Assignment 5

221300079 Juntong Wang

June 30, 2024

1 Question 1

(1) Consider the database instance $\mathcal{D}_{\mathsf{PGBBW}}$ (PGBBW stands for Pleasant Goat and Big Big Wolf) below:

$\mathsf{Sheep(weslie)}$ $\mathsf{Sheep(slowy)}$ $\mathsf{LazySheep(paddi)}$

$\mathsf{Sheep(tibbie)}$ $\mathsf{BrownSheep(fitty)}$ $\mathsf{Sheep(jonie)}$

$\mathsf{Wolf(wolffy)}$ $\mathsf{Wolf(wolnie)}$ $\mathsf{Wolf(wilie)}$

$\mathsf{hasFriend(weslie, slowy)}$ $\mathsf{hasFriend(tibbie, jonie)}$

$\mathsf{hasEnemy(paddi, wolffy)}$ $\mathsf{hasWife(wolffy, wolnie)}$ $\mathsf{hasSon(wolnie, wilie)}$

We query $\mathcal{D}_{\mathsf{PGBBW}}$ under closed world assumption (relational database) and under open world assumption. Recall that under the closed world assumption we consider the interpretation $\mathcal{I} := \mathcal{I}_{\mathcal{D}_{\mathsf{PGBBW}}}$ defined as follows:

$\Delta^{\mathcal{I}} = \{\mathsf{weslie, slowy, paddi, tibbie, fitty, jonie, wolffy, wolnie, wilie}\}$

$\mathsf{Sheep}^{\mathcal{I}} = \{\mathsf{weslie, slowy, tibbie, jonie}\}$

$\mathsf{LazySheep}^{\mathcal{I}} = \{\mathsf{paddi}\}$

$\mathsf{BrownSheep}^{\mathcal{I}} = \{\mathsf{fitty}\}$

$\mathsf{Wolf}^{\mathcal{I}} = \{\mathsf{wolffy, wolnie, wilie}\}$

$\mathsf{hasFriend}^{\mathcal{I}} = \{(\mathsf{weslie, slowy}), (\mathsf{tibbie, jonie})\}$

$\mathsf{hasEnemy}^{\mathcal{I}} = \{(\mathsf{paddi, wolffy})\}$

$\mathsf{hasWife}^{\mathcal{I}} = \{(\mathsf{wolffy, wolnie})\}$

$\mathsf{hasSon}^{\mathcal{I}} = \{(\mathsf{wolnie, wilie})\}$

Consider the following non-Boolean queries F_i:

1. $F_1(x) = \mathsf{Wolf}(x)$

2. $F_2(x) = \neg\mathsf{Sheep}(x)$

3. $F_3(x, y) = \mathsf{hasFriend}(x, y)$

4. $F_4(x) = \mathsf{Sheep}(x) \wedge \neg\mathsf{hasFriend}(x, \mathsf{jonie})$

For each F_i, give

- for closed world assumption: $\mathsf{answer}(F_i, \mathcal{D}_{\mathsf{PGBBW}})$;
- for open world assumption: $\mathsf{certanswer}(F_i, \mathcal{D}_{\mathsf{PGBBW}})$.

(2). Following (1), consider now the TBox $\mathcal{T}$ given as:

$$\begin{aligned} \mathsf{LazySheep} &\sqsubseteq \mathsf{Sheep} \\ \mathsf{LazySheep} \sqcap \mathsf{BrownSheep} &\sqsubseteq \bot \\ \mathsf{Sheep} \sqcap \mathsf{Wolf} &\sqsubseteq \bot \\ \top &\sqsubseteq \forall \mathsf{hasFriend}.\mathsf{Sheep} \\ \exists \mathsf{hasFriend}.\mathsf{Sheep} &\sqsubseteq \mathsf{Sheep} \\ \mathsf{Sheep} &\sqsubseteq \exists \mathsf{hasFriend}.\top \end{aligned}$$

Fill in the table below with the answers "Yes", "No" or "Don't know" to the Boolean queries.

