

The Power of SQL Lambda Functions

Maximilian E. Schüle, Dimitri Vorona, Linnea Passing, Harald Lang, Alfons Kemper, Stephan Günemann, Thomas Neumann
 {schuele,vorona,passing,langh,kemper,guennemann,neumann}@in.tum.de

The screenshot shows the HyPerInsight interface. On the left, a SQL query is displayed:

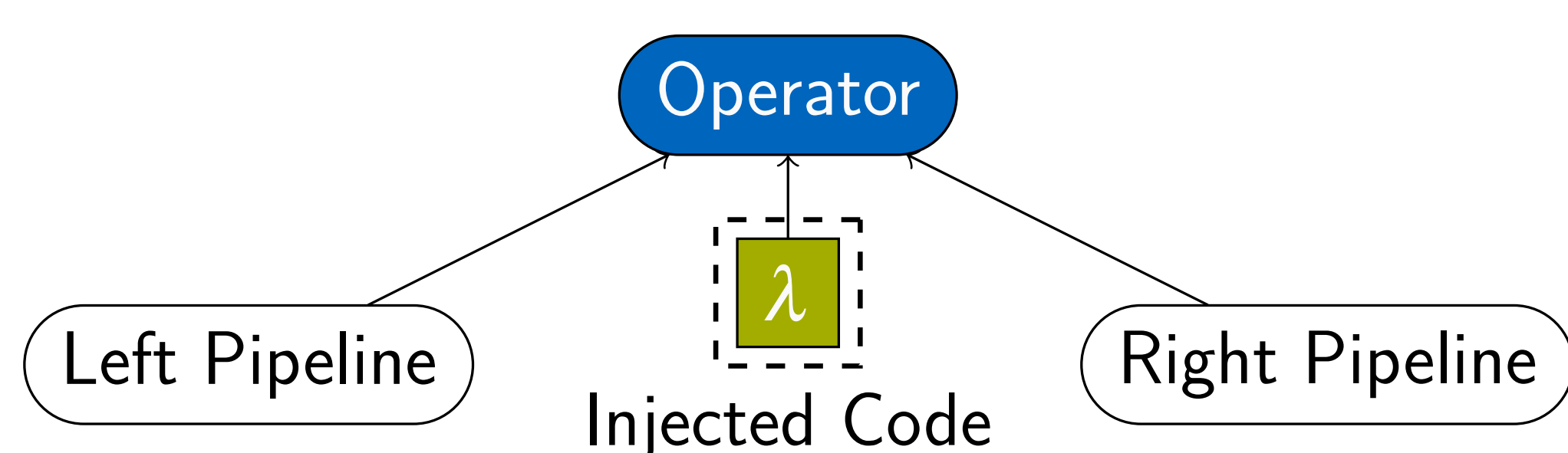

```

    1 SELECT lng,
    2     lat,
    3     cluster,
    4     airport_name
    5 FROM kmeans(
    6   (SELECT (longitude / 180 * pi()) AS lng,
    7         (latitude / 180 * pi()) AS lat,
    8         airport_name
    9   FROM airports), λ(a,b) (2 * atan2(sqrt(sin((b.lat-a.lat)/2) ^ 2 + cos(a.lat) *
    10  cos(b.lat) * (sin((b.lng - a.lng) / 2) ^ 2)), sqrt(1-sin((b.lat-a.lat)/2) ^ 2 +
    cos(a.lat) * cos(b.lat) * (sin((b.lng - a.lng) / 2) ^ 2))), 10 )
    ORDER BY lng, lat
    
```

 A histogram on the right shows execution times for various algorithms: Apriori, DBScan, KMeans, KModes, NaiveBayes, PageRank, Gradient Descent, and Labeling. Below the histogram is a table of results with columns: LNG, LAT, CLUSTER, and AIRPORT_NAME. A map shows the geographic distribution of airports. On the right, an 'OPTIMIZED PLAN' shows a query plan with steps: AIRPORTS, X, KMEANS, SORT, and RESULT.

