Practices for Lesson 16: Analyzing Your Relational Model

Chapter 16

Practice 16-1: Analyze Your Relational Model

Task

For the following relational model, which is based on the Cascade Cruise case study in Practice 14-1, change the tables' colors to the default yellow background, and then add or modify existing design components based on the following requirements:

A. Ability to know how many passengers were on a particular cruise for each month

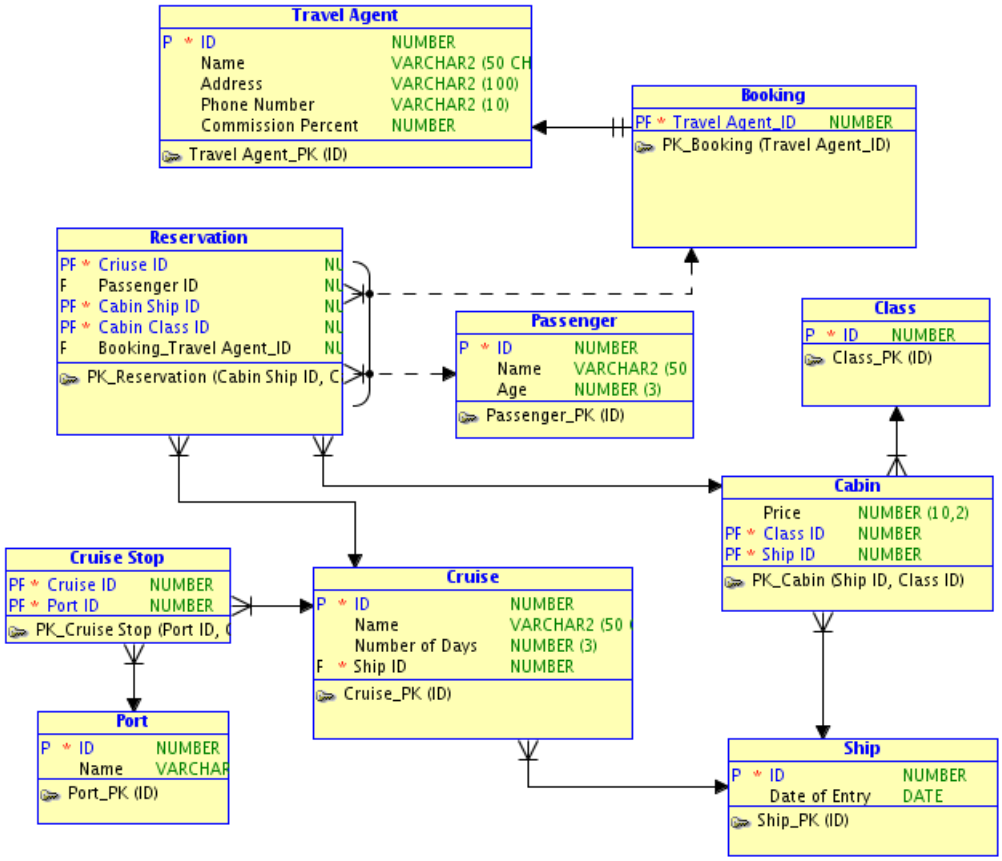
B. Ability to quickly see the cruises that a particular ship has made

C. Ability to know how well each cruise did as far as revenue

D. Ability to know the total commission that each travel agent made

E. Ability to quickly see the average age of passengers on a particular cruise

Note: The DFD provided in the following is the relational view of `sol_14_01.dmd`.



Solution 16-1: Analyze Your Relational Model

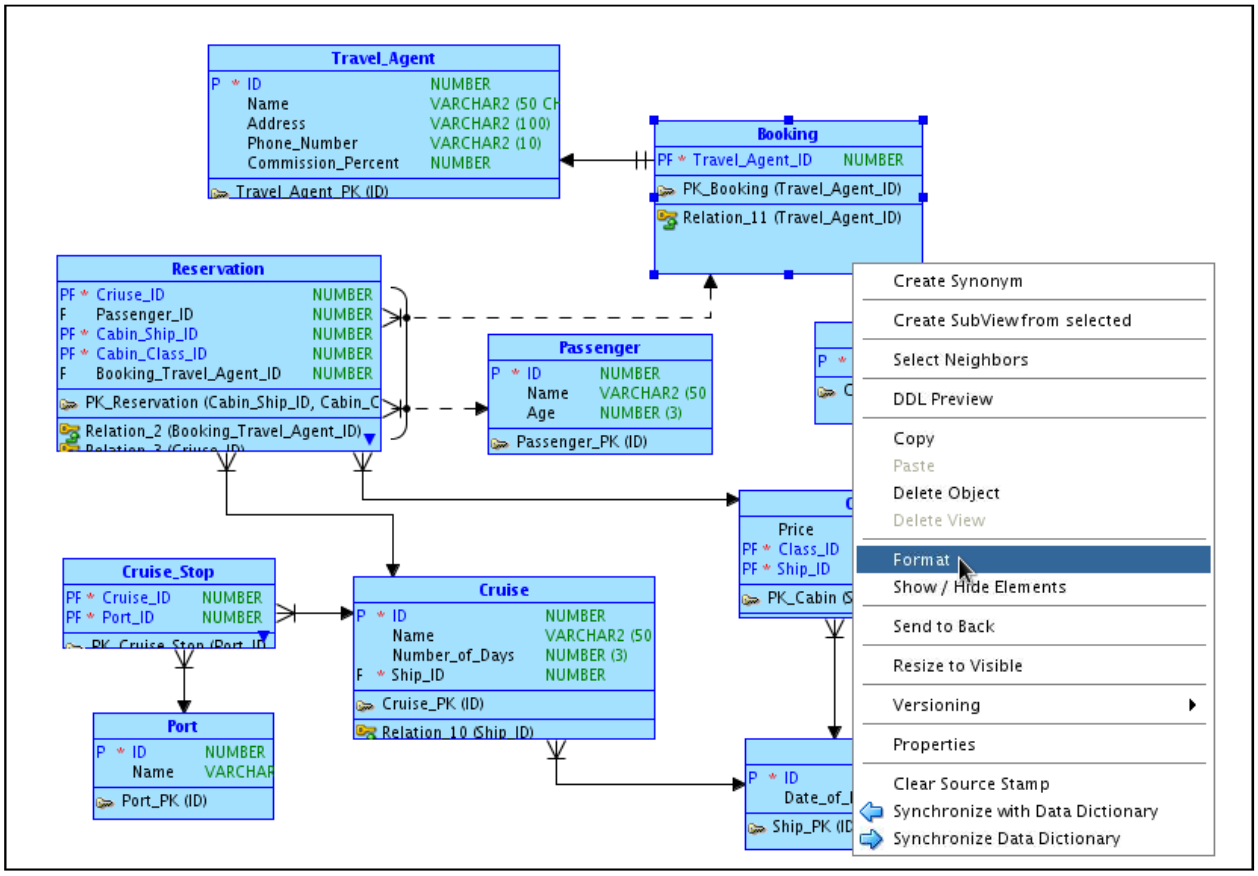
For the following relational model, add or modify existing design components based on the following requirements.

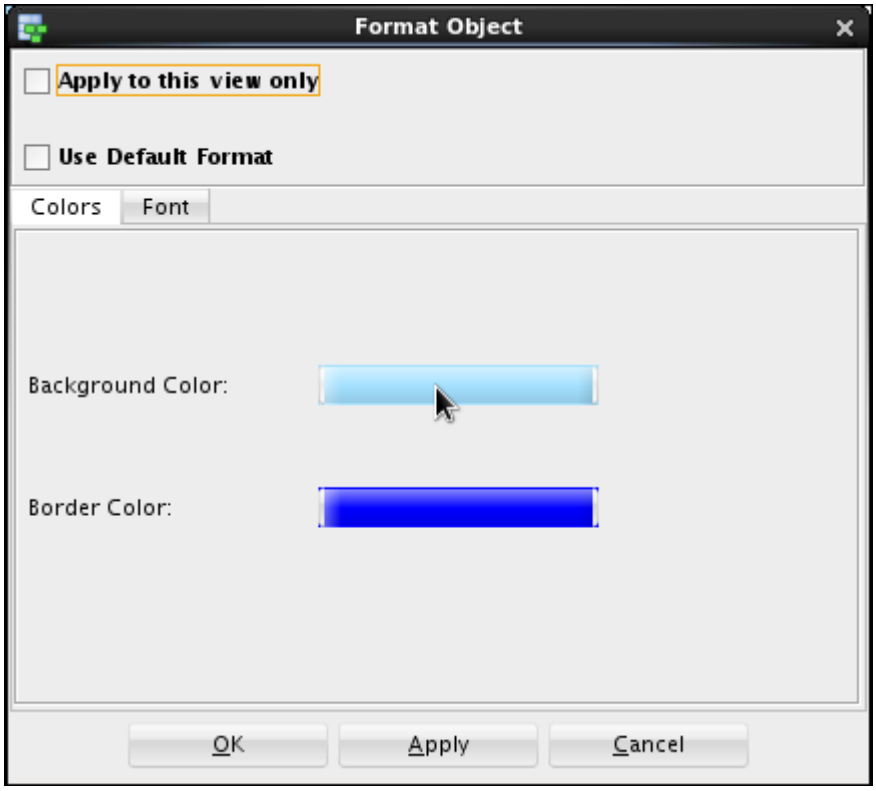
A. Ability to know how many passengers were on a particular cruise for each month

