Part 2: Indexing Analysis

Query 1: Average Price of Watches per Country

SQL Query

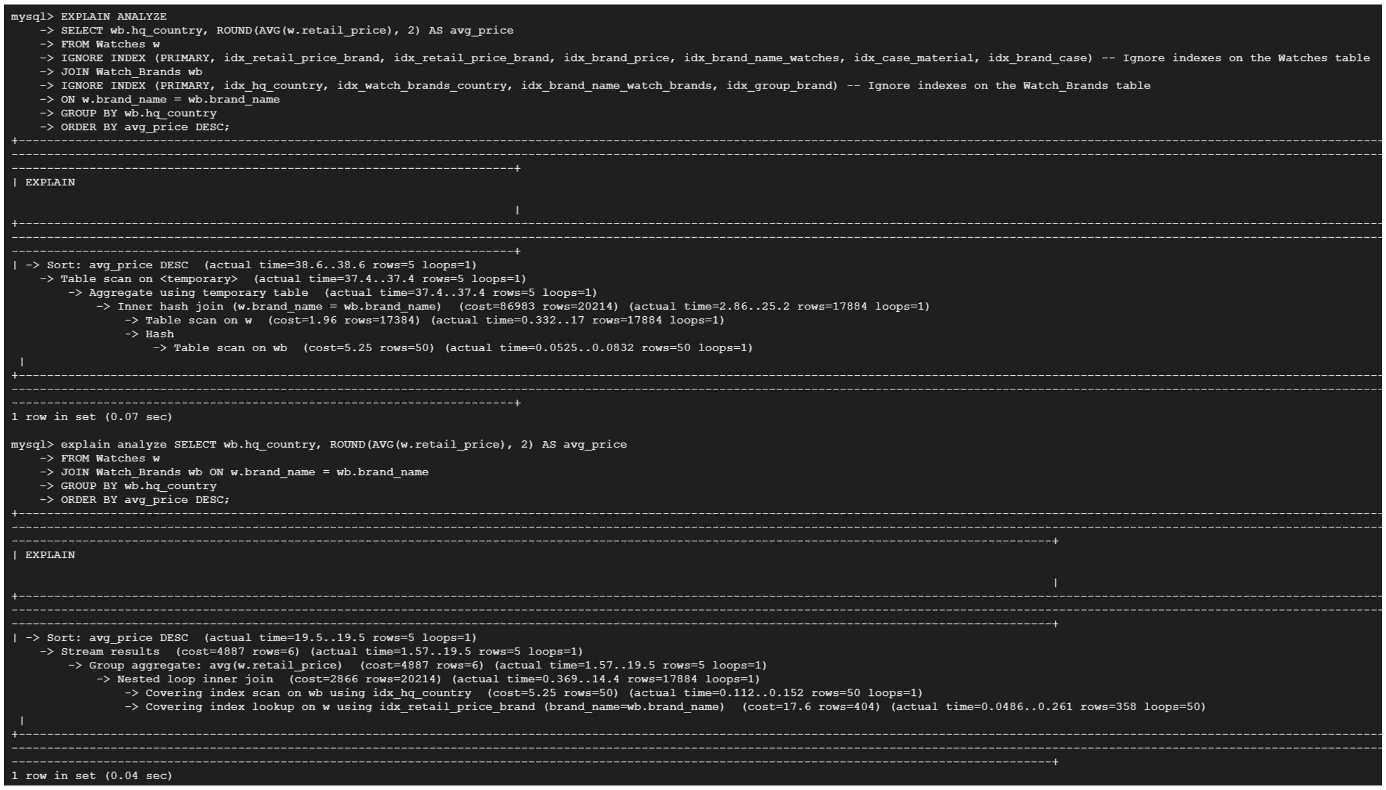
```
SELECT wb.hq_country, ROUND(AVG(w.retail_price), 2) AS avg_price
FROM Watches w
JOIN Watch_Brands wb ON w.brand_name = wb.brand_name
GROUP BY wb.hq_country
ORDER BY avg_price DESC;
```

Indexes Tried

```
CREATE INDEX idx_brand_name_watches ON Watches(brand_name);
CREATE INDEX idx_watch_brands_country ON Watch_Brands(hq_country);
CREATE INDEX idx_brand_price ON Watches(brand_name, retail_price);
```

