Fitness Tracker Data Analysis - SQL Case Study

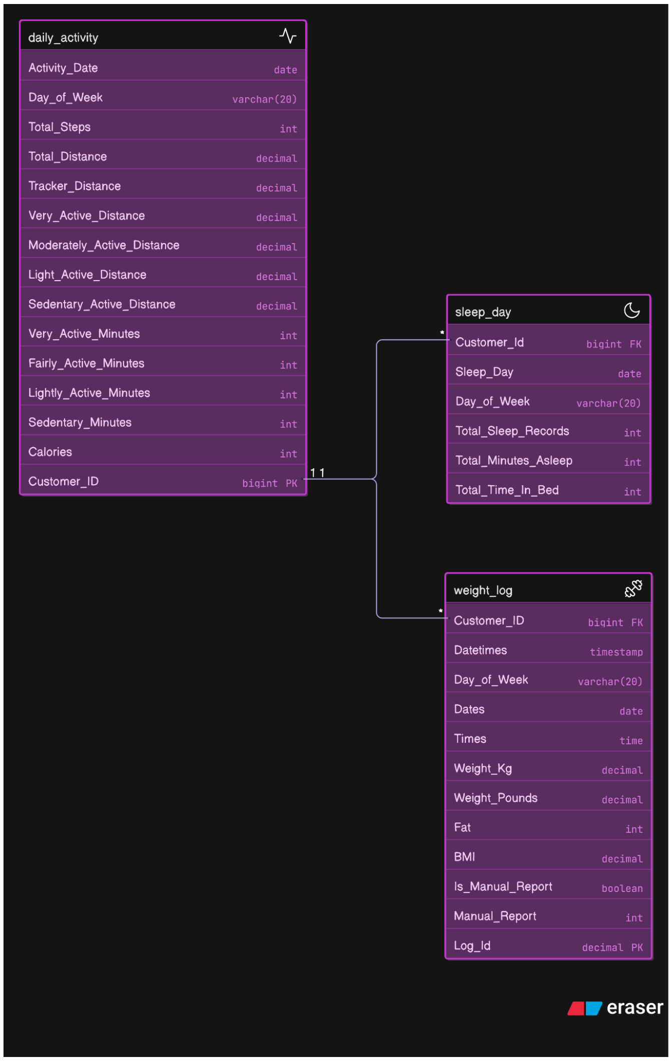
Project Overview

A comprehensive SQL analysis of Bellebeat fitness tracker data, examining user activity patterns, sleep quality, and weight trends to uncover behavioural insights and optimize user engagement strategies.

Dataset: 3 tables with 1,410 total records

- `daily_activity` - 940 activity logs
- `sleep_day` - 410 sleep records
- `weight_log` - 60 weight entries

ERD:



daily_activity	
Activity_Date	date
Day_of_Week	varchar(20)
Total_Steps	int
Total_Distance	decimal
Tracker_Distance	decimal
Very_Active_Distance	decimal
Moderately_Active_Distance	decimal
Light_Active_Distance	decimal
Sedentary_Active_Distance	decimal
Very_Active_Minutes	int
Fairly_Active_Minutes	int
Lightly_Active_Minutes	int
Sedentary_Minutes	int
Calories	int
Customer_ID	bigint PK

sleep_day	
Customer_Id	bigint FK
Sleep_Day	date
Day_of_Week	varchar(20)
Total_Sleep_Records	int
Total_Minutes_Asleep	int
Total_Time_In_Bed	int

weight_log	
Customer_ID	bigint FK
Datetimes	timestamp
Day_of_Week	varchar(20)
Dates	date
Times	time
Weight_Kg	decimal
Weight_Pounds	decimal
Fat	int
BMI	decimal
Is_Manual_Report	boolean
Manual_Report	int
Log_Id	decimal PK

Business Context: Understanding when users are most active, identifying sleep efficiency patterns, and analyzing engagement across different metrics to improve product features and user retention.

Business Questions & Solutions

1. Peak Activity Day Identification

Question: Which day of the week shows maximum and minimum user activity based on steps?

```
WITH active_days AS(
    SELECT
        day_of_week,
        SUM(total_steps),
        FIRST_VALUE(day_of_week) OVER (ORDER BY SUM(total_steps) DESC) AS
most_active,
        FIRST_VALUE(day_of_week) OVER (ORDER BY SUM(total_steps)) AS least_active
    FROM daily_activity
    GROUP BY day_of_week
)
SELECT
    DISTINCT most_active,
    least_active
FROM active_days;
```

Most Active	Least Active
Tuesday	Sunday

Key Insight: Tuesday shows peak user engagement, while Sunday has lowest activity - suggesting potential for targeted weekend motivation campaigns.

Technical Note: Used window functions with `FIRST_VALUE()` to efficiently identify extremes without multiple subqueries.

