Advanced Databases
INZ000109P
Project

Group: A2
Mustafa Tayyip BAYRAM 257639
Furkan ÖCALAN 257638
Bauyrzhan Marat

- NO MEMCOMPRESS

  In-Memory data is populated without compression.

- MEMCOMPRESS FOR DML

  Level mainly intended for increasing DML performance and minimal compression.

- MEMCOMPRESS FOR QUERY LOW – the default

  Optimized for query performance (default).

- MEMCOMPRESS FOR QUERY HIGH

  Optimized for query performance and space saving.

- MEMCOMPRESS FOR CAPACITY LOW

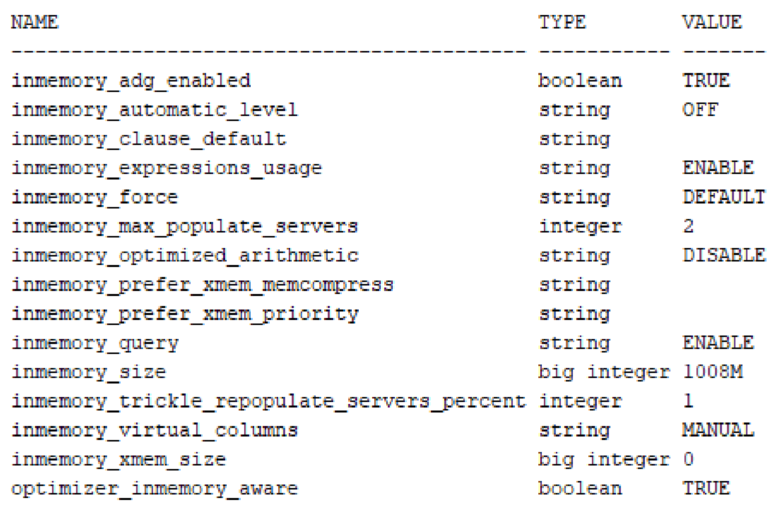
  Higher space saving level compared to Query High and Low

- MEMCOMPRESS FOR CAPACITY HIGH

  Level optimized for space saving and slightly less capacity.

➢ ALTER SYSTEM SET INMEMORY_SIZE=1008M SCOPE=SPFILE

➢ SHOW PARAMETER INMEMORY



```
NAME                                         TYPE        VALUE
-------------------------------------------- ----------- -------
inmemory_adg_enabled                         boolean     TRUE
inmemory_automatic_level                     string      OFF
inmemory_clause_default                      string
inmemory_expressions_usage                   string      ENABLE
inmemory_force                               string      DEFAULT
inmemory_max_populate_servers                integer     2
inmemory_optimized_arithmetic                string      DISABLE
inmemory_prefer_xmem_memcompress             string
inmemory_prefer_xmem_priority                string
inmemory_query                               string      ENABLE
inmemory_size                                big integer 1008M
inmemory_trickle_repopulate_servers_percent  integer     1
inmemory_virtual_columns                     string      MANUAL
inmemory_xmem_size                           big integer 0
optimizer_inmemory_aware                     boolean     TRUE
```

Index improvements and partitioning techniques is more effective way to optimizing query performance because we used mostly QUERY HIGH and CAPACITY HIGH that means we gain from storage and lost from performance .

Which tables/columns will be stored in columnar store?

- **Products Table => list_price [QUERY HIGH] and model_year [CAPACITY HIGH] columns [Effects 1st, , 2nd ,3rd , 4th queries]**
- **Order_Items Table => discount [CAPACITY HIGH] column [2nd ,3rd ,4th , 6th query]**
- **Staffs Table => salary [QUERY HIGH] column [Effects 2nd ,3rd query]**
- **Stores Table => city [QUERY HIGH] column [Effects 2nd ,4thquery]**
- **Orders Table => order_status [CAPACITY HIGH] column [Effects 2nd ,3rd ,4th,5th query]**

- FIRST QUERY

```
SELECT product_id,brand_name ,product_name, model_year, list_price, category_name FROM
(
    (SELECT * FROM
        (SELECT * FROM
          ( SELECT * from MUTABAY.products prod_outer
              where 1 = (
                   SELECT COUNT(Distinct list_price)
                   FROM MUTABAY.products prod_inner
                   WHERE prod_outer.brand_id = prod_inner.brand_id
                   AND prod_outer.list_price < prod_inner.list_price
                   )
          ) prod
          FULL OUTER JOIN
          MUTABAY.brands brands on brands.brand_id = prod.brand_id
        ) prod_brand
        FULL OUTER JOIN
        MUTABAY.categories categories on categories.category_id = prod_brand.category_id
    )prod_brand_cat
)
where list_price > 980000 AND model_year < 2020

GROUP BY product_id,brand_name ,product_name, model_year, list_price, category_name
ORDER BY product_id ASC;
```

