

# Application-time periods in MariaDB

---

Nikita Malyavin  
MariaDB Corporation



# TABLES THAT HOLD A PERIOD

trans_id	node	start	end
1	2	2020-02-02 01:02:03.00010	2020-02-02 01:02:03.23012
2	2	2020-02-02 01:02:03.23202	2020-02-02 01:02:04.00118
3	3	2020-02-02 01:02:03.00123	2020-02-02 01:02:04.567890

```
CREATE TABLE transactions (  
  trans_id INT PRIMARY KEY,  
  node INT,  
  start TIMESTAMP(6),  
  end TIMESTAMP(6));
```

# TABLES THAT HOLD A PERIOD

trans_id	node	start	end
1	2	2020-02-02 01:02:03.00010	2020-02-02 01:02:03.23012
2	2	2020-02-02 01:02:03.23202	2020-02-02 01:02:04.00118
3	3	2020-02-02 01:02:03.00123	2020-02-02 01:02:04.567890

```
CREATE TABLE transactions (  
  trans_id INT PRIMARY KEY,  
  node INT,  
  start TIMESTAMP(6),  
  end TIMESTAMP(6),  
  PERIOD FOR trans_time(start, end));
```

*period*

```
CONSTRAINT(end > start)
```

# PERIODS: DATA MODEL

```
CREATE TABLE bookings(  
  room INT,  
  date_start DATE,  
  date_end DATE,  
  PERIOD FOR booking(date_start,  
                      date_end));
```

room	date_start	date_end
1408	2007-06-12	2007-06-13
1408	2020-01-30	2020-02-02
1337	2020-01-30	2020-02-02
1337	2021-01-30	2021-02-02

# 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));
```

room	date_start	date_end
1408	2007-06-12	2007-06-13
1408	2020-01-30	2020-02-02
1337	2020-01-30	2020-02-02
1337	2021-01-30	2021-02-02

# PERIODS: DATA MODEL

trans_id	node	start	end
1	2	2020-02-02 01:02:03.00010	2020-02-02 01:02:03.23012
2	2	2020-02-02 01:02:03.23202	2020-02-02 01:02:04.00118
3	3	2020-02-02 01:02:03.00123	2020-02-02 01:02:04.567890

```
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));
```



# 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));
```

room	date_start	date_end
1408	2007-06-12	2007-06-13
1408	2020-01-30	2020-02-02
1337	2020-01-30	2020-02-02
1337	2021-01-30	2021-02-02

# PERIODS: DATA MODEL

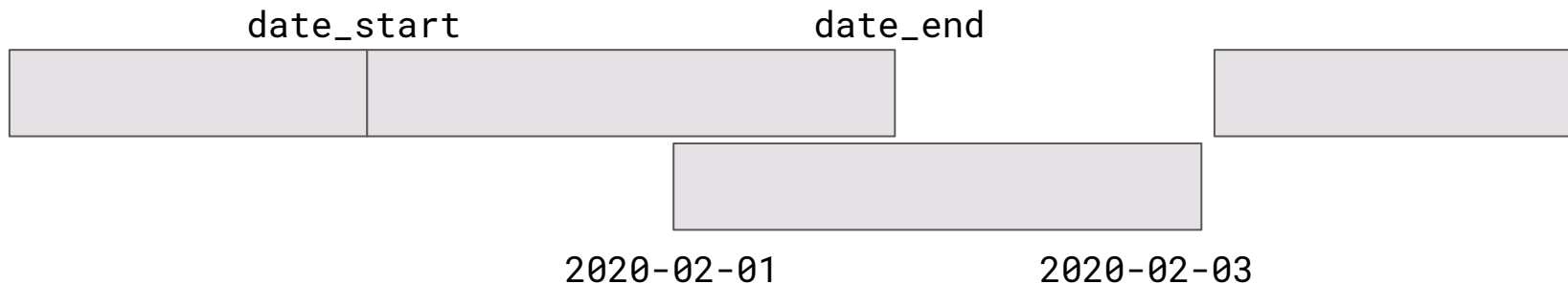
```
CREATE TABLE bookings(  
  room INT,  
  date_start DATE,  
  date_end DATE,  
  PERIOD FOR booking(date_start,  
                      date_end),  
  UNIQUE(room,  
         booking WITHOUT OVERLAPS));
```

room	date_start	date_end
1408	2007-06-12	2007-06-13
1408	2020-01-30	2020-02-02
1337	2020-01-30	2020-02-02
1337	2021-01-30	2021-02-02
1337	2021-01-29	2021-02-05



