DBMS QUERIES

CREATE TABLE EMPLOYEE

(Fname VARCHAR(15) NOT NULL,

Minit CHAR,

Lname VARCHAR(15) NOT NULL,

Ssn CHAR(9) NOT NULL,

Bdate DATE,

Address VARCHAR(30),

Sex CHAR,

Salary DECIMAL(10,2),

Super_ssn CHAR(9),

Dno INT NOT NULL,

PRIMARY KEY (Ssn),+

FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn),

FOREIGN KEY (Dno) REFERENCES DEPARTMENT(Dnumber));

CREATE TABLE DEPARTMENT

(Dname VARCHAR(15) NOT NULL,

Dnumber INT NOT NULL,

Mgr_ssn CHAR(9) NOT NULL,

Mgr_start_date DATE,

PRIMARY KEY (Dnumber),

UNIQUE (Dname),

FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn));

CREATE TABLE DEPT_LOCATIONS

(Dnumber INT NOT NULL,

Dlocation VARCHAR(15) NOT NULL,

PRIMARY KEY (Dnumber, Dlocation),

FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber));

CREATE TABLE PROJECT

(Pname VARCHAR(15) NOT NULL,

Pnumber INT NOT NULL,

Plocation VARCHAR(15),

Dnum INT NOT NULL,

PRIMARY KEY (Pnumber),

UNIQUE (Pname),

FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber));

CREATE TABLE WORKS_ON

(Essn CHAR(9) NOT NULL,

Pno INT NOT NULL,

Hours DECIMAL(3,1) NOT NULL,

PRIMARY KEY (Essn, Pno),

FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),

FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber));

CREATE TABLE DEPENDENT

(Essn CHAR(9) NOT NULL,

Dependent_name VARCHAR(15) NOT NULL,

Sex CHAR,

Bdate DATE,

Relationship VARCHAR(8),

PRIMARY KEY (Essn, Dependent_name),

FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn));

Query 1. Retrieve the name and address of all employees who work for the 'Research' department.

Q1:

SELECT Fname, Lname, Address

FROM EMPLOYEE, DEPARTMENT

WHERE Dname='Research' AND Dnumber=Dno;

Query 2. For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birth date.

Q2:

SELECT Pnumber, Dnum, Lname, Address, Bdate

FROM PROJECT, DEPARTMENT, EMPLOYEE

WHERE Dnum=Dnumber AND Mgr_ssn=Ssn AND

Plocation='Stafford';

Query 8. For each employee, retrieve the employee's first and last name and the first and last name of his or her immediate supervisor.

Q8:

SELECT E.Fname, E.Lname, S.Fname, S.Lname

FROM EMPLOYEE AS E, EMPLOYEE AS S

WHERE E.Super_ssn=S.Ssn;

Queries 9 and 10. Select all EMPLOYEE Ssns (Q9) and all combinations of

EMPLOYEE Ssn and DEPARTMENT Dname (Q10) in the database.

Q9:

SELECT Ssn

FROM EMPLOYEE;

Q10:

SELECT Ssn, Dname

FROM EMPLOYEE, DEPARTMENT;

Query 11. Retrieve the salary of every employee (Q11) and all distinct salary

values (Q11A).

Q11: SELECT ALL Salary

FROM EMPLOYEE;

Q11A: SELECT DISTINCT Salary

FROM EMPLOYEE;

Query 4. Make a list of all project numbers for projects that involve an employee whose last name is 'Smith', either as a worker or as a manager of the department that controls the project.

Q4A:

(SELECT DISTINCT Pnumber

FROM PROJECT, DEPARTMENT, EMPLOYEE

WHERE Dnum=Dnumber AND Mgr_ssn=Ssn

AND Lname='Smith')

UNION

(SELECT DISTINCT Pnumber

FROM PROJECT, WORKS_ON, EMPLOYEE

WHERE Pnumber=Pno AND Essn=Ssn

AND Lname='Smith');

Query 12. Retrieve all employees whose address is in Houston, Texas.

Q12:

SELECT Fname, Lname

FROM EMPLOYEE

WHERE Address LIKE '%Houston,TX%';

Query 12A. Find all employees who were born during the 1950s.

Q12:

SELECT Fname, Lname

FROM EMPLOYEE

WHERE Bdate LIKE '_ _ 5 _ _ _ _ _ _ _';

Query 13. Show the resulting salaries if every employee working on the 'ProductX' project is given a 10 percent raise.

Q13:

SELECT E.Fname, E.Lname, 1.1 * E.Salary AS Increased_sal

FROM EMPLOYEE AS E, WORKS_ON AS W, PROJECT AS P

WHERE E.Ssn=W.Essn AND W.Pno=P.Pnumber AND

P.Pname='ProductX';

Query 14. Retrieve all employees in department 5 whose salary is between $30,000 and $40,000.

