REPORT OF INTERSHIP PROGRAM 2024

On

“Toys Data Analysis Using SQL”

**Mentorness**



**Submitted by:**

M.Sivapriya

MIP-DA-14

# Toys Data Analysis Using SQL

**Market Overview**

* **Toy Industry Growth**: Analyze the overall market trends for the toy industry, including revenue growth, key drivers (e.g., technological advancements, eco-friendly toys), and emerging markets.
* **Consumer Spending**: Look at consumer spending patterns on toys, particularly during holidays or major events.
* **Demographic Breakdown**: Analyze the age groups, income levels, and geographic locations of your target market.

**Example Sources**:

* Statista (market size and revenue reports)
* IBISWorld (toy store industry analysis)
* NPD Group (global toy industry trends)

**2. Competitive Analysis**

* **Key Competitors**: Identify major players in the toy industry, such as big-box stores (Walmart, Target), online retailers (Amazon), and specialty stores.
* **Market Share**: Evaluate the market share of key competitors and how they position themselves.
* **Competitive Advantages**: Understand what differentiates your toy store (e.g., exclusive products, customer experience, unique branding).

**Example Sources**:

* Company reports and press releases (e.g., Hasbro, Mattel)
* Retail Dive (news and trends for retail stores)
* Toy News Online (updates on industry competition)

**3. Customer Behavior**

* **Shopping Habits**: Analyze where customers prefer to buy toys (physical stores vs. online) and the factors influencing their decisions (price, convenience, quality).
* **Seasonality**: Consider the seasonality of toy sales, particularly around major holidays like Christmas and Black Friday.
* **Product Trends**: Look into which types of toys are trending (e.g., educational toys, tech toys, action figures) and why.

**Example Sources**:

* Mintel (consumer behavior reports)
* Deloitte Insights (consumer spending behavior)
* Social media platforms (for real-time trends and consumer sentiment)

**4. Product Analysis**

* **Popular Toy Categories**: Break down sales by categories such as action figures, dolls, video games, educational toys, and outdoor toys.
* **Top-Selling Products**: Identify which products or brands perform the best within your store or industry-wide.
* **Product Innovation**: Understand the role of innovation in the toy industry, such as augmented reality toys, smart toys, or eco-friendly materials.

**Example Sources**:

* NPD Group (top toy categories and sales)
* Euromonitor (product trends and innovations)
* Toy Association (reports on toy safety, innovation, and new product launches)

**5. Sales Channels**

* **Online vs. Brick-and-Mortar**: Assess the differences in sales between online channels and physical stores, and determine how each affects your revenue.
* **Omnichannel Strategies**: Consider how omnichannel approaches (click-and-collect, in-store experiences, mobile apps) are reshaping the toy retail landscape.

**Example Sources**:

* eMarketer (retail and ecommerce data)
* Shopify (insights on ecommerce growth)
* McKinsey & Company (reports on omnichannel strategies)

**6. SWOT Analysis**

* **Strengths**: Unique selling points (e.g., personalized services, rare toys).
* **Weaknesses**: Areas to improve (e.g., limited online presence, pricing).
* **Opportunities**: Trends and growth opportunities (e.g., subscription-based toy services).
* **Threats**: Risks (e.g., competition from large retailers, supply chain issues).

**7. Regulatory Environment**

* **Toy Safety Standards**: Understand the regulations surrounding toy safety, materials, and labeling.
* **Environmental Concerns**: Look into regulations around the use of plastics, sustainability, and packaging.

