redgate

PostgreSQL Partitioning: Slicing and Dicing for Better Performance and Maintenance

Ryan Booz

Posette 2024

redgate | About me

Ryan Booz

PostgreSQL & DevOps Advocate

@ryanbooz

/in/ryanbooz

www.softwareandbooz.com

youtube.com/@ryanbooz

Agenda

01 The What and Why

02 Types of Partitioning

03 Creating and Maintaining Partitions

04 SQL Tips and Best Practices

01/04
The What and Why


fastest 3G network.
LANE CLOSED
E-ZPass
CASH
NO TRUCKS
NO BUSES
12-1

Large tables don't scale well...

...and table partitioning helps to solve that scaling problem

What Are Partitions?

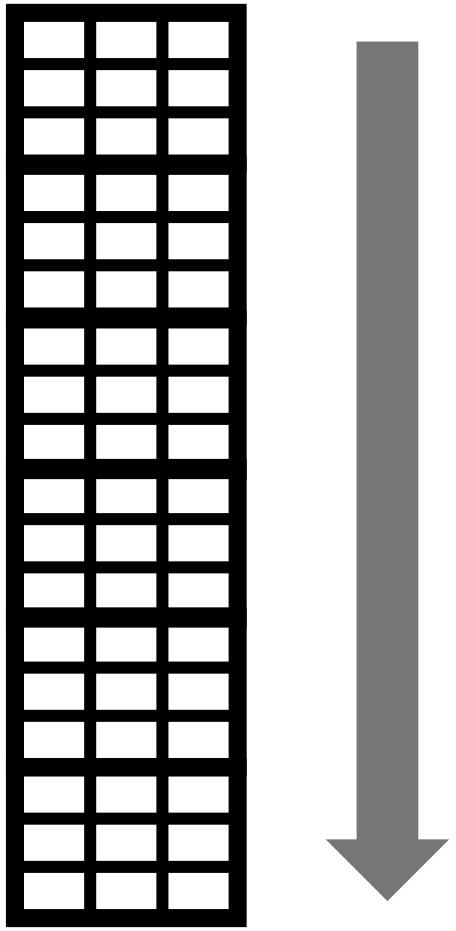
- Regular PostgreSQL tables
  - Schema
  - Tablespace
- Attached to a parent table (children)
- Self-contained indexes
- Can also be parent tables (sub-partitioning)

