

Leonardo Cappella (lcapella\_accenture@am.brasil.com.br) has a  
non-transferable license to use this Student Guide.  
**Siebel 8.1.x Tools**  
**Volume I • Student Guide**

D70458GC11

Edition 1.1

November 2009

D63749

**ORACLE®**

**Copyright © 2009, Oracle. All rights reserved.**

**Disclaimer**

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice. If you find any problems in the document, please report them in writing to: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA. This document is not warranted to be error-free.

**Restricted Rights Notice**

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

**U.S. GOVERNMENT RIGHTS**

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

**Trademark Notice**

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Leonardo Cappella (lcapella\_accenture@timbrasil.com.br) has a  
non-transferable license to use this Student Guide.

## **Table of Contents**

---

- Lesson 10 Business Components and Joins
- Lesson 11 Party Business Components
- Lesson 12 Business Components and Fields
- Lesson 13 Business Objects and Links
- Lesson 14 Creating a New Business Component Using a Standard 1:M Extension Table
- Lesson 15 Extending the Siebel Database
- Lesson 16 Additional Data Layer Configuration
- Lesson 17 Configuring Drilldowns and Applet Toggles
- Lesson 18 Configuring Picklists

# 10

## Business Components and Joins

ORACLE

Copyright © 2008, Oracle. All rights reserved.

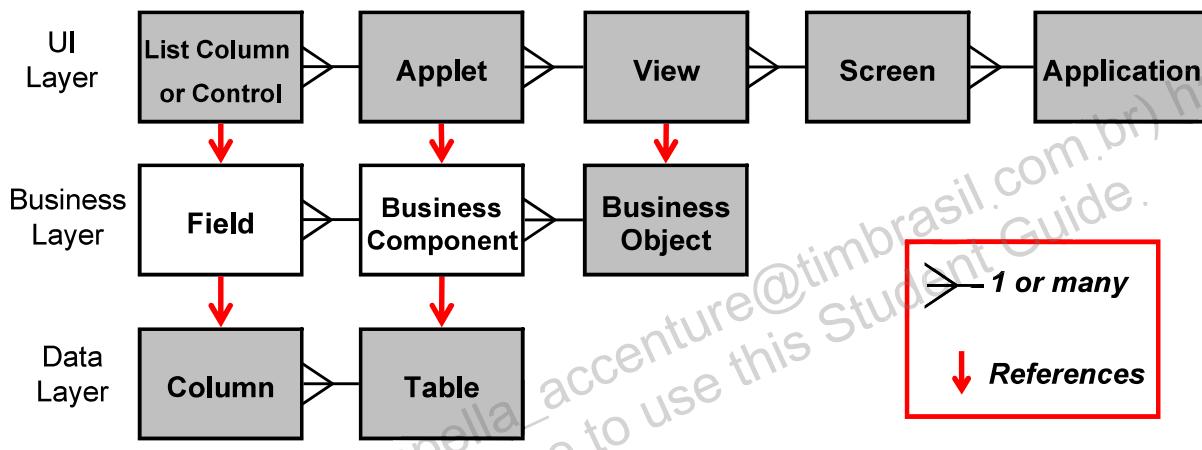
# Objectives

After completing this lesson you should be able to:

- Describe the structure of business components and joins
- Map fields in business components to base, joined, and extension tables
- Create a join and join specification to bring in data from a joined table
- Create a business component using the Business Component wizard

# Business Components

- Represent fundamental business entities in the user's world
- Store their data in one or more tables
- Are referenced by applets



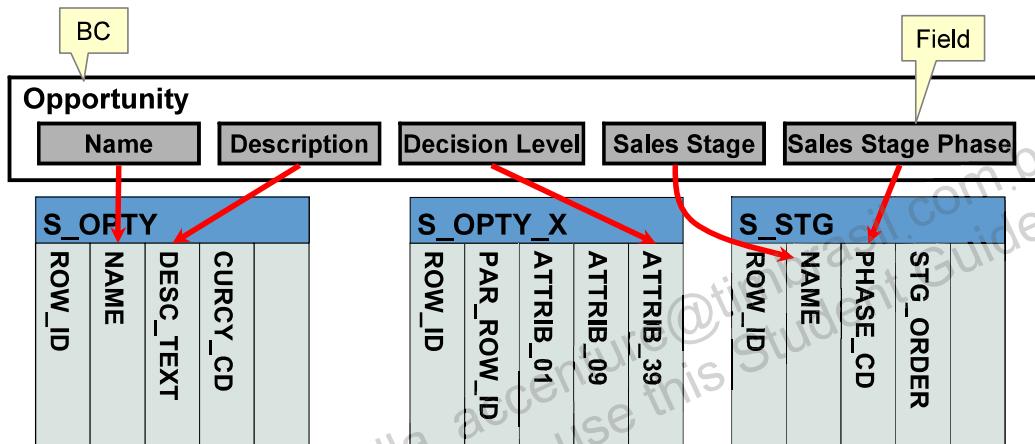
10 - 3

Copyright © 2008, Oracle. All rights reserved.

ORACLE

## Business Component (BC)

- Is a logical grouping of data from one or more tables
- Consist of several types of fields including single-value fields (SVF)
  - A single-value field references a single column in a table



## Business Component (BC)

In addition to single value fields, business components may also include multi value fields and calculated fields. These types of fields will be discussed in subsequent lessons. All fields can be accessed using Field type.

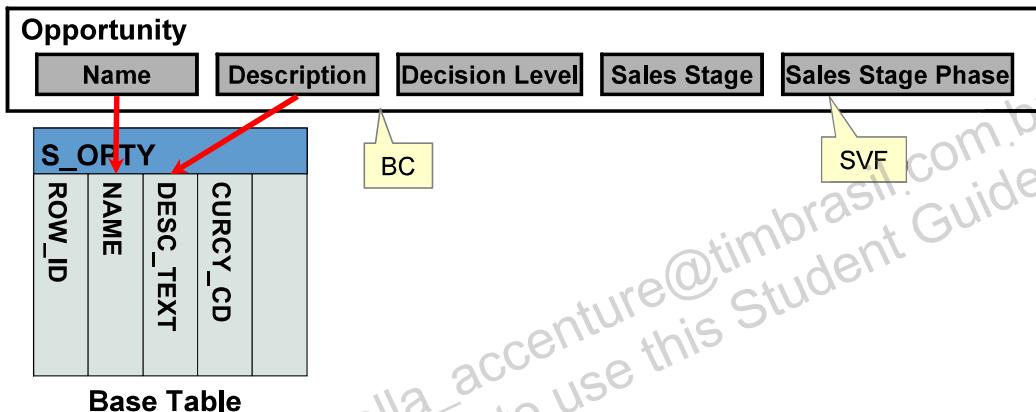
## Business Component Properties

- Table: Specifies the base table
- Class: Specifies the C++ class used at run time for the business component
  - Determines the behavior of the business component
  - Is set to CSSBusComp or a specialized child class
    - Many business components use specialized classes that support extra processing specific to that business component



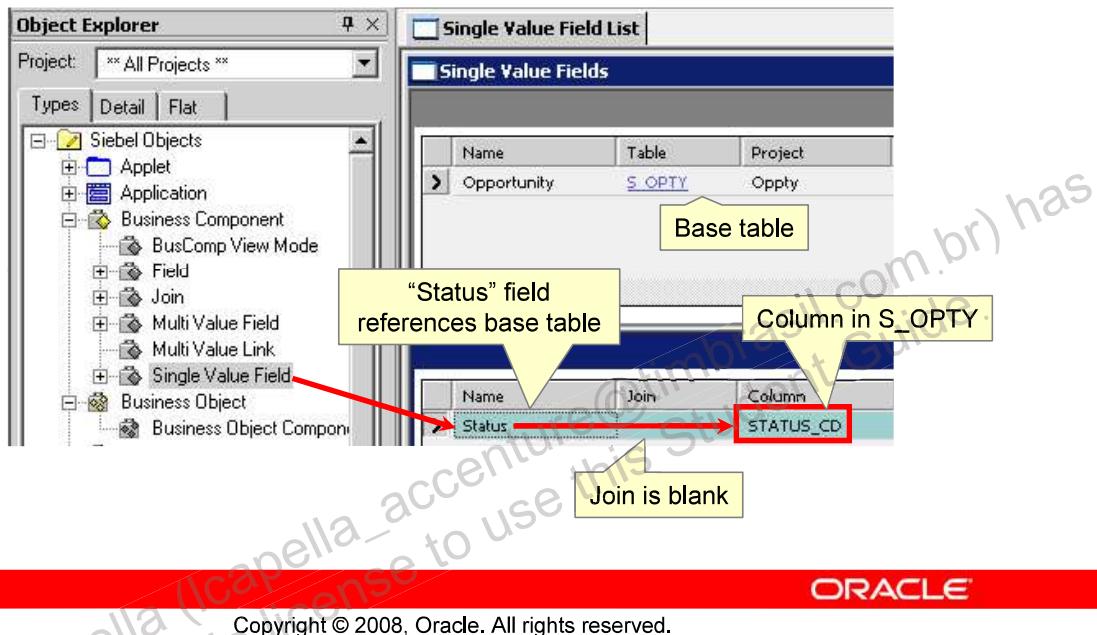
## Base Table

- Contains the columns that store the main fields for the business component
  - These fields can be edited unless:
    - The field references a system column
    - The field is designated as read only in the business component



## SVFs That Map to a Base Table

- Join property is blank
- Column property is a column in the base table

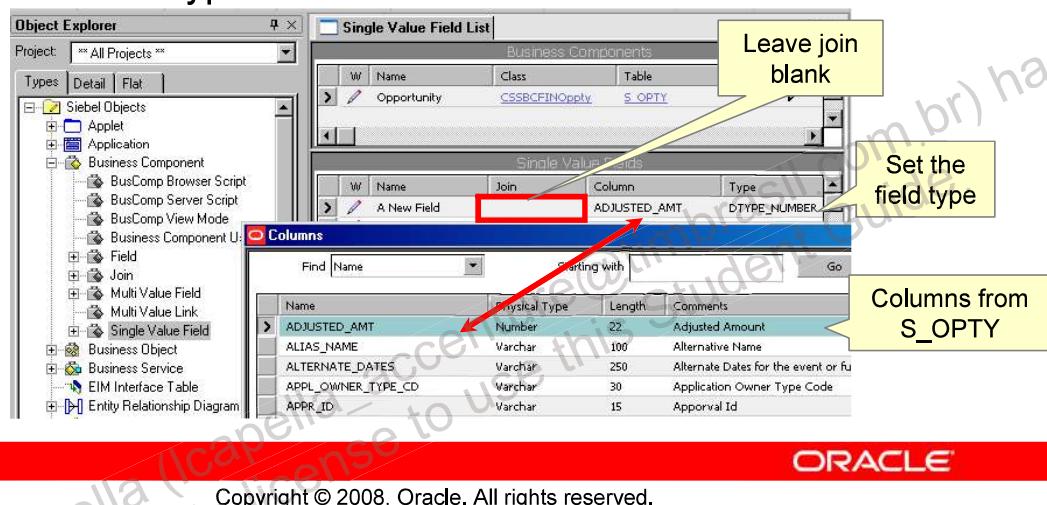


### SVFs That Map to a Base Table

Single value field object definitions can also be examined and edited in the Field object type. The Field object type shows both single and multi value fields, and includes properties that apply to both object types. You will learn about multi value fields in a later lesson.

## Mapping a Field to a Column in a Base Table

- Create a new Single Value Field definition
  - Leave the Join property blank
- Select the column to store the data
  - Column picklist shows columns from the base table
- Set the field type



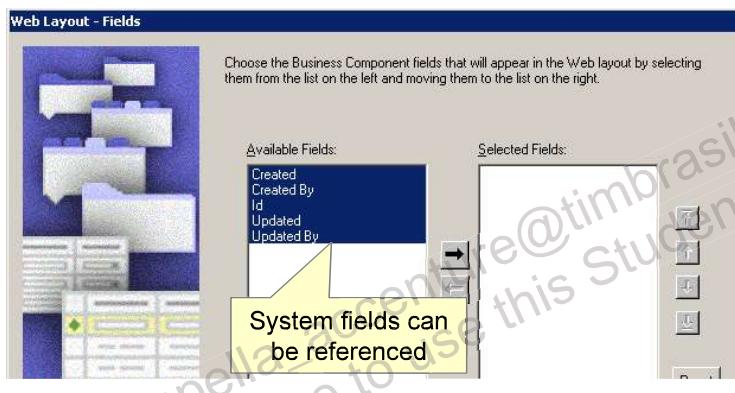
## Type Property for Fields

- Specifies the data type for the field
- Must correspond to the physical type of the column
- Consists of a set of Siebel-defined types
- Must be set by the configurator
  - Are defaulted automatically by Siebel Tools to DTYPE\_TEXT when fields are created



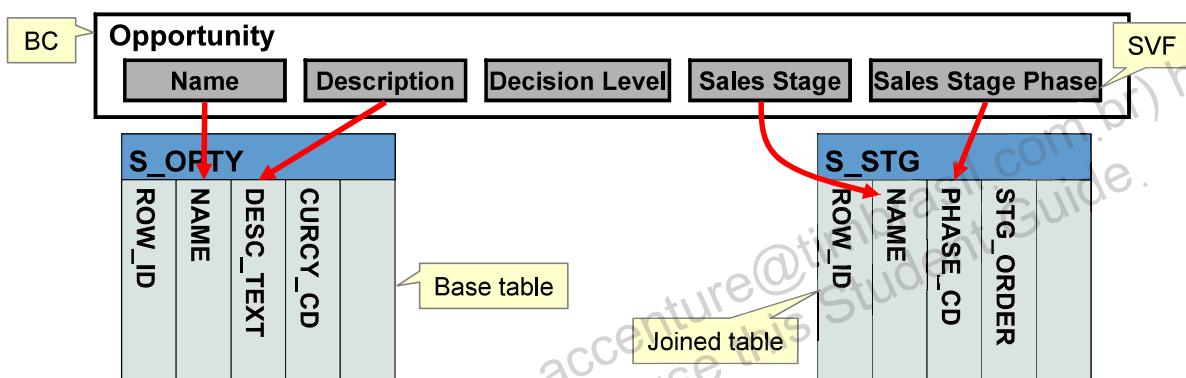
## Default Single Value Fields

- Each business component has a small set of fields that map to system columns such as ROW\_ID and CREATED
- These fields:
  - Do not appear explicitly as single value fields
  - Are available when configuring object definitions (such as applets) that map to the business component



## Joined Tables

- Business components can include data from additional related (*joined*) tables
  - For display in applets
  - For use in processing by the business component
- SVFs referencing columns in joined tables are read-only

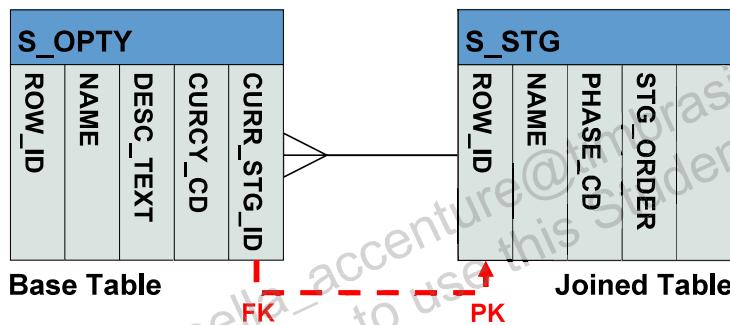


### Joined Tables

Reference: “Configuring Joins” in *Configuring Siebel Business Applications*

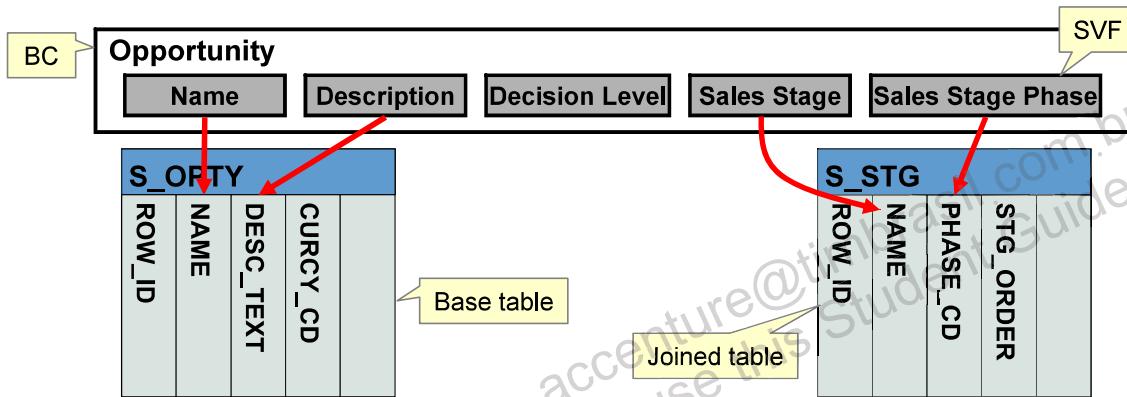
## Joining Data from Related Tables

- Returns only one row from the other (joined-to) table
- Is a relationship (M:1 or 1:1) from the BC to the related (joined-to) table
- Requires a:
  - Foreign key (FK) column on the base table that points to a primary key (PK) column on the joined-to table
  - FK field on the BC to expose the FK column



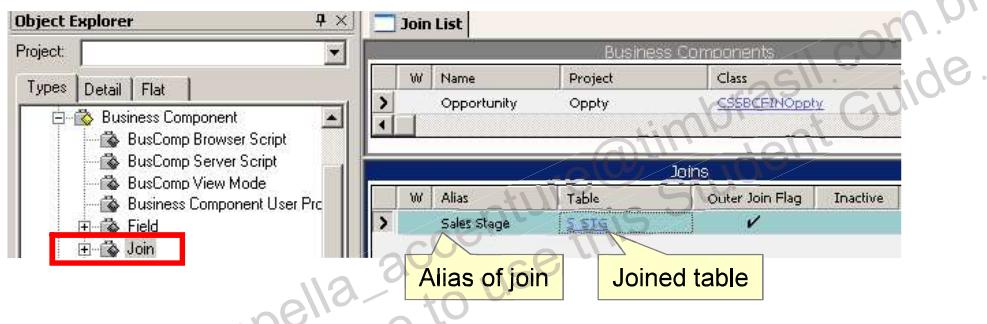
## Explicit Join

- Specifies the relationship from the BC to the joined table
- Includes a join definition and a join specification
- Is used to map single-value fields to a joined table



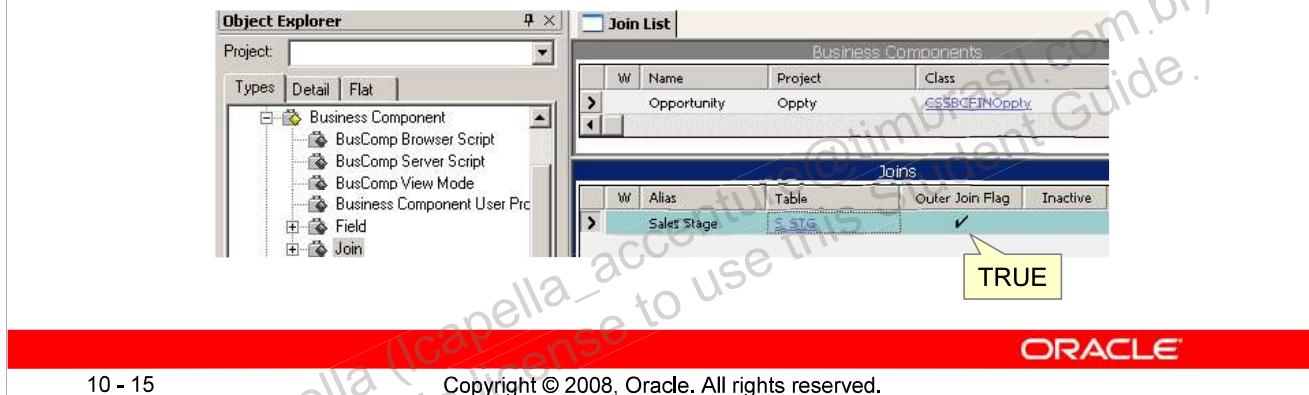
## Join Definition

- Is a child object of the business component
- Specifies the joined table from which to retrieve data
  - A given table can be joined multiple times
- Uses the alias property to distinguish multiple joins to the same table
  - Defaults the alias property to the name of the joined table
  - Modify alias when there are multiple joins to the same table



## Outer Join Flag

- If the Outer Join flag is set to TRUE, the join returns all records from the base table, even when there is no related row in the joined table
- Outer joins affect performance
  - If there will always be a related row in the joined table, an inner-join should be used
  - Business logic determines if an outer join is required



## Outer Join Example

- Opportunities that do not have a stage assigned will display or not display based on the Outer Join Flag setting

S_OPTY		
ROW_ID	NAME	CURR_STG_ID
111	Opp 1	2
222	Opp 2	1
333	Opp 3	

S_STG	
ROW_ID	NAME
1	Stage 1
2	Stage 2
3	Stage 3

No sales stage



Results with Outer Join TRUE

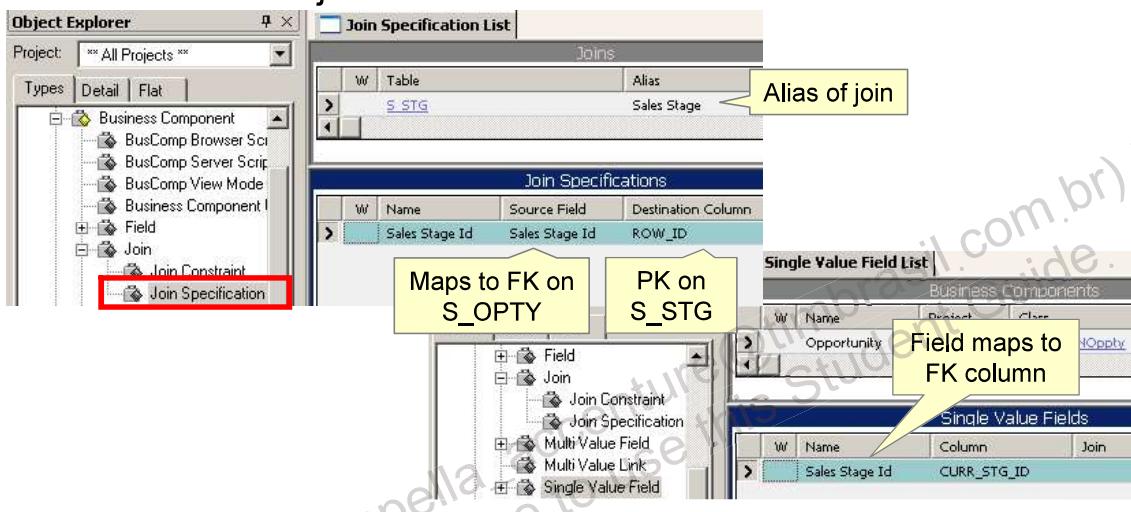
Opportunity	Stage
Opp 1	Stage 2
Opp 2	Stage 1
Opp 3	

Results with Outer Join FALSE

Opportunity	Stage
Opp 1	Stage 2
Opp 2	Stage 1

## Join Specification

- Specifies how to retrieve the related row from the joined table
  - Based on the foreign and primary keys used to relate the base and joined tables



## Fields That Map to a Joined Table

- Join property specifies the join object definition being referenced
- Column property maps to a column in the joined table



## Mapping a Field to a Column in a Joined Table

1. Determine if a join definition already exists
2. Create the required join if it does not exist
3. Create the single-value field



## 1. Determine if a Join Definition Already Exists

- Examine the join definitions for the business component
- Identify the joins that reference the desired table
- Verify that the join specification uses the desired relationship
  - There may be multiple relationships to the joined table

The screenshot shows the Siebel Object Explorer interface. In the left pane, under the 'Business Component' category, the 'Join' node is selected. In the right pane, the 'Join List' window is open, showing a table titled 'Joins'. The table has columns: W, Name, Class, Changed, and Project. A single row is visible for the 'Opportunity' business component, with the Class value being 'CSSBCFINOppty'. Below this table is another table titled 'Joins' with columns: W, Table, Changed, and Alias. This table lists several joins, with two specific ones highlighted by red boxes:

- A join to 'S\_ADDR\_PER' with the alias 'Primary Address'.
- A join to 'S\_ADDR\_PER' with the alias 'S\_ADDR\_PER'.

Yellow callout boxes point to these rows with the text: 'Uses Primary Address Id as FK' and 'Uses Statement Address Id as FK' respectively. The Oracle logo is visible at the bottom right of the application window.

W	Name	Class	Changed	Project
	Opportunity	CSSBCFINOppty		Opty

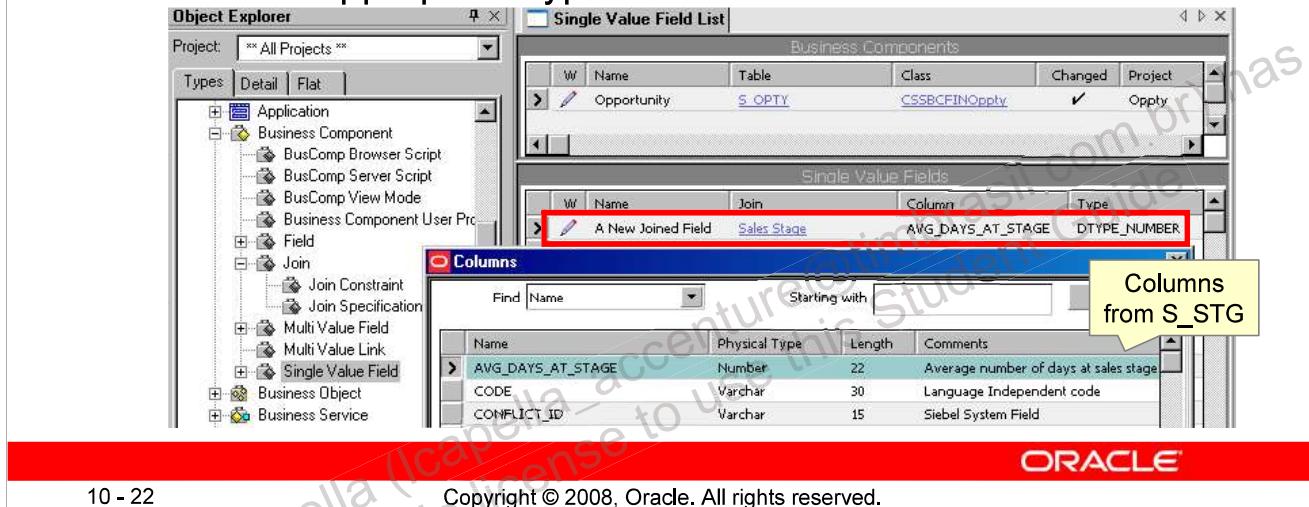
W	Table	Changed	Alias
	S_ADDR_PER		Primary Address
	S_ADDR_PER		S_ADDR_PER
	S_BU		Partner
	S_BU		Primary Organization
	S_BU		Primary Organization D
	S_CONTACT		Sales Manager
	S_CONTACT		Intermediary Contact
	S_CONTACT		Event Manager

## 2. Create a Join

- If the required join does not exist:
  - Verify the relationship is 1:1 or M:1
  - Verify there is a foreign key column for the relationship in the base table
  - Create a field in the business component to reference the foreign key column
  - Create the Join object definition
    - Set the Outer Join flag to TRUE if there will not always be a related record in the joined table
  - Create the Join Specification object definition

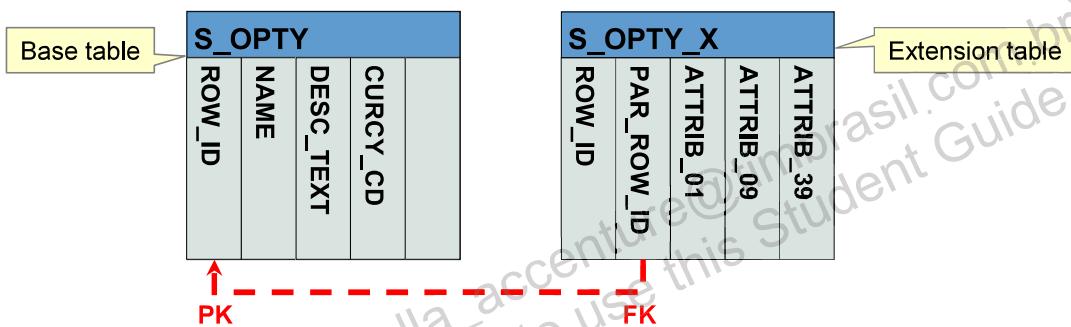
### 3. Create the Single Value Field

- Set the Join property
  - Select the alias for the desired join
- Then select the column in the joined table to be displayed
  - The column picklist shows the fields from the joined table
- Set the appropriate type



## 1:1 Extension Table

- Is a special table that has a 1:1 relationship with a base table
  - A FK column (PAR\_ROW\_ID) on the extension table is used to establish the relationship with the base table
- Base table and extension table columns can be considered to be one logical table

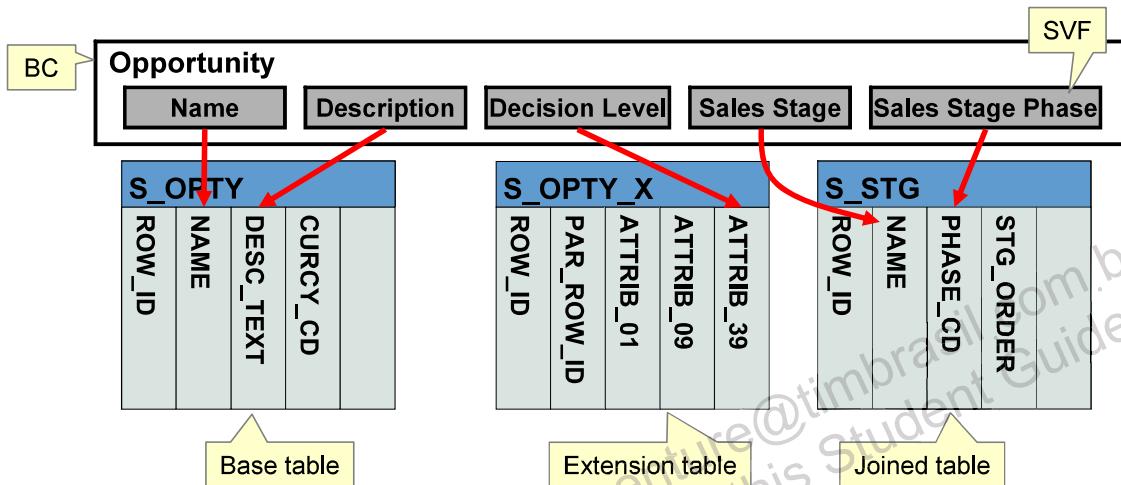


### 1:1 Extension Table

Reference: “Configuring Tables and Columns” in *Configuring Siebel Business Applications*

## 1:1 Extension Table

- Provides additional physical columns to store information for a business component



### 1:1 Extension Table

Many commonly-used base tables have standard 1:1 extension tables (such as the S\_OPTY\_X table illustrated above) with a large number of generic columns. In addition several tables for the Siebel industry applications have additional 1:1 extension tables to store fields specific to a Siebel industry applications. For instance the Siebel Hospitality application uses the S\_OPTY\_TNTX opportunity extension table.

