服务器密码GUshouchuan523

私有ip地址192.168.0.201

公网1.94.60.68

启动neo4j：进入解压后的文件夹的 bin 目录下，使用./neo4j console 或./neo4j start 命令即可启动数据库

mysql密码123456 启动mysql服务器：sudo service mysql start 然后mysql -uroot -p 使用完毕 sudo service mysql stop

wget <https://obs-community.obs.cn-north-1.myhuaweicloud.com/obsutil/current/obsutil_linux_amd64.tar.gz>

tar -zvxf obsutil\_linux\_amd64.tar.gz

./obsutil config -i=ZTZPBN7BWSS01GDSZ6EF -k=5S6Y7twdUz6GwNsYx2R8m9ZChJ842lhR0GFoEPRl -e=obs.cn-east-3.myhuaweicloud.com

./obsutil cp obs://big-data-management/dataset/data\_for\_mysql.zip ../data/mysql

./obsutil cp obs://big-data-management/dataset/data\_for\_mongo.zip ../data/mongo

./obsutil cp obs://big-data-management/dataset/data\_for\_neo4j.zip ../data/neo4j

a) . JSON 基本查询:

1.在 business 表中,查询 city 位于 Tampa 的商户所有信息,

按被评论数降序排序,限制返回 10 条. 正确

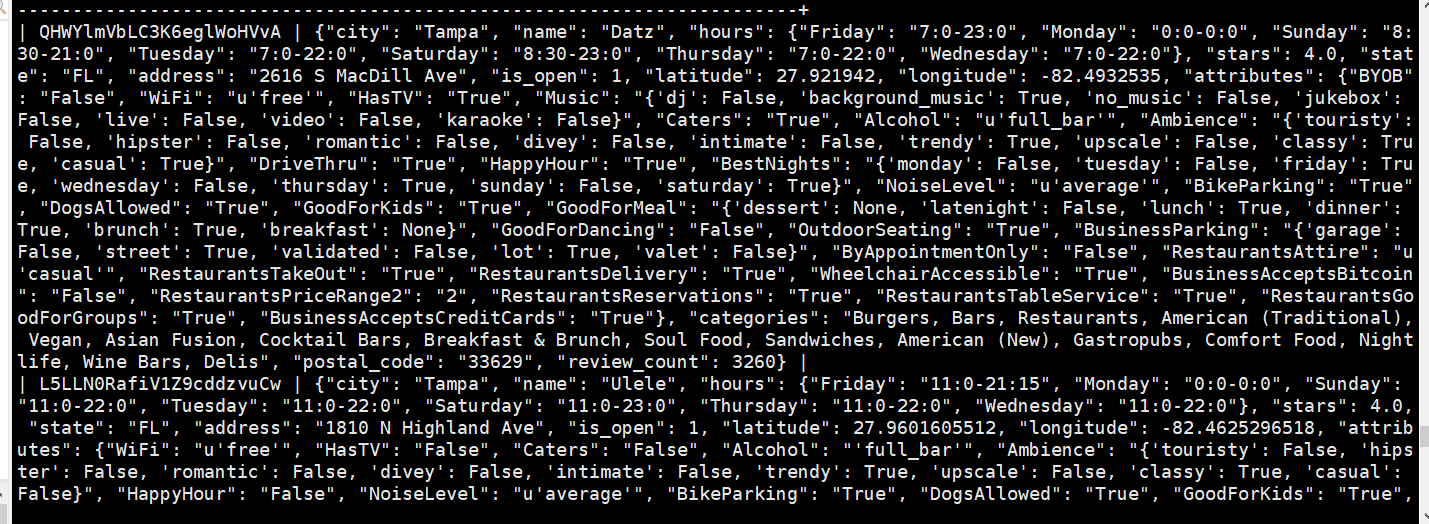
SELECT \*

FROM business

WHERE JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.city')) = 'Tampa'

ORDER BY JSON\_EXTRACT(business\_info, '$.review\_count') DESC

LIMIT 10;



2. 在 business 表中,查询前五条记录的 business\_info 列和

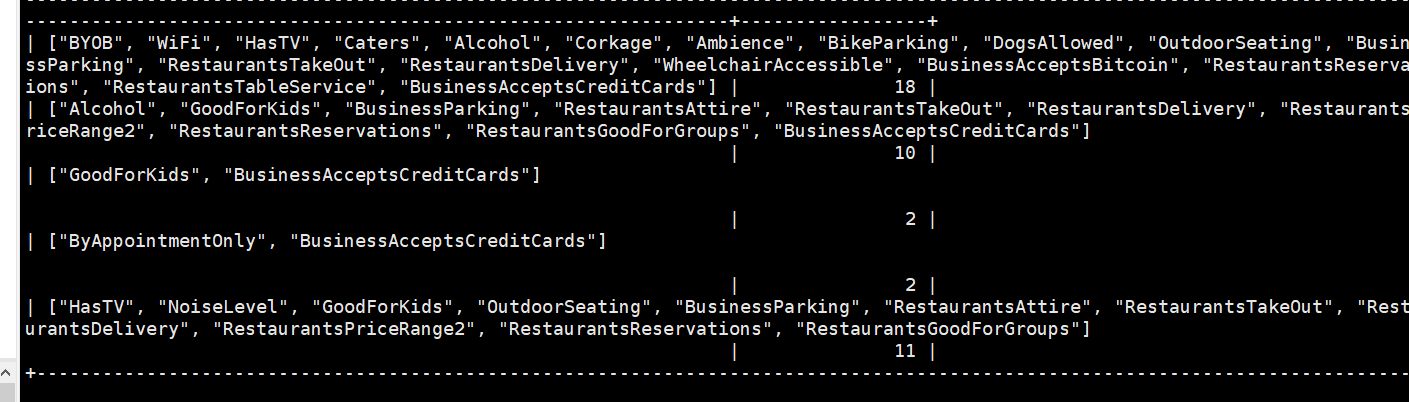
business\_info 中attributes 的所有键,以 Json 数组形式返回,同时返回对应键的数量.

