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
LIMIT 5;
```

index	year	first_name	sex	num
0	1920	Mary	F	70982
1	1920	Dorothy	F	36643
2	1920	Helen	F	35097
3	1920	Margaret	F	27994
4	1920	Ruth	F	26101

Rows: 5

Projects Data DataFrame as name_types

```sql
-- Use this table for the answer to question 1:
-- List the overall top five names in alphabetical order and find out if each name is "Classic" or "Trendy."
SELECT DISTINCT first_name, COUNT(first_name) AS sum,
        CASE WHEN COUNT(first_name) = 50 OR COUNT(first_name) > 50 THEN 'Classic'
        ELSE 'Trendy'
        END AS popularity_type
FROM baby_names
GROUP BY first_name
ORDER BY first_name ASC
LIMIT 5;
```

index	first_name	sum	popularity_type
0	Aaliyah	3	Trendy
1	Aaron	51	Classic
2	Abigail	28	Trendy
3	Adam	46	Trendy
4	Addison	13	Trendy

Rows: 5

Projects Data DataFrame as top_2

```sql
-- Use this table for the answer to question 2:
-- What were the top 20 male names overall, and how did the name Paul rank?
SELECT DENSE_RANK() OVER(ORDER BY COUNT(first_name) DESC) AS name_rank, first_name,
                COUNT(first_name) AS sum
FROM baby_names
GROUP BY first_name
LIMIT 20;
```

index	name_rank	first_name	sum
0	1	David	101
1	1	Charles	101
2	1	James	101
3	1	Thomas	101
4	1	William	101
5	1	Joseph	101
6	1	John	101
7	1	Elizabeth	101
8	2	Robert	99
9	3	Anthony	86
10	4	Michael	85
11	5	Richard	84
12	5	Mary	84
13	5	Daniel	84
14	6	Kenneth	76
15	6	Paul	76
16	7	Edward	73

Rows: 20

Projects Data DataFrame as a

```sql
SELECT a.first_name, (a.num + b.num) AS total_occurrences
FROM baby_names a
JOIN baby_names b
-- Join on first name
ON a.first_name = b.first_name
-- Filter for the years 1920 and 2020 and sex equals 'F'
WHERE a.year = 1920 AND a.sex = 'F'
AND b.year = 2020 AND b.sex = 'F';
```

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

Rows: 6