Relational Database Design

Module 5: Converting to Relational

Hugo Kornelis
hugo@perFact.info

pluralsight
hardcore developer training

Outline

- **Conversion between representations**
  - Entity Relationship diagram
  - Relational database design
- **Why do we need two representations?**
- **How to represent a relational database design?**
- **Conversions**
  - ER diagram to relational
  - Relational to ER diagram

Why convert from ER to relational?

- **Entity Relationship diagram**
  - Good for design
    - Quick overview
    - Easy to read
    - Accessible
  - Not good for implementation
    - Mapping not exactly one on one

- **Relational database design**
  - Good for implementation
  - Not good for design

Why convert from relational to ER?

- **Inherited database**
  - No documentation
  - Outdated documentation
- **Convert to Entity Relationship diagram**
  - Quick overview
  - Understand how tables relate

When to convert?

- **Ideal?**
  - Initial model (ER)
  - Normalize (ER)
  - Convert (Relational)

- **Practical?**
  - Initial model (ER)
  - Convert (Relational)
  - Normalize (Relational)
  - Convert changes back (ER)

What to convert?

- **Relational representation**
  - Physical model
    - Optimized for performance
    - Vendor-specific tweaks
  - Logical model

- **Entity Relationship representation**
  - Logical model
  - Physical model?
    - (Only when converting FROM relational implementation – use with care!)

Representing a relational database design

- **Compact form**
  - Benefits
    - Compact
    - Quick to create
  - Downside
    - Lack of detail

Optional? Reference?

Table1 (Column1, Column2, Column3)

Table2 (Column4, Column5)

Data type?

Table3 (Primary key, Alternate key, Foreign key)

or

Table3 (Primary key, Alternate key, Foreign key)

Representing a relational database design

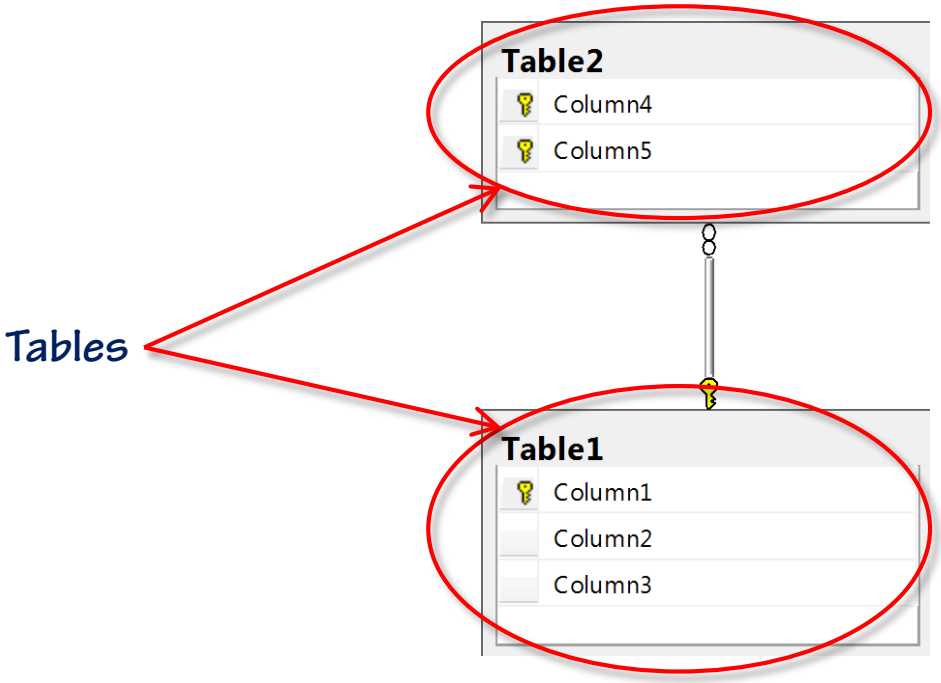
- **DDL**
  - All details included
  - Not accessible

```sql
CREATE TABLE Table1
    (Column1 varchar(20) NOT NULL,
     Column2 date NULL,
     Column3 int NOT NULL,
     PRIMARY KEY (Column1),
     UNIQUE (Column3),
     CHECK (Column3 > 0)
    );

CREATE TABLE Table2
    (Column4 varchar(20) NOT NULL,
     Column5 varchar(30) NOT NULL,
     PRIMARY KEY (Column4, Column5),
     FOREIGN KEY (Column4)
            REFERENCES Table1(Column1)
    );
```

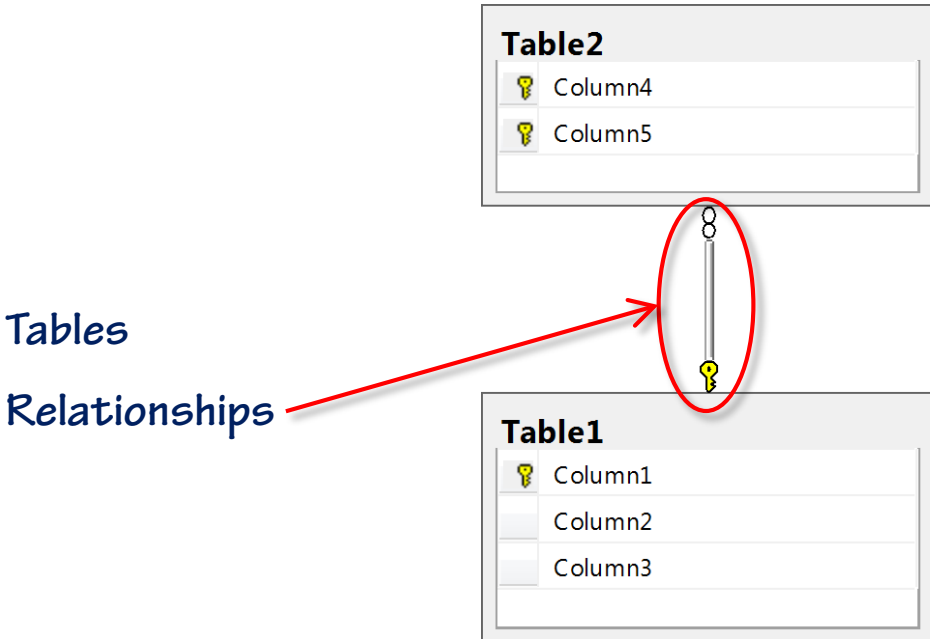
Representing a relational database design

- Graphical



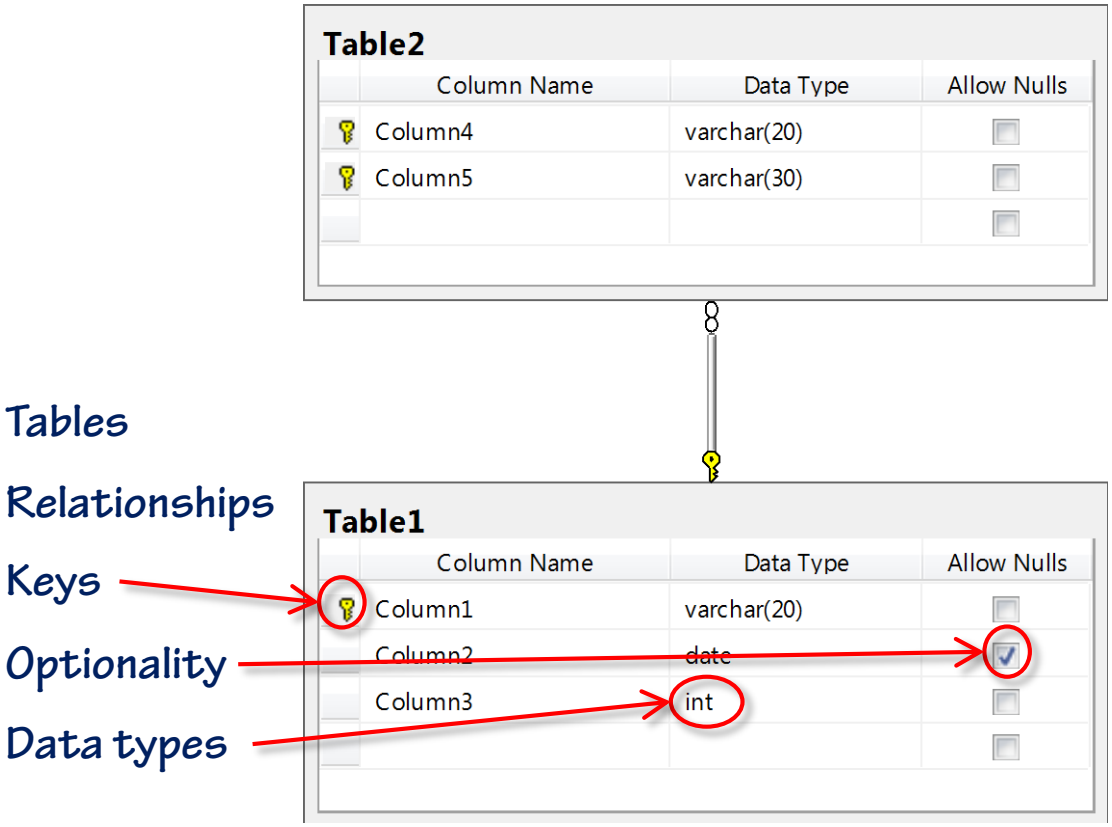
Representing a relational database design

- Graphical



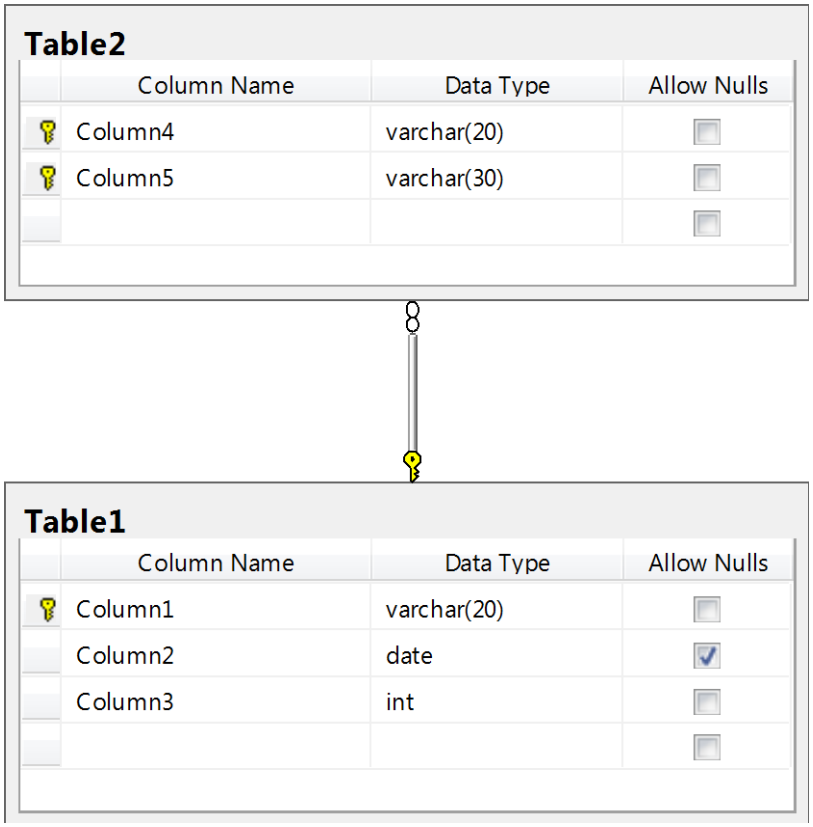
Representing a relational database design

- Graphical



Representing a relational database design

- **Graphical**
  - Notation similar to ER diagrams
    - Easy to understand
    - Blurs distinction between design and implementation
      - Leads to implementation choices in design



Tables

Relationships

Keys

Optionality

Data types

ER to relational: Entity types

- **Entity type**
  - Converted to a relational table
  - Same name or different name?

  - ER naming convention
    - Singular (e.g. Person)

  - Relational naming convention
    - Plural (e.g. Persons)
    - Or group (e.g. People)

ER to relational: Entity types

- **Entity type**
  - Converted to a relational table
  - Same name or different name?

