ODC Associate Developer Certification - Sample Exam

Before Starting

This sample exam has 15 questions that will help you get ready for the ODC Associate Developer exam. We recommend that you prepare a real exam environment as much as possible.

- Find a quiet room just for you.
- Print this document, apart from the last page.
- Get a stopwatch or set a timer for the (recommended) duration of 35 minutes.

The last page of this document has the correct answers. Don't peek! Use it only after completing your exam, to check how well you did.

During the Sample Exam

To accurately simulate a real exam environment, we suggest that you:

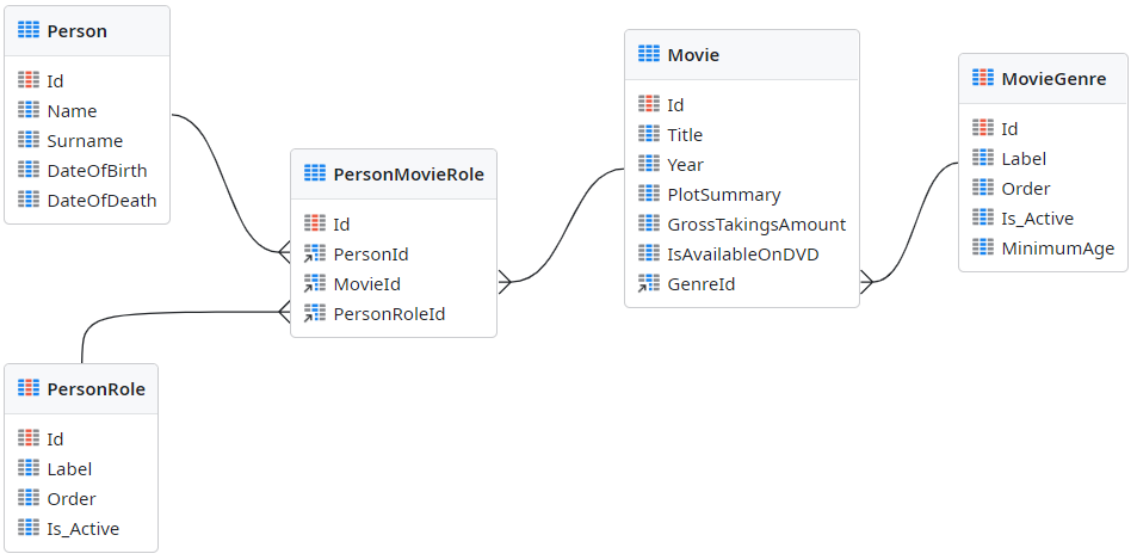
- Read each question and its answers carefully.
- Take your time! Questions may be revisited and your choices can be changed.
- Mark the questions that you want to review at the end.
- Pick only one answer per question, as only one is correct.
- Answer all questions, as there's no benefit in not doing so.
- Try turning off all electronic devices during the exam.
- Refrain from using or reading any external materials during the exam.

After Completing the Sample Exam

After completing the exam, validate the answers you selected by checking the ones provided on the last page of this document and count the total number of correct answers. Since the passing score is 70% or higher, you should get at least 11 questions right. In case you chose any wrong answers, we suggest you review the study materials where that specific topic is covered.

