Music Store Database Project Report

Easy Level Queries

Q1: Most Senior Employee Based on Job Title

Explanation: This query retrieves all employee records and sorts them by the title column in descending order. The assumption is that more senior positions have titles that sort later alphabetically (like "Senior Manager" vs "Junior Associate"). The LIMIT 1 returns only the top result - the most senior employee.

SELECT *

FROM employee

ORDER BY title DESC

LIMIT 1;

```
SELECT *
FROM employee
ORDER BY title DESC
LIMIT 1;
```

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

employee_id	last_name	first_name	title	reports_to	levels	birthdate	hire_date	address	city
3	Peacock	Jane	Sales Support Agent	2	L1	29-08-1973 00:00	01-04-2017 00:00	1111 6 Ave SW	Calga

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

Q2: Countries with Most Invoices

Explanation: This query groups all invoices by their billing_country and counts how many invoices exist for each country. The results are sorted from highest to lowest invoice count, showing which countries generate the most business.

SELECT billing_country, COUNT(*) AS invoice_count

FROM invoice

GROUP BY billing_country

ORDER BY invoice_count DESC;

```
SELECT billing_country, COUNT(*) AS invoice_count
FROM invoice
GROUP BY billing_country
ORDER BY invoice_count DESC;
```

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

billing_country	invoice_count
USA	131
Canada	76
Brazil	61
France	50
Germany	41
Czech Republic	30
Portugal	29
United Kingdom	28
India	21
Ireland	13
Chile	13
Finland	11
Spain	11
Poland	10
Denmark	10
Australia	10
Hungary	10
Sweden	10

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

Result 2 | Read Only

Q3: Top 3 Invoice Totals

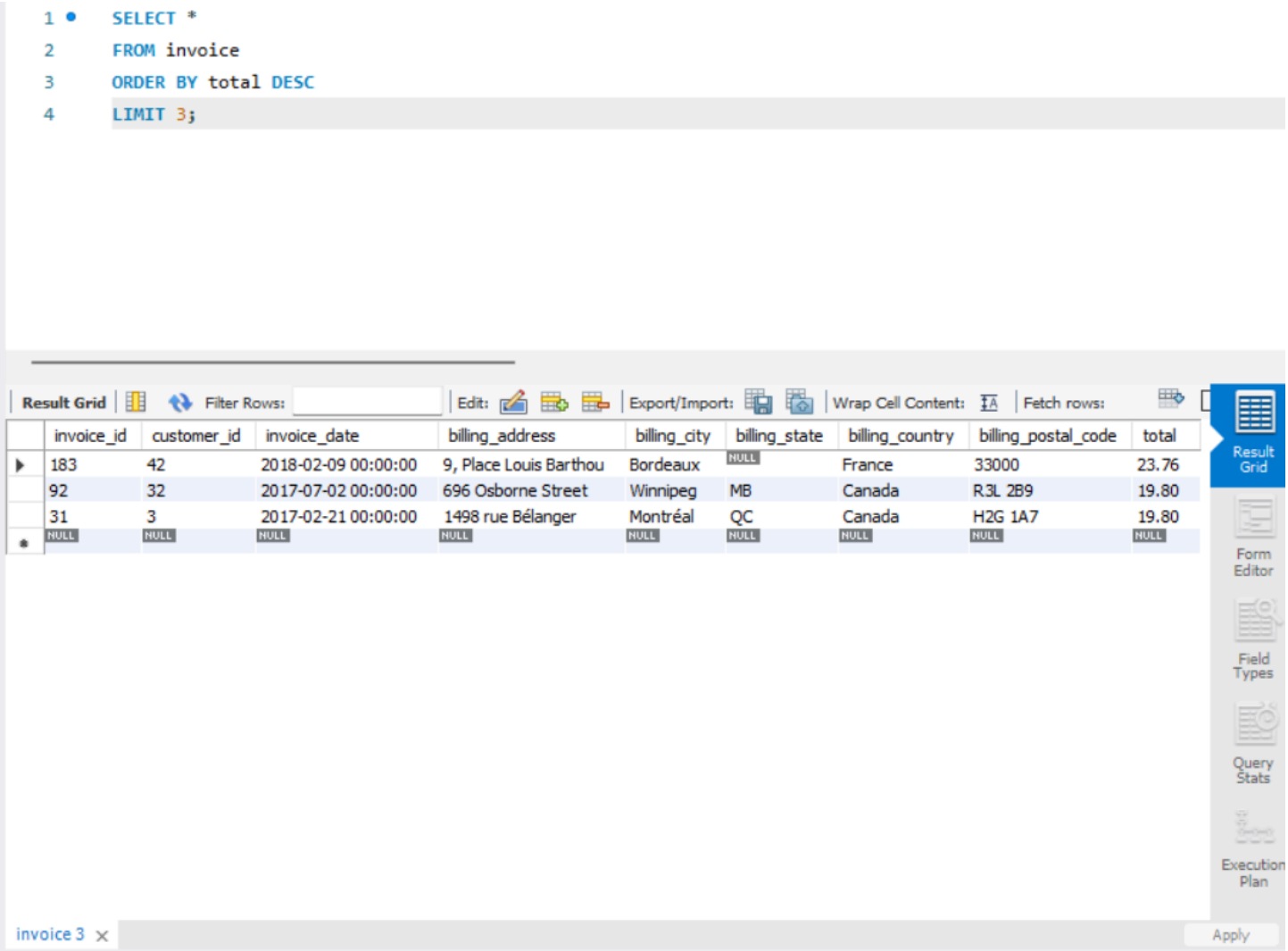
Explanation: This simple query sorts all invoices by their total amount in descending order and returns only the top 3 records. This identifies the three largest individual transactions in the database.

