Application-time periods in MariaDB

Nikita Malyavin
MariaDB Corporation
CREATE TABLE transactions (
  trans_id INT PRIMARY KEY,
  node INT,
  start TIMESTAMP(6),
  end TIMESTAMP(6));

<table>
<thead>
<tr>
<th>trans_id</th>
<th>node</th>
<th>start</th>
<th>end</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2020-02-02 01:02:03.00010</td>
<td>2020-02-02 01:02:03.23012</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2020-02-02 01:02:03.23202</td>
<td>2020-02-02 01:02:04.00118</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2020-02-02 01:02:03.00123</td>
<td>2020-02-02 01:02:04.567890</td>
</tr>
</tbody>
</table>
TABLES THAT HOLD A PERIOD

<table>
<thead>
<tr>
<th>trans_id</th>
<th>node</th>
<th>start</th>
<th>end</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2020-02-02 01:02:03.00010</td>
<td>2020-02-02 01:02:03.23012</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2020-02-02 01:02:03.23202</td>
<td>2020-02-02 01:02:04.00118</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2020-02-02 01:02:03.00123</td>
<td>2020-02-02 01:02:04.567890</td>
</tr>
</tbody>
</table>

CREATE TABLE transactions (  
  trans_id INT PRIMARY KEY,  
  node INT,  
  start TIMESTAMP(6),  
  end TIMESTAMP(6),  
  CONSTRAINT(end > start)  
  PERIOD FOR trans_time(start, end));
PERIODS: DATA MODEL

CREATE TABLE bookings(
    room INT,
    date_start DATE,
    date_end DATE,
    PERIOD FOR booking(date_start, date_end));
CREATE TABLE bookings(
  room INT,
  date_start DATE,
  date_end DATE,
  PERIOD FOR booking(date_start, date_end),
  UNIQUE(room, booking WITHOUT OVERLAPS));

<table>
<thead>
<tr>
<th>room</th>
<th>date_start</th>
<th>date_end</th>
</tr>
</thead>
<tbody>
<tr>
<td>1408</td>
<td>2007-06-12</td>
<td>2007-06-13</td>
</tr>
<tr>
<td>1408</td>
<td>2020-01-30</td>
<td>2020-02-02</td>
</tr>
<tr>
<td>1337</td>
<td>2020-01-30</td>
<td>2020-02-02</td>
</tr>
<tr>
<td>1337</td>
<td>2021-01-30</td>
<td>2021-02-02</td>
</tr>
</tbody>
</table>
### PERIODS: DATA MODEL

<table>
<thead>
<tr>
<th>trans_id</th>
<th>node</th>
<th>start</th>
<th>end</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2020-02-02 01:02:03.00010</td>
<td>2020-02-02 01:02:03.23012</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2020-02-02 01:02:03.23202</td>
<td>2020-02-02 01:02:04.00118</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2020-02-02 01:02:03.00123</td>
<td>2020-02-02 01:02:04.567890</td>
</tr>
</tbody>
</table>

CREATE TABLE transactions (  
  trans_id INT PRIMARY KEY, node INT,  
  start TIMESTAMP(6), end TIMESTAMP(6),  
  PERIOD FOR trans_time(start, end),  
  UNIQUE(node, trans_time WITHOUT OVERLAPS));
CREATE TABLE bookings(
    room INT,
    date_start DATE,
    date_end DATE,
    PERIOD FOR booking(date_start, date_end),
    UNIQUE(room, booking WITHOUT OVERLAPS));
PERIODS: DATA MODEL

CREATE TABLE bookings(
    room INT,
    date_start DATE,
    date_end DATE,
    PERIOD FOR booking(date_start, date_end),
    UNIQUE(room,
        booking WITHOUT OVERLAPS));
WITHOUT OVERLAPS: AN ALGORITHM

INSERT INTO bookings VALUES (1408, '2020-02-01', '2020-02-03');

Find the smallest date_end, such that

\[ \text{date}_\text{end} > '2020-02-01' \]
WITHOUT OVERLAPS: AN ALGORITHM

```
INSERT INTO bookings VALUES (1408, '2020-02-01', '2020-02-03');
```

Find the smallest date_end, such that

```
date_end > '2020-02-01'
```

KEY(room, date_end) is enough
REFERENTIAL INTEGRITY

CREATE TABLE bookings(
    room INT,
    date_start TIMESTAMP(6),
    date_end TIMESTAMP(6),
    PERIOD FOR booking(date_start,
        date_end),
    FOREIGN KEY(room, PERIOD booking)
        REFERENCES prices(room,
            PERIOD term)
    UNIQUE(...));

CREATE TABLE prices(
    room INT,
    date_start TIMESTAMP(6),
    date_end TIMESTAMP(6),
    PERIOD FOR term(date_start,
        date_end),
    price INT,
    UNIQUE(room,
        term WITHOUT OVERLAPS));
prices

Jan 01  Jan 15  Feb 04  Feb 05  Feb 29

bookings
ALGORITHMS: DELETE

Find the smallest ref_end, such that
ref_end > del_start

KEY(..., del_end, del_start)

KEY(..., ref_end, ref_start)
ALGORITHMS: DELETE

Find the smallest ref_end, such that
ref_end > del_start

NO WITHOUT OVERLAPS!

R-TREE? 😞
ALGORITHMS: INSERT

Ensure that \((\text{ins}_\text{start}, \text{ins}_\text{end})\)
is continuous (i.e. no holes) in parent table.
ALGORITHMS: UPDATE

delete + insert
ALGORITHMS: UPDATE
ALGORITHMS: UPDATE

delete

insert
ARCHITECTURE

sql

insert_row() -> handler

update_row() -> handler

delete_row() -> handler

InnoDB

ColumnStore

MyISAM

RI constraint is implemented here
MOVING FOREIGN KEY TO SQL

- MDEV-20480 Obsolete internal parser for FK in InnoDB

- MDEV-16417 Store Foreign Key metadata outside of InnoDB
  - MDEV-20865 Store foreign key info in TABLE_SHARE
  - MDEV-21051 Store and read foreign key info into/from FRM files

work by Alexey Midenkov (midenok)
FUTURE

- Period operations (OVERLAPS, SUCCEEDS, PRECEDES, etc.)
- Views
- More than one period?
- Optimizations
- Cross-engine references