**Example Sources**:

* U.S. Consumer Product Safety Commission (CPSC)
* European Toy Safety Directive
* Toy Association (regulatory updates)

## Source code:

-- four tables: inventory, products, sales, and stores

-- nulls from inventory

SELECT \*

FROM dbo.mt\_inventory$

WHERE Store\_ID IS NULL

OR Product\_ID IS NULL

OR Stock\_On\_Hand IS NULL

-- nulls from products

SELECT Product\_ID, Product\_Name, Product\_Category, Product\_Cost, Product\_Price

FROM dbo.mt\_products$

WHERE Product\_ID IS NULL

OR Product\_Name IS NULL

OR Product\_Category IS NULL

OR Product\_Cost IS NULL

OR Product\_Price IS NULL

-- nulls from sales

-- renaming date column first

SP\_RENAME 'dbo.mt\_sales$.Date', 'dateofsale'

SELECT Sale\_ID, dateofsale, Store\_ID, Product\_ID, Units

FROM dbo.mt\_sales$

WHERE Sale\_ID IS NULL

OR dateofsale IS NULL

OR Store\_ID IS NULL

OR Product\_ID IS NULL

OR Units IS NULL

-- nulls from stores

SELECT Store\_ID, Store\_Name, Store\_City, Store\_Location, Store\_Open\_Date

FROM mt\_stores$

WHERE Store\_ID IS NULL

OR Store\_Name IS NULL

OR Store\_City IS NULL

OR Store\_Location IS NULL

OR Store\_Open\_Date IS NULL

-- dupes from sales - 98,103 rows containing multiple instances in sales

SELECT dateofsale, Store\_ID, Product\_ID, Units,

COUNT(\*) AS occurrences

FROM dbo.mt\_sales$

GROUP BY dateofsale, Store\_ID, Product\_ID, Units

HAVING COUNT(\*) > 1

-- test - 829,262 total rows in sales

SELECT COUNT(\*)

FROM dbo.mt\_sales$

-- test: finding first iteration with MIN(Sale\_ID)

SELECT \* FROM dbo.mt\_sales$

WHERE Sale\_ID IN (

SELECT MIN(Sale\_ID)

FROM dbo.mt\_sales$

GROUP BY dateofsale, Store\_ID, Product\_ID, Units

)

-- implies that only 103,753 rows (12.5% of total sales) are not duplicates... we will assume that all sales are non-duplicates,

-- but this matter should be investigated further with more precise datesofsale and/or foreign keys

-- lower() column names in each table

-- inventory

SP\_RENAME 'dbo.mt\_inventory$.Store\_ID', 'store\_id'

SP\_RENAME 'dbo.mt\_inventory$.Product\_ID', 'product\_id'

SP\_RENAME 'dbo.mt\_inventory$.Stock\_On\_Hand','stock\_on\_hand'

-- products

SP\_RENAME 'dbo.mt\_products$.Product\_ID','product\_id'

SP\_RENAME 'dbo.mt\_products$.Product\_Name','product\_name'

SP\_RENAME 'dbo.mt\_products$.Product\_Category','product\_category'

SP\_RENAME 'dbo.mt\_products$.Product\_Cost','product\_cost'

SP\_RENAME 'dbo.mt\_products$.Product\_Price','product\_price'

-- sales

SP\_RENAME 'dbo.mt\_sales$.Sale\_ID','sale\_id'

SP\_RENAME 'dbo.mt\_sales$.Store\_ID','store\_id'

SP\_RENAME 'dbo.mt\_sales$.Product\_ID','product\_id'

SP\_RENAME 'dbo.mt\_sales$.Units','units'

-- stores

SP\_RENAME 'dbo.mt\_stores$.Store\_ID','store\_id'

SP\_RENAME 'dbo.mt\_stores$.Store\_Name','store\_name'

SP\_RENAME 'dbo.mt\_stores$.Store\_City','store\_city'

SP\_RENAME 'dbo.mt\_stores$.Store\_Location','store\_location'

SP\_RENAME 'dbo.mt\_stores$.Store\_Open\_Date','store\_open\_date'

-- inventory basic eda

SELECT store\_id,

COUNT(DISTINCT product\_id) AS prods,

ROUND((STDEV(stock\_on\_hand)),2) AS stock\_stdev,

ROUND((AVG(stock\_on\_hand)),2) AS avg\_stock,

MIN(stock\_on\_hand) AS min\_stock,

MAX(stock\_on\_hand) AS max\_stock

FROM dbo.mt\_inventory$

GROUP BY store\_id

ORDER BY store\_id

-- unique store\_id

SELECT DISTINCT(store\_id)

