CS 327E Lecture 11

Shirley Cohen

March 2, 2016

Agenda

- Announcements
- Readings for today
- Reading Quiz
- Concept Questions
- Homework for next time

Announcements

- Midterm 2 will be next Wednesday
- There will be a short review on Monday

Homework for Today

- Chapter 7 from the Beginning Database Design book
- Exercises at the end of Chapter 7

Quiz Question 1

What is one point emphasized by Churcher in Chapter 7 of *Beginning Database Design*?

A. The development of a good abstract model allows us to translate it into SQL tables easily
B. The design of SQL tables should accurately reflect the essential requirements of the real-world problem
C. Inheritance can easily and precisely be represented using SQL tables
D. None of the above

Quiz Question 2

How is a `many-to-many` relationship represented in SQL?

A. Add foreign keys in each of the respective tables
B. Add an additional row to the table
C. Add a “junction" table with two foreign keys
D. None of the above

Quiz Question 3

How is a `one-to-many` relationship represented in SQL?

A. Add a foreign key to the many-side of the relationship
B. Add a foreign key to the one-side of the relationship
C. Add a new table with two foreign keys
D. None of the above

Quiz Question 4

How is a `one-to-one` relationship represented in SQL?

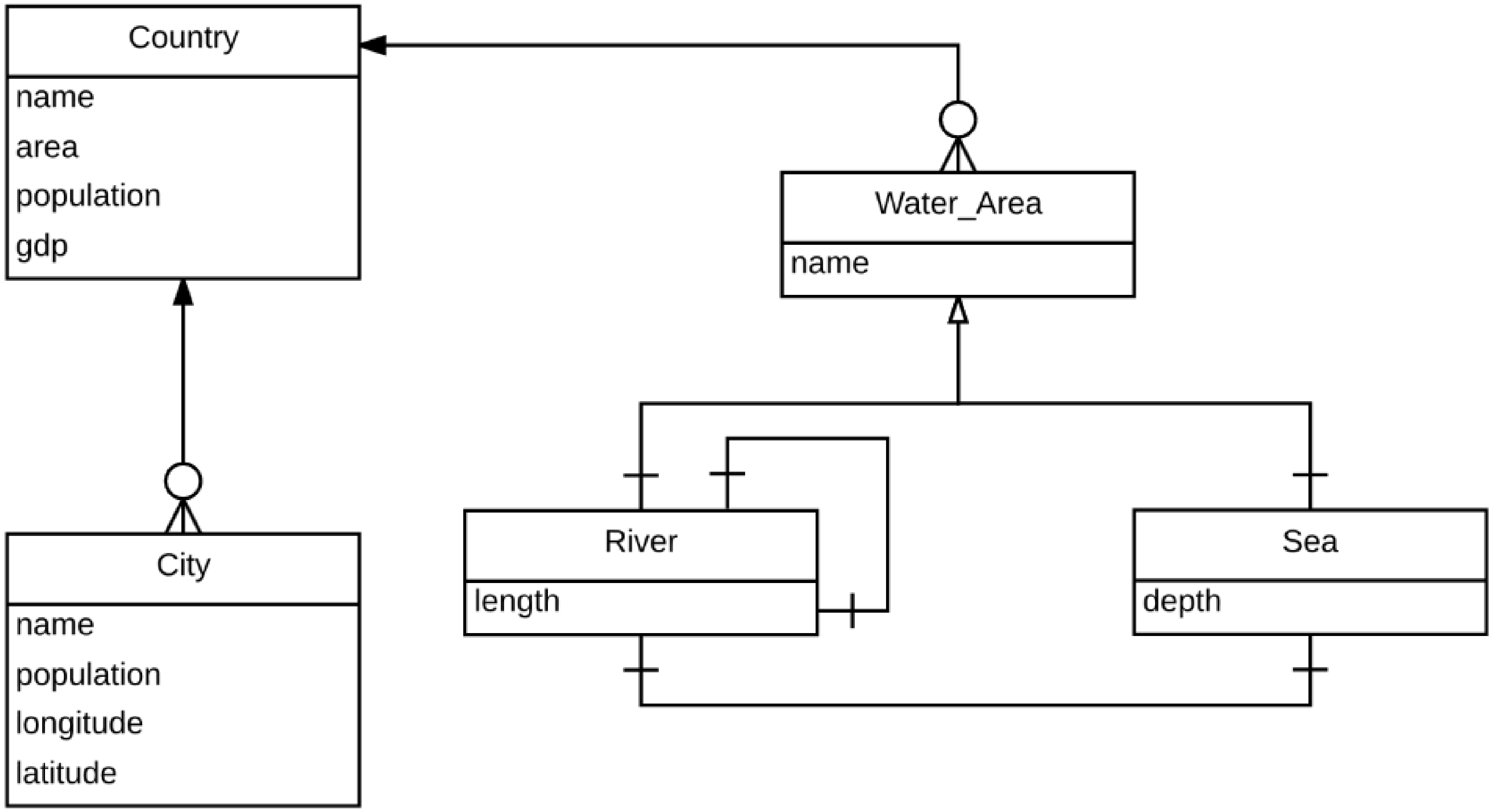
A. Add a foreign key in either direction
B. Add an additional table with a foreign key that represents the parent table
C. Add an additional row to the table
D. Add a new table with two foreign keys

Quiz Question 5

How should phone numbers be stored in a table?

