Business School

UNIVERSITY OF COLORADO DENVER

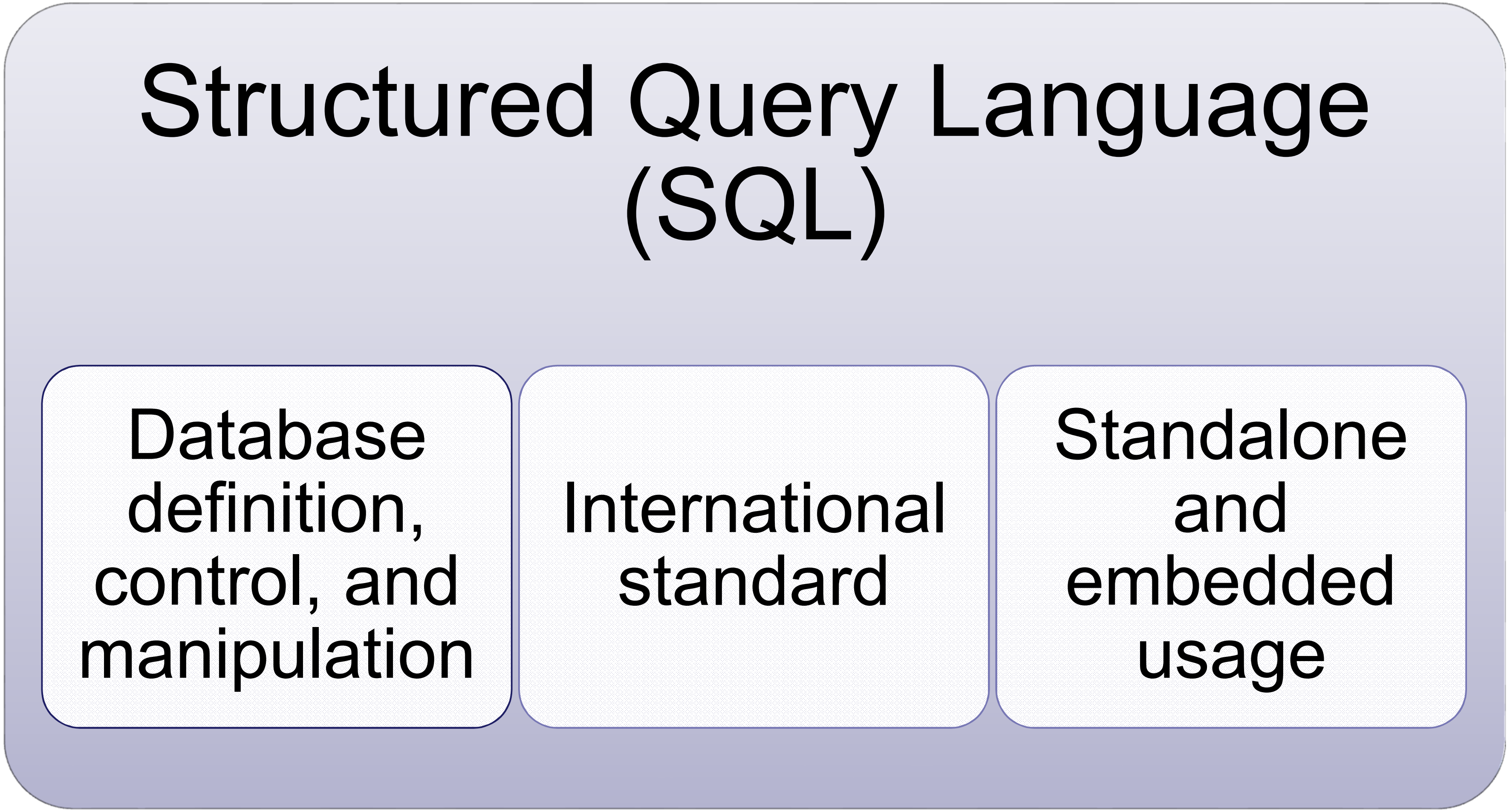
Information Systems Program

Module 4
Basic Query Formulation with SQL

Lesson 1: SQL Overview

Lesson Objectives

- Briefly explain the three types of SQL statements
- Gain insight about the level of conformance for the SQL standard



Structured Query Language (SQL)

Database definition, control, and manipulation

International standard

Standalone and embedded usage

Major SQL Statements

Statement	Statement Type
CREATE TABLE	Definitional, Control
CREATE VIEW	Definitional
CREATE TYPE	Definitional
SELECT	Manipulation
INSERT, UPDATE, DELETE	Manipulation
COMMIT, ROLLBACK	Manipulation
CREATE TRIGGER	Control, Manipulation
GRANT, REVOKE	Control

SQL Conformance

- No official conformance testing
- Vendor claims about conformance
- Reasonable conformance on Core SQL
- Large variance on conformance outside of Core SQL
- Difficult to write portable SQL code outside of Core SQL

Summary

- SQL: broad language with weak conformance
- SELECT statement: important and complex
- Lots of practice to master query formulation and the SELECT statement

Business School
UNIVERSITY OF COLORADO DENVER

Information Systems Program

Module 4
Basic Query Formulation with SQL

Lesson 2: SELECT Statement Introduction

Lesson Objectives

- Write SQL SELECT statements involving row conditions on single tables
- Execute SELECT statements
- Write and execute practice problems

