Microsoft

Power BI Developer in a Day

Lab 02D

Enforcing Row-Level Security

Overview

The estimated time to complete the lab is 30 minutes

*You must have completed **Lab 02C** before commencing this lab.*

In this lab, you will explore three different security scenarios by creating static roles, a dynamic role, and also by developing a data-driven security design.

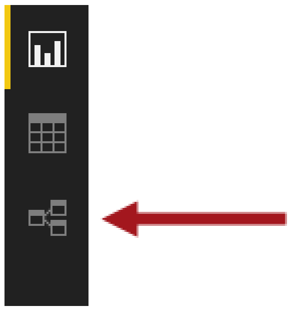
Creating Static Roles

In this exercise, you will first review the model design, and then create static roles to enforce row-level security for specific regions.

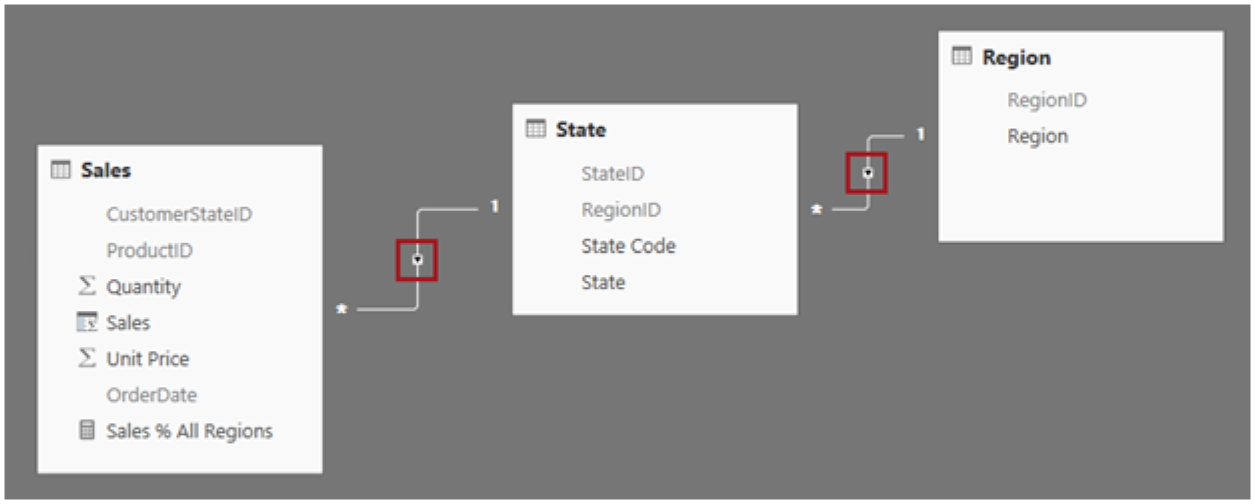
Reviewing the Model Design

In this task, you will review the model design, with a focus on the **Region** table.

1. In Power BI Desktop, at the top-left corner, switch to **Relationships** view.



2. Review the model tables and relationships, noticing that the **Region** table relates to the **State** table, and then to the **Sales** table, and also the direction of the filters.



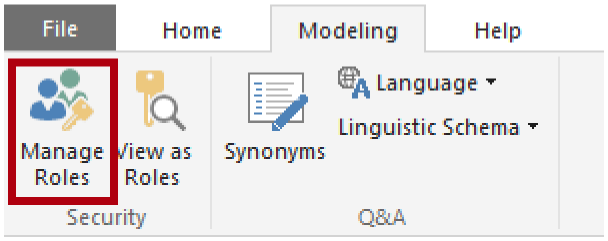
*The direction describes how filter propagation flows. In this model, when a filter is applied to the **Region** table (e.g. **New England**), then the **State** table will filter to the states related to the **New England** region, and then the **Sales** table will filter to sales transactions recorded for the states of **New England**.*

Note that at the Tailspin Toys company, six regions are defined to geographically categorize the 51 US states (including Washington DC).

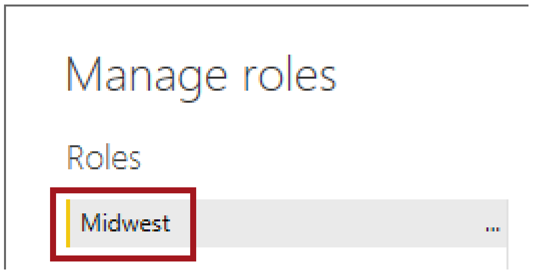
Creating Static Roles

In this task, you will create two static roles to filter by specific regions.

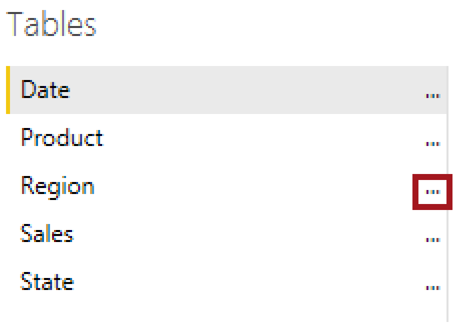
1. On the **Modeling** ribbon, from inside the **Security** group, select **Manage Roles**.



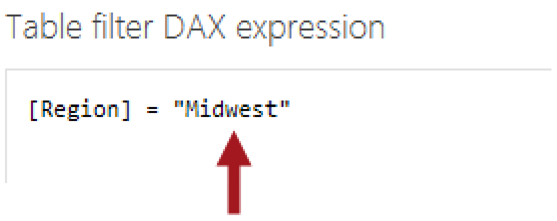
2. In the **Manage Roles** window, click **Create**.
3. In the box, replace the selected text with **Midwest**, and then press **Enter**.



4. In the **Tables** list, to the right of the **Region** table, click the ellipsis.

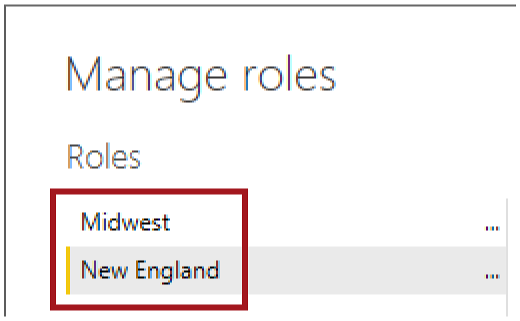


5. In the context menu, select **Add Filter | [Region]**.
6. In the **Table Filter DAX Expression** area, replace **Value** with **Midwest**.

