CPSC 304 – Administrative notes October 23 & October 24, 2024

- Project:
  - Milestone 3: Project check in – due October 25
    - Sign up now!
  - Milestone 4: Project implementation – due November 29
  - Milestone 5: Group demo – week of December 2
  - Milestone 6: Individual Assessment – Due November 29
- Tutorials: pivoting to project work/open office hours – this week is nominally SQL Plus
- Note that your SQL accounts and repositories may go away at the end of the term!

Now where were we...

- SQL!
- When in doubt, start with
  SELECT
  FROM
  WHERE

New Students Example

- Class(name,meets_at,room,fid)
- Student(snum,sname,major,standing,age)
- Enrolled(snum,cname)
- Faculty(fid,fname,deptid)

Class Table

Name	Meets_at	Room	FID
Data Structures	MWF 10	R128	489456522
Database Systems	MWF 12:30-1:45	1320 DCL	142519864
Operating System Design	TuTh 12-1:20	20 AVW	489456522
Archaeology of the Incas	MWF 3-4:15	R128	248965255
Aviation Accident Investigation	TuTh 1-2:50	Q3	011564812
Air Quality Engineering	TuTh 10:30-11:45	R15	011564812
Introductory Latin	MWF 3-4:15	R12	248965255
American Political Parties	TuTh 2-3:15	20 AVW	619023588
Social Cognition	Tu 6:30-8:40	R15	159542516
Perception	MTuWTh 3	Q3	489221823
Multivariate Analysis	TuTh 2-3:15	R15	090873519
Patent Law	F 1-2:50	R128	090873519
Urban Economics	MWF 11	20 AVW	489221823
Organic Chemistry	TuTh 12:30-1:45	R12	489221823
Marketing Research	MW 10-11:15	1320 DCL	489221823
Seminar in American Art	M 4	R15	489221823
Orbital Mechanics	MWF 8 1320	DCL	011564812
Dairy Herd Management	TuTh 12:30-1:45	R128	356187925
Communication Networks	MW 9:30-10:45	20 AVW	141582651
Optical Electronics	TuTh 12:30-1:45	R15	254099823
Introduction to Math	TuTh 8-9:30	R128	489221823

Student Table

SNUM	SNAME	MAJOR	ST	AGE
51135593	Maria White	English	SR	21
60839453	Charles Harris	Architecture	SR	22
99354543	Susan Martin	Law	JR	20
112348546	Joseph Thompson	Computer Science	SO	19
115987938	Christopher Garcia	Computer Science	JR	20
132977562	Angela Martinez	History	SR	20
269734834	Thomas Robinson	Psychology	SO	18
280158572	Margaret Clark	Animal Science	FR	18
301221823	Juan Rodriguez	Psychology	JR	20
318548912	Dorthy Lewis	Finance	FR	18
320874981	Daniel Lee	Electrical Engineering	FR	17
322654189	Lisa Walker	Computer Science	SO	17
348121549	Paul Hall	Computer Science	JR	18
351565322	Nancy Allen	Accounting	JR	19
451519864	Mark Young	Finance	FR	18
455798411	Luis Hernandez	Electrical Engineering	FR	17
462156489	Donald King	Mechanical Engineering	SO	19
550156548	George Wright	Education	SR	21
552455318	Ana Lopez	Computer Engineering	SR	19
556784565	Kenneth Hill	Civil Engineering	SR	21
567354612	Karen Scott	Computer Engineering	FR	18
573284895	Steven Green	Kinesiology	SO	19
574489456	Betty Adams	Economics	JR	20
578875478	Edward Baker	Veterinary Medicine	SR	21

Enrolled Table

SNUM	CNAME
112348546	Database Systems
115987938	Database Systems
348121549	Database Systems
322654189	Database Systems
552455318	Database Systems
455798411	Operating System Design
552455318	Operating System Design
567354612	Operating System Design
112348546	Operating System Design
115987938	Operating System Design
322654189	Operating System Design
567354612	Data Structures
552455318	Communication Networks
455798411	Optical Electronics
455798411	Organic Chemistry
301221823	Perception
301221823	Social Cognition
301221823	American Political Parties
556784565	Air Quality Engineering
99354543	Patent Law
574489456	Urban Economics

Faculty Table

FID	FNAME	DEPTID
142519864	I. Teach	20
242518965	James Smith	68
141582651	Mary Johnson	20
011564812	John Williams	68
254099823	Patricia Jones	68
356187925	Robert Brown	12
489456522	Linda Davis	20
287321212	Michael Miller	12
248965255	Barbara Wilson	12
159542516	William Moore	33
090873519	Elizabeth Taylor	11
486512566	David Anderson	20
619023588	Jennifer Thomas	11
489221823	Richard Jackson	33
548977562	Ulysses Teach	20

Running Examples

Movie(MovieID, Title, Year)
StarsIn(MovieID, StarID, Character)
MovieStar(StarID, Name, Gender)

Student(snum,sname,major,standing,age)
Class(name,meets_at,room,fid)
Enrolled(snum,cname)
Faculty(fid,fname,deptid)

What kinds of queries can you answer so far?

