

1 . Business Components fetches data from only one table. TRUE/FALSE?

**FALSE**

2 . Identify the incorrect statement regarding Explicit Joins:

**Automatically created for 1:1 Extension Tables**

Used to map single valued fields from joined table

Includes a join specification and join definition

Specifies the relationship between BC and Table

3 . Relationship between BC and Joined Table is always (Choose appropriate options):

**1:1**

M:M

1:M

**M:1**

4 . Implicit joins are used to bring data from Party Tables into Non-Party BC. State TRUE/FALSE ?

**FALSE**

5 . S\_CONTACT is a Party Table. State TRUE/FALSE ?

**TRUE**

6 . 1:M Link requires an Intersection table to establish the relationship between parent and child BC. True / False.

**FALSE**

7 . Business Objects definition includes a Primary BC and Multiple Child BCs and Links that relate these BCs. True / False.

**TRUE**

8 . No Insert, No Delete, No Update and No Merger properties are available for both applet and BC object definitions. State TRUE / FALSE

**TRUE**

9 . Postdefault and Predefault value are set for a field. The user creates a new record and does not provide the value to this field and steps off. What value would be stored in this field?

**Predefault**

10 . Postdefault and Predefault value are set for a field. The user edits the above record and removes the values stored in the field. What value would be stored in this field?

**Postdefault**

11 . Sorting is allowed in Calculated Fields? State TRUE/ FALSE

**FALSE**

12 . BC Read Only Field user property is defined under which object definition?

**BC User Property**

13 . Required property when set to TRUE makes the field required? State TRUE/ FALSE

**TRUE**

14 . 1:M extension tables have NAME, TYPE, ROW\_ID as User key columns. State TRUE/ FALSE.

**FALSE**

15 . An \_XM Table can contain multiple child BC records. State TRUE/ FALSE.

**TRUE**

16 . Arrange the sequence of steps for assigning a BC to BO.

- A . Create a List Applet
- B . Assign a Applet to View
- C . Create a Link
- D . Assign BC to BO

**C,D,A,B**

17 . It is advised to map the unused existing columns to fields. State TRUE/FALSE.

**TRUE**

18 . Which Statement is true regarding creating Intersection table using Wizard

Type of table is Data(Public)

PAR\_ROW\_ID is the foreign key to the base table

**2 Foreign Keys are specified**

Type and Name columns are created

19 . When changing the schema, Changes are propagated to other developers before migrating it to server. State TRUE/FALSE.

**FALSE**

20 . APPLY / DDL and Activate is clicked while committing the changes to the local database schema. State TRUE/FALSE.

**FALSE – done on server**

21 . Static Drilldown enables to drilldown to different view. TRUE/ FALSE ?

**TRUE**

- 22 . Thread bar is updated whenever the user navigates to views of different Business Object. TRUE / FALSE  
**TRUE**
- 23 . Dynamic Applet Toggle automatically toggles based on the value in the field on that business component. TRUE / FALSE  
**TRUE**
- 24 . Dynamic picklist displays values from a dropdown for user selection. TRUE / FALSE ?  
**FALSE**
- 25 . Administration – Data > List of Values Screen is where the static picklist data is administered? TRUE/FALSE ?  
**TRUE**
- 26 . Multi Value Field is a field from the Parent BC which is mapped to the Child BC field. State TRUE/FALSE.  
**TRUE**
- 27 . A Multi Value Group applet shows the parent records in a list applet. State TRUE/FALSE.  
**FALSE**
- 28 . A Primary allows fast retrieval of a designated child records for display and this record will be called as primary. State TRUE/FALSE.  
**TRUE**
- 29 . Business Process is a sequence of tasks executed to accomplish a definite business objective. State TRUE/FALSE.  
**TRUE**
- 30 . Identify the correct statement regarding property sets:  
**Represents data using name/value pairs**  
Consists of one or more operations called methods  
Has specified set of input and output arguments
- 31 . Steps to test a BS using simulator are provided. Arrange them in the right sequence.
- A . Examine the output set name/value pairs
  - B . Click the run on one input button
  - C . Select a business service name and business service method
  - D . Create the input property set with name and value pairs
- C,D,B,A**

32 . Identify the incorrect statement from below about Workflows :

Workflows are used to automate some parts of the Business Process

Process Properties provide input to the WF Steps

**WF are created in Siebel client**

Workflow mode describes the runtime behavior

33 . Process Properties are variables that used to store the inputs to WF steps and outputs from the WF Steps. State TRUE/FALSE.

**TRUE**

34 . Below are the steps to configure a WF Process. Arrange them in the right sequence.

A . Create a New WF Process

B . Configure WF Steps

C . Validate WF Process

D . Add Workflow Steps

E . Specify Process Properties

**A,E,D,B,C**

35 . Steps involved in testing WF using Simulator is provided. Please arrange them in right sequence.

A . Start the Simulator

B . Start the Simulation

C . Specify the test record

D . Execute the Workflow

**C,A,B,D**

36 . Impacts on the revision of workflows. Select all that is applicable :

**Create a new version of WF**

Incorporate edit changes in the original version

**Increment the WF version number**

**WF Status is set to In Progress**

Validate the WF

# SIEBEL

Business Components and Joins

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Identify the structure of business component and joins
- Define mapping of fields to base tables, joint tables and extension tables
- Describe how to create join to pull data from joined table



# Topic List

#No	Module Topics
1	Business Component
2	Joins

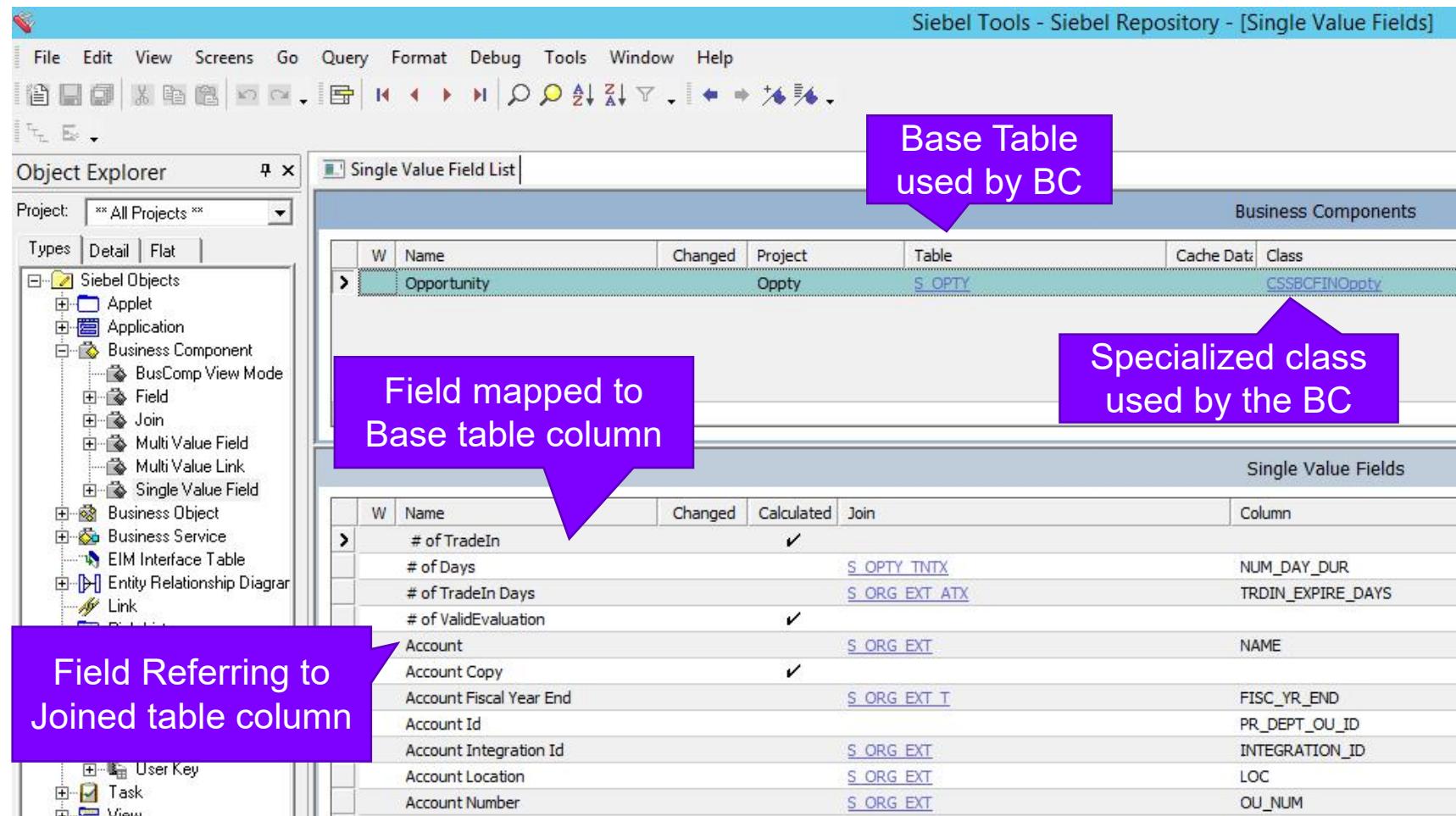
# Topic List

#No	Module Topics
1	Business Component
2	Joins

# Business Component

## Overview

- Is logical grouping of data from one or more tables.
  - One Base Table and One/More Joined Tables
- Consists of several fields including Single Valued Fields.
  - References single column in the Table
- Class Property indicates the C++ used at runtime by the BC.
  - Determines the behaviour of the BC
  - By default, CSSBusComp is used
  - Other special



# Business Component

## Single Value Field (SVF)

### SVF mapped to Base Table

- Join property is blank
- Column property will display column from Base table.
  - Select the column

The screenshot shows the Siebel Object Explorer and the Single Value Field List windows.

**Object Explorer:** Shows the project "All Projects" and various object types under "Siebel Objects".

**Single Value Field List:** Shows a table with columns: W, Name, Changed, Table, Project, Cache Data, Class, Data. One row is listed: Opportunity, ✓, S\_OPTY, Oppy, CSSBCFINOppty.

**Single Value Fields:** Shows a table with columns: W, Name, Changed, Calculated, Join, Column. One row is listed: New Single Value Field, ✓, ADJUSTED\_AMT.

**Columns:** A dialog box showing a list of columns from a base table. The table has columns: Name, Physical Type, Length, Comments. Several columns are listed, including ADJUSTED\_AMT (Number, 22, Adjusted Amount), ALIAS\_NAME (Varchar, 100, Alternative Name), and others like ALTERNATE\_DATES, APPL\_OWNER\_TYPE\_CD, APPR\_ID, ASGN\_DT.

# Business Component

## Single Value Field (SVF)

### Data Type for Fields

- Type Property specifies the Data Type for the field
- It must correspond to the Physical type of the column in table.
- By default, Siebel populates DTYPE\_TEXT.
  - This can be changed by selecting the values from Siebel defined data types in the drop down.

The screenshot shows the Siebel Object Explorer and the Single Value Field List windows. In the Object Explorer, the 'Single Value Field' node under 'Business Component' is selected. In the Single Value Field List window, a new field named 'New Single Value Field' is being created for the 'Opportunity' object. The 'Column' is set to 'ADJUSTED\_AMT' and the 'Type' is set to 'DTYPE\_NUMBER'. A dropdown menu shows other available data types: DTTYPE\_BOOL, DTTYPE\_CLOB, DTTYPE\_CURRENCY, DTTYPE\_DATE, DTTYPE\_DATETIME, DTTYPE\_ID, DTTYPE\_INTEGER, DTTYPE\_NOTE, DTTYPE\_NUMBER, DTTYPE\_PHONE, DTTYPE\_TEXT, DTTYPE\_TIME, DTTYPE\_UTCDATETIME, and DTTYPE\_TEXT. The 'Text Length' is set to 100. Other fields listed include '# of TradeIn', '# of Days', '# of TradeIn Days', '# of ValidEvaluation', 'Account', 'Account Copy', 'Account Fiscal Year End', 'Account Id', 'Account Integration Id', 'Account Location', and 'Account Number'.

Name	Join	Column	Type	Text Length
New Single Value Field		ADJUSTED_AMT	DTYPE_NUMBER	100
# of TradeIn			DTTYPE_BOOL	
# of Days	S_OPTY_TNTX	NUM_DAY_DUR	DTTYPE_CLOB	
# of TradeIn Days	S_ORG_EXT_ATX	TRDIN_EXPIRE_DAYS	DTTYPE_CURRENCY	
# of ValidEvaluation			DTTYPE_DATE	
Account	S_ORG_EXT	NAME	DTTYPE_DATETIME	
Account Copy			DTTYPE_ID	
Account Fiscal Year End	S_ORG_EXT_T	FISC_YR_END	DTTYPE_INTEGER	
Account Id		PR_DEPT_OU_ID	DTTYPE_NOTE	
Account Integration Id	S_ORG_EXT	INTEGRATION_ID	DTTYPE_NUMBER	30
Account Location	S_ORG_EXT	LOC	DTTYPE_PHONE	50
Account Number	S_ORG_EXT	OU_NUM	DTTYPE_TEXT	30

# Business Component

## Single Value Field (SVF)

### Default SVF

- Each System Columns are mapped to set of Single Valued Fields which are not visible in UI.
- These are available to be mapped to Applets and for scripting purposes.
  - Example: ID Field is mapped to ROW\_ID column
  - Created Field is mapped to CREATED Column



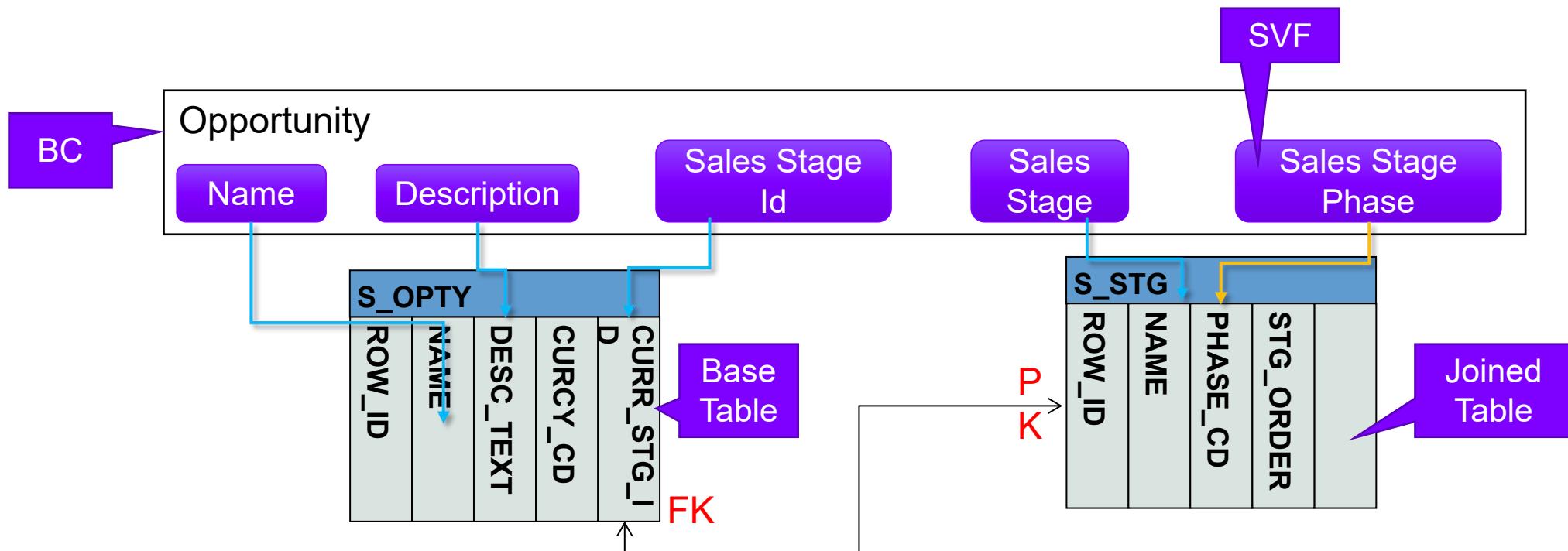
# Topic List

#No	Module Topics
1	Business Component
2	Joins

# Joins

## Joined Tables

- Business Component very often needs to pull data from additional or related tables
  - To display them on Applets
  - For further processing in the Business component
- SVFs referring to these joined table columns are always read only.



# Joins

## Joined Tables

- Join always returns one row
- Relationship between BC and Joined Table is always M:1 or 1:1
- A foreign key is needed to establish this relationship.
  - A FK column on the Base table will refer to the PK column of the joined table.
  - FK field on the BC is mapped to FK column.



# Joins

## Types of Joins

### Implicit Join

The implicit join allows the business component to pull data from the extension tables on 1:1 basis

- SVF pointing to extension table will have:
  - Join Property set to the Name of the extension table
  - Column property is the column from the extension table
  - Example: Hobby Field in Contact BC is mapped to ATTRIB\_06 column from S\_CONTACT\_X table

### Explicit Join

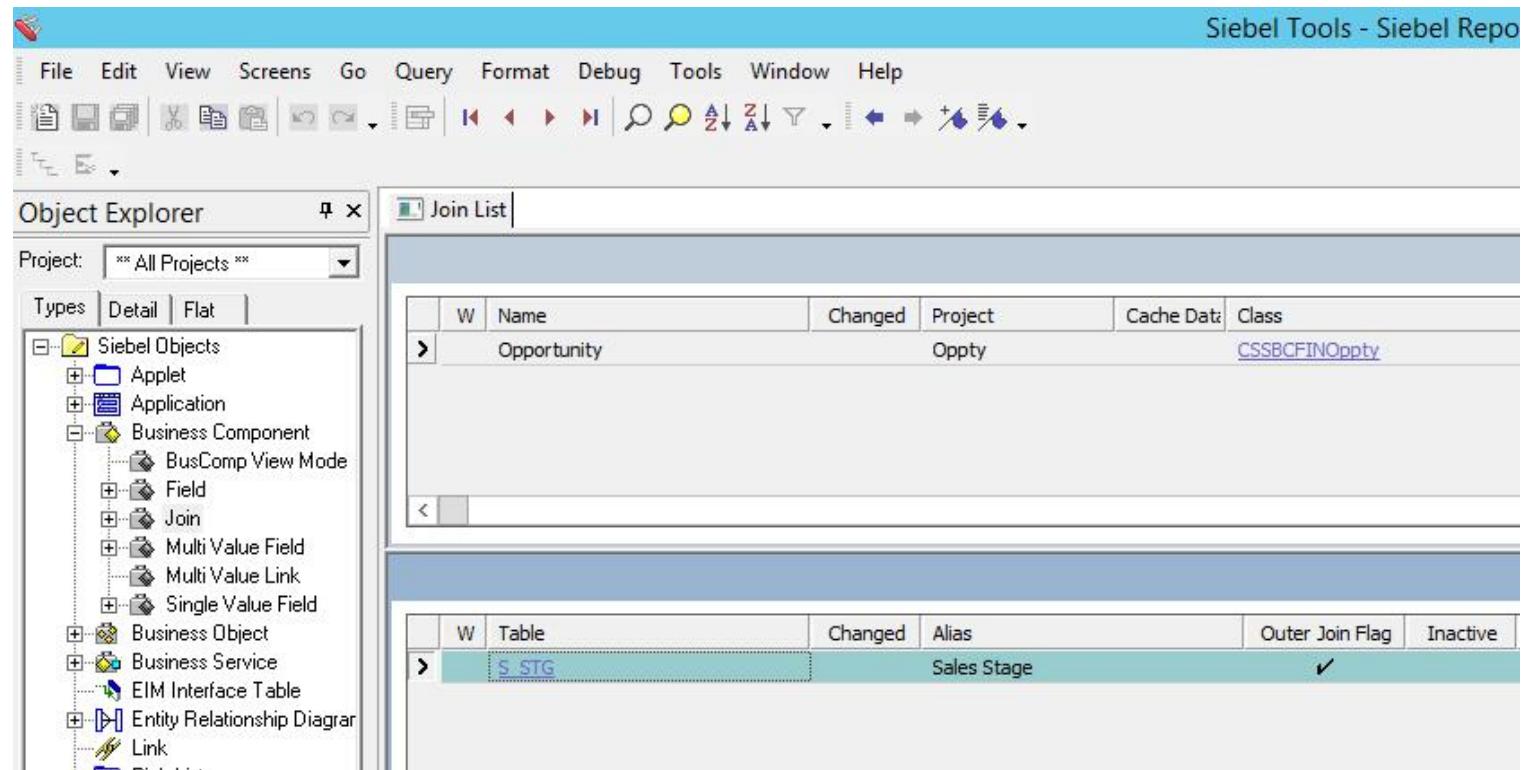
- The explicit join specifies the relationship between the BC and Joined Table
- It Includes a Join Definition and a Join Specification which are used to map SVFs to Joined table columns.



# Joins

## Join Definition

- Join is the child object definition of the Business component.
- Indicates the joined table from which to retrieve data.
- Multiple join definition can be created for the same table.
- ALIAS property in Join is used to distinguish multiple joins.
- **Outer Join Flag** when set to TRUE will return all records from the base table even when no records exists in the joined table.
  - Affects performance
  - Should be used on need basis alone



# Joins

## Join Specification

- A Join specification must be created along with join definition
- Defines how the record is retrieved
  - Foreign key and the primary key which is used to relate the base table to the joined table

The screenshot shows two windows from the Siebel Object Explorer:

- Join Specification List:** This window lists join specifications. One entry is shown: **S\_STG** (Table: Sales Stage). Below it is a detailed view of a join specification row:

Name	Source Field	Destination Column
Sales Stage Id	Sales Stage Id	ROW_ID

A purple callout box points to the "Sales Stage Id" field with the text: **Maps to FK of S\_OPTY**.
- Single Value Field List:** This window lists single value fields. One entry is shown: **Opportunity** (Table: S\_OPTY).

The screenshot shows two windows from the Siebel Object Explorer:

- Join Specification List:** This window lists join specifications. One entry is shown: **S\_STG** (Table: Sales Stage). Below it is a detailed view of a join specification row:

Name	Join	Column	Type	Text Length
Sales Stage Id		CURR_STG_ID	DTYPE_ID	15
- Single Value Field List:** This window lists single value fields. One entry is shown: **Opportunity** (Table: S\_OPTY).



# Joins

## Joined Fields

- Join Property will have the ALIAS name of the Join.
- Column property will be mapped to Joined Table.

The screenshot shows the Siebel Object Explorer and Field List windows. The Object Explorer window on the left displays a tree view of Siebel objects, with the 'Join' node under 'Business Component' expanded. The Field List window on the right shows a list of fields for the 'Opportunity' table (S\_OPTY). A specific field, 'DesignOppy Sales Stage', is selected, revealing its properties in the bottom pane. The 'Fields' table lists various sales stage-related fields, each with a 'Join' column indicating the alias and a 'Column' column indicating the mapped table or field name.

Name	Join	Column
DesignOppy Sales Stage	Sales Stage	NAME
Opportunity Sales Status	Sales Stage	STAGE_STATUS_CD
Sale Stage Phase	Sales Stage	PHASE_CD
Sales Stage	Sales Stage	NAME
Sales Stage - ACAPS	Sales Stage	NAME
Sales Stage Order	Sales Stage	STG_ORDER
Sales Stage Status	Sales Stage	STAGE_STATUS_CD
Sales Stage Win Percent	Sales Stage	RCMND_WIN_PROB



# Joins

## Implicit Joins

- Prebuilt Siebel application has many 1:1 extension tables.
  - These have 1:1 relationship with the base table
  - Example : S\_OPTY\_X is 1:1 extension table for S\_OPTY
- PAR\_ROW\_ID in the extension table sets up the relationship with the base table.
  - Example : PAR\_ROW\_ID in S\_OPTY\_X is mapped to ROW\_ID of S\_OPTY



# Joins

## Implicit Joins

- Join is established in the Table column and not under Join Definition
- Fields mapped Implicit Joined Columns are editable and always involve an outer join

The screenshot shows the Oracle Database Object Explorer interface. On the left, the Object Explorer tree view is expanded to show various object types like Application, Business Component, Field, Join, Multi Value Field, Single Value Field, Business Object, Business Service, EIM Interface Table, Entity Relationship Diagram, Link, Pick List, Project, Screen, Table, Column, and Index. The 'Join' node is selected. To the right, the 'Column List' window is open, displaying a table structure for the 'S\_OPTY\_X' object. The top section of the Column List window has tabs for 'Tables' and 'Columns'. Below these tabs is a header row with columns: Module, Object Language Locked, Name, Object Locked, and Object Locked By Name. A single row is present in the table body, showing 'Module' as '>', 'Name' as 'S\_OPTY\_X', and 'Object Locked' as checked. The bottom section of the Column List window is titled 'Columns' and contains a detailed table with columns: Required, W, Name, User Name, Alias, Type, Foreign Key Table, Primary Key, Physical Type, and User Key Sequence. One row is shown, with 'Required' checked, 'Name' as 'PAR\_ROW\_ID', 'User Name' as 'Parent Row Id', 'Alias' as 'PARENT\_ROW\_ID', 'Type' as 'System', 'Foreign Key Table' as 'S\_OPTY', 'Physical Type' as 'Varchar', and 'User Key Sequence' as '1'.

# Joins

## Implicit Joins

- SVF based on the Extension table columns will have:
  - Join property as the Name of the 1:1 Extension Table
  - Column Property will refer to columns from the Joined Table
- These fields are editable in the UI

The screenshot shows the Oracle ADF Business Components IDE interface. On the left, the Object Explorer pane displays a tree view of project components under 'All Projects'. The 'Types' tab is selected, showing categories like List, Tree, Application, Business Component, Field, Join, Multi Value Field, Multi Value Link, and Single Value Field. The 'Single Value Field' node is expanded. On the right, the 'Single Value Field List' window is open, showing a table of single value fields for the 'Opportunity' business component. The table includes columns for Name, Join, Column, Type, and Text Length. Below this, the 'Single Value Fields' table lists detailed properties for each field, such as Budget Amt, Budgeted, Champion, Deal Horizon, etc.

Name	Join	Column	Type	Text Length
Budget Amt	S_OPTY_X	ATTRIB_44	DTYPE_TEXT	15
Budgeted	S_OPTY_X	ATTRIB_09	DTYPE_BOOL	
Champion	S_OPTY_X	ATTRIB_11	DTYPE_BOOL	
Deal Horizon	S_OPTY_X	ATTRIB_37	DTYPE_TEXT	15
Decision Level	S_OPTY_X	ATTRIB_39	DTYPE_TEXT	15
Est HW Disc	S_OPTY_X	ATTRIB_38	DTYPE_TEXT	15
Existing Cust	S_OPTY_X	ATTRIB_08	DTYPE_BOOL	
HW Percent	S_OPTY_X	ATTRIB_34	DTYPE_TEXT	15
JG Fee to Firm	S_OPTY_X	ATTRIB_25	DTYPE_CURRENCY	
PCS Laptops	S_OPTY_X	ATTRIB_04	DTYPE_TEXT	15
Profile Industry	S_OPTY_X	ATTRIB_41	DTYPE_TEXT	50
Profile Revenue	S_OPTY_X	ATTRIB_42	DTYPE_TEXT	15

# Knowledge Checks

## Answer the following:

1. Business Components fetches data from only one table. TRUE/FALSE?
  - A. TRUE
  - B. FALSE
2. Identify the incorrect statement regarding Explicit Joins:
  - A. Automatically created for 1:1 Extension Tables
  - B. Used to map single valued fields from joined table
  - C. Includes a join specification and join definition
  - D. Specifies the relationship between BC and Table
3. Relationship between BC and Joined Table is always (Choose appropriate options):
  - A. 1:1
  - B. M:M
  - C. 1:M
  - D. M:1



# Module Summary

**Now, you should be able to:**

- Identify the structure of business component and joins
- Define mapping of fields to base tables, joint tables and extension tables
- Describe how to create join to pull data from joined table



# **Thank You**

SIEBEL

Party Business Components

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Describe party concept
- Identify party business components
- Define join and join specification to bring party data
- Explain how to map fields to party extension tables



# Topic List

#No	Module Topics
1	Party Entities
2	Party Data Model
3	Joins in Party Data

# Topic List

#No	Module Topics
1	Party Entities
2	Party Data Model
3	Joins in Party Data

# **Party Entities (1)**

## **Party Data**

Data that describes the Company itself and the way it is organised

- Like the Employees and other users of the company's applications
- They include Customers, Partners and other Enterprises associated with the Company
- Also Contacts and other people outside the company.

Party Data can be Individual Person or Groups.



# Party Entities (2)

## S\_PARTY

Base Table for all Party Data

Consists of few columns that store party data common to all party business components

S_PARTY				
ROW_ID	NAME	PARTY_TYPE_CD	PARTY_UID	PAR_PARTY_ID

Indicates the type of Party

# **Party Entities (3)**

## **Party Business Components**

These have S\_PARTY as their Base Table

Data is stored in the Extension Tables

Party Data is represented using Party Business Components

Example :

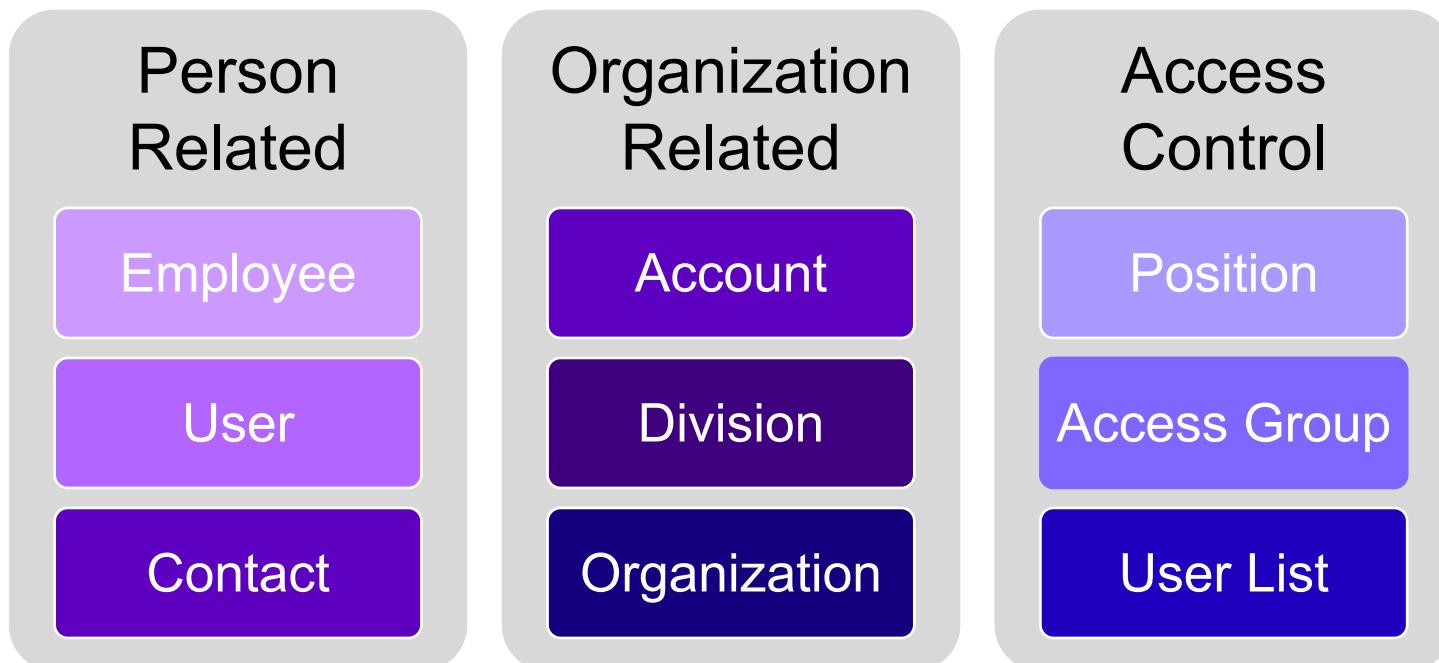
- Accounts
- Contacts
- Employee
- Position



# Party Entities (4)

## Classification of Party Business Components

Party Business Component can be classified as follows :



# Party Entities (5)

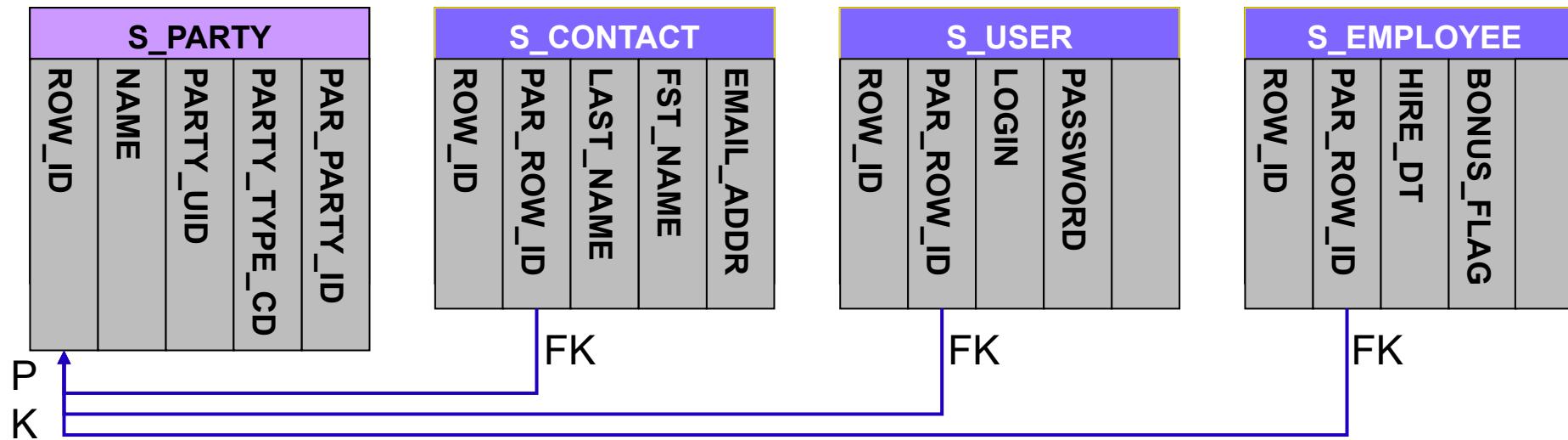
## Person Related Party BC

Used to Store Person, User and Employee specific data.

Party BC and their Extension Tables are :

- ✓ **S\_CONTACT** for Contact
- ✓ **S\_USER** for User
- ✓ **S\_EMPLOYEE** for Employee

Store main data in S\_CONTACT



# Party Entities (6)

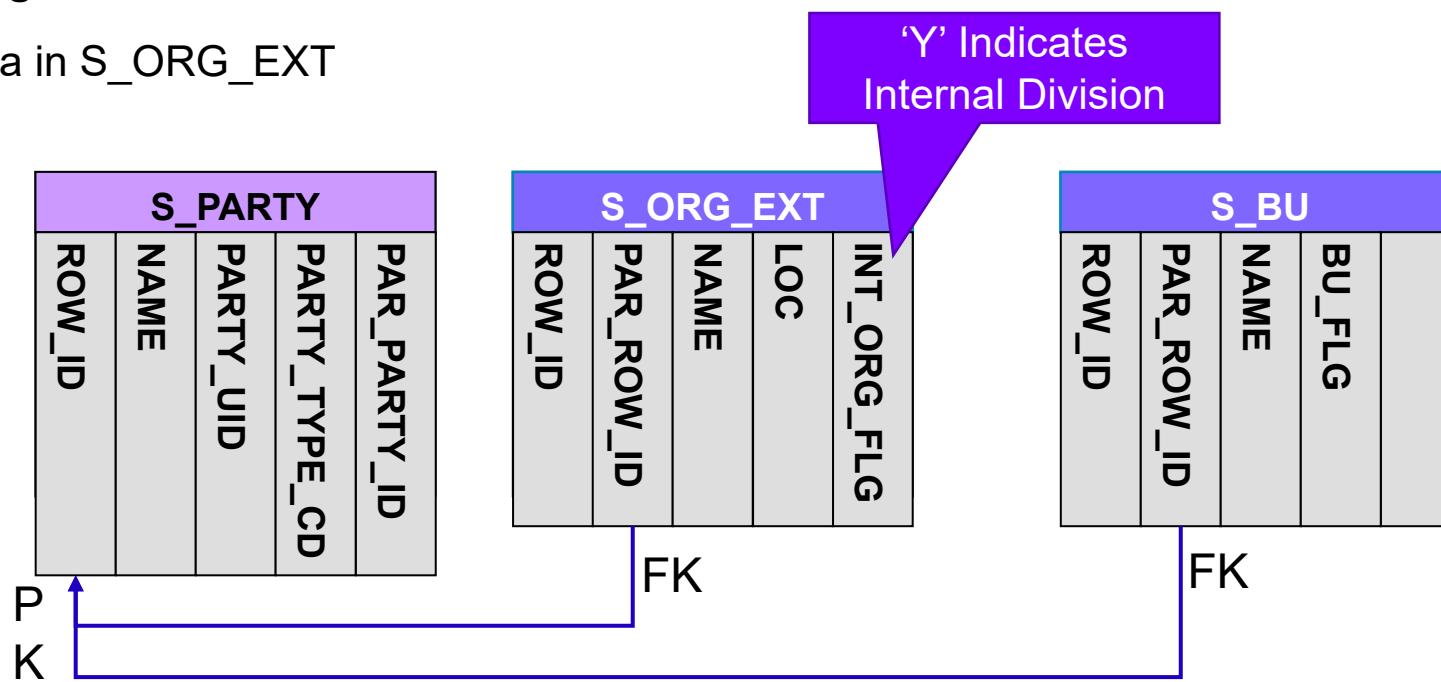
## Organization Related Party BC

Stores data related to Organization, Division and Accounts.

Party BC and their Extension Tables are :

- ✓ **S\_ORG\_EXT** for **Account** and **Division**
- ✓ **S\_BU** for **Organization**

Store main data in **S\_ORG\_EXT**

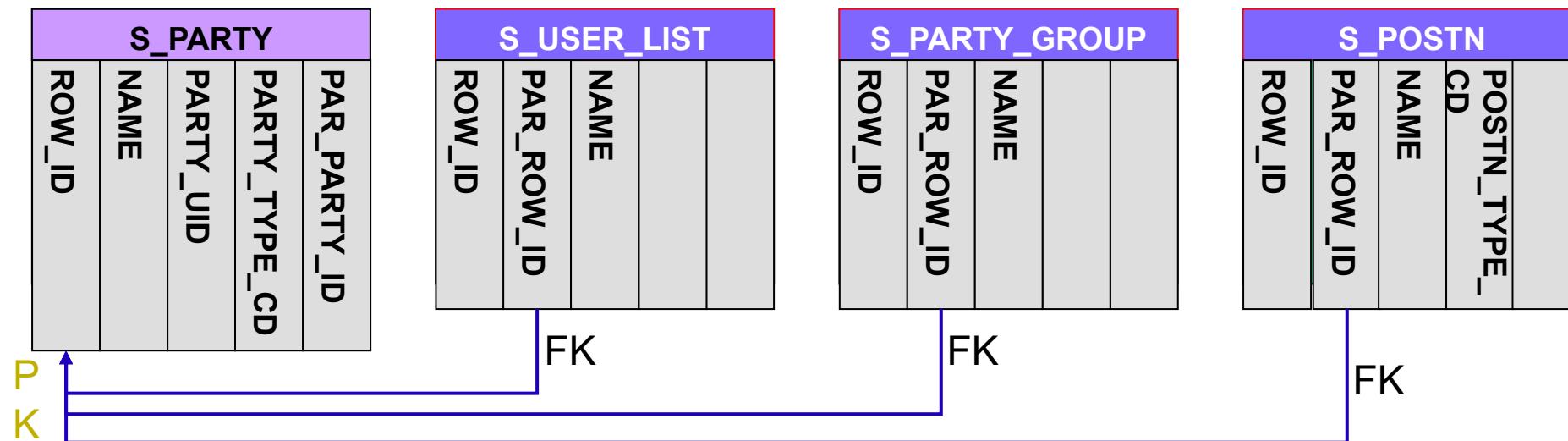


# Party Entities (7)

## Access Control Party BC

Certain Business Components are used for Access Control and they include :

- ✓ User List – **S\_USER\_LIST**
- ✓ Access Groups – **S\_PARTY\_GROUP**
- ✓ Positions – **S\_POSTN**



# **Party Entities (8)**

## **Party Inter Tables**

### **S\_PARTY\_PER**

- ✓ Relates other parties with Person Party BC
- ✓ Example: relationship between user list and users/users and position

### **S\_PARTY\_REL**

- ✓ Relates any two party BCs.
- ✓ Example: relationship between contact and account / account and account

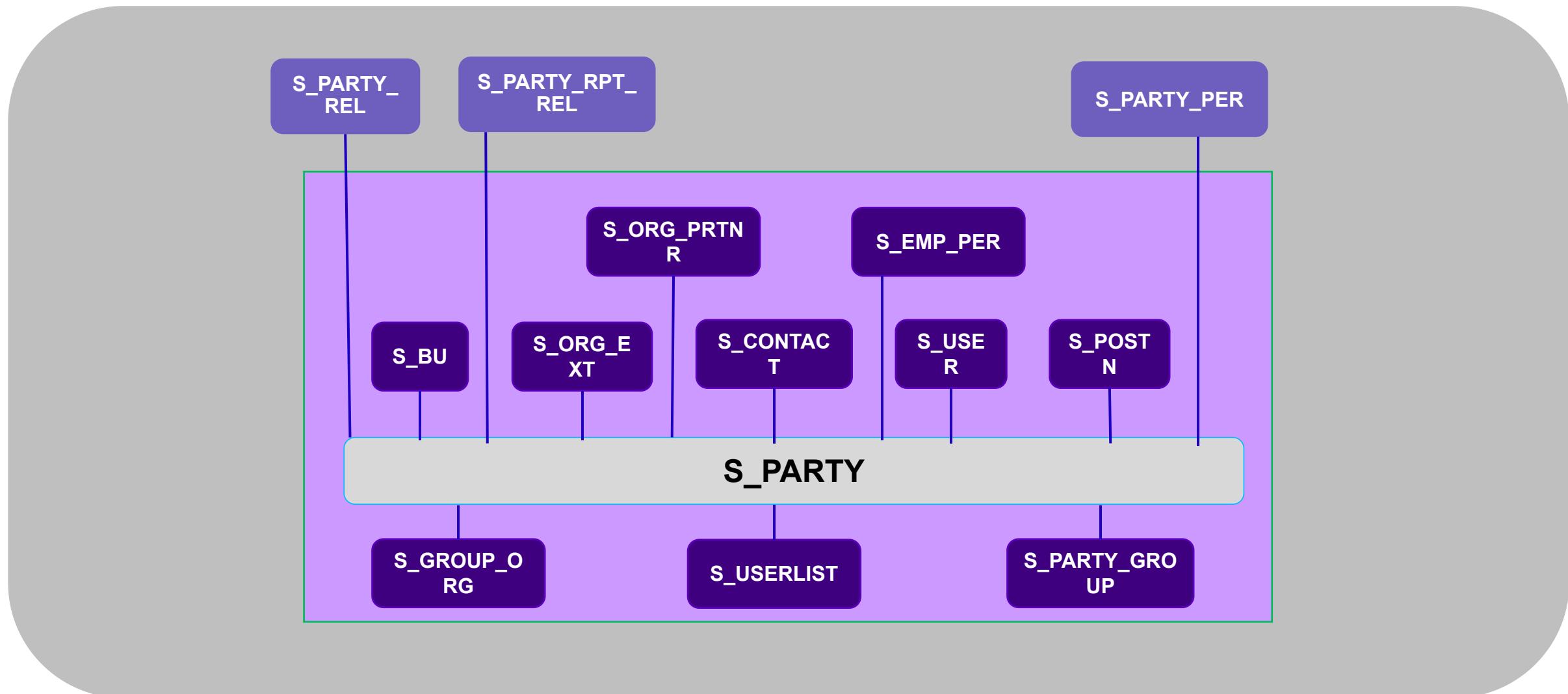


# Topic List

#No	Module Topics
1	Party Entities
2	Party Data Model
3	Joins in Party Data

# Party Data Model (1)

## Overview



# **Party Data Model (2)**

## **Benefits of Party Model**

It allows all instances of Party BCs to be treated similar in certain aspects

- ✓ They have common primary key and user keys

Provides ways to relate Instances of different party BCs

- ✓ Example: an access group of positions and accounts can be created.



# Topic List

#No	Module Topics
1	Party Entities
2	Party Data Model
3	Joins in Party Data

# Joins in Party Data (1)

## Implicit Join

Used between Party Business Components and Party Extension Tables

Name of the Join is the name of the extension table.

For Example, Contact BC fetches contact details such as First Name , Last Name from S\_CONTACT Extension Table.

This is used for mapping Party Fields in Party Business Components.

The screenshot shows the Oracle ADF Designer interface. On the left is the 'Object Explorer' window, which lists various project types under 'Business Component'. In the center is the 'Single Value Field List' window, which displays a table of fields and their mappings. The table has columns for Name, Join, and Column.

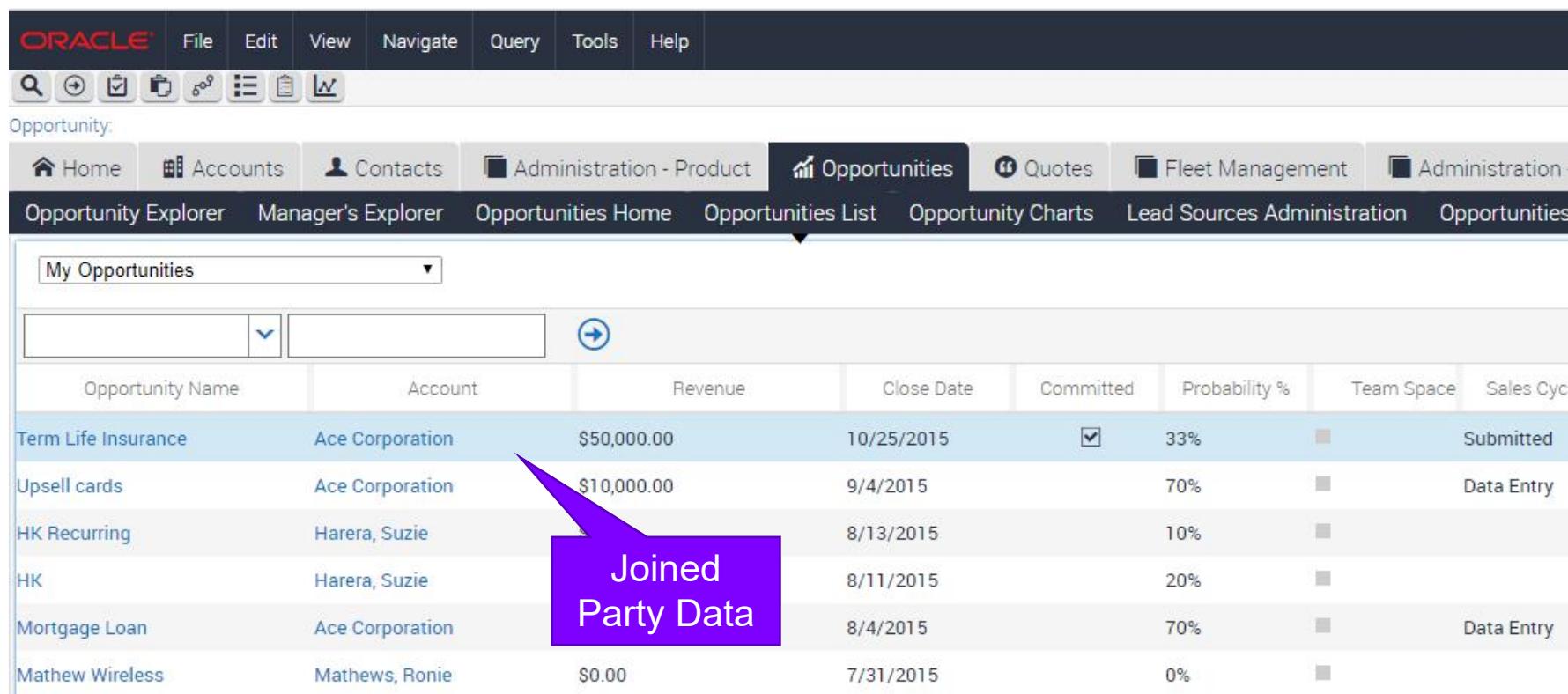
W	Name	Join	Column
>	Application Name		
	ContactImgFileName	S_LIT	FILE_NAME
	Created By Name	S_CONTACT	CREATOR_LOGIN
	Employee Login Name	Login_Employee	LOGIN
	Employer Name	Employer	NAME
	First Name	S_CONTACT	FST_NAME
	First Name Last Name		
	Full Name		
	INS Employers Name	S_CONTACT_X	ATTRIB_01
	Influencer First Name	Household_Analysis	FST_NAME
	Influencer Last Name	Household_Analysis	LAST_NAME
	Last Name	S_CONTACT	LAST_NAME
	Last Name, First Name		
	Login Name	S_USER	LOGIN
	Loyalty Contact Full Name		
	Maiden Name	S_CONTACT	MAIDEN_NAME
	Manager First Name		

# Joins in Party Data (2)

## Explicit Join

It is created to fetch data from party tables to the non party business component.

Such as account details in opportunity list applet.



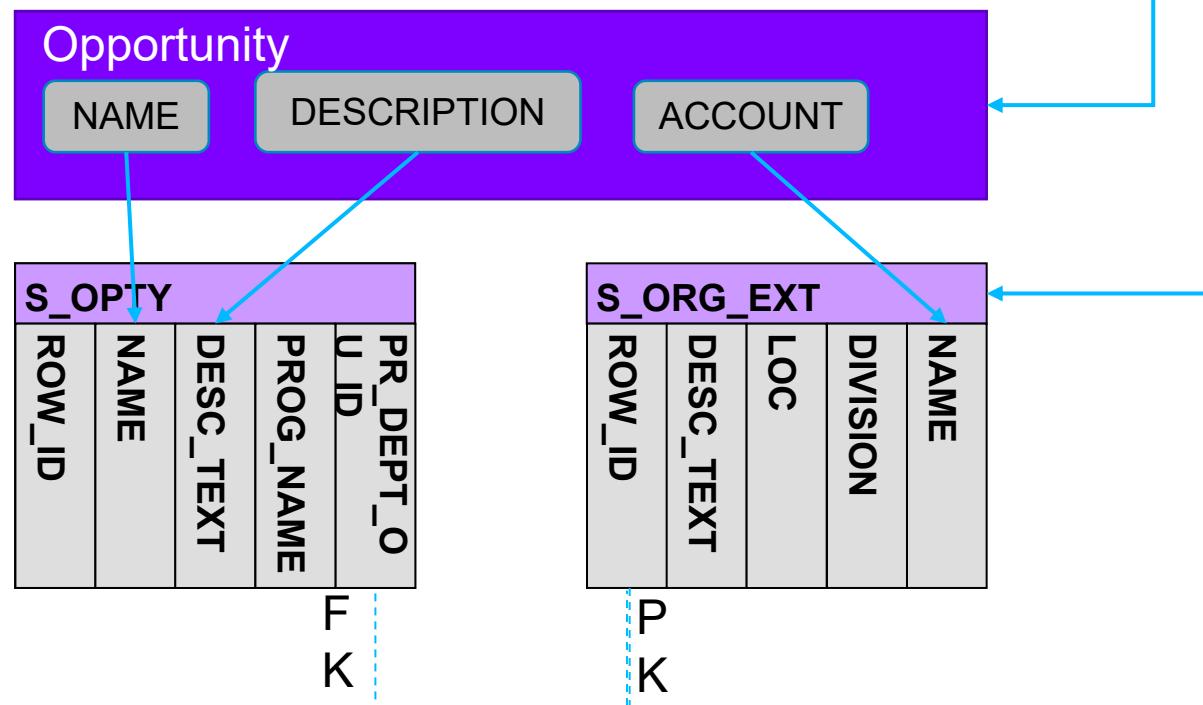
The screenshot shows the Oracle Application Express interface for managing opportunities. The top navigation bar includes links for Home, Accounts, Contacts, Administration - Product, Opportunities (which is the active tab), Quotes, Fleet Management, and Administration. Below the navigation is a search bar and a menu for 'My Opportunities'. The main content area displays a table of opportunities with columns for Name, Account, Revenue, Close Date, Committed, Probability %, Team Space, and Sales Cycle. The first row, 'Term Life Insurance', has its 'Account' value, 'Ace Corporation', highlighted with a purple box and labeled 'Joined Party Data' with a purple arrow pointing to it. The other rows show various opportunities like 'Upsell cards', 'HK Recurring', etc., each associated with an account.

Opportunity Name	Account	Revenue	Close Date	Committed	Probability %	Team Space	Sales Cycl
Term Life Insurance	Ace Corporation	\$50,000.00	10/25/2015	<input checked="" type="checkbox"/>	33%		Submitted
Upsell cards	Ace Corporation	\$10,000.00	9/4/2015		70%		Data Entry
HK Recurring	Harera, Suzie		8/13/2015		10%		
HK	Harera, Suzie		8/11/2015		20%		
Mortgage Loan	Ace Corporation		8/4/2015		70%		Data Entry
Mathew Wireless	Mathews, Ronie	\$0.00	7/31/2015		0%		

# Joins in Party Data (3)

## Explicit Join

Explicit Join definitions are created to fetch data from S\_ORG\_EXT to Opportunity BC



The screenshot shows the SAP Business Component Designer interface with the following panels:

- Join List:** Shows an entry for "Opportunity" with alias "Oppy".
- Table:** Shows the "S\_ORG\_EXT" table with an alias "S\_ORG\_EXT".
- Join Specification List:** Shows an entry for "S\_ORG\_EXT".
- Source Field:** Shows a mapping for "Account Id" as the source field, mapped to "PAR\_ROW\_ID" as the destination column.
- Single Value Field List:** Shows an entry for "Opportunity" with alias "S\_OPTY".
- Single Value Fields:** Shows a mapping for "Account Id" with column "PR\_DEPT\_OU\_ID" and type "DTYPE\_ID".



# Joins in Party Data (4)

## Explicit Join

They are used to fetch party data into party BC.

Example: parent account data into account list applet

The screenshot shows the Oracle Application Express interface for managing accounts. The top navigation bar includes links for Home, Accounts, Contacts, Administration - Product, Opportunities, Quotes, Fleet Management, and Administration. Below the navigation is a toolbar with various icons. The main content area is titled 'Account' and displays a list of accounts under 'My Accounts'. The grid columns are: New, Name, Site, Parent, Account Team, Main Phone #, Status, Account Type, and Account ID. A purple callout box points to the 'Parent' column, which lists the names of parent accounts such as '3 Com', 'A.E.Sorenson & Associates', 'ABC Constructions', 'AG Edwards & Sons, Inc', 'AH Frank', 'Aaron-Jones Dry Cleaning', 'Abbey General Hospital', 'Abbot Designs', and 'Abbot School for Medicine'. The 'Status' column indicates all accounts are 'Active'.

New	Name	Site	Parent	Account Team	Main Phone #	Status	Account Type	Acco
	3 Com	HQ-Distribution	SADMIN	(650) 555-1212	Active	Customer	Cust...	
	A.E.Sorenson & Associates	Charlotte	SADMIN	(704) 653-4755	Active	Retailer	Cust...	
	ABC Constructions		SADMIN	(908) 987-7678	Active	Customer	Cust...	
	AG Edwards & Sons, Inc	HQ-Distribution	Dynamic Investment Realty Corp.	SADMIN	(212) 489-1500	Active	Commercial	Cust...
	AH Frank	San Francisco	AG Edwards & Sons, Inc	SADMIN	(415) 555-1234	Active	Commercial	Cust...
	Aaron-Jones Dry Cleaning	San Francisco	SADMIN	(415) 555-605	Active	Customer	Cust...	
★	Abbey General Hospital	NJ	Premier Healthcare System	SADMIN	(973) 555-0000	Active	Customer	Cust...
	Abbot Designs	San Francisco	SADMIN	(650) 423-5244	Active	Subcontractor	Cust...	
	Abbot School for Medicine	San Francisco	SADMIN	(987) 098-2675	Active	Subcontractor	Cust...	



