

SAS Dictionary Tables: Die Nutzung von Meta-Daten zur Lösung alltäglicher Programmieraufgaben

KSFE 30. März 2023





At Alira Health,
**our mission is to
humanize healthcare**

We complement your expertise with a full spectrum of patient-centric data and tech-enabled services to uncover opportunity, accelerate innovation, and improve outcomes for patients around the world.



**PRODUCT
DEVELOPMENT**



REGULATORY



CLINICAL



BIOMETRICS



**MARKET
ACCESS**



**MANAGEMENT
CONSULTING**



**TRANSACTION
ADVISORY**



**PATIENT
ENGAGEMENT**



**REAL-WORLD
EVIDENCE**



Tables und Views

5

Abfrage von Dictionary Tables/SASHELP Views

8

Beispiele

12

Referenzen

21

Agenda

Tables und Views	<u>5</u>
Abfrage von Dictionary Tables/SASHELP Views	<u>8</u>
Beispiele	<u>13</u>
Referenzen	<u>21</u>

Dictionary Tables und Views

Metainformation über die aktuelle SAS-Sitzung wird laufend in SAS Tabellen gespeichert.

Wir finden sie an zwei verschiedenen Orten:

> **Tables:**

- Library **DICTIONARY**

> **Views:**

- Library **SASHELP**

DICTIONARY Tables and Associated SASHELP Views

DICTIONARY Table	SASHELP View	Description
CATALOGS	VCATALG	Contains information about known SAS catalogs.
CHECK_CONSTRAINTS	VCHKCON	Contains information about known check constraints.
COLUMNS	VCOLUMN	Contains information about columns in all known tables.
CONSTRAINT_COLUMN_USAGE	VCNCOLU	Contains information about columns that are referred to by integrity constraints.
CONSTRAINT_TABLE_USAGE	VCNTABU	Contains information about tables that have integrity constraints defined on them.
DATAITEMS	VDATAIT	Contains information about known information map data items.
DESTINATIONS	VDEST	Contains information about known ODS destinations.
DICTIONARIES	VDCTNRY	Contains information about all DICTIONARY tables.
ENGINES	VENGINE	Contains information about SAS engines.
EXTFILES	VEXTFL	Contains information about known external files.
FILTERS	VFILTER	Contains information about known information map filters.
FORMATS	VFORMAT VCFORMAT	Contains information about currently accessible formats and informats.
FUNCTIONS	VFUNC	Contains information about currently accessible functions.
GOPTIONS	VGOPT VALLOPT	Contains information about currently defined graphics options (SAS/GRAPH software). SASHELP.VALLOPT includes SAS system options as well as graphics options.
INDEXES	VINDEX	Contains information about known indexes.
INFOMAPS	VINFOMP	Contains information about known information maps.
LIBNAMES	VLIBNAM	Contains information about currently defined SAS libraries.
MACROS	VMACRO	Contains information about currently defined macro variables.

Dictionary Tables und Views

Tables und Views haben etwas unterschiedliche Eigenschaften, aber den gleichen Inhalt.

> **Tables:**

- Dienen für Abfragen (Queries), können aber nicht als Viewtable geöffnet werden

> **Views:**

- Sind "nur" Kopien der Tables, können als Viewtable geöffnet werden.

MEMBERS	VMEMBER VSACCES VSCATLG VSLIB VSTABLE VSTABVW VSVIEW	Contains information about all objects that are in currently defined SAS libraries. SASHELP.VMEMBER contains information for all member types; the other SASHELP views are specific to particular member types (such as tables or views).
OPTIONS	VOPTION VALLOPT	Contains information about SAS system options. SASHELP.VALLOPT includes graphics options as well as SAS system options.
REFERENTIAL_CONSTRAINTS	VREFCON	Contains information about referential constraints.
REMEMBER	VREMEMB	Contains information about known remembers.
STYLES	VSTYLE	Contains information about known ODS styles.
TABLE_CONSTRAINTS	VTABCON	Contains information about integrity constraints in all known tables.
TABLES	VTABLE	Contains information about known tables.
TITLES	VTITLE	Contains information about currently defined titles and footnotes.
VIEWS	VVIEW	Contains information about known data views.

