Task8 : Normalizing Database using functional dependencies up to BCNF
3/9/23

Aim: To prove normalizing database using functional dependencies up to BCNF

Determine Possible functional Dependencis(FDs)

Cust → ID → Cust-Name, Cust_PhoneNo, Cust-City,

Bill_ID → Price, cust_ID

phone_ID → Model_Name

Admin_ID → Password

Convert to 1NF

Already in 1NF

All attributes have atomic single values;

there are no repeating groups

Convert to 2NF

Check for partial dependencies_ they occur only if theires a composite key.

Since all tables have singe attributes primary keys

Cust-IP, Bill-ID, Phone_ID, Admin-ID)

Convert to 3NF

1) Ensure there are no transitive dependencies
2) Move non-key attributs to seperate tables if they depend on another key.

Cust-ID → cust_Name, Cust_Phone NO

Cust-city, Cust-Amount Paid

Convert to BCNF

1) Ensure every determinant is a candidate key
2) Check for overlapping Candidate keys
3) Decompose relations to eliminate redundancy

No fusther decomposition needed.

Using Griffith Tool

1) Input relational Schema and functional dependencies
2) Griffith tool generates a dependency graph.
3) Analyze the graph to identify normalization issue
4) Apply normalization rules to transform the Schema.
5) verify the resulting schema meets BCNF Criteria

Griffith Tool steps

1) Create a new project in Griffith
2) Define the relational Schema and FD's
3) Analyze the graph fix normalization issues
4) Apply transformation using the "Normalization" tool
5) Verify BCNF compliance using "BCNF check" tool

Normalized Schema:

Customer (Cust-ID PK, Cust-Name, Cust-Phoneno, Cust-city, Cust-Amount Paid)

Bill (Bill_ID PK, Price, Cust-ID FK → Customer-Cust-ID)

MOBILE (Phone_ID PK, Model-Name, Model price)

LOGIN (Admin_ID PK, Password).

VEL TECH	
EX NO.	4
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	4
RECORD (5)	
TOTAL (20)	14
...TH DATE	

Result: Thus to prove normalizing database using functional dependencies upto BCNF are verified successfully