Analytics Mentor Course – Guided Project

/\* Change the brand name where the food\_id is 2 and make it Haagen-Dazs.\*/

update foods

set brand\_name = 'Haagen-Dazs'

where food\_id = 2

--checking to make sure the update is good to go.

select

\*

from

foods

where food\_id = 2

/\*

Find a count for all the items in the foods table

\*/

select

count(\*)

from

foods

/\*

Change the ordering of the columns -- make the last column the first

\*/

select

f.price\_last\_updated\_ts

, f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom

, f.brand\_name

, f.package\_price

from

foods f

/\*

Change the name package\_size\_uom to

something that is not as cryptic to users -- reordering columns

\*/

select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts

from

foods f

/\*

What foods are we working with today that are private label?

\*/

select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts

from

foods f

where

lower (brand\_name) = 'h-e-b (private label)'

-- using another way

select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts

from

foods f

where

brand\_name ilike 'H-E-B (private label)'

/\*

Is the item a canned item? And add a column that specifies yes/no.

\*/

select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts

, CASE when item\_name ilike '%canned%' then 'Yes' ELSE 'No' END as Is\_Canned\_YN

from

foods f

/\*

Find me any null values that may exist for the brand name

\*/

select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts

from

foods f

where brand\_name is null

/\*

how many records for each brand\_name? sort by ascending order.

\*/

select

count (\*)

, f.brand\_name

from

foods f

group by

f.brand\_name

order by

f.brand\_name asc

/\*

only give me the unique storage types

\*/

select

distinct f.storage\_type

from

foods f

/\*

one value is null...update that to unknown

\*/

update foods

set storage\_type = 'unknown'

where storage\_type is null

/\*

return all items with food\_id that equals 13, 15, 17 and H-E-B privtate label

\*/

select

\*

from

foods f

where

(

food\_id = 13

or food\_id = 15

or food\_id = 17

)

and brand\_name ilike 'h-e-b (private\_label)'

/\*

Find a count for brand name and storage type.

\*/

select

count(\*)

, brand\_name

, storage\_type

from

foods f

group by

brand\_name

, storage\_type

/\*

what percentage of the items are H-E-B (private label)

1.HEB private label items/Total

2.Do a count for all

\*/

select

cast(n.heb\_records as decimal (10,2)) / cast (d.total\_records as decimal (10,2))

from

( select

count (\*) as heb\_records

from

foods f

where

lower (brand\_name) = 'h-e-b (private label)'

)n -- numerator

cross join

(select

count (\*) as total\_records

from

foods f)d -- denominator

/\*

Find all items that havent had their price updated recently.

\*/

select

distinct f.price\_last\_updated\_ts

from

foods f

select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts at time zone 'EST' as price\_last\_updated\_est\_tz

, current\_timestamp

,(

current\_timestamp

- (f.price\_last\_updated\_ts at time zone 'EST')

) as days\_since\_price\_last\_updated

from

foods f

/\*

Try it with dates instead of timestamps.

\*/

select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts at time zone 'EST' as price\_last\_updated\_est\_tz

, current\_timestamp

,(

current\_timestamp

- (f.price\_last\_updated\_ts at time zone 'EST')

) as days\_since\_price\_last\_updated

, current\_date - cast(

(f.price\_last\_updated\_ts at time zone 'EST') as date

)

from

foods f

/\*

Return all items where days since last updated is greater than 30.

\*/

select

\*

from

( select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts at time zone 'EST' as price\_last\_updated\_est\_tz

, current\_timestamp

,(

current\_timestamp

- (f.price\_last\_updated\_ts at time zone 'EST')

) as days\_since\_price\_last\_updated

, current\_date - cast(

(f.price\_last\_updated\_ts at time zone 'EST') as date

) as days\_since\_last\_updated

from

foods f

) f --subquery

where

f.days\_since\_last\_updated > 30

/\*

Output all the foods and drinks in a result set.

\*/

select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts

from

foods f

union all

select

d.drink\_id

, d.item\_name

, d.storage\_type

, d.package\_size

, d.package\_size\_uom

, d.brand\_name

, d.package\_price d

, d.price\_last\_updated\_ts

from

drinks d

/\*

Identify where the data is coming from - what is the source table?

\*/

select

f.food\_id

, null as drink\_id -- null value f a food\_id is being returned

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts

, 'foods table' as source\_table

from

foods f

union all

select

d.drink\_id

, null as food\_id -- null value if a drink id is being returned

, d.item\_name

, d.storage\_type

, d.package\_size

, d.package\_size\_uom

, d.brand\_name

, d.package\_price d

, d.price\_last\_updated\_ts

, 'drinks table' as source\_table

from

drinks d

select

\*

from

food\_inventories

/\*

Find the latest inventory -- get the MAX date.

\*/

select

max(i.inventory\_dt) as max\_inventory\_date

from

food\_inventories i

/\*

Get the latest inventory information.

\*/

select

food\_inventory\_id

, food\_item\_id

, quantity

, inventory\_dt

from

food\_inventories i

where

inventory\_dt >=

( select

max(i.inventory\_dt) as max\_inventory\_date

from

food\_inventories i

)

select

f.food\_id

, f.item\_name

, f.storage\_type

, f.package\_size

, f.package\_size\_uom as package\_size\_unit\_of\_measurement

, f.brand\_name

, f.package\_price

, f.price\_last\_updated\_ts

, i.quantity as inventory\_quantity

from

foods f

left join (select

food\_inventory\_id

, food\_item\_id

, quantity

, inventory\_dt

from

food\_inventories i

where

inventory\_dt >=

( select

max(i.inventory\_dt) as max\_inventory\_date

from

food\_inventories i

)

)i

on f.food\_id = i.food\_item\_id