Indexes of PostgreSQL

Emre Hasegeli

InnoGames

CREATE INDEX

```
CREATE [ UNIQUE ] INDEX [ CONCURRENTLY ] [ name ]
    ON table_name [ USING method ]
    ( { column_name | ( expression ) }
    [ COLLATE collation ] [ opclass ]
    [ ASC | DESC ] [ NULLS { FIRST | LAST } ] [, ...] )
    [ WITH ( storage_parameter = value [, ... ] ) ]
    [ TABLESPACE tablespace_name ]
    [ WHERE predicate ]
```

DROP INDEX

```
DROP INDEX [ CONCURRENTLY ] [ IF EXISTS ] name [, ...]
[ CASCADE | RESTRICT ]
```

REINDEX

```
REINDEX [ ( { VERBOSE } [, ...] ) ]
{ INDEX | TABLE | SCHEMA | DATABASE | SYSTEM } name
```

Expression Index Example

```
CREATE UNIQUE INDEX ON product
    ((listed_at::date));

CREATE UNIQUE INDEX ON product
    (lower(name));

CREATE EXTENSION unaccent;

CREATE INDEX ON product
    (lower(unaccent(name)));
```

WHERE Clause Example

```
CREATE UNIQUE INDEX ON product (stock_id)
    WHERE status = 'available';
```

```
SELECT * FROM product
WHERE status = 'available'
  AND stock_id = 3;
```

Index Access Nodes

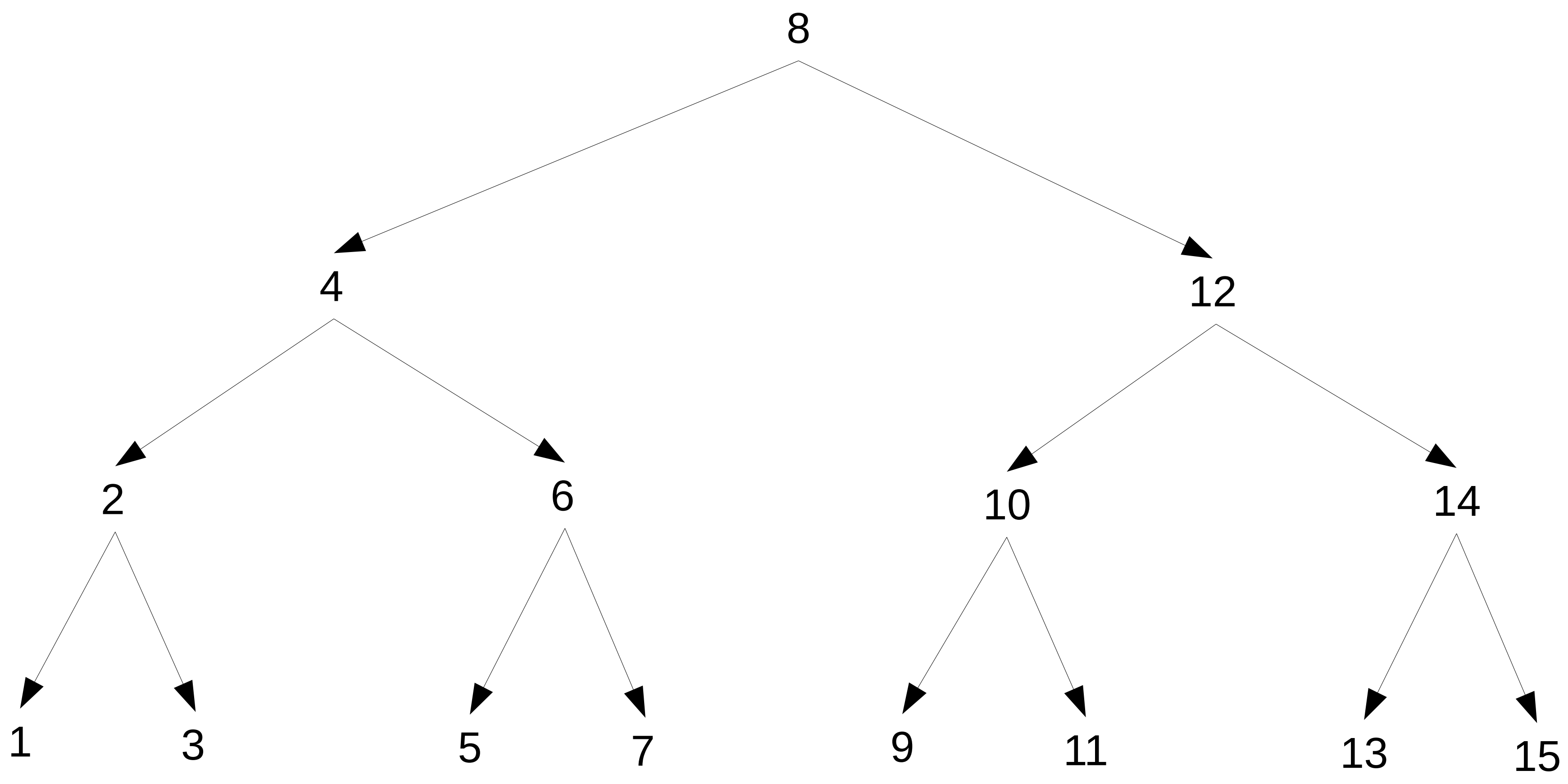
- Index Scan
  - Ordered
- Bitmap Index Scan
- Index Only Scan
  - Ordered

Index Access Methods in PostgreSQL 9.5

- B-tree
- GiST
- SP-GiST
- GIN
- BRIN

- (Hash)

