Banking Data Management with MySQL

Mohammad Suhail

Problem Statement

The ICIC Bank database system lacked a structured and efficient solution for managing critical banking operations, such as:

- Retrieving employee and customer information based on specific criteria.
- Analyzing account details, balances, and transaction patterns.
- Ensuring seamless integration of multiple departments and job roles within the organization.
- Generating dynamic reports for operational insights.

To address these challenges, a robust relational database was designed and implemented using MySQL. to store, manage, and retrieve banking data efficiently while maintaining data integrity and consistency.

Problem Statement

Challenges Addressed :-

1. Data Redundancy:
   - Ensuring no duplicate records across different departments and customer accounts.
2. Efficient Data Retrieval:
   - Managing complex queries to retrieve customer and employee details across multiple tables.
3. Data Integrity.
   - Maintaining accurate relationships between customers, accounts, employees, and departments.
4. Scalability:
   - Handling a large dataset across multiple branches with flexibility for future growth.
5. Query Optimization:
   - Designing optimized queries for fast retrieval of data, minimizing response time.
6. Security:
   - Ensuring sensitive customer information like ATM numbers and PINs are securely managed.

About the Data

The data consists of six key tables:

1.Customers

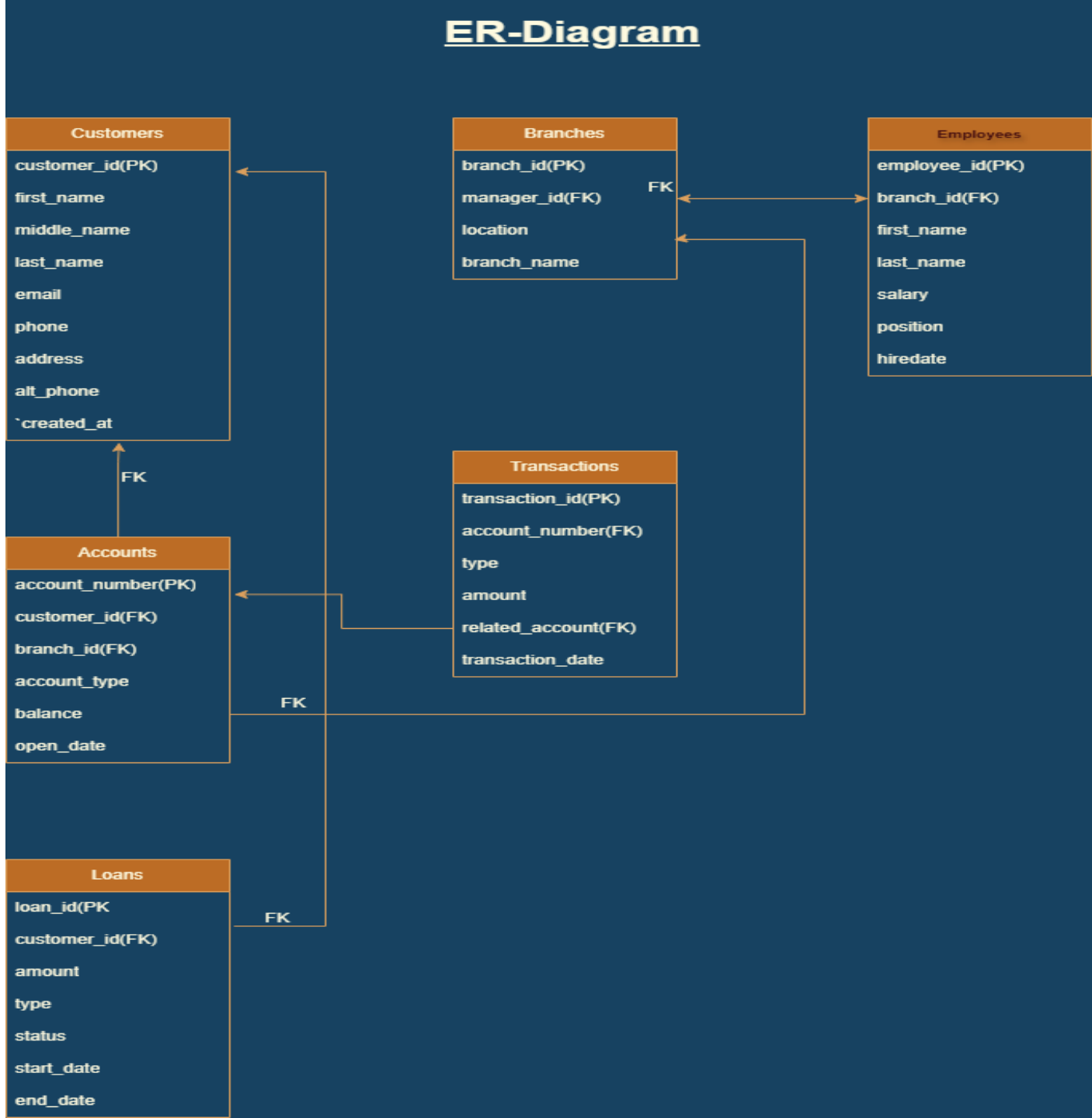