SELECT *

FROM invoice

ORDER BY total DESC

LIMIT 3;



Q4: City with Highest Total Invoice Amount

Explanation: This query calculates the sum of all invoice totals for each city (using GROUP BY billing_city), then sorts these aggregated amounts to find the single city with the highest total sales. This helps identify the best location for promotional events.

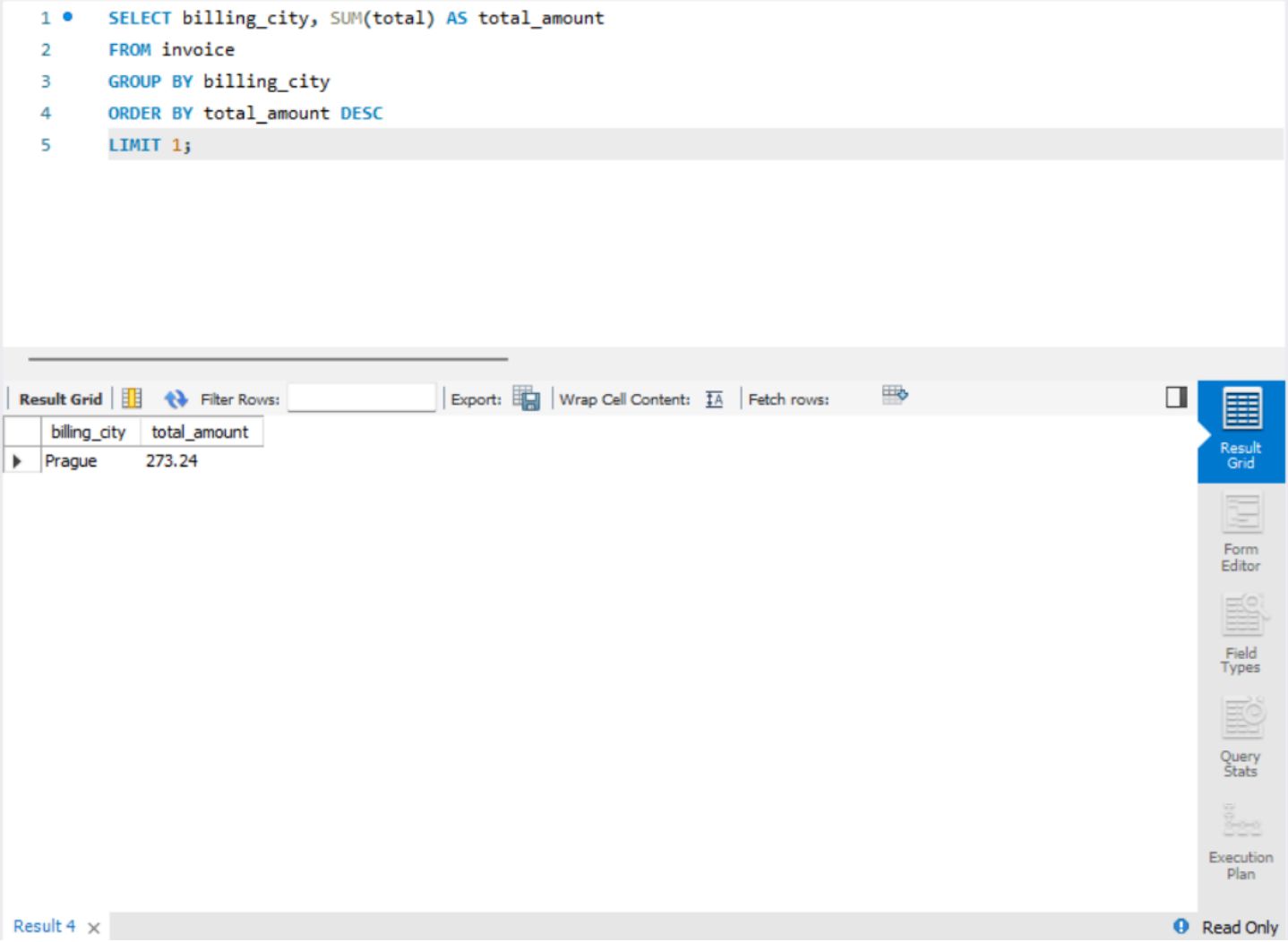
SELECT billing_city, SUM(total) AS total_amount