B-tree



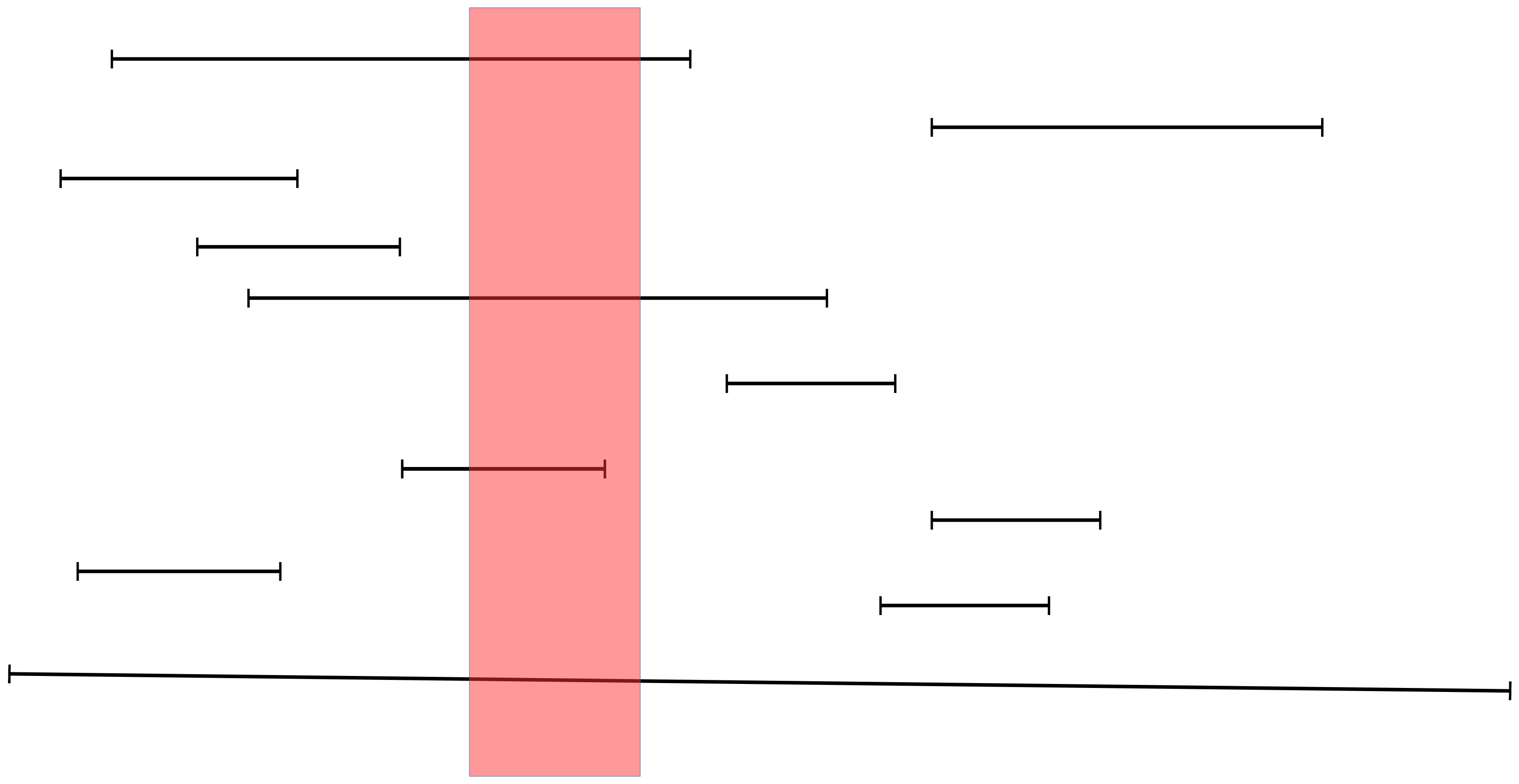
B-tree operators

=	Equals
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
~<~	Variations on text_pattern_ops
~>~	
~<=~	
~>=~	

