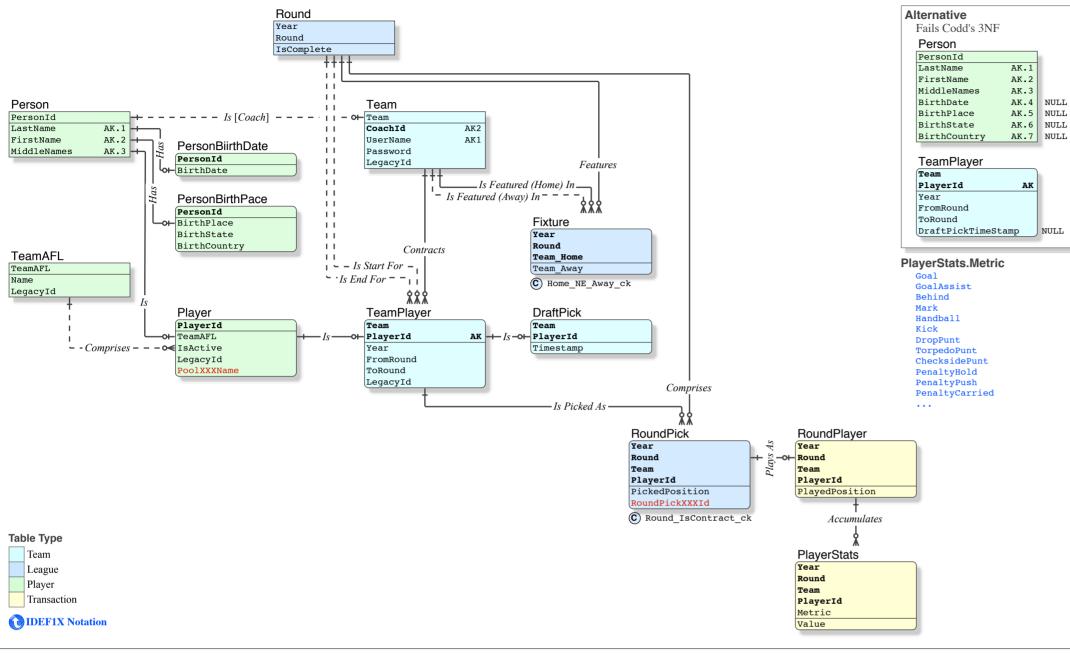
Fantasy Football • Relational, Normalised



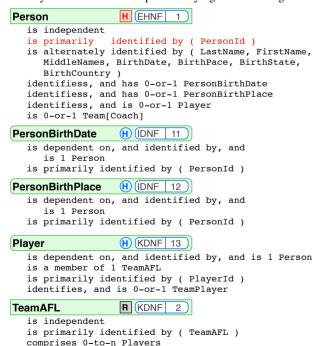


Predicate

• All FOPC Predicates can be read directly from an IDEF1X data model. They are povided here (except the descriptors, which are obvious in the model) in text form to assist those who are new to the Relational Model.

• They form a feedback loop for verifying the modelling exercise. As such, they should be checked carefully for veracity (ie. the declaration is true in the real world). Prefix each declaration with Each.

H (KDNF 3)



is independ	lent
is 1 Person	1 [Coach]
is primaril	ly identified by (Team)
is alternat	tely identified by (CoachId)
	cces (Coach is coach of 1 Team)
is alternat	tely identified by (UserName)
enfor	cces (UserName is admin of 1 Team)
	, and contracts 0-to-n TeamPlayers
	, and is featured (home) in 0-to-n Fixtures
is featured	d (away) in 0-to-n Fixtures
TeamPlayer	(H) (KDNF 31)
is dependen	nt on, and identified by, and is contracted by 1 Team
	nt on, and identified by, and is 1 Player
is primaril	ly identified by (Team, PlayerId)
is alternat	tely identified by (PlayerId)
prese	erves (Player is 0-or-1 TeamPlayer)
Furth	ner, Relationally, allows Player Key in the Round*
conte	ext to be PlayerId, even though it is (Team, PlayerId)
identifies,	, and is 0-or-1 DraftPick
identifies,	, and is picked as 0-to-n RoundPicks
DraftPick	(H) (IDNF 32)
is dependen	nt, and identified by, and
is 1 Tea	amPlayer
is primaril	ly identified by (Team, PlayerId)

identifies, and identifies, and identifies, and is start for 0-to-is end for 0-to-reliance. Fixture is dependent on, is dependent on, features 1 Team is Hom—Team is Hom—Team Away is RoundPick is dependent on, a constituent is dependent on, a pick of 1 Team is primarily identifies constrained to (Year & Round (Year & Round identifies, and processing the RoundPlayer	n TeamPlayers (2) (KDNF 41) and identified by, and features 1 Team (home) and identified by, and a feature of 1 Round (away) ntified by (Year, Round, Team_Home) me once per Year, Round is not so constrained (2) (KDNF 42) and identified by, and of 1 Round
is dependent on, is dependent on, features 1 Team is primarily ider Team is Hon Team Away i RoundPick is dependent on, a constituent is dependent on, a pick of 1 Te is primarily ider is constrained to (Year & Round (Year & Round identifies, and p	and identified by, and features 1 Team (home) and identified by, and a feature of 1 Round (away) tified by (Year, Round, Team_Home) me once per Year, Round is not so constrained ② KDNF 42 and identified by, and of 1 Round
is dependent on, features 1 Team (is primarily ider Team is Hon Team Away is RoundPick is dependent on, a constituent is dependent on, a pick of 1 Team a pick of 1 Team are is constrained to (Year & Round (Year & Round identifies, and proundPlayer	and identified by, and a feature of 1 Round (away) ntified by (Year, Round, Team_Home) me once per Year, Round is not so constrained 2 KDNF 42 and identified by, and of 1 Round
is dependent on, a constituent is dependent on, a pick of 1 Te is primarily ider is constrained to (Year & Round (Year & Round identifies, and p	and identified by, and of 1 Round
a constituent is dependent on, a pick of 1 Te is primarily ider is constrained to (Year & Round (Year & Round identifies, and p	of 1 Round
	ntified by (Year, Round, Team, PlayerId)
is dependent on,	2 KDNF 43
	and identified by, and a play of 1 RoundPick ntified by (Year, Round, Team, PlayerId) accumulates 0-or-n PlayerStats
PlayerStats	2 KDNF 44
is dependent on, an accumulation is primarily iden	and identified by, and