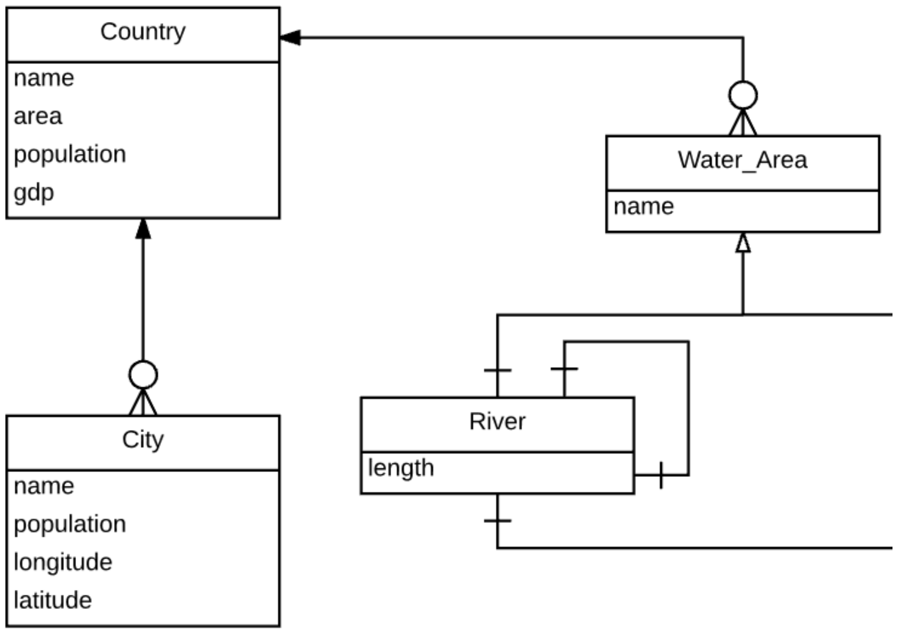
A. Using a `clob` type
B. Using a `float` type
C. Using a `varchar` or `char` type
D. Using a `date` type

Recall Geography Diagram

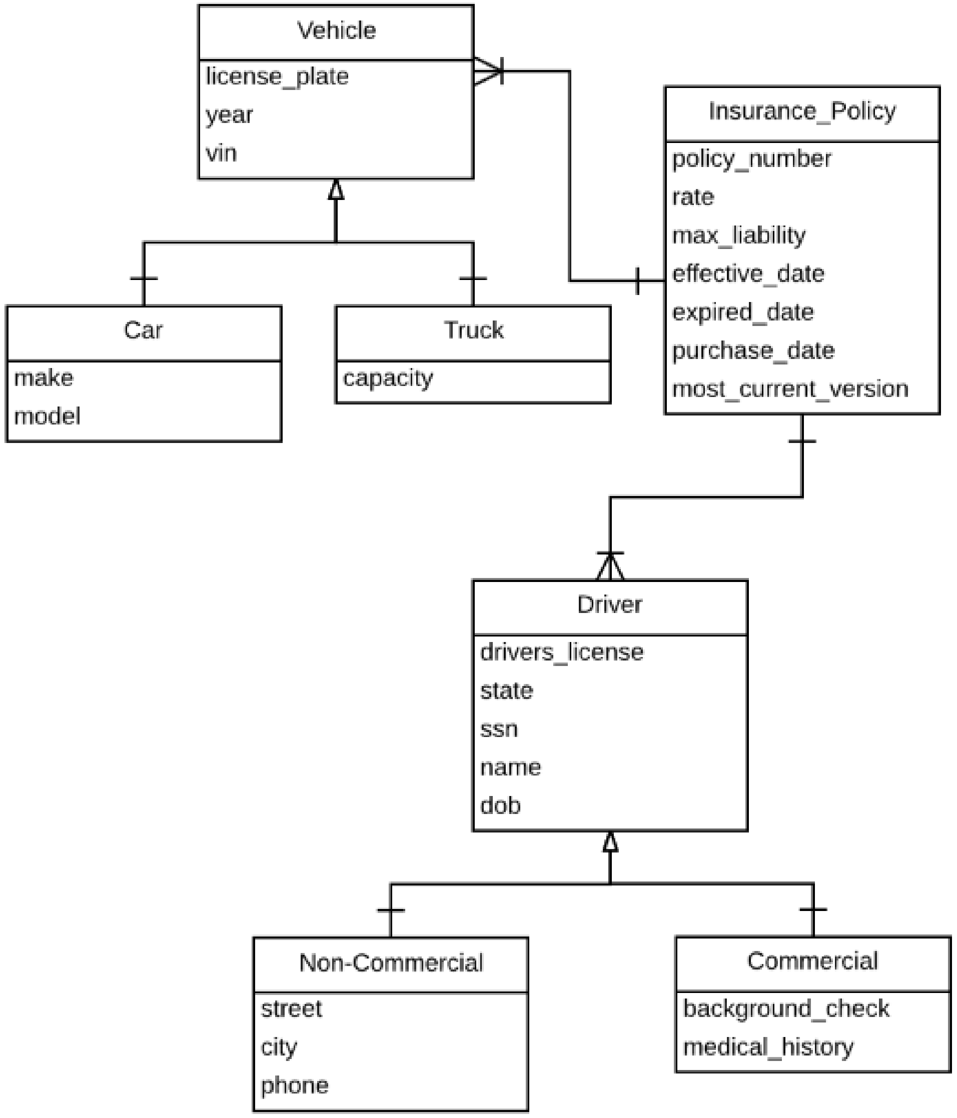


Converting Geography to Relations

```
CREATE TABLE Country
(
  country_code INT PRIMARY KEY,
  name VARCHAR(30) NOT NULL,
  area INT,
  population INT,
  gdp INT
)

CREATE TABLE Water_Area
(
   water_id INT PRIMARY KEY,
   name VARCHAR(50) NOT NULL
)

CREATE TABLE Country_Water_Area
(
   country_code INT,
   water_area_id INT,
   PRIMARY KEY (country_code, water_area_id),
   FOREIGN KEY (country_code) REFERENCES Country(country_code),
   FOREIGN KEY (water_area_id) REFERENCES Water_Area(water_id)
)
```



