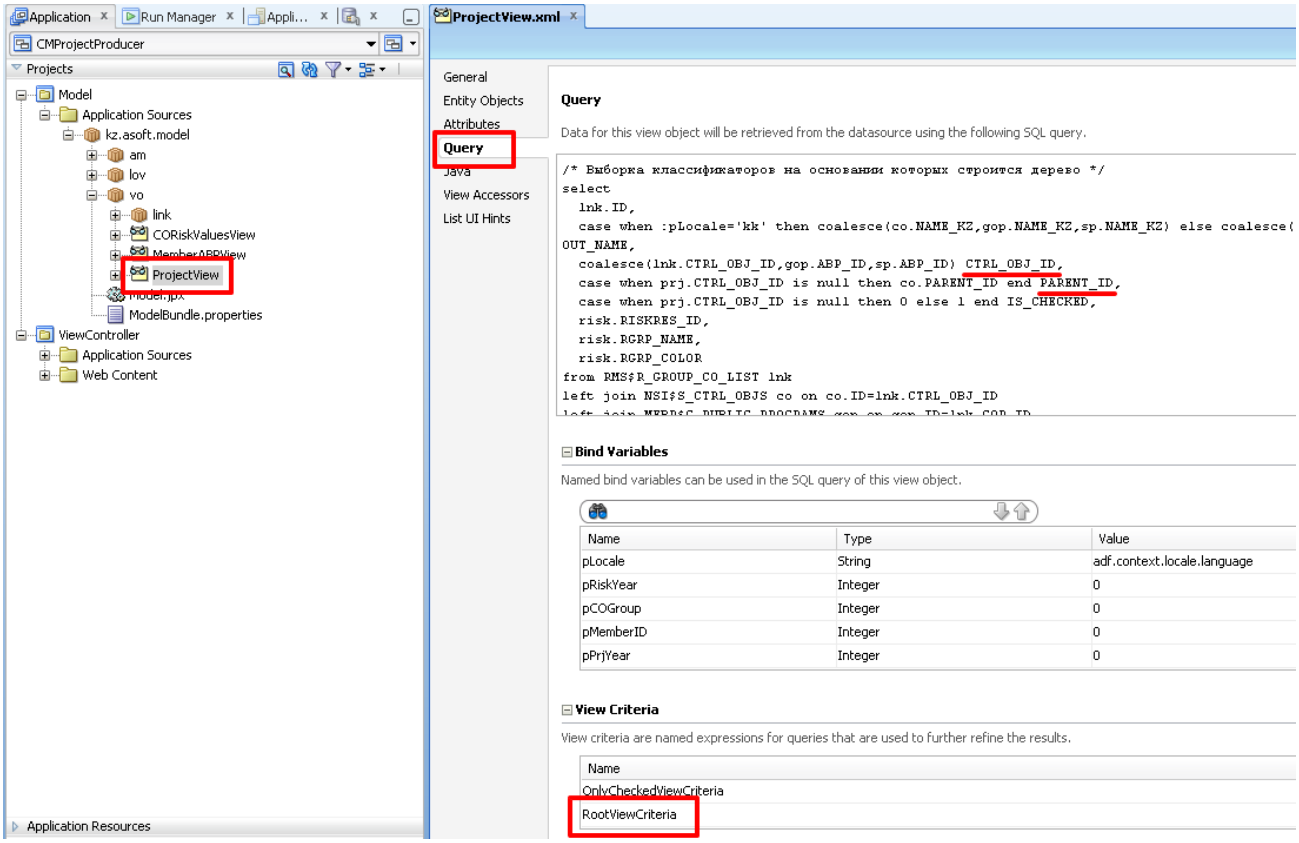
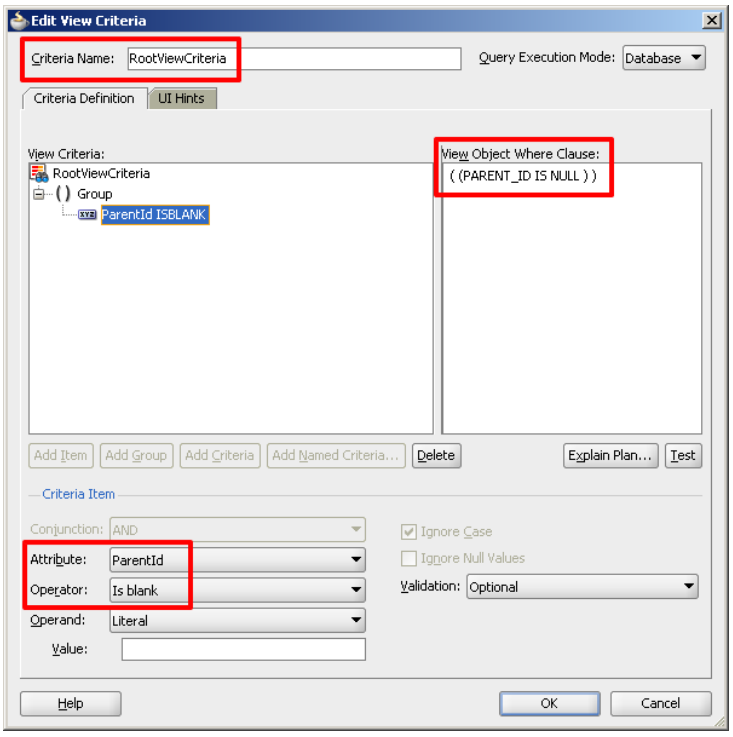
Иерархическое дерево в ADF-приложении, его параметры и критерии