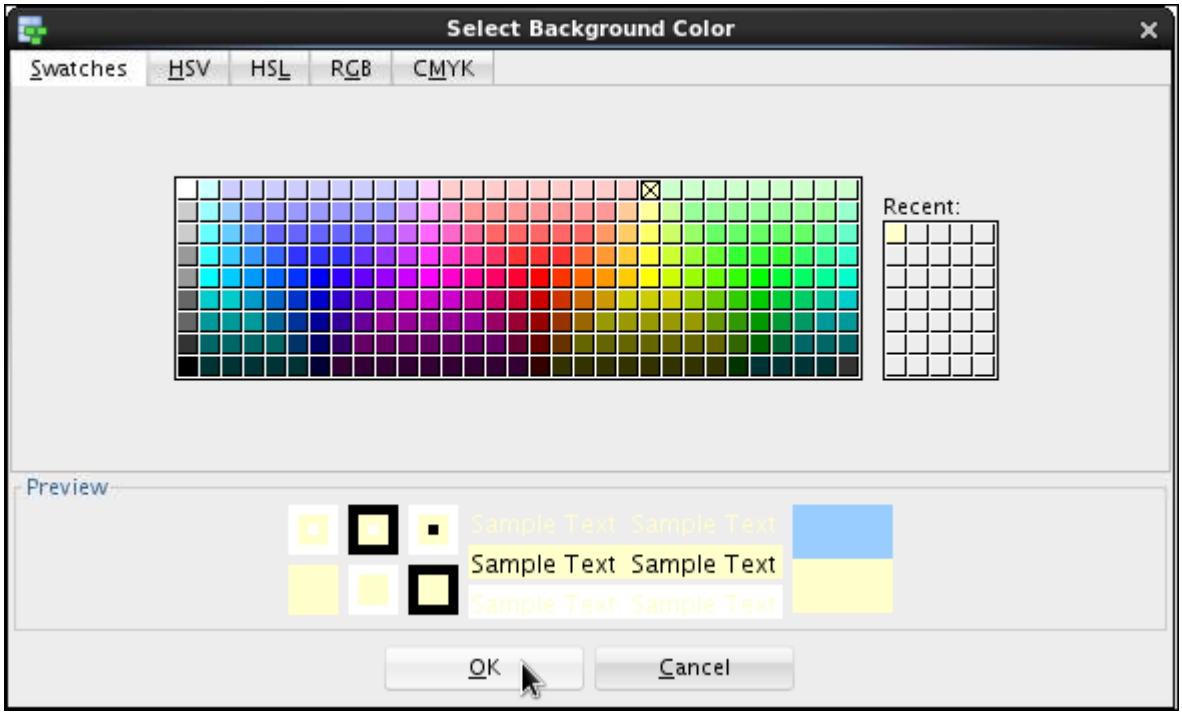
To accomplish this requirement, you could create a view on the `Passenger` and `Cruise` tables. Perform the following steps:

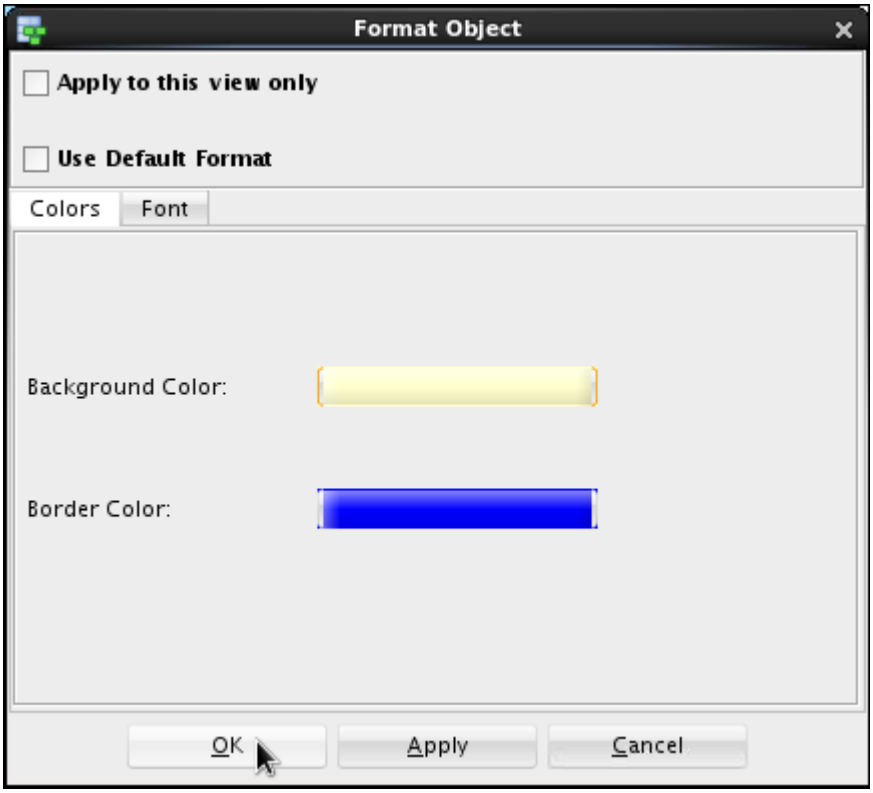
1. Open `sol_14_01.dmd`. If the logical model is not displayed, right-click the **Logical** node under the `sol_14_01` node, and then select **Show** from the pop-up menu. The Logical design is displayed. To display the labels, right-click in the white space of the Logical model, and then select **Show > Labels** from the pop-up menu.
2. Forward engineer the model to create the relational model.
3. In this practice, you will change the color of the tables from blue to the yellow as follows: Select each table one by one, then right-click and select **Format**. Select yellow color and click **OK**. The background color of the table is now yellow.

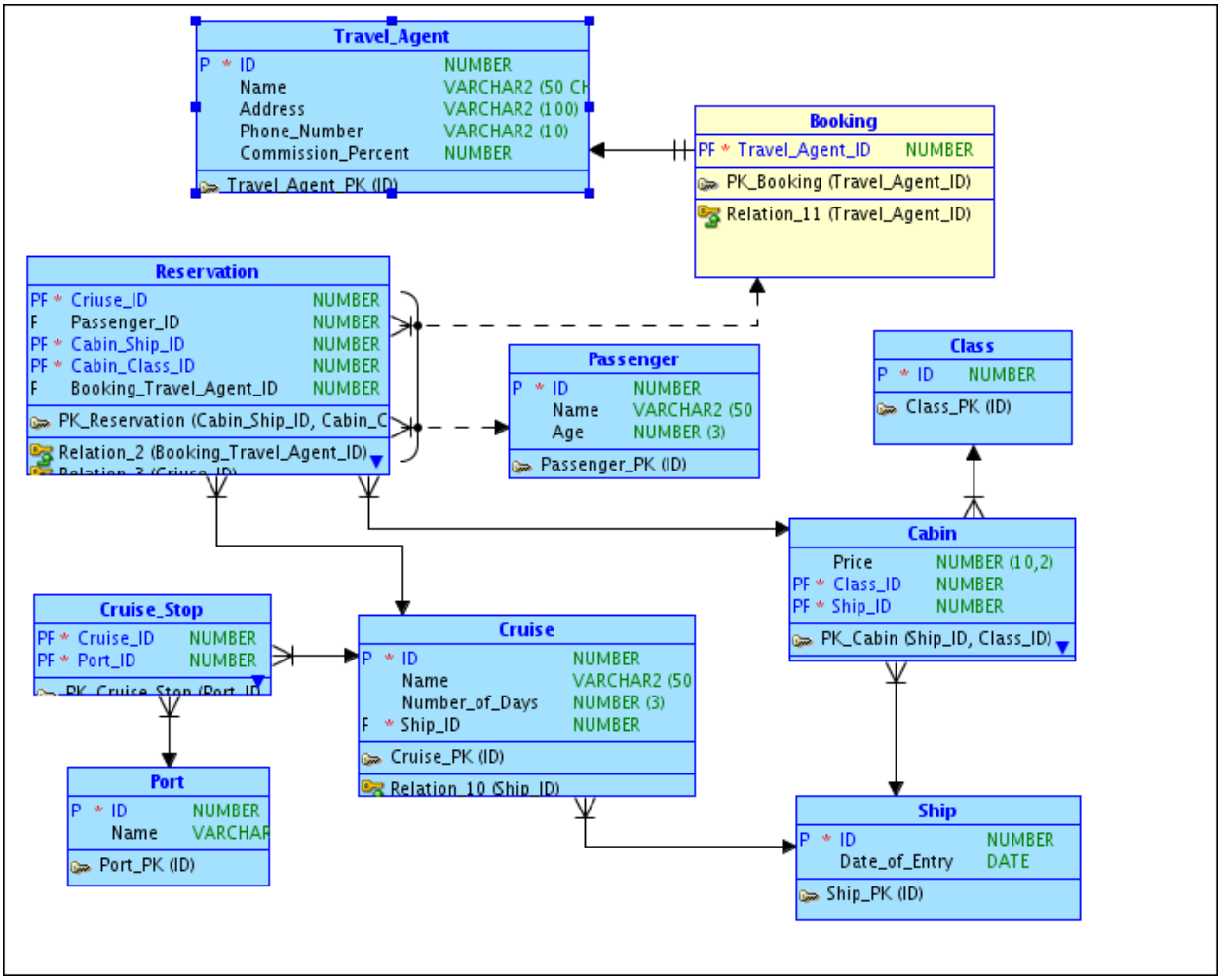
   Note: The default color for the Relational Model is green. In the practices, yellow color is used.



