Warepro Workshop

Introduction to WarePro: Basic Functionality Training

Key Component Of WarePro

System Administration:

* Table & Window
* Menu
* Callout
* Process
* Report
* DocAction
* Dashboard Widget
* Workflow

**Creating a new window**

A window provides create, read, update, and delete (CRUD) access to the data to a user.

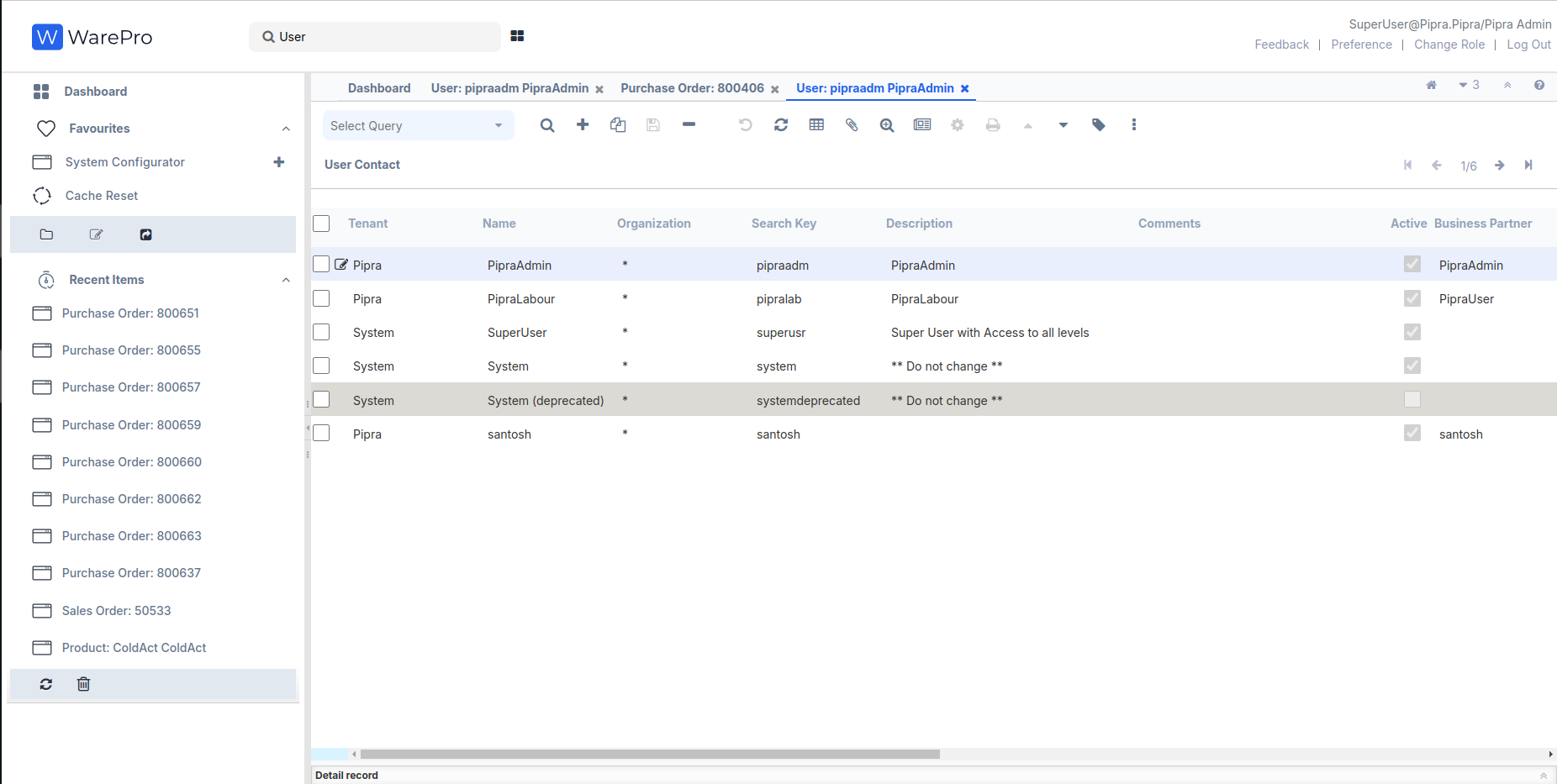
These functionalities are provided by the standard tools and menus. A standard layout of a

window has the following parts:

* **Search bar**
* **Tab bar**
* **Tool bar**
* **Tabs panel**
* **Status bar**
* **Menu**

The following screenshot shows the different parts:

**Search bar Tab bar Tool Bar**



**Status Bar Tabs Panel**

Given the iDempiere architecture, as an author of a new window, you do not have to worry about how and what gets displayed in the **Search Bar**, **Menu bar**, **Tool bar**, and **Status bar**. All we need to focus on is the **Tabs panel**. And, in this recipe, we will go through the steps required to create a complete working new window in iDempiere, which will act as the foundation for building our MOM window.

## 1. **Create the following table in your Adempiere schema:**

Here's the corrected version of our instructions

1. **Check running Docker containers:**docker ps
2. **Login to the PostgreSQL container:**  
   docker exec -it postgres bash
3. **Access the PostgreSQL database:**  
   psql -U postgres
4. **Switch to the idempiere database:**  
   \c idempiere
5. **Run our SQL query:**

CREATE TABLE adempiere.c\_mom (

c\_mom\_id numeric(10,0) NOT NULL PRIMARY KEY,

ad\_client\_id numeric(10,0) NOT NULL,

ad\_org\_id numeric(10,0) NOT NULL,

isactive character(1) DEFAULT 'Y'::bpchar NOT NULL,

created timestamp without time zone DEFAULT now() NOT NULL,

createdby numeric(10,0) NOT NULL,

updated timestamp without time zone DEFAULT now() NOT NULL,

updatedby numeric(10,0) NOT NULL,

value character varying(30) NOT NULL,

name character varying(255) NOT NULL,

start\_date date NOT NULL,

start\_time timestamp without time zone NOT NULL,

end\_time timestamp without time zone NOT NULL,

chairperson character varying(80),

participants character varying(4000),

agenda character varying(4000),

discussion\_detail character varying(8000));

iDempiere requires the following standard columns to be present on the tables, which

iDempiere populates on its own:

ad\_client\_id: Client Identifier

ad\_org\_id: Organization Identifier

isactive: Flag to indicate whether the record is active

created: Time when the record was created

createdby: ID of the user who created the record

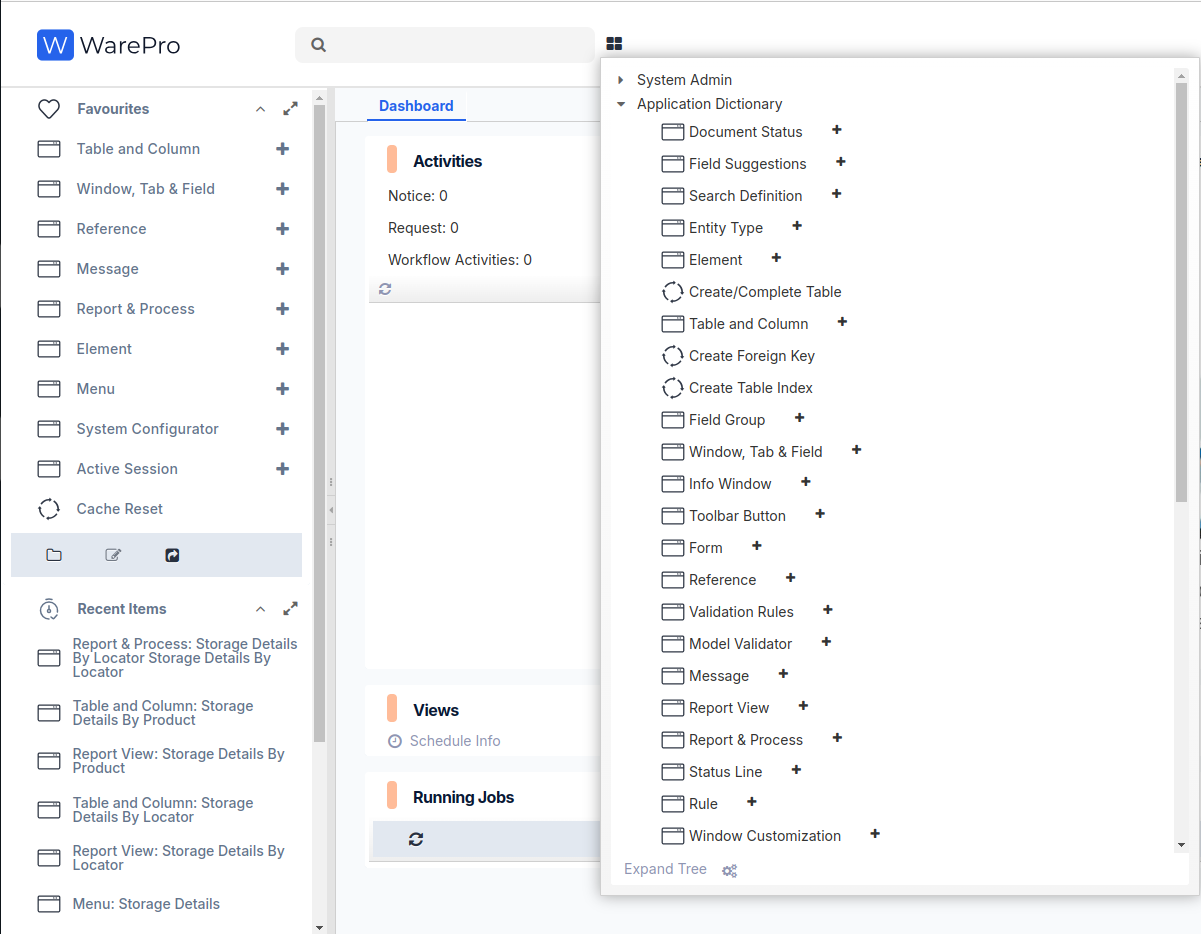
updated: Time when the record was last updated

updatedby: ID of the user who last updated the record

Additionally, every table must have a primary key, which must follow the naming convention

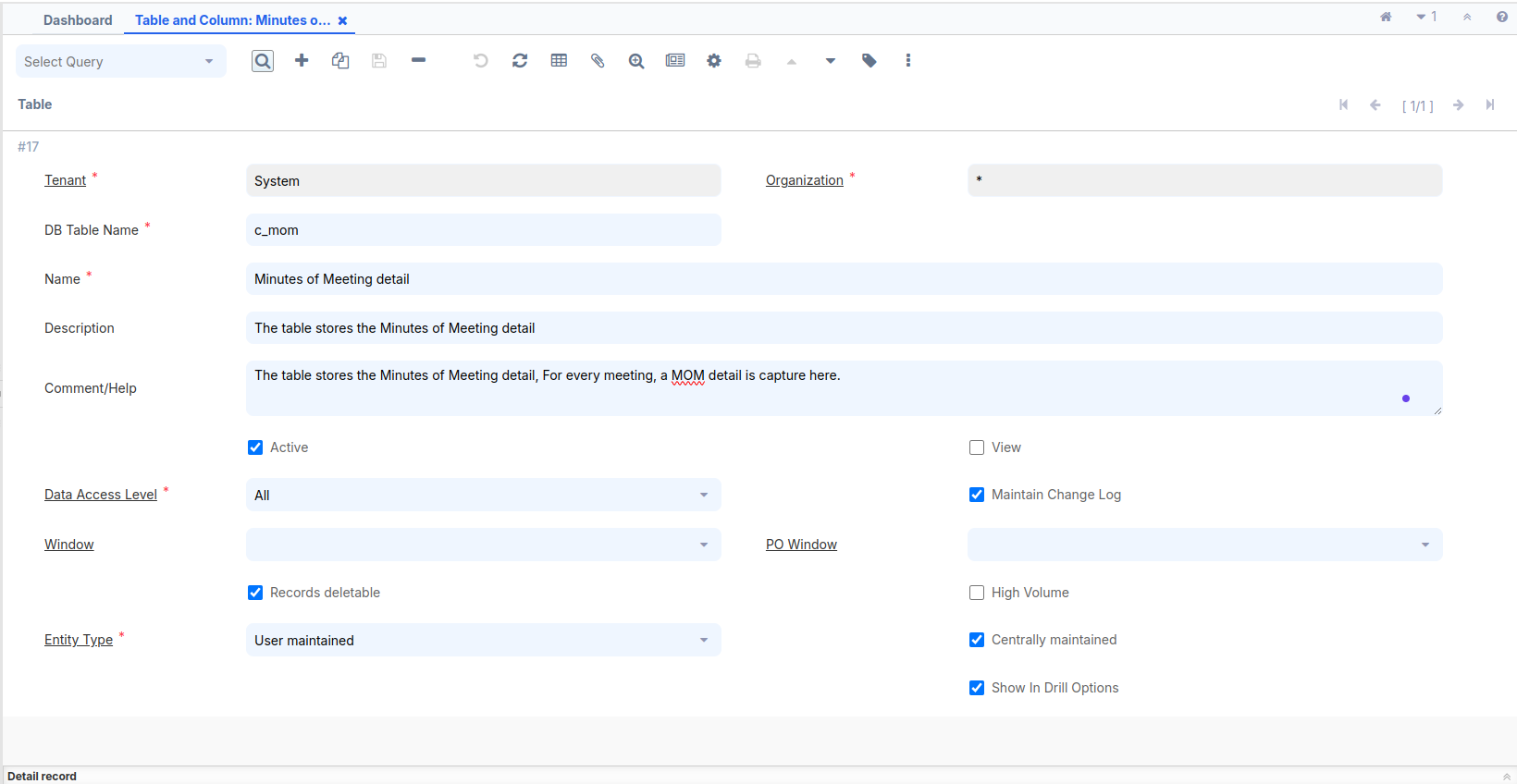
of <table name>\_id. c\_mom\_id which is the primary key of the c\_mom table.

**2.** Run the desktop version of ADempiere and log in using **System/System**. Select the **System Administrator** as **Role**. After a successful login, You will see the **Menu** on the left-hand side of the window, and you will see the **Application Dictionary** related items:  
Go to link and login   
URL: <https://localhost:8443/>

****

**3.** Click the **Table and Column**. It will pop up the **Lookup Record: Table** window where you click on the **New Record +** button on the bottom-left corner.

**4.** Enter the following details on to the **Table** tab as shown in the following screenshot:

  
DB Table Name = c\_mom  
Name = Minutes of Meeting Detail  
Description and Comment filed added your according or fill empty  
Data Access Level = All  
Entity Type = User maintained

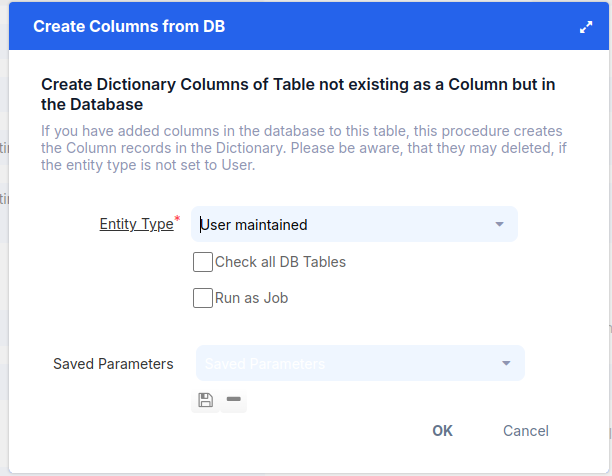
Check the Show in Drill Options Check Box

**5.** In the previous image, we used the default **Data Access Level—All**. This is only to make sure that we are able to access the window using any role so that we don't have to log out as the **System Administrator** and log in with another role to be able to access the window. But, in a real world scenario, you may have to use other options such as Client level, Client and Organization level, System level, and so on.

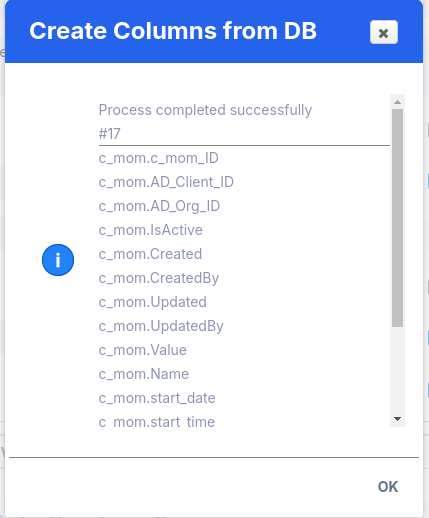
**6.** Press the **Setting** button on the middle toolbar Or press Alt + O.

select **Create Columns** **from** **DB** button. This will bring up the **Create Columns** from DB window where it prompts you to select the **Entity Type**.

**7.** Select **User maintained** as the **Entity Type**, as shown in the following screenshot:

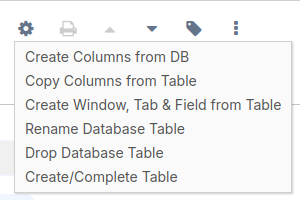


**8.** Click the **OK** button. It will pop-up a window showing all the columns generated from the table, as shown in the following screenshot:

****

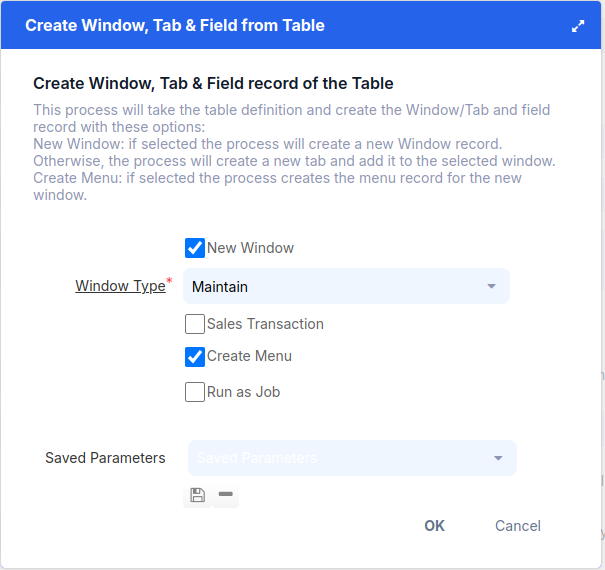
**9.** Click the ok button. This completes the creation of the columns, which you can verify by clicking on the **Column** tab.

Based on the data type of the columns in your database table, iDempiere guesses the data type of the columns, which you may want to use to store and retrieve data.For example, the **Date** on the column defaults to **Date+Time** in iDempiere, which may not be appropriate always as you may want to only store the date part of the timestamp. So, it is worth spending some time reviewing the Reference for the generated columns and making the required changes, if any.



**10.** After making any column corrections, or directly, return to the main window.Press the **Setting** button on the middle toolbar Or press Alt + O.

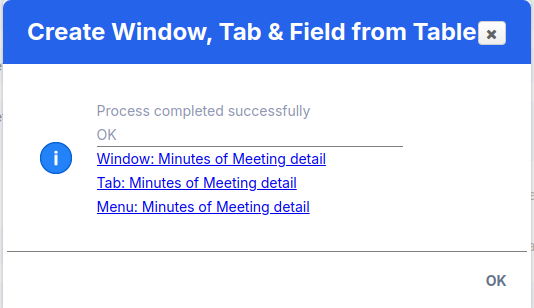
Select **Create Window,Tab & Field from Table**, which is the third option on the list

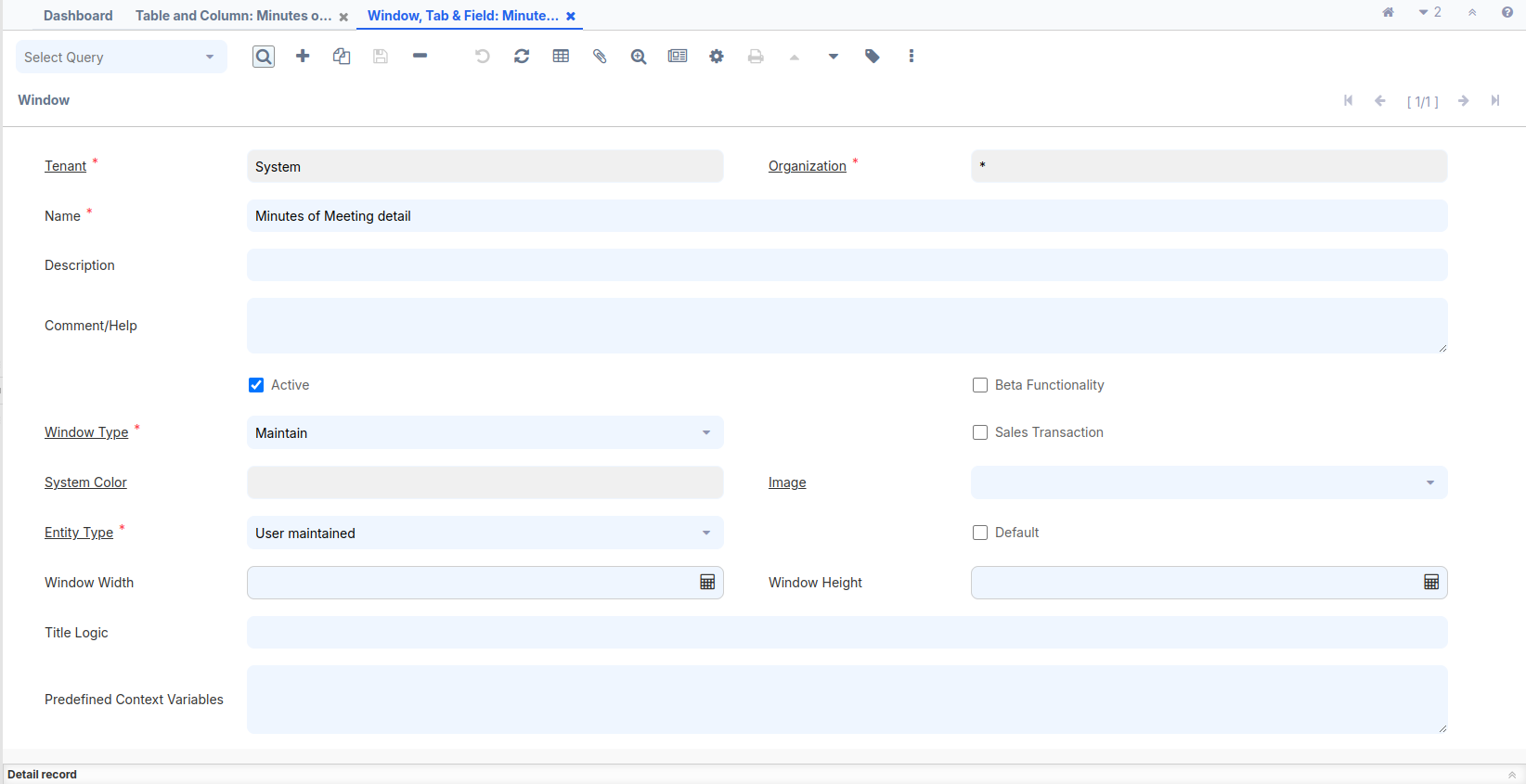
A new pop-up window will appear.

If we want to create a window, tab, and field, we can also create a menu.

Select the **Create Menu** checkbox and press **OK**.

Once the menu is created, it will be visible in the search bar.





Open the **Window, Tab,**

**and Field** window,

and review the screenshot.

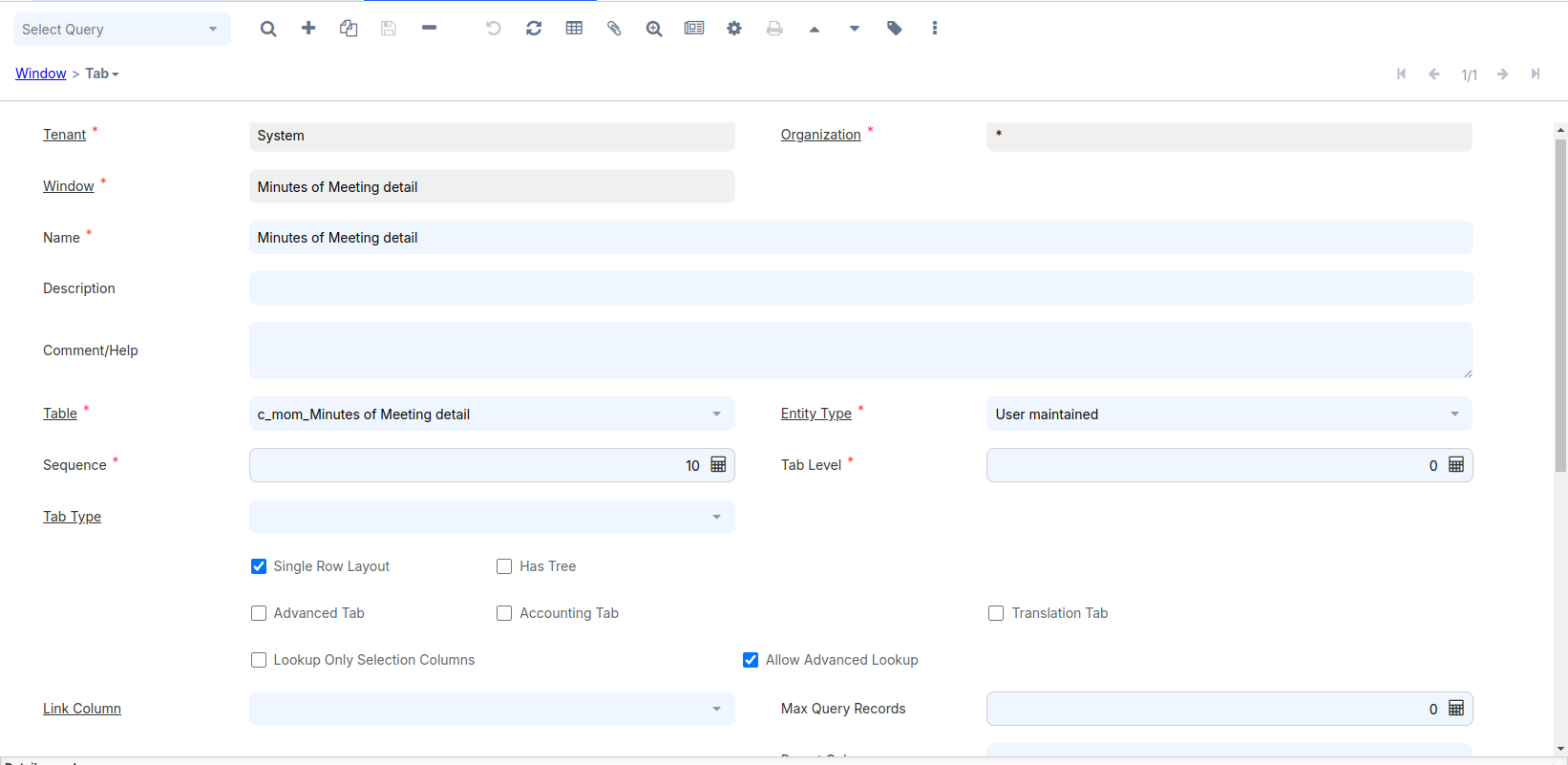
Make the necessary

changes according to

our requirements.

