Akilesh K

k.akilesh123@gmail.com

Data engineering - Batch 1

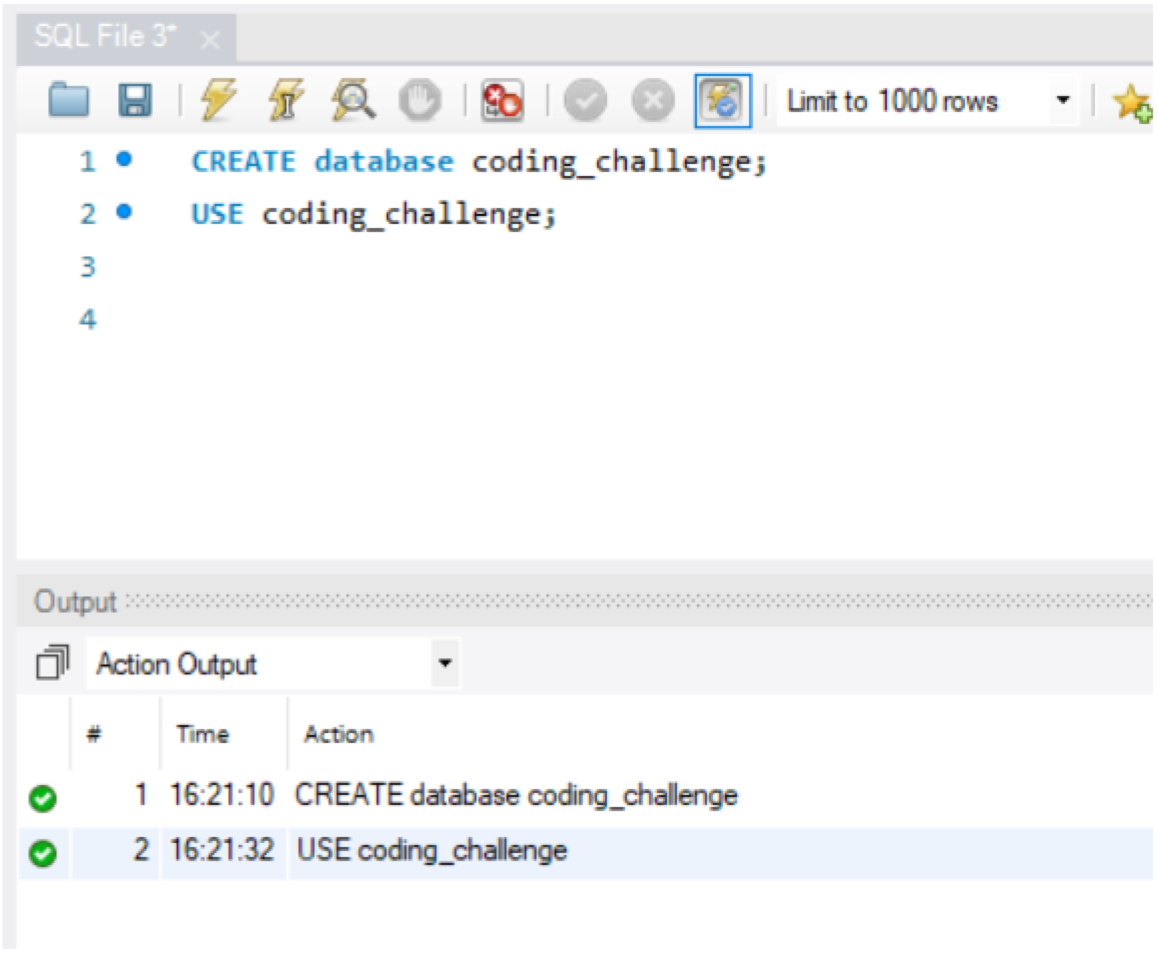
Date: 25-01-24

CODING CHALLENGE

A) Execute OVER and PARTITION BY Clause in SQL Queries

B) creating subtotals &Total Aggregations using SQL Queries.

