Analysis of the Capabilities of the Virginia Longitudinal Data System (VLDS) to support Baseline Distinct Counts of Select Data Sets to support the Establishment of a Virginia Early Childhood Integrated Data System (ECIDS)

Principal Investigator
Aaron D. Schroeder, Associate Research Professor, Social and Decision Analytics, Biocomplexity Institute, University of Virginia

Research Assistant
Devika Nair, Research Scientist, Social and Decision Analytics, Biocomplexity Institute, University of Virginia

SUMMARY

The goal of this project was to ascertain the capabilities of the Virginia Longitudinal Data System (VLDS) to support the integration of data to generate a distinct count of children birth to five served by one or more early childhood programs and/or services, as a foundational metric for a range of future early childhood policy and programmatic analyses and uses.

The established **project objectives** to achieve this goal were to:

1) produce a data fitness analysis of assessed data sets
2) produce analyses and presentation of basic demographic breakdowns over time for multiple combinations of the selected early childhood service data sets, and
3) produce an initial composite index, for demonstration purposes, using the produced distinct counts.

All Services, Locations, Races, Ethnicity and Birth Year per Child 2013-2016

Datasets: DSS Customers by Year, DSS SNAP Customers by Year, DSS TANF, Customers by Year, DSS Foster Customers by Year, OCS Services by Year, DOE Student Records

unique_id	year	tanf	fips_tanf	snap	fips_snap	foster	fips_foster	ocs	part_b	wht	blk	asn	ind	isl	oth	ethn	brthyr	gndr
803970010374	2013	N	NA	Y	155	N	NA	N	N	N	N	N	N	N	N	0	2010	1
803970012961	2014	Y	53	Y	53	N	NA	N	N	Y	N	N	N	N	N	0	2014	2
803970015086	2013	Y	67	Y	67	N	NA	N	N	Y	N	N	N	N	N	2	2009	1
803970017236	2013	N	770	N	770	Y	770	N	N	Y	N	N	N	N	N	2	2009	2
803970030654	2013	N	187	Y	157	N	157	N	N	N	N	N	N	N	N	2	2012	2
803970043945	2013	N	NA	N	NA	Y	810	N	N	Y	N	N	N	N	N	2	2008	1
803970050281	2014	N	NA	Y	740	Y	740	N	N	N	Y	N	N	N	N	0	2011	2
803970058949	2013	N	NA	Y	840	N	NA	N	N	N	Y	N	N	N	N	2	2010	2
803970074611	2013	N	NA	Y	63	N	NA	N	N	Y	N	N	N	N	N	2	2010	1
803970103982	2013	Y	163	Y	163	N	NA	N	N	Y	N	N	N	N	N	2	2007	2
803970116217	2013	N	NA	Y	191	N	191	N	N	Y	N	N	N	N	N	2	2009	1
803970124890	2013	N	NA	Y	730	Y	730	N	N	N	Y	N	N	N	N	2	2010	2
803970128311	2013	Y	143	N	590	N	NA	N	N	N	Y	N	N	N	N	2	2007	1
803970152451	2013	Y	13	Y	13	Y	13	N	N	N	Y	N	N	N	N	2	2010	1
803970172247	2013	N	NA	Y	3	N	NA	N	N	Y	N	N	N	N	N	1	2007	2

Figure 1 An Example of Unique-Count Cross-Dataset Linkage Possible using the VLDS

All three objectives were achieved and, as Figure 1 demonstrates, successful integration and analysis of data relevant to the target population is achievable. In addition, a custom algorithm for the R programming language was developed to facilitate quick deduplication of these datasets by others in the future (see Appendix B).

However, significant hurdles remain to successfully utilize the VLDS as a basis for a future Virginia Early Childhood Integrated Data System (ECIDS). The two primary hurdles are 1) that the VLDS does not currently collect all of the needed data for the target population (this was already well known going into the study), and 2) the time and effort required to successfully query, process, and re-query the data system, as is necessary in any investigative process, be it for policy or research purposes, is significantly onerous. While the people administering the system are knowledgeable and competent, the time and steps necessary to determine which data is required as well as the time it takes for a query to make it through a queue with a continuous backlog, suggest an under-resourcing of the technical infrastructure necessary to support its increased use as a system to support early childhood policy and program analyses.

BACKGROUND

To understand how policies, services, and supports work for which children at what time, policymakers need comprehensive data about the accessibility, quality, and effectiveness of services. The potential value of integrated administrative data systems (IDS) to provide this crucial policy-relevant data is increasing (Fantuzzo & Culhane, 2016). This is particularly relevant for the establishment and continuous evaluation of public programs focused on young children ages 0-5, a group for whom services are historically fragmented and disconnected from systems serving school-aged children, and siloed among health, human services, and education agencies.

Accordingly, the Virginia Early Childhood Foundation (VECF) is investigating the potential for establishing an Early Childhood Integrated Data System (ECIDS) to collect, store, integrate, and maintain data from early childhood programs across multiple agencies within the state. The combined information from a Virginia ECIDS can be used to inform service delivery, public policy, and future investments to ensure all children have access to the supports they need to succeed in school and in life.

In developing a Virginia ECIDS, a first necessary step is to ascertain what aspects of an ECIDS may be provided by already established and supported data systems. The goal of this study is **to determine the current ability of the Virginia Longitudinal Data system (VLDS) to support the generation of a distinct count of children, birth to five, served by one or more early childhood programs and/or services, as a foundational metric for a range of future early childhood policy and programmatic analyses and uses**.

Funded by the 2009 Statewide Longitudinal Data Systems Grant Program of the United States Department of Education, the Virginia Longitudinal Data system (VLDS) was established to provide a cost-effective mechanism for extracting, shaping and analyzing partner agency data in an environment that ensures the highest levels of privacy. State agencies currently participating in the VLDS include the Virginia Department of Education (VDOE), the State Council of Higher Education for Virginia (SCHEV), the Virginia Employment Commission (VEC), the Virginia Department of Social Services (VDSS), the Virginia Community College System

(VCCS), the Virginia Department for Aging and Rehabilitative Services (DARS), and Virginia Department of Health Professions (DHP).

PROJECT TASKS TO COMPLETE OBJECTIVES

The tasks completed to achieve the project objectives were:

1) Work with VECF and partners to derive most desired/useful counts to be provided
2) Work with VLDS contributors to determine most appropriate data sets
   a) Understand data processing procedures, and attendant issues, used by data providers to create the unique demographic log required by the VLDS
   b) Select the optimal combination of data sets balancing potentially desired/useful counts with an understanding of the data quality and fitness
   c) Profile the fitness (quality, structure, metadata, duplication, etc) of the data sets as provided by the VLDS to provide the necessary demographics and other requested measures for analysis
   d) Secure access to data provider records and conduct deterministic and probabilistic de-identification analyses to establish rough error estimates for each demographic log to be used in the project
3) Query Construction - Derive and verify queries for selecting distinct counts
   a) Distinct counts by race, gender, age, economic status, and location for each selected EC services over multiple time periods (e.g. months, years)
   b) Distinct counts for each combination of the selected EC services over multiple time periods which may encompass difference subsets of both children and programs
   c) Execute queries and verify counts produced vs counts expected given previous analysis of rough errors existing in the provided demographic logs

Initial Dataset Selection and Filtering

Initial dataset and field selection were guided by the objective of finding the best available demographic information to specify who, what, when, and where of service receipt across agencies. The following table shows the datasets and specific demographic fields under consideration with Information on children 0-5 years of age.

Source		Demographics				Location		Time
Agency	Dataset	Race	Ethnicity	Gender	Age	Person	Service	Service
VDSS	Customers by Year	customer_race_is_white_indicator customer_race_is_black_indicator customer_race_is_asisan_indicator cust_race_is_amer_indian_alaska_native_ind cust_race_is_hawaiian_pacific_islander_ind customer_race_is_other_indicator	ethnicity_code	gender_code	age_class_code age_group_code age_type_code month_of_birth year_of_birth			calendar_year_number service_year
VDSS	SNAP Customers by Year					zip_code	county_fips_code	calendar_year_number
VDSS	TANF Customers by Year					zip_code	county_fips_code	calendar_year_number
VDSS	Foster Customers by Year						county_fips_code	calendar_year_number
VOCS	OCS Services by Year							service_begin_date service_end_date service_duration program_year
VDOE	Student Records	race_type	ethnicity_flag	gender	birth_month birth_year grade_code			school_year entry_date

Figure 2 Combining and Deduplicating Across Sources
Datasets and field under consideration for demographic information

SPECIFIED DATA FILTERS

The datasets were queried using the following filters to narrow the selections to the target population.

- DOE|Unique Students Listing|School Year greater than 2012
- And DOE|Unique Students Listing|Grade Code in list JK,KA,KG,KP,PK,T1,TT,UG
- And DOE|VPI+|Birth Year greater than 2007
- And OCS|OCS Services By Year|Program Year greater than 2012
- And DSS|DSS Customers By Year|Calender Year number greater than 2012

ISSUES EXPERIENCED IN ACQUIRING DATASETS

- Working with the VLDS interface for data selection and extraction is very onerous.
- Although the datasets are previously linked, it is not possible to limit a search by criteria on one of the datasets only. Each dataset must have criteria set. However, each dataset does not have the same available criteria.
- Additionally, the system only allows for viewing of sample data responses from the first dataset with criteria set. After setting additional criteria on the additional datasets, the system responds that No Records are found. This is not actually the case and you do not know how many records will result until the system actually returns a data package in a day or two.
- To get to a useable set of data tables to begin to answer a research question, a guessing game has to be pursued with many days wait in-between guesses.
- Additionally, if the query has been too broadly defined (too many records have been requested), an error will be returned and the researcher will need to begin attempting to reduce the size of the query, which is a guessing game in and of itself.

COMPLETED PROJECT OBJECTIVES

Produce data fitness analysis of assessed data sets

We have determined that while there are some issues in demographic quality over time in the VDOE, VDSS and OCS data, the issues are small and would not detract from the ability to link the data and have the linked data be used to analyze demographic patterns of children over time. Also, because much of the data that will be coming into the system for kids 0-5 is going to be newly generated data, it is expected that the quality of these collections will be generally high, certainly no lower.

An example of the data profiling conducted to assess fitness for use is shown here for the dataset "DSS Customers by Year" provided by VDSS to the VLDS. Additional data profiling results can be seen in Appendix A.

A general note of DSS dataset structures encountered

For those working with these datasets in the future, it is important to note that DSS "Individual by Year" datasets are actually "Individual by Year by Location".

