

ELEVATE LABS TASK 3

Task 3: SQL for Data Analysis

1) Avatar group with highest average membership length and spending

```
SELECT Avatar, AVG(Length_of_Membership) AS Avg_Membership,  
AVG(Yearly_Amount_Spent) AS Avg_Spending
```

```
FROM clean_data
```

```
GROUP BY Avatar
```

```
ORDER BY Avg_Spending DESC, Avg_Membership DESC
```

```
LIMIT 1;
```

The screenshot shows a SQL query editor window titled 'Query 1'. The query is as follows:

```
1 -- 1. Avatar group with highest average membership length and spending
2 • SELECT Avatar, AVG(Length_of_Membership) AS Avg_Membership, AVG(Yearly_Amount_Spent) AS Avg_Spending
3 FROM clean_data
4 GROUP BY Avatar
5 ORDER BY Avg_Spending DESC, Avg_Membership DESC
6 LIMIT 1;
```

Below the query editor, the 'Result Grid' is displayed. It shows a single row of results:

Avatar	Avg_Membership	Avg_Spending
DarkGray	5.140000000000001	633.135

The interface includes various toolbars for query execution, filtering, and exporting. The 'Result Grid' tab is active, and the 'Read Only' status is indicated at the bottom right.

2) Spending efficiency: Yearly Amount Spent per hour on App

```
SELECT Email, (Yearly_Amount_Spent / Time_on_App) AS Spending_per_App_Hour
```

```
FROM clean_data
```

```
WHERE Time_on_App > 0
```

```
ORDER BY Spending_per_App_Hour DESC
```

LIMIT 10;

The screenshot shows a SQL query editor window titled "Query 1" with a toolbar at the top. The query text is as follows:

```
1  -- 2. Spending efficiency: Yearly Amount Spent per hour on App
2  SELECT Email, (Yearly_Amount_Spent / Time_on_App) AS Spending_per_App_Hour
3  FROM clean_data
4  WHERE Time_on_App > 0
5  ORDER BY Spending_per_App_Hour DESC
6  LIMIT 10;
```

Below the query editor is a "Result Grid" window showing the results of the query. It has a toolbar with options like "Filter Rows", "Export", "Wrap Cell Content", and "Fetch rows". The results are displayed in a table with two columns: "Email" and "Spending_per_App_Hour".

Email	Spending_per_App_Hour
waltonkaren@gmail.com	61.52341597796143
asilva@yahoo.com	60.95167895167895
david80@knight.com	56.11392405063291
rhonda01@gmail.com	55.690494893951296
alicia85@lee.com	55.218724778046806
edwardbrown@yahoo.com	54.67874794069192
youngbarbara@yahoo.com	54.37703646237393
millerrachel@gallagher-nichols.com	53.64345403899722
alvareznancy@lucas.biz	52.9592684954281
floresarthur@yahoo.com	52.77939646201873

At the bottom of the results grid, it says "Result 20" and "Read Only".

3) Rank customers by combined time on app and website (most engaged)

SELECT Email, (Time_on_App + Time_on_Website) AS Total_Time_Spent

FROM clean_data

ORDER BY Total_Time_Spent DESC

LIMIT 10;

Query 1 x

Limit to 50000 rows

```
1 -- 3. Rank customers by combined time on app and website (most engaged)
2 SELECT Email, (Time_on_App + Time_on_Website) AS Total_Time_Spent
3 FROM clean_data
4 ORDER BY Total_Time_Spent DESC
5 LIMIT 10;
```

Result Grid

Email	Total_Time_Spent
pcline@hotmail.com	53.11
russellbaldwin@ferrell.info	53.06
gregoryholmes@hotmail.com	53.01
alejandro75@hotmail.com	52.96
randall85@williams.com	52.6600000000000004
susanibarra@yahoo.com	52.5299999999999994
michael86@yahoo.com	52.4800000000000004
kyang@diaz.org	52.29
kylemelendez@hotmail.com	52.27
mstephenson@fernandez.com	52.2399999999999995