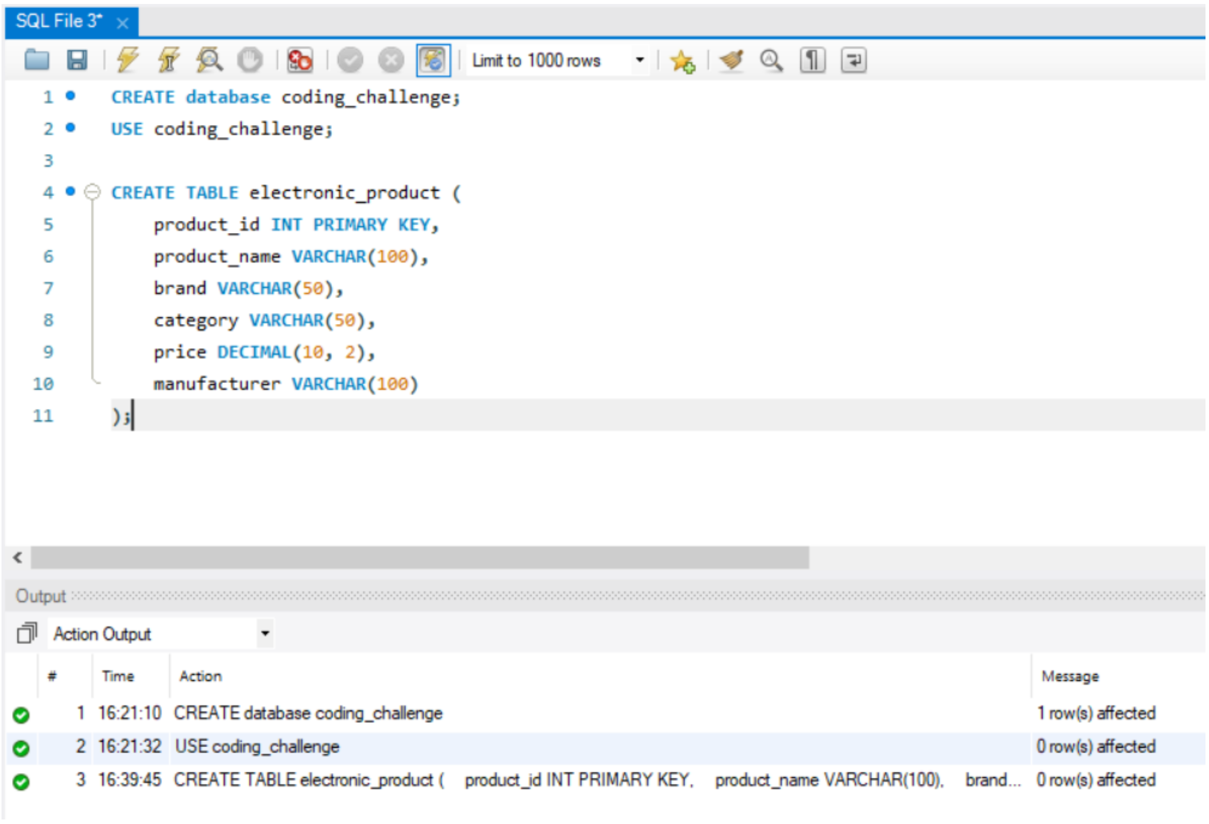
Creating a database named coding challenge



```
CREATE database coding_challenge;
USE coding_challenge;
```

```
#  Time      Action
1  16:21:10  CREATE database coding_challenge
2  16:21:32  USE coding_challenge
```

Creating a table named electronic product with a structured schema and its datatypes

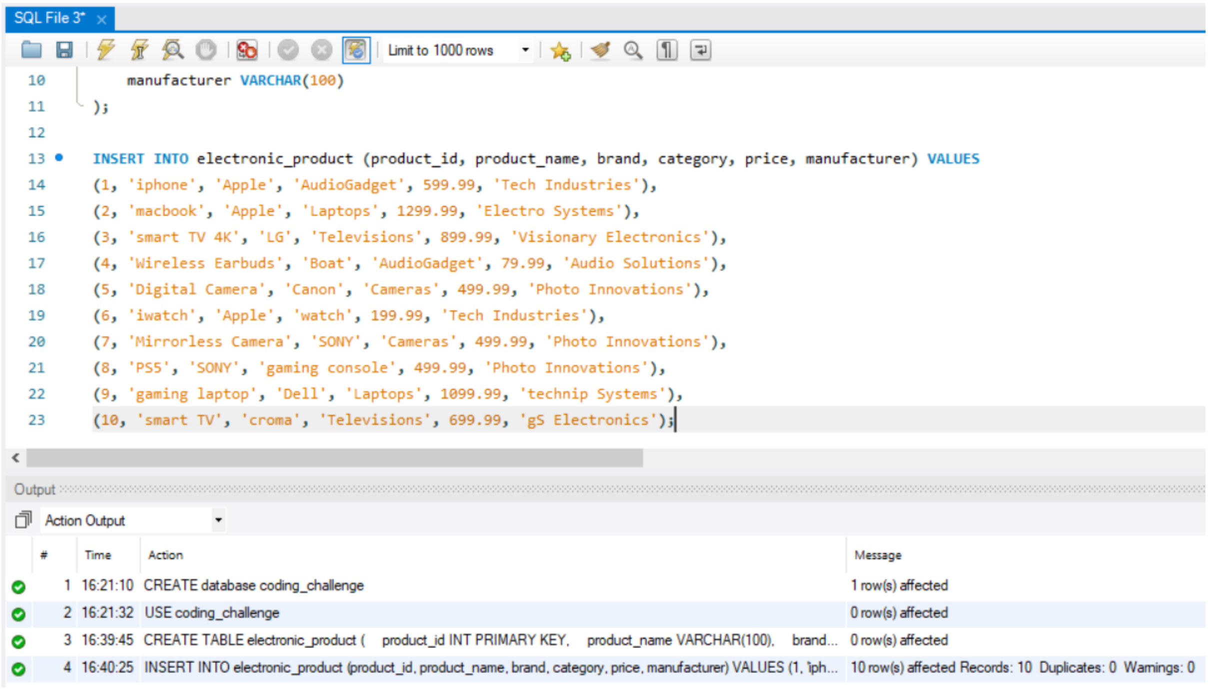


```sql
CREATE database coding_challenge;
USE coding_challenge;

CREATE TABLE electronic_product (
    product_id INT PRIMARY KEY,
    product_name VARCHAR(100),
    brand VARCHAR(50),
    category VARCHAR(50),
    price DECIMAL(10, 2),
    manufacturer VARCHAR(100)
);
```

```
    #  Time      Action                                                                                  Message
    1  16:21:10  CREATE database coding_challenge                                                        1 row(s) affected
    2  16:21:32  USE coding_challenge                                                                    0 row(s) affected
    3  16:39:45  CREATE TABLE electronic_product (    product_id INT PRIMARY KEY,    product_name VARCHAR(100),    brand...   0 row(s) affected
```

