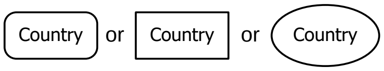
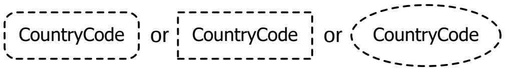
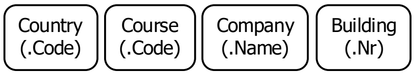
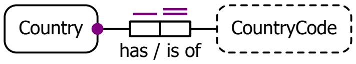
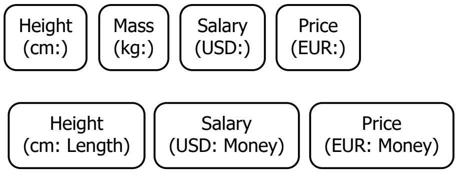
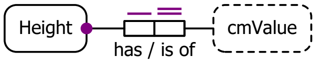
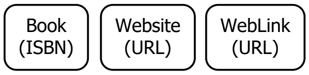
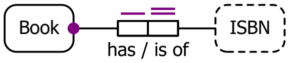
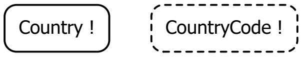
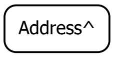
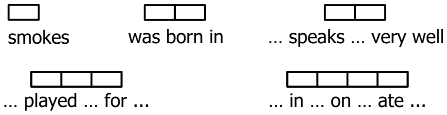
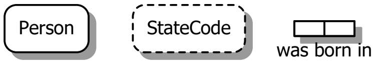
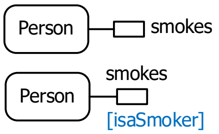
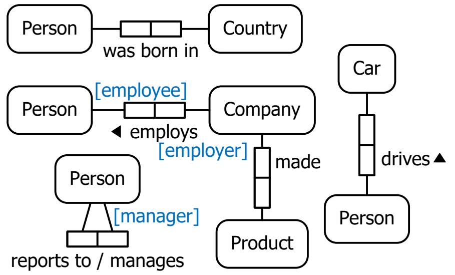
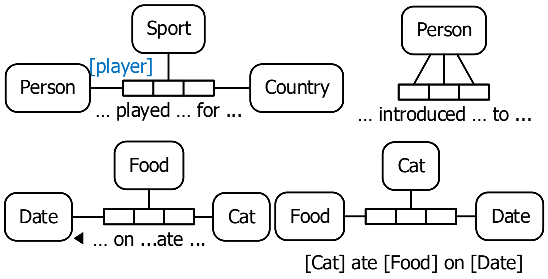
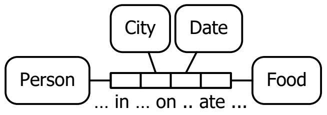
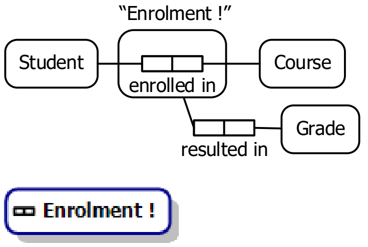
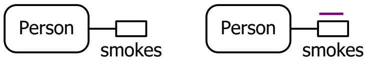
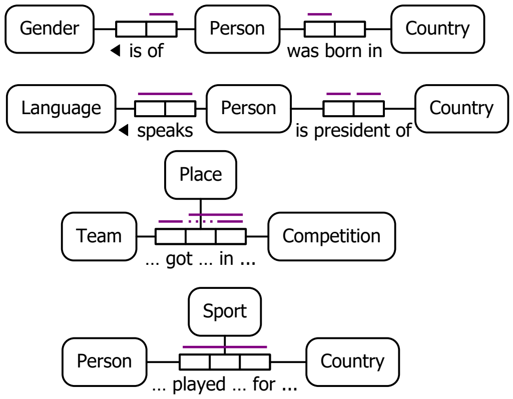
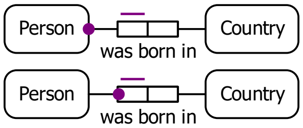
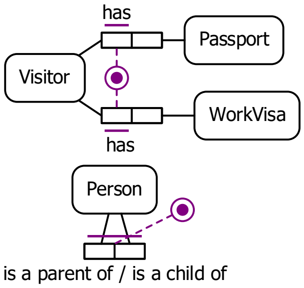
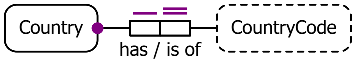
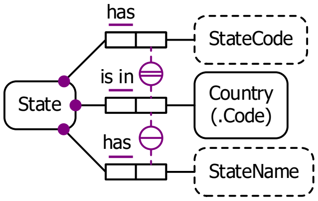
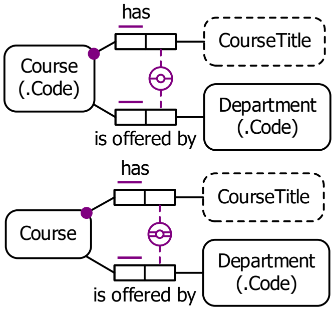
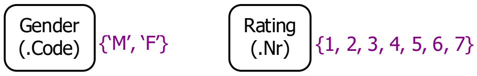
ORM 2 Graphical Notation