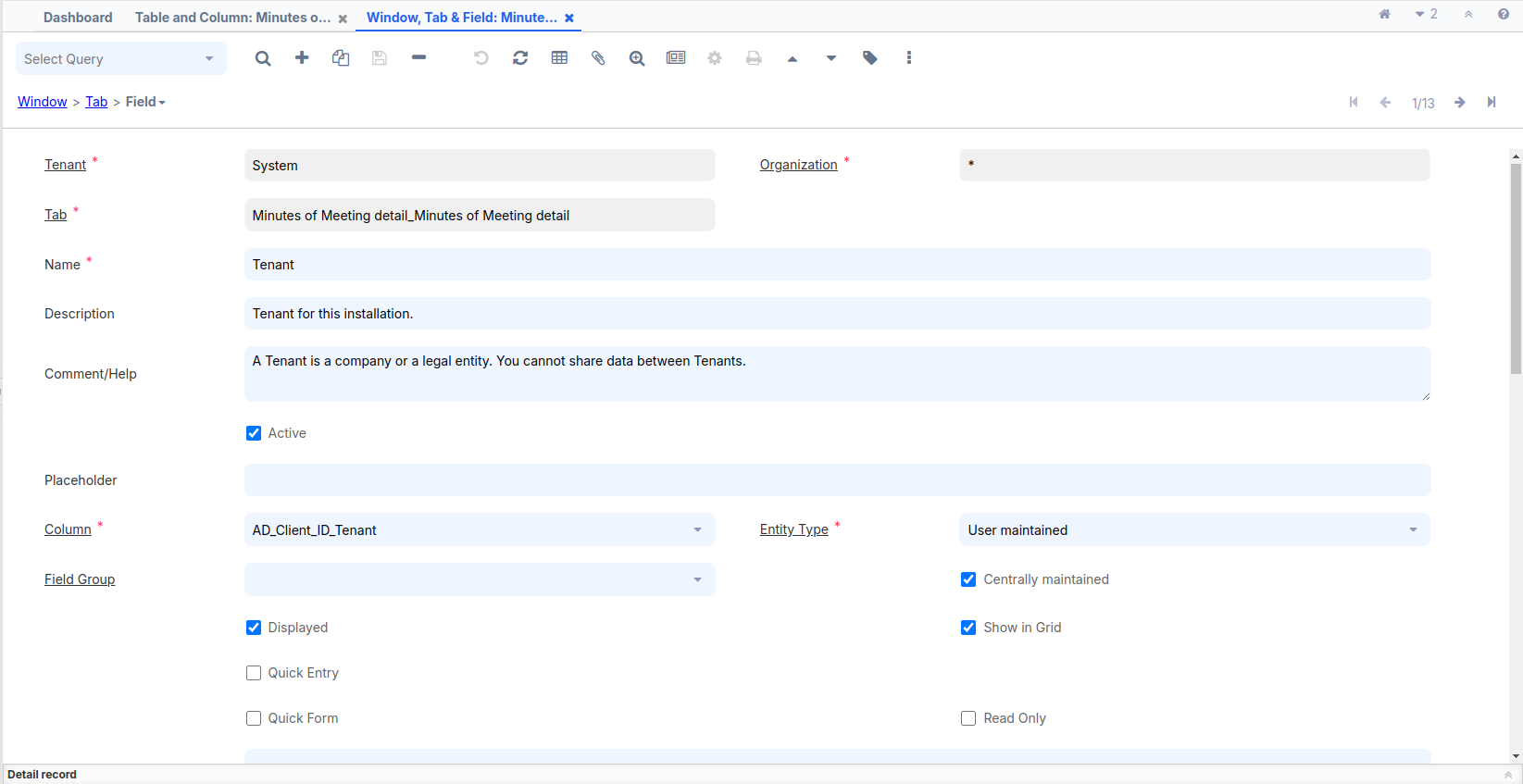
**Window:-**

* Name
* Window Type
* Sales Transaction

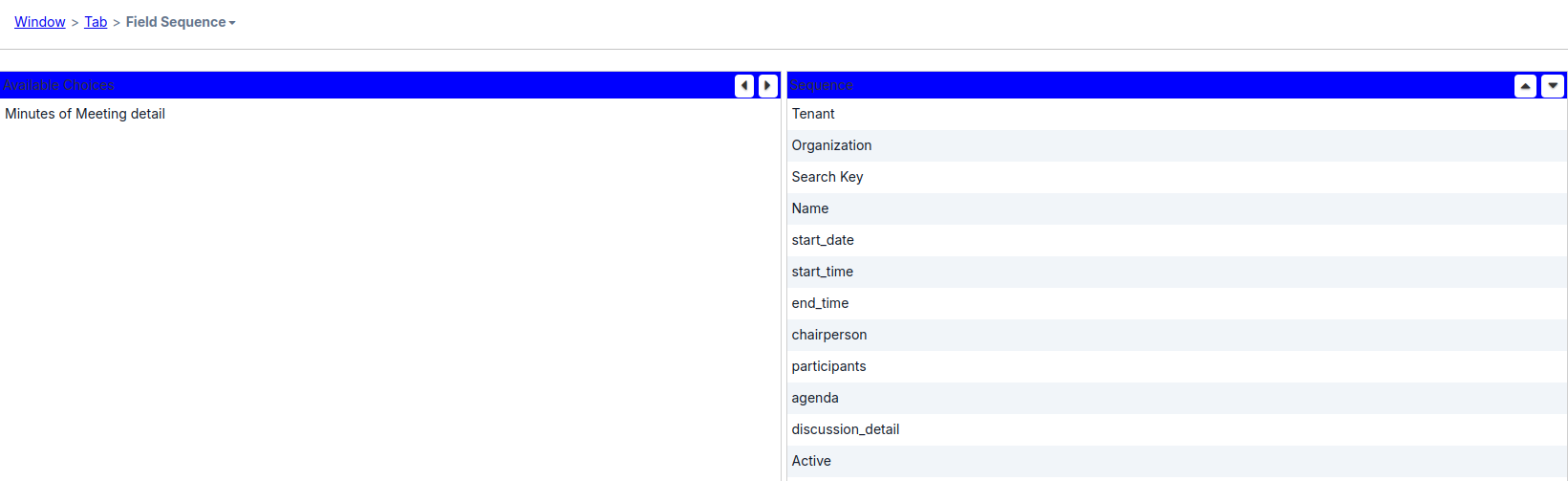


**Tab:-**

* Name
* Tab Level
* Single Row Layout
* Link Column
* Process
* Display Logic
* Read Only
* Sql Order By

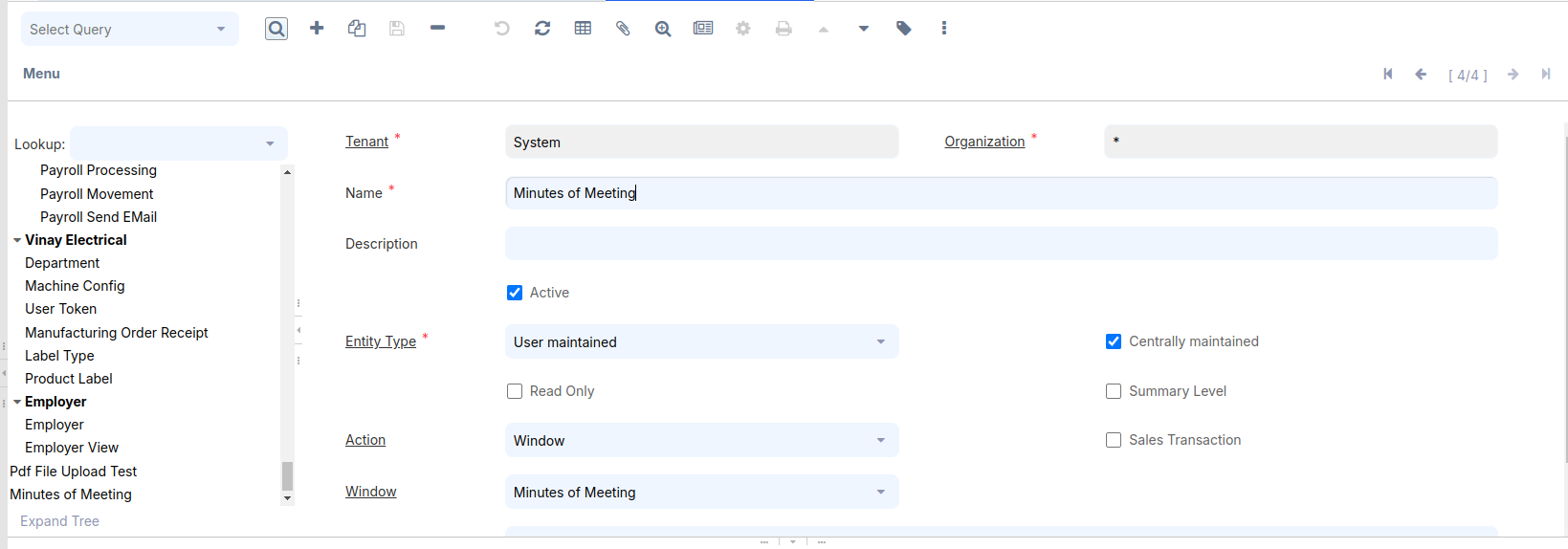
**Field:-**

* Name
* Description
* Field Group
* Displayed
* Read Only
* Show in Grid
* Field Only
* X Position
* Column Span



Go to the Field Sequence tab to see the default sequence in which the fields will appear on the tab. In case you want to change the sequence, you can do it in theSequence area. Select the field, whose sequence you want to change, and use the up and down arrow keys on the right to change the position of that field. Alternatively,you can also use the drag-n-drop for quicker re-ordering. The following screenshot

shows our fields after ordering:

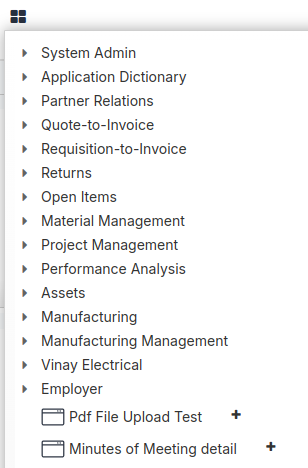


Enter **Menu** in

the search bar:

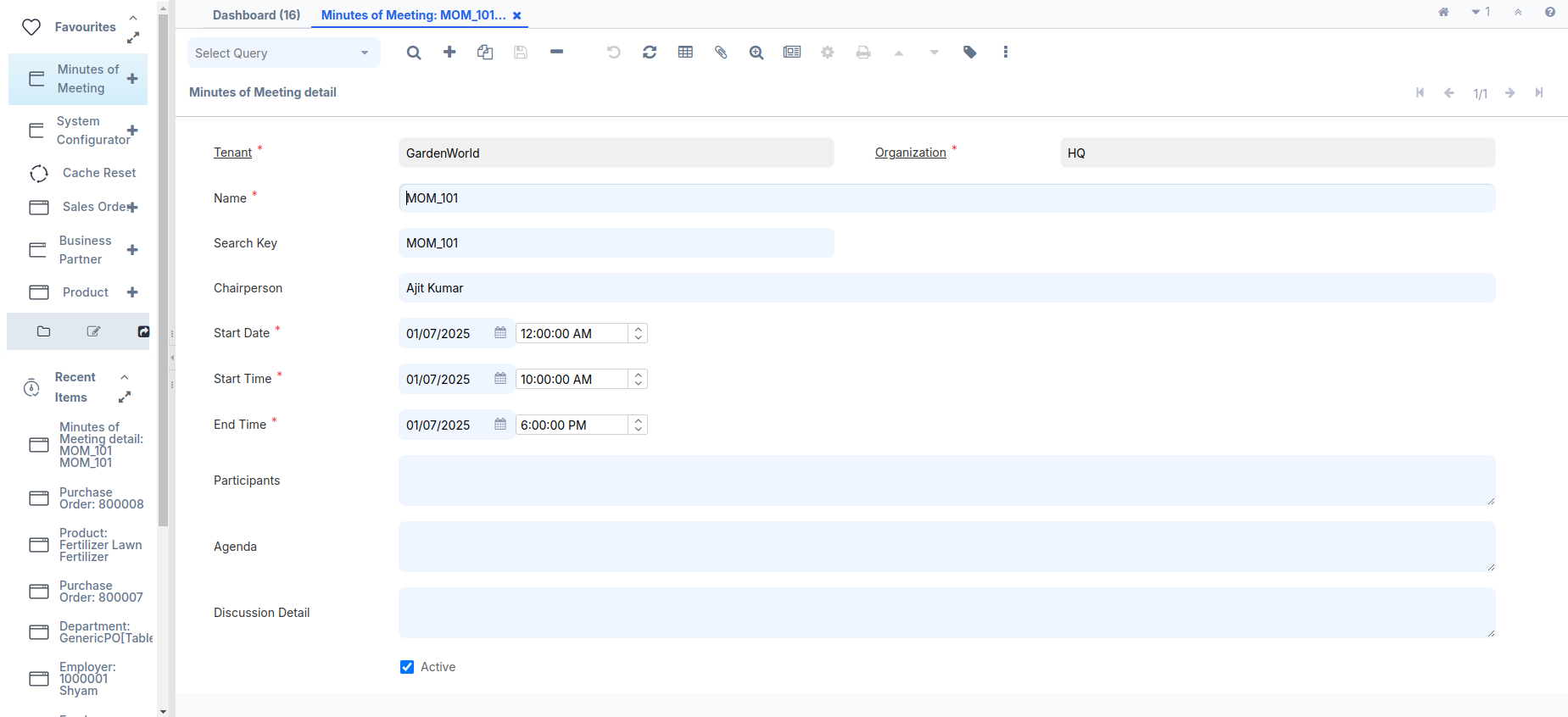
**Menu:-**

* Name
* Entity Type
* Action
* Window

Menu Tree

Log out and log in as **GardenAdmin/GardenAdmin**. You will see the following **Minutes Of Meeting menu** item:

Click on the **Minutes Of Meeting**  menu item. This will bring up the Minutes Of Meeting window with one tab, MOM. You will now be able to enter your MOM details and save it, as shown in the following screenshot:



At this stage, we have got our basic MOM window ready where we are able to save/update the information. As a standard practice in ADempiere, the Search Key shall appear before the Name field. We will do this in the later part of this chapter. However, you may work with the field sequence to accomplish this.

**Creating a window with multiple tabs**

It is pretty common to logically relate the information using tabs. This recipe describes the steps to add multiple tabs in a window. So far, we have our basic MOM window. Now, we will break it into multiple tabs:

1. The Participants detail is moved to a new tab—Participants
2. The Discussion detail is refined and we are going to capture the following information as part of every discussion item:  
   **#** Item number  
   **#** Discussion description  
   **#** Actioned by  
   **#** Status

**How to do it …**

Log in to Adempiere using the credentials System/System  
1.Delete some records from the c\_mom table.

2. Enter the Search bar, go to the **Window**, **Tab** and **Field** sections, enter the window name **Minutes of Meeting**, navigate to the **Field** section, and delete the specified fields:

1. Participants
2. Discussion Detail

3. Enter the Search bar, go to the **Table and Column** sections,enter the Table and column name **Minutes of Meeting**,navigate to the **column** section, and delete the specified column:

1. Participants
2. discussion\_detail

Please follow (**Create the following table in your Adempiere schema)** heading for create or alter table query,previously alreadyguide

4. Alter the adempiere.c\_mom table by running the following SQL:

ALTER TABLE adempiere.c\_mom  
DROP COLUMN participants,  
DROP COLUMN discussion\_detail;

5. Create two new tables by running the following SQL:

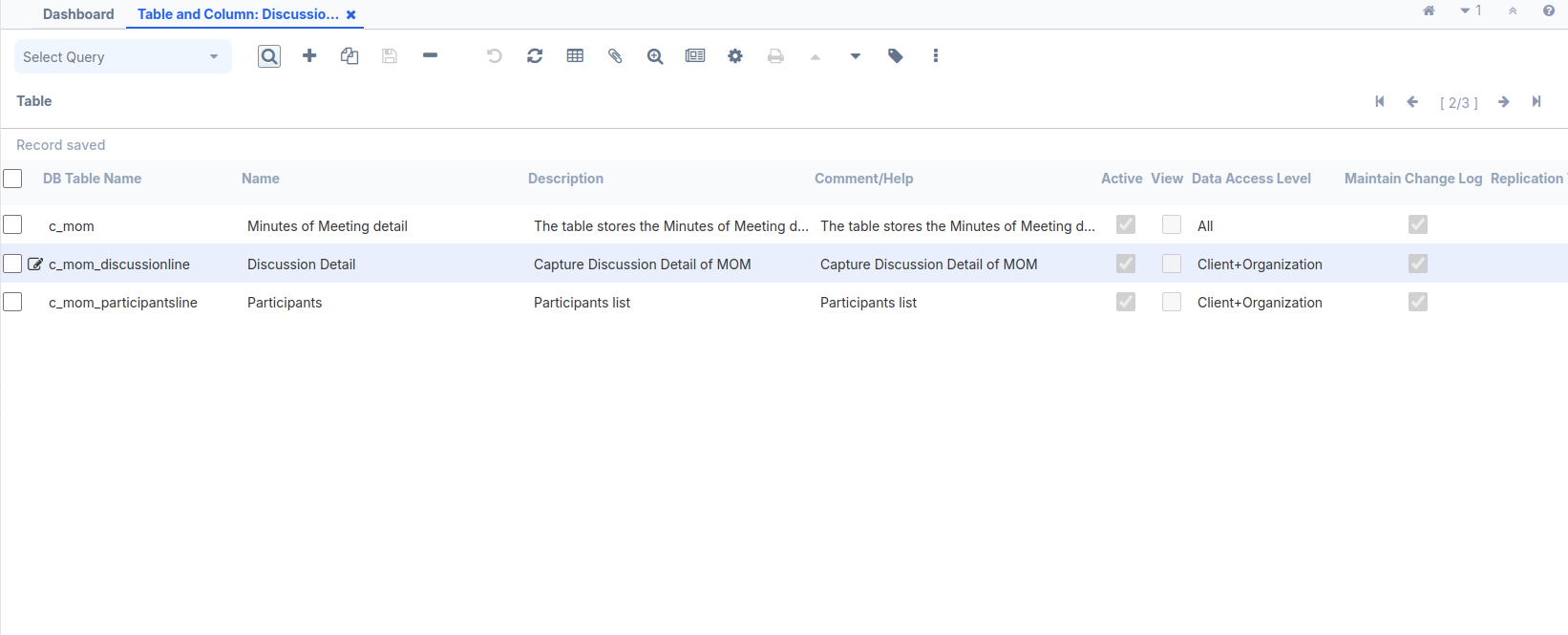
CREATE TABLE adempiere.c\_mom\_discussionline (  
c\_mom\_discussionline\_id numeric(10,0) NOT NULL PRIMARY KEY,  
c\_mom\_id numeric(10,0) NOT NULL,  
ad\_client\_id numeric(10,0) NOT NULL,  
ad\_org\_id numeric(10,0) NOT NULL,  
isactive character(1) DEFAULT 'Y'::bpchar NOT NULL,  
created timestamp without time zone DEFAULT now() NOT NULL,  
createdby numeric(10,0) NOT NULL,  
updated timestamp without time zone DEFAULT now() NOT NULL,  
updatedby numeric(10,0) NOT NULL,  
item\_nbr numeric (10,0) NOT NULL,  
discussion\_desc character varying(2000),  
actionedby character varying(80) NOT NULL,  
status character varying(80),  
FOREIGN KEY (c\_mom\_id) REFERENCES adempiere.c\_mom(c\_mom\_id));

CREATE TABLE adempiere.c\_mom\_participantsline (  
c\_mom\_participantsline\_id numeric(10,0) NOT NULL PRIMARY KEY,  
c\_mom\_id numeric(10,0) NOT NULL,  
ad\_client\_id numeric(10,0) NOT NULL,  
ad\_org\_id numeric(10,0) NOT NULL,  
isactive character(1) DEFAULT 'Y'::bpchar NOT NULL,  
created timestamp without time zone DEFAULT now() NOT NULL,  
createdby numeric(10,0) NOT NULL,  
updated timestamp without time zone DEFAULT now() NOT NULL,  
updatedby numeric(10,0) NOT NULL,  
participant character varying(80),  
company character varying(80) NOT NULL,  
FOREIGN KEY (c\_mom\_id) REFERENCES adempiere.c\_mom(c\_mom\_id));

c\_mom: Represents the basic MOM detail  
c\_mom\_discussionline: Represents each MOM discussion detail  
c\_mom\_participantsline: Represents each MOM participant detail

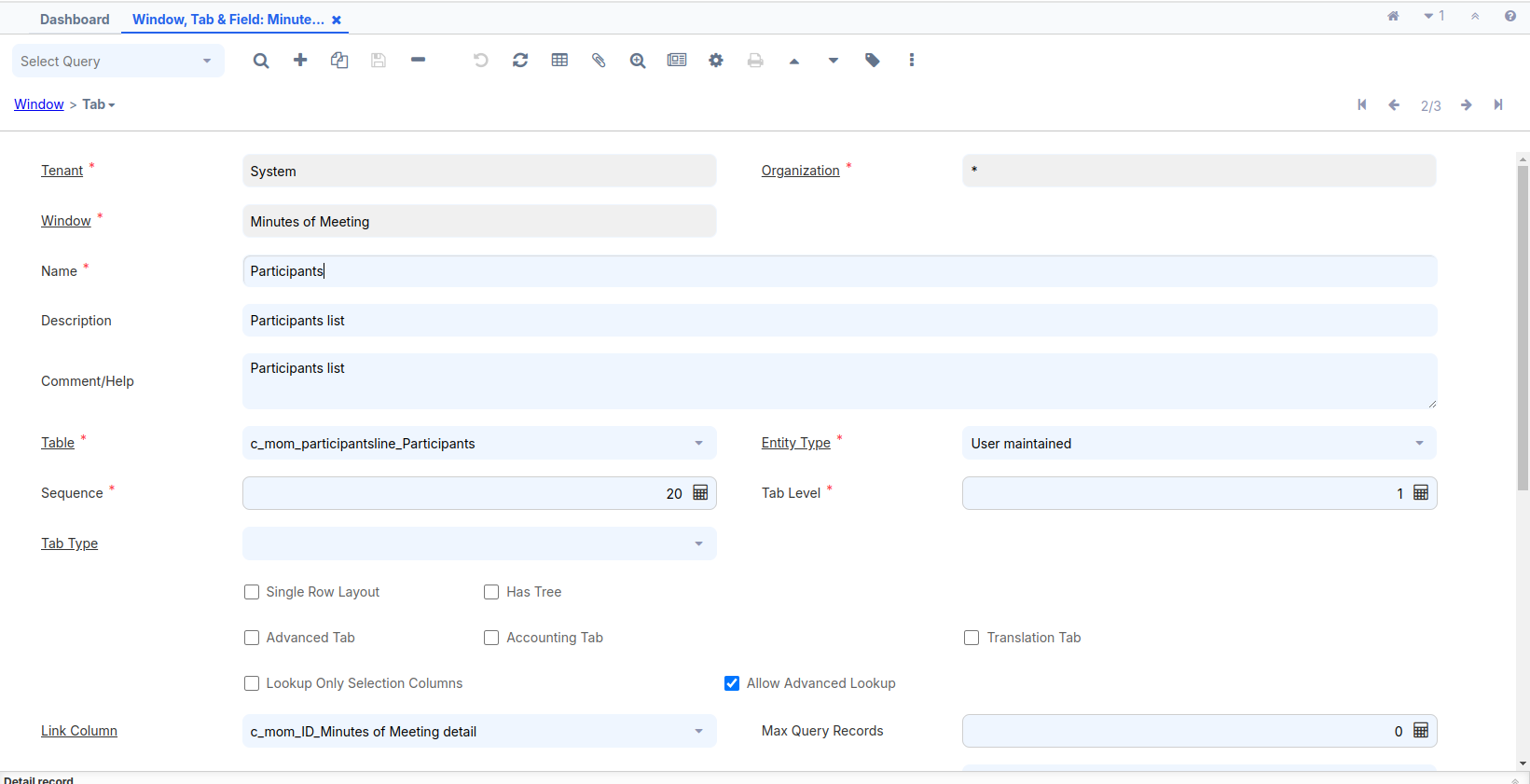
6. Follow the steps mentioned in the Create a new window recipe to create the tables and their columns. The following screenshot shows the list of tables after the completion of the steps:  
I have already mentioned how to create a table previously.  
Create C\_mom\_discussionline table:  
DB Table Name = C\_mom\_discussionline  
Name = Discussion Detail  
Description and Comment filed added your according or fill empty  
Data Access Level = Client+Organization  
Entity Type = User maintained  
Check the Show in Drill Options Check Box  
After saving the table press Alt + O and select **Create Column from DB** and all columns created.  
  
Same create c\_mom\_participantsline table:

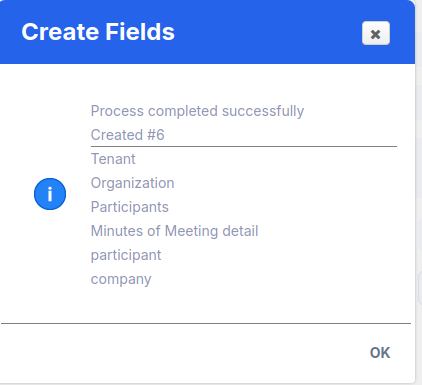
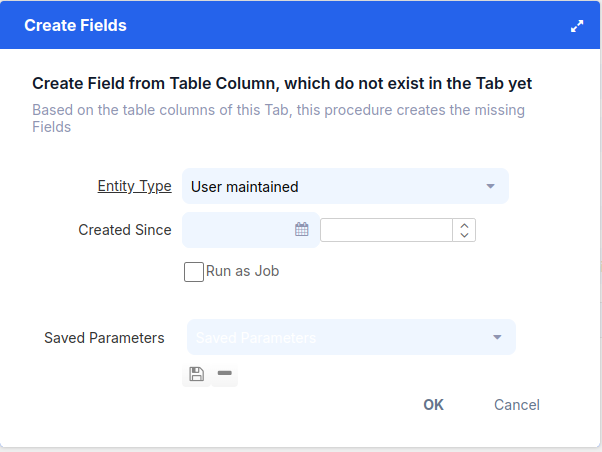
DB Table Name = c\_mom\_participantsline  
Name = Participants  
Description and Comment filed added your according or fill empty  
Data Access Level = Client+Organization  
Entity Type = User maintained  
Check the Show in Drill Options Check Box  
After saving the table press Alt + O and select **Create Column from DB** and all columns created.



