

IT420: Database Management and Organization

SQL - SELECT Chapter 2

1

Class Exercise

- Division(Name, Building, OfficeNb)
- Department(DeptName, ChairName, WebAddress, DivName)
- Create tables
- Modify Department to add a FK constraint for DivName
- Create table Colleges with same structure as Division
- Insert everything from Division into Colleges
- Remove Division table
- Find the name of the Chair of the 'Math' Department

2

Last Time vs. Today

- Last Time:
 - CREATE, DROP, ALTER
 - INSERT, DELETE, UPDATE
 - SELECT
- Today: More about SELECT
 - Joins, sub-queries
 - Sorting, wild cards
 - Arithmetic operations, aggregates
 - Groups

3

The SQL SELECT Statement

- Basic SQL Query:


```
SELECT      [DISTINCT] column_name(s) | *
FROM        table_name(s)
[WHERE      conditions]
```

4

SELECT from Two or More Tables

Find the names of students enrolled in IT420

```
SELECT SName
FROM Students S, Enrolled E
WHERE S.SnB = E.SnB AND E.Cid = 'IT420'
```

Students

SNb	SName	Email
190	Smith	jsmith@usna.edu
673	Doe	jdoe@usna.edu
312	Doe	jdoe2@usna.edu

Courses

Cid	CName	CDept
IT420	Database	ComSci
IT340	Networks	ComSci
SM121	Calculus1	Math

Enrolled

SNb	Cid	Semester
190	IT340	Spring2006
312	IT420	Fall2005

5

SELECT - Conceptual Evaluation Strategy

- Semantics of an SQL query defined in terms of the following conceptual evaluation strategy:
 - Compute the cross-product of *table_names*
 - Discard resulting rows if they fail condition
 - Delete columns that are not in *column_names*
 - If DISTINCT is specified, eliminate duplicate rows
- This strategy is probably the least efficient way to compute a query!
 - An optimizer will find more efficient strategies to compute the same answers.

6

Example Conceptual Evaluation

```
SELECT SName
FROM Students S, Enrolled E
WHERE S.SNb = E.SNb AND E.Cid = 'IT420'
```

S.SNb	SName	Email	E.SNb	Cid	Semester
190	Smith	jsmith@usna.edu	190	IT340	Spring2006
190	Smith	jsmith@usna.edu	312	IT420	Fall2005
673	Doe	jdoe@usna.edu	190	IT340	Spring2006
673	Doe	jdoe@usna.edu	312	IT420	Fall2005
312	Doe	jdoe2@usna.edu	190	IT340	Spring2006
312	Doe	jdoe2@usna.edu	312	IT420	Fall2005

7

Example Conceptual Evaluation

```
SELECT SName
FROM Students S, Enrolled E
WHERE S.SNb = E.SNb AND E.Cid = 'IT420'
```

S.SNb	SName	Email	E.SNb	Cid	Semester
190	Smith	jsmith@usna.edu	190	IT340	Spring2006
190	Smith	jsmith@usna.edu	312	IT420	Fall2005
673	Doe	jdoe@usna.edu	190	IT340	Spring2006
673	Doe	jdoe@usna.edu	312	IT420	Fall2005
312	Doe	jdoe2@usna.edu	190	IT340	Spring2006
312	Doe	jdoe2@usna.edu	312	IT420	Fall2005

8

Example Conceptual Evaluation

```
SELECT SName
FROM Students S, Enrolled E
WHERE S.SNb = E.SNb AND E.Cid = 'IT420'
```

SName
Doe

S.SNb	SName	Email	E.SNb	Cid	Semester
190	Smith	jsmith@usna.edu	190	IT340	Spring2006
190	Smith	jsmith@usna.edu	312	IT420	Fall2005
673	Doe	jdoe@usna.edu	190	IT340	Spring2006
673	Doe	jdoe@usna.edu	312	IT420	Fall2005
312	Doe	jdoe2@usna.edu	190	IT340	Spring2006
312	Doe	jdoe2@usna.edu	312	IT420	Fall2005

9

Modified Query

```
SELECT SNb
FROM Students S, Enrolled E
WHERE S.SNb = E.SNb AND E.Cid = 'IT420'
```

- Would the result be different with DISTINCT?

10

Class Exercise

- Students(SNb, SName, Email)
- Courses(Cid, CName, Dept)
- Enrolled(SNb, Cid, Semester)
- Find the student number and name for each student enrolled in 'Spring2007' semester
- Find the names of all students enrolled in 'ComSci' courses

11

Sorting the Results

```
SELECT [DISTINCT] column_name(s) | *
FROM table_name(s)
[WHERE conditions]
[ORDER BY column_name(s) [ASC/DESC]]
```

Example:
Students(SNb, SName, Email, Major)

```
SELECT SNb, SName
FROM Students
ORDER BY SName ASC, SNb DESC
```

12

WHERE Clause Options

- AND, OR
- IN, NOT IN, BETWEEN
- LIKE

```
SELECT SNb, SName
FROM Students
WHERE SNb LIKE '8%' AND
Major IN ('SIT', 'SCS')
```

Wild cards:

- SQL-92 Standard (SQL Server, Oracle, etc.):
 - _ = Exactly one character
 - % = Any set of one or more characters
- MS Access
 - ? = Exactly one character
 - * = Any set of one or more characters

Example:

Students(SNb, SName, Email, Major)

Find alpha and name of SCS or SIT students with SNb starting with '8'

13

Calculations in SQL

- Simple arithmetic
- Five SQL Built-in Functions:
 - COUNT
 - SUM
 - AVG
 - MIN
 - MAX