Inserting values into the table



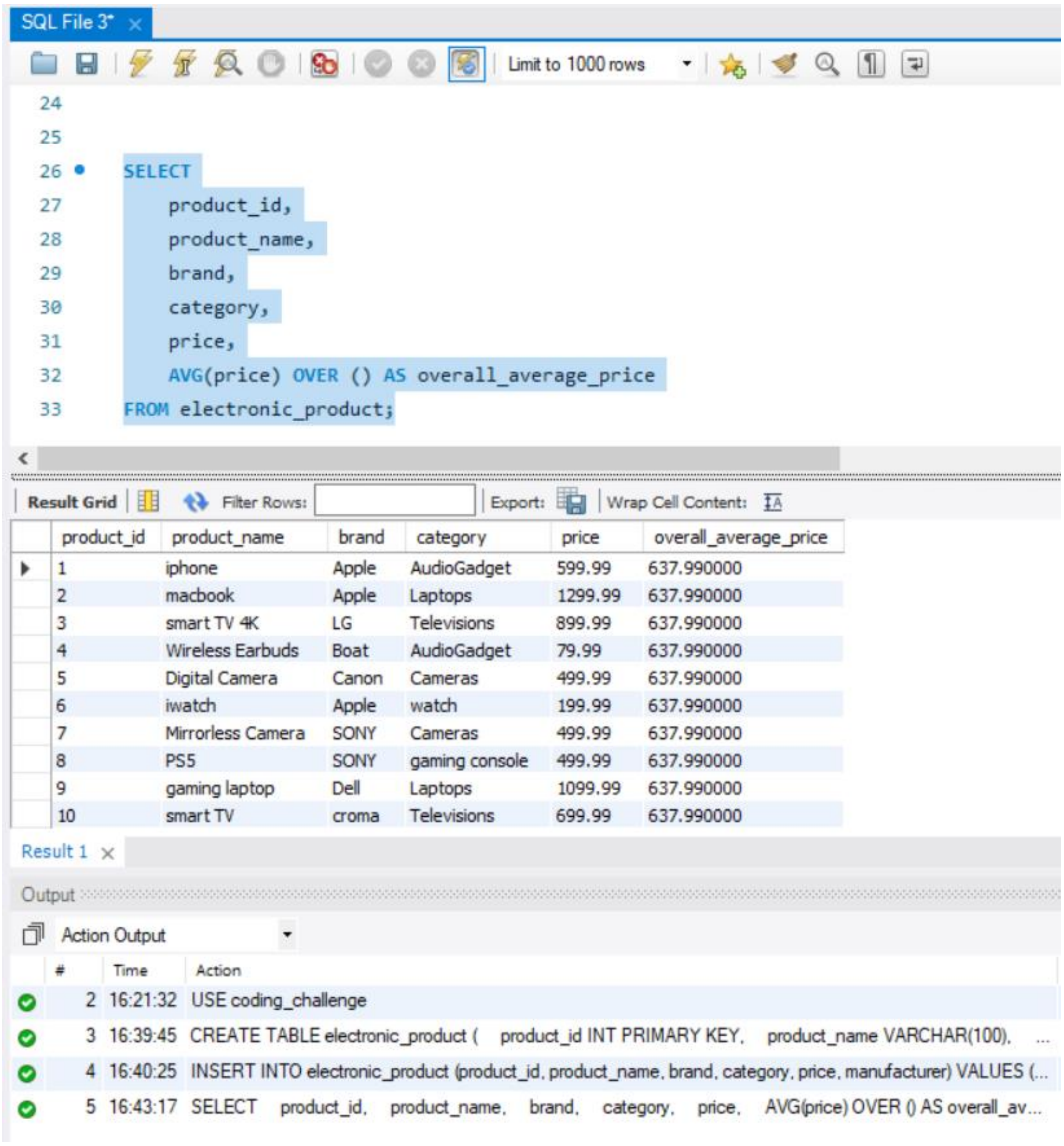
```sql
    manufacturer VARCHAR(100)
);

INSERT INTO electronic_product (product_id, product_name, brand, category, price, manufacturer) VALUES
(1, 'iphone', 'Apple', 'AudioGadget', 599.99, 'Tech Industries'),
(2, 'macbook', 'Apple', 'Laptops', 1299.99, 'Electro Systems'),
(3, 'smart TV 4K', 'LG', 'Televisions', 899.99, 'Visionary Electronics'),
(4, 'Wireless Earbuds', 'Boat', 'AudioGadget', 79.99, 'Audio Solutions'),
(5, 'Digital Camera', 'Canon', 'Cameras', 499.99, 'Photo Innovations'),
(6, 'iwatch', 'Apple', 'watch', 199.99, 'Tech Industries'),
(7, 'Mirrorless Camera', 'SONY', 'Cameras', 499.99, 'Photo Innovations'),
(8, 'PS5', 'SONY', 'gaming console', 499.99, 'Photo Innovations'),
(9, 'gaming laptop', 'Dell', 'Laptops', 1099.99, 'technip Systems'),
(10, 'smart TV', 'croma', 'Televisions', 699.99, 'gS Electronics');
```

```
    #  Time      Action                                                                                              Message
    1  16:21:10  CREATE database coding_challenge                                                                    1 row(s) affected
    2  16:21:32  USE coding_challenge                                                                                0 row(s) affected
    3  16:39:45  CREATE TABLE electronic_product (    product_id INT PRIMARY KEY,    product_name VARCHAR(100),    brand...   0 row(s) affected
    4  16:40:25  INSERT INTO electronic_product (product_id, product_name, brand, category, price, manufacturer) VALUES (1, 'iph...   10 row(s) affected Records: 10  Duplicates: 0  Warnings: 0
```

