

Computer Science COMP3420 in 2010 – Answers to Assignment One

Due: 5pm Thursday, April 22
Late Penalty: 20% per day

*No programming is needed for this assignment. Before you hand in your assignment, make sure you have put **your name, student ID and your tut/lab group** into the front page of your assignment. Put your work into the COMP3420 box on the ground floor in CSIT Building. The total mark for this assignment are 20 points (20% of the final marks). In addition, there are extra 3 points (3% of the final marks) for bonus questions. Notice that for all calculation questions, show all your major working steps. In other words, if you just write down the final result of each question, you would not receive any marks!*

Question 1 (2/20) What are the major differences between an enterprise data warehouse and an operational database? Can you describe the main components of a data warehouse and their functions?

Answer: The data in a data warehouse (DW) is historical, consolidated data (e.g., past 5-10 years), the operations on the data are summarizations, aggregations, and multi-dimensional data analysis. The queries in DW are usually ad hoc nature and very complicated. DW serves for data analysis and decision making support by providing aggregated, summarized data. Thus, data warehousing and OLAP are an essential step towards the knowledge discovery process. Operational databases deal with the current data, on which the frequent operation is the OLTP. An operational database is designed and tuned for known tasks and workloads such as indexing and hashing using primary keys, searching for particular records, and optimizing “standard” queries. It also supports the concurrent processing of multiple transactions. Concurrency control and recovery mechanisms are required to ensure the consistency and robustness of transactions.

Refer to Fig. 3.12 on page 131 in the textbook. The three-tier data warehouse architecture consists of the bottom tier (data warehouse server), the middle tier (OLAP server), and the top-tier (front-end tools).

- The bottom tier is almost a relational database system. Back-end tools and utilities are used to feed data into the bottom tier from operational databases or other external sources. These tools and utilities (ETL) perform data extraction, cleaning, and transformation, as well as load and refresh functions to update the data warehouse. The data are extracted using application program interfaces known as gateways. A gateway is supported by the underlying DBMS and allows clients program to generate SQL code to be executed at a server. This tier also contains a metadata repository, which stores information about the data warehouse and its contents (source, staging, data integration [ETL/ELT], metadata repository).
- The middle tier is typically implemented using either a relational OLAP model

(ROLAP) or a multidimensional OLAP model (MOLAP), that is, a special purpose server directly implements multidimensional data and operations (presentation area).

- The top tier contains query and reporting tools, analysis tools, and/or data mining tools (access).

The bottom tier is used for providing high quality data for the data warehouse (the middle tier), the middle tier is major operational engine, OLAP operations are executed at this tier. The top tier is an interface between the human being and the machine, which provide consumers with the query results through different means like charts, dashboard, etc.

Question 2 (3/20) Suppose a group of 21 *sensory reading* values has been sorted as follows.

4, 6, 8, 9, 9, 11, 14, 14, 16, 21, 22, 30, 35, 40, 45, 50, 51, 59, 70, 92, 115.

Partition the data into **four** bins using the following four methods.

- (a) equal-width partitioning
- (b) equal-depth partitioning
- (c) smooth the data in bins by bin means
- (d) smooth the data in bins by bin boundaries

Answer: We allocate the data into four bins as follows.

- (a) Equal-width partitioning: $R_{min} = 4, R_{max} = 115, N = 4$, and $W = \frac{R_{max}-R_{min}}{N} = \frac{111}{4} \approx 27.7 = \lfloor 27.7 \rfloor = 27$. Therefore, the three intervals are $[4, 31], [32,59], [60,87], [88,115]$.
 - Bin 1: 4, 6, 8, 9, 9, 11, 14, 14, 16, 21, 22, 30
 - Bin 2: 35, 40, 45, 50, 51, 59
 - Bin 3: 70
 - Bin 4: 92, 115
- (b) Equal-frequency partitioning
 - Bin 1: 4, 6, 8, 9, 9
 - Bin 2: 11, 14, 14, 16, 21
 - Bin 3: 22, 30, 35, 40, 45
 - Bin 4: 50, 51, 59, 70, 92, 115
- (c.1) Smooth the data in bins by bin means(equal-width)
 - Bin 1: 13.6, 13.6, 13.6, 13.6, 13.6, 13.6, 13.6, 13.6, 13.6, 13.6,13.6, 13.6
 - Bin 2: 46.6, 46.6, 46.6, 46.6, 46.6