SELECT JSON\_KEYS(JSON\_EXTRACT(business\_info, '$.attributes')) AS attribute\_keys,

JSON\_LENGTH(JSON\_EXTRACT(business\_info, '$.attributes')) AS attribute\_count

FROM business

LIMIT 5;



3. 在 business 表中,查询该表 business\_info 列中 name,

stars, attributes 的内容和JSON 类型,限制返回行数为 5.

SELECT

JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.name')) AS name,

JSON\_TYPE(JSON\_EXTRACT(business\_info, '$.name')) AS name\_type,

JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.stars')) AS stars,

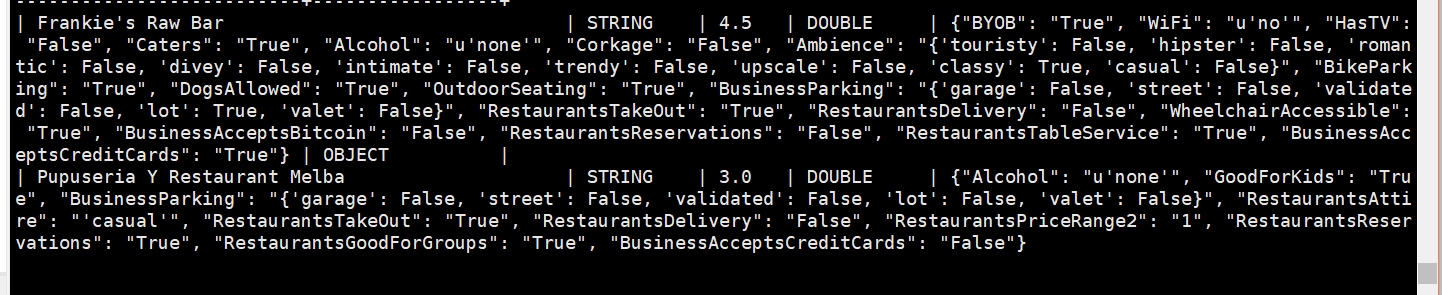
JSON\_TYPE(JSON\_EXTRACT(business\_info, '$.stars')) AS stars\_type,

JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.attributes')) AS attributes,

JSON\_TYPE(JSON\_EXTRACT(business\_info, '$.attributes')) AS attributes\_type

FROM business

LIMIT 5;



4.在business表中,查询拥有电视("HasTV"是"True")且星期天不营业

(Sunday键不存在,当然也可以hours键不存在,即is null)的商户,

返回它的名字(不带双引号),属性,营业时间,并按名字升序排序,限制10条记录.正确

SELECT JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.name')) AS name, JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.attributes')) AS attributes, JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.hours')) AS hours

FROM business

WHERE JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.attributes.HasTV')) = 'True' AND (JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.hours.Sunday')) IS NULL OR JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.hours')) IS NULL)

ORDER BY name ASC

LIMIT 10;



5.使用explain查看select \* from user where user\_info->'$.name'='Wanda'的

执行计划,其中执行计划按JSON格式输出;并且实际执行一次该查询,请注意观察语句

消耗的时间并与MongoDB的查询方式进行对比(MongoDB要执行此查询要求,相应

的语句是什么?执行计划是怎样的?并给出查询效率对比).

b）json的增删修改

6. 在busines表中,查询id为4r3Ck65DCG1T6gpWodPyrg的商户business\_info,这 里对info列的显示需要使用JSON\_PRETTY(business\_info)让可读性更高

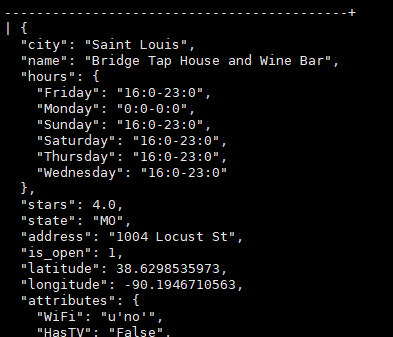
查询原来状态：

SELECT JSON\_PRETTY(JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$'))) AS business\_info

FROM business

WHERE business\_id = '4r3Ck65DCG1T6gpWodPyrg';

原来状态：



修改，对比：

-- 查询修改前的数据

SELECT JSON\_PRETTY(JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$'))) AS business\_info\_before