Sample Exam Questions

1. Considering the following Entity Diagram, which of the following options is **incorrect**?



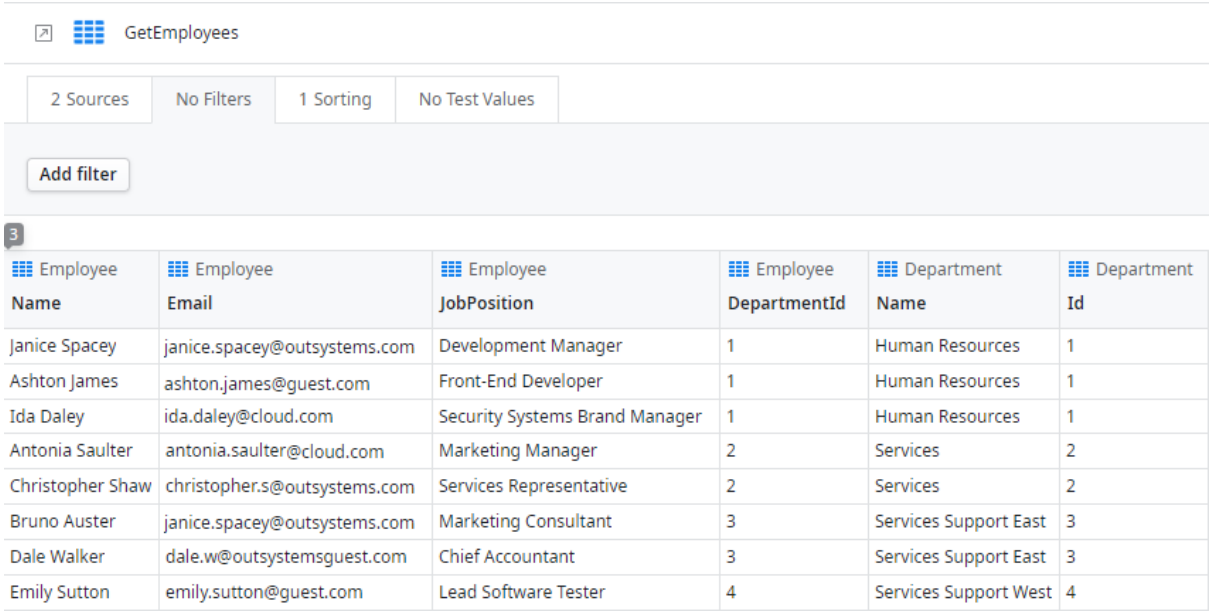
   A. The PersonMovieRole Entity record can only be associated with a single Movie record.

   B. A PersonRole record can be associated with several PersonMovieRole records.

   C. A Movie record can have several MovieGenres.

   D. A Person record can be associated with several Movies.

2. A developer wants to add a Filter to the *GetEmployees* Aggregate to guarantee that only Employees with a corporate email from OutSystems (ending with @outsystems.com) are returned. How can they create that Filter?



Employee Name	Employee Email	Employee JobPosition	Employee DepartmentId	Department Name	Department Id
Janice Spacey	janice.spacey@outsystems.com	Development Manager	1	Human Resources	1
Ashton James	ashton.james@guest.com	Front-End Developer	1	Human Resources	1
Ida Daley	ida.daley@cloud.com	Security Systems Brand Manager	1	Human Resources	1
Antonia Saulter	antonia.saulter@cloud.com	Marketing Manager	2	Services	2
Christopher Shaw	christopher.s@outsystems.com	Services Representative	2	Services	2
Bruno Auster	janice.spacey@outsystems.com	Marketing Consultant	3	Services Support East	3
Dale Walker	dale.w@outsystemsguest.com	Chief Accountant	3	Services Support East	3
Emily Sutton	emily.sutton@guest.com	Lead Software Tester	4	Services Support West	4

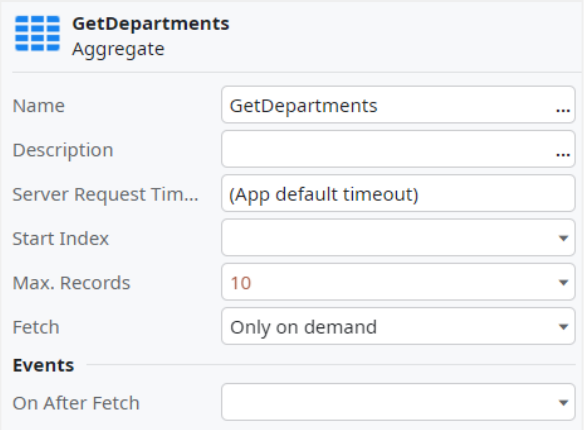
A. *Employee.Email = "%@outsystems.com"*

B. *Employee.Email like "@outsystems.com"*

C. *Employee.Email like "%@outsystems.com"*

D. *Employee.Email = "@outsystems.com"*

3. The following image shows the *GetDepartments* Aggregate with the Fetch property set to **Only on Demand**. When will the *GetDepartments* Aggregate be triggered?



A. The *GetDepartments* will be triggered after the information on the Screen is ready.

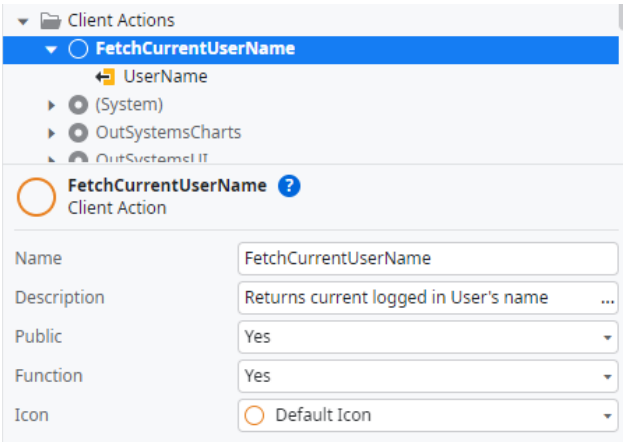
B. The *GetDepartments* will only run when explicitly triggered in the app's logic.

