|  |
| --- |
|  |
| Queries in VLDS |
| A Simple Overview |
|  |
|  |
| **Version 1.0** |
| **8/21/2015** |

|  |
| --- |
| An introduction to the development of queries in the VLDS system with numerous examples.  |

Table of Contents

[1 Introduction 3](#_Toc427915986)

[1.1 Simple Queries 4](#_Toc427915987)

[1.1.1 AND operators 5](#_Toc427915988)

[1.1.2 OR operators 5](#_Toc427915989)

[1.2 Mixing AND and OR 5](#_Toc427915990)

[2 Queries between tables 6](#_Toc427915991)

[2.1 Tables without filters 6](#_Toc427915992)

[2.2 Tables with AND filters 6](#_Toc427915993)

[2.3 Tables with OR filters 7](#_Toc427915994)

[2.4 Mixing AND and OR Between Tables 8](#_Toc427915995)

[3 Queries between Agencies 10](#_Toc427915996)

[3.1 Introduction 10](#_Toc427915997)

[3.2 Examples of matching between agencies 10](#_Toc427915998)

[3.3 Understanding your VLDS results 12](#_Toc427915999)

[4 Revision History 13](#_Toc427916000)

# Introduction

In order to properly use VLDS one must have a basic understanding of the way queries are processed within the system. This guide is an attempt to provide a general introduction to the way the queries are handled and how to interpret the results that are being returned.

For the examples in this guide we will consider three different agencies ( Agency A, Agency B and Agency C) having the following tables. The layout and contents of these tables will be as follows:

*Agency A*

*Table A1\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *1* | *M* | *2* | *50* | *48* |
| *2* | *F* | *4* | *44* | *53* |
| *3* | *M* | *5* | *49* | *53* |
| *4* | *F* | *2* | *59* | *49* |
| *5* | *M* | *2* | *54* | *60* |

*Table A2\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *English Score* | *Reading Score* |
| *1* | *M* | *2* | *29* | *28* |
| *4* | *F* | *1* | *29* | *29* |
| *5* | *M* | *2* | *23* | *25* |
| *8* | *F* | *5* | *29* | *22* |
| *52* | *F* | *2* | *21* | *19* |

*Table A3\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Chemistry* | *Biology* | *Physics* | *Geology* |
| *1* | *3* | *3* | *3* | *5* |
| *2* | *2* | *3* | *5* | *2* |
| *3* | *3* | *5* | *3* | *4* |
| *4* | *5* | *3* | *3* | *2* |
| *5* | *4* | *4* | *4* | *4* |

Agency B

 *Table B1\_Courses*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Student ID* | *Worker ID* | *Gender* | *Institution ID* | *Dept. Num.* | *Course Num* |
| *1* | *789A* | *M* | *145* | *50* | *101* |
| *1* | *789A* | *M* | *145* | *50* | *153* |
| *1* | *789A* | *M* | *145* | *50* | *154* |
| *2* | *234B* | *F* | *120* | *12* | *102* |
| *5* | *788A* | *M* | *200* | *54* | *260* |

 *Table B2\_Status*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Student ID* | *Worker ID* | *Gender* | *Full Time* | *Housing Code* | *Tuition Code* |
| *1* | *789A* | *M* | *F* | *1* | *B* |
| *2* | *234B* | *F* | *P* | *2* | *A* |
| *5* | *788A* | *M* | *F* | *4* | *A* |
| *6* | *244B* | *F* | *F* | *1* | *C* |
| *7* | *799A* | *M* | *F* | *3* | *B* |

*Agency C*

*Table C1\_Wages*

|  |  |  |  |
| --- | --- | --- | --- |
| *Worker ID* | *Year* | *Quarter* | *Wages* |
| *789A* | *2009* | *1* | *12323* |
| *234B* | *2009* | *1* | *32341* |
| *788A* | *2009* | *1* | *4521* |
| *245B* | *2009* | *1* | *7843* |
| *790A* | *2009* | *1* | *7645* |

## Simple Queries

The simplest queries are those which operate on a single table. For these examples let us consider only *Agency A Table A1\_SCORES.*

###  AND operators

When using the AND operator, in order for a record to be included in the result set all the conditions must be met. For example if our filter on Table A1 is

*RACE=2 AND GENDER=F*

Then we would only have the record for Student 4 being returned since it was the only record which met both conditions.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *4* | *F* | *2* | *59* | *49* |

