IE301

Analysis and Design of Data Systems

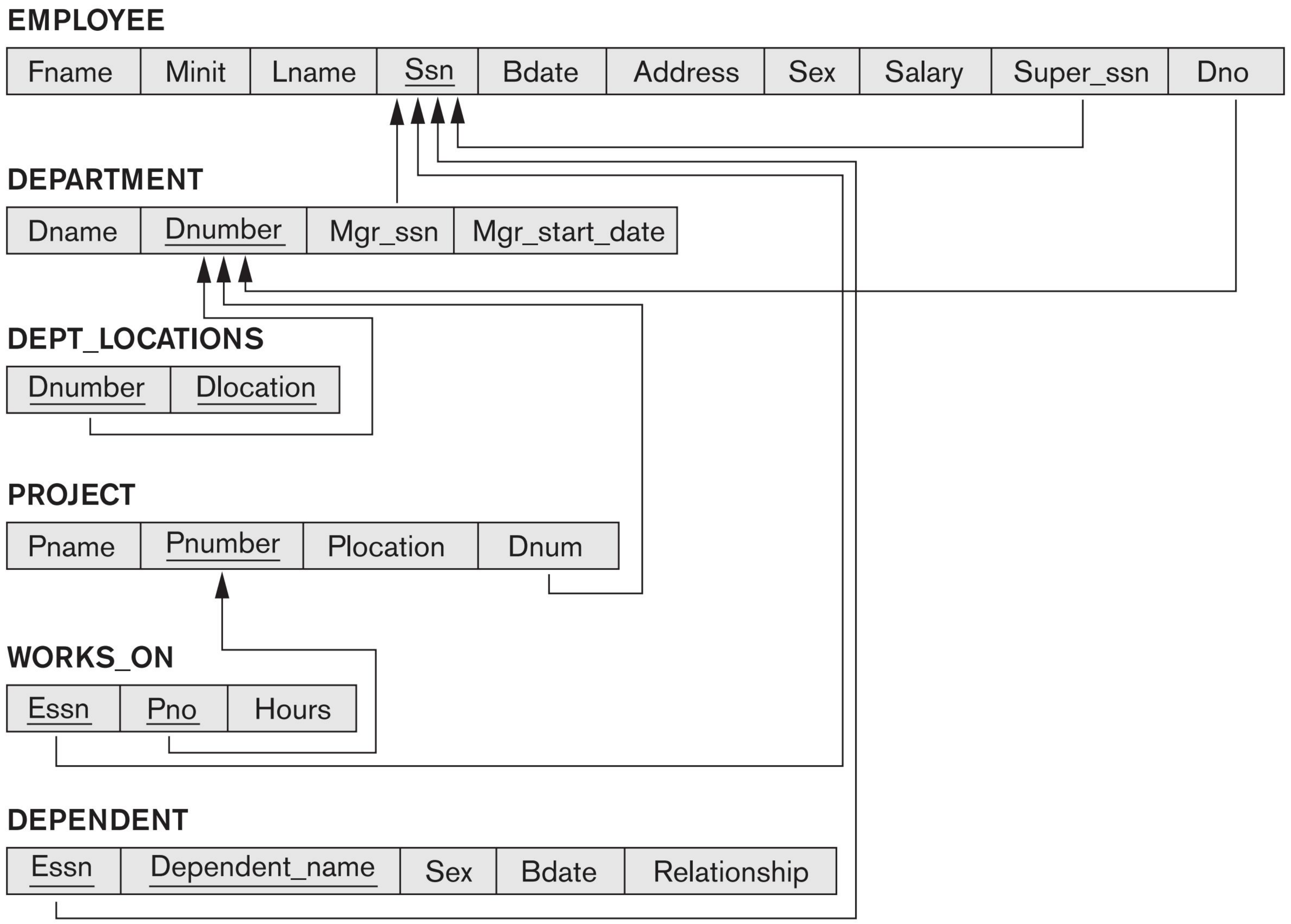
Lecture 13

Complex SQL Queries

Aram Keryan

October 26, 2015

Employee database



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

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

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

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

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

Unspecified WHERE Clause

A *missing* WHERE clause indicates no condition on tuple selection

```
SELECT Fname FROM EMPLOYEE;
```

