COSC 3380
Design of Database Systems

Complex Queries, Triggers, Views, and Schema Modification

March 20, 2024

Nested Queries

- Use tuples of values in comparisons
  - Place them within parentheses

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

Select Essns of all employees who work the same (project, hours) combination on some project that employee 'John Smith' works on.

Nested Queries

- Use other comparison operators to compare a single value *v*
  - = ANY (or = SOME) operator
    - Returns TRUE if the value *v* is equal to some value in the set *V* and is hence equivalent to IN
  - Other operators that can be combined with ANY (or SOME): >, >=, <, <=, and <>
  - ALL: value must exceed all values from nested query

```
SELECT   Lname, Fname
FROM     EMPLOYEE
WHERE    Salary > ALL  ( SELECT   Salary
                         FROM    EMPLOYEE
                         WHERE   Dno=5 );
```

Nested Queries

- Avoid potential errors and ambiguities
  - Create tuple variables (aliases) for all tables referenced in SQL query

Query 16. Retrieve the name of each employee who has a dependent with the same first name and is the same sex as the employee.

```
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 );
```

Nested Queries

- Queries that are nested using the = or IN comparison operator can be collapsed into one single block

```
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 );
```

```
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;
```

Aggregate Functions in SQL

- Used to summarize information from multiple tuples into a single-tuple summary
- Built-in aggregate functions
  - `COUNT`, `SUM`, `MAX`, `MIN`, and `AVG`
- **Grouping**
  - Create subgroups of tuples before summarizing
- To select entire groups, `HAVING` clause is used
- Aggregate functions can be used in the `SELECT` clause or in a `HAVING` clause

Results of Aggregation

```
SELECT    SUM (Salary), MAX (Salary), MIN (Salary), AVG (Salary)
FROM      EMPLOYEE;
```

```
SELECT    SUM (Salary), MAX (Salary), MIN (Salary), AVG (Salary)
FROM      (EMPLOYEE JOIN DEPARTMENT ON Dno=Dnumber)
WHERE     Dname='Research';
```

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

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

The asterisk * refers to the rows (tuples), so COUNT returns the number of rows in the result

Results of Aggregation

```
SELECT    Lname, Fname
FROM      EMPLOYEE
WHERE     ( SELECT    COUNT (*)
            FROM  DEPENDENT
            WHERE     Ssn=Essn ) >= 2;
```

Aggregate Functions in SQL

- What happens to NULL values in aggregate functions?

```
SELECT    SUM (Salary), MAX (Salary), MIN (Salary), AVG (Salary)
FROM      EMPLOYEE;
```

Aggregate Functions on Booleans

- SOME and ALL may be applied as functions on Boolean values.
- SOME returns true if at least one element in the collection is TRUE (similar to OR)
- ALL returns true if all of the elements in the collection are TRUE (similar to AND)

Grouping: The GROUP BY Clause

- **Partition** relation/table into subsets of tuples
  - Based on **grouping attribute(s)**
  - Apply function to each such group independently
- **`GROUP BY`** clause
  - Specifies grouping attributes
- COUNT (*) counts the number of rows in the group

Examples of GROUP BY

- The grouping attribute must appear in the SELECT clause:

```
SELECT      Dno, COUNT (*), AVG (Salary)
FROM        EMPLOYEE
GROUP BY    Dno;
```

- How are NULLs handled?
- If the grouping attribute has NULL as a possible value, then a separate group is created for the null value (e.g., null Dno in the above query)

GROUP BY and HAVING Clauses

- **HAVING** clause
  - Provides a condition to select or reject an entire group:

```
SELECT    Pnumber, Pname, COUNT (*)
FROM      PROJECT, WORKS_ON
WHERE     Pnumber=Pno
GROUP BY  Pnumber, Pname;
```

```
SELECT    Pnumber, Pname, COUNT (*)
FROM      PROJECT, WORKS_ON
WHERE     Pnumber=Pno
GROUP BY  Pnumber, Pname
HAVING    COUNT (*) > 2;
```