Summary Table

Index Strategy	Cost Estimate	Actual Time (ms)
No indexes	86893	196–286
Join indexes on `brand_name`	4887	1.11–16.2
+ Index on `hq_country`	4887	1.00–15.6
+ Composite index on `brand_name, price`	4887	1.07–15.4



```
mysql> EXPLAIN ANALYZE
    -> SELECT wb.hq_country, ROUND(AVG(w.retail_price), 2) AS avg_price
    -> FROM Watches w
    -> IGNORE INDEX (PRIMARY, idx_retail_price_brand, idx_retail_price_brand, idx_brand_price, idx_brand_name_watches, idx_case_material, idx_brand_case) -- Ignore indexes on the Watches table
    -> JOIN Watch_Brands wb
    -> IGNORE INDEX (PRIMARY, idx_hq_country, idx_watch_brands_country, idx_brand_name_watch_brands, idx_group_brand) -- Ignore indexes on the Watch_Brands table
    -> ON w.brand_name = wb.brand_name
    -> GROUP BY wb.hq_country
    -> ORDER BY avg_price DESC;
+-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------+
| EXPLAIN

                                                                       |
+-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------+
| -> Sort: avg_price DESC  (actual time=38.6..38.6 rows=5 loops=1)
    -> Table scan on <temporary>  (actual time=37.4..37.4 rows=5 loops=1)
        -> Aggregate using temporary table  (actual time=37.4..37.4 rows=5 loops=1)
            -> Inner hash join (w.brand_name = wb.brand_name)  (cost=86983 rows=20214) (actual time=2.86..25.2 rows=17884 loops=1)
                -> Table scan on w  (cost=1.96 rows=17384) (actual time=0.332..17 rows=17884 loops=1)
                -> Hash
                    -> Table scan on wb  (cost=5.25 rows=50) (actual time=0.0525..0.0832 rows=50 loops=1)
 |
+-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------+
1 row in set (0.07 sec)

mysql> explain analyze SELECT wb.hq_country, ROUND(AVG(w.retail_price), 2) AS avg_price
    -> FROM Watches w
    -> JOIN Watch_Brands wb ON w.brand_name = wb.brand_name
    -> GROUP BY wb.hq_country
    -> ORDER BY avg_price DESC;
+-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN

                                                                                                                                                   |
+-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Sort: avg_price DESC  (actual time=19.5..19.5 rows=5 loops=1)
    -> Stream results  (cost=4887 rows=6) (actual time=1.57..19.5 rows=5 loops=1)
        -> Group aggregate: avg(w.retail_price)  (cost=4887 rows=6) (actual time=1.57..19.5 rows=5 loops=1)
            -> Nested loop inner join  (cost=2866 rows=20214) (actual time=0.369..14.4 rows=17884 loops=1)
                -> Covering index scan on wb using idx_hq_country  (cost=5.25 rows=50) (actual time=0.112..0.152 rows=50 loops=1)
                -> Covering index lookup on w using idx_retail_price_brand (brand_name=wb.brand_name)  (cost=17.6 rows=404) (actual time=0.0486..0.261 rows=358 loops=50)
 |
+-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.04 sec)
```

Analysis

Overall the cost estimate decreased significantly from the join index on brand_name while remaining constant on the additional indexes. This shows a strong positive index and we must take this into consideration in the final analysis.

