CS561: Summer I 2013
Professor S. Kim

Stevens Institute of Technology
Department of Computer Science

Midterm Exam

June 18, 2013

Do not open the exam until you are instructed to do so.

This exam is worth 100 points (30% of your grade), and you have 120 minutes to finish the exam. This exam is closed book and closed notes.

Please write your name on the space provided on every page of the exam. The space provided should be sufficient for the answers. If you need more space, use the back sides of the exam pages and indicate clearly which question you are answering. Write your answers "clearly."

Name: Shen Yue

CWID: 10372545

Signature:

1	2	3	4	5	6	7	Total
25	25	15	8	6	6	15	**100**
~~21~~ 22	~~25~~ 25	15	7	6	6	15	96

YOUR NAME: Shen Yue

1. (25 pts.) Express the following queries from the programming assignments in SQL.

Use the schema, `sales (prod, cust, yr, mo, day, state, quant)`

(a) (10 pts.) For 2008, show for each product, the *total sales quantities* along with *average*, *maximum* and *minimum sales quantities* if the average sales quantities are more than 50% of the maximum sales quantities and less than twice the minimum sales quantities.

```
Select prod, sum(quant), avg(quant), max(quant), min(quant)
from sales
where yr = 2008
group by prod
having avg(quant) > 0.5 * max(quant) and
       avg(quant) < 2 * min(quant)
```

10

(b) (15 pts.) For each combination of *customer* and *product*, output the *average* sales quantity and the *maximum* sales quantity along with the *date of the maximum sales quantity*.

```
create view v1 as
Select prod, cust, avg(quant)
from sales
group by prod, cust
```

12

cust-prod combination

~~select v1.prod, v1.cust, v1.avg(quant), sales.max(quant),~~
~~sales.yr, sales.mo, sales.day~~
~~from v1, sales~~
~~where v1.prod = sales.prod and~~
~~v1.cust = sales.cust~~
~~group by prod, cust~~

Rita. office hours?

See behind please

1.(b). cont.

```
create view V2 as
Select V1.prod, V1.cust, max(V1.quant), sales.yr, sales.mo, sales.day
from sales, V1
where V1.prod = sales.prod and V1.cust = sales.cust
group by prod, cust
```

Select V_1.prod, V_1.cust, V_2.avg(quant), V_2.max(quant),
V_2.yr, V_2.mo, V_2.day

from V_1, V_2

where V_1.prod = V_2.prod and
V_1.cust = V_2.cust

group by prod, cust

YOUR NAME: Shen Yue

2. (25 pts.) Provide an expression in the relational algebra to express each of the following queries. Use the following relational database:

employee (person-name, street, city)
works (person-name, company-name, salary)
company (company-name, city)
manages (person-name, manager-name)

a) Give all employees of First Bank Corporation a 10 percent salary raise.

5 works ← Π person_name, company_name, 1.1 * salary (σ company_name = "First Bank Corporation" (works))

∪ (works − σ company_name = "First Bank Corporation" (works))

b) Find the names, street addresses, and cities of residence of all employees who work for First Bank Corporation and earn more than $10,000.

5 Π person_name, street, city (σ company_name = "First Bank Corporation" ∧ salary > 10,000 (works ⋈ employee))

c) Find the names of all employees who live in the same city and on the same street as do their managers.

5 Π person_name (σ a.person_name = c.person_name and b.person_name = c.manger_name and a.city = b.city and a.street = b.street (ρa (employee) × ρb (employee) × ρc (manages)))

d) Find the names of all employees who earn more than every employee of Small Bank Corporation.

5 Π person_name (works) − Π works2.person_name (works ⋈ σ work.salary ≤ works2.salary and works2.company_name = "Small Bank Corporation" (ρ works2 (works)))

t1 ← company_name g count distinct person_name (works)

5 t2 ← Max num_employee (ρ company_strength (company_name g, num_employee (t1)))

Π company_name (ρ t3 (company_name, num_employee) (t1) ⋈ ρ t4 (num_employee) (t2))

YOUR NAME: Shen Yue

3. (15 pts.) Answer the following questions

a) (2 pts.) Define the relational operator, 'intersection' using another relational operator.

2 $r \cap s =$ r-(r-s)

b) (3 pts.) Describe the following expression in English.

$\Pi_{customer\_name}\ (\sigma_{branch\_name\ =\ "Perryridge"}\ (\sigma_{borrower.loan\_number\ =\ loan.loan\_number}\ (borrower \times loan)))$

3 Find out all the customers who has a loan at Perryridge branch.

c) (5 pts.) The expression above requires a full Cartesian product between "borrower" and "loan". Provide a relational expression equivalent to the one above (in (b)) but more "efficient".

5 $\Pi_{customer\_name}(\sigma_{borrower.loan\_number = loan.loan\_number}(\sigma_{branch\_name = "perryridge"}(loan) \times borrower))$