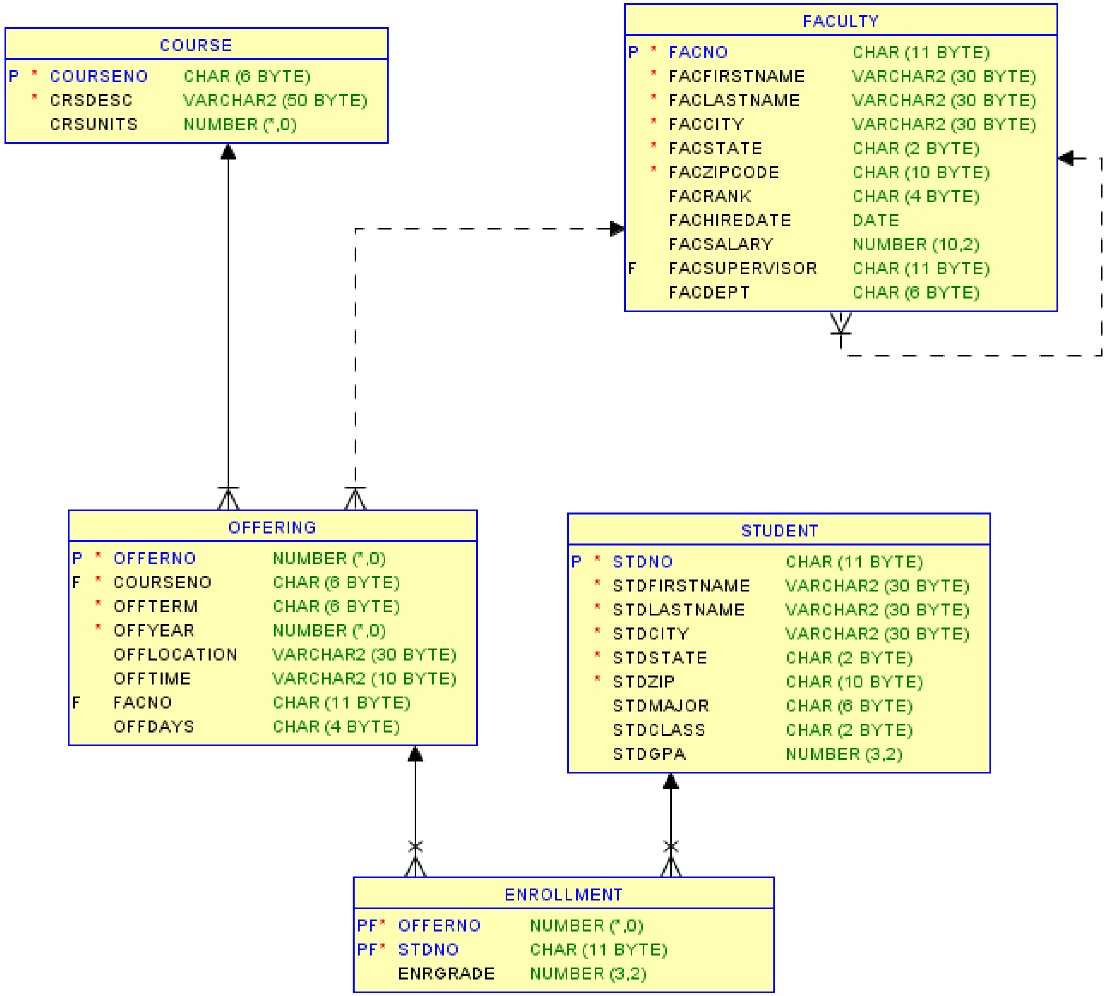
SELECT Statement Overview

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

- Column expression: combination of columns, constants, operators, and functions
  - `FacSalary * 1.1`

- Logical expression: conditions connected by AND, OR, and NOT
  - `OffTerm = 'FALL' AND OffYear = 2016`

University Database Diagram



COURSE

P	*	COURSENO	CHAR (6 BYTE)
	*	CRSDESC	VARCHAR2 (50 BYTE)
		CRSUNITS	NUMBER (*,0)

FACULTY

P	*	FACNO	CHAR (11 BYTE)
	*	FACFIRSTNAME	VARCHAR2 (30 BYTE)
	*	FACLASTNAME	VARCHAR2 (30 BYTE)
	*	FACCITY	VARCHAR2 (30 BYTE)
	*	FACSTATE	CHAR (2 BYTE)
	*	FACZIPCODE	CHAR (10 BYTE)
		FACRANK	CHAR (4 BYTE)
		FACHIREDATE	DATE
		FACSALARY	NUMBER (10,2)
F		FACSUPERVISOR	CHAR (11 BYTE)
		FACDEPT	CHAR (6 BYTE)

OFFERING

P	*	OFFERNO	NUMBER (*,0)
F	*	COURSENO	CHAR (6 BYTE)
	*	OFFTERM	CHAR (6 BYTE)
	*	OFFYEAR	NUMBER (*,0)
		OFFLOCATION	VARCHAR2 (30 BYTE)
		OFFTIME	VARCHAR2 (10 BYTE)
F		FACNO	CHAR (11 BYTE)
		OFFDAYS	CHAR (4 BYTE)

STUDENT

P	*	STDNO	CHAR (11 BYTE)
	*	STDFIRSTNAME	VARCHAR2 (30 BYTE)
	*	STDLASTNAME	VARCHAR2 (30 BYTE)
	*	STDCITY	VARCHAR2 (30 BYTE)
	*	STDSTATE	CHAR (2 BYTE)
	*	STDZIP	CHAR (10 BYTE)
		STDMAJOR	CHAR (6 BYTE)
		STDCLASS	CHAR (2 BYTE)
		STDGPA	NUMBER (3,2)

ENROLLMENT

PF	*	OFFERNO	NUMBER (*,0)
PF	*	STDNO	CHAR (11 BYTE)
		ENRGRADE	NUMBER (3,2)