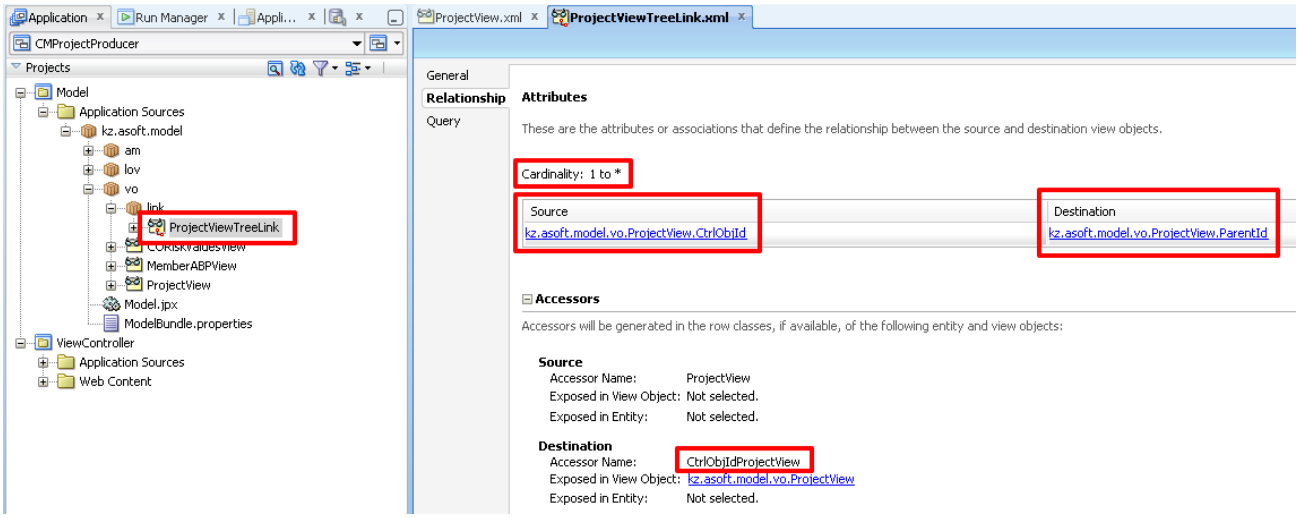
Строим **ViewObject** содержащий классическую связку по полям **CTRL_OBJ_ID** и **PARENT_ID** (здесь, если **PARENT_ID=NULL** то это корневая запись):



