Paper 52-2018

Using a lookup table to transform electronic laboratory results into standardized case-specific information to manage a record-breaking flu season

Rachel Perry, MSc, MLS(ASCP)^{CM}, Laura Erhart, MPH, and Shane Brady, MPH
Arizona Department of Health Services

ABSTRACT

Using a lookup table to add new information efficiently to a primary data set is a powerful technique. This paper explores how a lookup table was utilized with base SAS® software to match laboratory reports of influenza in an Arizona infectious disease database**,** the Medical Electronic Disease Surveillance Intelligence System (MEDSIS), to a list of test and result combinations in order to assign standard categories within the database. Reports received in MEDSIS via electronic laboratory reporting (ELR) include the test name, test result, and possibly additional information in a notes field. For ease of analysis, MEDSIS contains fields for categorizing this information into standard options for type of test performed and result, but traditionally required manual data entry for each field. We created a lookup table using the information received from ELR to classify each lab report into standardized options. Reported cases within the database are matched to the lookup table routinely, allowing for an efficient method to assign the test type and result for each report. The categorized information is fed into MEDSIS to automatically populate these variables for each case record. Because reports received may vary over time or between laboratories, we need to be able to dynamically update the lookup table. This program compares the lookup table to the data set used for analysis and combinations of test names and results not found in the lookup table are exported for review. Ultimately, this process has saved hours of time by eliminating much of the manual categorization, resulting in an efficient way to update MEDSIS.

INTRODUCTION

There are many techniques used to combine multiple data sets; however, using a lookup table allows you the capability to continuously update the lookup table as more information is gathered. In Arizona, all laboratories are required to report positive test results of influenza to the state health department. Laboratories report the test name, test result, and additional information via electronic laboratory reporting (ELR) or via reports received through fax or mail. All information is stored in Arizona’s infectious disease database, the Medical Electronic Disease Surveillance Intelligence System (MEDSIS). MEDSIS contains standardized fields that users manually complete to designate the type of influenza test performed and the result. These standardized fields allow public health to efficiently analyze data on the types of tests performed and the lab results, which includes the type of influenza circulating, alerting us to any changes in the influenza season.

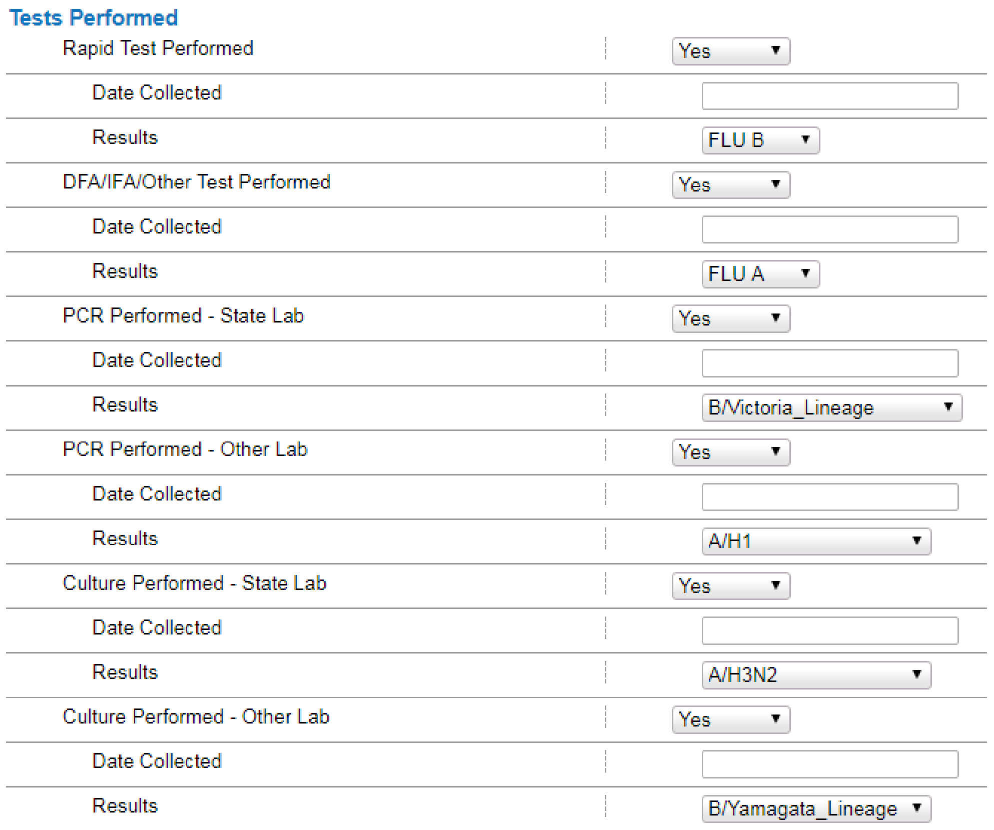
The 2017-2018 influenza season was record-breaking for Arizona, totaling over 35,000 cases compared to a 5-season average of 14,436 cases. During the last week of December 2017 alone, 5,250 positive influenza cases were reported. Staff entered all reports of positive flu into MEDSIS and manually filled out the standardized fields so that we would know what was happening in Arizona at the time. However, receiving over 3,250 cases a week for 4 weeks meant that there was a backlog of case entry. Working with our Information Technology (IT) and MEDSIS teams, we were able to create a solution by which the standardized fields for each case could be updated via an upload to MEDSIS rather than by manual entry; after the solution was implemented, all we needed to do was to manipulate the information into a format that MEDSIS could process.