Q14:

```
SELECT  *
FROM EMPLOYEE
WHERE (Salary BETWEEN 30000 AND 40000) AND
Dno = 5;
```

The condition (Salary BETWEEN 30000 AND 40000) in Q14 is equivalent to the condition ((Salary >= 30000) AND (Salary <= 40000)).

Query 15. Retrieve a list of employees and the projects they are working on, ordered by department and, within each department, ordered alphabetically by last name, then first name.

Q15:

```
SELECT D.Dname, E.Lname, E.Fname, P.Pname
FROM DEPARTMENT D, EMPLOYEE E, WORKS_ON W,
PROJECT P
WHERE D.Dnumber= E.Dno AND E.Ssn= W.Essn AND
W.Pno= P.Pnumber
ORDER BY D.Dname, E.Lname, E.Fname;
```

SELECT <attribute list>

FROM <table list>

[WHERE <condition>]

[ORDER BY <attribute list>]

INSERT INTO EMPLOYEE (Fname, Lname, Dno, Ssn)

VALUES ('Richard', 'Marini', 4, '653298653');

```
INSERT INTO WORKS_ON_INFO ( Emp_name,
Proj_name,
Hours_per_week )
SELECT E.Lname, P.Pname, W.Hours
FROM PROJECT P, WORKS_ON W, EMPLOYEE E
WHERE P.Pnumber=W.Pno AND W.Essn=E.Ssn;
```

DELETE FROM EMPLOYEE

WHERE Lname='Brown';

```
DELETE FROM EMPLOYEE
WHERE Ssn='123456789';
U4C: DELETE FROM EMPLOYEE
WHERE Dno=5;
U4D: DELETE FROM EMPLOYEE;
```

UPDATE PROJECT

SET Plocation = 'Bellaire', Dnum = 5

WHERE Pnumber=10;

```
UPDATE EMPLOYEE
SET Salary = Salary * 1.1
WHERE Dno = 5;
```

Query 18. Retrieve the names of all employees who do not have supervisors.

Q18:

```
SELECT Fname, Lname
FROM EMPLOYEE
WHERE Super_ssn IS NULL;
```

SELECT DISTINCT Pnumber

FROM PROJECT

WHERE Pnumber IN

(SELECT Pnumber

FROM PROJECT, DEPARTMENT, EMPLOYEE

WHERE Dnum=Dnumber AND

Mgr_ssn=Ssn AND Lname='Smith')

OR

Pnumber IN

(SELECT Pno

FROM WORKS_ON, EMPLOYEE

WHERE Essn=Ssn AND Lname='Smith');

```
SELECT DISTINCT Essn
FROM WORKS_ON
```

```
WHERE (Pno, Hours) IN ( SELECT Pno, Hours
FROM WORKS_ON
WHERE Essn='123456789' );

SELECT Lname, Fname
FROM EMPLOYEE
WHERE Salary > ALL ( SELECT Salary
FROM EMPLOYEE
WHERE Dno=5 );
```

Query 16. Retrieve the name of each employee who has a dependent with the same first name and is the same sex as the employee.

Q16:

```
SELECT E.Fname, E.Lname
FROM EMPLOYEE AS E
WHERE E.Ssn IN ( SELECT Essn
FROM DEPENDENT AS D
WHERE E.Fname=D.Dependent_name
AND E.Sex=D.Sex );
```

Q16A: SELECT E.Fname, E.Lname

FROM EMPLOYEE AS E, DEPENDENT AS D

WHERE E.Ssn=D.Essn AND E.Sex=D.Sex

AND E.Fname=D.Dependent_name;

```
Q16B: SELECT E.Fname, E.Lname
FROM EMPLOYEE AS E
WHERE EXISTS ( SELECT *
FROM DEPENDENT AS D
WHERE E.Ssn=D.Essn AND E.Sex=D.Sex
AND E.Fname=D.Dependent_name);
```

Query 6. Retrieve the names of employees who have no dependents.

Q6:

```
 SELECT Fname, Lname
FROM EMPLOYEE
WHERE NOT EXISTS ( SELECT *
FROM DEPENDENT
WHERE Ssn=Essn );
```

Query 7. List the names of managers who have at least one dependent.