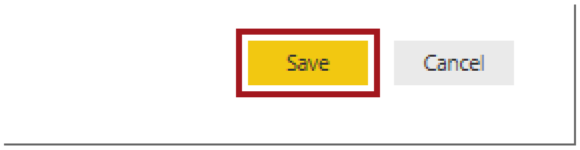


```
[Region] = "Midwest"
```

7. To create a second role, click **Create**.
8. Repeat the steps in this task to create a role named **New England**, which filters by the value **New England**.
9. Verify that two roles have been created.



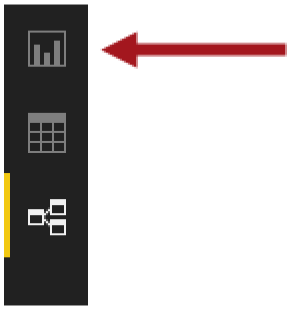
10. Click **Save**.



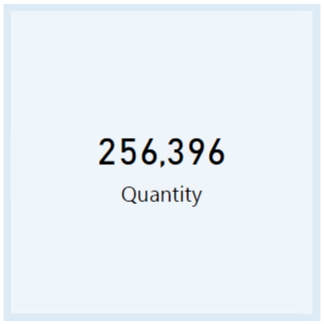
Testing the Roles

In this task, you will test the two roles created in the previous task.

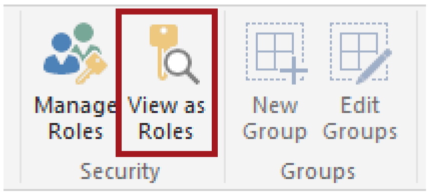
1. Switch to **Report** view.



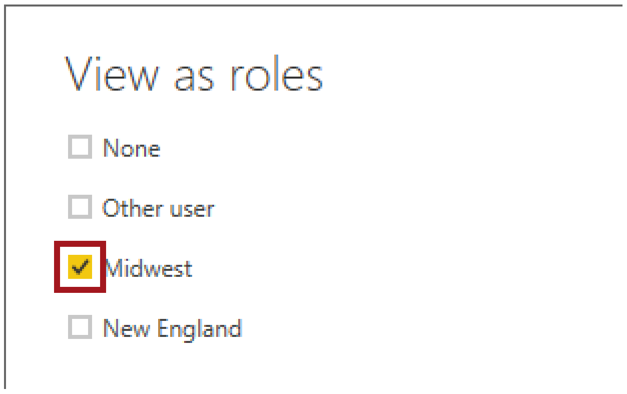
2. Ensure that the **Regional Sales Analysis** page is selected.
3. Note that the total quantity of sales for **FY2017** is **256,396**.



4. On the **Modeling** ribbon, from inside the **Security** group, select **View As Roles**.

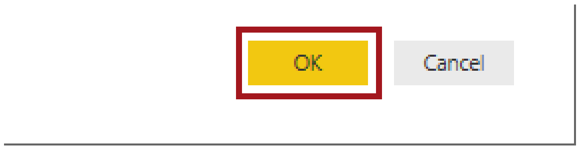


5. In the **View As Roles** window, check the **Midwest** checkbox.

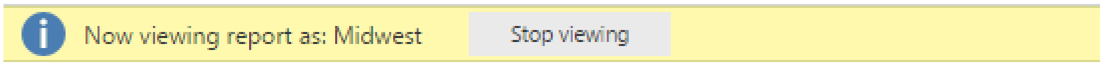

View as roles

- [ ] None
- [ ] Other user
- [✓] Midwest
- [ ] New England

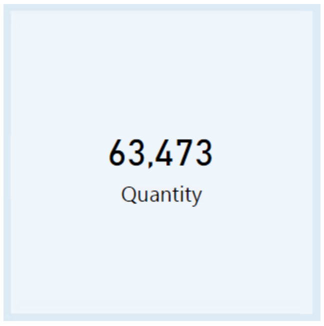
6. Click **OK**.



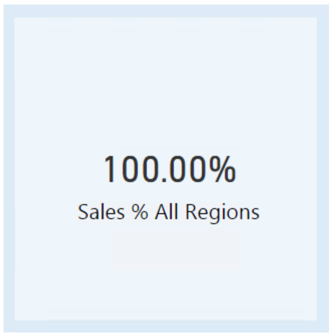
7. Across the top of the report, notice the yellow banner describing the enforced security role.


Now viewing report as: Midwest

8. Note that the total quantity of sales for **FY2017** for **Midwest** is **63,473**.

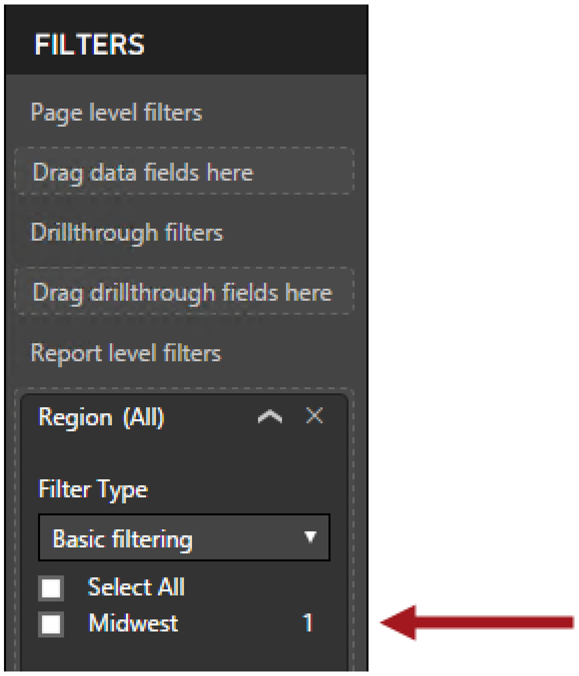


9. Note that the **Sales % All Regions** value is **100%.**



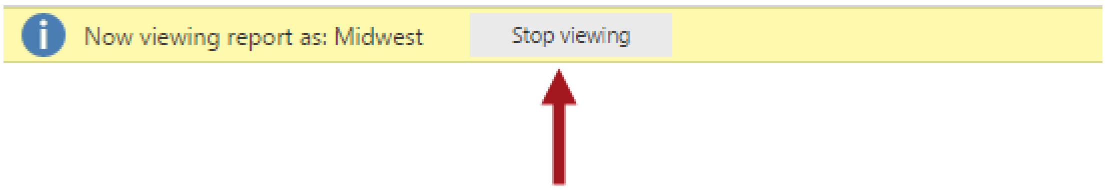
This card is based on a measure. The measure formula divides current in-context regional sales by all regional sales. However, it's important to understand that the enforced row-level security will only ever consider rows for ***Midwest****, and so "all regional sales" are those for only that region. Therefore, the card will always display 100%.*

