*** (10.2.0.4) Create a table based on dba_tables ordered by table_name so that all rows are tightly clustered on table_name

```
SQL> CREATE TABLE cf_test AS SELECT * FROM dba_tables ORDER BY table_name;

Table created.
```

*** Now create a concatenated index with the well clsutering table_name column as the leading column

```
SQL> CREATE INDEX cf_test_good_i ON cf_test(table_name, num_rows);

Index created.

SQL> EXEC dbms_stats.gather_table_stats(ownname=>'BOWIE',
tabname=>'CF_TEST, estimate_percent=> null, cascade=> true,
method_opt=>'FOR ALL COLUMNS SIZE 1');

PL/SQL procedure successfully completed.

SQL> SELECT t.table_name, i.index_name, t.blocks, t.num_rows,
i.clustering_factor
     FROM user_tables t, user_indexes i WHERE t.table_name = i.table_name
AND i.index_name='CF_TEST_GOOD_I';


TABLE_NAME INDEX_NAME          BLOCKS   NUM_ROWS CLUSTERING_FACTOR
---------- --------------- ---------- ---------- -----------------
CF_TEST    CF_TEST_GOOD_I          49       1644                49
```

*** The CF is perfect as expected, as all rows in the table are ordered in exactly the same manner as all the index entries

*** However, create the "same" index but with the column order reversed ...

```
SQL> CREATE INDEX cf_test_bad_i ON cf_test(num_rows, table_name);

Index created.

SQL> SELECT t.table_name, i.index_name, t.blocks, t.num_rows,
i.clustering_factor
     FROM user_tables t, user_indexes i WHERE t.table_name = i.table_name
AND i.index_name='CF_TEST_BAD_I';

TABLE_NAME INDEX_NAME          BLOCKS   NUM_ROWS CLUSTERING_FACTOR
---------- --------------- ---------- ---------- -----------------
CF_TEST    CF_TEST_BAD_I           49       1644               543
```

*** And the CF is over 10 times worse than the other index as the index entries are no longer ordered in a similar manner as the table