IT2201 / IT2601 / IT2564 / IT2621 / IT2521 / IT2323

Database Management Systems

### Unit 8 (Part B)

Structured Query Language (Advanced SELECT)

1

# Topics (Advanced SELECT)

- Using Aggregate functions
- Using GROUP BY
- Using HAVING
- Using Subqueries

2

- 5 aggregate functions can be applied on column values :
  - COUNT, SUM, AVG, MIN, MAX
  - Each operates on a single column of a table and return single value
    - Select SUM(total\_price) from order\_detail;
  - Use in <u>SELECT</u>, <u>HAVING</u> clauses only

3

#### ISO standard defines five aggregate functions:

Function	Format	Returns
COUNT	COUNT(*)	Counts all rows of a table, regardless of whether nulls or duplicate values occur.
	COUNT(DISTINCT column-name)	No. of unique values in the specified column.
AVG	AVG (column-name)	Average of values in a specified numeric column.
SUM	SUM (column-name)	Sum of values in a specified numeric column.
MIN	MIN (column-name)	Lowest value in the specified column.
MAX	MAX (column-name)	Highest value in the specified column.

#### COUNT

- Find the total number of orders received
  - Select COUNT(\*) from orders ;

ORDER_NUM	ORDER_DATE	CUSTOMER _NUM	SHIP_DATE
1001	20-May-06	103	1-Jun-06
1002	21-May-06	101	26-May-06
1003	22-May-06	103	23-May-06
1004	22-May-06	102	30-May-06

Count(\*)

1 row selected

**Orders** 

## Use of Aggregate Functions

#### COUNT(DISTINCT)

- Find the total number of customers who have placed orders
  - Select COUNT(DISTINCT customer\_num) from orders;

		CUSTOMER	
ORDER_NUM	ORDER_DATE	_NUM	SHIP_DATE
1001	20-May-06	103	1-Jun-06
1002	21-May-06	101	26-May-06
1003	22-May-06	103	23-May-06
1004	22-May-06	102	30-May-06

1 row selected

Orders

•What would be the result if the keyword DISTINCT is not used?

- AVG()
  - Find the average product unit price
    - Select AVG(unit\_price) from product;

PROD_NUM	UNIT_PRICE
113	681
120	37

1 row selected

Avg(unit\_price)

**Product** 

# Use of Aggregate Functions

- □ SUM()
  - Find the total sales from all the orders
    - Select SUM(total\_price) from order\_detail;

ORDER_NUM	TOTAL_PRICE
1001	100
1001	250
1002	100

Sum(total\_price) 450

1 row selected

Order\_detail

- MIN(), MAX()
  - Find the maximum and the minimum shipment charge
    - Select MAX(ship\_charge), MIN(ship\_charge) from orders;

ORDER_NUM	SHIP_CHARGE
1001	150
1001	250
1002	75

Max(ship\_charge) min(ship\_charge)

1 row selected

Order\_detail

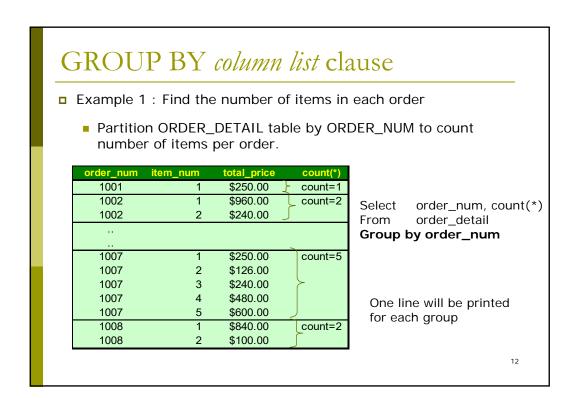
### GROUP BY column list clause

- To group rows and produce sub-totals
  - i.e. to produce a single summary row for each group
- □ Column list:
  - any base table <u>columns</u>
  - calculated columns
- How it works :
  - Partitions a table into groups of rows that have something in common to apply a function on each group of rows.
  - Most often combined with aggregate functions that produce summary values for each of those groups 10

11

#### GROUP BY column list clause Example 1 : Find the <u>number of items</u> in <u>each order</u> Order 1001 has 1 item \$250.00 1001 \$960.00 1002 Order 1002 has 2 items \$126.00 1007 5 \$600.00 1007 \$240.00 1002 Order 1007 has 5 items \$250.00 1007 \$840.00 1008 3 \$240.00 1007 \$480.00 4 1007 Order 1008 has 2 items \$100.00 1008 This kind of operation requires the **Group by** clause. Order\_Detail In this example, we are grouping by order\_num.