## Implicit Joins

- Base tables are automatically able to join to their 1:1 extension tables.
  - Referred to as an implicit join
- Are created automatically for 1:1 extension tables
  - Name of an implicit join is always the name of the \_X table
  - Are available to all business components using the base table
- Do not appear as Join object definitions
- Do appear in the picklist for the Join property in an SVF.
- Are read/write
  - Allow joined fields in extension tables to be edited
- Always involve an outer join

## SVFs That Map to an Extension Table

- Join property specifies the extension table being referenced
- Column property references a column in the extension table
- Field is editable in the UI

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer tree is visible with nodes like Business Component, Field, Join, Multi Value Field, Multi Value Link, Single Value Field, and Business Object. In the center, there are two windows: 'Single Value Field List' and 'Single Value Fields'. The 'Single Value Fields' window displays a table of fields for the 'Opportunity' object, mapped to the 'S\_OPTY' extension table. One row is highlighted, showing 'Decision Level' as the name, 'S\_OPTY\_1' as the join, 'ATTRIB\_39' as the column, and 'DTYPE\_TEXT' as the type. The 'Single Value Field List' window shows a list of fields for the 'Opportunity' object, including 'Decision Level', 'Decision Level Score', 'Decision Orientation', and 'Deliver Flg'. The Oracle logo is at the bottom right of the interface.

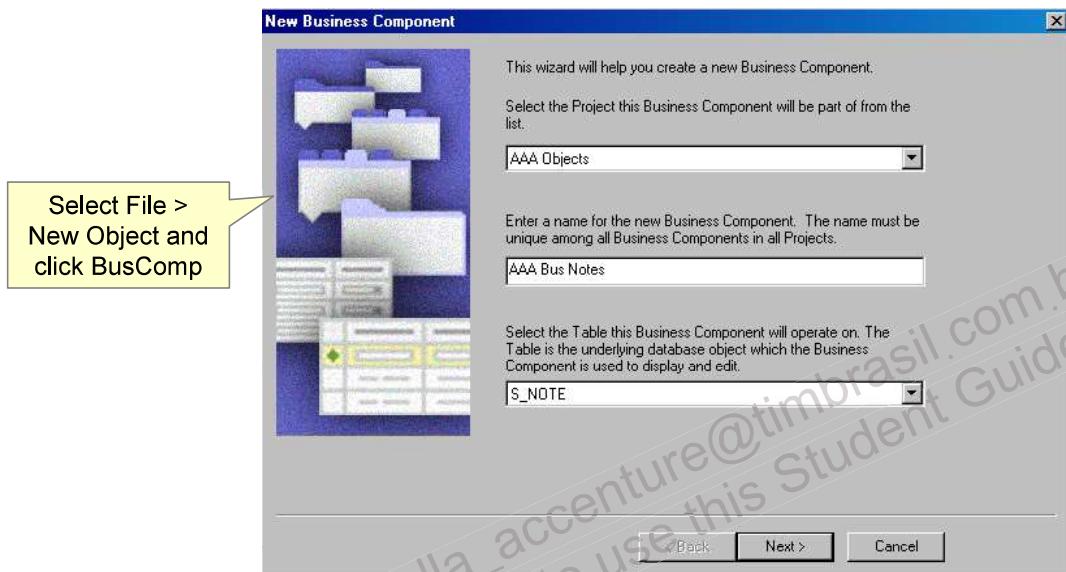
W	Name	Table	Class	Changed	Project
>	Opportunity	S_OPTY	CSSBCFINOppty	✓	Opty

W	Name	Join	Column	Type
>	Decision Level	S_OPTY_1	ATTRIB_39	DTYPE_TEXT
<	Decision Level Score			DTYPE_INTEGER
<	Decision Orientation	S_OPTY_GON	DEC_ORIENT_CD	DTYPE_TEXT
<	Deliver Flg		CLOSED_FLG	DTYPE_BOOL

## Creating a New Business Component

- Use the business component wizard to create a new business component



10 - 27

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### Creating a New Business Component

It is good practice to append a project-specific prefix (such as AAA) to all object definitions that you create. This allows you to easily distinguish your newly created object definitions from those delivered in the Siebel repository.

## Business Component Wizard Inputs

- Project to contain the new business component
- Name of the business component
- Base table
- One or more fields and the corresponding columns in the base table to which each maps

## Business Component Wizard

- Creates the following:
  - A business component based on the CSSBusComp class
  - Fields with types corresponding to the related table column type
    - Default values are used for all other properties

The screenshot shows the Siebel Object Explorer and the Single Value Field List windows. In the Object Explorer, under the 'Types' tab, a 'Business Component' node is expanded, showing sub-items like 'BusComp Browser Script', 'BusComp Server Script', etc. In the Single Value Field List window, there are two tabs: 'Business Components' and 'Single Value Fields'. The 'Business Components' tab shows a single row for 'AAA Bus Notes' with columns: Name (S\_NOTE), Table (S\_NOTE), Class (CSSBusComp), and Project (AAA Objects). The 'Single Value Fields' tab shows three rows of fields with columns: Name (Note, Note Type, Priv Flg), Column (NOTE, NOTE\_TYPE, PRIV\_FLG), and Type (DTYPE\_NOTE, DTYPE\_TEXT, DTYPE\_BOOL).

W	Name	Table	Class	Project
>	AAA Bus Notes	S_NOTE	CSSBusComp	AAA Objects

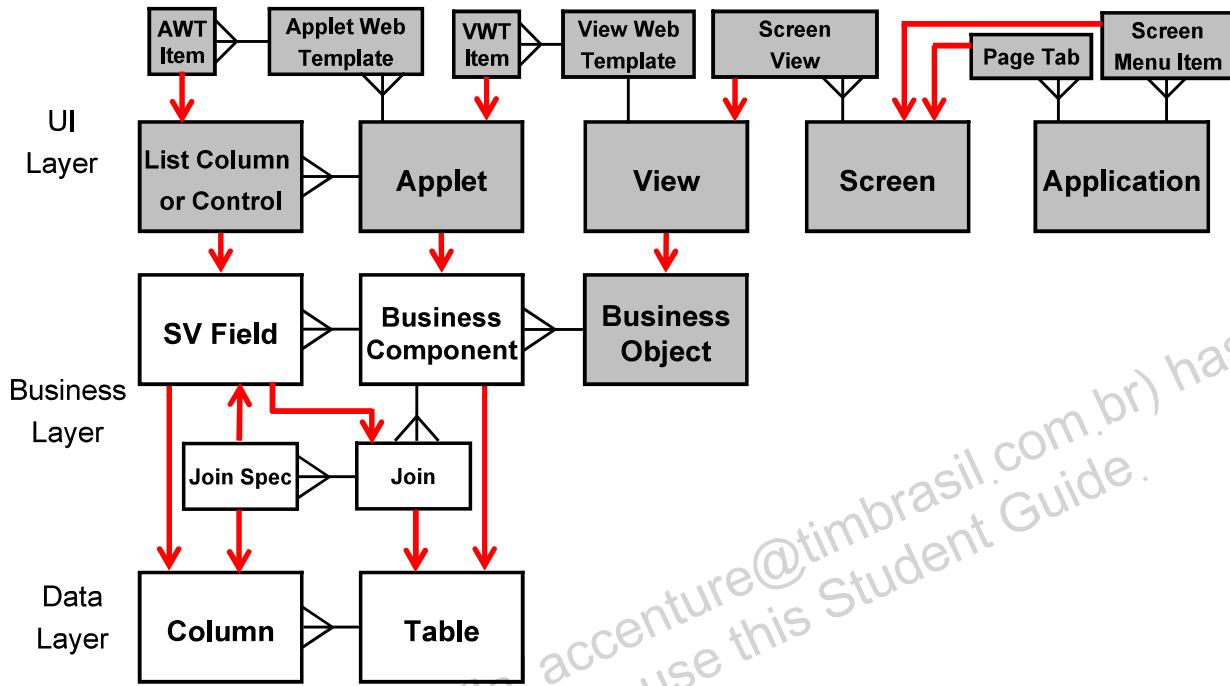
  

W	Name	Column	Type
>	Note	NOTE	DTYPE_NOTE
	Note Type	NOTE_TYPE	DTYPE_TEXT
	Priv Flg	PRIV_FLG	DTYPE_BOOL

## Configuring a New Business Component

- Create additional fields that map to the base table or 1:1 extension tables (if any)
- Create joins as required to other tables and map new fields
  - Remember a foreign key to the other table must exist in the base table

## Summary of Object Types



## Lesson Highlights

- A business component (BC) represents a fundamental business entity in the user's world
- A BC references a single base table
  - Fields can map directly to columns in a base table
- A BC can include data from joined tables
  - A join definition specifies the related table
  - A join specification specifies how to get the data from the joined table
  - Fields can map to columns in a joined table by referencing an explicit join
- A BC can include data from an extension table
  - Fields can map to columns in an extension table by referencing an implicit join

## Practice 10 Overview: Business Components and Joins

This practice covers the following topics:

- Examining Single Value Field Mappings
- Creating a Join

Unauthorized reproduction or distribution prohibited. Copyright© 2014, Oracle and/or its affiliates.

Leonardo Cappella (lcapella\_accenture@timbrasil.com.br) has a  
non-transferable license to use this Student Guide.

# 11

## Party Business Components

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Objectives

After completing this lesson you should be able to:

- Describe the role of S\_PARTY and its extension tables in storing party business component data
- Create joins and join specifications to bring in party data
- Map fields to S\_PARTY extension tables

## Party Data

- Refers to data for enterprise applications that describe:
  - The company itself and the way in which it is organized
  - Employees and other users of the company's application(s)
  - Customers, partners, and other enterprises associated with the company
  - Contacts and other people outside the company
- Can refer to both individual people and groups

## Party Business Components

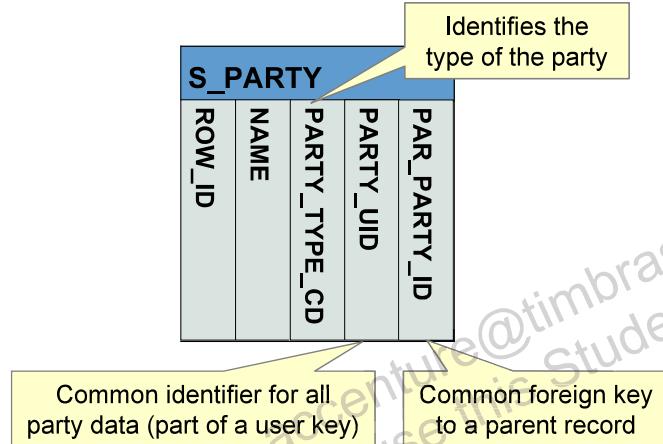
- Are business components in the Siebel Data Model used to represent party data
  - Examples:
    - Account
    - Position
    - Contact
    - Organization
    - And so forth
- Differ from regular Siebel business components:
  - Use a special table S\_PARTY as their base table
  - Store their main data in a small set of S\_PARTY extension tables

### Party Business Components

Reference: “Configuring Business Components” in *Configuring Siebel Business Applications*

## **S\_PARTY**

- Is the base table for all party business components
- Consists of a small number of data columns
  - Store party data that is common to all instances of party business components



11 - 5

Copyright © 2008, Oracle. All rights reserved.

**ORACLE**

### **S\_PARTY**

**Reference:** “Configuring Tables and Columns” in *Configuring Siebel Business Applications*

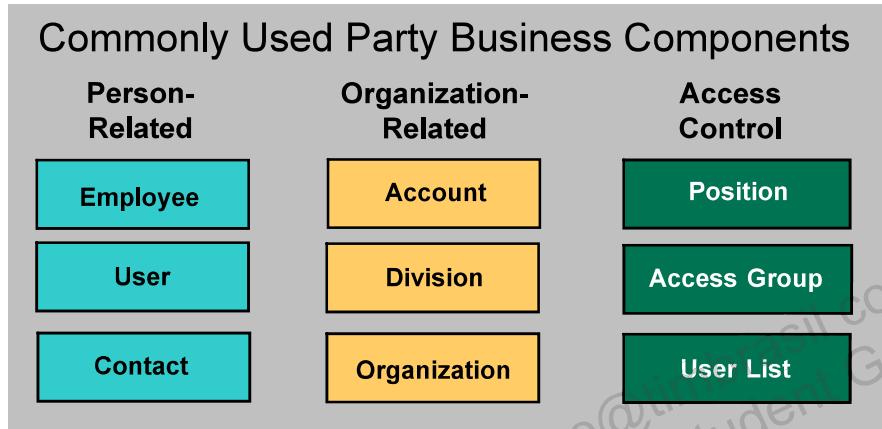
The Siebel party model supports the following party types: AccessGroup, Household, Organization, Person, Position, and UserList.

## Benefits of the Party Data Model

- Allows all instances of party business components to be treated similarly in certain respects
  - For example they all have a common primary key and user keys
- Provides a way to relate instances of different party business components
  - For example an access group of positions and accounts can be created

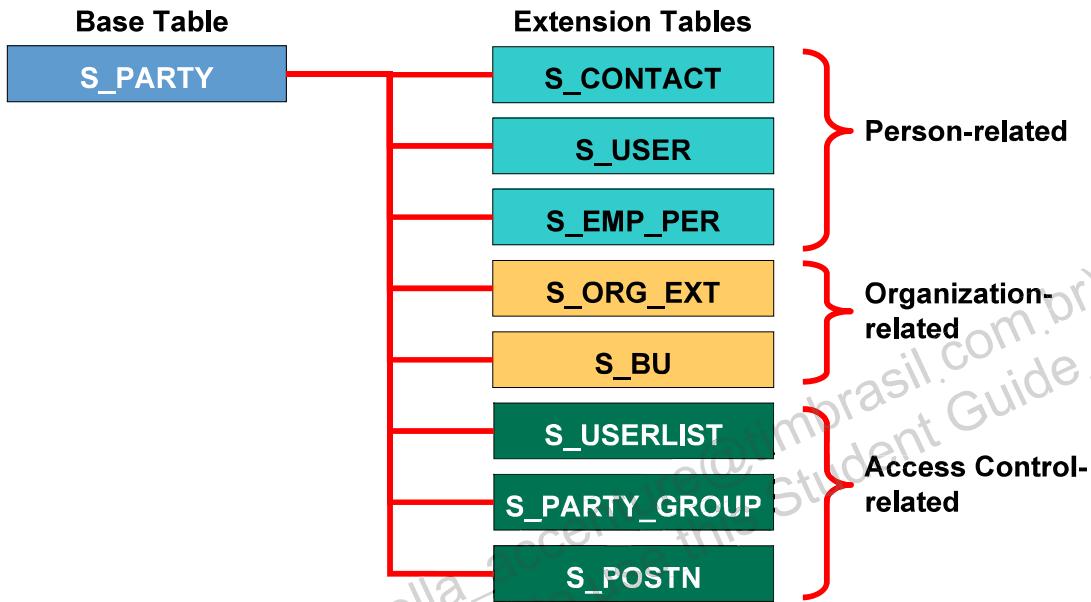
## Commonly Used Party Business Components

- Represent a variety of entities that can be arranged into groups related to persons, organizations, or access control



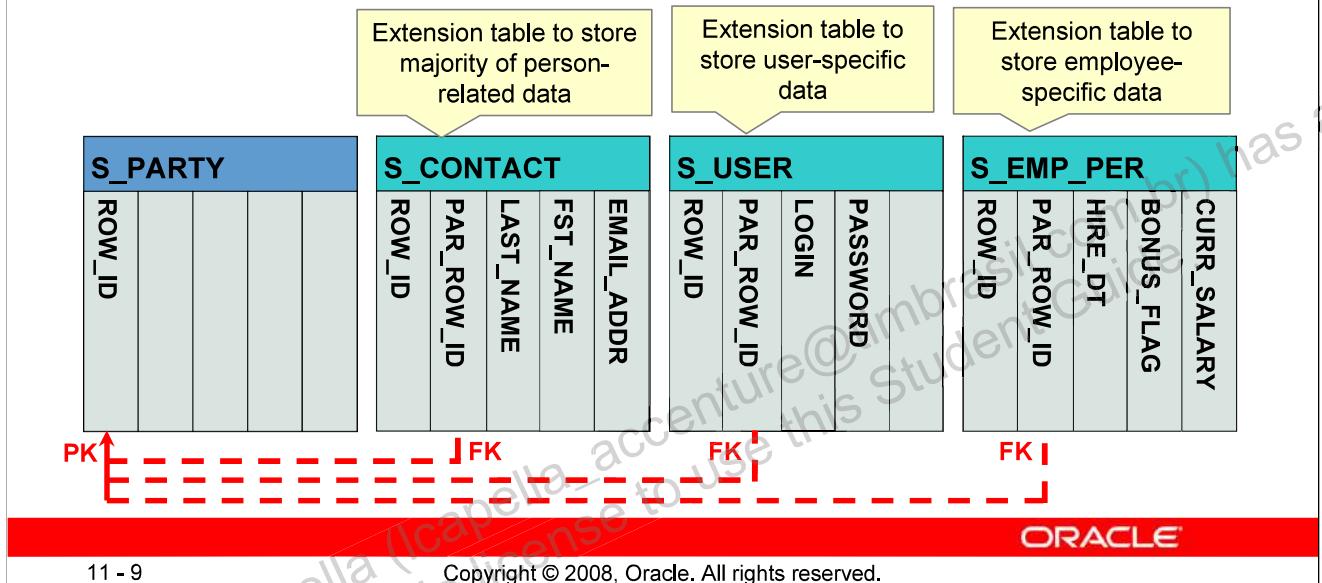
## S\_PARTY and Its Extension Tables

- Eight prominent S\_PARTY extension tables store the data



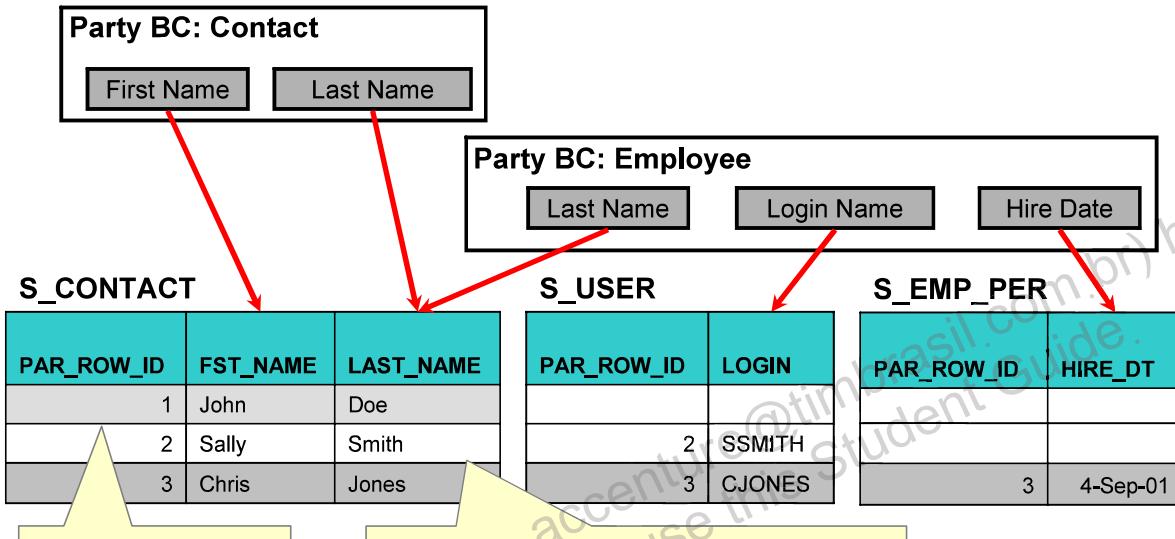
## Person-Related Party Business Components

- Store their main data in S\_CONTACT
- May store additional data in S\_USER and S\_EMP\_PER
  - All tables are extension tables of S\_PARTY



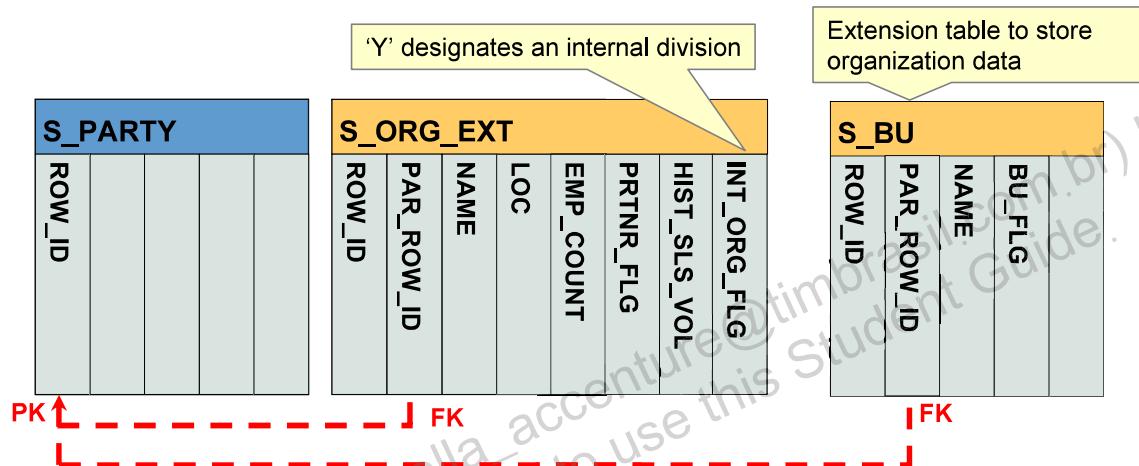
## Person-Related Party Business Components

- Multiple person-related business components use these tables.



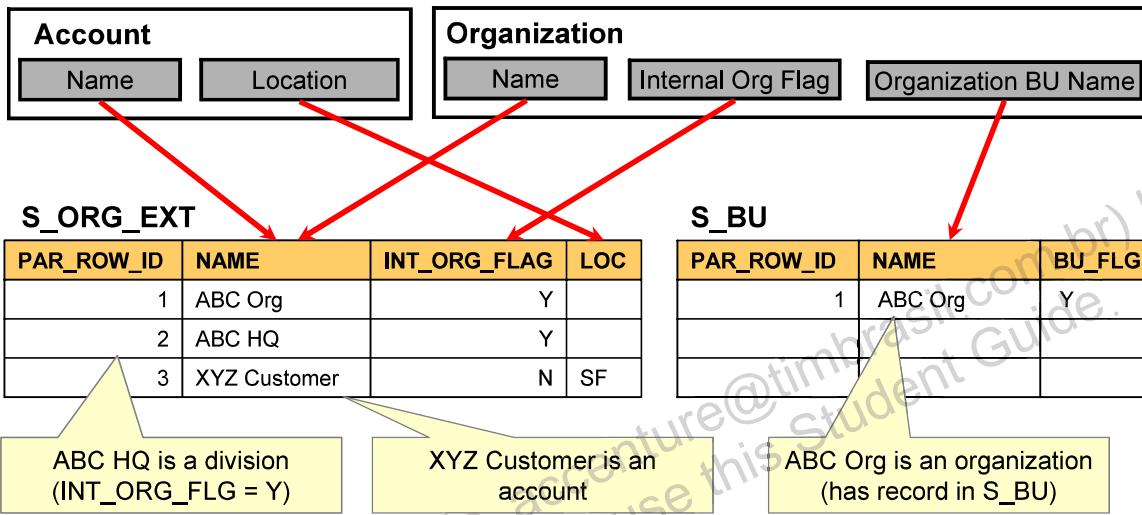
## Organization-Related Party Business Components

- Store their main data in S\_ORG\_EXT
- Organization business components store additional data in S\_BU
  - S\_BU supports indexing on Organization Name



## Organization-Related Party Business Components

- Multiple organization-related business components use these tables.



## Party Business Components for Access Control

- Represent groupings of party instances
  - User List
  - Access Group
  - Position

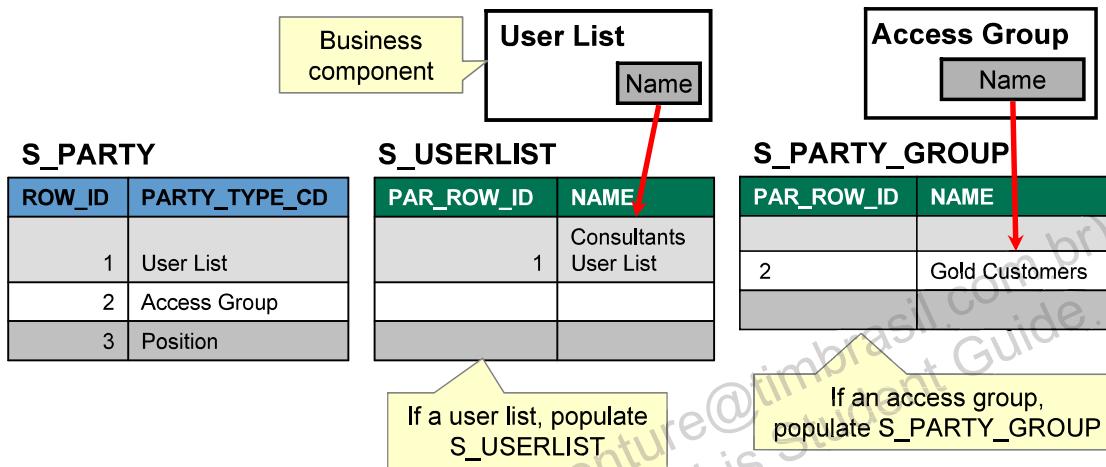
S_USERLIST		
ROW_ID	PARTY_ROW_ID	NAME

S_PARTY_GROUP		
ROW_ID	PARTY_ROW_ID	NAME

S_POSTN		
ROW_ID	PARTY_ROW_ID	NAME
POSTN_TYPE_CD		

## Groupings for Access Control

- Access Group and User List represent groupings of party records



## S\_PARTY\_PER

- Is an intersection table that relates two instances of parties
- Is used to implement relationships between
  - User lists and users
  - Access groups and members.

S\_PARTY\_PER

PARTY_ID	PERSON_ID
003	001
003	002

S\_PARTY

ROW_ID	PARTY_TYPE_CD	NAME
001	Person	Smith, Sally
002	Person	Doe, John
003	User List	Consultants User List

FK

FK

PK

ORACLE

## Role of Joins in Party Data

- Implicit joins are used to map fields in party business components to the party extension tables
- Explicit joins are used to bring data in a party extension table into:
  - A non-party business component
  - Another instance of a party business component

11 - 16

Copyright © 2008, Oracle. All rights reserved.