# Joins in Party Data (5)

## Explicit Join

Join definitions are created for establishing the relationship.

The screenshot displays four stacked tables representing join definitions:

- Join List:** Shows a single entry for 'Account' with 'Account' as the Name, 'CSSBCAccountSIS' as the Data Source, and 'Account' as the Project.
- Joins:** Shows a single entry for 'S\_ORG\_EXT' with 'Parent Account' as the Alias and a checked 'Outer Join Flag'.
- Join Specification List:** Shows a single entry for 'S\_ORG\_EXT' with 'Parent Account' as the Alias and a checked 'Outer Join Flag'.
- Join Definition:** Shows a single entry for 'Parent Account Id' with 'Parent Account Id' as the Source Field and 'PAR\_ROW\_ID' as the Destination Column.



# Joins in Party Data (6)

## Map Field to Party Table Column

Determine if a valid join definition already exists

If not, then create the required join

- ✓ Verify the relationship either 1:1 or M:1
- ✓ Identify the Foreign Key to the joined table
- ✓ Select / Create SVF for the Foreign Key column
- ✓ Provide ALIAS property for the join created
- ✓ Join specification is created

- Foreign Key field as Source Field
- Use PAR\_ROW\_ID as the Destination Column

Create a Single Valued Field mapping to the Party Table Column

- ✓ Join property will be updated with the ALIAS of the Join definition
- ✓ Select the appropriate Columns from the Column Property list
- ✓ Set the appropriate Type Property



# Knowledge Checks

**Answer the following:**

1. Implicit joins are used to bring data from Party Tables into Non-Party BC. State TRUE/FALSE ?
  - A. TRUE
  - B. FALSE
2. S\_CONTACT is a Party Table. State TRUE/FALSE ?
  - A. TRUE
  - B. FALSE



# Module Summary

**Now, you should be able to:**

- Describe party concept
- Identify party business components
- Define join and join specification to bring party data
- Explain how to map fields to party extension tables



# **Thank You**

# SIEBEL

Business Objects and Links

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Describe role of business object and links
- Explain how to create links and business objects



# Topic List

#No	Module Topics
1	Business Objects
2	Links
3	BO & Links Best Practice

# Topic List

#No	Module Topics
1	Business Objects
2	Links
3	BO & Links Best Practice

# Business Objects (1)

## Introduction

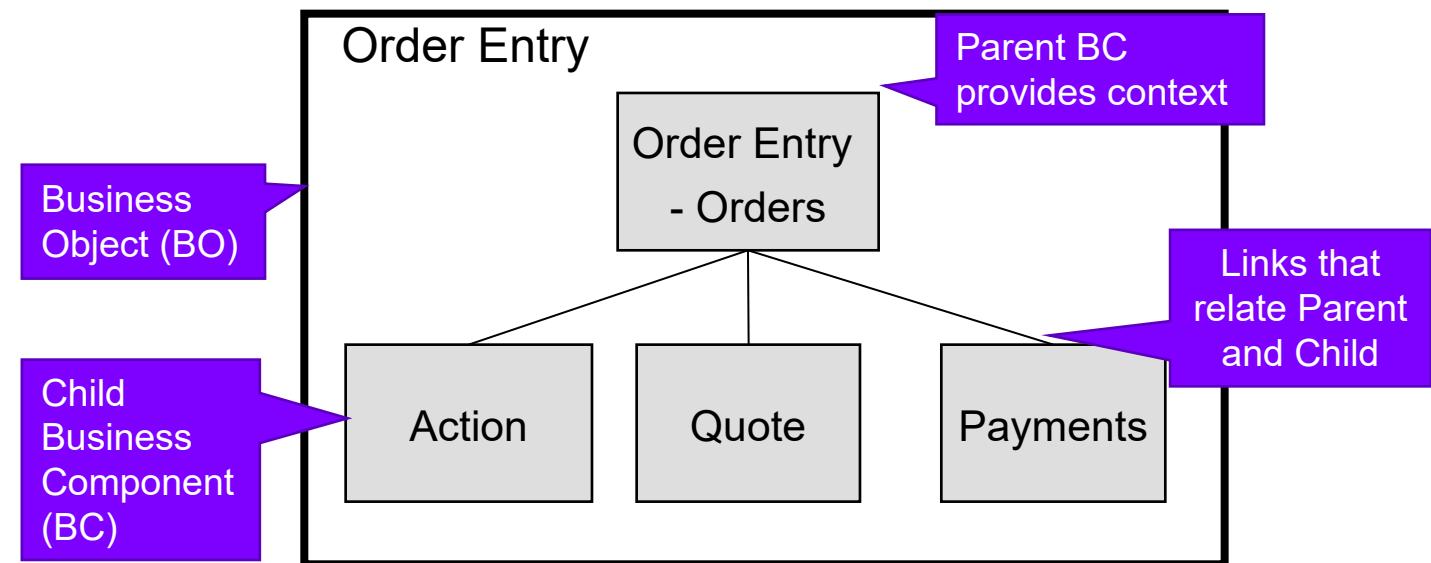
BO is collection of related business components

It represents a major business entity

It includes:

- ✓ Parent Business Component
- ✓ Child Business Components
- ✓ Links that relates Parent and Child Data

They provide foundation for views

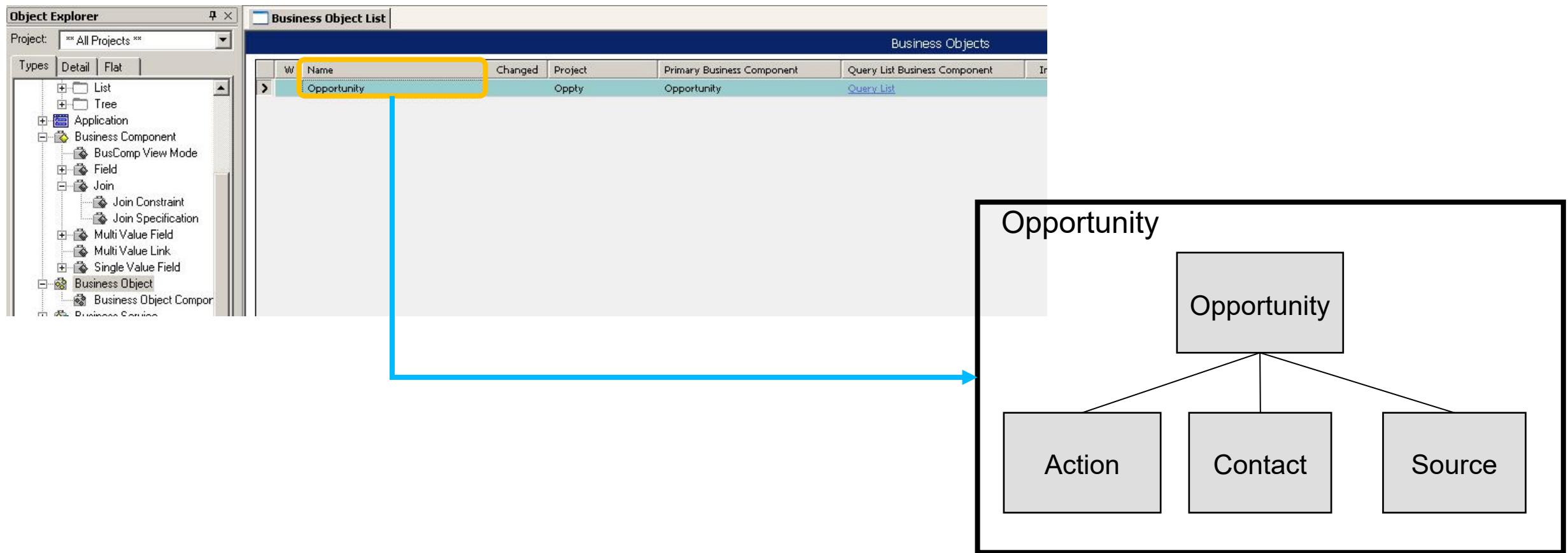


# Business Objects (2)

## Business Object Definition

Business Object Definition includes Name Property which is the Name of the BO

- ✓ Generally it is the Name of the Primary



# Business Objects (3)

## Business Object Component

It is child object type of the business object

It specifies the business component included in the business object

- ✓ **BusComp** property is the name of the business component
- ✓ **Link** property relates the parent and child BCs

The screenshot shows the SAP BusinessObjects Designer interface. On the left is the Object Explorer with a tree view of project components like Application, Business Component, and Business Object. The main area is the Business Object Component List, which displays a table of components. A specific row for 'Opportunity' is highlighted with a yellow box around its 'Primary Business Component' field, which contains 'Opportunity'. A purple callout points from this field to another purple callout below it, which says 'Parent BC has link property blank'. The bottom part of the screen shows the Business Object Components table, listing components like Account, Action, Opportunity, and Source, each with their respective Bus Comp and Link properties.

W	Name	Changed	Project	Primary Business Component	Query List Business Component	Inactive
>	Opportunity		Oppty	Opportunity	<a href="#">Query List</a>	

W	Bus Comp	Changed	Inactive	Link	Comments
	Account				Opportunity/Account
	Action				Opportunity/Action
>	Opportunity				Opportunity/Source
	Source				Added for palm products



# Topic List

#No	Module Topics
1	Business Objects
2	Links
3	BO & Links Best Practice

# Links (1)

## Link Definition

Link definition identifies which child data to fetch

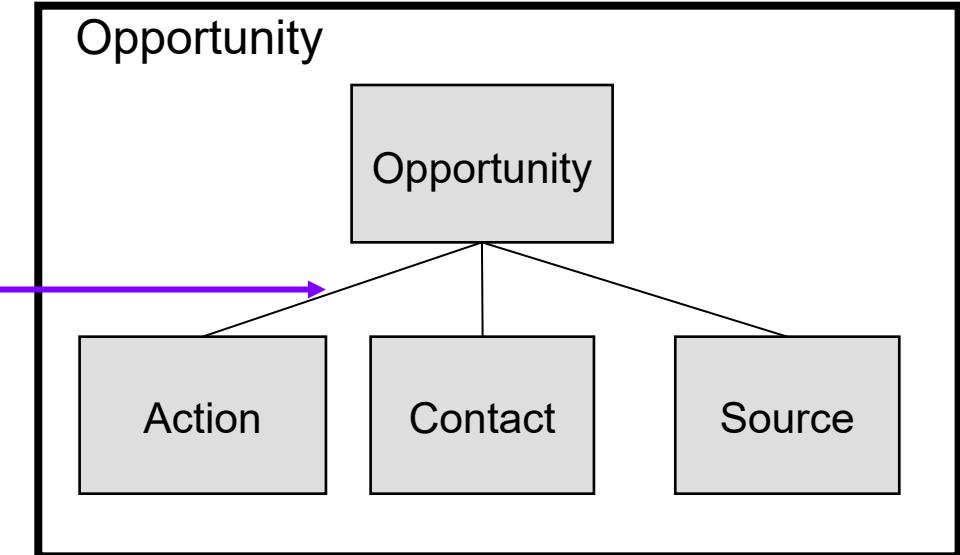
- ✓ Includes foreign keys to be populated when new child records are created

Link definition are used with both 1:M and M:M relationships between parent and child data

Business Object Component List				
W	Name	Changed	Project	Primary Business Component
>	Opportunity		Oppty	Opportunity
<				

W	Bus Comp	Changed	Inactive	Link
>	Action			Opportunity/Action



# Links (2)

## 1:M Link Definition

Identifies 1:M relationship between Parent and Child BC

Name is always defaulted to Parent BC / Child BC

The screenshot shows the 'Properties' window for a 'Link [Opportunity/Action]' object. The 'Alphabetic' tab is selected. Several properties are highlighted with blue boxes and annotated with purple callouts:

Property	Value
Cascade Delete	None
Child Business Component	Action
Comments	
Destination Field	Opportunity Id
Inactive	FALSE
Inter Child Column	
Inter Child Delete	FALSE
Inter Parent Column	
Inter Table	
Module	
<b>Name</b>	<b>Opportunity/Action</b>
No Associate	FALSE
No Delete	FALSE
No Insert	FALSE
No Inter Delete	FALSE
No Update	FALSE
Object Language Locked	
Object Locked	FALSE
Object Locked By Name	
Object Locked Date	
Parent Business Component	Opportunity
Primary Id Field	
Search Specification	
Sort Spec	
Source Field	
Visibility Auto All	
Visibility Rule Applied	Never
Visibility Type	

Annotations:

- A purple callout labeled "Name of Link" points to the "Name" property.
- A purple callout labeled "Child BC" points to the "Child Business Component" property.
- A purple callout labeled "FK in Child BC" points to the "Destination Field" property.
- A purple callout labeled "Defaults to Parent BC / Child BC" points to the "Name" property.
- A purple callout labeled "Parent BC" points to the "Parent Business Component" property.

# Links (3)

## Cascade Delete Property

The Cascade Delete property of a link determines if a child record is deleted when the parent record is deleted.

The Cascade Delete property on the link is set using following values :

Value	Description
<b>Delete</b>	If the parent record is deleted, then all child records are deleted.
<b>Clear</b>	If the parent record is deleted, the foreign key reference is removed and the value in the foreign key column is cleared.
<b>None</b>	If the parent record is deleted, no child records are deleted and no foreign key column is cleared. The default setting is None.

### Note:

- *Cascade Delete is not available for a many-to-many link.*
- *If set incorrectly, the Cascade Delete property might cause data integrity problems or orphaned records.*



# Links (4)

## M:M Link Definition

Identifies M:M relationship between Parent and Child BC

An intersection table is used to establish the link definition and resolve the relationship

### Inter Child Delete Property:

- ✓ Is TRUE: to delete the entry in both intersection table and child BC when the child record is deleted.
- ✓ Is FALSE: to delete the entry only in the intersection table and keep the child record

Always set  
Cascade  
Delete to  
None

Link [Opportunity/Source]	
	Alphabetic   Categorized
Cascade Delete	None
Child Business CompoSource	
Comments	
Destination Field	
Inactive	FALSE
Inter Child Column	SRC_ID
Inter Child Delete	FALSE
Inter Parent Column	OPTY_ID
Inter Table	S_OPTY_SRC
Module	
Name	Opportunity/Source
No Associate	FALSE
No Delete	FALSE
No Insert	FALSE
No Inter Delete	FALSE
No Update	FALSE
Object Language Loc	
Object Locked	FALSE
Object Locked By Na	
Object Locked Date	
Parent Business CompOpportunity	
Primary Id Field	Primary Source Id
Search Specification	
Sort Spec	Name
Source Field	
Visibility Auto All	
Visibility Rule Applied	
Visibility Type	

# Links (5)

## Links for Grandchild Data

To fetch Grandchild Data,

- Parent - Grandchild relationship is defined
- Example :
  - ✓ **Opportunity/Activity Plan**
  - ✓ **Activity Plan/Activity Plan Action**

Activity Plan Action is Grandchild of Opportunity

The screenshot shows the 'Business Object Component List' interface. It displays two main sections: 'Business Object Components' and 'Business Object Component List'. In the 'Business Object Components' section, there is a table with columns: W, Name, Changed, Project, and Primary Business Component. A single row is visible: 'Opportunity' under 'Name', 'Oppty' under 'Changed', and 'Opportunity' under 'Primary Business Component'. In the 'Business Object Component List' section, there is another table with columns: W, Bu, Comp, Changed, Inactive, Link. Three rows are listed: 'Opportunity' (highlighted with a blue border), 'Activity Plan', and 'Activity Plan Action'. To the right of these rows, their respective links are shown: 'Opportunity/Activity Plan' and 'Activity Plan/Activity Plan Action'.

The screenshot shows the 'Opportunities' page in a CRM system. The top navigation bar includes 'Home', 'Accounts', 'Contacts', 'Administration - Product', 'Opportunities' (highlighted with a blue border), 'Opportunity Explorer', 'Manager's Explorer', 'Opportunities Home', 'Opportunities List', and 'Opportunities'. The main content area is titled 'Term Life Insurance'. It contains several input fields: 'Name' (Term Life Insurar), 'Last Name' (Aamos), 'Account' (Ace Corporat), 'First Name' (James), 'Revenue' (\$50,000.00), 'Sales Team' (SADMIN), 'Currency' (USD), and 'Sales Stage' (Submitted). Below these fields are tabs: 'Assignment Skills', 'Registrations', 'More Info', 'Capture', 'Activities', 'Activity Plans' (highlighted with a blue border), and 'Asses'. Under the 'Activities' tab, there is a table with columns: Description, Type, Start, End, Status, Priority, and Cost. Under the 'Activity Plans' tab, there is a table with columns: Planned Start, Template, and Description.

# Topic List

#No	Module Topics
1	Business Objects
2	Links
3	BO & Links Best Practice

# BO & Links Best Practice (1)

## Business Object

Before creating a New BO

- Check if a BO exists for the BC
- If not, then create a new record in Business Object Definition
  - ✓ Name property – Name of the BO
  - ✓ Project property – Name of the Project

Before adding the BC to BO

- Check if the BC already exists under the Business Object Components Definition while adding for existing BO
- If not, then create a new record
  - ✓ Bus Comp Property – Name of the BC
  - ✓ Link Property – Name of the Link
    - Check if the Link exists between the BC and the Parent BC
    - If Primary BC, then leave the Link Property blank



# BO & Links Best Practice (2)

## Links

Decide the relationship between the Parent and Child BC

- 1:M or M:M relationship

Resolve the Foreign Keys to set up the relationship

Create a new record in Link Definition in Tools

- Parent Business Component property – Parent BC name
  - Child Business Component property– Child BC name
  - Name Property – is auto filled
  - In case of 1:M Link
    - ✓ Source Field – Primary Key of Parent BC
    - ✓ Destination Field – Foreign Key from Child BC
  - In case of M:M Link
    - ✓ Inter Table
    - ✓ Inter Parent Column
    - ✓ Inter Child Column
- ✓ Properties must be filled

# Knowledge Checks

## True or False?

1. 1:M Link requires an Intersection table to establish the relationship between parent and child BC. True / False.
  - A. TRUE
  - B. FALSE
2. Business Objects definition includes a Primary BC and Multiple Child BCs and Links that relate these BCs. True / False.
  - A. TRUE
  - B. FALSE



# Module Summary

**Now, you should be able to:**

- Describe role of business object and links
- Explain how to create links and business objects



# **Thank You**

# SIEBEL

Business Components, Fields,  
User Properties

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Explain how to:
  - Edit business component properties
  - Edit field properties
- Identify user properties in Application, Applets, BC and Fields



# Topic List

#No	Module Topics
1	Business Component Properties
2	Field Properties
3	User Properties
4	Additional User Properties

# Topic List

#No	Module Topics
1	Business Component Properties
2	Field Properties
3	User Properties
4	Additional User Properties

# Business Component Properties (1)

## BC Properties

Business Components have set properties that can prevent from Deletion, Insertion, Merging and Updating records

- ✓ Example: Once pricelist have been created, they cannot be deleted or changed

These properties apply to all interaction with the Business component via applets , workflows etc.

**Owner Delete Property:** When set to TRUE enables only the record owner to delete the record

- If the owner is a team then the record can be deleted only by the team member who is primary

Properties	
Business Component [Price List]	
Alphabetic Categorized	
Basic	
Class	CSSBCPriceList
Comments	The search spec wa
Inactive	FALSE
Module	
Name	Price List
No Delete	TRUE
No Insert	TRUE
No Merge	TRUE
No Update	TRUE
Object Language	
Object Locked	FALSE
Object Locked By	
Object Locked Da	
Owner Delete	FALSE
Scripted	FALSE
Search Specification	Type='PRICE LIST'
Sort Specification	
Table	S_PRI_LST
Misc	
Cache Data	FALSE
Data Source	
Dirty Reads	TRUE
Distinct	FALSE
Enclosure Id Field	
Extension Type	
Force Active	FALSE
GenReassignAct	FALSE
Hierarchy Parent F	
Insert Update All C	FALSE
Log Changes	TRUE

# Business Component Properties (2)

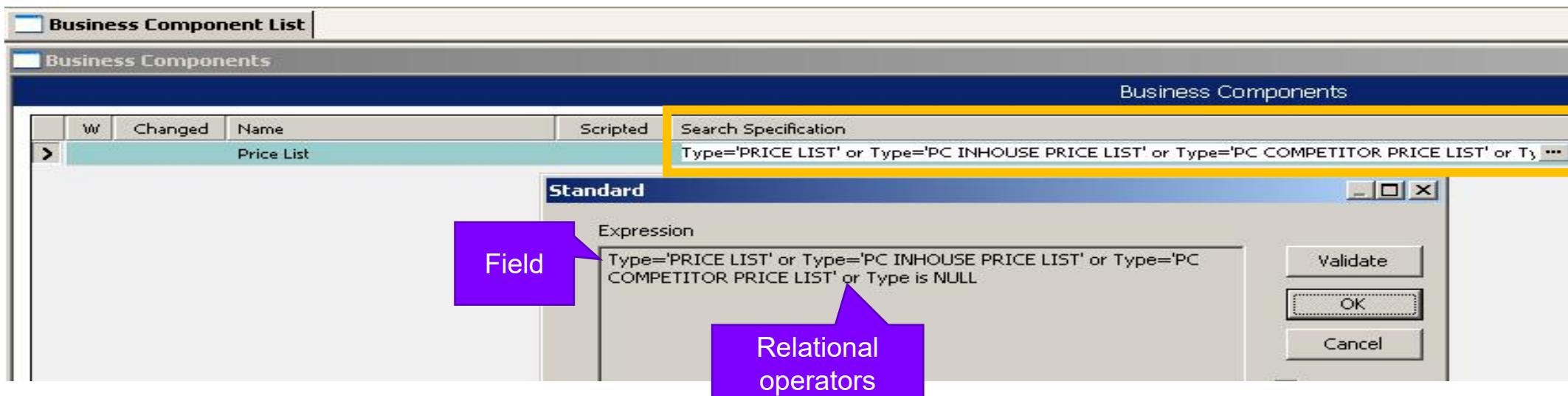
## BC Properties

**Search Specification** : Enables a specific condition-based record fetching

- ✓ Consists of fields, operators, constants, functions, profile attributes
- ✓ Typically used when multiple BCs based on same table are involved

Search Spec at Applet and at BC gets ANDed to fetch the resultant set and the Applet displays it

- ✓ They generate a WHERE clause that is run on the DB layer (in SQL)



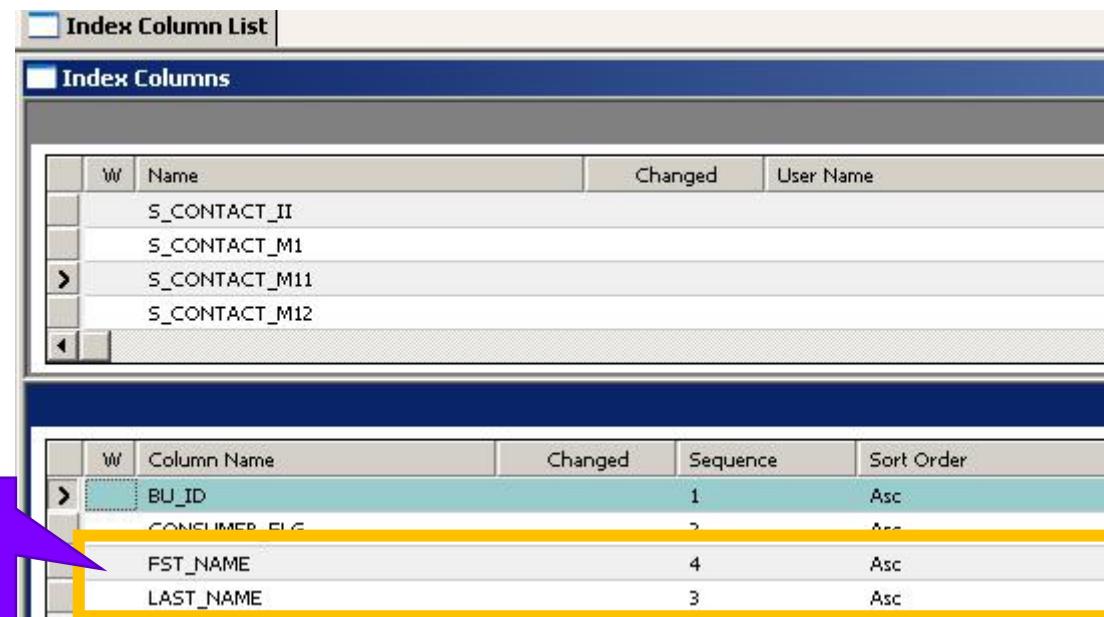
# Business Component Properties (3)

## BC Properties

**Sort Specification** Determines the sorted order for the retrieved records

- ✓ Use DESC to sort in reverse order

Cannot be set at Applet level



Ensure that Index exist to support the Sort specification

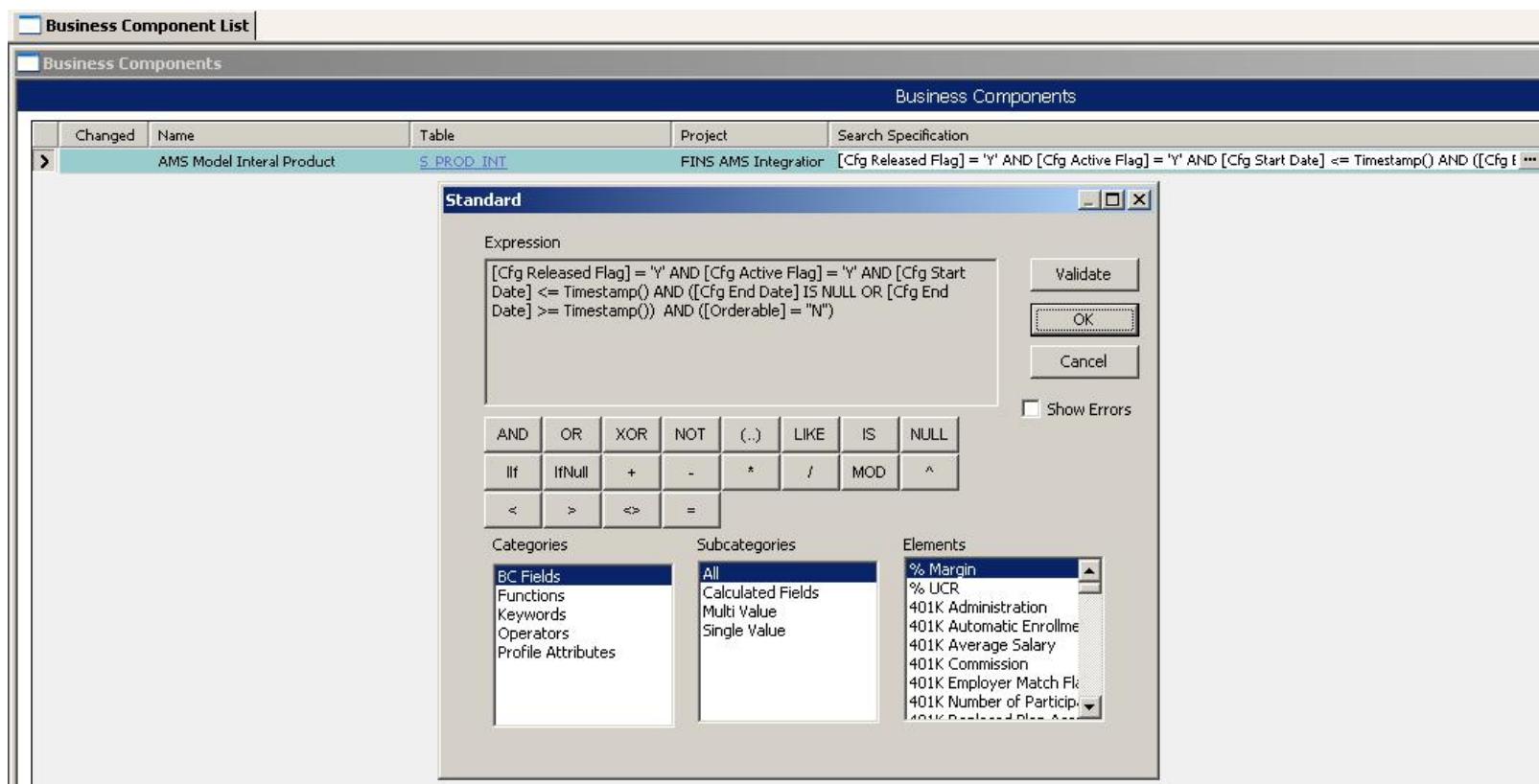
Fields in the BC

# Business Component Properties (4)

## Expression Builder

Expression builder is used to build expressions for search spec and Sort spec

- ✓ Useful in finding syntax errors
- ✓ Can validate the expression
- ✓ Fields, operators, functions etc are available and can be used to build the deserved expression



# Topic List

#No	Module Topics
1	Business Component Properties
2	Field Properties
3	User Properties
4	Additional User Properties

# Field Properties (1)

## Properties

**Required:** Is set to TRUE to make the field required at Business Layer

- Asterisk in red is displayed along with the caption of the field

**Read Only:** When set to TRUE, the field will remain non-editable

- Such fields are greyed out on the UI

**Force Case:** Specifies the case for the field

- Can validate these values only : UPPER, LOWER, FIRSTUPPER

The screenshot shows the Siebel Field List interface. On the left is the Object Explorer with a tree view of Siebel Objects like Applet, Application, and Business Component. The main area is titled 'Field List' and contains two tabs: 'Fields' and 'Columns'. In the 'Fields' tab, a table lists fields for the 'Account' table, specifically the 'S\_PARTY' column. The 'Required' checkbox is checked, indicated by a red asterisk (\*) next to the field name. A blue arrow points from this checked 'Required' checkbox to a corresponding red box highlighting the red asterisk on the 'Account Name' field in a floating UI window. The floating window shows the 'Account Name' field with the value '3 Com' and other fields for Address, City, and Zip Code.



# Field Properties (2)

## Properties

**Validation:** Restricts the values that can be entered for a field which is single value

- Does not support Multi Value Fields validation
- Only when the field is modified in the GUI, it is evaluated

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer tree view is open, showing categories like Siebel Objects, Applet, Application, Business Component, and Single Value Field. The 'Single Value Field' node is selected. The main workspace displays two tables: 'Single Value Field List' and 'Single Value Fields'.

**Single Value Field List:** This table lists fields by Name (e.g., Quote, S\_DOC\_QUOTE, Quote, CSSIABCQuote). The 'Name' column is highlighted with a yellow box.

Name
Quote
S_DOC_QUOTE
Quote
CSSIABCQuote

**Single Value Fields:** This table lists validation rules for various fields. A yellow box highlights the 'Validation' column.

Name	Validation
Approved Date	>= [Created] and >= Today()
Departure Date	>= [Arrival Date]
Discount	>= 0 AND <= 100
Discount Percent MRC	>= 0 AND <= 100
Due Date	= Default Value

# Field Properties (3)

## Properties

**Validation Message:** Custom message which is displayed when the validation property is violated

- Message display mode needs to be selected appropriately for which option to be displayed

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer tree view is open, showing categories like Siebel Objects, Applet, Application, and Business Component. Under Business Component, several sub-options are listed. The main workspace contains two tabs: 'Field List' (selected) and 'Fields'. The 'Field List' tab shows a single entry for 'Opportunity' under the table 'S\_OPTY'. The 'Fields' tab displays a table of fields with columns: W, Name, Column, Validation, Validation Message, and Message Display Mode. A yellow box highlights the 'Validation Message' column. The table data is as follows:

W	Name	Column	Validation	Validation Message	Message Display Mode
>	# of Days	NUM_DAY_DUR	> 0		User Msg with Error Code Only
	Discount %	DISCOUNT_PCT	>= 0 AND <= 100		User Msg
	Expiration Date	EXPIRATION_DT	>=[Submitted Date]		User Msg with Error Code Only
	Hospitality Close Date	DEPARTURE_DT	>= [Hospitality Start Date]		User Msg with Error Code/Msg
	Hospitality Start Date	ARRIVAL_DT	>= Today()		User Msg with Error Code Only

# Field Properties (4)

## Properties

**Predefault Value:** Default value assigned to field upon New Record creation

- Can be a constant value or expression
  - ✓ Expression can consists of fields from same BC or Parent BC

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer tree is visible, showing categories like Siebel Objects, Application, Business Component, and others. The main area is titled 'Field List' and contains a table titled 'Fields'. The table has columns for Name, Table, Changed, Project, Cache Data, Class, and Data Source. One row is selected for 'Opportunity' (Table: S\_OPTY, Project: Oppy, Class: CSSBCFINOppty). Below this, another table titled 'Fields' shows detailed properties for various fields of the Opportunity object. A yellow box highlights the 'Predefault Value' column, which contains expressions such as 'Parent: 'Contact.Account Primary Ship To Address Id'', 'Expr: 'LookupValue("SHM\_BUSINESS\_TYPE\_CD", "Local")'', and 'Expr: 'OrganizationName()''. Two purple callout boxes point to these expressions: one labeled 'Parent BC Fields' pointing to the first two examples, and another labeled 'Expression' pointing to the last example.

Name	Column	Predefault Value
Account Primary Ship To Address Id	PR_SHIP_ADDR_ID	Parent: 'Contact.Account Primary Ship To Address Id', 'Account.Primary Ship To Address Id'
Account Primary Ship To Person Id	PR_SHIP_PER_ID	Parent: 'Contact.Account Primary Ship To Person Id', 'Account.Primary Ship To Person Id'
Application Flag	CONSUMER_OPTY_FLG	N
Assignment Excluded	ASGN_USR_EXCLD_FLG	N
Business Type	BUSINESS_TYPE_CD	Expr: 'LookupValue("SHM_BUSINESS_TYPE_CD", "Local")'
CloseOut Flg	INACTIVE_FLG	N
Created At		
Created By Name	LOGIN	Expr: 'OrganizationName()'



# Field Properties (5)

## Properties

**Postdefault Value:** Assigns a value to the Field only when the user does not enter the value before the record is saved to the Database.

- Expression will have same syntax as the Predefault

The screenshot shows the Siebel Object Explorer and Field List windows. The Object Explorer window on the left displays a tree structure of Siebel Objects, including Applet, Application, and Business Component. The Business Component node has a 'Field' item selected. The Field List window on the right shows a table of fields. A yellow box highlights the 'Post Default Value' column for the 'Created At' field, which contains the expression 'Expr: 'OrganizationName()''.

Name	Column	Post Default Value
Created At	REVN_SPLIT_FLG	Expr: 'OrganizationName()'
Split Flag		N

# Field Properties (6)

## Calculated Field

An expression for calculating the value of a specific field is specified by the calculated value property.

- The calculated value and properties of field validation are restricted to 255 characters.
- Column property remains blank, which means these are not stored in database
- Calculated field is evaluated at the run-time and application does not validate the values that these fields contain
- Sorting of calculated fields is not possible
- Allows queries on calculated fields.
- Calculated field may be based on the results of another calculated field lying in the same business component
- Created by setting calculated property as TRUE and providing the expression in calculated value
- Read only fields, can lead to performance issues

The screenshot shows the Siebel Object Explorer interface. On the left is the Object Explorer window with a tree view of Siebel Objects like Applet, Application, and Business Component. The main area is the Field List window, which displays a table of fields for the Opportunity table. One row is selected, showing the field name 'Created At' and its properties. A yellow box highlights the 'Calculated' column, which is checked, and the 'Calculated Value' column, which contains the expression 'OrganizationName'. Below the Field List is the Fields window, showing a list of fields including 'Split Flag' and 'REVN\_SPLIT\_FLG'.

W	Name	Table	Changed	Project	Cache Data	Class
>	Opportunity	S_OPTY		Opty		CSSBCFINOppty

W	Name	Column	Calculated	Calculated Value
>	Created At		<input checked="" type="checkbox"/>	OrganizationName
	Split Flag	REVN_SPLIT_FLG		

# Topic List

#No	Module Topics
1	Business Component Properties
2	Field Properties
3	User Properties
4	Additional User Properties

# User Properties (1)

## About User Properties

These are object definitions which are used for configuring specialized behaviour beyond what is configured in the parent object definition's properties. These user properties can be configured for object types:

- Applet
- Application
- Assignment
- Business Component
- Business Service Method Arg
- Business Service
- Control
- Field
- Integration Object
- Integration Component Field
- Integration Component
- List Column
- View
- Virtual Business Component



# User Properties (2)

## User Properties of Business Component

**BC Read Only Field:** It specifies the business component field which determines whether a record is read-only

- The current record is read-only when the value that this field contains is TRUE,

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer tree is visible with nodes like Siebel Objects, Applet, Application, Business Component, etc. The main window title is "Business Component User Prop..." and it displays the "Business Component User Properties" screen. This screen has two tabs: "Business Components" and "Business Component User Properties". The "Business Components" tab is selected, showing a list of records with columns: W, Name, Changed, Project, Cache Data, Class, Data Source, Dirty Reads, and Dir. The records listed are: LS Clinical Action (Project: LS Clinical Activity, Class: CSSBCCLSiteCalendarActivity), LS Clinical Contract (Project: LS Clinical Protocol SII, Class: CSSBCCLContract), LS Clinical Protocol Versions (Project: LS Clinical Protocol, Class: CSSBCBase), and LS Clinical Site Contract (Project: LS Clinical Protocol SII, Class: CSSBCBase). The "Business Component User Properties" tab is also shown below. A purple callout box points to the "BC Read Only Flag" row in the "Business Component User Properties" table, which has columns: W, Name, Changed, Value, Inactive, and Comments. The row for "BC Read Only Flag" has the value "BC Read Only Flag" in the "Value" column. The callout box contains the text: "BC Record becomes non-editable when the field(BC Read Only Flag) is set with a value".

W	Name	Changed	Project	Cache Data	Class	Data Source	Dirty Reads	Dir
>	LS Clinical Action		LS Clinical Activity		CSSBCCLSiteCalendarActivity		✓	
	LS Clinical Contract		LS Clinical Protocol SII		CSSBCCLContract			
	LS Clinical Protocol Versions		LS Clinical Protocol		CSSBCBase		✓	
	LS Clinical Site Contract		LS Clinical Protocol SII		CSSBCBase		✓	

W	Name	Changed	Value	Inactive	Comments
>	BC Read Only Field		BC Read Only Flag		



# User Properties (3)

## User Properties of Business Component

BC Read Only Flag

The screenshot shows the Siebel Object Explorer interface. The left pane displays a tree view of Siebel objects, including Applet, Application, Business Component, and various BusComp sub-components like Browser Script, Server Script, View Mode, and User F. The right pane is titled 'Field List' and shows a table of fields under the 'Business Components' category. One row is selected, highlighting the 'BC Read Only Flag'. A purple callout box points to the 'Calculated Value' column, which contains the expression: `[Status] = LookupValue("EVENT_STATUS","Submitted") or (AMS Activity Id IS NOT NULL and [Activity Code]<>'CR')`. The table has columns: W, Name, Table, Changed, Project, Cache Data, Class, and Data Source.

W	Name	Table	Changed	Project	Cache Data	Class	Data Source
>	LS Clinical Action	S_EVT_ACT		LS Clinical Activity			
	LS Clinical Contract	S_DOC_AGREE		LS Clinical Protocol Sil			
	LS Clinical Protocol Versions	S_PTCL_VER_LS		LS Clinical Protocol			
	LS Clinical Site Contract	S_AGREE_SITE_LS		LS Clinical Protocol Sil			

W	Name	Changed	Calculated	Calculated Value
>	BC Read Only Flag		✓	<code>[Status] = LookupValue("EVENT_STATUS","Submitted") or (AMS Activity Id IS NOT NULL and [Activity Code]&lt;&gt;'CR')</code>

The calculated expression is evaluated and the calculated field (BC Read Only Flag) is set

# User Properties (4)

## Field User Properties

**Required:** makes the field a required field under certain conditions

**Text Length Override:** specifies the text length of the field, rather than that of the database column, defines the maximum field length

The screenshot shows the Siebel Object Explorer on the left and the Field User Prop editor on the right.

**Object Explorer:**

- Project: All Projects
- Types: Siebel Objects, Applet, Application, Business Component, Field
- Business Component sub-items: BusComp Browser Script, BusComp Server Script, BusComp View Mode, Business Component User F
- Field sub-items: Field Locale, Field User Prop

**Field User Prop Editor:**

**Fields Tab:**

W	Name	Changed	Calculated	Calculated Value
	Point Id			
	Point Internal Name			
>	Point Name			
	Point New Accrual Total			
<				

**Field User Props Tab:**

W	Name	Changed	Value
>	Required		[PointNameRequired] = "Y"

A purple callout points from the text "Point Name becomes mandatory when the field PointNameRequired = "Y"" to the "Required" row in the Field User Props table.

# Topic List

#No	Module Topics
1	Business Component Properties
2	Field Properties
3	User Properties
4	Additional User Properties

# Additional User Properties (1)

## Application User Properties

**ClientBusinessService0**: user property calls a business service from a browser script

**PDQDisabledView0**: user property disables the Predefined Query (PDQ) dropdown list for a view

**OverrideViewCache0**: user property disables caching for a view

The screenshot shows the Siebel Object Explorer interface. The left pane displays a tree structure of Siebel Objects, including Applet, Application, Application Browser Script, Application Event Services, Application Find, Application Locale, Application Server Script, Application User Prop (which is selected), Page Tab, Screen Menu Item, Business Component, and Business Object. The right pane shows two tables: 'Application User Property List' and 'Application User Props'. The 'Application User Property List' table has columns: W, Name, Changed, Project, Menu, Scripted, and Acknowledgment. It contains one row for 'Siebel Universal Agent' with values: W, Siebel Universal Agent, Siebel Universal Agent, Generic WEB, and several empty fields. The 'Application User Props' table has columns: W, Name, Changed, Value, Inactive, and Comments. It contains two rows: 'ClientBusinessService0' with Value 'Message Bar' and 'PDQDisabledView0' with Value 'Order History View (eSales)'. A purple callout points to the 'ClientBusinessService0' row with the text 'Message Bar Business Service is accessible in the Browser Script'. Another purple callout points to the 'PDQDisabledView0' row with the text 'PDQ is disabled in this View'.

W	Name	Changed	Project	Menu	Scripted	Acknowledgment
>	Siebel Universal Agent		Siebel Universal Agent	Generic WEB		

W	Name	Changed	Value	Inactive	Comments
>	ClientBusinessService0		Message Bar		
	PDQDisabledView0		Order History View (eSales)		

# Additional User Properties (2)

## Applet User Properties

**DisableNewRecord:** Siebel CRM calling NewRecord in the current applet is prevented by this user property

**CanInvokeMethod:** *MethodName*: user property enables or disables a method or a button

The screenshot shows the Siebel Object Explorer and the Applet User Property List window.

**Object Explorer:** Shows the project list as "All Projects" and the object types: Siebel Objects, Applet, Applet Browser Script, Applet Locale, Applet Message, Applet Method Menu Item, Applet Server Script, Applet Toggle, Applet User Prop, Applet Web Template, Chart, and Control.

**Applet User Property List:** Shows the "Applet User Properties" table with the following data:

Name	Changed	Project	Business Component
Account Contact Read Only List Applet		eAuto CF Bankruptcy	Contact
Account Contact User Admin List Applet		Admin	Person
Account Contact User Admin List Applet (Delete)		eApp Admin	Person
Account Coverage Company List Applet		Account (SSE)	Position

A purple callout box points to the "Disable CopyRecord" row in the "Applets" section, stating: "Copy Record method is Disabled Always".

Name	Changed	Value
Disable CopyRecord		Y



# Additional User Properties (3)

## Applet User Properties

The screenshot shows the Siebel Object Explorer interface. On the left, under the 'Siebel Objects' section, the 'Applet' category is expanded, and 'Applet User Prop' is selected. A purple callout box points from this selection to a text annotation at the bottom left.

Object Explorer

Project: \*\* All Projects \*\*

Types Detail Flat

Siebel Objects

- Applet
  - Applet Browser Script
  - Applet Locale
  - Applet Message
  - Applet Method MenuItem
  - Applet Server Script
  - Applet Toggle
  - Applet User Prop
- Applet Web Template
- Chart

AddRecords Method will be invoked when Active field has a value

Applet User Property List

Applet User Properties

Applets

	W	Name	Changed	Project	Business Component
>		Agreement Asset Assoc List Applet		Asset Contract Task	Asset Mgmt - Asset

CanInvokeMethod: AddRecords [Active]

EnableStandardMethods Y

High Interactivity Enabled Y



# Additional User Properties (4)

## Business Component User Properties

**BC Read Only Field:** specifies the field of business component which determines whether a record is read-only

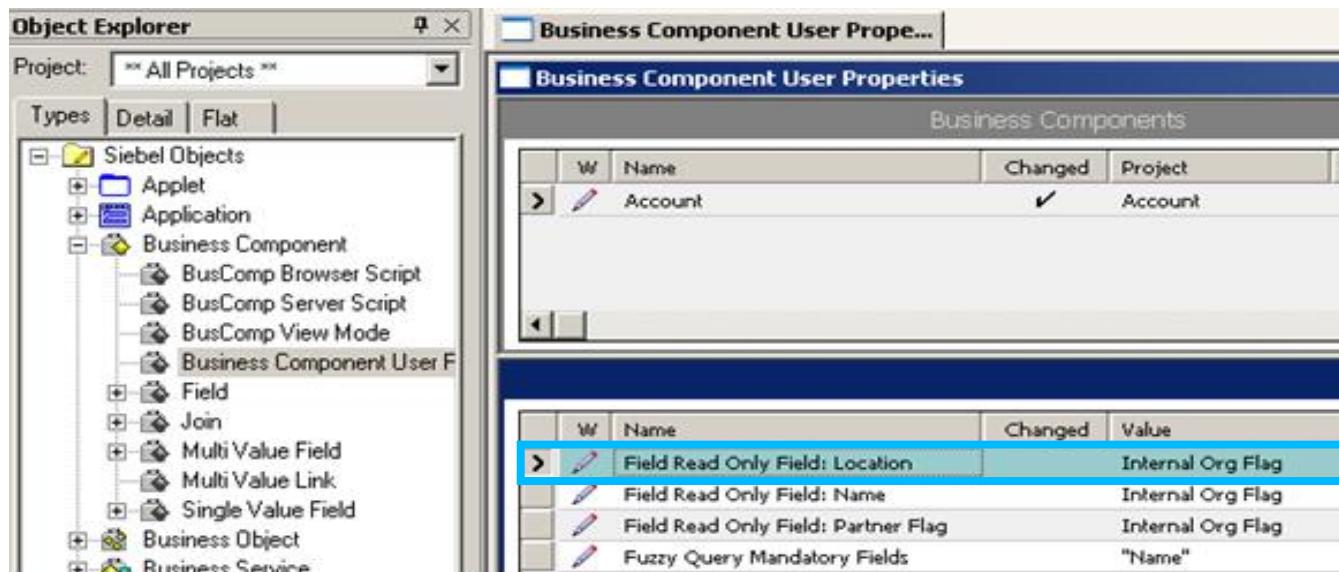
- ✓ The current record is read-only if the value that this field contains is TRUE.

**Field Read Only Field: *fieldname*:** sets a business component field to read-only

**Named Method *n*:** calls a business component or business service method, or sets a field value.

**On Field Update Invoke *n*:** calls a business component method when Siebel updates a field

**On Field Update Set *n*:** sets the value of a business component field when Siebel updates another field



Location field is non-editable when the Field Internal Org Flag is set

# Additional User Properties (5)

## Business Component User Properties

The screenshot shows the Siebel Object Explorer on the left and the Business Component User Properties screen on the right. The Object Explorer lists various Siebel object types, including Applet, Application, and Business Component. The Business Component category is expanded, showing sub-options like BusComp Browser Script, BusComp Server Script, BusComp View Mode, and Business Component User Properties. The Business Component User Properties option is selected.

**Business Component User Properties Screen:**

- Business Components Table:** Shows a single record named "Quote" with Project set to "Quote" and Class set to "CSSSIABCQuote".
- Business Component User Properties Table:** Shows two rows of properties:
  - Row 1: Name is "Named Method 1", Value is "AuthorizeAll", "INVOKEALL", "Payments", "CreditCardAuthentication".
  - Row 2: Name is "On Field Update Invoke 1", Value is "Price List", "Quote Item", "CalculatePriceAll".
  - Row 3: Name is "On Field Update Invoke 10", Value is "Status", "Quote", "Update Opty Status", "[Projected Property Flag] = 'Y'", "Requested Ship Date", "Quote Item", "UpdateFulfilmentField".

**Annotations:**

- A purple callout points from the "Business Component User Properties" section to the "On Field Update Invoke 1" row in the table, with the text: "Syntax for On Field Update Invoke : "[FieldToCheck]", "[BusCompName]", "[MethodName]", "[Condition]"
- A purple callout points from the "Business Component User Properties" section to the "On Field Update Invoke 10" row in the table, with the text: "When Status field is updated and if [Projected Property Flag] = "Y", Then BC – Quote, Method – Update Opty Status is invoked"
- A purple callout points from the "Business Component User Properties" section to the "Named Method 1" row in the table, with the text: "When AuthorizeAll method is invoked it shall invoke the Payments BC CreditCardAuthetication method for that record"



# Knowledge Checks

## Answer the following

1. No Insert, No Delete, No Update and No Merger properties are available for both applet and BC object definitions. State TRUE / FALSE
2. Postdefault and Predefault value are set for a field. The user creates a new record and does not provide the value to this field and steps off. What value would be stored in this field?
3. Postdefault and Predefault value are set for a field. The user edits the above record and removes the values stored in the field. What value would be stored in this field?
4. Sorting is allowed in Calculated Fields? State TRUE/ FALSE
5. BC Read Only Field user property is defined under which object definition?
6. Required property when set to TRUE makes the field required? State TRUE/ FALSE



# Module Summary

**Now, you should be able to:**

- Explain how to:
  - Edit business component properties
  - Edit field properties
- Identify user properties in Application, Applets, BC and Fields



# **Thank You**

# SIEBEL

Creating New BC

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Define 1:M relationship using child Business Component
- Explain how to:
  - create a new BC using 1:M extension table
  - add the BC to the Business Object



# Topic List

#No	Module Topics
1	Business Challenge
2	Enterprise Solution
3	Creating a New BC
4	Assign BC to Business Object

# Topic List

#No	Module Topics
1	Business Challenge
2	Enterprise Solution
3	Creating a New BC
4	Assign BC to Business Object

# Business Challenge

## Overview

Siebel provides Business Components to capture details of most common entities , but not all possibilities

- Example, Sales Organization might require details of the Contact
  - Educational background Like, Highest education qualification , which colleges attended , Marks scored , Hobbies , Sports played , Awards and recognition received
  - Details of the favourite Food and restaurant Like Name, location , type of cuisine , cost , ambience etc..

To capture these information :

- More fields are required
- 1:M relationship to the Contact BC as parent is required



# Topic List

#No	Module Topics
1	Business Challenge
2	Enterprise Solution
3	Creating a New BC
4	Assign BC to Business Object

# Enterprise Solution (1)

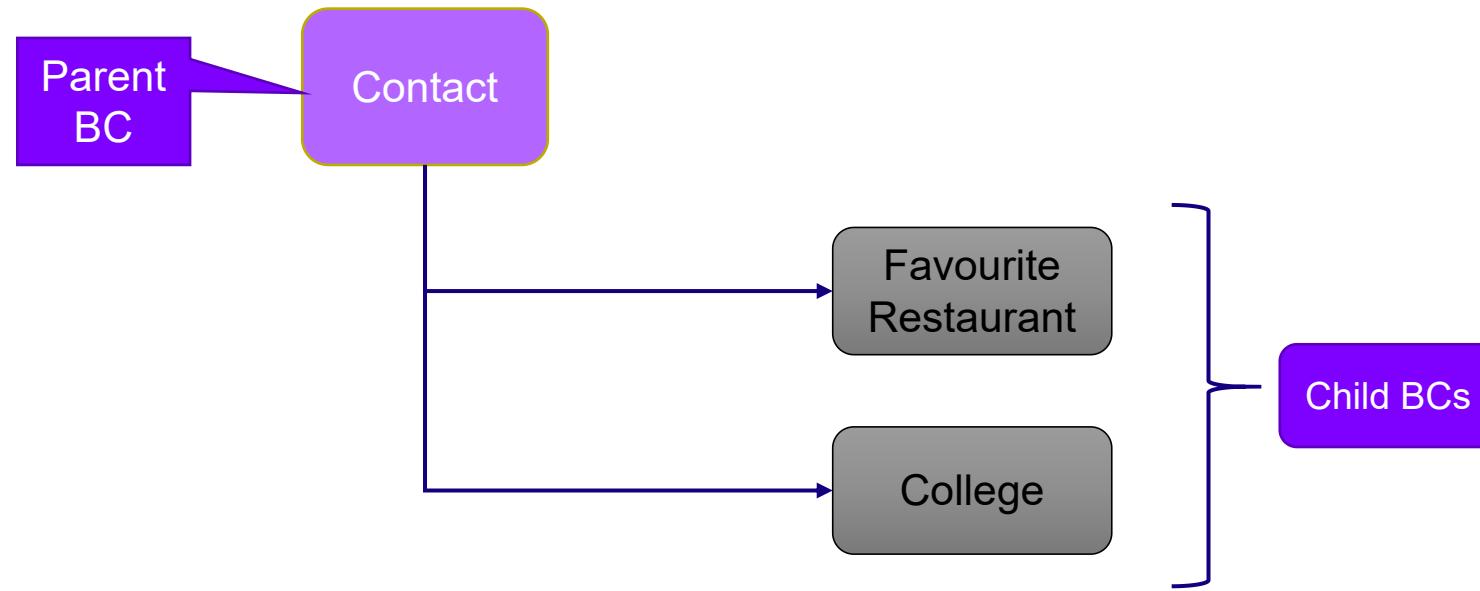
## Overview

To capture those extra details, Create a new BC which is child to the existing BC (Contact)

Based on the existing 1:M extension table supplied by the Siebel Data Model

- ✓ Example: Create 2 New BC as child to Contact BC

- Favourite Restaurant BC
- College BC



# Enterprise Solution (2)

## 1:M Extension Table

To base the BC on a 1:M extension table , Check if the table already exists in the repository

These extension tables have predefined columns ATTRIB\* to store data of various data types

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer tree is visible, with the 'Table' node expanded to show 'Column'. The main window is titled 'Column List' and contains a 'Columns' tab. A callout bubble points to the 'Name' column in the table, which is highlighted with a blue border and contains the value 'S\_CONTACT\_XM'. The table has columns for 'Module', 'Object Language Locked', 'Name', 'Object Locked', and 'Object Locks'. Below this, another table titled 'Columns' is shown, listing 14 attributes from 'ATTRIB\_01' to 'ATTRIB\_14' with their respective details.

