Supplementary Information: Towards Human-in-the-Loop LLM-Enabled Domain Modeling Modeling Patterns

Class vs Att	Class vs Attribute				
Decision	Use a class or attribute to	represent a concept.			
Patterns	1. For all attributes: Alternative_1: attribute, Alternative_2: class				
	2. Class with only one Ass	sociation: Alternative_1: class, Alternative_2	l: attribute		
Template	We know that each {class_	name} has an {attribute_name}. When we t	hink about {attribute_name} is it just a	simple piece of information, like a	
	line of text? Or is {attribute	e_name} made up of different parts that mig	ght be important?. For example:		
	1. For each {class_name} {	attribute_name}, the {sub_attributes} inforn	nation is saved separately, so users can j	filter {class_name} by	
	{sub_attributes}.				
	2. The {attribute_name} is	a single piece of text. This works well when	we just need to show the {attribute_nan	ne}.	
	Which option is better for y	you: <answers></answers>			
Answers	Choice 1: we need the {att	ribute_name} details separately.	Choice 2: a single line of text for {attri	bute_name} is enough.	
Actions	1. Add class {attribute_nar	ne}.	1. Add attribute {attribute_name} to {a	class_name}.	
	2. Set confidence for class	high (Threshold x 1.1).	2. Set confidence for attribute high (Th	reshold x 1.1)	
	3. Add Association from clo	ass {class_name} to class {attribute_name}	3. Set attribute type to {sub_attributes	s} type	
	4. Update the cardinality t	o "1" in the { class_name } side and "1" in	4. Remove the association		
	the { attribute_name} side		5. Remove class {attribute_name}		
	5. Set confidence for assoc	iation low (Threshold / 2)			
	6. Set confidence for cardii	nalities low (Threshold / 2)			
	7. Remove {attribute_nam	e} from garage.			
Domain: Car Service		Domain: Robot tasks			
© Garage		C Robot			
	1		String id	(C) Robot	
	has	© Garage	*		
	1	address : String	1 (C) RobotVersion	String id String version	
	C Address street : String		String model		
	city: String		String software		
	Var	iables:	Variabl	es:	
{class_name	s_name}: Garage		{class_name}: Robot		
{attribute_r	name}: Address		{attribute_name}: Version		
{sub_attribu	utes}: Street, City		{sub_attributes}: Model, Software		
Question		Question			
We know th	nat each garage has an addre	ess. When we think about address, is it just	We know that each robot has a version. When we think about version,		
a simple piece of information, like a line of text? Or is address made up of		is it just a simple piece of information, like a line of text? Or is version			
different parts that might be important?. For example:		made up of different parts that might be important?. For example:			
1. For each garage address, the street and city information is saved separately, so		1. For each robot version, the model and software information is saved			
users can filter garages by street or city.		separately, so users can filter robots by model or software.			
_		of text. This works well when we just need	2. The robot version is a single piece of text. This works well when we		
to show	the address.		just need to show the version.		
		er Options	Answer O		
	ve need the address details	Choice 2: a single line of text for address	Choice 1: we need the version	Choice 2: a single line of text for	
separately.		is enough.	details separately.	version is enough.	

Concrete C	Class vs Abstract Class				
Decision	Represent concept as abstract or does it require a concrete implementation.				
Patterns	1. For all the concrete superclasses: Alternative 1: concrete, Alternative 2: abstract				
	2. For all the abstract su	perclasses: Alternative_1: abstract, Alter	native_2: concrete		
Template	When we talk about {clas	s_name}, do we always have a clear ideo	a of a specific example, or do we somet	imes speak about it in a more general	
	way? Should we think of	{class_name} as a general concept that c	overs all possible forms, or is it always	tied to a particular type or instance?.	
	For example:				
		} can exists that is not a {sub_classes}.			
	2. A {class_name} is just a concept and cannot exist on its own. We only need specific types like {sub_classes}.				
_	Which option is better for	•			
Answers	of its types.	k of general {class_name} independent	Choice 2: the general {class_name} i. types.	s not needed, and we only need its	
Actions	1. Update class {class_na	me} to concrete	1. Update class {class_name} to abst	ract	
		s being concrete high (Threshold x 1.1)	2. Set confidence for class being abst.		