  - Same name
    - Easier mapping between representations
    - Plural entity type names make the ER diagram look wrong
      - (And the relationship readings sound weird)
    - Singular table names introduce wrong mindset
      - Encourage one-row-at-a-time coding instead of set-based
      - Queries sound better with plural table names
      - Less chance of hitting a reserved word

ER to relational: Entity types

- **Entity type**
  - Converted to a relational table
  - One column for every attribute
    - Column name is (almost always) equal to attribute name
    - Optional = nullable; mandatory = NOT NULL
    - Add data type
      - Implementation independent (e.g. “character” / “numeric”)
      - Optionally shortened
        - C = character data
        - N = numeric data
        - D = date/time data
      - Full details in appendix
        - Range
        - Precision
        - Maximum length
        - …

ER to relational: Candidate keys

- **Candidate key**
  - Enforced by constraint
    - Primary key → **PRIMARY KEY** constraint
      - Required by Codd's 2nd rule
      - At most one
    - Alternate key → **UNIQUE** constraint

  - Primary key vs alternate key:
    - Primary key is default target for **FOREIGN KEY** constraint
    - Primary key must be on **NOT NULL** columns
    - Implementation-dependent differences
      - Final choice of primary key in physical model

ER to relational: Candidate keys

- **Representation of primary key**

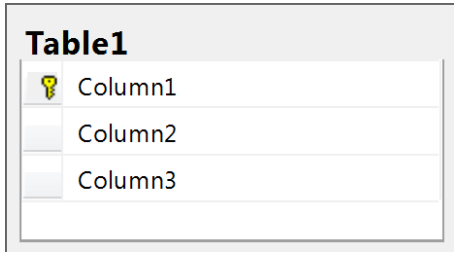
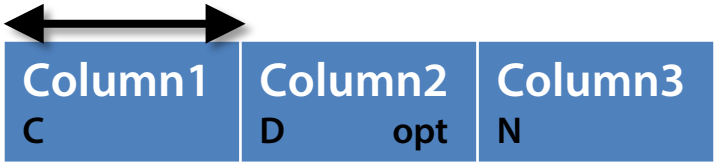


Table1	
PK	**Column1**
	Column2 **Column3**

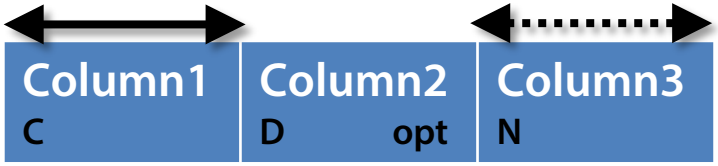
Table1 (Column1, Column2, Column3)



- **Representation of alternate key**
  - Appendix

Table1	
PK	**Column1**
U1	Column2 **Column3**

Table1 (Column1, Column2, Column3)



ER to relational: Candidate keys

- **Representation of primary key**

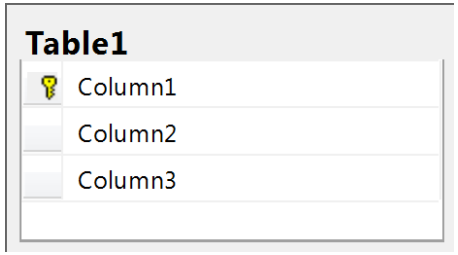
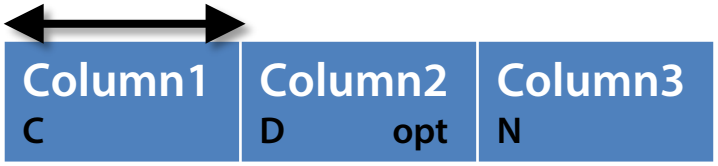


Table1	
PK	**Column1**
	Column2 **Column3**

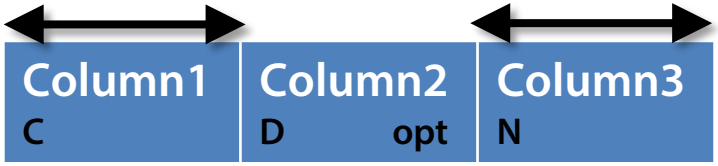
Table1 (Column1, Column2, Column3)



- **Representation of alternate key**
  - Appendix

Table1	
PK	**Column1**
U1	Column2 **Column3**

Table1 (Column1, Column2, Column3)




```
PRIMARY KEY (Column1),
UNIQUE (Column3),
```

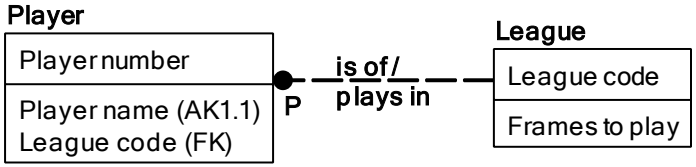
ER to relational: Candidate keys

- **Every key in ER diagram**
  - Candidate key in relational database design
  - Primary or alternate?
    - Stick to choice made in ER diagram
    - Choose now

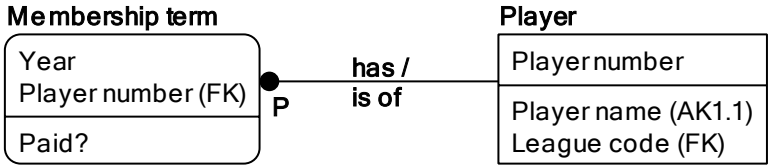
Final choice in physical model!!!

ER to relational: One-to-many relationships

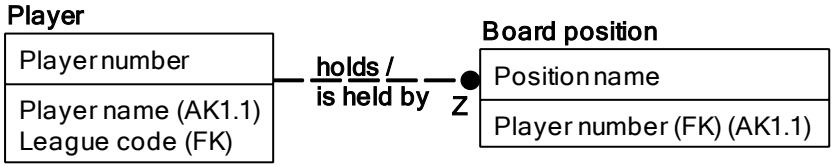
- **One-to-many relationship**



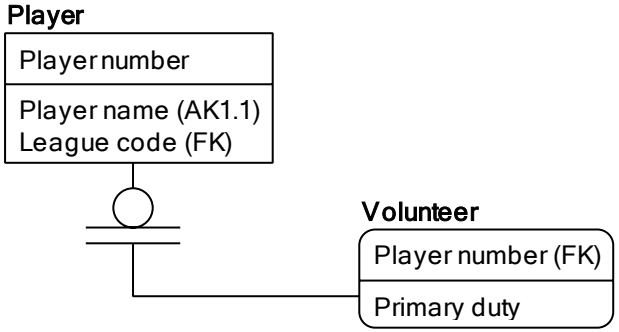
  - Special case: identifying relationship



- **One-to-one relationship**

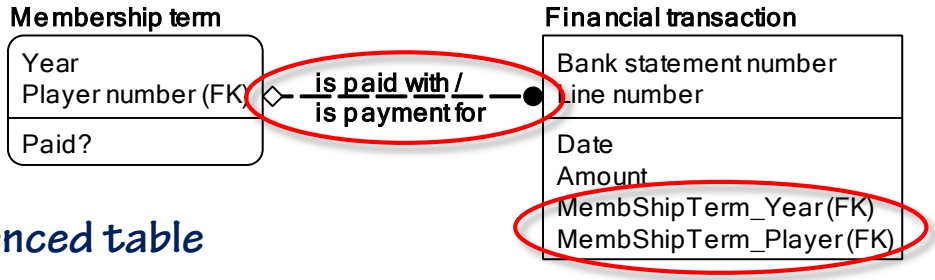


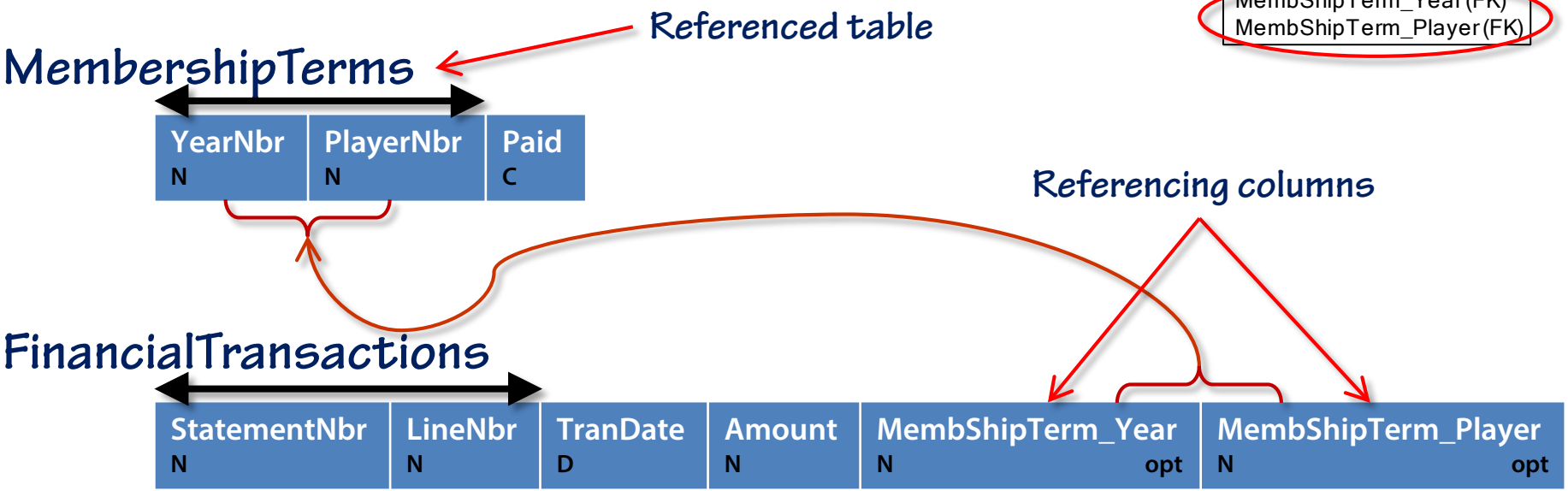
  - Special case: subtype relationship



ER to relational: One-to-many relationships

- **One-to-many relationship**
  - FOREIGN KEY constraint






```
FOREIGN KEY (MembShipTerm_Year, MembShipTerm_Player)
    REFERENCES MembershipTerms (YearNbr, PlayerNbr)
```

ER to relational: One-to-many relationships

- **One-to-many relationship**
  - FOREIGN KEY constraint
  - Foreign key attributes represented in ER diagram?
    - Just add the FOREIGN KEY constraint
  - Using ER method that leaves out the foreign key attributes?
    - Add referencing columns first, then add FOREIGN KEY constraint
    - Choose one of the candidate keys in the referenced table
      - Default to PRIMARY KEY

ER to relational: One-to-many relationships

- **One-to-one relationship**
  - FOREIGN KEY constraint
  - Choice of parent (referenced) and child (referencing) table:
    - Copy from ER diagram, if choice was made there
    - Choose now otherwise
    - **CHOICE CAN BE CHANGED LATER** (physical model)

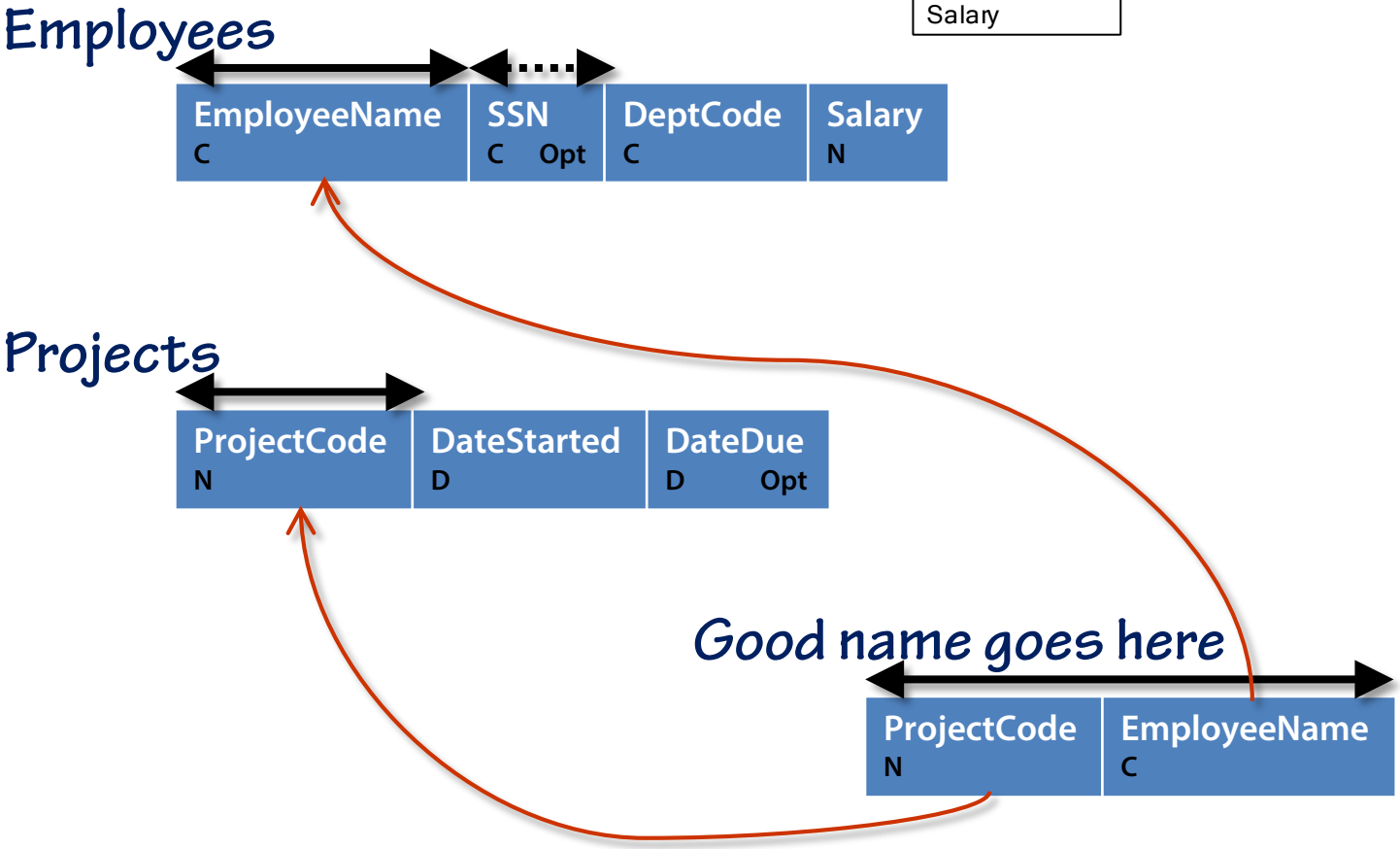
- **Subtype relationship**
  - FOREIGN KEY constraint
  - Always subtype = referencing table / supertype = referenced table
  - Appendix for additional information
    - Discriminator
    - Complete / incomplete subtype relationship
    - Mutual exclusive subtype relationships

ER to relational: One-to-many relationships

- **One-to-many / one-to-one relationship**
  - Minimum cardinality of child
    - If zero, allow NULLs in referencing columns
    - If one, define NOT NULL on referencing columns
  - Minimum cardinality of parent
    - If zero, nothing needed (standard behavior)
    - If one, cannot be represented
      - Specify in appendix
      - Has to be enforced in code
        - Only possible with *deferred constraint checking*
  - Special maximum cardinality of parent
    - Cannot be represented
      - Specify in appendix
      - Has to be enforced in code