Query 2: Average Price per Brand

SQL Query

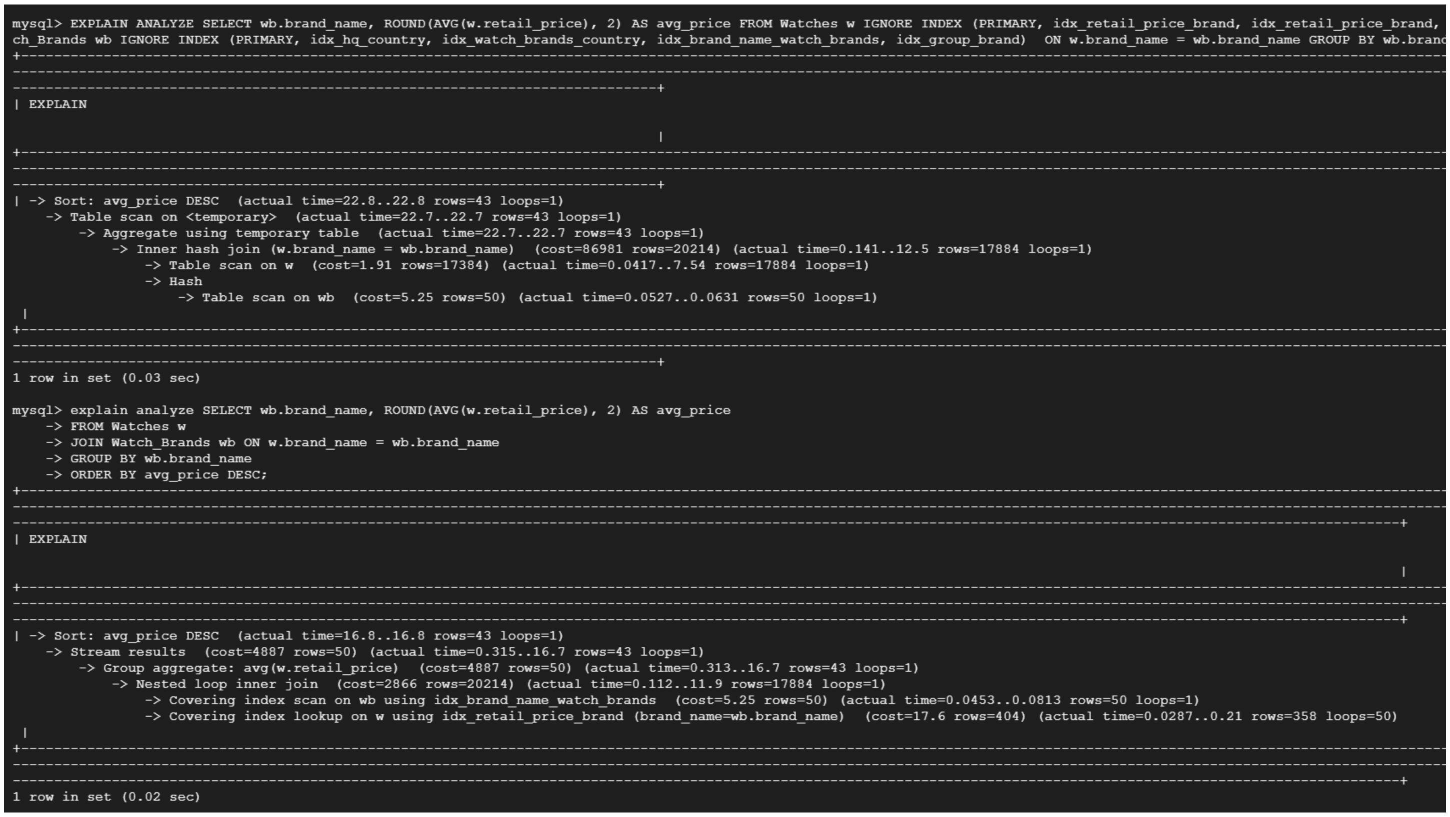
```
SELECT wb.brand_name, ROUND(AVG(w.retail_price), 2) AS avg_price
FROM Watches w
JOIN Watch_Brands wb ON w.brand_name = wb.brand_name
GROUP BY wb.brand_name
ORDER BY avg_price DESC;
```

Indexes Tried

```
CREATE INDEX idx_brand_name_watch_brands ON Watch_Brands(brand_name);
CREATE INDEX idx_retail_price_brand ON Watches(brand_name, retail_price);
```