10. In the **Filters** pane, expand the **Region** filter, and verify that **Midwest** is the only available option.



To the report user, the model will never deliver data unrelated to the ***Midwest*** *region, its states and their sales.*

11. To remove the security context, in the yellow banner, click **Stop Viewing**.

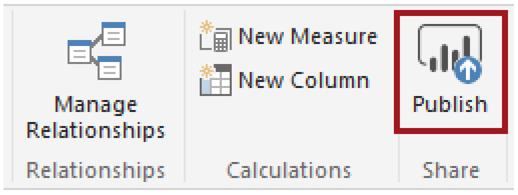


12. Test the **New England** role also.

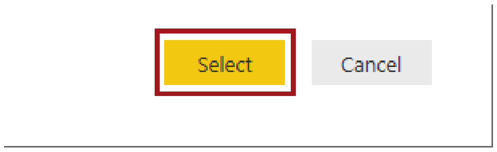
Publishing the Report

In this task, you will publish the Power BI Desktop report.

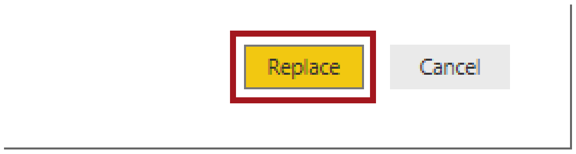
1. On the **File** menu, select **Save**.
2. On the **Home** ribbon, from inside the **Share** group, click **Publish**.



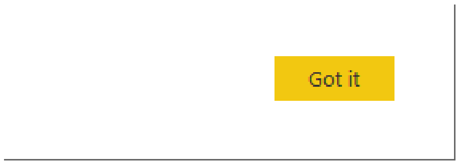
3. If prompted to sign in, use the Power BI user credentials from the **MySettings.txt** file.
4. In the **Publish to Power BI** window, select the app workspace you created in **Lab 02A**.
5. Click **Select**.



6. When prompted to replace the dataset, click **Replace**.



7. When the Power BI Desktop report has been published, click **Got It**.



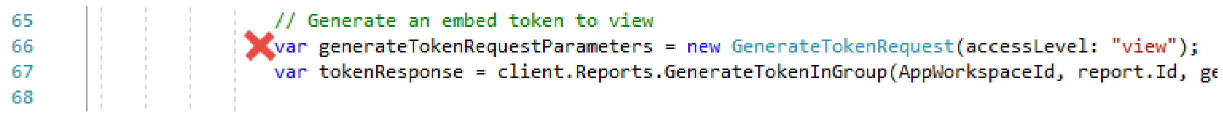
Securing the Embedded Report

In this task, you will modify the **EmbedReport** page to enforce permissions based on the two static roles.

1. In Visual Studio, open the code-behind file for the **EmbedReport.aspx** item.

2. In the **Page_Load** method, replace the **generateTokenRequestParameters** variable declaration with the following:

   For convenience, all remaining code in this lab can be copied from the ***<CourseFolder>\PowerBIDevIAD\Lab02D\Assets\Snippets.txt*** *file.*



```
// Generate an embed token to view
var generateTokenRequestParameters = new GenerateTokenRequest(accessLevel: "view");
var tokenResponse = client.Reports.GenerateTokenInGroup(AppWorkspaceId, report.Id, ge
```

Visual C#

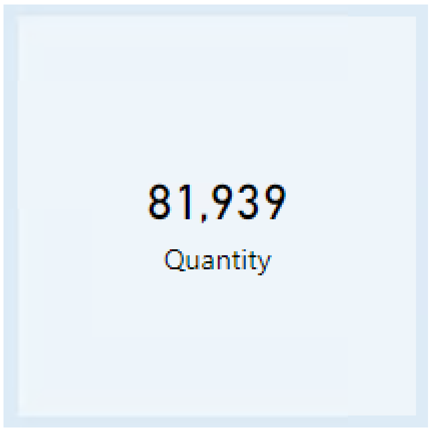
```
var generateTokenRequestParameters =
     new GenerateTokenRequest("view", identities: new List<EffectiveIdentity> {
         new EffectiveIdentity(
          username: "UserName",
          roles: new List<string> { "Midwest", "New England" },
          datasets: new List<string> { report.DatasetId })
});
```

   This variable declaration adds an ***EffectiveIdentity*** *collection (consisting of a single item) to the* ***GenerateTokenRequest*** *constructor. Note that* ***username*** *is passed the text value* ***UserName****, and the identity is associated with the two roles created earlier in this exercise. Note that* ***username*** *must pass a value (and not an empty string), though it is not used by the roles to enforce permissions (this will happen in the next exercise).*

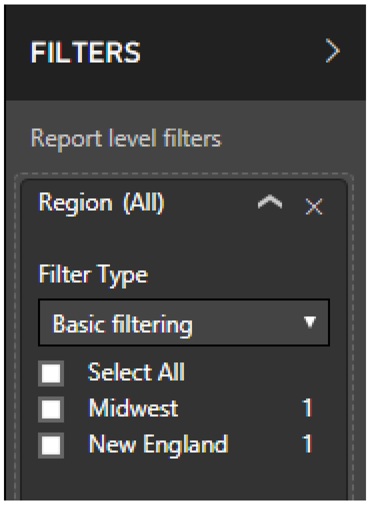
3. Start the web application, and then navigate to the **Embed Reports** page.

   Note that the Q&A experience will no longer work when roles are defined.

4. Verify that the total quantity of sales for **FY2017** for the two roles is **81,939**.



5. Open the **Filters** pane, expand the **Region** filter, and verify that the **Midwest** and **New England** regions are available.