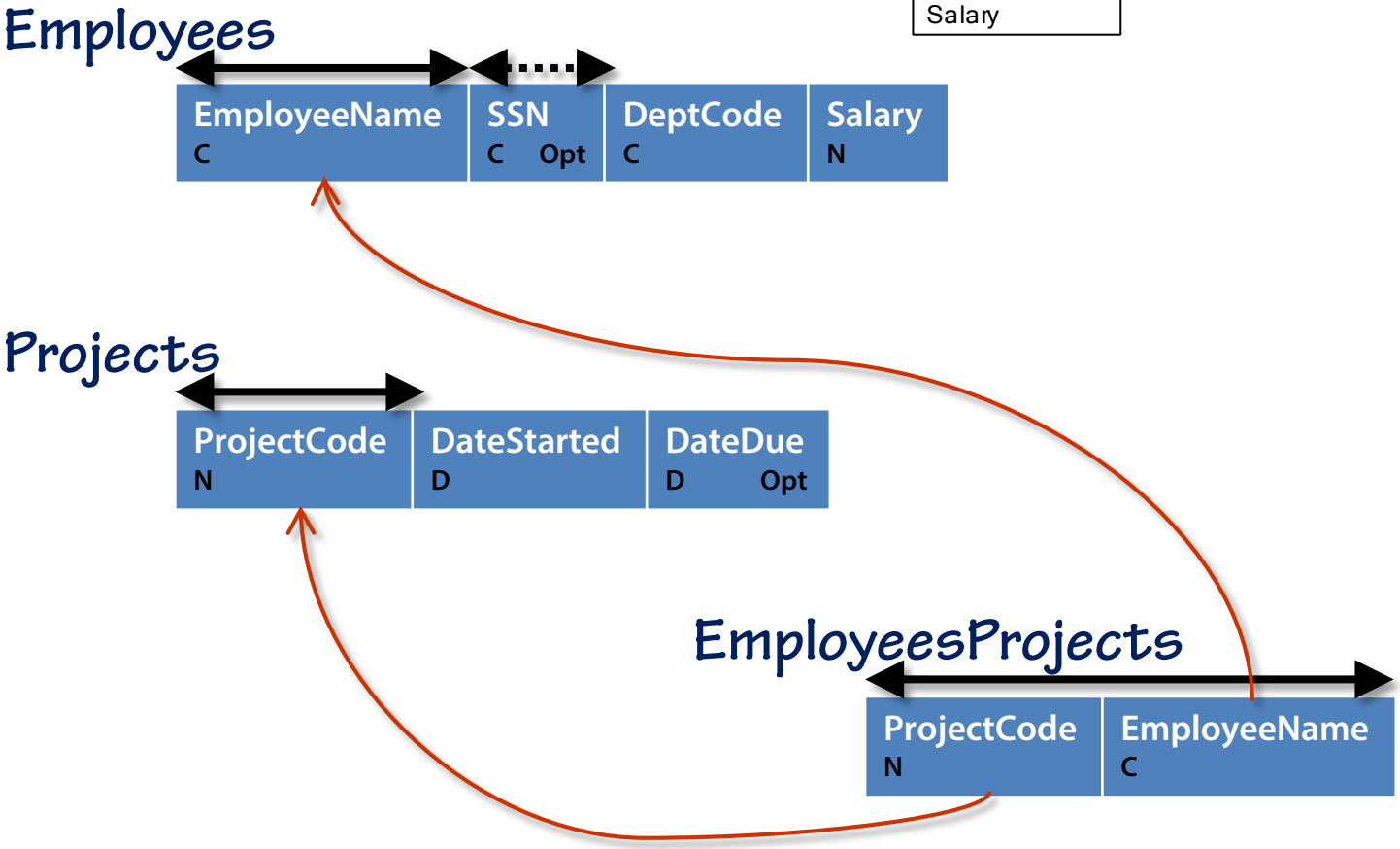
ER to relational: Many-to-many relationships

- **Many-to-many relationship**
  - Minimum/special cardinality
    - Appendix / code



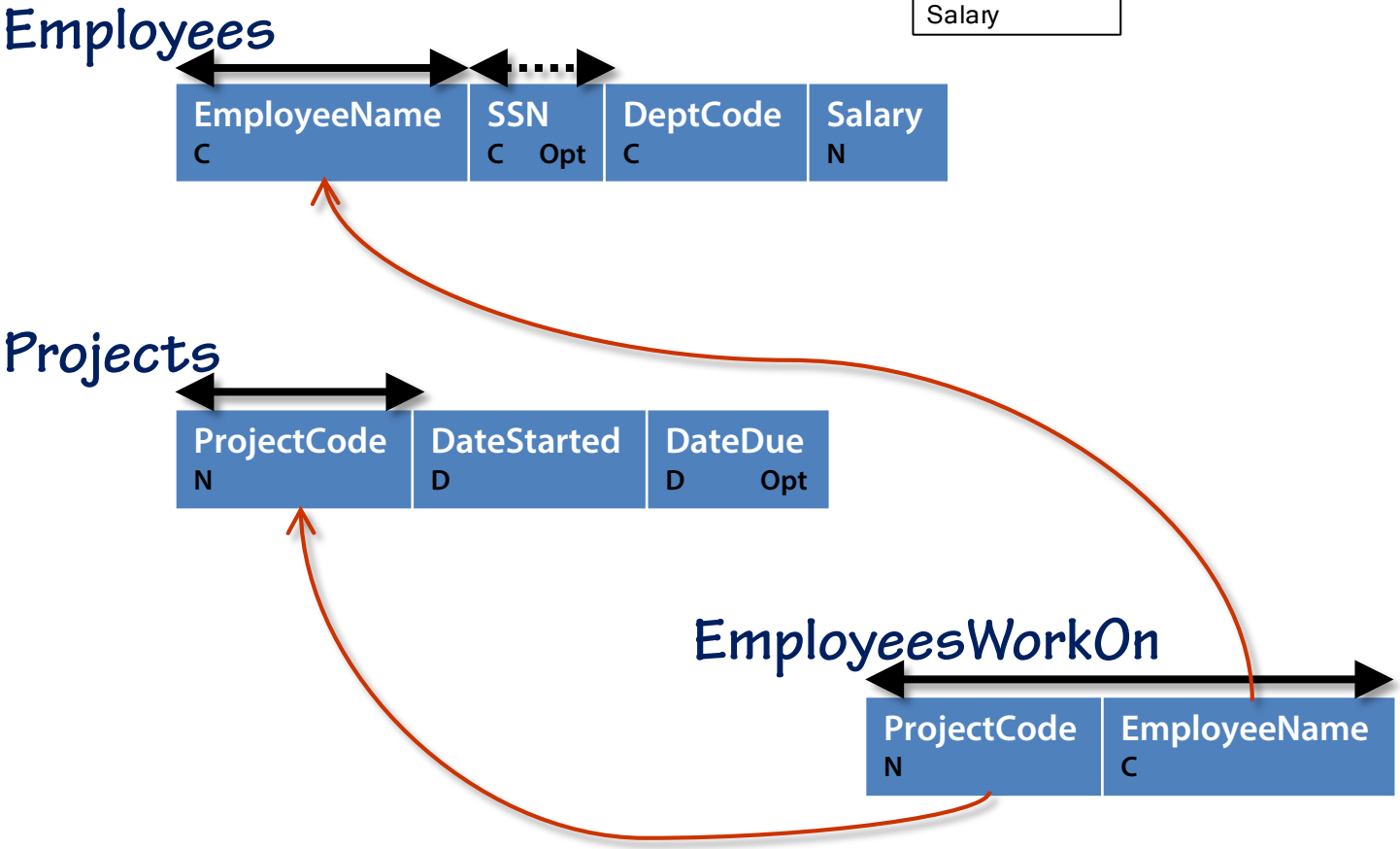
ER to relational: Many-to-many relationships

- **Many-to-many relationship**
  - Minimum/special cardinality
    - Appendix / code



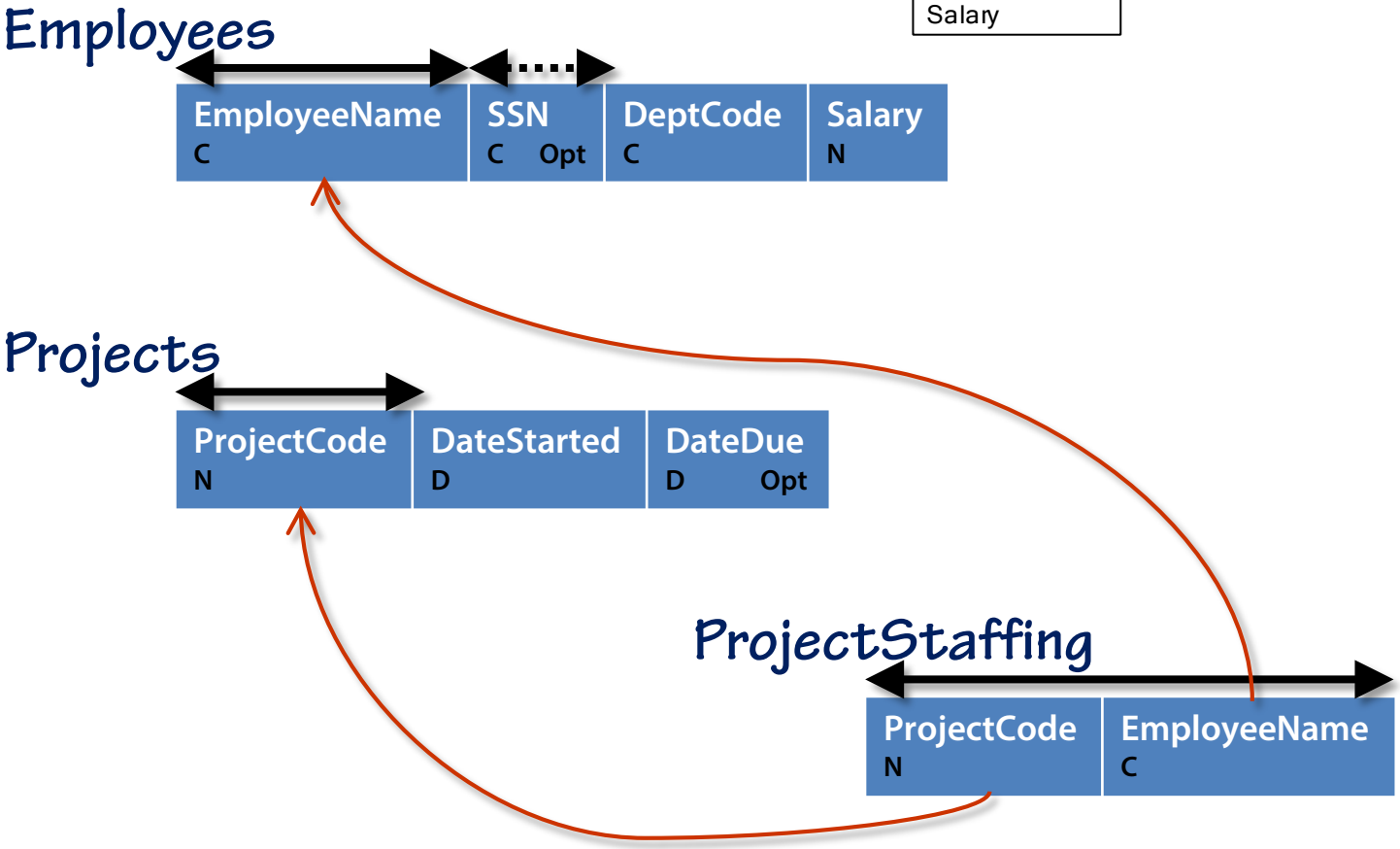
ER to relational: Many-to-many relationships

- **Many-to-many relationship**
  - Minimum/special cardinality
    - Appendix / code



ER to relational: Many-to-many relationships

- **Many-to-many relationship**
  - Minimum/special cardinality
    - Appendix / code



ER to relational: Many-to-many relationships

- **Many-to-many relationship**
  - Implemented as extra table
    - Junction table
    - Linking table
    - Cross-reference table
    - Join table
    - …
  - Not different from other tables!

ER to relational: Many-to-many relationships

- **Relationships between three or more entity types**
  - Not supported in IDEF1X
  - Possible in some other ER methods
  - Implemented as extra table (similar to many-to-many relationship)
    - Three or more foreign keys, for each connected entity type
    - Primary key over all columns

ER to relational: Other constraints

- **CHECK constraint**
  - Logical expression
    - May never evaluate to FALSE for any row
  - Can include one column, or several column
    - Cannot reference other tables
    - Cannot reference other rows in the same table

```
ALTER TABLE Matches
ADD CHECK (FramesWon + FramesLost <= 5);
```

```
ALTER TABLE Matches
ADD CHECK (FramesWon + FramesLost
            <= (SELECT MaxFrames
                FROM   MatchTypes
                WHERE  MatchTypes.MatchTypeID = Matches.MatchTypeID);
```

ER to relational: Other constraints

- **Generated column**
  - What if only a single value is allowed?
  - Database can compute it for you
    - Generated column (aka derived column / computed column)
    - No access to other tables or other rows in same table

```
ALTER TABLE Matches
ADD CHECK (FramesWon + FramesLost  = 5);
```

FramesLost always equal to (5 – FramesWon)

```
ALTER TABLE Matches
ADD FramesLost AS (5 - FramesWon);
```

ER to relational: Other constraints

- **Assertion**
  - Similar to CHECK constraint
  - Does allow access to other rows / other tables
  - Not supported by many RDBMS vendors
    - Can be used as a way to specify logic for database developers

```
CREATE ASSERTION MaxFrames
ADD CHECK (FramesWon + FramesLost
            <= (SELECT MaxFrames
                FROM   MatchTypes
                WHERE  MatchTypes.MatchTypeID = Matches.MatchTypeID);
```

ER to relational: Other constraints

- **DEFAULT constraint**
  - Not really a constraint
  - Provides standard value to use if no value specified
  - Not the same as a generated column!
    - DEFAULT used only for new rows
    - DEFAULT can be overridden or changed
    - DEFAULT cannot reference any other columns or contain logic
      - Constant values
      - Built-in functions

```sql
ALTER TABLE Members
ADD DEFAULT 'NL' FOR CountryCode;
```

```sql
ALTER TABLE Matches
ADD DEFAULT CURRENT_TIMESTAMP FOR DatePlayed;
```

