LATERAL Derived Tables in MySQL 8.0

Norvald H. Ryeng
Software Development Senior Manager
MySQL Optimizer Team
May 29, 2019
LATERAL what?
Derived Tables

• Derived tables are subqueries in FROM clauses
  
  SELECT ... FROM \textit{t1}, (\textit{subquery}) \textit{AS derived}, \textit{t2} ...

• Two execution methods
  
  – Materialized
  – Merged into the outer query
lateral  adjective
lat-er-al  |  \'la-ter-əl also \'la-trel  \\  

Definition of lateral (Entry 1 of 3)

1  : of or relating to the side
   // a lateral view

2  : situated on, directed toward, or coming from the side
   // the lateral branches of a tree

3  : extending from side to side
   // the lateral axis of an airplane

4  phonetics : produced with passage of breath around the side
   of a constriction formed with the tongue
   // \ is lateral
LATERAL Derived Tables

- Can refer to other tables in the same FROM clause
  
  ```sql
  SELECT ... FROM t1, LATERAL (SELECT ... FROM ... WHERE ... = t1.col) AS derived, t2 ...
  ```
  
  - Only to tables that **appear before it** in the FROM clause
  - Including other derived tables

- Two execution methods
  
  - Materialized
  - Merged into the outer query

- SQL feature T491
LATERAL Derived Tables

- Can refer to other tables in the same FROM clause

```
SELECT ... FROM t1, LATERAL (SELECT ... FROM ... WHERE ... = t1.col) AS derived, t2 ...
```

- Only to tables that appear before it in the FROM clause
- Including other derived tables

- Two execution methods
  - Materialized
  - Merged into the outer query

- SQL feature T491
Implicitly LATERAL Table Functions

- Table functions are implicitly LATERAL
  - Not allowed to explicitly specify LATERAL
- MySQL has one table function: JSON_TABLE

```sql
SELECT people.*
FROM t1,
     JSON_TABLE(t1.json_col, '$.people[*]' COLUMNS (
              name VARCHAR(40) PATH '$.name',
              address VARCHAR(100) PATH '$.address'))
AS people;
```
LATERAL Without Lateral References

• Declared as LATERAL, but contains no lateral references

  SELECT ... FROM t1, LATERAL (SELECT 1) AS derived, t2 ...
  – Optimized by MySQL as if LATERAL was not present
  – No performance penalty

• Don't add LATERAL to all your derived tables!
  – Won't get any warning when accidentally using a lateral reference
  – Confusing the next person to read your query
Examples, please!
Example Setup

CREATE TABLE cities (  
    city_name VARCHAR(40),  
    population BIGINT,  
    country_name VARCHAR(40)  
);  

INSERT INTO cities VALUES  
    ('Shanghai', 24183300, 'China'),  
    ('Beijing', 20794000, 'China'),  
    ...  
);
The Largest City of Each Country, Option 1

SELECT `dt.population`, `dt.city_name`, `c.country_name` 
FROM
  (SELECT DISTINCT `country_name` FROM cities) AS `c`,
  LATERAL ( 
    SELECT `city_name`, `population` 
    FROM cities 
    WHERE cities.country_name = `c.country_name` 
    ORDER BY `population` DESC 
    LIMIT 1 
  ) AS `dt`;
The Largest City of Each Country, Option 2

SELECT dt.pop, dt2.city_name, dt.country_name
FROM
  (SELECT country_name, MAX(population) AS pop
   FROM cities
   GROUP BY country_name
  ) AS dt,
  LATERAL (SELECT city_name
            FROM cities
            WHERE cities.country_name = dt.country_name
             AND cities.population = dt.pop
      ) AS dt2;
The Largest City of Each Country, No LATERAL!

```
SELECT dt.pop, cities.city_name, dt.country_name
FROM
  (SELECT country_name, MAX(population) AS pop
   FROM cities
   GROUP BY country_name
  ) AS dt
JOIN cities
  ON cities.country_name = dt.country_name
     AND cities.population = dt.pop;
```
The Largest City of Each Country, No LATERAL!

```sql
SELECT dt.pop, cities.city_name, dt.country_name
FROM
  (SELECT country_name, MAX(population) AS pop
   FROM cities
   GROUP BY country_name
  ) AS dt
JOIN cities
  ON cities.country_name = dt.country_name
  AND cities.population = dt.pop;
```

This is how MySQL rewrites the previous query after merging the lateral derived table into the outer query.
Feature descriptions and design details directly from the source.

http://mysqlserverteam.com/
Integrated Cloud
Applications & Platform Services
Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.