C. The *GetDepartments* will be triggered whenever anything on the Screen changes.

D. The *GetDepartments* will be triggered automatically when the Screen is initializing.

---

4. Consider the *FetchCurrentUserName* Client Action in the image below. Where can this Action be used?



A. It can be used in any other Client or Server Action.

B. It can only be used inside Expression widgets.

C. It can be used in any other Client Actions and on Expression widgets.

D. It can only be used inside Client Actions that are set as Functions.

---

5. Consider a Form with a *SaveOnClick* Action that allows saving information into the database. If the developer is not performing any custom validations, is it a good practice to check if the *Form.Valid* property is *True* before saving the data?

A. Yes. The *Built-In Validations* by themselves will not prevent the data from being saved to the database, even if there is invalid data added by the user.

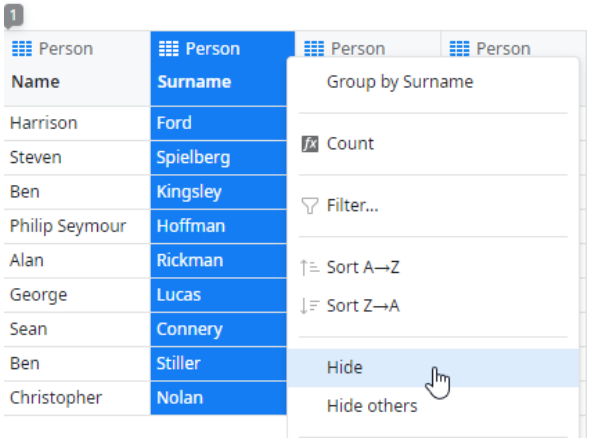
B. No. The *Valid* property of the Form only needs to be validated when the developer has defined custom validations.

C. Yes. The *Built-In Validations* do not check the data types of the Inputs, so this validation is important to guarantee that the Input values match the expected data types.

D. No. When the *Built-In Validations* fail, the *SaveOnClick* Action is not executed.

---

6. What happens if an attribute is hidden in the Aggregate's Editor window?



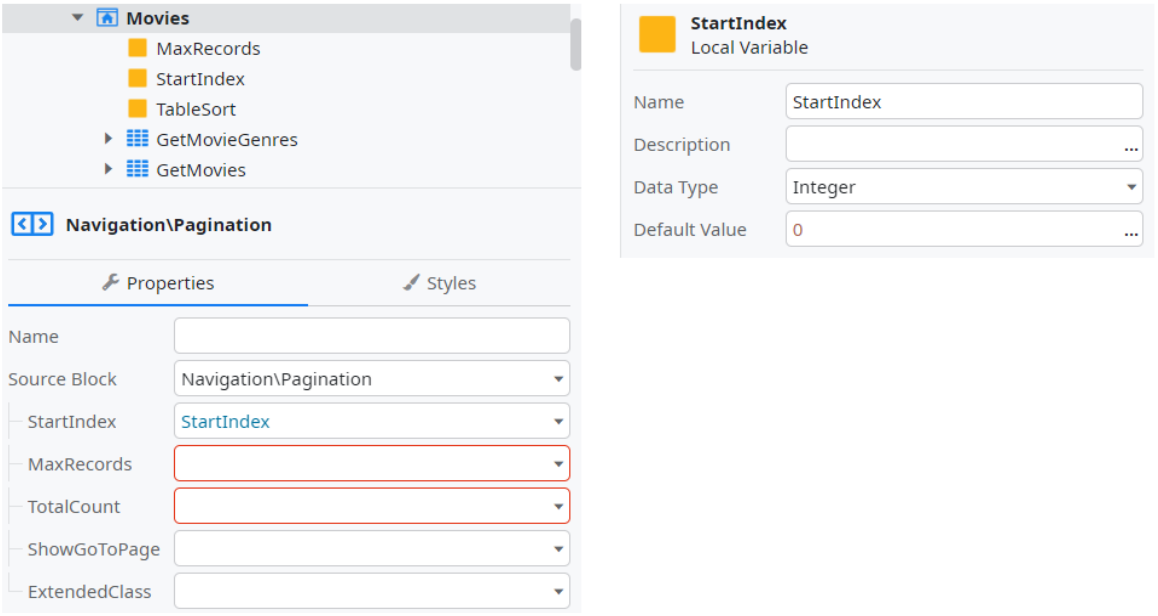
A. If the attribute is not being used in the app, the hidden column will not appear in the Aggregate Editor window and the corresponding attribute will not be available in the Aggregate's output.