7. Create the **Participants** tab inside **Minutes of Meeting** window and switch to the single row view. The important fields are:

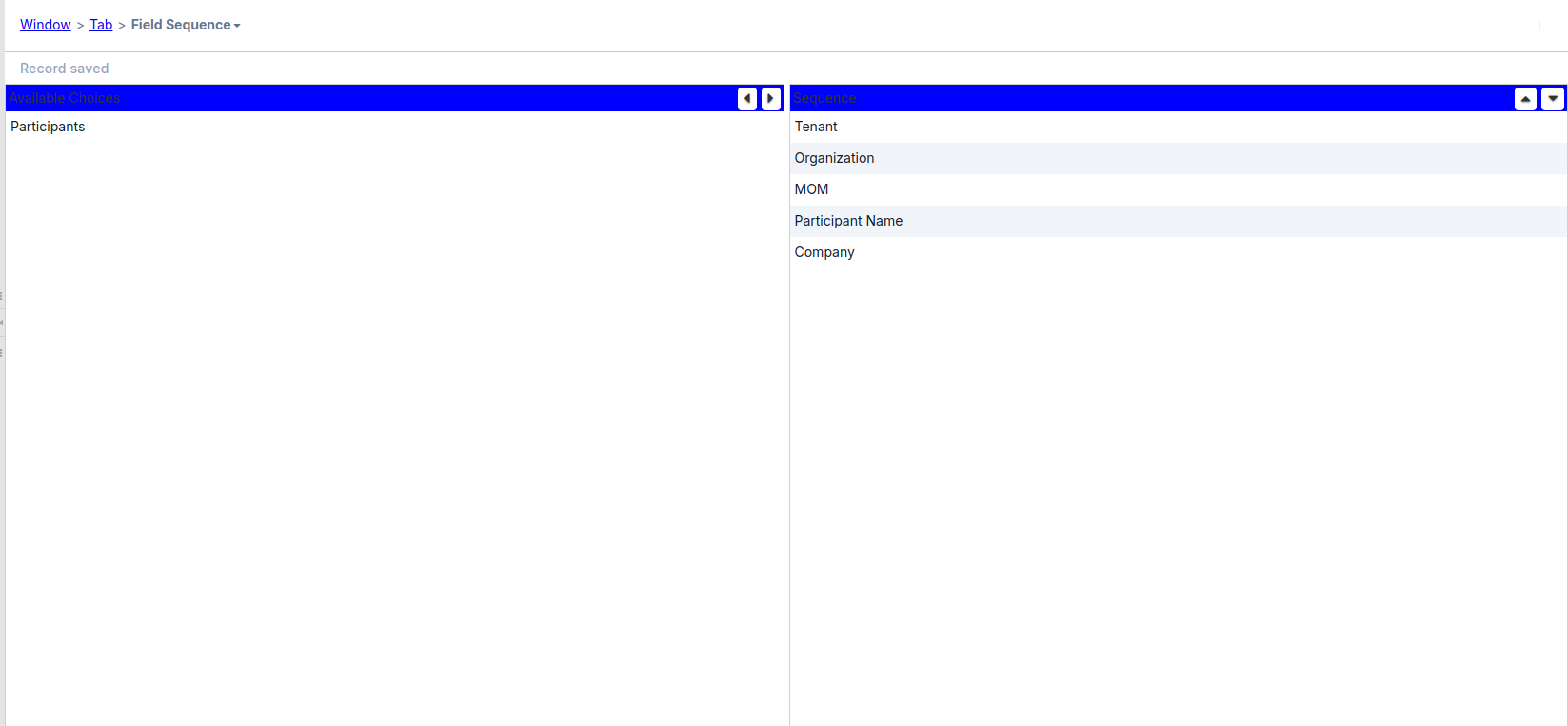
* Name = Participants
* Table: Select the participants table (c\_mom\_participantsline)
* Entity Type = User maintained
* Tab Level: 1
* UnCheck the Single Row Layout Check Box
* Link Column: Select the MOM (c\_mom\_ID\_Minutes of Meeting details)



iDempiere uses **Tab Level** to indent the tab on a window and also, internally, uses it to create SQL joins to fetch the related records. The main tab has the tab level as 0, by default. Moreover, a tab with level 1' becomes the child of the main tab and is indented accordingly. For example, in the previous image, the tab **Access** has a tab level of the main tab—**Window**. Tabs with the same tab level are siblings or peers. For example, in the 

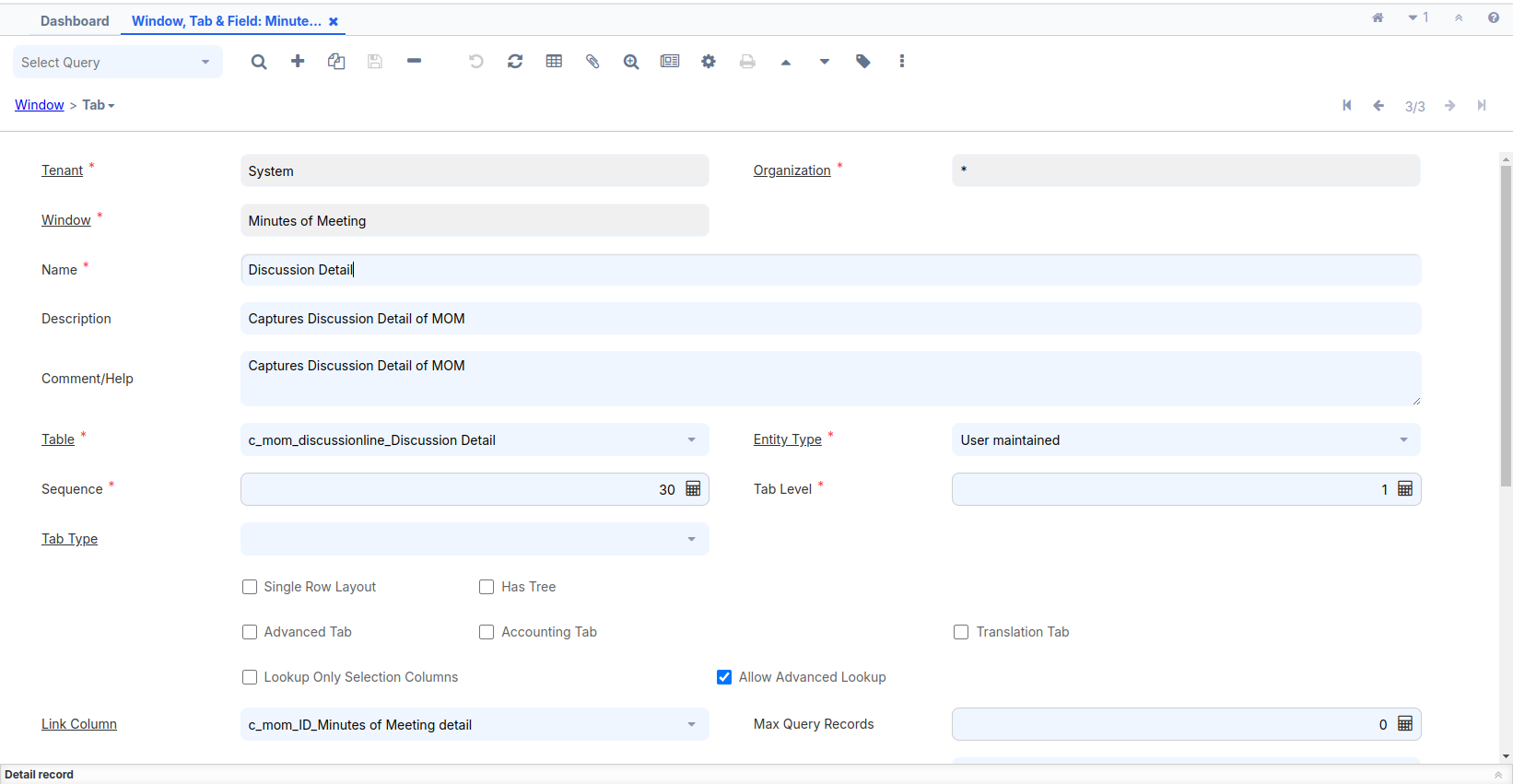
In the previous image, the **Access** and **Tab** tabs are at the same tab level.  
8. The above screen will appear after the tab record is created, and then we create a field.  
Press the **Settings Button** on the middle **toolbar** or press **Alt + O**.

9. Review the **Field Sequence** tab and make the required changes to the sequence, as shown in the following screenshot:

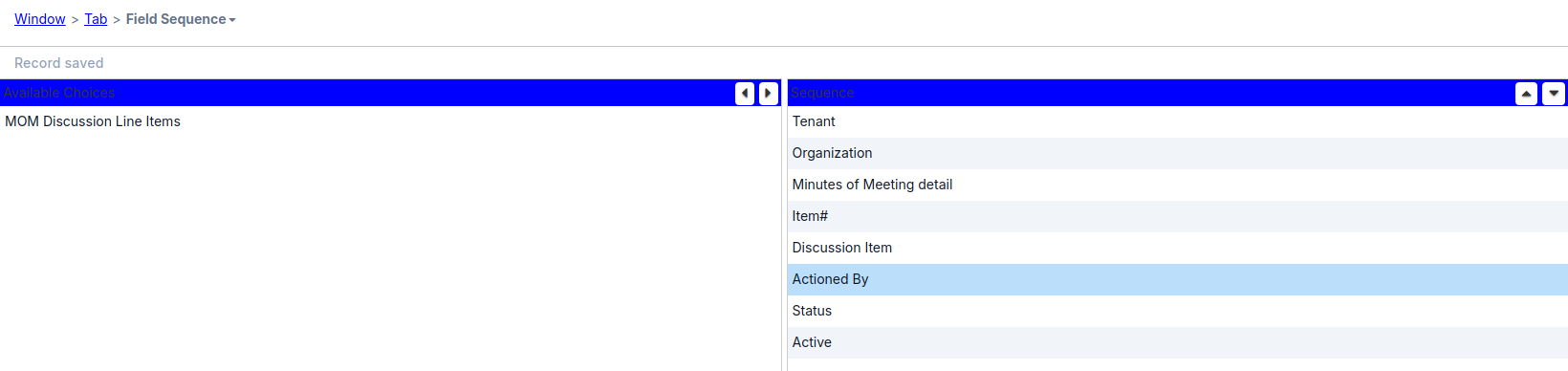


10. Create the **Discussion Detai**l tab and switch to the single row view. The important fields are:

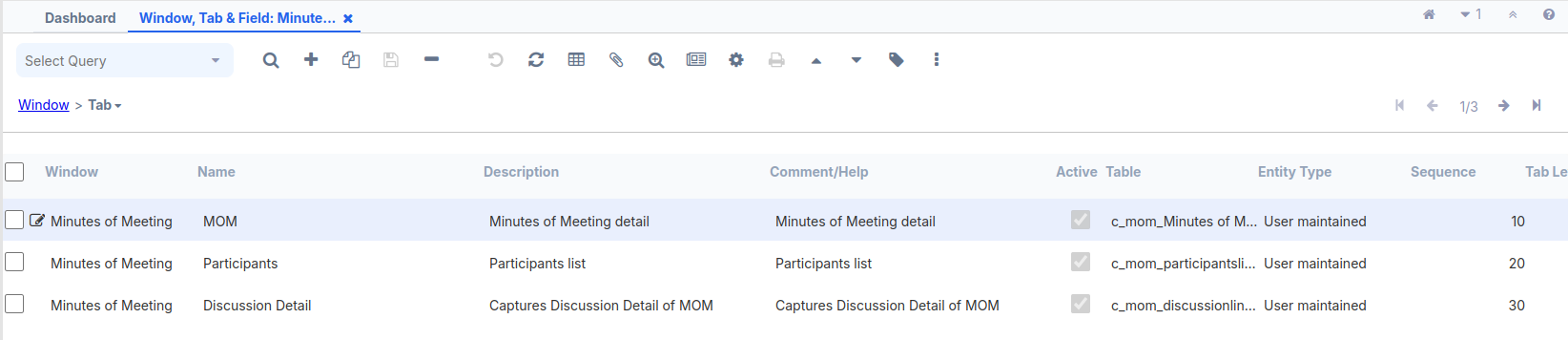
* Name = Discussion Detail
* Table: Select the participants table (c\_mom\_discussionline)
* Entity Type = User maintained
* UnCheck the Single Row Layout Check Box
* Tab Level: 1
* Link Column: Select the MOM (c\_mom\_ID\_Minutes of Meeting details)



11. Review the Field Sequence tab and make the required changes to the sequence, as shown in the following screenshot:



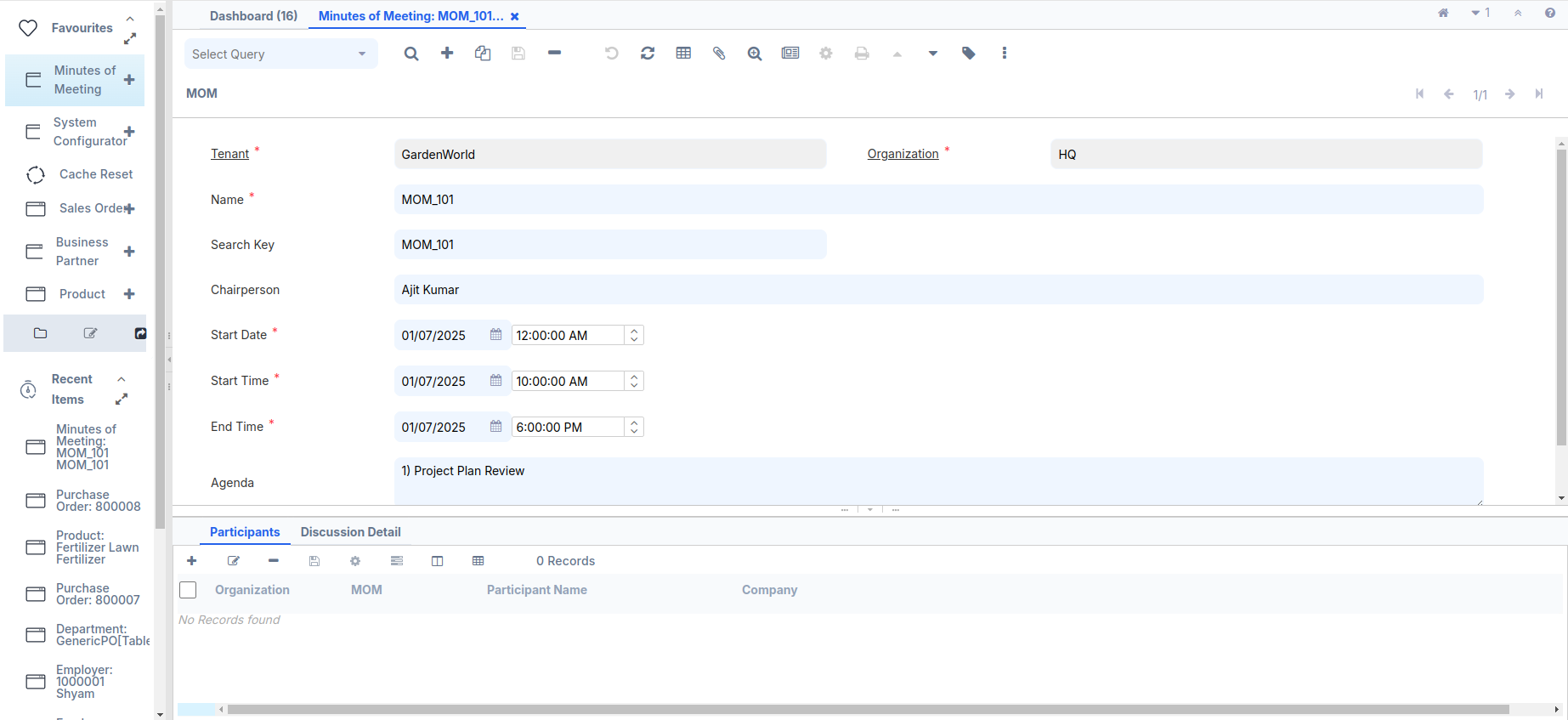
12. c\_mom\_participantsline and c\_mom\_discussionline tables, respectively, as shown in the following screenshot:



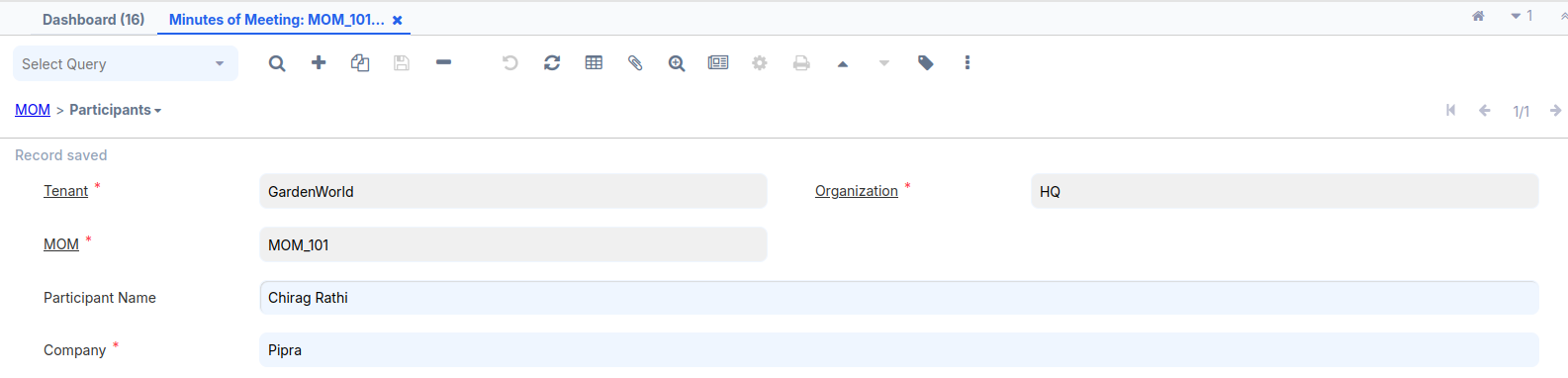
13. The **Sequence** field on the **Tab** screen determines the tab sequence. A tab with the least value of **Sequence** appears as the first tab and the one with the largest value appears as the last tab.

14. Log out and log in as **GardenAdmin**/**GardenAdmin**. You will see the **Minutes Of Meeting** menu tree,Or Use the **search bar** at the top left and enter **Minutes Of Meeting** :

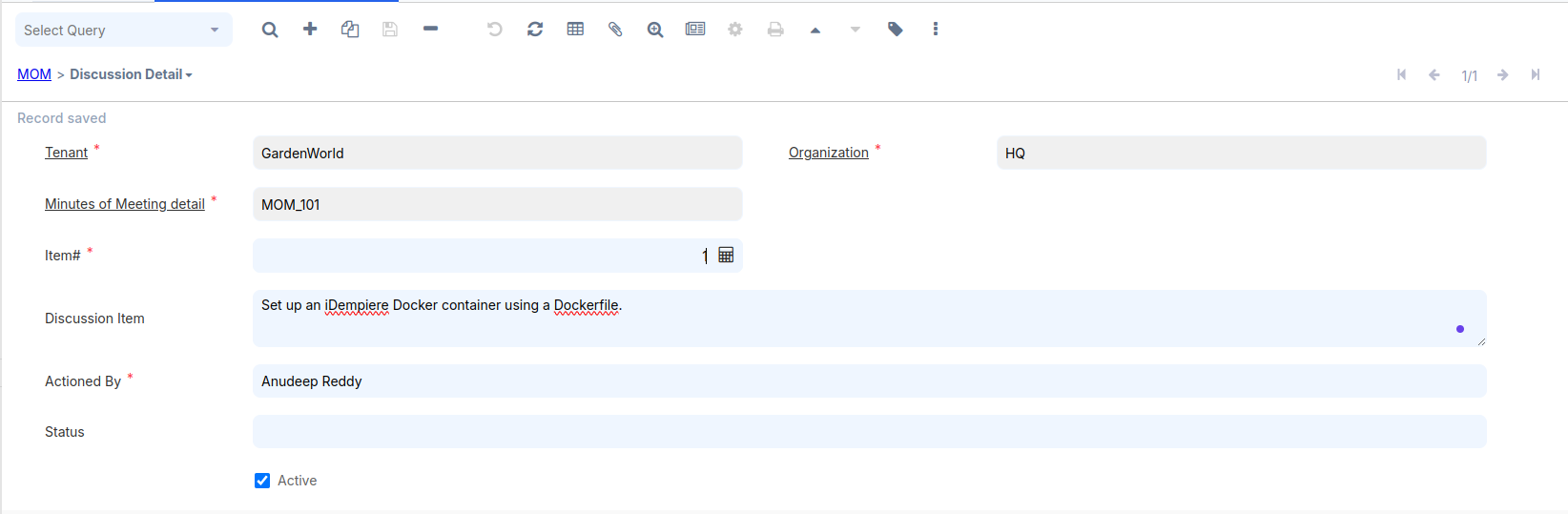
15. Click on the **Minutes Of Meeting** menu tree. This will bring up the **Minutes Of Meeting** window with one tab, **MOM**. Our MOM window now has three tabs and looks more organized, as shown here in the following screenshot:



16. Add one or more participants for a MOM:



17. Add one or more discussion items for a MOM:



# **Creating a Search Widget**

On the MOM tab, we have the Chairperson field where we are entering the   
username.Similarly, we have the participant's name on the Participants tab and the Actioned By person name on the Discussion Detail tab. iDempiere maintains User/Contact detail. To provide a better finishing of our MOM window, it would be good if we can connect these fields with ADempiere's User/Contact so that a user can find the right User/Contact and assign them to a MOM. This way, all these fields need to be made like a search widget where a Search button appears next to these fields. When a user clicks on the **Search** button, he/she will be able to find the User/Contact, and upon selection, the selected User/Contact will appear in the field. As part of this recipe, we will follow through the steps required to convert each of these fields into a search widget.

Log in to Adempiere using the credentials System/System

1. Delete some records from the c\_mom\_participantsline table.

2. Enter the Search bar, go to the **Window**, **Tab** and **Field** sections, enter the window name **Minutes of Meeting**,Go to **Participants** tab, navigate to the **Field** section, and delete the specified fields:

Participants

3. Enter the Search bar, go to the **Table and Column** sections,enter the Table and column name **Participants**,navigate to the **column** section, and delete the specified column:

Participant

Please follow (**Create the following table in your Adempiere schema)** heading for create or alter table query,previously alreadyguided.

**Alter** the adempiere.c\_mom\_participantsline table by executing the following SQL:

ALTER TABLE adempiere.c\_mom\_participantsline DROP COLUMN participant;  
ALTER TABLE adempiere.c\_mom\_participantsline ADD COLUMN ad\_user\_id NUMERIC(10,0) ;  
ALTER TABLE adempiere.c\_mom\_participantsline ADD CONSTRAINT c\_mom\_participantsline\_ad\_user\_id\_fkey FOREIGN KEY (ad\_user\_id)REFERENCES adempiere.ad\_user(ad\_user\_id);

1. Delete some records from the c\_mom table.

2. Enter the Search bar, go to the **Window**, **Tab** and **Field** sections, enter the window name **Minutes of Meeting**,Go to **Minutes of Meeting** tab, navigate to the **Field** section, and delete the specified fields:

Chairperson

3. Enter the Search bar, go to the **Table and Column** sections,enter the Table and column name **Minutes of Meeting**,navigate to the **column** section, and delete the specified column:

chairperson

**Alter** the adempiere.c\_mom table by executing the following SQL:

ALTER TABLE adempiere.c\_mom DROP COLUMN chairperson;  
ALTER TABLE adempiere.c\_mom ADD COLUMN ad\_user\_id NUMERIC(10,0) ;  
ALTER TABLE adempiere.c\_mom ADD CONSTRAINT c\_mom\_ad\_user\_id\_fkey FOREIGN KEY (ad\_user\_id)REFERENCES adempiere.ad\_user(ad\_user\_id);

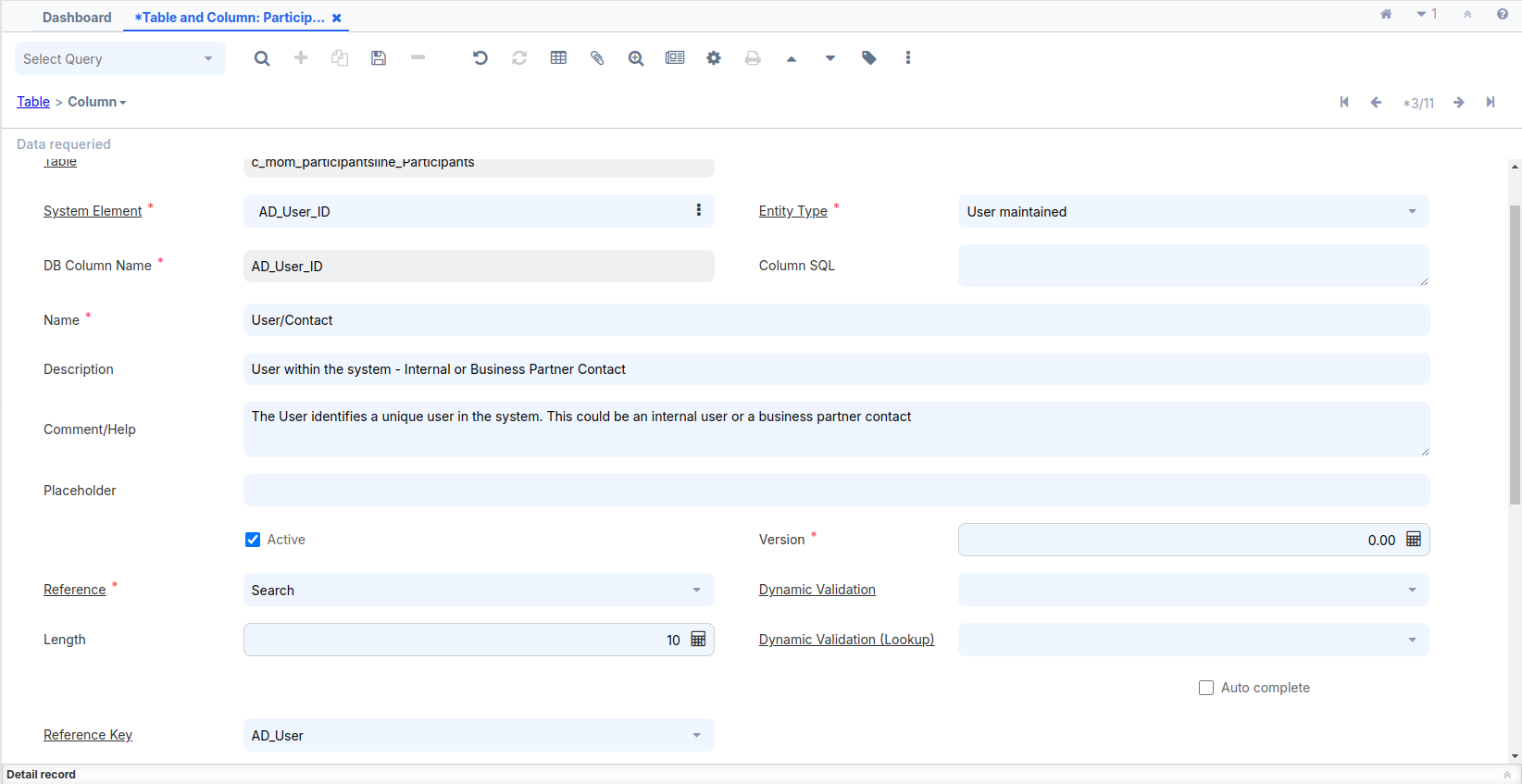
Log in to Adempiere using the credentials System/System. Follow steps:

1. Search for the table and column.
2. Enter the **Minutes of Meeting** table name.
3. Press **Alt + O** and select **Create Columns from DB** to update with new columns.
4. Navigate to the new column and update some fields as needed.

