به نام خدا

دانشگاه تهران

دانشکدگان فنی

دانشکده مهندسی برق و کامپیوتر

درس داده کاوی

تمرین دوم

نام و نام خانوادگی : حسین سیفی

شماره دانشجویی : ۸۱۰۱۰۰۳۸۶

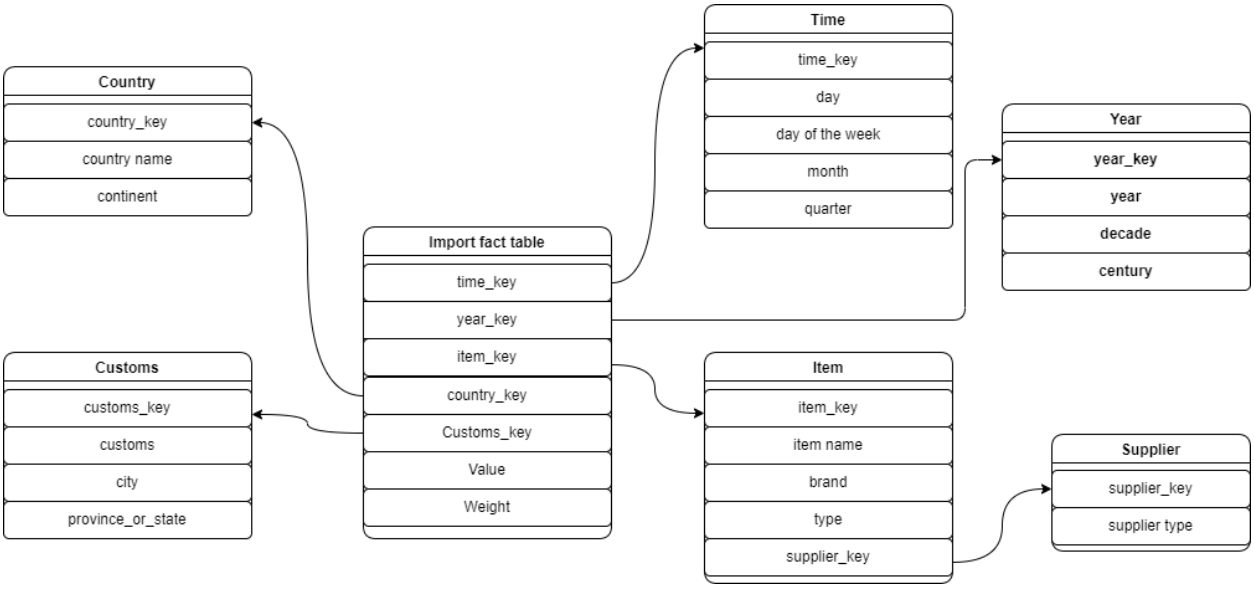
اردی‌بهشت ۱۴۰۱

فهرست

سوال ۱

الف

برای ذخیره سازی داده‌های مربوط به واردات مدل برف دانه(Snow flake) را پیشنهاد می‌کنیم. این مدل نسبت به مدل ستاره‌ای سرعت بازیابی پایین‌تری را ارائه می‌دهد اما مزیت آن نسبت به روش ستاره‌ای این است که در مصرف حافظه صرفه جویی می‌کند و داده‌های تکراری را نگه‌داری نمی‌کند. مدل پیشنهادی طراحی شده برای این سوال در شکل ۱ قابل مشاهده است:



شکل ۱

در این نمودار یک جدول Fact برای واردات وجود دارد که در این جدول کلیدهایی به ابعاد(Dimensions) جنس(Item)، تاریخ(Time)، سال(Year)(این بعد به منظور توانایی پاسخگویی به پرس و جو ۱ ایجاد شده است)، کشور(Country) و گمرک(Customs) اشاره می‌کنند که هرکدام از ابعاد ذکر شده به ترتیب نشان دهنده جنس وارد شده، تاریخ(شامل روز و ماه و فصل)،سال، کشور صادر کننده و محل ورود هستند. همچنین معیارهای(Measures) وزن(Weight) و ارزش(Value) نیز در این جدول وجود دارند.

با توجه به مدل برف‌دانه این سیستم، بُعد «جنس» شامل کلید بعد تولید کننده نیز می‌شود. همچنین هرکدام از ابعاد شامل سلسله مراتبی از مقادیر هستند.

در نهایت به نظر می‌رسد که مدل فوق از پرس و جوهای مطرح شده پشتیبانی می‌کند.

