

SQL HackerRank Advanced Solution

5m left

BETA Can't read the text? Switch theme

2. Winners Chart

ALL

1

2

There were a number of contests where participants each made multiple attempts. The attempt with the highest score is the only one considered. Write a query to list the contestants ranked in the top 3 for each contest. If multiple contestants have the same score in a contest, they are at the same rank.

Report *event_id*, rank 1 name(s), rank 2 name(s), rank 3 name(s). Order the contests by *event_id*. Names that share a rank should be ordered alphabetically and separated by a comma.

Order the report by *event_id*.

▼ Schema

There is one table: `scoretable`.

scoretable		
Name	Type	Description
<code>event_id</code>	<code>int</code>	id of the event.
<code>participant_name</code>	<code>varchar(25)</code>	name of the participant.
<code>score</code>	<code>float</code>	the score

▼ Sample Data Tables

Hackerrank advance level SQL certification question solution by creating dummy table(dataset).

```
use hackerrank_advance;
```

```
--Creating Table Contestants
```

```
create table contestants  
( id int identity(1,1) not null,  
  personName nvarchar(10) );
```

```
--Inserting Values
```

```
insert into contestants ( personName )  
values ( 'Sunil' ), ( 'Shankar' ), ( 'Gaurav' ), ( 'Alankrit' );
```

```
--Creating Table attempts
```

```
create table attempts  
( id int identity(1,1) not null,  
  contestantid int not null,  
  score int not null );
```

```
--Inserting Values
```

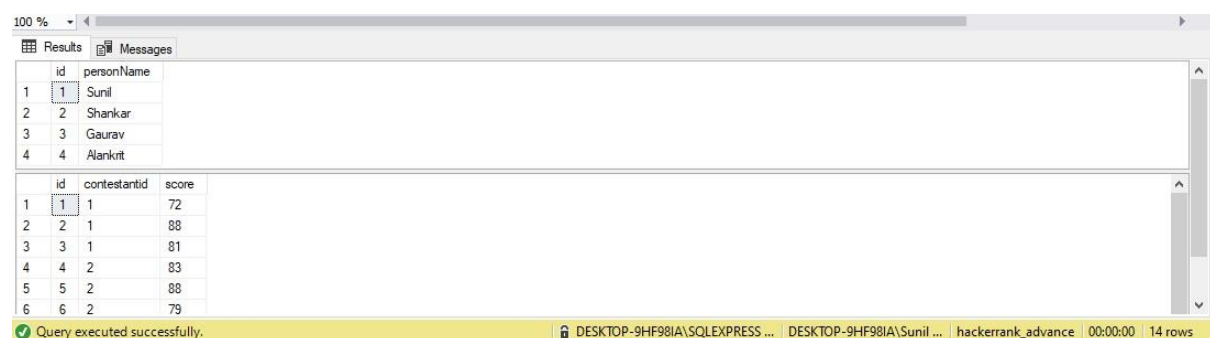
```
insert into attempts ( contestantid, score )  
values  
  ( 1, 72 ), ( 1, 88 ), ( 1, 81 ),  
  ( 2, 83 ), ( 2, 88 ), ( 2, 79 ), ( 2, 86 ),  
  ( 3, 94 ),  
  ( 4, 79 ), ( 4, 87 );
```

```
drop table attempts;
```

```
--Displaying all data from both the tables
```

```
Select * from contestants;
```

```
Select * from attempts;
```



The screenshot shows a SQL Server query window with two result grids. The first grid displays the 'contestants' table with columns 'id' and 'personName'. The second grid displays the 'attempts' table with columns 'id', 'contestantid', and 'score'. The status bar at the bottom indicates the query was executed successfully and returned 14 rows.

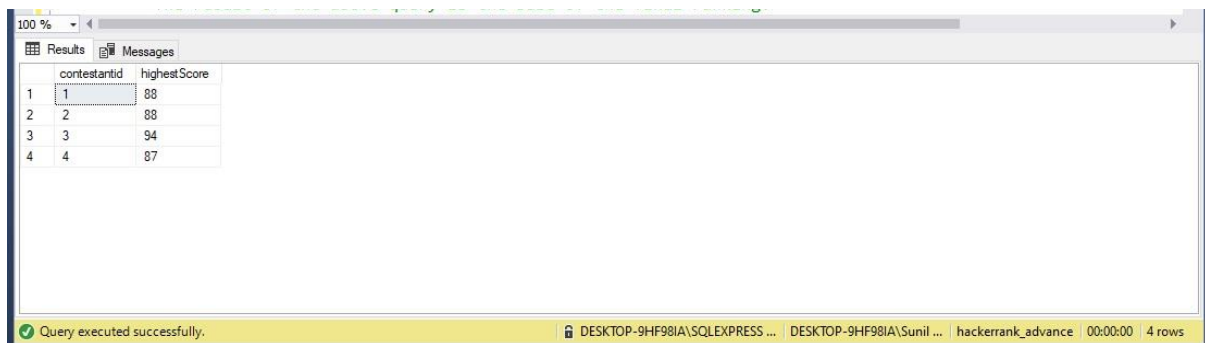
id	personName
1	Sunil
2	Shankar
3	Gaurav
4	Alankrit

id	contestantid	score
1	1	72
2	1	88
3	1	81
4	2	83
5	2	88
6	2	79

Query executed successfully. | DESKTOP-9HF98IA\SQLEXPRESS ... | DESKTOP-9HF98IA\Sunil ... | hackerrank_advance | 00:00:00 | 14 rows

--Each contestants best score which is a simple maximum of the score per contestant.

```
select
    contestantid,
    max( score ) highestScore
from
    attempts
group by
    contestantid;
```



The screenshot shows a SQL query results window with a table containing 4 rows and 2 columns: contestantid and highestScore. The data is as follows:

	contestantid	highestScore
1	1	88
2	2	88
3	3	94
4	4	87

The status bar at the bottom indicates "Query executed successfully." and "4 rows".

--The result of the above query is the base of the final ranking.
--So I have put that query as the FROM source. So instead of a table, the from is the result of the above query
--which I have aliased "Pre" for the pre-aggregation per contestant.

```
select
    ContestantID,
    c.personName,
    DENSE_RANK() OVER ( order by HighestScore DESC ) as
FinalRank
from
    (select
        contestantid,
        max( score ) highestScore
    from
        attempts
    group by
        contestantid ) pre
JOIN Contestants c
    on pre.contestantid = c.id;
```

100 %			
Results Messages			
	ContestantID	personName	FinalRank
1	3	Gaurav	1
2	1	Sunil	2
3	2	Shankar	2
4	4	Alankrit	3

Query executed successfully. DESKTOP-9HF98IA\SQLEXPRESS ... DESKTOP-9HF98IA\Sunil ... hackerrank_advance 00:00:00 4 rows

--Now, if you only wanted a limit such as the top 3 ranks and you actually had 15+ competitors,
 --just wrap this up one more time where a
 where clause

```

select * from
(
select
    ContestantID,
    c.personName,
    DENSE_RANK() OVER ( order by HighestScore DESC ) as
FinalRank
from
    (select
        contestantid,
        max( score ) highestScore
    from
        attempts
    group by
        contestantid ) pre
JOIN Contestants c
    on pre.contestantid = c.id ) dr
where
    dr.FinalRank < 3;

```

100 %			
Results Messages			
	ContestantID	personName	FinalRank
1	3	Gaurav	1
2	1	Sunil	2
3	2	Shankar	2

Query executed successfully. DESKTOP-9HF98IA\SQLEXPRESS ... DESKTOP-9HF98IA\Sunil ... hackerrank_advance 00:00:00 3 rows