Select the AD\_User\_ID column and specify the following values for the two important fields, which are more relevant in this context :

* **Reference**: Select **Search**
* **Reference** **Key**: Select **AD\_User**

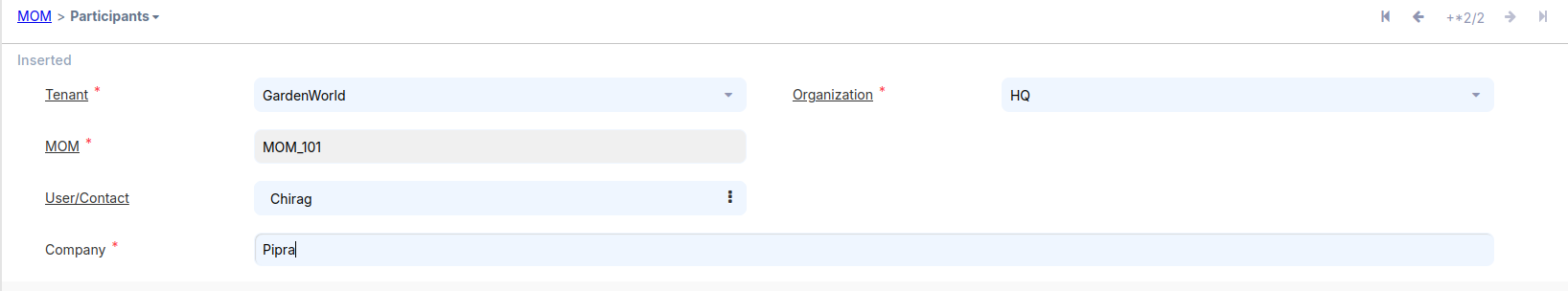
After completed above same step do **Participants** table name



Go to the Window, **Tab**, and Field window and look at the details of the **Minutes Of Meeting** window.

Click on the **Create Fields** button so that the newly added **User/Contact** column is added as a field. Change the sequence of the **User/Contact** field such that it comes before Company.  
Rename the field name as follows:  
1. In the **Minutes of Meeting** tab, change the field name **User/Contact** to   
 **Chairperson**.

2. In the **Participants** tab, change the field name **User/Contact** to   
 **Participant**.  
Log out and log in as **GardenAdmin**/**GardenAdmin** with the GardenWorld Admin role.  
Go to the **Minutes Of Meeting** window and go to the **Participants** tab. You will notice that the **User/Contact** field is populated, by default, with the current logged in username and a search button appears next to the field, as shown in the following screenshot:



# **Populating the Combo-box list**

There is a **Status** field on the **Discussion Detail** tab. Currently, this field accepts free text.For various practical reasons, such as, to maintain the consistency in communication, to support project guidelines, to have clean data for further analysis and reporting, and so on, it makes sense to convert the **Status** field into a combo-box and the possible values are listed for the user selection. In this recipe, we will follow the steps to convert the **Status** field into a combo-box and configure it in such a way that it is populated with values.

Log in to the **System**/**System** account, and remove the **status** **field** in the Window, Tab, and Field window. After that, remove the **status column** in the Table and Column window. Once both steps are completed, **drop** the column from the **PostgreSQL** table.

1. Delete some records from the c\_mom\_discussionline table.

2. Enter the Search bar, go to the **Window**, **Tab** and **Field** sections, enter the window name **Minutes of Meeting**,Go to **Discussion Detail** tab, navigate to the **Field** section, and delete the specified fields:

Status

3. Enter the Search bar, go to the **Table and Column** sections,enter the Table and column name **Discussion Detail**,navigate to the **column** section, and delete the specified column:

status

Drop the status column by executing the following SQL:

ALTER TABLE adempiere.c\_mom\_discussionline DROP COLUMN status;

Create a new table, c\_momstatus, to capture the list of status values by executing the following SQL:

CREATE TABLE adempiere.c\_mom\_status (  
c\_mom\_status\_id NUMERIC(10,0) NOT NULL PRIMARY KEY,  
c\_mom\_status\_uu VARCHAR(36) DEFAULT NULL::bpchar,  
ad\_client\_id NUMERIC(10, 0) NOT NULL,  
ad\_org\_id NUMERIC(10, 0) NOT NULL,  
name varchar(25) NOT NULL,  
value character varying(40) NOT NULL,  
created TIMESTAMP without time zone DEFAULT now() not null,  
createdby numeric(10,0) not null,  
updated TIMESTAMP without time zone DEFAULT now() not null,  
updatedby NUMERIC(10,0) not null,  
description VARCHAR(255),  
isactive CHAR(1) not null DEFAULT 'Y'::bpchar,  
isdefault character(1) NOT NULL DEFAULT 'N');

Log in to the **System**/**System** account and create a **table and columns** for the new table. Similarly, create the corresponding **window, tab, field**, and **menu**. These steps have already been performed previously for the **c\_mom** table.

Create C\_mom\_status table:

DB Table Name = c\_mom\_status  
Name = MOM Status  
Description and Comment filed added your according or fill empty  
Data Access Level = Client+Organization  
Entity Type = User maintained  
Check the Show in Drill Options Check Box  
After saving the table press Alt + O and select **Create Column from DB** and all columns created.

After done press Alt + O and Select **Create Window,Tab & Field from Table**, which is the third option on the list

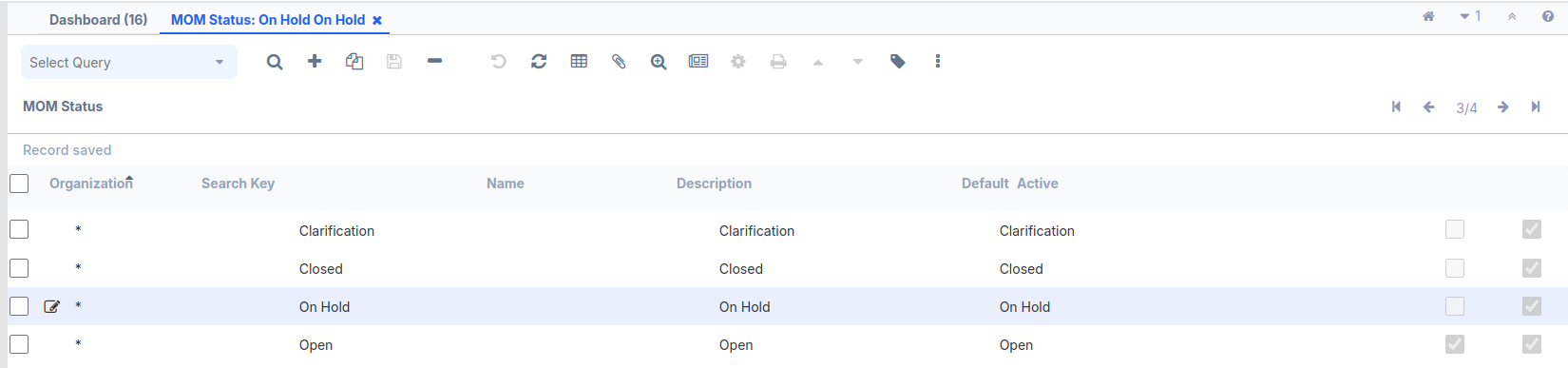
A new pop-up window will appear.

If we want to create a window, tab, and field, we can also create a menu.

Select the **Create Menu** checkbox and press **OK**.

Once the menu is created, it will be visible in the search bar.

Log out and log in as **GardenAdmin**/**GardenAdmin** with the GardenWorld Admin role.  
Go to the **MOM Status** window and create different statuses. For example, Let us say that Open is marked as the default status, as shown in the following screenshot:



The c\_mom\_status\_ID column is linked to the c\_mom\_discussionline table. A foreign key should be used to establish the relationship between the two tables.

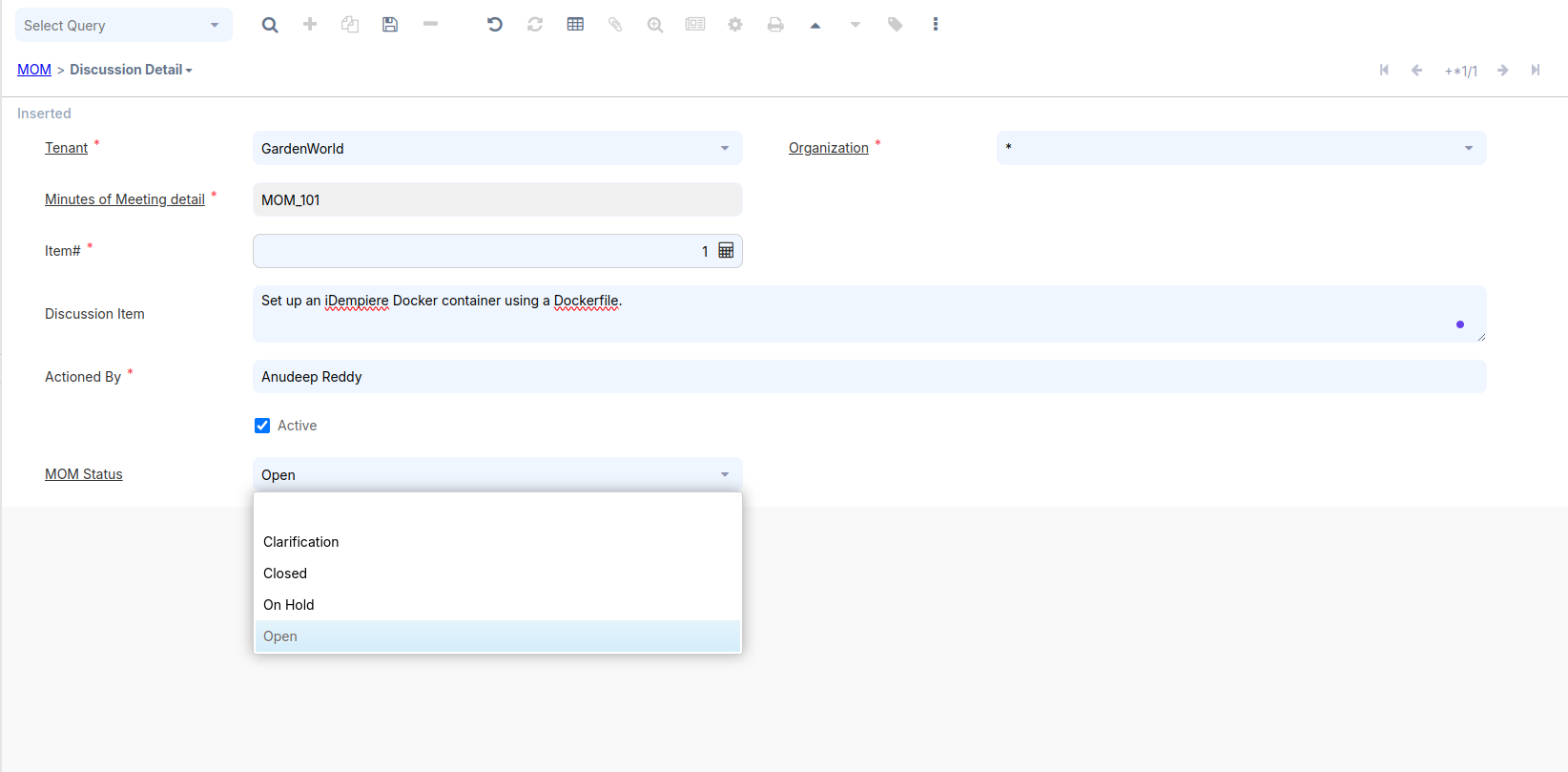
**Alter** the adempiere.c\_mom\_discussionline table by executing the following SQL:

ALTER TABLE adempiere.c\_mom\_discussionline ADD COLUMN c\_mom\_status\_id NUMERIC(10,0) ;  
ALTER TABLE adempiere.c\_mom\_discussionline ADD CONSTRAINT c\_mom\_discussionline\_c\_mom\_status\_id\_fkey FOREIGN KEY (c\_mom\_status\_id) REFERENCES adempiere.c\_mom\_status(c\_mom\_status\_id);

Log in to Adempiere using the credentials System/System. Follow steps:

1. Search for the table and column.
2. Enter the **Discussion Detail** table name.
3. Press **Alt + O** and select **Create Columns from DB** to update with new columns.
4. Navigate to the new column and update some fields as needed.
5. Go to the Window, **Tab**, and Field window and look at the details of the **Minutes Of Meeting** window.
6. Click on the Create **Fields** button so that the newly added **MOM Status** column is added as a field.

Log out and log in as **GardenAdmin**/**GardenAdmin** with the GardenWorld Admin role.  
Go to the **Minutes Of Meeting** window and go to the **Discussion Detail** tab. You will notice that the **MOM Status** field is populated, by default, with the current logged in username and a **Combo-Box list** appears next to the field, as shown in the following screenshot:



**CallOut**

Callout is mainly used to perform some dynamic actions when a value in a particular field is changed. Each field in ADempiere maps to a column in a particular table, so callout is configured at column level. **Callout** is a Java method that gets invoked when there is a change in the field data in a tab.

# **How to do it…**

1. **For Eclipse  
   We have a Maven plug-in (org.mom.calloutandprocessanddocaction) that needs to be imported into Eclipse.**Include the required code to import the plugin. Then, in the   
   **Run Configurations**, located in the **Run** dropdown in the top-middle   
   toolbar, select the plugin and ensure the following settings:

* Set the **Start Level** to 1.
* Enable **Auto-Start** by setting it to true.

After making these changes, apply them and run the application.

**OR**First, download the **org.mom.calloutandprocessanddocaction** JAR file. Then, use this JAR file within the **OSGi** framework.

Added Plug-in using **OSGI** frame work:-   
 A. Open your browser and go to:<https://localhost:8443/>  
 B. A new pop-up window will appear. On the top right, you will  
 see three tools. Click on the **Settings** tool.

C. use credential **SuperUser/System**  
 D. Click **Install/Update** at the top right.  
 E. A new pop-up window will open—check the first two checkboxes,   
 select your JAR file, and press the **Install/Update** button.  
 F. After the installation is complete, search for the plugin (mom).  
 G. Start the plugin, refresh the page, and the **OSGi** framework  
 setup will be completed.

**Java Code:-**

package org.mom.callout;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.Properties;

import java.util.logging.Level;

import org.compiere.model.CalloutEngine;

import org.compiere.model.GridField;

import org.compiere.model.GridTab;

import org.compiere.util.CLogger;

import org.compiere.util.DB;

import org.compiere.util.Env;

public class MyCallout extends CalloutEngine{

CLogger log = CLogger.getCLogger(MyCallout.class);

public String getNextItemNbr (Properties ctx, int WindowNo,

GridTab mTab, GridField mField, Object value)

{

Integer momId = (Integer)mTab.getValue("c\_mom\_ID");

String sql = "SELECT MAX(item\_nbr) "

+ "FROM c\_mom\_discussionline "

+ "WHERE c\_mom\_ID=?";

PreparedStatement pstmt = null;

ResultSet rs = null;

try {

pstmt = DB.prepareStatement(sql, null);

pstmt.setInt(1, momId);

rs = pstmt.executeQuery();

Integer maxItemNbr = 0;

if (rs.next())

{

maxItemNbr = rs.getInt(1);

Env.setContext(ctx, WindowNo, "item\_nbr", maxItemNbr+1);

mTab.setValue("item\_nbr", maxItemNbr+1);

}

DB.close(rs, pstmt);

rs = null;

pstmt = null;

}catch(SQLException e) {

log.log(Level.SEVERE,sql,e);

return e.getLocalizedMessage();

}finally {

DB.close(rs,pstmt);

rs = null;

pstmt = null;

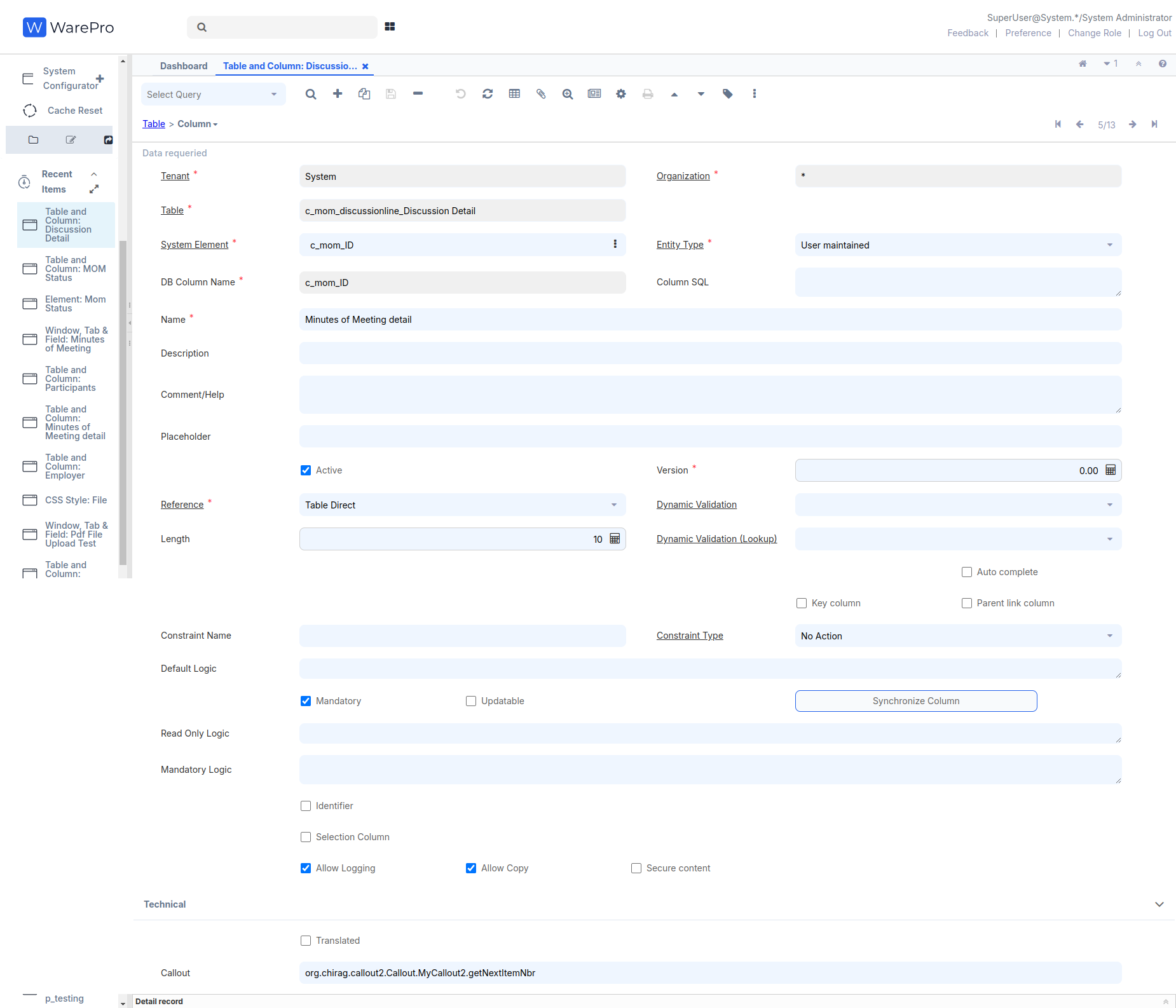
}

return "";

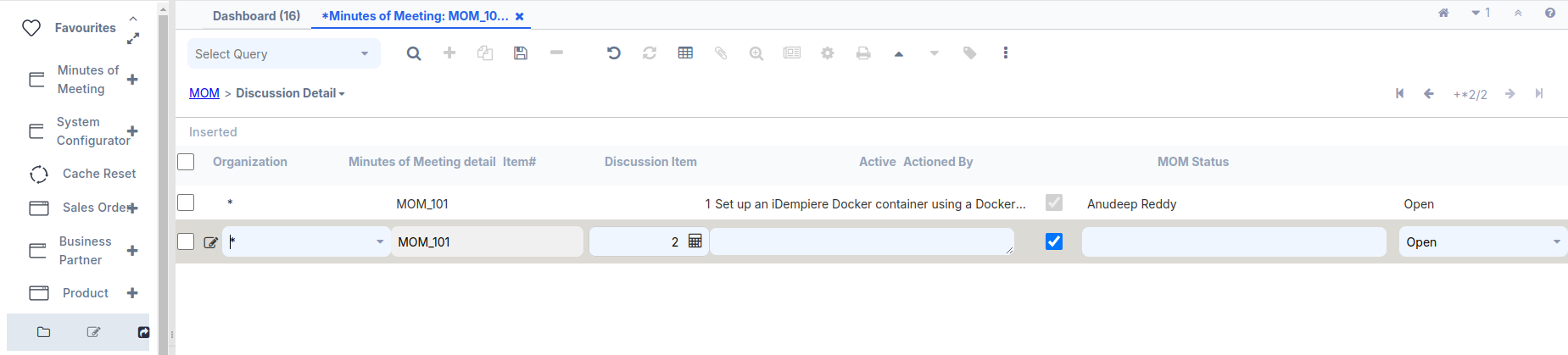
}

}

1. Launch the desktop version of the project and log in as **System/System** with the **System Administrator** role.
2. Go to the **Table and Column** window and open the details of the c\_mom\_ID column in the c\_mom\_discussionline table and mention org.mom.callout.MyCallout.getNextItemNbr as the **Callout**, as shown in the following screenshot:

  
After saving the record, go to the search bar, enter **Cache Reset**, and click **OK**.

1. Log out and log in as **GardenAdmin/GardenAdmin** with the **GardenWorld Admin** role.
2. Go to the **Discussion Detail** tab of the **Minutes Of Meeting** window and click on the **New Record (+ Icon)** toolbar button. The item number is automatically generated and the sequence is maintained, as shown in the next screenshot:



**Process**

A process is another way to execute business logic. Usage of a process is not limited to a window, tab, or field. It is not related to any table or column. You can write a process and link it with a button on a window/tab for further processing (for example, the Complete button on the Sales Order window). Moreover, we can write any business logic inside it and invoke it directly. All the items that appear with an icon in the Menu tree represent a process. This recipe will walk us through the steps to create a new process and invoke it. Here we are going to create a process, which prompts the user to select a MOM from the existing MOMs, and sends MOM-related mail to the participants of the selected MOM.

# **How to do it…**

If the steps below are already performed in the **Callout**, then there is no need to do them again.

First, download the **org.mom.calloutandprocessanddocaction** JAR file. Then, use this JAR file within the **OSGi** framework.

Added Plug-in using **OSGI** frame work:-   
 A. Open your browser and go to:<https://localhost:8443/>  
 B. A new pop-up window will appear. On the top right, you will  
 see three tools. Click on the **Settings** tool.

C. use credential **SuperUser/System**  
 D. Click **Install/Update** at the top right.  
 E. A new pop-up window will open—check the first two checkboxes,   
 select your JAR file, and press the **Install/Update** button.  
 F. After the installation is complete, search for the plugin (mom).  
 G. Start the plugin, refresh the page, and the **OSGi** framework  
 setup will be completed.