Result 21 x Read Only

4) Customers with membership longer than average but spending below average

SELECT Email, Length_of_Membership, Yearly_Amount_Spent

FROM clean_data

WHERE Length_of_Membership > (SELECT AVG(Length_of_Membership) FROM clean_data)

AND Yearly_Amount_Spent < (SELECT AVG(Yearly_Amount_Spent) FROM clean_data);

Query 1 x

Limit to 50000 rows

```

1  -- 4. Customers with membership longer than average but spending below average
2  SELECT Email, Length_of_Membership, Yearly_Amount_Spent
3  FROM clean_data
4  WHERE Length_of_Membership > (SELECT AVG(Length_of_Membership) FROM clean_data)
5  AND Yearly_Amount_Spent < (SELECT AVG(Yearly_Amount_Spent) FROM clean_data);

```

Result Grid | Filter Rows: | Export: | Wrap Cell Contents: |

Email	Length_of_Membership	Yearly_Amount_Spent
brenda25@gmail.com	3.67	465.18
oolson@colins.com	3.62	485.92
michellejohnson@sanders-rodri...	3.81	443.97
melissa08@nelson.com	3.85	496.55
longphillip@yahoo.com	4.84	496.93
patriciajackson@jenkins.info	3.8	463.75
hallthomas@lane.com	3.66	490.6
wellsjuan@schroeder.com	3.76	446.42
tammyjones@griffin.com	3.81	423.31
zachary94@gmail.com	3.59	463.59
johnsonhannah@gmail.com	3.77	473.95
suzanne63@gmail.com	4.09	478.26
mitchellscott@gmail.com	4.02	473.36
dbenson@simpson.net	4.08	442.06
pamelahampton@martin-cobb....	3.87	486.08
robertamirez@kaiser.com	3.54	461.79
tanya20@gmail.com	3.62	430.59
aaron89@gmail.com	4.24	418.6
randywhite@armstrong.biz	3.54	486.16
brandonsmith@yahoo.com	3.56	474.53
stevenjohnson@yahoo.com	3.78	467.8
elizabethnunez@vasquez-nelso...	4.79	493.18
ashleymoreno@gmail.com	3.98	475.73
wesleyvance@moore.com	3.7	483.54
dana59@hotmail.com	3.86	461.11

clean_data 22 x

Read Only

5) Percentage contribution of each avatar to total spending

SELECT Avatar,

SUM(Yearly_Amount_Spent) AS Total_Spending,

ROUND(SUM(Yearly_Amount_Spent) * 100 / (SELECT SUM(Yearly_Amount_Spent)
FROM clean_data), 2) AS Percent_Contribution

FROM clean_data

GROUP BY Avatar

ORDER BY Percent_Contribution DESC;

Query 1 x

```

1  -- 5. Percentage contribution of each avatar to total spending
2  SELECT Avatar,
3         SUM(Yearly_Amount_Spent) AS Total_Spending,
4         ROUND(SUM(Yearly_Amount_Spent) * 100 / (SELECT SUM(Yearly_Amount_Spent) FROM clean_data), 2) AS Percent_Contribution
5  FROM clean_data
6  GROUP BY Avatar
7  ORDER BY Percent_Contribution DESC;

```

Result Grid

Avatar	Total_Spending	Percent_Contribution
Orchid	2634.69	1.06
SaddleBrown	2613.04	1.05
MediumBlue	2612.95	1.05
BlanchedAlmond	2624.5200000000004	1.05
White	2580.14	1.03
DarkMagenta	2512.02	1.01
Purple	2522.7000000000003	1.01
LightGreen	2481.6299999999997	0.99
DarkGreen	2447.44	0.98
Olive	2446.02	0.98
LemonChiffon	2391.6800000000003	0.96
LightSalmon	2389.5299999999997	0.96
MediumSpring...	2405.06	0.96