B. If the attribute is being used in the app, the Aggregate will get an error since the corresponding attribute will not be available in the Aggregate's output.

C. The hidden attribute will not appear in the Aggregate Editor window and will not be available in the Aggregate's output, regardless if it is being used in the app or not.

D. The hidden attribute will not appear in the Aggregate Editor window, but it will still be available in the Aggregate's output, regardless if it is being used in the app or not.

7. The image below shows a *Pagination* widget that is being defined. What is the purpose of the ***StartIndex*** Local Variable?



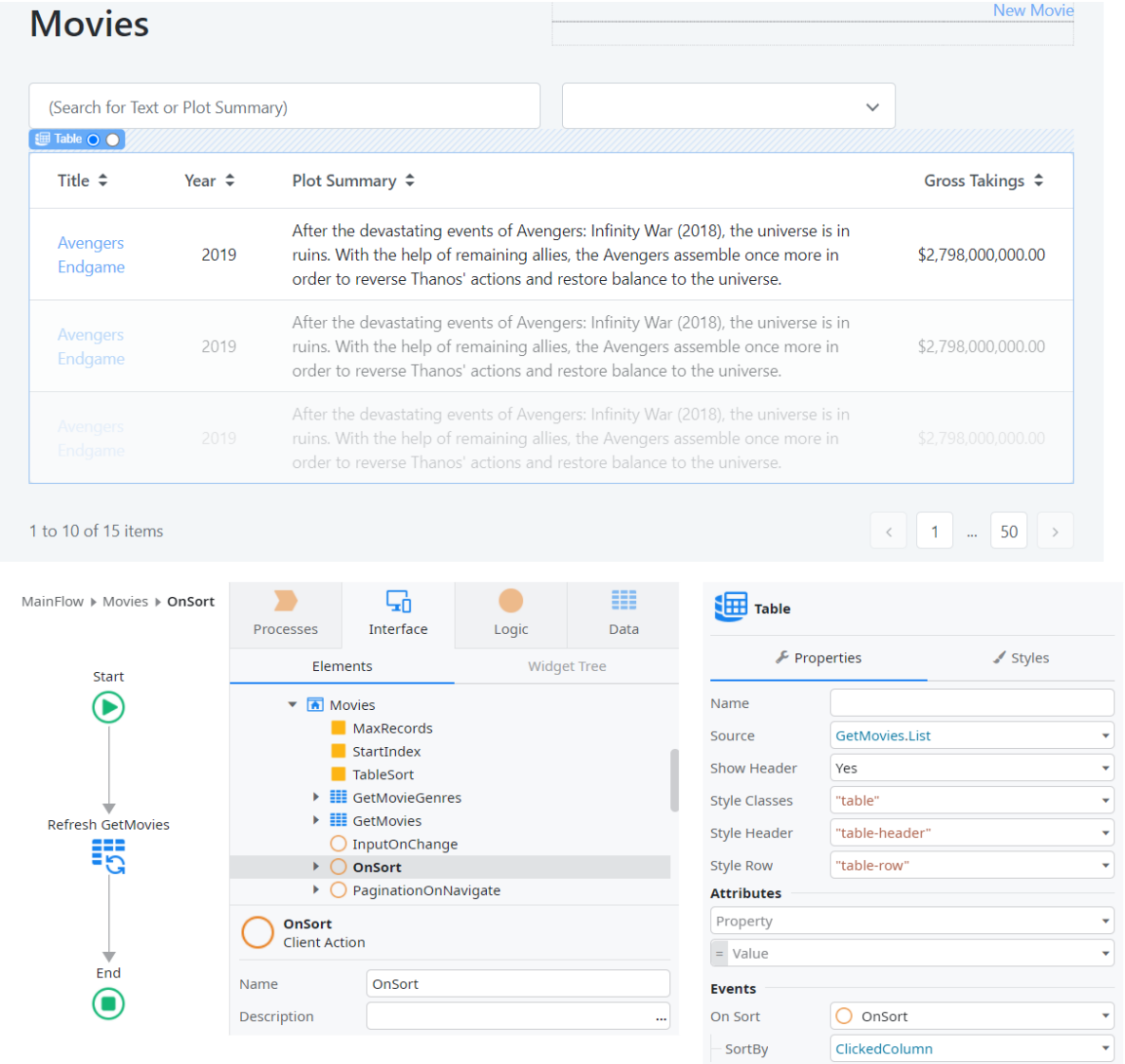
A. The *StartIndex* indicates the point where each page of the *Pagination* widget starts.