**Java Code:-**

package org.mom.process;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.logging.Level;

import org.compiere.model.MClient;

import org.compiere.model.MUser;

import org.compiere.process.ProcessInfoParameter;

import org.compiere.process.SvrProcess;

import org.compiere.util.DB;

import org.compiere.util.EMail;

import org.mom.model.Mmom;

public class SendMOMMail extends SvrProcess {

private int m\_c\_mom\_ID = -1;

private Mmom m\_mom = null;

private int m\_AD\_User\_ID = -1;

private MClient m\_client = null;

private int m\_AD\_Client\_ID = -1;

private MUser m\_from = null;

private ArrayList<Integer> m\_list = new ArrayList<Integer>();

private int m\_counter = 0;

private int m\_errors = 0;

@Override

protected void prepare() {

ProcessInfoParameter[] para = getParameter();

for (int i = 0; i < para.length; i++) {

String name = para[i].getParameterName();

if (para[i].getParameter() == null)

;

else if (name.equals("c\_mom\_ID"))

m\_c\_mom\_ID = para[i].getParameterAsInt();

else

log.log(Level.SEVERE, "Unknown Parameter: " + name);

}

}

@Override

protected String doIt() throws Exception {

log.info("c\_mom\_ID=" + m\_c\_mom\_ID);

m\_mom = new Mmom(getCtx(), m\_c\_mom\_ID, get\_TrxName());

m\_AD\_User\_ID = m\_mom.getCreatedBy();

// Client Info

m\_client = MClient.get(getCtx());

m\_AD\_Client\_ID = m\_client.getAD\_Client\_ID();

if (m\_AD\_Client\_ID == 0)

throw new Exception("Not found @AD\_Client\_ID@");

if (m\_client.getSMTPHost() == null || m\_client.getSMTPHost().length() == 0)

throw new Exception("No SMTP Host found");

if (m\_AD\_User\_ID > 0) {

m\_from = new MUser(getCtx(), m\_AD\_User\_ID, get\_TrxName());

if (m\_from.getAD\_User\_ID() == 0)

throw new Exception("No found @AD\_User\_ID@=" + m\_AD\_User\_ID);

}

log.fine("From " + m\_from);

if(m\_AD\_Client\_ID > 0) {

m\_client = new MClient(getCtx(), m\_AD\_Client\_ID, get\_TrxName());

if(m\_AD\_Client\_ID == 0)

throw new Exception("No found @As\_Client\_ID=" + m\_AD\_Client\_ID);

}

log.fine("From " + m\_client);

long start = System.currentTimeMillis();

send2Participants();

return "@Created@=" + m\_counter + ", @Errors@=" + m\_errors + " - " +

(System.currentTimeMillis() - start) + "ms";

}

private void send2Participants() {

log.info("C\_Mom\_ID=" + m\_c\_mom\_ID);

String sql = "SELECT a.ad\_user\_id " + "FROM adempiere.c\_mom\_participantsline a " +

"WHERE a.c\_mom\_id=?";

PreparedStatement pstmt = null;

try {

pstmt = DB.prepareStatement(sql, get\_TrxName());

pstmt.setInt(1, m\_c\_mom\_ID);

ResultSet rs = pstmt.executeQuery();

while (rs.next()) {

Boolean ok = sendIndividualMail(rs.getInt(1));

if (ok == null)

;

else if (ok.booleanValue())

m\_counter++;

else

m\_errors++;

}

rs.close();

pstmt.close();

pstmt = null;

} catch (SQLException ex) {

log.log(Level.SEVERE, sql, ex);

}

try {

if (pstmt != null)

pstmt.close();

} catch (SQLException ex1) {

}

pstmt = null;

}

private Boolean sendIndividualMail(int AD\_User\_ID) {

Integer ii = new Integer(AD\_User\_ID);

if (m\_list.contains(ii))

return null;

m\_list.add(ii);

MUser to = new MUser(getCtx(), AD\_User\_ID, null);

String message = "Hi " + to.getFirstName() + "\n " + m\_mom.getName()

+ " has been updated in the system!\nThanks";

String subject = "MOM : " + m\_mom.getName();

// Validate sender email

if (m\_client.getRequestEMail() == null || m\_client.getRequestEMail().isEmpty()) {

System.out.println("Sender email is not configured: " + m\_client.getName());

return Boolean.FALSE;

}

// Validate recipient email

if (to.getEMail() == null || to.getEMail().isEmpty()) {

System.out.println("Recipient email is not configured: " + to.getName());

return Boolean.FALSE;

}

EMail email = m\_client.createEMailFrom(m\_client.getRequestEMail(), to.getEMail(), subject, message,false);

if (email == null) {

System.out.println("Failed to create email. Check SMTP configuration and email addresses.");

return Boolean.FALSE;

}

email.setSubject(subject);

email.setMessageText(message);

if (!email.isValid() && !email.isValid(true)) {

log.warning("NOT VALID - " + email);

to.setIsActive(false);

to.addDescription("Invalid EMail");

to.save();

return Boolean.FALSE;

}

boolean OK = EMail.SENT\_OK.equals(email.send());

if (OK)

log.fine(to.getEMail());

else

log.warning("FAILURE - " + to.getEMail());

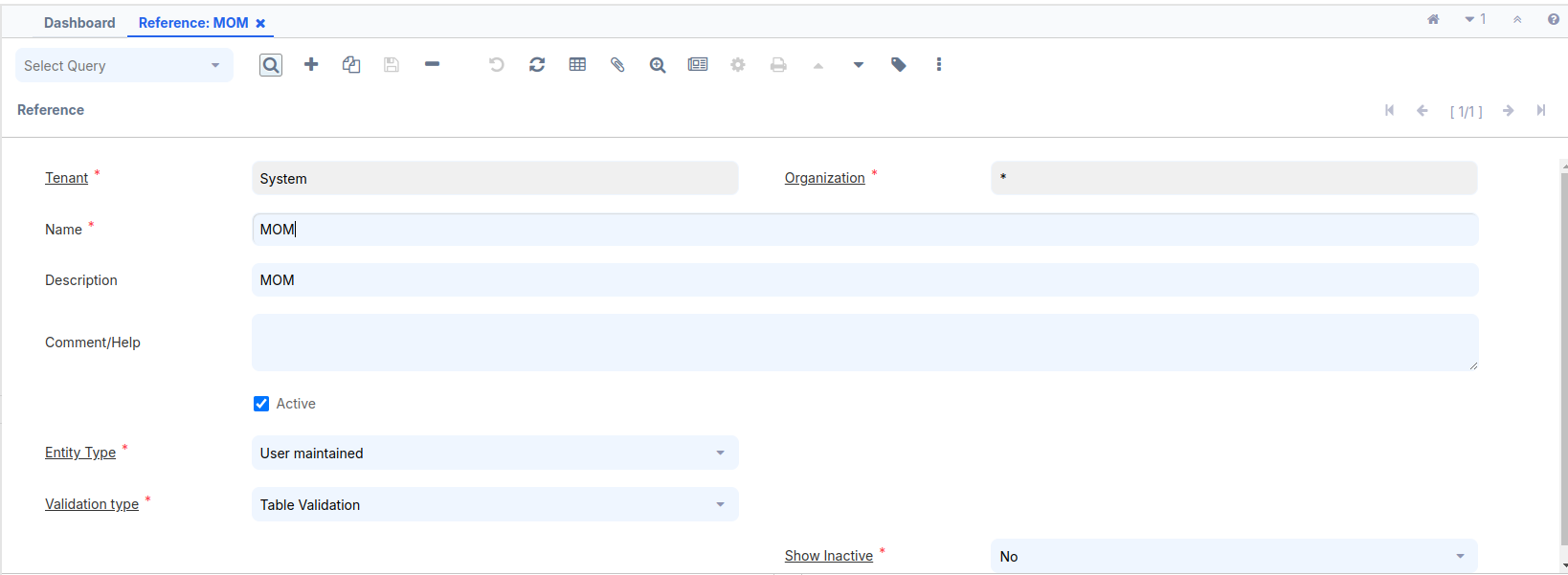
addLog(0, null, null, (OK ? "@OK@" : "@ERROR@") + " - " + to.getEMail());

return new Boolean(OK);

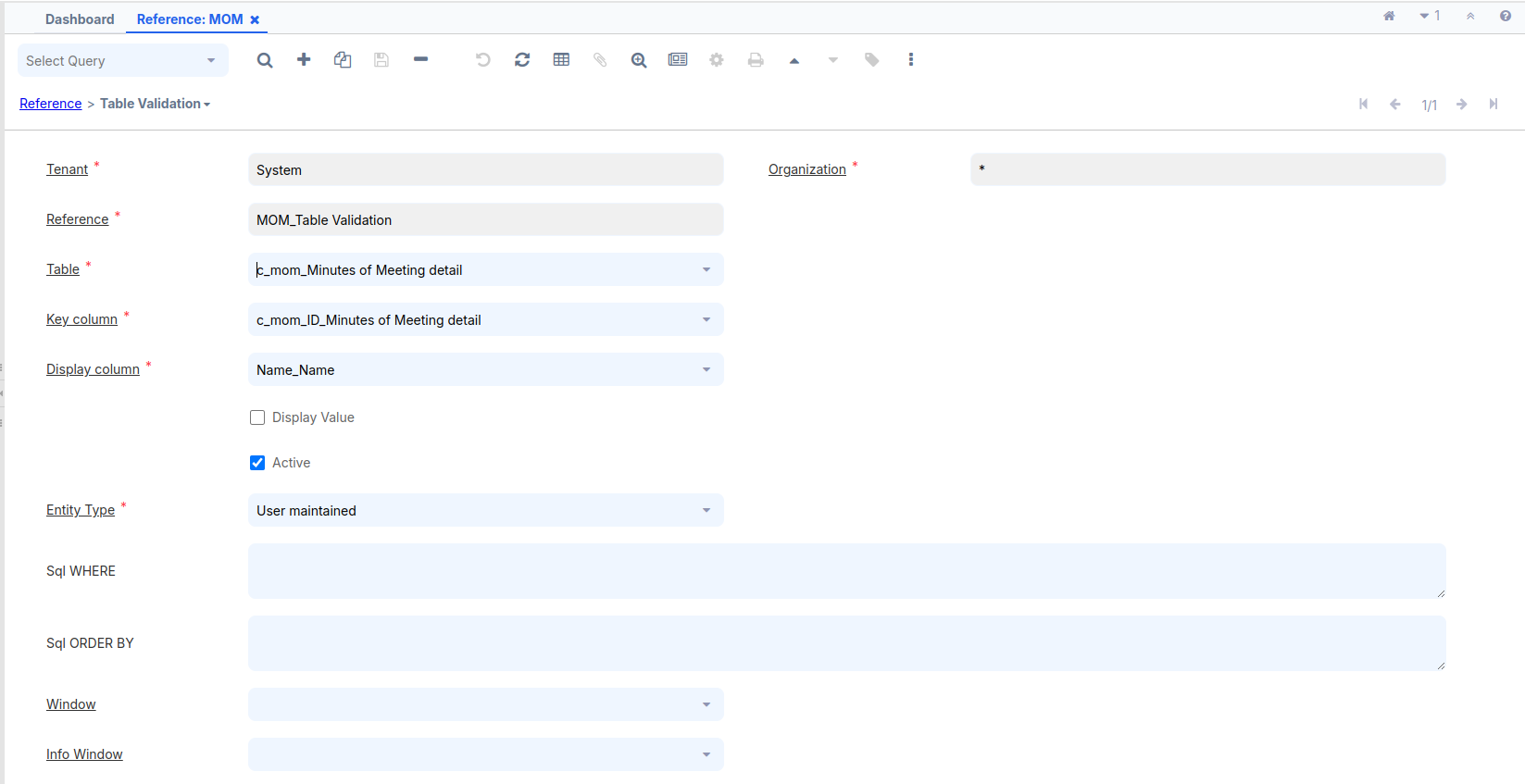
}

}

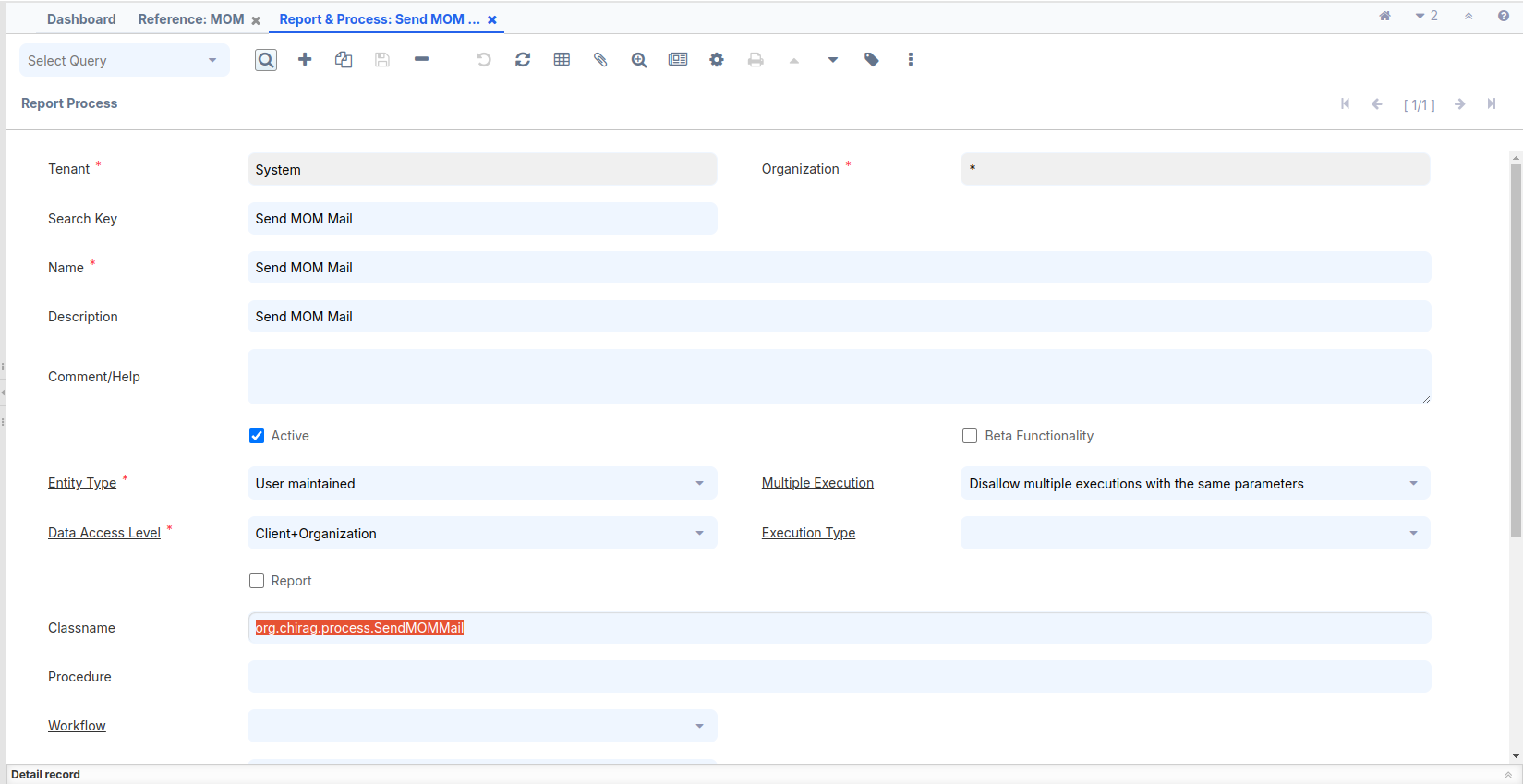
* Log in as **System**/**System** with the **System Administrator** role.
* Go to the **Menu** | **Application Dictionary** | **Reference** window **Or** Enter the Search bar, go to the **Reference** window and create a new reference, **MOM**, as shown in the following screenshot:



* Go to the **Table Validation** tab and create a new record for the c\_mom table and the c\_mom\_ID column, as shown in the following screenshot:

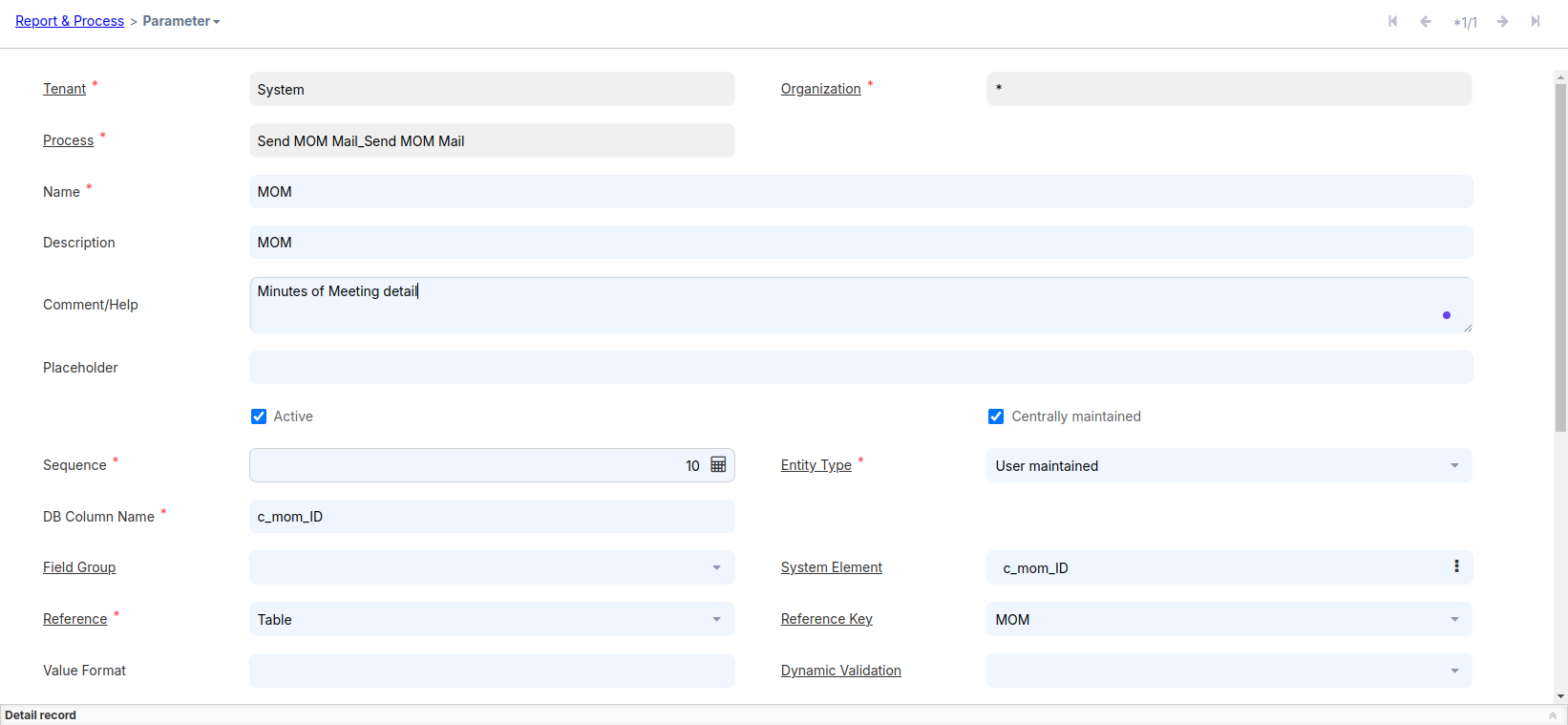


* Go to the **Menu** | **Application Dictionary** | **Report & Process** window and create a new record—**Send MOM Mail**. In the **Classname**, mention the **org.mom.process.SendMOMMail** process name, as shown in the following screenshot:



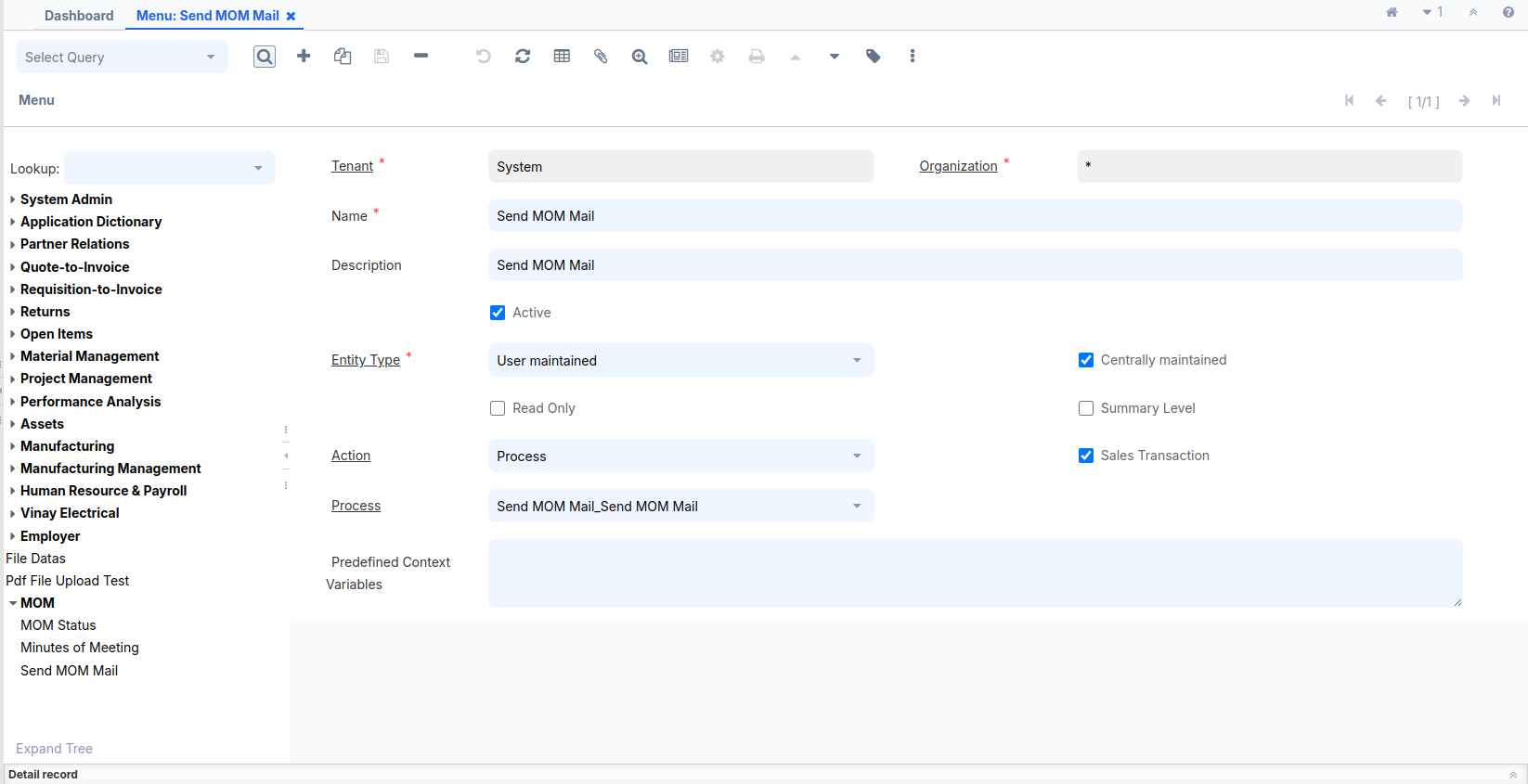
* Go to the **Parameter** tab to create a process parameter where we can select the **MOM** for which e-mail needs to be sent to their participants.
* Create a new parameter, as shown in the following screenshot, with the following important details:

1. **DB Column Name**: c\_mom\_ID
2. **System Element**: c\_mom\_ID
3. **Reference**: Table
4. **Reference Key**: MOM (this is the reference that we created in the previous steps)



Save the record.

* Go to the **Menu** | **System Admin** | **General Rules** | **System Rules** | **Menu** window and create a new menu item—**Send MOM Mail**. Add this menu item to the **MOM** node, as shown in the following screenshot:

  
After saving the record, go to the search bar, enter **Cache Reset**, and click **OK**.

* Log out and log in as **GardenAdmin**/**GardenAdmin** with the **GardenWorld Admin** role.
* Enter the Search bar,Go to the **Tenant** window and fill all required fields below mention, click on the **Test EMail** button. If everything is correct, you will see a success message, as shown in the following screenshot:

Before pressing the **Test Email** button, ensure that some  
 important fields are filled; otherwise, an error message will   
 appear instead of a success message.

