

# SQL Optimization



BLACK N WHITE



#1

Use 'regexp\_like' to replace 'LIKE' clauses

```
SELECT*
FROM
  Table1
WHERE
  lower(item_name) Like '%hp%' OR
  lower(item_name) Like '%apple%' OR
  lower(item_name) Like '%dell%' OR
  lower(item_name) Like '%asus%' OR
  ----- and on
```



```
SELECT*
FROM
  table1
WHERE
  REGEXP_LIKE(lower(item_name), 'hp' | apple | dell | asus ')
```

#2

Use 'regexp\_extract' to replace 'Case-when LIKE'

```
SELECT*
CASE
WHEN concat('',item_name,' ') LIKE '%lms%' then 'Lms'
WHEN concat('',item_name,' ') LIKE '%manipal%' then 'Manipal'
WHEN concat('',item_name,' ') LIKE '%epic.u%' then 'Epic.u'
-
As brand
FROM item_list
```



```
SELECT*
regexp_extract(item_name,'(Dmbs| Java | Php | Sql | Ph sql | C language)')
As brand
FROM item_list
```



#3

Convert long list of IN clause into a temporary table

```
SELECT*
FROM Table1 as t1
WHERE
  itemid in(2114103301, 2114103302, 2114103303)
```



```
SELECT*
FROM Table1 as t1
JOIN(
  SELECT
    Itemid
  FROM( SELECT
    split('2114103301, 2114103302, 2114103303,--,',' ','')
    as bar
  )
  CROSS JOIN
    UNNEST(bar) As t(itemid)
) As Table2 as t2
ON
  t1.itemid =t2.itemid
```



#4

Always order your JOINS from largest table to smallest tables

SELECT

\*

FROM

small\_table

JOIN

large\_table

ON small\_table.id = large\_table.id



SELECT

\*

FROM

large\_table

JOIN

small\_table

ON small\_table.id = large\_table.id

WHITE  
Lead Tomorrow

#5

## Use simple ewui-joins

Two tables with date string e.g., '2023-02-06', but one the tables only has columns for year, month, day values

```
SELECT*
```

```
FROM
```

```
table a
```

```
JOIN
```

```
table2 b
```

```
ON a.date=CONCAT(b.year, '-', b.month, '-', b.day)
```

BLACK  
Learn Today



```
SELECT*
```

```
FROM
```

```
table a
```

```
JOIN
```

```
table2 b
```

```
ON a.date=CONCAT(b.year, '-', b.month, '-', b.day)
```


WHITE  
Lead Tomorrow



#6

Always "GROUP BY" by the attribute/column with the largest number of unique/values

```
select
  main_category,
  sub_category,
  itemid,
  sum(price)
form
  table1
group by
  main_category, sub_category, itemid
```



```
select
  main_category,
  sub_category,
  itemid,
  sum(price)
form
  table1
group by
  itemid, sub_category ,main_category
```

#7

## Avoid sub-queries in WHERE clause

```
select
sum(price)
from
table1
where
itemid in(
select itemid
from table2
)
```



```
With t2 as(
Select itemid
From table2
) select
sum(price)
from
table as t1
join
t2
on t1.itemid =t2.itemid
```





#8

## Use Max instead of Rank

```
SELECT*  
from (  
  select  
  userid,  
  rank()over( order by prdate desc ) as rank  
  from table1  
where ranking = 1  
)
```



BLACK  
Learn Today

WHITE  
Lead Tomorrow

```
SELECT userid , max (prdate)
```

```
From table1
```

```
group by 1
```



# Extra Tips



- Use `apporx_distinct()` instead of `count (distinct)` for very lary data-sets .
- Use `apporx_percentile(metric,0.5)` for median
- Avoid UNIONS where possible
- Use WITH statements VS. nested sub-queries

# THANKS

**BLACK**  
Learn Today

**N**  
**WHITE**  
Lead Tomorrow



*Our journey is endless*