FILTERS

Report level filters

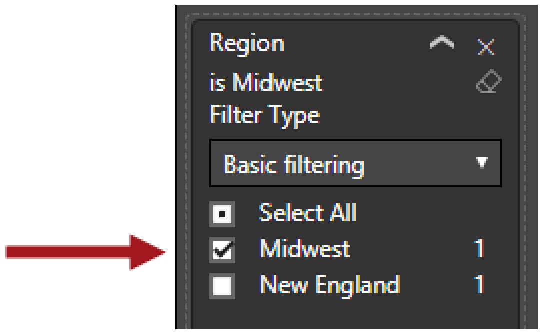
Region (All)

Filter Type

Basic filtering

- [ ] Select All
- [ ] Midwest 1
- [ ] New England 1

6. Filter by **Midwest** only.



Region

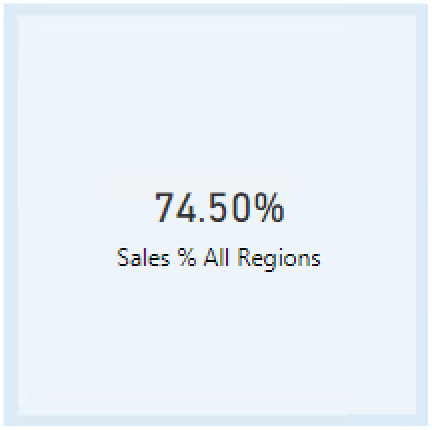
is Midwest

Filter Type

Basic filtering

- [ ] Select All
- [x] Midwest 1
- [ ] New England 1

7. Verify that the **Sales % All Regions** is **74.50%**.



74.50%

Sales % All Regions

*This value represents the value **Midwest** sales divided by the combined value of **Midwest** and **New England** sales.*

8. Close the web session, and then stop debugging.

Removing the Static Roles

In this task, you will remove the two static roles.

1. In Power BI Desktop, on the **Modeling** ribbon, from inside the **Security** group, select **Manage Roles**.
2. To delete the first selected role, click **Delete**.
3. When prompted to delete the role, click **Yes, Delete**.
4. Delete the **New England** role also.
5. Click **Save**.

   As achieved in this exercise, the developer can generate an embed token for multiple roles.

   While creating static roles is easy, it can quickly become difficult to maintain when many roles must be created, or when new regions are added. In the next exercise, you will create a single dynamic role which allows filtering by a value passed by the developer when generating an embed token.

Creating a Dynamic Role

In this task, you will add a dynamic range to filter the **Region** table.

1. In Power BI Desktop, add a role named **Region**.
2. Set the filter on the **Region** table, **Region** column:

   For convenience, the expression can be copied from the ***<CourseFolder>\PowerBIDevIAD\Lab02D\Assets\Snippets.txt*** *file.*

   The ***CUSTOMDATA*** *function is not presently supported in Power BI Desktop.*

   DAX

   ```
   [Region] = SUBSTITUTE(USERNAME(), "~", " ")
   ```

   This expression passes the app user's region via the ***username*** *parameter, which can be retrieved by the* ***USERNAME*** *function in a DAX expression.*

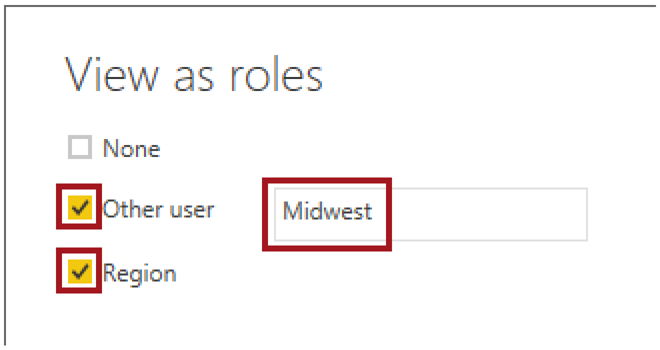
   Therefore, this expression will filter the ***Region*** *column by the result of the* ***USERNAME*** *function, substituting tilde (~) characters for spaces.*

   Note that when programmatically passing a username value, the property does not allow embedded spaces—it expects an actual User Principal Name (UPN). As some region names contain a space (like ***New England****), any spaces will be first replaced with tildes. Therefore, the DAX expression must revert these characters back to spaces to perform the correct filtering.*

Testing the Role

In this task, you will test the dynamic role.

1. To test the role, check **Other User**, and then in the corresponding box, enter **Midwest**.
2. Check the **Region** role.



View as roles

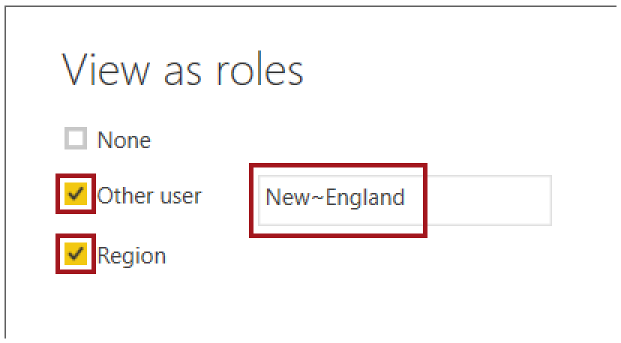
☐ None

☑ Other user   Midwest

☑ Region

3. Click **OK**.

4. In the **Filters** pane, verify that only the **Midwest** region is an available option.
5. Test the role for **New~England** also.



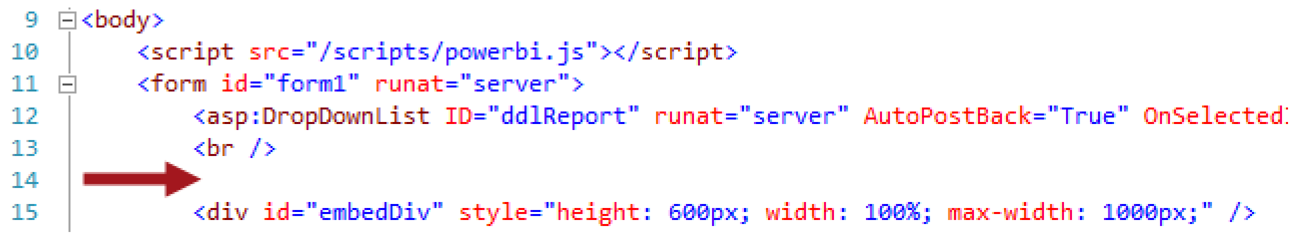
6. Stop viewing the report as the **Region** role.
7. Save, and then publish (replace) the report to the app workspace.

