Tuesday, 22 October 2013 General Assemblation SQL

Table Manipulations



INSERT Table Data

1. Adding President Obama to presidents table

```
INSERT INTO
       sampdb.president
       SET
           last name
                            'Obama',
           first name
                            'Barack',
           suffix
                            NULL,
                            'Honolulu',
           city
           state
                            'HI',
                            '1961-08-04',
           birth
           death
                            NULL
12
13 ;
14
```

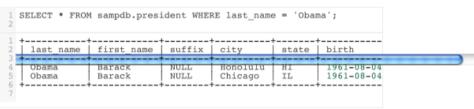
2. Check:

Alternate INSERT Syntax

- 1. That entry is wrong, so let's add the right one
- 2. VALUES provides another syntax for adding data

```
INSERT INTO
sampdb.president
VALUES ('Obama', 'Barack', NULL, 'Chicago', 'IL', '1961-08-04',
;
```

3. Check:



4. VALUES statement have the nice property of allowing multiple usages

so we can load lots of data with them.

see the database setup files for examples of this

) 4 b

14 b

Editing database data

1. We could also fix the problem with UPDATE

```
UPDATE
       sampdb.president
   SET
       state = 'IL',
       city = 'Chicago'
   WHERE
       last name = 'Obama'
 8
 9
10
11
                  first name
                                 suffix
                                           city
                                                               birth
     last name
                                                      state
                                           Chicago
 4
     Obama
                  Barack
                                 NULL
                                                      IL
                                                               1961-08-04
 5
                                           Chicago
                                                      IL
                                                               1961-08-04
     Obama
                  Barack
                                 NULL
```

DELETE -- Removing Records

1. Now we have two entries, let's remove one

```
DELETE FROM sampdb.president WHERE last_name = 'Obama' LIMIT 1;

SELECT * FROM sampdb.president WHERE last_name = 'Obama';

last_name | first_name | suffix | city | state | birth

Obama | Barack | NULL | Chicago | IL | 1961-08-04
```

DELETE -- Removing Many Records

1. Without the LIMIT, DELETE takes all records

```
DELETE FROM sampdb.president WHERE last_name = 'Obama';

SELECT * FROM sampdb.president WHERE last_name = 'Obama';

1
2
```

DELETE -- Removing LOTS Many Records

1. Be careful with DELETE!!

```
DELETE FROM sampdb.president;

SELECT
COUNT(*)
FROM
Sampdb.president;

COUNT(*)

COUNT(*)
```

2. Here, not a big deal:

```
1 INSERT INTO sampdb.president
2 VALUES skshington 'George', NULL, 'Nakefield', 'VA', '1732-02-22'
3 ('Adams' John', NULL, 'Braintree', 'NA', '1735-10-30', '1826-
5 ('Jefferson', 'Thomas', NULL, 'Albemarle County', 'VA', '1746-6
6 ('Madison', 'James', NULL, 'Not Conway', 'VA', '1751-216', 'Yan', '1758-0
7 ('Monroe', 'James', NULL, 'Westmoreland County', 'VA', '1758-0
9 ;
```

- 3. This is why databases have permissions.
- I don't leave uncommented DELETE statements lying around.

DROP neither.

Databases can protect against these mistakes

UNIQUE provides a data consistency rule

Prevents insertion of a record with a first_name, last_name tuple matching an extant record

Thus prevents addition of data inconsistent with data already entered

2. "Consistent" != "Right"

If we enter the Hawaii values first, the Illinois ones are prevented

But, we are forced to solve the problem in some way that doesn't risk mistakes like two duplicating records

Constraints are nice when operations are large

```
INSERT INTO
sampdb.student
SELECT
first_name AS name,
'M' AS sex,
NULL as student_id
FROM
sampdb.president
;
```

We're adding 43 records to the data

Getting harder to see what consistency assumptions we might be violating

Notice that expressions in SELECT fields are handy

We can use these to handle data requirements of a target table unaddressed by our source table

And, to handle edge cases

```
1 INSERT INTO
2    sampdb.student
3 SELECT
4    first_name AS name,
5    IF(first_name='Hillary', 'F', 'M') AS sex,
6    NULL as student_id
7 FROM
8    sampdb.president
9;
10
```

More Query Syntax



UNION combines query results

 Say we want a list of names used in all our more_syntax

```
1 SELECT name AS first name FROM sampdb.student
2 UNION
3 SELECT first name FROM sampdb.president
4 UNION
5 SELECT first name FROM sampdb.member
6 ORDER BY first name ASC
7 LIMIT 5
9
10
     first name
    Abby
    Abraham
 6
    Alan
    Alma
     Amanda
10
```

Again, field manipulation bridges data

1. Getting last_name is a problem:

```
1 -- PAILS
2 SELECT name AS first_name FROM sampdb.student
3 UNION
4 SELECT first_name, last_name FROM sampdb.president
5 UNION
6 SELECT first_name, last_name FROM sampdb.member
7 ORDER BY first_name ASC
8 LIMIT 5
9 ;
10
11
12
```

2. Solved by generating a 'new' table on the fly:

Data Cleaning

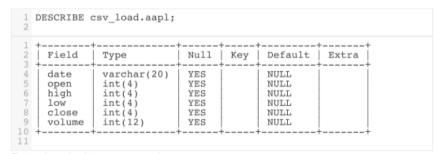


Remember our AAPL data?