B. The *StartIndex* shows the total number of pages of records that the user will be able to navigate to.

C. The *StartIndex* points to the total number of records displayed across the pagination.

D. The *StartIndex* represents the number of records shown per page.

8. The image below shows the *Movies* Screen, which has a Table with movies that can be dynamically sorted. When the user clicks on a table column header, the movies should appear sorted by the respective attribute in the column. Whenever the user selects a new column to sort by, the *OnSort* Client Action is triggered and refreshes the list of movies. Considering this scenario, what is missing in the *OnSort* Action flow, if anything, to make sure the movies are sorted the way they should?



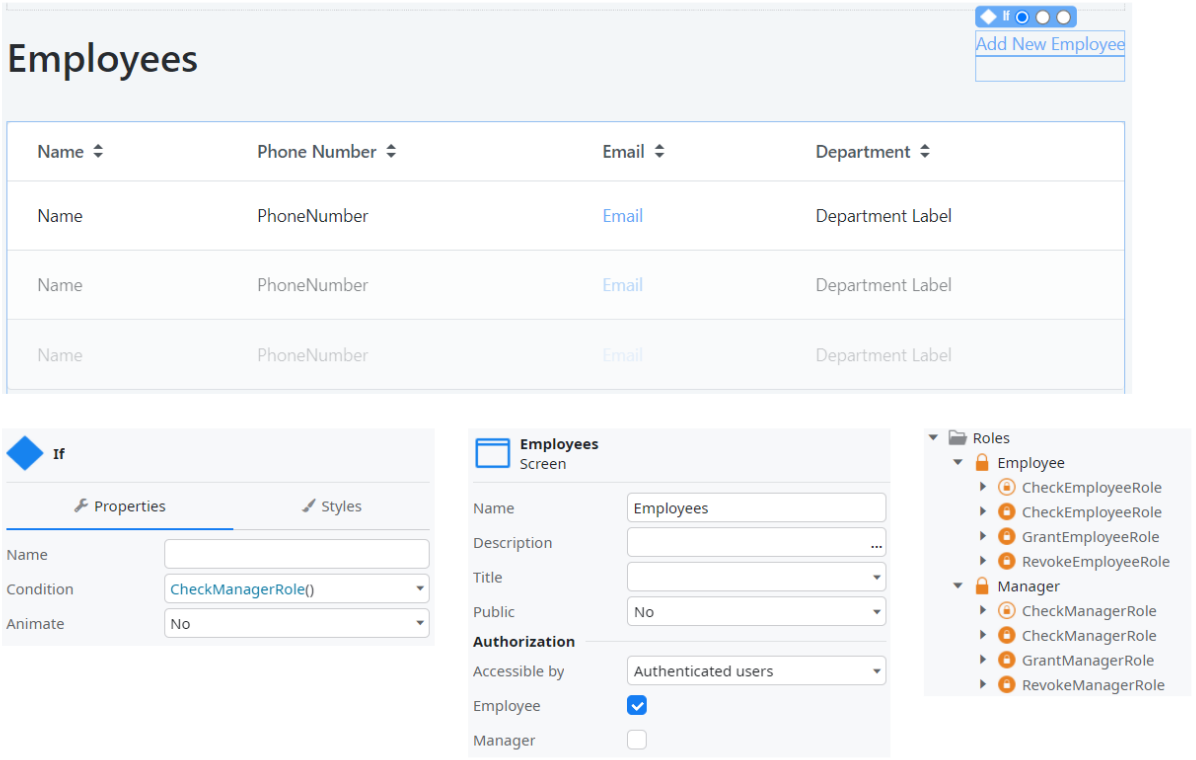
A. Add a Refresh Data element for the *GetMovieGenres* Aggregate.

B. Add an Assign below the Start with the following assignment: *TableSort = GetMovies.List.Current*.

C. Add an Assign below the Start with the following assignment: TableSort = SortBy.

D. Nothing else needs to be done. Refreshing the *GetMovies* Aggregate already guarantees that the data will be fetched with the new sorting criteria.