- Bin 3: 70
- Bin 4: 103.5, 103.5
- (c.2) Smooth the data in bins by bin boundaries (equal-frequency)
 - Bin 1: 7.2, 7.2, 7.2, 7.2, 7.2, 7.2
 - Bin 2: 15.2, 15.2, 15.2, 15.2, 15.2
 - Bin 3: 34, 34, 34, 34, 34
 - Bin 4: 72.8, 72.8, 72.8, 72.8, 72.8
- (d.1) Smooth the data in bins by bin boundaries (equal-width)
 - Bin 1: 4, 4, 4, 4, 4, 4, 30, 30, 30, 30, 30, 30 (mean=13.6)
 - Bin 2: 35, 35, 35, 59, 59, 59 (mean=46.6)
 - Bin 3: 70 (mean=70)
 - Bin 4: 92, 105 (mean=103.5)
- (d.2) Smooth the data in bins by bin boundaries (equal-frequency)
 - Bin 1: 4, 4, 9, 9, 9 (mean=7.2)
 - Bin 2: 11, 11, 11, 21, 21 (mean=15.2)
 - Bin 3: 22, 22, 45, 45, 45 (mean=34)
 - Bin 4: 50, 50, 50, 115, 115, 115 (mean=72.8)

Question 3 (4/20)

Suppose a hospital tested the age and body fat data for 18 randomly selected adults with the following results:

age	21	23	23	27	39	44	47	48	51
%fat	8.5	28.5	9.8	19.8	35.4	27.9	26.4	27.2	29.2

age	52	54	54	57	57	57	58	59	61
%fat	34.6	43.5	29.8	38.4	32.2	34.1	31.9	41.9	34.7

- (a) Calculate the mean, median, and standard deviation of *age* and *%fat*.
- (b) Draw the boxplots for *age* and *%fat*.
- (c) Normalise the two variables *age* and *%fat* using *min-max normalization* where $min = 0$ and $max = 1$.
- (d) Calculate the *correlation coefficient* (Pearson's product moment coefficient). Are these two variables positively or negatively correlated?

Answer:

- (a) Calculate the mean, median, and standard deviation of *age* and *%fat*.

$$\text{age_mean} = \frac{\sum_{i=1}^{18} a_i}{18} = \frac{21+23+\dots+59+61}{18} = 46.22.$$

$$\text{age_median} = \frac{51+52}{2} = 51.5.$$

$$\text{age_standrd_deviation} = \sqrt{\frac{1}{18} \sum_{i=1}^{18} (a_i - \text{age_mean})^2} \approx 13.31.$$

$$\text{\%fat_mean} = \frac{\sum_{i=1}^{18} f_i}{18} = \frac{8.5+28.5+\dots+41.9+34.7}{18} \approx 29.65.$$

$$\text{\%fat_median} = \frac{29.2+31.9}{2} = 30.85.$$

$$\text{\%fat_standrd_deviation} = \sqrt{\frac{1}{18} \sum_{i=1}^{18} (f_i - \text{\%fat_mean})^2} \approx 9.10.$$

- (b) Draw the boxplots for *age* and *%fat*.

age: five number summary: 21 (minimum), 39 (Q1), 51.5 (median), 57 (Q3), 61 (maximum)

%fat: five number summary: 8.5 (minimum), 27.2 (Q1), 30.85 (median), 34.7 (Q3), 43.5 (maximum)

- (c) Normalise the two variables *age* and *%fat* using *min-max normalization* where $new_{min} = 0$ and $new_{max} = 1$.

