TEMPORAL ALIGNMENT

BY ANTON DIGNÖS, MICHAEL H. BÖHLEN, JOHANN GAMPER

Paul A. Jungwirth

25 September 2024

Papers We Love Portland

OUTLINE

- Background
- Temporal Relational Operators
- Postgres Implementation

THE PROBLEM

LOST INFORMATION

THE PROBLEM

LOST INFORMATION

- finance: market data

THE PROBLEM

LOST INFORMATION

- finance: market data
- questionnaires: changing questions, options

THE PROBLEM

LOST INFORMATION

- finance: market data
- questionnaires: changing questions, options
- e-commerce: product price, other attributes

THE PROBLEM

LOST INFORMATION

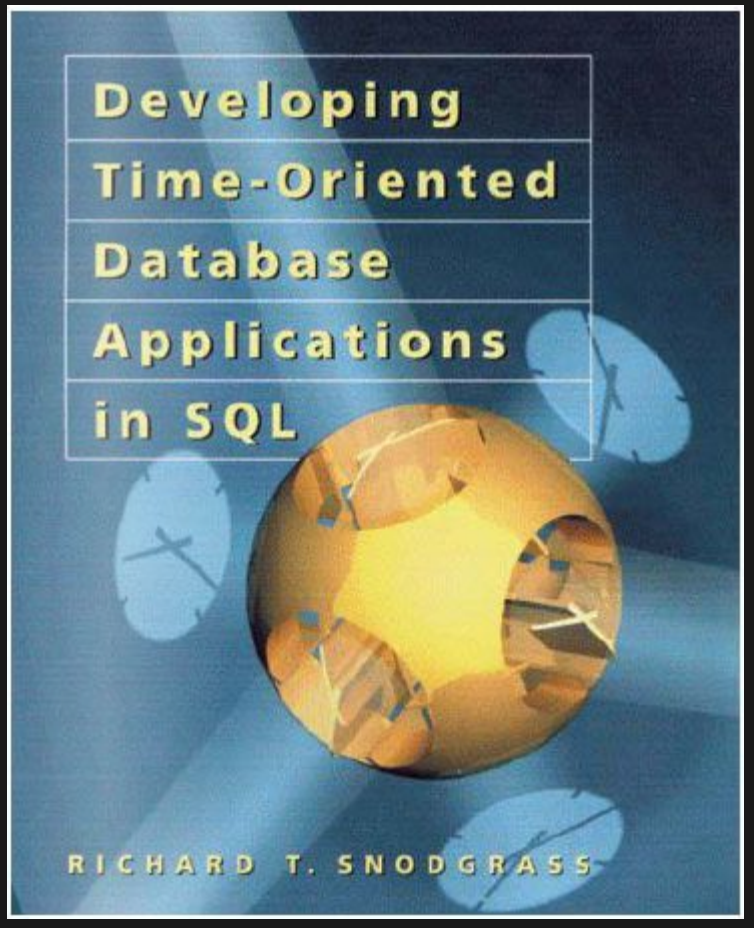
- finance: market data
- questionnaires: changing questions, options
- e-commerce: product price, other attributes
- real estate: house renovations

THE PROBLEM

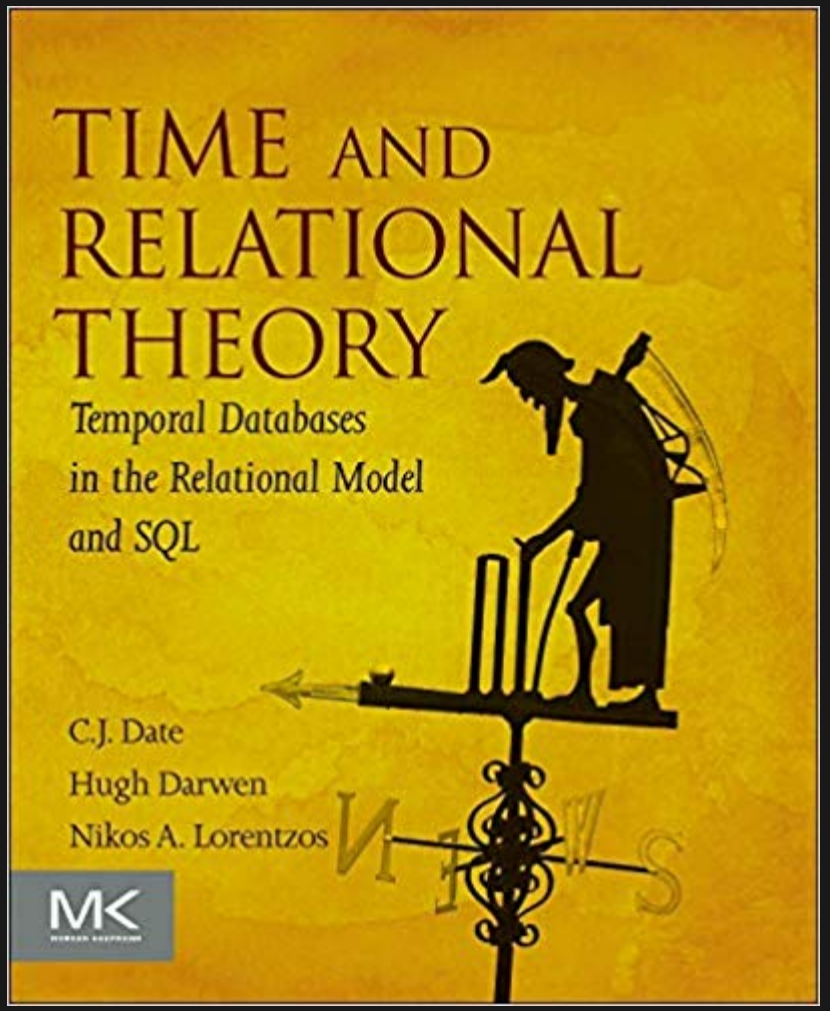
LOST INFORMATION

- finance: market data
- questionnaires: changing questions, options
- e-commerce: product price, other attributes
- real estate: house renovations
- employees: position, salary, employment period