For example, SNAP and TANF Customer Records by Year are actually Customer Records by Year AND by "Location", so multiple records per customer occur if a customer received benefits in more than one FIPS code or zip code in a calendar year. As a single record is needed per customer per year for linkage purposes, additional columns must be created to account for all possible locations. The number of columns added is based on the customer with the highest number of locations in a single year. In this case it is six, but the code automatically determines the number. An example of the resulting location record individuals can be seen below.

DATASET PROFILE: DSS CUSTOMERS BY YEAR

Dataset Preparation

Provided datasets are often vastly different from each other in terms of both schema and structure. To prepare for data profiling, dataset fields are checked for spelling errors and converted to a standardized format. If the dataset does not provide records at the level of aggregation required (e.g. each row is unique for a person and year) then the dataset is restructured.

Task: Preparation of Field/Column Names

Field/Column names standardized.

fields_original	fields_prepared
Unique ID	unique_id
Age Class Code	age_class_code

Age Group Code	age_group_code
Age Type Code	age_type_code
Calender Year number	calendar_year_number
Customer race is Black indicator	customer_race_is_black_indicator
Customer race is Asisan indicator	customer_race_is_asisan_indicator
Cust race is Hawaiian/Pacific Islander ind	cust_race_is_hawaiian_pacific_islander_ind
Cust race is Amer Indian/Alaska Native Ind	cust_race_is_amer_indian_alaska_native_ind
Customer race is White indicator	customer_race_is_white_indicator
Customer race is Other indicator	customer_race_is_other_indicator
Foster Care case Indicator	foster_care_case_indicator
Ethnicity Code	ethnicity_code
Gender Code	gender_code
Month of Birth	month_of_birth
SNAP Case Indicator	snap_case_indicator
TANF Case indicator	tanf_case_indicator
Year of Birth	year_of_birth

Task: Restructuring of Dataset to Required Level of Aggregation

No restructuring was required for this dataset.

Uniqueness

The concept of data uniqueness can be generalized as the number of unique valid values that have been entered in a record field, or as a combination of record field values within a dataset. Uniqueness is not generally discussed in terms of data quality, but for the purposes of answering research questions, the variety and richness of the data is of paramount importance. Most notably, if a record field has very little value uniqueness (e.g. entries in the field 'State' for an analysis of housing within a county, which of course would be within a single state), then its utility would be quite low and can be conceptualized as having low quality in terms of the research question at hand.

Test: Numerical Frequencies

There were no numerical items in this dataset

Test: Categorical Frequencies

Assessing the breakdown of the frequency of categorical variables can be very informative when selecting appropriate fields for linkage and/or analysis. For example, as can be seen in Figure 2, when selecting a field to best serve as the "age" variable from this dataset it becomes instantly clear that 'age_class_code' and 'age_type_code' are not appropriate for this use as they only contain one ("1") and two values ("1", "2"), respectively. The field 'age_group_code', however, appears much more suitable as it is comprised of a distinct value for each year of age.

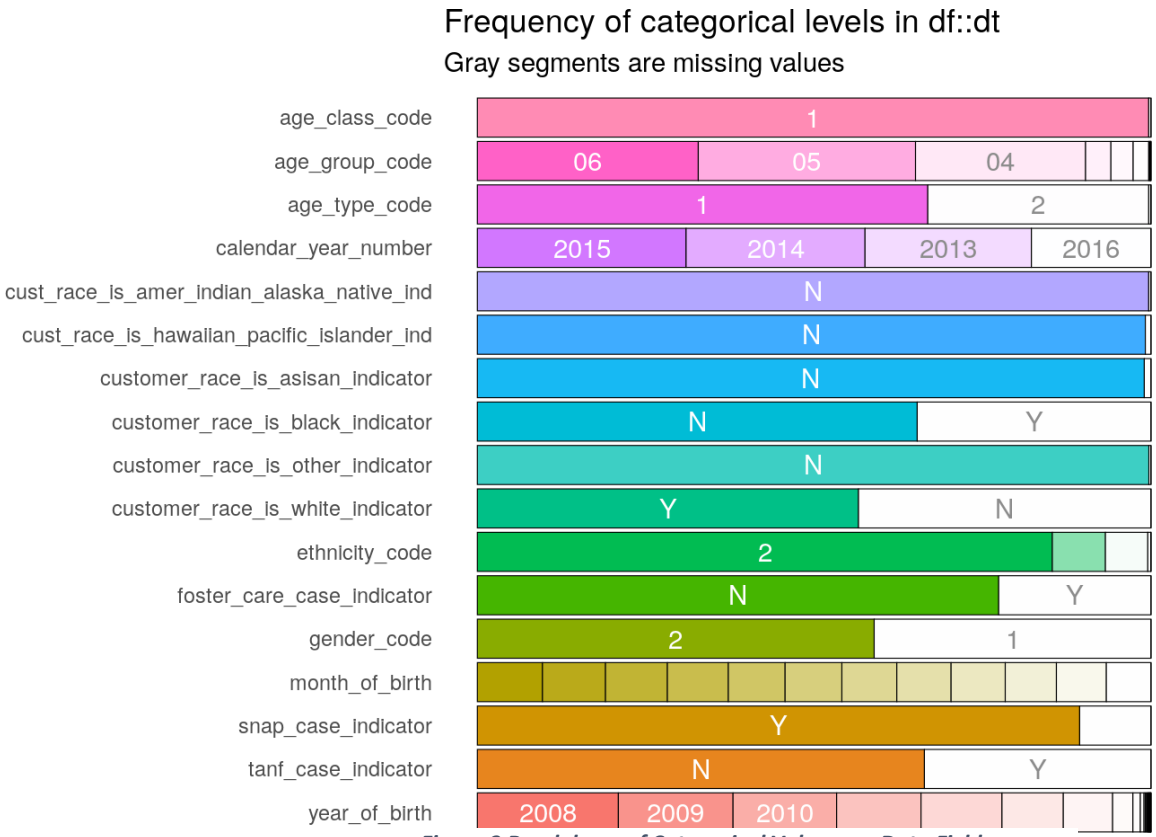


Figure 3 Breakdown of Categorical Values per Data Field

Completeness

The concept of data completeness can be generalized as the proportion of data provided versus the proportion of data required. Data that is missing may additionally be categorized as record fields not containing data, records not containing necessary fields, or datasets not containing the requisite records. The most common conceptualization of completeness is the first, record field not containing data. This conceptualization of data completeness can be thought of as the proportion of the data that has values to the proportion of data that 'should' have values. That is, a set of data is complete with respect to a given purpose if the set contains all the relevant data for that purpose.

Test: Record Completeness (The Number of Records with Empty Values in a Field/Column)

rows_with_empties
0

Test: Item Completeness (The Number Cells Missing Values in each Field/Column)

item	empties
unique_id	0
age_class_code	0
age_group_code	0
age_type_code	0
calendar_year_number	0
cust_race_is_amer_indian_alaska_native_ind	0
cust_race_is_hawaiian_pacific_islander_ind	0
customer_race_is_asisan_indicator	0
customer_race_is_black_indicator	0
customer_race_is_white_indicator	0
customer_race_is_other_indicator	0
ethnicity_code	0
foster_care_case_indicator	0
gender_code	0
month_of_birth	0
snap_case_indicator	0
tanf_case_indicator	0
year_of_birth	0

Valid Values

The concept of value validity can be conceptualized as the percentage of elements whose attributes possess expected values. The actualization of this concept generally comes in the form of straight-forward domain constraint rules.

Test: Count and Percentage of Invalid Values in each Field/Column

From a quick analysis of the graphs below it can be seen that the demographic fields in this dataset are maintained at a very high level in terms of the validity of the values stored with a score of 100% for almost all. A small number of invalid values were detected, however, in the field ‘ethnicity’. This number represents approximately 6.8% of the values stored in that field. However, having 93% valid values is still relatively high and this field may still serve as a decent indicator of ethnicity.

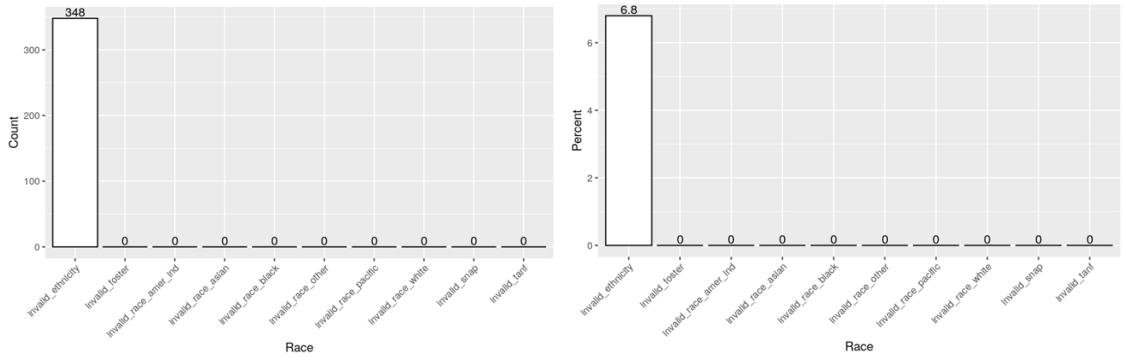


Figure 4 Count and Percentage of Individuals with Invalid Values
dataset: DSS Customers by Year

Longitudinal Consistency (Unexpected Changes in Demographics)

Longitudinal Consistency refers to a check for inconsistency in the data when checked over time (longitudinally), to see if the same value is recorded for every new record when it should be (i.e. birthdate and other demographics). Causes of longitudinal inconsistency are varied, but a common source of inconsistency comes from situations where locally derived information is provided with no associated master list or file. An exhaustive 'master list' of individuals receiving a public service are, in fact, quite rare. Many times, demographics are recorded in multiple records about the same individual, sometimes in the same time period. In these cases, truth must be derived from the aggregation of multiple observations.

Test: Count and Percentage of Individuals with Multiple Values per Demographic Item

While a small number of fluctuating demographic values are detected in this dataset, the respective percentages are fairly low and it is expected that they are manageable using standard deduplication approaches.