Q7:

```
SELECT Fname, Lname
FROM EMPLOYEE
WHERE EXISTS ( SELECT *
FROM DEPENDENT
WHERE Ssn=Essn )
AND
EXISTS ( SELECT *
FROM DEPARTMENT
WHERE Ssn=Mgr_ssn );
```

The query Q3: Retrieve the name of each employee who works on all the projects con trolled by department number 5 can be written using EXISTS and NOT EXISTS in SQL

Q3A:

```
SELECT Fname, Lname
FROM EMPLOYEE
WHERE NOT EXISTS ( ( SELECT Pnumber
FROM PROJECT
WHERE Dnum=5)
EXCEPT ( SELECT Pno
FROM WORKS_ON
WHERE Ssn=Essn) );
```

Q3B:

```
SELECT Lname, Fname
FROM EMPLOYEE
WHERE NOT EXISTS ( SELECT *
FROM WORKS_ON B
```

```
WHERE ( B.Pno IN ( SELECT Pnumber
FROM PROJECT
WHERE Dnum=5 )
AND
NOT EXISTS ( SELECT *
FROM WORKS_ON C
WHERE C.Essn=Ssn
AND C.Pno=B.Pno )));
```

Query 17. Retrieve the Social Security numbers of all employees who work on project numbers 1, 2, or 3.

Q17:

```
SELECT DISTINCT Essn
FROM WORKS_ON
WHERE Pno IN (1, 2, 3);
```

Query 19. Find the sum of the salaries of all employees, the maximum salary, the minimum salary, and the average salary.

Q19:

```
SELECT SUM (Salary), MAX (Salary), MIN (Salary),
AVG (Salary)
FROM EMPLOYEE;
```

Query 20. Find the sum of the salaries of all employees of the 'Research' department, as well as the maximum salary, the minimum salary, and the aver age salary in this department.

Q20:

```
SELECT SUM (Salary), MAX (Salary), MIN (Salary),
AVG (Salary)
FROM (EMPLOYEE JOIN DEPARTMENT ON
Dno=Dnumber)
WHERE Dname='Research';
```

Queries 21 and 22. Retrieve the total number of employees in the company (Q21) and the number of employees in the 'Research' department (Q22).

Q21:

```
SELECT COUNT (*)
FROM EMPLOYEE;
```

Q22:

```
SELECT COUNT (*)
FROM EMPLOYEE, DEPARTMENT
WHERE DNO=DNUMBER AND DNAME='Research';
```

Here the asterisk (*) refers to the rows (tuples), so COUNT (*) returns the number of

rows in the result of the query. We may also use the COUNT function to count values

in a column rather than tuples, as in the next example.

Query 23. Count the number of distinct salary values in the database.

Q23:

```
SELECT COUNT (DISTINCT Salary)
FROM EMPLOYEE;
```

Query 24. For each department, retrieve the department number, the number of employees in the department, and their average salary.

Q24:

```
SELECT Dno, COUNT (*), AVG (Salary)
FROM EMPLOYEE
GROUP BY Dno;
```

Query 25. For each project, retrieve the project number, the project name, and the number of employees who work on that project.

Q25:

```
SELECT Pnumber, Pname, COUNT (*)
FROM PROJECT, WORKS_ON
WHERE Pnumber=Pno
GROUP BY Pnumber, Pname;
```

Query 26. For each project on which more than two employees work, retrieve the project number, the project name, and the number of employees who work on the project.

Q26:

```
SELECT Pnumber, Pname, COUNT (*)
FROM PROJECT, WORKS_ON
WHERE Pnumber=Pno
GROUP BY Pnumber, Pname
HAVING COUNT (*) > 2;
```

Query 27. For each project, retrieve the project number, the project name, and the number of employees from department 5 who work on the project.

Q27:

```
SELECT Pnumber, Pname, COUNT (*)
FROM PROJECT, WORKS_ON, EMPLOYEE
WHERE Pnumber=Pno AND Ssn=Essn AND Dno=5
GROUP BY Pnumber, Pname;
```

Query 28. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than $40,000.

Q28:

```
SELECT Dnumber, COUNT (*)
FROM DEPARTMENT, EMPLOYEE
WHERE Dnumber=Dno AND Salary>40000 AND
( SELECT Dno
FROM EMPLOYEE
GROUP BY Dno
HAVING COUNT (*) > 5);
```

SELECT <attribute and function list>

FROM <table list>

[WHERE <condition>]

[GROUP BY <grouping attribute(s)>]

[HAVING <group condition>]

[ORDER BY <attribute list>];

Relational algebra

Query 1. Retrieve the name and address of all employees who work for the 'Research' department

RESEARCH_DEPT ← σDname='Research'(DEPARTMENT)

RESEARCH_EMPS ← (RESEARCH_DEPT Dnumber=DnoEMPLOYEE)

RESULT ← πFname, Lname, Address(RESEARCH_EMPS)

As a single in-line expression, this query becomes:

πFname, Lname, Address (σDname='Research'(DEPARTMENT Dnumber=Dno(EMPLOYEE))

Tuple Relational algebra

Query 1. List the name and address of all employees who work for the 'Research' department.

Q1: {t.Fname, t.Lname, t.Address | EMPLOYEE(t) AND (∃d)(DEPARTMENT(d)

AND d.Dname='Research' AND d.Dnumber=t.Dno)}

Query 2. For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, birth date, and address.