OVER

SQL query is using the OVER clause with a window function to calculate the overall average price for each row in the electronic_product table.



SQL File 3*

Limit to 1000 rows

```sql
SELECT
    product_id,
    product_name,
    brand,
    category,
    price,
    AVG(price) OVER () AS overall_average_price
FROM electronic_product;
```

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

product_id	product_name	brand	category	price	overall_average_price
1	iphone	Apple	AudioGadget	599.99	637.990000
2	macbook	Apple	Laptops	1299.99	637.990000
3	smart TV 4K	LG	Televisions	899.99	637.990000
4	Wireless Earbuds	Boat	AudioGadget	79.99	637.990000
5	Digital Camera	Canon	Cameras	499.99	637.990000
6	iwatch	Apple	watch	199.99	637.990000
7	Mirrorless Camera	SONY	Cameras	499.99	637.990000
8	PS5	SONY	gaming console	499.99	637.990000
9	gaming laptop	Dell	Laptops	1099.99	637.990000
10	smart TV	croma	Televisions	699.99	637.990000

Result 1

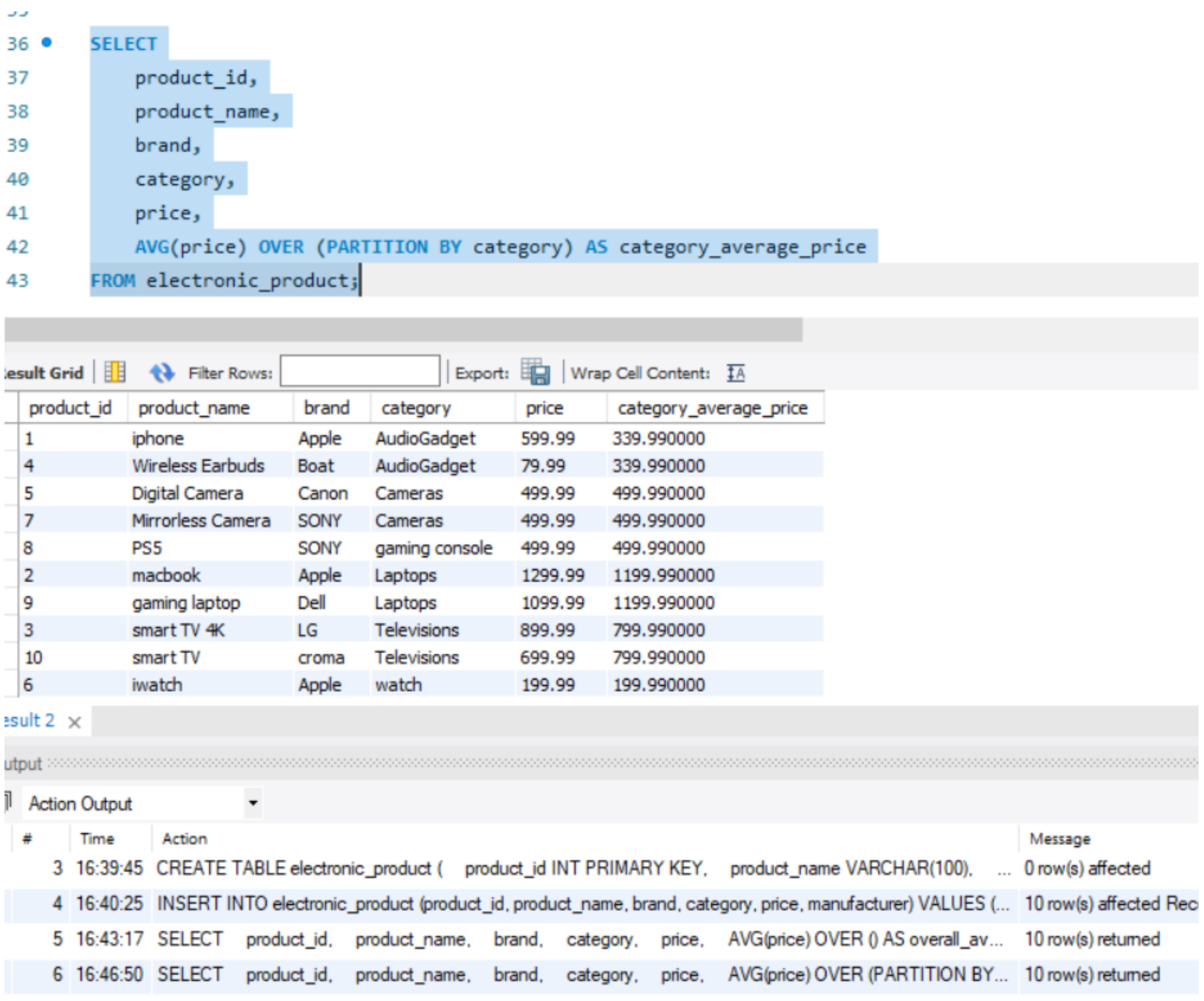
Output

Action Output

#	Time	Action
2	16:21:32	USE coding_challenge
3	16:39:45	CREATE TABLE electronic_product (product_id INT PRIMARY KEY, product_name VARCHAR(100), ...