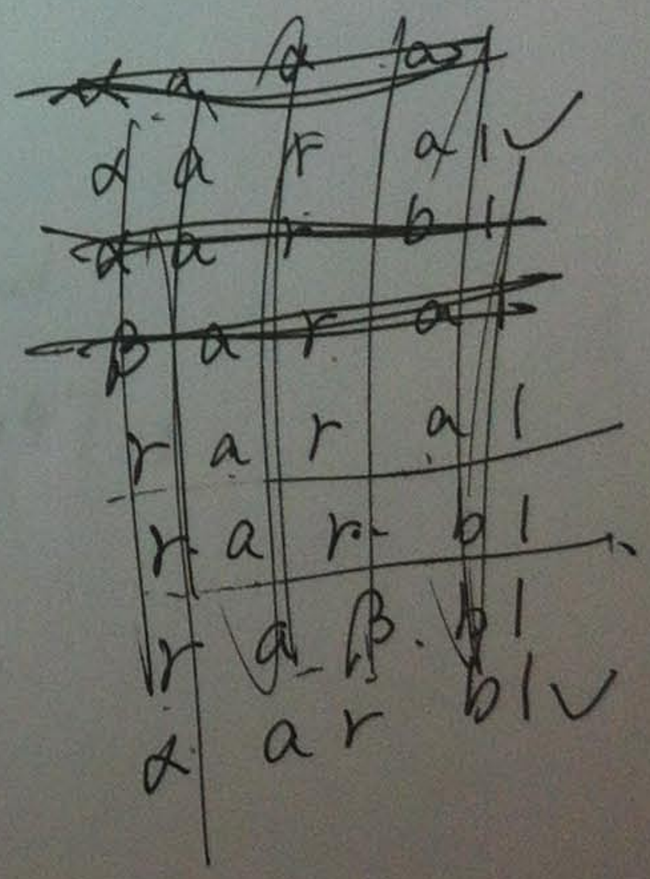
d) (5 pts.) Given the following two relations, r and s, what is r ÷ s?

A	B	C	D	E
α	a	α	a	1
α	a	γ	a	1
α	a	γ	b	1
β	a	γ	a	1
β	a	γ	b	3
γ	a	γ	a	1
γ	a	γ	b	1
γ	a	β	b	1

r

D	E
a	1
b	1

s



5 r ÷ s?

A	B	C
α	a	r
r	a	r

YOUR NAME: Shen Yue

4. (8 pts.) Based on the following schema, give an expression in SQL for each of the following queries.

*person (**driver_id**, name, address)* ✓
*car (**license**, model, year)*
△ *accident (**report_number**, **date**, location)*
*owns (**driver_id**, **license**)*
*participated (**driver_id**, **license**, **report_number**, damage_amount)* ✓

Figure 3.11. Insurance database.

a) Find the number of accidents in which the cars belonging to "John Smith" were involved.

3

```
Select count *          —— distinct
from accident
where report_number in (select report_number
                        from person a, participated b
                        where a.name = "John Smith" and
accident.report_number = b.report_number and a.driver_id = b.driver_id)
```

b) Update the damage amount for the car with license number "AABB2000" in the accident with report number "AR2197" to $3000.

4

```
update participated
set damage_amount = 3000
where license = "AABB2000" and
      report_number = "AR2197";
```

YOUR NAME: Shen Yue

5. (6 pts.) Express the following SQL queries in English:

a)
```
select distinct customer-name
from borrower, loan
where borrower.loan-number = loan.loan-number and
        branch-name = "Perryridge" and
        (branch-name, customer-name) in
                (select branch-name, customer-name
                 from depositor, account
                 where depositor.account-number = account.account-number)
```

3

Find out all the customers who both have account and loan at Perryridge branch and show the ~~customer name~~ customer name once.

b)
```
select T.customer-name
from depositor as T
where unique (
        select R.customer-name
        from account, depositor as R
        where T.customer-name = R.customer-name and
                R.account-number = account.account-number and
                account.branch-name = "Perryridge")
```

3

Find out all the customers who has only an account at Perryridge branch.

6. (6 pts.) Answer the questions below regarding the following SQL query.

```
select branch_name, avg_balance
from (select branch_name, avg (balance)
        from account
        group by branch_name )
  as branch_avg ( branch_name, avg_balance )
where avg_balance > 1200
```

3

a) Express the query in English

Find out all the branches which ~~avg~~ average balance is greater than 1200 and show the branch_name with their average balance.

b) Express the above query using a 'having' clause

```
select branch_name, avg(balance)
from account
group by branch_name
having avg(balance) > 1200
```

3

YOUR NAME: Shen Yue

7. (15 pts.) Given the following 2 relations, 'loan' and 'borrower', provide results to the following join queries (draw the resulting output of the join queries).

loan:

branch-name	*loan-number*	*amount*
Downtown	L-170	3000
Redwood	L-230	4000
Perryridge	L-260	1700

borrower:

customer-name	*loan-number*
Jones	L-170
Smith	L-230
Hayes	L-155

a) loan **inner join** borrower **on** loan.loan-number = borrower.loan-number

3

branch_name	loan_number	amount	customer_name	loan_number
Downtown	L-170	3000	Jones	L-170
Redwood	L-230	4000	Smith	L-230

b) loan **left outer join** borrower **on** loan.loan-number=borrower.loan-number

3

branch_name	loan_number	amount	customer_name	loan_number
Downtown	L-170	3000	Jones	L-170
Redwood	L-230	4000	Smith	L-230
Perryridge	L-260	1700	NULL	NULL

c) loan **natural inner join** borrower

3

branch_name	loan_number	amount	customer_name
Downtown	L-170	3000	Jones
Redwood	L-230	4000	Smith

d) loan **natural right outer join** borrower

3

branch_name	loan_number	amount	customer_name
Downtown	L-170	3000	Jones
Redwood	L-230	4000	Smith
NULL	L-155	NULL	Hayes

e) loan **full outer join** borrower **using** (loan-number)

3

branch_name	loan_number	amount	customer_name	loan_number
Downtown	L-170	3000	Jones	L-170
Redwood	L-230	4000	Smith	L-230
Perryridge	L-260	1700	NULL	~~L-260~~
NULL	~~L-155~~	NULL	Hayes	L-155