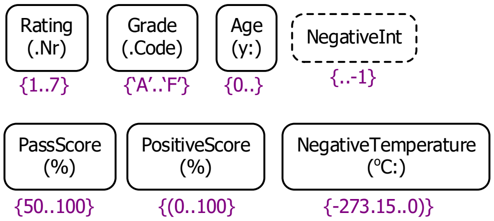
Terry Halpin

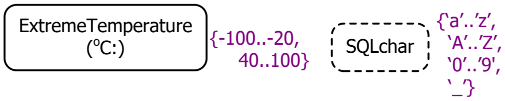
Construct	*Examples*	*Description/Notes*
Entity Type		Named soft rectangle, named hard rectangle, or named ellipse. The soft rectangle shape is the default.
Value Type		Named, dashed, soft rectangle (or hard rectangle or ellipse).
Entity type with popular reference mode	 Country (.Code) Course (.Code) Company (.Name) Building (.Nr)	Abbreviation for injective reference relationship to value type, e.g. 
Entity type with unit-based reference mode	 Height (cm:) Mass (kg:) Salary (USD:) Price (EUR:) Height (cm: Length) Salary (USD: Money) Price (EUR: Money)	Abbreviation for reference type, e.g.  Optionally, the unit type may be displayed. (as shown opposite).
Entity type with general reference mode		Abbreviation for reference type, e.g. 
Independent Object Type		Instances of the type may exist, without playing any elementary fact roles
External Object Type		This notation is tentative, and is not supported by the NORMA tool.
Predicate (unary, binary, ternary, etc.)	 smokes was born in … speaks … very well … played … for … … in … on … ate …	Ordered set of 1 or more role boxes with at least one predicate reading. If shown, object placeholders are denoted by "…". If placeholders are not shown, unaries use prefix notation and binaries use infix notation. Ternaries etc. use mixfix notation.
Duplicate type or predicate shape		If an object type or predicate shape is displayed more than once (on the same page or different pages) it is shadowed.
Unary fact type		Attaching a role box to an object type shape means that only instances of that object type may play that role. A role name may be added in square brackets.

