  set when you combine the predicates with OR.  Table 6-2 Determining the Result Set with OR   |  |  |  | | --- | --- | --- | | Value of Position | Value of DateHired | Result | | Graduate Advisor | 1993-08-02 | The row is included in the result set because it meets both conditions. | | Graduate Advisor | 1994-09-05 | The row is included in the result set because it meets the first condition. | | Registrar | 1993-08-02 | The row is included in the result set because it meets the second condition. | | Graduate Advisor | Null | The row is included in the result set because it meets the first condition. | | Null | 1993-08-02 | The row is included in the result set because it meets the second condition. | | Null | Null | The row is excluded from the result set because it does not meet either condition. | |
| 225 | Text at top of page:  When you suspect that a result set is displaying incorrect information,  test any columns you’re using as criteria with the Null condition. This  will give you the opportunity to deal with any Null values as appropriate,  and you can then execute your original SELECT statement once  again. For example, if you think there might be a few graduate counselors  missing from the result set, you could execute the following SELECT  statement to determine whether this is true:  SQL SELECT StfLastName, StfFirstName, Title  FROM Staff  WHERE Title IS NULL  If there are Null values in the Title column, this SELECT statement will  produce a result set that contains the names of all staff members who  do not have a title specified in the database. Now you can deal with this  data as appropriate and then return to your original SELECT statement. | Should read:  When you suspect that a result set is displaying incorrect information,  test any columns you’re using as criteria with the Null condition. This  will give you the opportunity to deal with any Null values as appropriate,  and you can then execute your original SELECT statement once  again. For example, if you think there might be a few graduate advisors  missing from the result set, you could execute the following SELECT  statement to determine whether this is true:  SQL SELECT StfLastName, StfFirstName, Position  FROM Staff  WHERE Position IS NULL  If there are Null values in the Position column, this SELECT statement will  produce a result set that contains the names of all staff members who  do not have a position specified in the database. Now you can deal with this  data as appropriate and then return to your original SELECT statement. |

**Corrections for June 26, 2019**

|  |  |  |
| --- | --- | --- |
| **Pg** | **Error** | **Correction** |
| 184 | First paragraph:  You can also encounter unexpected results when trying to compare two character strings of unequal length, such as “John” and "John" or “Mitch” and “Mitchell.” | Should read:  You can also encounter unexpected results when trying to compare two character strings of unequal length, such as "John" and "John " or "Mitch" and “Mitchell.” (Note that the second "John " has an extra space at the end.) |

This errata sheet is intended to provide updated technical information. Spelling and grammar misprints are updated during the reprint process but are not listed on this errata sheet.