In contrast to paper laboratory results that come in many formats, ELR reports are much more standardized with a laboratory generally reporting test names and results the same way each time, and with more specific information about the test performed. This standardization allowed us to create a lookup table where we could match all combinations of test names and test results received via ELR to the standardized fields within MEDSIS. Since the start of the 2015 season, approximately 33% (roughly 29,000) of all laboratory reports received for influenza are ELR, allowing us to capture a good amount of these results. Although ELR reports are standardized, there is always the potential that a laboratory might send a new test name or result combination. To catch these additions, we need to be able to update the lookup table. The program compares all combinations of test names and results used in the lookup table to the MEDSIS data set and exports all new combinations not found in the lookup table for review. If necessary, we add these new combinations to the lookup table, allowing us to have a comprehensive lookup table that captures all possibilities.

This paper will discuss the fields to consider when designing your lookup table and the methodology to determine if your lookup table captures everything you intend to match.

INFLUENZA BACKGROUND IN MEDSIS

We wanted to automatically update in MEDSIS the type of test performed and the result indicated from the ELR report. MEDSIS contains six influenza test type fields that each have a “yes” or “no” dropdown option, indicating whether the test was performed. These test types are: rapid; direct fluorescent antigen (DFA) or indirect fluorescent antigen (IFA); polymerase chain reaction (PCR) performed at the Arizona State Public Health Laboratory (ASPHL); PCR performed at another clinical laboratory; culture performed at ASPHL; and culture performed at another clinical laboratory. When we receive a report for a particular type of test, we want to mark “yes” for that test type field. Each of these test type fields has a corresponding dropdown for the test result. The dropdown options vary for each of the types of tests performed since each test type can only yield certain results. In the corresponding test result field, we want to mark the result from the laboratory report. This ranges from “Flu A” and “Flu B” to indicating the specific subtype of influenza detected, such as “A/H3”, “A/H1N1”, “B/Victoria”, “B/Yamagata”, etc. Figure 1 shows a screenshot of MEDSIS displaying the multiple test types and dropdown options (note that this is a hypothetical situation to display the multiple test types and test results).



Tests Performed

Rapid Test Performed	Yes
Date Collected	
Results	FLU B
DFA/IFA/Other Test Performed	Yes
Date Collected	
Results	FLU A
PCR Performed - State Lab	Yes
Date Collected	
Results	B/Victoria_Lineage
PCR Performed - Other Lab	Yes
Date Collected	
Results	A/H1
Culture Performed - State Lab	Yes
Date Collected	
Results	A/H3N2
Culture Performed - Other Lab	Yes
Date Collected	
Results	B/Yamagata_Lineage

Figure 1: MEDSIS Screen Showing Dropdown Fields

In the backend of MEDSIS, the test performed and test result variables each have a Q code, test code, and a free text field that corresponds to the display text in the dropdown options. Information for all three components is needed in order to automatically update each field for each case using this method. It is crucial that all three components match the laboratory report or we will not accurately update MEDSIS.