FROM business

WHERE business\_id = '4r3Ck65DCG1T6gpWodPyrg';

-- 更新商户数据

UPDATE business

SET business\_info = JSON\_SET(

business\_info,

'$.hours.Tuesday', '16:0-23:0',

'$.stars', 4.5,

'$.attributes.WiFi', 'Free'

)

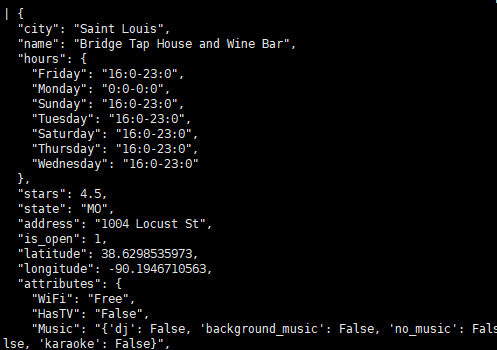
WHERE business\_id = '4r3Ck65DCG1T6gpWodPyrg';

-- 查询修改后的数据

SELECT JSON\_PRETTY(JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$'))) AS business\_info\_after

FROM business

WHERE business\_id = '4r3Ck65DCG1T6gpWodPyrg';



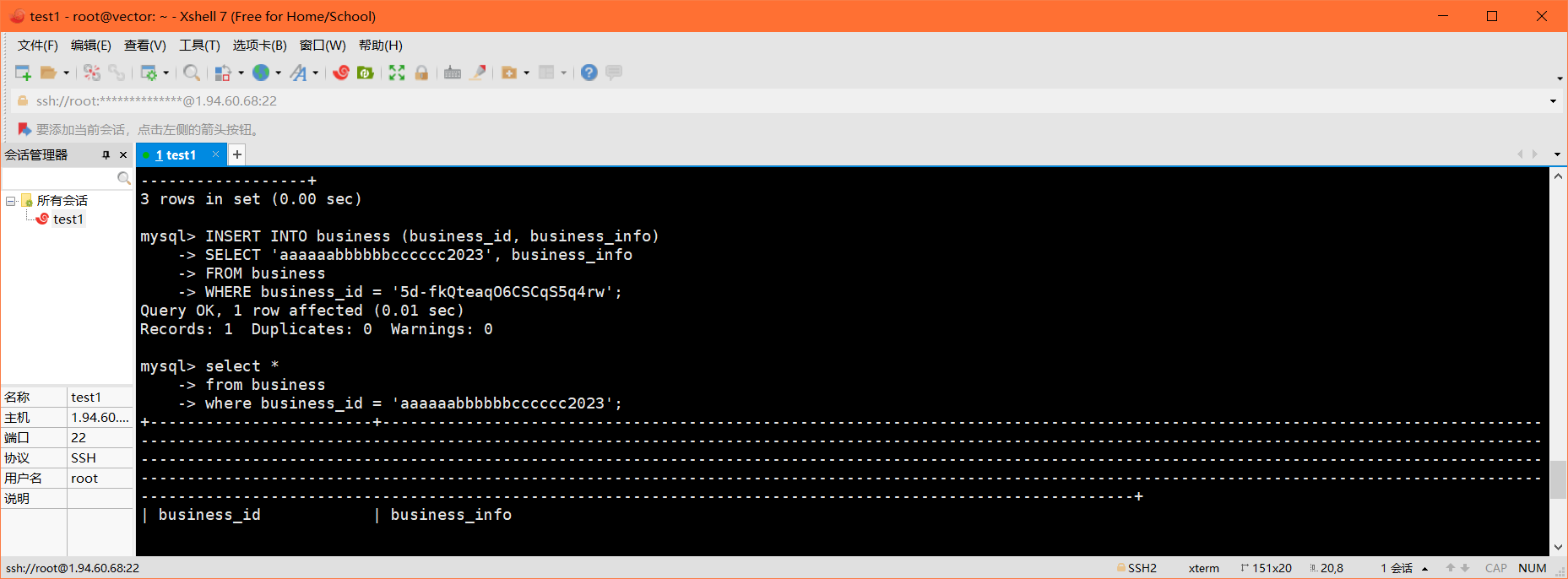
7. 向 business 表插入一个 id 是'aaaaaabbbbbbcccccc2023'的商户,其商户信息与 id 为'5d-fkQteaqO6CSCqS5q4rw'的商户完全一样,插入完成之后,将这个新记录 的 info 中的 name 键值对删去,最后查询'aaaaaabbbbbbcccccc2023'的所有信息.

