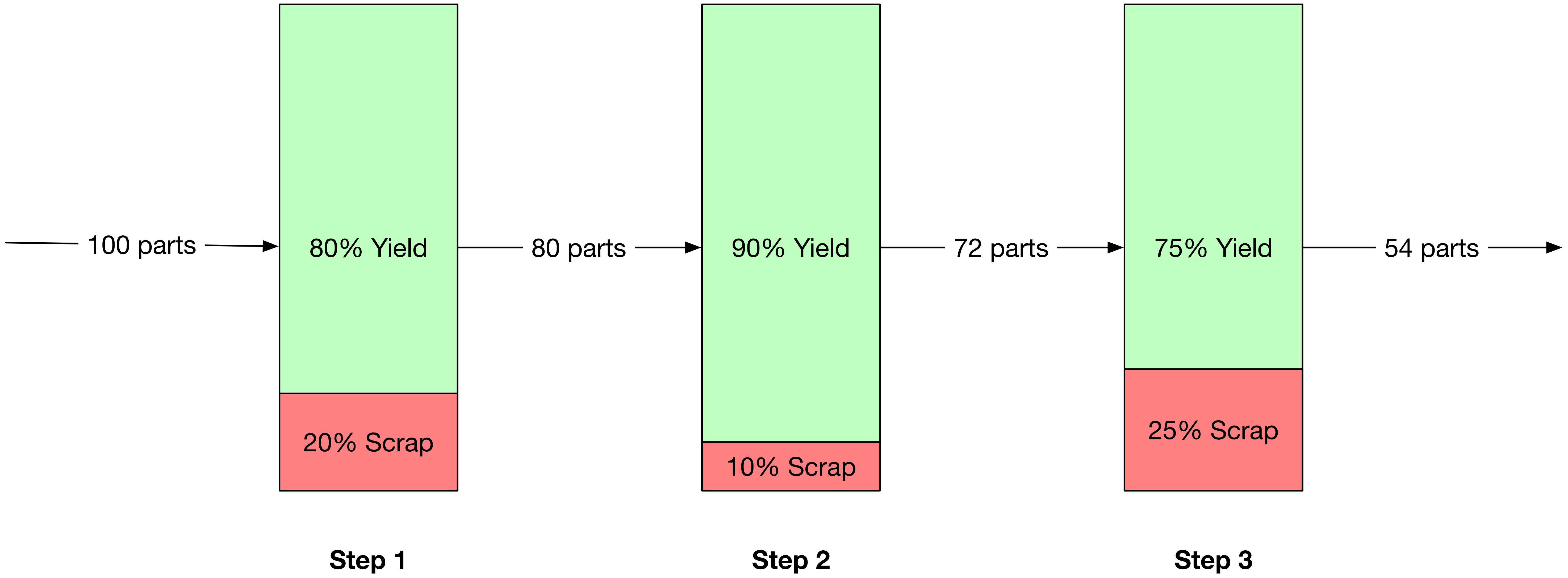
User-defined Aggregates

Felix Geisendörfer

2017-12-19 Berlin PostgreSQL Meetup

Manufacturing 🏭



Rolled Throughput Yield [1] 🔮

$$\begin{aligned}
rty &= \mathit{yield\ at\ step\ 1} * \mathit{yield\ at\ step\ 2} * \ldots * \mathit{yield\ at\ step\ n} \\
&= 80\% * 90\% * 75\% \\
&= 54\%
\end{aligned}$$

[1] https://en.wikipedia.org/wiki/Rolled_throughput_yield

RTY: Sample Data 📊

```sql
CREATE TABLE yields AS
SELECT *
FROM (VALUES
  ('day-1', 'step-1', 0.80),
  ('day-1', 'step-2', 0.90),
  ('day-1', 'step-3', 0.75),
  ('day-2', 'step-1', 0.90),
  ('day-2', 'step-2', 0.80),
  ('day-2', 'step-3', 0.99)
) vals(day, step, yield);
```