Query	Answer for $\mathcal{I}$	Certain Answer for $\mathcal{D}_{\mathsf{PGBBW}}$	Certain Answer for $(\mathcal{T}, \mathcal{D}_{\mathsf{PGBBW}})$
LazySheep(paddi)	Yes	Yes	Yes
LazySheep(fitty)	No	DK	No
Sheep(paddi)	No	DK	Yes
¬Sheep(paddi)	Yes	DK	No
BrownSheep(wilie)	No	DK	DK
Wolf(fitty)	No	DK	DK
∃hasFriend.⊤(fitty)	No	DK	DK
∀hasFriend.⊤(wolffy)	Yes	Yes	Yes
∀hasFriend.∃hasFriend.⊤(wolnie)	Yes	DK	Yes
∃hasFriend.∀hasFriend.⊤(jonie)	No	DK	Yes

Here is the answer:
(1).For the closed world assumption:
$Answer(F_1, D_{PGBBW}) = \{wolffy, wolnie, wilie\}$
$Answer(F_2, D_{PGBBW}) = \{wolffy, wolnie, wilie, paddi, fitty\}$
$Answer(F_3, D_{PGBBW}) = \{(weslie, slowy), (tibbie, jonie)\}$
$Answer(F_4, D_{PGBBW}) = \{weslie, slowy, jonie\}$
For the open world assumption:
$CetAnswer(F_1, D_{PGBBW}) = \{wolffy, wolnie, wilie\}$
$CetAnswer(F_2, D_{PGBBW}) = \{\emptyset\}$
$CetAnswer(F_3, D_{PGBBW}) = \{(weslie, slowy), (tibbie, jonie)\}$
$CetAnswer(F_4, D_{PGBBW}) = \{\emptyset\}$
(2).The answer is in the box above:(where DK stands for Don't know)

2 Question 2

Computing $\mathcal{I}_{\mathcal{T},\mathcal{A}}$ in $\mathcal{EL}$
Consider the following $\mathcal{EL}$ TBox $\mathcal{T}$:

$$\begin{aligned}
\mathsf{Vocalist} &\sqsubseteq \exists \mathsf{plays_for}.\mathsf{Band} \\
\mathsf{Guitarist} &\sqsubseteq \exists \mathsf{plays_for}.\mathsf{Band} \\
\mathsf{Bassist} &\sqsubseteq \exists \mathsf{plays_for}.\mathsf{Band} \\
\mathsf{Drummer} &\sqsubseteq \exists \mathsf{plays_for}.\mathsf{Band} \\
\mathsf{LeadGuitarist} &\sqsubseteq \mathsf{Guitarist} \\
\mathsf{RhythmGuitarist} &\sqsubseteq \mathsf{Guitarist} \\
\mathsf{Band} &\sqsubseteq \exists \mathsf{captained_by}.\mathsf{Captain} \\
\mathsf{Band} &\sqsubseteq \exists \mathsf{managed_by}.\mathsf{Manager} \\
\mathsf{Manager} &\sqsubseteq \exists \mathsf{managed_by}.\mathsf{Manager}
\end{aligned}$$

and the following ABox $\mathcal{A}$:

$$\begin{aligned}
&\mathsf{Captain}(\mathsf{Monster}) && \mathsf{Vocalist}(\mathsf{Ashin}) \\
&\mathsf{LeadGuitarist}(\mathsf{Monster}) && \mathsf{Bassist}(\mathsf{Masa}) \\
&\mathsf{RhythmGuitarist}(\mathsf{Stone}) && \mathsf{Drummer}(\mathsf{Ming}) \\
&\mathsf{Band}(\mathsf{Mayday}) && \mathsf{managed_by}(\mathsf{Mayday}, \mathsf{Amuse})
\end{aligned}$$