Jared	Josh	Jeff	Joyce	Lyle	Helga	James
Jon	Andy	Franklin	John	Billie	Naveen	Jennifer
Justin	Tom	Alex	Nandita	Jon	Carl	Ahmad
Brad	Jenny	Bonnie	Bob	Ray	Sammy	Alicia
John	Chris	Alec	Jill	Gerald	Red	
Evan	Kim	Sam	Kate	Arnold	Ramesh	

✔ First names of all employees are retrieved

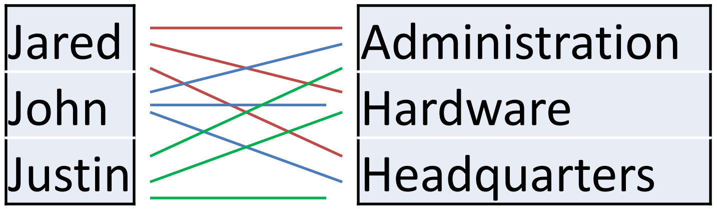
Unspecified WHERE Clause

SELECT Fname, Dname **FROM** EMPLOYEE, DEPARTMENT;

What is the outcome?

One might think that the result is "first name of employee" plus "name of corresponding department he works at" **BUT**

If more than one relation is specified in the FROM clause and there is no WHERE clause, then the **CROSS PRODUCT**—*all possible tuple combinations*—of these relations is selected



For expected result we have to add WHERE clause:

SELECT Fname, Dname **FROM** EMPLOYEE e, DEPARTMENT d
WHERE e.Dno = d.Dnumber;

Asterisk (*)

To retrieve all the attribute values of the selected tuples, we specify an *asterisk* (*), which stands for *all the attributes*

1)
```
SELECT      *
FROM        EMPLOYEE
WHERE       Dno=5;
```

2)
```
SELECT      *
FROM        EMPLOYEE, DEPARTMENT
WHERE       Dname='Research' AND Dno=Dnumber;
```

3)
```
SELECT      *
FROM        EMPLOYEE, DEPARTMENT;
```

✔ Try these examples at home on MySQL

Tables as Sets in SQL

Generally saying, tables in SQL, unlike relations, allow duplicates

 SQL does not automatically eliminate duplicate tuples in the results of queries, for the following reasons:

- Duplicate elimination is an expensive operation.
- The user may want to see duplicate tuples in the result of a query.
- When an aggregate function (will learn later) is applied to tuples, in most
  cases we do not want to eliminate duplicates.

✔ In that context table is a *multiset* rather than a set

Tables as Sets in SQL (DISTINCT)

```
SELECT Fname FROM EMPLOYEE;
```

Jared	Josh	Jeff	Joyce	Lyle	Helga	James
Jon	Andy	Franklin	**John**	Billie	Naveen	Jennifer
Justin	Tom	Alex	Nandita	**Jon**	Carl	Ahmad
Brad	Jenny	Bonnie	Bob	Ray	Sammy	Alicia
John	Chris	Alec	Jill	Gerald	Red	
Evan	Kim	Sam	Kate	Arnold	Ramesh	

```
SELECT DISTINCT Fname FROM EMPLOYEE;
```

Jared	Evan	Chris	Bonnie	Bob	Ray	Carl	Jennifer
Jon	Josh	Kim	Alec	Jill	Gerald	Sammy	Ahmad
Justin	Andy	Jeff	Sam	Kate	Arnold	Red	Alicia
Brad	Tom	Franklin	Joyce	Lyle	Helga	Ramesh	
John	Jenny	Alex	Nandita	Billie	Naveen	James	

Tables as Sets in SQL (UNION, EXCEPT, INTERSECT)

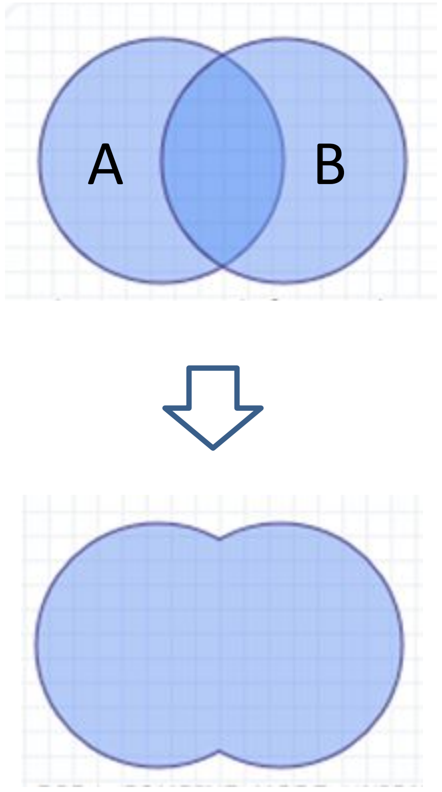
SQL has directly incorporated some of the set operations from mathematical *set theory*

- ✔ The relations resulting from these set operations are sets of tuples; that is, *duplicate tuples are eliminated from the result.*
- ✔ These set operations apply only to *union-compatible relations*, so we must make sure that the two relations on which we apply the operation have the same attributes and that the attributes appear in the same order in both relations.

