Task 10

Normalizing databases using functional dependencies upto Normal form

Aim: to normalize the below relation and create the simplified table and with suitable constraints.

cricket Board (BoardID, Name, Address, contact-No, TeamID, TName, coach, cntact no, Batting, Bowling, MatchID, Match-Date & time, Result, GroundID, GName, location, capacity, umpire ID, UFName, ULName, U Dateof Birth, country, Usemail, Ucontact-no).

- Apply the functional dependency, normalize to 1NF
- Normalize the relations using FDt and αt
- Find the minimal cover, canonical cover.
- Normalize to 2NF (add/alter constraints if necessary to 3NF, add/alter constraints if necessary

Procedure:

Normalize the given relation and create simplified tables with suitable constraints, we need to identify the functional dependencies and separate them into different tables: Normalization involves breaking down the data into smaller, related tables to minimize data redundancy and maintain data Integrity.

Functional Dependency:

BoardID → Name, Address, contact-No

TeamID → TName, coach, captain

PlayerID → PFName, PLName, Age, P Date of Birth, Playing Role, email.

MatchID → Match-Date, time, Result, GroundID

GroundID → GName, location, Capacity

UmpireID → UFName, ULName, UAge, U Date of Birth, country, Usemail, U contact-no

Now, we can create simplified tables:

cricket Board (BoardID [PK], Name, Address, contact_no)

Cricket team (TeamID [PK], Tname, Coach, captain)

cricket player (Player ID [PK], TeamID [FK], PFName, PLName, Age, P Date of Birth, playing Role, email, contact - no, Batting, Bowling)

cricket ground (Ground ID [PK], GName location, capacity)

Cricket Umpire (Umpire ID [PK], UFName, ULName, U Age, U Date of Birth, country, UseMail,

In these tables ([PK] denotes the primary key, [FK] denotes the foreign key and suitable constraints should be added to maintain data Integrity.

CReate tables for all non-prime attributes using at (Alpha plus) allows to group attributes based on their functional dependencies and candidates key the candidate keys in this case are BoardID TEAMID, MATCH, ID and UmpireID

cricket Board table: BoardID (PK), Name, Address, contact-No

Team table: TeamID (PK), TName, Coach, captain

player table: Player ID (PK), TeamID (FK), PFName, PL Name, Age, P Date of Birth, playing Role

Email, contact - no, Batting, Bowling

Match table: MatchID (PK), TeamID (PK), Match - Date time (), Result

Ground table: Ground ID (PK), GName, Location, capacity

umpire table: Umpire ID (PK), UFName, UL Name, U Age, U Date of Birth, country usmail U contact - no

create additional tables to represent transitive dependencies.

Already addressed transitive dependencies in previous normalization steps by Ground ID through the Result attribute.

MatchVenueTable : MatchID (PK, FK), GroundID (FK)

First Normal form:

the given relation into the first Normal form (1NF) to need to ensure that each attribute already contains atomic values, & there are repeating groups to eliminate.

second Normal form:

To determine whether the given relation is in the second Normal form (2NF), we need to check two conditions:

The relation must already be in 1NF (First Normal form).

all non-prime attributes (attributes not part of any candidate key) must be fully functional dependencies.

It appears that the potential candidates keys could be:

1. BoardID
2. TeamID
3. PlayerID
4. MatchID
5. UMPIREID

Next, we need to check if all non-prime attributes are fully functionally dependent on their respective candidate key(s).

third Normal form:

To determine whether the given relation is in the third Normal form (3NF), need to

1. check two conditions:
   the relation satisfies the conditions of the second Normal form (2NF). Now, it is

the given relations satisfies the conditions of the second Normal form (2NF). Now, let's check for transitive dependencies:

Now, let's analyze each functional dependency and check for transitive dependencies:

BoardID → Name, Address, contact-No

there are no transitive dependencies in this case, as Name, Adress, and contact-No are directly dependent on BoardID

TeamID → TName, coach, Captain

there are no transitive dependencies here either, as TName, coach, and captain are directly dependent on TeamID.

PlayerID → PNTName, PLName, Age, DateofBirth, playingRole, Email, contactno, Batting, Bowling

there are no transitive denpendencies for playerID, as all the mention ed attributes are directly dependent on playerID.

MatchID → MatchDate, time, Result, GroundID

there is tranditive dependency between MatchID and GroundID through the Result attribute. to resolve this, we create a new table called MatchVenue:

MatchVenue (MatchID [PK], GroundID [FK])

GroundID → GName, location, capacity

there are no transitive dependencies for GroundID, as GName, location and capacity are directly dependent on GroundID.

UmpireID → UfName, ULName, UAge, UDate of Birth

there are no transitive dependencies for Umpire ID, as UF Name, UL Name, UAge, UDate of Birth, country, Use mail, and Ucontact - no are directly dependent on (UmpIRE ID)

with the introduction of the match venue table to resolve the transitive dependency, the relation now satisfies the conditions of the third Normal form (3NF)

VEL TECH - CSE	
EX NO.	
PERFORMANCE (5)	
RESULT AND ANALYSIS (3)	
VIVA VOCE (3)	
RECORD (4)	
TOTAL (15)	
SIGN WITH DATE	

Results thus Normalization of the given relation is created the simplified tables with suitable constraint successfully.