Introduction to Database Systems CSE 444

Lecture 03: SQL

October(!) 1, 2007
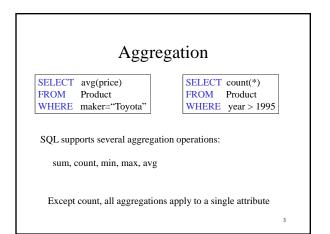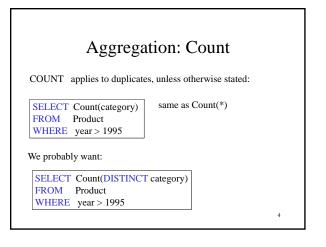
1
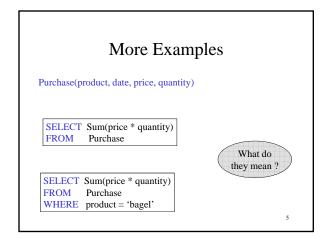
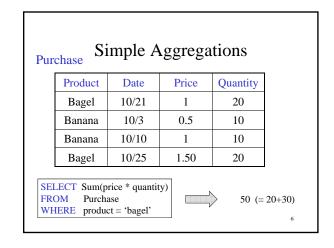# Outline

- Subqueries (6.3)
- Aggregations (6.4.3 – 6.4.6)
- Examples, examples, examples…

Read the entire chapter 6 !

2

Aggregation

SELECT	avg(price)
FROM	Product
WHERE	maker="Toyota"

SELECT	count(*)
FROM	Product
WHERE	year > 1995

SQL supports several aggregation operations:

sum, count, min, max, avg

Except count, all aggregations apply to a single attribute

3

Aggregation: Count

COUNT applies to duplicates, unless otherwise stated:

SELECT	Count(category)
FROM	Product
WHERE	year > 1995

same as Count(*)

We probably want:

SELECT	Count(DISTINCT category)
FROM	Product
WHERE	year > 1995

4

More Examples

Purchase(product, date, price, quantity)

| SELECT | Sum(price * quantity) |
| FROM | Purchase |

What do they mean ?

SELECT	Sum(price * quantity)
FROM	Purchase
WHERE	product = 'bagel'

5

Simple Aggregations

Purchase

Product	Date	Price	Quantity
Bagel	10/21	1	20
Banana	10/3	0.5	10
Banana	10/10	1	10
Bagel	10/25	1.50	20

SELECT	Sum(price * quantity)
FROM	Purchase
WHERE	product = 'bagel'

50 (= 20+30)

6

1

Grouping and Aggregation

Purchase(product, date, price, quantity)

Find total sales after 10/1/2005 per product.

```
SELECT      product, Sum(price*quantity) AS TotalSales
FROM        Purchase
WHERE       date > '10/1/2005'
GROUP BY    product
```

Let's see what this means…

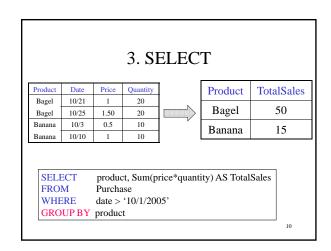## Grouping and Aggregation
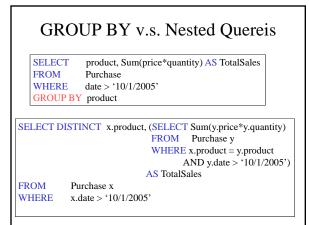
1. Compute the FROM and WHERE clauses.

2. Group by the attributes in the GROUPBY

3. Compute the SELECT clause: grouped attributes and aggregates.

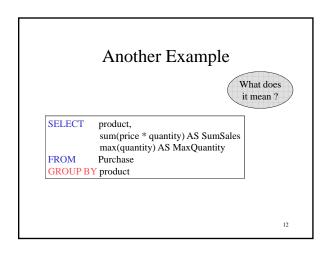
1&2. FROM-WHERE-GROUPBY

Product	Date	Price	Quantity
Bagel	10/21	1	20
Bagel	10/25	1.50	20
Banana	10/3	0.5	10
Banana	10/10	1	10

---

3. SELECT

Product	Date	Price	Quantity
Bagel	10/21	1	20
Bagel	10/25	1.50	20
Banana	10/3	0.5	10
Banana	10/10	1	10

Product	TotalSales
Bagel	50
Banana	15

```
SELECT      product, Sum(price*quantity) AS TotalSales
FROM        Purchase
WHERE       date > '10/1/2005'
GROUP BY    product
```