Requir	W	Name	User Name	Alias	Type	Foreign Key Table	Primary Key	Physical Type	Nullable	Length	Precision
		ATTRIB_01	Attribute 01		Data (Public)			Varchar	✓	100	
		ATTRIB_02	Attribute 02		Data (Public)			Varchar	✓	100	
		ATTRIB_03	Attribute 03		Data (Public)			Varchar	✓	30	
		ATTRIB_04	Attribute 04		Data (Public)			Varchar	✓	30	
		ATTRIB_05	Attribute 05		Data (Public)			Varchar	✓	30	
		ATTRIB_06	Attribute 06		Data (Public)			Varchar	✓	30	
		ATTRIB_07	Attribute 07		Data (Public)			Varchar	✓	30	
		ATTRIB_08	Attribute 08		Data (Public)			Character	✓	1	
		ATTRIB_09	Attribute 09		Data (Public)			Character	✓	1	
		ATTRIB_10	Attribute 10		Data (Public)			Character	✓	1	
		ATTRIB_11	Attribute 11		Data (Public)			Character	✓	1	
		ATTRIB_12	Attribute 12		Data (Public)			Date Time	✓	7	
		ATTRIB_13	Attribute 13		Data (Public)			Date Time	✓	7	
		ATTRIB_14	Attribute 14		Data (Public)			Number	✓	22	22

# Enterprise Solution (3)

## 1:M Extension Table

Contain 3 User key columns :

- ✓ NAME
- ✓ TYPE
- ✓ PAR\_ROW\_ID

- These collectively must be always unique for a record

Type column can be used to store the type of the record

- ✓ Example: if the record type is favourite restaurant or College details

The screenshot shows the SAP Column List interface. At the top, there's a toolbar with 'Column List' and buttons for 'Extend', 'Apply/DDL', and 'Activate'. Below the toolbar, the table name 'S\_CONTACT\_XM' is displayed. The main area is titled 'Columns' and contains a grid of columns. A purple arrow points from the text 'User Keys' to the 'Name' column, which is highlighted with a blue border. The 'User Key Sequence' column also has a blue border around its values (2, 1, 3). The grid includes columns for 'Requir', 'W', 'Name', 'User Name', 'Alias', 'Type', 'User Key Sequence', 'Primary Key', and 'Physical Type'. Other columns shown include 'MODIFICATION\_', 'LAST\_UPD\_BY', 'LAST\_UPD', and 'LAST\_UPDATED'.

Requir	W	Name	User Name	Alias	Type	User Key Sequence	Primary Key	Physical Type
✓		TYPE	Type		Data (Public)	2		Varchar
✓		ROW_ID	Row Id		System		✓	Varchar
✓		PAR_ROW_ID	Parent Row Id		System	1		Varchar
✓		NAME	Name		Data (Public)	3		Varchar
		MODIFICATION_	Modification Number		System			Number
		LAST_UPD_BY	Last Updated By		System			Varchar
		LAST_UPD	Last Updated		System			UTC Date Time

# Topic List

#No	Module Topics
1	Business Challenge
2	Enterprise Solution
3	Creating a New BC
4	Assign BC to Business Object

# Creating a New BC (1)

## Introduction

To create a new BC , New Object Wizard is used

- ✓ Select Business Component

Specify S\_CONTACT\_XM as the base table for the BC

Create fields mapping to columns:

- ✓ NAME , PAR\_ROW\_ID , TYPE

To store other details ATTRIB\* columns can also be mapped

The screenshot shows the Siebel Object Explorer and the Single Value Field List windows. In the Object Explorer, under 'Business Component', a new entry for 'Restaurant' is being created. The 'Single Value Fields' table in the main area lists three columns: 'Contact Id' (PAR\_ROW\_ID, DTYPE\_ID, 15), 'Contact Type' (TYPE, DTYPE\_TEXT, 30), and 'Restaurant Name' (NAME, DTYPE\_TEXT, 100). A purple callout bubble points to this table with the text: 'Upon creation of BC using wizard verify the details of the Columns'.

Name	Column	Type	Text Length
Contact Id	PAR_ROW_ID	DTYPE_ID	15
Contact Type	TYPE	DTYPE_TEXT	30
Restaurant Name	NAME	DTYPE_TEXT	100

# Creating a New BC (2)

## Configuring BC Properties

An XM table stores multiple data of different types

To fetch the data of specific type, **Search Specification** and **Predefault** properties must be configured

When Restaurant details are entered, its type is always defaulted to “Restaurant” using Predefault property

When this BC is instantiated it shall always retrieve the Contact type = Restaurant

The screenshot shows the Siebel Object Explorer interface with the following details:

- Object Explorer:** Shows a tree structure under "Project: \*\* All Projects \*\*". Nodes include Siebel Objects (Applet, Application), Business Component (Business Comp View Mode, Field, Join, Multi Value Field, Multi Value Link, Single Value Field), Business Object, Business Service, EIM Interface Table, and Entity Relationship Diagram.
- Single Value Field List:** A window titled "Single Value Field List" is open, showing a table of single value fields for a Business Component named "S\_CONTACT\_XM".

Name	Table	Search Specification	Project
Restaurant	S_CONTACT_XM	[Contact Type] = 'Restaurant'	Account
- Single Value Fields:** A second window titled "Single Value Fields" is open, showing a table of fields for the same Business Component.

Name	Join	Column	Predefault Value	Type	Text Length	Calculated	Calculated
Contact Id		PAR_ROW_ID		DTYPE_ID	15		
Contact Type		TYPE	Restaurant	DTYPE_TEXT	30		
Restaurant Name		NAME		DTYPE_TEXT	100		

A blue arrow points from the "Search Specification" row in the top table down to the "Predefault Value" column for the "Contact Type" field in the bottom table.

# Topic List

#No	Module Topics
1	Business Challenge
2	Enterprise Solution
3	Creating a New BC
4	Assign BC to Business Object

# **Assign BC to Business Object (1)**

## **Overview**

To assign the BC to a BO :

- ✓ A link needs to be created
- ✓ Add the BC to BO as child object definition

Create applet and views as required

Assign the view to the screen and administer it



# Assign BC to Business Object (2)

## Link

To create a link:

- ✓ Specify the parent and child BC
- ✓ Relation between them in source and destination field

Name get auto populated

W	Name	Changed	Project	Parent Business Component	Child Business Component	Source Field	Destination Field
>	Contact/Restaurant	✓	Account	Contact	Restaurant	Id	Contact Id
	04-HYQ7K		ATP	Quote	Quote Item (Simpl.)	Line Number (Seq)	Quote Id
	ABO Bulk Request Action Set/ABO Bulk Requ		ABO Bulk Request	ABO Bulk Request	Action Set	ABO Bulk Request Action Set Orders	Action Set Id
	ABO Bulk Request Action Set/ABO Bulk Requ		ABO Bulk Request	ABO Bulk Request	Action Set	ABO Bulk Request Actions	Bulk Request Action Set Id
	ABO Bulk Request Action Set/ABO Bulk Requ		ABO Bulk Request	ABO Bulk Request	Action Set	ABO Bulk Request Actions.Sequence	Bulk Request Action Set Id
	ABO Bulk Request Action Set/ABO Bulk Requ		ABO Bulk Request	ABO Bulk Request	Action Set	ABO Bulk Request Exceptions	Action Set Id
	ABO Bulk Request Action Set/ABO Bulk Requ		ABO Bulk Request	ABO Bulk Request	Action Set	ABO Bulk Request Exceptions.Seque	Action Set Id
	ABO Bulk Request Action Set/ABO Bulk Requ		ABO Bulk Request	ABO Bulk Request	Action Set	ABO Bulk Request Instance	Bulk Request Action Set Id

**S\_CONTACT**

ROW_ID	NAME	LAST_NAME	FST_NAME
PK			

**S\_CONTACT\_XM**

ROW_ID	PAR_ROW_ID	TYPE	NAME	ATTRIB_01
FK				



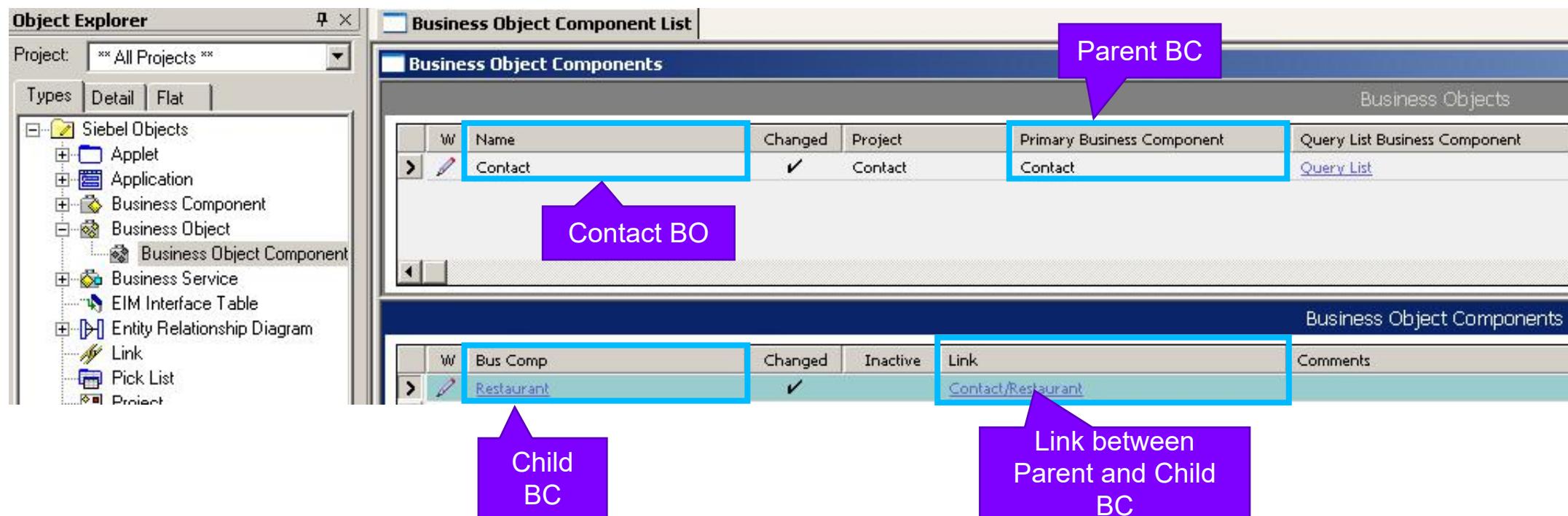
# Assign BC to Business Object (3)

## Add BC to BO

Firstly identify the BO that corresponds to primary BC

Add the Child BC created under the Business Object Components definitions

- ✓ Make sure that we set the link appropriately

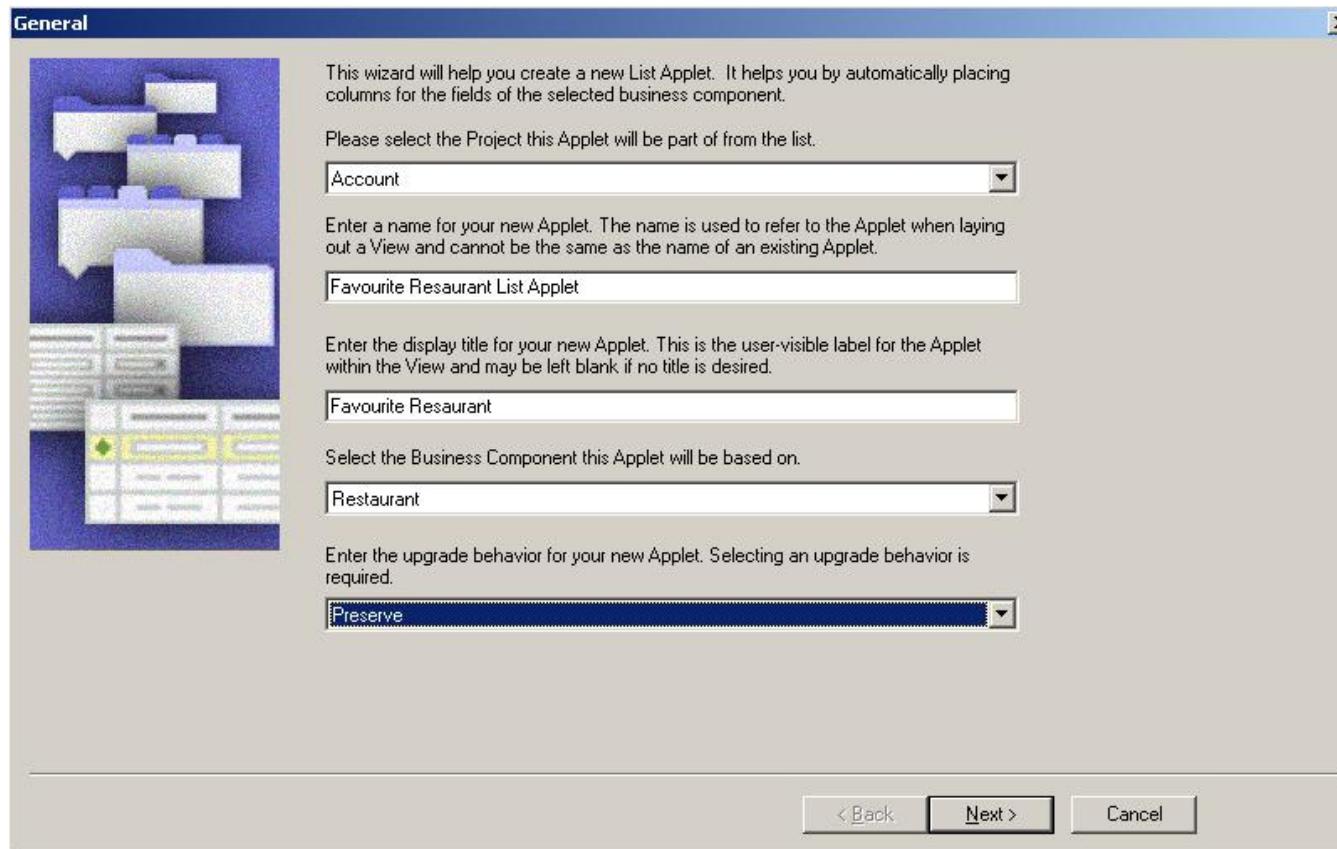


# Assign BC to Business Object (4)

## List Applet

List applet is required is to display the child data , New Object wizard is used to create the same.

- ✓ Type field will not be displayed on the applet as we have defaulted the value in the BC to prevent users from editing the value



# Assign BC to Business Object (5)

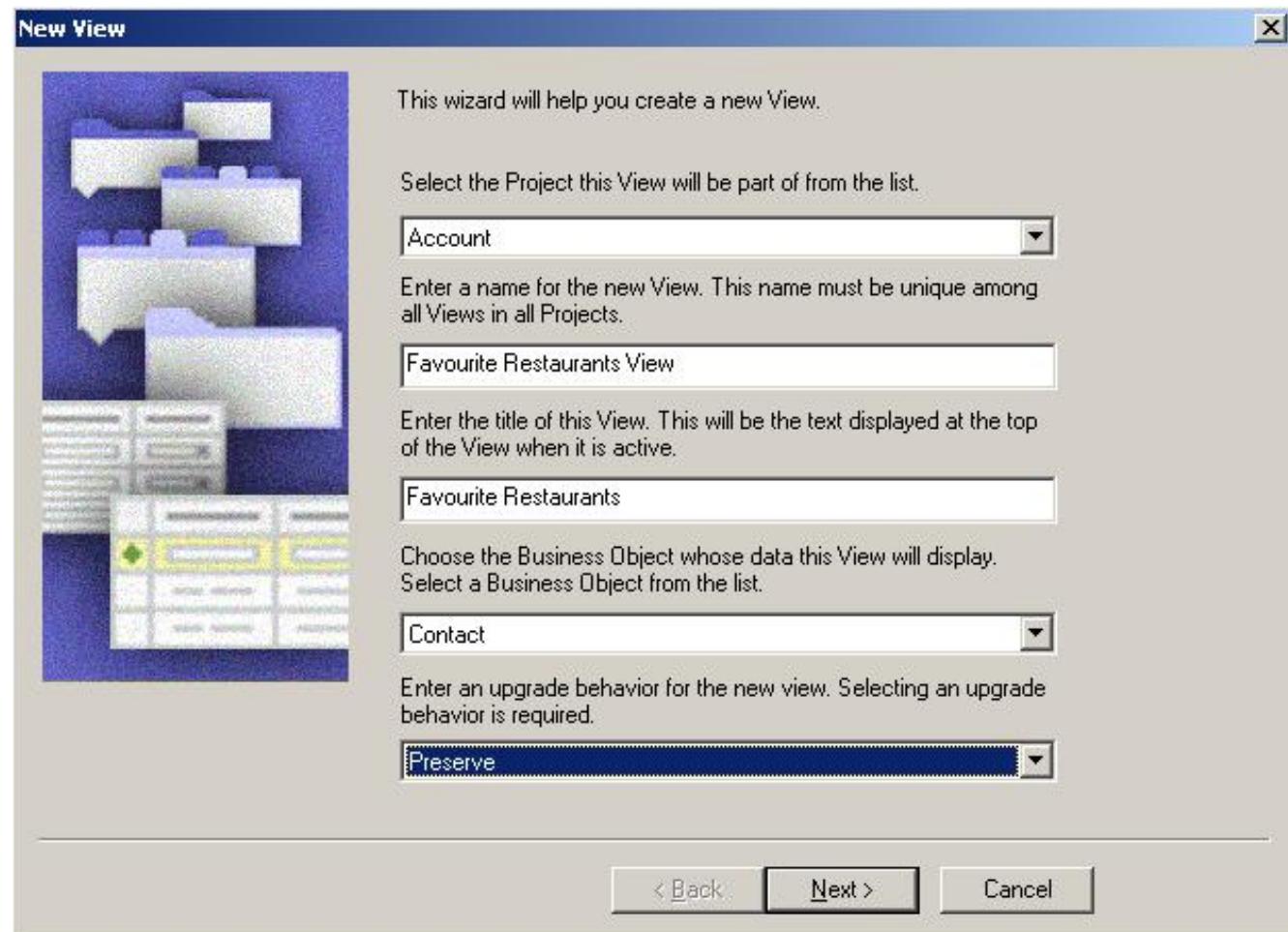
## View

To expose the new list applet on UI , it is required to create a new View

- ✓ Use New Object Wizard to create

Assign the new view to a screen

Administer the view in the client



# Knowledge Checks

**Answer the following:**

1. 1:M extension tables have NAME, TYPE, ROW\_ID as User key columns. State TRUE/ FALSE.
2. An \_XM Table can contain multiple child BC records. State TRUE/ FALSE.
3. Arrange the sequence of steps for assigning a BC to BO.
  - A. Create a List Applet
  - B. Assign a Applet to View
  - C. Create a Link
  - D. Assign BC to BO



# Module Summary

**Now, you should be able to:**

- Define 1:M relationship using child Business Component
- Explain how to:
  - create a new BC using 1:M extension table
  - add the BC to the Business Object



# **Thank You**

# SIEBEL

Extending Siebel Database

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Describe how to create:
  - Extension columns in a table
  - A new table:
    - Standalone Table
    - 1:1 Table
    - 1:M Table
    - Intersection Table
- Apply these changes to the database



# Topic List

#No	Module Topics
1	Extension Column Introduction
2	Standalone Table
3	1:1 Extension Table Creation
4	1:M Extension Table Creation
5	Intersection Table Creation
6	Apply Changes to Database

# Topic List

#No	Module Topics
1	Extension Column Introduction
2	Standalone Table
3	1:1 Extension Table Creation
4	1:M Extension Table Creation
5	Intersection Table Creation
6	Apply Changes to Database

# Extension Column Introduction (1)

## Requirement

- Requirements may arise which asks for new fields to capture more details
  - ✓ Requires a new BC to capture additional business entities
- Consider examining the Siebel data model first
  - ✓ Check whether the required columns and fields exist
  - ✓ If not, use existing 1:M Extension tables for the new BC
- If Data Model does not support the desired new fields, then create new columns in the base table
- Avoid mapping new fields to existing unused columns, they might have been used elsewhere or they can be used in future releases
  - ✓ Avoid mapping fields to 1:1 extension table columns
- Always use Siebel Tools to extend the Database layer
  - ✓ Extend columns to existing tables or create new tables using tools alone
  - ✓ Never create using SQL scripts



# Extension Column Introduction (2)

## How to Create it

Select the table to be extended

Create New Column record, fill up the desired properties with the appropriate values

- ✓ Name automatically prefixed with X\_

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer tree is visible, showing categories like Siebel Objects, Application, Business Component, and Field. In the center, the 'Column List' window is open under the 'Tables' tab. A new column record for 'S\_OPTY' is being created, with the name 'X\_BONUS\_AMT' entered. The 'Type' dropdown menu is open, showing options like Extension, Number, CLOB, Character, Date, Date Time, Long, and Time. The 'Physical Type' dropdown is also open, showing Number selected. Other columns in the table include ACTL\_CLS\_DT, ADJUSTED\_AMT, ALIAS\_NAME, ALTERNATE\_DATES, APPL\_OWNER\_TYPE\_CD, APPR\_ID, and ASGN\_DT.

W	Requir	Changed	Name	Type	Physical Type	Length	Precision	User Name
>		✓	X_BONUS_AMT	Extension	Number	22	10	X_BONUS_AMT Ex
			ACTL_CLS_DT	Data (Public)	CLOB	7		Actual Close Date
			ADJUSTED_AMT	Data (Public)	Character	22	22	ADJUSTED_AMT
			ALIAS_NAME	Data (Public)	Date	100		Alias Name
			ALTERNATE_DATES	Data (Public)	Date Time	250		Alternate Dates
			APPL_OWNER_TYPE_CD	Data (Public)	Long	30		Appl Owner Type
			APPR_ID	Data (Public)	Number	Time		APPR_ID
			ASGN_DT	Data (Public)	UTC Date Time	15		Assignment Date
					Varchar	7		



# Topic List

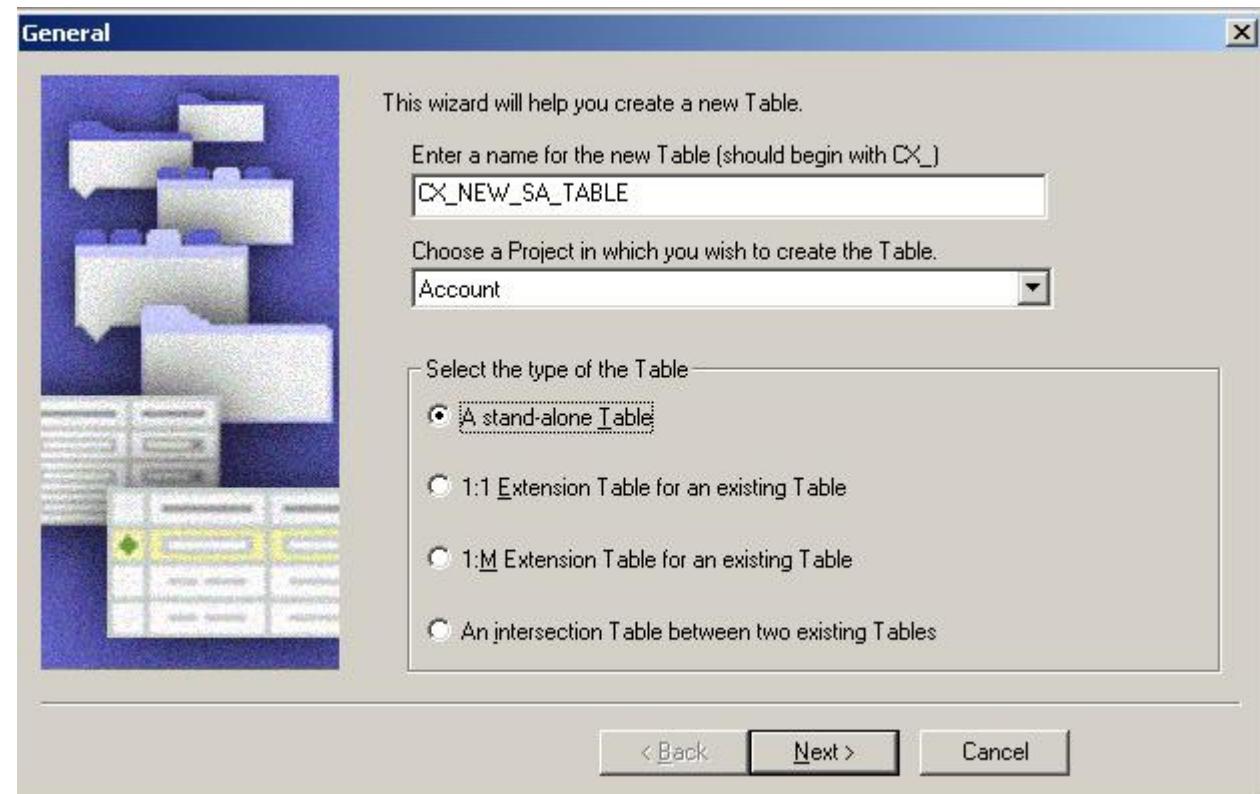
#No	Module Topics
1	Extension Column Introduction
2	Standalone Table
3	1:1 Extension Table Creation
4	1:M Extension Table Creation
5	Intersection Table Creation
6	Apply Changes to Database

# Standalone Table Creation (1)

## Steps

### 1. Select Table from the New Object Wizard

- Provide the details as desired
- Select a stand-alone Table



# Standalone Table Creation (2)

## Steps

2. New Object Wizard creates a stand-alone table with:

- Type: Data (Public)
- Required System columns
- Index P1 on the ROW\_ID

The screenshot shows the SAP GUI Object Explorer and Column List interface. The Object Explorer on the left lists various project components under 'Business Component' and 'Field'. The Column List on the right displays the 'Tables' and 'Columns' sections for the newly created table 'CX\_NEW\_SA\_TABLE'. The 'Tables' section shows the table name 'CX\_NEW\_SA\_TABLE' with its type set to 'Data (Public)'. The 'Columns' section lists several system columns: CONFLICT\_ID, CREATED, CREATED\_BY, DB\_LAST\_UPD, DB\_LAST\_UPD\_SRC, LAST\_UPD, LAST\_UPD\_BY, MODIFICATION\_NUM, and ROW\_ID. The 'ROW\_ID' column is highlighted with a blue border.

W	Name	Type	Changed	Project	User Name	Alias	Object Locked	Object Language Locked	Object Locked
>	CX_NEW_SA_TABLE	Data (Public)	✓	Account	CX_NEW_SA_TABLE				

W	Requir	Changed	Name	Type	Physical Type	Length	Precision	User Name	Foreign Key Table	Primary Key	Alias	Nullable	User Key Sequence
>	✓	✓	CONFLICT_ID	System	Varchar	15		Conflict Id				1	
	✓	✓	CREATED	System	UTC Date Time	7		Created					
	✓	✓	CREATED_BY	System	Varchar	15		Created By	S_USER				
		✓	DB_LAST_UPD	System	UTC Date Time	7		DB Last Updated				✓	
		✓	DB_LAST_UPD_SRC	System	Varchar	50		DB Last Updated By				✓	
	✓	✓	LAST_UPD	System	UTC Date Time	7		Last Updated					
	✓	✓	LAST_UPD_BY	System	Varchar	15		Last Updated By	S_USER				
	✓	✓	MODIFICATION_NUM	System	Number	22	10	Modification Num					
	✓	✓	ROW_ID	System	Varchar	15		Row Id				✓	

# Topic List

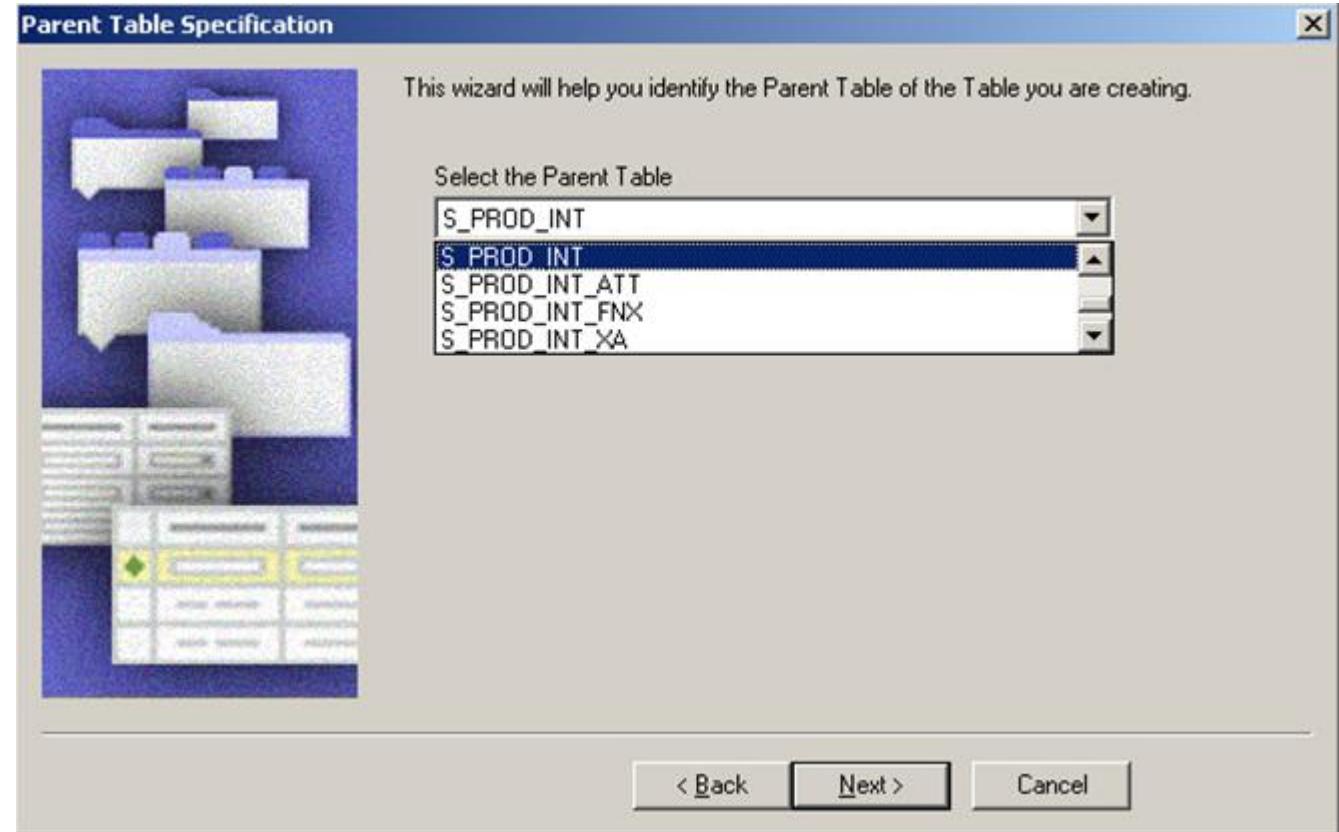
#No	Module Topics
1	Extension Column Introduction
2	Standalone Table
3	1:1 Extension Table Creation
4	1:M Extension Table Creation
5	Intersection Table Creation
6	Apply Changes to Database

# 1:1 Extension Table Creation (1)

## Steps

To create a 1:1 Extension Table:

1. Select a base table
2. Type of the table will be restricted to Data (Public)
3. Multiple extension tables relate to same base table but not to each other



# 1:1 Extension Table Creation (2)

## Steps

4. New Object Wizard creates 1:1 extension table with:

- Type : Extension
- Required System columns
- PAR\_ROW\_ID foreign key column for the parent base table
- 2 indices : Index P1 on the ROW\_ID & U1 on PAR\_ROW\_ID and CONFLICT\_ID

The screenshot shows the Oracle Database Object Explorer interface. The left pane displays a tree view of database objects under a project named "All Projects". The right pane shows two tabs: "Column List" and "Tables". The "Tables" tab is active, showing a table named "CX\_PROD\_INT\_X" with one row. A blue box highlights the "Type" column of this row, which is set to "Extension". The "Columns" tab is also visible, showing the detailed structure of the table. A second blue box highlights the "Name" column of the first row in the "Columns" table, which is "CONFLICT\_ID".

Name	Type	Physical Type	Length	Precision	User Name	Foreign Key Table	Primary Key	Nullable	User Key Sequence
CONFLICT_ID	System	Varchar	15		Conflict Id				2
CREATED	System	UTC Date Time	7		Created				
CREATED_BY	System	Varchar	15		Created By	S_USER			
DB_LAST_UPD	System	UTC Date Time	7		DB Last Updated				
DB_LAST_UPD_SRC	System	Varchar	50		DB Last Updated By				
LAST_UPD	System	UTC Date Time	7		Last Updated				
LAST_UPD_BY	System	Varchar	15		Last Updated By	S_USER			
MODIFICATION_NUM	System	Number	22	10	Modification Number				
PAR_ROW_ID	System	Varchar	15		Par Row Id	S_PROD_INT			1
ROW_ID	System	Varchar	15		Row Id				

# Topic List

#No	Module Topics
1	Extension Column Introduction
2	Standalone Table
3	1:1 Extension Table Creation
4	1:M Extension Table Creation
5	Intersection Table Creation
6	Apply Changes to Database

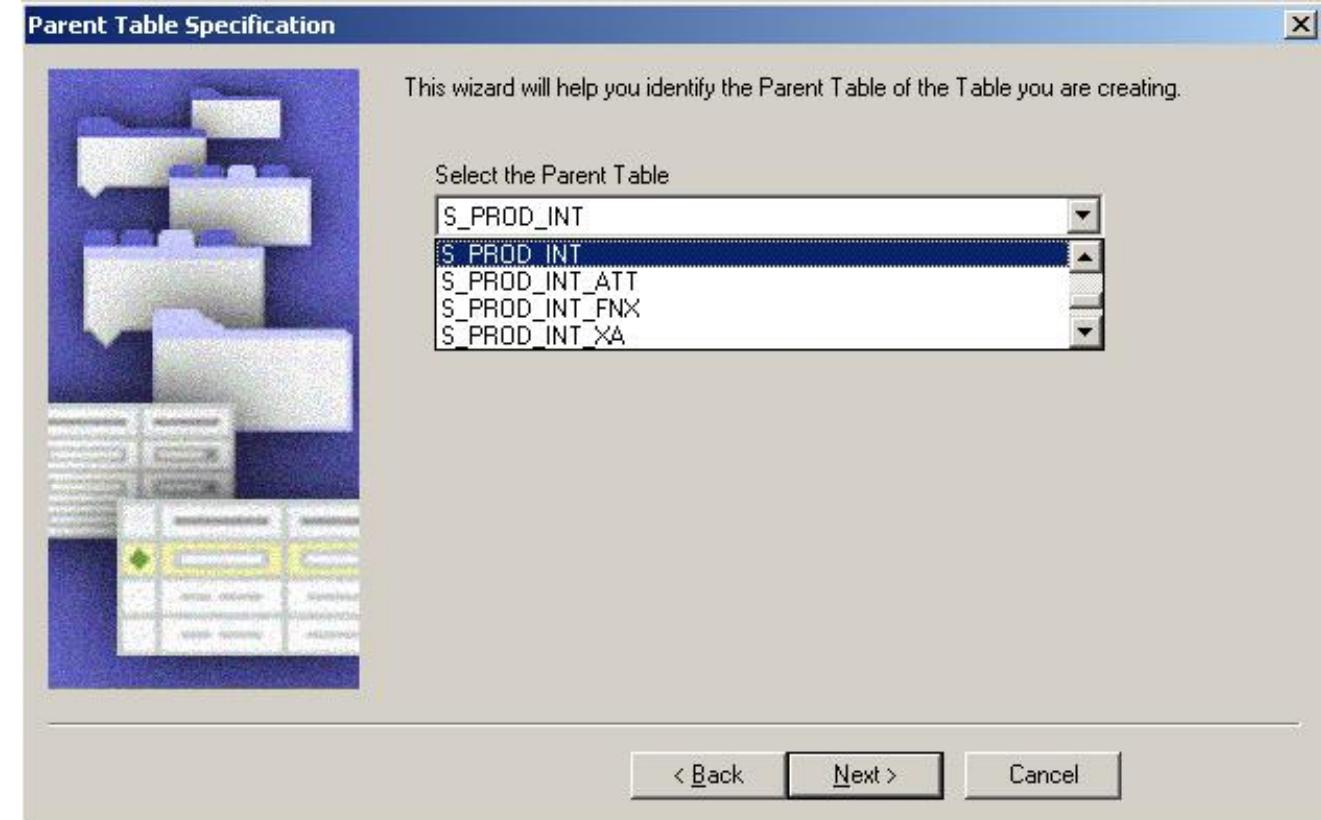
# 1:M Extension Table Creation (1)

## Overview

To create a 1:M Extension Table, Select a base table

- ✓ Should create only if the parent table does not have any other existing 1:M extension table

Type of the table will be restricted to Data (Public)



# 1:M Extension Table Creation (2)

New Object Wizard creates 1:1 extension table with:

- Type : Data(Public)
  - Required System columns
  - PAR\_ROW\_ID Column foreign key to the parent table S\_PROD\_INT
  - TYPE and NAME columns also gets created
- 3 Indices are created :
    - ✓ P1 : on ROW\_ID
    - ✓ U1 : on PAR\_ROW\_ID, TYPE, NAME, CONFLICT\_ID
    - ✓ M1 : on TYPE, NAME

The screenshot shows the Oracle Database Object Explorer interface. On the left, the Object Explorer tree displays various database objects like BusComp Server Script, BusComp View Mode, Business Component User, Field, Join, Multi Value Field, Business Object, Business Service, EIM Interface Table, Entity Relationship Diagram, Link, Pick List, Project, Screen, Table, Column, and Index.

The main area is titled "Column List" and contains two tabs: "Columns" and "Tables". The "Columns" tab is active, showing a list of columns for a table named CX\_PROD\_INT\_XM. The "Type" column for this table is highlighted with a blue box. Below this, a detailed view of the "NAME" column is shown in a "Columns" grid, with its properties like Type (Data (Public)), Physical Type (Varchar), Length (100), and Precision (3) visible.

W	Requir	Changed	Name	Type	Physical Type	Length	Precision	User Name	Foreign Key Table	Primary Key	Nullable	User Key Sequence
>	✓	✓	CONFICT_ID	System	Varchar	15		Conflict Id				4
>	✓	✓	CREATED	System	UTC Date Time	7		Created				
>	✓	✓	CREATED_BY	System	Varchar	15		Created By	S_USER			
>			DB_LAST_UPD	System	UTC Date Time	7		DB Last Updated				✓
>			DB_LAST_UPD_SRC	System	Varchar	50		DB Last Updated By				✓
>	✓	✓	LAST_UPD	System	UTC Date Time	7		Last Updated				
>	✓	✓	LAST_UPD_BY	System	Varchar	15		Last Updated By	S_USER			
>	✓	✓	MODIFICATION_NUM	System	Number	22	10	Modification Number				
>	✓	✓	NAME	Data (Public)	Varchar	100		Name				3
>	✓	✓	PAR_ROW_ID	System	Varchar	15		Par Row Id	S_PROD_INT			1
>	✓	✓	ROW_ID	System	Varchar	15		Row Id				✓
>	✓	✓	TYPE	Data (Public)	Varchar	30		Type				2

# Topic List

#No	Module Topics
1	Extension Column Introduction
2	Standalone Table
3	1:1 Extension Table Creation
4	1:M Extension Table Creation
5	Intersection Table Creation
6	Apply Changes to Database

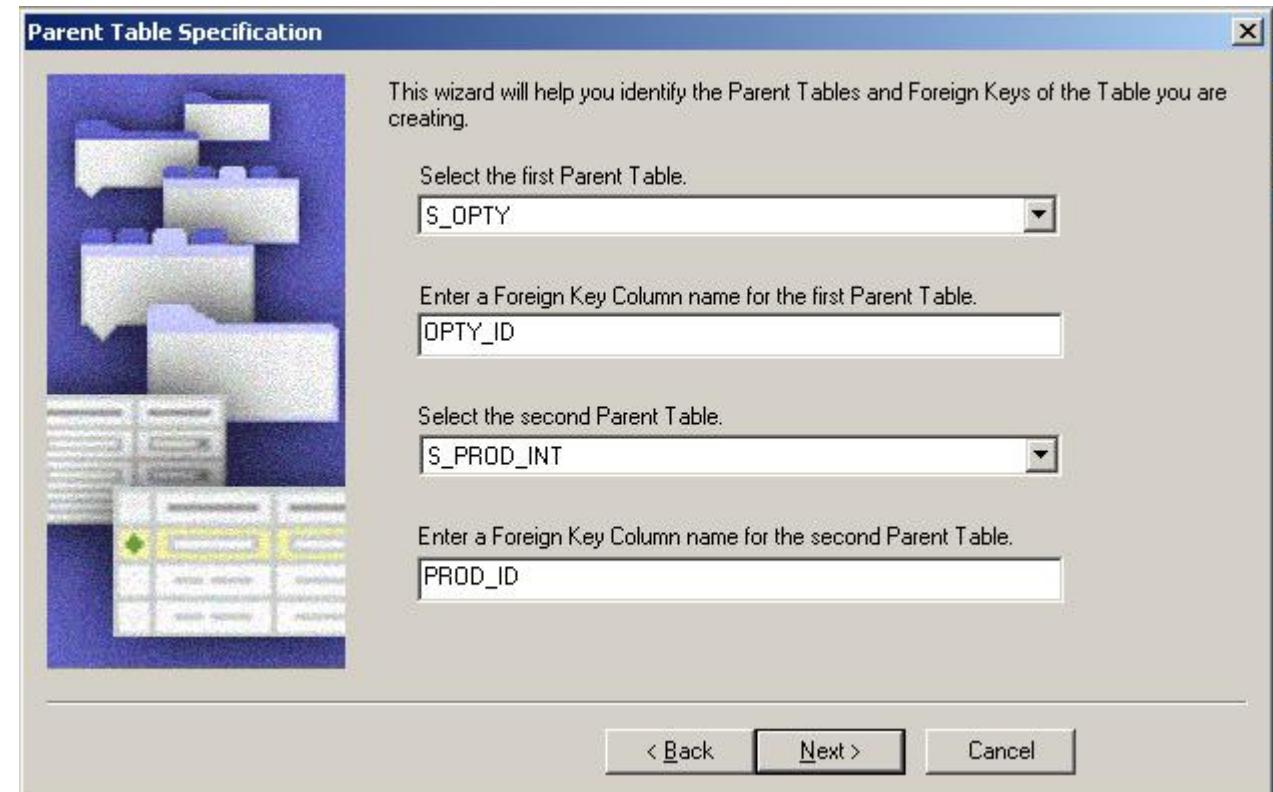
# Intersection Table Creation (1)

## Overview

To create an intersection table select 2 parent tables that will be intersected

Wizard will set the type to Data(Public)

Also the Foreign Key for both the tables also will be created



# Intersection Table Creation (2)

New Object Wizard creates intersection table with:

- Type : Data(Intersection)
- Required System columns
- PROD\_ID Column foreign key to S\_PROD\_INT
- OPTY\_ID Column foreign key to S\_OPTY
- 3 Indices are also created :
  - ✓ P1 – on ROW\_ID
  - ✓ U1 – on PROD\_ID, OPTY\_ID , TYPE, CONFLICT\_ID
  - ✓ F1 – on PROD\_ID (FK to second table)

The screenshot shows the Oracle Database Object Explorer interface. On the left, the Object Explorer tree view is expanded to show various database objects like BusComp Server Script, Field, Join, Multi Value Field, Single Value Field, Business Object, Business Service, EIM Interface Table, Entity Relationship Diagram, Link, Pick List, Project, Screen, Table, Column, Index, and Index Column. The 'Table' node under 'Object Explorer' is selected.

The main area displays the 'Column List' for a table named 'CX\_OPTY\_PROD'. The 'Columns' tab is active, showing the following columns:

W	Name	Type	Changed	Project	User Name	Alias	Object Locked	Object Language
>	CX_OPTY_PROD	Data (Intersection)	✓	Account	CX_OPTY_PROD			
<								

Below this, the 'Columns' tab shows detailed column properties:

W	Requir	Changed	Name	Type	Physical Type	Length	Precision	User Name	Foreign Key Table	Primary Key	Nullable	User Key Sequence
>	✓	✓	CONFLICT_ID	System	Varchar	15		Conflict Id				3
	✓	✓	CREATED	System	UTC Date Time	7		Created				
	✓	✓	CREATED_BY	System	Varchar	15		Created By	S_USER			
	✓	✓	DB_LAST_UPD	System	UTC Date Time	7		DB Last Updated				✓
	✓	✓	DB_LAST_UPD_SRC	System	Varchar	50		DB Last Updated By				✓
	✓	✓	LAST_UPD	System	UTC Date Time	7		Last Updated				
	✓	✓	LAST_UPD_BY	System	Varchar	15		Last Updated By	S_USER			
	✓	✓	MODIFICATION_NUM	System	Number	22	10	Modification Numbe				
	✓	✓	OPTY_ID	Data (Public)	Varchar	15		OPTY_ID	S_OPTY			1
	✓	✓	PROD_ID	Data (Public)	Varchar	15		PROD_ID	S_PROD_INT			2
	✓	✓	ROW_ID	System	Varchar	15		Row Id				✓
	✓	✓	TYPE	Data (Public)	Varchar	30		Type				✓



# Topic List

#No	Module Topics
1	Extension Column Introduction
2	Standalone Table
3	1:1 Extension Table Creation
4	1:M Extension Table Creation
5	Intersection Table Creation
6	Apply Changes to Database

# Apply Changes to Database (1)

## Introduction

Before propagating the changes to the server database :

- ✓ Test changes locally by applying them to local database
- ✓ It reduces the chances of mistakes in the server schema

Best practice is to:

- ✓ Apply changes to the local database and test
- ✓ Migrate the changes to Server Database
- ✓ Propagate these changes to other developers



# Apply Changes to Database (2)

## Apply/DDL

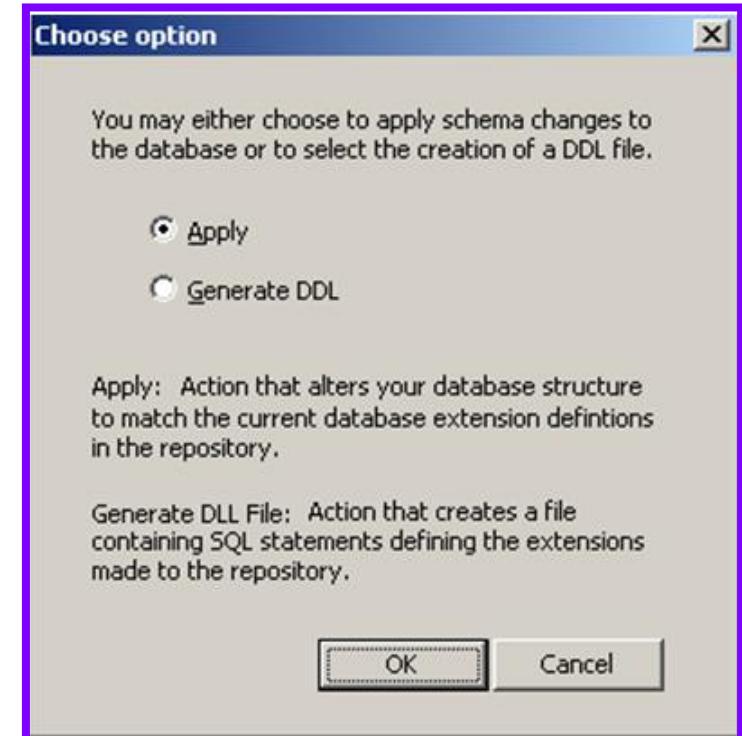
Click apply/DDL button to make changes to database:

- ✓ Choose either apply to database or generate DDL script
- ✓ Changes will be preserved across Siebel upgrades

Compile all the relevant object definitions and projects

Test the changes in local before migrating them to server

- ✓ Query for the table and columns in the DB using a SLQ developer and check



New Table created

Tables

Extend

Apply/DDL

Activate

Click the button

Name	Type	Changed	Project	User Name	Alias	Object Locked	Object La
CX_OPTY_PROD	Data (Intersection)	✓	Account	CX_OPTY_PROD			

# Apply Changes to Database (3)

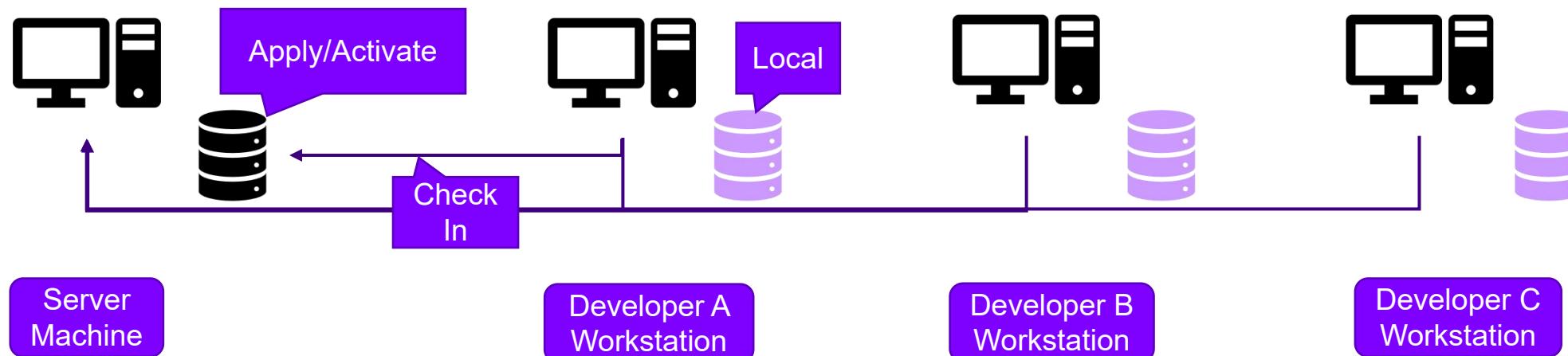
## Migrate Changes to the Server DB

Check in the project to the server

- ✓ Copies the new table and column object definitions to the server

Open server tools

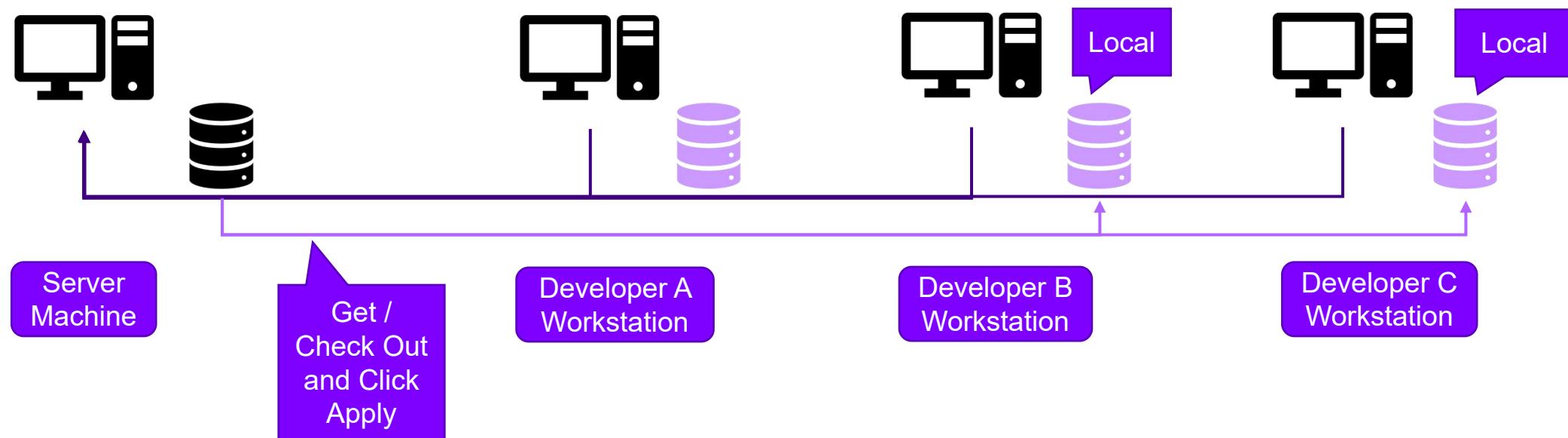
- ✓ Select the table
- ✓ Click apply/DDL to make changes to the physical database
- ✓ Click activate button to update the database schema version
- ✓ Compile all objects and test the changes in the server DB



# Apply Changes to Database (4)

## Propagate Changes to Other Developers

To access the newly deployed changes, other developers can perform a GET (or) CHECK OUT of these objects from the Server.



# Knowledge Checks

## Answer the Following

1. It is advised to map the unused existing columns to fields. State TRUE/FALSE.
2. Which Statement is true regarding creating Intersection table using Wizard
  - A. Type of table is Data(Public)
  - B. PAR\_ROW\_ID is the foreign key to the base table
  - C. 2 Foreign Keys are specified
  - D. Type and Name columns are created
3. When changing the schema, Changes are propagated to other developers before migrating it to server. State TRUE/FALSE.
4. APPLY / DDL and Activate is clicked while committing the changes to the local database schema. State TRUE/FALSE.



# Module Summary

**Now, you should be able to:**

- Describe how to create:
  - Extension columns in a table
  - A new table:
    - Standalone Table
    - 1:1 Table
    - 1:M Table
    - Intersection Table
- Apply these changes to the database



# **Thank You**

# SIEBEL

Drilldowns and Toggle Applets

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Explain how to:
  - Configure a Drilldown
  - Configure a Thread Bar
  - Configure Toggle Applet (Additional Info)



# Topic List

#No	Module Topics
1	Drilldown Configuration
2	Thread Bar Configuration
3	Additional Materials - Toggle Applets

# Topic List

#No	Module Topics
1	Drilldown Configuration
2	Thread Bar Configuration
3	Additional Materials - Toggle Applets

# Drilldown Configuration (1)

## Drilldowns

A drilldown object allows user to drill down on a field in a list applet and takes the user to another view

Hyperlinked fields with coloured texts and underline allows drilldown

Drilldown object are created in Object Explorer > Applet > Drilldown Object

Drilldown is not supported on Pick applets, MVG applets or Association applets

Drilldowns can be either **STATIC** or **DYNAMIC**

- A **static drilldown** always navigates to the same view
- A **dynamic drilldown** navigates to different views, based on certain conditions, such as a field value



# Drilldown Configuration (2)

## Static Drilldown

Static drilldown are of two types :

### Drilldown with in the **same Business Component**

- Drilldown to a detail view for same record
- Destination View uses the same BO and BC

### Drilldown to a **different Business Component**

- Drilldown for a detail view of a related record
- Destination View used different BO and BC



# Drilldown Configuration (3)

## Static Drilldown

Same Business Component

The screenshot displays two views of an Oracle Application Development Framework (ADF) application, illustrating a static drilldown configuration.

**Top View (List View):** This view shows a list of quotes. A purple callout bubble points to the "BO : Quote" label above the list, indicating the business object type. A blue box highlights the first quote entry, "Sample1".

**Bottom View (Detail View):** This view shows the details for the selected quote "Sample1". A purple callout bubble points to the "BO : Quote" label above the detail view, indicating the business object type. The detail view includes tabs for Catalog, Line Items, Pricing, Shipping, Payments, Summary, Approvals, and Orders. The Catalog tab is active, showing a catalog dropdown set to "New Catalog" and a list item "test1 (1)".

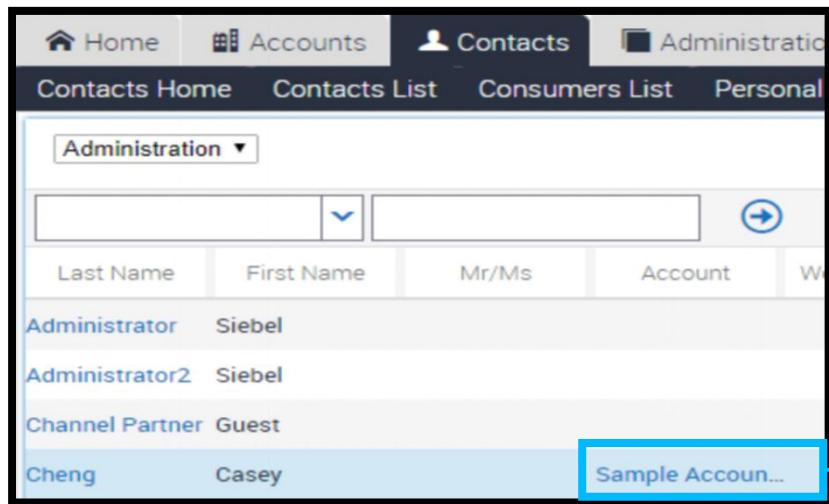
A blue arrow points from the "Sample1" row in the top list view to the "Catalog" tab in the bottom detail view, visually representing the drilldown relationship between the two components.