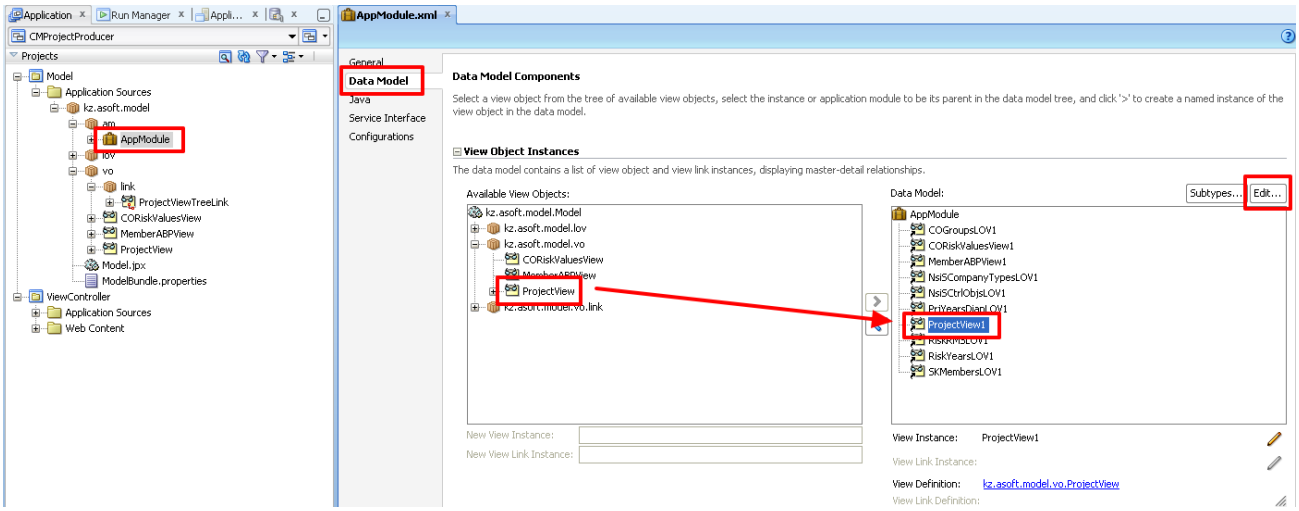
Добавляем критерий «**RootViewCriteria**» отвечающий за выборку корневых узлов:



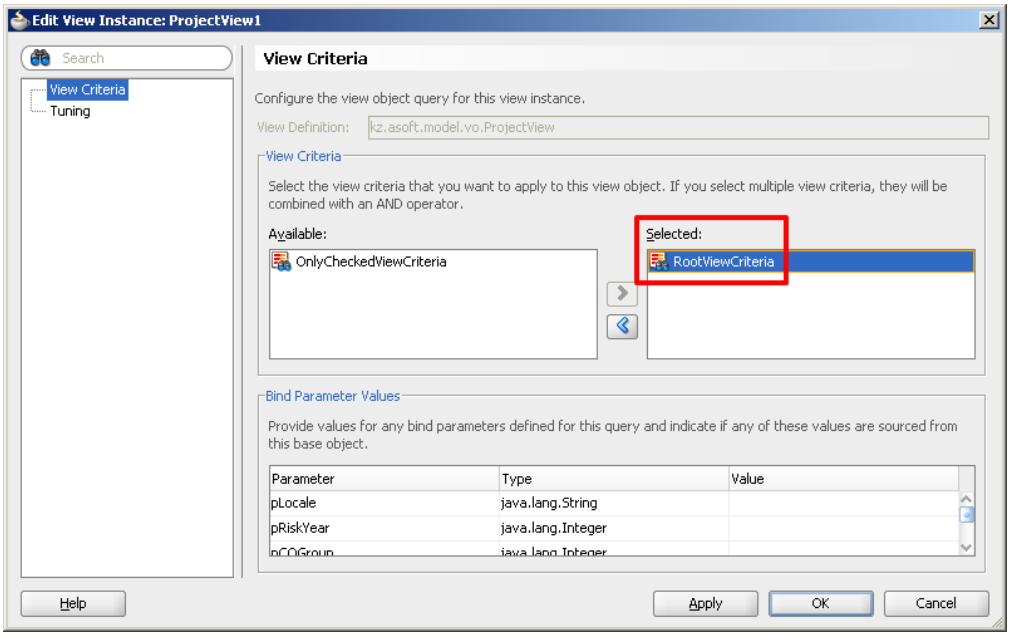
Строим **Link**, который будет определять связь по полям **CTRL_OBJ_ID** и **PARENT_ID**:



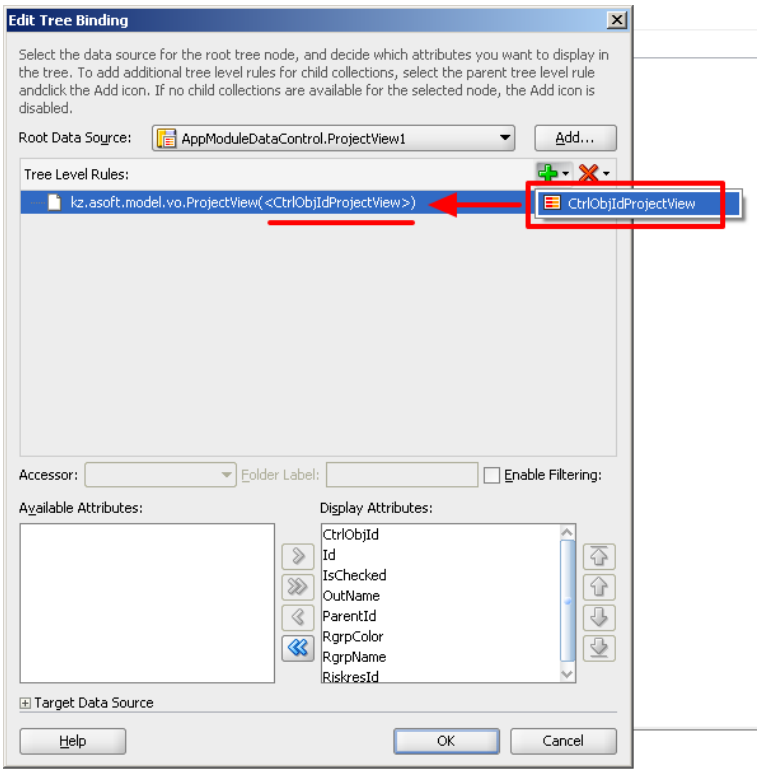
Добавляем **ViewObject** в **AppModule**:



Задаем у данного **ViewObject** критерий, отвечающий за выборку корневых узлов:



Строим дерево и указываем, что связь будет осуществляться при помощи аксесора **CtrlObjIdProjectView** (данное наименование задается при создании **Link'a**):

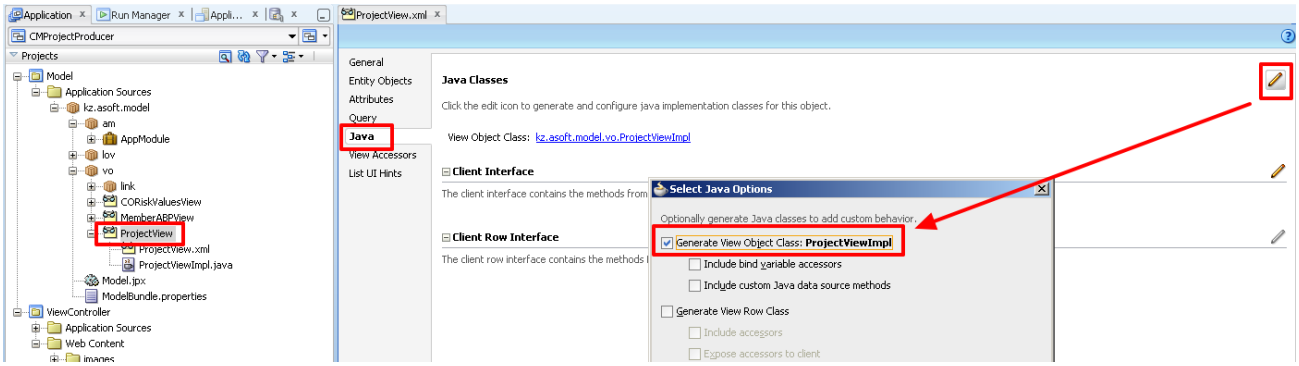


Дерево будет строится по следующей логике:

1) На основании **ViewObject (ProjectView1)** определенного в **AppModule** делается выборка корневых узлов (и соответственно, здесь будет задействован критерий **RootViewCriteria**).
2) Для выборки дочерних узлов формируются новые **ViewObject** вида **ProjectView**, у которых **все параметры будут установлены значениями по умолчанию (т.е. pRiskYear=0; pCOGroup=0; …) и все критерии будут отключены**! По этой причине, если мы меняем параметры и критерии **ProjectView1**, это никак не будет влиять на дочерние узлы! О том, как задать параметры и критерии для дочерних узлов описано ниже.