THE LOOKUP TABLE

You can use a lookup table when one data set, i.e., the lookup table, contains additional information not found in your **primary data set**. To do so, combine the lookup table to your **primary data set** by matching on a **key variable**.

Traditionally, a lookup table contains one **key variable** shared with your **primary data set**. This key variable should be unique to your lookup table, being found only once, but it can be listed many times in the primary data set. The lookup table should contain every unique key variable found in your primary data set. The lookup table also contains information stored in other variables specific to that key variable. When you combine the lookup table with your primary data set by matching on the **key variable**, the information from the lookup table unique to your key variable is added to the observations in the **primary data set** that contain the key variable. This results in a **new data set**.

LOOKUP TABLE EXAMPLE

The following example shows combining a lookup table with a primary data set by matching on a key variable. In the example below, Table 1 is the primary data set and Table 2 is the lookup table. Combining the two data sets on the key variable, state, creates the resulting new data set in Table 3.

Name	Date of Birth	City	State
Amanda	3/15/1985	Miami	Florida
Bob	6/24/1994	Dallas	Texas
Jason	9/5/1956	Rochester	New York
Stephanie	2/21/1972	Boulder	Colorado
Matthew	11/15/2005	Houston	Texas

Table 1: Primary Data Set

State	Time Zone
Florida	Eastern
Texas	Central
New York	Eastern
Colorado	Mountain
Oregon	Western

Table 2: Lookup Table

Name	Date of Birth	City	State	Time Zone
Amanda	3/15/1985	Miami	Florida	Eastern
Bob	6/24/1994	Dallas	Texas	Central
Jason	9/5/1956	Rochester	New York	Eastern
Stephanie	2/21/1972	Boulder	Colorado	Mountain
Matthew	11/15/2005	Houston	Texas	Central

Table 3: New Data Set

This example uses a one-to-many match, where the lookup table has each state listed once and the primary data set has one state listed more than once (i.e., Texas is listed for both Bob and Matthew). If the primary data set does not contain the key variable in the lookup table, then the resulting new data set will not contain the information. For example, the lookup table contains Oregon, but because the primary data set does not contain Oregon, then the new data set will not contain information pertaining to Oregon.

FIELDS TO CONSIDER ADDING TO THE LOOKUP TABLE

The lookup table should contain all variables you need to create an exact match to the key variable in your primary data set. In the example above, an exact match was on state alone, but what if your primary data set included states with more than one time zone? For example, most regions of Arizona do not observe daylight savings time, but some regions do. In this case, it would be ideal to do an exact match using the city to make your matching more specific to include the appropriate time zone. This is what we did with our lookup table. One test name could have multiple test results, each of which are mapped differently to the standardized variables in MEDSIS.

Table 4 shows an example of some of the possibilities for test results for one single test name. These test results are also not unique to this test name; "influenza A(H3)" may be a valid result for another test name.

Test Name	**Test Result**
48509-4 INFLUENZA PCR ()	260373001 DETECTED 2009 H1N1 INFLUENZA IS DETECTED
48509-4 INFLUENZA PCR ()	419984006 INCONCLUSIVE
48509-4 INFLUENZA PCR ()	446645007 INFLUENZA A(H3)
48509-4 INFLUENZA PCR ()	INFLUENZA A(H3)
48509-4 INFLUENZA PCR ()	INFLUENZA B VICTORIA
48509-4 INFLUENZA PCR ()	INFLUENZA B YAMAGATA
48509-4 INFLUENZA PCR ()	NOT DETECTED
48509-4 INFLUENZA PCR ()	PCR POSITIVE FOR INFLUENZA B VICTORIA
48509-4 INFLUENZA PCR ()	PCR POSITIVE FOR INFLUENZA B YAMAGATA
48509-4 INFLUENZA PCR ()	PLR3 INFLUENZA A (H3) PCR POSITIVE FOR INFLUENZA A H3
48509-4 INFLUENZA PCR ()	PLR65 INFLUENZA A 2009 H1N1
48509-4 INFLUENZA PCR ()	SPECIMEN FORWARDED FOR FURTHER TESTING.