Securing the Embedded Report

In this task, you will modify the **EmbedReport** page to enforce permissions based on a specific region.

1. In Visual Studio, open the **EmbedReport.aspx** item.
2. To add a dropdown list, beneath the **<br />** element after the **asp:DropDownList** element, add the following HTML elements:

   *For convenience, the elements can be copied from the **<CourseFolder>\PowerBIDevIAD\Lab02D\Assets\Snippets.txt** file.*



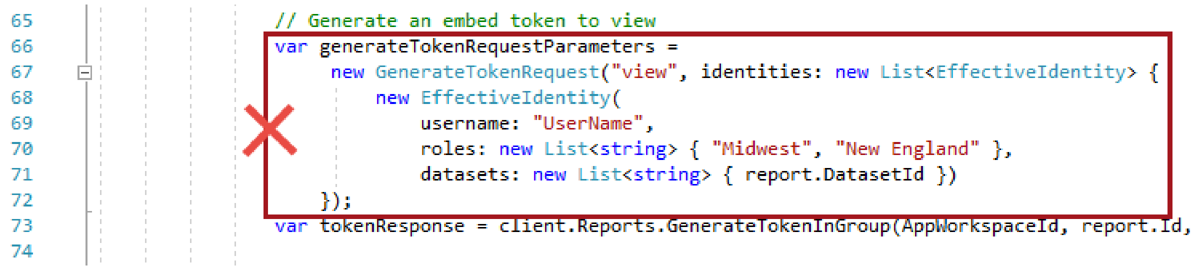
```
<body>
    <script src="/scripts/powerbi.js"></script>
    <form id="form1" runat="server">
        <asp:DropDownList ID="ddlReport" runat="server" AutoPostBack="True" OnSelected
        <br />

        <div id="embedDiv" style="height: 600px; width: 100%; max-width: 1000px;" />
```

HTML

```
<span>View as Region: </span>
<asp:DropDownList ID="ddlRegion" runat="server" AutoPostBack="True" OnSelectedIndexChanged="Page_Load">
      <asp:ListItem Text="Midwest" />
      <asp:ListItem Text="New England" />
      <asp:ListItem Text="Northeast" />
      <asp:ListItem Text="Pacific Northwest" />
      <asp:ListItem Text="Southern" />
      <asp:ListItem Text="Southwest" />
</asp:DropDownList>
<br />
```

This dropdown list is for testing purposes only. Typically, the developer will determine the appropriate region from the profile of the authenticated user.

3. In the code-behind file for the **EmbedReport.aspx** item, in the **Page_Load** method, replace the **generateTokenRequestParameters** variable declaration with the following:



```
// Generate an embed token to view
var generateTokenRequestParameters =
    new GenerateTokenRequest("view", identities: new List<EffectiveIdentity> {
        new EffectiveIdentity(
            username: "UserName",
            roles: new List<string> { "Midwest", "New England" },
            datasets: new List<string> { report.DatasetId })
    });
var tokenResponse = client.Reports.GenerateTokenInGroup(AppWorkspaceId, report.Id,
```

Visual C#

```
var generateTokenRequestParameters =
      new GenerateTokenRequest("view", identities: new List<EffectiveIdentity> {
           new EffectiveIdentity(
            username: ddlRegion.SelectedValue.Replace(' ', '~'),
            roles: new List<string> { "Region" },
            datasets: new List<string> { report.DatasetId })
});
```

*This differences between this and the previous declaration code are that the **username** parameter is passed the selected region (with spaces replaced with tildes), and the **role** parameter is set to only the **Region** role.*

4. Start the web application, and then navigate to the **Embed Reports** page.

5. Select different regions, and verify that the report presents only data for the selected region.

6. Close the web session, and then stop debugging.

Removing the Dynamic Role

In this task, you will remove the dynamic role.

1. In Power BI Desktop, remove the **Region** role.

   Creating a single dynamic role which allows filtering by the value passed in as the user name is a valid technique. It can simplify the security design by minimizing the number of roles to create. However, it will only support filtering by a single value, rather than multiple values.

   In the next task, you will develop a data-driven approach. You will modify the model design to enable filtering by the user's association to one or more regions, which are defined in a table of the model.

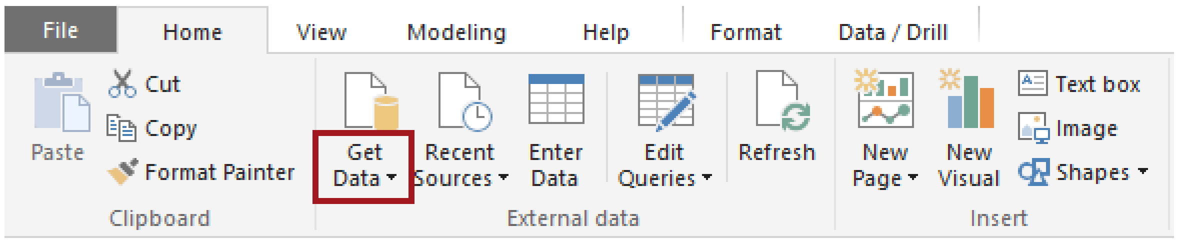
Developing a Data-Driven Security Design

In this exercise, you will develop a data-driven design by adding a **Manager** table to the model, which will enable a many-to-many relationship between managers and regions.

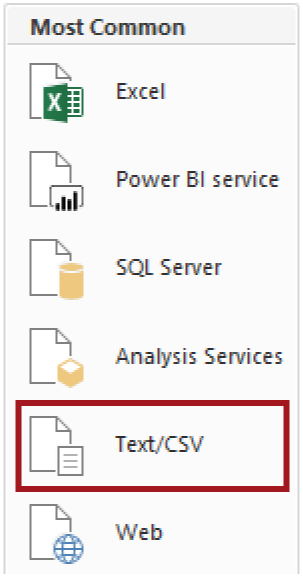
Adding the Manager Table

In this task, you will add the **Manager** table sourced from a CSV file.