- Compute the interpretation $\mathcal{I}_{\mathcal{T},\mathcal{A}}$ as described in the slides.
- For $\mathcal{EL}$ concept queries, we know that $\mathcal{I}_{\mathcal{T},\mathcal{A}}$ gives the answer "Yes" iff $(\mathcal{T},\mathcal{A})$ gives the certain answer "Yes". Check this for the following queries:
  1. $\exists \mathsf{plays\_for}.\mathsf{Band}(\mathsf{Ashin})$;
  2. $\exists \mathsf{managed\_by}.\mathsf{Manager}(\mathsf{Masa})$;
  3. $\exists \mathsf{plays\_for}.\exists \mathsf{captained\_by}.\mathsf{Captain}(\mathsf{Monster})$;
  4. $\exists \mathsf{plays\_for}.\exists \mathsf{managed\_by}.\mathsf{Manager}(\mathsf{Ming})$.
- For more complex queries, $\mathcal{I}_{\mathcal{T},\mathcal{A}}$ can give the answer "Yes" even if $(\mathcal{T},\mathcal{A})$ does not give the certain answer "Yes". Check this for:
  1. $F(x,y) = \exists z.(\mathsf{plays\_for}(x,z) \wedge \mathsf{plays\_for}(y,z))$.
  2. $F = \exists x.\mathsf{managed\_by}(x,x)$.

Here is the answer:
(1):The initial situation like this:

- $S(d_{Vocalist}) = \{Vocalist\}$
- $S(d_{Guitarist}) = \{Guitarist\}$
- $S(d_{Bassist}) = \{Bassist\}$
- $S(d_{Drummer}) = \{Drummer\}$
- $S(d_{LeadGuitarist}) = \{LeadGuitarist\}$
- $S(d_{RhythmGuitarist}) = \{RhythmGuitarist\}$
- $S(d_{Band}) = \{Band\}$
- $S(d_{Captain}) = \{Captain\}$
- $S(Monster) = \{Captain\}$
- $S(Ashin) = \{Vocalist\}$
- $S(Monster) = \{LeadGuitarist\}$
- $S(Masa) = \{Bassist\}$
- $S(Stone) = \{RhythmGuitarist\}$
- $S(Ming) = \{Drummer\}$
- $S(Mayday) = \{Band\}$
- $S(Amuse) = \{\emptyset\}$
- $R(managed\_by) = \{(Mayday, Amuse)\}$
- $R(captained\_by) = \{\emptyset\}$
- $R(plays\_for) = \{\emptyset\}$

Then, we use the rules in the slides(simpleR):

- $S(d_{LeadGuitarist}) = \{LeadGuitarist, Guitarist\}$
- $S(d_{RhythmGuitarist}) = \{RhythmGuitarist, Guitarist\}$

Then, we use the rules in the slides(rightR):

- $R(managed\_by) = \{(Mayday, Amuse), (d_{Band}, d_{Manager}), (Mayday, d_{Manager}), (d_{Manager}, d_{Manager})\}$
- $R(captained\_by) = \{(d_{Band}, d_{Captain}), (Mayday, d_{Captain})\}$
- $R(plays\_for) = \{(d_{Vocalist}, d_{Band}), (Ashin, d_{Band}), (d_{Guitarist}, d_{Band}), (Monster, d_{Band}),$
  $(Stone, d_{Band}), (d_{Bassist}, d_{Band}), (Masa, d_{Band}), (d_{Drummer}, d_{Band}), (Ming, d_{Band})\}$

So the final we have:

- $\Delta^{I_{T,A}} = \{d_{Vocalist}, d_{Guitarist}, d_{Bassist}, d_{Drummer}, d_{LeadGuitarist}, d_{RhythmGuitarist}, d_{Band}, d_{Captain}, Monster, Ashin, Masa, Stone, Ming, Mayday, Amuse\}$
- $Guitarist^{I_{T,A}} = \{d_{Guitarist}, Monster, Stone\}$
- $Vocalist^{I_{T,A}} = \{d_{Vocalist}, Ashin\}$
- $Bassist^{I_{T,A}} = \{d_{Bassist}, Masa\}$
- $Drummer^{I_{T,A}} = \{d_{Drummer}, Ming\}$
- $LeadGuitarist^{I_{T,A}} = \{d_{LeadGuitarist}, Monster\}$
- $RhythmGuitarist^{I_{T,A}} = \{d_{RhythmGuitarist}, Stone\}$
- $Band^{I_{T,A}} = \{d_{Band}, Mayday\}$
- $Captain^{I_{T,A}} = \{d_{caption}, Monster\}$
- $R(managed\_by) = \{(Mayday, Amuse), (d_{Band}, d_{Manager}), (Mayday, d_{Manager}), (d_{Manager}, d_{Manager})\}$
- $R(captained\_by) = \{(d_{Band}, d_{Captain}), (Mayday, d_{Captain})\}$
- $R(plays\_for) = \{(d_{Vocalist}, d_{Band}), (Ashin, d_{Band}), (d_{Guitarist}, d_{Band}), (Monster, d_{Band}), (Stone, d_{Band}), (d_{Bassist}, d_{Band}), (Masa, d_{Band}), (d_{Drummer}, d_{Band}), (Ming, d_{Band})\}$

(2).For this question:
2.1:∃plays_for.Band(Ashin);
We look for the relation playes-for and check if have (Ashin, Band),here the answer is Yes.
So the certain answer is Yes.

2.2:∃managed_by.Manager(Masa);
We look for the relation managed-by and check if have (Masa, Manager),here the answer is No.(not Yes)
So the certain answer is No.(not Yes)

2.3:∃plays_for.∃captained_by.Captain(Monster);
We look for the relation playes-for and captained-by and check if have $(Monster, d_{Band})$, $(d_{Band}, d_{Captain})$,here the answer is Yes.
So the certain answer is Yes.

2.4:∃plays_for.∃managed_by.Manager(Ming).
We look for the relation playes-for and managed-by and check if have $(Ming, d_{Band})$, $(d_{Band}, d_{Manager})$,here the answer is Yes.
So the certain answer is Yes.

(3).For this question:
3.1.$F(x, y) = \exists z.(\mathsf{plays\_for}(x, z) \wedge \mathsf{plays\_for}(y, z))$.
we could find that every relation in the plays-for, the second part always be the d_{Band}, so every first part could be the answer of F(x,y).
So the answer is a lot which looks like in this form:$(1\ of\ 7\ first\ part, 7\ parts)$.There are 49 elements.
But the certain is $(Ashin, Ashin), (Monster, Monster), (Stone, Stone), (Masa, Masa)$ $(Ming, Ming)$.So could give Yes.

3.2.$F = \exists x.\mathsf{managed\_by}(x, x)$.
In the rule of manager-by, we have $(d_{Manager}, d_{Manager})$ the answer is Yes.
But the certain answer could not be Yes, because we don't the answer in (T,A).

3 Question 3

Consider the following database $\mathcal{D}$ consisting of the following tables:

Person:	
ID	Name
2101	Jay_Chou
2102	Jolin_Tsai
2103	Stefanie_Sun
2104	Ta-yu_Lo

Enrollment:	
StudentID	Since
2102	2020
2103	2021
2104	2020

Attendance:	
StudentID	CourseID
2101	30000160
2102	30000160
2102	30000170
2103	30000150

Course:	
ID	Title
30000150	ML
30000160	KRP
30000170	NLP

- Define the finite first-order interpretation $\mathcal{I}_{\mathcal{D}}$ corresponding to $\mathcal{D}$.
- Reformulate each of the following SQL queries Q into first-order queries f_Q, and identify which of them are conjunctive queries.
- Answer Q in the context of $\mathcal{D}$ and $f_{\mathcal{Q}}$ in the context of $\mathcal{I}_{\mathcal{D}}$.