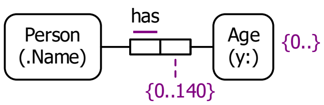
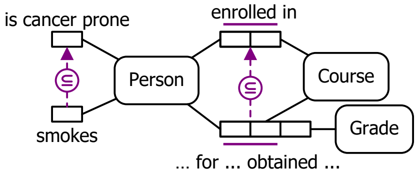
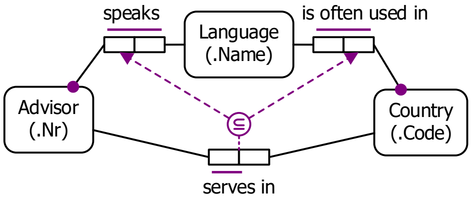
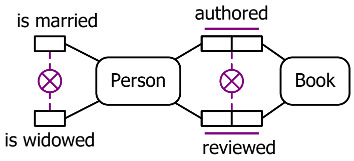
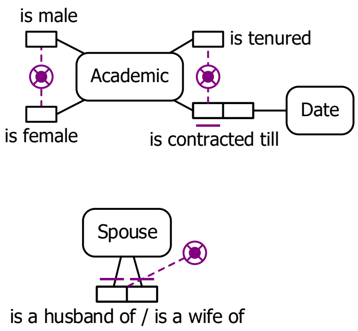
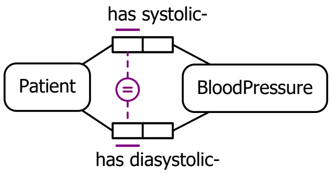
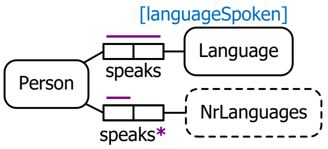
Construct	*Examples*	*Description/Notes*
Binary fact type	 Person was born in Country Car Person [employee] Company ◄ employs [employer] made drives ▲ Person [manager] reports to / manages Product Person	By default, predicate readings (binary or longer) are read left-to-right or top-to-bottom. An arrow-tip is used to display a different reading direction. Role names may be displayed in square brackets beside their role. Forward and inverse readings for binaries may be shown together, separated by /”.
Ternary fact type	 Sport Person [player] Country … played … for … Person … introduced … to … Food Date Cat ◄ … on …ate … Cat Food Date [Cat] ate [Food] on [Date]	Role names may be added in square brackets. Arrow-tips are used to reverse the default left-right or top-down reading order. Reading orders other than forward and reverse are shown using named placeholders.
Quaternary fact type		The above notes for the ternary case apply here also. Fact types of higher arity (number of roles) are also permitted.
Objectification (a.k.a. nesting)	 "Enrolment !” Student enrolled in Course resulted in Grade Enrolment !	The enrolment* fact type is objectified as an entity type whose instances can play roles. In this example, the objectification type is independent, so we can know about an enrolment before the grade is obtained. Objectification object types may also be displayed without their defining components, using an object type shape containing a small predicate shape, as shown. *Australian spelling used here.
Internal uniqueness constraint on unaries		These examples are equivalent. By default, fact types are assumed to be populated with sets of facts (not bags of facts), so no whole fact may be duplicated.
Internal UC on binaries **Internal UC on ternaries.** For *n*-aries (*n* > 1) each UC must span at least *n*-1 roles	 Gender ◄ is of Person was born in Country Language ◄ speaks Person is president of Country Place Team … got … in … Competition Sport Person … played … for … Country	The examples show the 4 possible patterns: 1:*n* (one-to-many); *n*:1 (many-to-one); *m*:*n* (many-to-many); 1:1 (one-to-one) The first example has two, 2-role UCs: the top UC forbids ties; the other UC ensures that each team gets only place per competition (a dotted line excludes its role from the UC). The second example has a spanning UC (many-to-many-to-many).

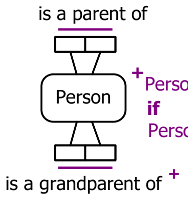
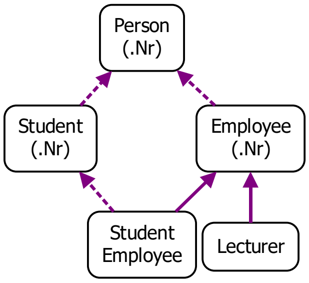
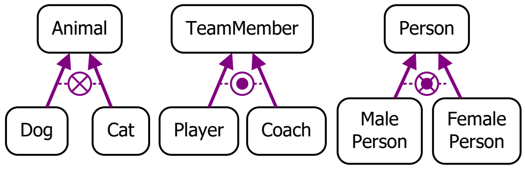
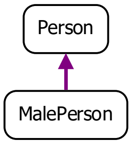
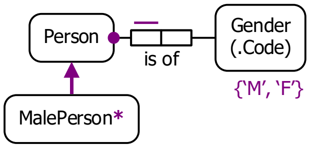
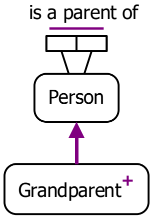
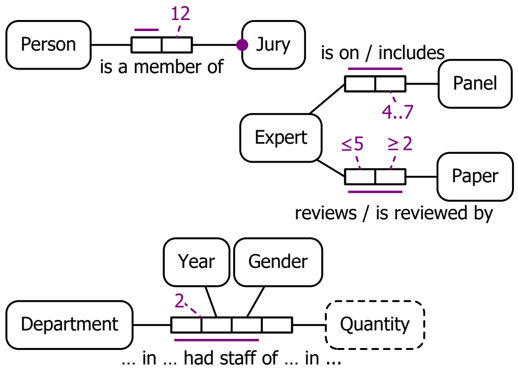
Construct	*Examples*	*Description/Notes*
Simple mandatory role constraint	 Person was born in Country Person was born in Country	The example constraint means that each person was born in some country. The mandatory role dot may be placed at either end of the role connector.
Inclusive-or constraint (disjunctive mandatory role constraint)	 has Passport Visitor WorkVisa has Person is a parent of / is a child of	The constraint is displayed as a circled dot connected to the constrained roles (or the junction of adjacent roles hosted by the same object type). The first constraint means that each visitor referenced in the model must have a passport or a work visa (or both). The second example means that each person is a parent of a person or a child of a person (or both).
Preferred internal UC		A double bar on a UC indicates it underlies the preferred reference scheme.
External UC (inner-join semantics)	 has StateCode is in State Country (.Code) has StateName	A double-bar indicates that the constrained roles provide the preferred reference for the object type at the other end. Here, each state is primarily identified by combining its country and state code. Each combination of country and state name also applies to only one state.
External UC (outer-join semantics)	 has CourseTitle Course (.Code) Department (.Code) is offered by has CourseTitle Course Department (.Code) is offered by	An inner "o" through the uniqueness bars indicates that the external UC has outer join semantics, with the added proviso that nulls produced in the outer join are treated as actual values. A double uniqueness bar indicates the UC is used in the preferred reference scheme. The inner "o" notation is not yet supported by the NORMA tool.

