

**Metamodeling the Entity Relationship and Relational Data Model
and
their Transformation**

Martin Gogolla

Transformation of an ER Schema into a Relational DB Schema

- Entity → Relational schema
Entity key attributes → Key attributes in relational schema
Non-key attributes → non-key attributes in relational schema
- Relship → Relational schema
Each 'arm' of the relship →
 Key attributes of entity into relational schema
 Key of relational schema consists of key attributes of all entities
Attributes of relship → Attributes in relational schema
- Example

ER schema:

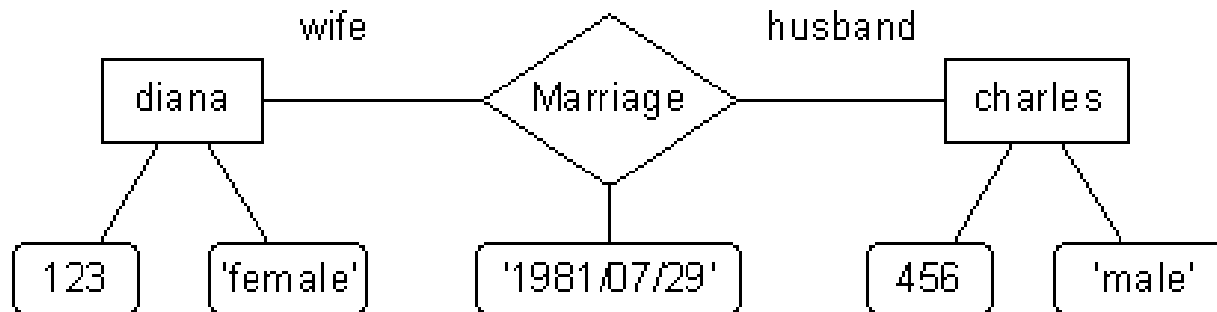
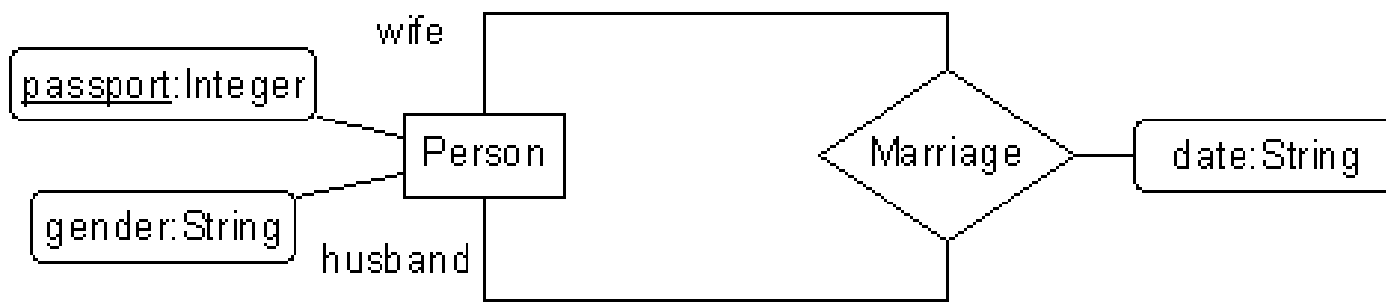
Entity [Person(passport,gender)]

Relship < Marriage(wife:Person,husband:Person;date:String) >

Relational DB schema:

Person(passport,gender)

Marriage(wife_passport,husband_passport,date)

