4 2023 CAPP Databases Midterm B

This exam was given to Master's level students in the University of Chicago CAPP program in the Spring of 2023. There were two versions of the exam – this is version "B".

The following table contains information about workers applying to a temp agency for data entry positions. You can assume that each name uniquely defines a person and that a person only appears once in the table.

- **name:** The name of the person (string, NOT NULL).
- **exp:** Years of experience (integer, NOT NULL).
- **wpm:** The number of words per minute (wpm) the person types (float, NOT NULL).
- **state:** The state that the worker is from (string, NOT NULL).
- **degree:** Highest educational attainment. You can assume this is all lower case (string, NOT NULL).
- **wage:** Preferred hourly wage (float, NOT NULL).
- **certificate:** If the person has a special certificate, such as for dealing with health care or bank data. If Null, this means that the person has no certificate. You can assume this is all lower case. A person can, AT MOST, have a single certificate (string, HAS NULLS).
- The name of the table / DataFrame is **agency**. No need to use a schema or load the DataFrame.
- Only use syntax covered in class.
- Interpret all inequalities as strict unless explicitly stated.

Figure D.2: *agency* Table: 12,345 Rows

name	exp	wpm	state	degree	wage	certificate
Ringly Roberson	7	68.5	NY	high-school	18.00	
Crash Bandicoot	14	88.0	MS	ba	22.00	healthcare
Alligator Reynolds	4	72.25	PA	ms	17.5	

SQL Section

Please answer the following questions making sure to return *only* the information requested.

1. Using SQL, write a query which returns the names (name only) of the 6 slowest typers (smallest wpm) who are from Pennsylvania ("PA").

```
SELECT
        name
FROM
        agency
where state = 'PA'
order by wpm desc
limit 6;
```

2. Using SQL, write a query which returns all workers (name and state) who have less than 10 years of experience, have no certificate and are either from Idaho ("ID") or California ("CA").

```
select name, state from agency where
        state in ('ID', 'CA')
        and exp < 10
        and certificate is null.
```

3. Write an SQL query which returns the number of workers from each state who have a certificate. This should be two columns: one with the state and the other with the number of workers from that state who have a certificate.

```
SELECT
        state, count(1) as ct
from
        agency
where certificate is not null
group by 1;
```

4. Write an SQL query which returns all rows and columns for workers whose degree is equal to "high-school". You can assume that all degrees are lower case. Sort the data by words per minute from highest to lowest.

```
select * from agency
where degree = 'high-school'
order by wpm desc;
```

5. We are analyzing degrees and certificates. We want to figure out which degrees have more than 20% of their workers with a certificate. Write an SQL query which returns one column. In this column should be a list of degrees where more than 20% of the workers with that degree have a certificate.

```
select degree from
        (select degree
                , sum( case when certificate is not null then 1 else 0 end)::float / sum(1) as rat
        from agency
        group by 1 ) as innerQ
where rat >= .2
```

6. We calculate the dollar-adjusted wpm ("DAWPM") by taking a worker's WPM, squaring it and dividing it by their wage (in pennies) ($\frac{wpm^2}{100 \cdot wage}$). Write a query which returns three columns: name, degree and the DAWPM.

```
select (wpm * wpm) / (100 * wage) as DAWPM, name, degree
from agency;
```

7. Write a query which returns three columns: name, degree, and DAWPM_Flag. DAWPM_Flag should be equal to 0 if the DAWPM is less than or equal to 3, 1 if the DAWPM is greater than 3 and less than or equal to 10 and 2 otherwise. DAWPM is defined in the previous problem.

```
select
        name, degree
        , case
                when (wpm * wpm) / (100 * wage)   <= 3 then 0
                when (wpm * wpm) / (100 * wage)  <= 10 then 1
                else 2 end as DAWPM_Flag
from
        agency;
```

8. If a person's DAWPM is greater than or equal to 3 they are defined as "hyper-efficient". Write a query which returns the *percentage* of workers of each degree who are hyper-efficient. This should have two columns: degree and percentage of the workers with that degree who are hyper-efficient.

```
select
       degree,
       sum( case when (wpm * wpm) / (100 * wage) >= 3 then 1 else 0 end )::float / count(1) as pctEff
from
       agency
group by 1;
```