1. SELECT $*$ FROM Person
2. SELECT Person.Name FROM Person, Attendance, Course
   WHERE Person.ID = Attendance.PersonID
   AND Course.ID = Attendance.CourseID
   AND Course.Title = "KRP"
3. SELECT Person.Name FROM Person, Enrollment
   WHERE Person.ID = Enrollment.PersonID
   AND NOT EXISTS (
   SELECT $*$ FROM Attendance
   WHERE Person.ID = Attendance.PersonID)

Here is the answer:
(1).The I_D is as follow:

- $ID^{I_D} = \{2101, 2102, 2103, 2104, 30000150, 30000160, 30000170\}$
- $Name^{I_D} = \{Jay\_Chou, Jolin\_Tsai, Stefanie\_Sun, Ta-yu\_Lo\}$
- $StudentID^{I_D} = \{2101, 2102, 2103, 2104\}$
- $Since^{I_D} = \{2020, 2021\}$
- $CourseID^{I_D} = \{30000150, 30000160, 30000170\}$
- $Title^{I_D} = \{ML, KRP, NLP\}$
- $Person^{I_D} = \{(2101, Jay\_Chou), (2102, Jolin\_Tsai), (2103, Stefanie\_Sun), (2104, Ta-yu\_Lo)\}$
- $Enrollment^{I_D} = \{(2102, 2020), (2103, 2021), (2104, 2020)\}$
- $Attendance^{I_D} = \{(2101, 30000160), (2102, 30000160), (2102, 30000170), (2103, 30000150)\}$
- $Course^{I_D} = \{(30000150, ML), (30000160, KRP), (30000170, NLP)\}$

(2).The answer of changing the under 3 sequence into f_Q are as followed:
1.$F(x, y) = Person(x, y)$
2.$F(x) = \exists z \exists x(Person(x, y) \wedge Attendance(x, z) \wedge Course(z, KRP))$.
3.$F(x) = \exists z \exists x(Person(x, y) \wedge Enrollment(x, z) \wedge \forall w \neg Attendance(x, w))$
In the following 3 f_Qs we could get that the 1 and 2 are conjunctive query, and the 3 has forall, so it is not the conjunctive query.
(3).When in D context:
1.$\{(2101, Jay\_Chou), (2102, Jolin\_Tsai), (2103, Stefanie\_Sun), (2104, Ta-yu\_Lo)\}$
2.$\{Jay\_Chou, Jolin\_Tsai\}$
3.$\{Ta-yu\_Lo\}$
When in f_Q context:
1.$\{(2101, Jay\_Chou), (2102, Jolin\_Tsai), (2103, Stefanie\_Sun), (2104, Ta-yu\_Lo)\}$
2.$\{Jay\_Chou, Jolin\_Tsai\}$
3.$\{Ta-yu\_Lo\}$

4 Question 4

Consider the following $\mathcal{ALC}$ knowledge base $\mathcal{K} := (\mathcal{T}, \mathcal{A})$ with:

$\mathcal{T}$:=$\{\mathsf{X} \sqsubseteq Y, Y \sqsubseteq \exists r.X, X \sqsubseteq \forall r.Y, \forall r.X \sqsubseteq Y, W \equiv \neg V, \exists r.Y \sqsubseteq \neg V\}$

$\mathcal{A}$:={(Jay_Chou, Jolin_Tsai) : r, (Jolin_Tsai, Stefanie_Sun) : r, (Stefanie_Sun, Jay_Chou) : r, (Jolin_Tsai, Jolin_Tsai) : r, (Stefanie_Sun, Stefanie_Sun) : r, Stefanie_Sun : X}

