

Chandigarh College of Engineering & Technology (Degree Wing)



Department of Computer Science and Engineering Database Systems (Practical)

CS 352

Practical - 7

DOP: - 23/08/2024

DOS:-30/08/2024

Submitted By:

Abhay Pratap Singh
(Roll No: - CO23306)

Submitted To:

Dr. Dheerendra Singh
CSE Department

Running of at least 10 different SQL queries on each “Banking, and University database Project” on XAMPP server.

❖ Running 10 different queries on banking database

SQL QUERY 1:-

```
select distinct customer_name from borrower where customer_name in (select customer_name from depositor);
```

Output :- Find all customers who have both an account and a loan at the bank.

| customer_name |
|---------------|
| Hayes |
| Jones |
| Smith |

SQL QUERY 2:-

```
select distinct T.branch_name from branch as T, branch as S where T.assets > S.assets and S.branch_city = 'Brooklyn';
```

Output :- Find all branches that have greater assets than some branch located in Brooklyn.

| branch_name |
|-------------|
| Downtown |
| Round Hill |

SQL QUERY 3:-

```
select loan_number from loan where branch_name = 'Perryridge' and amount > 1200;
```

Output :- find all loan number for loans made at the Perryridge branch with loan amounts greater than \$1200

| loan_number |
|-------------|
| L-15 |
| L-16 |
| L-15 |
| L-16 |

SQL QUERY 4:-

(select customer_name from depositor) union (select customer_name from borrower);

Output :- Find all customers who have a loan, an account, or both.

| customer_name |
|---------------|
| Hayes |
| Johnson |
| Jones |
| Lindsay |
| Smith |
| Turner |
| Curry |
| Jackson |
| Williams |
| Adams |

SQL QUERY 5:-

select branch_name, avg (balance) from account group by branch_name;

Output :- Find the avg. account balance at each branch.

| branch_name | avg (balance) |
|-------------|---------------|
| Brighton | 825.0000 |
| Downtown | 500.0000 |
| Mianus | 700.0000 |
| Perryridge | 400.0000 |
| Redwood | 700.0000 |
| Round Hill | 350.0000 |

SQL QUERY 6:-

(select customer_name from depositor) intersect (select customer_name from borrower);

Output : - Find all customers who have both a loan and an account.

customer_name

Hayes

Jones

Smith

SQL QUERY 7:-

SELECT branch_name, COUNT(DISTINCT customer_name) FROM depositor INNER JOIN account ON depositor.account_number = account.account_number GROUP BY branch_name;

Output :- Find the no. of depositors in each branch.

| branch_name | COUNT(DISTINCT customer_name) |
|-------------|-------------------------------|
| Brighton | 2 |
| Downtown | 1 |
| Mianus | 1 |
| Perryridge | 1 |
| Redwood | 1 |
| Round Hill | 1 |

SQL QUERY 8:-

select avg (balance) from account;

Output :- Find the average balance for all accounts.

avg (balance)

614.2857

SQL QUERY 9 :-

SELECT * FROM `account` ORDER BY balance ASC, branch_name DESC;

Output :- Arranges data by balance in ascending and branch name in descending.

| account_number | branch_name ▼ 2 | balance ▲ 1 |
|----------------|-----------------|-------------|
| A-305 | Round Hill | 350 |
| A-305 | Round Hill | 350 |
| A-102 | Perryridge | 400 |
| A-102 | Perryridge | 400 |
| A-101 | Downtown | 500 |
| A-101 | Downtown | 500 |
| A-222 | Redwood | 700 |
| A-222 | Redwood | 700 |
| A-215 | Mianus | 700 |
| A-215 | Mianus | 700 |
| A-217 | Brighton | 750 |
| A-217 | Brighton | 750 |
| A-201 | Brighton | 900 |
| A-201 | Brighton | 900 |