First SELECT Examples

Example 1

```
SELECT * FROM Faculty;
```

Example 2

```
SELECT *
 FROM Faculty
 WHERE FacNo = '543-21-0987';
```

Example 3

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

Example 4

```
SELECT FacCity, FacState
 FROM Faculty;

SELECT DISTINCT FacCity, FacState
 FROM Faculty;
```

Using Expressions

Example 5 (Oracle)

```
SELECT FacFirstName, FacLastName, FacCity,
       FacSalary*1.1 AS IncreasedSalary,
       FacHireDate
 FROM Faculty
 WHERE to_number(to_char(FacHireDate, 'YYYY')) > 2005;
```

Example 5 (MySQL)

```
 SELECT FacFirstName, FacLastName, FacCity,
        FacSalary*1.1 AS IncreasedSalary,
        FacHireDate
  FROM Faculty
  WHERE (date_format(FacHireDate, '%Y')) > 2005;
```

Inexact Text Matching

- Use LIKE operator to match against a pattern
- Use meta characters to specify patterns
  - Wildcard (%)
  - Any single character (_)

Example 6

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

Using Date Constants

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

Example 7 (Oracle)

```
SELECT FacFirstName, FacLastName, FacHireDate
 FROM Faculty
 WHERE FacHireDate BETWEEN '1-Jan-2008'
   AND '31-Dec-2009';
```

Example 7 (MySQL)

```
SELECT FacFirstName, FacLastName, FacHireDate
 FROM Faculty
 WHERE FacHireDate BETWEEN '2008-01-01'
   AND '2009-12-31';
```

Combining Conditions

Example 8: Testing for null values

```
SELECT OfferNo, CourseNo
 FROM Offering
 WHERE FacNo IS NULL AND OffTerm = 'SUMMER'
   AND OffYear = 2017;
```

Example 9: Mixing AND and OR

```
SELECT OfferNo, CourseNo, FacNo
 FROM Offering
 WHERE (OffTerm = 'FALL' AND OffYear = 2016)
    OR (OffTerm = 'WINTER' AND OffYear = 2017);
```

Summary

- Subset of the SELECT statement
- Background about database details essential for query formulation
- Lots of practice to master query formulation and SQL

Business School
UNIVERSITY OF COLORADO DENVER

Information Systems Program

Module 4
Basic Query Formulation with SQL

Lesson 3: Join Operator

Lesson Objectives

- Create the result of a join operation on sample tables
- Briefly explain the components of the natural join operator

