

# SAS High-Performance Computing Series

---

## Teil 1 – In-Database Computing

20. SAS Club  
November 2010

Nicolas Adamek  
Christoph Wendl



THE  
POWER  
TO KNOW.

# Im Zeitalter der Datenflut ...

## 1,2 Zettabyte

Weltweite **Datenaufkommen** 2010, entspricht der Menge an digitalen Informationen, die entsteht, wenn jeder Erdenbewohner 100 Jahre lang ununterbrochen *twittern* würde.

## 15 Petabyte

Menge der pro Tag **neu generierten Informationen**. Dies entspricht dem Achtfachen der Informationen in allen US-Bibliotheken.

## +3x

“the size of the largest **data warehouse** ... triples approximately every two years.”

## 18 Months

Alle 18 Monate **verdoppelt sich das Datenaufkommen** weltweit.

## 1 Million

**Kundentransaktionen** bei Wal-Mart pro Stunde

## 2 Trillion ( $10^{12}$ )

Frost & Sullivan's "2004 Healthcare Storage Report" predicts that by 2010, **medical centers** will need to be equipped to hold almost 1 billion terabytes of data, or the equivalent of **2 trillion file cabinets** worth of information.

## Doppelt

Laut IDC wächst die Menge an **unstrukturierten Daten** doppelt so schnell wie in Datenbanken gespeicherte strukturierte Daten

# ... brauchen wir High-Performance Lösungen, die damit umgehen können

- **Was ist High-Performance Computing?**

- *„**Hochleistungsrechnen** (englisch: high-performance computing – HPC) ist ein Bereich des computergestützten Rechnens. Er umfasst alle Rechenarbeiten, deren Bearbeitung einer **hohen Rechenleistung** oder **Speicherkapazität** bedarf.“*

Quelle: de.wikipedia.org

- **Einsatzmöglichkeiten** von SAS in einer ganz neuen Dimension

- Hoch-performante und skalierbare Durchführung **analytisch komplexer und/oder Daten-intensiver Verarbeitungsaufträge**
- Extreme Reduzierung von Verarbeitungszeiten bis hin zur Anwendung komplexer Analyseverfahren in nahezu **Echtzeit**

# SAS High-Performance Computing Initiative

## SAS High-Performance Computing Lösungen

### SAS In-Database Computing

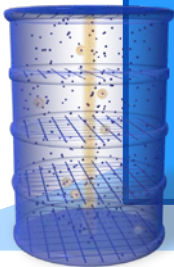
- Von SAS gesteuerte Analysen und Verarbeitungen direkt in den Datenbank-Systemen

### SAS Grid Manager

- Skalierbare Lastverteilung von SAS Prozessen und SAS Analysen in einem Rechnerverbund

### SAS In-Memory Analytics

- Verteilung von komplexen SAS Analysen und deren Daten zur In-Memory Verarbeitung in einem Rechnerverbund



# SAS® In-Database Computing

## Was ist die SAS IDC Initiative?

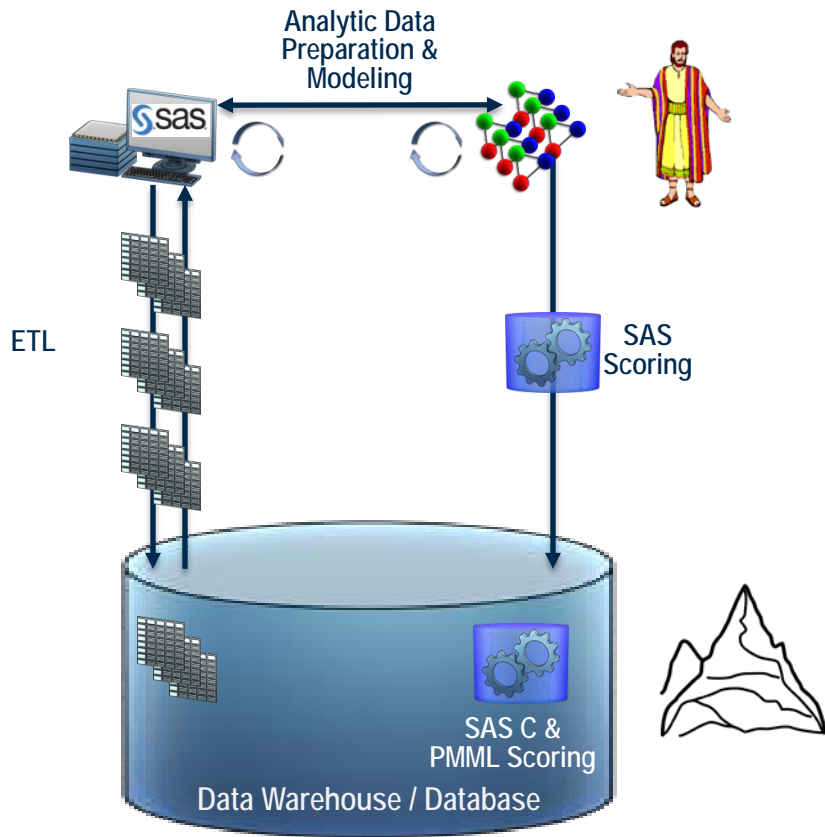


- Ziel der Initiative ist es **SAS** mit verschiedenen **Datenbank-Systemen tiefer zu integrieren**
- ... und damit **SAS Verarbeitungen** und **Analytik** im Zusammenspiel **noch schneller zu machen**, vor allem durch die Vermeidung unnötiger Datentransfers und des **Ausnutzen hoch-performerter Datenbanken und Data Warehouse Appliances (MPP)**
- Seit dem Beginn der SAS In-Database Computing Initiative **2007** wurde und wird das Spektrum kontinuierlich weiterentwickelt
- Derzeit arbeiten wir mit folgenden **Hersteller-Partnern**:
  - Teradata, Oracle, IBM, Netezza, AsterData, HP und Greenplum

