



2009 Annual Conference  
New Orleans, LA

## Inside an Oracle Clinical Validation Procedure

Steven Rifkin  
Sr. Director of Consulting  
BioPharm Systems



# Topics

- What is an Oracle Clinical Procedure
- Review of PL/SQL concepts
  - Cursors and Package Structure
- Oracle Clinical Cursors
- Structure of Oracle Clinical MAIN Procedure
- Fetch Limitation Techniques and effect on program structure
  - Correlation
  - Qualifying Expressions
  - Where Clause Extensions
  - Visit Range limits
- Performance Considerations

# What is an Oracle Clinical Procedure?

Definition => Validation Procs => Procedures

The screenshot shows a software window titled "Maintain Study Validation Procedure (Study: ASD0553\_00)". The main area is labeled "Procedure Definitions" and contains a table with the following columns: Procedure Name, Domain, Version, Status, Exec Context, Description, and Category. The first row is populated with the following data:

Procedure Name	Domain	Version	Status	Exec Context	Description	Category
V_DEMOG_OTHER_SPEC	ASD0553_00	0	A	OFF-LINE	Check Choice OTHER is consistent with OTHER_SPECIFY	1-EVENT 1-DC

At the bottom of the window, there is a horizontal scroll bar and a row of buttons: Exit, Save, Change Study, Q-Grps, Details, Sys Vars, User Vars, and Custom Code.

# What is an Oracle Clinical Procedure?

Definition => Validation Procs => Procedures

Procedure Name	Domain	Version	Status	Exec Context	Description	Category
V_DEMOG_OTHER_SPEC	ASD0553_00	0	A	OFF-LINE	Check Choice OTHER is consistent with OTHER_SPECIFY	1-EVENT 1-DC

# What is an Oracle Clinical Procedure?

- Procedure templates are completed
- Procedure is “generated” and

```
Editor
/*=====*/
create or replace package RXCPD_1_0 as

/* cursor for getting DEMOGRAPHY Production */
cursor A_CUR(
  I_PATIENT_POSITION_ID in RECEIVED_DCMS.PATIENT_POSITION_ID%TYPE,
  I_BEGIN_VISIT_NUMBER in RECEIVED_DCMS.VISIT_NUMBER%TYPE := RXCPDSTD.C_BEGIN_VISIT_NUMBER,
  I_END_VISIT_NUMBER in RECEIVED_DCMS.VISIT_NUMBER%TYPE := RXCPDSTD.C_END_VISIT_NUMBER) is
  select /*+ ordered use_nl(RDCM RES)
    index(RDCM RECEIVED_DCM_UK2_IDX) */
    RDCM.RECEIVED_DCM_ID,
    RDCM.RECEIVED_DCM_ENTRY_TS,
    RDCM.INVESTIGATOR_ID,
    RDCM.SITE_ID,
    RDCM.DCM_ID,
    RDCM.DCM_SUBSET_SN,
    RDCM.DCM_DATE,
    RDCM.DCM_TIME,
    RDCM.ACTUAL_EVENT_ID,
    RDCM.LAB_ID,
    RDCM.LAB LAB,
    RDCM.LAB_RANGE_SUBSET_NUM,
    RDCM.QUALIFYING_VALUE,
    RDCM.SUBEVENT_NUMBER,
    RDCM.CLIN_PLAN_EVE_ID,
    RDCM.CLIN_PLAN_EVE_NAME,
    RDCM.VISIT_NUMBER,
    RES.REPEAT_SN,
    max(decode(RES.DCM_QUESTION_ID,4001,substr(RES.VALUE_TEXT,1,20),null)) RACE,
    max(decode(RES.DCM_QUESTION_ID,4001,substr(RES.EXCEPTION_VALUE_TEXT,1,20),NULL)) RACE$EXC_VAL,
    max(decode(RES.DCM_QUESTION_ID,4001,RES.RESPONSE_ID,NULL)) RACE$RESP_ID,
    max(decode(RES.DCM_QUESTION_ID,4001,RES.RESPONSE_ENTRY_TS,NULL)) RACE$ENT_TS,
    max(decode(RES.DCM_QUESTION_ID,4201,substr(RES.VALUE_TEXT,1,20),null)) OTHER_SPECIFY,
    max(decode(RES.DCM_QUESTION_ID,4201,substr(RES.EXCEPTION_VALUE_TEXT,1,20),NULL)) OTHER_SPECIFY$EXC_VAL,
    max(decode(RES.DCM_QUESTION_ID,4201,RES.RESPONSE_ID,NULL)) OTHER_SPECIFY$RESP_ID,
    max(decode(RES.DCM_QUESTION_ID,4201,RES.RESPONSE_ENTRY_TS,NULL)) OTHER_SPECIFY$ENT_TS
```

**A PL/SQL Program is created!**

**To get the most out of OC Procedures  
you need to understand the structure  
of the program you create**

# Review of PL/SQL Concepts

- Detailed knowledge of PL/SQL syntax is **not** required for understanding the overall program structure
  - However, knowledge of PL/SQL cursors and of the PL/SQL Package structure is required

# PL/SQL Cursors ...

In SQL\*Plus, a "select statement" retrieves an entire set of records:

```
SQL> select patient, patient_position_id from patient_positions;
```

<u>PATIENT</u>	<u>PATIENT_POSITION_ID</u>
100	5001
101	5101
102	5201
103	5301
104	5401
105	5501
106	5601
107	5701
108	5801
109	5901

**Cursors also retrieve of a set of records; but, unlike SQL\*Plus, cursors allow access to one record at a time for procedural operations on data within the record**

# PL/SQL Cursors ...

- Cursors are objects in a PL/SQL program
- To use, cursors must be
  1. Declared (to define the set of results)
  2. Opened (to initialize)
  3. Fetched (to retrieve one row for operations)
  4. Closed (to release)