Natural Join Example I

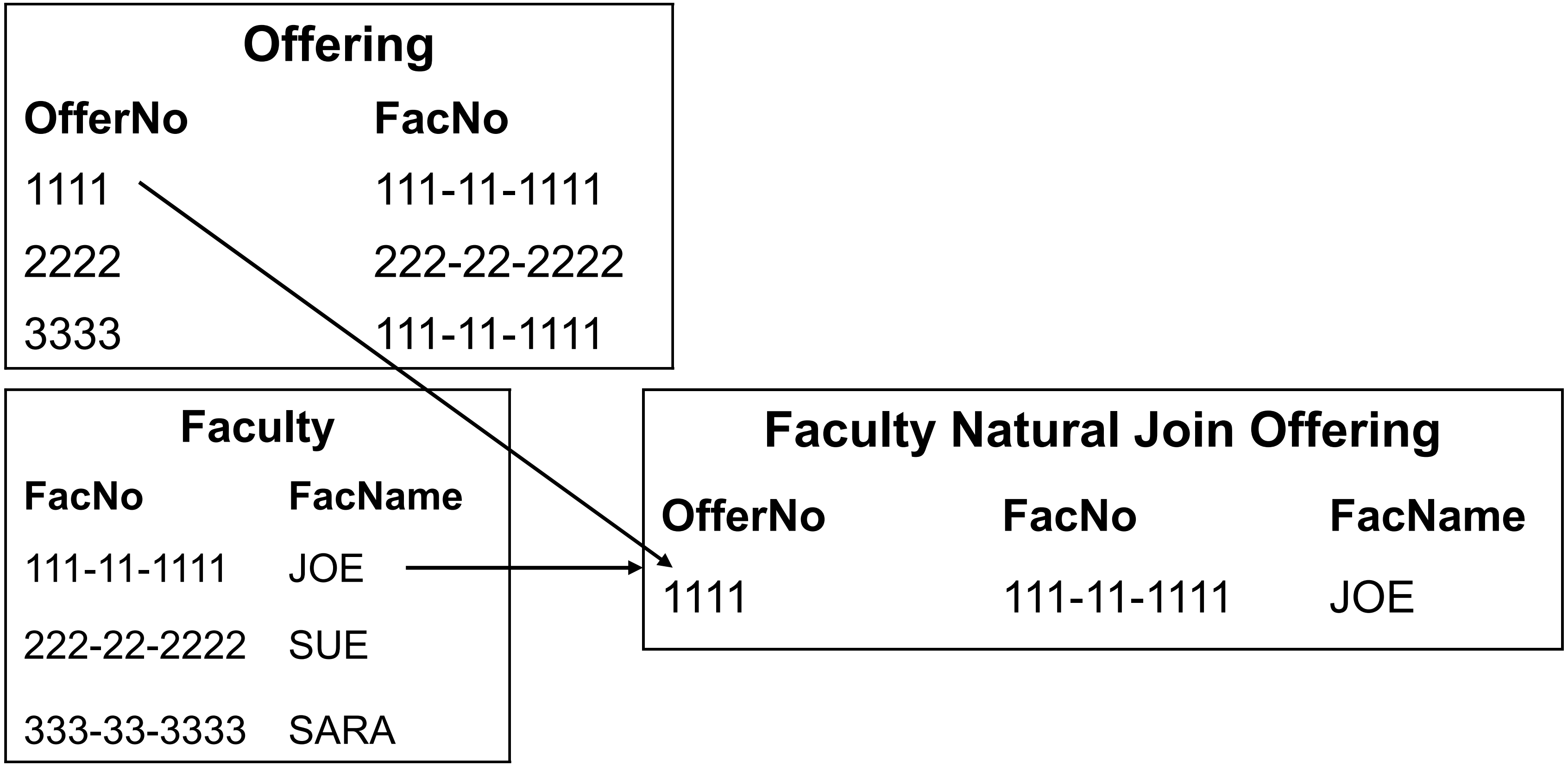
Offering

OfferNo	FacNo
1111	111-11-1111
2222	222-22-2222
3333	111-11-1111

Faculty

FacNo	FacName
111-11-1111	JOE
222-22-2222	SUE
333-33-3333	SARA

Natural Join Example II



Offering

OfferNo	FacNo
1111	111-11-1111
2222	222-22-2222
3333	111-11-1111

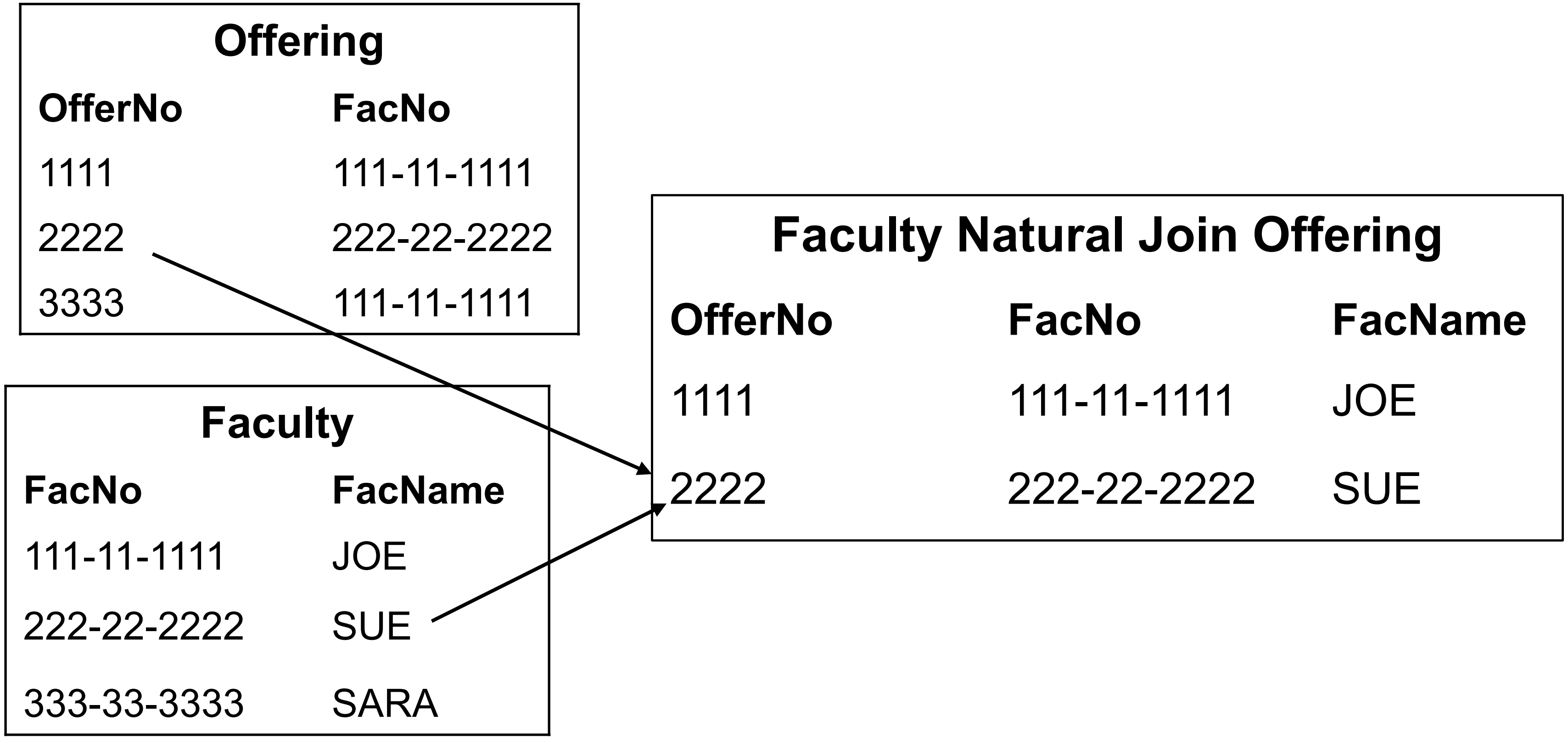
Faculty

FacNo	FacName
111-11-1111	JOE
222-22-2222	SUE
333-33-3333	SARA

Faculty Natural Join Offering

OfferNo	FacNo	FacName
1111	111-11-1111	JOE

Natural Join Example III



Offering

OfferNo	**FacNo**
1111	111-11-1111