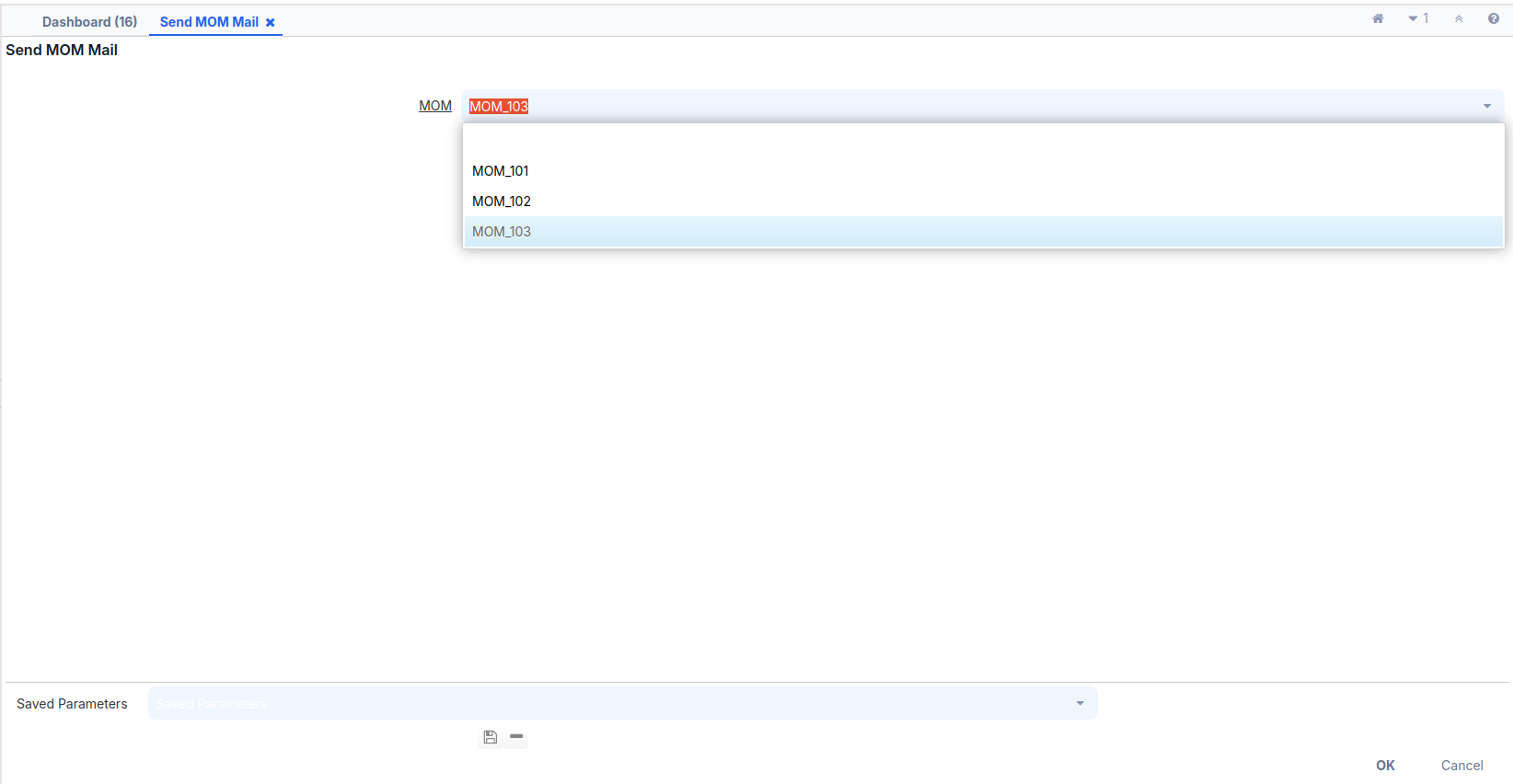
* **Mail Host = smtp.gmail.com**
* **SMTP Port = 587**
* **Request EMail =** [**chiragrathiji111@gmail.com**](mailto:chiragrathiji111@gmail.com)
* **Request User =** [**chiragrathiji111@gmail.com**](mailto:chiragrathiji111@gmail.com)
* **Request Folder = request**
* **Check the SMTP Authentication Check Box**
* **Request User Password = Need to generate**

# **Steps to Enable 2-Step Verification and Generate an App Password:**

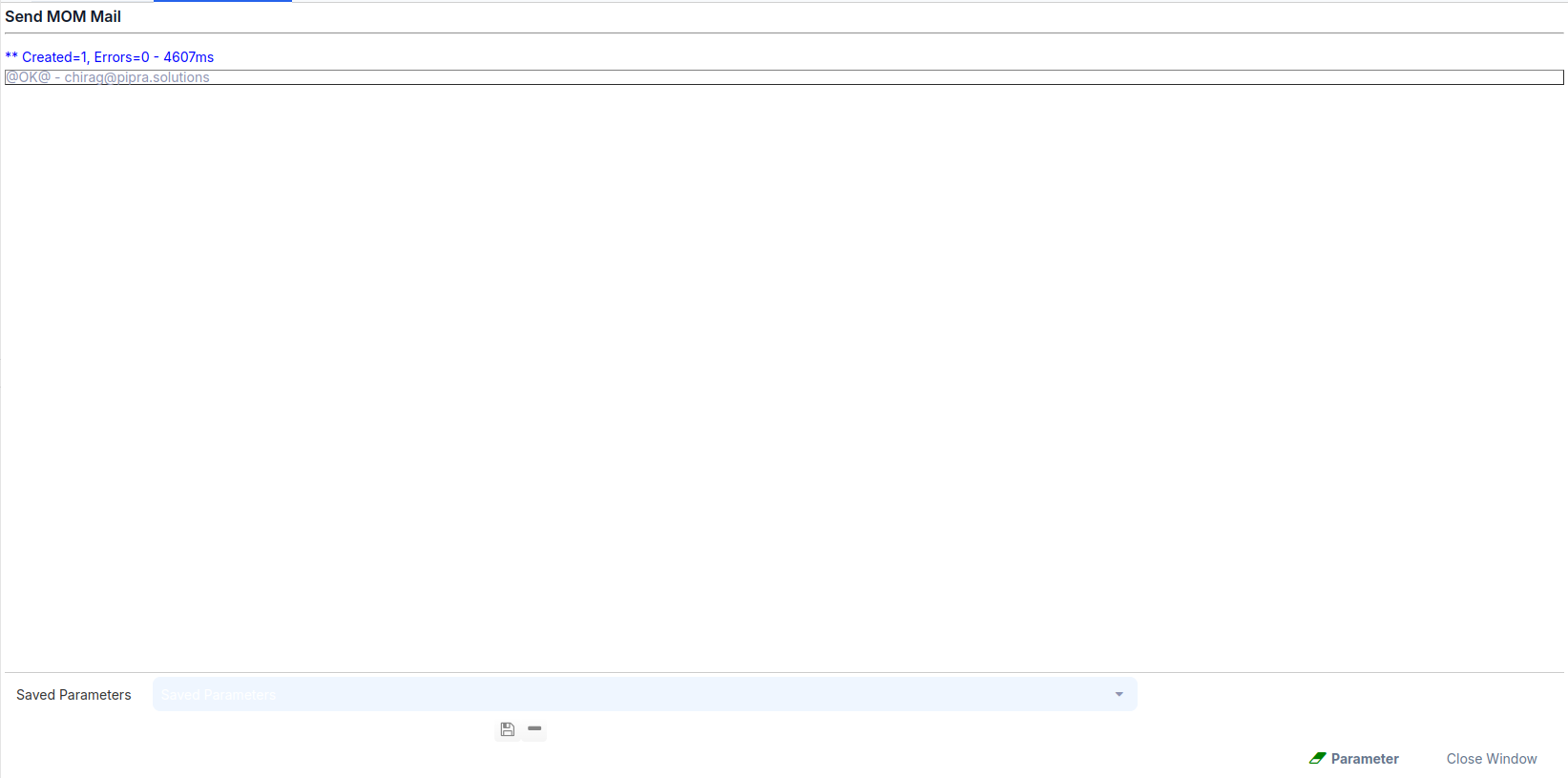
1. **Access Your Google Account:**
   * Open your email account.
   * Click on your Google account icon (profile picture) in the top-right corner.
   * From the dropdown menu, click **Manage your Google Account**. This will open a new page within the same window.
2. **Enable 2-Step Verification:**
   * On the new page, locate the **Security** option in the left-side navigation panel and click on it.
   * Scroll down to the **2-Step Verification** section in the middle of the page.
   * Check if 2-Step Verification is enabled. If it is not, follow the steps to enable it by verifying your phone number or another verification method.
3. **Generate an App Password:**
   * After enabling 2-Step Verification, return to the Security section.
   * Scroll down to the **App Passwords** option and click on it.
   * You will be prompted to enter your Google account password.
   * On the App Passwords page, select an app or use case (e.g., **Mail** or **Custom App**) and generate the app password.
   * A **16-digit password** will be displayed. Copy this password and save it securely in a text editor or password manager.
   * Note: Once you close this page, you cannot retrieve the password again. Make sure to save it securely before leaving the page.
4. **Use the App Password:**
   * Use the generated **16-digit password** in place of your normal Google account password when setting up the selected app or service.

| Copy and paste this 16-digit password into the **Request User Password** field and click **Save**.    Click the **Test Email** button, and in the new pop-up window, press **OK** to see the **Successful** message. |
| --- |

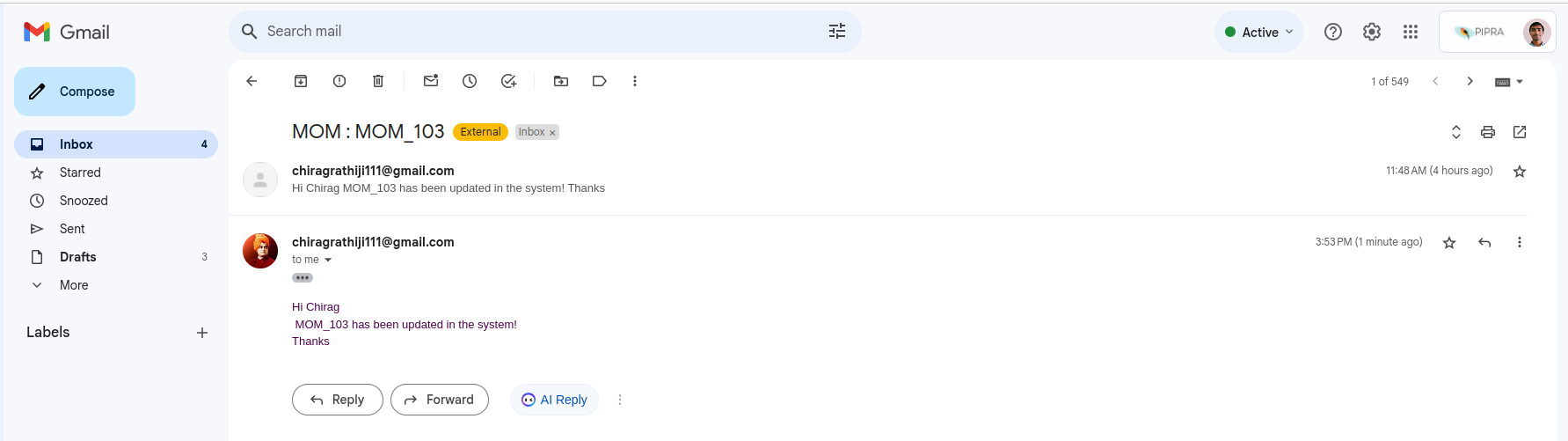
* Click on the **Send MOM Mail** process. It will prompt you to select a **MOM**, as shown in the following screenshot:



* Select an **MOM** from the drop-down list and click on the **OK** button. This will send the e-mail and present the status, finally, as shown in the following screenshot:

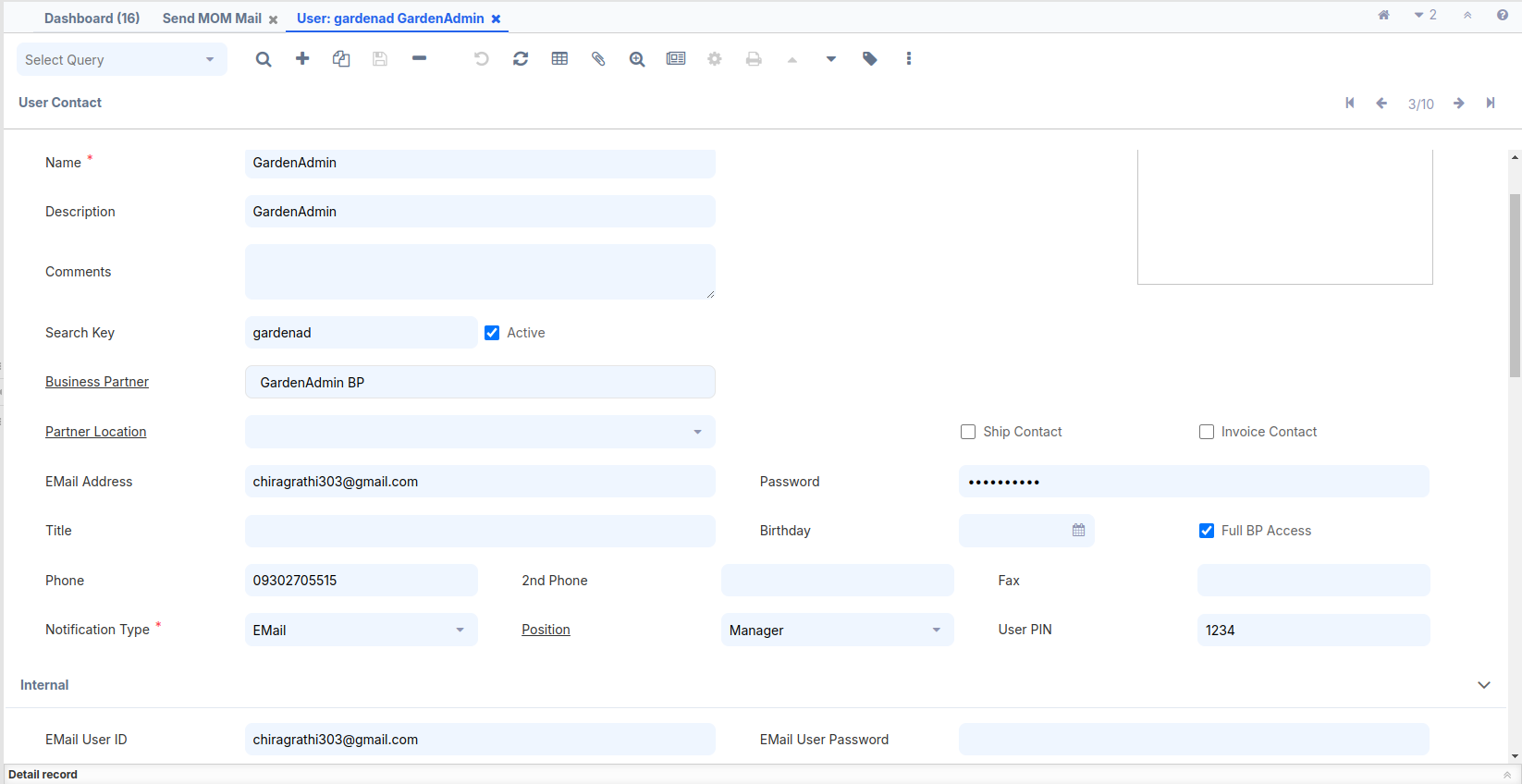


* Check your mailbox to see if you have received e-mail with the MOM details, as shown in the next screenshot:



If you encounter an error, it means the **User** does not have an **Email** address. To check this:

1. Verify which user you selected, such as the **Minute of Meeting** record. Below it, navigate to the **Participants** tab, open the record, and check the **User** name.
2. Then, go to the **User** window through the search bar and check the email address for that user.



**Report**

The main purpose of creating a report is to display specific data according to user requirements. For example, if a window has 10 fields, but we want to show only 5 fields, and filter the data based on certain parameters and date criteria, we can create the report accordingly.

iDempiere supports creating reports in two ways:-

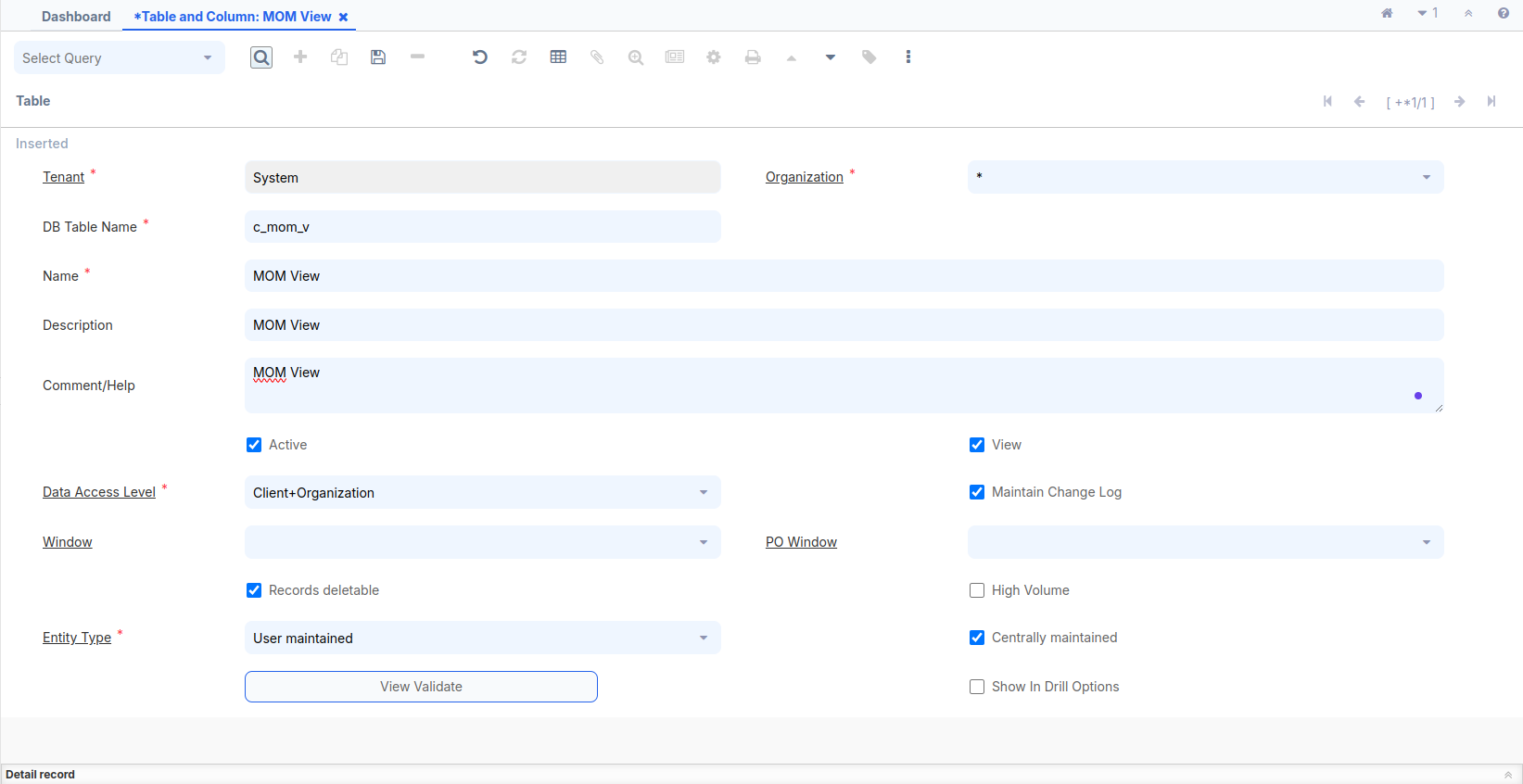
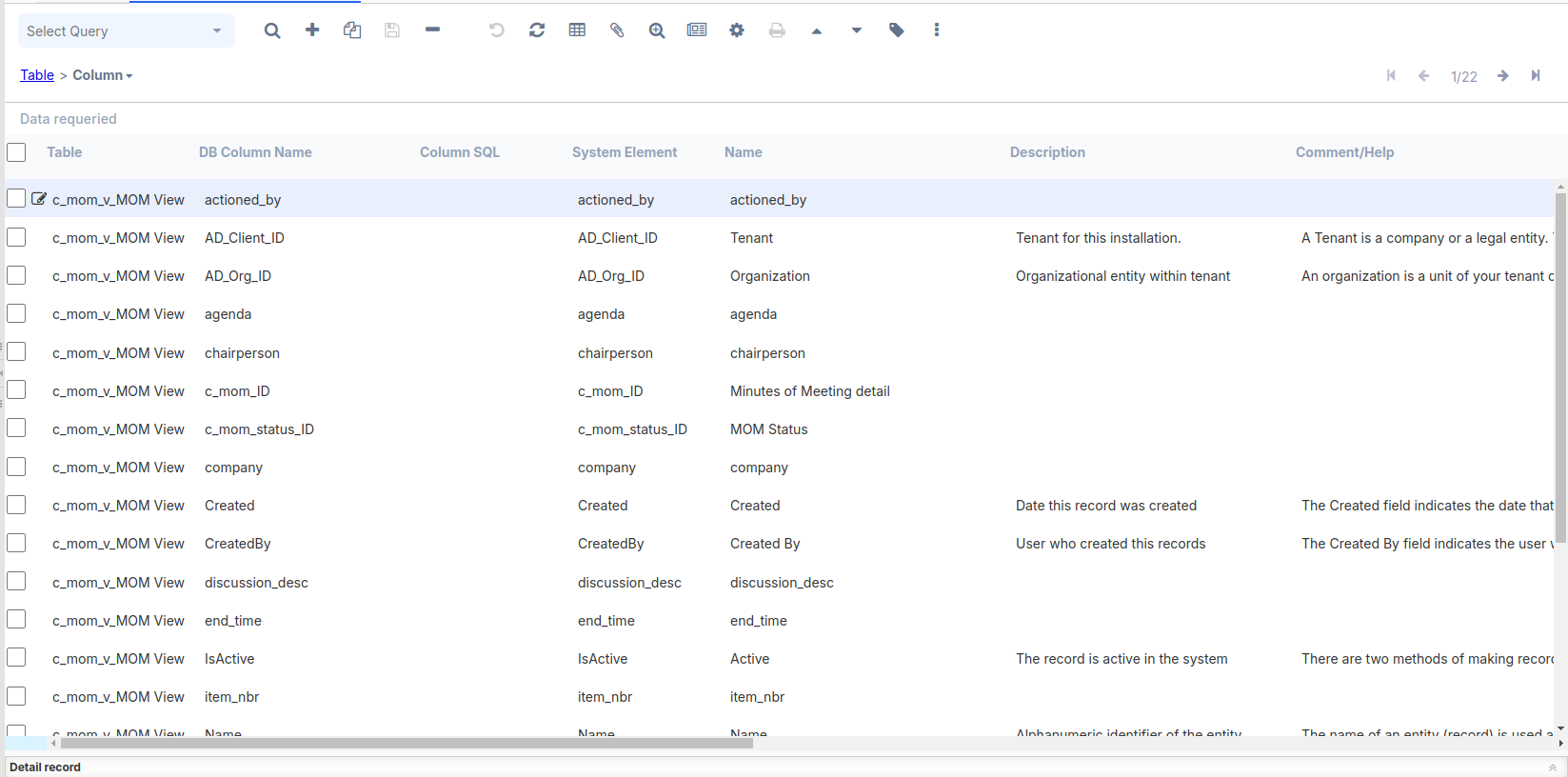
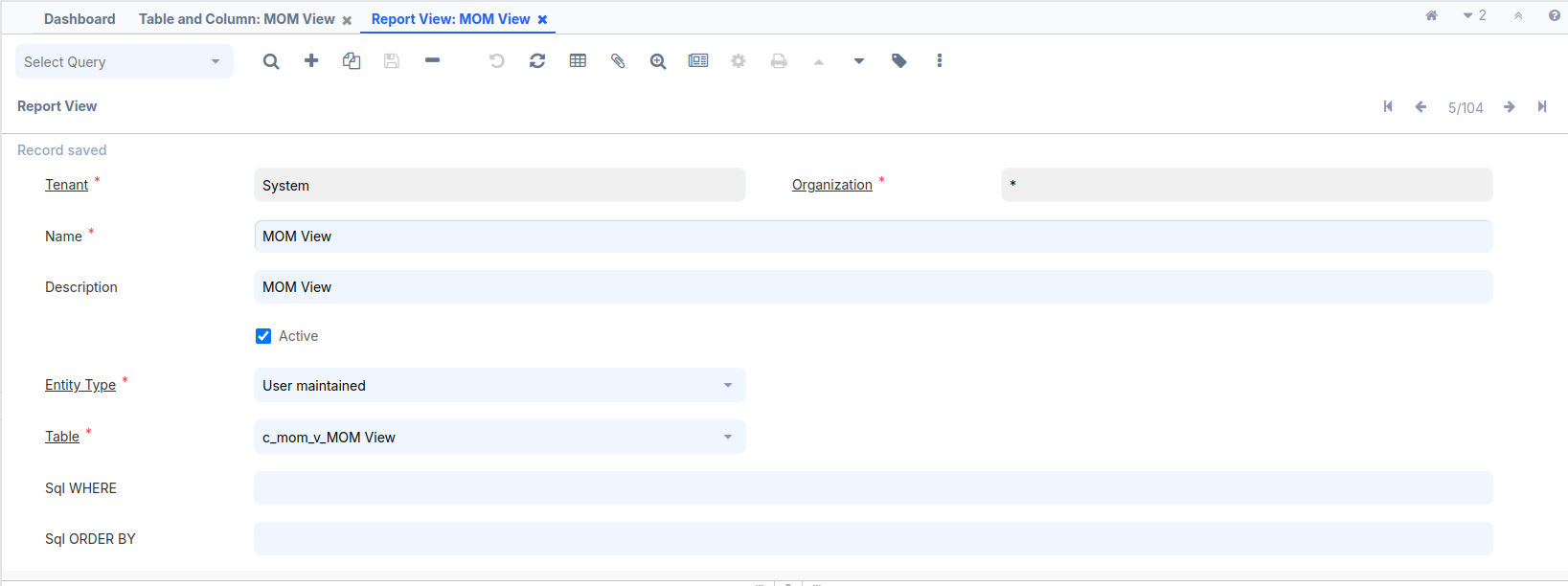
1. Create a report view table based on an existing table.
2. Use a SQL query to create a Jasper report.