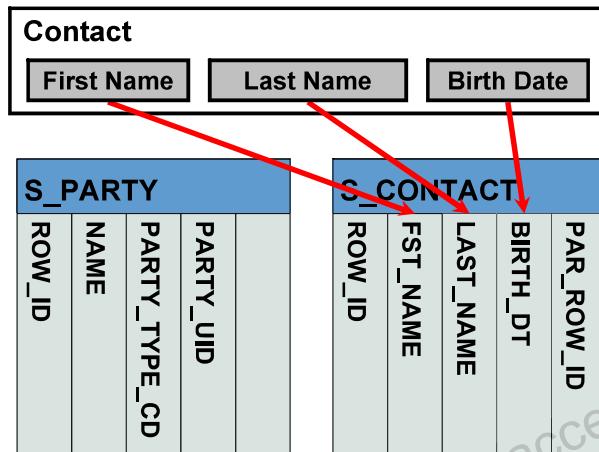
ORACLE

### Role of Joins in Party Data

Reference: “Configuring Joins” in *Using Siebel Tools*

## Mapping Fields in Party Business Components

- Main fields are mapped using the implicit join for the extension table
  - Name of join is the name of the table



Business Components		
Name	Table	Class
> Contact	S_PARTY	CSSBCUser
Single Value Fields		
Name	Join	Column
Last Name	S_CONTACT	LAST_NAME
Last Name, First N:	S_CONTACT	LAST_UPD
Login Name	S_USER	LOGIN
Login Password	S_CONTACT_X	ATTRIB_03
M/F	S_CONTACT	SEX_MF
M/M	S_CONTACT	PER_TITLE
Mail Stop	S_CONTACT	OU_MAIL_STOP

Main columns use the implicit join to S\_CONTACT

## Joining Party Data Into a Business Component

- Data in a party table can be joined into a non-party business component.
  - Example: Bringing account data into the Opportunity business component for display in an Opportunity applet

The screenshot shows the Siebel Opportunities List interface. At the top, there's a navigation bar with links like Home, Accounts, Contacts, Opportunities, Quotes, Sales Orders, and Service. Below that is a toolbar with buttons for New, Delete, and Query, and a status bar showing '1 - 10 of 18+'. The main area is a grid titled 'My Opportunities' with columns: New, Opportunity Name, Account, and Revenue. Three rows of opportunities are listed, each with an asterisk icon. The 'Opportunity Name' column contains: 'Fast Ethernet NIC PCI 10/100 - 2500 units', 'Digi Phones for mobile field engineers', and 'Pentium Server for new web sites'. The 'Account' column contains: 'Marriott International HQ', 'Economy Printing & Copying', and 'RS Semiconductors'. The 'Revenue' column shows values: '\$687,500.00', '\$500,000.00', and '\$250,000.00'. A red box highlights the 'Opportunity Name' and 'Account' columns. A yellow callout box labeled 'Joined party data' points to the 'Account' column. Below the grid, a box labeled 'Non-Party BC: Opportunity' contains four buttons: Name, Description, Account, and Account Location. Red arrows point from the 'Account' column in the grid to the 'Account' button in the Non-Party BC box.

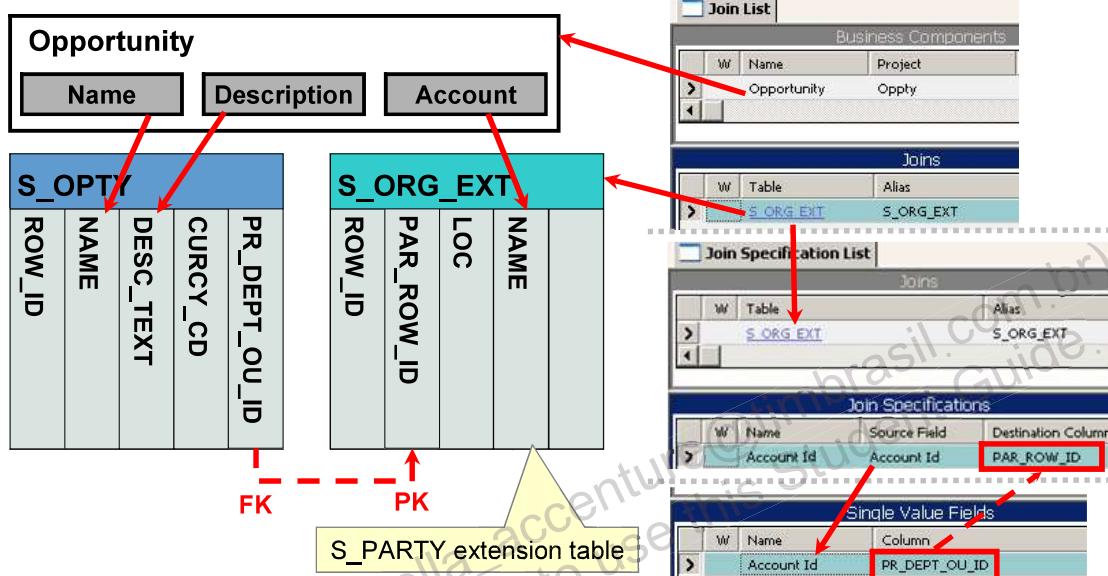
New	Opportunity Name	Account	Revenue
*	Fast Ethernet NIC PCI 10/100 - 2500 units	Marriott International HQ	\$687,500.00
*	Digi Phones for mobile field engineers	Economy Printing & Copying	\$500,000.00
*	Pentium Server for new web sites	RS Semiconductors	\$250,000.00

Non-Party BC: Opportunity

Name    Description    Account    Account Location

## Explicit Join Definition

- References the extension table that contains data of interest



11 - 19

Copyright © 2008, Oracle. All rights reserved.

ORACLE

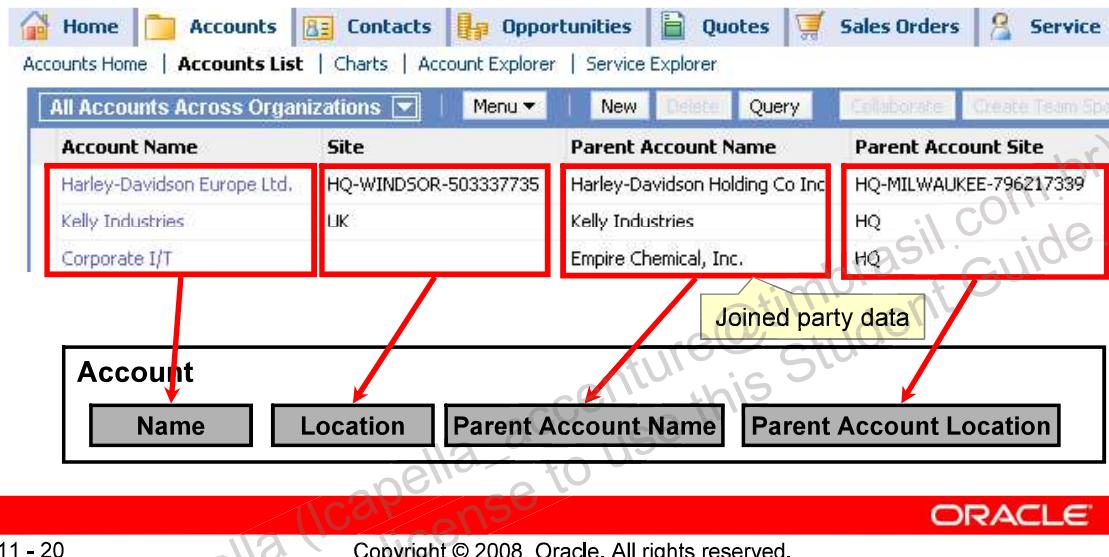
### Explicit Join Definition

#### PAR\_ROW\_ID

- Join specifications usually reference the primary key column (ROW\_ID) of the destination (joined) table. However for joined tables for party business components the destination column is set to PAR\_ROW\_ID.
- Later in the course you will learn that the foreign key column PR\_DEPT\_OU\_ID is populated with the primary key of the associated account and that value will be ROW\_ID of the S\_PARTY table. The column in any extension table that is guaranteed to contain the value of the ROW\_ID of the base table is PAR\_ROW\_ID. Hence the join specification must use PAR\_ROW\_ID as the destination column.

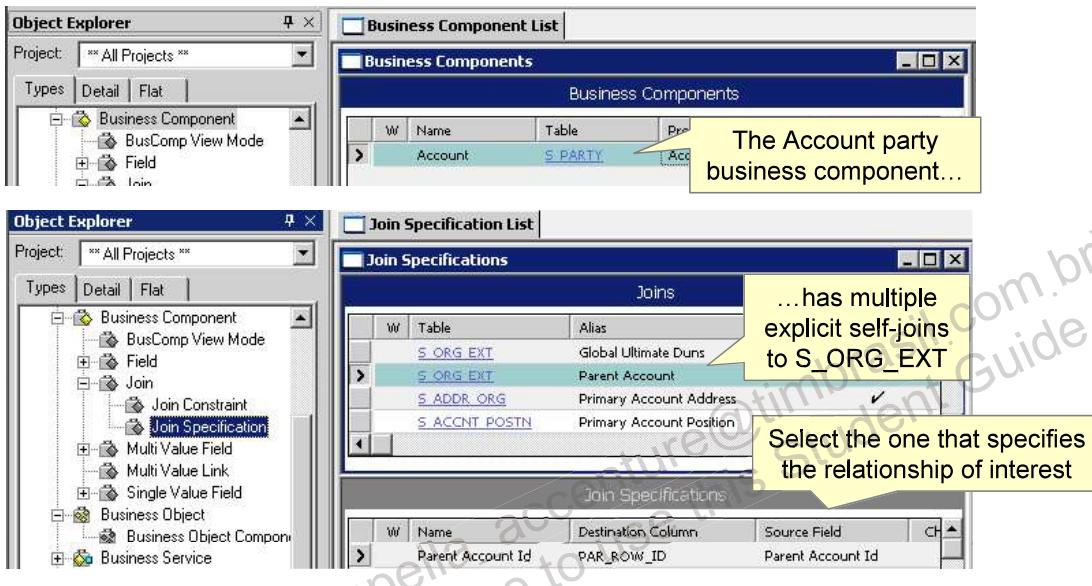
## Joining Party Data into a Party Business Component

- Uses an explicit join to the party table, and not the implicit one
  - Example: Bringing parent account data into the Account business component for display in an Account applet



## Explicit Join Definition

- Select or create an explicit join to the desired S\_PARTY extension table.



11 - 21

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### Explicit Join Definition

When you are joining party data into a party business component, you must be careful not to select the implicit join to the table of interest. Remember that the source business component also uses the party extension table. The implicit join returns the data associated with the source business component instance, and not the related business component instance

## Mapping a Field to a Column in a Party Table

- Determine if a join definition already exists
- Create the required join if it does not exist
- Create the single-value field

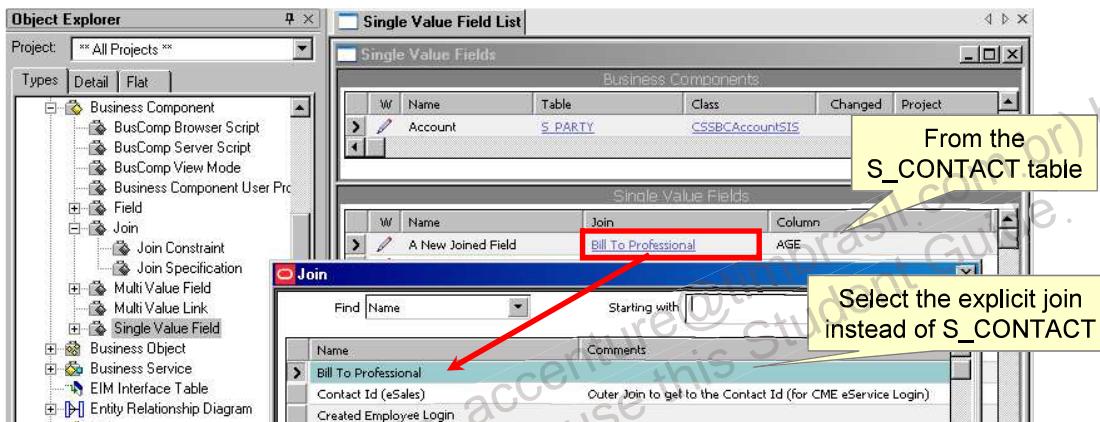


## Creating a Join to a Party Table

- Verify that the relationship is 1:1 or M:1
- Identify the foreign key column for the desired relationship to the joined table
- Create, if necessary, a single value field in the business component to reference the foreign key column
- Create the join
  - Assign an appropriate alias property
- Create the join specification
  - Use the foreign key field for the source
  - Use *PAR\_ROW\_ID* for the destination column

## Create the Single-Value Field

- Select the appropriate explicit join
  - *Do not use the implicit join to the table*
- Select the desired column in the joined table
- Set the appropriate type



## Key Considerations for Party Joins

- When mapping fields in party business components
  - Use the implicit join for the extension table
- When bringing party data into a non-party business component
  - Create the join specification based on PAR\_ROW\_ID
- When bringing party data into a party business component
  - Use the appropriate explicit join

## Lesson Highlights

- Party BCs are used to represent a wide range of party data
  - All party BCs use S\_PARTY as the base table but store their data in S\_PARTY extension tables
- Person-related party BCs store data in S\_CONTACT
- Organization-related party BCs store data in S\_ORG\_EXT
- Implicit joins are used to map fields in party business components to the party extension tables
- Explicit joins are used to bring data in a party extension table into other business components
  - Join specification must reference PAR\_ROW\_ID in the party extension table

## Practice 11 Overview: Party Business Components

This practice covers the following topics:

- Examining Fields in Party Business Components
- Creating a Join to a Party Business Component Table

# 12

## Business Components and Fields

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Objectives

After completing this module you should be able to:

- Edit business component properties to capture business logic
- Edit field properties to capture business logic

## Business Component Properties

- Determine the behavior of business components
  - Implement business logic
- Include:
  - Properties for editing
    - No Delete
    - No Insert
    - No Merge
    - No Update
  - Owner Delete
  - Search Specification
  - Sort Specification

Properties	
Business Component [Account]	
Alphabetic Categorized	
Name	Account
No Delete	FALSE
No Insert	FALSE
No Merge	FALSE
No Update	FALSE
Object Language Locked	
Object Locked	FALSE
Object Locked By Name	
Object Locked Date	
Owner Delete	TRUE
Scripted	FALSE
Search Specification	([Internal Org Flag] <> 'Y' or
Sort Specification	Name(ASCENDING), Location
Table	S_PARTY



ORACLE

## Business Component Properties

Reference: “Configuring Business Components” in *Configuring Siebel Business Applications*

## Business Component Editing Properties

- Set editing properties in the business component to prevent deleting, inserting, merging, and updating records
  - Example: To maintain record continuity, users cannot delete or change price lists once they have been created
- Applies to all interactions with a business component including applets, workflows, and so forth



## Applet Editing Properties

- These editing properties also appear on applets
- Set the business component editing properties to FALSE
  - Allows some applets to be read only (set applet editing properties = TRUE) and others to allow editing (set applet editing properties = FALSE)

Business Component

Properties	
Business Component [Contact]	
Alphabetic Categorized	
Name	Contact
No Delete	FALSE
No Insert	FALSE
No Merge	FALSE
No Update	FALSE
Object Language Locked	

Setting properties for a business component  
affects all applets referencing this BC

Applet

Properties	
Applet [Contact Form ReadOnly Applet]	
Alphabetic Categorized	
Name	Contact Form ReadOnly Applet
No Delete	TRUE
No Insert	TRUE
No Merge	TRUE
No Update	TRUE
Object Language Locked	

...can be customized for  
each applet

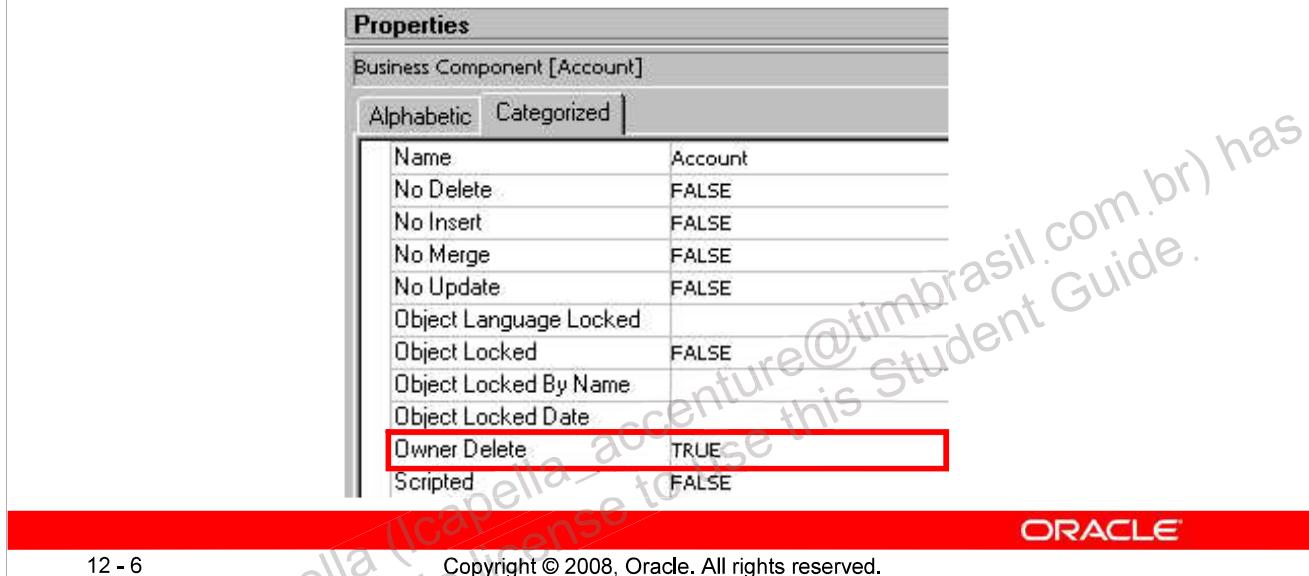
ORACLE

## Applet Editing Properties

The editing properties on business components are ignored in a set of special views in which the (view) admin mode flag is set to TRUE. These views are typically reserved for administrators and allow them to create, edit, and delete instances of business components that regular users can only read.

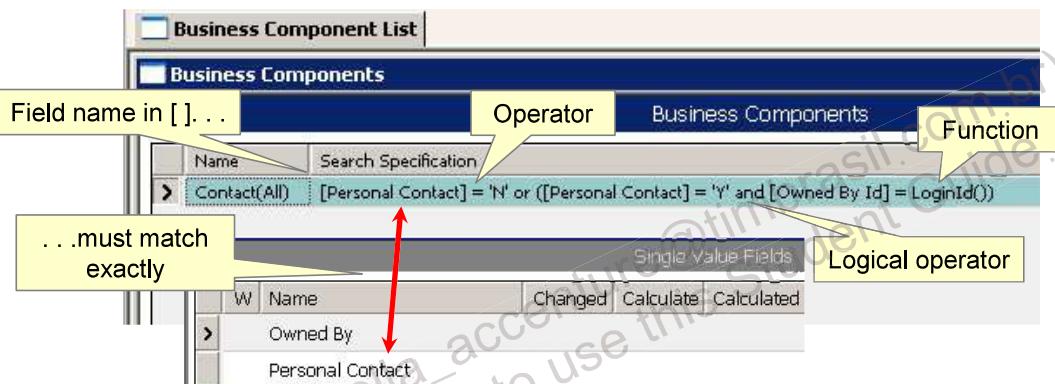
## Owner Delete Property

- Set the Owner Delete property to TRUE to enable only the owner of a record to delete the record
  - For team-based business components the owner of the record is the primary on the team



## Search Specification Property

- Specifies a subset of records to be retrieved by the business component
  - Consists of field names, constants, functions, logical operators, and comparison operators
  - Typically used when there are multiple business components based on the same table



12 - 7

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### Search Specification Property

The search specification illustrated in the slide retrieves contacts for which:

- the Personal Contact flag is “N” OR
- the Personal Contact flag is “Y” and the owner of the contact is the user logged in to application

This retrieves all the regular (non personal) contacts as well as the personal contacts associated with the user.

## Search Specification Considerations

- Search specifications appear on applets as well
  - A search specification on the applet is combined (using an AND) with a search specification on the BC to generate the WHERE clause in the resulting SQL statement
    - Predefined queries will also contribute to the WHERE clause
- Beware of mutually exclusive search specifications

## Sort Specification Property

- Determines the sort order of the retrieved records
  - Use (DESC) or (DESCENDING) to sort field in reverse order
  - Cannot be set at the applet level

To sort by last name, then first name...

Must match exactly

Columns for fields in sort spec

Ensure that an index exists to support the sort spec; (index columns must be in same order as the sort spec)

ORACLE

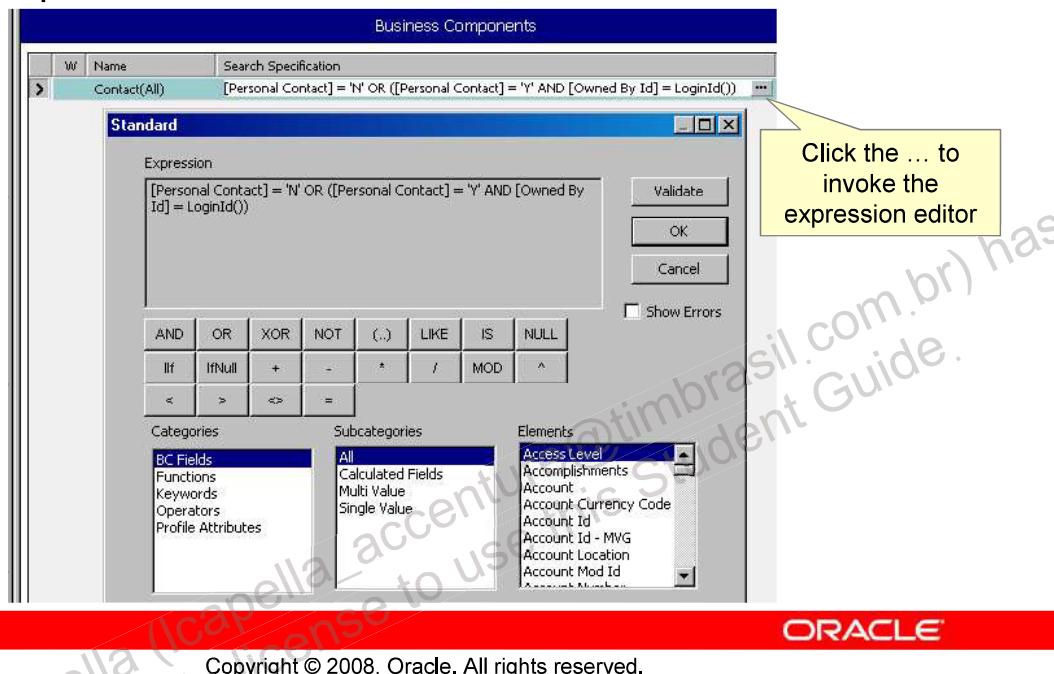
### Sort Specification Property

A sort specification should have a corresponding index to allow the records to be retrieved and sorted efficiently, as indexes permit the records to be retrieved in sorted order. If there is no corresponding index, retrieval of the records will take much longer as records have to be sorted on-the-fly as they are being retrieved.

Users can select the sort order for records displayed in a list applet by clicking on column headings for sort-enabled columns.

## Expression Builder

- Use the expression builder to examine and edit search and sort specifications



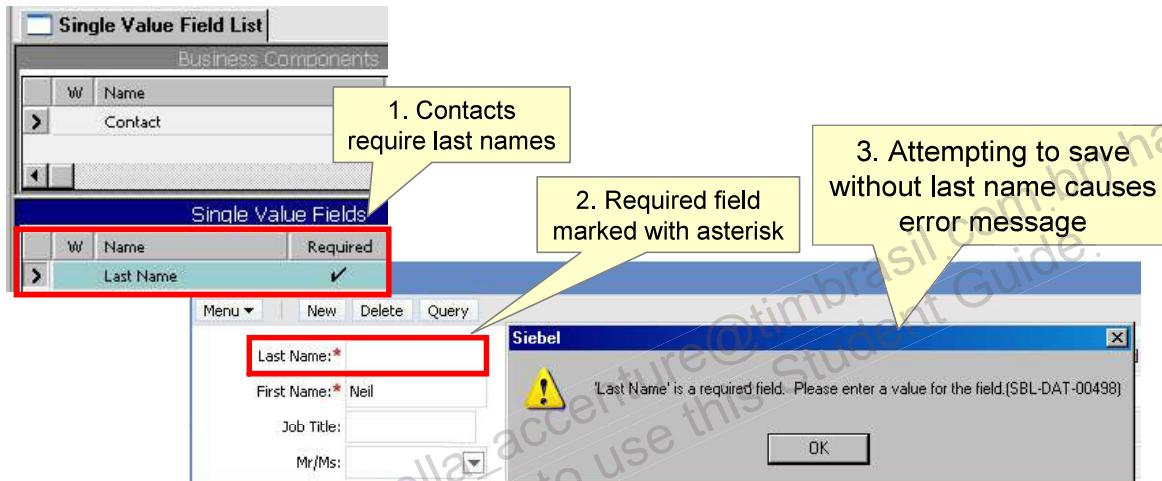
## Business Component Field Properties

- Fields can be configured to implement desired business logic using the following field properties:
  - Required
  - Read Only
  - Force Case
  - Validation
  - Validation Message
  - Predefault
  - Post Default
  - Calculated Value



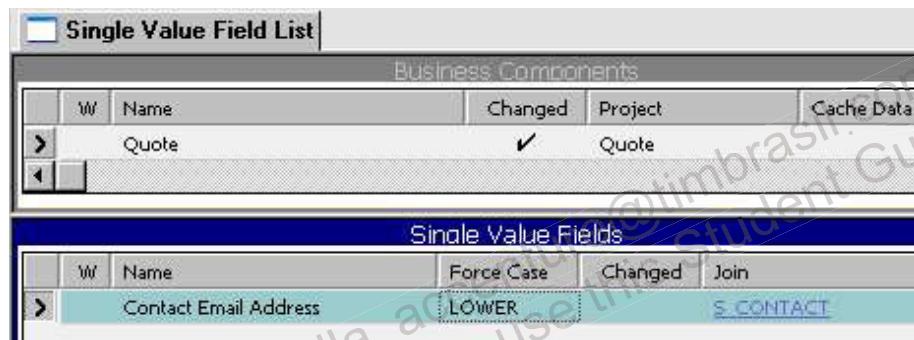
## Required and Read Only Properties

- Setting Required to TRUE prevents the user from leaving the field blank
- Setting Read Only to TRUE prevents the user from editing the value



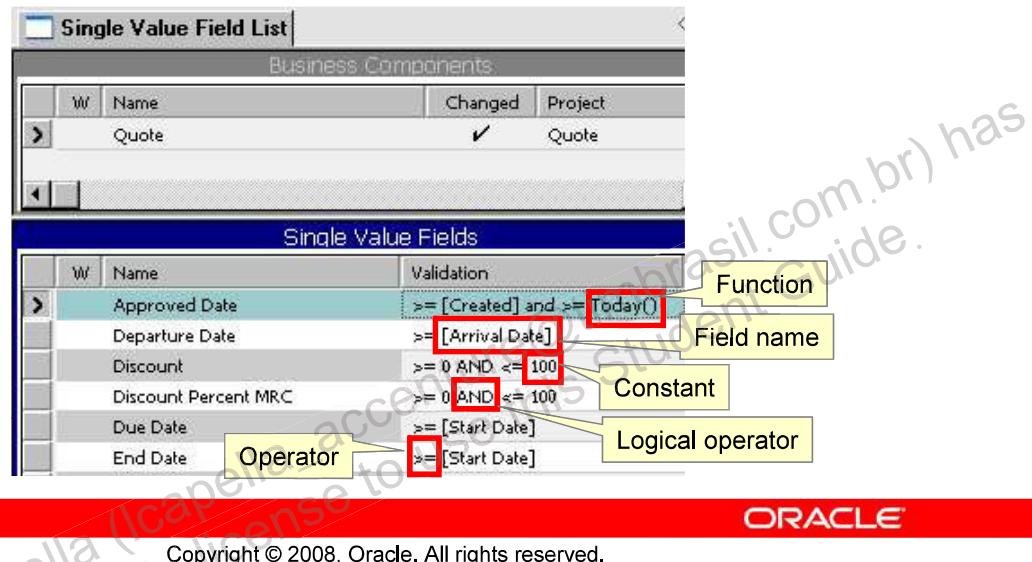
## Force Case Property

- Specifies a case for a field value
- Valid values:
  - UPPER
  - LOWER
  - FIRSTUPPER



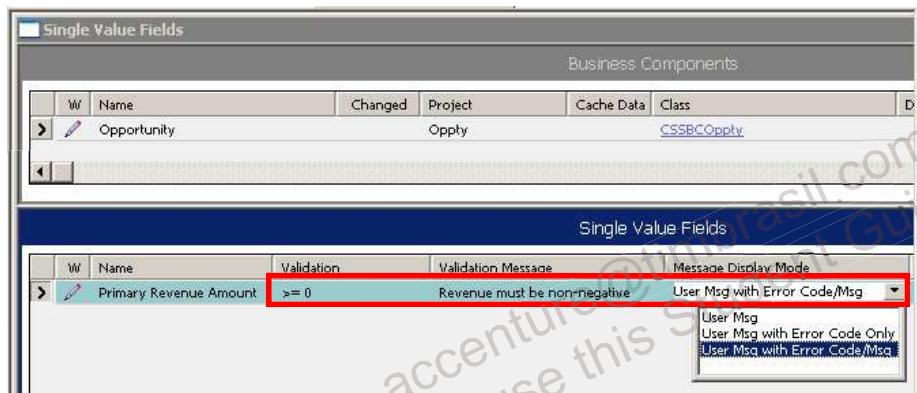
## Validation Property

- Checks a validation rule when a record is saved to ensure that the field data is valid
- Can refer only to business component fields in the same record