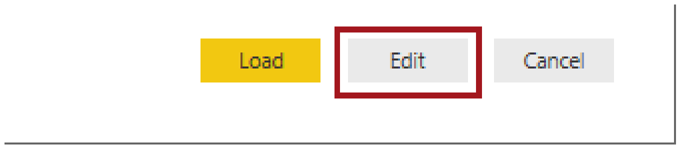
1. In Power BI Desktop, on the **Home** ribbon, from inside the **External Data** group, open the **Get Data** dropdown list.



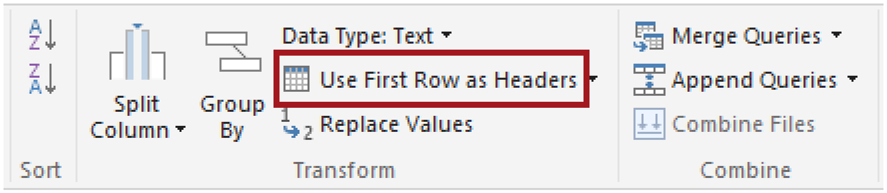
2. In the list, select **Text/CSV**.



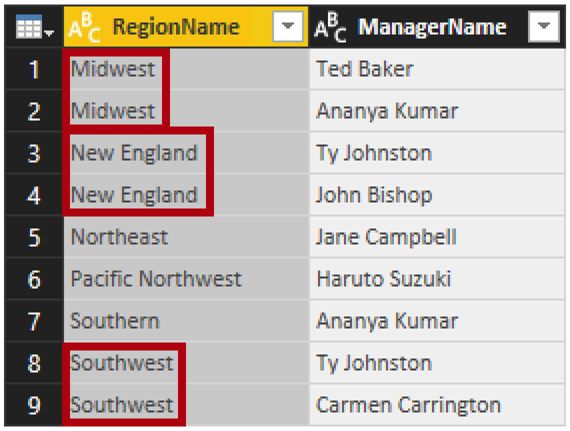
3. In the **Open** window, navigate to the **<CourseFolder>\PowerBIDevIAD\Lab02D\Assets** folder.
4. Select the **Manager.csv** file.
5. Click **Open**.
6. In the **Manager.csv** window, click **Edit**.



7. In the **Query Editor** window, on the **Home** ribbon, from inside the **Transform** group, click **Use First Row As Headers**.

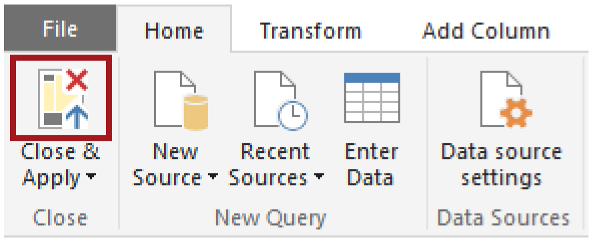


8. Review the query result, noticing that some regions have multiple managers, and some managers have multiple regions.

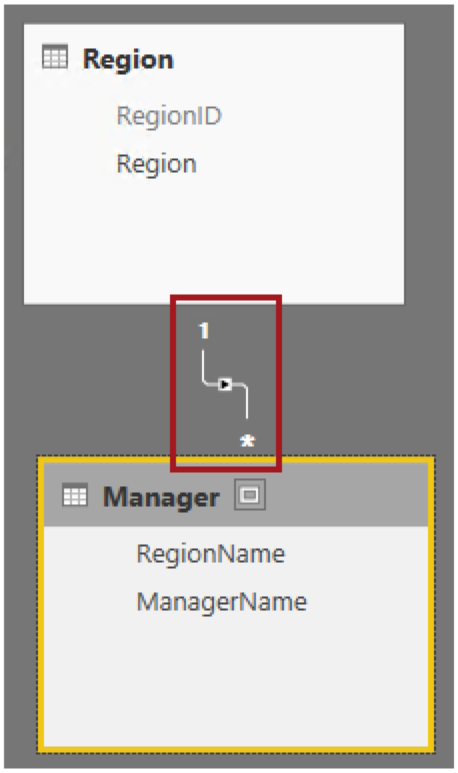


	RegionName	ManagerName
1	Midwest	Ted Baker
2	Midwest	Ananya Kumar
3	New England	Ty Johnston
4	New England	John Bishop
5	Northeast	Jane Campbell
6	Pacific Northwest	Haruto Suzuki
7	Southern	Ananya Kumar
8	Southwest	Ty Johnston
9	Southwest	Carmen Carrington

9. To load the query to a model table, on the **Home** ribbon, click **Close & Apply**.



10. In **Relationships** view, notice that the table has been added, and also a relationship connecting the new table to the **Region** table.

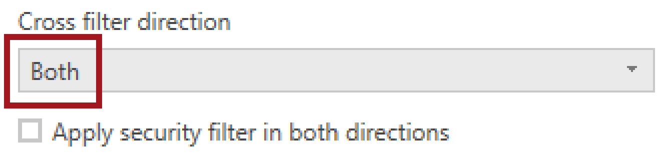


11. Notice that the relationship filters from the **Region** table to the **Manager** table, but not in the opposite direction.

    When the security role enforces permissions for a manager, the filter must propagate to ***Manager >Region > State > Sales****.*

12. To configure the relationship properties, double-click the relationship between the **Region** and the **Manager** tables.

13. In the **Edit Relationship** window, in the **Cross Filter Direction** dropdown list, select **Both**.

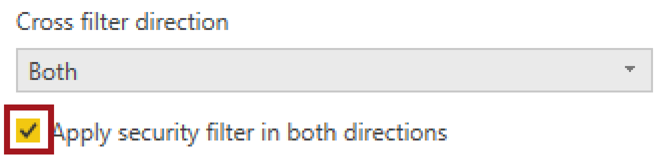


Cross filter direction

Both

Apply security filter in both directions

14. To enforce filter propagation when used in a security role, check the **Apply Security Filter in Both Directions** checkbox.

