SQL Project Report – Indian Railways Fare & Route Analysis

Author: *SAHIL SHAIKH*
Date: *JULY 2025*

---

1. Project Overview

The objective of this project is to analyze Indian Railways fare and route data to identify trends in pricing, popular routes, and class-based revenue distribution.
SQL was used to query the dataset, and insights were drawn from the results.

Dataset Details:

- **Source:** *["Provided CSV file in Dataset Folder downloaded from Kaggle"]*
- **Rows:** *Number of rows*
- **Columns:** *Number of columns*
- **Key Fields:** fromStnCode, toStnCode, classCode, totalFare, distance, trainNumber, timeStamp

---

2. Tools & Technologies Used

- Oracle SQL Developer
- SQL (SELECT, GROUP BY, ORDER BY, UNION, Aggregate Functions)
- CSV dataset

---

3. Queries & Insights

Query 1 – Top 5 Most Expensive Routes

```
SELECT fromStnCode, toStnCode, MAX(totalFare) AS highest_fare

FROM RAILWAY

GROUP BY fromStnCode, toStnCode

ORDER BY highest_fare DESC

FETCH FIRST 5 ROWS ONLY;
```

Result Screenshot:

Worksheet | Query Builder

```
select * from Railway;

-- The top 5 most expensive train routes
SELECT fromStnCode, toStnCode, MAX(totalFare) AS highest_fare
FROM RAILWAY
GROUP BY fromStnCode, toStnCode
ORDER BY highest_fare DESC
FETCH FIRST 5 ROWS ONLY;
```

Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.031 seconds

	FROMSTNCODE	TOSTNCODE	HIGHEST_FARE
1	NU	MAN	3420
2	KY	MAN	3375
3	KKB	MHD	3375
4	JBP	ANND	3375
5	GAR	MAN	3330

Insight:

These routes represent premium travel sectors, possibly due to long distances, higher-class coaches, or seasonal pricing.

Query 2 – Average Fare by Travel Class

```
SELECT classCode, ROUND(AVG(totalFare), 2) AS avg_fare

FROM RAILWAY

GROUP BY classCode

ORDER BY avg_fare DESC;
```

Result Screenshot:

Worksheet | Query Builder

```
select * from Railway;

-- The top 5 most expensive train routes
SELECT fromStnCode, toStnCode, MAX(totalFare) AS highest_fare
FROM RAILWAY
GROUP BY fromStnCode, toStnCode
ORDER BY highest_fare DESC
FETCH FIRST 5 ROWS ONLY;

-- Average fare by travel class
SELECT classCode, ROUND(AVG(totalFare), 2) AS avg_fare
FROM RAILWAY
GROUP BY classCode
ORDER BY avg_fare DESC;

-- Shortest vs longest distance journeys
SELECT fromStnCode, toStnCode, distance
FROM RAILWAY
ORDER BY distance DESC;
```

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.015 seconds

	CLASSCODE	AVG_FARE
1	1A	1580.63
2	2A	1029.63
3	3A	710.76
4	CC	365.7
5	SL	234.38
6	2S	83.89

Insight:
Higher-class compartments like 1A and 2A have significantly higher fares compared to sleeper classes.

Query 3 – Shortest vs Longest Distance Journey

SELECT fromStnCode, toStnCode, distance

FROM RAILWAY

ORDER BY distance DESC;Result Screenshot:

Worksheet | Query Builder

```
SELECT fromStnCode, toStnCode, MAX(totalFare) AS highest_fare
FROM RAILWAY
GROUP BY fromStnCode, toStnCode
ORDER BY highest_fare DESC
FETCH FIRST 5 ROWS ONLY;

-- Average fare by travel class
SELECT classCode, ROUND(AVG(totalFare), 2) AS avg_fare
FROM RAILWAY
GROUP BY classCode
ORDER BY avg_fare DESC;

-- Shortest vs longest distance journeys
SELECT fromStnCode, toStnCode, distance
FROM RAILWAY
ORDER BY distance DESC;
```