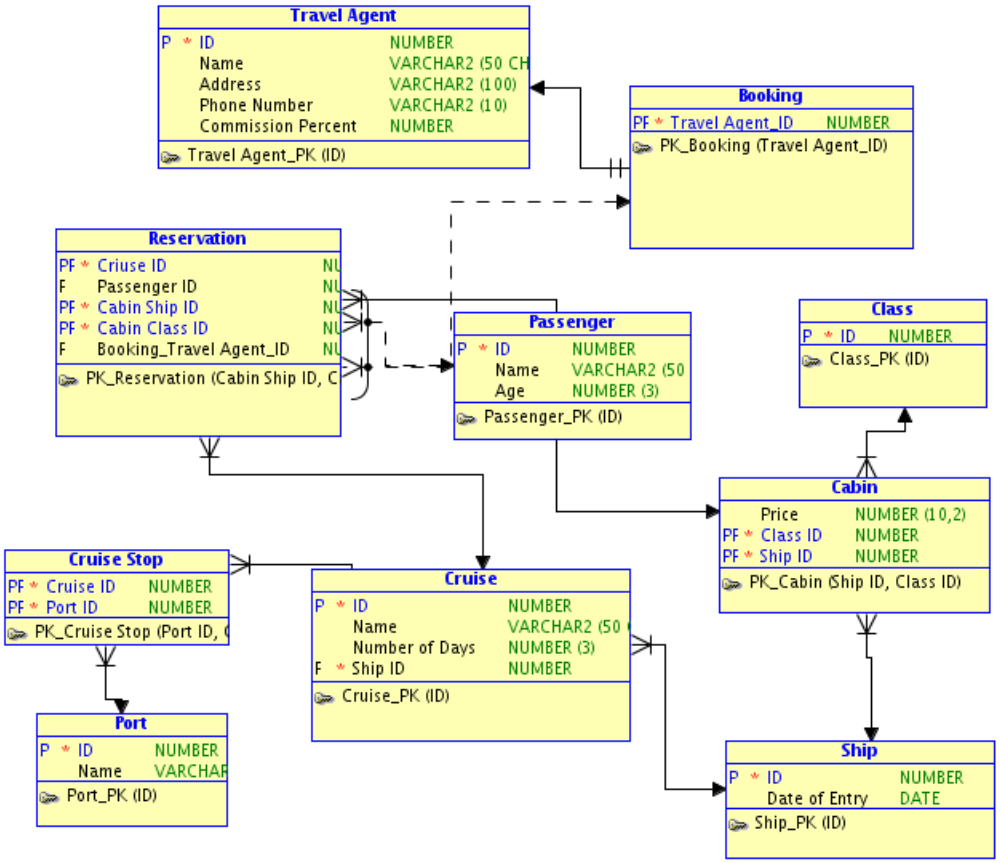

Format Object
Apply to this view only
Use Default Format
Colors
Font
Background Color:
Border Color:
OK
Apply
Cancel


Select Background Color
Swatches
HSV
HSL
RGB
CMYK
Recent:
Preview
Sample Text Sample Text
OK
Cancel

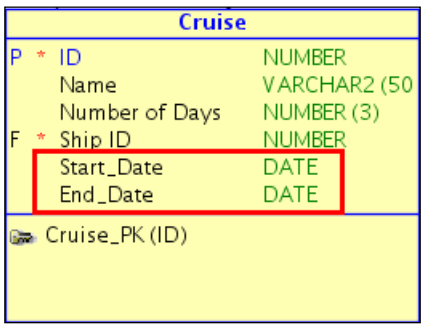

Format Object
Apply to this view only
Use Default Format
Colors
Font
Background Color:
Border Color:
OK
Apply
Cancel



Repeat the same step for each table.

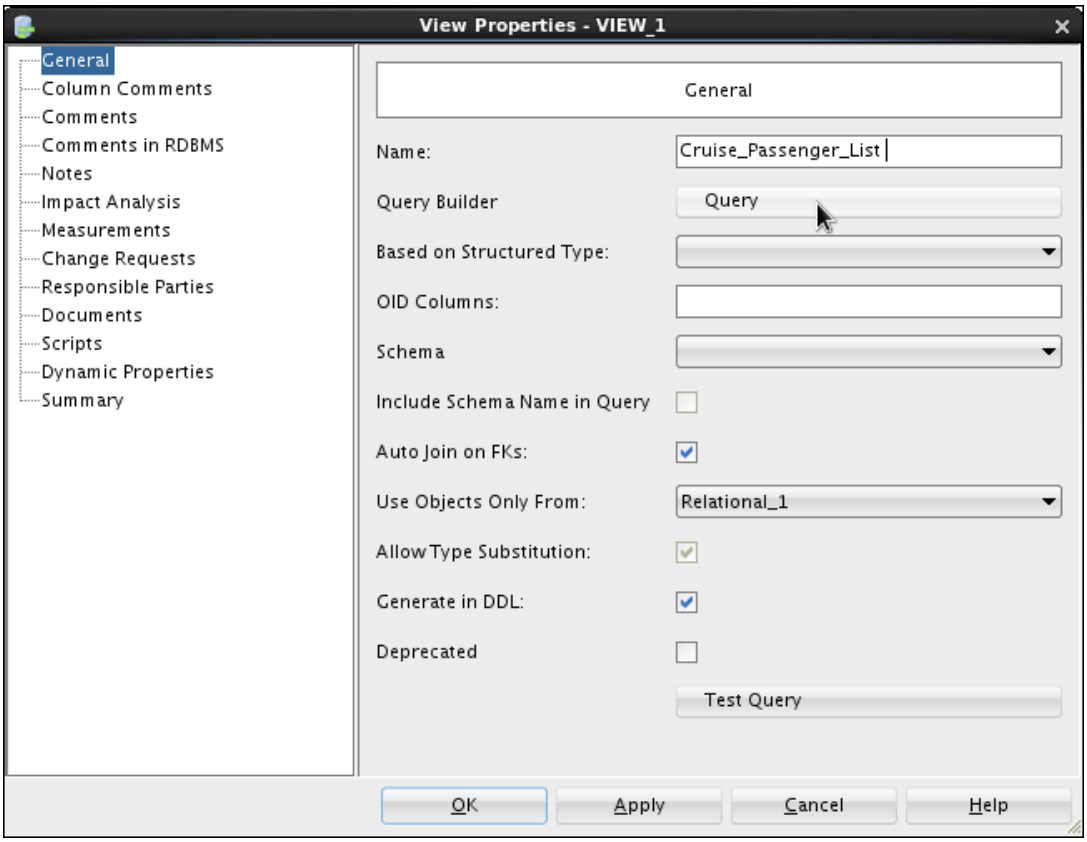


4. Add two `DATE` columns to the `Cruise` table to store `Start_Date` and `End_Date`. In the **Relational Model** diagram, double-click the `Cruise` table, and then select the **Columns** property in the left navigator. Add the two columns, and then click **OK**.

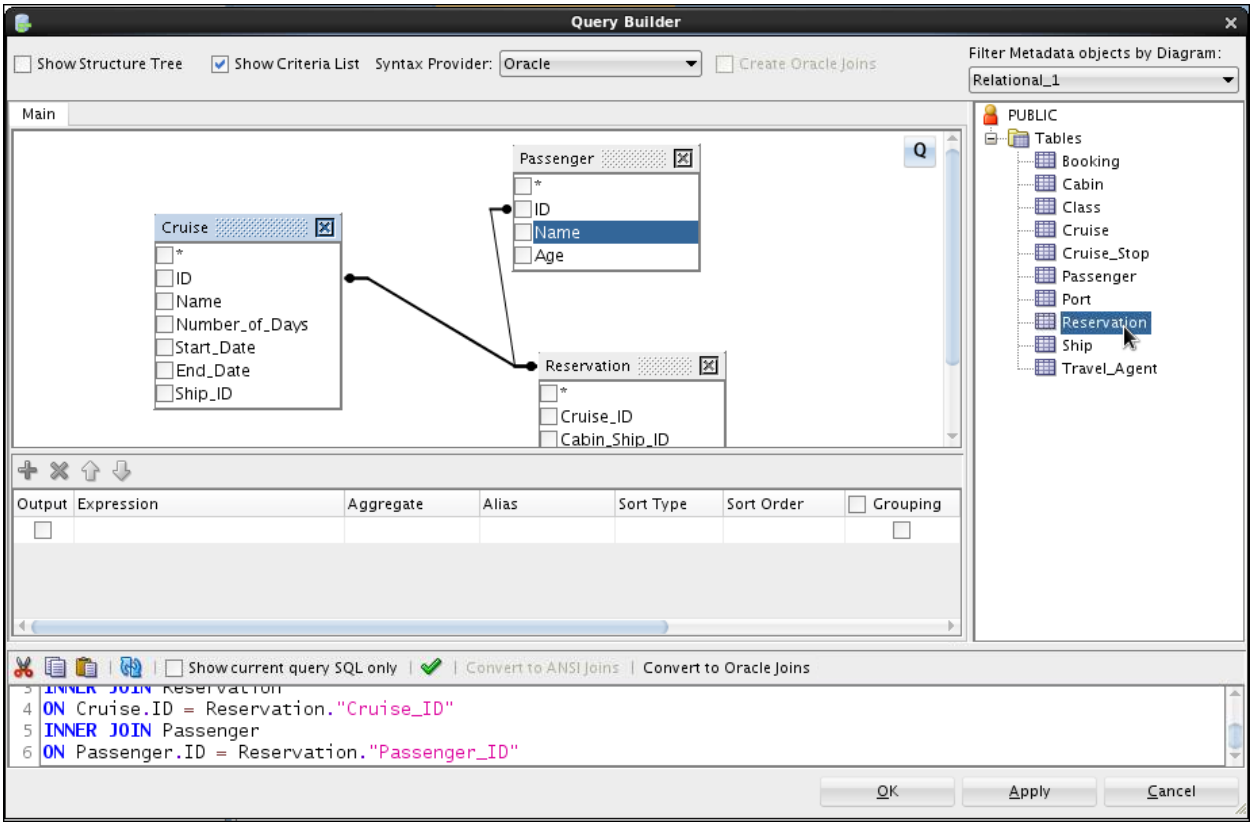


5. Add a view on the `Passenger`, `Cruise`, and `Reservation` tables. Click the **New View**  icon, and then click in the white space of the relational diagram.