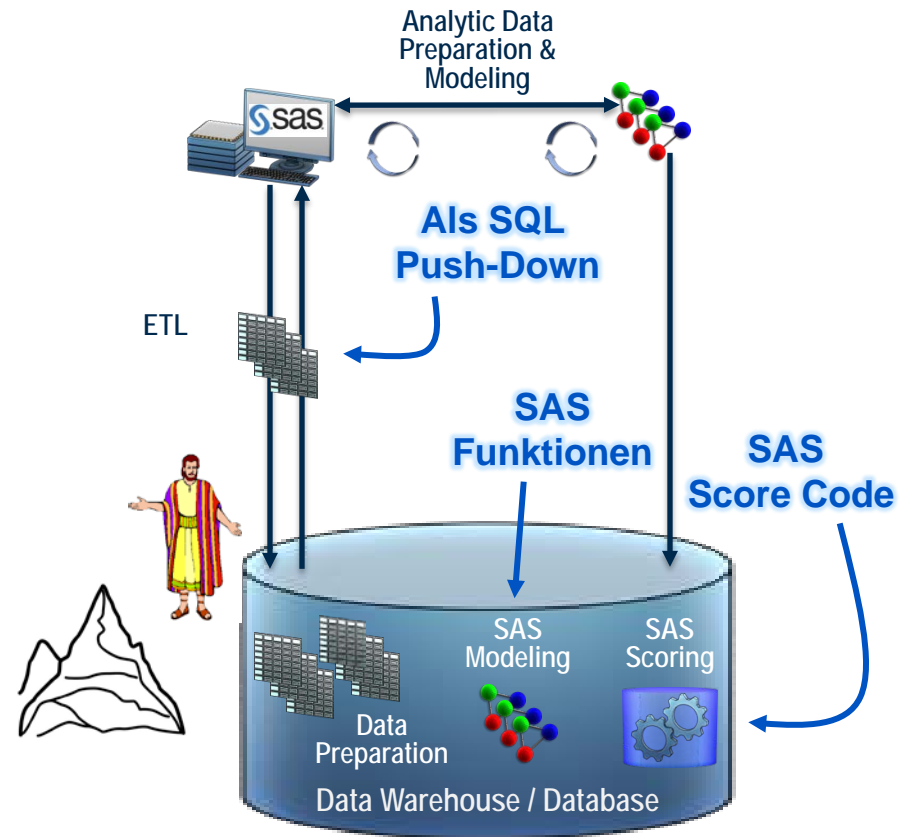
# SAS® In-Database Computing

## Architektur-Vergleich – was haben wir implementiert?

### Traditional Architecture



### In-Database Architecture



# SAS® In-Database Computing Technologien

## Data Integration

1

- **SAS Data Integration:** Engere Integration mit Datenbanken durch umfangreiche SQL Push-Down Funktionen und eigenen Transformationen für ausgewählte Datenbank-Systeme
- **SAS/ACCESS:** Stellt native Verbindungen zwischen SAS und vielen Datenbanken-Systemen zur Verfügung, und bietet damit implizites und explizites SQL Pass-Through

# In-Database Computing in SAS® Data Integration

Release Status: SAS® 9.2 M3

- **SAS/ACCESS® 9.2 für Relational Databases**
  - Optimierung von (PROC) SQL Verarbeitung durch **implizites Pass-Through (IPT)** (Push-Down) in das datenhaltende Datenbank-System und Reduktion der Datenübertragung:
    1. **SQL Aggregations-Funktionen** (MIN, MAX, AVG, MEAN, FREQ, N, SUM, und COUNT) werden durch das ANSI SQL Äquivalent im Datenbank-System ausgeführt
    2. **Ausgewählte SAS Funktionen** werden in Datenbank-Äquivalente übersetzt. Die Liste der Funktionen ist Datenbank-spezifisch, **z.B. für Oracle**: ABS, ARCOS (ACOS), ARSIN (ASIN), ATAN, CEIL, COS, COSH, DATEPART, DATETIME (SYSDATE), DTEXTDAY, DTEXTMONTH, TEXTYEAR, EXP, FLOOR, LOG, LOG10, LOG2, LOWCASE (LCASE), SIGN, SIN, SINH, SOUNDEX, SQRT, STRIP (TRIM), TAN, TRANSLATE, TRIMN (RTRIM), UPCASE (UPPER), COALESCE\*, DAY (EXTRACT)\*, MONTH (EXTRACT)\*, YEAR (EXTRACT)\*