FROM invoice

GROUP BY billing_city

ORDER BY total_amount DESC

LIMIT 1;



Q5: Customer Who Spent the Most Money

Explanation: This query joins the customer and invoice tables to calculate the total spending for each customer. By grouping on customer details and summing all their invoice totals, then sorting in descending order, we identify the top-spending customer.

SELECT c.customer_id, c.first_name, c.last_name, SUM(i.total) AS total_spent

FROM customer c

JOIN invoice i ON c.customer_id = i.customer_id

GROUP BY c.customer_id

ORDER BY total_spent DESC

LIMIT 1;

```sql
SELECT c.customer_id, c.first_name, c.last_name, SUM(i.total) AS total_spent
FROM customer c
JOIN invoice i ON c.customer_id = i.customer_id
GROUP BY c.customer_id
ORDER BY total_spent DESC
LIMIT 1;
```

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

customer_id	first_name	last_name	total_spent
5	František	Wichterlová	144.54

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

Result 5 | Read Only

Moderate Level Queries

Q1: Customers Who Listen to Rock Music

Explanation: This complex join connects customers to tracks they've purchased through invoices. The WHERE clause filters for only Rock genre tracks. DISTINCT ensures each customer is listed only once, even if they've purchased multiple Rock tracks. Results are ordered by email for easy reading.

SELECT DISTINCT c.email, c.first_name, c.last_name

FROM customer c

JOIN invoice i ON c.customer_id = i.customer_id

JOIN invoice_line il ON i.invoice_id = il.invoice_id

JOIN track t ON il.track_id = t.track_id

JOIN genre g ON t.genre_id = g.genre_id

WHERE g.name = 'Rock'

ORDER BY c.email;

```
SELECT DISTINCT c.email, c.first_name, c.last_name
FROM customer c
JOIN invoice i ON c.customer_id = i.customer_id
JOIN invoice_line il ON i.invoice_id = il.invoice_id
JOIN track t ON il.track_id = t.track_id
JOIN genre g ON t.genre_id = g.genre_id
WHERE g.name = 'Rock'
ORDER BY c.email;
```

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

email	first_name	last_name
aaronmitchell@yahoo.ca	Aaron	Mitchell
alero@uol.com.br	Alexandre	Rocha
astrid.gruber@apple.at	Astrid	Gruber
bjorn.hansen@yahoo.no	Bjørn	Hansen
camille.bernard@yahoo.fr	Camille	Bernard
daan_peeters@apple.be	Daan	Peeters
diego.gutierrez@yahoo.ar	Diego	Gutiérrez
dmiller@comcast.com	Dan	Miller
dominiquelefebvre@gmail.com	Dominique	Lefebvre
edfrancis@yachoo.ca	Edward	Francis
eduardo@woodstock.com.br	Eduardo	Martins
ellie.sullivan@shaw.ca	Ellie	Sullivan
emma_jones@hotmail.com	Emma	Jones
enrique_munoz@yahoo.es	Enrique	Muñoz
fernadaramos4@uol.com.br	Fernanda	Ramos
fharris@google.com	Frank	Harris
fralston@gmail.com	Frank	Ralston
frantisekw@jetbrains.com	František	Wichterlová

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

Result 6 | Read Only

Q2: Top 10 Rock Artists by Track Count

Explanation: This query navigates from artists through albums to tracks, counting how many Rock tracks each artist has. The GROUP BY aggregates tracks per artist, and ORDER BY track_count DESC with LIMIT 10 shows the most prolific Rock artists in the catalog