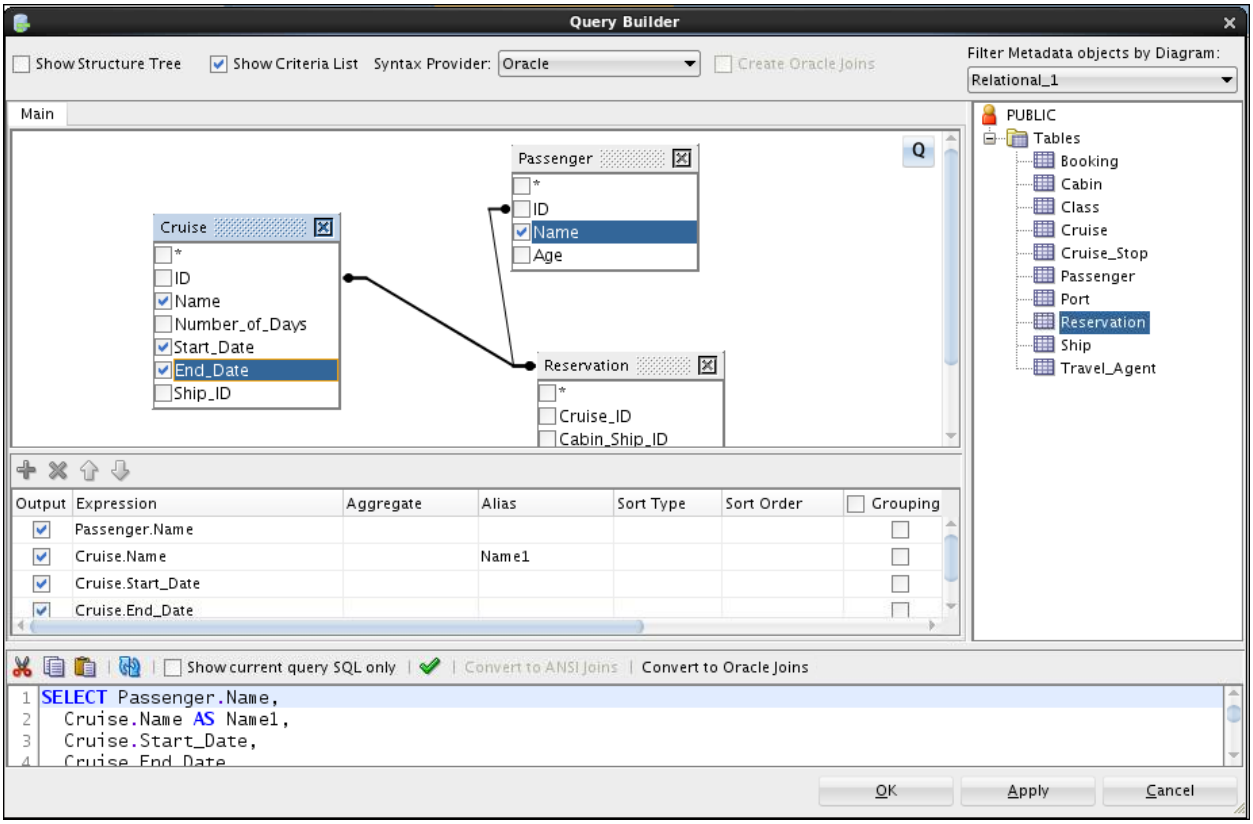
6. Enter `Cruise_Passenger_List` for the **Name**, and then click the **Query** button because you are going to base this view on a query. The **Query Builder** dialog box is displayed.



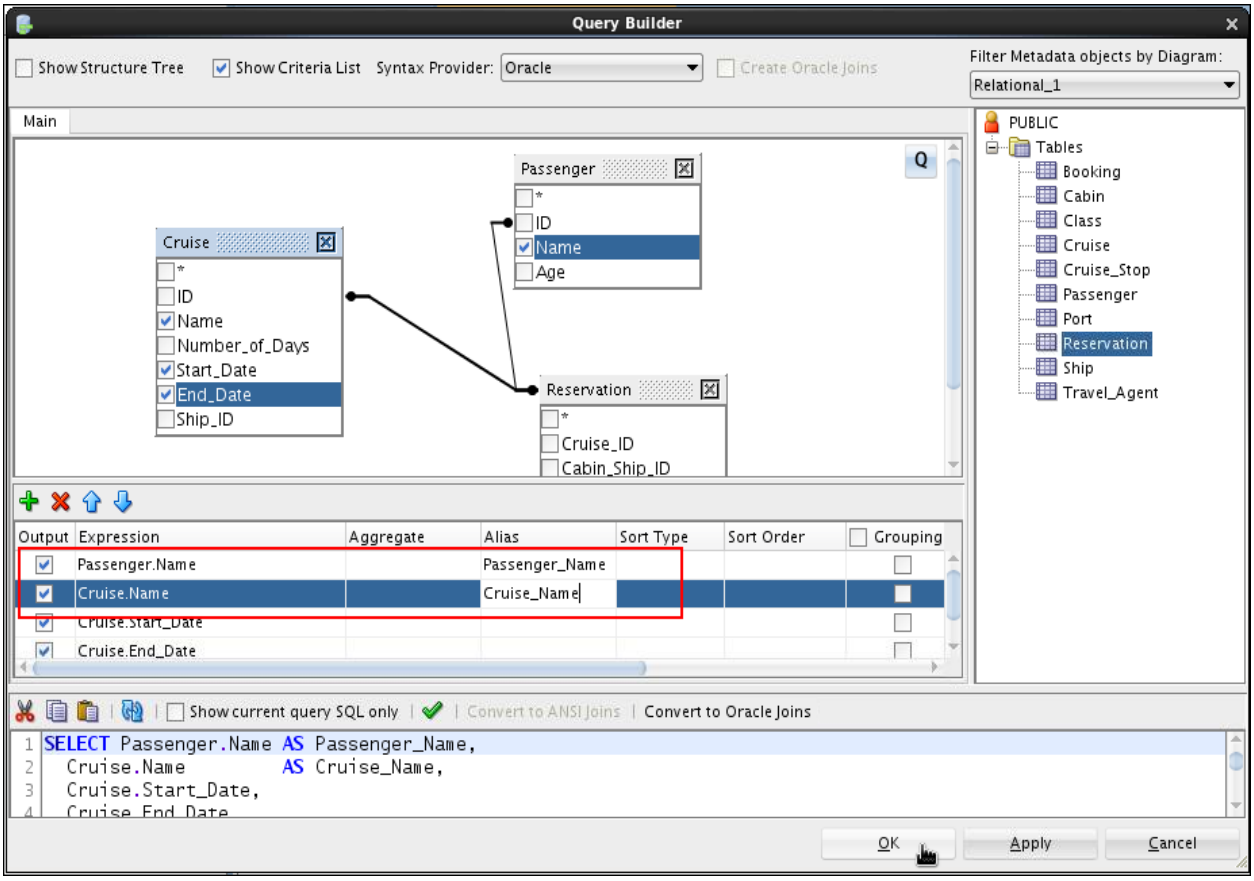
7. Select the `Cruise`, `Passenger`, and `Reservation` tables by double-clicking the table names in the `PUBLIC` region. The selected tables are displayed in the **Main** tab.

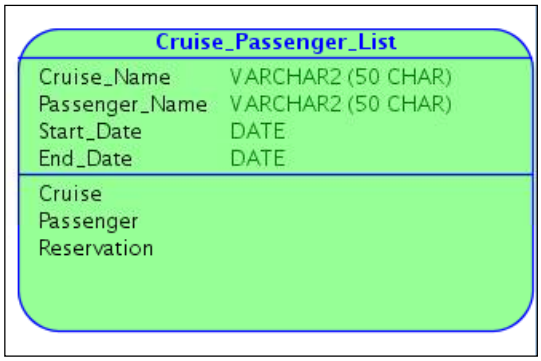


8. Select the `Name` column from both the `Cruise` and `Passenger` tables and `Start_Date` and `End_Date` from the `Cruise` table. You select a column by clicking the square to the left of the column name.