# PL/SQL Cursors ...

- Cursor declaration defines the set of records to be retrieved from the database:

```
CURSOR cHeight IS  
SELECT height, height_unit  
FROM height_table WHERE  
patient='100' ORDER BY  
visit_number;
```

- Like a "file" a cursor must be opened before it can be used:

```
OPEN cHeight;
```

# PL/SQL Cursors ...

- A “fetch” on the cursor retrieves a single row from the set of selected records and places the result into program variables:

```
FETCH cHeight INTO height_var, height_unit_var;
```

- Cursors are closed to release memory and perform internal cleanup:

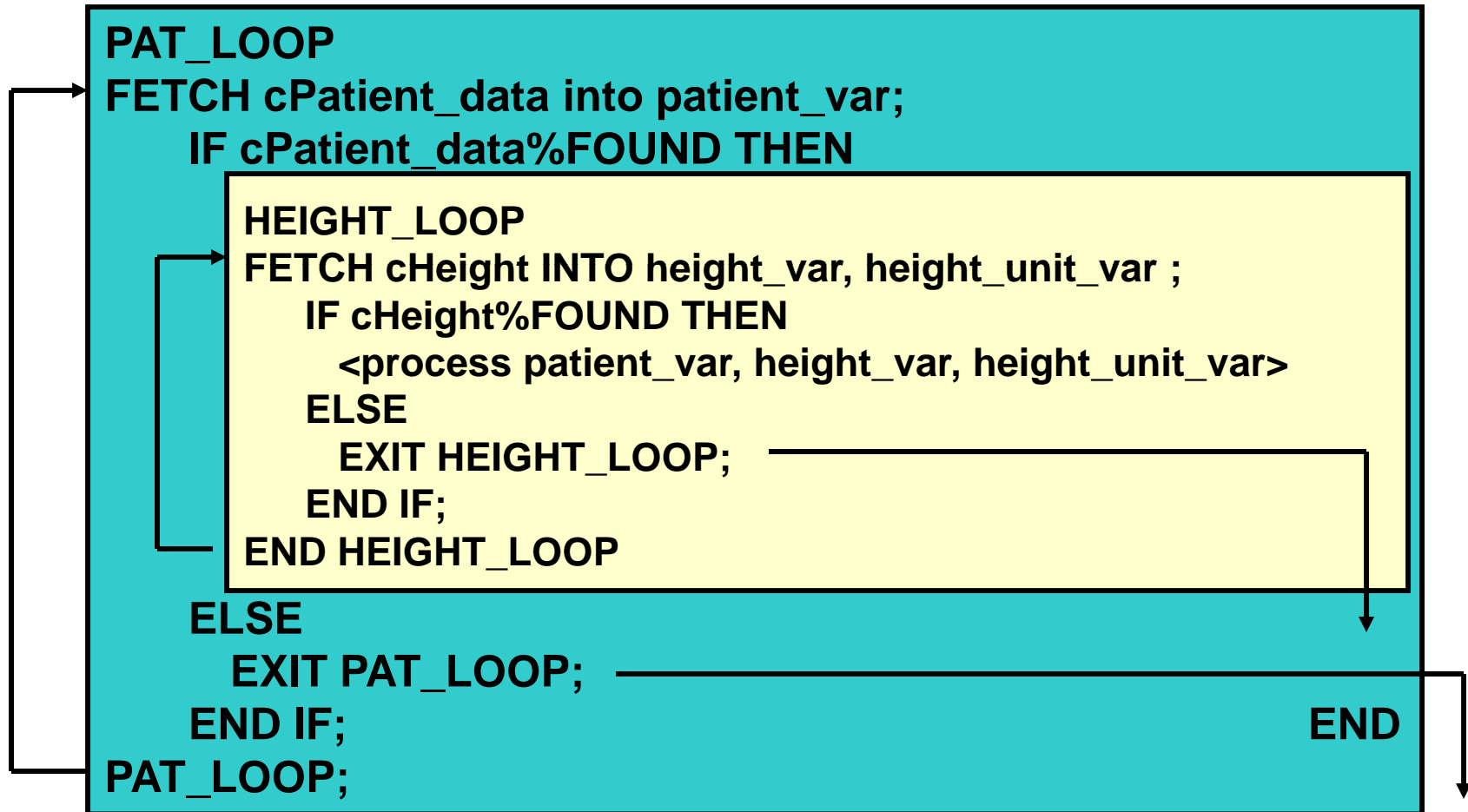
```
CLOSE cHeight;
```

# PL/SQL Cursors

- Cursors are often fetched in a “loop” which processes the record:

```
LOOP
  FETCH cHeight INTO height_var, height_unit_var ;
  IF cHeight%NOTFOUND THEN
    EXIT LOOP;
  ELSE
    <process current height_var, height_unit_var>
  END IF;
END LOOP;
```

# Nesting PL/SQL Cursors



# Cursors used by Oracle Clinical

- Patient Information Cursor
  - Retrieves a record with all the information recorded for a single patient
- DCM Cursor(s)
  - A DCM Cursor for each “alias” listed on the Procedure Question Group Form
  - Retrieves the DCM responses (for the questions on the Procedure Question Screen) and DCM Header Information