Script Output | Query Result

SQL | Fetched 900 rows in 0.035 seconds

	FROMSTNCODE	TOSTNCODE	DISTANCE
1	NK	STA	998
2	NK	STA	998
3	NK	STA	998
4	NK	STA	998
5	STA	NK	998
6	STA	NK	998
7	WR	CHI	998
8	NK	STA	998
9	NK	STA	998
10	NK	STA	998
11	WR	CHI	998
12	WR	CHI	998

Worksheet | Query Builder

```
FROM RAILWAY
GROUP BY classCode
ORDER BY avg_fare DESC;

-- Shortest vs longest distance journeys
SELECT fromStnCode, toStnCode, distance
FROM RAILWAY
ORDER BY distance DESC;
```

Query Result

SQL | Fetched 50 rows in 0.016 seconds

	FROMSTNCODE	TOSTNCODE	DISTA...
1	PVP	PVPT	1
2	PVP	PVPT	1
3	PVP	PVPT	1
4	PVP	PVPT	1
5	PVP	PVPT	1
6	LKR	KIUL	1
7	PVP	PVPT	1
8	CSB	NDLS	1
9	LKR	KIUL	1
10	LKR	KIUL	1
11	PVP	PVPT	1
12	CPA	CNB	2
13	CPA	CNB	2
14	LGL	KRP	2
15	CPA	CNB	2
16	LGL	KRP	2
17	CPA	CNB	2
18	BSB	BSBS	3
19	CGR	CNS	3

Insight:

Longest distance is 998km and shortest is 1km

Query 4 – Total Revenue by Train Number

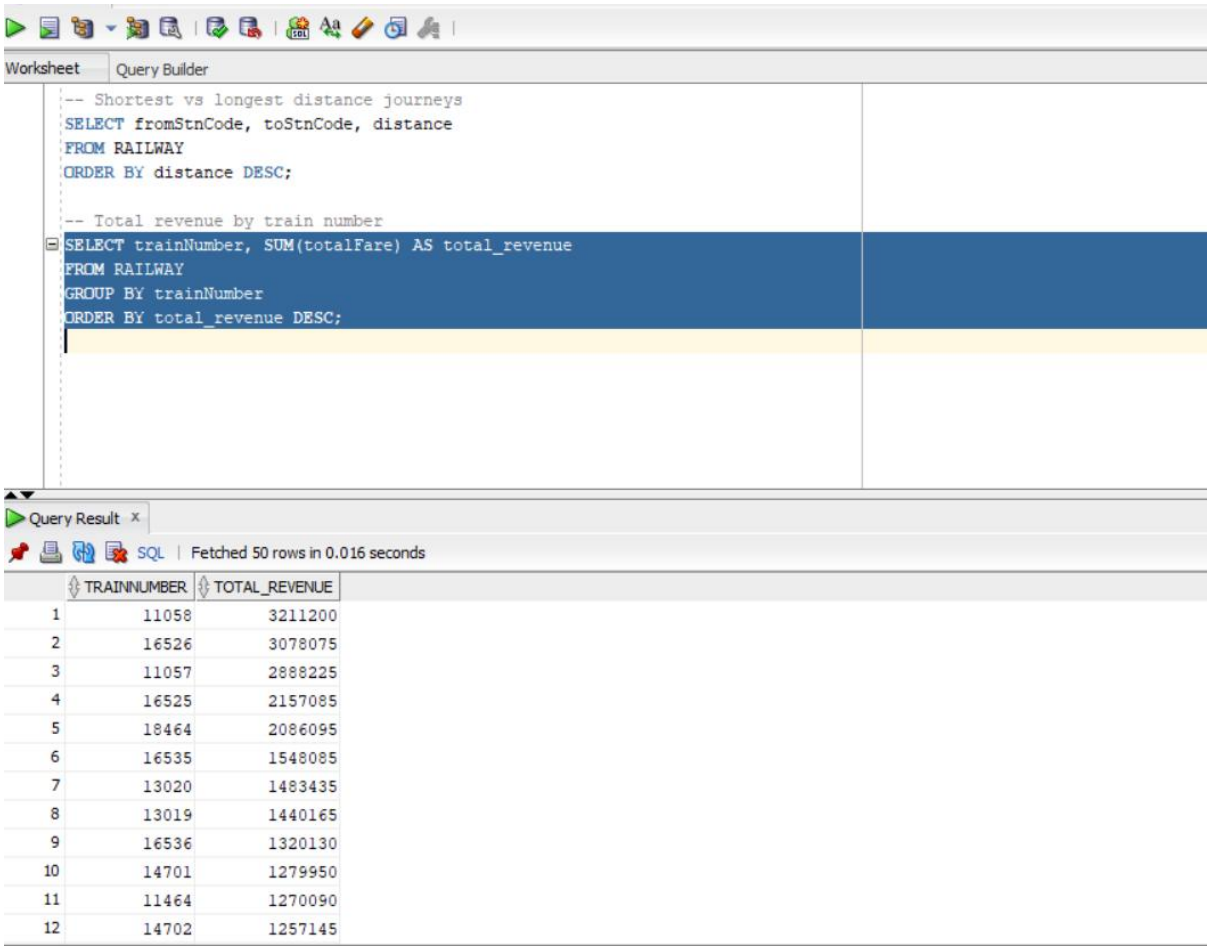
SELECT trainNumber, SUM(totalFare) AS total_revenue

