Assignment-2 CS303

Shashank P
200010048

September 27, 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 $\leq$ score $<$ 60, grade B if 60 $\leq$ score ¡ 80, and grade A if 80 $\leq$ score. Write SQL queries to do the following:

1.1 Part (a)

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

```
select ID, score,
  case
      when score>=80 then 'A'
      when score>=60 then 'B'
      when score>=40 then 'C'
      else 'F'
  end as grade
from marks
```

1.2 Part (b)

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

```
select case
    when score>=80 then 'A'
    when score>=60 then 'B'
    when score>=40 then 'C'
    else 'F'
  end as grade, count(*) as grade_count
from marks
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.

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

2.2 Part (b)

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

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

2.3 Part (c)

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

```
delete from works
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.

```
select distinct employee.employee_name
from employee, works, company
where employee.city=company.city
    and employee.employee_name=works.employee_name
    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.

```
with manager(manager_name, street, city) as
  (select employee_name, street, city
  from employee
  where employee_name in
      (select distinct T.manager_name
      from manages as T))
select distinct E.employee_name
from employee as E, manages as Ms, manager as Mr
where E.employee_name=Ms.employee_name
  and Mr.manager_name=Ms.manager_name
  and E.street=Mr.street
  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.

```
with avg_salary(c_name, val) as
  (select company_name, avg(salary)
  from works
  group by company_name)
select distinct W.employee_name
from works as W inner join avg_salary
on W.company_name=avg_salary.c_name
where W.salary>avg_salary.val
```

2.7 Part (g)

Find the company that has the smallest payroll.

```
with salary_sum(c_name, val) as
  (select company_name, sum(salary)
  from works
  group by company_name)
with min_sum(val) as
  (select min(val)
  from salary_sum)
select distinct salary_sum.c_name
from salary_sum, min_sum
where salary_sum.val=min_sum.val
```

2.8 Part (h)

Find the company that has the most employees.

```
with pay_counts(c_name, val) as
  (select company_name, count(distinct employee_name)
  from works
  group by company_name)
with max_count(val) as
  (select max(val)
  from pay_counts)
select distinct pay_counts.c_name
from pay_counts, max_count
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".

```
with FBC_avg(val) as
  (select avg(salary)
  from works
  where company_name='First Bank Corporation')
with avg_salary(c_name, val) as
  (select company_name, avg(salary)
  from works
  group by company_name)
select distinct c_name
from avg_salary, FBC_avg
where avg_salary.val>FBC_avg.val
```

2.10 Part (j)

Modify the database so that "Jones" now lives in "Newtown".

```
update employee
set city='Newtown'
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.

```
update works
set salary = case
    when salary*1.1>100000 then salary*1.03
    else salary*1.1
    end
where employee_name in
    (select distinct M.manager_name
    from works as W inner join manages as M
    on W.employee_name=M.manager_name
    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.

```
select *
from users natural join
    (select user_id, training_id, training_date
    from training_details
    group by user_id, training_id, training_date
    having count(*)>1)
order by training_date desc
```

4 Problem 4

What is the meaning of the query given below?

```
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.

```
select *
from books left outer join publishers
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.

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