SOFTWARE GEMS

Diagram (IDEF1X Data Model)

Round
- Year
- Round
- IsComplete

Person
- PersonId
- LastName — AK.1
- FirstName — AK.2
- MiddleNames — AK.3

Is [Coach]

Team
- Team
- CoachId — AK2
- UserName — AK1
- Password
- LegacyId

Features

PersonBiirthDate
- PersonId
- BirthDate

Has

PersonBirthPace
- PersonId
- BirthPlace
- BirthState
- BirthCountry

Contracts

Is Featured (Home) In
Is Featured (Away) In

Fixture
- Year
- Round
- Team_Home
- Team_Away

C Home_NE_Away_ck

Is Start For
Is End For

TeamAFL
- TeamAFL
- Name
- LegacyId

Is

Player
- PlayerId
- TeamAFL
- IsActive
- LegacyId
- PoolXXXName

Is — **TeamPlayer**
- Team
- PlayerId — AK
- Year
- FromRound
- ToRound
- LegacyId

Is — **DraftPick**
- Team
- PlayerId

Is — **Draft**
- Team
- PlayerId
- Timestamp

Comprises

Is Picked As

Comprises

PlayerStats.Metric
- Goal
- GoalAssist
- Behind
- Mark
- Handball
- Kick
- DropPunt
- TorpedoPunt
- ChecksidePunt
- PenaltyHold
- PenaltyPush
- PenaltyCarried
- ...

IDEF1X Notation

Table Type
- Team
- Round
- Person etc
- Transaction

Alternative
Fails Codd's 3NF

Person
- PersonId
- LastName — AK.1
- FirstName — AK.2
- MiddleNames — AK.3
- BirthDate — AK.4 — NULL
- BirthPlace — AK.5 — NULL
- BirthState — AK.6 — NULL
- BirthCountry — AK.7 — NULL

RoundPick
- Year
- Round
- Team
- PlayerId
- PickedPosition
- RoundPlayerXXXId

C Round_IsContract_ck

Plays As

RoundPlayer
- Year
- Round
- Team
- PlayerId
- PlayedPosition

Accumulates

PlayerStats
- Year
- Round
- Team
- PlayerId
- Metric
- Value

TeamPlayer
- Team
- PlayerId — AK
- Year
- FromRound
- ToRound
- IsDraftPick
- DraftedTimeStamp — NULL

Constraint
C RoundPick: Round_IsContract_ck
   TeamPlayer(Year, FromRound) >= RoundPick(Year, Round) AND
   TeamPlayer(Year, ToRound) =< RoundPick(Year, Round)

C Fixture: Home_NE_Away_ck
   Team_Home != Team_Away

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`.

Person — H — EHNF — 1
is independent
is primarily identified by (PersonId)
is alternately identified by (LastName, FirstName,
  MiddleNames, BirthDate, BirthPace, BirthState,
  BirthCountry)
identifiess, and has 0-or-1 PersonBirthDate
identifiess, and has 0-or-1 PersonBirthPlace
identifiess, and is 0-or-1 Player
is 0-or-1 Team[Coach]

PersonBirthDate — H — IDNF — 11
is dependent on, and identified by, and
  is 1 Person
is primarily identified by (PersonId)

PersonBirthPlace — H — IDNF — 12
is dependent on, and identified by, and
  is 1 Person
is primarily identified by (PersonId)

Player — H — KDNF — 13
is dependent on, and identified by, and is 1 Person
is a member of 1 TeamAFL
is primarily identified by (PlayerId)
identifies, and is 0-or-1 TeamPlayer

TeamAFL — R — KDNF — 2
is independent
is primarily identified by (TeamAFL)
comprises 0-to-n Players

Team — H — KDNF — 3
is independent
is 1 Person [Coach]
is primarily identified by (Team)
is alternately identified by (CoachId)
  -- enforces (Coach is coach of 1 Team)
is alternately identified by (UserName)
  -- enforces (UserName is admin of 1 Team)
identifies, and contracts 0-to-n TeamPlayers
identifies, and is featured (home) in 0-to-n Fixtures
is featured (away) in 0-to-n Fixtures

TeamPlayer — H — KDNF — 31
is dependent on, and identified by, and
  is contracted by 1 Team
is dependent on, and identified by, and is 1 Player
is primarily identified by (Team, PlayerId)
is alternately identified by (PlayerId)
  -- 1. preserves (Player is 0-or-1 TeamPlayer)
  -- 2. allows Player Key in the Round* context
  --   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 — KDNF — 32
is dependent, and identified by, and
  is 1 TeamPlayer
is primarily identified by (Team, PlayerId)