9. Using SQL, write a query which returns one row and two columns. The first column should be the number of workers who have exactly 3 years experience and have a "ba" degree (call this column ba3). The second column should be the number of workers who have exactly 7 years of experience and have an "ms" degree (call this column ms7).

```
select
        sum( case when degree = 'ba' and exp = 3 then 1 else 0 end) as ba3
        , sum( case when degree = 'ms' and exp = 7 then 1 else 0 end) as ms7
from agency;
```

Pandas Section

Please answer the following question, making sure to return only the information required. You can assume that a DataFrame named *agency* is already loaded. If a specific output is not specified you can return anything (DataFrame/Series/List/Array/etc.)

1. Using Pandas, return the name (as a Series) of the top 6 fastest typers (largest wpm) who have 3 years of work experience.

```
agency.loc[(agency.loc[:, 'exp'] == 3), :].nlargest(6, wpm).loc[:, 'name']
```

   OR:

```
(agency
        .loc[(agency.loc[:, 'exp'] == 3), :]
        .sort_values( 'wpm', ascending=False)
        .head(6)
        .loc[:, 'name']
)
```

2. Using Pandas, return a DataFrame with two columns: name and state. The dataset should only contain workers that are either (a) over 10 years experience and from Georgia ("GA") or (b) under

8 years experience and from Nevada ("NV").

```
agency.loc[((agency.loc[:, 'exp'] > 10) & (agency.loc[:, 'state'] == 'GA') )
        | ((agency.loc[:, 'exp'] < 8) & (agency.loc[:, 'state'] == 'NV') )
        , ['name', 'state']]
```

3. Using Pandas, return a **DataFrame** which contains all workers (name only) who have less than 9 years experience and are either from New Mexico ('NM') or Texas ('TX'). Only include those workers who have a certificate (any certificate).

```
agency.loc[(agency.loc[:, 'exp'] < 9)
        & ~(agency.loc[:, 'certificate'].isna())
        & (agency.loc[:, 'state'].isin( ['NM', 'TX']))
        , ['name']]
```

4. Using Pandas, return all states (this should be without duplicates) which have a worker with a "healthcare" certificate. You can assume that all certificate names are lower case.

```
agency.loc[ (agency.loc[:, 'certificate'] == 'healthcare'), 'state'].unique()
```

5. Return all rows and columns for workers who have a "doctorate" degree (you can assume that all degrees are lower case). Sort the resulting DataFrame first by state (alphabetically) and then, within state, from fastest to slowest wpm.

```
(agency.loc[ (agency.loc[:, 'degree'] == 'doctorate'), :]
        .sort_values( ['state', 'wpm'], ascending=[True, False])
        )
```

6. Please return a DataFrame which has all the original data and adds a column called "DAWPM" which is the dollar-adjusted words per minute (as in the other problems it is defined as words per minute squared divided by wages in pennies: $\frac{wpm^2}{100 \cdot wage}$).

```
agency.loc[:, 'dawpm'] = (agency.loc[:, 'wpm'] * agency.loc[:, 'wpm']) / (100.0 * agency.loc[:,  'wage'])
```

7. Please return a DataFrame which has all the original data as well as adds a column called "exp_flag" which is equal to 0 if the worker has less than 10 years experience, 1 if they have greater than or equal to 10 years and less than 20 years experience and 2 otherwise.

```
agency.loc[:, 'exp_flag'] = 0
agency.loc[  (agency.loc[:, 'exp'] >= 10) & (agency.loc[:, 'exp'] < 20), 'exp_flag'] = 1
agency.loc[ (agency.loc[:, 'exp'] > 20), 'exp_flag'] = 2
```

8. There was an error and workers from Maryland ("MD") had their WPM recorded as 5 too large. Please return an updated DataFrame which has this error fixed. Specifically the DataFrame should have all rows and columns, but the `wpm` column should have this error fixed.

```
agency.loc[ (agency.loc[:, 'state'] == 'MD'), 'wpm'] = agency.loc[ (agency.loc[:, 'state'] == 'MD'), 'wpm'] - 5
```