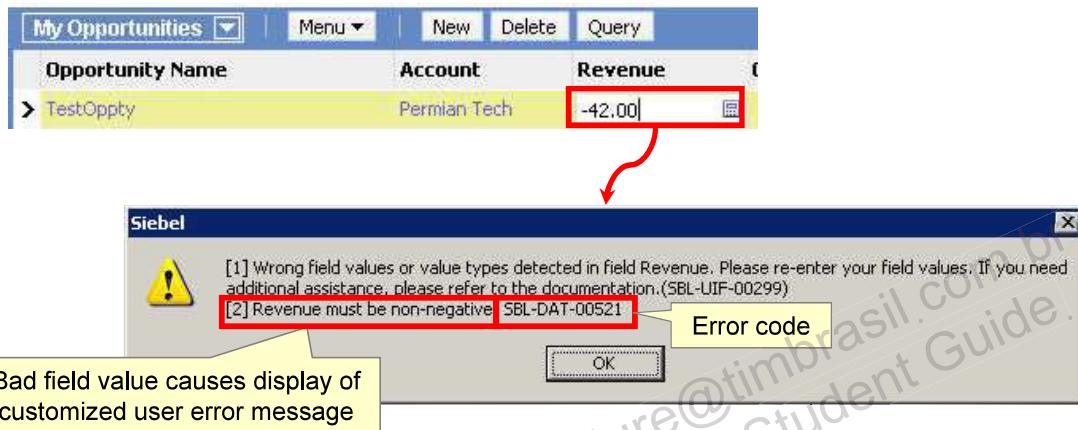
## Validation Message

- Specifies a custom message to be displayed when a business component (BC) field's Validation property is violated
  - Specify Message Display Mode to select information displayed



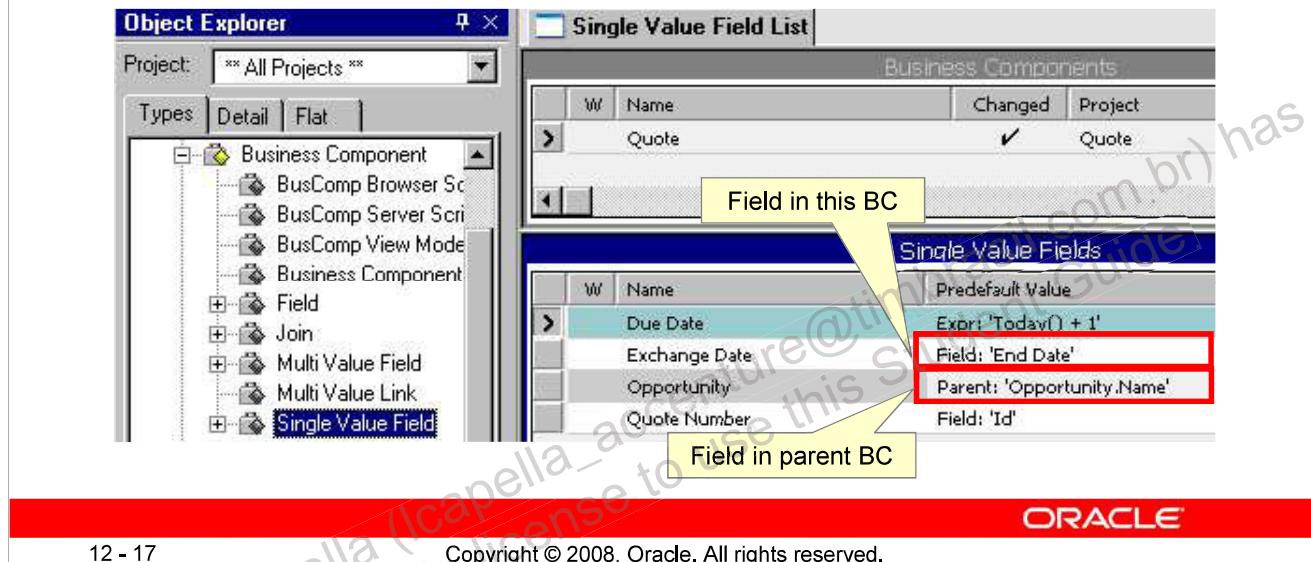
## Displaying A Validation Message

- Custom message is displayed when an invalid field is entered



## Predefault Value Property

- Automatically assigns a value to a field for a new record before the record is displayed to the user for editing
  - Expression can refer to other fields in the business component or to fields in its parent business component



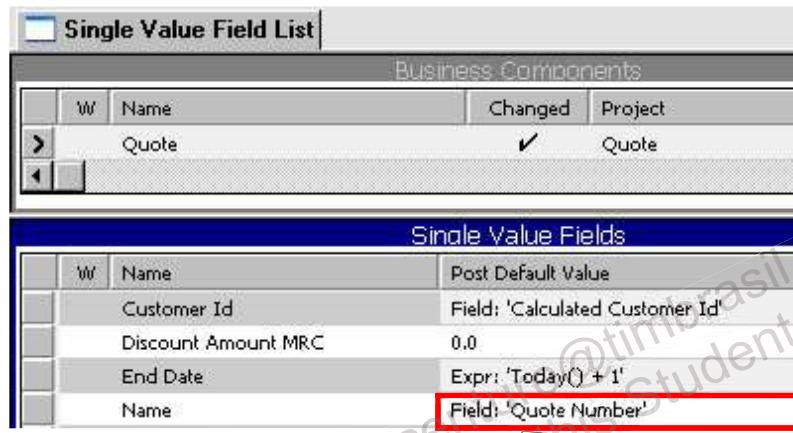
### Predefault Value Property

#### The Parent: 'BusComp:Field' Function

- The Parent: "BusComp:Field" function returns the value of the Field from the parent BusComp business component. In the example on the slide, the Opportunity field in the Quote business component is assigned the value from the Name field in the Opportunity parent business component. If the Quote business component is not currently in the context of an Opportunity business object then no value will be assigned.
- You can have multiple 'BusComp.Field' parameters separated by commas; the list is checked from first to last until a value is found. You can also terminate a chain of Parent calls with a System call—for example: Parent: 'Opportunity.Currency Code', 'Account.Currency Code', System: Currency

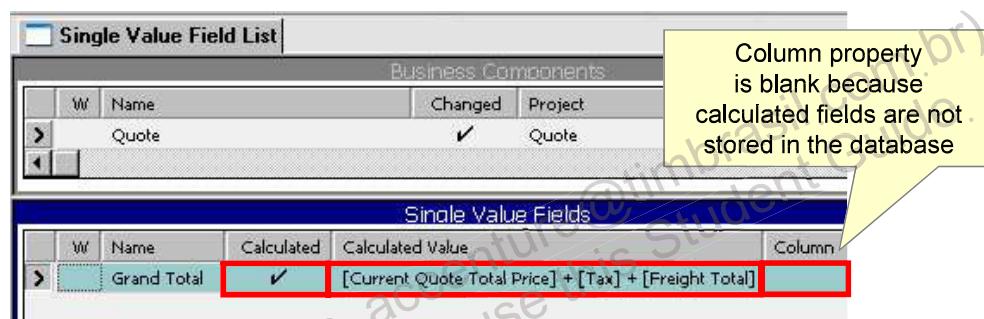
## Post Default Value Property

- Assigns a value to a field, if not entered by the user, before the record is inserted into the database
  - Expression uses the same syntax as predefault value



## Calculated Field

- Derives its value from the values in other fields in the business component
- Is not stored in the database and column property is blank
- Is configured by setting the Calculated property to TRUE (checked) and entering an expression in the Calculated Value property



12 - 19

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### Calculated Field

Since the value of calculated fields are not stored in the database, the values of such fields are calculated whenever they are retrieved. Application performance might be affected if the calculations are complex.

## Calculated Value Property

Is an expression built from:

- Field names in the same business component
- Field names from the parent business component
  - Current BC must be the child in a detail view
- Standard functions
- String, numeric, and logical operators

Single Value Fields			
Name	Calculate	Calculated Value	Function Category
Name and Location	✓	[Name] + ":" + [Location]	String
Row Status Asterisk	✓	IIF ([Row Status] = "Y", "*", "")	Logical
Timestamp	✓	Timestamp()	Date/Time
Today	✓	Today()	Date/Time

ORACLE

## Calculated Value Property

Expressions for Calculated Values

- A description of the functions that can be included in calculation expressions can be found in Siebel Developer's Reference > Operators, Expressions, and Conditions > Functions in Calculation Expressions in the Siebel Tools Online Help.

### IIF Function

- The IIF(testval,exp1,exp2) function evaluates testval. If testval is TRUE it returns the value of exp1, otherwise it returns the value of exp2.

## Restrictions on Calculated Fields

- Calculated fields are read-only
- Application does not validate values of calculated fields
- Sorting on calculated fields is not supported
- Querying on calculated fields is supported
  - Performance depends upon whether functions in the query expression can be incorporated into the SQL statement

12 - 21

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### Restrictions on Calculate Fields

When a query is performed on a calculated field, the action taken by the Siebel application (and thus the resulting performance) depends on which functions are used within the calculation. Functions that can be incorporated directly into the SQL statement are incorporated. Functions that cannot be directly incorporated, such as If() and Lookup(), result in testing each record in the business component to determine which records to display to the user, at a considerable performance cost.

## Lesson Highlights

- Business component properties can be configured to specify the behavior of the business component such as:
  - Editing (insert, delete, update, and merge)
  - Searching and sorting
- Field properties can be configured to specify:
  - Required and read only
  - Pre and post default values
  - Calculated values
  - Validation

## Practice 12 Overview: Business Components and Fields

This practice covers the following topics:

- Configuring field properties

Unauthorized reproduction or distribution prohibited. Copyright© 2014, Oracle and/or its affiliates.

Leonardo Cappella (lcapella\_accenture@timbrasil.com.br) has a  
non-transferable license to use this Student Guide.

# Business Objects and Links

13

ORACLE

Copyright © 2008, Oracle. All rights reserved.

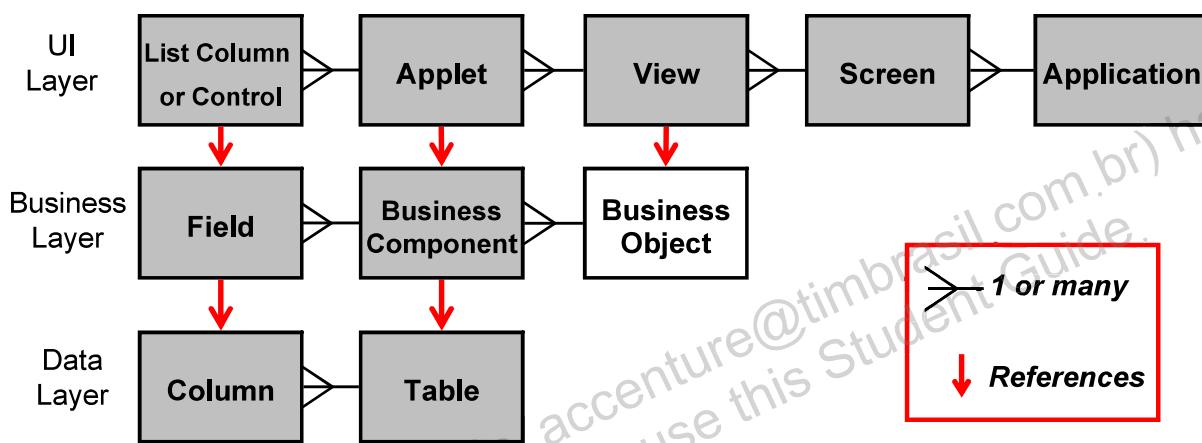
# **Objectives**

After completing this module you should be able to:

- Describe the role of business objects and links
- Create links and business objects

## Business Objects (BOs)

- Provide a way to organize business components (BCs) into major areas according to your business logic requirements
- Provide the foundation for views

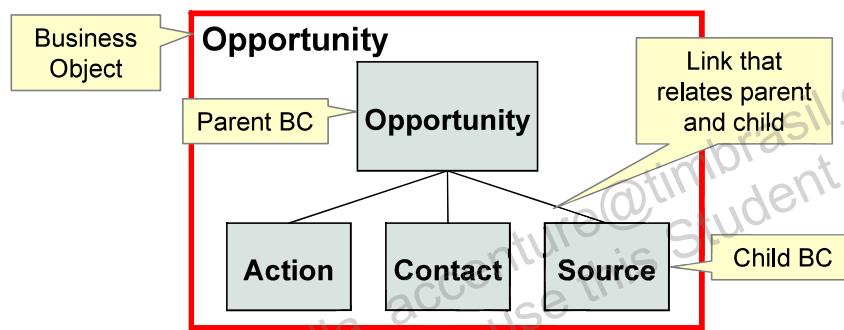


### Business Objects (BOs)

Reference: “Configuring Business Objects” in *Configuring Siebel Business Applications*

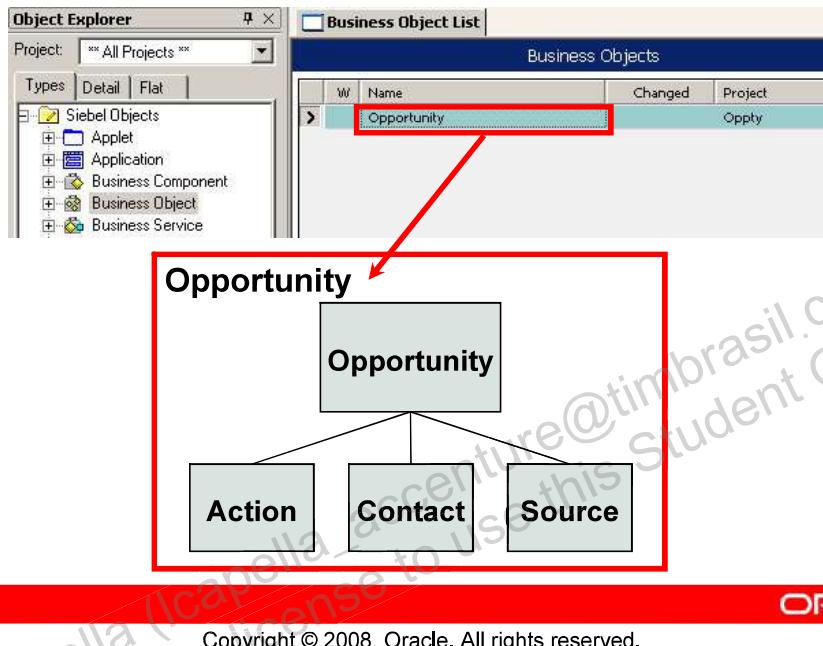
## Business Objects (BOs)

- Are a collection of related business components (BCs) that represent a major area of the business
- Include:
  - A parent business component
  - Multiple child business components
  - Links that relate the parent and child data



# Business Object Definition

- Identifies the name of the business object
  - Is usually named for the parent business component



13 - 5

Copyright © 2008, Oracle. All rights reserved.

ORACLE

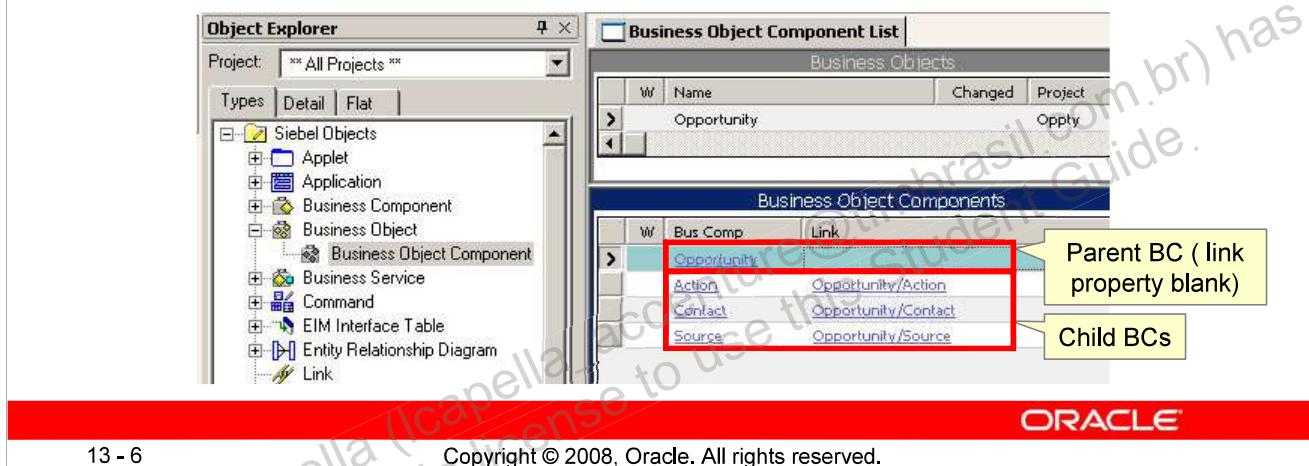
## Business Object Definition

The primary business component property in the business object definition identifies the parent or master business component in a business object. It is an optional field and not required for all business objects. The parent business component can be determined by inspecting the business object components for the component with a blank link property.

The primary business component property does not need to be specified to support views. However the property must be specified and is required to support Siebel workflows that reference the business object.

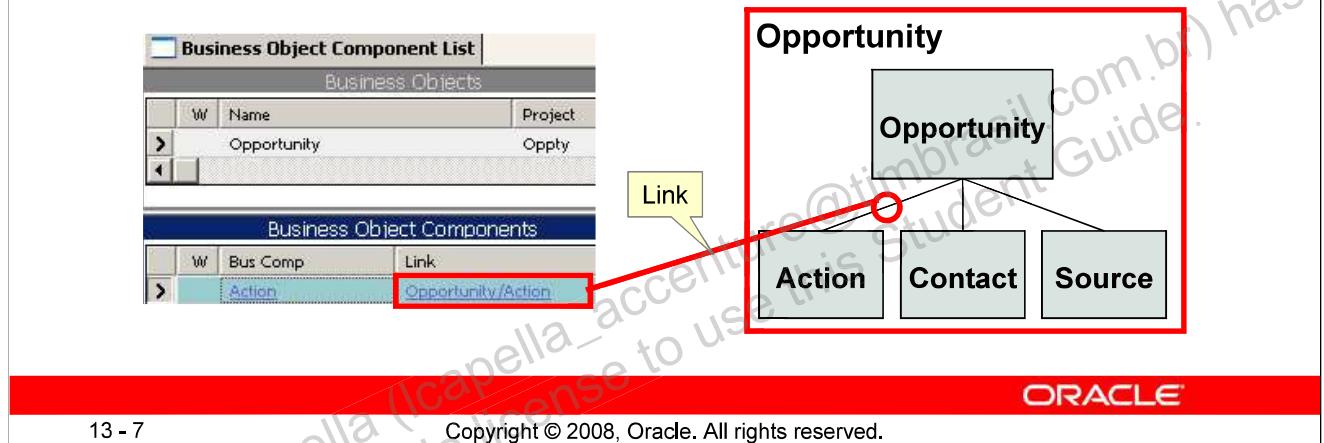
## Business Object Component

- Is a child object type of business object
- Specifies a business component to include in the business object
  - BusComp property identifies the business component
  - Link property identifies the Link object definition that relates the parent and child business component to each other



## Link Definition

- Identifies which records to retrieve from the child BC
  - Identifies the foreign keys to populate when new child records are created
- Is used with both 1:M and M:M relationships between parent and child data



13 - 7

Copyright © 2008, Oracle. All rights reserved.

### Link Definition

Reference: "Configuring Links" in *Using Siebel Tools*

## 1:M Link Definition

- Is used for 1:M relationship between parent and child BCs

The screenshot shows the Siebel Object Explorer interface with the 'Link [Opportunity/Action]' properties selected. The 'Link' type is highlighted in the tree view. Key properties shown include:

- Child BC:** Cascade Delete (None), Child Business Component (Action).
- FK field in child BC:** Destination Field (Opportunity Id).
- Parent BC:** Primary Id Field (Source Field).
- Defaults to parent BC/child BC:** Name (Opportunity/Action).
- PK field in parent BC (defaults to Id if blank):** Primary Id Field (Source Field).

Annotations with red boxes and callouts point to these specific fields. The Siebel logo is visible at the bottom right.

13 - 8

Copyright © 2008, Oracle. All rights reserved.

### 1:M Link Definition

Link object definitions are not child object types of business objects. Rather the Link object type is a top level object type. This allows link object definitions to be referenced by other object types. You will learn in a later lesson how multi value links reference links as well.

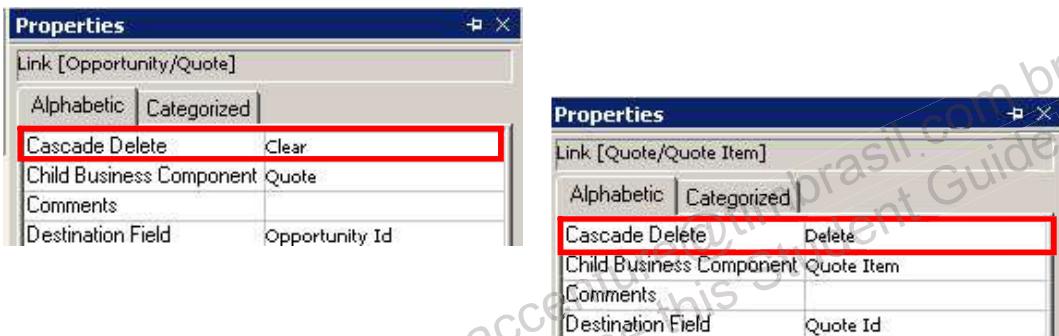
## Cascade Delete Property

- Specifies whether child records of a 1:M relationship are deleted when the parent record is deleted.
  - None: Neither deletes nor clears the foreign key column
  - Clear: Does not delete, but clears the foreign key column
  - Delete: If a parent record is deleted, all child records are deleted



## Cascade Delete Property

- Set the Cascade Delete property as determined by business requirements
  - Child quotes should not be deleted when a parent opportunity is deleted
  - Quote items should be deleted when the quote is deleted



## M:M Link Definition

- Is used for M:M relationship between parent and child business components
- Uses an intersection table to resolve the relationship

Object Explorer      Properties

Project: \*\* All Projects \*\*      Link [Opportunity/Source]

Types Detail Flat

Alphabetic Categorized

Cascade Delete	None
Child Business Component	Source
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 Locked	
Object Locked	FALSE
Object Locked By Name	
Object Locked Date	
Parent Business Component	Opportunity

13 - 11      Copyright © 2008, Oracle. All rights reserved.      ORACLE

## Child Deletion for M:M Links

- Always set Cascade Delete to None
- Set Inter Child Delete to:
  - TRUE to delete both the entry in the intersection table as well as the child record when deleting a child record in the list applet in a detail view
  - FALSE to delete only the entry in the intersection table and keep the child record

The screenshot shows two Siebel interface components. On the left is the 'Properties' window for a 'Link [Opportunity/Contact]' object. It displays various configuration settings, with 'Cascade Delete' and 'Inter Child Delete' both highlighted with red boxes. The 'Cascade Delete' field is set to 'None' and the 'Inter Child Delete' field is set to 'FALSE'. On the right is a 'Contacts' list applet showing five records. A yellow callout box points to the 'Delete' button in the toolbar above the list, with the text 'Deletes only the relationship not the contact itself'. The contacts listed are Charles Decamp, Martha Gleason, Dana Kingrey, Robert Libert, and an unselected record (indicated by a checkmark icon).

## Grandchild Business Components

- Business objects may include grandchild business components used in parent-child-grandchild views

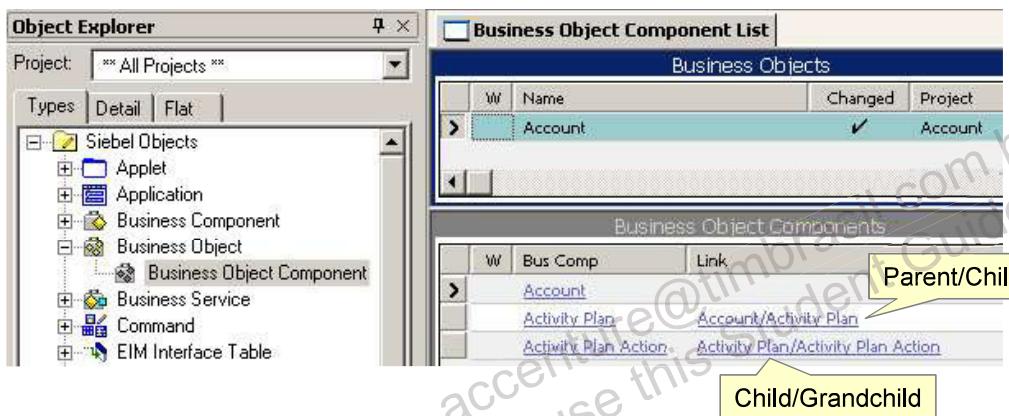
The screenshot shows a Siebel application interface with three main components highlighted by yellow callouts:

- Account (Parent BC)**: A callout points to the top-level account record for "AG Edwards & Sons, Inc." with fields for Name, Address, City, State, Zip Code, Site, and Country.
- Activity Plan (Child BC)**: A callout points to a list of activity plans associated with the account, including "Planned Start", "Template", and "Description". One item is expanded to show "8/5/2007 3:47:07 PT 03 - PCS Qualification".
- Activity Plan Action (grandchild BC)**: A callout points to a list of activities associated with the selected activity plan, showing columns for "Description", "Type", "Start", "End", and "Status". Two items are listed: "03 - Research Accol" and "03 - Needs Assessm".
- Activity plan associated with the account**: A callout points to the expanded activity plan entry.
- Activities associated with the selected activity plan**: A callout points to the list of activities under the selected activity plan.

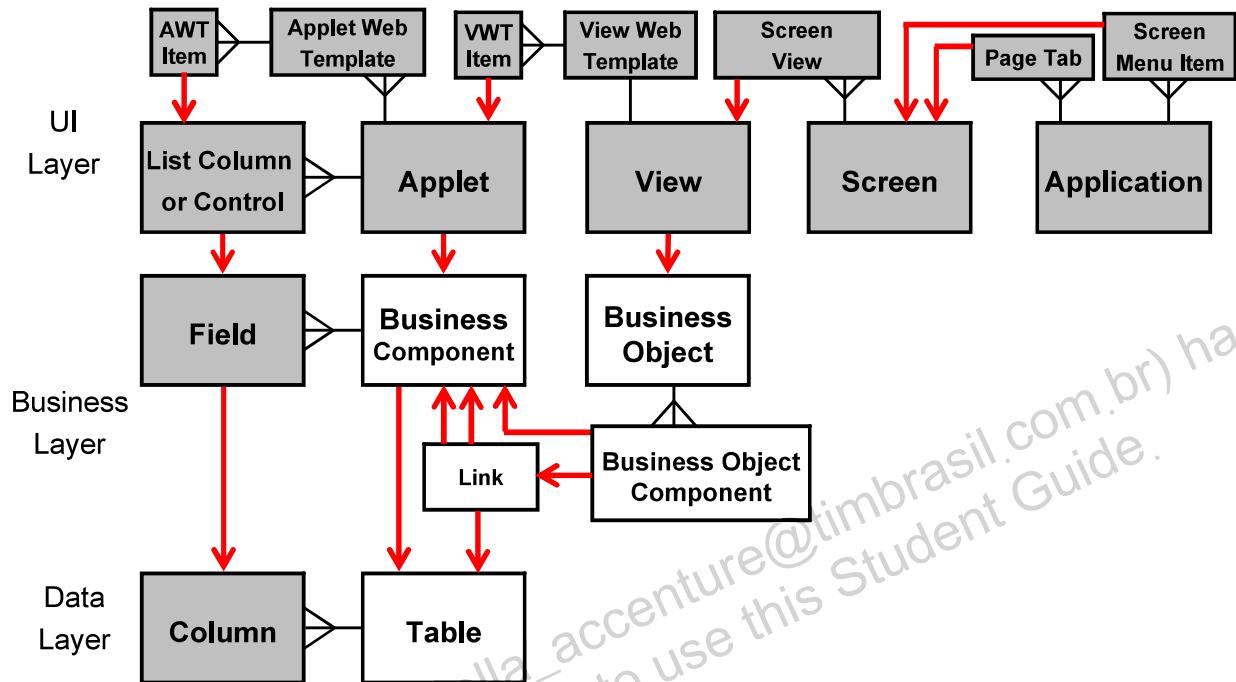
At the bottom of the interface, there is a red bar containing the text "ORACLE" and "Copyright © 2008, Oracle. All rights reserved."

