Tutorial 3 – Simple SQL

CSC343 - Introduction to Databases
Fall 2008

TA: Lei Jiang

Simple SQL

- **Question**
  - Given the relation schemas, write the following queries in SQL.

    Student (snum: integer, sname: string, major: string, level: string, age: integer)
    Class (name: string, meets_at: string, room: string, fid: integer)
    Enrolled (snum: integer, cname: string)
    Faculty (fid: integer, fname: string, deptid: integer)

  - Q1: Find the department id of the faculty member named *I. Teach*.
  - Q2: Find the names of all junior students (level='JR'), and list in the order of age.
  - Q3: Find the number of classes that have an enrolment greater than 0.
- **Key points**
  - *Understand the semantics*
    - Entities: Student, Class, Faculty; Relationships: Enrolment, Teaching (where is the schema for it?)
    - Meaning of attributes, keys, foreign keys, …
- **Answer**

Q1:
```
SELECT deptid
FROM Faculty
WHERE fname = 'I.Teach'
```

Q2:
```
SELECT S.sname
FROM Student S
WHERE S.level = 'JR'
ORDERED BY S.age
```

Q3:
```
SELECT COUNT(DISTINCT E.cname)
FROM Enrolled as E
```

Simple SQL: Join

- **Question**

  Student (snum: integer, sname: string, major: string, level: string, age: integer)
  Class (name: string, meets_at: string, room: string, fid: integer)
  Enrolled (snum: integer, cname: string)
  Faculty (fid: integer, fname: string, deptid: integer)

  – Q4: Find names and majors of students who have enrolled in at least one class.

  – Q5: Find the number of students who have enrolled in at least two classes.

- **Answer**

```
Q4:
SELECT S.sname, S.major
FROM Student S, Enrolled E
WHERE S.snum = E.snum
```

```
Q5:
SELECT COUNT(DISTINCT S.sname)
FROM Student S, Enrolled E1, Enrolled E2
WHERE E1.snum = E2.snum AND
          E1.cnum <> E2.cnum AND
          S.snum = E1.snum
```

Simple SQL: Join

- **Question**

  Student (snum: integer, sname: string, major: string, level: string, age: integer)
  Class (name: string, meets_at: string, room: string, fid: integer)
  Enrolled (snum: integer, cname: string)
  Faculty (fid: integer, fname: string, deptid: integer)

  – (E5.1.1) Q6: Find distinct names of all Juniors (*level* = *JR*) enrolled in a class taught by *I. Teach*.

- **The way of thinking**
  - Given: ***Student****.level = 'JR' and* ***Faculty****.fname = 'I.Teach'* ……………………………. (1)
  - Asked: distinct ***Student****.sname* values …………………………………………………….. (2)
  - Connection: *Student <~>* ***Enrolled*** *<~>* ***Course*** *<~> Faculty* ………………..... (3)

- **Answer**

```
Q6:
SELECT DISTINCT S.sname as Student_Name
FROM Student S, Class C, Enrolled E, Faculty F
WHERE S.snum = E.snum AND E.cname = C.name AND C.fid = F.fid
         AND F.fname = 'I.Teach' AND S.level = 'JR'
```

specify what is asked (2)

All relations involved (1, 2, 3)

Specify connections (3)

Specifying what is given (1)

Simple SQL: Join

- **Question**
  - Given following database instance, answer Q6.

Student

snum	sname	major	level	age
101	Helen	CS	JR	19
102	Charles	CS	SR	21
103	Andy	CS	GR	25
104	Bob	CS	SR	23
105	Zorba	CS	GR	31

Enrolled

snum	cname
101	CSC343
101	CSC443
101	ECE300
102	CSC343
102	ECE300
103	CSC343
103	CSC443
103	ECE300
103	ECE201
105	CSC343

Class

name	meets_at	room	fid
CSC343	W1	BA1180	201
CSC443	T2	BA1170	202
ECE300	M1	BA1180	203
ECE201	F12	BA1160	203
CSC165	R3	BA1170	202

Faculty

fid	fname	deptid
201	S. Jackson	301
202	M. Shanks	301
203	I. Teach	302

Q6:

```
SELECT DISTINCT S.sname as Student_Name
FROM Student S, Class C, Enrolled E, Faculty F
WHERE S.snum = E.snum AND E.cname = C.name
          AND C.fid = F.fid
          AND F.fname = ‘I.Teach’ AND S.level = ‘JR’
```

- **Answer**

Student_Name
Helen

Simple SQL: Set Operation

- **Question**
  - Given following database instance, answer Q7.

