Chapter 8

Query Formulation with SQL
(Supplement notes)

Outline

- Background
- Joining tables
- Summarizing tables
- Problem solving guidelines
- Advanced problems in SQL

What is SQL?

- Structured Query Language
- Language for database definition, manipulation, and control
- International standard
- Standalone and embedded usage
- Intergalactic database speak

4-3

SQL Standardization

- Relatively simple standard: SQL-86 and revision (SQL-89)
- Modestly complex standard: SQL-92
- Complex standards: SQL:1999 and SQL:2003

4-4

SELECT Statement Overview

```
SELECT <list of column expressions>
  FROM <list of tables and join operations>
  WHERE <list of logical expressions for rows>
  GROUP  BY <list of grouping columns>
  HAVING <list of logical expressions for groups>
  ORDER  BY <list of sorting specifications>
```

- Expression: combination of columns, constants, operators, and functions

4-5

University Database

Student (**StdSSN**, StdFirstName, StdLastName, StdCity, StdState, StdMajor, StdClass, StdGPA, StdZip)
Faculty (**FacSSN**, FacFirstName, FacLastName, FacCIty, FacState, FacDept, FacRank, FacSalary, FacSupervisor, FacHireDate, FacZipCode)
Course (**CourseNo**, CrsDesc, CrsUnits)
Offering (**OfferNo**, CourseNo, OffTerm, OffYear, OffLocation, OffTIme, FacSSN, OffDays)
Enrollment (**OfferNo**, **StdSSN**, EnrGrade)

4-6

First SELECT Examples

Example 1

```
SELECT * FROM Faculty
```

Example 2 (Access)

```
SELECT *
 FROM Faculty
 WHERE FacSSN = '543210987'
```

Example 3

```
SELECT FacFirstName, FacLastName, FacSalary
 FROM Faculty
```

Example 4

```
SELECT FacFirstName, FacLastName, FacSalary
 FROM Faculty
 WHERE FacSalary > 65000 AND FacRank = 'PROF'
```

4-7

Using Expressions

Example 5 (Access)

```
SELECT FacFirstName, FacLastName, FacCity,
       FacSalary*1.1 AS IncreasedSalary,
       FacHireDate
 FROM Faculty
 WHERE year(FacHireDate) > 1996
```

Example 5 (Oracle)

```
SELECT FacFirstName, FacLastName, FacCity,
       FacSalary*1.1 AS IncreasedSalary,
       FacHireDate
 FROM Faculty
 WHERE to_number(to_char(FacHireDate, 'YYYY'))
        > 1996
```

4-8

Inexact Matching

- Match against a pattern: LIKE operator
- Use meta characters to specify patterns
  - Wildcard (* or %)
  - Any single character (? or _)

Example 6 (Access)

```
SELECT *
 FROM Offering
 WHERE CourseNo LIKE 'IS*'
```

Example 6 (Oracle)

```
SELECT *
 FROM Offering
 WHERE CourseNo LIKE 'IS%'
```

4-9

Using Dates

- Dates are numbers
- Date constants and functions are not standard

Example 7 (Access)

```
SELECT FacFirstName, FacLastName, FacHireDate
 FROM Faculty
 WHERE FacHireDate BETWEEN #1/1/1999#
   AND #12/31/2000#
```

Example 7 (Oracle)

```
SELECT FacFirstName, FacLastName, FacHireDate
 FROM Faculty
 WHERE FacHireDate BETWEEN '1-Jan-1999'
   AND '31-Dec-2000'
```

4-10

Other Single Table Examples

Example 8: Testing for null values

```
SELECT OfferNo, CourseNo
 FROM Offering
 WHERE FacSSN IS NULL AND OffTerm = 'SUMMER'
   AND OffYear = 2006
```

Example 9: Mixing AND and OR

```
SELECT OfferNo, CourseNo, FacSSN
 FROM Offering
 WHERE (OffTerm = 'FALL' AND OffYear = 2005)
    OR (OffTerm = 'WINTER' AND OffYear = 2006)
```

