**SUMMARISING DATA**

**PRACTICE QUESTIONS**

ESSENTIAL = 1-7               VERY USEFUL =  8

Sometimes we need to get summary information, rather than single rows from the database.    To do this we need to use aggregate functions and depending on our requirements, the GROUP BY and HAVING clauses.

1.     SELECT SUM(monthly\_sal) AS 'Total of all monthly salaries',

AVG(monthly\_sal) AS 'Average monthly salary

FROM emp;

|  |  |
| --- | --- |
| **Total of all monthly salaries** | **Average monthly salary** |
| 29025 | 2073.21429 |

2.     SELECT MIN(monthly\_sal) AS 'Minimum monthly salary',

MAX(monthly\_sal) AS 'Maximum monthly salary',

COUNT(empno) AS 'Number of employees'

FROM emp;

|  |  |  |
| --- | --- | --- |
| **Minimum monthly salary** | **Maximum monthly salary** | **Number of employees** |
| 800 | 5000 | 14 |

  In questions 1 and 2, we wanted to summarise all rows (the highest level of summarisation) so we didn’t use a GROUP BY clause. However, for the remaining questions, we want to increase the granularity to produce subtotals for particular column(s) so a GROUP BY clause is required.

3.        SELECT deptno, SUM(monthly\_sal) AS 'Total of all monthly salaries',

AVG(monthly\_sal) AS 'Average monthly salary'

FROM emp

GROUP BY deptno;

|  |  |  |
| --- | --- | --- |
| **DEPTNO** | **Total of all monthly salaries** | **Average monthly salary** |
| 10 | 8750 | 2916.66667 |
| 20 | 10875 | 2175 |
| 30 | 9400 | 1566.66667 |

  4. SELECT job, deptno,

SUM(monthly\_sal) AS 'Total of all monthly salaries',

AVG(monthly\_sal) AS 'Average monthly salary'

FROM emp

GROUP BY job, deptno;

|  |  |  |  |
| --- | --- | --- | --- |
| **JOB** | **DEPTNO** | **Total of all monthly salaries** | **Average monthly salary** |
| ANALYST | 20 | 6000 | 3000 |
| CLERK | 10 | 1300 | 1300 |
| CLERK | 20 | 1900 | 950 |
| CLERK | 30 | 950 | 950 |
| MANAGER | 10 | 2450 | 2450 |
| MANAGER | 20 | 2975 | 2975 |
| MANAGER | 30 | 2850 | 2850 |
| PRESIDENT | 10 | 5000 | 5000 |
| SALESMAN | 30 | 5600 | 1400 |

Note: The more columns in the GROUP BY clause, the higher the level of granularity of the subtotals and the lower-level of summarisation.  Results are summarised by department within each job. Only valid groups are shown e.g. there are no Analysts in departments 10 or 30 so no results are shown.

5.        SELECT d.deptno, d.dname,

SUM(e.monthly\_sal) AS 'Total of all monthly salaries',

AVG(e.monthly\_sal) AS 'Average monthly salary'

FROM emp e

INNER JOIN   dept d ON e.deptno = d.deptno

GROUP BY d.deptno, d.dname;

|  |  |  |  |
| --- | --- | --- | --- |
| **DEPTNO** | **DNAME** | **Total of all monthly salaries** | **Average monthly salary** |
| 10 | ACCOUNTING | 8750 | 2916.66667 |
| 20 | RESEARCH | 10875 | 2175 |
| 30 | SALES | 9400 | 1566.66667 |

6.        Aggregate functions can also be used to compare values across multiple tables.   For example:  to compare the no. of items ordered with the total number of items delivered (perhaps in more than one delivery)

When looking at joins a few weeks’ ago, we used the following query:

SELECT i.ordid, i.itemid, i.qty, di.qty

FROM item i

INNER JOIN delivered\_item di ON i.ordid  = di.ordid

        AND i.itemid = di.itemid

WHERE i.qty > di.qty

AND i.ordid    < 615;

|  |  |  |  |
| --- | --- | --- | --- |
| **ORDID** | **ITEMID** | **QTY** | **QTY** |
| 612 | 1 | 100 | 75 |
| 612 | 1 | 100 | 25 |
| 612 | 3 | 150 | 100 |
| 612 | 3 | 150 | 50 |
| 614 | 1 | 444 | 400 |
| 614 | 1 | 444 | 44 |

which showed that each of the items was delivered in two separate batches.

