Week 4
1/5

Page No.
Date 17/02/23

## ⊛ Data Calculations :-

The better you get at SQL and pulling data from data tables, the faster you'll complete your analysis.

This week :-

- > Formula for basic calculations
- > Conditional formulas that use the IF function
- > The SUMPRODUCT function
- > Pivot tables to organize calculations.
- > Queries & calculations in SQL
- > Temporary tables in SQL.

---

## * Common Calculation Formulas :-

=Sum( B2:B12)

= MAX

= MIN

---

## * Functions

COUNTIF (range, "value")

Summary table :-

A table used to summarise statistical information about data.

SUMIF( range, cond1, SumRange)

| F | G | H | I |
|---|---|---|---|
| Quantity | Count | Revenue Total | Average revenue per transaction |
| 1 | 25 | | |
| >1 | | | |

G11 ⇒ COUNTIF (B3:B50, "=1")

G12 ⇒ COUNTIF (B3:B50, ">1")

H11 ⇒ SUMIF (B3:B50, "=1", C3:C50)

H12 ⇒ SUMIF (B3:B50, ">1", C3:C50)

I11 ⇒ H11/G11

I12 ⇒ H12/G12

* COUNTIFS :→ (both True)

= COUNTIFS ( range1, cond1, range2, cond2, ... )

* SUMIFS :→

= SUMIFS ( sum-Range1, Criteria1, Criteria2, cond2, ... )

= SUMIFS ( B2:B22, E12:E20, "=1", D10:D15, "=2")

* MAXIFS ( maxrange, range1, Criteria1, range2, Criteria2 ... )

MAXIFS ( B2:B32, D2:D10, "=Apple", E2:E10, "=purchased")

finds maximum quantity of Apples purchased

* Composite functions :→

SUMPRODUCT :-

A function that multiplies arrays and returns the sum of those products.

=sumproduct (array1, array2, ...)

| B | C | | |
|---|---|---|---|
| Quantity | Price per item | Profit margin | Total = $B_1C_1 + B_2C_2 + \ldots$ |

∴ =sumproduct(B3:B7, C3:C7) = $655

→

Profit margin :-

A percentage that indicates how many cents of profit has been generated for each dollar of sale.

Apple sold at 5$ : Price per item  Profit 1$ : Profit margin  Profit margin 20%

quantity

=Sumproduct (B3:B7, C3:C7, D3:D7)

= Total profit

individually multiplies  B3×C3×D3 +
B4×C4×D4 +
B5×C5×D5 +
B6×C6×D6
B7×C7×D7

Total Profit =

Week 4
2/5

# Using Pivot table

(#) Movie dataset analysis :-

Analysis steps :-

- Find out how much revenue was generated each year.
- Build a pivot table to show the revenue per year.
- Find the average revenue per movie.
- Checkout findings for some possible trends.

Insert > create Pivot Table

Rows > Add Release Date
But there are too many dates
Right click on any cell > "group by year"
Now only 4 years

Values > Sum of Box office collections

The drop-down is used to change functions applied to the values.

Add more to values > Average of Box office Collections

Add to values > COUNT of Box office Revenues
i.e. no. of movies released each year.

Year 2015 has highest no. of releases but lowest all around Box office Collection.

maybe the movies didn't earn much, etc.

To test our hypothesis that "may-be movies didn't earn that much"

↓

first copy the pivot table and paste it somewhere (maybe in same sheet)

↓

Change the column names

Sum<$10M  Average<$10M  COUNT<$10M

* next we are going to use filters to find out how many movies earned less than 10$ M revenue in 2015

Then ⊛ Create a calculated field to determine what percentage of total movies from that year they represent.

---

Applying filter :-

Select cols & add filter, it'll be applied to the entire table

Filter by condition: Less than
10,000,000.00

there are 20 movies that earned < 20$M but not enough to be called an insight

'for_spaces in names' in Sheet1

Page No.
Date

Create a new column "Calculated Field"

A calculated field is a new field within a pivot table that carries out certain calculations based on the values of other fields.

formula :→
Sum('Box office Revenue ($)')/COUNT(BOR)

Summarize by → [custom]

Show as → [default]

percent of total movies
= D11/D2 [<10M$ / Total movie]

& make it percentage

We came to know, 16.13% movies in 2013 earned <10M$
while other years have this number around 10% only.

→

We covered : function, formulas & Pivot Tables.

Week 4
3/5

Page No.
Date 20 02 23

## Learn more SQL calculations

Operators
A symbol that names the type of operation or calculation to be performed in a formula.