FROM dbo.mt\_inventory$

ORDER BY store\_id

-- unique prod\_id

SELECT DISTINCT(product\_id)

FROM dbo.mt\_inventory$

ORDER BY product\_id

-- products basic eda

SELECT COUNT(DISTINCT(product\_id)) AS unique\_prod\_ids,

COUNT(DISTINCT(product\_name)) AS unique\_prod\_names,

COUNT(DISTINCT(product\_category)) AS unique\_prod\_cats

FROM dbo.mt\_products$

-- creation of profit column within prods

SELECT \*,

ROUND((product\_price - product\_cost),2) AS profit

FROM dbo.mt\_products$

-- sales basic eda

SELECT COUNT(DISTINCT sale\_id) AS unique\_sales,

COUNT(DISTINCT dateofsale) AS unique\_dos,

COUNT(DISTINCT store\_id) AS unique\_stores,

COUNT(DISTINCT product\_id) AS unique\_prods,

COUNT(DISTINCT units) AS unique\_qtys

FROM dbo.mt\_sales$

-- beginning and ending of sales timeframe

SELECT MIN(dateofsale) AS beg\_of\_timeframe,

MAX(dateofsale) AS end\_of\_timeframe

FROM dbo.mt\_sales$

-- stores

SELECT \*

FROM dbo.mt\_stores$

-- stores basic eda

SELECT COUNT(DISTINCT store\_id) AS uniq\_store\_ids,

COUNT(DISTINCT store\_name) AS uniq\_store\_names,

COUNT(DISTINCT store\_city) AS uniq\_cities,

COUNT(DISTINCT store\_location) AS store\_types,

COUNT(DISTINCT store\_open\_date) AS uniq\_openings

FROM dbo.mt\_stores$

-- task 01: product categories driving profit

WITH prof AS

(

SELECT product\_id,

product\_category,

ROUND((product\_price - product\_cost),2) AS profits

FROM dbo.mt\_products$

),

sums AS

(

SELECT (SUM(s.units) \* (p.profits)) AS total\_profits,

p.product\_category

FROM prof p

JOIN mt\_sales$ s ON p.product\_id = s.product\_id

GROUP BY p.product\_category, p.profits

),

pcts AS

(

SELECT product\_category,

SUM(total\_profits) AS total\_profits,

(SUM(total\_profits) \* 100) / SUM(total\_profits) OVER() AS pct

FROM sums

GROUP BY product\_category, total\_profits

)

SELECT product\_category,

SUM(total\_profits) AS total\_profits,

ROUND((SUM(pct)),2) AS total\_pct

FROM pcts

GROUP BY product\_category

ORDER BY total\_pct DESC

-- is this true across store locations?

WITH prof AS

(

SELECT product\_id,

product\_category,

ROUND((product\_price - product\_cost),2) AS profits

FROM dbo.mt\_products$

),

sums AS

(

SELECT

(SUM(s.units) \* (p.profits)) AS total\_profits,

p.product\_category,

st.store\_name

FROM prof p

JOIN mt\_sales$ s ON p.product\_id = s.product\_id

JOIN mt\_stores$ st ON st.store\_id = s.store\_id

GROUP BY p.product\_category, p.profits, st.store\_name

),

rankings AS

(

SELECT product\_category,

ROW\_NUMBER () OVER (

PARTITION BY store\_name

ORDER BY total\_profits DESC) rn\_test,

store\_name,

SUM(total\_profits) AS total\_profits

FROM sums

GROUP BY store\_name, product\_category, total\_profits

)

SELECT store\_name,

product\_category,

total\_profits

FROM rankings

WHERE rn\_test = 1

ORDER BY total\_profits DESC

-- task 02: seasonality and/or trends

SELECT DAY(dateofsale) AS day\_of\_month,

COUNT(DISTINCT(sale\_id)) AS num\_sales

FROM dbo.mt\_sales$

GROUP BY DAY(dateofsale)