## Links for Grandchild Data

- Specify how the child and grandchild business components are related
  - Are used to retrieve grandchild records on parent-child-grandchild views



## Summary of Object Types



### Summary of Object Types

In addition to the references from links to the business components and tables, there are also references in links to single value fields and columns. These references are not shown on the diagram for clarity.

## Lesson Highlights

- A business object represents a major area of the business
- A business object consists of business components and links that relate parent and child BCs
- Link definitions specify the foreign and primary keys that relate child and parent records
  - A 1:M link definition defines the FK on the child BC that points to the PK on the parent BC
  - A M:M link definition uses an intersection table to relate the parent and child BCs
- BOs may include grandchild BCs

## Practice 13 Overview: Business Objects and Links

This practice covers the following topics:

- Examining Links
- Adding a New Business Object Component

Unauthorized reproduction or distribution prohibited. Copyright© 2014, Oracle and/or its affiliates.

Leonardo Cappella (lcapella\_accenture@timbrasil.com.br) has a  
non-transferable license to use this Student Guide.

# **Creating a New Business Component Using a Standard 1:M Extension Table**

14

Copyright © 2008, Oracle. All rights reserved.

**ORACLE**

# Objectives

After completing this module you should be able to:

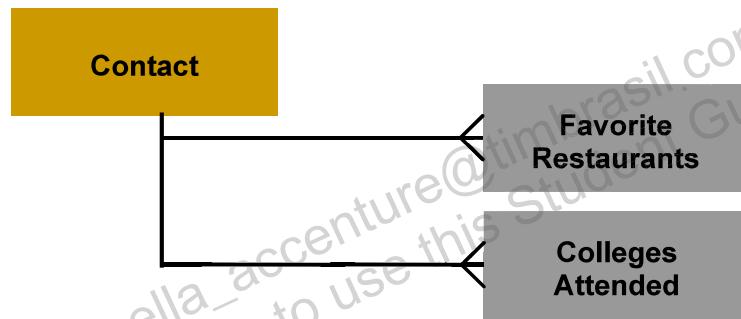
- Describe the structure of a 1:M business component
- Create a new business component (BC) using a 1:M extension table
- Add the business component to a business object

## Business Challenge

- Siebel-provided business components capture most commonly-used business entities, but they do not cover every possibility
- Example: Sales organizations might record personal data about contacts, such as:
  - What colleges the contact attended
    - Name of college, years attended, major field of study, sports played, honors received, and so on
  - The contact's favorite restaurants
    - Name and location, price range, type of cuisine, and so on
- Capturing this kind of information requires:
  - Multiple fields to capture the details
  - A 1:M relationship to the parent contact business component

## Solution: Create New Business Components

- To capture this information, create new business components as children of an existing business component
- Base these business components on 1:M extension tables supplied as part of the Siebel data model
  - For example, create college and restaurant child BCs of the Contact parent BC



## Review: 1:M Extension Table

- Exists in the repository for many tables
- Has ATTRIB\* predefined columns to store different types of user data

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer tree is expanded to show 'Siebel Objects' and 'Table'. In the center, the 'Column List' window displays a table named 'S\_CONTACT\_XM'. A yellow callout box points to the 'Name' column header, with the text 'Name is name of parent table appended with \_XM'. The table has two sections: 'Tables' and 'Columns'. The 'Tables' section shows the table name 'S\_CONTACT\_XM' and type 'Data (Public)'. The 'Columns' section lists eight columns: ATTRIB\_01 through ATTRIB\_08. The ATTRIB columns have a length of 100, while ATTRIB\_07 and ATTRIB\_08 have a length of 30. ATTRIB\_08 is defined as a Character type.

Name	Type	Length
ATTRIB_01	Varchar	100
ATTRIB_02	Varchar	100
ATTRIB_03	Varchar	30
ATTRIB_04	Varchar	30
ATTRIB_05	Varchar	30
ATTRIB_06	Varchar	30
ATTRIB_07	Varchar	30
ATTRIB_08	Character	1

14 - 5

Copyright © 2008, Oracle. All rights reserved.

ORACLE

## Review: 1:M Extension Table

Reference: “Configuring Tables and Columns” in *Configuring Siebel Business Applications*

## Review: 1:M Extension Table

- Has NAME, TYPE, and PAR\_ROW\_ID column values
  - Together must be unique per row
- Table can store multiple different “types” of data
  - For example, Colleges and Restaurants
- TYPE column identifies the different types of data

The screenshot shows the Siebel 8.1.x Tools interface with the 'Column List' window open. The left pane is the 'Object Explorer' showing various project components like Application, Business Object, and Entity Relations. The right pane is the 'Column List' window with the 'Tables' tab selected. A table named 'S\_CONTACT\_XM' is listed. Below it is a detailed view of the columns:

Name	Type
S_CONTACT_XM	Data (Public)

Name	Required	Physical Type	Length	User Key Sequence	Default
PAR_ROW_ID	✓	Varchar	15	1	
TYPE	✓	Varchar	30	2	
NAME	✓	Varchar	100	3	
CONFLICT_ID	✓	Varchar	15	4	0

A yellow callout box points to the first three columns (PAR\_ROW\_ID, TYPE, NAME) with the text "Serve as a user key".

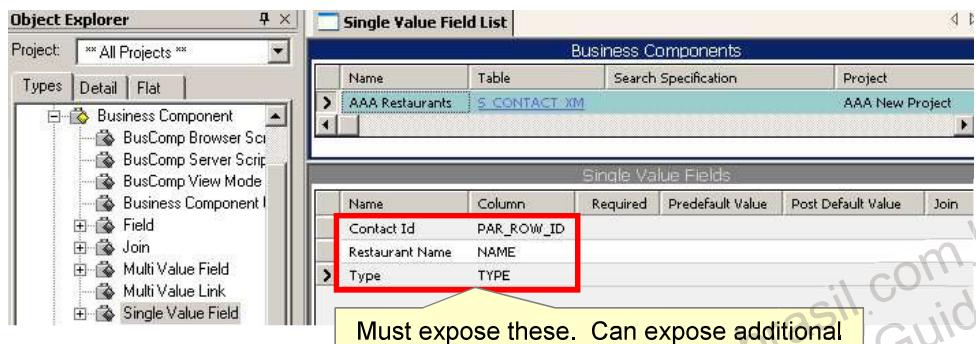
14 - 6      Copyright © 2008, Oracle. All rights reserved.      ORACLE

## Create a New Business Component

- Create a new business component to represent the child entity
  - Use the Business Component wizard
- Specify the appropriate 1:M extension table as the base table
  - Example: S\_CONTACT\_XM
- Create fields that map to the following columns
  - NAME, TYPE, and PAR\_ROW\_ID
- Create additional fields that map to ATTRIB\_ columns for additional data as required

## New 1:M Business Component

- Examine the business component and verify that it has the required fields

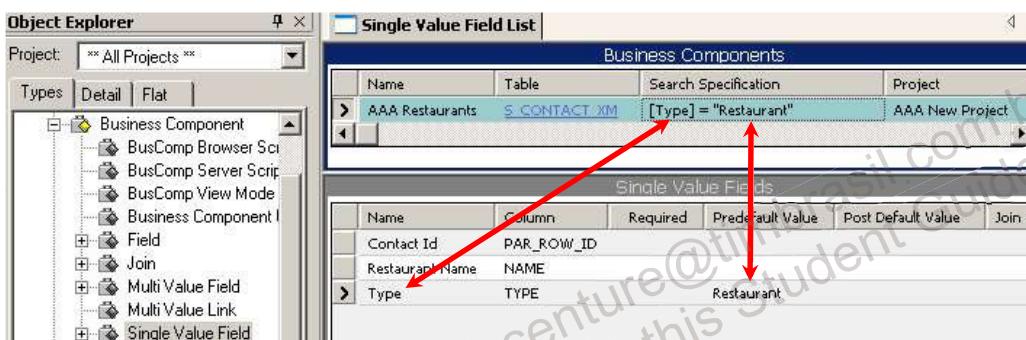


## Multiple Business Components

- An \_XM table can store multiple user-defined child business components
- Each business component has a unique type value
  - Stored in the TYPE column of the \_XM table
- Each business component retrieves only those rows with its type value
  - Search Specification and Predefault Value properties must be configured

## Set Values for Type

- Set a Predefault Value to identify the type of data being stored
  - This tags each record entered
- Set the Search Specification to search for rows of that type
  - Only records that have this type will be part of the BC



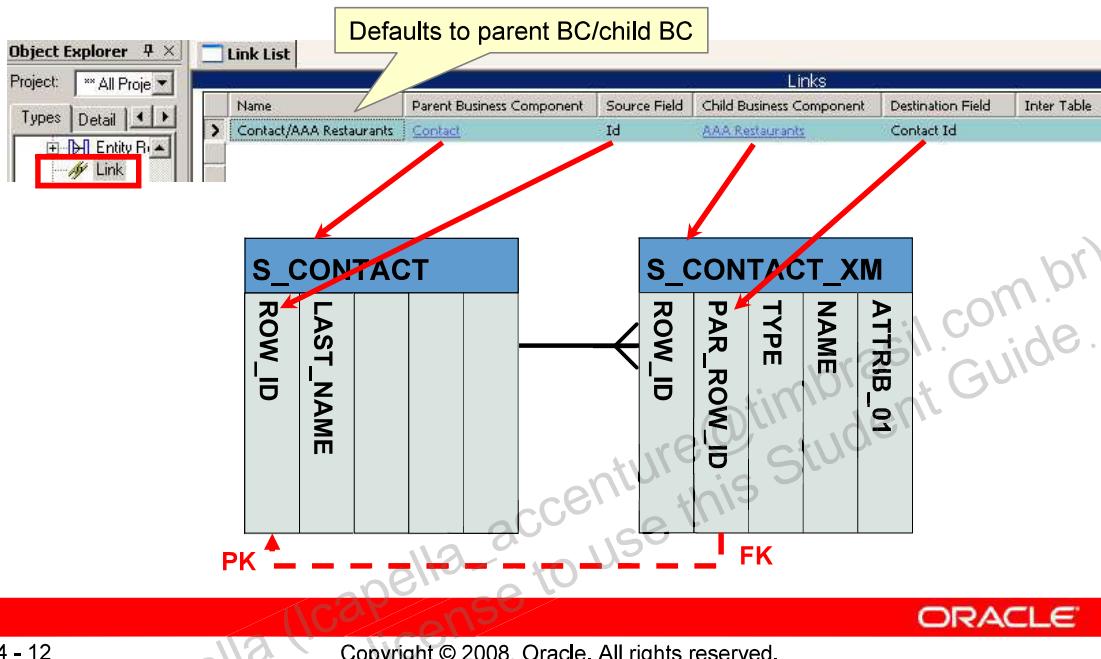
## Steps to Use the New BC

1. Create a link
2. Assign BC to a business object
3. Create a list applet
4. Assign the applet to a view
  - Create a new view if necessary and administer the new view



## 1. Create a Link

- Specify how to relate the child business component to the parent



14 - 12

Copyright © 2008, Oracle. All rights reserved.

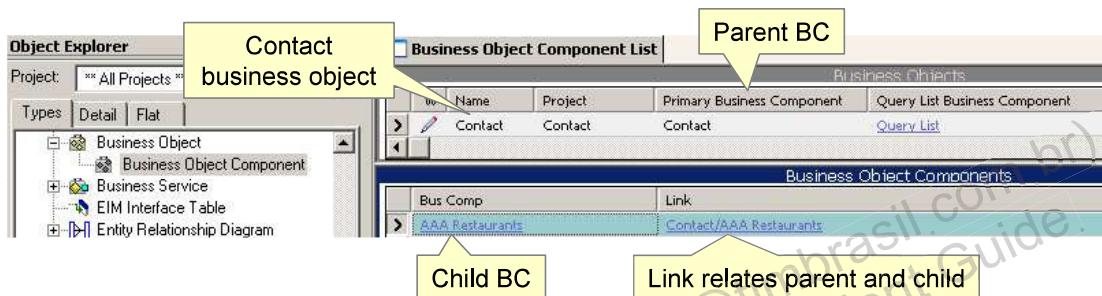
ORACLE

### 1. Create a Link

Reference: "Configuring Links" in *Configuring Siebel Business Applications*

## 2. Add BC to a Business Object

- Identify the business object that corresponds to the parent BC
- Add the child BC as a business object component
  - Set the Link property



14 - 13

Copyright © 2008, Oracle. All rights reserved.

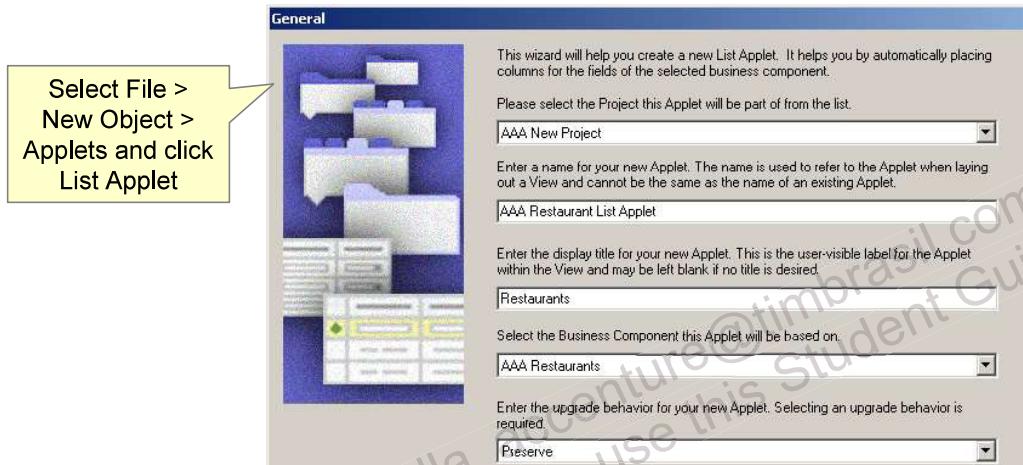
ORACLE

### 2. Add BC to a Business Object

Reference: “Configuring Business Objects” in *Configuring Siebel Business Applications*

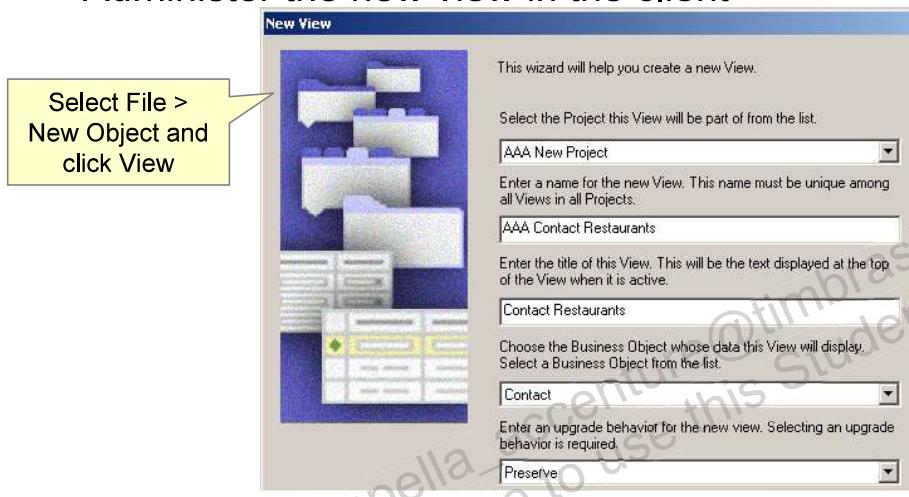
### 3. Create a List Applet

- Use the List Applet wizard to create an applet to display the data from the business component
  - To prevent users from changing the value, do not display the Type field



## 4. Assign the Applet to a View

- Use the View wizard to create a new view based on the business object for the new BC
- Add the view to a screen
- Administer the new view in the client



14 - 15

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### 4. Assign the Applet to a View

Reference: "Configuring Screens and Views" in *Configuring Siebel Business Applications*

## Lesson Highlights

- Create a new BC as a child of an existing BC to capture new information that requires a 1:M relationship
- When possible, use predefined \_XM tables to store the data
- Create a new business component and expose the NAME, TYPE, and PAR\_ROW\_ID columns
  - Configure the search specification and predefault value for the business component to retrieve the desired records
- Create a link definition to define the relationship between the parent and child BC
- Add the new BC to a business object, then create applets and views to display data

## Practice 14 Overview: Creating a New Business Component Using a Standard 1:M Extension Table

This practice covers the following topics:

- Creating a business component and link
- Displaying the business component in a detail view

Unauthorized reproduction or distribution prohibited. Copyright© 2014, Oracle and/or its affiliates.

Leonardo Cappella (lcapella\_accenture@timbrasil.com.br) has a  
non-transferable license to use this Student Guide.

# Extending the Siebel Database

# 15

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Objectives

After completing this lesson you should be able to:

- Create extension columns in a table
- Create a custom table
  - Standalone table
  - 1:1 extension table
  - 1:M extension table
  - Intersection table
- Apply database extensions

## Incorporating Additional Data

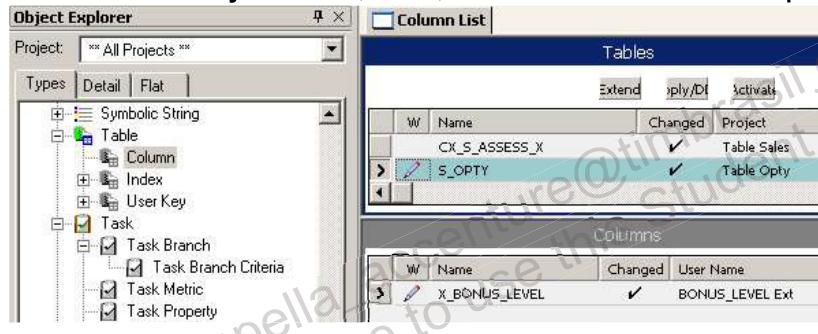
- Your business requirements may include:
  - Adding new fields to capture additional data
  - Creating new business components to capture additional business entities
- First examine the Siebel Data Model carefully to determine whether the desired fields and business components already exist
  - Consider using a 1:M extension table for a new business component that is a child of an existing business component

## Adding New Fields

- When the data model does not support the desired new fields, extend the as-delivered Siebel database by creating extension columns on base tables
- Avoid mapping new fields to existing columns that appear to be unused
  - The columns may be in fact used or could be used in future releases
- Avoid mapping new fields to the standard 1:1 extension table
  - Some columns are already in use and others may be used in future releases
  - Such fields may encounter performance issues

## Extending the Siebel Database

- Use Siebel Tools to extend the Siebel database by:
  - Creating extension columns on tables
  - Creating new tables
- Developers:
  - Create new object definitions for the database extension
  - Make the corresponding physical database changes
  - Do not directly create, use, or maintain SQL scripts



15 - 5

Copyright © 2008, Oracle. All rights reserved.

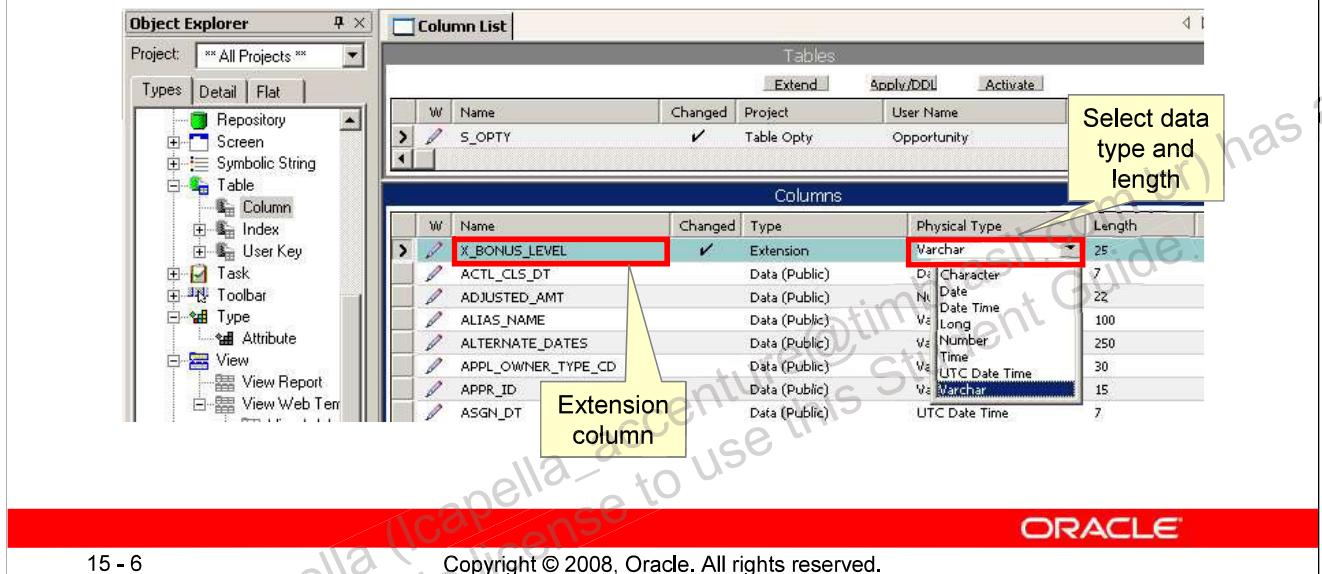
ORACLE

### Extending the Siebel Database

Reference: “Configuring Tables and Columns” in *Configuring Siebel Business Applications*

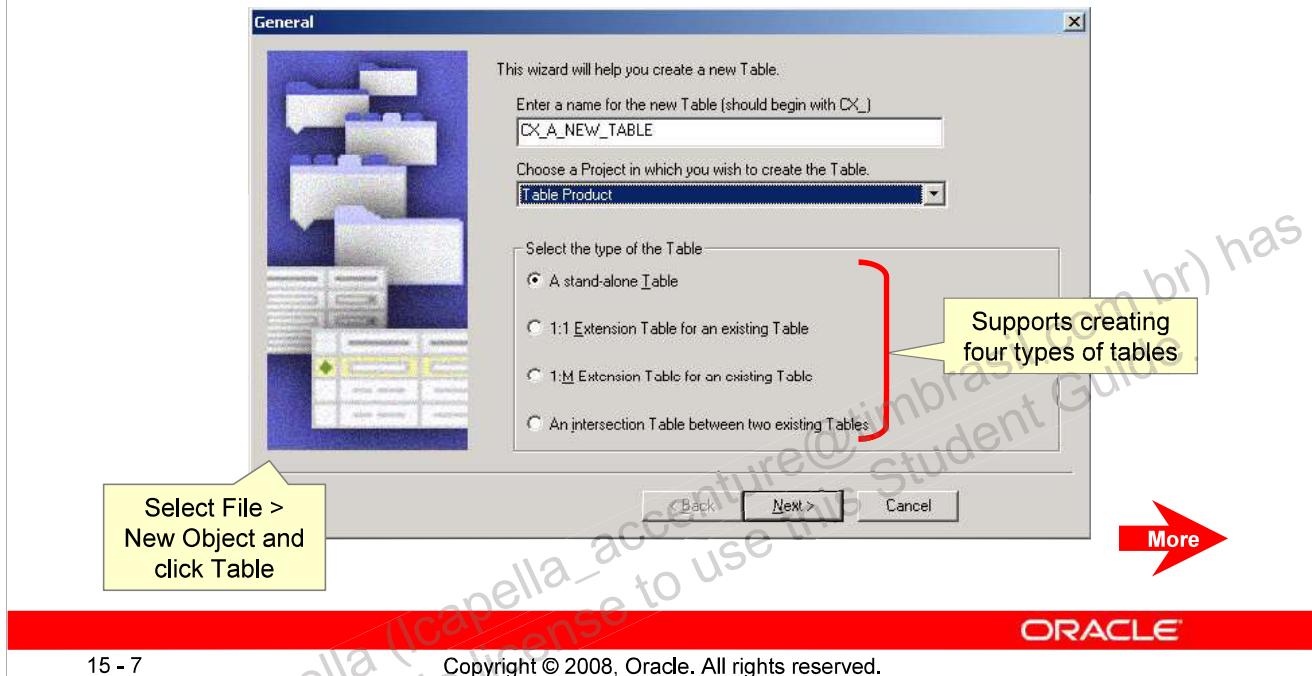
## Creating a Custom Extension Column

- Select the table to be extended
- Create a new column record with the desired properties
  - Name is automatically prefixed with X\_



## Creating a Stand-Alone Table

- Use the Table Wizard to create a new table



# Table Wizard

- Creates a standalone table with:
  - Data (Public) as its type
  - The required system columns
  - One index P1 on ROW\_ID

The screenshot shows the Siebel Object Explorer interface. On the left, the Object Explorer tree is visible with various project types like Application, Bitmap Category, Business Component, etc. The 'Table' node under 'Table' is selected. On the right, the 'Column List' window is open, showing the configuration for the table 'CX\_A\_NEW\_TABLE'. The 'Tables' tab is selected, and the 'Columns' tab displays a list of columns with their names, changed status, types, and lengths. A red box highlights the 'Type' column, which shows 'System' for most columns. A yellow callout bubble with the text 'System columns' points to this red box. The Oracle logo is at the bottom right.

W	Name	Changed	Type	Physical Type	Length
>	CONFLICT_ID	✓	System	Varchar	15
>	CREATED	✓	System	UTC Date Time	7
>	CREATED_BY	✓	System	Varchar	15
>	DB_LAST_UPD	✓	System	UTC Date Time	7
>	DB_LAST_UPD_SRC	✓	System	Varchar	50
>	LAST_UPD	✓	System	UTC Date Time	7
>	LAST_UPD_BY	✓	System	Varchar	15
>	MODIFICATION_NUM	✓	System	Number	22
>	ROW_ID	✓	System	Varchar	15

15 - 8

Copyright © 2008, Oracle. All rights reserved.

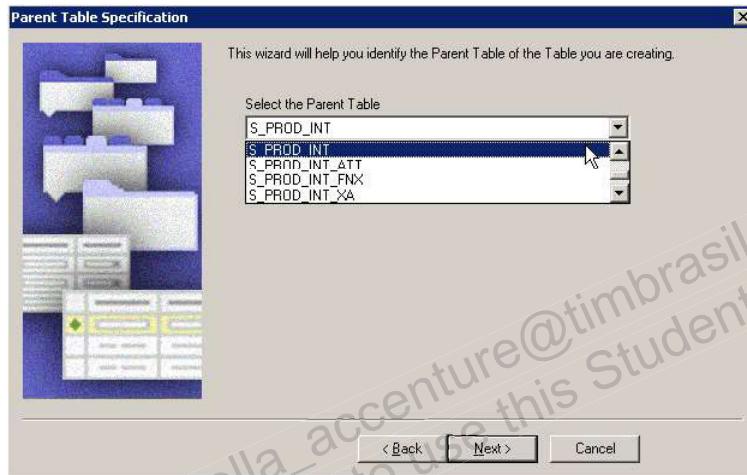
ORACLE

## Table Wizard

See “Configuring Tables and Columns” in *Configuring Siebel Business Applications* for details about each of the system columns.

## Creating a 1:1 Extension Table

- Select a base table as input to the Table Wizard
  - Choice restricted to the Data (Public) type
  - Multiple extension tables relate directly to the base table, and not to each other



ORACLE

15 - 9

Copyright © 2008, Oracle. All rights reserved.

### Creating a 1:1 Extension Table

In most cases a new field should be mapped to an extension column in a base table rather than an existing column in a 1:1 extension table. This avoids possible performance issues associated with the required join. As such it is unlikely that you would be creating new 1:1 extension tables.

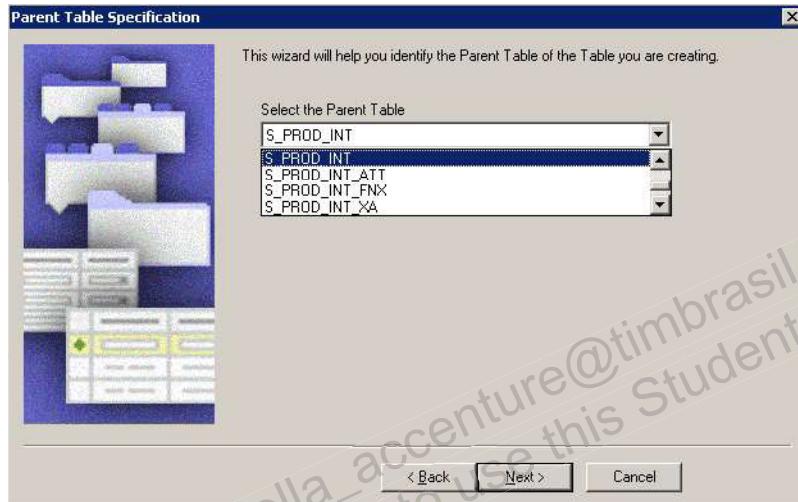
However the database system you are using may have limitations on the size of a row in a database table or the number of columns permitted in a table. If this is the case, you may need to consider creating a 1:1 extension table.