- Compute the certain answers to the following conjunctive queries in the context of $\mathcal{A}$.
- Compute the certain answers to the following conjunctive queries in the context of $\mathcal{K}$.
    1. $r(x, y) \wedge Y(y)$
    2. $\exists y(r(x, y) \wedge Y(y))$
    3. $\exists x, y(r(x, y) \wedge r(y, x))$
    4. $\exists z, w(r(x, y) \wedge r(y, z) \wedge r(z, x) \wedge r(z, w) \wedge W(w))$

Here is the answer:
First, with context of A:
1.certain answer:$\emptyset$
2.certain answer:$\emptyset$
3.certain answer:$\{(Stefanie\_Sun, Stefanie\_Sun), (Jolin\_Tsai, Jolin\_Tsai)\}$,So it is $\top$
4.certain answer:$\emptyset$

Second, with the context K:
1.certain answer:$(Jolin\_Tsai, Stefanie\_Sun), (Stefanie\_Sun, Stefanie\_Sun), (Stefanie\_Sun, Jay\_Chou)$
2.certain answer:$\{Stefanie\_Sun, Jay\_Chou\}$
3.certain answer:$\{(Stefanie\_Sun, Stefanie\_Sun), (Jolin\_Tsai, Jolin\_Tsai)\}$,So it is $\top$
4.certain answer:$\{(Stefanie\_Sun, Stefanie\_Sun), (Jolin\_Tsai, Jolin\_Tsai), (Jay\_Chou, Jolin\_Tsai), (Jolin\_Tsai, Stefanie\_Sun), (Stefanie\_Sun, Jay_Chou)\}$

5 Question 5

CW5 focuses on implementing an OWL-based querying system. Your primary task involves querying the reasoner for sub-classes, equivalent classes, and instances, then storing these results in a QueryResult object. Within the Java class CW5, you are expected to implement specific methods. Please ensure you do not modify any classes other than CW5, as this is the only class you may submit. To evaluate your implementation, you can utilize the provided classes, App.java and PizzaOrderingSystemApp.java. These classes provide a user-friendly way to verify your code's correctness. You can run these classes in Eclipse (or your preferred IDE) by clicking the small white and green arrow icon when the class is open. For a more direct approach, consider using the pre-implemented unit tests to gauge your progress. Note that passing these unit tests does not guarantee a perfect score, as additional tests will be used to assess your solution more comprehensively. Upon completing the assignment, please compress your CW5.java into a zip file for submission. Good luck!

1. public Set<QueryResult> performQuery (OWLClassExpression exp, Query Type type) {...}
   This method takes in an arbitrary expression in OWL, like "Person and hasBirthYear value 1964" or "hasTopping some MeatTopping" and queries the ontology for the following three types of knowledge:
   EquivalentClasses: you are asked to return all those (named) classes that are equivalent to the expression (exp), using the (already fully initialized) reasoner.
   Sub-classes: return all those classes that are (named) sub-classes of the expression, using the reasoner.
   Instances: return all those individuals that are instances of the expression, using the reasoner.
   Query results are stored in query result objects. These are created for example as follows:
   QueryResult qr = new QueryResult(ind, false, type);
   Note that you need to include the information whether the inference is direct or indirect. Example: Given three classes with A subclass B subclass C, then A is a direct subclass of B and B is a direct subclass of C, but A is an indirect subclass of C (through B!). In order to solve this problem, you will query the reasoner for direct and indirect sub-class separately. After creating the QueryResult, add it to the provided set.

2. public Boolean isValidPizza (OWLClassExpression exp) {...};
   In this method, you check whether the supplied class expression exp is a valid pizza expression, i.e., whether it is inferred to be a Pizza.

3. public Set<QueryResult> filterNamedPizzas (Set<QueryResult> results) {...};
   This question is similar to the one before, only that you are asked to filter from a set of results those that correspond to NamedPizza's, such as Margherita or AmericanHot.