	Domain:	Car Service	Domain:	Robot tasks	
	(C) Garage	(A) Garage		(A) Robot	
	address : String	□ address : String	Robot	String id	
(String id	String id	
		<i>y</i> ×	× ×		
_	© OfficialGarage		€ CleaningRobot	ℂ CleaningRobot ℂ TransportRobot	
	warranty : Integer warranty : Integer				
	Vari	ables:	Variables:		
{class_nam	ne}: Garage		{class_name}: Robot		
{sub_classe	es}: OfficialGarage		{sub_classes}: CleaningRobot, Transp		
		estion	,	estion	
		ways have a clear idea of a specific	When we talk about robots, do we al	-	
example, or do we sometimes speak about it in a more general way? Should			example, or do we sometimes speak about it in a more general way? Should		
we think of garage as a general concept that covers all possible forms, or is it			we think of robot as a general concept that covers all possible forms, or is it		
always tied to a particular type or instance?. For example:			always tied to a particular type or instance?. For example:		
 A general garage can exists that is not an official garage. A garage is just a concept and cannot exist on its own. We only need 			A general robot can exists that is not a cleaning robot, or transport robot.		
specific types like official garages. 2. A garage is just a concept and cannot exist on its own. We only need 2. A robot is just a concept and cannot exist on its own. We or				not exist on its own. We only need	
specific types like official garages. 2. A robot is just a concept and cannot exist of its own, we of specific types like cleaning robot and transport robot.				•	
Answer Options			r Options		
Choice 1: v	we should think of	Choice 2: the general garage is not	Choice 1: we should think of	Choice 2: the general robot is not	
	rage independent of its	needed, and we only need its types.	general robot independent of its	needed, and we only need its types.	
types.	J		types.	, , , , , , , , , , , , , , , , , , , ,	

Enumeration vs Inheritance					
Decision	Represent a concepts a	s literals in enumeration or sub classes.			
Patterns			sides inheritance: Alternative_1: Inherita	nce, Alternative_2: Enumeration	
	2. All enumerations: A	lternative_1: Enumeration, Alternative_	_2: Inheritance		
Template	There are different {attı	ibute} of {class}: {enumeration_literals}	Do these {attribute} have distinct characteristics or requirements? . For		
	example:				
	· —	20 0	llowing for distinct details for each {attrib	•	
	, , , , ,	:e} is classified as either {enumeration_l	iterals}. Both {attribute} have the same d	etails, and there's no need for extra	
	information.				
	Which option is better f	-	I		
Answers	=	_literals} do have distinct	Choice 2: {enumeration_literals} doesn	't have distinct characteristics.	
	characteristics.				
Actions		numeration_literals} + {class}}	1. Create enumeration {attribute}		
		bclasses high (Threshold x 1.1)	2. Add literals {enumeration_literals}.		
	_	om created classes to {class}	3. Set confidence for enumeration and I		
		neritance relationships high (Threshold	4. Add attribute {attribute} in class {class}		
	x 1.1)	()	5. Set confidence for attribute high (Thr	•	
	5. Remove enumeration	•	6. Update class {class_name} to concret		
	6. Remove attribute {at	tribute}	7. Remove inheritance from {subclasses	} to {ciass}	
			8. Remove subclasses {subclasses}		
		Domain: Car Service Domain: Robot tasks		lobot tasks	
© Service		© Location			
	date : int	E ServiceType C Service	String id	C Location E LocationType	
MAINTENANCE		× ×	String id CORRIDOR		
(C) Maintena	anceService C RepairService	type : Servicerype	© Corridor © Room ROOM		
		ables:		bles:	
{class}: Serv			{class}: Location		
{attribute}:	· ·		{attribute }: Type		
	ion_literals}: Maintenance	e, Repair	{enumeration_literals }: Corridor, Room		
	ion}: ServiceType		{enumeration}: LocationType		
{subclasses	s}: MaintenanceService, R	•	{subclasses}: Corridor, Room		
- -1		estion	Question		
	• •	maintenance and repair. Do these	There are different types for locations: corridors and rooms. Do these		
types have distinct characteristics or requirements?. For example: 1. Maintenance and repair are considered different service types,		locations have distinct characteristics or requirements? . For example:			
	·	•••	1. Corridors and rooms are considered different locations, allowing for		
	g for distinct details for ea		distinct details for each location.		
