

Crystal Reports User Group – October 20, 2006

Strategies to Improve Report Speed:

- 1- Pass all elements of the record selection criteria to the **SQL** statement.
 - a- Do not use (most) formulas for record selection
 - b- Follow guidelines in the following FAQ:
<http://www.tek-tips.com/faqs.cfm?fid=3825>
 - c- Replace if-thens with ands-ors
 - d- Enclose specific clauses within parens.
- 2- Link tables by selecting a lead table that has one indexed record to link to tables containing multiple corresponding records.
3. Link tables on indexed fields whenever possible.
- 4- Select on indexed fields when possible.
- 5- Remove unused tables, unused formulas, unused running totals from the report.
- 6- *Avoid nested formulas.***
- 7- Use conditional formulas instead of running totals when possible.
- 8- Use conditional formulas to return a desired field result or not, instead of using suppression to eliminate unwanted records.
- 9- If using running totals, evaluate the minimum number of times, i.e., on change of group instead of for every record, if this is an option.
- 10- Avoid subreports (in most cases), although they may result in a faster report if they eliminate significant row inflation.
- 11- If using subreports, on-demand reports will be faster, since they won't fire unless clicked on. For regular subreports, avoid placement in a detail section—the subreport will execute once per section.
- 12- Using a command as the datasource will be faster than using CR to link tables, select records, etc.
- 13- If using a command as datasource, develop parameters within the command.
- 14- Whenever possible, limit records through selection, not suppression.
- 15- Use SQL expressions to convert fields to be used in record selection instead of using formula functions.
- 16- Perform grouping on server (in some cases, it's faster NOT to, though)

- 17- ***Group on indexed fields when possible, especially if performing grouping on server.***
- 18- ***Group on database fields instead of formulas when possible, especially if performing grouping on server.***
- 19- Suppress unnecessary sections; suppress or hide details to facilitate grouping on the server. ***Remove unnecessary fields from suppressed sections (can remove fields after summaries are inserted).***
- 20- Set parameters to one value versus multiple values whenever possible, e.g., use {CS_CAREER_CENTERS.CAREER_CENTER_ID} instead of choosing multiple local offices for one career center, or use {CS_LOCAL_OFFICES.ROLLUP_LWIB_AREA} instead of multiple local offices or multiple career centers.
- 21- Reorganizing the where clause can sometimes improve performance.
- 22- ***Link/filter tables in Access and then use Access as your datasource.***
- 23- Use pagenumber instead of page N of M (unless you want to force the report to reach the last page in order to process subreports).
- 24- Select distinct slows reports. In commands, using a "union" instead of a "union all" slows reports, since a union selects distinct records.

Note: Not sure about ones in bold italics.

Here are some more tips, maybe more relevant to writing commands (Excerpted from http://www.dba-oracle.com/art_sql_tune.htm, and article about writing SQL queries for Oracle databases)

Tips for more efficient SQL

Space doesn't permit me to discuss every detail of Oracle tuning, but I can share some general rules for writing efficient SQL in Oracle regardless of the optimizer that is chosen. These rules may seem simplistic but following them in a diligent manner will generally relieve more than half of the SQL tuning problems that are experienced:

- Never do a calculation on an indexed column (e.g., WHERE salary*5 > myvalue).
- Whenever possible, use the UNION statement instead of OR conditions.
- Avoid the use of NOT IN or HAVING in the WHERE clause. Instead, use the NOT EXISTS clause.
- Always specify numeric values in numeric form and character values in character form (e.g., WHERE emp_number = 565, WHERE emp_name = 'Jones').
- Avoid specifying NULL in an indexed column.
- Avoid the LIKE parameter if = will suffice. Using any Oracle function will invalidate the index, causing a full-table scan.
- Never mix data types in Oracle queries, as it will invalidate the index. If the column is numeric, remember not to use quotes (e.g., salary = 50000). For char index columns, always use single quotes (e.g., name = 'BURLESON').
- Remember that Oracle's rule-based optimizer looks at the order of table names in the FROM clause to determine the driving table. Always make sure that the last table specified in the FROM clause is the table that will return the smallest number of rows. In other words, specify multiple tables with the largest result set table specified first in the FROM clause.
- Avoid using subqueries when a JOIN will do the job.
- Use the Oracle "decode" function to minimize the number of times a table has to be selected.
- To turn off an index you do not want to use (only with a cost-based optimizer), concatenate a null string to the index column name (e.g., name||') or add zero to a numeric column name (e.g., salary+0). With the rule-based optimizer, this allows you to manually choose the most selective index to service your query.
- If your query will return more than 20 percent of the rows in the table, a full-table scan may be better than an index scan.
- Always use table aliases when referencing columns.