# WITHOUT OVERLAPS: AN ALGORITHM

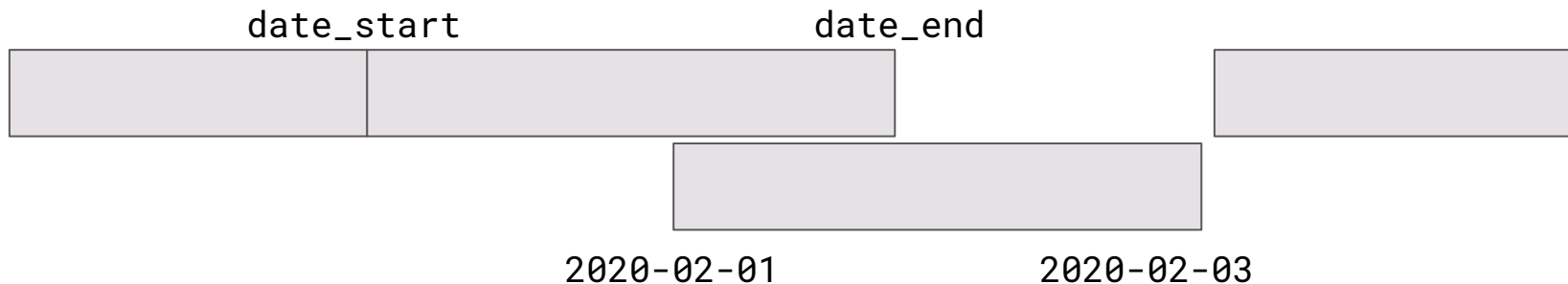
```
INSERT INTO bookings VALUES (1408, '2020-02-01', '2020-02-03');
```



Find the smallest `date_end`, such that  
`date_end > '2020-02-01'`

# 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

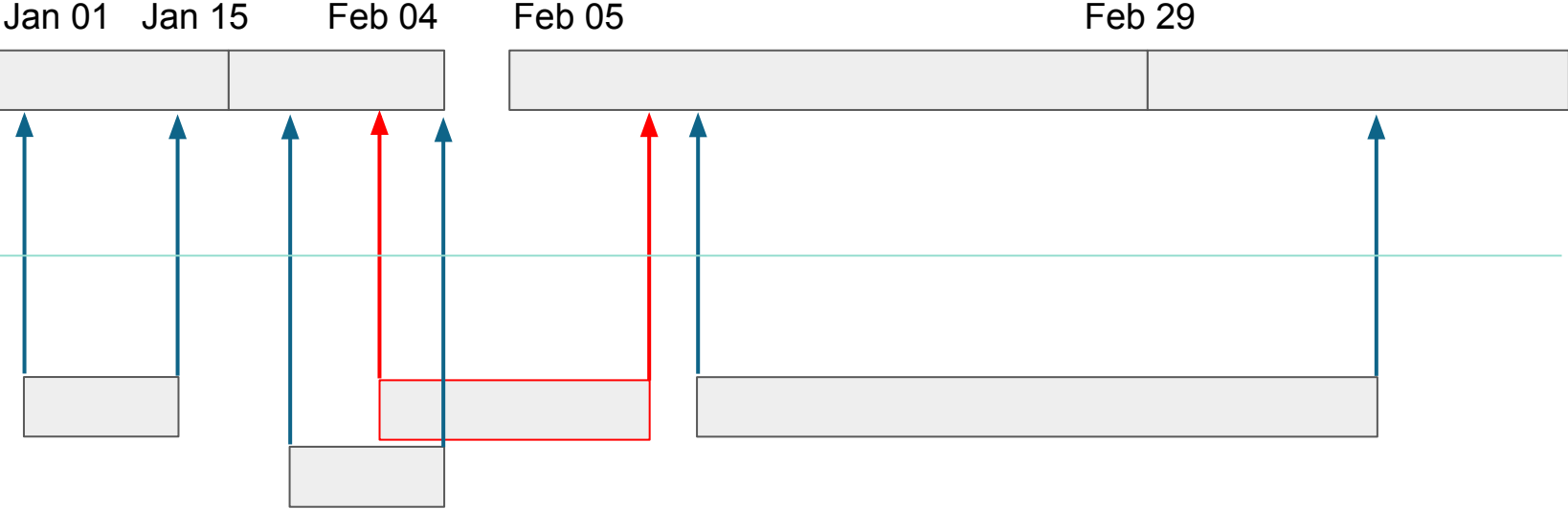
# REFERENTIAL INTEGRITY

```
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));
```