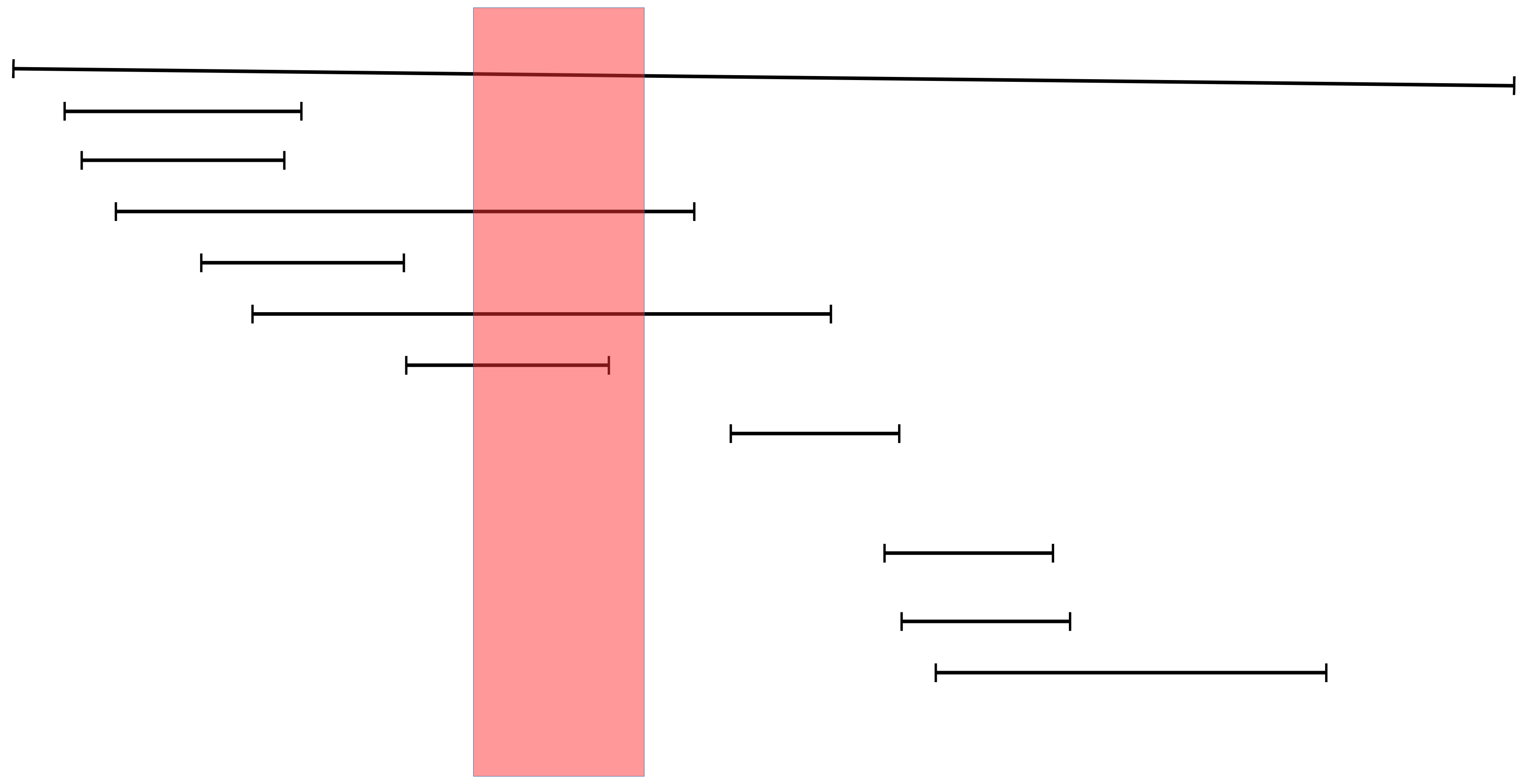
text_pattern_ops

```
# CREATE INDEX ON product (name text_pattern_ops);

# EXPLAIN SELECT * FROM product
                  WHERE name LIKE 'aa%';

                             QUERY PLAN
--------------------------------------------------------------------------
 Bitmap Heap Scan on product
   Filter: (name ~~ 'aa%'::text)
   ->  Bitmap Index Scan on product_name_idx1
         Index Cond: ((name ~>=~ 'aa'::text) AND (name ~<~ 'ab'::text))
```