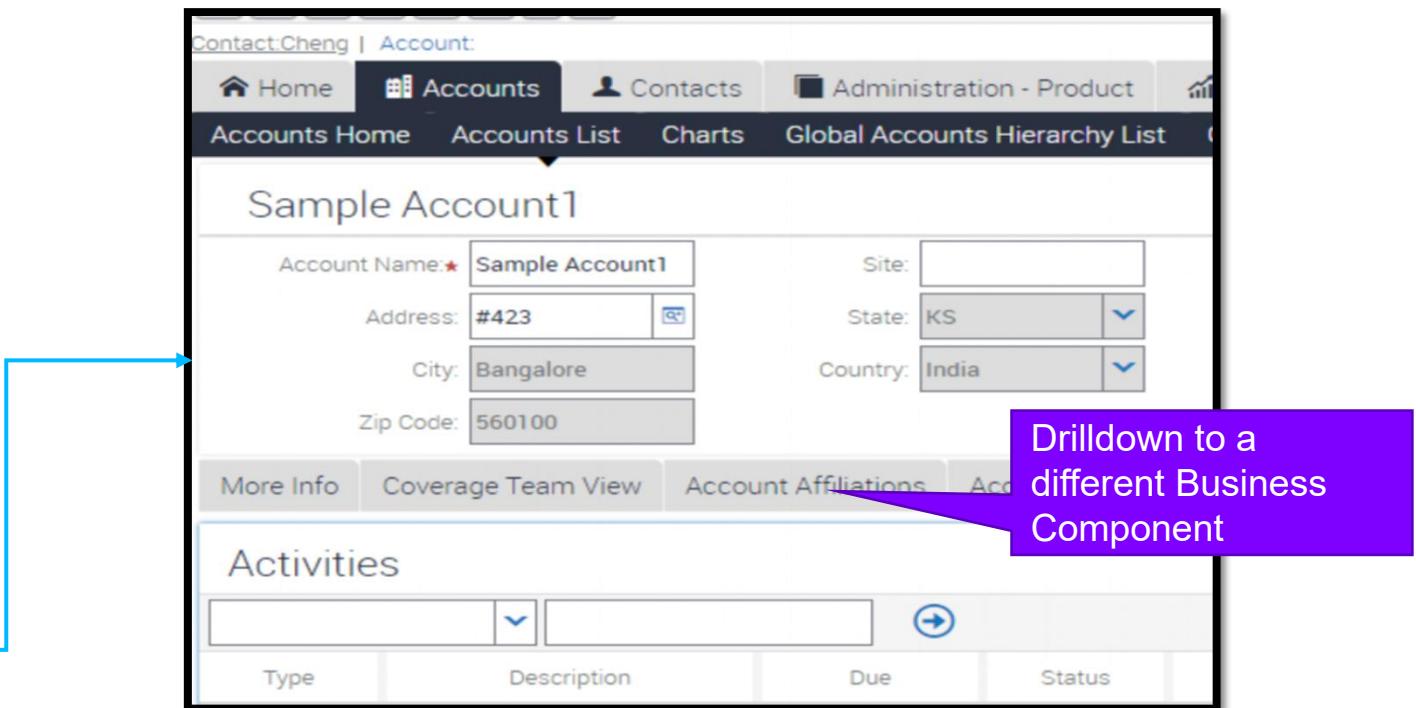
# Drilldown Configuration (4)

## Static Drilldown

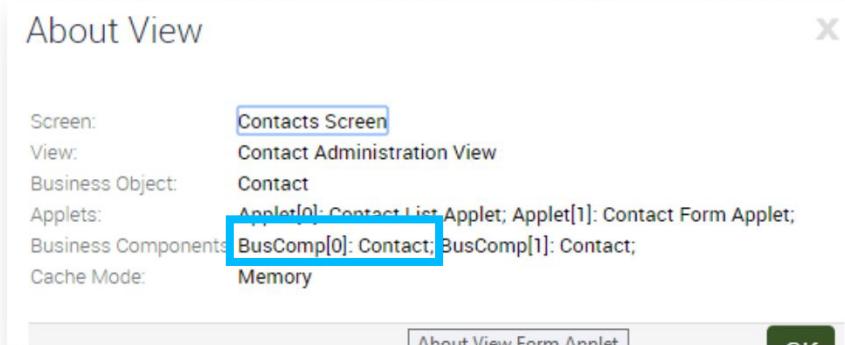
Different Business Component



Screenshot of the Siebel Contacts component. The top navigation bar includes Home, Accounts, Contacts, and Administration. Below the navigation is a search bar and filter options for Last Name, First Name, Mr/Ms, Account, and Work. The main list displays several contact entries, including Administrator, Administrator2, Channel Partner, and Cheng. A blue box highlights the "Sample Account..." button in the bottom right corner.

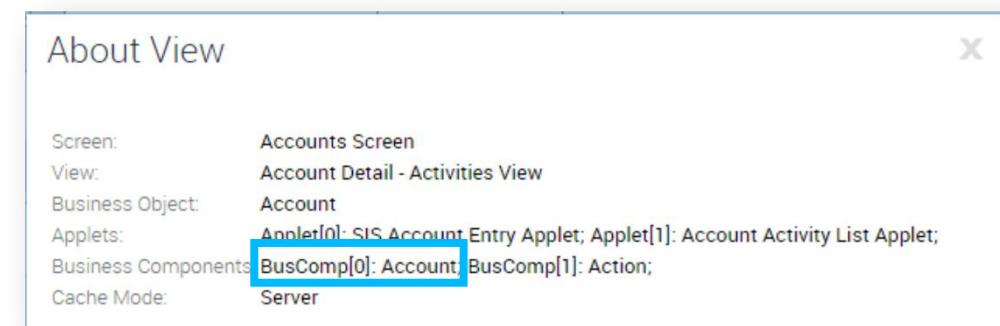


Screenshot of the Siebel Accounts component. The top navigation bar includes Home, Accounts, Contacts, and Administration - Product. Below the navigation is a search bar and filter options for Account Name, Address, City, Zip Code, Site, State, and Country. The main view shows a "Sample Account1" with details: Account Name: Sample Account1, Address: #423, City: Bangalore, Zip Code: 560100, Site: (empty), State: KS, and Country: India. Below the details is a section titled "Activities" with columns for Type, Description, Due, and Status. A purple callout points to the "Activities" section with the text "Drilldown to a different Business Component".



About View

Screen:	Contacts Screen
View:	Contact Administration View
Business Object:	Contact
Applets:	Applet[0]: Contact List Applet; Applet[1]: Contact Form Applet;
Business Components:	BusComp[0]: Contact; BusComp[1]: Contact;
Cache Mode:	Memory



About View

Screen:	Accounts Screen
View:	Account Detail - Activities View
Business Object:	Account
Applets:	Applet[0]: SIS Account Entry Applet; Applet[1]: Account Activity List Applet;
Business Components:	BusComp[0]: Account; BusComp[1]: Action;
Cache Mode:	Server



# Drilldown Configuration (5)

Object Explorer

Project: \*\* All Projects \*\*

Types Detail Flat

- Siebel Objects
  - Applet
    - Applet Method Menu I
    - Applet Web Template
    - Control
    - Drilldown Object
    - List
    - Tree
  - Application
  - Business Component
  - Business Object
  - Business Service
  - EIM Interface Table
  - Entity Relationship Diagram
  - Link
  - Pick List
  - Project
  - Screen
  - Table
  - Task
  - Task Group
    - Task Group Item
    - Task Group Locale
  - View

Drilldown Objects | Task - ACN Create New Contact- 0

Drilldown Objects

Applets

W	Name	Changed	Project	Business Component	Class	Title	Title - String Reference
>	Quote List Applet		Quote (UI)	Quote	CSSSWEFrameListSIAQuote	Quotes	SBL_QUOTES-1004225057-4ZU

Drilldown Objects

W	Name	Changed	Hyperlink Field	View	Source Field	Business Component	Destination Field
	Account		Account	Account Detail - Contacts View	Account Id	Account	
	Admin		Name	Pricer Quotes Detail View		Quote	
	Billing Account		Billing Account	SIS OM Customer Account Portal Vir	Bill To Account Id	Account	
	Contact		Contact Last Name	Contact Detail View	Contact Id	Contact	
	Line Item Detail		Quote Number	TAF Quote Line Item Detail View	Id	Quote	
	Opportunity		Opportunity	Opportunity Detail - Contacts View	Opportunity Id	Opportunity	
>	Original		Name	Quote-Browse Catalog View		Quote	
	Service Account		Service Account	SIS OM Customer Account Portal Vir	Service Account Id	Account	
	TNT SHM Functions		Name	TNT SHM Quote Function Line Items			
	TNT SHM Room Blocks		Quote Number	TNT SHM Quote Room Blocks View			
	eDealer Quote Detail		Quote Number	Quote Detail View	Id	Quote	



# Drilldown Configuration (6)

## Static Drilldown to Different BC

The screenshot shows the Siebel Object Explorer interface with the 'Drilldown Objects' tab selected. The left pane displays a tree view of Siebel Objects, with 'Applet' expanded to show various sub-options like 'Applet Browser Script', 'Applet Locale', etc. The main pane shows two tables: 'Applets' and 'Drilldown Objects'. The 'Applets' table has one row: 'Contact List Applet' (Name: Contact (SSE), Project: Contact, Class: CSSFrameListFINGenericButton, Title: Contacts, Title - String Reference: SBL\_CONTACTS-1004224924-40K). The 'Drilldown Objects' table has one row: 'Account' (Hyperlink Field: Account, View: Account Detail - Activities View, Source Field: Account Id, Business Component: Account, Destination Field: [empty], Menu Text: [empty]). A blue box highlights the 'Account' row in the Drilldown Objects table.

Name	Changed	Project	Business Component	Class	Title	Title - String Reference	Title - String Override
Contact List Applet	✓	Contact (SSE)	Contact	CSSFrameListFINGenericButton	Contacts	SBL_CONTACTS-1004224924-40K	

Name	Changed	Hyperlink Field	View	Source Field	Business Component	Destination Field	Menu Text
Account		Account	Account Detail - Activities View	Account Id	Account		

# Drilldown Configuration (7)

## Dynamic Drilldown

The screenshot illustrates a dynamic drilldown configuration in a CRM application, specifically focusing on the Opportunities module.

**Top Navigation:** Home, Accounts, Contacts, Administration - Product, Opportunities (selected), Quotes.

**Sub-navigation:** Opportunity Explorer, Manager's Explorer, Opportunities Home, Opportunities List, Opportunities Chart.

**Main Content:** My Opportunities search bar and grid. The grid shows four opportunities: Term Life Insurance (Life Insurance, Ace Corporation, \$50,000.00), Upsell cards (Fixed Income, Ace Corporation, \$10,000.00), HK Recurring (Harera, Suzie, \$0.00), and HK (Harera, Suzie, \$120.00).

**Drilldown 1: Term Life Insurance**

**Fields:** Name: Term Life Insurance, Last Name: Aamos, Work #: (967) 876-7878, Account: Ace Corporation, First Name: James, Home #: (967) 565-6767, Revenue: \$50,000.00, Sales Team: SADMIN, Lead Partner: [empty], Currency: USD, Sales Stage: Submitted, Description: [empty].

**Buttons:** Assignment Skills, Registrations, More Info, Capture, Activities, Policies / Quotes.

**Drilldown 2: Upsell cards**

**Fields:** Name: Upsell cards, Last Name: Aamos, Work #: (967) 876-7878, Account: Ace Corporation, First Name: James, Home #: (967) 565-6767, Revenue: \$10,000.00, Sales Team: SADMIN, Lead Partner: [empty], Currency: USD, Sales Stage: Data Entry, Description: [empty].

**Buttons:** Assignment Skills, Registrations, More Info, Capture, Activities, Opportunity Detail (selected), Advisory, Commercial Loan, Equity, Fixed Income, Lending Syndicate.

**Bottom Content:** Policy List search bar and grid. The grid columns include Policy #, Type, Status, Substatus, Line Of Business, Last Name, First Name, Middle.

**Right Panel Fields:** Type, Coupon %, Amount, Offering Price, Maturity Date, Yield %, Denomination.



# Configuring a Drilldown (7)

Object Explorer

Project: \*\* All Projects \*\*

Types: Detail | Flat

- Siebel Objects
  - Applet
    - Applet Method Menu
    - Applet Web Template
    - Control
    - Drilldown Object
    - List
    - Tree
  - Application
  - Business Component
  - Business Object
  - Business Service
  - ELM Interface Table
  - Entity Relationship Diagram
  - Link
  - Pick List
  - Project
  - Screen
  - Table
  - Task
  - Task Group
    - Task Group Item
    - Task Group Locale

Drilldown Objects

Applets

W	Name	Changed	Project	Business Component	Class	Title
>	Opportunity List Applet		Opty (SSE)	Opportunity	CSSSWETNTFrameListBase	Opportunities

Drilldown Objects

W	Name	Chan	Hyperlink Field	Sequence	View	Source Field	Business Component	Destination Field
>	Auto Insurance View		Name	12	INS Opportunity Detail - Policy View		Opportunity	Id
	Commercial View		Name	7	FINCORP Deal Detail View		Opportunity	Id
	Function Agenda View		Name	16	TNT SHM Opportunity Agenda View		Opportunity	Id
	Investment Advisory View		Name	8	FINCORP Deal Advisory View		Opportunity	Id
	Investment Debt View		Name	9	FINCORP Deal Debt View		Opportunity	Id
	Investment Equity View		Name	10	FINCORP Deal Equity View		Opportunity	Id
	Life Insurance View		Name	6	INS Opportunity Detail - Policy View		Opportunity	Id
	Line of Business		Name	2	Opportunity Detail - Contacts View		Opportunity	Id
	Original		Name	1	Opportunity Detail - Contacts View		Opportunity	Id
	Property Insurance View		Name	3	INS Opportunity Detail - Policy View		Opportunity	Id



# Drilldown Configuration (8)

Object Explorer    x

Project: \*\* All Projects \*\*

Types Detail Flat

Siebel Objects

- Applet
  - Applet Method M...
  - Applet Web Temp...
  - Control
  - Drilldown Object
    - Dynamic Drilldo...
  - List
    - List Column
  - Tree
- Application
- Business Component
  - BusComp View Mo...
  - Field
  - Join
  - Multi Value Field
  - Multi Value Link
  - Single Value Field
- Business Object
- Business Service
- EIM Interface Table
- Entity Relationship Dia...

Dynamic Drilldown Destinations | Task - ACN Create New Contact- 0 |

Drilldown Objects

	W	Name	Chan	Hyperlink Field	Sequence	View	Source Field	Business Component	Destination Field
		Auto Insurance View		Name	12	<a href="#">INS Opportunity Detail - Policy View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>
		Commercial View		Name	7	<a href="#">FINCORP Deal Detail View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>
		Function Agenda View		Name	16	<a href="#">TNT SHM Opportunity Agenda View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>
		Investment Advisory View		Name	8	<a href="#">FINCORP Deal Advisory View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>
		Investment Debt View		Name	9	<a href="#">FINCORP Deal Debt View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>
		Investment Equity View		Name	10	<a href="#">FINCORP Deal Equity View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>
		Life Insurance View		Name	6	<a href="#">INS Opportunity Detail - Policy View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>
	>	Line of Business		Name	2	<a href="#">Opportunity Detail - Contacts View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>
		Original		Name	1	<a href="#">Opportunity Detail - Contacts View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>
		Property Insurance View		Name	3	<a href="#">INS Opportunity Detail - Policy View</a>		<a href="#">Opportunity</a>	<a href="#">Id</a>

Dynamic Drilldown Destinations

	W	Name	Changed	Field	Value	Destination Drilldown Object	Sequence
	>	Agenda Function View		Hospitality Flag	Y	Function Agenda View	14
		Auto Insurance View		Deal Type	Auto Insurance	Auto Insurance View	6
		Investment Advisory View		Deal Type	Advisory Services	Investment Advisory View	5
		Investment Debt View		Deal Type	Fixed Income	Investment Debt View	4
		Investment Equity View		Deal Type	Equity	Investment Equity View	3
		Life Insurance View		Deal Type	Life Insurance	Life Insurance View	1
		Property Insurance View		Deal Type	Property Insurance	Property Insurance View	2



# Topic List

#No	Module Topics
1	Drilldown Configuration
2	Thread Bar Configuration
3	Additional Materials - Toggle Applets

# Thread Bar Configuration (1)

## Thread Bar

It is used to track the previous active record

It provides a hyperlink to the former view and is updated when the user navigates to a different BO

The screenshot shows a CRM application interface. At the top, there is a navigation bar with various links: Home, Accounts, Contacts, Administration - Product, Opportunities, Quotes, Fleet Management, and Administration. Below this is a secondary navigation bar with links: Accounts Home, Accounts List, Charts, Global Accounts Hierarchy List, Global Accounts Administration, Account Management, and Accounts Administ. A purple callout box labeled "Thread Bar displaying the previous active record drilldown" points to the "Contact:Cheng | Account:" link in the top navigation bar, which is highlighted with a blue border. The main content area displays a form for "Sample Account1" with fields for Account Name (Sample Account1), Site, Account Team (SADMIN), Address (#423), State (KS), Main Phone #, City (Bangalore), Country (India), Main Fax #, Zip Code (560100), and URL. Below the form is a tabbed section with tabs: More Info, Coverage Team View, Account Affiliations, Account Catalogs, Account Hierarchy, Account Opportunities, Account Profile, and Activities. The "Activities" tab is selected. At the bottom, there is a table titled "Activities" with columns: Type, Description, Due, Status, Priority, and Opportunity.

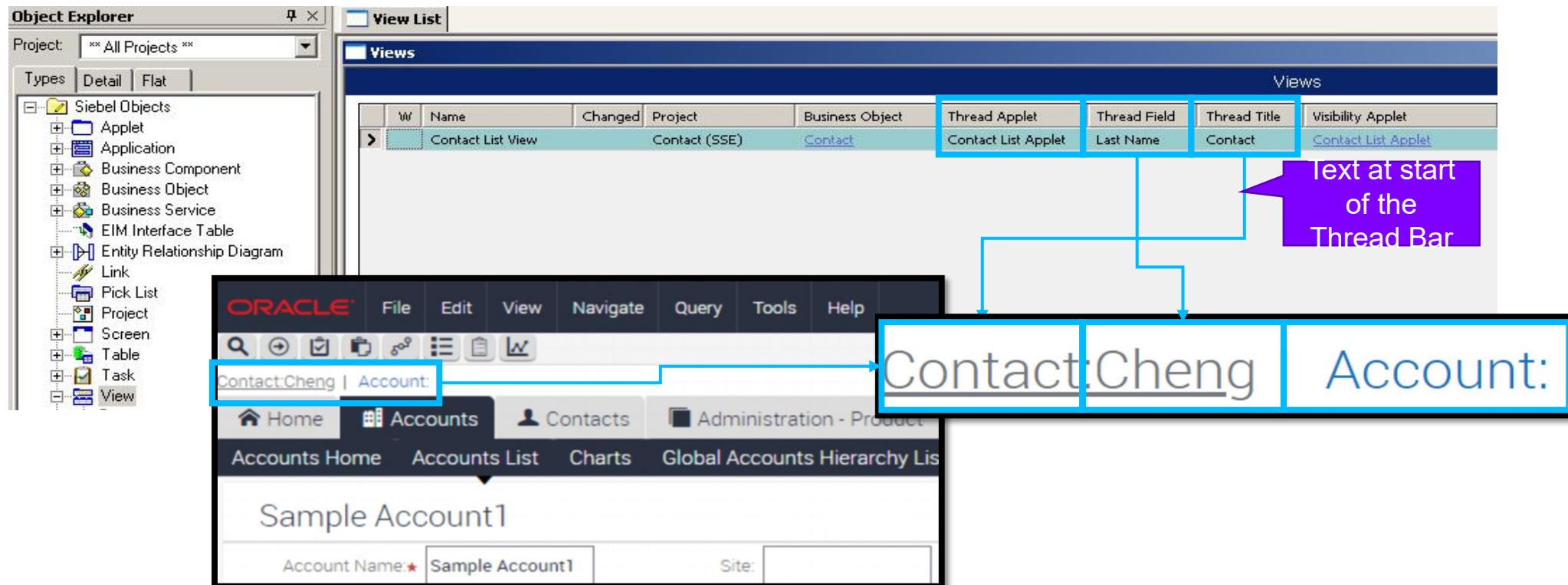


# Thread Bar Configuration (2)

## How to?

To configure a thread bar:

Thread bar properties needs to be set at View object definition



# Topic List

#No	Module Topics
1	Drilldown Configuration
2	Thread Bar Configuration
3	Additional Materials - Toggle Applets

# Additional Materials—Toggle Applets (1)

- An applet toggle allows the user to switch between different applets in the same view
- This object type occurs in : Object Explorer > Applet > Applet Toggle
- Same placeholder could be shared by several applets in the view
  - Configuring Toggle Applet involves adding only one applet in the View Web Template
  - Applets can be switched by configuring Applet Toggle object definitions
- Applet web template needs to contains SWE *Toolbar* tag to enable Applet Toggle
- Applet Toggle can be either **STATIC** or **DYNAMIC**
  - In **Static Applet Toggle** user manually chooses from a dropdown list and selects the applet
  - In **Dynamic Applet Toggle** the applet dynamically changes based on a field value or certain conditions



# Additional Materials—Toggle Applets (2)

## Static Applet Toggle

The screenshot shows the Oracle Sales Cloud interface for a quote named "Sample1". A purple callout box labeled "TNT SHM Quote Notes View" points to the "Notes" tab in the top navigation bar. Another purple callout box labeled "Applet Toggle" points to a dropdown menu in the "Public Notes" section, which contains options: "Public Notes", "Public Notes", and "Private Notes". The "Public Notes" option is highlighted with a blue border.

**TNT SHM Quote Notes View**

**Applet Toggle**

Quote Name: Sample1  
Account: [ ]  
Opportunity: [ ]  
Quote Total: \$0.00  
Price List: [ ]  
Currency: USD  
Status: In Progress  
Last Name: Administrator  
Active:   
First Name: Siebel  
Effective Through: 2/26/2018  
Discount: [ ]

More Info Catalog Line Items Pricing Shipping Payments Summary Approvals Orders Documents Functions Room Block Info Room Blocks Summary Time Shift Shipments Product Search Personal Bill To/Ship To Evaluations Notes

Public Notes

Description	Created By	Created
Public Notes	SADMIN	2/28/2018 9:20:11 PM

Private Notes

Description	Created By	Created
Private Notes	SADMIN	2/28/2018 9:20:33 PM



# Additional Materials—Toggle Applets (3)

## Configuring Applet Toggle – Static Toggle

- Create Applet Toggle definition for each applet in the view to be added in the toggle list
- Create these definitions under the Applet from the View web template items

The screenshot shows the Siebel Object Explorer interface with several windows open:

- Object Explorer:** Shows the navigation tree with "Siebel Objects" expanded, and "Applet" selected.
- Applet Toggle List:** Shows the "Applet Toggles" section with two entries:
  - Quote Entry Applet (Item Identifier 1)
  - TNT SHM Quote Note List Applet (Item Identifier 2, highlighted in blue)
- View Web Template Items:** Shows the "Base" item in the list.
- Applet Toggles:** Shows the configuration for the TNT SHM Quote Note List Applet, with fields: Applet (TNT SHM Quote Note List Applet), Project (TNT SHM Quote), Business Component (TNT SHM Quote Note), Class (CSSFrameList), and Title (Public Notes).
- View Web Template Items:** Shows the configuration for the TNT SHM Quote Private Note List Applet, with fields: Applet (TNT SHM Quote Private Note List Applet), Auto Toggle Field, Auto Toggle Value, Sequence, Inactive, and Comments.

# Additional Materials—Toggle Applets (4)

## Dynamic Applet Toggle

The screenshot shows a Siebel application interface with a navigation bar at the top. Below the navigation bar is a toolbar with three buttons: 'Inbox Items List', 'Completed Items List', and 'Submitted Items List'. The main area is titled 'Inbox Items' and displays a list of items. The first three items are tasks, and the fourth item is a budget request. A purple callout box with the text 'Based on the type of the inbox item the detail view applet toggles dynamically' is overlaid on the interface, pointing to the fourth item. At the bottom of the screen, there is a footer with three buttons: 'More Info', 'Detail' (which is highlighted with a blue border), and 'History'.

Completed	Category	Name	Type	From	Action	Priority	Received	Last Updated	Context	Cor
		Create New Contact	Task	Siebel Administ...			3/12/2018 9:31...	3/12/2018 9:31...		
		Create New Contact	Task	Siebel Administ...			3/12/2018 9:22...	3/12/2018 9:22...		
		Create New Contact	Task	Siebel Administ...			3/12/2018 9:15...	3/12/2018 9:15...		
BUDGET REQU...	In-Store Discounts	Budget Request	Marion May	Not Yet Review...	High		11/26/2006 8:0...	11/26/2006 8:0...		
		176914-5338931	SRManager	Siebel Administ...			6/6/2005 11:25...	6/6/2005 11:25...		
		176914-5338771	SRManager	Siebel Administ...			6/6/2005 10:53...	6/6/2005 10:53...		
		176914-5286999	SRManager	Siebel Administ...			5/25/2005 7:35...	5/25/2005 7:35...		

The screenshot shows a Siebel application interface with a navigation bar at the top. Below the navigation bar is a toolbar with several buttons: Home, Accounts, Contacts, Administration - Product, Opportunities, Quotes, Service, and Inbox. The main area is titled 'Inbox Item' and displays a list of items. The fourth item in the list is highlighted with a blue border. Below the list, there are tabs for 'More Info', 'Detail' (which is highlighted with a blue border), and 'History'. At the bottom of the screen, there is a footer with four buttons: 'Smart Answer', 'Auto Quote', 'Verify', and 'Verify Best Time & Create Case'. The 'Verify Best Time & Create Case' button is highlighted with a green background.

Completed	Category	Name	Type	From	Action	Priority	Received	Last Updated
		Create New Contact	Task	Siebel Administ...			3/12/2018 9:31...	3/12/2018 9:31...
		Create New Contact	Task	Siebel Administ...			3/12/2018 9:22...	3/12/2018 9:22...
		Create New Contact	Task	Siebel Administ...			3/12/2018 9:15...	3/12/2018 9:15...
BUDGET REQU...	In-Store Discounts	Budget Request	Marion May	Not Yet Review...	High		11/26/2006 8:0...	11/26/2006 8:0...
		176914-5338931	SRManager	Siebel Administ...			6/6/2005 11:25...	6/6/2005 11:25...
		176914-5338771	SRManager	Siebel Administ...			6/6/2005 10:53...	6/6/2005 10:53...
		176914-5286999	SRManager	Siebel Administ...			5/25/2005 7:35...	5/25/2005 7:35...



# Additional Materials—Toggle Applets (5)

## Configuring Applet Toggle – Dynamic Toggle

Properties of Applet Toggle

- **Applet** : Name of the applet to be toggled
- **Auto Toggle Field** : Specifies the business component field for dynamic toggle. Siebel examines the current value of this field to the value that the Auto Toggle Value property contains
- **Auto Toggle Value** : Specifies the business component field value for dynamic toggle
- **Name** : Name of the applet to be toggled
- **Sequence** : Specifies the order of display for dynamic toggle



# Additional Materials—Toggle Applets (6)

## Configuring Applet Toggle – Dynamic Toggle

The screenshot shows the Siebel Object Explorer interface with the 'Applet' node selected in the tree view. Two windows are open: 'Applet Toggle List' and 'Applet Toggles'.

**Applet Toggle List:** This window lists applets categorized by project. One applet, 'UInbox Item Task List Applet', is highlighted.

W	Name	Changed	Project	Business C
>	UInbox Item Task List Applet		Universal Inbox	UInbox It

**Applet Toggles:** This window lists applets with their corresponding auto-toggle fields and values.

W	Applet	Changed	Auto Toggle Field	Auto Toggle Value	Sequence	Inactive	Comments
>	Activity Form Applet		Item Type BusObj Name	Action			
	Campaign Description Parent Form /		Item Type BusObj Name	Campaign			
	Marketing Budget Request Form Ap		Item Type BusObj Name	Marketing Budget Request			
	Marketing Plan Form Applet		Item Type BusObj Name	Marketing Plans			
	Offer Detail Form Applet		Item Type BusObj Name	Offer			
	Parent Offer Detail Form Applet		Item Type BusObj Name	Parent Offer			
	Program Detail Form Applet (DBM) -		Item Type BusObj Name	Program (DBM)			
	Service Request Detail Applet		Item Type BusObj Name	Service Request			
	Task Instance Detail Applet		Item Type Name	Task			
	UInbox Task Form Player Applet (Pl		Item Type BusObj Name	Smart Script Player			
	eEvents Parent Event Form Applet		Item Type BusObj Name	Parent eEvents			

This screenshot shows the 'View Web Template Items' module with the 'UInbox Item Task List Applet' selected.

**Top Bar:** View Web Template Items, View Web Templates

**Table Headers:** W, Name, Changed, User Layout, Web Template, Upgrade Behavior, ICL Upgrade Path

**Data Rows:**

>	Base					
				View Detail 2 (Parent with Pointer)	Admin	

**Bottom Bar:** View Web Template Items

**Table Headers:** W, Name, Changed, Item Identifier, Applet, Applet Mode

**Data Rows:**

>	UInbox Item Task List Applet	1		UInbox Item Task List Applet	Edit List
	UInbox Item Task Toggle List Applet	3		UInbox Item Task Toggle List Applet	Edit

# Knowledge Checks

## Answer the following

1. Static Drilldown enables to drilldown to different view. TRUE / FALSE ?
2. Thread bar is updated whenever the user navigates to views of different Business Object. TRUE / FALSE
3. Dynamic Applet Toggle automatically toggles based on the value in the field on that business component. TRUE / FALSE



# Module Summary

**Now, you should be able to:**

- Explain how to:
  - Configure a Drilldown
  - Configure a Thread Bar
  - Configure Toggle Applet (Additional Info)



# **Thank You**

# SIEBEL

Picklist

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Define picklists
- Identify the different types of picklist
- Explain how to:
  - Configure picklists
  - Administer the static picklist



# Topic List

#No	Module Topics
1	Picklists Overview
2	Picklist Configuration
3	Additional Materials - Constrain Picklist
4	Additional Materials - Hierarchical Picklist

# Topic List

#No	Module Topics
1	Picklists Overview
2	Picklist Configuration
3	Additional Materials - Constrain Picklist
4	Additional Materials - Hierarchical Picklist

# Picklists Overview (1)

## Introduction

Picklist updates fields when a value is selected from the list

Users are forced to pick values from the associated list. This:

- ✓ Ensures business rules and policies are met
- ✓ Makes data entry faster
- ✓ Reduces the errors

Picklist are of two types : Static and Dynamic



# Picklists Overview (2)

## Static Picklist

A static picklist allows user to select value for a column from the predefined list of values.

- ✓ When a value is chosen from the list, the value is saved on the field

A static picklist derives values from a table that a Siebel administrator maintains

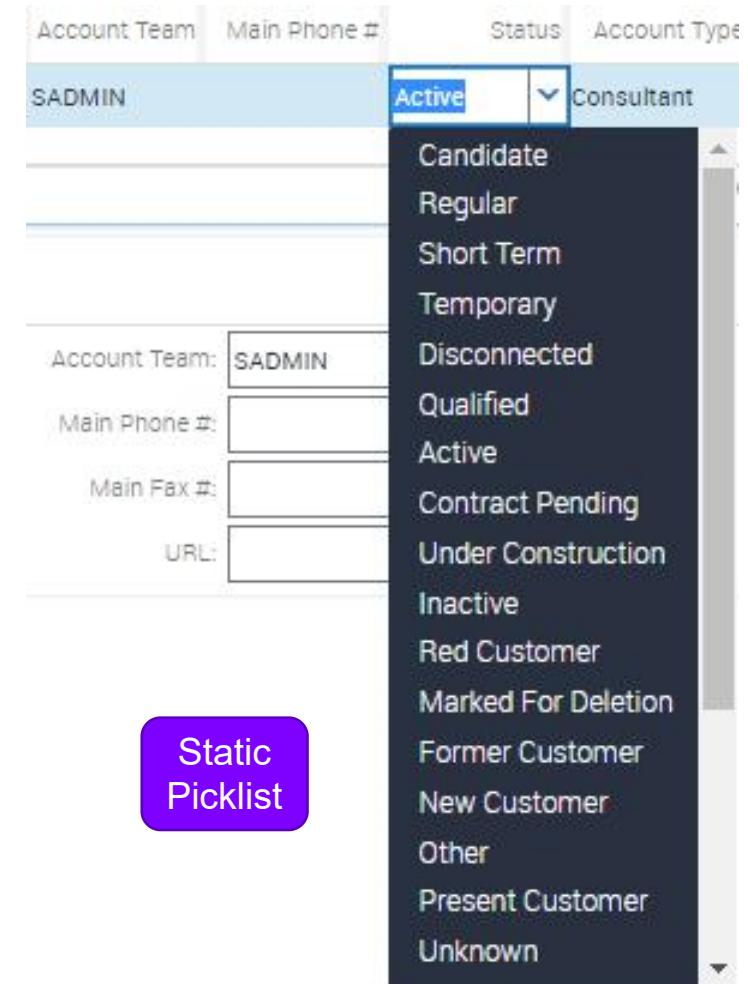
- ✓ These values do not change during runtime or upon user action

Picklist can be:

- ✓ Bounded: User can only choose from the provided set of values. User cannot enter his own values

- ✓ Unbounded: User can enter any values he desires

- This value is not entered into the picklist values



# Picklists Overview (3)

## Static Picklist Values

Static picklist values are stored in a table, S\_LST\_OF\_VAL

Type field indicates the type of static picklist it belongs to

The screenshot shows two instances of the Oracle Database Application Express interface, each displaying a table of static picklist values.

**Top Table (List of Values):**

Type	Display Value	Language-Independent Code	Language Name	Parent LOID	Order	Active	Translate	Multilingual	Replication Level
CONTACT_TYPE	Central Lab Technician	Central Lab Technician	English-American		1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	
CONTACT_TYPE	Company	Company	English-American		1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	
CONTACT_TYPE	HMO Executive	HMO Executive	English-American		2	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	
CONTACT_TYPE	HMO Pharmacy Director	HMO Pharmacy Director	English-American		3	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	

**Bottom Table (List of Values):**

Type	Display Value	Language-Independent Code	Language Name	Parent LOID	Order	Active	Translate	Multilingual	Replication Level
LOV_TYPE	CONTACT_TYPE	CONTACT_TYPE	English-American		1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	



# Picklists Overview (4)

## Administer Static Picklist

Navigate to Administration – Data → List of Values Explorer

Select an existing picklist → Expand the type and select the child values folder

Edit or Add the Picklist Values here

The screenshot shows the Oracle List of Values Explorer interface. On the left, a tree view displays a hierarchy under 'CONTACT\_TYPE'. A blue box highlights the 'Values' node, which contains a list of contact types: Central Lab Technician (English-American), Company (English-American), HMO Executive (English-American), HMO Pharmacy Director (English-American), Household (English-American), Hospital Administrator (English-American), Hospital Executive (English-American), Lab Technician (English-American), Nurse Practitioner (English-American), Other (English-American), Pharmacist (English-American), Physician (English-American), Physician Investigator (English-American), Investigator (English-American), and Site Coordinator (English-American). A purple arrow points from this list to a callout box containing the text: "Uncheck the flag to make it inactive". On the right, a table titled 'List of Values' shows the details for each contact type, including Code, Display Value, Language Name, Active status (indicated by a checked checkbox), Translate, Multilingual, and Organization.

Code	Display Value	Language Name	Active	Translate	Multilingual	Organization
Central Lab Tec...	Central Lab Tec...	English-American	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
Company	Company	English-American	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
HMO Executive	HMO Executive	English-American	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
HMO Pharmac...	HMO Pharmac...	English-American	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
Household	Household	English-American	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
Hospital Admin...	Hospital Admin...	English-American	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
Hospital Execu...	Hospital Execu...	English-American	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
Lab Technician	Lab Technician	English-American	<input checked="" type="checkbox"/>			
Nurse Practitio...	Nurse Practitio...	English-American	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		
Other	Other	English-American	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		

# Picklists Overview (5)

## Administer Static Picklist Using Tools

LOVs can be administered using Siebel Tools as well.

Select Screens → System Administration → List of Values

Query for the Picklist Type

Edit or Add the Picklist Values in the object editor and click Clear Cache

Create a picklist by:

- ✓ Creating a new picklist type
- ✓ Creating Values for the LOV Type



The screenshot shows the Siebel Tools interface with the title bar "Siebel Tools - Siebel Repository - Picklist List". The "Screens" menu is open, showing "Application Upgrader" and "System Administration". The "System Administration" option is also open, showing "System Preferences", "Analytics Strings", and "List of Values", which is highlighted with a blue selection bar. Below the menu is a toolbar with various icons. The main window has tabs at the top: "List of Values Administration" (selected) and "List of Values". The "List of Values" tab is active, showing a table with the following data:

Type	Display Value	Changed	Translate	Multilingual	Language-Independent Code	Order	Active	Parent LIC	High	Low	Language Name
CONTACT_TYPE	Central Lab Technician		✓		Central Lab Technician	1	✓				English-American
CONTACT_TYPE	Company		✓		Company	1	✓				English-American
CONTACT_TYPE	HMO Executive		✓		HMO Executive	2	✓				English-American
CONTACT_TYPE	HMO Pharmacy Director		✓		HMO Pharmacy Director	3	✓				English-American
CONTACT_TYPE	Household		✓		Household	3	✓				English-American
CONTACT_TYPE	Hospital Administrator		✓		Hospital Administrator	4	✓				English-American
CONTACT_TYPE	Hospital Executive		✓		Hospital Executive	5	✓				English-American
CONTACT_TYPE	Lab Technician		✓		Lab Technician	6	✓				English-American



# Picklists Overview (6)

## Dynamic Picklist

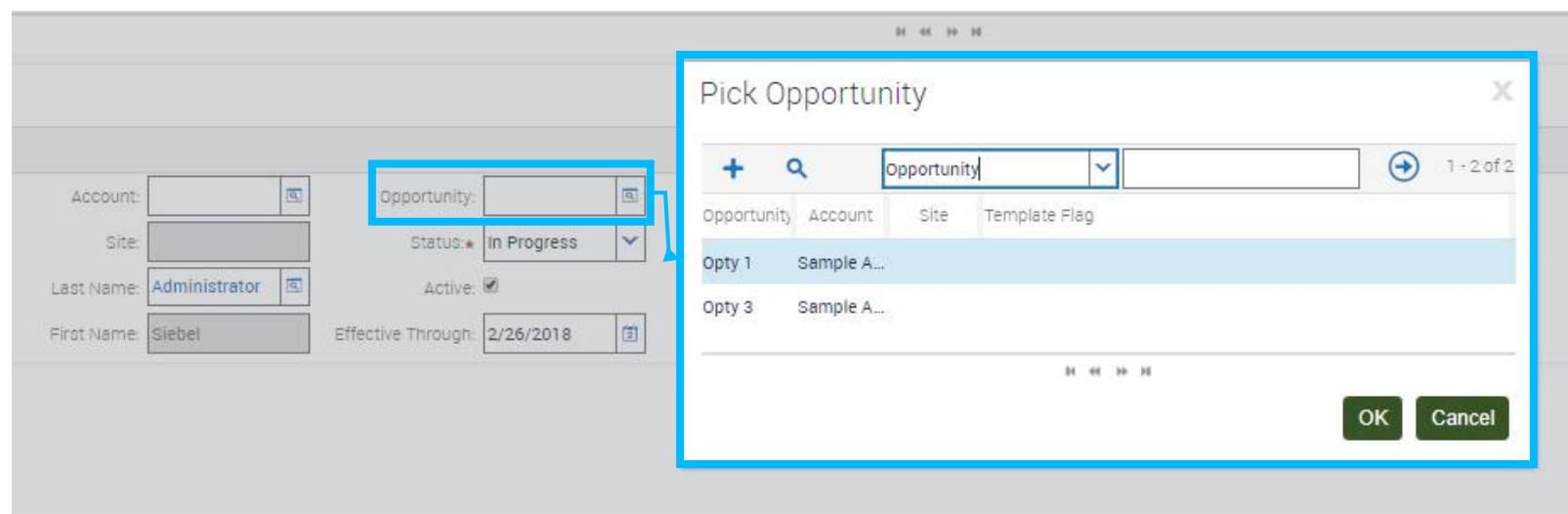
A dynamic picklist allows users to select records from a pick applet

Dynamic picklists are created on a joined field. It displays records from Pick BC.

- ✓ The values are dynamic in nature as they depend on the current BC

When a record is picked in the Pick Applet

- ✓ Copies the primary key into the Foreign key field
- ✓ Updates the values of the joined fields



# Picklists Overview (7)

## Picklist Generic

Business component with base table as S\_LST\_OF\_VAL

It is the Pick BC for the Static picklists

- ✓ Static picklists are based on the PickList Generic BC

The screenshot shows the Siebel Object Explorer interface. The left pane displays a tree view of Siebel objects, including Applet, Application, Business Component, Field, Join, Multi Value Field, Multi Value Link, Single Value Field, Business Object, Business Service, EIM Interface Table, and Entity Relationship Diagram. The 'Business Component' node is expanded, showing sub-options like BusComp Browser Script, BusComp Server Script, BusComp View Mode, and Business Component User F. The right pane is titled 'Field List' and contains two tables: 'Fields' and 'Business Components'. The 'Business Components' table lists a single entry: 'PickList Generic' with 'S\_LST\_OF\_VAL' as the table, 'Picklist' as the project, and 'CSSBusComp' as the class. The 'Fields' table lists several fields: Description (DESC\_TEXT, DTYPE\_TEXT), Name (NAME, DTYPE\_TEXT), Order By (ORDER\_BY, DTYPE\_INTEGER), Parent Id (PAR\_ROW\_ID, DTYPE\_ID), Type (TYPE, DTYPE\_TEXT), and Value (VAL, DTYPE\_TEXT). The 'Description' field is highlighted with a blue selection bar.

# Picklists Overview (8)

## Picklist Object Definition

Defines static or dynamic picklist

- ✓ Identifies the pick BC
- ✓ Also specifies if the picklist is Bounded or Non-Bounded

Object Explorer

Picklist List

Picklists

W	Name	Project	Static	Bounded	Business Component	Type Field	Type Value	No Delete	No Insert	No Merge	No Update
>	CUT Account Type PickList	VERT CUT Common	✓	✓	PickList Generic	Type	CUT_ACCOUNT_TYPE	✓	✓	✓	✓

Indicates Static picklist

LOV is Bounded

Static picklist

Pick BC

LOV Type

Object Explorer

Picklist List

Picklists

W	Name	Project	Static	Bounded	Business Component	Type Field	Type Value	No Delete	No Insert	No Merge	No Update
>	PickList Parent Account	Picklist		✓	Account			✓	✓	✓	✓

Dynamic picklist

Not used in the dynamic picklist



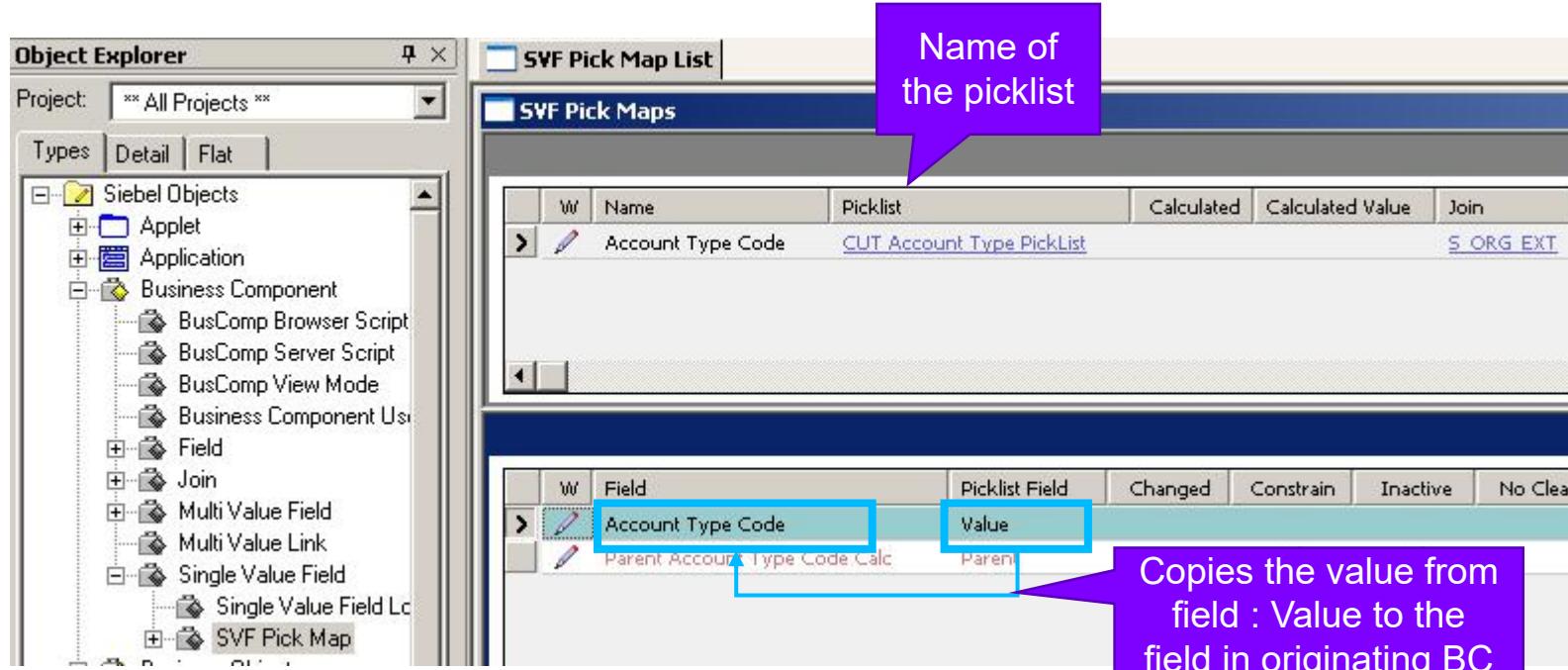
# Picklists Overview (9)

## Single Value Field and SVF Pick Map

Picklist Property in the Single Value field specifies the picklist associated with the field

- ✓ A dropdown is seen on runtime on this field

SVF Pick Maps specifies one or more fields to be copied from the pick BC to the originating BC



# Picklists Overview (10)

## SVF Pick Map

Through SVF pick maps many SVF can be updated with values from the pick BC

Dynamic Picklists will have following fields always copied :

- ✓ Foreign Key field to identify the pick record
- ✓ Additional fields that needs to be updated and displayed on the screen

The screenshot shows the Siebel Object Explorer and the SVF Pick Map List windows.

**Object Explorer:** Shows the project as "All Projects" and lists various object types under "Types".

**SVF Pick Map List:** Shows the "SVF Pick Maps" table with one entry:

W	Name	Picklist
>	Parent Account Name	PickList Parent Account

**SVF Pick Maps:** Shows the mapping details for the "Parent Account Name" picklist:

W	Field	Picklist Field
>	Account Type Code	Dummy
>	Master Account Id	Master Account Id
>	Master Account Name	Master Account Name
>	Parent Account Id	<b>Id</b>
>	Parent Account Integration Id	Integration Id
>	Parent Account Location	Location
>	Parent Account Name	<b>Name</b>
>	Parent Account Type Code	Account Type Code

Annotations highlight specific fields:

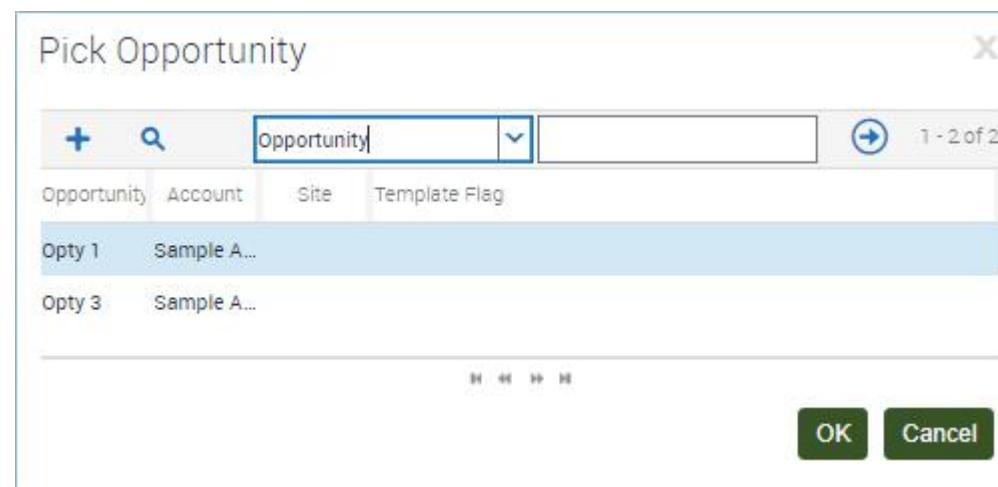
- A purple arrow points to the "Parent Account Id" row with the text: "Foreign Key field is updated with the Id of the picked record".
- A purple arrow points to the "Parent Account Name" row with the text: "Name is copied and is displayed in the UI".

# Picklists Overview (11)

## Pick Applet

Applet that displays the records for the dynamic picklist

- ✓ Displays multiple columns to make it easier for appropriate selection of record



# Picklists Overview (12)

## Control and List Column

Set the runtime to TRUE to enable picklist

For dynamic picklist , specify the pick applet property

For static picklist, pick applet property should be left blank

- ✓ Dropdown will be enabled at runtime

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer pane displays a tree structure of Siebel Objects, with 'List' selected under 'Applet'. The main area contains two tabs: 'List Column List' and 'List Columns'. The 'List Column List' tab shows a table with one row named 'List'. The 'List Columns' tab shows a table with two rows. The first row has 'Module' set to 'Account', 'Runtime' checked, 'Name' set to 'Account Type Code', 'Field' set to 'Account Type Code', 'Display Name' set to 'Account Class', and 'Pick Applet' left blank. The second row has 'Module' set to 'Parent', 'Runtime' checked, 'Name' set to 'Parent Account Name', 'Field' set to 'Parent Account Name', 'Display Name' set to 'Parent', and 'Pick Applet' set to 'Account Pick Applet'.

