Database Systems

Chapter # 7

More SQL: Complex Queries, Triggers, Views, and Schema Modification

Chapter Outlines

- **More Complex SQL Retrieval Queries**
- **Views (Virtual Tables) in SQL**
- **Schema Change Statements in SQL**

Joined Tables in SQL and Outer Joins

- SQL joins are used to combine two or more tables.
- The joining of two or more tables is based on common field between them.
- Syntax:

  select col1,col2,....

  from table1 JOIN table2 ON (common columns in both tables)

  where condition

Types of Joins

- Cartesian Product
- Equi JOIN/ Inner Join/ Join
- Left outer Join/ Left Join
- Right Outer Join/ Right Join
- Self Join

Basic SQL Queries: The SELECT-FROM-WHERE Structure with join

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

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Example

- Retrieves the name and addre of every employee who works for the 'Research' department.

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

```
Q1A:    SELECT    Fname, Lname, Address
        FROM      (EMPLOYEE JOIN DEPARTMENT ON Dno = Dnumber)
        WHERE     Dname = 'Research';
```

Basic SQL Queries: The SELECT-FROM-WHERE Structure with join

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;
```

Fname	Lname	Address
John	Smith	731 Fondren, Houston, TX
Franklin	Wong	638 Voss, Houston, TX
Ramesh	Narayan	975 Fire Oak, Humble, TX
Joyce	English	5631 Rice, Houston, TX

Basic SQL Queries: The SELECT-FROM-WHERE Structure with join

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.

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Basic SQL Queries: The SELECT-FROM-WHERE Structure with join

(c)

Pnumber	Dnum	Lname	Address	Bdate
10	4	Wallace	291Berry, Bellaire, TX	1941-06-20
30	4	Wallace	291Berry, Bellaire, TX	1941-06-20

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 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

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Ambiguous Attribute Names, Aliasing, Renaming, and Tuple Variables

- Same name can be used for two (or more) attributes in different relations
  - As long as the attributes are in different relations
  - Must **qualify** the attribute name with the relation's name to prevent ambiguity

```
Q1A:  SELECT   Fname, EMPLOYEE.Name, Address
      FROM     EMPLOYEE, DEPARTMENT
      WHERE    DEPARTMENT.Name = 'Research' AND
               DEPARTMENT.Dnumber = EMPLOYEE.Dnumber;
```

Fully qualified attribute names can be used for clarity even if there is no ambiguity in attribute names. Q1 can be rewritten as Q1′ below with fully qualified attribute names. We can also rename the table names to shorter names by creating an *alias* for each table name to avoid repeated typing of long table names (see Q8 below).

```
Q1′:  SELECT   EMPLOYEE.Fname, EMPLOYEE.LName,
               EMPLOYEE.Address
      FROM     EMPLOYEE, DEPARTMENT
      WHERE    DEPARTMENT.DName = 'Research' AND
               DEPARTMENT.Dnumber = EMPLOYEE.Dno;
```

SELF JOIN

Query 3. For each employee, retrieve the employee's first and last name and the first and last name of his or her immediate supervisor.

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

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.

```
SELECT E.Fname, E.Lname, S.Fname, S.Lname FROM EMPLOYEE E, EMPLOYEE S
WHERE E.Super_ssn = S.Ssn;
```

(d)

E.Fname	E.Lname	S.Fname	S.Lname
John	Smith	Franklin	Wong
Franklin	Wong	James	Borg
Alicia	Zelaya	Jennifer	Wallace
Jennifer	Wallace	James	Borg
Ramesh	Narayan	Franklin	Wong
Joyce	English	Franklin	Wong
Ahmad	Jabbar	Jennifer	Wallace

ORDER BY

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

