دانشگاه تهران
دانشکده‌ی مهندسی کامپیوتر

تمرین دوم

داده کاوی

خانم دکتر شاکری

محمد امین عرب خراسانی
۸۱۰۱۰۲۲۰۵

بهار ۱۴۰۳

بخش تشریحی

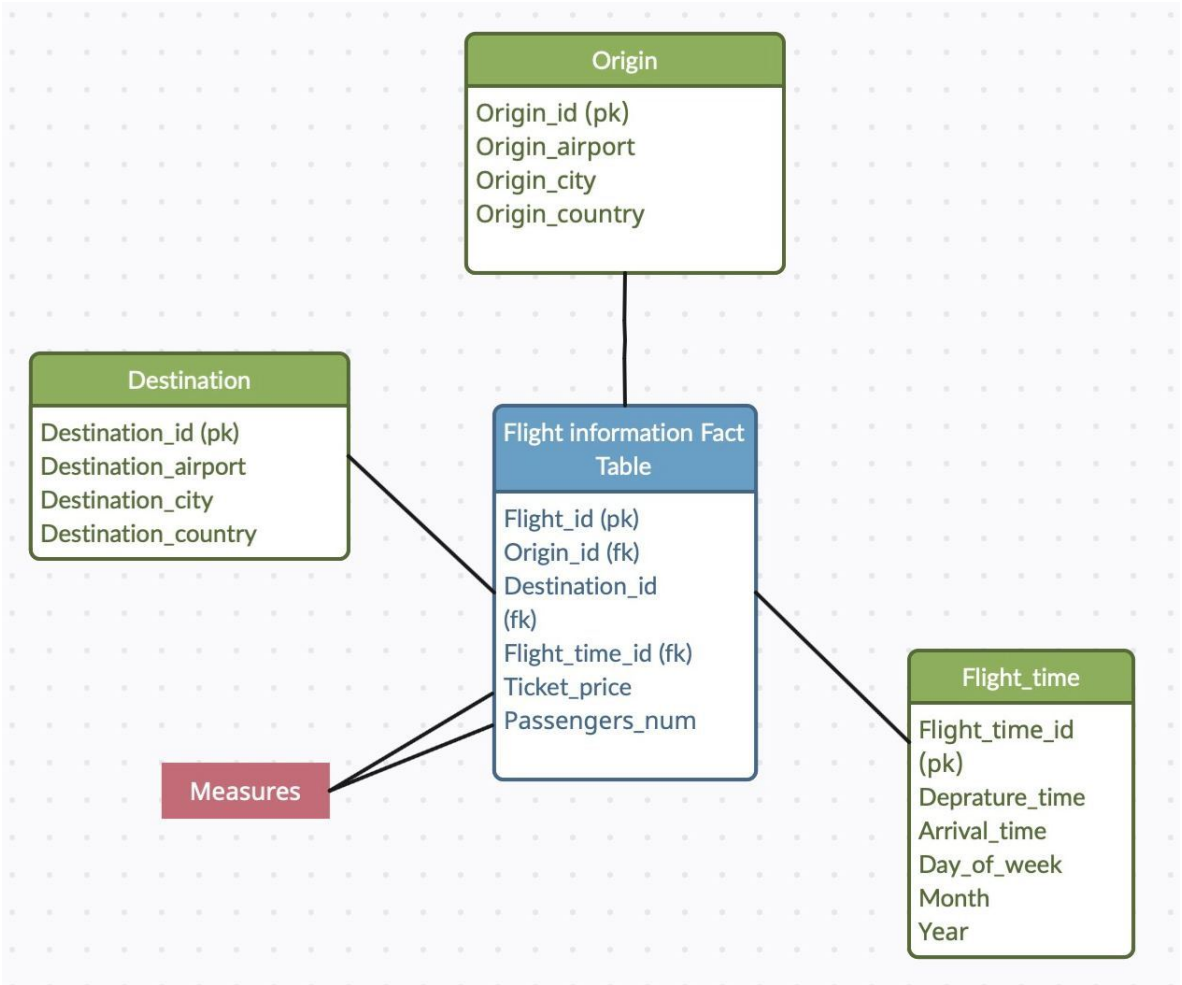
۱ سوال اول

الف) شمای ستاره‌ای در زمینهٔ ذخیره و سازماندهی داده‌ها به منظور تجزیه و تحلیل و گزارش‌دهی استفاده می‌شود. مزیت اصلی استفاده از شمای ستاره‌ای آسانی تجزیه و تحلیل داده‌هاست. در شمای ستاره‌ای، داده‌ها به دو دسته‌ی اصلی تقسیم می‌شوند:

۱) Fact table: این جدول حاوی اطلاعات اصلی است.