Module	Runtime	Name	Field	Display Name	Pick Applet
Account	✓	Account Type Code	Account Type Code	Account Class	
Parent	✓	Parent Account Name	Parent Account Name	Parent	<a href="#">Account Pick Applet</a>

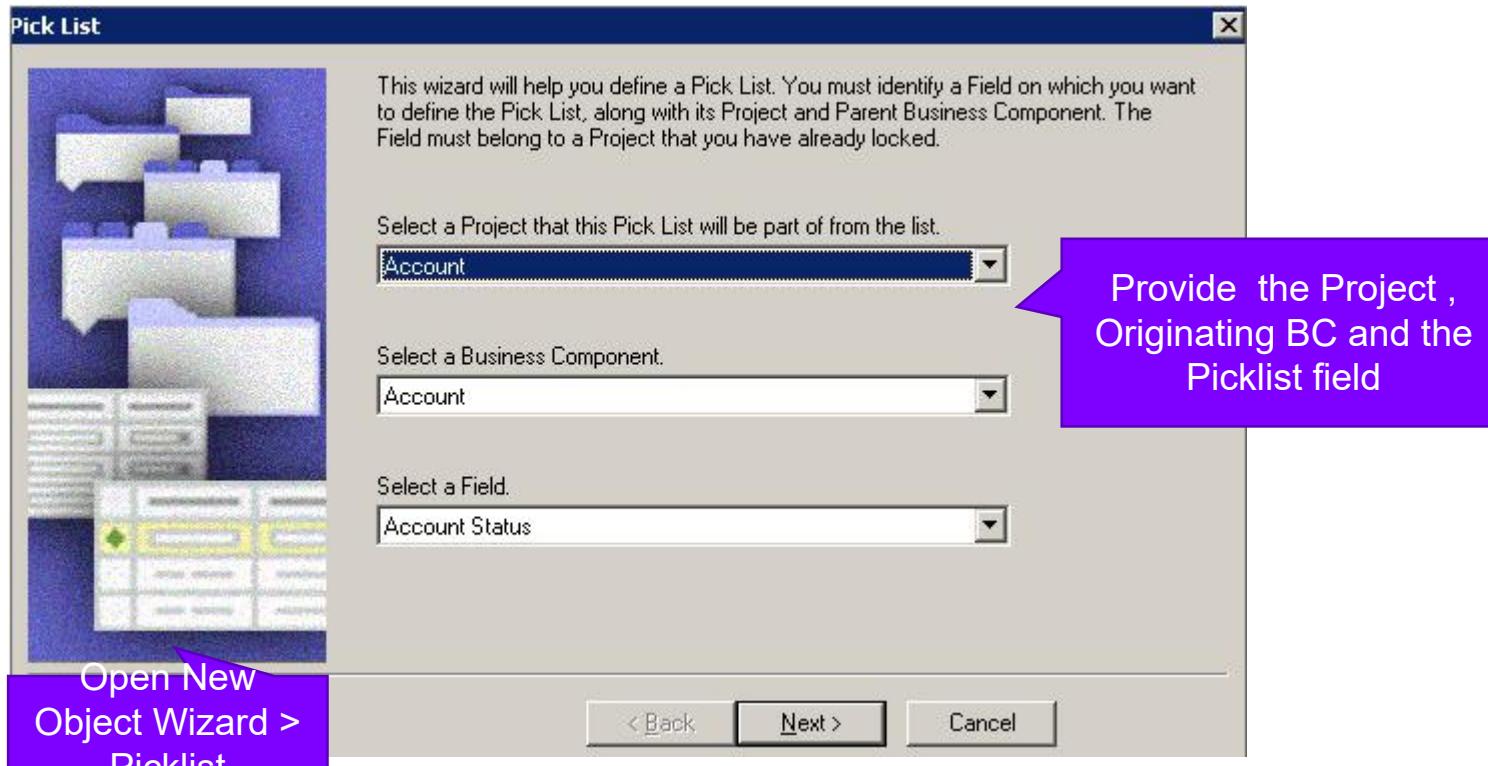
# Topic List

#No	Module Topics
1	Picklists Overview
2	Picklist Configuration
3	Additional Materials - Constrain Picklist
4	Additional Materials - Hierarchical Picklist

# Picklist Configuration (1)

## Steps

Create a picklist using the new object wizard



# Picklist Configuration (2)

## Static Picklist Properties

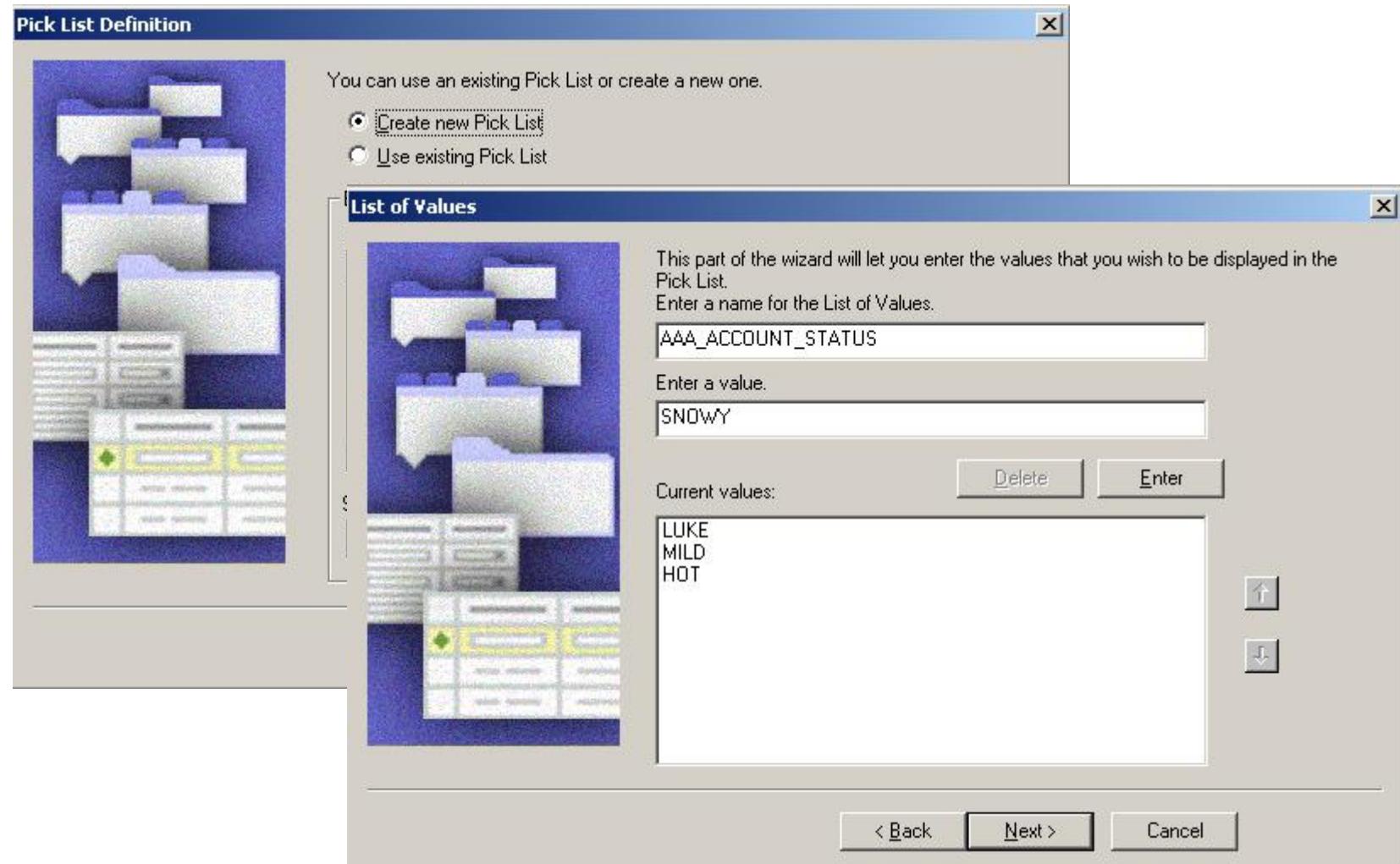
Provide the following details :

- ✓ Project: Name of the Project
- ✓ Business Component: Name of the Pick BC
- ✓ Field: Name of the picklist Field
- ✓ Type: Static Picklist

Select create new picklist

- ✓ Provide name for the picklist

Enter the LOV Type name and the Values



# Picklist Configuration (3)

## Pick List Wizard

Creates the following :

- ✓ Values in the S\_LST\_OF\_VAL table
- ✓ Static Picklist Object definition with the specific LOV Type
- ✓ SVF Pick maps to the copy the Pick BC field values to the originating BC

Sets Runtime Property to TRUE in all list columns and controls that refers to any static picklist

- ✓ To active picklist at runtime



# Picklist Configuration (4)

## Dynamic Picklist Configuration

Using New Object Wizard configure a dynamic picklist:

- ✓ Select Project
- ✓ Select the Originating BC
- ✓ Provide the Field Name for the Picklist field
- ✓ Select type as Dynamic PickList

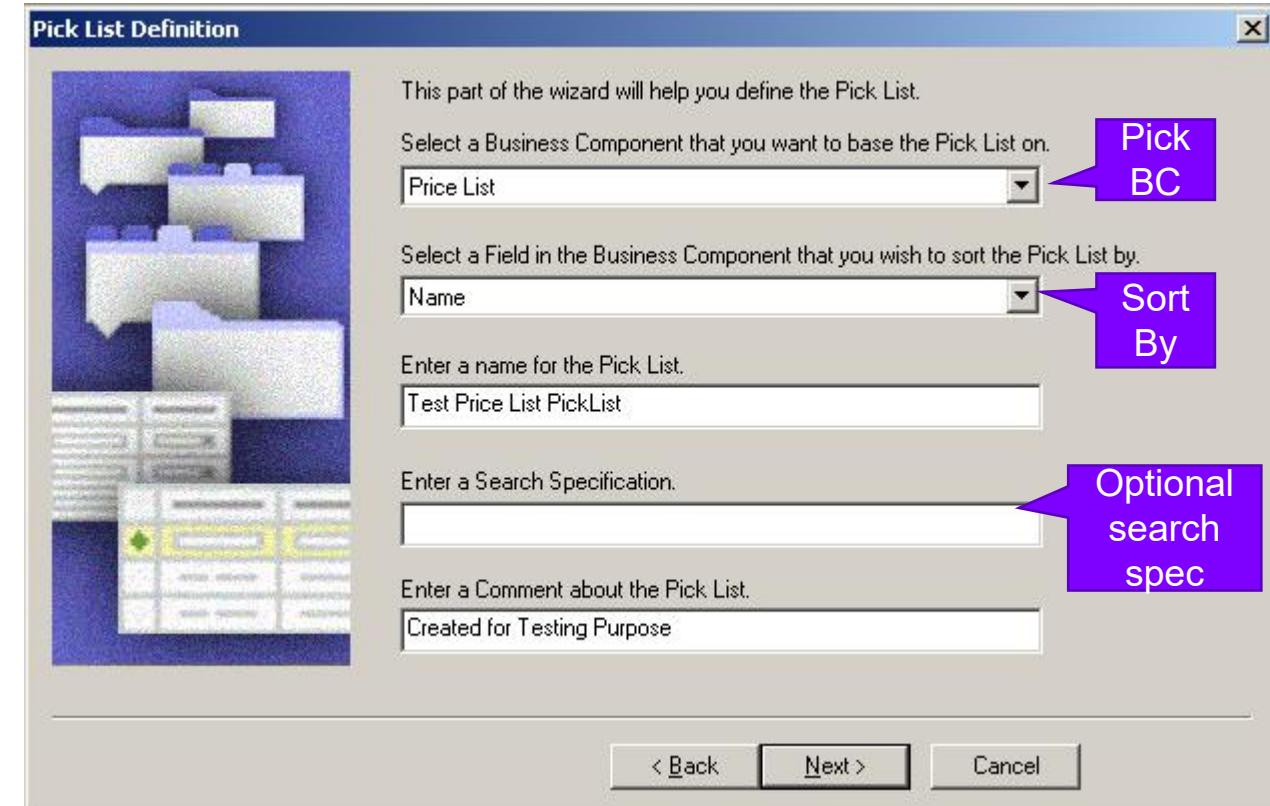
Wizards prompts for a new picklist creation or use existing picklist

- ✓ Choose appropriately

Select:

- ✓ Pick BC
- ✓ Sort Field in case required
- ✓ Provide the name of the picklist and Search Spec if any

No Update/No Delete/No Merge/No Insert properties can be configured next

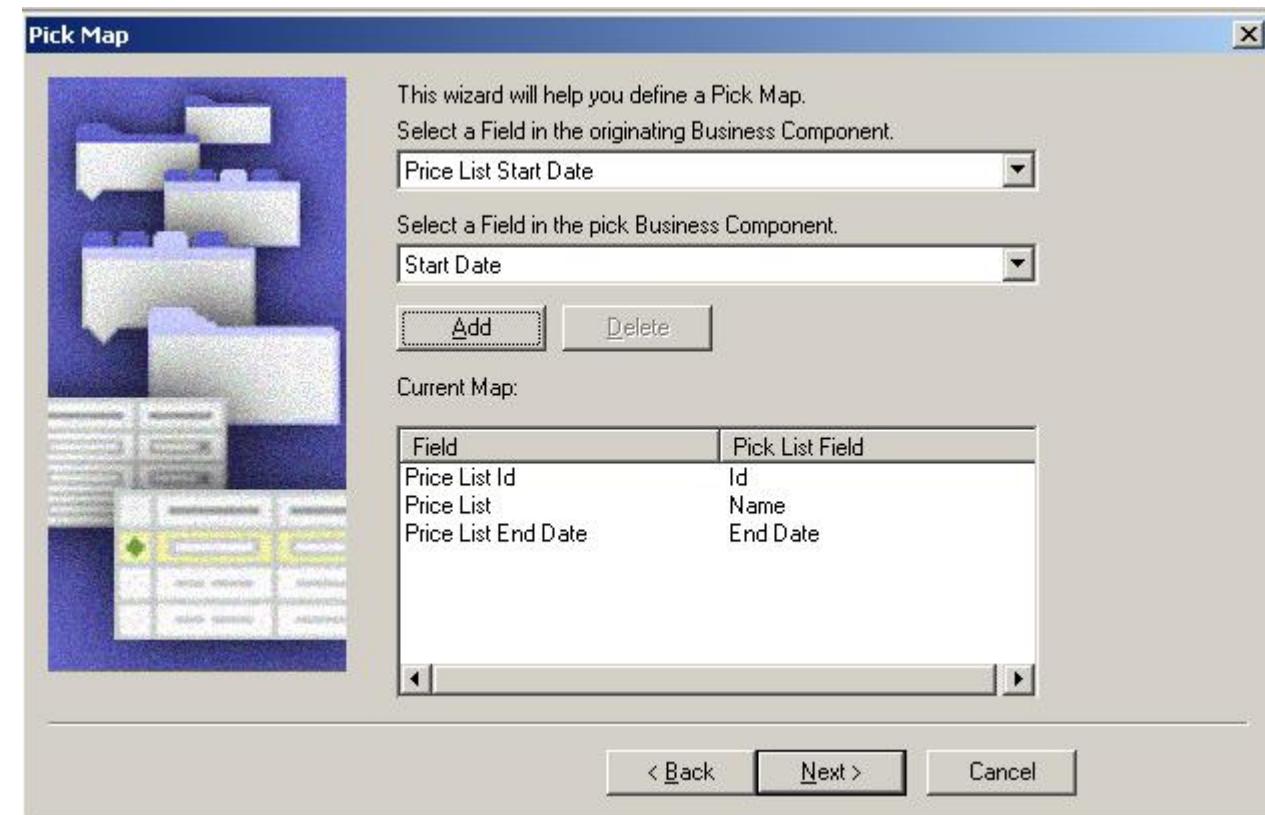


# Configure Picklist (5)

## Define Pickmaps for Dynamic Picklist

Wizard helps to describe the pick maps and the foreign keys

- ✓ Create multiple pick maps
  - Map the primary key in the pick BC to the foreign key field in the originating BC



# Configure Picklist (6)

## Pick List Wizard for Dynamic Picklist

Creates the following :

- ✓ Dynamic Picklist Object definition with Pick BC
- ✓ SVF Pick maps to copy the field values from the Pick BC to the originating BC

Sets Runtime Property to TRUE in all controls and list columns referring to the static picklist

- ✓ To active picklist at runtime

The wizards also open a Applet Wizard to create a pick applet if required



# Topic List

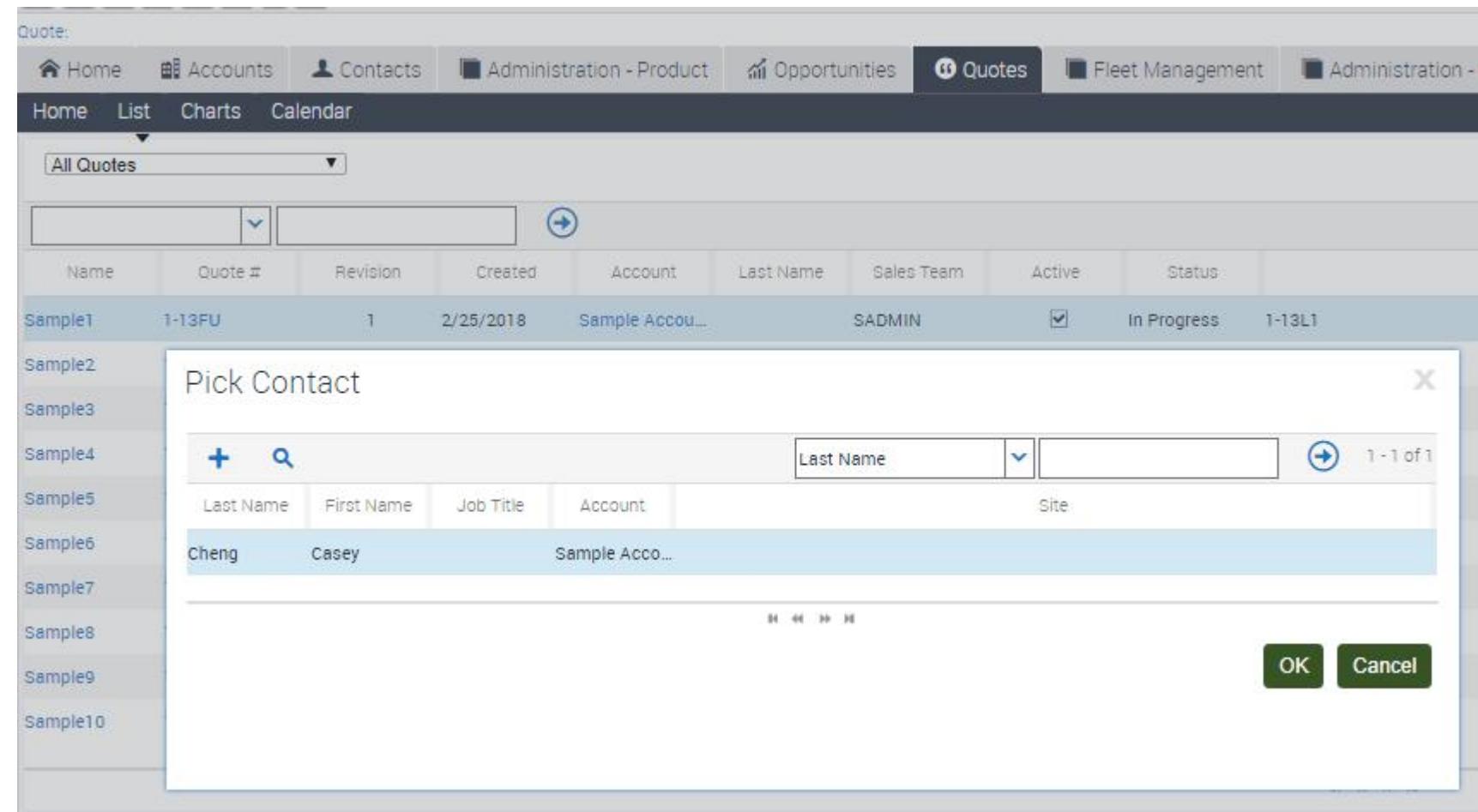
#No	Module Topics
1	Picklists Overview
2	Picklist Configuration
3	Additional Materials - Constrain Picklist
4	Additional Materials - Hierarchical Picklist

# Additional Materials—Constrain Picklist (1)

## Introduction

Constrain picklist filter records dynamically that have field values matching that of the corresponding fields in the originating BC

- ✓ Example: Contacts displayed in the quote list applet is dependant on the Account associated with the quote



# Additional Materials—Constrain Picklist (1)

To constrain a picklist :

Create Pick Maps

Create a SVF Pick Map that needs to be matched with the originating BC.

- ✓ Set the Constrain Flag to TRUE
- ✓ Picklist now filters out only the matching records

The screenshot shows the Siebel Object Explorer and the SVF Pick Map List windows. In the Object Explorer, under Siebel Objects > Applet > Single Value Field, a 'SVF Pick Map' is selected. In the SVF Pick Map List window, a new entry 'Contact Last Name' is being created, mapping to the S\_CONTACT table's LAST\_NAME field. This entry is highlighted with a blue border. In the SVF Pick Maps window, multiple fields are listed, and 'Joined Account Id' is checked under the 'Constrain' column. A callout box points to this field with the text 'Returns Contact records if Account Id matches'. Another callout box at the bottom points to the 'Constrain' column with the text 'Copies values for these fields'.

SVF Pick Map List

Name	Changed	Calculated	Calculated Value	Join	Column	Currency Code Field	Exchange
Contact Last Name	S_CONTACT	LAST_NAME					

SVF Pick Maps

Field	Changed	Constrain	Inactive	No Clear	Picklist Field	Comments	Module
Contact First Name					First Name		
Contact Id					Id		
Contact Integration Id					Integration Id		
Contact Last Name					Last Name		
Contact Primary Account Address Id					Primary Address Id	Restored to be the s.	
Contact Primary Payment Profile Id					Primary Personal Payment Profile Id	Restored to be the s.	
Contact Primary Personal Address Id					Primary Personal Address Id	Restored to be the s.	
Contact Work Phone #					Work Phone #	Restored to be the s.	
Joined Account Id		✓			Account Id	CR#12-PK10VK	
Personal Bill To Address Id					Primary Personal Address Id	Restored to be the s.	
Personal Bill To City					Personal City	Restored to be the s.	
Personal Bill To Country					Personal Country	Restored to be the s.	
Personal Bill To Postal Code					Personal Postal Code	Restored to be the s.	
Personal Bill To State					Personal State	Restored to be the s.	
Personal Bill To Street Address					Personal Street Address	Restored to be the s.	
Personal Ship To Address Id					Primary Personal Address Id	Restored to be the s.	



# Topic List

#No	Module Topics
1	Picklists Overview
2	Picklist Configuration
3	Additional Materials - Constrain Picklist
4	Additional Materials - Hierarchical Picklist

# Additional Materials—Hierarchical Picklist (1)

## Overview

A hierarchical picklist shows values which is dependant on value chosen from another picklist.

- ✓ Example: the Area and Subarea fields in the Service Request Detail Applet, are both picklists fetching values from the List of values table (S\_LST\_OF\_VAL).
- ✓ The LOVs available in the Subarea picklist is dependant on value selected in the Area picklist.

The screenshot shows the SR Information section of the Service Request Detail Applet. The 'Area' field is set to 'Engine'. A dropdown menu is open over the 'Subarea' field, listing three options: 'Failure to Start', 'Knocking', and 'Stalling'. A blue box highlights the 'Subarea' field and its dropdown menu. Another blue box highlights the 'Area' field. Arrows point from the highlighted fields to their respective LOV lists.

The screenshot shows the SR Information section of the Service Request Detail Applet. The 'Area' field is set to 'Comfort'. A dropdown menu is open over the 'Subarea' field, listing two options: 'Rattle' and 'Wind Noise'. A blue box highlights the 'Subarea' field and its dropdown menu. Another blue box highlights the 'Area' field. Arrows point from the highlighted fields to their respective LOV lists.

# Additional Materials—Hierarchical Picklist (2)

List of Values for Area and Sub Area are specified using the one LOV Type, SR\_AREA LOV in S\_LST\_OF\_VAL

Parent LIC is used to state value of the parent.

- ✓ For each child value, parent LIC is updated with the parent value

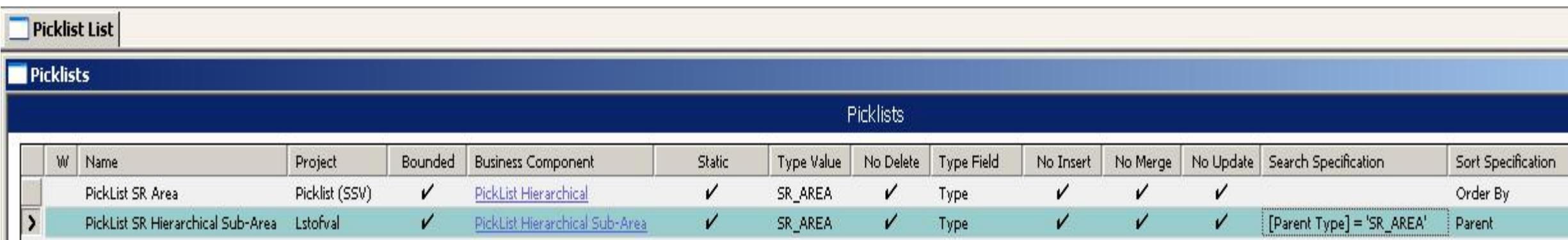
The screenshot shows a List of Values page in Oracle Application Express. The top navigation bar includes File, Edit, View, Navigate, Query, Tools, and Help. Below the navigation is a toolbar with various icons. The main menu has Home, Accounts, Contacts, Administration - Product, Opportunities, Quotes, Fleet Management, Administration - Fleet Management, Sales Orders, Service, Calendar, and Administration - Data. The sub-menu for Administration - Data includes Units of Measurement Administration, Project Mappings, Projects, Fulfillment Centers, Activity Templates, Competitors, Decision Issues, Locale, Currencies, Industries, Inventory Locations, Languages, List of Values Explorer, and List of Values. The current page is 'List of Values' under 'Administration - Data'. A purple callout box labeled 'Parent Area Values' points to the first two rows of the table, which show 'Comfort' and 'Engine' as parent values. Another purple callout box labeled 'Child Sub Area Values' points to the bottom five rows, which show 'Wind Noise', 'Rattle', 'Knocking', 'Stalling', and 'Failure to Start' as child values, all associated with the 'Comfort' and 'Engine' parent values respectively. The table columns include Type, Display Value, Language Name, Language-Independent Code, Parent LIC, Order, Active, Translate, Multilingual, and Replication Level.

Type	Display Value	Language Name	Language-Independent Code	Parent LIC	Order	Active	Translate	Multilingual	Replication Level
SR_AREA	Comfort	English-American	Comfort	External	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	
SR_AREA	Engine	English-American	Engine	External	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	
Type	Display Value	Language Name	Language-Independent Code	Parent LIC	Order	Active	Translate	Multilingual	Replication Level
SR_AREA	Wind Noise	English-American	Wind Noise	Comfort	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	
SR_AREA	Rattle	English-American	Rattle	Comfort	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	
SR_AREA	Knocking	English-American	Knocking	Engine	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	
SR_AREA	Stalling	English-American	Stalling	Engine	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	
SR_AREA	Failure to Start	English-American	Failure to Start	Engine	1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	All	

# Additional Materials—Hierarchical Picklist (3)

To configure picklists to support this hierarchy :

- ✓ The parent picklist is based on the PickList Hierarchical BC
- ✓ The child picklist is based on the PickList Hierarchical Sub-Area BC
- ✓ Also a search spec is added for Child Picklist to retrieve Parent type value selected
  - Parent Type = SR\_AREA selected



The screenshot shows the Oracle ADF Faces application interface for configuring picklists. The top navigation bar has tabs for 'Picklist List' and 'Picklists'. The main content area is titled 'Picklists' and displays a table of picklist configurations.

W	Name	Project	Bounded	Business Component	Static	Type Value	No Delete	Type Field	No Insert	No Merge	No Update	Search Specification	Sort Specification
	PickList SR Area	Picklist (SSV)	✓	<a href="#">PickList Hierarchical</a>	✓	SR_AREA	✓	Type	✓	✓	✓		Order By
➤	PickList SR Hierarchical Sub-Area	Lstofval	✓	<a href="#">PickList Hierarchical Sub-Area</a>	✓	SR_AREA	✓	Type	✓	✓	✓	[Parent Type] = 'SR_AREA'	Parent

# Additional Materials—Hierarchical Picklist (4)

## SVF Maps for Hierarchical Picklists

Create SVF Pick map to copy the field values after associating the picklist

Set constrain property to TRUE for the child picklist on parent field

- ✓ To constrain the value of sub area based on the area chosen

Picklist fields in the originating BC

SVF Pick Maps created for the parent picklist

SVF Pick Maps for the child picklist

The screenshot shows the Oracle Business Components interface with several windows open:

- Single Value Field List:** Shows a table with columns W, Name, Changed, Table, Project, Cache Data, Class. One row is selected: Service Request (SR\_SRV\_REQ) under Project, CSSBServiceRequest under Class.
- Single Value Fields:** Shows a table with columns W, Name, Join, Column, Picklist, Type, Text Length, Calculated. Two rows are listed:
  - Area (SR\_AREA) with Picklist: PickList\_SR\_Area, Type: DTYPE\_TEXT, Text Length: 30.
  - Sub-Area (SR\_SUB\_AREA) with Picklist: PickList\_SR\_Hierarchical\_Sub-Area, Type: DTYPE\_TEXT, Text Length: 30.
- SVF Pick Map List:** Shows a table with columns W, Name, Changed, Calculated, Calculated Value, Join, Column, Currency Code Field. One row is selected: Area (SR\_AREA) with Join: SR\_SUB\_AREA.
- SVF Pick Maps:** Shows a table with columns W, Field, Changed, Constrain, Inactive, No Clear, Picklist Field, Comments. One row is selected: Area (Field: Area, Picklist Field: Value). The Constrain checkbox is checked.
- SVF Pick Map List:** Shows a table with columns W, Name, Changed, Calculated, Calculated Value, Join, Column, Currency Code Field. Two rows are listed:
  - Area (SR\_AREA) with Join: SR\_SUB\_AREA.
  - Sub-Area (SR\_SUB\_AREA).
- SVF Pick Maps:** Shows a table with columns W, Field, Changed, Constrain, Inactive, No Clear, Picklist Field, Comments. Three rows are listed:
  - Area (Field: Area, Picklist Field: Parent).
  - CurrentArea (Field: CurrentArea, Picklist Field: Parent). The Constrain checkbox is checked.
  - Sub-Area (Field: Sub-Area, Picklist Field: Value). The Constrain checkbox is checked.A message "This is causing problem!!" is displayed next to the CurrentArea row.



# Knowledge Checks

## Answer the following

1. Dynamic picklist displays values from a dropdown for user selection. TRUE / FALSE ?
2. Administration – Data > List of Values Screen is where the static picklist data is administered? TRUE/FALSE ?



# Module Summary

**Now, you should be able to:**

- Define picklists
- Identify the different types of picklist
- Explain how to:
  - Configure picklists
  - Administer the static picklist



# **Thank You**

# SIEBEL

Multi Value Groups

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Explain how to:
  - Add Multi Value Fields to BC
  - Create MVG applet to display the child data
  - Create Primary for an MVG



# Topic List

#No	Module Topics
1	Multi Value Group
2	Primaries in MVG
3	Creating MVG

# Topic List

#No	Module Topics
1	Multi Value Group
2	Primaries in MVG
3	Creating MVG

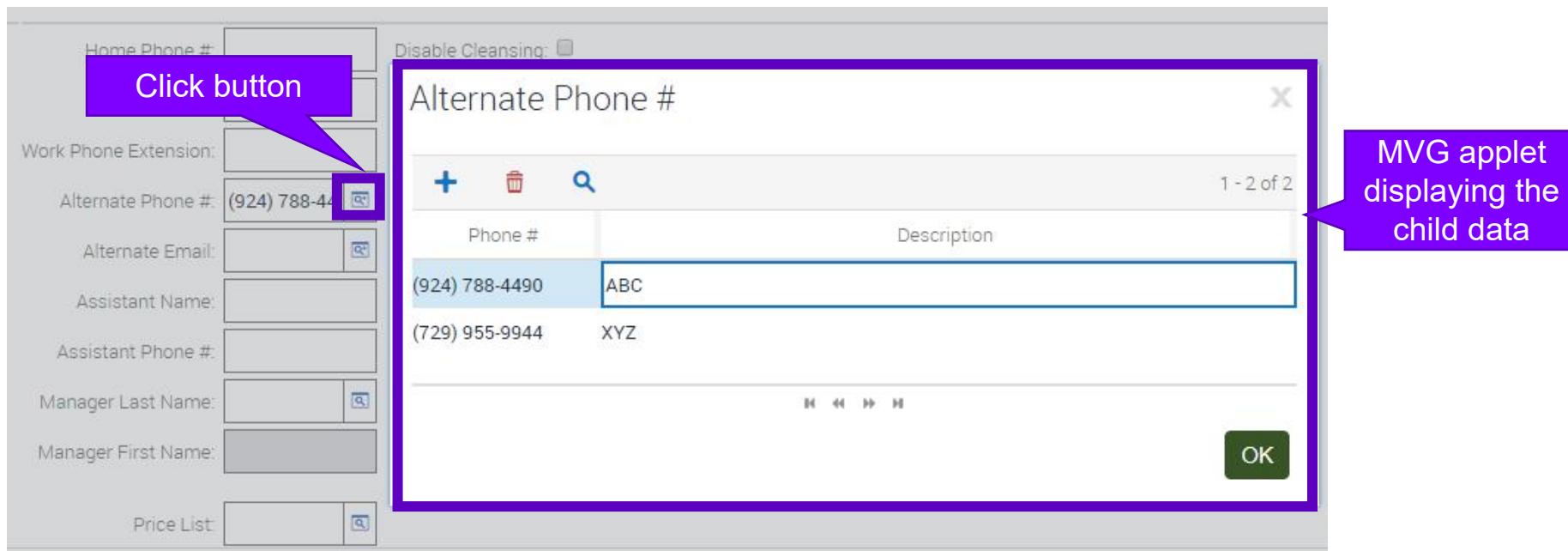
# Multi Value Group (1)

## Introduction

A multi-value group is a UI element that displays child data in an applet

Displays multiple child records for a single parent record

MVG button on parent applet is clicked to pop MVG applet which contains child data



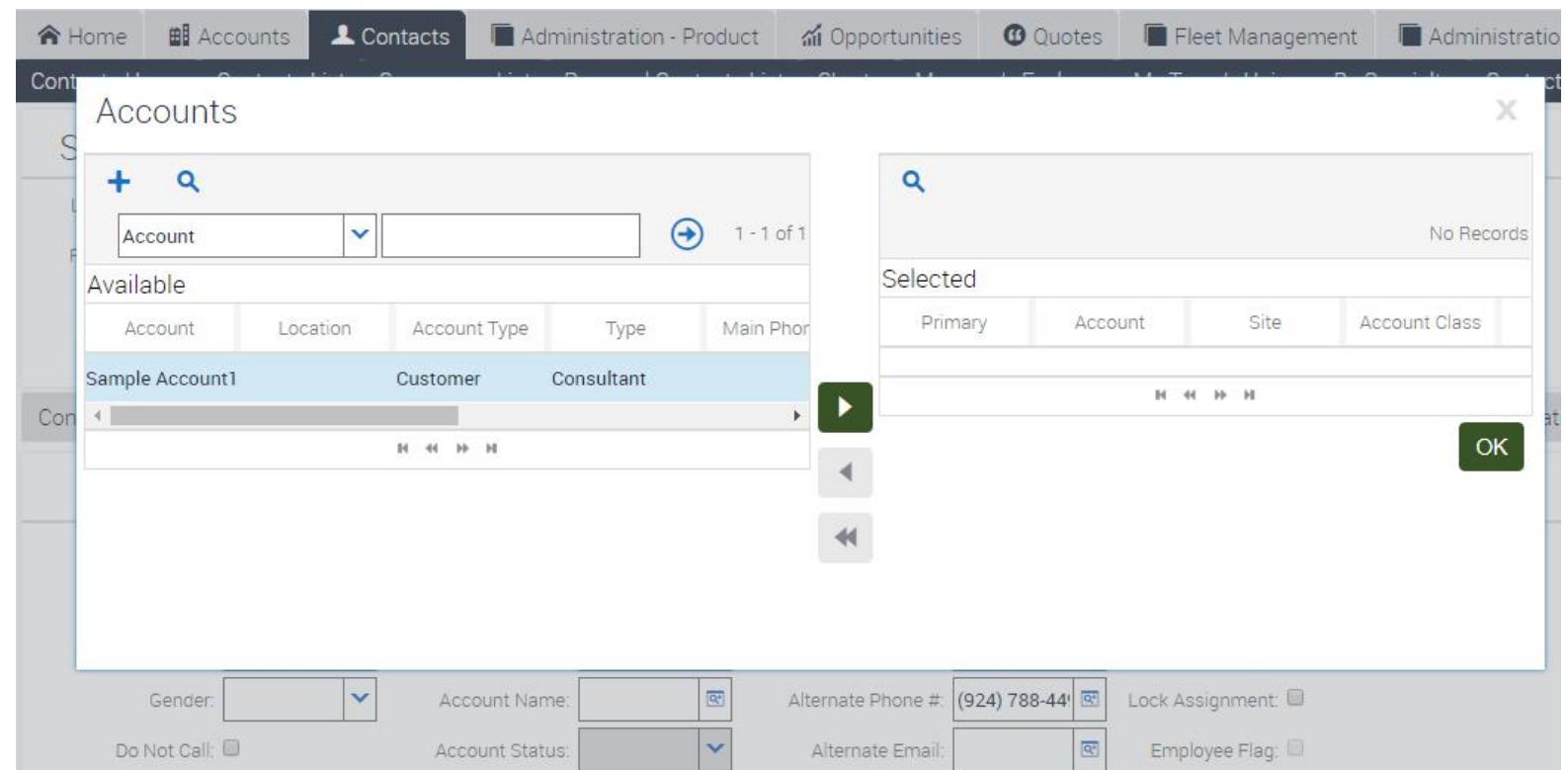
# Multi Value Group (2)

## Shuttle Applet

MVG for a M:M relationship displays Shuttle Applet

Displays both available child records on the left and selected child records on the right

Applet appears only in HI Mode



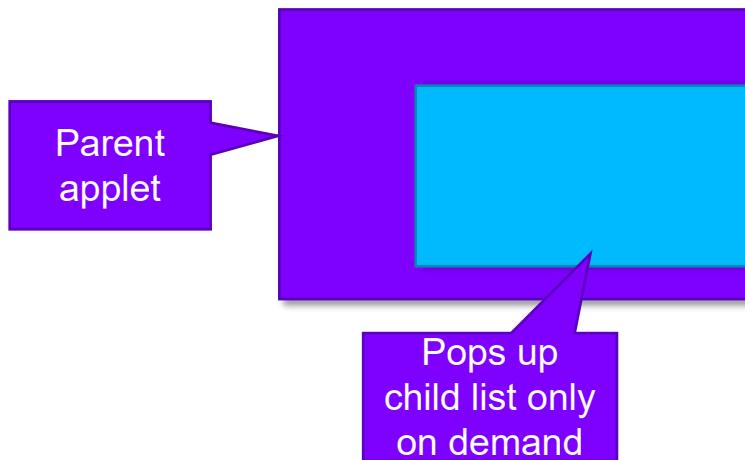
# Multi Value Group (3)

## MVG Display

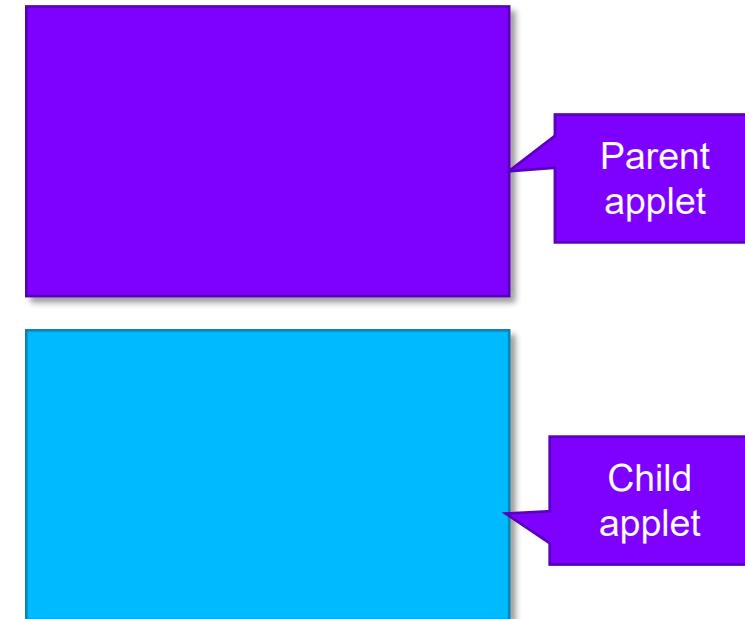
MVGs are an alternative to detail views where parent and related child records are displayed

It makes:

- ✓ effective use of the space
  - Does not require dedicated space in a view
- ✓ easier to display multiple set of detailed records available from a single view



V/s



# Multi Value Group (4)

## Advantages of MVGs

Allows creating queries that include both parent and child fields

First Name:★ <input type="text" value="&lt;Case Sensitive&gt;"/>	Contact Status: <input type="text" value="&lt;Case Sensitive&gt;"/>	Home Phone #:
Middle Initial: <input type="text" value="&lt;Case Sensitive&gt;"/>	Households: <input type="text" value="&lt;Case Sensitive&gt;"/>	Work Phone #:
Last Name:★ <input type="text" value="&lt;Case Sensitive&gt;"/>	Household Status: <input type="text" value="&lt;Case Sensitive&gt;"/>	Work Phone Extension: <input type="text" value="&lt;Case Sensitive&gt;"/>
Gender: <input type="text" value="&lt;Case Sensitive&gt;"/>	Account Name: <input type="text" value="&lt;Case Sensitive&gt;"/>	Alternate Phone #:
Do Not Call: <input type="checkbox"/>	Account Status: <input type="text" value="&lt;Case Sensitive&gt;"/>	Alternate Email: <input type="text" value="&lt;Case Sensitive&gt;"/>
Do Not Email: <input type="checkbox"/>	Contact Team: <input type="text" value="&lt;Case Sensitive&gt;"/>	Assistant Name: <input type="text" value="&lt;Case Sensitive&gt;"/>
Do Not Mail: <input type="checkbox"/>	Global Owner: <input type="text" value="&lt;Case Sensitive&gt;"/>	Assistant Phone #:
Contact Method: <input type="text" value="&lt;Case Sensitive&gt;"/>	Organization: <input type="text" value="&lt;Case Sensitive&gt;"/>	Manager Last Name: <input type="text" value="&lt;Case Sensitive&gt;"/>
Send Email Updates: <input type="checkbox"/>	Registration Source: <input type="text" value="&lt;Case Sensitive&gt;"/>	Manager First Name: <input type="text" value="&lt;Case Sensitive&gt;"/>
Comments: <input type="text" value="&lt;Case Sensitive&gt;"/>		Price List: <input type="text" value="&lt;Case Sensitive&gt;"/>

Query on Contact status

Query on First Name

Query on Alternate Phone number

Query on Organization

# Multi Value Group (5)

## MVG Concepts

Basic object definitions that are needed for MVG:

- Multi Value Fields (MVF)
- Multi Value Links (MVL)
- Map MVFs to the Child BC SVF
- MVG Applet
- Association Applet
- Primaries for the Performance

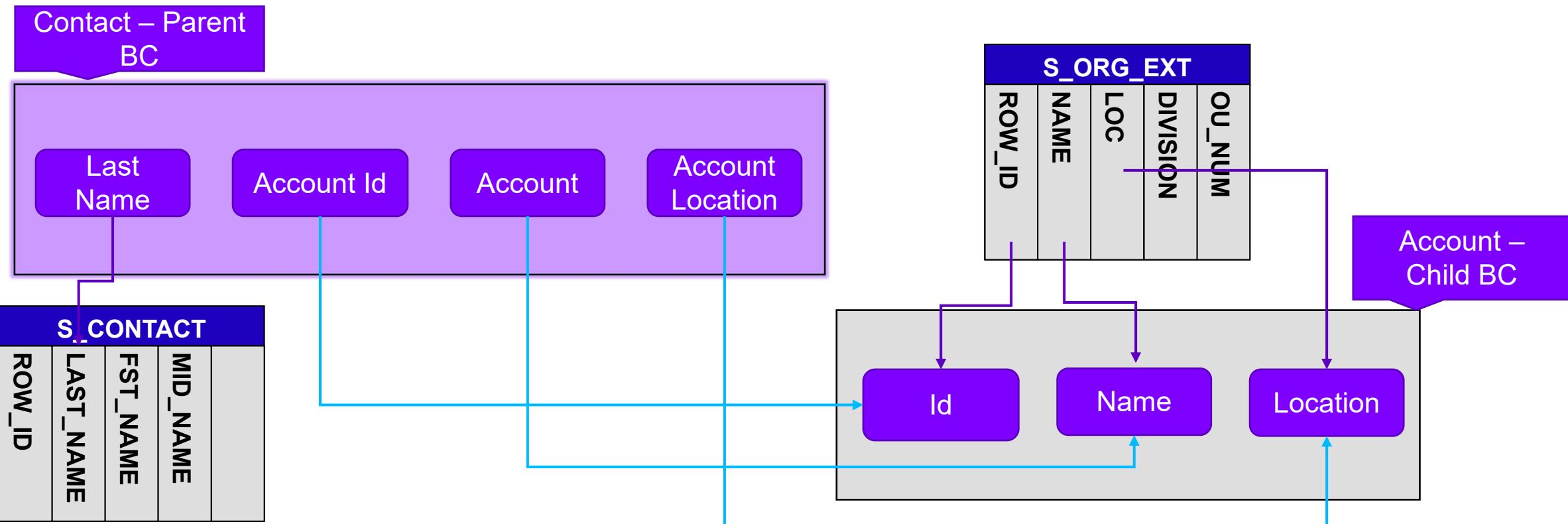


# Multi Value Group (6)

## Multi Value Field (MVF)

MVF is a field in the parent BC referring to field in Child BC

- ✓ Never refers a column directly



# Multi Value Group (7)

## Multi Value Link (MVL)

Child object definition of BC that indicates retrieval of data from the Child BC

References Link object definition which explains relationship between the Parent and Child BC

The screenshot shows the Siebel Object Explorer interface with the following details:

- Object Explorer:** Shows the project "All Projects" and various object types under "Siebel Objects" and "Business Component".
- Multivalue Link List:** A list of Multivalue Links. One entry is shown for "Contact" with "CSSBCCContactSIS" as the Data Source.
- Multivalue Links:** A table showing the relationship between Business Components. It lists "Account" as the Destination Business Component and "Contact/Account" as the Destination Link.
- Links:** A table showing the detailed link definition. It lists "Contact/Account" as the Name, "Contact" as the Parent Business Component, and "Account" as the Child Business Component. The Inter Table is "S\_PARTY\_PER", the Inter Parent Column is "PERSON\_ID", and the Inter Child Column is "PARTY\_ID".

# Multi Value Group (8)

## MVF and MVL

Every MVF requires a MVL

MVF is mapped to SVF of  
Child BC through MVL

The screenshot shows the Siebel Object Explorer interface with the 'Multivalue Field List' and 'Multivalue Fields' windows open.

**Object Explorer:** Shows the Siebel Objects hierarchy, with 'Multi Value Field' selected under 'Business Component'.

**Multivalue Field List:** A table titled 'Multivalue Fields' showing a single entry for 'Contact'.

W	Name	Changed	Project	Cache Data	Class	Data Source
>	Contact		Contact			<a href="#">CSSBCCContactSIS</a>

**Multivalue Fields:** A table titled 'Multivalue Fields' listing various fields for 'Account'. The first row ('Account') is highlighted.

W	Name	Changed	Multivalue Link	Field
>	Account		Account	Name
	Account Currency Code		Account	Currency Code
	Account Location		Account	Location
	Account Mod Id		Account	Modification Number
	Account Number		Account	Account Number
	Account Organization		Account	Primary Organization
	Account Party UId		Account	Party UId
	Account Primary Bill To Address Id		Account	Primary Bill To Address Id
	Account Primary Bill To Person Id		Account	Primary Bill To Person Id
	Account Primary Market		Account	Primary Market
	Account Primary Ship To Address Id		Account	Primary Ship To Address Id
	Account Primary Ship To Person Id		Account	Primary Ship To Person Id
	Account Row Id		Account	Id
	Account Status		Account	Account Status
	Account Street Address		Account	Street Address
	Account Type		Account	Type
	Affiliated Account Id		Account	Id
	CSN		Account	CSN
	Joined Account Id		Account	Id

**Annotations:**

- A purple arrow points from the 'Parent BC Fields' label to the 'Account' row in the Multivalue Fields table.
- A purple arrow points from the 'Child BC Fields' label to the 'Name' column in the same table.

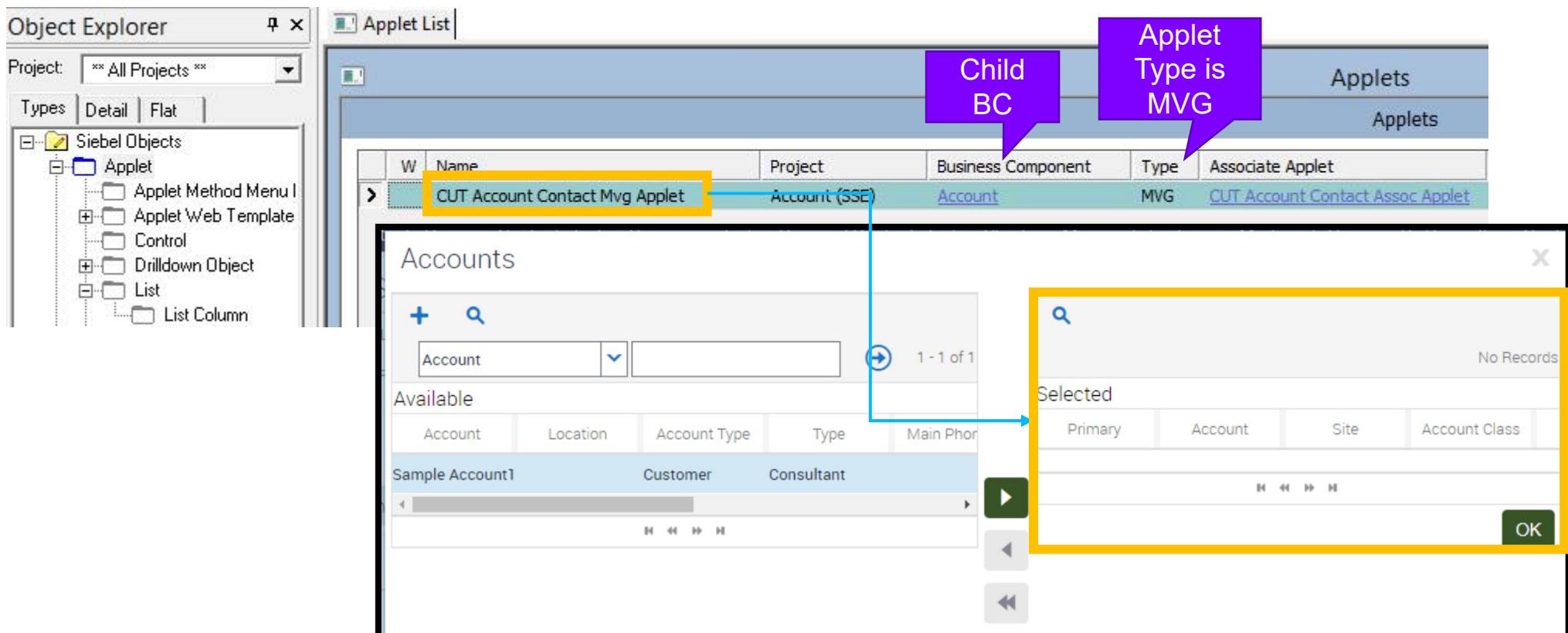
# Multi Value Group (9)

## MVG Applet

This is a list applet where child records are displayed

MVG button when clicked from a list or form applet pops up MVG applet.

It is based on Child BC

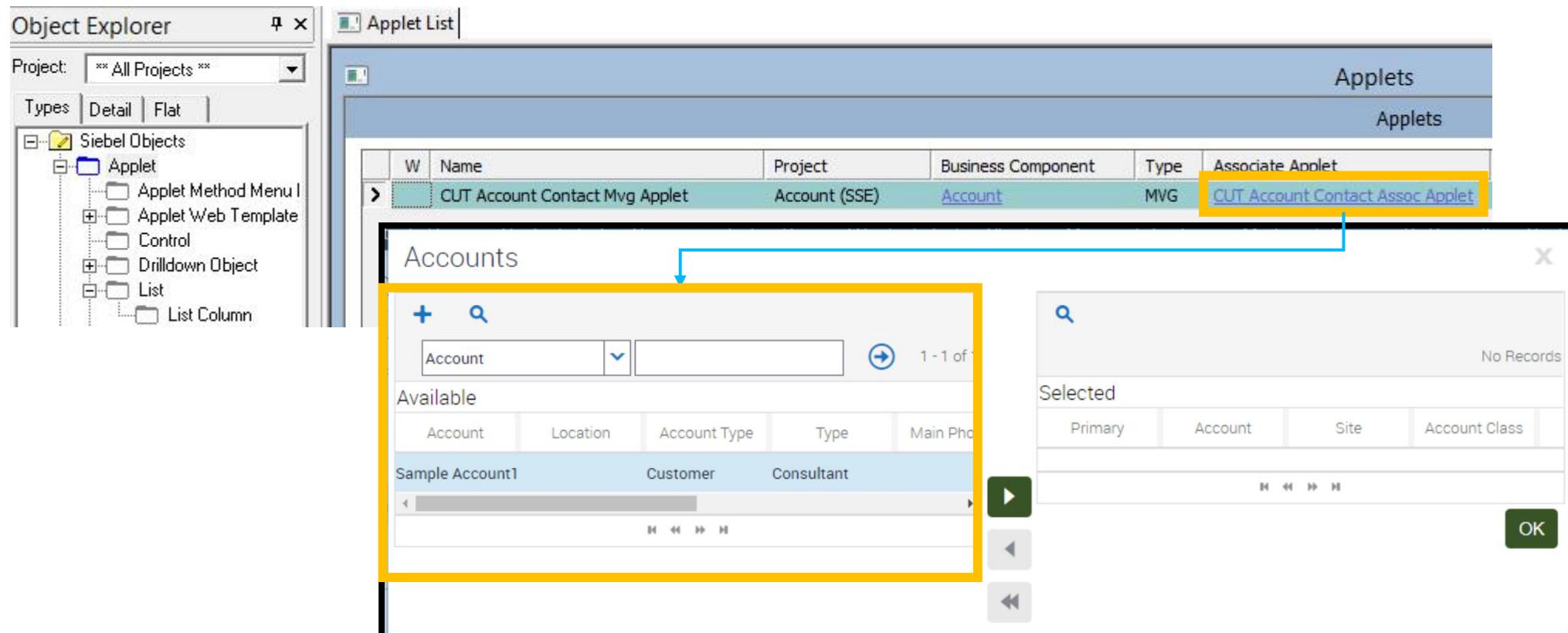


# Multi Value Group (10)

## Association Applet

Required for MVG with M:M relationship

Displays the list of child records available for selection



# Multi Value Group (11)

## Invoke MVG Applet

Specify the MVG applet to be invoked from the List Column / Control in MVG Applet Property

Set Runtime Property = TRUE, to enable the MVG applet

The screenshot shows the Siebel Object Explorer on the left and the List Column List on the right.

**Object Explorer:**

- Project: \*\* All Projects \*\*
- Types: Siebel Objects, Applet, Application, Business Component
- Applet subtypes: Applet Method Menu, Applet Web Template, Control, Drilldown Object, List, Tree
- Business Component subtypes: BusComp View Mode, Field, Join

**List Column List:**

**Lists:**

W	Name	Changed	HTML Hierarchy Bitmap	HTML Multi-Row	HTML Multi-Row	Total Displayed
>	List					✓

**MVG Applet Properties:**

Runtime	Name	Field	MVG Applet	Display Name	Changed	Available	Available - Langu
✓	Account	Account	CUT Account Contact Mva Applet	Account		✓	

# Topic List

#No	Module Topics
1	Multi Value Group
2	Primaries in MVG
3	Creating MVG

# Primaries in MVG (1)

## Introduction

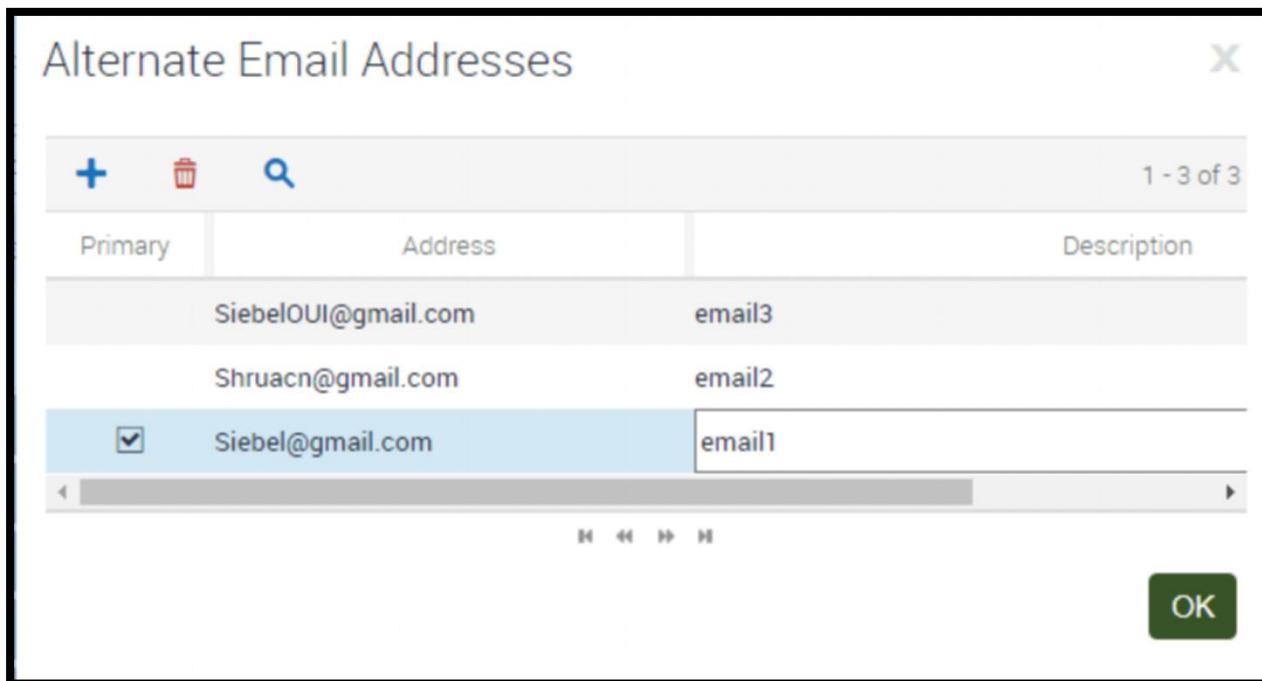
Each MVG Applet will have a primary

Allows faster retrieval of child data for display in the parent applet

- ✓ This record is designated as primary

Primaries improves performance by eliminating query to the Child BC separately

- ✓ One query will populate all the fields in the list applet



# Primaries in MVG (2)

## Primary Foreign Key

Primary Foreign Key field is a FK field in the parent BC which references the PK of the Child BC

- ✓ Child data is retrieved using the join to the child table



# Primaries in MVG (3)

## Primary Foreign Key

Siebel data model includes Primary Keys for many relationships

- ✓ These keys begin with PR\_

To find Primary Foreign Key:

- ✓ Select the table of the primary BC
- ✓ Query in Foreign Key Table for Child BC

Required	W	Name	User Name	Alias	Type	Foreign Key Table	Primary Key	Physical Type
>		PR_DEPT_OU_ID	Primary Account or Organization	PRIMARY_DEPT_OU_ID	Data (Public)	S_ORG_EXT		Varchar



# Primaries in MVG (4)

## Primary Foreign Key

Map the Primary Foreign Key column to a field in the parent BC

In MVL , set Use Primary Join and Primary Id Field properties

The screenshot shows the Siebel Object Explorer and a Properties window for a 'Multi Value Link [Account]' object.

**Object Explorer:** Shows the project 'xx All Projects xx' and various object types under 'Siebel Objects'.

**Field List:** Shows the 'Fields' section for the 'Contact' business component ('S\_PARTY'). A specific field, 'Account Id', is selected and highlighted with a purple border. It is mapped to the 'S\_CONTACT' column and the 'PR\_DEPT\_OU\_ID' picklist.