```
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;
```

ORDER BY D.Dname DESC, E.Lname ASC, E.Fname ASC

Figure 5.6

One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

NATURAL JOIN

- In a NATURAL JOIN on two relations R and S, no join condition is specified;
- An implicit EQUIJOIN condition for each pair of attributes with the same name from R and S is created.
- Each such pair of attributes is included only once in the resulting relation.

Q1B: SELECT Fname, Lname, Address, D.Dnumber AS Dno

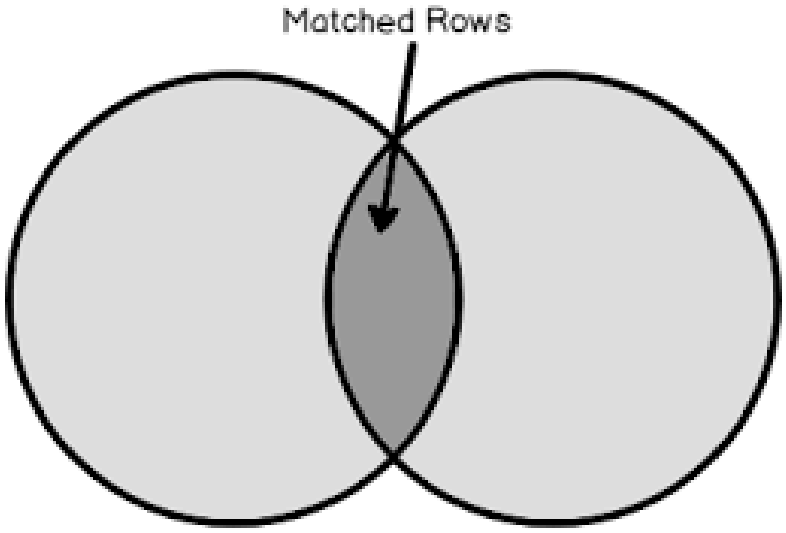
FROM (EMPLOYEE E NATURAL JOIN DEPARTMENT D

WHERE Dname = ‘Research’;

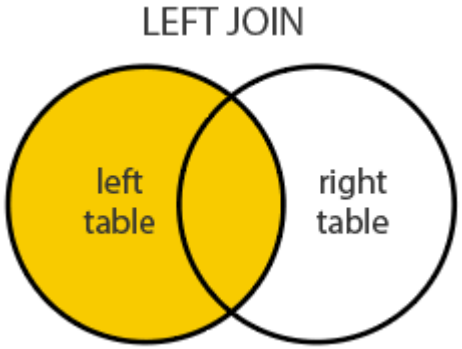
Implicit join condition: EMPLOYEE.Dno = DEPT.Dno

Inner join/Join

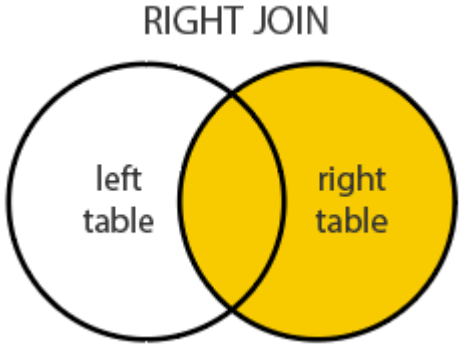
- Default type of join in a joined table
- Tuple is included in the result only if a matching tuple exists in the other relation.



Outer JOIN



- LEFT OUTER JOIN
  - Every tuple in left table must appear in result
  - No matching tuple: attributes of right table are Padded with NULL values
- RIGHT OUTER JOIN
  - Every tuple in right table must appear in result
  - No matching tuple: attributes of left table are Padded with NULL values



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.

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON **PROJECT**

```
Q8B:    SELECT    E.Lname AS Employee_name,
                  S.Lname AS Supervisor_name
        FROM      (EMPLOYEE AS E LEFT OUTER JOIN EMPLOYEE AS S
                  ON E.Super_ssn = S.Ssn);
```

Comparison Operator: IS or IS NOT

- **IS or IS NOT:** check whether an attribute value is NULL or NOT NULL

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