Declarative Partitioning

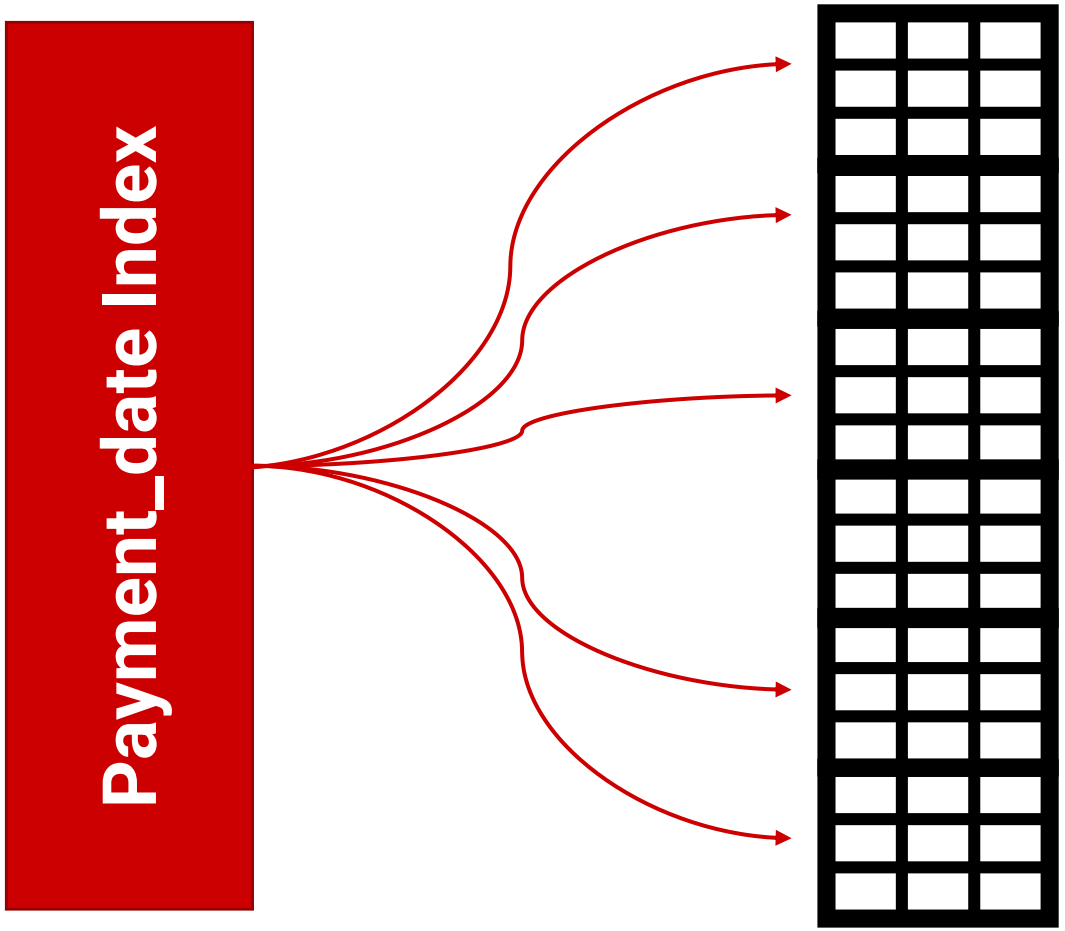
- PostgreSQL 11+
  - Technically PG10, but use 11+
- Identify partitioning key column
- Specify method (RANGE, LIST, HASH)
- Partitions must be created before data arrives
- Default Partitions = catch all = maintenance/trouble

```
CREATE TABLE public.payment (
   payment_id serial4 NOT NULL,
   customer_id int4 NOT NULL,
   rental_id int4 NOT NULL,
   amount numeric(5, 2) NOT NULL,
   payment_date timestamptz NOT NULL,
   CONSTRAINT payment_bak_pkey
      PRIMARY KEY (payment_date, payment_id)
);
```