Result 23 x

Read Only

6) Customers spending above average but with short membership (new high spenders)

SELECT Email, Length_of_Membership, Yearly_Amount_Spent

FROM clean_data

WHERE Yearly_Amount_Spent > (SELECT AVG(Yearly_Amount_Spent) FROM clean_data)

AND Length_of_Membership < (SELECT AVG(Length_of_Membership) FROM clean_data)

ORDER BY Yearly_Amount_Spent DESC;

Query 1 x

```

1  -- 6. Customers spending above average but with short membership (new high spenders)
2  SELECT Email, Length_of_Membership, Yearly_Amount_Spent
3  FROM clean_data
4  WHERE Yearly_Amount_Spent > (SELECT AVG(Yearly_Amount_Spent) FROM clean_data)
5  AND Length_of_Membership < (SELECT AVG(Length_of_Membership) FROM clean_data)
6  ORDER BY Yearly_Amount_Spent DESC;
7

```

Result Grid

Email	Length_of_Membership	Yearly_Amount_Spent
leonardhancock@hotmail.com	3.45	510.5
brian28@sanchez.org	3.16	510.4
david47@hotmail.com	3.48	508.77
walkererica@scott.com	3.02	506.54
bushsharon@barber.com	3.45	506.38
davischristina@hotmail.com	2.96	506.13
linda90@yoder.org	3.35	505.77
jessica04@christian-riley.com	1.92	504.87
aaron11@luna.com	2.94	503.98
phillipskarrie@gmail.com	3.37	503.22
rcarter@crane-thompson.org	3.34	503.18
deborah48@white.com	3.38	502.13
alexandermichael@hotmail.com	3.47	501.93
vancealicia@hotmail.com	2.96	501.12
cunninghamkyle@ellis-parker.org	3.44	501.1

clean_data 24 x

Read Only

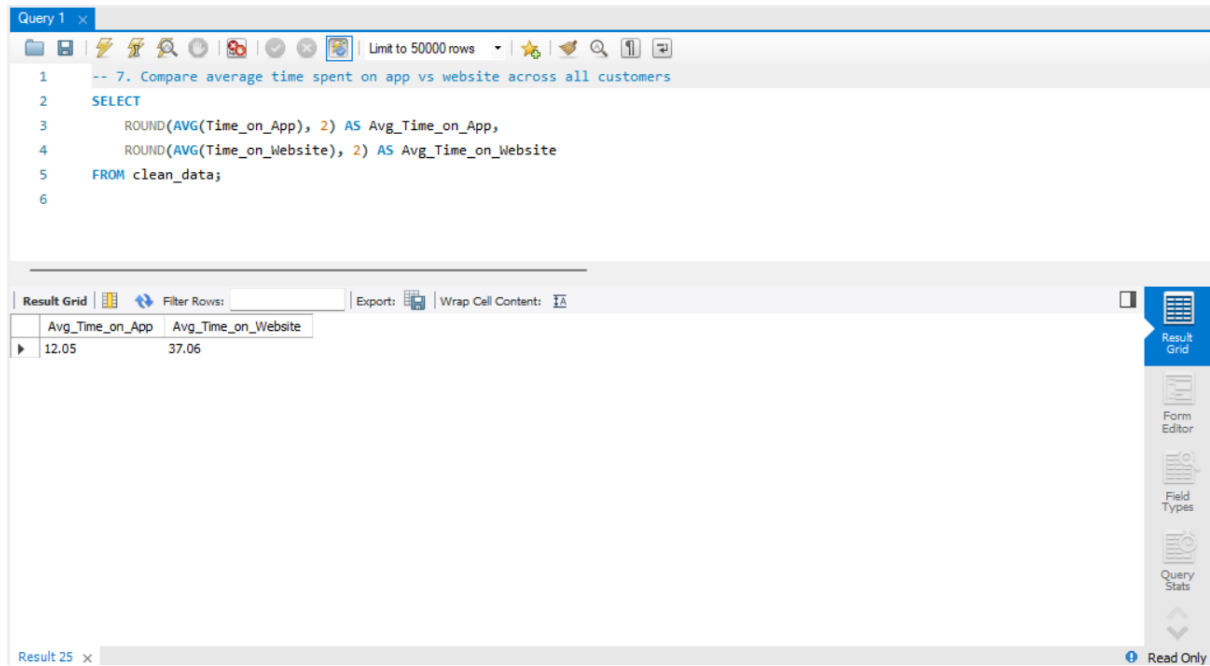
7) Compare average time spent on app vs website across all customers

