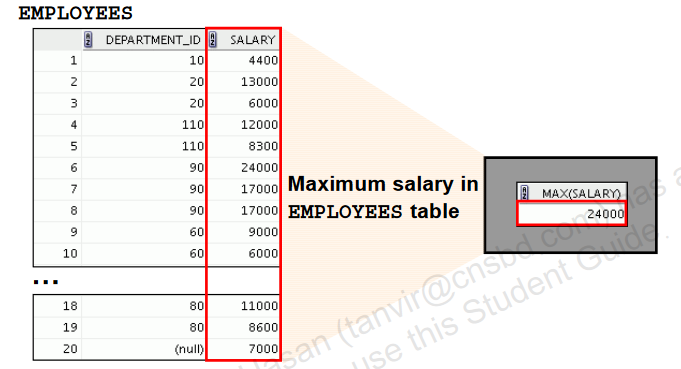
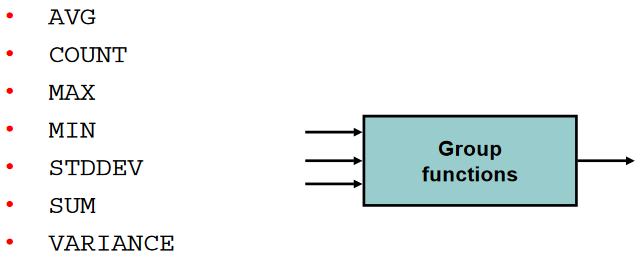
GROUP FUNCTION

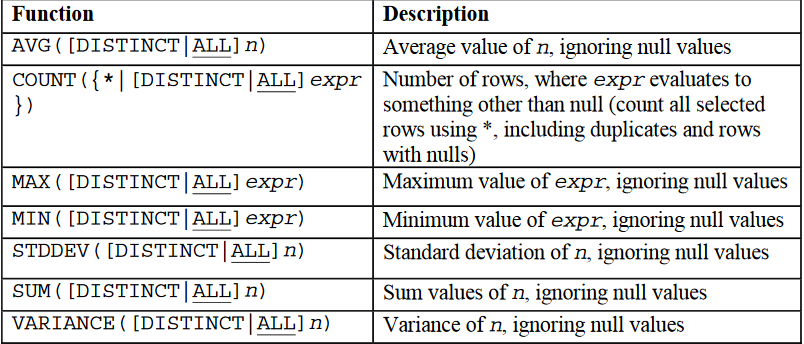
**What is group function ?**

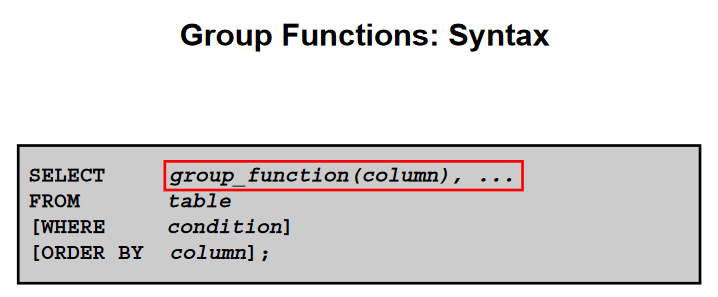
Group functions operate on sets of rows to give one result per group.



**Type of Group function**







**Using the AVG and SUM Functions**

* **You can use AVG and SUM for numeric data.**

|  |
| --- |
| Example:  **SELECT AVG(salary) "Average Salary"**  **FROM employees;**    **SELECT MAX(salary) "MAX Salary"**  **FROM employees;**    **SELECT MIN(salary) "Minimum Salary"**  **FROM employees;**    **SELECT SUM(salary) "SUM Salary"**  **FROM employees;**    **SELECT AVG(salary) "Average Salary", MAX(salary) "MAX Salary",**  **MIN(salary) "Minimum Salary", SUM(salary) "SUM Salary"**  **FROM employees**  **WHERE job\_id LIKE '%REP%';** |

|  |
| --- |
| **Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number.**  **SELECT MAX(salary), MIN(salary), SUM(salary), AVG(salary)**  **FROM employees;** |