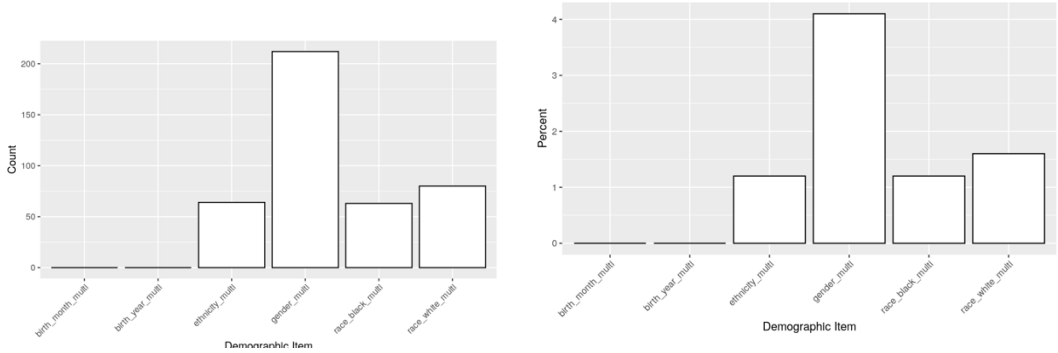


Figure 5 Count and Percentage of Individuals with Multiple Values per Demographic Item
dataset: DSS Customers by Year

Produce analyses and presentation of basic demographic breakdowns over time for all combinations of the select EC service data sets.

For the data sources we were able to link and pull (VDOE student record, VDSS services by year, OCS services), restructuring the data and the creation of deduplicated demographic breakdowns over time was straight-forward, presenting only issues normally experienced in such exercises. There isn't much data yet for some (OCS), but it's clear that it's not only possible to deduplicate and link the data sets to create unduplicated counts, but that such linkage could prove quite useful.

For example, we can now produce cross-tabulations of distinct counts across multiple services over time.

snap & tanf	black hispanic or latino	954	746	657	501
snap & tanf	black hispanic or latino not reported	1159	1710	1766	1333
snap & tanf	black not hispanic or latino	72710	62084	56655	40908
snap & tanf	other hispanic or latino	2112	1752	1529	1025
snap & tanf	other hispanic or latino not reported	143	162	129	66
snap & tanf	other not hispanic or latino	1870	1616	1287	877
snap & tanf	race not reported hispanic or latino	1438	1123	1064	749
snap & tanf	race not reported hispanic or latino not reported	4461	5727	8795	6941
snap & tanf	race not reported not hispanic or latino	5409	4375	4190	3213
snap & tanf	white hispanic or latino	5357	4412	4325	3424
snap & tanf	white hispanic or latino not reported	902	1889	1958	1264
snap & tanf	white not hispanic or latino	41489	34649	30686	21683

Figure 6 Excerpt from Distinct Count Cross-Tabulation, Service by Race/Ethnicity by Year
datasets: SNAP, TANF, FOSTER, OCS

As well as distinct count graphs.

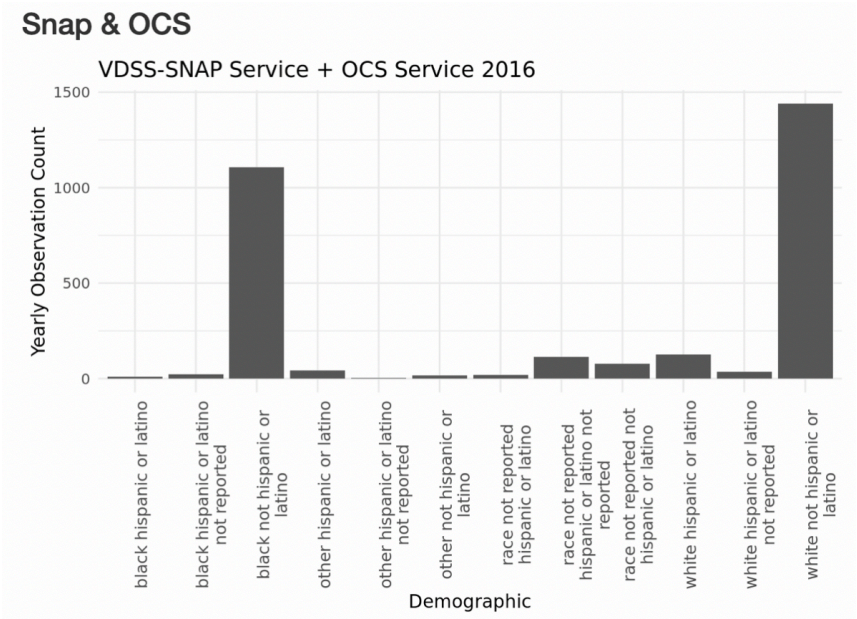


Figure 7 Individuals receiving both VDSS SNAP and OCS Services by Race and Ethnicity

And we can now visualize counts across geographies.

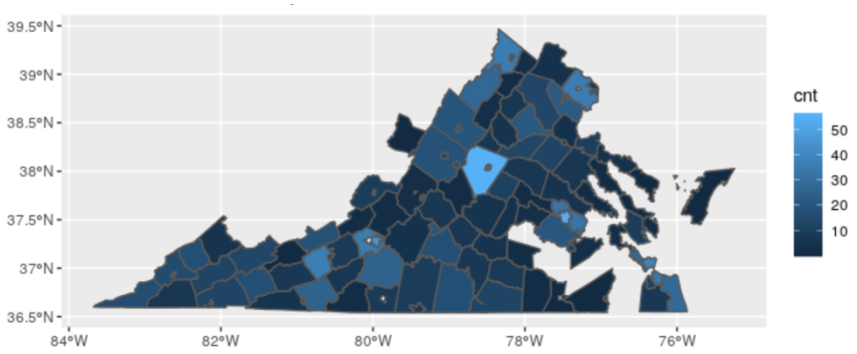


Figure 8 Count of Individuals Receiving SNAP and Part B Benefits 2015
datasets: DSS Customers by Year, DOE Student Records

Produce an initial composite index, for demonstration purposes, using the produced distinct counts.

In terms of demonstrating how such deduplication and linkage may prove useful, we created an initial scaled composite index combining nutritional and behavioral assistance rates per county in Virginia. The code for producing this index can be found in Appendix C.

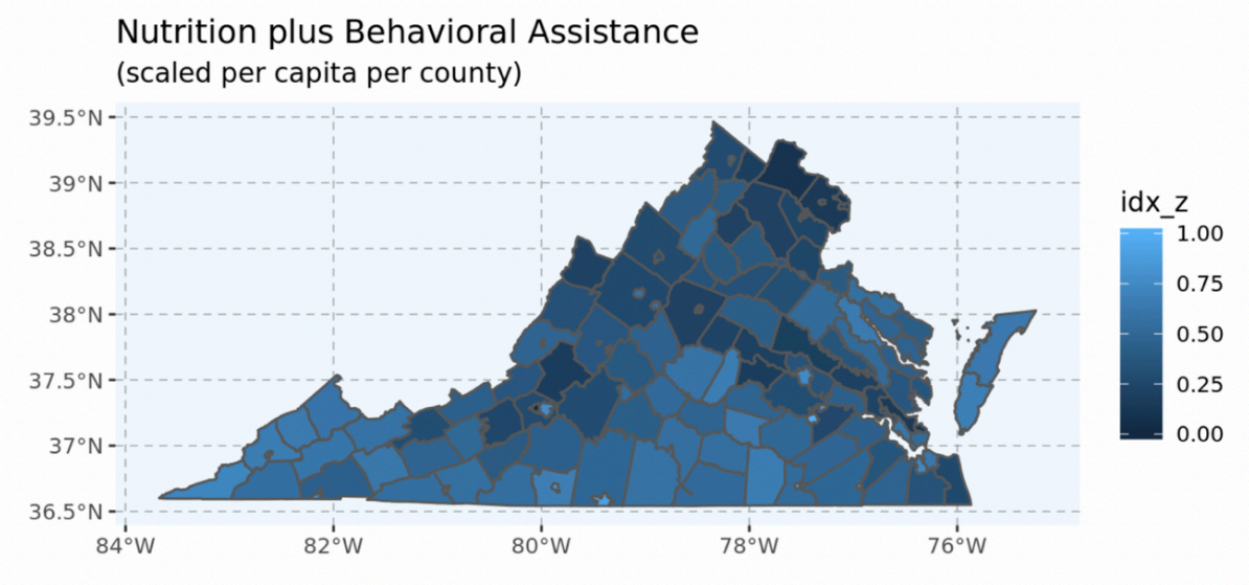


Figure 9 Example of Composite Index Constructed Using VLDS Distinct Count Linked Data

Profile Dataset: DSS SNAP Customers By Year

Dataset Preparation

Provided datasets are often vastly different from each other in terms of both schema and structure. To prepare for data profiling, datsets fields are checked for spelling errors and converted to a standardized format. If the dataset does not provide records at the level of aggregation required (e.g. each row is unique for a person and year) then the dataset is restructured.

Task: Preparation of Field/Column Names

Field/Column names standardized.

Task: Restructuring of Datset to Required Level of Aggregation

Significant restructuring required for use for this case. SNAP Customer Records By Year are actually **Customer Records by Year AND by "Location"**, so multiple records per customer occur if a customer received benefits in more than one fips code or zip code in a calendar year. As a single record is needed per customer per year for linkage purposes, additional columns must be created to account for all possible locations. The number of columns added is based on the customer with the highest number of locations in a single year. The code automatically determines the number. The result of the restructing can be seen in the addition of multiple county fips and zipcode columns.

fields_original	**fields_prepared**
Unique ID	unique_id
Calendar Year Number	calendar_year_number
County FIPS Code	county_fips_code_1
Number of Months enrolled in SNAP	county_fips_code_2
Zip Code	county_fips_code_3
NA	county_fips_code_4
NA	county_fips_code_5
NA	zip_code_1
NA	zip_code_2
NA	zip_code_3
NA	zip_code_4
NA	zip_code_5

unique_id	calendar_year_number	county_fips_code_1	county_fips_code_2	county_fips_code_3	county_fips_code_4	county_fips_code_5	zip_code_1	zip_code_2	zip_code_3	zip_code_4	zip_code_5
803970199814	2011	165	660	790			22812	22802	24401		
803970419330	2012	087	683	730	760		23227	20110	23803	23222	
803970564901	2012	009	031	037			24572	24598	23959		
803970709469	2013	015	163	530			24477	24416	24416		
803970725328	2015	121	155	750			24073	24141	24141		