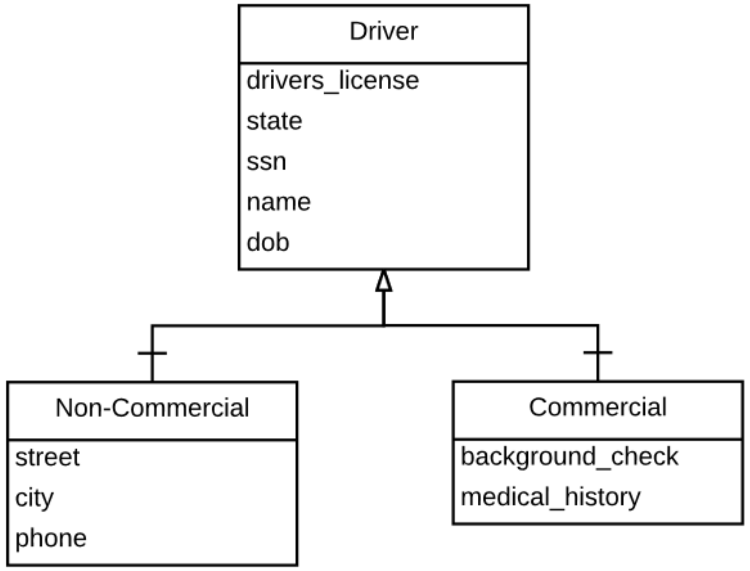
Recall Car Insurance Diagram



Concept Question 1

What can go wrong with this design?

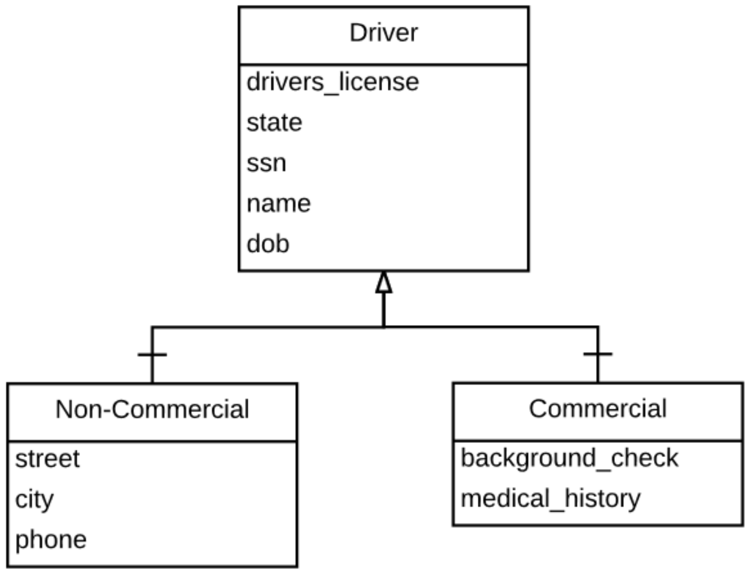
```
CREATE TABLE Driver (
  ssn INT,
  name VARCHAR(50) NOT NULL,
  dob DATE NOT NULL,
  drivers_license CHAR(8) NOT NULL,
  state CHAR(2) NOT NULL,
  driver_type CHAR(1)
  CHECK driver_type IN ('N', 'C'),
  PRIMARY KEY (ssn, driver_type))

CREATE TABLE NonCommercial (
   ssn INT PRIMARY KEY,
   street VARCHAR(50) NOT NULL,
   city VARCHAR(50) NOT NULL,
   phone VARCHAR(15) NOT NULL,
   FOREIGN KEY (ssn) REFERENCES Driver(ssn))

CREATE TABLE Commercial (
   ssn INT PRIMARY KEY,
   background_check VARCHAR(50),
   medical_history CLOB
   FOREIGN KEY (ssn) REFERENCES Driver(ssn))
```



A. The foreign keys pointing to `ssn`
B. The composite primary key (`ssn, driver_type`)
C. The primary key on `ssn`
D. All of the above