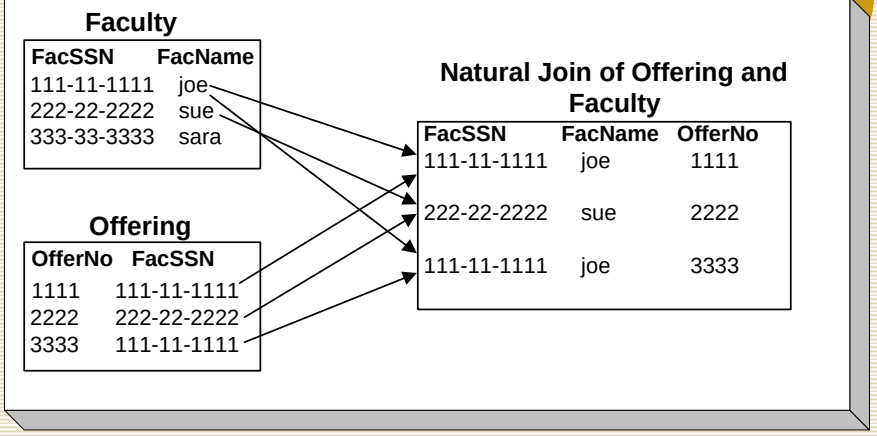
4-11

Join Operator

- Most databases have many tables
- Combine tables using the join operator
- Specify matching condition
  - Can be any comparison but usually =
  - PK = FK most common join condition
  - Relationship diagram useful when combining tables

4-12

Join Example



4-13

Cross Product Style

- List tables in the FROM clause
- List join conditions in the WHERE clause

Example 10 (Access)

```
SELECT OfferNo, CourseNo, FacFirstName,
       FacLastName
 FROM Offering, Faculty
 WHERE OffTerm = 'FALL' AND OffYear = 2005
   AND FacRank = 'ASST' AND CourseNo LIKE 'IS*'
   AND Faculty.FacSSN = Offering.FacSSN
```

4-14

Join Operator Style

- Use INNER JOIN and ON keywords
- FROM clause contains join operations

Example 11 (Access)

```
SELECT OfferNo, CourseNo, FacFirstName,
       FacLastName
 FROM Offering INNER JOIN Faculty
   ON Faculty.FacSSN = Offering.FacSSN
 WHERE OffTerm = 'FALL' AND OffYear = 2005
   AND FacRank = 'ASST' AND CourseNo LIKE 'IS*'
```

4-15

Name Qualification

- Ambiguous column reference
  - More than one table in the query contains a column referenced in the query
  - Ambiguity determined by the query not the database
- Use column name alone if query is not ambiguous
- Qualify with table name if query is ambiguous
- Readability versus writability

4-16

Summarizing Tables

- Row summaries important for decision-making tasks
- Row summary
  - Result contains statistical (aggregate) functions
  - Conditions involve statistical functions
- SQL keywords
  - Aggregate functions in the output list
  - GROUP BY: summary columns
  - HAVING: summary conditions

4-17

GROUP BY Examples

Example 12: Grouping on a single column

```
SELECT FacRank, AVG(FacSalary) AS AvgSalary
  FROM Faculty
  GROUP BY FacRank
```

Example 13: Row and group conditions

```
SELECT StdMajor, AVG(StdGPA) AS AvgGpa
  FROM Student
  WHERE StdClass IN ('JR', 'SR')
  GROUP BY StdMajor
  HAVING AVG(StdGPA) > 3.1
```

4-18

SQL Summarization Rules

- Columns in SELECT and GROUP BY
  - SELECT: non aggregate and aggregate columns
  - GROUP BY: list all non aggregate columns
- WHERE versus HAVING
  - Row conditions in WHERE
  - Group conditions in HAVING