Uniqueness

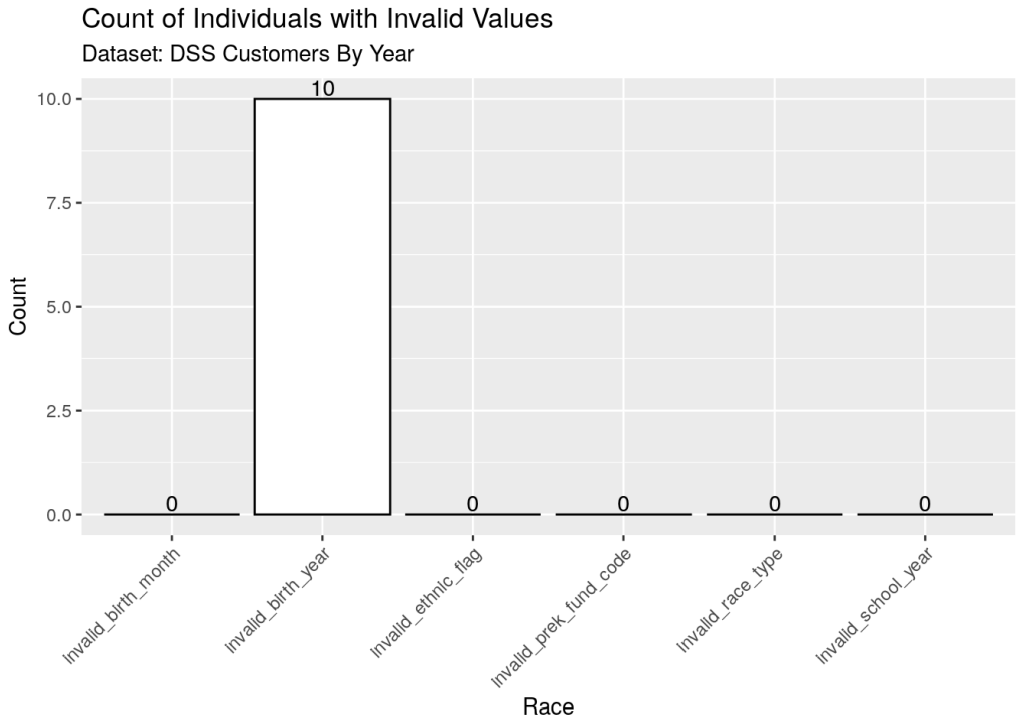
The concept of data uniqueness can be generalized as the number of unique valid values that have been entered in a record field, or as a combination of record field values within a dataset. Uniqueness is not generally discussed in terms of data quality, but for the purposes of answering research questions, the variety and richness of the data is of paramount importance. Most notably, if a record field has very little value uniqueness (e.g. entries in the field 'State' for an analysis of housing within a county, which of course would be within a single state), then its utility would be quite low and can be conceptualized as having low quality in terms of the research question at hand.

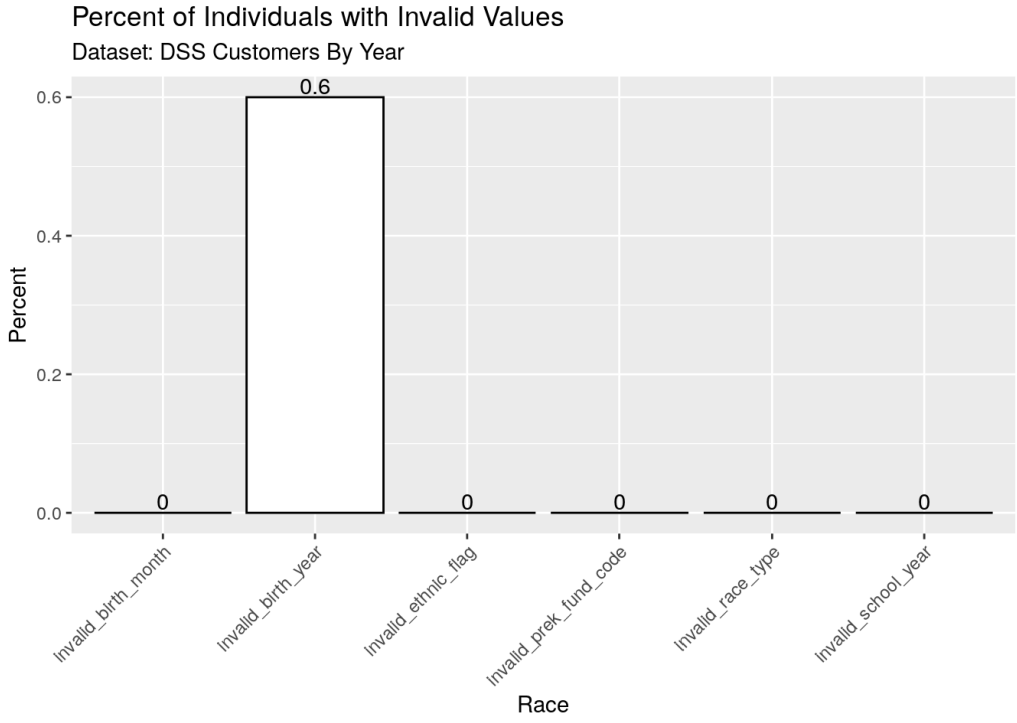
Test: Numerical Frequencies

There were no numerical items in this dataset

Longitudinal Consistency (Unexpected Changes in Demographics)

There are no longitudinal tests for this dataset.


Count of Individuals with Invalid Values
Dataset: DSS Customers By Year
Count
10.0
7.5
5.0
2.5
0.0
10
0
0
0
0
0
Invalid_birth_month
Invalid_birth_year
Invalid_ethnic_flag
Invalid_prek_fund_code
Invalid_race_type
Invalid_school_year
Race


Percent of Individuals with Invalid Values
Dataset: DSS Customers By Year
Percent
0.6
0.4
0.2
0.0
0.6
0
0
0
0
0
Invalid_birth_month
Invalid_birth_year
Invalid_ethnic_flag
Invalid_prek_fund_code
Invalid_race_type
Invalid_school_year
Race

rows_with_empties
1149

Test: Item Completeness (The Number Cells Missing Values in each Field/Column)

item	empties
prek_funding_code	1149
unique_id	0
school_year	0
birth_month	0
birth_year	0
race_type	0
ethnic_flag	0

Valid Values

The concept of value validity can be conceptualized as the percentage of elements whose attributes possess expected values. The actualization of this concept generally comes in the form of straight-forward domain constraint rules.

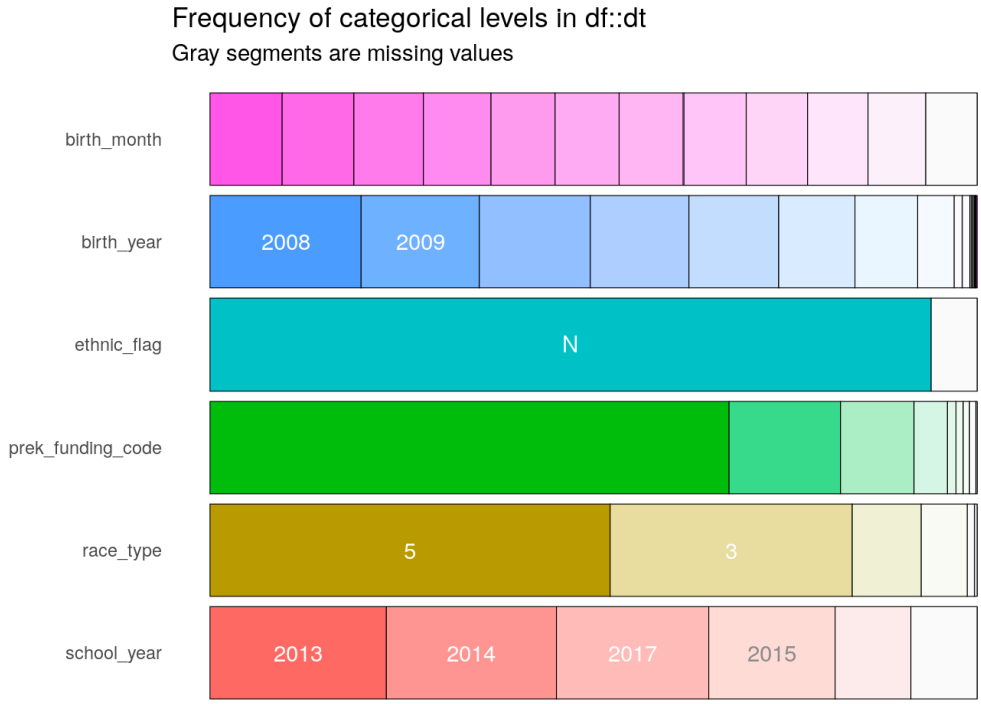
Test: Count and Percetage of Invalid Values in each Field/Column

of the data is of paramount importance. Most notably, if a record field has very little value uniqueness (e.g. entries in the field 'State' for an analysis of housing within a county, which of course would be within a single state), then its utility would be quite low and can be conceptualized as having low quality in terms of the research question at hand.

Test: Numerical Frequencies

There were no numerical items in this dataset

Test: Categorical Frequencies



Completeness

The concept of data completeness can be generalized as the proportion of data provided versus the proportion of data required. Data that is missing may additionally be categorized as record fields not containing data, records not containing necessary fields, or datasets not containing the requisite records. The most common conceptualization of completeness is the first, record field not containing data. This conceptualization of data completeness can be thought of as the proportion of the data that has values to the proportion of data that 'should' have values. That is, a set of data is complete with respect to a given purpose if the set contains all the relevant data for that purpose.

Test: Record Completeness (The Number of Records with Empty Values in a Field/Column)

rows_with_empties

Profile Dataset: DOE Student Records

Dataset Preparation

Provided datasets are often vastly different from each other in terms of both schema and structure. To prepare for data profiling, datsets fields are checked for spelling errors and converted to a standardized format. If the dataset does not provide records at the level of aggregation required (e.g. each row is unique for a person and year) then the dataset is restructured.

Task: Preparation of Field/Column Names

Field/Column names standardized.