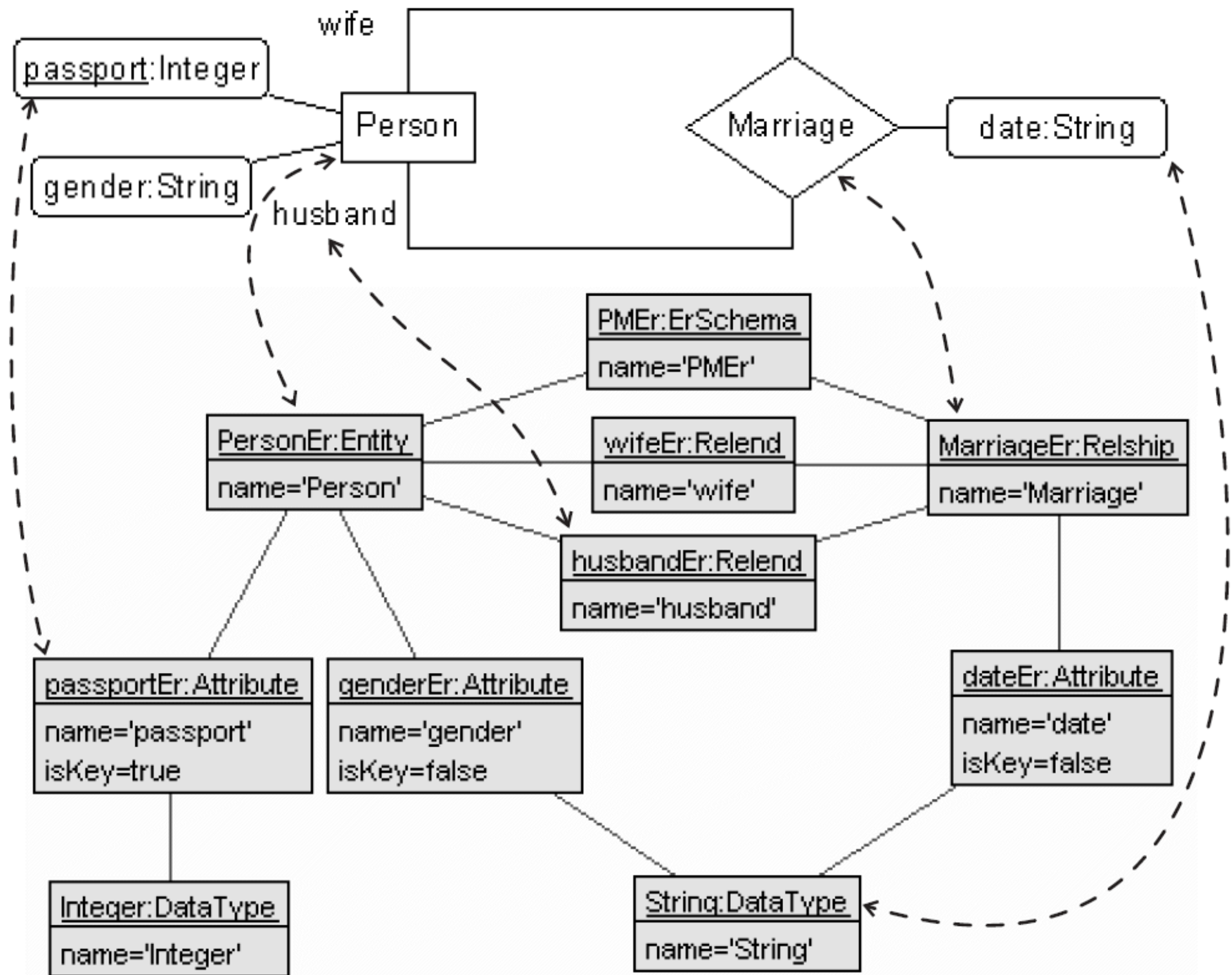


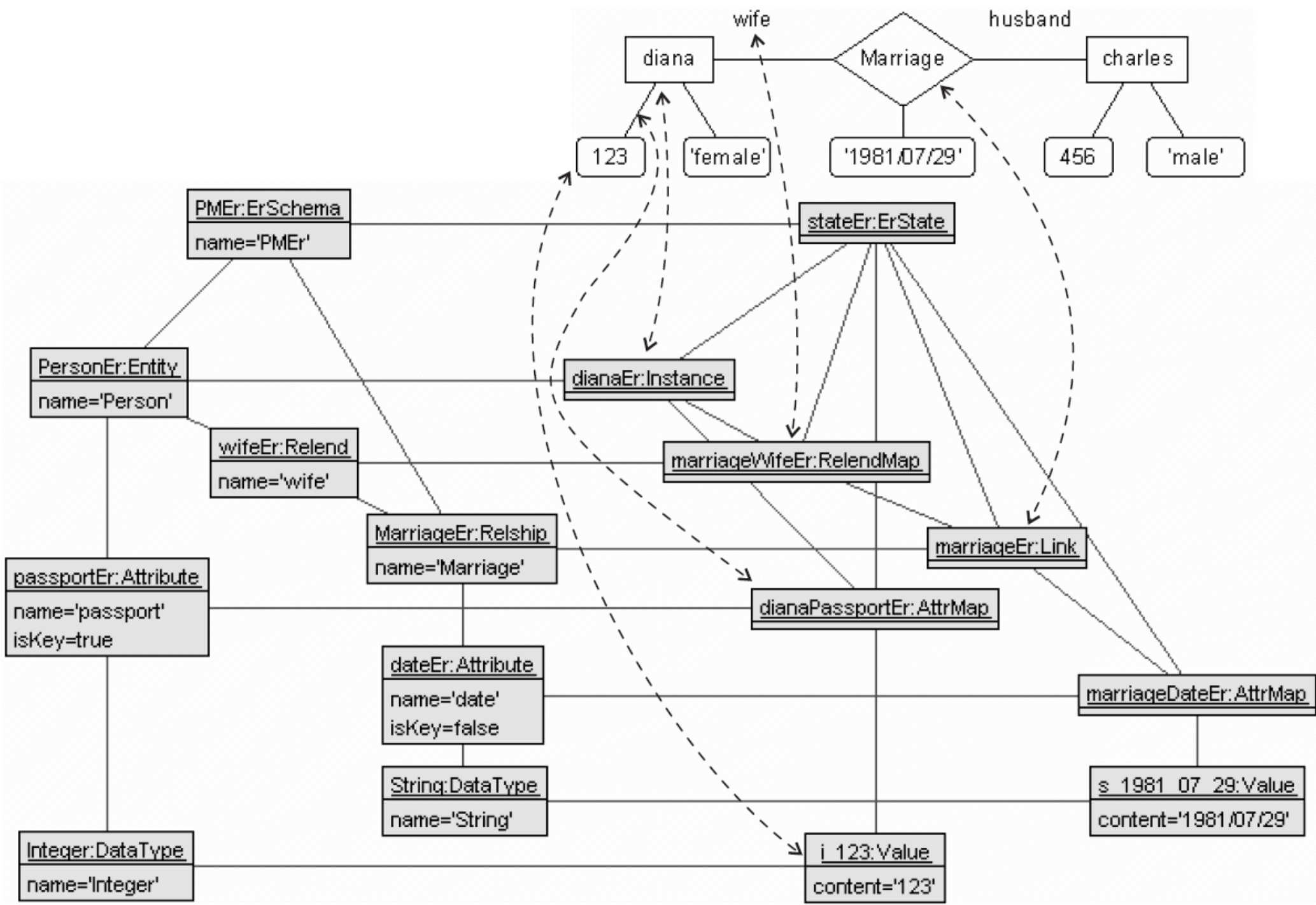
```
Person(passport: Integer, gender: String)
```

```
Marriage(wife passport: Integer, husband passport: Integer, date: String)
```

Person	passport	gender
	123	'female'
	456	'male'

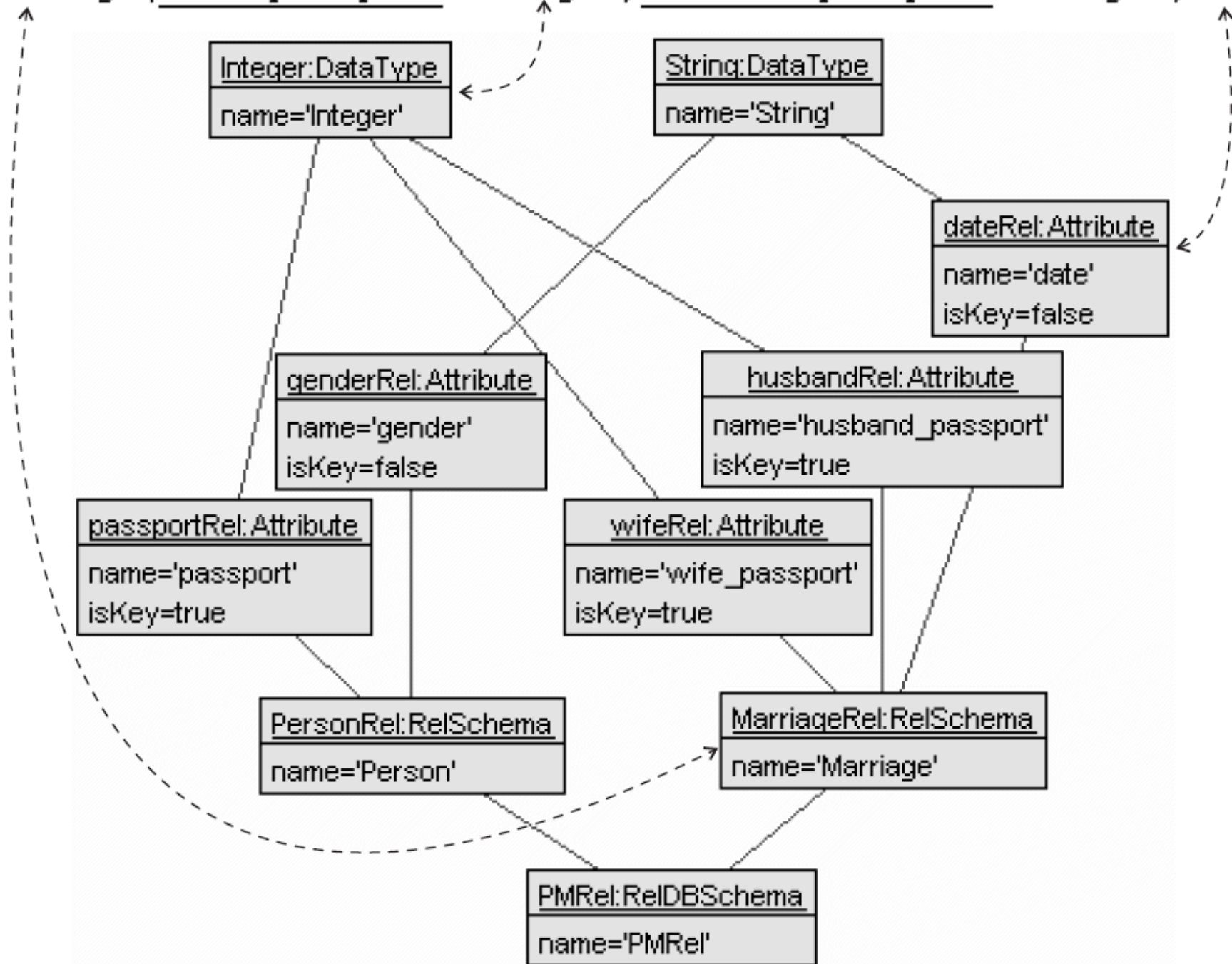
Marriage	wife_passport	husband_passport	date
	123	456	'1981/07/29'