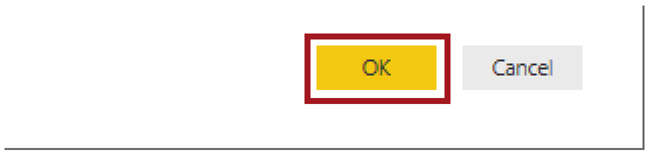


Cross filter direction

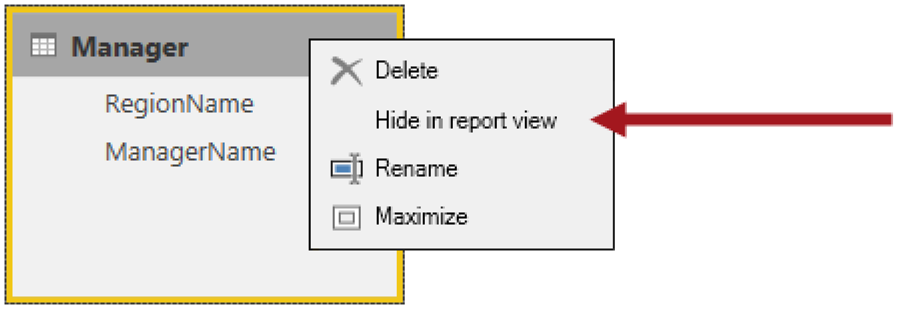
Both

✓ Apply security filter in both directions

15. Click **OK**.



16. To hide the table, right-click the **Manager** table header, and then select **Hide in Report View**.



This table was added for security design purposes, and should not be made visible to report users (including Q&A).

Creating a Dynamic Role

In this task, you will add a dynamic range to filter the **Manager** table.

1. Add a role named **Manager**.
2. Set the filter on the **Manager** table, **ManagerName** column:

   For convenience, the expression can be copied from the ***<CourseFolder>\PowerBIDevIAD\Lab02D\Assets\Snippets.txt*** *file.*

   DAX

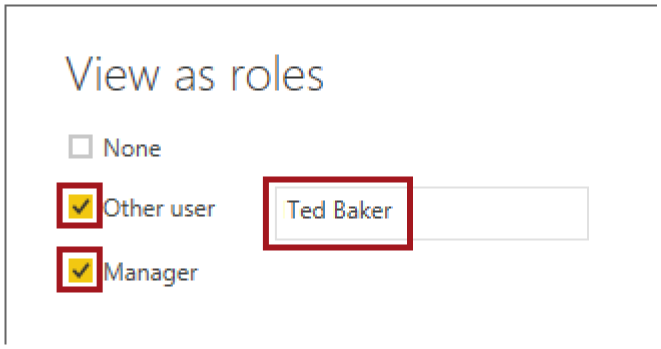
   ```
   [ManagerName] = SUBSTITUTE(USERNAME(), "~", " ")
   ```

   In the next task, you will be passing the app user's manager name (i.e. app user name) via the ***username*** *parameter, which is retrieved by the* ***USERNAME*** *function.*

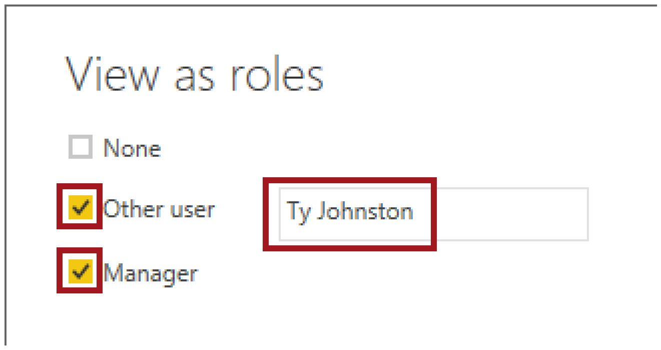
Testing the Role

In this task, you will test the dynamic role.

1. To test the role, check **Other User**, and then in the corresponding box, enter **Ted Baker**.
1. Check the **Manager** role.



2. Click **OK**.
3. In the **Filters** pane, verify that only the **Midwest** region is an available option.
4. View the report as **Ty Johnston** in the **Manager** role



5. Verify that two regions are available: **New England** and **Southwest**.
6. What happens when a manager name which is not stored in the **Manager** table is used?

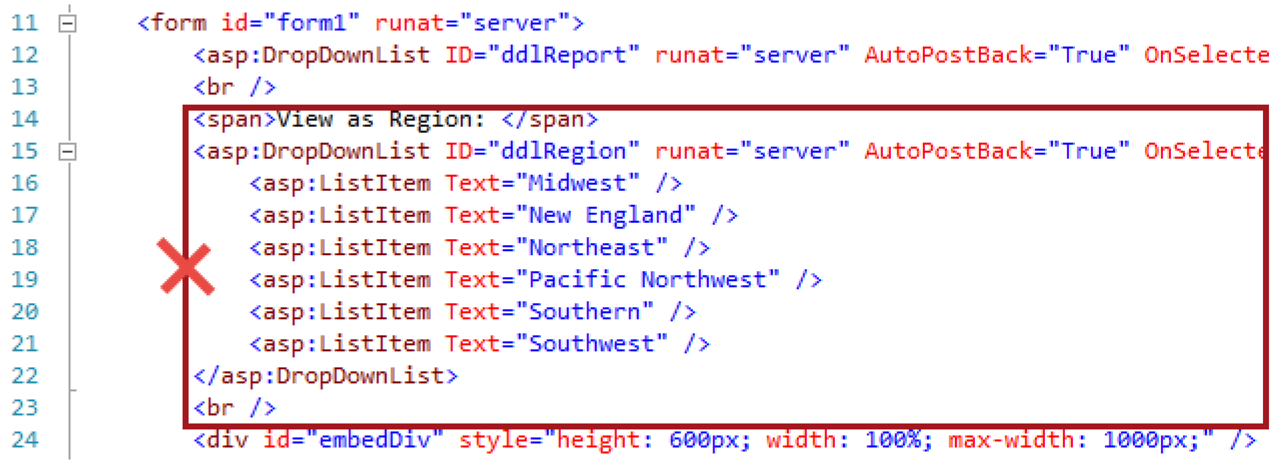
   *Answer: The **Manager** table filters to zero rows, resulting in the **Sales** table also being filtered to zero rows, producing blank results for all calculations.*

7. Stop viewing the report as the **Manager** role.
8. Save, and then publish the report to the app workspace.

Securing the Embedded Report

In this task, you will modify the **EmbedReport** page to enforce permissions based on a specific manager.

