ORM Normative Abstract Syntax and Semantics: non-normative glossary

ORM.net Proposed Recommendation

**Version: Public BETA 3 -** 19 December 2024  
**Editors:** Enrico Franconi, Terry Halpin

**Abstract**

Object-Role Modeling (ORM) is a rigorous approach to modeling and querying at the conceptual level the information semantics of arbitrary domains. This glossary document lists key terms and symbols used in ORM, and briefly explains their meaning by means of examples. It shows examples of the main graphical conceptual model constructs - namely declarations, constraints, and derivation rules - together with their corresponding abstract syntactic expressions, and their semantics specified as closed first-order logic formulas. This non-normative document makes use of the definitions specified in the normative document defining the abstract syntax and formal semantics of ORM conceptual models. The semantics of an ORM conceptual model is defined by transforming the model to first-order logic axioms, whose finite models denote the legal abstract information structures of the conceptual specification.

**Status of this Document**

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of the revisions of this technical report can be found in the ORM.net Technical Recommendations index at <https://gitlab.com/orm-syntax-and-semantics/orm-syntax-and-semantics-docs.git>.

This document is part of the ORM document suite. It summarizes the abstract syntax of the main graphical symbols used in ORM by means of examples. The companion document “ORM Abstract Syntax and Semantics: normative specifications” formally defines the core ORM concepts. Both documents of the ORM document suite can be found at   
<https://gitlab.com/orm-syntax-and-semantics/orm-syntax-and-semantics-docs.git>.

This document is published on ORM.net as a Proposed Recommendation. If you wish to make comments regarding this document, please send them to <[orm-semantics@googlegroups.com](mailto:orm-semantics@googlegroups.com)>, after having registered at <<https://groups.google.com/group/orm-semantics>>. All comments are welcome.

Once this document becomes an ORM.net Recommendation, it will be a stable document and may be used as reference material or cited from other documents. ORM.net's role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability of data models based on ORM or other fact-based modeling approaches.

**Change History**

* Public BETA 3: fixed typos.

