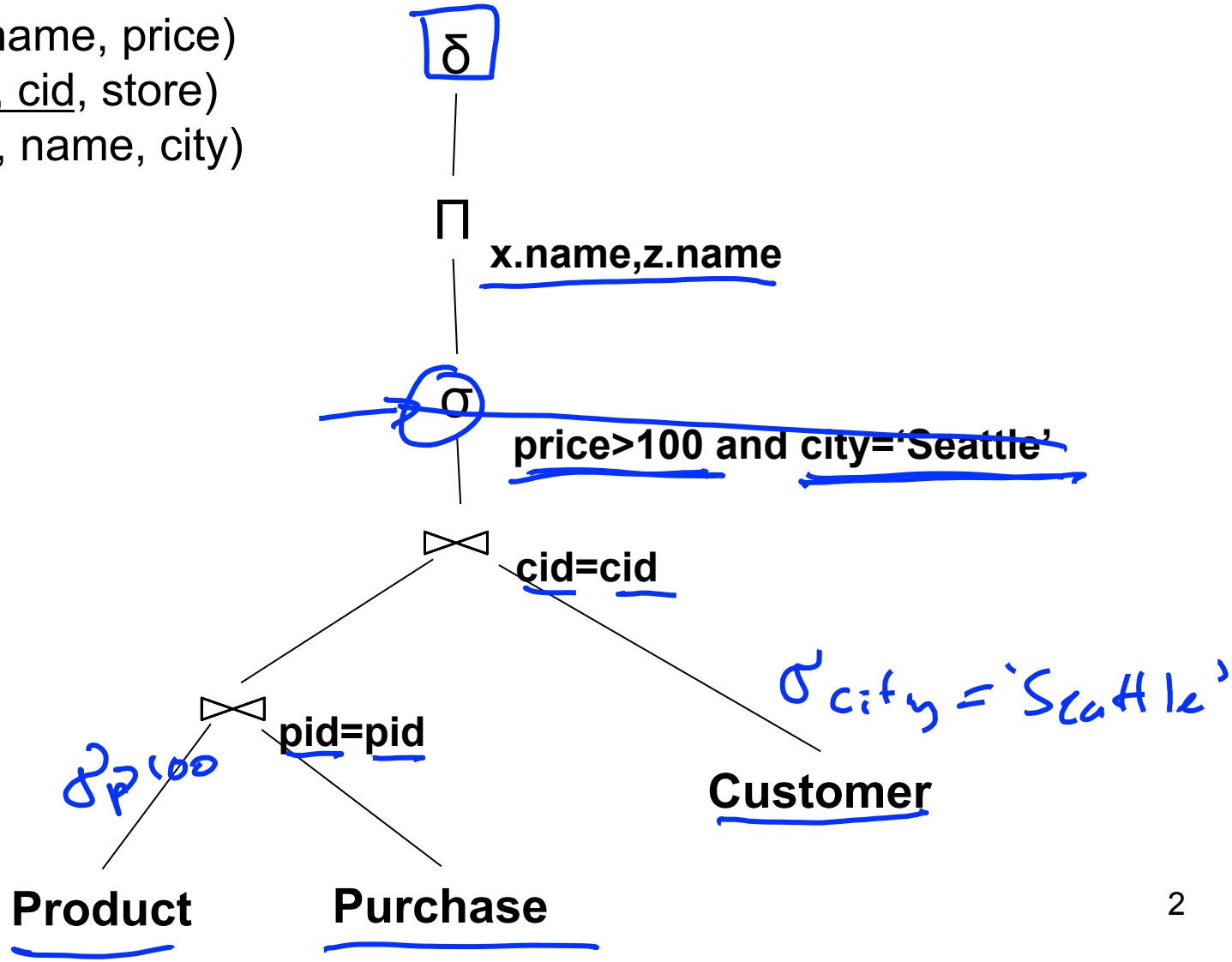
From SQL to RA

Product(pid, name, price)
Purchase(pid, cid, store)
Customer(cid, name, city)

```
SELECT DISTINCT x.name, z.name
FROM Product x, Purchase y, Customer z
WHERE x.pid = y.pid and y.cid = z.cid and
      x.price > 100 and z.city = 'Seattle'
```

