11/11 Questions Answered
Saved at 6:55 PM

TIME REMAINING
3:33

Final Exam

Q1 Pandas
18 Points

For this question, use only DataFrame or Series methods. You may not use any non-Pandas code, and you not use iteration.

Assume the following imports:

```
from pandas import Series, DataFrame
```

Q1.1
4 Points

Modify a DataFrame named **df** by replacing each missing value (NaN) with the previous non-NaN value in its column. Update **df**, do not assign to a result.

```
df.fillna(method = 'ffill', inplace = True)
```

Q1.2
6 Points

To a DataFrame named **df**, add a new column named **diff** in which each value is the difference between the maximum and minimum values in that row.

```
df['diff'] = df.max(axis = 1) - df.min(axis=1)
```

Q1.3
8 Points

Given a Series **ser** of numeric values that may repeat, find the value(s) that repeat the most number of times. Your result should be a Python list.

```
most_freq = ser.value_counts()[:1]
```

Q2 Pandas/NumPy
36 Points

For this question, you are required to use Pandas and NumPy, and the least amount of other Python code as may be absolutely needed.

Assume the following imports:

```
import pandas as pd
import numpy as np
from pandas import Series, DataFrame
```

A DataFrame **df** stores sales information for sedans sold in the US in 2020. The columns of **df** are 'Make', 'Model', 'Price', and 'Category', in that order. 'Price' is the only numerical column, the others are all non-numerical. The 'Category' column holds one of four values: 'HP' (high performance), 'LX' (luxury), 'MR' (mid-range), and 'EC' (economy).

Perform the following sequence of actions on **df**. If any of the actions do not directly update **df**, then make sure to reassign the result back to **df**.

Q2.1
6 Points

Remove rows from **df** that have missing values for *both* 'Price' and 'Category'.

```
index = df.index[df['Category'].isnull() & df['Price'].isnull()]
df.drop(index, 0, inplace=True)
```

Q2.2

13 Points

On the updated **df** after Q2.1, replace missing (NaN) values in the 'Price' column with the average price of cars in the same category. For instance, if there is a missing price in the category 'LX', replace it with the average price of cars in the 'LX' category.

```
avg = round(df[df.Category == 'HP']['Price'].mean(), 2)
df[df.Category == 'HP'] = df[df.Category == 'HP'].fillna({'Price': avg})
avg = round(df[df.Category == 'LX']['Price'].mean(), 2)
df[df.Category == 'LX'] = df[df.Category == 'LX'].fillna({'Price': avg})
avg = round(df[df.Category == 'MR']['Price'].mean(), 2)
df[df.Category == 'MR'] = df[df.Category == 'MR'].fillna({'Price': avg})
avg = round(df[df.Category == 'EC']['Price'].mean(), 2)
df[df.Category == 'EC'] = df[df.Category == 'EC'].fillna({'Price': avg})
```

Q2.3

17 Points

On the updated **df** after Q2.2, replace missing category values (NaNs) by basing category on price range. Assume that the maximum possible price for a category is the maximum of all prices currently listed in that category. You may assume that no two categories overlap in price range.
(Hint: You can check if an isolated non-numerical cell value is NaN with the Python function `isinstance(value,str)`, which returns `False` if value is NaN, `True` otherwise.)

```
for i in df.index:
    if(not isinstance(df.loc[i, 'Category'], str)):
        if (df[df.Category == 'HP']['Price'].min() <= df.loc[i, 'Price'] <=
df[df.Category == 'HP']['Price'].max()):
            df.loc[i, 'Category'] = 'HP'
        elif (df[df.Category == 'LX']['Price'].min() <= df.loc[i, 'Price'] <=
df[df.Category == 'LX']['Price'].max()):
            df.loc[i, 'Category'] = 'LX'
        elif (df[df.Category == 'MR']['Price'].min() <= df.loc[i, 'Price'] <=
df[df.Category == 'MR']['Price'].max()):
            df.loc[i, 'Category'] = 'MR'
        elif (df[df.Category == 'EC']['Price'].min() <= df.loc[i, 'Price'] <=
```

```
df[df.Category == 'EC']['Price'].max()):
        df.loc[i, 'Category'] = 'EC'
```

Q3 SQL Database Schema

30 Points

Design the schema (tables) for a bookstore database described below.

Each book has a title, one or more authors, a unique ISBN (International Standard Book Number, exactly 13 digits long), a publication date, a format (which could be either hardcover, paperback, ebook, or audiobook), a price, and number of copies sold. There could be more than one book with the same title.

The original format of any book is hardcover. In other words, a book is always available in hardcover, but it may or may not have been published in other formats. Each format of a book has a different ISBN, and the publication date, price, and copies sold would vary by format. The title and authors for a book are the same across all formats.