+	+	high		close	+ volume	
3 +	open		low		++	
4 1-Apr-13 5 1-Aug-13	442 456	444 457	428 453	429 457	13918927 7366046	
6 1-Feb-13	459	457	448	454	19266727	
7 1-Jul-13	403	412	401	409	13970435	
8 1-Mar-13	438	438	430	430	19730256	
0	,				,,	

1. That ain't the right order

Date data isn't typed properly



Date data isn't type properly

Server is sorting alphabetically, because that's how it sorts strings

So: Get a date typed table

```
1 DROP TABLE IF EXISTS csv load.aapl dated;
2 CREATE TABLE csv load.aapl dated (
       -- date
                      VARCHAR(10),
       date
                   DATE,
                   DECIMAL(5, 2),
       open
6
       high
                   DECIMAL(5, 2),
       low
                   DECIMAL(5, 2),
       close
                   DECIMAL(5, 2),
                   INTEGER (12)
       volume
10);
```

Notice that field date is here assigned type DATE

But to load that field, we need YYYY-MM-DD formatted strings

Create those strings in a query

```
SELECT
CONCAT(
CONCAT(
'20',
RIGHT(date, 2)
               );,

MONTH(

STR TO DATE(

- MID)

date;

LOCKTE('-', date) + 1,

LOCKTE('-', date) + 1,

( LOCKTE('-', date, 4) - LOCKTE('-', date) - 1 )
                    CONCAT(
                                 LENGTH(
LEFT(
date, LOCATE('-', date) - 1
                                  ) = 1, '0', ''
                           LEFT(date, LOCATE('-', date) - 1 )
             ) AS date,
open,
high,
low,
close,
wolume
     csv_load.aapl
          date
                               open high low
                                                                      close
                                                                                    volume
         2013-10-01
2013-9-30
2013-9-26
2013-9-25
2013-9-24
                                                                                   12638665
9291344
8472169
11319881
13012249
                                  478
477
486
489
495
                                              489
482
489
490
495
                                                           478
474
484
481
488
                                                                         488
477
486
482
489
```

1. Debugging that is where your text editor matters

Load that

```
INSERT INTO
         csv load.aapl dated
    SELECT
         CONCAT (
              CONCAT(
'20',
RIGHT(date, 2)
              MONTH (
                   STR_TO_DATE(
                              date,
                              LOCATE('-', date) + 1,
( LOCATE('-', date, 4) - LOCATE('-', date) - 1 )
                   '%b'),
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
              CONCAT(
                         LENGTH(
                                   date, LOCATE('-', date) - 1
                         ) = 1, '0', ''
                   ),
LEFT(date, LOCATE('-', date) - 1 )
         ) AS date,
         open,
         high,
         low,
         close,
36
         volume
37 FROM
38
         csv_load.aapl
39 ;
40
```

Voila

+ date	open	high	low	close	volume
2012-10-04 2012-10-05 2012-10-08 2012-10-09 2012-10-10	671.00 665.00 647.00 639.00 640.00	674.00 666.00 648.00 640.00 645.00	666.00 651.00 636.00 624.00 637.00	667.00 653.00 638.00 636.00 641.00	13241259 21214444 22784981 29949841 18226990



Advanced MySQL

1. Put that in a user defined function

Now the call is much simpler

```
SELECT
       csv load.reformatter(date) AS date,
       open,
       high,
 5
       low,
 6
       close,
       volume
   FROM
       csv load.aapl
   ORDER BY
       date
  LIMIT 5
13;
14
                            high
                                           close
                                                    volume
     date
                    open
                                    low
     2012-10-04
                     671
                             674
                                     666
                                              667
                                                     13241259
     2012-10-05
                     665
                             666
                                     651
                                              653
                                                    21214444
 6
     2012-10-08
                     647
                             648
                                     636
                                              638
                                                    22784981
     2012-10-09
                                     624
                     639
                             640
                                              636
                                                    29949841
     2012-10-10
                     640
                             645
                                     637
                                              641
                                                    18226990
10
```

This is "Abstraction" or "Encapsulation"

1. We "hide" complexity in some "object"

Call on the "object" when we need that complex task

2. Here, "functions" and "views" are "objects"

We use them to make references simpler

Data manipulation is irreducibly complex

- 1. Tools like Excel abstract a lot of that for us
- 2. But, we are confined to the tools they offer us
- 3. Learning syntax allows us to shape our own tools
- Learning concepts gives us new vision on using those tools
- 5. That's what programming is about, and why it matters