## Creating a 1:1 Extension Table

- The Table Wizard creates an extension table with:
  - Required system columns
  - PAR\_ROW\_ID column as the foreign key column to the base table
  - Two indexes:
    - P1 index on ROW\_ID
    - U1 index on PAR\_ROW\_ID and CONFLICT\_ID

## Creating a 1:M Extension Table

- Select a parent table as input to the Table Wizard
  - Create if the parent table does not have an existing 1:M table
  - Choice restricted to the Data (Public) type



15 - 11

Copyright © 2008, Oracle. All rights reserved.

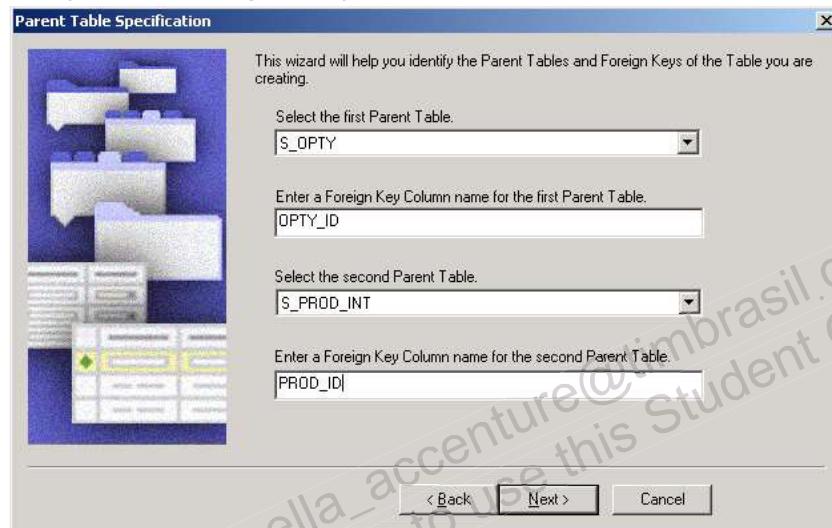
ORACLE

## Creating a 1:M Extension Table

- Table wizard creates a 1:M extension table with:
  - Data (Public) as its type
  - The required system columns
  - A PAR\_ROW\_ID column as the foreign key column to the base table
  - TYPE and NAME columns
  - Three indexes:
    - P1 index on ROW\_ID
    - U1 index on PAR\_ROW\_ID, TYPE, NAME, and CONFLICT\_ID
    - M1 index on TYPE and NAME

## Creating an Intersection Table

- Select both parent tables
  - Choices restricted to the Data (Public) type
- Specify the foreign key column name for each parent table



15 - 13

Copyright © 2008, Oracle. All rights reserved.

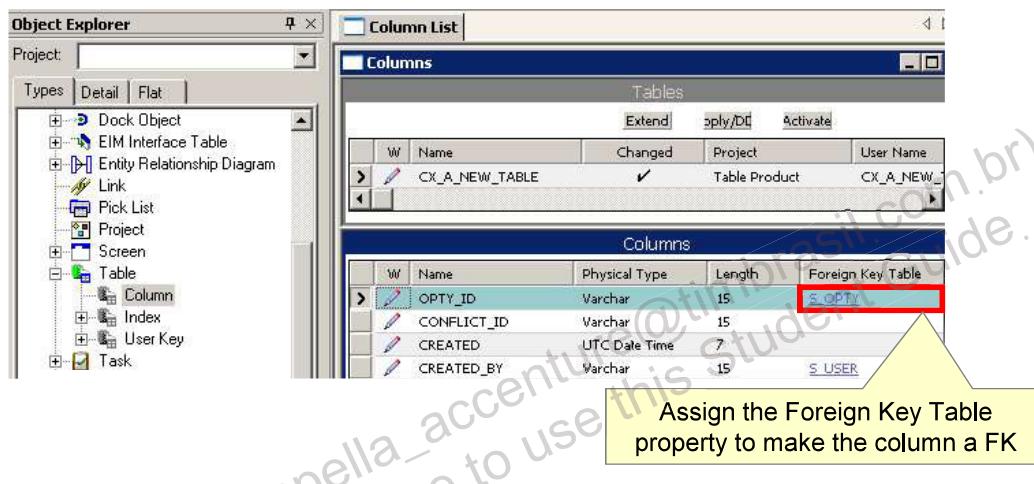
ORACLE

## Creating an Intersection Table

- Table Wizard creates an intersection table with:
  - Data (Intersection) as its type
  - Required system columns
  - Two foreign key columns as specified
  - Three indexes:
    - P1 index on ROW\_ID
    - U1 index on two foreign key columns, TYPE, and CONFLICT\_ID
    - F1 index on foreign key to second parent table

## Creating Foreign Keys

- Relationships can be created between tables
  - Create a foreign key column in one of the tables
  - Create join and link object definitions to make sure the foreign key columns are properly populated



15 - 15

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### Creating New Foreign Keys

In some cases, there may already be one or more foreign key columns that relate two tables. Each foreign key column formalizes a specific relationship. If you wish to create another relationship between the tables, it is recommended that you create a new foreign key column to formalize the new relationship. You should avoid repurposing one of the existing relationships (even if you are not planning to make use of it) because knowledge of the foreign keys is already captured in existing EIM mappings and dock objects.

## Propagating Database Changes to Other Developers

- Test changes locally before applying them to the server database
  - Reduces the likelihood of undesired changes to the server schema
- Best practices for changing the schema:
  1. Apply Changes to the Local Database
  2. Propagate Changes to the Server Database
  3. Propagate Changes to Other Developers



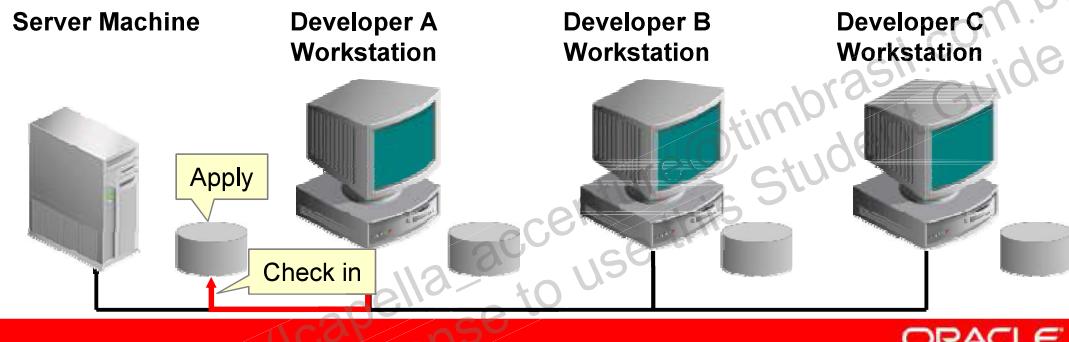
## 1. Apply Changes to the Local Database

- Click Apply/DDL to make the physical database changes
  - Choice to apply schema changes or generate DDL script
  - Changes are preserved across Siebel application upgrades
- Compile relevant objects and projects
- Test changes locally before checking in to the server
  - Query tables/columns using a database SQL utility



## 2. Propagate Changes to Server Database

- Check the project into the server
  - Copies the table and column object definitions
- Select the table and click Apply/DDL to make the physical database changes
- Click the Activate button to update the database schema version
- Compile and test against the server database

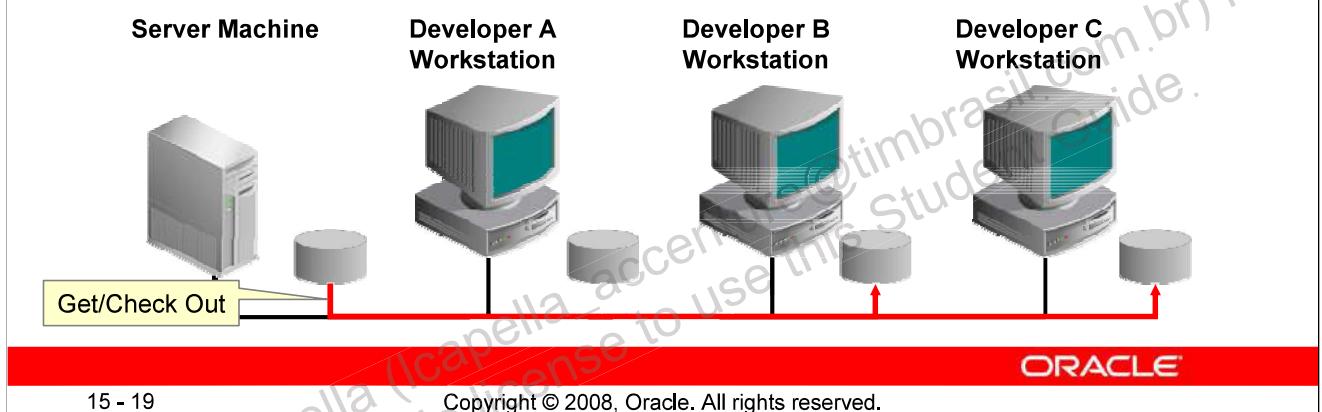


## 2. Propagate Changes to Server Database

Alternatively you can run the Database Configuration Utility to apply a set of schema changes to the server database.

### 3. Propagate Changes to Other Developers

- Other developers need to apply changes to their local databases
  - Have other developers get or check out the project and apply the changes locally
  - Alternatively re-extract developers and have them get all projects
    - Ensure all work on the local machines has been checked in first



## Lesson Highlights

- Extend the Siebel database in Siebel Tools:
  - Add an extension column to an existing table
  - Use the Table wizard to create a standalone, 1:1 extension, 1:M extension, or intersection table
- Best practices to modify Siebel database schema:
  - Apply changes locally and test
  - Propagate changes to the server database
  - Propagate changes to other developers

## Practice 15 Overview: Extending the Siebel Database

This practice covers the following topics:

- Exploring the Siebel database
- Extending a Siebel database table
- Creating a Siebel database table

Unauthorized reproduction or distribution prohibited. Copyright© 2014, Oracle and/or its affiliates.

Leonardo Cappella (lcapella\_accenture@timbrasil.com.br) has a  
non-transferable license to use this Student Guide.

## Additional Data Layer Configuration

# 16

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Objectives

After completing this lesson you should be able to:

- Create custom indexes
- Modify dock objects to incorporate extension tables
- Extend a database table to support case-insensitive and accent-insensitive (CIAI) queries

## Additional Considerations

- After creating a new column or database table, you might need to:
  - Create additional custom indexes
  - Create additional EIM mappings
  - Modify the dock objects



## Custom Indexes

- Custom indexes based on new columns can be created to improve database performance
  - Standard indexes cannot be modified or deleted
  - Work with the database administrator to determine whether custom indexes are required
- Custom indexes:
  - Must be specified carefully
    - Consider consulting Oracle Expert Services
  - Can impact the benefit of standard indexes
  - Need to be thoroughly tested prior to release to production

ORACLE

16 - 4

Copyright © 2008, Oracle. All rights reserved.

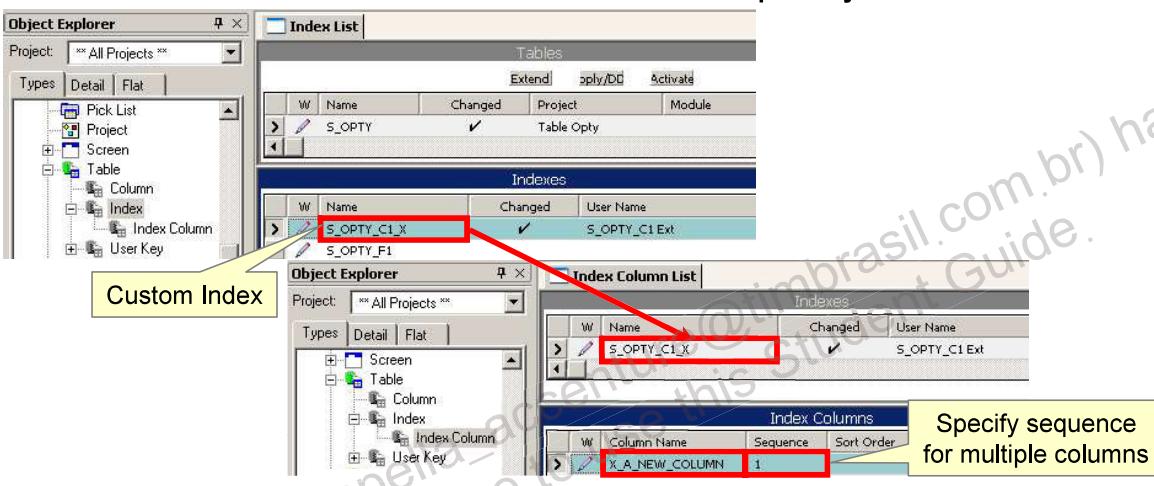
### Custom Indexes

Custom indexes are created using Siebel Tools to make sure that they can be migrated from development to test and production environments. In addition this allows developers to be aware of custom indexes when they are configuring search and sort specifications.

**Reference:** “Configuring Tables and Columns” in *Configuring Siebel Business Applications*

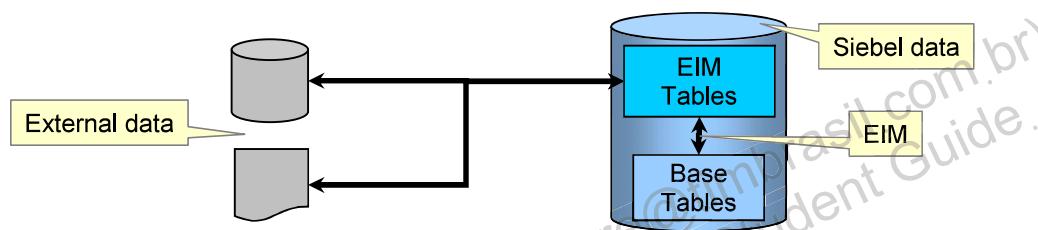
## Creating a Custom Index

- Select the table
- Create a child Index object definition
  - Name is automatically suffixed with \_X
- Create new Index Column records to specify the columns



## Enterprise Integration Manager (EIM)

- Is a server component that enables the batch exchange of data between a Siebel database and an external data source
- Moves data between Siebel EIM tables and Siebel base tables



### Enterprise Integration Manager

Enterprise Integration Manager is covered in detail in its own Oracle University course.

## EIM Mappings

- Identify the columns in a base table that are populated by columns in an EIM table
  - Are specified as child object types of the EIM Interface Table object type
- Are used by EIM when importing or exporting data
  - New EIM mappings will need to be created to allow EIM processes to import data into or export data from new tables and columns
    - For new columns in an extended or a new table:
      - Add a new column to an EIM table
      - Create a new mapping between the EIM and base table
    - The EIM Mapping wizard can implement the configuration for many common scenarios

ORACLE

16 - 7

Copyright © 2008, Oracle. All rights reserved.

### EIM Mappings

Reference: “Configuring EIM Interfaces” in *Configuring Siebel Business Applications*

## Dock Objects

- Are a set of related tables used to determine which records are synchronized with mobile users
- Consist of a driving table and other related tables
  - Are specified as Dock Object Table object definitions

The screenshot shows the Siebel Object Explorer and the Dock Object Table List windows. In the Object Explorer, the 'Siebel Objects' node is expanded, showing various object types like Applet, Application, Business Component, etc. In the Dock Object Table List window, a 'Dock Objects' table is displayed with a single row for 'Opportunity'. A yellow box highlights the 'Driving or primary table' section of the Dock Object Tables table, which lists several tables: S\_OPTY, S\_OPTYCON\_AUTH, S\_OPTY\_ORG, S\_OPTY\_TERR, S\_OPTY\_STG, S\_OPTY\_SRC, S\_OPTY\_SLS\_STEP, and S\_OPTY\_SKILL. The 'S\_OPTY' table is selected, and its details are shown in the table rows below.

Table Name	Source Column Name	Target Table Name	Target Column Name
S_OPTY			
S_OPTYCON_AUTH	OPTYCON_ID	S_OPTY_GON	ROW_ID
S_OPTY_ORG	OPTY_ID	S_OPTY	ROW_ID
S_OPTY_TERR	OPTY_ID	S_OPTY	ROW_ID
S_OPTY_STG	OPTY_ID	S_OPTY	ROW_ID
S_OPTY_SRC	OPTY_ID	S_OPTY	ROW_ID
S_OPTY_SLS_STEP	OPTY_ID	S_OPTY	ROW_ID
S_OPTY_SKILL	OPTY_ID	S_OPTY	ROW_ID

16 - 8

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### Dock Objects

Reference: “Configuring Docking Rules” in *Configuring Siebel Business Applications*

## Types of Dock Objects

- Dock objects are classified into three types based on how records in tables are distributed during remote synchronization
  - Enterprise: allows all records to be synchronized
  - Private: prevents records from being synchronized
  - Limited: records are synchronized for particular users based on characteristics of the remote user
    - Reduces the amount of data exchanged during synchronization
    - Provides remote users with data they need to have

W	Name	Project	Visibility Level
	Industry	Dock Reference Data	Enterprise
	Mobile Device	Dock Mobile Device	Private
	Opportunity	Dock Opportunity	Limited

## Dock Object Visibility Rules

- Limited dock objects have a set of rules that determine:
  - The users that receive updated or new records
  - The tables in a dock object from which they receive such records

W	Sequence	Comments	Visibility Strength
1		You are on the sales team of the Opportunity	100
2		You are the manager of the primary sales rep on the Op	100
3		Related Opportunity for an Opportunity you have full v	50
5		Opportunity for an Activity you have full visibility on	50
6		Opportunity for an Account you have full visibility on	50
7		Opportunity for an Indirect Account you have full visib	50
8		Opportunity for a Contact you have full visibility on	50
9		Opportunity for an Agreement you have full visibility on	50

## Modifying Dock Objects

- Docks objects might need to be created or modified to allow records to be synchronized with mobile users when:
  - Creating a table
  - Adding a foreign key column to a new or an existing table
- Use the Dock Object wizard to create a new dock object for a new standalone table
- Consult Expert Services to modify existing dock objects

## Extending a Table to Support Case Insensitive Queries

- A database table can be extended to support case-insensitive and accent-insensitive (CIAI) queries
  - Restricted to columns of type text
- Process involves:
  - Creating a shadow column to store the value in the original column in uppercase
  - Creating new indexes on the shadow column to permit rapid access to records using the uppercase value

S_ORG_EXT			
ROW_ID	NAME	LOC	X_NAME_CI
			X_LOC_CI

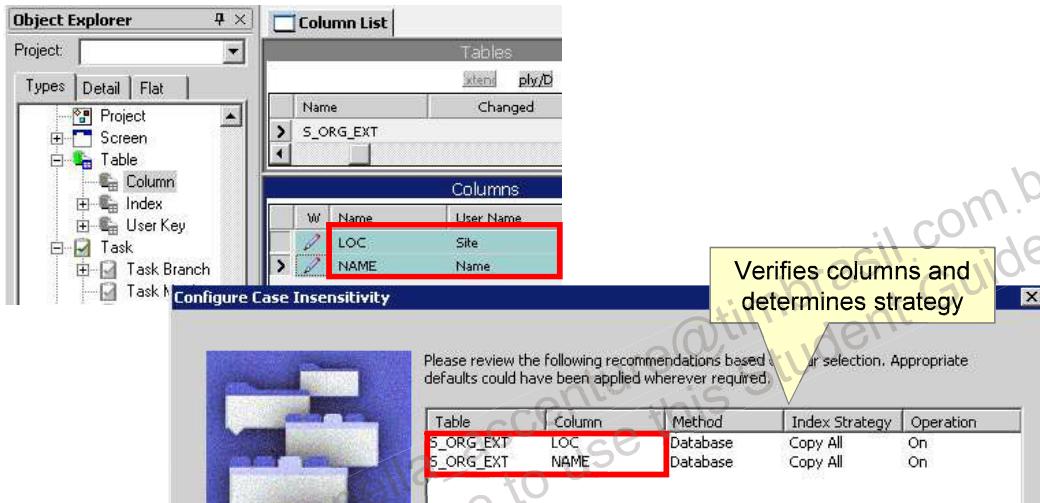
Shadow column to store Location in upper case

## Extending a Table to Support Case Insensitive Queries

Reference: “Configuring Tables and Columns” in *Configuring Siebel Business Applications*

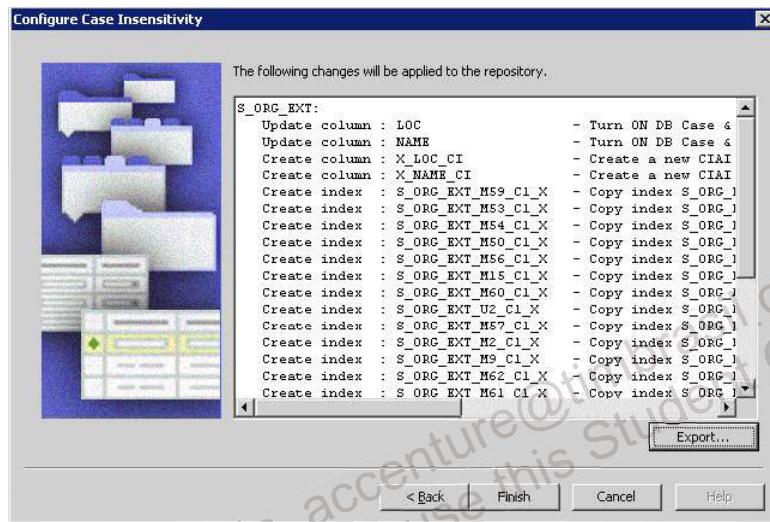
## Creating CIAI

- Select the columns in the object list editor to convert to CIAI
- Right-click and select Case Insensitivity to invoke the CIAI wizard



## CIAI Wizard

- Identifies columns and indexes to create
- Applies them to the repository



ORACLE

16 - 14

Copyright © 2008, Oracle. All rights reserved.

## CIAI Wizard

When you apply these repository changes to the physical database, the resultant changes are database dependent. In some databases the shadow columns are explicitly created, while in other databases the shadow columns are implemented by creating database indexes.

## Lesson Highlights

- Custom indexes based on new columns can be created to improve database performance
- Use the EIM Mapping wizard to allow EIM processes to import data into or export data from new tables and columns
- Dock objects may need to be modified to allow data to be synchronized with remote users
  - Use the Dock Object wizard to create a new dock object for a new standalone table
- Use the CIAI wizard to extend a table to support case-insensitive and accent-insensitive (CIAI) queries

## Practice 16 Overview: Additional Data Layer Configuration

This practice covers the following topics:

- Configuring case insensitive columns

# 17

## Configuring Drilldowns and Applet Toggles

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# **Objectives**

After completing this lesson you should be able to:

- Configure a drilldown to a related view
- Configure a thread bar
- Configure a toggle applet

## Navigation Using Drilldown

- You can configure list columns for drilldown
  - When the user clicks the hyperlinked value, the application navigates to another view
- You can configure drilldown as static or dynamic
  - Static: Clicking the hyperlink always navigates to the same target view
  - Dynamic: Clicking the hyperlink navigates to a target view determined by the value of a field in the record

### Navigation Using Drilldown

Drilldowns are not supported for controls in a form applet.

**Reference:** “Configuring Screens and Views” in *Configuring Siebel Business Applications*

## Static Drilldown: Same Business Component

- Drill down to a detail view for the same record
  - Destination view uses the same business object and business component (BC)

The screenshot illustrates a static drilldown within the same business component. It shows two views of the Order Entry module:

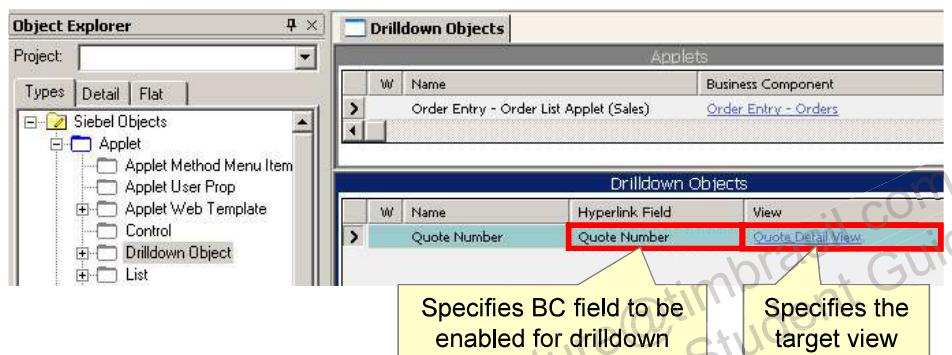
- List View:** A grid titled "My Sales Orders" showing five sales orders. The first row, with Order # 130258-2220112, is selected and highlighted in yellow. A red arrow points from this row to the detail view below.
- Detail View:** A form for Order # 130258-2220112. The top section displays basic details: Order #: 130258-2220112, Revision: 1, Account: Brown, Nick, Opportunity: (empty), Site: (empty), and Status: Complete. Below this are tabs for Line Item Detail, More Info, Catalog, Work Orders, Proposals, Notes, Transactions, Line Items, Shipping, and Payment. The Catalog tab is currently selected.

Annotations provide additional context:

- A yellow callout box points to the list view: "View with business object Order Entry (Sales) and business component Order Entry - Orders".
- A yellow callout box points to the detail view: "View with business object Order Entry (Sales) and business component Order Entry - Orders".
- A yellow box on the left of the list view contains the text: "Drilldown on Order # shows details for same record".

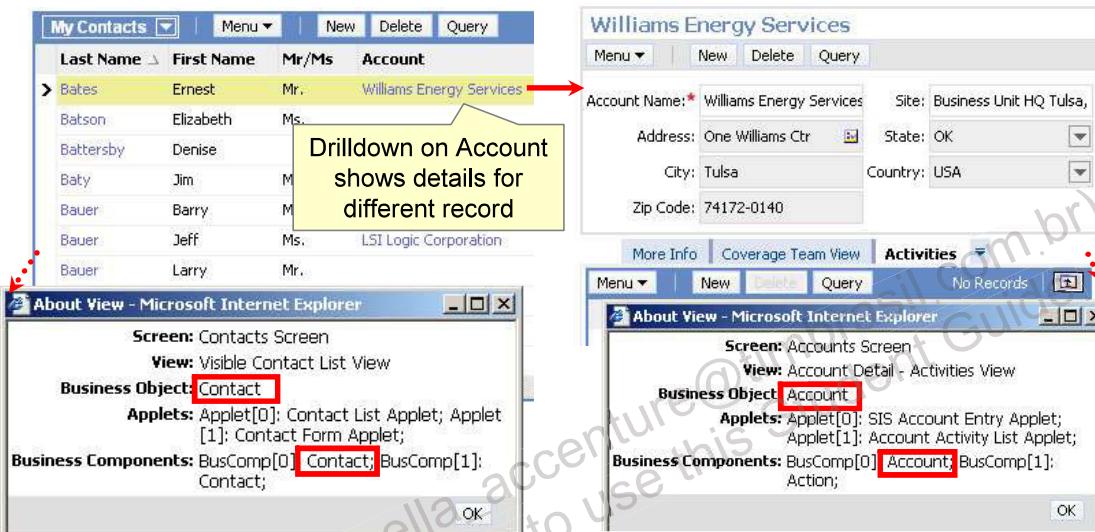
## Configuring Drilldown Within the Same Business Component

- Create a drilldown object (child of applet)
  - Set the Hyperlink Field to the field to be enabled for drilldown
  - Set the View to the target view when the field is clicked



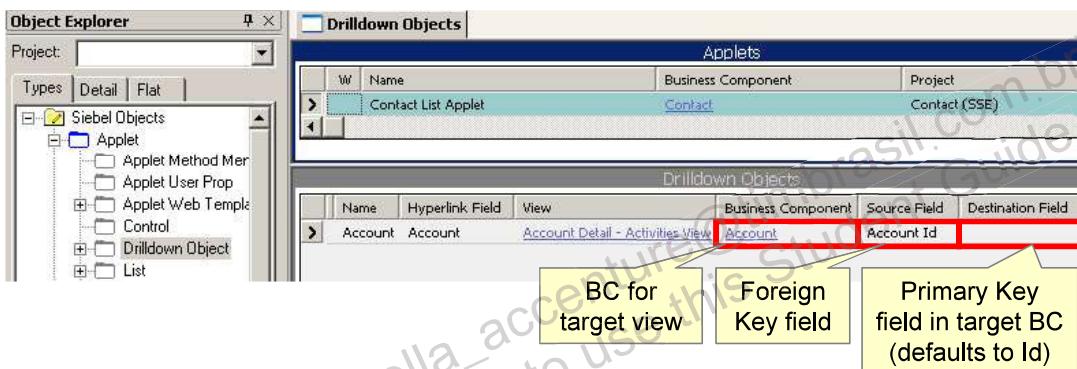
## Static Drilldown: Different Business Component

- Drill down to a detail view for a related record
  - Destination view uses a different business object and business component (BC)



## Configuring Drilldown to a Different Business Component

- Create a drilldown object (child of applet)
  - Set the Hyperlink Field and View properties as before
  - Specify how to identify the related record:
    - Business Component: the BC the target view is based on
    - Source Field: the foreign key field that points to the target BC
    - Destination Field: the primary key field on the target BC



## Dynamic Drilldown

- Enables drilldown to different views from the same hyperlink field
  - Determined by a value in a field for the active record

The screenshot illustrates the dynamic drilldown feature in Siebel. On the left, the 'All Opportunities Across Organizations' list view shows two opportunities: 'Life Insurance Q4 2000' and 'BTC - \$50M High Yield Debt'. The 'Life Insurance Q4 2000' record is selected and highlighted with a red box. A red arrow points from this record to the right-hand 'Policies / Quotes' view, which is also highlighted with a red box. This view displays detailed information for the selected opportunity, including Name, Account, Revenue, Currency, and Sales Stage. A yellow callout box on the left says 'For Fixed Income opportunities go to Opportunity Detail view'. Another yellow callout box on the right says 'For Life Insurance opportunities go to Policies/Quotes view'. The Oracle logo is visible at the bottom right of the interface.