SELECT ar.artist_id, ar.name, COUNT(*) AS track_count

FROM artist ar

JOIN album al ON ar.artist_id = al.artist_id

JOIN track t ON al.album_id = t.album_id

JOIN genre g ON t.genre_id = g.genre_id

WHERE g.name = 'Rock'

GROUP BY ar.artist_id

ORDER BY track_count DESC

LIMIT 10;

```
SELECT ar.artist_id, ar.name, COUNT(*) AS track_count
FROM artist ar
JOIN album al ON ar.artist_id = al.artist_id
JOIN track t ON al.album_id = t.album_id
JOIN genre g ON t.genre_id = g.genre_id
WHERE g.name = 'Rock'
GROUP BY ar.artist_id, ar.name
ORDER BY track_count DESC
LIMIT 10;
```

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

artist_id	name	track_count
1	AC/DC	18
3	Aerosmith	15
8	Audioslave	14
22	Led Zeppelin	14
4	Alanis Morissette	13
5	Alice In Chains	12
23	Frank Zappa & Captain Beefheart	9
2	Accept	4

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

Result 7 | Read Only

Q3: Tracks Longer Than Average Length

Explanation: The subquery (SELECT AVG(milliseconds) FROM track) calculates the average track length. The main query then finds all tracks longer than this average, presenting them from longest to shortest. This helps identify unusually long tracks

SELECT name, milliseconds

FROM track

WHERE milliseconds > (SELECT AVG(milliseconds) FROM track)

ORDER BY milliseconds DESC;

```sql
SELECT name, milliseconds
FROM track
WHERE milliseconds > (SELECT AVG(milliseconds) FROM track)
ORDER BY milliseconds DESC;
```

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

name	milliseconds
How Many More Times	711836
Advance Romance	677694
Sleeping Village	644571
You Shook Me(2)	619467
Talkin' 'Bout Women Obviously	589531
Stratus	582086
No More Tears	555075
The Alchemist	509413
Wheels Of Confusion / The Straightener	494524
Book Of Thel	494393
You Oughta Know (Alternate)	491885
Terra	482429
Snoopy's search-Red baron	456071
Sozinho (Hitmakers Classic Mix)	436636
Master Of Puppets	436453
Stone Crazy	433397
Snowblind	420022
Computadores Fazem Arte	404323

track 8 | Read Only

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

Advanced Level Queries

Q1: Customer Spending per Artist

Explanation: This uses a CTE (Common Table Expression) named artist_sales to first calculate how much was spent on each artist per invoice. The main query then joins this with customer data to show how much each customer spent on each artist. This provides detailed customer preference data.

```
WITH artist_sales AS (
  SELECT
    il.invoice_id,
    ar.artist_id,
    ar.name AS artist_name,
```

```
    SUM(il.unit_price * il.quantity) AS amount_spent
  FROM invoice_line il
  JOIN track t ON il.track_id = t.track_id
  JOIN album al ON t.album_id = al.album_id
  JOIN artist ar ON al.artist_id = ar.artist_id
  GROUP BY il.invoice_id, ar.artist_id
)

SELECT
  c.customer_id,
  c.first_name,
  c.last_name,
  as.artist_name,
  SUM(as.amount_spent) AS total_spent
FROM customer c
JOIN invoice i ON c.customer_id = i.customer_id
JOIN artist_sales as ON i.invoice_id = as.invoice_id
GROUP BY c.customer_id, as.artist_id
ORDER BY total_spent DESC;
```

```
WITH artist_sales AS (
    SELECT
        il.invoice_id,
        ar.artist_id,
        ar.name AS artist_name,
        SUM(il.unit_price * il.quantity) AS amount_spent
    FROM invoice_line il
    JOIN track t ON il.track_id = t.track_id
    JOIN album al ON t.album_id = al.album_id
    JOIN artist ar ON al.artist_id = ar.artist_id
    GROUP BY il.invoice_id, ar.artist_id, ar.name
)

SELECT
    c.customer_id,
```

Result Grid

customer_id	first_name	last_name	artist_name	total_spent
54	Steve	Murray	AC/DC	17.82
15	Jennifer	Peterson	Aerosmith	14.850000000000001
55	Mark	Taylor	Aerosmith	14.850000000000001
2	Leonie	Köhler	Audioslave	13.860000000000001
13	Fernanda	Ramos	AntÃ´nio Carlos Jobim	13.860000000000001
30	Edward	Francis	Alanis Morissette	12.870000000000001
34	João	Fernandes	Alanis Morissette	12.870000000000001
52	Emma	Jones	Alanis Morissette	12.870000000000001
25	Victor	Stevens	Alice In Chains	11.88
21	Kathy	Chase	AC/DC	10.89
49	Stanisław	Wójcik	Buddy Guy	10.89
53	Phil	Hughes	AC/DC	10.89