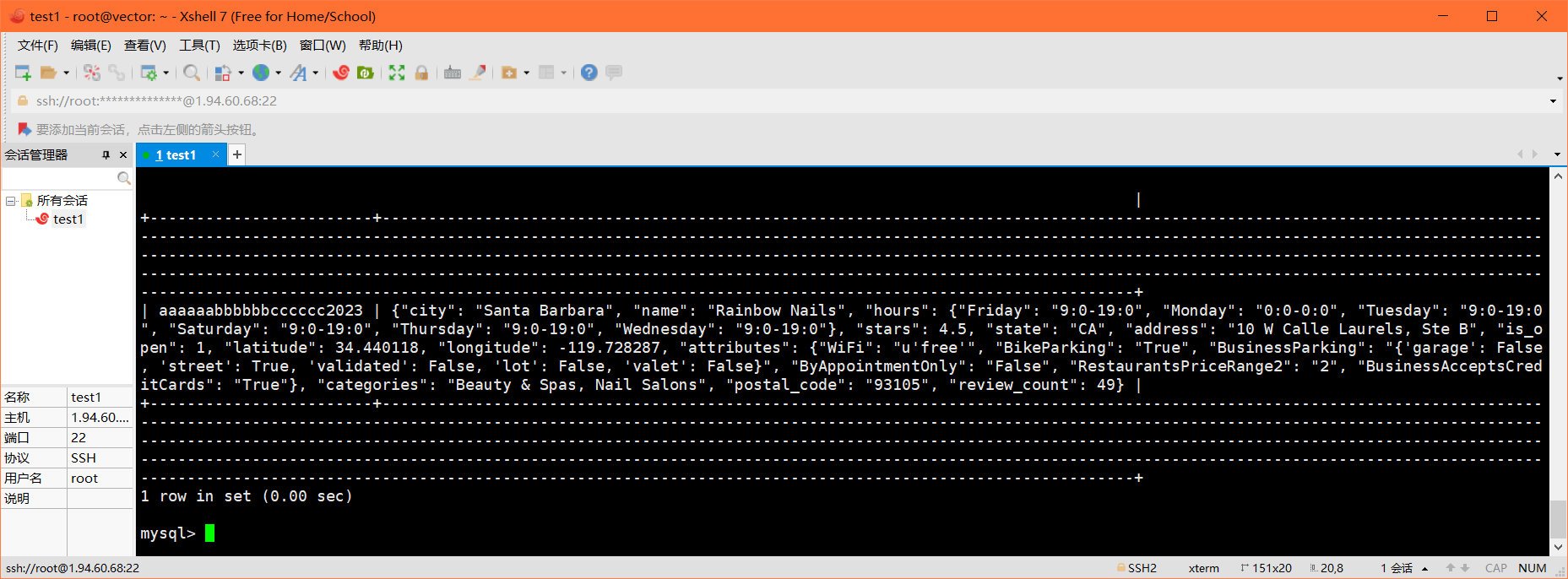
INSERT INTO business (business\_id, business\_info)

SELECT 'aaaaaabbbbbbcccccc2023', business\_info

FROM business

WHERE business\_id = '5d-fkQteaqO6CSCqS5q4rw';





删除name键值对：

UPDATE business

SET business\_info = JSON\_REMOVE(business\_info, '$.name')

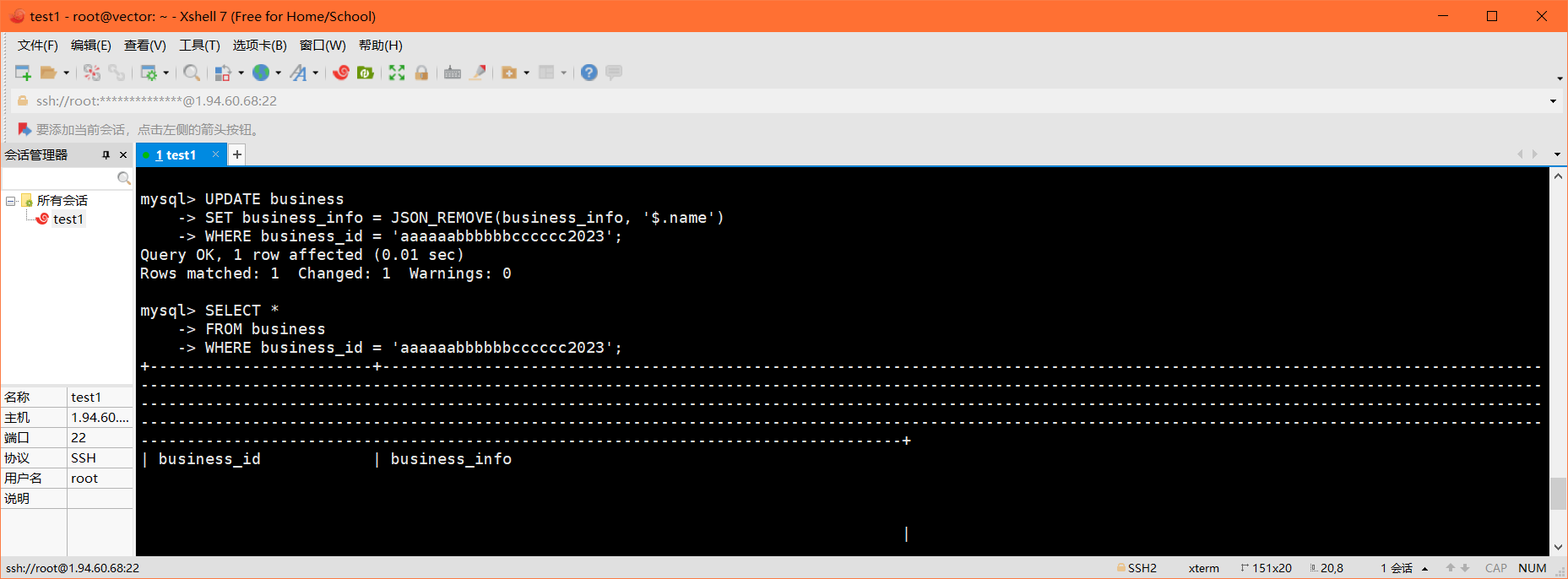
WHERE business\_id = 'aaaaaabbbbbbcccccc2023';

