1. SAS Basics

SAS Basics

Creating Tables

Example SAS code to create a table

```
data weight_club;
input IdNumber 1-4 Name $ 6-24 Team $ StartWeight EndWeight;
Loss = StartWeight - EndWeight;
datalines;
1023 David Shaw         red 189 165
1049 Amelia Serrano     yellow 145 124
1219 Alan Nance         red 210 192
1246 Ravi Sinha         yellow 194 177
1078 Ashley McKnight    red 127 118
;
run;
```

`data` keyword is used to define the table name (table names must not contain spaces)

`input` keyword is used to define the column names.

`IdNumber 1-4` is used to state that the characters in position `1-4` of each row would fall under the column `IdNumber` and similarly for `Name $ 6-24`. This eliminates the need of delimiters.

The `$` after `Name` and `Team` indicates that values of the `Name` and `Team` columns are strings.

`Loss = StartWeight - EndWeight` is used to defined a new colum named `Loss` derived from `StartWeight` and `EndWeight`.

`dataline` or `cards` is used to indicate the beginning of the table.

All command lines end with semicolons `;`.

Example of SAS code to create a table with missing values

```
data weight_club;
input IdNumber 1-4 Name $ 6-24 Team $ StartWeight EndWeight;
Loss = StartWeight - EndWeight;
cards;
     David Shaw         red 189 165
1049 Amelia Serrano     . 145 124
1219 Alan Nance         red . 192
1246 Ravi Sinha         yellow 194 .
1078 Ashley McKnight    red 127 118
;
run;
```

Classwork 1

```
data company_data;
input Sr_No 1-2 Company $ 4-17 NCI_2021 NCI_2020 NCI_2019 NCI_2018 NCI_2017;
NCI_Avg = ( NCI_2021 + NCI_2020 + NCI_2019 + NCI_2018 + NCI_2017 ) / 5;
cards;
1  RIL           74257 -143583 -53949 -59109 -54949
2  ICICI Bank    -54185.5 -36945.4 -24040.8 -38965.6 7000.3
3  Axis Bank     -54179.7 -9667.6 -18748.5 -10252.7 -12632.7
4  IOCL          -22154 -26882.4 -20771.5 -15778.7 -14733.9
5  Tata Steel    -13008.5 -17634.7 -16350 -12273.4 -3956.4
6  JSW Steel     -2609 -19092 -11432 -6134 -6284
7  HDFC Life     -8995.29 -7782.02 -10185.6 -4422.69 -5106.26
8  ICICI Pru     -5089.82 -10802.2 -7562.81 -5391.74 -699.01
9  HDFC          -8499.78 -5854.23 -9951.8 -3586.61 -1397.83
10 Maruti Suzuki -7283.9 -463.9 -3538.3 -8282.1 -1397.83
11 M&M           -14563.9 -2576.44 -2549.02 -5109.71 -2856.93
12 Bajaj Finance 424.26 -9632.54 -6637.58 504.94 -3048.24
13 UltraTech     -8986.53 -3950.86 -3987.95 1896.74 -2365.12
14 Federal Bank  -3900.27 -4664.81 -3371.98 -1576.45 -2383.34
15 Adani Ports   -7966.6 -31.35 -2403.12 -550.04 -4181.08
;
run;
```

Importing Tables

Importing from text file

2021-08-30

Example SAS code to import a table from text file

```
data user_data;
infile "/home/u59242738/Data Files/Uni/DATA_column.txt";
input name $ 1-5 gender $ 6 weight 7-9 dob $ 10-19;
```

```
run;
proc print data = user_data;
```

`data` command is used to specify the table name of the imported table.

`infile` is used to specify the file path of the external data.

Importing from Excel file

2021-08-31

Example SAS code to import a table from excel file

```
proc import datafile = '/home/u59242738/Data Files/Uni/SAS Data1.xlsx'
out = user_data_2
dbms = xlsx replace;
run;
```

`proc import` command is used to import external data from an excel file.

`datafile` argument specifies the path of the file to be imported.

`out` argument is used to specify the table name of the imported data.

`dbms` argument is used to specify the dbms / file type in which case it's `xlsx`.

Example SAS code to copy a table onto a new one and add a derived column

```
data user_data_new;
set user_data_2;
Loss = StartWeight - EndWeight;
run;

proc print data = user_data_new;
```

`set` command is used to specify the table to be copied from

Classwork 2

```
proc import datafile = '/home/u59242738/Data Files/Uni/SAS Data2.xlsx'
out = company_data_2
dbms = xlsx replace;
run;

data company_data_2;
set company_data_2;
NCI_Avg = (NCI_2021 + NCI_2020 + NCI_2019 + NCI_2018 + NCI_2017) / 5;
NCI_Avg_2 = mean(NCI_2021, NCI_2020, NCI_2019, NCI_2018, NCI_2017);
run;
```

```
proc print data = company_data_2;
```

Importing from CSV file

Example SAS code to import data from csv using 'infile'

```
data user_data_c;
infile "/home/u59242738/Data Files/Uni/DATA_commas.csv" dsd;
input name $ gender $ age weight $;
run;
proc print data = user_data_c;
```

`dsd` argument is used to specify that the file uses atandard delimiter.

Example SAS code to import data from csv using 'proc import'

```
proc import datafile = "/home/u59242738/Data Files/Uni/DATA_commas.csv"
out = user_data_c_2
dbms = csv replace;
run;
proc print data = user_data_c_2;
```

Importing with custom delimiters

Example SAS code to import data from csv using 'infile' and custom delimiter

```
data sem_dat;
infile "/home/u59242738/Data Files/Uni/Bank_full_semi colon.txt"
dlm = ";"
firstobs = 2;
input age job $ marital $ education $ default $ balance housing $ loan $
contact $ day month $ duration campaign pdays previous poutcome $ y $;
run;
proc print data = sem_dat;
```

`dlm = ";"` argument is used to specify the delimiter which the data file uses, which in this case is `;` (alternate command : `delimiter`).

`firstobs = 2` argument is used to specify that the observations begin from the second row.

Example SAS code to import data from csv using 'proc import' and custom delimiter

```
proc import
datafile = "/home/u59242738/Data Files/Uni/Bank_full_semi colon.txt"
out = sem_dat_2
```

```
dbms = csv replace;
delimiter = ";";
getnames = yes;
run;
proc print data = sem_dat_2;
```

`getnames = yes` command specifies that the first row of the data file contains the column names.

Commenting

Syntax for comments

```
* this is a single line comment;

/* this is a
multi line comment */
```

Single line comments are enclosed in `*` and `;`

Multi line comments are enclosed in `/*` and `*/`