```
Q18:    SELECT    Fname, Lname
        FROM      EMPLOYEE
        WHERE     Super_ssn IS NULL;
```

Nested Queries/ Subqueries

Subqueries:

- can be placed in a number of SQL clauses like
  - WHERE clause
  - FROM clause
  - HAVING clause.
- can be used with SELECT, UPDATE, INSERT, DELETE statements along with the operators like =, <, >, >=, <=, IN, BETWEEN, etc.
- A subquery is a query within another query. The outer query is known as the main query, and the inner query is known as a subquery.
- Subqueries are on the right side of the comparison operator and enclosed in parentheses.
- **Note: In the Subquery, ORDER BY command cannot be used. But GROUP BY command can be used to perform the same function as ORDER BY command.**

Sub Queries: Syntax

SELECT column_name

FROM table_name

WHERE column_name expression operator

(SELECT column_name from table_name WHERE ...
);

Demo Database: Employee Table

ID	NAME	AGE	ADDRESS	SALARY
1	John	20	US	2000.00
2	Stephan	26	Dubai	1500.00
3	David	27	Bangkok	2000.00
4	Alina	29	UK	6500.00
5	Kathrin	34	Bangalore	8500.00
6	Harry	42	China	4500.00
7	Jackson	25	Mizoram	10000.00

Display the name of the employee who is getting max salary.

Display NAME, LOCATION, PHONE_NUMBER of the students whose section is A.

Students Relation

NAME	ROLL_NO	LOCATION	PHONE_NUMBER
Ram	101	Chennai	9988775566
Raj	102	Coimbatore	8877665544
Sasi	103	Madurai	7766553344
Ravi	104	Salem	8989898989
Sumathi	105	Kanchipuram	8989856868

Section Relation

NAME	ROLL_NO	SECTION
Ravi	104	A
Sumathi	105	B
Raj	102	A

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

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Select the Essns of all employees who work the same (project, hours) combination on some project that employee 'John Smith' (whose Ssn = '123456789')

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Write a query that returns the names of employees whose salary is greater than the salary of all the employees in department 5

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Type of Subqueries

- **Single row subquery :** Returns zero or one row. Uses Aggregate functions (max, min, avg, count,sum)
- **Multiple row subquery :** Returns one or more rows.
- **Multiple column subqueries :** Returns one or more columns.
- **Correlated subqueries :** Reference one or more columns in the outer SQL statement. The subquery is known as a correlated subquery because the subquery is related to the outer SQL statement.

Equal to and In

- If a nested query returns a single attribute and a single tuple: **use = instead of IN for the comparison operator.**
- In general, the nested query will return a table (relation), which is a set or multiset of tuples.

ALL

- ALL: used to compare a value to a list or subquery. It must be preceded by =, !=, >, <, <=, >= and followed by a list or subquery.
- "x = ALL (...)": The value must match all the values in the list to evaluate to TRUE.
- "x != ALL (...)": The value must not match any values in the list to evaluate to TRUE.
- "x > ALL (...)": The value must be greater than the biggest value in the list to evaluate to TRUE.
- "x < ALL (...)": The value must be smaller than the smallest value in the list to evaluate to TRUE.
- "x >= ALL (...)": The value must be greater than or equal to the biggest value in the list to evaluate to TRUE.
- "x <= ALL (...)": The value must be smaller than or equal to the smallest value in the list to evaluate to TRUE.

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-1980 00:00:00	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-1981 00:00:00	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-1981 00:00:00	1250	500	30
7566	JONES	MANAGER	7839	02-APR-1981 00:00:00	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-1981 00:00:00	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-1981 00:00:00	2850		30
7782	CLARK	MANAGER	7839	09-JUN-1981 00:00:00	2450		10
7788	SCOTT	ANALYST	7566	19-APR-1987 00:00:00	3000		20
7839	KING	PRESIDENT		17-NOV-1981 00:00:00	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-1981 00:00:00	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-1987 00:00:00	1100		20
7900	JAMES	CLERK	7698	03-DEC-1981 00:00:00	950		30
7902	FORD	ANALYST	7566	03-DEC-1981 00:00:00	3000		20
7934	MILLER	CLERK	7782	23-JAN-1982 00:00:00	1300		10

SELECT empno, sal FROM emp WHERE sal > ALL (2000, 3000, 4000);

EMPNO

Output : 7839 5000

SQL> -- Transformed to equivalent statement without ALL. SELECT empno, sal FROM emp WHERE sal > 2000 AND sal > 3000 AND sal > 4000;

Output: 7839 5000

