**Generating Query Pattern:**

**/\*Q1. For one Application, get the list of the warehouse from the APP Mapping table\*/**

Collect all warehouse name for your application

**/\*Q2. Load the last 10days of data into a local table for one warehouse\*/**

create or replace table demo\_db.public.query\_trend DATA\_RETENTION\_TIME\_IN\_DAYS =0 as

select to\_date(start\_time)as start\_dt, total\_elapsed\_time,CLUSTER\_NUMBER,PERCENTAGE\_SCANNED\_FROM\_CACHE,QUERY\_TYPE,QUERY\_LOAD\_PERCENT

From

"SNOWFLAKE\_METADATA"."CUSTOM\_VIEWS.<FunctionShortname>\_<AppShortname>\_QUERY\_HISTORY\_V

where to\_date(start\_time)>=current\_date-10 and to\_date(start\_time)<current\_date

and warehouse\_name=<Warehosue Name>

and warehouse\_size is not null ;

**/\*Q3. Below query results date wise count of queries running between**

**"Running in less than 1sec",**

**"Running between 1sec and 1min",**

**"Running between 1min and 3min",**

**"Running between 3min and 10min",**

**"Running greater than 10min".\*/**

select start\_dt,

SUM(CASE WHEN total\_elapsed\_time\_sec<=1 THEN 1 ELSE 0 END) AS "Running in less than 1sec",

SUM(CASE WHEN total\_elapsed\_time\_sec >1 and total\_elapsed\_time\_sec<=60 THEN 1 ELSE 0 END) AS "Running between 1sec and 1min",

SUM(CASE WHEN total\_elapsed\_time\_sec >60 and total\_elapsed\_time\_sec<=180 THEN 1 ELSE 0 END) AS "Running between 1min and 3min",

SUM(CASE WHEN total\_elapsed\_time\_sec >180 and total\_elapsed\_time\_sec<=600 THEN 1 ELSE 0 END) AS "Running between 3min and 10min",

SUM(CASE WHEN total\_elapsed\_time\_sec >600 THEN 1 ELSE 0 END) AS "Running greater than 10min",

count(1) as tot\_query\_cnt

FROM (select start\_dt,total\_elapsed\_time/1000 as total\_elapsed\_time\_sec from demo\_db.public.query\_trend)a

GROUP BY 1 ORDER BY 1 ASC;

**/\*Q4. Below query results date wise % of queries running between**

**"Running in less than 1sec",**

**"Running between 1sec and 1min",**

**"Running between 1min and 3min",**

**"Running between 3min and 10min",**

**"Running greater than 10min". \*/**

select start\_dt,

count(1) as "Total query count",

(SUM(CASE WHEN total\_elapsed\_time\_sec<=1 THEN 1 ELSE 0 END)/"Total query count")\*100 AS "Running in less than 1sec",

(SUM(CASE WHEN total\_elapsed\_time\_sec >1 and total\_elapsed\_time\_sec<=60 THEN 1 ELSE 0 END)/"Total query count")\*100 AS "Running between 1sec and 1min",

(SUM(CASE WHEN total\_elapsed\_time\_sec >60 and total\_elapsed\_time\_sec<=180 THEN 1 ELSE 0 END)/"Total query count")\*100 AS "Running between 1min and 3min",

(SUM(CASE WHEN total\_elapsed\_time\_sec >180 and total\_elapsed\_time\_sec<=600 THEN 1 ELSE 0 END)/"Total query count")\*100 AS "Running between 3min and 10min",

(SUM(CASE WHEN total\_elapsed\_time\_sec >600 THEN 1 ELSE 0 END)/"Total query count")\*100 AS "Running greater than 10min"

FROM (select start\_dt,total\_elapsed\_time/1000 as total\_elapsed\_time\_sec from demo\_db.public.query\_trend)a

GROUP BY 1 ORDER BY 1 ASC;

**/\*Q5. Below query results date wise number of queries utilizing warehouse cache**

**"0% Warehouse Cache",**

**"Less than 10% Warehouse Cache",**

**"Between 10% and 30% Warehouse Cache",**

**"Between 30% and 50% Warehouse Cache",**

**"Between 50% and 70% Warehouse Cache",**

**"More than 70% Warehouse Cache". \*/**

select start\_dt,

SUM(CASE WHEN PERCENTAGE\_SCANNED\_FROM\_CACHE=0 THEN 1 ELSE 0 END) AS "0% Warehosue Cache",

SUM(CASE WHEN PERCENTAGE\_SCANNED\_FROM\_CACHE>0 AND PERCENTAGE\_SCANNED\_FROM\_CACHE<10 THEN 1 ELSE 0 END) AS "Less than 10% Warehosue Cache",

SUM(CASE WHEN PERCENTAGE\_SCANNED\_FROM\_CACHE>=10 AND PERCENTAGE\_SCANNED\_FROM\_CACHE<30 THEN 1 ELSE 0 END) AS "Between 10% and 30% Warehosue Cache",