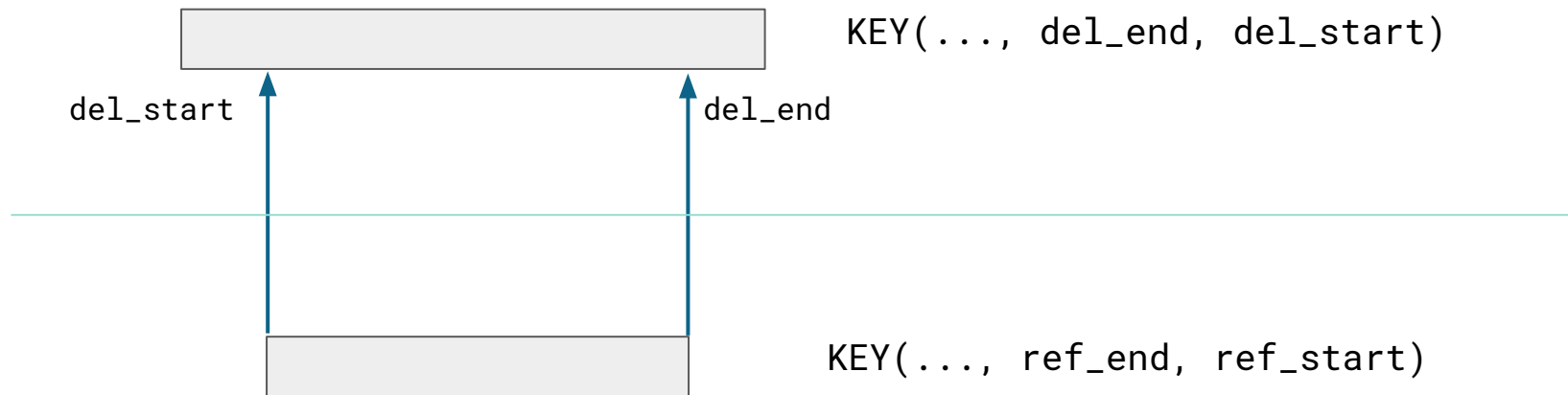


prices



bookings

# ALGORITHMS: DELETE



Find the smallest `ref_end`, such that  
`ref_end > del_start`

# ALGORITHMS: DELETE

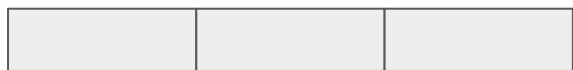


Find the smallest `ref_end`, such that  
`ref_end > del_start`

**NO WITHOUT  
OVERLAPS!**

**R-TREE?** 🤔

# ALGORITHMS: INSERT



Ensure that `[ins_start, ins_end)`  
is continuous (i.e. no holes) in parent table