ORDER BY num\_sales DESC

-- we see that top 10 days for sales are within 5 days of the 15th and the 1st,

-- which is typically when people with biweekly pay receive their checks

SELECT DATENAME(dw,dateofsale) AS day\_of\_week,

COUNT(DISTINCT(sale\_id)) AS num\_sales

FROM dbo.mt\_sales$

GROUP BY DATENAME(dw,dateofsale)

ORDER BY num\_sales DESC

-- weekends have the most total unique sales

-- we can also check avg profit and avg sales for each day of week

WITH prof AS

(

SELECT product\_id,

ROUND((product\_price - product\_cost),2) AS profits

FROM dbo.mt\_products$

),

sums AS

(

SELECT s.dateofsale,

COUNT(DISTINCT(s.sale\_id)) AS num\_sales,

(SUM(s.units) \* (p.profits)) AS total\_profits

FROM prof p

JOIN mt\_sales$ s ON p.product\_id = s.product\_id

GROUP BY s.dateofsale,p.profits

)

SELECT DATENAME(dw,sums.dateofsale) AS day\_of\_week,

AVG(num\_sales) AS avg\_num\_of\_sales,

ROUND((AVG(total\_profits)),2) AS avg\_profit

FROM sums

JOIN mt\_sales$ s ON s.dateofsale = sums.dateofsale

GROUP BY DATENAME(dw,sums.dateofsale)

ORDER BY avg\_num\_of\_sales DESC

-- weekends are still on top for both avgs

-- let's check seasonality

-- sanity check

SELECT MIN(dateofsale) AS min\_d,

MAX(dateofsale) AS max\_d

FROM dbo.mt\_sales$

-- about 1.83 years of data according to timeframe

-- total num sales per season

WITH snss AS

(

SELECT CASE WHEN MONTH(dateofsale) IN(3, 4, 5) THEN 'spring'

WHEN MONTH(dateofsale) IN(6, 7, 8) THEN 'summer'

WHEN MONTH(dateofsale) IN(9, 10, 11) THEN 'autumn'

WHEN MONTH(dateofsale) IN(12, 1, 2) THEN 'winter' END AS season,

COUNT(DISTINCT(sale\_id)) AS num\_sales

FROM dbo.mt\_sales$

GROUP BY dateofsale

)

SELECT season,

SUM(num\_sales) AS total\_sales

FROM snss

GROUP BY season

ORDER BY total\_sales DESC

-- total sales per month/season

SELECT MONTH(dateofsale) AS mnth,

CASE WHEN MONTH(dateofsale) IN(3, 4, 5) THEN 'Spring'

WHEN MONTH(dateofsale) IN(6, 7, 8) THEN 'Summer'

WHEN MONTH(dateofsale) IN(9, 10, 11) THEN 'Autumn'

WHEN MONTH(dateofsale) IN(12, 1, 2) THEN 'Winter' END AS season,

COUNT(DISTINCT(sale\_id)) AS num\_sales

FROM dbo.mt\_sales$

GROUP BY MONTH(dateofsale)

ORDER BY num\_sales DESC

-- spring and summer are the most popular months for num unique sales

-- we should note that 10, 11, and 12 are at a disadvantage since the timeframe ends in September

-- task 03: out of stock holdups

WITH zeros AS

(

SELECT \*

FROM dbo.mt\_inventory$

WHERE stock\_on\_hand = 0

)

SELECT s.store\_name,

COUNT(DISTINCT(z.product\_id)) AS num\_products,

SUM(z.stock\_on\_hand) AS stock\_on\_hand

FROM dbo.mt\_stores$ s

JOIN dbo.mt\_inventory$ i ON s.store\_id = i.store\_id

JOIN zeros z ON z.store\_id = s.store\_id

GROUP BY s.store\_name

ORDER BY num\_products DESC

-- q1: which products have the lowest and highest sales?