4	16:40:25	INSERT INTO electronic_product (product_id, product_name, brand, category, price, manufacturer) VALUES (...
5	16:43:17	SELECT product_id, product_name, brand, category, price, AVG(price) OVER () AS overall_av...

PARTITION

The PARTITION BY category clause divides the result set into groups based on the category column. The AVG(price) function is then applied independently within each partition, giving the average price for each category.



```sql
SELECT
    product_id,
    product_name,
    brand,
    category,
    price,
    AVG(price) OVER (PARTITION BY category) AS category_average_price
FROM electronic_product;
```

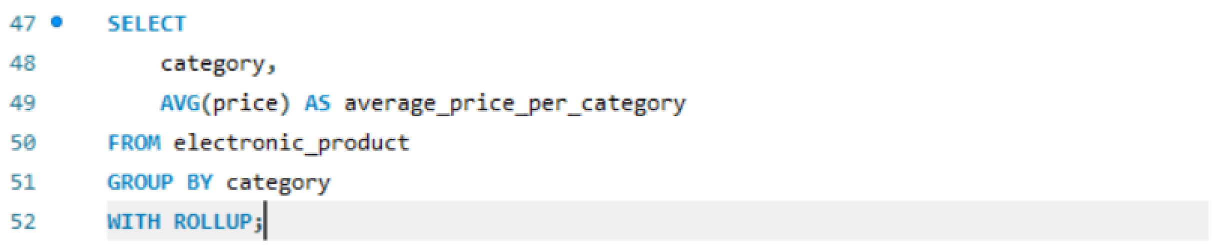
product_id	product_name	brand	category	price	category_average_price
1	iphone	Apple	AudioGadget	599.99	339.990000
4	Wireless Earbuds	Boat	AudioGadget	79.99	339.990000
5	Digital Camera	Canon	Cameras	499.99	499.990000
7	Mirrorless Camera	SONY	Cameras	499.99	499.990000
8	PS5	SONY	gaming console	499.99	499.990000
2	macbook	Apple	Laptops	1299.99	1199.990000
9	gaming laptop	Dell	Laptops	1099.99	1199.990000
3	smart TV 4K	LG	Televisions	899.99	799.990000
10	smart TV	croma	Televisions	699.99	799.990000
6	iwatch	Apple	watch	199.99	199.990000