**Properties Window:** Displays properties for 'Multi Value Link [Account]'. The 'Primary Id Field' is set to 'Account Id' and 'Use Primary Join' is set to 'TRUE'. Other properties like 'Auto Primary', 'Check No Match', and 'Destination Link' are also listed.

Properties	
Multi Value Link [Account]	
Alphabetic	Categorized
Auto Primary	Default
Check No Match	FALSE
Comments	
Destination Business Component	Account
Destination Link	Contact/Account
Inactive	FALSE
Module	
Name	Account
No Associate	FALSE
No Copy	FALSE
No Delete	FALSE
No Insert	FALSE
No Update	FALSE
Parent Name	Contact
Popup Update Only	TRUE
Primary Id Field	Account Id
Source Field	
Type Field	
Type Value	
Use Primary Join	TRUE

# Primaries in MVG (5)

## Auto Primary Property

This property determines how the Primary will be assigned when not selected :

- ✓ Default: First child record retrieved will be the Primary
- ✓ Selected: Highlighted record becomes Primary
- ✓ None: User must specify the primary

Properties	
Multi Value Link [Account]	
Alphabetic	Categorized
Auto Primary	Default
Check No Match	FALSE
Comments	
Destination Business Component	Account
Destination Link	Contact/Account
Inactive	FALSE
Module	
Name	Account
No Associate	FALSE
No Copy	FALSE
No Delete	FALSE
No Insert	FALSE
No Update	FALSE
Parent Name	Contact
Popup Update Only	TRUE
Primary Id Field	Account Id
Source Field	
Type Field	
Type Value	
Use Primary Join	TRUE

# Primaries in MVG (6)

## Setting the Primary Record

SSA Primary field is exposed on the MVG applet to make it available for the user to set Primary for the child record

Primary child record is set during the initial loading using EIM

- ✓ If not set, then make the auto primary property to Default.

The screenshot illustrates the configuration and usage of the SSA Primary Field in the MVG applet.

- Properties Dialog:** Shows the "Multi Value Link [Account]" properties. The "Auto Primary" dropdown is highlighted with a purple box, and "Default" is selected.
- List Columns Dialog:** Shows the column configuration for the list view. The "Name" column is selected, and the "Primary" column is highlighted with a purple box, labeled "SSA Primary Field".
- Accounts Dialog:** Shows the list of accounts. A specific account, "Sample Account1", is selected and highlighted with a purple box. A blue arrow points from the "Primary" column in the List Columns dialog to this selected account in the Accounts dialog.



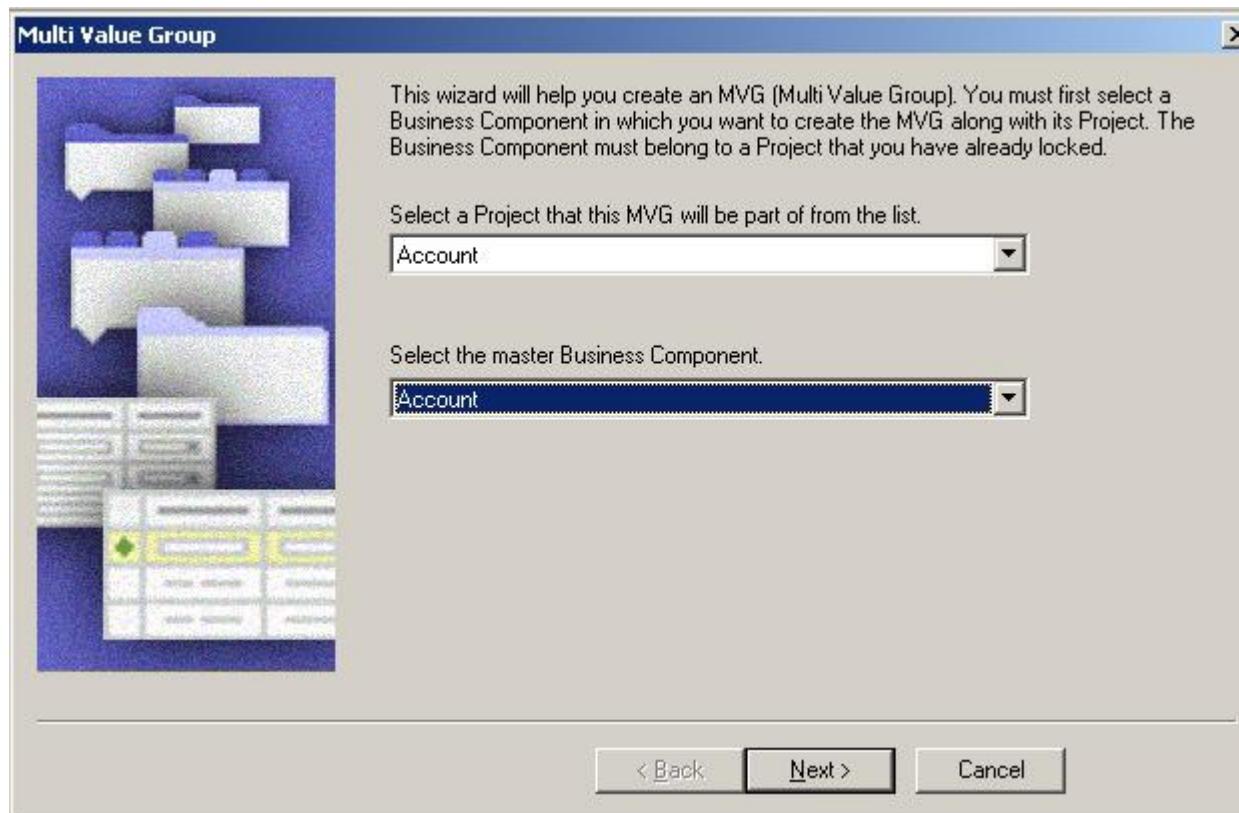
# Topic List

#No	Module Topics
1	Multi Value Group
2	Primaries in MVG
3	Creating MVG

# Creating MVG (1)

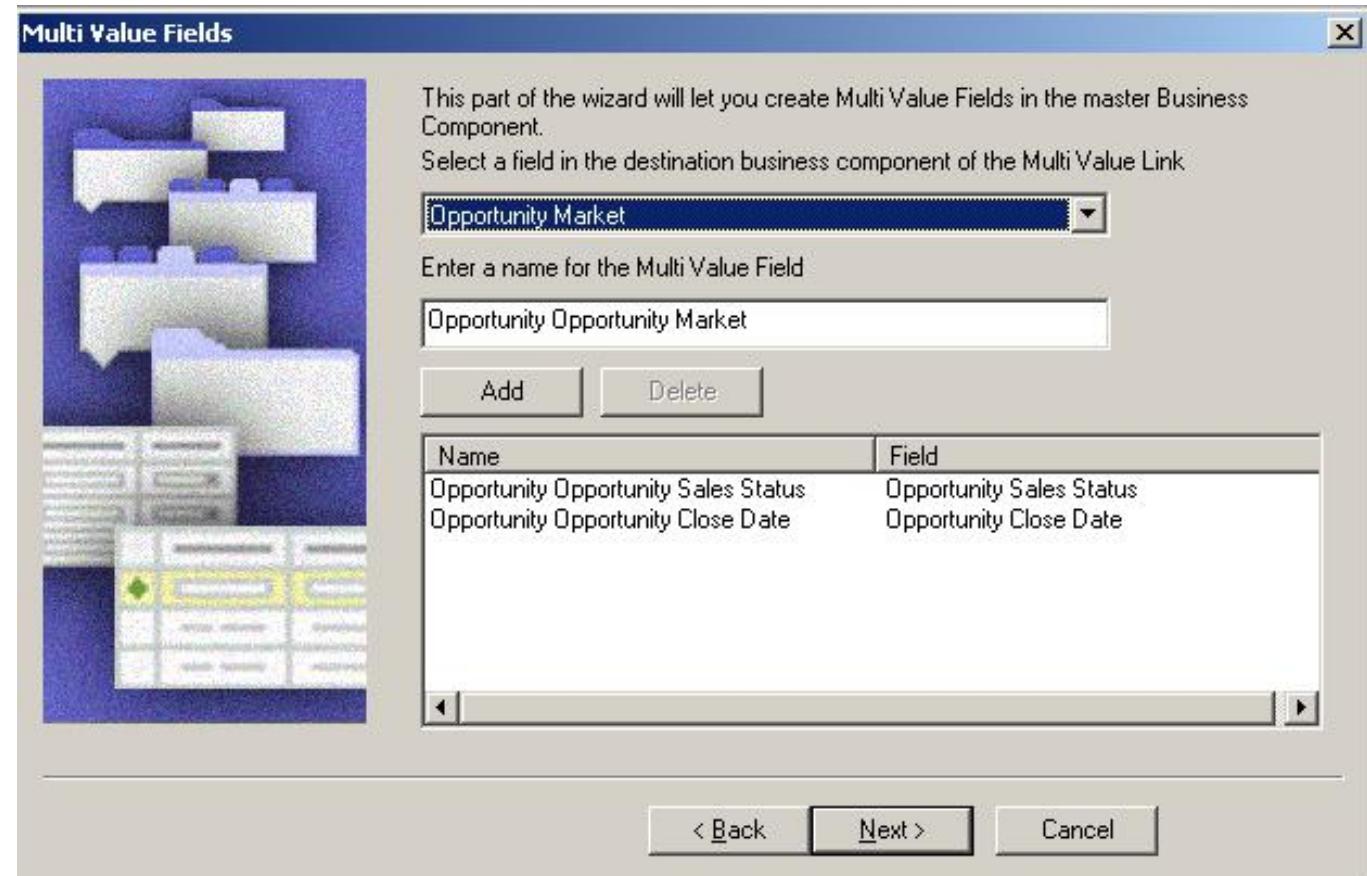
## Use New Object Wizard to Create MVG

Firstly verify if a link exists between parent and child BC



# Creating MVG (2)

- Provide Project
- Select the parent and child BC
- Select the Link that MVF is based on
- The Primary Key Field
- Fields from the child BC



# Creating MVG (3)

MVG Wizard creates:

- MVL
- MVF
- Does not set the Runtime Property or MVG Applet properties on the list columns / controls
  - ✓ Set them manually
- Invokes MVG Applet wizard if no MVG applet exists suiting the selected parent and child BC



# Knowledge Checks

**Answer the following :**

1. Multi Value Field is a field from the Parent BC which is mapped to the Child BC field. State TRUE/FALSE.
2. A Multi Value Group applet shows the parent records in a list applet. State TRUE/FALSE.
3. A Primary allows fast retrieval of a designated child records for display and this record will be called as primary. State TRUE/FALSE.



# Module Summary

**Now, you should be able to:**

- Explain how to:
  - Add Multi Value Fields to BC
  - Create MVG applet to display the child data
  - Create Primary for a MVG



# **Thank You**

# SIEBEL

## Introduction to Business Service

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Describe a business service
- Define the structure and role of the property sets
- Explain how to use business service simulator to test a business service



# Topic List

#No	Module Topics
1	Siebel Business Process Automation
2	Business Service Overview
3	Property Set
4	Business Service Simulator

# Topic List

#No	Module Topics
1	Siebel Business Process Automation
2	Business Service Overview
3	Property Set
4	Business Service Simulator

# **Siebel Business Process Automation (1)**

## **Automating a Business Process**

A Business Process is a sequence of tasks executed to accomplish a definite business objective

- ✓ Eg: Order Management – Creation of quote to order and convert it to Order Submission

Automation options address the following challenges:

- ✓ Standardize and maintain same business process across business units
- ✓ Assigning and routing tasks efficiently
- ✓ Timely response to the Customer's tickets and service requests
- ✓ Assisting Users with business process and best practices



# Siebel Business Process Automation (2)

## Automation Technologies

**Workflow Process:** Is similar to a flowchart which is used to detail out business processes of a company. It is one of the key automation techniques used for building business processes in any Siebel application.

**Workflow Policy:** Defines conditions which when met invokes actions, actions runs the associated workflow process. Workflow Policies is effective for performing simple actions such as sending email, or creating an activity or assignment.

**Task UI:** Enables creation of a UI involving steps, interactive operations along with branching and decision logic to help users through the task flow. We can also pause or resume any task execution if need arises.

**Assignment Manager:** Runs assignment rules to assign data to candidates.

**SmartScript:** SmartScript guides users with suggestions and services built on their profile and purchase history.

**State Model:** Controls transition of status depending on the existing value and user's current position.



# Topic List

#No	Module Topics
1	Siebel Business Process Automation
2	Business Service Overview
3	Property Set
4	Business Service Simulator

# **Business Service Overview (1)**

**Business Service is a business logic which can be reused and accessed throughout the application**

Following are 2 types of business services:

- ✓ Built-in BS which is defined in the Siebel Tools
- ✓ Run-time BS which is defined in the Web Client

You can use a business service as a reusable code library that Siebel CRM can call from another script

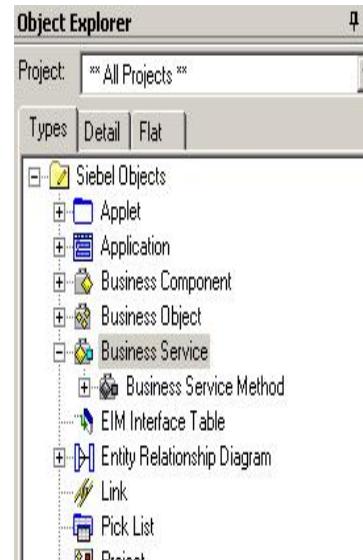
Configuring BS involves properties and VB script or Siebel eScript



# Business Service Overview (2)

## Repository Stored Business Service

- References the predefined CSSService class or a specialized class
- These can be custom or Pre built-in BS
- Many Pre built-in Vanilla BS are available to be used for specific operations
  - ✓ Customer Order Management , Enterprise Application Integration etc.
  - ✓ Editing/Modifying is not allowed for these objects
  - ✓ Written in C++
- Custom Business service are written in Siebel VB Script or Siebel eScript
  - ✓ Needs to be compiled to SRF
    - When ever the BS Is modified, these must be compiled into SRF
- These are stored in the repository



The screenshot shows the Siebel Object Explorer interface. The 'Project' dropdown is set to 'All Projects'. The 'Types' tab is selected, showing a tree view of Siebel objects. The 'Business Services' node is expanded, revealing its sub-components: Business Service Method, EIM Interface Table, Entity Relationship Diagram, Link, Pick List, and Project.

W	Name	Changed	Project	Cache	Class	Display Name
>	ISS ATP Service		ISS Order Management	✓	CSSISSTfulfillmentService	ISS ATP Service
	ISS Add Attachment Service		ISS Order Management		CSSISSAddAttachmentService	Add Attachment
	ISS Approval Bus Service		ISS Approval		CSSISApprUIService	ISS Approval
	ISS Authoring Import Export Service		ISS Authoring Admin		CSSISSAuthImportSvc	ISS Authoring
	ISS Compatibility Multi-Popup Service		Eligibility Compatibility	✓	CSSISMultiPopupService	Compatibility
	ISS Context Import Export Service		ISS Context		CSSISSCtxObjImportSvc	ISS Context Import
	ISS Context Service		ISS Context	✓	CSSISSContextService	Context Service
	ISS Copy Service		ISS Context		CSSISSCopyService	ISS Copy Service

# Business Service Overview (3)

## Client Stored Business Service

These are stored in run-time database in addition to customer data

Administration – Business Service Screen enables writing , editing and modify client BS

- ✓ Written using Siebel VB or Siebel eScript

It is never executed if a BS with same name exists in the repository

It is not compiled to SRF

- ✓ Hence directly available for Applications

We can move the BS from repository to client and vice-versa

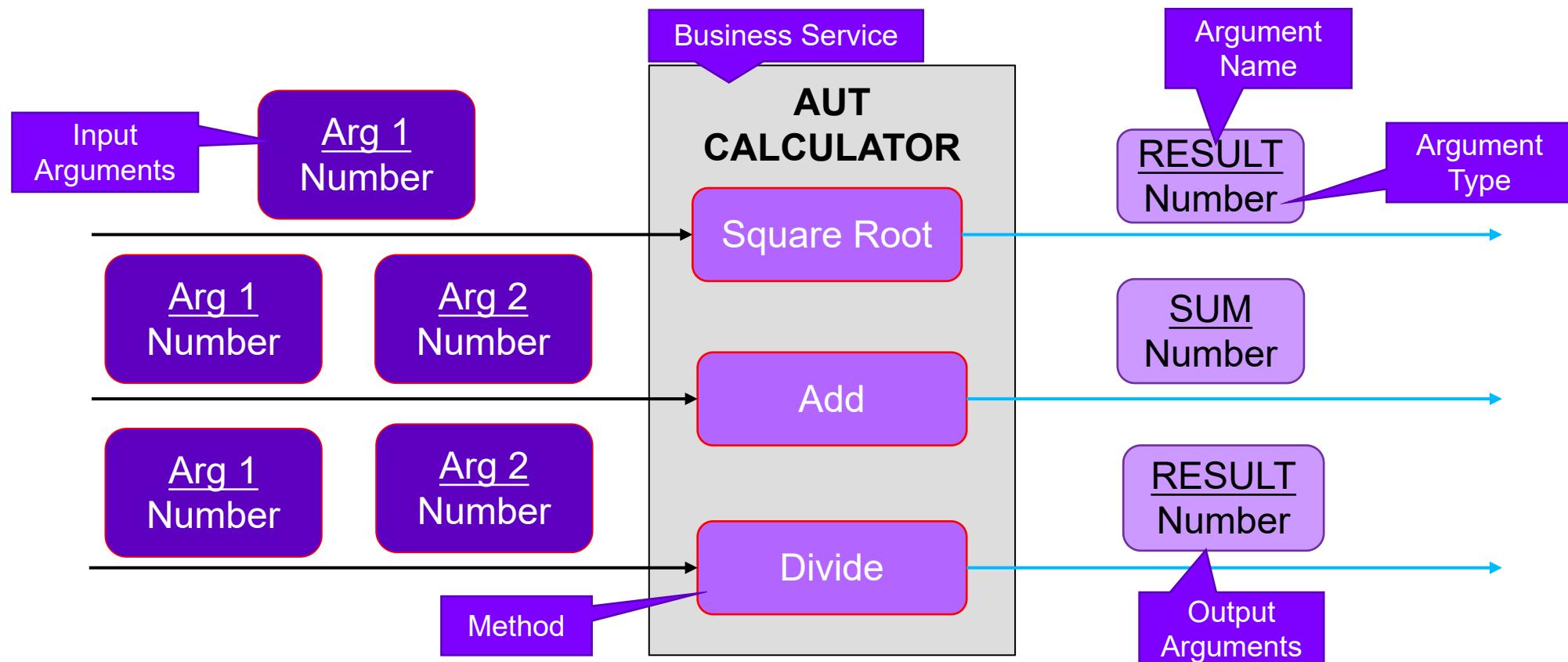
Query Results			
Name	Inactive	State Management Type	Comments
A Fee Dispute Script	Inactive	Stateful	Business service used by FINS Service to Sales demo - Daniel Matica.
CDAToAnything	Inactive	Stateful	CDA (eAdvisor): Transfers data from CDA to Buscomps as defined in a mapping feature table
COMSignalModifyService	Inactive	Stateful	This service modifies signal admin settings [HMZ 03/25/05]

# Business Service Overview (4)

## Methods

Business Service contains multiple operations called Methods

Each method has set of input and output arguments of specific type



# Business Service Overview (5)

## Methods

To identify the methods of a business service:

- ✓ Open Siebel tools, navigate to business service in object explorer
- ✓ Select business service methods child object definition

The screenshot shows the Siebel Object Explorer and a Business Service Method List window.

In the Object Explorer (left), under the 'Business Service' node, the 'Business Service Method' child node is selected and highlighted with a purple box. A purple callout points from this node to the text 'Business Service Methods'.

The Business Service Method List window (right) displays a table of methods:

Name	Changed	Project	Display Name	Display Name - String Reference	Display Name - String Override
AUT Calculator	✓	Account	AUT Calculator	AUT Calculator	AUT Calculator
Add	✓		Add	Add	Add
Subtract	✓		Subtract	Subtract	Subtract
Multiply	✓		Multiply	Multiply	Multiply
Divide	✓		Divide	Divide	Divide
Square Root	✓		Sqaure Root	Sqaure Root	Sqaure Root

A purple callout points from the 'Display Name' column of the first method row to the text 'Name that appears in Client while selecting from the drop down on Business Service Method'.



# Business Service Overview (6)

## Arguments and Types for Method

In Siebel tools, Goto Business Service → Business Service Method → Business Service Method Arg

- ✓ All Input and Output arguments and their type are available

The screenshot shows the Siebel Object Explorer interface. On the left, under the 'Siebel Objects' section, 'Business Service' is expanded, and 'Business Service Method' is selected. Under 'Business Service Method', 'Business Service Method Arg' is also selected. The main window displays two tables: 'Business Service Methods' and 'Business Service Method Arguments'. The 'Business Service Methods' table lists four methods: Subtract, Multiply, Divide, and Sqaure Root. The 'Business Service Method Arguments' table shows the arguments for the 'Divide' method. A purple callout box points to the 'Result' argument in this table, which is highlighted with a blue border. The 'Result' argument is defined as a 'Number' type, 'Output', 'Optional', and 'Storage Type' as 'Property'. Other arguments listed are 'Arg2' and 'Arg1', both of which are 'Input' types.

W	Name	Changed	Display Name	Display Name - String Reference	Display Name - String Override
	Subtract	✓	Subtract		Subtract
	Multiply	✓	Multiply		Multiply
>	Divide	✓	Divide		Divide
	Sqaure Root	✓	Sqaure Root		Sqaure Root

W	Name	Data Type	Type	Optional	Storage Type
>	Result	Number	Output		Property
	Arg2	Number	Input		Property
	Arg1	Number	Input		Property

Arguments for Divide Method

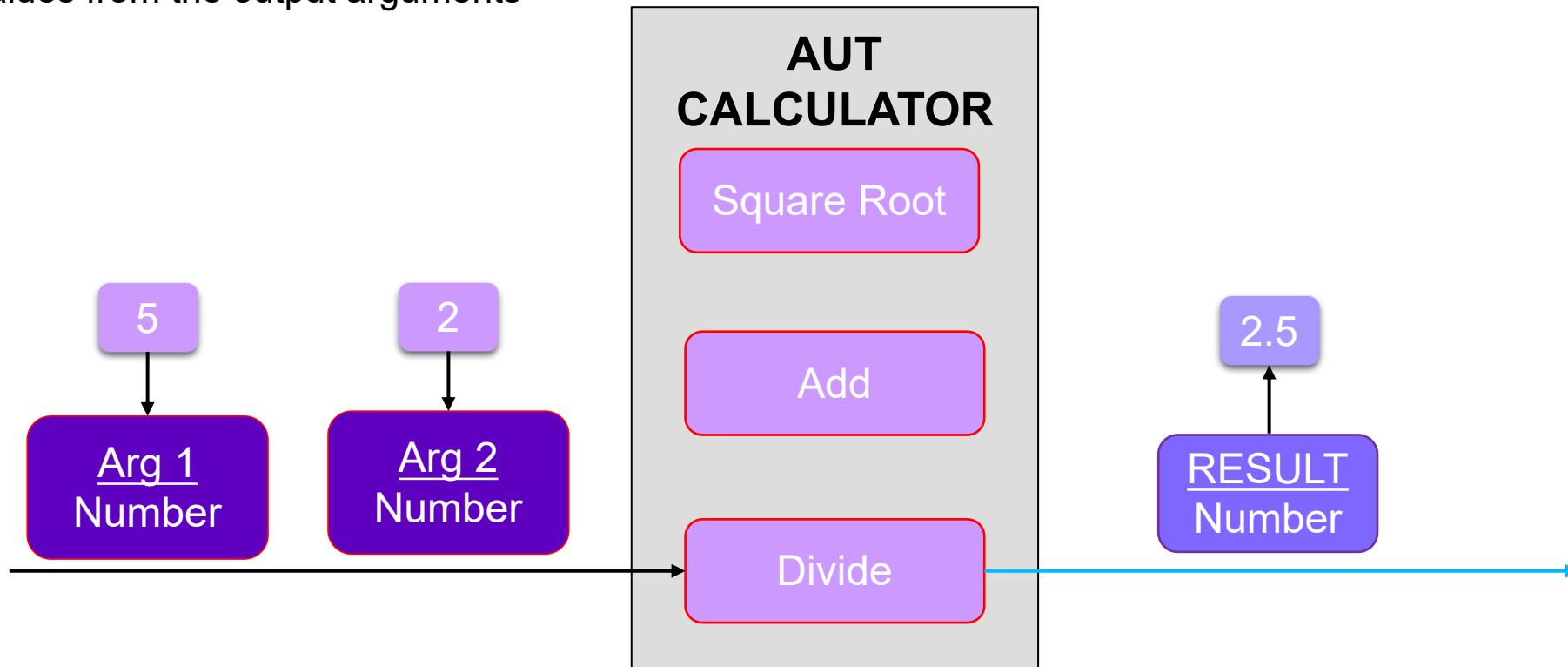


# Business Service Overview (7)

## Invoking a Method

Invoking method involves:

- ✓ Assigning values to the input arguments
  - Not all input arguments are mandatory
- ✓ Retrieve values from the output arguments



# Topic List

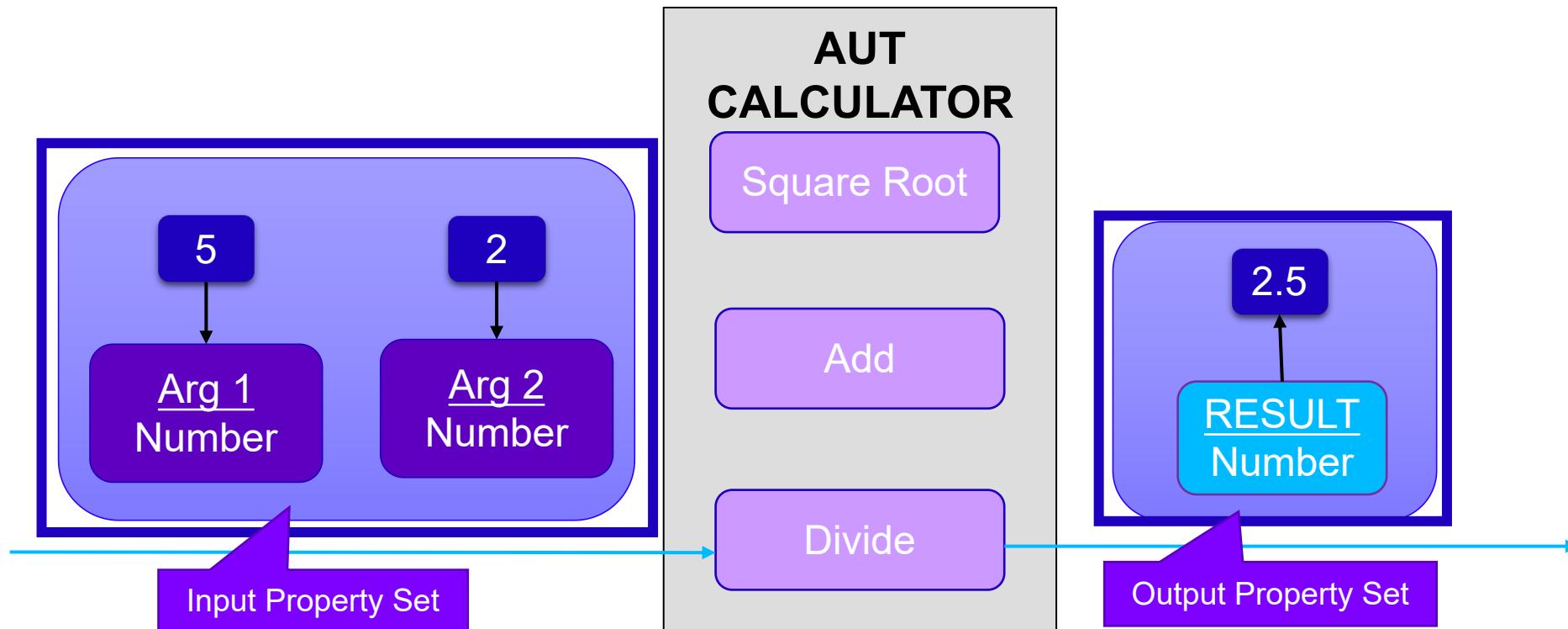
#No	Module Topics
1	Siebel Business Process Automation
2	Business Service Overview
3	Property Set
4	Business Service Simulator

# Property Set (1)

## What is Property Set?

Property set is a in-memory data structure used to:

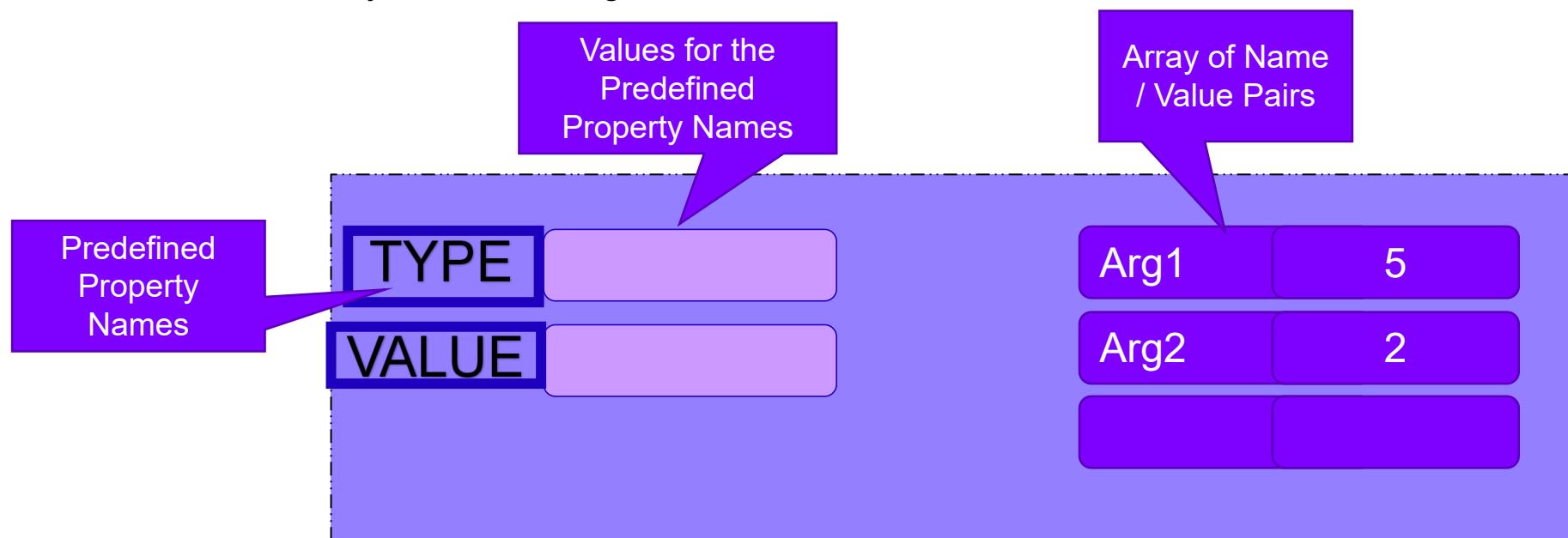
- Pass set of input arguments to the method to be invoked
- Receives set of output arguments from the method



# Property Set (2)

A property set:

- Represents data using name/value pairs
- Has two defined properties → **Type** and **Value**
- Has an array for storing name/value pairs
- Is created automatically while invoking a BS



# Topic List

#No	Module Topics
1	Siebel Business Process Automation
2	Business Service Overview
3	Property Set
4	Business Service Simulator

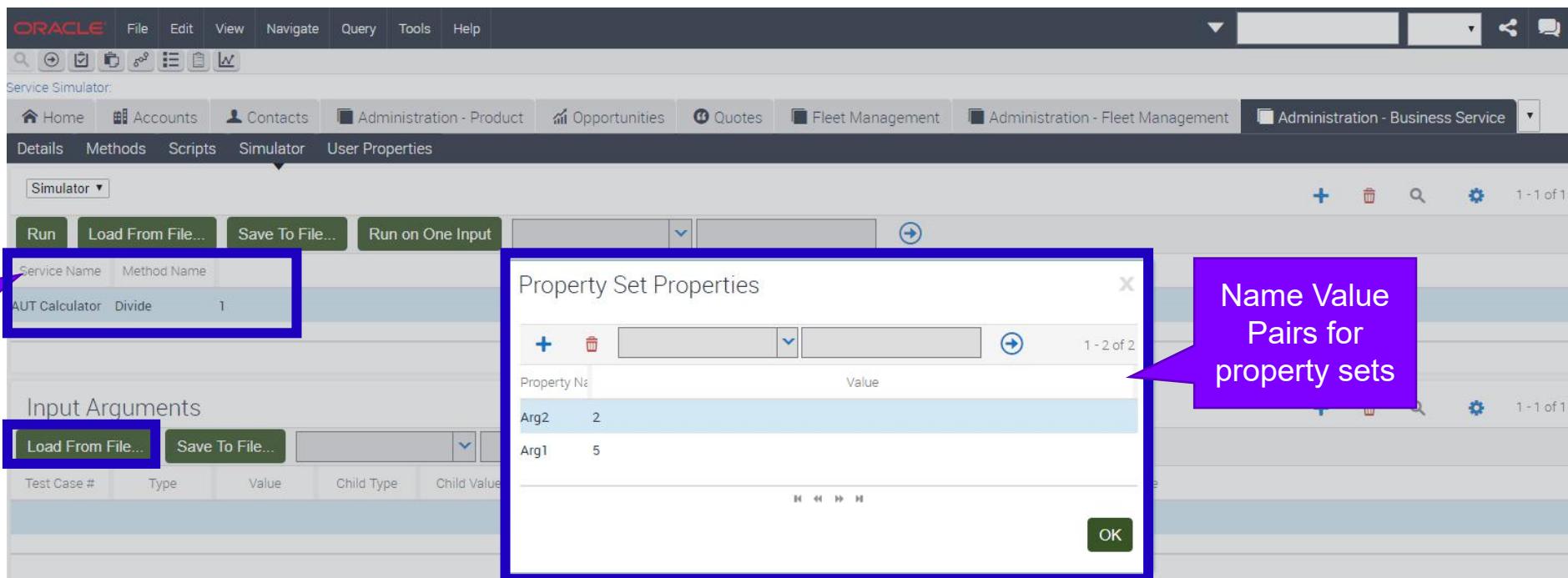
# Business Service Simulator (1)

Navigate Administration Business Service → Simulator

Select the Business Service Name and Method

Specify the Input Arguments through Property Sets

- ✓ Name/Value pairs are provided
- ✓ Also can be loaded using Input File



# Business Service Simulator (2)

To simulate the BS:

Click run on one input

Output arguments will display the resultant name/value pair

This can be saved to a file.

The screenshot shows the Oracle Business Service Simulator interface. At the top, there's a navigation bar with links like Home, Accounts, Contacts, etc. Below it is a toolbar with buttons for Run, Load From File..., Save To File..., and Run on One Input (which is highlighted with a blue box). The main area has tabs for Details, Methods, Scripts, Simulator (which is selected), and User Properties. Under the Simulator tab, there's a section for AUT Calculator Divide with an iteration count of 1. Below this are two tables: 'Input Arguments' and 'Output Arguments'. The 'Input Arguments' table has a single row with Test Case # 1, Type Arg2, Value 2. The 'Output Arguments' table is currently empty, indicated by a 'No Records' message. A large purple callout points from the bottom center towards the 'Output Arguments' table.

# Knowledge Checks

## Answer the following:

1. Business Process is a sequence of tasks executed to accomplish a definite business objective.

State TRUE/FALSE.

2. Identify the correct statement regarding property sets:

- A. Represents data using name/value pairs
- B. Consists of one or more operations called methods
- C. Has specified set of input and output arguments

3. Steps to test a BS using simulator are provided. Arrange them in the right sequence.

- A. Examine the output set name/value pairs
- B. Click the run on one input button
- C. Select a business service name and business service method
- D. Create the input property set with name and value pairs



# Module Summary

**Now, you should be able to:**

- Describe a business service
- Define the structure and role of the property sets
- Explain how to use business service simulator to test a business service



# **Thank You**

# SIEBEL

Develop Workflow Process

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- List the types of workflow process and workflow steps
- Explain how to:
  - Create a new WF to configure a BS, Siebel operation and decision steps
  - Create and modify workflow processes



# Topic List

#No	Module Topics
1	Workflow Process Overview
2	Siebel Workflow Configuration

# Topic List

#No	Module Topics
1	Workflow Process Overview
2	Siebel Workflow Configuration

# Workflow Process Overview (1)

- Siebel Workflow Process is a tailor-made business automation application that defines, manages, and enforces business processes.
- Workflow Process is similar to a flowchart which has series of steps detailing out business processes of a company
- Specifies input and output properties for every step, also for the complete WF Process.
- WF process could be :
  - ✓ Simple, E.g., Adding products to your cart
  - ✓ Complex, E.g., handling entire Order Management
- Workflow processes can manipulate data, can check for conditions, also can invoke a business service, task UI, or subprocess

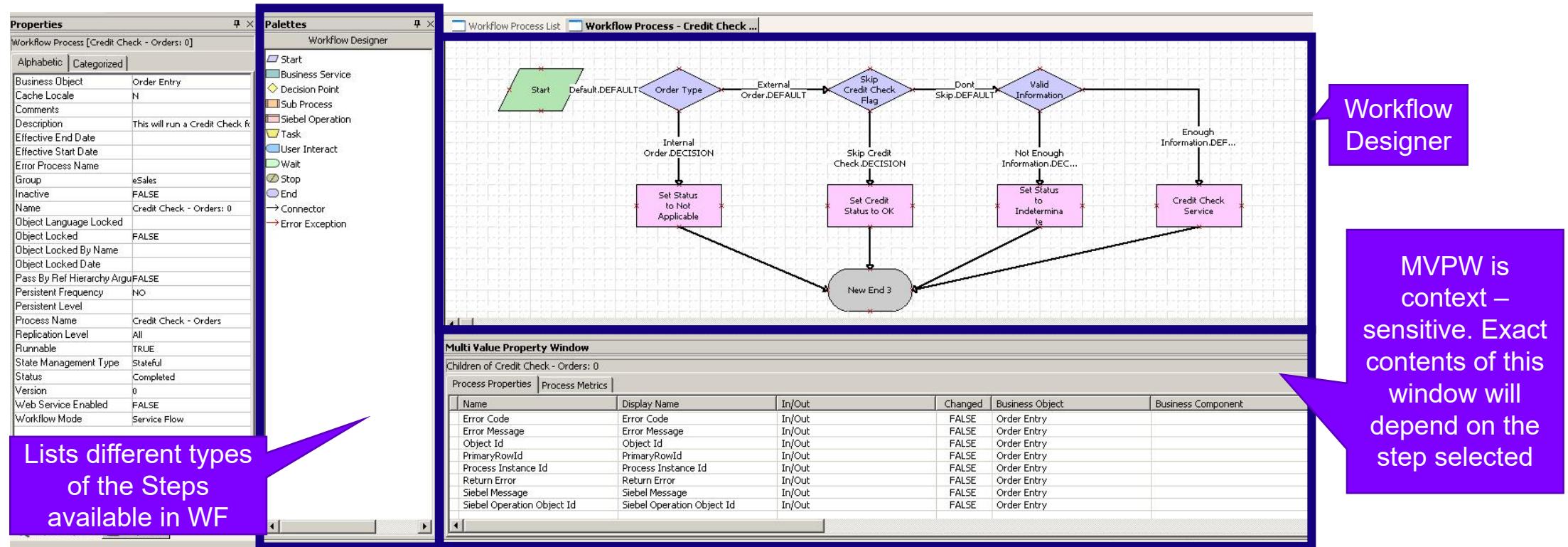
# Workflow Process Overview (2)

## Workflow Designer

Workflow designer in Siebel tools is used to create, examine, modify workflow processes

Select workflow process from object explorer.

- Choose any existing WF
- Right-click → Edit Workflow Process → Opens Workflow Designer



# Workflow Process Overview (3)

## Common Workflow Steps

Every Workflows must have these steps:

- ✓ **Start** – WF always starts with this step
- ✓ **End** – WF always ends with this step

Workflows can also include these based on the operation to be performed:

- ✓ **Business Service**
- ✓ **Siebel Operation**
- ✓ **Decision Point**



# Workflow Process Overview (4)

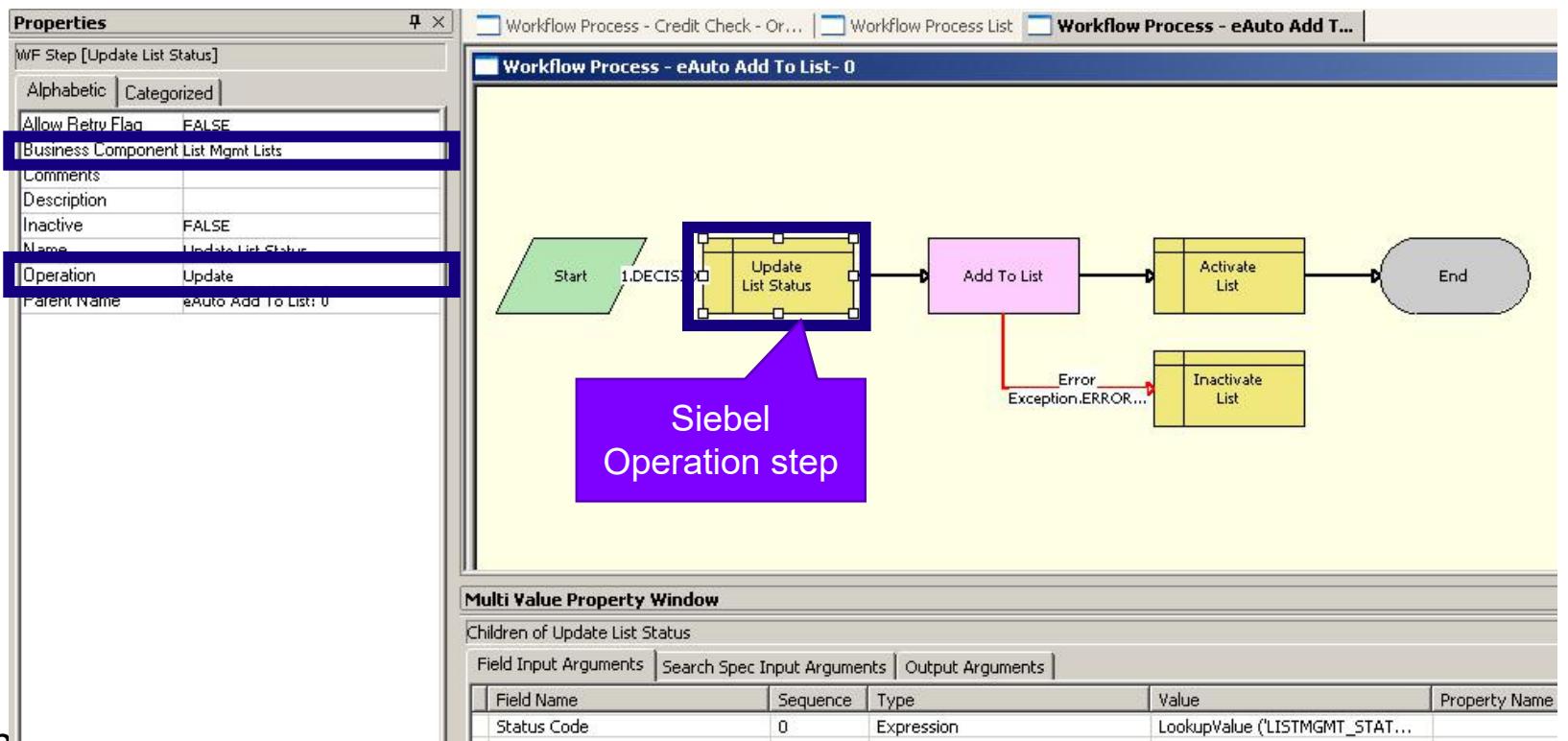
## Siebel Operation Step

This can perform database operations such as:

- ✓ Query
- ✓ Insert
- ✓ Delete
- ✓ Update
- ✓ NextRecord
- ✓ Upsert
- ✓ PreviousRecord
- ✓ QueryBiDirectional

Always refers to one business component.

Search specification property can be used to locate the records to modify

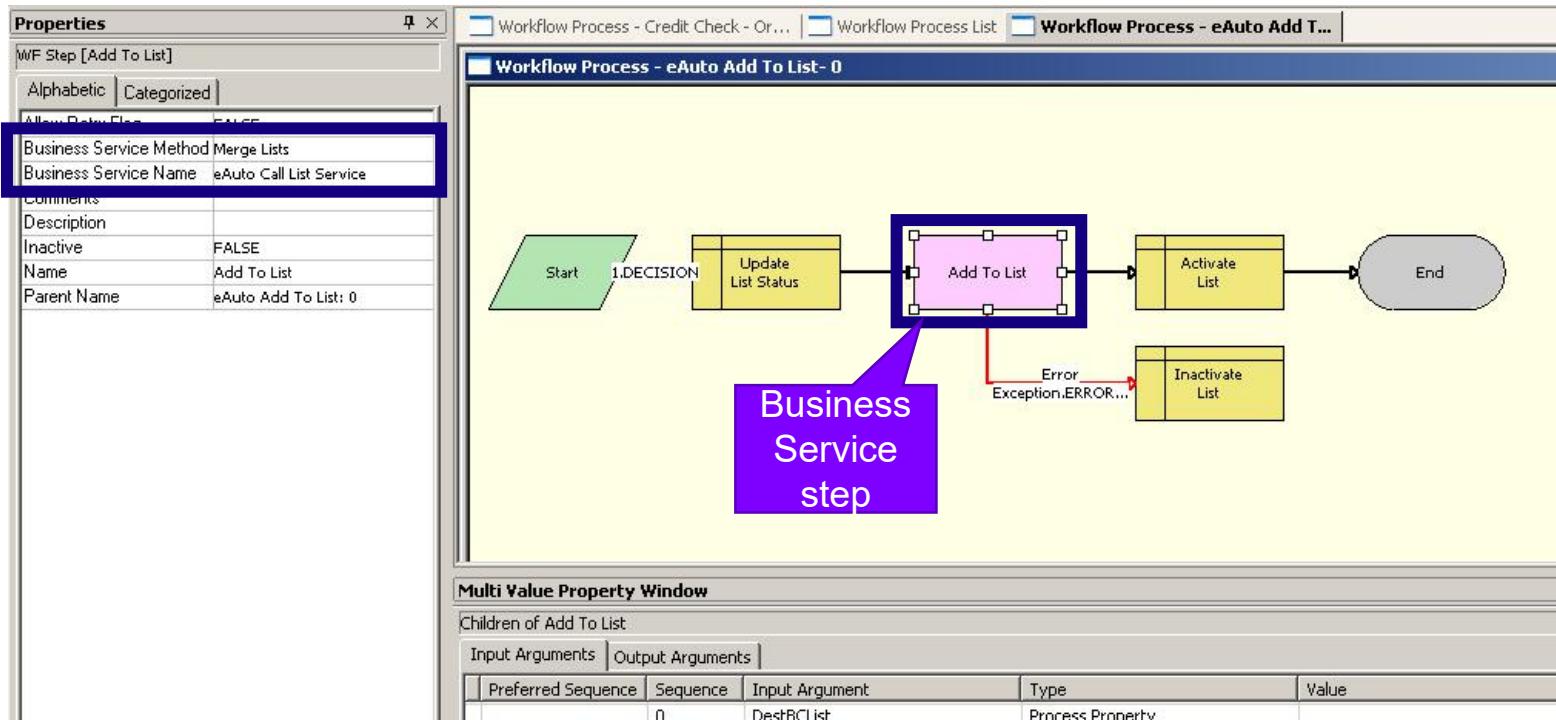


# Workflow Process Overview (5)

## Business Service Step

It invokes a method of the business service

It can invoke both custom and vanilla BS



# Workflow Process Overview (6)

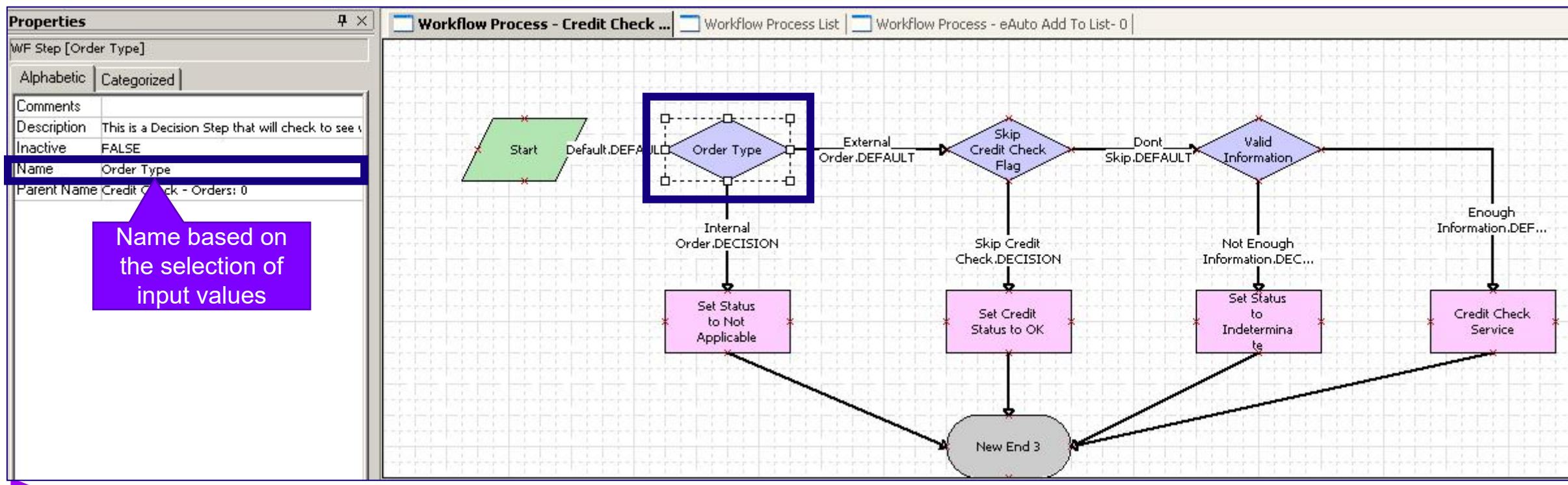
## Decision Point Step

It enables branching to one of the multiple steps based on the Input values

Each Decision step can have multiple branches based on the value

- ✓ Here, only two branches are present – default branch and conditional branch

The decision is made based on the contents of the branch. Right-click on branch > Edit Conditions to evaluate the condition



# Workflow Process Overview (7)

## Process Properties

Process Properties are used to store Inputs used by and Outputs produced by the workflow steps

Multi Value Property Window shows Process Properties and Process Metrics that are global to the WF

- ✓ Click on the white space in the workspace to see the window

Upon New WF Creation, Set of Default Process Properties are created

- ✓ Contain information regarding Current Object in process, error information, current operation steps being performed etc.

