Data Transformations and Queries

Using SAS

Dr. Austin Brown

Kennesaw State University

Simple Data Transformations/Queries

- **Objective**: Learn to perform basic data transformations such as selecting columns, filtering rows, and creating new columns.
- **Importance**: Essential for cleaning and preparing data for analysis.
- **Key Points**
  - **KEEP Statement**: Specifies variables to retain in the dataset.
  - **DROP Statement**: Specifies variables to drop in the dataset.
  - **Syntax**:

```
data new_dataset;
 set old_dataset (keep=variables);
run;`


data new_dataset;
 set old_dataset (Drop=variables);
run;
```

  - **Example**: Code snippet showing how to select specific columns.

Simple Data Transformations/Queries

- In many instances, you may need to perform simple data transformations or queries to extract specific information from your dataset.
- This could involve selecting specific columns, filtering rows based on certain conditions, or creating new columns based on existing data.
- In SAS, the `KEEP` and `DROP` statements within the `DATA` step are used to select specific columns from a dataset.

Selecting Columns in SAS with KEEP Statement

- InSAS Studio, upload the HEART.csv file as we have done previously.
- Go ahead and import the data using PROC IMPORT as we have done already.
- Now, suppose instead of working with the full dataframe, I want to only focus on a few specific columns:
  - Chol_Status
  - BP_Status
  - Weight_Status
  - Smoking_Status

Selecting Columns in SAS with `KEEP` Statement

- While there are multiple ways of creating a new dataset which contains only these four columns, one of the most straightforward ways is to use the `KEEP` statement within the `DATA` step.
- The `KEEP` statement allows you to choose specific columns from a dataset and create a new dataset with only those columns.
- This can be useful when you have a large dataset with many columns, but you are only interested in a subset of them.
- Let's see how this works with the HEART dataset.

Selecting Columns in SAS with KEEP Statement

```
/* Import HEART.csv file */
proc import
  datafile="HEART.csv"
  out=heart
  dbms=csv
  replace;
  getnames=yes;
  run;

/* KEEP specific columns */
data selected_columns;
  set heart
  (keep=Chol_Status
        BP_Status
        Weight_Status
        Smoking_Status);
run;

/* Print the new dataset to verify */
proc print
  data=selected_columns(obs=5);
run;
```

Selecting Columns in SAS with KEEP Statement

- In the above code snippet, we first import the HEART.csv file using the PROC IMPORT procedure and store it in a dataset called heart.
- We then use the KEEP statement within the DATA step to select the specific columns we are interested in and store the result in a new dataset called selected_columns.

Filtering Rows in SAS with `WHERE` Statement

- Not only can we select columns, but we can also filter rows based on specific conditions.
- For example, in the HEART dataset, we may want to filter out all rows where the `Chol_Status` is `High`.
- That is, we want to keep only the rows where `Chol_Status` is not `High`.
- To do this, we can use the `WHERE` statement within the `DATA` step.

Filtering Rows in SAS with `WHERE` Statement

```
/* Filter rows where Chol_Status is not High */
data filtered_rows;
  set heart;
  where Chol_Status ne 'High';
run;

/* Print the new dataset to verify */
proc print
  data=filtered_rows;
run;
```

Filtering Rows in SAS with `WHERE` Statement

- In the above code snippet, we use the `WHERE` statement within the `DATA` step to filter out rows where the `Chol_Status` is `High`.
- The syntax `where Chol_Status ne 'High';` specifies that we want to keep only the rows where `Chol_Status` is not `High`.
- Note, `ne` is the SAS operator for “not equal to”.

Creating New Columns in SAS with `DATA` Step

- Many times, you may need to create new columns based on existing data in your dataset.
- For example, in the HEART dataset, you may want to create a new column that represents patient BMI.
- The `DATA` step in SAS is used to create new columns.

Creating New Columns in SAS with DATA Step

```
/* Add BMI to heart dataset */
data heart;
  set heart;
  BMI = (Weight / (Height**2))*703;
run;
/* Print the new dataset to verify */
proc print
  data=heart;
run;
```