17 - 8

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### Dynamic Drilldown

In the example shown in the screen shot, the Deal Type field in the Opportunity business component is used to determine the destination view. The Deal Type field typically appears with a caption Product Line in opportunity form applets.

## Determining the Target View

- Identify one or more matching conditions to trigger drilling down to a specific target view
  - For example:
    - If Deal Type = Life Insurance, go to INS Opportunity Detail - Policy View
    - If Deal Type = Fixed Income, go to FINCORP Deal Debt View
    - Else go to Opportunity Detail - Contacts View
- Determine the order to check the matching conditions
  - Conditions can involve different fields
  - Multiple matches could occur for a given record

## Configuring Dynamic Drilldown

- Create multiple drilldown objects for the same hyperlink field
  - Each specifies a different target view
- Assign Sequence Numbers to determine the default drilldown object for the set

The screenshot shows the Siebel Object Explorer interface. On the left, under 'Siebel Objects' > 'Applet' > 'Drilldown Object', there is a list of four objects: 'Line of Business', 'Property Insurance View', 'Life Insurance View', and 'Commercial View'. A callout box points to this list with the text: 'These drilldown objects have the same Hyperlink field but have different target views'.

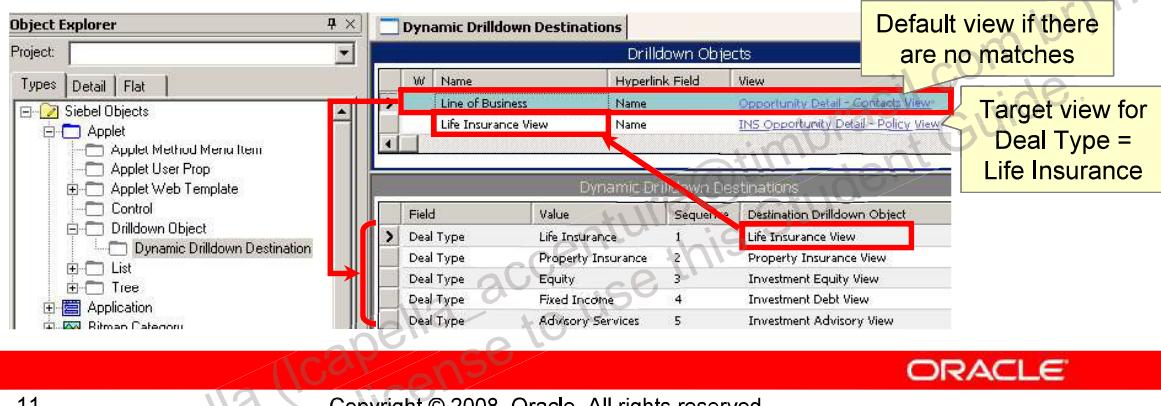
On the right, the 'Drilldown Objects' window displays a table with four rows:

Name	Hyperlink Field	Sequence	View
Line of Business	Name	2	Opportunity Detail - Contacts View
Property Insurance View	Name	3	INS Opportunity Detail - Policy View
Life Insurance View	Name	6	INS Opportunity Detail - Policy View
Commercial View	Name	7	FINCORP Deal Detail View

A callout box points to the 'Sequence' column with the text: 'Drilldown object with lowest sequence number is the default drilldown object'.

# Specifying the Conditions

- For the default drilldown object, configure a dynamic drilldown destination for each condition
  - Specify the Field, Value, and Sequence properties
  - Specify the Destination Drilldown Object property to correspond to the desired target view
- If no default condition is configured, the target view is specified by the parent drilldown object



17 - 11

Copyright © 2008, Oracle. All rights reserved.

ORACLE

## Specifying the Conditions

### Sequence

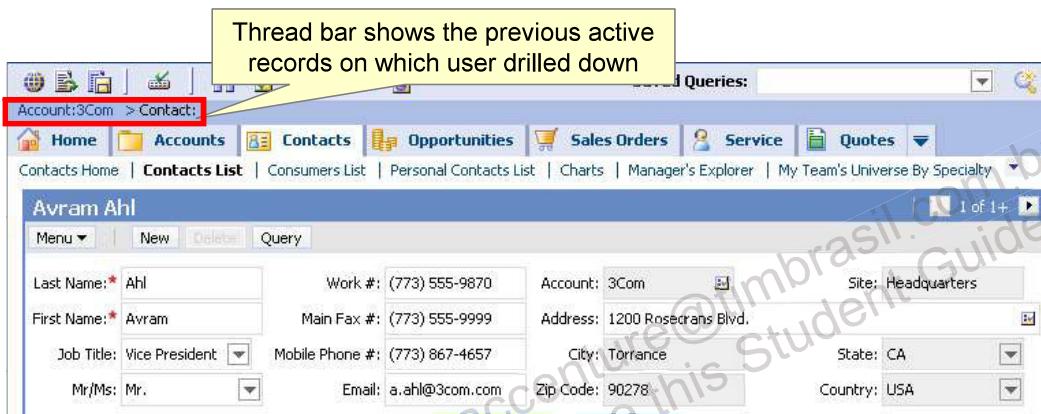
- The sequence property for a dynamic drilldown destination object definition determines the order in which the conditions are checked. The first match is used to determine the drilldown object with the target view. A sequence number is required since the conditions may not be mutually exclusive.

### Determining the Destination View

- As shown in the previous slide the Line of Business drilldown object has the lowest sequence number so its dynamic drilldown destinations are checked. If the Deal Type = Life Insurance, the search stops after the first dynamic drilldown destination is checked. This specifies a destination drilldown object of Life Insurance View. This destination object (a sibling of the parent object definition) is then checked to determine that the target view is INS Opportunity Detail - Policy View.
- If the search through the dynamic drilldown destinations does not result in a match, then the view specified in the parent drilldown object (Line of Business) becomes the target view.

## Thread Bar

- Tracks the previous active record
  - Provides a hyperlink to the previous view
- Is updated whenever the user drill downs to a different business object



17 - 12

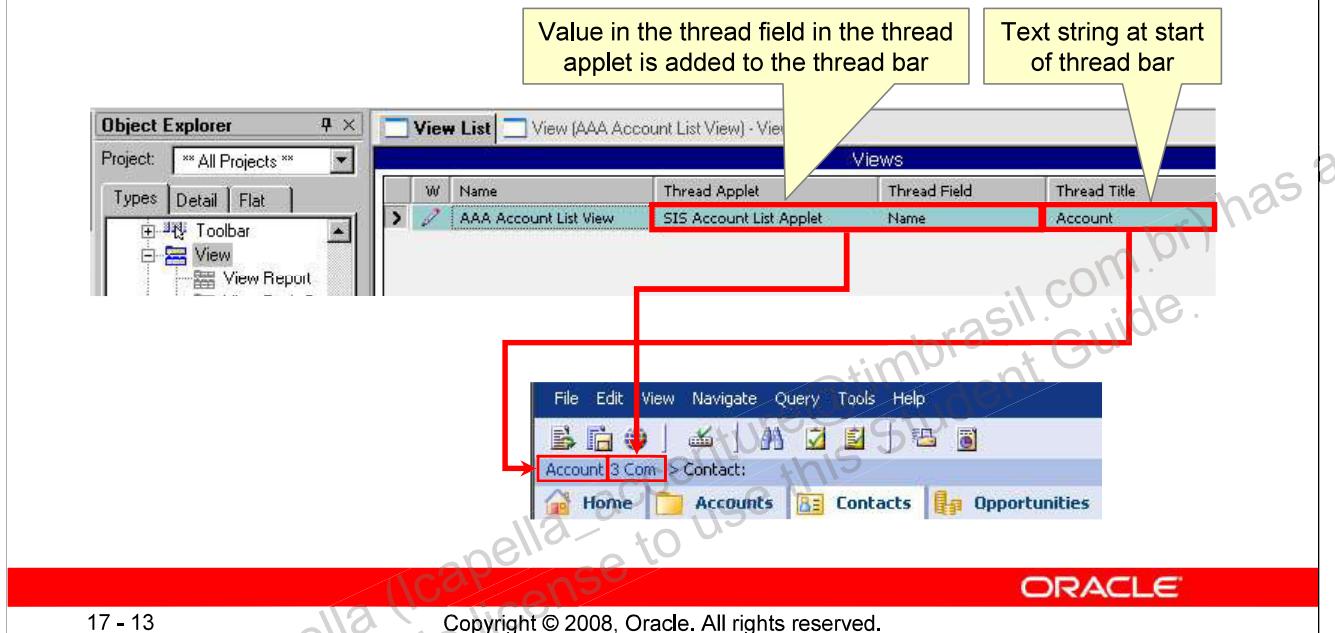
Copyright © 2008, Oracle. All rights reserved.

### Thread Bar

The browser back button can also be used to return to the previous record.

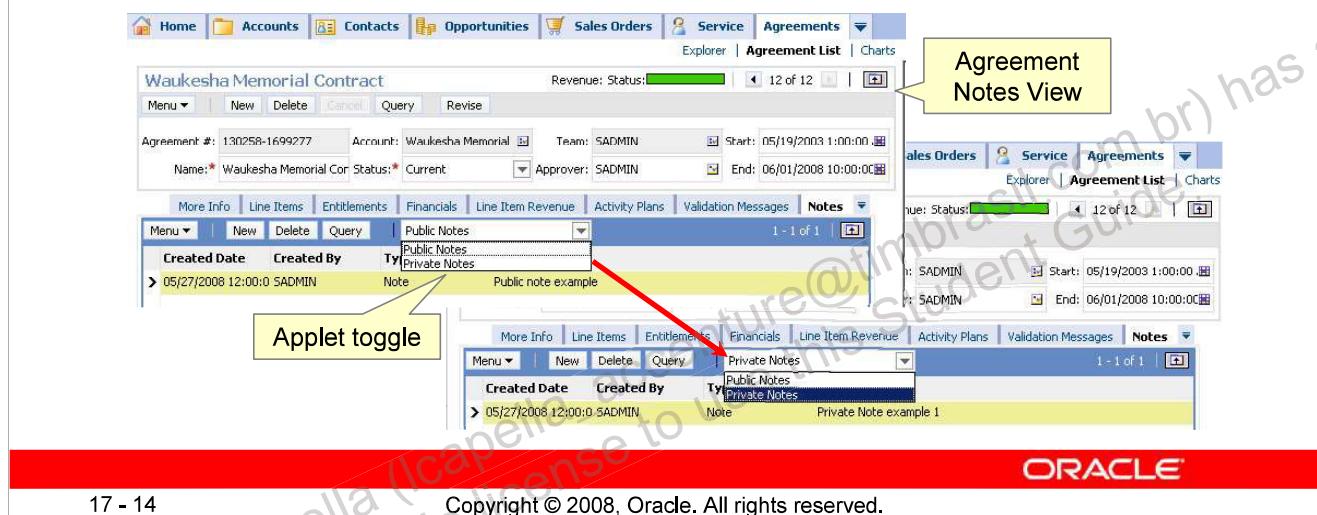
## Configuring the Thread Bar

- Set the thread bar properties to generate the thread bar to track user navigation



## Applet Toggles

- You can configure several applets to share the same space in a view
  - You only add one applet to the view Web template
  - Users can switch this applet with others by an applet toggle that you define on the applet

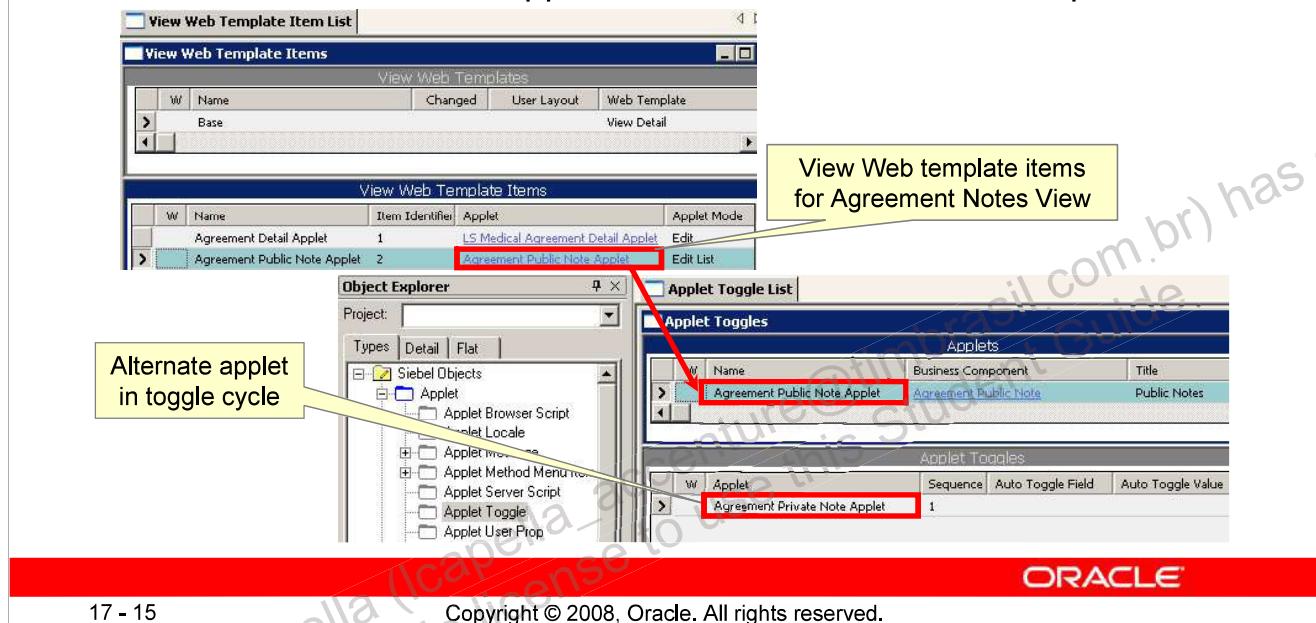


### Applet Toggles

Reference: “Configuring Screens and Views” in *Configuring Siebel Business Applications*

## Configuring Applet Toggles

- Create a new applet toggle definition for each applet to be added to the toggle list
  - Create it on the applet defined in the view Web template



## Togglebar Tag

- The toggle list will appear only if the applet template includes a SWE *togglebar* tag
  - Togglebar is rendered as a drop-down select control
  - Example:

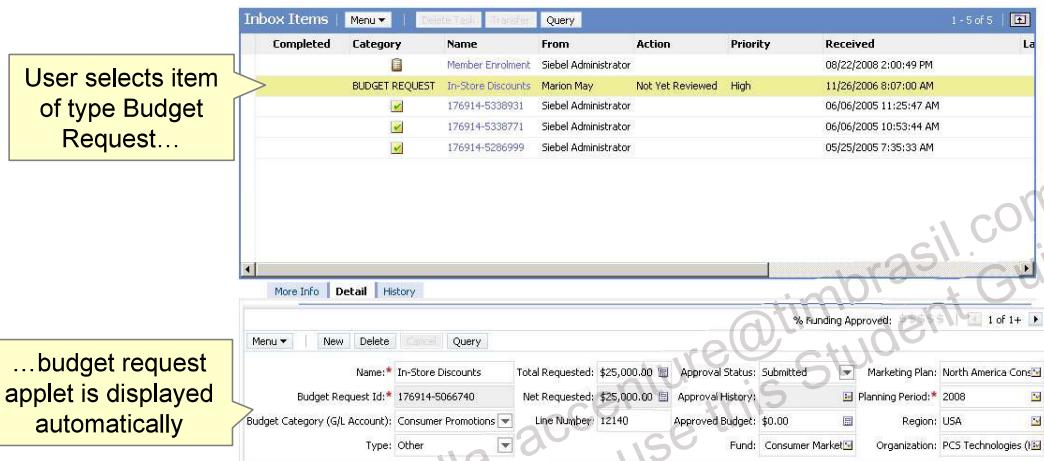
Example html  
from a template

```
<swe:togglebar type="Select">
<table>
<tr> <td> <swe:control id="1"
property="DisplayName" > </td>
<td> <swe:this property="FormattedHtml"/> </td>
</tr>
</table>
</swe:togglebar>
```



## Dynamic Applet Toggling

- Automatically selects the applet to be displayed in a view based on the value of a field in the underlying business component
  - User does not explicitly select a toggle applet



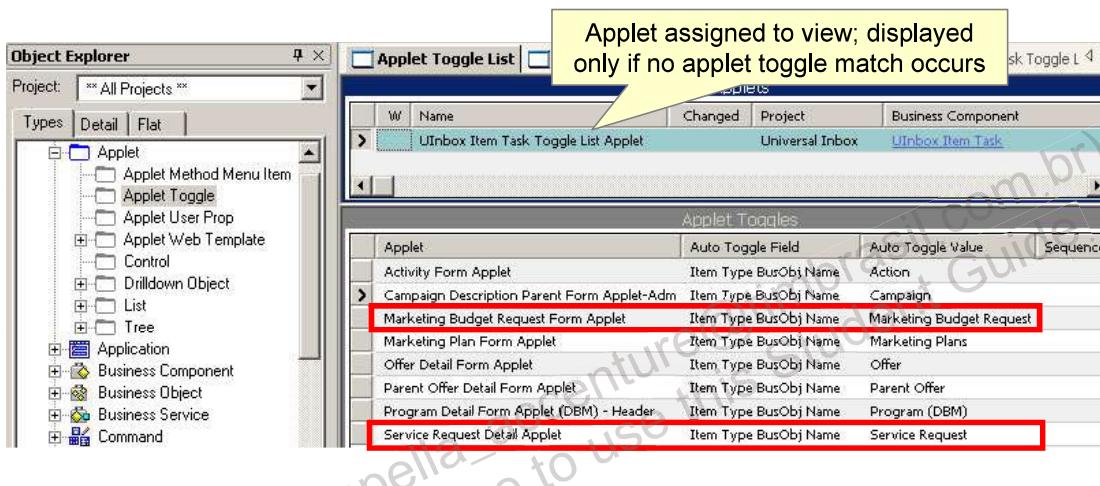
## Dynamic Applet Toggling

- When user selects a different inbox item, the lower applet toggles to an applet determined by the category of the item

The screenshot illustrates the dynamic applet toggling feature in Siebel 8.1.x. At the top, there is an 'Inbox Items' list view showing five items. The second item, 'BUDGET REQUEST' (Category: In-Store Discounts), is highlighted with a yellow background and has a checkmark next to it. A callout bubble points to this item with the text 'User selects item of type SR...'. Below the list view, a detailed 'SR Information' applet is displayed. This applet includes fields for SR #, Type, Status, Last Name, Area, Sub Status, First Name, Sub Area, and Priority. A callout bubble points to this applet with the text '... SR specific applet is displayed automatically'. At the bottom of the screen, there is a red footer bar with the text 'ORACLE' and 'Copyright © 2008, Oracle. All rights reserved.'

## Configuring Dynamic Applet Toggling

- Create a child applet toggle for each possible field value
  - Set the Auto Toggle Field and Auto Toggle Value properties
  - Assign a sequence if conditions are not mutually exclusive
- If there are no matches, parent applet is displayed



### Configuring Dynamic Applet Toggling

The default toggle applet is the UIInbox Item Task Toggle List Applet. Although it is a list applet, it is rendered in the view in the Edit mode and appears as a form applet.

## Lesson Highlights

- List columns or controls can be configured for drilldown
- Drilldowns can be static or dynamic
  - Static drilldowns always navigate the user to the same target view
  - Dynamic drilldowns navigate the user to a target view determined by the values shown in the current view
  - Static drilldowns can be configured to drill down to the same business component or to a different business component
- Toggle applets allow several applets to share the same space in a view
  - Applets are cycled by user selection
- Dynamic applet toggles display different applets in a view based on the value of a field in the business component

## Practice 17 Overview: Configuring Drilldowns and Applet Toggles

This practice covers the following topics:

- Configuring drilldown
- Configuring a dynamic drilldown
- Configuring a dynamic toggle

} *Choose one to complete*

Unauthorized reproduction or distribution prohibited. Copyright© 2014, Oracle and/or its affiliates.

Leonardo Cappella (lcapella\_accenture@timbrasil.com.br) has a  
non-transferable license to use this Student Guide.

# 18

## Configuring Picklists

ORACLE

Copyright © 2008, Oracle. All rights reserved.

# Objectives

After completing this lesson you should be able to:

- Describe the differences between dynamic and static picklists
- Administer a static picklist
- Configure a static or dynamic picklist

# Picklists

- Allow users to populate one or more single-value fields by selecting a value from a list
  - Enforces business rules and policies
  - Makes data entry faster
  - Reduces errors
- Can be either static or dynamic



18 - 3

Copyright © 2008, Oracle. All rights reserved.

ORACLE

## Static Picklist

- Displays values in a drop-down list for user selection
- Copies the selected value into a destination field
  - There is no link to the original picklist data
- Can be bounded or unbounded
  - Bounded picklist forces users to enter only a value in the picklist
  - Unbounded picklist permits users to enter any value into the field
    - User entered value is not added to the picklist values
- Draws values from picklist data managed by an administrator
  - Values displayed in static picklists do not change during run time as result of user activity

## Static PickList Values

- Are stored for all static picklists in the S\_LST\_OF\_VAL table
  - Is a user data table in the application database
- Have a type that identifies the static picklist the value belongs to

The screenshot shows the Siebel List of Values interface. The top navigation bar includes Home, Accounts, Contacts, Opportunities, Sales Orders, Administration - Data, and a red-highlighted 'List of Values' tab. Below the navigation is a toolbar with New, Delete, Query, and Clear Cache buttons. A status bar at the bottom indicates '1 - 10 of 10+'. The main area displays a table with columns: Type, Display Value, Order, Active, and Language-Independent Code. The table lists five entries for 'MR\_MS': Dr. (Order 4), Ms. (Order 2), Miss (Order 1), Mr. (Order 1), and Mrs. (Order 3). A sixth row, highlighted with a yellow background, contains 'LOV\_TYPE' in the Type column and 'MR\_MS' in the Display Value column. A red double-headed arrow points from the 'Type' column header to this row. A red box highlights the 'List of Values' tab in the top navigation bar and the 'LOV\_TYPE' row in the table.

Type	Display Value	Order	Active	Language-Independent Code
MR_MS	Dr.	4	✓	Dr.
MR_MS	Ms.	2	✓	Ms.
MR_MS	Miss	1	✓	Miss,
MR_MS	Mr.	1	✓	Mr,
MR_MS	Mrs.	3	✓	Mrs,
LOV_TYPE	MR_MS		✓	MR_MS

ORACLE

### Static PickList Values

Reference: “Creating and Administering Lists of Values” in *Configuring Siebel Business Applications*

## Administering Static Picklists

- Use the List of Values Explorer to administer static picklist data
  - Navigate to Administration - Data > LOV Explorer
  - Select an existing picklist
  - Expand the type and select the child Values folder
  - Edit the picklist values in the List of Values applet

The screenshot shows the Siebel List of Values Explorer interface. The left sidebar displays a tree structure under 'Types' for 'MR\_MS', with 'Values' expanded to show items like 'Miss (English-American)', 'Mr. (English-American)', etc. The main panel shows a table titled 'List of Values' with columns: Code, Display Value, Language Name, and Active. The data is as follows:

Code	Display Value	Language Name	Active
> Miss,	Miss	English-American	✓
Mr.	Mr.	English-American	✓
Ms.	Ms.	English-American	✓
Mrs.	Mrs.	English-American	✓
Dr.	Dr.	English-American	✓

Two callout boxes provide instructions: 'Do not change names of Siebel-supplied values' and 'OK to set inactive'.

## Administering Static Picklists Using Siebel Tools

- List of Values can also be administered from Siebel Tools
  - Select Screens > System Administration > List of Values
  - Query for the picklist type
  - Edit values in the object list editor
  - Create a picklist by:
    - Creating a new LOV Type
    - Creating values for the LOV Type

The screenshot shows a Siebel application window titled "List of Values Administration". The main title bar is "List of Values". The window contains a table with the following data:

Type	Display Value	Changed	Translate	Multilingual	Language-Independent Code
MR_MS	Miss	✓	✓	✓	Miss,
MR_MS	Mr.	✓	✓	✓	Mr.,
MR_MS	Ms.	✓	✓	✓	Ms.,
MR_MS	Mrs.	✓	✓	✓	Mrs.,
MR_MS	Dr.	✓	✓	✓	Dr.,

## Administering Static Picklists Using Siebel Tools

Any changes you make to the entries in the LOV table (whether by Siebel Tools or the Siebel client application) must be migrated from the development database to the test and production databases. Entries in the list of values are not complied into the .srf file, nor can they be checked into the server repository.