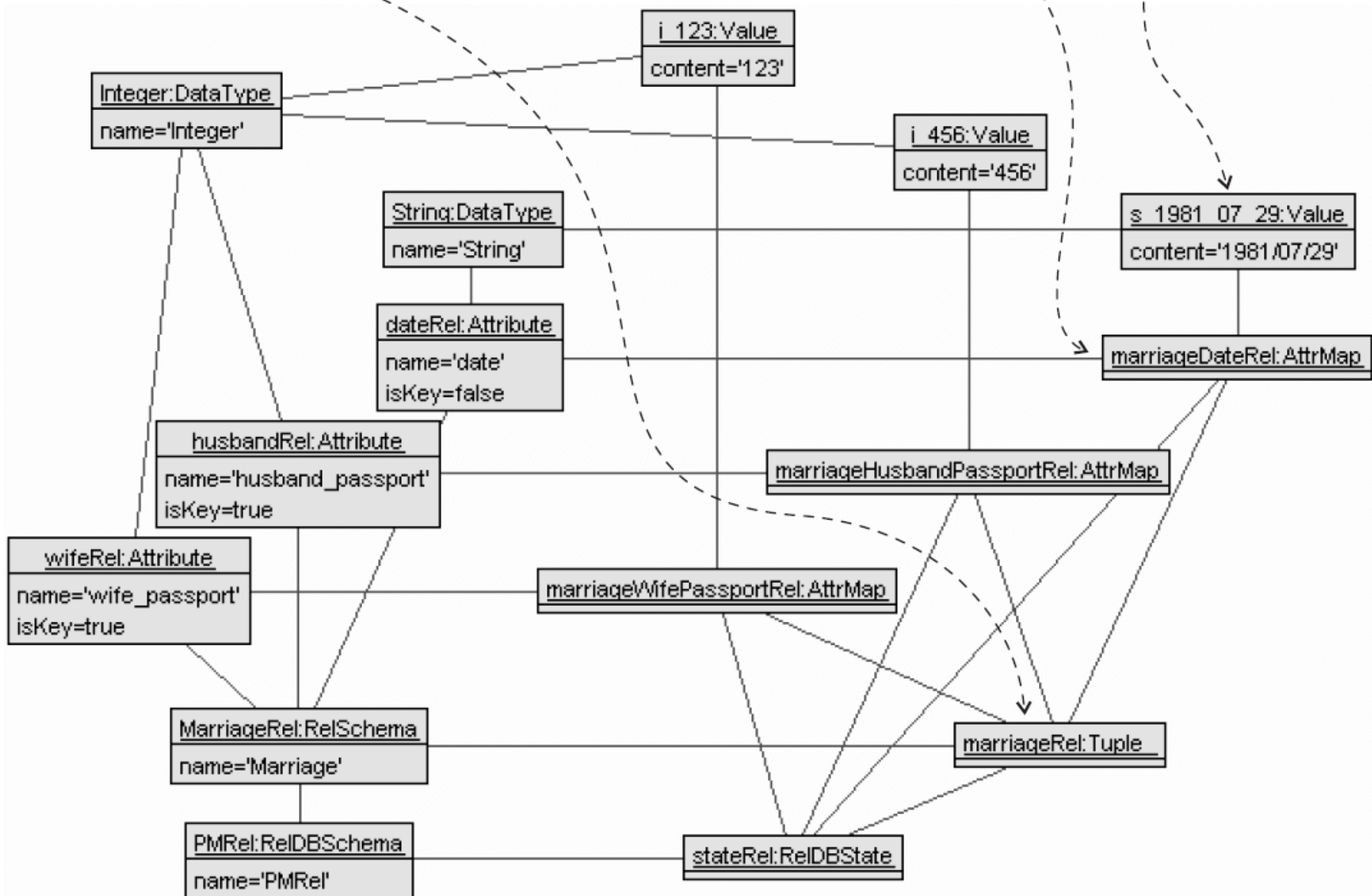
Person(passport: Integer, gender: String)

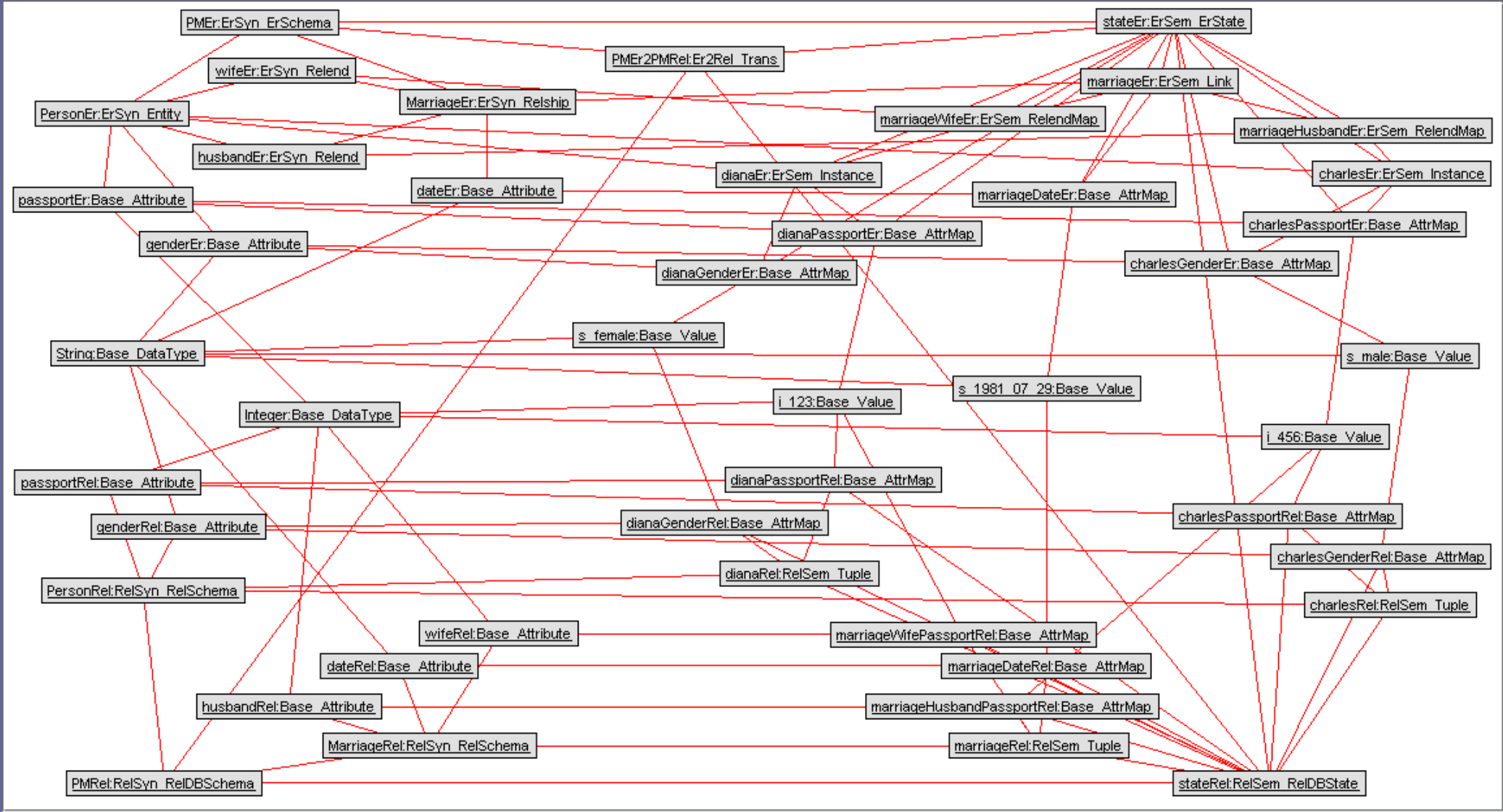
Marriage(wife passport: Integer, husband passport: Integer, date: String)

