|  |  |
| --- | --- |
| **Name** | Adwait S Purao |
| **UID no.** | 2021300101 |
| **Experiment No.** | 10 |

|  |  |
| --- | --- |
| **AIM:** | To apply the concepts of Normalization |
| **Program 1** | |
| **PROBLEM STATEMENT :** | . |
| **Theory :** | **Normalization** A large database defined as a single relation may result in data duplication. This repetition of data may result in:   * Making relations very large. * It isn't easy to maintain and update data as it would involve searching many records in relation. * Wastage and poor utilization of disk space and resources. * The likelihood of errors and inconsistencies increases.   So to handle these problems, we should analyze and decompose the relations with redundant data into smaller, simpler, and well-structured relations that are satisfy desirable properties. Normalization is a process of decomposing the relations into relations with fewer attributes.   **What is Normalization?**  * Normalization is the process of organizing the data in the database. * Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies. * Normalization divides the larger table into smaller and links them using relationships. * The normal form is used to reduce redundancy from the database table.   Why do we need Normalization?  The main reason for normalizing the relations is removing these anomalies. Failure to eliminate anomalies leads to data redundancy and can cause data integrity and other problems as the database grows. Normalization consists of a series of guidelines that helps to guide you in creating a good database structure.  **Data modification anomalies can be categorized into three types:**   * **Insertion Anomaly:** Insertion Anomaly refers to when one cannot insert a new tuple into a relationship due to lack of data. * **Deletion Anomaly:** The delete anomaly refers to the situation where the deletion of data results in the unintended loss of some other important data. * **Updatation Anomaly:** The update anomaly is when an update of a single data value requires multiple rows of data to be updated.  **Types of Normal Forms:** Normalization works through a series of stages called Normal forms. The normal forms apply to individual relations. The relation is said to be in particular normal form if it satisfies constraints.  **Following are the various types of Normal forms:**  DBMS Normalization   **Advantages of Normalization**  * Normalization helps to minimize data redundancy. * Greater overall database organization. * Data consistency within the database. * Much more flexible database design. * Enforces the concept of relational integrity.  **Disadvantages of Normalization**  * You cannot start building the database before knowing what the user needs. * The performance degrades when normalizing the relations to higher normal forms, i.e., 4NF, 5NF. * It is very time-consuming and difficult to normalize relations of a higher degree. * Careless decomposition may lead to a bad database design, leading to serious problems.  **First Normal Form (1NF)**  * A relation will be 1NF if it contains an atomic value. * It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute. * First normal form disallows the multi-valued attribute, composite attribute, and their combinations.   **Example:** Relation EMPLOYEE is not in 1NF because of multi-valued attribute EMP\_PHONE.     **Second Normal Form (2NF)**  * In the 2NF, relational must be in 1NF. * In the second normal form, all non-key attributes are fully functional dependent on the primary key   **Example:** Let's assume, a school can store the data of teachers and the subjects they teach. In a school, a teacher can teach more than one subject.    In the given table, non-prime attribute TEACHER\_AGE is dependent on TEACHER\_ID which is a proper subset of a candidate key. That's why it violates the rule for 2NF.  To convert the given table into 2NF, we decompose it into two tables:     **Third Normal Form (3NF)**  * A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency. * 3NF is used to reduce the data duplication. It is also used to achieve the data integrity. * If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.   