FROM RAILWAY

GROUP BY trainNumber

ORDER BY total_revenue DESC;

Result Screenshot:



	TRAINNUMBER	TOTAL_REVENUE
1	11058	3211200
2	16526	3078075
3	11057	2888225
4	16525	2157085
5	18464	2086095
6	16535	1548085
7	13020	1483435
8	13019	1440165
9	16536	1320130
10	14701	1279950
11	11464	1270090
12	14702	1257145

Insight:
Certain trains generate higher revenue due to route popularity and demand.

Query 5 – Combine expenses with long-distance journeys

SELECT fromStnCode, toStnCode, totalFare, distance

FROM RAILWAY

WHERE totalFare > 1000

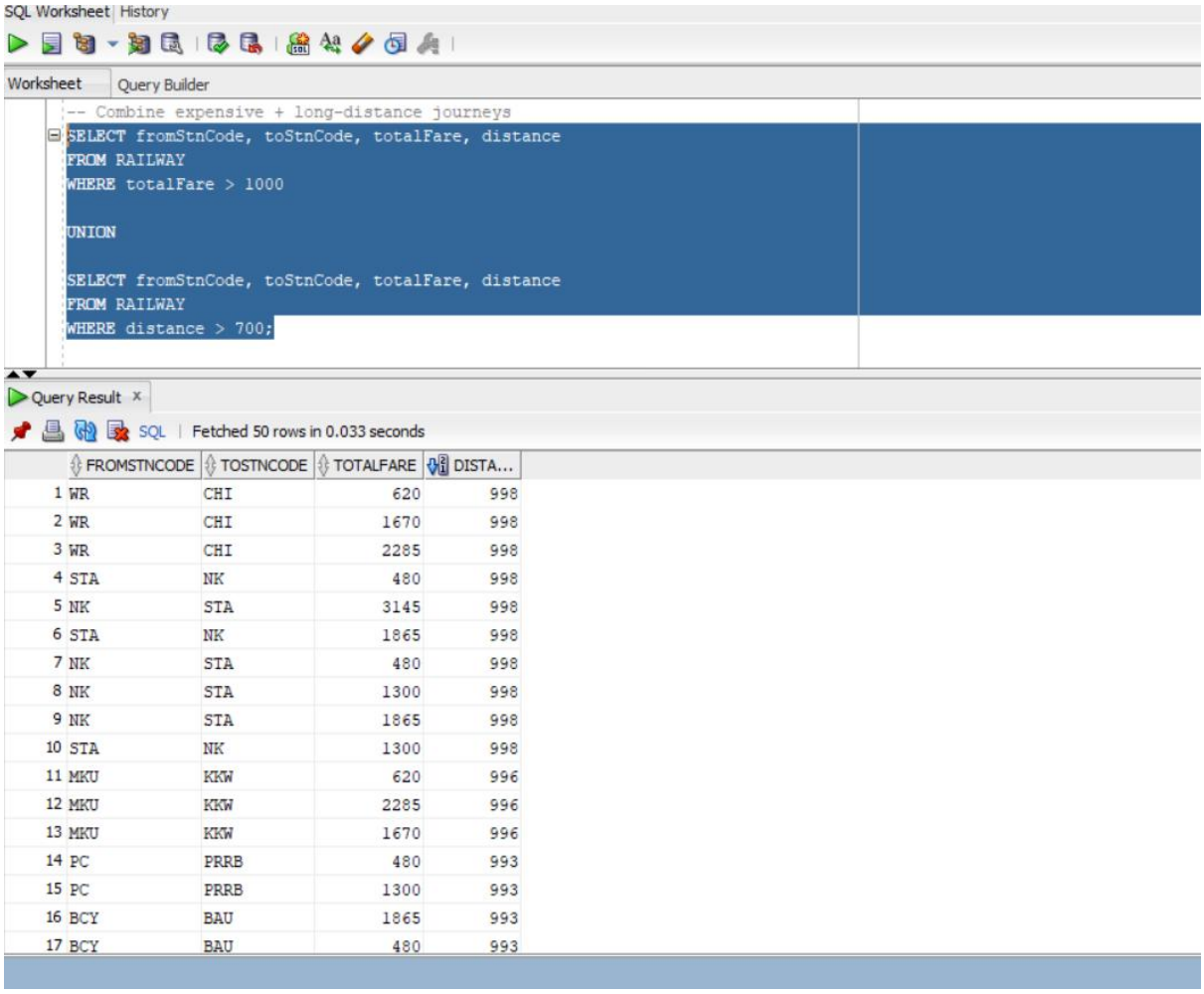
UNION

SELECT fromStnCode, toStnCode, totalFare, distance

FROM RAILWAY

WHERE distance > 700;

Result Screenshot:



	FROMSTNCODE	TOSTNCODE	TOTALFARE	DISTA...
1	WR	CHI	620	998
2	WR	CHI	1670	998
3	WR	CHI	2285	998
4	STA	NK	480	998
5	NK	STA	3145	998
6	STA	NK	1865	998
7	NK	STA	480	998
8	NK	STA	1300	998
9	NK	STA	1865	998
10	STA	NK	1300	998
11	MKU	KKW	620	996
12	MKU	KKW	2285	996
13	MKU	KKW	1670	996
14	PC	PRRB	480	993
15	PC	PRRB	1300	993
16	BCY	BAU	1865	993
17	BCY	BAU	480	993

Insight:
By combining high-fare and long-distance routes using a UNION query, we identified premium train sectors that either generate high revenue due to fare pricing or cover major long-haul corridors. These routes highlight where Indian Railways can focus on improving onboard services, optimizing coach compositions, and potentially implementing dynamic pricing to balance affordability and profitability.

---

4. Key Findings

- Premium train routes and higher travel classes contribute the most to overall revenue.
- Certain long-distance routes have lower fare per km, indicating competitive pricing.
- Data can help optimize pricing and improve class allocation.

---

5. Conclusion

The SQL analysis successfully identified revenue-generating routes, high-cost travel sectors, and fare trends across classes. These insights can support better operational and pricing decisions for Indian Railways.