

# Performance monitoring

# **Top 5 reasons for slow queries**

# 1. Bad Query

This is the most common reason for long running query. Your query is not using any kind of index: Primary Index, Secondary Index or Partition Primary Index. The tables are getting redistributed and those big tables taking long time in it.

## 2. Delay Time

Check for Delay time in DBQL for your query. It may happen that query execution time is in seconds however it was in Delay queue for hours.

## 3. Blocking

It may happen that your query is blocked by some other query. Also watch out for TDM jobs. Such jobs generally apply Exclusive locks on the table thereby blocking all other queries requesting the same table. You can check this Teradata Viewpoint Monitor Portlet.

## 4. Server State

Generally Teradata Server status is classified as Healthy, Degraded, Critical and Down. As the number of concurrent users increase the load on server also increases. With more load on server and AMP doing more work, your query may take more time than usual time.

## 5. Skewness

Your Source table is highly skewed because the Primary Index chosen is not good. Consider changing PI for the table if skewness factor is high.

**Improving your query**



## DISTINCT

- `DISTINCT` is better for columns with a low number of rows per value:  
Number of rows < Number of AMPs
- `GROUP BY` is better for columns with a large number of rows per value:  
Number of rows > Number of AMPs

## IN

- No theoretical limit to the number of items contained within an IN clause.
- But: performance degradations if the list starts to exceed a few hundred items.
- Optimizer tries to build a spool file when a large IN list is encountered.
- When the number of values exceeds the acceptable limit, the hard-coded values can be inserted into a volatile table and the IN clause can be made to an join.

# Date Comparison

- When comparing values of date in a particular range, the query may result in product join. This can be avoided with the usage of `SYS_CALENDAR.CALENDAR` , which is Teradata's in-built database.

# Example

```
select
t2.a1, t2.a2, t2.a3, t2.a4
from
table_2 t2
join table_3 t3
on t2.a1=t3.a1
and t2.a5_dt>=t3.a4_dt
and t2.a5_dt<=t3.a5_dt;
```

```
select
t2.a1, t2.a2, t2.a3, t2.a4
from table_2 t2
join SYS_CALENDAR.CALENDAR sys_cal
on sys_cal.calendar_date = t2.a5_dt
join table_3 t3
on t2.a1=t3.a1
and sys_cal.calendar_date >=t3.a4_dt
and sys_cal.calendar_date <=t3.a5_dt;
```

# SQL concatenation

- **Example:** Find the employee with the highest salary in each department.

```
SELECT Dept_No ,Salary ,LName ,Fname
FROM Employee_Table
WHERE Dept_No, Salary IN
    (
        SELECT Dept_No, MAX(Salary)
        FROM Employee_Table
        GROUP BY Dept_No)
ORDER BY Dept_No;
```

```
SELECT Dept_No, MAX(Salary|| ' ' || LName || ', ' || Fname
FROM Employee_Table
GROUP BY Dept_No ORDER BY Dept_No ;
```

- If two or more employees have the same max salary, this query selects only one employee per department.

**Identify suspect queries**

## Beyond **EXPLAIN**

- **Product Join Indicator:** the ratio of CPU Seconds to IO for a query.  $(\text{AMPCPUTime} * 1000) / \text{TotalIOCount}$ 
  - $\geq 3$ : the query should be reviewed.
  - $\geq 6$ : potentially an unnecessary product join.

## Beyond **EXPLAIN** (cont.)

- **Unnecessary IO Indicator:** is the ratio of IO to CPU Seconds,  
$$\text{TotalIOCount} / (\text{AMPCPUTime} * 1000)$$
  - $\geq 3$ : the query should be reviewed to eliminate full-table scans and possibly redistribution steps.
  - UIOI is a reasonable indicator to identify queries that may benefit from additional statistics to indexing improvements.
- Requires `dbqlogtbl` & `dbqlsqltbl` tables in `dbc`.



**Demo**

## Beyond **EXPLAIN** (cont.)

- PJI is the measure of how CPU intensive your query is.
  - If PJI is relatively high for a query, then the query takes many CPU cycles for the given number of I/Os.
  - This value is high during a product join, but not only.

## Beyond **EXPLAIN** (cont.)

- If UII is relatively high for a query, then it could mean that many I/O blocks are read, meaning how IO intensive your query is, but a relatively low number of rows is actually processed.
- If it is a full table scan with only a few rows qualifying, then an index could reduce the I/O consumption in this case.
- Both metrics are available in Viewpoint's Query Monitor portlet, and every individual query has these values displayed when you click on the session ID.