Views

```sql
SELECT l.partkey
FROM lineitem l, orders o
WHERE l.orderkey = o.orderkey
  AND o.orderdate > DATE('2015-03-31')
ORDER BY l.shipdate DESC
LIMIT 10;

SELECT l.partkey, COUNT(*)
FROM lineitem l, orders o
WHERE l.orderkey = o.orderkey
  AND o.orderdate > DATE('2015-03-31')
GROUP BY l.partkey;

SELECT l.suppkey, COUNT(*)
FROM lineitem l, orders o
WHERE l.orderkey = o.orderkey
  AND o.orderdate > DATE('2015-03-31')
GROUP BY l.suppkey;
```

MATERIALIZED

```
CREATE VIEW salesSinceLastMonth AS
  SELECT l.*
  FROM lineitem l, orders o
  WHERE l.orderkey = o.orderkey
    AND o.orderdate > DATE('2015-03-31')
```

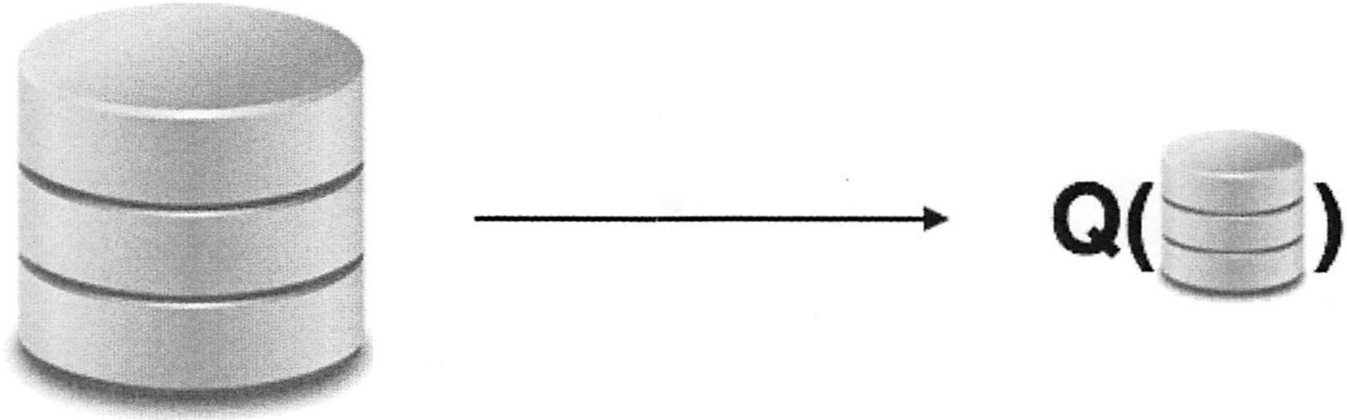
```
SELECT partkey FROM salesSinceLastMonth
ORDER BY shipdate DESC LIMIT 10;
```

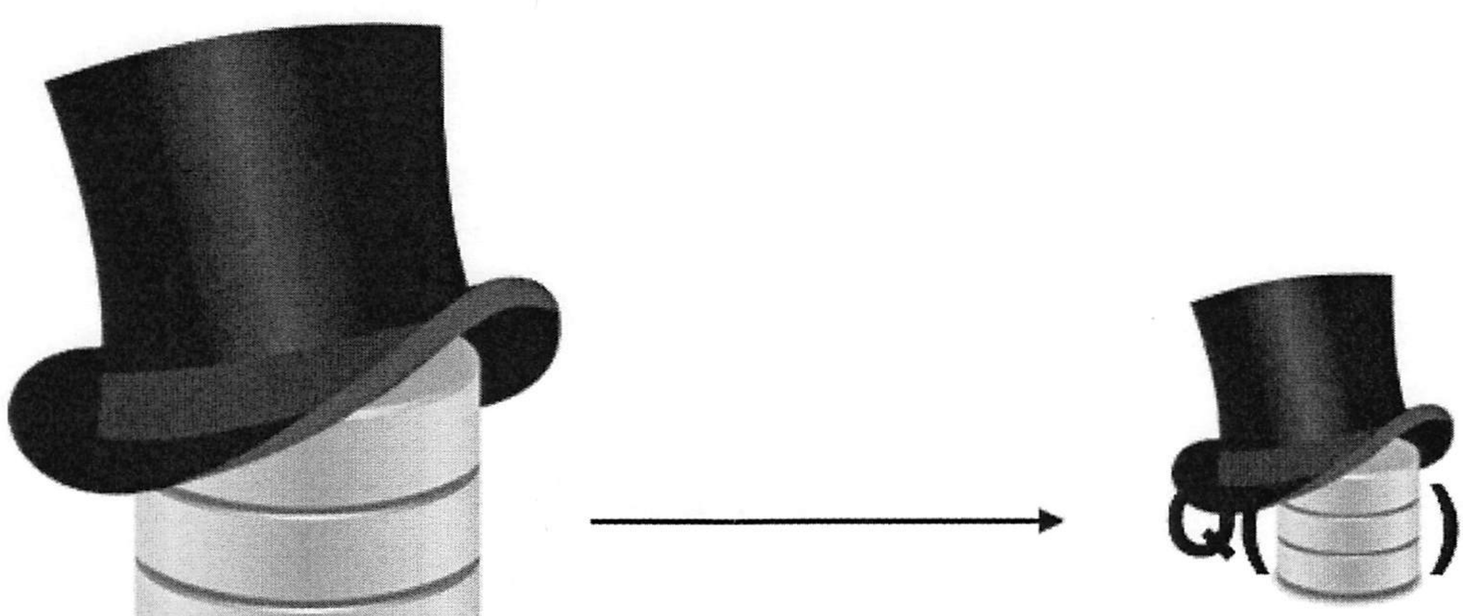
```
SELECT suppkey, COUNT(*)
FROM salesSinceLastMonth
GROUP BY suppkey;
```

```
SELECT partkey, COUNT(*)
FROM salesSinceLastMonth
GROUP BY partkey;
```

```
SELECT partkey FROM ordersSinceLastMonth
ORDER BY shipdate DESC LIMIT 10;
```

```
SELECT partkey FROM
  (
    SELECT l.*
    FROM lineitem l, orders o
    WHERE l.orderkey = o.orderkey
      AND o.orderdate > DATE('2015-03-31')
  ) AS salesSinceLastMonth
ORDER BY shipdate DESC LIMIT 10;
```


Q( )


Q( )

D : Database

ΔD : Change

Insert
Delete
Update

Q : Query

$Q(D)$: Resu~~t~~lt

We have: $Q(D)$, ΔD

We want: $$Q(D + \Delta D) \quad \overbrace{Q(D)}^{\text{we have}} \underset{\text{fast}}{+} \overbrace{\Delta Q(D, \Delta D)}^{\text{fast}}$$