###  OR operators

When using the OR operator, in order for a record to be included in the resulting set only 1 of the conditions must be met. For example, if our filter on Table A1 is

*RACE=2 OR GENDER=F*

Then we would have the records for Students 1,2,4 and 5 since these all meet at least one of our filters.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *1* | *M* | *2* | *50* | *48* |
| *2* | *F* | *4* | *44* | *53* |
| *4* | *F* | *2* | *59* | *49* |
| *5* | *M* | *2* | *54* | *60* |

##  Mixing AND and OR

When mixing AND and ORs in a single query, the user must understand that the AND operator takes precedence over the OR operator and thus the results are dependent on the order which the conditions are listed.

Consider the query

GENDER=M OR RACE=2 AND GENDER=F

This would return the following:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *1* | *M* | *2* | *50* | *48* |
| *3* | *M* | *5* | *49* | *53* |
| *4* | *F* | *2* | *59* | *49* |
| *5* | *M* | *2* | *54* | *60* |

 Either the GENDER=M or (GENDER=F AND RACE=2) is true. Thus the conditions on either side of the AND statement are evaluated together.

Note this is totally different from the query

 GENDER=M AND RACE=2 OR GENDER=F

This would return the following:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *1* | *M* | *2* | *50* | *48* |
| *2* | *F* | *4* | *44* | *53* |
| *4* | *F* | *2* | *59* | *49* |
| *5* | *M* | *2* | *54* | *60* |

In this query note that we have grouped the (GENDER=M AND RACE=2) together due to the AND operator.

# Queries between tables

Now that we have an understanding of how a query works against a single table let’s consider what happens when we are using multiple tables within the same agency:

##  Tables without filters

We now know what happens when we put filters on a table but what happens when we select multiple tables within agency. VLDS requires that we have at least one filter on one table in an agency. But what values would be returned for a table that did not have filters?

Suppose we have selected all the columns from Table A1\_SCORES and Table A2\_SCORES.

We use the same filter we used in Section 2.1 on Table A1 :

*RACE=2 AND GENDER=F*

We know Table A1 will come back with

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *4* | *F* | *2* | *59* | *49* |

But what will be returned for Table A2\_SCORES? All the records matching that Student ID in Table A2\_SCORES, which in this case is the single record below

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *English Score* | *Reading Score* |
| *4* | *F* | *1* | *29* | *29* |

##  Tables with AND filters

We can further restrict the data being returned by adding multiple filters on the tables. For example consider the filter GENDER=M on Table A1\_SCORES and filter English Score>28 on Table A2\_SCORE.

We have three students (1,3 and 5) who have GENDER=M in Table A1\_SCORE and three students (1,4,8) who have an English Score>28 in Table A2\_SCORE. Since this is an AND condition we need to take the intersection so only Student 1 is returned in both tables.

*Table A1\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *1* | *M* | *2* | *50* | *48* |

*Table A2\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *English Score* | *Reading Score* |
| *1* | *M* | *2* | *29* | *28* |

##  Tables with OR filters

In Section 3.2 we showed how AND filters could be applied between tables. But how would the OR filter work? In our example if we change the AND to an OR : GENDER=M on Table A1\_SCORES OR filter English Score>28 on Table A2\_SCORE. As we stated earlier we have three students (1,3 and 5) who have GENDER=M in Table A1\_SCORE and three students (1,4,8) who have an English Score>28 in Table A2\_SCORE. Since we are now doing an OR we would have

*Table A1\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *1* | *M* | *2* | *50* | *48* |
| *3* | *M* | *5* | *49* | *53* |
| *4* | *F* | *2* | *59* | *49* |
| *5* | *M* | *2* | *54* | *60* |

*Table A2\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *English Score* | *Reading Score* |
| *1* | *M* | *2* | *29* | *28* |
| *4* | *F* | *1* | *29* | *29* |
| *5* | *M* | *2* | *23* | *25* |
| *8* | *F* | *5* | *29* | *22* |

Note that Student 5 appears in Table A2\_Scores since they have Gender=M in Table A1\_Scores and student 4 appears in Table A1\_Score since they have an English Score>28 in Table A2\_Score.