2.Employees

3.Loans

4.Transactions

5.Branches

6.Accounts



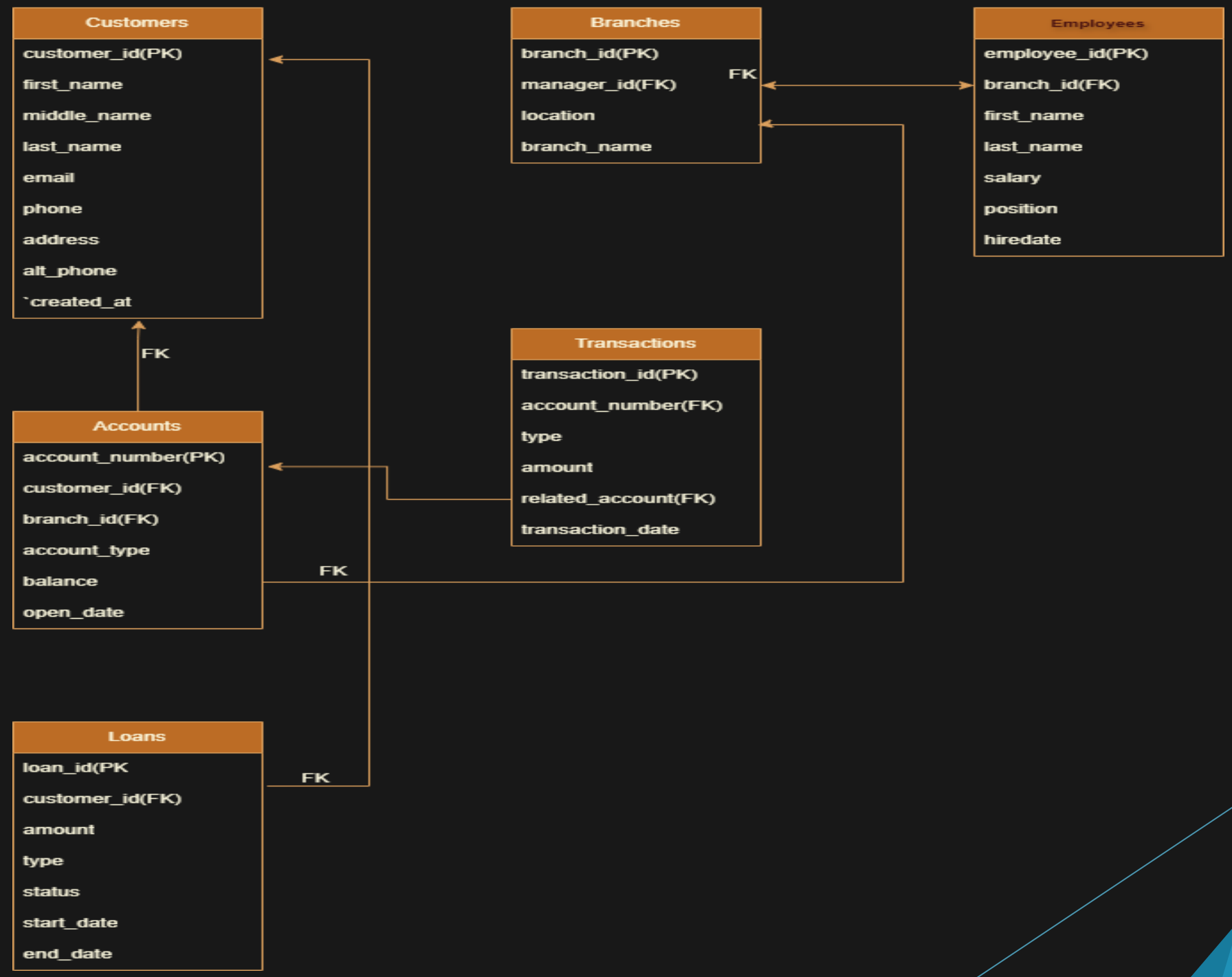

Customers
customer_id(PK)
first_name
middle_name
last_name
email
phone
address
alt_phone
`created_at
Branches
branch_id(PK)
manager_id(FK)
location
branch_name
FK
Employees
employee_id(PK)
branch_id(FK)
first_name
last_name
salary
position
hiredate
FK
Transactions
transaction_id(PK)
account_number(FK)
type
amount
related_account(FK)
transaction_date
Accounts
account_number(PK)
customer_id(FK)
branch_id(FK)
account_type
balance
open_date
FK
Loans
loan_id(PK
customer_id(FK)
amount
type
status
start_date
end_date
FK

Table And Data

```
1 •   select * from customers;
2
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_id	first_name	middle_name	last_name	email	phone	alt_phone	address
1	Rahul	Kumar	Sharma	rahul@example.com	9876543210	NULL	Mumbai
2	Priya	NULL	Patel	priya@example.com	9876543211	9876543212	Delhi
3	Amit	Ramesh	Verma	amit@example.com	9876543213	NULL	Bangalore
4	Sneha	NULL	Singh	sneha@example.com	9876543214	NULL	Hyderabad
5	Vikram	Raj	Gupta	vikram@example.com	9876543215	9876543216	Chennai
6	Anjali	Priya	Mehta	anjali@example.com	9876543217	NULL	Kolkata
7	Ravi	NULL	Joshi	ravi@example.com	9876543218	9876543219	Pune
8	Neha	Sunil	Malhotra	neha@example.com	9876543220	NULL	Ahmedabad
9	Sanjay	NULL	Reddy	sanjay@example.com	9876543221	NULL	Jaipur
10	Pooja	Anil	Desai	pooja@example.com	9876543222	9876543223	Lucknow
11	Arun	NULL	Iyer	arun@example.com	9876543224	NULL	Chandigarh