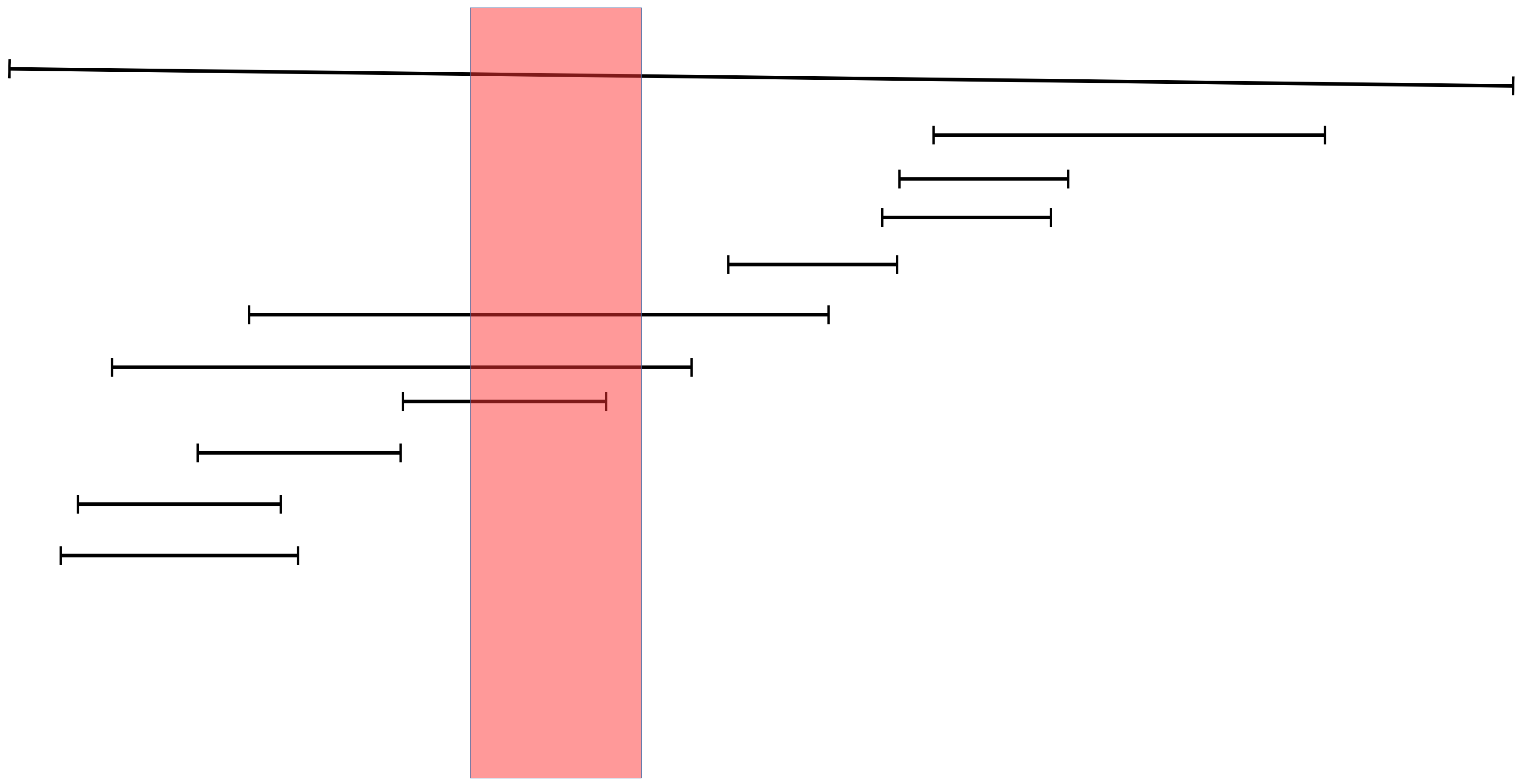
GiST



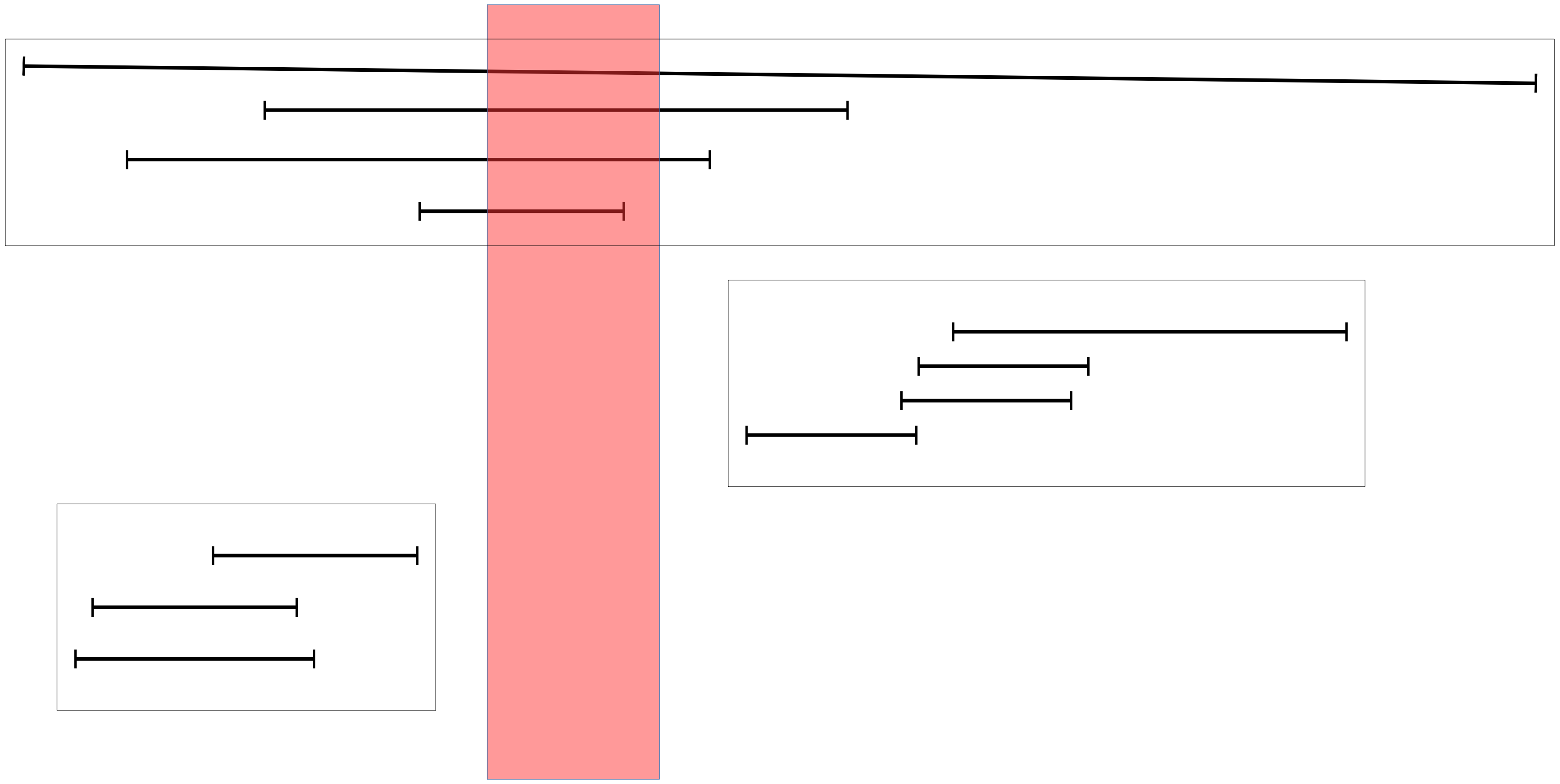
Sort by min



Sort by max



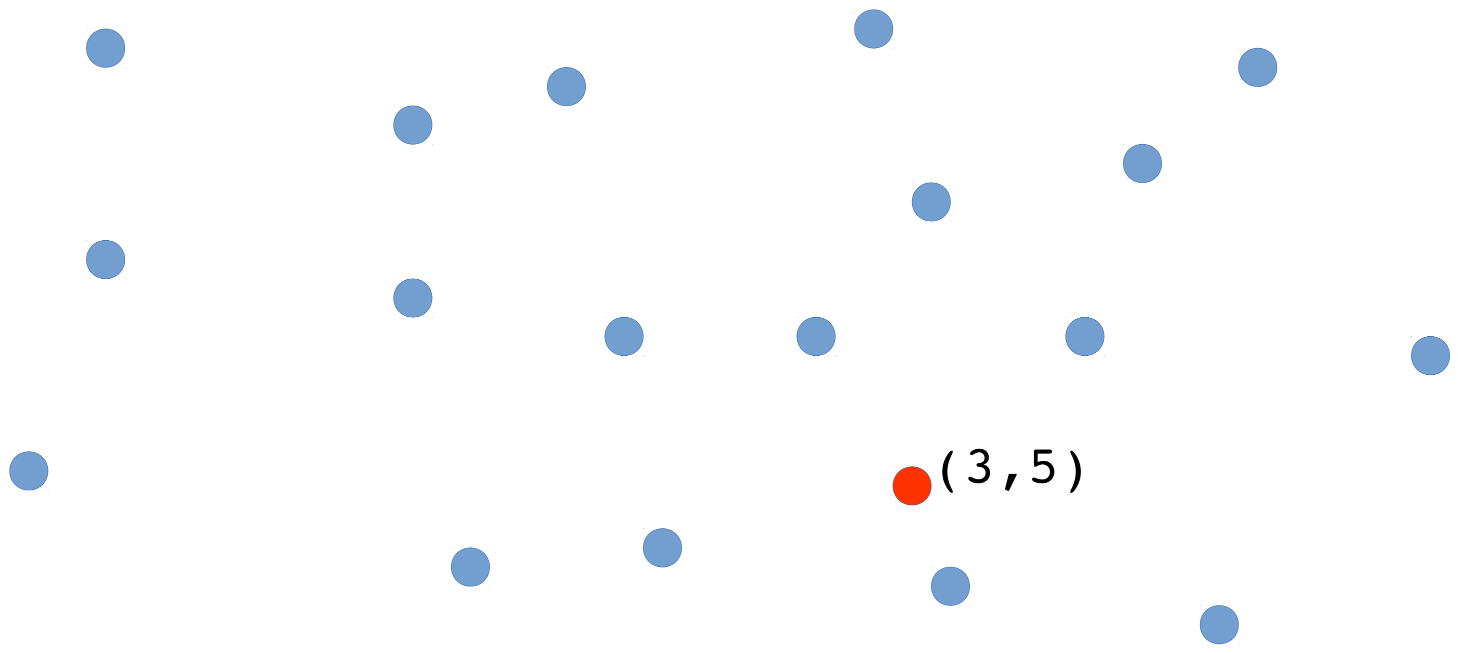
Group into clusters



GiST operators