Combining the WHERE and the HAVING Clause

- Suppose we want to count the ***total*** number of employees whose salaries exceed $40,000 in each department, but only for departments where more than five employees work.

```
SELECT      Dno, COUNT (*)
FROM        EMPLOYEE
WHERE       Salary>40000
GROUP BY    Dno
HAVING      COUNT (*) > 5;
```

```
SELECT      Dnumber, COUNT (*)
FROM        DEPARTMENT, EMPLOYEE
WHERE       Dnumber=Dno AND Salary>40000 AND
            ( SELECT      Dno
              FROM        EMPLOYEE
              GROUP BY Dno
              HAVING      COUNT (*) > 5)
```

- WHERE clause applies to tuple by tuple, whereas HAVING applies to entire group of tuples

Summary of SQL Query

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

Views (Virtual Tables) in SQL

- Concept of a view in SQL
  - Single table derived from other tables called the **defining tables**
  - Considered to be a virtual table that is not necessarily populated
  - Not stored in the hard-disk

Specification of Views in SQL

- **`CREATE VIEW`** command
  - Give table name, list of attribute names, and a query to specify the contents of the view

```
V1:   CREATE VIEW    WORKS_ON1
      AS SELECT      Fname, Lname, Pname, Hours
           FROM      EMPLOYEE, PROJECT, WORKS_ON
           WHERE     Ssn=Essn AND Pno=Pnumber;

V2:   CREATE VIEW    DEPT_INFO(Dept_name, No_of_emps, Total_sal)
      AS SELECT      Dname, COUNT (*), SUM (Salary)
           FROM      DEPARTMENT, EMPLOYEE
           WHERE     Dnumber=Dno
           GROUP BY  Dname;
```

  - In V1, attributes retain the names from base tables. In V2, attributes are assigned names

Specification of Views in SQL

- Once a View is defined, SQL queries can use the View relation in the FROM clause
- View is always up-to-date
  - Responsibility of the DBMS and not the user
- **`DROP VIEW`** command
  - Dispose of a view

DROP Command

- DROP command
  - Used to drop named schema elements, such as tables, domains, or constraint
- Drop behavior options:
  - CASCADE and RESTRICT
- Example:
  - DROP SCHEMA COMPANY CASCADE;
  - Removes the schema and all its elements including tables, views, constraints, etc.

Joining Tables in SQL

Joined Tables in SQL

- **Joined table**
  - Permits users to specify a table resulting from a join operation in the FROM clause of a query
- The FROM clause in the query below
  - Contains a single joined table

```
SELECT    Fname, Lname, Address
FROM      (EMPLOYEE JOIN DEPARTMENT ON Dno=Dnumber)
WHERE     Dname='Research';
```

Joined Tables in SQL

- Specify different types of join
  - NATURAL JOIN
  - Various types of OUTER JOIN
- NATURAL JOIN on two relations R and S
  - No join condition specified
  - Implicit EQUIJOIN condition for each pair of attributes with same name from R and S

```
Q1B:   SELECT   Fname, Lname, Address
       FROM     (EMPLOYEE NATURAL JOIN
                (DEPARTMENT AS DEPT (Dname, Dno, Mssn, Msdate)))
       WHERE    Dname='Research';
```

Implied join condition is EMPLOYEE.Dno=DEPT.Dno

Joined Tables in SQL

- **Inner join**
  - Default type of join in a joined table
  - Tuple is included in the result only if a matching tuple exists in the other relation

```
SELECT    E.Lname AS Employee_name, S.Lname AS Supervisor_name
FROM      EMPLOYEE AS E, EMPLOYEE AS S
WHERE     E.Super_ssn=S.Ssn;
```

An EMPLOYEE tuple whose value for Super_ssn is NULL is excluded.