Q2:

{p.Pnumber, p.Dnum, m.Lname, m.Bdate, m.Address | PROJECT(p) AND

EMPLOYEE(m) AND p.Plocation='Stafford' AND ((∃d)(DEPARTMENT(d)

AND p.Dnum=d.Dnumber AND d.Mgr_ssn=m.Ssn))}

Query 3 . List the name of each employee who works on some project con trolled by department number 5. This is a variation of Q3 in which all changed to some. In this case we need two join conditions and two existential quantifiers.

Q0 :

{e.Lname, e.Fname | EMPLOYEE(e) AND ((∃x)(∃w)(PROJECT(x) AND

WORKS_ON(w) AND x.Dnum=5 AND w.Essn=e.Ssn AND x.Pnumber=w.Pno))}

Query 4. Make a list of project numbers for projects that involve an employee whose last name is 'Smith', either as a worker or as manager of the controlling department for the project.

Q4:

{ p.Pnumber | PROJECT(p) AND (((∃e)(∃w)(EMPLOYEE(e)

AND WORKS_ON(w) AND w.Pno=p.Pnumber

AND e.Lname='Smith' AND e.Ssn=w.Essn))

OR

((∃m)(∃d)(EMPLOYEE(m) AND DEPARTMENT(d)

AND p.Dnum=d.Dnumber AND d.Mgr_ssn=m.Ssn

AND m.Lname='Smith'))))}

Query 3. List the names of employees who work on all the projects controlled by department number 5. One way to specify this query is to use the universal quantifier as shown:

Q3:

{e.Lname, e.Fname | EMPLOYEE(e) AND ((∀x)(NOT(PROJECT(x)) OR NOT

(x.Dnum=5) OR ((∃w)(WORKS_ON(w) AND w.Essn=e.Ssn AND

x.Pnumber=w.Pno)))))}

Q3A:

{e.Lname, e.Fname | EMPLOYEE(e) AND (NOT (∃x) (PROJECT(x) AND

(x.Dnum=5) AND (NOT (∃w)(WORKS_ON(w) AND w.Essn=e.Ssn

AND x.Pnumber=w.Pno)))))}

We now give some additional examples of queries that use quantifiers.

Query 6. List the names of employees who have no dependents.

Q6:

{e.Fname, e.Lname | EMPLOYEE(e) AND (NOT (∃d)(DEPENDENT(d)

AND e.Ssn=d.Essn))}

Using the general transformation rule, we can rephrase Q6 as follows:

Q6A: {e.Fname, e.Lname | EMPLOYEE(e) AND ((∀d)(NOT(DEPENDENT(d))

OR NOT(e.Ssn=d.Essn)))}

Query 7. List the names of managers who have at least one dependent.

Q7: {e.Fname, e.Lname | EMPLOYEE(e) AND ((∃d)(∃ρ)(DEPARTMENT(d)

AND DEPENDENT(ρ) AND e.Ssn=d.Mgr_ssn AND ρ.Essn=e.Ssn))}

Domain Relational algebra

Query 0. List the birth date and address of the employee whose name is 'John B. Smith'.

Q0:

{u, v | (∃q) (∃r) (∃s) (∃t) (∃w) (∃x) (∃y) (∃z) (EMPLOYEE(qrstuvwxyz) AND q='John' AND r='B' AND s='Smith')}

Query 1. Retrieve the name and address of all employees who work for the 'Research' department.

Q1:

{q, s, v | (∃z) (∃l) (∃m) (EMPLOYEE(qrstuvwxyz) AND DEPARTMENT(lmno) AND l='Research' AND m=z)}

Query 2. For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, birth date, and address.

Q2:

{i, k, s, u, v | (∃j)(∃m)(∃n)(∃t)(PROJECT(hijk) AND EMPLOYEE(qrstuvwxyz) AND DEPARTMENT(lmno) AND k=m AND

n=t AND j='Stafford')}

Query 6. List the names of employees who have no dependents.

Q6:

{q, s | (∃t)(EMPLOYEE(qrstuvwxyz) AND

(NOT(∃l)(DEPENDENT(lmnop) AND t=l)))}

Q6 can be restated using universal quantifiers instead of the existential quantifiers,

as shown in Q6A:

Q6A:

{q, s | (∃t)(EMPLOYEE(qrstuvwxyz) AND

((∀l)(NOT(DEPENDENT(lmnop)) OR NOT(t=l)))))}

Query 7. List the names of managers who have at least one dependent.

Q7:

{s, q | (∃t)(∃j)(∃l)(EMPLOYEE(qrstuvwxyz) AND DEPARTMENT(hijk)

AND DEPENDENT(lmnop) AND t=j AND l=t)}