```
SELECT e1.empno, e1.sal
FROM   emp e1
WHERE  e1.sal > ALL
(SELECT e2.sal
 FROM   emp e2
 WHERE  e2.deptno = 20);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-1980 00:00:00	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-1981 00:00:00	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-1981 00:00:00	1250	500	30
7566	JONES	MANAGER	7839	02-APR-1981 00:00:00	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-1981 00:00:00	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-1981 00:00:00	2850		30
7782	CLARK	MANAGER	7839	09-JUN-1981 00:00:00	2450		10
7788	SCOTT	ANALYST	7566	19-APR-1987 00:00:00	3000		20
7839	KING	PRESIDENT		17-NOV-1981 00:00:00	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-1981 00:00:00	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-1987 00:00:00	1100		20
7900	JAMES	CLERK	7698	03-DEC-1981 00:00:00	950		30
7902	FORD	ANALYST	7566	03-DEC-1981 00:00:00	3000		20
7934	MILLER	CLERK	7782	23-JAN-1982 00:00:00	1300		10

EMPNO	SAL
7839	5000

ANY

- The ANY comparison condition is used to compare a value to a list or subquery. It must be preceded by =, !=, >, <, <=, >= and followed by a list or subquery.
- "x = ANY (...)": The value must match one or more values in the list to evaluate to TRUE.
- "x != ANY (...)": The value must not match one or more values in the list to evaluate to TRUE.
- "x > ANY (...)": The value must be greater than the smallest value in the list to evaluate to TRUE.
- "x < ANY (...)": The value must be smaller than the biggest value in the list to evaluate to TRUE.
- "x >= ANY (...)": The value must be greater than or equal to the smallest value in the list to evaluate to TRUE.
- "x <= ANY (...)": The value must be smaller than or equal to the biggest value in the list to evaluate to TRUE.

```
SELECT empno, sal
FROM   emp
WHERE
sal > ANY (2000, 3000, 4000);
```

EMPNO	SAL
7566	2975
7698	2850
7782	2450
7788	3000
7839	5000
7902	3000

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-1980 00:00:00	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-1981 00:00:00	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-1981 00:00:00	1250	500	30
7566	JONES	MANAGER	7839	02-APR-1981 00:00:00	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-1981 00:00:00	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-1981 00:00:00	2850		30
7782	CLARK	MANAGER	7839	09-JUN-1981 00:00:00	2450		10
7788	SCOTT	ANALYST	7566	19-APR-1987 00:00:00	3000		20
7839	KING	PRESIDENT		17-NOV-1981 00:00:00	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-1981 00:00:00	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-1987 00:00:00	1100		20
7900	JAMES	CLERK	7698	03-DEC-1981 00:00:00	950		30
7902	FORD	ANALYST	7566	03-DEC-1981 00:00:00	3000		20
7934	MILLER	CLERK	7782	23-JAN-1982 00:00:00	1300		10

```
SELECT e1.empno, e1.sal
FROM   emp e1
WHERE  e1.sal > ANY (SELECT e2.sal
FROM   emp e2
WHERE  e2.deptno = 10);
```