The formula for min-max normalization is as follows.

$$v' = \frac{v - min}{max - min} (new_{max} - new_{min}) + new_{min},$$

where v' is the resulting value, v is the original value, $min = 21$, $max = 61$ for *age* and $min = 8.5$, $max = 43.5$ for *fat*, $new_{min} = 0$ and $new_{max} = 1$. Thus, we have

age	21	23	23	27	39	44	47	48	51
age'	0	0.05	0.05	0.15	0.45	0.58	0.65	0.68	0.75
%fat	8.5	9.8	19.8	26.4	27.2	27.9	28.5	29.2	29.8
%fat'	0	0.04	0.32	0.51	0.53	0.55	0.57	0.59	0.61

age	52	54	54	57	57	57	58	59	61
age'	0.78	0.83	0.83	0.9	0.9	0.9	0.93	0.95	1
%fat	31.9	32.2	34.1	34.6	34.7	35.4	38.4	41.9	43.5
%fat'	0.67	0.68	0.73	0.75	0.75	0.77	0.85	0.95	1

- (d) Calculate the *correlation coefficient* (Pearson's product moment coefficient). Are these two variables positively or negatively correlated?

Person's coefficient given as follows.

$$\frac{\sum_{i=1}^{18} [(a_i \times f_i) - (N \times \text{age_mean} \times \text{fat_mean})]}{N \times \sigma_A \times \sigma_B} \approx 0.79$$

As the correlation coefficient is positive, we can say that the two variables are positively correlated.

Question 4 (4/20)

Suppose that a data warehouse consists of the four dimensions *hospital*, *time*, *doctor*, and *patient*, and the two measures *count* and *charge*, where *count* is the number of visits of a patients to a doctor and *charge* is the fee that a doctor charges a patient for a visit.

- (a) Draw a *star* and a *snowflake* schema diagrams for the above data warehouse.
- (b) Starting with the base cuboid [hospital, day, doctor, patient], what specific *OLAP operations* (e.g. roll-up) should be performed in order to list the total fee collected by each doctor in 2009?
- (c) Starting with the base cuboid [hospital, day, doctor, patient], what specific *OLAP operations* should be performed in order to list the total fee collected by Dr John Rudd in Canberra hospital from 2002 to 2009?
- (d) To obtain the same list as (c), write an SQL query assuming the data is stored in a relational database with the schema *fee*(day, month, year, doctor, hospital, patient, count, charge).

Answer:

- (a) Draw a *star* and a *snowflake* schema diagrams for the above data warehouse.

Star schema:

```
define cube fee [hospital_key, time_key, doctor_key, patient_key]:charge_cost,count
define dimension time as (time_key, day, month, year);
define dimension doctor as (doctor_key, doctor_name, clinic_address, address, phone_number,
clinic name);
define dimension patient as (patient_key, patient_name, home_address, phone_number,
date_of_birth,sex, medical_record); define dimension hospital as (hospital_key, hospital_name, address, phone_number);
```

Snowflake schema:

```
define cube [fee hospital_key, time_key, doctor_key, patient_key]:charge_cost,count
define dimension hospital as (hospital_key, hospital_name, address, phone_number);
define dimension time as (time_key, day, month, quarter, year);
define dimension doctor as (doctor_key, doctor_name, clinic_address, address,
phone_id(handphone_number,mobile_number,house_phone_number), clinic name);
define dimension patient as (patient_key, patient_name, home_address, address, phone_number,
dat_of_birth, sex, medical_record_ID(description_ID, blood_type, allergies,medical_history));
```

- (b) Starting with the base cuboid [hospital,day, doctor, patient], what specific *OLAP operations* (e.g. roll-up) should be performed in order to list the total fee collected by each doctor in 2009?

Roll up on dimension of time from “day” to “year” (or from “day” to “month”, then from “ month” to “year”), then **slice** on time “year=2009” to see the total charges of all doctors in 2009.