| ***Construct and Examples*** | ***Normative Abstract Syntax of Examples*** | ***Normative Semantics of Examples*** |
| --- | --- | --- |
| **Signature: Entity type name** | *Signature:*  *Entity Type name*: Country |  |
|  |  |  |
| **Signature: Value type name** | *Signature:*  *Value Type name*: Country Code  (Case-sensitive. Spaces are allowed. Hyphenated words are allowed) |  |
|  |  |  |
| **Signature: Value Syntax** | *Signature:*  **ValueSyntax**(CountryCode /[A-Z][A-Z]/)  **ValueSyntax**(Population Natural)  (Instances of any Object type having a ValueSyntax may be identified by a value matching that syntax) | Values of this ObjectType must match the defined syntax.  Syntax is defined by matching a Regular Expression or the syntax of another ObjectType which has a (non-recursively defined) syntax.  Natural, Integer, Real, String, Date, Time and RegularExpression are predefined. |
|  |  |  |
| **Signature: Predicate name** | *Signature:*  *Unary predicate name*: smokes, ? has cancer  *Binary predicate names*: was born in,  ? speaks ? very well  reports to, employs  *Ternary predicate name*: ? played ? for ?  *Quaternary predicate name*: ? in ? on ? ate ?  (Case-sensitive. Spaces surrounding ? are ignored. Unary predicate may elide an initial ? Binary predicate may elide implicit ? at both ends. Words may be hyphenated, but hyphens adjacent to a word and a space or question mark signals hyphen-binding to a verbaliser)  *Alternate predicate name:*  **AlternatePredicate**(reports to, manages (2 1)) |  |
|  |  |  |
| **Signature: Role name** | *Role names:*  **RoleNaming**(smokes.1, smoker)  **RoleNaming**(employs.1, employer)  **RoleNaming**(employs.2, employee)  **RoleNaming**(drives.1, driver)  *Signature:*  *Role identifier* *for the unary predicate* smokes*:* smokes.1  *Role identifiers for the binary predicate* employs*:* employs.1, employs.employee |  |
|  |  |  |
| **Unary fact type** | **FactType**(smokes (Person)) |  |
|  |  |  |
| **Binary fact type** | **FactType**(was born in (Person, Country))  **FactType**(employs (Company, Person))  **FactType**(made (Company, Product))  **FactType**(? drives ? (Person, Car))  **FactType**(reportsTo (Person, Person)) |  |
|  |  |  |
| **Ternary fact type** | **FactType**(?played?for?   (Person, Sport, Country))  **FactType**(? introduced ? to ?   (Person, Person, Person))  **FactType**(?ate?on? (Cat, Food, Date)) |  |
|  |  |  |
| **Quaternary fact type** | **FactType**(?in?on?ate? (Person, City, Date, Food)) |  |
|  |  |  |
| **Objectification** | **FactType**(enrolledIn (Student, Course))  **Objectifies**(Enrolment, enrolledIn)  **FactType**(resultedIn (Enrolment, Grade)) |  |
|  |  |  |
|  |  |  |
| **UCs on a binary fact type** | **Unique**(isOf.1)  **Unique**(wasBornIn.1)  **Unique**(speaks.1, speaks.2)  **Unique**(isPresidentOf.1)  **Unique**(isPresidentOf.2) |  |
|  |  |  |
| **UCs on ternaries** | **Unique**(?got?in?.1, ?got?in?.3)  **RoleNaming**(?got?in?.2, placing)  **Unique**(?got?in?.placing, ?got?in?.3)  **Unique**(?played?for?.1, ?played?for?.2,   ?played?for?.3) |  |
|  |  |  |
| **Simple mandatory role constraint** | **Mandatory**(Person, wasBornIn.1) |  |
|  |  |  |
| **Inclusive-or constraint** | **Mandatory**(Visitor,  hasPassport.1, hasDriverLicence.1) |  |
|  |  |  |
| **Preferred internal UC** | **Identification**(Country, has.1 (has.2)) | *well-founded*(has) |
|  |  |  |
| **External UC** | **ExternalIdentification**(State   (hasStateCode.2, isIn.2))  **ExternalUnique**(hasStateName.2, isIn.2) | *well-founded*(hasStateCode ∪ isIn) |
|  |  |  |
| **Object type value constraint** | **ValuesOf**(GenderCode (M F))  … |  |
|  |  |  |
| **Role value constraint** | **ValuesOf**(has.2 (0 … 140)) |  |
|  |  |  |
| **Subset constraint** | **Subset**((smokes.1, isCancerProne.1))  **Subset**((?for?obtained?.1, enrolledIn.1)  (?for?obtained?.2, enrolledIn.2)) |  |
|  |  |  |
| **Join subset constraint** | **JoinPath**(P (speaks.1, speaks.2)  (isOftenUsedIn.1, isOftenUsedIn.2))  **Subset**((servesIn.1, P.1)(servesIn.2, P.2)) |  |
|  |  |  |
| **Exclusion constraint** | **Exclusive**((isWidowed.1, isMarried.1))  **Exclusive**((reviewed.1, authored.1)  (reviewed.2, authored.2)) |  |
|  |  |  |
|  |  |  |
| **Equality constraint** | **Equal**((has systolic-.1, has diasystolic-.1)) |  |
|  |  |  |
|  |  |  |
| **Subtyping** | **Subtype**(Lecturer, Employee)  **Subtype**(Employee, Person)  **Subtype**(Student, Person)  **Subtype**(StudentEmployee, Student)  **Subtype**(StudentEmployee, Employee) |  |
|  |  |  |
| **Subtyping constraints** | **ExclusiveSubtypes**((Dog, Cat) Animal)  **ExhaustiveSubtypes**((Player, Coach) TeamMember)  **ExclusiveSubtypes**((MalePerson, FemalePerson)   Person) **ExhaustiveSubtypes**((MalePerson, FemalePerson)   Person) |  |
|  |  |  |
|  |  |  |
| **Internal frequency constraint** | **Frequency**(isAMemberOf.2 (12))  **Frequency**(isOn.2 (4, 7))  **Frequency**(reviews.1 (..5))  **Frequency**(reviews.2 (2..))  **Frequency**(?in?hadStaffOf?in?.1,  ?in?hadStaffOf?in?.2 (2)) |  |
|  |  |  |
| **External frequency constraint** | **ExternalFrequency**(isBy.2, isIn.2 (..2)) |  |
|  |  |  |
| **Value-comparison constraint** | **≥**(endedOn.2, startedOn.2) |  |
|  |  |  |
| **Object cardinality constraint** | **TypeCardinality**(President (0, 1)) |  |
|  |  |  |
| **Role cardinality constraint** | **RoleCardinality**(isThePresidentOf (0, 1)) |  |
|  |  |  |
| **Ring constraints** | **LocallyReflexive**(P.1, P.2)  *etc.* | *etc.* |