\*) ab Oracle 9i

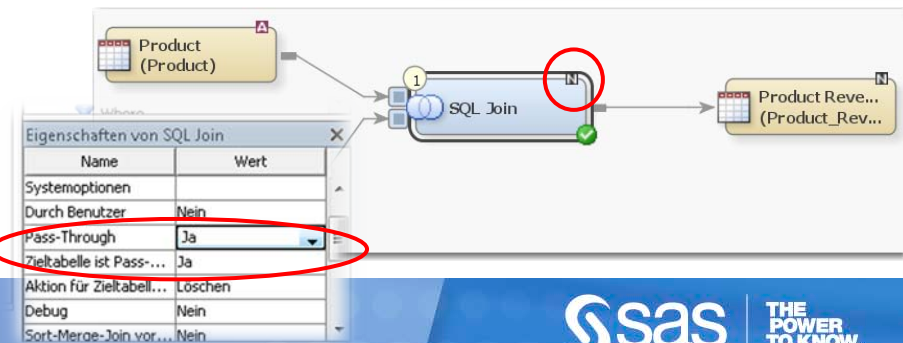
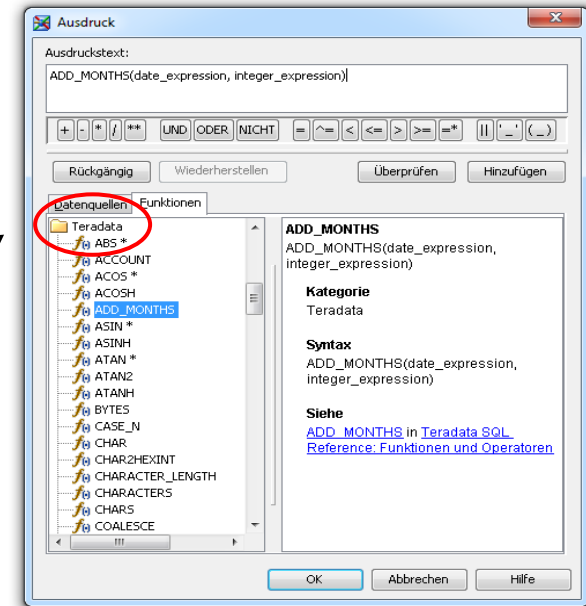


# In-Database Computing in SAS® Data Integration

- **SAS/ACCESS® 9.2 für Relational Databases**
  - Optimierung von (PROC) SQL Verarbeitung durch **implizites Pass-Through (IPT)** (Push-Down) in das datenhaltende Datenbank-System und Reduktion der Datenübertragung:
    3. **IPT Funktions-Katalog** (SQL Dictionary) ist erweiterbar
    4. Durchreichen von **Table-Joins**, **WHERE-Bedingung** und **UNION/DISTINCT** in das Datenbank-System, sofern möglich (abhängig vom jeweiligen Datenbank-System)
  - Wenn IPT nicht oder nur teilweise möglich, dann werden die Daten für die fehlenden „Operationen“ zum SAS Server übertragen
  - Optimierung der Datenbank-Interaktion durch **explizites Pass-Through (EPT)** von Datenbank-Statements innerhalb von SAS Programmen mittels:  
`PROC SQL ... CONNECT TO ... EXECUTE`

# In-Database Computing in SAS® Data Integration

- Mit **SAS® Data Integration Studio**:
  - (PROC) SQL basierte Transformationen nutzen **IPT** Funktionalität der jeweiligen SAS/ACCESS® Engine
  - **Teradata Funktionen** stehen im Expression Builder zur Verfügung
    - » Expression Builder ist auch erweiterbar!
  - Entwickler-Unterstützung durch **visuelle Darstellung** von In-Database Computing Schritten vor der Durchführung
  - In-Database fähige **Transformationen**
    - » Extract, SQL Join, SCD (teilweise)
    - » Table Loader, Teradata Table Loader
  - **EPT** mit User-Written Code und Transformationen



# In-Database Computing in SAS® Data Integration - Anwendungsfall

Damit können DWH-Beladungsstrecken komplett als **ELT** Prozesse **mit SAS** direkt im Datenbank-System ablaufen.

## Extract

- Nur mehr benötigte Daten extrahieren und weiterverarbeiten
- Sehr oft befinden sich diese im selben Datenbank-System

## Load

- Rohdaten direkt vom Vorsystem in der Ziel-Datenbank ablegen und direkt dort transformieren
- Nutzen High-Speed Bulk-Loader der Datenbank um diese Daten zu laden

## Transform

- Transformieren der Daten mit optimierten und hochperformanten Datenbanken
- Verwenden von Datenbank-spezifischem SQL und Benutzerfunktionen

# SAS In-Database Computing Technologien

## Data Integration

1

- **SAS Data Integration:** Engere Integration mit Datenbanken durch umfangreiche SQL Push-Down Funktionen und eigenen Transformationen für ausgewählte Datenbank-Systeme
- **SAS/ACCESS:** Stellt native Verbindungen zwischen SAS und vielen Datenbanken-Systemen zur Verfügung, und bietet damit implizites und explizites SQL Pass-Through

## Foundation & Analytics

2

- **Base SAS + Module:** Ausgewählte SAS Prozeduren (PROC) werden in SQL „umgeschrieben“ und der Datenbank zur Verarbeitung übergeben

# In-Database Computing in der SAS® Foundation

Release Status: SAS® 9.2 M3

- In der aktuellen Version von SAS® 9.2 stehen folgende **Base SAS** Prozeduren als **In-Database PROCEDURES** zur Verfügung
  - PROC **FREQ**, PROC **MEANS**, PROC **SUMMARY**, PROC **TABULATE**
  - PROC **REPORT**, PROC **SORT**, PROC **RANK**
- Generieren zur Laufzeit **Datenbank-SQL** für die gesamte oder Teile der Prozedur-Verarbeitung
- Voraussetzung ist die jeweilige **SAS/ACCESS®** Engine als Verbindungsmechanismus zwischen dem SAS Server und der Datenbank
- Als Datenbank-System werden derzeit **Teradata**, **DB2** (UNIX und PC Hosts) und **Oracle** unterstützt
- Aktiviert werden die In-Database PROCEDURES durch die System (oder LIBNAME) Option **SQLGENERATION**