Task: Restructuring of Datset to Required Level of Aggregation

Significant restructuring required for use for this case. The Student Records dataset is longitudinal, meaning by definition there are multiple records per student per year. Therefore, some form of deduplication becomes necessary to get to the required level of aggregation, in this case one record per student per year. A deduplication algorithm based on the premise “majority wins, tie goes to most recently entered” was developed and employed for the deduplication, and the columns were subselected to those necessary for the study.

fields_original	fields_prepared
Unique ID	unique_id
Birth Month	school_year
Birth Year	birth_month
Division Number - Reporting School Number	birth_year
Division Number - Responsible School Number	race_type
Division Number - Serving School Number	ethnic_flag
Ethnic Flag	prek_funding_code
Grade Code	NA
PK Experience Type	NA
Prek Funding Code	NA
Race Type	NA
Reported Race Type	NA
School Year	NA

unique_id	school_year	birth_month	birth_year	race_type	ethnic_flag	prek_funding_code
603974774098	2016	12	2011	5	N	3
603987125435	2017	03	2012	5	N	1
603978415800	2017	04	2012	5	N	5
603975118878	2017	02	2012	5	N	3
603987255304	2017	11	2012	3	N	1

Uniqueness

The concept of data uniqueness can be generalized as the number of unique valid values that have been entered in a record field, or as a combination of record field values within a dataset. Uniqueness is not generally discussed in terms of data quality, but for the purposes of answering research questions, the variety and richness

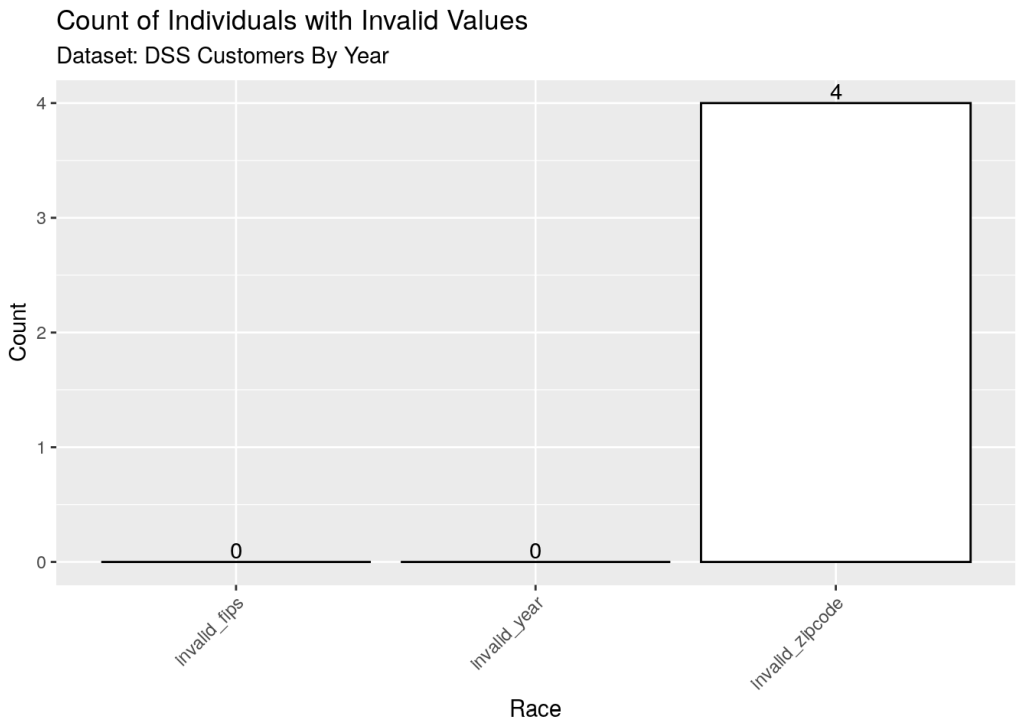
Investigate Zipcode Further

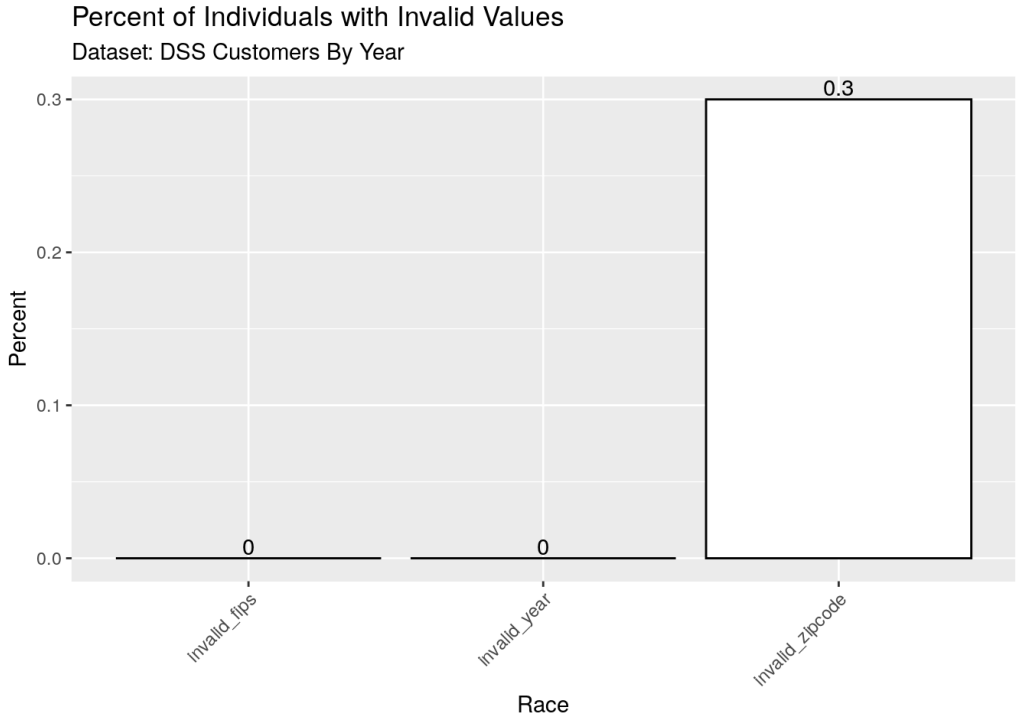
Invalid Zip Codes

current_placement_zip_code
37683
232201298
237045343
33873

Longitudinal Consistency (Unexpected Changes in Demographics)

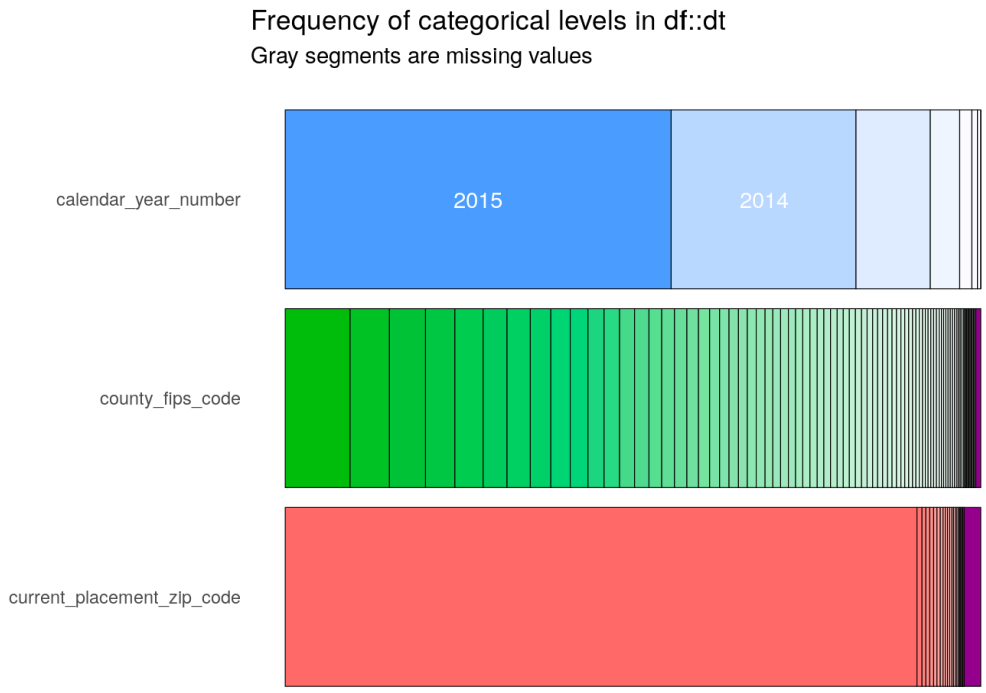
There are no longitudinal tests for this dataset.


Count of Individuals with Invalid Values
Dataset: DSS Customers By Year
Count
4
3
2
1
0
0
0
4
Invalid_fips
Invalid_year
Invalid_zipcode
Race


Percent of Individuals with Invalid Values
Dataset: DSS Customers By Year
Percent
0.3
0.2
0.1
0.0
0
0
0.3
Invalid_fips
Invalid_year
Invalid_zipcode
Race

The concept of value validity can be conceptualized as the percentage of elements whose attributes possess expected values. The actualization of this concept generally comes in the form of straight-forward domain constraint rules.

Test: Count and Percetage of Invalid Values in each Field/Column



Completeness

The concept of data completeness can be generalized as the proportion of data provided versus the proportion of data required. Data that is missing may additionally be categorized as record fields not containing data, records not containing necessary fields, or datasets not containing the requisite records. The most common conceptualization of completeness is the first, record field not containing data. This conceptualization of data completeness can be thought of as the proportion of the data that has values to the proportion of data that 'should' have values. That is, a set of data is complete with respect to a given purpose if the set contains all the relevant data for that purpose.

Test: Record Completeness (The Number of Records with Empty Values in a Field/Column)

rows_with_empties
1138

Test: Item Completeness (The Number Cells Missing Values in each Field/Column)

item	empties
current_placement_zip_code	1138
unique_id	0
calendar_year_number	0
county_fips_code	0

Valid Values

Profile Dataset: DSS Foster Customers By Year

Dataset Preparation

Provided datasets are often vastly different from each other in terms of both schema and structure. To prepare for data profiling, datsets fields are checked for spelling errors and converted to a standardized format. If the dataset does not provide records at the level of aggregation required (e.g. each row is unique for a person and year) then the dataset is restructured.

Task: Preparation of Field/Column Names

fields_original	fields_prepared
Unique ID	unique_id
Calendar Year Number	calendar_year_number
County FIPS Code	county_fips_code
Current Placement Zip Code	current_placement_zip_code

Task: Restructuring of Datset to Required Level of Aggregation

No restructuring was required for this dataset.

Uniqueness

The concept of data uniqueness can be generalized as the number of unique valid values that have been entered in a record field, or as a combination of record field values within a dataset. Uniqueness is not generally discussed in terms of data quality, but for the purposes of answering research questions, the variety and richness of the data is of paramount importance. Most notably, if a record field has very little value uniqueness (e.g. entries in the field 'State' for an analysis of housing within a county, which of course would be within a single state), then its utility would be quite low and can be conceptualized as having low quality in terms of the research question at hand.