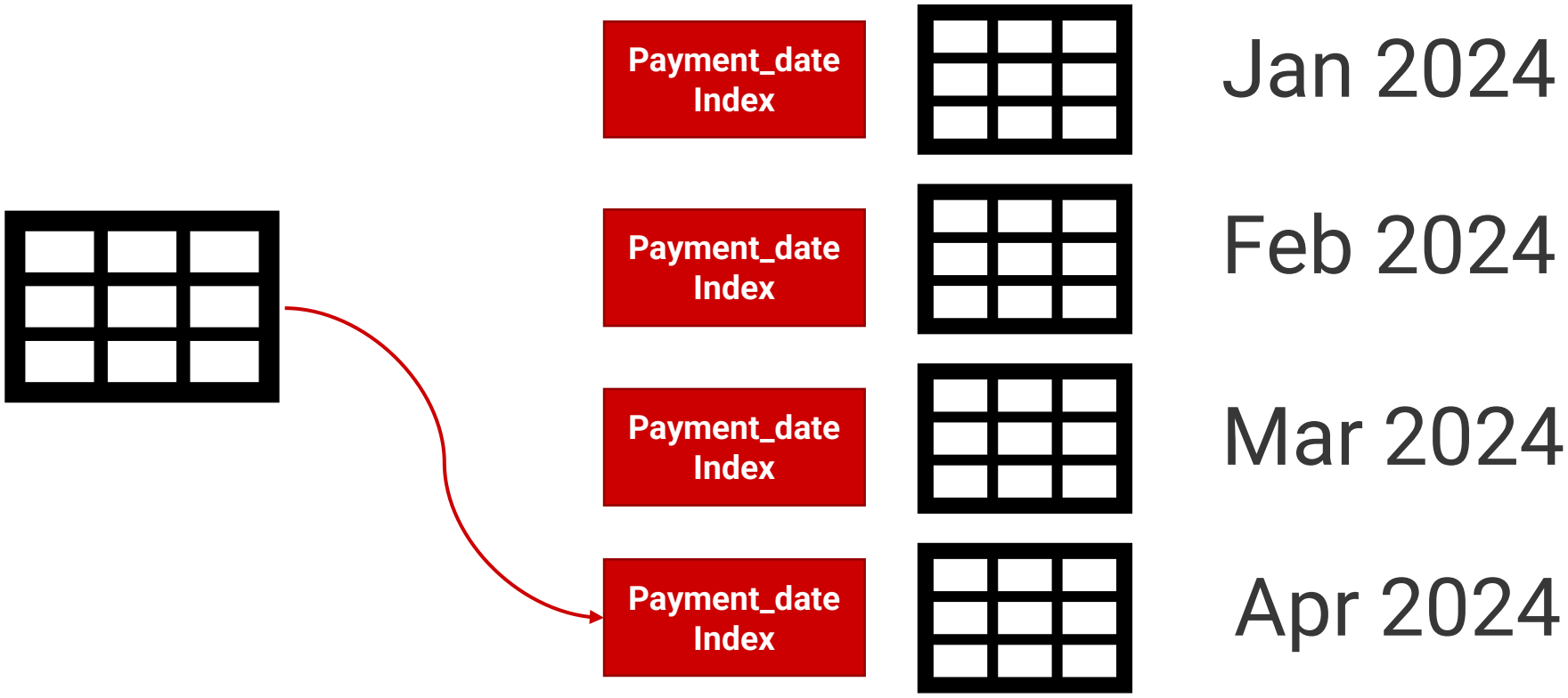
```
SELECT * FROM payment WHERE
    payment_date = '2024-04-19';
```