Как задать параметры и критерии для дочерних узлов

Создаем Java-класс для **ProjectView**:



В данном классе необходимо переопределить метод **createViewLinkAccessorRS**, который вызывается для формировании **ViewObject** для каждого дочернего узла. В данном методе мы и можем повлиять на параметры и критерии дочерних **ViewObject**.

Код класса **ProjectViewImpl**:

```
package kz.asoft.model.vo;

import oracle.jbo.Row;
import oracle.jbo.VariableValueManager;
import oracle.jbo.ViewCriteria;
import oracle.jbo.ViewCriteriaManager;
import oracle.jbo.server.AssociationDefImpl;
import oracle.jbo.server.ViewAccessorDef;
import oracle.jbo.server.ViewObjectImpl;
import oracle.jbo.server.ViewRowSetImpl;

public class ProjectViewImpl extends ViewObjectImpl {

    Object memberID;
    Object riskYear;
    Object grpCOID;
    Object reqYear;
    Boolean onlySelected;

    /**
     * установка параметров и критериев ViewObject
     */
    public void setProjectParams(Object memberID, Object riskYear, Object grpCOID, Object reqYear, Boolean
onlySelected) {
        this.memberID = memberID;
        this.riskYear = riskYear;
        this.grpCOID = grpCOID;
        this.reqYear = reqYear;
        this.onlySelected = onlySelected;

        // обновляем параметры текущего ViewObject
        VariableValueManager vvm = ensureVariableManager();
        vvm.setVariableValue("pRiskYear", riskYear);
        vvm.setVariableValue("pCOGroup", grpCOID);
        vvm.setVariableValue("pMemberID", memberID);
        vvm.setVariableValue("pPrjYear", reqYear);

        // отключаем критерий
        removeApplyViewCriteriaName("OnlyCheckedViewCriteria");

        if (onlySelected) {
            // Получить менеджер критериев для текущего ViewObject
            ViewCriteriaManager vcm = getViewCriteriaManager();
            // Получить критерий поиска
            ViewCriteria vc = vcm.getViewCriteria("OnlyCheckedViewCriteria");
            // Применить критерий
            applyViewCriteria(vc, true);
        }
    }

    /**
     * данный переопределенный метод вызывается при формировании ViewObject для дочерних записей
     */
    @Override
    protected ViewRowSetImpl createViewLinkAccessorRS(AssociationDefImpl associationDefImpl, ViewObjectImpl
viewObjectImpl, Row row, Object[] object) {
        // задаем свойства дочернего ViewObject, чтобы задействовать их в createViewLinkAccessorRS
        // уже дочернего ViewObject по отношению к viewObjectImpl
((ProjectViewImpl)viewObjectImpl).setProjectParams(memberID, riskYear, grpCOID, reqYear, onlySelected);

        // обновляем параметры дочернего ViewObject
        VariableValueManager vvm = viewObjectImpl.ensureVariableManager();
        vvm.setVariableValue("pRiskYear", riskYear);
        vvm.setVariableValue("pCOGroup", grpCOID);
        vvm.setVariableValue("pMemberID", memberID);
        vvm.setVariableValue("pPrjYear", reqYear);
```

```
        // отключаем критерий дочернего ViewObject
        viewObjectImpl.removeApplyViewCriteriaName("OnlyCheckedViewCriteria");

        if (onlySelected) {
            // Получить менеджер критериев для дочернего ViewObject
            ViewCriteriaManager vcm = viewObjectImpl.getViewCriteriaManager();
            // Получить критерий поиска
            ViewCriteria vc = vcm.getViewCriteria("OnlyCheckedViewCriteria");
            // Применить критерий
            viewObjectImpl.applyViewCriteria(vc, true);
        }

        return super.createViewLinkAccessorRS(associationDefImpl, viewObjectImpl, row, object);
    }

    /**
     * This is the default constructor (do not remove).
     */
    public ProjectViewImpl() {
    }
}
```