14

Simple Arithmetic

- SELECT NbHours*
HourlyRate AS
Charge
FROM FlightEvents

Charge
150
400
50
400

- SELECT SFirstName
+ ' ' + SLastName
FROM Students

(No column name)
John Doe
Brad Johnson
Jessica Smith
Mary Davis

15

Aggregate Operators

- SELECT COUNT(*)
FROM Students
- SELECT COUNT(DISTINCT SName)
FROM Students
WHERE SNb > 700
- SELECT AVG(Age)
FROM Students
WHERE SNb LIKE '08_____'

16

Aggregate Operators Limitations

- Return only one row
- Not in WHERE clause

17

Select oldest students and their age

- ~~SELECT S.SName, MAX (Age)
FROM Students S~~ Illegal!
- SELECT S.SName, S.Age
FROM Students S
WHERE S.AGE = (SELECT MAX(Age)
FROM Students)

Sub-query

18

Select students with age higher than average

- ~~SELECT *
 FROM Students
 WHERE Age > AVG(Age)~~

- SELECT *
 FROM Students
 WHERE Age > (SELECT AVG(Age)
 FROM Students)
 

19

Class Exercise

- Students(SNb, SName, Email)
- Courses(Cid,CName, Dept)
- Enrolled(SNb,Cid, Semester)
- List SNb of all students enrolled in 'IT420' or 'IT340', ordered by SNb

20

Grouping rows

- Find the age of the youngest student for each class year
- SELECT MIN (S.Age)
 FROM Students S
 WHERE S.ClassYear = 2007

(no column name)
21

21

GROUP-BY Clause

- SELECT [DISTINCT] column_name(s) |
 aggregate_expr
 FROM table_name(s)
 [WHERE conditions]
 GROUP BY grouping_columns

- Example:
 SELECT ClassYear, MIN(Age)
 FROM Students
 GROUP BY ClassYear

ClassYear	(no column name)
2007	21
2010	17
2009	18
2008	20

22

Conceptual Evaluation

- Semantics of an SQL query defined as follows:
 - Compute the cross-product of tables in FROM (*table_names*)
 - Discard resulting rows if they fail WHERE *conditions*
 - Delete columns that are not in SELECT or GROUP BY (*column_names* or *grouping-columns*)
 - Remaining rows are partitioned into groups by the value of the columns in *grouping-columns*
 - One answer row is generated per group
- Note: Does not imply query will actually be evaluated this way!

23

HAVING Clause

- SELECT [DISTINCT] column_name(s) |
 aggregate_expr
 FROM table_name(s)
 [WHERE conditions]
 GROUP BY grouping_columns
 HAVING group_conditions
- GROUP BY groups the rows
- HAVING restricts the groups presented in the result

24

Example- HAVING

- SELECT ClassYear, MIN(Age)
FROM Students
WHERE MajDeptName = 'ComSci'
GROUP BY ClassYear
HAVING COUNT(*) > 20

25

Conceptual Evaluation

- SQL query semantics:
 - Compute the cross-product of *table_names*
 - Discard resulting rows if they fail *conditions*
 - Delete columns that are not specified in SELECT, GROUP BY
 - Remaining rows are partitioned into groups by the value of the columns in *grouping-columns*
 - One answer row is generated per group
 - Discard resulting groups that do not satisfy *group_conditions*

26

Example

- SELECT Class, MIN(Age)
FROM Students
WHERE MajDeptName = 'ComSci'
GROUP BY Class
HAVING COUNT(*) > 2

27

Class Exercise

- Students(SNb, SName, Email)
- Courses(Cid, CName, Dept)
- Enrolled(SNb, Cid, Semester)

- List all course names, and the number of students enrolled in the course

28

Subqueries

- SELECT *
FROM Students
WHERE Age > (SELECT AVG(Age)
FROM Students)
- Second select is a **subquery** (or **nested query**)
- You can have subqueries in FROM or HAVING clause also

30

Subqueries in FROM Clause

- Find name of students enrolled in both 'IT420' and 'IT334'
- SELECT FName + ' ' + LName AS StudentName
FROM Students, (SELECT Alpha
FROM Enroll
WHERE CourseID = 'IT420'
AND Alpha IN
(SELECT Alpha
FROM Enroll
WHERE CourseID = 'IT334'))
AS ResultAlphaTable
WHERE Students.Alpha = ResultAlphaTable.Alpha

31

Subqueries Exercise

- Students(Alpha, LName, FName, Class, Age)
 - Enroll(Alpha, CourseID, Semester, Grade)
1. Find alpha for students enrolled in **both** 'IT420' and 'IT334'
 2. Find name of students enrolled in **both** 'IT420' and 'IT334'

32

Class Exercise

- Students(Alpha, LName, FName, Class, Age)
 - Enroll(Alpha, CourseID, Semester, Grade)
- Find the name of students enrolled in 'IT420'
 - Usual way
 - Use subqueries

33

Class Exercise

- What does this query compute:
- SELECT FName, LName
FROM Students S, Enroll E1, Enroll E2
WHERE S.Alpha = E1.Alpha
AND S.Alpha = E2.Alpha
AND E1.CourseID = 'IT420'
AND E2.CourseID = 'IT344'

34

Summary

- SELECT [DISTINCT] column_name(s) |
aggregate_expr
FROM *table_name(s)*
WHERE *conditions*
GROUP BY *grouping_columns*
HAVING *group_conditions*
ORDER BY column_name(s) [ASC/DESC]

35