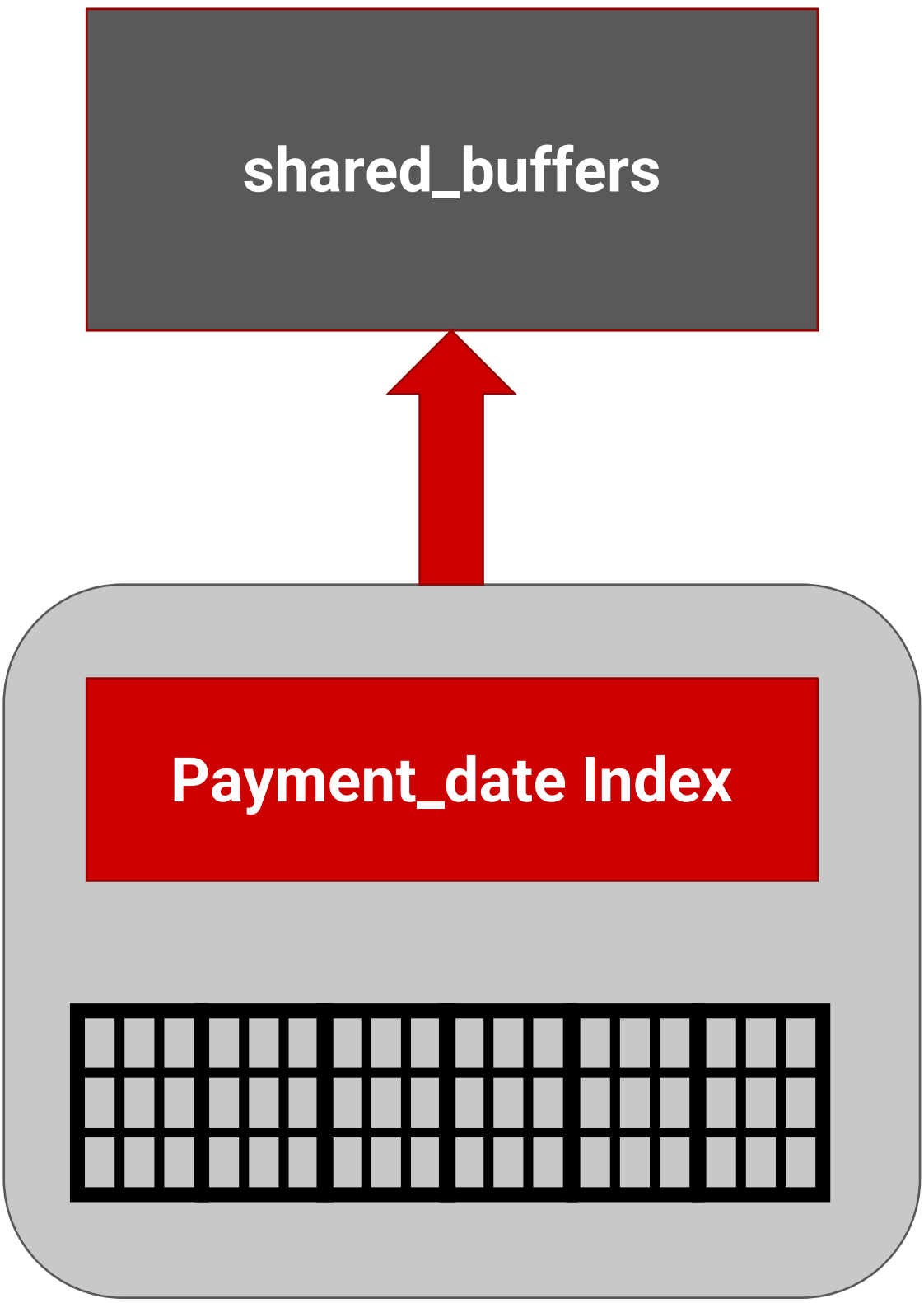


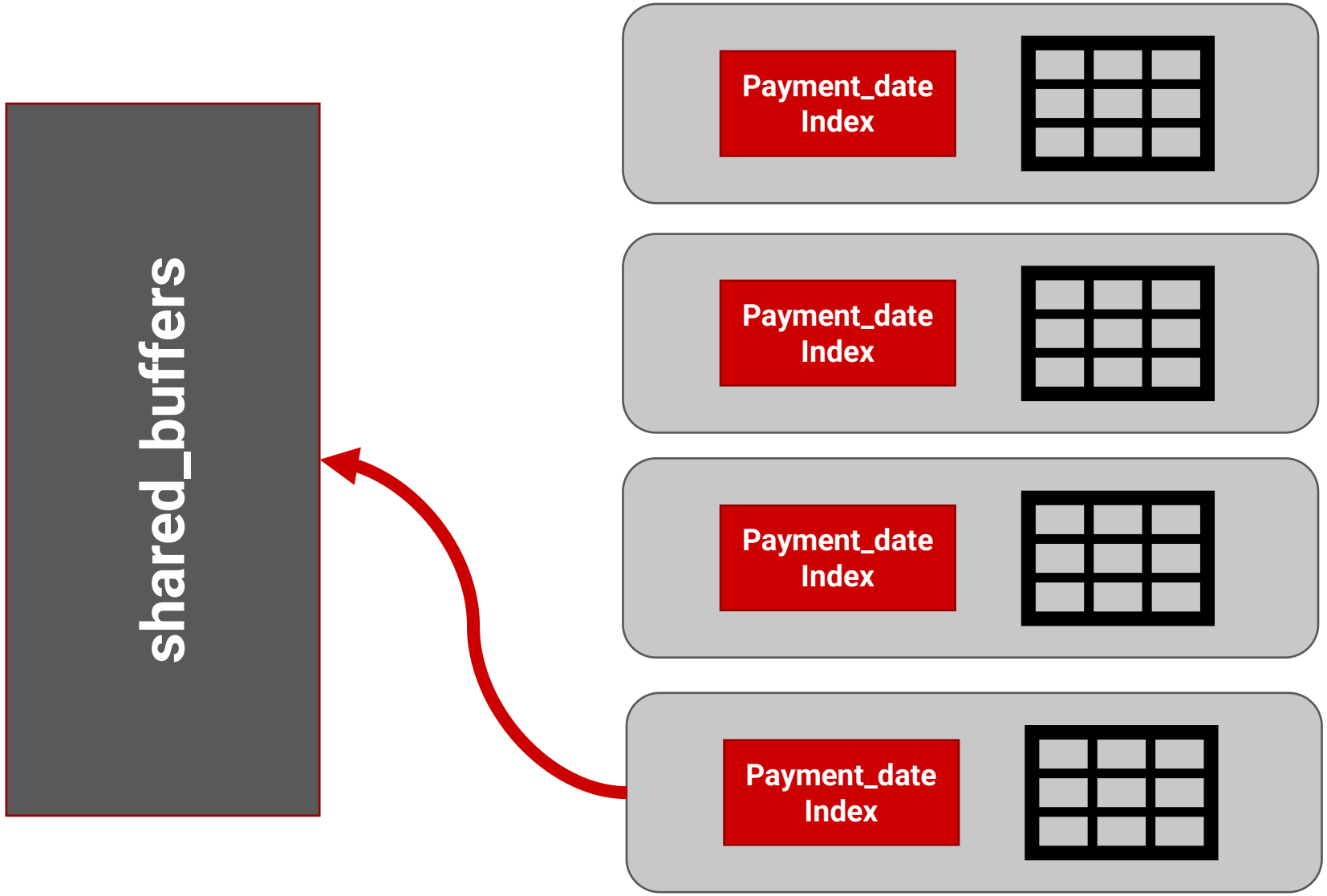
```sql
SELECT * FROM payment WHERE
    payment_date = '2024-04-19';
```



```sql
SELECT * FROM payment WHERE
    payment_date = '2024-04-19';
```




shared_buffers
Payment_date Index


shared_buffers
Payment_date Index
Payment_date Index
Payment_date Index
Payment_date Index

Data Retention/Archiving

- Without partitions, DELETE adds significant overhead
  - MVCC bloat
  - Index maintenance
  - Potential blocking
- With partitions
  - DROP TABLE