<<	Left of
&&	Overlaps
>>	Right of
@>	Contains
<@	Contained by
<->	Distance

Nearest Neighbor Search



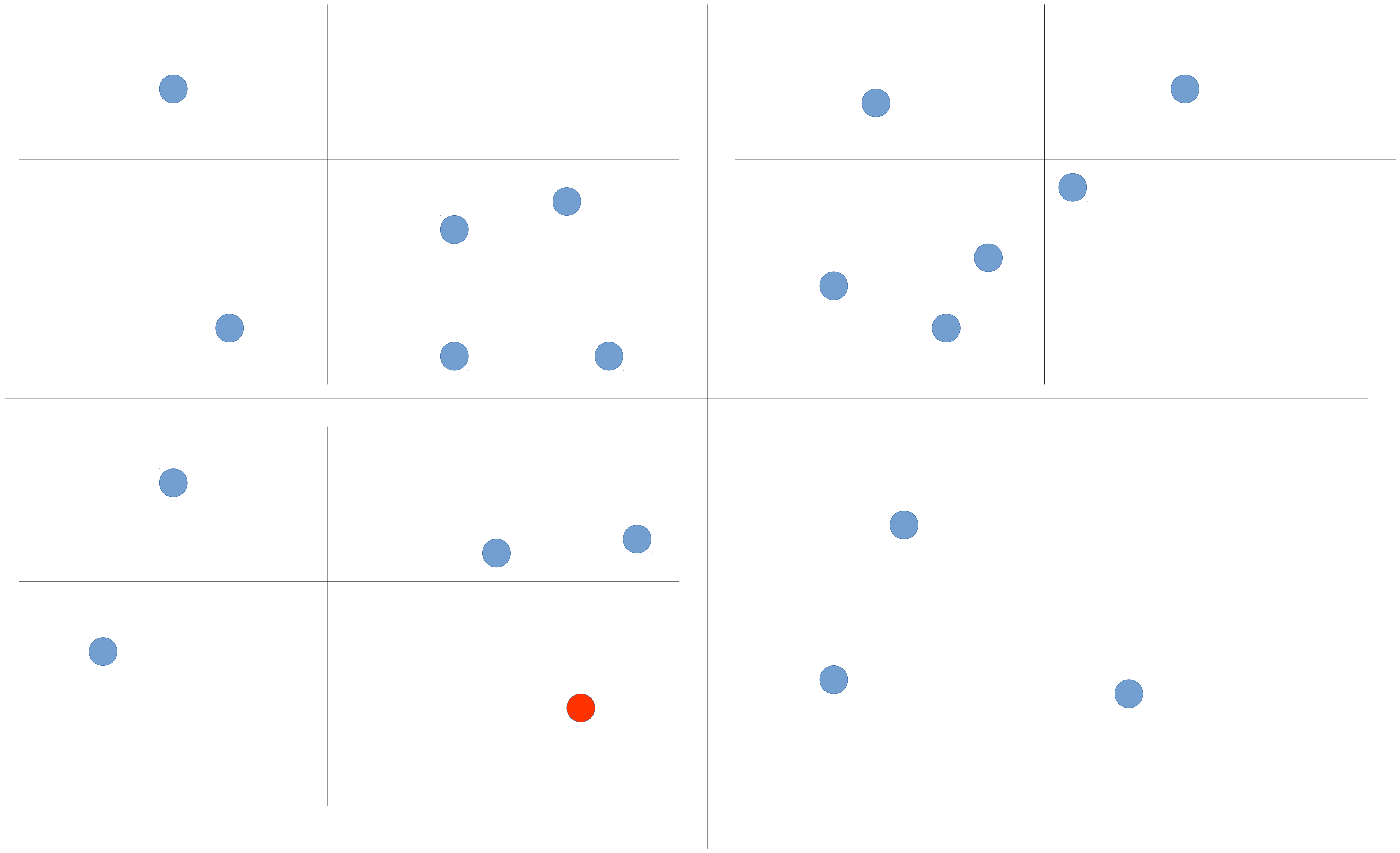
```
SELECT * FROM people
ORDER BY point <-> '(3,5)' LIMIT 5;
```

