PostGresConf : Dev Track
South Africa
5 Oct 2021

Karel van der Walt

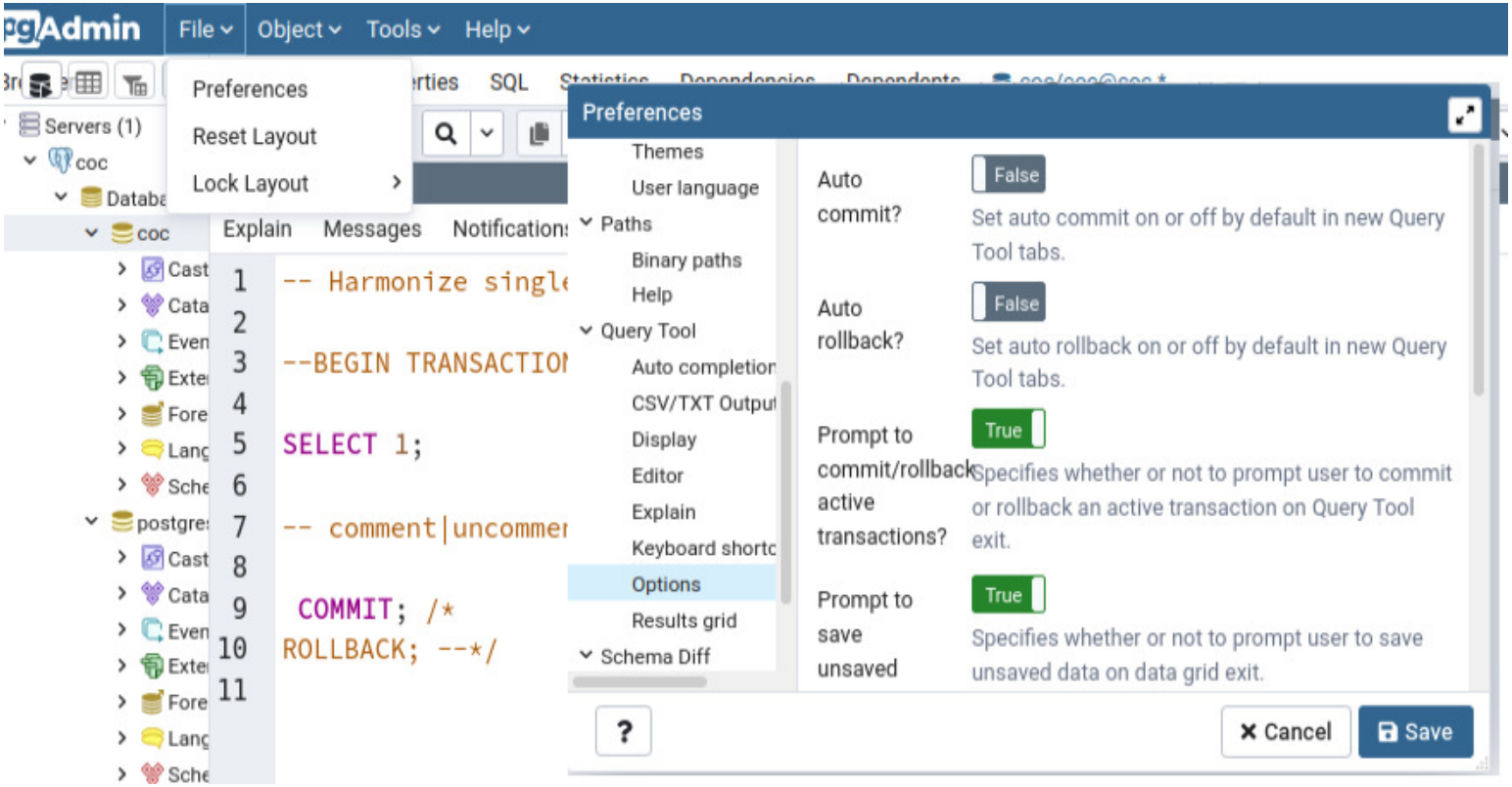
Modularizing computation via Common Table Expressions (CTEs)

Functional-declarative vs. Imperative styles of computation

Agenda

- Preliminaries
- CTE vs ‘recursive’ CTE
- The CTE template
- Realistic Example
- Fun Example
- Q&A (5 min)