	• •	either maintenance or repair. Both	2. Every location type is classified as either corridor or room. Both locations have the same details, and there's no need for extra information.		
types n		there's no need for extra information.		Options	
Choice 1: N	Maintenance and repair	r Options Choice 2: Maintenance and repair	Choice 1: Corridors and rooms do	Choice 2: Corridors and rooms	
	stinct characteristics.	doesn't have distinct characteristics.	have distinct characteristics.	doesn't have distinct characteristics.	
ao nave uis	stillet characteristics.	docon t have distilled characteristics.	וומיכ עוזנוווכו כוומומכופווזנוכז.	doesn't nave distinct characteristics.	

Attribute v	oute vs Inheritance			
Decision	A concept is an attribute or a sub class.			
Patterns	 Parent class has only one subclass with no attributes and no further associations besides inheritance: Alternative_1: Inheritance, Alternative_2: Attribute All boolean attributes: Alternative_1: Attribute, Alternative_2: Inheritance 		: Alternative_1: Inheritance,	
Template	When thinking about {class_name}, what makes a {subclass_name} different from a regular one? Is there a specific role or responsibility that comes with being a {subclass_name}? Are there additional factors we need to consider when dealing with {subclass_name} compared to regulars_name}?. For example: 1. A {subclass_name} is treated as a different kind of {class_name}. 2. {class_name} include a status that identifies if it as {subclass_name} or not. Which option is better for you: <answers></answers>			
Answers	Choice 1: {subclass_name	} are a special kind of garage.	Choice 2: {subclass_name} are simil	ar to other robots.
Actions	-	ass high (Threshold x 1.1) created class to {class_name} itance relationships high (Threshold x	Choice 2: {subclass_name} are similar to other robots. 1. Create attribute {"is"+{subclass_name}}) 2. Set confidence for attribute high (Threshold x 1.1) 3. Update class {class_name} to concrete 4. Remove inheritance from {subclass_name} to {class_name} 5. Remove subclass {subclass_name}	
	Domain: Car Service Domain: Robot tasks		Robot tasks	
☐ Garage ☐ address: String ☐ address: String ☐ isOfficial: Boolean ☐ isOfficial: Boolean		A Robot String id C CleaningRobot	C Robot String id Boolean isCleaningRobot	
	Varia	ables:	Variables:	
_	e }: Garage name}: OfficialGarage isOfficial		{class_name }: Robot {subclass_name}: CleaningRobot {attribute}: isCleaningRobot	
	Que	stion	Question	
When thinking about garages, what makes an official garage different from a regular one? Is there a specific role or responsibility that comes with being an official garage? Are there any additional factors we need to consider when dealing with official garages compared to regular garages?. For example: 1. An official garage is treated as a different kind of garage. 2. Garages include a status that identify if it as official garage or not.		When thinking about robots, what makes a cleaning robot different from a regular robot? Is there a specific role or responsibility that comes with being a cleaning robot? Are there any additional factors we need to consider when dealing with cleaning robots compared to a regular robot?. For example: 1. A cleaning robot is treated as a different type of robot.		
			2. Robots include a status that identify if it as cleaning robot or not.	
	Answer	Options	Answe	er Options
	Official garages are a d of garage.	Choice 2: Official garages are similar to other garages.	Choice 1: cleaning robots are a special kind of robot.	Choice 2: cleaning robot are similar to other robots.

Composition	omposition vs Association				
Decision	A relationship is strong and affects the lifecycle (composition) or is between classes with independent lifecycle (association).				
Patterns	1. Association name includes 'has' or 'part': Alternative_1: Association, Alternative_2: Composition				
Template	Does a {target_class} alwa	ays require a {source_class} to exist,	or could it exist independently and possibl	ly be linked to multiple {source_class}	
	instances? If a {source_cla	iss} is removed, would all the associo	ated {target_class} also need to be remove	ed, or could a {target_class} still make	
	sense on its own?. For exa	mple:			
	1. A {target_class} belongs	s to a {source_class} and cannot exis	t without a specific {source_class}.		
	2. A {target_class} is linked	d to a {source_class} but can exist in	dependently of it.		
	Which option is better for	you: <answers></answers>			
Answers	Choice 1: a {target_class}	is always for a specific	Choice 2: a {target_class} is independent	nt and can still exist even if the	
	{source_class}.		{source_class} is removed.		
