

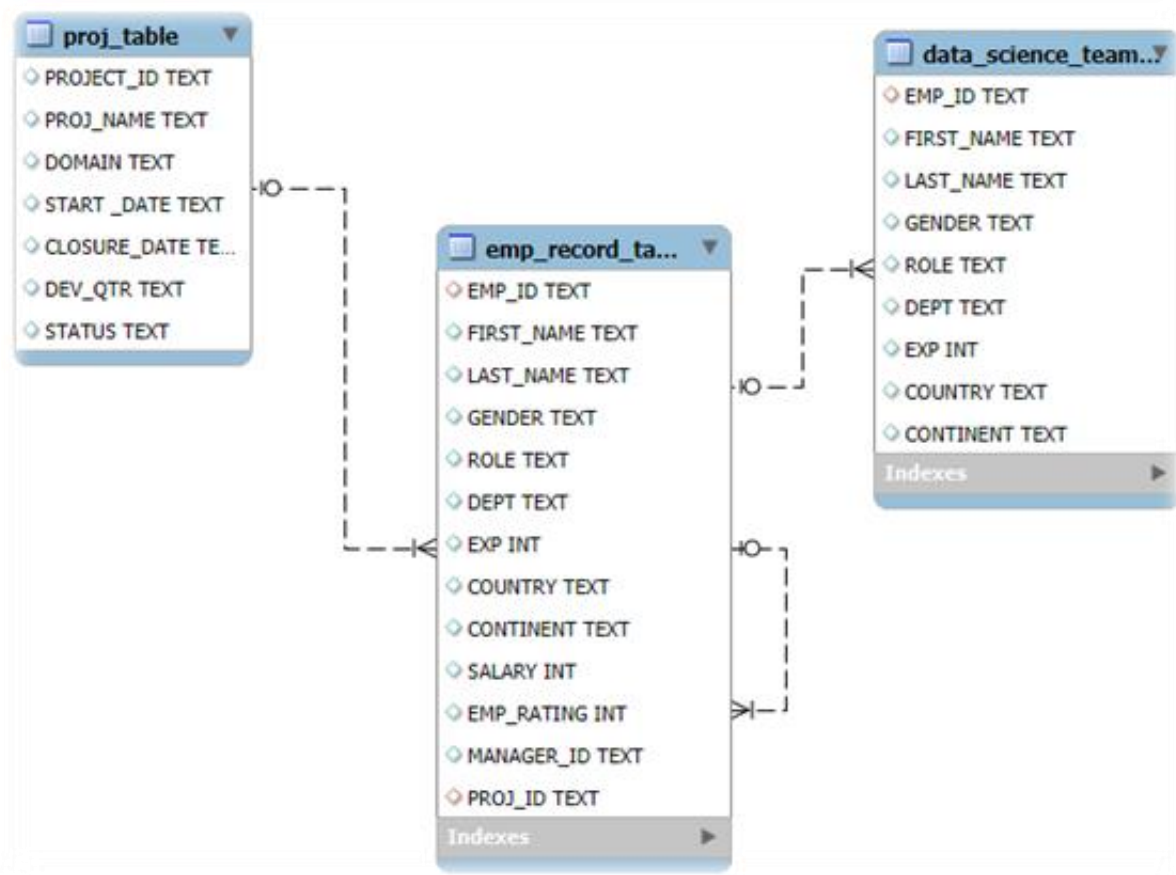
## Project Query and Attachment

Q1. Create a database named employee, then import data\_science\_team.csv  
proj\_table.csv and emp\_record\_table.csv into the employee database from the given  
resources.

**Query:**

```
CREATE database employee;
```

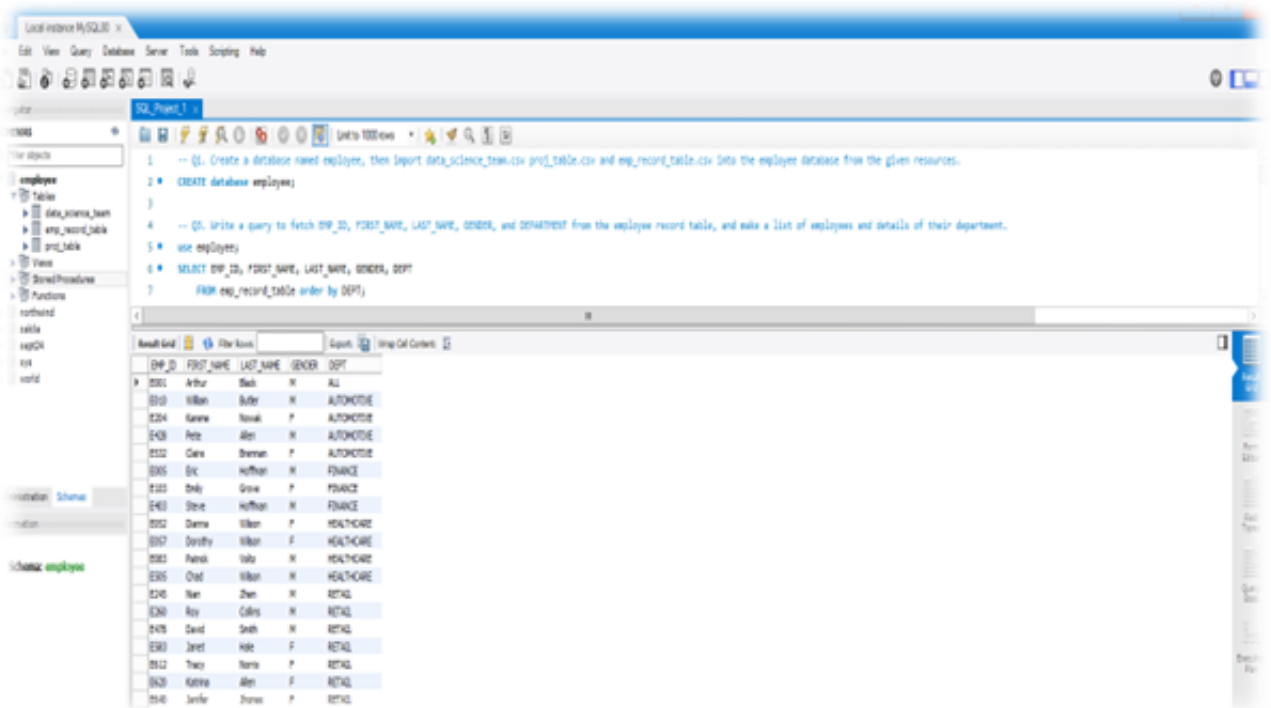
Q2. Create an ER diagram for the given **employee** database.