SUM(CASE WHEN PERCENTAGE\_SCANNED\_FROM\_CACHE>=30 AND PERCENTAGE\_SCANNED\_FROM\_CACHE<50 THEN 1 ELSE 0 END) AS "Between 30% and 50% Warehosue Cache",

SUM(CASE WHEN PERCENTAGE\_SCANNED\_FROM\_CACHE>=50 AND PERCENTAGE\_SCANNED\_FROM\_CACHE<70 THEN 1 ELSE 0 END) AS "Between 50% and 70% Warehosue Cache",

SUM(CASE WHEN PERCENTAGE\_SCANNED\_FROM\_CACHE>=70 THEN 1 ELSE 0 END) AS "More than 70% Warehosue Cache"

FROM demo\_db.public.query\_trend

GROUP BY 1 ORDER BY 1

**/\* Q6. For each QUERY\_TYPE below query results date wise average runtime and query count. \*/**

SELECT start\_dt,QUERY\_TYPE,AVG(total\_elapsed\_time)/1000 as total\_elapsed\_in\_Sec,COUNT(1) FROM

demo\_db.public.query\_trend

GROUP BY 1,2 ORDER BY 1,2

**/\* Q7. For each QUERY\_TYPE below query results, average runtime, and query count. \*/**

SELECT QUERY\_TYPE,AVG(total\_elapsed\_time)/1000,COUNT(1) FROM

demo\_db.public.query\_trend

GROUP BY 1 ORDER BY 1

**/\* Q8. Below query results date wise average load on the warehouse. \*/**

SELECT start\_dt,AVG(QUERY\_LOAD\_PERCENT)

FROM demo\_db.public.query\_trend

GROUP BY 1 ORDER BY 1

**/\* Q9. The below query results in a date wise number of queries running on each cluster. \*/**

SELECT start\_dt,CLUSTER\_NUMBER,COUNT(1)

FROM demo\_db.public.query\_trend

GROUP BY 1,2 ORDER BY 1,2;

Repeat process (2 to 9) for all the warehouse.

**Identifying long running /recurring queries:**

**/\*Q1. For one Application, get the list of the warehouse from the APP Mapping table\*/**

Collect all warehouse name for your application

**/\*Q2. Load the last 15days of data into a local table for one warehouse\*/**

CREATE OR REPLACE table demo\_db.public.WH\_LONG\_QUERY1 DATA\_RETENTION\_TIME\_IN\_DAYS =0 AS

select warehouse\_name, warehouse\_size, database\_name, schema\_name, query\_id, query\_text, QUERY\_TYPE, user\_name, total\_elapsed\_time/(60000) as total\_elapsed\_time\_min ,

start\_time, end\_time, ROW\_NUMBER() over (PARTITION BY warehouse\_name, warehouse\_size, query\_text ORDER BY start\_time DESC ) row\_nr

from "SNOWFLAKE\_METADATA"."CUSTOM\_VIEWS.<FunctionShortName>\_<AppShortName>\_QUERY\_HISTORY\_V

where to\_date(start\_time)>= current\_date-15 and to\_date(start\_time)<current\_date and warehouse\_size is not null and warehouse\_name =’<Warehouse name>’

order by total\_elapsed\_time desc;

create or replace table demo\_db.public.Query\_WH\_15\_days DATA\_RETENTION\_TIME\_IN\_DAYS =0 as

select a.warehouse\_name, a.warehouse\_size, a.query\_text, a.cnt, a.AVG\_TIME as Avg\_Time\_Min, b.query\_id As Last\_Query\_Id, start\_time as Last\_Start\_Tm, end\_time As Last\_End\_Tm

from

(

SELECT warehouse\_name,warehouse\_size,query\_text,COUNT(1) cnt,AVG(total\_elapsed\_time\_min) AVG\_TIME

FROM demo\_db.public.WH\_LONG\_QUERY1

GROUP BY warehouse\_name,warehouse\_size,query\_text

)a

inner join

(select distinct warehouse\_name,warehouse\_size,query\_text,query\_id,start\_time,end\_time

from demo\_db.public.WH\_LONG\_QUERY1 where row\_nr=1 )b

on (a.warehouse\_name=b.warehouse\_name and a.query\_text=b.query\_text and coalesce(a.warehouse\_size,'')=coalesce(b.warehouse\_size,''))

--where AVG\_TIME>5

order by AVG\_TIME desc, cnt desc

**/\*Q3. Top 50 queries by number of runs\*/**

select warehouse\_name, warehouse\_size, query\_text, cnt, Avg\_Time\_Min, Last\_Query\_Id, Last\_Start\_Tm, Last\_End\_Tm

from demo\_db.public.Query\_WH\_15\_days

order by cnt desc, Avg\_Time\_Min desc limit 50

**/\*Q4. Top 50 queries by duration\*/**

select warehouse\_name, warehouse\_size, query\_text, cnt, Avg\_Time\_Min, Last\_Query\_Id, Last\_Start\_Tm, Last\_End\_Tm

from demo\_db.public.Query\_WH\_15\_days

order by Avg\_Time\_Min desc, cnt desc

limit 50

Repeat process (2 to 4) for all the warehouse.