۲) Dimension tables: این جداول حاوی اطلاعات توصیفی در مورد ابعاد مختلف داده‌های موجود در Fact table هستند.

با توجه به ساختار اصلی شمای ستاره‌ای، برای ذخیره‌ی اطلاعات پروازهای خارجی این شما به شکل زیر می‌باشد.



ویژگی‌های (attributes) هر کدام از بعدها در Dimension tables خود آورده شده است.

همچنین لازم به ذکر است که Flight-id در Fact table به عنوان کلید اصلی در نظر گرفته شده است. بسته به نیازی که تصور می‌شود می توان آن را حذف کرد یا نگه داشت. اما به صورت کلی وجود یک متغیر منحصر به فرد برای ثبت و پرس و جوی داده‌ها مناسب است.

ب) برای این مقایسه هر کدام از میانگین‌ها محاسبه و در پایان با یکدیگر مقایسه شده است.
میانگین قیمت بلیط‌های تهران به میلان در خرداد ماه سال ۱۴۰۲:

Roll up on origin (from Origin-airport to Origin-city)
Roll up on Destination (from Destination-airport to Destination-city)
Roll up on Flight-time (from Departure-time to Month)
Dice for (Origin = “Tehran”) and (Destination = “Milan”)
Dice for (Flight-time = “Khordad”)
Roll up on Flight-time (from Month to Year)
Dice for (Flight-time = ۱۴۰۲)
Select AVG(Ticket-price)

میانگین قیمت بلیط‌های تهران به آمستردام در فروردین ماه سال ۱۴۰۱:

Roll up on origin (from Origin-airport to Origin-city)
Roll up on Destination (from Destination-airport to Destination-city)
Roll up on Flight-time (from Departure-time to Month)
Dice for (Origin = “Tehran”) and (Destination = “Amesterdam”)
Dice for (Flight-time = “Farvardin”)
Roll up on Flight-time (from Month to Year)
Dice for (Flight-time = ۱۴۰۱)
Select AVG(Ticket-price)

۲ سوال دوم

الف) برای پیمایش بهینه در chunk ها باید توجه داشت که صفحات بر اساس اندازه‌شان به ترتیب صعودی محاسبه می‌شوند. در این سوال صفحات ایجاد شده AB ، AC و BC هستند که از نظر اندازه‌ی صفحه‌ها به شکل زیر می‌باشند.

$$AC > AB > BC$$

با توجه به آن که BC کوچک‌تر از بقیه‌ی صفحات می‌باشد بنابراین پیمایش از این صفحه آغاز می‌شود و در ادامه در ستون بعدی در بعد A مجددا محاسبه خواهد شد.

ب) برای محاسبه‌ی کمترین فضایی که در حافظه‌ی اصلی نیاز است از بهینه‌ترین پیمایش در قسمت الف استفاده می‌شود. ابتدا تعداد معیارها محاسبه می‌شود. برای محاسبه‌ی تعداد آن‌ها، می‌بایست تعداد تمام cell های صفحه‌ی BC در کنار یک chunk از صفحه‌ی AC و یک ستون از صفحه‌ی AB انتخاب شود.

$$\text{memory} = ۴ * (۱۰۰ * ۱۰۰۰ + ۱۰۰ * ۱۰۰۰۰۰ + ۱۰۰ * ۱۰۰۰۰۰) = ۸۰۴۰۰۰۰۰ \text{ byte}$$

ج) بهترین روش برای محاسبه‌ی cuboid های یک بعدی با استفاده از cuboid های دو بعدی مانند روش قبل است به این صورت که کوچکترین cuboid های دو بعدی انتخاب می‌شوند (آن‌هایی که تعداد cell کمتری دارند). به همین منظور برای cuboid های یک بعدی A ، B و C بهترین حالت استفاده از cuboid های دو بعدی AB ، BC و BC است.