Actions	1. Update relationship to (1. Update relationship to Association		
		omposition to high (Threshold x	2. Set confidence for the Association to l	- ,	
	1.1)		3. Update the cardinality on the {source		
	I	on the {source_class} side to 1	4. Set confidence for the cardinality to lo	ow (Threshold / 2)	
	•	ardinality to high (Threshold x 1.1)	_		
	Domain: Ca	r Service	Domain: R	obot tasks	
	C Car	© Car	A Robot String id String id		
	registrationDate : int plate : String	registrationDate : int plate : String			
E	place . String	D place : Salling	•	1*	
	Services	Services	performs	performs	
	*	*	*	*	
	© Service	© Service	€ Task	(C) Task	
	date : int type : ServiceType	date : int type : ServiceType	String id	String id	
	Variah	loa	Varia	bloo:	
(course ele	Variab	ies:	Variables:		
{source_cla {target_clas			{source_class}: Robot {target_class}: Task		
target_cras	-	ion		ction	
Does a serv	vice always require a car to e	Question Question Question Does a task always require a robot to exist, or could it exist independently			
	' '	•	Does a task always require a robot to exist, or could it exist independently and possibly be linked to multiple robot instances? If a robot is removed, would all		
independently and possibly be linked to multiple car instances? If a car is removed, would all the associated services also need to be removed, or		the associated tasks also need to be removed, or could a task still make sense			
could a service still make sense on its own?. For example:		on its own? . For example:			
1. A service belongs to a car and cannot exists without a specific car.		A task belongs to a robot and cannot exists without a specific robot.			
	2. A service is linked to a car but can exists independently of it.		2. A task is linked to a robot but can exists independently of it.		
	Answer Options			Options	
Choice 1: a	service is always for a	Choice 2: a service is	Choice 1: a task is always for a specific		
specific car.	•	independent and can still exist	robot.	can still exist even if the robot is	
		even if the car is removed.		removed.	

Upperbound Cardinality: 1 vs Many (*)					
Decision	The upperbound cardinality is	s to only one instance or multiple instanc	es.		
Patterns	1. For all the upperbound ca	rdinalities with 1: Alternative_1: Upperb	ound cardinality is 1, Alternative_2:	: Upperbound cardinality is *	
	2. For all the upperbound ca	rdinalities with Many: Alternative_1: Upp	perbound cardinality is *, Alternativ	e_2: Upperbound cardinality is 1	
Template	In the context of a {source_clo	ass}, can there be more than one {target_	class} involved, or is it always assoc	ciated with just one?. For example:	
	1. A {source_class} is linked to	a single {target_class}.			
	2. A {source_class} is linked to	multiple {target_class}.			
	OR				
	In the context of a {target_cla	rss}, can there be more than one {source_	class} involved, or is it always assoc	ciated with just one?. For example:	
	1. A {target_class} is linked to	a single {source_class}.			
	2. A {target_class} is linked to	multiple {source_class}.			
	Which option is better for you	: <answers></answers>			
Answers	Choice 1: Each {source_class}	is always linked to a single	Choice 2: A {source_class} can be	linked to multiple {target_class}.	
	{target_class}.		OR		
	OR		to "1" Choice 2: A {target_class} can be linked to multiple {source_c} to "1" 1. Update the cardinality on the {source_class} side from "1"		
	Choice 1: Each {target _class}	} } is always linked to a single			
	{source_class}.				
Actions		he {source_class} side from "1*" to "1"		source_class} side from "1" to "1*"	
	OR		OR 1. Update the cardinality on the {target_class} side from "1" to "1*"		
	1	he {target_class} side from "1*" to "1"			
	·	cardinality to high (Threshold x 1.1)	2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks		
Domain: Car Service		Domain:	Robot tasks		
	C Service		(A) Robot	(A) Robot	
	date : int	date: int	String id	String id	
	type : ServiceType	□ type : ServiceType	1	1*	
	* place	* place	performs	performs	
	1	1*	*	*	
	© Garage	© Garage	(C) Task	€ Task	
	□ address : String	address : String	String id	String id	
Variables:		les:	Var	riables:	
{source_clas			{source class}: Robot		
{target clas			{target class}: Task		
Question		Question			
In the context of a service, can there be more than one garage involved, or is it		In the context of a task, can there be more than one robot involved, or			
always associated with just one?. For example:		is it always associated with just one?. For example:			