Converting Car Insurance to Relations

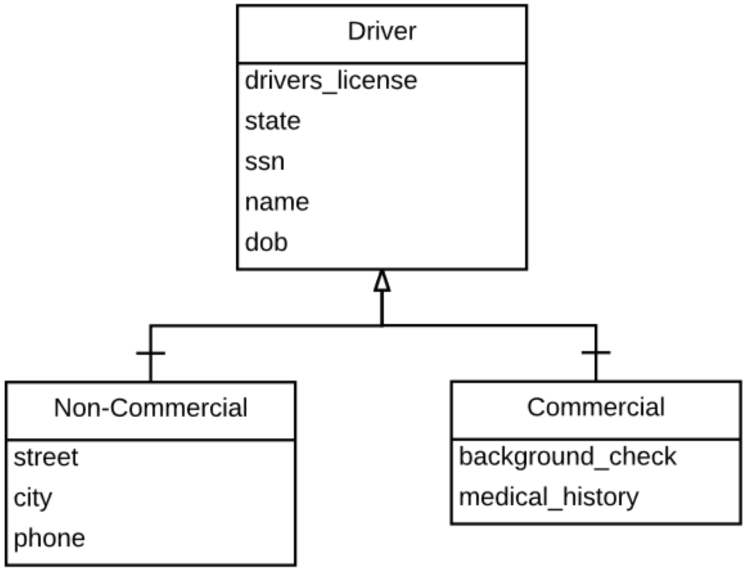
```
CREATE TABLE Driver (
  ssn INT PRIMARY KEY,
  name VARCHAR(50) NOT NULL,
  dob DATE NOT NULL,
  drivers_license CHAR(8) NOT NULL,
  state CHAR(2) NOT NULL,
  driver_type CHAR(1)
  CHECK driver_type IN ('N', 'C', 'B'))

CREATE TABLE NonCommercial (
   ssn INT PRIMARY KEY,
   street VARCHAR(50) NOT NULL,
   city VARCHAR(50) NOT NULL,
   phone VARCHAR(15) NOT NULL,
   FOREIGN KEY (ssn) REFERENCES Driver(ssn))

CREATE TABLE Commercial (
   ssn INT PRIMARY KEY,
   background_check VARCHAR(50),
   medical_history CLOB
   FOREIGN KEY (ssn) REFERENCES Driver(ssn))
```



Concept Question 2

How can we support *n* number of overlapping driver types?

```
CREATE TABLE Driver (
  ssn INT PRIMARY KEY,
  name VARCHAR(50) NOT NULL,
  dob DATE NOT NULL,
  drivers_license CHAR(8) NOT NULL,
  state CHAR(2) NOT NULL)

CREATE TABLE NonCommercial (
   ssn INT PRIMARY KEY,
   street VARCHAR(50) NOT NULL,
   city VARCHAR(50) NOT NULL,
   phone VARCHAR(15) NOT NULL,
   FOREIGN KEY (ssn) REFERENCES Driver(ssn))

CREATE TABLE Commercial (
   ssn INT PRIMARY KEY,
   background_check VARCHAR(50),
   medical_history CLOB
   FOREIGN KEY (ssn) REFERENCES Driver(ssn))
```