EMPNO	SAL
7839	5000
7902	3000
7788	3000
7566	2975
7698	2850
7782	2450
7499	1600
7844	1500

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-1980 00:00:00	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-1981 00:00:00	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-1981 00:00:00	1250	500	30
7566	JONES	MANAGER	7839	02-APR-1981 00:00:00	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-1981 00:00:00	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-1981 00:00:00	2850		30
7782	CLARK	MANAGER	7839	09-JUN-1981 00:00:00	2450		10
7788	SCOTT	ANALYST	7566	19-APR-1987 00:00:00	3000		20
7839	KING	PRESIDENT		17-NOV-1981 00:00:00	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-1981 00:00:00	1500	0	30
7876	ADAMS	CLERK	7788	23-MAY-1987 00:00:00	1100		20
7900	JAMES	CLERK	7698	03-DEC-1981 00:00:00	950		30
7902	FORD	ANALYST	7566	03-DEC-1981 00:00:00	3000		20
7934	MILLER	CLERK	7782	23-JAN-1982 00:00:00	1300		10

- SELECT empno, sal
- FROM emp
- WHERE sal > 2000 OR sal > 3000 OR sal > 4000;

- EMPNO SAL
- ---------- ----------
- 7566 2975
- 7698 2850
- 7782 2450
- 7788 3000
- 7839 5000
- 7902 3000

Correlated Nested Queries

- Whenever a condition in the WHERE clause of a nested query references some attribute of a relation declared in the outer query, the two queries are said to be correlated.
- Evaluated as a top-down approach

Exists and Not Exists

- The EXISTS function: used to check whether the result of a nested query is empty (contains no tuples) or not.
- The result of EXISTS is a Boolean value TRUE if the nested query result contains at least one tuple, or FALSE if the nested query result contains no tuples.

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

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Retrieve the names of employees who have no dependents.

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

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Aggregate Functions

- Used to summarize information from multiple tuples into a single-tuple summary
- Grouping is used to create subgroups of tuples before summarization
- Built-in aggregate functions: COUNT, SUM, MAX, MIN, COUNT and AVG.
- These functions can be used in the SELECT clause or in a HAVING clause.

Aggregate Functions: Example

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

```
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 average salary in this department.**

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland

- 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).

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

Q21: SELECT COUNT (*) FROM EMPLOYEE

Q22: SELECT COUNT (*)
FROM EMPLOYEE, DEPARTMENT
WHERE DNO = DNUMBER AND DNAME = 'Research';

Counting Number of Distinct Salary

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

```
Q23: SELECT COUNT (DISTINCT Salary)
FROM EMPLOYEE;
```

- Q5: Retrieve the names of all employees who have two or more dependents

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

```
SELECT Lname, Fname
FROM EMPLOYEE
WHERE ( SELECT COUNT (*)
FROM DEPENDENT
WHERE Ssn = Essn ) > = 2;
```

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Grouping: The GROUP BY and HAVING Clauses

- GROUP BY: specifies the grouping attributes
  - For example, find the average salary of employees in each department or the number of employees who work on each project.
- Grouping attributes must appear in the SELECT clause, so that the value resulting from applying each aggregate function to a group of tuples appears along with the value of the grouping attribute(s).

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

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

(a)

Fname	Minit	Lname	Ssn	...	Salary	Super_ssn	Dno
John	B	Smith	123456789		30000	333445555	5
Franklin	T	Wong	333445555		40000	888665555	5
Ramesh	K	Narayan	666884444		38000	333445555	5
Joyce	A	English	453453453	...	25000	333445555	5
Alicia	J	Zelaya	999887777		25000	987654321	4
Jennifer	S	Wallace	987654321		43000	888665555	4
Ahmad	V	Jabbar	987987987		25000	987654321	4
James	E	Bong	888665555		55000	NULL	1

Grouping EMPLOYEE tuples by the value of Dno

Dno	Count (*)	Avg (Salary)
5	4	33250
4	3	31000
1	1	55000

Result of Q24

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.

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

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

HAVING clause

- HAVING : Provides a condition to select or reject an entire group:
  - Only the groups that satisfy the condition are retrieved in the result of the query.

- 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.

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

```
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;

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

EXPANDED Block Structure of SQL Queries

SELECT <attribute and function list>

FROM <table list>

[WHERE <condition>]

[GROUP BY <grouping attribute(s)>]

[HAVING <group condition>]

