SQL

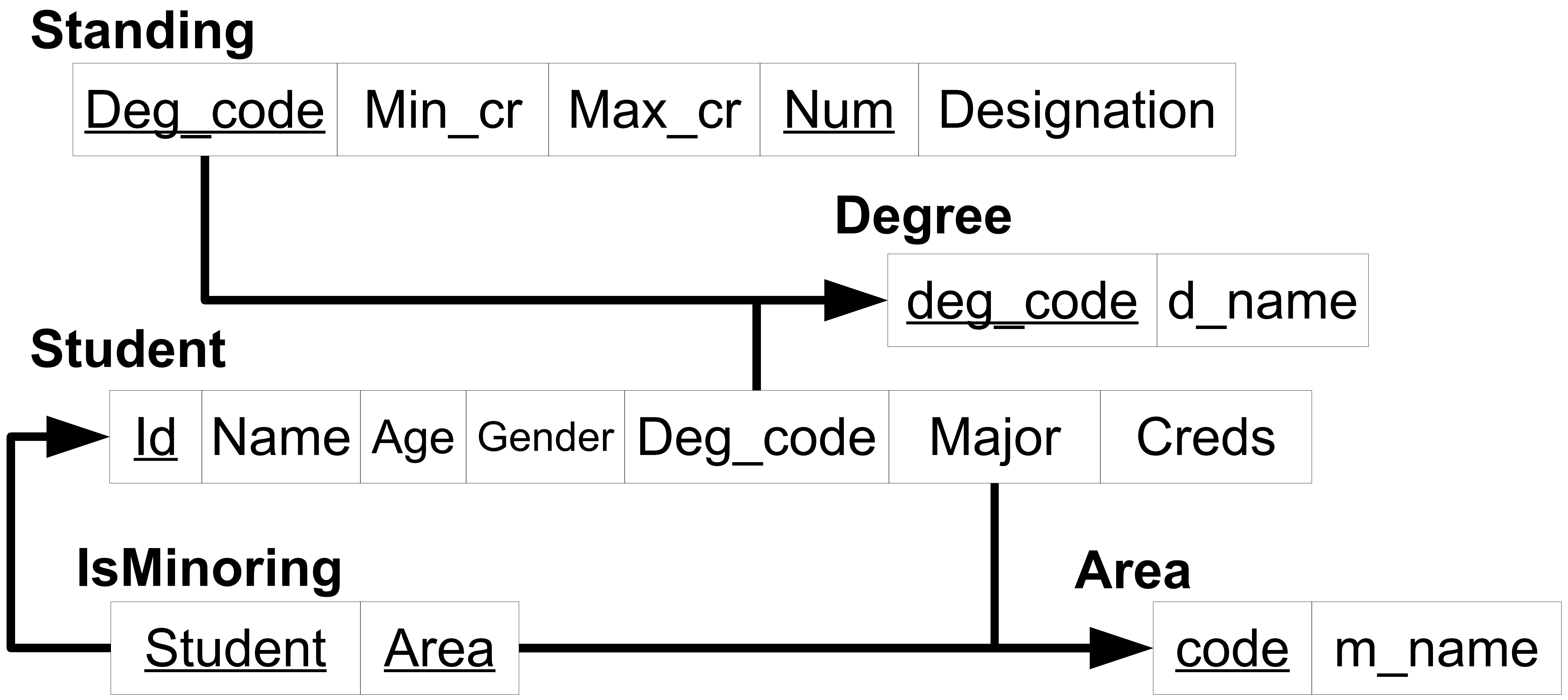
Simple JOIN statements

Orlando Karam
okaram@spsu.edu

JOINS

- Many times we need to combine information from several tables.
- In SQL we can do this through:
  - –Using several tables in FROM (implicit joins)
  - –Explicit JOIN clauses
  - –Subqueries
- A join conceptually consists of:
  - –Generating all possible combinations of rows (cartesian product)
  - –Selecting only the ‘matching’ combinations (join predicate)

Sample Schema



This is the schema of our joins worksheet

Implicit JOINS

- Just use two (or more) tables in the FROM clause, separated by comma
  - And then put the join predicate on the WHERE clause
  - Simple, but easy to forget the join predicate, especially with multiple tables or complex conditions
- Example - get student name with degree name
  - SELECT name, d_name
  - FROM Student , Degree
  - WHERE Standing.deg_code=Degree.Deg_code

You try

- Get student with MAJOR's name

- What about undeclared majors ?
  - OUTER joins, covered later

Disambiguating field names

- Different tables may have fields with the same name (related or not)
- We can disambiguate by using table.field
- We need to disambiguate when we refer to that field in a query that uses both tables
- It may be easier to alias each table, use alias.field
- Example - get student name with degree name
  - –SELECT s.name, d.d_name
  - –FROM Student S, Degree D
  - –WHERE S.deg_code=D.Deg_code

Explicit JOIN clause

- use Table1 JOIN Table2 in FROM clause
  - Join predicate required by syntax (ON, USING ...)
  - JOIN clause acts syntactically as a table name
- Example - get student name with degree name
  - SELECT name, d_name
  - FROM Student S JOIN Degree D ON s.deg_code=D.Deg_code

You try

- Get student with MAJOR's name

- What about undeclared majors ?
  - –OUTER joins, covered later

You try

- Give students' id,name with their standing's designation
  - –Obtained through standing table, depending on the student's credits and degree

Try it with implicit joins

- Give students’ id,name with their standing’s designation

- Which one is easier ? more error prone ?

NATURAL JOIN, USING

- Most joins use only = on one or more fields (equi-joins)
- Sometimes tables have the same names for corresponding fields
- NATURAL JOIN matches on fields with same name having same value
  - Brittle ! what happens if schema changes ?
- USING allows you to list field names, matches on those fields having same value

Examples

- NATURAL - get student name with degree name
  - SELECT name, d_name
  - FROM Student S NATURAL JOIN Degree D
- USING - get student name with degree name
  - SELECT name, d_name
  - FROM Student S JOIN Degree D USING (Deg_code)

Joining 3 tables

- A JOIN B now acts, syntactically as *one* table, so so A JOIN B ON ... JOIN C ON ...
  - –expression associates left-to-right (for outer)
- Example: Print Student's name with name of their *minor(s)*
  - –SELECT S.Name,A.Name
  - –FROM Student S JOIN IsMinoring M ON (S.Id=M.Student)
    - –JOIN Area A ON (A.Code=M.Area)
- For implicit joins, just list all in FROM with commas, use AND to combine the conditions