$\Delta D : R \to R \uplus \Delta R$

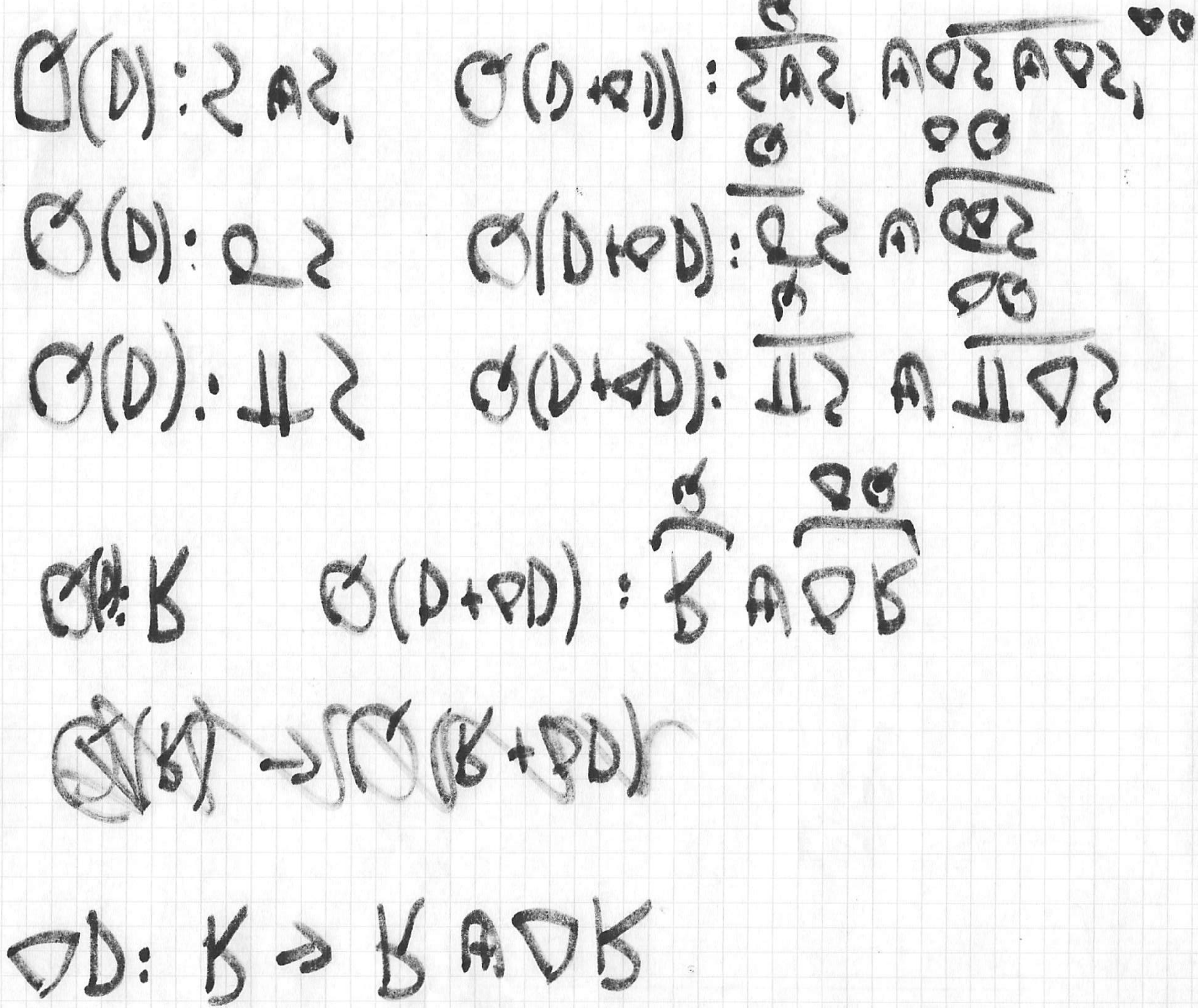
$\cancel{Q(R) \to Q(R + \Delta R)}$

$Q(D) : R \qquad Q(D+\Delta D) : \underbrace{R}_{Q} \uplus \underbrace{\Delta R}_{\Delta Q}$

$Q(D) : \pi S \qquad Q(D+\Delta D) : \underbrace{\pi S}_{Q} \uplus \underbrace{\pi \Delta S}_{\Delta Q}$

$Q(D) : \sigma S \qquad Q(D+\Delta D) : \underbrace{\sigma S}_{Q} \uplus \underbrace{\sigma \Delta S}_{\Delta Q}$

$Q(D) : S \uplus S' \qquad Q(D+\Delta D) : \underbrace{S \uplus S'}_{Q} \uplus \underbrace{\Delta S \uplus \Delta S'}_{\Delta Q}$



$QD = \cancel{Q}(S \times S')$ $Q(D + \Delta D)$

$$\underbrace{(S \times S')}_{Q} \uplus \underbrace{(S \times \Delta S') \uplus (\Delta S \times S') \uplus (\Delta S \times \Delta S')}_{\Delta Q}$$

Also works for

$\bowtie$

∇

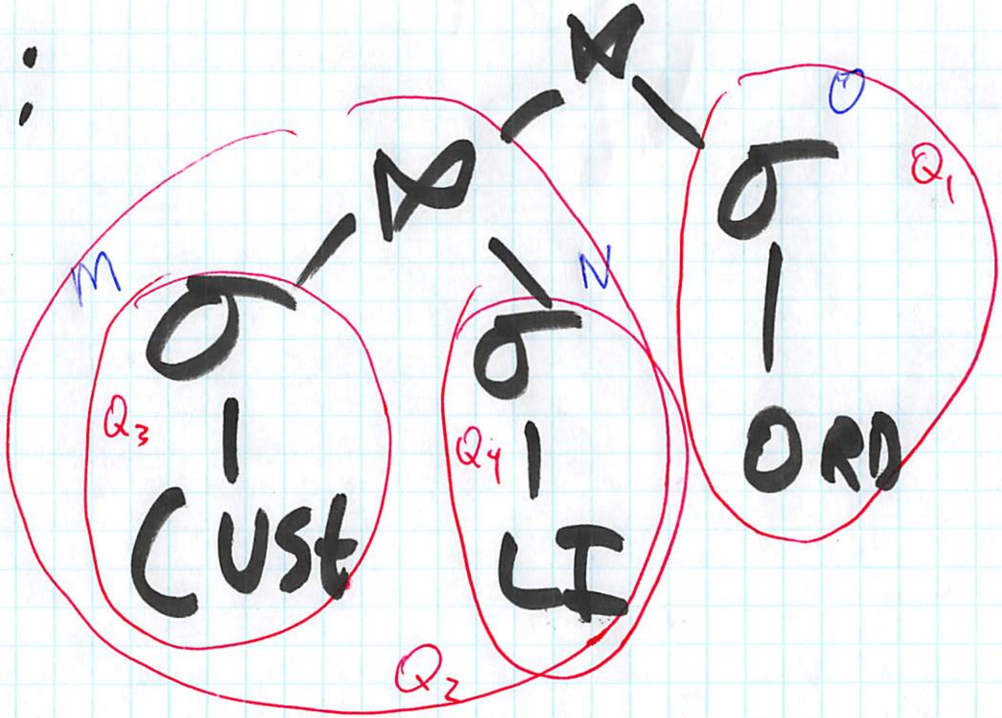
Also works for

α $\quad$ $D\alpha$

$$\overbrace{(Z\times Z_1)}^{\alpha} \oplus \overbrace{(Z\times \nabla Z_1) \oplus (DZ\times Z_1) \oplus (DZ\times \nabla Z_1)}^{D\alpha}$$

$$M(D) = M(Z\times Z_1) \qquad M(D\cdot \nabla D)$$

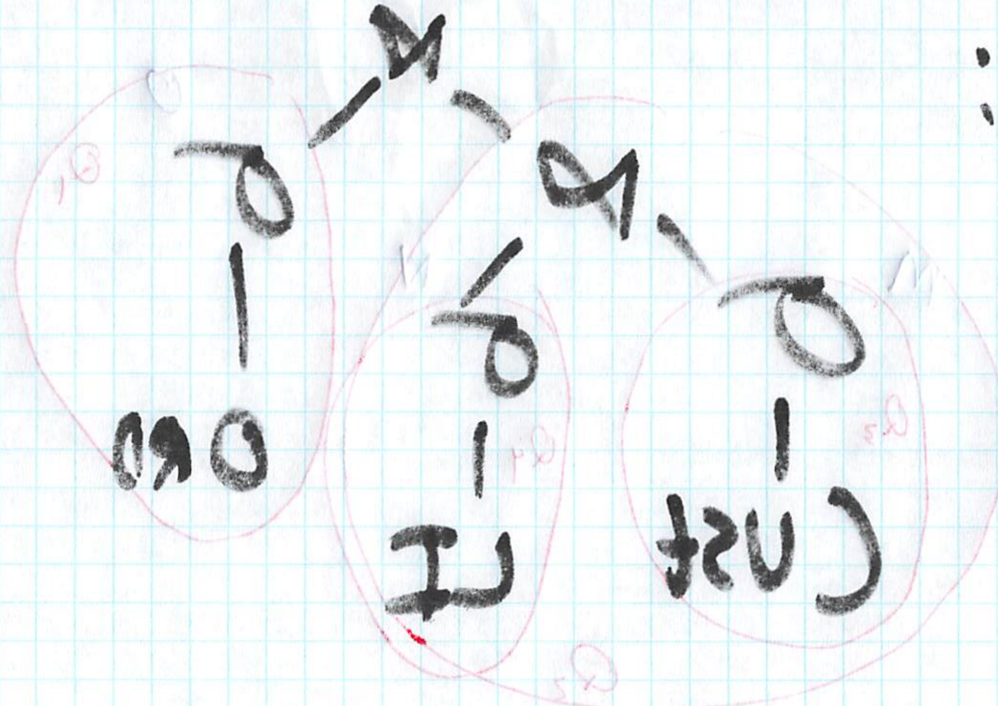
Q3:



ΔD = Insert into LI

$$\underset{LI}{\Delta} Q : \underset{O}{(\sigma\, Ord)} \bowtie \underset{M}{(\sigma\, Cust)} \bowtie \underset{C \text{ small const}}{(\sigma\, \Delta LI)}$$

Q5:



ΔD = Insert into LI

ΔQ_{LI} : $(\sigma\ Ord) \bowtie (\sigma\ Cust) \bowtie (\sigma\ \Delta LI)$

O M small const

$$Q_1 = \sigma\, Ord \qquad Q_2 = \sigma C \bowtie \sigma LI$$

$$\Delta Q = (Q_1 \bowtie \Delta Q_2) \uplus \underbrace{(\underbrace{\Delta Q_1}_{\emptyset} \bowtie Q_2)}_{\emptyset} \uplus \underbrace{(\underbrace{\Delta Q_1}_{\emptyset} \bowtie \Delta Q_2)}_{\emptyset}$$

(for an insert into LI)

but $\Delta Q_1 = \emptyset$

and $\emptyset \bowtie R = \emptyset$

and $\emptyset \uplus R = R$

$$= (Q_1 \bowtie \Delta Q_2)$$

$$Q_3 = \sigma C \qquad Q_4 = \sigma LI$$

$$\Delta Q_2 = (Q_3 \bowtie \Delta Q_4) = (Q_3 \bowtie \sigma \Delta LI)$$

so

$$\Delta Q = (\sigma\, Ord) \bowtie (\sigma\, Cust) \bowtie (\sigma LI)$$

$$\nabla\phi = (\rho \circ \eta) \ltimes (\rho \subset^{n+}) \ltimes (\rho \Gamma I)$$

2ϕ

$$\nabla\phi^5 = (\phi^3 \ltimes \nabla\phi^4) = (\phi^3 \ltimes \rho \nabla \Gamma I)$$

$\phi^3 = \rho \subset$ $\qquad$ $\phi^4 = \rho \Gamma I$

$$= (\phi^1 \ltimes \nabla\phi^5)$$

and $\phi A B = B$

and $\phi \ltimes B = \phi$

but $\nabla\phi^1 = \phi$

(follow insert
enfo ΓI)

$$\nabla\phi = (\phi^1 \ltimes \nabla\phi^5)\, A\, (\phi^1 \ltimes \phi^5)\, A\, (\nabla\phi^1 \ltimes \phi^5)$$

$\phi^1 = \rho \circ \eta$ $\qquad$ $\phi^5 = \rho \subset \ltimes \rho \Gamma I$

```
CREATE MATERIALIZED VIEW salesSinceLastMonth AS
  SELECT l.*
  FROM lineitem l, orders o
  WHERE l.orderkey = o.orderkey
    AND o.orderdate > DATE('2015-03-31')
```

```
SELECT l.partkey
FROM lineitem l, orders o
WHERE l.orderkey = o.orderkey
  AND o.orderdate > DATE('2015-03-31')
ORDER BY l.shipdate DESC
LIMIT 10;
```

Update SalesSinceLastMonth
Set statuscode='q' WHERE orderkey=22

```
CREATE TRIGGER salesSinceLastMonthInsert
INSTEAD OF INSERT ON salesSinceLastMonth
REFERENCING NEW ROW AS newRow
FOR EACH ROW
  IF NOT EXISTS (
     SELECT * FROM ORDERS
     WHERE ORDERS.orderkey = newRow.orderKey)
  ) THEN
    INSERT INTO ORDERS(orderkey)
      VALUES (orderkey)
  END IF;
  INSERT INTO LINEITEM VALUES newRow;
END FOR;
```

Motivation — Why are Views Useful?

Give an example query:

Workloads often have repeating patterns:

- ```
  SELECT l.partkey
  FROM lineitem l, orders o
  WHERE l.orderkey = o.orderkey
    AND o.orderdate > DATE('2015-03-31')
  ORDER BY l.shipdate DESC
  LIMIT 10;
  ```
- ```
  SELECT l.partkey, COUNT(*)
  FROM lineitem l, orders o
  WHERE l.orderkey = o.orderkey
    AND o.orderdate > DATE('2015-03-31')
  GROUP BY l.partkey;
  ```
- ```
  SELECT l.suppkey, COUNT(*)
  FROM lineitem l, orders o
  WHERE l.orderkey = o.orderkey
    AND o.orderdate > DATE('2015-03-31')
  GROUP BY l.suppkey;
  ```

View Definition

- ```
  CREATE VIEW salesSinceLastMonth AS
   SELECT l.*
   FROM lineitem l, orders o
   WHERE l.orderkey = o.orderkey
     AND o.orderdate > DATE('2015-03-31')
  ```
- ```
  SELECT partkey FROM salesSinceLastMonth
  ORDER BY shipdate DESC LIMIT 10;
  ```
- ```
  SELECT suppkey, COUNT(*)
  FROM salesSinceLastMonth
  GROUP BY suppkey;
  ```
- ```
  SELECT partkey, COUNT(*)
  FROM salesSinceLastMonth
  GROUP BY partkey;
  ```

Definition — What is a View / How are they used?

Views act as normal relations

- ```
  SELECT partkey FROM salesSinceLastMonth
  ORDER BY shipdate DESC LIMIT 10;
  ```
- ```
  SELECT partkey FROM
   (
     SELECT l.*
     FROM lineitem l, orders o
     WHERE l.orderkey = o.orderkey
       AND o.orderdate > DATE('2015-03-31')
  ```
  ```

```
  ) AS salesSinceLastMonth
ORDER BY shipdate DESC LIMIT 10;
```

- Views contain and abstract concepts
  - Analogous to a function
  - Complex query patterns can be given an shorthand
  - Can freely change view logic “in the background” (Change ‘last month’)
- But not quite normal relations…

View Updates

- ```
  UPDATE salesSinceLastMonth
     SET statusCode = ‘q’;
   WHERE orderkey = 22;
  ```
  - Easy… rows in salesSinceLastMonth go 1-1 with LINEITEM.
  - Can find the row of line item that matches a given row of salesSinceLastMonth and update it.
- ```
  INSERT INTO salesSinceLastMonth
    (orderkey, partkey, suppkey, …)
  VALUES
    (22, 99, 42, …);
  ```
  - Harder…
  - What happens if order #22 doesn’t exist?
  - How does the insertion interact with sequences (e.g., Lineitem.lineno)
- ```
  CREATE TRIGGER salesSinceLastMonthInsert
  INSTEAD OF INSERT ON salesSinceLastMonth
  REFERENCING NEW ROW AS newRow
  FOR EACH ROW
    IF NOT EXISTS (
       SELECT * FROM ORDERS
       WHERE ORDERS.orderkey = newRow.orderKey)
    ) THEN
      INSERT INTO ORDERS(orderkey)
        VALUES (orderkey)
    END IF;
    INSERT INTO LINEITEM VALUES newRow;
  END FOR;
  ```
- InsteadOf triggers update rows

View Materialization

- Views exist because they’re queried frequently…
- Why not use them to make computations faster.
  - Precompute (materialize) the view’s contents (like an index)
- Challenges:
  ```

- What happens when the data behind the view changes?
- What happens when the view definition changes?
- What happens when we write a query without realizing we have a view?