#	Time	Action	Message
3	16:39:45	CREATE TABLE electronic_product (product_id INT PRIMARY KEY, product_name VARCHAR(100), ...	0 row(s) affected
4	16:40:25	INSERT INTO electronic_product (product_id, product_name, brand, category, price, manufacturer) VALUES (...	10 row(s) affected Rec
5	16:43:17	SELECT product_id, product_name, brand, category, price, AVG(price) OVER () AS overall_av...	10 row(s) returned
6	16:46:50	SELECT product_id, product_name, brand, category, price, AVG(price) OVER (PARTITION BY...	10 row(s) returned

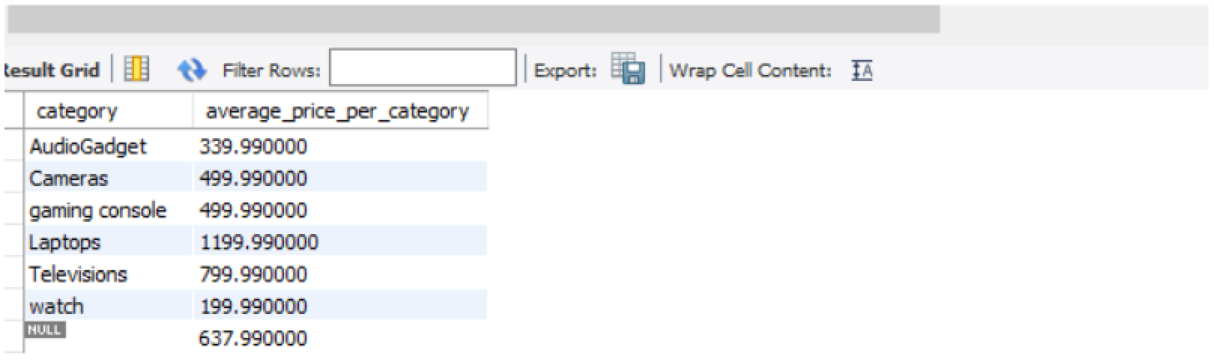
TOTAL AND SUBTOTAL USING AGGREGATION

AVG USING ROLLUP

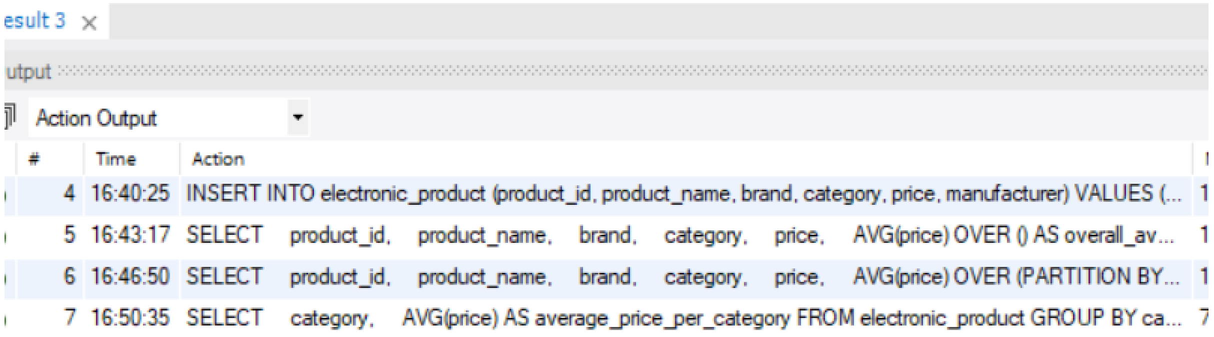
The query calculates the average price for each product category and includes subtotals for each category and a grand total using the WITH ROLLUP clause.



```sql
SELECT
    category,
    AVG(price) AS average_price_per_category
FROM electronic_product
GROUP BY category
WITH ROLLUP;
```