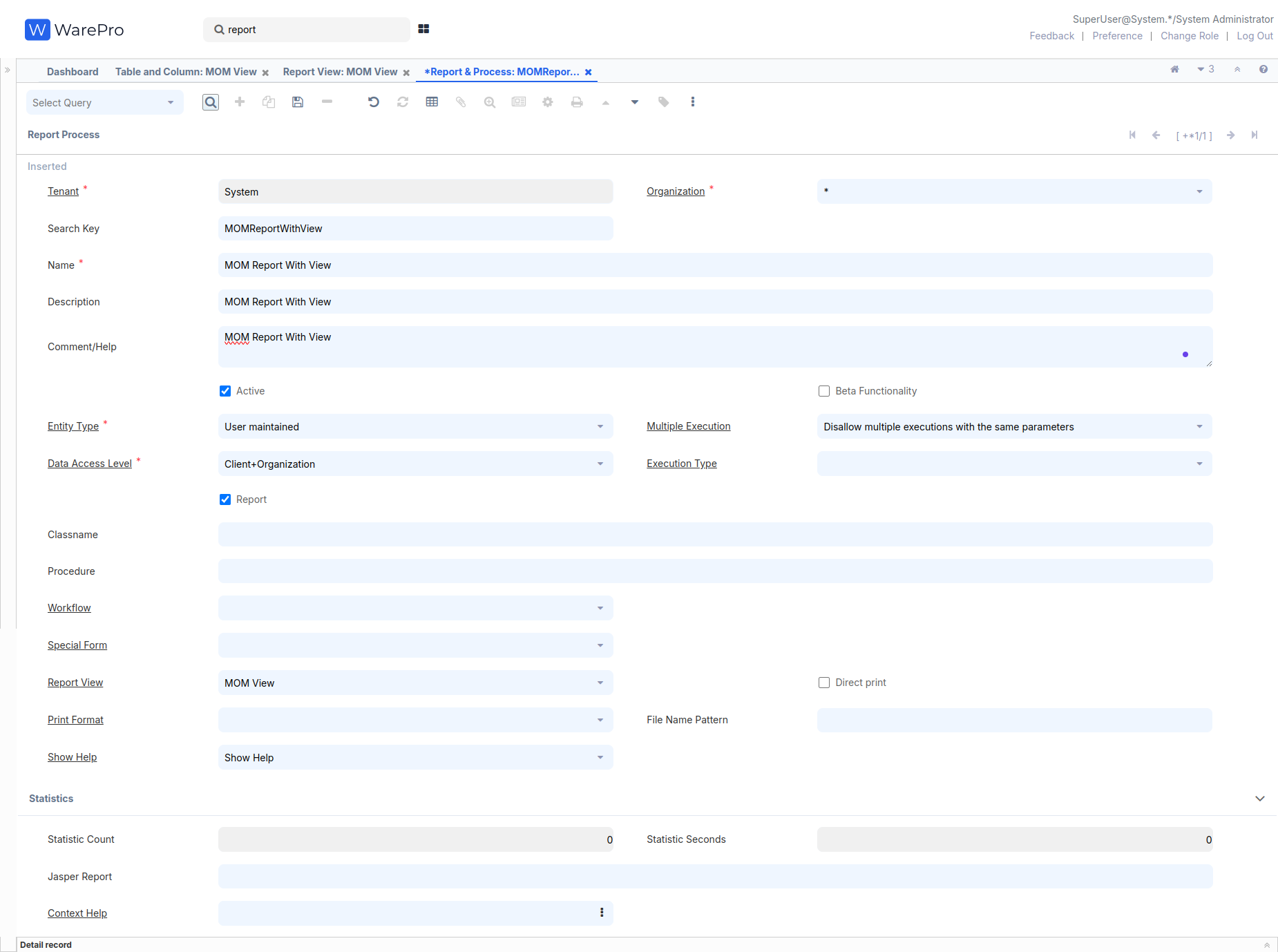
**Create a report view table based on an existing table:-**

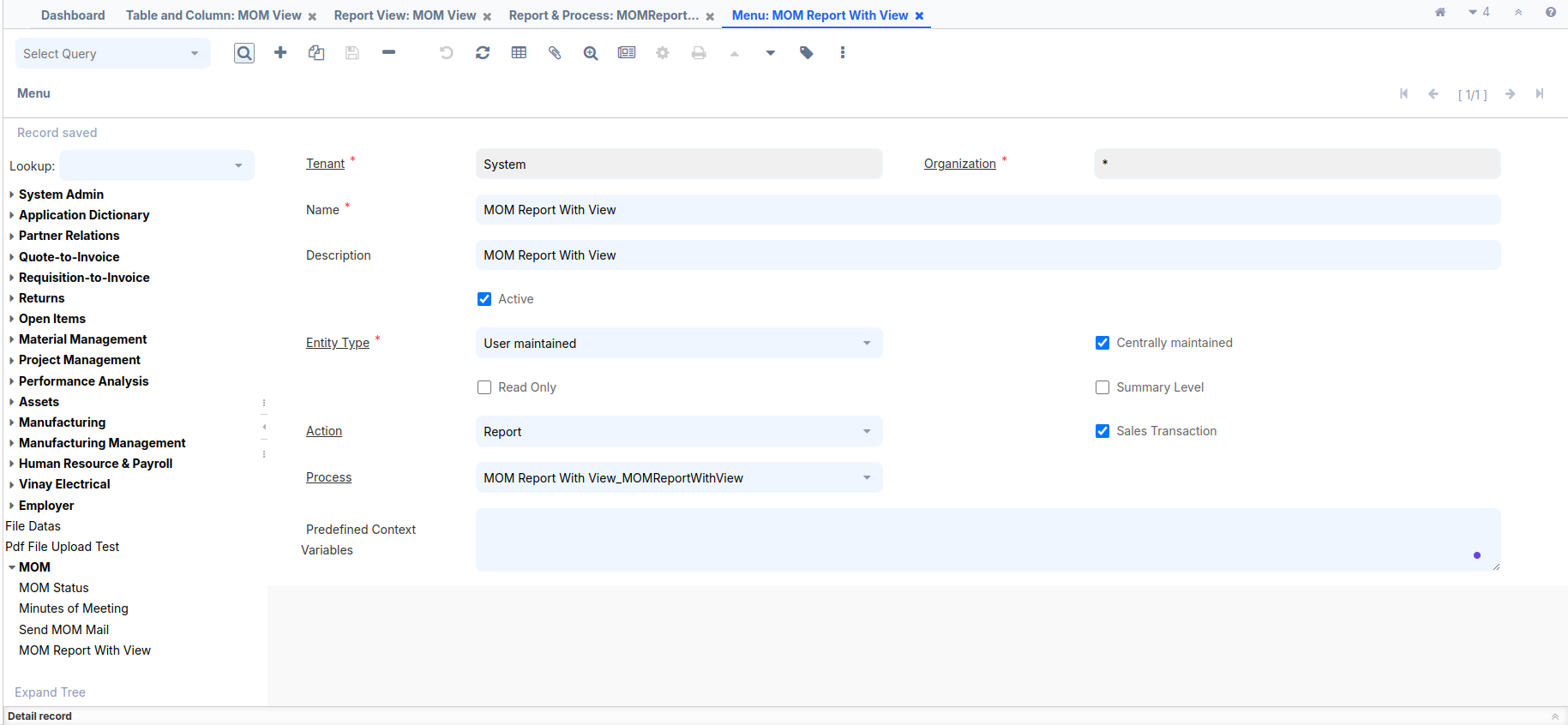
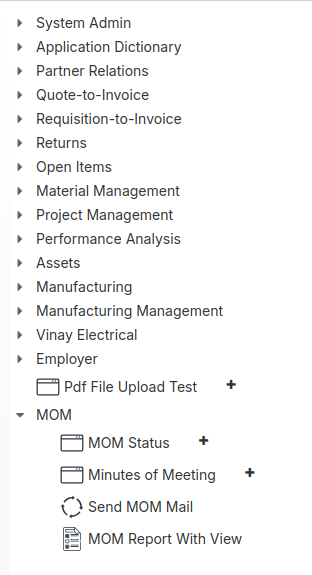
If we want to create a report using the Report View feature, follow these steps:

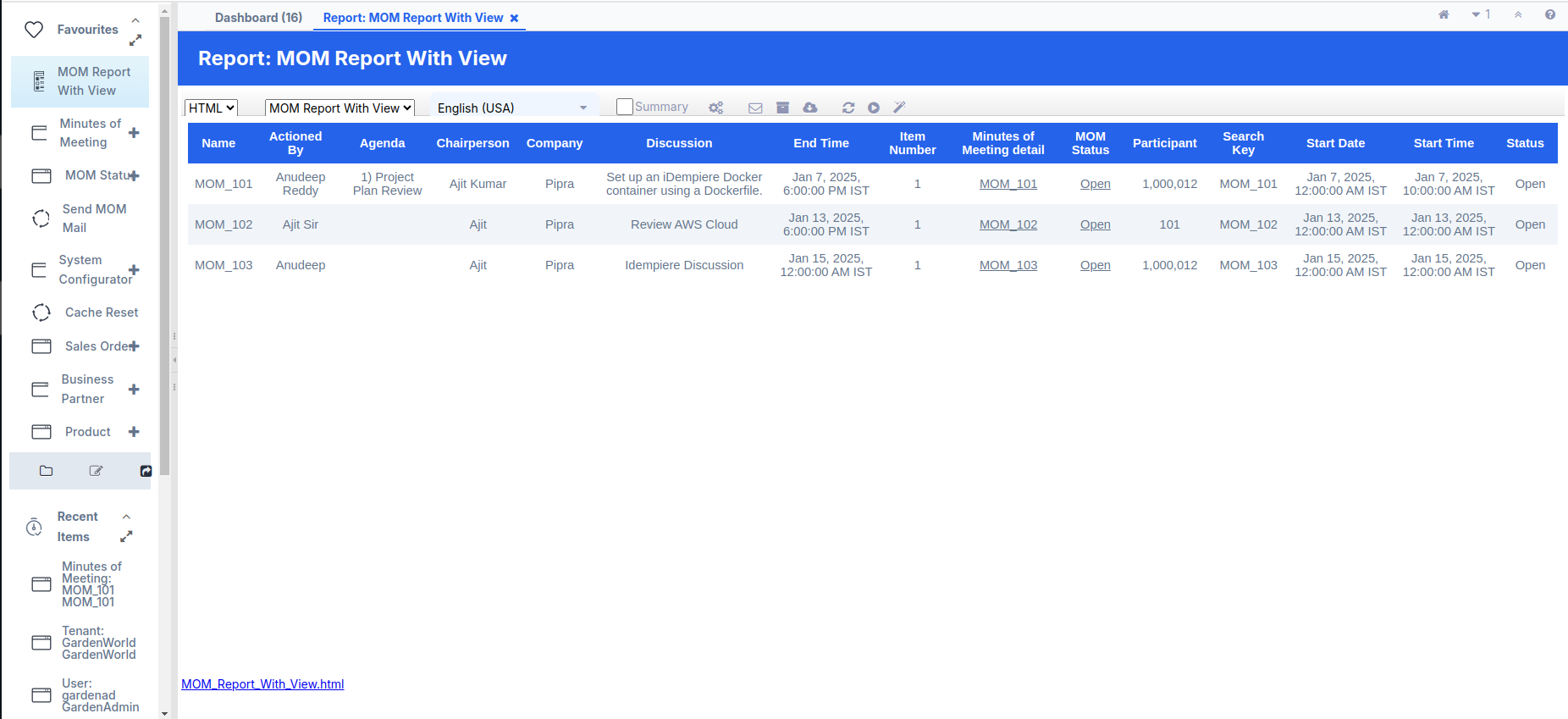
# 

# **How to do it…**

1. Create the following database view:  
   CREATE VIEW adempiere.c\_mom\_v AS  
   SELECT   
   a.\*,  
   b.item\_nbr,b.discussion\_desc,b.actionedby AS actioned\_by,b.c\_mom\_status\_id,  
   c.ad\_user\_id AS participant,c.company,  
   d.name AS status  
   FROM adempiere.c\_mom a  
   JOIN adempiere.c\_mom\_discussionline b ON a.c\_mom\_id=b.c\_mom\_id  
   JOIN adempiere.c\_mom\_participantsline c ON c.c\_mom\_id=b.c\_mom\_id  
   JOIN adempiere.c\_mom\_status d ON b.c\_mom\_status\_id=d.c\_mom\_status\_id;
2. Log in as **System**/**System** with the **System Administrator** role.
3. Open **Application Dictionary** | **Table** and then the **Column** window. Then create a new record for the newly created view— c\_mom\_v and check the View check box.  
     
   
4. Click on the **Create Columns from DB** button and verify that all the columns are created on the **Column** tab.  
     
   
5. Open the **Application Dictionary** | **Report View** window and create a new record using the newly created table in the dictionary—  
   MOM View and enter table field c\_mom\_v .  
   
6. Open the **Application Dictionary** | **Report & Process** window, create a report entry,and select MOM View, which was created in the previous step, as the **Report View**.

Name = MOM Report with View  
Entity Type = User maintained  
Data Access Level = Client+Organization  
Check the Report check box  
Report View = MOM View  
  


1. Verify the access to the report on the **Report Access** tab. By default, all the roles are given access. You may review the list and grant access to the roles according to your need.
2. Now, go to the **Menu** | **System Admin** | **General Rules** | **System Rules** | **Menu** window and create a new menu node for the report under the **MOM** menu  
     
   
3. Log out and log in as **GardenAdmin**/**GardenAdmin** with the **GardenWorld Admin** role. You shall see the newly added menu item under the **MOM** menu.  
     
   

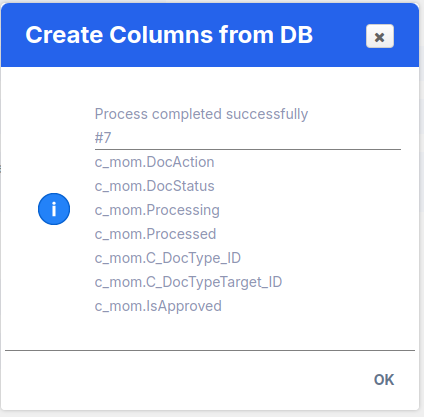
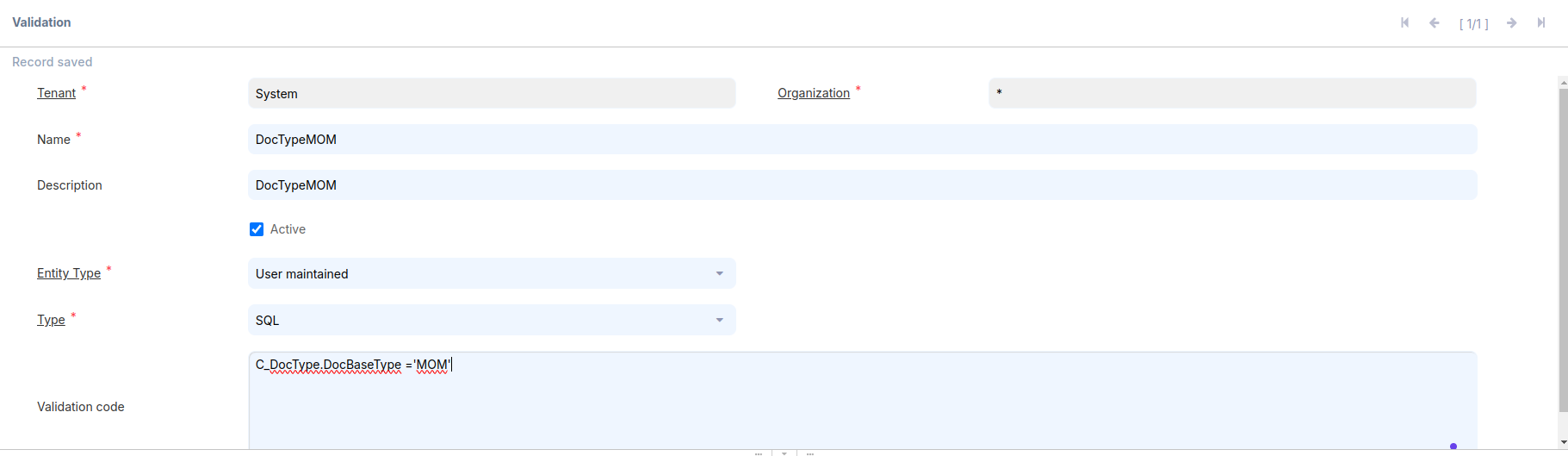
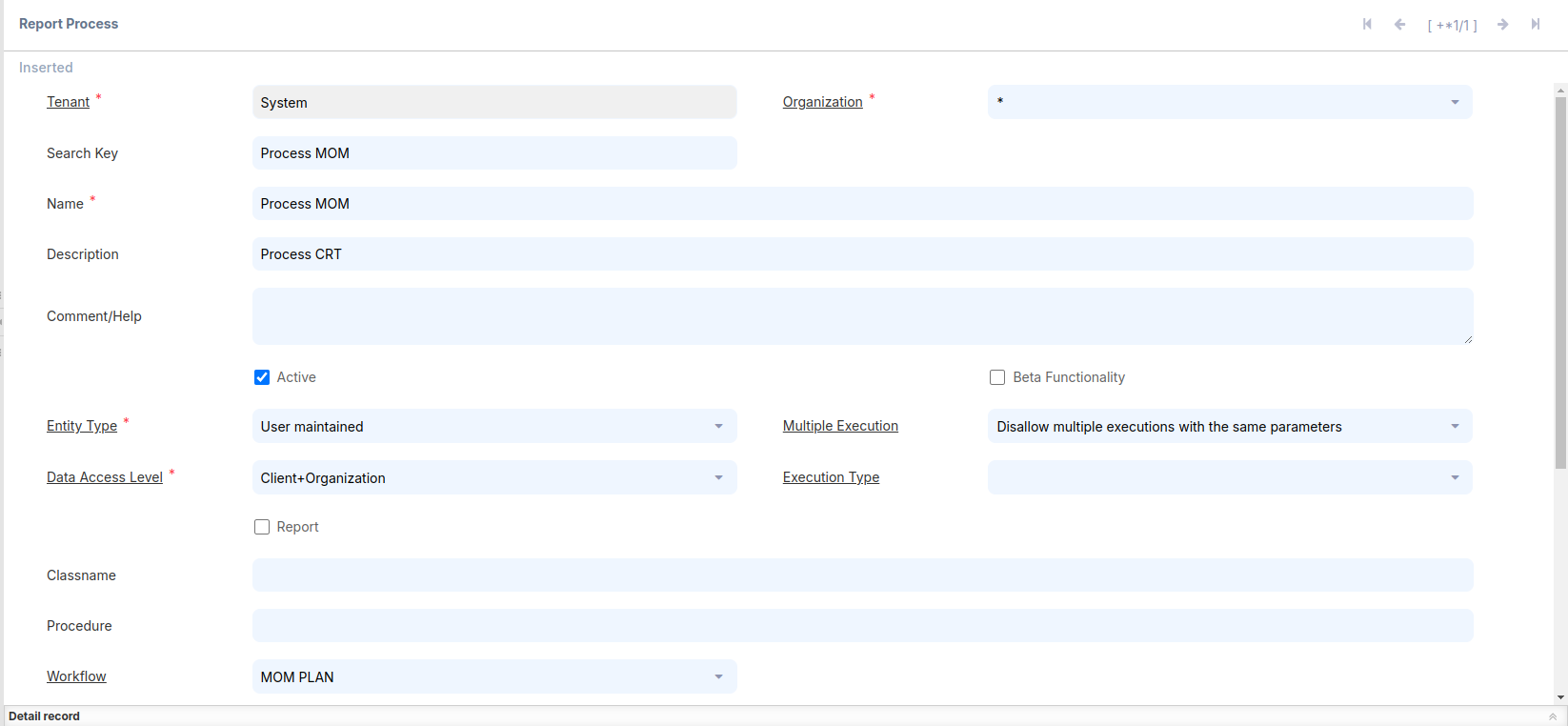
1. Click on the **MOM** **Report With View** menu and run the report. You shall get the following report output, which indicates that we are able to create a report from a database view.  
     
   

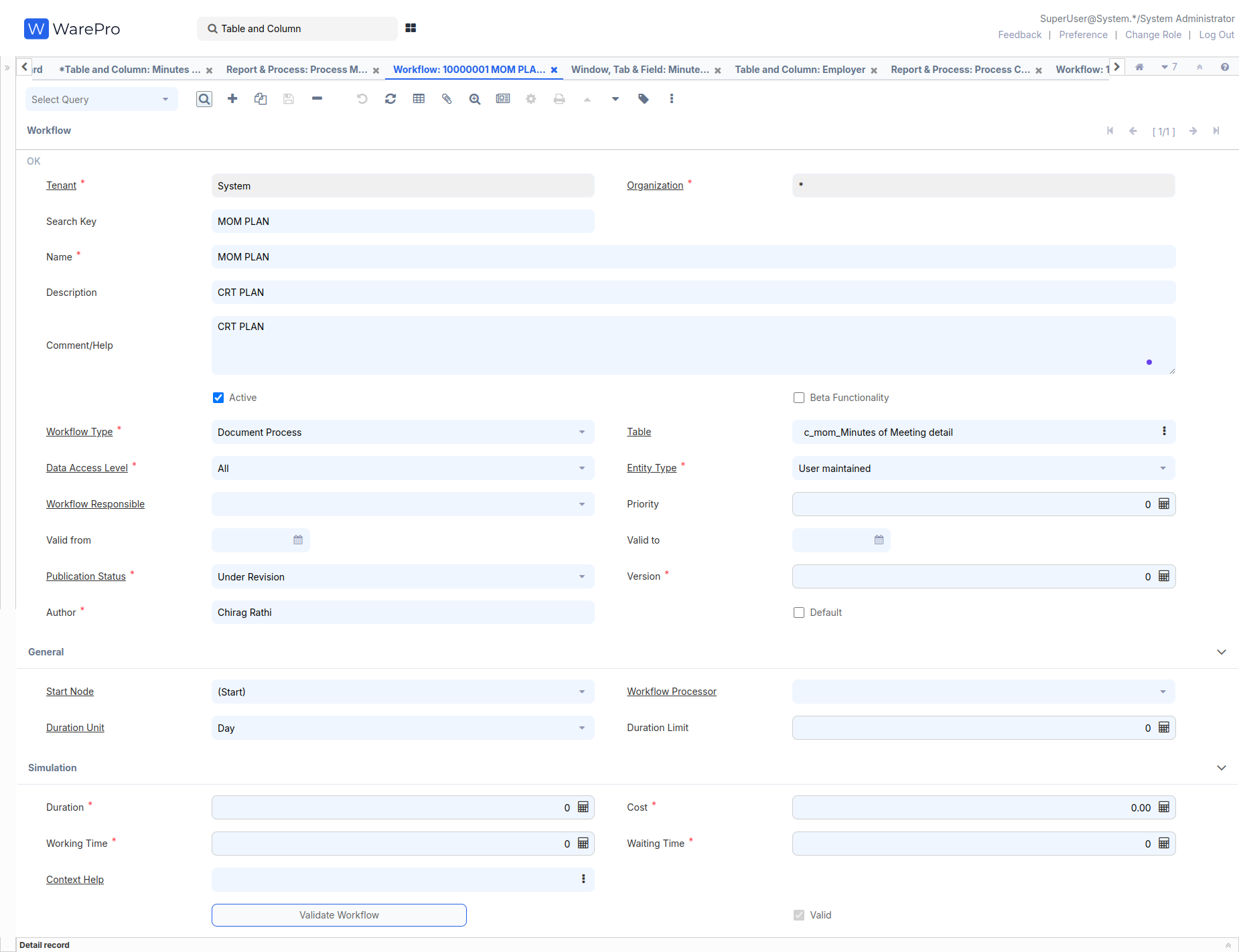
**Doc Action Setup**

Once the **Doc Action** is completed, the record cannot be edited.   
Additionally, you cannot create or edit any tabs associated with the record after the Doc Action.

# **How to do it…**

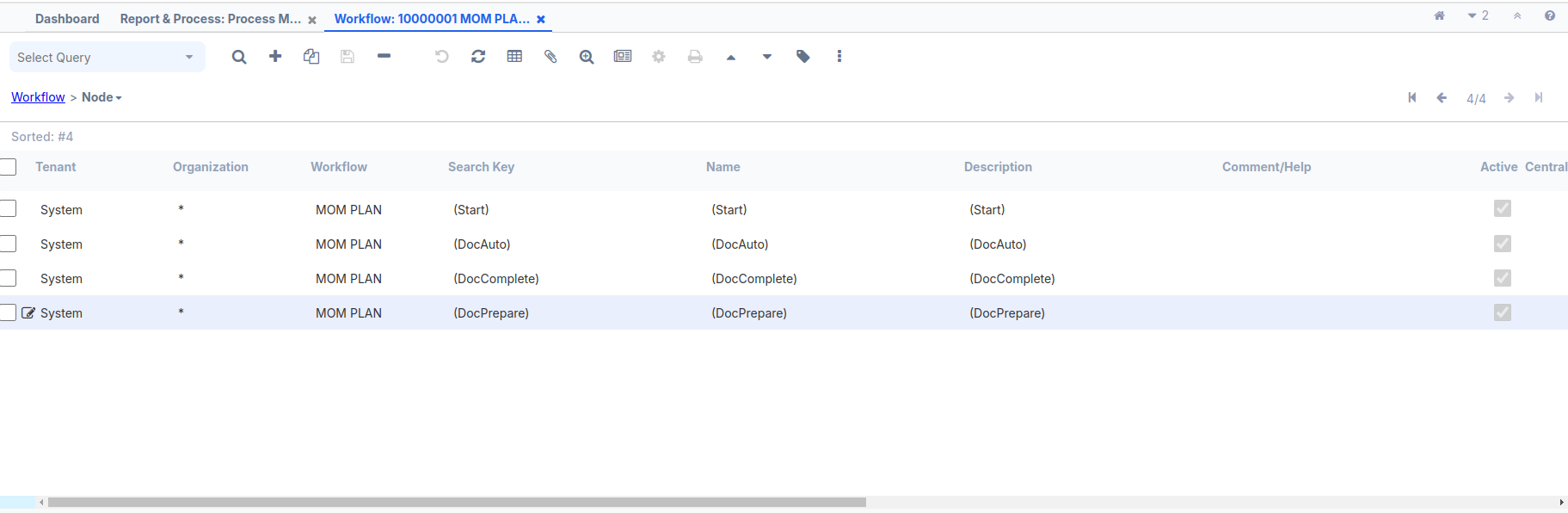
Please follow (**Create the following table in your Adempiere schema)** heading for create or alter table query,previously alreadyguided.

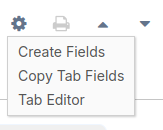
1. **First, update the table query for Doc Action setup:-  
     
   ALTER TABLE adempiere.c\_mom   
   ADD COLUMN docaction CHAR(2) NOT NULL DEFAULT 'CO'::bpchar,  
   ADD COLUMN DocStatus CHAR(2) NOT NULL DEFAULT 'DR'::bpchar,  
   ADD COLUMN processing CHAR(1) DEFAULT 'N'::bpchar,  
   ADD COLUMN processed character(1) NOT NULL DEFAULT 'N'::bpchar,  
   ADD COLUMN c\_doctype\_id numeric(10,0) NOT NULL DEFAULT 0,  
   ADD COLUMN c\_doctypetarget\_id numeric(10,0) NOT NULL DEFAULT 0,  
   ADD COLUMN isapproved character(1) NOT NULL DEFAULT 'Y'::bpchar;**
2. Log in as **System**/**System** with the **System Administrator** role.
3. Open **Application Dictionary** | **Table** and **Column** window. Locate and select the existing record for — c\_mom.
4. Click on the **Create Columns from DB** button and verify that all new columns are created on the **Column** tab  
     
   
5. Some **Column** fields have updated to our requirement :
6. **C\_DocType\_ID** : Default Logic = 0,Check the Updatable Check Box
7. **C\_DocTypeTarget\_ID** : Reference = Table,  
   Reference Key = C\_DocType,  
   Uncheck the Updatable Check Box,  
   Dynamic Validation = DocTypeMOM  
   Click on the **Dynamic Validation** field, create a new **Dynamic Validation**, and fill in the mandatory fields.  
   Name = DocTypeMOM  
   Validation Code = C\_DocType.DocBaseType ='MOM'  
   (MOM This value using our validation and note this key required 3 character)
8. **DocStatus :** Reference = List,  
   Reference Key = \_Document Status,  
   Default Logic = DR
9. **Processed** : Uncheck the Updatable Check Box
10. **Processing** : Reference = Button,  
    Process = Process MOM  
    (This Process created flow show in below)
11. **DocAction** : Reference = Button,  
    Reference Key = \_Document Action,  
    Default Logic = CO  
    Process = Process MOM(Create a new **Process** and select)  
    Create a **Process** and fill mandatory field  
    Search Key = Process MOM  
    Name = Process MOM  
    Entity Type = User maintained  
    Data Access Level = Client + Organization  
    Workflow = MOM Plan   
    (Below will be created)  
      
      
    
12. **Work Flow:-**Create a **Work Flow** and fill mandatory field  
    Search Key = MOM PLAN  
    Name = MOM PLAN  
    Workflow Type = Document Process  
    Table = c\_mom\_Minutes of Meeting Detail  
    Data Access Level = All  
    Author = Chirag Rathi (Write any name)



At the bottom left, go to the **Node** tab, enter it, and create four records:-

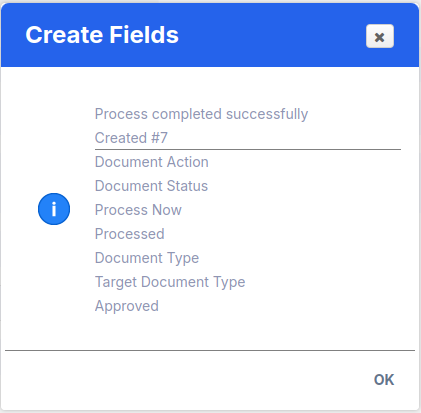
1. Name = (Start), Search Key = (Start), Entity Type - Dictionary, Action = Wait (Sleep)
2. Name = (DocPrepare), Search Key = (DocPrepare),   
   Entity Type - Dictionary, Action = Document Action,  
   Document Action = Prepare
3. Name = (DocComplete), Search Key = (DocComplete),  
   Entity Type - Dictionary, Action = Document Action,   
   Document Action = Complete
4. Name = (DocAuto), Search Key = (DocAuto),   
   Entity Type - Dictionary, Action = Document Action,   
   Document Action = <None>



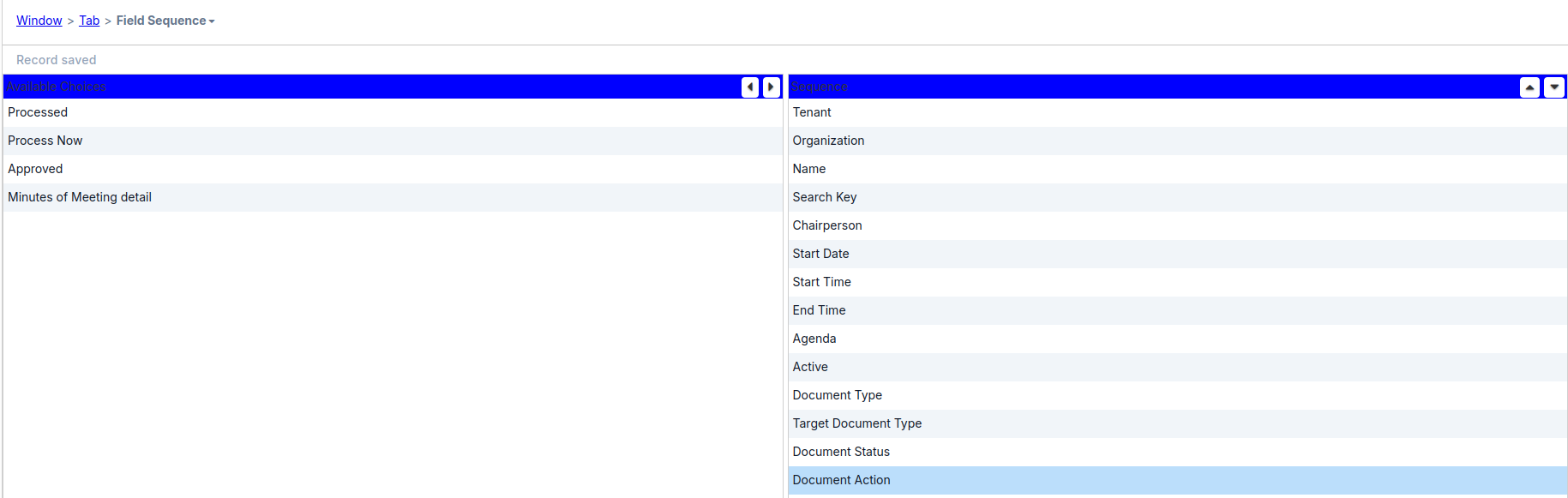
Go to the main **Workflow** screen, select the Start Node = (Start) and then click Save  
After creating the **Workflow**, select it on the **Process** screen and click Save.  
After creating the **Process**, select it on the **DocAction** Column screen 

Toolbar Button = Both and click Save.