Test: Numerical Frequencies

There were no numerical items in this dataset

Test: Categorical Frequencies

Investigate Zipcode Further

Invalid
Zip
Codes

zipcode
20508
23353
21740
22505
23353

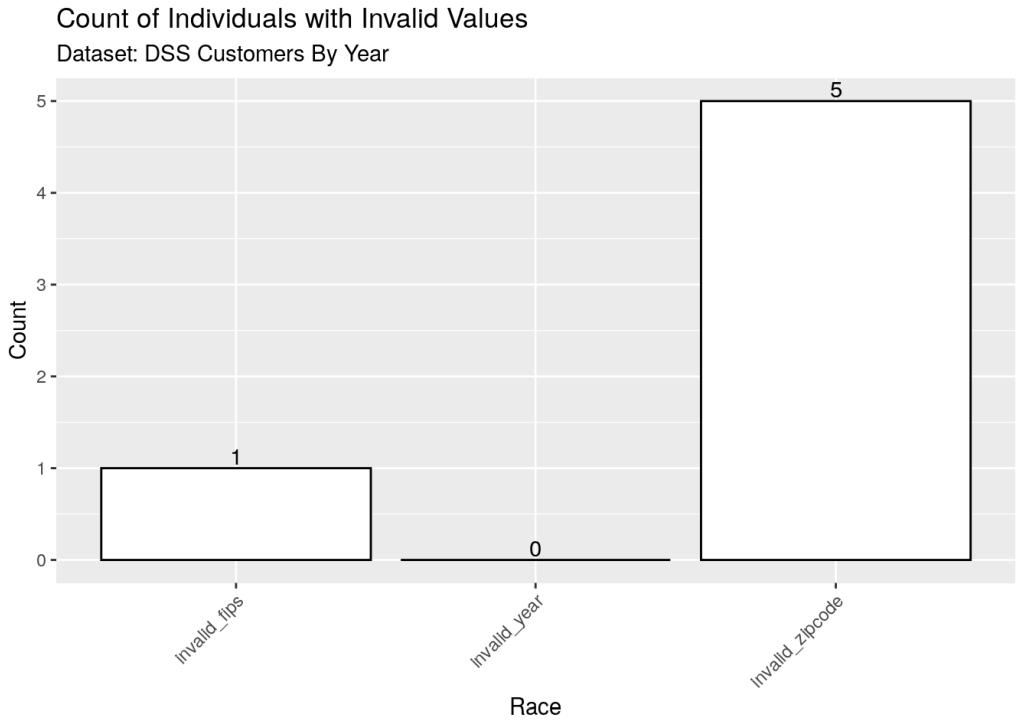
Investigate County FIPS Further

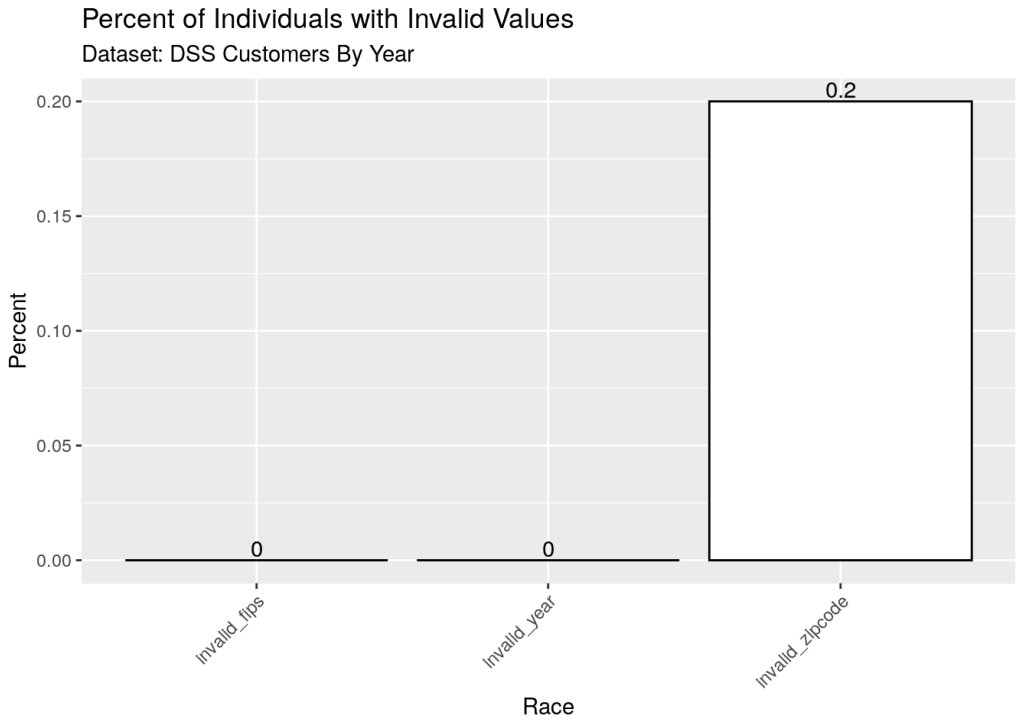
Invalid County
FIPS Codes

county_fips_code
515

Longitudinal Consistency (Unexpected Changes in Demographics)

There are no longitudinal tests for this dataset.


Count of Individuals with Invalid Values
Dataset: DSS Customers By Year
Count
5
4
3
2
1
0
1
0
5
Invalid_fips
Invalid_year
Invalid_zipcode
Race


Percent of Individuals with Invalid Values
Dataset: DSS Customers By Year
Percent
0.20
0.15
0.10
0.05
0.00
0
0
0.2
Invalid_fips
Invalid_year
Invalid_zipcode
Race

Test: Record Completeness (The Number of Records with Empty Values in a Field/Column)

rows_with_empties
3064

Test: Item Completeness (The Number Cells Missing Values in each Field/Column)

item	empties
county_fips_code_4	3064
zip_code_4	3064
county_fips_code_3	3025
zip_code_3	3025
county_fips_code_2	2716
zip_code_2	2716
unique_id	0
calendar_year_number	0
county_fips_code_1	0
zip_code_1	0

Valid Values

The concept of value validity can be conceptualized as the percentage of elements whose attributes possess expected values. The actualization of this concept generally comes in the form of straight-forward domain constraint rules.

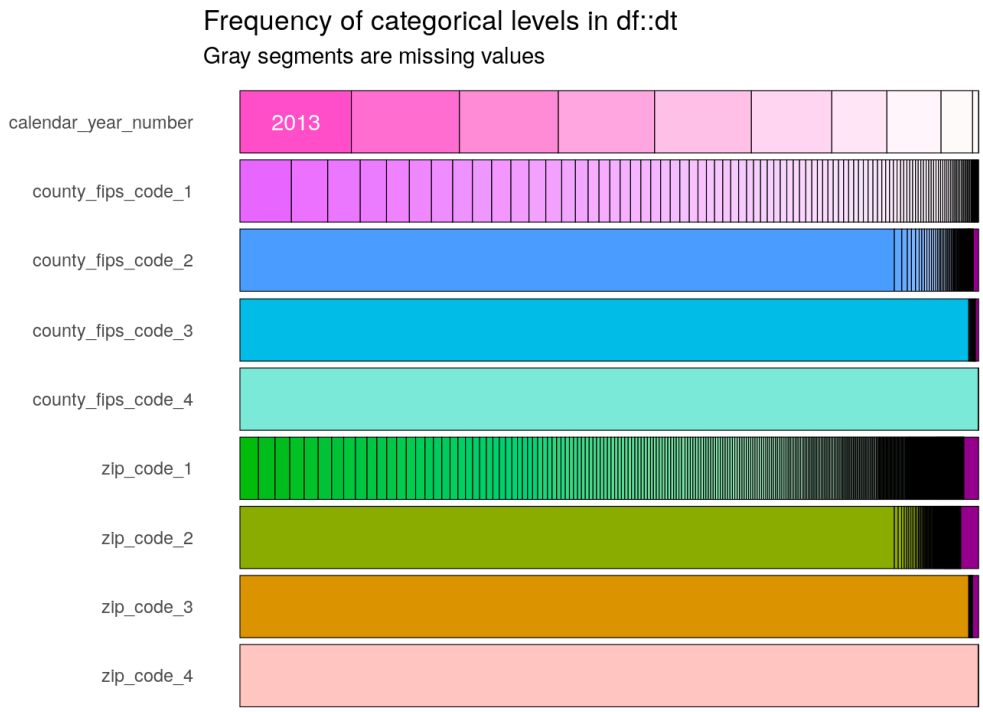
Test: Count and Percetage of Invalid Values in each Field/Column

The concept of data uniqueness can be generalized as the number of unique valid values that have been entered in a record field, or as a combination of record field values within a dataset. Uniqueness is not generally discussed in terms of data quality, but for the purposes of answering research questions, the variety and richness of the data is of paramount importance. Most notably, if a record field has very little value uniqueness (e.g. entries in the field 'State' for an analysis of housing within a county, which of course would be within a single state), then its utility would be quite low and can be conceptualized as having low quality in terms of the research question at hand.

Test: Numerical Frequencies

There were no numerical items in this dataset

Test: Categorical Frequencies



Completeness

The concept of data completeness can be generalized as the proportion of data provided versus the proportion of data required. Data that is missing may additionally be categorized as record fields not containing data, records not containing necessary fields, or datasets not containing the requisite records. The most common conceptualization of completeness is the first, record field not containing data. This conceptualization of data completeness can be thought of as the proportion of the data that has values to the proportion of data that 'should' have values. That is, a set of data is complete with respect to a given purpose if the set contains all the relevant data for that purpose.

Profile Dataset: DSS TANF Customers By Year

Dataset Preparation

Provided datasets are often vastly different from each other in terms of both schema and structure. To prepare for data profiling, datsets fields are checked for spelling errors and converted to a standardized format. If the dataset does not provide records at the level of aggregation required (e.g. each row is unique for a person and year) then the dataset is restructured.

Task: Preparation of Field/Column Names

Field/Column names standardized.

