

# ADF Code Corner

## 67. How-to create a query form in a popup dialog



[twitter.com/adfcodecorner](https://twitter.com/adfcodecorner)

### Abstract:

In this article I explain how to create a search form that opens in an af:popup dialog. Defining the search criteria and executing the query closes the dialog and refreshes the table. The solution uses the af:query component for building the searchform, which is a simple and straight forward approach that many may not be aware of.

Author:

Frank Nimphius, Oracle Corporation  
[twitter.com/fnimphiu](https://twitter.com/fnimphiu)  
11-JAN-2011

*Oracle ADF Code Corner is a loose blog-style series of how-to documents that provide solutions to real world coding problems.*

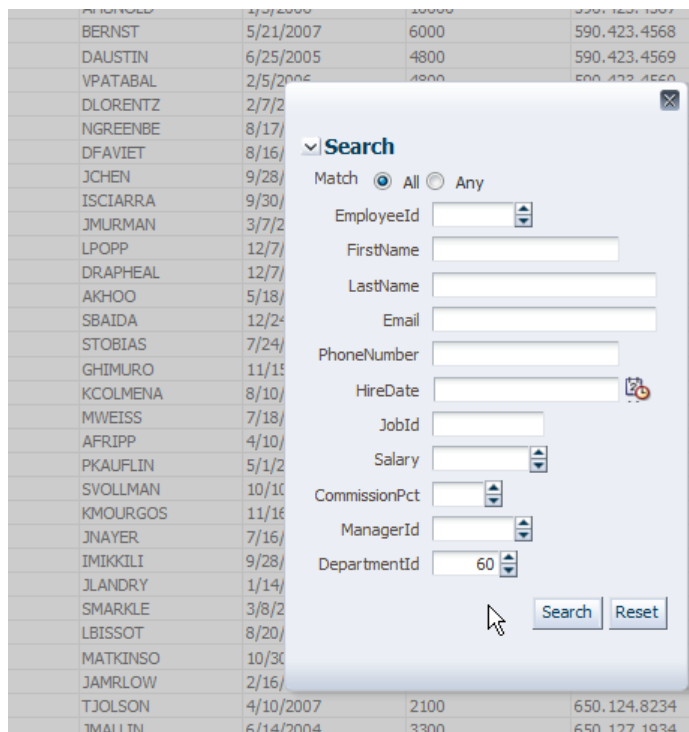
*Disclaimer: All samples are provided as is with no guarantee for future upgrades or error correction. No support can be given through Oracle customer support.*

*Please post questions or report problems related to the samples in this series on the OTN forum for Oracle JDeveloper: <http://forums.oracle.com/forums/forum.jspa?forumID=83>*

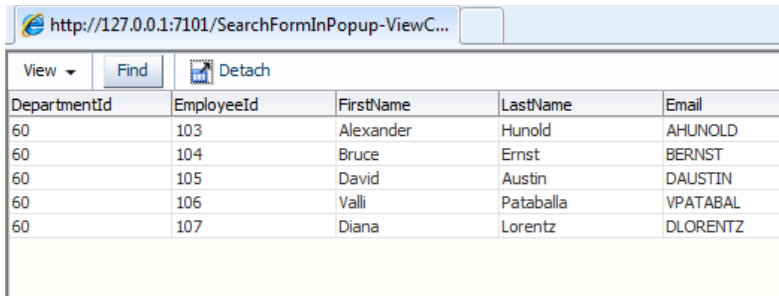
## Introduction

This ADF Code Corner article is in response to a question posted to the Oracle JDeveloper forum on OTN. The requirement was to provide a toolbar option to launch a search form that users use to filter the table content. The form had to be in a popup.

The screenshot below shows the final example. The af:query component is configured to not show the query mode change button or the saved query option.