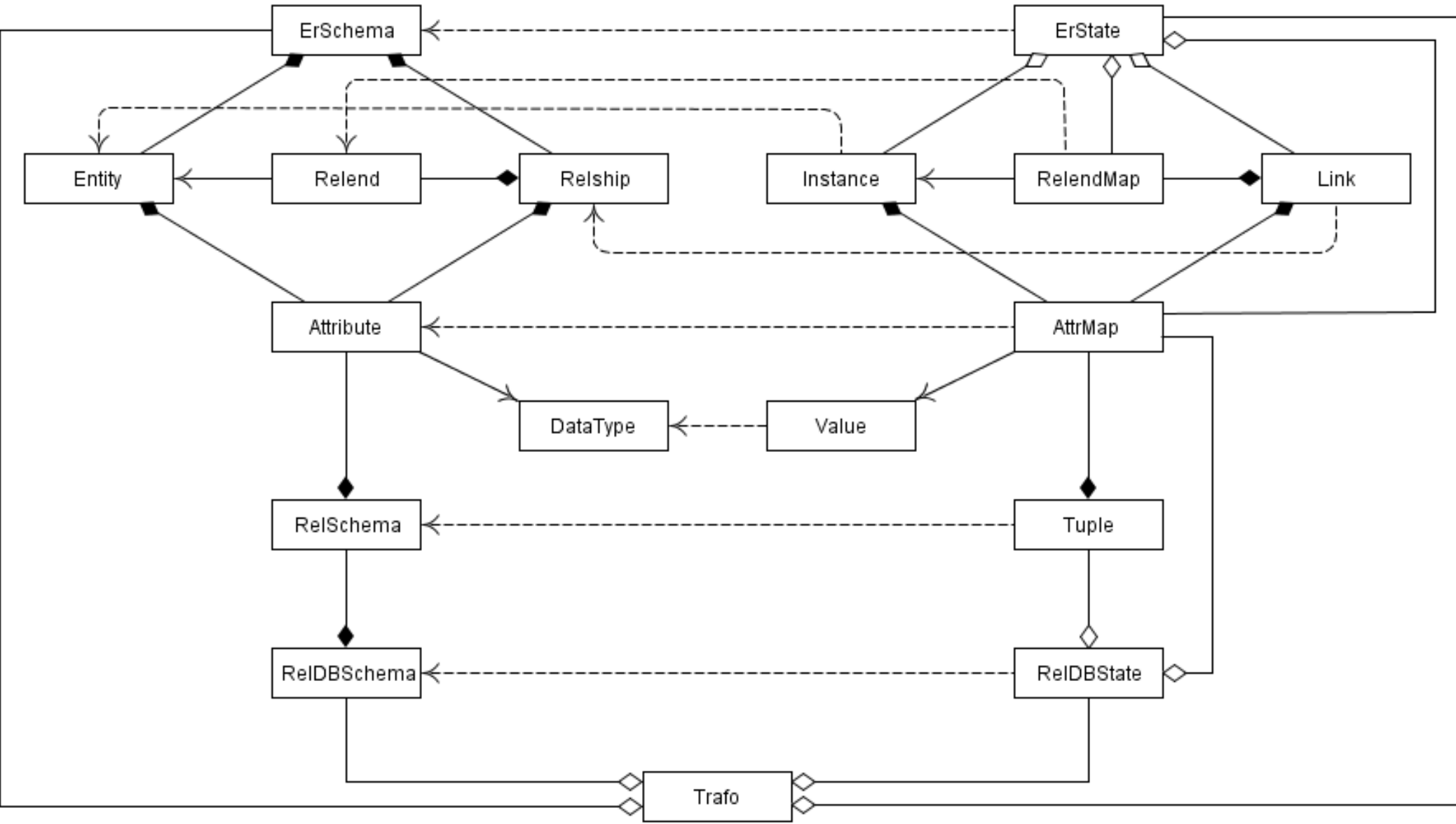


Marriage | wife_passport | husband_passport | date

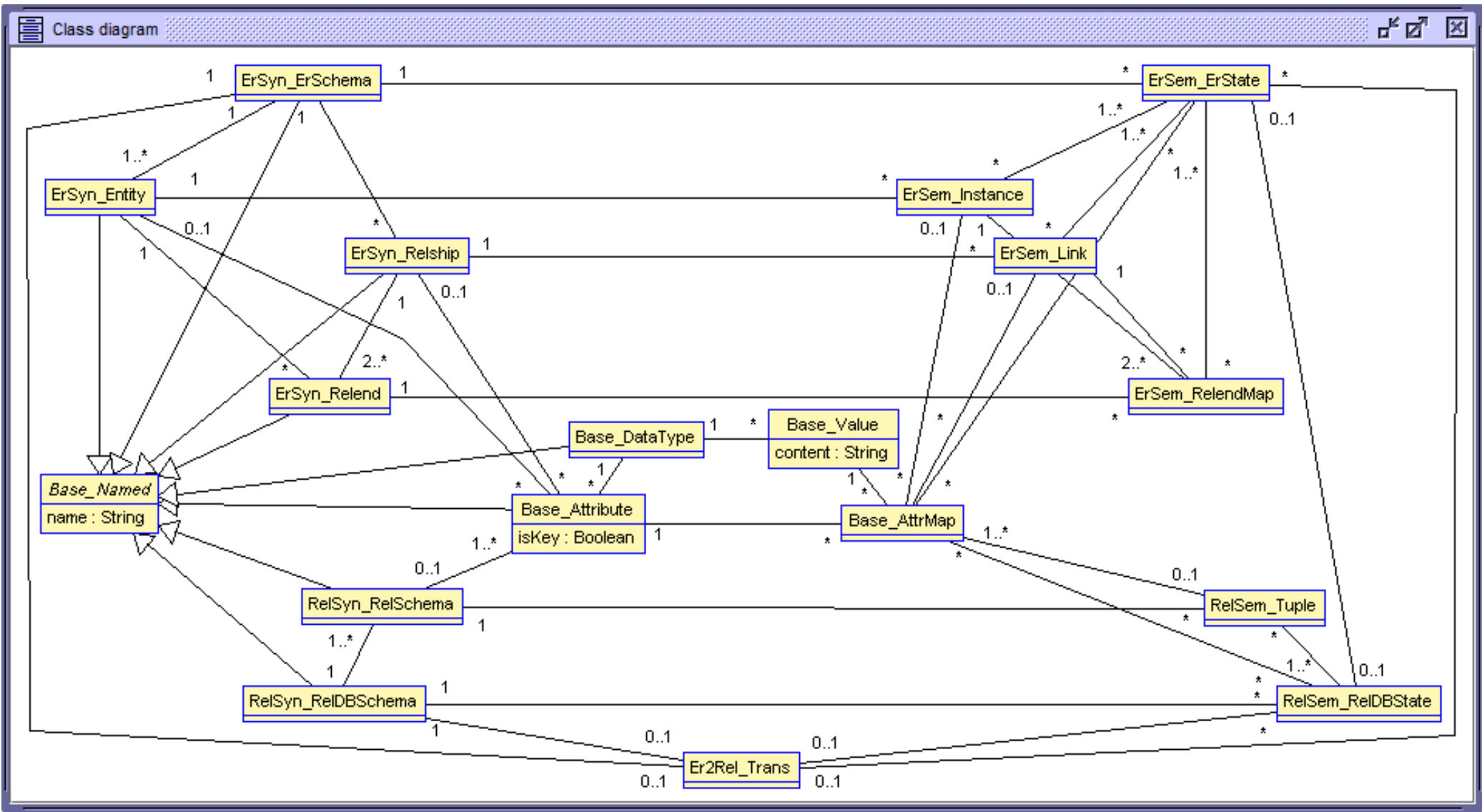
-----+-----+-----+-----
| 123 | 456 | '1981/07/29'

