(5) SQL SAS

```
*** SQL QUERIES;

        * Basic Query with Columns and Stats;

                proc sql;
                    create table/view <table3/view3> as
                    select distinct
                            <col1> format=<zW.d>,...,
                            <col1>/sum(<col1>) as <pct1>,...,
                            function(<col1>) as <new_col1>,...,
                            function(calculated <new_col1>) as <new_col2>,...,
                            case
                                when (<condition>) then function(<col1>)
                                when (<condition>) then function(<col1>)
                                else function(<col1>)
                            end as <new_col3>,
                            sum(<col1>),count(<col1>),avg(<col1>),std(<col1>),count(*)
as <count_rows>,...,
                            sum(<col3>='<subset>') as <count_if>,
                            sum((<col4>*(<col3>='<subset>')) as <sum_if>
                        from <table1>,<table2>,...
                        where <condition(<col1>)>,<condition(calculated
<new_col1>)>,...
                        group by <col1>,<col2>,...
                        having <condition(<col2>)>,<condition(<new_col2>)>,...
                        order by <col1>,<new_col2>,<select_num1>,<select_num2> desc
                    ;
                quit;

        * Complex Query with In-line Views;

                proc sql;
                    select <col1>,<col2>,...
                        from <table1> as a,(
                            select <col3>,<col4>
                                from <table2>
                                where <condition>
                        ) as b
                        where <condition>
                    ;
                quit;

*** SQL SUBQUERIES;

        * Noncorrelated Subquery;

                proc sql;
                    select name,salary
                        from salaries
                        where name in (
                            select name
                                from birthdays
                                where month(birthday)=2
                        )
                        having salary>=(
                            select avg(salary)
                                from salaries
                        )
                    ;
                quit;
```

```
        * Noncorrelated Subquery (Any/All & Min/Max);

                proc sql;
                    select name,salary
                        from salaries_and_positions
                        where position=1 and salary>all(
                            select salary
                                from salaries_and_positions
                                where position in (2,3,4)
                        )
                    ;
                quit;

        * Correlated Subquery;

                proc sql;
                    select name,avg(salary)
                        from salaries
                        where 'AU'=(
                            select country
                                from countries
                                where salaries.name=countries.name
                        )
                    ;
                quit;

        * Correlated Subquery (Exists/Not Exists);

                proc sql;
                    select name,position
                        from positions
                        where not exists (
                            select *
                                from sales
                                where positions.name=sales.name
                        )
                    ;
                quit;

*** SQL JOIN;
        * SQL Join (Cartesian Product);

                proc sql;
                    create table <dataset> as
                        select *
                            from <dataset_many_or_one1>,<dataset_many_or_one2>,...
                    ;
                quit;

        * SQL Join (Inner Join 2+);

                proc sql;
                    create table <dataset> as
                        select *
                            from <dataset_many_or_one1>,<dataset_many_or_one2>,...
                            where
<dataset_many_or_one1>.<var1>=<dataset_many_or_one2>.<var1>
                    ;
                quit;

        * SQL Join (Inner Join);

                proc sql;
                     create table <dataset> as
```

```
                        select a.*,b.<var2>,b.<var3>,b.<var4>
                            from <dataset_many_or_one1> as a inner join
<dataset_many_or_one2> as b
                            on a.<var1>=b.<var1>
                            where <further subsetting>
                    ;
                quit;

        * SQL Join (Left Join);

                proc sql;
                     create table <dataset> as
                        select a.*,b.<var2>,b.<var3>,b.<var4>
                        from <dataset_many_or_one1> as a left join
<dataset_many_or_one2> as b
                        on a.<var1>=b.<var1>
                        where <further subsetting>
                    ;
                quit;

        * SQL Join (Right Join);

                proc sql;
                     create table <dataset> (drop=<var1_old>) as
                        select
coalsece(dataset_many_or_one1.<var1_old>,dataset_many_or_one2.<var1_old>) as <var1>,
a.*, b.*
                            from <dataset_many_or_one1> (rename=(<var1>=<var1_old>))
as a right join <dataset_many_or_one2> (rename=(<var1>=<var1_old>)) as b
                            on a.<var1_old>=b.<var1_old>
                            where <further subsetting>
                    ;
                quit;

        * SQL Join (Full Join);

                proc sql;
                     create table <dataset> (drop=<var1_old>) as
                        select
coalsece(dataset_many_or_one1.<var1_old>,dataset_many_or_one2.<var1_old>) as <var1>,
a.*, b.*
                            from <dataset_many_or_one1> (rename=(<var1>=<var1_old>))
as a full join <dataset_many_or_one2> (rename=(<var1>=<var1_old>)) as b
                            on a.<var1_old>=b.<var1_old>
                            where <further subsetting>
                    ;
                quit;

*** SQL SET OPERATORS;

        * Except, Intersect, Union;

                proc sql;
                    select *
                        from <table1>
                        where ...
                    except/intersect/union all corr
                    select *
                        from <table2>
                        where ...
                    ;
                quit;
```

```
        * Outer Union;

                proc sql;
                    select *
                        from <table1>
                        where
                    outer union corr
                    select *
                        from <table2>
                    ;
                quit;

*** SQL MACRO VARIABLES;

        * Into Single Row;

                proc sql noprint;
                    select avg(salary),min(salary),max(salary)
                        into :<mean_salary>,:<min_salary>,:<max_salary>
                        from salaries
                    ;
                quit;

        * Into Multiple Rows (Grid);

                proc sql noprint;
                    select <col1>,<col2>,...
                        into :<a1>-:<an>,:<b1>-:<bn>,...
                        from <table1>
                        order by ...
                    ;
                %let numrows=&sqlobs;
                quit;

        * Into Multiple Rows (Delimited);

                proc sql noprint;
                    select <col1>,<col2>,...
                        into :<macro_var1> separated by '<dlm>',:<macro_var2>
separated by '<dlm>',...
                        from <table1>
                    ;
                %let numrows=&sqlobs;
                quit;

        * Data Step Analog;

                data _null_;
                    set <dataset> end=last;
                    call symputx('<macro_var>'||left(_n_),<var1>);
                    if last then call symputx('<total_macro_var>',_n_);
```