Preliminaries – comment Nesting



Preliminaries – comment Nesting

```
-- Harmonize single-line and multi-line Comments

BEGIN TRANSACTION; -- say 'TRANSACTION'

SELECT 1;

-- comment|uncomment to alternate between COMMIT|ROLLBACK
-- COMMIT; /*
ROLLBACK; --*/
```

Derived Table vs Common Table Expression

```
SELECT *
FROM -- derived table
    ( VALUES
        -- add lines here
       (1::int, '2014-01-01'::date)
    ) _startDate( "Nr", "Date" )
;

;WITH
-- Common Table Expression
_startDate( "Nr", "Date" ) AS ( VALUES
      -- add lines here
     (1::int, '2014-01-01'::date)
)
SELECT *
FROM _startDate
```

Make more data : Columns vs Rows

```
;WITH
_principal(principal) AS ( VALUES
          (1000::decimal(10,2))
          ...
) -- SELECT * FROM _principal /*
,_rate(rate) AS ( VALUES
          ( 0.05::float)
          ...
) --  SELECT * FROM _rate /*
,_term(term) AS ( VALUES
          (4::int)
          ...
) --  SELECT * FROM _term /*
,_product(option, principal, rate, term) AS (
    SELECT
          ROW_NUMBER() OVER (  --PARTITION BY ...
                               ORDER BY p.principal, r.rate, t.term
                             ) rnum
          ,p.principal, r.rate, t.term
    FROM _principal p, _rate r, _term t
) SELECT * FROM _product

--*/ --*/ --*/
```

Preliminaries – Style Guide|Conventions

- Naming : Table|View vs CTE|Derived table

  LATERAL JOINs

- Position : “,” that separates CTEs

- Multi|single-line comments

Recursive CTE cf. other Products

- WITH vs WITH RECURSIVE
- UNION vs UNION ALL
- … RETURNING …
- ::Typed columns in Induction Base

CTE – the template

```
;WITH RECURSIVE
_startDate( "Nr", "Date" ) AS ( VALUES
    -- ad lines here
    (1::int, '2014-01-01'::date)
) -- SELECT * FROM _startDate /*
,_subsequentDate( "Nr", "Date" ) AS (
    SELECT
        "Nr", "Date"
    FROM _startDate
--/*
UNION ALL
    SELECT
        ("Nr"+1)
        , "Date" + 1::integer
    --FROM _startDate /* -- one iteration
    FROM _subsequentDate
    WHERE
        "Nr" < 365 --*/
--*/
) -- SELECT * FROM _subsequentDate /*
SELECT *
FROM _subsequentDate

--*/ --*/ -- one escape per CTE
```

CTE – 1 of 5

```
;WITH
_startDate( "Nr", "Date" ) AS ( VALUES
    -- add lines here
    (1::int, '2014-01-01'::date)
)
SELECT * FROM _startDate
```

CTE – 2 of 5

```
;WITH
_startDate( "Nr", "Date" ) AS ( VALUES
    -- add lines here
    (1::int, '2014-01-01'::date)
) -- SELECT * FROM _startDate /*
,_subsequentDate( "Nr", "Date" ) AS (
    SELECT
        "Nr", "Date"
    FROM _startDate
)
SELECT * FROM _subsequentDate
```

CTE – 3 of 5

```
;WITH
_startDate( "Nr", "Date" ) AS ( VALUES
    -- add lines here
    (1::int, '2014-01-01'::date)
) -- SELECT * FROM _startDate /*
,_subsequentDate( "Nr", "Date" ) AS (
    SELECT
        "Nr", "Date"
    FROM _startDate
/*
UNION ALL
    SELECT
        ("Nr"+1)
       , "Date" + 1::integer
    FROM _startDate
--*/
)
SELECT * FROM _subsequentDate

--*/ -- one escape per CTE
```

CTE – 4 of 5

```
;WITH --RECURSIVE
_startDate( "Nr", "Date" ) AS ( VALUES
    -- add lines here
    (1::int, '2014-01-01'::date)
) -- SELECT * FROM _startDate /*
,_subsequentDate( "Nr", "Date" ) AS (
    SELECT
        "Nr", "Date"
    FROM _startDate
--/*
UNION ALL
    SELECT
        ("Nr"+1)
       , "Date" + 1::integer
    FROM _startDate /* -- one iteration
    FROM _subsequentDate
    WHERE
        "Nr" < 365 --*/
--*/
)
SELECT * FROM _subsequentDate

--*/ -- one escape per CTE
```

