Database Management Systems

BITS Pilani
Hyderabad Campus

Dr.R.Gururaj
CS&IS Dept.

Lecture Session-8

Structured Query Language (SQL)-2

innovate achieve lead

Content

- *Nested queries and correlated nested queries*
- *Use of EXISTS and NOT EXISTS*
- *Explicit join operations*
- *Aggregate functions*
- *Group by and Having classes*
- *Insert/ Update / Delete operations*
- *Views*

Nested Queries

Ex.1 Retrieve the name of each employee who has a dependent with the same name as the employee.

```
SELECT  E.Fname
FROM EMPLOYEE AS E
WHERE E.ssn IN(SELECT ESSN FROM DEPENDENT
        WHERE E.FNAME = DEPENDENT_NAME);
```

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, then the two queries are said to be correlated.

Use of NOT EXISTS clause

Ex. 2

Retrieve the names, salary of employees who have no dependents

```
SELECT Fname, Salary
FROM EMPLOYEE
WHERE NOT EXISTS (SELECT * FROM DEPENDENT WHERE SSN
= ESSN);
```

We can also use ‘EXISTS’ to check the existence of at least one tuple in the result.

It is also possible to use an explicit set of values in the WHERE – clause.

We can also check whether a value is NULL

Renaming Attributes in the Result

Ex. 3

```
SELECT name AS Emp_name
FROM EMPLOYEE
WHERE Dno = 5;
```

Join Operation

We can also perform

- Join – using key word 'JOIN'
- Natural join – using key word 'NATURAL JOIN'
- Left outer join – using key word 'LEFT OUTER JOIN'
- Right outer join – using key word 'RIGHT OUTER JOIN'

Aggregate Functions and Grouping

- COUNT
- SUM
- MAX
- MIN
- AVG

Ex. 4 *SELECT SUM (Salary), AVG (Salary) from EMPLOYEE;*

Ex. 5 To retrieve number of rows in Employee table

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

Ex. 6 Retrieve the name of employees who have two or more dependents

```
SELECT Fname
FROM EMPLOYEE
WHERE (SELECT COUNT (*) FROM DEPENDENT WHERE SSN
=       ESSN) > = 2;
```

Group by

Ex. 7 For each department retrieve the department number and no of employees.

```
SELECT dno, count (*)
FROM EMPLOYEE
GROUP BY Dno;
```

Group by and Having clause

Ex. 8 Retrieve the department number and no of employees for the departments which have more than 5 employees working for it.

```
SELECT dno, count (*)
FROM EMPLOYEE
GROUP BY Dno
HAVING count(*)>5;
```

INSERT operation

For Inserting a new tuple into the relation

General Form

INSERT INTO <table name>
VALUES(v_1, v_2, v_3,……..v_n);

Ex. 9 *INSERT INTO DEPARTMENT*
VALUES('MARKETING',10, 103, '2000-06-25');

Deleting a tuple

Ex. 10 *DELETE FROM <table name>*
WHERE <condition>;

Ex. 11 *DELETE FROM DEPARTMENT*
WHERE dnumber=10;

If we don't specify the condition all tuples are deleted.

Update command

Ex. 12 *UPDATE EMPLOYEE*
SET salary = 60000
WHERE ssn = 141;

Updates tuples in Employee table for the tuples with ssn = 141, sets the value of the attribute salary to 60,000

Views in SQL

A view in SQL is a single table that is derived from other tables.

These other tables are known as *base tables*.

A view does not necessarily exist in physical form, it can be considered as a *virtual table*.

The tuples of base tables are actually stored in database.

This limits the updates on views.

In fact when a view is updated, the corresponding base tables are the structures which are to be updated.

This makes update operations on views complex.

Creating View

```
CREATE VIEW EMP_DETAILS
AS SELECT name, salary, dname, age, dloc
    FROM EMPLOYEE, DEPARTMENT
    WHERE dno = dnumber;
```

Whenever the view definition is executed, the new temporary table is generated with specified attributes from specified base tables.

View definitions are stored in database, not the result of the view. From then onwards view can be seen as a table and queries can be posed on it.