# Fields Retrieved by the Patient Cursor

- V\_CLINICAL\_STUDY\_VERSION\_ID
- V\_DATA\_MODIFIED\_FLAG
- V\_PROCEDURE\_ID
- V\_PROCEDURE\_VERSION\_SN
- V\_PROCEDURE\_TYPE\_CODE
- V\_USERNAME
- V\_DEBUG
- V\_LAST\_BATCH\_TS
- V\_CURRENT\_LOCATION
- V\_MODE
- V\_LAB\_DEPENDENT\_FLAG
- REPORTED\_SEX
- REPORTED\_BIRTH\_DATE
- PATIENT\_POSITION\_ID
- CLINICAL\_STUDY\_ID
- REPORTED\_DEATH\_DATE
- PATIENT
- INVESTIGATOR\_ID
- SITE\_ID
- EARLY\_TERMINATION\_FLAG
- PATIENT\_ENROLLMENT\_DATE
- CLINICAL\_SUBJECT\_ID
- INCLUSION\_EXCLUSION\_DATE
- REPORTED\_PATIENT\_REFERENCE
- REPORTED\_INITIALS
- REPORTED\_DATE\_LAST\_PREGNACY
- FIRST\_SCREENING\_DATE
- TERMINATION\_DATE

# Fields in a DCM Cursor

- ACTUAL\_EVENT\_ID
- CLIN\_PLAN\_EVE\_ID
- CLIN\_PLAN\_EVE\_NAME
- DCM\_DATE
- DCM\_ID
- DCM\_SUBSET\_SN
- DCM\_TIME
- INVESTIGATOR\_ID
- LAB
- LAB\_ID
- LAB\_RANGE\_SUBSET\_NUM
- QUALIFYING\_VALUE
- RECEIVED\_DCM\_ENTRY\_TS
- RECEIVED\_DCM\_ID
- REPEAT\_SN
- SITE\_ID
- SUBEVENT\_NUMBER
- VISIT\_NUMBER

# Structure of PL/SQL Packages

- Generation of an Oracle Clinical Procedure results in a PL/SQL Package
- PL/SQL Packages contain one or more functions and/or procedures and have two sections
  - Specification
    - Lists all procedures and functions in the package
    - Declares all cursors and variables to be available to all the functions and procedures (i.e. the “common” area)
  - Body
    - Contains all the procedural code for the procedures and functions listed in the specification

# Specification for a Package in Oracle Clinical

```
CREATE PACKAGE rxcpd_xxx_xx as
```

```
CURSOR 1 is ....
```

```
CURSOR 2 is ....
```

```
■ ■ ■
```

```
CURSOR n is ....
```

```
PROCEDURE main (... ..);
```

```
PROCEDURE insert_discrepancy(... ..);
```

```
PROCEDURE exception_handling (... ..);
```

```
END rxcpd_xxx_xx;
```

Package is named with internal id and version numbers

Definition and number of cursors declared depends on the number of “aliases” on the Procedure Question Group form.

Three PL/SQL procedures are part of each Oracle Clinical Package

# Body for a Package in Oracle Clinical

```
CREATE PACKAGE BODY rxcpd_xxx_xx as
```

```
  PROCEDURE main (... ..);
```

```
    <Program Code >
```

```
  END main;
```

```
  PROCEDURE insert_discrepancy(...
```

```
    < Program Code>
```

```
  END insert_discrepancy;
```

```
  PROCEDURE exception_handling (... ..);
```

```
    <Program Code>
```

```
  END exception_handling;
```

```
END rxcpd_xxx_xx;
```

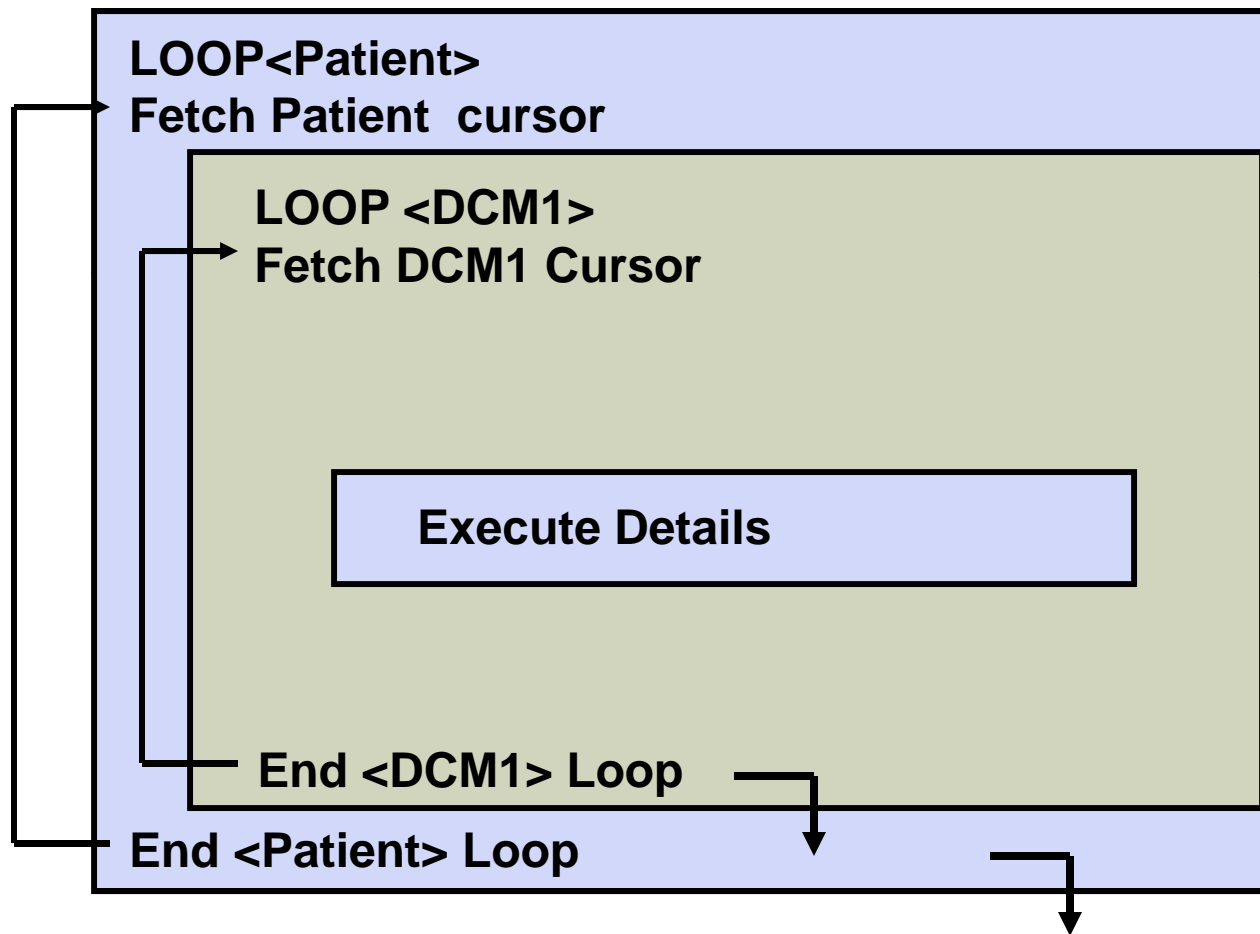
Program code in the MAIN procedure depends on options chosen on the Procedure definition forms.