RTY: Use product aggregate 😢

```sql
SELECT day, product(yield)
FROM yields
GROUP BY 1;
```

```
ERROR:  function product(numeric) does not exist
LINE 1: SELECT day, product(yield)
```

Create a User-defined Aggregate 🧙🏼‍♂️

```
CREATE FUNCTION product_sf(state anyelement, val anyelement) RETURNS anyelement
LANGUAGE sql IMMUTABLE
AS $$
  SELECT state * val;
$$;


CREATE AGGREGATE product(anyelement) (
  initcond = 1,
  sfunc    = product_sf,
  stype    = anyelement
);
```

Use User-defined Aggregate ✨

```
SELECT day, product(yield)
FROM yields
GROUP BY 1;
```

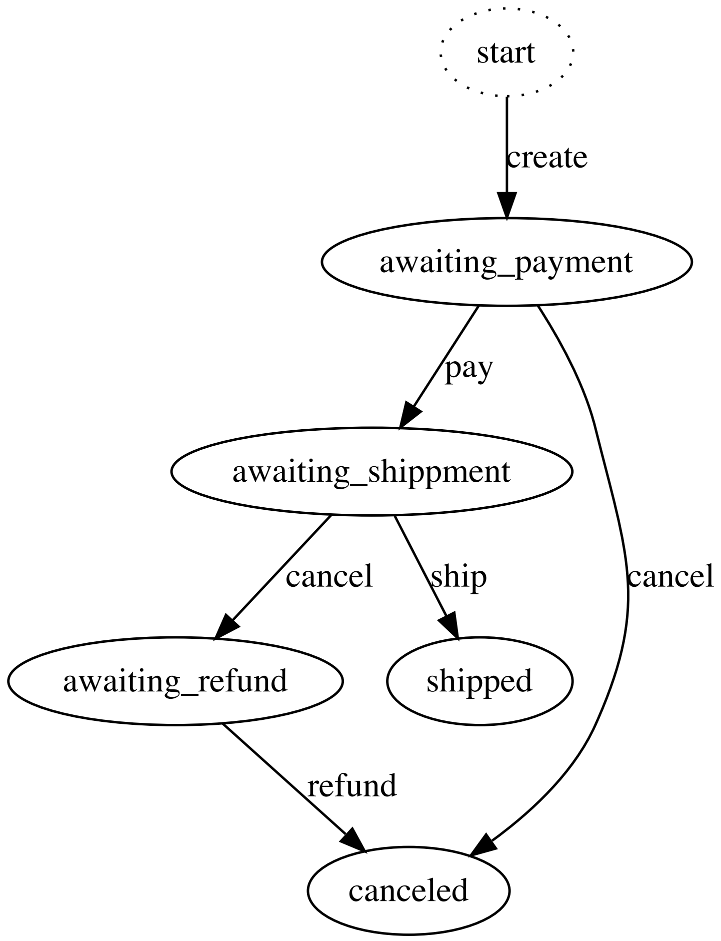
```
  day  | product
-------+----------
 day-2 | 0.712800
 day-1 | 0.540000
```

Even works as a Window Function 🤩

```
SELECT *, product(yield) OVER (PARTITION BY day ORDER BY step)
FROM yields;
```

```
  day  |  step  | yield | product
-------+--------+-------+----------
 day-1 | step-1 |  0.80 |     0.80
 day-1 | step-2 |  0.90 |   0.7200
 day-1 | step-3 |  0.75 | 0.540000
 day-2 | step-1 |  0.90 |     0.90
 day-2 | step-2 |  0.80 |   0.7200
 day-2 | step-3 |  0.99 | 0.712800
```

State Machines as UDAs[2] 🔥



[2] http://felixge.de/2017/07/27/implementing-state-machines-in-postgresql.html

Thanks

- Slides: github.com/felixge/talks
- Twitter: @felixge
- Jobs: felixge@apple.com (Cupertino and Shanghai)