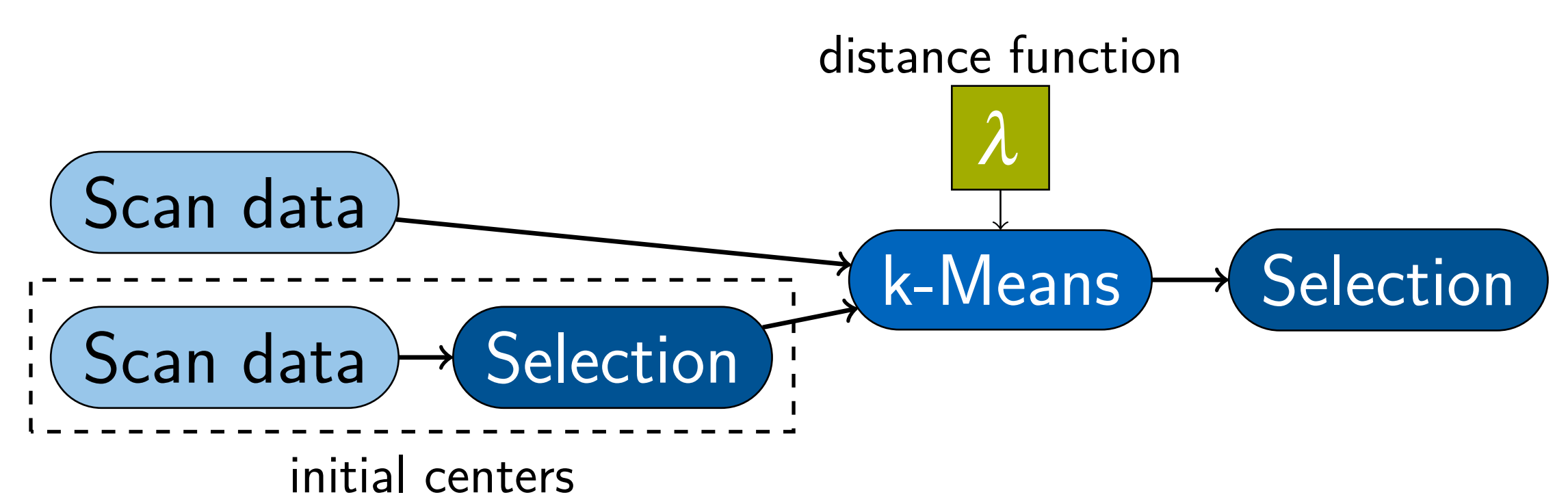
TU München 2018, Chair for Database Systems

Lambda Functions for SQL



- + Allow users to inject expressions in operators
- + Applicable on unary and binary operators as well

Example: k-Means



- + Distance function as lambda expression
- + User-specified distance functions are allowed

```

CREATE TABLE data(x FLOAT, y INT);
CREATE TABLE center(x FLOAT, y INT);
INSERT INTO ...

SELECT * FROM kmeans(
  (SELECT x,y FROM data),
  (SELECT x,y FROM center),
  -- the distance function
  λ(a,b) (a.x-b.x)^2+(a.y-b.y)^2,
  3 -- max. number of iterations
);
    
```

Listing 1: k-Means.

```

CREATE TABLE edges (a BIGINT,b BIGINT);
INSERT INTO ...

SELECT * FROM pagerank(
  (SELECT * FROM edges),
  λ(src) src.a, -- source
  λ(dst) dst.b, -- destination
  0.85, -- damping factor
  0.00001, -- threshold
  100 -- iterations
);
    
```

Listing 2: PageRank.

```

CREATE TABLE data (x FLOAT, y FLOAT);
CREATE TABLE weights(a FLOAT, b FLOAT);
INSERT INTO ...

SELECT * FROM gradientdescent(
  -- the loss function
  λ(d,w) (w.a*d.x+w.b-d.y)^2,
  -- training data set
  (SELECT x,y FROM data d),
  -- initial weights
  (SELECT a,b FROM weights w),
  0.05, -- learning rate
  100 -- max. number of iteration
);
    
```

Listing 3: Gradient descent.

```

CREATE TABLE testdata (x FLOAT);
CREATE TABLE weights(a FLOAT, b FLOAT);
INSERT INTO ...

SELECT * FROM labeling(
  -- the model function
  λ(d,w) (w.a*d.x+w.b),
  -- test data set
  (SELECT x,y FROM testdata d),
  -- optimized weights
  (SELECT a,b FROM weights w)
);
    
```

Listing 4: Labeling.