Q3. Write a query to fetch EMP\_ID, FIRST\_NAME, LAST\_NAME, GENDER, and DEPARTMENT from the employee record table, and make a list of employees and details of their department.

**Query:**

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT
FROM emp_record_table order by DEPT;
```



The screenshot shows a MySQL IDE window titled 'Local instance MySQL80'. The 'SQL\_Prompt' tab is active, displaying a query with seven lines. The query creates a database named 'employee', imports two CSV files, and then runs a SELECT statement to fetch employee details from the 'emp\_record\_table' ordered by department. The 'Results Grid' at the bottom shows the output of the query, listing 20 employees with their IDs, first names, last names, genders, and departments. The departments are grouped: ALL, AUTOMOTIVE, FINANCE, HEALTHCARE, and RETAIL.

EMP_ID	FIRST_NAME	LAST_NAME	GENDER	DEPT
E001	Arthur	Black	M	ALL
E010	William	Butler	M	AUTOMOTIVE
E204	Karen	Nouak	F	AUTOMOTIVE
E408	Pete	Allen	M	AUTOMOTIVE
E552	Cara	Brennan	F	AUTOMOTIVE
E005	Eric	Hoffman	M	FINANCE
E101	Emily	Grove	F	FINANCE
E460	Steve	Hoffman	M	FINANCE
E052	Germa	Wilson	F	HEALTHCARE
E057	Jordyfy	Wilson	F	HEALTHCARE
E082	Patrick	Vala	M	HEALTHCARE
E305	Chad	Wilson	M	HEALTHCARE
E240	Nan	Zhen	M	RETAIL
E260	Ary	Colins	M	RETAIL
E478	David	Smith	M	RETAIL
E380	Jared	Hale	F	RETAIL
E012	Tracy	Norris	F	RETAIL
E620	Kathie	Allen	F	RETAIL
E040	Jessie	Thomas	F	RETAIL

Q4. 4. Write a query to fetch EMP\_ID, FIRST\_NAME, LAST\_NAME, GENDER, DEPARTMENT, and EMP\_RATING if the EMP\_RATING is:

- less than two
- greater than four
- between two and four

**Query:**

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT,  
EMP_RATING,
```

case

when EMP\_RATING < 2 then "Less than 2"

when EMP\_RATING between 2 and 4 then "Between two and four"

else 'Greater than Four'

end as Rating

```
FROM emp_record_table;
```

The screenshot shows a SQL IDE interface. The top pane displays a query that fetches employee details and categorizes them based on their rating. The bottom pane shows the resulting data grid.

