**Results**

Goals of reports

Current report provides information about different aspects of sales and explores impact of various factors on final sales outcomes. Its main purpose is to help business (to cote one example, labels or music aggregators) to make right decisions related to researched dependencies, trends, etc.

Business tasks

**First business task*:*** for top 3 labels (by number of units sold within the entire timeframe), pick the performers (1 per label) who spent the greatest number of weeks in top-10. Consider the artists making more than 20% of label units.

*SQL query:*

**SELECT** parent\_label\_name,

performer\_name,

number\_of\_copies\_sold,

sales\_by\_label,

**TO\_char**(number\_of\_copies\_sold/sales\_by\_label \*100,'999.99')||'%' **AS** percent\_from\_label\_sales

**FROM**

(**SELECT** \*, **RANK** () **OVER** (**ORDER** **BY** sales\_by\_label **DESC**) **AS** lbl\_rnk

**FROM**

(**SELECT** l.parent\_label\_name,

p.performer\_name,

**sum**(s.copies\_sold) **AS** number\_of\_copies\_sold,

**sum**(**sum**(s.copies\_sold)) **OVER** (**PARTITION** **BY** l.parent\_label\_name) **AS** sales\_by\_label,

**count**(s.song\_id ) **AS** cnt,

**RANK** () **OVER** (**PARTITION** **BY** l.parent\_label\_name

**ORDER** **BY** **count**(s.song\_id) **DESC**) **AS** pfr\_within\_label\_rnk

**FROM** sales s

**JOIN** performers p **ON** p.performer\_id=s.performer\_id

**JOIN** songs sn **ON** s.song\_id=sn.song\_id

**JOIN** times t **ON** s.time\_id=t.time\_id

**JOIN** labels l **ON** l.label\_id=sn.song\_label\_id

**GROUP** **BY** l.parent\_label\_name, p.performer\_name) tab

**WHERE** number\_of\_copies\_sold/sales\_by\_label \* 100>20 **AND** pfr\_within\_label\_rnk<4

) foo

**WHERE** lbl\_rnk<4

**ORDER** **BY** 5;

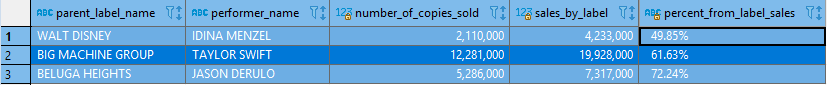
It is provided in project archive as SQL file as well.

*Logics of the query:* firstly, we define top performers within all labels (inner SELECT) who suits condition about 20%. Then, we use middle SELECT to “cut” top labels. In the highest level of query percentage is calculated.

*Window functions used:* ***sum()*** as a window function, ***rank()*** twice to define top within artists and within labels. **WINDOW** section is not provided as there are different definitions of **OVER** clauses.

*Performance:* the query performs ~14ms which is enough to accept it.

*Final dataset:*



*Visual representation:*

Since the key point is to compare individual sellings to labels ones, the most appropriate visualization type is bar chart.

***Second business task:***within specified timeframe, choose the songs of 3 those performers who works with labels made more than 5M copies sales for all period of dataset.

*SQL query:*

**WITH** sell\_art

**AS**

(**SELECT** performer\_id

**FROM**

(**SELECT** p.performer\_id,

**sum**(s.copies\_sold) **AS** sold,

**count**(p.performer\_id) **AS** cnt,

**sum**(**sum**(s.copies\_sold)) **OVER** (**PARTITION** **BY** l.parent\_label\_name) **AS** sales\_by\_label

**FROM** sales s

**JOIN** performers p **ON** p.performer\_id=s.performer\_id

**JOIN** songs sn **ON** s.song\_id=sn.song\_id

**JOIN** times t **ON** s.time\_id=t.time\_id

**JOIN** labels l **ON** l.label\_id=sn.song\_label\_id

**GROUP** **BY** l.parent\_label\_name, p.performer\_id) tab

**WHERE** sales\_by\_label > 5000000)

**SELECT** song\_id,

performer\_name, **max**(time\_one) **AS** first\_week, **max**(time\_two) **AS** second\_week, **max**(time\_three) **AS** third\_week,