If we use a GROUP BY, we can summarise the delivery quantities for the same order items in one row.

SELECT i.ordid, i.itemid, i.qty, **SUM(di.qty)** AS 'Total delivered'

FROM item i

INNER JOIN delivered\_item di ON i.ordid  = di.ordid

AND i.itemid = di.itemid

WHERE i.qty > di.qty

AND i.ordid < 615

**GROUP BY i.ordid, i.itemid, i.qty;**

|  |  |  |  |
| --- | --- | --- | --- |
| **ORDID** | **ITEMID** | **QTY** | **Total delivered** |
| 612 | 1 | 100 | 100 |
| 612 | 3 | 150 | 150 |
| 614 | 1 | 444 | 444 |

7.         The HAVING clause is used to show only those summarised results that meet certain criteria. For example, if we want only want to display those job / department combinations which have 2 or more employees, we need to add a HAVING clause:

SELECT job, deptno, COUNT(empno) AS 'No. of employees',

SUM(monthly\_sal) AS 'Total of all monthly salaries',

AVG(monthly\_sal) AS 'Average monthly salary'

FROM emp

GROUP BY job, deptno

**HAVING COUNT(empno) > 1;**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DEPTNO** | **JOB** | **No. of employees** | **Total of all monthly salaries** | **Average monthly salary** |
| 20 | ANALYST | 2 | 6000 | 3000 |
| 20 | CLERK | 2 | 1900 | 950 |
| 30 | SALESMAN | 4 | 5600 | 1400 |

8. Take this initial query:

SELECT job, AVG(monthly\_sal) AS 'Average monthly salary',

SUM(monthly\_sal) AS 'Total of all monthly salaries',

COUNT(empno) AS 'No. of employees'

FROM emp

GROUP BY job;

|  |  |  |  |
| --- | --- | --- | --- |
| **JOB** | **Average monthly salary** | **Total of all monthly salaries** | **No. of employees** |
| ANALYST | 3000 | 6000 | 2 |
| CLERK | 1037.5 | 4150 | 4 |
| MANAGER | 2758.33333 | 8275 | 3 |
| PRESIDENT | 5000 | 5000 | 1 |
| SALESMAN | 1400 | 5600 | 4 |

To limit the rows that are included before the calculation of the summarised results, use a WHERE clause as follows:

SELECT job, AVG(monthly\_sal) AS 'Average monthly salary',

SUM(monthly\_sal) AS 'Total of all monthly salaries',

COUNT(empno) AS 'No. of employees'

FROM emp

**WHERE deptno != 20**

GROUP BY   job;

|  |  |  |  |
| --- | --- | --- | --- |
| **JOB** | **Average monthly salary** | **Total of all monthly salaries** | **No. of employees** |
| CLERK | 1125 | 2250 | 2 |
| MANAGER | 2650 | 5300 | 2 |
| PRESIDENT | 5000 | 5000 | 1 |
| SALESMAN | 1400 | 5600 | 4 |

As department 20 has 2 Analysts, 2 Clerks and 1 Manager, adding the WHERE clause removes these employees from the calculations, changing the result for those jobs but the other jobs not in department 20 are unchanged.

The addition of a HAVING clause will not change any grouped results but will cause some of them to be excluded.

SELECT job, AVG(monthly\_sal) AS 'Average monthly salary',

SUM(monthly\_sal) AS 'Total of all monthly salaries',

COUNT(empno) AS 'No. of employees'

FROM emp

WHERE deptno != 20

GROUP BY job

**HAVING COUNT(empno) > 2**;

|  |  |  |  |
| --- | --- | --- | --- |
| **JOB** | **Average monthly salary** | **Total of all monthly salaries** | **No. of employees** |
| SALESMAN | 1400 | 5600 | 4 |