<https://support.sas.com/documentation/cdl/en/sqlproc/63043/HTML/default/viewer.htm#n02s19q65mw08gn140bwfdh7spx7.htm>



Tables und Views

5

Abfrage von Dictionary Tables/SASHELP Views

8

Beispiele

13

Referenzen

21

Abfrage von Dictionary Tables/SASHELP Views

Methoden und Performance:

Die Performance ist wesentlich abhängig von der Komplexität der Abfrage und der Menge der vorhandenen Metadaten.

Abfrage mit Data Step:

> Erste Abfrage:

```
data TAB1;  
  set SASHELP.VTABLE;  
  where libname='WORK';  
run;
```

```
real time    10.26 seconds  
cpu time     0.39 seconds
```

> Bei der zweiten Abfrage ist die Performance wesentlich besser:

```
real time    0.31 seconds  
cpu time     0.31 seconds
```


Abfragen von Dictionary Tables/SASHELP Views

Abfrage mit Proc sql:

> 2: Proc sql – SASHELP view:

```
proc sql noprint;
  create table TAB1 as select * from SASHELP.VTABLE
  where libname='WORK';
quit;
```

real time	0.01 seconds
cpu time	0.01 seconds

> Identische Bedingungen zum Data Step auf dem vorherigen Slide!

> 2: Proc sql – DICTIONARY Table:

```
proc sql noprint;
  create table TAB1 as select * from DICTIONARY.TABLES
  where libname='WORK';
quit;
```

real time	0.00 seconds
cpu time	0.00 seconds

→ Hier ist kein Data Step möglich!

→ Die Runtime ist sehr kompetitiv!


Abfragen von Dictionary Tables/SASHELP Views

Abfrage mit Proc sql:

> 2: Proc sql – SASHELP view:

```
proc sql noprint;
  create table TAB1 as select * from SASHELP.VTABLE
  where libname='WORK';
quit;
```

real time 0.01 seconds
cpu time 0.01 seconds




> Identische Bedingungen zum Data Step auf dem vorherigen Slide!

> 2: Proc sql – DICTIONARY Table:

```
proc sql noprint;
  create table TAB1 as select * from DICTIONARY.TABLES
  where libname='WORK';
quit;
```

real time 0.00 seconds
cpu time 0.00 seconds



→ Hier ist kein Data Step möglich!

→ Die Runtime ist sehr kompetitiv!

Queries to Dictionary Tables/SASHELP Views

Summary on Performance

- > Proc sql auf DICTIONARY Table ist meist die ökonomischere Lösung, wenn die Where-Abfrage nicht zu oft auf die Meta-Tabelle zugreift.
- > Bei mehrfachen Zugriffen bewährt sich das Erzeugen eines temporären Data Sets für die Abfrage.



Tables und Views

5

Abfrage von Dictionary Tables/SASHELP Views

8

Beispiele

13

Referenzen

21

Beispiele

> “Wie heißt nochmal diese Spalte...?”

- Man muss nicht gleich die ganze Tabelle abfragen!

```
proc sql;
  describe table
  DICTIONARY.OPTIONS;
quit;
```

- Log-Ausgabe:

- Alternative:

```
proc contents data=SASHELP.VOPTION /*short*/;
run;
```

NOTE: SQL table DICTIONARY.OPTIONS was created like:

```
create table DICTIONARY.OPTIONS
(
  optname char(32) label='Option Name',
  opttype char(8) label='Option type',
  offset num label='Offset into option value',
  setting char(1024) label='Option Setting',
  optdesc char(160) label='Option Description',
  level char(8) label='Option Location',
  optstart char(8) label='Option Set',
  group char(32) label='Option Group'
);
```

Beispiele

- > "Jetzt kann ich meine Abfrage schreiben und den aktuellen Status der Option herausfinden..."

```
1 proc sql noprint;  
2  
3   create table checkopt as select * from dictionary.options  
4  
5   where optname='SASAUTOS';  
6  
7 quit;|
```

→ Tip: Immer Uppcase für optname und die meisten anderen Variablen!