Department Table

DEPTNO	DEPTNAME	MGRNO
A00	SPIFFY COMPUTER SERVICE DIV.	000010
B01	PLANNING	000020
C01	INFORMATION CENTER	000030
D01	DEVELOPMENT CENTER	-

Employee Table

EMPNO	LASTNAME	WORKDEPT
000010	HAAS	A00
000020	THOMPSON	B01
000030	KWAN	C01
000110	LUCCHESSI	A00
000120	O'CONNELL	A00
000130	QUINTANA	C01

Project Table

PROJNO	PROJNAME	DEPTNO	RESPEMP
AD3100	ADMIN SERVICES	D01	000010
IF1000	QUERY SERVICES	C01	000030
IF2000	USER EDUCATION	E01	000030
MA2100	WELD LINE AUTOMATION	D01	000010
PL2100	WELD LINE PLANNING	B01	000020

```
SELECT PROJNO, PROJNAME, P.DEPTNO, D.DEPTNO, DEPTNAME
FROM   PROJECT P INNER JOIN DEPARTMENT D
ON     P.DEPTNO = D.DEPTNO
```

Project Table

PROJNO	PROJNAME	DEPTNO	RESPEMP
AD3100	ADMIN SERVICES	D01	000010
IF1000	QUERY SERVICES	C01	000030
IF2000	USER EDUCATION	E01	000030
MA2100	WELD LINE AUTOMATION	D01	000010
PL2100	WELD LINE PLANNING	B01	000020

Department Table

DEPTNO	DEPTNAME	MGRNO
A00	SPIFFY COMPUTER SERVICE DIV.	000010
B01	PLANNING	000020
C01	INFORMATION CENTER	000030
D01	DEVELOPMENT CENTER	-

```
SELECT PROJNO, PROJNAME, P.DEPTNO, D.DEPTNO, DEPTNAME
FROM   PROJECT P INNER JOIN DEPARTMENT D
ON     P.DEPTNO = D.DEPTNO
```

PROJNO	PROJNAME	DEPTNO
AD3100	ADMIN SERVICES	D01
IF1000	QUERY SERVICES	C01
IF2000	USER EDUCATION	E01
MA2100	WELD LINE AUTOMATION	D01
PL2100	WELD LINE PLANNING	B01

DEPTNO	DEPTNAME
A00	SPIFFY COMPUTER SERVICE DIV.
B01	PLANNING
C01	INFORMATION CENTER
D01	DEVELOPMENT CENTER

ON

PROJNO	PROJNAME	P.DEPTNO	D.DEPTNO	DEPTNAME
AD3100	ADMIN SERVICES	D01	D01	DEVELOPMENT CENTER
IF1000	QUERY SERVICES	C01	C01	INFORMATION CENTER
MA2100	WELD LINE AUTOMATION	D01	D01	DEVELOPMENT CENTER
PL2100	WELD LINE PLANNING	B01	B01	PLANNING

Joined Tables in SQL

- LEFT OUTER JOIN
  - Every tuple in left table must appear in result
  - If no matching tuple
    - Padded with NULL values for attributes of right table

```
SELECT    E.Lname AS Employee_name,
          S.Lname AS Supervisor_name
FROM      (EMPLOYEE AS E LEFT OUTER JOIN EMPLOYEE AS S
          ON E.Super_ssn=S.Ssn);
```

```
SELECT PROJNO, PROJNAME, P.DEPTNO, D.DEPTNO, DEPTNAME
FROM   PROJECT P LEFT OUTER JOIN DEPARTMENT D
ON     P.DEPTNO = D.DEPTNO
```

Project Table

PROJNO	PROJNAME	DEPTNO	RESPEMP
AD3100	ADMIN SERVICES	D01	000010
IF1000	QUERY SERVICES	C01	000030
IF2000	USER EDUCATION	E01	000030
MA2100	WELD LINE AUTOMATION	D01	000010
PL2100	WELD LINE PLANNING	B01	000020

Department Table

DEPTNO	DEPTNAME	MGRNO
A00	SPIFFY COMPUTER SERVICE DIV.	000010
B01	PLANNING	000020
C01	INFORMATION CENTER	000030
D01	DEVELOPMENT CENTER	-

```
SELECT PROJNO, PROJNAME, P.DEPTNO, D.DEPTNO, DEPTNAME
FROM   PROJECT P LEFT OUTER JOIN DEPARTMENT D
ON     P.DEPTNO = D.DEPTNO
```