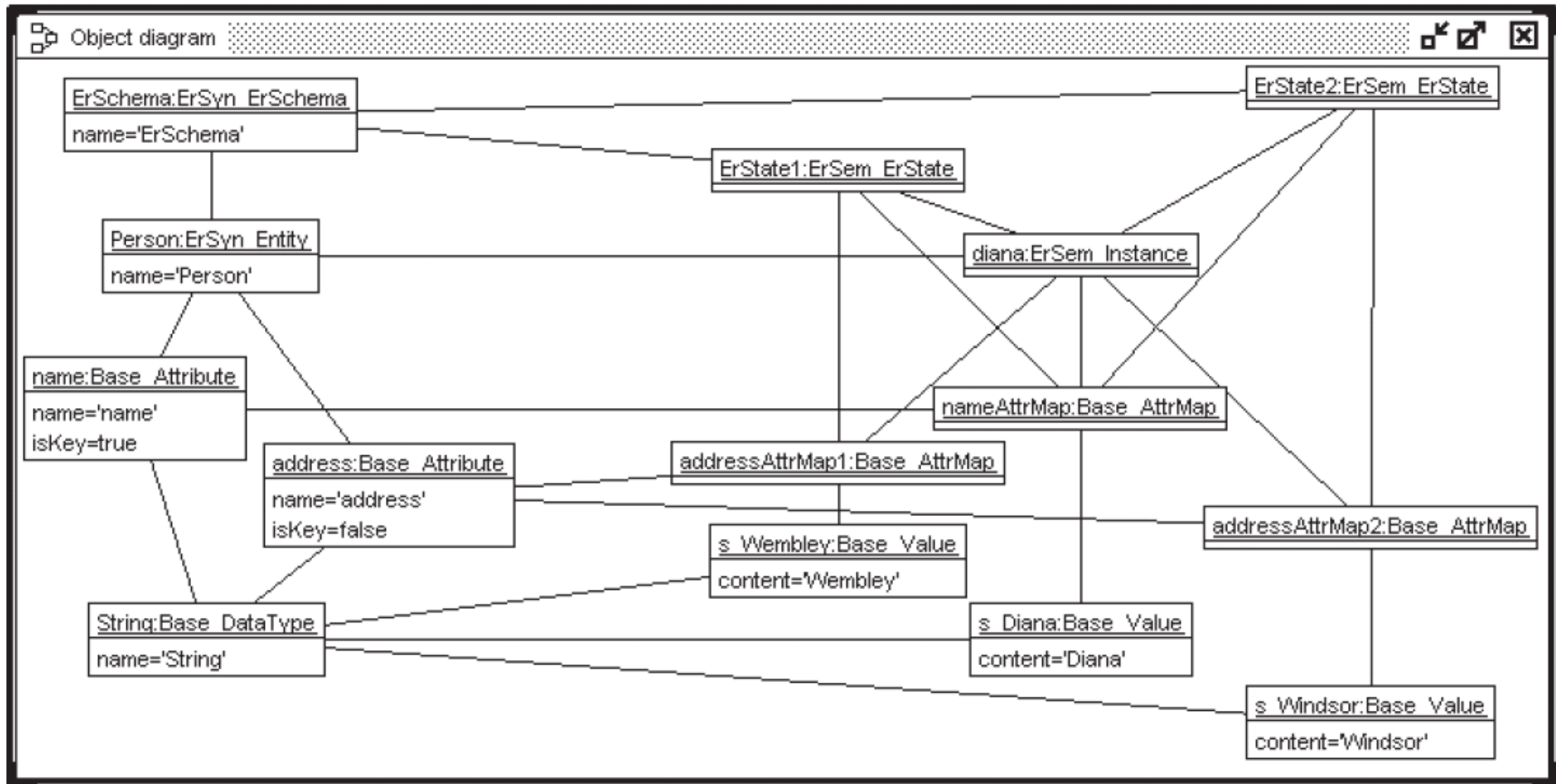






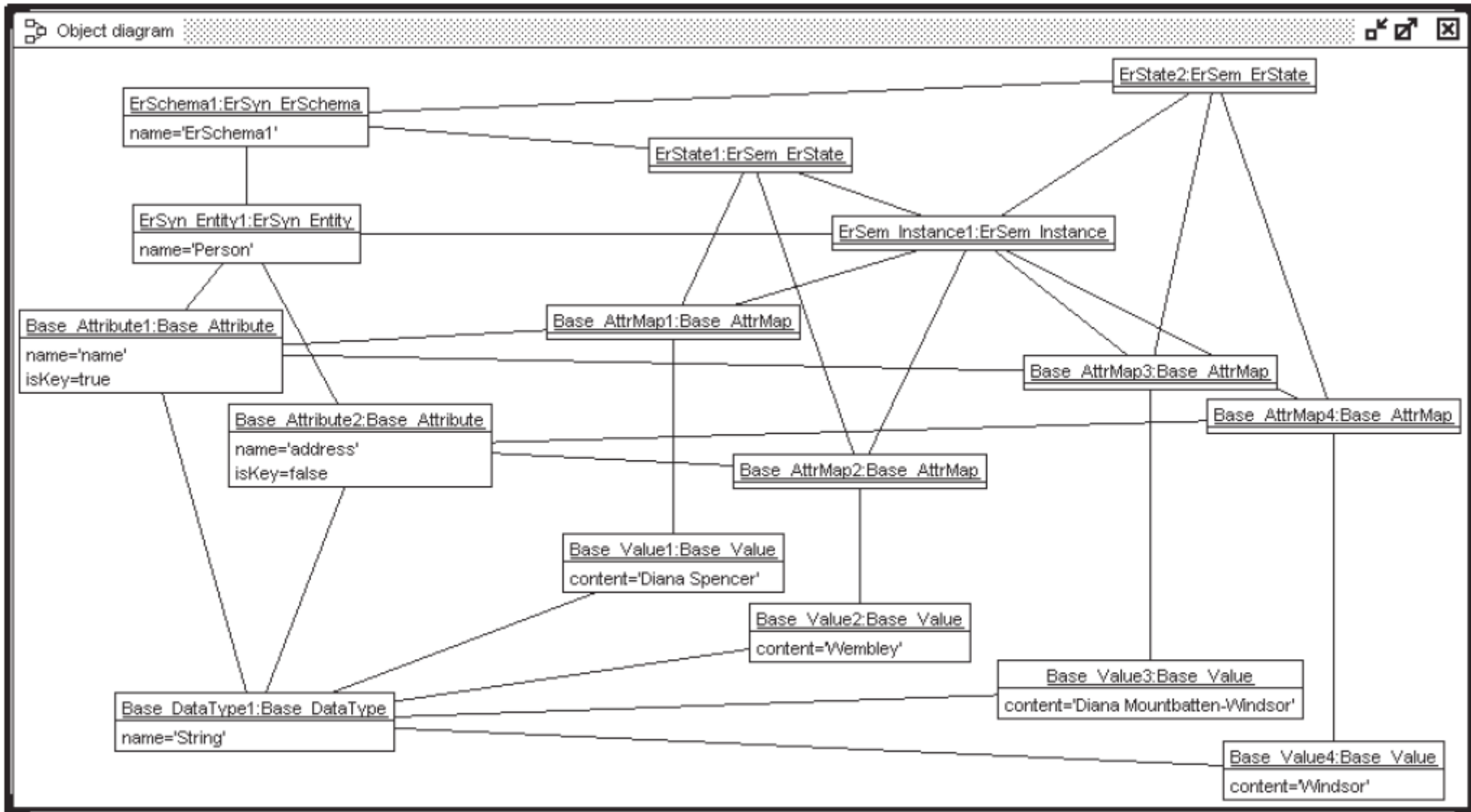
Used Association kinds: Composition, Aggregation, Functional association with arrow, Functional association (from 'Semantics' part to 'Syntax' part) with dashed arrow





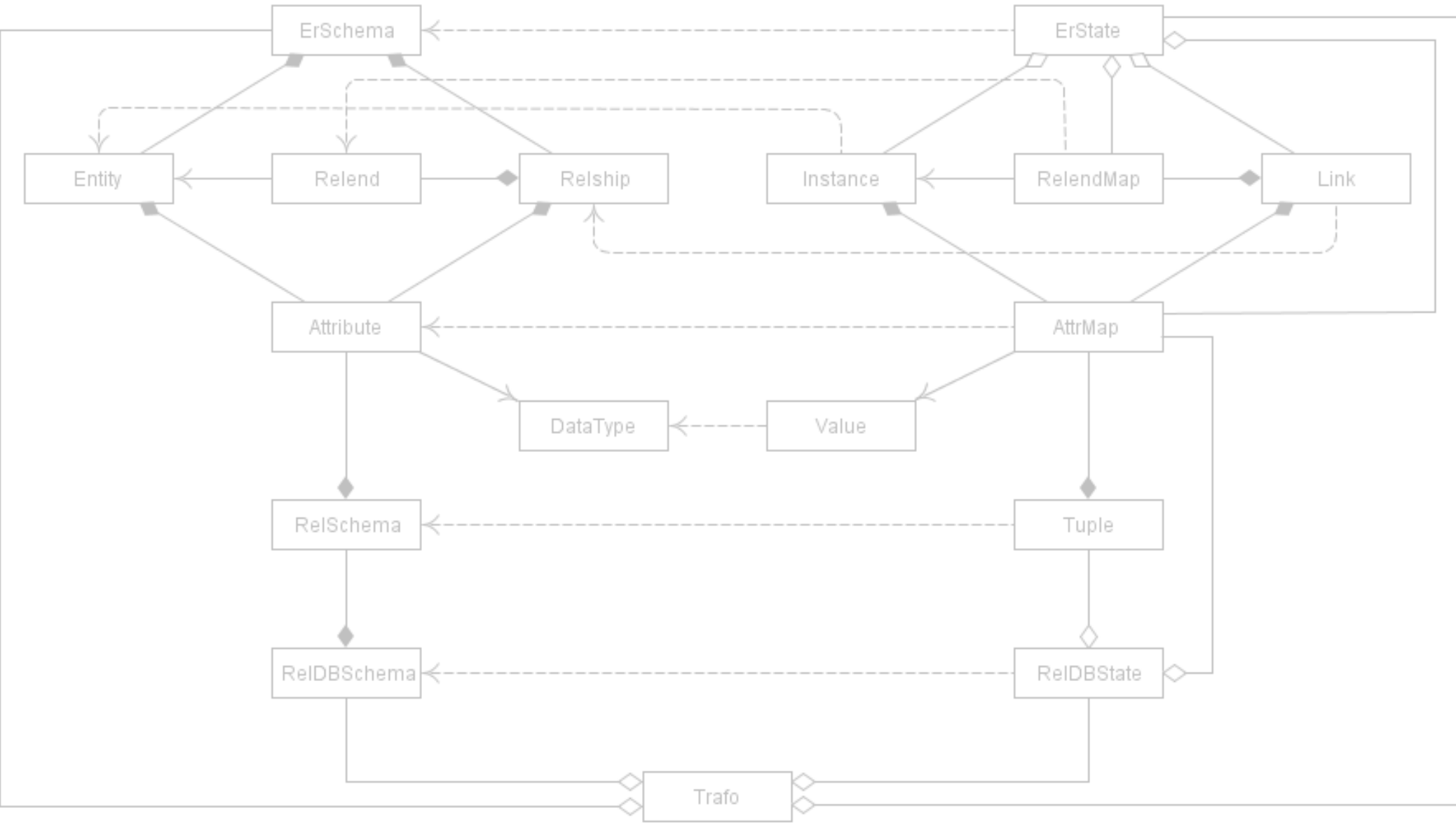
Object Diagram for *Diana moves from Wembley to Windsor*