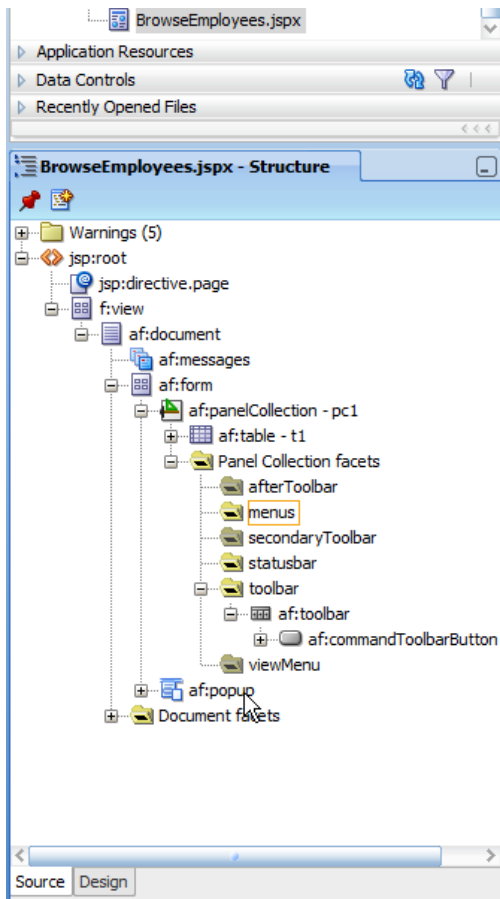
Pressing the search button executes the filtered query and closes the popup dialog. Note that the popup dialog contains an af:dialog component to create a modal popup that doesn't close when clicking outside of the popup area.



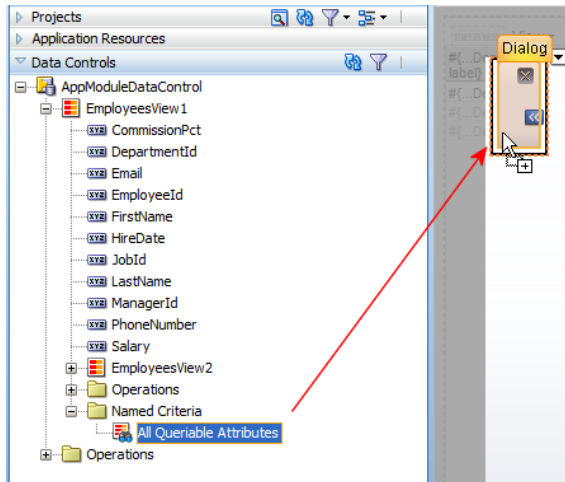
DepartmentId	EmployeeId	FirstName	LastName	Email
60	103	Alexander	Hunold	AHUNOLD
60	104	Bruce	Ernst	BERNST
60	105	David	Austin	DAUSTIN
60	106	Valli	Pataballa	VPATABAL
60	107	Diana	Lorentz	DLORENTZ

## How-to build this Solution

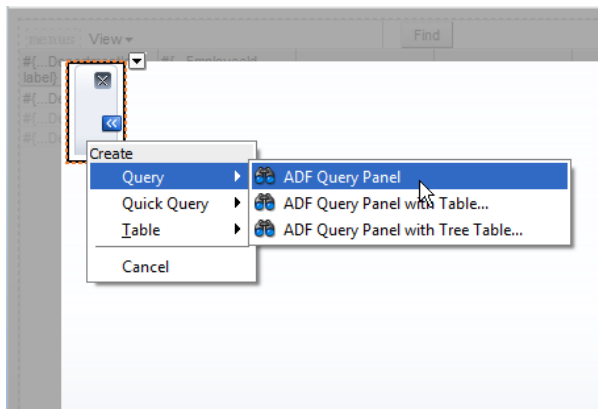
Starting from an ADF Faces page that contains an ADF bound table, contained in an **af:panelCollection** that has an **af:commandToolBarButton** component defined in its **toolbar** facet, and an **af:popup** component:



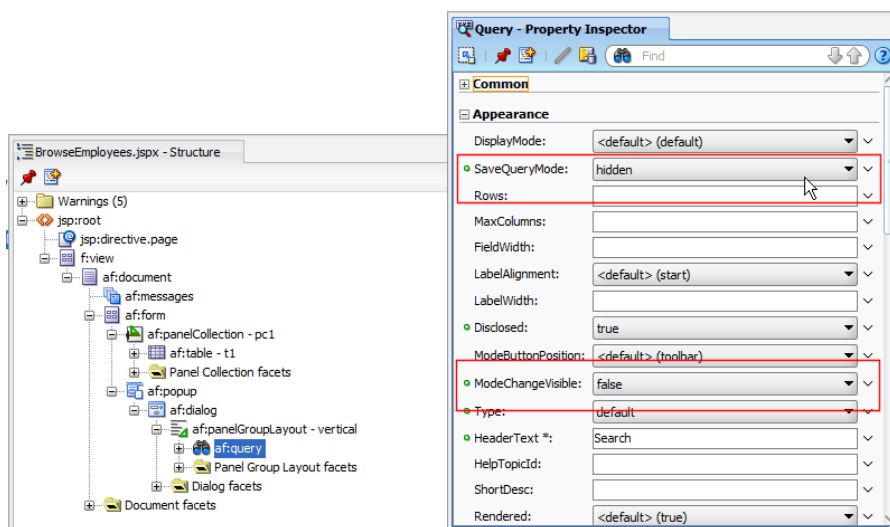
In the Data Control panel, expand the **Named Criteria** node of the Collection (EmployeesView1 in the sample) and drag the **All Queriable Attributes** into the **af:dialog** component contained in the **af:popup**.