ER to relational: Other constraints

- CHECK constraint
- Generated column
- Assertion
- DEFAULT constraint

- Found in appendix
- Not represented in ER diagram

ER to relational: demo

Calendar

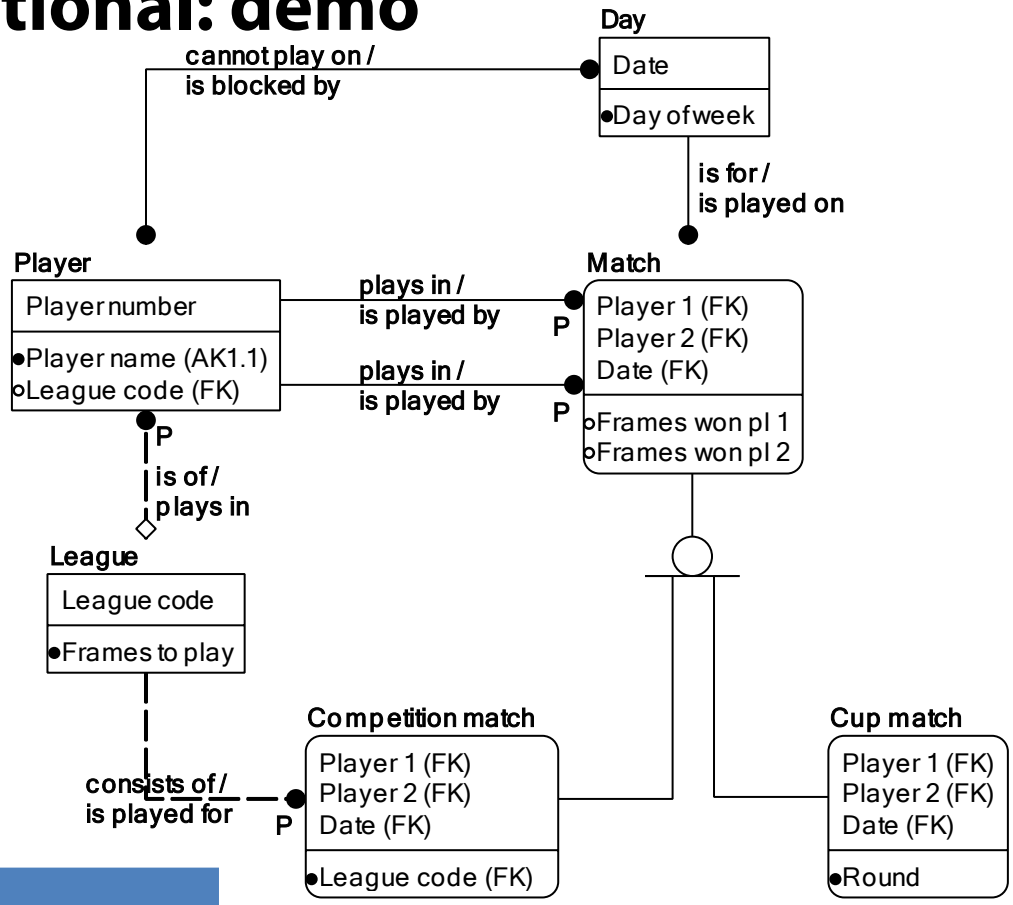
Players

Leagues

Matches

CompetitionMatches

CupMatches



ER to relational: demo

Calendar

Date	DayOfWeek
D	C

Players

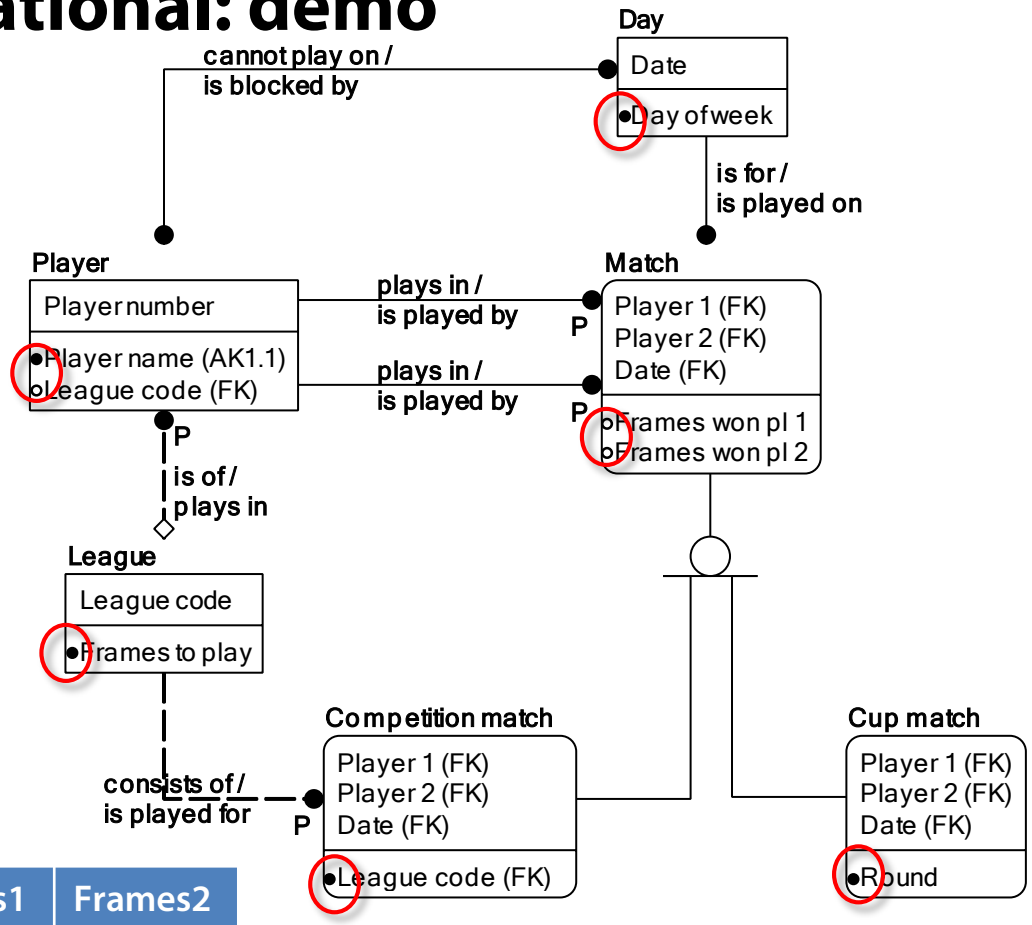
PlayerNbr	PlayerName	LeagueCode
N	C	C opt

