Comparing and Ranking Data with MySQL

By Julie Odenbach

We often need to compare and rank records in databases. By comparing and re-ordering our data for various reports and ranking our data, we are able to provide more useful information for our clients and also for our own business needs.

We look at several MySQL functions in this article that are helpful in making those comparisons.

- **ROW_NUMBER()** – returns a unique sequential number for records in a given result set.
- **RANK()** – allows us to find the highest or lowest field value in a result set. Skips positions after records with equal values. Values {2, 4, 4, 6, 7} would result in rank values of 1, 2, 2, 4, 5.
- **DENSE_RANK()** – Similar to rank(), but does not skip positions, so that values {2, 4, 4, 6,7} would result in rank values of 1, 2, 2, 3, 4, with no gap in rank values following rows with equal values.
- **PERCENT_RANK()** – gives the percentile rank of a row within a given result set.

The general format for these commands is:

```
SELECT column, ROW_NUMBER() OVER(
        PARTITION BY column1
        ORDER BY column2 [ASC/DESC]
) FROM TABLE
```

See the SQL commands used to create the database table used for our ranking examples:

```
CREATE TABLE AnimalRescue (
   ID INT PRIMARY KEY,
   Name VARCHAR(50),
   Category VARCHAR(50),
   Breed VARCHAR(50),
   Gender VARCHAR(1),
   Expenditures int,
   AdoptionFee int,
   Adopted boolean
);
INSERT INTO AnimalRescue Values (1001, 'Rosie', 'Dog', 'Golden Retriever', 'F', 350, 425, true);
INSERT INTO AnimalRescue Values (1002, 'Kingston', 'Dog', 'Labrador Retriever', 'M', 200, 250, false);
INSERT INTO AnimalRescue Values (1003, 'Chloe', 'Dog', 'Springer Spaniel', 'F', 225, 200, true);
INSERT INTO AnimalRescue Values (1004, 'Mosby', 'Dog', 'Poodle', 'M', 450, 410, false);
INSERT INTO AnimalRescue Values (1005, 'Maddie', 'Dog', 'Beagle', 'F', 625, 250, true);
INSERT INTO AnimalRescue Values (1006, 'Max', 'Dog', 'Terrier', 'M', 225, 350, false);
INSERT INTO AnimalRescue Values (1007, 'Boots', 'Cat', 'Domestic Short Hair', 'F', 150, 200, true);
INSERT INTO AnimalRescue Values (1008, 'Kiara', 'Cat', 'Siamese', 'M', 200, 225, true);
INSERT INTO AnimalRescue Values (1009, 'Hazel', 'Cat', 'Rag Doll', 'F', 150, 250, true);
INSERT INTO AnimalRescue Values (1010, 'Goldie', 'Cat', 'Domestic Shorthair', 'M', 280, 160, false);
INSERT INTO AnimalRescue Values (1011, 'Midnight', 'Cat', 'Domestic Shorthair', 'F', 320, 220, true);
INSERT INTO AnimalRescue Values (1012, 'Maverick', 'Cat', 'Persian', 'M', 410, 250, false);
```

ROW_NUMBER Examples

The **ROW_NUMBER()** function generates a unique sequential rank for each row retrieved for a specified partition. This SQL query below will generate a unique row number for each record retrieved for each of the two partitions resulting for each Category. Our results list all of the cats and dogs in the AnimalRescue table, listed in the order of their AdoptionStatus for each Category.

```
SELECT
        ROW_NUMBER() over(
                PARTITION BY Category
                ORDER BY Adopted
        ) as RowNumber,
 Category,
 Breed,
 Name,
 CASE
        WHEN Adopted = 0 THEN 'InFosterCare'
        WHEN Adopted = 1 THEN 'Adopted'
END as AdoptionStatus
FROM AnimalRescue;
```

RowNumber	Category	Breed	Name	AdoptionStatus
1	Cat	Domestic Shorthair	Goldie	InFosterCare
2	Cat	Persian	Maverick	InFosterCare
3	Cat	Domestic Short Hair	Boots	Adopted
4	Cat	Siamese	Kiara	Adopted
5	Cat	Rag Doll	Hazel	Adopted
6	Cat	Domestic Shorthair	Midnight	Adopted
1	Dog	Labrador Retriever	Kingston	InFosterCare
2	Dog	Poodle	Mosby	InFosterCare
3	Dog	Terrier	Max	InFosterCare
4	Dog	Golden Retriever	Rosie	Adopted
5	Dog	Springer Spaniel	Chloe	Adopted
6	Dog	Beagle	Maddie	Adopted

RANK Examples

Let's generate a list of pets that ranks expenditures for each pet using the **RANK()** function. We are partitioning by Category, so all the cats are listed first, and then all the dogs are listed. The pets in each Category are ranked in descending order for Expenditures. Note that for Id=1007 and 1009, that the ExpendituresRank is 5 for both Cat rows, since the $510 Expenditures are equal. Also note that for Id = 1003 and 1006, the ExpendituresRank is 4 for both dogs. The ExpendituresRank = 6 for dog Id = 1002. The rank generated following multiple equal values is handled differently if you are using **RANK()** versus **DENSE_RANK()**.

```
# list expenditures for all pets in desc order
select Id, Name, Category, Expenditures,
            Rank() over (partition by Category order by Expenditures DESC) as ExpendituresRank
FROM AnimalRescue;
```