Find the student ids of those who have taken a course named “Database Systems”

```
SELECT  snum
FROM    Enrolled
WHERE   cname = 'Database Systems'
```

Do we need distinct? A. Yes. B. No

Student(snum,sname,major,standing,age)
Class(name,meets_at,room,fid)
Enrolled(snum,cname)
Faculty(fid,fname,deptid)

What kinds of queries can you answer so far?

Find the names of all classes taught by Elizabeth Taylor.

```
SELECT  name
FROM    Faculty f, class c
WHERE   f.fid = c.fid and f.fname = 'Elizabeth Taylor'
```

Do we need distinct? A. Yes. B. No

Student(snum,sname,major,standing,age)
Class(name,meets_at,room,fid)
Enrolled(snum,cname)
Faculty(fid,fname,deptid)

What kinds of queries can you answer so far?

Find the departments that have at least one faculty member.

Student(snum,sname,major,standing,age)
Class(name,meets_at,room,fid)
Enrolled(snum,cname)
Faculty(fid,fname,deptid)

What kinds of queries can you answer so far?

Find the departments that have at least one faculty member

```
SELECT DISTINCT deptid
FROM faculty
```

Clicker question: do we need DISTINCT?

a. Yes b. No

Student(snum,sname,major,standing,age)
Class(name,meets_at,room,fid)
Enrolled(snum,cname)
Faculty(fid,fname,deptid)

What kinds of queries can you answer so far?

Find the departments that have more than one faculty member (express not equal by "<>")

```
SELECT DISTINCT f1.deptid
FROM faculty f1, faculty f2
WHERE f1.fid <>f2.fid AND
f1.deptid = f2.deptid
```

That is why renaming is important

f1

fid	fname	Deptid
90873519	Elizabeth Taylor	11
619023588	Jennifer Thomas	11
…	…	…

f2

fid	fname	Deptid
90873519	Elizabeth Taylor	11
619023588	Jennifer Thomas	11
…	…	…

A good example for using the same table twice in a query

What kinds of queries can you answer so far?

Find the departments that have more than one faculty member (express not equal by "<>")

```
SELECT DISTINCT f1.deptid
FROM faculty f1, faculty f2
WHERE f1.fid <>f2.fid AND
f1.deptid = f2.deptid
```

That is why renaming is important

f1

fid	fname	Deptid
90873519	Elizabeth Taylor	11
619023588	Jennifer Thomas	11
…	…	…

f2

fid	fname	Deptid
90873519	Elizabeth Taylor	11
619023588	Jennifer Thomas	11
…	…	…

A good example for using the same table twice in a query

Do I need Distinct? A Yes B No

What kinds of queries can you answer so far?

Find the departments that have more than one faculty member (express not equal by "<>")

```
SELECT DISTINCT f1.deptid
FROM faculty f1, faculty f2
WHERE f1.fid <>f2.fid AND f1.deptid = f2.deptid
```

$$f1 \bowtie_{f1.fid \neq f2.fid \wedge f1.deptid = f2.deptid} f2$$

f1.fid	f1.fname	f1.deptid	f2.fid	f2.fname	f2.deptid
90873519	Elizabeth Taylor	11	619023588	Jennifer Thomas	11
619023588	Jennifer Thomas	11	90873519	Elizabeth Taylor	11
…	…	…	…	…	…

String comparisons

What are the student ids of those who have taken a course with “System” in the name?

A string walks into a bar…

```
SELECT DISTINCT snum
FROM   enrolled
WHERE cname LIKE '%System%'
```

- LIKE is used for string matching:
  - '_' stands for any one character and
  - '%' stands for 0 or more arbitrary characters.
- SQL supports string operations such as
  - concatenation (using "||")
  - converting from upper to lower case (and vice versa)
  - finding string length, extracting substrings, etc.

A string walks into a bar…

```
SELECT DISTINCT snum
FROM   enrolled
WHERE cname LIKE '%System%'
```

- LIKE is used for string matching:
  - '_' stands for any one character and
  - '%' stands for 0 or more arbitrary characters.
- SQL supports string operations such as
  - concatenation (using "||")
  - converting from upper to lower case (and vice versa)
  - finding string length, extracting substrings, etc.

Clicker question: do we need DISTINCT?

a. Yes b. No

In class exercise:

- SQL 1 – not for credit

Ordering of Tuples

- List in alphabetic order the names of actors who were in a movie in 1939

```
SELECT distinct Name
FROM   Movie, StarsIn, MovieStar
WHERE Movie.MovieID = StarsIn.MovieID and StarsIn.StarID
   = MovieStar.StarID and year = 1939
ORDER BY Name
```

Order is specified by:

- **desc** for descending order
- **asc** for ascending order (default)
- E.g. **order by** *Name* **desc**
- You can order within order: for example, … "ORDER BY Year, Name" would first order by Year, then Name within years

Clicker question: sorting

- Relation R has schema R(a,b,c). In the result of the query
  SELECT a, b, c
  FROM R
  ORDER BY c DESC, b ASC;
- What condition must a tuple *t* satisfy so that *t* **necessarily precedes** the tuple (5,5,5)? Identify one such tuple from the list below.