PROJNO	PROJNAME	DEPTNO
AD3100	ADMIN SERVICES	D01
IF1000	QUERY SERVICES	C01
IF2000	USER EDUCATION	E01
MA2100	WELD LINE AUTOMATION	D01
PL2100	WELD LINE PLANNING	B01

DEPTNO	DEPTNAME
A00	SPIFFY COMPUTER SERVICE DIV.
B01	PLANNING
C01	INFORMATION CENTER
D01	DEVELOPMENT CENTER

Preserved Row Table

ON

NULL-Supplying Table

PROJNO	PROJNAME	P.DEPTNO	D.DEPTNO	DEPTNAME
AD3100	ADMIN SERVICES	D01	D01	DEVELOPMENT CENTER
IF1000	QUERY SERVICES	C01	C01	INFORMATION CENTER
IF2000	USER EDUCATION	E01	-	-
MA2100	WELD LINE AUTOMATION	D01	D01	DEVELOPMENT CENTER
PL2100	WELD LINE PLANNING	B01	B01	PLANNING

Joined Tables in SQL

- RIGHT OUTER JOIN
  - Every tuple in right table must appear in result
  - If no matching tuple
    - Padded with NULL values for the attributes of left table

```
SELECT    E.Lname AS Employee_name,
          S.Lname AS Supervisor_name
FROM      (EMPLOYEE AS E RIGHT OUTER JOIN EMPLOYEE AS S
          ON E.Super_ssn=S.Ssn);
```

```
SELECT PROJNO, PROJNAME, P.DEPTNO, D.DEPTNO, DEPTNAME
FROM   PROJECT P RIGHT OUTER JOIN DEPARTMENT D
ON     P.DEPTNO = D.DEPTNO
```

Project Table

PROJNO	PROJNAME	DEPTNO	RESPEMP
AD3100	ADMIN SERVICES	D01	000010
IF1000	QUERY SERVICES	C01	000030
IF2000	USER EDUCATION	E01	000030
MA2100	WELD LINE AUTOMATION	D01	000010
PL2100	WELD LINE PLANNING	B01	000020

Department Table

DEPTNO	DEPTNAME	MGRNO
A00	SPIFFY COMPUTER SERVICE DIV.	000010
B01	PLANNING	000020
C01	INFORMATION CENTER	000030
D01	DEVELOPMENT CENTER	-

```
SELECT PROJNO, PROJNAME, P.DEPTNO, D.DEPTNO, DEPTNAME
FROM   PROJECT P RIGHT OUTER JOIN DEPARTMENT D
ON     P.DEPTNO = D.DEPTNO
```

PROJNO	PROJNAME	DEPTNO
AD3100	ADMIN SERVICES	D01
IF1000	QUERY SERVICES	C01
IF2000	USER EDUCATION	E01
MA2100	WELD LINE AUTOMATION	D01
PL2100	WELD LINE PLANNING	B01

DEPTNO	DEPTNAME
A00	SPIFFY COMPUTER SERVICE DIV.
B01	PLANNING
C01	INFORMATION CENTER
D01	DEVELOPMENT CENTER

NULL-Supplying Table

ON

Preserved Row Table

PROJNO	PROJNAME	P.DEPTNO	D.DEPTNO	DEPTNAME
-	-	-	A00	SPIFFY COMPUTER SERVICE DIV.
PL2100	WELD LINE PLANNING	B01	B01	PLANNING
IF1000	QUERY SERVICES	C01	C01	INFORMATION CENTER
AD3100	ADMIN SERVICES	D01	D01	DEVELOPMENT CENTER
MA2100	WELD LINE AUTOMATION	D01	D01	DEVELOPMENT CENTER

Joined Tables in SQL

- FULL OUTER JOIN
  - Combines the effect of applying both left and right joins
  - NULL values in every column of the table without a matching row

Employee table

LastName	DepartmentID
Rafferty	31
Jones	33
Heisenberg	33
Robinson	34
Smith	34
John	NULL