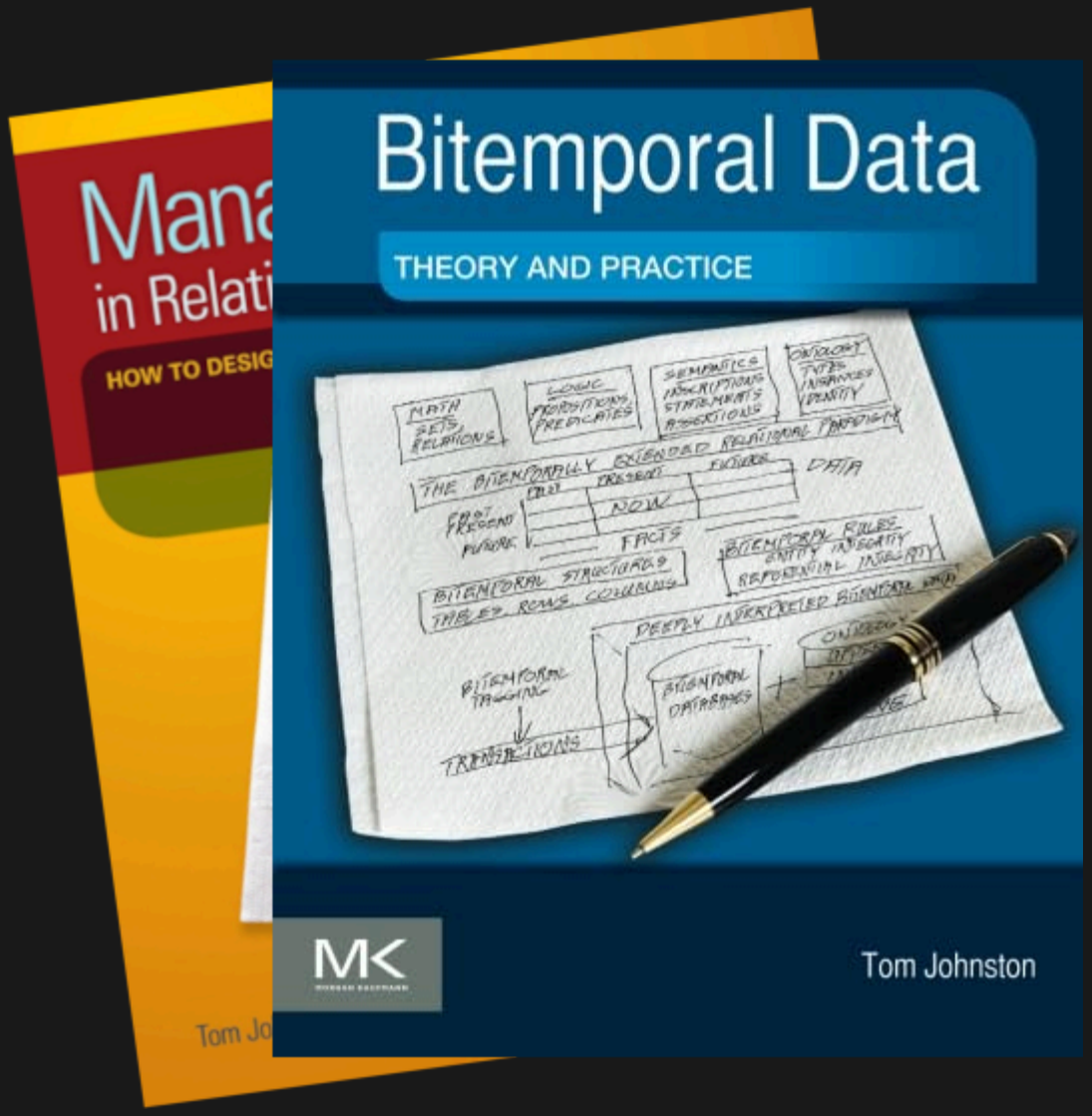
RESEARCH


Developing
Time-Oriented
Database
Applications
in SQL
RICHARD T. SNODGRASS

RESEARCH


TIME AND RELATIONAL THEORY
Temporal Databases in the Relational Model and SQL
C.J. Date
Hugh Darwen
Nikos A. Lorentzos
MK

RESEARCH


Mana
in Relati
HOW TO DESIG
Tom Jo
Bitemporal Data
THEORY AND PRACTICE
MATH
SETS
RELATIONS
LOGIC
PROPOSITIONS
PREDICATES
SEMANTICS
INTERPRETATIONS
STATEMENTS
ASSERTIONS
ONTOLOGY
TYPES
INSTANCES
IDENTITY
THE BITEMPORALLY EXTENDED RELATIONAL MODEL
DATA
PAST
PRESENT
FUTURE
NOW
FACTS
BITEMPORAL RULES
ENTITY INTEGRITY
REFERENTIAL INTEGRITY
BITEMPORAL STRUCTURES
TABLES ROWS COLUMNS
DEEPLY INTERPRETED BITEMPORAL MODEL
BITEMPORAL TAGGING
TRANSACTIONS
BITEMPORAL DATABASE
MK
Tom Johnston

TEMPORAL IS DISTINCT FROM TIME-SERIES

time-series	temporal
single timestamp	two timestamps
records events	records things, "versions"
IoT sensors, finance	auditing, history
challenge is scale	challenge is complexity
partitioning	ranges, exclusion constraints
TimescaleDB	`periods`, `pg_bitemporal`

TWO DIMENSIONS

Application Time	System Time
history of the thing	history of the database
application features	auditing, compliance
user can edit	immutable
maintained by you	maintained by Postgres
constraints matter	look Ma, no hands!
`periods`, `pg_bitemporal`	`temporal_tables`, `pg_audit_log`
nothing	Rails: `papertrail`, `audited`, `chronomodel`, ...

TERMINOLOGY

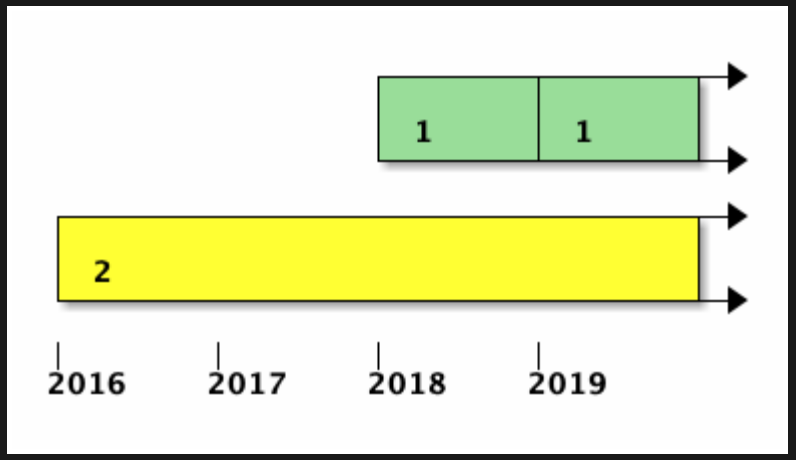
Snodgrass	valid time	transaction time
Fowler	actual time	record time
Date/Darwen/Lorentzos	stated time	logged time
Johnston	effective time/ state time	assertion time
SQL:2011	application time	system time

SIMPLE EXAMPLE

products				
id	name	price	valid_from	valid_til
1 1	shoe shoe	$5 $7	Jan 2018 Jan 2019	Jan 2019
2	snow	$2	Jan 2016	