# In-Database Computing in der SAS® Foundation

## Was passiert unter der Motorhaube?

- 1 **OPTIONS SQLGENERATION=none;**
- 2 **LIBNAME ORA ORACLE path=xe USER='sasdemo' PASSWORD='{sas001}U0FTcHcx';**
- 3 **PROC FREQ DATA=ora.cars;**  
TABLE Origin;  
**RUN;**

### Die Prozedur FREQ

ORACLE\_2: Prepared: on connection 0  
**SELECT "ORIGIN" FROM CARS**

NOTE: Es wurden **428** Beobachtungen gelesen aus  
Datei ORA.CARS.

ORIGIN				
	Häufigkeit	Prozent	Kumulative Häufigkeit	Kumulativer Prozentwert
	158	36.92	158	36.92
	123	28.74	281	65.65
USA	147	34.35	428	100.00

# In-Database Computing in der SAS® Foundation

## Was passiert unter der Motorhaube?

- 1 **OPTIONS SQLGENERATION=****DBMS**;
- 2 LIBNAME ORA ORACLE path=xe USER='sasdemo' PASSWORD='{sas001}U0FTcHcx';
- 3 **PROC FREQ DATA=ora.cars;**  
TABLE Origin;  
**RUN;**

*Die Prozedur FREQ*

ORACLE\_3: Prepared: on connection 0  
**select COUNT(\*) as ZSQL1, case when COUNT(\*) > COUNT(TXT\_1."ORIGIN") then ' ' else MIN(TXT\_1."ORIGIN") end as ZSQL2 from CARS TXT\_1 group by TXT\_1."ORIGIN"**

ACCESS ENGINE: SQL-Anweisung wurde an das DBMS zum Abrufen von Daten übergeben.

**NOTE: SQL-Generierung wird zur Erstellung von Häufigkeits- und Kreuztabellen verwendet.**

ORIGIN				
Häufigkeit	Prozent	Kumulative Häufigkeit	Kumulativer Prozentwert	
158	36.92	158	36.92	
123	28.74	281	65.65	
147	34.35	428	100.00	

ZSQL1	ZSQL2
147	USA
123	Europe
158	Asia



# In-Database Computing in der SAS® Foundation

Release Status: 1.2 mit SAS® 9.2 M3

1

## ■ SAS® Analytics Accelerator for Teradata

- Folgende **Base SAS®**, **SAS/STAT®** und **SAS/ETS®** Prozeduren stehen zur teilweisen Verarbeitung innerhalb eines Teradata Systems zur Verfügung:
  - » PROC CORR (Base SAS), PROC CANCORR, PROC FACTOR, PROC PRINCOMP, PROC REG, PROC SCORE, PROC VARCLUS, PROC TIMESERIES (SAS/ETS)
- Keine eigene PROC Implementierung (selbe Syntax), sondern die bestehenden PROCs wurden erweitert:
  - » Generieren **dynamisch SQL Code**, der bei der Ausführung in Teradata, die ...
  - » ... von SAS bereitgestellten UDFs (**User-Defined Functions**) durchführt, z.B. SAS\_ZACORR zur Ermittlung einer **Sum-of-Squares-and-CrossProducts (SSCP)** Matrix
- **Voraussetzungen:**
  - » SAS® 9.2: Base SAS, SAS/ACCESS für Teradata, SAS/STAT® und SAS/ETS®
  - » Teradata V13 (Linux)