Progression

- The original model is a Record Filing System, which is characterised by Record IDs (physical poinetrs) as "keys"; all files are Independent; all relations are Non-Identifying,. That results in horrendous navigation & query code. The two main tasks executed here are:
- 1 complying with the *Relational Model*, logical Relational Keys (composite),
- 2 and Normalising the data
- That obtained Relational powers that are not possible in RFS:

Relational Integrity	Eg. Stats for Round::Pick::Play constrained to RoundPick (not any Player)	
	Minimal JOINs (eg. see TeamStats); any report satisfied by a single SELECT; easy navigation; straight-forward query code	
Relational Speed	Minimal indices; fastest DML; smallest packets (cache memory & network)	

- Assumption: that PlayerStats are relevant (collected) for RoundPlayers, not TeamPlayers, not RoundPicks.
- PlayerStats is given in Isolated Descriptor NF (tighter than "6NF"): this allows pivoting on any two Dimensions, etc.
- Statistics for {Year, Round, Team [Coach], Player [TeamAFL] } are now simple.
- When a Player is contracted to a Team, he is a TeamPlayer. Contract equals TeamPlayer
- Assumption: that a contract (FromRound; ToRound) is for one year. Otherwise TeamPlayer needs FromYear; ToYear
- Constraining RoundPicks to a contracted Round requires a CONSTRAINT that calls a Function (Standard SQL)
- Player & Coach details Normalised into Person: the international convention for Person Identity has been implemented
- Surrogates are prohibited in the *RM*. They break the Access Path Independence rule. PersonId is the only surrogate retained, it is justified because the otherwise PK is too long for migration as an FK. Nevertheless it remains a breach: it cuts off the reference-ability from the descendant rows to all tables above the breach (eg. BirthCountry.
- NULL is an indicator that Normalisation is incomplete. Nulls eliminated via Normalisation of Nullable columns:
- PersonBirthDate, PersonBirthPlace, DraftPick (the Alternative, with NULL columns, is given)
- RoundPlayer (overcomes the difficulty determining a RoundPick who has played, but has no PlayerStats)
- PlayerStats

C RoundPick: Round IsContract ck

Constraint · Model

• Red columns are simply those that could not be read from the original PNG. Such are, of course, unknown, and thus excluded from the exercise.

```
TeamPlayer(Year, FromRound) =< RoundPick(Year, Round) AND</pre>
     TeamPlayer(Year, ToRound) >= RoundPick(Year, Round)
   C Fixture: Home NE Away ck
     Team Home != Team Awav
Function
  CREATE FUNCTION
       Round_IsContract_fn (
@Year DATE,
        @Round TINYINT.
        @Team CHAI
@PlayerId INT
       RETURNS TINYINT
  DECLARE @Return TINYINT
  IF EXISTS (
SELECT 1
FROM TeamPlayer
            WHERE Team
                                = @Team
                  PlayerId = @PlayerId
DATEPART( YY, Year ) >= DATEPART( YY, @Year )
FromRound >= @Round
            AND
       AND EXISTS (
       SELECT 1
            FROM TeamPlayer
WHERE Team = @Team
AND PlayerId = @PlayerId
                  DATEPART( YY, Year ) <= DATEPART( YY, @Year )
ToRound <= @Round
       SET @Return = 1
  ELSE
       SET @Return = 0
  RETURN @Return
Constraint · DDL
  ALTER TABLE TeamPlayer
ADD CONSTRAINT TeamPlayer_Round_IsContract_ck
            CHECK ( Round_IsContract_fn
                 Round,
                 Team,
                 PlayerId
Team Stats
  SELECT Year,
            Team.Team, -- ShortName
Coach = LastName + "," + FirstName,
            SUM( Value )
       FROM Team

JOIN PlayerStats ON Team.Team = PlayerStats.Team
                                 ON Team.CoachId = Person.PersonId,
             JOIN Person
       TeamAFL Stats
  SELECT TeamAFL. TeamAFL, -- ShortName
            Player = LastName + "," + FirstName
            SUM( Value )
            JOIN Player
                                ON TeamAFL.TeamAFL = Player.TeamAFL
       JOIN PlayerStats ON Player.PlayerId = PlayerStats.PlayerId
JOIN Person ON Player.PlayerId = Person.PersonId
WHERE Year = "2017" AND Metric = "Goal"
       GROUP BY Team, Player
```