Q2: Most Popular Genre by Country

Explanation: The CTE calculates purchase counts for each genre in each country, using RANK() with PARTITION BY to rank genres within each country. The main query filters for only the top-ranked genre (rank = 1) in each country, showing regional music preferences.

```
WITH genre_sales_by_country AS (
  SELECT
    i.billing_country,
    g.genre_id,
    g.name AS genre_name,
    COUNT(*) AS purchase_count,
    RANK() OVER (PARTITION BY i.billing_country ORDER BY COUNT(*) DESC) AS genre_rank
```

FROM invoice i

JOIN invoice_line il ON i.invoice_id = il.invoice_id

JOIN track t ON il.track_id = t.track_id

JOIN genre g ON t.genre_id = g.genre_id

GROUP BY i.billing_country, g.genre_id

)

SELECT billing_country, genre_name, purchase_count

FROM genre_sales_by_country

WHERE genre_rank = 1

ORDER BY billing_country;

```sql
WITH genre_sales_by_country AS (
    SELECT
        i.billing_country,
        g.genre_id,
        g.name AS genre_name,
        COUNT(*) AS purchase_count,
        RANK() OVER (PARTITION BY i.billing_country ORDER BY COUNT(*) DESC) AS genre_rank
    FROM invoice i
    JOIN invoice_line il ON i.invoice_id = il.invoice_id
    JOIN track t ON il.track_id = t.track_id
    JOIN genre g ON t.genre_id = g.genre_id
    GROUP BY i.billing_country, g.genre_id, g.name
)

SELECT billing_country, genre_name, purchase_count
FROM genre_sales_by_country
WHERE genre_rank = 1
```

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

billing_country	genre_name	purchase_count
Argentina	Rock	1
Australia	Rock	18
Austria	Rock	6
Belgium	Rock	5
Brazil	Rock	26
Canada	Rock	57
Chile	Rock	7
Czech Republic	Rock	14
Denmark	Rock	6

Result 3 | Read Only

Q3: Top-Spending Customer per Country

Explanation: Similar to Q2, this uses a CTE with window functions to rank customers by spending within each country. The main query selects only the top-spending customer (rank = 1) from each country, providing valuable information about key customers in different markets.

```
WITH customer_spending AS (
  SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    i.billing_country,
    SUM(i.total) AS total_spent,
    RANK() OVER (PARTITION BY i.billing_country ORDER BY SUM(i.total) DESC) AS spending_rank
  FROM customer c
  JOIN invoice i ON c.customer_id = i.customer_id
  GROUP BY c.customer_id, i.billing_country
)

SELECT customer_id, first_name, last_name, billing_country, total_spent
FROM customer_spending
WHERE spending_rank = 1
ORDER BY billing_country;
```

```sql
        c.customer_id,
        c.first_name,
        c.last_name,
        i.billing_country,
        SUM(i.total) AS total_spent,
        RANK() OVER (PARTITION BY i.billing_country ORDER BY SUM(i.total) DESC) AS spending_rank
    FROM customer c
    JOIN invoice i ON c.customer_id = i.customer_id
    GROUP BY c.customer_id, i.billing_country
)

SELECT customer_id, first_name, last_name, billing_country, total_spent
FROM customer_spending
WHERE spending_rank = 1
ORDER BY billing_country;
```

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

customer_id	first_name	last_name	billing_country	total_spent
56	Diego	Gutiérrez	Argentina	39.60
55	Mark	Taylor	Australia	81.18
7	Astrid	Gruber	Austria	69.30
8	Daan	Peeters	Belgium	60.39
1	Luís	Gonçalves	Brazil	108.90
3	François	Tremblay	Canada	99.99
57	Luis	Rojas	Chile	97.02
5	František	Wichterlová	Czech Republic	144.54
9	Kara	Nielsen	Denmark	37.62
44	Terhi	Hämäläinen	Finland	79.20
42	Wyatt	Girard	France	99.99
37	Fynn	Zimmermann	Germany	94.05
45	Ladislav	Kovács	Hungary	78.21

Result Grid | Form Editor | Field Types

Result 2 | Read Only