Instance object occurs in 2 ErState objects,
AttrMap objects in 1 or 2 ErState objects

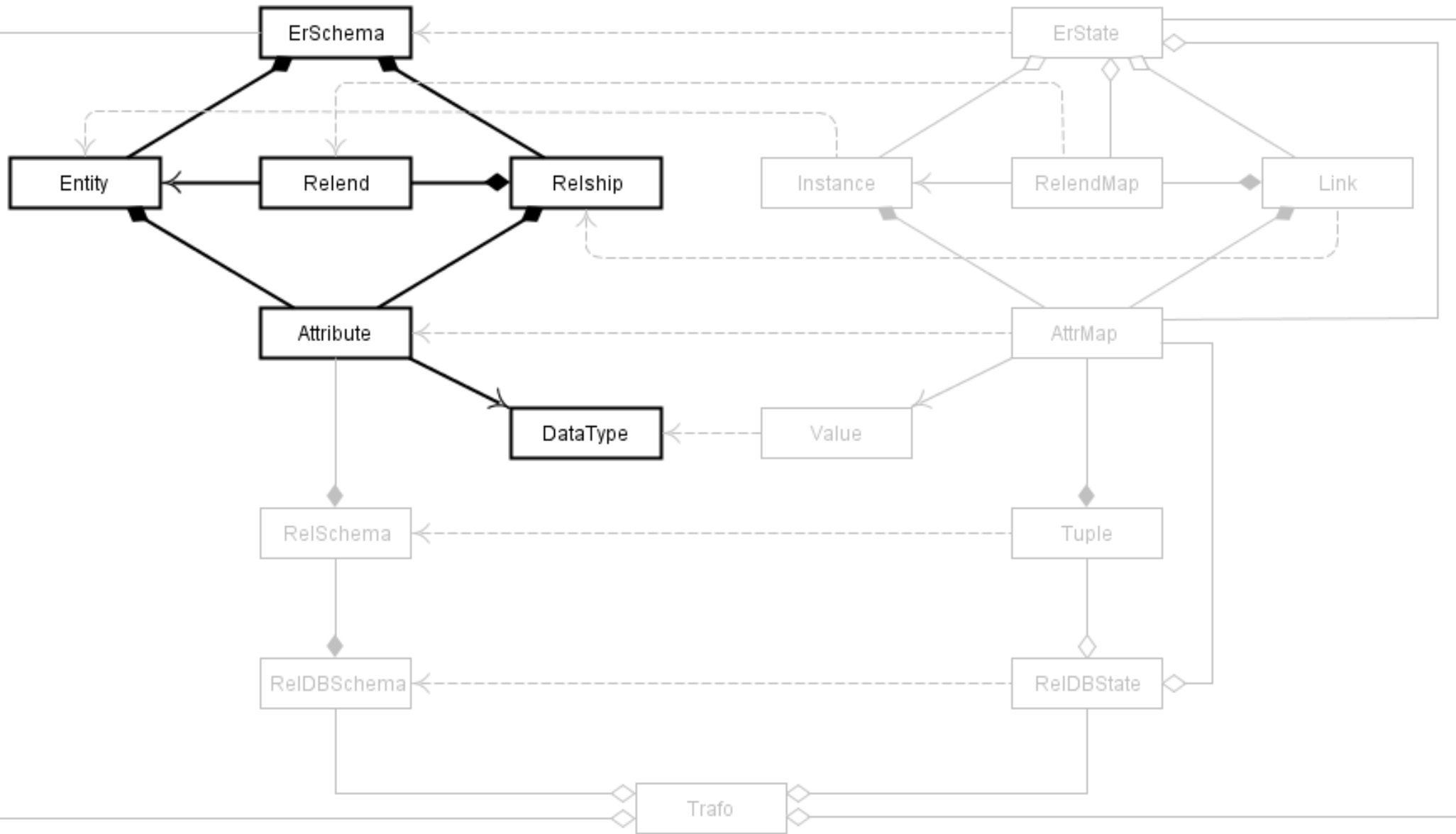


Object Diagram for *Diana moves and marries*

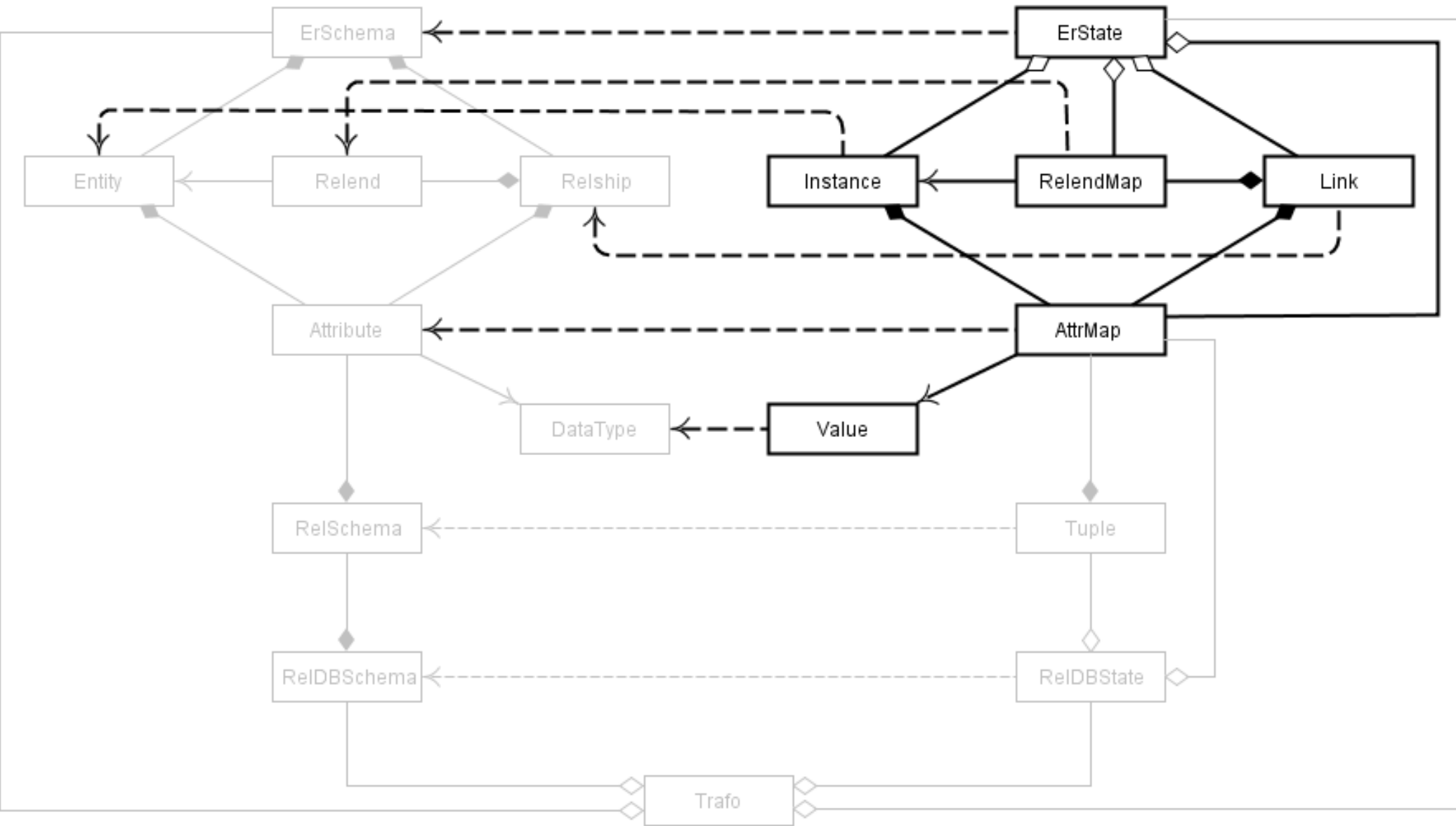
2 ErState objects only have the Instance object in common,
rest is disjoint