[ORDER BY <attribute list>];

Views (Virtual Tables) in SQL

- **Virtual table :** A view does not necessarily exist in physical form;
- A single table that is derived from other tables
- Other tables can be base tables or previously defined views.
- **Syntax:**
  - CREATE VIEW view_name AS
  - SELECT column1, column2, ...
  - FROM table_name
  - WHERE condition;

Specification of Views in SQL

VIEW WITH NO NEW ATTRIBUTES
V1: CREATE VIEW **WORKS_ON1**
AS SELECT Fname, Lname, Pname, Hours
FROM EMPLOYEE, PROJECT, WORKS_ON
WHERE Ssn = Essn AND Pno = Pnumber;

VIEW WITH NEW ATTRIBUTES

V2: CREATE VIEW DEPT_INFO(Dept_name, No_of_emps, Total_sal)
AS SELECT Dname, COUNT (*), SUM (Salary)
FROM DEPARTMENT, EMPLOYEE
WHERE Dnumber = Dno
GROUP BY Dname;

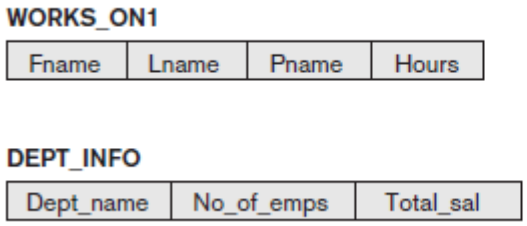


Figure 7.2
Two views specified on the database schema of Figure 5.5.

Specification of Views in SQL

- **A view is always up-to-date:** if the tuples are modified in the base tables on which the view is defined, the view must automatically reflect these changes.
- **DROP VIEW command:**
- **Syntax:**
  - DROP VIEW WORKS_ON1;

View Implementation, View Update, and Inline Views

- Efficiently implement a view for efficient querying is complex.
- Two main approaches have been suggested.
- **Strategy # 1:** (query modification) involves transforming the view query (submitted by the user) into a query on the underlying base tables.
- For example, the query QV1 would be automatically modified to the following query by the DBMS:

```
SELECT Fname, Lname
FROM WORKS_ON1
WHERE Pname = 'ProductX';
```

```
SELECT Fname, Lname
FROM EMPLOYEE, PROJECT, WORKS_ON
WHERE Ssn = Essn AND Pno = Pnumber
AND Pname = 'ProductX';
```

- **Disadvantage of this approach:** inefficient for views defined via complex queries that are time-consuming to execute

View Implementation, View Update, and Inline Views

- **Strategy # 2 (view materialization):** involves physically creating a temporary or permanent view table
- In order to keep the view up-to-date an efficient strategy for automatically updating the view table when the base tables are updated must be developed.

Create materialized view

```
CREATE TABLE STUDENTS (Roll_no int primary key, name
varchar2(20) not null, address varchar2(20) not null);
        insert into students values(1,'Ali', 'Hyderaabad');
        insert into students values(2,'Ahmed', 'Hyderaabad');
        insert into students values(3,'Azam', 'Hyderaabad');
        insert into students values(4,'Asif', 'Hyderaabad');
```

```
CREATE MATERIALIZED VIEW LOG ON students ;
create MATERIALIZED view s1
refresh fast on commit
as
select * from students;
select * from s1;
insert into students values(5,'Aslam','karachi');
```

Why Materialized Views are used

- To increase the speed of queries on very large databases.
- Queries to large databases often involve joins between tables, aggregations such as SUM, or both.
- These operations are expensive in terms of time and processing power.
- The type of materialized view is created determines how the materialized view is refreshed and used by query rewrite.

Why Materialized Views are used

- The query optimizer automatically recognizes when an existing materialized view can and should be used to satisfy a request.
- It then transparently rewrites the request to use the materialized view.
- Queries go directly to the materialized view and not to the underlying detail tables. In general, rewriting queries to use materialized views rather than detail tables improves response time.