Custom process properties can also be created

The screenshot shows the 'Multi Value Property Window' interface. At the top, there's a title bar with the window name and tabs for 'Process Properties' and 'Process Metrics'. Below the tabs is a table with columns for Name, Display Name, In/Out, Changed, Business Object, Default String, Default Date, Data Type, Default Number, and Integration Object. The 'Process Properties' tab is selected, displaying a list of default properties. A purple callout box labeled 'Default Process Properties' points to the first three rows of the table, which are highlighted with a blue border. Another purple callout box labeled 'Custom Process Property' points to the last two rows of the table, which are highlighted with a red border. The table contains the following data:

Name	Display Name	In/Out	Changed	Business Object	V	Default String	Default Date	Data Type	Default Number	Integration Object
Error Code	Error Code	In/Out	FALSE	Order Entry				String		
Error Message	Error Message	In/Out	FALSE	Order Entry				String		
Object Id	Object Id	In/Out	FALSE	Order Entry	10-5FG3XQ			String		
PrimaryRowId	PrimaryRowId	In/Out	FALSE	Order Entry				String		
Process Instance Id	Process Instance Id	In/Out	FALSE	Order Entry				String		
Return Error	Return Error	In/Out	FALSE	Order Entry	Y			String		
Siebel Message	Siebel Message	In/Out	FALSE	Order Entry				Hierarchy		
Siebel Operation Object Id	Siebel Operation Object Id	In/Out	FALSE	Order Entry				String		

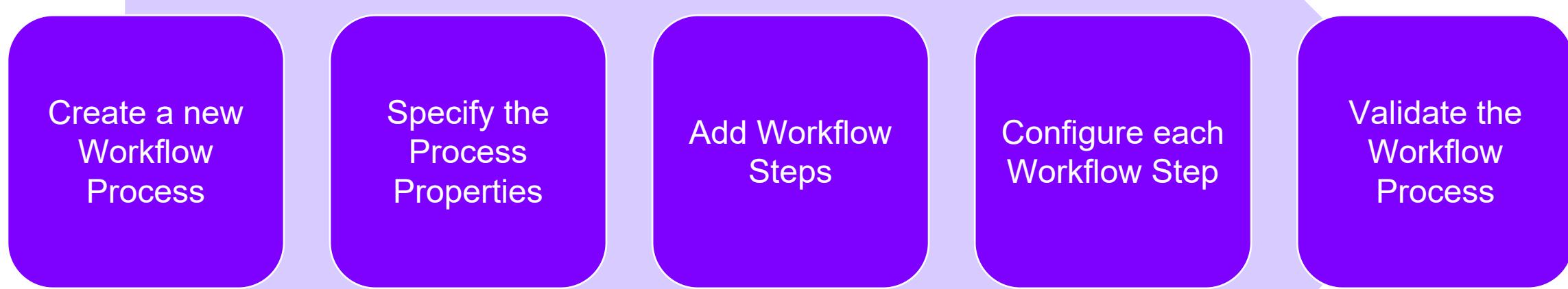


# Topic List

#No	Module Topics
1	Workflow Process Overview
2	Siebel Workflow Configuration

# Siebel Workflow Configuration (1)

Steps to configure a WF using workflow designer:



# Siebel Workflow Configuration (2)

To create a new workflow process:

Inside Siebel Tools, select the Workflow Process Object Type

Right-click → New Record. Create a new WF Process Object Definition

- ✓ Provide a Name to the WF
- ✓ Assign a Project
- ✓ Select a Business Object to the WF

➤ Provides Context for BC and Fields

Status of the WF is In Progress

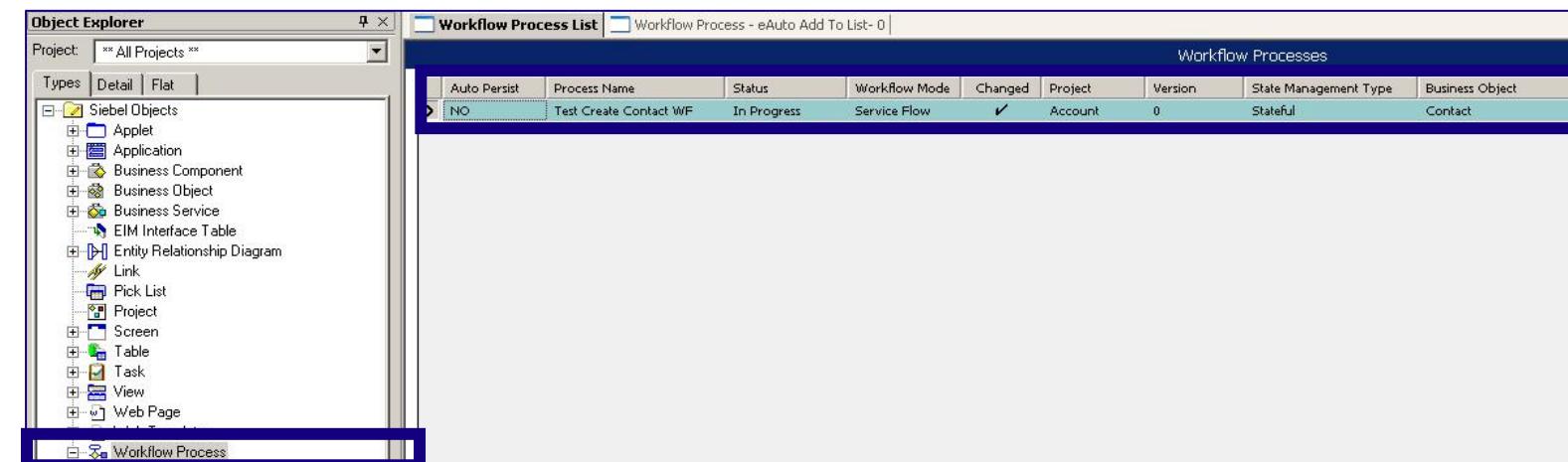
- ✓ Changes after deploying the WF

Right-click → Edit Workflow Process

- ✓ To invoke the workflow designer

Version of the WF is Zero (0)

- ✓ Version changes when we revise the WF



# Siebel Workflow Configuration (3)

## Workflow Modes

Every WF has a runtime behaviour described by the Workflow Mode

Following are available WF Modes:

### Service Flow:

A WF in this mode executes multiple steps to complete a task from start to end

- ✓ This workflow completes the task in short span of time without stopping or pausing for any event or activity

Can be invoked as a sub process from another service flow WF, an interactive flow WF, or a long-running flow WF, but not from a 7.0 flow WF

Following are the limitations:

- ✓ Never waits for any run-time event
- ✓ Never use wait or user interact step

### Interactive Flow:

- A WF in this mode assists users in navigating across Siebel application
- Includes a run-time event and one or more user interact steps



# Siebel Workflow Configuration (4)

## Workflow Modes

### Long-Running Flow:

- A WF in this mode can last for hours, days, or months
- Not allowed to use user interact or wait step
- We cannot simulate a long-running workflow using Tools simulator
- can be paused and resumed as an inbox item

### 7.0 Flow:

- These WF provides backward compatibility for an existing workflow process
- New WF should not be created with 7.0 Flow workflow mode



# Siebel Workflow Configuration (5)

## Specify the Process Properties

Select the Process Property Tab in MVPW Window to display the default process properties

- ✓ Never change the default process properties

Create additional Process Properties required to store the values and used by WF Steps

Multi Value Property Window												
Children of Test Create Contact WF: 0												
Process Properties		Process Metrics										
Name	Display Name	In/Out	Changed	Business Object	Business Component	Virtual Field	Default String	Default Date	Data Type	Default Number	Integration Object	Correlator Flag
First Name		In/Out	TRUE						String			FALSE
Error Code		In/Out	TRUE						String			FALSE
Error Message		In/Out	TRUE						String			FALSE
Object Id		In/Out	TRUE						String			FALSE
Process Instance Id		In/Out	TRUE						String			FALSE
Siebel Operation Object Id		In/Out	TRUE						String			FALSE

Default Process Properties



# Siebel Workflow Configuration (6)

## Adding Workflow Steps

Add a Start and End Steps into the Workspace

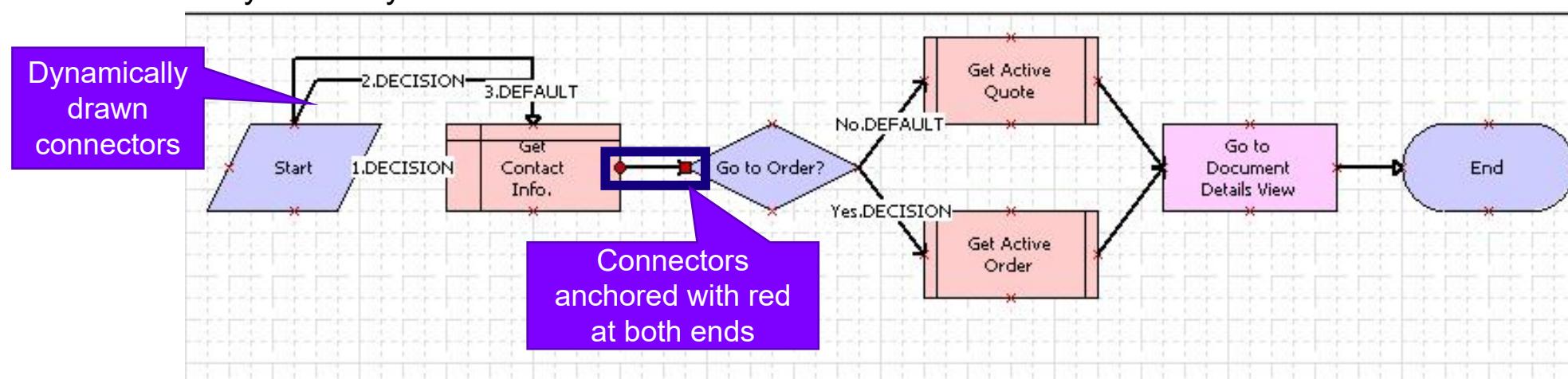
- ✓ Drag the steps from the palette to the Workspace

Insert other steps as and when required

- ✓ Based on the requirement

Insert connectors to connect the steps

- ✓ Make sure connectors are anchored and red at both ends when connected between steps
- ✓ If not, then it shall throw an error while validating the WF
- ✓ Connectors are dynamically drawn



# Siebel Workflow Configuration (7)

## Configuring Siebel Operation Step

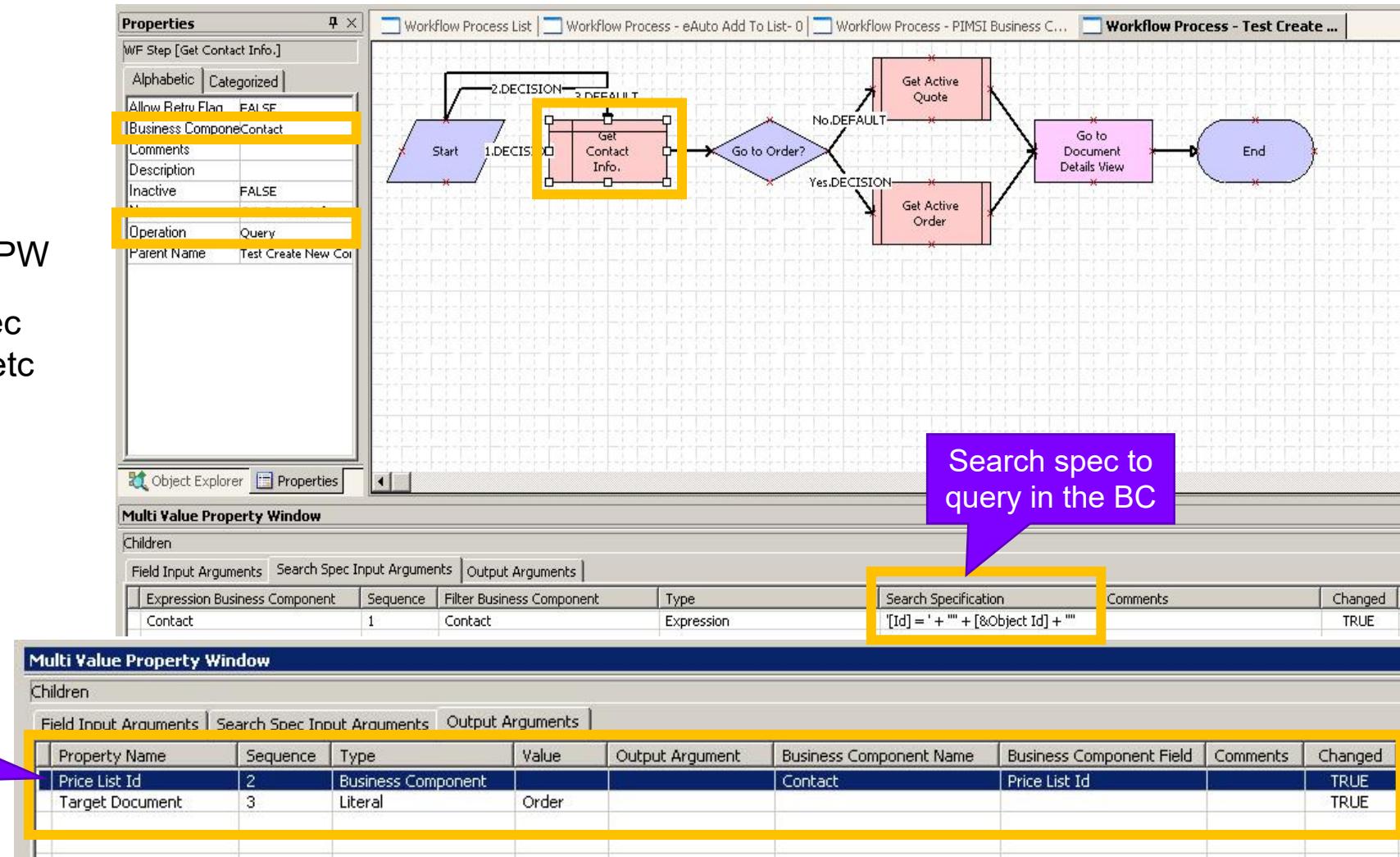
For each Siebel Operation step :

Specific Business Component and Operation

- ✓ Use properties window

Specific other child entities in the MVPW

- ✓ Field Input Arguments, Search Spec Input Arguments, Output Arguments etc

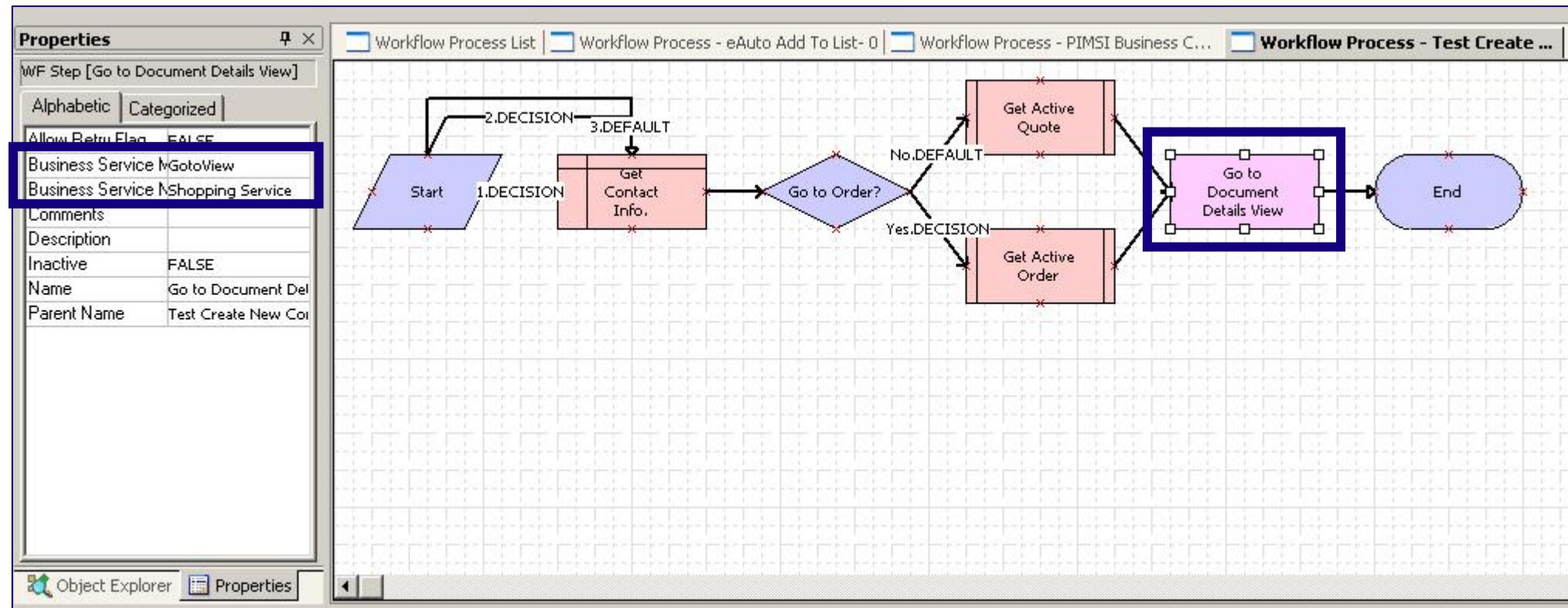


# Siebel Workflow Configuration (8)

## Configuring the Business Service Step

Specify the Business Service Name and Business Service Method

- ✓ Use the properties Window



# Siebel Workflow Configuration (9)

## Configuring the Business Service Step

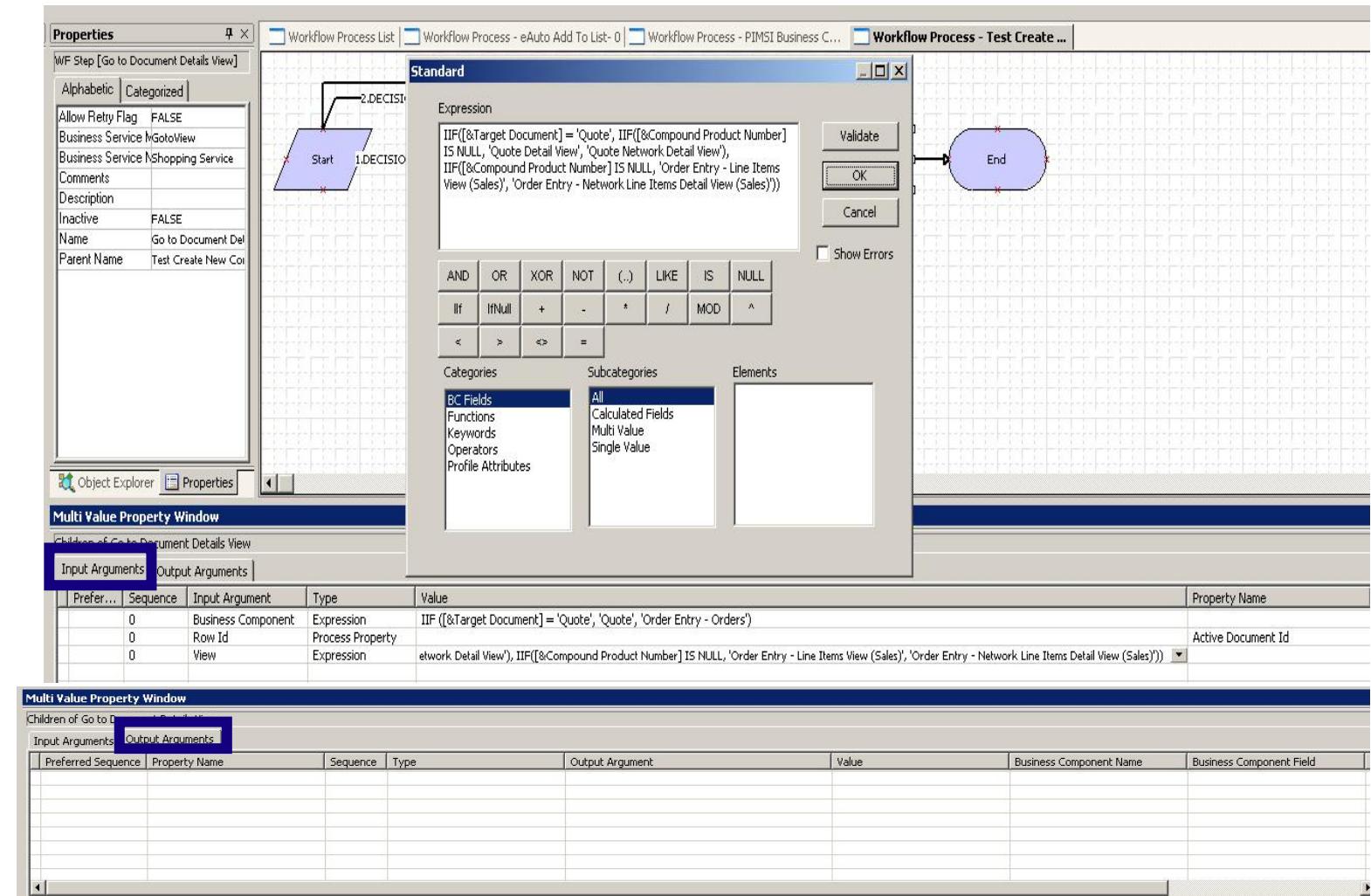
For each Business Service Step :

Specify the inputs to be used by the WF

- ✓ Select the Input Arguments tab in the MVPW
- ✓ Assign a Literal Value / Expression / Process Property to each of the Input arguments

Specify the outputs of the step

- ✓ Select the Output Arguments tab in MVPW
- ✓ Not always mandatory to get outputs
- ✓ If any outputs exists , these should be mapped to process properties

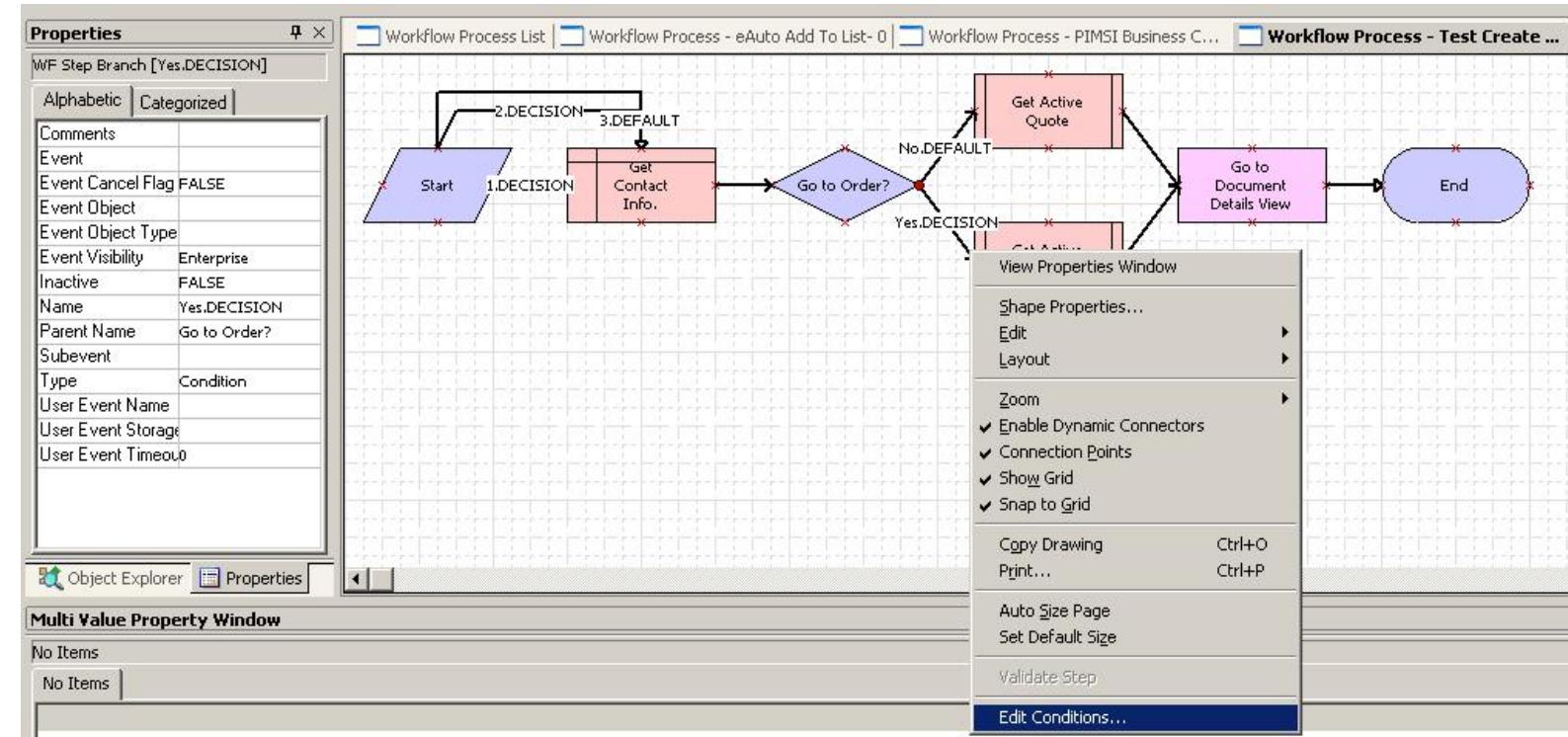


# Siebel Workflow Configuration (10)

## Configuring Decision Point Step

For each branch originating from the Decision Step set the conditions

- ✓ Select the connector
- ✓ Right-click → Edit conditions
- ✓ Every Decision step must have a default connector
- ✓ Every condition connector will have a conditional criteria
  - Can contain fields and Properties compared to literals or Expressions



# Siebel Workflow Configuration (11)

## Configuring Decision Point Step (2)

Using Composer Condition Criteria box the conditions for branch are specified

- ✓ Never create conditions on default branch
  - This is the path selected when none of the conditions satisfies

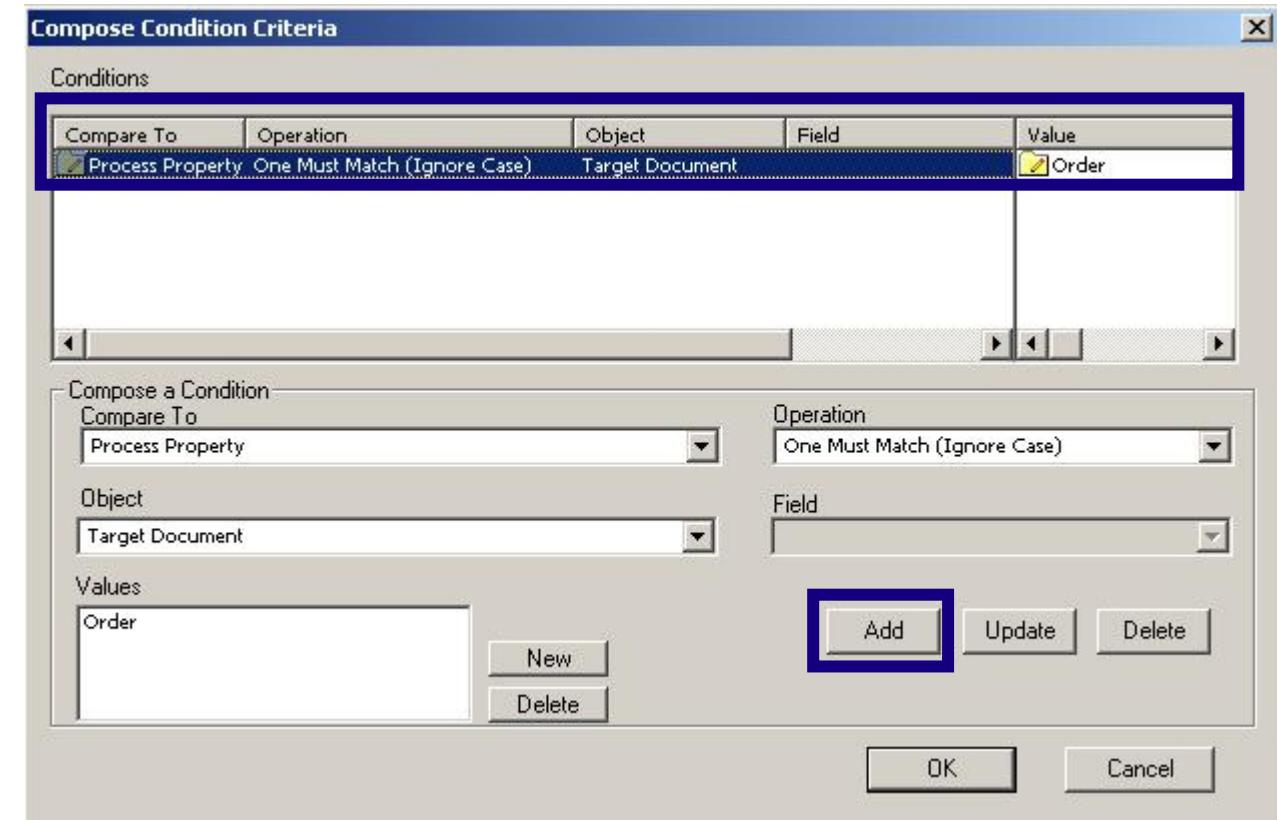
Conditions can be compared to Applet , BC , Process Property or Expression

- ✓ Here it is compared to Process Property

In Object, Choose the desired property

On Click of Add Button, Conditions get added in the above pane.

Multiple conditions can also be added



# Siebel Workflow Configuration (12)

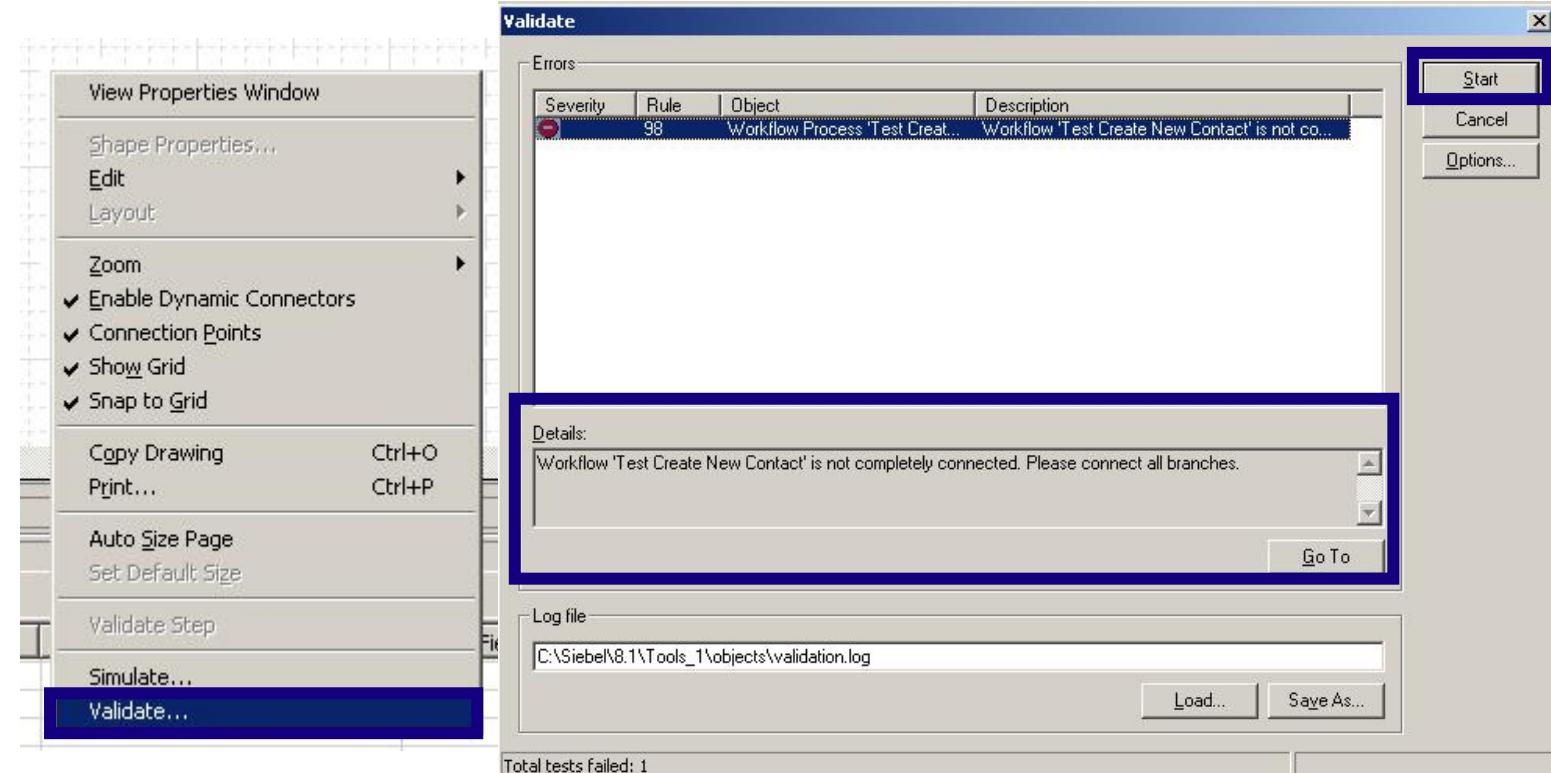
## Validate Workflow Process

Validation is done to check various syntactic errors that may occur in WF

To validate any WF:

- ✓ First save the configuration done to WF
- ✓ Right-click on the designer > Select Validate
- ✓ Validate Windows pops out
- ✓ Click on Start button
- ✓ Check for errors and correct them

Validation doesn't ensure all the errors to be caught, some may arise during runtime



# Siebel Workflow Configuration (13)

## Additional Workflow Steps

### SUB PROCESS:

- ✓ Invokes another WF as a subprocess

### USER INTERACT:

- ✓ Navigates the user to a different view and waits for user activity

### WAIT:

- ✓ Pauses the WF for specific time interval before proceeding

### STOP:

- ✓ Stops execution of the WF when a specific exception occurs and terminates WF in error state

### TASK:

- ✓ Invokes a Siebel TASK UI



# Knowledge Checks

## Answer the following

1. Identify the incorrect statement from below about Workflows :
  - A. Workflows are used to automate some parts of the Business Process
  - B. Process Properties provide input to the WF Steps
  - C. WF are created in Siebel client
  - D. Workflow mode describes the runtime behavior
2. Process Properties are variables that used to store the inputs to WF steps and outputs from the WF Steps. State TRUE/FALSE.
3. Below are the steps to configure a WF Process. Arrange them in the right sequence.
  - A. Create a New WF Process
  - B. Configure WF Steps
  - C. Validate WF Process
  - D. Add Workflow Steps
  - E. Specify Process Properties



# Module Summary

**Now, you should be able to:**

- List the types of workflow process and workflow steps
- Explain how to:
  - Create a new WF to configure a BS, Siebel operation and decision steps
  - Create and modify workflow processes



# **Thank You**

SIEBEL

Test and Deploy Workflow Process

accenture >

# Module Objectives

**At the end of this module, you should be able to:**

- Explain how to:
  - Test a workflow process using workflow simulator
  - Deploy a Siebel WF Process



# Topic List

#No	Module Topics
1	Testing Workflow Process
2	Deploying Workflow Process

# Topic List

#No	Module Topics
1	Testing Workflow Process
2	Deploying Workflow Process

# **Testing Workflow Process (1)**

## **Managing Workflow Process**

WF Process differ from other object definitions :

- ✓ Though created in Siebel Tools, They are not compiled to SRF
- ✓ These can be migrated using import and export of WF in form of XML files

After configuring the WF in Tool

- ✓ Simulate the Workflow
- ✓ Deploy the Workflow



# Testing Workflow Process (2)

## Workflow Simulator

The Process Simulator is a simulation tool that allows to step through a workflow process

- ✓ Results of each step can also be viewed

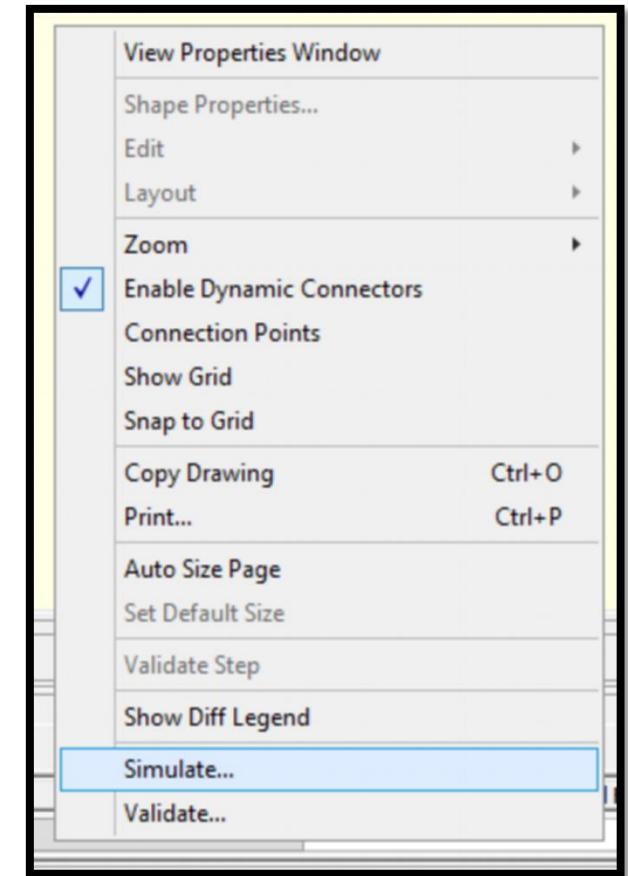
Simulating a workflow process before deployment to production environment helps :

- ✓ Verify if it works correctly
- ✓ The results meet your business requirements

Workflow Simulation is controlled in Siebel Tools

To enable the Simulator :

- ✓ Right-click on the Designer → Select Simulate



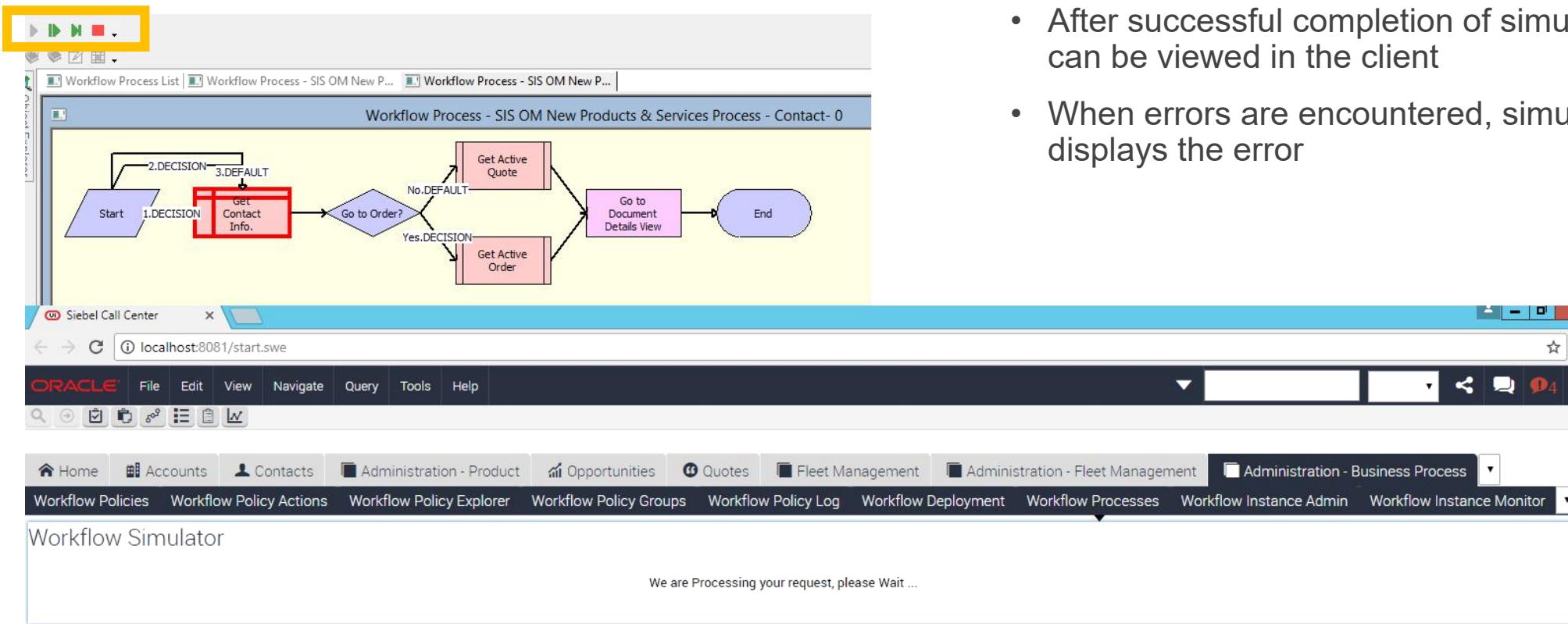
# Testing Workflow Process (3)

## Workflow Simulator

Workflow is executed in a instance of Siebel client.

- ✓ Both tools and client should be connected to same database

- Use Simulate toolbars to start simulation and to navigate to next step
- Upon Start , Siebel Client opens up
- After successful completion of simulation , the results can be viewed in the client
- When errors are encountered, simulation will stop and displays the error



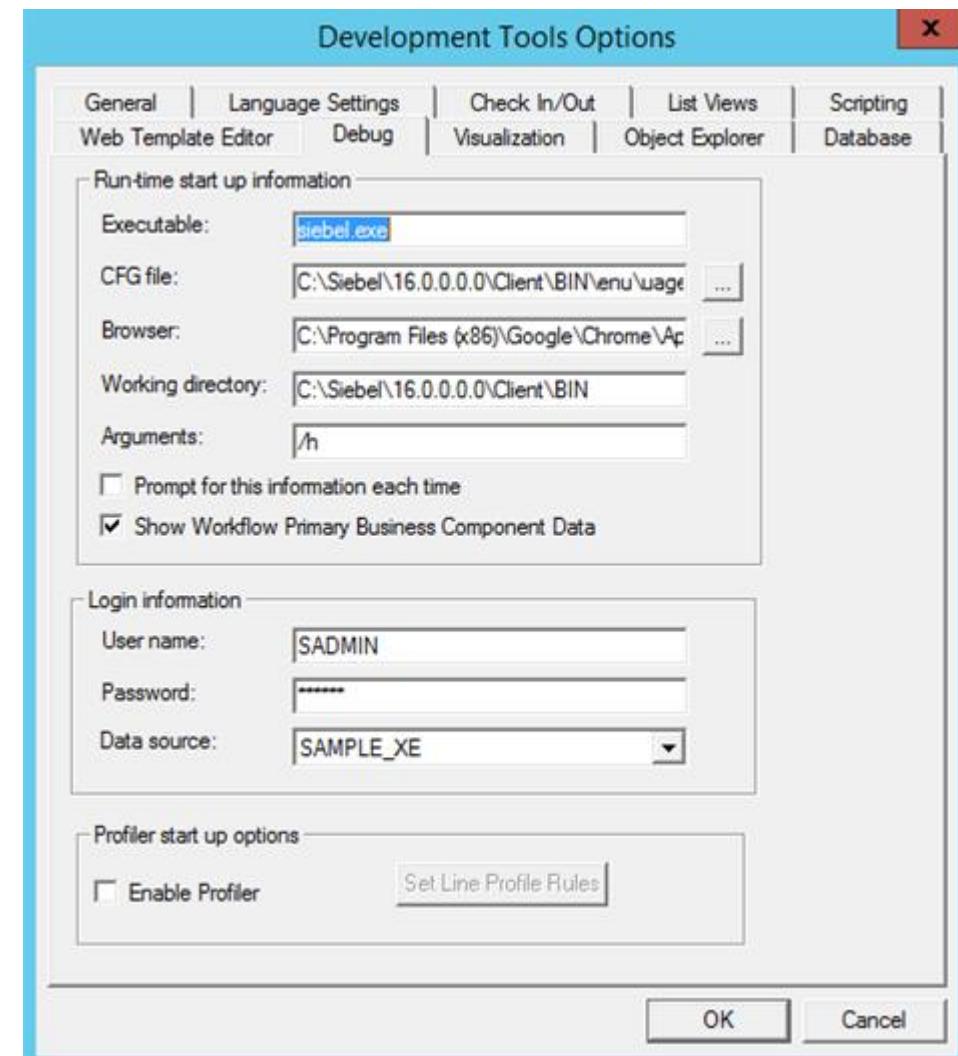
# Testing Workflow Process (4)

## Enabling Workflow Simulation

Configure the connection to the Siebel Run-Time Database :

In Siebel Tools → Select View → Options → Debug

- ✓ Simulator share the parameters used by the debugger
- ✓ Executable: Siebel.exe
- ✓ CFG File: File along with the path – specific to application
- ✓ Browser: Chrome executable file path
- ✓ Working Directory: Bin folder of Client
- ✓ Arguments: /h → Enables debug mode
- ✓ Specify the runtime Siebel instance:
  - User Name, Password and Data Source



# **Testing Workflow Process (5)**

## **Testing using Workflow Simulator**

Steps to follow for simulating a WF using WF Simulator:

1. Specify the Test Record
2. Start the Simulator
3. Start the Simulation
4. Execute the Workflow



# Testing Workflow Process (6)

## Specify the Test Records

In Client, create test records for simulation

- ✓ Click Help → About Record to fetch the ROW\_ID of the record

Enter the above ROW\_ID in the Default String of Object Id Process Property in MVPW of the WF to be tested

- ✓ When the WF is invoked in Production environment, the ROW\_ID will be passed automatically
- ✓ Simulating locally requires the value to be passed explicitly
- ✓ After successfully testing the WF , remove the default string and check in the WF

Multi Value Property Window								
Children of Test Create New Contact: 0								
Process Properties		Process Metrics						
Name	Display Name	In/Out	Changed	Business Object	Business Component	Virtual Field		Default String
Active Document Id	Active Document Id	None	TRUE	Contact				
Compound Product Number	Compound Product Number	In	TRUE	Contact				
Object Id	Object Id	In	TRUE	Contact				95IA-9E5BV
Price List Id	Price List Id	None	TRUE	Contact				
Target Document	Target Document	None	TRUE	Contact				Order



# Testing Workflow Process (7)

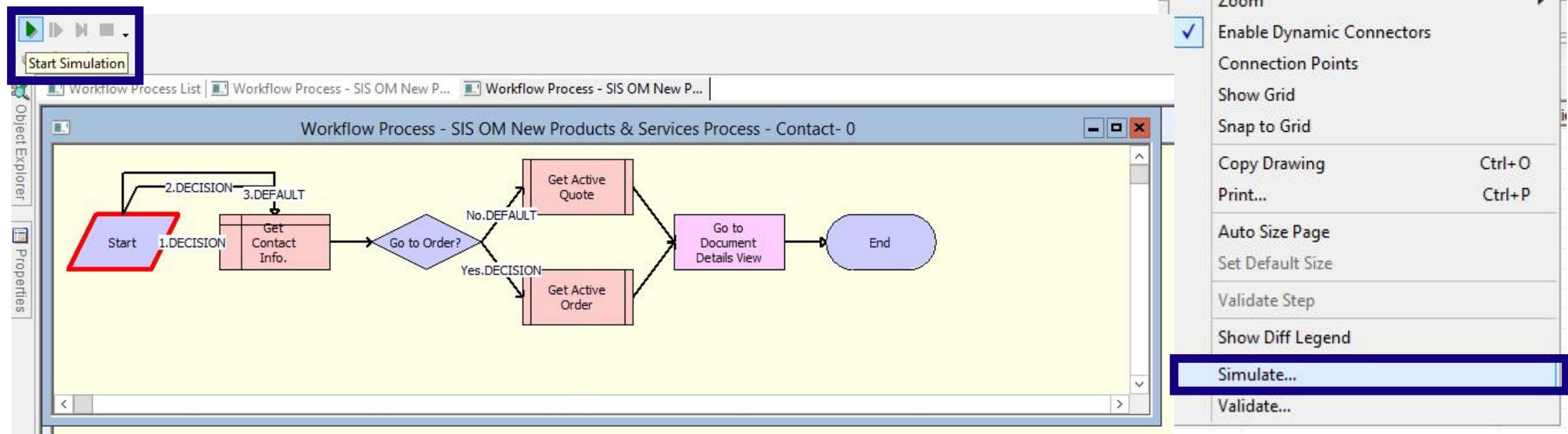
## Start the Simulator

Close all the instances of Siebel client

Right-click on the Designer → Select Simulate

✓ Designer now opens a simulator window

- Simulator Window is a read only version of WF
- Background looks different when compared to the designer



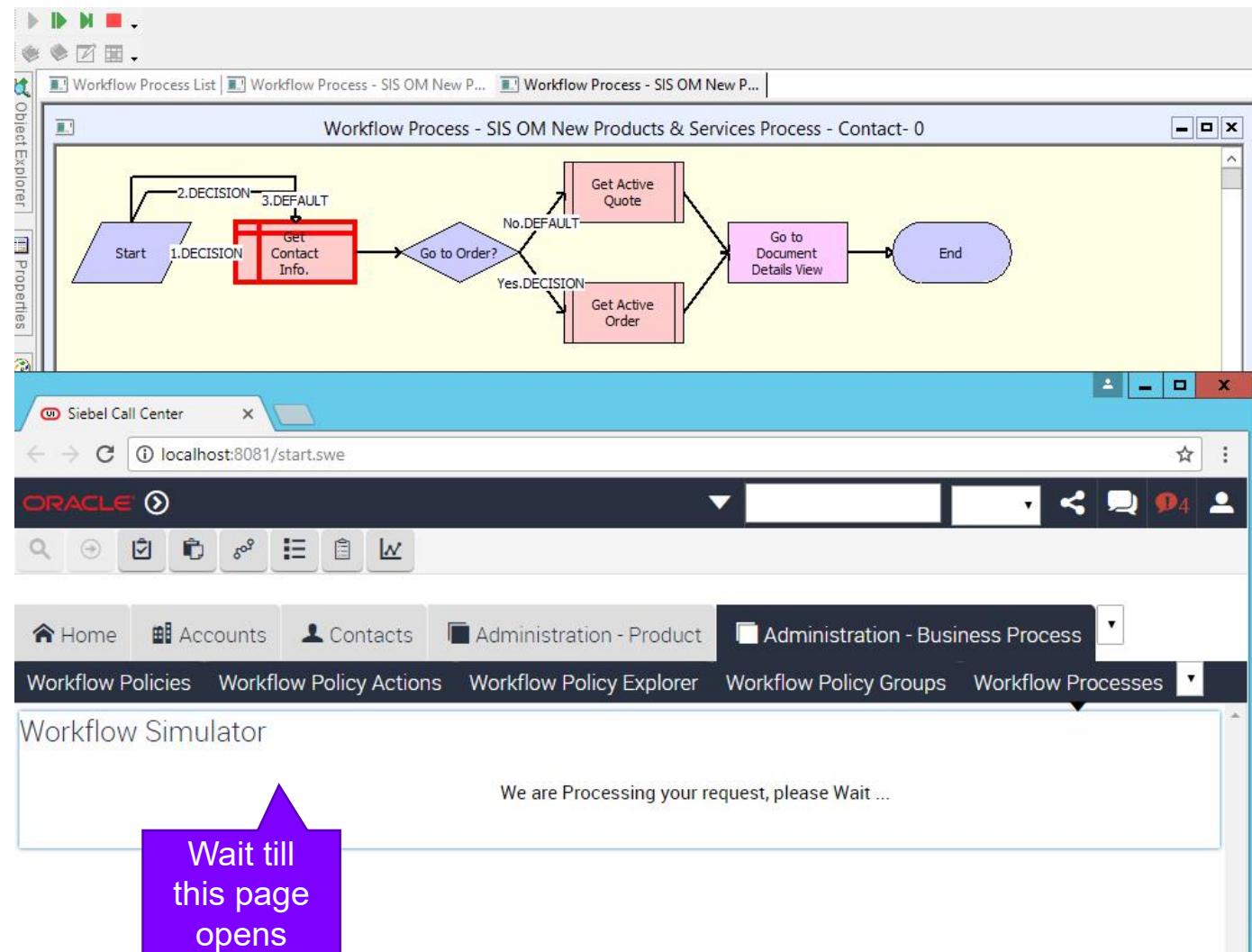
# Testing Workflow Process (8)

## Start Simulation

Click the start button in the simulator toolbar

- View → Toolbars → Simulation to display this toolbar

New Siebel client instance is launched



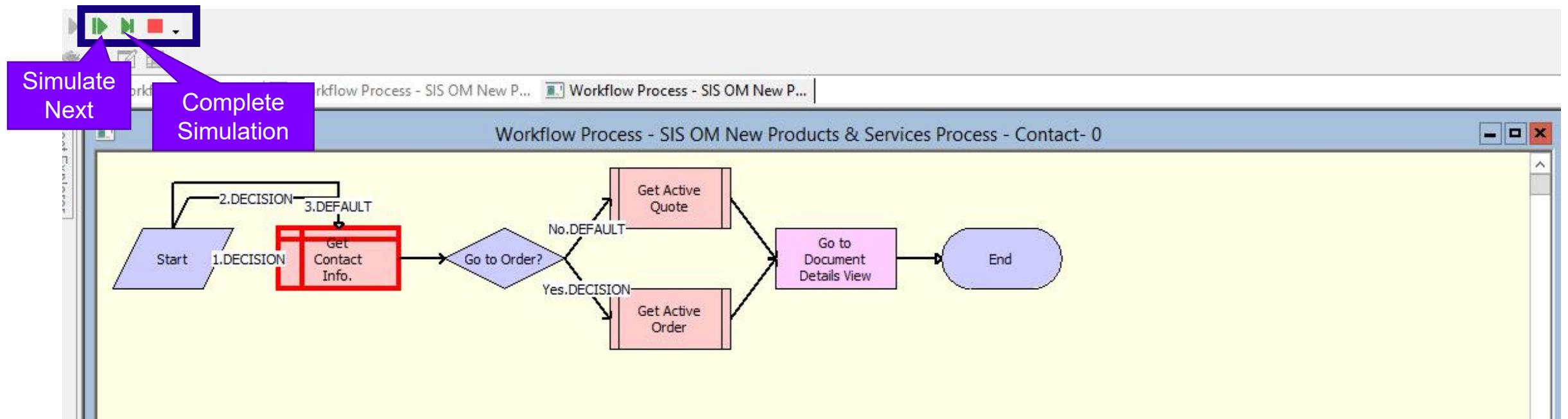
# Testing Workflow Process (9)

## Execute the Workflow

Simulate Next button allows step by step execution of WF

Complete Simulation button when clicked will execute WF steps till the end from the current step at once without stopping

- To check branching conditions, Simulate Next button to be used
- Watch window can be used to inspect:
  - ✓ The values of the process properties at each step
  - ✓ Value of Fields in BC

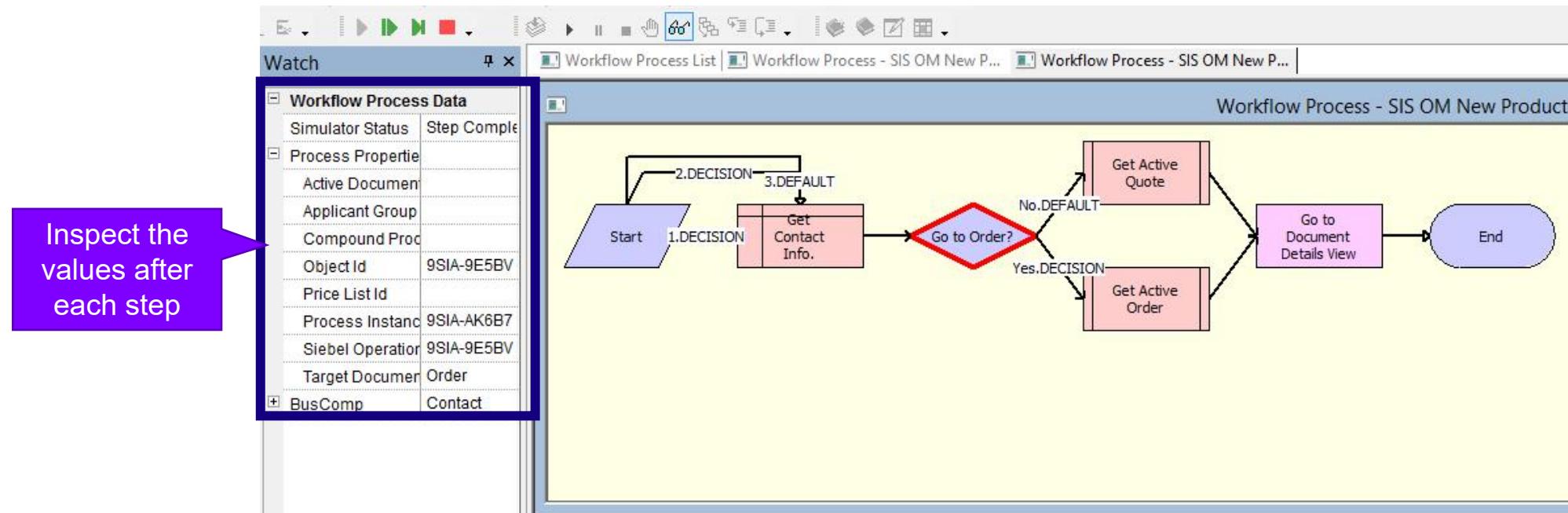


# Testing Workflow Process (10)

## Execute the Workflow

Inspect the Watch Window to verify the values of the process properties

Values of the user added Process Properties can be edited during simulation



# Testing Workflow Process (11)

## Execute the Workflow

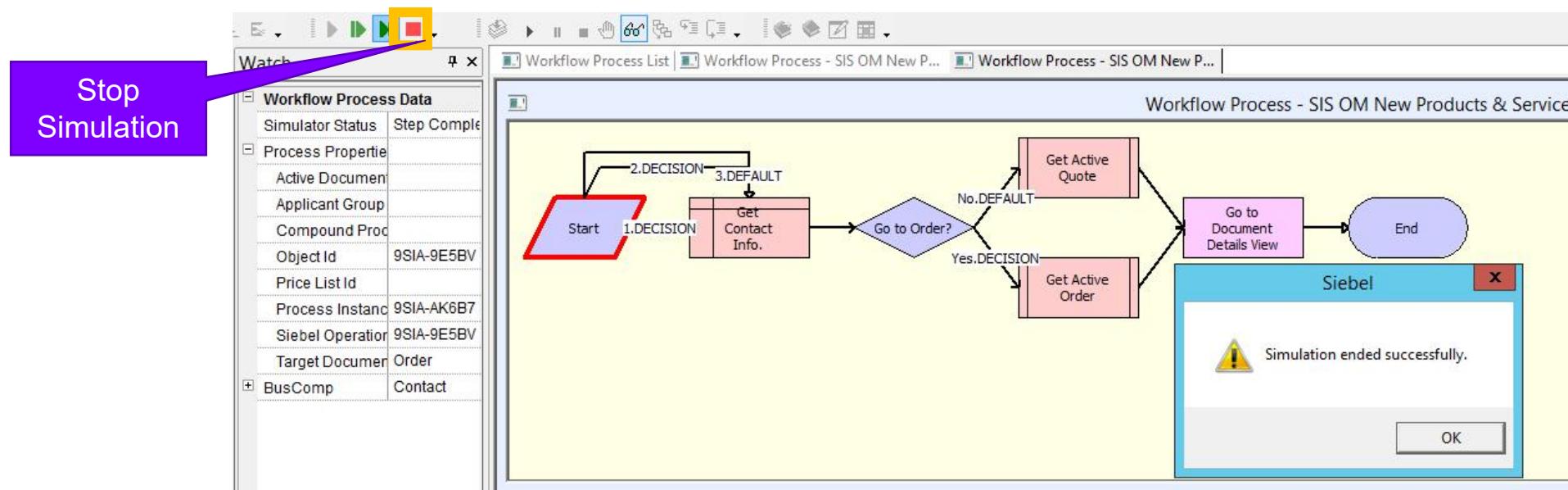
To complete the simulation:

- ✓ Click on Simulate Next or Complete Simulation Button

Using the Watch window verify if the process properties contain correct values

Click the Stop button to Stop the simulation anytime during the simulation

Inspect the test record in client and Verify the results



# Testing Workflow Process (11)

## Workflow Simulator Considerations

Cannot simulate WF that invoke Server Components

- ✓ Must simulate these directly on the Thin client

Cannot simulate WF with run-time events on the start step

Can simulate WF having User interact step

- ✓ Requires to perform the activity in the Client to proceed with simulation



# Topic List

#No	Module Topics
1	Testing Workflow Process
2	Deploying Workflow Process

# Deploying Workflow Process (1)

## Overview

This includes transferring of the Object definitions from Repository Tables to the Run-Time tables to make it available for use