INSERT\_DISCREPANCY and EXCEPTION\_HANDLING procedures call Oracle Clinical built-in packages and cannot be readily changed.

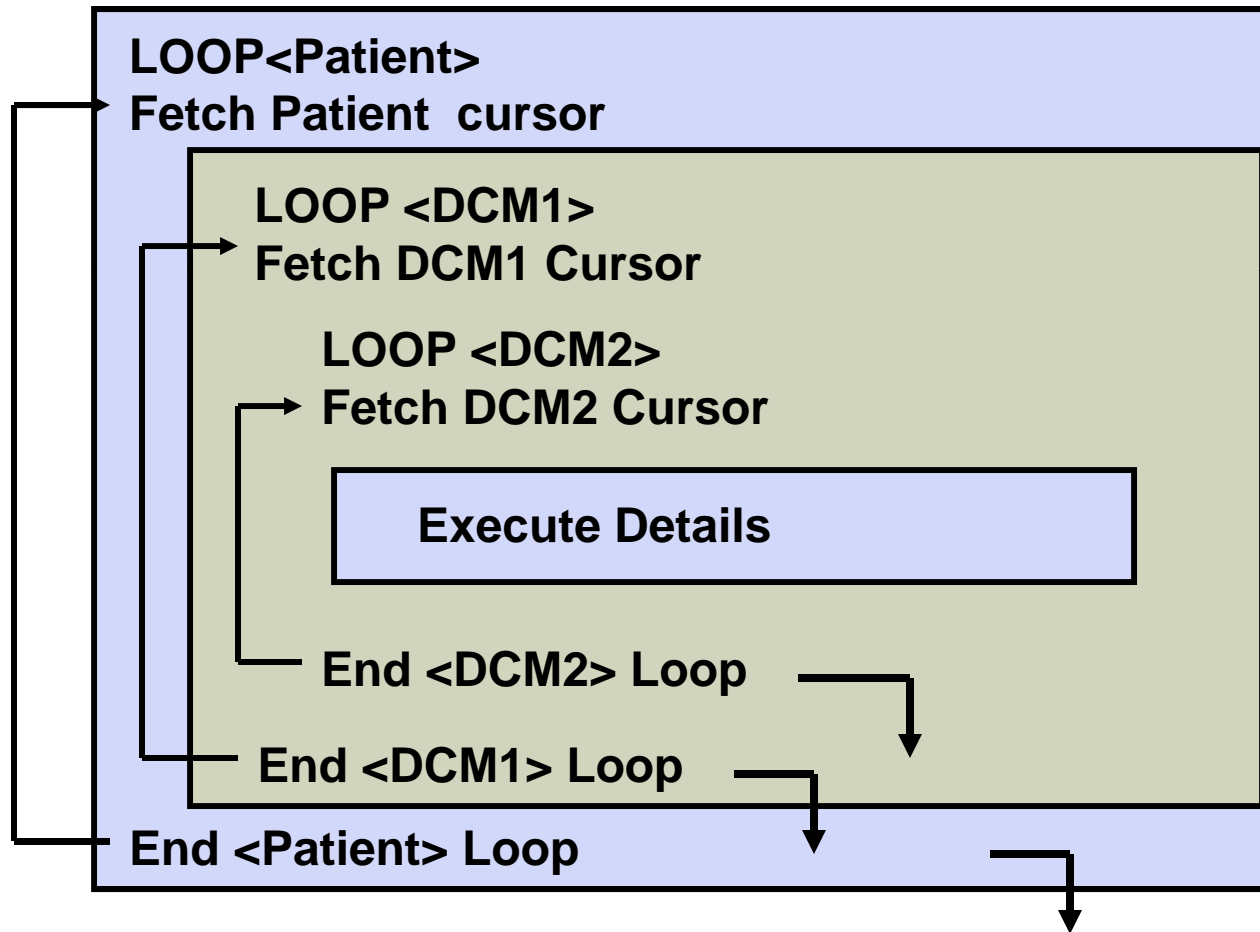
# Simplified Structure of a MAIN Procedure ...

