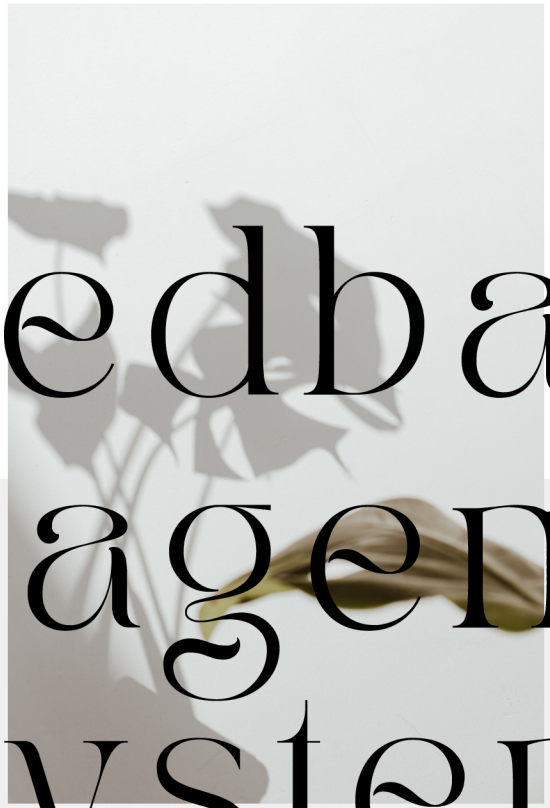
Feedback Management System



OSMONALIEVA AYANA COMFCI-23
SHAKIROVA SAIKAL IEMIT-23

Project Overview

The Feedback Management System is designed to manage and analyze feedback from users about products or services. It provides a database structure, queries, and an application for managing user feedback effectively.

Project Objectives

- - Create a relational database for managing user feedback
- - Store product and user information
- - Allow users to submit feedback and ratings
- - Provide insights through data analysis and SQL queries

Entities:

- User: Stores user details like name, email, and role
- Product: Stores product details like name and category
- Feedback: Stores feedback details like rating, comment, and date

Relationships:

- User to Feedback: One-to-Many
- Product to Feedback: One-to-Many

Relational Model (Tables)

- 1. User(user_id, name, email, role, password)
- 2. Product(item_id, name, category, description)
- 3. Feedback(feedback_id, rating, comment, feedback_date, user_id, item_id)

Primary and Foreign Keys:

- - Primary Keys: user_id, item_id, feedback_id
- - Foreign Keys: user_id (Feedback), item_id (Feedback)

SQL Queries Examples

- Retrieve feedback with product and user details
- Calculate average ratings for a product
- Find products with low ratings (< 3)
- Identify users who have not submitted feedback
- Find the most active users by feedback count

Data Generation

- Used Python script to generate sample data
 - Populated tables with realistic data:
- Users: Names, emails, roles
 - Products: Names, categories, descriptions
- Feedback: Ratings, comments, and dates

Application Demonstration

Features of the Feedback Management System:
- User registration and authentication
- Adding and managing feedback
- Viewing top-rated and low-rated products
- Statistical analysis of feedback

Query Questions for Feedback System

1. RETRIEVE A LIST OF FEEDBACK WITH PRODUCT DETAILS AND USER NAMES:

```
SELECT
F.FEEDBACK_ID, U.NAME AS USER_NAME,
P.NAME AS  PRODUCT_NAME, F.RATING,
F.COMMENT, F.FEEDBACK_DATE
FROM
FEEDBACK F
JOIN
USER U ON F.USER_ID = U.USER_ID
JOIN
PRODUCT P ON F.ITEM_ID = P.ITEM_ID;
```

2. **FIND THE AVERAGE RATING FOR A SPECIFIC PRODUCT:**

```
SELECT
P.NAME AS PRODUCT_NAME, AVG(F.RATING) AS
AVERAGE_RATING
FROM
FEEDBACK F
JOIN
PRODUCT P ON F.ITEM_ID = P.ITEM_ID
WHERE
P.ITEM_ID = 1
GROUP BY
P.NAME;
```