ب

عملیات OLAP مورد نیاز برای هر کدام از پرس و جوها به ترتیب مراحل به شرح زیر می‌باشد:

پرس و جو I: میانگین وزن واردات گندم در سال ۱۴۰۰ از یک گمرک خاص در هر ماه

1. Roll up from time_key to month
2. Roll up from year_key to year
3. Roll up from item_key to item_name
4. Roll up from customs_key to customs
5. Dice on item_name == ”wheat” and year == 1400 and customs == “Gomrok_khas”

6. Average Weight on month

پرس و جو II: ارزش کل واردات از چین از شهریور ۱۴۰۰ تا کنون

1. Roll up from country_key to country
2. Roll up from year_key to year
3. Roll up from time_key to month
4. Dice on ((year == 1400 and (6<=month<=12)) or (year == 1401 and month == 1 or 2)) and (country == “China”)
5. Sum Value

پرس و جو III: کالایی که ارزش واردات آن در اسفند ۱۴۰۰ بیشتر از سایر کالاها بوده است

1. Roll up from time_key to month
2. Roll up from item_key to item
3. Roll up from year_key to year
4. Dice on month == 12 and year == 1400
5. Max Value on item

پرس و جو IV: ماهی از سال ۱۴۰۰ که بیشترین میزان واردات گندم از نظر وزن را داشته است

1. Roll up from time_key to month
2. Roll up from item_key to item_name
3. Roll up from time_key to month
4. Roll up from year_key to year
5. Dice on item == “Wheat” and year == 1400
6. Max Weight on month

سوال ۲

الف

A. $(a_1,a_2,b_3,a_4,\ldots,a_{10}):count=20$
B. $(b_1,b_2,a_3,a_4,\ldots,a_{10}):count=10$
C. $(b_1,a_2,b_3,a_4,\ldots,a_{10}):count=20$
D. $(a_1,b_2,b_3,a_4,\ldots,a_{10}):count=50$

برای محاسبه تعداد تمامی Aggregate cellها می‌توان از اصل شمول و عدم شمول به شکل زیر استفاده کرد.

$$C(A \cup B \cup C \cup D)$$
$$= C(A) + C(B) + C(C) + C(D) + C(A \cap B) + C(A \cap C) + C(A \cap D) + C(B \cap C)$$
$$+ C(B \cap D) + C(C \cap D) - C(A \cap B \cap C) - C(A \cap B \cap D) - C(A \cap C \cap D)$$
$$- C(B \cap C \cap D) + C(A \cap B \cap C \cap D)$$

هر کدام از تعداد سلول‌های تکی برابر تعداد زیر مجموعه‌ها آن(2^N) و هر کدام از سلول‌هایی که حاصل تجمیع(Aggregate) بیش از یکی از سلول‌های پایه هستند برابر $2^N - 2^M$است که N برابر تعداد ابعاد هرکدام و M برابر تعداد ابعاد مشترک است.همچنین تعداد اشتراکات هر کدام از سلول‌های پایه نیز برابر 2^M می‌باشد که M برابر تعداد ابعاد مشترک است. در نتیجه تعداد هر کدام از جمله‌های فوق به شکل زیر محاسبه می‌شود:

1. $C(A) = C(B) = C(C) = C(D) = 2^{10}$
2. $C(A \cap B) = C(A \cap B \cap C) = C(A \cap B \cap D) = C(B \cap C \cap D) = C(A \cap B \cap C \cap D) = 2^7$
3. $C(B \cap C) = C(B \cap D) = C(C \cap D) = C(A \cap C \cap D) = 2^8$
4. $C(A \cap C) = C(A \cap D) = 2^9$

در نتیجه تعداد Aggregate cellهای حاصل از یک سلول پایه برابر $۴*۲^{۱۰}$ ، اشتراکات دو سلول پایه برابر ۱۹۲۰، اشتراکات سه سلول پایه برابر ۶۴۰ و اشتراکات هر چهار سلول برابر ۱۲۸ می‌باشد. پس تعداد کل Aggregate cellها به شکل زیر به دست می‌اید:

$$C(A \cup B \cup C \cup D) = 4096 - 1920 + 640 - 128 = 2688$$

ب

در صورتی که سلول‌های پایه را به شکل فوق شماره گذاری کنیم، با رعایت کردن شرط Iceberg تنها می‌توان سلول‌ها Aggregate شده زیر که به همراه تعداد هر کدام در ادامه قابل مشاهده است را ایجاد کرد:

۱. A و D : $2*2^{10} - 2^9$
۲. C و D : $2*2^{10} – 2^8$
۳. A و B و D : $3*2^{10} – 2^7 – 2^9 - 2^8 + 2^7$
۴. A و C و D : $3*2^{10} – 2^9 – 2^9 - 2^8 + 2^8$
۵. B و C و D : $3*2^{10} – 2^8 – 2^8 - 2^8 + 2^7$
۶. A و B و C و D : همانطور که در بخش قبل محاسبه شد برابر۲۶۸۸ است.