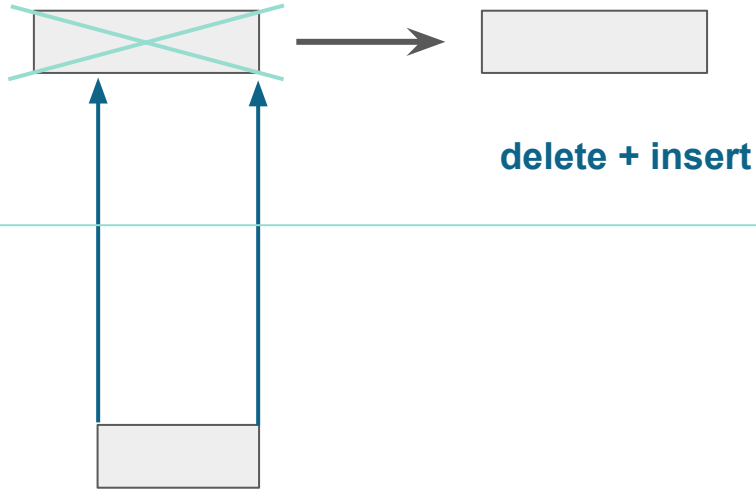
`ins_start`

`ins_end`

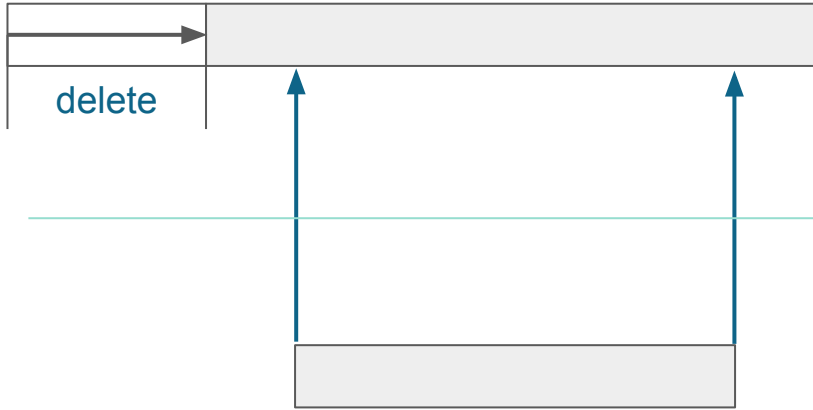




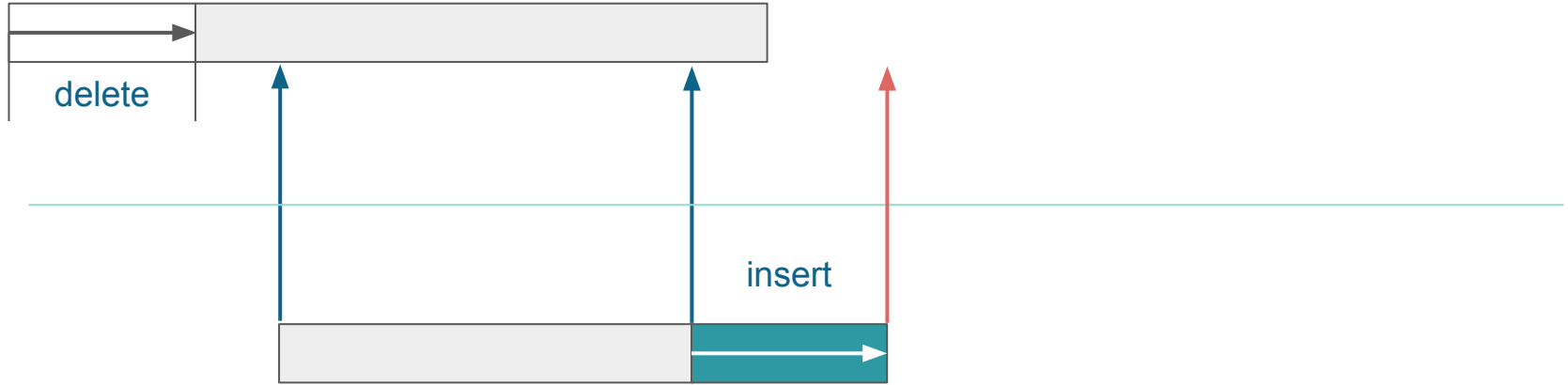