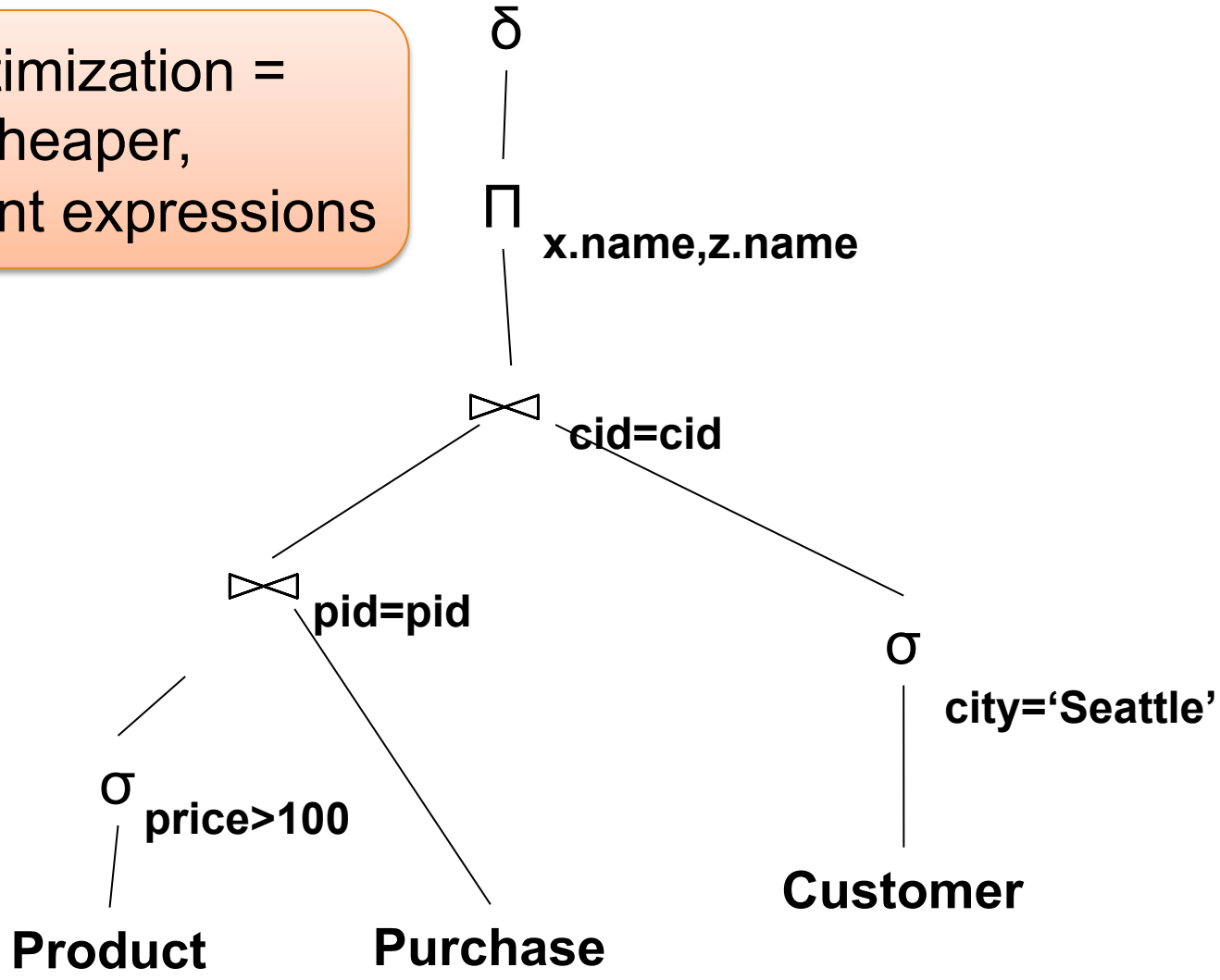
From SQL to RA

Product(pid, name, price)
Purchase(pid, cid, store)
Customer(cid, name, city)



An Equivalent Expression

Query optimization = finding cheaper, equivalent expressions



Extended RA: Operators on Bags

- Duplicate elimination d
- Grouping g
- Sorting t

DISTINCT

GROUP BY

ORDER BY

Logical Query Plan

```
SELECT city, count(*)
FROM sales
GROUP BY city
HAVING sum(price) > 100
```

T3(city, c)

$\Pi_{\text{city, c}}$

T2(city,p,c)

$\sigma_{\text{p > 100}}$

T1(city,p,c)

$\gamma_{\text{city, sum(price)}\rightarrow\text{p, count(*)}\rightarrow\text{c}}$

T1, T2, T3 = temporary tables

sales(product, city, price)

