

Using Outer Join to Select Actors - Full Solution

The screenshot shows a web browser with a course page titled 'Using outer join to select actors'. The page includes a Venn diagram with two overlapping circles labeled 'Left DataFrame' and 'Right DataFrame', with an arrow pointing to the intersection labeled 'Results'. Below the diagram are instructions for the exercise, including saving the merge of 'iron_1_actors' and 'iron_2_actors' to 'iron_1_and_2' and creating an index that returns True if 'name_1' or 'name_2' are null. To the right of the instructions is a code editor with a Python script that performs the merge and filtering. The script is as follows:

```
1 # Merge iron_1_actors to iron_2_actors on id with outer join using suffixes
2 iron_1_and_2 = iron_1_actors.merge(iron_2_actors, on='id', how='outer',
3                                     suffixes=('_1', '_2'))
4
5
6
7 # Create an index that returns True if name_1 or name_2 are null
8 m = ((iron_1_and_2['name_1'].isnull() | iron_1_and_2['name_2'].isnull())
9       (iron_1_and_2['_1'].isnull() | iron_1_and_2['_2'].isnull()))
10
11 # Print the first few rows of iron_1_and_2
12 print(iron_1_and_2[m].head())
```

The code editor also shows the output of the script, which displays the first few rows of the merged DataFrame where either 'name_1' or 'name_2' is null. The output is as follows:

```
iron_1_actors
  character  id  name
1  Vinson  17857  Shaun Toub
4  Virginia "Popper" Putts  12052  Gwyneth Paltrow

iron_2_actors
  character  id  name
4  Ian Yano / Whiplash  2295  Mickey Rourke
5  Natalie Rushan / Natasha Romanoff / Black Widow  1245  Scarlett Johansson
```

Screenshot showing the exercise context for using an outer join to select actors.

Code Answer:

```
# Merge iron_1_actors to iron_2_actors on id with outer join using suffixes
iron_1_and_2 = iron_1_actors.merge(iron_2_actors, on='id', how='outer',
                                   suffixes=('_1', '_2'))
```

```
# Create an index that returns True if name_1 or name_2 are null
m = iron_1_and_2['name_1'].isnull() | iron_1_and_2['name_2'].isnull()
```

```
# Print the first few rows of iron_1_and_2 where the condition is True
print(iron_1_and_2[m].head())
```

Explanation:

1. The `merge` function performs an outer join on the 'id' column between the 'iron_1_actors' and 'iron_2_actors' DataFrames. The `how='outer'` parameter ensures that all rows from both DataFrames are included, with missing values filled with NaN where matches are not found. The

``suffixes=('_1', '_2')`` parameter adds suffixes to distinguish columns from the two DataFrames.

2. The condition ``m`` identifies rows where either 'name_1' or 'name_2' is null, using the ``isnull()`` method combined with the logical OR operator (``|``). This step isolates rows corresponding to actors who appear in only one of the two movies.

3. The final step filters the 'iron_1_and_2' DataFrame using the condition ``m`` and displays the first few rows with ``print(iron_1_and_2[m].head())``. This allows verification of actors who are exclusive to one movie.