

Assignment-2 CS303

Shashank P
200010048

September 28, 2022

1 Problem 1

Suppose that we have a relation marks(ID, score) and we wish to assign grades to students based on the score as follows: grade F if score ≤ 40 , grade C if $40 < \text{score} < 60$, grade B if $60 \leq \text{score} < 80$, and grade A if $80 \leq \text{score}$. Write SQL queries to do the following:

1.1 Part (a)

Display the grade for each student, based on the marks relation.

```
1 select ID, score ,
2     case
3         when score >= 80 then 'A'
4         when score >= 60 then 'B'
5         when score >= 40 then 'C'
6         else 'F'
7     end as grade
8 from marks
```

1.2 Part (b)

Display the grade for each student, based on the marks relation.

```
1 select case
2     when score >= 80 then 'A'
3     when score >= 60 then 'B'
4     when score >= 40 then 'C'
5     else 'F'
6 end as grade, count(*) as grade_count
7 from marks
8 group by grade
```

2 Problem 2

Using tables given in the question, write SQL queries for the following:

2.1 Part (a)

Give all employees of “First Bank Corporation” a 10 percent raise.

```
1 update works
2 set salary = salary*1.1
3 where company_name='First Bank Corporation'
```

2.2 Part (b)

Give all managers of “First Bank Corporation” a 10 percent raise.

```
1 update works
2 set salary = salary*1.1
3 where employee_name in
4     (select distinct M.manager_name
5      from works as W inner join manages as M
6      on W.employee_name=M.manager_name
7      where W.company_name='First Bank Corporation')
```

2.3 Part (c)

Delete all tuples in the works relation for employees of “Small Bank Corporation”.

```
1 delete from works
2 where company_name='Small Bank Corporation'
```

2.4 Part (d)

Find all employees in the database who live in the same cities as the companies for which they work.

```
1 select distinct employee.employee_name
2 from employee, works, company
3 where employee.city=company.city
4     and employee.employee_name=works.employee_name
5     and works.company_name=company.company_name
```

2.5 Part (e)

Find all employees in the database who live in the same cities and on the same streets as do their managers.

```
1 with manager(manager_name, street, city) as
2   (select employee_name, street, city
3    from employee
4    where employee_name in
5      (select distinct T.manager_name
6       from manages as T))
7 select distinct E.employee_name
8 from employee as E, manages as Ms, manager as Mr
9 where E.employee_name=Ms.employee_name
10    and Mr.manager_name=Ms.manager_name
11    and E.street=Mr.street
12    and E.city=Mr.city
```

2.6 Part (f)

Find all employees who earn more than the average salary of all employees of their company.

```
1 with avg_salary(c_name, val) as
2   (select company_name, avg(salary)
3    from works
4    group by company_name)
5 select distinct W.employee_name
6 from works as W inner join avg_salary
7 on W.company_name=avg_salary.c_name
8 where W.salary>avg_salary.val
```

2.7 Part (g)

Find the company that has the smallest payroll.

```
1 with salary_sum(c_name, val) as
2   (select company_name, sum(salary)
3    from works
4    group by company_name)
5 with min_sum(val) as
6   (select min(val)
7    from salary_sum)
8 select distinct salary_sum.c_name
9 from salary_sum, min_sum
10 where salary_sum.val=min_sum.val
```

2.8 Part (h)

Find the company that has the most employees.

```
1 with pay_counts(c_name, val) as
2   (select company_name, count(distinct employee_name)
3    from works
4    group by company_name)
5 with max_count(val) as
6   (select max(val)
7    from pay_counts)
8 select distinct pay_counts.c_name
9 from pay_counts, max_count
10 where pay_counts.val=max_count.val
```

2.9 Part (i)

Find those companies whose employees earn a higher salary, on average, than the average salary at “First Bank Corporation”.

```
1 with FBC_avg(val) as
2   (select avg(salary)
3    from works
4    where company_name='First Bank Corporation')
5 with avg_salary(c_name, val) as
6   (select company_name, avg(salary)
7    from works
8    group by company_name)
9 select distinct c_name
10 from avg_salary, FBC_avg
11 where avg_salary.val>FBC_avg.val
```

2.10 Part (j)

Modify the database so that “Jones” now lives in “Newtown”.

```
1 update employee
2 set city='Newtown'
3 where employee_name='Jones'
```

2.11 Part (k)

Give all managers of “First Bank Corporation” a 10 percent raise unless the salary becomes greater than \$100,000; in such cases, give only a 3 percent raise.

```
1 update works
2 set salary = case
3     when salary*1.1>100000 then salary*1.03
4     else salary*1.1
5     end
6 where employee_name in
7     (select distinct M.manager_name
8     from works as W inner join manages as M
9     on W.employee_name=M.manager_name
10    where W.company_name='First Bank Corporation')
```

3 Problem 3

Write a query to get the list of users who took a training lesson more than once in the same day, grouped by user and training lesson, each ordered from the most recent lesson date to oldest date.

```
1 select *
2 from users natural join
3     (select user_id, training_id, training_date
4     from training_details
5     group by user_id, training_id, training_date
6     having count(*)>1)
7 order by training_date desc
```

4 Problem 4

What is the meaning of the query given below?

```
1 SELECT * FROM runners WHERE id NOT IN (SELECT winner_id FROM races)
```

The **meaning of the query** is to get the details of all those runners who did not win in any of the races. But it will return as empty set as **NOT IN** key word is used, which will return an **empty set** if any value in the sub query is **NULL**.

5 Problem 5

Consider books and publishers table. A publisher may have zero or many books while a book may belong to zero or one publisher. The relationship between the books table and the publishers table is zero-to-many.

5.1 Part (a)

A query which will return information about books with publishers, irrespective of whether a book has associated publishers or not.

```
1 select *
2 from books left outer join publishers
3 on books.publisher_id=publishers.publisher_id
```

5.2 Part (b)

A query which will return information about books with publishers, irrespective of whether the publisher has any published books or not.

```
1 select *
2 from books right outer join publishers
3 on books.publisher_id=publishers.publisher_id
```