**Query:**

```
1 FROM emp_record_table order by DEPT;  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

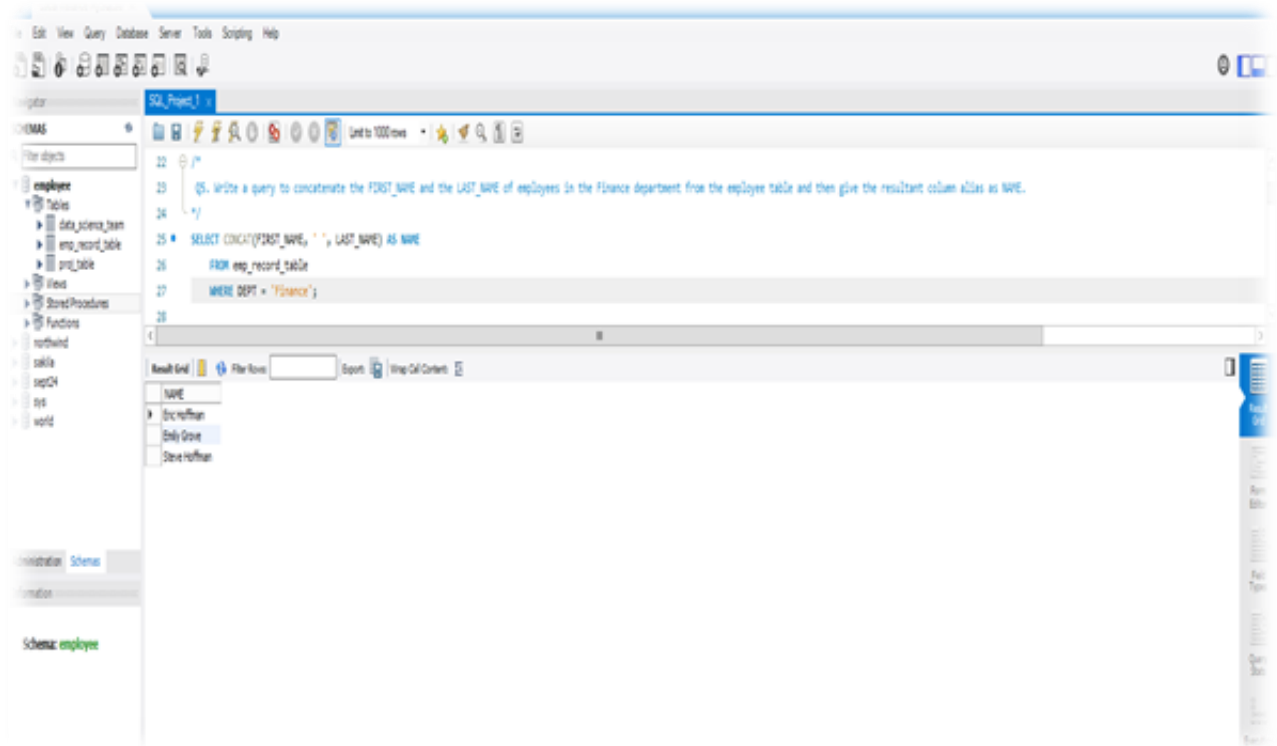
**Result Grid:**

EMP_ID	FIRST_NAME	LAST_NAME	GENDER	DEPT	EMP_RATING	Rating
1001	Arthur	Bulk	M	ACC	5	Greater than Four
1002	Dhr	Hoffman	M	FINANC	5	Between two and four
1003	William	Buller	M	AUTHORITY	2	Between two and four
1004	Diana	Wilson	F	HEALTHCARE	5	Greater than Four
1005	David	Wilson	F	HEALTHCARE	1	Less than 2
1006	Patricia	Wife	M	HEALTHCARE	5	Greater than Four
1007	Emily	Grove	F	FINANC	4	Between two and four
1008	Karen	Novak	F	AUTHORITY	5	Greater than Four
1009	Nan	Zhu	M	RETAIL	2	Between two and four
1010	Roy	Colts	M	RETAIL	3	Between two and four
1011	Steve	Hoffman	M	FINANC	3	Between two and four
1012	Pete	Allen	M	AUTHORITY	4	Between two and four
1013	David	Smith	M	RETAIL	4	Between two and four
1014	Chad	Wilson	M	HEALTHCARE	2	Between two and four
1015	Clara	Brenner	F	AUTHORITY	1	Less than 2
1016	Janet	Hale	F	RETAIL	2	Between two and four
1017	Tracy	Harris	F	RETAIL	4	Between two and four
1018	Kathleen	Allen	F	RETAIL	1	Less than 2
1019	Barbara	Harris	F	RETAIL	4	Between two and four

Q5. Write a query to concatenate the FIRST\_NAME and the LAST\_NAME of employees in the Finance department from the employee table and then give the resultant column alias as NAME.

**Query:**

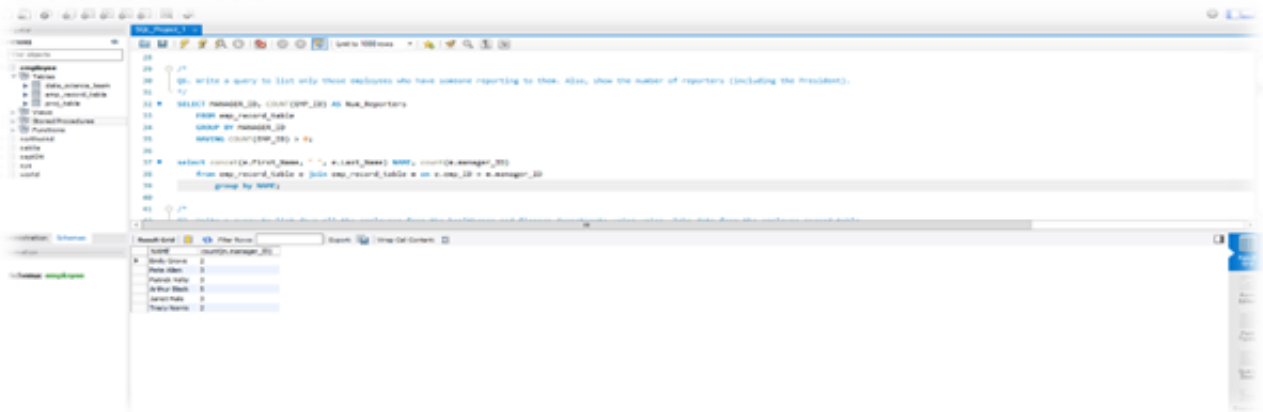
```
SELECT CONCAT(FIRST_NAME, ' ', LAST_NAME) AS NAME
FROM emp_record_table
WHERE DEPT = 'Finance';
```



Q6. Write a query to list only those employees who have someone reporting to them. Also, show the number of reporters (including the President)

**Query:**

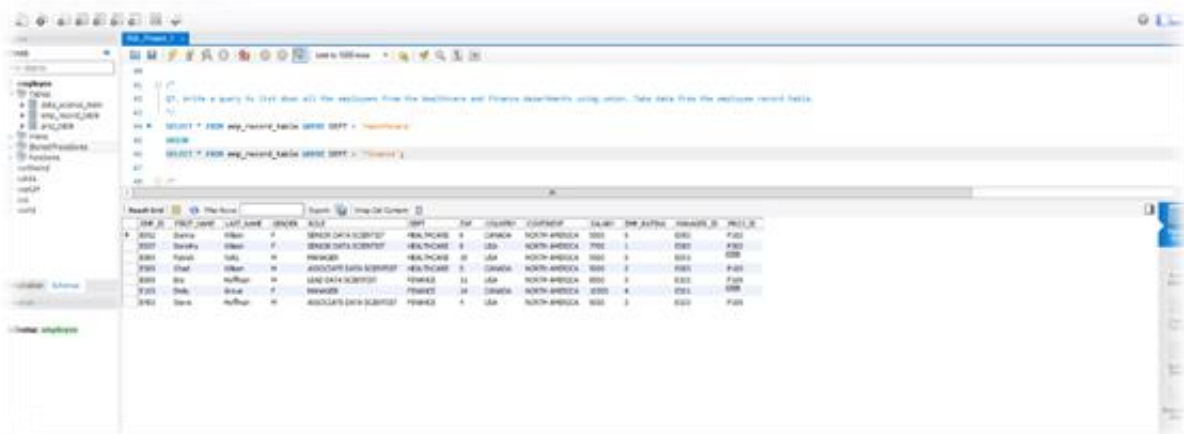
```
select concat(e.First_Name, " ", e.Last_Name) NAME, count(m.manager_ID)
from emp_record_table e join emp_record_table m on e.emp_ID =
m.manager_ID
group by NAME;
```



Q7. Write a query to list down all the employees from the healthcare and finance departments using union. Take data from the employee record table.

**Query:**