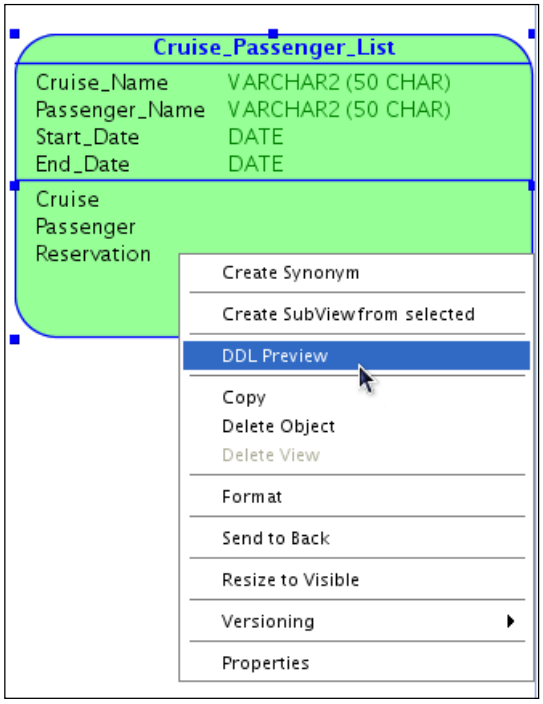


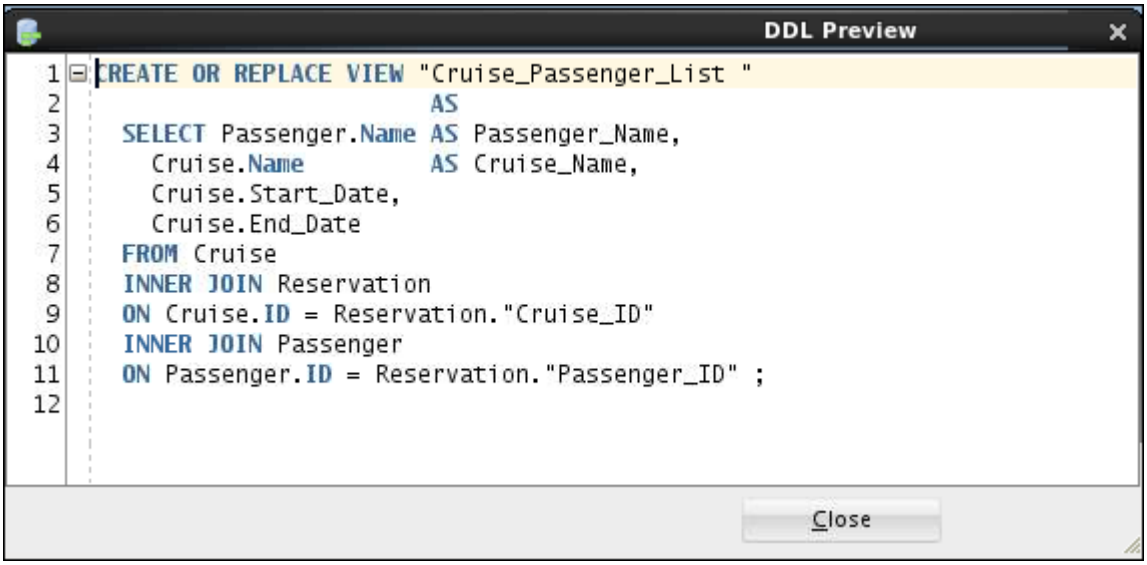
9. Because the `Name` column in both `Cruise` and `Passenger` are the same, you must create an alias for each in the alias field. Enter `Cruise_Name` as an alias for the `Cruise.Name` column in the **Alias** column. Enter `Passenger_Name` as the alias name for the `Passenger.Name` column in the **Alias** column, and then click **Apply**. Click **OK** twice. The new view is displayed.





10. To review the DDL that will be generated, right-click the view, and then select **DDL Preview** from the pop-up menu. The **DDL Preview** dialog box is displayed. When done viewing the code, click **Close**.





B. Ability to quickly see the cruises that a particular ship has made

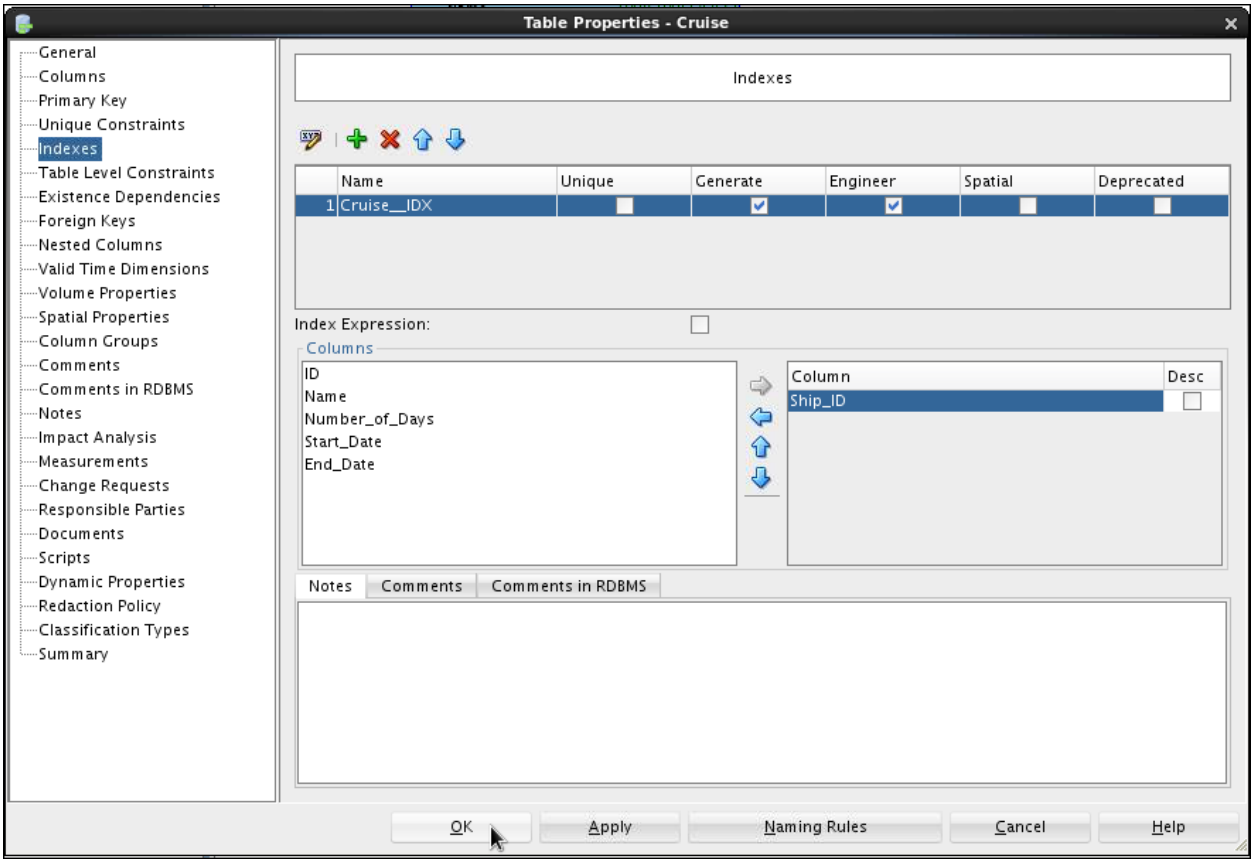
To accomplish this requirement, you could create an index on the `SHIP_ID` foreign key in the `Cruise` table. Perform the following steps:

1. Add an index on the `SHIP_ID` foreign key in the `Cruise` table. Double-click the `Cruise` table.

2. Select the **Indexes** property in the left navigator.

3. Click the **Add** icon.

4. Select the `Ship_ID` column, and then click the arrow to move the column over to the right. Next, click **OK** to create your index.