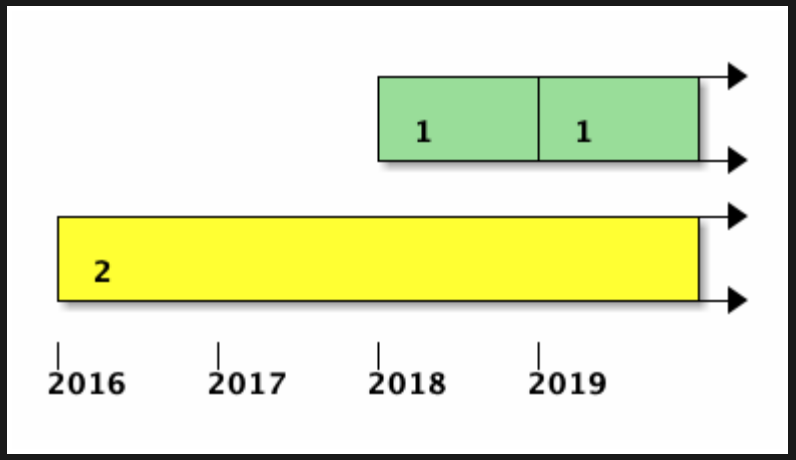
SIMPLE EXAMPLE

products				
id	name	price	valid_from	valid_til
1 1	shoe shoe	$5 $7	Jan 2018 Jan 2019	Jan 2019
2	snow	$2	Jan 2016	


1
1
2
2016
2017
2018
2019

SIMPLE EXAMPLE

products			
id	name	price	valid_at
1	shoe	$5	[Jan 2018, Jan 2019)
1	shoe	$7	[Jan 2019,)
2	snow	$2	[Jan 2016,)


1
1
2
2016
2017
2018
2019

SQL:2011

- `PERIOD FOR valid_at (valid_from, valid_til)`

CONSTRAINTS

- `PRIMARY KEY (id, valid_at WITHOUT OVERLAPS)`
- `UNIQUE (id, valid_at WITHOUT OVERLAPS)`
- `FOREIGN KEY (employee_id, PERIOD valid_at)`
  `REFERENCES employees (id, PERIOD valid_at)`

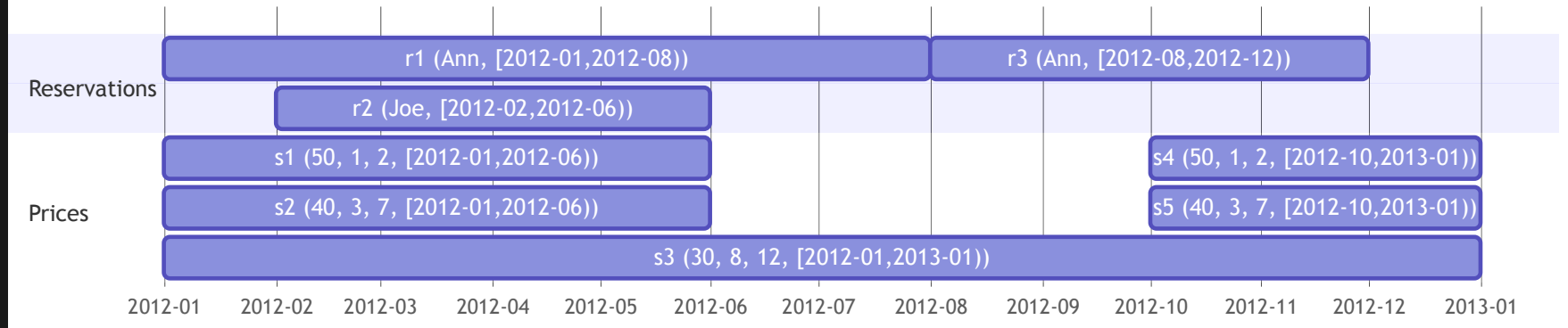
DML

- `UPDATE t FOR PORTION OF valid_at FROM x TO y`
- `DELETE FROM t FOR PORTION OF valid_at FROM x TO y`

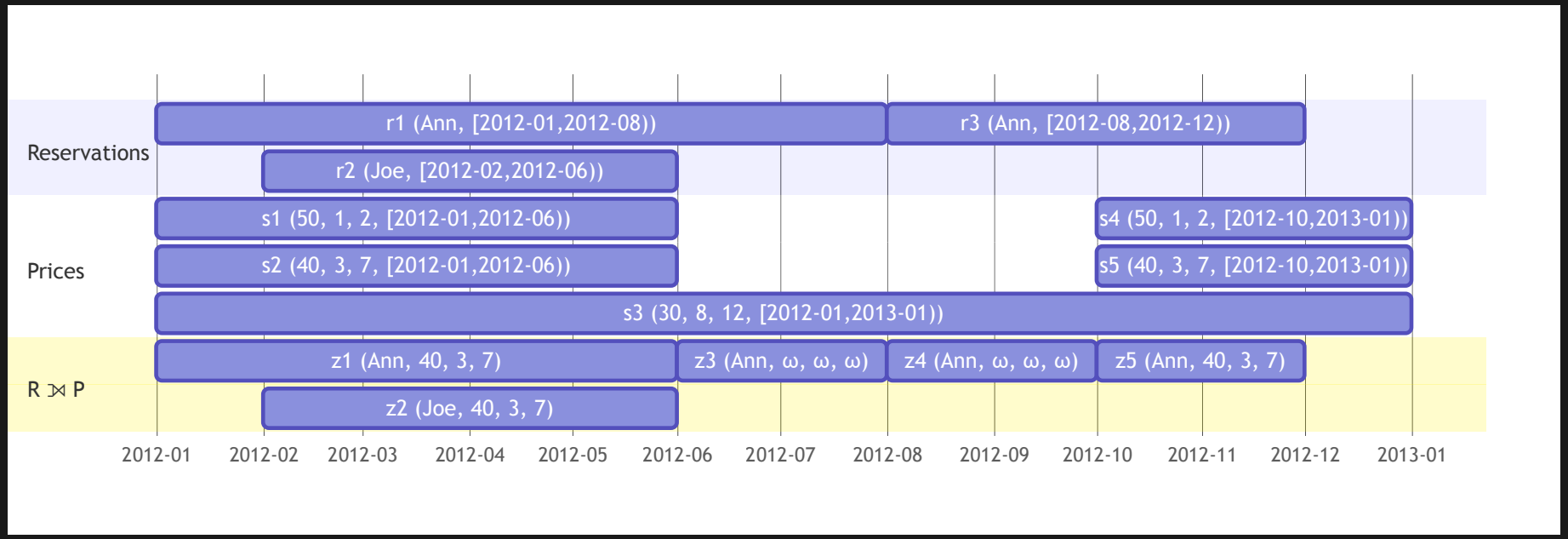
FULL EXAMPLE

N	T
Ann	[2012-01,2012-08)
Joe	[2012-02,2012-06)
Ann	[2012-08,2012-12)