Summary Table

Index Strategy	**Cost Estimate**	**Actual Time (ms)**
No indexes	86981	~20.2
Index on `brand_name` (Watch_Brands)	4887	~13.8
Composite index (brand + price)	4887	~14.9



```
mysql> EXPLAIN ANALYZE SELECT wb.brand_name, ROUND(AVG(w.retail_price), 2) AS avg_price FROM Watches w IGNORE INDEX (PRIMARY, idx_retail_price_brand, idx_retail_price_brand,
ch_Brands wb IGNORE INDEX (PRIMARY, idx_hq_country, idx_watch_brands_country, idx_brand_name_watch_brands, idx_group_brand)  ON w.brand_name = wb.brand_name GROUP BY wb.brand
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------+
| EXPLAIN

                                                                                |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------+
| -> Sort: avg_price DESC  (actual time=22.8..22.8 rows=43 loops=1)
    -> Table scan on <temporary>  (actual time=22.7..22.7 rows=43 loops=1)
        -> Aggregate using temporary table  (actual time=22.7..22.7 rows=43 loops=1)
            -> Inner hash join (w.brand_name = wb.brand_name)  (cost=86981 rows=20214) (actual time=0.141..12.5 rows=17884 loops=1)
                -> Table scan on w  (cost=1.91 rows=17384) (actual time=0.0417..7.54 rows=17884 loops=1)
                -> Hash
                    -> Table scan on wb  (cost=5.25 rows=50) (actual time=0.0527..0.0631 rows=50 loops=1)
 |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------+
1 row in set (0.03 sec)

mysql> explain analyze SELECT wb.brand_name, ROUND(AVG(w.retail_price), 2) AS avg_price
    -> FROM Watches w
    -> JOIN Watch_Brands wb ON w.brand_name = wb.brand_name
    -> GROUP BY wb.brand_name
    -> ORDER BY avg_price DESC;
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN

                                                                                                                                                                             |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Sort: avg_price DESC  (actual time=16.8..16.8 rows=43 loops=1)
    -> Stream results  (cost=4887 rows=50) (actual time=0.315..16.7 rows=43 loops=1)
        -> Group aggregate: avg(w.retail_price)  (cost=4887 rows=50) (actual time=0.313..16.7 rows=43 loops=1)
            -> Nested loop inner join  (cost=2866 rows=20214) (actual time=0.112..11.9 rows=17884 loops=1)
                -> Covering index scan on wb using idx_brand_name_watch_brands  (cost=5.25 rows=50) (actual time=0.0453..0.0813 rows=50 loops=1)
                -> Covering index lookup on w using idx_retail_price_brand (brand_name=wb.brand_name)  (cost=17.6 rows=404) (actual time=0.0287..0.21 rows=358 loops=50)
 |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.02 sec)
```

Analysis

There was once again a very large difference just like query 1.

Query 3: Watches That Exceed Market Price

SQL Query

```
SELECT wb.brand_name, ROUND(AVG(w.retail_price), 2) AS avg_retail_price
FROM Watches w
JOIN Watch_Brands wb ON w.brand_name = wb.brand_name
GROUP BY wb.brand_name
HAVING AVG(w.retail_price) > (
   SELECT AVG(bmp.Price)
   FROM Brand_Market_Data bmp
   WHERE bmp.brand_name = wb.brand_name
);
```

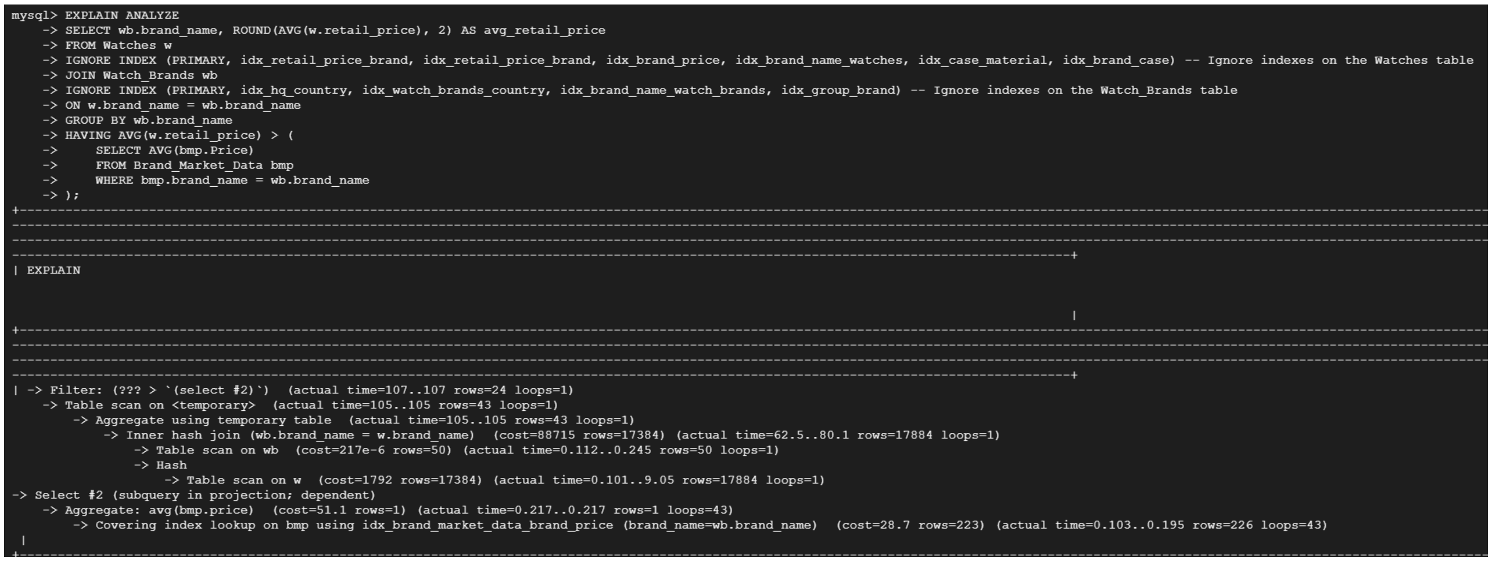
Indexes Tried

```
CREATE INDEX idx_brand_name_bmd ON Brand_Market_Data(brand_name);
CREATE INDEX idx_bmd_brand_price ON Brand_Market_Data(brand_name, price);
CREATE INDEX idx_brand_price ON Watches(brand_name, retail_price);
```