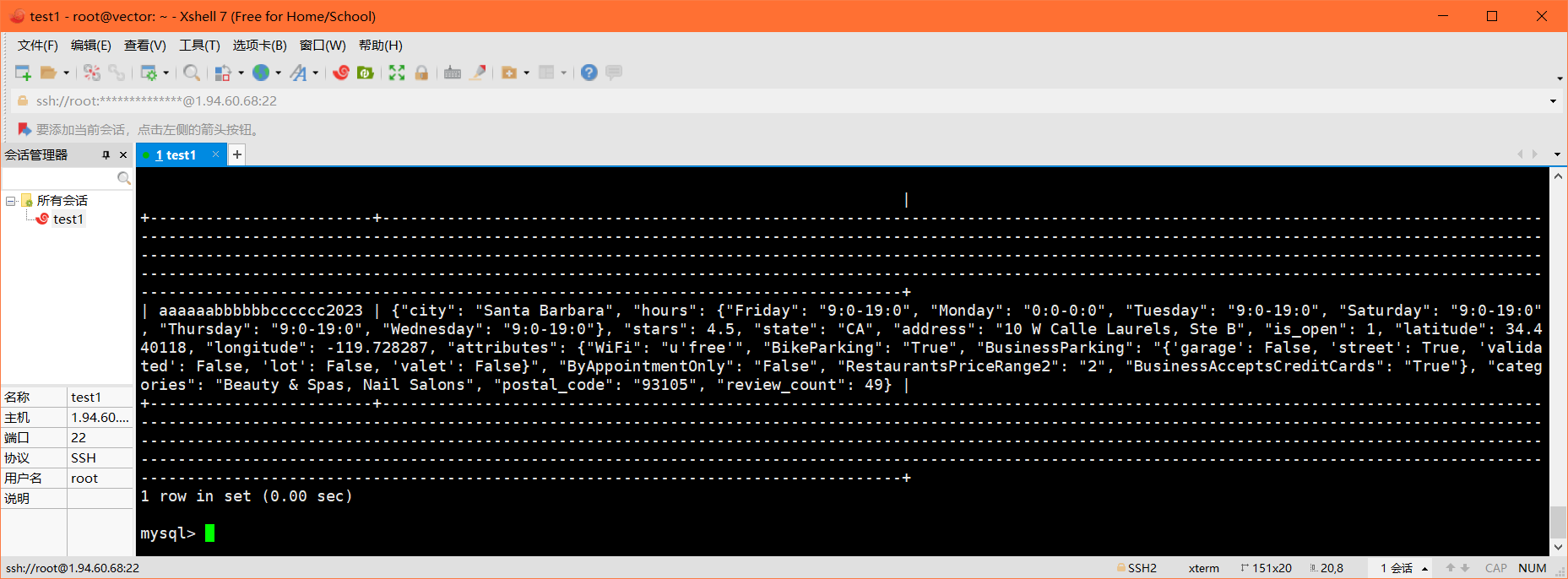
查询

SELECT \*

FROM business

WHERE business\_id = 'aaaaaabbbbbbcccccc2023';





C）json聚合

8. 在 business 表的所有商户中,按所在州(state)进行聚合,对于每个州返还一个 JSON 对象,这个对象的每一个键值对中,key 是城市,value 是城市总共出现的 次数,结果按照州名升序排序. 提示:这里需要去掉引号让 group by 的 key 更好一些

SELECT

state,

JSON\_OBJECTAGG(city, city\_count) AS city\_occurrences

FROM (

SELECT

TRIM(BOTH '"' FROM JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.state'))) AS state,

TRIM(BOTH '"' FROM JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.city'))) AS city,