A. (3,6,3)
B. (1,5,5)
C. (5,5,6)
D. All of the above
E. None of the above

clickerorder.sql and clickerorder2.sql produce different ordering for 7,5,5 vs. 1,5,5

Clicker question: sorting

- Relation R has schema R(a,b,c). In the result of the query
  SELECT a, b, c
  FROM R
  ORDER BY c DESC, b ASC;
- What condition must a tuple *t* satisfy so that *t* **necessarily precedes** the tuple (5,5,5)? Identify one such tuple from the list below.

A. (3,6,3)
B. (1,5,5)
C. (5,5,6)
D. All of the above
E. None of the above

3 < 5

Not specified

Right

Set Operations

- **union, intersect,** and **except** correspond to the relational algebra operations $\cup$, $\cap$, $-$.
- Each automatically eliminates duplicates;
  To retain all duplicates use the corresponding multiset versions:

  union all, intersect all and **except all.**
- Suppose a tuple occurs *m* times in *r* and *n* times in *s,* then, it occurs:
  - *m* + *n* times in *r* **union all** *s*
  - min(*m,n)* times in *r* **intersect all** *s*
  - max(0, *m* – *n)* times in *r* **except all** *s*

Find IDs of MovieStars who've been in a movie in 1944 *or* 1974

Find IDs of MovieStars who've been in a movie in 1944 *or* 1974

- **UNION:** Can union any two *union-compatible* sets of tuples (i.e., the result of SQL queries).

```
SELECT  StarID
FROM  Movie M, StarsIn S
WHERE  M.MovieID=S.MovieID AND
( year = 1944 OR year = 1974)
```

- The two queries though quite similar return different results, why?
  - Use UNION ALL to get the same answer

```
SELECT  StarID
FROM     Movie M, StarsIn S
WHERE   M.MovieID = S.MovieID AND
year = 1944
UNION
SELECT  StarID
FROM     Movie M, StarsIn S
WHERE   M.MovieID = S.MovieID AND
year = 1974
```

Set Operations: Intersect

Example: Find IDs of stars who have been in a movie in 1944 *and* 1974.

- **INTERSECT:** Can be used to compute the intersection of any two *union-compatible* sets of tuples.
- In SQL/92, but some systems don't support it.

Set Operations: Intersect

Example: Find IDs of stars who have been in a movie in 1944 *and* 1974.

- **INTERSECT:** Can be used to compute the intersection of any two *union-compatible* sets of tuples.
- In SQL/92, but some systems don't support it.

```
SELECT  StarID
FROM    Movie M, StarsIn S
WHERE   M.MovieID = S.MovieID AND
year = 1944
INTERSECT
SELECT  StarID
FROM    Movie M, StarsIn S
WHERE   M.MovieID = S.MovieID AND
year = 1974
```

Oracle does
MYSQL doesn't

Rewriting INTERSECT with Joins

- Example: Find IDs of stars who have been in a movie in 1944 *and* 1974 without using **INTERSECT.**

Rewriting INTERSECT with Joins

- Example: Find IDs of stars who have been in a movie in 1944 *and* 1974 without using **INTERSECT.**

```
SELECT  distinct S1.StarID
FROM    Movie M1, StarsIn S1,
        Movie M2, StarsIn S2
WHERE
        M1.MovieID = S1.MovieID AND M1.year = 1944 AND
        M2.MovieID = S2.MovieID AND M2.year = 1974 AND
        S2.StarID = S1.StarID
```

Set Operations: EXCEPT/MINUS

- Find the sids of all students who took Operating System Design but did not take Database Systems

Student(snum,sname,major,standing,age)
Class(name,meets_at,room,fid)
Enrolled(snum,cname)
Faculty(fid,fname,deptid)

Set Operations: EXCEPT/MINUS

- Find the sids of all students who took Operating System Design but did not take Database Systems

```
SELECT snum
FROM enrolled
WHERE cname = 'Operating System Design'
EXCEPT  ← Oracle uses MINUS rather than EXCEPT
SELECT snum
FROM enrolled
WHERE cname = 'Database Systems'
```

Can we do it in a different way?
(We'll come back to this)

But what about…

- Select the IDs of all students who have not taken “Operating System Design”

Student(snum,sname,major,standing,age)
Class(name,meets_at,room,fid)
Enrolled(snum,cname)
Faculty(fid,fname,deptid)

But what about…

- Select the IDs of all students who have not taken "Operating System Design"
  - One way to do is to find all students that taken "Operating System Design".
  - Do all students MINUS those who have taken "Operating System Design"

```
SELECT snum
FROM student
EXCEPT  ← Oracle uses MINUS rather than EXCEPT
SELECT snum
FROM enrolled
WHERE cname = 'Operating System Design'
```

In class exercise

- SQL 2 – for credit
- Can’t do parts 3 and 4 yet, so due after *next* class