SELECT

ROUND(AVG(Time_on_App), 2) AS Avg_Time_on_App,

ROUND(AVG(Time_on_Website), 2) AS Avg_Time_on_Website

FROM clean_data;



The screenshot shows a SQL query editor window titled "Query 1" with a toolbar at the top. The query text is as follows:

```
-- 7. Compare average time spent on app vs website across all customers
1  SELECT
2      ROUND(AVG(Time_on_App), 2) AS Avg_Time_on_App,
3      ROUND(AVG(Time_on_Website), 2) AS Avg_Time_on_Website
4  FROM clean_data;
```

Below the query editor is a "Result Grid" showing the output of the query. The grid has two columns: "Avg_Time_on_App" and "Avg_Time_on_Website". The first row contains the values 12.05 and 37.06 respectively. The grid is labeled "Result 25" and has a "Read Only" status.

Avg_Time_on_App	Avg_Time_on_Website
12.05	37.06

8) Emails of customers who spend the most time on website but least on app

SELECT Email, Time_on_Website, Time_on_App

FROM clean_data

ORDER BY Time_on_Website DESC, Time_on_App ASC

LIMIT 10;

Query 1 x

Limit to 50000 rows

```
1 -- 8. Emails of customers who spend the most time on website but least on app
2 SELECT Email, Time_on_Website, Time_on_App
3 FROM clean_data
4 ORDER BY Time_on_Website DESC, Time_on_App ASC
5 LIMIT 10;
6
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows: | Read Only

	Email	Time_on_Website	Time_on_App
▶	davisrobert@hicks-smith.com	40.01	11.19
	randall85@williams.com	39.67	12.99
	sharon82@saunders.info	39.6	12.6
	mstephenson@fernandez.com	39.58	12.66
	kaitlyn78@chang.com	39.29	11.92
	anneingram@miller-alexander.com	39.25	11.61
	gregoryholmes@hotmail.com	39.25	13.76
	alicia28@fuller.com	39.24	11.48
	pkline@hotmail.com	39.22	13.89
	brianwilson@yahoo.com	39.13	12.17

clean_data 26 x

9) Top 3 customers with highest spending efficiency on website (spent per website hour)

SELECT Email, (Yearly_Amount_Spent / Time_on_Website) AS Spending_per_Website_Hour

FROM clean_data

WHERE Time_on_Website > 0

ORDER BY Spending_per_Website_Hour DESC

LIMIT 3;

Query 1 x

Limit to 50000 rows

```
1 -- 9. Top 3 customers with highest spending efficiency on website (spent per website hour)
2 SELECT Email, (Yearly_Amount_Spent / Time_on_Website) AS Spending_per_Website_Hour
3 FROM clean_data
4 WHERE Time_on_Website > 0
5 ORDER BY Spending_per_Website_Hour DESC
6 LIMIT 3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows: | Read Only

	Email	Spending_per_Website_Hour
▶	asilva@yahoo.com	21.946918313181957
	kyang@diaz.org	20.60064585575888
	rhonda01@gmail.com	19.56776152359923