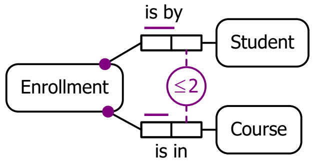
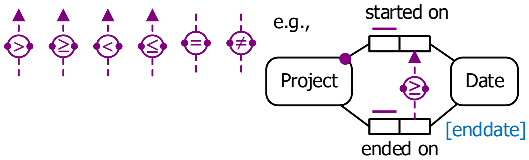
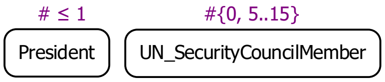
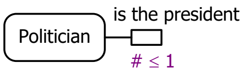
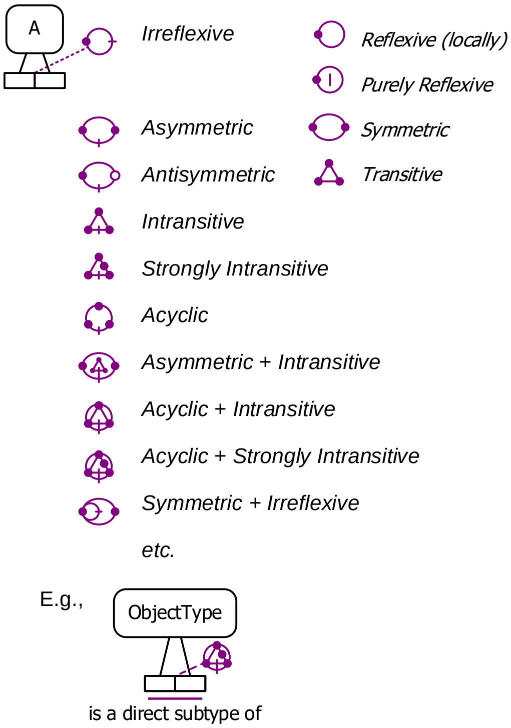
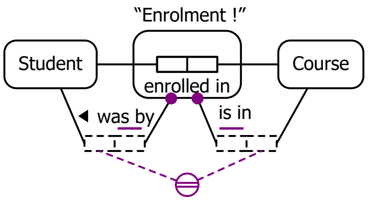
Object Type Value Constraint	 Gender (.Code)
	{‘M’, ‘F’}
	Rating (.Nr)
	{1, 2, 3, 4, 5, 6, 7}
		Enumerations

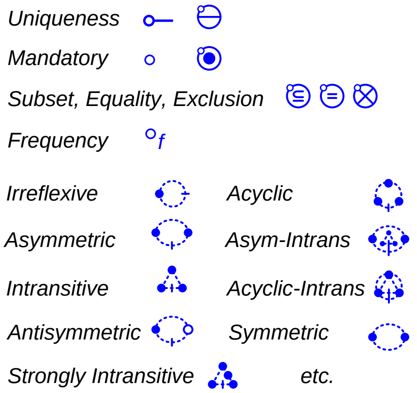
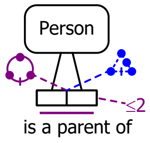
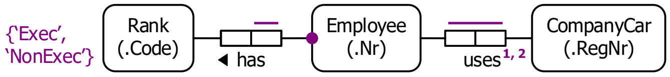
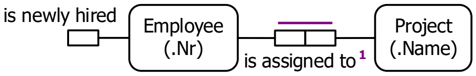
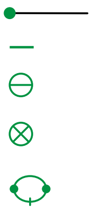
	 Rating (.Nr)
	{1..7}
	Grade (.Code)
	{‘A’..‘F’}
	Age (y:)
	{0..}
	NegativeInt
	{..-1}
	PassScore (%)
	{50..100}
	PositiveScore (%)
	{(0..100}
	NegativeTemperature (ºC:)
	{-273.15..0)}
		Ranges are inclusive of end values by default. Round brackets are used to exclude an end value. Square brackets may be added to explicitly declare inclusion, e.g. the constraint on PositiveScore may also be specified as {(0..100]}.