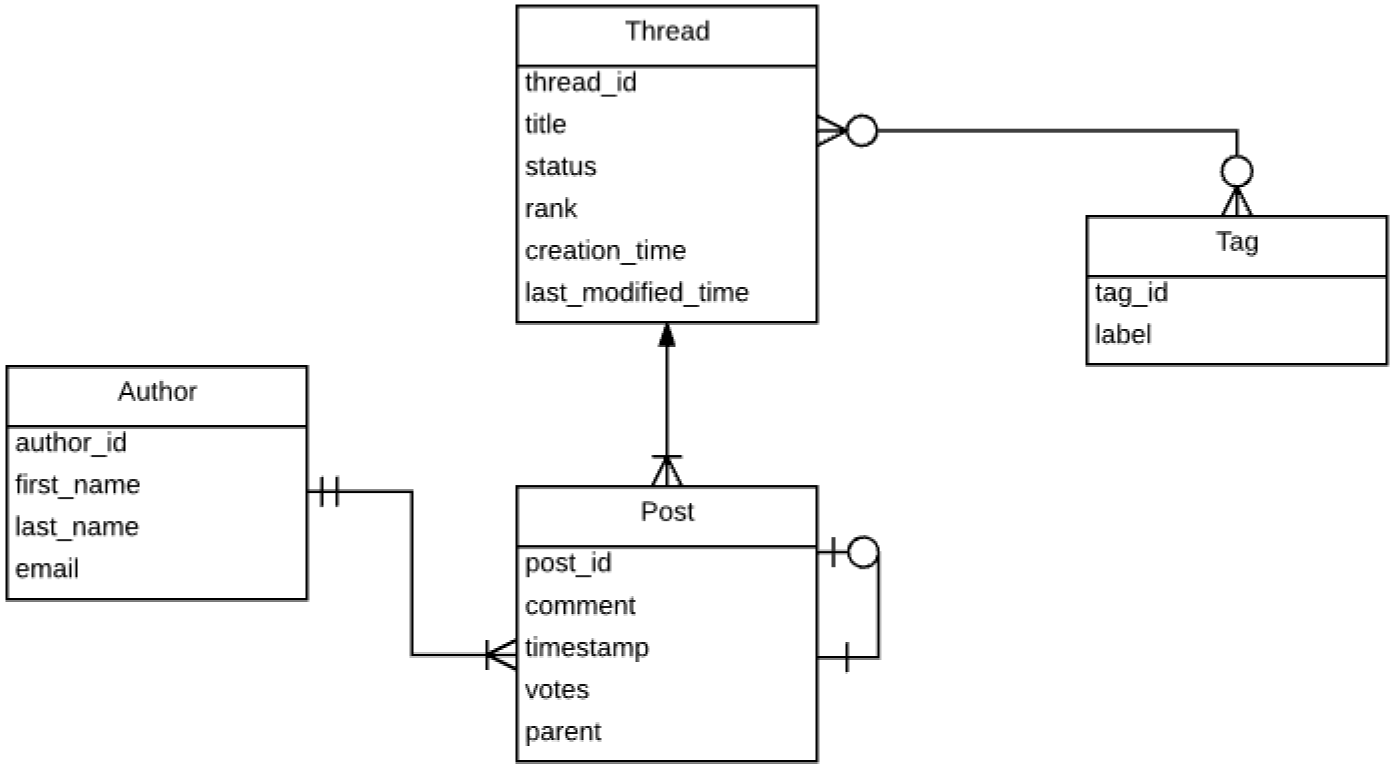


A. Create a `DriverType` table = (`ssn, type`)

B. Create a `DriverType` table = (`type`)

C. Create a `DriverType` table = (`ssn`)

Recall Discussion Forum Diagram



Converting Discussion Forum to Relations

```
CREATE TABLE Thread (
 thread_id INT PRIMARY KEY,
 title VARCHAR(30) NOT NULL,
 status CHAR(1) NOT NULL,
 rank DOUBLE,
 creation_time DATETIME,
 last_modified_time DATETIME)
```

post_id	comment	author	parent
1	Team outing anyone?	Andrew	NULL
2	Count me in! When? Where?	Sunil	1
3	Great idea!	Jen	1
4	I vote for SXSW	Jen	2
5	No, too crowded	Sunil	4
6	I'm open, whenever	Phil	2
7	How about Parkside?	Andrew	5

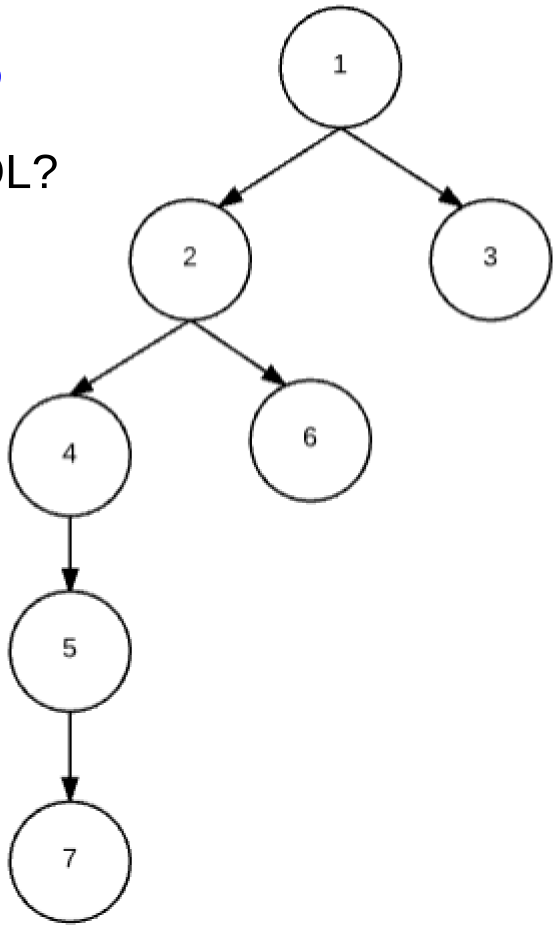
Note: The sample dataset uses the author's first name (instead of the `author_id`) for readability

```
CREATE TABLE Post (
  post_id INT PRIMARY KEY,
  author_id INT NOT NULL,
  comment VARCHAR(5000) NOT NULL,
  timestamp DATETIME NOT NULL,
  votes INT,
  thread_id INT NOT NULL,
  parent INT,
  FOREIGN KEY (parent) REFERENCES Post(post_id),
  FOREIGN KEY (author_id) REFERENCES Author(author_id)
  FOREIGN KEY (thread_id) REFERENCES Thread(thread_id))
```

Concept Question 3

How can we find the chain of replies to `post_id` = 1 in SQL?

post_id	comment	author	parent
1	Team outing anyone?	Andrew	NULL
2	Count me in! When? Where?	Sunil	1
3	Great idea!	Jen	1
4	I vote for SXSW	Jen	2
5	No, too crowded	Sunil	4
6	I'm open, whenever	Phil	2
7	How about Parkside?	Andrew	5


$$\begin{tikzcd} & & 1 \arrow[dl] \arrow[dr] & \\ & 2 \arrow[dl] \arrow[d] & & 3 \\ 4 \arrow[d] & 6 & & \\ 5 \arrow[d] & & & \\ 7 & & & \end{tikzcd}$$

For these answer choices, assume that the `select` clause contains all the fields we want to retrieve and the `where` clause filters by `post_id = 1`

