CS 327E Lecture 6

Shirley Cohen

October 3, 2016
Roadmap

• This week: how to manipulate data in a MySQL database
• Next week: lab 2
• Extra time for labs 2 and 3 (and other schedule updates)
Homework for Today

- Chapter 7 from the *Learning SQL* book
- Exercises at the end of Chapter 7
Quiz Question 1

The SQL command for adding a new record to a table is:

A. ADD
B. PUT
C. NEW
D. INSERT
E. None of the above
Quiz Question 2

The SQL command for updating an existing record in a table is:

A. MODIFY
B. ALTER
C. UPDATE
D. CHANGE
E. None of the above
Quiz Question 3

The SQL command for deleting a record from a table is:

A. REMOVE
B. DELETE
C. DROP
D. TRUNCATE
E. None of the above
Quiz Question 4

The SQL command for retrieving a record from a table is:

A. SELECT
B. READ
C. RETRIEVE
D. GET
E. None of the above
Quiz Question 5

The SQL command to convert a VARCHAR type to an INTEGER type is:

A. CONVERT
B. TO_INT
C. CAST
D. TO_INTEGER
E. None of the above
Bank ERD
Concept Question 1

What is the correct sequence to populate the Bank tables?

A. Transaction, Account, Customer, Business, Individual

B. Customer, Business, Individual, Transaction, Account

C. Customer, Transaction, Business, Individual, Account

D. Customer, Account, Business, Individual, Transaction

E. Account, Transaction, Business, Customer, Individual
Insert Statements

Option 1:

```
INSERT INTO T (c_0, c_1, ..., c_n) VALUES (v_0, v_1, ..., v_n)
```

**e.g.**
```
INSERT INTO Product (id, name, description)
    VALUES (1700, 'iPhone7', 'latest Apple phone')
```

Option 2:

```
INSERT INTO T VALUES (v_0, v_1, ..., v_n)
```

**e.g.**
```
INSERT INTO Product VALUES (1700, 'iPhone7', 'latest Apple phone')
```

Option 3:

```
INSERT INTO T' SELECT * FROM T <WHERE c_0 = v_0>
```

**e.g.**
```
INSERT INTO Product_History SELECT * FROM Product
    WHERE id < 500
```
Concept Question 2

What can go wrong with this insert statement?

CREATE TABLE Product
(
    id INT PRIMARY KEY,
    name VARCHAR(1000),
    contact_id VARCHAR(100), -- comma-separated list
...
)

INSERT INTO Product (id, name, contact_id)
VALUES (1037, 'Evian Eau Minérale', '100,501,755');

A. Special characters (e.g. 'é' and ',') are not legal in SQL
B. Not using an auto-generated sequence for id
C. A VARCHAR(1000) type is too long for name
D. Table name Product is not descriptive enough
E. None of the above
Concept Question 3

What can go wrong with this insert statement?

CREATE TABLE Product
(
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    description VARCHAR(250),
    ...
)

CREATE TABLE Music
(
    song_id INT PRIMARY KEY,
    song_name VARCHAR(30),
    song_description VARCHAR(200),
    ...
)

INSERT INTO Product (id, name, description)
SELECT song_id, song_name, song_description FROM Music

A. Can’t select inside of an insert statement
B. Can result in duplicate entries for id
C. Can cause a gap in the sequence for id
D. Can’t assign a value to id
E. None of the above
Update Statements

Option 1:

```
UPDATE T SET c_0 = v_0, c_1 = v_1, c_n = v_n
```

e.g. UPDATE Product SET description = NULL

Option 2:

```
UPDATE T SET c_1 = v_1, c_2 = v_2, c_n = v_n WHERE c_0 = v_0
```

e.g. UPDATE Product SET description = '2015 Apple phone'
WHERE name = 'iPhone6s'

Option 3:

```
UPDATE T SET c_0 = (SELECT c_1 FROM T' WHERE c_2 = v_2)
```

e.g. UPDATE Product SET price = (SELECT MAX(price) from Product_Summary)

Note: T <> T’
Why transactions?

Suppose we have a discussion forum database and we want to create a new post rooted at post_id 7:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Team lunch anyone?</td>
<td>Andrew</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Count me in! Where? When?</td>
<td>Sunil</td>
<td>1/2</td>
</tr>
<tr>
<td>3</td>
<td>How about now?</td>
<td>Jen</td>
<td>1/3</td>
</tr>
<tr>
<td>4</td>
<td>I vote for Salt Lick</td>
<td>Jen</td>
<td>1/2/4</td>
</tr>
<tr>
<td>5</td>
<td>No, too far</td>
<td>Sunil</td>
<td>1/2/4/5</td>
</tr>
<tr>
<td>6</td>
<td>I'm open, whenever</td>
<td>Phil</td>
<td>1/2/6</td>
</tr>
<tr>
<td>7</td>
<td>How about Capitol Grille?</td>
<td>Andrew</td>
<td>1/2/4/5/7</td>
</tr>
</tbody>
</table>

Option 1:

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

Option 2:

```sql
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
```
Deleting Nodes and Subtrees

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

<table>
<thead>
<tr>
<th>post_id</th>
<th>comment</th>
<th>author</th>
<th>path</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Team lunch anyone?</td>
<td>Andrew</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Count me in! Where? When?</td>
<td>Sunil</td>
<td>1/2</td>
</tr>
<tr>
<td>3</td>
<td>How about now?</td>
<td>Jen</td>
<td>1/3</td>
</tr>
<tr>
<td>4</td>
<td>I vote for Salt Lick</td>
<td>Jen</td>
<td>1/2/4</td>
</tr>
<tr>
<td>5</td>
<td>No, too far</td>
<td>Sunil</td>
<td>1/2/4/5</td>
</tr>
<tr>
<td>6</td>
<td>I'm open, whenever</td>
<td>Phil</td>
<td>1/2/6</td>
</tr>
<tr>
<td>7</td>
<td>How about Capitol Grille?</td>
<td>Andrew</td>
<td>1/2/4/5/7/</td>
</tr>
<tr>
<td>8</td>
<td>We'll need a reservation</td>
<td>Jen</td>
<td>1/2/4/5/7/8</td>
</tr>
</tbody>
</table>

Removes post_id 4:

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

Removes the subtree rooted at post_id 4:

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