SELECT TOP 5

p.product\_name,

COUNT(DISTINCT s.sale\_id) AS num\_sales,

SUM(s.units) AS total\_units

FROM dbo.mt\_products$ p

JOIN mt\_sales$ s ON s.product\_id = p.product\_id

GROUP BY p.product\_name

ORDER BY total\_units DESC

-- q1

SELECT TOP 5

p.product\_name,

COUNT(DISTINCT s.sale\_id) AS num\_sales,

SUM(s.units) AS total\_units

FROM dbo.mt\_products$ p

JOIN mt\_sales$ s ON s.product\_id = p.product\_id

GROUP BY p.product\_name

ORDER BY total\_units ASC

-- q2: are there any seasonal patterns in sales?

-- see task 02

-- q3: are there specific regions or stores performing exceptionally well or poorly?

SELECT DISTINCT store\_city

FROM dbo.mt\_stores$

-- inserting data into coords table for region eval

-- see map and insert regions into new col as well

-- store evaluation (top total sales)

SELECT TOP 5

st.store\_name AS store\_name,

st.regions,

COUNT(DISTINCT s.sale\_id) AS total\_sales

FROM dbo.mt\_stores$ st

JOIN dbo.mt\_sales$ s ON st.store\_id = s.store\_id

GROUP BY st.store\_name, st.regions

ORDER BY total\_sales DESC

-- store evaluation (bottom total sales)

SELECT TOP 5

st.store\_name,

st.regions,

COUNT(DISTINCT s.sale\_id) AS total\_sales

FROM dbo.mt\_stores$ st

JOIN dbo.mt\_sales$ s ON st.store\_id = s.store\_id

GROUP BY st.store\_name, st.regions

ORDER BY total\_sales ASC

-- store evaluation (total by month)

SELECT

st.store\_name,

COUNT(DISTINCT s.sale\_id) AS total\_sales,

MONTH(s.dateofsale) AS mnth

FROM dbo.mt\_stores$ st

JOIN dbo.mt\_sales$ s ON st.store\_id = s.store\_id

GROUP BY st.store\_name, MONTH(s.dateofsale)

ORDER BY mnth

-- q4: how often does the company experience stockouts?

SELECT COUNT(DISTINCT i.product\_id) AS prods,

DATEPART(wk, s.dateofsale) AS wk

FROM dbo.mt\_inventory$ i

JOIN dbo.mt\_sales$ s ON i.product\_id = s.product\_id

AND i.store\_id = s.store\_id

WHERE i.stock\_on\_hand = 0

GROUP BY DATEPART(wk, s.dateofsale)

ORDER BY wk

-- find weekly avg from this

WITH weekly\_sums AS

(

SELECT COUNT(DISTINCT i.product\_id) AS prods,

DATEPART(wk, s.dateofsale) AS wk

FROM dbo.mt\_inventory$ i

JOIN dbo.mt\_sales$ s ON i.product\_id = s.product\_id

AND i.store\_id = s.store\_id

WHERE i.stock\_on\_hand = 0

GROUP BY DATEPART(wk, s.dateofsale)

)

SELECT AVG(prods) AS avg\_prods

FROM weekly\_sums

-- 11 outstocks per week on avg across all locations

-- q5: which prods have the highest/lowest inventory turnover?

WITH weekly\_totals AS

(

SELECT

p.product\_name,

COUNT(DISTINCT s.sale\_id) AS sales,

SUM(s.units) AS total\_units,

DATEPART(wk, dateofsale) AS wek

FROM dbo.mt\_sales$ s

JOIN dbo.mt\_products$ p ON p.product\_id = s.product\_id

GROUP BY p.product\_name, DATEPART(wk, dateofsale)

)

SELECT TOP 5

product\_name,

AVG(sales) AS avg\_sales,

ROUND((AVG(total\_units)),2) AS avg\_total\_units

FROM weekly\_totals

GROUP BY product\_name

ORDER BY avg\_total\_units DESC

-- lowest weekly turnover

WITH weekly\_totals AS

(

SELECT

p.product\_name,

COUNT(DISTINCT s.sale\_id) AS sales,

SUM(s.units) AS total\_units,

DATEPART(wk, dateofsale) AS wek

FROM dbo.mt\_sales$ s

JOIN dbo.mt\_products$ p ON p.product\_id = s.product\_id

GROUP BY p.product\_name, DATEPART(wk, dateofsale)

