Cheat Sheet for comprehensive SAS Certified Data Scientist

Data Manipulation and Transformation

Importing Data

- **PROC IMPORT**: Import data from various formats (CSV, Excel, etc.)

```
proc import datafile="path/to/file.csv"
            out=work.mydata
            dbms=csv
            replace;
getnames=yes;
run;
```

- **LIBNAME Statement**: Assign a library reference to a directory

```
libname mylib "path/to/directory";
```

Data Step

- **Reading Data**: Use `SET` statement to read data

```
data newdata;
    set olddata;
run;
```

- **Creating Variables**: Use `LENGTH` and `INPUT` statements

```
data newdata;
    set olddata;
    length newvar $10;
    newvar = input(oldvar, best12.);
run;
```

- **Conditional Processing**: Use `IF-THEN-ELSE` statements

```
data newdata;
    set olddata;
    if var1 > 10 then newvar = 1;
    else newvar = 0;
run;
```

- **Arrays**: Use arrays for repetitive tasks

```
data newdata;
    set olddata;
    array vars[3] var1-var3;
    do i = 1 to 3;
        vars[i] = vars[i] * 2;
    end;
run;
```

PROC SQL

- **Basic Queries**: Select, filter, and sort data

```
proc sql;
    select *
    from mydata
    where var1 > 10
    order by var2;
quit;
```

- **Joins**: Perform inner, left, right, and full joins

```
proc sql;
    select a.*, b.var
    from data1 as a
    left join data2 as b
    on a.id = b.id;
quit;
```

- **Aggregations**: Use `SUM`, `AVG`, `COUNT`, etc.

```
proc sql;
    select avg(var1) as avg_var1, count(*) as total
    from mydata;
quit;
```

Statistical Analysis

Descriptive Statistics

- **PROC MEANS**: Calculate summary statistics

```
proc means data=mydata;
    var var1 var2;
    class var3;
run;
```

- **PROC UNIVARIATE**: Detailed univariate statistics

```
proc univariate data=mydata;
    var var1;
run;
```

Inferential Statistics

- **PROC TTEST**: Perform t-tests

```
proc ttest data=mydata;
    class group;
    var var1;
run;
```

- **PROC REG**: Linear regression

```
proc reg data=mydata;
    model y = x1 x2;
run;
```

- **PROC LOGISTIC**: Logistic regression

```
proc logistic data=mydata;
    model y(event='1') = x1 x2;
run;
```

Data Visualization

PROC SGPLOT

- **Scatter Plot**: Plot two variables

```
proc sgplot data=mydata;
    scatter x=var1 y=var2;
run;
```

- **Histogram**: Visualize distribution

```
proc sgplot data=mydata;
    histogram var1;
run;
```

- **Bar Chart**: Display categorical data

```
proc sgplot data=mydata;
    vbar var3;
run;
```

PROC GCHART

- **Bar Chart**: Create bar charts

```
proc gchart data=mydata;
    vbar var3;
run;
```

- **Pie Chart**: Display proportions

```
proc gchart data=mydata;
    pie var3;
run;
```

Machine Learning

PROC HPFOREST

- **Random Forest**: Build a random forest model

```
proc hpforest data=mydata;
    input var1 var2 var3;
    target y;
run;
```

PROC HPLOGISTIC

- **Logistic Regression**: High-performance logistic regression

```
proc hplogistic data=mydata;
    model y(event='1') = x1 x2;
run;
```

PROC HPNEURAL

- **Neural Network**: Train a neural network

```
proc hpneural data=mydata;
    input var1 var2 var3;
    target y;
run;
```

Advanced Topics

Macros

- **Creating Macros**: Define and use macros

```
%macro mymacro(data=, var=);
    proc means data=&data;
        var &var;
    run;
%mend;

%mymacro(data=mydata, var=var1);
```

ODS (Output Delivery System)

- **ODS SELECT**: Control output display

```
ods select none;
proc means data=mydata;
    var var1;
run;
ods select all;
```

- **ODS EXCEL**: Export results to Excel

```
ods excel file="path/to/output.xlsx";
proc means data=mydata;
    var var1;
run;
ods excel close;
```

Tips and Tricks

Debugging

- **PUT Statement**: Print values to log

```
data newdata;
    set olddata;
    put var1= var2=;
run;
```

- **OPTIONS MPRINT**: Display macro code execution

```
options mprint;
%mymacro(data=mydata, var=var1);
```

Performance Optimization

- **INFILE Statement**: Use `DSD` and `DLM` options for better file handling

```
data newdata;
    infile "path/to/file.csv" dsd dlm=',';
    input var1 var2 var3;
run;
```

- **PROC FCMP**: Create and use functions for complex calculations

```
proc fcmp outlib=work.funcs.myfuncs;
    function myfunc(x);
        return(x * 2);
    endsub;
run;

options cmplib=work.funcs;
data newdata;
    set olddata;
    newvar = myfunc(var1);
run;
```

Shortcuts and Best Practices

Shortcuts

- **Run Entire Program**: `F3`
- **Run Selected Code**: `F4`
- **Comment/Uncomment**: `Ctrl+/`

Best Practices

- **Use Meaningful Variable Names**
- **Comment Your Code**
- **Regularly Save and Backup Your Work**
- **Use Version Control for Macros and Programs**

Additional Resources

Documentation

- **SAS Help and Documentation**: Access comprehensive documentation

```
help sas;
```

Online Communities

- **SAS Communities**: Engage with other SAS users

- [SAS Communities](https://communities.sas.com/)

Training and Certification

- **SAS Certification**: Prepare for SAS certification exams

- [SAS Certification](https://www.sas.com/en_us/certification.html)

By Ahmed Baheeg Khorshid

ver 1.0