1. A service is linked to a single garage.		1. A task is linked to a single robot.			
2. A service	is linked to multiple garages.		2. A task is linked to multiple robots.		
	Answer C)ptions	Answe	er Options	
Choice 1: E	ach service is always linked to	Choice 2: A service can be linked to	Choice 1: Each task is always	Choice 2: A task can be linked to	
a single gara	age.	multiple garages.	linked to a single robot.	multiple robots.	

Decision The lowerbound cardinality is to zero instance or one instance. Patterns 1. For all the lowerbound cardinalities with 0: Alternative 1: Lowerbound cardinality is 0. Alternative 2: Lowerbound cardinality is 0 In the context of a (target_class), does it always involve at least one {source_class}, or can it exist without any?. For example: 1. A {target_class} can exist without any {source_class}. 2. A {target_class} can exist without any {source_class}. 2. A {target_class} can exist without any {source_class}. 2. A {tource_class} must have at least one {source_class}, or can it exist without any?. For example: 1. A {source_class} can exist without any {target_class}. 2. A {source_class} can exist without any {target_class}. 3. A {source_class} can exist without any {source_class}. 4. A {source_class} can exist without any {source_class}. 5. Choice 1: A {source_class} can exist without any {source_class}. 6. Choice 1: A {source_class} can exist without any {source_class}. 7. Choice 2: A {target_class} must have at least one {source_class}. 8. Choice 1: A {source_class} can exist without any {source_class}. 9. Choice 2: A {source_class} must have at least one {source_class}. 9. Choice 2: A {source_class} must have at least one {source_class}. 9. Choice 2: A {source_class} must have at least one {source_class}. 1. Update the cardinality on the {source_class} side from "1" to "#" 2. Set the confidence for the cardinality on the {source_class} side from "#" to "1" 9. Choice 2: A {source_class} must have at least one {source_class} side from "#" to "1" 1. Update the cardinality on the {source_class} side from "#" to "1" 1. Update the cardinality on the {source_class} side from "#" to "1" 1. Update the cardinality on the {source_class} side from "#" to "1" 1. Update the cardinality on the {source_class} side from "#" to "1" 1. Update the cardinality on the {source_class} side from "#" to "#"	Lowerbound Cardinality: 0 vs 1						
### Patterns 1. For all the lowerbound cardinalities with 0: Alternative_1: Lowerbound cardinality is 0, Alternative_2: Lowerbound cardinality is 1 2. For all the lowerbound cardinalities with 1: Alternative_1: Lowerbound cardinality is 1, Alternative_2: Lowerbound cardinality is 0 Template		·	to zero instance or one instance.				
2. For all the lowerbound cardinalities with 1: Alternative_1: Lowerbound cardinality is 1, Alternative_2: Lowerbound cardinality is 0 In the context of a (target_class), does it always involve at least one {source_class}, or can it exist without any?. For example: 1. A {target_class} can exist without any {source_class}. 2. A {target_class} can exist without any {source_class}. OR In the context of a {source_class}, does it always involve at least one {target_class}, or can it exist without any?. For example: 1. A {source_class}, does it always involve at least one {target_class}, or can it exist without any?. For example: 1. A {source_class}, does it always involve at least one {target_class}, or can it exist without any?. For example: 1. A {source_class}, and exist without any {target_class}. Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {target_class} must have at least one {source_class}. OR 1. Update the cardinality on the {source_class} side from "1*" to "*" OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service Service Service Orange					Lowerbound cardinality is 1		
In the context of a (target_class), does it always involve at least one {source_class}, or can it exist without any? For example: 1. A {target_class} can exist without any {source_class}. 2. A {target_class} must have at least one {source_class}. OR In the context of a (source_class), does it always involve at least one {target_class}, or can it exist without any? For example: 1. A {source_class} can exist without any {target_class}. 2. A {source_class} must have at least one {target_class}, or can it exist without any? For example: 1. A {source_class} can exist without any {target_class}. Which option is better for you: <answers> Answers Choice 1: A {target_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. OR 1. Update the cardinality on the {source_class} side from "1*" to "*" OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service OR Carage Car</answers>	. 44441113				-		
1. A {target_class} can exist without any {source_class}. 2. A {target_class} must have at least one {source_class}. OR In the context of a {source_class}, does it always involve at least one {target_class}, or can it exist without any?. For example: 1. A {source_class} can exist without any {target_class}. 2. A {source_class} must have at least one {target_class}. Which option is better for you: Answers Answers Choice 1: A {target_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. Choice 2: A {target_class} must have at least one {source_class}. OR 1. Update the cardinality on the {source_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service Quadre int Domain: Car Service Carage	Template						
2. A {target_class} must have at least one {source_class}. OR In the context of a {source_class}, does it always involve at least one {target_class}, or can it exist without any?. For example: 1. A {source_class} can exist without any {target_class}. 2. A {source_class} must have at least one {target_class}. Which option is better for you: <answers> Answers Choice 1: A {target_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. Actions 1. Update the cardinality on the {source_class} side from "1*" to "*" OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service Domain: Robot tasks </answers>	icinpiate			aree_erass), or earlie exist without	anyror example.		