```
SELECT * FROM emp_record_table WHERE DEPT = 'Healthcare'
UNION
SELECT * FROM emp_record_table WHERE DEPT = 'Finance';
```



Q8. Write a query to list down employee details such as EMP\_ID, FIRST\_NAME, LAST\_NAME, ROLE, DEPARTMENT, and EMP\_RATING grouped by dept. Also include the respective employee rating along with the max emp rating for the department.

**Query:**

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT,
EMP_RATING, MAX(EMP_RATING) OVER (PARTITION BY DEPT) AS
MAX_RATING
FROM emp_record_table;
```

EMP_ID	FIRST_NAME	LAST_NAME	ROLE	DEPT	EMP_RATING	MAX_RATING
8861	Neena	Kochhar	MANAGER	HR	9	9
9018	Lex	DeHaan	MANAGER	HR	7	9
9199	Alexander	Toh	MANAGER	HR	5	9
9200	Bruce	Smith	MANAGER	HR	3	9
9302	David	Turner	MANAGER	HR	5	9
9303	Jane	King	MANAGER	HR	4	9
9304	John	King	MANAGER	HR	4	9
9305	Michael	King	MANAGER	HR	4	9
9306	Pat	King	MANAGER	HR	4	9
9307	Rene	King	MANAGER	HR	4	9
9308	Timothy	King	MANAGER	HR	4	9
9309	Walter	King	MANAGER	HR	4	9
9310	Yves	King	MANAGER	HR	4	9
9311	Zoe	King	MANAGER	HR	4	9
9312	Anna	Turner	MANAGER	HR	4	9
9313	Barbara	Turner	MANAGER	HR	4	9
9314	Chris	Turner	MANAGER	HR	4	9
9315	Diana	Turner	MANAGER	HR	4	9
9316	Edward	Turner	MANAGER	HR	4	9
9317	Emma	Turner	MANAGER	HR	4	9
9318	Eric	Turner	MANAGER	HR	4	9
9319	Fred	Turner	MANAGER	HR	4	9
9320	Garry	Turner	MANAGER	HR	4	9
9321	Helen	Turner	MANAGER	HR	4	9
9322	Ian	Turner	MANAGER	HR	4	9
9323	Jane	Turner	MANAGER	HR	4	9
9324	Jerry	Turner	MANAGER	HR	4	9
9325	John	Turner	MANAGER	HR	4	9
9326	Karen	Turner	MANAGER	HR	4	9
9327	Larry	Turner	MANAGER	HR	4	9
9328	Mark	Turner	MANAGER	HR	4	9
9329	Nancy	Turner	MANAGER	HR	4	9
9330	Oliver	Turner	MANAGER	HR	4	9
9331	Peter	Turner	MANAGER	HR	4	9
9332	Quinn	Turner	MANAGER	HR	4	9
9333	Rachel	Turner	MANAGER	HR	4	9
9334	Randy	Turner	MANAGER	HR	4	9
9335	Scott	Turner	MANAGER	HR	4	9
9336	Terry	Turner	MANAGER	HR	4	9
9337	Uma	Turner	MANAGER	HR	4	9
9338	Vern	Turner	MANAGER	HR	4	9
9339	Wendy	Turner	MANAGER	HR	4	9
9340	Xavier	Turner	MANAGER	HR	4	9
9341	Yara	Turner	MANAGER	HR	4	9
9342	Zoe	Turner	MANAGER	HR	4	9

Q9. Write a query to calculate the minimum and the maximum salary of the employees in each role. Take data from the employee record table.

**Query:**

```
SELECT ROLE, MIN(SALARY) AS Min_Salary, MAX(SALARY) AS
Max_Salary FROM emp_record_table GROUP BY ROLE;
```

ROLE	MIN_SALARY	MAX_SALARY
MANAGER	9000	10000
SENIOR DATA SCIENTIST	8500	9500
ASSOCIATE DATA SCIENTIST	7500	8500
DATA SCIENTIST	6500	7500
ANALYST	5500	6500
ASSOCIATE ANALYST	4500	5500
CLERK	3500	4500

Q10. Write a query to assign ranks to each employee based on their experience. Take data from the employee record table