+ addition
- subtraction
* multiplication
/ division

} In Both SQL & Sheets

```
SELECT Col A, Col B, Col A + Col B AS Col X
FROM T_name;
```

For multiple calculation use parenthesis :-

```
SELECT Col A, Col B, Col C,
       (Col A + Col B) * Col C AS col Y
FROM t_name;
```

* % Modulo :- Returns remainder when one num. divided by another

| Spreadsheet function | SQL function |
|---|---|
| Sum | Sum |
| AVERAGE | AVG |

[<> != ] NOT Equal

Embedding Simple calculation in SQL

```
SELECT Date, Region, Small_Bags, Large_Bags, XLarge_Bags,
       Total_Bags, Small_Bags + Large_Bags + XLarge_Bags
                AS Total_Bags_Calc
FROM avocado_data, avocado_prices.
```

Q. what percentage bags were Small_Bags, Large_Bags and XLarge_Bags

```
SELECT Date, Region, Total_Bags, Small_Bags,
       Large_Bags, XLarge_Bages,
       (Small_Bags / Total_Bags) * 100 AS SmallPercent,
       (Large_Bags / Total_Bags) * 100 AS Large_Percent,
       (XLarge_Bags / Total_Bags) * 100 AS XLarge_Percent
FROM avocado_data. avocado_prices;
```

mutiplying by 100 gives us Percentage

Error → You can't divide by zero

i.e. somewhere, total_bags = 0
so find it & fix it

include a WHERE

```
WHERE 0 <> ALL (small, large, XLarge);
```

You can also use SAFE_DIVIDE

Extract ( YEAR FROM Col)

ORDER BY ASC/DESC

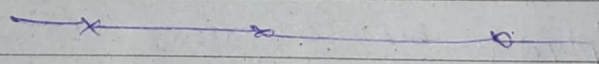
Group By :

A command that groups rows that have the same values from a table into summary rows.

```
SELECT
FROM
WHERE
GROUP BY
```

* Extracts :-
Lets us pull one part of date

```
SELECT EXTRACT (YEAR FROM STARTTIME) AS Year,
       COUNT (*) AS NO-of-rides
FROM
       BigQ-pub-data. new-york . citibik_trips
GROUP BY Year
ORDER BY Year
```



Data Validation :→

> ADDING DROP-DOWN to cells.

Checking and rechecking the quality of your data so that it is complete, accurate, secure and consistent.

* Types of Data validation :→

| | | |
|---|---|---|
| ① | Data Types | Str, num |
| ② | Data Range | 1-12 |
| ③ | Data Constraints | must be whole numbers. |
| ④ | Data Consistencies | final date can't be before initial date |
| ⑤ | Data Structure | music in MP3 & not HTML. |
| ⑥ | Code validation | Checking system perform previously mentioned validations systematically. |

⑦ Before you upload files to SQL, you can import them into a spreadsheet in sheets to get comfortable with the data before you start.

⑧ This might not be possible with large datasets, but we should explore as much as possible.

6 Steps of Data Analytics

Ask
Prepare
Process
Analyze
Share
Act

### Description in BigQuery

→ makes it less complex & easier to read.

## * Temporary Tables in SQL :-

Temporary table :-
A database table that is created and exists temporarily on a database server.

> Usually used to store subsets of data from standard data tables for certain period & they're automatically deleted, when you end session!

Ex. Running a query and storing result.

(+) A dAnalyst has a large number of records in a table. They want to perform calculations on a small subset of the table. Rather than filtering data over and over

They should use a "Temporary Table"

---

The WITH clause is a type of temporary table that you can query from multiple times.

Ex.

Temp created → WITH tempname AS ( SELECT * FROM old_Table
WHERE Cond^n >= 60)

Temp used → SELECT COUNT(*) AS CO FROM temp_name;

There's always more than one way!

WITH (3)
SELECT INTO
CREATE TABLE
(another syntax) CREATE TEMP TABLE

* SELECT INTO :→
copies data from existing table to new table

> Useful if you wanna make copy with a specific condition
like query with a WHERE clause

Ex:→
```
SELECT *
INTO Africasales
FROM GlobalSales
WHERE Region = "Africa" ;
```

* Good practice when you want to keep Database uncluttered & you don't need other people using the table

* If lots of people use it, then CREATE TABLE is a better option.
Adds table into database.

```
CREATE TABLE Africa_Sale AS (
    SELECT * FROM GlobalSales
    WHERE Region = 'Africa')
```

Adding metadata :→ to describe the data