A relation is in third normal form if it holds atleast one of the following conditions for every non-trivial function dependency X → Y.   1. X is a super key. 2. Y is a prime attribute, i.e., each element of Y is part of some candidate key.   **Example:**  **EMPLOYEE\_DETAIL table:**    **Super key in the table above:**    **Candidate key:** {EMP\_ID}  **Non-prime attributes:** In the given table, all attributes except EMP\_ID are non-prime.  Here, EMP\_STATE & EMP\_CITY dependent on EMP\_ZIP and EMP\_ZIP dependent on EMP\_ID. The non-prime attributes (EMP\_STATE, EMP\_CITY) transitively dependent on super key(EMP\_ID). It violates the rule of third normal form.  That's why we need to move the EMP\_CITY and EMP\_STATE to the new <EMPLOYEE\_ZIP> table, with EMP\_ZIP as a Primary key.  **EMPLOYEE table:**    **EMPLOYEE\_ZIP table:**   **Boyce Codd normal form (BCNF)**  * BCNF is the advance version of 3NF. It is stricter than 3NF. * A table is in BCNF if every functional dependency X → Y, X is the super key of the table. * For BCNF, the table should be in 3NF, and for every FD, LHS is super key.   **Example:** Let's assume there is a company where employees work in more than one department.  **EMPLOYEE table:**      **Candidate key: {EMP-ID, EMP-DEPT}**  The table is not in BCNF because neither EMP\_DEPT nor EMP\_ID alone are keys.  To convert the given table into BCNF, we decompose it into three tables: |
| **Queries** | **Query 1: Converting a table into 1 NF form**  **Statement** : Remove the multiple values in column Phone\_no  **Code :**  CREATE database NORMALIZATION;  USE NORMALIZATION;  CREATE TABLE EMPLOYEES(  Emp\_ID int NOT NULL,  Emp\_Name varchar(255),  Company\_Name varchar(255),  Phone\_No varchar(255),  Emp\_Sal float  );  INSERT INTO EMPLOYEES  VALUES (12379,'Adwait Purao','Ethereum',"1235467890,3342289346", 168512.34);  INSERT INTO EMPLOYEES  VALUES (87121,'Pranav Nalawade','Google',"9876542130,2617312344", 234601.32);  INSERT INTO EMPLOYEES  VALUES (87121,'Pranav Nalawade','Rockstar','2617312344', 91312.34);  INSERT INTO EMPLOYEES  VALUES (98323,'Jay Nadkarni','Apple','8356289346', 689512.34);  INSERT INTO EMPLOYEES  VALUES (51239,'Vikas Patel','Blockchain Inc','1343289346', 689512.34);  CREATE TABLE EMPLOYEESn(  Emp\_ID int NOT NULL,  Emp\_Name varchar(255),  Company\_Name varchar(255),  Phone\_No1 bigint,  Phone\_No2 bigint,  Emp\_Sal float  );  INSERT INTO EMPLOYEESn  VALUES (12379,'Adwait Purao','Ethereum',1235467890,3342289346, 168512.34);  INSERT INTO EMPLOYEESn  VALUES (87121,'Pranav Nalawade','Google',9876542130,2617312344, 234601.32);  INSERT INTO EMPLOYEESn  VALUES (87121,'Pranav Nalawade','Rockstar',2617312344,null, 91312.34);  INSERT INTO EMPLOYEESn  VALUES (98323,'Jay Nadkarni','Apple',8356289346,null, 689512.34);  INSERT INTO EMPLOYEESn  VALUES (51239,'Vikas Patel','Blockchain Inc',1343289346,null, 689512.34);  select \* from employees;  select \* from employeesn;  **Original Table:**    **Table in 1NF :**    **Query 2: Converting a table into 2NF Form**  **Statement** : Split Employee table into two tables Employee List and Details  **Code :**  CREATE TABLE EMPLOYEE\_LIST(  Emp\_ID int NOT NULL,  Emp\_Name varchar(255)  );  CREATE TABLE DETAILS(  Emp\_ID int NOT NULL,  Company\_Name varchar(255),  Phone\_No1 bigint,  Phone\_No2 bigint,  Emp\_Sal float  );  INSERT INTO EMPLOYEE\_LIST  VALUES (12379,'Adwait Purao');  INSERT INTO EMPLOYEE\_LIST  VALUES (87121,'Pranav Nalawade');  INSERT INTO EMPLOYEE\_LIST  VALUES (98323,'Jay Nadkarni');  INSERT INTO EMPLOYEE\_LIST  VALUES (51239,'Vikas Patel');  INSERT INTO DETAILS  VALUES (12379,'Ethereum',3342289346,1235467890, 168512.34);  INSERT INTO DETAILS  VALUES (87121,'Google',2617312344,9876542130, 234601.32);  INSERT INTO DETAILS  VALUES (98323,'Apple',8356289346,null, 689512.34);  INSERT INTO DETAILS  VALUES (51239,'Blockchain Inc',1343289346,null, 689512.34);  select \* from employee\_list;  select \* from details;  **Output :**  **Original Table:**    **New Table 1:**    **New Table 2:**    **Query 3: Converting a table into 3NF Form**  **Statement** : Split EmpAdresss table into two tables  **Code :**  create table EmpAddress(  Emp\_ID int,  