**Query:**

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, EXP,
RANK() OVER (ORDER BY EXP DESC) AS Rank_By_Experience
FROM emp_record_table;
```

EMP_ID	FIRST_NAME	LAST_NAME	EXP	Rank_By_Experience
1001	Arthur	Bark	20	1
1002	Patrick	Nato	15	2
1003	Willy	Grove	14	3
1004	Royce	Abou	14	3
1005	Ornat	Nale	14	3
1006	Tracy	Harris	13	6
1007	William	Bulter	12	7
1008	Eric	Huffman	11	8
1009	Charity	Wilson	9	9
1010	Karen	Hood	8	10
1011	Roy	Colles	7	11
1012	Dennis	Wilson	6	12
1013	Han	Phan	6	12
1014	Chad	Wilson	5	14
1015	Dave	Huffman	4	15
1016	Daniel	Smith	3	16
1017	Clare	Branson	3	16
1018	Kenneth	Abou	2	18
1019	Walter	Phan	1	19

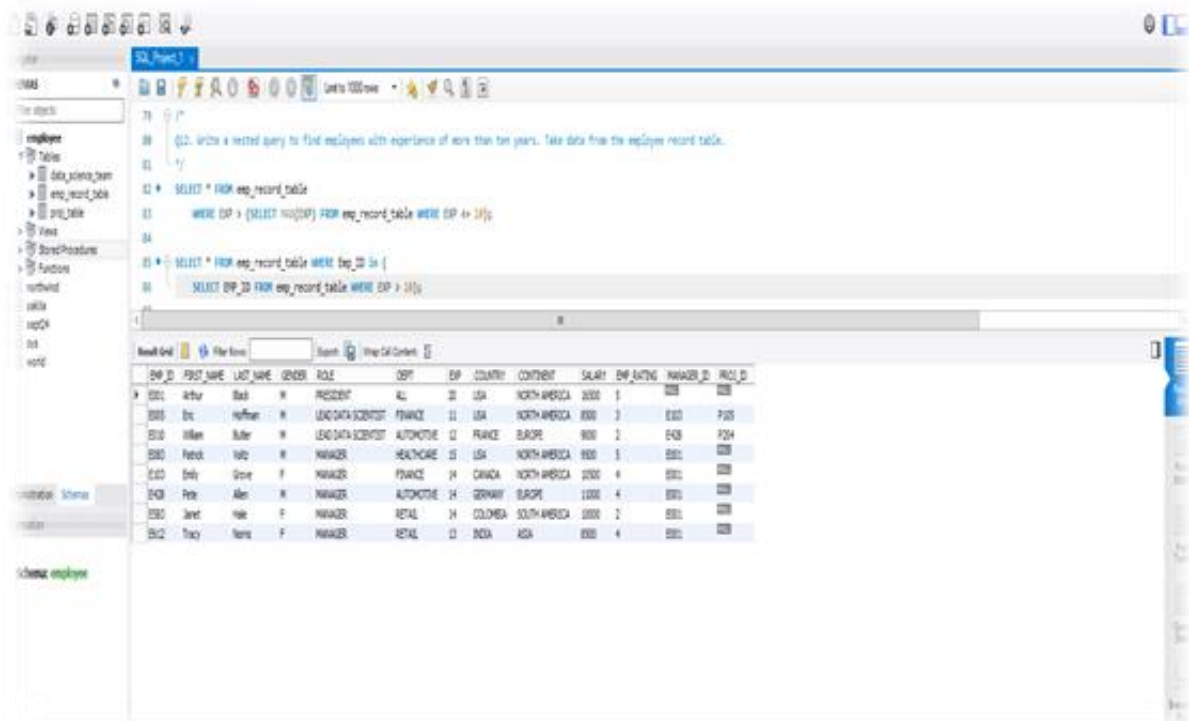
Q11. Write a query to create a view that displays employees in various countries whose salary is more than six thousand. Take data from the employee record table.

**Query:**

```
CREATE VIEW Emp_Salary_Greater_Than_6k AS
SELECT EMP_ID, FIRST_NAME, LAST_NAME, COUNTRY, SALARY
FROM emp_record_table
WHERE SALARY > 6000 order by country;
SELECT * FROM Emp_Salary_Greater_Than_6k;
```

EMP_ID	FIRST_NAME	LAST_NAME	COUNTRY	SALARY
1001	Arthur	Bark	Canada	10000
1002	Patrick	Nato	China	6000
1003	Willy	Grove	USA	10000
1004	Royce	Abou	USA	10000
1005	Ornat	Nale	USA	10000
1006	Tracy	Harris	USA	10000
1007	William	Bulter	USA	10000
1008	Eric	Huffman	USA	10000
1009	Charity	Wilson	USA	10000
1010	Karen	Hood	USA	10000