SQL QUERY 10:-

select distinct S.customer_name from depositor as S where not exists ((select branch_name from branch where branch_city = 'Brooklyn') except (select R.branch_name from depositor as T, account as R where T.account_number = R.account_number and S.customer_name = T.customer_name));

Output :- Find all customers who have an account at all branches located in Brooklyn.

| |
|---------------|
| customer_name |
| Johnson |

❖ Running 10 different queries on University database

SQL QUERY 1:-

```
select course_id from section where semester = 'Spring' and year = 2010;
```

Output :- Set of all courses taught in the Spring 2010 semester.

| course_id |
|-----------|
| CS-101 |
| CS-315 |
| CS-319 |
| CS-319 |
| FIN-201 |
| HIS-351 |
| MU-199 |

SQL QUERY 2:-

```
(select course_id from section where semester = 'Fall' and year = 2009) intersect (select course_id from section where semester = 'Spring' and year = 2010);
```

Output :- Find courses that ran in Fall 2009 and in Spring 2010.

| course_id |
|-----------|
| CS-101 |

SQL QUERY 3:-

```
(select course_id from section where semester = 'Fall' and year = 2009) except (select course_id from section where semester = 'Spring' and year = 2010);
```

Output :- Find courses that ran in Fall 2009 or in Spring 2010.

| course_id |
|-----------|
| CS-347 |
| PHY-101 |

SQL QUERY 4:-

```
select distinct T.salary from instructor as T, instructor as S where T.salary < S.salary;
```

Output :- Find the salaries of all instructors that are less than the largest salary.

| salary |
|-----------|
| 53603.82 |
| 80405.75 |
| 83085.93 |
| 87106.23 |
| 110052.22 |
| 100507.18 |
| 105165.61 |
| 96486.88 |
| 113847.14 |
| 114160.36 |

SQL QUERY 5:-

```
select avg (salary) from instructor where dept_name= 'Comp. Sci.';
```

Output :- Find the average salary of instructors in the Computer Science department.

| avg (salary) |
|--------------|
|--------------|

100591.256667

SQL QUERY 6:-

```
select count(distinct ID) from teaches where semester = "Spring" and year = 2010;
```

Output :- Find the total number of instructors who teach a course in the Spring 2010 semester.

| count(distinct ID) |
|--------------------|
|--------------------|

6

SQL QUERY 7:-

```
select dept_name, avg (salary) as avg_salary from instructor group by dept_name;
```

Output :- Find the average salary of instructors in each department.

| dept_name | avg_salary |
|------------|---------------|
| Biology | 96486.880000 |
| Comp. Sci. | 100591.256667 |
| Elec. Eng. | 105165.610000 |
| Finance | 109506.375000 |
| History | 81745.840000 |
| Music | 53603.820000 |
| Physics | 113967.605000 |

SQL QUERY 8:-

```
select dept_name, count(distinct ID) as instr_count from instructor NATURAL JOIN teaches where semester = "spring" and year = 2010 group by dept_name;
```

Output :- Find the number of instructor in each department who teach a course in the Spring 2010 semester.

| dept_name | instr_count |
|------------|-------------|
| Comp. Sci. | 3 |
| Finance | 1 |
| History | 1 |
| Music | 1 |

SQL QUERY 9:-

```
select distinct course_id from section where semester = 'Fall' and year= 2009 and course_id in (select course_id from section where semester = 'Spring' and year= 2010);
```

Output :- Find courses offered in Fall 2009 and in Spring 2010.

| course_id |
|-----------|
| CS-101 |

SQL QUERY 10:-

```
with dept_total (dept_name, value) as (select dept_name, sum(salary) from instructor group by dept_name),  
dept_total_avg(value) as (select avg(value) from dept_total) select dept_name from dept_total, dept_total_avg  
where dept_total.value > dept_total_avg.value;
```

Output :- Find all departments where the total salary is greater than the average of the total salary at all departments.

| dept_name |
|------------|
| Comp. Sci. |
| Finance |
| Physics |