Advanced Select-Occupations-Pivot

Q. Pivot the *Occupation* column in **OCCUPATIONS** so that each *Name* is sorted alphabetically and displayed underneath its corresponding *Occupation*. The output should consist of four columns (*Doctor*, *Professor*, *Singer*, and *Actor*) in that specific order, with their respective names listed alphabetically under each column.

Note: Print **NULL** when there are no more names corresponding to an occupation.

Input Format

The **OCCUPATIONS** table is described as follows:

Column	Type
Name	String
Occupation	String

Occupation will only contain one of the following values: **Doctor**, **Professor**, **Singer** or **Actor**.

Name	Occupation
Samantha	Doctor
Julia	Actor
Maria	Actor
Meera	Singer
Ashely	Professor
Ketty	Professor
Christeen	Professor
Jane	Actor
Jenny	Doctor
Priya	Singer

Sample Output

```
Jenny    Ashley     Meera  Jane
Samantha Christeen  Priya  Julia
NULL     Ketty      NULL   Maria
```

Explanation

The first column is an alphabetically ordered list of Doctor names.

The second column is an alphabetically ordered list of Professor names.

The third column is an alphabetically ordered list of Singer names.

The fourth column is an alphabetically ordered list of Actor names.

The empty cell data for columns with less than the maximum number of names per occupation (in this case, the Professor and Actor columns) are filled with **NULL** values.

- Create a table

```
CREATE TABLE OCCUPATIONS  (
   Name VARCHAR(255),
   Occupation VARCHAR(255)
);
```

```
INSERT INTO OCCUPATIONS (Name, Occupation) VALUES
('Samantha', 'Doctor'),
('Julia', 'Actor'),
('Maria', 'Actor'),
('Meera', 'Singer'),
('Ashely', 'Professor'),
('Ketty', 'Professor'),
('Christeen', 'Professor'),
('Jane', 'Actor'),
('Jenny', 'Doctor'),
('Priya', 'Singer');

select* from OCCUPATIONS;
```

Name	Occupation
Samantha	Doctor
Julia	Actor
Maria	Actor
Meera	Singer
Ashely	Professor
Ketty	Professor
Christeen	Professor
Jane	Actor
Jenny	Doctor
Priya	Singer

- **Partition by Occupation & order by Name:**

```
SELECT *, ROW_NUMBER()  OVER (PARTITION BY Occupation ORDER BY Na
me) AS RN
FROM OCCUPATIONS
;
```

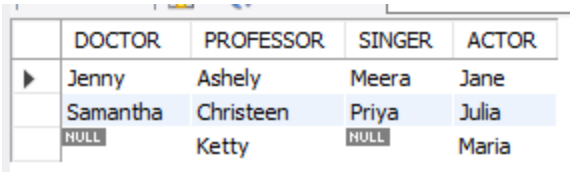
	Name	Occupation	RN
▶	Jane	Actor	1
	Julia	Actor	2
	Maria	Actor	3
	Jenny	Doctor	1
	Samantha	Doctor	2
	Ashely	Professor	1
	Christeen	Professor	2
	Ketty	Professor	3
	Meera	Singer	1
	Priya	Singer	2

- We are getting columns grouped by Occupation, in ASC order
- + We got the **Row numbers**

- Now, we have to convert **rows into → columns**
- **Use PIVOT for the same**
  - **BUT PIVOT doesn't work in MySQL**

- So, first we have to perform GROUP BY RN
  - For that use Subquery and MAX/MIN window function
- It doesn't matter if we use MIN OR MAX. We are just printing the names from above table

```
Select
Min(IF (OCCUPATION = "DOCTOR",NAME,NULL)) AS DOCTOR,
MAX(IF (OCCUPATION = "PROFESSOR",NAME,NULL)) AS PROFESSOR,
Min(IF (OCCUPATION = "SINGER",NAME,NULL)) AS SINGER,
Min(IF (OCCUPATION = "ACTOR",NAME,NULL)) AS ACTOR
FROM
(select name,occupation,Row_number() Over (PARTITION BY occupation ORD
ER BY name) as RN
FROM occupations) as t
group by RN;
```



DOCTOR	PROFESSOR	SINGER	ACTOR
Jenny	Ashely	Meera	Jane
Samantha	Christeen	Priya	Julia
NULL	Ketty	NULL	Maria

- `IF (OCCUPATION = "DOCTOR",NAME,NULL)` → if Occupation is Doctor, print the name, Else→ Print NULL
- You use `MIN` or `MAX` on these because when you group by a number, there is only one doctor (or professor, or singer, or actor) in that group.

`MIN()` **or** `MAX()` is used to "pick" the name from the group.

- Since there's **only one name per RN group** (e.g., only one Doctor with `RN=1`), `MIN()` or `MAX()` will grab that name.
- Example for Row 1:

Doctor	Professor	Singer	Actor
Amy	Alice	Eve	Frank

What Happens When We "GROUP BY RN":

Grouping by **RN** means we take all rows from the temporary table "t" that have the same row number and group them together. Let's see what those groups look like without considering the IF functions.

Group for RN = 1:

RN	Name	Occupation
1	Aamina	Doctor
1	Ashley	Professor
1	Eve	Actor
1	Christeen	Singer

Group for RN = 2:

RN	Name	Occupation
2	Julia	Doctor

RN	Name	Occupation
2	Belvet	Professor
2	Jennifer	Actor
2	Jane	Singer

How the IF Functions Work with GROUP BY

Now the outer query applies the IF functions on each group:

- For each group, it checks every row:
  - For example, in Group RN = 1:
    - `IF(occupation = "DOCTOR", name, NULL)` returns “Aamina” for the Doctor row and NULL for others.
    - `MIN(...)` then gives “Aamina” for the DOCTOR column.
  - Similarly:
    - `IF(occupation = "PROFESSOR", name, NULL)` returns “Ashley” for the Professor row.
    - `IF(occupation = "ACTOR", name, NULL)` returns “Eve” for the Actor row.
    - `IF(occupation = "SINGER", name, NULL)` returns “Christeen” for the Singer row.
- This process is repeated for each RN group, so the final output table is “pivoted” by row number.

Final Result Table:

Based on our groups, the final output becomes:

DOCTOR	PROFESSOR	ACTOR	SINGER
Aamina	Ashley	Eve	Christeen
Julia	Belvet	Jennifer	Jane
Priya	Britney	Ketty	Jenny
NULL	Maria	Samantha	Kristeen
NULL	Meera	NULL	NULL
NULL	Naomi	NULL	NULL

DOCTOR	PROFESSOR	ACTOR	SINGER
NULL	Priyanka	NULL	NULL