EXECUTION TIME COMPARISON

TRYING	BEFORE COLUMNAR	AFTER COLUMNAR
1	**2.383**	**2.008**
2	**1.921**	**1.586**
3	**1.756**	**1.09**
4	**1.764**	**1.765**
5	**1.765**	**1.594**
6	**2.187**	**1.598**
7	**1.74**	**1.585**
8	**1.711**	**1.61**
9	**1.709**	**1.584**
10	**1.754**	**1.947**
MAX	**2.383**	**2.008**
MIN	**1.709**	**1.584**
AVERAGE	**1.869**	**1.6367**

WITHOUT ANY IMPROVEMENTS AVERAGE TIME = 1.869
COLUMNAR STORAGE AVG. TIME = 1.6367

- **SECOND QUERY**

```
SELECT first_name,last_name, active, salary, stores.store_name, stores.city, stores.state
FROM
(
  SELECT staffs.*,avg(salary) over (partition by store_id) as avgSalary
  from MUTABAY.staffs staffs
)staffs
FULL OUTER JOIN MUTABAY.stores stores
ON staffs.store_id=stores.store_id
FULL OUTER JOIN MUTABAY.orders orders
ON stores.store_id = orders.store_id
FULL OUTER JOIN MUTABAY.order_items order_items
ON orders.order_id = order_items.order_id
FULL OUTER JOIN MUTABAY.products products
ON order_items.product_id = products.product_id
WHERE staffs.salary < staffs.avgsalary or order_items.discount > 0.05 OR customer_id > 1500
GROUP BY store_name, first_name, salary, city, state, last_name, active
having (avg(staffs.salary) > 1000 OR state IS NOT NULL) OR (city = 'Aberdeen' AND active = 1)
ORDER BY store_name asc;
```

EXECUTION TIME COMPARISON

TRYING	BEFORE COLUMNAR	AFTER COLUMNAR
1	1.67	1.475
2	1.208	1.038
3	1.188	1.202
4	1.194	1.347
5	1.217	1.084
6	1.164	1.073
7	1.314	1.064
8	1.357	1.104
9	3.513	1.074
10	3.329	1.074
MAX	3.513	1.475
MIN	1.164	1.038
AVERAGE	1.7514	1.1535

WITHOUT ANY IMPROVEMENTS AVERAGE TIME = 1.7514
COLUMNAR STORAGE AVG. TIME = 1.1535

- **THIRD QUERY**

```
SELECT  products.product_id, products.product_name, products.list_price,
        orders.order_date, orders.required_date, orders.order_status,
        categories.category_name, brands.brand_name, quantity_id,
        discount,COUNT(quantity_id) quantity_count, (quantity_id * discount * products.list_price) total
FROM MUTABAY.order_items order_items
    full outer join MUTABAY.orders orders on
        (orders.order_id = order_items.order_id)
    full outer join MUTABAY.products products on
        (products.product_id = order_items.product_id)
    full outer join MUTABAY.brands brands on
        (brands.brand_id = products.brand_id)
    full outer join MUTABAY.categories categories on
        (categories.category_id = products.category_id)
    full outer join MUTABAY.staffs staffs on
        (staffs.staff_id = orders.staff_id)
    WHERE   (shipped_date  - order_date ) > 2  OR
        (shipped_date - order_date ) = 0   OR
        (shipped_date - order_date ) < 0
GROUP BY products.product_id, products.product_name, products.list_price,
       orders.order_date, orders.required_date, orders.order_status,
       categories.category_name, brands.brand_name, quantity_id,
       discount
having AVG(list_price) > 10000
Order by order_status;
```

EXECUTION TIME COMPARISON

TRYING	BEFORE COLUMNAR	AFTER COLUMNAR
1	**0.713**	**0.618**
2	**0.707**	**0.538**
3	**0.722**	**0.517**
4	**0.952**	**0.534**
5	**0.912**	**0.539**
6	**0.772**	**0.547**
7	**0.822**	**0.531**
8	**0.795**	**0.534**
9	**0.714**	**0.537**
10	**0.926**	**0.535**
MAX	**0.952**	**0.618**
MIN	**0.707**	**0.517**
AVERAGE	**0.7985**	**0.543**

WITHOUT ANY IMPROVEMENTS AVERAGE TIME = 0.7985
COLUMNAR STORAGE AVG. TIME = 0.543

- **FOURTH QUERY**

```
update MUTABAY.products products
set model_year =
            (
      select distinct(product_id) as total_product
      from MUTABAY.order_items order_items
        full outer join MUTABAY.orders orders on
          (orders.order_id = order_items.order_id)
        full outer join MUTABAY.stores stores on
          (orders.store_id = stores.store_id)
        where stores.store_id =
          (
          select store_id from MUTABAY.stocks
            full outer join MUTABAY.products products on
              (stocks.product_id = products.product_id)
                where ((model_year between 2020 and 1958) or (ROUND(list_price) < 990.000))
                    or UPPER ( SUBSTR(product_name,2,3 ) )LIKE 'D%'
                fetch first 1 rows only
          )
      fetch first 1 rows only
            );
```