```
SELECT * FROM emp_record_table WHERE Emp_ID in (
    SELECT EMP_ID FROM emp_record_table WHERE EXP > 10);
```





Q13. Write a query to create a stored procedure to retrieve the details of the employees whose experience is more than three years. Take data from the employee record table.

**Query:**

DELIMITER \$\$

USE `employee` \$\$

CREATE PROCEDURE Emp3plusExp ()

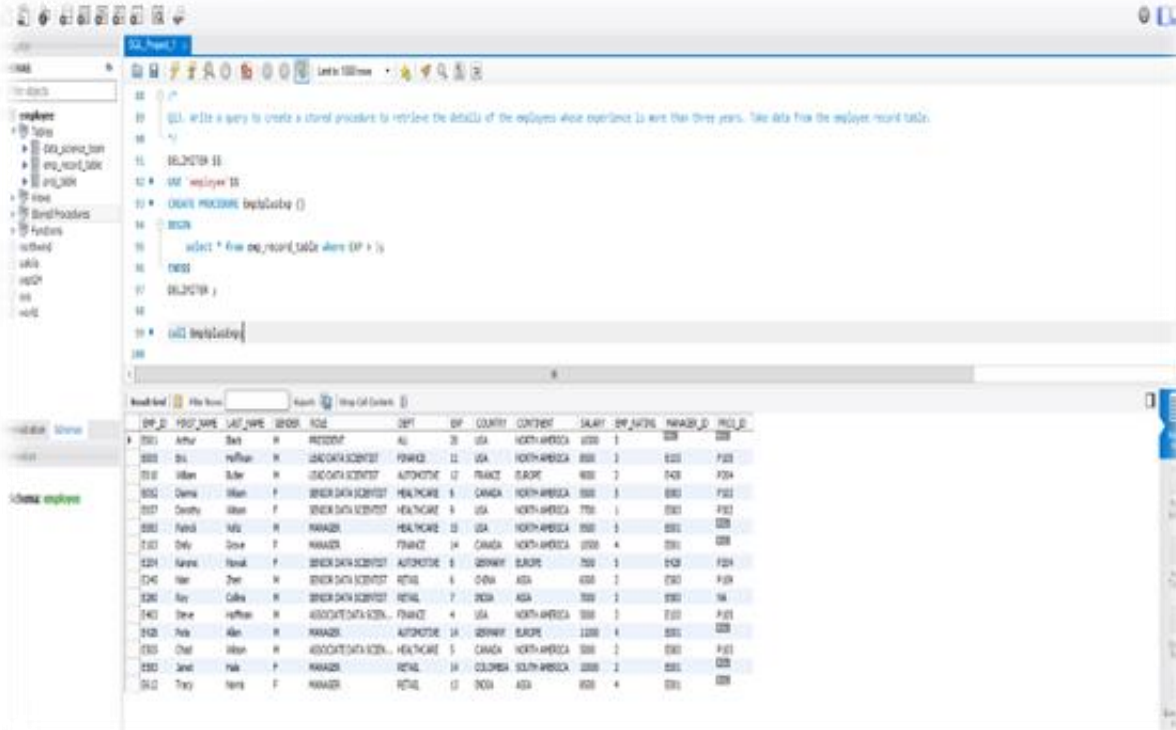
BEGIN

select \* from emp\_record\_table where EXP > 3;

END\$\$

DELIMITER ;

call Emp3plusExp;



The screenshot shows a SQL IDE with the following content:

**SQL Script:**

```

11 DELIMITER $$
12 USE `employee` $$
13 CREATE PROCEDURE Emp3plusExp ()
14 BEGIN
15     select * from emp_record_table where EXP > 3;
16 END$$
17 DELIMITER ;
18
19 call Emp3plusExp;

```

**Result Set:**

EMP_ID	FIRST_NAME	LAST_NAME	SEX	STATUS	ROLE	DEPT	EXP	COUNTRY	CONTINENT	SALARY	EXP_AGEING	MANAGER_ID	PROJ_ID
0001	Arthur	Bark	M	REGULAR	FINANCE	11	USA	NORTH AMERICA	0000	0	000	000	
0002	Bruce	Jefferson	M	ASSOCIATE DATA SCIENTIST	FINANCE	12	USA	NORTH AMERICA	0000	0	000	000	
0003	William	Suber	M	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	12	FRANCE	EUROPE	0000	0	000	000	
0004	Diana	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	9	CANADA	NORTH AMERICA	0000	0	000	000	
0005	Georgi	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	9	USA	NORTH AMERICA	7000	1	000	000	
0006	Patrick	Wills	M	MANAGER	HEALTHCARE	10	USA	NORTH AMERICA	0000	0	000	000	
0007	Emily	Cooper	F	MANAGER	FINANCE	14	CANADA	NORTH AMERICA	0000	4	000	000	
0008	Marina	Rowland	F	SENIOR DATA SCIENTIST	AUTOMOTIVE	9	GERMANY	EUROPE	7000	1	000	000	
0009	Neil	Zhao	M	SENIOR DATA SCIENTIST	RETAIL	6	CHINA	ASIA	0000	0	000	000	
0010	Roy	Chen	M	SENIOR DATA SCIENTIST	RETAIL	7	INDIA	ASIA	0000	0	000	000	
0011	Steve	Jefferson	M	ASSOCIATE DATA SCIENTIST	FINANCE	4	USA	NORTH AMERICA	0000	0	000	000	
0012	Paula	Allen	M	MANAGER	AUTOMOTIVE	16	GERMANY	EUROPE	11000	6	000	000	
0013	Chad	Wilson	M	ASSOCIATE DATA SCIENTIST	HEALTHCARE	5	CANADA	NORTH AMERICA	0000	0	000	000	
0014	Jared	Park	F	MANAGER	RETAIL	18	COLUMBIA	SOUTH AMERICA	0000	0	000	000	
0015	Tracy	Harris	F	MANAGER	RETAIL	17	INDIA	ASIA	0000	4	000	000	

Q14. Write a query using stored functions in the project table to check whether the job profile assigned to each employee in the data science team matches the organization's set standard.

The standard being:

For an employee with experience less than or equal to 2 years assign 'JUNIOR DATA SCIENTIST',

For an employee with the experience of 2 to 5 years assign 'ASSOCIATE DATA SCIENTIST',

For an employee with the experience of 5 to 10 years assign 'SENIOR DATA SCIENTIST',

For an employee with the experience of 10 to 12 years assign 'LEAD DATA SCIENTIST',

For an employee with the experience of 12 to 16 years assign 'MANAGER'.

**Query:**

DELIMITER \$\$

USE `employee` \$\$

CREATE FUNCTION Check\_JobProfiles (eid varchar(5))

RETURNS varchar(100)

DETERMINISTIC

BEGIN

declare ex int;

declare r varchar(80);

declare vrole varchar(100);

declare flag varchar(10);

select exp, ROLE into ex, VROLE from data\_science\_team where emp\_ID = eid;

if ex > 12 and ex < 16 then

if VROLE = 'Manager' then

set flag = 'Yes';

else

set flag = 'No';

end if;

# set r = 'Manager';