9. Consider an *Employee* Screen that should only be accessed by Managers and Employees. This Screen has a Link to the *EmployeeDetail* Screen that only users with the Manager role can access. The following image shows how this scenario is currently implemented. What do we need to change on this Screen, if anything, to ensure the expected behavior is properly implemented?



A. Nothing is wrong in this scenario and the logic is implemented correctly.

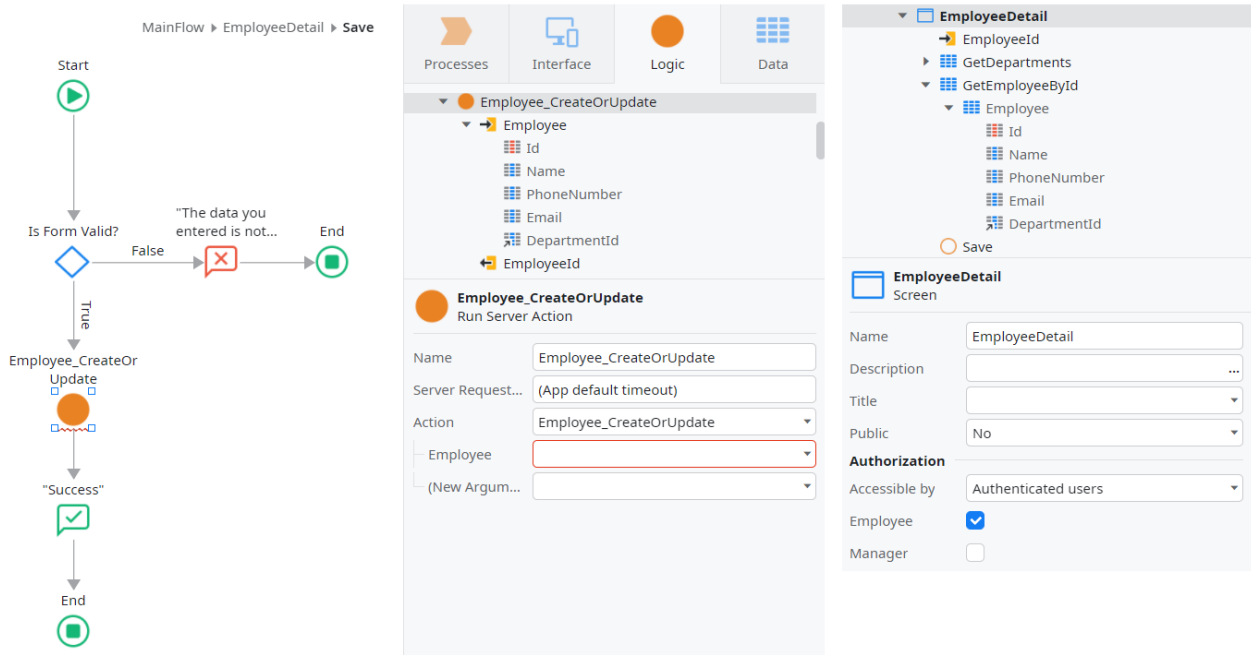
B. The condition of the *If* needs to be replaced by *GetUserId() <> NullIdentifier()*.

C. The condition of the *If* needs to be replaced by *CheckEmployeeRole()*.

D. The *Employee* Screen needs to have the Manager selected in the Authorization properties.

---

10. The following image shows the *Save* Client Action that allows creating or updating information about an Employee in the database. What should be the value of the *Employee* Input Parameter in the *Employee_CreateOrUpdate* Server Action?