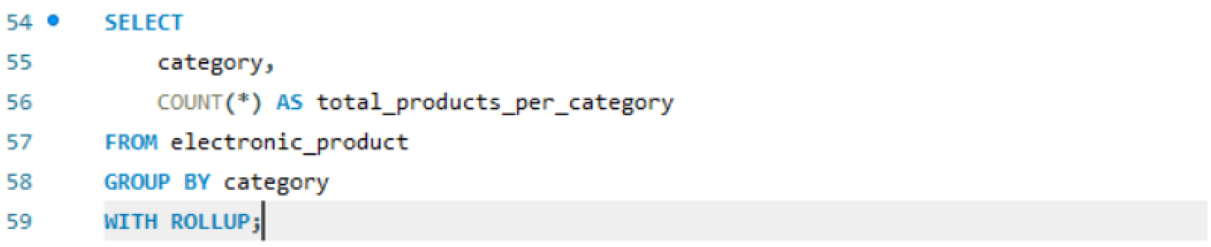
category	average_price_per_category
AudioGadget	339.990000
Cameras	499.990000
gaming console	499.990000
Laptops	1199.990000
Televisions	799.990000
watch	199.990000
NULL	637.990000



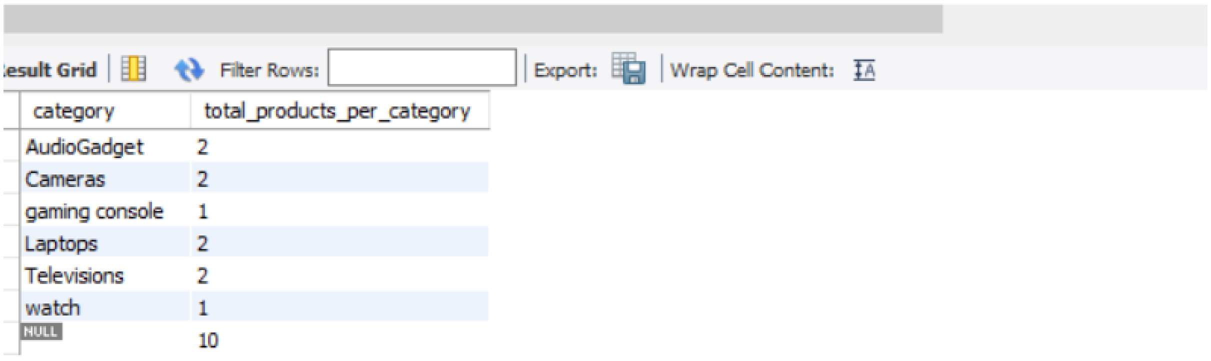
```
#  Time      Action
4  16:40:25  INSERT INTO electronic_product (product_id, product_name, brand, category, price, manufacturer) VALUES (...  1
5  16:43:17  SELECT   product_id,   product_name,   brand,   category,   price,   AVG(price) OVER () AS overall_av...  1
6  16:46:50  SELECT   product_id,   product_name,   brand,   category,   price,   AVG(price) OVER (PARTITION BY...  1
7  16:50:35  SELECT   category,   AVG(price) AS average_price_per_category FROM electronic_product GROUP BY ca...  7
```

COUNT USING ROLLUP

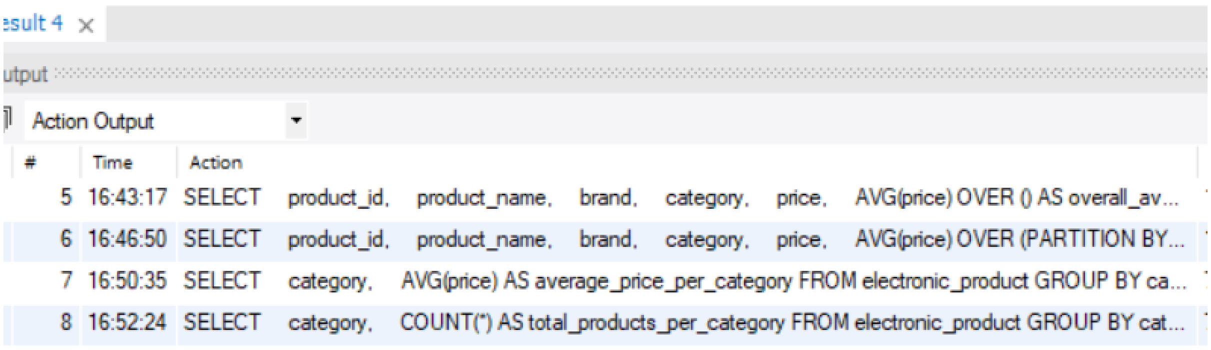
The result set will include counts for each unique category and additional rows for subtotals and a grand total.



```
SELECT
    category,
    COUNT(*) AS total_products_per_category
FROM electronic_product
GROUP BY category
WITH ROLLUP;
```