---

۳ سوال سوم

الف) با توجه به آن که cuboid های پایه‌ی a داده شده است، بعد i ام می‌تواند a_i یا b_i باشد. در این حالت تعداد cuboid ها برابر با ۲۹ می‌باشد.

ب) سلول‌های پایه‌ی غیر تهی aggregate با تعویض a یا b با * ایجاد می‌شوند. با توجه به

سلول اول و دوم برای هر کدام $۲^۹$ سلول پایه‌ی غیر تهی وجود دارد. اما ۲ تا از این سلول‌ها غیر تهی می‌باشند بنابراین تعداد سلول‌های aggregate غیر تهی برابر با مقدار زیر می‌باشد.

تعداد سلول‌های aggregate غیر تهی $= ۲^۹ - ۲$

ج) تعداد سلول‌های بسته‌ی غیر تهی یک cube data شامل سلول‌های cuboid پایه به علاوه‌ی سلولی که داده‌ها یا مقادیر درون آن، برای تمامی ابعاد مربوطه موجود هستند. در واقع برای این سوال سلول‌های بسته‌ی غیر تهی عبارتند از:

$(a_۱، a_۲، b_۳، a_۴، a_۵، b_۶، a_۷، a_۸، b_۹) : ۱۵$

$(b_۱، a_۲، a_۳، b_۴، a_۵، a_۶، b_۷، a_۸، a_۹) : ۱۰$

$(* ، a_۲ ، * ، * ، a_۵ ، * ، * ، a_۸، *) : ۲۵$

بنابراین با توجه به سلول‌های ارائه شده تعداد سلول‌های بسته‌ی غیر تهی برابر با ۳ می‌باشد.

د) با توجه به شرط minimum support = ۲۰ در این سوال تنها سلول زیر می‌تواند این شرط را ارضا کند.

$(* ، a_۲ ، * ، * ، a_۵ ، * ، * ، a_۸، *) : ۲۵$

که با توجه به تعداد مقادیر ثابت که می‌توانند یا خودشان باشند یا * باشند تعداد سلول‌های aggregate غیر تهی با شرط minimum support = ۲۰ برابر با ۲۳ می‌باشد.

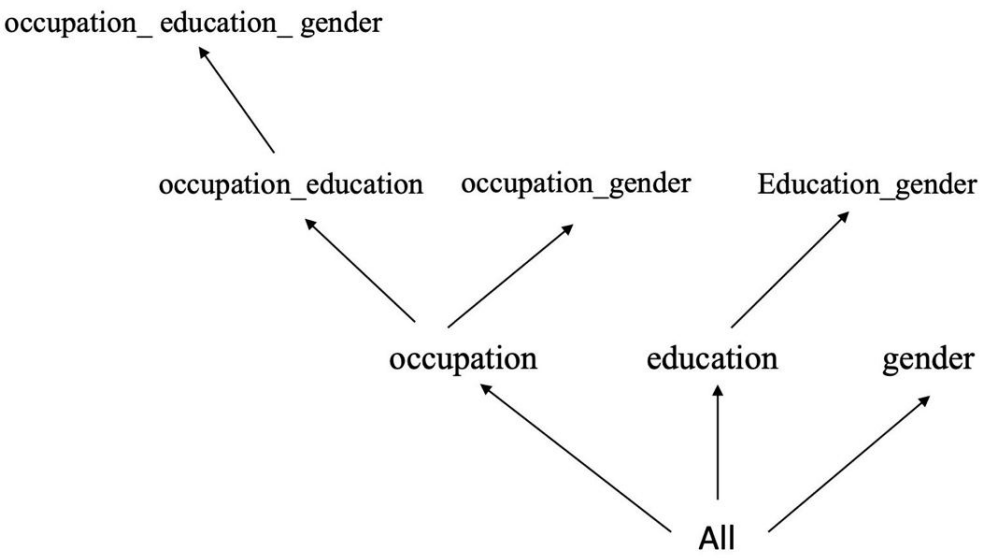
---

۴ سوال چهارم

الف) بهترین کارایی برای الگوریتم BUC زمانی است که از dimension با cardinality بالاتر پردازش شروع شود. این روش منجر به محاسبات کارآمد با استفاده از pruning بیشتر، تشخیص سریع‌تر سلول‌های خالی و کاهش مقدار مموری مصرف می‌باشد.
با توجه به روش ارائه شده، پردازش در این سوال از occupation شروع شده، با education