Id	Name	Category	Expenditures	ExpendituresRank
1012	Maverick	Cat	410	1
1011	Midnight	Cat	320	2
1010	Goldie	Cat	280	3
1008	Kiara	Cat	200	4
1007	Boots	Cat	150	5
1009	Hazel	Cat	150	5
1005	Maddie	Dog	625	1
1004	Mosby	Dog	450	2
1001	Rosie	Dog	350	3
1003	Chloe	Dog	225	4
1006	Max	Dog	225	4
1002	Kingston	Dog	200	6

What is the highest expenditure for a dog?

```
# What is the highest expenditure for a dog?
select Id, Name, Category, Expenditures,
        Rank() over (partition by Category order by Expenditures DESC) as DogExpendituresRank
FROM AnimalRescue
WHERE Category = 'Dog' LIMIT 1;
```

Id	Name	Category	Expenditures	DogExpendituresRank
1005	**Maddie**	**Dog**	**625**	**1**

DENSE_RANK Examples

```
# Showing the dense_rank() numbering for ordering catExpenditures:
select Id, Name, Category, Expenditure,
        Dense_Rank() over (partition by Category order by AdoptionFee DESC) as CatExpendituresRank
FROM AnimalRescue
WHERE Category = 'Cat';
```

Id	Name	Category	Expenditures	catExpendituresRank
1012	Maverick	Cat	410	1
1011	Midnight	Cat	320	2
1010	Goldie	Cat	280	3
1008	Kiara	Cat	200	4
1007	Boots	Cat	150	5
1009	Hazel	Cat	150	5

The **DENSE_RANK()** function is used below to find the 2nd highest expenditure for cats. In this query, we specify 'WHERE Rnk=2' to find the 2nd highest expenditure in our result set for cats.

```
## What is the second highest expenditure for a cat?
WITH T AS
(
SELECT ID, NAME, Category, Breed, Expenditures,
  DENSE_RANK() OVER (ORDER BY Expenditures Desc) AS Rnk
FROM AnimalRescue
WHERE Category = 'Cat'
GROUP BY ID
)
SELECT ID, Name, Category, Breed, Expenditures as 2ndHighestCatExpenditure
FROM T
WHERE Rnk=2;
```

ID	Name	Category	Breed	2ndHighestCatExpenditure
1011	Midnight	Cat	Domestic Shorthair	320

PERCENT_RANK Examples

Sometimes it’s useful to know the percent ranking of our data to tell us where our field values fall along the spectrum of values for a given field. The PERCENT_RANK() function is used for this purpose. In the example below we rank the expenditures for each pet in our AnimalRescue table.

```
WITH t AS (
   SELECT ID, Name, Category, Breed, Expenditures
   FROM
      AnimalRescue
   GROUP BY ID
)
SELECT
   Name, Category, Breed, Expenditures,
   Concat(
               FORMAT(percent_rank() over (
                              order by Expenditures ASC
               )
               * 100,0), '%'
) as PercentileRank
FROM
   t
order by Expenditures ASC;
```

Name	Category	Breed	Expenditures	PercentileRank
Boots	Cat	Domestic Short Hair	150	0%
Hazel	Cat	Rag Doll	150	0%
Kingston	Dog	Labrador Retriever	200	18%
Kiara	Cat	Siamese	200	18%
Chloe	Dog	Springer Spaniel	225	36%
Max	Dog	Terrier	225	36%
Goldie	Cat	Domestic Shorthair	280	55%
Midnight	Cat	Domestic Shorthair	320	64%
Rosie	Dog	Golden Retriever	350	73%
Maverick	Cat	Persian	410	82%
Mosby	Dog	Poodle	450	91%
Maddie	Dog	Beagle	625	100%

Our last example shows how to display our expenditures as a percentage spent on the dogs versus the cats.

```
With P as (
        SELECT Category, Expenditures,
                (SELECT SUM(Expenditures) FROM AnimalRescue WHERE Category = 'Dog') as dogExpenditures,
                (SELECT SUM(Expenditures) FROM AnimalRescue WHERE Category = 'Cat') as catExpenditures,
                (SELECT  SUM(Expenditures)  FROM  AnimalRescue  WHERE  Category  =  'Cat'  OR  Category  =  'Dog')  as
        TotalExpenditures
        FROM AnimalRescue
)
SELECT
        Category,
        CASE
                WHEN Category ='Dog' THEN dogExpenditures
                WHEN Category = 'Cat' THEN catExpenditures
        END as Expenditures,
  Concat(
        FORMAT(
                (CASE
                        WHEN Category ='Dog' THEN dogExpenditures/TotalExpenditures
                        WHEN Category = 'Cat' THEN catExpenditures/totalExpenditures
                END) * 100,0), '%'
        ) as CatVsDogExpenditures
FROM  P
Group by (Category);
```

Category	Expenditures	CatVsDogExpenditures
Dog	2075	58%
Cat	1510	42%

Conclusion for Comparing and Ranking Data

We have provided examples for demonstrating ROW_NUMBER(), RANK(), DENSE_RANK(), and PERCENT_RANK() functions which offer useful ways to order database records and determine the highest or lowest rankings of data. Of course, our examples only used a small sample database, but these functions provide an excellent way to make sense of your data when dealing with much larger real-life databases.

I hope that you found these examples a good starting point for using ranking functions, and there will be much more to learn as you apply these data ranking functions to your data.