Problem 1:

Create a query displaying the department_name, minimum salary, maximum salary, average salary for employees by department_name. Round the average salary to two decimal places. Order the query by department name. Label the columns Min Salary, Max Salary, Average Salary.

SQL script:

```
SELECT d.department_name, MIN(e.salary) "Min Salary", MAX(e.salary) "Max Salary”,
ROUND(AVG(e.salary), 2) "Average Salary"
FROM departments d
INNER JOIN employees e USING (department_id)
GROUP BY department_name
ORDER BY department_name;
```

Console Output:

```
SQL> SELECT d.department_name, MIN(e.salary) "Min Salary", MAX(e.salary) "Max Salary", ROUND(AVG(e.salary), 2) "Average Salary"
  2  FROM departments d
  3  INNER JOIN employees e USING (department_id)
  4  GROUP BY department_name
  5* ORDER BY department_name;
DEPARTMENT_NAME               Min Salary Max Salary Average Salary
------------------------------ ---------- ---------- --------------
Accounting                           8300      12000          10150
Administration                       4400       4400           4400
Executive                           17000      24000       19333.33
Finance                              6900      12000           8600
Human Resources                      6500       6500           6500
IT                                   4200       9000           5760
Marketing                            6000      13000           9500
Public Relations                    10000      10000          10000
Purchasing                           2500      11000           4150
Sales                                6100      14000        8955.88
Shipping                             2100       8200        3475.56

11 rows selected.
```

Problem 2:

Write a query to show job title, average salary, difference between maximum and minimum of salary of employees for sales people only (those having a commission or job_id starts with SA). Order the query by job title. Use the column names “Average Salary” and “Diff Max – Min”.

SQL script:

```
SELECT j.job_title, AVG(e.salary) "Average Salary", MAX(e.salary)-MIN(e.salary) "Diff Max-Min"
FROM employees e
INNER JOIN jobs j USING (job_id)
```

WHERE SUBSTR(job_id, 1, 2) = 'SA'

GROUP BY j.job_title

ORDER BY j.job_title;

Console Output:

```
SQL> SELECT j.job_title, AVG(e.salary) "Average Salary", MAX(e.salary)-MIN(e.salary) "Diff Max-Min"
  2  FROM employees e
  3  INNER JOIN jobs j USING (job_id)
  4  WHERE SUBSTR(job_id, 1, 2) = 'SA'
  5  GROUP BY j.job_title
  6  ORDER BY j.job_title;

JOB_TITLE                          Average Salary Diff Max-Min
---------------------------------- -------------- ------------
Sales Manager                               12200         3500
Sales Representative                         8350         5400
```

Problem 3:

Create a query showing all employees (employee_id, Last_name, Salary) that make more money than the maximum salary of all sales people. (those having a commission or job_id starts with SA). Sort it by last_name.

SQL script:

SELECT employee_id, last_name, salary

FROM employees

WHERE salary > (

SELECT MAX(salary)

FROM employees

WHERE SUBSTR(job_id, 1, 2) = 'SA')

ORDER BY last_name;

Console Output:

```
SQL> SELECT employee_id, last_name, salary
  2  FROM employees
  3  WHERE salary > (
  4  SELECT MAX(salary)
  5  FROM employees
  6  WHERE SUBSTR(job_id, 1, 2) = 'SA'
  7  )
  8* ORDER BY last_name;
EMPLOYEE_ID LAST_NAME                     SALARY
----------- ------------------------- ----------
        102 De Haan                        17000
        100 King                           24000
        101 Kochhar                        17000
```

Problem 4:

Create a query counting the number of locations by region name and country name. Alias the count column with # Locs. Implement the following custom sort order for region name: Europe, Americas, Asia. Within each region name, sort it by country name.

SQL script:

```
SELECT COUNT(l.location_id) "#Locs", r.region_name, c.country_name
FROM locations l
INNER JOIN countries c USING (country_id)
INNER JOIN regions r USING (region_id)
GROUP BY r.region_name, c.country_name
ORDER BY CASE
WHEN r.region_name='Europe' THEN 1
WHEN r.region_name='Americas' THEN 2
ELSE 3 END,
c.country_name;
```

Console Output:

```
SQL> SELECT COUNT(l.location_id) "#Locs", r.region_name, c.country_name
  2  FROM locations l
  3  INNER JOIN countries c USING (country_id)
  4  INNER JOIN regions r USING (region_id)
  5  GROUP BY r.region_name, c.country_name
  6  ORDER BY CASE
  7  WHEN r.region_name='Europe' THEN 1
  8  WHEN r.region_name='Americas' THEN 2
  9  ELSE 3 END,
 10* c.country_name;
     #Locs REGION_NAME               COUNTRY_NAME
---------- ------------------------ ----------------------------------------
         1 Europe                   Germany
         2 Europe                   Italy
         1 Europe                   Netherlands
         2 Europe                   Switzerland
         3 Europe                   United Kingdom
         1 Americas                 Brazil
         2 Americas                 Canada
         1 Americas                 Mexico
         4 Americas                 United States of America
         1 Asia                     Australia
         1 Asia                     China
         1 Asia                     India
         2 Asia                     Japan
         1 Asia                     Singapore

14 rows selected.
```

Problem 5:

Create a query counting the number of employees by administration, sales, or other. Any job id starting with IT, AD, AC, or PU is considered administration, any job id starting with ST, SA, SH is considered sales. For any other job id use other. Use the column headings “Job Category” and “Count of Emp”. Sort it by job category expression.

