Plan for Today

• Reading Quiz
• MySQL + JSON
• Final Project Assignment
Readings for Today

- JSON Data Type from the MySQL Reference Manual (section 12.6)
- JSON Functions from the MySQL Reference Manual (section 13.16)
Question 1

Which of the following statements is **false** about MySQL support for JSON:

A. MySQL has a native JSON datatype
B. MySQL converts JSON documents to a binary format
C. MySQL indexes JSON documents directly
D. MySQL supplies a number of functions for operating on JSON data
E. MySQL automatically validates JSON documents that are stored in JSON columns
In the context of JSON, normalization means the process of removing duplicate keys (or names) from a document.

A. True
B. False
Question 3

What does the path expression below evaluate to?

$.retweeted_status.entities.user_mentions[0].indices[0]

A. {}  
B. 75  
C. NULL  
D. 91  
E. “id” : 104481993
Let \( j \) be to the JSON document shown. What does the select statement below evaluate to?

```sql
select JSON_EXTRACT(j, '$.retweeted_status.entities.user_mentions[0].id');
```

A. "indices" : [75, 91]  
B. 75  
C. 91  
D. 104481993  
E. None of the above
Let \( j \) be the JSON document shown. What does the `select` statement below evaluate to?

```sql
select JSON_SEARCH(j, 'one', 'RT');
```

A. "$\cdot text"

B. "$\cdot *"

C. {}

D. NULL

E. None of the above
Demo 1

JSON Editor Online

```
"favorited": false,
"retweeted_status": {
  "contributors": null,
  "truncated": false,
  "text": "Meeting tonight at 7pm in JGB 2.216! We'll have Chick-fil-A and a rep from @airliquidegroup will present! #utexas",
  "is_quote_status": false,
  "in_reply_to_status_id": null,
  "id": 7941648880169238500,
  "favorite_count": 1,
}
```

JSONPath Online Evaluator - jsonpath.com

**Inputs**

**JSONPath Syntax**

```
$.retweeted_status.id
```

Example ‘$.phoneNumbers[0].type’ See also JSONPath expressions

**Evaluation Results**
UT Class Enrollment & Twitter

Logical ERD - UT Class Enrollment - CS 327E Fall 2016
New DDL

drop table if exists Major;
create table Major (  
  code int auto_increment primary key,  
  name varchar(32) not null,  
  college varchar(32) not null
);

alter table Student add column major_code int;
alter table Student add constraint fk_major_code  
  foreign key (major_code) references Major(code);

drop table if exists Tweet;
create table Tweet (  
  tweet_id varchar(32) generated always  
  as (json_unquote(json_extract(tweet_doc, '$.id_str'))) stored primary key,  
  screen_name varchar(32) generated always  
  as (json_unquote(json_extract(tweet_doc, '$.user.screen_name'))) stored,  
  created_at datetime generated always  
  as (str_to_date(json_unquote(json_extract(tweet_doc, '$.created_at')),
    '%a %b %d %H:%i:%s +0000 %Y')) stored,
  tweet_doc json,  
  major_code int,  
  foreign key (major_code) references Major(code)
);
```python
def do_data_pull(api_inst):
    sql_query = "select code, name from Major order by name"

    try:
        conn = create_connection()
        db_cursor = conn.cursor()
        query_status = run_stmt(db_cursor, sql_query)
        resultset = db_cursor.fetchall()

        for record in resultset:
            major_code = record[0]
            major_name = record[1]

            utexas_query = "(#UTexas OR @UTAustin OR url:utexas.edu) AND "
            twitter_query = utexas_query + "'" + major_name + "'"
            print "twitter_query: " + twitter_query
            twitter_cursor = tweepy.Cursor(api_inst.search, q=twitter_query, lang="en")

            for page in twitter_cursor.pages():
                for item in page:
                    json_str = json.dumps(item._json)
                    print "found a " + major_name + " tweet"
                    insert_stmt = "insert into Tweet(tweet_doc, major_code) values(%s, %s)"
                    run_prepared_stmt(db_cursor, insert_stmt, (json_str, major_code))
                    do_commit(conn)

    except pymysql.Error as error:
        is_success = False
        print "do_data_pull: " + e.strerror
```
Demo 2
Concept Question 1

We want to extend the Twitter Client to check for duplicate tweets before doing the insert into MySQL. Assume that in Python we extract the id of the tweet and store the value in the variable $id. How can we formulate a SQL query that checks for duplicate tweets given $id?

A. select count(*) from Tweet where tweet_id = $id
B. select * from Tweet order by tweet_id
C. select count(distinct tweet_id) from Tweet where tweet_id = $id
D. select * from Tweet where tweet_id = $id
E. select count(tweet_id) from Tweet where tweet_id = $id
We want to implement a more accurate count of tweets per UT major. More specifically, we want to filter out all retweets and only add up the origin tweets. Assume that for all tweets, we extract the origin tweet id from the tweet and we store this value in a new field called Tweet.origin_tweet_id. How can we modify the query below to only count unique tweets?

```
select m.name, m.code, count(t.tweet_id) as tweet_count
from Major m left outer join Tweet t
on m.code = t.major_code
group by m.name, m.code
order by tweet_count desc;
```

A. Replace: `count(t.tweet_id)` with: `count(t.origin_tweet_id)`
B. Replace: `count(t.tweet_id)` with: `count(distinct t.origin_tweet_id)`
C. Change the outer join to an inner join
D. Change the left outer join to a right outer join
E. None of the above
Final Project

http://www.cs.utexas.edu/~scohen/project/final_project.pdf