**Using the MIN and MAX Functions**

* You can use MIN and MAX for numeric, character, and date data types.

|  |
| --- |
| **SELECT MIN(hire\_date), MAX(hire\_date)**  **FROM employees;**    **SELECT MIN(last\_name), MAX(last\_name)**  **FROM employees;** |

**Using the COUNT Function**

|  |
| --- |
| * COUNT(\*) returns the number of rows in a table:   **SELECT COUNT(\*)**  **FROM employees**  **WHERE department\_id = 50;**     * COUNT(expr) returns the number of rows with non-null values for expr:   **SELECT COUNT(commission\_pct)**  **FROM employees;**    **SELECT COUNT(commission\_pct)**  **FROM employees**  **WHERE department\_id = 80;** |

**Using the DISTINCT Keyword**

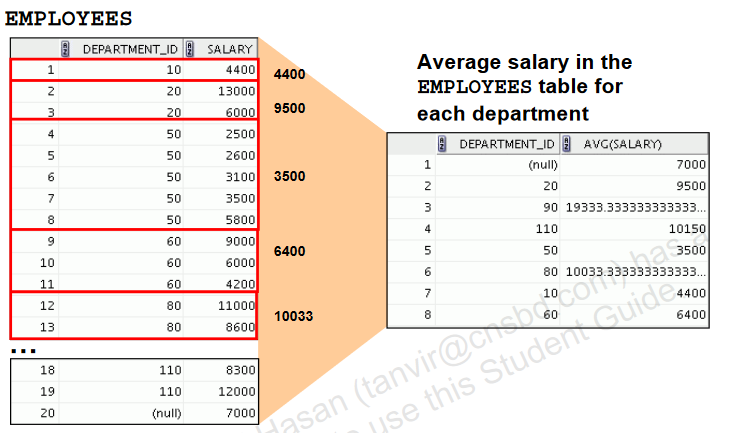
|  |
| --- |
| * COUNT(DISTINCT expr) returns the number of distinct non-null values of expr. * To display the number of distinct department values in the EMPLOYEES table:   **SELECT COUNT(DISTINCT department\_id)**  **FROM employees;** |

**Group Functions and Null Values**

|  |
| --- |
| * Group functions ignore null values in the column:   **SELECT AVG(commission\_pct)**  **FROM employees;**     * The NVL function forces group functions to include null values:   **SELECT AVG(NVL(commission\_pct, 0))**  **FROM employees;** |

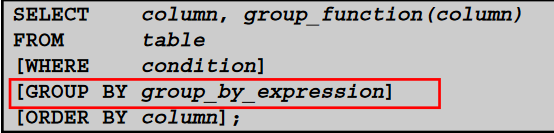
|  |
| --- |
| * **Determine the number of managers without listing them. Label the column Number of Managers.**   **Hint: Use the MANAGER\_ID column to determine the number of managers.**  **SELECT COUNT(DISTINCT manager\_id) "Number of Managers"**  **FROM employees;**     * **Find the difference between the highest and lowest salaries. Label the column DIFFERENCE**   **SELECT MAX(salary) - MIN(salary) DIFFERENCE**  **FROM employees;**     * Create a query that will display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998. Create appropriate column headings.   **SELECT COUNT(\*) total,**  **SUM(DECODE(TO\_CHAR(hire\_date,**  **'YYYY'),1995,1,0))"1995",**  **SUM(DECODE(TO\_CHAR(hire\_date,**  **'YYYY'),1996,1,0))"1996",**  **SUM(DECODE(TO\_CHAR(hire\_date,**  **'YYYY'),1997,1,0))"1997",**  **SUM(DECODE(TO\_CHAR(hire\_date, 'YYYY'),1998,1,0))"1998"**  **FROM employees;** |

**Creating Groups of Data**

****

**Creating Groups of Data: GROUP BY Clause Syntax**

* You can divide rows in a table into smaller groups by using the GROUP BY clause.