A. 1 Left Outer Self Join on `Post`

B. 2 Left Outer Self Joins on `Post`

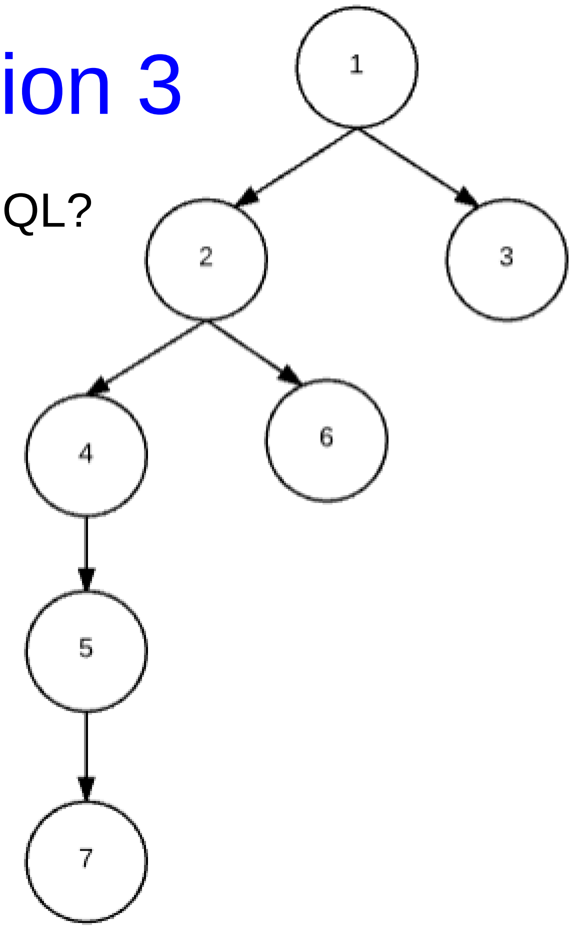
C. 3 Left Outer Self Joins on `Post`

D. None of the above

Solution to Concept Question 3

How can we find the chain of replies to `post_id` = 1 in SQL?

post_id	comment	author	parent
1	Team outing anyone?	Andrew	NULL
2	Count me in! When? Where?	Sunil	1
3	Great idea!	Jen	1
4	I vote for SXSW	Jen	2
5	No, too crowded	Sunil	4
6	I'm open, whenever	Phil	2
7	How about Parkside?	Andrew	5

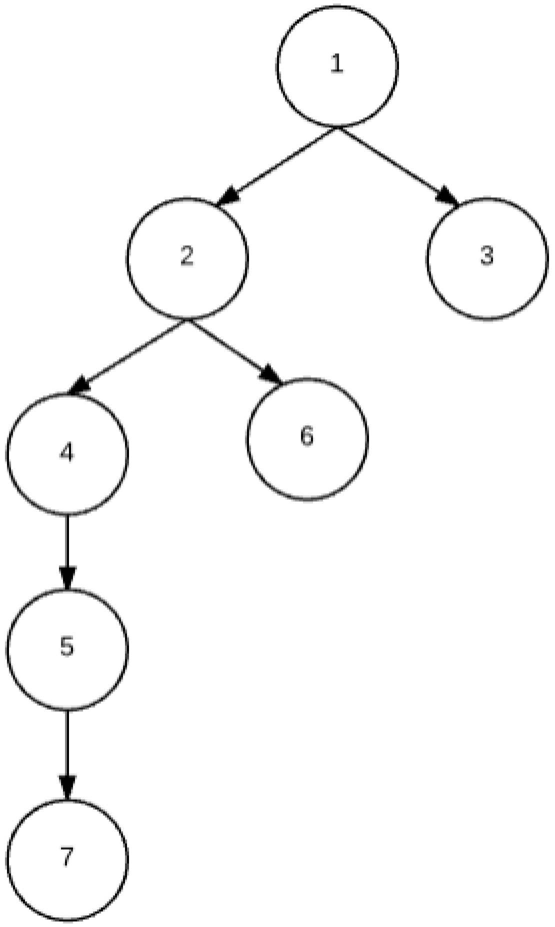


```
SELECT *
FROM Post p1
   LEFT OUTER JOIN Post p2 ON p1.post_id = p2.parent
   LEFT OUTER JOIN Post p3 ON p2.post_id = p3.parent
   LEFT OUTER JOIN Post p4 ON p3.post_id = p4.parent
   LEFT OUTER JOIN Post p5 ON p4.post_id = p5.parent
WHERE p1.post_id = 1
```

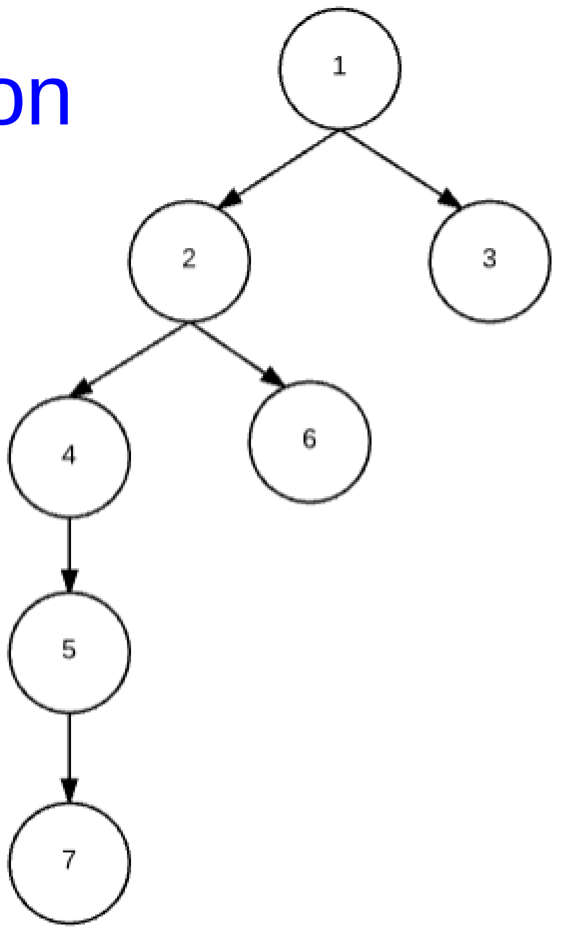
Path Enumeration Technique

```
CREATE TABLE Post (
  post_id INT PRIMARY KEY,
  author_id INT NOT NULL,
  comment VARCHAR(5000) NOT NULL,
  timestamp DATETIME NOT NULL,
  votes INT,
  thread_id INT NOT NULL,
  path VARCHAR(2000),
  FOREIGN KEY (author_id)
     REFERENCES Author(author_id),
  FOREIGN KEY (thread_id)
     REFERENCES Thread(thread_id))
```

post_id	comment	author	path
1	Team outing anyone?	Andrew	1
2	Count me in! When? Where?	Sunil	1/2
3	Great idea!	Jen	1/3
4	I vote for SXSW	Jen	1/2/4
5	No, too crowded	Sunil	1/2/4/5
6	I'm open, whenever	Phil	1/2/6
7	How about Parkside?	Andrew	1/2/4/5/7