In the opened menu, choose **ADF Query Panel**.



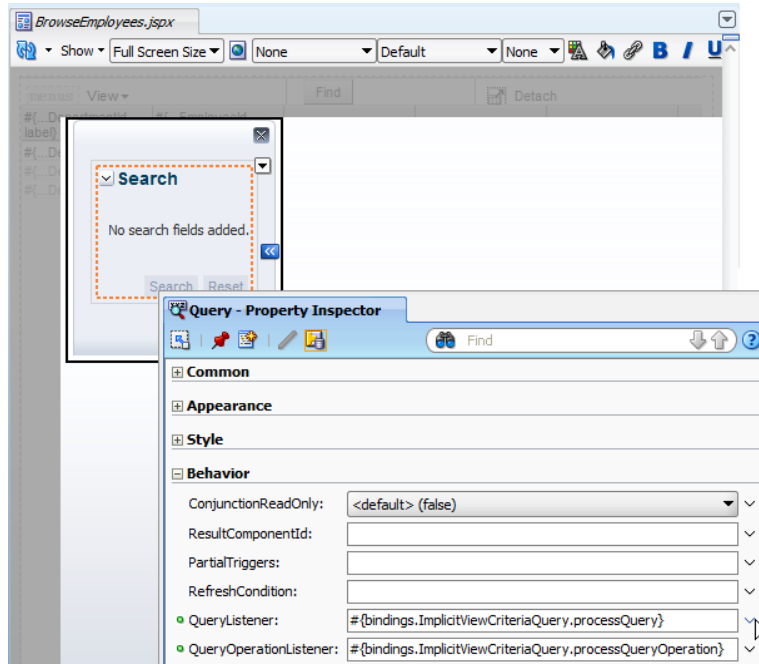
By default, the **af:query** component is surrounded by a panel header. Remove the panel header by moving the **af:query** component on top the **af:panelGroupLayout** component, selecting the panel header and pressing delete.



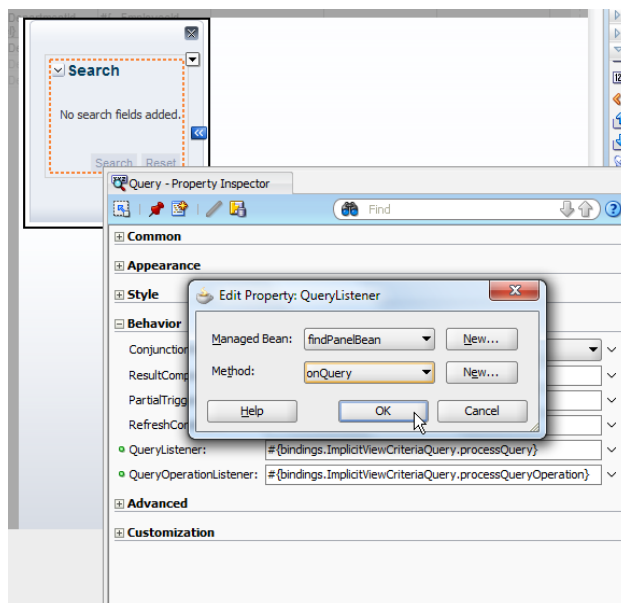
Select the **af:query** component and open the Property Inspector (ctrl+shift+I). Set the **SaveQueryMode** property to hidden and the **ModeChangeVisible** property to false. This simplifies the search dialog in

that the options for saving and selecting saved queries do not become available. Also, the **Basic** mode button is hidden.