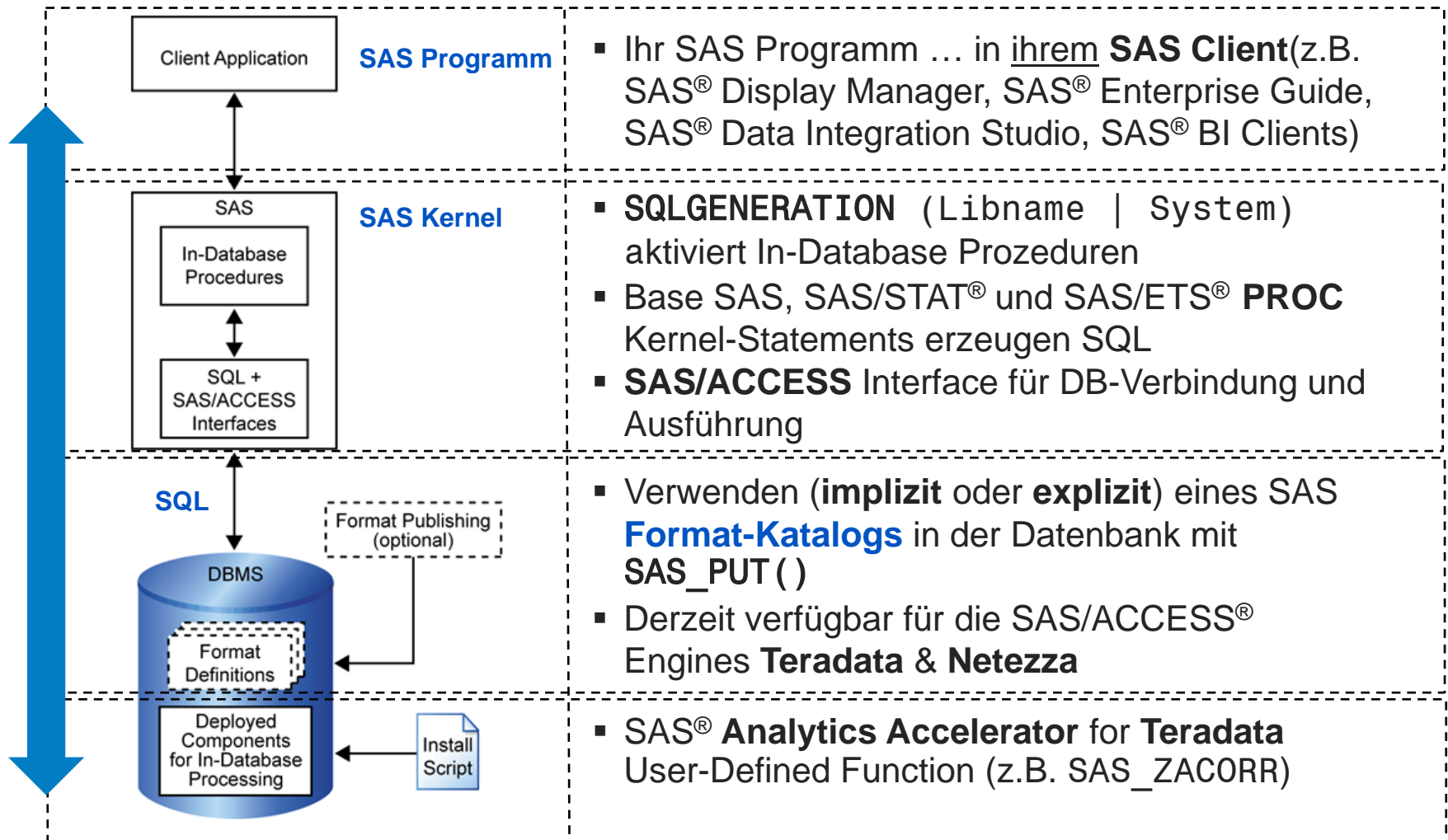
16



# In-Database Computing in der SAS® Foundation

1

Release Status: SAS® 9.2 M3



17

# SAS In-Database Computing Technologien

## Data Integration

1

- **SAS Data Integration:** Engere Integration mit Datenbanken durch umfangreiche SQL Push-Down Funktionen und eigenen Transformationen für ausgewählte Datenbank-Systeme
- **SAS/ACCESS:** Stellt native Verbindungen zwischen SAS und vielen Datenbanken-Systemen zur Verfügung, und bietet damit implizites und explizites SQL Pass-Through

## Foundation & Analytics

2

- **Base SAS + Module:** Ausgewählte SAS Prozeduren (PROC) werden in SQL „umgeschrieben“ und der Datenbank zur Verarbeitung übergeben