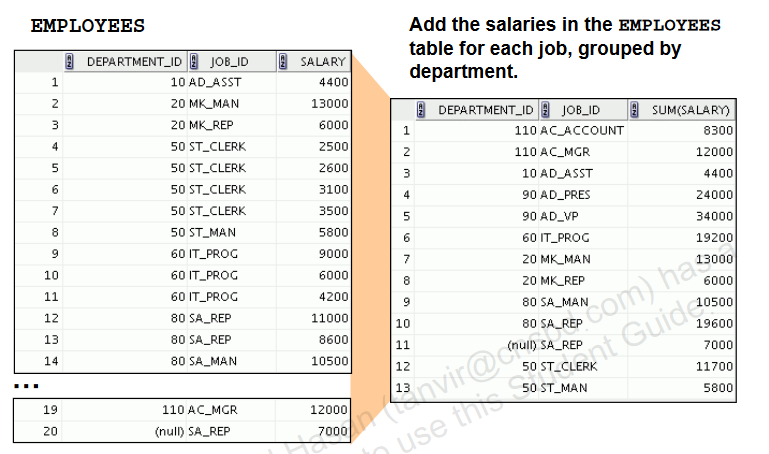
****

**Using the GROUP BY Clause**

|  |
| --- |
| * All the columns in the SELECT list that are not in group functions must be in the GROUP BY clause.   **SELECT department\_id, AVG(salary)**  **FROM employees**  **GROUP BY department\_id ;**     * The GROUP BY column does not have to be in the SELECT list.   **SELECT AVG(salary)**  **FROM employees**  **GROUP BY department\_id ;**     * Also use the group function in the ORDER BY clause:   **SELECT department\_id, AVG(salary)**  **FROM employees**  **GROUP BY department\_id**  **ORDER BY AVG(salary);** |

|  |
| --- |
| * **To dispaly the maximum, sum, and average salary for each job type.**   **SELECT job\_id, MAX(salary) "Maximum",**  **MIN(salary) "Minumum",**  **SUM(salary) "Sum",**  **AVG(salary) "Average"**  **FROM employees**  **GROUP BY job\_id;**     * Write a query to display the number of people with the same job.   **SELECT job\_id, COUNT(\*)**  **FROM employees**  **GROUP BY job\_id;**     * Create a matrix query to display the job, the salary for that job based on the department number, and the total salary for that job, for departments 20, 50, 80, and 90, giving each column an appropriate heading.   **SELECT job\_id "Job",**  **SUM(DECODE(department\_id , 20, salary)) "Dept 20",**  **SUM(DECODE(department\_id , 50, salary)) "Dept 50",**  **SUM(DECODE(department\_id , 80, salary)) "Dept 80",**  **SUM(DECODE(department\_id , 90, salary)) "Dept 90",**  **SUM(salary) "Total"**  **FROM employees**  **GROUP BY job\_id;** |