Interpreting Complicated SQL

```
SELECT binid,
        round(avg(cast(Fluo as float)),3) as Fluo,
        round(avg(cast(Oxygen as float)),3) as Oxygen,
        round(avg(cast(Nitrate_uM as float)),3) as Nitrate_uM,
        round(avg(cast(longitude as float)),3) as longitude,
        round(avg(cast(latitude as float)),3) as latitude
 FROM (
  SELECT *, cast(floor(ts) +
            floor((ts - floor(ts))*24*60/binsize) *
            binsize / (24*60) as datetime) as binid
   FROM (
     SELECT *, cast(timestamp as float) as ts, 5.0 as binsize
     FROM Tokyo_4_merged_data_time
     ) x
  ) bins
GROUP BY binid
ORDER BY binid asc
```

Interpreting Complicated SQL

```
SELECT binid,
        round(avg(cast(Fluo as float)),3) as Fluo,
        round(avg(cast(Oxygen as float)),3) as Oxygen,
        round(avg(cast(Nitrate_uM as float)),3) as Nitrate_uM,
        round(avg(cast(longitude as float)),3) as longitude,
        round(avg(cast(latitude as float)),3) as latitude
 FROM (
   SELECT *, cast(floor(ts) +
                floor((ts - floor(ts))*24*60/binsize) *
                binsize / (24*60) as datetime) as binid
    FROM (
      SELECT *, cast(timestamp as float) as ts, 5.0 as binsize
      FROM Tokyo_4_merged_data_time
      ) x
   ) bins
GROUP BY binid
ORDER BY binid asc
```

Interpreting Complicated SQL

```
SELECT x.strain, x.chr, x.region as snp_region, x.start_bp as snp_start_bp
   , x.end_bp as snp_end_bp, w.start_bp as nc_start_bp, w.end_bp as nc_end_bp
   , w.category as nc_category
   , CASE WHEN (x.start_bp >= w.start_bp AND x.end_bp <= w.end_bp)
                THEN x.end_bp - x.start_bp + 1
           WHEN (x.start_bp <= w.start_bp AND w.start_bp <= x.end_bp)
                THEN x.end_bp - w.start_bp + 1
           WHEN (x.start_bp <= w.end_bp AND w.end_bp <= x.end_bp)
                THEN w.end_bp - x.start_bp + 1
     END  AS len_overlap

FROM hotspots_deserts x
INNER JOIN table_noncoding_positions w
ON x.chr = w.chr
WHERE (x.start_bp >= w.start_bp AND x.end_bp <= w.end_bp)
OR (x.start_bp <= w.start_bp AND w.start_bp <= x.end_bp)
OR (x.start_bp <= w.end_bp AND w.end_bp <= x.end_bp)
ORDER BY x.strain, x.chr ASC, x.start_bp ASC
```

Interpreting Complicated SQL

```
SELECT x.strain, x.chr, x.region as snp_region, x.start_bp as snp_start_bp
   , x.end_bp as snp_end_bp, w.start_bp as nc_start_bp, w.end_bp as nc_end_bp
   , w.category as nc_category
   , CASE WHEN (x.start_bp >= w.start_bp AND x.end_bp <= w.end_bp)
                THEN x.end_bp - x.start_bp + 1
           WHEN (x.start_bp <= w.start_bp AND w.start_bp <= x.end_bp)
                THEN x.end_bp - w.start_bp + 1
           WHEN (x.start_bp <= w.end_bp AND w.end_bp <= x.end_bp)
                THEN w.end_bp - x.start_bp + 1
     END AS len_overlap

FROM hotspots_deserts x
INNER JOIN table_noncoding_positions w
ON x.chr = w.chr
WHERE (x.start_bp >= w.start_bp AND x.end_bp <= w.end_bp)
OR (x.start_bp <= w.start_bp AND w.start_bp <= x.end_bp)
OR (x.start_bp <= w.end_bp AND w.end_bp <= x.end_bp)
ORDER BY x.strain, x.chr ASC, x.start_bp ASC
```