View Update

- **Incremental updates for Materialized View.**
  - Determine what new tuples must be inserted, deleted, or modified in a materialized view table when a database update is applied to one of the defining base tables.
- Different strategies for updating materialized views are possible:
- **immediate update strategy:** updates a view as soon as the base tables are changed;
- **lazy update strategy:** updates the view when needed by a view query;
- **periodic update strategy:** updates the view periodically

View update

- issuing an INSERT, DELETE, or UPDATE command on a view table **is in many cases not possible.**
- view involving joins: an update operation may be mapped to update operations on the underlying base relations in *multiple ways*.
- Hence, it is often not possible for the DBMS to determine which of the updates is intended.

WITH CHECK OPTION

- WITH CHECK OPTION: added at the end of the view definition if a view is to be updated by INSERT, DELETE, or UPDATE statements.
- This allows the system to reject operations that violate the SQL rules for view updates.

Example

- create table test (salary int not null);
- insert into test values(200000);
- insert into test values(300000);
- insert into test values(300000);
- insert into test values(400000);
- insert into test values(500000);

- create or replace view test_view1 as select * from test where salary=300000 with check option;
- update test_view1 set salary=800000 where salary = 300000; --will not be updated and changes will not be reflected to base table.
- insert into test_view1 values(800000); --will not be inserted and changes will not be reflected to base table.
- delete from test_view1 where salary=300000; successfully deleted.

In-Line Views

- The subquery specified in the FROM clause of a query is called an inline view.
- An inline view can replace a table in a query, it is also called a derived table. Sometimes, you may hear the term subselect, which is the same meaning as the inline view.
- You often use the inline view in Oracle to simplify complex queries by eliminating join operations or condensing separate queries into a single query.

In-line view

Syntax:

```
SELECT
   column_list
FROM
   (
      SELECT
         *
      FROM
         table_name
   ) ;
```

Example of Inline Views

```
SELECT
   *
FROM
   (
      SELECT
         product_id,
         product_name,
         list_price
      FROM
         products
      ORDER BY
         list_price DESC
   )
WHERE
   ROWNUM <= 10;
```

Views as Authorization Mechanisms

- Views can be used to hide certain attributes or tuples from unauthorized users.
- User is only allowed to see employee information for employees who work for department 5;
  - CREATE VIEW DEPT5EMP AS SELECT * FROM EMPLOYEE WHERE Dno = 5;

Views as Authorization Mechanisms

- A view can restrict a user to only see certain columns;
  - for example, only the first name, last name, and address of an employee may be visible as follows:

CREATE VIEW BASIC_EMP_DATA AS SELECT Fname, Lname, Address FROM EMPLOYEE;

Schema Change Statements in SQL

- **DROP TABLE table_name;**
- **SQL TRUNCATE TABLE:** used to delete the data inside a table, but not the table itself.
  - TRUNCATE TABLE table_name;

SQL ALTER TABLE Statement

ADD COLUMNS

ALTER TABLE table_name

ADD column_name datatype;

Column Modification

ALTER TABLE table_name

MODIFY column_name datatype;

DROP COLUMNS AND CONSTRAINTS

- **Drop Column**

  ALTER TABLE table_name

  DROP COLUMN column_name;

- **Drop constraints**
  - ALTER TABLE table_name
  - DROP CONSTRAINT Constraint_name;

Add constraints

- **Unique Constraint**

  ALTER TABLE table_name

  ADD CONSTRAINT constraint_name UNIQUE(column1, column2...);

- **Check Constraint**

  ALTER TABLE table_name

  ADD CONSTRAINT constraint_name CHECK (CONDITION);

- **Primary Key**

  ALTER TABLE table_name

  ADD CONSTRAINT constraint_name Primary Key (COL1,COL2,…);

Add constraints

- **FOREIGN KEY Key**

  ALTER TABLE table_name

  ADD CONSTRAINT constraint_name FOREIGN KEY (COLUMN_NAME) REFFERENCES TABLE_NAME (COL1);