A. *EmployeeId*

B. *GetEmployeeById.List.Current.Employee*

C. *GetEmployeeById.List*

D. *GetEmployeeById.List.Current.Employee.Id*

---

11. Which of the following options describes a common use case for the *On Initialize* Event?

A. Set the focus on an Input widget.

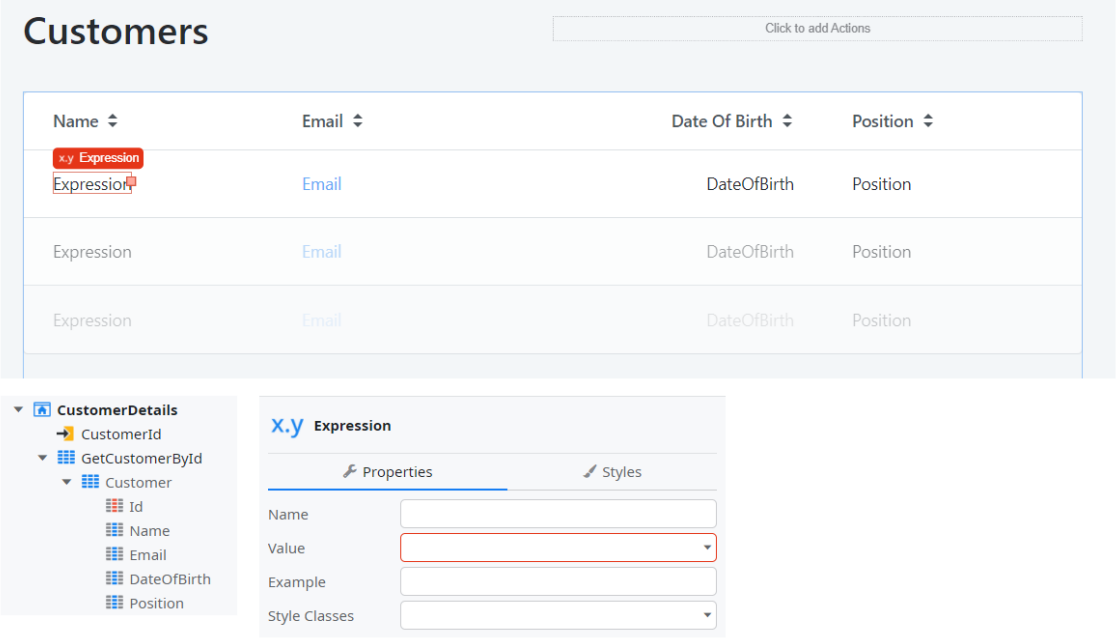
B. Clean the structure (DOM) of the Screen.

C. Define logic to act upon a change in data.

D. Set default values for Screen's Local Variables.

---

12. The following image shows the *CustomerDetails* Screen, which has an Expression that displays the customer's Name fetched by the *GetCustomerById* Aggregate. What should be the **Value** of the Expression?