4-19

Summarization and Joins

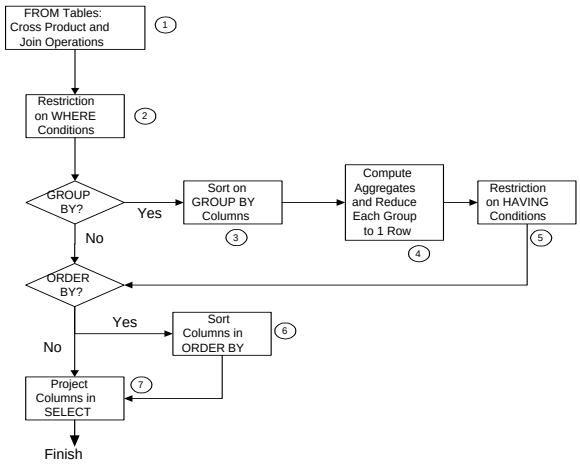
- Powerful combination
- List join conditions in the WHERE clause

Example 14: List the number of students enrolled in each fall 2003 offering.

```
SELECT Offering.OfferNo,
       COUNT(*) AS NumStudents
  FROM Enrollment, Offering
  WHERE Offering.OfferNo = Enrollment.OfferNo
    AND OffYear = 2006
  GROUP BY Offering.OfferNo
```

4-20

Conceptual Evaluation Process



4-21

Conceptual Evaluation Lessons

- Row operations before group operations
  - FROM and WHERE before GROUP BY and HAVING
  - Check row operations first
- Grouping occurs only one time
- Use small sample tables

4-22

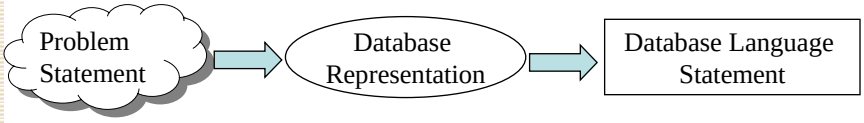
Conceptual Evaluation Problem

Example 15: List the number of offerings taught in 2006 by faculty rank and department. Exclude combinations of faculty rank and department with less than two offerings taught.

```
SELECT FacRank, FacDept,
       COUNT(*) AS NumOfferings
 FROM Faculty, Offering
 WHERE Offering.FacSSN = Faculty.FacSSN
   AND OffYear = 2006
 GROUP BY FacRank, FacDept
 HAVING COUNT(*) > 1
```

4-23

Query Formulation Process



4-24

Critical Questions

- What tables?
  - Columns in output
  - Conditions to test (including join conditions)
- How to combine the tables?
  - Usually join PK to FK
  - More complex ways to combine
- Individual rows or groups of rows?
  - Aggregate functions in output
  - Conditions with aggregate functions

4-25

Efficiency Considerations

- Little concern for efficiency
- Intelligent SQL compilers
- Correct and non redundant solution
  - No extra tables
  - No unnecessary grouping
  - Use HAVING for group conditions only

4-26

Advanced Problems

- Joining multiple tables
- Self joins
- Grouping after joining multiple tables
- Traditional set operators

4-27

Joining Three Tables

Example 16: List Leonard Vince's teaching schedule in fall 2005. For each course, list the offering number, course number, number of units, days, location, and time.

```
SELECT OfferNo, Offering.CourseNo, OffDays,
       CrsUnits, OffLocation, OffTime
 FROM Faculty, Course, Offering
 WHERE Faculty.FacSSN = Offering.FacSSN
   AND Offering.CourseNo = Course.CourseNo
   AND OffYear = 2005 AND OffTerm = 'FALL'
   AND FacFirstName = 'Leonard'
   AND FacLastName = 'Vince'
```