Result 27 x

10) Avatars with highest number of customers having spending above average

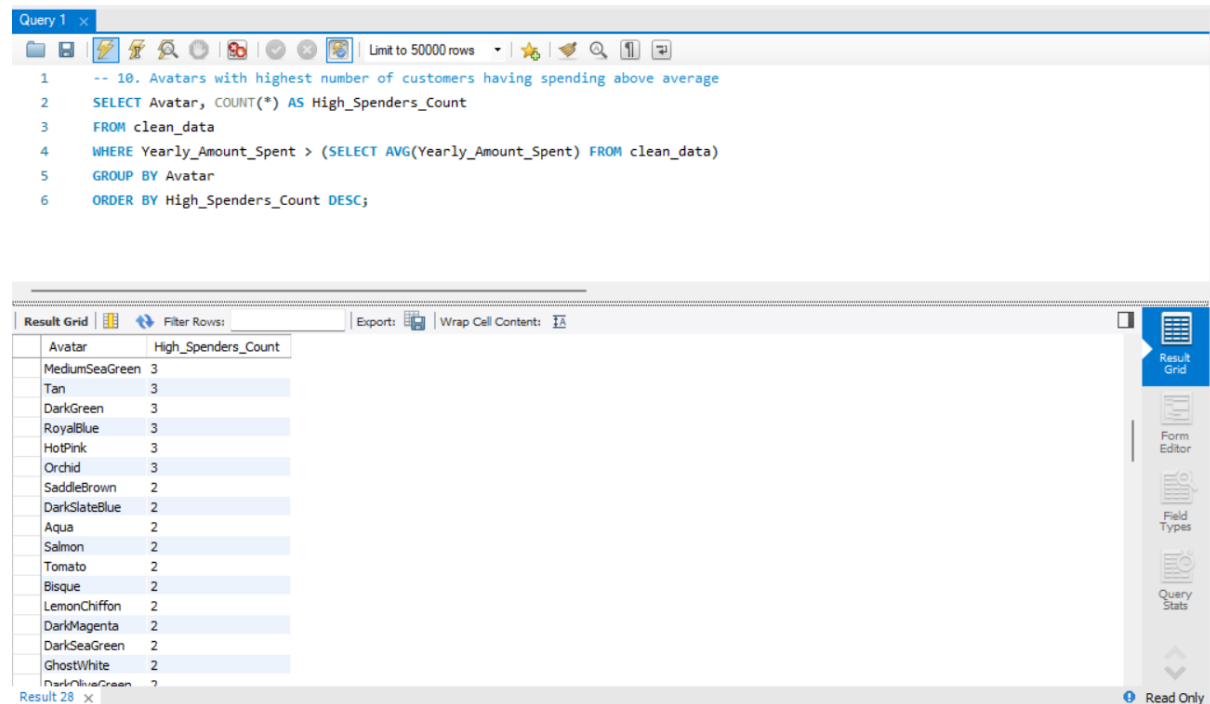
```
SELECT Avatar, COUNT(*) AS High_Spenders_Count
```

```
FROM clean_data
```

```
WHERE Yearly_Amount_Spent > (SELECT AVG(Yearly_Amount_Spent) FROM clean_data)
```

```
GROUP BY Avatar
```

```
ORDER BY High_Spenders_Count DESC;
```



Query 1

```
1 -- 10. Avatars with highest number of customers having spending above average
2 SELECT Avatar, COUNT(*) AS High_Spenders_Count
3 FROM clean_data
4 WHERE Yearly_Amount_Spent > (SELECT AVG(Yearly_Amount_Spent) FROM clean_data)
5 GROUP BY Avatar
6 ORDER BY High_Spenders_Count DESC;
```

Result Grid

Avatar	High_Spenders_Count
MediumSeaGreen	3
Tan	3
DarkGreen	3
RoyalBlue	3
HotPink	3
Orchid	3
SaddleBrown	2
DarkSlateBlue	2
Aqua	2
Salmon	2
Tomato	2
Bisque	2
LemonChiffon	2
DarkMagenta	2
DarkSeaGreen	2
GhostWhite	2
DarkOliveGreen	2

