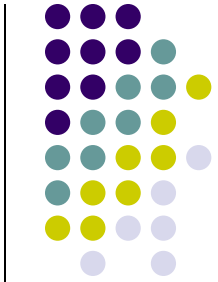
Contents

Retrieval Queries in SQL



1. Simple SQL Queries
2. Aliases
3. *
4. DISTINCT
5. Empty WHERE-clause
6. NULLS IN SQL
7. AGGREGATE FUNCTIONS
8. SUBSTRING COMPARISON
9. ARITHMETIC OPERATIONS
10. GROUPING
11. THE HAVING-CLAUSE
12. ORDER BY
13. EXPLICIT SETS

Relational Database Schema

EMPLOYEE

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO

DEPARTMENT

DNAME	DNUMBER	MGRSSN	MGRSTARTDATE

DEPT_LOCATIONS

DNUMBER	DLOCATION

PROJECT

PNAME	PNUMBER	PLOCATION	DNUM

WORKS_ON

ESSN	PNO	HOURS

DEPENDENT

ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP

Populated Database

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	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	1968-07-19	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

DEPT_LOCATIONS	DNUMBER	DLOCATION
	1	Houston
	4	Stafford
	5	Bellaire
	5	Sugarland
	5	Houston

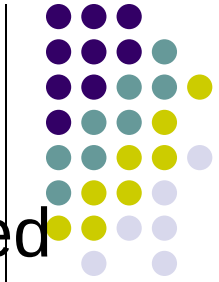
DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
	Research	5	333445555	1988-05-22
	Administration	4	987654321	1995-01-01
	Headquarters	1	888665555	1981-06-19

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

Retrieval Queries in SQL

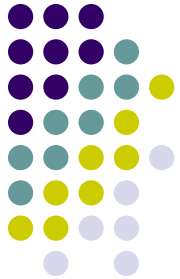


- Basic form of the SQL SELECT statement is called a *mapping* or a SELECT-FROM-WHERE *block*

SELECT <attribute list>
FROM <table list>
WHERE <condition>

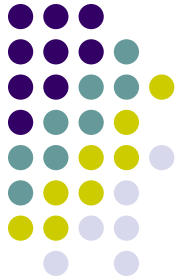
  - <attribute list> is a list of attribute names whose values are to be retrieved by the query
  - <table list> is a list of the relation names required to process the query
  - <condition> is a conditional (Boolean) expression that identifies the tuples to be retrieved by the query

Simple SQL Queries



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

Simple SQL Queries

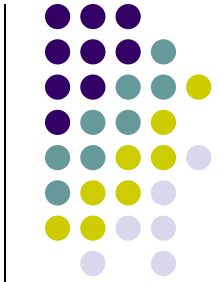


Query 0: Retrieve the birth date and address of the employee whose name is 'John B. Smith'.

Q0:

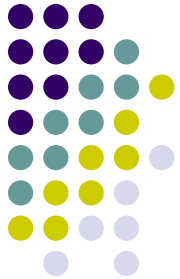
```
SELECT    BDATE, ADDRESS
FROM      EMPLOYEE
WHERE     FNAME='John' AND MINIT='B'
          AND LNAME='Smith'
```

Simple SQL Queries



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

Simple SQL Queries

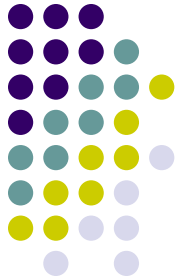


- Query 1: Retrieve the name and address of all employees who work for the 'Research' department.

```
Q1: SELECT    FNAME, LNAME, ADDRESS
    FROM      EMPLOYEE, DEPARTMENT
    WHERE     DNAME='Research' AND
              DNUMBER=DNO
```

(DNUMBER=DNO) is a join condition

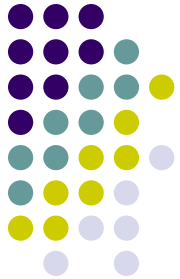
Simple SQL Queries



- 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 birthdate.