3. LIST PRODUCTS WITH MORE THAN 5 FEEDBACKS:

```
SELECT
P.NAME AS PRODUCT_NAME,
COUNT(F.FEEDBACK_ID) AS FEEDBACK_COUNT
FROM
FEEDBACK F
JOIN
PRODUCT P ON F.ITEM_ID = P.ITEM_ID
GROUP BY
P.NAME
HAVING
COUNT(F.FEEDBACK_ID) > 5;
```

4. IDENTIFY USERS WHO HAVE SUBMITTED FEEDBACK:

```
SELECT
DISTINCT U.NAME AS USER_NAME, U.EMAIL
FROM
FEEDBACK F
JOIN
USER U ON F.USER_ID = U.USER_ID;
```

5. RETRIEVE PRODUCTS WITH AVERAGE RATINGS BELOW 3:

```
SELECT
P.NAME AS PRODUCT_NAME, AVG(F.RATING)
AS AVERAGE_RATING
FROM
FEEDBACK F
JOIN
PRODUCT P ON F.ITEM_ID = P.ITEM_ID
GROUP BY
P.NAME
HAVING
AVG(F.RATING) < 3;
```

6. FIND THE LATEST FEEDBACK LEFT BY USERS:

```
SELECT
U.NAME AS USER_NAME, P.NAME AS PRODUCT_NAME, F.RATING, F.COMMENT, F.FEEDBACK_DATE
FROM
FEEDBACK F
JOIN
USER U ON F.USER_ID = U.USER_ID
JOIN
PRODUCT P ON F.ITEM_ID = P.ITEM_ID
ORDER BY
F.FEEDBACK_DATE DESC LIMIT 10;
```

7. RETRIEVE THE TOP N USERS WHO GAVE THE MOST FEEDBACK:

```
SELECT
U.NAME AS USER_NAME, COUNT(F.FEEDBACK_ID) AS FEEDBACK_COUNT
FROM
FEEDBACK F
JOIN
USER U ON F.USER_ID = U.USER_ID
GROUP BY
U.NAME
ORDER BY
FEEDBACK_COUNT DESC LIMIT 5;
```

8. CALCULATE THE TOTAL NUMBER OF FEEDBACKS RECEIVED PER PRODUCT:

```
SELECT
P.NAME AS PRODUCT_NAME, COUNT(F.FEEDBACK_ID) AS TOTAL_FEEDBACKS
FROM
FEEDBACK F
JOIN
PRODUCT P ON F.ITEM_ID = P.ITEM_ID
GROUP BY
P.NAME;
```

9. IDENTIFY USERS WHO HAVE NOT SUBMITTED ANY FEEDBACK:

```
SELECT
U.NAME AS USER_NAME, U.EMAIL
FROM
USER U
LEFT JOIN
FEEDBACK F ON U.USER_ID = F.USER_ID
WHERE
F.FEEDBACK_ID IS NULL;
```

10. FIND THE HIGHEST-RATED PRODUCT:

```
SELECT
P.NAME AS PRODUCT_NAME, AVG(F.RATING) AS AVERAGE_RATING
FROM
FEEDBACK F
JOIN
PRODUCT P ON F.ITEM_ID = P.ITEM_ID
GROUP BY
P.NAME
ORDER BY
AVERAGE_RATING DESC LIMIT 1;
```

Conclusion

THE FEEDBACK MANAGEMENT SYSTEM DEMONSTRATES:

- EFFECTIVE DATABASE DESIGN AND IMPLEMENTATION
- PROFICIENCY IN SQL QUERIES FOR DATA ANALYSIS
- INTEGRATION OF A USER-FRIENDLY APPLICATION FOR MANAGING

FEEDBACK FUTURE IMPROVEMENTS:

- ADD MORE ADVANCED ANALYTICS AND REPORTING
- ENHANCE THE USER INTERFACE FOR BETTER ACCESSIBILITY