SQL script:

```
SELECT COUNT(employee_id) "Count of Emp",
CASE
WHEN SUBSTR(job_id, 1, 2) IN ('IT', 'AD', 'AC', 'PU') THEN 'administration'
WHEN SUBSTR(job_id, 1, 2) IN ('ST', 'SA', 'SH') THEN 'sales'
ELSE 'other' END "Job Category"
FROM employees
GROUP BY CASE
WHEN SUBSTR(job_id, 1, 2) IN ('IT', 'AD', 'AC', 'PU') THEN 'administration'
```

WHEN SUBSTR(job_id, 1, 2) IN ('ST', 'SA', 'SH') THEN 'sales'

ELSE 'other' END

ORDER BY "Job Category";

Console Output:

```
SQL> SELECT COUNT(employee_id) "Count of Emp",
  2  CASE
  3  WHEN SUBSTR(job_id, 1, 2) IN ('IT', 'AD', 'AC', 'PU') THEN 'administration'
  4  WHEN SUBSTR(job_id, 1, 2) IN ('ST', 'SA', 'SH') THEN 'sales'
  5  ELSE 'other' END "Job Category"
  6  FROM employees
  7  GROUP BY CASE
  8  WHEN SUBSTR(job_id, 1, 2) IN ('IT', 'AD', 'AC', 'PU') THEN 'administration'
  9  WHEN SUBSTR(job_id, 1, 2) IN ('ST', 'SA', 'SH') THEN 'sales'
 10  ELSE 'other' END
 11* ORDER BY "Job Category";
Count of Emp Job Category
------------ --------------
          17 administration
          10 other
          80 sales
```

Problem 6:

Show the count of employees by department_id for employees that are not sales people. Display only those records where the count is greater than 5. Order it by department_id.

SQL script:

SELECT COUNT(employee_id) "Count of Emp",

department_id

FROM employees

WHERE SUBSTR(job_id, 1, 2) NOT IN ('ST', 'SA', 'SH')

GROUP BY department_id

HAVING COUNT(employee_id) > 5

ORDER BY department_id;

Console Output:

```
SQL> SELECT COUNT(employee_id) "Count of Emp",
  2  department_id
  3  FROM employees
  4  WHERE SUBSTR(job_id, 1, 2) NOT IN ('ST', 'SA', 'SH')
  5  GROUP BY department_id
  6  HAVING COUNT(employee_id) > 5
  7* ORDER BY department_id;
Count of Emp DEPARTMENT_ID
------------ -------------
           6            30
           6           100
```

Problem 7:

Display the hire year and the count of employees hired in each year. Additionally, show the total number of employees (use analytic function sum) and alias it with sum_total. Extend this query to show the percentage of employees hired in each year (use the count, multiply it by 100, then divide by sum_total expression). Round this value to two decimal places, format the number so that always two decimal digits are displayed, add the percent sign at the end (2.8 → 2.80%) and finally, alias it with sum_percent. Sort the output by hire year.

SQL script:

```
SELECT TO_CHAR(hire_date, 'YYYY') "Hire Year",
COUNT(employee_id) "Count of Emp",
SUM(COUNT(employee_id)) OVER() sum_total,
ROUND(COUNT(employee_id) * 100 / (SUM(COUNT(employee_id)) OVER()), 2) || '%' sum_percent
FROM employees
GROUP BY TO_CHAR(hire_date, 'YYYY')
ORDER BY "Hire Year";
```

Console Output:

```
SQL> SELECT TO_CHAR(hire_date, 'YYYY') "Hire Year",
  2  COUNT(employee_id) "Count of Emp",
  3  SUM(COUNT(employee_id)) OVER() sum_total,
  4  ROUND(COUNT(employee_id) * 100 / (SUM(COUNT(employee_id)) OVER()), 2) || '%' sum_percent
  5  FROM employees
  6  GROUP BY TO_CHAR(hire_date, 'YYYY')
  7* ORDER BY "Hire Year";
Hire Count of Emp  SUM_TOTAL SUM_PERCENT
---- ------------ ---------- ----------------------------------------
1987            2        107 1.87%
1989            1        107 .93%
1990            1        107 .93%
1991            1        107 .93%
1993            1        107 .93%
1994            7        107 6.54%
1995            4        107 3.74%
1996           10        107 9.35%
1997           28        107 26.17%
1998           23        107 21.5%
1999           18        107 16.82%
2000           11        107 10.28%

12 rows selected.
```