Leagues

LeagueCode	FramesToPlay
C	N

Matches

Player1	Player2	MatchDate	Frames1	Frames2
N	N	D	N opt	N opt



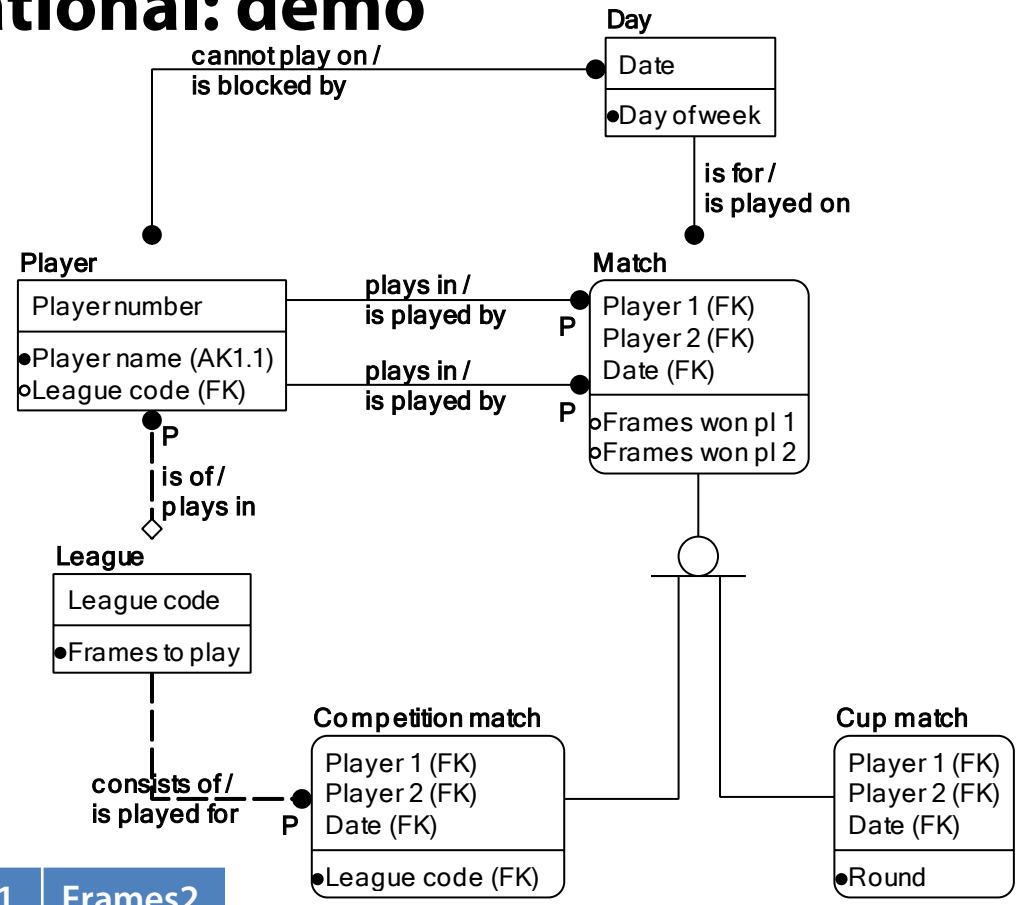
CompetitionMatches

LeagueCode	Player1	Player2	MatchDate
C	N	N	D

CupMatches

Player1	Player2	MatchDate	Round
N	N	D	C

ER to relational: demo



Calendar

Date	DayOfWeek
D	C

Players

PlayerNbr	PlayerName	LeagueCode
N	C	C opt

Leagues

LeagueCode	FramesToPlay
C	N

Matches

Player1	Player2	MatchDate	Frames1	Frames2
N	N	D	N opt	N opt

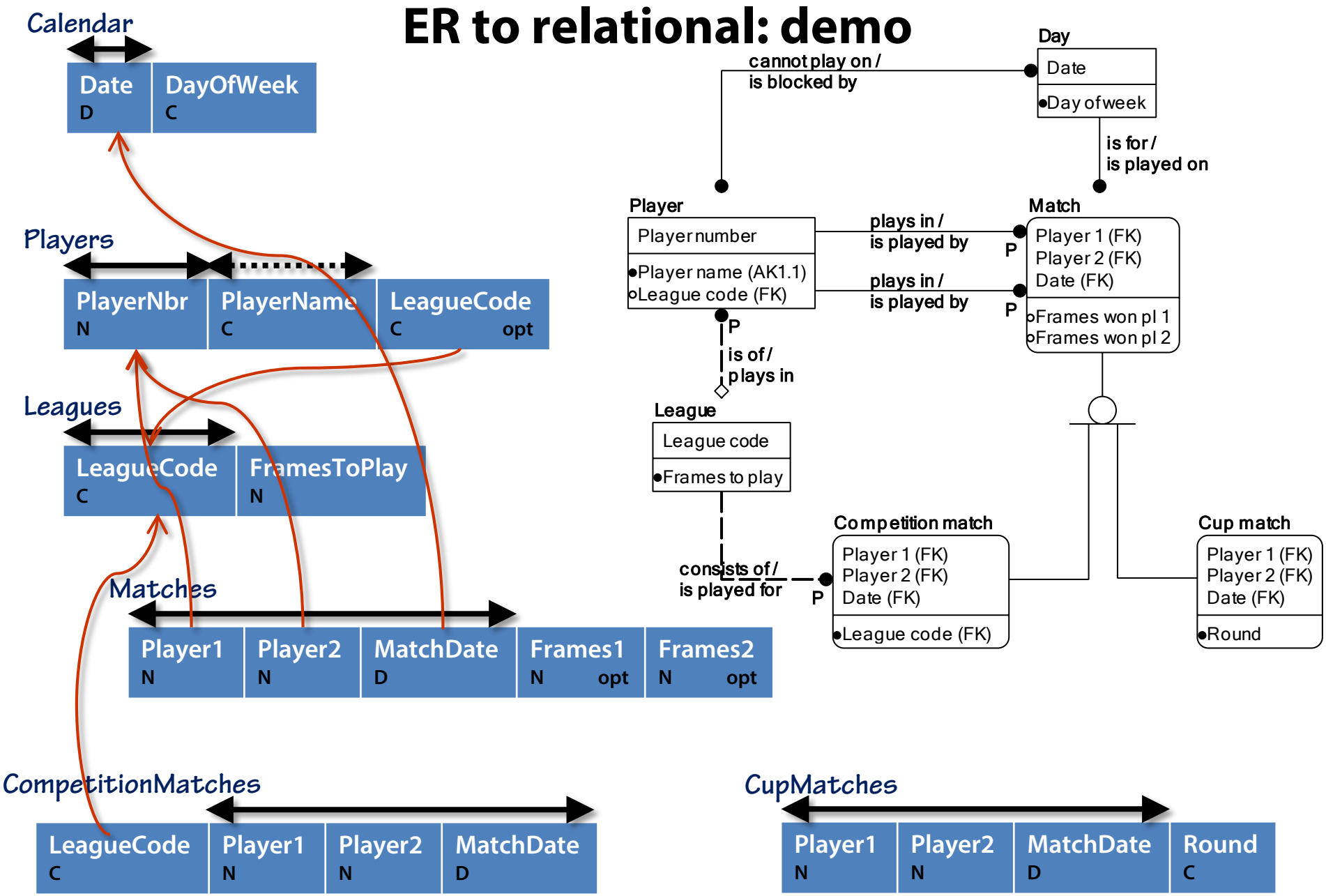
CompetitionMatches

LeagueCode	Player1	Player2	MatchDate
C	N	N	D

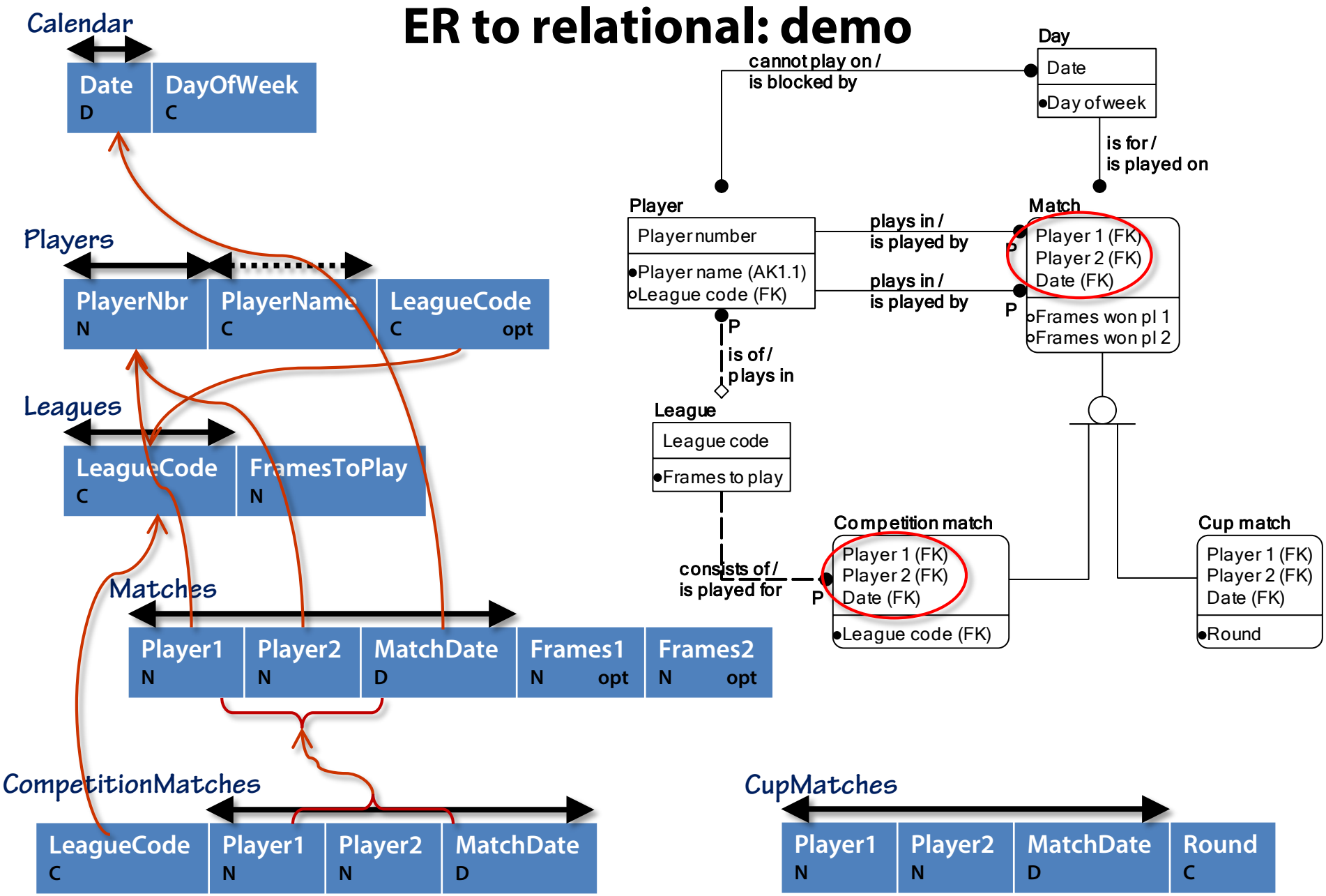
CupMatches

Player1	Player2	MatchDate	Round
N	N	D	C

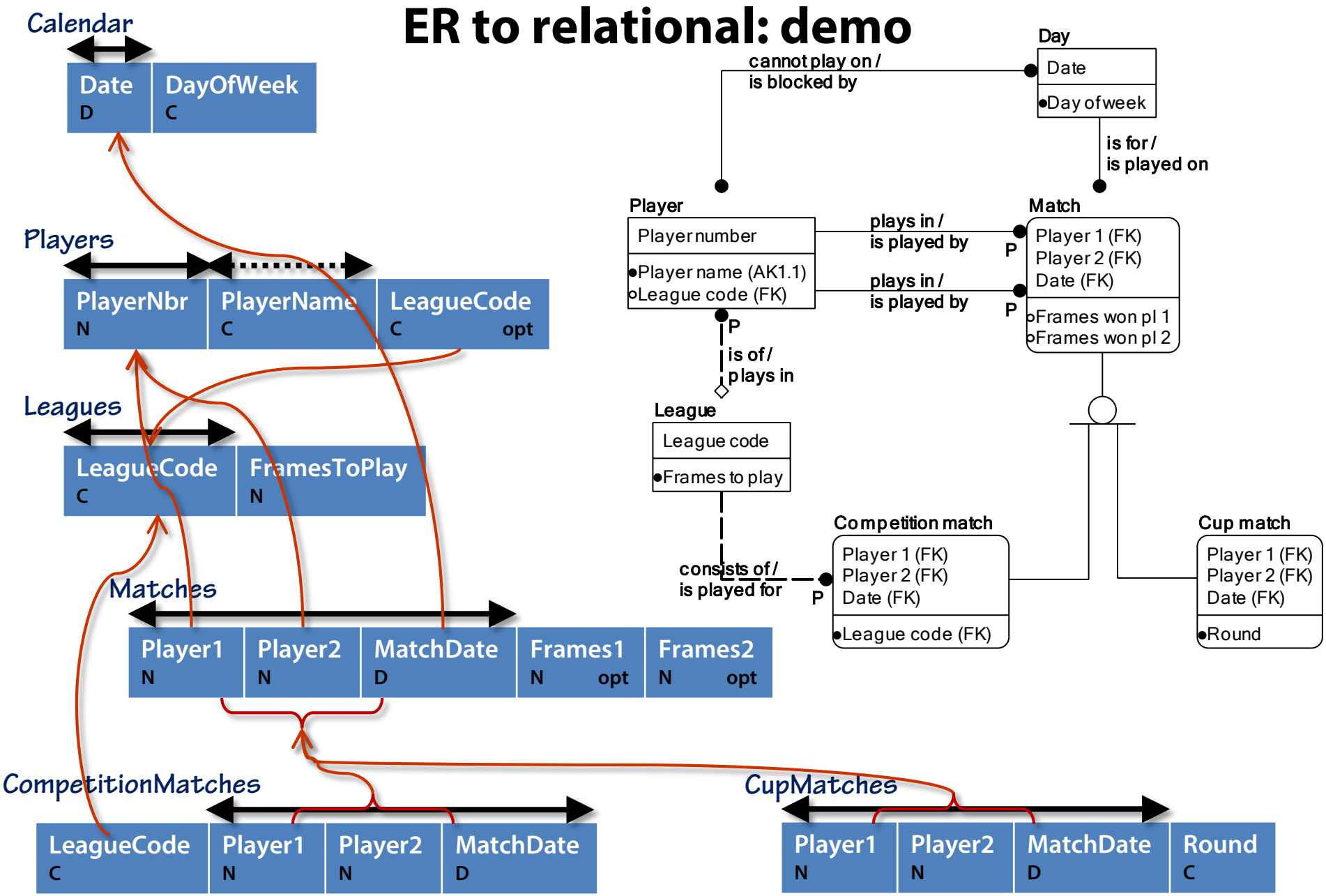
ER to relational: demo



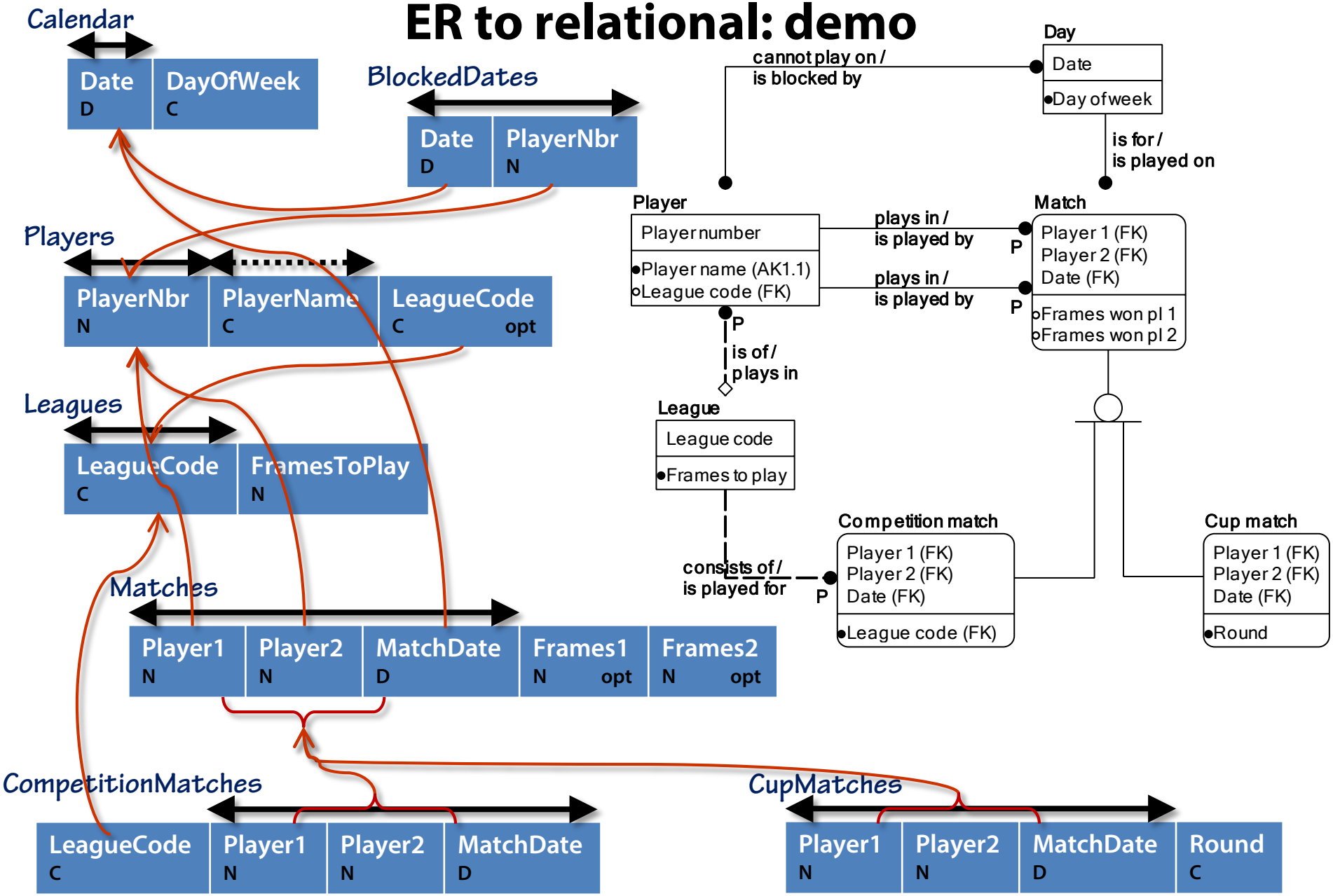
ER to relational: demo



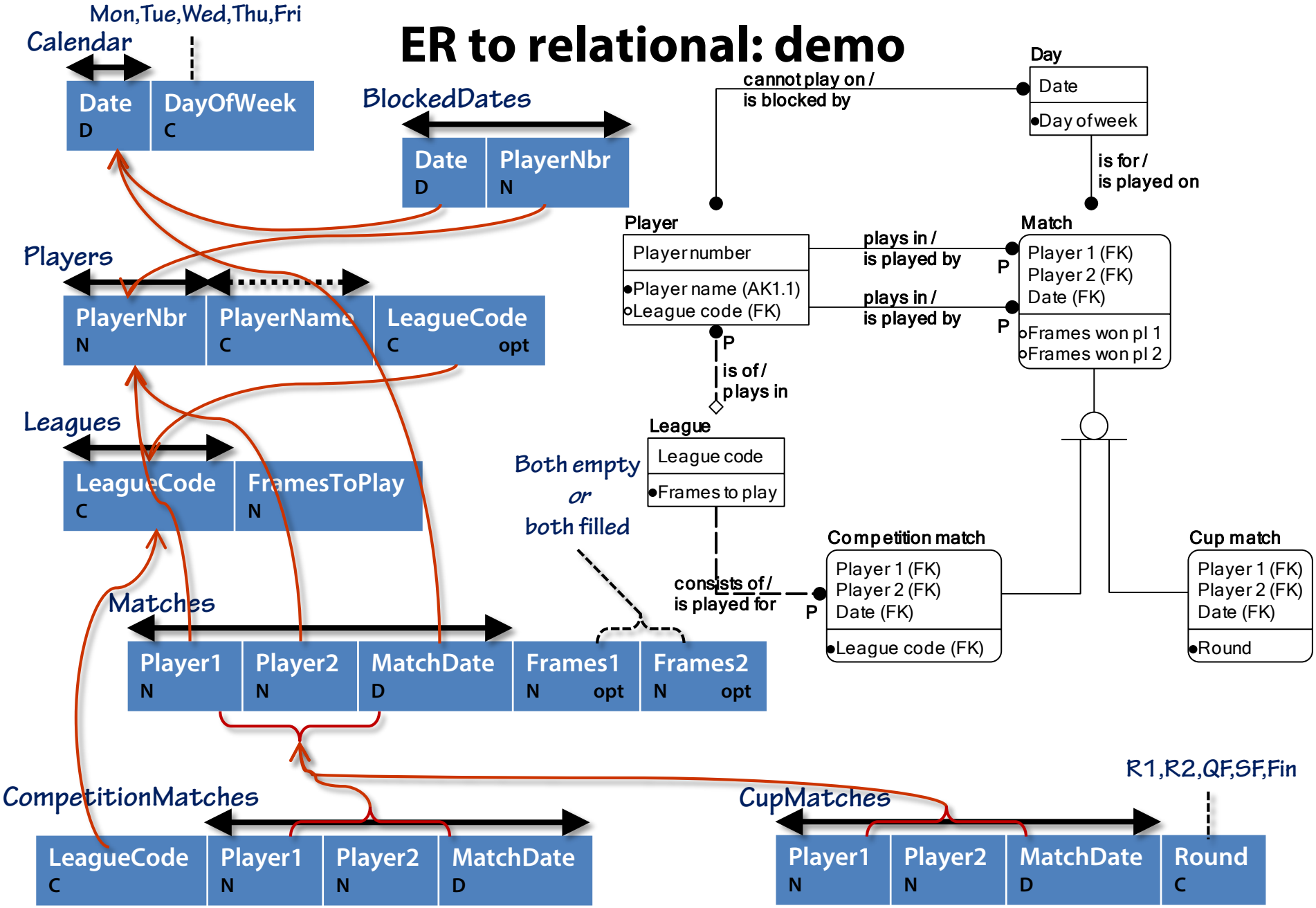
ER to relational: demo



ER to relational: demo



ER to relational: demo



Relational to ER: First draft

- **Reverse from conversion ER to relational**
  - How to handle tables?
    - Can result from entity type **OR** from many-to-many relationship!
    - For first draft, assume they all come from entity type
    - Alternative explored in second phase