DraftP — H — KDNF — 33
is dependent, and identified by, and
  is 1 DraftPick
is primarily identified by (Team, PlayerId)

Round — H — KDNF — 4
is independent
is primarily identified by (Year, Round)
identifies, and features 0-to-n Fixtures
identifies, and 0-to-n RoundPicks
is start for 0-to-n TeamPlayers
is end for 0-to-n TeamPlayers

Fixture — 2 — KDNF — 41
is dependent on, and identified by, and features 1 Team (home)
is dependent on, and identified by, and a feature of 1 Round
features 1 Team (away)
is primarily identified by (Year, Round, Team_Home)
  -- Team is Home once per Year, Round
  -- Team Away is not so constrained

RoundPick — 2 — KDNF — 42
is dependent on, and identified by, and
  a constituent of 1 Round
is dependent on, and identified by, and
  a pick of 1 TeamPlayer
is primarily identified by (Year, Round, Team, PlayerId)
is constrained to
  (Year & Round) >= TeamPlayer.(Year & FromRound) AND
  (Year & Round) =< TeamPlayer.(Year & ToRound)
identifies, and plays as 0-or-1 RoundPlayer

RoundPlayer — 2 — KDNF — 43
is dependent on, and identified by, and a play of 1 RoundPick
is primarily identified by (Year, Round, Team, PlayerId)
identifies, and accumulates 0-or-n PlayerStats

PlayerStats — 2 — IDNF — 44
is dependent on, and identified by, and
  an accumulation of 1 RoundPlayer
is primarily identified by (Year, Round, Team, PlayerId, Metric)

Database design advice (specific example- Fantasy Football)

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 constrained to a RoundPlayer (not any Player)
Relational Power	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 the logical, natural Key for Round is (Year, Round), not (Round). Otherwise it is limited to 1 year.
- **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*. It breaks 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 (eg. RoundPlayer.PlayerId) to all tables (ancestors) above the breach (eg. PersonBirthPlace.BirthCountry).
- `NULL` is an indicator that Normalisation is incomplete. Nulls eliminated via Normalisation of Nullable columns:
  • PersonBirthDate; PersonBirthPlace; DraftPick; Draft (the Alternative, with `NULL` columns, is given)
  • RoundPlayer (eliminates the difficulty determining a RoundPick who has played, but has no PlayerStats)
  • PlayerStats. Additionally, now that it is IDNF, none of the Stats columns are Null.
- 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.

SQL • Declaration
Constraint
```
ALTER TABLE TeamPlayer
    ADD CONSTRAINT TeamPlayer_Round_IsContract_ck
        CHECK ( Round_IsContract_fn (
                Year,
                Round,
                Team,
                PlayerId
                ) = 1
            )
```

Function
```
CREATE FUNCTION
    Round_IsContract_fn (
    @Year      DATE,
    @Round     TINYINT,
    @Team      CHAR(12),
    @PlayerId INT
    )
    RETURNS TINYINT
AS
DECLARE @Return TINYINT
IF EXISTS (
    SELECT  1
    FROM    TeamPlayer
    WHERE   Team      = @Team
    AND     PlayerId  = @PlayerId
    AND     DATEPART( YY, Year ) >= DATEPART( YY, @Year )
    AND     FromRound >= @Round
    )
    AND EXISTS (
    SELECT  1
    FROM    TeamPlayer
    WHERE   Team      = @Team
    AND     PlayerId  = @PlayerId
    AND     DATEPART( YY, Year ) <= DATEPART( YY, @Year )
    AND     ToRound   <= @Round
    )
    SET @Return = 1
ELSE
    SET @Return = 0
RETURN @Return
```

SQL • Report
Team Stats
```
SELECT  Year,
        Team.Team, -- ShortName
        Coach = LastName + "," + FirstName,
        SUM( Value )
FROM    Team
        JOIN PlayerStats ON Team.Team = PlayerStats.Team
        JOIN Person      ON Team.CoachId = Person.PersonId,
        ( SELECT DISTINCT  Year
            FROM Round ) Year
WHERE   Metric = "Goal" AND SUM( Value ) NOT NULL
GROUP BY Year, Team, Coach
```

TeamAFL Stats
```
SELECT  TeamAFL.TeamAFL, -- ShortName
        Player = LastName + "," + FirstName,
        SUM( Value )
FROM    TeamAFL
        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
```