Adjoined Dimension Column Clustering (ADC Clustering) to improve Data Warehouse Query Performance

Xuedong Chen Patrick O'Neil Elizabeth O'Neil
Computer Science Department, University of Massachusetts Boston
{xuedchen/eoneil/poneil}@cs.umb.edu

1. Introduction

Data warehouses are typically made up of multiple star schemas, called data marts. A star schema example from [6] is shown in Figure 1, with a central Fact table and four dimension tables. Queries on star schemas usually restrict columns in the dimension tables to retrieve rows from the central fact table.

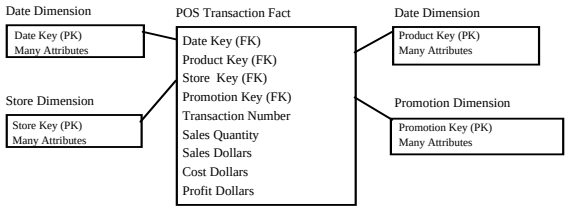

Figure 1. Example of a Star Schema

Star Schemas were first devised by Red Brick Systems [11] to speed up queries on data loaded from operational databases at intervals of about a day, with no updates between loads. Variant designs (allowing some updates) were adopted by Oracle [8,12], DB2 [1,7], Sybase IQ [14] and recently Vertica [13].

Dimension tables have relatively small numbers of rows compared to the fact table. The POS Transaction fact table of Figure 1 has nine columns, a total of 40 bytes, and disks on an inexpensive PC can contain a fact table of a few hundred gigabytes, or several billion rows, while the largest dimension table is usually Products, with up to a few million rows. (There are only a few thousand rows in Dates, Stores, etc.). As a result, practitioners commonly place a large number of descriptive columns in dimensions; most queries restrict these dimension columns, and combined joins from dimensions determine fact table rows retrieved.

In the late 1980s, a query filter factor of about 1/300, selecting rows on about one disk page out of ten, led to a DB2 query plan that retrieved just the selected subset of disk pages; if more than one disk page in ten was selected, a sequential scan that retrieved all table pages gave better performance. Since that time, disk technology has sped up sequential scans about 35 times more than it has sped access to selected disk pages. A query filter factor of about 1/10,000 or less is now required to retrieve just the selected pages, so most Data Warehouse queries scan across hundreds of GB of the fact table. To improve performance, some form of clustering must limit query retrieval range on most queries.

A form of clustering by several attributes at once was introduced in 2003 with IBM's Multi-dimensional Clustering (MDC) technology, explained in Section 2.2. MDC provides a powerful new tool, but does not directly solve the star schema access problem because the attributes typically used in queries on star schemas lie in dimension tables, not in the fact table as assumed by MDC. The fact table normally has just the foreign keys to the dimension tables, typically meaningless surrogate keys.

1.1 Contribution of this Paper

1. We introduce *ADC clustering* to cluster the fact table by commonly queried dimension columns. This accelerates star schema data warehouse queries that have range predicates on these column hierarchies.

2. We show how ADC clustering applies to database products with either good concatenated column indexing or native clustering support on multiple dimensions, such as DB2's MDC or Oracle's Partitioning.

3. We introduce the Star Schema benchmark (SSB) derived from the TPC-H benchmark [15], and demonstrate improved performance of three commercial Windows database products using ADC clustering.

2. Introducing ADC Clustering

Single column clustering works well when there is one standout column on which to sort the data that will speed up most queries of interest. But what if there is not? Figure 2 displays the schema for the Star Schema Benchmark (SSB); SSB is derived from the TPC-H benchmark in a natural way as described in the extended ADC Clustering paper [2].

The SSB Star Schema has dimensions customer, supplier, part and date. Queries of SSB restrict ranges on one to four hierarchies within these dimensions (queries that restrict only one dimension hierarchy also restrict columns in the lineorder fact table). The customer dimension has a hierarchy c_city(250)-c_nation(25)-c_region(5), where parenthesized numbers represent cardinalities of the named columns; The

supplier dimension has the same city-nation-region hierarchy, but with s_ prefixes; the part dimension has