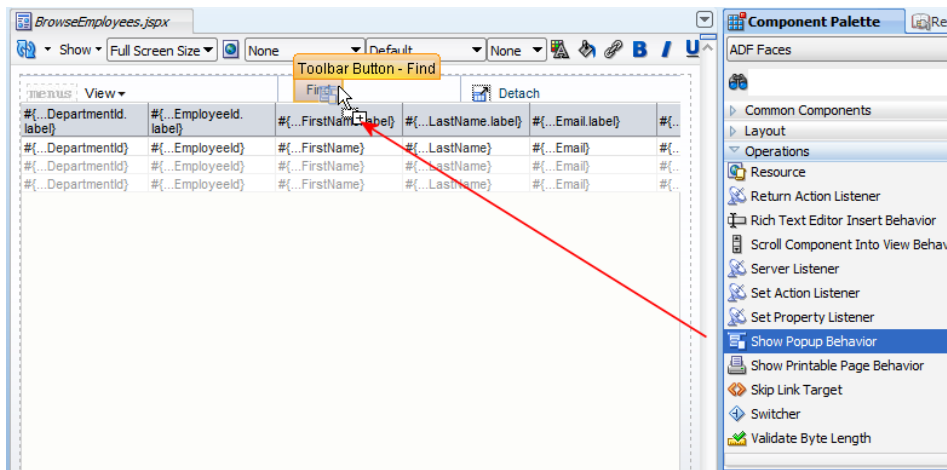
When the user hits the **Search** button to query the collection, we want to close the **af:popup** dialog and partially refresh the **af:table** component. The hook point for doing this is the **QueryListener** property of the **af:query** component. The **QueryListener** property points to the ADF binding layer by default. Copy the EL value of the property into the clipboard and press the **arrow** icon to the right of the property field.



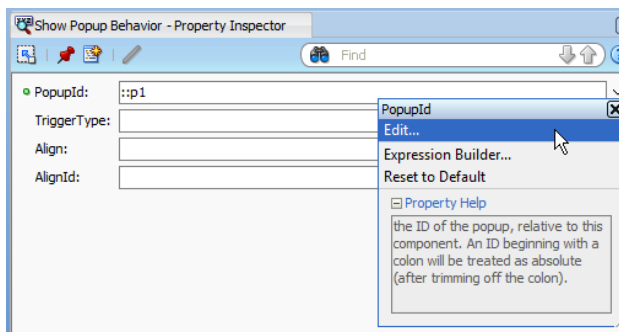
In the opened context menu, select the **Edit** option to create or select a managed bean. Define a method name for the **QuickListener** handler you create in the managed bean.



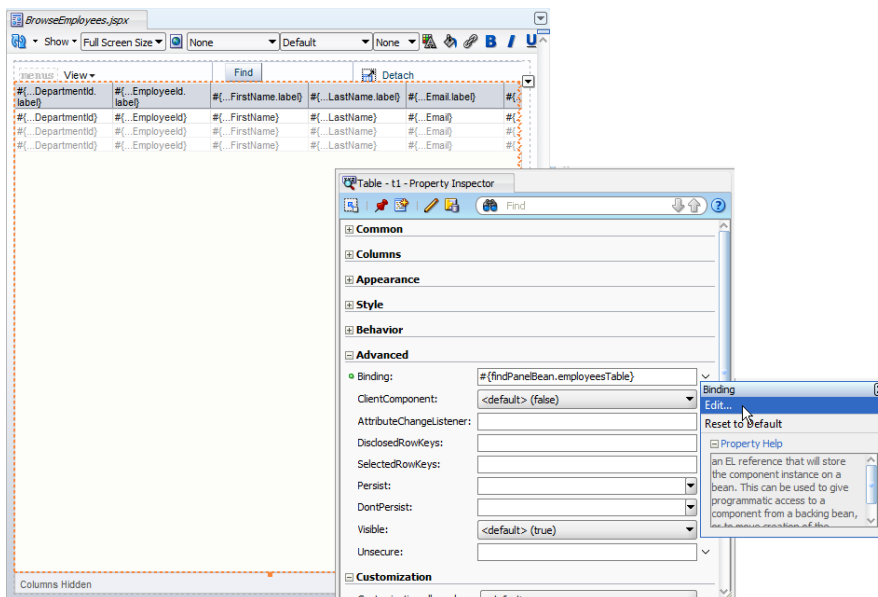
To launch the **af:popup** dialog containing the **af:query** component, drag and drop the **Show popup Behavior** entry from the component palette onto the tool bar button in the **af:panelCollection**.