Relational to ER: First draft

- **Tables → Entity types**
  - Naming?
- **PRIMARY KEY columns → Key attributes**
- **Other columns → Non-key attributes**
  - NOT NULL → mandatory / otherwise optional
- **UNIQUE constraints → Alternate keys**
- **Elements that go to the appendix:**
  - CHECK constraints
  - Assertions
  - DEFAULT constraints
  - Generated columns

Relational to ER: First draft

- **FOREIGN KEY constraints → relationships**
  - Normally one-to-many
    - Referenced table = parent entity type
    - Referencing table = child entity type
  - One-to-one when referencing columns are PRIMARY KEY / UNIQUE
  - Referencing columns NOT NULL → Mandatory relationship
  - Referencing columns nullable → Optional relationship
  - Minimum cardinality at parent side is always zero
    - Unless …
  - Relationship readings cannot be reconstructed from relational model

Relational to ER: First draft

- **FOREIGN KEY constraints → relationships**
  - All referencing columns included in the PRIMARY KEY?
    - Relationship is *identifying*
    - Referencing entity type is *weak*

  - Referencing columns **exactly equal** to the PRIMARY KEY?
    - Relationship is *subtype relationship*
    - Referencing entity type is *subtype*
    - Not possible to reconstruct:
      - Discriminator
      - Complete/incomplete
      - Mutually exclusive

Relational to ER: First draft

- **FOREIGN KEY constraints → relationships**
  - IDEF1X and other methods that include foreign key attribute
    - Done
  - ER diagramming methods that don't show foreign key attribute
    - Remove attributes that correspond to referencing columns

Relational to ER: Variations

- **First draft: all tables → entity types**
- **Alternative: some tables → many-to-many relationships**
  - Only possible for some entity types
    - All attributes included in key
    - Entity type participates in **exactly** two relationships
      - Both identifying
      - Entity type must be child in both
  - Option to replace
    - Remove entity type
    - Remove both identifying relationships
    - Replace with many-to-many relationship
      - Minimum cardinalities copied from (replaced) identifying relationships
      - Naming?

Relational to ER: Variations

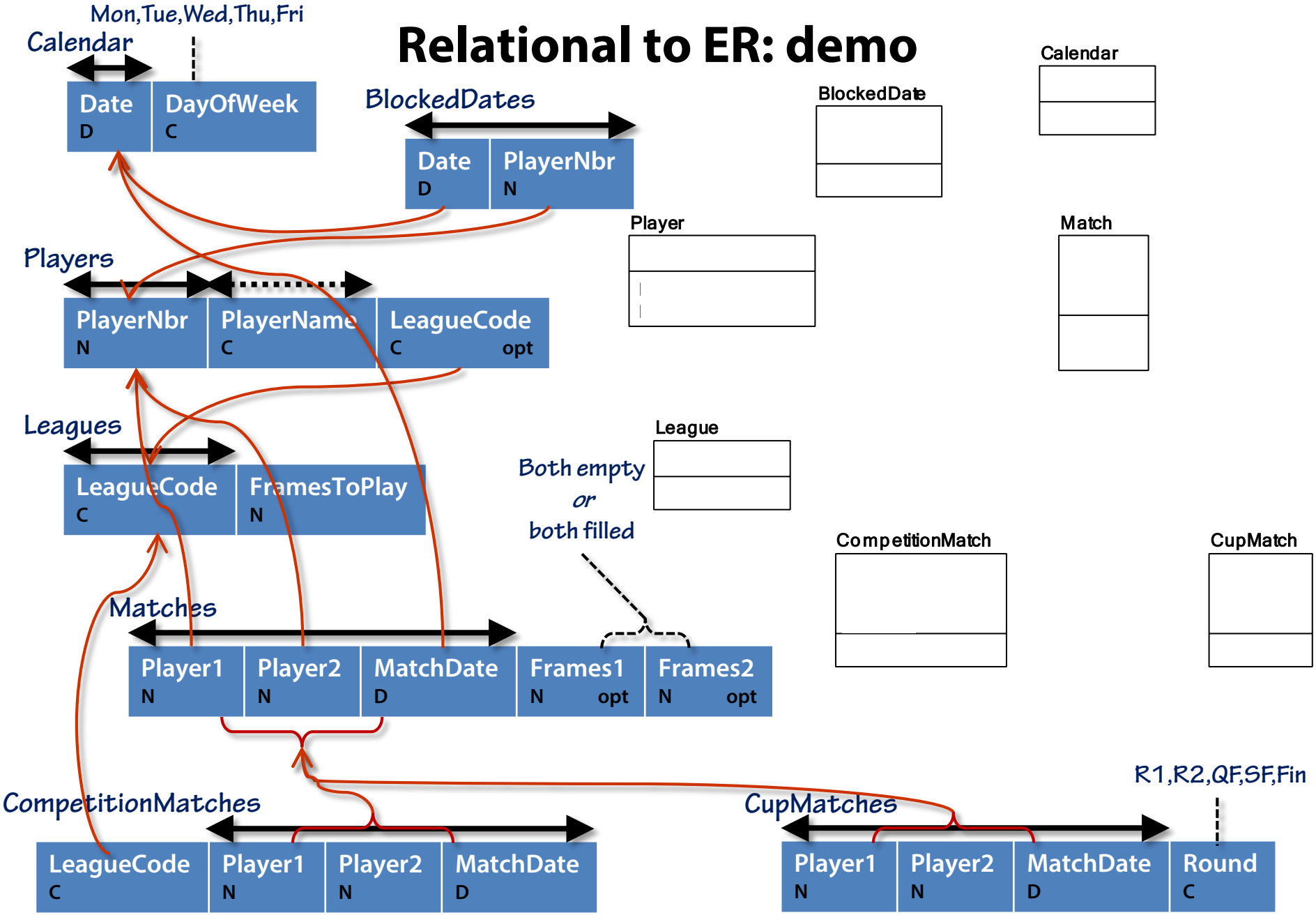
- **IDEF1X: All relationships connect TWO entity types**
- **Some methods support relationships with more entity types**
  - Only possible for some entity types
    - All attributes included in key
    - Entity type participates in **at least** two relationships
      - All identifying
      - Entity type must be child in each
  - Option to replace
    - Remove entity type
    - Remove all identifying relationships
    - Replace with many-to-many relationship or relationship with >2 entity types
      - Minimum cardinalities copied from (replaced) identifying relationships
      - Naming?

Relational to ER: Variations

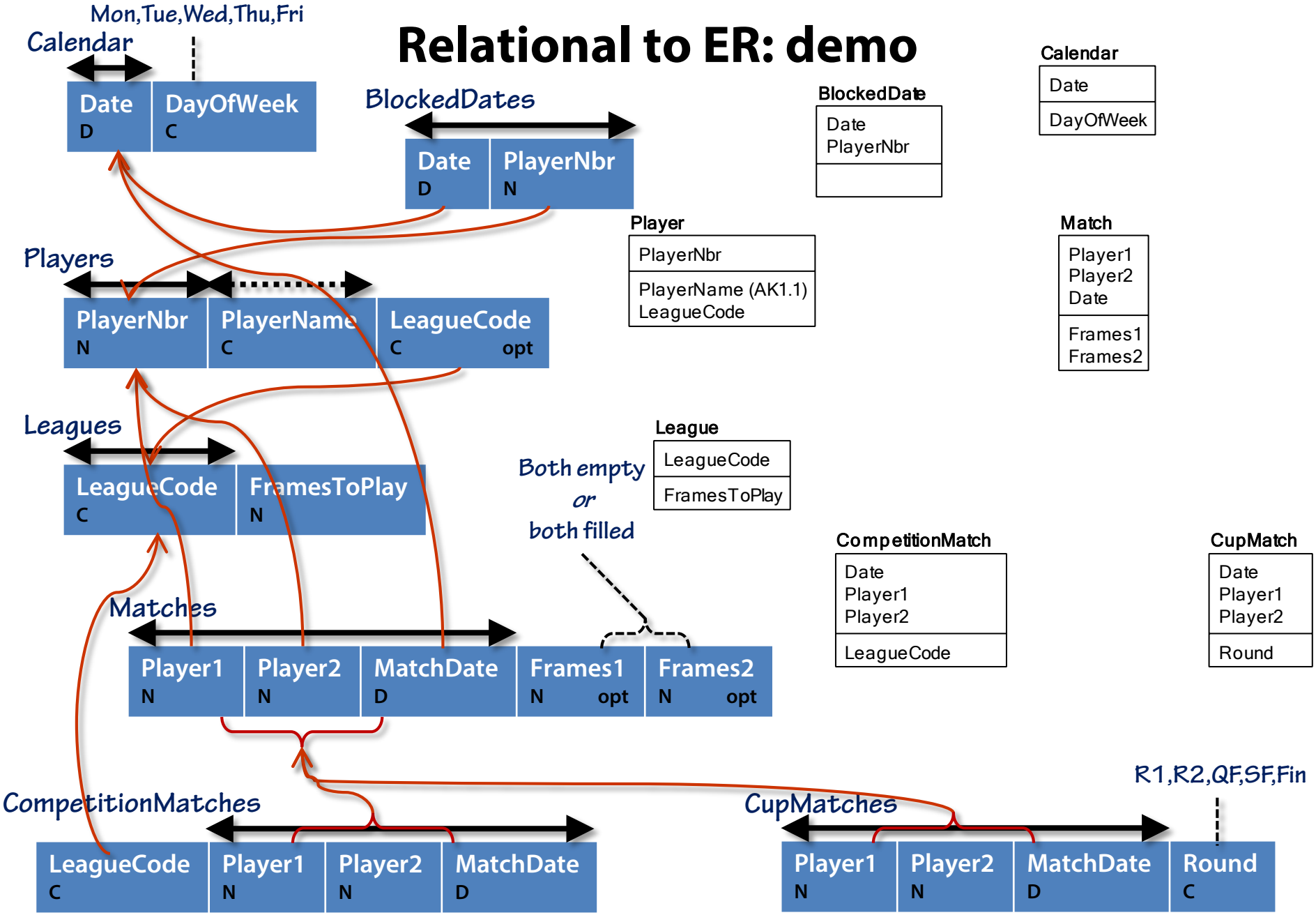
- **Use many-to-many or leave as entity type?**
  - Both versions are correct
  - Choose the one that best expresses the meaning of the facts represented
    - Might even mix & match, depending on audience

- **Use relationship between >2 entity types or leave as entity type?**
  - (Only relevant if supported by your method)
  - Both versions are correct
  - Choose the one that best expresses the meaning of the facts represented
    - Might even mix & match, depending on audience