A	Min	Max	T
50	1	2	[2012-01,2012-06)
40	3	7	[2012-01,2012-06)
30	8	12	[2012-01,2013-01)
50	1	2	[2012-10,2013-01)
40	3	7	[2012-10,2013-01)


Reservations
Prices
r1 (Ann, [2012-01,2012-08))
r3 (Ann, [2012-08,2012-12))
r2 (Joe, [2012-02,2012-06))
s1 (50, 1, 2, [2012-01,2012-06))
s4 (50, 1, 2, [2012-10,2013-01))
s2 (40, 3, 7, [2012-01,2012-06))
s5 (40, 3, 7, [2012-10,2013-01))
s3 (30, 8, 12, [2012-01,2013-01))
2012-01
2012-02
2012-03
2012-04
2012-05
2012-06
2012-07
2012-08
2012-09
2012-10
2012-11
2012-12
2013-01

FULL EXAMPLE



GROUP-BASED OPERATORS

- Many input tuples
- One output tuple

π	projection
ϑ	aggregation
∪	union
−	difference
∩	intersection

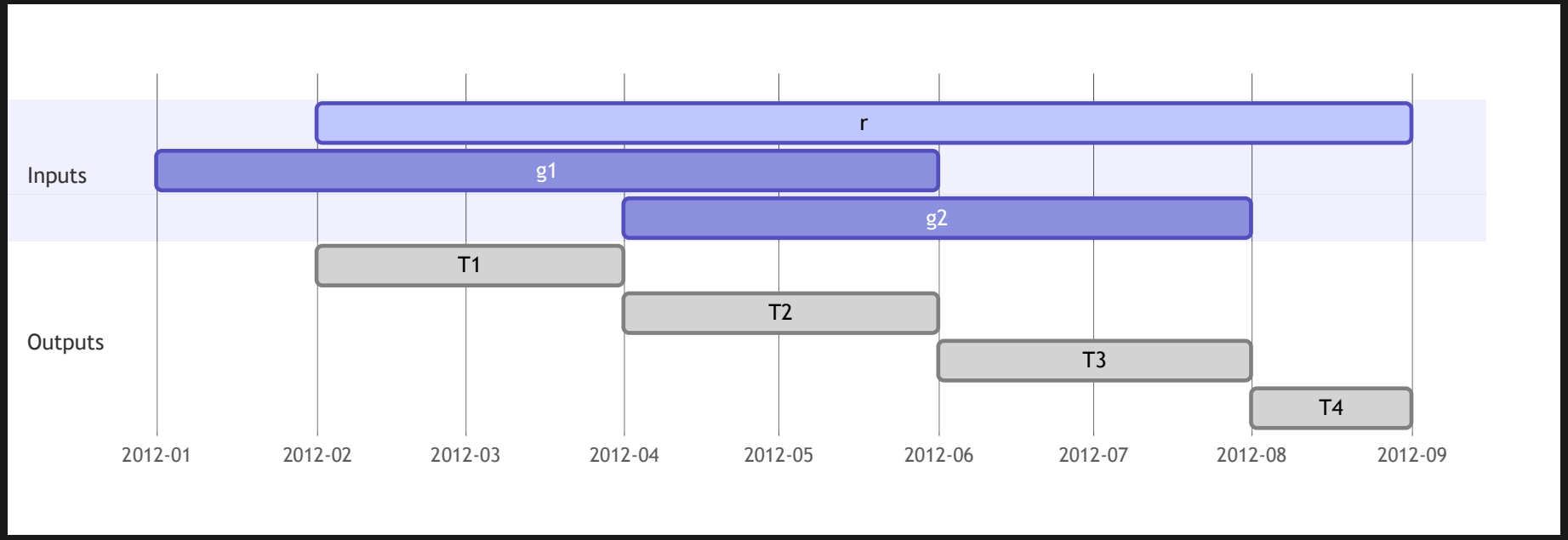
TUPLE-BASED OPERATORS

- At most one input value (per argument)
- One output value

σ	selection
$\times$	cross join
$\bowtie$	inner join
$⟕$	left outer join
$⟖$	right outer join
$⟗$	full outer join
$\triangleright$	antijoin

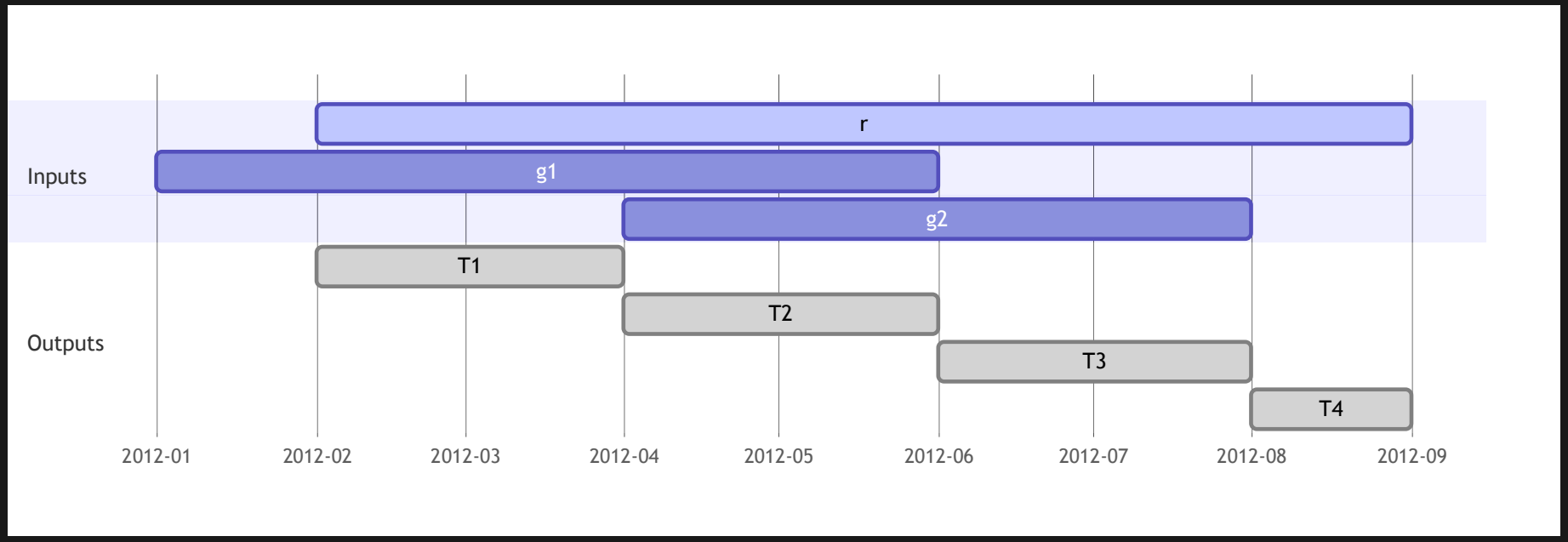
SPLIT FUNCTION

- For group-based operators (π, ϑ, ∪, −, ∩).
- Cut at the start/end of tuples in the group