Beispiele

- > “Welche Age Variablen habe ich in meiner Datenbank, und welche Eigenschaften haben sie?”

```
proc sql;  
  create table colinfo_dict as  
  select memname, name, type, length, varnum, label, sortedby  
  from DICTIONARY.COLUMNS  
  where libname='WORK' and (index(label, 'Age') or name='AGE');  
quit;
```

	MEMNAME	NAME	TYPE	LENGTH	VARNUM	LABEL	SORTEDBY
1	T1	AGE	num	8	3		0
2	T2	AGECHDDIAG	num	8	3	Age CHD Diagnosed	0
3	T2	AGEATSTART	num	8	5	Age at Start	1
4	T2	AGEATDEATH	num	8	12	Age at Death	0

→ Tip: wieder Uppcase für Libname!

Beispiele

> Alternative: Proc contents mit einigen Zeilen mehr Code:

```
proc contents data=T1 out=colinfo_T1;
run;

proc contents data=T2 out=colinfo_T2;
run;

data colinfo_cont;
  set colinfo_T1 colinfo_T2;
  where index(label, 'Age') or name='AGE';
  keep memname name type length varnum label sortedby;
run;
```

VIEWTABLE: WORK.COLINFO_CONT							
	MEMNAME	NAME	TYPE	LENGTH	VARNUM	LABEL	SORTEDBY
1	T1	AGE	1	8	3		.
2	T2	AGEATDEATH	1	8	12	Age at Death	.
3	T2	AGEATSTART	1	8	5	Age at Start	1
4	T2	AGECHDDIAG	1	8	3	Age CHD Diagnosed	.

- Achtung: unterschiedliche Codierung der Variablen SORTEDBY!
- Manche Informationen aus proc contents sind in den DICTIONARY Tables anderen Tabellen zugeordnet, z.B. Creation Date, Modification Date, Engine, Number of Obs sind in DICTIONARY.**TABLES**, nicht in DICTIONARY.**COLUMNS**

Beispiele

> “Wie kann ich ein leeres Data Set erkennen?”

- Der Programm-Verlauf soll abhängig sein von der **Existenz** einer Tabelle: Das ist kein Problem mit der EXIST() Funktion:

```
%macro outtab(dsname);  
  %if %sysfunc(exist(&dsname.)) %then %do;  
    %put ...code for output following...;  
  %end;  
  %else %do;  
    %put Dataset !!!&dsname. does not exist!!!;  
  %end;  
%mend outtab;
```

- Aber oft existiert eine Tabelle, aber sie hat 0 Zeilen.
- Mit dieser einfachen Abfrage kann ich eine Macro Variable erzeugen. Sie kann dann in einer %if Bedingung angewendet werden.

```
proc sql noprint;  
  select nobs into: NROWS trimmed  
  from DICTIONARY.TABLES  
  where libname='WORK' and memname='EMPTYDS';  
quit;  
%put &=NROWS;
```

Beispiele

> “Welche Makro Variablen habe ich aktuell, und wie sind sie belegt?”

- Alle Macro Variablen
- Mit dem folgenden Proc sql wird ein Data Set macinfo erzeugt. Ohne das “create table...” werden sie einfach ins Log Fenster geschrieben.

```
proc sql;  
  create table macinfo as  
  select * from DICTIONARY.MACROS;  
quit;
```

> “Aber ich möchte nur meine User-defined Macro Variablen sehen!”