|  |  |  |
| --- | --- | --- |
| **Derivation Rules** | **SubTypeRule**(Smoker, (Person ∧ smokes)) |  |
|  | **SubTypeSemiRule**(Taxpayer, Person)  **SubTypeSemiRule**(Taxpayer, Company) |  |
|  | **SubTypeRule**(Resident,   (Person ∧ (isAResidentCitizen ∨   isAResidentAlien))  **SubTypeRule**(SelfTransporter,   (Person ∧   ((drives.1 ➤ [drives.2 ⋈ Car]) ∨   (rides.1 ➤ [rides.2 ⋈ Motorcycle])))) |  |
|  | **SubTypeRule**(NonSmoker, (Person ∖ smokes))  **SubTypeRule**(NonDriver,   (Person ∖ (drives.1 ➤ [drives.2 ⋈ Car]))  **SubTypeRule**(TeeTotaller,   (Person ∖   (drinks.1 ➤ [drinks.2 ⋈   (Beverage ∧ isAlcoholic)])) |  |
|  | **FactTypeRule**(isATypicalSportsPerson,   (Person ∧ ?played?for?.1 ➤   [?played?for?.2 ⋈ (Sport ∧ isPopular)]   [?played?for?.3 ⋈ (Country ∧ isLarge)])) |  |
|  | **FactTypeRule**(livesInCountry,   (Person ∧ livesInState.1 ➤   [livesInState.2 ⋈ (State ∧ isIn.1 ➤   [isIn.2 ⋈ (Country ∧ ?x)])]),   (Country ∧ ?x)) |  |
|  | **FactTypeRule**(canFullyCommunicateIn,   (Person ∧   (canSpeak.1 ➤ [canSpeak.2 ⋈ (Language ∧ ?x)]) ∧  (canWrite.1 ➤ [canwrite.2 ⋈ (Language ∧ ?x)])),  (Language ∧ ?x))  **FactTypeRule**(canCommunicateIn,  (Person ∧   ((canSpeak.1 ➤ [canSpeak.2 ⋈ (Language ∧ ?x)]) ∨  (canWrite.1 ➤ [canwrite.2 ⋈ (Language ∧ ?x)]))),  (Language ∧ ?x)) | () |
|  | **FactTypeRule**(soldIn,  (CarModel ∧ ?x),  (Region ∧   (livesIn.2 ➤ [livesIn.1 ⋈ Customer ∧  (bought.1 ➤ [bought.2 ⋈ Car ∧  (isOf.1 ➤ [isOf.2 ⋈ (CarModel ∧ ?x)])])])))  **FactTypeRule**(?in?bought?,   (Customer ∧   (livesIn.1 ➤ [livesIn.2 ⋈ (Region ∧ ?x)]) ∧  (bought.1 ➤ [bought.2 ⋈ (Car ∧  (isOf.1 ➤ [isOf.2 ⋈ CarModel ∧ ?y]))])),  (Region ∧ ?x),  (CarModel ∧ ?y)) |  |