11) Customers with time on app and website both above average

```
SELECT Email, Time_on_App, Time_on_Website
```

```
FROM clean_data
```

```
WHERE Time_on_App > (SELECT AVG(Time_on_App) FROM clean_data)
```


AND Time_on_Website > (SELECT AVG(Time_on_Website) FROM clean_data);

The screenshot shows a SQL query editor with a query window and a results grid. The query is as follows:

```
1 -- 11. Customers with time on app and website both above average
2 • SELECT Email, Time_on_App, Time_on_Website
3 FROM clean_data
4 WHERE Time_on_App > (SELECT AVG(Time_on_App) FROM clean_data)
5 AND Time_on_Website > (SELECT AVG(Time_on_Website) FROM clean_data);
```

The results grid displays the following data:

Email	Time_on_App	Time_on_Website
mstephenson@fernandez.com	12.66	39.58
mstephens@davidson-herman.com	12.8	37.54
awatkins@yahoo.com	12.35	37.37
vchurch@walter-martinez.com	13.39	37.53
andrew06@peterson.com	13.34	37.23
alejandro75@hotmail.com	14.72	38.24
samuel46@love-west.net	13.99	37.19
agolden@yahoo.com	12.88	37.44
vstafford@hotmail.com	13.38	38.73
youngbarbara@yahoo.com	12.89	37.64
heatherhall@yahoo.com	13.01	37.85
joshuaodom@gmail.com	13.1	38.88
jasonrichardson@elliott.com	12.51	37.14
brianwilson@yahoo.com	12.17	39.13
gonzaleskatie@gmail.com	13.15	37.34
william82@gmail.com	13.86	37.78
emithraou@mcnae.com	12.65	38.47

12) Correlation proxy: Avg spending by membership group

SELECT

CASE

WHEN Length_of_Membership < 2 THEN '<2 years'

WHEN Length_of_Membership BETWEEN 2 AND 4 THEN '2-4 years'

ELSE '>4 years'

END AS Membership_Group,

AVG(Yearly_Amount_Spent) AS Avg_Spending

FROM clean_data

GROUP BY Membership_Group

ORDER BY Membership_Group;

Query 1 x

Limit to 50000 rows

```
1 -- 12. Correlation proxy: Avg spending by membership group
2 SELECT
3 CASE
4 WHEN Length_of_Membership < 2 THEN '<2 years'
5 WHEN Length_of_Membership BETWEEN 2 AND 4 THEN '2-4 years'
6 ELSE '>4 years'
7 END AS Membership_Group,
8 AVG(Yearly_Amount_Spent) AS Avg_Spending
9 FROM clean_data
```

Result Grid

Membership_Group	Avg_Spending
<2 years	370.2331428571429
>4 years	569.7993548387101
2-4 years	478.6453548387097

Result 30 x

Read Only

13) Customers with short membership (< 1 year) but high app usage (> avg app time)

SELECT Email, Length_of_Membership, Time_on_App

FROM clean_data

WHERE Length_of_Membership < 1

AND Time_on_App > (SELECT AVG(Time_on_App) FROM clean_data);

Query 1 x

Limit to 50000 rows

```
1 -- 13. Customers with short membership (< 1 year) but high app usage (> avg app time)
2 SELECT Email, Length_of_Membership, Time_on_App
3 FROM clean_data
4 WHERE Length_of_Membership < 1
5 AND Time_on_App > (SELECT AVG(Time_on_App) FROM clean_data);
```