	 ExtremeTemperature (ºC:)
	{-100..-20, 40..100}
	SQLchar
	{‘a’..‘z’, ‘A’..‘Z’, ‘0’..‘9’, ‘_’}
		Multiple combinations are allowed.

Construct	*Examples*	*Description/Notes*
Role value constraint	 Person (.Name) has Age (y:) {0..} {0..140}	As for object type value constraints, but connected to the constrained role. Here, an age of a person must be at most 140 years.
Subset constraint	 is cancer prone smokes Person enrolled in Course Grade ... for ... obtained ...	The arrow points from the subset end to the superset end (e.g. if a person smokes then that person is cancer prone). The role sequences at both ends must be compatible. A connection to the junction of 2 roles constrains that role pair.
Join subset constraint	 speaks Language (.Name) is often used in Advisor (.Nr) Country (.Code) serves in	The constrained role pair at the superset end is projected from a role path that involves a conceptual join on Language. The constraint declares that if an advisor serves in a country then that advisor must speak a language that is often used in that country.
Exclusion constraint	 is married is widowed Person authored reviewed Book	These constraints mean that no person is both married and widowed, and no person reviewed and authored the same book. Exclusion may apply between 2 or more compatible role sequences, possibly involving joins.
Exclusive-or constraint Also known as an xor constraint	 is male Academic is female is tenured is contracted till Date Spouse is a husband of / is a wife of	An exclusive-or constraint is simply the conjunction of an inclusive-or constraint and an exclusion constraint. The first two examples say that each academic is male or female but not both, and is tenured or contracted till some date but not both. The third example says each spouse is a husband or wife of a spouse but not both. The 1:1 nature of the fact type assumes monogamy.
Equality constraint	 has systolic- Patient BloodPressure has diasystolic-	This constraint means that a patient's systolic BP is recorded if and only if his/her diastolic BP is recorded. An equality constraint may apply between 2 or more compatible role sequences, possibly involving joins.
Derived fact type, and derivation rule	 [languageSpoken] Person speaks Language speaks* NrLanguages ***For each** Person, nrLanguages = **count**(languageSpoken).	A fact type is either asserted, derived, or semiderived. A derived fact type is marked with an asterisk "*". A derivation rule is supplied. A double asterisk "**" indicates derived and stored (eager evaluation).

Construct	*Examples*	*Description/Notes*
Semiderived fact type, and derivation rule	 $^+$Person$_1$ is a grandparent of Person$_2$ **if** Person$_1$ is a parent of **some** Person$_3$ **who** is a parent of Person$_2$.	A fact type is semiderived if some of its instances may be derived, and some of its instances may be simply asserted. It is marked by “$^+$” (half an asterisk). “$^{++}$”indicates semiderived and stored (eager evaluation for derived instances).
Subtyping	 Person (.Nr) Student (.Nr) Employee (.Nr) Student Employee Lecturer	All subtypes are proper subtypes. An arrow runs from subtype to supertype. A solid arrow indicates a path to the subtype’s preferred identifier (e.g. here, student employees are primarily identified by their employee number). A dashed arrow indicates the supertype has a different preferred identifier.
Subtyping constraints	 Animal Dog Cat TeamMember Player Coach Person Male Person Female Person	⊗ indicates the subtypes are mutually exclusive. ⦿ indicates the supertype equals the union of the subtypes. The combination (xor constraint) indicates the subtypes partition the supertype (exclusive and exhaustive).
Subtype derivation status	  Person is of Gender (.Code) {‘M’, ‘F’} MalePerson* ***Each** MalePerson **is a** Person **who** is of Gender ‘M’.  $^+$**Each derived** Grandparent **is a** Person **who** is a parent of **some** Person **who** is a parent of **some** Person.	A subtype may be • asserted, • derived (denoted by “*”), • or semiderived (denoted by “$^+$”). If the subtype is asserted, it has no mark appended and has no derivation rule. If the subtype derived or semiderived, a derivation rule is supplied.
Internal frequency constraint	 Person 12 is a member of Jury is on / includes Panel 4..7 Expert ≤5 ≥ 2 Paper reviews / is reviewed by Year Gender 2 Department Quantity … in … had staff of … in …	This constrains the number of times an occurring instance of a role or role sequence may appear in each population. Here: each jury has exactly 12 members; each panel that includes an expert includes at least 4 and at most 7 experts; each expert reviews at most 5 papers; each paper that is reviewed is reviewed by at least 2 experts; and each department and year that has staff numbers recorded in the quaternary appears there twice (once for each gender).

