Python Summer Party Challenge

by Interview Master

Day 3 of 15

Disney

You are a data analyst working with the Disney Parks revenue team to understand nuanced guest spending patterns across different park experiences. The team wants to develop a comprehensive view of visitor purchasing behaviors. Your goal is to uncover meaningful insights that can drive personalized marketing strategies.

Challenge Questions

Q1:

What is the average spending per guest per visit for each park experience type during July 2024? Ensure that park experience types with no recorded transactions are shown with an average spending of 0.0. This analysis helps establish baseline spending differences essential for later segmentation.

Q2:

For guests who visited our parks more than once in August 2024, what is the difference in spending between their first and their last visit? This investigation, using sequential analysis, will reveal any shifts in guest spending behavior over multiple visits.

Q3:

In September 2024, how can guests be categorized into distinct spending segments such as Low, Medium, and High based on their total spending? Use the following thresholds for categorization:
-Low: Includes values from $0 up to, but not including, $50.
-Medium: Includes values from $50 up to, but not including, $100.
-High: Includes values from $100 and above.
 Exclude guests who did not make any purchases in the period.

Want to try this yourself?

Join the Challenge

Sign up for the Python Summer Party Challenge and solve 21 days of data science problems

www.interviewmaster.ai/python-party

Or keep scrolling to see my solutions

My Solution - Q1

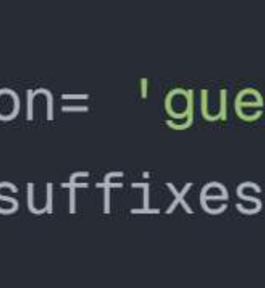
Day 3 Python Challenge

```
# Note: pandas and numpy are already imported as pd and np
# The following tables are loaded as pandas DataFrames with th
e same names: fct_guest_spending
# Please print your final result or dataframe
july_df = fct_guest_spending[(fct_guest_spending['visit_dat
e'].dt.year == 2024) &
                             (fct_guest_spending['visit_dat
e'].dt.month == 7)]

per_visit_df = july_df.groupby(['guest_id', 'visit_date', 'par
k_experience_type'])['amount_spent'].sum().reset_index()

avg_df = per_visit_df.groupby('park_experience_type')['amount_
spent'].mean().reset_index()

experience_types = fct_guest_spending['park_experience_type'].
dropna().unique()

experience_df = pd.DataFrame({'park_experience_type' : experie
nce_types})

final_df = pd.merge(experience_df, avg_df, on= 'park_experienc
e_type', how= 'left')

final_df['amount_spent'] = final_df['amount_spent'].fillna(0.
0)

final_df.rename(columns = {'amount_spent' : 'avg_spending_per_
guest_per_visit'}, inplace= True)
print(final_df)
```

My Solution - Q2

Day 3 Python Challenge

```
august_df = fct_guest_spending[(fct_guest_spending['visit_dat
e'].dt.year == 2024) &
                               (fct_guest_spending['visit_dat
e'].dt.month == 8)]

spending_df = august_df.groupby(['guest_id', 'visit_date'])['a
mount_spent'].sum().reset_index()

visits = spending_df['guest_id'].value_counts()

multi_visits = visits[visits > 1].index

multi_visit_df = spending_df[spending_df['guest_id'].isin(mult
i_visits)]

sorted_df = multi_visit_df.sort_values(['guest_id', 'visit_dat
e'])

first_visit = sorted_df.groupby('guest_id').first().reset_inde
x()
last_visit = sorted_df.groupby('guest_id').last().reset_index
()

spending_diff_df = pd.merge(first_visit[['guest_id', 'amount_s
pent']],
                            last_visit[['guest_id', 'amount_sp
ent']],
                           on= 'guest_id',
                           suffixes= ('_first', '_last'))

spending_diff_df['spending_diff'] = spending_diff_df['amount_s
pent_last'] - spending_diff_df['amount_spent_first']

print(spending_diff_df)
```



InterviewMaster.AI

My Solution - Q3

Day 3 Python Challenge

```
sept_df = fct_guest_spending[(fct_guest_spending['visit_dat
e'].dt.year == 2024) &
                             (fct_guest_spending['visit_date'].
dt.month == 9)]

spending = sept_df.groupby('guest_id')['amount_spent'].sum().r
eset_index()

spending = spending[spending['amount_spent'] > 0]

def category(amount):
  if amount < 50:
    return 'Low'
  elif amount < 100:
    return 'Medium'
  else:
    return 'High'

spending['category'] = spending['amount_spent'].apply(categor
y)

print(spending)
```

Ready for your own challenge?

Try this yourself by signing up for the Python Summer Party challenge:

www.interviewmaster.ai/python-party

InterviewMaster.AI