```
SELECT /*+ MAX_EXECUTION_TIME(1000) */ * FROM cte;
```

Beginning with MySQL 8.0.19, you can also use `LIMIT` within the recursive query to impose a maximum nuber of rows to be returned to the outermost `SELECT`, for example:

```
WITH RECURSIVE cte (n) AS
(
  SELECT 1
  UNION ALL
  SELECT n + 1 FROM cte LIMIT 10000
)
SELECT * FROM cte;
```

You can do this in addition to or instead of setting a time limit. Thus, the following CTE terminates after returning ten thousand rows or running for one thousand seconds, whichever occurs first:

```
WITH RECURSIVE cte (n) AS
(
  SELECT 1
  UNION ALL
  SELECT n + 1 FROM cte LIMIT 10000
)
SELECT /*+ MAX_EXECUTION_TIME(1000) */ * FROM cte;
```

If a recursive query without an execution time limit enters an infinite loop, you can terminate it from another session using `KILL QUERY`. Within the session itself, the client program used to run the query might provide a way to kill the query. For example, in `mysql`, typing **Control+C** interrupts the current statement.

Recursive Common Table Expression Examples

As mentioned previously, recursive common table expressions (CTEs) are frequently used for series generation and traversing hierarchical or tree-structured data. This section shows some simple examples of these techniques.

- Fibonacci Series Generation
- Date Series Generation
- Hierarchical Data Traversal

Fibonacci Series Generation

A Fibonacci series begins with the two numbers 0 and 1 (or 1 and 1) and each number after that is the sum of the previous two numbers. A recursive common table expression can generate a Fibonacci series if each row produced by the recursive `SELECT` has access to the two previous numbers from the series. The following CTE generates a 10-number series using 0 and 1 as the first two numbers:

```
WITH RECURSIVE fibonacci (n, fib_n, next_fib_n) AS
(
  SELECT 1, 0, 1
  UNION ALL
  SELECT n + 1, next_fib_n, fib_n + next_fib_n
    FROM fibonacci WHERE n < 10
)
SELECT * FROM fibonacci;
```

The CTE produces this result:

```
+------+-------+------------+
| n    | fib_n | next_fib_n |
```