SPLIT FUNCTION

$$T \in split(r, \mathbf{g}) \iff$$
$$T \subseteq r.T \land$$
$$\forall g \in \mathbf{g}(g.T \cap T = \varnothing \lor T \subseteq g.T) \land$$
$$\forall T' \supset T(\exists g \in \mathbf{g}(T' \cap g.T \neq \varnothing \land T' \nsubseteq g.T) \lor T' \nsubseteq r.T)$$

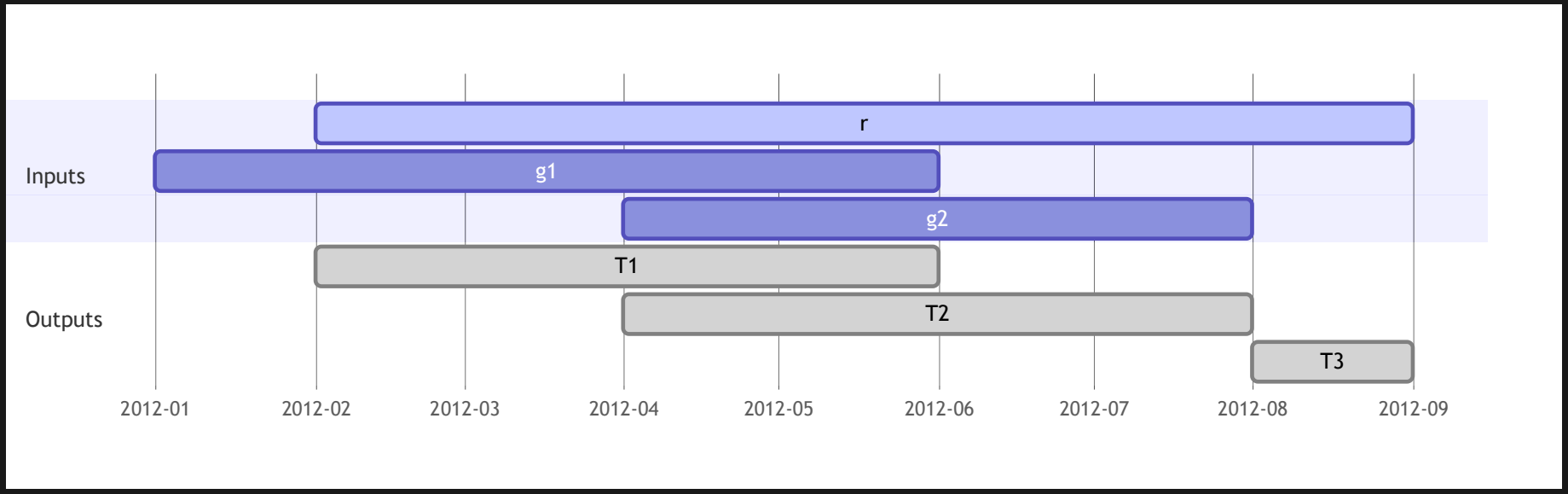


NORMALIZE OPERATOR

$$\tilde{r} \in \mathcal{N}_B(\mathbf{r};\mathbf{s}) \iff \exists r \in \mathbf{r}(\tilde{r}.A = r.A \wedge \tilde{r}.T \in split(r, \{s \in \mathbf{s} \mid s.B = r.B\}))$$

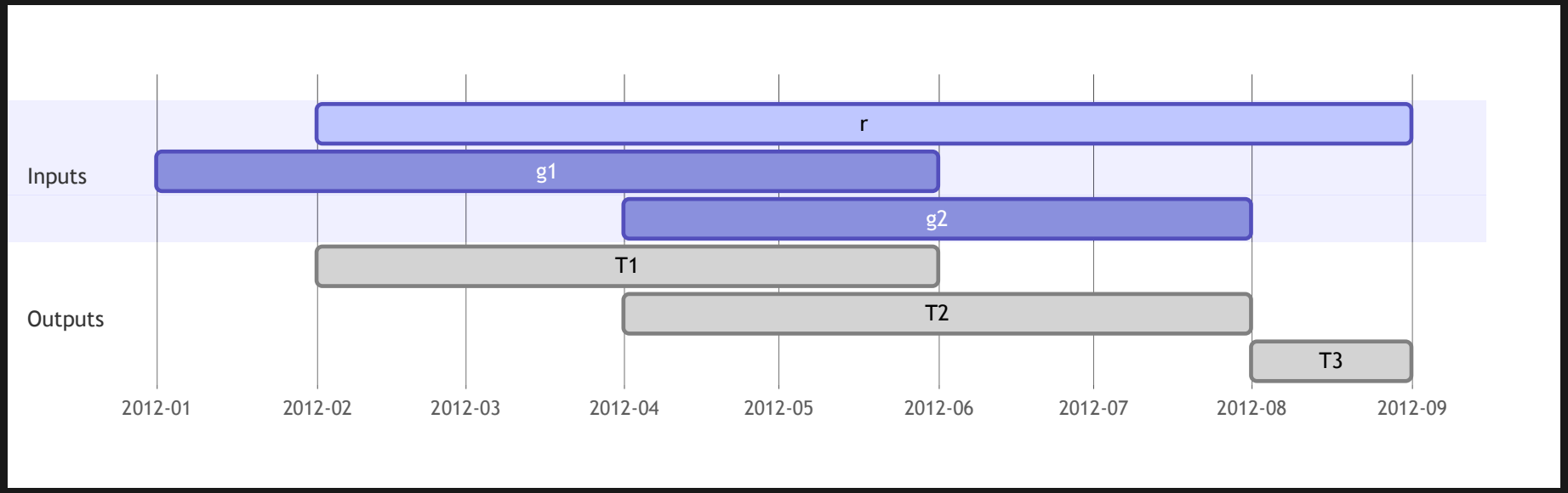
ALIGN FUNCTION

- For tuple-based operators (σ, ×, ⋈, ⟕, ⟖, ⟗, ▷)
- Intersection with each tuple in the group
- Plus any part that is never intersected



