CS565: Data Mining

Written Assignment 1

Due Date: October 1st, 2007 in class

1. *Bookmania* has two bookstores located in New York and San Francisco. These bookstores manage their information using operational database systems. The two databases have the following schemata:

New York:
Employee (employee_key, name, date_of_birth, dept_key, start_date, contact_number)
Department (department_key, name, manager_id, staff_number)
Book (book_key, book_title, price, year, publisher, authors, book_type)
//book_type is fully specified. E.g., computer science, literature, etc.
Customer (customer_key, name, customer_type, customer_address, customer_age)
Author (author_id, name, author_age, author_address)
Publisher (publisher_key, publisher_address, publisher_tel, publisher_fax)
Purchase (purchase_key, book_key, book_number, book_price, purchase_time)
Sales (book_key, customer_key, unit_of_book, discount, sales_time)
Time (time_key, day, day_of_week, month, quarter, year)
Inventory (book_key, book_number_balanced, time)

San Francisco:
Employee (employee_key, name, age, dept_belongs_to, start_date, phone)
Department (department_key, name, manager_id, staff_number)
Book (book_key, book_title, price, year, publisher, authors, book_type)
//book_type is abbreviated. E.g., CS denotes computer science books; LI denotes literature books, etc.
Customer (customer_key, name, customer_type, customer_address, customer_age)
Author (author_id, name, author_age, author_address)
Publisher (publisher_key, publisher_address, publisher_tel, publisher_fax)
Purchase (purchase_key, book_key, unit_of_book, purchase_price, purchase_time)
Sales (book_key, customer_key, unit_of_book, discount, sales_time)
Time (time_key, day, day_of_week, month, quarter, year)
Inventory (book_key, book_number_balanced, time)

In order to improve sales strategy and inventory plan, the company wants to perform the following analysis tasks:

(1) Analyze the total sales for each book (title, type), time periods (month, quarter, year), customer (customer_type, customer_age, customer_age_group). Note: any meaningful age split for customer age group is acceptable.

(2) Analyze the average inventory of some book when that book was bought for each time period and each book title and/or type. For example, in January, the bookstore bought book A 3 times, when buying book A at the first time, the number of such books in the bookstore was 10; when buying it for the 2nd time, the number of books was 12; When buying it for the 3rd time, there were still 8 books in the bookstore. Then the average inventory of book A in this month was 10.
Notice, there are two subjects: *sales* and *inventory*, with different dimensions and measures.

Design the data warehouse schema for this company using some model (star, snowflake, fact constellation). Provide full details of the tables you use (table names, attribute names and types). In the design, you need to specify:

a) Which tables/attributes are not used in the construction of the data warehouse;

b) The relationship between the keys that appear in the fact tables and dimension tables;
c) The concept hierarchies of some dimensions if they exist; and
d) If there exist conflicts on attribute names, attribute types, or values, state how to achieve consistency in the derived data warehouse schema.

2. Consider only the data cube on *sales*, above. How many cuboids do you need to compute if you adopt the full materialization strategy?

3. Consider the data cube on sales and assume the following:
a) A MOLAP server is used to store the summarized data, and the basic structure used is the multidimensional array.
b) There are no hierarchies in the dimensions Customer, Book, and Time.
c) The numbers of distinct values in the dimensional attributes Customer, Book and Time, are 200, 2000, and 1000, respectively.
d) Each dimension is split into 10 equi-sized partitions. For example, the number of customers is 200 and these are split into 10 partitions C_0, C_1, ..., C_9, with 20 customers each. Similarly, the partitions for Book and Time are denoted by B_0, B_1, ..., B_9 and T_0, T_1, ..., T_9. The ID of a chunk $C_xB_yT_z$ is defined by 100*x+10*y+z. For example, the ID of chunk $C_3B_9T_2$ is 392. Thus:

chunk	**ID**
$C_0B_0T_0$	0
$C_0B_0T_1$	1
…	…
$C_0B_0T_9$	9
$C_0B_1T_0$	10
…	…
$C_0B_9T_0$	90
…	…
$C_0B_9T_9$	99
$C_1B_0T_0$	100
…	…
$C_9B_9T_9$	999

If there is enough memory space and only one-scan is permitted on the base cuboids, show the most efficient ordering of the multiway array aggregation and the corresponding memory requirements.

4. For the *sales* data cube, consider the 2-D cuboid {book_type, customer-age-group}. Assume that there are 8 book types in the bookstore, and 6 customer age groups.
The following table shows part of the data in this cuboid.
a) show the bitmap indexes for this cuboid.
b) provide an example query for which these indexes would be useful.

RID	book_type	costomer_age_group
R1	CS	G3
R2	EE	G4
R3	LAW	G5
R4	MEDICAL	G4
R5	MEDICAL	G5
R6	CARTOON	G1
R7	CARTOON	G3
...	...	...

5. Consider the *sales* data cube and explain:
a) which of the following cuboids can be used to process the query, and
b) which one can be used to answer the query most efficiently?
Justify your answer and assumptions.

Query: compute the total sales for each {book_type, quarter} where the customer age is between [18,30) and year=2006.

Five materialized cuboids are available:
Cuboid 1: {book_type, quarter, customer_age} where year=2006
Cuboid 2: {book_type, month, customer_age} where year=2006
Cuboid 3: {year, customer_age} where book_type="CS"
Cuboid 4: {book_title, quarter, customer_age} where year=2006
Cuboid 5: {book_title, customer_age}

6. Consider the *sales* data cube again. Assume that the cuboid {book_title, month, customer_age} is materialized and its size is 18GB. Assume that the disk size of your system is 20GB. The following is a list of potential cuboids to be materialized (note: not all cuboids are included!), together with their sizes:

Cuboid	**size**
{book_title, quarter, customer_age}	7GB
{book_title, year, customer_age}	3GB
{book_title, year, customer_age_group}	1.5GB
{book_type, month, customer_age_group}	700MB
{book_type, quarter, customer_age}	800MB
{book_type, month, customer_age}	1GB
{book_type, month}	100MB
{book_title, year}	800MB
{book_title, customer_age}	2GB
{book_type, customer_age_group}	100MB
{month, customer_age_group}	400MB
{year, customer_age}	300MB
{book_title}	600MB
{month}	100KB
{customer_age}	1KB
{}	8bytes

Now consider the following *frequent* queries:

- Q1: Compute the total sales per book type and customer_age_group for a selected month.
- Q2: Compute the total sales per year for a selected book
- Q3: Compute the total sales per quarter and book type for a selected customer age
- Q4: Compute the total sales of all books for a selected quarter

selected means that any value can be selected in the query. For example in Q1 any month can be in the select clause of the query with equal probability.
a) Assume that only the base cuboid {book_title, month, customer_age} is materialized. What is the average number of bytes that have to be read in order to answer the queries above? Hint: To find the average access cost you need to sum the costs of all queries and divide by four.
b) Which of the cuboids above you would select to materialize, given the fact that you cannot store all of them due to space constraints? Select the ones that will give the maximum benefit to your queries, assuming that there are no indexes on them. What is the total space occupied by the cuboids you have selected to materialize? How many bytes do your queries access on the average now?
Hint: Your selection should be based on the space limitation you have and the cost savings the materialized cuboids provide for your queries.

Note: For some questions there may be more than one solution. You need to justify your solution in order to be considered correct.