Summary Table

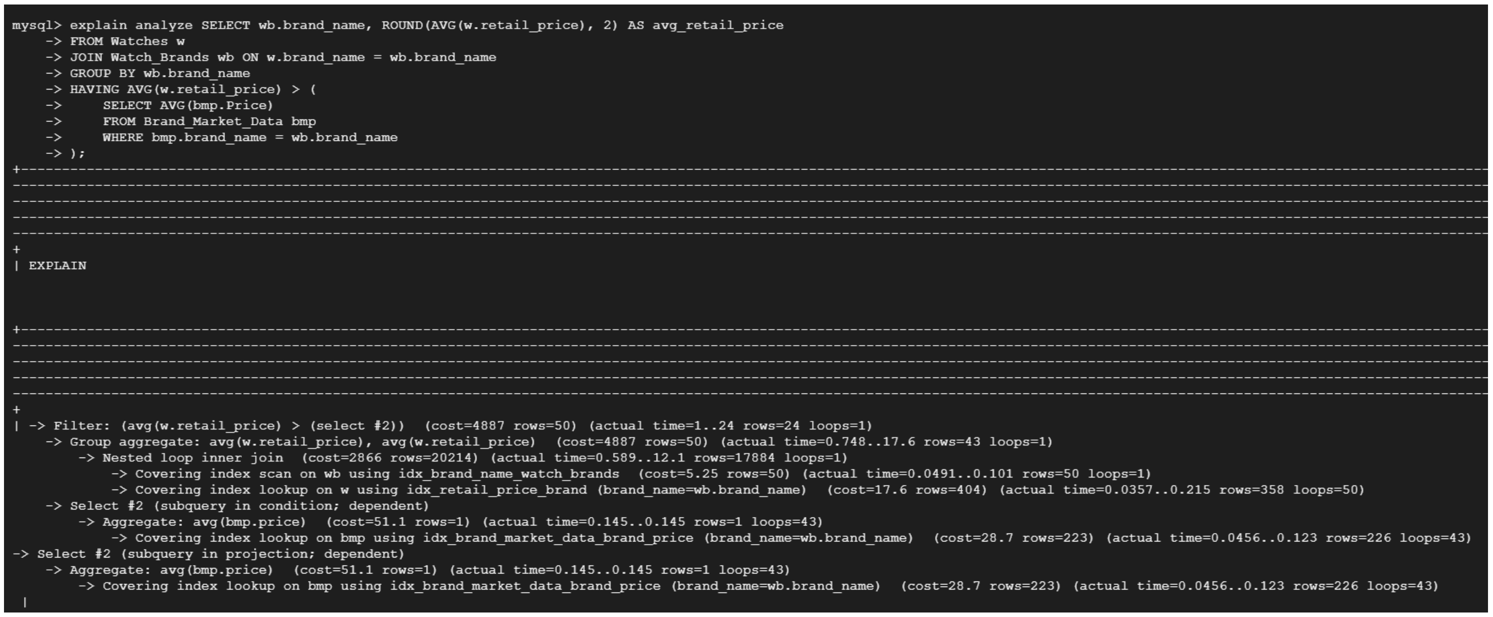
Index Strategy	**Cost Estimate**	**Actual Time (ms)**
No indexes	88715	~280
Index on `brand_name` (BMD)	4887	~54.1
Composite index on `(brand_name, price)`	4887	~52–54
Composite index on Watches	4887	~50–53