4. public Set<OWLClassExpression> getAllSuperClassExpressions (OWLClass ce) {...};

This question requires a bit of thinking. You are asked to query the ontology to gather all available information about the class ce. In order to get an idea of what "AL" means, you can open Protégé and look at the "Classe" tab. Clicking on a class will reveal its super-classes, equivalent-classes, as well as inherited super-classes (Anonymous Ancestors) and indirect super-classes. Unfortunately, the reasoner will not easily give you access to those. In order to get full marks for this task, it is sufficient to query for all super-classes (direct and indirect, using the reasoner), and somehow find a way to obtain the anonymous super-classes using the ontology directly (that will need a bit of thought, do not despair too quickly). A perfect solution will test, for all sub-expressions in the ontology, whether ce is a sub-concept of it. Attempt this only if you are really confident with the OWL API by now.

6 Question 6

The problem is too long, so here we only give out the answer of the 3 question:

- Show that for every two $\mathcal{L}$-TBoxes $\mathcal{T}_2$, $\mathcal{T}_1 \subseteq \mathcal{T}_2$, and a signature Σ over $\mathcal{L}$, if $\mathcal{T}_2$ is a model Σ-conservative extension of $\mathcal{T}_1$, then it is a also deductive Σ-conservative extension of $\mathcal{T}_1$.
- Show that there exists two $\mathcal{ALC}$-TBoxes $\mathcal{T}_2$, $\mathcal{T}_1 \subseteq \mathcal{T}_2$ such that $\mathcal{T}_2$ is a deductive Σ-conservative extension of $\mathcal{T}_1$ but it is NOT a model conservative extension of $\mathcal{T}_1$.
- Consider the TBox $\mathcal{P}_1$ consisting of the axioms P1—P4 and E1 and the TBox $\mathcal{Q}$ consisting of the axioms M1—M5 from Figure 1. Show that $\mathcal{P}_1 \cup \mathcal{Q}$ is a deductive conservative extension of $\mathcal{Q}$.

Here is the answer
(1).As the problem goes, if we need to prove it is also a $\Sigma-$*conservative extension* of T_1, we need to prove like:
we assume we have a T_2, and T_2 is a Σ conservative extension of T_1.so if we have L-axiom α that $sig(\alpha) \subseteq \Sigma$, so for every model of T_1, we have $I_1 \models \alpha$. So for every model of I_1, if T_2 is a Σ conservative extension of T_1, there must be the model I_2 according to the problem that we could get $I_2 \models T_2$ and also $I_1|\Sigma = I_2|\Sigma$,so for α, there is $I_2 \models \alpha$.
So $T_2 \models \alpha$ if $T_1 \models \alpha$.
Also if we suppose $T_2 \models \alpha$, akso we could find I_2 that $I_2 \models \alpha$,and then we could get the same $I_1 \models T_1$,$I_1|\Sigma = I_2|\Sigma$. So $I_1 \models \alpha, T_1 \models \alpha$.
In the end, we proved that for every two $\mathcal{L}$-TBoxes $\mathcal{T}_2$, $\mathcal{T}_1 \subseteq \mathcal{T}_2$, and a signature Σ over $\mathcal{L}$, if $\mathcal{T}_2$ is a model Σ-conservative extension of $\mathcal{T}_1$, then it is a also deductive Σ-conservative extension of $\mathcal{T}_1$.
(2)Here is a example:$T_1 : \{A \sqsubseteq \exists r.B\}$,$T_2 : \{A \sqsubseteq \exists r.B, B \sqsubseteq \exists r.C\}$,and $\Sigma = \{A, B, r\}$
T_2 is a deductive $\Sigma-$conservative of T_1,this is easy, and T_2 is not a model $\Sigma-$conservative extension of T_1, this is because of C.
(3)For this problem we use the contradictive proving, if the $P_1 \cup Q$ is not a deductive conservative extension of Q.Then we have a α that $Sig(\alpha) \subseteq Sig(Q)$ so that we have $Q \not\models \alpha$ and $Q \cup \models \alpha$.But this is not hold for the axioms P1—P4 and E1,not in $Sig(Q)$.So proved.