Department table

DepartmentID	DepartmentName
31	Sales
33	Engineering
34	Clerical
35	Marketing

```
SELECT *
FROM employee FULL OUTER JOIN department
  ON employee.DepartmentID = department.DepartmentID;
```

```
SELECT PROJNO, PROJNAME, P.DEPTNO, D.DEPTNO, DEPTNAME
FROM   PROJECT P FULL OUTER JOIN DEPARTMENT D
ON     P.DEPTNO = D.DEPTNO
```

Project Table

PROJNO	PROJNAME	DEPTNO	RESPEMP
AD3100	ADMIN SERVICES	D01	000010
IF1000	QUERY SERVICES	C01	000030
IF2000	USER EDUCATION	E01	000030
MA2100	WELD LINE AUTOMATION	D01	000010
PL2100	WELD LINE PLANNING	B01	000020

Department Table

DEPTNO	DEPTNAME	MGRNO
A00	SPIFFY COMPUTER SERVICE DIV.	000010
B01	PLANNING	000020
C01	INFORMATION CENTER	000030
D01	DEVELOPMENT CENTER	-

```
SELECT PROJNO, PROJNAME, P.DEPTNO, D.DEPTNO, DEPTNAME
FROM   PROJECT P FULL OUTER JOIN DEPARTMENT D
ON     P.DEPTNO = D.DEPTNO
```

PROJNO	PROJNAME	DEPTNO
AD3100	ADMIN SERVICES	D01
IF1000	QUERY SERVICES	C01
IF2000	USER EDUCATION	E01
MA2100	WELD LINE AUTOMATION	D01
PL2100	WELD LINE PLANNING	B01

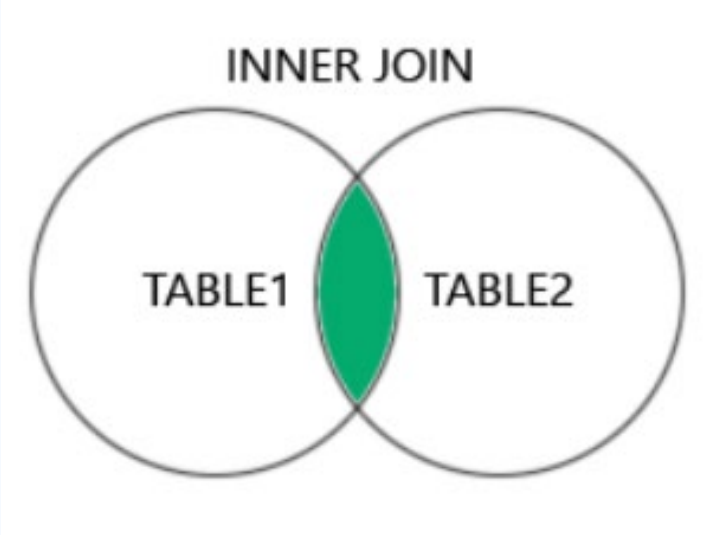
DEPTNO	DEPTNAME
A00	SPIFFY COMPUTER SERVICE DIV.
B01	PLANNING
C01	INFORMATION CENTER
D01	DEVELOPMENT CENTER