4-28

Joining Four Tables

Example 17: List Bob Norbert's course schedule in spring 2006. For each course, list the offering number, course number, days, location, time, and faculty name.

```
SELECT Offering.OfferNo, Offering.CourseNo,
       OffDays, OffLocation, OffTime,
       FacFirstName, FacLastName
 FROM Faculty, Offering, Enrollment, Student
 WHERE Offering.OfferNo = Enrollment.OfferNo
   AND Student.StdSSN = Enrollment.StdSSN
   AND Faculty.FacSSN = Offering.FacSSN
   AND OffYear = 2006 AND OffTerm = 'SPRING'
   AND StdFirstName = 'BOB'
   AND StdLastName = 'NORBERT'
```

4-29

Self-Join

- Join a table to itself
- Usually involve a self-referencing relationship
- Useful to find relationships among rows of the same table
  - Find subordinates within a preset number of levels
  - Find subordinates within any number of levels requires embedded SQL

4-30

Self-Join Example

Example 18: List faculty members who have a higher salary than their supervisor. List the social security number, name, and salary of the faculty and supervisor.

```
SELECT Subr.FacSSN, Subr.FacLastName,
       Subr.FacSalary, Supr.FacSSN,
       Supr.FacLastName, Supr.FacSalary
 FROM Faculty Subr, Faculty Supr
 WHERE Subr.FacSupervisor = Supr.FacSSN
   AND Subr.FacSalary > Supr.FacSalary
```

4-31

Multiple Joins Between Tables

Example 19: List the names of faculty members and the course number for which the faculty member teaches the same course number as his or her supervisor in 2006.

```
SELECT FacFirstName, FacLastName, O1.CourseNo
 FROM Faculty, Offering O1, Offering O2
 WHERE Faculty.FacSSN = O1.FacSSN
   AND Faculty.FacSupervisor = O2.FacSSN
   AND O1.OffYear = 2006 AND O2.OffYear = 2006
   AND O1.CourseNo = O2.CourseNo
```

4-32

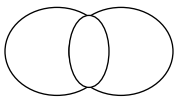
Multiple Column Grouping

Example 20: List the course number, the offering number, and the number of students enrolled. Only include courses offered in spring 2006.

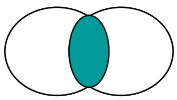
```
SELECT CourseNo, Enrollment.OfferNo,
       Count(*) AS NumStudents
 FROM Offering, Enrollment
 WHERE Offering.OfferNo = Enrollment.OfferNo
   AND OffYear = 2006 AND OffTerm = 'SPRING'
 GROUP BY Enrollment.OfferNo, CourseNo
```

4-33

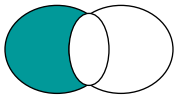
Traditional Set Operators



A UNION B



A INTERSECT B



A MINUS B

4-34

Union Compatibility

- Requirement for the traditional set operators
- Strong requirement
  - Same number of columns
  - Each corresponding column is compatible
  - Positional correspondence
- Apply to similar tables by removing columns first

4-35

SQL UNION Example

Example 21: Retrieve basic data about all university people

```
SELECT
   FacSSN AS SSN, FacFirstName AS FirstName,
     FacLastName AS LastName, FacCity AS City,
     FacState AS State
 FROM Faculty
   UNION
SELECT
   StdSSN AS SSN, StdFirstName AS FirstName,
     StdLastName AS LastName, StdCity AS City,
     StdState AS State
 FROM Student
```

4-36

Oracle INTERSECT Example

Example 22: Show teaching assistants, faculty who are students. Only show the common columns in the result.

```
SELECT FacSSN AS SSN, FacFirstName AS
       FirstName, FacLastName AS LastName,
       FacCity AS City, FacState AS State
 FROM Faculty
   INTERSECT
SELECT StdSSN AS SSN, StdFirstName AS
       FirstName, StdLastName AS LastName,
       StdCity AS City, StdState AS State
 FROM Student
```

4-37

Oracle MINUS Example

Example 23: Show faculty who are not students (pure faculty). Only show the common columns in the result.

```
SELECT FacSSN AS SSN, FacFirstName AS
       FirstName, FacLastName AS LastName,
       FacCity AS City, FacState AS State
 FROM Faculty
   MINUS
SELECT StdSSN AS SSN, StdFirstName AS
       FirstName, StdLastName AS LastName,
       StdCity AS City, StdState AS State
 FROM Student
```