Student

snum	sname	major	level	age
101	Helen	CS	JR	19
102	Charles	CS	SR	21
103	Andy	CS	GR	25
104	Bob	CS	SR	23
105	Zorba	CS	GR	31

Enrolled

snum	cname
101	CSC343
101	CSC443
101	ECE300
102	CSC343
102	ECE201
103	CSC343
103	CSC443
103	ECE300
103	ECE201
105	CSC343

Class

name	meets_at	room	fid
CSC343	W1	BA1080	201
CSC443	T2	BA1170	202
ECE300	M1	BA1080	203
ECE201	F12	BA1060	203
CSC165	R3	BA1170	202

Faculty

fid	fname	deptid
201	S. Jackson	301
202	M. Shanks	301
203	I. Teach	302

Q7:

```
SELECT DISTINCT S.sname as Student_Name
FROM Student S, Enrolled E
WHERE S.snum = E.snum AND E.snum = ‘CSC343’
EXCEPT
SELECT DISTINCT S2.sname
FROM Student S2, Enrolled E2
WHERE S2.snum = E2.cnum AND E2.cnum = ‘CSC443’
```

- **Answer**

Student_Name
Charles
Zorba

Simple SQL: Set Operation

- **Question**

Student (snum: integer, sname: string, major: string, level: string, age: integer)
Class (name: string, meets_at: string, room: string, fid: integer)
Enrolled (snum: integer, cname: string)
Faculty (fid: integer, fname: string, deptid: integer)

  – Q7: Find the names of all students who have enrolled in both CSC343 and CSC443.

- **Answer**

```
Q7 Sol#1:
SELECT DISTINCT S.sname as Student_Name
FROM Student S, Enrolled E
WHERE S.snum = E.snum AND E.cnum = ‘CSC343’
INTERSECT
SELECT DISTINCT S2.sname as Student_Name
FROM Student S2, Enrolled E2
WHERE S2.snum = E2.snum AND E2.cnum = ‘CSC443’
```

```
Q7 Sol#2:
SELECT DISTINCT S.sname
FROM Student S, Enrolled E1, Enrolled E2
WHERE S.snum = E1.snum AND
        E1.snum = E2.snum AND
        E1.cnum = ‘CSC343’ AND
        E2.cnum = ‘CSC443’
```

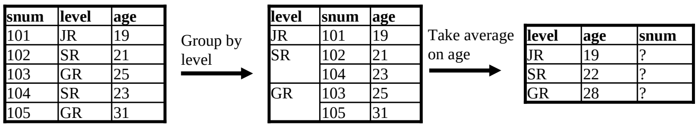
Simple SQL: Aggregation

- **Question**

Student (snum: integer, sname: string, major: string, level: string, age: integer)
Class (name: string, meets_at: string, room: string, fid: integer)
Enrolled (snum: integer, cname: string)
Faculty (fid: integer, fname: string, deptid: integer)

  – (E5.1.8) Q8: For all levels except JR, print the level and the average age of students for that level.

- **The way of thinking**



snum	level	age
101	JR	19
102	SR	21
103	GR	25
104	SR	23
105	GR	31

Group by level

level	snum	age
JR	101	19
SR	102	21
	104	23
GR	103	25
	105	31

Take average on age

level	age	snum
JR	19	?
SR	22	?
GR	28	?

- **Answer**

```
Q8:
SELECT S.level, AVG(S.age)
FROM Student S
WHERE S.level <> ‘JR’
GROUP BY S.level
```

1. An attribute *attr* is allowed in the select clause if
   (a) it appears in the group by clause or
   (b) It’s used with an aggregation function
2. One exception: if you group by the primary key of a relation
   (a) Every attribute of that relation can be used
   (b) Why?

Simple SQL: Outer Join

- **Question**

Student (snum: integer, sname: string, major: string, level: string, age: integer)
Class (name: string, meets_at: string, room: string, fid: integer)
Enrolled (snum: integer, cname: string)
Faculty (fid: integer, fname: string, deptid: integer)

  - Q9: Find the names of all students and the names of all classes they are enrolled in (if any)

- **Key points**
  - Some students may have not been enrolled in any course
  - we cannot exclude them from the list
  - use left outer join the handle this situation.

- **Answer**

Q9:

```
SELECT S.sname, E.cname
FROM Student S LEFT JOIN Enrolled E ON S.snum = E.snum
```