Analysis



```
mysql> EXPLAIN ANALYZE
    -> SELECT wb.brand_name, ROUND(AVG(w.retail_price), 2) AS avg_retail_price
    -> FROM Watches w
    -> IGNORE INDEX (PRIMARY, idx_retail_price_brand, idx_retail_price_brand, idx_brand_price, idx_brand_name_watches, idx_case_material, idx_brand_case) -- Ignore indexes on the Watches table
    -> JOIN Watch_Brands wb
    -> IGNORE INDEX (PRIMARY, idx_hq_country, idx_watch_brands_country, idx_brand_name_watch_brands, idx_group_brand) -- Ignore indexes on the Watch_Brands table
    -> ON w.brand_name = wb.brand_name
    -> GROUP BY wb.brand_name
    -> HAVING AVG(w.retail_price) > (
    ->     SELECT AVG(bmp.Price)
    ->     FROM Brand_Market_Data bmp
    ->     WHERE bmp.brand_name = wb.brand_name
    -> );
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN

                                                                                                                          |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------+
| -> Filter: (??? > `(select #2)`)  (actual time=107..107 rows=24 loops=1)
    -> Table scan on <temporary>  (actual time=105..105 rows=43 loops=1)
        -> Aggregate using temporary table  (actual time=105..105 rows=43 loops=1)
            -> Inner hash join (wb.brand_name = w.brand_name)  (cost=88715 rows=17384) (actual time=62.5..80.1 rows=17884 loops=1)
                -> Table scan on wb  (cost=217e-6 rows=50) (actual time=0.112..0.245 rows=50 loops=1)
                -> Hash
                    -> Table scan on w  (cost=1792 rows=17384) (actual time=0.101..9.05 rows=17884 loops=1)
-> Select #2 (subquery in projection; dependent)
    -> Aggregate: avg(bmp.price)  (cost=51.1 rows=1) (actual time=0.217..0.217 rows=1 loops=43)
        -> Covering index lookup on bmp using idx_brand_market_data_brand_price (brand_name=wb.brand_name)  (cost=28.7 rows=223) (actual time=0.103..0.195 rows=226 loops=43)
 |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
```



```
mysql> explain analyze SELECT wb.brand_name, ROUND(AVG(w.retail_price), 2) AS avg_retail_price
    -> FROM Watches w
    -> JOIN Watch_Brands wb ON w.brand_name = wb.brand_name
    -> GROUP BY wb.brand_name
    -> HAVING AVG(w.retail_price) > (
    ->     SELECT AVG(bmp.Price)
    ->     FROM Brand_Market_Data bmp
    ->     WHERE bmp.brand_name = wb.brand_name
    -> );
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
+
| EXPLAIN


+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
+
| -> Filter: (avg(w.retail_price) > (select #2))  (cost=4887 rows=50) (actual time=1..24 rows=24 loops=1)
    -> Group aggregate: avg(w.retail_price), avg(w.retail_price)  (cost=4887 rows=50) (actual time=0.748..17.6 rows=43 loops=1)
        -> Nested loop inner join  (cost=2866 rows=20214) (actual time=0.589..12.1 rows=17884 loops=1)
            -> Covering index scan on wb using idx_brand_name_watch_brands  (cost=5.25 rows=50) (actual time=0.0491..0.101 rows=50 loops=1)
            -> Covering index lookup on w using idx_retail_price_brand (brand_name=wb.brand_name)  (cost=17.6 rows=404) (actual time=0.0357..0.215 rows=358 loops=50)
    -> Select #2 (subquery in condition; dependent)
        -> Aggregate: avg(bmp.price)  (cost=51.1 rows=1) (actual time=0.145..0.145 rows=1 loops=43)
            -> Covering index lookup on bmp using idx_brand_market_data_brand_price (brand_name=wb.brand_name)  (cost=28.7 rows=223) (actual time=0.0456..0.123 rows=226 loops=43)
-> Select #2 (subquery in projection; dependent)
    -> Aggregate: avg(bmp.price)  (cost=51.1 rows=1) (actual time=0.145..0.145 rows=1 loops=43)
        -> Covering index lookup on bmp using idx_brand_market_data_brand_price (brand_name=wb.brand_name)  (cost=28.7 rows=223) (actual time=0.0456..0.123 rows=226 loops=43)
 |
```