Task: Restructuring of Datset to Required Level of Aggregation

Significant restructuring required for use for this case. TANF Customer Records By Year are actually **Customer Records by Year AND by "Location"**, so multiple records per customer occur if a customer received benefits in more than one fips code or zip code in a calendar year. As a single record is needed per customer per year for linkage purposes, additional columns must be created to account for all possible locations. The number of columns added is based on the customer with the highest number of locations in a single year. The code automatically determines the number. The result of the restructing can be seen in the addition of multiple county fips and zipcode columns.

fields_original	**fields_prepared**
Unique ID	unique_id
Calendar Year Number	calendar_year_number
County FIPS Code	county_fips_code_1
Number of Months enrolled in TANF	county_fips_code_2
Zip Code	county_fips_code_3
NA	county_fips_code_4
NA	zip_code_1
NA	zip_code_2
NA	zip_code_3
NA	zip_code_4

unique_id	calendar_year_number	county_fips_code_1	county_fips_code_2	county_fips_code_3	county_fips_code_4	zip_code_1	zip_code_2	zip_code_3	zip_code_4
803970199814	2012	015	770	790		24477	24477	24401	
803970584901	2012	031	009	037		24588	24572	23959	
803970725328	2015	121	155	750		24073	24141	24141	
803971492392	2013	111	147	680		23944	23901	23944	
803971805461	2009	047	137	683		20110	22960	20110	

Uniqueness

Invalid
Zip
Codes

zipcode
20508
23353
27587
24950
24950
23136
08854
22505
23907
23907
34343
34343
27217
22012
22012
49109
23353
24950
24950
27030
27356
49109
71446
27217
23353
22890
22890
22890
24817
29168
22006
28675
90293

Investigate County FIPS Further

Invalid County
FIPS Codes

county_fips_code
515
515
515
515
515

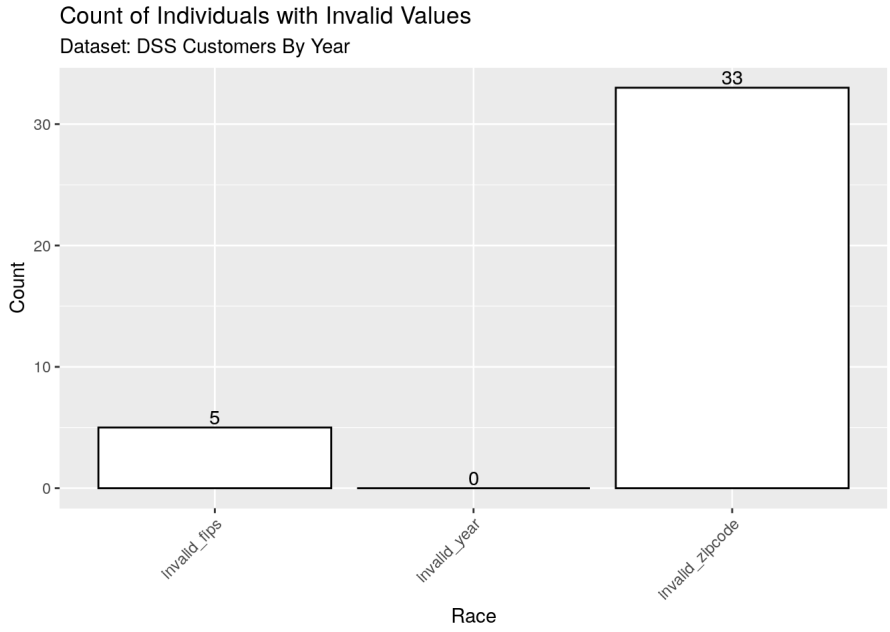
Longitudinal Consistency (Unexpected Changes in Demographics)

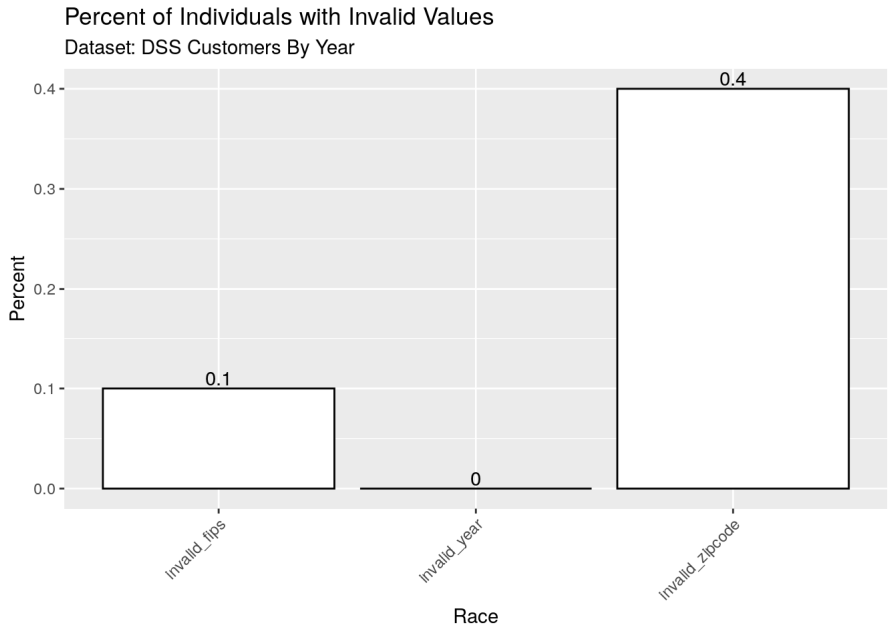
There are no longitudinal tests for this dataset.

Valid Values

The concept of value validity can be conceptualized as the percentage of elements whose attributes possess expected values. The actualization of this concept generally comes in the form of straight-forward domain constraint rules.

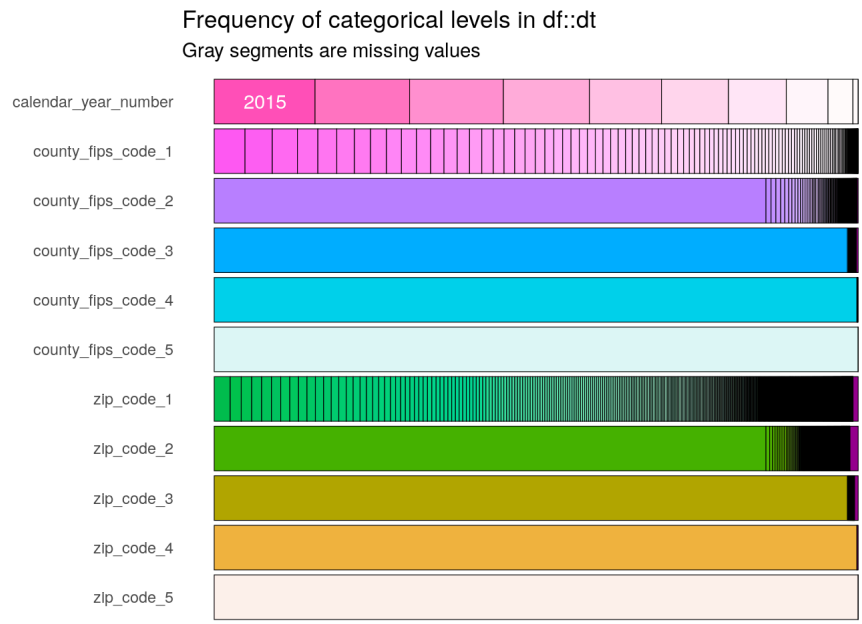
Test: Count and Percetage of Invalid Values in each Field/Column





Investigate Zipcode Further

Test: Categorical Frequencies



Completeness

The concept of data completeness can be generalized as the proportion of data provided versus the proportion of data required. Data that is missing may additionally be categorized as record fields not containing data, records not containing necessary fields, or datasets not containing the requisite records. The most common conceptualization of completeness is the first, record field not containing data. This conceptualization of data completeness can be thought of as the proportion of the data that has values to the proportion of data that 'should' have values. That is, a set of data is complete with respect to a given purpose if the set contains all the relevant data for that purpose.

Test: Record Completeness (The Number of Records with Empty Values in a Field/Column)

rows_with_empties
8310

Test: Item Completeness (The Number Cells Missing Values in each Field/Column)

item	empties
county_fips_code_5	8310
zip_code_5	8310
county_fips_code_4	8295
zip_code_4	8295
county_fips_code_3	8173
zip_code_3	8173
county_fips_code_2	7121
zip_code_2	7121
zip_code_1	6
unique_id	0
calendar_year_number	0
county_fips_code_1	0

APPENDIX B – APPLICATION OF DEDUPLICATION ALGORITHM FOR DOE STUDENT RECORD DEMOGRAPHICS

Ingest DOE Student Record Demographics

LOAD LIBRARIES AND FUNCTIONS

```
library(data.table)
library(dataplumbr)
library(here)
library(inspectdf)
library(maditr)
```

DOE Student Record Demographics

LOAD DATA FILE

```
doe_student_records <- fread(here("data/original/q5/DOE/Student Records.csv")
, colClasses = "character")
```

STANDARDIZE COLUMN NAMES

```
colnames(doe_student_records) <- name.standard_col_names(colnames(doe_student
_records))
```

CHECK IF MORE THAN ONE RECORD PER UNIQUE_ID AND CALENDAR_YEAR

```
multiples <- nrow(doe_student_records[, .N,.(unique_id, school_year)][N > 1])
multiples
```

```
## [1] 2810
```

APPLY DEDUPLICATION ALGORITHM TO GET DEMOGRAPHICS BY YEAR

```
doe_student_dmgs <- doe_student_records[, .(birth_month, birth_year, race_typ
e, ethnic_flag, prek_funding_code),.(unique_id, school_year)]

set.dedup_choice <- function(df) {
    dt <- data.table::setDT(df)
    for (j in colnames(dt)) {
        data.table::set(dt, j = j, value = dt[get(j) != "", .N, j][order(-N)]
[, ..j][1])
    }
    dt[1]
}

set.dedup_choice_by_key <- function(df, key = "uid") {
    if (exists("out_dt") == TRUE) rm(out_dt, envir = globalenv())

    dt <- data.table::setDT(df)
    unique_keys <- unique(dt[, get(key)])
```

```
    key_cnt <- length(unique_keys)
    pb <- progress::progress_bar$new(format = "[:bar] :current/:total :percen
t eta: :eta", total = key_cnt)

    for (k in unique_keys) {
        pb$tick()
        g <- dt[get(key)==k]
        r <- set.dedup_choice(g)
        if (exists("out_dt") == FALSE) out_dt <- r else out_dt <- rbindlist(l
ist(out_dt, r))
    }

    out_dt
}

doe_student_dmgs_dedup <- set.dedup_choice_by_key(doe_student_dmgs, "unique_i
d")

# verify only one code per id per year
nrow(doe_student_dmgs_dedup[, .N, .(unique_id)][N > 1])

## [1] 0
```

WRITE TO CSV

```
fwrite(doe_student_dmgs_dedup, here("data/working/DOE/doe_student_records_by_
year_dmgs_prek.csv"))
```

APPENDIX C – CODE TO GENERATE COMPOSITE INDEX

LOAD LIBRARIES

```
library(data.table)
library(dataplumbr)
library(tidycensus)
library(sf)
library(ggplot2)
library(here)
library(knitr)
library(kableExtra)
```

DSS RECORDS

Load DSS records and standardize column names

```
dss_customers_by_year <- fread(here("data/original/q4/DSS/DSS Customers By Year.csv"), colClasses = "character")

colnames(dss_customers_by_year) <- standard_col_names(colnames(dss_customers_by_year))
```

fix a misspelling for future joining

```
colnames(dss_customers_by_year)[colnames(dss_customers_by_year) == "calender_year_number"] <- "calendar_year_number"
```

subset to just the data columns needed

```
dss_customers_by_year_sub <-
  dss_customers_by_year[, .(unique_id,
                            calendar_year_number,
                            snap_case_indicator,
                            tanf_case_indicator,
                            foster_care_case_indicator)]
# print table
kable(dss_customers_by_year[1:4]) %>% kable_styling() %>% scroll_box(width = "910px")
```