OR In the context of a {source_class}, does it always involve at least one {target_class}, or can it exist without any?. For example: 1. A {source_class} can exist without any {target_class}. 2. A {source_class} must have at least one {target_class}. Which option is better for you: <answers> Answers Choice 1: A {target_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. Choice 2: A {target_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {target_class}. Choice 2: A {source_class} must have at least one {target_class}. OR 1. Update the cardinality on the {source_class} side from "1.*" to "*" OR 1. Update the cardinality on the {target_class} side from "1.*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service Domain: Robot tasks </answers>		1 2 2 2					
In the context of a {source_class}, does it always involve at least one {target_class}, or can it exist without any?. For example: 1. A {source_class} can exist without any {target_class}. 2. A {source_class} must have at least one {target_class}. Which option is better for you: <answers> Answers Choice 1: A {target_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {source_class}. OR Choice 1: A {source_class} must have at least one {source_class}. OR Choice 2: A {target_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {target_class}. Choice 2: A {source_class} must have at least one {target_class}. 1. Update the cardinality on the {source_class} side from "1*" to "*" OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service Service Service Service OR 1. Update the cardinality to high (0.9) Domain: Robot tasks OR ARobot String id Variables: {source_class}: Service {source_class}: Service</answers>		' '	at least one (source_class).				
1. A {source_class} can exist without any {target_class}. 2. A {source_class} must have at least one {target_class}. Which option is better for you: <answers> Choice 1: A {target_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. Choice 2: A {source_class} must have at least one {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. Actions 1. Update the cardinality on the {source_class} side from "1*" to "*" OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service Service Service Service Service Source_class}: Service Variables: {source_class}: Service {source_class}: Robot</answers>							
2. A {source_class} must have at least one {target_class}. Which option is better for you: <answers> Answers Choice 1: A {target_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. Choice 2: A {target_class} must have at least one {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR 1. Update the cardinality on the {source_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks OR Choice 2: A {target_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {target_class} must have at least one {source_class}. OR Choice 2: A {target_class} must have at least one {source_class}. OR Choice 2: A {target_class} must have at least one {source_class}. OR Choice 2: A {target_class} must have at least one {source_class}. OR Choice 2: A {target_class} must have at least one {source_class}. OR OR 1. Update the cardinality on the {source_class} side from "*" to "1* 2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks Or Or Service Or Or 1. Price 2: A {target_class} must have at least one {source_class}. OR 1. Update the cardinality on the {source_class} must have at least one {source_class}. OR 1. Update the cardinality on the {source_class} must have at least one {source_class}. OR 1. Update the cardinality on the {source_class} must have at least one {source_class}. OR 1. Update the cardinality on the</answers>		_	· · ·	.gct_c.acc,, c. can it chief without	any, it or example.		
