StefanPochmann ★ 46835 Last Edit: October 24, 2018 3:12 AM 66.4K VIEWS

332

These are four different solutions.

With Variables: 841 ms

First one uses two variables, one for the current rank and one for the previous score.

```
SELECT
  Score,
  @rank := @rank + (@prev <> (@prev := Score)) Rank
FROM
  Scores,
  (SELECT @rank := 0, @prev := -1) init
ORDER BY Score desc
```

Always Count: 1322 ms

This one counts, for each score, the number of distinct greater or equal scores.

```
SELECT
  Score,
  (SELECT count(distinct Score) FROM Scores WHERE Score >= s.Score) Rank
FROM Scores s
ORDER BY Score desc
```

Always Count, Pre-uniqued: 795 ms

Same as the previous one, but faster because I have a subquery that "uniquifies" the scores first. Not entirely sure *why* it's faster, I'm guessing MySQL makes `tmp` a temporary table and uses it for every outer Score.

```
SELECT
  Score,
  (SELECT count(*) FROM (SELECT distinct Score s FROM Scores) tmp WHERE s >= Score) Rank
FROM Scores
ORDER BY Score desc
```

Filter/count Scores^2: 1414 ms

Inspired by the attempt in wangkan2001's answer. Finally `Id` is good for something :-)

```
SELECT s.Score, count(distinct t.score) Rank
FROM Scores s JOIN Scores t ON s.Score <= t.score
GROUP BY s.Id
ORDER BY s.Score desc
```

mysql sql

Type comment here... (Markdown is supported)

Post

wyp70627768 ★ 771 Last Edit: October 10, 2018 9:51 AM

Pass me some of your knowledge, I need it for interview...

52 Show 1 reply Reply

sallyyao2001 ★ 67 June 19, 2015 7:46 AM

My solution is strait forward:
select m2.Score,
count(distinct m1.Score)+1 as Rank
from Scores m1, Scores m2
where m1.Score > m2.Score