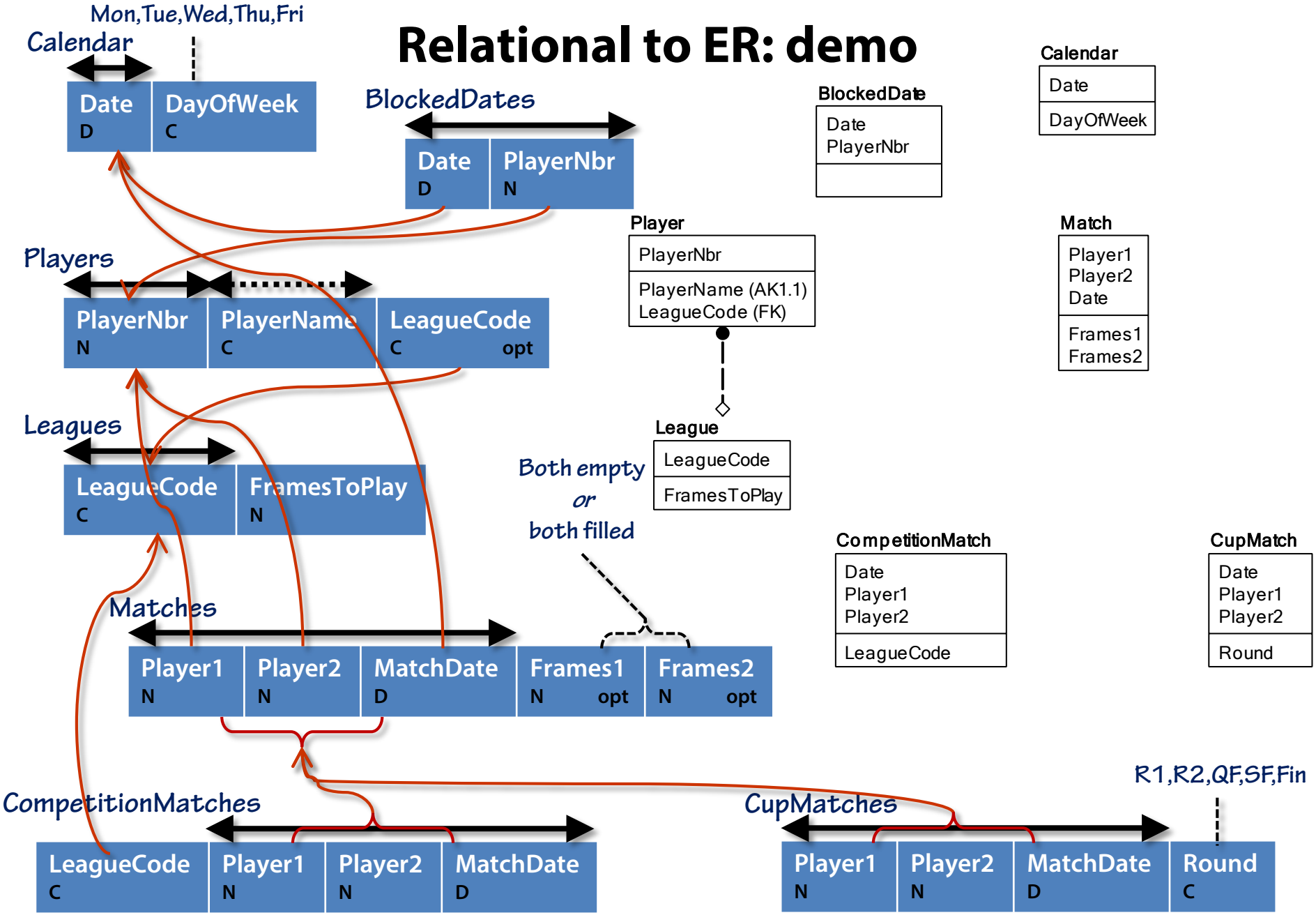
Relational to ER: demo



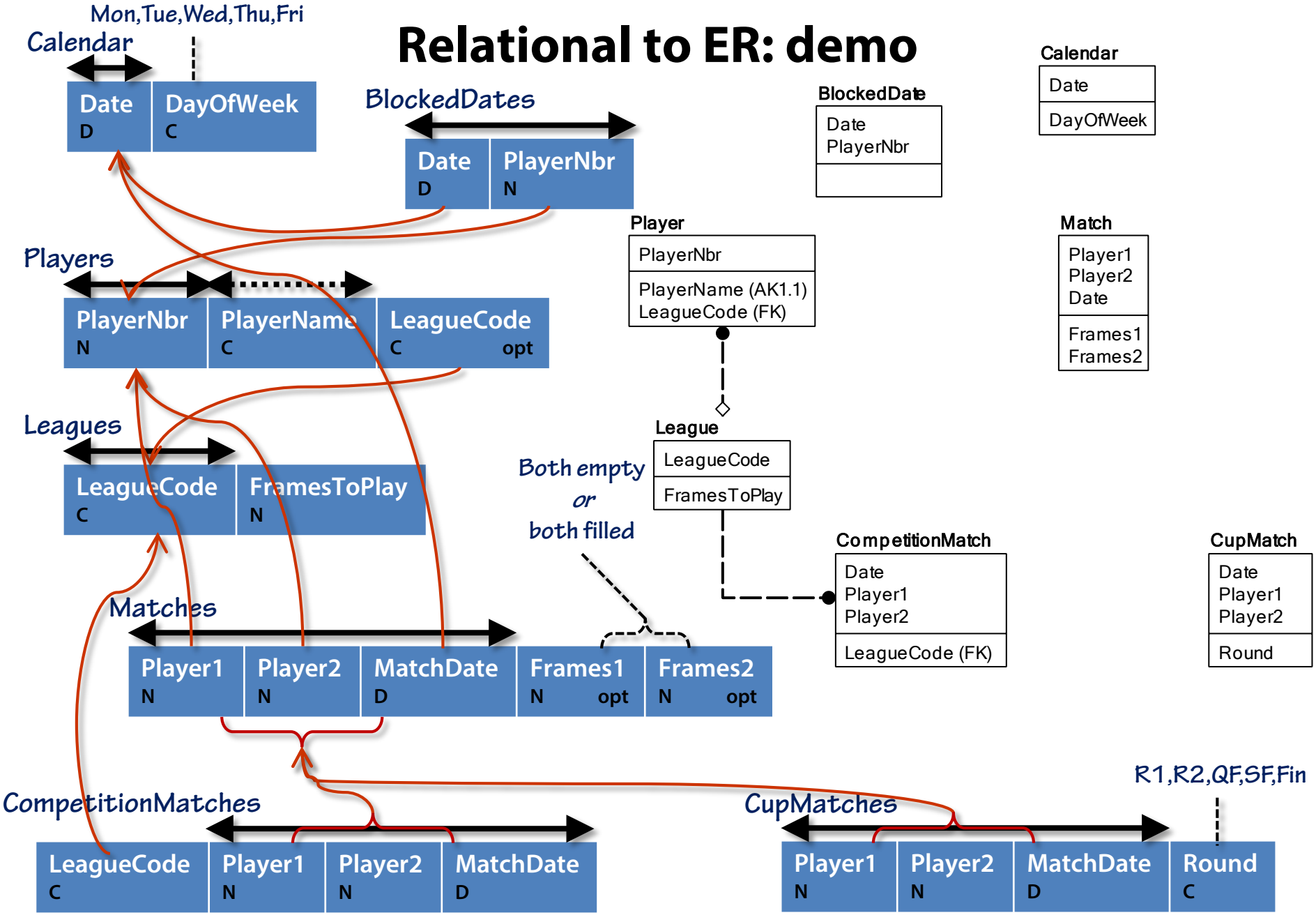
Relational to ER: demo



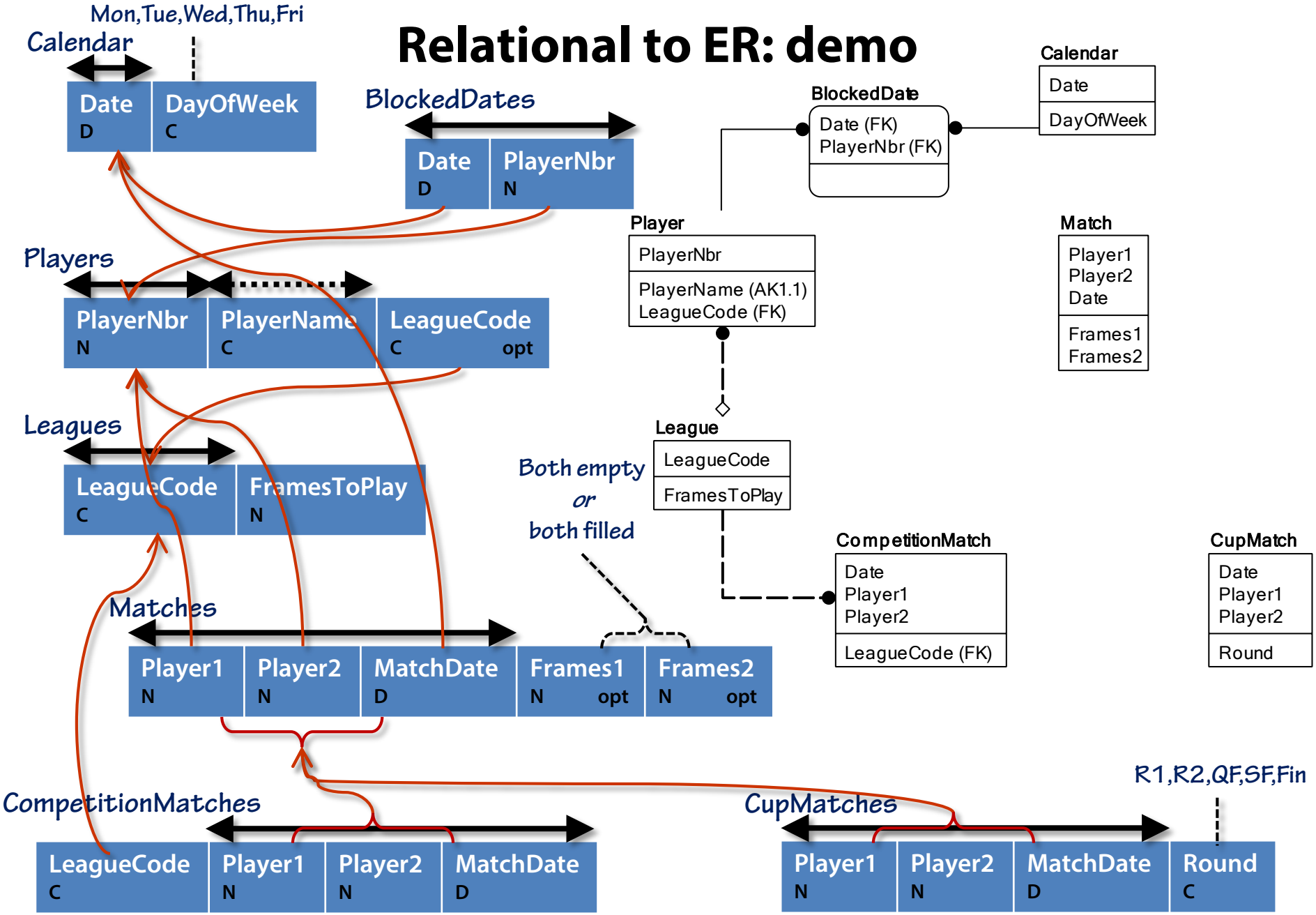
Relational to ER: demo



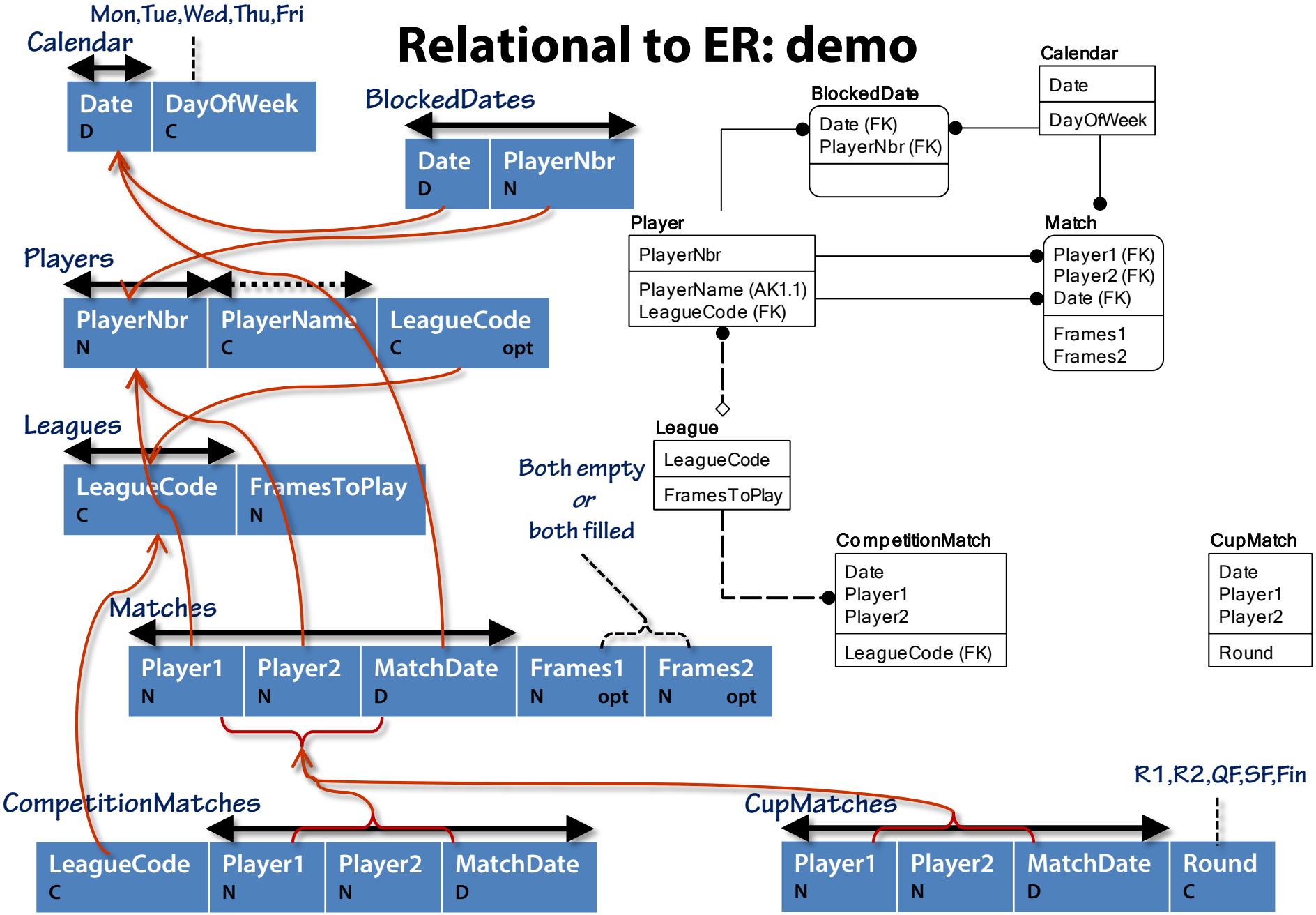
Relational to ER: demo



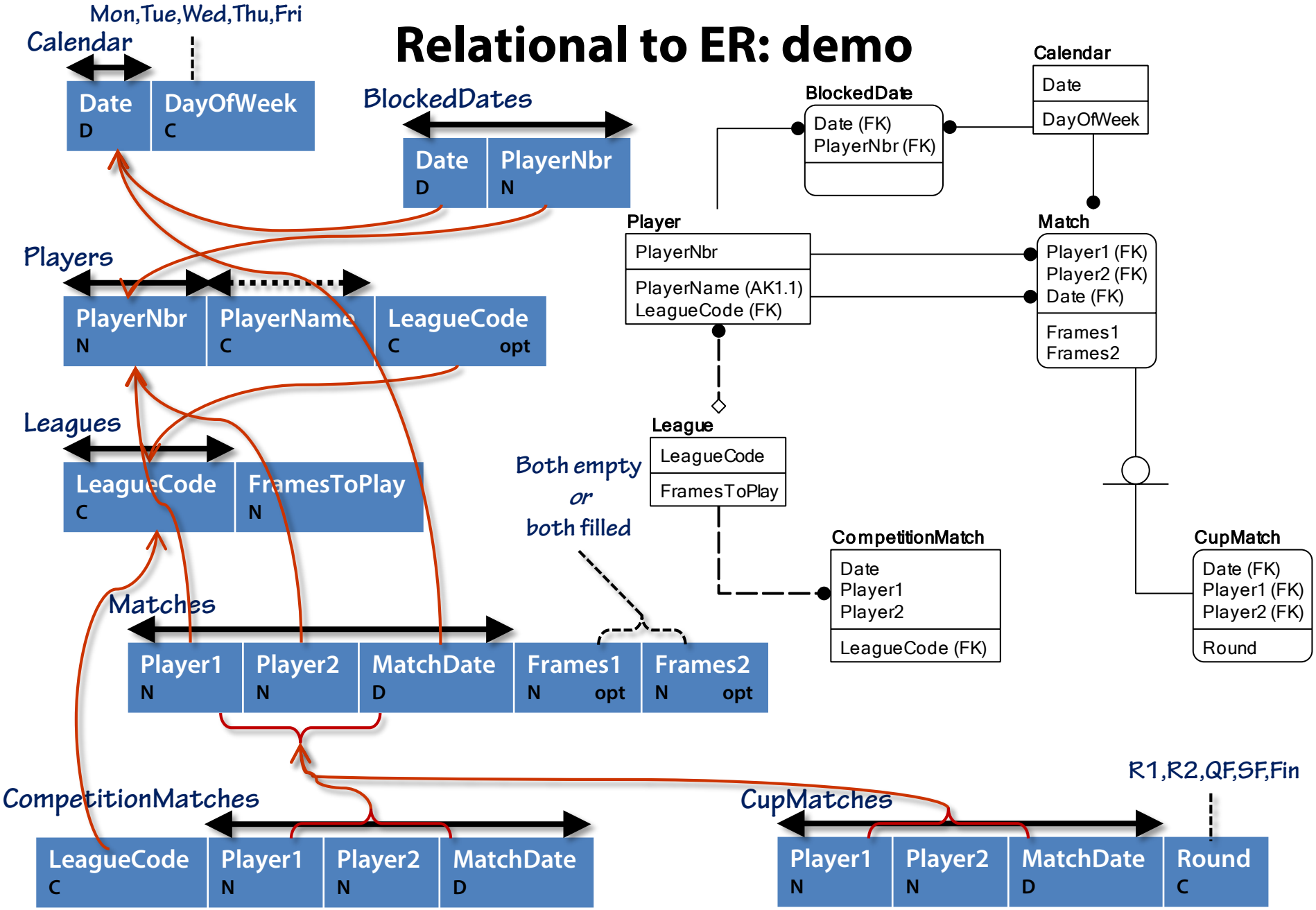
Relational to ER: demo



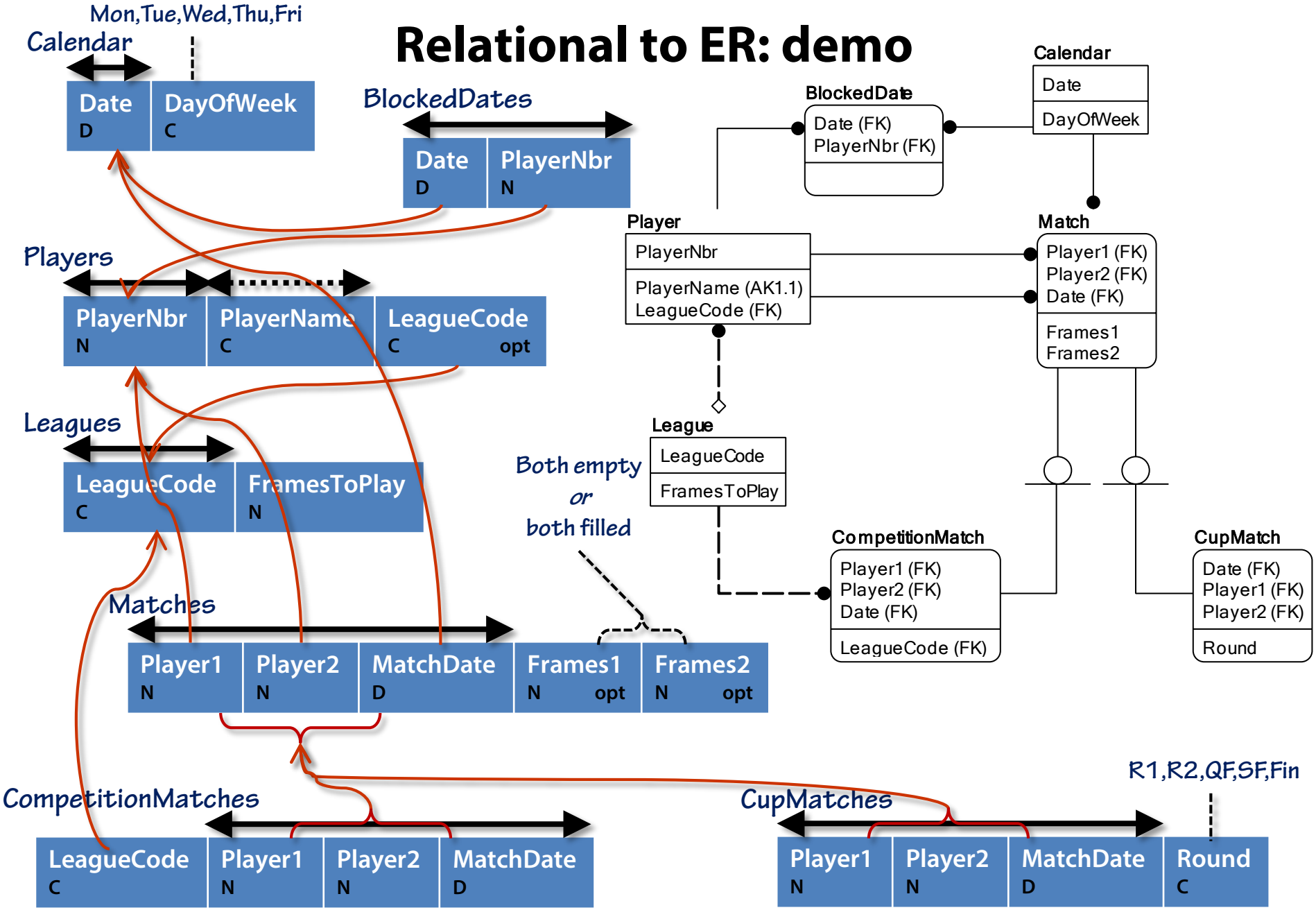
Relational to ER: demo



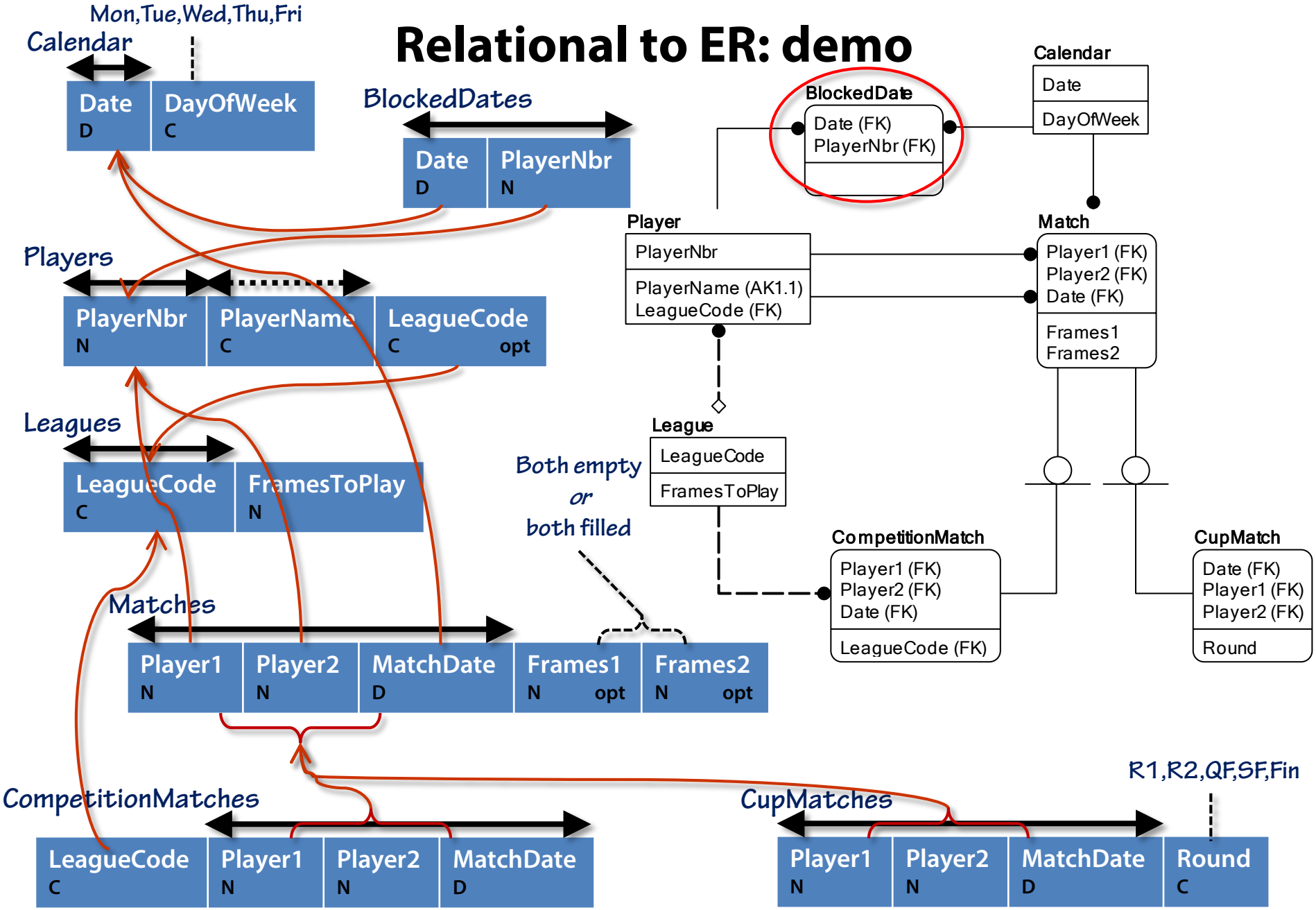
Relational to ER: demo