Exclusion Constraints

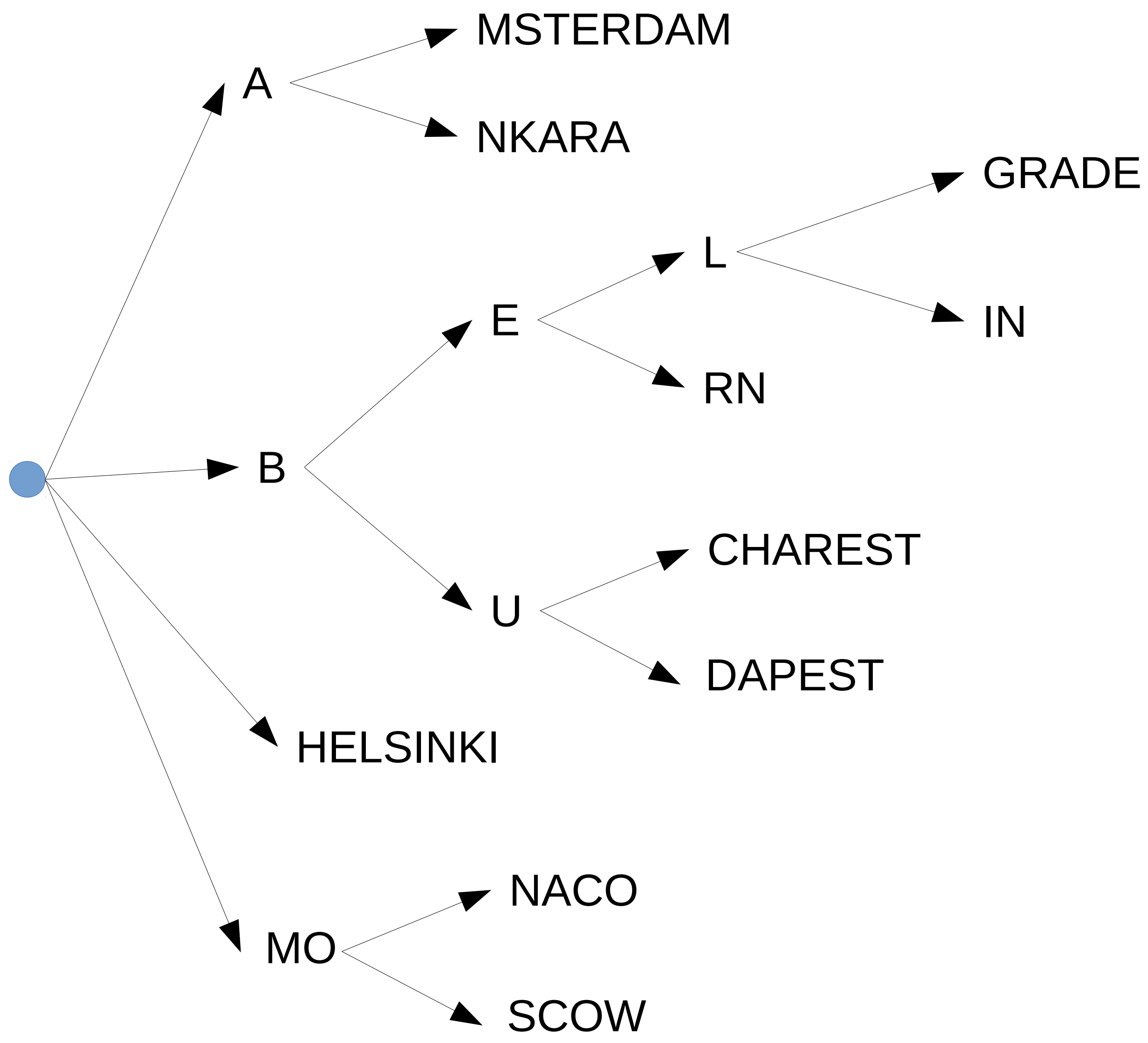
```
# ALTER TABLE allowed_network
    ADD CONSTRAINT allowed_network_address_excl
    EXCLUDE USING gist (
        address WITH &&
    );
```

```
# ALTER TABLE meeting
    ADD CONSTRAINT meeting_room_excl
    EXCLUDE USING gist (
        period WITH &&,
        room_id WITH =
    );
```