---

2. Sleep Efficiency Champion

Question: Which user optimizes their sleep time most effectively (minimal time in bed awake)?

```
WITH effectiveness AS(
    SELECT
        customer_id,
        (SUM(total_time_in_bed) - SUM(total_minutes_asleep)) AS wasted_time,
        RANK() OVER(ORDER BY (SUM(total_time_in_bed) - SUM(total_minutes_asleep)))
AS effctv_rank
    FROM sleep_day
    GROUP BY customer_id
)
SELECT
    customer_id
FROM effectiveness
WHERE effctv_rank = 1;
```

Result: Customer 7007744171 (most efficient sleeper)

Key Insight: This user spends minimal time in bed awake, indicating excellent sleep hygiene - potential candidate for case study or beta testing sleep features.

---

3. Sleep Data Gap Analysis

Question: Identify users who haven't logged any sleep data.

Approach 1 - Correlated Subquery (Recommended):

```
SELECT
    DISTINCT d.customer_id
FROM daily_activity d
WHERE NOT EXISTS (
    SELECT customer_id
    FROM sleep_day s
    WHERE s.customer_id = d.customer_id
);
```

Approach 2 - NOT IN Method:

```
SELECT DISTINCT customer_id
FROM daily_activity
WHERE customer_id NOT IN (
    SELECT customer_id FROM sleep_day
);
```

Result: 9 users without sleep data

Key Insight: 27% of active users (9/33) don't track sleep - major opportunity for feature education and onboarding improvements.

Technical Note: `NOT EXISTS` typically performs better than `NOT IN` for large datasets, especially with NULL values.

---

4. Complete Data Users

Question: Which users have logged all three metrics (activity, sleep, weight)?

```
SELECT customer_id FROM daily_activity
INTERSECT
SELECT customer_id FROM weight_log
INTERSECT
SELECT customer_id FROM sleep_day;
```

Alternative using JOINs:

```
SELECT DISTINCT da.customer_id
FROM daily_activity da
JOIN weight_log wl ON da.customer_id = wl.customer_id
JOIN sleep_day sd ON da.customer_id = sd.customer_id;
```

Result: 6 users (power users)

Customer ID
4558609924
6962181067
1503960366
4319703577
5577150313
1927972279

Key Insight: Only 18% of users are fully engaged across all tracking features - these power users are prime candidates for premium features and testimonials.

5. Weekly Sleep Pattern Matrix

Question: Display total sleep hours per user across all days of the week.

Approach 1 - CASE Statement (Universal):

```
SELECT customer_id,
    SUM(CASE WHEN day_of_week = 'Monday' THEN total_minutes_asleep ELSE 0 END) AS
monday,
    SUM(CASE WHEN day_of_week = 'Tuesday' THEN total_minutes_asleep ELSE 0 END) AS
tuesday,
    SUM(CASE WHEN day_of_week = 'Wednesday' THEN total_minutes_asleep ELSE 0 END)
AS wednesday,
    SUM(CASE WHEN day_of_week = 'Thursday' THEN total_minutes_asleep ELSE 0 END)
AS thursday,
    SUM(CASE WHEN day_of_week = 'Friday' THEN total_minutes_asleep ELSE 0 END) AS
friday,
    SUM(CASE WHEN day_of_week = 'Saturday' THEN total_minutes_asleep ELSE 0 END)
AS saturday,
    SUM(CASE WHEN day_of_week = 'Sunday' THEN total_minutes_asleep ELSE 0 END) AS
sunday
FROM sleep_day
GROUP BY customer_id
ORDER BY customer_id;
```

Approach 2 - PostgreSQL CROSSTAB:

```
CREATE EXTENSION tablefunc;

SELECT customer_id,
    COALESCE(monday, 0) AS monday,
    COALESCE(tuesday, 0) AS tuesday,
    COALESCE(wednesday, 0) AS wednesday,
    COALESCE(thursday, 0) AS thursday,
    COALESCE(friday, 0) AS friday,
    COALESCE(saturday, 0) AS saturday,
    COALESCE(sunday, 0) AS sunday
FROM CROSSTAB(
    'SELECT customer_id, day_of_week, SUM(total_minutes_asleep) AS total_sleep
     FROM sleep_day
     GROUP BY customer_id, day_of_week
     ORDER BY customer_id, day_of_week',
    'SELECT DISTINCT day_of_week FROM sleep_day'
) AS result(
    customer_id bigint, monday bigint, tuesday bigint, wednesday bigint,
    thursday bigint, friday bigint, saturday bigint, sunday bigint
);
```