Query 4: Brands with >3 Case Material Variants

SQL Query

SELECT w.brand_name, COUNT(DISTINCT w.case_material) AS case_material_variants
FROM Watches w
JOIN Brand_Market_Data bmd ON w.brand_name = bmd.brand_name
GROUP BY w.brand_name
HAVING COUNT(DISTINCT w.case_material) > 3;

Indexes Tried

CREATE INDEX idx_brand_name_bmd ON Brand_Market_Data(brand_name);
CREATE INDEX idx_case_material ON Watches(case_material);
CREATE INDEX idx_brand_case ON Watches(brand_name, case_material);

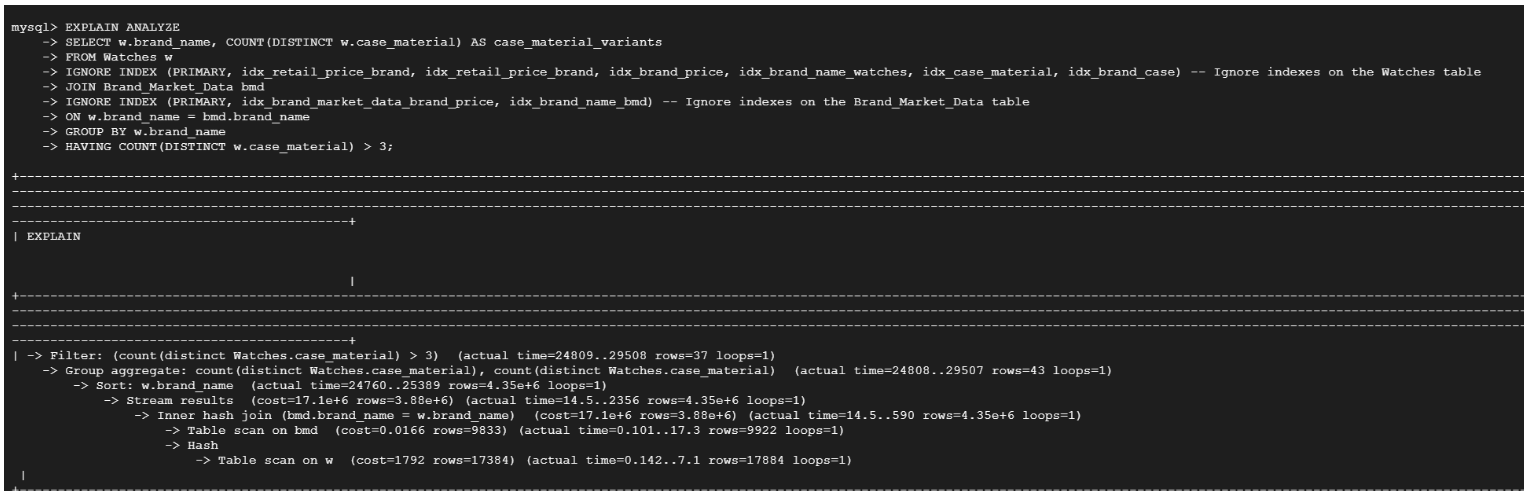
Summary Table

Index Strategy	**Cost Estimate**	**Actual Time (ms)**
No indexes	17.1e+6	~6221
Index on `brand_name` (BMD)	886269	~5858
Index on `case_material`	886269	~5513
Index on `(brand, material)`	886269	~6007

Analysis

The cost estimate decreased significantly with the index on brand_name while remaining constant on case_material and a composite index.