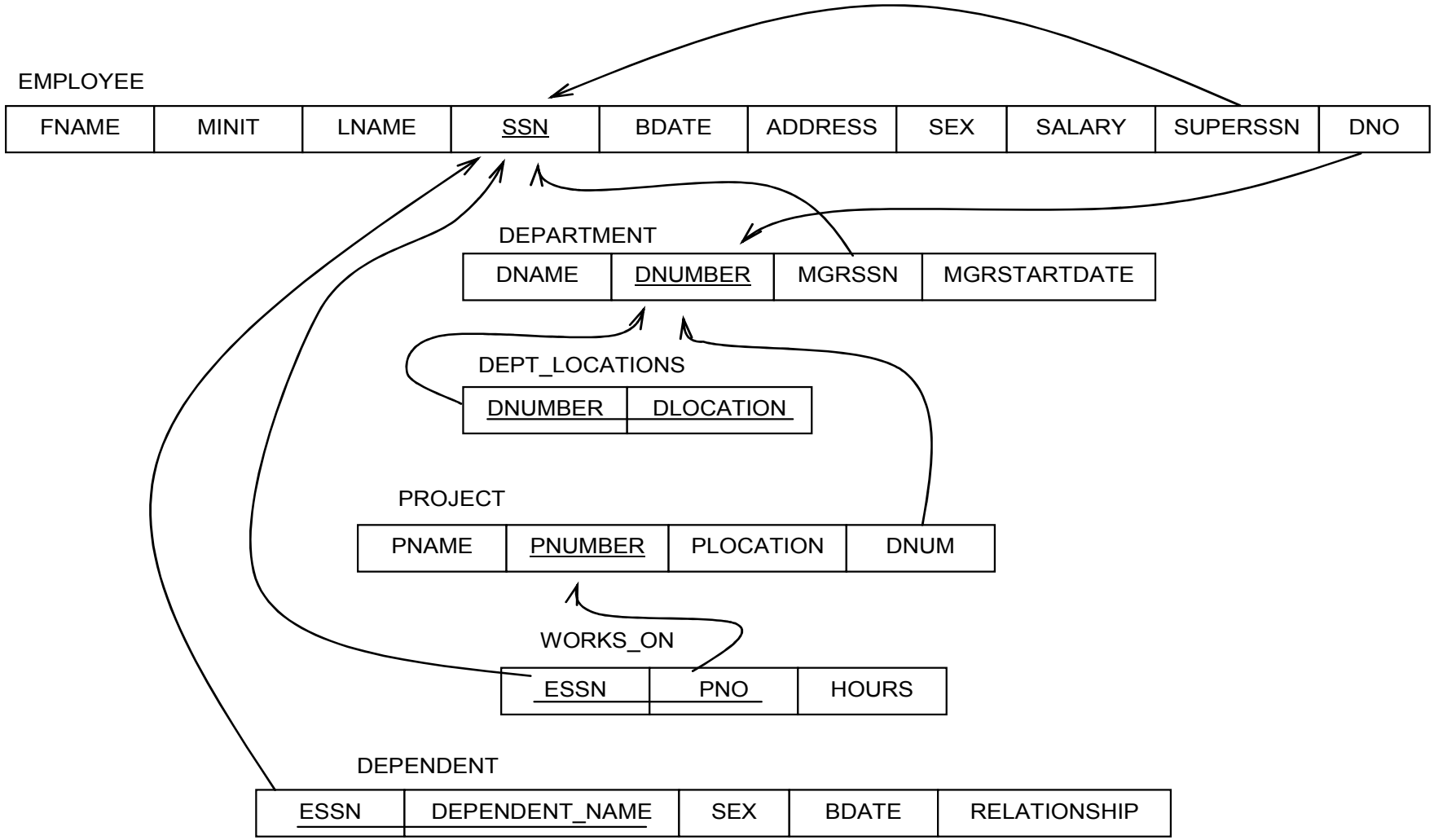
Ex. *SELECT name, dname FROM EMP_DETAILS WHERE dno = 5;*

Here EMP_DETAILS is a view. Where this query is executed, first the view definition for EMP_DETAILS is executed and the select and where operation are performed on the temporary table.

Note:

- A view is always up to date.
- Updates are generally not possible on views.
- Meant for querying only.
- Some times it is possible to store views for some duration.
- Those views are known as *materialized views*.

Example SQL statements



1. Get the list of employee IDs who have no dependents.

```
select ssn
from Employee
where ssn NOT IN  ( select essn
                    from Dependent
                   );
```

```
(select ssn from Employee)
except
(select essn from Dependent);
```

2. Get the list of employee IDs who have more than two dependents.

```
select essn
from Dependent
group by essn
having count(*) > 2;
```

3. Get the list of projects controlled by department with name "ACCOUNTS".

```
select pnumber, pname
from Projects
where Dnum IN    ( select dnumber
                   from  Department
                   where Dname='ACCOUNTS');

select pnumber, pmname
from Project, Department
where Dnum=Dnumber AND Dname= 'ACCOUNTS';
```

4. Get the list of employee IDs working on all projects

```
Select essn
From Works_on
Group By essn
Having COUNT(*) = (select COUNT(*)  from project);

select E.essn
from Works_on as E
where ((select pno from Works_on where essn=E.essn)
            contsins
           (select pnumber from Project));
```

5. Find the projects controlled by departments located in Mumbai.

```
select pnumber, pname
from project
where dnum = (select dnumber
                    from Dept_locations
                    where Dlocation='Mumbai');
```

6. Update the salary of those employees working with department- HR , to Rs. 20000

```
update Employee
set salary=20000
where dnum = (select dnumber
                    from Department
                    where Dname='HR');
```

7. Delete the records of employees who get salary less than 5000.

```
delete
from Employee
where salary < 5000;

delete
from Employee;
```

Summary

- ✓ *How to write nested queries in SQL*
- ✓ *Writing queries using the clauses EXISTS, NOT XISTS, BETWEEN AND, IN, NOT IN*
- ✓ *How to perform explicit JOIN operations*
- ✓ *How to use GROUP BY and HVING*
- ✓ *The concept of views in SQL*
- ✓ *Some examples on SQL*