Answers

Aggregate operators and subqueries

Handout: aggregate functions in select.2.21.20.pdf

(markdown converted at https://md2pdf.netlify.com/)

Editor’s Database

UName(**id**,name) one name per id
Email(**id**,email)
Papers(**narid**, *authorid*, title, year, keywords, decision)
Reviewers(**narid**, *reviewerid*, agree_decline, days, rating, year)

Table names are **bold**. **Primary keys** are **bold**. *Foreign keys* are shown in *italics* and match the same field in another table unless specified.

Notes:

1. *authorid* in **Papers** refers to **id** in **Name** and **Email**
2. *reviewerid* in **Reviewers** refers to **id** in **Name** and **Email**

Questions

1. Number of papers. Find the total number of papers submitted across all years (total).

```
select count(title)
from Papers

select count(narid)
from Papers

select count(year)
from Papers

select count(*)
from Papers
```

2. Number of papers by year. Find the number of submitted papers each year (year, count).

```
select count(*), year
from Papers
group by year

#or

select year, count(narid)
from Papers
group by year
```

3. Number of years in database. Find the number of years covered by the papers.

```
select count(distinct year)
from Papers
```

```
select distinct year
from Papers
```

4. Average number of papers per year across all years. Find the average number of papers per year, considering all years (average).

```
select avg(total)
from (select count(narid) as total
                from papers
                group by year) as x

select count(title)/count(distinct year)
from Papers

select (select count(narid) from Papers)/(select count(distinct year) from Papers)
```

5. Prolific authors. Find the authors who have submitted 4 or more papers (id, name, count). List by number of papers descending. Find authors rejected more than 2 times.

```
#four or more papers submitted
select id, name, count(narid) as c
from Uname join Papers on id=authorid
group by authorid
having c >= 4
order by c desc

#four or more papers not rejected
select id, name, count(narid) as c
```

```
from Uname join Papers on id=authorid
where finaldecision <> "reject"
group by authorid
having c >= 4
order by c desc

#two or more papers rejected
select id, name, count(narid) as c
from Uname join Papers on id=authorid
where finaldecision = "reject"
group by authorid
having c >= 2
order by c desc
```

6. Most frequently asked to review. Find the people who have been asked to review 3 or more times (id, name, count). List by number of papers descending. Find those only in the last three years (excluding this year).

```
#asked to review three or more times
select id, name, count(agree_decline) as c
from Reviewers join Uname on id=reviewerid
group by reviewerid
having c > 2
order by c desc

#asked three or more time in the last three years
where year in (2017, 2018, 2019)
```

7. Most frequently returned reviews. Find the people who have returned reviews 3 or more times (id, name, count). List by number of papers descending. Determine that a review is returned by testing: rating is not null.

```
#returned a review three or more times
select id, name, count(narid) as c
from Reviewers join Uname on id=reviewerid
where rating is not null
group by reviewerid
having c > 2
order by c desc

#same, uses fact that count of named field only counts non-null values
select id, name, count(rating) as c
from Reviewers join Uname on id=reviewerid
group by reviewerid
having c > 2
order by c desc
```

- Limit to only those reviews that are rated good or excellent and add the average number of days for the review and the standard deviation in the number of days. (id, name, count, avg days, std days)

```
#
select id, name, count(narid) as c, avg(days), std(days)
from Reviewers join Uname on id=reviewerid
where rating in ("Good", "excellent")
group by reviewerid
having c > 2
order by c desc
```

- For each person, produce up to two rows of output, one for good and one for excellent rating and include rating field in the output where the sum of both types of reviews is at least 4.

```
#
select id, name, count(narid) as c, avg(days) as d, std(days) as s, rating
from Reviewers join Uname on id=reviewerid
where rating in ("Good", "excellent")
and id in (select reviewerid
              from Reviewers
              where rating in ("Good", "excellent")
              group by reviewerid
              having count(narid) >= 4)
group by reviewerid, rating
```

8. More than one review per year. Find people who did more than one review in the same year (id, name, year).

```
select reviewerid, name, year, count(rating) as c
from reviewers join uname on id=reviewerid
group by id, year
having c>1

#or

select reviewerid, name, year, count(year) as c
from reviewers join uname on id=reviewerid
where rating is not null
group by reviewerid, year
having c>1
```

- Get the count by year of people asked to review more than once in that year (1 row per year). (year, count).

```
select year, count(reviewerid)
from (select reviewerid, name, year, count(year) as c
      from reviewers join uname on id=reviewerid
      group by reviewerid, year
      having c>1) as x
group by year

#or

select year, count(*)
from (select *, count(*) as count
      from reviewers
      group by reviewerid, year
      having count>1) as a
group by year
```