COUNT(\*) AS city\_count

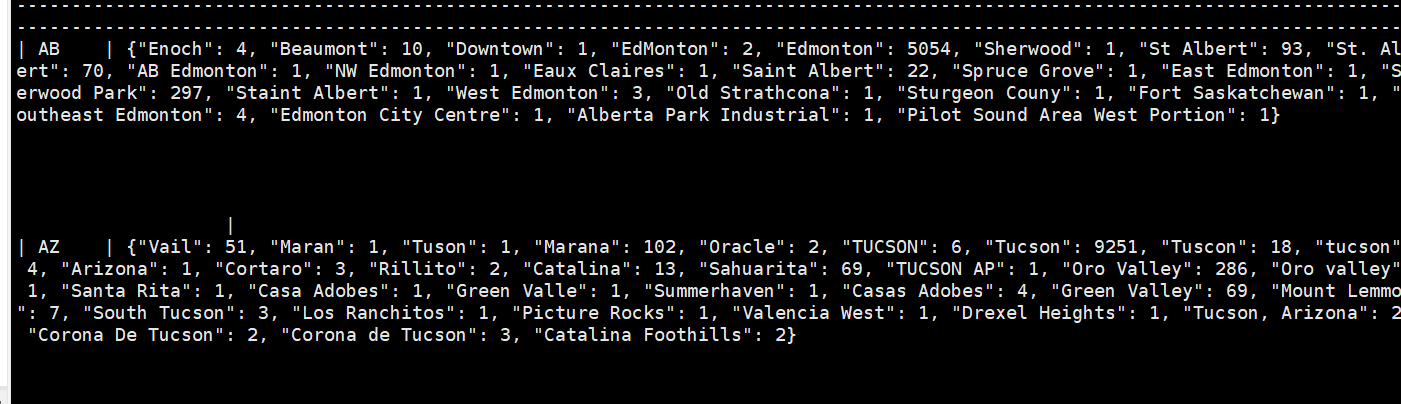
FROM business

GROUP BY state, city

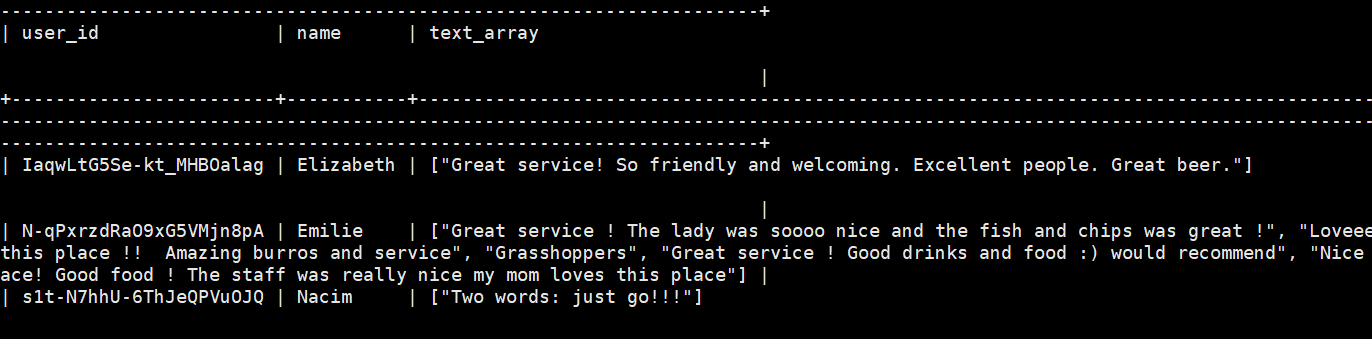
) AS city\_counts

GROUP BY state

ORDER BY state;



9.查询user\_id是'\_\_1cb6cwl3uAbMTK3xaGbg'的用户的所有朋友的建议,并按用户 进行分组聚合,对每一个用户,返还用户的id,用户的名字,由他/她的所有建议构成的 字符串数组,最后按名字升序排序输出。

Select u.user\_id as user\_id,  
json\_unquote(json\_extract(u.user\_info, ‘$.name‘)) as name,  
json\_arrayagg(t.tip\_info->‘$.text’) as text\_array  
from `user` u  
left join tip t on t.user\_id = u.user\_id  
where regexp\_like(u.user\_info->‘$.friends‘, ’\_\_1cb6cwl3uAbMTK3xaGbg‘)  
group by user\_id, name  
order by name asc;

d) . JSON实用函数的使用:

10.在business表中,分别查询城市在EdMonton和Elsmere的商铺,并使用 JSON\_OVERLAPS()判断在这两个城市的商铺之间是否在一周中至少有一天营业时间完全重合(即hours对象中至少有一个键值对相同,不考虑键不存在的情况),是返回 1,不是返回0。

SELECT

b1.business\_info->'$.name' AS name\_1,

b1.business\_info->'$.city' AS city\_1,

b1.business\_info->'$.hours' AS hours\_1,

b2.business\_info->'$.name' AS name\_2,

b2.business\_info->'$.city' AS city\_2,

b2.business\_info->'$.hours' AS hours\_2,

IF(

(

JSON\_UNQUOTE(JSON\_EXTRACT(b1.business\_info->'$.hours', '$.Monday')) = JSON\_UNQUOTE(JSON\_EXTRACT(b2.business\_info->'$.hours', '$.Monday'))

OR (JSON\_UNQUOTE(JSON\_EXTRACT(b1.business\_info->'$.hours', '$.Monday')) = "0:0-0:0" AND JSON\_UNQUOTE(JSON\_EXTRACT(b2.business\_info->'$.hours', '$.Monday')) = "0:0-0:0")

)

AND JSON\_UNQUOTE(JSON\_EXTRACT(b1.business\_info->'$.hours', '$.Tuesday')) = JSON\_UNQUOTE(JSON\_EXTRACT(b2.business\_info->'$.hours', '$.Tuesday'))

AND JSON\_UNQUOTE(JSON\_EXTRACT(b1.business\_info->'$.hours', '$.Wednesday')) = JSON\_UNQUOTE(JSON\_EXTRACT(b2.business\_info->'$.hours', '$.Wednesday'))