ادامه می‌یابد و در نهایت پردازش با بعد Gender به پایان می‌رسد. زیرا تعداد مقادیر متمایز در occupation برابر با ۴ ، در education برابر با ۳ و در Gender مقادیر متمایز برابر با ۲ می‌باشد.

ب) با توجه به تعریف بهترین روش برای الگوریتم BUC ، درخت پردازش برای این سوال رسم شده است. درخت پردازش به این نحو عمل می‌کند که اگر شرط minimum support = ۳ برای یک شاخه ارضا شود به زیرشاخه‌های دیگر نفوذ می‌کند در غیر این صورت از آن شاخه می‌گذرد و آن شاخه در درخت پردازش لحاظ نمی‌شود. شکل زیر این درخت را نشان می‌دهد.



در Iceberg cube شرط minimum support = ۳ برقرار است که به معنای آن است که یک cube باید حداقل ۳ عضو داشته باشد. با توجه به درخت رسم شده، پردازش از اول درخت، یعنی All شروع می‌شود. cube متناظر با این سطح (* ، * ، *) می‌باشد که همه‌ی تعداد مقادیر را شامل می‌شود و تعداد آن ۱۰ تا می‌باشد. سپس این الگوریتم به سراغ occupation می‌رود و شرط minimum support = ۳ را برای همه‌ی مقادیر این بعد بررسی می‌کند. نتیجه در زیر آورده شده است.

(programmer ، * ، *) : ۴ not prune
(teacher ، * ، *) : ۳ not prune
(CEO ، * ، *) : ۲ prune
(doctor ، * ، *) : ۱ prune

با توجه به مقادیر فوق (* ، * ، CEO) و (* ، * ، teacher) حذف یا prune می‌شوند.
این الگوریتم در ادامه برای هر کدام از cube های prune نشده مقادیر موجود را بررسی می‌کند.
با توجه به ترتیب انتخاب شده در بخش اول، بعد education بعد از بعد occupation قرار می‌گیرد.

(programmer ، college ، *) : ۳ not prune
(teacher ، college ، *) : ۲ prune
(programmer ، high school ، *) : ۱ prune
(teacher ، graduate ، *) : ۱ prune
با توجه به آن که (* ، programmer ، college) شرط را ارضا کرد و prune نشد همچنان این الگوربتم برای بعد gender نیز شرط را بررسی می‌کند.

(programmer ، college ، female) : ۲ prune
(programmer ، high school ، male) : ۱ prune

با توجه به نتیجه‌ی حاصل الگوریتم در این نقطه از شاخه‌ی مربوط به occupation-education- gender خارج شده و وارد occupation-gender می‌شود تا شرط Iceberg cube را بررسی کند. نتیجه به شکل زیر خواهد بود.

(programmer ، * ، female) : ۲ prune
(teacher ، * ، male) : ۲ prune
(teacher ، * ، female) : ۱ prune
(teacher ، * ، male) : ۱ prune
با توجه به prune شدن تمام cube ها و عدم وجود حالت دیگر در شاخه‌ی occupation این الگوریتم از این شاخه خارج می‌شود و وارد شاخه‌ی education می‌شود. نتیجه در زیر آورده شده است.

(* ، college ، *) : ۵ not prune
(* ، high school ، *) : ۳ not prune
(* ، graduate ، *) : ۲ prune
از آنجایی که ترکیب بعد occupation با education بررسی شده است در ادامه این شاخه بعد gender بررسی می‌شود. نتیجه به شکل زیر خواهد بود.

(* ، college ، female) : ۳ not prune

(* ، college ، male) : ۲ prune

(* ، high school ، male) : ۲ prune

(* ، high school ، female) : ۱ prune

از آن جایی که در قسمت قبل occupation-education-gender بررسی شده است الگوریتم وارد شاخه‌ی اصلی gender می‌شود. نتیجه‌ی بررسی این شرط در زیر آورده شده است.