2222	222-22-2222
3333	111-11-1111

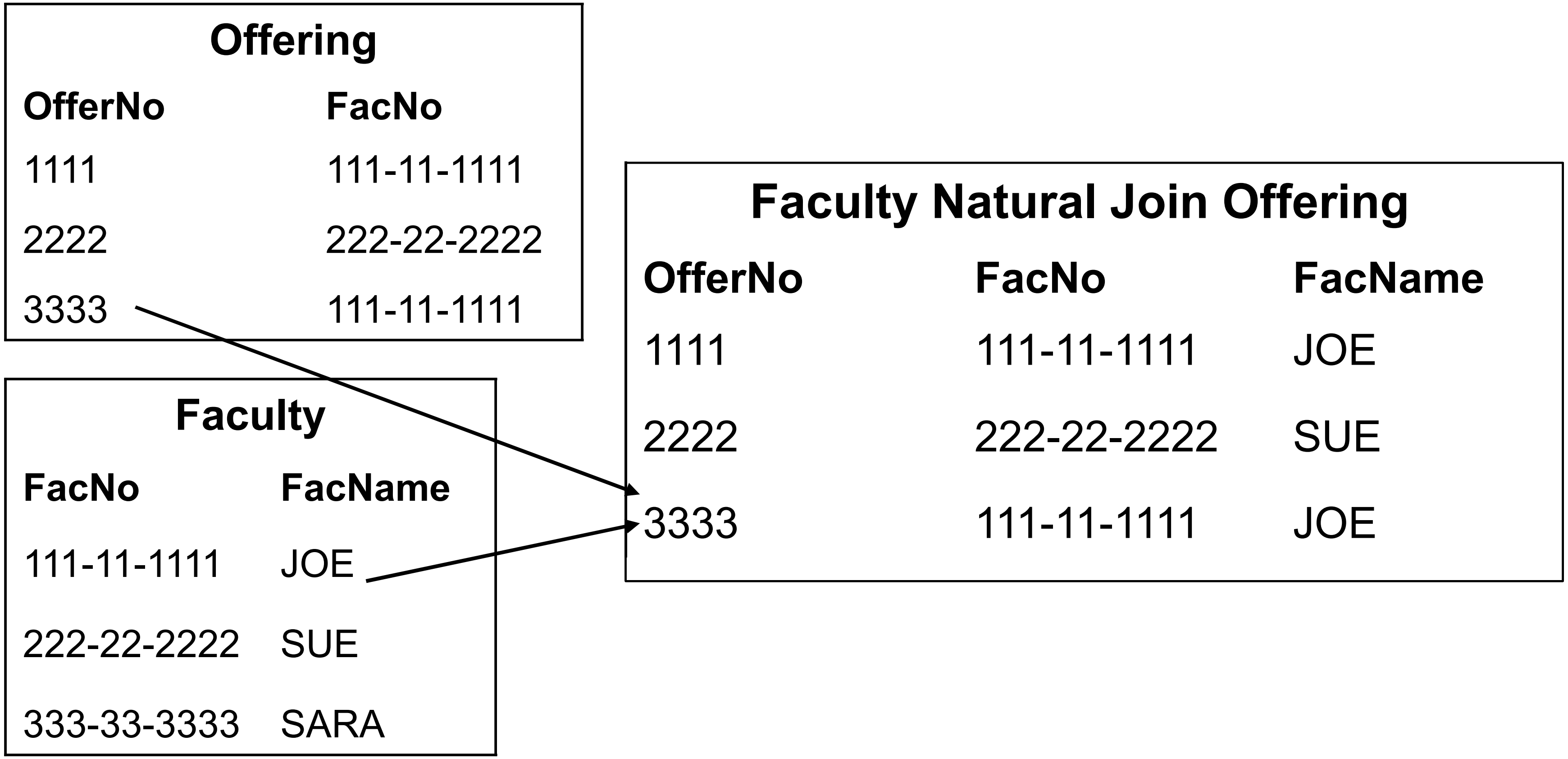
Faculty

FacNo	**FacName**
111-11-1111	JOE
222-22-2222	SUE
333-33-3333	SARA

Faculty Natural Join Offering

OfferNo	**FacNo**	**FacName**
1111	111-11-1111	JOE
2222	222-22-2222	SUE

Natural Join Example IV



Offering

OfferNo	**FacNo**
1111	111-11-1111
2222	222-22-2222
3333	111-11-1111

Faculty

FacNo	**FacName**
111-11-1111	JOE
222-22-2222	SUE
333-33-3333	SARA

Faculty Natural Join Offering

OfferNo	**FacNo**	**FacName**
1111	111-11-1111	JOE
2222	222-22-2222	SUE
3333	111-11-1111	JOE