After creating the Process then fill the **Processing** Column and click Save.  
**6.** Open **Application Dictionary** | **Window,Tab** and **Field** window.  
 Locate and select the existing record for — Minutes of Meeting.

**7.** Select the **MOM/Minutes of Meeting** tab and make the necessary edits.  
In the bottom-left corner, press **Ctrl + O** or  
 click the **Settings** icon.  
  
  
 

**8.** Review the **Field Sequence** tab and make the required changes to the sequence, as shown in the following screenshot:



**9.** Some **Field** value Updated:-

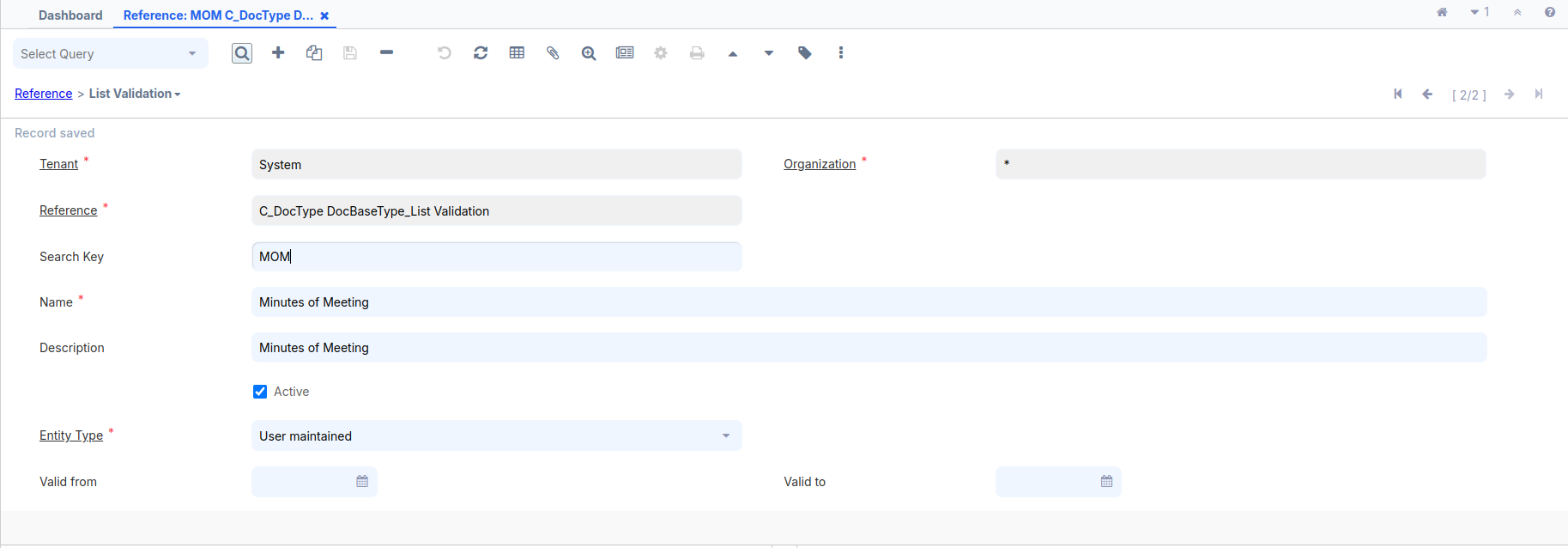
**Document Type :** Check the **Read Only** Check Box**,** X Position = 1

**Target Document Type :** X Position = 4

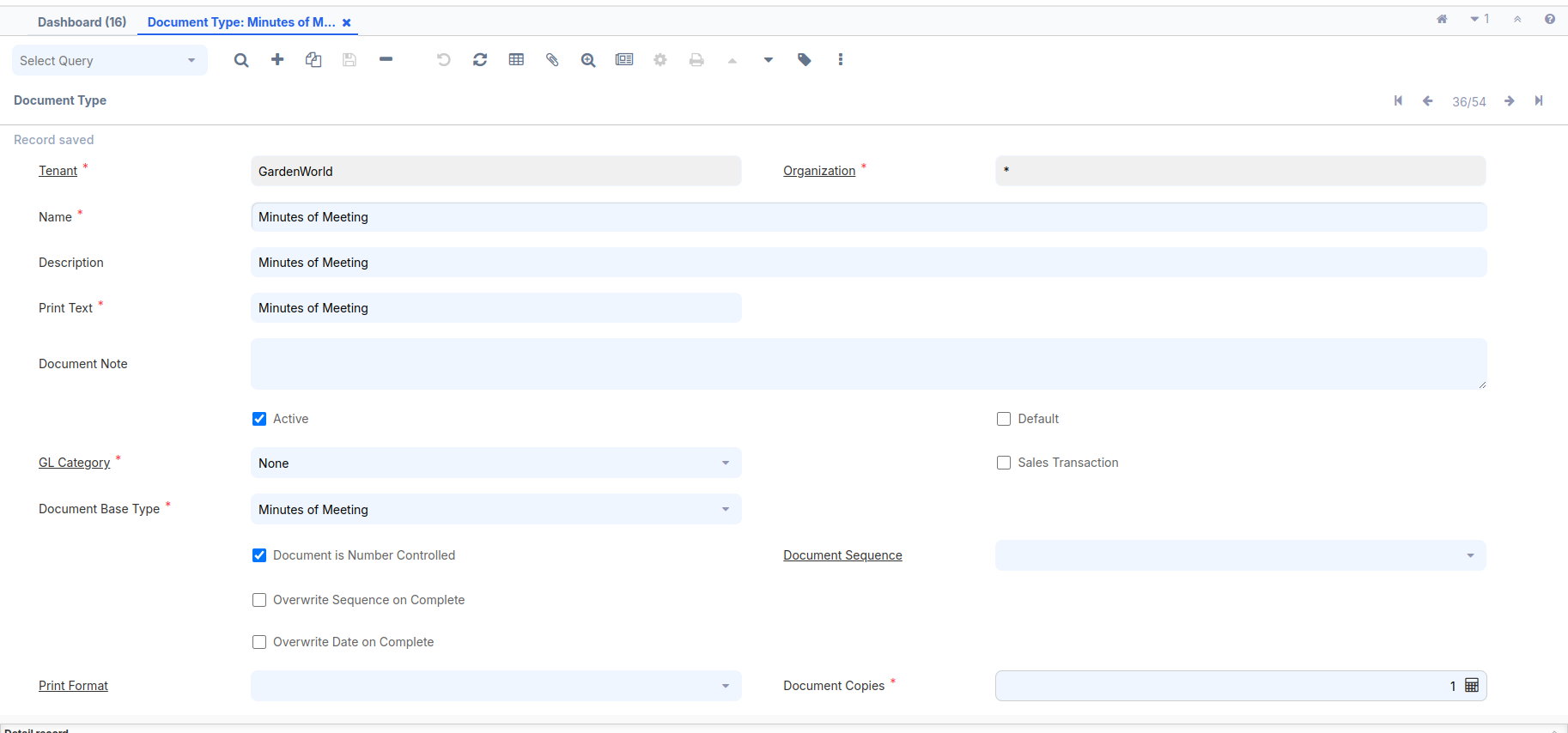
**Document Status :** Field Group = Status

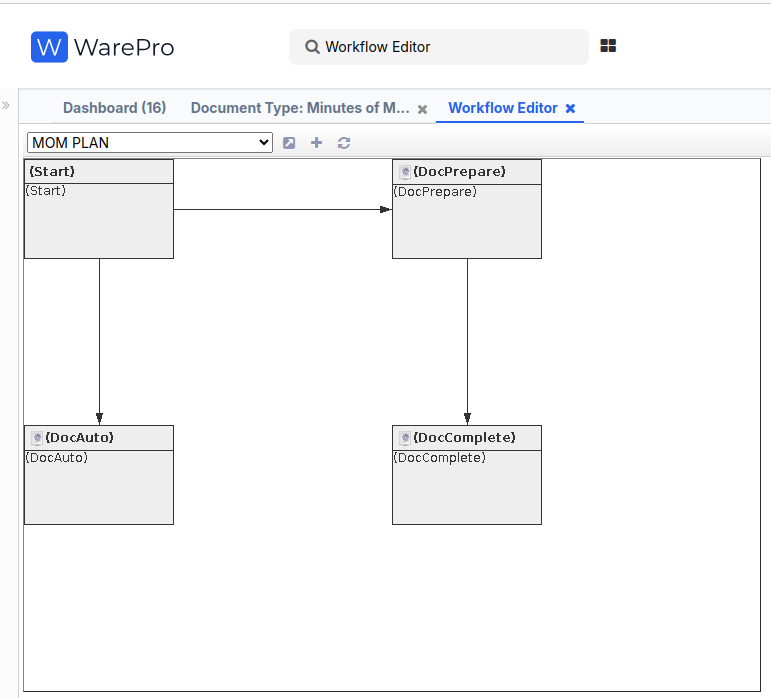
**Document Action :**  Field Group = Status, X Position = 5  
  
**10.** **Create A Custom Document :**-

Use the search bar at the top left, type **Reference**, then enter **C\_DocType DocBaseType** in the Name field and press EnterAt the bottom left, go to the **List Validation** tab, enter it, and create a new record:  
Search Key = MOM(This key corresponds to the dynamic validation inside **C\_DocTypeTarget\_ID**),   
Name = Minutes of Meeting and   
Entity Type - Dictionary   
click Save.

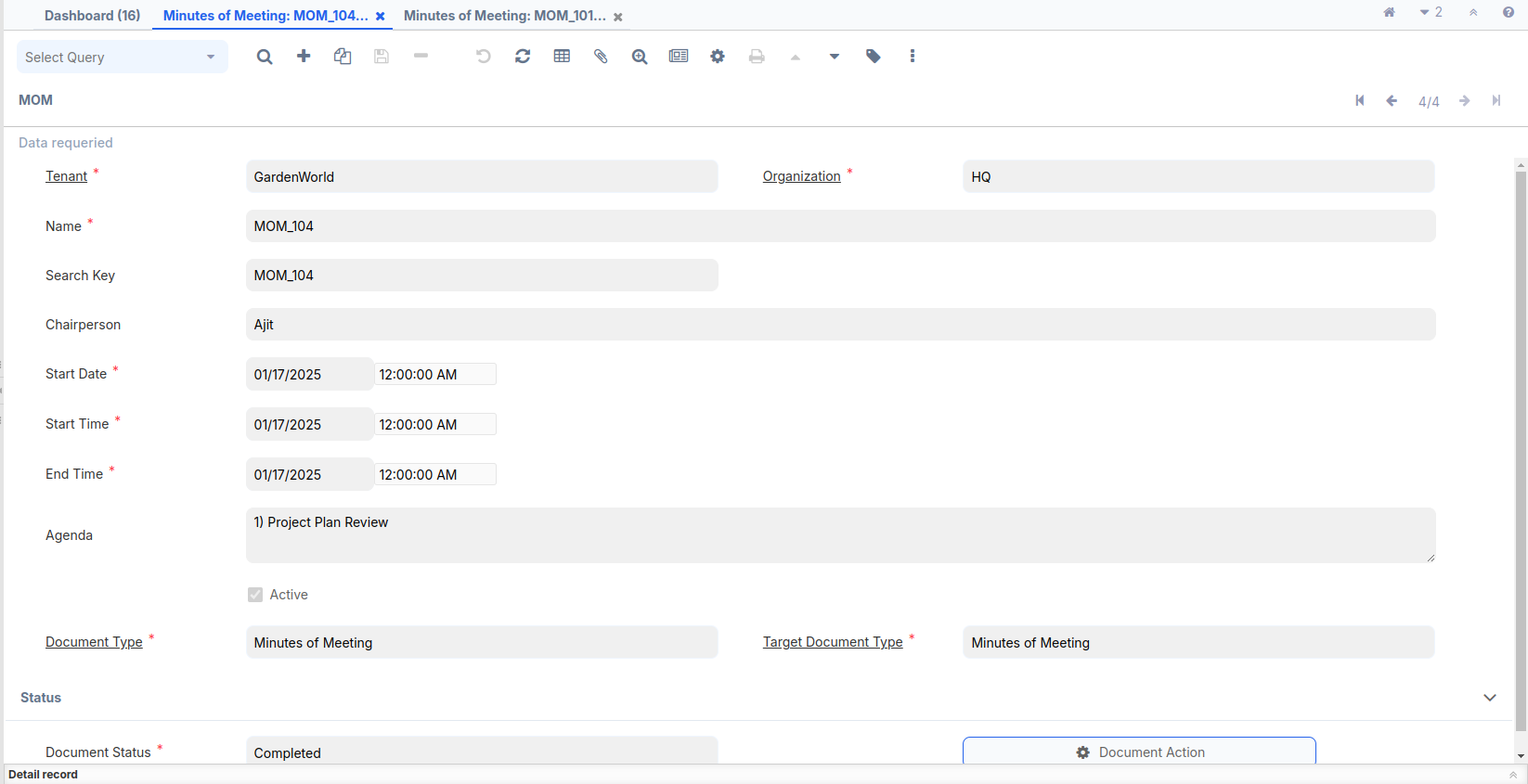


**11.** Log out and log in as **GardenAdmin**/**GardenAdmin** with the   
 **GardenWorld Admin** role.

**12.** Use the search bar at the top left and enter **Document Type** and create  
 a new document  
 Name = Minutes of Meeting  
 Print Text = Minutes of Meeting  
 GL Category = None  
 Document Base Type = Minutes of Meeting  
(Select the same record while logged in as System for creating it)click save.  


**13.** Use the search bar at the top left and enter **Workflow Editor**Select the Workflow name(Select the same record while logged in as System for creating it) from the top-left toolbar  


**14.** Create a new **Minute of Meeting** record and add a screenshot after the **Doc Action**.



**Java Code :-**

**DocumentFactory:-**

package org.mom.doc.factory;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.util.logging.Level;

import org.adempiere.base.IDocFactory;

import org.compiere.acct.Doc;

import org.compiere.model.MAcctSchema;

import org.compiere.model.MTable;

import org.compiere.util.CLogger;

import org.compiere.util.DB;

import org.compiere.util.Env;

import org.mom.model.DocMOM;

import org.mom.model.MOMModel;

import org.mom.model.Mmom;

public class DocumentFactory implements IDocFactory{

CLogger log = CLogger.getCLogger(DocumentFactory.class);

@Override

public Doc getDocument(MAcctSchema as, int AD\_Table\_ID, int Record\_ID, String trxName) {

String tableName = MTable.getTableName(Env.getCtx(), AD\_Table\_ID);

Doc doc = null;

StringBuffer sql = new StringBuffer("SELECT \* FROM ")

.append(tableName)

.append(" WHERE ").append(tableName).append("\_ID=? AND Processed='Y'");

PreparedStatement pstmt = null;

ResultSet rs = null;

try

{

pstmt = DB.prepareStatement (sql.toString(), trxName);

pstmt.setInt (1, Record\_ID);

rs = pstmt.executeQuery ();

if (rs.next ())

{

doc = getDocument(as, AD\_Table\_ID, rs, trxName);

}

else

log.severe("Not Found: " + tableName + "\_ID=" + Record\_ID);

}

catch (Exception e)

{

log.log (Level.SEVERE, sql.toString(), e);

}

finally

{

DB.close(rs, pstmt);

rs = null;

pstmt = null;

}

return doc;

}

@Override

public Doc getDocument(MAcctSchema as, int AD\_Table\_ID, ResultSet rs, String trxName) {

Doc doc = null;

String tableName = MTable.getTableName(Env.getCtx(), AD\_Table\_ID);

if(tableName.equals(MOMModel.Table\_Name)) {

return new DocMOM(as, getClass(), rs, tableName, trxName);

}

if (doc == null)

log.log(Level.SEVERE, "Unknown AD\_Table\_ID=" + AD\_Table\_ID);

return doc;

}

}

**ModelFactory :-**

package org.mom.doc.factory;

import java.sql.ResultSet;

import org.adempiere.base.IModelFactory;

import org.compiere.model.PO;

import org.mom.model.MOMModel;

public class ModelFactory implements IModelFactory{

@Override

public Class<?> getClass(String tableName) {

if (tableName.equalsIgnoreCase(MOMModel.Table\_Name))

return MOMModel.class;

return null;

}

@Override

public PO getPO(String tableName, int Record\_ID, String trxName) {

if(tableName.equalsIgnoreCase(MOMModel.Table\_Name))

return new MOMModel(null,Record\_ID,trxName);

return null;

}

@Override

public PO getPO(String tableName, ResultSet rs, String trxName) {

if(tableName.equalsIgnoreCase(MOMModel.Table\_Name))

return new MOMModel(null,rs,trxName);

return null;

}

}

**DocMom:-**

package org.mom.model;

import java.math.BigDecimal;

import java.sql.ResultSet;

import java.util.ArrayList;

import org.compiere.acct.Doc;

import org.compiere.acct.Fact;

import org.compiere.model.MAcctSchema;

public class DocMOM extends Doc{

public DocMOM(MAcctSchema as, Class<?> clazz, ResultSet rs, String defaultDocumentType,

String trxName) {

super(as, MOMModel.class, rs, null, trxName);

// TODO Auto-generated constructor stub

}

@Override

protected String loadDocumentDetails() {

// TODO Auto-generated method stub

return null;

}

@Override

public BigDecimal getBalance() {

// TODO Auto-generated method stub

return BigDecimal.ZERO;

}

@Override

public ArrayList<Fact> createFacts(MAcctSchema as) {

ArrayList<Fact> facts = new ArrayList<Fact>();

return facts;

}

}

**MOMModel** :-

package org.mom.model;

import java.io.File;

import java.math.BigDecimal;

import java.sql.ResultSet;

import java.util.Properties;

import org.compiere.process.DocAction;

import org.compiere.process.DocOptions;

import org.compiere.process.DocumentEngine;

public class MOMModel extends X\_c\_mom implements DocAction,DocOptions{

public MOMModel(Properties ctx, int c\_mom\_ID, String trxName) {

super(ctx, c\_mom\_ID, trxName);

// TODO Auto-generated constructor stub

}

public MOMModel(Properties ctx, ResultSet rs, String trxName) {

super(ctx, rs, trxName);

// TODO Auto-generated constructor stub

}

private static final long serialVersionUID = 1L;

@Override

public int customizeValidActions(String docStatus, Object processing, String orderType, String isSOTrx,

int AD\_Table\_ID, String[] docAction, String[] options, int index) {

if (options == null)

throw new IllegalArgumentException("Option array parameter is null");

if (docAction == null)

throw new IllegalArgumentException("Doc action array parameter is null");

if (docStatus.equals(DocumentEngine.STATUS\_Drafted) ||

docStatus.equals(DocumentEngine.STATUS\_Invalid)) {

options[index++] = DocumentEngine.ACTION\_Complete;

options[index++] = DocumentEngine.ACTION\_Prepare;

options[index++] = DocumentEngine.ACTION\_Void;

} else if (docStatus.equals(DocumentEngine.STATUS\_Completed)) {

options[index++] = DocumentEngine.ACTION\_Void;

options[index++] = DocumentEngine.ACTION\_ReActivate;

}

return index;

}

@Override

public void setDocStatus(String newStatus) {

// TODO Auto-generated method stub

}

@Override

public String getDocStatus() {

// TODO Auto-generated method stub

return null;

}

@Override

public boolean processIt(String action) throws Exception {

DocumentEngine engine = new DocumentEngine(this, getDocStatus());

return engine.processIt(action, getDocAction());

}

@Override

public boolean unlockIt() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean invalidateIt() {

// TODO Auto-generated method stub

return true;

}

@Override

public String prepareIt() {

setC\_DocType\_ID(getC\_DocTypeTarget\_ID());

return DocAction.STATUS\_InProgress;

}

@Override

public boolean approveIt() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean rejectIt() {

// TODO Auto-generated method stub

return true;

}

@Override

public String completeIt() {

setProcessed(true);

return DocAction.STATUS\_Completed;

}

@Override

public boolean voidIt() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean closeIt() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean reverseCorrectIt() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean reverseAccrualIt() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean reActivateIt() {

// TODO Auto-generated method stub

return true;

}

@Override

public String getSummary() {

// TODO Auto-generated method stub

return null;

}

@Override

public String getDocumentNo() {

// TODO Auto-generated method stub

return null;

}

@Override

public String getDocumentInfo() {

// TODO Auto-generated method stub

return null;

}

@Override

public File createPDF() {

// TODO Auto-generated method stub

return null;

}

@Override

public String getProcessMsg() {

// TODO Auto-generated method stub

return null;

}

@Override

public int getDoc\_User\_ID() {

// TODO Auto-generated method stub

return 0;

}

@Override

public int getC\_Currency\_ID() {

// TODO Auto-generated method stub

return 0;

}

@Override

public BigDecimal getApprovalAmt() {

// TODO Auto-generated method stub

return BigDecimal.ONE;

}

@Override

public String getDocAction() {

// TODO Auto-generated method stub

return null;

}

}

**Dashboard Widget**

In iDempiere, a **Dashboard Widget** is a component designed to provide real-time, customizable, and interactive insights into various aspects of business processes. These widgets enhance user productivity by offering quick access to critical information directly from the iDempiere dashboard.

# **How to do it…**

# Added **ui.zk** Plug-in using **OSGI** frame work:- A. Open your browser and go to:<https://localhost:8443/>. B. A new pop-up window will appear. On the top right, you will see three tools. Click on the **Settings** tool. C. A new page will open. On the top left, you’ll find a search bar. Enter **ui.zk** in the search bar. D. Click on the **Update** icon (symbol) next to it. E. Select your JAR file and proceed with the installation. F. After the installation is complete, search for the **ui.zk** plugin Again. G. Start the plugin, refresh the page, and the **OSGi** framework setup will be completed.

### Log in client side and search bar enter dashboard content and create a new record :- Fill name = MOM Gadget URI = /zul/mom.zul

**WorkFlow**

# **How to do it…**

## **Create a new Workflow window** SearchKey = Setup MOM Name = Setup MOM Workflow Type = Wizard Data Access Level = Client+Organization Entity Type = Dictionary Author = Chirag Rathi

## **At the bottom left, go to the Node tab, enter it, and create four records:-**

## Name = Add People, Search Key = Add People, Entity Type - Dictionary, Action = User Window, Window = User

1. Name = Status, Search Key = Status,   
   Entity Type - Dictionary, Action = Document Action,  
   Action = User Window, Window = MOM Status
2. Name = MOM, Search Key = MOM,  
   Entity Type - Dictionary, Action = Document Action,   
   Action = User Window, Window = Minutes of Meeting
3. Name = Report, Search Key = Report,   
   Entity Type - Dictionary, Action = Document Action,   
   Action = Apps Process, Process =MOM Report With View\_MOMReportWithView

## 3. Use the search bar at the top left and enter **Workflow Editor** Select the Workflow name(Select the same record while logged in as System for creating it) from the top-left toolbar

1. Log out and log in as **GardenAdmin**/**GardenAdmin** with the GardenWorld Admin role.
2. Use the search bar at the top left and enter **Role** to locate the **Role** window.
3. In the **Role** window, use the search tool to find the role with the name **GardenWorld Admin**.
4. Navigate to the **Workflow Access** tab at the middle-bottom section and click **New Record**.
5. In the **Workflow** field, select **Setup MOM**.
6. Check the **Read Write** checkbox and click **Save**.
7. After saving, log out of the system and log in again.
8. Open the menu tree and verify if the **Setup MOM** window is visible.