CTE – 5 of 5

```
;WITH RECURSIVE
_startDate( "Nr", "Date" ) AS ( VALUES
    -- add lines here
    (1::int, '2014-01-01'::date)
) -- SELECT * FROM _startDate /*
,_subsequentDate( "Nr", "Date" ) AS (
    SELECT
        "Nr", "Date"
    FROM _startDate
--/*
UNION ALL
    SELECT
        ("Nr"+1)
       , "Date" + 1::integer
    --FROM _startDate /* -- one iteration
    FROM _subsequentDate
    WHERE
        "Nr" < 365 --*/
--*/
) -- SELECT * FROM _subsequentDate /*
SELECT
    "Date"
FROM _subsequentDate
WHERE
    EXTRACT(DOW FROM "Date") IN (0, 6);

--*/ --*/ -- one escape per CTE
```

Amortization Schedule – 1 of 6

```
;WITH
_product(option, principal, rate, terms) AS ( VALUES
    -- add lines here
    (1::int, 1000::decimal(10,2), 0.05::float, 4::int)
   ,(2::int, 2000::decimal(10,2), 0.05::float, 8::int)
)  -- SELECT * FROM _product /*
,_first_installment(option, period, opening_balance, payment, closing_balance) AS (
    SELECT p.option , 1, p.principal::float
          ,__payment.amount
          ,(p.principal*(1+p.rate)       -- accrue interest
            - __payment.amount)::float   -- reduce principal & interest
    FROM _product p  LEFT JOIN LATERAL (  SELECT
           -- PMT = principal*rate*(1/(1-e**(-term*ln(1+rate)))) round up
          (p.principal*p.rate*(1/(1-EXP(-p.terms*LOG(1+p.rate)/LOG(EXP(1))))))::float
                                      ) AS __payment(amount) ON true
    --WHERE
    --p.option = 1
)
SELECT * FROM _first_installment

--*/ -- one escape per CTE
```

Amortization Schedule – 2 of 6

```
;WITH
_product(option, principal, rate, terms) AS ( VALUES
    -- add lines here
    (1::int, 1000::decimal(10,2), 0.05::float, 4::int)
)  -- SELECT * FROM _product /*
,_first_installment(option, period, opening_balance, payment, closing_balance) AS (
    SELECT p.option , 1, p.principal::float
          ,__payment.amount
          ,(p.principal*(1+p.rate)       -- accrue interest
            - __payment.amount)::float   -- reduce principal & interest
    FROM _product p  LEFT JOIN LATERAL (  SELECT
          -- PMT = principal*rate*(1/(1-e**(-term*ln(1+rate)))) round up
          (p.principal*p.rate*(1/(1-EXP(-p.terms*LOG(1+p.rate)/LOG(EXP(1)))))))::float
                                      ) AS __payment(amount) ON true
    --WHERE
    --p.option = 1
) -- SELECT * FROM _first_installment /*
,_amortization_schedule(option, period, opening_balance, payment, closing_balance) AS (
    SELECT
        option, period, opening_balance, payment, closing_balance
    FROM _first_installment
)
SELECT * FROM _amortization_schedule s ORDER BY s.option, s.period

--*/ -- one escape per CTE
```

Amortization Schedule – 3 of 6

```
;WITH
_product(option, principal, rate, terms) AS ( VALUES
    ...
)  -- SELECT * FROM _product /*
,_first_installment(option, period, opening_balance, payment, closing_balance) AS (
    SELECT p.option , …
    FROM _product p  ...
    ...
) -- SELECT * FROM _first_installment /*
,_amortization_schedule(option, period, opening_balance, payment, closing_balance) AS (
    SELECT
        option, period, opening_balance, payment, closing_balance
    FROM _first_installment
--/*
UNION ALL
    SELECT
         s.option, s.period+1
        ,s.closing_balance                          -- previous period closing is subsequent opening
        ,s.payment
        ,(s.closing_balance*(1+p.rate) - s.payment)  -- subsequent closing
    FROM _first_installment s INNER JOIN _product p ON s.option = p.option
--*/
)
SELECT * FROM _amortization_schedule s ORDER BY s.option, s.period

--*/ --*/ --*/ --*/
```