- MAIN Procedure is a series of nested loops
- For a simple procedure, outermost loop fetches the Patient information
  - Next Loop fetches DCM information as defined by the first Procedure Question Group definition
- After fetching the innermost loop, the expression details are evaluated

# Simplified Structure of a MAIN Procedure



# Simplified Structure of a MAIN Procedure



# Some of the Techniques to Limit Retrievals

- Correlate inner loop with a more outer loop
  - on Event
  - on Qualifying Question Value
  - on Questions
- Qualifying Expressions
- Where Clause Extension
- Limit visits retrieved by the cursor

# Correlations

Correlation limits the fetches for inner cursors based on data obtained by a fetch in a more outer cursor

- Can correlate on Events
  - Inner cursor retrieves only a visit retrieved by the outer correlated cursor
- Can correlate on Qualifying Questions
  - Inner cursor only retrieves DCMs which have the same value for the DCM Qualifying Question
- Can correlate on Questions
  - Inner cursor fetches only if a question's value is equal to a question's value in the outer cursor

# Correlation on Event ...

## Retrieval with No Correlation

Fetch data for Patient 100

Fetch SYSTOLIC\_BP from VITALS DCM for Visit 1, Pt 100

Fetch HYPERTENSIVE\_YN from HYPERTENSION DCM for **Visit 1**, Pt 100  
Test

Fetch HYPERTENSIVE\_YN from HYPERTENSION DCM for **Visit 2**, Pt 100  
Test

Fetch SYSTOLIC\_BP from VITALS DCM for Visit 2, Pt 100

Fetch HYPERTENSIVE\_YN from HYPERTENSION DCM for **Visit 1**, Pt 100  
Test

Fetch HYPERTENSIVE\_YN from HYPERTENSION DCM for **Visit 2**, Pt 100  
Test

# Correlation on Event ...

## Retrieval with Correlation on Event

Fetch data for Patient 100

Fetch SYSTOLIC\_BP from VITALS DCM for Visit 1, Pt 100

Fetch HYPERTENSIVE\_YN from HYPERTENSION DCM for **Visit 1**, Pt 100  
Test

**X** *Fetch HYPERTENSIVE\_YN from HYPERTENSION DCM for Visit 2, Pt 100  
Test*

Fetch SYSTOLIC\_BP from VITALS DCM for Visit 2, Pt 100

**X** *Fetch HYPERTENSIVE\_YN from HYPERTENSION DCM for Visit 1, Pt 100  
Test*

Fetch HYPERTENSIVE\_YN from HYPERTENSION DCM for **Visit 2**, Pt 100  
Test

# Correlation on Event ...

Definition => Validation Procs => Procedures, Press [Q-Groups]

Maintain Study Validation Procedure With Edit Option (Study: ASD0553\_00)

Question Groups for V\_HYPERTENSION

Alias	der -----> Extension	First/Last Event Only	Single Repeat Only?	Create Place Holder ?	<----- Correlate With -----> Alias	Event	Qualif. Quest
A	RES.REPEA		<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>
B	RES.REPEA		<input type="checkbox"/>	<input type="checkbox"/>	A	ACTUAL	<input type="checkbox"/>
			<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>
			<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>
			<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>
			<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>
			<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>
			<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>
			<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>
			<input type="checkbox"/>	<input type="checkbox"/>			<input type="checkbox"/>

Back Save Questions Correlating Questions Custom Code

Alias to correlate to

Inner loops use the same event (including subevent) as retrieved for the outer (correlated) loop

# Correlation on Qualifying Question Value ...

- Assume two Qualified DCMs recorded at a Visit
  - Vitals DCM: qualified at Pre-Dose, 1 Hr Post-dose
  - EKG DCM: qualified at Pre-Dose, 1 Hr Post-dose
- Correlate on both Visit and Qualifying Question

Alias	Create Place Holder ?	Alias	Event	Qualif. Quest
A	<input type="checkbox"/>			<input type="checkbox"/>
B	<input type="checkbox"/>	A	ACTUAL ...	<input checked="" type="checkbox"/>
	<input type="checkbox"/>			<input type="checkbox"/>

# Correlation on Qualifying Question Value

Fetch data for Patient 100

Fetch SYSTOLIC\_BP from VITALS DCM for Visit 1, Pt 100, Pre-Dose

Fetch EKG from EKG DCM for Visit 1, Pt 100, Pre-dose

X Fetch EKG from EKG DCM for Visit 1, Pt 100, Post-dose

X Fetch EKG from EKG DCM for Visit 2, Pt 100, Pre-dose

X Fetch EKG from EKG DCM for Visit 2, Pt 100, Post-dose

Fetch SYSTOLIC\_BP from VITALS DCM for Visit 1, Pt 100, Post-Dose

X Fetch EKG from EKG DCM for Visit 1, Pt 100, Pre-dose

Fetch EKG from EKG DCM for Visit 1, Pt 100, Post-dose

X Fetch EKG from EKG DCM for Visit 2, Pt 100, Pre-dose

X Fetch EKG from EKG DCM for Visit 2, Pt 100, Post-dose

Fetch SYSTOLIC\_BP from VITALS DCM for Visit 2, Pt 100, Pre-Dose

X Fetch EKG from EKG DCM for Visit 1, Pt 100, Pre-dose

X Fetch EKG from EKG DCM for Visit 1, Pt 100, Post-dose

Fetch EKG from EKG DCM for Visit 2, Pt 100, Pre-dose

X Fetch EKG from EKG DCM for Visit 2, Pt 100, Post-dose

Fetch SYSTOLIC\_BP from VITALS DCM for Visit 2, Pt 100, Post-Dose

X Fetch EKG from EKG DCM for Visit 1, Pt 100, Pre-dose

X Fetch EKG from EKG DCM for Visit 1, Pt 100, Post-dose

X Fetch EKG from EKG DCM for Visit 2, Pt 100, Pre-dose

Fetch EKG from EKG DCM for Visit 2, Pt 100, Post-dose

# Correlating Questions ...

- Use responses to DCM Questions to correlate between outer and inner fetch loops
  - May want to fetch AE records where AE description matches the Indication given for ConMed

Definition => Validation Procs => Procedures, Press [Q-Groups]

Alias	DCM	DCM Domain	DCM Question Group	Aggregate?	Primary Refer?
AE	ADVERSE EVENTS	ASD0554	ADVERSE_EVENT	<input type="checkbox"/>	<input checked="" type="checkbox"/>
CM	CONMED	ASD0554	CONMED	<input type="checkbox"/>	<input type="checkbox"/>