GROUP BY v.s. Nested Quereis

```
SELECT      product, Sum(price*quantity) AS TotalSales
FROM        Purchase
WHERE       date > '10/1/2005'
GROUP BY    product
```

```
SELECT DISTINCT  x.product, (SELECT Sum(y.price*y.quantity)
                             FROM    Purchase y
                             WHERE x.product = y.product
                                 AND y.date > '10/1/2005')
                             AS TotalSales
FROM        Purchase x
WHERE       x.date > '10/1/2005'
```

Another Example

What does it mean ?

```
SELECT      product,
            sum(price * quantity) AS SumSales
            max(quantity) AS MaxQuantity
FROM        Purchase
GROUP BY product
```
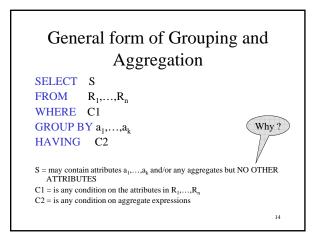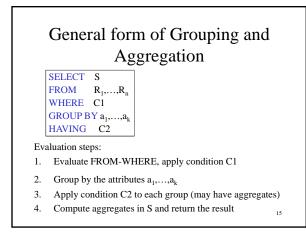
HAVING Clause

Same query, except that we consider only products that had at least 100 buyers.
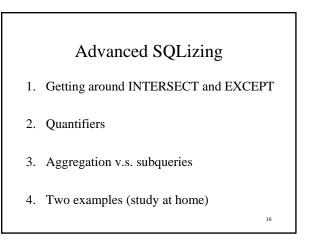
```
SELECT    product, Sum(price * quantity)
FROM      Purchase
WHERE     date > '10/1/2005'
GROUP BY  product
HAVING    Sum(quantity) > 30
```
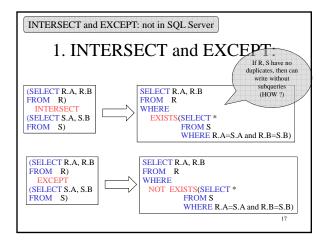
HAVING clause contains conditions on aggregates.

13

General form of Grouping and Aggregation

SELECT   S
FROM $R_1,\ldots,R_n$
WHERE   C1
GROUP BY $a_1,\ldots,a_k$
HAVING   C2

Why ?

S = may contain attributes $a_1,\ldots,a_k$ and/or any aggregates but NO OTHER ATTRIBUTES

C1 = is any condition on the attributes in $R_1,\ldots,R_n$

C2 = is any condition on aggregate expressions

14

General form of Grouping and Aggregation

```
SELECT    S
FROM      R₁,…,Rₙ
WHERE     C1
GROUP BY  a₁,…,aₖ
HAVING    C2
```

Evaluation steps:

1. Evaluate FROM-WHERE, apply condition C1
2. Group by the attributes $a_1,\ldots,a_k$
3. Apply condition C2 to each group (may have aggregates)
4. Compute aggregates in S and return the result

15

Advanced SQLizing

1. Getting around INTERSECT and EXCEPT

2. Quantifiers

3. Aggregation v.s. subqueries

4. Two examples (study at home)

16

INTERSECT and EXCEPT: not in SQL Server

1. INTERSECT and EXCEPT:

If R, S have no duplicates, then can write without subqueries (HOW ?)

```
(SELECT R.A, R.B
 FROM   R)
   INTERSECT
(SELECT S.A, S.B
 FROM   S)
```

```
SELECT R.A, R.B
FROM   R
WHERE
  EXISTS(SELECT *
         FROM S
         WHERE R.A=S.A and R.B=S.B)
```

```
(SELECT R.A, R.B
 FROM   R)
   EXCEPT
(SELECT S.A, S.B
 FROM   S)
```

```
SELECT R.A, R.B
FROM   R
WHERE
  NOT EXISTS(SELECT *
             FROM S
             WHERE R.A=S.A and R.B=S.B)
```

17

---

2. Quantifiers

Product (pname, price, company)
Company(cname, city)

Find all companies that make <u>some</u> products with price < 100

```
SELECT DISTINCT  Company.cname
FROM    Company, Product
WHERE   Company.cname = Product.company and Product.price < 100
```
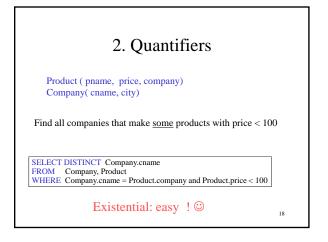
Existential: easy ! ☺

18

---

2. Quantifiers

Product (pname, price, company)
Company(cname, city)