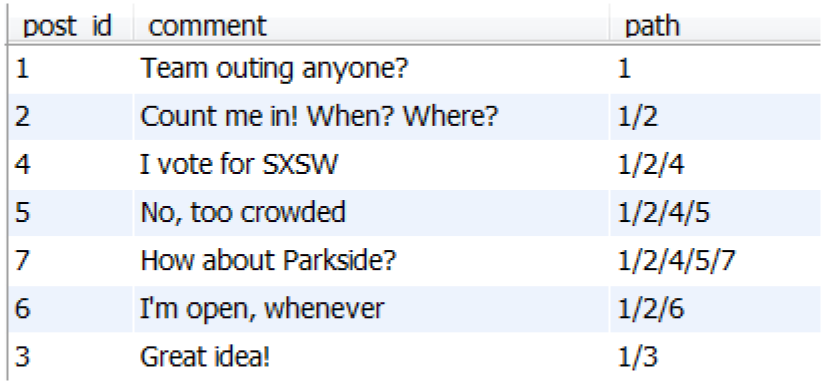
Using Path Enumeration



How can we find the chain of replies to `post_id` = 1 in SQL?

post_id	comment	author	path
1	Team outing anyone?	Andrew	1
2	Count me in! When? Where?	Sunil	1/2
3	Great idea!	Jen	1/3
4	I vote for SXSW	Jen	1/2/4
5	No, too crowded	Sunil	1/2/4/5
6	I'm open, whenever	Phil	1/2/6
7	How about Parkside?	Andrew	1/2/4/5/7

```
SELECT *
FROM Post
WHERE path LIKE '1%'
ORDER BY path
```

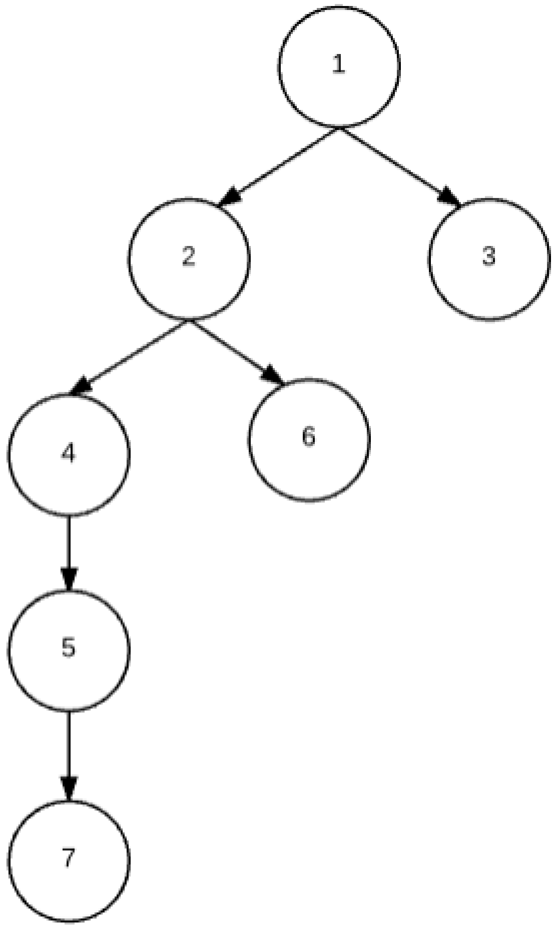


post_id	comment	path
1	Team outing anyone?	1
2	Count me in! When? Where?	1/2
4	I vote for SXSW	1/2/4
5	No, too crowded	1/2/4/5
7	How about Parkside?	1/2/4/5/7
6	I'm open, whenever	1/2/6
3	Great idea!	1/3

Concept Question 4

How can we count the posts per author in the subtree starting at `post_id = 2`?

post_id	comment	author	path
1	Team outing anyone?	Andrew	1
2	Count me in! When? Where?	Sunil	1/2
3	Great idea!	Jen	1/3
4	I vote for SXSW	Jen	1/2/4
5	No, too crowded	Sunil	1/2/4/5
6	I'm open, whenever	Phil	1/2/6
7	How about Parkside?	Andrew	1/2/4/5/7