```

elseif ex > 10 and ex <= 12 then
    if VROLE = 'LEAD DATA SCIENTIST' then
        set flag = 'Yes';
    else
        set flag = 'No';
    end if;
elseif ex > 5 and ex <=10 then
    if VROLE = 'SENIOR DATA SCIENTIST' then
        set flag = 'Yes';
    else
        set flag = 'No';
    end if;
elseif ex > 2 and ex <=5 then
    if VROLE = 'ASSOCIATE DATA SCIENTIST' then
        set flag = 'Yes';
    else
        set flag = 'No';
    end if;
elseif ex <= 2 then
    if VROLE = 'JUNIOR DATA SCIENTIST' then
        set flag = 'Yes';
    else
        set flag = 'No';
    end if;
end if;

```

RETURN flag;

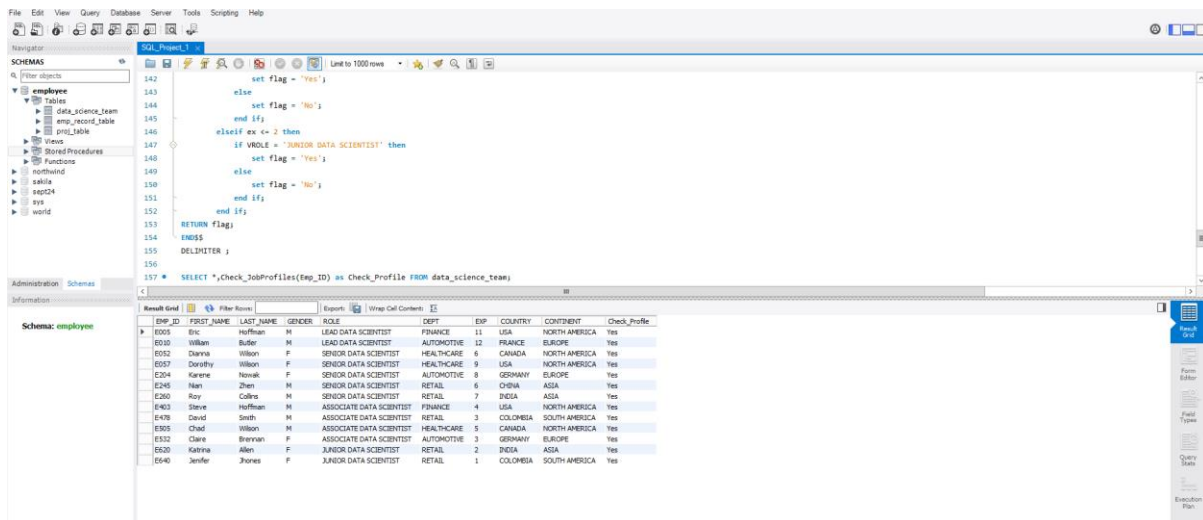
END\$\$

DELIMITER ;

```

SELECT *,Check_JobProfiles(Emp_ID) as Check_Profile FROM
data_science_team;

```



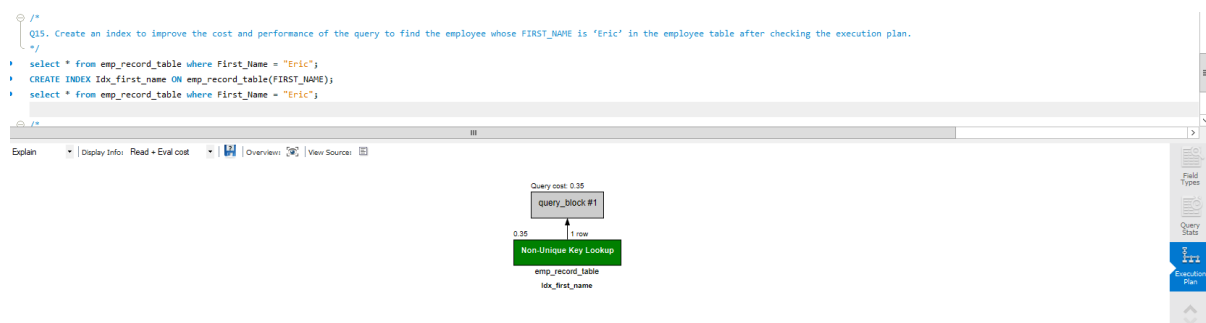
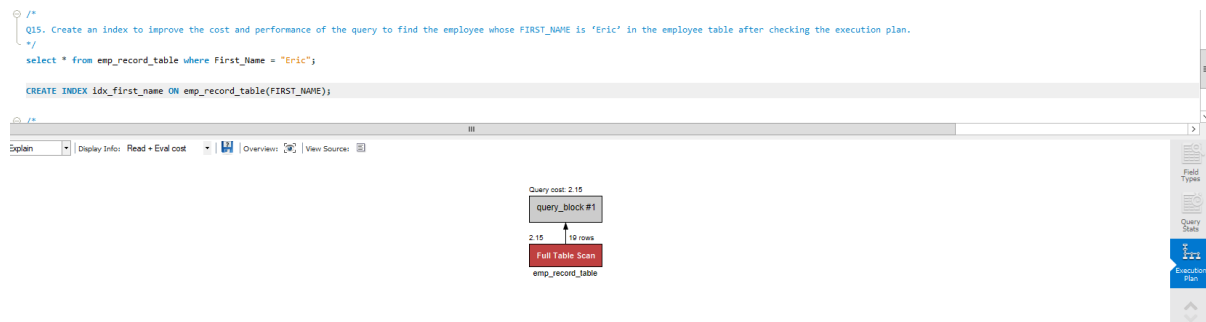
Q15. Create an index to improve the cost and performance of the query to find the employee whose FIRST\_NAME is 'Eric' in the employee table after checking the execution plan.

### Query:

```
select * from emp_record_table where First_Name = "Eric";
```

```
CREATE INDEX Idx_first_name ON emp_record_table(FIRST_NAME);
```

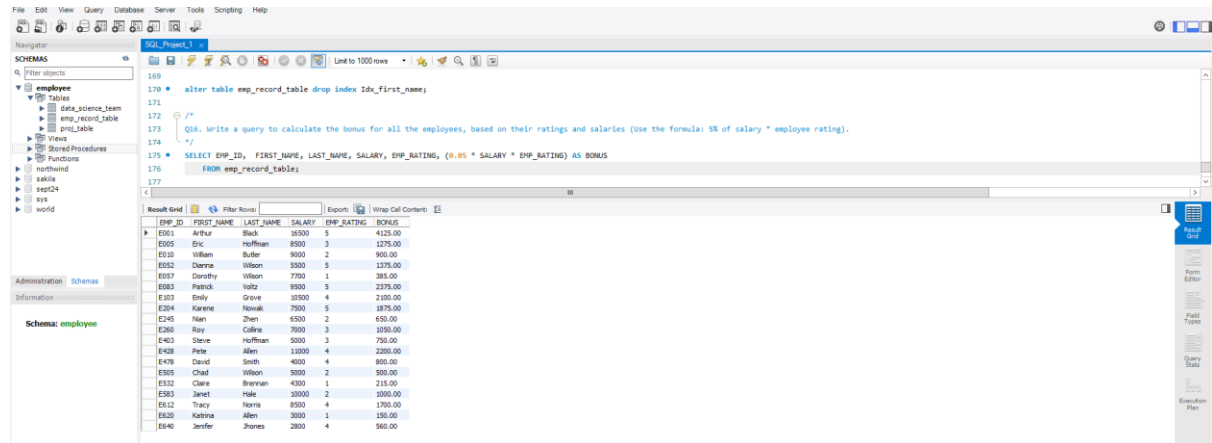
```
select * from emp_record_table where First_Name = "Eric";
```



Q16. Write a query to calculate the bonus for all the employees, based on their ratings and salaries (Use the formula: 5% of salary \* employee rating).

### Query:

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, SALARY, EMP_RATING, (0.05 *  
SALARY * EMP_RATING) AS BONUS  
  
FROM emp_record_table;
```

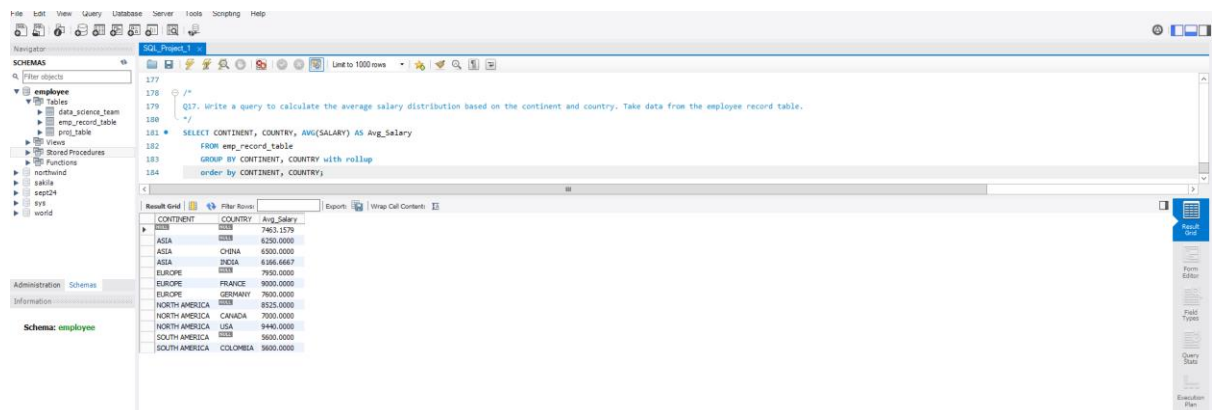


EMP_ID	FIRST_NAME	LAST_NAME	SALARY	EMP_RATING	BONUS
E001	Arthur	Black	10500	5	4125.00
E005	Eric	Hoffman	8500	3	1275.00
E010	William	Butler	9000	2	900.00
E052	Dierma	Wilson	5500	5	1375.00
E057	Dorothy	Wilson	7700	1	385.00
E083	Patrick	Voltz	9900	5	2375.00
E103	Emily	Grove	10500	4	2100.00
E204	Karene	Hovaki	7500	5	1875.00
E245	Nan	Zhen	6500	2	650.00
E300	Roy	Collins	7000	3	1050.00
E403	Stere	Hoffman	5000	3	750.00
E408	Pete	Allen	11000	4	2200.00
E478	David	Smith	4000	4	800.00
E555	Chad	Wilson	5000	2	500.00
E532	Clare	Brennan	4300	1	215.00
E583	Janet	Hale	10000	2	1000.00
E612	Tracy	Norris	8500	4	1700.00
E630	Katrina	Allen	3000	1	150.00
E640	Jenfer	Jones	2800	4	560.00

Q17. Write a query to calculate the average salary distribution based on the continent and country. Take data from the employee record table.

### Query:

```
SELECT CONTINENT, COUNTRY, AVG(SALARY) AS Avg_Salary  
  
FROM emp_record_table  
  
GROUP BY CONTINENT, COUNTRY with rollup  
  
order by CONTINENT, COUNTRY;
```



CONTINENT	COUNTRY	Avg_Salary
ASIA	CHINA	6250.0000
ASIA	INDIA	6166.6667
ASIA	THAILAND	7950.0000
EUROPE	FRANCE	9000.0000
EUROPE	GERMANY	7800.0000
NORTH AMERICA	CANADA	7000.0000
NORTH AMERICA	USA	8440.0000
SOUTH AMERICA	BRAZIL	5800.0000
SOUTH AMERICA	COLOMBIA	9600.0000
ASIA	ROLLUP	7463.1579