OCS RECORDS

Load OCS records and standardize column names

```
ocs_services_by_year <- fread(here("data/original/q4/OCS/OCS Services By Year
.csv"), colClasses = "character")
```

```
## Warning in fread(here("data/original/q4/OCS/OCS Services By Year.csv"), :

## Discarded single-line footer: <<ZKAUQQQ,D,1644259,2,>>
```

```
colnames(ocs_services_by_year) <- standard_col_names(colnames(ocs_services_by
_year))
```

Group the OCS service records by year to create a single record per customer per year

```
ocs_customers_by_year <- ocs_services_by_year[, .(ocs_service_entries = .N),
.(unique_id, program_year)]

# print table

kable(ocs_customers_by_year[1:4]) %>% kable_styling() %>% scroll_box(width =
"910px")
```

unique_id	program_year
MZ4CQQQ	2016
2942AQQ	2016
K929QQQ	2016
GHCHQQQ	2016

Join the DSS and OCS records

```
colnames(ocs_customers_by_year)[colnames(ocs_customers_by_year) == "program_y
ear"] <- "calendar_year_number"

dss_ocs_cust_by_year <- merge(dss_customers_by_year_sub, ocs_customers_by_yea
r, by = c("unique_id", "calendar_year_number"), all.x = TRUE)
```

add a service indicator variable for later use

```
dss_ocs_cust_by_year[!is.na(ocs_service_entries), ocs_indicator := "Y"]

dss_ocs_cust_by_year[is.na(ocs_service_entries), ocs_indicator := "N"]
```

SNAP RECORDS

Load SNAP records and standardize column names

```
snap_cust_by_loc_year <- fread(here("data/original/q4/DSS/DSS SNAP Customers by Year.csv"), colClasses = "character")

colnames(snap_cust_by_loc_year) <- standard_col_names(colnames(snap_cust_by_loc_year))
```

SNAP records are actually Customer by Year by “Location”, so multiple records per customer

if they received benefits in more than one fips code or zip code. As a single record is needed per customer, additional columns must be created to account for all possible locations. The number of columns added is based on the customer with the highest number of locations in a single year. In this case it is six, but the code automatically determines the number.

```
# each county fips code gets it's own column, each zip code gets its own column

snap_cust_by_loc_year[, county_fips_code_no := paste("county_fips_code", seq_len(.N), sep="_"), by=c("unique_id", "study_group_id", "calendar_year_number")]

snap_cust_by_loc_year[, zip_code_no := paste("zip_code", seq_len(.N), sep="_"), by=c("unique_id", "study_group_id", "calendar_year_number")]

fips <- dcast(snap_cust_by_loc_year, unique_id + study_group_id + calendar_year_number ~ county_fips_code_no, value.var=c("county_fips_code"))

zips <- dcast(snap_cust_by_loc_year, unique_id + study_group_id + calendar_year_number ~ zip_code_no, value.var=c("zip_code"))

snap_cust_by_year <- merge(fips, zips, by=c("unique_id", "study_group_id", "calendar_year_number"))

# print table

kable(snap_cust_by_year[!is.na(county_fips_code_5)][order(-county_fips_code_6)][1:25]) %>% kable_styling() %>% scroll_box(width = "100%")
```

Join the DSS, OCS and SNAP records

```
dss_ocs_snap_cust_by_year <- merge(dss_ocs_cust_by_year, snap_cust_by_year, by = c("unique_id", "calendar_year_number"), all.x = TRUE)
```

```
dss_ocs_snap_cust_by_year <- dss_ocs_snap_cust_by_year[!is.na(county_fips_cod
e_1)]

ocs_snap_cnt_fips_by_year <- dss_ocs_snap_cust_by_year[, .N, c("county_fips_c
ode_1", "calendar_year_number")]

# print table

kable(ocs_snap_cnt_fips_by_year[1:4]) %>% kable_styling() %>% scroll_box(widt
h = "910px")
```

Get population by county by year for Virginia

```
va_pop_co_2013 <- data.table::setDT(tidycensus::get_acs(geography = "county",
variables = "B01001_001", state = "VA", year = 2013))

va_pop_co_2013[, year := "2013"]

colnames(va_pop_co_2013)[colnames(va_pop_co_2013) == 'estimate'] <- 'estimate
_2013'

va_pop_co_2013 <- va_pop_co_2013[, .(GEOID, estimate_2013, year)]

va_pop_co_2014 <- data.table::setDT(tidycensus::get_acs(geography = "county",
variables = "B01001_001", state = "VA", year = 2014))
va_pop_co_2014[, year := "2014"]
colnames(va_pop_co_2014)[colnames(va_pop_co_2014) == 'estimate'] <- 'estimate
_2014'
va_pop_co_2014 <- va_pop_co_2014[, .(GEOID, estimate_2014, year)]

va_pop_co_2015 <- data.table::setDT(tidycensus::get_acs(geography = "county",
variables = "B01001_001", state = "VA", year = 2015))
va_pop_co_2015[, year := "2015"]
colnames(va_pop_co_2015)[colnames(va_pop_co_2015) == 'estimate'] <- 'estimate
_2015'
va_pop_co_2015 <- va_pop_co_2015[, .(GEOID, estimate_2015, year)]

va_pop_co_2016 <- data.table::setDT(tidycensus::get_acs(geography = "county",
variables = "B01001_001", state = "VA", year = 2016))
va_pop_co_2016[, year := "2016"]
colnames(va_pop_co_2016)[colnames(va_pop_co_2016) == 'estimate'] <- 'estimate
_2016'
va_pop_co_2016 <- va_pop_co_2016[, .(GEOID, estimate_2016, year)]
```

Combine Population Counts for Each Year

```
colnames(ocs_snap_cnt_fips_by_year) <- c("GEOID", "year", "N")

ocs_snap_cnt_fips_by_year[, GEOID := paste0("51", GEOID)]

ocs_snap_cnt_fips_by_year <- merge(ocs_snap_cnt_fips_by_year, va_pop_co_2013,
by = c("GEOID", "year"), all.x = T)

ocs_snap_cnt_fips_by_year <- merge(ocs_snap_cnt_fips_by_year, va_pop_co_2014,
by = c("GEOID", "year"), all.x = T)
```

```
ocs_snap_cnt_fips_by_year <- merge(ocs_snap_cnt_fips_by_year, va_pop_co_2015,
by = c("GEOID", "year"), all.x = T)

ocs_snap_cnt_fips_by_year <- merge(ocs_snap_cnt_fips_by_year, va_pop_co_2016,
by = c("GEOID", "year"), all.x = T)

ocs_snap_cnt_fips_by_year[, pop_est := gsub("NA", "", paste0(estimate_2013, e
stimate_2014, estimate_2015, estimate_2016))]
```

Create Index “Idx” as the count of those with both SNAP and OCS in a county for a particular year

```
ocs_snap_cnt_fips_by_year <- ocs_snap_cnt_fips_by_year[, .(GEOID, year, snap_
plus_ocs = N, pop_est, idx = N/as.numeric(pop_est))]

# print table

kable(ocs_snap_cnt_fips_by_year[1:4]) %>% kable_styling() %>% scroll_box(widt
h = "910px")
```

Create the Nutritional - Behavioral Index Map

Download the Geography

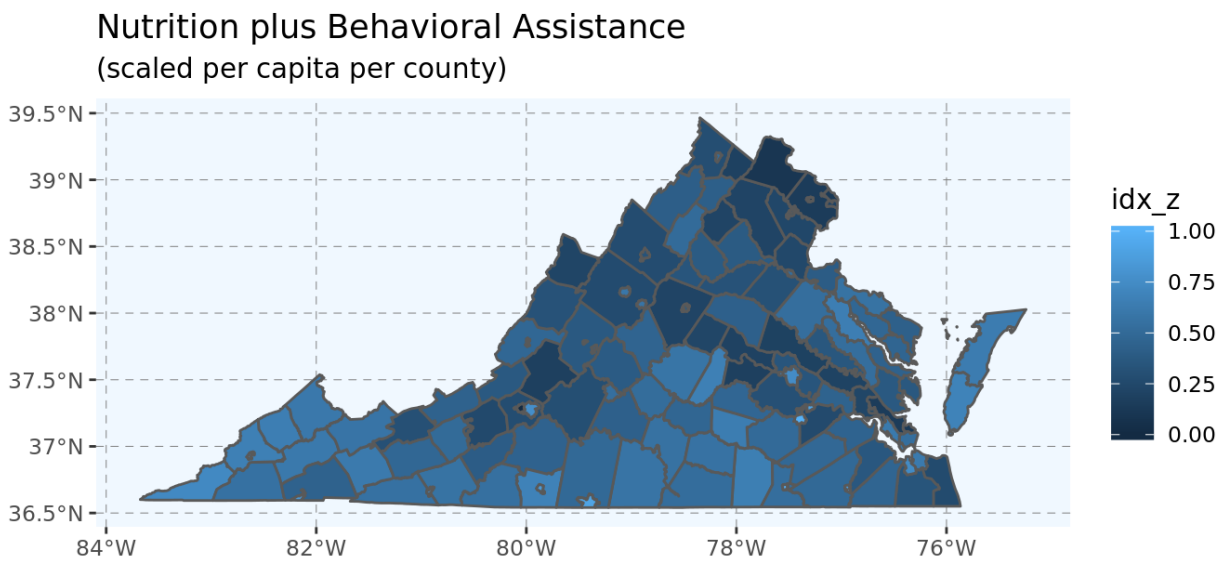
```
va_geo <- tidycensus::get_acs(geography = "county", variables = "B01001_001",
state = "VA", year = 2016, geometry = TRUE)
```

Chose year to map and create a standardized index from 0 to 1

Combine the data and geography and create the map

```
va_geo_idx_2013 <- merge(va_geo, idx_2013, by = "GEOID")

ggplot(data = va_geo_idx_2013) +
geom_sf(aes(fill = idx_z)) +
ggtitle("Nutrition plus Behavioral Assistance", subtitle = "(scaled per capit
a per county)") +
theme(panel.grid.major = element_line(color = gray(0.5), linetype = "dashed",
size = 0.5), panel.background = element_rect(fill = "aliceblue"))
```


Nutrition plus Behavioral Assistance
(scaled per capita per county)
39.5°N
39°N
38.5°N
38°N
37.5°N
37°N
36.5°N
84°W
82°W
80°W
78°W
76°W
idx_z
1.00
0.75
0.50
0.25
0.00