Fullname varchar(255),  zipcode int,  state varchar(255),  city varchar(255)  );  insert into EmpAddress values(12379,'Adwait Purao',200290,"Maharashtra","Mumbai");  insert into EmpAddress values(87121,'Pranav Nalawade',349940,"Karnataka","Bengaluru");  insert into EmpAddress values(98323,'Jay Nadkarni',321200,"Punjab","Amritsar");  insert into EmpAddress values(51239,'Vikas Patel',669294,"Kerala","Palakkad");  create table address1(  Emp\_ID int,  Fullname varchar(255),  zipcode int  );  insert into address1 values (12379,'Adwait Purao',200290);  insert into address1 values (87121,'Pranav Nalawade',349940);  insert into address1 values (98323,'Jay Nadkarni',321200);  insert into address1 values (51239,'Vikas Patel',669294);  create table address2(  zipcode int,  state varchar(255),  city varchar(255)  );  insert into address2 values(200290,"Maharashtra","Mumbai");  insert into address2 values(349940,"Karnataka","Bengaluru");  insert into address2 values(321200,"Punjab","Amritsar");  insert into address2 values(669294,"Kerala","Palakkad");  select \* from empaddress;  select \* from address1;  select \* from address2;  **Output :**  **Original Table:**    **New Table 1:**    **New Table 2:**    **Query 4: Converting a table into BCNF Form**  **Statement** : Split Supportstaff table into two tables  **Code:**  create table Supportstaff(  StaffID int,  Staffstate varchar(255),  Company varchar(255),  depttype varchar(255),  DeptID int  );  insert into Supportstaff values(1,"Maharahtra","Ethereum","Computer",100);  insert into Supportstaff values(1,"Maharahtra","Google","Electronics",101);  insert into Supportstaff values(2,"Karnataka","Apple","Marketing",365);  insert into Supportstaff values(2,"Karnataka","Blockchain Inc.","Computer",729);  create table staff1(  StaffID int,  Staffstate varchar(255)  );  insert into staff1 values (1,"Maharahstra");  insert into staff1 values (2,"Karnataka");  create table staff2(  Company varchar(255),  depttype varchar(255),  DeptID int  );  insert into staff2 values("Ethereum","Computer",100);  insert into staff2 values("Google","Electronics",101);  insert into staff2 values("Apple","Marketing",365);  insert into staff2 values("Blockchain Inc.","Computer",729);  create table staff3(  StaffID int,  DeptID int  );  insert into staff3 values(1,100);  insert into staff3 values(1,101);  insert into staff3 values(2,365);  insert into staff3 values(2,729);  select \* from Supportstaff;  select \* from staff1;  select \* from staff2;  select \* from staff3;  **Output :**  **Original Table:**    **New Table 1:**    **New Table 2:**    **New Table 3:**    **Query 5**: **Converting a table into 4 NF Form**  **Statement** : Split CEOTable table into two tables  **Code:**  create table CEOTable(  FullName varchar(255),  Company varchar(255),  Hobby varchar(255)  );  insert into CEOTable values ("Vitalik Buterin","Ethereum","Reading Books");  insert into CEOTable values ("Sundar Pichai","Google","Football");  insert into CEOTable values ("Tim Cook","Apple","Singing");  insert into CEOTable values ("Jay Nadkarni","Blockchain","Dancing");  create table CEO2(  FullName varchar(255),  Company varchar(255)  );  insert into CEO2 values ("Vitalik Buterin","Ethereum");  insert into CEO2 values ("Sundar Pichai","Google");  insert into CEO2 values ("Tim Cook","Apple");  insert into CEO2 values ("Jay Nadkarni","Blockchain");  create table CEO3(  FullName varchar(255),  Hobby varchar(255)  );  insert into CEO3 values ("Vitalik Buterin","Reading Books");  insert into CEO3 values ("Sundar Pichai","Football");  insert into CEO3 values ("Tim Cook","Singing");  insert into CEO3 values ("Jay Nadkarni","Dancing");  select \* from CEOTable;  select \* from CEO2;  select \* from CEO3;  **Output :**  **Original Table:**    **New Table 1:**    **New Table 2:** |
| **Conclusion**  From this experiment, we learnt about normalisation . We saw the various forms of normalisation such as 1 NF, 2 NF, 3 NF, BC NF, 4 NF.. We saw the advantages of normalisation such as removing redundancy in the tables and also we can minimize issues with data modification . We also learnt about Insertion , Updation and Deletion anomaly. | |