## Data Mining

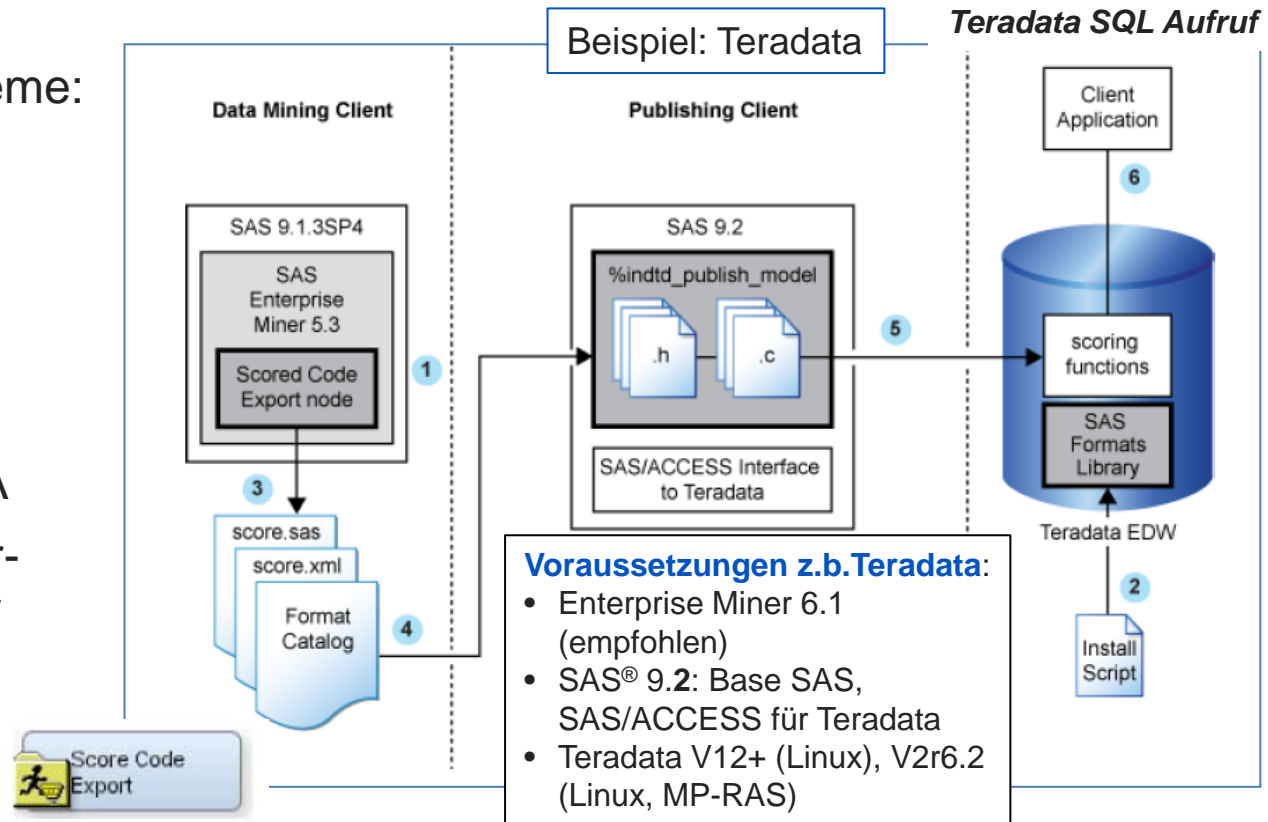
3

- **SAS Scoring Accelerator:** Erweiterung zu SAS Enterprise Miner, um Scoring Modelle innerhalb von Datenbank-Systemen zu registrieren und durchzuführen

# In-Database Computing in SAS Analytics

## ■ SAS® Scoring Accelerator:

- Erweiterung für **SAS® Enterprise Miner®**, um Scoring-Modelle direkt im Datenbank-System auszuführen – *Score Code Export* Knoten
- Integriert mit **SAS® Model Manager** mittels **Model Packages**
- Verfügbar für die Datenbank-Systeme:
  - » **DB2**
  - » **Netezza**
  - » **Teradata**
- **Unterstützt** die Übergabe von Modellen an DBA
- **Reduziert** Fehler-Anfälligkeit in der Übergabe
- **Beschleunigt** den Prozess



# SAS In-Database Computing Technologien

## Data Integration

1

- **SAS Data Integration:** Engere Integration mit Datenbanken durch umfangreiche SQL Push-Down Funktionen und eigenen Transformationen für ausgewählte Datenbank-Systeme
- **SAS/ACCESS:** Stellt native Verbindungen zwischen SAS und vielen Datenbanken-Systemen zur Verfügung, und bietet damit implizites und explizites SQL Pass-Through

## Foundation & Analytics

2

- **Base SAS + Module:** Ausgewählte SAS Prozeduren (PROC) werden in SQL „umgeschrieben“ und der Datenbank zur Verarbeitung übergeben

## Data Mining

3

- **SAS Scoring Accelerator:** Erweiterung zu SAS Enterprise Miner, um Scoring Modelle innerhalb von Datenbank-Systemen zu registrieren und durchzuführen

## Solutions

4

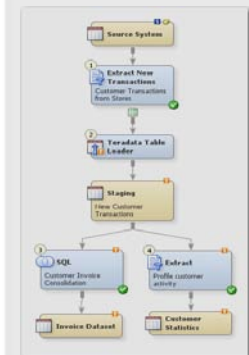
- z.B. **SAS Anti-Money Laundering:** Durchführen von Regeln (Szenarios) direkt in einem Teradata System
- **SAS Industry Solutions:** Die DDS Datenmodelle der Lösungen können in verschiedenen Datenbank-Systemen eingesetzt werden