Updates to Materialized Views

Let's say you have a database D and a query Q

- Q(D) is the result of your query on the database

Let's say you make a change ΔD (e.g., Insert Tuple)

- Q(D+ΔD) is the new result

If we have Q(D), can we get Q(D+ΔD) faster?

- Analogy to Sum {34,29,10,15} + {12} (== 88+12)

Specific query examples

- Projection
- Selection
- Union
- Cross-Product
- Aggregation

Interactions with...

- Insert
- Delete
- Update

View Selection

Can we use materialized views without knowing about them?

- ```
  CREATE MATERIALIZED VIEW salesSinceLastMonth AS
    SELECT l.*
    FROM lineitem l, orders o
    WHERE l.orderkey = o.orderkey
      AND o.orderdate > DATE('2015-03-31')
  ```
- ```
  SELECT l.partkey
  FROM lineitem l, orders o
  WHERE l.orderkey = o.orderkey
    AND o.orderdate > DATE('2015-03-31')
  ORDER BY l.shipdate DESC
  LIMIT 10;
  ```

Simplify the query model:

- View: SELECT Lv FROM Rv WHERE Cv
- Query: SELECT Lq FROM Rq WHERE Cq

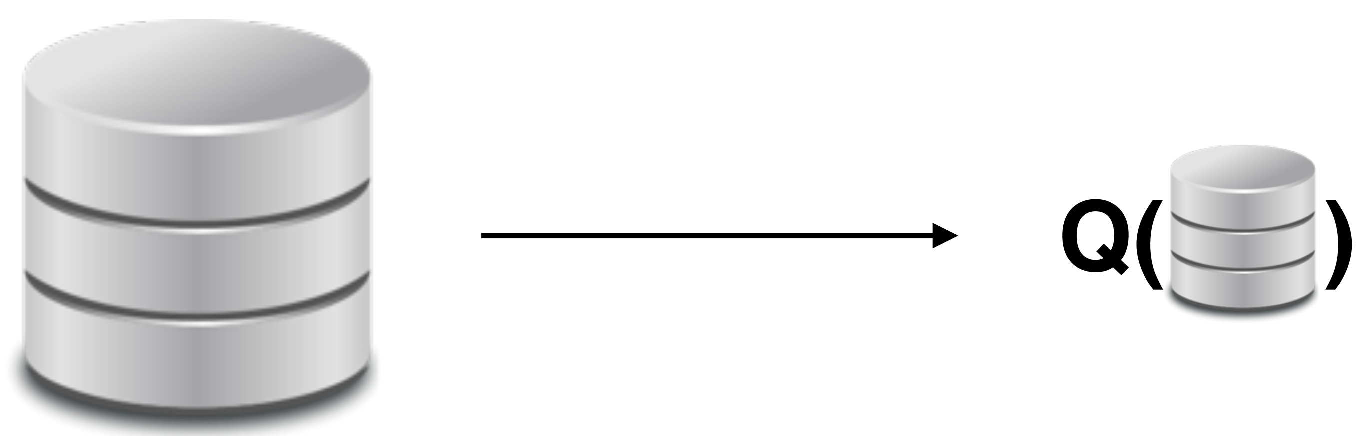
- When can we rewrite this query?
    - Rv ⊆ Rq (All relations in the view are in the query join)
    - Cq = Cv ∧ C’ (The view condition is weaker than the query condition)
    - Lq ∩ attrs(Rv) ⊆ Lv (The view doesn’t project away attributes needed for the output)
    - attrs(C’) ∩ attrs(Rv) ⊆ Lv (The view doesn’t project away attributes needed for the condition)
- The whole thing rewrites to:
    - SELECT Lq FROM (Rq-Rv), view WHERE C’

- Views for Transactions

Incremental View Maintenance

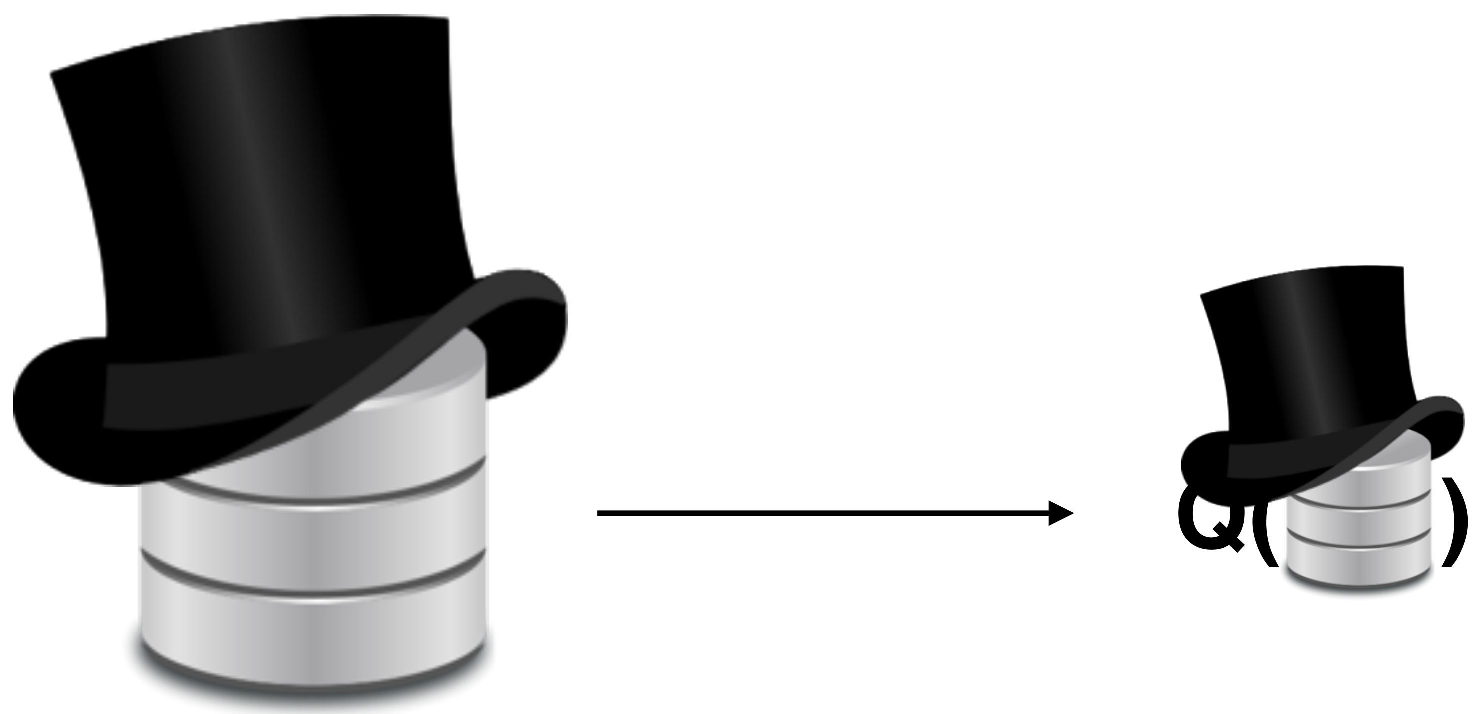
Not covered by Database Systems: TCB

Materialized Views



When the base data changes, the view needs to be updated

Materialized Views



When the base data changes, the view needs to be updated

View Maintenance

```
VIEW ← Q(D)
```

View Maintenance

```
WHEN D ← D+ΔD DO:
  VIEW ← Q(D+ΔD)
```

Re-evaluating the query from scratch is expensive!

View Maintenance

(ideally) Smaller & Faster Query

```
WHEN D ← D+ΔD DO:
  VIEW ← VIEW+ΔQ(D,ΔD)
```

(ideally) Fast “merge” operation.