4-38

Data Manipulation Statements

- INSERT: adds one or more rows
- UPDATE: modifies one or more rows
- DELETE: removes one or more rows
- Use SELECT statement to INSERT multiple rows
- UPDATE and DELETE can use a WHERE clause
- Not as widely used as SELECT statement

4-39

INSERT Example

Example 24: Insert a row into the *Student* table supplying values for all columns.

```
INSERT INTO Student
 (StdSSN, StdFirstName, StdLastName,
  StdCity, StdState, StdZip, StdClass,
  StdMajor, StdGPA)
 VALUES
('999999999','JOE','STUDENT','SEATAC',
   'WA','98042-1121','FR','IS', 0.0)
```

4-40

UPDATE Example

Example 25: Change the major and class of Homer Wells.

```
UPDATE Student
 SET StdMajor = 'ACCT',
     StdClass = 'SO'
 WHERE StdFirstName = 'HOMER'
   AND StdLastName = 'WELLS'
```

4-41

DELETE Example

Example 26: Delete all CS majors who are seniors.

```
DELETE FROM Student
 WHERE StdMajor = ‘CS'
   AND StdClass = 'SR'
```

4-42

Advance problems in SQL

- Outer join problems
- Type I nested queries
- Type II nested queries and difference problems
- Nested queries in the FROM clause
- Division problems
- Null value effects

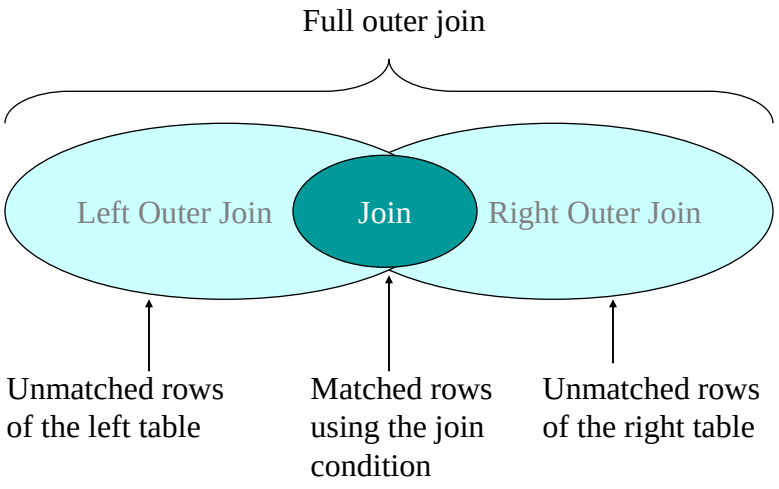
4-43

Outer Join Overview

- Join excludes non matching rows
- Preserving non matching rows is important in some business situations
- Outer join variations
  - Full outer join
  - One-sided outer join

4-44

Outer Join Operators



4-45

Full Outer Join Example

Faculty

FacSSN	FacName
111-11-1111	joe
222-22-2222	sue
333-33-3333	sara

Offering

Offerno	FacSSN
1111	111-11-1111
2222	222-22-2222
3333	111-11-1111
4444	

Outer Join of Offering and Faculty

FacSSN	FacName	OfferNo
111-11-1111	joe	1111
222-22-2222	sue	2222
111-11-1111	joe	3333
333-33-3333	sara	
		4444

4-46

LEFT JOIN and RIGHT JOIN Keywords

Example 27 (Access)

```
SELECT OfferNo, CourseNo, Offering.FacSSN,
       FacFirstName, FacLastName
  FROM Offering LEFT JOIN Faculty
       ON Offering.FacSSN = Faculty.FacSSN
  WHERE CourseNo LIKE 'IS*'
```