Table 4: Example of One Test Name Having Multiple Test Results

Thus, accurately matching information from the lookup table to the primary data set required us to use a combination of test name and test result. Using the CATX function in SAS, we created the key variable "final":

```
data lookup_table;
set lookup_table;
final=catx(" ", test_name, test_result);
run;
```

The CATX function concatenates different items together into one variable. The first argument is the delimiter, or the character you wish to use to separate each item. The following arguments are the items, which will be separated by the delimiter in the new variable. For this example, we create a new variable, "final", which contains the test name and the test result, separated by one space. (For further information, see the Recommended Reading section).

ARE THERE OTHER VARIABLES OF INTEREST?

You also need to consider what variables are important to you. What do you need to gather from the lookup table to add to each record that matches in your primary data set? For us, it was the Q code, text code, and free text components for both the test name and the test result for each combination of test name and its corresponding results. Without this information, we would not be able to update each MEDSIS case with the appropriate information. Table 5 is the lookup table with the addition of these variables. For each test name and test result combination, we needed to add two different sets of the three variables: one set corresponding to the test performed and one set corresponding to the test result. “Q code” corresponds to the test name and identifies the type of test performed. Since each test name and test result has its own Q code, test code, and free text, the variable names in the two sets needed to be different from each other. The variables corresponding to the test *result* have a “1” on the end to signify this difference.

In Table 5 below, the “PCRS” Q code specifies that the test name is specific for a polymerase chain reaction (PCR) performed at the Arizona State Public Health Laboratory (ASPHL). The “PCRO” Q code specifies that an external laboratory performed the PCR. The first three rows contains the same Q code since all three test names correspond to PCR from ASPHL; however, each test result is different. Q code 1 is the same for these first three rows since this Q code denotes a PCR test result from ASPHL, but test code 1 and free text 1 vary to account for three different test results. Furthermore, one test code can mean different things for different Q codes. In the third row of Table 5, a test code of 20 corresponds to an A/H3 result from a PCR performed at ASPHL; however, in the last row, a test code of 20 corresponds to an influenza B result for a DFA/IFA test. For the latter example, DIOTST is the test performed Q code for a DFA/IFA and RESDIO is the test result Q code for a DFA/IFA test.

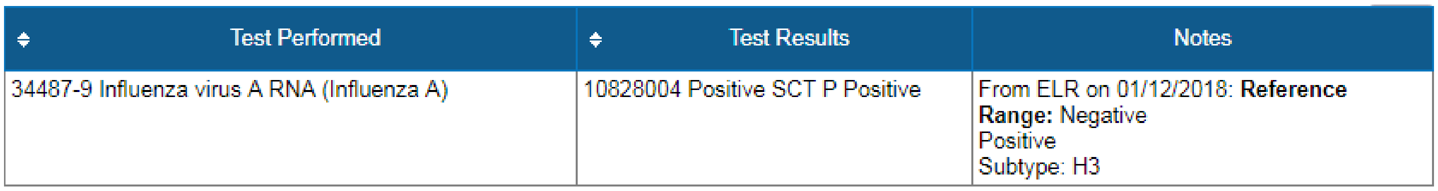
Test Name	Q code	Test code	Free text	Test Result	Q code 1	Test code 1	Free text 1
48509-4 INFLUENZA PCR ()	PCRS	30	Yes	260373001 DETECTED 2009 H1N1 INFLUENZA IS DETECTED	RESPS	15	A/H1N1 (swine-origin)
48509-4 INFLUENZA PCR ()	PCRS	30	Yes	419984006 INCONCLUSIVE	RESPS	70	Inconclusive
48509-4 INFLUENZA PCR ()	PCRS	30	Yes	446645007 INFLUENZA A(H3)	RESPS	20	A/H3
34487-9 INFLUENZA VIRUS A RNA (INFLUENZA A PCR)	PCRO	30	Yes	10828004 POSITIVE SCT P POSITIVE	RESPO	5	A
46082-4 INFLUENZA VIRUS A AG (INFLUENZA A AG, EIA)	RPTST	30	Yes	10828004 POSITIVE SCT POS POSITIVE	RESRA	10	FLU A
5867-7 INFLUENZA B-DFA ()	DIOTST	30	Yes	POSITIVE	RESDIO	20	FLU B