Find all companies that make only products with price < 100

same as:

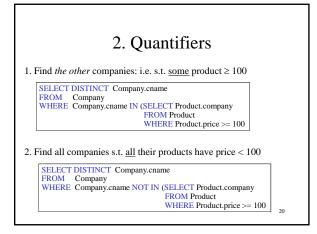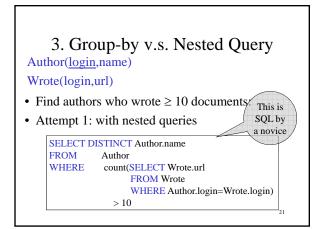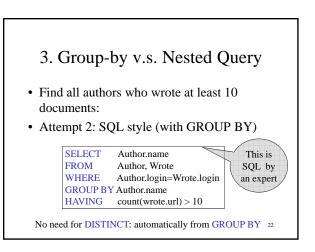Find all companies s.t. all of their products have price < 100

Universal: hard ! ☹

19

---

2. Quantifiers

1. Find *the other* companies: i.e. s.t. <u>some</u> product ≥ 100

```
SELECT DISTINCT  Company.cname
FROM      Company
WHERE  Company.cname IN (SELECT Product.company
                         FROM Product
                         WHERE Product.price >= 100
```

2. Find all companies s.t. <u>all</u> their products have price < 100

```
SELECT DISTINCT  Company.cname
FROM      Company
WHERE  Company.cname NOT IN (SELECT Product.company
                             FROM Product
                             WHERE Product.price >= 100
```

20

---

3. Group-by v.s. Nested Query

Author(<u>login</u>,name)

Wrote(login,url)

- Find authors who wrote ≥ 10 documents
- Attempt 1: with nested queries

This is SQL by a novice

```
SELECT DISTINCT Author.name
FROM       Author
WHERE      count(SELECT Wrote.url
                 FROM Wrote
                 WHERE Author.login=Wrote.login)
           > 10
```

21

---

3. Group-by v.s. Nested Query

- Find all authors who wrote at least 10 documents:
- Attempt 2: SQL style (with GROUP BY)

This is SQL by an expert

```
SELECT        Author.name
FROM          Author, Wrote
WHERE         Author.login=Wrote.login
GROUP BY Author.name
HAVING        count(wrote.url) > 10
```

No need for DISTINCT: automatically from GROUP BY   22

---

3. Group-by v.s. Nested Query

Author(<u>login</u>,name)

Wrote(login,url)

Mentions(url,word)

Find authors with vocabulary ≥ 10000 words:

```
SELECT       Author.name
FROM         Author, Wrote, Mentions
WHERE        Author.login=Wrote.login AND Wrote.url=Mentions.url
GROUP BY Author.name
HAVING       count(distinct Mentions.word) > 10000
```

23

---

4. Two Examples

Store(sid, sname)
Product(pid, pname, price, sid)

Find all stores that sell *only* products with price > 100

same as:

Find all stores s.t. all their products have price > 100)

24

4

Slide 25

SELECT Store.name
FROM   Store, Product
WHERE   Store.sid = Product.sid
GROUP BY Store.sid, Store.name
HAVING 100 < min(Product.price)

Why both ?

Almost equivalent…

SELECT Store.name
FROM   Store
WHERE
   100 < ALL (SELECT Product.price
                    FROM product
                    WHERE Store.sid = Product.sid)

SELECT Store.name
FROM   Store
WHERE   Store.sid NOT IN
              (SELECT Product.sid
                    FROM Product
                    WHERE Product.price <= 100)

25

Slide 26

Two Examples

Store(sid, sname)
Product(pid, pname, price, sid)

For each store,
find its most expensive product

26

Slide 27

Two Examples

This is easy but doesn't do what we want:

SELECT Store.sname, max(Product.price)
FROM   Store, Product
WHERE Store.sid = Product.sid
GROUP BY Store.sid, Store.sname

Better:

SELECT Store.sname, x.pname
FROM   Store, Product x
WHERE Store.sid = x.sid and
                x.price >=
              ALL (SELECT y.price
                    FROM Product y
                    WHERE Store.sid = y.sid)

But may
return
multiple
product names
per store

Slide 28

Two Examples

Finally, choose some pid arbitrarily, if there are many
with highest price:

SELECT Store.sname, max(x.pname)
FROM   Store, Product x
WHERE Store.sid = x.sid and
                x.price >=
              ALL (SELECT y.price
                    FROM Product y
                    WHERE Store.sid = y.sid)
GROUP BY Store.sname

28