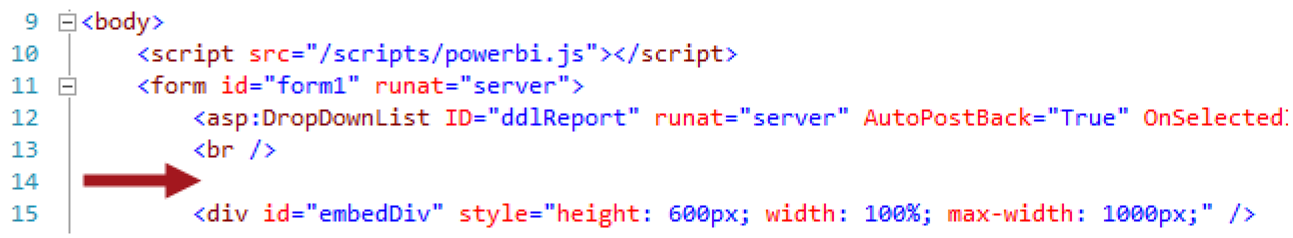
1. In Visual Studio, in the **EmbedReport.aspx** item, remove the HTML added for the region dropdown list.



```
    <form id="form1" runat="server">
        <asp:DropDownList ID="ddlReport" runat="server" AutoPostBack="True" OnSelecte
        <br />
        <span>View as Region: </span>
        <asp:DropDownList ID="ddlRegion" runat="server" AutoPostBack="True" OnSelecte
            <asp:ListItem Text="Midwest" />
            <asp:ListItem Text="New England" />
            <asp:ListItem Text="Northeast" />
            <asp:ListItem Text="Pacific Northwest" />
            <asp:ListItem Text="Southern" />
            <asp:ListItem Text="Southwest" />
        </asp:DropDownList>
        <br />
        <div id="embedDiv" style="height: 600px; width: 100%; max-width: 1000px;" />
```

2. To add a dropdown list, beneath the **<br />** element after the **asp:DropDownList** element, add the following HTML elements:

   *For convenience, the elements can be copied from the **<CourseFolder>\PowerBIDevIAD\Lab02D\Assets\Snippets.txt** file.*



```
<body>
    <script src="/scripts/powerbi.js"></script>
    <form id="form1" runat="server">
        <asp:DropDownList ID="ddlReport" runat="server" AutoPostBack="True" OnSelected
        <br />

        <div id="embedDiv" style="height: 600px; width: 100%; max-width: 1000px;" />
```

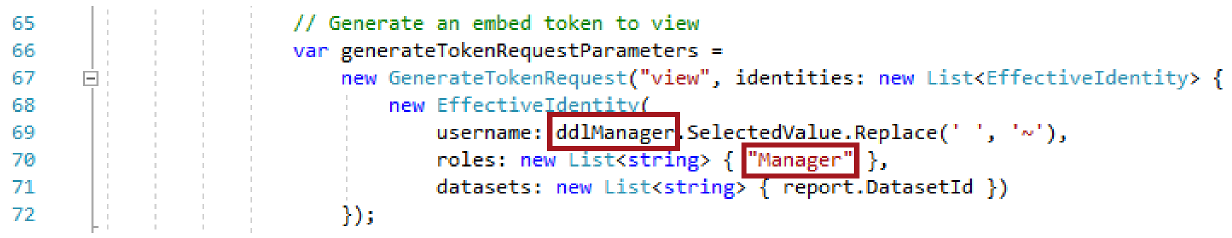
HTML

```
<span>View as Manager: </span>
<asp:DropDownList ID="ddlManager" runat="server" AutoPostBack="True" OnSelectedIndexChanged="Page_Load">
      <asp:ListItem Text="Ananya Kumar" />
      <asp:ListItem Text="Carmen Carrington" />
      <asp:ListItem Text="Haruto Suzuki" />
      <asp:ListItem Text="Jane Campbell" />
      <asp:ListItem Text="John Bishop" />
      <asp:ListItem Text="Ted Baker" />
      <asp:ListItem Text="Ty Johnston" />
</asp:DropDownList>
<br />
```

This dropdown list is for testing purposes only. Ordinarily, the developer will determine the manager from the profile of the authenticated user.

1. In the code-behind file for the **EmbedReport.aspx** item, in the **Page_Load** method, modify the **generateTokenRequestParameters** variable declaration:

   - For the **username** parameter, replace **ddlRegion** with **ddlManager**
   - For the **roles** parameter, replace **Region** with **Manager**



```
// Generate an embed token to view
var generateTokenRequestParameters =
    new GenerateTokenRequest("view", identities: new List<EffectiveIdentity> {
        new EffectiveIdentity(
            username: ddlManager.SelectedValue.Replace(' ', '~'),
            roles: new List<string> { "Manager" },
            datasets: new List<string> { report.DatasetId })
    });
```

2. Start the web application, and then navigate to the **Embed Reports** page.
3. Select different managers, and verify that the report presents only data for the selected manager (some managers are associated with two regions).
4. Close the web session, and then stop debugging.
5. To close Visual Studio, on the **File** menu, select **Exit**.

   This exercise demonstrated how model data and relationships can be used to enforce row-level security, in this case for a manager associated with one or more regions.

Summary

In this lab, you explored three different security scenarios by creating static roles, a dynamic role, and also by developing a data-driven security design.

Terms of Use

Document Version

#	Date	Author	Comments
2	31-DEC-2017	Peter Myers	Visual Studio 2017 v15.2 (26430.14) Power BI service v13.0.3495.220 Microsoft.IdentityModel.Clients.ActiveDirectory v3.17.3 Microsoft.PowerBI.Api v2.0.8 MicrosoftPowerBI.JavaScript v2.4.2 Microsoft.Rest.ClientRuntime v2.0.1 Newtonsoft.Json v7.01
3	21-FEB-2018	Peter Myers	Visual Studio 2017 v15.2 (26430.14) Power BI service v13.0.4385.126 Microsofot.IdentityModel.Clients.ActiveDirectory v3.19.1 Microsoft.PowerBI.Api v2.0.10 MicrosoftPowerBI.JavaScript v2.4.7 Microsoft.Rest.ClientRuntime v2.3.10 Newtonsoft.Json v11.0.1
4	30-JUN-2018	Peter Myers	(Resequenced from **Lab 02C** to **Lab 02D**) Power BI service v13.0.5702.156 Power BI Desktop v2.59.5135.601 (June 2018) 64-bit Microsofot.IdentityModel.Clients.ActiveDirectory v3.19.8 Microsoft.PowerBI.Api v2.0.12 MicrosoftPowerBI.JavaScript v2.5.1 Microsoft.Rest.ClientRuntime v2.3.11 Newtonsoft.Json v11.0.2