**max**(time\_one)+**max**(time\_two)+**max**(time\_three) **AS** total

**FROM**

(

**SELECT** sn.song\_id, sn.song\_title, p.performer\_name,

**FIRST\_VALUE** (**sum**(s.copies\_sold)) **OVER** sum\_w **AS** time\_one,

**NTH\_VALUE** (**sum**(s.copies\_sold), 2) **OVER** sum\_w **AS** time\_two,

**LAST\_VALUE** (**sum**(s.copies\_sold)) **OVER** sum\_w **AS** time\_three

**FROM** sales s

**JOIN** performers p **ON** p.performer\_id=s.performer\_id

**JOIN** songs sn **ON** s.song\_id=sn.song\_id

**JOIN** times t **ON** s.time\_id=t.time\_id

**JOIN** labels l **ON** l.label\_id=sn.song\_label\_id

**WHERE** t.time\_id **IN** ('2014-10-05' , '2014-10-12','2014-10-19') **AND** sn.song\_type='SOLO'

**AND** p.performer\_id **IN** (**SELECT** \* **FROM** sell\_art)

**GROUP** **BY** t.time\_id, sn.song\_id, p.performer\_name

--#WINDOW SECTION FOR BETTER REEADABILITY

**WINDOW** sum\_w **AS** (**PARTITION** **BY** sn.song\_id **ORDER** **BY** t.time\_id

**ROWS** **BETWEEN** **UNBOUNDED** **PRECEDING** **AND** **UNBOUNDED** **FOLLOWING**)) tab

**GROUP** **BY** song\_id, performer\_name

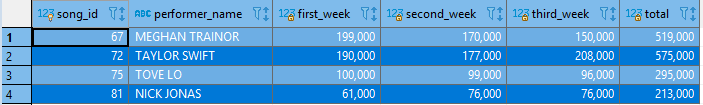
**HAVING** **max**(time\_one)+**max**(time\_two)+**max**(time\_three) **IS** **NOT** **NULL** ;

*Logics of the query:* condition related to labels sales copies is represented in *sell\_art* CTE. After defining required labels, we use window functions to get 1, 2 and 3 performers in the top within label. Then, we define max values of these figures. Filtering HAVING clause is used in case in any week there are no sales for a particular song.

*Window functions used: first\_value(), nth\_value(), last\_value()* to get corresponding tuples. **OVER** section is relocated to **WINDOW** clause as there are the same definitions.

*Performance:* the latest try took 19ms.

*Final dataset:*



*Visual representation:*

Since we need to understand comparison within elements against one parameter, horizontal bar chart is implemented.

***Third business task:*** define song producers whose songs (sum of copies) sold in 2nd half year are lower than in the 1st one. Illustrate their percentage as well. Consider specified months’ timeframe.

*SQL query:*

**SELECT** song\_producer, percentage

**FROM**

(**SELECT** sn.song\_producer,

t.calendar\_half\_year ,

**sum** (s.copies\_sold ) **AS** sales,

**lag**(**sum**(s.copies\_sold )) **OVER** lag\_w **AS** prev\_period,

((**lag**(**sum**(s.copies\_sold )) **OVER** lag\_w)-**sum** (s.copies\_sold ))\*100/**lag**(**sum**(s.copies\_sold ))

**OVER** lag\_w ||'%' **AS** percentage

**FROM** sales s

**JOIN** performers p **ON** p.performer\_id=s.performer\_id

**JOIN** songs sn **ON** s.song\_id=sn.song\_id

**JOIN** times t **ON** s.time\_id=t.time\_id

**JOIN** labels l **ON** l.label\_id=sn.song\_label\_id

**WHERE** t.calendar\_month\_desc **IN** ('2014-1','2014-3','2014-5','2014-8','2014-10','2014-12')

**GROUP** **BY** sn.song\_producer,

t.calendar\_half\_year

**WINDOW** lag\_w **AS** (**PARTITION** **BY** song\_producer **ORDER** **BY** t.calendar\_half\_year )) tab

**WHERE** prev\_period>sales

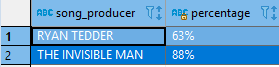
**ORDER** **BY** 1;

*Logics of the query:* firstly, producers, their sales and percentage are defined in subquery. Then, we use **WHERE** clause to get ones with decreased sales.

*Window functions used: lag()* for getting sales for previous period (1st half year). Window definition is put in corresponding section due to complexity of its logics.

*Performance:* the latest try took 6ms.

*Final dataset:*



*Visual representation:*

As there are only two elements to be compared, the bar chart is viable.