University of Kentucky

Creation of Knowledgebase and Reasoning using Datalog for Student Recognition Awards and Prizes System

To: Ajay Sharma, Assistant Deputy Director of Financial Services at the University of Kentucky
From: Amol Shyamsunder Govekar, Hrishikesh Potdar, Jun Liu
Date: 7th November 2021
Subject: Creation of knowledgebase and inference rules for Student Recognition Awards and Prizes System.

We have created knowledgebase assertions in Datalog and we have written four inference rules to reason about "General things" in our system.

Explanation of our knowledge base facts:

We have taken the instance level data from our milestone 2. We have created a group7-ph3.dl file and in that file, we have put the entries that we made in tables for the SQL database created in milestone 2. We have seven assertions that we have created based on the tables from the previous milestone. These assertions/facts are as follows:

1. universityemployee (_, _, _, _). – We have added a total of five employees to our knowledge base. The university employee records the following arguments - Employee_Id, Employee_Type, Cardholding_Status, and Department_Name.
2. approvingofficial (_, _). – We have added records of three approving officials to our knowledge base. The approving official records following arguments - Employee_Id and Approver_Type.
3. nonapprovingofficial (_, _) – We have included two non-approving officials in our knowledge base. It records arguments like Employee_Id and Type.
4. universityfinancialservices (_, _) – We have added three records for university financial services. It records arguments like Request_id and Request_Type.
5. procurementcard (_, _, _, _, _, _) – We have added five procurement cards to our knowledge base. It records arguments like Card_No, Cardholder_Name, Spend_Limit, Validity_Duration, Employee_Id, Request_Id.
6. paymentrequestdocument (_, _) – We have added two records for the payment request document to our knowledge base. It has arguments like Transaction_Id and Submitter_Id.
7. award (_, _, _, _, _, _, _, _, _, _, _, _, _) – We have added seven award entries in our knowledge base. It has arguments like Award_ID, Purpose, Category, Source_Of_Funds, Criteria, Value, Description, Type, Card_No, Approver_Id, Purchase_Type, Prd_Transaction_Id.

Examples of some of the data we have inserted.

universityemployee (1, 'Approving', 'False', 'Information Sciences').
approvingofficial (1, 'Dean').
nonapprovingofficial (3, 'Business Officer').
universityfinancialservices (10000, 'Procurement Card').
procurementcard (1000, 'John', 200, '15th Oct 2023', 1, 10000).
paymentrequestdocument (1000001, 3).
award (100, 'Social Development', 'Unrestricted General funds', 'Restricted', 'Academics', 60, 'This award has been granted', 'gift', 1000, 1, 'Card', NULL).

Explanation of Inference Rules.

In milestone two, we implemented three SQL queries. We have tried to create a total of four inference rules and of these, two are based on the SQL queries from the previous milestone. Below is a detailed explanation for each of them.

Inference Rule 1 (Based on SQL query)

Requirement - Client wants Card Number, Cardholder Name, and approver type of employees where cardholding status is true.
Inference Rule – cardinfo (X, Y, Z): -
procurementcard (X, Y, _, _, E, _),
approvingofficial (E, Z),

universityemployee (E, _, 'True', _).

In the above inference rule, we have used AND clause between procurementcard, approvingofficial and universityemployee. To fulfill the client's requirement, we have got records from universityemployee where cardholding status is true, and we are subjecting it to AND operation with approvingofficial and procurementcard based on Employee_Id. The inference rule cardinfo has three arguments: X: Card Number, Y: Cardholder Name and, Z: Approval Type. Thus, this inference rule has arity 3.

Output - DES> cardinfo (X, Y, Z)

{
 cardinfo (1003,'Annie','Provost')
}

Info: 1 tuple computed**.**

Inference Rule 2

Requirement - List the card numbers of cards owned by university employees belonging to 'Information Sciences' department.

Inference Rule – listcardnumbers (X, Y): -
universityemployee (X, _, _, 'Information Sciences'),
procurementcard (_, Y, _, _, X, _).

In the above inference rule, we have AND clause on facts/assertions universityemployee & procurementcard. To fulfill client's requirement, we have considered only those university employees which belong to 'Information Sciences' department and we have subjected it to AND operation with procurementcard based on Employee_Id. The inference rule listcardnumbers has two arguments: X: Employee_Id, Y: Name. Thus, this inference rule has arity 2.