Intuition

$D = \{1, 2, 3, 4\}$ $\Delta D = \{5\}$

$Q(D) = SUM(D)$

$Q(D+\Delta D) \sim O(|D|+|\Delta D|)$

$VIEW + SUM(\Delta D) \sim O(|\Delta D|)$

Intuition

$R = \{1, 2, 3\}, S = \{5,6\} \quad \Delta R = \{4\}$

$Q(R,S) = COUNT(R \mathbf{x} S)$

$Q(R+\Delta R,S) \sim O((|R|+|\Delta R|) * |S|)$

$VIEW + COUNT(|\Delta R|*|S|) \sim O(|\Delta R|*|S|)$

Intuition

+ ~ U

*** ~ x**

Are these kinds of patterns common?

Rings/Semirings

This kind of pattern occurs frequently.

Semiring : $< S, +, x, S_0, S_1 >$

Any set of 'things' **S** such that…

Closed

$$S_i + S_j = S_k$$

$$S_i \times S_j = S_k$$

$$S_i + S_0 = S_i$$

$$S_i \times S_1 = S_i$$

$$S_i \times S_0 = S_0$$

Additive & Multiplicative "zeroes"

$$S_i \times (S_j + S_k) = (S_i \times S_j) + (S_j \times S_k)$$

Distributive

Rings/Semirings

Ring : $< S, +, x, S_0, S_1, - >$

Any semiring where every element has an additive inverse…

$$S_i + (-S_i) = S_0$$

THE TANGENT ENDS NOW

Incremental View Maintenance

```
WHEN D ← D+ΔD DO:
  VIEW ← VIEW+ΔQ(D,ΔD)
```

Basic Challenges of IVM

What does ΔR represent?

How to interpret R $\pm$ ΔR?

How to compute ΔQ?

What is ΔR?

What does it need to represent?

$\boxed{\begin{array}{c}\text{Insertions}\\ \\ \text{Deletions}\end{array}}$

Updates

(Delete Old Record & Insert Updated Record)

What is ΔR?

A Set/Bag of Insertions

A Set/Bag of Deletions

What is +?

$$\mathbf{R} \quad + \quad \mathbf{\Delta R}$$

A Set/Bag **+** A Set/Bag of Insertions

A Set/Bag of Deletions

$$\mathtt{R} \;\cup\; \Delta\mathtt{R}_{\mathtt{inserted}}$$
$$-\; \Delta\mathtt{R}_{\mathtt{deleted}}$$

But this breaks closure of '+'!

Incremental View Maintenance

$\texttt{VIEW} \leftarrow \texttt{VIEW} + \boxed{\Delta\texttt{Q}(\texttt{D},\Delta\texttt{D})}$

Given Q(R,S,…)

Construct ΔQ(R,ΔR,S,ΔS,…)

Delta Queries

$$\Delta(\sigma(R))$$

σ

|

R

R

σ

|

ΔR

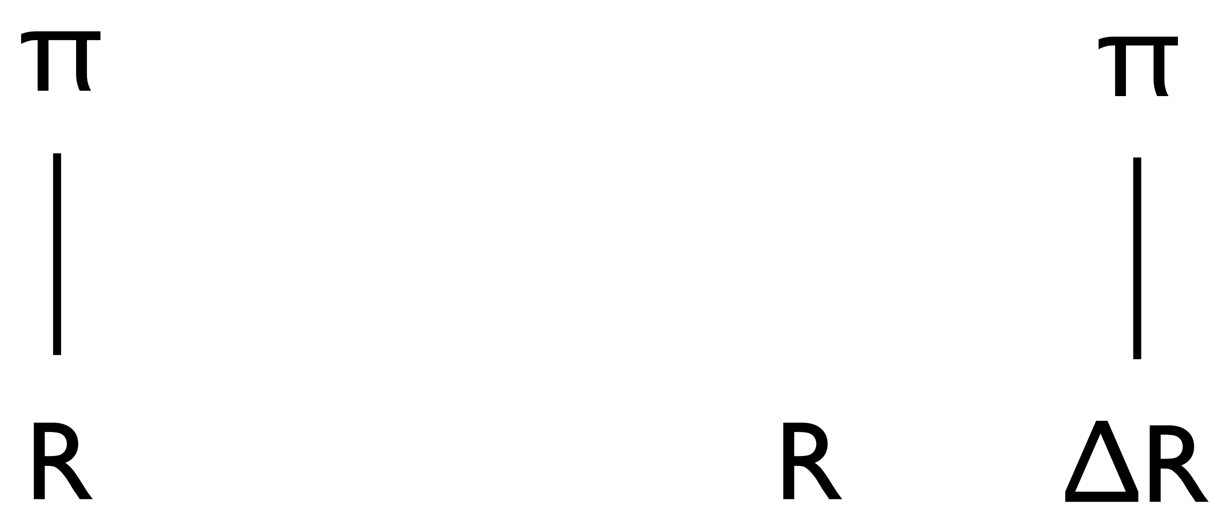
Original R

Inserted Tuples of R

Does this work for deleted tuples?

Delta Queries

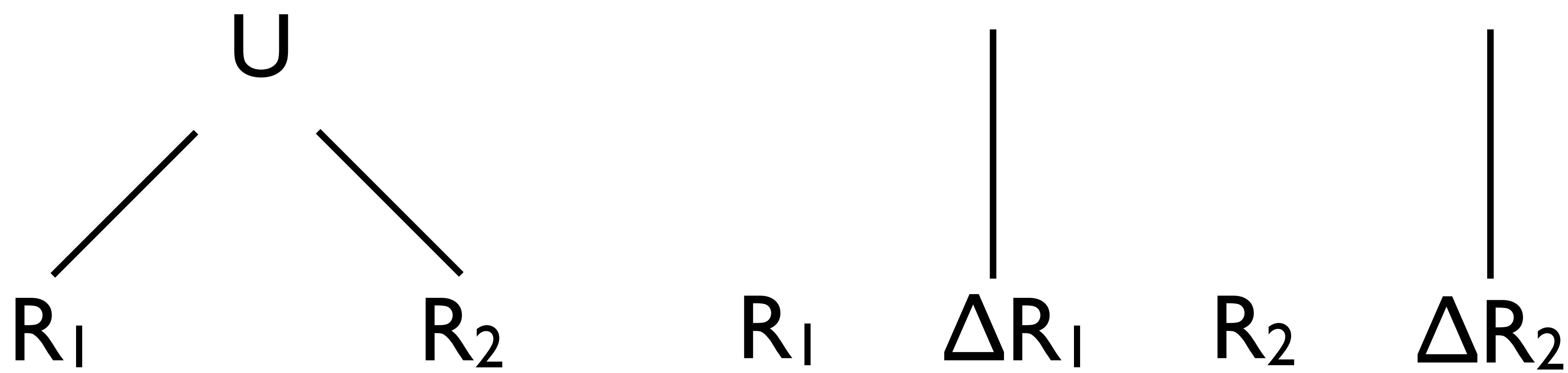
$$\Delta(\pi(R)) = \pi(\Delta R)$$



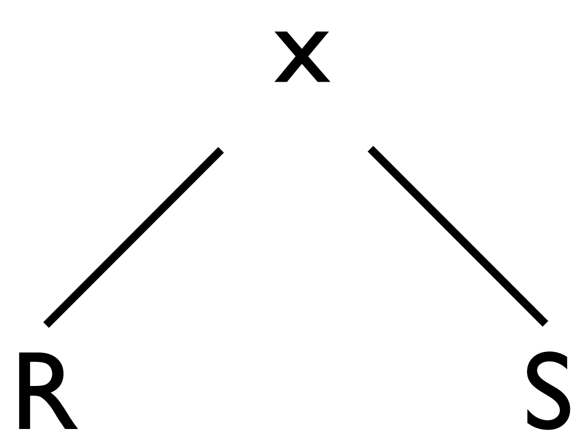
Does this work (completely) under set semantics?