# SAS In-Database Computing

## *End-to-End In-Database Verarbeitung mit der SAS Plattform*

### Reporting

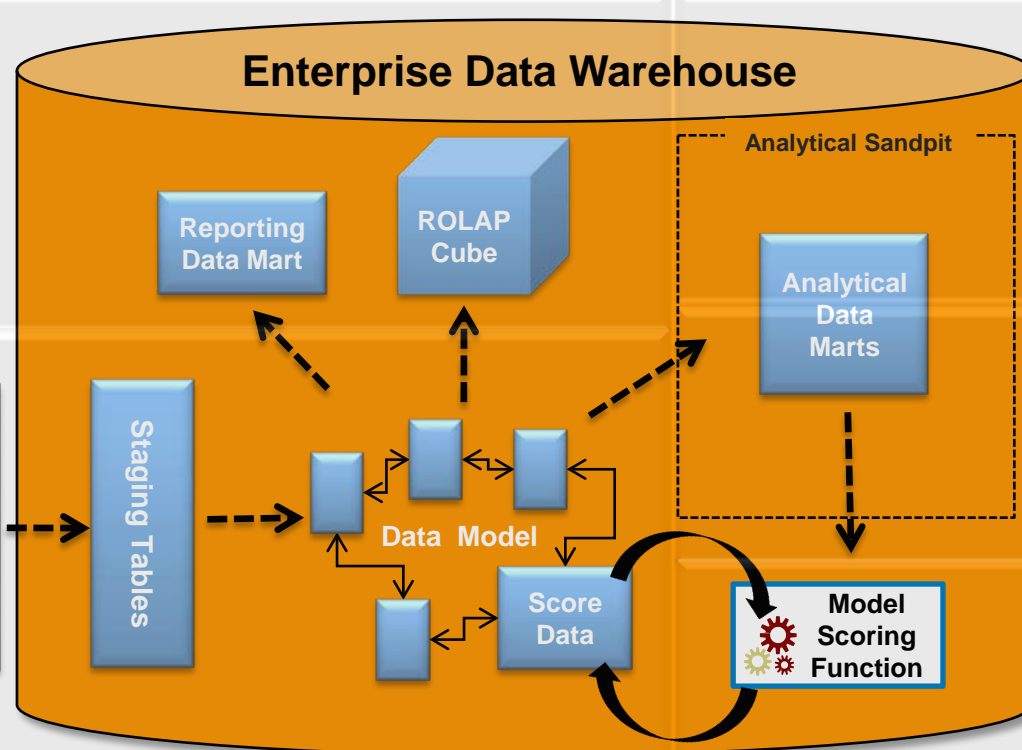
SAS Enterprise Business Intelligence



### Data Preparation

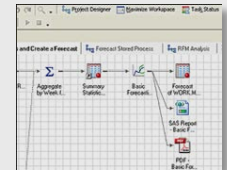
SAS Enterprise Data Integration

Data Sources



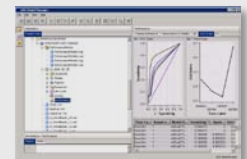
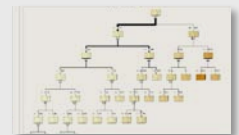
### Analytical Discovery

SAS Enterprise Guide



### Model Development

SAS Enterprise Miner



### Model Deployment

SAS Model Manager & SAS Scoring Accelerator

SAS Platform for Business Analytics

SAS Metadata Server

# Motivation für SAS In-Database Computing

## Wieso ist In-Database *Ihr Thema*?

### Einsparungen

- Es „wandern“ **weniger Daten** zwischen der Datenbank & SAS:
  - **Reduktion** der Datenmengen, die über ein Netzwerk übertragen werden muss
  - Weniger **Daten-Redundanzen** zwischen DWH und Analyse-System
- **Geringerer Storage-Bedarf** für das SAS System
- **Bessere Ausnutzung** bereits vorhandener High-Performance Datenbank- und Data Warehouse Technologien

### Akzeptanz

- Datenbank-Systeme sind in der Regel bereits vorhanden, **etabliert** und sehr oft quasi schon „bezahlt“
- Sie sind bereits Bestandteil von IT Management Prozessen (**Operations**)
- Es kann oft auf umfangreiches **Know How** zurückgegriffen werden
- Fachlogik wird dabei weiterhin sicher und zentral verwaltet (**Compliance & Governance**)
- **Geringeres Sicherheitsrisiko** (Daten „wandern“ weniger)

# Wie starte ich mit SAS In-Database Computing in meinem Unternehmen?

1. Klären Sie das Umfeld ...
  - Was für eine SAS Anwendung haben Sie? Wie nutzen Sie SAS?
  - Welche SAS Software ist im Einsatz? Wo befinden sich ihre Daten?
  - Welche Datenbank-Systeme sind im Einsatz? Gibt es bereits einen Zugriff auf diese Datenbank-Systeme?
  - Wer ist dafür zuständig (DBA)?
2. Identifizieren Sie mögliche In-Database Verarbeitungen
  - Welche SAS Prozesse lassen sich in das Datenbank-System verlagern?
  - Welche Voraussetzungen müssen erfüllt sein?
3. ... und dann „Probieren Sie es aus“!
4. Nutzen Sie unser Informationsangebot:
  - SAS Support: [SAS In-Database Technology](#), SAS/ACCESS [Produktdokumentation](#)
  - Hilfe? - sprechen Sie mit SAS Professional Services ...

# SAS In-Database Computing in der Praxis

- **Christoph Wendl** – SAS Consultant Professional Services – zeigt Ihnen...



- Beispiele:
  - **ELT** mit SAS<sup>®</sup> Data Integration Studio und SAS/ACCESS<sup>®</sup> Interface für Oracle
    - » Transformieren und Verknüpfen von Daten direkt in einer Oracle Datenbank
  - In-Database Prozedur **PROC MEANS** in SAS<sup>®</sup> Enterprise Guide und SAS/ACCESS<sup>®</sup> für Oracle
    - » Datenanalyse aus der Sicht eines Endbenutzers mit SAS Enterprise Guide und direkter Verarbeitung (intransparent) im Oracle Datenbank-System



# Beispiel 1 – ELT - SQL Join direkt in der Datenbank (Oracle)

```
proc sql;  
  connect to ORACLE  
  (  
    PATH=xe USER=sasdemo PASSWORD="{SAS002}97ECE44B0F19E0DE0F1E0782"  
  );  
  execute  
  (  
    insert into sasdemo.ORDER_FACT_QUANTITY_COUNT  
    select  
      count(BI_ORDER_STAGING.ITEM_QTY) as QUANTITY_COUNT,  
      BI_STORE_DIM.STORE_ID,  
      BI_STORE_DIM.RETAIL_OUTLET_NAME  
    from  
      sasdemo.BI_ORDER_STAGING BI_ORDER_STAGING,  
      sasdemo.BI_STORE_DIM BI_STORE_DIM  
    where  
      BI_ORDER_STAGING.STORE_ID = BI_STORE_DIM.STORE_ID  
    group by  
      BI_STORE_DIM.STORE_ID,  
      BI_STORE_DIM.RETAIL_OUTLET_NAME  
  ) by ORACLE;  
  %rcSet(&sqlrc);  
  disconnect from ORACLE;  
quit;
```