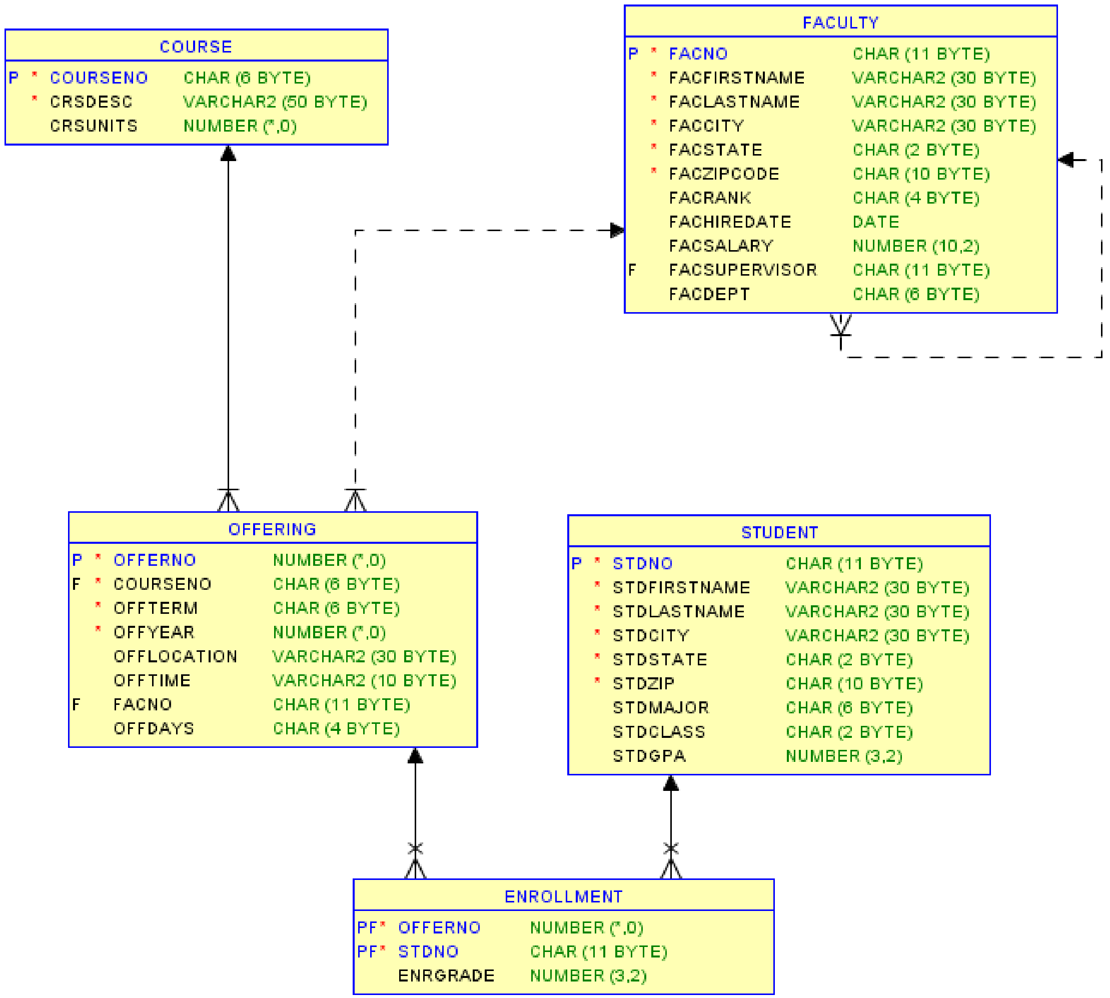
Join Operator

- 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

Natural Join Operator

- Most common join operator
- Requirements
  - Equality matching condition
  - Matching columns with the same unqualified names
  - Remove one join column in the result
- Usually performed on PK-FK join columns

University Database Diagram



COURSE		
P *	COURSENO	CHAR (6 BYTE)
*	CRSDESC	VARCHAR2 (50 BYTE)
	CRSUNITS	NUMBER (*,0)

FACULTY		
P *	FACNO	CHAR (11 BYTE)
*	FACFIRSTNAME	VARCHAR2 (30 BYTE)
*	FACLASTNAME	VARCHAR2 (30 BYTE)
*	FACCITY	VARCHAR2 (30 BYTE)
*	FACSTATE	CHAR (2 BYTE)
*	FACZIPCODE	CHAR (10 BYTE)
	FACRANK	CHAR (4 BYTE)
	FACHIREDATE	DATE
	FACSALARY	NUMBER (10,2)
F	FACSUPERVISOR	CHAR (11 BYTE)
	FACDEPT	CHAR (6 BYTE)

OFFERING		
P *	OFFERNO	NUMBER (*,0)
F *	COURSENO	CHAR (6 BYTE)
*	OFFTERM	CHAR (6 BYTE)
*	OFFYEAR	NUMBER (*,0)
	OFFLOCATION	VARCHAR2 (30 BYTE)
	OFFTIME	VARCHAR2 (10 BYTE)
F	FACNO	CHAR (11 BYTE)
	OFFDAYS	CHAR (4 BYTE)

STUDENT		
P *	STDNO	CHAR (11 BYTE)
*	STDFIRSTNAME	VARCHAR2 (30 BYTE)
*	STDLASTNAME	VARCHAR2 (30 BYTE)
*	STDCITY	VARCHAR2 (30 BYTE)
*	STDSTATE	CHAR (2 BYTE)
*	STDZIP	CHAR (10 BYTE)
	STDMAJOR	CHAR (6 BYTE)
	STDCLASS	CHAR (2 BYTE)
	STDGPA	NUMBER (3,2)

ENROLLMENT		
PF*	OFFERNO	NUMBER (*,0)
PF*	STDNO	CHAR (11 BYTE)
	ENRGRADE	NUMBER (3,2)

Summary

- Essential operator in query formulation
- Use sample tables to learn the join operator
- Explicit join specification in the SELECT statement