ج

متاسفانه در این سوال به ابهام در تعریف برخوردیم و به نظر می‌رسد که تعریف کتاب با تعریف استاد در مورد سلول‌های بسته متفاوت است. بنابر تعریف کتاب DataCube کامل شامل ۱۴ سلول بسته به شکل زیر می‌باشد:

1. $(a_1,a_2,*,*,\ldots,*)$: count=20
2. $(b_1,b_2,*,*,\ldots,*)$: count=10
3. $(a_1,b_2,*,*,\ldots,*)$: count=50
4. $(b_1,*,a_3,*,\ldots,*)$:$count$=10
5. $(*,b_2,a_3,*,\ldots,*)$:$count$=10
6. $(b_1,a_2,*,*,\ldots,*)$:$count$=20
7. $(b_1,*,b_3,*,\ldots,*)$:$count$=20
8. $(*,a_2,b_3,*,\ldots,*)$:$count$=20
9. $(*,b_2,b_3,*,\ldots,*)$:$count$=50
10. $(*,*,a_3,*,\ldots,*)$:$count$=10
11. $(a_1,a_2,b_3,a_4,\ldots,a_{10})$:$count$=20
12. $(b_1,b_2,a_3,a_4,\ldots,a_{10})$:$count$=10
13. $(b_1,a_2,b_3,a_4,\ldots,a_{10})$:$count$=20
14. $(a_1,b_2,b_3,a_4,\ldots,a_{10})$:$count$=50

و بنابر تعریف استاد شامل سلول بسته زیر است:

1. $(*,*,*,a_4,\ldots,a_{10})$:$count$=20

سوال ۳

الف

- Multiway Array Computation: یک روش بالا به پایین است که از Cuboid پایه شروع می‌کند و به ترتیب Cuboidهای سطوح بعدی را می‌سازد. در این روش از قطعه‌های(chunk) چند بعدی استفاده می‌کند و اندازه این قطعات را به طوری انتخاب می‌کند که در حافظه اصلی گنجایش آن‌ها را داشته باشد. همچنین این روش برای کاهش هزینه تجمیع(Aggregation) را روی ابعاد مختلف انجام می‌دهد. به دلیل روند پایین به بالای این روش، نمیتوان با استفاده از خاصیت Apriori عملیات هرس کردن را انجام داد و در نتیجه امکان استفاده از بهینه سازی به روش کوه یخ(Iceberg) وجود ندارد.
- Bottom-Up Computation (BUC): این روش یک روش پایین به بالای محاسبه Datacube است که برای محاسبه Iceberg cubeها هم مناسب است. ساختن Cuboid در روش BUC از Apex Cuboid شروع می‌شود و به سمت Cuboid پایه حرکت می‌کند و به این روش امکان استفاده از خاصیت Apriori را می‌دهد. در این روش در صورت عدم وجود شروط مد نظر در هر پارتیشن، آن پارتیشن را هرس می‌کنیم و به مرحله قبل باز می‌گردیم و در غیر این صورت داده‌ها بر اساس پارتیشن‌ها گروه بندی می‌شوند و BUC به صورت بازگشتی روی آن پارتیشن فراخوانی می‌شود و این کار تا مرحله‌ای انجام می‌شود که تمامی ابعاد دیده شوند.

ب

برای محاسبه یک Datacube کامل چگال با تعداد ابعاد پایین روش Multiway array aggregation می‌تواند مناسب‌تر باشد چرا که این روش به کمک انجام محاسبات Cuboidهای متفاوت در بارگزاری(Load) یک قطعه از داده‌ها زمان کوتاه‌تری را برای داده‌های کامل چگال ارائه می‌دهد اما به دلیل نگهداری بخشی از همه‌ی Cuboidهای مورد محاسبه(در بعضی موارد Cuboid کامل را نگهداری می‌کند)، حافظه مصرفی بیشتری دارد. روش BUC می‌تواند مشکل حافظه مصرفی را حل می‌کند اما زمان اجرای بیشتری به خصوص برای داده‌های کامل چگال دارد چون در این نوع داده‌ها در هر زمان تنها یک Cuboid محاسبه می‌شود و همچنین هیچ بخشی هرس نمی‌شود و در نتیجه تمامی محاسبات باید به صورت کامل انجام شود و کاهش سرعت اجرا را نسبت به روش Multiway array aggregation شاهد خواهیم بود. از دیدی دیگر به دلیل وجود تعداد ابعاد پایین در این داده‌ها مشکل حافظه به صورت جدی وجود ندارد پس Multiway array aggregation می‌تواند روش مناسبی برای انجام محاسبات مورد نظر باشد.