Select the **af:showPopupBehavior** component in the Structure window and open the Property Inspector. In the Property Inspector, press the arrow icon next to the **PopupId** field. Press the **Edit** option in the opened context menu to search for the **af:popup** component on the page. This adds the popup id as a value to the **PopupId** field. Leave the other properties empty so the popup opens in the center of the page.



Select the **af:table** component and navigate to the **Binding** property using the Property Inspector. Press the arrow icon on the right and select **Edit** from the context menu.



Select the managed bean you created for the query listener. This creates a component binding to the managed bean, which is used to refresh the table after the user query.

Select the **af:popup** component and navigate to the **Binding** property using the Property Inspector. Press the arrow icon on the right and select **Edit** from the context menu. Select the managed bean you created for the query listener. This creates a component binding to the managed bean, which is used to close the dialog at the end of the query.

## The managed bean code

Of course, the magic is in the managed bean that executes the QueryListener. But though this is the trick that makes it all working, the implementing code is quite simple

```
import javax.el.ELContext;
import javax.el.ExpressionFactory;
import javax.el.MethodExpression;
import javax.faces.application.Application;
import javax.faces.context.FacesContext;

import oracle.adf.view.rich.component.rich.RichPopup;
import oracle.adf.view.rich.component.rich.data.RichTable;
import oracle.adf.view.rich.context.AdfFacesContext;
import oracle.adf.view.rich.event.QueryEvent;

public class FindPanelBean {
    private RichTable employeesTable;
    private RichPopup findPopup;

    public FindPanelBean() {
    }

    /*
     * method called from the af:query component's query listener
     */

    public void onQuery(QueryEvent queryEvent) {
        //preserve default query behavior, accessing the ADF binding layer
        String mexpr = "#{bindings.ImplicitViewCriteriaQuery.processQuery}";
        processMethodExpression(mexpr, queryEvent, QueryEvent.class);

        //close dialog
        findPopup.hide();
        AdfFacesContext adfFacesContext =
            AdfFacesContext.getCurrentInstance();
        adfFacesContext.addPartialTarget(employeesTable);
    }
}
```

```
/*
 * simplified method for invoking an EL for a single argument and
 * argument class
 */
public Object processMethodExpression(String methodExpression,
                                     Object event,
                                     Class eventClass)
{
    return processMethodExpression(methodExpression,
                                   new Object[] {event},
                                   new Class[] { eventClass });
}

/*
 * method that executes a method expression
 */
private Object processMethodExpression(String methodExpression,
                                     Object[] parameters,
                                     Class[] expectedParamTypes) {
    FacesContext fctx = FacesContext.getCurrentInstance();
    ELContext elctx = fctx.getELContext();
    Application app = fctx.getApplication();
    ExpressionFactory exprFactory = app.getExpressionFactory();
    MethodExpression methodExpr =
        exprFactory.createMethodExpression(elctx, methodExpression,
                                         Object.class,
                                         expectedParamTypes);
    return methodExpr.invoke(elctx, parameters);
}

/*
 * *****
 * JSF Component Bindings
 * *****
 */
public void setEmployeesTable(RichTable employeesTable) {
    this.employeesTable = employeesTable;
}

public RichTable getEmployeesTable() {
    return employeesTable;
}

public void setFindPopup(RichPopup findPopup) {
    this.findPopup = findPopup;
}
}
```



```

public RichPopup getFindPopup() {
    return findPopup;
}
}

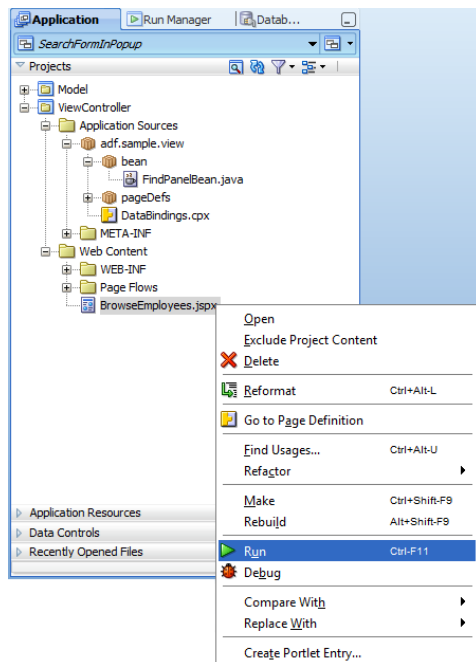
```

## Download the Sample

The Sample can be downloaded from sample 67 at ADF Code Corner:

<http://www.oracle.com/technetwork/developer-tools/adf/learnmore/index-101235.html>

To run the sample, configure the application database connect information to point to a database with the HR schema installed, select the **BrowseEmployees.jspx** page and choose the **Run** option.