Which option is better for you: <answers> Answers Choice 1: A {target_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. OR Choice 1: A {source_class} can exist without any {target_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {target_class}. OR Choice 2: A {source_class} must have at least one {target_class}. OR Choice 2: A {source_class} must have at least one {target_class}. OR Choice 2: A {source_class} must have at least one {target_class}. OR Choice 2: A {source_class} must have at least one {target_class}. OR Choice 2: A {source_class} must have at least one {target_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}.</answers>			· · · · ·				
Answers Choice 1: A {target_class} can exist without any {source_class}. OR Choice 1: A {source_class} can exist without any {target_class}. OR Choice 1: A {source_class} can exist without any {target_class}. Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {target_class}. 1. Update the cardinality on the {source_class} side from "1*" to "*" OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service Domain: Robot tasks Variables: Variables: {source_class}: Service Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR Choice 2: A {source_class} must have at least one {source_class}. OR 1. Update the cardinality on the {source_class} side from "*" to "1* 2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks A Robot			· ·				
OR Choice 1: A {source_class} can exist without any {target_class}. Actions 1. Update the cardinality on the {source_class} side from "1*" to "*" OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service adate: int atype: ServiceType adate: int atype: ServiceType address: String Variables: {source_class}: Robot Variables: Variables: Variables: Variables: Source_class}: Robot	Answers			Choice 2: A {target class} must	have at least one {source_class}.		
Choice 1: A {source_class} can exist without any {target_class}. Actions 1. Update the cardinality on the {source_class} side from "1*" to "*" OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service address: String Variables: Choice 2: A {source_class} must have at least one {target_class}. 1. Update the cardinality on the {source_class} side from "*" to "1* OR 1. Update the cardinality on the {target_class} side from "*" to "1* 2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks A Robot String id performs Domain: Robot tasks			, (<u>_</u>				
Actions 1. Update the cardinality on the {source_class} side from "1*" to "*" OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service date: int type: Service place place address: String Variables: {source_class}: side from "1*" to "*" 1. Update the cardinality on the {source_class} side from "*" to "1* 2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks Variables: {source_class}: Service {source_class}: Robot			n exist without any {target class}.		have at least one {target_class}.		
OR 1. Update the cardinality on the {target_class} side from "1*" to "*" 2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) Domain: Car Service OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality to high (0.9) OR 1. Update the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality on the {target_class} side from "*" to "1 2. Set the confidence for the cardinality on the {target_class} side from "*" to "1 3. Update the cardinality on the {target_class} side from "*" to "1 3. Update the cardinality on the {target_class} side from "*" to "1 3. Update the cardinality on the {target_class} side from "*" to "1 3. Update the cardina	Actions						
2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service date: int type: ServiceType lace lace laddress: String Variables: Source_class}: Service Service laddress: String 2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks A Robot String id lace lace lace lace lace lace lace lace			, ,	1	. = , ,		
2. Set the confidence for the cardinality to high (Threshold x 1.1) Domain: Car Service date: int type: ServiceType lace lace laddress: String Variables: Source_class}: Service Service laddress: String 2. Set the confidence for the cardinality to high (0.9) Domain: Robot tasks A Robot String id lace lace lace lace lace lace lace lace		1. Update the cardinality on t	he {target_class} side from "1*" to "*"	1. Update the cardinality on the {target_class} side from "*" to "1*"			
© Service date: int type: ServiceType place date: int type: ServiceType place date: int type: ServiceType							
date: int type: ServiceType c		Domain: Ca	ar Service	Domain: Robot tasks			
date: int type: ServiceType c	© Service		(A) Robot	Robot			
type: ServiceType Description Descripti		date : int	date : int				
Variables: \[\begin{align*} \text{performs} \\ \text{performs} \\ \text{string id} \\ \text{Variables:} \\ \text{Source_class}: Service \end{align*} \] \[\begin{align*} \text{performs} \\ \text{string id} \\ String i		type : ServiceType	type : ServiceType	1	1		
Variables: \[\begin{align*} \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \		* place		performs	performs		
Variables: \[\begin{array}{c} \text{String id} St		1	1	*			
Variables: {source_class}: Service {source_class}: Robot		© Garage	© Garage	€ Task	© Task		
{source_class}: Service {source_class}: Robot		address : String	address : String	String id	String id		
{source_class}: Service {source_class}: Robot		Varial	bles:	Variables:			
	{source cla						
	-			_			
Question Question							
In the context of a garage, does it always involve at least one service, or can it exist In the context of a robot, does it always involve at least one task, or	In the context of a garage, does it always involve at least one service, or can it exist		In the context of a robot, does it always involve at least one task, or				
without any?. For example: can it exist without any?. For example:			•				
1. A garage can exist without any service. 1. A robot can exist without any task.	A garage can exist without any service.		1. A robot can exist without any task.				