Output- DES> listcardnumbers (X, Y)

{
 listcardnumbers (1, 'John'),
 listcardnumbers (4, 'Annie'),
 listcardnumbers (5, 'Owen')
}

Info: 3 tuples computed.

Inference Rule 3

Requirement - Find employee name, type and validity expiration dates of procurement card owned by non- approving officials.

Inference Rule – expirationdates (X, Y, Z, W): -
nonapprovingofficial (X, Y),
procurementcard (_, Z, _, W, X, _).

In the above inference rule, we have AND clause on facts/assertions nonapprovingofficial & procurementcard. To fulfill the client's requirement, we have subjected nonapprovingofficial and procurementcard to AND operation based on Employee_Id. The inference rule expirationdates has four arguments: X: Employee_Id, Y: Type, Z: Name, and W: Validity_Duration. Thus, this inference rule has arity 4.

Output- DES> expirationdates (X, Y, Z, W)

{
 expirationdates (3, 'Business Officer', 'Ursula', '20th Sep 2024'),
 expirationdates (5, 'Supervisor', 'Owen', '08th Apr 2024')
}

Info: 2 tuples computed.

Inference Rule 4 (Based on SQL query)

Requirement - Client wants award ids of those awards (purchased by card) whose value is greater than average award value in database.

Inference Rule – awardinfo (X, Y, Z, W): -

```
award (X, _, _, _, _, Y, _, _, _, W, 'Card', _),
approvingofficial (W, Z).
```

We faced some challenges while designing this inference rule. It is based on the SQL query from our milestone 2 submission. To meet the requirement, we had written a nested SQL query and the inner query had an aggregate function. We found it difficult to incorporate the nested part of the query syntactically using Datalog Inference rules system. So, we decided to simplify our requirement and wrote a query that will return all the award ids purchased by the procurement card. The inference rule awardinfo has four arguments: X: Award_Id, Y: Value, W: Validity_Duration, and Z: Approver Type and. Thus, this inference rule has arity 4.

Output -

```
Info: Processing:
  awardinfo (X, Y, Z, W): -
    award (X, _, _, _, _, Y, _, _, _, W, 'Card', _),
    approvingofficial (W, Z).
{
  awardinfo (100, 60, 'Dean', 1),
  awardinfo (101, 30, 'President', 2),
  awardinfo (102, 40, 'Dean', 1),
  awardinfo (103, 80, 'Provost', 4),
  awardinfo (104, 35, 'President', 2)
}
Info: 5 tuples computed.
```

Inference Rule 5 (which we found impossible to design using Datalog)

Requirement - Client wants to get the cardholder's name for the card that has the highest expenditure ratio i.e., Award Value/Spend Limit.

Solution using SQL:

```
Select pc.Cardholder_Name, a.Card_No, pc.Employee_Id, (SUM(a.Value)*100/pc.Spend_Limit) as
Percentage_Expenditure FROM AWARD a  INNER JOIN PROCUREMENT_CARD pc ON a.Card_No = pc.Card_No AND
a.Purchase_type = "Card" GROUP BY a.Card_No  ORDER BY Percentage_Expenditure DESC LIMIT 1
```

As we can see, the above query included an inner join along with group by and order by statements, coupled with aggregate function value as a projection. We found it impossible to design such a complex query in Datalog as we didn't know how to incorporate group by and order by statements in Datalog.

Important Controversies in Design and how we addressed it:

1. A controversy arose while designing new inference rules. Initially, we decided to go ahead with rules which didn't include any AND/OR clauses. But later, we came to the consensus that we can add little complexity to our inference rules, thus we incorporated multiple clauses (AND/OR) in our rules.
2. A difficulty arose while implementing nested queries from our previous milestone. We researched extensively to find a solution but finally went ahead with simplifying the query and getting the output. Other than this we also faced several issues with spaces in the group7-ph3.dl file. Because of this, we encountered many syntax errors which we were able to resolve once we shifted from notepad to notepad++ for editing.

Contributions of each group member:

During our group meetings, we collectively analyzed the milestone requirements, worked on writing the Memo and group3-ph3.dl file.

Amol S Govekar: Amol worked on creating the knowledgebase and resolving initial errors with executing in Datalog.
Hrishikesh Potdar: Hrishikesh worked on designing and executing inference rules and debugged the errors encountered.
Jun Liu: Jun wrote the initial draft of inference rules based on previous milestone queries.