Example 28 (Oracle)

```
SELECT OfferNo, CourseNo, Offering.FacSSN,
       FacFirstName, FacLastName
  FROM Faculty RIGHT JOIN Offering
       ON Offering.FacSSN = Faculty.FacSSN
  WHERE CourseNo LIKE 'IS%'
```

4-47

Full Outer Join Example I

Example 29 (SQL:2003 and Oracle 9i/10g)

```
SELECT FacSSN, FacFirstName, FacLastName,
       FacSalary, StdSSN, StdFirstName,
       StdLastName, StdGPA
  FROM Faculty FULL JOIN Student
       ON Student.StdSSN = Faculty.FacSSN
```

4-48

Full Outer Join Example II

Example 30 (Access)

```
SELECT FacSSN, FacFirstName, FacLastName,
       FacSalary, StdSSN, StdFirstName,
       StdLastName, StdGPA
 FROM Faculty RIGHT JOIN Student
      ON Student.StdSSN = Faculty.FacSSN
          UNION
SELECT FacSSN, FacFirstName, FacLastName,
       FacSalary, StdSSN, StdFirstName,
       StdLastName, StdGPA
 FROM Faculty LEFT JOIN Student
      ON Student.StdSSN = Faculty.FacSSN
```

4-49

Mixing Inner and Outer Joins I

Example 31 (Access)

```
SELECT OfferNo, Offering.CourseNo, OffTerm,
       CrsDesc, Faculty.FacSSN, FacLastName
 FROM ( Faculty RIGHT JOIN Offering
      ON Offering.FacSSN = Faculty.FacSSN )
   INNER JOIN Course
      ON Course.CourseNo = Offering.CourseNo
 WHERE Course.CourseNo LIKE 'IS*'
```

4-50

Type I Nested Queries

- Query inside a query
- Use in WHERE and HAVING conditions
- Similar to a nested procedure
- Executes one time
- No reference to outer query
- Also known as non-correlated or independent nested query

4-51

Type I Nested Query Examples I

Example 32 (Access): List finance faculty who teach IS courses.

```
SELECT FacSSN, FacLastName, FacDept
 FROM Faculty
 WHERE FacDept = 'FIN' AND FacSSN IN
  ( SELECT FacSSN FROM Offering
     WHERE CourseNo LIKE 'IS*'  )
```

4-52

Type I Nested Query Examples II

Example 33 (Oracle): List finance faculty who teach 4 unit IS courses.

```
SELECT FacSSN, FacLastName, FacDept
 FROM Faculty
 WHERE FacDept = 'FIN' AND FacSSN IN
  ( SELECT FacSSN FROM Offering
     WHERE CourseNo LIKE 'IS%' AND CourseNo IN
      ( SELECT CourseNo FROM Course
         WHERE CrsUnits = 4 ) )
```

4-53

DELETE Example

- Use Type I nested queries to test conditions on other tables
- Use for UPDATE statements also

Example 34: Delete offerings taught by Leonard Vince.

```
DELETE FROM Offering
 WHERE Offering.FacSSN IN
  ( SELECT FacSSN FROM Faculty
     WHERE FacFirstName = 'Leonard'
       AND FacLastName = 'Vince' )
```

4-54

Type II Nested Queries

- Similar to nested loops
- Executes one time for each row of outer query
- Reference to outer query
- Also known as correlated or variably nested query
- Use for difference problems not joins

4-55

Type II Nested Query Example for a Difference Problem

Example 35: Retrieve MS faculty who are not teaching in winter 2006.

```
SELECT FacSSN, FacLastName, FacDept
 FROM Faculty
 WHERE FacDept = 'MS' AND NOT EXISTS
  ( SELECT * FROM Offering
     WHERE OffTerm = 'WINTER'
       AND OffYear = 2006
       AND Faculty.FacSSN = Offering.FacSSN )
```

4-56

Limited Formulations for Difference Problems

- Type I nested query with NOT IN condition
- One-sided outer join with IS NULL condition
- Difference operation using MINUS (EXCEPT) operator