AND JSON\_UNQUOTE(JSON\_EXTRACT(b1.business\_info->'$.hours', '$.Thursday')) = JSON\_UNQUOTE(JSON\_EXTRACT(b2.business\_info->'$.hours', '$.Thursday'))

AND JSON\_UNQUOTE(JSON\_EXTRACT(b1.business\_info->'$.hours', '$.Friday')) = JSON\_UNQUOTE(JSON\_EXTRACT(b2.business\_info->'$.hours', '$.Friday'))

AND JSON\_UNQUOTE(JSON\_EXTRACT(b1.business\_info->'$.hours', '$.Saturday')) = JSON\_UNQUOTE(JSON\_EXTRACT(b2.business\_info->'$.hours', '$.Saturday'))

AND JSON\_UNQUOTE(JSON\_EXTRACT(b1.business\_info->'$.hours', '$.Sunday')) = JSON\_UNQUOTE(JSON\_EXTRACT(b2.business\_info->'$.hours', '$.Sunday'))

, 1, 0) AS has\_same\_opentime

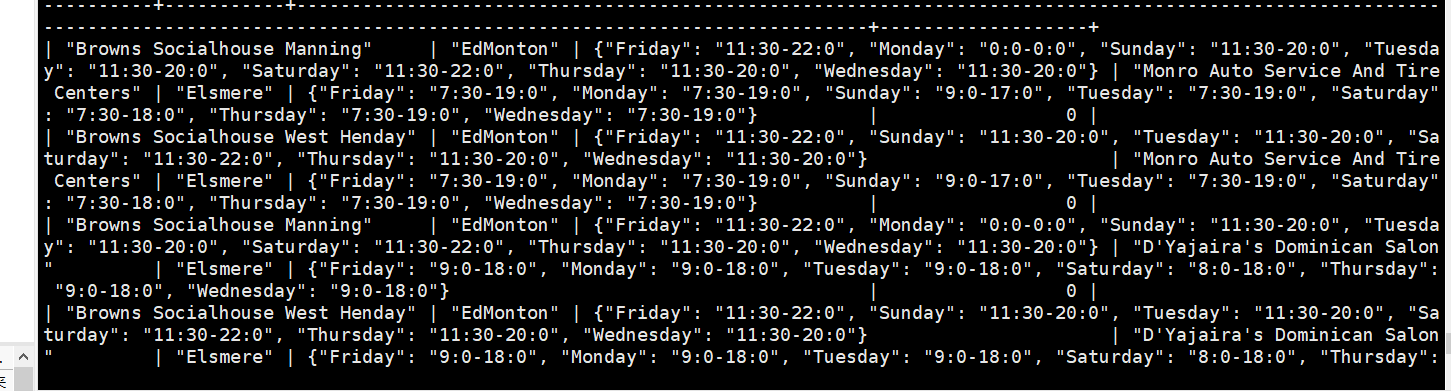
FROM business AS b1

JOIN business AS b2

ON b1.business\_id != b2.business\_id

WHERE b1.business\_info->'$.city' = 'EdMonton'

AND b2.business\_info->'$.city' = 'Elsmere';



11. 在user表中,查询funny大于2000且平均评分大于4.0的用户,返回他们的名字,平 均评分,以及按json数组形式表示的funny, useful, cool,三者的和,限制10条;尝试 按平均评分降序排序,并使用explain查看排序的开销,与第一题的排序情况做对比, 主要关注"rows\_examined\_per\_scan"和"cost\_info"

每次会话调整设置：SET sort\_buffer\_size = 10000000;

SELECT

JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.name')) AS name,

JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.average\_stars')) AS avg\_stars,

JSON\_ARRAY(

JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.funny')),

JSON\_ARRAY(

CAST(JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.funny')) AS SIGNED) + CAST(JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.useful')) AS SIGNED) + CAST(JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.cool')) AS SIGNED)

) AS '[funny, useful, cool, sum]'

FROM

user

JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.funny')),

JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.useful')),

JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.cool')),

CAST(JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.funny')) AS SIGNED) + CAST(JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.useful')) AS SIGNED) + CAST(JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.cool')) AS SIGNED)

) AS '[funny, useful, cool, sum]'

FROM

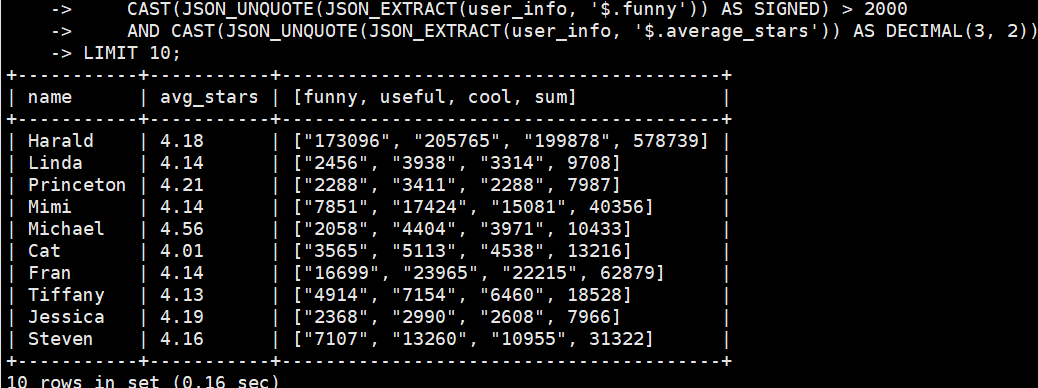
user

WHERE

CAST(JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.funny')) AS SIGNED) > 2000

AND CAST(JSON\_UNQUOTE(JSON\_EXTRACT(user\_info, '$.average\_stars')) AS DECIMAL(3, 2)) > 4.0

LIMIT 10;



12.