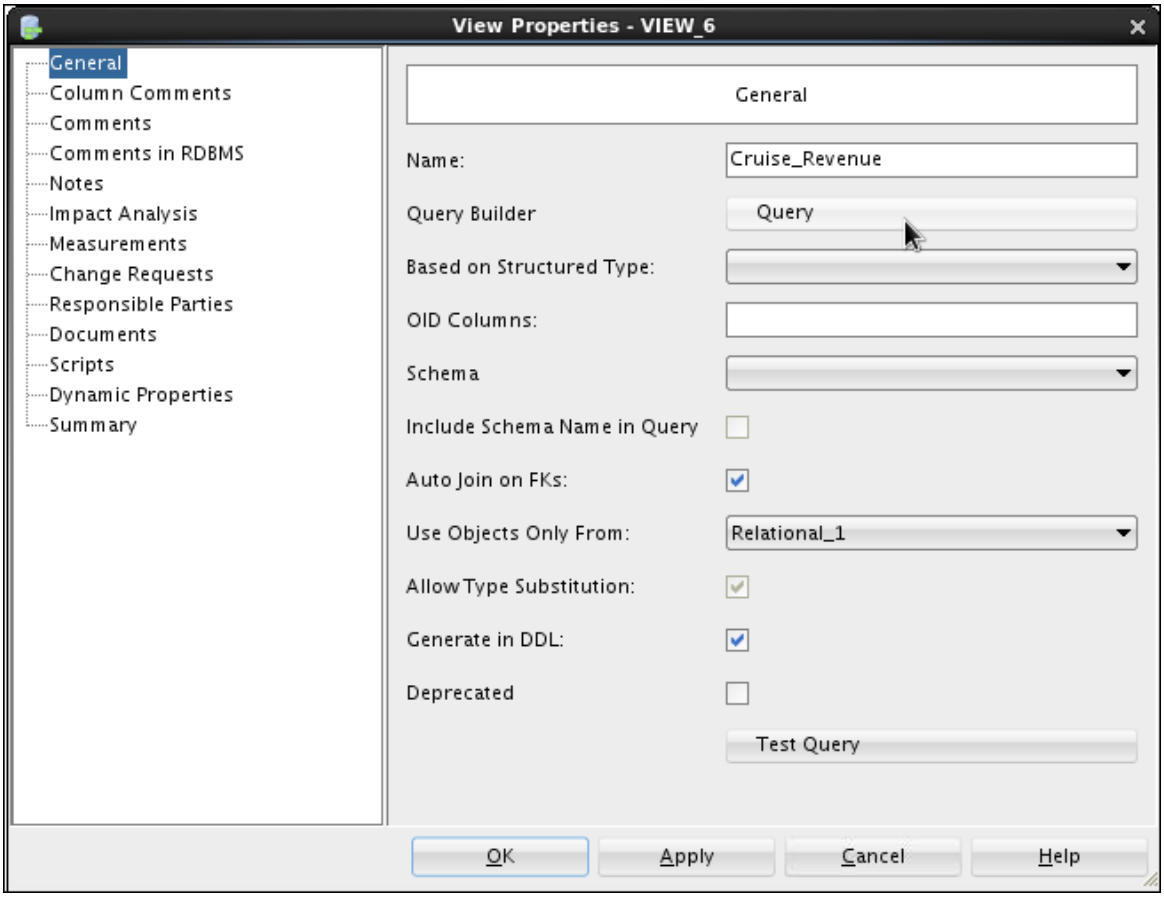


C. Ability to know how well each cruise did as far as revenue

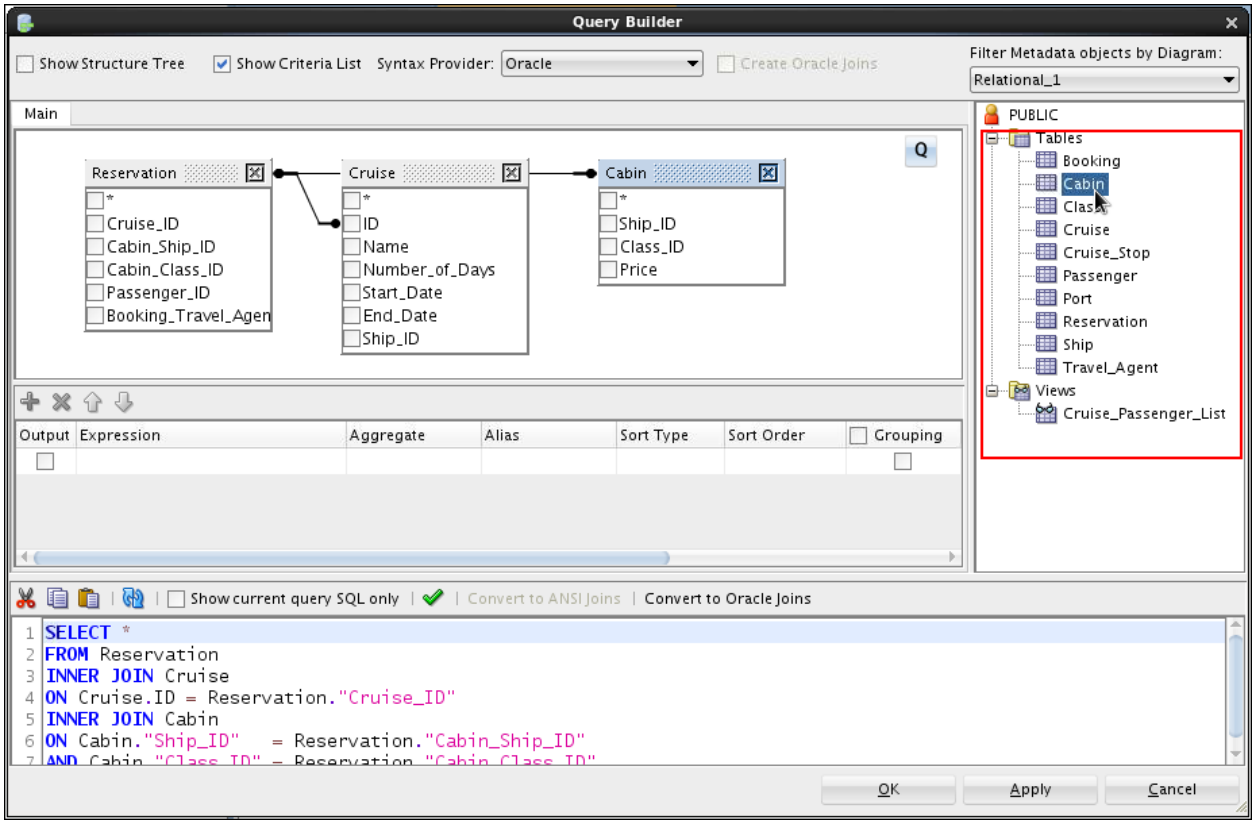
To accomplish this requirement, you could create a view between `Reservation`, `Cruise`, and `Cabin` where you sum the total cabins for a particular cruise. Perform the following steps:

1. Add a view on `Reservation`, `Cruise`, and `Cabin`. Click the New View icon, and then click in the white space of the relational diagram.

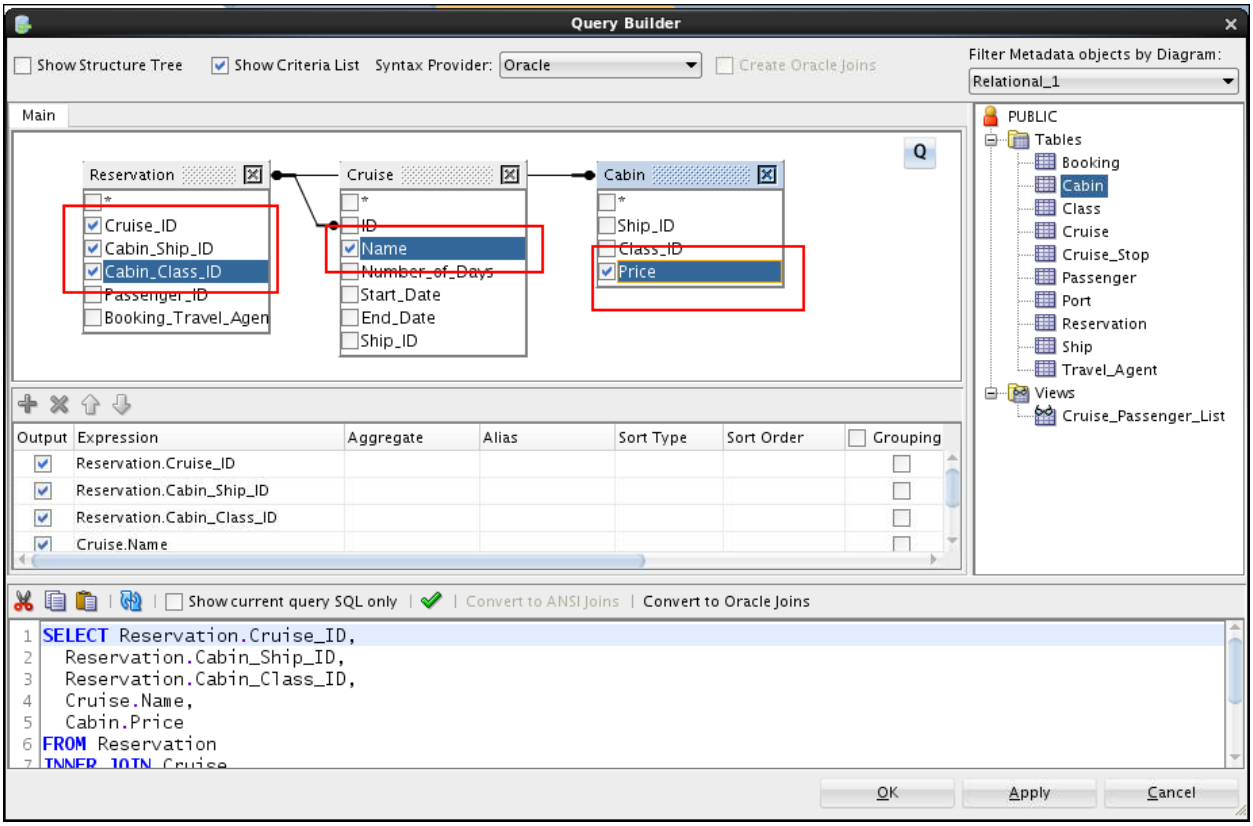
2. Enter `Cruise_Revenue` for the **Name**, and then click the **Query** button because you are going to base this view on a query. The **Query Builder** dialog box is displayed.



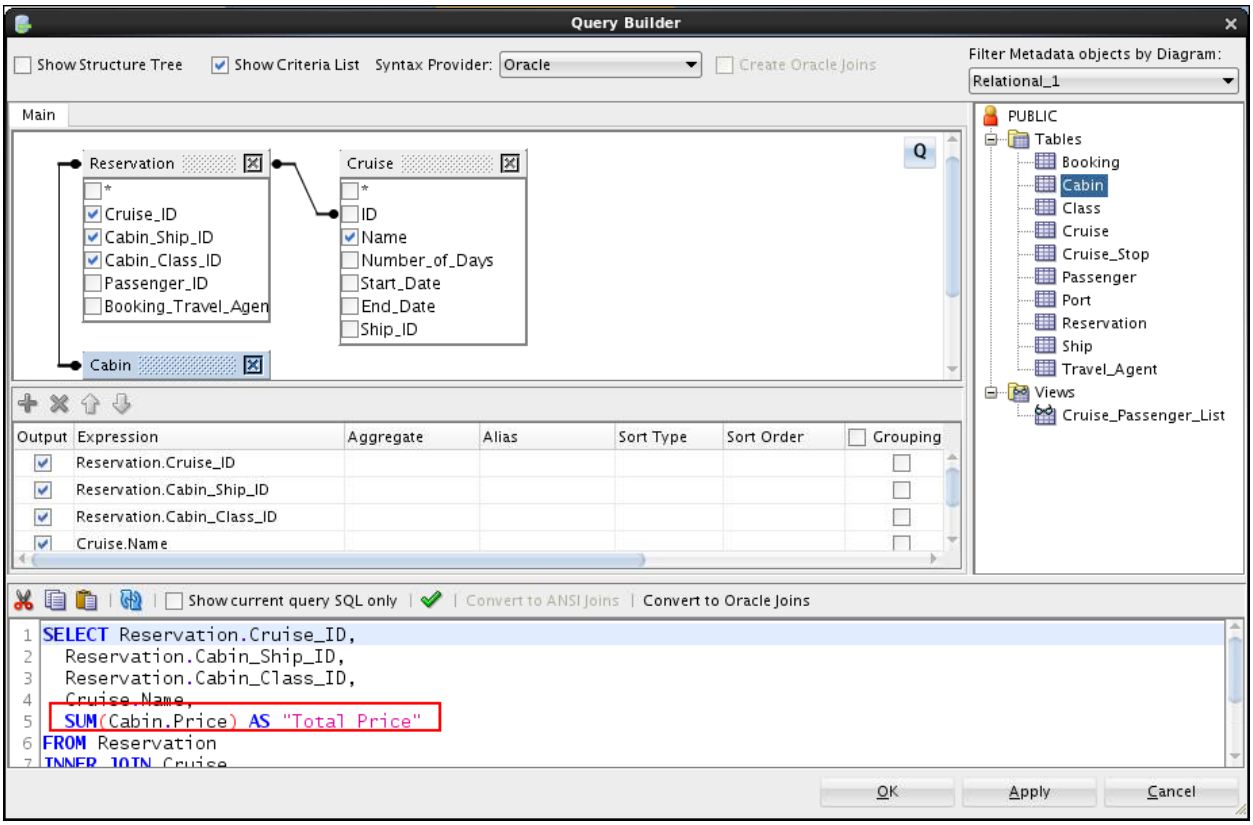
3. Double-click the `Reservation`, `Cruise`, and `Cabin` tables from the list of the tables on the right side of the dialog box.