- (c) Starting with the base cuboid [hospital, day, doctor, patient], what specific *OLAP operations* should be performed in order to list the total fee collected by Dr John Rudd in Canberra hospital from 2002 to 2009?
 - slice on hospital at “Canberra”
 - slice on doctor for ‘John Rudd’
 - roll up on time from day to year
 - slice time from 2002 to 2009 and roll up
 - roll up on patient

- (d) To obtain the same list as (c), write an SQL query assuming the data is stored in a relational database with the schema $fee(day, month, year, doctor, hospital, patient, count, charge)$.

```

SELECT sum(charge*count)
FROM fee
WHERE doctor='John Rudd'
AND
hospital='Canberra'
AND
year > 2001
AND
year < 2010

```

Question 5 (3/20)

Assume that a data warehouse contains 100 dimensions, each with ten levels of granularity.

- (a) Users are mainly interested in 10 particular dimensions, each having four frequently accessed levels for rolling up and drilling down. Design a data cube structure to efficiently support this preference, and justify your design.
- (b) A user wants to drill through the data cube, down to the raw data for one or two particular dimensions. How would you support this feature?

Answer:

- (a) Users are mainly interested in 10 particular dimensions, each having four frequently accessed levels for rolling up and drilling down. Design a data cube structure to efficiently support this preference, and justify your design.

Consider specific ten dimensions $A, B, C, D, E, F, G, H, I, J$, assume that the four frequently accessed levels of these dimensions are $A_i, B_i, C_i, D_i, \dots, I_i, 1 \leq i \leq 4$. Thus, instead of computing all cuboids (11^{100} cuboids) of the data cube, we only compute $(L'_A + 1)(L'_B + 1)(L'_C + 1)(L'_D + 1) \cdot (L'_I + 1) = 5^{10}$ cuboids, and they are $A_i, B_i, C_i, D_i, \dots, I_i, A_i B_j, A_i C_j, A_i D_j, \dots, A_i I_k, B_i C_j, B_i D_j, C_i D_j, A_i B_j C_k, B_i C_j D_k, A_i C_j D_k, A_i B_j C_k D_l, \dots, 1 \leq i, j, k, l \leq 4$.

Because the queries are only applied for these dimensions and levels, we can save much computation for the entire data cube and the space requirement. For any specific query (for this preference), it can be answered by one of the cuboids without need of other cuboids.

- (b) A user wants to drill through the data cube, down to the raw data for one or two particular dimensions. How would you support this feature?

To support the drill down operation, we assume that the lowest level (specified) of the hierarchy of a dimension must be stored in the iceberg of datacube. For example, assume that level A_1 is higher than A_2 , A_2 is higher than A_3 , the similar assumption can be applied to other dimensions. Thus, if we are working at level A_1 currently and want to drill down to A_3 , we can drill down to A_2 first, followed by A_3 .

Question 6 (4/20) There are several cube computation methods including *multiway array computation*, *bottom-up computation*, and *star-cubing*. Briefly describe these three methods (i.e., use one or two lines to outline the key points), and compare their feasibility and performance under the following conditions.

- (a) Computing a dense full cube of low dimensionality (e.g., less than 8 dimensions)
- (b) Computing a sparse iceberg cube of high dimensionality (e.g., over 100 dimensions)
- (c) Computing an iceberg cube of around 10 dimensions with a highly skewed data distribution

Answer: Multiarray array aggregation approach is used for computing a full datacube when the cube representation is a multi-dimensional array. The basic idea is to partition the base cuboid into chunks that fits into memory, and perform aggregation on a chunk for many different sharded cuboid computation. In other words, Multi-way's computation starts from the base cuboid and progresses upward toward more generalized, ancestor cuboids. It is suitable for dense and low dimensionality cube computation. However, it cannot take advantage of Apriori pruning, which requires a parent node to be computed before its child nodes.

Bottom-Up Computation (BUC) is used for computing a sparse, high dimensionality iceberg cube, and its representation is based on the relational tables. BUC's major contribution is the idea of sharing partitioning costs. However, unlike multi-way, it does not share the computation of aggregates between parent and children group-bys.

To compute a low dimensionality cube with highly skewed dataset, the Star-Cubing approach should be adopted, which combines the advantages of both multiway and BUC methods, its first traverse the base cuboid to the apex cuboid as does Multiway, it then prunes in downward direction as BUC by the Apriori property. It aims to exploit the simultaneous computation of shard dimensions.