(* ، * ، female) : ۵ not prune

(* ، * ، male) : ۵ not prune

در نتیجه Iceberg cube با شرط minimum support = ۳ به شکل زیر می‌باشد.

(* ، * ، *) : ۱۰

(* ، * ، female) : ۵

(* ، * ، male) : ۵

(* ، college ، *) : ۵

(programmer ، * ، *) : ۴

(teacher ، * ، *) : ۳

(programmer ، college ، *) : ۳

(* ، high school ، *) : ۳

(* ، college ، female) : ۳

بخش عملی

۱) این دیتاست شامل یک فایل csv است که شامل ۱۴۰۵۸۵ داده در ۲۳ ستون می‌باشد و حاوی اطلاعات آب و هوای شهرهای استرالیا می‌باشد. برخلاف تمرین اول این دیتاست دارای ۲۳ ستون با نام‌های درست است و نیازی به اصلاح نام ستون‌های دیتاست نیست. همچنین همه‌ی داده‌های دمایی به واحد سانتی‌گراد می‌باشد و نیازی به تبدیل مقادیر آن نیست. اما از آن جایی که برای

اطلاعات دمایی نیازی به دقت اعشار نیست، مقادیر مربوط به دما رند می‌شوند. به این ترتیب که اگر اعشار بیشتر از ۰.۵ باشد به بالا و در غیر این صورت به پایین گرد می‌شود.
دیتاست در یک دیتافریم ذخیره می‌شود. انواع این دیتا به کمک dtype گرفته می‌شود. نتیجه‌ی اجرای این کد در زیر آورده شده است.

```
Date             object
Location         object
MinTemp         float64
MaxTemp         float64
Rainfall        float64
Evaporation     float64
Sunshine        float64
WindGustDir      object
WindGustSpeed   float64
WindDir9am       object
WindDir3pm       object
WindSpeed9am    float64
WindSpeed3pm    float64
Humidity9am     float64
Humidity3pm     float64
Pressure9am     float64
Pressure3pm     float64
Cloud9am        float64
Cloud3pm        float64
Temp9am         float64
Temp3pm         float64
RainToday        object
RainTomorrow     object
dtype: object
```

همچنین با دستور ()info نتیجه‌ی زیر حاصل می‌شود.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 140585 entries, 0 to 140584
Data columns (total 23 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Date           140585 non-null  object
 1   Location       140584 non-null  object
 2   MinTemp        139184 non-null  float64
 3   MaxTemp        139371 non-null  float64
 4   Rainfall       137397 non-null  float64
 5   Evaporation    79490 non-null   float64
 6   Sunshine       73907 non-null   float64
 7   WindGustDir    130377 non-null  object
 8   WindGustSpeed  130436 non-null  float64
 9   WindDir9am     130137 non-null  object
 10  WindDir3pm     136409 non-null  object
 11  WindSpeed9am   138855 non-null  float64
 12  WindSpeed3pm   137559 non-null  float64
 13  Humidity9am    138026 non-null  float64
 14  Humidity3pm    136918 non-null  float64
 15  Pressure9am    125530 non-null  float64
 16  Pressure3pm    125566 non-null  float64
 17  Cloud9am       85968 non-null   float64
 18  Cloud3pm       83112 non-null   float64
 19  Temp9am        138874 non-null  float64
 20  Temp3pm        137704 non-null  float64
 21  RainToday      137397 non-null  object
 22  RainTomorrow   137394 non-null  object
dtypes: float64(16), object(7)
memory usage: 24.7+ MB
```

به منظور بهینه کردن حجم اشغال شده داده‌های اسمی از object به category تبدیل می‌شوند. نتیجه‌ی این تغییر پس از اجرای دستور ()info به شکل زیر می‌باشد.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 140585 entries, 0 to 140584
Data columns (total 23 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Date           140585 non-null  object
 1   Location       140584 non-null  category
 2   MinTemp        139184 non-null  float64
 3   MaxTemp        139371 non-null  float64
 4   Rainfall       137397 non-null  float64
 5   Evaporation    79490 non-null   float64
 6   Sunshine       73907 non-null   float64
 7   WindGustDir    130377 non-null  category
 8   WindGustSpeed  130436 non-null  float64
 9   WindDir9am     130137 non-null  category
 10  WindDir3pm     136409 non-null  category
 11  WindSpeed9am   138855 non-null  float64
 12  WindSpeed3pm   137559 non-null  float64
 13  Humidity9am    138026 non-null  float64
 14  Humidity3pm    136918 non-null  float64
 15  Pressure9am    125530 non-null  float64
 16  Pressure3pm    125566 non-null  float64
 17  Cloud9am       85968 non-null   float64
 18  Cloud3pm       83112 non-null   float64
 19  Temp9am        138874 non-null  float64
 20  Temp3pm        137704 non-null  float64
 21  RainToday      137397 non-null  category
 22  RainTomorrow   137394 non-null  category
dtypes: category(6), float64(16), object(1)
memory usage: 19.0+ MB
```