Construct	*Examples*	*Description/Notes*
External frequency constraint		The example constraint has the following meaning. In this context, each combination of student and course relates to at most two enrolments (i.e. a student may enroll at most twice in the same course)
Value-comparison constraints	 > ≥ < ≤ = ≠ e.g., started on Project ≥ Date [enddate] ended on	The example constraint verbalizes as: **For each** Project, **existing** enddate >= startdate.
Object cardinality constraint	 # ≤ 1 #{0, 5..15} President UN_SecurityCouncilMember	The example constraints ensure that at any given time there is at most one president and either 0 or at least 5 and at most 15 members of the UN Security Council.
Role cardinality constraint		The example constraint ensures that at most one politician is the president (at any given time).
Ring constraints	 A *Irreflexive* *Reflexive (locally)* *Purely Reflexive* *Asymmetric* *Symmetric* *Antisymmetric* *Transitive* *Intransitive* *Strongly Intransitive* *Acyclic* *Asymmetric + Intransitive* *Acyclic + Intransitive* *Acyclic + Strongly Intransitive* *Symmetric + Irreflexive* *etc.* E.g., ObjectType is a direct subtype of	A ring predicate R is locally reflexive if and only if, for all x and y, xRy implies xRx. E.g. "knows" is locally but not globally reflexive. Reflexive, symmetric and transitive properties may also be enforced using semiderivation rather than by constraining asserted fact types. The example constrains the subtyping relationship in ORM to be acyclic (no cycles can be formed by a chain of subtyping connections) and strongly intransitive (no object type A can be both a direct subtype of another type B and an indirect subtype of B, where indirect subtyping means there is a chain of two or more subtyping relationships that lead from A to B). Ring constraints may be combined only if they are compatible, and one is not implied by the other. ORM tools ensure that only legal combinations are allowed.
Objectification display options: link fact types, and compact display.	 "Enrolment !" Student enrolled in Course ◄ was by is in	Internally, link fact types connect objectified associations to their component object types. By default, display of link fact types is suppressed. If displayed, dashed lines are used.

Construct	*Examples*	*Description/Notes*
Deontic constraints	 *Uniqueness* *Mandatory* *Subset, Equality, Exclusion* *Frequency* f *Irreflexive* *Acyclic* *Asymmetric* *Asym-Intrans* *Intransitive* *Acyclic-Intrans* *Antisymmetric* *Symmetric* *Strongly Intransitive* *etc.* E.g.  	Unlike alethic constraints, deontic constraint shapes are colored blue rather than violet. Most include "o" for "obligatory". Deontic ring constraints use dashed lines. In the parenthood example, the alethic frequency constraint ensures that each person has at most two parents, the alethic ring constraint ensures that parenthood is acyclic, and the deontic ring constraint makes it obligatory for parenthood to be strongly intransitive.
Textual constraints	 {'Exec', 'NonExec'} Rank (.Code) ◄ has Employee (.Nr) uses 1, 2 CompanyCar (.RegNr) [1] **Each** Employee **who** has Rank 'NonExec' uses **at most one** CompanyCar. [2] **Each** Employee **who** has Rank 'Exec' uses **some** CompanyCar.  is newly hired Employee (.Nr) is assigned to 1 Project (.Name) [1] **Each** Employee **who** is newly hired is assigned **at most one** Project.	First-order constraints with no graphic notation may be expressed textually in the FORML 2 language. These examples use footnoting to capture restricted uniqueness constraints and a restricted mandatory role constraint.
Derived constraints	E.g., 	Derived constraints are colored green. This notation is supported in the Visio ORM stencil, but is not currently supported in NORMA.