  - In Q2, there are two join conditions
  - The join condition DNUM=DNUMBER relates a project to its controlling department
  - The join condition MGRSSN=SSN relates the controlling department to the employee who manages that department

Simple SQL Queries

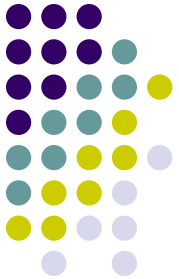


- Query 2: For every project located in 'Stafford', list the project number, the controlling department number, and the department manager's last name, address, and birthdate.

```
SELECT   PNUMBER, DNUM, LNAME, BDATE, ADDRESS
FROM     PROJECT, DEPARTMENT, EMPLOYEE
WHERE    PLOCATION='Stafford‘ AND DNUM=DNUMBER
         AND MGRSSN=SSN
```

Contents

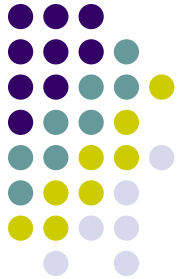
Retrieval Queries in SQL



1. Simple SQL Queries
2. Aliases
3. *
4. DISTINCT
5. Empty WHERE-clause

6. NULLS IN SQL
7. AGGREGATE FUNCTIONS
8. SUBSTRING COMPARISON
9. ARITHMETIC OPERATIONS
10. GROUPING
11. THE HAVING-CLAUSE
12. ORDER BY
13. EXPLICIT SETS

Aliases, * and DISTINCT, Empty WHERE-clause



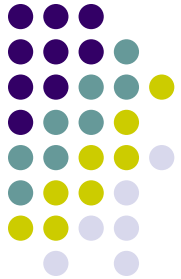
- In SQL, we can use the same name for two (or more) attributes as long as the attributes are in *different relations*

- A query that refers to two or more attributes with the same name must *qualify* the attribute name with the relation name by *prefixing* the relation name to the attribute name

Example:

EMPLOYEE.LNAME, **DEPARTMENT.**DNAME

ALIASES



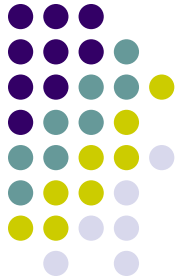
- Some queries need to refer to the same relation twice
  In this case, *aliases* are given to the relation name

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

```
Q8:  SELECT    E.FNAME, E.LNAME, S.FNAME, S.LNAME
     FROM      EMPLOYEE E S
     WHERE     E.SUPERSSN=S.SSN
```

- In Q8, the alternate relation names E and S are called *aliases* or *tuple variables* for the EMPLOYEE relation
- We can think of E and S as two different *copies* of EMPLOYEE; E represents employees in role of *supervisees* and S represents employees in role of *supervisors*

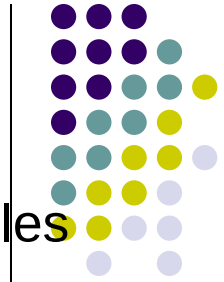
ALIASES (contd.)



- Aliasing can also be used in any SQL query for convenience
- Can also use the AS keyword to specify aliases

```
Q8:SELECT   E.FNAME, E.LNAME, S.FNAME, S.LNAME
   FROM     EMPLOYEE AS E, EMPLOYEE AS S
   WHERE    E.SUPERSSN=S.SSN
```

UNSPECIFIED WHERE-clause



- A *missing WHERE-clause* indicates no condition; hence, all tuples of the relations in the FROM-clause are selected

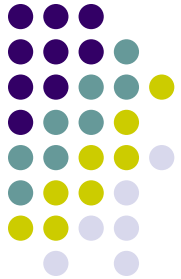
  This is equivalent to the condition WHERE TRUE

- Query 9: Retrieve the SSN values for all employees.

```
Q9: SELECT      SSN
    FROM        EMPLOYEE
```

- If more than one relation is specified in the FROM-clause *and* there is no join condition, then the *CARTESIAN PRODUCT* of tuples is selected

UNSPECIFIED WHERE-clause (contd.)