ALIGN FUNCTION

$$T \in align(r, \mathbf{g}) \iff$$
$$\exists g \in \mathbf{g}(T = r.T \cap g.T\ ) \wedge T \neq \varnothing \vee$$
$$T \subseteq r.T \wedge \forall g \in \mathbf{g}(g.T \cap T = \varnothing) \wedge$$
$$\forall T' \supset T(\exists g \in \mathbf{g}(T' \cap g.T \neq \varnothing) \vee T' \notin r.T)$$



ALIGN OPERATOR

$$\tilde{r} \in \mathbf{r}\ \Phi_\theta\ \mathbf{s} \iff \exists r \in \mathbf{r}(\tilde{r}.A = r.A \wedge \tilde{r}.T \in \mathit{align}(r,\{s \in \mathbf{s} \mid \theta(r,s)\}))$$

REDUCTION RULES

selection	$\sigma_{\theta}^{T}(\mathbf{r})$	$= \sigma_{\theta}(\mathbf{r})$
projection	$\pi_{B}^{T}(\mathbf{r})$	$= \pi_{B,T}(\mathcal{N}_{B}(\mathbf{r};\mathbf{r})$
aggregation	${}_{B}\vartheta_{F}^{T}(\mathbf{r})$	$= {}_{B,T}\vartheta_{F}(\mathcal{N}_{B}(\mathbf{r};\mathbf{r}))$

REDUCTION RULES

difference	$\mathbf{r} \; -^{T} \; \mathbf{s} \;=\; \mathcal{N}_{\mathsf{A}}(\mathbf{r};\mathbf{s}) \; - \; \mathcal{N}_{\mathsf{A}}(\mathbf{s};\mathbf{r})$
union	$\mathbf{r} \; \cup^{T} \; \mathbf{s} \;=\; \mathcal{N}_{\mathsf{A}}(\mathbf{r};\mathbf{s}) \; \cup \; \mathcal{N}_{\mathsf{A}}(\mathbf{s};\mathbf{r})$
intersection	$\mathbf{r} \; \cap^{T} \; \mathbf{s} \;=\; \mathcal{N}_{\mathsf{A}}(\mathbf{r};\mathbf{s}) \; \cap \; \mathcal{N}_{\mathsf{A}}(\mathbf{s};\mathbf{r})$

REDUCTION RULES

cross join	$\mathbf{r} \times^{T} \mathbf{s} = \alpha((\mathbf{r} \Phi_{true} \mathbf{s}) \bowtie_{\mathbf{r}.T=\mathbf{s}.T} (\mathbf{s} \Phi_{true} \mathbf{r}))$
inner join	$\mathbf{r} \bowtie^{T} \mathbf{s} = \alpha((\mathbf{r} \Phi_{\theta} \mathbf{s}) \bowtie_{\theta \wedge \mathbf{r}.T=\mathbf{s}.T} (\mathbf{s} \Phi_{\theta} \mathbf{r}))$
left join	$\mathbf{r} ⟕^{T} \mathbf{s} = \alpha((\mathbf{r} \Phi_{\theta} \mathbf{s}) ⟕_{\theta \wedge \mathbf{r}.T=\mathbf{s}.T} (\mathbf{s} \Phi_{\theta} \mathbf{r}))$
right join	$\mathbf{r} ⟖^{T} \mathbf{s} = \alpha((\mathbf{r} \Phi_{\theta} \mathbf{s}) ⟖_{\theta \wedge \mathbf{r}.T=\mathbf{s}.T} (\mathbf{s} \Phi_{\theta} \mathbf{r}))$
full join	$\mathbf{r} ⟗^{T} \mathbf{s} = \alpha((\mathbf{r} \Phi_{\theta} \mathbf{s}) ⟗_{\theta \wedge \mathbf{r}.T=\mathbf{s}.T} (\mathbf{s} \Phi_{\theta} \mathbf{r}))$
antijoin	$\mathbf{r} \triangleright^{T} \mathbf{s} = (\mathbf{r} \Phi_{\theta} \mathbf{s}) \triangleright_{\theta \wedge \mathbf{r}.T=\mathbf{s}.T} (\mathbf{s} \Phi_{\theta} \mathbf{r})$

EXTEND OPERATOR

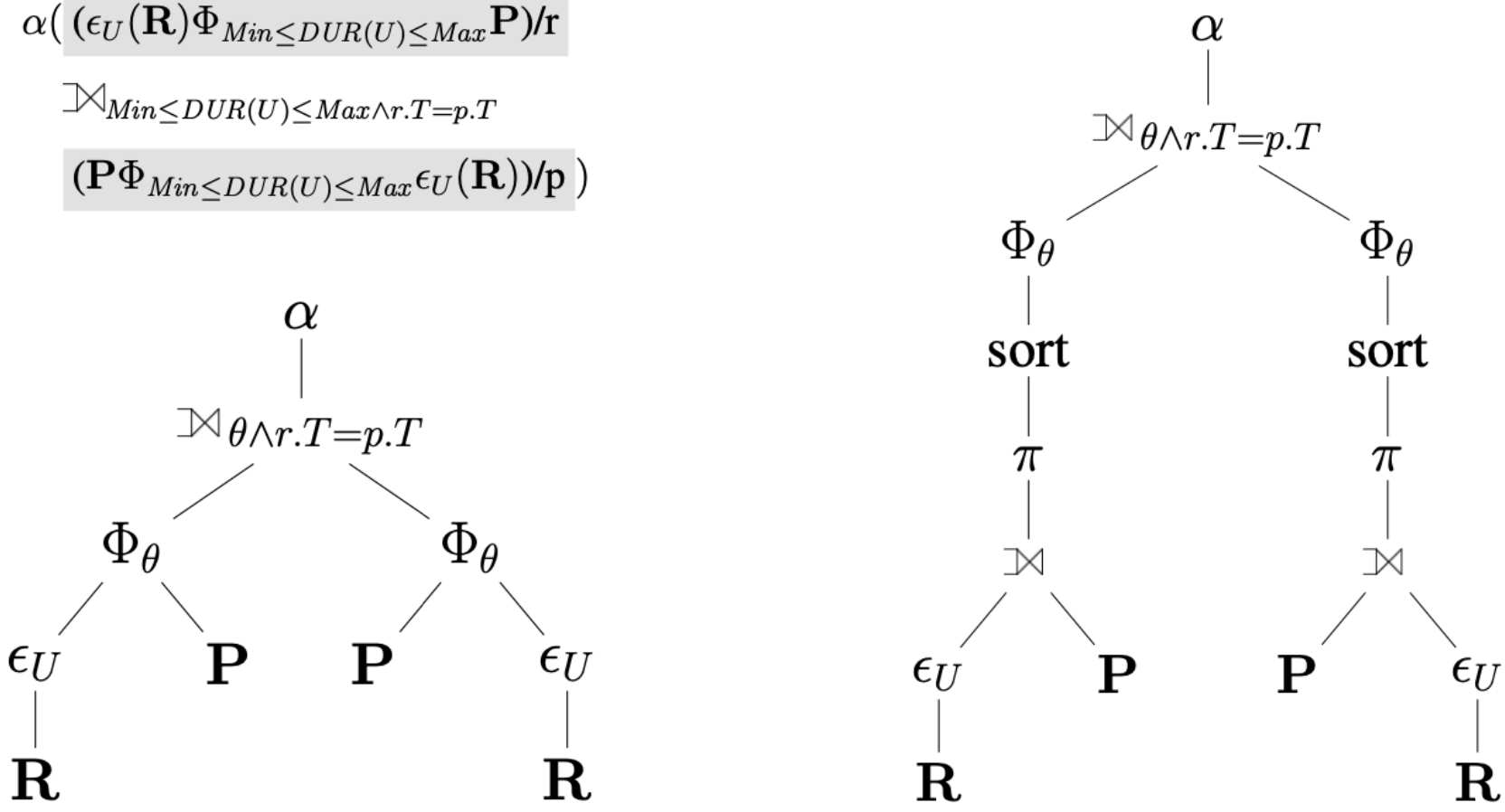
$$
\begin{aligned}
&z \in \varepsilon_U(\mathbf{r}) \iff \\
&\quad \exists r \in \mathbf{r}(z.\mathbf{A}=r.\mathbf{A} \wedge \\
&\qquad\qquad z.U=r.T \wedge \\
&\qquad\qquad z.T=r.T)
\end{aligned}
$$