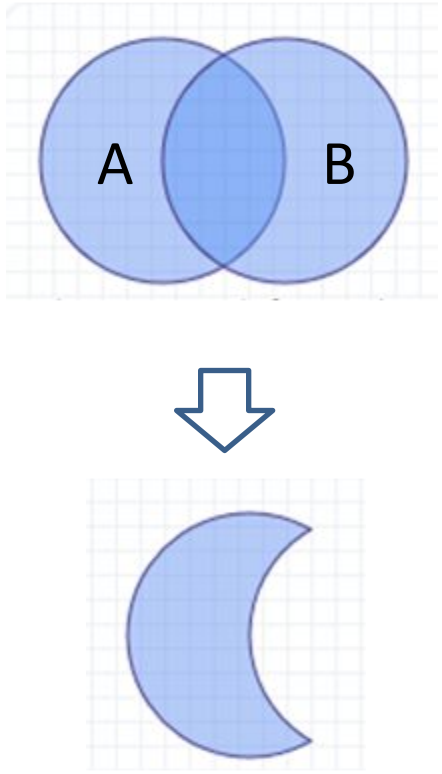
- UNION ALL, EXCEPT ALL, INTERSECT ALL: read in section 4.3.4

Tables as Sets in SQL (UNION, EXCEPT, INTERSECT)

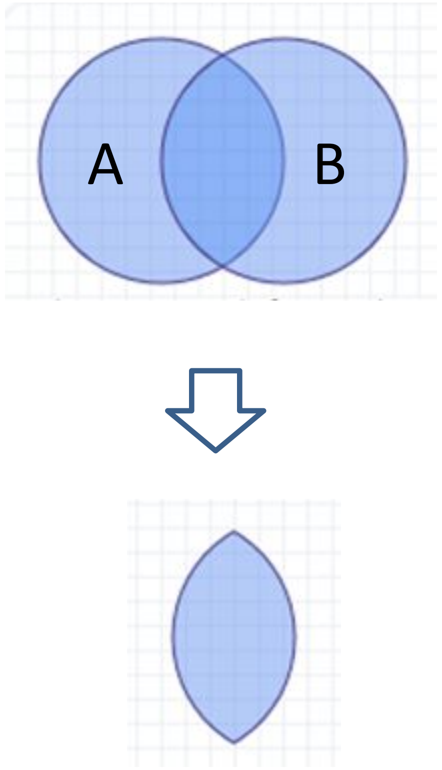
UNION



EXCEPT



INTERSECTION



Tables as Sets in SQL (UNION)