Result Grid

Email	Length_of_Membership	Time_on_App
stevenking@patterson.com	0.97	12.4
sharongraves@yahoo.com	0.27	12.56

clean_data 32 x

Read Only

14) Avg time spent grouped by whether customers have an avatar or not

SELECT

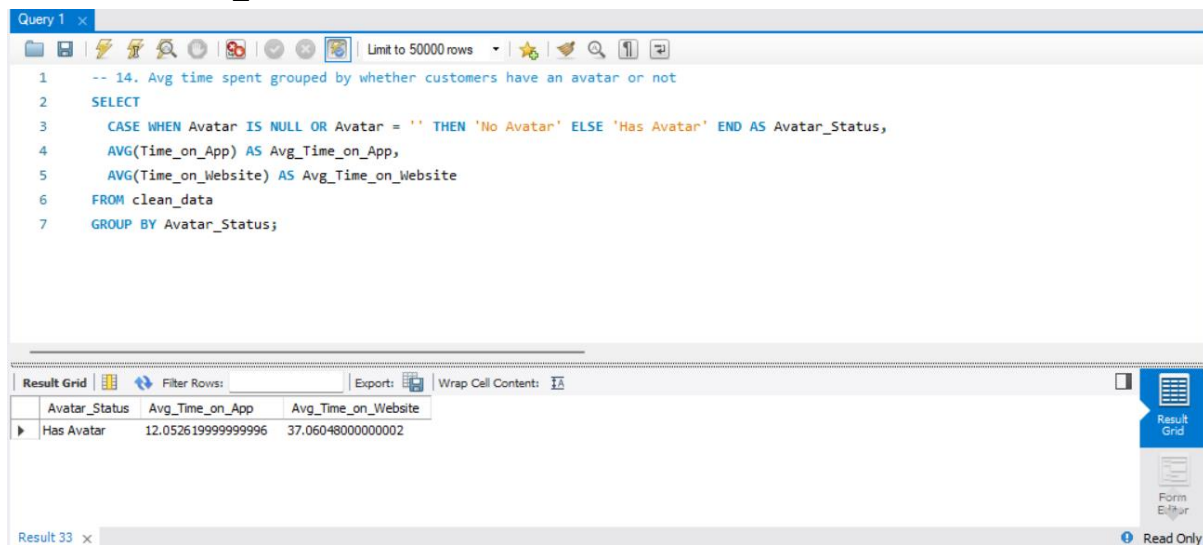
CASE WHEN Avatar IS NULL OR Avatar = '' THEN 'No Avatar' ELSE 'Has Avatar' END AS Avatar_Status,

AVG(Time_on_App) AS Avg_Time_on_App,

AVG(Time_on_Website) AS Avg_Time_on_Website

FROM clean_data

GROUP BY Avatar_Status;



The screenshot shows a SQL query editor window titled "Query 1". The query is as follows:

```
-- 14. Avg time spent grouped by whether customers have an avatar or not
SELECT
  CASE WHEN Avatar IS NULL OR Avatar = '' THEN 'No Avatar' ELSE 'Has Avatar' END AS Avatar_Status,
  AVG(Time_on_App) AS Avg_Time_on_App,
  AVG(Time_on_Website) AS Avg_Time_on_Website
FROM clean_data
GROUP BY Avatar_Status;
```

Below the query editor, the "Result Grid" is displayed, showing the results of the query. The grid has three columns: Avatar_Status, Avg_Time_on_App, and Avg_Time_on_Website. The results are as follows:

Avatar_Status	Avg_Time_on_App	Avg_Time_on_Website
Has Avatar	12.052619999999996	37.060480000000002

15) Top 5 addresses by total spending

SELECT Address, SUM(Yearly_Amount_Spent) AS Total_Spending

FROM clean_data

GROUP BY Address

ORDER BY Total_Spending DESC

LIMIT 5;

Query 1 x

Limit to 50000 rows

```
1 -- 15. Top 5 addresses by total spending
2 SELECT Address, SUM(Yearly_Amount_Spent) AS Total_Spending
3 FROM clean_data
4 GROUP BY Address
5 ORDER BY Total_Spending DESC
6 LIMIT 5;
7
```