SELECT  
  json\_pretty(json\_object(  
    ’business\_with\_most\_tips’,  
    json\_object(‘business\_id’, t1.business\_id, ’tips\_count’, t1.max\_tips\_count),  
    ’business\_info’,   
    t1.business\_info,  
    ’user\_with\_most\_tips’,  
    json\_object(‘user\_id’, t2.user\_id, ’tips\_count’, t2.max\_tips\_count),  
    ’user\_info’,   
    t2.user\_info SELECT

json\_pretty(json\_object(

'business\_with\_most\_tips',

json\_object('business\_id', t1.business\_id, 'tips\_count', t1.max\_tips\_count),

'business\_info',

t1.business\_info,

'user\_with\_most\_tips',

json\_object('user\_id', t2.user\_id, 'tips\_count', t2.max\_tips\_count),

'user\_info',

t2.user\_info

)) AS merged\_info

FROM (

SELECT t.business\_id, COUNT(\*) AS max\_tips\_count, b.business\_info

FROM tip t

JOIN business b ON t.business\_id = b.business\_id

GROUP BY t.business\_id

ORDER BY max\_tips\_count DESC

LIMIT 1

) AS t1

JOIN (

SELECT t.user\_id, COUNT(\*) AS max\_tips\_count, u.user\_info

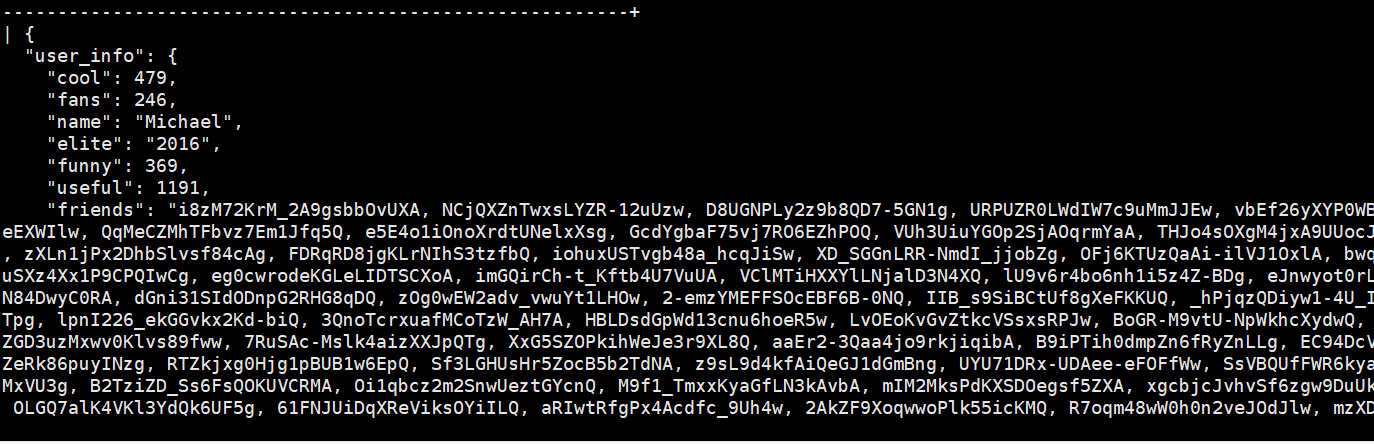
FROM tip t

JOIN user u ON t.user\_id = u.user\_id

GROUP BY t.user\_id

ORDER BY max\_tips\_count DESC

LIMIT 1

) AS t2;

13.

WITH BusinessInfo AS (

SELECT

JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.name')) AS business\_name,

CAST(JSON\_UNQUOTE(JSON\_EXTRACT(business\_info, '$.review\_count')) AS SIGNED) AS business\_review\_count,

IF(JSON\_UNQUOTE(JSON\_EXTRACT(business\_info->'$.hours', '$.Tuesday')), 1, 0) AS business\_open\_on\_Tuesday,

JSON\_EXTRACT(business\_info, '$.hours') AS business\_hours

FROM business

ORDER BY business\_review\_count DESC

LIMIT 3

)

SELECT

business\_name,

business\_review\_count,

business\_open\_on\_Tuesday,

num,

hours\_in\_a\_week

FROM (

SELECT

business\_name,

business\_review\_count,

business\_open\_on\_Tuesday,

num,

hours\_in\_a\_week

FROM BusinessInfo

CROSS JOIN JSON\_TABLE(

business\_hours,

'$.\*' COLUMNS (

num FOR ORDINALITY,

hours\_in\_a\_week VARCHAR(100) PATH '$'

)

) AS hours

) AS SubQuery

ORDER BY business\_name ASC;