IMPLEMENTATION

Postgres Query Pipeline

- Parse
- Analyze
- Rewrite
- Optimize
- Execute

EXECUTOR PHASE



(a) Parse Tree

(b) Query Tree

Figure 12: Parse Tree and Query Tree.

LEFT JOIN

```
WITH R AS (SELECT Ts Us, Te Ue, * FROM R)
SELECT ABSORB n, a, min, max, r.Ts, r.Te
FROM
  (R ALIGN P ON DUR(Us, Ue) BETWEEN Min AND Max) r
  LEFT OUTER JOIN
  (P ALIGN R ON DUR(Us, Ue) BETWEEN Min AND MAX) p
  ON DUR(Us, Ue) BETWEEN Min AND Max
     AND r.Ts = p.Ts AND r.Te = p.Te
```

LEFT JOIN

```
WITH R AS (SELECT Ts Us, Te Ue, * FROM R)
SELECT ABSORB n, a, min, max, r.Ts, r.Te
FROM
  (R ALIGN P ON DUR(Us, Ue) BETWEEN Min AND Max) r
  LEFT OUTER JOIN
  (P ALIGN R ON DUR(Us, Ue) BETWEEN Min AND MAX) p
  ON DUR(Us, Ue) BETWEEN Min AND Max
     AND r.Ts = p.Ts AND r.Te = p.Te
```

LEFT JOIN

```
WITH R AS (SELECT Ts Us, Te Ue, * FROM R)
SELECT ABSORB n, a, min, max, r.Ts, r.Te
FROM
  (R ALIGN P ON DUR(Us, Ue) BETWEEN Min AND Max) r
  LEFT OUTER JOIN
  (P ALIGN R ON DUR(Us, Ue) BETWEEN Min AND MAX) p
  ON DUR(Us, Ue) BETWEEN Min AND Max
     AND r.Ts = p.Ts AND r.Te = p.Te
```

LEFT JOIN

```
WITH R AS (SELECT Ts Us, Te Ue, * FROM R)
SELECT ABSORB n, a, min, max, r.Ts, r.Te
FROM
  (R ALIGN P ON DUR(Us, Ue) BETWEEN Min AND Max) r
  LEFT OUTER JOIN
  (P ALIGN R ON DUR(Us, Ue) BETWEEN Min AND MAX) p
  ON DUR(Us, Ue) BETWEEN Min AND Max
     AND r.Ts = p.Ts AND r.Te = p.Te
```

LEFT JOIN

```
WITH R AS (SELECT Ts Us, Te Ue, * FROM R)
SELECT ABSORB n, a, min, max, r.Ts, r.Te
FROM
  (R ALIGN P ON DUR(Us, Ue) BETWEEN Min AND Max) r
  LEFT OUTER JOIN
  (P ALIGN R ON DUR(Us, Ue) BETWEEN Min AND MAX) p
  ON DUR(Us, Ue) BETWEEN Min AND Max
     AND r.Ts = p.Ts AND r.Te = p.Te
```

EVALUATION

To express a temporal outer join in SQL we have to express the join part using overlap predicates on timestamps and evaluate the negative part of the temporal outer join using joins and NOT EXISTS statements.[21]

EVALUATION

```
SELECT  a.*, b.*,
        UNNEST(multirange(a.valid_at) * multirange(b.valid_at)
FROM    a
JOIN    b
ON      a.id = b.id AND a.valid_at && b.valid_at
UNION ALL
...
```

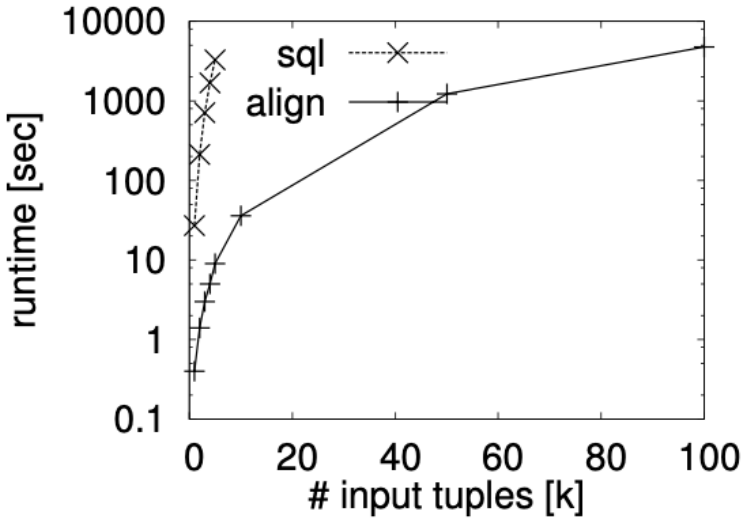
EVALUATION

```
...
SELECT  a.*, (NULL::b).*,
        UNNEST(
          CASE WHEN j.valid_at IS NULL
               THEN multirange(a.valid_at)
               ELSE multirange(a.valid_at) - j.valid_at END
        )
FROM    a
LEFT JOIN (
  SELECT  b.id, range_agg(b.valid_at) AS valid_at
  FROM    b
  GROUP BY b.id
) AS j
ON      a.id = j.id AND a.valid_at && j.valid_at;
```

