Bridge 2

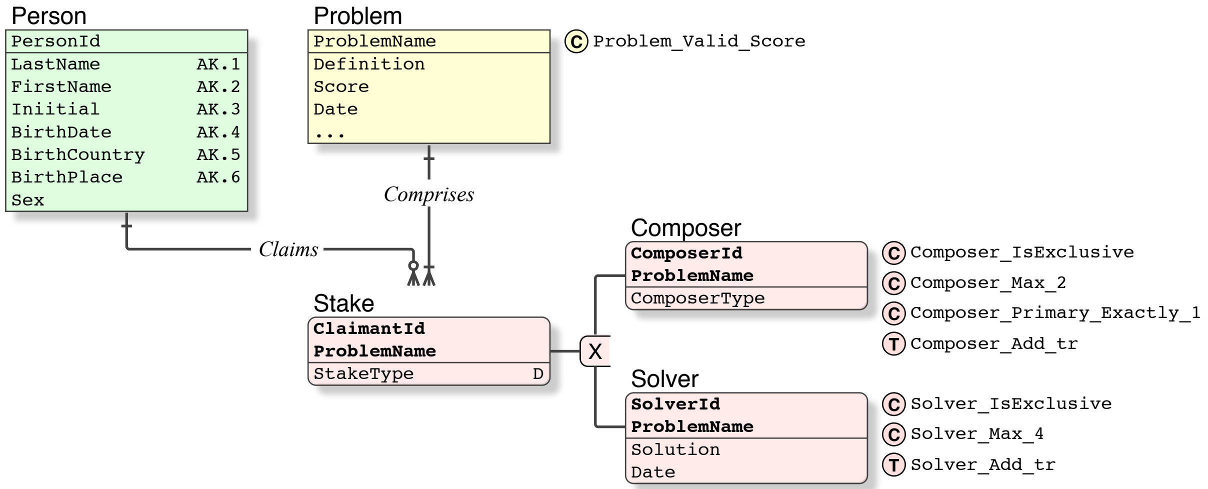
Relational Data Model

History

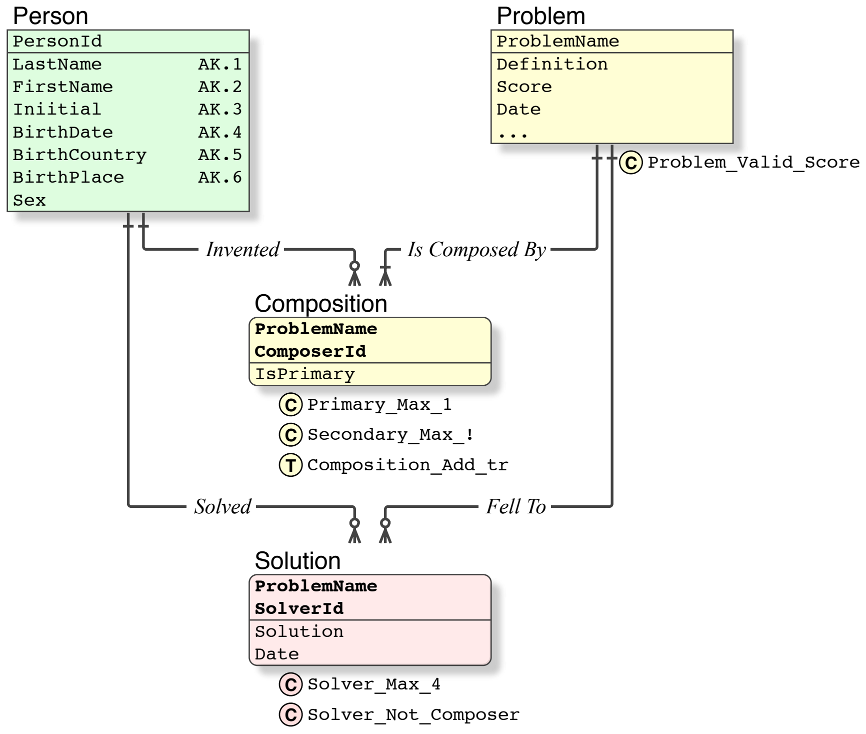
Increment 1, **Increment 2**, **Increment 3**.

This is Increment 4.

Vitacolonna Model



Asirvadem Model



Requirement

- Database for playing Bridge, for a Bridge club.
- Track:
  Players
    Problems, composed, to be solved
      Composers
      Solvers
- Service any report request with a single `SELECT`.

```
Ⓣ Composer_Add_tr
  Xact( [ +Problem(), ]
        +Stake(), +Composer() )
Ⓣ Solver_Add_tr
  Xact( +Stake(), +Solver() )
```

Resolution

> *Role ranges over {primary, secondary, solver}.*

The Discriminator is then lost. Resolved by:

- adding `StakeType{Composer|Solver}`.
- adding `ComposerType{Primary|Secondary}`.

Naming

I have taken liberties, based on my naming standard. The benefits may be obvious. Reading the Predicates should entertain.

RoleName

The cumbersome `<Role.PK>` is replaced with `<Role>`. That which it is a RoleName of, is obvious.

ConstraintName

For brevity the standard prefix of TableName, which is also obvious, is omitted. The content is a name (not a clause), which should be in the expansion below.

Constraint

One constraint per <LogicCondition>, rather than per <LogicClause> or per <Table>, allows each to be enabled discretely.

```
Ⓒ Composition_Primary_Max_1
  CHECK Composer.IsPrimary[ @ProblemName ] NOT EXISTS
Ⓒ Composition_Secondary_Max_1
  CHECK Composer.IsPrimary[ @ProblemName ] EXISTS AND
  CHECK ( Composer.IsPrimary[ @ProblemName ] = 0 ) NOT EXISTS
Ⓣ Composition_Add_tr
  Xact( [ +Problem(), ]
        +Composition() )
Ⓒ Solver_Max_4
  CHECK COUNT( Solution[ @ProblemName ] ) < 4
Ⓒ Solver_Not_Composer
  CHECK Composition[ @ProblemName, @SolverId ] NOT EXISTS
Ⓣ Solution_Add_tr
  Xact[ +Solution{} ]
```

Entity Type

🟩	Identifying Entity
🟨	Transaction
🟥	TransactionDetail

IDEF1X Notation

Blank