  - DETACH PARTITION

02/04

Types of Partitioning

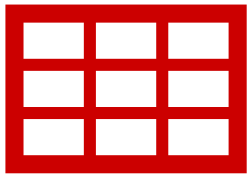
redgate

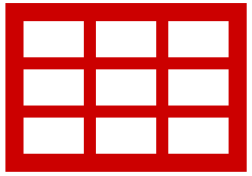
Range

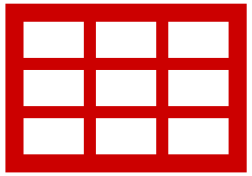
- Most common partitioning method
- Typically
  - Time-series = date/timestamp
  - Object identifiers = integers
- Inclusive of lower value, exclusive of upper value

Range

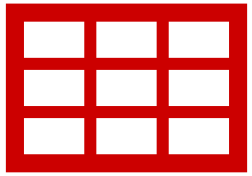
Time-series (logdate)

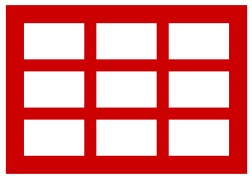

[Jan 1 – Feb 1)

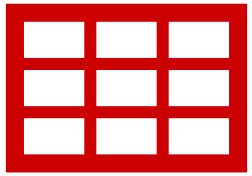

[Feb 1 – Mar 1)