# Correlating Questions

Definition => Validation Procs => Procedures, Press [Q-Groups],  
[Correlating Questions]

The screenshot shows a window titled 'Maintain Study Validation Procedure (Study: BP0554)' with a sub-header 'Correlating Questions for Alias CM'. The window contains a table with four columns: 'Current Group Question Name', 'Occ#', 'Other Group Question Name', and 'Occ#'. The first row has 'INDICATION' and '0' in the first two columns, and 'AE' and '0' in the last two. The second row has 'MEDICATION' and '0' in the first two columns, and 'MEDICATION' and '0' in the last two. A box labeled 'Inner Loop' points to the 'MEDICATION' row in the first column, and a box labeled 'Outer Loop' points to the 'MEDICATION' row in the third column. A callout box below the table states: 'Inner loop will fetch records only if the values of the responses to the two pairs of questions are the same!'. At the bottom of the window are 'Back' and 'Save' buttons.

Current Group Question Name	Occ#	Other Group Question Name	Occ#
INDICATION	0	AE	0
MEDICATION	0	MEDICATION	0

**Inner Loop**      **Outer Loop**

Inner loop will fetch records only if the values of the responses to the two pairs of questions are the same!

Back   Save

**How does correlation  
change the generated  
Program?**

# Effect of Correlation on Event on Outer Cursor

```
/*=====*/
create or replace package RXCPD_24901_0 as

/* cursor for getting ADVERSE_EVENT Production */
cursor AE_CUR(
I_PATIENT_POSITION_ID in RECEIVED_DCMS.PATIENT_POSITION_ID%TYPE,
I_BEGIN_VISIT_NUMBER in RECEIVED_DCMS.VISIT_NUMBER%TYPE := RXCPDSTD.C_BEGIN_VISIT_NUMBER,
I_END_VISIT_NUMBER in RECEIVED_DCMS.VISIT_NUMBER%TYPE := RXCPDSTD.C_END_VISIT_NUMBER) is
select /*+ ordered use_nl(RDCM RES)
index(RDCM RECEIVED_DCM_UK2_IDX) */
RDCM.RECEIVED_DCM_ID,
RDCM.RECEIVED_DCM_ENTRY_TS,
.....
RDCM.DCM_TIME,
RDCM.ACTUAL_EVENT_ID,
RDCM.LAB_ID,
RDCM.LAB LAB,
RDCM.LAB RANGE SUBSET NUM,
```

**Outer Cursor**

**ACTUAL\_EVENT\_ID is retrieved as part of the standard cursor variables in the outer cursor**

# Effect of Correlation on Event on Inner Cursor

**Inner Cursor**

```
/* cursor for getting CONMED Production */
cursor CM_CUR(
I_PATIENT_POSITION_ID in RECEIVED_DCMS.PATIENT_POSITION_ID%TYPE,
I_BEGIN_VISIT_NUMBER in RECEIVED_DCMS.VISIT_NUMBER%TYPE := RXCPDSTD.C_BEGIN_VISIT_NUMBER,
I_END_VISIT_NUMBER in RECEIVED_DCMS.VISIT_NUMBER%TYPE := RXCPDSTD.C_END_VISIT_NUMBER,
I_ACTUAL_EVENT_ID in RECEIVED_DCMS.ACTUAL_EVENT_ID%TYPE) is
select /*+ ordered use_nl(RDCM RES)
      index(RDCM RECEIVED_DCM
            RDCM.RECEIVED_DCM_ID,
            . . . . .
            )
      from RECEIVED_DCMS RDCM,
           RESPONSES RES
      where RDCM.PATIENT_POSITION_ID = I_PATIENT_POSITION_ID
            and RDCM.DCM_ID = 54901
            and RDCM.END_TS = TO_DATE(3000000,'J')
            and RDCM.RECEIVED_DCM_ID+0 = RES.RECEIVED
            and RES.END_TS = TO_DATE(3000000,'J')
            and RES.DCM_QUESTION_GROUP_ID = 61001 and RES.CLINICAL_STUDY_ID = 10101
            and RDCM.ACCESSIBLE_TS <= SYSDATE
            and RDCM.VISIT_NUMBER between I_BEGIN_VISIT_NUMBER and I_END_VISIT_NUMBER
            and RDCM.ACTUAL_EVENT_ID = I_ACTUAL_EVENT_ID
            and res.dcm_question_id in (
              73901, 74301)
      group by RDCM.RECEIVED_DCM_ID,
```

**Cursor variable i\_actual\_event\_id specified when cursor is opened**

**used in the where clause**

# Call to Open Cursor for the Inner Loop

```
create or replace package body RXCPD_24901_0 as
procedure MAIN /*-----*/(
  open RXCPDSTD.PATIENTS_CUR; /* Production */
  loop <<fetch_next_patient>>
    fetch RXCPDSTD.PATIENTS_CUR into RXCPDSTD.PATIENTS_REC;
    open AE_CUR(RXCPDSTD.PATIENTS_REC.PATIENT_POSITION_ID);
    loop << fetch_AE_cur>>
      fetch AE_CUR into AE;
      open CM_CUR(RXCPDSTD.PATIENTS_REC.PATIENT_POSITION_ID, AE.actual_event_id);
      loop << fetch_CM_cur>>
        fetch CM_CUR into CM;
```

**Fetch on the Outer (AE) cursor retrieves a value of actual\_event\_id**

**Call to open inner (CM) cursor uses the actual\_event\_id retrieved by the outer\_cursor**

# Qualifying Expressions ...

- Allow selection of records based on any information retrieved by the current or any more outer cursor
  - Can join values between cursors in the qualifying expression