همچنین در ادامه با دستور dtype نوع داده‌ها پس از تغییر به category چک می‌شود.

```
Date              object
Location        category
MinTemp          float64
MaxTemp          float64
Rainfall         float64
Evaporation      float64
Sunshine         float64
WindGustDir     category
WindGustSpeed    float64
WindDir9am      category
WindDir3pm      category
WindSpeed9am     float64
WindSpeed3pm     float64
Humidity9am      float64
Humidity3pm      float64
Pressure9am      float64
Pressure3pm      float64
Cloud9am         float64
Cloud3pm         float64
Temp9am          float64
Temp3pm          float64
RainToday       category
RainTomorrow    category
dtype: object
```

در ادامه به کمک دو تابع sum و isna ابتدا دیتافریم مربوط به مقادیر NaN حاصل می‌شود و در ادامه تعداد آن‌ها برای هر ستون گزارش می‌شود.

```
Date                0
Location            1
MinTemp          1401
MaxTemp          1214
Rainfall         3188
Evaporation     61095
Sunshine        66678
WindGustDir     10208
WindGustSpeed   10149
WindDir9am      10448
WindDir3pm       4176
WindSpeed9am     1730
WindSpeed3pm     3026
Humidity9am      2559
Humidity3pm      3667
Pressure9am     15055
Pressure3pm     15019
Cloud9am        54617
Cloud3pm        57473
Temp9am          1711
Temp3pm          2881
RainToday        3188
RainTomorrow     3191
dtype: int64
```

برای ویژگی‌هایی که بیشتر از ۱۰۰۰۰ داده‌ی از دست رفته دارند، مقادیر از دست رفته با میانگین جایگزین می‌شوند و از ویژگی‌هایی که کمتر از این مقدار داده‌ی از دست رفته دارند صرف نظر شده و از دیتاست حذف می شوند. با توجه به این که ستون‌هایی که تعداد داده‌های از دست رفته‌ی زیادی

دارند زیاد است، حذف کردن ویژگی‌های دیگر همراه با این مقدار داده‌ی زیاد باعث از بین رفتن دیتاست می‌شود. اما با حذف ویژگی‌هایی که تعداد داده‌ی از دست رفته‌ی کمی دارند، مشکلی برای دیتاست ایجاد نمی‌شود.

همانطور که از گزارش مشخص است WindGustDir و WindDir۹am به عنوان یک داده‌ی categorical تعداد داده‌ی از دست رفته‌ی بیش از ۱۰۰۰۰ دارند. از آن جایی که برای داده‌ی cat-egorical میانگین تعریف نمی‌شود، این داده‌ها یا باید حذف شده و یا با mode جایگزین شوند. این امر به دلیل حفظ ارزش دیتاست انجام می‌شود.

در نهایت بعد از پیاده‌سازی استراتژی ذکر شده، دیتاست به از ۱۴۰۵۸۵ داده به ۱۳۰۷۰۳ داده تبدیل می‌شود.

