# Database Programming with SQL

### 9-1: Using GROUP BY and HAVING Clauses

- 1. In the SQL query shown below, which of the following is true about this query?
  - a. Kimberly Grant would not appear in the results set.
  - b. The GROUP BY clause has an error because the manager\_id is not listed in the SELECT clause.
  - c. Only salaries greater than 16001 will be in the result set.
  - d. Names beginning with Ki will appear after names beginning with Ko.
  - e. Last names such as King and Kochhar will be returned even if they don't have salaries > 16000.

SELECT last\_name, MAX(salary)
FROM employees WHERE last\_name LIKE 'K%'
GROUP BY manager\_id, last\_name
HAVING MAX(salary) > 16000
ORDER BY last\_name DESC;

#### None of the statements are True

- 2. Each of the following SQL queries has an error. Find the error and correct it. Use Oracle Application Express to verify that your corrections produce the desired results.
  - a. SELECT manager\_id FROM employees WHERE AVG(salary) < 93;</li>

SELECT manager\_id FROM employees GROUP BY manager\_id HAVING AVG(salary) < 93;

b. SELECT cd\_number, COUNT(title) FROM d\_cds WHERE cd number < 93;

SELECT cd\_number, COUNT(title) FROM d\_cds WHERE cd\_number < 93 GROUP BY cd\_number;

c. SELECT ID, MAX(ID), artist AS Artist FROM d\_songs WHERE duration IN('3 min', '6 min', '10 min') HAVING ID < 50 GROUP by ID;

SELECT MAX(ID) AS max\_id, artist AS Artist FROM d\_songs WHERE duration IN ('3 min', '6 min', '10 min') GROUP BY artist HAVING MAX(ID) < 50;

d. SELECT loc\_type, rental\_fee AS Fee FROM d\_venues WHERE id

SELECT loc\_type, rental\_fee AS Fee FROM d\_venues WHERE id IS NOT NULL;

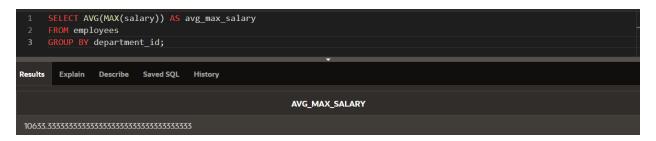
3. Rewrite the following query to accomplish the same result: SELECT DISTINCT MAX(song\_id) FROM d\_track\_listings WHERE track IN (1, 2, 3);



- 4. Indicate True or False
  - a. If you include a group function and any other individual columns in a SELECT clause, then each individual column must also appear in the GROUP BY clause. (**True**)
  - b. You can use a column alias in the GROUP BY clause. (False)
  - c. The GROUP BY clause always includes a group function. (False)
- 5. Write a query that will return both the maximum and minimum average salary grouped by department from the employees table.

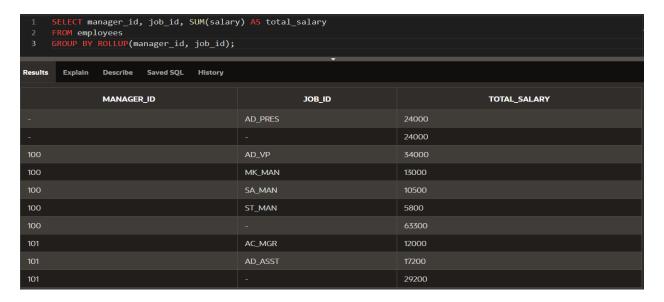


6. Write a query that will return the average of the maximum salaries in each department for the employees table.



## 9-2: Using ROLLUP and CUBE Operations and GROUPING SETS

- 1. Within the Employees table, each manager\_id is the manager of one or more employees who each have a job\_id and earn a salary. For each manager, what is the total salary earned by all of the employees within each job\_id? Write a query to display the Manager\_id, job\_id, and total salary. Include in the result the subtotal salary for each manager and a grand total of all salaries.
  - You can use the ROLLUP operation to get the total salary for each manager\_id and job\_id, along with the subtotals for each manager\_id and a grand total.



2. Amend the previous query to also include a subtotal salary for each job\_id regardless of the manager\_id.

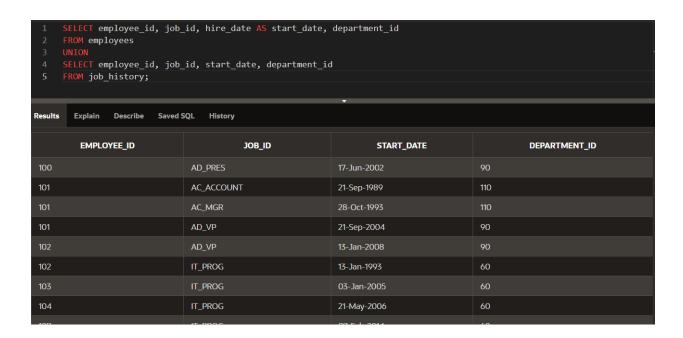


- 3. Using GROUPING SETS, write a query to show the following groupings:
  - department id, manager id, job id
  - manager id, job id
  - department id, manager id

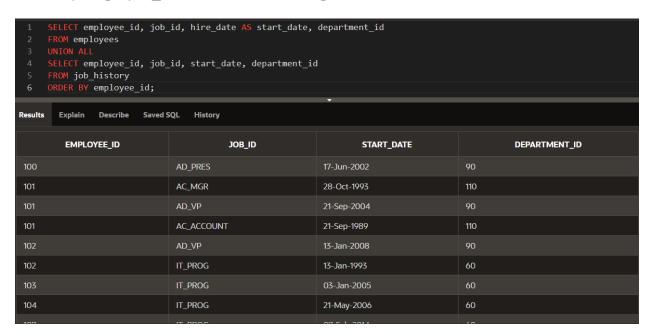
SELECT department_id, manager_id, job_id, SUM(salary) AS total_salary  FROM employees  GROUP BY GROUPING SETS (  (department_id, manager_id, job_id),  (manager_id, job_id),  (department_id, manager_id));  Results Explain Describe Saved SQL History					
DEPARTMENT_ID	MANAGER_ID	JOB_ID	TOTAL_SALARY		
90		AD_PRES	24000		
90	100	AD_VP	34000		
20	100	MK_MAN	13000		
80	100	SA_MAN	10500		
50	100	ST_MAN	5800		
110	101	AC_MGR	12000		
10	101	AD_ASST	17200		
60	102	IT_PROG	9000		

### 9-3: Set Operators

- 1. Name the different Set operators?
  - UNION: Combines results of two queries, removing duplicates.
  - UNION ALL: Combines results of two queries, including duplicates.
  - **INTERSECT**: Returns only the rows that are common between two queries.
  - MINUS: Returns the rows from the first query that are not present in the second query.
- 2. Write one query to return the employee\_id, job\_id, hire\_date, and department\_id of all employees and a second query listing employee\_id, job\_id, start\_date, and department\_id from the job\_history table and combine the results as one single output. Make sure you suppress duplicates in the output.



3. Amend the previous statement to not suppress duplicates and examine the output. How many extra rows did you get returned and which were they? Sort the output by employee\_id to make it easier to spot.



4. List all employees who have not changed jobs even once. (Such employees are not found in the job history table)



5. List the employees that HAVE changed their jobs at least once.



6. Using the UNION operator, write a query that displays the employee\_id, job\_id, and salary of ALL present and past employees. If a salary is not found, then just display a 0 (zero) in its place