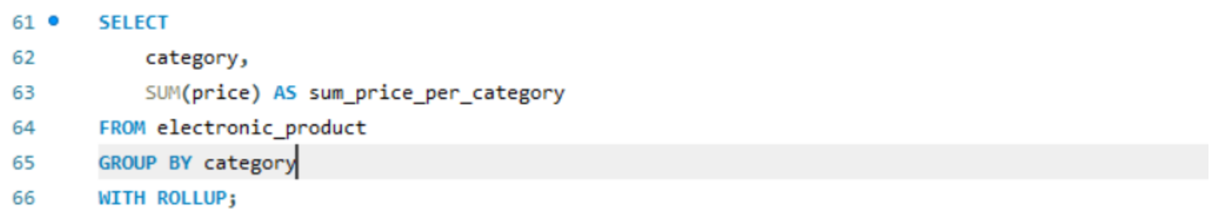


category	total_products_per_category
AudioGadget	2
Cameras	2
gaming console	1
Laptops	2
Televisions	2
watch	1
NULL	10

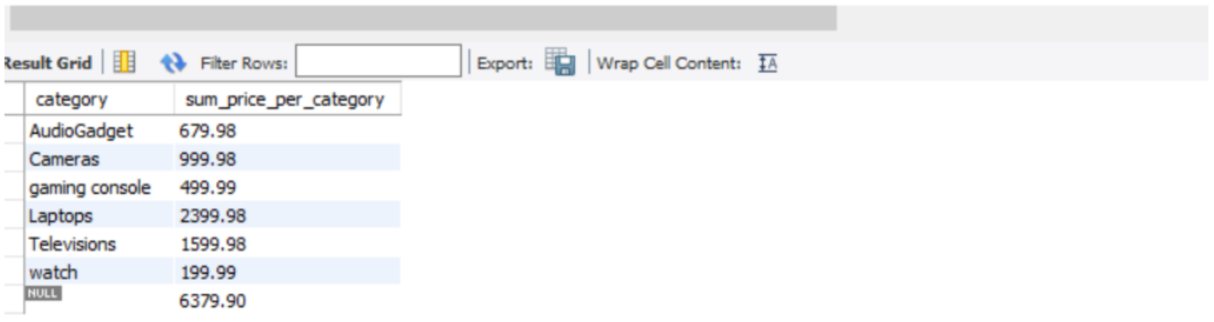


```
#  Time      Action
5  16:43:17  SELECT   product_id,   product_name,   brand,   category,   price,   AVG(price) OVER () AS overall_av...
6  16:46:50  SELECT   product_id,   product_name,   brand,   category,   price,   AVG(price) OVER (PARTITION BY...
7  16:50:35  SELECT   category,   AVG(price) AS average_price_per_category FROM electronic_product GROUP BY ca...
8  16:52:24  SELECT   category,   COUNT(*) AS total_products_per_category FROM electronic_product GROUP BY cat...
```

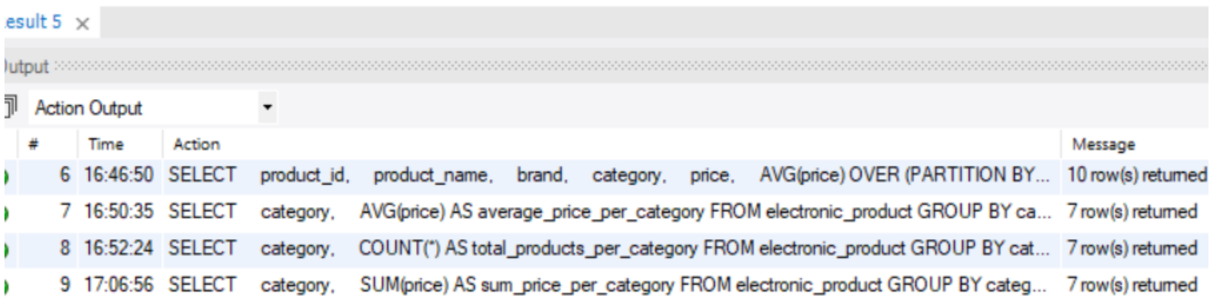
SUM USING ROLLUP



```sql
SELECT
    category,
    SUM(price) AS sum_price_per_category
FROM electronic_product
GROUP BY category
WITH ROLLUP;
```



category	sum_price_per_category
AudioGadget	679.98
Cameras	999.98
gaming console	499.99
Laptops	2399.98
Televisions	1599.98
watch	199.99
NULL	6379.90



#	Time	Action	Message
6	16:46:50	SELECT product_id, product_name, brand, category, price, AVG(price) OVER (PARTITION BY...	10 row(s) returned
7	16:50:35	SELECT category, AVG(price) AS average_price_per_category FROM electronic_product GROUP BY ca...	7 row(s) returned
8	16:52:24	SELECT category, COUNT(*) AS total_products_per_category FROM electronic_product GROUP BY cat...	7 row(s) returned
9	17:06:56	SELECT category, SUM(price) AS sum_price_per_category FROM electronic_product GROUP BY categ...	7 row(s) returned