In the page, press the **Find** button to launch the query dialog. Enter a search filter condition and press the **Search** button. The table is re-queried and the popup closed after this.

DepartmentId	EmployeeId	FirstName	LastName	Email	HireDate	Salary	PhoneNumber	JobId	Com
90	100	Steven	King	SKING	6/17/2003	24000	515.123.4567	AD_PRES	
90	101	Neena	Kochhar	NKOCHHAR	9/21/2005	17000	515.123.4568	AD_VP	
90	102	Lex	De Haan	LDEHAAN	1/13/2001	17000	515.123.4569	AD_VP	
60	103	Alexander	Hunold	AHLNOLD	1/3/2006	10000	590.423.4567	IT_PROG	
60	104	Bruce	Ernst	BERNST	5/21/2007	6000	590.423.4568	IT_PROG	
60	105	David	Austin	DAUSTIN	6/25/2005	4800	590.423.4569	IT_PROG	
60	106	Valli	Pataballa	VPATABAL	2/5/2006	4800	590.423.4567	IT_PROG	
60	107	Diana	Lorentz	DLORENTZ	2/7/06	4800	590.423.4568	IT_PROG	
100	108	Nancy	Greenberg	NGREENBE	8/17/06	4800	590.423.4567	IT_PROG	
100	109	Daniel	Faviet	DFAVIET	8/16/06	4800	590.423.4568	IT_PROG	
100	110	John	Chen	JCHEN	9/28/06	4800	590.423.4569	IT_PROG	
100	111	Ismael	Scarra	ISCIARRA	9/30/06	4800	590.423.4567	IT_PROG	
100	112	Jose Manuel	Urman	JMURMAN	3/7/07	4800	590.423.4568	IT_PROG	
100	113	Luis	Popp	LPOPP	12/7/07	4800	590.423.4569	IT_PROG	
30	114	Den	Raphaely	DRAPHEAL	12/7/07	4800	590.423.4567	IT_PROG	
30	115	Alexander	Khoo	AKHOO	5/18/08	4800	590.423.4568	IT_PROG	
30	116	Shelli	Baida	SBAIDA	12/2/08	4800	590.423.4569	IT_PROG	
30	117	Sigal	Tobias	STOBIAS	7/24/08	4800	590.423.4567	IT_PROG	
30	118	Guy	Hirano	GHIRANO	11/15/08	4800	590.423.4568	IT_PROG	
30	119	Karen	Colmenares	KCOLMENA	8/10/09	4800	590.423.4569	IT_PROG	
50	120	Matthew	Weiss	MWEISS	7/18/09	4800	590.423.4567	IT_PROG	
50	121	Adam	Fripp	AFRIPP	4/10/09	4800	590.423.4568	IT_PROG	
50	122	Payam	Kaufling	PKAULFIN	5/1/09	4800	590.423.4569	IT_PROG	
50	123	Shanta	Vollman	SVOLLMAN	10/1/09	4800	590.423.4567	IT_PROG	
50	124	Kevin	Murugesu	KMURUGES	11/16/09	4800	590.423.4568	IT_PROG	
50	125	Julia	Nayer	JNAYER	7/16/09	4800	590.423.4569	IT_PROG	
50	126	Irene	Mikkilineni	IMIKKILI	9/28/09	4800	590.423.4567	IT_PROG	
50	127	James	Landry	JLANDRY	1/14/10	4800	590.423.4568	IT_PROG	
50	128	Steven	Markle	SMARKLE	3/8/10	4800	590.423.4569	IT_PROG	
50	129	Laura	Bissot	LBISSOT	8/20/10	4800	590.423.4567	IT_PROG	
50	130	Mozhe	Atkinson	MATKINSO	10/28/10	4800	590.423.4568	IT_PROG	
50	131	James	Marlow	JMARLOW	2/16/11	4800	590.423.4569	IT_PROG	
50	132	TJ	Olson	TJOLSON	4/10/2007	2100	650.124.8234	ST_CLERK	

---

RELATED DOCUMENTATION

---

<input type="checkbox"/>	
<input type="checkbox"/>	
<input type="checkbox"/>	