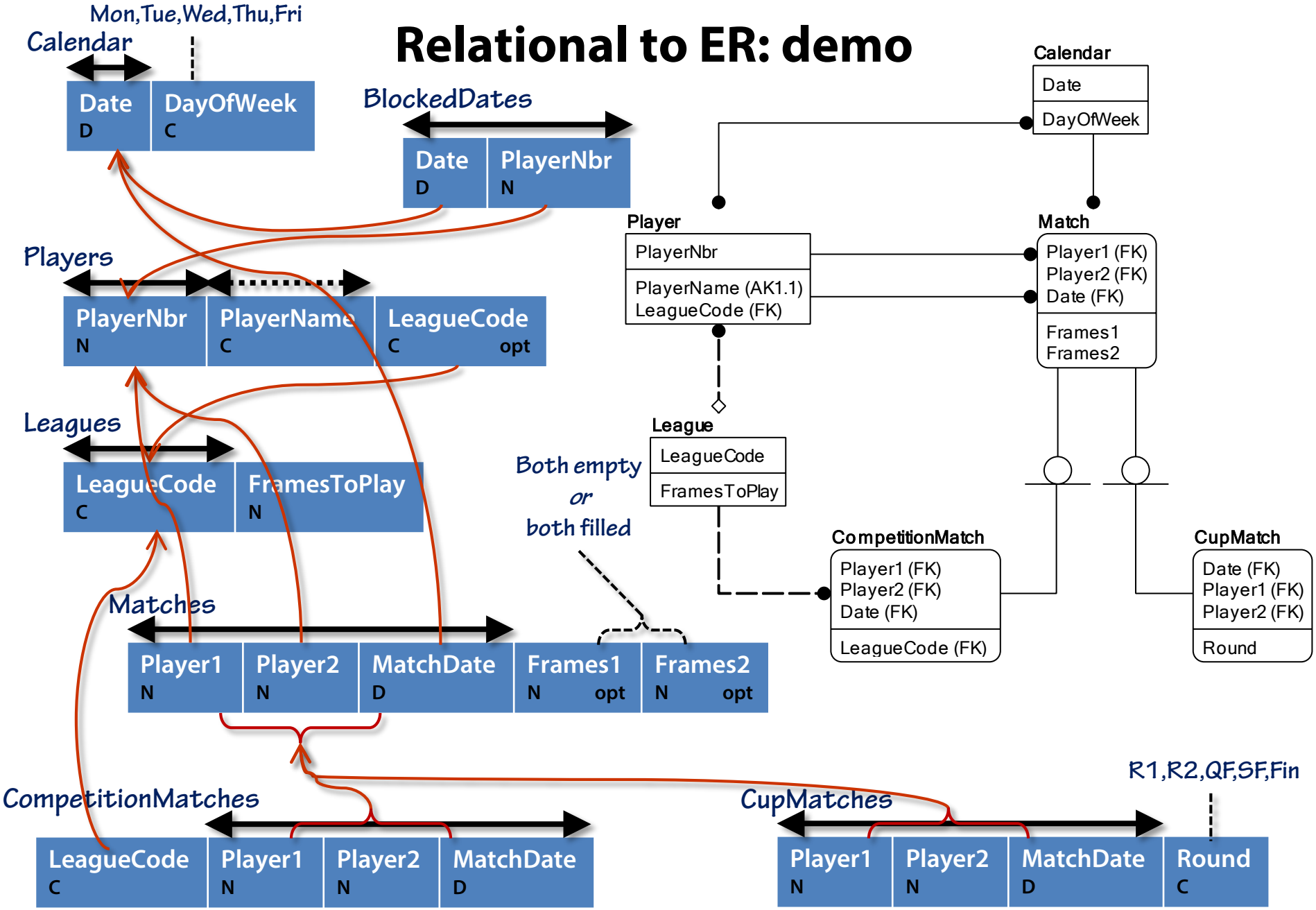
Relational to ER: demo



Relational to ER: demo



Relational to ER: demo



Summary

- **Convert between ER and relational**
  - Reasons for two representations
- **How to represent a relational design**
- **From ER to relational**
  - Entity types
  - Attributes
  - Keys
  - Relationships
    - One-to-many / one-to-one
    - Many-to-many
  - Other elements
- **From relational to ER**
  - First version
  - Alternatives (with many-to-many relationships)