It consists of completing the WF in the Tools

- Click on Publish Button to Complete the WF
  - ✓ Status of the WF will now be Complete
- Check in the Object definitions

Administrator will login to Server Environment to activate the Workflow in the run-time environment

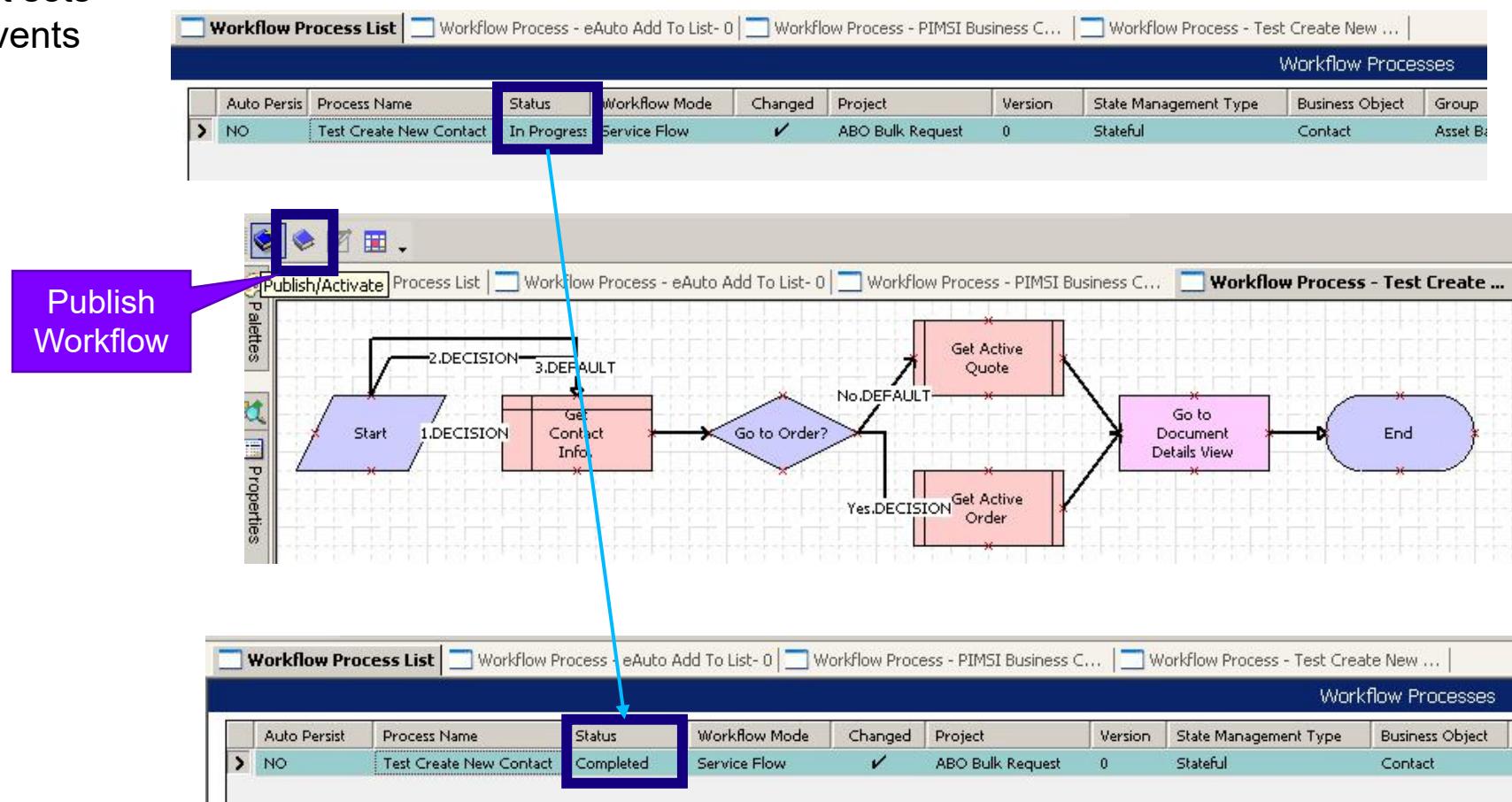


# Deploying Workflow Process (2)

## Publish Workflow

In Siebel Tool, click on the Publish button on the Workflow Toolbar. It sets the status to Completed and prevents further editing of Workflows

It makes workflow available for activation

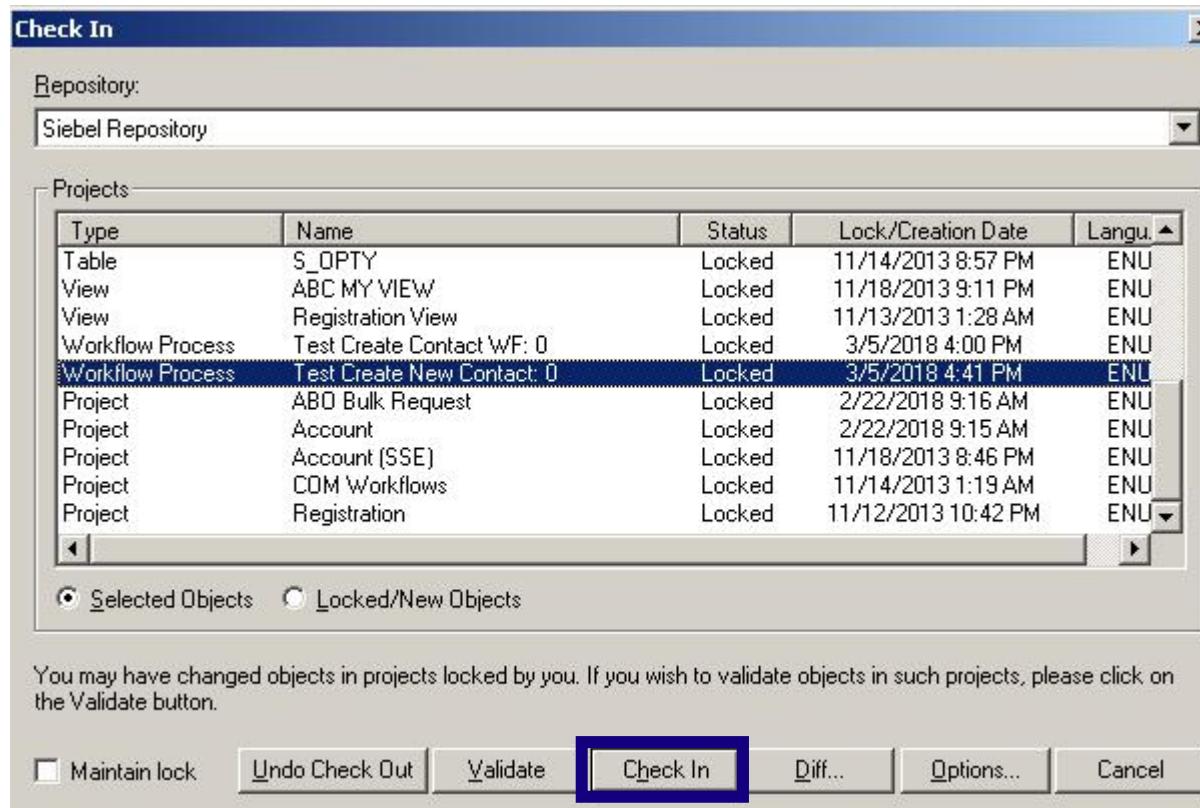


# Deploying Workflow Process (3)

## Check-in the Workflow

Check in the completed WF to the server repository

- ✓ Siebel web client can now access the Workflow



# Deploying Workflow Process (4)

## Activating the Workflow

To activate the Workflow:

Go to Administration – Business Process → Workflow Deployment

Query for the Workflow and Click Activate

- ✓ The WF Process will be seen in Active Workflow Processes with Deployment Status as Active

The screenshot shows the SAP Fiori interface for managing workflow processes. At the top, there's a navigation bar with various icons and links: Home, Accounts, Contacts, Administration - Product, Opportunities, Quotes, Fleet Management, Administration - Fleet Management, and Administration - Business Process. Below this is a secondary navigation bar with links: Workflow Policies, Workflow Policy Actions, Workflow Policy Explorer, Workflow Policy Groups, Workflow Policy Log, Workflow Deployment, Workflow Processes, Workflow Instance Admin, and Workflow Instance Monitor. The main area displays two tables.

**Repository Workflow Processes:** This table lists workflow processes that have been completed. A callout bubble highlights this table with the text: "Repository WFs which are completed are listed". One row is visible: "Test Create New Contact" (Business Object: Contact, Status: Completed, Group: Asset Based Or..., Version: 0Service Flow).

**Active Workflow Processes:** This table lists activated workflow processes. A callout bubble highlights this table with the text: "Activated WFs are seen here. Refers to Run- Time Database". One row is visible: "Test Create New Contact" (Business Object: Contact, Status: Active, Group: Asset Based Or..., Version: 1, Deployment Status: Active, Activation Date: /, Expiration Date: /, Replication: None, Monitoring Level: None).

At the bottom left, there's a purple arrow pointing right. The bottom right corner contains the copyright notice: "Copyright © 2020 Accenture. All rights reserved."

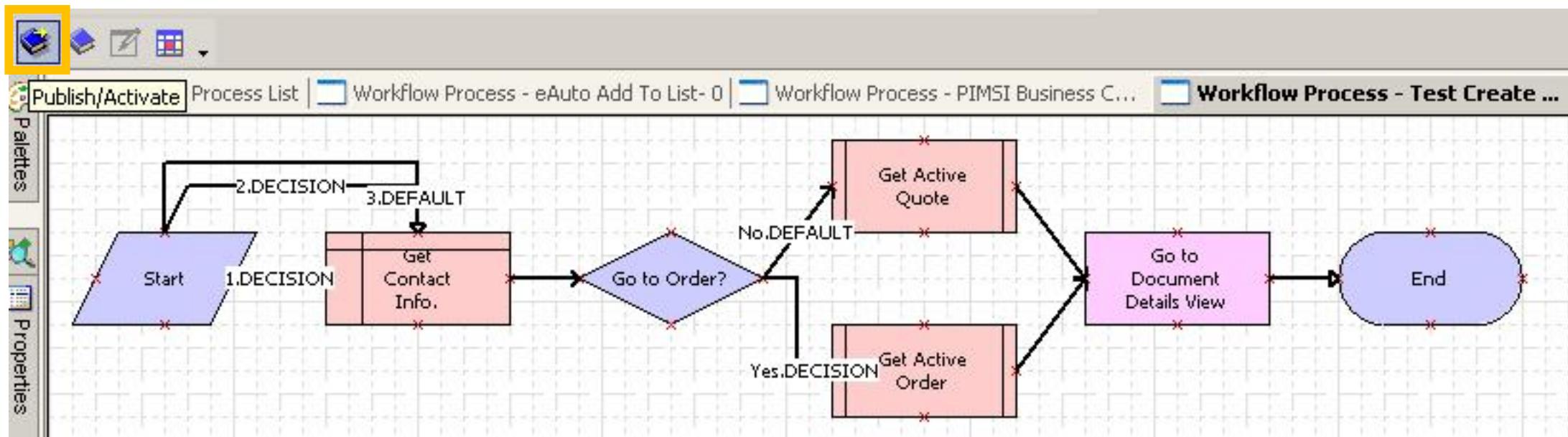
# Deploying Workflow Process (5)

## Publish/Activate Button

From Siebel Tools Workflows can be published and activated at once

Click Publish/Activate Button on the toolbar

- ✓ Sets the WF status to completed
- ✓ Transfers the definitions to run-time tables from the repository tables
  - Client used for testing this WF should be same as the Siebel Tools Database



# Deploying Workflow Process (6)

## Deployment Considerations

Publish/Activate all the child workflows first , i.e. Sub Process, that are referenced by the deployed Workflow

- These should be available for the Workflow to access

Compile any new repository object referenced in the deployed workflow

- Example: BC, Fields, Views



# Deploying Workflow Process (7)

## Revising Workflows

Workflows are versioned

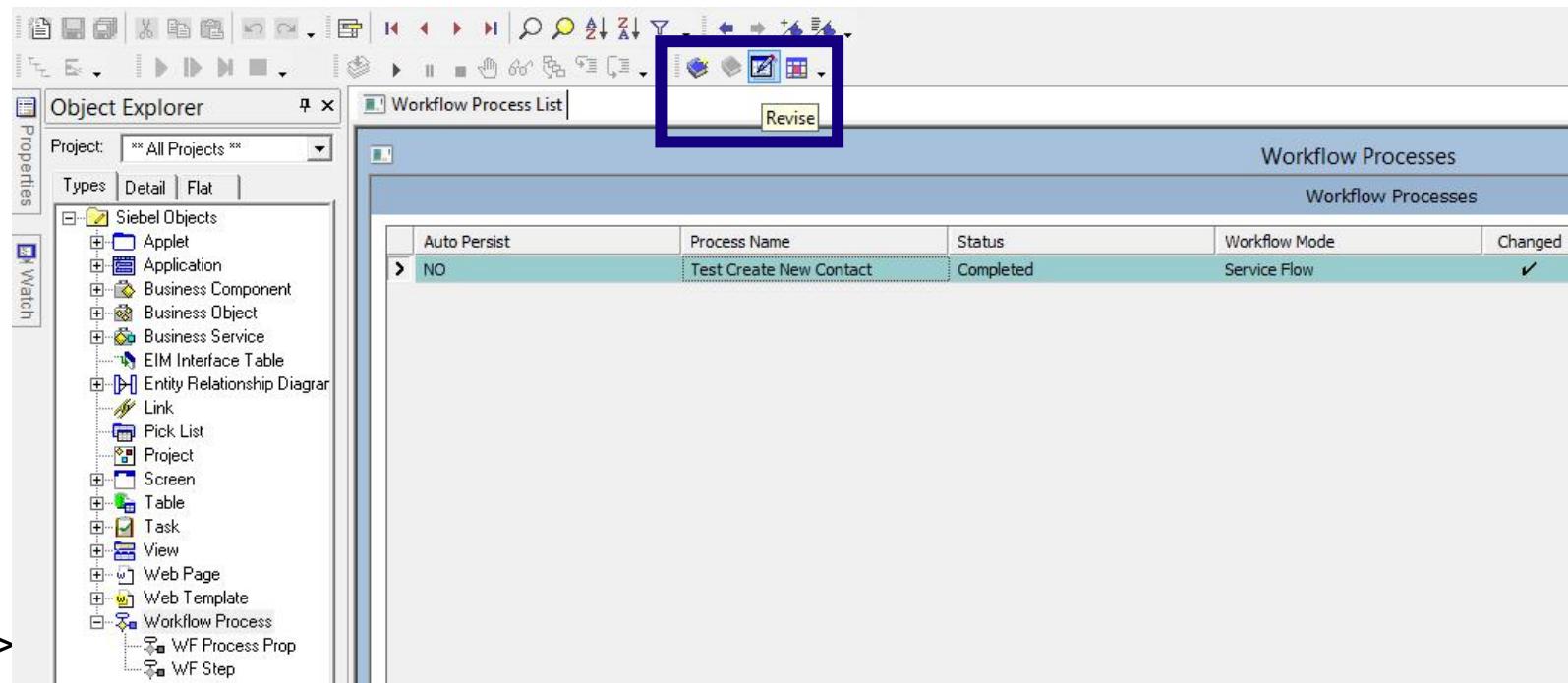
- ✓ Existing versions are kept and New versions are created

To Revise a Workflow:

- ✓ Select the desired WF and check out
- ✓ Click on Revise Button from toolbar
  - Creates a copy of the WF
  - Status = In Progress
  - Version = Incremented by 1
- ✓ Edit, Test and Deploy the WF

Upon activating revised WF :

- ✓ In Administration – Business Process ->
  - New Active version of the WF will be Active
  - Prior WF versions will be Outdated



# Knowledge Checks

## Answer the following

1. Steps involved in testing WF using Simulator is provided. Please arrange them in right sequence.
  - A. Start the Simulator
  - B. Start the Simulation
  - C. Specify the test record
  - D. Execute the Workflow
2. Impacts on the revision of workflows. Select all that is applicable :
  - A. Create a new version of WF
  - B. Incorporate edit changes in the original version
  - C. Increment the WF version number
  - D. WF Status is set to In Progress
  - E. Validate the WF



# Module Summary

**Now, you should be able to:**

- Explain how to:
  - Test a workflow process using workflow simulator
  - Deploy a Siebel WF Process



# **Thank You**

1. **Business Component** is a logical grouping of data from one or more tables. One base table and one or more joined tables.
2. BC consists of several fields including Single Valued Fields.
3. Single value field references to single column in the table.
4. Class property indicates the C++ used at runtime by the BC>
5. By default, CSSBusComp is used.
6. Single value field – join property is blank, column property will display column from Base Table.
7. In Single Value field- Type property specifies the data type for the field. It must correspond to the physical type of the column in table. By default, Siebel populates DTYPE\_TEXT.
8. ID field is mapped with ROW\_ID column, created field is mapped to CREATED column.
9. **Joins**- Business Components needs to pull data from additional or related tables to display them on applets, for further processing in the BC.
10. Single value field referring to these joined table columns are always read only.
11. Join always returns one row.
12. Relationship between BC and Joined Table is always M:1 or 1:1.
13. A foreign key is needed to establish this relationship. An FK column on the base table will refer to the PK column of the joined table.
14. FK field on the BC is mapped to FK column.
15. Implicit Join allows the BC to pull data from the extension tables on 1:1 basis
16. SVF pointing to extension table will have- join property set to the name of the extension table, Column property is the column from the extension table. Ex. Hobby field in contact bc is mapped to ATTRIB\_06 column from S\_CONTACT\_X table.
17. Explicit Join specifies the relationship between the BC and Joined Table. It includes a Join Definition and a join specification which are used to map SVFs to Joined table columns.
18. Join is the child object definition of the bc. It indicated the joined table from which to retrieve data.
19. Multiple join definition can be created for the same table.
20. ALIAS property in join is used to distinguish multiple joins.
21. Outer Join Flag when set to TRUE will return all records from the base table even when no records exist in the joined table.
22. A join specification must be created along with join definition, it defines how the record is retrieved. Foreign key and primary key which is used to relate the base table to the joined table.
23. Join property will have the ALIAS name of the join. Column property will be mapped to Joined Table.
24. Prebuilt Siebel application has many 1:1 extension table. These have 1:1 relationship with the base table. Ex. S\_OPTY\_X is 1:1 extension table for S\_OPTY.
25. PAR\_ROW\_ID in the extension table sets up the relationship with the base table. Ex. PAR\_ROW\_ID in S\_OPTY is S\_OPTY\_X mapped to ROW\_ID of S\_OPTY.
26. Implicit join is established in the table column and not under join definition. Fields mapped implicit joined columns are editable and always involve an outer join.
27. SVF based on the extension table columns will have- join property as the name of the 1:1 extension table, column property will refer to columns from the joined table. These fields are editable in the UI.
28. Business Components fetches data from only one table. False
29. Identify the incorrect statement regarding explicit joins. Automatically created for 1:1.
30. Relationship between BC and Joined table is always 1:1 and M:1.

31. Party Entities, Party data- data that describes the company itself and the way it is organised.
32. Party data can be individual person or group.
33. S\_PARTY is the base table for all party data. It consists of few columns that store party data common to all party bc.
34. Party BC, data is stored in the extension tables.
35. Party data is represented using Part BC, ex. Accounts, Contacts, Employee, Position.
36. Party BC can be classified as follows: Person Related (Employee, User, Contact) Organization Related (Account, Division, Organization) Access Control (Position, Access Group, User List)
37. Person Related Party BC is used to store person, user and employee specific data.
38. Party BC and their extension tables are: S\_CONTACT for Contact, S\_USER for User, S\_EMPLOYEE for Employee. Store main data in S\_CONTACT.
39. Organization Related Party BC stores data related to organization, division and accounts.
40. Party BC and their extension tables are: S\_ORG\_EXT for Account and Division, S\_BU for Organization. Store main data in S\_ORG\_EXT.
41. Access Control Party BC, certain business components are used for access control and they include- User List – S\_USER\_LIST, Access Groups – S\_PARTY\_GROUP, Positions – S\_POSITN.
42. Party Inter Tables- S\_PARTY\_PER, relates other parties with person party BC, ex. Relationship between user list and user/users and position. S\_PARTY\_REL, relates any two party BC/ ex. Relationship between contact and account/account and account.
43. **Party data Model**, it allows all instances of party BCs to be treated similar in certain aspects. They have common primary key and user keys, provides ways to relate instances of different party BCs. Ex. An access group of positions and accounts can be created.
44. Implicit join used between party business components and party extension tables. Name of the join is the name of the extension table. Ex. Contact BC fetches contact details such as First Name, Last Name from S\_CONTACT Extension Table. This is used for mapping party fields in party BC.
45. Explicit Join is created to fetch data from party tables to the non party business component. Such as account details in opportunity list applet.
46. Explicit join definitions are created to fetch data from S\_ORG\_EXT to Opportunity BC.
47. Explicit join is used to fetch party data into party BC. Ex. Parent account data into account list applet.
48. Join definitions are created for establishing the relationship.
49. Business Objects is a collection of related business components, it represents a major business entity. It includes Parent BC, Child BC, Links that relate Parent and Child Data. They provide foundation for views.
50. Business Object definition includes Name Property which is the Name of the BO, generally it is the name of the Primary.
51. Business Object Component is the child object type of the business object, it specifies the business component included in the business object.
52. BusComp property is the name of the BC. Link property relates the parent and child BCs.
53. Link Definition identifies which child data to fetch. Includes foreign keys to be populated when new child records are created. Link definition are used with both 1:M and M:M relationships between parent and child data.
54. 1:M Link definition identifies 1:M relationship between Parent and Child BC. Name is always defaulted to Parent BC/ Child BC.
55. Cascade delete property of a link determines if a child record is deleted when the parent record is deleted. The cascade delete property on the link is set using following values: Delete- if parent record is deleted then all child records are deleted. Clear- If the parent

record is deleted, the fk references is removed and the value in the fk column is cleared.

None- If the parent record is deleted, no child records are deleted and no foreign key column is cleared. The default setting is None.

56. Cascade delete is not available for a many-to-many link.
57. If set incorrectly, the cascade delete property might cause data integrity problems or orphaned records.
58. M:M Link definitions identifies M:M relationship between Parent and Child BC. An intersection table is used to establish the link definition and resolve the relationship.
59. Inter Child Delete Property: Is true- to delete the entry in both intersection table and child BC when the child record is deleted. Is false- to delete the entry only in the intersection table and keep the child record.
60. Links for grandchild data, to fetch grandchild data, Parent- Grandchild relation is defined. Ex. Opportunity/Activity Plan, Activity Plan/Activity Plan Action.
61. Activity Plan Action is Grandchild or Opportunity.
62. BO, before creating a new BO- check if a BO exists for the BC, if not, then create a new record in business object definition. Name property- Name of the BO.  
Project Property- Name of the Project.
63. Before adding the BC to BO, check if the BC already exists under the business object components definition while adding for existing BO. If not, then create a new record- Bus Comp Property- Name of the BC, Link Property- Name of the Link.
64. Check if the Link exists between the BC and the Parent BC. If primary BC, then leave the link property blank.
65. **Links** decide the relationship between the parent and child BC. 1:M or M:M relationship.
66. Resolve the foreign keys to set up the relationship, create a new record in the link definition tools. Parent business component property- parent BC name. Child bc property- child bc name.
67. In case of 1:M link: source field - Primary key of parent BC, Destination field- foreign key of child bc.
68. In case of M:M link – inter table, inter parent column, inter child column, properties must be filled.
69. Business component properties business components have set properties that can prevent from deletion insertion merging and updating records example once price list have been created the cannot be deleted or changed this property is applied to all interaction with the business component via applets workflows extra owner delete property when set to true enables only the record owner to delete the record if the owner is a team then the record can be deleted only by the team member who is primary.
70. Business component properties: Search specification enables a specific condition based record fetching, it consists of fields, operators, constants, functions, profile attributes. Typically used when multiple BCs based on same table are involved. Search specification at applet and at BC gets ANDed to fetch the resultant set and the applet displays it. They generate a WHERE clause that is run on the DB layer in SQL.
71. Sort specification determines the sorted order for the retrieve records it use DESC to sort in reverse order it cannot be set at applet level.
72. Expression builder is used to build expressions for search specification and sort specification it is useful in finding Syntax errors it can validate the expression seals operators functions at Sector are available and can be used to build the deserved expression
73. Field properties: Required is set to true to make the field required at business layer, asterisk in red is displayed along with the caption of the field.

74. Read Only when set to true, the field will remain non editable, such fields are greyed out on the UI.
75. Force Case specifies the case of the field. can validate these values only: UPPER, LOWER, FIRSTUPPER
76. Validation restricts the values that can be entered for a field which single value it does not support multi value field validation and only when the field is modified in the GUI it is evaluated
77. Validation message is custom message which is displayed when the validation property is violated. message display mode needs to be selected appropriately for which option to be displayed
78. Pre default value is default value assigned to field upon new record creation it can be constant value or expression. expression can consist of fields from same BC or parent BC
79. Post fault value assigns a value to the field only when the user does not enter the value before the record is saved to the database. expression will have same Syntex at the predefault
80. Calculated field is an expression for calculating the value of a specific field is specified by the calculated value properties. 1) the calculated value and properties of field validation are restricted to 255 characters.  
 2) column property remains blank which means these are not stored in database.  
 3) calculated field is evaluated at the runtime and application does not validate the values that this field contains.  
 4) sorting of calculated fields, it not possible.  
 5) allows queries on calculated fields  
 6)calculated fields may be based on the result of another calculated field lying in the same business component.  
 7) created by setting calculated property as true and providing the expression in calculated value.  
 8) read only fields can lead to performance issue.
81. About user properties these are object definitions which are used for configuring a specialised behaviour beyond what is configured in the parent object definitions properties these user properties can be configured for object types. Applet
82. User properties of business component read only field it is specified of business component field which determines whether or record is read only and the current record is read only when the value that this will contains is true
83. B C red only flag
84. Field user properties: Required makes the field a required field under concert in conditions. Text length override specifies that text to length of the field rather than that of the database column defines the maximum field length
85. Additional user properties: application user properties: ClientBusinessService- user property calls a business service from a browser script.  
 PDQDisabledView- user property disables predefined query (PDQ) dropdown list for view.  
 OverrideViewCache- user property disables caching for a view.
86. Applet user properties: DisableNewRecord- Siebel CRM calling Newrecord in the current applet is prevented by this user property.  
 CanInvokeMethod : MethodName user property enables or disables a method or a button
87. BC read only field is specified the field of business component which determines whether a record is read only and the current record is read only if the value that this will contain this true.

Field Read Only Field: fieldname: sets a business component field to read only.

Named Method n calls a business component or business service method or sets a field value.

On Field Update Invoke n: calls a business component method when Siebel updates a field.

On field Update Set n: sets the value of a business component field when Siebel updates another field.

88. Siebel provides business components to capture details of most common entities, but not all possibilities. Ex. Sales organization might require details of the contact. Educational background like highest education qualification, which colleges attended, marks scored, hobbies, sports played. To capture this information: more fields are required, 1:M relationship to the contact BC as parent is required.
89. To capture those extra details, create a new BC which is child to the existing BC (Contact). Based on the existing 1:M extension table supplied by the Siebel data model. Ex. Create 2 new BC as child to Contact BC (favourite restaurant BC).
90. 1:M Extension Table: to base the BC on a 1:M extension table, check if the table already exists in the repository, these extension tables have predefined columns ATTRIB\* to store data of various data types.
91. 1:M extension table contain 3 user key columns: NAME, TYPE, PAR\_ROW\_ID, these collectively must be always unique for a record.
92. Type column can be used to store the type of the record, ex. If the record type is favourite restaurant or college details.
93. To create a new BC, new object wizard is used, Select Business Component, specify S\_CONTACT\_XM as the base table for the BC, create fields mapping to columns: NAME, PAR\_ROW\_ID, Type. To store other details ATTRIB\* columns can also be mapped.
94. An XM table stores multiple data of different types, to fetch the data of specific type, Search Specification and Predefault properties must be configured, When Restaurant details are entered, its type is always defaulted to "Restaurant" using Predefault property. When this BC is initiated, it shall always retrieve the Contact type= Restaurant.
95. To assign BC to a BO: 1. a link needs to be created, 2. add the BC to BO as child definition, 3. create applet and views as required, 4. assign the view to the screen and administer it.
96. To create a link, specify the parent and child BC, relation between them in source and destination field.
97. To add BC to BO, firstly identify the BO that corresponds to primary BC, add child BC created under the business object components definitions, make sure that we set the link appropriately.
98. List applet is required to display the child data, new object wizard is used to create the same. Type field will not be displayed on the applet as we have defaulted the value in the BC to prevent users from editing the value.
99. To expose the new list applet on UI, it is required to create a new View, Use New Object Wizard to create, assign the new view to a screen, Administer the view in the client.
100. Requirements may arise which asks for new fields to capture more details, requires a new BC to capture additional business entities.
101. Examining the Siebel data model first, check whether the required columns and fields exist, if not, use existing 1:M extension tables for the new BC.
102. If data model does not support the desired new fields, then create new columns in the base table.

103. Avoid mapping new fields to existing unused columns, they might have been used elsewhere or they can be used in future releases. Avoid mapping fields 1:1 extension table column.
104. Always use Siebel tools to extend the database layer, extend columns to existing tables or create new tables using tools alone, never create using SQL scripts.
105. Select the table to be extended, create new column record, fill up the desired properties with the appropriate values, name automatically prefixed with X\_.
106. Standalone table creation: 1. Select table from new object wizard 2. Provide the details as desired 3. Select a stand-alone table.
107. New object wizard creates a stand-alone table with, 1. Type: data (public), 2. Required system columns, 3. Index P1 on the ROW\_ID.
108. To create 1:1 Extension Table, 1. Select a base table 2. Type of the table will be restricted to data(public) 3. Multiple extension tables relate to same base table but not to each other.
109. New object wizard creates 1:1 extension table with, 1. Type: extension 2. Required System columns 3. PAR\_ROW\_ID foreign key column for the parent base table. 2 indices: index P1 on the ROW\_ID & on PAR\_ROW\_ID and CONFLICT\_ID.
110. To create a 1:M extension table, select a base table, should create only if the parent table does not have any other existing 1:M extension table, type of the table will be restricted to data (public).
111. New object wizard creates 1:1 Extension table with, 1. Type: data (public) 2. Required system columns 3. PAR\_ROW\_ID column foreign key to the parent table S\_PROD\_INT 4. TYPE and NAME columns also gets created.
112. To create an select 2 parent tables that will be intersected, wizard will set the type to data (public), also foreign key for both the tables also will be created.
113. New object wizard creates intersection table with, 1. Type: data (public) 2. Required system columns 3. PROD\_ID column foreign key to S\_PROD\_INT 4. OPTY\_ID column foreign key to S\_OPTY.
114. 3 indices are also created, 1. P1- on ROW\_ID, 2. U1- on PROD\_ID, OPTY\_ID, TYPE, CONFLICT\_ID, 3. F1- on PROD\_ID (FK to second table)
115. Before propagating the changes to the server database: test changes locally by applying them to local database, it reduces the chances of mistakes in the server schema.
116. Best practice is to Apply changes to the local database and test, migrate the changes to server database, propagate these changes to other developers.
117. Apply/DDL: click Apply/DDL button to make changes to database, choose either apply to database or generate DDL script, changes will be preserved across Siebel upgrades. Compile all the relevant object definitions and projects. Test the changes in local before migrating them to server, query for the table and columns in the DB using a SLQ developer and check.
118. Migrate changes to the server DB: Check in the project to the server, 1. Copies the new table and column object definitions to the server Open server tools, 2. Select the table 3. Click Apply/DDL to make changes to the physical database 4. Click activate button to update the database schema version 5. Compile all objects and test changes in the server DB.
119. Propagate changes to other developers, to access the newly deployed changes, other developers can perform a GET (or) CHECK OUT of these objects from the Server.
120. A drilldown object allows user to drill down on a field in a list applet and takes the user to another view.
121. Hyperlinked fields with coloured texts and underline allow drilldown.

122. Drilldown object are created in Object Explorer > Applet > Drilldown object.
123. Drilldown is not supported on Pick applets, MVG applets and Association applets.
124. Drilldown can be either STATIC or DYNAMIC, a static drilldown always navigates to the same view, a dynamic drilldown navigates to different views, based on certain conditions, such as a field value.
125. Static Drilldown are of two types: 1. Drilldown within the same business components:  
a) Drilldown to detail view for same record b) destination view uses the same BO and BC. 2. Drilldown for a different business component: a) drilldown for a detail view of a related record. B) destination view used different BO and BC.
126. Thread Bar is used to track the previous active record, it provides a hyperlink to the former view and is updated when the user navigates to a different BO.
127. To configure a thread bar, thread bar properties need to be set at View object definition.
128. An applet toggle allows the user to switch between applets in the same view: This object type occurs in: Object Explorer> Applet > Applet Toggle.
129. Same placeholder could be shared by several applets in the view: configuring toggle applet involves adding only one applet in the view web templet, applets can be switched by configuring Applet Toggle object definitions.
130. Applet web templet needs to contain SWE Togglebar tag to enable Applet Toggle.
131. Applet Toggle can be either STATIC or DYNAMIC. In Static Applet Toggle user manually chooses from a dropdown list and selects the applet. In Dynamic Applet Toggle the applet dynamically changes based on a field value or certain conditions.
132. Configuring Applet Toggle – Static toggle: Create applet toggle definition for each applet in the view to be added in the toggle list, create these definitions under the applet from the view web templet items.
133. Configuring Applet Toggle – Dynamic Toggle: Properties of applet toggle, 1. **Applet:** Name of the applet to be toggled. 2. **Auto Toggle Field:** Specifies the business component field for dynamic toggle. Siebel examines the current value of this field to the value that the Auto Toggle Value property contains. 3. **Auto Toggle Value:** Specifies the business component field value for dynamic toggle. 4. **Name:** Name of the applet to be toggled. 5. **Sequence:** Specifies the order of display for dynamic toggle.
134. **PICKLIST** updates fields when a value is selected from the list, users are forced to pick values from the associated list. This ensures business rules and policies are met, makes data entry faster, reduces the errors.
135. **Static Picklist:** it allows user to select value for a column from the predefined list of values. When a value is chosen from the list, the value is saved on the field, A static picklist derives values from a table that a Siebel administrator maintains. These values do not change during runtime or upon user action, picklist can be BOUNDED: user can only choose from the provided set of values. UNBOUNDED: user can enter values he desires. This value is not entered into the picklist values.
136. **Static picklist values:** are stored in a table S\_LST\_OF\_VAL, type field indicates the type of static picklist it belongs to.
137. **Administer static picklist:** Navigate to administration- Data -> List of Values Explorer, select an existing picklist -> expand the type and select the child values folder, edit or add the picklist values here.
138. **Administer static picklist using tools:** LOVs can be administered using Siebel tools as well. Select screens-> System administration-> List of values. Query for the picklist type, edit or add the picklist values in the object editor and click clear cache.

139. **Create a picklist by** creating a new picklist type, creating values for the LOV type.
140. **A dynamic picklist** allows users to select records from a pick applet, they are created on a joined field. It displays records from pick BC. The values are dynamic in nature as they depend on the current BC. When a record is picked in the Pick Applet, it copies the primary key into the foreign key field, updates the values of the joined fields.
141. **Static picklists are based on the PickList Generic BC**, BC with base table as S\_LST\_OF\_VAL, it is the pick BC for the static picklists.
142. **Picklist object definition** defines static or dynamic picklist, it identifies the pick BC, also specifies if the picklist is bounded or non-bounded.
143. **Picklist property in the single value field** specifies the picklist associated with the field, a dropdown is seen on runtime on this field, SVF pick maps specifies one or more fields to be copied from the pick BC to the originating BC.
144. **Through SVF pick map many SVF can be updated** with the values from the pick BC, dynamic picklists will have following fields always copied: foreign key field to identify the pick record, additional fields that need to be updated and displayed on the screen.
145. **Pick applet** that displays the records for the dynamic picklist, displays multiple columns to make it easier for appropriate selection of record.
146. **Control and list column**, set the runtime to TRUE to enable picklist, for dynamic picklist: specify the pick applet property. For static picklist: pick applet property should be left blank. Dropdown will be enabled at runtime.
147. **Static Picklist Properties:** Project: Name of the Project, Business Component: Name of the Pick BC, Field: Name of the picklist field, Type: Static Picklist. Select create new picklist, provide name for the picklist. Enter the LOV type name and the values.
148. **Pick list wizard**, creates following: 1. Values in the S\_LST\_OF\_VAL table. 2. Static picklist object definition with the specific LOV type. 3. SVF pick maps to copy the pick BC field values to the originating BC. Sets runtime property to TRUE in all list columns and controls that refers to any static picklist, to activate picklist at runtime.
149. **Dynamic picklist configuration:** Using new object wizard configure a dynamic picklist, 1. Select Project 2. Select the originating BC 3. Provide the field name for the picklist field 4. Select types as dynamic picklist. Wizard prompts for a new picklist creating or use existing picklist, choose appropriately, select: pick BC, sort field in case required provide the name of the picklist and search spec if any. No update/no delete/ no merge/ no insert properties can be configured next.
150. **Wizard helps to describe the pick maps and the foreign keys**, create multiple pick maps, map the primary key in the pick BC to the foreign key field in the originating BC.
151. **Pick list wizard for dynamic picklist:** creates the: dynamic picklist object definition with pick BC, SVF pick maps to copy the field values from the pick BC to the originating BC. Sets runtime property to TRUE in all controls and list columns referring to the static picklist, to activate picklist at runtime. The wizard also opens an applet to create a pick applet if required.
152. **Constrain Picklist:** Constrain picklist filter records dynamically that have field values matching that of the corresponding fields in the originating BC. Ex. contacts displayed in the quote list applet is dependent on the account associated with the quote.
153. **To constrain a picklist:** Create pick maps, create a SVF pick map that needs to be matched with the originating BC. Set the constrain flag to TRUE, picklist now filters out only the matching records.
154. **A hierarchical picklist** shows values which is dependent on value chosen from another picklist. Ex. The area and subarea fields in the service request detail applet, are both

picklists fetching values from the list of values tables(S\_LST\_OF\_VAL). The LOVs available in the subarea picklist is dependent on value selected in the area picklist.

155. **List of values** for area and subarea are specified using the one LOV type, SR\_AREA LOV in S\_LST\_OF\_VAL. Parent LIC is used to state value of the parent. For each child value, parent LIC is updated with parent value.
156. **To configure picklists to support this hierarchy:** 1. The parent picklist is based on the PickList Hierarchical BC. 2. The child picklist is based on the PickList Hierarchical Sub-Area BC. Also, a search spec is added for Child Picklist to retrieve Parent type value selected. Parent Type= SR\_AREA selected.
157. **SVF Maps for Hierarchical Picklists:** create SVF Pick map to copy the field values after associating the picklist. Set constrain property to TRUE for the child picklist on parent field. To constrain the value of sub are based on the area chosen.
158. **A Multi Value Group (MVG)** is a UI element that displays child data in an applet, displays multiple child records for a single parent record. MVG button on parent applet is clicked to pop MVG applet which contains child data.
159. **Shuttle applet** MVG for a M:M relationship displays Shuttle Applet, displays both available child records on the left and selected child records on the right. Applets appears only in HI Mode.
160. **MVGs are an alternative** to detail views where parent and related child records are displayed. It makes: effective use of the space, does not require dedicated space in a view, easier to display multiple set of detailed records available from a single view.
161. **Advantages of MVGs** It allows creating queries that include both parent and child fields.
162. **Basic object definitions that are needed for MVG:** Multi Value Fields (MVF), Multi Value Links (MVL), Map MVFs to the Child BC SVF, MVG Applet, Associated Applet, Primaries for the performance.
163. **Multi Value Field (MVF)** is a field in the parent BC referring to field on Child BC, never refers a column directly.
164. **Multi Value Link (MVL)** child object definition of BC that indicates retrieval of data from the child BC. References Link object definition which explains relationship between the Parent and Child BC.
165. **Every MVF requires MVL**, MVF is mapped to SVF of Child BC through MVL.
166. **MVG Applet** This is a list applet where child records are displayed, MVG button when clicked from a list or from applet pops up MVG applet. It is based on Child BC.
167. **Association applet** required for MVG with M:M relationship, displays the list of child records available for selection.
168. **Invoke MVG applet** specify the MVG applet to be invoked from the List Column/ Control in MVG Applet Property, set runtime property as TRUE, to enable the MVG applet.
169. **Each MVG applet will have a primary** allows faster retrieval of child data for display in the parent applet, this record is designated as primary. Primaries improves performance by eliminating query to the child BC separately, one query will populate all the fields in the list applet.
170. **Primary Foreign key filed** is an FK field in the parent BC which references the PK of the Child BC. Child data is retrieved using the join to the child table.
171. **Siebel data model includes primary keys for many relationships** These keys begin with PR\_, to find primary foreign key: select the table of the primary BC, query in foreign key table for child BC.

172. **Primary foreign key** Map the primary foreign key column to a field in the parent BC. In MVL, set use primary join and primary id field properties.
173. **Auto Primary Property:** This property determines how the primary will be assigned when not selected: Default: First child record retrieved will be the primary, Selected: Highlighted record becomes Primary, None: User must specify the primary.
174. **Setting the Primary Record:** SSA primary field is exposed on the MVG applet to make it available for the user to set primary for the child record. Primary child record is set during the initial loading using EMI, if not set, then make the auto primary property to default.
175. **Use new object wizard to create MVG:** firstly, verify if a link exists between parent and child BC. 1. Provide Project 2. Select the parent and child BC 3. Select the link that MVF is based on the primary key field, fields from the child BC.
176. **MVG wizard creates:** MVL, MVF, does not set the runtime property or MVG applet properties on the list columns/controls, set them manually. Invokes MVG applet wizard if no MVG applet exists suiting the selected parent and child BC.
177. **Automating a Business Process:** A business Process is a sequence of tasks executed to accomplish a definite business objective. Ex. Order Management- Creation of quote to order and convert it to order submission. Automation options address the following challenges: 1. Standardize and maintain same business process across business units. 2. Assigning and routing tasks efficiently. 3. Timely response to the customer's tickets and service requests. 4. Assisting users with business process and best practices.
178. **Automation Technologies: workflow process** is similar to a flowchart which is used to detail about business processes of a company, it is one of the key automation techniques used for building businesses processes in any siebel application.
179. **workflow policy** defines conditions which when met invokes actions, actions runs the associated workflow process. workflow policies is effective for performing simple actions such a sending email or creating an activity or assignment.
180. **task UI** enables creation of a UI involving the steps, interactive operations along with branching and decision logic to help users through the task flow. we can also pause or resume any task execution if need arises.
181. **assignment manager** runs assignment rules to assign data to candidates.
182. **SmartScript** guides users with suggestions and services built on their profile and purchase history.
183. **State Model** controls transition of status depending on the existing value and user's current position.
184. **Business service is a business logic which can be reused and accessed throughout the application:** following are 2 types of business services: 1. Built-in BS which is defined in the Siebel Tools 2. Run-time BC which is defined in the web client. You can use a business service as a reusable code library that Siebel CRM can call from another script configuring BS involves properties and VB script or Siebel eScript.
185. **Repository Stored Business Service:** References the predefined CSSService class or a specialized class, these can be custom on Pre built-in BS.
186. **Many Pre built-in** vanilla BS are available to be used for specific operations, Customer order management, enterprise application integration etc. Editing/Modifying is not allowed for these objects, written in C++.
187. **Custom Business service are written in Siebel VB** script or Siebel eScript, needs to be compiled to SRF, when ever the BS is modified, these must be compiled into SRF. There are stored in the repository.

188. **Client stored business service:** These are stored in run-time database in addition to customer data. Administration- business service screen enables writing, editing and modify client BS. Written using Siebel VB or Siebel eScript, it is never executed if a BS with same name exists in the repository, it is not compiled to SRF. Hence directly available for Applications, we can move the BS from repository to client and vice-versa.
189. **Business service contains** multiple operations called methods, each method has set of input and output argument of specific type.
190. **To identify the methods of a business service:** Open Siebel tools, navigate to business service in object explorer, select business service methods child object definition.
191. **Arguments and types for method:** In Siebel tools, Goto business service -> Business service method -> Business service method Arg, all input and output arguments and their type are available.
192. **Invoking a method:** assigning values to input arguments, not all input arguments are mandatory. Retrieve values from the output arguments.
193. **Property set is a in-memory data structure used to:** Pass set of input arguments to the method to be invoked. Receives set of output argument from the method.
194. **A t:** 1. Represents data using name/value pairs 2. Has two defined properties-> Type and Value 3. Has an array for storing name/value pairs 4. Is created automatically while invoking a BS.
195. **Navigate Administration Business service ->** Simulator, select bs name and method, specify the input arguments through property sets. Name/value pairs are provided, also can be loaded using input file.
196. **To simulate the BS:** click run on one input, output arguments will display the resultant name/value pair, this can be saved to a file.
197. Siebel Workflow Process is a tailor-made business automation application that defines, manages, and enforces business processes.
198. Workflow Process is similar to a flowchart which has series of steps detailing out business processes of a company
199. Specifies input and output properties for every step, also for the complete WF Process.
200. WF process could be:  
Simple, E.g., Adding products to your cart  
Complex, E.g., handling entire Order Management
201. Workflow processes can manipulate data, can check for conditions, also can invoke a business service, task UI, or subprocess
202. Workflow designer in Siebel tools is used to create, examine, modify workflow processes  
Select workflow process from object explorer.  
Choose any existing WF  
Right-click → Edit Workflow Process → Opens Workflow Designer
203. Workflow steps: Every Workflows must have these steps:  
**Start** – WF always starts with this step  
**End** – WF always ends with this step  
Workflows can also include these based on the operation to be performed:  
**Business Service**  
**Siebel Operation**  
**Decision Point**
204. Operation steps: This can perform database operations such as:

Query

Insert

Delete

Update

NextRecord

Upsert

PreviousRecord

QueryBiDirectional

Always refers to one business component.

Search specification property can be used to locate the records to modify

205. Business Service step: It invokes a method of the business service

It can invoke both custom and vanilla BS

206. Decision point step: It enables branching to one of the multiple steps based on the

Input values

Each Decision step can have multiple branches based on the value

Here, only two branches are present – default branch and conditional branch

The decision is made based on the contents of the branch. Right-click on branch > Edit Conditions to evaluate the condition

207. Process Properties are used to store Inputs used by and Outputs produced by the workflow steps

Multi Value Property Window shows Process Properties and Process Metrics that are global to the WF

Click on the white space in the workspace to see the window

Upon New WF Creation, Set of Default Process Properties are created

Contain information regarding Current Object in process, error information, current operation steps being performed etc.

Custom process properties can also be created

208. To create a new workflow process:

Inside Siebel Tools, select the Workflow Process Object Type

Right-click → New Record. Create a new WF Process Object Definition

Provide a Name to the WF

Assign a Project

Select a Business Object to the WF

Provides Context for BC and Fields

Status of the WF is In Progress

Changes after deploying the WF

Right-click → Edit Workflow Process

To invoke the workflow designer

Version of the WF is Zero (0)

Version changes when we revise the WF

209. Steps to configure a WF using workflow designer: Create a new workflow process

->Specify the process properties-> Add workflow steps -> configure each workflow step -> validate the workflow process.

210. Workflow modes: Every WF has a runtime behaviour described by the Workflow Mode

Following are available WF Modes:

**Service Flow:**

A WF in this mode executes multiple steps to complete a task from start to end

This workflow completes the task in short span of time without stopping or pausing for any event or activity

Can be invoked as a sub process from another service flow WF, an interactive flow WF, or a long-running flow WF, but not from a 7.0 flow WF

Following are the limitations:

Never waits for any run-time event

Never use wait or user interact step

#### **Interactive Flow:**

A WF in this mode assists users in navigating across Siebel application

Includes a run-time event and one or more user interact steps

#### **211. Long-Running Flow:**

A WF in this mode can last for hours, days, or months

Not allowed to use user interact or wait step

We cannot simulate a long-running workflow using Tools simulator  
can be paused and resumed as an inbox item

#### **7.0 Flow:**

These WF provides backward compatibility for an existing workflow process

New WF should not be created with 7.0 Flow workflow mode

#### **212. Specify the Process Properties**

Select the Process Property Tab in MVPW Window to display the default process properties

Never change the default process properties

Create additional Process Properties required to store the values and used by WF Steps

#### **213. Adding Workflow Steps**

Add a Start and End Steps into the Workspace

Drag the steps from the palette to the Workspace

Insert other steps as and when required

Based on the requirement

Insert connectors to connect the steps

Make sure connectors are anchored and red at both ends when connected between steps

If not, then it shall throw an error while validating the WF

Connectors are dynamically drawn

#### **214. Configuring Siebel Operation Step**

For each Siebel Operation step:

Specific Business Component and Operation

Use properties window

Specific other child entities in the MVPW

Field Input Arguments, Search Spec Input Arguments, Output Arguments etc

#### **215. Configuring the Business Service Step**

Specify the Business Service Name and Business Service Method

Use the properties Window

#### **216. Configuring the Business Service Step**

For each Business Service Step:

Specify the inputs to be used by the WF

- ✓ Select the Input Arguments tab in the MVPW
- ✓ Assign a Literal Value / Expression / Process Property to each of the Input arguments

Specify the outputs of the step

- ✓ Select the Output Arguments tab in MVPW
- ✓ Not always mandatory to get outputs

- ✓ If any outputs exists, these should be mapped to process properties

## 217. Configuring Decision Point Step

For each branch originating from the Decision Step set the conditions

- ✓ Select the connector
- ✓ Right-click → Edit conditions
- ✓ Every Decision step must have a default connector
- ✓ Every condition connector will have a conditional criterion
  - Can contain fields and Properties compared to literals or Expressions

## 218. Configuring Decision Point Step (2)

Using Composer Condition Criteria box the conditions for branch are specified

- ✓ Never create conditions on default branch
  - This is the path selected when none of the conditions satisfies
- Conditions can be compared to Applet , BC , Process Property or Expression
  - ✓ Here it is compared to Process Property

In Object, Choose the desired property

On Click of Add Button, Conditions get added in the above pane.

Multiple conditions can also be added

## 219. Validate workflow process: Validation is done to check various syntactic errors that may occur in WF

To validate any WF:

First save the configuration done to WF

Right-click on the designer > Select Validate

Validate Windows pops out

Click on Start button

Check for errors and correct them

Validation doesn't ensure all the errors to be caught, some may arise during runtime

## 220. Additional workflow steps: SUB PROCESS:

Invokes another WF as a subprocess

USER INTERACT:

Navigates the user to a different view and waits for user activity

WAIT:

Pauses the WF for specific time interval before proceeding

STOP:

Stops execution of the WF when a specific exception occurs and terminates WF in error state

TASK:

Invokes a Siebel TASK UI

## 221. Managing workflow process: WF Process differ from other object definitions :

Though created in Siebel Tools, They are not compiled to SRF

These can be migrated using import and export of WF in form of XML files

After configuring the WF in Tool

Simulate the Workflow

Deploy the Workflow

## 222. Workflow simulator: The Process Simulator is a simulation tool that allows to step through a workflow process

Results of each step can also be viewed

Simulating a workflow process before deployment to production environment helps  
Verify if it works correctly

The results meet your business requirements

Workflow Simulation is controlled in Siebel Tools

To enable the Simulator:

Right-click on the Designer → Select Simulate

Workflow is executed in an instance of Siebel client.

Both tools and client should be connected to same database

Use Simulate toolbars to start simulation and to navigate to next step

Upon Start, Siebel Client opens up

After successful completion of simulation, the results can be viewed in the client

When errors are encountered, simulation will stop and displays the error

223. Enabling workflow simulation: Configure the connection to the Siebel Run-Time

Database:

In Siebel Tools → Select View → Options → Debug

Simulator share the parameters used by the debugger

Executable: Siebel.exe

CFG File: File along with the path – specific to application

Browser: Chrome executable file path

Working Directory: Bin folder of Client

Arguments: /h → Enables debug mode

Specify the runtime Siebel instance:

User Name, Password and Data Source

224. Testing using workflow simulator: Steps to follow for simulating a WF using WF

Simulator:

Specify the Test Record

Start the Simulator

Start the Simulation

Execute the Workflow

225. Specify the work records: In Client, create test records for simulation

Click Help → About Record to fetch the ROW\_ID of the record

Enter the above ROW\_ID in the Default String of Object Id Process Property in MVPW of the WF to be tested

When the WF is invoked in Production environment, the ROW\_ID will be passed automatically

Simulating locally requires the value to be passed explicitly

After successfully testing the WF, remove the default string and check in the WF

226. Start the simulator: Close all the instances of Siebel client

Right-click on the Designer → Select Simulate

Designer now opens a simulator window

Simulator Window is a read only version of WF

Background looks different when compared to the designer

227. Start simulation: Click the start button in the simulator toolbar

View → Toolbars → Simulation to display this toolbar

New Siebel client instance is launched

228. **Execute the workflow:** Simulate Next button allows step by step execution of WF

Complete Simulation button when clicked will execute WF steps till the end from the current step at once without stopping.

To check branching conditions, Simulate Next button to be used

Watch window can be used to inspect:

The values of the process properties at each step

Value of Fields in BC

Inspect the Watch Window to verify the values of the process properties

Values of the user added Process Properties can be edited during simulation

To complete the simulation:

Click on Simulate Next or Complete Simulation Button

Using the Watch window verify if the process properties contain correct values

Click the Stop button to Stop the simulation anytime during the simulation

Inspect the test record in client and verify the results

229. Workflow simulators considerations: Cannot simulate WF that invoke Server Components

Must simulate these directly on the thin client

Cannot simulate WF with run-time events on the start step

Can simulate WF having User interact step

Requires to perform the activity in the Client to proceed with simulation

230. This includes transferring of the Object definitions from Repository Tables to the Run-Time tables to make it available for use

It consists of completing the WF in the Tools

Click on Publish Button to Complete the WF

Status of the WF will now be Complete

Check in the Object definitions

Administrator will login to Server Environment to activate the Workflow in the run-time environment

231. **Publish workflow:** In Siebel Tool, click on the Publish button on the Workflow

Toolbar. It sets the status to Completed and prevents further editing of Workflows

It makes workflow available for activation

232. **Check-in workflow:** Check in the completed WF to the server repository

Siebel web client can now access the Workflow

233. **Activating the workflow:** To activate the Workflow:

Go to Administration – Business Process → Workflow Deployment

Query for the Workflow and Click Activate

The WF Process will be seen in Active Workflow Processes with Deployment Status As Active

234. **Publish/Activate button:** From Siebel Tools Workflows can be published and

activated at once

Click Publish/Activate Button on the toolbar

Sets the WF status to completed

Transfers the definitions to run-time tables from the repository tables

Client used for testing this WF should be same as the Siebel Tools Database

235. **Deployment consideration:** Publish/Activate all the child workflows first , i.e. Sub

Process, that are referenced by the deployed Workflow

These should be available for the Workflow to access

Compile any new repository object referenced in the deployed workflow

Example: BC, Fields, Views

236. **Revising workflows:** Workflows are versioned

Existing versions are kept and New versions are created

To Revise a Workflow:

Select the desired WF and check out

Click on Revise Button from toolbar

Creates a copy of the WF

Status = In Progress

Version = Incremented by 1

Edit, Test and Deploy the WF

Upon activating revised WF :

In Administration – Business Process ->

New Active version of the WF will be Active

Prior WF versions will be Outdated