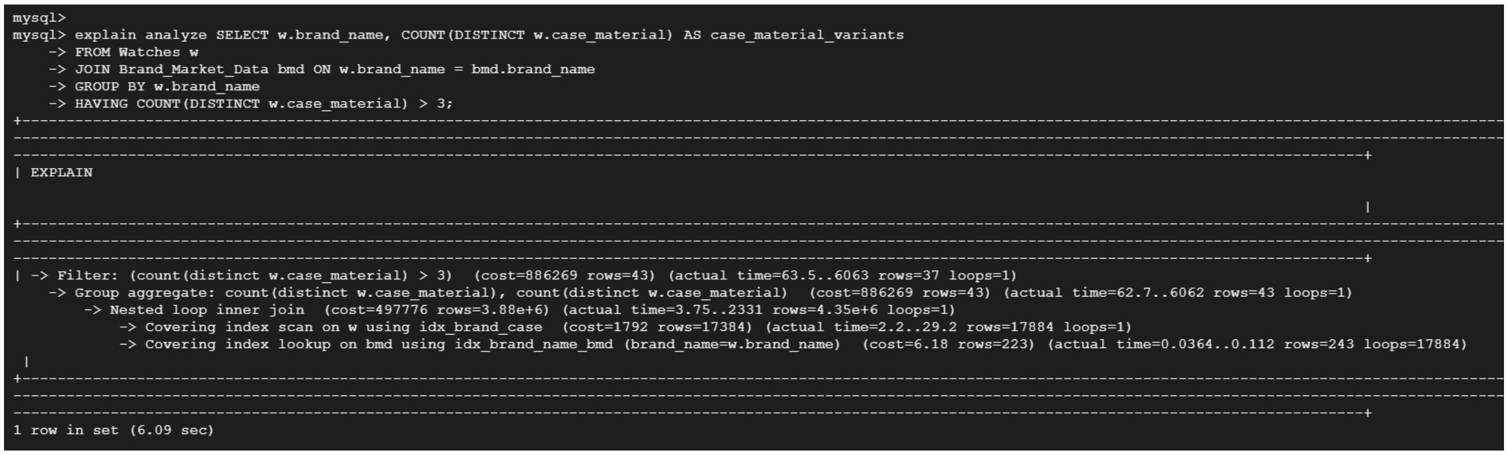
This was by far our most intensive and taxing compute so it’s good to see a large improvement.



```
mysql> EXPLAIN ANALYZE
    -> SELECT w.brand_name, COUNT(DISTINCT w.case_material) AS case_material_variants
    -> FROM Watches w
    -> IGNORE INDEX (PRIMARY, idx_retail_price_brand, idx_retail_price_brand, idx_brand_price, idx_brand_name_watches, idx_case_material, idx_brand_case) -- Ignore indexes on the Watches table
    -> JOIN Brand_Market_Data bmd
    -> IGNORE INDEX (PRIMARY, idx_brand_market_data_brand_price, idx_brand_name_bmd) -- Ignore indexes on the Brand_Market_Data table
    -> ON w.brand_name = bmd.brand_name
    -> GROUP BY w.brand_name
    -> HAVING COUNT(DISTINCT w.case_material) > 3;

+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------+
| EXPLAIN


                                            |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------+
| -> Filter: (count(distinct Watches.case_material) > 3)  (actual time=24809..29508 rows=37 loops=1)
    -> Group aggregate: count(distinct Watches.case_material), count(distinct Watches.case_material)  (actual time=24808..29507 rows=43 loops=1)
        -> Sort: w.brand_name  (actual time=24760..25389 rows=4.35e+6 loops=1)
            -> Stream results  (cost=17.1e+6 rows=3.88e+6) (actual time=14.5..2356 rows=4.35e+6 loops=1)
                -> Inner hash join (bmd.brand_name = w.brand_name)  (cost=17.1e+6 rows=3.88e+6) (actual time=14.5..590 rows=4.35e+6 loops=1)
                    -> Table scan on bmd  (cost=0.0166 rows=9833) (actual time=0.101..17.3 rows=9922 loops=1)
                    -> Hash
                        -> Table scan on w  (cost=1792 rows=17384) (actual time=0.142..7.1 rows=17884 loops=1)
 |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
```



```
mysql>
mysql> explain analyze SELECT w.brand_name, COUNT(DISTINCT w.case_material) AS case_material_variants
    -> FROM Watches w
    -> JOIN Brand_Market_Data bmd ON w.brand_name = bmd.brand_name
    -> GROUP BY w.brand_name
    -> HAVING COUNT(DISTINCT w.case_material) > 3;
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN

                                                                                                                                                         |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Filter: (count(distinct w.case_material) > 3)  (cost=886269 rows=43) (actual time=63.5..6063 rows=37 loops=1)
    -> Group aggregate: count(distinct w.case_material), count(distinct w.case_material)  (cost=886269 rows=43) (actual time=62.7..6062 rows=43 loops=1)
        -> Nested loop inner join  (cost=497776 rows=3.88e+6) (actual time=3.75..2331 rows=4.35e+6 loops=1)
            -> Covering index scan on w using idx_brand_case  (cost=1792 rows=17384) (actual time=2.2..29.2 rows=17884 loops=1)
            -> Covering index lookup on bmd using idx_brand_name_bmd (brand_name=w.brand_name)  (cost=6.18 rows=223) (actual time=0.0364..0.112 rows=243 loops=17884)
 |
+--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (6.09 sec)
```