ج

برای داده‌هایی با توزیع ناهمگون نیز Multiway array aggregation می‌تواند زمان اجرای بهینه تری را ارائه دهد. با توجه به مفهوم چولگی، روش BUC نمی‌تواند تضمینی برای هرس کردن بخشی از سلول‌ها و در نتیجه بهبود زمان اجرا و حافظه مورد نیاز ارائه دهد چرا که هر چند داده‌ها ناهمگون توزیع شده باشند، اما به هر حال ممکن است که شرط Iceberg را برآورده سازند و در نتیجه هرس نشوند. اما از سوی دیگر Multiway array aggregation تضمین می‌کند که هر قطعه(Chunk) از داده‌ها تنها یکبار در حافظه بارگزاری خواهند شد و محاسبات مربوط به هر قطعه در همان یکبار انجام خواهد شد و زمان اجرای بهینه و پایداری را ارائه می‌کند و تنها نیاز به مقدار مشخصی از حافظه دارد.

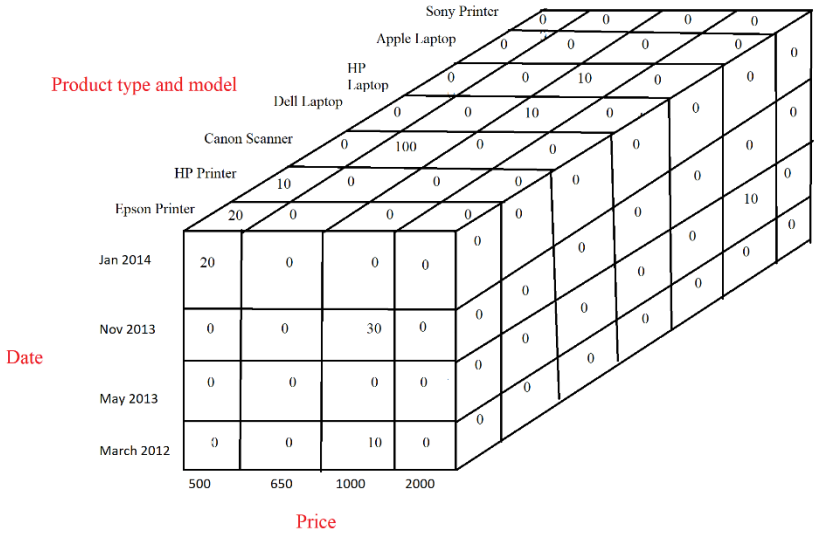
د

بهترین روش برای محاسبه Datacube خلوت(Sparse) با تعداد ابعاد بالا BUC می‌باشد چرا که این روش از خاصیت Apriori و شرط Iceberg استفاده می‌کند در نتیجه محاسبه سلول‌های تجمیع شده زیادی با اعمال شروط مد نظر انجام نمی‌شود تا محاسبات را به حداقل برساند در حالیکه روش Multiway array aggregation که روشی بالا به پایین است هیچ خلاصه سازی انجام نمی‌دهد و بخش‌هایی از

جداول هر یک از Cuboidها را نگهداری می‌کند و با توجه به تعداد ابعاد بالای داده‌ها مورد نظر، حافظه بسیار زیادی مصرف می‌کند و همچنین تمام محاسبات مورد نیاز برای یک سلول کامل را برای یک سلول خالی در داده‌های خلوت نیز انجام می‌دهد و در نتیجه این روش از نظر زمان انجام محاسبات نیز بهینه نیست. با توجه به توضیحات ارائه شده به نظر می‌رسد که BUC از نظر زمان اجرا و مصرف حافظه از Multiway array aggregation بهینه‌تر است.