Space-Partitioned GiST



Prefix Tree


A
MSTERDAM
NKARA
B
E
L
GRADE
IN
RN
U
CHAREST
DAPEST
HELSINKI
MO
NACO
SCOW

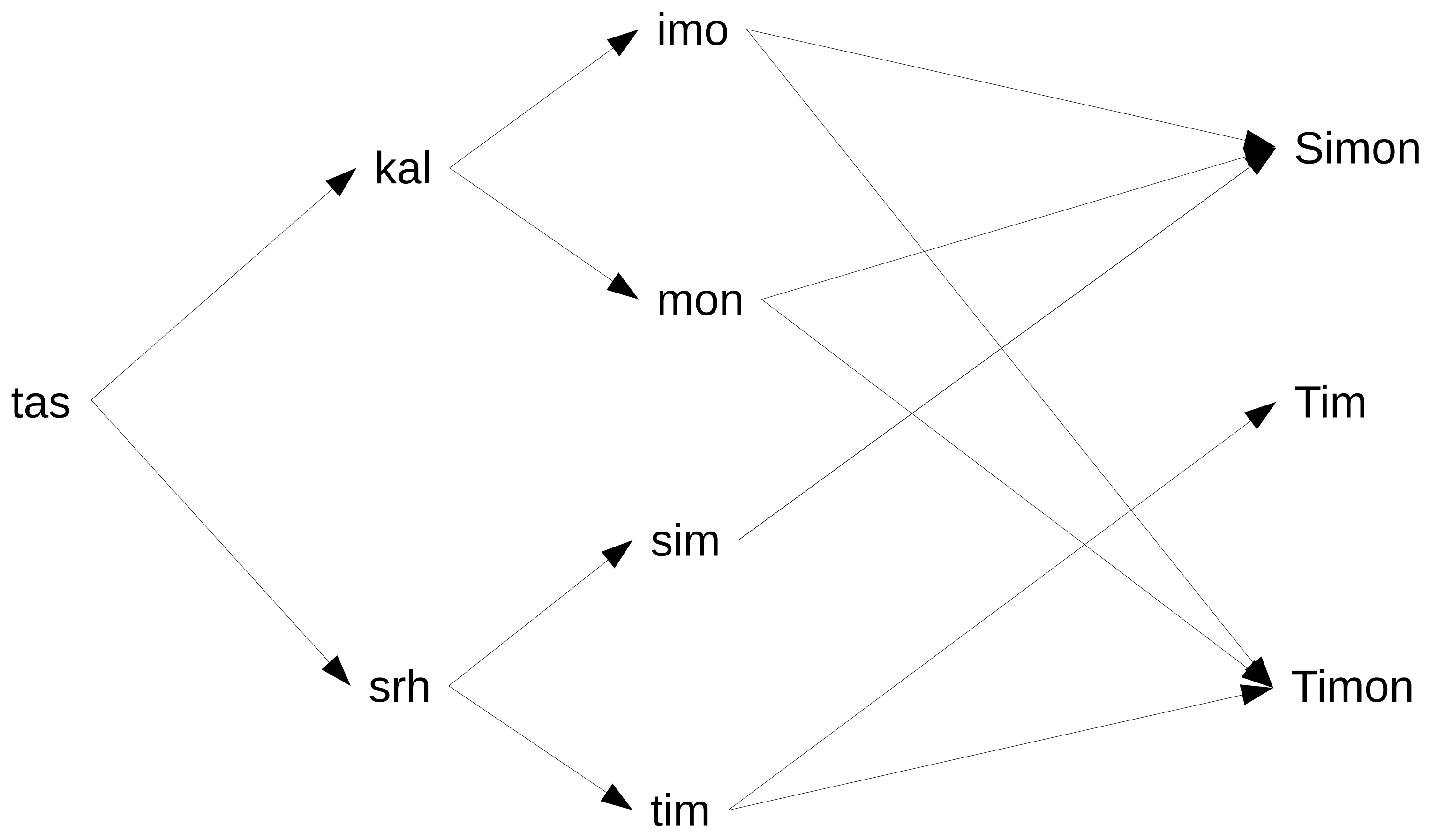
GIN

$$\begin{tikzcd}
 & & 1 \arrow[dr] \arrow[dddddr] & \\
 & 3 \arrow[ur] \arrow[dr] & & \{1\} \\
 & & 4 \arrow[dr] & \\
6 \arrow[uur] \arrow[ddr] & & & \{4, 9\} \\
 & & 6 \arrow[dr] & \\
 & 7 \arrow[ur] \arrow[dr] & & \{1, 5, 9\} \\
 & & 9 \arrow[uuur] \arrow[ur] &
\end{tikzcd}$$

pg_trgm

```
# CREATE EXTENSION pg_trgm;

# CREATE INDEX ON people USING gin (name gin_trgm_ops);

# EXPLAIN SELECT * FROM people
                   WHERE name ~ '.*(a|b)cd.*';


                               QUERY PLAN
-----------------------------------------------------------------------
 Bitmap Heap Scan on people
   Recheck Cond: (name ~ '.*(a|b)cd.*'::text)
   ->  Bitmap Index Scan on people_name_idx
         Index Cond: (name ~ '.*(a|b)cd.*'::text)
```

pg_trgm



hstore

```
# CREATE EXTENSION hstore;

# CREATE INDEX ON server USING gin (attributes);

# EXPLAIN SELECT * FROM server
                 WHERE attributes @> 'market => en';


                            QUERY PLAN
--------------------------------------------------------------------
 Bitmap Heap Scan on server
   Recheck Cond: (attributes @> '"market"=>"en"'::hstore)
   ->  Bitmap Index Scan on server_attributes_idx
         Index Cond: (attributes @> '"market"=>"en"'::hstore)
```