4-57

Type I Difference Formulation

Example 36: Retrieve MS faculty who are not teaching in winter 2006.

```
SELECT FacSSN, FacLastName, FacDept
 FROM Faculty
 WHERE FacDept = 'MS' AND FacSSN NOT IN
  ( SELECT FacSSN FROM Offering
     WHERE OffTerm = 'WINTER'
       AND OffYear = 2006 )
```

4-58

One-Sided Outer Join Difference Formulation

Example 37: Retrieve MS faculty who have never taught a course (research faculty).

```
SELECT FacSSN, FacLastName, FacDept
 FROM Faculty LEFT JOIN Offering
   ON Faculty.FacSSN = Offering.FacSSN
 WHERE FacDept = 'MS'
   AND Offering.FacSSN IS NULL
```

4-59

MINUS Operator Difference Formulation

Example 38 (Oracle): Retrieve faculty who are not students

```
SELECT FacSSN AS SSN, FacFirstName AS FirstName,
       FacLastName AS LastName, FacCity AS City,
       FacState AS State
 FROM Faculty
      MINUS
SELECT StdSSN AS SSN, StdFirstName AS FirstName,
       StdLastName AS LastName, StdCity AS City,
       StdState AS State
 FROM Student
```

4-60

Nested Queries in the FROM Clause

- More recent introduction than nested queries in the WHERE and HAVING clauses
- Consistency in language design
- Wherever table appears, table expression can appear
- Specialized uses
  - Nested aggregates
  - Multiple independent aggregate calculations

4-61

Nested FROM Query Example

Example 39: Retrieve the course number, course description, the number of offerings, and the average enrollment across offering.

```
SELECT T.CourseNo, T.CrsDesc,
       COUNT(*) AS NumOfferings,
       Avg(T.EnrollCount) AS AvgEnroll
 FROM
  (SELECT Course.CourseNo, CrsDesc,
          Offering.OfferNo,
          COUNT(*) AS EnrollCount
   FROM Offering, Enrollment, Course
   WHERE Offering.OfferNo = Enrollment.OfferNo
     AND Course.CourseNo = Offering.CourseNo
   GROUP BY Course.CourseNo, CrsDesc,
         Offering.OfferNo) T
 GROUP BY T.CourseNo, T.CrsDesc
```

4-62

Divide Operator

- Match on a subset of values
  - Suppliers who supply all parts
  - Faculty who teach every IS course
- Specialized operator
- Typically applied to associative tables representing M-N relationships

4-63

Division Example

SuppPart

SuppNo	PartNo
s3	p1
s3	p2
s3	p3
s0	p1
s1	p2

Part

PartNo
p1
p2
p3

SuppPart DIVIDEBY Part

SuppNo
s3

s3 {p1, p2, p3}
contains {p1, p2, p3}

4-64

COUNT Method for Division Problems

- Compare the number of rows associated with a group to the total number in the subset of interest
- Type I nested query in the HAVING clause

Example 40: List the students who belong to all clubs.

```
SELECT StdNo
 FROM StdClub
 GROUP BY StdNo
 HAVING COUNT(*) =
  ( SELECT COUNT(*) FROM Club )
```

4-65

Typical Division Problems

- Compare to an interesting subset rather than entire table
- Use similar conditions in outer and nested query

Example 41: List the students who belong to all social clubs.

```
SELECT Student1.StdNo, SName
 FROM StdClub, Club, Student1
 WHERE StdClub.ClubNo = Club.ClubNo
   AND Student1.StdNo = StdClub.StdNo
   AND CPurpose = 'SOCIAL'
 GROUP BY Student1.StdNo, SName
 HAVING COUNT(*) =
  ( SELECT COUNT(*) FROM Club
     WHERE CPurpose = 'SOCIAL' )
```