Т.к. код по заданию параметров и определению критериев для корней и дочерних записей в данном случаем получается однообразным, то его можно вынести в отдельный метод. В итоге получим:

```
package kz.asoft.model.vo;

import oracle.jbo.Row;
import oracle.jbo.VariableValueManager;
import oracle.jbo.ViewCriteria;
import oracle.jbo.ViewCriteriaManager;
import oracle.jbo.server.AssociationDefImpl;
import oracle.jbo.server.ViewAccessorDef;
import oracle.jbo.server.ViewObjectImpl;
import oracle.jbo.server.ViewRowSetImpl;

public class ProjectViewImpl extends ViewObjectImpl {

    Object memberID;
    Object riskYear;
    Object grpCOID;
    Object reqYear;
    Boolean onlySelected;

    /**
     * установка параметров и критериев ViewObject
     */
    public void setProjectParams(Object memberID, Object riskYear, Object grpCOID, Object reqYear, Boolean
onlySelected) {
        this.memberID = memberID;
        this.riskYear = riskYear;
        this.grpCOID = grpCOID;
        this.reqYear = reqYear;
        this.onlySelected = onlySelected;

        // задаем параметры и критерии текущего ViewObject
        setVOParams(this);
    }

    /**
     * общий метод для определения параметров и критериев ViewObject
     */
    private void setVOParams(ProjectViewImpl vo){
        // обновляем параметры ViewObject
        VariableValueManager vvm = vo.ensureVariableManager();
```

```
        vvm.setVariableValue("pRiskYear", riskYear);
        vvm.setVariableValue("pCOGroup", grpCOID);
        vvm.setVariableValue("pMemberID", memberID);
        vvm.setVariableValue("pPrjYear", reqYear);

        // отключаем критерий ViewObject
        vo.removeApplyViewCriteriaName("OnlyCheckedViewCriteria");

        if (onlySelected) {
            // Получить менеджер критериев для ViewObject
            ViewCriteriaManager vcm = vo.getViewCriteriaManager();
            // Получить критерий поиска
            ViewCriteria vc = vcm.getViewCriteria("OnlyCheckedViewCriteria");
            // Применить критерий
            vo.applyViewCriteria(vc, true);
        }
    }

    /**
     * данный переопределенный метод вызывается при формировании ViewObject для дочерних записей
     */
    @Override
    protected ViewRowSetImpl createViewLinkAccessorRS(AssociationDefImpl associationDefImpl, ViewObjectImpl
viewObjectImpl, Row row, Object[] object) {
        // задаем свойства дочернего ViewObject, чтобы задействовать их в createViewLinkAccessorRS
        // уже дочернего ViewObject по отношению к viewObjectImpl
        ((ProjectViewImpl)viewObjectImpl).setProjectParams(memberID, riskYear, grpCOID, reqYear,
onlySelected);

        // задаем параметры и критерии дочерних ViewObject
        setVOParams((ProjectViewImpl)viewObjectImpl);

        return super.createViewLinkAccessorRS(associationDefImpl, viewObjectImpl, row, object);
    }

    /**
     * This is the default constructor (do not remove).
     */
    public ProjectViewImpl() {
    }
}
```

Получили удобство в том, что если у ViewObject появятся или удалятся параметры, то их нужно будет менять только в одном месте.

Теперь мы можем изменять параметры дерева обращаясь через метод **setProjectParams**:

```
ProjectViewImpl projectVO;

// каким либо образом получаем ссылку на ViewObject, например, через итератор
projectIterator = Util.getDCIterator("ProjectView1Iterator");
projectVO = (ProjectViewImpl)projectIterator.getViewObject();

// переопределяем параметры дерева
projectVO.setProjectParams(memberID, riskYear, grpCOID, reqYear, false);
projectVO.executeQuery();
```