EXECUTION TIME COMPARISON

TRYING	BEFORE COLUMNAR	AFTER COLUMNAR
1	**1.379**	**2.817**
2	**0.813**	**1.645**
3	**0.85**	**1.095**
4	**0.741**	**0.827**
5	**1.251**	**0.874**
6	**0.653**	**0.692**
7	**0.798**	**0.76**
8	**0.666**	**0.655**
9	**0.731**	**1.295**
10	**0.667**	**0.799**
MAX	**1.379**	**2.817**
MIN	**0.653**	**0.655**
AVERAGE	**0.8549**	**1.1459**

WITHOUT ANY IMPROVEMENTS AVERAGE TIME = 0.8549
COLUMNAR STORAGE AVG. TIME = 1.1459

- **FIFTH QUERY**

```
update MUTABAY.order_items set quantity_id =
(
  select quantity_id from MUTABAY.products products
  full outer join MUTABAY.order_items order_items on order_items.product_id = products.product_id
  full outer join MUTABAY.stocks stocks on stocks.product_id = products.product_id
  full outer join MUTABAY.orders orders on order_items.order_id=orders.order_id
  full outer join MUTABAY.customers customers on orders.customer_id=customers.customer_id
  full outer join MUTABAY.stores stores on orders.store_id=stores.store_id
  full outer join MUTABAY.staffs staffs on orders.staff_id=staffs.staff_id
  where products.product_id in
    (
      Select product_id from MUTABAY.order_items where order_id in
      (
        Select order_id from MUTABAY.orders
        WHERE
        (order_status = 1 AND ( shipped_date - required_date = 1 ))
        OR
        (order_status = 2 AND (shipped_date - required_date = 0 ))
      )
    )fetch next 1 rows only
);
```

EXECUTION TIME COMPARISON

TRYING	BEFORE COLUMNAR	AFTER COLUMNAR
1	**3.365**	**2.391**
2	**3.598**	**3.262**
3	**4.786**	**4.004**
4	**2.713**	**1.841**
5	**2.517**	**3.272**
6	**3.69**	**3.462**
7	**2.986**	**1.626**
8	**2.786**	**2.657**
9	**1.884**	**1.042**
10	**2.616**	**3.159**
MAX	**4.786**	**4.004**
MIN	**1.884**	**1.042**
AVERAGE	**3.0941**	**2.6716**

WITHOUT ANY IMPROVEMENTS AVERAGE TIME = 3.0941
COLUMNAR STORAGE AVG. TIME = 2.6716

- **SIXTH QUERY**

```
UPDATE MUTABAY.stores
SET MUTABAY.stores.store_name = (
  SELECT store_name FROM MUTABAY.stores stores
  INNER JOIN
  (
    SELECT order_i.staff_id, first_name, last_name, phone, email, order_i.store_id, manager_id, active,
salary ,
    order_i.item_id ,order_i.product_id ,order_i.quantity_id ,order_i.discount ,order_i.customer_id
,order_i.order_status ,
    order_i.order_date ,order_i.required_date ,order_i.shipped_date
    FROM MUTABAY.staffs staffs
    FULL OUTER JOIN
    (
      SELECT orders.order_id, item_id, product_id, quantity_id, discount, orders.customer_id,
orders.order_status,
          orders.order_date, orders.required_date, orders.shipped_date, orders.store_id, orders.staff_id
      FROM MUTABAY.order_items order_items
      FULL OUTER JOIN MUTABAY.orders orders
      ON orders.order_id = order_items.order_id
      WHERE ORDERS.ORDER_ID IN (SELECT ORDER_ID FROM MUTABAY.ORDER_ITEMS WHERE
DISCOUNT > (SELECT AVG(DISCOUNT) FROM MUTABAY.ORDER_ITEMS))
      OR
      (order_status = 2)
    ) order_i
    ON order_i.staff_id = staffs.staff_id
    WHERE (discount > 0.48 AND discount < 0.05) AND salary > 5000
    OR
    (active = 1 AND discount = 0.4)
  )order_i_staff
  ON order_i_staff.store_id = stores.store_id
  WHERE street='1 Fremont Point' or STATE IS NOT NULL
  fetch first 1 rows only
);
```

EXECUTION TIME COMPARISON

TRYING	BEFORE COLUMNAR	AFTER COLUMNAR
1	**8.448**	**7.341**
2	**6.416**	**6.312**
3	**6.016**	**5.585**
4	**5.839**	**5.961**
5	**5.698**	**6.362**
6	**6.269**	**6.576**
7	**5.893**	**5.841**
8	**5.741**	**5.623**
9	**5.745**	**5.669**
10	**5.763**	**5.699**
MAX	**8.448**	**7.341**
MIN	**5.698**	**5.585**

AVERAGE	6.1828	6.0969

WITHOUT ANY IMPROVEMENTS AVERAGE TIME = 6.1828
COLUMNAR STORAGE AVG. TIME = 6.0969