 ***VLDS users should be cautious in using OR between tables since this may result in a very large data sets and potential timeouts occurring.***

## Mixing AND and OR Between Tables

When mixing AND and ORs between multiple tables in a single query, the user must understand that the AND operator takes precedence over the OR operator and the results are dependent on the order which the conditions are listed.

Consider a query with filters

GENDER=M on Table A1\_SCORES AND English Score>28 on Table A2\_SCORES OR CHEMISTRY\_SCORE>3 on Table A3\_SCORES

Remember AND has precedence so we evaluate it first and we know from the example in Section 3.2 that Student 1 would be the only results of this. But now we must consider the CHEMISTRY\_SCORE>3 on Table AE\_SCORE. This brings in students 4 and 5 as well.

So our resulting tables would be :

*Table A1\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *1* | *M* | *2* | *50* | *48* |
| *4* | *F* | *2* | *59* | *49* |
| *5* | *M* | *2* | *54* | *60* |

*Table A2\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *English Score* | *Reading Score* |
| *1* | *M* | *2* | *29* | *28* |
| *4* | *F* | *1* | *29* | *29* |
| *5* | *M* | *2* | *23* | *25* |

*Table A3\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Chemistry* | *Biology* | *Physics* | *Geology* |
| *1* | *3* | *3* | *3* | *5* |
| *4* | *5* | *3* | *3* | *2* |
| *5* | *4* | *4* | *4* | *4* |

Now consider a filter with the same filters but the AND and OR changed:

GENDER=M on Table A1\_SCORES OR English Score>28 on Table A2\_SCORES AND CHEMISTRY\_SCORE>3 on Table A3\_SCORES

The AND takes precedence so we first evaluate English Score>28 on Table A2\_SCORES AND CHEMISTRY\_SCORE>3 on Table A3\_SCORES. The English>28 on Table A2\_SCORES yields students 1,4 and 8. The filter CHEMISTRY\_SCORE>3 on Table A3\_SCORES yields students 4 and 5. Since this is an AND we are left with only Student 4 meeting both criteria. We now consider the OR filter of GENDER=M on Table A1\_SCORES . This yields students 1,3 and 5. Since OR is a union we end up with students (1,3,4, and 5) in our resulting tables as shown below:

*Table A1\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *Math Score* | *Writing Score* |
| *1* | *M* | *2* | *50* | *48* |
| *3* | *M* | *5* | *49* | *53* |
| *4* | *F* | *2* | *59* | *49* |
| *5* | *M* | *2* | *54* | *60* |

*Table A2\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Gender* | *Race* | *English Score* | *Reading Score* |
| *1* | *M* | *2* | *29* | *28* |
| *4* | *F* | *1* | *29* | *29* |
| *5* | *M* | *2* | *23* | *25* |

*Table A3\_Scores*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Student ID* | *Chemistry* | *Biology* | *Physics* | *Geology* |
| *1* | *3* | *3* | *3* | *5* |
| *3* | *3* | *5* | *3* | *4* |
| *4* | *5* | *3* | *3* | *2* |
| *5* | *4* | *4* | *4* | *4* |

# Queries between Agencies

##  Introduction

Now that we have an idea what will be returned by the queries against a single agency let’s consider what happens when we go between agencies. A few concepts must be covered up front:

There are a couple of basic rules when matching between multiple agencies:

In VLDS you can’t have an OR between agencies, only ANDs.

Each agency must have at least 1 filter on it.

When creating a data package you must select a “Primary Agency”. This is the population which we will be matching the other agencies against. The data for the Primary Agency will be returned based on the rules we covered in the previous sections. The VLDS user must also select Matched or Unmatched for each query. When we select Matched for a query we are instructing VLDS to return only those records which both conform to the filter(s) placed on the agency and which also match the records being returned from the Primary Agency. When we select Unmatched we will have both unmatched and matched data being returned. When dealing with queries that have only a single agency involved this setting has no effect.

##  Examples of matching between agencies

In our examples we will use Agency B as the starting agency and match to Agency C.