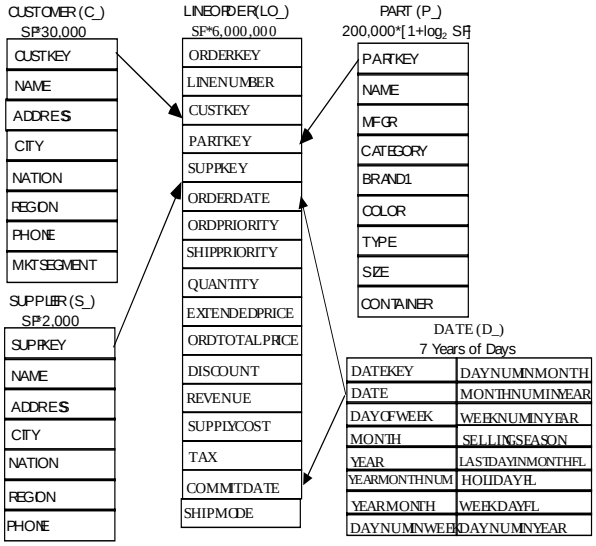

Figure 2: Star Schema Benchmark (SSB) Schema

hierarchy brand1(1000)-category(25)-mfgr(5) and the date dimension has a day-week-month-quarter-year hierarchy (though weeks do not roll up to months, they range in the same order, a crucial point in clustering; there are just under seven years of days). The SSB we ran for this paper has Scale Factor 10 (SF10), and thus 60,000,000 rows in the 6 GB lineorder fact table.

2.1 ADC Indexed Fact Table in Sort Order

One approach to ADC clustering adjoins physical copies of a column from each dimension to the fact table. We adjoin columns from high levels of dimension hierarchies commonly used in query restrictions, and sort the fact table data in order by a concatenation of these columns, then index these adjoined columns in the result. In SSB, we achieve this by using a Select statement on the original SSB Star Schema that extracts the fact table and the adjoined columns, write these rows out as data, then load this data into the adjoined version of the fact table and index it.

From SSB hierarchies named at the end of the last section, here are the adjoined columns (and their cardinalities) that determine fact table sort order: d_year(7), s_region(5), c_region(5) and p_category(25). These columns generate a moderate number of conjoint values: 7x5x5x25 = 4375. All rows having the same conjoint value will be clustered when loaded in this sort order, and if we picture the fact table as a cube with each edge a dimension range of adjoined columns, the clustered conjoint values will be cubical cells partitioning the cube, while most multi-dimensional range restrictions intersect a local rectilinear region of these cells. With only 4375 such cells (on about 6 GB of lineorder data at SF10) we ensure that cells contain enough data that sequential access within the cell outweighs disk inter-cell access time. The right number of cells depends on the size of the fact table and disk performance. We note a few minor drawbacks of the sort order.

1. Queries with restrictions on dimension columns in the ADC hierarchy need restrictions added to the adjoined columns of that hierarchy in the fact table.
2. An insert of a new row to the fact table must have appropriate ADC column values adjoined. The row to be inserted will have foreign keys to dimension rows that determine these ADC column values. Naturally the newly inserted rows will not be in sort order.

2.2 ADC Native Multi-Dim Clustering

DB2 was the first database product to provide a native ability to cluster by more than one column at a time, using *Multi-Dimensional Clustering (MDC)*, introduced in 2003 [1,3,4,5,10]. This method partitions table data into cells (physically organized as sets of *Blocks* on disk), by treating some columns ***within the table*** as orthogonal axes of a cube. As with the concatenated sort order index of Section 2.1, each cell corresponding to a different conjoint combination of individual values of these cube axis columns. However the fact that the "dimensions" of a table in MDC are columns within the table means that there is still a need to adjoin columns from the dimensions of a Star Schema to the fact table to accelerate typical Star Schema queries. This can be done in DB2 either by adjoining selected dimension columns in a select statement and reloading the table as in Section 2.1, or by creating a materialized view (MV) to adjoin the columns. It turns out, however, that presorting the fact table data by the adjoined columns before loading into MDC improves performance, probably because otherwise MDC Blocks on disk don't cluster the cells sufficiently for best performance. Adjoining columns to MDC using a MV solves minor drawback 2, but not 1, listed at the end of Section 2.1.

Oracle too has a method called *partitioning* (and *sub-partitioning*) that provides native multi-dimensional clustering [9,12]. Many of the values of MDC apply, such as the need to adjoin columns, and sorting the fact table in advance being of value in MVs.

3. The Star Schema Benchmark

The Star Schema Benchmark, or SSB, was devised to evaluate database system performance of star schema data mart queries. The SSB schema is derived from the TPC-H benchmark [15], but in a highly modified form. The details of the modification [2] are of interest to da-

ta warehouse practitioners: Given a normalized database schema, how can it be transformed to star schema form? Figure 2 gives the Schema of the SSB.

The 13 queries of SSB [2] are grouped into *Query Flights* that represent different *types* of queries, e.g.: different number of restrictions on dimension columns, while queries within a Flight vary selectivity of the clauses so that later queries have smaller filter factors.

3.1 Experimental Results

We used 3 commercial database products, anonymized with names A, B and C, running SSB tables at SF10. Our measurements were performed on a Dell 2900 running Windows Server 2003, with 8 GB of RAM, two dual-core processors (3.20 GHz), and data on RAID0 with 4 Seagate 15000 RPM SAS disks (136 GB each), stripe size 64KB. All queries were run from cold starts. Parallelism to support disk read ahead was employed on all products to the extent possible.