JSONB

```
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ],
  "children": [],
  "spouse": null
}
```

ltree

```
# SELECT DISTINCT category FROM product;

                    category
-----------------------------------------------------------------------
book.fiction.fantasy
book.fiction.horror
book.nonfiction.art
book.nonfiction.history
electronic

# CREATE INDEX ON product USING gist (category);

# SELECT * FROM product WHERE category <@ 'book'::ltree;
```

Block Range Index

0: Amsterdam – Astana
1: Athens – Berlin
2: Bern – Bucharest
3: Budapest – Dublin
4: Helsinki – Ljubljana

Amsterdam
Andorra la Vella
Ankara
Astana

Athens
Baku
Belgrade
Berlin

Bern
Bratislava
Brussels
Bucharest

Budapest
Chișinău
Copenhagen
Dublin

Helsinki
Kiev
Lisbon
Ljubljana

Ordering

UPDATE cities SET name='Zagreb' WHERE ...

0: Amsterdam – Zagreb
1: Athens – Zagreb
2: Bern – Zagreb
3: Budapest – Zagreb
4: Helsinki – Zagreb

Amsterdam
Zagreb
Ankara
Astana

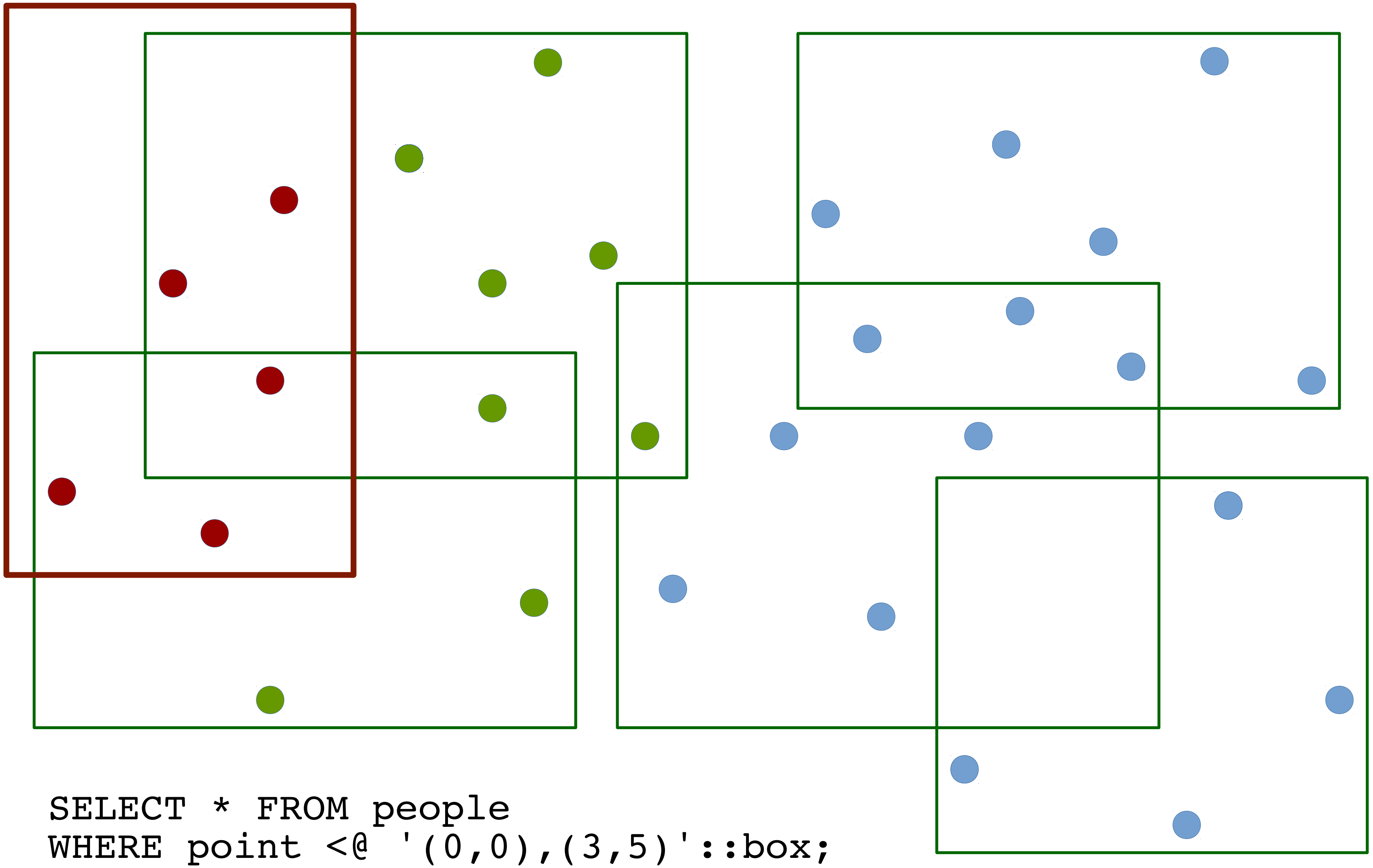
Athens
Baku
Zagreb
Berlin

Bern
Zagreb
Brussels
Bucharest

Budapest
Zagreb
Copenhagen
Dublin

Helsinki
Kiev
Zagreb
Ljubljana

Points



```
SELECT * FROM people
WHERE point <@ '(0,0),(3,5)'::box;
```

Summary

- B-tree
  - Default
  - Unique indexes
- GiST
  - Containment
  - KNN search
- SP-GIST
  - Non-overlapping
- GIN
  - Multiple values per row
  - Stores duplicates efficiently
- BRIN
  - Containment
  - For ordered data
  - Tiny index