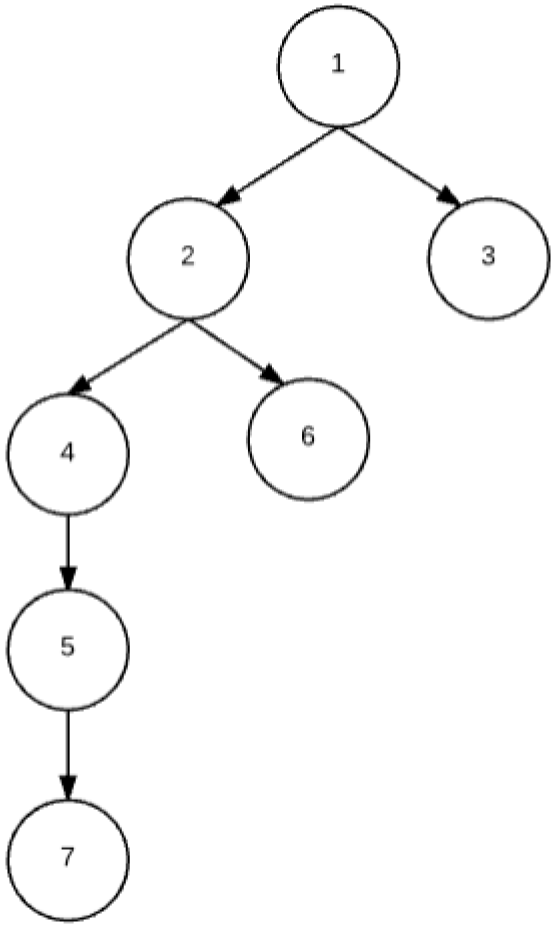


```
A. SELECT author, COUNT(*)
   FROM Post WHERE path LIKE '%/2/%'
   GROUP BY author

B. SELECT COUNT(*)
   FROM Post WHERE path LIKE '%/2%'

C. SELECT author, COUNT(*)
   FROM Post WHERE path LIKE '%/2%'
   GROUP BY author

D. None of the above
```

Inserting Nodes

How can we add a node rooted at `post_id = 7` in SQL?

post_id	comment	author	path
1	Team outing anyone?	Andrew	1
2	Count me in! When? Where?	Sunil	1/2
3	Great idea!	Jen	1/3
4	I vote for SXSW	Jen	1/2/4
5	No, too crowded	Sunil	1/2/4/5
6	I'm open, whenever	Phil	1/2/6
7	How about Parkside?	Andrew	1/2/4/5/7



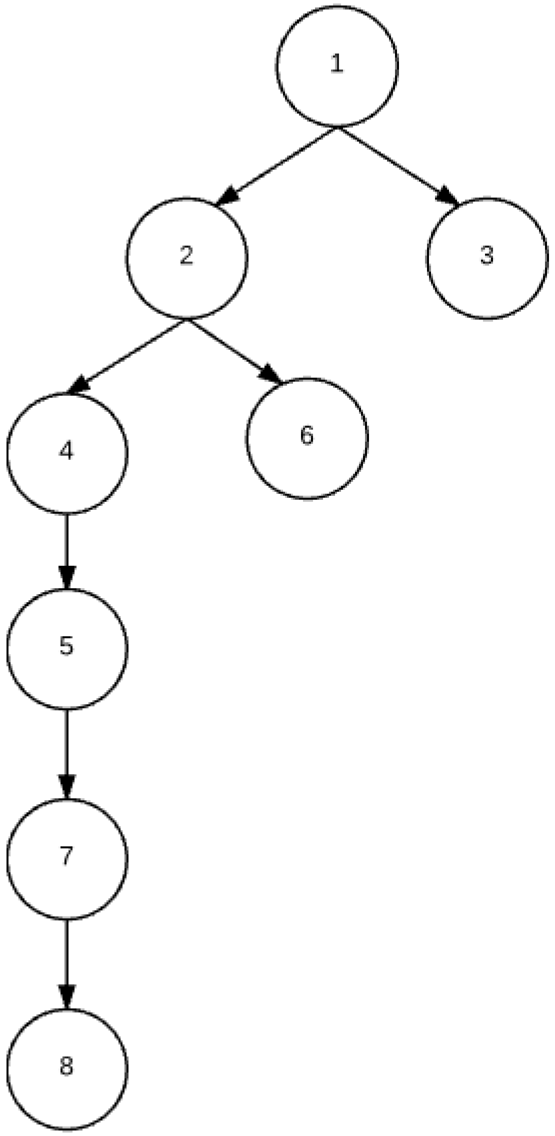
```
START TRANSACTION;
INSERT INTO Post (comment, author)
VALUES ('We''ll need a reservation', 'Jen');
UPDATE Post SET path = '1/2/4/7/' || LAST_INSERT_ID()
WHERE post_id = LAST_INSERT_ID();
COMMIT;
```

```
INSERT INTO Post (post_id, comment, author, path)
VALUES (8, 'We''ll need a reservation', 'Jen', '1/2/4/7/8')
```

Deleting Nodes and Subtrees

How can we remove a node from this tree in SQL?

post_id	comment	author	path
1	Team outing anyone?	Andrew	1
2	Count me in! When? Where?	Sunil	1/2
3	Great idea!	Jen	1/3
4	I vote for SXSW	Jen	1/2/4
5	No, too crowded	Sunil	1/2/4/5
6	I'm open, whenever	Phil	1/2/6
7	How about Parkside?	Andrew	1/2/4/5/7
8	We'll need a reservation	Jen	1/2/4/5/7/8



Removes node `post_id = 4`:

```
UPDATE Post SET path = REPLACE(path, '/4', '')
DELETE FROM Post WHERE post_id = 4
```

Removes the subtree rooted at `post_id = 4`:

```
DELETE FROM Post WHERE path LIKE '%/4%'
```

Homework for Next Time

- Read chapters 8 and 9 from the Beginning Database Design book
- Exercises at the end of chapters 8 and 9