Delta Queries

$$\Delta(R_1 \cup R_2)$$



Delta Queries



R ΔR S

Delta Queries

$R : \{ 1, 2, 3 \} \qquad S : \{ 5, 6 \}$

$R \times S = \{ \langle 1,5 \rangle, \langle 1, 6 \rangle, \langle 2,5 \rangle, \langle 2,6 \rangle, \langle 3,5 \rangle, \langle 3,6 \rangle \}$

$\Delta R_{inserted} = \{ 4 \}$

$\Delta R_{deleted} = \{ 3,2 \}$

$(R+\Delta R) \times S = \{ \langle 1,5 \rangle, \langle 1, 6 \rangle, \mathbf{\langle 4,5 \rangle, \langle 4,6 \rangle} \}$

$\Delta_{inserted}(R \times S) = \Delta R_{inserted} \times S$

$\Delta_{deleted}(R \times S) = \Delta R_{deleted} \times S$

What if R and S <u>both</u> change?

Delta Queries

Computing a Delta Query

$$\Delta(\sigma(R)) = \sigma(\Delta R)$$

$$\Delta(\pi(R)) = \pi(\Delta R)$$

$$\Delta(R_1 \cup R_2) = \Delta R_1 \cup \Delta R_2$$

$$\Delta(R_1 \times R_2) = ??$$

Delta Queries

$$(R_1 \cup \Delta R_1) \times (R_2 \cup \Delta R_2)$$

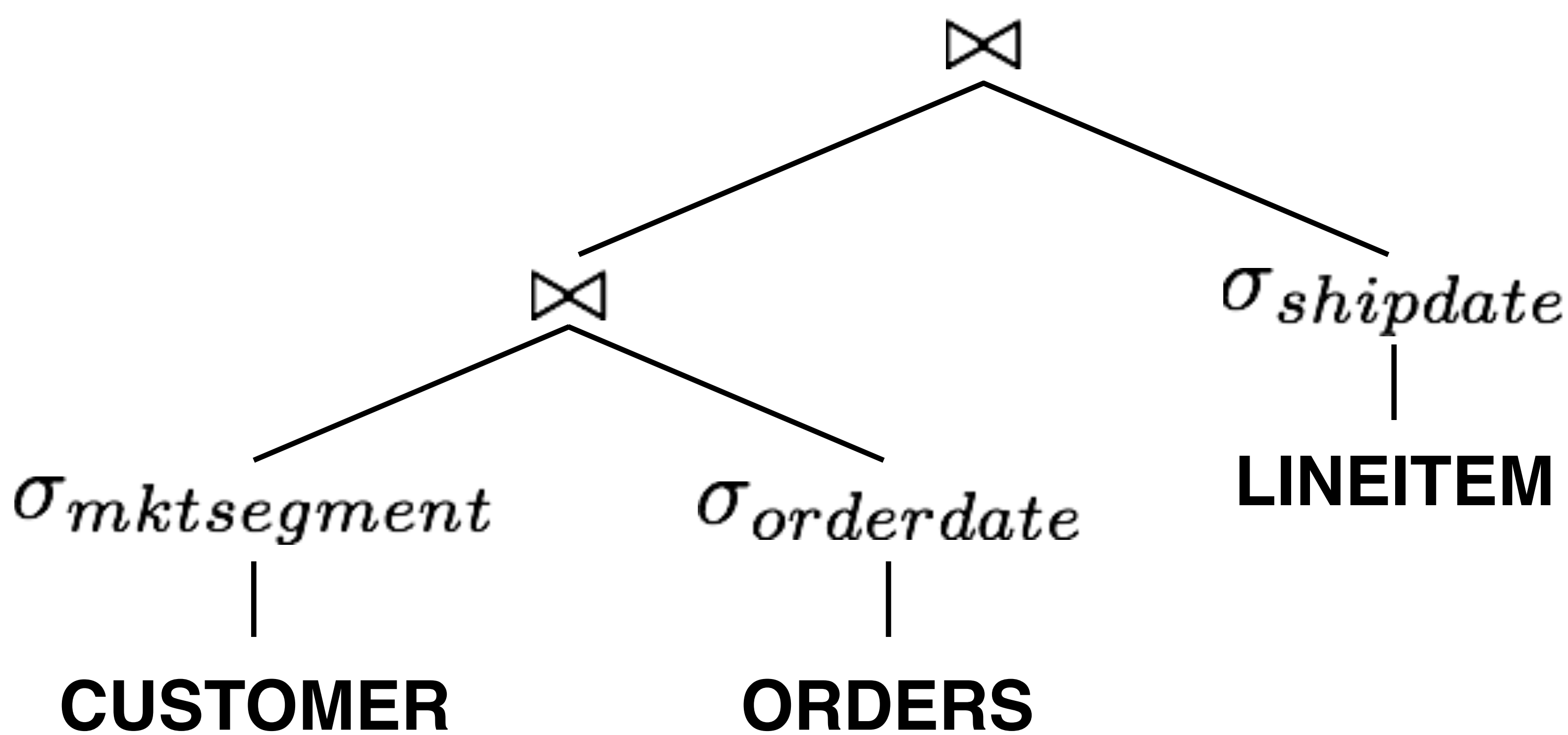
$$\boxed{(R_1 \times R_2)} \cup \boxed{(R_1 \times \Delta R_2) \cup (\Delta R_1 \times R_2) \cup (\Delta R_1 \times \Delta R_2)}$$