Amortization Schedule – 4 of 6

```
;WITH --RECURSIVE
_product(option, principal, rate, terms) AS ( VALUES
    ...
)  -- SELECT * FROM _product /*
,_first_installment(option, period, opening_balance, payment, closing_balance) AS (
    ...
) -- SELECT * FROM _first_installment /*
,_amortization_schedule(option, period, opening_balance, payment, closing_balance) AS (
    SELECT
        option, period, opening_balance, payment, closing_balance
    FROM _first_installment
--/*
UNION ALL
    SELECT
         s.option, s.period+1
        ,s.closing_balance                                  -- previous period closing is subsequent opening
        ,s.payment
        ,(s.closing_balance*(1+p.rate) - s.payment)  -- subsequent closing
    --FROM _first_installment s INNER JOIN _product p ON s.option = p.option /* -- one iteration
    FROM _amortization_schedule s INNER JOIN _product p ON s.option = p.option
    WHERE
        s.period < p.terms  --*/
--*/
)
SELECT * FROM _amortization_schedule s ORDER BY s.option, s.period

--*/ --*/ --*/ --*/
```

Amortization Schedule – 5 of 6

```
;WITH RECURSIVE
_product(option, principal, rate, terms) AS ( VALUES
    ...
)  -- SELECT * FROM _product /*
,_first_installment(option, period, opening_balance, payment, closing_balance) AS (
    ...
) -- SELECT * FROM _first_installment /*
,_amortization_schedule(option, period, opening_balance, payment, closing_balance) AS (
    ...
) -- SELECT * FROM _amortization_schedule s ORDER BY s.option, s.period /*
,_amortization_schedule_adjusted_last_period (option, period, opening_balance, payment, closing_balance) AS (
    SELECT
        s.option, period, opening_balance::decimal(10,2), payment::decimal(10,2), closing_balance::decimal(10,2)
    FROM _amortization_schedule s INNER JOIN _product p ON p.option = s.option
    WHERE
        s.period < p.terms
UNION
    SELECT
         s.option, period, opening_balance::decimal(10,2), payment::decimal(10,2)
       -- force
       ,0.00 AS closing_balance
    FROM _amortization_schedule s INNER JOIN _product p ON p.option = s.option
    WHERE
        s.period = p.terms
)
SELECT * FROM _amortization_schedule_adjusted_last_period
--*/ --*/ --*/ --*/
```

Amortization Schedule – 6 of 6

```
;WITH RECURSIVE
_product(option, principal, rate, terms) AS ( VALUES
    ...
)  -- SELECT * FROM _product /*
,_first_installment(option, period, opening_balance, payment, closing_balance) AS (
    ...
) -- SELECT * FROM _first_installment /*
,_amortization_schedule(option, period, opening_balance, payment, closing_balance) AS (
    ...
) -- SELECT * FROM _amortization_schedule s ORDER BY s.option, s.period /*
,_amortization_schedule_adjusted_last_period (option, period, opening_balance, payment, closing_balance) AS (
    ...
) -- SELECT * FROM _amortization_schedule_adjusted_last_period /*
,_amortization_schedule_split_payment( option, period, opening_balance, payment
                                      ,principal, interest, closing_balance ) AS (
    SELECT
        option, period, opening_balance, payment
       ,(opening_balance - closing_balance) AS principal
       -- sacrifice on interest
       ,(payment - (opening_balance - closing_balance)) AS interest
       ,closing_balance
    FROM _amortization_schedule_adjusted_last_period
)  SELECT * FROM _amortization_schedule_split_payment ORDER BY option /*


--*/ --*/ --*/ --*/
```

References

From models to hosted OpenAPI Specification (OAS).

Karel van der Walt, PostgresConf South Africa 2019

https://www.youtube.com/watch?v=VwbjkaOG2YE

Modern SQL: A lot has changed since SQL-92.

https://modern-sql.com

The Mother of all Query Languages: SQL in Modern Times.

Markus Winand

https://www.youtube.com/watch?v=swR33jlhW8Q