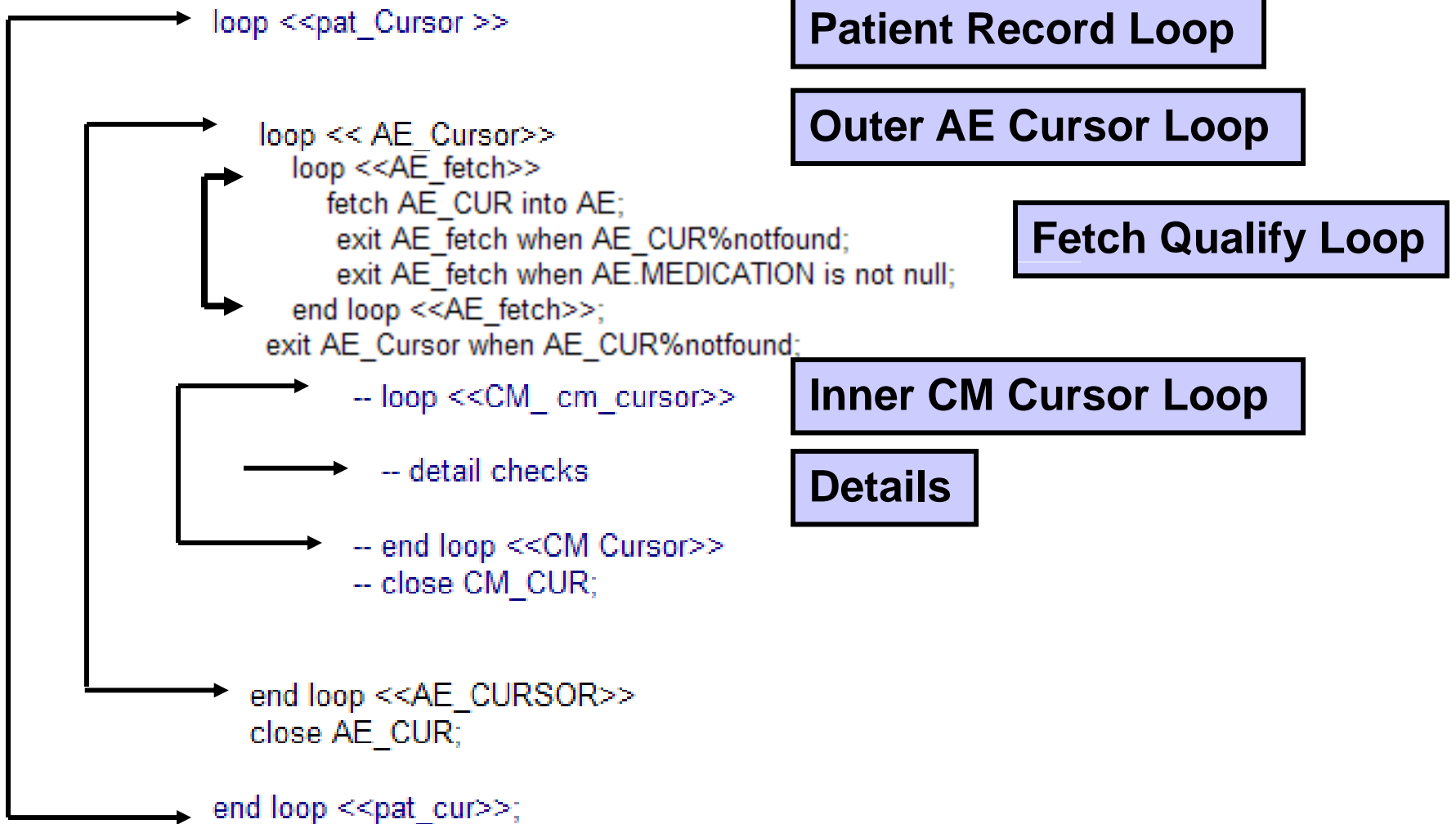
Alias	Correlate With	Event	Qualif. Quest	Qualifying Expression	Where Clause Extension
AE			<input type="checkbox"/>	AE.MEDICATION is not	
CM	AE	ACTUAL	<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		

Buttons: Back, Save, Questions, Correlating Questions, Custom Code

**Continue only if Medication on the AE form is not null**

# How do Qualifying Expressions change the generated Program?

# Qualifying Expressions: Test after Fetch ...



# Qualifying Expressions: Test after Fetch ...

```
loop << AE_Cursor >>  
  loop << AE_fetch >>  
    fetch AE_CUR into AE;  
    exit AE_fetch when AE_CUR%notfound;  
    exit AE_fetch when AE.MEDICATION is not null;  
  end loop << AE_fetch >>;  
exit AE_Cursor when AE_CUR%notfound;
```

Fetch an AE record

Exit AE Fetch Qualify loop if fetch qualifies

Exit AE Fetch Qualify loop if no more AEs

Process inner loops/details or exit the cursor loop

# Where Clause Extension ...

- Limit cursor fetch with SQL expression comprised of key DCM fields
  - An LOV is available for fields which can be used in the Extension
  - With caution, can use other variables in the **RECEIVED\_DCMS** and **RESPONSES** tables, but only for the current cursor
- One way to limit retrievals to specific visits

# Where Clause Extension

The screenshot shows the 'Maintain Study Validation Procedure (Study: ASD0553\_00)' window. The main window is titled 'Question Groups for V\_TEST' and contains a table with columns: Alias, Correlate With, Event, Qualif. Quest, Qualifying Expression, and Where Clause Extension. The 'Where Clause Extension' column has a dropdown menu open, showing a list of fields. A callout box points to this dropdown with the text: 'Can join only fields from the current cursor'. A separate dialog box titled 'List of DCM key fields' is also open, showing a search field 'Find R%' and a list of field names, with 'RDCM.RECEIVED\_DCM\_ID' selected. The dialog box has 'OK', 'Cancel', and 'Find' buttons.