The original query

The delta query

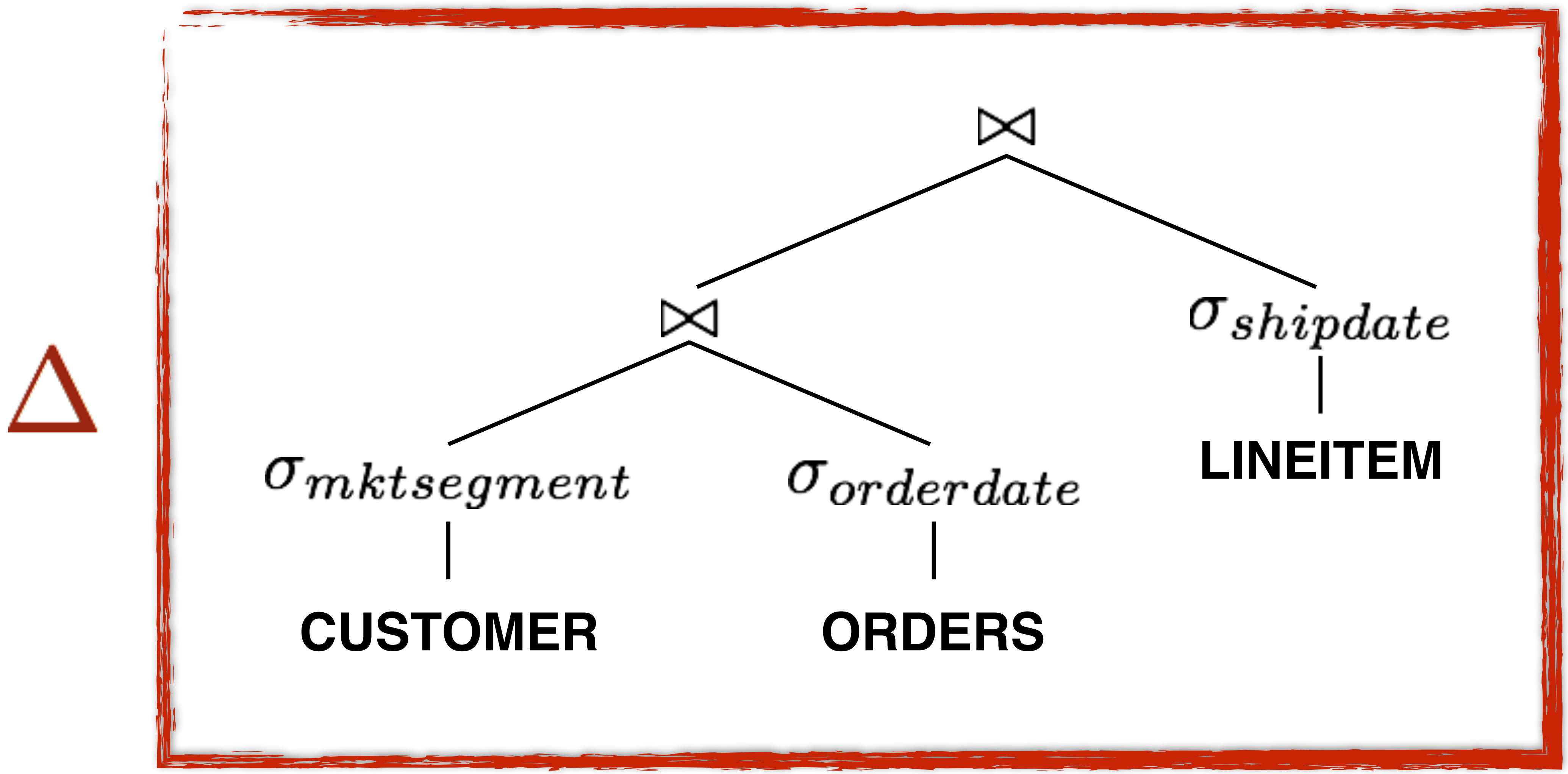
How about an example…

Delta Queries



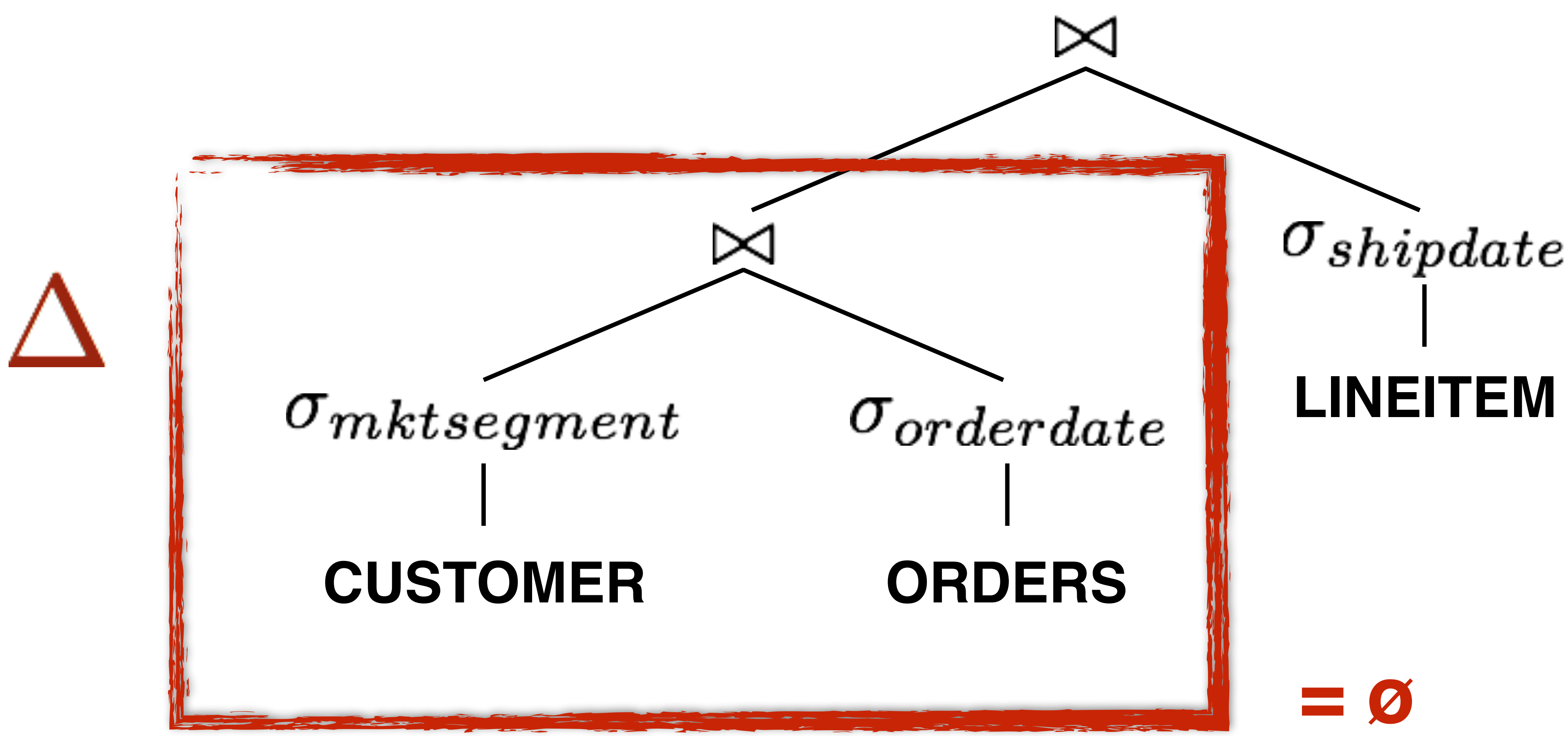
Let's say you have an insertion into LINEITEM

Delta Queries



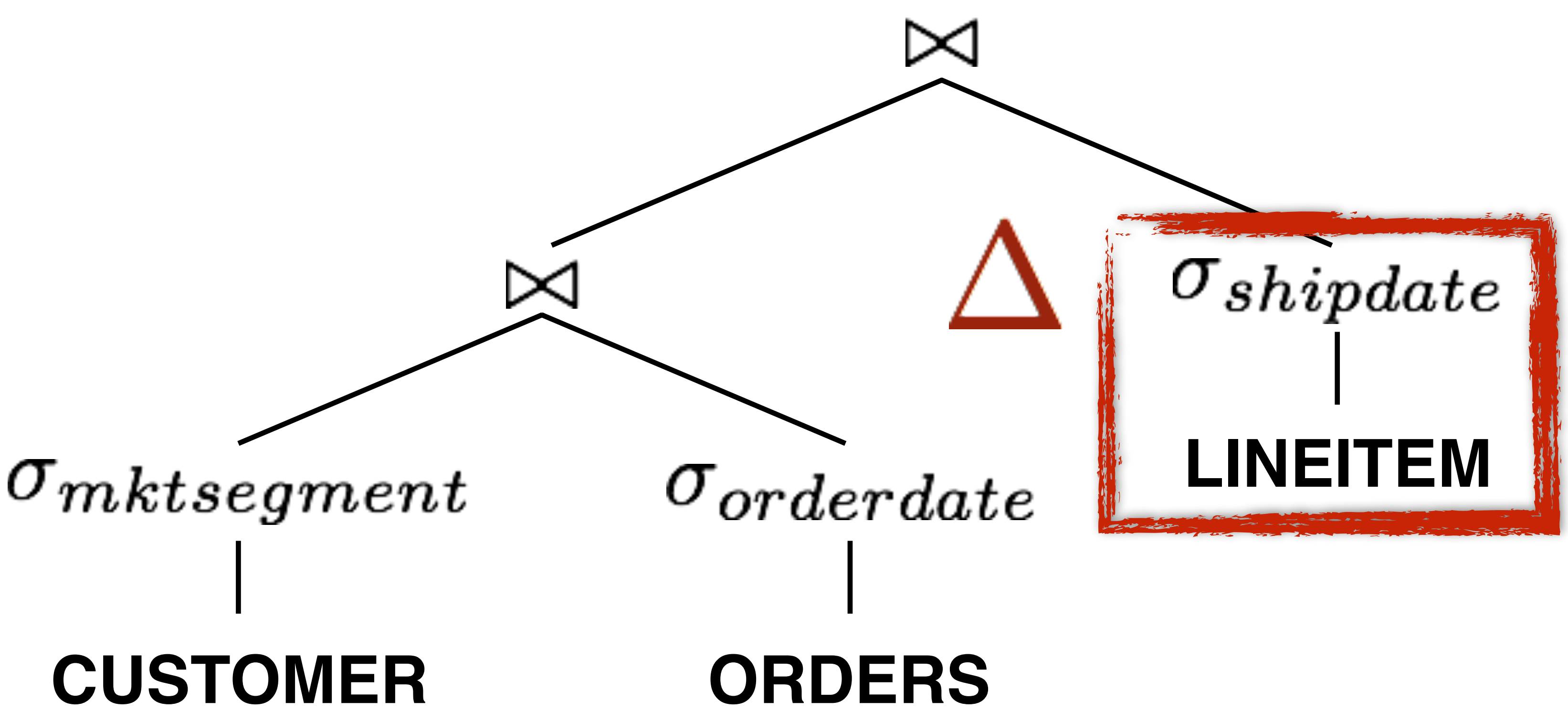
$$\Delta((\sigma(C) \bowtie \sigma(O)) \bowtie (\sigma(L))$$

