Beyond UNIQUE Exclusion Constraints in PostgreSQL 9.0

FOSDEM 2010
Brussels, Belgium

https://www.postgresql.eu/events/feedback/

Magnus Hagander
Redpill Linpro AB
First things first

Exclusion Constraints

!=

constraint_exclusion
What are constraints

• Declarative
• Part of the data model
• Always checked
• CHECK, NOT NULL, UNIQUE, FOREIGN KEY
UNIQUE constraints

- Two rows can conflict with each other
  - No other constraints has this property
- Implemented only on btree indexes
- Simple predicate lock
When unique is not enough

- Unique geographical regions
  - PostGIS
  - Means «non-overlapping»

- Non-overlapping time ranges
  - Booking of a room
  - Scheduling an event
Trivial example

- «Booking a conference room»
- Multiple rooms
- Multiple people booking it
- Dealing with overlaps
Enforce non-overlapping today

• Suggestions?
Enforce non-overlapping today

- Serialize – table level lock
  - And manually search before each insert
  - Will never scale

- Check using trigger
  - Concurrency issues
  - Performance issues
  - Not reusable
  - Very easy to get wrong
Enforce non-overlapping today

• Delayed check
  – Accept all bookings
  – Reject later, «hope it doesn't happen often»

• Solve outside the database
  – No real need for a constraint
Enforce non-overlapping today

• Conflicts will appear eventually
  – Application level checks not 100%

• Conflicts will get *resolved* eventually
  – Unfortunately, too late
  – Who hasn't had a double-booked room?
  – The later you reject, the more costly
How about an actual solution?

- Exclusion Constraints
- New in PostgreSQL 9.0
- General constraint mechanism
  - Many different operators
  - Based on GiST
Short side-track

• The PERIOD datatype
  - Not in 9.0 – pgFoundry
  - Makes dealing with time intervals much nicer
  - *Not* a requirement, but easier

• Single datatype for start and end time
CREATE TABLE bookings(title text, room text, during period)

INSERT INTO bookings values ('Constraint talk', 'AW1.121',
  period('2010-02-06 16:15', '2010-02-06 17:00'))

INSERT INTO bookings values ('Zoo talk', 'AW1.121',
  period('2010-02-06 17:15', '2010-02-06 18:00'))

INSERT INTO bookings values ('Features talk', 'AW1.121',
  period('2010-02-06 17:30', '2010-02-06 18:15'))
Short side-track

```sql
SELECT b1.title, b2.title
FROM bookings b1, bookings b2
WHERE
  overlaps(b1.during, b2.during)
  AND b1.title<b2.title;
```

<table>
<thead>
<tr>
<th>title</th>
<th>title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Features talk</td>
<td>Zoo talk</td>
</tr>
</tbody>
</table>

(1 row)
Back to constraints

● We inserted a conflict
● But the system knew it was there
  – Create a trigger!
  – Using the overlaps function

• overlaps() function is also && operator
  – P1 && P2 same as overlaps(P1, P2)
Exclusion constraints

• Let's redefine our table

```
CREATE TABLE bookings(
    title text,
    room text,
    during period,
    EXCLUDE USING gist
    (room WITH =,
     during WITH &&)
)
```

NOTICE: CREATE TABLE / EXCLUDE will create implicit index "bookings_room_during_exclusion" for table "bookings"
Constraint violations

INSERT INTO bookings values ('Features talk', 'AW1.121', period('2010-02-06 17:30', '2010-02-06 18:15'));

ERROR: conflicting key value violates exclusion constraint "bookings_room_during_exclusion"
DETAIL: Key (room, during)=(AW1.121, [2010-02-06 17:30:00+01, 2010-02-06 18:15:00+01]) conflicts with existing key (room, during)=(AW1.121, [2010-02-06 17:15:00+01, 2010-02-06 18:00:00+01]).
Syntax details

CREATE TABLE bookings(
    title text,
    room text,
    during period,
    EXCLUDE USING gist
        (room WITH =,
         during WITH &&)
)

• Currently, only GiST is supported
Syntax details

CREATE TABLE bookings(
    title text,
    room text,
    during period,
    EXCLUDE USING gist
    (room WITH =,
    during WITH &&)
)

• Columns or expressions supported
Syntax details

CREATE TABLE bookings(
    title text,
    room text,
    during period,
    EXCLUDE USING gist
    (room WITH =,  
     during WITH &&)
)

• Exclusion operator. Must support GiST.
Operator

- Operator is used to find conflicts
- Must return **TRUE** when two values conflict
- Must return **TRUE** when two values conflict
- Thus, «overlaps» makes sure there are no tuples that overlap
Multi-column constraints

• Multi-column constraints are always ANDed

• As long as one of the columns is not in conflict, tuple is allowed

• To do OR, create multiple constraints
Multiple constraints

CREATE TABLE bookings(
    title text,
    room text,
    during period,
    EXCLUDE USING gist
        (room WITH =),
    EXCLUDE USING gist
        (during WITH &&)
)
Multiple constraints

CREATE TABLE bookings(
    title text,
    room text,
    teacher text,
    during period,
    EXCLUDE USING gist
    (room WITH =, during WITH &&),
    EXCLUDE USING gist
    (teacher WITH =, during WITH &&)
)
Redefining UNIQUE

CREATE TABLE bookings(
    title text, room text,
    during period,
    EXCLUDE USING gist
    (room WITH =),
)

CREATE TABLE bookings(
    title text, room text UNIQUE,
    during period
)

- Worse performance, but more datatypes
Partial constraints

- Work just like partial indexes

```sql
CREATE TABLE bookings(
    title text,
    room text,
    during period,
    EXCLUDE USING gist
    (room WITH =, during WITH &&)
    WHERE (during >>
        period('2010-01-01'::timestamptz))
)
```
Thank You!

Please leave feedback:
https://www.postgresql.eu/events/feedback/

Questions?