# SQL (Continued)
# Querying Multiple Tables



## Joining with SQL

- Attributes in the answer (result) may come from two or more tables.
- Joining is actually to link multiple tables together under certain conditions, which is similar to the join operator in relational algebra.
- Two ways to define a join with SQL92 Standard
  - List all the participating tables in the from clause, and specify the join condition in the where clause.
  - Define a join directly in the from clause



## Join Example

- Query:
  - List the student ID, and name of all students with the name of the courses in which the student enrolled

student(SID, Name, Age) enrollment(*SNo*, *CName*, Grade)

**SELECT** SID, Name, CName
**FROM** student, enrollment
**WHERE** SID = SNo;

**SELECT** SID, Name, CName
**FROM** student **JOIN** enrollment **ON** SID = SNo;



## Disambiguate Attribute Name

- What happens if two table share the same attribute name?
- Use table_name.attribute_name

student(SID, Name, Age) enrollment(*SID*, *CName*, Grade)

**SELECT** SID, Name, CName
**FROM** student, enrollment
**WHERE** student.SID = enrollment.SID;

**SELECT** SID, Name, CName
**FROM** student **JOIN** enrollment
**ON** student.SID = enrollment.SID;



## Outer Join

- Remember that join only keeps matched tuples
- What if I want to keep rows that are not matched?
- Outer join comes to the rescue
- Three type of outer joins
  - Left outer join
  - Right outer join
  - Full outer join



## Left Outer Example

Query: Get the name and address of all students, and, if any, list all the courses that the students have enrolled in.

STUDENT(SSN, Name, Address)
ENROLLMENT(SSN, CName, Grade)

**SELECT** Name, Address
**FROM** STUDENT **LEFT OUTER JOIN** ENROLLMENT
**ON** (STUDENT.SSN = ENROLLMENT.SSN);

**SELECT** Name, Address
**FROM** STUDENT, ENROLLMENT
**WHERE** STUDENT.SSN = ENROLLMENT.SSN (+);
(ORACLE Syntax)

## Right Outer Example

Query: Get the name and address of all students, and, if any, list all the courses that the students have enrolled in.

STUDENT(SSN, Name, Address)

ENROLLMENT(SSN, CName, Grade)

```
SELECT Name, Address
FROM ENROLLMENT RIGHT OUTER JOIN STUDENT
        ON (ENROLLMENT.SSN = STUDENT.SSN);

SELECT Name, Address
FROM STUDENT, ENROLLMENT
WHERE ENROLLMENT.SSN (+) = STUDENT.SSN;
(ORACLE Syntax)
```



## Full Outer Join Example

STUDENT(SSN, Name, Address, *DeptID*)
DEPT(ID, DeptName)

```
SELECT SSN, Name, DeptName
FROM STUDENT FULL OUTER JOIN DEPT
        ON (STUDENT.DeptID = DEPT.ID);

SELECT Name, Address
FROM STUDENT, ENROLLMENT
WHERE STUDENT.DeptID (+) = DEPT.ID (+);
(ORACLE Syntax)
```



## Sub-queries

- Placing a sub-query within a WHERE or HAVING clause of the main query.
- The sub-query provides values for the search condition of the main query.
- Sometimes either the joining or the sub-query technique may be used to accomplish the same result
- Two types of sub-queries
  - Correlated sub-queries
  - Non-correlated sub-queries



## An Example Using Join

DEPT(ID, Name, Address, Phone)
FACULTY(*ID*, Name, Phone, *DeptID*)

Query: Find the name and phone of all CS faculty

```
SELECT    F.Name, Phone
FROM      FACULTY AS F, DEPT AS D
WHERE     D.ID = F.DeptID AND D.Name = 'CS';
```



## Equivalent Query Using Sub-query

```
SELECT    Name, Phone
FROM      FACULTY
WHERE     DeptID = (    SELECT ID
                        FROM DEPT
                        WHERE Name ='CS'
                    );
```

Please note:

You have to ensure that the sub-query only get one answer because = is not a set operator; Or otherwise you will get an error message.



## Sub-query & IN (1)

- What faculty are not in CS?
- You cannot use = at this time. Instead, you can use IN operator.

```
SELECT    Name, Phone
FROM      FACULTY
WHERE     DeptID IN (   SELECT ID
                        FROM DEPT
                        WHERE Name <> 'CS'
                    );
```

## Sub-query & IN (2)

- Which students take database?

```
SELECT   SID
FROM  STUDENT
WHERE    SID IN (SELECT   SNo
                 FROM     enrollment
                 WHERE    CName = 'Database'
                 );
```

Questions:

1. What could be the equivalent query by using Joining?
2. How about "Which students did not take database"?
3. Can you define an equivalent query by using joining for 2?



## EXISTS & NOT EXISTS

- EXISTS will be *true* if the sub-query returns one or more rows, and *false* if no rows are returned
- Conversely, NOT EXISTS will be *true* if no rows are returned, and *false* if one or more rows are returned.



## Sub-queries with EXISTS

- Which students enrolled in more than one class?

```
SELECT    DISTINCT SNo
FROM      ENROLLMENT AS A
WHERE     EXISTS ( SELECT *
                   FROM ENROLLMENT AS B
                   WHERE A.SNo = B.SNo
                   AND A.CName <> B.CName );
```



## Sub-queries with EXISTS

A

| CName | SNo |
|---|---|
| Database | A1 |
| Compiler | B2 |
| Operating System | A1 |
| Data Structure | C3 |
| Software Engineering | B2 |

'A1' ='A1'
'Database' <> 'Operating System'

B

| SNo | CName |
|---|---|
| A1 | Database |
| B2 | Compiler |
| A1 | Operating System |
| C3 | Data Structure |
| B2 | Software Engineering |

Possible evaluation steps:

- Take each record x in A
- Filter each record y in B to see if (x.SNo = y.SNo) AND (x.CName <> y.CName)
- If found at least one record y in B, then keep record x in A
- Remove duplicates at the end
- The query result is {A1, B2}



## An Equivalent Query

- Which students enrolled in more than one class?

```
SELECT    SNo
FROM      ENROLLMENT
GROUP BY  SNo
HAVING    COUNT(*) > 1;
```



## Which classes are not taken?

CLASS(CID, Name)  ENROLLMENT(CID, SID)

```
SELECT Name
FROM CLASS AS C
WHERE NOT EXISTS ( SELECT *
                   FROM ENROLLMENT AS E
                   WHERE C.CID = E.CID);
```

## Correlated vs. Non-correlated

- Correlated:
  - A sub-query in which processing the inner query depends on data from the outer query.
  - The inner query is somewhat different for each row referenced in the outer query.
  - The inner query must be computed for each outer row.
- Non-correlated:
  - The inner query was computed only once for all rows processed in the outer query.