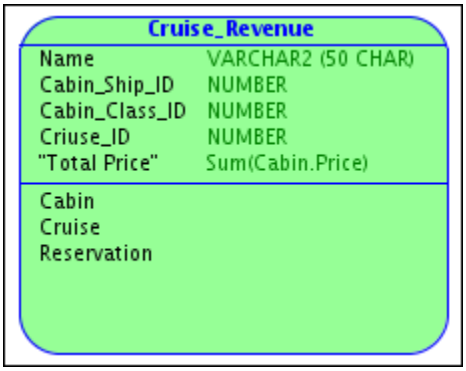
4. Select the `Price` column from the `Cabin` table, the `Name` column from the `Cruise` table, the `Cruise_ID`, `Cabin_Ship_ID`, and `Cabin_Class_ID` columns from the `Reservation` table, by clicking the boxes next to each of those columns in the **Main** tab area.



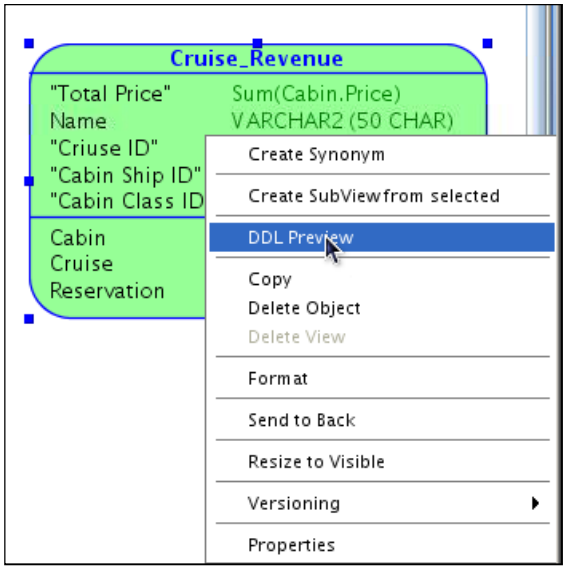
5. Modify the DDL to sum the total for price. In the DDL section of the **Query Builder** dialog box, under `SELECT`, change the `Cabin.Price` DDL to `SUM(Cabin.Price) AS "Total Price"`, click the **Update Diagram** icon, click Apply, and then click **OK**.

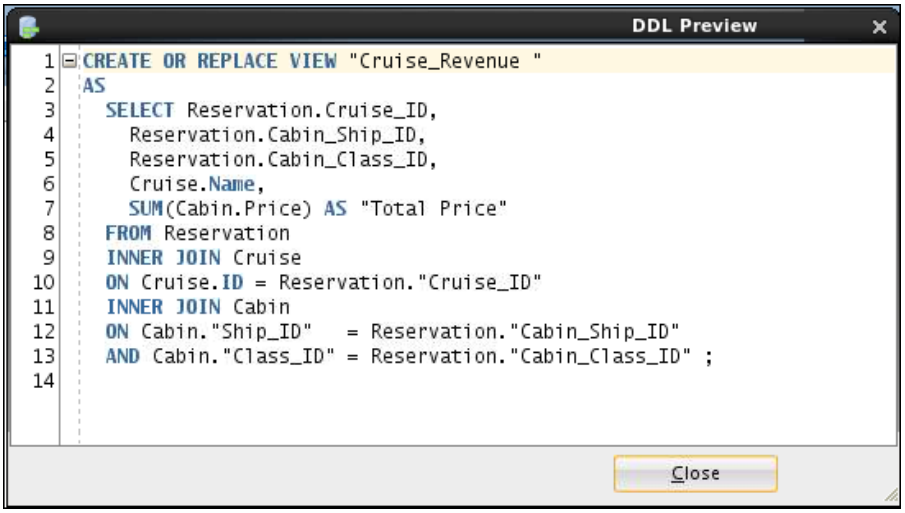


6. Click **OK** again to create the view. The newly created view is displayed.



7. To view the DDL for the view, right-click the view, and then select **DDL Preview** from the pop-up menu. The **DDL Preview** dialog box is displayed.





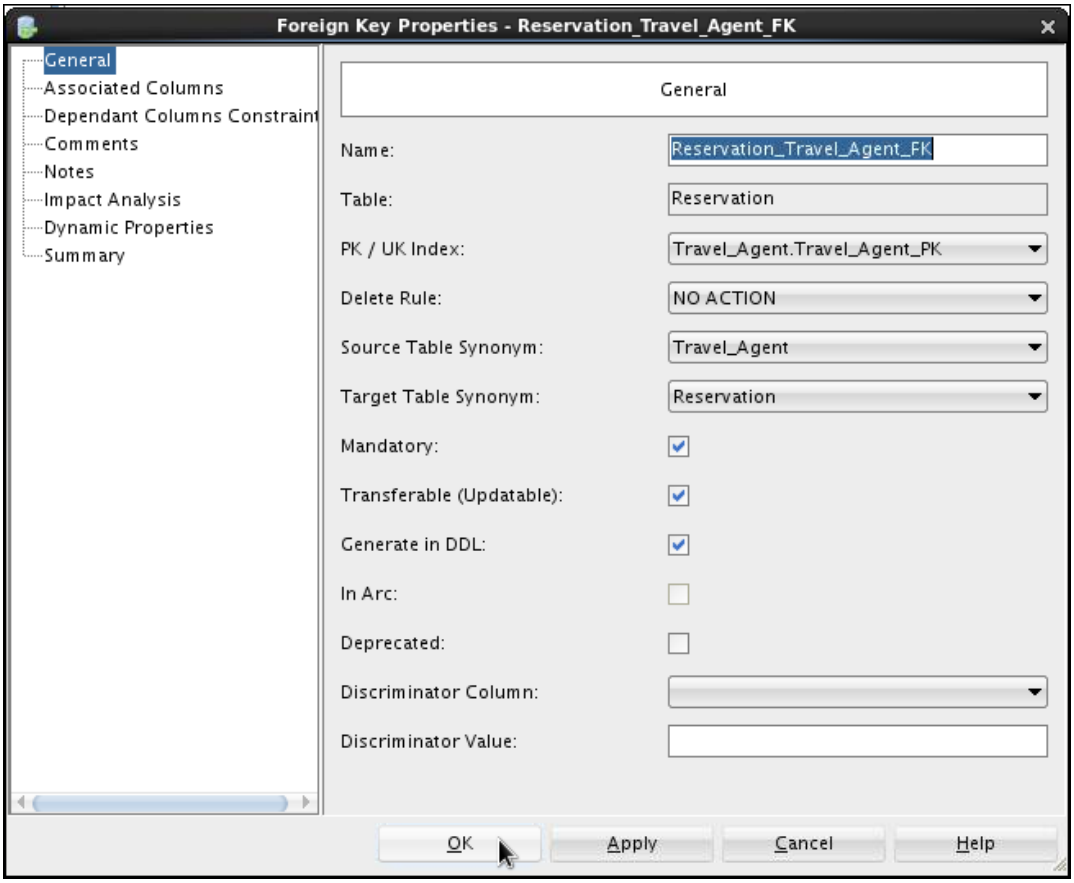
8. When done viewing the code, click **Close**.

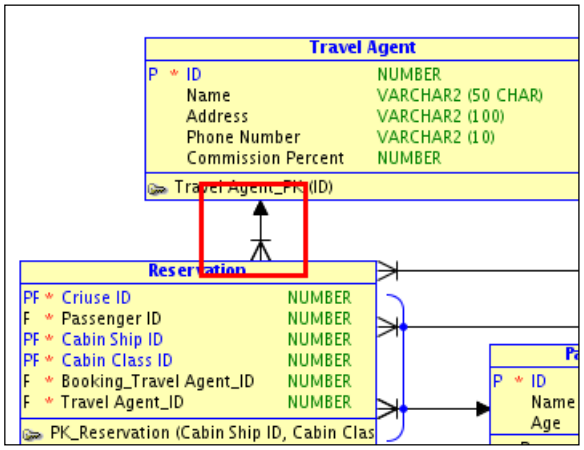
D. Ability to know the total commission that each travel agent made

To accomplish this requirement, you could create an index on booking for each travel agent, and then create a view to calculate commission for each booking by reservation. Perform the following steps:

1. Create a foreign key between `Travel_Agent` and `Reservation` tables. Click the **New Foreign Key** icon, and then click the `Travel_Agent` table and then the `Reservation` table.

