PROJECT: EXPLORING TRENDS IN AMERICAN BABY NAMES

How have American baby name tastes changed since 1920? Which names have remained popular for over 100 years, and how do those names compare to more recent top baby names? These are considerations for many new parents, but the skills you'll practice while answering these queries are broadly applicable. After all, understanding trends and popularity is important for many businesses, too!

You'll be working with data provided by the United States Social Security Administration, which lists first names along with the number and sex of babies they were given to in each year. For processing speed purposes, the dataset is limited to first names which were given to over 5,000 American babies in a given year. The data spans 101 years, from 1920 through 2020.

The Data

`baby_names`

column	type	description
`year`	int	year
`first_name`	varchar	first name
`sex`	varchar	`sex` of babies given `first_name`
`num`	int	number of babies of `sex` given `first_name` in that `year`

Projects Data DataFrame as `usa_baby_names`

```
-- Run this code to view the data in baby_names
SELECT *
FROM baby_names;
```

index	year	first_name	sex
0	1920	Mary	F
1	1920	Dorothy	F
2	1920	Helen	F
3	1920	Margaret	F
4	1920	Ruth	F
5	1920	Mildred	F
6	1920	Virginia	F
7	1920	Elizabeth	F
8	1920	Frances	F
9	1920	Anna	F
10	1920	Betty	F
11	1920	Evelyn	F
12	1920	Marie	F
13	1920	Doris	F
14	1920	Alice	F
15	1920	Florence	F

Rows: 12,649 Expand Table

Projects Data DataFrame as `name_types`

```
-- List the first five names in alphabetical order and find out if each name is "Classic" or "Trendy." Save your query as a
DataFrame name_types with three columns: first_name, sum, and popularity_type.
SELECT first_name, SUM(num),
    CASE WHEN COUNT(year) > 50 THEN 'Classic'
        ELSE 'Trendy' END AS popularity_type -- Classify first names as 'Classic' or 'Trendy'
FROM baby_names
GROUP BY first_name -- Group by first_name to use aggregate functions
ORDER BY first_name -- Order the results alphabetically by first_name
LIMIT 5;-- Limit to the first 5 names
```

index	first_name	sum	popularity_type
0	Aaliyah	15870	Trendy
1	Aaron	530592	Classic
2	Abigail	338485	Trendy
3	Adam	497293	Trendy
4	Addison	107433	Trendy

Rows: 5 Expand Table

Projects Data DataFrame as top_20

```sql
-- What were the top 20 male names overall, and how did the name Paul rank? Save your query as a DataFrame top_20 with three columns: name_rank, first_name, and sum.
SELECT
    RANK() OVER(ORDER BY SUM(num) DESC) AS name_rank,
    first_name, SUM(num)
FROM baby_names
WHERE sex = 'M' -- Filter the data for results where sex equals 'M'
GROUP BY first_name -- Group by first name, oder by rank, and limit to the top 20
ORDER BY name_rank
LIMIT 20;
```

	n...	firs...	...
0	1	James	47481...
1	2	John	45107...
2	3	Robert	44951...
3	4	Michael	42788...
4	5	William	36144...
5	6	David	35714...
6	7	Richard	24148...
7	8	Joseph	23613...
8	9	Thomas	21668...
9	10	Charles	21123...
10	11	Christopher	20127...
11	12	Daniel	18242...
12	13	Matthew	15672...
13	14	Anthony	13443...
14	15	Donald	12802...
15	16	Mark	12659...

Rows: 20 Expand Table

Projects Data DataFrame as a

```sql
-- Which female names appeared in both 1920 and 2020? Save your query as a DataFrame a_names with two columns: first_name, and total_occurrences.
SELECT a.first_name, (a.num + b.num) AS total_occurrences
FROM baby_names a
JOIN baby_names b
ON a.first_name = b.first_name -- Join on first name
WHERE a.year = 1920 AND a.sex = 'F'
AND b.year = 2020 AND b.sex = 'F'; -- Filter for the years 1920 and 2020 and sex equals 'F'
```

	fi...	total_occurre...
0	Emma	20818
1	Evelyn	23283
2	Elizabeth	23125
3	Eleanor	14832
4	Grace	12741
5	Hazel	12765

Rows: 6 Expand Table