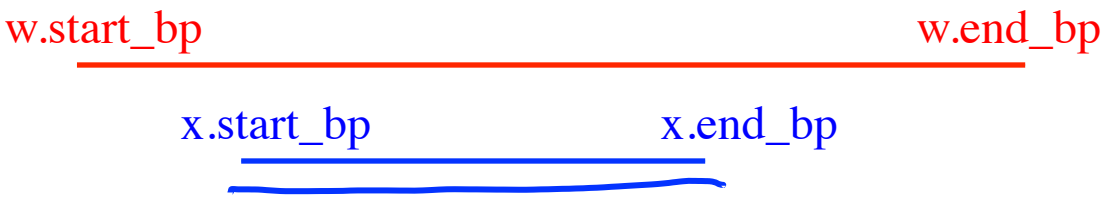
Interpreting Complicated SQL

```
SELECT x.strain, x.chr, x.region as snp_region, x.start_bp as snp_start_bp
   , x.end_bp as snp_end_bp, w.start_bp as nc_start_bp, w.end_bp as nc_end_bp
   , w.category as nc_category
   , len_overlap(x.start_bp, x.end_bp, w.start_bp, w.end_bp)
 FROM hotspots_deserts x
 INNER JOIN table_noncoding_positions w
 ON x.chr = w.chr
 WHERE (x.start_bp >= w.start_bp AND x.end_bp <= w.end_bp)
 OR (x.start_bp <= w.start_bp AND w.start_bp <= x.end_bp)
 OR (x.start_bp <= w.end_bp AND w.end_bp <= x.end_bp)
 ORDER BY x.strain, x.chr ASC, x.start_bp ASC
```

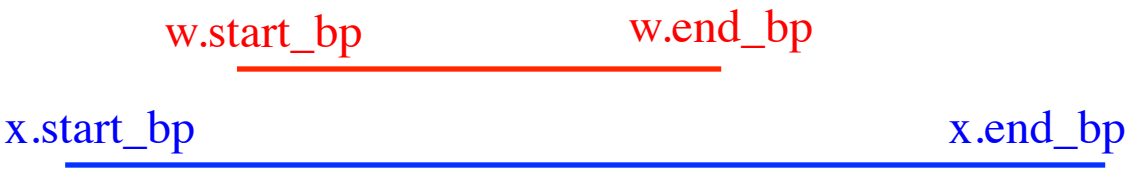
Interpreting Complicated SQL

```
SELECT x.strain, x.chr, x.region as snp_region, x.start_bp as snp_start_bp
   , x.end_bp as snp_end_bp, w.start_bp as nc_start_bp, w.end_bp as nc_end_bp
   , w.category as nc_category
   , len_overlap(x.start_bp, x.end_bp, w.start_bp, w.end_bp)
 FROM hotspots_deserts x
 INNER JOIN table_noncoding_positions w
 ON x.chr = w.chr
 WHERE (x.start_bp >= w.start_bp AND x.end_bp <= w.end_bp)
 OR (x.start_bp <= w.start_bp AND w.start_bp <= x.end_bp)
 OR (x.start_bp <= w.end_bp AND w.end_bp <= x.end_bp)
 ORDER BY x.strain, x.chr ASC, x.start_bp ASC
```

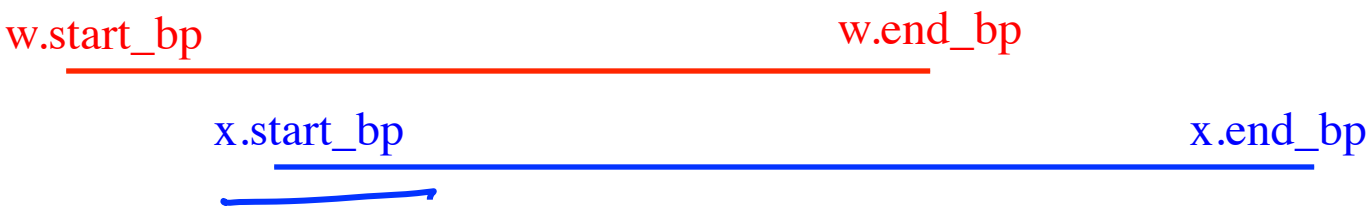
(x.start_bp >= w.start_bp AND x.end_bp <= w.end_bp)



OR (x.start_bp <= w.start_bp AND w.start_bp <= x.end_bp)



OR (x.start_bp <= w.end_bp AND w.end_bp <= x.end_bp)



Interpreting Complicated SQL

```
SELECT x.strain, x.chr, x.region as snp_region, x.start_bp as snp_start_bp
   , x.end_bp as snp_end_bp, w.start_bp as nc_start_bp, w.end_bp as nc_end_bp
   , w.category as nc_category
   , len_overlap(x.start_bp, x.end_bp, w.start_bp, w.end_bp)
 FROM hotspots_deserts x
        , table_noncoding_positions w
WHERE x.chr = w.chr
   AND overlaps(x.start_bp, x.end_bp, w.start_bp, w.end_bp)
ORDER BY x.strain, x.chr ASC, x.start_bp ASC
```

Theta-Join!