Delta Queries



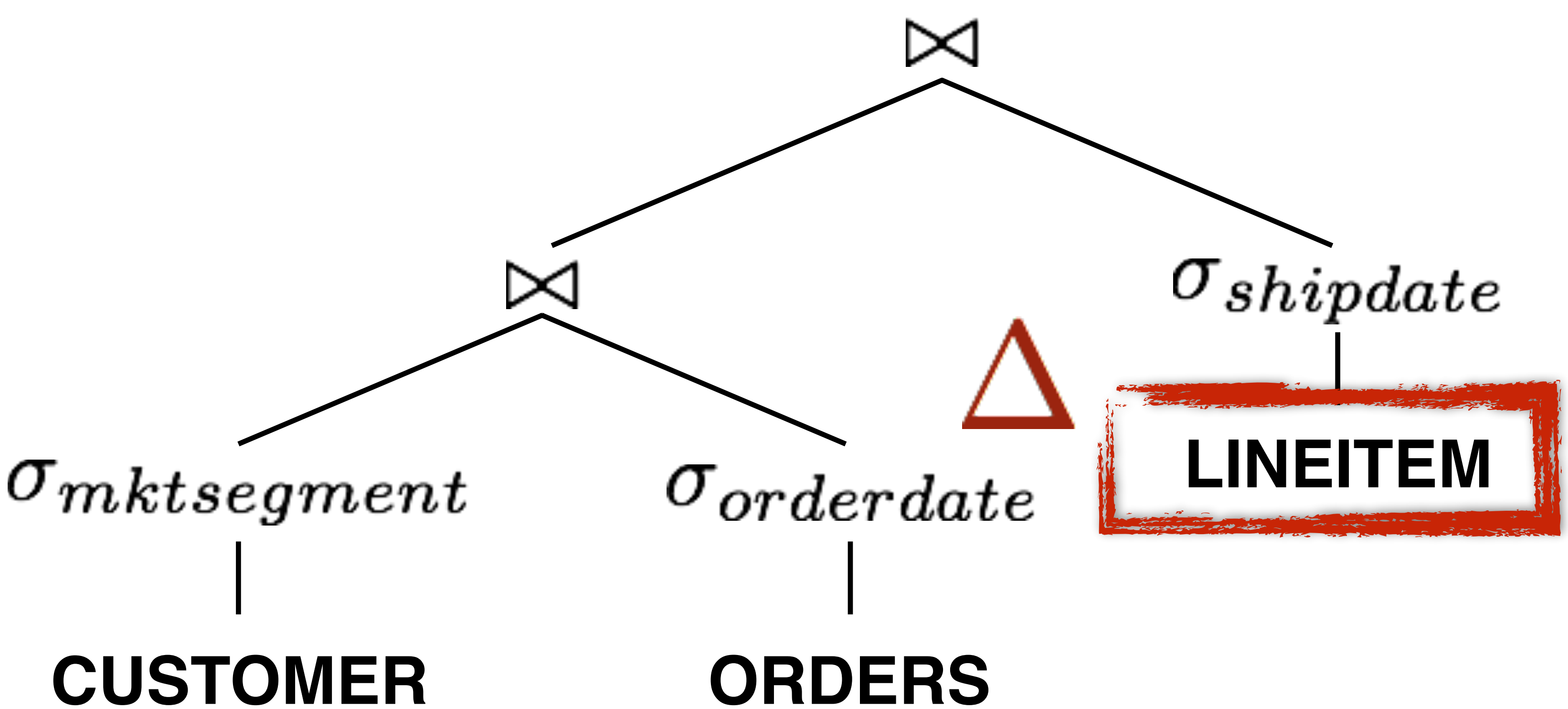
$$\Delta((\sigma(C) \bowtie \sigma(O)) \bowtie (\sigma(L))$$

Delta Queries



$$((\sigma(C) \bowtie \sigma(O)) \bowtie \Delta(\sigma(L))$$

Delta Queries



Delta Queries

```
SELECT *
FROM CUSTOMER C, ORDERS O, DELTA_LINEITEM DL
WHERE C.custkey = O.custkey
  AND DL.orderkey = O.orderkey
  AND C.mktsegment = …
  AND O.orderdate = …
  AND DL.shipdate = …
```

Multisets

{ 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 4, 4, 4, 4, 4, 4, 5 }

(not compact)

{ 1 → x3, 2 → x5, 3 → x2, 4 → x6, 5 → x1 }

Multiset representation: Tuple → ~~# of occurrences~~

multiplicity

Multiset Deltas

Insertions = Positive Multiplicity

Deletions = Negative Multiplicity

+ = Bag/Multiset Union

Multiset Deltas

What does Union do?

$\{ A \rightarrow 1, B \rightarrow 3 \} \cup \{ B \rightarrow 2, C \rightarrow 4 \} = \{ A \rightarrow 1, B \rightarrow 5, C \rightarrow 4 \}$

$\{ A \rightarrow 1 \} \cup \{ A \rightarrow -1 \} = \{ A \rightarrow 0 \} = \{ \}$

Multiset Deltas

What does Union do?

$\{ A \rightarrow 1, B \rightarrow 3 \} \cup \{ B \rightarrow 2, C \rightarrow 4 \} = \{ A \rightarrow 1, B \rightarrow 5, C \rightarrow 4 \}$

$\{ A \rightarrow 1 \} \cup \{ A \rightarrow -1 \} = \{ A \rightarrow 0 \} = \{ \}$

What does Cross Product do?

$\{ A \rightarrow 1, B \rightarrow 3 \} \times \{ C \rightarrow 4 \} = \{ \langle A,C \rangle \rightarrow ?, \langle B,C \rangle \rightarrow ? \}$

Multiset Deltas

What does Union do?

$$\{ A \rightarrow 1,\ B \rightarrow 3 \} \cup \{ B \rightarrow 2,\ C \rightarrow 4 \} = \{ A \rightarrow 1,\ B \rightarrow 5,\ C \rightarrow 4 \}$$

$$\{ A \rightarrow 1 \} \cup \{ A \rightarrow -1 \} = \{ A \rightarrow 0 \} = \{ \}$$

What does Cross Product do?

$$\{ A \rightarrow 1,\ B \rightarrow 3 \} \text{ x } \{ C \rightarrow 4 \} = \{ <A,C> \rightarrow 4,\ <B,C> \rightarrow ? \}$$

Multiset Deltas

What does Union do?

$$\{ A \rightarrow 1, B \rightarrow 3 \} \cup \{ B \rightarrow 2, C \rightarrow 4 \} = \{ A \rightarrow 1, B \rightarrow 5, C \rightarrow 4 \}$$

$$\{ A \rightarrow 1 \} \cup \{ A \rightarrow -1 \} = \{ A \rightarrow 0 \} = \{ \}$$

What does Cross Product do?

$$\{ A \rightarrow 1, B \rightarrow 3 \} \times \{ C \rightarrow 4 \} = \{ <A,C> \rightarrow 4, <B,C> \rightarrow 12 \}$$

Multiset Deltas

What does projection do?

$$\pi_{Attr1}\{ \langle A,X\rangle \rightarrow 1,\ \langle A,Y\rangle \rightarrow 2,\ \langle B,Z\rangle \rightarrow 5 \}$$

$$= \{ \langle A\rangle \rightarrow 1,\ \langle A\rangle \rightarrow 2,\ \langle B\rangle \rightarrow 5 \}$$

$$= \{ \langle A\rangle \rightarrow 3,\ \langle B\rangle \rightarrow 5 \}$$

This effect seems… familiar

If you find this subject interesting… let's chat.

http://www.dbtoaster.org