Table 5: Lookup Table with Additional Variables

When setting up the lookup table the first time, we manually examined all of the combinations of test name and test results to mark the Q code, test code, and free text for both the test name and the test result. For example, the test name “48509-4 Influenza PCR” is a test name only used to reflect a PCR performed at ASPHL. The Q code, test code, and free text reflect this. One of the corresponding results for this test name, “446645007 Influenza A(H3)”, indicates an A/H3 result. Likewise, the Q code, test code, and free texts correspond to an A/H3 result from ASPHL. However, sometimes the test name indicates the subtype and the test result is simply marked as “positive”. Having additional information in the test name therefore directly influences the Q code, test code, and free text variables filled out for the test result. For example, in the last row of Table 5, the test name is “5867-7 Influenza B-DFA ()” and the test result is “Positive”. The test name is indicating that the test result is positive for influenza B and the Q code, test code, and free text for the test result reflect this. Without the test name, we would not be able to appropriately analyze the test result.

WHAT ABOUT OTHER INFORMATION OF USE?

For some instances, the ELR report contains additional information on subtype found in a “lab notes” section. Figure 2 shows an example of an ELR report where the subtype is contained in the lab notes field rather than in the test results.



Test Performed	Test Results	Notes
34487-9 Influenza virus A RNA (Influenza A)	10828004 Positive SCT P Positive	From ELR on 01/12/2018: **Reference Range:** Negative Positive Subtype: H3

Figure 2: Example of Additional Information in the Lab Notes Section of MEDSIS

The test performed indicates that RNA testing (PCR) was performed for influenza A and the results indicate that it was positive, but it is in the notes section where we find that it was specifically positive for A/H3. We wanted to capture this in our code and lookup table, to ensure we had a complete picture. In order to do this, we took a multi-step approach.

We already knew that the lab notes section only provided the subtype for a minority results. To date, using combinations of the test name and the test result have always provided us with a complete picture by indicating the subtype, if available, except for a few A/H3 results. We also knew that only two test names and test result combinations included the subtype in the lab notes section (these were only for A/H3). These two pieces of information were crucial to our code.

First, we wanted a way to mark in our primary data set that the lab notes section contained additional information stating that the subtype was A/H3. We did this by using the INDEX function to create a flag variable. The INDEX function will search the designated source variable for a specific string. If the string is present within the source variable, SAS will return the value of the position the first instance the string is present. If the string is not present in the source variable, SAS will return a value of 0. Put another way, if the specified string is present within the source variable, the position will always be >0. (For further information, see the Recommended Reading section). We hardcoded the use of the INDEX function into our primary data set, so that if the specific test name and test result were used and the lab note section (the source variable) contained “SUBTYPE: H3” (the searched string), then the index function would return a value >0 and we would assign a value to “H3” to the flag variable:

```
data primary_data_set;
set primary_data_set;
   if test_name="34487-9 INFLUENZA VIRUS A RNA (INFLUENZA A)" and
      test_result in ("10828004 POSITIVE SCT P POSITIVE", "G-A200
      POSITIVE") then do;
         if INDEX(LABNOTES, "SUBTYPE: H3")>0 then flag="H3";
      end;
run;
```

Second, we created the same flag variable in our lookup table, as shown in Table 6. If the flag variable was marked as “H3” in our primary data set then it indicated to us that the lab notes section contained additional information, letting us know the subtype was A/H3. We incorporated that same flag variable into our lookup table, which allowed us to use that additional subtype information in the following step.