Approach 3 - SQL Server PIVOT:

```
SELECT customer_id,
    COALESCE([Monday], 0) AS monday,
    COALESCE([Tuesday], 0) AS tuesday,
    COALESCE([Wednesday], 0) AS wednesday,
    COALESCE([Thursday], 0) AS thursday,
    COALESCE([Friday], 0) AS friday,
    COALESCE([Saturday], 0) AS saturday,
    COALESCE([Sunday], 0) AS sunday
FROM (
    SELECT customer_id, day_of_week, total_minutes_asleep
    FROM sleep_day
) src
PIVOT (
    SUM(total_minutes_asleep)
    FOR day_of_week IN ([Monday], [Tuesday], [Wednesday], [Thursday], [Friday],
[Saturday], [Sunday])
) p
ORDER BY customer_id;
```

Sample Output:

Customer ID	Mon	Tue	Wed	Thu	Fri	Sat	Sun
1503960366	1532	1436	938	1087	1029	2093	892
2026352035	1358	2130	2459	2057	2561	2060	1548
4319703577	1927	2195	1393	1971	1338	1553	2016

Key Insight: Reveals individual sleep consistency patterns - users with high variability may benefit from sleep coaching features.

Technical Note: CASE statement works universally; PIVOT/CROSSTAB offer cleaner syntax but are database-specific.

---

6. Weight Fluctuation Tracking

Question: For each user, identify dates and values for their highest and lowest recorded weights.

```
SELECT DISTINCT d.customer_id,
    COALESCE(
        FIRST_VALUE(dates || '  (' || weight_kg || ' kgs)')
        OVER (PARTITION BY d.customer_id ORDER BY weight_kg DESC),
        'NA'
    ) AS highest_weight_on,
    COALESCE(
        FIRST_VALUE(dates || '  (' || weight_kg || ' kgs)')
        OVER (PARTITION BY d.customer_id ORDER BY weight_kg),
        'NA'
    ) AS lowest_weight_on
FROM weight_log w
RIGHT JOIN daily_activity d ON d.customer_id = w.customer_id
ORDER BY highest_weight_on;
```

Sample Output:

Customer ID	Highest Weight	Lowest Weight
6962181067	2016-04-12 (62.5 kgs)	2016-05-03 (61 kgs)
4558609924	2016-04-25 (70.3 kgs)	2016-05-09 (69.1 kgs)
8877689391	2016-04-18 (85.8 kgs)	2016-05-12 (84 kgs)

Key Insight: Users tracking weight show 1-2kg fluctuations over tracking period - validates weight tracking feature value for weight management goals.

---

7. Peak Sleep Day Analysis

Question: Which day of the week do users sleep most?

```
WITH most_sleep AS(
    SELECT
        day_of_week,
        SUM(total_minutes_asleep) AS total_sleep_time,
        RANK() OVER(ORDER BY SUM(total_minutes_asleep) DESC) AS rnk
    FROM sleep_day
    GROUP BY day_of_week
)
SELECT day_of_week
FROM most_sleep
WHERE rnk = 1;
```

Result: Wednesday

Key Insight: Mid-week sleep peak suggests users prioritize recovery midweek - opportunity for "Wednesday Wellness" content campaigns.

---

8. Sleep Efficiency by Day

Question: Calculate percentage of time spent in bed actually sleeping for each day.

```
SELECT
    day_of_week,
    (CAST(SUM(total_time_in_bed) AS DECIMAL) - CAST(SUM(total_minutes_asleep) AS
DECIMAL)) AS time_in_bed_without_sleep,
    ROUND((CAST(SUM(total_minutes_asleep) AS DECIMAL) /
CAST(SUM(total_time_in_bed) AS DECIMAL)) * 100, 2) AS pct
FROM sleep_day
GROUP BY day_of_week
ORDER BY pct DESC;
```

Day	Time Awake in Bed	Sleep Efficiency %
Wednesday	2333 min	92.48%
Thursday	2149 min	92.28%
Monday	1741 min	91.72%

Tuesday	2519 min	91.26%
Saturday	2324 min	91.13%
Friday	2259 min	91.10%
Sunday	2792 min	89.92%

Key Insight: Sunday has lowest sleep efficiency (89.92%) - users struggle with "Sunday scaries" affecting sleep quality. Target sleep hygiene tips for weekend nights.

9. Data Coverage Analysis

Question: Which day appears most frequently across all tracking tables?

```
WITH all_days AS (
    SELECT day_of_week FROM daily_activity
    UNION ALL
    SELECT day_of_week FROM weight_log
    UNION ALL
    SELECT day_of_week FROM sleep_day
),
day_counts AS (
    SELECT
        day_of_week,
        COUNT(1) AS occurrence,
        RANK() OVER (ORDER BY COUNT(1) DESC) AS rank_by_frequency
    FROM all_days
    GROUP BY day_of_week
)
SELECT day_of_week AS most_repeated_day_of_week
FROM day_counts
WHERE rank_by_frequency = 1;
```

Result: Wednesday

Key Insight: Wednesday has highest data collection rate across all metrics - suggests consistent mid-week engagement. Use this pattern to time data quality reminders.