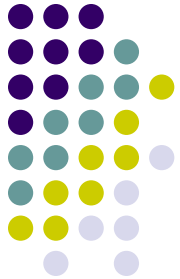


- Example:

```
Q10:SELECT      SSN, DNAME
    FROM        EMPLOYEE, DEPARTMENT
```

  - It is extremely important not to overlook specifying any selection and join conditions in the WHERE-clause; otherwise, incorrect and very large relations may result

USE OF *



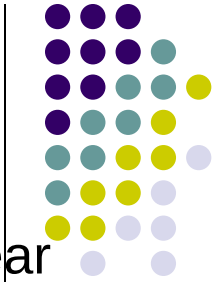
- To retrieve all the attribute values of the selected tuples, a * is used, which stands for *all the attributes*

Examples:

```
Q1C:    SELECT    *
        FROM      EMPLOYEE
        WHERE     DNO=5

Q1D:    SELECT    *
        FROM      EMPLOYEE, DEPARTMENT
        WHERE     DNAME='Research' AND
                  DNO=DNUMBER
```

USE OF DISTINCT



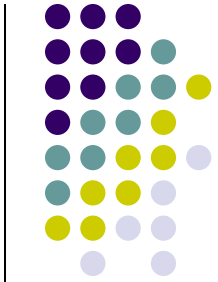
- SQL does not treat a relation as a set; duplicate tuples can appear
- To eliminate duplicate tuples in a query result, the keyword **DISTINCT** is used
- For example, the result of Q11 may have duplicate SALARY values whereas Q11A does not have any duplicate values

```
Q11:    SELECT    SALARY
        FROM      EMPLOYEE
```

```
Q11A:   SELECT    DISTINCT SALARY
        FROM      EMPLOYEE
```

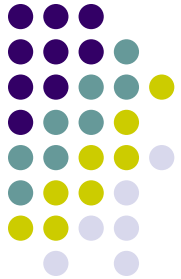
Contents

Retrieval Queries in SQL



1. Simple SQL Queries
2. Aliases
3. *
4. DISTINCT
5. Empty WHERE-clause
6. NULLS IN SQL
7. AGGREGATE FUNCTIONS
8. SUBSTRING COMPARISON
9. ARITHMETIC OPERATIONS
10. GROUPING
11. THE HAVING-CLAUSE
12. ORDER BY
13. EXPLICIT SETS

NULLS IN SQL QUERIES



- Query 14: Retrieve the names of all employees who do not have supervisors.

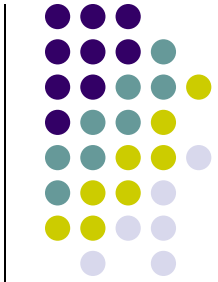
- **Q14:**

```
SELECT    FNAME, LNAME
FROM      EMPLOYEE
WHERE     SUPERSSN  IS  NULL
```

- Note: If a join condition is specified, tuples with NULL values for the join attributes are not included in the result

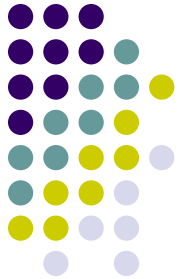
Contents

Retrieval Queries in SQL



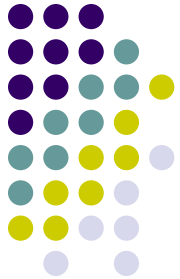
1. Simple SQL Queries
2. Aliases
3. *
4. DISTINCT
5. Empty WHERE-clause
6. NULLS IN SQL
7. AGGREGATE FUNCTIONS
8. SUBSTRING COMPARISON
9. ARITHMETIC OPERATIONS
10. GROUPING
11. THE HAVING-CLAUSE
12. ORDER BY
13. EXPLICIT SETS

AGGREGATE FUNCTIONS



- Include **COUNT, SUM, MAX, MIN, and AVG**
- Query 15: Find the maximum salary, the minimum salary, and the average salary among all employees.

AGGREGATE FUNCTIONS

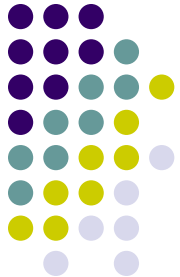


- Include **COUNT, SUM, MAX, MIN, and AVG**
- Query 15: Find the maximum salary, the minimum salary, and the average salary among all employees.

```
Q15:    SELECT    MAX(SALARY),
                  MIN(SALARY), AVG(SALARY)
        FROM   EMPLOYEE
```

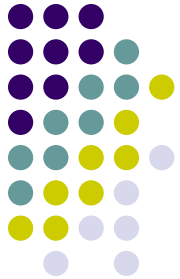
- Some SQL implementations *may not allow more than one function* in the SELECT-clause

AGGREGATE FUNCTIONS



- Query 16: Find the maximum salary, the minimum salary, and the average salary among employees who work for the 'Research' department.