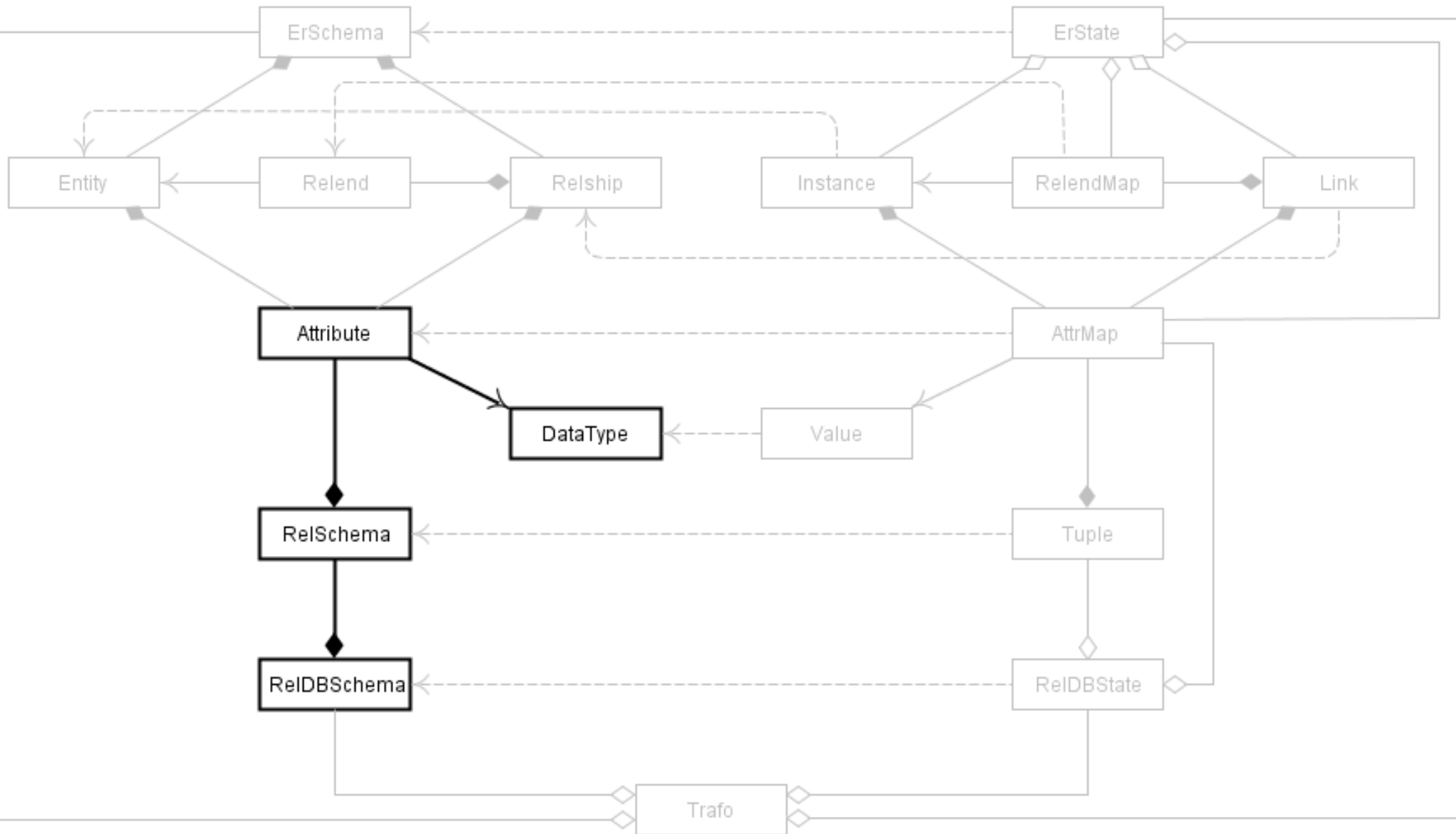
ER Syntax



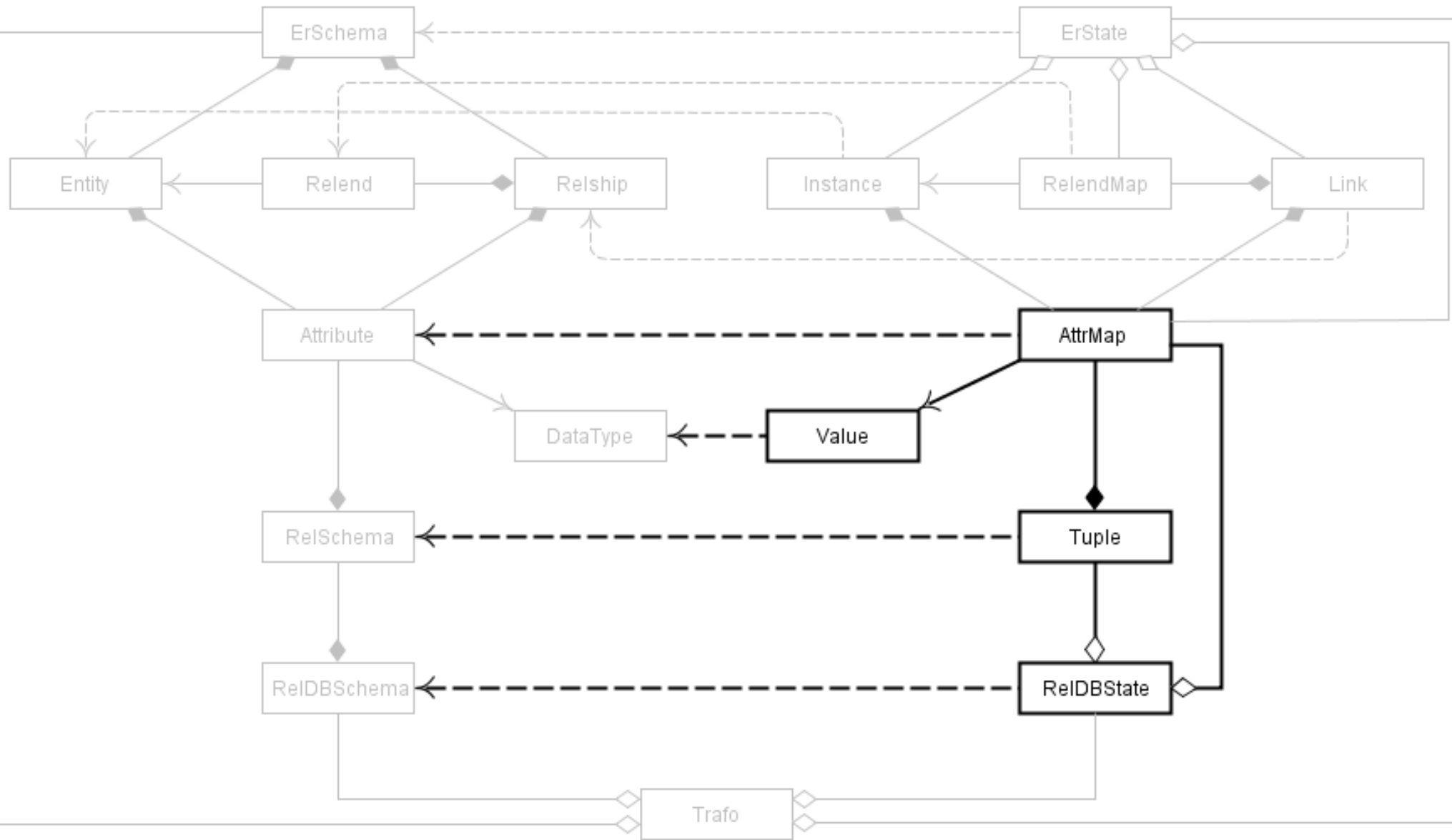
ER Semantics



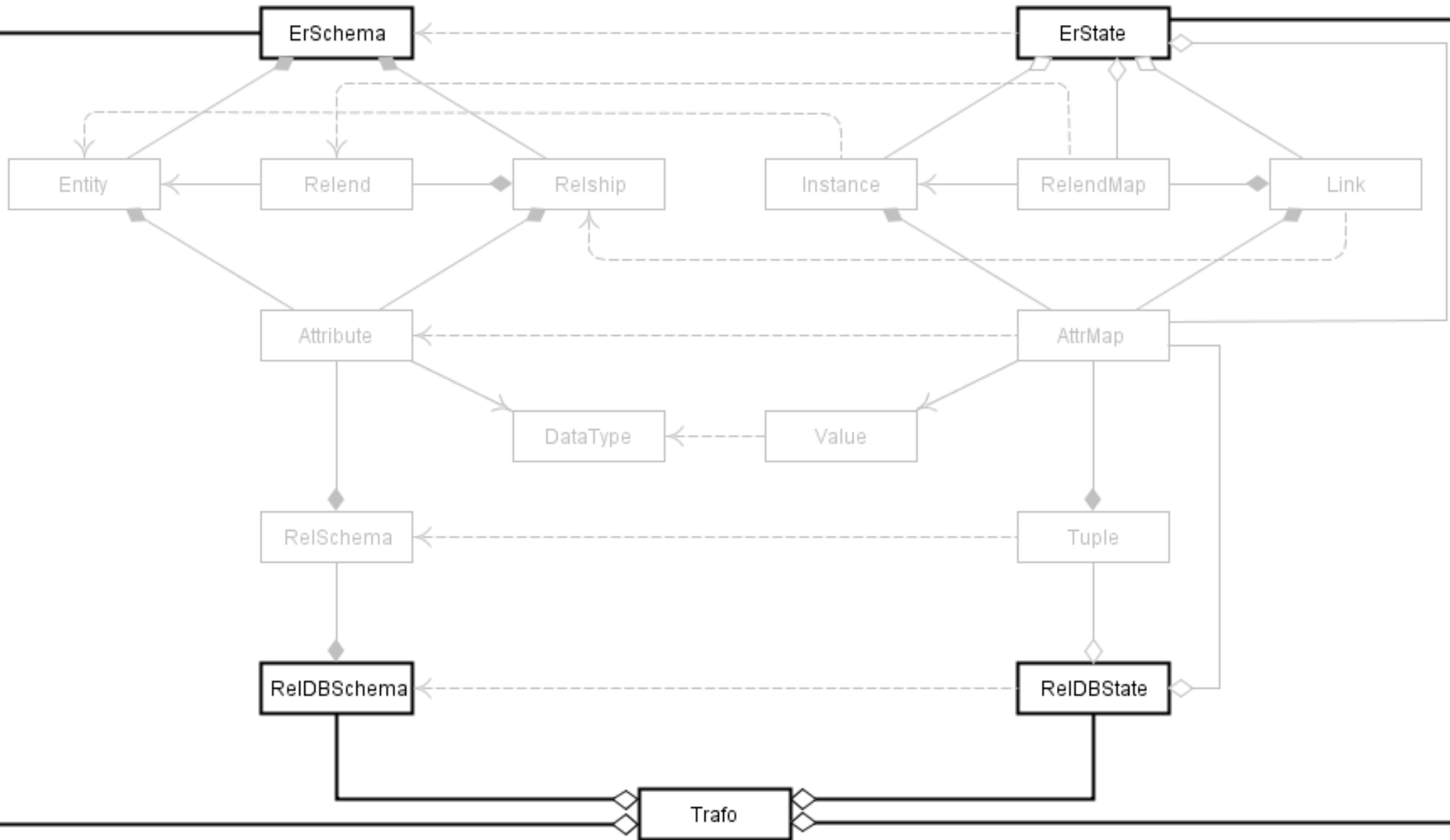
Relational Syntax



Relational Semantics



Transformation



CONSTRAINTS (EXCERPTS)

BASE

-- Naming restriction: Different DataTypes have different names

```
context self:Base_DataType inv uniqueDataTypeNames:
```

```
  Base_DataType.allInstances->
```

```
    forAll(self2 | self.name=self2.name implies self=self2)
```

CONSTRAINTS (EXCERPTS)

ER SYNTAX

-- Different ErSchemas have different names

```
context self:ErSyn_ErSchema inv uniqueErSchemaNames:
  ErSyn_ErSchema.allInstances->
    forAll(self2 | self.name=self2.name implies self=self2)
```

-- Within one ErSchema, different Entities have different names

```
context self:ErSyn_ErSchema inv uniqueEntityNamesWithinErSchema:
  self.entity->forAll(e1,e2 | e1.name=e2.name implies e1=e2)
```

-- The set of key attributes of an Entity is not empty

```
context self:ErSyn_Entity inv entityKeyNotEmpty:
  self.key()->notEmpty
```

-- The set of key attributes of a Relship is empty

```
context self:ErSyn_Relship inv relshipKeyEmpty:
  self.attribute->select(a | a.isKey)->isEmpty
```

CONSTRAINTS (EXCERPTS)

ER SEMANTICS

-- Two different Instances of one Entity can be distinguished in every
-- ErState where both Instances occur by a key Attribute of the Entity

```
context self:ErSem_Instance inv keyMapUnique:
  ErSem_Instance.allInstances->forall(self2 |
    self<>self2 and self.entity=self2.entity
    implies
    self.erState->intersection(self2.erState)->forall(s |