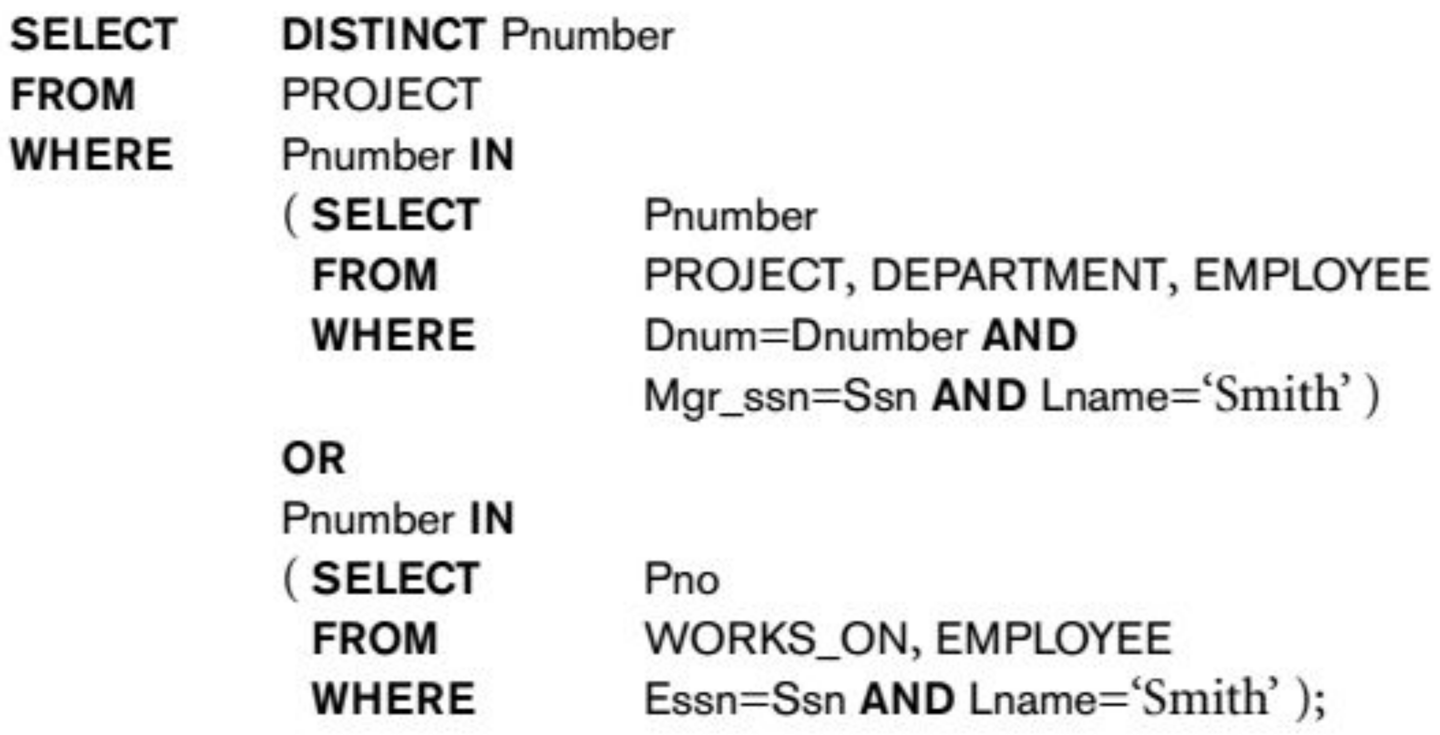
Query: Make a list of all project numbers for projects that involve an employee whose last name is ‘Smith’, either as a worker or as a manager of the department that controls the project.

```
(SELECT  DISTINCT Pnumber
FROM          PROJECT p, DEPARTMENT d, EMPLOYEE e
WHERE         p.Dnum = d.Dnumber AND d.Mgr_ssn = e.Ssn
         AND e.Lname = ‘Wong’)
UNION
(SELECT  DISTINCT Pnumber
FROM          WORKS_ON w, PROJECT p, EMPLOYEE e
WHERE         w.Essn = e.Ssn AND w.Pno = p.Pnumber
         AND e.Lname = 'Wong');
```

- LIKE, AS, BETWEEN, ORDER BY: read in sections 4.3.5 – 4.3.6

Nested Queries

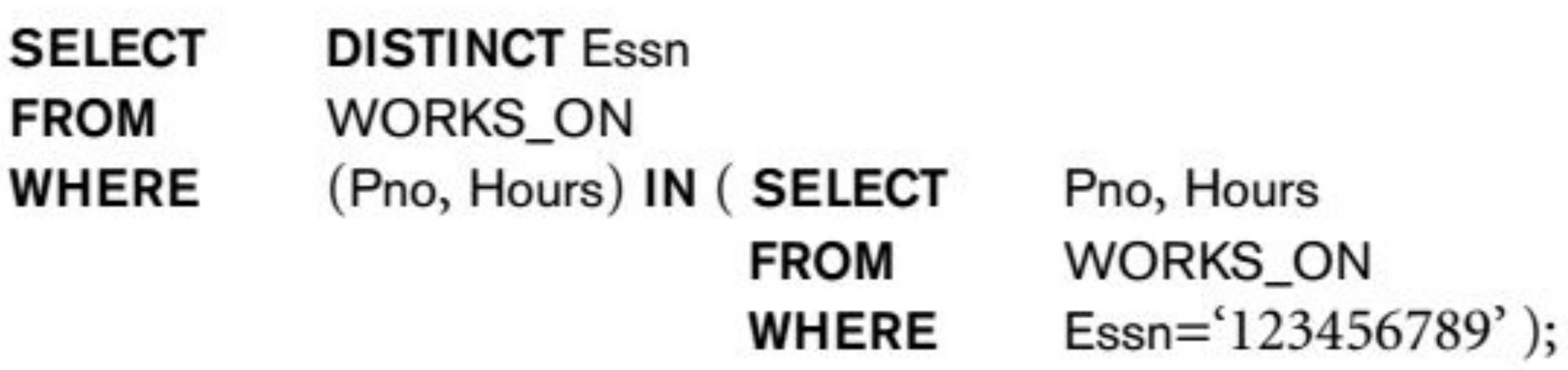
Some queries require that existing values in the database be fetched and then used in a comparison condition



```
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' );
```