4-66

Advanced Division Problems

- Count distinct values rather than rows
  - Faculty who teach at least one section of selected course offerings
  - Offering table has duplicate CourseNo values
- Use COUNT(DISTINCT column)
- Use stored query or nested FROM query in Access

4-67

Advanced Division Problem Example

Example 42: List the SSN and the name of faculty who teach at least one section of all of the fall 2005, IS courses.

```
SELECT Faculty.FacSSN, FacFirstName,
       FacLastName
 FROM Faculty, Offering
 WHERE Faculty.FacSSN = Offering.FacSSN
   AND OffTerm = 'FALL' AND CourseNo LIKE 'IS%'
   AND OffYear = 2005
 GROUP BY Faculty.FacSSN, FacFirstName,
          FacLastName
 HAVING COUNT(DISTINCT CourseNo) =
  ( SELECT COUNT(DISTINCT CourseNo)
     FROM Offering
     WHERE OffTerm = 'FALL' AND OffYear = 2005
       AND CourseNo LIKE 'IS%' )
```

4-68

Null Value Effects

- Simple conditions
- Compound conditions
- Grouping and aggregate functions
- SQL:2003 standard but implementation may vary

4-69

Simple Conditions

- Simple condition is null if either left-hand or right-hand side is null.
- Discard rows evaluating to false or null
- Retain rows evaluating to true
- Rows evaluating to null will not appear in the result of the simple condition or its negation

4-70

Compound Conditions

AND	True	False	Null
True	True	False	Null
False	False	False	False
Null	Null	False	Null

OR	True	False	Null
True	True	True	True
False	True	False	Null
Null	True	Null	Null

NOT	True	False	Null
	False	True	Null

4-71

Aggregate Functions

- Null values ignored
- Effects can be subtle
  - COUNT(*) may differ from Count(Column)
  - SUM(Column1) + SUM(Column2) may differ from SUM(Column1 + Column2)

4-72

Grouping Effects

- Rows with null values are grouped together
- Grouping column contains null values
- Null group can be placed at beginning or end of the non-null groups

4-73

Oracle 8i Notation for One-Sided Outer Joins

Example A1 (Oracle 8i)

```
SELECT OfferNo, CourseNo, Offering.FacSSN,
       FacFirstName, FacLastName
  FROM Faculty, Offering
  WHERE Offering.FacSSN = Faculty.FacSSN (+)
    AND CourseNo LIKE 'IS%'
```

Example A2 (Oracle 8i)

```
SELECT OfferNo, CourseNo, Offering.FacSSN,
       FacFirstName, FacLastName
  FROM Faculty, Offering
  WHERE Faculty.FacSSN (+) = Offering.FacSSN
    AND CourseNo LIKE 'IS%'
```

4-74

Full Outer Join Example III

Example A3 (Oracle 8i)

```
SELECT FacSSN, FacFirstName, FacLastName,
       FacSalary, StdSSN, StdFirstName,
       StdLastName, StdGPA
  FROM Faculty, Student
  WHERE Student.StdSSN = Faculty.FacSSN (+)
          UNION
SELECT FacSSN, FacFirstName, FacLastName,
       FacSalary, StdSSN, StdFirstName,
       StdLastName, StdGPA
  FROM Faculty, Student
  WHERE Student.StdSSN (+) = Faculty.FacSSN
```

4-75

Mixing Inner and Outer Joins II

Example A4 (Oracle 8i)

```
SELECT OfferNo, Offering.CourseNo, OffTerm,
       CrsDesc, Faculty.FacSSN, FacLastName
  FROM  Faculty, Course, Offering
  WHERE Offering.FacSSN = Faculty.FacSSN (+)
    AND Course.CourseNo = Offering.CourseNo
    AND Course.CourseNo LIKE 'IS%'
```

4-76

Summary

- SQL is a broad language
- SELECT statement is complex
- Advanced matching problems not common but important when necessary
- Understand outer join, difference, and division operators
- Nested queries important for advanced matching problems
- Lots of practice to master query formulation and SQL

4-77