10. High-Activity Distance Analysis

Question: For users taking 6000+ daily steps, what's their average walking distance?

```
SELECT
    customer_id,
    ROUND(AVG(total_distance), 2) AS distance_kms
FROM daily_activity
WHERE total_steps > 6000
GROUP BY customer_id
ORDER BY distance_kms DESC;
```

Top 5 Results:

Customer ID	Avg Distance (km)
8877689391	13.53
8053475328	12.48
7007744171	9.78
4388161847	8.76
2022484408	8.28

Key Insight: Top user averages 13.5km on active days - significant variance in stride length and activity type suggests need for personalized distance goals.

Key Findings Summary

User Engagement Patterns

1. **Activity Peaks:** Tuesday shows the highest engagement, Sunday the lowest (potential for weekend motivation campaigns)
2. **Sleep Tracking Gap:** 27% of active users don't log sleep data (feature education opportunity)
3. **Power Users:** Only 18% track all three metrics (target for premium features)
4. **Data Consistency:** Wednesday has the highest logging rate across all features

Sleep Insights

1. **Best Sleepers:** User 7007744171 demonstrates optimal sleep efficiency
2. **Sleep Quality:** Average 89-92% sleep efficiency across the week
3. **Weekend Effect:** Sunday has the lowest sleep efficiency (89.92%)
4. **Peak Sleep Day:** Wednesday shows maximum total sleep time

Behavioral Trends

1. **Weight Tracking:** Limited adoption (60 records), but shows valuable 1-2kg tracking granularity
2. **Activity Levels:** Wide variance in distance per 6000 steps (4.6-13.5km) indicates diverse user types
3. **Consistency:** Mid-week (Tue-Wed) shows the strongest engagement across all metrics

---

Business Recommendations

Immediate Actions

1. **Weekend Engagement Campaign:** Launch Sunday motivation program to boost the lowest-activity day
2. **Sleep Feature Onboarding:** Target 9 users without sleep data with educational content
3. **Power User Program:** Engage 6 complete-data users as beta testers/ambassadors
4. **Wednesday Wellness:** Leverage mid-week engagement peak for feature announcements

Product Enhancements

1. **Sleep Coaching:** Implement Sunday night sleep hygiene tips based on efficiency data
2. **Personalized Goals:** Create stride-length-adjusted distance goals (variance: 4.6-13.5km)
3. **Weight Tracking Incentives:** Increase adoption from the current 60 entries with goal-setting features
4. **Consistency Rewards:** Gamify mid-week tracking streaks where engagement is naturally high

Technical Highlights

Advanced SQL Techniques

- **Window Functions:** `FIRST_VALUE()`, `RANK()`, `PARTITION BY` for efficient ranking
- **CTEs:** Complex multi-step aggregations with improved readability
- **Set Operations:** `INTERSECT`, `UNION ALL` for data overlap analysis
- **Pivoting:** Multiple approaches (CASE, PIVOT, CROSSTAB) for cross-database compatibility
- **Subqueries:** Correlated vs. non-correlated performance optimization

Performance Optimization

- `NOT EXISTS` preferred over `NOT IN` for exclusion logic
- Window functions over self-joins for ranking operations
- Type casting for precise decimal calculations
- RIGHT JOIN with COALESCE for comprehensive null handling

Data Quality Management

- NULL handling with `COALESCE()` and `IS NULL`
- Cross-table validation with INTERSECT
- Gap analysis using exclusion patterns
- String concatenation for user-friendly date displays

Skills Demonstrated

SQL Mastery:

- Window Functions (RANK, FIRST_VALUE, PARTITION BY)
- Common Table Expressions (CTEs)
- Set Operations (INTERSECT, UNION ALL)
- Pivoting Techniques (CASE, PIVOT, CROSSTAB)
- Complex JOINs (INNER, LEFT, RIGHT)
- Correlated Subqueries
- Aggregate Functions
- Date Functions
- Type Casting

Analytical Skills:

- Gap Analysis
- Efficiency Metrics Calculation
- Behavioural Pattern Recognition

- Data Quality Assessment
- Cross-Feature Analysis

Business Intelligence:

- User Segmentation
- Engagement Metrics
- Product Usage Analysis
- Actionable Insight Generation

Impact Metrics

- **Data Coverage:** Analyzed 1,410 records across 3 core features
- **User Base:** 33 unique users tracked
- **Insights Generated:** 10 actionable business questions answered
- **Feature Adoption:** Identified 18% power user segment for targeting
- **Opportunity Sizing:** 27% sleep feature adoption gap quantified

---

Analysis performed using PostgreSQL with cross-database compatible solutions provided for SQL Server (T-SQL)