[Mar 1 – Apr 1)

Numeric (category_id)


[1-10)


[10-20)


[20-30)

redgate

List

- Known list of key values
- Product categories/regions
  - 'bike','ball','drone'
  - 'region_1','region_2','region_3'

Hash

- Specify a modulus and remainder
- Partitions contain
  - Key column value/modulus = remainder
- No clear partitioning key
- Generally even distribution of data

03/04
Creating and Maintaining Partitions

```
CREATE TABLE public.payment2 (
   payment_id serial4 NOT NULL,
   customer_id int4 NOT NULL,
   rental_id int4 NOT NULL,
   amount numeric(5, 2) NOT NULL,
   payment_date timestamptz NOT NULL,
   CONSTRAINT payment_bak_pkey
      PRIMARY KEY (payment_date, payment_id)
)
PARTITION BY RANGE (payment_date);
```

```
CREATE TABLE payment2_y2024m01 PARTITION OF payment2
FOR VALUES FROM ('2024-01-01') TO ('2024-02-01');

CREATE TABLE payment2_y2024m02 PARTITION OF payment2
FOR VALUES FROM ('2024-02-01') TO ('2024-03-01');

CREATE TABLE payment2_y2024m03 PARTITION OF payment2
FOR VALUES FROM ('2024-03-01') TO ('2024-04-01');
```

Table administration/DDL is applied through the parent table

redgate

Table/Partition Modifications

```
-- Altering the parent propagates to the children
ALTER TABLE payment2 ADD COLUMN status TEXT;
```

```
-- Adding an index propagates to the children
CREATE INDEX payment2_payment_date_customer_id_idx
  ON public.payment2
     USING btree (payment_date,customer_id);
```

NOT A GLOBAL INDEX!!

redgate

Data Retention

```
-- Partition is deleted and no longer queryable
DROP TABLE payment2_y2024m01;

-- Partition still exists but no longer queryable
ALTER TABLE payment2 DETACH PARTITION payment2_y2024m02;

-- Table must have the same schema and valid, checked
-- constraints that match the partition key
ALTER TABLE payment2 ATTACH PARTITION payment2_y2024m02
FOR VALUES FROM ('2024-02-01') TO ('2024-03-01');
```

Default Partition

- Special "catch all" partition
- Prevents data loss
- Maintenance headache as new partitions are created
- TL;DR; - often a necessary ~~evil~~ help

```
CREATE TABLE payment2_default
      PARTITION OF payment2 DEFAULT;

CREATE TABLE payment2_y2024m01 PARTITION OF payment2
FOR VALUES FROM ('2024-01-01') TO ('2024-02-01');


CREATE TABLE payment2_y2024m02 PARTITION OF payment2
FOR VALUES FROM ('2024-02-01') TO ('2024-03-01');


CREATE TABLE payment2_y2024m03 PARTITION OF payment2
FOR VALUES FROM ('2024-03-01') TO ('2024-04-01');
```

Partition Automation

- Don't plan on manual creation or retention
- Common extensions:
  - pg_partman/pg_cron
  - timescaledb (not declarative partitioning)
  - citus

04/04

Tips and Best Practices

redgate

Partition size and number

- `shared_buffers` 25%+/- memory
- Hot partitions should fit in shared buffers
- Too many partitions can increase planning time
  - >~1,000 partitions may require constraint/range changes
- For large partitions, ensure relevant indexes are available after partition exclusion

Always filter by the partition key

- Partition key should always be a predicate
- Query specific ranges if possible
- For dates, avoid interval math if possible
  - `payment_date > '2024-03-19'` vs.
  - `payment_date > now() - '1 month'::interval`

Use Schemas and Tablespace

- Consider creating partitions in a separate schema
- Separate tablespaces = data tiering
  - Hot data in fastest memory-based storage
  - Warm data in cheaper storage
  - Cold data to disk or object storage

🎉 THANK YOU! 🎉

redgate