We measured two different forms of the lineorder table, one with no adjoined columns from the dimensions (called the BASE form), and one with four dimension column values adjoined, d_year, s_region, c_region and p_category. Results are shown in Table 3. The Geometric mean (Gmean) in the last line summarize (as in TPC-H) the preceding measures of the column. For products A and B, both row stores, the ratio of BASE Elapsed time to ADC Elapsed time is 12.4 to 1 (for A) and 8.7 to 1 (for B), a significant speedup. For Product C, a column store, the Elapsed time ratio is 5.48 to 1, a more moderate speedup, but the base times for C are much smaller because only part of the table data needs to be accessed. Product C is the fastest for both the base and ADC cases.

	A Base Case		B Base Case		C Base Case		A ADC Case		B ADC Case		C ADC Case	
Query	Elapsed	CPU	Elapsed	CPU	Elapsed	CPU	Elapsed	CPU	Elapsed	CPU	Elapsed	CPU
Q1_1	99	9.9	43	2.62	9.8	1.04	7.6	1.04	7.9	0.49	4.1	0.018
Q1_2	58	5.22	41	2.26	8.7	0.8	8.1	1	8.4	0.45	2.7	0.015
Q1_3	55	4.4	37	0.52	6.6	0.56	7.6	0.93	8.1	0.4	2	0.004
Q2_1	63	10.08	49	3.19	14.4	2.17	2.4	0.14	6.4	0.42	1.2	0.002
Q2_2	66	1.98	45	2.79	14.1	1.99	1.9	0.09	5.9	0.35	0.8	0.001
Q2_3	23	0.92	41	1.44	14.3	1.07	1.6	0.06	5.8	0.33	3.3	0.01
Q3_1	66	10.56	58	3.54	14	3.12	6.3	0.7	7.8	0.72	6.2	0.065
Q3_2	55	9.35	46	1.06	12.6	2.21	3.2	0.24	4	0.24	1.1	0.002
Q3_3	13	0.39	15	0.39	13.6	1.25	2.8	0.21	3.5	0.18	2.7	0.007
Q3_4	11	0.22	6	0.2	7.1	0.72	4.7	0.09	1.8	0.05	0.7	0.001
Q4_1	70	11.2	58	3.54	18.2	3.79	5.6	0.7	3.3	0.29	2.2	0.006
Q4_2	66	10.56	56	3.08	20.3	3.39	2.3	0.13	1.8	0.15	5.8	0.099
Q4_3	39	1.95	49	1.62	20.2	2.69	1.5	0.04	1	0.07	4.3	0.052
GMean	44.7	3.40	36.8	1.50	12.6	1.60	3.60	0.24	4.23	0.256	2.29	0.0081

Table 3 Measured Performance of Queries on Products A, B and C

4. REFERENCES

[1] Bhattacharjee B. et al. *Efficient Query Processing for Multi-Dimensional Clustered Tables in DB2*, VLDB 2003.

[2] Chen, X., O'Neil, P., O'Neil, E. *Adjoined Dimension Column Clustering (ADC Clustering) to improve Data Warehouse Query Performance.* http://www.cs.umb.edu/~poneil/ADCClustering.pdf

[3] Cranston, L. *MDC Performance: Customer Examples & Experiences.* http://www.research.ibm.com/mdc/db2.pdf

[4] IBM Research. *DB2's Multi-Dimensional Clustering*. http://www.research.ibm.com/mdc/

[5] Kennedy, J. *Introduction to Multidimensional Clustering with DB2 UDB.* IBM DB2 Inf Mgt Tech Conf,, 2005.

[6] Kimball, R. and Ross, M, *The Data Warehouse Toolkit, Second Edition*, Wiley, 2002.

[7] Lightstone, S., Teorey, T. and Nadeau, T., Physical Database Design, Morgan Kaufman, 2007.

[8] Oracle White Paper. *Star Queries in Oracle 8.* June 1997.

[9] Oracle, *Partitioning in Database 10g Release 2*. May 2005.http://www.oracle.com/solutions/business_intelligence/partitioning.html

[10] Padmanabhan S. et al. *Multi-Dimensional Clustering: A New Data Layout Scheme in DB2*. SIGMOD 2003.

[11] Red Brick Systems White Paper. *Star Schema Processing for Complex Queries*. 1995. http://www-306.ibm.com/software/data/informix/redbrick/

[12] Scalzo, B. *Oracle DBA Guide to Data Warehousing and Star Schemas*. Prentice Hall PTR Oracle Series.

[13] Stonebraker M. et al. *One Size Fits All? Part2: Benchmarking Results*. Keynote address, CIDR 2007, http://www-db.cs.wisc.edu/cidr/cidr2007/papers/cidr07p20.pdf

[14] Sybase Data Warehousing White Paper. *Sybase Interactive Warehouse*. 1997.
http://www.dbmsmag.com/9708d17.html

[15] *TPC-H Version 2.4.0 in PDF Form*.
http://www.tpc.org/tpch/default.asp