12	Kavita	Vijay	Rao	kavita@example.com	9876543225	9876543226	Bhopal
13	Rajesh	NULL	Thakur	rajesh@example.com	9876543227	NULL	Surat
14	Swati	Mohan	Chopra	swati@example.com	9876543228	9876543229	Nagpur

customers 2

Proposed solution

```
#17.Show transactions with customer names.
SELECT t.transaction_id, c.first_name, t.amount
FROM Transactions t
JOIN Accounts a ON t.account_number = a.account_number
JOIN Customers c ON a.customer_id = c.customer_id;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_id	first_name	middle_name	last_name	email	phone	alt_phone	address
1	Rahul	Kumar	Sharma	rahul@example.com	9876543210	NULL	Mumbai
2	Priya	NULL	Patel	priya@example.com	9876543211	9876543212	Delhi
3	Amit	Ramesh	Verma	amit@example.com	9876543213	NULL	Bangalore
4	Sneha	NULL	Singh	sneha@example.com	9876543214	NULL	Hyderabad
5	Vikram	Raj	Gupta	vikram@example.com	9876543215	9876543216	Chennai
6	Anjali	Priya	Mehta	anjali@example.com	9876543217	NULL	Kolkata
7	Ravi	NULL	Joshi	ravi@example.com	9876543218	9876543219	Pune
8	Neha	Sunil	Malhotra	neha@example.com	9876543220	NULL	Ahmedabad
9	Sanjay	NULL	Reddy	sanjay@example.com	9876543221	NULL	Jaipur
10	Pooja	Anil	Desai	pooja@example.com	9876543222	9876543223	Lucknow
11	Arun	NULL	Iyer	arun@example.com	9876543224	NULL	Chandigarh
12	Kavita	Vijay	Rao	kavita@example.com	9876543225	9876543226	Bhopal
13	Rajesh	NULL	Thakur	rajesh@example.com	9876543227	NULL	Surat
14	Swati	Mohan	Chopra	swati@example.com	9876543228	9876543229	Nagpur