Alias	Correlate With	Event	Qualif. Quest	Qualifying Expression	Where Clause Extension
AE			<input type="checkbox"/>		
CM			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		

Can join only fields from the current cursor

List of DCM key fields

Field Name

- RDCM.RECEIVED\_DCM\_ID
- RDCM.RECEIVED\_DCM\_ENTRY\_TS
- RDCM.INVESTIGATOR\_ID
- RDCM.SITE\_ID
- RDCM.DCM\_ID
- RDCM.DCM\_SUBSET\_SN
- RDCM.DCM\_DATE
- RDCM.DCM\_TIME
- RDCM.ACTUAL\_EVENT\_ID
- RDCM.LAB\_ID
- RDCM.LAB
- RDCM.LAB\_RANGE\_SUBSET\_NUM
- RDCM.QUALIFYING\_VALUE
- RDCM.SUBEVENT\_NUMBER
- RDCM.CLIN\_PLAN\_EVE\_ID
- RDCM.CLIN\_PLAN\_EVE\_NAME
- RDCM.VISIT\_NUMBER
- RES.REPEAT\_SN

**How do Where Clause Expressions change the generated Program?**

# Effect of Where Clause Extensions

## Extension of “where visit\_number=1”

```
where RDCM.PATIENT_POSITION_ID = I_PATIENT_POSITION_ID  
and RDCM.DCM_ID = 12101  
and RDCM.END_TS = TO_DATE(30000000,'J')  
and RDCM.RECEIVED_DCM_ID+0 = RES.RECEIVED_DCM_ID  
and RES.END_TS = TO_DATE(30000000,'J')  
and RES.DCM_QUESTION_GROUP_ID = 16001 and RES.CLINICAL_STUDY_ID = 2101  
and RDCM.ACCESSIBLE_TS <= SYSDATE  
and RDCM.VISIT_NUMBER between I_BEGIN_VISIT_NUMBER and I_END_VISIT_NUMBER  
and RDCM.VISIT_NUMBER=1  
and res.dcm_question_id in (  
18301, 18401)
```

**Added to the default Cursor definition**

# Controlling Range of Visits

**By default, all patient visits will be retrieved by the DCM cursor**

- Can limit to a range of visits fetched by providing the names of the first and last visits on the Procedure Question Group form

# Controlling Range of Visits ...

Definition => Validation Procs => Procedures, Press [Q-Groups]

Question Groups for V\_DEMOG\_OTHER\_SPEC

Alias	re- ? Refer?	Primary Refer?	Event Range		Sort Order		First/Last Event Only
			First	Last	Event	Extension	
A		<input checked="" type="checkbox"/>	VISIT 1	VISIT 3	RDCM.VISIT_NI	RES.REPEA	
		<input type="checkbox"/>					
		<input type="checkbox"/>					
		<input type="checkbox"/>					
		<input type="checkbox"/>					
		<input type="checkbox"/>					
		<input type="checkbox"/>					
		<input type="checkbox"/>					
		<input type="checkbox"/>					

Only DCMs recorded for visits starting at "Visit 1" and going through "Visit 3" will be retrieved

Back Save Questions Correlating Questions Custom Code

**How do specifying the  
Range of visits effect the  
generated program?**

# Effect of Range of Visits Control

```
where RDCM.PATIENT_POSITION_ID = I_PATIENT_POSITION_ID
and RDCM.DCM_ID = 24604
and RDCM.END_TS = TO_DATE(3000000,'J')
and RDCM.RECEIVED_DCM_ID+0 = RES.RECEIVED_DCM_ID
and RES.END_TS = TO_DATE(3000000,'J')
and RES.DCM_QUESTION_GROUP_ID = 31404 and RES.CLINICAL_STUDY_ID = 9504
and RDCM.ACCESSIBLE_TS <= SYSDATE
and RDCM.VISIT_NUMBER between I_BEGIN_VISIT_NUMBER and I_END_VISIT_NUMBER
and res.dcm_question_id in (
148804, 148504)
```

**These are passed as input when the DCM cursor is opened by the main procedure**

# Performance

- Generated PL/SQL code contains a multitude of loops
  - Cursors are opened, fetched and closed many times in the internal fetch loops
- Use as much correlation as possible to improve performance
  - Some types of correlations affect performance more than others
  - Correlation may be necessary to make sure the procedure does what is expected

# Effecting Performance

<u>Parameter</u>	<u>Where Implemented</u>	<u>Effect on Performance</u>
Event Range	In DCM Cursors	Large
Correlation		
Event	In DCM Cursor	Large
Qualifying Value	In DCM Cursor	Large
Correlating Questions	In DCM Cursor	Large
Where Clause Extension	In DCM Cursor	Large
Qualifying Expression	Test after DCM Fetch	Smaller

**Note: In many cases, parameters must be changed so procedure performs correctly**

# Summary

- Procedure Generation produces a complex PL/SQL program
- Program makes extensive use of cursors and looping structure
- Program performance can be changed by correct use of many techniques available from the form templates

# Contact Information

Steve Rifkin  
BioPharm Systems  
908 822 0553  
[srifkin@biopharm.com](mailto:srifkin@biopharm.com)



Steve Rifkin has almost 14 years of experience working with the OLS Product Suite. As an Functional Consultant, and then as a Practice Manager with Oracle Consulting in the Oracle Clinical Group, Steve has assisted over 60 companies with implementation of Oracle Clinical, RDC and TMS. He has led the implementation process and has provided training and performed custom coding. Since joining Biopharm Systems in 2000, Steve has been responsible for developing Oracle Clinical training courses, and providing training and implementation services for Biopharm clients.