2. When the **Foreign Key Properties** dialog box is displayed, click **OK**.



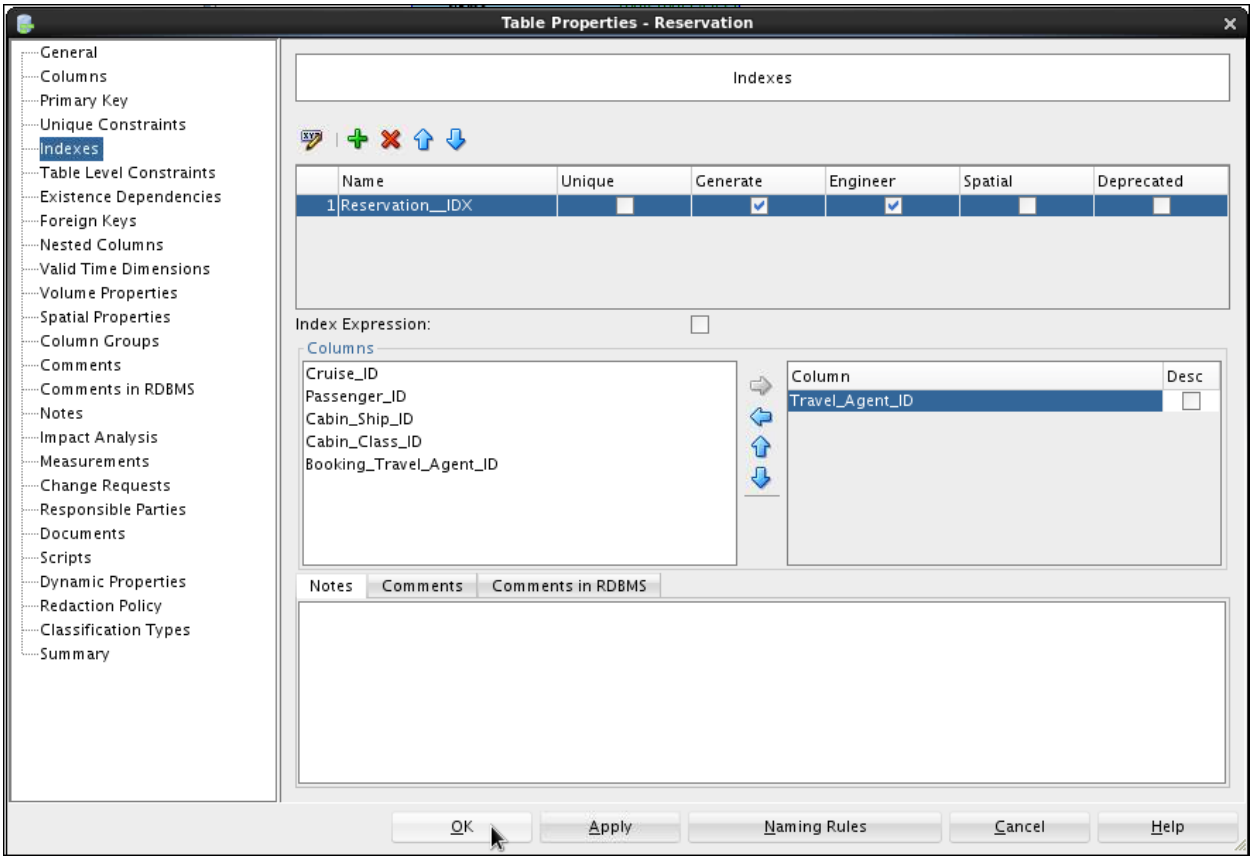


Note: The name of the foreign key that you create and display in the **Foreign Key Properties** dialog box might be different from what you see in this solution.

3. Double-click the `Reservation` table to display the **Table Properties** dialog box.
   Note: You must click the **Select** icon on the toolbar first.
4. Select the **Indexes** property in the left navigator.

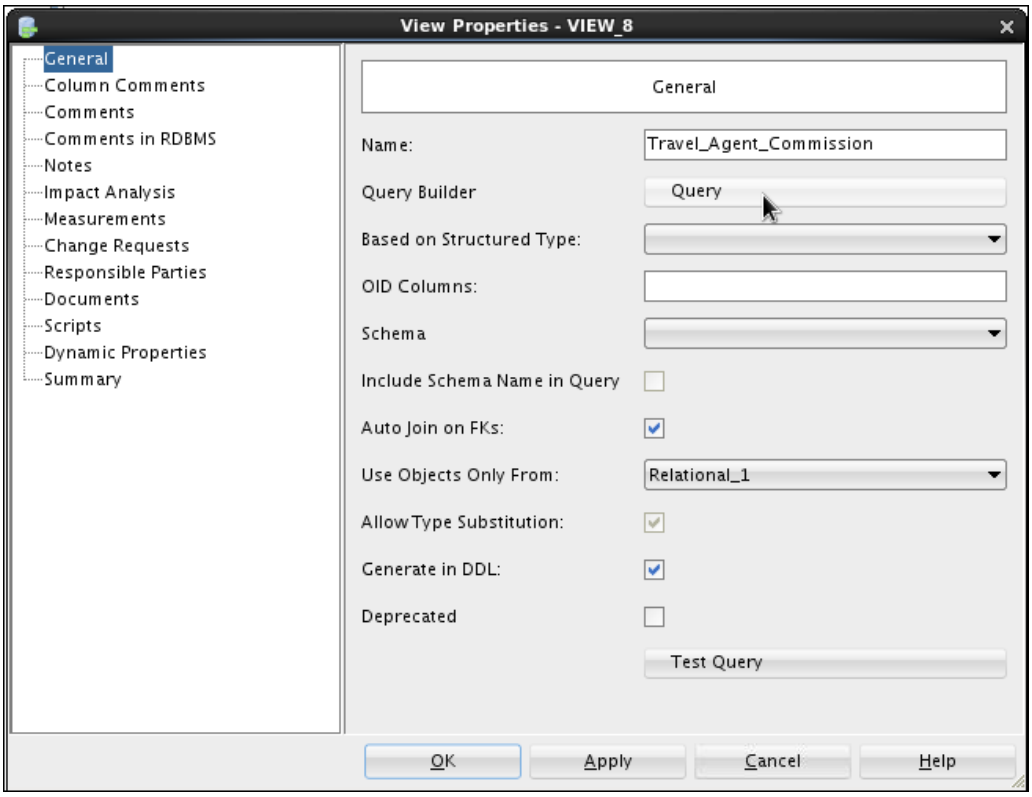
5. Click the Add icon.

6. Select the `Travel_Agent_ID` column, click the **Add** icon, and then click **OK**.



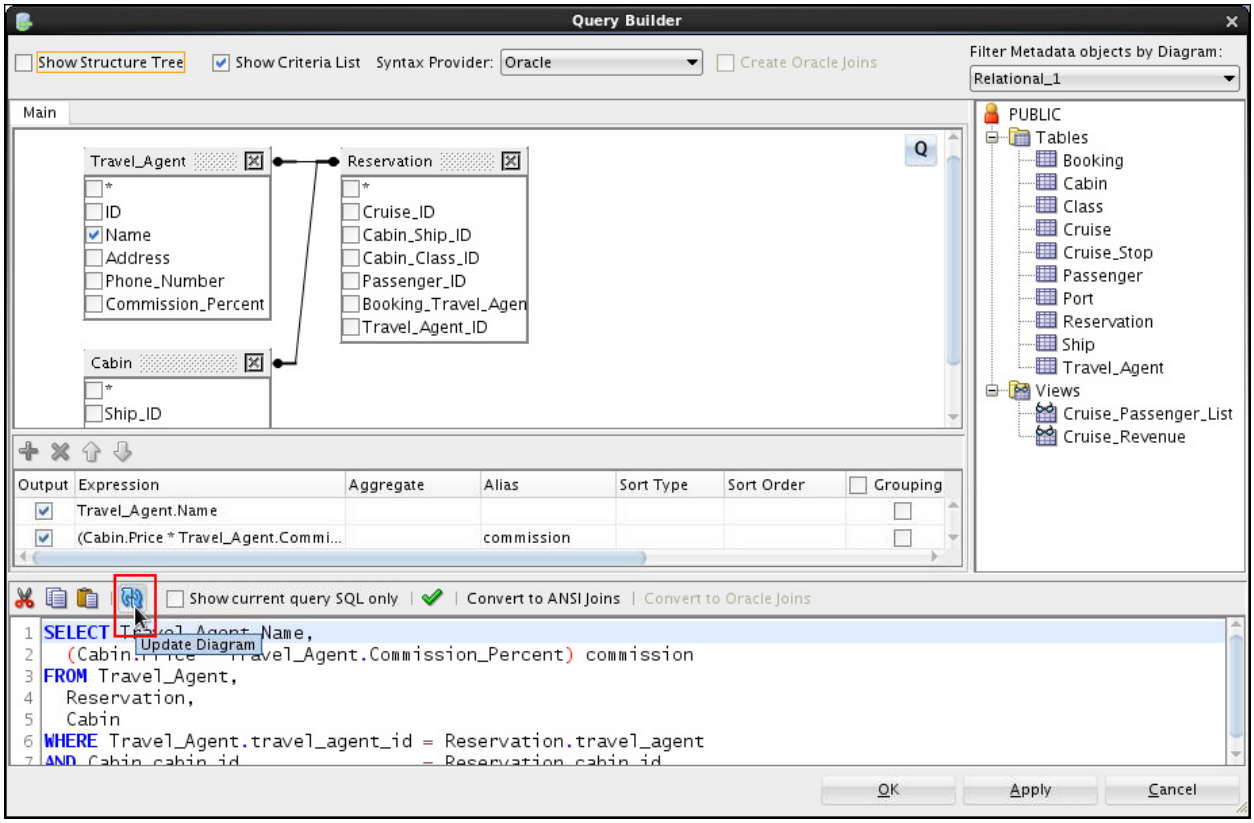
7. Now you can create the view. Click the New View icon, and then click in the white space of the relational diagram.

8. Enter `Travel_Agent_Commission` for the **Name**, and then click the **Query** button because you are going to base this view on a query.

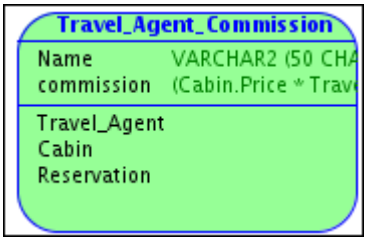


9. You can simply type the view's query in the DDL section of the **Query Builder** dialog box. Enter the following query, and then click the **Update Diagram** icon. The **Query Builder** dialog box and other areas are automatically populated. Click **OK**.

```
SELECT travel_agent.name,
(cabin.price*travel_agent.commission_percent) commission
FROM travel_agent,
reservation,
cabin
WHERE
travel_agent.travel_agent_id = reservation.travel_agent
AND cabin.cabin_id = reservation.cabin_id
```



10. Click **OK** again to create the view.



E. Ability to quickly see the average age of passengers on a particular cruise

To accomplish this requirement, you could create a view between `Passenger` and `Reservation`, and calculate the average age.

Note: There is no formal solution for this step.