ustomers 3

Proposed solution

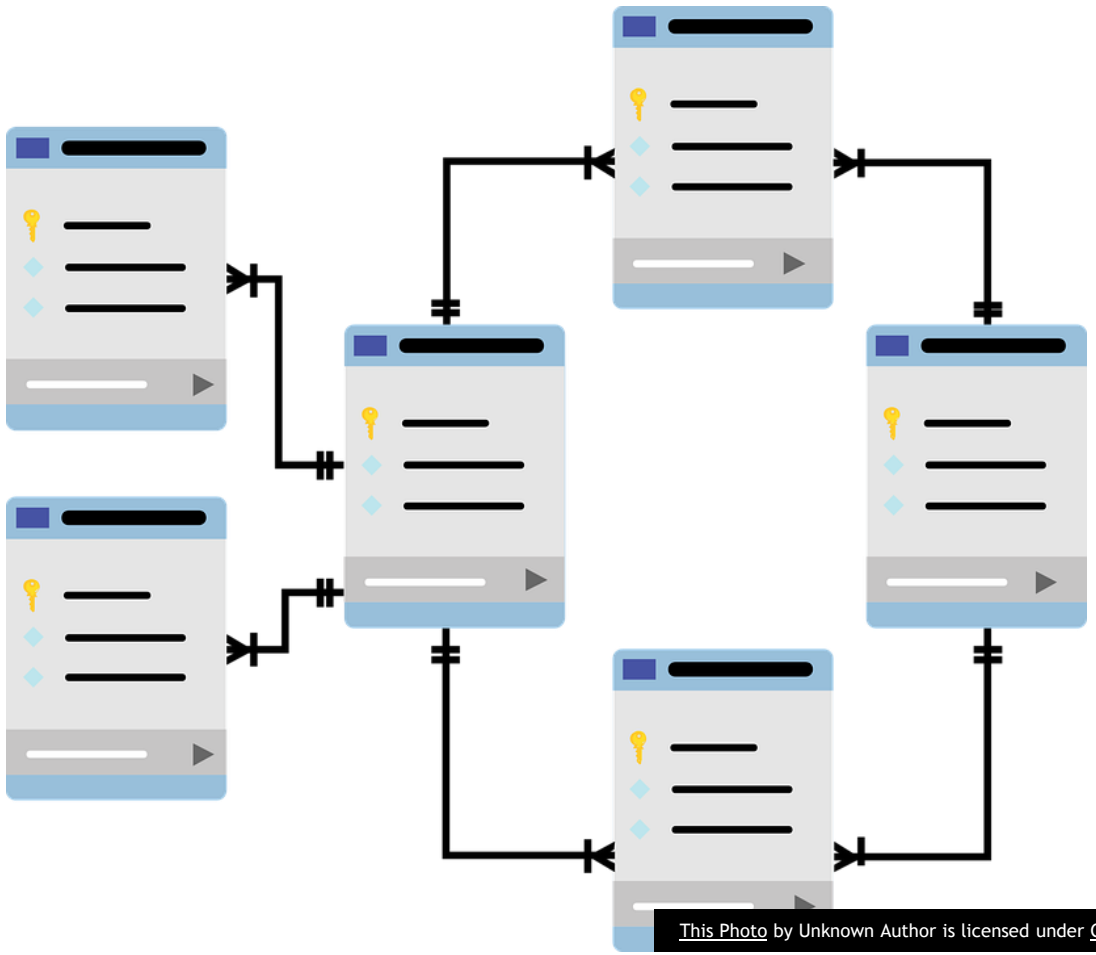
```sql
# 28.List branches with total employee salaries.
SELECT b.branch_name, SUM(e.salary) AS total_salary
FROM Branches b
LEFT JOIN Employees e ON b.branch_id = e.branch_id
GROUP BY b.branch_id;
```

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

transaction_id	first_name	amount
1	Rahul	10000.00
2	Priya	5000.00
3	Amit	20000.00
4	Sneha	15000.00
5	Vikram	10000.00
6	Anjali	3000.00
7	Ravi	50000.00
8	Neha	12000.00
9	Sanjay	8000.00
10	Pooja	7000.00
11	Arun	25000.00
12	Kavita	4500.00
13	Rajesh	9000.00

Conclusion

- Created a working database system to manage bank operations smoothly.
- Used SQL queries to easily find and update important information about customers and employees.
- Improved how the bank handles data, making it easier for different departments to access and use the information.
- This project helps the bank work more efficiently, leading to better decisions and improved customer service.



Future Scope

- Advanced Analytics: Use Al to analyze customer behavior and predict trends.
- Task Automation: Automate routine tasks like loan approvals with Al.
- Data Security: Improve security to protect customer information.
- Mobile Integration: Create a mobile-friendly system for easier access.
- Scalability: Expand the database to support more customers and services