همچنین برای شناسایی داده‌های پرت از IQR استفاده می‌شود. به این صورت که ابتدا ۱Q و ۳Q برای هر ستون در یک حلقه‌ی for روی ستون‌ها با شرط عددی بودن محاسبه می‌شود. با توجه به نوع دیتاست استفاده از ضریب ۱.۵ برای IQR برای شناسایی داده‌های پرت بخش زیادی از دیتا را حذف می‌کند. بنابراین این ضریب ۵ انتخاب می‌شود. در نهایت تعداد داده‌ها در این مرحله از ۱۳۰۷۰۳ به ۱۱۳۲۱۱ می‌رسد. گزارش زیر حاصل اجرای دستور ()info برای دیتافریم نهایی است.

```
<class 'pandas.core.frame.DataFrame'>
Index: 113211 entries, 0 to 140583
Data columns (total 23 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Date           113211 non-null  object
 1   Location       113211 non-null  category
 2   MinTemp        113211 non-null  float64
 3   MaxTemp        113211 non-null  float64
 4   Rainfall       113211 non-null  float64
 5   Evaporation    113211 non-null  float64
 6   Sunshine       113211 non-null  float64
 7   WindGustDir    113211 non-null  category
 8   WindGustSpeed  113211 non-null  float64
 9   WindDir9am     113211 non-null  category
 10  WindDir3pm     113211 non-null  category
 11  WindSpeed9am   113211 non-null  float64
 12  WindSpeed3pm   113211 non-null  float64
 13  Humidity9am    113211 non-null  float64
 14  Humidity3pm    113211 non-null  float64
 15  Pressure9am    113211 non-null  float64
 16  Pressure3pm    113211 non-null  float64
 17  Cloud9am       113211 non-null  float64
 18  Cloud3pm       113211 non-null  float64
 19  Temp9am        113211 non-null  float64
 20  Temp3pm        113211 non-null  float64
 21  RainToday      113211 non-null  category
 22  RainTomorrow   113211 non-null  category
dtypes: category(6), float64(16), object(1)
memory usage: 16.2+ MB
```

۲) الف) برای محاسبه‌ی میانگین بارش در استرالیا از ستون Rainfall استفاده می‌شود.

اگر فرض بر آن باشد که این میانگین بدون در نظر گرفتن بارش یا عدم بارش در یک روز مطرح

باشد این مقدار برابر با ۰.۳۹ میلی متر می‌باشد. اما در صورتی که هدف از این سوال میانگین بارش در روزهای بارانی باشد، این مقدار برابر با ۱.۳۸ میلی متر خواهد بود.

ب) ابتدا با استفاده از کتاب‌خانه‌ی datetime ستون Date به datetime تبدیل می‌شود. سپس دیتافریم بر اساس اسم Location که می‌بایست Watsonia باشد و سال ثبت دیتا که مربوط به ۲۰۱۵ می‌باشد فیلتر می‌شود. در ادامه به کمک ستون RainTomorrow تعداد داده‌هایی که برابر با yes می‌باشند محاسبه می‌شود. در نهایت تعداد روزهای بارانی شهر Watsonia در سال ۲۰۱۵ برابر با ۶۱ می‌باشد.

ج) برای به دست آوردن بیشترین رطوبت ثبت شده از دو ویژگی Humidity۹am و Humid-ity۳pm استفاده می‌شود. با توجه به آن که قضاوتی راجع به رطوبت در این ساعات نمی‌توان داشت بهتر است هر دوی آن‌ها بررسی شوند.
ابتدا دیتافریم بر اساس اسم شهر Townsville فیلتر می‌شود. در ادامه هر دو ویژگی Humid-ity۹am و Humidity۳pm با کمک تابع melt() در یک ستون با نام time قرار می‌گیرند. همچنین مقادیر آن‌ها در ستون humidity ثبت می‌شود. بیشترین مقدار رطوبت در ساعت ۹ صبح برابر با ۹۸ و در ساعت ۳ بعد از ظهر برابر ۹۶ می‌باشد. نتیجه‌ی مرتب شده‌ی ۵ داده‌ی اول این دیتافریم بر اساس تاریخ ثبت شده به شکل زیر می‌باشد.

