Fall 2022 MAT265 New SQL Commands

- Shaokang Li

Introduction:

For this week's assignment, I try to experiment with new SQL Commands and Functions. Through such practices, I furthured my knowledge in SQL programming and might also discover new pattern with these tools.

Query 01:

For the first query, I tried to do the same thing as last week, calculate the average borrow time of a single item, but use MySQL queries only.

NOTE:

Below query only works in MySQL Version 8.0 or later, since it uses the `last` keyword in JSON array index.

For MySQL version 5.7, we can only use specific number to index.

USAGE:

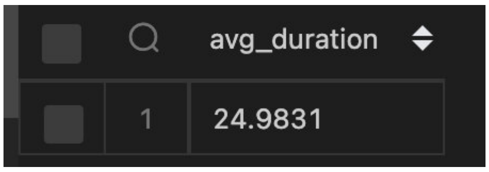
`JSON_ARRAYAGG` aggregates an column to a JSON list.

`JSON_UNQUOTE` cast a JSON-type data into its original type.

```sql
select
    AVG((DATEDIFF(lastday, firstday))) as avg_duration
from(
        select
            JSON_UNQUOTE(`dates` -> "$[0]") as firstday,
            JSON_UNQUOTE(`dates` -> "$[last]") as lastday # modify 'last' to 1 to calculate the time difference between first 2 checkouts.
        from (
                select
                    JSON_ARRAYAGG(cout) as dates
                from
                    spl_2016.inraw
                where
                    bibNumber = 2851592
                group by
                    itemNumber
            ) as res
) as dates;
```

RESULT:

Since the query won't work on a MySQL 5.7 version server, I modified `JSON_UNQUOTE('dates' -> "$[last]") as lastday` to `JSON_UNQUOTE('dates' -> "$[1]") as lastday` . I get:


```
avg_duration
1   24.9831
```

This result means the average duration between first two checkouts of the album is 25 days.

Query 02:

For the second query, I'm interested in finding CD's with a panlidrome title.

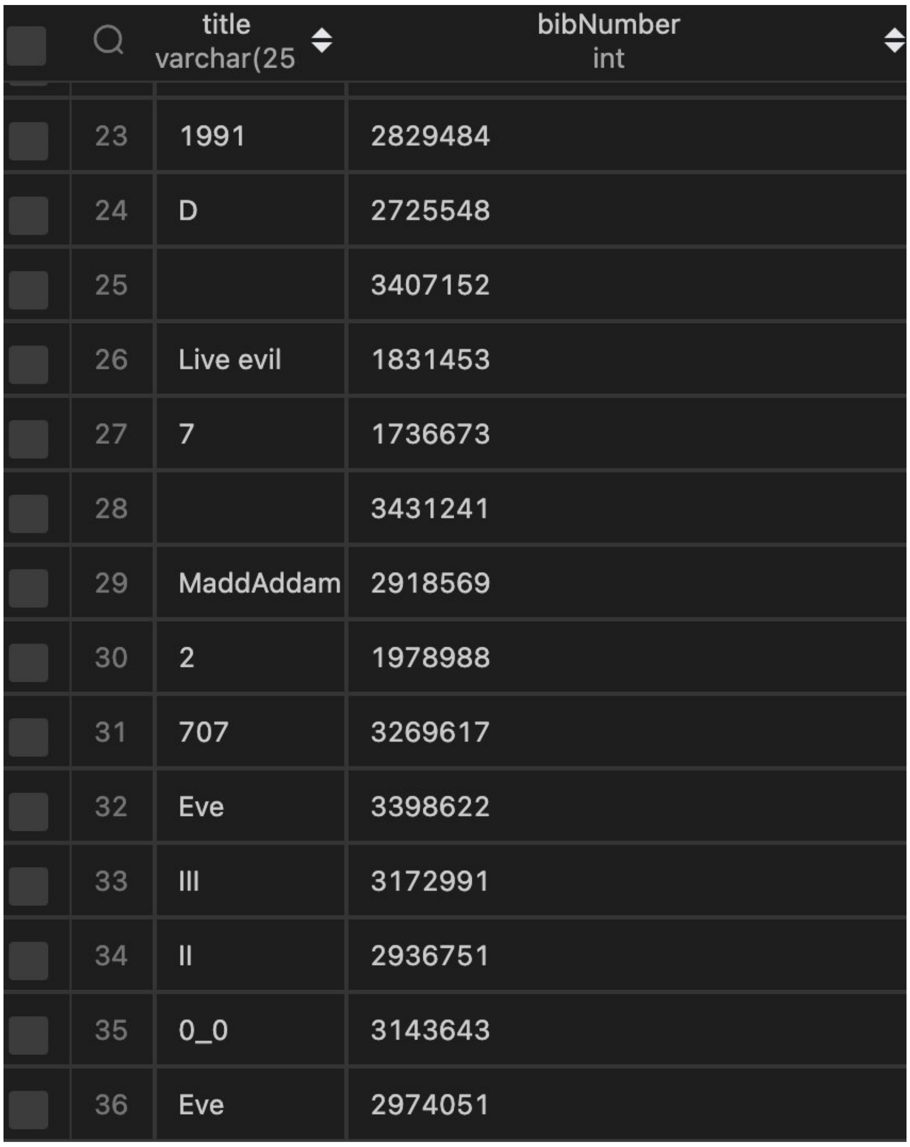
USAGE:

`REVERSE` a string function takes a string as input, outputs the reverse of input.

```sql
select
    distinct title,
    bibNumber
from spl_2016.inraw
where
    itemtype in (
        'arcd',
        'nacd',
        'jrcd',
        'accd',
        'cacd',
        'cccd',
        'jccd',
        'nccd'
    )
    and cout > '2019-01-01'
    and title = REVERSE(title)
```

RESULT:

Here are some of the interesting palinrome titles of album.



	title varchar(25	bibNumber int
23	1991	2829484
24	D	2725548
25		3407152
26	Live evil	1831453
27	7	1736673
28		3431241
29	MaddAddam	2918569
30	2	1978988
31	707	3269617
32	Eve	3398622
33	III	3172991
34	II	2936751
35	0_0	3143643
36	Eve	2974051

Query 03:

Find out how titles that “SOUNDS LIKE” numeric numbers (1~9), but doesn’t contain number in it, and only contains 1 word.

USAGE:

`sounds like`: return items that has similar ‘soundex’.

`NOT REGEXP`: return the opposite of an regular expression

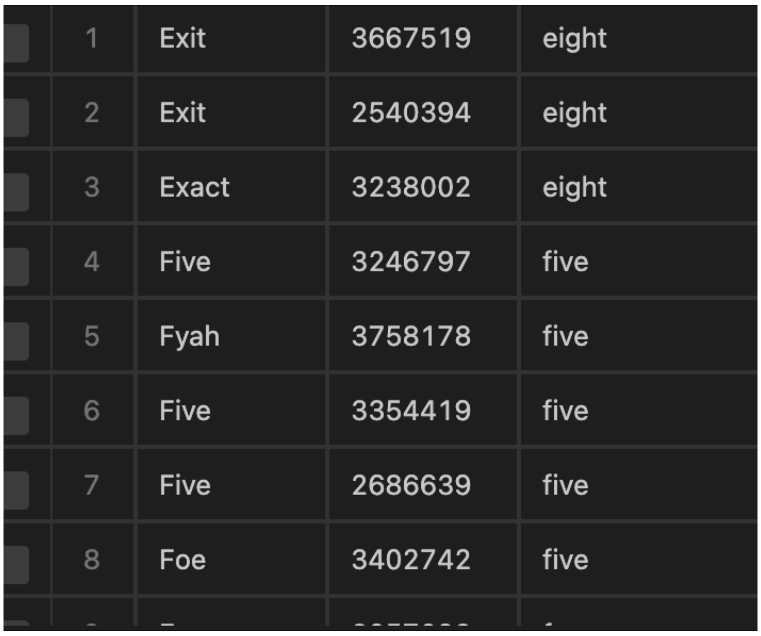
`LENGTH`: get the length of a string

`REPLACE`: replace a certain character in a string.

```sql
select *
from (
        select
            distinct title,
            bibNumber,
            case
                when title sounds like 'one' then 'one'
                when title sounds like 'two' then 'two'
                when title sounds like 'three' then 'three'
                when title sounds like 'four' then 'four'
                when title sounds like 'five' then 'five'
                when title sounds like 'six' then 'six'
                when title sounds like 'seven' then 'seven'
                when title sounds like 'eight' then 'eight'
                when title sounds like 'nine' then 'nine'
                when title sounds like 'ten' then 'ten'
                else 'other'
            end as numeric_name
        from spl_2016.inraw
        where
            itemtype in (
                'arcd',
                'nacd',
                'jrcd',
                'accd',
                'cacd',
                'cccd',
                'jccd',
                'nccd'
            )
            and cout > '2022-01-01'
            and title NOT REGEXP '[0-9]'
            and (#word count function, select strings that #words = 1