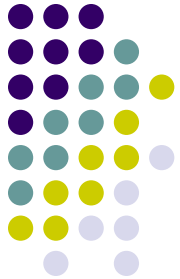
AGGREGATE FUNCTIONS



- Query 16: Find the maximum salary, the minimum salary, and the average salary among employees who work for the 'Research' department.

```
Q16:    SELECT    MAX(SALARY),
                  MIN(SALARY), AVG(SALARY)
        FROM      EMPLOYEE, DEPARTMENT
        WHERE     DNO=DNUMBER AND
                  DNAME='Research'
```

AGGREGATE FUNCTIONS

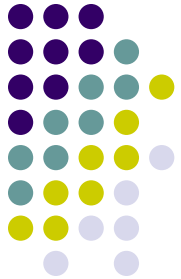


- Queries 17 and 18: Retrieve the total number of employees in the company (Q17), and the number of employees in the 'Research' department (Q18).

```
Q17:    SELECT    COUNT (*)
        FROM      EMPLOYEE

Q18:    SELECT    COUNT (*)
        FROM      EMPLOYEE, DEPARTMENT
        WHERE     DNO=DNUMBER AND
                  DNAME='Research'
```

SUBSTRING COMPARISON



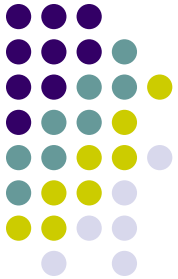
- The **LIKE** comparison operator is used to compare partial strings

- Two reserved characters are used:

'**%**' (or '*****' in some implementations) replaces an arbitrary number of characters

'_' replaces a single arbitrary character

SUBSTRING COMPARISON

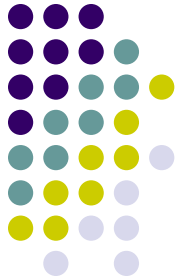


- Query 25: Retrieve all employees whose address is in Houston, Texas. Here, the value of the ADDRESS attribute must contain the substring 'Houston,TX‘ in it.

- Q25:

```
SELECT    FNAME, LNAME
FROM      EMPLOYEE
WHERE     ADDRESS LIKE '%Houston,TX%'
```

SUBSTRING COMPARISON



- Query 26: Retrieve all employees who were born during the 1950s.

Here, '5' must be the 8th character of the string (according to our format for date), so the BDATE value is '_____5_', with each underscore as a place holder for a single arbitrary character.

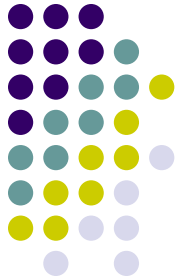
- Q26:

```
SELECT    FNAME, LNAME
FROM      EMPLOYEE
WHERE     BDATE LIKE    '_____5_'
```

- The LIKE operator allows us to get around the fact that each value is considered atomic and indivisible.

  Hence, in SQL, character string attribute values are not atomic

ARITHMETIC OPERATIONS



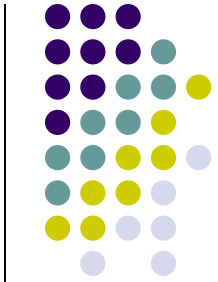
- The standard arithmetic operators **'+', '-'. '*', and '/'** (for addition, subtraction, multiplication, and division, respectively) can be applied to numeric values in an SQL query result

- Query 27: Show the effect of giving all employees who work on the 'ProductX' project a 10% raise.

- Q27:SELECT FNAME, LNAME, 1.1*SALARY
  FROM EMPLOYEE, WORKS_ON, PROJECT
  WHERE SSN=ESSN AND PNO=PNUMBER
  AND PNAME='ProductX'

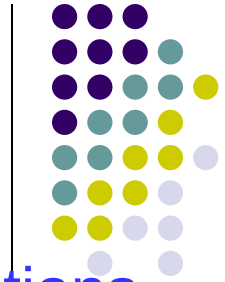
Contents

Retrieval Queries in SQL



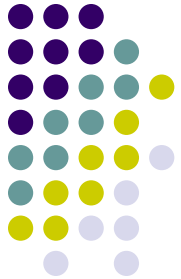
1. Simple SQL Queries
2. Aliases
3. *
4. DISTINCT
5. Empty WHERE-clause
6. NULLS IN SQL
7. AGGREGATE FUNCTIONS
8. SUBSTRING COMPARISON
9. ARITHMETIC OPERATIONS
10. **GROUPING**
11. THE HAVING-CLAUSE
12. ORDER BY
13. EXPLICIT SETS