      self.entity.key()->exists(ka |
        self.applyAttr(s,ka)<>self2.applyAttr(s,ka))))
```

CONSTRAINTS (EXCERPTS)

REL SYNTAX

-- The set of key Attributes of a RelSchema is not empty

```
context self:RelSyn_RelSchema inv relSchemaKeyNotEmpty:  
  self.key()->notEmpty
```

CONSTRAINTS (EXCERPTS)

REL SEMANTICS

```
-- Two different Tuples of one RelSchema can be distinguished in every
-- RelDBState where both Tuples occur by a key Attribute of the
-- RelSchema
```

```
context self:RelSem_Tuple inv keyMapUnique:
  RelSem_Tuple.allInstances->forall(self2 |
    self<>self2 and self.relSchema=self2.relSchema
  implies
    self.relDBState->intersection(self2.relDBState)->forall(s |
      self.relSchema.key()->exists(ka |
        self.applyAttr(s,ka)<>self2.applyAttr(s,ka))))
```

CONSTRAINTS (EXCERPTS)

TRANSFORMATION

```
-- For every Relship in the ErSchema there is a RelSchema having the
-- same name, Relends representing the arms of the relationship, and
-- Attributes with the same properties, i.e., name, DataType, and key
-- property
```

```
context self:Er2Rel_Trans inv forRelshipExistsOneRelSchema:
  self.erSchema.relship->forall(rs |
    self.relDBSchema.relSchema->one(rl |
      rs.name=rl.name and
      rs.relend->forall(re | re.entity.key()->forall(rek |
        rl.attribute->one(ra |
          re.name.concat('_').concat(rek.name)=ra.name and
          rek.dataType=ra.dataType and ra.isKey))) and
      rs.attribute->forall(rsa |
        rl.attribute->one(ra |
          rsa.name=ra.name and rsa.dataType=ra.dataType and
          ra.isKey=false))))
```


Thanks for your attention!