Introduction to Data Structure (Data Management) Lecture 5(Continuation)

Felipe P. Vista IV



Ordering Results

Purchase(pid, product, price, quantity, month)

```
SELECT product, sum(price*quantity)
FROM purchase

GROUP BY product
ORDER BY sum(price*quantity) DESC
```

FWGOS

Ordering Results

Purchase(pid, product, price, quantity, month)

```
SELECT product, sum(price*quantity) as rev
FROM purchase

GROUP BY product
ORDER BY rev DESC
```

FWGOS

Note: some SQL engines will want to use the syntax ORDER BY sum(price*quantity)

HAVING Clause

Purchase(pid, product, price, quantity, month)

Same query as before, except that consider only products that had at least 15 items sold.

```
SELECT product, sum(price*quantity)
FROM purchase
WHERE price > 125.00
GROUP BY product

(HAVING) sum(quantity) > 15
```

FWGHOS

HAVING clause contains conditions on groups.

Practice

Purchase (pid, product, price, quantity, month)

Compute the total income per month Show only months with less than 10 items sold Order by quantity sold and display as "totalSold"

FWGHOS

Practice

Purchase(pid, product, price, quantity, month)

Compute the total income per month Show only months with less than 6 items sold Order by quantity sold and display as "totalSold"

```
SELECT month, sum(price*quantity),
sum(quantity) AS totalSold
FROM purchase
GROUP BY month
HAVING sum(quantity) < 10
ORDER BY sum(quantity)
```

FWGHOS

WHERE vs HAVING

WHERE

- condition is applied to individual rows/records/tuple
- the rows may or may not contribute to the aggregate
- no aggregates allowed

WHERE vs HAVING

WHERE

- condition is applied to individual rows/records/tuple ∫ ≤ ⊱ ⋈/
- the rows may or may not contribute to the aggregate
- no aggregates allowed

HAVING

- condition is applied to entire group $\subset G \mathcal{B} \mathcal{H}$
- entire group is returned, or not at all
- may use aggregate functions in the group

Mystery QUERIES

```
SELECT month, sum(quantity), max(price)
FROM purchase
GROUP BY month
```

```
SELECT month, sum(quantity)
FROM purchase
GROUP BY month
```

```
SELECT month
FROM purchase
GROUP BY month
```

Note:
DISTINCT is a special case of GROUP BY

Aggregates and Joins

```
CREATE table Product(
    pid int Primary Key,
    pname varchar(15),
    manufacturer varchar(15);

Insert into product values(1,'bike','Patty pats');
Insert into product values(2,'scooter','Divanas');
Insert into product values(3,'genesis','YhaNins');
Insert into product values(4,'suv','Sattavis');
Insert into product values(5,'truck','MattBurts');
```

Aggregates + Join Example

Purchase (pid, product, price, quantity, month)
Product (pid, pname, manufacturer)

```
SELECT manufacturer, count(*)

FROM product, purchase

WHERE pname = product

GROUP BY manufacturer
```

Let's figure out what these mean...

```
SELECT manufacturer, month, count(*)

FROM product, purchase

WHERE pname = product

GROUP BY manufacturer, month
```

Nested Loop Semantics for SFW

```
SELECT x_1.a_1, x_2.a_2,..., x_m.a_m

FROM R_1 AS x_1, R_2 AS x_2,..., R_m AS x_m

WHERE condition(s)
```

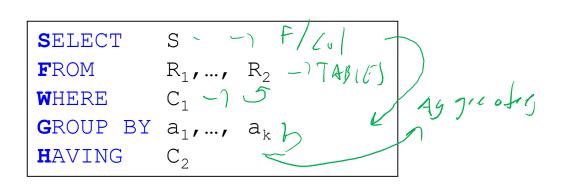
```
for x_1 in R_1: x_1 x_2 x_1 x_2 x_3 x_4 x_4 x_5 x_6 x_6 x_6 in x_6: x_6 x_7 x_8 x_8:

if x_8:

output (x_1, x_2, ..., x_m):

output (x_1, a_1, x_2, a_2, ..., x_m)
```

Semantics for SFWGH



- = may contain attributes $a_1,...,a_k$ and/or any aggregates, but NO OTHER ATTRIBUTES
- \underline{C}_1 = is any condition on the attributes in $R_1,...,R_n$
- C_2 = is any condition on the aggregate expressions and on attributes $a_1,...,a_k$

Semantics for SFWGH

```
SELECTSFROMR_1, ..., R_2WHEREC_1GROUP BYa_1, ..., a_kHAVINGC_2
```

Execution Order:



Evaluation Steps:

- 1. Evaluate FROM-WHERE using Nested Loop Semantics
- 2. GROUP by the attributes $a_1,...,a_k$
- 3. Apply condition C2 to each group (may have aggregates)
- 4. Compute aggregates in S and return the result

Aggregates + Join Example

Purchase (pid, product, price, quantity, month)
Product (pid, pname, manufacturer)

```
SELECT manufacturer, count(*)

FROM product, purchase

WHERE pname = product

GROUP BY manufacturer
```

SO, what do these queries mean?

```
SELECT manufacturer, month, count(*)
FROM product, purchase
WHERE pname = product
GROUP BY manufacturer, (month
```

Empty Groups

- In the result of a group by query
 - there is one row per group in result
- No group can be empty!
- Specifically, count(*) is never 0

```
SELECT manufacturer, count(*)
FROM product, purchase
WHERE pname = product
GROUP BY manufacturer
```

What if there are no purchases for a manufacturer?

Empty Group Solution: Outer Join

```
SELECT manufacturer, count (quantity) FROM product LEFT OUTER JOIN purchase
ON pname = product
GROUP BY manufacturer
```

Why not count(*)?

RECALL

Exercise 1:

- Purchase (pid, product, price, quantity, month)
- -) Product (pid, pname, manufacturer)

Find all manufacturers with more than 10 items sold
Return the name of the manufacturer and num of items sold

Exercise 1:

```
Purchase (pid, product, price, quantity, month)
Product(pid, pname, manufacturer)
```

Find all manufacturers with more than 10 items sold Return the name of the manufacturer and num of items sold

```
SELECT manufacturer, sum(quantity)
FROM product, purchase
WHERE pname = product
GROUP BY manufacturer
HAVING sum(quantity) > 10
```

Exercise 2:

```
Purchase (pid, product, price, quantity, month)
Product (pid, pname, manufacturer)
```

Find all manufacturers with more than 1 distinct product sold

Return the name of the manufacturer and num of distinct products sold

```
SELECT ?????, count (DISTINCT ?????)

FROM ?????, ?????

WHERE ????? = ?????

GROUP BY ????? Many & plod 1

HAVING count (DISTINCT ?????) > ?????
```

Exercise 2:

```
Purchase (pid, product, price, quantity, month)
Product (pid, pname, manufacturer)
```

Find all manufacturers with more than 1 distinct product sold Return the name of the manufacturer and num of distinct products sold

```
SELECT manufacturer, count(DISTINCT product)
FROM product, purchase
WHERE pname = product
GROUP BY manufacturer
HAVING count(DISTINCT product) > 1
```

Thank you.