	Date	time	humidity
1105	2012-07-14	Humidity9am	98.0
1332	2013-05-07	Humidity9am	97.0
5302	2017-01-31	Humidity3pm	96.0
5011	2016-03-08	Humidity3pm	96.0
97	2009-04-13	Humidity9am	95.0

د) در این سوال اختلاف میانه‌ها در هر سال محاسبه می‌شود. برای این منظور یک ستون جدید به نام Year به دیتافریم اصلی اضافه می‌شود. این ستون به کمک کتابخانه‌ی datetime حاصل می‌شود. در ادامه با فیلتر کردن دیتافریم اصلی بر اساس ماه ثبت دیتا (ژانویه) دیتافریم جدیدی ایجاد می‌شود که فقط مربوط به ماه ژانویه می‌باشد. در ادامه دیتافریمی ایجاد می‌شود که شامل

میانه‌ی ویژگی‌های MinTemp و MaxTemp در هر سال بوده و به آن اختلاف این دو مقدار اضافه می‌شود. دیتافریم نهایی به شکل زیر خواهد بود.

Year	MinTemp	MaxTemp	Difference
2008	15.0	30.0	15.0
2009	17.0	29.0	12.0
2010	17.0	29.0	12.0
2011	18.0	29.0	11.0
2012	17.0	28.0	11.0
2013	17.0	29.0	12.0
2014	17.0	29.0	12.0
2015	16.0	28.0	12.0
2016	17.0	29.0	12.0
2017	17.0	30.0	13.0

ه) در این مثال همانند مثال‌های قبل فیلترهای لازم بر اساس نام شهر و سه ماه اول سال اعمال می‌شود. سپس دیتافریم نهایی به صورت مرتب شده از کوچک به بزرگ مرتب می‌شود. دیتافریم زیر ۵ روز سرد سه ماه اول سال در شهر MountGinini را نشان می‌دهد.

	Date	Location	MinTemp
54976	2017-02-20	MountGinini	-2.0
55015	2017-03-31	MountGinini	-1.0
54280	2015-03-27	MountGinini	-1.0
52476	2010-01-19	MountGinini	-1.0
54574	2016-01-15	MountGinini	-0.0

۳) الف) از آن جایی که در این قسمت میانگین داده‌های قبلی محاسبه شده است با توجه به مقادیر و تعداد داده‌های جدید، میانگین جدید محاسبه می‌شود. به این ترتیب که تعداد و مجموع داده‌ی جدید با sum() و len() محاسبه می‌شود و با توجه به کد زیر، میانگین بارش آپدیت می‌شود.

ب) با اضافه شدن داده‌ی جدید به دیتاست موجود از آنجایی که محاسبه‌ی تعداد روزهای بارانی cube data از نوع distributive می‌باشد، می‌توان با محاسبه‌ی تعداد روزهای بارانی برای با شرایط خاص با تعداد روزهای بارانی‌ای که قبلا محاسبه شده است جمع زده شود.

ج) از آنجایی که محاسبه‌ی max() رطوبت ثبت شده یک cube data از نوع distributive می‌باشد، بنابراین max() داده‌های جدید می‌تواند با max() داده‌های قبلی مقایسه شود و بیشترین مقدار از بین آن دو به عنوان بیشترین رطوبت در نظر گرفته شود.

د) از آنجایی که محاسبه‌ی median یک در دسته‌بندی Holistic در یک cube data قرار می‌گیرد، بنابراین با اضافه شدن دیتای جدید برای محاسبه‌ی میانه‌ی دو ویژگی ذکر شده و در ادامه اختلاف آن‌ها نیاز به محاسبه‌ی median برای کل دیتاست جدید می‌باشد.

ه) در این مورد با توجع به آن که max() در دسته‌بندی distributive است، با محاسبه‌ی سردترین ۵ روز شهر MountGinini در دیتای جدید و مرتب‌سازی از سردترین به گرم‌ترین روز آن در کنار سردترین ۵ روز به دست آمده در دیتای قبلی، سردترین ۵ روز که ۵ داده‌ی اول است حاصل می‌شود.

برای این بخش تابع تمامی قسمت‌ها غیر از محاسبه‌ی میانه در فایل پایتون قرار داده شده است. فرض بر آن بوده است که دیتای اضافه شده از فیلترهای مورد انتظار مانند نام شهرها و یا تاریخ‌ها گذشته است.