# ALGORITHMS: UPDATE



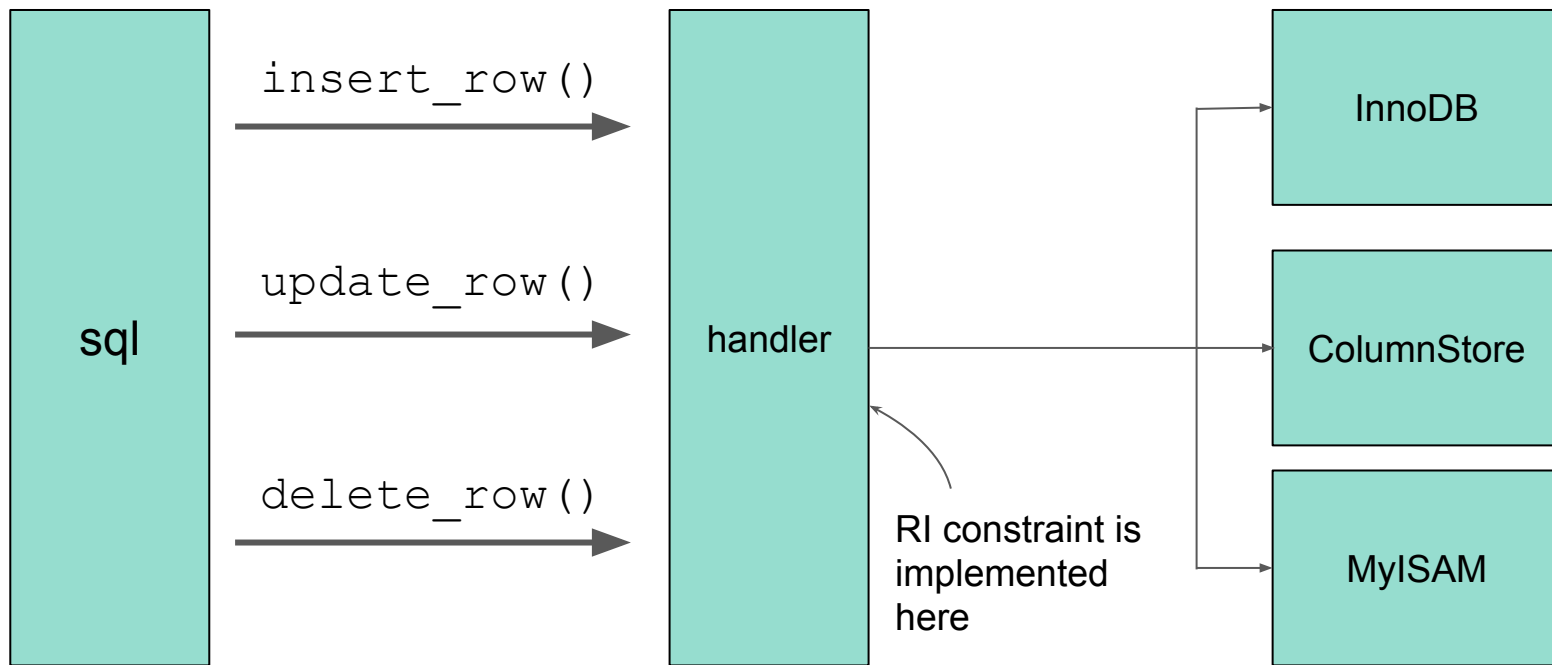
# ALGORITHMS: UPDATE



# ALGORITHMS: UPDATE



# ARCHITECTURE



# 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