# Beispiel 2 – SQL Join: Upload von SAS Dataset in die Datenbank (Oracle)

```
LIBNAME ORA ORACLE PATH=xe SCHEMA=sasdemo USER=sasdemo PASSWORD="{SAS002}97ECE44B0F19E0DE0F1E0782" ;
```

```
proc datasets lib = ORA nolist nowarn memtype = (data view);  
  delete BI_STORE_DIM_SAS;  
quit;
```

```
/*----- Create a new table -----*/
```

```
data ORA.BI_STORE_DIM_SAS  
  (dbnull = (store_id = YES  
             retail_outlet_name = YES  
             retail_outlet_format_cd = YES  
             retail_outlet_type_cd = YES  
             store_size = YES  
             effective_from_dttm = YES  
             effective_to_dttm = YES  
             processed_dttm = YES));  
  attrib store_id length = 8 format = 12. informat = 12. label = 'Retail Outlet Location Key';  
  attrib retail_outlet_name length = $40 label = 'Retail Outlet Name';  
  attrib retail_outlet_format_cd length = $32 label = 'Retail Outlet Format Code';  
  attrib retail_outlet_type_cd length = $32 label = 'Retail Outlet Type Code';  
  attrib store_size length = 8 format = NLNUM10.2 informat = NLNUM10.2 label = 'Total Size';  
  attrib effective_from_dttm length = 8 format = DATETIME20. informat = DATETIME20.  
    label = 'Effective From Datetime';  
  attrib effective_to_dttm length = 8 format = DATETIME20. informat = DATETIME20.  
    label = 'Effective To Datetime';  
  attrib processed_dttm length = 8 format = DATETIME20. informat = DATETIME20.  
    label = 'Processed Datetime';
```


```
run;
```

```
proc append base = ORA.BI_STORE_DIM_SAS  
  data = mysas.BI_STORE_DIM_SAS force;  
run;
```

# Beispiel 2 – SQL Join: Ausführen des Joins nach Upload in die Datenbank

```
proc sql;
  connect to ORACLE
  (
    PATH=xe USER=sasdemo PASSWORD="{SAS002}97ECE44B0F19E0DE0F1E0782"
  );
  execute
  (
    insert into sasdemo.ORDER_FACT_QUANTITY_COUNT
    select
      count(BI_ORDER_STAGING.ITEM_QTY) as QUANTITY_COUNT,
      BI_STORE_DIM_SAS.store_id,
      BI_STORE_DIM_SAS.retail_outlet_name
    from
      sasdemo.BI_STORE_DIM_SAS BI_STORE_DIM_SAS,
      sasdemo.BI_ORDER_STAGING BI_ORDER_STAGING
    where
      BI_STORE_DIM_SAS.store_id = BI_ORDER_STAGING.STORE_ID
    group by
      BI_STORE_DIM_SAS.store_id,
      BI_STORE_DIM_SAS.retail_outlet_name

  ) by ORACLE;
  %rcSet(&sqlrc);
  disconnect from ORACLE;
quit;
```



# Beispiel 3 – PROC MEANS

- Project Log
- LIBNAME Option: **SQLGENERATION=DBMS**

```
options sastrace = ',,,d' sastraceloc = saslog;

libname ORA ORACLE path=xe sqlgeneration=dbms
       schema='sasdemo' user='sasdemo' password='{sas001}U0FTcHcx';

PROC MEANS data=ORA.BI_ANALYSIS_DATA
  min max sum nonobs;
  var TOTAL_LINE_ITEM_SALE_AMT;
  class YEAR / order=UNFORMATTED ascending;
  class PRODUCT_LINE / order=UNFORMATTED ascending;
  class CUSTOMER_TYPE / order=UNFORMATTED ascending;

output out=WORK.MEANS_CODE;
RUN;
```

# Tipps & Tricks (1)

## ■ Allgemeine Tips:

- SASTRACE statement um den SQL Code, der an die Datenbank geschickt wird zu sehen: *options sastrace=',,,d' sastraceloc=saslog nostsuffix;*
- SASTRACE statement um die Durchführungszeit der einzelnen Schritte zu sehen: *options sastrace=',,,ts' sastraceloc=saslog nostsuffix;*
- Funktionen verwenden, die die Datenbank verarbeiten kann (z. Bsp.: TRIM kann nicht in Oracle verarbeitet werden, TRIMN schon), SAS/ACCESS Dokumentation überprüfen!
- Primary Index: SAS-Datasets brauchen keinen, manche DBMS aber schon (Beispiel Teradata zwecks Verteilung über die Knoten)
- System Option **DBIDIRECTEXEC**, leitet alle Statements die möglich sind an die Datenbank weiter
- SAS Makro statt einer SAS Funktion, um IPT zu gewährleisten:

```
proc sql;  
  select account, ship_date from dbms.neworders where ship_date < today() + 7 ;  
quit;
```

```
%let target_date=%eval(%sysfunc(today()) + 7 );  
proc sql;  
  select account, ship_date from dbms.neworders where ship_date < &target_date ;  
quit;
```

# Tipps & Tricks (2)

- **SAS Data Integration Studio:**

- ELT: Output Tabellen direkt in die Datenbank umleiten, Property kann auf der Tabelle, der Transformation, dem Job oder als globale Option gesetzt werden
- Check Database Processing

- **Mögliche Ursachen, wenn kein impliziter Pass-Through gelingt:**

- Zu einem Datenbankfeld fehlt im DBMS das SAS-Format
- Die gewählte SAS Funktion gibt es nicht (in *der* Form) im DBMS
- Es wird eine Transformation genutzt, die kein SQL produziert (z.B. PROC Transpose)
- Es wird auf eine Tabelle referenziert, die nicht im DBMS liegt (gerne auch eine WORK-Tabelle, hinter den Kulissen...)



Vielen Dank für Ihre Aufmerksamkeit

Haben Sie Fragen?