Report - Functional Dependencies and 3rd Normal Form

Adam Spindler, Aravind Vicinthangal Prathivaathi, Pranshu Bheda, Ted Xie

March 22, 2019

1 Functional Dependencies

The functional dependencies are is the canonical cover form. The functional dependencies in the schema are as follows:

1.1 Killer_Info

match_id, victim_name,killer_name → $killer\_placement$
$match\_id,\ victim\_name, killer\_name\ \rightarrow killer\_position\_x$
$match\_id,\ victim\_name, killer\_name\ \rightarrow killer\_position\_y$

1.2 Death

match_id,victim_name,death_time→ $killer\_name$
$match\_id, killer\_name, victim\_name \rightarrow weapon\_id$
$victim\_name, match\_id \rightarrow death\_time$

1.3 Victim_Info

victim_name,match_id→ $victim\_placement$
$victim\_name, match\_id \rightarrow victim\_position\_x$
$victim\_name, match\_id \rightarrow victim\_position\_y$

1.4 Weapon

weapon_id → $weapon\_name$

1.5 Team

team_id,match_id→ $team\_placement$
$team\_id, match\_id \rightarrow party\_size$

1.6 Player

name,match_id→ $kills$
$name, match\_id \rightarrow knockdowns$
$name, match\_id \rightarrow assists$

$name, match\_id \rightarrow hitpoints\_tot$
$name, match\_id \rightarrow dist\_driven$
$name, match\_id \rightarrow dist\_walked$
$name, match\_id \rightarrow team\_id(FK)$

1.7 Match

match_id$\rightarrow date$
$match\_id \rightarrow game\_size$
$match\_id \rightarrow match\_mode$
$match\_id \rightarrow map$

2 Conditions to be in Third Normal Form

- All the non-key attributes are fully and functionally determined by the primary keys.
- There are no trivial functional dependencies
- There are no transitive dependencies in any of the tables within the schema.
- All the attributes in a table are only dependent on the primary keys of the table.

3 Conversion of Schema to 3NF

1. Let the initial relation be

   R : {name, match_id, team_id, weapon_id, victim_name, killer_name, killer_placement, victim_placement, killer_position_x, killer_position_y, victim_position_x, victim_position_y, weapon_name, time, date, map, game_size, match_mode, team_placement, party_size, kills, knockdowns, assists, hitpoints_tot, dist_driven, dist_walked, team_id, survive_time}

   Primary keys, PK: {name, match_id, team_id, victim_name, killer_name, weapon_id}

   The functional dependencies given in the previous page are already in canonical form and cannot be reduced further.

   Let these dependencies be called Fc.

2. Now, for each FD, X Y in Fc, create a relation Ri = XY such that every relation contains only the set of attributes where for every non-key attribute only its primary keys are present in the relation and the respective non-key attributes are present in each relation i.e. all the non-key attributes in the relation are only dependent on their primary keys.

   Use the Armstrongs union rule, to combine the RHS which map to the same LHS for every Fc. We then convert the resultant dependency in to a relation.

   R1:
   name,match_id $\rightarrow match\_id, kills,\ knockdowns, assists,$

$hitpoints\_tot, dist\_driven, dist\_walked, team\_id, survive\_time$

$Therefore,$
Player:$\{name, match\_id, team\_id, kills, knockdowns, assists,$
$hitpoints\_tot, dist\_driven, dist\_walked, team\_id, survive\_time\}$

PK:$\{name, match\_id\}$

R2:
In the death table the non-key attributes map to different primary keys. Since for a relation to be in 3NF, all the non-key attributes must map to the same primary key in a given relation, we need to normalize the death table. This can be done by adding a separate attribute called death time in the Victim_Info relation and removing death_time from the death relation. Thus, the normalized table looks like:
Death: {match_id, weapon_id, victim_name, killer_name }

PK: {victim_name, match_id, killer_name}

The canonical cover for this new death relation is
victim_name,match_id,killer_name → $weapon\_id$

Then the victim info will change in the following way:
R3:
victim_name,match_id→ $victim\_placement, victim\_position\_x,\ victim\_position\_y, death\_time$

victim_Info:$\{match\_id, victim\_name, victim\_placement, victim\_position\_x, victim\_position\_y, death\_time$

PK:$\{match\_id, victim\_name\}$

R4:
match_id, victim_name,killer_name → $killer\_placement,\ victim\_position\_x,\ victim\_position\_y$

killer_Info: { match_id, victim_name, killer_name, killer_placement, killer_position_x, killer_position_y
}

PK: {match_id, victim_name, killer_name }

R5: team_id,match_id → $team\_placement, party\_size$

Team:$\{match\_id, team\_id, team\_placement, party\_size\}$

PK:$\{match\_id, team\_id\}$

R6:$match\_id \rightarrow date, game\_size,\ match\_mode,\ map$

Match:$\{match\_id, date, game\_size, match\_mode, map\}$

PK:$\{match\_id, team\_id\}$

R7:$weapon\_id \ \rightarrow weapon\_name$

Weapon:$\{weapon\_id, name\}$

PK:$\{weapon\_id\}$

Thus, the final normalized relations are:

Player: {name (PK), match_id (PK), team_id (FK), kills, knockdowns, assists, hitpoints_tot, dist_driven, dist_walked, team_id, survive_time}

Death: {match_id(PK), weapon_id(FK), victim_name(PK), killer_name(PK/FK) }

Victim_Info: { match_id(PK), victim_name(PK), victim_placement, victim_position_x, victim_position_y, death_time}

killer_Info: { match_id (PK), victim_name (PK), killer_name (PK), killer_placement, killer_position_x, killer_position_y }

Team: {match_id(PK), team_id(PK), team_placement, party_size}

Match: {match_id (PK), date, game_size, match_mode, map}

Weapon: {weapon_id(PK), name}