More examples

Formulate the query for the next SQL sintaxis:



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

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**.

✔ We can understand a correlated query better by considering that the *nested query is evaluated once for each tuple (or combination of tuples) in the outer query*

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

DEPENDENT (Essn, Dependent_name, Sex, Bdate, Realtionship);

```
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 );
```

Correlated Nested Queries

In general, a query written with nested select-from-where blocks and using the = or IN comparison operators can *always* be expressed as a single block query. For example, here is the same example as on the previous slide:

DEPENDENT (Essn, Dependent_name, Sex, Bdate, Realtionship);

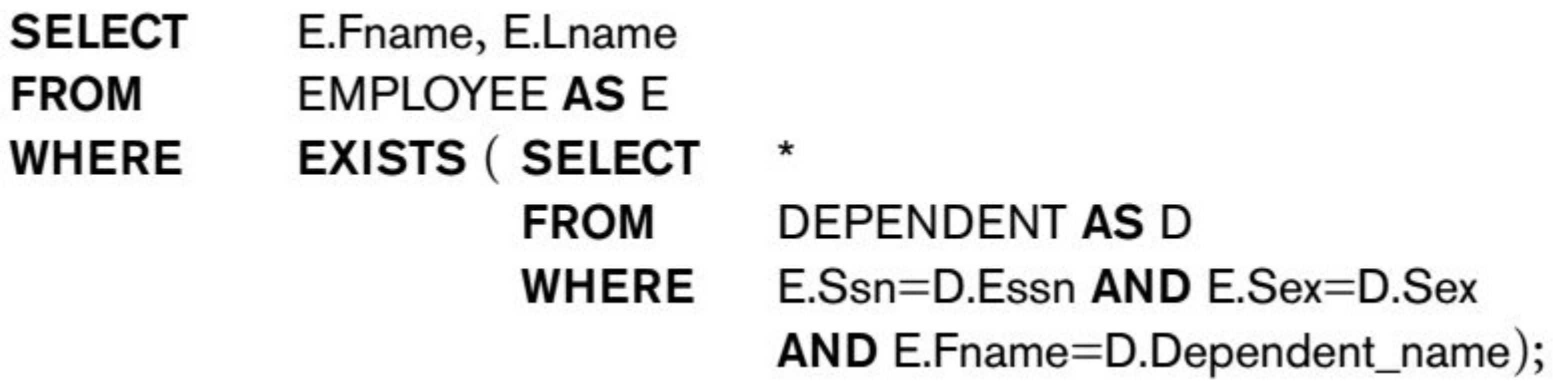
```
SELECT    E.Fname, E.Lname
FROM      EMPLOYEE AS E, DEPENDENT AS D
WHERE     E.Ssn=D.Essn AND E.Sex=D.Sex
          AND E.Fname=D.Dependent_name;
```

Correlated Nested Queries (EXISTS)

The EXISTS (NOT EXISTS) function in SQL is used to check whether the result of a correlated 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 .

Example from previous slide:



```
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);
```

More examples

Retrieve the names of employees who have no dependents.

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

More examples

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

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

More examples

Retrieve the name of each employee who works on all *the projects controlled by department number 5*

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

More examples

Retrieve the name of each employee who works on all *the projects controlled by department number 5*

```
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 )));
```

More examples (cont.)

Let's rephrase the query:

Before:

Retrieve the name of each employee who works on all the projects controlled by department number 5

After:

Select each employee such that there does not exist a project controlled by department 5 that the employee does not work on.

More examples (cont.)

Select each employee such that there does not exist a project controlled by department 5 that the employee does not work on.

```
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 )));
```