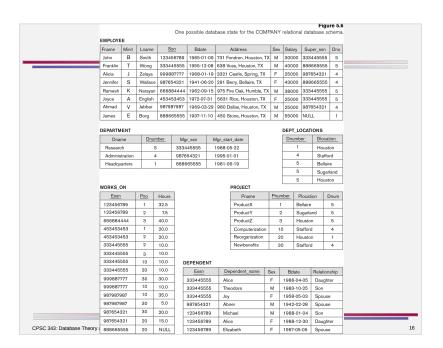## Set Operations and Subqueries – Key Points

- JOIN creates a new table with more columns
  - new rows are pairwise combinations of rows from the original relations
- UNION creates a new table with more rows
  - new rows are the rows from either of the original relations
  - removes duplicate rows; use UNION ALL to keep duplicates
- JOIN, NATURAL JOIN lose rows with no match in the other table
- OUTER JOIN keeps those rows, filling in with NULL values
- subqueries in FROM allow building up a more complex table from which to draw the result
- subqueries in WHERE allow more complex conditions for choosing rows
  - including conditions where one row is compared against another

---

OUTER JOIN

Join combines matching rows in two relations. But what if a row in one relation has no match in the other?

An OUTER JOIN includes rows with no match in the other relation, filling in the missing columns with NULL.
- R LEFT OUTER JOIN S includes rows in R with no match in S
- R RIGHT OUTER JOIN S includes rows in S with no match in R
- R FULL OUTER JOIN S includes both rows in R with no match in S and rows in S with no match in R

OUTER JOINs can be NATURAL or include an ON condition.

Note: MySQL does not support full outer joins.

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

UNION

```
( SELECT … )
UNION
( SELECT … )
```

- result consists of the rows from both result sets

Notes.

- columns need to match in number, order
- columns should match in data type
- does *not* include duplicates
  - use UNION ALL instead to keep

```
( SELECT Lname,Sex
  FROM EMPLOYEE )
UNION
( SELECT E.Lname,D.Sex
  FROM EMPLOYEE E JOIN DEPENDENT D ON E.Ssn=D.Essn )
```

Subqueries in WHERE

If the subquery returns only a single value (one column, one row), it can be used like a constant.

```
SELECT *
FROM EMPLOYEE
WHERE Ssn = ( SELECT Mgr_ssn
              FROM DEPARTMENT
              WHERE Dnumber=5 )
```

SQL Operations on Relations

operation	meaning	notes
EXISTS R	true if and only if R is not empty	
s IN R	true if and only if s is a row of R	
s > ALL R	true if and only if s is greater than every value in R	also >=, <. <=, =, <>
s > ANY R	true if and only if s is greater than at least one value in R	also >=, <. <=, =, <>

  – can also prefix with NOT
    • NOT EXISTS R, s NOT IN R, NOT s > ALL R, etc

All of these operations work when R has only one column.

Many of them also make sense when R has multiple columns, if s has the same columns as R.

SQL Operations on Relations

operation	meaning	notes
EXISTS R	true if and only if R is not empty	
s IN R	true if and only if s is a row of R	
s > ALL R	true if and only if s is greater than every value in R	also >=, <. <=, =, <>
s > ANY R	true if and only if s is greater than at least one value in R	also >=, <. <=, =, <>

  – can also prefix with NOT
    • NOT EXISTS R, s NOT IN R, NOT s > ALL R, etc

All of these operations work when R has only one column.

Many of them also make sense when R has multiple columns, if s has the same columns as R.

Correlated Subqueries

```
SELECT E.Fname,E.Lname,W.Pno,W.Hours
FROM EMPLOYEE E JOIN WORKS_ON W ON E.Ssn=W.Essn
WHERE W.Hours >= ALL ( SELECT Hours
                       FROM WORKS_ON W2
                       WHERE W2.Pno=W.Pno )
```

This is a *correlated subquery* – it must be evaluated for every row of the FROM relation because its result depends on the current row.

Compare to

```
SELECT *
FROM EMPLOYEE
WHERE Ssn = ( SELECT Mgr_ssn
              FROM DEPARTMENT
              WHERE Dnumber=5 )
```

The subquery will return the same value every time, so it could be evaluated once and then treated as a constant.

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

```
SELECT Fname,Lname,Ssn,Pno,Hours
FROM EMPLOYEE E JOIN WORKS_ON W ON E.Ssn=W.Essn
WHERE W.Hours >= ALL ( SELECT Hours
                       FROM WORKS_ON W2
                       WHERE W2.Pno=W.Pno )
```

```
SELECT *
FROM EMPLOYEE E
WHERE NOT EXISTS ( SELECT *
                   FROM EMPLOYEE E2
                   WHERE E.Ssn <> E2.Ssn AND E.Dno = E2.Dno )
```

Subqueries in FROM

You can also use a SELECT query to create a relation to be used in the FROM clause.
– the result must be given an alias

```
SELECT E.Fname,E.Lname,W.Pno,W.Hours
FROM EMPLOYEE E JOIN ( SELECT *
                       FROM WORKS_ON W
                       WHERE W.Hours >= ALL
                              ( SELECT Hours
                                FROM WORKS_ON W2
                                WHERE W2.Pno=W.Pno )
                       ORDER BY Hours DESC
                       LIMIT 3 ) W ON E.Ssn=W.Essn
```

Strategies for Writing Queries

Two strategies –

- start with FROM – build a table that contains all of the rows in the desired result and all of the columns you need for the WHERE and SELECT clauses
- add WHERE to pick the desired rows
- add SELECT to pick the desired columns

- start with FROM – build a table that contains all of the rows in the desired result and all of the columns that you need for the SELECT clause
- add WHERE to pick the desired rows, using a subquery if the condition involves information not in the FROM table
- add SELECT to pick the desired columns