A. *GetCustomerById.Name*

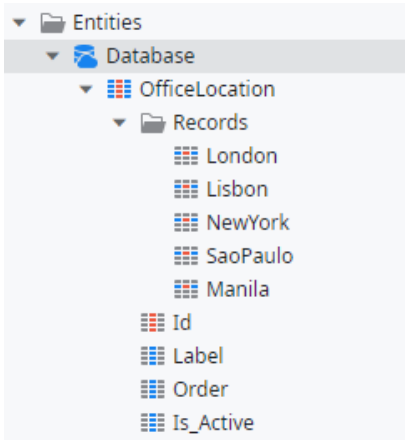
B. *GetCustomerById.List.Customer.Name*

C. *GetCustomerById.List.Current.Customer.Name*

D. *Customer.Name*

---

13. Considering the Entity shown in the image below, choose the **correct** option.



A. The Entity's records can be modified both at runtime, using the Create/Update Entity Actions, and at design time.

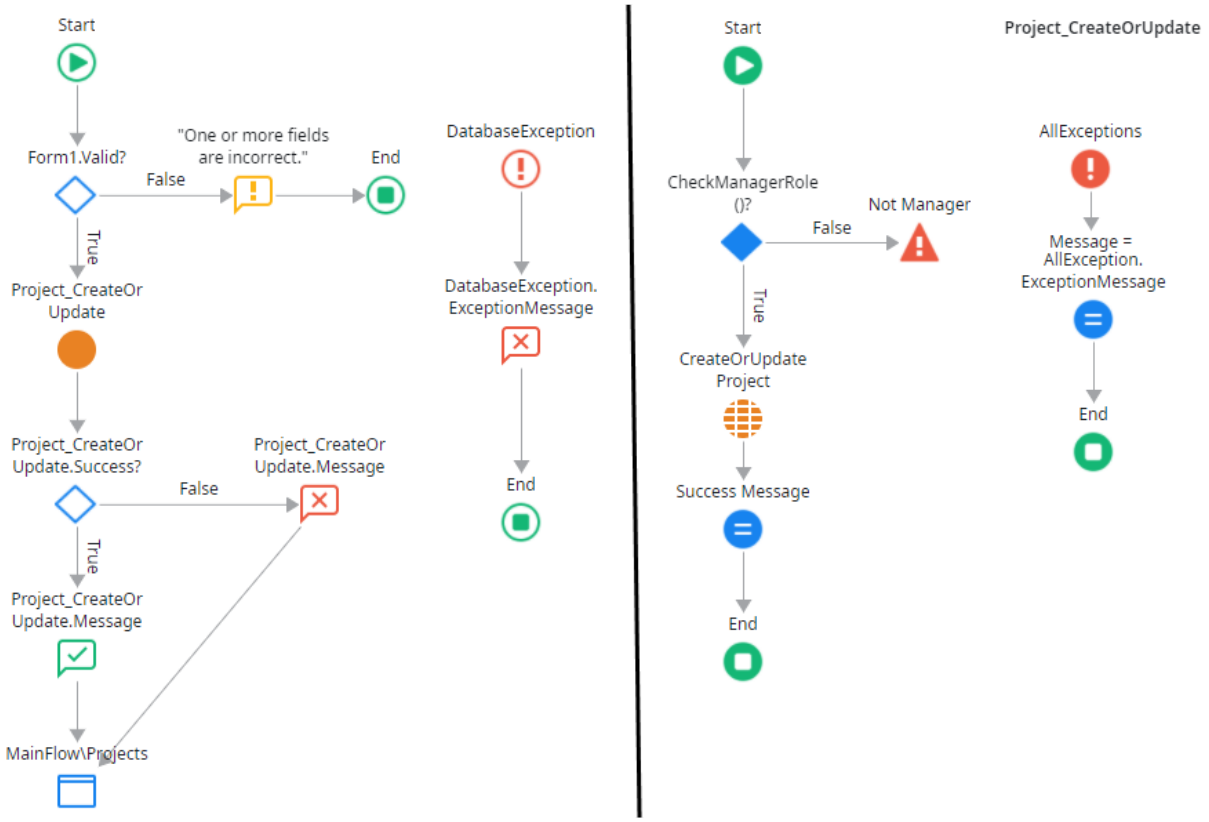
B. If the NewYork office had to be removed, this could only be done at design time.

C. The OfficeLocation Id attribute cannot be used to create relationships between this and other Entities.

D. It is not possible to delete the attributes in the OfficeLocation Entity since they are automatically created, but we can always add new ones.

---

14. The following image shows the *SaveOnClick* Client Action which is triggered when the end-user clicks on a *Save* Button to save a Project's details in the database. This Action calls the *Project_CreateOrUpdate* Server Action, where some of the necessary logic is implemented. What happens if an end-user without the Manager Role clicks on the *Save* Button?



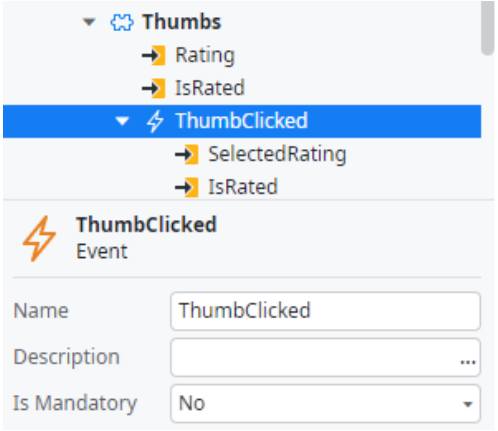
A. The warning message "One or more fields are incorrect" would be displayed to the end-user because the Form would be considered invalid.

B. The execution of the Action would proceed in the *DatabaseException* handler since the record could not be saved in the database.

C. The execution of the Action would proceed in the *AllExceptions* handler since this is the closest Exception handler that matches the *Not Manager* Exception.

D. The execution of the Action would end in the *Not Manager* Raise Exception, then continue in the *Global Exception* handler since there is no Exception handler in these Actions to handle the *Not Manager* Exception.

15. The *Thumbs* Block implements a thumbs up/thumbs down rating system where the *ThumbClicked* Event is triggered when an option is selected. Which of the following options is **correct**?



A. A Client Action to handle the Event must be created in every parent of the Block.

B. A Client Action to handle the Event can be created in the parents of the Block, but only where it makes sense to create some logic to respond to the Event.

C. A Client Action to handle the Event must be created inside the Block.

D. A Client Action to handle the Event can be created inside the Block if it is necessary to create some logic to respond to the Event.

Answers

1. C
2. C
3. B
4. C
5. A
6. D
7. A
8. C
9. D
10. B
11. D
12. C
13. B
14. C
15. B