

# Performing a Semi Join in Pandas

The screenshot shows a web browser window displaying a DataCamp exercise page. The browser tabs include 'Performing a semi join | Python', 'docx maker - Right Join Movie', and 'Leantime Setup Ubuntu'. The address bar shows the URL: [campus.datacamp.com/courses/joining-data-with-pandas/advanced-merging-and-concatenating?ex=4](https://campus.datacamp.com/courses/joining-data-with-pandas/advanced-merging-and-concatenating?ex=4). The page title is 'Learn / Courses / Joining Data with pandas'. The exercise is titled 'Performing a semi join'. The instructions state: 'Some of the tracks that have generated the most significant amount of revenue are from TV-shows or are other non-musical audio. You have been given a table of invoices that include top revenue-generating items. Additionally, you have a table of non-musical tracks from the streaming service. In this exercise, you'll use a semi join to find the top revenue-generating non-musical tracks. The tables non\_mus\_ticks, top\_invoices, and genres have been loaded for you.' The instructions list four steps: 1. Merge non\_mus\_ticks and top\_invoices on tid using an inner join. Save the result as tracks\_invoices. 2. Use .isin() to subset the rows of non\_mus\_ticks where tid is in the tid column of tracks\_invoices. Save the result as top\_tracks. 3. Group top\_tracks by gid and count the tid rows. Save the result as cnt\_by\_gid. 4. Merge cnt\_by\_gid with the genres table on gid and print the result. A 'Take Hint (-30 XP)' button is visible. The code editor shows the following solution: 

```
1 # Merge the non_mus_ticks and top_invoices tables on tid
2 tracks_invoices = non_mus_ticks.merge(top_invoices, on='tid', how='inner')
3
4 # Use .isin() to subset non_mus_ticks to rows with tid in tracks_invoices
5 top_tracks = non_mus_ticks[non_mus_ticks['tid'].isin(tracks_invoices['tid'])]
6
7 # Group the top_tracks by gid and count the tid rows
8 cnt_by_gid = top_tracks.groupby(['gid'], as_index=False).agg({'tid': 'count'})
9
10 # Merge the genres table to cnt_by_gid on gid and print
11 print(cnt_by_gid.merge(genres, on='gid'))
```

 The Python Shell shows 'In [1]:'. The bottom of the screen shows a Windows taskbar with various icons and a system clock showing 20:56 on 03/12/2024.

Screenshot showing the exercise context for performing a semi join in pandas.

## Code Answer:

```
# Merge the non_mus_ticks and top_invoices tables on tid
tracks_invoices = non_mus_ticks.merge(top_invoices, on='tid', how='inner')

# Use .isin() to subset non_mus_ticks to rows with tid in tracks_invoices
top_tracks = non_mus_ticks[non_mus_ticks['tid'].isin(tracks_invoices['tid'])]

# Group the top_tracks by gid and count the tid rows
cnt_by_gid = top_tracks.groupby(['gid'], as_index=False).agg({'tid': 'count'})

# Merge the genres table to cnt_by_gid on gid and print
print(cnt_by_gid.merge(genres, on='gid'))
```

## Explanation:

1. The `merge` function performs an inner join between `'non_mus_ticks'` and `'top_invoices'` tables on the `'tid'` column, resulting in rows that exist in both tables. This step identifies the relevant non-musical tracks with matching invoices.

2. The `.isin()` method is then used to filter the rows in 'non\_mus\_tcks' where the 'tid' exists in the resulting 'tracks\_invoices' table, retaining only those rows. The filtered DataFrame is stored as 'top\_tracks'.
3. The `groupby` function is used to group the 'top\_tracks' DataFrame by 'gid' (genre ID) and count the number of 'tid' rows for each genre. This grouped DataFrame is stored as 'cnt\_by\_gid'.
4. Finally, the 'cnt\_by\_gid' DataFrame is merged with the 'genres' table on 'gid', enriching the grouped data with genre names, and the result is printed.