2. A garage must have at least one service. 2. A robot must have at least one task.	2. A garage must have at least one service.		2. A robot must have at least one task.				
Answer Options Answer Options		Answer	Options	Answ	er Options		
Choice 1: A garage can exist without any Choice 2: A garage must have at least Choice 1: A robot can exist Choice 2: A robot must have at	Cl : 4 A	A garage can exist without any	Choice 2: A garage must have at least	Choice 1: A robot can exist	Choice 2: A robot must have at		
service. one service. without any task. least one task.	Choice 1: A		= =				

Association	n Class vs Class				
Decision	A concept is a class and asso	ociation at the same time (association clas	ss) or it is only a class.		
Patterns	1. A relationship with man	y to many cardinality on both sides: Alter	native_1: Association class, Alternative_2: Cl	ass	
Template	A {source_class} has multiple something that only exists we specific {association_class}? 1. A {source_class} has a direct that the source th	e {association_class} at different {target_o when a {association_class} happens? Or wo b. For example: ect relationship with a {target_class}. The the key event that establishes the link bet tion_class} occurs.	_class}. Should we think of the link between a {source_class} and a {target_class} as vould you say a {source_class} and a {target_class} and a {target_class} should have a link even outside of e {association_class} is a separate event that connects the {source_class} and etween a {source_class} and a {target_class}. The {source_class} and {target_class} are		
Answers	, , ,	ınd a {target_class} are always	Choice 2: The {source_class} and the {targetass} and the {targetass} and the {targetass} happens.	et_class} have a relationship only when a	
Actions	1. Update class {association_d 2. Set confidence for the asso 3. Create association from {so 4. Update the cardinality to " {target_class} side 5. Set confidence for the asso 6. Set confidence for the card 7. Remove the relationship fro	class} to association class ciation class to high (Threshold x 1.1)	 Update association class {association_class} to class Set confidence for the class to high (Threshold x 1.1) Create association from {source_class} to {association_class} Update the cardinality to "1" in the {source_class} side and "*" in the {association_class} side Create association from { association_class } to {target_class} Update the cardinality to "1" in the {association_class} side and "*" in the {target_class} Set confidence for the associations to low (Threshold / 2) Set confidence for the cardinalities "1" to low (Threshold / 2) Set confidence for the cardinalities "*" to high (Threshold x 1.1) Remove association between {source_class} and {target_class} 		
	Domain: C	ar Service	Domain: Robot tasks		
	© Car o registrationDate : int plate : String Services © Service a date : int b type : Servicetype Carage	© Car □ registrationDate : int □ plate : String © Service □ date : int □ type : ServiceType © Garage □ address : String	String id 1.* performs © TaskExecution String id int startTime Location location © Task String id Involves Invo		
	Varial	bles:	Varia	bles:	
{source_class {target_class} {association_	}: Garage _class}: Service		{source_class}: Robot {target_class}: Task {association_class}: TaskExecution		
	Ques	•			
 A car has multiple services at different garages. Should we think of the relationship between a car and a garage as something that only exists when a service happens? Or would you say a car and a garage should have a connection even outside of specific services?. For example: 1. A car has a direct relationship with a garage. The service is a separate event that connects the car and garage temporarily. 2. The service is the key event that establishes the connection between a car and a 		 A robot has multiple task execution at different tasks. Should we think of the relationship between a robot and a task as something that only exists when a task execution happens? On would you say a robot and a task should have a connection even outside of specific task execution? . For example: 1. A robot has a direct relationship with a task. The task execution is a separate event that connects the robot and task temporarily. 2. The task execution is the key event that establishes the connection between a robot and 			
garage. T	The car and garage are only linked		task. The robot and task are only linked wh		
	Answer (•	Answer	Options	
	car and a garage are always ven when no service happens.	Choice 2: The car and the garage have a relationship only when a service happens.	Choice 1: A robot and a task are always connected, even when no task execution happens.	Choice 2: The robot and the task have a relationship only when a task execution happens.	