Business School
UNIVERSITY OF COLORADO DENVER

Information Systems Program

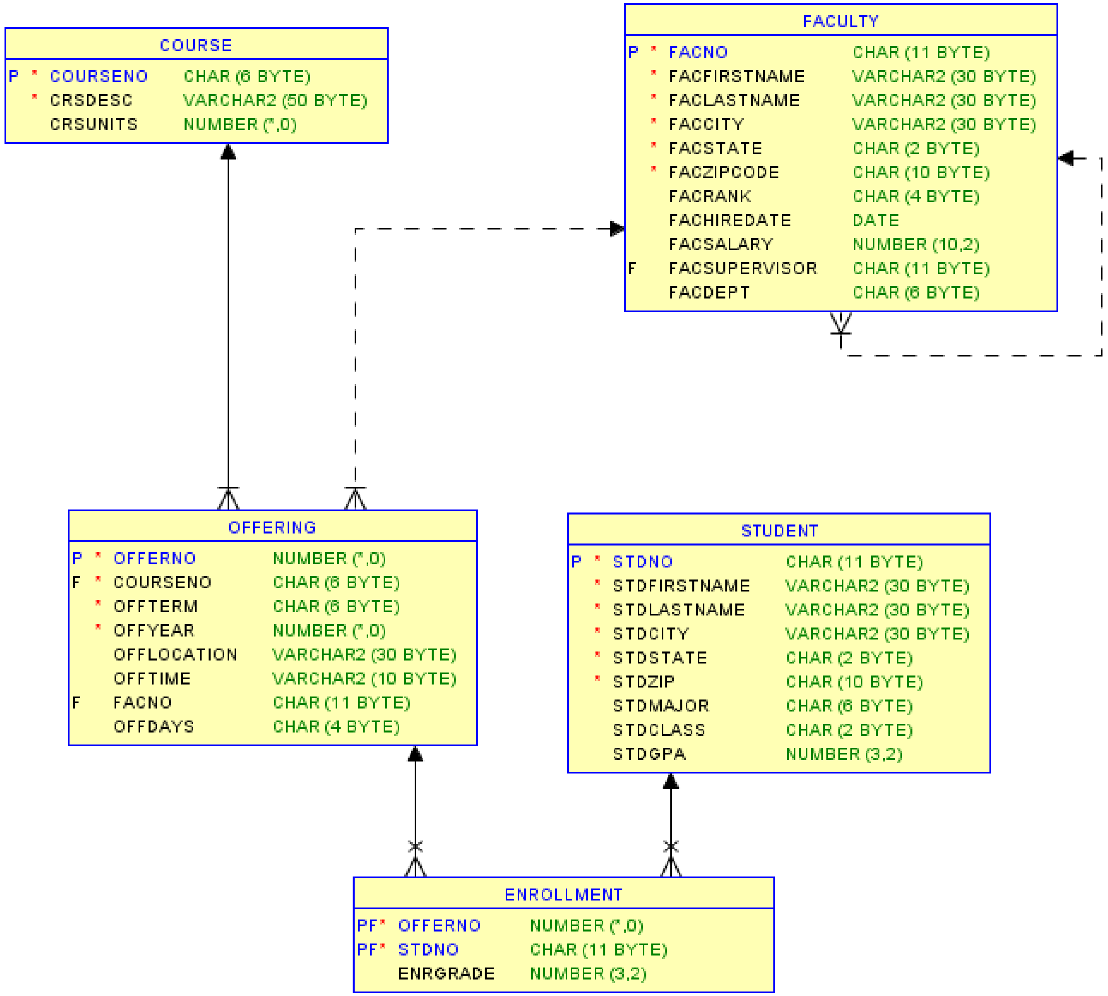
Module 4
Basic Query Formulation with SQL

Lesson 4: Using join operations in SQL SELECT statements

Lesson Objectives

- Write SELECT statements using both join styles
- Provide natural language explanations of SELECT statements using both join styles

University Database Diagram



COURSE

P	*	COURSENO	CHAR (6 BYTE)
	*	CRSDESC	VARCHAR2 (50 BYTE)
		CRSUNITS	NUMBER (*,0)

FACULTY

P	*	FACNO	CHAR (11 BYTE)
	*	FACFIRSTNAME	VARCHAR2 (30 BYTE)
	*	FACLASTNAME	VARCHAR2 (30 BYTE)
	*	FACCITY	VARCHAR2 (30 BYTE)
	*	FACSTATE	CHAR (2 BYTE)
	*	FACZIPCODE	CHAR (10 BYTE)
		FACRANK	CHAR (4 BYTE)
		FACHIREDATE	DATE
		FACSALARY	NUMBER (10,2)
F		FACSUPERVISOR	CHAR (11 BYTE)
		FACDEPT	CHAR (6 BYTE)

OFFERING

P	*	OFFERNO	NUMBER (*,0)
F	*	COURSENO	CHAR (6 BYTE)
	*	OFFTERM	CHAR (6 BYTE)
	*	OFFYEAR	NUMBER (*,0)
		OFFLOCATION	VARCHAR2 (30 BYTE)
		OFFTIME	VARCHAR2 (10 BYTE)
F		FACNO	CHAR (11 BYTE)
		OFFDAYS	CHAR (4 BYTE)

STUDENT

P	*	STDNO	CHAR (11 BYTE)
	*	STDFIRSTNAME	VARCHAR2 (30 BYTE)
	*	STDLASTNAME	VARCHAR2 (30 BYTE)
	*	STDCITY	VARCHAR2 (30 BYTE)
	*	STDSTATE	CHAR (2 BYTE)
	*	STDZIP	CHAR (10 BYTE)
		STDMAJOR	CHAR (6 BYTE)
		STDCLASS	CHAR (2 BYTE)
		STDGPA	NUMBER (3,2)

ENROLLMENT

PF	*	OFFERNO	NUMBER (*,0)
PF	*	STDNO	CHAR (11 BYTE)
		ENRGRADE	NUMBER (3,2)

Cross Product Style

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

Example 1

```
SELECT OfferNo, CourseNo, FacFirstName, FacLastName
 FROM Offering, Faculty
 WHERE OffTerm = 'FALL' AND OffYear = 2016
   AND FacRank = 'ASST' AND CourseNo LIKE 'IS%'
   AND Faculty.FacNo = Offering.FacNo;
```