Result Grid

Address	Total_Spending
223 Love Trail Suite 831Port Jeffrey, IN 46849	765.52
USNV JohnsonFPO AP 19026	744.22
11143 Park SquaresSamanthatown, UT 97073	725.58
297 Francis ValleySouth Lindsey, NY 13669-5367	725.58
939 Watson RunStaceyberg, VT 58376-0454	708.94

Result 34 x Read Only

16) Customers who spend more time on website than app but spend less than average

SELECT Email, Time_on_Website, Time_on_App, Yearly_Amount_Spent

FROM clean_data

WHERE Time_on_Website > Time_on_App

AND Yearly_Amount_Spent < (SELECT AVG(Yearly_Amount_Spent) FROM clean_data);

Query 1 x

Limit to 50000 rows

```
1 -- 16. Customers who spend more time on website than app but spend less than average
2 SELECT Email, Time_on_Website, Time_on_App, Yearly_Amount_Spent
3 FROM clean_data
4 WHERE Time_on_Website > Time_on_App
5 AND Yearly_Amount_Spent < (SELECT AVG(Yearly_Amount_Spent) FROM clean_data);
```

Result Grid

Email	Time_on_Website	Time_on_App	Yearly_Amount_Spent
jared39@hotmail.com	35.62	10.87	447.69
joshuaodom@gmail.com	38.88	13.1	491.07
enash@gmail.com	36.77	11.76	347.78
sandraharrison@bailey-go...	37.04	11.98	490.74
cunninghamwilliam@hotm...	37.39	9.95	478.17
christopher20@gmail.com	36.11	13.41	448.23
dongarcia@hotmail.com	36.58	11.86	479.73
kimberly46@garcia-nelson...	36.59	11.4	416.36
brenda82@maldonado-go...	37.27	10.96	442.67
josephgould@west.info	37.45	10.32	384.63
wbrady@yahoo.com	35.93	9.98	451.46
amberchase@fowler.info	35.25	11.59	483.67
jeffreydawson@gmail.com	36.17	10.57	453.17
rjohns@gmail.com	36.81	11.83	496.65
shirley78@comcast.net	37.41	11.67	407.16

clean_data 35 x Read Only

17) Calculate ratio of website to app time for each customer and show those with ratio > 2

SELECT Email, Time_on_Website, Time_on_App, (Time_on_Website / Time_on_App) AS Website_App_Ratio

FROM clean_data

WHERE Time_on_App > 0

AND (Time_on_Website / Time_on_App) > 2

ORDER BY Website_App_Ratio DESC;

Query 1

```
-- 17. Calculate ratio of website to app time for each customer and show those with ratio > 2
SELECT Email, Time_on_Website, Time_on_App, (Time_on_Website / Time_on_App) AS Website_App_Ratio
FROM clean_data
WHERE Time_on_App > 0
      AND (Time_on_Website / Time_on_App) > 2
ORDER BY Website_App_Ratio DESC;
```

Result Grid

Email	Time_on_Website	Time_on_App	Website_App_Ratio
dblair@gmail.com	35.46	8.51	4.166862514688602
martinkristi@sanchez-coleman.com	35.91	8.67	4.141868512110726
david80@knight.com	37.91	9.48	3.9989451476793243
cruiz@yahoo.com	36.91	9.32	3.960300429184549
jgray@khan-allen.com	38.35	10.01	3.8311688311688314
floresarthur@yahoo.com	36.49	9.61	3.7970863683662857
christianwade@butler.info	38.07	10.08	3.7767857142857144
lauriewilson@jackson.com	39.05	10.35	3.7729468599033815
aaron89@gmail.com	38.04	10.1	3.7663366336633666
cunninghamwilliam@hotmail.com	37.39	9.95	3.7577889447236186
blakekent@smith-pena.com	37.35	9.95	3.753768844221106
tammyjones@griffin.com	36.88	9.85	3.7441624365482236
aaron04@yahoo.com	37.76	10.16	3.7165354330708658
rileyalejandro@gmail.com	37.18	10.05	3.6995024875621887
...

Result 36