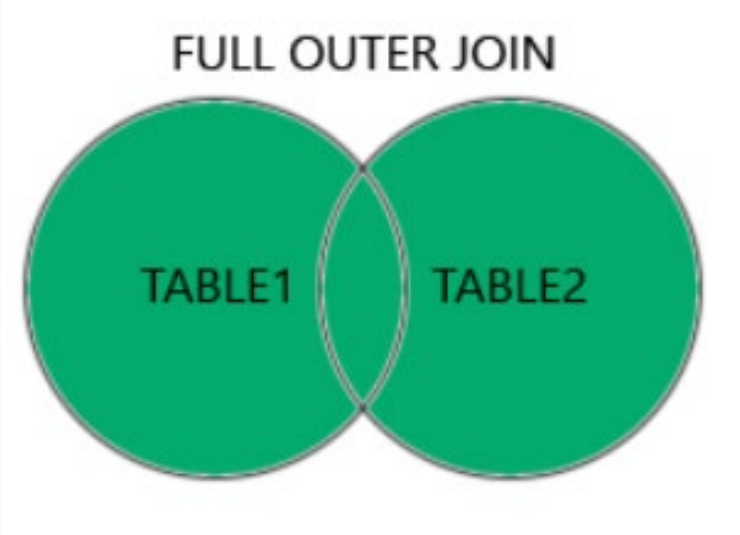
NULL-Supplying Table

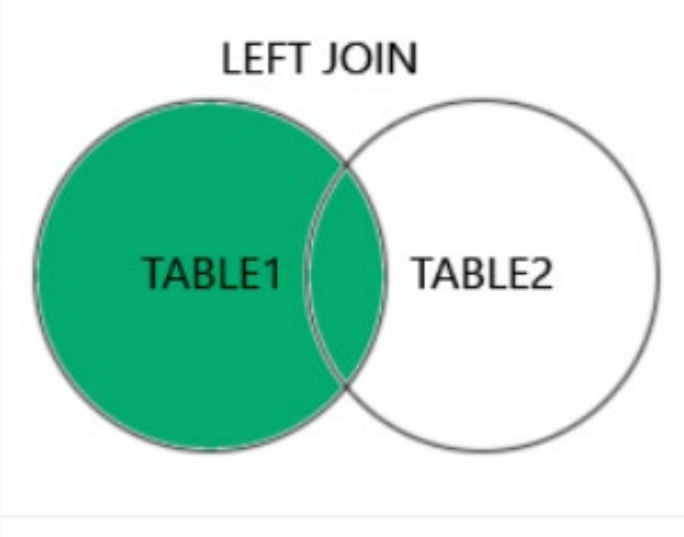
ON

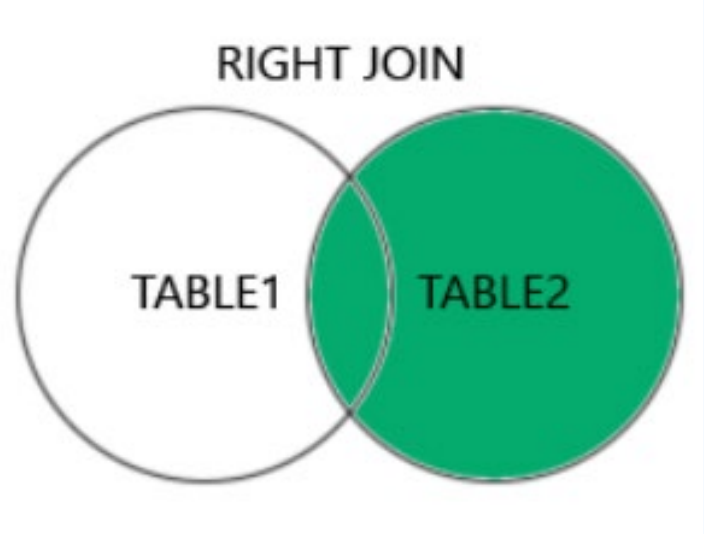
NULL-Supplying Table

PROJNO	PROJNAME	P.DEPTNO	D.DEPTNO	DEPTNAME
-	-	-	A00	SPIFFY COMPUTER SERVICE DIV.
PL2100	WELD LINE PLANNING	B01	B01	PLANNING
IF1000	QUERY SERVICES	C01	C01	INFORMATION CENTER
AD3100	ADMIN SERVICES	D01	D01	DEVELOPMENT CENTER
MA2100	WELD LINE AUTOMATION	D01	D01	DEVELOPMENT CENTER
IF2000	USER EDUCATION	E01	-	-


INNER JOIN
TABLE1
TABLE2


FULL OUTER JOIN
TABLE1
TABLE2


LEFT JOIN
TABLE1
TABLE2


RIGHT JOIN
TABLE1
TABLE2

Joined Tables in SQL

- Which JOIN to use in your queries?