Join Operator Style

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

Example 2

```
SELECT OfferNo, CourseNo, FacFirstName, FacLastName
 FROM Offering INNER JOIN Faculty
   ON Faculty.FacNo = Offering.FacNo
 WHERE OffTerm = 'FALL' AND OffYear = 2016
   AND FacRank = 'ASST' AND CourseNo LIKE 'IS%';
```

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

Name Qualification Example

Example 2

```
SELECT OfferNo, CourseNo, FacFirstName, FacLastName
 FROM Offering INNER JOIN Faculty
   ON Faculty.FacNo = Offering.FacNo
 WHERE OffTerm = 'FALL' AND OffYear = 2016
   AND FacRank = 'ASST' AND CourseNo LIKE 'IS%';
```

- FacNo must be qualified
- Can qualify other names for easier readability

Cross Product Style with 3 Tables

Example 3: List Leonard Vince's teaching schedule in fall 2016. 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.FacNo = Offering.FacNo
   AND Offering.CourseNo = Course.CourseNo
   AND OffYear = 2016 AND OffTerm = 'FALL'
   AND FacFirstName = 'LEONARD'
   AND FacLastName = 'VINCE';
```

Join Operator Style with 3 Tables

Example 4: List Leonard Vince’s teaching schedule in fall 2016. 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 Offering INNER JOIN Course
   ON Offering.CourseNo = Course.CourseNo
   INNER JOIN Faculty ON Offering.FacNo = Faculty.FacNo
 WHERE OffYear = 2016 AND OffTerm = 'FALL'
   AND FacFirstName = 'LEONARD'
   AND FacLastName = 'VINCE';
```

Summary

- Explicit join specification in the SELECT statement
- Read and write both join styles
- Consult database diagram when formulating join queries
- Work many problems

Business School
UNIVERSITY OF COLORADO DENVER

Information Systems Program

Module 4
Basic Query Formulation with SQL

Lesson 5: GROUP BY Clause

Lesson Objectives

- Write SELECT statements containing the GROUP BY clause
- Write SELECT statements with WHERE and HAVING clauses
- Write natural language explanations about SELECT statements containing the GROUP BY clause

Row Summaries

- Important for decision-making tasks
- Row summary details
  - Result contains statistical (aggregate) functions
  - Conditions involve statistical functions
- SQL keywords
  - Aggregate functions in the result list such as AVG and SUM
  - GROUP BY: summary columns
  - HAVING: summary conditions

Individual Rows versus Row Summaries

Example 1: Sort faculty by rank

```
SELECT FacNo, FacRank, FacSalary
FROM Faculty
ORDER BY FacRank;
```

```
FACNO       FACRANK  FACSALARY
----------- ------- ----------
654-32-1098 ASSC         70000
987-65-4321 ASSC         75000
876-54-3210 ASST         40000
098-76-5432 ASST         35000
765-43-2109 PROF         65000
543-21-0987 PROF        120000
```

Example 2: Compute average salary for each faculty rank

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

```
FACRANK  AVGSALARY
------- ----------
ASSC         72500
ASST         37500
PROF         92500
```

Filtering Rows and Groups

Example 3: List average GPA by major for upper class students

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

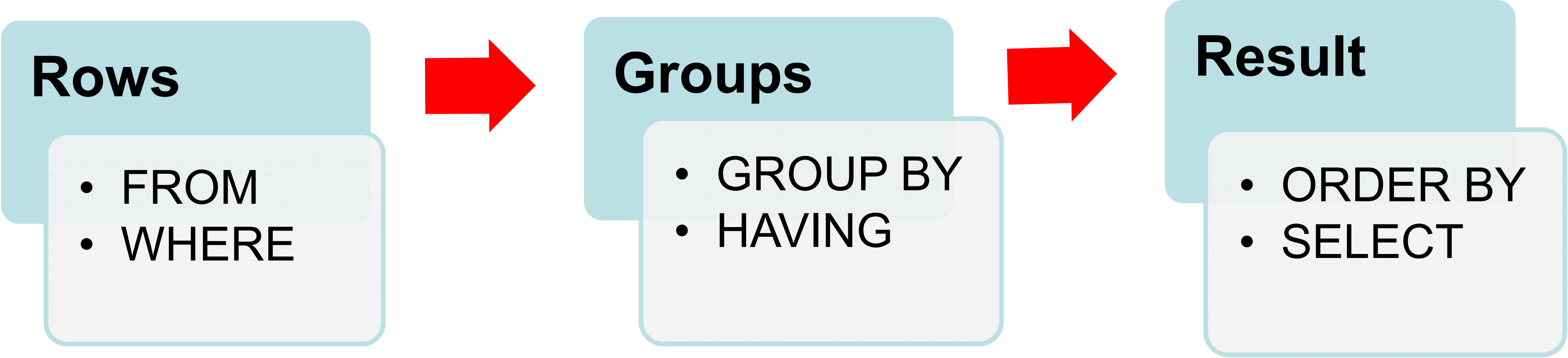
```
STDMAJOR       AVGGPA
--------- ----------
ACCT              3.5
FIN               2.8
IS               3.15
```

Example 4: List average GPA by major for upper class students in which average GPA is greater than 3.1.

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

```
STDMAJOR       AVGGPA
--------- ----------
ACCT              3.5
IS               3.15
```

Query Clause Evaluation Order



Rows

- FROM
- WHERE

Groups

- GROUP BY
- HAVING

Result

- ORDER BY
- SELECT

Evaluation Order 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

Summary

- Summarization queries common for business intelligence
- GROUP BY clause to calculate summary data for decision making
- Grouping after joins and row conditions
- Extensions to GROUP BY operator business intelligence queries