Each author has a name (first+last combined), unique numeric id (assigned by bookstore for accounting purposes), and email.

Each ISBN is published by a publishing company (uniquely named). Different formats (ISBNs) of the same book may have different publishers.

Write **create table** statements for all the tables in your schema. Be sure to choose the most appropriate/precise data type for each column, and whether it can be null or not. Be sure to specify primary key and unique columns as appropriate.

Your schema should be minimally redundant in storage of data, and should allow for effective querying.

```
create table Book (
    id int auto_increment primary key,
    title varchar(50) not null,
    authorid int not null,
    foreign key (authorid) references Author(id),
```

```
    isbnid int not null,
    foreign key (isbnid) references ISBN(id)
    formatid int not null,
    foreign key (formatid) references Format(id)
)
create table Format (
    id int auto_increment primary key,
    hardcover bool not null
    paperback bool
    ebook bool
    audiobook bool
    hardcover_isbn int not null
    paperback_isbn int
    ebook_isbn int
    audiobook_isbn int
    hardcover_pub date not null
    paperback_pub date
    ebook_pub date
    audiobook_pub date
    hardcover_price int not null
    paperback_price int
    ebook_price int
    audiobook_price int
    hardcover_copies int not null
    paperback_copies int
    ebook_copies int
    audiobook_copies int
)
create table Author (
    id int auto_increment primary key,
    author varcar(50) not null,
    auhtorid int not null,
    author_email varcar(50) not null
)
create table ISBN (
    id int auto_increment primary key,
    publisher_name varchar(50) not null,
)
```

Q4 SQL Database Queries

31 Points

You are given the following database schema of tables:

Hotel (HotelNo, HotelChainName, City, Country)
- HotelNo (hotel number) is the primary key
- A hotel chain may have multiple locations, even in the same city

Room (RoomNo, HotelNo, Type, Price)
- RoomNo is the room number
- HotelNo references HotelNo of the Hotel table
- The combination of RoomNo + HotelNo is unique

Guest(GuestNo, GuestName, GuestCity, GuestCountry)
- GuestNo (guest number) is the primary key
- GuestCity and GuestCountry identify where the guest is from

Booking (HotelNo, GuestNo, DateFrom, DateTo, RoomNo)
- Each row stores a guest booking for a room in a hotel
- The RoomNo+HotelNo combination for any row is present in the Room table
- GuestNo references GuestNo of the Guest table
- The combination of HotelNo, GuestNo, DateFrom, and RoomNo is unique
- Assume DateFrom and DateTo are within the same month, and they are each stored using the `date` type

Assume that all bookings have resulted in a stay for the entire duration of the booking (in other words, there were no cancellations or incomplete stays).

Note: Dates can be compared numerically for chronological ordering using the arithmetic relational operators such as <, <=, >, >=, and =. Also, you can use the SQL function `CURRENT_DATE()` to get the current date.

Using this schema, write SQL queries for each of the following.

Note: For partial credit, make sure that the foundational structure of your query is correct: choose the correct table(s), and if a join is needed, use the correct joining conditions for cross-table columns. If the foundation is incorrect, partial credit will be significantly impacted.

Q4.1

5 Points

What is the average price of a hotel room in London, England?

```
Select avg(r.Price) from Room r, Hotel h
Where r.HotelNo = h.HotelNo
&& h.City = "London"
&& h.Country = "England"
```

Q4.2

7 Points

How many different guests from New York, USA, have made bookings for the month of August for any year in any hotel?

```
Select count(Distinct g.GuestNO) from Booking b, Guest g
Where (b.DateFrom <= '8/31' and b.DateTo >= '8/1')
&& g.GuestCity = 'New York'
&& g.GuestCountry = 'USA'
```

Q4.3

10 Points

Get all rooms with corresponding hotel numbers that are currently unoccupied at the Hilton chain of hotels.

```
Select r.*, h.HotelNo from Room r, Hotel h
Where
(Select h.HotelNo, b.roomNo FROM Booking b, Hotel h
Where b.HotelNo = h.HotelNo
&& HotelChainName = 'Hilton'
&& b.dateFrom != CURRENT_DATE()
&& b.DateTo != CURRENT_DATE()
)
```

Q4.4

9 Points

Get the country-wise revenue for each hotel chain for the year 2021. The result should have columns named 'HotelChain', 'Country' and 'Revenue'

```
Select h.HotelChainName as HotelChain,
h.Country as Country, sum(r.Price) as Revenue
From Hotel h, Room r, Booking b
Where b.dateFrom >= '1/1/2021'
&& b.DateTo <= '12/31/2021'
```

Submit & View Submission