Test Result	Flag	Q code 1	Test code 1	Free text 1
10828004 POSITIVE SCT P POSITIVE	H3	RESPO	20	A/H3
G-A200 POSITIVE	H3	RESPO	20	A/H3

Table 6: Lookup Table with the Flag Variable

Third, we updated the key variable, "final", in our SAS code for both the lookup table and the primary data set by incorporating the "flag" variable. We used the CATX function to combine the test name, test result, and flag for both the lookup table and the primary data set. We could have created this concatenated key variable outside of SAS and have it stored in the lookup table, but we needed to create it in SAS each time for the primary data set. Creating the key variable in SAS also prevented any errors that may have occurred from combining the variables manually within the lookup table file and importing that file into SAS. Therefore, we utilized the CATX function in SAS to combine these variables for both data sets to ensure consistency between them:

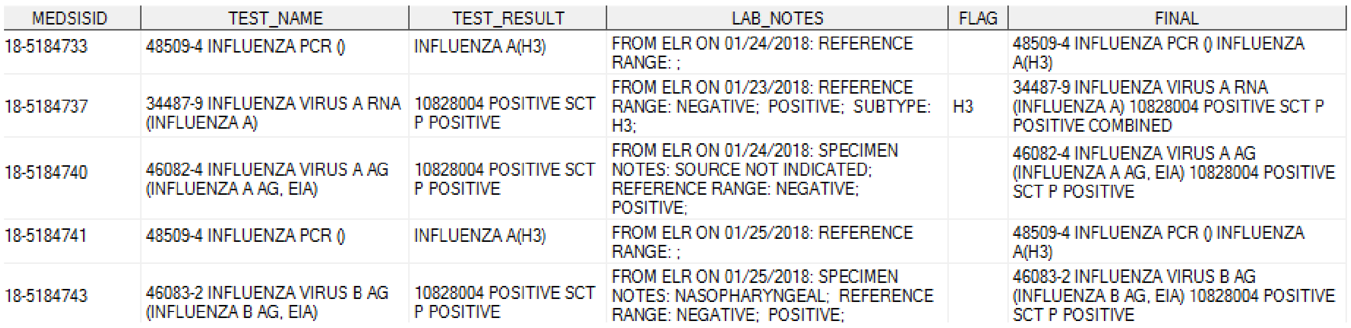
```
data primary_data_set;
set primary_data_set;
final=catx(" ", test_name, test_result, flag);
run;

data lookup_table;
set lookup_table;
final=catx(" ", test_name, test_result, flag);
run;
```

COMBINING THE LOOKUP TABLE WITH THE PRIMARY DATA SET

There are many methods described elsewhere for combining the lookup table and the primary data set. We opted for the PROC SQL procedure. Described briefly, we combined the primary data set (designated as "a") and the lookup table (designated as "b") with a left join to create a new table or data set (named here, "new_data_set"). The left join selects all observations, specified by the ON clause, that are contained in both the primary data set and the lookup table and also retains all observations from the left data set, in this case, the primary data set, that do not share the ON clause specifications with the lookup table. Our ON clause specifies that the variable "final", our key variable contained in both the primary data set (a.final) and the lookup table (b.final), match. Put another way, we are keeping all observations from the primary data set (the left data set) and pulling over all designated variables from the lookup table where the "final" variables match. In this example, our resulting data set (new_data_set) contains only certain variables selected from each of the two input data sets by using the SELECT clause. Finally, we sort our resulting data set by the variable MEDSIS ID with using the ORDER BY clause. (For further information, see the Recommended Reading section):