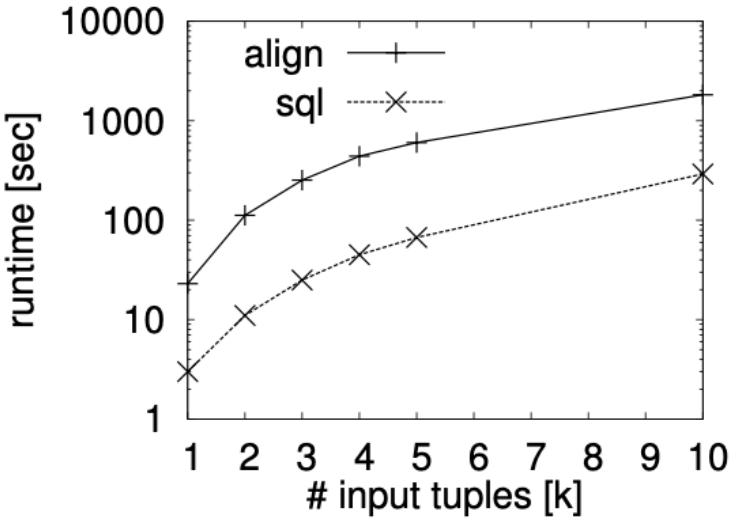
EVALUATION

$$O_1 = \mathbf{r} \,\mathbb{I}\!\bowtie^{T}_{true}\, \mathbf{s}$$

D_{disj} = disjoint intervals, D_{eq} = equal intervals



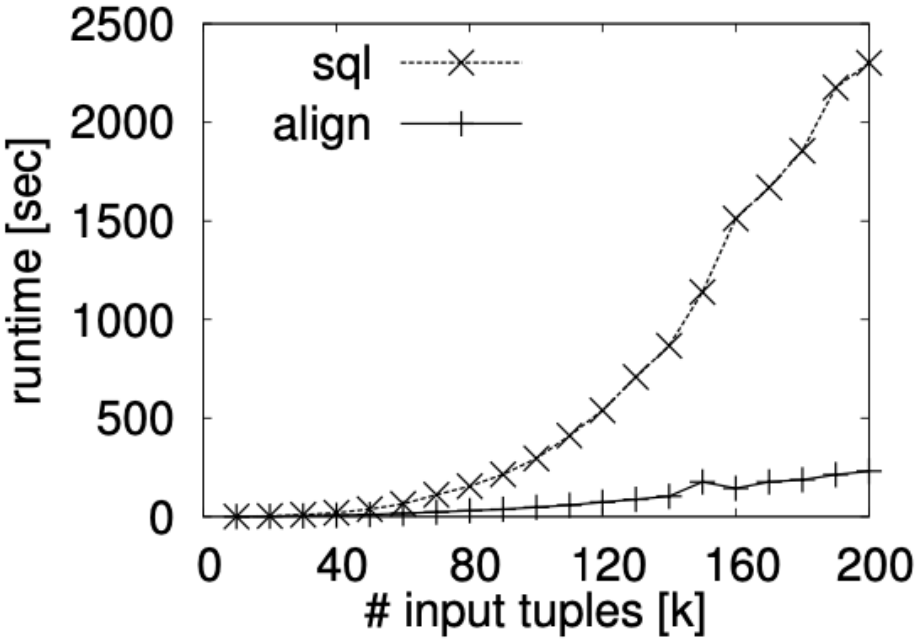
(a) Runtime O_1 on $\mathbf{D}^{disj}$



(b) Runtime O_1 on $\mathbf{D}^{eq}$

EVALUATION

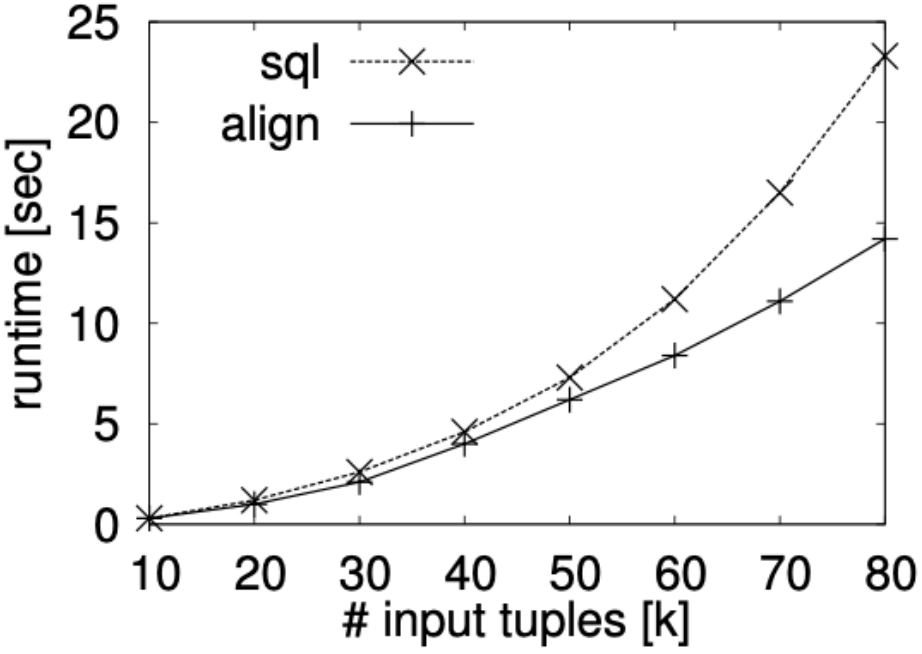
$$O_2 = \mathbf{r} \rtimes^{T}_{Min \leq DUR(\mathbf{r}.T) \leq Max} \mathbf{s}$$



(c) Runtime O_2 on $\mathbf{D}^{rand}$

EVALUATION

$$O_3 = \mathbf{r} \rtimes^{T}_{\mathbf{r}.pcn=\mathbf{s}.pcn} \mathbf{s}$$



(d) Runtime O_3 on *Incumben*

THANKS!

THIS TALK

- https://files.ifi.uzh.ch/boehlen/Papers/modf174-dignoes.pdf
- https://github.com/pjungwir/temporal-alignment-talk

RESEARCH

- https://illuminatedcomputing.com/posts/2017/12/temporal-databases-bibliography/
- https://www2.cs.arizona.edu/~rts/publications.html
- http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=F78723B857463955C76E540DCAB8FDF5?doi=10.1.1.116.7598&rep=rep1&type=pdf
- http://www.zora.uzh.ch/id/eprint/130374/1/Extending_the_kernel.pdf
- https://github.com/pjungwir/temporal_ops

SQL:2011

- https://www.wiscorp.com/SQLStandards.html
- https://sigmodrecord.org/publications/sigmodRecord/1209/pdfs/07.industry.kulkarni.pdf

THANKS!

https://github.com/pjungwir/temporal-alignment-talk