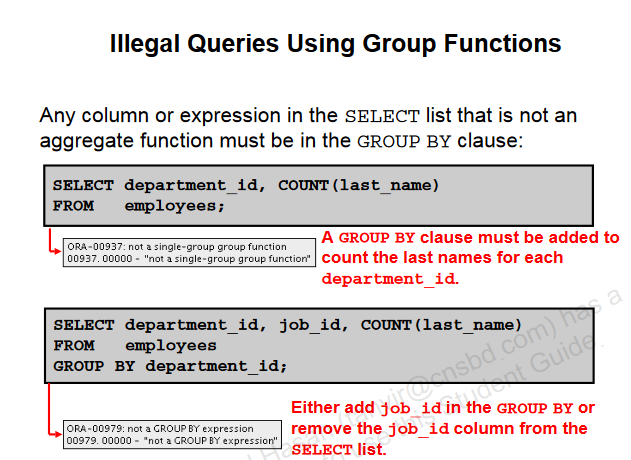
**Grouping by More Than One Column**



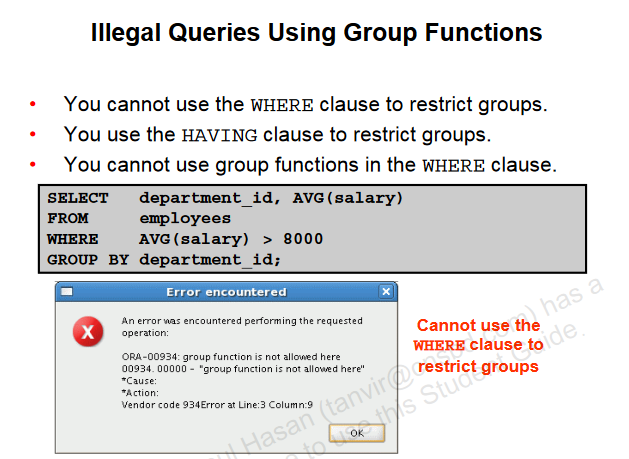
|  |
| --- |
| **SELECT department\_id , job\_id, SUM(salary)**  **FROM employees**  **GROUP BY department\_id, job\_id**  **ORDER BY job\_id;** |

**Using the GROUP BY Clause on Multiple Columns**

|  |
| --- |
| **SELECT department\_id , job\_id, SUM(salary)**  **FROM employees**  **WHERE department\_id > 40**  **GROUP BY department\_id, job\_id**  **ORDER BY job\_id;** |

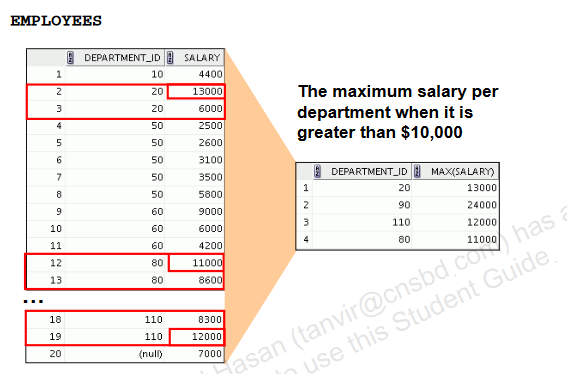


|  |
| --- |
| * From 1 solve   **SELECT department\_id, COUNT(last\_name)**  **FROM employees**  **GROUP BY department\_id;**     * From 2 solve   **SELECT department\_id,job\_id, COUNT(last\_name)**  **FROM employees**  **GROUP BY department\_id,job\_id;** |



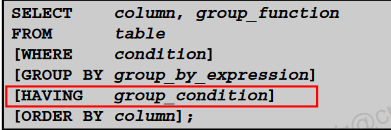
|  |
| --- |
| * SOLVE   **SELECT department\_id, AVG(salary)**  **FROM employees**  **GROUP BY department\_id**  **HAVING AVG(salary) > 8000;** |

**Restricting Group Results**



**Restricting Group Results with the HAVING Clause**

* Rows are grouped.
* The group function is applied.
* Groups matching the HAVING clause are displayed.



**Using the HAVING Clause**

|  |
| --- |
| **SELECT department\_id, MAX(salary)**  **FROM employees**  **GROUP BY department\_id**  **HAVING MAX(salary) > 10000;**    **SELECT department\_id, AVG(salary)**  **FROM employees**  **GROUP BY department\_id**  **HAVING MAX(salary) > 10000;**    **SELECT job\_id, SUM(salary) PAYROLL**  **FROM employees**  **WHERE job\_id NOT LIKE '%REP%'**  **GROUP BY job\_id**  **HAVING SUM(salary) > 13000**  **ORDER BY SUM(salary);** |

|  |
| --- |
| * **Create a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is $6,000 or less. Sort the output in descending order of salary.**   **SELECT manager\_id, MIN(salary)**  **FROM employees**  **WHERE manager\_id IS NOT NULL**  **GROUP BY manager\_id**  **HAVING MIN(salary) > 6000**  **ORDER BY MIN(salary) DESC;** |

**Nesting Group Functions**

|  |
| --- |
| **SELECT AVG(salary)**  **FROM employees**  **GROUP BY department\_id;**    **SELECT MAX(salary)**  **FROM employees**  **GROUP BY department\_id;**    **SELECT MAX(AVG(salary))**  **FROM employees**  **GROUP BY department\_id;** |