GROUPING



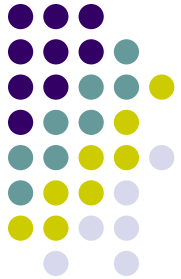
- In many cases, we want to apply the aggregate functions to *subgroups of tuples* in a relation
- Each subgroup of tuples consists of the set of tuples that have the *same value* for the *grouping attribute(s)*
- The function is applied to each subgroup independently
- SQL has a **GROUP BY**-clause for specifying the grouping attributes, which *must also appear in the SELECT-clause*

GROUPING



- Query 20: For each department, retrieve the department number, the number of employees in the department, and their average salary.

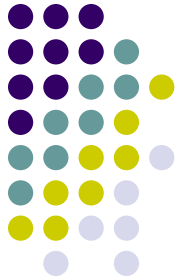
GROUPING



- Query 20: For each department, retrieve the department number, the number of employees in the department, and their average salary.

```
Q20: SELECT      DNO, COUNT (*), AVG (SALARY)
     FROM        EMPLOYEE
     GROUP BY    DNO
```

GROUPING



- Query 20: For each department, retrieve the department number, the number of employees in the department, and their average salary.

```
Q20:  SELECT      DNO, COUNT (*), AVG (SALARY)
      FROM        EMPLOYEE
      GROUP BY    DNO
```

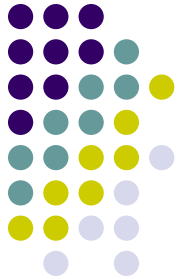
The EMPLOYEE tuples are divided into groups-

Each group having the same value for the grouping attribute DNO

The COUNT and AVG functions are applied to each such group of tuples separately

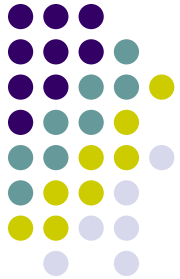
The SELECT-clause includes only the grouping attribute and the functions to be applied on each group of tuples

GROUPING



- Query 21: For each project, retrieve the project number, project name, and the number of employees who work on that project.

GROUPING



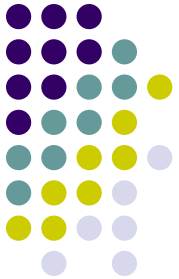
- Query 21: For each project, retrieve the project number, project name, and the number of employees who work on that project.

```
Q21: SELECT     PNUMBER, PNAME, COUNT (*)
     FROM       PROJECT, WORKS_ON
     WHERE      PNUMBER=PNO
     GROUP BY   PNUMBER
```

A join condition can be used in conjunction with grouping

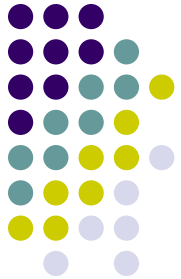
Contents

Retrieval Queries in SQL



1. Simple SQL Queries
2. Aliases
3. *
4. DISTINCT
5. Empty WHERE-clause
6. NULLS IN SQL
7. AGGREGATE FUNCTIONS
8. SUBSTRING COMPARISON
9. ARITHMETIC OPERATIONS
10. GROUPING
11. **THE HAVING-CLAUSE**
12. ORDER BY
13. EXPLICIT SETS

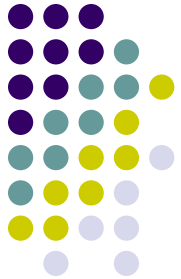
THE HAVING-CLAUSE



- Sometimes we want to retrieve the values of these functions for only those *groups that satisfy certain conditions*

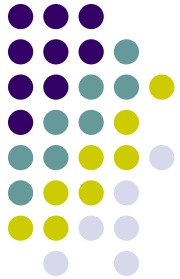
- The **HAVING**-clause is used for specifying a selection condition on groups (rather than on individual tuples)

THE HAVING-CLAUSE



- Query 22: For each project *on which more than two employees work* , retrieve the project number, project name, and the number of employees who work on that project.