)

SELECT TOP 5

product\_name,

AVG(sales) AS avg\_sales,

ROUND((AVG(total\_units)),2) AS avg\_total\_units

FROM weekly\_totals

GROUP BY product\_name

ORDER BY avg\_total\_units ASC

-- q6: slow or obselete stock

WITH slowest\_sellers AS

(

SELECT p.product\_name,

DATEPART(wk, s.dateofsale) AS wek

FROM mt\_products$ p

JOIN mt\_sales$ s ON p.product\_id = s.product\_id

GROUP BY p.product\_name, DATEPART(wk, s.dateofsale)

HAVING COUNT(DISTINCT s.sale\_id) < 100

)

SELECT TOP 5

product\_name,

COUNT(DISTINCT wek) AS num\_slow\_weeks

FROM slowest\_sellers

GROUP BY product\_name

ORDER BY num\_slow\_weeks DESC

-- q7: are there any geographical patterns in the sales data?

ALTER TABLE dbo.mt\_stores$

ADD regions AS (CASE

WHEN store\_city IN('La Paz', 'Mexicali') THEN 'Baja California'

WHEN store\_city IN('Hermosillo','Santiago','Chihuahua','Saltillo','Monterrey','Ciudad Victoria','Durango','Culican','Culiacan') THEN 'Northern Mexico'

WHEN store\_city IN('Tuxtla Gutierrez','Villahermosa','Merida','Campeche','Chetumal') THEN 'Yucatan Peninsula'

WHEN store\_city IN('Oaxaca','Chilpancingo','Chilpancigo','Morelia','Guadalajara') THEN 'Pacific Coast'

WHEN store\_city IN('Xalapa','Toluca','Pachuca','Cuernavaca','Ciudad de Mexico','Cuidad de Mexico','Puebla') THEN 'Central Mexico'

WHEN store\_city IN('Aguascalientes','Zacatecas','San Luis Potosi','Guanajuato') THEN 'The Bajio'

ELSE 'Invalid'

END);

-- sanity checks

SELECT \*

FROM mt\_stores$

WHERE regions = 'Invalid'

ALTER TABLE dbo.mt\_stores$

DROP COLUMN region,regional

SELECT \*

FROM mt\_stores$

ALTER TABLE dbo.mt\_stores$

ADD ste AS (CASE

WHEN store\_city = 'La Paz' THEN 'Baja California Sur'

WHEN store\_city = 'Mexicali' THEN 'Baja California'

WHEN store\_city = 'Hermosillo' THEN 'Sonora'

WHEN store\_city = 'Santiago' THEN 'Nuevo Leon'

WHEN store\_city = 'Chihuahua' THEN 'Chihuahua'

WHEN store\_city = 'Saltillo' THEN 'Coahuila'

WHEN store\_city = 'Monterrey' THEN 'Nuevo Leon'

WHEN store\_city = 'Ciudad Victoria' THEN 'Tamaulipus'

WHEN store\_city = 'Durango' THEN 'Durango'

WHEN store\_city IN('Culican','Culiacan') THEN 'Sinaloa'

WHEN store\_city = 'Tuxtla Gutierrez' THEN 'Chiapas'

WHEN store\_city = 'Villahermosa' THEN 'Tabasco'

WHEN store\_city = 'Merida' THEN 'Yucatan'

WHEN store\_city = 'Campeche' THEN 'Campeche'

WHEN store\_city = 'Chetumal' THEN 'Quintana Roo'

WHEN store\_city = 'Oaxaca' THEN 'Oaxaca'

WHEN store\_city IN('Chilpancingo','Chilpancigo') THEN 'Guerrero'

WHEN store\_city = 'Morelia' THEN 'Michoacan'

WHEN store\_city = 'Guadalajara' THEN 'Jalisco'

WHEN store\_city = 'Xalapa' THEN 'Veracruz'

WHEN store\_city = 'Toluca' THEN 'Estado de Mexico'

WHEN store\_city = 'Pachuca' THEN 'Hidalgo'