Group by order\_num



### GROUP BY column list clause

Select order\_num, count(\*) from order\_detail

group by order\_num;

■ Example 1 (output) :

order_num	count(*)
1001 1002	1 2
1007	5
1008	2
••••	••••

13

#### GROUP BY column list clause

- Example 2 : Find the total price of each order
  - Partition ORDER\_DETAIL table by ORDER\_NUM to find out the sum of the item's total price in each order.

order_num	total_price	 sum(total_price)
1001	\$250.00	\$250.00
1002	\$960.00	_
1002	\$240.00	\$1,200.00
••		
1007	\$250.00	
1007	\$126.00	
1007	\$240.00	<b>\$1,696.00</b>
1007	\$480.00	
1007	\$600.00	J
1008	\$840.00	
1008	\$100.00	J \$940.00

Select order\_num, sum(total\_price) from order\_detail group by order\_num;

14

#### GROUP BY column list clause

Select order\_num, sum(total\_price)

from order\_detail group by order\_num;

■ Example 2 (output) :

order_num	sum(total_price)
1001	250
1002	1200
1007	1696
1008	940

15

#### GROUP BY column list clause

■ Example 3 (GROUP BY used when joining tables):

Select o.order\_num, count(\*)

from orders o

inner join order\_detail d on o.order\_num = d.order\_num

where o.customer\_num = 110

group by o.order\_num;

■ This example is to count the number of items in each of the orders placed by customer 110

Output :	order_num	count (*)
	1008	2
	1015	1

16

#### GROUP BY - Additional Notes

All column names in SELECT list must appear in GROUP BY clause unless name is used only in an aggregate function, e.g.

```
SELECT
            a, b, sum(c)
 FROM
            t1
 GROUP BY a, b
                       ... correct
```

```
SELECT
           x, y, avg(z)
 FROM
 GROUP BY x
                      ... incorrect
```

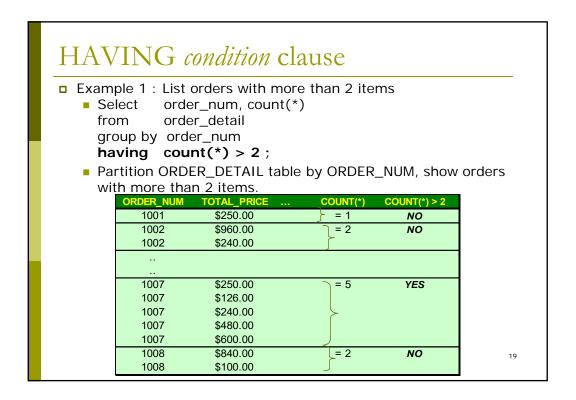
□ If WHERE is used with GROUP BY, WHERE is applied first, then groups are formed from remaining rows satisfying predicate.

#### HAVING condition clause

- □ To restrict the groups in the final result
- Compare with WHERE which filters individual rows:
  - The condition in "HAVING clause" always includes at least one aggregate function, otherwise the search condition could be moved to the WHERE clause and applied to individual rows.
  - Thus aggregate functions cannot be used in the WHERE clause.

18

Unit 8b : SQL (Advanced SELECT) AY2018/2019 S1



#### HAVING condition clause

■ Example 1 (output) :

order\_num count(\*)
1007 5

- Example 2 (HAVING can be used without GROUP BY clause)
  - Select avg(total\_price) average from order\_detail having count(\*) > 1;

20

#### Review

- Study the SQLs below, explain the differences. How many rows do you expect to find in the output list?
  - Select avg(unit\_price) from product;
  - Select avg(unit\_price) from product where prod\_num = '101';
  - Select prod\_num, avg(unit\_price) from product group by prod\_num;

21

### Questions for query formulation

- □ SELECT ..
  - What are the <u>output</u>? Base table/derived columns, aggregate values?
- □ FROM ...
  - Where to get the <u>output from</u>? Which table(s)?
  - Also include tables of which the columns are needed for specifying the WHERE conditions
- □ INNER JOIN.. ON
  - Include other tables needed in joining. One for each additional table.
  - What are the common columns in each table?
- □ WHERE ...
  - Specify the <u>filter conditions</u> on individual records
- GROUP BY ..
  - How do you want to <u>partition the records</u> in the table ?
- □ HAVING ..
  - How do you want to <u>select the groups</u>?
- ORDER BY ..
  - How do you want the output to be sorted?

22

# Subqueries in SQL

- Subqueries
  - Some SQL statements can have a SELECT embedded within them.
  - A subselect can be used in WHERE and HAVING clauses of an outer SELECT, where it is called a <u>subquery</u> or <u>nested query</u>.
  - Subselects may also appear in INSERT, UPDATE, and DELETEs.

23

### Subqueries in SQL

- Subquery with Equality
  - List the most expensive product.
    - Select prod\_num from product

where **unit\_price** = (select **max(unit\_price)** from product);

24

### Subqueries in SQL

- Subquery with Equality
  - List customers who reside in 'California'

Select fname, Iname from customer

where state\_code = (select state\_code

from state

where state\_name = 'California');

Note that this query can also be written using a join :

Select fname, Inamefrom customer cinner join state s

25

### Subqueries in SQL

- Subquery with Aggregate
  - List those employees with salary higher than the average salary of all employees

Select emp\_id from employee

where salary > (select AVG(salary) from employee);

- Cannot write 'WHERE salary > AVG(salary)'.
   (recall : cannot use aggregate functions in WHERE clause)
- Instead, use subquery to find the average salary, and then use outer SELECT to find those employees with salary greater than that.

26

#### Summary

- Advanced SELECT statement
  - Includes GROUP BY and HAVING clause in the basic statement.
  - Creates subqueries.
- The syntax to include all the clauses in the SELECT statement is:

FROM table list
{[INNER JOIN tablename ON condition]}
[WHERE condition]
[GROUP BY column list]
[HAVING condition]
[ORDER BY column list [DESC]]

27

### **QUIZ**

- □ Give an example of using GROUP BY clause.
- When do we need to use the HAVING clause?
- Which clause shall be used if the result displayed is arrange in descending?

28

Unit 8b : SQL (Advanced SELECT)

Page 14

AY2018/2019 S1