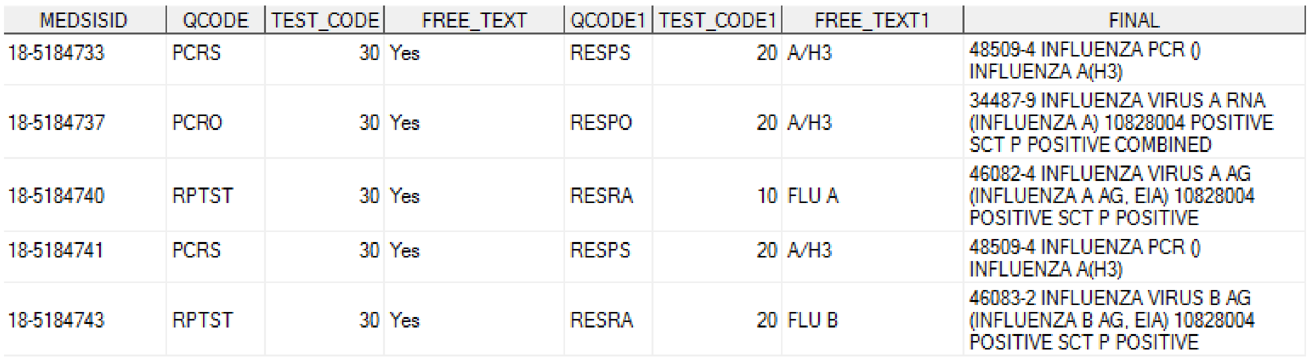
```
proc sql;
create table new_data_set as
   select a.medsisid, b.qcode, b.test_code, b.free_text, b.qcode1,
      b.test_code1, b.free_text1, b.final
   from primary_data_set a left join lookup_table b
   on (a.final=b.final)
   order by medsisid;
quit;
```

Figure 3 shows a sample of the primary data set generated from the code above and Figure 4 shows the new data set, keeping only the variables of interest.



MEDSISID	TEST_NAME	TEST_RESULT	LAB_NOTES	FLAG	FINAL
18-5184733	48509-4 INFLUENZA PCR ()	INFLUENZA A(H3)	FROM ELR ON 01/24/2018: REFERENCE RANGE: ;		48509-4 INFLUENZA PCR () INFLUENZA A(H3)
18-5184737	34487-9 INFLUENZA VIRUS A RNA (INFLUENZA A)	10828004 POSITIVE SCT P POSITIVE	FROM ELR ON 01/23/2018: REFERENCE RANGE: NEGATIVE; POSITIVE; SUBTYPE: H3;	H3	34487-9 INFLUENZA VIRUS A RNA (INFLUENZA A) 10828004 POSITIVE SCT P POSITIVE COMBINED
18-5184740	46082-4 INFLUENZA VIRUS A AG (INFLUENZA A AG, EIA)	10828004 POSITIVE SCT P POSITIVE	FROM ELR ON 01/24/2018: SPECIMEN NOTES: SOURCE NOT INDICATED; REFERENCE RANGE: NEGATIVE; POSITIVE;		46082-4 INFLUENZA VIRUS A AG (INFLUENZA A AG, EIA) 10828004 POSITIVE SCT P POSITIVE
18-5184741	48509-4 INFLUENZA PCR ()	INFLUENZA A(H3)	FROM ELR ON 01/25/2018: REFERENCE RANGE: ;		48509-4 INFLUENZA PCR () INFLUENZA A(H3)
18-5184743	46083-2 INFLUENZA VIRUS B AG (INFLUENZA B AG, EIA)	10828004 POSITIVE SCT P POSITIVE	FROM ELR ON 01/25/2018: SPECIMEN NOTES: NASOPHARYNGEAL; REFERENCE RANGE: NEGATIVE; POSITIVE;		46083-2 INFLUENZA VIRUS B AG (INFLUENZA B AG, EIA) 10828004 POSITIVE SCT P POSITIVE

Figure 3: Primary Data Set



MEDSISID	QCODE	TEST_CODE	FREE_TEXT	QCODE1	TEST_CODE1	FREE_TEXT1	FINAL
18-5184733	PCRS	30	Yes	RESPS	20	A/H3	48509-4 INFLUENZA PCR () INFLUENZA A(H3)
18-5184737	PCRO	30	Yes	RESPO	20	A/H3	34487-9 INFLUENZA VIRUS A RNA (INFLUENZA A) 10828004 POSITIVE SCT P POSITIVE COMBINED
18-5184740	RPTST	30	Yes	RESRA	10	FLU A	46082-4 INFLUENZA VIRUS A AG (INFLUENZA A AG, EIA) 10828004 POSITIVE SCT P POSITIVE
18-5184741	PCRS	30	Yes	RESPS	20	A/H3	48509-4 INFLUENZA PCR () INFLUENZA A(H3)
18-5184743	RPTST	30	Yes	RESRA	20	FLU B	46083-2 INFLUENZA VIRUS B AG (INFLUENZA B AG, EIA) 10828004 POSITIVE SCT P POSITIVE

Figure 4: New Data Set

HOW TO DETERMINE IF YOUR LOOKUP TABLE IS UP-TO-DATE

Since the ultimate goal of the lookup table is to add information to your primary data set, you want to make sure that all combinations of what you are searching for in your primary data set exist in your lookup table. Stated another way, all options for the key variable in your primary data set match must also be in your lookup table. This was crucial for us because new influenza lab reports continue to be reported and we needed a way to identify any new test names, test results, or combinations of these in order to include them in the lookup table.