WHEN store\_city = 'Cuernavaca' THEN 'Morelos'

WHEN store\_city IN('Ciudad de Mexico', 'Cuidad de Mexico') THEN 'Ciudad de Mexico'

WHEN store\_city = 'Puebla' THEN 'Puebla'

WHEN store\_city = 'Aguascalientes' THEN 'Aguascalientes'

WHEN store\_city = 'Zacatecas' THEN 'Zacatecas'

WHEN store\_city = 'San Luis Potosi' THEN 'San Luis Potosi'

WHEN store\_city = 'Guanajuato' THEN 'Guanajuato'

ELSE 'Invalid'

END);

-- sanity checks

SELECT \*

FROM mt\_stores$

WHERE ste = 'Invalid'

ALTER TABLE mt\_stores$

DROP COLUMN stte

SELECT \*

FROM mt\_stores$

-- see tableau for more details

-- q8: how do different store locations contribute to overall sales?

WITH prof AS

(

SELECT product\_id,

product\_category,

ROUND((product\_price - product\_cost),2) AS profits

FROM dbo.mt\_products$

),

sums AS

(

SELECT

(SUM(s.units) \* (p.profits)) AS total\_profits,

st.store\_name

FROM prof p

JOIN mt\_sales$ s ON p.product\_id = s.product\_id

JOIN mt\_stores$ st ON st.store\_id = s.store\_id

GROUP BY st.store\_name, p.profits

),

percents AS

(

SELECT su.store\_name,

SUM(su.total\_profits) AS total\_profits,

(su.total\_profits \* 100 / SUM(su.total\_profits) OVER()) AS pct

FROM sums su

GROUP BY su.store\_name, total\_profits

)

SELECT

store\_name,

SUM(total\_profits) AS total\_profits,

ROUND((SUM(pct)),2) AS total\_pct

FROM percents

GROUP BY store\_name

ORDER by total\_pct DESC

-- q10: which product categories contribute the most to overall revenue?

-- see task 01

-- adding grouped data to mt\_stores for analysis within tableau

ALTER TABLE mt\_stores$

ADD total\_sales INT

-- adding new calculated columm

WITH salesdata AS

(

SELECT st.store\_id,

COUNT(DISTINCT s.sale\_id) AS total\_sales

FROM dbo.mt\_stores$ st

JOIN dbo.mt\_sales$ s ON st.store\_id = s.store\_id

GROUP BY st.store\_id

)

UPDATE st

SET st.total\_sales = sd.total\_sales

FROM dbo.mt\_stores$ st

JOIN salesdata sd ON st.store\_id = sd.store\_id

-- sanity check

SELECT \*

FROM mt\_stores$

ORDER BY total\_sales DESC

-- adding profits column

ALTER TABLE mt\_stores$

ADD total\_profit INT

-- adding data to calc column total\_profit

WITH prof AS

(

SELECT product\_id,

product\_category,

ROUND((product\_price - product\_cost),2) AS profits

FROM dbo.mt\_products$

),

sums AS

(

SELECT

(SUM(s.units) \* (p.profits)) AS total\_profits,

st.store\_name

FROM prof p

JOIN mt\_sales$ s ON p.product\_id = s.product\_id

JOIN mt\_stores$ st ON st.store\_id = s.store\_id

GROUP BY st.store\_name, p.profits

),

percents AS

(

SELECT su.store\_name,

SUM(su.total\_profits) AS total\_profits,

(su.total\_profits \* 100 / SUM(su.total\_profits) OVER()) AS pct

FROM sums su

GROUP BY su.store\_name, total\_profits

),

final AS

(

SELECT

store\_name,

SUM(total\_profits) AS total\_profits,

ROUND((SUM(pct)),2) AS total\_pct

FROM percents

GROUP BY store\_name

)

UPDATE st

SET st.total\_profit = f.total\_profits

FROM dbo.mt\_stores$ st

JOIN final f ON st.store\_name = f.store\_name

-- sanity check

SELECT \*

FROM mt\_stores$

ORDER BY total\_sales DESC