                LENGTH(title) - LENGTH(
                    REPLACE (title, ' ', '')
                ) + 1
            ) = 1
    ) as final
where
    final.numeric_name != 'other';
```

RESULT:

It's interesting to see how the phonetics are 'sounds like'. And there are around 89 titles that sounds like nuumeric numbers in CD category.



1	Exit	3667519	eight
2	Exit	2540394	eight
3	Exact	3238002	eight
4	Five	3246797	five
5	Fyah	3758178	five
6	Five	3354419	five
7	Five	2686639	five
8	Foe	3402742	five

Query 04:

For the last query, I want to try out the bit operation queries. One application of bitwise operations is to check if an item appears odd number of times or even number of times.

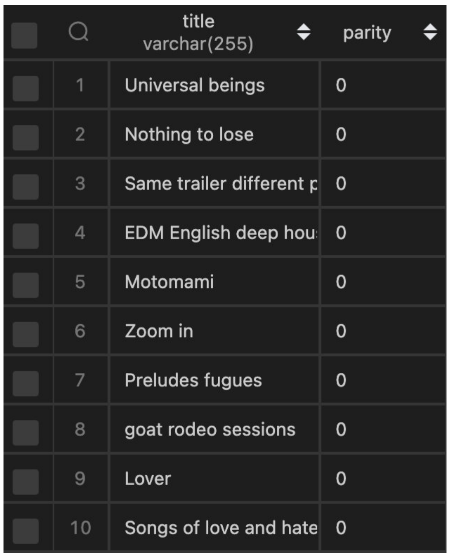
USAGE:

`BIT_XOR`: BIT XOR operations can be used to check if a number appear's odd number of times or even number of times. This is because $a_{XOR(a)} = 0$ and $0_{XOR(a)} = a$.

```sql
select *
from (
        select
            temp.title,
            BIT_XOR(temp.bibNumber) as parity
        from(
                select *
                from
                    spl_2016.inraw
                where
                    itemtype in (
                        'arcd',
                        'nacd',
                        'jrcd',
                        'accd',
                        'cacd',
                        'cccd',
                        'jccd',
                        'nccd'
                    )
                    and cout > '2022-09-01'
            ) as temp
        group by
            title
    ) as final
where parity = 0;
```

RESULT:

We have 911 different titles that appears even number of time. Another way to check if an item appears odd number of times or even number of times is to count times and then use `MOD` function.



	title varchar(255)	parity
1	Universal beings	0
2	Nothing to lose	0
3	Same trailer different p	0
4	EDM English deep hou:	0
5	Motomami	0
6	Zoom in	0
7	Preludes fugues	0
8	goat rodeo sessions	0
9	Lover	0
10	Songs of love and hate	0

Conclusion:

It's interesting to play with some of these new queries. And I can explore some patterns about the items' names, especially the palindrome and "Sounds like" part. This will give me new insights in finding interesting item titles.