The code below is used to identify whether we have all combinations of test name and test results, or any instances of our key variable, that are not in the lookup table. The PROC FREQ step outputs all unique combinations contained in the primary data set. The flag variable created earlier is not necessary for this step since we only want to ensure we are capturing all unique combinations of test name and test result. We created a new key variable here, “match”, from the primary data set and from the lookup table. The match variable represents all combinations of test name and test result from each data set. The primary data set and lookup table are sorted by the match variable so that the two data sets could be merged via the MERGE function. In this merge, we are keeping the values of “match” found in the primary data set but *not* found in the lookup table (using the “if a and not b” line in the last step) since these represent combinations of test name and test result not captured in the lookup table:

```
proc freq data= primary_data_set;
tables test_name*test_result / out=ELR_tests;
where ELR="YES";
run;
```

```
data ELR_tests (keep=test_name test_result match);
set ELR_tests;
match=catx("", test_name, test_result);
run;

data old_tests (keep=test_name test_result match);
set lookup_table;
match=catx("", test_name, test_result);
run;

proc sort data=ELR_tests; by match; run;
proc sort data=old_tests; by match; run;

data new_tests;
merge ELR_tests (in=a) old_tests (in=b);
by match;
if a and not b;
run;
```

We then export any new combinations identified in “new_tests”, review them, add the appropriate Q code, test code and free text for each new combination, and add these to the lookup table. Afterwards we re-run with the updated lookup table. This process occurs whenever the program is run, allowing us to continuously update the lookup table to reflect the current use of test names and test results.

CONCLUSION

Lookup tables are a powerful technique used to gather data from one data set and add it to another data set. If the key variable is present in both the lookup table and the primary data set, then the information from the lookup table will be added to those observations in the primary data set. It does not matter if the lookup table contains records that are not present in the primary data set – they will be ignored. However, the value of a dynamic lookup table is when there are observations in your primary data set that do not match the lookup table. It is critical that you have code written to determine if this is the case, if you work with data sets that are always changing.

Being able to use a dynamic lookup table to match to our primary data set has saved public health officials hours of time. This process has eliminated the need to manually enter data into standardized fields for each influenza case with an ELR report received, allowing us to manage a record-breaking influenza season.

ACKNOWLEDGMENTS

Thank you to the MEDSIS team for adding the ability to batch update cases in such a timely manner.

RECOMMENDED READING

- Croonen, N. and Theuwissen, H. 2002. “Table Lookup: Techniques Beyond the Obvious.” http://www2.sas.com/proceedings/sugi27/p011-27.pdf.
- Lafler, K.P., 2009. “Exploring PROC SQL Joins and Join Algorithms.” https://support.sas.com/resources/papers/proceedings09/035-2009.pdf
- Yang, W., Chen, F., Zhang, L., and Hu, W. 2010. “Table Lookup in SAS.” https://www.lexjansen.com/nesug/nesug10/cc/cc37.pdf

- SAS Support. “CATX Function.” http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a002257076.htm
- SAS Support. “INDEX Function.” http://support.sas.com/documentation/cdl/en/lrdict/64316/HTML/default/viewer.htm#a000212242.htm

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Rachel Perry
Arizona Department of Health Services
602-364-4068
Rachel.Perry@azdhs.gov
azhealth.gov

Laura Erhart
Arizona Department of Health Services
602-319-9397
Laura.Erhart@azdhs.gov
azhealth.gov

Shane Brady
Arizona Department of Health Services
602-364-3147
Shane.Brady@azdhs.gov
azhealth.gov