***Example 1***

 Consider a request with the following tales and filters

For Agency B we want Table B1\_Courses with Gender=M

For Agency C we want to make sure we find all wages for the male students so we select Table C1\_Wages.We have to have a filter on this table as well so we put in the very broad filter Wages>0. **Note that by doing broad filters like this we run the risk of a time out by VLDS. When doing queries in VLDS it is highly recommended that filters be as restrictive as possible.**

We know that Agency B will return records for

*Table B1\_Courses*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Student ID* | *Worker ID* | *Gender* | *Institution ID* | *Dept. Num.* | *Course Num* |
| *1* | *789A* | *M* | *145* | *50* | *101* |
| *1* | *789A* | *M* | *145* | *50* | *153* |
| *1* | *789A* | *M* | *145* | *50* | *153* |
| *5* | *788A* | *M* | *200* | *54* | *260* |

Since we are doing matched we will only have records from Agency C returned for worker 789A.

*Agency C*

*Table C1\_Wages*

|  |  |  |  |
| --- | --- | --- | --- |
| *Worker ID* | *Year* | *Quarter* | *Wages* |
| *789A* | *2009* | *1* | *12323* |

***Example 2***

So what would have happened if we had selected UNMATCHED using the same filters? We would have had exactly be same set returned from out starting agency, Agency B.

*Table B1\_Courses*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Student ID* | *Worker ID* | *Gender* | *Institution ID* | *Dept. Num.* | *Course Num* |
| *1* | *789A* | *M* | *145* | *50* | *101* |
| *1* | *789A* | *M* | *145* | *50* | *153* |
| *1* | *789A* | *M* | *145* | *50* | *153* |
| *5* | *788A* | *M* | *200* | *54* | *260* |

But now all of Agency C would have been returned. **As was stated earlier, when doing unmatched queries the filters should be as restrictive as possible to avoid time outs.**

*Table C1\_Wages*

|  |  |  |  |
| --- | --- | --- | --- |
| *Worker ID* | *Year* | *Quarter* | *Wages* |
| *789A* | *2009* | *1* | *12323* |
| *234B* | *2009* | *1* | *32341* |
| *788A* | *2009* | *1* | *4521* |
| *245B* | *2009* | *1* | *7843* |
| *790A* | *2009* | *1* | *7645* |

##  Understanding your VLDS results

The results returned for the primary agency will look something this

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Unique ID | Study Group ID | Column 1 data | Column 2 data | ..Column N data |
| LBGWQQQ | 49086 | XX | YY |  NN |

The unique ID is used for matching across tables (and agencies). The Study Group ID is for internal VLDS use and not useful to the researcher.

The results from the non-primary agency will look something like the example below when Unmatched is selected

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Unique ID | Match Type | Associated ID | Column 1 data | Column 2 data | ..Column N data |
| LBGWQQQ | D | 1990596 | 6 | 13 | 12 |
| KMGXQQQ | D | 1996030 | 4 | 8 | 10 |
| U6MDQQQ | D | 1994484 | 6 | 10 | 11 |
| CV7XAQQ | P | 2029349 | 4 | 13 | 14 |
| GWUGQQQ | D | 2039602 | 6 | 15 | 18 |
| D6UDQQQ | P | 2063043 |  | 11 | 8 |
| 5KVWQQQ | D | 1963169 | 6 | 14 | 14 |
| 83R3QQQ | D | 1944029 |  | 11 | 14 |
|  |  | 2026838 |  | 12 | 16 |

The Unique ID will correspond to the Unique ID in the primary agency (when there was a match). The Match Type will be D (Deterministic) or P (Probabilistic). Since we selected Unmatched sometimes these will be blank (as shown in the last row of our example) which indicates this record didn’t have a match in the Primary Agency. The Associated ID is a unique ID that can be used to find matches between the tables in the non-primary agency.

As a final note on understanding the VLDS results, the user may at times encounter duplicate rows in a table. This is due to the fact that the user has selected only a few columns from a table and the differences lie in the fields that were not selected. For example, in Table B1\_Courses without seeing the CourseNum field, three of the rows would appear identical. If the user had selected all the columns for a given table the differences would become apparent.

VLDS operates on very large tables (sometimes involving millions of records) and a package of queries may take days to run. The users are strongly encouraged to carefully consider their data needs and apply the proper filters prior to submitting their queries for approval. If the user is unsure how to design their queries, it is strongly recommended that they consult with their agency sponsor.

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Reason for Changes** | **Date Modified** | **Modified by** |
| 0.1 | First draft | 2014.03.31 | Ron Parrish |
| 1.0 | Added Revision History Table, TOC & Removed “Draft” Watermark | 2015.08.21 | Elsie Dawson |
|  |  |  |  |
|  |  |  |  |