## Dynamic Picklist

- Displays values in a pick applet
- Draws values from a business component (BC) whose records are edited by users
  - Values are dynamic and depend on current BC records
- Copies the primary key of the selected record into the foreign key field
  - Updates the values of joined fields displayed in the applet



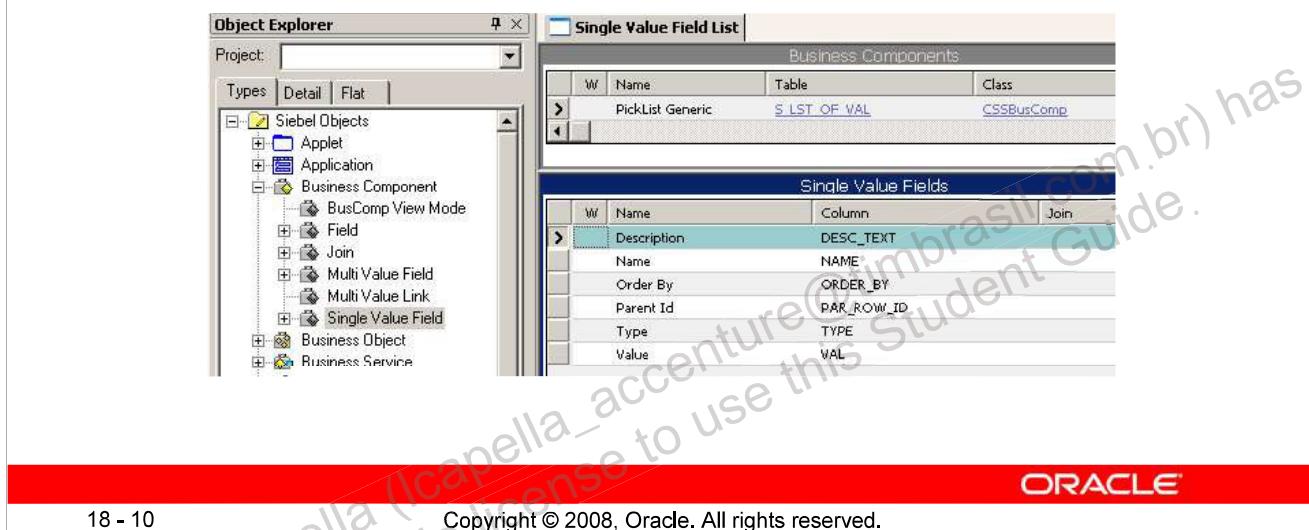
## Picklist Terms

- Siebel picklists:
  - Are associated with a field in the *originating business component*
  - Draws values from a *pick business component*
    - Copies values from one or more fields in the pick BC into corresponding fields in the originating BC



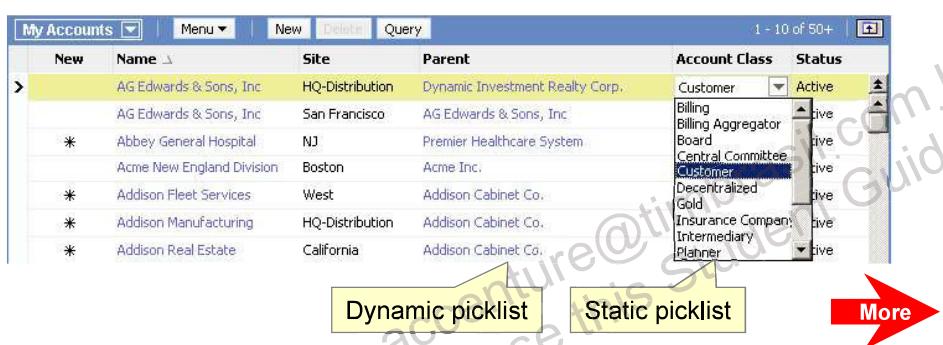
## PickList Generic Business Component

- Is the business component in the Siebel repository that references the S\_LST\_OF\_VAL table
- Serves as the pick business component for Static picklists
  - Static picklists are based on PickList Generic



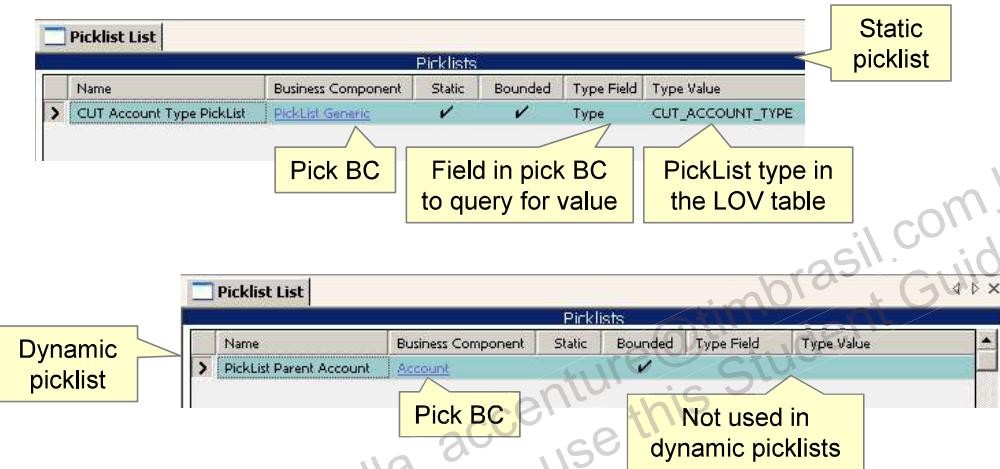
## Object Definitions for PickLists

- Implementing picklists requires configuring several object definitions:
  - PickList
  - Single Value Field and SVF Pick Map
  - Pick Applet (required for dynamic picklists)
  - Control and List Column



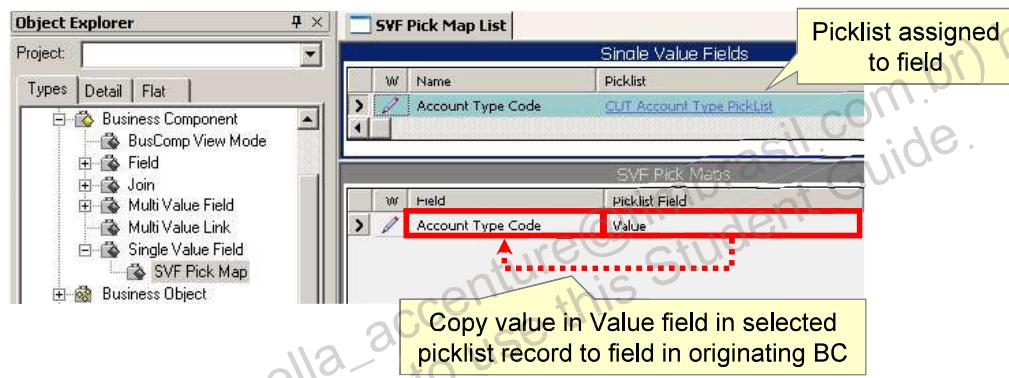
## PickList Object Definition

- Defines a static or a dynamic picklist
  - Identifies the pick business component
  - Specifies if the picklist is bounded



## Single Value Field and SVF Pick Map

- Picklist property in Single Value Field specifies which picklist is associated with the field
  - Users see a picklist at run time on applets displaying this field
- SVF Pick Maps identify one or more fields to be copied from the pick BC to the originating BC



## SVF Pick Maps

- Several fields from a picklist may be copied into the originating BC
  - For dynamic picklists, the following fields are typically copied
    - Foreign key field to identify the picked record
    - Additional fields to update fields displayed in the UI

The screenshot shows the Siebel Object Explorer on the left and the SVF Pick Map List on the right. The Object Explorer displays various business components and their fields. The SVF Pick Map List shows a mapping between a Single Value Field (SVF) and a Picklist Field. The mapping includes fields like Master Account Id, Parent Account Id, Parent Account Name, and Parent Account Type Code. A yellow callout points to the Parent Account Id field in the SVF Pick Maps table, stating: "FK field populated with the Id of the picked record". Another yellow callout points to the Parent Account Name field in the same table, stating: "Name field copied to be displayed in the applet".

SVF	Picklist Field
Master Account Id	Master Account Id
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

18 - 14

Copyright © 2008, Oracle. All rights reserved.

ORACLE

### SVF Pick Maps

There are numerous dynamic picklists in the as-delivered repository which are unbounded. In these cases the SVF pick map typically copies the value from a field, other than the primary key field, in the pick business component to a field in the originating business component. There is no mechanism to update the value in the originating business component if the value changes in the pick business component. Since the picklist is unbounded users are not restricted by the choices in the picklist, and can enter another value if desired.

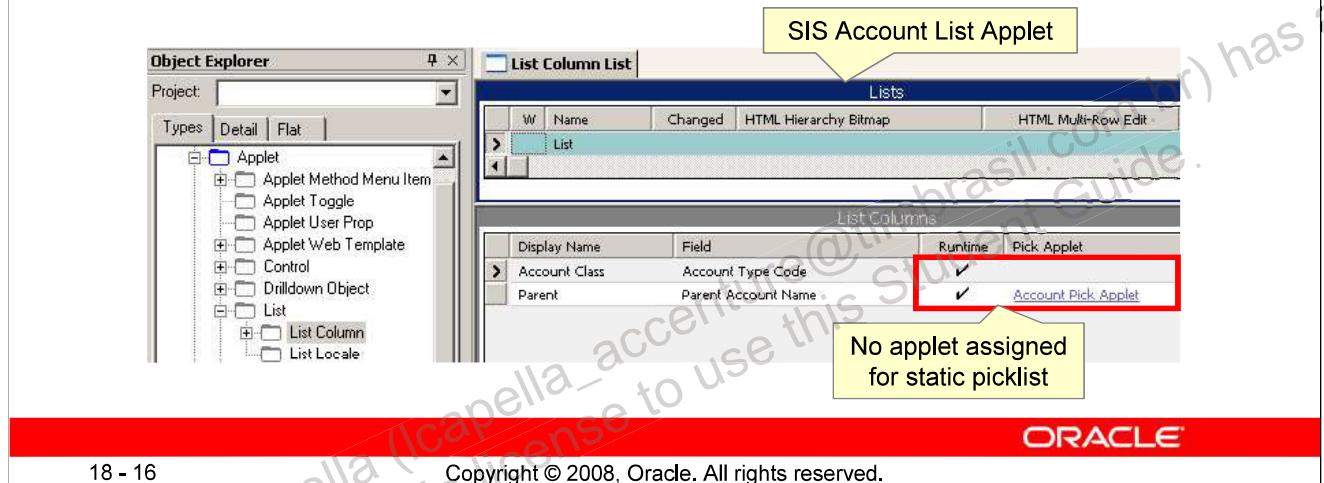
## Pick Applet

- Is a floating applet used to display records for dynamic picklists
  - Typically displays multiple columns from the pick BC to assist users with selecting the desired record



## Control and List Column

- Set Runtime to TRUE to enable the picklist in the applet
- For dynamic picklists, specify a Pick Applet property
  - Identifies the applet to display the pick business component
- For static picklists, leave the Pick Applet property blank
  - Drop-down list is generated at run time

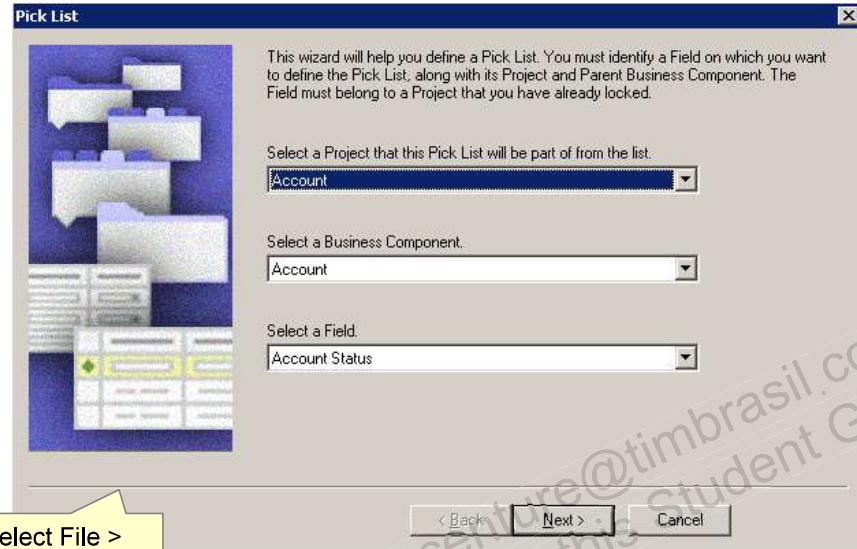


## Control and List Column

When a control or a list column has its Runtime property set to TRUE, the application at run time is directed to display a supplementary icon for the field, based on the values of Pick Applet and MVG Applet. For instance if a pick applet is specified, a single select button is displayed. In addition if neither applet is specified, then the application examines the type of underlying field to determine if an icon for a calculator, calendar, or currency pop-up applet should be displayed.

## Creating a Picklist

- Use the Pick List wizard to create a new picklist



18 - 17

Copyright © 2008, Oracle. All rights reserved.

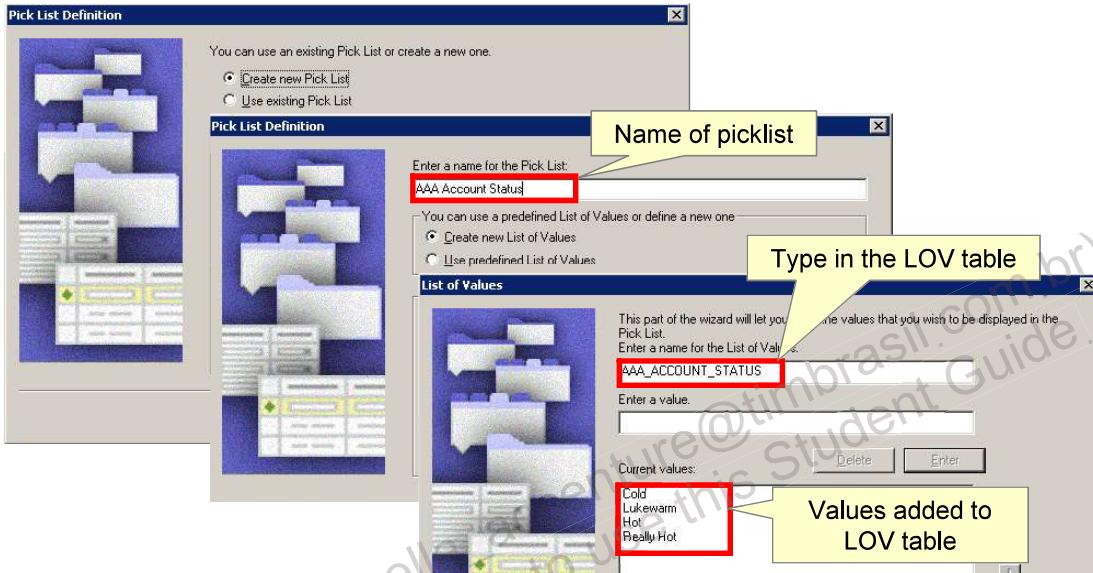
ORACLE

## Pick List Wizard Inputs for Static Picklists

- The project to contain the new picklist
- The originating business component and field to assign the picklist
- Type of picklist (static)

## Defining the Static Picklist

- Wizard allows using either an existing picklist or creating a picklist based on a new set of list of values



## Pick List Wizard

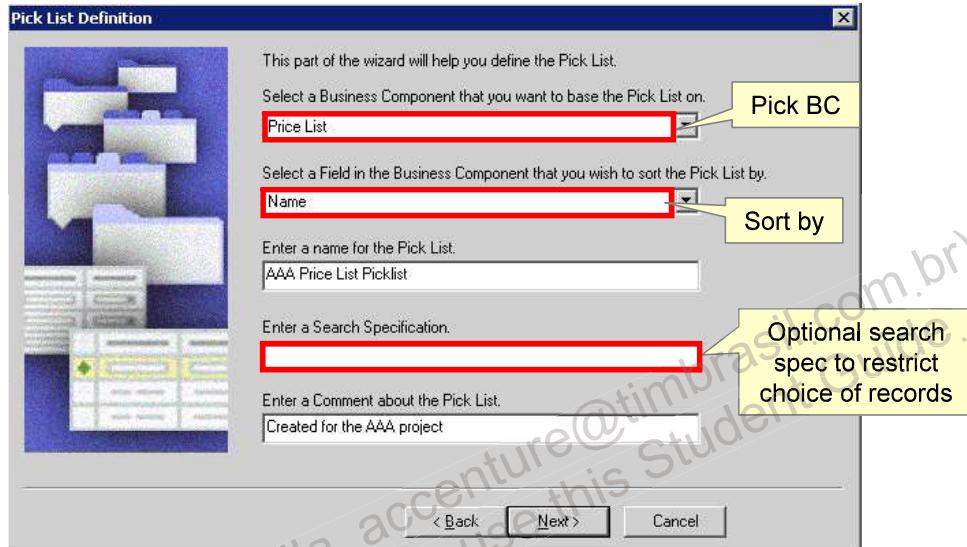
- Creates the following:
  - Values in the S\_LST\_OF\_VAL table
  - A static PickList object definition that references the specified LOV values
  - A SVF Pick Map to copy the LOV value to the specified field in the originating BC
- Sets Runtime to TRUE in the controls or list columns to activate the picklist for users

## Pick List Wizard Inputs for Dynamic Picklists

- Project to contain the new picklist
- The originating business component and field to assign the picklist
- Type of picklist (dynamic)

## Defining the Dynamic Picklist

- Wizard allows either using an existing dynamic picklist or creating a new dynamic picklist



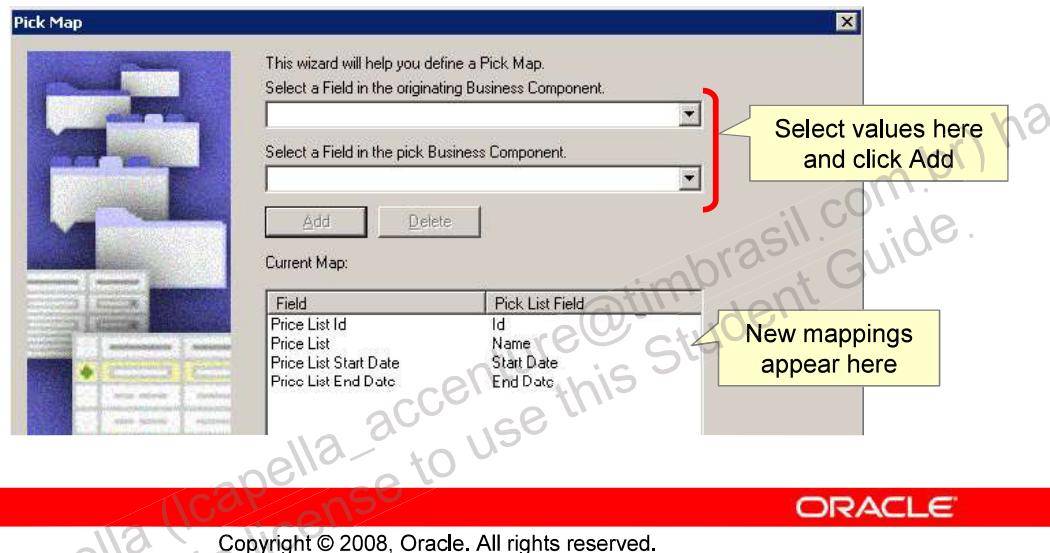
18 - 22

Copyright © 2008, Oracle. All rights reserved.

ORACLE

## Defining the Pick Maps

- Specify the editing requirements
- Create one or more pick maps
  - Map the primary key in the pick BC to the FK in the originating BC



## Pick List Wizard

- Creates the following:
  - Picklist object definition that references the pick BC
  - SVF Pick Maps for the BC field with the picklist
- Sets Runtime to TRUE in the controls or list columns to activate the picklist for users
- Invokes the Applet wizard to create a pick applet if required

## Constrained Picklist

- Filters values dynamically to display only those records with one or more fields that match corresponding fields in the originating BC record

The screenshot shows two windows. The top window is a grid titled 'All Agreements' with columns: Agreement #, Name, Type, Status, Account, Last Name, First Name, and Valid. It lists three rows of data. The bottom window is a 'Pick Contact' dialog from Microsoft Internet Explorer, titled 'http://edpsr43p1:8080 - Pick Contact - Microsoft Internet Explorer'. It has a search bar 'Find: Last Name' and a dropdown 'Starting with: <Case Required>'. A red arrow points from the 'Account' column in the grid to the 'Account' column in the picklist. A callout box below the picklist says 'Displays only contacts from the account in the parent record'. The picklist shows three contacts: Able, Mann, and Marlow, all associated with 'HT 51 Active Systems - HQ'.

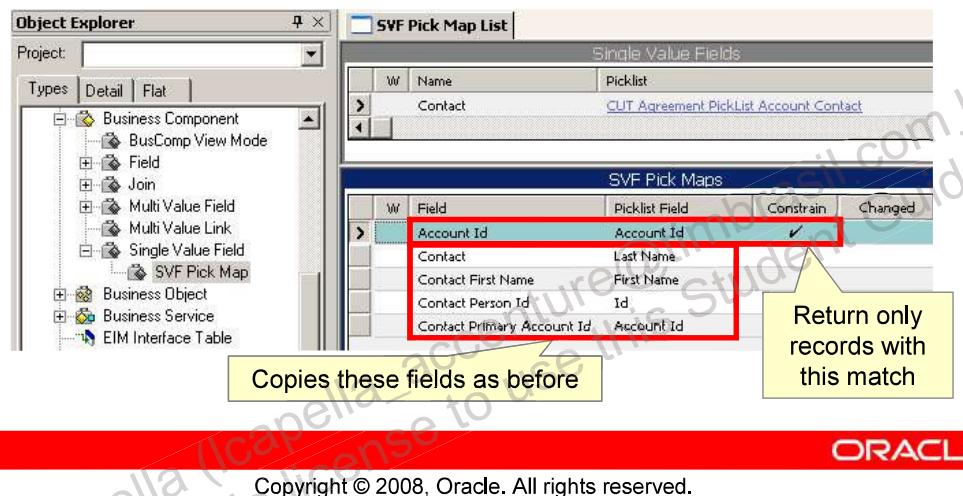
Agreement #	Name	Type	Status	Account	Last Name	First Name	Valid
83602-556780	2003 Albany Hospital	Service Level Agree	Active	Albany General Hospital	Anjani	Mohit	✓
83602-510601	Abbey Master Contr	Contract	Current	Abbey General Hospital	Shaw	Cynthia	✓
> 176914-4448956	Active Systems Prici	Price Protection	Current	HT 51 Active Systems - HQ			✓

Last Name	First Name	Status	Account	Site	Job Title	Work Phone #
Able	Janet		HT 51 Active Systems - HQ	HQ	Admin Assistant	(415) 832-4242
Mann	Owen		HT 51 Active Systems - HQ	HQ	Partner Operations I	(650) 434-2231
Marlow	Robin		HT 51 Active Systems - HQ	HQ	Partner Sales Manaç	(650) 434-2230

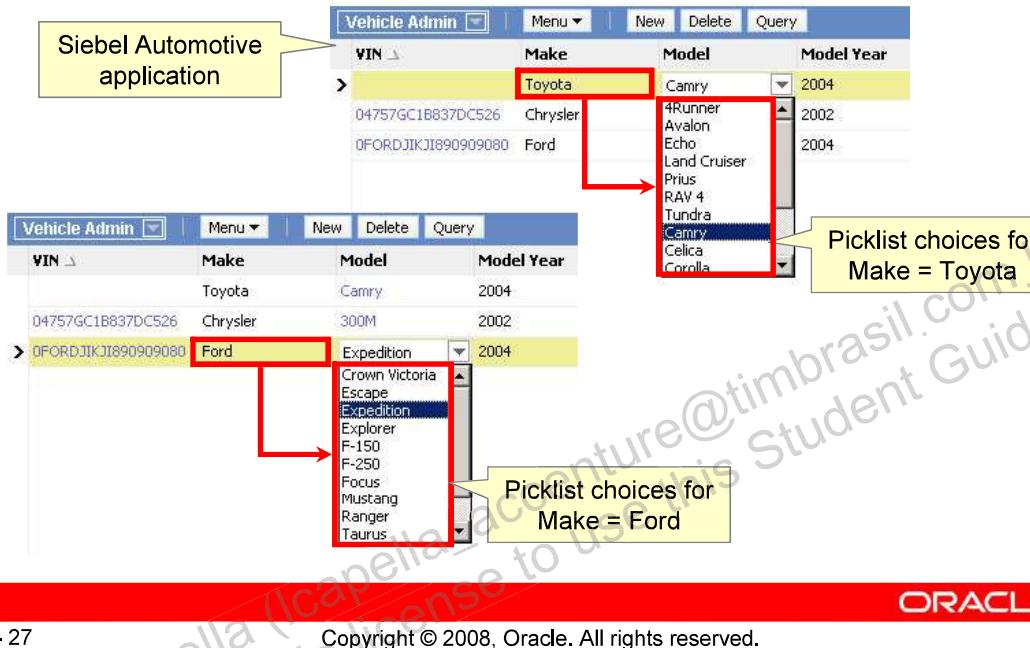
## Constraining a Picklist

- Create the regular pick maps
- Create a SVF Pick Map for each field that must match a field in the originating business component
  - Set the Constrain property to TRUE for the matching fields
  - Filters the pick BC records for matches



## Hierarchical Picklists

- Displays values that are constrained by the value in another picklist



## LOVs for Hierarchical Picklists

- List of values in the S\_LST\_OF\_VAL table are organized hierarchically
  - All LOVs in hierarchy have the same type

The screenshot shows two Siebel List of Values (LOV) windows. The top window displays a hierarchy where 'Make' is the parent category for 'Toyota'. The bottom window shows the detailed list of car models under 'Toyota', including '4Runner', 'Avalon', 'Echo', 'Land Cruiser', 'Prius', and 'RAV 4'. Red boxes highlight the 'Display Value' column in both tables, and arrows point from the 'Make' entry in the top table to the 'Toyota' entry in the bottom table, and from the 'Toyota' entry in the bottom table back to the 'Make' entry in the top table, illustrating the bidirectional relationship in the hierarchy.

Picklist choices for Make = Toyota

Type	Display Value	Language-Independent Code	Parent LIC	Language Name
AUTO_MAKE_TYPE	Make	Make		English-American
AUTO_MAKE_TYPE	Toyota	Toyota	Make	English-American

Type	Display Value	Language-Independent Code	Parent LIC	Language Name
AUTO_MAKE_TYPE	4Runner	4Runner	Toyota	English-American
AUTO_MAKE_TYPE	Avalon	Avalon	Toyota	English-American
AUTO_MAKE_TYPE	Echo	Echo	Toyota	English-American
AUTO_MAKE_TYPE	Land Cruiser	Land Cruiser	Toyota	English-American
AUTO_MAKE_TYPE	Prius	Prius	Toyota	English-American
AUTO_MAKE_TYPE	RAV 4	RAV 4	Toyota	English-American

ORACLE

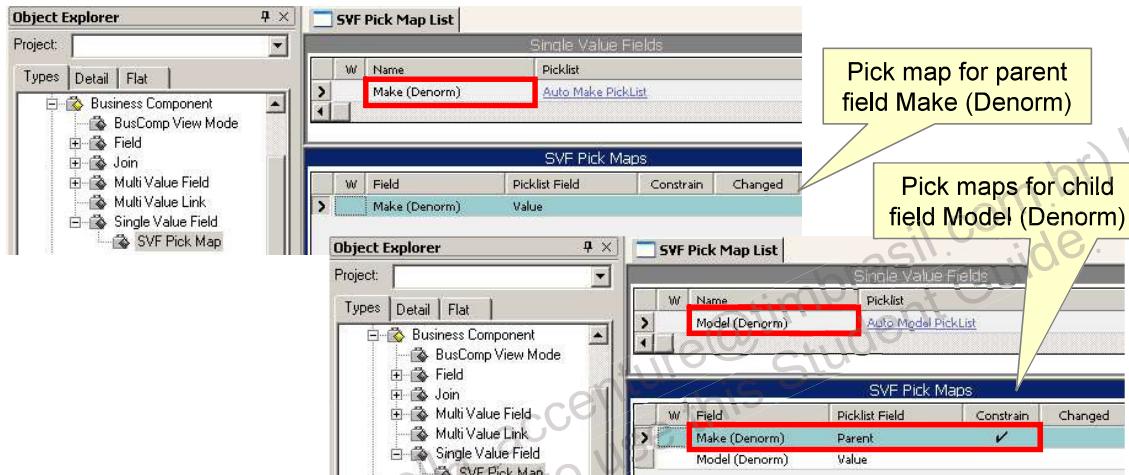
## Configuring a Hierarchical Picklist

- Set the BC property to PickList Hierarchical
  - Provides functionality to manage hierarchical picklists
- Add a search specification to the Auto Make Picklist to retrieve only those LOVs with Parent = Make

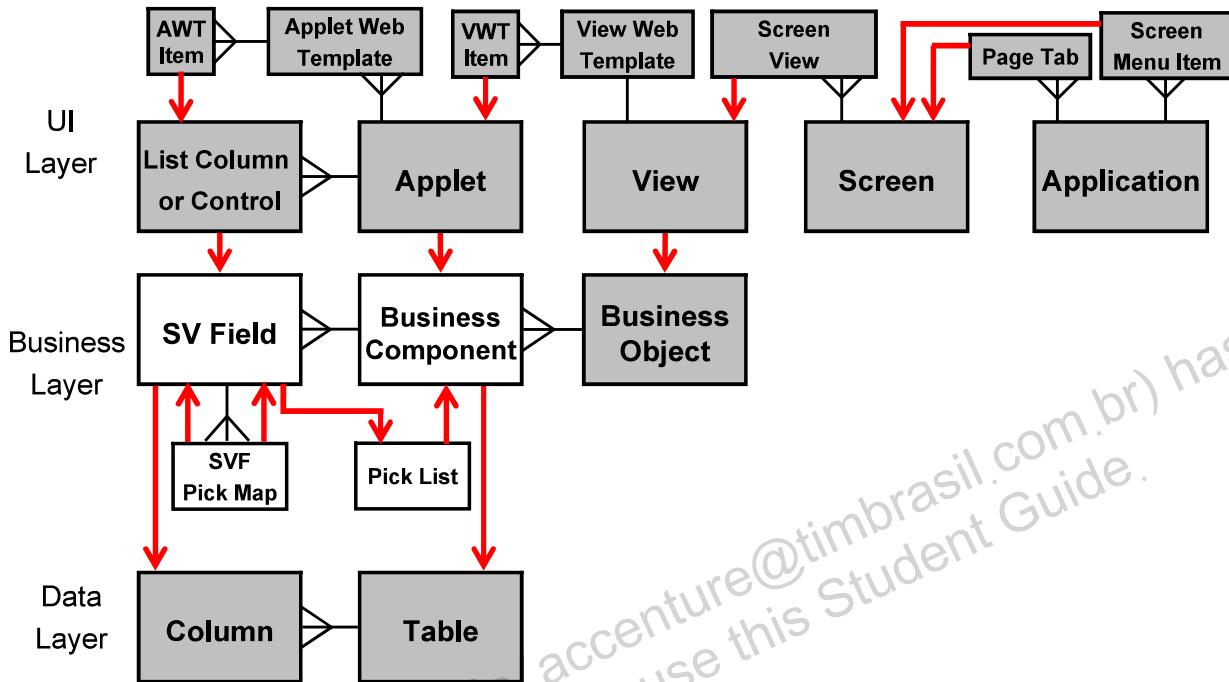
Name	Business Component	Static	Bounded	Type Field	Type Value	Search Specification
Auto Make PickList	PickList Hierarchical	✓	✓	Type	AUTO_MAKE_TYPE	Parent = LookupValue('AUTO_MAKE_TYPE', 'Make')
Auto Model PickList	PickList Hierarchical	✓	✓	Type	AUTO_MAKE_TYPE	

## Configuring the SVF Pick Maps

- For the parent field, create a pick map to copy the selected picklist value into the target field in the originating BC
- For the child field, add a constraint based on the value of the model selected



## Summary of Object Types



### Summary of Object Types

For clarity the detailed object types for joins and links are not shown on this rendition. Also note that there are two references from SVF Pick Map to SVF Field. This indicates that a SVF Pick Map references a field in the pick business component and another field in the originating business component.

## Lesson Highlights

- Static picklists:
  - Display values in a drop-down list
  - Contain static values, which are managed through List of Values administrative views
  - Store values in S\_LST\_OF\_VAL table
- Dynamic picklists:
  - Display values in a pick applet
  - Contain dynamic data, which is typically the result of user transactions
  - Access data in pick business component using a foreign key
- Picklists are created using Siebel Tools' Pick List Wizard
- Picklists can be configured to display a constrained set of values that depend upon the value of another field

## Practice 18 Overview: Configuring Picklists

This practice covers the following topics:

- Examining picklists
- Configuring a static picklist
- Configuring a dynamic picklist *(Optional)*