Lecture 6: Aggregate queries

Monday, February 9, 2015
Agenda for today

- Chapters 5: aggregate queries
- Practice aggregate queries in class
Standard aggregate operators

- count
- sum
- avg
- max
- min
Count examples

SELECT COUNT(*)
FROM invoices
WHERE invoice_date >= '09-FEB-2014'

SELECT COUNT(*) AS number_of_invoices
FROM invoices
WHERE invoice_total > 50
More count examples

SELECT COUNT(*)
FROM customers

**********

SELECT COUNT(customer_city)
FROM customers

COUNT(customer_city) != COUNT(*) Why?

**********

SELECT COUNT(DISTINCT customer_city)
FROM customers
Sum examples

SELECT SUM(invoice_total - payment_total)
FROM invoices

SELECT SUM(order_qty * unit_price)
FROM order_details, items
WHERE order_details.item_id = items.item_id
Min and Max examples

SELECT MIN(invoice_total) AS lowest_invoice_total, 
MAX(invoice_total) AS highest_invoice_total, 
COUNT(*) AS number_of_invoices 
FROM invoices

SELECT MIN(vendor_name) AS first_vendor, 
MAX(vendor_name) AS last_vendor, 
COUNT(vendor_name) AS number_of_vendors 
FROM vendors
Six clauses in SQL query:

SELECT
FROM
WHERE
GROUP BY
HAVING
ORDER BY
Grouping example

SELECT c.customer_city, SUM(od.order_qty * i.unit_price) AS total_sales
FROM customers c, orders o, order_details od, items i
WHERE c.customer_id = o.customer_id
AND o.order_id = od.order_id
AND od.item_id = i.item_id
GROUP BY c.customer_city
ORDER BY c.customer_city

Evaluation steps:
1. Compute the FROM and WHERE clauses
2. Compute the attribute(s) in the GROUP BY
3. Compute the aggregate value(s) in the SELECT clause
Grouping with left outer join example

```
SELECT c.customer_city, sum(od.order_qty * i.unit_price) as total_sales
FROM customers c LEFT JOIN orders o
ON c.customer_id = o.customer_id
LEFT JOIN order_details od
ON o.order_id = od.order_id
LEFT JOIN items i
ON od.item_id = i.item_id
GROUP BY c.customer_city
ORDER BY c.customer_city
```

Observation: Now empty groups are also included
Grouping with having example

SELECT c.customer_city, SUM(od.order_qty * i.unit_price) AS total_sales
FROM customers c, orders o, order_details od, items i
WHERE c.customer_id = o.customer_id
AND o.order_id = od.order_id
AND od.item_id = i.item_id
GROUP BY c.customer_city
HAVING SUM(od.order_qty * i.unit_price) > 50
ORDER BY c.customer_city

Evaluation steps:
1. Compute the FROM and WHERE clauses
2. Compute the attribute(s) in the GROUP BY and apply the HAVING condition to each group
3. Compute the aggregate value(s) in the SELECT clause
Syntactic rule: Every non-aggregated attribute that is in the SELECT clause of a GROUP BY query must also appear in the GROUP BY clause. Why?

SELECT c.customer_state, c.customer_city, 
       SUM(od.order_qty * i.unit_price) As total_sales 
FROM customers c 
LEFT JOIN orders o 
ON c.customer_id = o.customer_id 
LEFT JOIN order_details od 
ON o.order_id = od.order_id 
LEFT JOIN items i 
ON od.item_id = i.item_id 
GROUP BY c.customer_state, c.customer_city 
ORDER BY c.customer_state, c.customer_city
What is wrong with this query?

```sql
SELECT v.vendor_id, v.vendor_name, count(*) AS number_invoices
FROM invoices i, vendors v
WHERE i.vendor_id = v.vendor_id
GROUP BY v.vendor_id
HAVING COUNT(*) >= 2
ORDER BY v.vendor_id, v.vendor_name
```
Using WITH to find for each vendor, the invoice_number for the highest invoice.

```sql
WITH temp AS (SELECT v.vendor_id, v.vendor_name,
               MAX(i.invoice_total) AS highest_invoice
FROM invoices i, vendors v
WHERE i.vendor_id = v.vendor_id
GROUP BY v.vendor_id, v.vendor_name)
SELECT v.vendor_id, v.vendor_name, i.invoice_number, t.highest_invoice
FROM vendors v, invoices i, temp t
WHERE v.vendor_id = i.vendor_id
AND v.vendor_id = t.vendor_id
AND v.vendor_name = t.vendor_name
AND i.invoice_total = t.highest_invoice
ORDER BY t.highest_invoice DESC
```
In-class exercises

Items (item_id, item_description, item_price)
Order_Details (order_id, item_id, order_qty)
Orders (order_id, customer_id, order_date, shipped_date)
Customers (customer_id, customer_first_name, customer_last_name, customer_address, customer_city, customer_zip…)

Ex #1: Find customers who are from California, but not LA.

Ex #2: Find the number of customers who purchased the same items. Return the item_description along with the number of customers who purchased that item.

Ex #3: Find customers who have spent less than $5 on an order as well as customers who have never placed an order.

Ex #4: Compute the total number of sales for each item. Want to also include items that didn’t sell.
Next class

- Conceptual design: Chapter 9 in Murach textbook
- Quiz #2