سوال ۴

با توجه به جدول داده شده، داده‌های مورد نظر شامل ۳ بُعد Date، Price و Product type and model می‌باشند و برای هر کدام به ترتیب ۴، ۴ و ۷ مقدار متفاوت داریم در نتیجه Cuboid دارای ۴*۴*۷ سلول مجزا می‌باشد. Cuboid پایه به شکل زیر می‌باشد:



شکل ۲

با استفاده از روش Multiway Array Function و Aggregate Function = Sum(count) و با کمک گرفتن از جدول شکل ۲ می‌توان Ancestorهای آن (Price/Date ، Price/Product type and model و Date/Product type and model) را به دست آورد. جداول به دست آمده به شکل زیر می‌باشند:

	500	650	1000	2000
Jan 2014	30	100	20	0
Nov 2013	0	50	30	0
May 2013	0	50	40	10
March 2012	0	0	10	0

Table 1:Price/Date

	500	650	1000	2000
Epson Printer	20	0	40	0
HP Printer	10	0	0	0
Canon Scanner	0	180	0	0
Dell Laptop	0	0	10	0
HP Laptop	0	0	50	0
Apple Laptop	0	0	0	10

Sony Printer	0	20	0	0

Table 2: Price/Product type and model

	March 2012	May 2013	Nov 2013	Jan 2014
Epson Printer	10	0	30	20
HP Printer	0	0	0	10
Canon Scanner	0	50	30	100
Dell Laptop	0	0	0	10
HP Laptop	0	40	0	10
Apple Laptop	0	10	0	0
Sony Printer	0	0	20	0

Table 3: Date/Product type and model

با استفاده از جدول‌های ۱ تا ۳ می‌توان جداول تک بعدی(Product type and model، Date و Price) را برای داده‌های داده شده محاسبه کرد. جداول سطح بعدی را به نحوی محاسبه می‌کنیم که کمترین مقدار محاسبات و حافظه مصرفی را داشته باشیم، به همین منظور جدول Price و Date با استفاده از جدول شماره ۱ و جدول Product type and model با استفاده از جدول ۲ یا ۳ (به دلیل برابری اندازه دو بُعد دیگر، از نظر تعداد محاسبات و حافظه مصرفی تفاوتی ندارند) ایجاد می‌شوند.

Date	March 2012	May 2013	Nov 2013	Jan 2014
Sum(count)	10	100	80	150

Table 4:Date

Price	500	650	1000	2000
Sum(count)	30	200	100	10

Table 5:Price

Product type and model	Sum(count)
Epson Printer	60
HP Printer	10
Canon Scanner	180
Dell Laptop	10
HP Laptop	50
Apple Laptop	10
Sony Printer	20

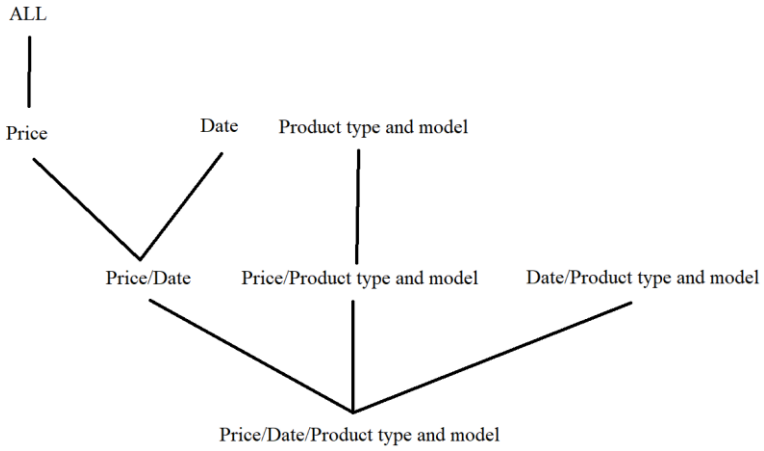
Table 6: Product type and model

در نهایت با استفاده از جداول بالا می‌توانیم Apex Cuboid را بسازیم که با توجه به صرفه جویی در محاسبات و مصرف حافظه، می‌توان از جدول ۴ یا ۵ استفاده کرد. Apex Cuboid به شکل زیر می‌باشد:

Sum(count)	340

Table 7:All

نحوه به دست آوردن Cuboidهای مختلف و سلسله مراتب آن‌ها در شکل ۳ نشان داده شده است:



شکل ۳

همانطور که دیده شد DataCube فوق یک DataCube خلوت(Sparse) محسوب می‌شود و بهترین راه برای ایجاد Cuboidهای مختلف آن استفاده از روش BUC می‌باشد تا توان استفاده از شروط Iceberg را داشته باشیم اما بنابر خواسته سوال از روش Multiway Array Aggregation استفاده شد.