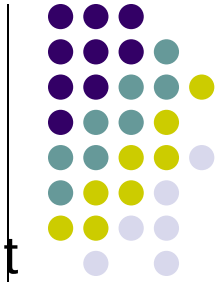
THE HAVING-CLAUSE



- Query 22: For each project *on which more than two employees work* , retrieve the project number, project name, and the number of employees who work on that project.

```
Q22: SELECT     PNUMBER, PNAME, COUNT (*)
     FROM       PROJECT, WORKS_ON
     WHERE      PNUMBER=PNO
     GROUP BY   PNUMBER, PNAME
     HAVING     COUNT (*) > 2
```

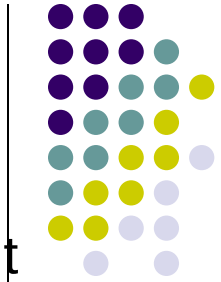
ORDER BY



- The **ORDER BY** clause is used to sort the tuples in a query result based on the values of some attribute(s)

- Query 28: Retrieve a list of employees and the projects each works in, ordered by the employee's department, and within each department ordered alphabetically by employee last name.

ORDER BY

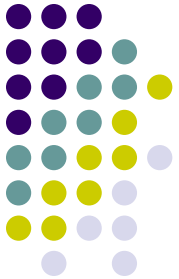


- The **ORDER BY** clause is used to sort the tuples in a query result based on the values of some attribute(s)

- Query 28: Retrieve a list of employees and the projects each works in, ordered by the employee's department, and within each department ordered alphabetically by employee last name.

SELECT DNAME, LNAME, FNAME, PNAME
FROM DEPARTMENT, EMPLOYEE, WORKS_ON, PROJECT
WHERE DNUMBER=DNO AND SSN=ESSN AND PNO=PNUMBER
ORDER BY DNAME, LNAME

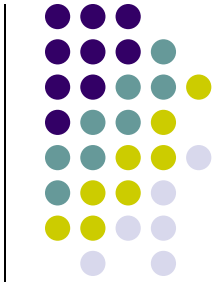
ORDER BY



- The default order is in ascending order of values
- We can specify the keyword **DESC** if we want a descending order; the keyword **ASC** can be used to explicitly specify ascending order, even though it is the default

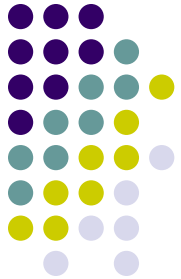
Contents

Retrieval Queries in SQL



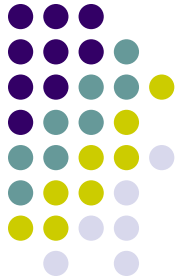
1. Simple SQL Queries
2. Aliases
3. *
4. DISTINCT
5. Empty WHERE-clause
6. NULLS IN SQL
7. AGGREGATE FUNCTIONS
8. SUBSTRING COMPARISON
9. ARITHMETIC OPERATIONS
10. GROUPING
11. THE HAVING-CLAUSE
12. ORDER BY
13. EXPLICIT SETS

EXPLICIT SETS



- It is also possible to use an **explicit (enumerated) set of values** in the WHERE-clause rather than a nested query
- Query 13: Retrieve the social security numbers of all employees who work on project number 1, 2, or 3.

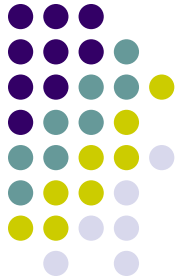
EXPLICIT SETS



- It is also possible to use an **explicit (enumerated) set of values** in the WHERE-clause rather than a nested query
- Query 13: Retrieve the social security numbers of all employees who work on project number 1, 2, or 3.

```
Q13:    SELECT    DISTINCT ESSN
        FROM      WORKS_ON
        WHERE     PNO IN  (1, 2, 3)
```

Summary of SQL Queries



- A query in SQL can consist of up to six clauses, but only the first two, SELECT and FROM, are mandatory. The clauses are specified in the following order:

SELECT <attribute list>
FROM <table list>
[WHERE <condition>]
[GROUP BY <grouping attribute(s)>]
[HAVING <group condition>]
[ORDER BY <attribute list>]