- SAS unterscheidet nur zwischen SCOPE=GLOBAL or AUTOMATIC. Hierzu brauchen wir eine Namenskonvention.

```
data macinfo2;  
  set SASHELP.VMACRO;  
  where name=:'_USER' or name=:'_G';  
run;
```

VIEWTABLE: WORK.MACINFO			
	SCOPE	NAME	VALUE
1	GLOBAL	SQLXITCODE	0
2	GLOBAL	SQLOBS	0
3	GLOBAL	SQLLOOPS	0
4	GLOBAL	SQLRC	0
5	GLOBAL	SQLXOBS	0
6	GLOBAL	SYS_SQL_IP_ALL	-1
7	GLOBAL	SYS_SQL_IP_STMT	
8	GLOBAL	_GDBDAT	20160630
9	GLOBAL	_GPATH	U:\scheiner-spama\
10	GLOBAL	_USERMV1	
11	GLOBAL	_USERMV2	
12	GLOBAL	_USERMV3	
13	GLOBAL	_USERMV4	
14	AUTOMATIC	AFDSID	0
15	AUTOMATIC	AFDSNAME	
16	AUTOMATIC	AFDIR	

VIEWTABLE: WORK.MACINFO2			
	SCOPE	NAME	VALUE
1	GLOBAL	_GDBDAT	20160630
2	GLOBAL	_GPATH	U:\scheiner-spama\
3	GLOBAL	_USERMV1	
4	GLOBAL	_USERMV2	
5	GLOBAL	_USERMV3	
6	GLOBAL	_USERMV4	

Beispiele

> “Ich möchte meine User-defined Macro Variablen löschen. Nur diese.”

- Schreibe alle Macro Variablen, die der Namens-Konvention entsprechen, in eine Macro Variable:

```
proc sql noprint;
  select name into: usermv separated by ' '
  from SASHELP.VMACRO
  where index(name, '_USER') or index(name, '_G');
quit;
%put &=usermv;
```

- Log: **USERMV=_GDBDAT _GPATH _USERMV1 _USERMV2 _USERMV3 _USERMV4**
- Lösche sie, einschließlich der USERMV Macro Variablen selbst:

```
%symdel &usermv. usermv;
```



Tables and Views

5

Abfrage von Dictionary Tables/SASHELP Views

8

Beispiele

13

Referenzen

21

Referenzen

Thornton, P: SAS Dictionary: Step by Step. SUGI paper 264-2011.

<http://support.sas.com/resources/papers/proceedings11/264-2011.pdf>

Lafler, KP: Exploring DICTIONARY Tables and Views. SUGI paper 070-30, 2005.

<http://www2.sas.com/proceedings/sugi30/070-30.pdf>

Accessing SAS System Information by Using DICTIONARY Tables.

<http://support.sas.com/documentation/cdl/en/sqlproc/62086/HTML/default/viewer.htm#a001385596.htm>

Sort information in DICTIONARY.TABLES.

<http://stackoverflow.com/questions/29008938/check-if-sas-datasets-are-sorted>

<http://support.sas.com/documentation/cdl/en/lrcon/62955/HTML/default/viewer.htm#a000766829.htm>

Summary of SAS Dictionary Tables and Views.

<http://www.codecraftersinc.com/pdf/DictionaryTablesRefCard.pdf>

SAS SQL Procedure User's Guide (2011).

<https://communities.sas.com/t5/Base-SAS-Programming/Getoption-argument-to-sysfunc-9-3-changes/td-p/34617>

Scheiner-Sparna R (2017): Dictionary Tables - Metadaten meines Workflows in SAS. In: Weyermann M, Minkenberg M, Muche R (Hg.): KSFE 2017. Proceedings der 21. Konferenz der SAS®-Anwender in Forschung und Entwicklung (KSFE). 1. Auflage. Herzogenrath: Shaker (Berichte aus der Statistik), S. 281–298.

Hands-On



Discussion/Questions?



Thank You



LET'S KEEP IN TOUCH



www.alirahealth.com



info@alirahealth.com



NORTH AMERICAN OFFICES

Toronto | Canada
Boston | US
San Francisco | US

EUROPEAN OFFICES

Vienna | Austria
Paris, Bordeaux | France
Munich | Germany
Bologna, Milan, Verona | Italy

Zevenbergen | Netherlands
Barcelona | Spain
Basel | Switzerland
London | UK

ASIAN PACIFIC OFFICES

Sydney | Australia
Singapore